Lecture Notes in Computer Science 7778

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

Kaoru Kurosawa Goichiro Hanaoka (Eds.)

Public-Key Cryptography – PKC 2013

16th International Conference
on Practice and Theory in Public-Key Cryptography
Nara, Japan, February 26 – March 1, 2013
Proceedings

Springer

Volume Editors

Kaoru Kurosawa
Ibaraki University
Department of Computer and Information Sciences
4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan
E-mail: kurosawa@mx.ibaraki.ac.jp

Goichiro Hanaoka
National Institute of Advanced Industrial Science and Technology (AIST)
Research Institute for Secure Systems (RISEC)
1-1-1 Umezono, Tsukuba, Ibaraki 305-8568, Japan
E-mail: hanaoka-goichiro@aist.go.jp

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-36361-0 e-ISBN 978-3-642-36362-7
DOI 10.1007/978-3-642-36362-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013930237

CR Subject Classification (1998): E.3, K.6.5, E.4, K.4.4, C.2.0, D.4.6, J.1

LNCS Sublibrary: SL 4 – Security and Cryptology

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

Printed on acid-free paper

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

Preface

PKC 2013 was held at the Nara Prefectual New Public Hall in Nara, Japan, during February 26–March 1, 2013. The conference was sponsored by the International Association for Cryptologic Research (IACR).

The conference received 97 submissions (from which one submission was withdrawn), and each submission was reviewed by at least three of the 30 Program Committee members. Submissions co-authored by the Program Committee members were reviewed by at least five committee members. Committee members were allowed to submit at most one paper, or two if the second one was co-authored by a student.

Due to the large number of high-quality submissions, the review process was a challenging and hard task. After 11 weeks of extensive discussions, the Program Committee selected 28 submissions for presentation. The program also included two invited talks: "Functional Encryption: Origins and Recent Developments" given by Brent Waters, and "Techniques for Efficient Secure Computation Based on Yao's Protocol" given by Yehuda Lindell. On behalf of the Program Committee, I would like to thank Brent and Yehuda for accepting our invitation.

There are many people who contributed to the success of PKC 2013. I would like to thank many authors from all over the world for submitting their papers. I am deeply grateful to the Program Committee for their hard work to ensure that each paper received a thorough and fair review. I gratefully acknowledge the external reviewers listed on the following pages. The committee's work was tremendously simplified by Shai Halevi's submission/review software. Finally many thanks go to the General Chair, Goichiro Hanaoka, for organizing the conference.

March 2013 Kaoru Kurosawa

Organization

General Chair

Goichiro Hanaoka	National Institute of Advanced Industrial Science and Technology (AIST), Japan

Program Chair

Kaoru Kurosawa	Ibaraki University, Japan

Local Organizing Co-chairs

Takeshi Chikazawa	Information-technology Promotion Agency (IPA), Japan
Ryo Nojima	National Institute of Information and Communications Technology (NICT), Japan

Program Committee

Nuttapong Attrapadung	AIST, Japan
David Cash	Rutgers University, USA
Jean-Sebastien Coron	University of Luxembourg, Luxembourg
Jintai Ding	University of Cincinnati, USA
Stefan Dziembowski	Uniwersytet Warszawski, Poland and Università di Roma "La Sapienza", Italy
Marc Fischlin	Technische Universität Darmstadt, Germany
Pierre-Alain Fouque	École Normale Supérieure, France
Steven Galbraith	Auckland University, New Zealand
Rosario Gennaro	City College of New York, USA
Dov Gordon	Applied Communication Sciences, USA
Shai Halevi	IBM Research, USA
Carmit Hazay	Bar-Ilan University, Israel
Tibor Jager	Ruhr-Universität Bochum, Germany
Antoine Joux	DGA and Université de Versailles Saint-Quentin-en-Yvelines, France
Eike Kiltz	Ruhr-Universität Bochum, Germany
Noboru Kunihiro	The University of Tokyo, Japan
Kaoru Kurosawa	Ibaraki University, Japan
Allison Lewko	Microsoft Research, USA
Benoit Libert	Technicolor, France
Alexander May	Ruhr-Universität Bochum, Germany

David Naccache	École Normale Supérieure, France
Tatsuaki Okamoto	NTT Labs, Japan
Claudio Orlandi	Aarhus University, Denmark
Chris Peikert	Georgia Institute of Technology, USA
Ludovic Perret	UPMC/INRIA, France
Nigel Smart	University of Bristol, UK
Tsuyoshi Takagi	Kyushu University, Japan
Katsuyuki Takashima	Mitsubishi Electric, Japan
Vinod Vaikuntanathan	University of Toronto, Canada
Hoeteck Wee	George Washington University, USA

Steering Committee

Ronald Cramer	CWI, The Netherlands
Yvo Desmedt	The University of Texas at Dallas, USA
Hideki Imai	Chuo University, Japan
David Naccache	École Normale Supérieure, France
Tatsuaki Okamoto	NTT Labs, Japan
David Pointcheval	École Normale Supérieure, France
Moti Yung	Google and Columbia University, USA
Yuliang Zheng	University of North Carolina, USA

External Reviewers

Masayuki Abe	Craig Costello	Akinori Kawachi
Roberto Araujo	Giovanni Di Crescenzo	Yutaka Kawai
Gilad Asharov	Christophe Doche	Marcel Keller
John Baena	Léo Ducas	Aggelos Kiayias
Rana Barua	Konrad Durnoga	David Kohel
Mihir Bellare	Sebastian Faust	Hugo Krawczyk
David Bernhard	Nelly Fazio	Fabien Laguillaumie
Nir Bitansky	Dario Fiore	Fagen Li
Olivier Blazy	Robert Fitzpatrick	Joseph Liu
Colin Boyd	David Mandell Freeman	Zhen Hua Liu
Elette Boyle	Georg Fuchsbauer	Patrik Longa
Zvika Brakerski	Jun Furukawa	Adriana Lopez-Alt
Christina Brzuska	Sanjam Garg	Vadim Lyubashevsky
Jan Camenisch	Sergey Gorbunov	Mark Manulis
Angelo De Caro	Jens Groth	Ben Martin
Sanjit Chatterjee	Takuya Hayashi	Takahiro Matsuda
Jie Chen	Gottfried Herold	Payman Mohassel
Ashish Choudhury	Dennis Hofheinz	Daisuke Moriyama
Sherman Chow	William Skeith III	Ciaran Mullan
Ran Cohen	Sorina Ionica	Ryo Nishimaki

Adam O'Neill
Dan Page
Jiaxin Pan
Valerio Pastro
Kenny Paterson
Arpita Patra
Thomas Peters
Christophe Petit
Le Trieu Phong
Krzysztof Pietrzak
David Pointcheval
Mike Rosulek
Dominique Schroeder
Jacob Schuldt
Jae Hong Seo
Victor Shoup
Thomas Sirvent
Ron Steinfeld
Mario Strefler
Koutarou Suzuki
Chendong Tao
Boaz Tsaban
Alexander Ushakov
Daniele Venturi
Frederik Vercauteren
Damien Vergnaud
Bogdan Warinschi
Daniel Wichs
Douglas Wikström
David Wilson
Keita Xagawa
Shota Yamada
Zheng Yang
Takanori Yasuda
Arkady Yerukhimovich
Kazuki Yoneyama
Reo Yoshida
Ching-hua Yu
Jean-Christophe Zapalowicz
Hila Zarosim
Hui Zhang
Mingwu Zhang
Rui Zhang
Youwen Zhu
Angela Zottarel

Sponsors

International Association for Cryptologic Research (IACR), National Institute of Advanced Industrial Science and Technology (AIST), Japan, Information-technology Promotion Agency, Japan (IPA), and National Institute of Information and Communications Technology (NICT), Japan

Table of Contents

Homomorphic Encryption

Packed Ciphertexts in LWE-Based Homomorphic Encryption 1
Zvika Brakerski, Craig Gentry, and Shai Halevi

Feasibility and Infeasibility of Adaptively Secure Fully Homomorphic Encryption 14
Jonathan Katz, Aishwarya Thiruvengadam, and Hong-Sheng Zhou

Chosen Ciphertext Secure Keyed-Homomorphic Public-Key Encryption 32
Keita Emura, Goichiro Hanaoka, Go Ohtake, Takahiro Matsuda, and Shota Yamada

Invited Talk (1)

Functional Encryption: Origins and Recent Developments 51
Brent Waters

Primitives

Vector Commitments and Their Applications 55
Dario Catalano and Dario Fiore

Efficient, Adaptively Secure, and Composable Oblivious Transfer with a Single, Global CRS 73
Seung Geol Choi, Jonathan Katz, Hoeteck Wee, and Hong-Sheng Zhou

Cryptography Using Captcha Puzzles 89
Abishek Kumarasubramanian, Rafail Ostrovsky, Omkant Pandey, and Akshay Wadia

Improved Zero-Knowledge Proofs of Knowledge for the ISIS Problem, and Applications 107
San Ling, Khoa Nguyen, Damien Stehlé, and Huaxiong Wang

Functional Encryption/Signatures

Decentralized Attribute-Based Signatures 125
Tatsuaki Okamoto and Katsuyuki Takashima

On the Semantic Security of Functional Encryption Schemes 143
Manuel Barbosa and Pooya Farshim

Attribute-Based Encryption with Fast Decryption 162
Susan Hohenberger and Brent Waters

On RSA

Recovering RSA Secret Keys from Noisy Key Bits with Erasures and Errors 180
Noboru Kunihiro, Naoyuki Shinohara, and Tetsuya Izu

Combined Attack on CRT-RSA: Why Public Verification Must Not Be Public? 198
Guillaume Barbu, Alberto Battistello, Guillaume Dabosville, Christophe Giraud, Guénaël Renault, Soline Renner, and Rina Zeitoun

IBE and IPE

Revocable Identity-Based Encryption Revisited: Security Model and Construction 216
Jae Hong Seo and Keita Emura

Improved (Hierarchical) Inner-Product Encryption from Lattices 235
Keita Xagawa

Invited Talk (2)

Techniques for Efficient Secure Computation Based on Yao's Protocol 253
Yehuda Lindell

Key Exchange

Non-Interactive Key Exchange 254
Eduarda S.V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson

Efficient UC-Secure Authenticated Key-Exchange for Algebraic Languages 272
Fabrice Ben Hamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud

Signature Schemes I

Tighter Reductions for Forward-Secure Signature Schemes 292
Michel Abdalla, Fabrice Ben Hamouda, and David Pointcheval

Tagged One-Time Signatures: Tight Security and Optimal Tag Size 312
Masayuki Abe, Bernardo David, Markulf Kohlweiss, Ryo Nishimaki, and Miyako Ohkubo

Encryption

Key Encapsulation Mechanisms from Extractable Hash Proof Systems, Revisited 332
Takahiro Matsuda and Goichiro Hanaoka

Robust Encryption, Revisited 352
Pooya Farshim, Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia

Sender-Equivocable Encryption Schemes Secure against Chosen-Ciphertext Attacks Revisited 369
Zhengan Huang, Shengli Liu, and Baodong Qin

Signature Schemes II

Efficient Completely Context-Hiding Quotable and Linearly Homomorphic Signatures 386
Nuttapong Attrapadung, Benoît Libert, and Thomas Peters

Verifiably Encrypted Signatures with Short Keys Based on the Decisional Linear Problem and Obfuscation for Encrypted VES 405
Ryo Nishimaki and Keita Xagawa

Sequential Aggregate Signatures with Short Public Keys: Design, Analysis and Implementation Studies 423
Kwangsu Lee, Dong Hoon Lee, and Moti Yung

New Constructions and Applications of Trapdoor DDH Groups 443
Yannick Seurin

Protocols

Rate-Limited Secure Function Evaluation: Definitions and Constructions 461
Özgür Dagdelen, Payman Mohassel, and Daniele Venturi

Verifiable Elections That Scale for Free 479
Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn

On the Connection between Leakage Tolerance and Adaptive Security 497
Jesper Buus Nielsen, Daniele Venturi, and Angela Zottarel

Author Index 517

Packed Ciphertexts in LWE-Based Homomorphic Encryption

Zvika Brakerski[1], Craig Gentry[2], and Shai Halevi[2]

[1] Stanford University
[2] IBM Research

Abstract. In this short note we observe that the Peikert-Vaikuntanathan-Waters (PVW) method of packing many plaintext elements in a single Regev-type ciphertext, can be used for performing SIMD homomorphic operations on packed ciphertext. This provides an alternative to the Smart-Vercauteren (SV) ciphertext-packing technique that relies on polynomial-CRT. While the SV technique is only applicable to schemes that rely on ring-LWE (or other hardness assumptions in ideal lattices), the PVW method can be used also for cryptosystems whose security is based on standard LWE (or more broadly on the hardness of "General-LWE").

Although using the PVW method with LWE-based schemes leads to worse asymptotic efficiency than using the SV technique with ring-LWE schemes, the simplicity of this method may still offer some practical advantages. Also, the two techniques can be used in tandem with "general-LWE" schemes, suggesting yet another tradeoff that can be optimized for different settings.

1 Introduction

Homomorphic Encryption (HE) [Gen09] supports arbitrarily computation on encrypted data, even by parties that do not have the secret decryption key. Despite rapid recent advances [BV11b, BV11a, BGV12, Bra12], HE is still quite expensive. In a nutshell, this is because security considerations dictate that the ciphertexts be large, making homomorphic operations slow (as they have to manipulate these large ciphertexts). The main technique for dealing with this problem is to work with *packed ciphertexts*, namely ciphertexts that encrypt a vector of plaintext values, not just a single value. Homomorphic operations are applied to these vectors component-wise in a SIMD fashion (Single Instruction Multiple Data). Smart and Vercauteren described a ciphertext-packing technique based on polynomial-CRT [SV11], and Gentry et al. [GHS12a] used that technique to achieve a nearly optimal homomorphic evaluation (upto polylogarithic factors). These techniques rely on working over polynomial rings, and their security is based on the assumed hardness of problems in ideal lattices (such as ring-LWE [LPR10]).

Although Brakerski et al. [BV11a, BGV12, Bra12] describe also how to apply homomorphic operations to the Regev cryptosystem [Reg09] (whose security is based on standard LWE), and Peikert et al. [PVW08] show how to pack many plaintext elements in one Regev ciphertext, so far the literature does not contain a description of how to use PVW packed ciphertexts to perform SIMD-type operations. Applying basic homomorphic operations to PVW packed ciphertexts is straightforward, but applying

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 1–13, 2013.

the re-linearizion technique from [BV11a] (which is needed to get more than a constant-degree homomorphism) takes some care. In this short note we describe how this is done, and discuss some practical considerations regarding using this technique.

1.1 Overview

Recall that in Regev's cryptosystem [Reg09] there is a system parameter $q \in Z$, plaintexts are bits, and secret keys and ciphertexts are vectors in $\mathbb{Z}^n$. Decryption of ciphertext $\mathbf{c}$ with secret key $\mathbf{s}$ is done by taking the inner product, reducing modulo q to the interval $[-q/2, +q/2)$, then outputting 0 if the result is smaller than $q/4$ in magnitude and outputting 1 otherwise. In a few more details, the integer $z = \langle \mathbf{s}, \mathbf{c} \rangle$ is of the form $z = k \cdot q + b \cdot \frac{q}{2} + e$, where b is the plaintext bit and k, e, are small integers (and where $\langle \cdot, \cdot \rangle$ denotes inner-product).

Homomorphic operations for Regev's cryptosystem. It is well known that Regev's cryptosystem supports additive homomorphism (for "appropriate choice" of parameters), just by adding the ciphertext vectors modulo q. Moreover, Brakerski and Vaikuntanathan observed in [BV11a] that it can be made to support also multiplicative homomorphism, using tensor products.[1] For two vectors $\mathbf{c}_1, \mathbf{c}_2$, denote by $\mathbf{c}_1 \otimes \mathbf{c}_2$ the tensor product of $\mathbf{c}_1$ and $\mathbf{c}_2$ (arranged as a vectors). That is, $\mathbf{c}_1 \otimes \mathbf{c}_2$ has dimension n^2, with each entry obtained as a product of one entry from $\mathbf{c}_1$ by one entry from $\mathbf{c}_2$. It is easy to see that for four vectors $\mathbf{s}_1, \mathbf{s}_2, \mathbf{c}_1, \mathbf{c}_2$, we always have $\langle \mathbf{s}_1, \mathbf{c}_1 \rangle \cdot \langle \mathbf{s}_2, \mathbf{c}_2 \rangle = \langle \mathbf{s}_1 \otimes \mathbf{s}_2, \mathbf{c}_1 \otimes \mathbf{c}_2 \rangle$.

Given two ciphertext vectors $\mathbf{c}_1, \mathbf{c}_2$, that encrypt the bits b_1, b_2 (relative to secret key $\mathbf{s}$), we construct a product ciphertext by first computing $\mathbf{c}_1 \otimes \mathbf{c}_2$ (over the integers), then scaling down by a $(2/q)$-factor and rounding to the nearest integer vector. Namely, $\mathbf{c}^* \leftarrow \left\lfloor \frac{2}{q} \cdot (\mathbf{c}_1 \otimes \mathbf{c}_2) \right\rceil$. It can be seen that for "appropriate choice" of parameters, if $z_i = \langle \mathbf{s}, \mathbf{c}_i \rangle$ is of the form $z_i = k_i \cdot q + b_i \cdot \frac{q}{2} + e_i$ for $i = 1, 2$ (with $\mathbf{s}$, k_i and e_i small enough), then $z^* = \langle \mathbf{s} \otimes \mathbf{s}, \mathbf{c}^* \rangle$ is of the form $z^* = k^* \cdot q + b_1 b_2 \cdot \frac{q}{2} + e^*$ where k^*, e^* also rather small. Hence the product ciphertext $\mathbf{c}^*$ can be decrypted using the tensor product $\mathbf{s} \otimes \mathbf{s}$ to the product of the plaintext bits $b_1 b_2$.

Of course, using only tensor product would have led to dimension explosion: the dimension of $\mathbf{c}^*$ is n^2, if two such ciphertexts are multiplied then the dimension becomes n^4, etc. To overcome this problem, Brakerski and Vaikuntanathan introduced in [BV11a] a re-linearization technique that shrinks the dimension from n^2 back to n. In fact that technique is more general: given any two keys $\mathbf{s}$ and $\mathbf{s}'$, one can generate a key-switching gadget that can be added to the public key, allowing conversion of ciphertexts relative to $\mathbf{s}'$ to ciphertexts relative to $\mathbf{s}$. (Roughly speaking, the key-switching gadget consists of an encryption of $\mathbf{s}$ under $\mathbf{s}'$, which is a matrix because encrypting each entry in $\mathbf{s}$ takes a vector). Re-linearization is then obtained by putting in the public key a switching gadget from $(\mathbf{s} \otimes \mathbf{s})$ to $\mathbf{s}$.

[1] The observation in [BV11a] was initially made about a slightly different scheme that encodes the plaintext in the least significant bit, but here we use the variant of Brakerski [Bra12] that recovers the original form of Regev's cryptosystem.

Packing Regev ciphertexts. In [PVW08], Peikert et al. observed (roughly) that the same Regev ciphertext vector $\mathbf{c}$ can be used to encrypt many bits, by having many secret-key vectors. Decrypting the i'th bit is done by applying the same decryption procedure as above, using the i'th secret-key. Putting all these secret keys $\mathbf{s}_i$ as the rows of a matrix S, we have a ciphertext $\mathbf{c}$ encrypting a plaintext bit vector $\mathbf{b}$ if the integer vector $\mathbf{z} = S \cdot \mathbf{c}$ is of the form $\mathbf{z} = \mathbf{k} \cdot q + \mathbf{b} \cdot \frac{q}{2} + \mathbf{e}$ for some small integer vector $\mathbf{k}$ and $\mathbf{e}$.

Computing on packed ciphertexts. Since a packed Regev ciphertext as above is essentially the same as a standard ciphertext (except viewed relative to several different keys), then the basic homomorphic operation still work as before, i.e., homomorphic addition by adding ciphertexts (mod q) and homomorphic multiplication via tensor products.

Applying re-linearization to packed ciphertexts takes a little care, however. Although for each i we can put in the public key a switching gadget from $(\mathbf{s}_i \otimes \mathbf{s}_i)$ to $\mathbf{s}_i$, this will still not give us what we need: If $\mathbf{c}^*$ is a packed high-dimension ciphertext that for each i encrypts the product $b_i b'_i$ relative to the high-dimension key $\mathbf{s}_i \otimes \mathbf{s}_i$, then using all the $(\mathbf{s}_i \otimes \mathbf{s}_i)$-to-$\mathbf{s}_i$ gadgets will only yield a collection of non-packed ciphertexts, the i'th of which encrypts $b_i b'_i$ relative to $\mathbf{s}_i$. Instead, we need *a single* key-switching gadget, that simultaneously does all the translations $(\mathbf{s}_i \otimes \mathbf{s}_i)$-to-$\mathbf{s}_i$.

To this end, we recall that a $(\mathbf{s}_i \otimes \mathbf{s}_i)$-to-$\mathbf{s}_i$ key-switching gadget is roughly an encryption of $(\mathbf{s}_i \otimes \mathbf{s}_i)$ under $\mathbf{s}_i$. We thus will use packed ciphertexts also for this gadget, obtaining a single matrix that encrypts $(\mathbf{s}_i \otimes \mathbf{s}_i)$ under $\mathbf{s}_i$, simultaneously for all the i's. This gives us exactly what we need, letting us translate a packed ciphertext relative to the keys $(\mathbf{s}_i \otimes \mathbf{s}_i)$ to another packed ciphertext relative to the $\mathbf{s}_i$'s.

Other homomorphic operations. The above techniques are sufficient to implement SIMD-type homomorphic computation, where we compute the same function over many different inputs at once. However, we would like to use the techniques of Gentry et al. from [GHS12a] to also get efficient evaluation of a single copy. For that purpose, we need to be able to move plaintext elements between slots. For example, we need a way to transform a ciphertext that encrypts a vector $\mathbf{b}$ into another ciphertext that encrypts a cyclic shift of $\mathbf{b}$ (or any other known permutation of the entries of $\mathbf{b}$).

Moving elements between slots turns out to be very easy for this ciphertext packing method: To implement a permutation π over the slots of the plaintext vector, all we need is a packed ciphertext matrix, encrypting the $\mathbf{s}_{\pi(i)}$ under $\mathbf{s}_i$ simultaneously for all i. This is an advantage of the PVW packing method over the SV packing method using polynomial-CRT: In the SV method, plaintext slots movements are implemented using automorphisms, but only a small set of permutations can be implemented this way, hence additional work is needed to implement general permutations from this limited set (see details in [GHS12a]). In the PVW method, any permutation can be implemented directly by adding to the public key a corresponding key-switching gadget.

2 Background

Notations. We denote scalars by lower-case letters ($a, b, \ldots$), vectors by lower-case bold letters ($\mathbf{a}, \mathbf{b}, \ldots$), and matrices by upper-case bold letters ($\mathbf{A}, \mathbf{B}, \ldots$). We denote the Euclidean, l_1, and l_∞ norms of a vector by $\|\mathbf{v}\|$, $\|\mathbf{v}\|_1$, $\|\mathbf{v}\|_\infty$, respectively.

For an integer q we identify $\mathbb{Z}_q$ with the set of representatives from the interval $[-\frac{q}{2}, +\frac{q}{2})$, and denote by $[a]_q$ the reduction of a modulo q into this interval.[2] By $\lceil a \rfloor$ we denote the rounding of the rational a to the nearest integer, and $\lceil a \rfloor_q$ is a shorthand for $[\lceil a \rfloor]_q$. These notations are extended to vectors and matrices in the natural way.

2.1 Learning with Errors (LWE)

The LWE problem was introduced by Regev [Reg09] as a generalization of "learning parity with noise". This problem has parameters the security parameter n, another integer $q \geq \text{poly}(n)$, and a probability distribution χ on $\mathbb{Z}_q$, that outputs integers of magnitude much smaller than q with overwhelming probability. (Typically, χ is a discrete Gaussian distribution with zero mean and standard deviation q/β for some parameter $\beta = \text{poly}(n)$.)

The search version of this problem is to discover a "hidden" vector $\mathbf{s}$ given polynomially many samples of the form $(\mathbf{a}_i, b_i)$, where the $\mathbf{a}_i$'s are chosen at random in $\mathbb{Z}_q^n$, some "error terms" $e_i \leftarrow \chi$ are drawn from χ, and the b_i's are set as $b_i = [\langle \mathbf{s}, \mathbf{a}_i \rangle + e_i]_q$.

The decision variant of the LWE problem is to distinguish a sequence of such pairs $\{(\mathbf{a}_i, [\langle \mathbf{s}, \mathbf{a}_i \rangle + e_i]_q)\}_i$ from a sequence of uniform random pairs in $\mathbb{Z}_q^n \times \mathbb{Z}_q$. As defined by Regev, the hidden vector $\mathbf{s}$ is chosen uniformly at random in $\mathbb{Z}_q^n$, but Applebaum et al. proved in [ACPS09] that this is equivalent (in terms of hardness) to the variant where $\mathbf{s}$ is chosen from the error distribution, $\mathbf{s} \leftarrow \chi^n$. It is this latter variant that we use for homomorphic encryption.

Evidence for the hardness of the LWE problem follows from results of Regev [Reg09] who gave *quantum* reductions from approximation of certain problems on n-dimensional lattices in the worst case to solving LWE with some Gaussian error distributions, and Peikert [Pei09] who gave *classical* reductions for some other problems with similar parameters.

2.2 Regev's Cryptosystem

Regev described in [Reg09] a public-key encryption scheme whose security relies on the hardness of decision-LWE. To simplify the presentation, here we describe this cryptosystem as a symmetric (shared-key) encryption scheme. (Since this scheme supports homomorphic computation, one can use generic transformations to obtain a public-key scheme, see, e.g., [Rot11].) Below we also assume for simplicity that q is even, and concentrate on the case where the plaintext space is $\mathbb{Z}_2 = \{0, 1\}$. The extensions to larger plaintext spaces and arbitrary moduli q are straightforward. Also, below we denote the security parameter by n' and let $n = n' + 1$.

In this symmetric-key variant, the secret key is the LWE hidden vector, chosen (say) as $\mathbf{s}' \leftarrow \chi^{n'}$. Encrypting a bit σ is done by choosing a vector $\mathbf{a} \in \mathbb{Z}_q^{n'}$ uniformly at random and a small error term $e \leftarrow \chi$, setting $b = [\sigma\frac{q}{2} - \langle \mathbf{s}', \mathbf{a} \rangle + e]_q$, and outputting $(b, \mathbf{a})$. To decrypt, compute $d = [b + \langle \mathbf{s}', \mathbf{a} \rangle]_q$ and output 1 if d has magnitude more than $q/4$ and 0 otherwise. Decryption succeeds because the error term e had magnitude smaller than $q/4$ (as it is chosen from the error distribution) and we have $d = [\sigma\frac{q}{2} + e]_q$.

[2] The only exception is $\mathbb{Z}_2$, which we identify with $\{0, 1\}$ rather than $\{-1, 0\}$.

Considering the n-vectors $\mathbf{s} = (1|\mathbf{s}')$ and $\mathbf{c} = (b|\mathbf{a})$, the integer d is just the inner product $\langle \mathbf{s}, \mathbf{c} \rangle$ modulo q, and decryption can be expressed using the formula $\sigma = \lceil [\langle \mathbf{s}, \mathbf{c} \rangle]_q \,/\, (q/2) \rfloor_2$. Also, since $\mathbf{s}'$ was chosen from the error distribution χ and therefore has entries much smaller than q, then the integer $\langle \mathbf{s}, \mathbf{c} \rangle$ (without reduction modulo q) has magnitude at most $q/2 \cdot \|\mathbf{s}'\| \ll q^2$. It follows that a valid encryption of the bit σ relative to $\mathbf{s}$ is a vector $\mathbf{c}$ such that the inner product of $\mathbf{s}$ and $\mathbf{c}$ is of the form

$$\langle \mathbf{s}, \mathbf{c} \rangle \;=\; kq + \sigma\frac{q}{2} + e,$$

where k, e are of magnitude much smaller than q.

For the basic Regev cryptosystem we only need $|e| < q/4$ and the size of k does not matter. However for the homomorphic operations that are described below we need $k, e \ll q$. Typically q is set to be super-polynomial (or even sub-exponential) in n, and e, k are bounded by some polynomial in n.

2.3 Homomorphic Computation

Let $\mathbf{c}_1, \mathbf{c}_2$ be two valid encryptions of the bits b_1, b_2, respectively, relative to the same key $\mathbf{s}$. By the above, this means that we have $\langle \mathbf{s}, \mathbf{c}_i \rangle \;=\; k_i q + b_i\frac{q}{2} + e_i$ for small k_i, e_i. It therefore follows that their sum, $\mathbf{c}' = [\mathbf{c}_1 + \mathbf{c}_2]_q$, satisfies $\langle \mathbf{s}, \mathbf{c}' \rangle = k'q + (\sigma_1 \oplus \sigma_2)\frac{q}{2} + e'$, where $e' = e_1 + e_2$ and k' is either $k_1 + k_2$ or $k_1 + k_2 \pm 1$. Hence $\mathbf{c}'$ is a valid encryption of the bit $\sigma_1 \oplus \sigma_2$. More interesting is the observation from [BV11a, Bra12] that Regev's scheme also supports multiplication via tensor products. Let $\mathbf{c}^* = \mathbf{c}_1 \otimes \mathbf{c}_2$ denote the dimension-n^2 vector whose entries are all the products of one entry from $\mathbf{c}_1$ and one from $\mathbf{c}_2$ (without any modular reduction), and similarly denotes $\mathbf{s}^* = \mathbf{s} \otimes \mathbf{s}$. Then over the rationals we have:

$$\begin{aligned}\left\langle \mathbf{s}^*,\ \tfrac{2}{q}\cdot\mathbf{c}^* \right\rangle &= \tfrac{2}{q}\cdot\langle \mathbf{s}, \mathbf{c}_1\rangle\cdot\langle \mathbf{s}, \mathbf{c}_2\rangle = \tfrac{2}{q}\cdot\big(k_1 q + b_1(q/2) + e_1\big)\cdot\big(k_2 q + b_2(q/2) + e_2\big) \\ &= (2k_1k_2 + k_1b_2 + k_2b_1)\cdot q \;+\; b_1b_2\cdot(q/2) \\ &\quad + \underbrace{(2k_1 + b_1)e_2 + (2k_2 + b_2)e_1 + \tfrac{2e_1e_2}{q}}_{e''}\end{aligned}$$

Note that the error term e'' above is only polynomially (in n) larger than e_1, e_2 themselves, because k_1, k_2 are bounded by $\mathrm{poly}(n)$.

Rounding $\frac{2}{q}\mathbf{c}^*$ to an integer vector we have $\left\lceil \frac{2}{q}\mathbf{c}^* \right\rfloor = \frac{2}{q}\mathbf{c}^* + \mathbf{e}$ for some rounding-error vector $\mathbf{e}$ with $\|\mathbf{e}\|_\infty \le \frac{1}{2}$. Hence we get

$$\left\langle \mathbf{s}^*,\ \left\lceil \tfrac{2}{q}\mathbf{c}^* \right\rfloor \right\rangle \;=\; \left\langle \mathbf{s}^*,\ \tfrac{2}{q}\mathbf{c}^* \right\rangle + \langle \mathbf{s}^*, \mathbf{e}\rangle \;=\; k''q + b_1b_2(q/2) + e^*,$$

where $e^* = (2k_1 + b_1)e_2 + (2k_2 + b_2)e_1 + \frac{2e_1e_2}{q} + \langle \mathbf{s}^*, \mathbf{e}\rangle$ and k'' is some integer. Since both $\mathbf{s}^* = \mathbf{s} \otimes \mathbf{s}$ and $\mathbf{e}$ have small entries than the added term $\langle \mathbf{s}^*, \mathbf{e}\rangle$ is insignificant, and we have $|e^*| \le \mathrm{poly}(n) \cdot (|e_1| + |e_2|) \ll q$. Finally, reducing the rounded vector modulo q we get a vector $\mathbf{c}'' = \left\lceil \frac{2}{q}\mathbf{c}^* \right\rfloor_q$ satisfying

$$\langle \mathbf{s}^*, \mathbf{c}''\rangle \;=\; k^*q + b_1b_2(q/2) + e^*,$$

for the same small error term e^* are above. The factor k^* can be bounded by

$$k^* \leq 1+\frac{|\langle \mathbf{s}^*, \mathbf{c}''\rangle|}{q} \leq 1+\frac{\|\mathbf{s}^*\|\cdot\|\mathbf{c}''\|}{q} \overset{(\star)}{\leq} 1+\frac{\|\mathbf{s}\|^2\cdot nq/2}{q} = 1+\frac{\|\mathbf{s}\|^2\cdot n}{2} \ll q. \tag{1}$$

The Inequality $(\star)$ follow since $\|\mathbf{s}^*\| = \|\mathbf{s}\otimes\mathbf{s}\| = \|\mathbf{s}\|^2$, and $\mathbf{c}''$ is an n^2-vector with entries of magnitude no larger than $q/2$. We conclude that $\mathbf{c}''$ is a valid encryption of the bit b_1b_2 relative to key $\mathbf{s}^* = \mathbf{s}\otimes\mathbf{s}$.

Key-switching. The multiplication-via-tensoring technique from above comes with the unpleasant side-effect that the dimension of product ciphertext is squared. To overcome this problem, Brakerski and Vaikuntanathan introduced in [BV11a] a key-switching technique. They added to the public key a gadget to enable mapping ciphertexts relative to the high-dimension $\mathbf{s}^*$ into ciphertexts that encrypt the same thing relative to the lower-dimension $\mathbf{s}$. Below we describe a variant similar to the key-switching technique of Gentry et al. [GHS12b], which is a little easier to explain (and more efficient to implement) than the variants from [BV11a, Bra12].

On a high level, the $\mathbf{s}^*$-to-$\mathbf{s}$ key-switching gadget is a (slightly twisted) encryption of $\mathbf{s}^*$ under $\mathbf{s}$. In more detail, for each entry $\mathbf{s}^*[i]$ we put in the public key a *rational* "ciphertext" vector $\mathbf{w}_i$ (say, with $\ell = \Theta(\log q)$ bits of precision to the right of the binary point). This vector satisfies the equality $\langle \mathbf{s}, \mathbf{w}_i\rangle = k_i q + \mathbf{s}^*[i] + e_i$ over the rationals, where the factor k_i is an integer and the magnitude of the error term is bounded by $|e_i| \leq \mathrm{poly}(n)/q$. It can be shown that assuming hardness of decision-LWE with modulus $2^\ell q$ (and a circular-security assumption), these vectors $\mathbf{w}_i$ are pseudo-random. Putting all these vectors as the columns of a matrix, we get an n-by-n^2 rational matrix $\mathbf{W}$ such that over the rationals

$$\mathbf{s}\times\mathbf{W} = \mathbf{k}q + \mathbf{s}^* + \mathbf{e},$$

with $\mathbf{k}$ an integer vector and $\|\mathbf{e}\|_\infty \leq \mathrm{poly}(n)/q$.

Given the n^2-dimension vector $\mathbf{c}^*$ satisfying $\langle \mathbf{s}^*, \mathbf{c}^*\rangle = k'q + b(q/2) + e'$ (for small integers $k', e' \ll q$ and a bit b), we multiply $\mathbf{c}^*$ by $\mathbf{W}$, then round and reduce mod q, getting $\mathbf{c} = \lceil \mathbf{W}\mathbf{c}^*\rfloor_q$. We can express $\mathbf{c}$ as $\mathbf{c} = \mathbf{W}\mathbf{c}^* + \mathbf{e}^* + \mathbf{k}^*q$, with $\mathbf{e}^*$ the rounding error and $\mathbf{k}^*$ the integer factor from reduction modulo q. Then we have

$$\begin{aligned}\langle \mathbf{s}, \mathbf{c}\rangle &= \langle \mathbf{s},\ \mathbf{W}\mathbf{c}^* + \mathbf{e}^* + \mathbf{k}^*q\rangle \ = \ \mathbf{s}\mathbf{W}\mathbf{c}^* + \langle \mathbf{s}, \mathbf{e}^*\rangle + \langle \mathbf{s}, \mathbf{k}^*\rangle\, q \\ &= (\langle \mathbf{k}, \mathbf{c}^*\rangle\, q + \langle \mathbf{s}^*, \mathbf{c}^*\rangle + \langle \mathbf{e}, \mathbf{c}^*\rangle) + \langle \mathbf{s}, \mathbf{e}^*\rangle + \langle \mathbf{s}, \mathbf{k}^*\rangle\, q \\ &= (\underbrace{\langle \mathbf{s}, \mathbf{k}^*\rangle + \langle \mathbf{s}^*, \mathbf{k}\rangle + k'}_{\tilde{k}})q + b(q/2) + \underbrace{\langle \mathbf{e}, \mathbf{c}^*\rangle + \langle \mathbf{s}, \mathbf{e}^*\rangle + e'}_{\tilde{e}}\end{aligned}$$

The magnitude of the error term $\tilde{e}$ can be bounded by noticing the following:

- $\mathbf{e}^*$ is the rounding error, so $\|\mathbf{e}^*\|_\infty \leq \frac{1}{2}$, and therefore $|\langle \mathbf{s}, \mathbf{e}^*\rangle| < \|\mathbf{s}\|_1 \ll q$;
- We have $\|\mathbf{e}\|_\infty \leq \mathrm{poly}(n)/q$ and $\|\mathbf{c}^*\|_\infty \leq q/2$, hence $|\langle \mathbf{e}, \mathbf{c}^*\rangle| \leq n^2\cdot\|\mathbf{e}\|_\infty \cdot \|\mathbf{c}^*\|_\infty = \mathrm{poly}(n) \ll q$.

It thus follows that $|\tilde{e}| \leq |\langle \mathbf{s}, \mathbf{e}^*\rangle| + |\langle \mathbf{e}, \mathbf{c}^*\rangle| + e' \ll q$. As for the size of the factor $\tilde{k}$, here we have similarly to Equation (1):

$$\tilde{k} \leq 1+\frac{|\langle \mathbf{s}, \mathbf{c}\rangle|}{q} \leq 1+\frac{\|\mathbf{s}\|\cdot\|\mathbf{c}\|}{q} \leq 1+\frac{\|\mathbf{s}\|\cdot q\sqrt{n}/2}{q} = 1+\frac{\|\mathbf{s}\|\cdot\sqrt{n}}{2} \ll q.$$

Summing up, we obtained a dimension-n ciphertext vector $\mathbf{c}$ satisfying $\langle \mathbf{s}, \mathbf{c} \rangle = \tilde{k}q + b(q/2) + \tilde{e}$ with $\tilde{k}, \tilde{e} \ll q$, so this is a valid encryption of b relative to $\mathbf{s}$.

2.4 Packed Ciphertexts in Regev's Cryptosystem

As described above, we need a dimension-$(n'+1)$ ciphertext to encrypt a single plaintext bit. Peikert et al. observed in [PVW08] that this ciphertext-expansion ratio can be reduced by packing many plaintext bits in a single ciphertext. Specifically, we can encrypt m' plaintext bits in a ciphertext of dimension $n'+m'$. Below we denote $m = n'+m'$.

To this end, we choose m' (rather than one) secret vectors of dimension n', $\mathbf{s}'_i \leftarrow \chi^{n'}$, and store them as the rows of an m'-by-n' secret matrix $\mathbf{S}'$. Rather than using a dimension-$(n'+1)$ secret-key vector $\mathbf{s} = (1|\mathbf{s}')$ as before, we now use an m'-by-m secret-key matrix $\mathbf{S} = (\mathbf{I}|\mathbf{S}')$, where $\mathbf{I}$ is the m'-by-m' identity matrix.

Recall that for simplicity we describe this cryptosystem as a symmetric encryption scheme. To encrypt a vector of bits $\mathbf{b} \in \{0,1\}^{m'}$, we choose a random vector $\mathbf{a} \in \mathbb{Z}_q^{n'}$ and a random error vector $\mathbf{x} \leftarrow \chi^m$, set $\mathbf{d} = [\mathbf{b} \cdot \frac{q}{2} - \mathbf{S}'\mathbf{a} + \mathbf{x}]_q$ and output the ciphertext vector $\mathbf{c} = (\mathbf{d}|\mathbf{a}) \in \mathbb{Z}_q^m$. To decrypt the m-ciphertext $\mathbf{c}$, we multiply it by $\mathbf{S}$ modulo q, then for each entry of the result output 1 if that entry has magnitude larger than $q/4$ and 0 otherwise. This works because $\mathbf{Sc} = \mathbf{d} + \mathbf{S}'\mathbf{a} = \mathbf{b} \cdot \frac{q}{2} + \mathbf{x} \pmod q$, and the entries of $\mathbf{x}$ are all much smaller than q. Decryption can be expressed using the formula $\mathbf{b} = \lceil\, [\mathbf{Sc}]_q \;/\; (q/2) \,\rfloor_2$. Also, a valid ciphertext relative to $\mathbf{S}$ is a vector $\mathbf{c}$ such that $\mathbf{Sc}$ is of the form

$$\mathbf{Sc} \;=\; \mathbf{k} \cdot q + \mathbf{b} \cdot \frac{q}{2} + \mathbf{e},$$

where $\|k\|_\infty$ and $\|\mathbf{e}\|_\infty$ are of much smaller than q. In other words, the same vector $\mathbf{c}$ is a valid ciphertext relative to all the rows of $\mathbf{S}$, encrypting the i'th bit of $\mathbf{b}$ relative to the i'th row of $\mathbf{S}$.

3 Computing on Packed Ciphertexts

The techniques from Section 2 can be combined "right out of the box" to provide homomorphic evaluation of polynomial of constant degree on packed ciphertexts: Let $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{Z}_q^m$ be two packed ciphertexts, encrypting the bit vectors $\mathbf{b}_1, \mathbf{b}_2 \in \{0,1\}^{m'}$, respectively, relative to the secret key $\mathbf{S} \in \mathbb{Z}_q^{m' \times m}$. Denote the i'th row of $\mathbf{S}$ by $\mathbf{s}_i$, then for all i we have that $\mathbf{c}_1, \mathbf{c}_2$ encrypt the i'th bits of $\mathbf{b}_1, \mathbf{b}_2$, respectively, relative to $\mathbf{s}_i$.

Just like in Section 2.3, this means that for all i, the vector $\mathbf{c}' = [\mathbf{c}_1 + \mathbf{c}_2]_q$ is a valid ciphertext relative to $\mathbf{s}_i$, encrypting the XOR of the i'th bits of $\mathbf{b}_1$ and $\mathbf{b}_2$. In other words, $\mathbf{c}'$ is a valid encryption of the vector $[\mathbf{b}_1 + \; vb_2]_2$, relative to the secret key $\mathbf{S}$. Similarly, setting $\mathbf{c}'' = \left\lceil \frac{2}{q}\mathbf{c}_1 \otimes \mathbf{c}_2 \right\rfloor_q$, we get that for all i, the vector $\mathbf{c}''$ is a valid ciphertext relative to $\mathbf{s}_i^* = \mathbf{s}_i \otimes \mathbf{s}_i$, encrypting the product of the i'th bits of $\mathbf{b}_1$ and $\mathbf{b}_2$. In other words, denoting by $\mathbf{S}^*$ the $m' \times m^2$ matrix with the $\mathbf{s}_i^*$'s as rows, the vector $\mathbf{c}''$ is a valid ciphertext relative to $\mathbf{S}^*$, encrypting the bitwise product $\mathbf{b}_1 \boxdot \mathbf{b}_2 \in \{0,1\}^{m'}$.

3.1 Key-Switching with Packed Ciphertexts

Performing key-switching/relinearization on packed ciphertext takes some care. Clearly, one can put in the public key key-switching gadgets $\mathbf{W}_i$ that for each i enable switching from $\mathbf{s}_i^*$ back to $\mathbf{s}_i$. However, this will only let us convert the single high-dimension ciphertext $\mathbf{c}''$ (which is a valid ciphertext relative to all the $\mathbf{s}_i^*$'s) into a collection of m' low-dimension ciphertexts, the i'th of which is valid with respect to $\mathbf{s}_i$. Instead, we would like to have *a single gadget* that lets us convert the single ciphertext $\mathbf{c}''$ into a single low-dimension ciphertext which is valid relative to all the $\mathbf{s}_i$'s.

Recalling that a key-switching gadget from one key to another is roughly an encryption of the first key under the second, we use our ability to pack many plaintext into one ciphertext to "encrypt all the keys $\mathbf{s}_i^*$ in a single ciphertext" relative to the $\mathbf{s}_i$'s: Our key-switching gadget from $\mathbf{S}^*$ to $\mathbf{S}$ will be a rational matrix $\mathbf{W}$ such that $\mathbf{SW} \equiv \mathbf{S}^* + E \pmod q$ for a sufficiently small error matrix E. In more detail, denoting $m = n' + m'$, then for a secret key $\mathbf{S} = (\mathbf{I}|\mathbf{S}') \in \mathbb{Z}^{m' \times m}$ and an "extended secret key" $\mathbf{S}^* \in \mathbb{Z}^{m' \times m^2}$, we choose the key-switching matrix $\mathbf{W} \in \mathbb{Q}^{m \times m^2}$ as follows:

Let $\ell = \lceil \log q \rceil$, and let χ be an error distribution (over $\mathbb{Z}$) for which the decision-LWE problem with modulus $Q = 2^\ell q$ is hard, and such that with overwhelming probability, elements drawn from χ are much smaller in magnitude than q. Say $|e| \leq \mathrm{poly}(n) \ll q$ whp for $e \leftarrow \chi$. Note that we require hardness relative to the larger modulus Q, even though the error is much smaller than the small modulus q. Since $Q \approx q^2$ then all the known hardness results for LWE carry also to this case. (However, since we have larger modulus-to-noise ratio than we would need a larger dimension n' to get the same level of concrete security.)

For each $j \in \{1, 2, \ldots, m^2\}$, we denote the j'th column of the matrix $\mathbf{S}^*$ by $\tilde{\mathbf{s}}_j \in \mathbb{Z}^{m'}$. The j'th column of $\mathbf{W}$ is set by drawing a random vector $\mathbf{a}_j \in \mathbb{Z}_Q^{n'}$ uniformly at random, drawing an error vector $\mathbf{e}_j \leftarrow \chi^{m'}$, computing $\mathbf{d}_j = [2^\ell \tilde{\mathbf{s}}_j - \mathbf{S}'\mathbf{a}_j + \mathbf{e}_j]_Q$, then dividing by 2^ℓ and outputting the rational column vector (with ℓ bits of precision), $\mathbf{w}_j = (\mathbf{d}_j|\mathbf{a}_j)^T/2^\ell \in \mathbb{Q}^m$.

Let us denote by $\mathbf{k}_j \in \mathbb{Z}^m$ the integer vector containing the factors of the reduction modulo Q from above, so $\mathbf{d}_j = [2^\ell \tilde{\mathbf{s}}_j - \mathbf{S}'\mathbf{a}_j + \mathbf{e}_j]_Q = 2^\ell \tilde{\mathbf{s}}_j - \mathbf{S}'\mathbf{a}_j + \mathbf{e}_j + \mathbf{k}Q$. We thus have for every column $\mathbf{w}_j$:

$$\begin{aligned}\mathbf{S}\mathbf{w}_j = \frac{(\mathbf{I}|\mathbf{S}') \cdot (\mathbf{d}_j|\mathbf{a}_j)^T}{2^\ell} &= \frac{\mathbf{d}_j + \mathbf{S}'\mathbf{a}_j}{2^\ell} = \frac{(2^\ell \tilde{\mathbf{s}}_j - \mathbf{S}'\mathbf{a}_j + \mathbf{e}_j + \mathbf{k}Q) + \mathbf{S}'\mathbf{a}_j}{2^\ell} \\ &= \mathbf{k}_j q + \tilde{\mathbf{s}}_j + \tfrac{\mathbf{e}_j}{2^\ell} \equiv \tilde{\mathbf{s}}_j \pm \mathrm{poly}(n)/q \pmod q\end{aligned}$$

In other words, multiplying $\mathbf{S}$ by the rational matrix $\mathbf{W}$ (without any modular reduction), we have

$$\mathbf{SW} = \mathbf{K}q + \mathbf{S}^* + \mathbf{E}, \tag{2}$$

for some integer matrix $\mathbf{K}$ and an error matrix $\mathbf{E}$ satisfying $\|\mathbf{E}\|_\infty \leq \mathrm{poly}(n)/q \ll 1$ (whp).

Functionality of $\mathbf{W}$. Given a valid high-dimension ciphertext $\mathbf{c}^*$ relative to the secret key $\mathbf{S}^*$, we key-switch it to $\mathbf{S}$ by multiplying by $\mathbf{W}$, then rounding and reducing mod q.

Namely, we set $\mathbf{c} = \lceil \mathbf{W}\mathbf{c}^* \rfloor_q$. This yields a valid low-dimension ciphertext that encrypt the same thing, but relative to $\mathbf{S}$. Namely, if $\mathbf{S}^*\mathbf{c}^* = \mathbf{k}^* q + \mathbf{b}(q/2) + \mathbf{e}^*$ for a bit vector $\mathbf{b}$ and integer vectors $\mathbf{k}^*, \mathbf{e}^*$ of magnitude much smaller than q, then also $\mathbf{S}\mathbf{c} = \mathbf{k}q + \mathbf{b}(q/2) + \mathbf{e}$ for the same $\mathbf{b}$ and where also the magnitude of $\mathbf{k}, \mathbf{e}$ is much smaller than q. The analysis is identical to that in Section 2.3.

Of course, there is nothing special about $\mathbf{S}^*$ and $\mathbf{S}$ above. For every two secret-key matrices $\mathbf{S}_1, \mathbf{S}_2$ we can similarly generate a key-switching gadget $W[\mathbf{S}_1 \to \mathbf{S}_2]$ to enable switching ciphertext relative to $\mathbf{S}_1$ into ciphertexts relative to $\mathbf{S}_2$.

Security of $\mathbf{W}$. It is immediate to show that when $\mathbf{S}_2$ is drawn according to the error distribution χ and independently from $\mathbf{S}_1$, then the key-switching matrix $W[\mathbf{S}_1 \to \mathbf{S}_2]$ is pseudo-random, assuming the hardness of decision-LWE relative to error distribution χ and modulus $Q = 2^\ell q$. To see this, it is enough to observe that each vector $(\mathbf{d}_j | \mathbf{a}_j)$ (before the division by 2^ℓ) is pseudorandom in $\mathbb{Z}_Q$.

Lemma 1. *If the decision LWE problem with error distribution χ and modulus $Q = 2^\ell q$ is hard, then for a random secret key $\mathbf{S}_2 \leftarrow \chi^{m' \times m}$, the key-switching matrix $W[\mathbf{S}_1 \to \mathbf{S}_2]$ as above is indistinguishable from a uniformly random matrix with all the entries drawn independently at random from $[-q/2, q/2)$ with ℓ bits of precision to the right of the binary point. The indistinguishability holds even if the distinguisher gets as input the old secret key $\mathbf{S}_1$.*

Of course, the above lemma does not hold when $\mathbf{S}_1, \mathbf{S}_2$ are related, as in our case where each row of $\mathbf{S}^*$ is the tensor product of the corresponding row of $\mathbf{S}$ with itself. This issue is routinely addresses in one of two ways: One option is to construct a *leveled* HE scheme, where we choose many independent secret key matrices $\mathbf{S}_k$, $k = 1, 2, \ldots$, then put in the public key only the key-switching gadgets $W[\mathbf{S}_k^* \to \mathbf{S}_{k+1}]$. This requires that we switch to a new key after every multiplication, hence the multiplication depth of the circuits that we can handle is bounded by the number of key-switching gadgets in the public key. The other common option of dealing with this issue is to use related $\mathbf{S}_1, \mathbf{S}_2$ anyway, call it circular security, then wave our hands emphatically, saying that we think that the scheme remains secure nonetheless. (Indeed, there are no attacks in the literature that use this relation between $\mathbf{S}_1, \mathbf{S}_2$ to break the scheme.)

3.2 Moving Values between Plaintext Slot

Using the techniques thus far we can implement SIMD-type homomorphic operations on packed ciphertexts, where the same function is applied to m' different inputs at once. However, Gentry et al. pointed out in [GHS12a] that more is needed if we are to apply these techniques for efficient evaluation of (wide enough) circuits on just one input. To take advantage of internal parallelism opportunities within a circuit, we must also be able to move values between different plaintext slots, so as to move a value from the output of one gate in level i of the circuit to the input of another gate in level $i + 1$.

Specifically, following [GHS12a] we seek an efficient procedure for permuting the plaintext slots according to any given permutation. Below we denote by $\pi(\mathbf{b})$ the permutation according to π of the entries of the vector $\mathbf{b}$ (i.e., the vector whose $\pi(i)$'th

entry equals the i'th entry of $\mathbf{b}$). Similarly, for a matrix $\mathbf{S}$ we denote by $\pi(\mathbf{S})$ the permutation of the rows of $\mathbf{S}$ according to π. What we seek, therefore, is a transformation that given a ciphertext $\mathbf{c}$ encrypting a binary vector $\mathbf{b} \in \{0,1\}^{m'}$ relative to $\mathbf{S}$, outputs another ciphertext $\mathbf{c}'$ encrypting the permuted vector $\pi(\mathbf{b})$ relative to the same $\mathbf{S}$.

The technique from above for key-switching can be easily adopted to this purpose. Indeed, it follows from the decryption formula that if $\mathbf{c}$ is a valid encryption of $\mathbf{b}$ relative to $\mathbf{S}$, then the same $\mathbf{c}$ is also a valid encryption of $\pi(\mathbf{b})$ relative to $\pi(\mathbf{S})$. All we need, therefore, is to switch $\mathbf{c}$ from the key $\pi(\mathbf{S})$ back to the key $\mathbf{S}$, which we can do if we have in the public key a key-switching gadget $\mathbf{W} = W[\pi(\mathbf{S}) \to \mathbf{S}]$. Namely, permuting the plaintext slots according to π is done by setting $\mathbf{c}' = \lceil \mathbf{W}\mathbf{s} \rfloor_q$.

3.3 Discussion

In this note we described a very simple method of computing on packed ciphertext in the PVW variant of Regev's cryptosystem. This provides an efficiency boost for LWE-based HE schemes, similar to the boost that we get by using the techniques of [SV11, GHS12a] in RLWE-based schemes.

We stress, however, that schemes over polynomial rings still offer much better asymptotic efficiency than schemes over the ring of integers. The size of ciphertexts in both cases is roughly the same (upto a constant factor), since ciphertexts in polynomial-ring schemes consist of a constant number of ring elements, each element represented by $O(n)$ integers. However, the tensor product multiplication increases the ciphertext size over the integers to $O(n^2)$ integers, whereas over polynomial rings we still only need $O(n)$ integers to describe even the product ciphertext. Even more, the re-linearizion gadget over the integers is a $O(n)$-by-$O(n^2)$ matrix, which takes $O(n^3)$ integers to represent. In the polynomial-ring setting, this matrix is still only an $O(1)$-by-$O(1)$ matrix over the ring, so it still only takes $O(n)$ integers to represent. As a consequence, whereas in the polynomial-ring setting [GHS12a] were able to reduce the overhead of homomorphic evaluation to only polylogarithmic factor (for wide enough circuits), using the same techniques over the integers (with the ciphertext-packing tools from the current paper) would yield a quasi-quadratic overhead.

On the other hand, the security of the integer schemes is based on the hardness of standard LWE, which is arguably better understood than the hardness of ring-LWE (or the NTRU problem) which underly the security of schemes over polynomial rings. In addition, the techniques that we described in this note are perhaps more flexible and less "algebraically heavy" than their polynomial-ring counterparts.

For example, for polynomial-ring schemes, the number of plaintext slots in each ciphertext is determined by the algebra of the underlying ring, and thus not every number of slots is achievable. For example, Gentry et al reported in [GHS12b] on homomorphic evaluation of the AES circuit, that used only 16 plaintext slots (for the 16 bytes in the AES state). However, security considerations dictated that the ring be very large, which resulted in several hundred unused plaintext slots. In contrast, the number of plaintext slots in PVW ciphertext packing is a free parameter that can be set to any desired value.

Perhaps the main advantage of the packed-ciphertext computation techniques for integer-based schemes over their counterparts for polynomial-ring schemes is the simplicity of implementing data movement over plaintext slots. In polynomial rings, these

operations are implemented using automorphism, but only a small set of permutations (determined by the ring algebra) can be implemented this way. In [GHS12a] it was shown how to implement any data movement pattern using the small set that we get from the ring algebra, but that procedure requires a logarithmic number of basic automorphisms to implement a given data-movement pattern. In contrast, the technique from Section 3.2 lets us directly implement any data-movement pattern, just by putting in the public key the corresponding key-switching gadget. Hence if we know ahead of time the circuit that we want to evaluate homomorphically (e.g., the AES circuit), we can prepare the corresponding gadgets to enable computing the data-movement patterns of that circuit directly. Of course, if we do not know the circuit ahead of time, we can put in the public key the gadgets for just a few permutations, and then use the techniques from [GHS12a] to implement arbitrary patterns, incurring the same logarithmic slowdown.

Another thing which is easier to do in integer-based schemes than in polynomial-based scheme is to gradually reduce the dimension as the computation progresses: Fresh ciphertexts in all these schemes must have a very large modulus/noise ratio to enable computation, which implies that we need fairly high dimension (or fairly high ring-size) to get security. However, larger noise (and hence smaller modulus/noise ratio) is used as the computation progresses, so from a security standpoint it is permissible to switch to lower dimension (or smaller ring), thus speeding up further homomorphic operations. Recently, it was shown in [GHPS12] how to do this for schemes in polynomial rings, but this operation is highly constrained by the relevant algebra. Specifically, if the dimension of the initial ring is some m, then it is only possible to switch to other rings of dimension that divides m. In particular it means that the first time we can perform this transformation is when it is safe to switch to a ring of size $m/2$ (or less), which means that at least half the computation has to be done relative to the original large ring. In contrast, switching to a lower dimension is nearly trivial in LWE-based schemes: All we need is key-switching from the initial dimension-n key to a lower-dimension key, exactly as is done for re-linearization (with the noise magnitude in the key-switching gadget increased to provide adequate security relative to the lower dimension).

Finally, we mention that the techniques described in this note can also be used in conjunction with HE schemes over polynomial rings (with security based on the "general LWE" problem), as suggested, e.g., in [BGV12]. This setting offers a tradeoff, with schemes over the integers on one end and schemes over large polynomial rings on the other. In the middle we have schemes over smaller polynomial rings, in which ciphertexts are low-dimension vectors over these rings. (For example, the ring may have dimension $n/\log n$, and then ciphertexts and secret key can have dimension $O(\log n)$ over that ring.) This opens yet another avenue for tradeoffs and optimizations, for example we can choose the ring with best algebraic properties, even if it is too small to provide the security level that we seek, then use slightly longer vectors over that ring to recover security. In this context, it is possible to use both ciphertext packing techniques: pack many plaintext values relative to each secret-key vector using the polynomial-CRT techniques, and use many secret key vectors to reduce the ciphertext expansion ratio as described in this work.

Acknowledgments. The first author is sponsored by DARPA under agreement number FA8750-11-C-0096. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

The second and third authors are sponsored by DARPA and ONR under agreement number N00014-11C-0390. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, or the U.S. Government. Distribution Statement "A" (Approved for Public Release, Distribution Unlimited).

References

[ACPS09] Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 595–618. Springer, Heidelberg (2009)

[BGV12] Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: Goldwasser, S. (ed.) Innovations in Theoretical Computer Science – ITCS 2012, pp. 309–325. ACM (2012), http://eprint.iacr.org/2011/277

[Bra12] 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), http://eprint.iacr.org/2012/078

[BV11a] Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: Ostrovsky, R. (ed.) FOCS, pp. 97–106. IEEE (2011)

[BV11b] 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)

[Gen09] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) Proceedings of the 41st Annual Symposium on Theory of Computing, STOC 2009, pp. 169–178. ACM (2009)

[GHPS12] Gentry, C., Halevi, S., Peikert, C., Smart, N.P.: Ring Switching in BGV-Style Homomorphic Encryption. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 19–37. Springer, Heidelberg (2012), Full version at http://eprint.iacr.org/2012/240

[GHS12a] Gentry, C., Halevi, S., Smart, N.P.: Fully Homomorphic Encryption with Polylog Overhead. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 465–482. Springer, Heidelberg (2012), Full version at http://eprint.iacr.org/2011/566

[GHS12b] 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), Full version at http://eprint.iacr.org/2012/099

[LPR10] 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)

[Pei09] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Mitzenmacher, M. (ed.) Proceedings of the 41st Annual Symposium on Theory of Computing, STOC 2009, pp. 333–342. ACM (2009)

[PVW08] Peikert, C., Vaikuntanathan, V., Waters, B.: A Framework for Efficient and Composable Oblivious Transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008)

[Reg09] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM 56(6) (2009)

[Rot11] Rothblum, R.: Homomorphic Encryption: From Private-Key to Public-Key. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 219–234. Springer, Heidelberg (2011)

[SV11] Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations (2011), manuscript at `http://eprint.iacr.org/2011/133`

Feasibility and Infeasibility of Adaptively Secure Fully Homomorphic Encryption

Jonathan Katz*, Aishwarya Thiruvengadam, and Hong-Sheng Zhou**

Department of Computer Science, University of Maryland
{jkatz,aish,hszhou}@cs.umd.edu

Abstract. Fully homomorphic encryption (FHE) is a form of public-key encryption that enables arbitrary computation over encrypted data. The past few years have seen several realizations of FHE under different assumptions, and FHE has been used as a building block in many cryptographic applications.

Adaptive security for public-key encryption schemes is an important security notion proposed by Canetti et al. It is intended to ensure security when encryption is used within an interactive protocol and parties may be *adaptively* corrupted by an adversary during the course of the protocol execution. Due to the extensive applications of FHE to protocol design, it is natural to understand whether adaptively secure FHE is achievable.

In this paper we show two contrasting results in this direction. First, we show that adaptive security is *impossible* for FHE satisfying the (standard) *compactness* requirement. On the other hand, we show a construction of adaptively secure FHE that is not compact, but that does achieve circuit privacy.

1 Introduction

1.1 Fully Homomorphic Encryption

Fully homomorphic encryption (FHE) [18,11] is a form of public-key encryption that enables a third party (who does not know the associated secret key) to perform computations over encrypted data. That is, given a public key pk and a ciphertext $c = \mathsf{Enc}_{pk}(m)$ that is the encryption of some (unknown) plaintext message m, anyone can compute a ciphertext c' whose decryption is $f(m)$ for any

* Research sponsored in part by the US Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained here are those of the authors and should not be interpreted as representing the official policies, expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

** Supported by an NSF CI postdoctoral fellowship.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 14–31, 2013.

desired function f. The actual definition is even more general (see Section 2.1): given pk and ciphertexts

$$c_1 = \mathsf{Enc}_{pk}(m_1), \ldots, c_\ell = \mathsf{Enc}_{pk}(m_\ell),$$

it is possible to compute an encryption of $f(m_1, \ldots, m_\ell)$.

FHE has several applications. As one example, FHE can be used to construct simple protocols for secure computation. We restrict ourselves to the two-party setting with honest-but-curious parties. (In this setting two parties with inputs x and y, respectively, wish to compute a function $f(x, y)$ over their inputs without revealing to each other anything more than the result; in the honest-but-curious setting, the parties are again assumed to follow the protocol though privacy of their inputs must still be maintained.) Here, a party with input x can generate a public/private key pair (pk, sk) for an FHE scheme and send $pk, \mathsf{Enc}_{pk}(x)$ to the other party. This second party can then compute an encryption of the desired result $f(x, y)$ and send the resulting ciphertext back to the first party. The first party can then decrypt this ciphertext to recover $f(x, y)$.

FHE with the functionality described above can be realized trivially by the construction in which we define $f, \mathsf{Enc}_{pk}(m_1), \ldots, \mathsf{Enc}_{pk}(m_\ell)$ to be a valid encryption of $f(m_1, \ldots, m_\ell)$. This notion, however, does not suffice for most applications of FHE; in particular, it does not suffice for the application described above. Thus, some "non-triviality" requirement must be added to the definition of FHE in order to make the notion meaningful. Various requirements can be considered, and we consider two here: (1) *compactness*, which requires that ciphertexts have bounded length, and (2) *circuit privacy*, which informally requires that the encryption of $f(m_1, \ldots, m_\ell)$ should not reveal f. Note that the trivial scheme described earlier does not satisfy either of these conditions.

1.2 Adaptive Security

In a separate line of work, Canetti et al. [6] proposed the notion of *adaptive security* for (standard) public-key encryption schemes. Their motivation was to guarantee security when encryption schemes are used to encrypt messages sent during an interactive protocol, and parties running the protocol can be *adaptively* corrupted during the course of the protocol execution [4]. (The adaptive-corruption model stands in contrast to the *static*-corruption model where the attacker is assumed to corrupt parties only *before* the protocol begins.)

The primary challenge with regard to adaptively secure encryption is that the protocol *simulator* (used to prove security of the protocol) must simulate the ciphertexts being sent by the various parties without knowing the underlying plaintext. At some later point in time, the adversary may request to corrupt a party and the simulator must then simulate for the adversary any secret keys held by that party. (If secure erasure is not assumed, the simulator will also have to simulate for the adversary any random coins used by the sender. This only makes the problem harder.) These secret keys must be such that they correctly decrypt any ciphertexts previously sent to that party. Most natural public-key

encryption schemes will not be suitable here, in particular because a given public key typically has a unique secret key associated with it, which implies that any (correctly generated) ciphertext can be "opened" later to at most one plaintext.

Canetti et al. [6] show how to construct adaptively secure encryption schemes from general assumptions. Subsequent research [3,10,14,15,7,9] has shown more efficient constructions based on specific number-theoretic assumptions, or satisfying weaker (but still meaningful) notions of adaptive security.

1.3 Adaptively Secure FHE?

Because of the applications of FHE to protocol design, it is natural to ask whether adaptive security can be realized for FHE. We focus on *receiver* corruption; equivalently, we assume secure erasure and so senders can erase the random coins they use immediately before sending a ciphertext. (But the receiver cannot erase its secret key until it receives and decrypts the ciphertext.)

We show two results in this regard. First, we show unconditionally that adaptive security is *impossible* for FHE schemes satisfying compactness.[1] On the other hand, we show that adaptive security *is* possible for FHE schemes satisfying circuit privacy. Our results are interesting in their own right, but also show a separation between two notions of non-triviality (namely, compactness and circuit privacy) that have been considered in the literature.

2 Definitions

Throughout, we let k denote the security parameter.

2.1 Fully Homomorphic Encryption

We begin by formally defining the notion of fully homomorphic encryption.

Definition 1 (Fully homomorphic encryption). *Fix a function $\ell = \ell(k)$. An ℓ-*homomorphic encryption scheme *$\mathcal{HE}$ for a class of circuits $\{\mathcal{C}_k\}_{k\in\mathbb{N}}$ consists of four polynomial-time algorithms* Gen*,* Enc*,* Dec*, and* Eval *such that*

- Gen*, the* key-generation algorithm*, is a randomized algorithm that takes the security parameter 1^k as input and outputs a public key pk and secret key sk.*
- Enc*, the* encryption algorithm*, is a randomized algorithm that takes a public key pk and a message $m \in \{0,1\}$ as input, and outputs a ciphertext c.*
- Dec*, the* decryption algorithm*, is a deterministic algorithm that takes the secret key sk and a ciphertext c as input, and outputs a message $m \in \{0,1\}$.*

[1] The impossibility result of Nielsen [17] does not apply to our setting, since we are willing to place an *a priori* upper bound on the length of plaintext(s) that are encrypted under a single public key. This makes sense when encryption is used to encrypt messages sent within an interactive protocol, where the length of the messages to be encrypted is bounded in advance (and a fresh public key can be generated for each independent protocol execution).

– Eval, *the* homomorphic evaluation algorithm, *takes as input a public key* pk, *a circuit* $C \in \mathcal{C}_k$, *and ciphertexts*[2] $c_1, \cdots, c_{\ell(k)}$; *it outputs a ciphertext* c^*.

The following properties are required to hold:

1. *For any* k, *any* (pk, sk) *output by* $\mathsf{Gen}(1^k)$, *and any* $m \in \{0,1\}$ *we have* $m = \mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m))$.
2. *For any* k, *any* (pk, sk) *output by* $\mathsf{Gen}(1^k)$, *any* $m_1, \ldots, m_{\ell(k)} \in \{0,1\}$, *and any* $C \in \mathcal{C}_k$, *we have*

$$C(m_1, \ldots, m_\ell) = \mathsf{Dec}_{sk}(\mathsf{Eval}_{pk}(C, \mathsf{Enc}_{pk}(m_1), \ldots, \mathsf{Enc}_{pk}(m_\ell)))$$

We use the standard notion of security against chosen-plaintext attacks. (Although stronger notions of security could be considered, the question of adaptive security is tangential to these considerations.)

Definition 2. *A homomorphic encryption scheme* $\mathcal{HE}$ *is* CPA-secure *if for any polynomial-time adversary* $\mathcal{A}$ *the following is negligible in* k:

$$|\Pr[\mathcal{A}(pk, \mathsf{Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, \mathsf{Enc}_{pk}(1)) = 1]|,$$

where $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$.

As noted earlier, Definitions 1 and 2 are not enough to capture a meaningful notion of fully homomorphic encryption because they can be satisfied by a "trivial" construction starting from any CPA-secure (standard) public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}', \mathsf{Dec}')$ by defining Enc, Eval, and Dec as follows:

– $\mathsf{Enc}_{pk}(m) = (0, \mathsf{Enc}'_{pk}(m))$.
– $\mathsf{Eval}_{pk}(C, c_1, \ldots, c_\ell)$ outputs $(1, C, c_1, \ldots, c_\ell)$.
– $\mathsf{Dec}_{sk}(c)$ does as follows: if $c = (0, c')$, then output $\mathsf{Dec}'_{sk}(c')$ (i.e., decrypt as in Π). If $c = (1, C, c_1, \ldots, c_\ell)$, then output

$$C(\mathsf{Dec}'_{sk}(c_1), \ldots, \mathsf{Dec}'_{sk}(c_\ell))$$

(i.e., decrypt and then apply C to the results).

There are various ways one could imagine ruling out trivial schemes like the above. The first approach (following previous work in the literature) is to require that ciphertexts cannot grow arbitrarily large; this is known as *compactness*.

Definition 3 (Compactness). *An* ℓ*-homomorphic encryption scheme* $\mathcal{HE}$ *for a class of circuits* $\{\mathcal{C}_k\}_{k \in \mathbb{N}}$ *is* compact *if there exists a polynomial* $\alpha = \alpha(k)$ *such that ciphertexts output by* Eval *have length at most* α. *(For this to be non-trivial it should be the case that, for all* k, *we have* $\alpha(k) \leq |C|$ *for some* $C \in \mathcal{C}_k$.*)*

[2] We assume for simplicity that all circuits in $\mathcal{C}_k$ take exactly $\ell = \ell(k)$ input bits.

We say an ℓ-homomorphic encryption scheme is *ℓ-fully homomorphic* if it is homomorphic for all boolean circuits, CPA-secure, and compact.

An alternate non-triviality condition that has been considered is to require that the output of $\mathsf{Eval}_{pk}(C, c_1, \ldots, c_\ell)$ reveal nothing about C, even to the holder of the secret key sk. This notion is called *circuit privacy*. There are different ways of formalizing such a notion. The definition we use is weaker than the one introduced by Gentry [12], but similar to the notion considered in [13]. We also note that we allow (an upper bound on) the *size* of C to be revealed.

Definition 4 (Circuit privacy). *An ℓ-homomorphic encryption scheme $\mathcal{HE}$ for a class of circuits $\{\mathcal{C}_k\}_{k\in\mathbb{N}}$ is* circuit private *if there exists an efficient simulator $\mathcal{S}$ such that for every (pk, sk) generated by* Gen*, every $C \in \mathcal{C}_k$, and every $m_1, \ldots, m_\ell$, the following two distributions are computationally indistinguishable (even given $pk, sk, C, m_1, \ldots, m_\ell$):*

$$\left\{\forall i : c_i \leftarrow \mathsf{Enc}_{pk}(m_i) : \left(\mathsf{Eval}_{pk}(C, c_1, \cdots, c_\ell), c_1, \ldots, c_\ell\right)\right\}$$

$$\left\{\forall i : c_i \leftarrow \mathsf{Enc}_{pk}(m_i) : \mathcal{S}(1^k, pk, |C|, C(m_1, \ldots, m_\ell), c_1, \cdots, c_\ell)\right\}.$$

2.2 Adaptively Secure Fully Homomorphic Encryption

We consider here a security notion for FHE that captures adaptive corruption of the *receiver*. (Alternately, we assume secure erasure and thus the sender can erase the randomness it uses for encryption immediately after encryption is complete.) Here, a simulator is required to commit to (a bounded number of) simulated ciphertexts $c_1, \ldots, c_\ell$; the adversary then outputs messages $m_1, \ldots, m_\ell \in \{0,1\}$, and the simulator must give the adversary a (single) secret key sk that "explains" (i.e., decrypts) each ciphertext c_i as m_i.

Definition 5 (Adaptively secure FHE). *An ℓ-homomorphic encryption scheme $\mathcal{HE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ is* adaptively secure *if there exists a non-uniform, polynomial-time algorithm $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for all non-uniform, polynomial-time algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ we have*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

$\underline{\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k)}$
$(pk, c_1, \ldots, c_\ell, s) \leftarrow \mathcal{S}_1(1^k)$;
$(m_1, \ldots, m_\ell, \tau) \leftarrow \mathcal{A}_1(1^k, pk)$;
$sk \leftarrow \mathcal{S}_2(s, m_1, \ldots, m_\ell)$;
$b \leftarrow \mathcal{A}_2(\tau, pk, c_1, \ldots, c_\ell, sk)$;
return b.

$\underline{\mathsf{Real}_{\mathcal{A}}(k)}$
$(pk, sk) \leftarrow \mathsf{Gen}(1^k)$;
$(m_1, \ldots, m_\ell, \tau) \leftarrow \mathcal{A}_1(1^k, pk)$;
$c_1 \leftarrow \mathsf{Enc}_{pk}(m_1); \ldots;$
$c_\ell \leftarrow \mathsf{Enc}_{pk}(m_\ell)$;
$b \leftarrow \mathcal{A}_2(\tau, pk, c_1, \ldots, c_\ell, sk)$;
return b.

□

3 Impossibility Result

In this section, we show that adaptively secure ℓ-fully homomorphic encryption is *impossible*. We first give some intuition. Say adaptively secure FHE were possible, so there is a simulator as in Definition 5. This gives an alternate way of computing any function f, described by a circuit $C_f : \{0,1\}^\ell \to \{0,1\}$, as follows:

1. Run $\mathcal{S}_1$ to obtain pk, $c_1, \ldots, c_\ell$, and state s.
2. Compute $c' \leftarrow \mathsf{Eval}_{pk}(C_f, c_1, \ldots, c_\ell)$.
3. Given input $x \in \{0,1\}^\ell$, run $\mathcal{S}_2(s, x_1, \ldots, x_\ell)$ to obtain a secret key sk.
4. Compute $\mathsf{Dec}_{sk}(c')$ to obtain $f(x)$.

Note that steps 1 and 2 can be computed in advance of receiving the input x. Thus, we can hard-code s, c', and randomness (if any) for $\mathcal{S}_2$ into a circuit that, upon receiving input $x = (x_1, \ldots, x_\ell)$, computes $sk = \mathcal{S}_2(s, x)$ and then outputs $\mathsf{Dec}_{sk}(c')$. Adaptive security implies that this output must be correct for most inputs x. (More precisely, it guarantees that there *exist* values of s, c', and randomness for $\mathcal{S}_2$ for which the circuit is correct for most inputs x.) But because Dec and $\mathcal{S}_2$ are algorithms of some fixed complexity, and c' is of some bounded size (here is where we use the compactness property), we have some polynomial upper-bound t on the size of the circuit that we get above. Taking f to be a function that cannot be approximated by circuits of size t, but that can be computed by a circuit of some polynomial size $T \gg t$, we obtain a contradiction.

Theorem 1. *Let $\ell = \omega(\log k)$. Then, adaptively secure fully ℓ-homomorphic encryption does not exist.*

Proof. Assume, toward a contradiction, that such a scheme $\mathcal{HE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ exists. This implies the existence of a non-uniform family of circuits $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ satisfying Definition 5. Let $t(k)$ denote an upper bound on the size of the circuit for $\mathcal{S}_2$ plus the size of a circuit computing Dec for any ciphertext c' output by Eval. Using compactness (which says that the size of any such c' is bounded by some fixed polynomial) and the fact that Dec runs in polynomial time, we see that $t(k) = \mathsf{poly}(k)$.

Let $\{f_k : \{0,1\}^{\ell(k)} \to \{0,1\}\}_k$ be a function family that can be computed by polynomial-size circuits. Fix some particular k in the discussion that follows, and let C_f be the circuit computing $f = f_k$. Define a circuit $C^*_{s,c',\omega}$ as follows. First, compute $(pk, c_1, \ldots, c_\ell, s) \leftarrow \mathcal{S}_1(1^k)$. Then compute $c' \leftarrow \mathsf{Eval}_{pk}(C_f, c_1, \ldots, c_\ell)$. Choose random coins ω for $\mathcal{S}_2$, and define $C^*_{s,c',\omega}$ as:

- On input $x \in \{0,1\}^\ell$, run $\mathcal{S}_2(s, x_1, \ldots, x_\ell)$ (using random coins ω) to obtain sk. Then output $\mathsf{Dec}_{sk}(c')$.

We stress that s, c', and ω are hard-coded into the above circuit. Thus, the size of $C^*_{s,c',\omega}$ is at most $t(k)$.

A circuit $C : \{0,1\}^\ell \to \{0,1\}$ is an *ϵ-approximation of f* if

$$\Pr_{x \leftarrow \{0,1\}^\ell}[C(x) = f(x)] \geq \epsilon.$$

The theorem follows from the next two lemmas.

Lemma 1. *There exist s, c', ω such that the circuit $C^*_{s,c',\omega}$ constructed above is a 3/4-approximation of f.*

Proof. Consider the following non-uniform, polynomial-size adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$: adversary $\mathcal{A}_1$, on input pk, outputs random $x_1, \ldots, x_\ell \in \{0,1\}$. On input $c_1, \ldots, c_\ell$ and sk, adversary $\mathcal{A}_2$ computes $c' \leftarrow \mathsf{Eval}_{pk}(C_f, c_1, \ldots, c_\ell)$ followed by $y = \mathsf{Dec}_{sk}(c')$. (Non-uniformity is used to hard-wire into $\mathcal{A}_2$ a description of the circuit C_f.) Finally, $\mathcal{A}_2$ outputs 1 if and only if $y = f(x_1, \ldots, x_\ell)$.

Correctness of the FHE scheme implies that in $\mathsf{Real}_{\mathcal{A}}(k)$ the adversary always outputs 1. Adaptive security thus implies that the adversary outputs 1 with all but negligible probability in $\mathsf{Ideal}_{\mathcal{A}}(k)$. But this means that

$$\Pr_{x,s,c',\omega}[C^*_{s,c',\omega}(x) \neq f(x)] < \mathsf{negl}(k),$$

where $x \in \{0,1\}^\ell$ is chosen uniformly and s, c', ω are generated as in the construction of $C^*_{s,c',\omega}$ described earlier. But then there exist s, c', ω for which

$$\Pr_{x}[C^*_{s,c',\omega}(x) \neq f(x)] < \mathsf{negl}(k),$$

where the probability is now only over the uniform choice of $x \in \{0,1\}^\ell$. This circuit $C^*_{s,c',\omega}$ is thus a 3/4-approximation of f.

The contradiction is given by the fact that there exist functions f that can be computed by circuits of polynomial size T but cannot be 3/4-approximated by circuits of size t.

Lemma 2. *For any $t(k) = \mathsf{poly}(k)$ and $\ell(k) = \omega(\log k)$, there exists a function family $\{f_k : \{0,1\}^{\ell(k)} \to \{0,1\}\}_k$ that can be computed by circuits of polynomial size $T(k)$ but cannot be 3/4-approximated by any circuit of size $t(k)$.*

Proof. A proof follows via suitable modification of the proof of the standard hierarchy theorem for non-uniform computation [1]. Pick a random function f, and consider the probability that a fixed circuit C correctly computes f on at least 3/4 of its inputs. Using Chernoff bounds, we can show that this probability is at most $e^{-2^\ell/16}$. Since there are at most $2^{2S \log S + 5S}$ circuits of size S, we have that if $S = 2^{\ell/2}/16$ (and hence $2S \log S + 5S < 2^\ell/16$) there exists a function that is hard to 3/4-approximate for *all* circuits of size S.

Any function $g : \{0,1\}^n \to \{0,1\}$ is computable by a circuit of size $2^n 10n$. Let k be such that $\ell > 2.2 \log t(k)$ (here we use the fact that $\ell = \omega(\log k)$). If we set $n = 2.2 \log t(k)$ and let $f : \{0,1\}^\ell \to \{0,1\}$ be the function that applies g to the first n bits of its input, then f can be computed by a circuit of size $O(t(k)^3)$, but cannot be 3/4-approximated by circuits of size $t(k)$.

This concludes the proof of the theorem.

We note that our impossibility result also holds for a weaker definition of adaptive security where the adversary has to output the messages (whose encryptions the simulator has to explain) before getting the public key.

4 Feasibility Result

In this section, we show that adaptive security is possible for *circuit-private* fully homomorphic encryption. The main idea is to modify the constructions in [12,13,2] to use adaptively secure building blocks. Specifically, our construction is based on *(i)* a two-move semi-honest oblivious transfer (OT) protocol with receiver adaptive security, *(ii)* a projective garbling scheme leaking only the circuit size [19,5], and *(iii)* multiple-message, receiver-non-committing public-key encryption (which is a stronger version of single-message receiver non-committing encryption introduced in [14]).

We recall the high-level idea of [12,13,2], and explain how to upgrade the building blocks to achieve our goal. The idea is to use the key generation algorithm of a public-key encryption scheme to generate the public and secret keys. To perform encryption, encode the input to be sent and in addition, encrypt the randomness used for encoding. The evaluation algorithm forwards the encryption of the randomness in addition to an encoded value of the result of the evaluation of the circuit on the input sent. For decryption, using the secret key, we can decrypt to get the randomness used. This randomness can then be used to recover the result. This idea can be constructed using a regular public key encryption scheme, oblivious transfer protocol and Yao's garbling technique [19].

To achieve adaptive security, we need to upgrade the building blocks to adaptively secure ones. A semi-honest two-move OT protocol will suffice in the above construction, but to achieve adaptive security of the circuit-private homomorphic encryption we need semi-honest two-move OT but with *adaptive receiver security*. The semi-honest OT protocol in [8] is sufficient for our goal. We also need to replace the regular public key encryption with a *multi-message receiver non-commiting encryption* (a formal definition can be found below), a strengthened version of receiver non-committing encryption introduced in [14].

Before describing our construction, we recall the definitions of garbling schemes, semi-honest OT with adaptive receiver security, and then define the multi-message receiver non-committing encryption. Then, we present our construction of a circuit-private homomorphic scheme which achieves correctness, adaptive security, and circuit privacy, but not compactness.

4.1 Building Blocks

Garbling Schemes. Here, we define garbling schemes and introduce the security notion we consider for such schemes in this work. Our notation follows the recent work by Bellare, Hoang, and Rogaway [5].

A *garbling scheme* is a five-tuple of algorithms $\mathcal{G} = (\mathsf{Gb}, \mathsf{En}, \mathsf{De}, \mathsf{Ev}, \mathsf{ev})$. A string f, the original function, describes the function $\mathsf{ev}(f, .) : \{0,1\}^{\ell} \to \{0,1\}^{n}$ that we want to garble. On input f and a security parameter k, the probabilistic algorithm Gb returns a triple of strings $(F, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, f)$. String e describes an encoding function, $\mathsf{En}(\mathsf{e}, .)$, that maps an initial input $m \in \{0,1\}^{\ell}$ to a garbled input $X = \mathsf{En}(\mathsf{e}, m)$. String F describes a garbled function $\mathsf{ev}(F, .)$, that maps

each garbled input X to a garbled output $Y = \mathsf{ev}(F, X)$. String d describes a decoding function, $\mathsf{De}(\mathsf{d}, .)$, that maps a garbled output Y to a final output $y = \mathsf{De}(\mathsf{d}, Y)$.

We consider only projective garbling schemes in this work. A *projective garbling scheme* as described in [5] is one where e encodes a list of tokens, one pair for each bit in $m \in \{0,1\}^\ell$. Encoding function $\mathsf{En}(\mathsf{e}, .)$ uses the bits of $m = m_1 \cdots m_\ell$ to select from $\mathsf{e} = X_1^0, X_1^1, \cdots, X_\ell^0, X_\ell^1$ the subvector $X = (X_1^{m_1}, \cdots, X_\ell^{m_\ell})$. A garbling scheme $\mathcal{G} = (\mathsf{Gb}, \mathsf{En}, \mathsf{De}, \mathsf{Ev}, \mathsf{ev})$ is *projective* if for all $f, m, m' \in \{0,1\}^\ell$, $k \in \mathbb{N}$, and $i \in [\ell]$, when $(F, \mathsf{e}, \mathsf{d}) \in [\mathsf{Gb}(1^k, f)]$, $X = \mathsf{En}(\mathsf{e}, m)$ and $X' = \mathsf{En}(\mathsf{e}, m')$, then $X = (X_1 \cdots X_\ell)$ and $X' = (X'_1 \cdots X'_\ell)$ are ℓ vectors, $|X_i| = |X'_i|$, and $X_i = X'_i$ if m and m' have the same ith bit.

For the privacy notion considered, we allow that certain information about the function f can be revealed and this is captured by the *side information function* $\Phi(f)$. Specifically, for this work, the side information function is the size of the circuit.

For the security notion, we describe only the simulation-based notion of privacy in [5]. We present the definition of the simulation-based security notion of privacy of a garbling scheme.

Definition 6. *Consider the following game* $\mathrm{PrvSim}_{\mathcal{G},\Phi,\mathcal{S}}$ *associated with a garbling scheme* $\mathcal{G}$*, side information function* $\Phi(f)$ *and a simulator* $\mathcal{S}$*. The adversary* $\mathcal{A}$ *is run on input* 1^k *and makes exactly one* GARBLE *query. The* GARBLE *procedure is described as follows.*

$\underline{\textsc{Garble}(f, m)}$
$b \leftarrow \{0,1\}$
if $m \notin \{0,1\}^\ell$ *return* $\perp$
if $b = 1$ *then* $(F, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, f)$; $X \leftarrow \mathsf{En}(\mathsf{e}, m)$
else $y \leftarrow \mathsf{ev}(f, m)$; $(F, X, \mathsf{d}) \leftarrow \mathcal{S}(1^k, y, \Phi(f))$
return (F, X, d)

The adversary after getting the answer to the query must output a bit b'*. The adversary's advantage is given by:*

$$\mathbf{Adv}_{\mathcal{G}}^{\Phi,\mathcal{S}}(\mathcal{A}, k) = \left| \mathbf{Pr}\left[b' = b\right] - \frac{1}{2} \right|$$

The protocol $\mathcal{G}$ is secure over Φ if for every polynomial-time adversary $\mathcal{A}$ there is a polynomial-time simulator $\mathcal{S}$ such that $\mathbf{Adv}_{\mathcal{G}}^{\Phi,\mathcal{S}}(\mathcal{A}, k)$ is negligible.

Semi-honest OT with Adaptive Receiver Security. 1-out-of-2 Oblivious Transfer (OT) allows a receiver to obtain exactly one of two messages from a sender where the receiver remains oblivious to the other message, and the sender is oblivious to which value was received. Please refer to Figure 1 for 2-move OT. We next define secure 2-move OT scheme with adaptive receiver security.

Definition 7. *A* k*-bit 2-move oblivious-transfer scheme* $\mathcal{OT} = (\mathsf{OT1}, \mathsf{OT2}, \mathsf{OT3})$ *is* secure with adaptive receiver security *if the following properties hold:*

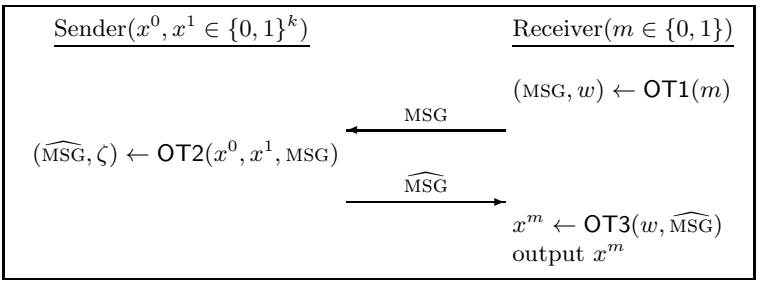


Fig. 1. A 2-move OT protocol

Correctness. *For all* $m \in \{0,1\}$*, and* $x^0, x^1 \in \{0,1\}^k$*:*

$$\Pr\left[\begin{array}{l}(\text{MSG}, w) \leftarrow \mathsf{OT1}(m);\\ \widehat{\text{MSG}} \leftarrow \mathsf{OT2}(x^0, x^1, \text{MSG})\end{array} : x^m = \mathsf{OT3}(w, \widehat{\text{MSG}})\right] \geq 1 - \mathsf{negl}(k)$$

Adaptive Receiver Security. *There exists a non-uniform, polynomial-time algorithm* $\mathcal{S}^{\text{recv}} = (\mathcal{S}_1^{\text{recv}}, \mathcal{S}_2^{\text{recv}})$ *such that for all non-uniform, polynomial-time algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*:*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\text{recv}}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

$\underline{\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\text{recv}}}(k)}$
$(m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
$(\text{MSG}, s) \leftarrow \mathcal{S}_1^{\text{recv}}(1^k)$;
$w \leftarrow \mathcal{S}_2^{\text{recv}}(s, m)$;
$b \leftarrow \mathcal{A}_2(\tau, w, \text{MSG})$;
return b

$\underline{\mathsf{Real}_{\mathcal{A}}(k)}$
$(m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
$(\text{MSG}, w) \leftarrow \mathsf{OT1}(m)$;
$b \leftarrow \mathcal{A}_2(\tau, w, \text{MSG})$;
return b

Sender Security. *There exists a non-uniform, polynomial-time algorithm* $\mathcal{S}^{\text{send}}$ *such that for all non-uniform, polynomial-time algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*:*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\text{send}}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

$\underline{\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\text{send}}}(k)}$
$(x^0, x^1, m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
$(\text{MSG}, w) \leftarrow \mathsf{OT1}(m)$;
$\widehat{\text{MSG}} \leftarrow \mathcal{S}^{\text{send}}(1^k, \text{MSG}, x^m, m)$;
$b \leftarrow \mathcal{A}_2(\tau, \text{MSG}, w, \widehat{\text{MSG}})$;
return b

$\underline{\mathsf{Real}_{\mathcal{A}}(k)}$
$(x^0, x^1, m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
$(\text{MSG}, w) \leftarrow \mathsf{OT1}(m)$;
$\widehat{\text{MSG}} \leftarrow \mathsf{OT2}(x^0, x^1, \text{MSG})$;
$b \leftarrow \mathcal{A}_2(\tau, \text{MSG}, w, \widehat{\text{MSG}})$;
return b

The construction from [8] satisfies the above properties.

Multi-message, Receiver Non-committing, Public-Key Encryption. Receiver non-committing encryption (RNCE) was introduced in [14] and further

studied in [7]. Here, we strengthen their notion to deal with multiple messages with an a priori bound $\alpha = \mathsf{poly}(k)$ on the number of messages; we call this α-message RNCE, and formally define it below.

- The randomized key-generation algorithm $\mathtt{gen}$ takes as input the security parameter and outputs a key-pair. This is denoted by: $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$. The public key pk defines the message space $\mathcal{M}$.
- The randomized encryption algorithm $\mathtt{enc}$ takes a public key pk and a message $m \in \mathcal{M}$. It returns a ciphertext $c \leftarrow \mathtt{enc}_{\mathrm{pk}}(m)$.
- The decryption algorithm $\mathtt{dec}$ takes as input a secret key sk and a ciphertext c, and returns a message $m \leftarrow \mathtt{dec}_{\mathrm{sk}}(c)$, where $m \in \mathcal{M} \cup \perp$.
- The randomized key-faking algorithm $\widetilde{\mathtt{gen}}$ takes as input the security parameter and outputs a public key as well as some auxiliary information. This is denoted by: $(\mathrm{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$.
- The fake encryption algorithm $\widetilde{\mathtt{enc}}$ takes as input a tuple (pk, z) as output by $\widetilde{\mathtt{gen}}$, and outputs a tuple of fake ciphertexts and some auxiliary information: $(c_1, \ldots, c_\alpha, z') \leftarrow \widetilde{\mathtt{enc}}(\mathrm{pk}, z)$.
- The reveal algorithm $\widetilde{\mathtt{rev}}$ takes as input a tuple $(c_1, \ldots, c_\alpha, z')$ as output by $\widetilde{\mathtt{enc}}$, and a tuple of messages $m_1, \ldots, m_\alpha \in \mathcal{M}$. It outputs a fake secret key $\mathrm{sk} \leftarrow \widetilde{\mathtt{rev}}(z', c_1, \ldots, c_\alpha, m_1, \ldots, m_\alpha)$. (Intuitively, sk is a valid-looking secret key for which c_i decrypts to m_i for all $i \in [\alpha]$.)

Definition 8. $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ *is a secure receiver non-committing encryption scheme for bounded* $\alpha = \mathsf{poly}(k)$*, if there exist non-uniform, polynomial-time algorithms* $\mathcal{S} = (\widetilde{\mathtt{gen}}, \widetilde{\mathtt{enc}}, \widetilde{\mathtt{rev}})$ *such that for all non-uniform, polynomial-time algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *we have*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

$\underline{\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k)}$
$(\mathrm{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$;
$(m_1, \ldots, m_\alpha, \tau) \leftarrow \mathcal{A}_1(1^k, \mathrm{pk})$;
$(c_1, \ldots, c_\alpha, z') \leftarrow \widetilde{\mathtt{enc}}(\mathrm{pk}, z)$;
$\mathrm{sk} \leftarrow \widetilde{\mathtt{rev}}(z', c_1, \ldots, c_\alpha, m_1, \ldots, m_\alpha)$;
$b \leftarrow \mathcal{A}_2(\tau, \mathrm{pk}, \mathrm{sk}, c_1, \ldots, c_\alpha)$;
return b.

$\underline{\mathsf{Real}_{\mathcal{A}}(k)}$
$(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$;
$(m_1, \ldots, m_\alpha, \tau) \leftarrow \mathcal{A}_1(1^k, \mathrm{pk})$;
$c_1 \leftarrow \mathtt{enc}_{\mathrm{pk}}(m_1); \ldots;$
$c_\alpha \leftarrow \mathtt{enc}_{\mathrm{pk}}(m_\alpha)$;
$b \leftarrow \mathcal{A}_2(\tau, \mathrm{pk}, \mathrm{sk}, c_1, \ldots, c_\alpha)$;
return b.

Following [14], we present a construction of a multiple-message, receiver non-committing scheme based on the DDH assumption. As in [14], we can only directly encrypt logarithmic-length messages since computation of a discrete logarithm is required by the receiver. In our case, however, we can then encrypt messages of length k by breaking the message into logarithmic-length blocks, encrypting block-by-block, and adjusting the parameter α accordingly.

- The randomized key-generation algorithm: $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$:
  Generate a group $\mathbb{G}$ of prime order q with generators $g_0, g_1, \ldots, g_\alpha$. Output the group parameters $(\mathbb{G}, q, g_0, g_1, \ldots, g_\alpha)$. Choose $x_0, \ldots, x_\alpha \leftarrow \mathbb{Z}_q$ and output the public key $\mathrm{pk} = \prod_{i=0}^{\alpha} g_i^{x_i}$ and secret key $\mathrm{sk} = (x_0, \ldots, x_\alpha)$.

- The randomized encryption algorithm $c \leftarrow \mathtt{enc}_{\mathrm{pk}}(m)$:
  View m as an element of $\mathbb{Z}_q$. Choose $r \in \mathbb{Z}_q$ at random and output the ciphertext $(g_0^r, \ldots, g_\alpha^r, P^r \cdot g_0^m)$.
- The decryption algorithm $m \leftarrow \mathtt{dec}_{\mathrm{sk}}(c)$:
  Parse $c = (A_0, \ldots, A_\alpha, B)$, and compute $C = B / \prod_i A_i^{x_i}$. Set $m = \log_{g_0} C$. (We assume m is small enough so that this computation can be done efficiently.)
- The randomized key-faking algorithm $(\mathrm{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$:
  Generate a group $\mathbb{G}$ of prime order q with generators $g_0, g_1, \ldots, g_\alpha$ where $g_i = g_0^{\sigma_i}$ and σ_i is known. Output the group parameters $(\mathbb{G}, q, g_0, g_1, \ldots, g_\alpha)$. Choose $x_0, \ldots, x_\alpha \leftarrow \mathbb{Z}_q$ and output the public key $\mathrm{pk} = \prod_{i=0}^{\alpha} g_i^{x_i}$.
- The fake encryption algorithm $\widetilde{\mathtt{enc}}$:
  For $i \in \{1, \ldots, \alpha\}$, choose $r_{i,0}, \ldots, r_{i,\alpha}, r'_i \leftarrow \mathbb{Z}_q$, and output the ciphertext $(g_0^{r_{i,0}}, \ldots, g_\alpha^{r_{i,\alpha}}, g_0^{r'_i})$.
- The reveal algorithm $\mathrm{sk} \leftarrow \widetilde{\mathtt{rev}}(z', c_1, \ldots, c_\alpha, m_1, \ldots, m_\alpha)$:
  Output $x'_0, \ldots, x'_\alpha$ satisfying

$$x_0 - x'_0 + \sum_{i=1}^{\alpha} (x_i - x'_i)\sigma_i = 0$$

$$\forall j \in \{1, \ldots, \alpha\} : \; m_j + x'_0 r_{j,0} + \sum_{i=1}^{\alpha} x'_i r_{j,i} \sigma_i = r'_j.$$

  Note that this is a system of $\alpha + 1$ equations in $\alpha + 1$ unknowns, and has a solution with all but negligible probability.

4.2 Adaptively Secure Circuit-Private ℓ-Homomorphic Encryption

Our construction of a circuit-private ℓ-homomorphic encryption scheme $\mathcal{HE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$. is based on (i) a projective garbling scheme (Gb, En, De, Ev, ev) leaking only the circuit size [19,16,5], (ii) a receiver adaptively secure semi-honest OT protocol ($\mathsf{OT1}, \mathsf{OT2}, \mathsf{OT3}$), and (iii) an ℓ-message receiver non-committing public-key encryption scheme ($\mathtt{gen}, \mathtt{enc}, \mathtt{dec}$).

- The key-generation algorithm $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$:
  Compute $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$, and set $pk := \mathrm{pk}$ and $sk := \mathrm{sk}$.
- The encryption algorithm $c \leftarrow \mathsf{Enc}_{pk}(m)$:
  Upon input $m \in \{0,1\}$, compute $(\mathrm{MSG}, w) \leftarrow \mathsf{OT1}(m)$. Compute $e \leftarrow \mathtt{enc}_{\mathrm{pk}}(w)$ and output $c := (\mathrm{MSG}, e)$.
- The decryption algorithm $m \leftarrow \mathsf{Dec}_{sk}(c)$:
  - If $c = (\mathrm{MSG}, e)$, compute $w \leftarrow \mathtt{dec}_{\mathrm{sk}}(e)$. Then run $\mathsf{OT1}$ with inputs 0 and 1, and randomness w, and output m for which $(\mathrm{MSG}, w) \leftarrow \mathsf{OT1}(m)$.
  - If $c = (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$ then for all $i \in [\ell]$, further parse $\hat{c}_i$ into $(\widehat{\mathrm{MSG}}_i, e_i)$, compute $w_i \leftarrow \mathtt{dec}_{\mathrm{sk}}(e_i)$, and compute $X_i^{m_i} \leftarrow \mathsf{OT3}(w_i, \widehat{\mathrm{MSG}}_i)$. This gives the garbled input $X = (X_1^{m_1}, \ldots, X_\ell^{m_\ell})$. Output $\mathsf{De}(\mathsf{d}, \mathsf{Ev}(\hat{C}, X))$ as message m.

– The evaluation algorithm $c^* \leftarrow \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$:
  - Let $\mathsf{Gb}(1^k, C) \rightarrow (\hat{C}, \mathsf{e}, \mathsf{d})$. Parse the encoding function represented by string e as $(X_1^0, X_1^1, \ldots, X_\ell^0, X_\ell^1)$.
  - Parse c_i into (MSG_i, e_i) for $i \in [\ell]$. Compute $\widehat{\text{MSG}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \text{MSG}_i)$, and set $\hat{c}_i := (\widehat{\text{MSG}}_i, e_i)$, for all $i \in [\ell]$.
  - Finally, set $c^* := (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$.

Theorem 2. *Construction $\mathcal{HE}$ presented above is an adaptively secure circuit-private ℓ-homomorphic encryption.*

Proof. We show below that the construction $\mathcal{HE}$ is a secure ℓ-homomorphic encryption that satisfies correctness, circuit privacy, and adaptive security.

Correctness. It is easy to verify the correctness. To compute $\mathsf{Dec}_{sk}(c)$ for $c = \mathsf{Enc}_{pk}(m)$ where $c = (\text{MSG}, e)$, we first compute $w' \leftarrow \mathsf{dec}_{\text{sk}}(e)$. Then run $\mathsf{OT1}$ twice with both inputs 0 and 1, and randomness w'. Output m' where $(\text{MSG}, w') \leftarrow \mathsf{OT1}(m)$. Given the fact that $(\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ is correct, it holds that $w = \mathsf{dec}_{\text{sk}}(\mathsf{enc}_{\text{pk}}(w))$, i.e., $w = w'$; furthermore, given the fact that the OT scheme is correct, it holds that $(\text{MSG}, w) = \mathsf{OT1}(m)$, i.e., $m = m'$. Therefore, we have $\mathsf{Dec}_{sk}(c) = m$ for $c = \mathsf{Enc}_{pk}(m)$.

To compute $\mathsf{Dec}_{sk}(c)$ for $c = \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$ where $c = (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$, parse $\hat{c}_i$ as $\hat{c}_i = (\widehat{\text{MSG}}_i, e_i)$ for all $i \in [\ell]$. We first compute $w_i' \leftarrow \mathsf{dec}_{\text{sk}}(e_i)$, $\hat{X}_i \leftarrow \mathsf{OT3}(w_i', \widehat{\text{MSG}}_i)$. Then we use the input key strings $X = (\hat{X}_1, \ldots, \hat{X}_\ell)$ to evaluate the garbled circuit $\hat{C}$ as $\mathsf{Ev}(\hat{C}, X)$ and obtain $C(\hat{m}_1, \ldots, \hat{m}_\ell)$. Given the fact that $(\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ is correct, we have $w_i = w_i'$; furthermore, given the fact that the OT is correct, $\hat{X}_i = X_i^{m_i}$; also, given the fact that the garbling scheme is correct, we have $C(m_1, \ldots, m_\ell) = \mathsf{De}(\mathsf{d}, \mathsf{Ev}(\hat{C}, X))$. Therefore, we have $\mathsf{Dec}_{sk}(c) = m$ for $c = \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$ and $m = C(m_1, \ldots, m_\ell)$.

Circuit privacy. The property of circuit privacy follows from the security of the garbling scheme and the sender security of the OT. By the security of the garbling scheme, we have that there exists a simulator $\mathcal{S}^{\mathcal{G}}$ on input the security parameter, output of the function and the side information function Φ outputs (F, X, d) except with negligible probability. As mentioned before, $\Phi(C) = |C|$ in our construction.

Let us construct a simulator $\mathcal{S}$ as follows:

– Upon receiving $(1^k, pk, |C|, C(m_1, \ldots, m_\ell), c_1, \ldots, c_\ell)$ where $c_i = \mathsf{Enc}_{pk}(m_i)$ for $i \in [\ell]$, the simulator $\mathcal{S}$ runs the simulator $\mathcal{S}^{\mathcal{G}}$ of the garbling scheme to obtain
$(F, X, \mathsf{d}) \leftarrow \mathcal{S}^{\mathcal{G}}(1^k, C(m_1, \ldots, m_\ell), |C|)$.
– Set $\hat{C} = F$. Parse X as $(\hat{X}_1, \ldots, \hat{X}_\ell)$.
– To compute the ciphertext $\hat{c}_i$, parse c_i into (MSG_i, e_i) as in the construction. Then compute $\widehat{\text{MSG}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \text{MSG}_i, \hat{X}_i)$ using the OT-simulator $\mathcal{S}^{\text{send}}$ for sender security, and set $\hat{c}_i := (\widehat{\text{MSG}}_i, e_i)$, for all $i \in [\ell]$.

– Finally, set $c^* := (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$.

Next, we develop a sequence of hybrids to show that Definition 4 is satisfied.

Hybrid $\mathbf{H}_0$: As in the real scheme, we run the evaluation algorithm Eval to compute c^*, i.e, $c^* \leftarrow \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$ where $c^* = (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$. Concretely, we use the projective garbling scheme, on input C, compute $(\hat{C}, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, C)$; then we parse the encoding function represented by string e as $(X_1^0, X_1^1, \ldots, X_\ell^0, X_\ell^1)$, and parse c_i into (MSG_i, e_i) for $i \in [\ell]$; after that, we compute $\widehat{\text{MSG}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \text{MSG}_i)$, and set $\hat{c}_i := (\widehat{\text{MSG}}_i, e_i)$, for all $i \in [\ell]$; finally, set $c^* := (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$. The adversary $\mathcal{A}$ is given c^* as well as the input m, circuit C and the secret key sk.

Hybrid $\mathbf{H}_{1,j}$: For $j \in [0 \ldots \ell]$, the hybrid $\mathbf{H}_{1,j}$ is the same as the Hybrid $\mathbf{H}_0$ except the following:

For all $i \in [j]$, parse c_i into (MSG_i, e_i). Compute $\widehat{\text{MSG}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \text{MSG}_i, X_i^{m_i})$, using the OT simulator for sender security and set $\hat{c}_i := (\widehat{\text{MSG}}_i, e_i)$.

We argue that for $j \in [1, \ldots, \ell]$, the hybrids $\mathbf{H}_{1,j}$ and $\mathbf{H}_{1,j+1}$ are computationally indistinguishable under the assumption that the OT satisfies sender security. If there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids with non-negligible probability, then we can construct an adversary $\mathcal{B}$ who breaks the sender security of the OT as follows.

The adversary $\mathcal{B}$ acts as follows:

– run $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$, i.e., run $(\text{pk}, \text{sk}) \leftarrow \mathsf{gen}(1^k)$ and set $pk := \text{pk}$, $sk := \text{sk}$.
– Choose $m = (m_1, \ldots, m_\ell)$. For all $i \in [\ell]$, compute $c_i \leftarrow \mathsf{Enc}_{pk}(m_i)$, i.e., compute $(\text{MSG}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$, and $e_i \leftarrow \mathsf{enc}_{\text{pk}}(w_i)$. Set $c_i := (\text{MSG}_i, e_i)$;
– Using the projective garbling scheme on a circuit C, let $(\hat{C}, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, C)$. Parse the encoding function represented by string e as $(X_1^0, X_1^1, \ldots, X_\ell^0, X_\ell^1)$;
– for all $i \in [j-1]$, parse c_i into (MSG_i, e_i). Then compute $\widehat{\text{MSG}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \text{MSG}_i, X_i^{m_i})$;
– for $i = j$, output (m_i, X_i^0, X_i^1). Parse c_i into (MSG_i, e_i) and obtain $\widehat{\text{MSG}}_i$, where it is either generated by $\mathcal{S}^{\text{send}}$, i.e., $\widehat{\text{MSG}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \text{MSG}_i, X_i^{m_i})$, or by the OT scheme honestly, i.e., $\widehat{\text{MSG}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \text{MSG}_i)$;
– for all $i \in [j+1, \ell]$, parse c_i into (MSG_i, e_i). Then compute $\widehat{\text{MSG}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \text{MSG}_i)$;
– for all $i \in [\ell]$, set $\hat{c}_i := (\widehat{\text{MSG}}_i, e_i)$;
– Return $(m, sk, pk, C, \hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$ to the internally simulated $\mathcal{A}$.

When $i = j$, if $\widehat{\text{MSG}}_i$ obtained by $\mathcal{B}$ is generated by $\mathcal{S}^{\text{send}}$, then $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j}$. Otherwise, if $\widehat{\text{MSG}}_i$ is generated by the OT scheme honestly, then $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j-1}$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can distinguish Ideal from Real as in Definition 7 for sender security. This contradicts our assumption that the OT is sender-secure. Therefore, Hybrid $\mathbf{H}_{1,j-1}$ and Hybrid $\mathbf{H}_{1,j}$ are indistinguishable.

Note that $\mathbf{H}_0$ is identical to $\mathbf{H}_{1,0}$. Since Hybrids $\mathbf{H}_{1,j-1}$ and $\mathbf{H}_{1,j}$ are indistinguishable as argued above, we also have that Hybrid $\mathbf{H}_0$ is computationally indistinguishable from Hybrid $\mathbf{H}_{1,\ell}$.

Hybrid $\mathbf{H}_2$: This is the same as Hybrid $\mathbf{H}_{1,\ell}$ except the following: Run the simulator $\mathcal{S}^{\mathcal{G}}$ for the projective garbling scheme to obtain the garbled circuit, input and the decoding function, i.e., $(F, X, \mathsf{d}) \leftarrow \mathcal{S}^{\mathcal{G}}(1^k, C(m_1, \ldots, m_\ell), |C|)$. Parse X as $(\hat{X}_1, \ldots, \hat{X}_\ell)$, and set $\hat{C} = F$. We note that this is exactly the output produced by the simulator $\mathcal{S}$ for circuit privacy.

The hybrids $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are indistinguishable under the assumption that $\mathcal{G}$ is a secure garbling scheme. If there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids with non-negligible probability, then we can construct an adversary $\mathcal{B}$ who breaks the security of the garbling scheme as defined in Definition 6.

Consider an adversary $\mathcal{B}$ who acts as follows:

- run $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$;
- Choose $m = (m_1, \ldots, m_\ell)$. For all $i \in [\ell]$, compute $c_i \leftarrow \mathsf{Enc}_{pk}(m_i)$;
- Choose a circuit C. Make a GARBLE query on (C, m) to obtain the challenge (F, X, d). Set $\hat{C} = F$.
- Parse X as $(\hat{X}_1, \ldots, \hat{X}_\ell)$. For all $i \in [\ell]$, parse c_i into (MSG_i, e_i), and compute $\widehat{\text{MSG}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \text{MSG}_i, \hat{X}_i)$; Set $\hat{c}_i = (\widehat{\text{MSG}}_i, e_i)$;
- Return $(m, sk, pk, C, \hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$ to the internally simulated $\mathcal{A}$.

When $\mathcal{B}$'s challenge (F, X, d) is generated honestly, then the internally simulated $\mathcal{A}$ interacts with $\mathbf{H}_{1,\ell}$. On the other hand, when $\mathcal{B}$'s challenge is generated by $\mathcal{S}^{\mathcal{G}}$, the simulated $\mathcal{A}$ interacts with $\mathbf{H}_2$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can gain a non-negligible advantage as in Definition 6. However, this is a contradiction to our assumption that $\mathcal{G}$ is a secure garbling scheme. Therefore, Hybrid $\mathbf{H}_{1,\ell}$ and Hybrid $\mathbf{H}_2$ are indistinguishable.

Thus, we have that the hybrids $\mathbf{H}_0$ and $\mathbf{H}_2$ are indistinguishable which implies that the scheme satisfies the circuit privacy requirement as defined in Definition 4.

ADAPTIVE SECURITY. Next we give the proof idea for proving the adaptive security as defined in Definition 5. We construct the simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ as follows. The simulator is based on the algorithms $(\widetilde{\mathtt{gen}}, \widetilde{\mathtt{enc}}, \widetilde{\mathtt{rev}})$ of the ℓ-RNCE scheme, and the simulator $(\mathcal{S}_1^{\text{recv}}, \mathcal{S}_2^{\text{recv}})$ of the OT scheme.

- $(pk, c_1, \ldots, c_\ell, s) \leftarrow \mathcal{S}_1(1^k)$:

  Compute $(\text{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$, and set $pk := \text{pk}$. Compute $(e_1, \ldots, e_\ell, z') \leftarrow \widetilde{\mathtt{enc}}(\text{pk}, z)$.

  For all $i \in [\ell]$, compute $(\text{MSG}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\text{recv}}(1^k)$, and set $c_i := (\text{MSG}_i, e_i)$. Store all information into state s.

– $sk \leftarrow \mathcal{S}_2(s, m_1, \ldots, m_\ell)$:
Upon obtaining $(m_1, \ldots, m_\ell)$, recover $\{\gamma_i\}_{i\in[\ell]}$ from the state s.
For all $i \in [\ell]$, compute $w_i \leftarrow \mathcal{S}_2^{\text{recv}}(\text{MSG}_i, \gamma_i, m_i)$.
Compute $\text{sk} \leftarrow \widetilde{\texttt{rev}}(z', \{e_i, w_i\}_{i\in[\ell]})$, and set $sk := \text{sk}$.

Next, we develop a sequence of hybrids to show that the real experiment defined in Definition 5 is indistinguishable from the ideal experiment.

Hybrid $\mathbf{H}_0$: This is the real experiment.
Compute $(\text{pk}, \text{sk}) \leftarrow \texttt{gen}(1^k)$ and set $pk := \text{pk}$, $sk := \text{sk}$. The adversary $\mathcal{A}$ outputs $(m_1, \ldots, m_\ell)$ after seeing the public key pk. Then for all $i \in [\ell]$, do the following: compute $(\text{MSG}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$, $e_i \leftarrow \texttt{enc}_{\text{pk}}(w_i)$. Set $c_i := (\text{MSG}_i, e_i)$. Return $(pk, c_1, \ldots, c_\ell, sk)$ to the adversary.

Hybrid $\mathbf{H}_{1,j}$: Let $j \in [0, \ldots, \ell]$. The hybrid is the same as the Hybrid $\mathbf{H}_0$ except the following:
For all $i \in [j]$, compute $(\text{MSG}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\text{recv}}(1^k)$. Then, compute $w_i \leftarrow \mathcal{S}_2^{\text{recv}}(\text{MSG}_i, \gamma_i, m_i)$.
We argue that for $j \in [1, \ldots, \ell]$, the hybrids $\mathbf{H}_{1,j-1}$ and $\mathbf{H}_{1,j}$ are computationally indistinguishable under the assumption that the OT satisfies adaptive receiver security. Assume there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids. For all $\mathcal{S}^{\text{recv}}$, we next show how to construct $\mathcal{B}$ to distinguish between the Real experiment and the Ideal experiment as in Definition 7.
$\mathcal{B}$ internally simulates $\mathcal{A}$ and receives $(m_1, \ldots, m_\ell)$ from it. Then $\mathcal{B}$ carries out the following:
– run $(\text{pk}, \text{sk}) \leftarrow \texttt{gen}(1^k)$, and set $pk := \text{pk}$, $sk := \text{sk}$;
– for all $i \in [j-1]$, use the OT simulator to obtain $(\text{MSG}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\text{recv}}(1^k)$ and $w_i \leftarrow \mathcal{S}_2^{\text{recv}}(\text{MSG}_i, \gamma_i, m_i)$. Compute $e_i \leftarrow \texttt{enc}_{\text{pk}}(w_i)$;
– for $i = j$, obtain the pair (MSG_i, w_i), where the pair is generated by either $\mathcal{S}^{\text{recv}} = (\mathcal{S}_1^{\text{recv}}, \mathcal{S}_2^{\text{recv}})$, i.e., $(\text{MSG}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\text{recv}}(1^k)$ and $w_i \leftarrow \mathcal{S}_2^{\text{recv}}(\text{MSG}_i, \gamma_i, m_i)$, or generated by the OT scheme honestly, i.e., $(\text{MSG}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$;
– for all $i \in [j+1, \ell]$, compute $(\text{MSG}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$ and $e_i \leftarrow \texttt{enc}_{\text{pk}}(w_i)$.
– for all $i \in [\ell]$, set $c_i := (\text{MSG}_i, e_i)$;
– Return $(pk, c_1, \ldots, c_\ell, sk)$ to the internally simulated $\mathcal{A}$.
Note that when $i = j$, if the pair (MSG_i, w_i) that $\mathcal{B}$ obtains are generated by $\mathcal{S}^{\text{recv}} = (\mathcal{S}_1^{\text{recv}}, \mathcal{S}_2^{\text{recv}})$, then $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j}$, while if the pair (MSG_i, w_i) that $\mathcal{B}$ obtains are generated by the OT scheme honestly, $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j-1}$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can distinguish Ideal from Real as in Definition 7 for defining adaptive receiver security. This leads to a contradiction to our assumption that the OT has adaptive receiver security. Therefore, Hybrid $\mathbf{H}_{1,j-1}$ and Hybrid $\mathbf{H}_{1,j}$ are indistinguishable.
Note that $\mathbf{H}_0$ is identical to $\mathbf{H}_{1,0}$. Since Hybrid $\mathbf{H}_{1,j-1}$ and Hybrid $\mathbf{H}_{1,j}$ are indistinguishable, as argued above, we also have that $\mathbf{H}_0$ is computationally indistinguishable from $\mathbf{H}_{1,\ell}$.

Hybrid $\mathbf{H}_2$: The hybrid is the same as Hybrid $\mathbf{H}_{1,\ell}$ except the following:

Compute $(\text{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$, and set $pk := \text{pk}$. Compute $(e_1, \ldots, e_\ell, z') \leftarrow \widetilde{\mathtt{enc}}(\text{pk}, z)$. Compute $\text{sk} \leftarrow \widetilde{\mathtt{rev}}(z', \{e_i, w_i\}_{i\in[\ell]})$, and set $sk := \text{sk}$. We note that this is exactly the ideal experiment.

We argue that $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are computationally indistinguishable based on the security of the ℓ-RNCE scheme. Assume there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids. We next show how to construct an adversary $\mathcal{B}$ that can break the security of the ℓ-RNCE scheme.

$\mathcal{B}$ receives pk, and internally runs $\mathcal{A}$. Upon receiving $(m_1, \ldots, m_\ell)$ from $\mathcal{A}$, $\mathcal{B}$ computes $(\text{MSG}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$ for all $i \in [\ell]$ and outputs $(w_1, \ldots, w_\ell)$ to its challenger. Upon receiving from its challenger $(e_1, \ldots, e_\ell, \text{sk})$, $\mathcal{B}$ sets $c_i := (\text{MSG}_i, e_i)$ for all $i \in [\ell]$. $\mathcal{B}$ returns $(pk, c_1, \ldots, c_\ell, sk)$ to $\mathcal{A}$ and returns $\mathcal{A}$'s output as its own output.

When $\mathcal{B}$'s received tuple $(\text{pk}, e_1, \ldots, e_\ell, \text{sk})$ is generated by $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ honestly, then the internally simulated $\mathcal{A}$ interacts with $\mathbf{H}_{1,\ell}$. On the other hand, when $\mathcal{B}$'s received tuple is generated by $(\widetilde{\mathtt{gen}}, \widetilde{\mathtt{enc}}, \widetilde{\mathtt{rev}})$, i.e, $(\text{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$, $(e_1, \ldots, e_\ell, z') \leftarrow \widetilde{\mathtt{enc}}(\text{pk}, z)$, and $\text{sk} \leftarrow \widetilde{\mathtt{rev}}(z', \{e_i, w_i\}_{i\in[\ell]})$, $\mathcal{A}$ interacts with $\mathbf{H}_2$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can distinguish Ideal from Real as in Definition 8. This contradicts the security of the ℓ-RNCE scheme. Therefore, Hybrids $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are indistinguishable.

Based on the above argument we have $\mathbf{H}_0$ and $\mathbf{H}_{1,\ell}$ are indistinguishable, and $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are indistinguishable. Therefore $\mathbf{H}_0$ and $\mathbf{H}_2$ are indistinguishable. Note that Hybrid $\mathbf{H}_2$ is exactly the ideal experiment, and $\mathbf{H}_0$ is the real experiment. We now have that the ideal and the real experiments are indistinguishable. This implies that the scheme satisfies adaptive security.

References

1. Arora, S., Barak, B.: Computational Complexity: A Modern Approach. Cambridge University Press (2009)
2. Barak, B., Haitner, I., Hofheinz, D., Ishai, Y.: Bounded Key-Dependent Message Security. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 423–444. Springer, Heidelberg (2010)
3. Beaver, D.: Plug and Play Encryption. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 75–89. Springer, Heidelberg (1997)
4. Beaver, D., Haber, S.: Cryptographic Protocols Provably Secure against Dynamic Adversaries. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 307–323. Springer, Heidelberg (1993)
5. Bellare, M., Hoang, V.T., Rogaway, P.: Foundations of garbled circuits. In: ACM Conference on Computer and Communications Security, pp. 784–796 (2012)
6. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: 28th Annual ACM Symposium on Theory of Computing (STOC), pp. 639–648. ACM Press (1996)
7. Canetti, R., Halevi, S., Katz, J.: Adaptively-Secure, Non-interactive Public-Key Encryption. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 150–168. Springer, Heidelberg (2005)

8. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: 34th ACM STOC Annual ACM Symposium on Theory of Computing (STOC), pp. 494–503. ACM Press (2002)
9. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Improved Non-committing Encryption with Applications to Adaptively Secure Protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 287–302. Springer, Heidelberg (2009)
10. Damgård, I., Nielsen, J.B.: Improved Non-committing Encryption Schemes Based on a General Complexity Assumption. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 432–450. Springer, Heidelberg (2000)
11. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: 41st Annual ACM Symp. on Theory of Computing (STOC), pp. 169–178. ACM Press (2009)
12. Gentry, C.: A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009), `http://crypto.stanford.edu/craig`
13. Gentry, C., Halevi, S., Vaikuntanathan, V.: i-Hop Homomorphic Encryption and Rerandomizable Yao Circuits. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 155–172. Springer, Heidelberg (2010)
14. Jarecki, S., Lysyanskaya, A.: Adaptively Secure Threshold Cryptography: Introducing Concurrency, Removing Erasures (Extended Abstract). In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 221–242. Springer, Heidelberg (2000)
15. Katz, J., Ostrovsky, R.: Round-Optimal Secure Two-Party Computation. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 335–354. Springer, Heidelberg (2004)
16. Lindell, Y., Pinkas, B.: A proof of security of Yao's protocol for two-party computation. Journal of Cryptology 22(2), 161–188 (2009)
17. Nielsen, J.B.: Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 111–126. Springer, Heidelberg (2002)
18. Rivest, R., Adleman, L., Dertouzos, M.: On data banks and privacy homomorphisms. In: Foundations of Secure Computation, pp. 169–177. Academic Press (1978)
19. Yao, A.C.: How to generate and exchange secrets. In: 27th Annual Symposium on Foundations of Computer Science (FOCS), pp. 162–167. IEEE (1986)

Chosen Ciphertext Secure Keyed-Homomorphic Public-Key Encryption

Keita Emura[1], Goichiro Hanaoka[2], Go Ohtake[3], Takahiro Matsuda[2], and Shota Yamada[4]

[1] National Institute of Information and Communications Technology (NICT), Japan
k-emura@nict.go.jp
[2] Japan Broadcasting Corporation, Japan
ohtake.g-fw@nhk.or.jp
[3] National Institute of Advanced Industrial Science and Technology (AIST), Japan
{t-matsuda,hanaoka-goichiro}@aist.go.jp
[4] The University of Tokyo, Japan
yamada@it.k.u-tokyo.ac.jp

Abstract. In homomorphic encryption schemes, anyone can perform homomorphic operations, and therefore, it is difficult to manage when, where and by whom they are performed. In addition, the property that anyone can "freely" perform the operation inevitably means that ciphertexts are malleable, and it is well-known that adaptive chosen ciphertext (CCA) security and the homomorphic property can never be achieved simultaneously. In this paper, we show that CCA security and the homomorphic property can be simultaneously handled in situations that the user(s) who can perform homomorphic operations on encrypted data should be controlled/limited, and propose a new concept of homomorphic public-key encryption, which we call *keyed-homomorphic public-key encryption* (KH-PKE). By introducing a secret key for homomorphic operations, we can control who is allowed to perform the homomorphic operation. To construct KH-PKE schemes, we introduce a new concept, a *homomorphic transitional universal hash family*, and present a number of KH-PKE schemes through hash proof systems. We also present a practical construction of KH-PKE from the DDH assumption. For ℓ-bit security, our DDH-based scheme yields only ℓ-bit longer ciphertext size than that of the Cramer-Shoup PKE scheme.

Keywords: homomorphic public key encryption, CCA2 security, hash proof system.

1 Introduction

1.1 Background and Motivation

In homomorphic encryption schemes, homomorphic operations can be performed on encrypted plaintexts without decrypting the corresponding ciphertexts. Owing to this attractive property, several homomorphic public key encryption (PKE)

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 32–50, 2013.

schemes have been proposed [13,16,25]. Furthermore, fully homomorphic encryption (FHE) that allows a homomorphic operation with respect to any circuit, has recently been proposed by Gentry [15]. This has had a resounding impact not only in the cryptographic research community, but also in the business community. One of the reasons for such a big impact is that FHE is suitable for ensuring security in cloud environments (e.g., encrypted data stored in a database can be updated without any decryption procedure).

Improvement in the security of homomorphic encryption will lead to wider deployment of cloud-type applications, whereas the property that anyone can "freely" perform homomorphic operations inevitably means that ciphertexts are malleable. Therefore, it is well-known that adaptive chosen ciphertext (CCA2) security and the homomorphic property can never be achieved simultaneously. In other words, security is sacrificed in exchange for the homomorphic property. Although several previous works (e.g., [1,6,17,26,27]) have attempted to construct homomorphic PKE schemes that offer security close to CCA2 security while retaining the homomorphic property, these schemes only guarantee security at limited levels. Note that not all functionalities of conventional homomorphic encryption are indispensable for real-world applications, and therefore there is the possibility of realizing a desirable security level by appropriately selecting the functionalities of conventional homomorphic encryption.

Here, we point out that the underlying cause of the incompatibility of CCA2 security and the homomorphic property, lies in the setting that any user can use the homomorphic property, and it is worth discussing whether the free availability of homomorphic operations is an indispensable functionality in real-world applications. For example, consider the situation where some data encrypted by a homomorphic PKE scheme is stored in a public database (e.g., public cloud computing environment) and it is modified by homomorphic operations. If anyone can perform a homomorphic operation, then it is hard to reduce the risk of unexpected changes to the encrypted data in the database in which resources are dynamically allocated. Even in a closed environment (e.g., private cloud computing environment), we cannot rule out the possibility of unexpected changes to a user's data by any user who is authorized to access the database. Of course, it is possible to protect such unexpected modification of encrypted data by setting access permissions of each user appropriately. However, in cloud environments, security of outsourced data storages may not be assured. Therefore, such access control functionality should be included in encrypted data itself.

From the above consideration, we see that the property that anyone can perform homomorphic operations not only inhibits the realization of CCA2 security, but also introduces the problem of unexpected modification of encrypted data.

1.2 Our Contribution

In this paper, we show that CCA2 security and the homomorphic property can be simultaneously handled in situations that the user(s) who can perform homomorphic operations should be controlled. Specifically, we propose a new concept of homomorphic PKE, which we call *keyed-homomorphic public-key encryption*

(KH-PKE), that has the following properties: (1) in addition to a conventional public/decryption key pair (pk, sk_d), another secret key for the homomorphic operation (denoted by sk_h) is introduced, (2) homomorphic operations cannot be performed without using sk_h, and (3) ciphertexts cannot be decrypted using only sk_h. Interestingly, KH-PKE implies conventional homomorphic PKE, since the latter can be implemented by publishing sk_h of KH-PKE.

To construct KH-PKE schemes, we introduce a new concept, a *homomorphic transitional universal hash family*, which can be constructed from any diverse group system [11], and present a number of KH-PKE schemes through hash proof systems (HPSs) [11].

Our Scenarios: Here we introduce situations that the user(s) who can perform homomorphic operations should be controlled/limited. For example, in the situation where encrypted data is stored in a public database, an owner of the data gives sk_h to the database manager, who updates the encrypted data after authentication of users. No outsider can modify the encrypted data in the public database without having sk_h. As another example, by considering sk_h, a counter can take over the role of aggregating an audience survey, voting, and so on. An advantage of separating ballot-counting and ballot-aggregation is that it is possible to reduce the aggregation costs of the counter and to collect the ballot results for individual areas, groups, and communities.

Naive Construction and Its Limitations: One might think that the functionality and the security of KH-PKE can be achieved by using the following double encryption methodology: A ciphertext of an "inner" CCA1 secure homomorphic PKE scheme is encrypted by an "outer" CCA2 secure PKE scheme, and the decryption key of the CCA2 secure PKE scheme is used as sk_h.

However, this naive construction is not secure in the sense of our security definition. Taking into account the exposure of the homomorphic operation key sk_h, an adversary can request sk_h to be exposed in our security definition. The adversary is allowed to use the decryption oracle "even after the challenge phase", just before the adversary requests sk_h. However, no such decryption query is allowed in the CCA1 security of the underlying "inner" scheme, and therefore it seems hard to avoid this problem.

Even if we turn a blind eye to the above problem, it is obvious that efficiency of the naive construction is roughly equal to the total costs of the building block PKE schemes. On the other hand, the efficiency of our KH-PKE instantiations is very close to the corresponding (non-keyed-homomorphic) PKE schemes based on HPSs. In particular, the efficiency of our decisional Diffie-Hellman (DDH)-based KH-PKE scheme is comparably efficient as the Cramer-Shoup PKE (CS) scheme [9], where for ℓ-bit security, our scheme yields only ℓ-bit longer ciphertext size than that of the CS PKE scheme. Whereas the naive construction yields 5ℓ-bit longer ciphertext size even if we choose the Kurosawa-Desmedt PKE scheme [23] and the Cramer-Shoup lite PKE scheme [9] that seems the most efficient combination under the DDH assumption. We give the comparison in Section 5.

To sum up, our construction is superior than the naive construction from both security and efficiency perspectives.

Our Methodology: As a well-known result, CCA2-secure PKE can be constructed via a HPS [11] which has two projective hash families as its internal structure: A *universal*$_2$ projective hash and a *smooth* projective hash. Also it is known that a weaker property of universal$_2$, that is called *universal*$_1$ property, was shown to be useful for achieving CCA1-secure PKE [22], and universal$_1$ property (and smooth property also) does not contradict the homomorphic property. That is, our aim seems to be achieved if we can establish a switching mechanism from universal$_2$ to universal$_1$. Moreover, we can simulate the decryption oracle even after the challenge phase and after revealing sk_h since the simulator knows all secret keys in the security proof.

In this paper, we show such a mechanism (which we call *homomorphic transitional universal hash family*) can be obtained from any diverse group system [11], and then we propose a generic construction of KH-PKE based on a homomorphic transitional universal HPS. Moreover, as an implication result, KH-PKE is implied by CPA-secure homomorphic PKE (with cyclic ciphertext space) which implies diverse group systems [19].

Instantiations: According to our methodology, we present a number of KH-PKE schemes from various major cryptographic assumptions such as the DDH assumption, the decisional composite residuosity (DCR) assumption, the decisional linear (DLIN) assumption, the decisional bilinear Diffie-Hellman (DBDH) assumption, and the decisional quadratic residuosity (DQR) assumption. This means that it is not difficult to extend all existing HPS to have the homomorphic transitional property, and thus a homomorphic transitional HPS is not a significantly stronger primitive in practice, compared to an ordinary HPS.

In this paper, we present a practical DDH-based KH-PKE scheme. Other KH-PKE schemes based on the DCR assumption and the DQR assumption from the Cramer-Shoup HPSs [11], based on the DLIN assumption from the Shacham HPS [28], and based on the DBDH assumption from the Galindo-Villar HPS [12], and an identity-based analogue of KH-PKE, called *keyed-homomorphic identity-based encryption* (KH-IBE) and its concrete construction from the Gentry IBE scheme [14] will be given in the full version of this paper.

1.3 Related Work

Several previous works have attempted to construct homomorphic PKE schemes that provide security close to CCA2 security, while retaining the homomorphic property. Canetti et al. [6] considered the notion of replayable CCA (RCCA), which leaves a room for an adversary who is given two ciphertexts (C, C'), to gain information on whether C' was derived from C. (Modified RCCA notions have also been proposed [17,26].) In the RCCA security game, the decryption oracle given to an adversary is restricted in such a way that the challenge ciphertext and ciphertexts derived from the challenge ciphertext cannot be queried to the oracle.

Similarly, in benignly-malleable (gCCA) security [1,29], ciphertexts related to the challenge one cannot be input to the decryption oracle. Therefore, RCCA and gCCA are strictly weaker notions than CCA2, and may not be sufficient if the encryption scheme is used as a building block for higher level protocols/systems.

In [27], Prabhakaran and Rosulek proposed homomorphic CCA (HCCA) security, where only the expected operation, and no other operations, can be performed for any ciphertext. (Targeted malleability, which is a similar concept to HCCA, was considered in [4].) In addition, they also showed that CCA2, gCCA, and RCCA are special cases of HCCA. Note that HCCA does not handle the homomorphic property and CCA2 security simultaneously, since anyone can perform the homomorphic operation. Chase et al. [8] showed that controlled-malleable non-interactive zero-knowledge can be used as a general tool for achieving RCCA and HCCA security.

Embedding a ciphertext of homomorphic PKE into that of CCA2-secure PKE, was considered in [24,3]. Note that their embedding encryption methods are nothing more than protecting a ciphertext of homomorphic PKE by that of CCA2 PKE, and therefore no homomorphic operation can be performed on embedded ciphertexts. Meanwhile, in our KH-PKE, even after performing the homomorphic operation, a ciphertext is still valid.

Barbosa and Farshim [2] proposed delegatable homomorphic encryption (DHE). The difference between KH-PKE and DHE is that in DHE a trusted authority (TA) issues a token to control the capability to evaluate circuits f over encrypted data M to untrusted evaluators. Furthermore, their security definitions of DHE (input/output privacy (TA-IND-CPA) and evaluation security (IND-EVAL2)) do not allow an adversary to access the decryption oracle and the evaluation oracle (the oracle for homomorphic operation) simultaneously. We note that although Barbosa and Farshim defined verifiability (VRF-CCA2), where no homomorphic operation can be performed without issuing a corresponding token, KH-CCA security for KH-PKE defined in this paper guarantees a similar level of security, since if there exists an adversary that can perform the homomorphic operation without using sk_h, then the adversary can break the KH-CCA security.

2 Preliminaries

In this section, we review the basic notations and definitions related to HPSs (mostly following [11] but slightly customized for our convenience).

Throughout this paper, PPT denotes *probabilistic polynomial time*. If n is a natural number, then $[n] = \{1, \ldots, n\}$. If D is a probabilistic distribution (over some set), then $[D]$ denotes its support, i.e. $[D] = \{x | \Pr_{x' \leftarrow D}[x' = x] > 0\}$. Let $\mathbf{X} = \{X_\ell\}_{\ell \geq 0}$ and $\mathbf{Y} = \{Y_\ell\}_{\ell \geq 0}$ be sequences of random variables X_ℓ and Y_ℓ, respectively, defined over a same finite set. As usual, we say that $\mathbf{X}$ and $\mathbf{Y}$ are *statistically (resp. computationally) indistinguishable* if $|\Pr[\mathcal{A}(X_\ell) = 1] - \Pr[\mathcal{A}(Y_\ell) = 1]|$ is negligible in ℓ for any computationally unbounded (resp. PPT) algorithm $\mathcal{A}$. Furthermore, we say that $\mathbf{X}$ and $\mathbf{Y}$ are ϵ-close if the statistical distance of X_ℓ and Y_ℓ is at most $\epsilon = \epsilon(\ell)$.

Projective Hash Families: Let X, Π, W, K, and S be finite, non-empty sets, and L be a proper subset of X (i.e., $L \subset X$ and $L \neq X$). Furthermore, let $H = \{H_k : X \to \Pi\}_{k \in K}$ be a collection of hash functions indexed by $k \in K$, and $\alpha : K \to S$ be a function. We say that $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ is a *projective hash family* for (X, L) if for all $k \in K$, the action of H_k on the subset L is uniquely determined by $\alpha(k) \in S$.

Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family, and let $\epsilon \geq 0$. We recall the following properties of a projective hash family: We say that $\mathbf{H}$ is ϵ-*universal*$_1$ if for all $s \in S$, $x \in X \setminus L$, and $\pi \in \Pi$, it holds that $\Pr_{k \xleftarrow{\$} K}[H_k(x) = \pi \wedge \alpha(k) = s] \leq \epsilon \cdot \Pr_{k \xleftarrow{\$} K}[\alpha(k) = s]$. We say that $\mathbf{H}$ is ϵ-*universal*$_2$ if for all $s \in S$, $x, x^* \in X \setminus L$ with $x^* \neq x$, and $\pi, \pi^* \in \Pi$, it holds that $\Pr_{k \xleftarrow{\$} K}[H_k(x) = \pi \wedge H_k(x^*) = \pi^* \wedge \alpha(k) = s] \leq \epsilon \cdot \Pr_{k \xleftarrow{\$} K}[H_k(x^*) = \pi^* \wedge \alpha(k) = s]$. We say that $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ is ϵ-*smooth* if the following two distributions are ϵ-close: $\{k \xleftarrow{\$} K;\ x \xleftarrow{\$} X \setminus L\ :\ (\alpha(k), x, H_k(x))\ \}$ and $\{k \xleftarrow{\$} K;\ x \xleftarrow{\$} X \setminus L;\ \pi \xleftarrow{\$} \Pi\ :\ (\alpha(k), x, \pi)\ \}$.

If a projective hash family is ϵ-universal$_1$ (resp. -universal$_2$, -smooth) for a negligible ϵ, then we simply call the projective hash family universal$_1$ (resp. universal$_2$, smooth).

Subset Membership Problems: A subset membership problem $\mathbf{M}$ specifies a collection of probabilistic distribution $\{I_\ell\}_{\ell \geq 0}$ (indexed by a security parameter ℓ) over instance descriptions. An instance description $\Lambda[X, L, W, R] \in [I_\ell]$ specifies non-empty sets X, W, and L, a binary relation R defined over $X \times W$, where X, W, and L are non-empty sets such that $L \subset X$, and an $x \in X$ is in the subset L if and only if there exists a "witness" $\omega \in W$ such that $(x, w) \in R$. (If X, L, W, and R are clear from the context, we will just write Λ to indicate an instance description.)

We require that a subset membership problem $\mathbf{M}$ provides the following algorithms: (1) the instance sampling algorithm takes as input 1^ℓ, and returns $\Lambda[X, L, W, R] \in [I_\ell]$ chosen according to I_ℓ, and (2) the subset sampling algorithm takes as input 1^ℓ and an instance $\Lambda[X, L, W, R] \in [I_\ell]$, and returns $x \in L$ and a witness $\omega \in W$ for x. We say that a subset membership problem $\mathbf{M} = \{I_\ell\}_{\ell \geq 0}$ is hard if the following two distributions are computationally indistinguishable: $\{\Lambda \leftarrow I_\ell; x \xleftarrow{\$} L : (\Lambda, x)\}$ and $\{\Lambda \leftarrow I_\ell; x \xleftarrow{\$} X \setminus L : (\Lambda, x)\}$.

Hash Proof System (HPS): Informally, a HPS is a special kind of (designated-verifier) non-interactive zero-knowledge proof system for a subset membership problem $\mathbf{M} = \{I_\ell\}_{\ell > 0}$. A HPS has, as its internal structure, a family of hash functions with the special projective property, and this projective hash family is associated with each instance of the subset membership problems. Although HPS does not treat for all NP languages, HPS leads to an efficient CCA2-secure PKE construction.

As in [11], we will occasionally introduce an arbitrary finite set E to extend the sets X and L in an instance $\Lambda[X, L, W, R] \in [I_\ell]$ of $\mathbf{M}$ into $X \times E$ and $L \times E$. If E is not required (e.g., for a smooth HPS in our construction), then we omit E

from the following algorithms. A HPS $\mathbf{P} = (\mathsf{HPS.param}, \mathsf{HPS.priv}, \mathsf{HPS.pub})$, for $\mathbf{M}$ associates each instance $\Lambda = \Lambda[X, L, W, R]$ of $\mathbf{M}$ with a projective hash family $\mathbf{H} = (H, K, X \times E, L \times E, \Pi, S, \alpha)$, provides the following three algorithms: (1) The index sampling algorithm $\mathsf{HPS.param}$ takes an instance Λ as input, and returns $k \in K$ and $s \in S$ such that $\alpha(k) = s$. (2) The private evaluation algorithm $\mathsf{HPS.priv}$ takes $\Lambda \in [I_\ell]$, $k \in K$ and $(x, e) \in X \times E$ as input, and returns $\pi = H_k(x, e) \in \Pi$. (3) The public evaluation algorithm $\mathsf{HPS.pub}$ takes $\Lambda \in [I_\ell]$, $s \in S$, $x \in L$, $e \in E$, and a witness ω for x as input, and returns $\pi = H_k(x, e) \in \Pi$. We say that $\mathbf{P}$ is ϵ-universal$_1$ (resp. ϵ-universal$_2$, ϵ-smooth) if for all $\ell > 0$ and for all $\Lambda[X, L, W, R] \in [I_\ell]$, $\mathbf{H}$ is an ϵ-universal$_1$ (resp. ϵ-universal$_2$, ϵ-smooth) projective hash family.

Note that the homomorphic property of the underlying smooth projective hash family is required in our construction, where for all $k \in K$, and $x_1, x_2 \in X$, we have $H_k(x_1) + H_k(x_2) = H_k(x_1 + x_2) \in \Pi$ holds. Then, we call this smooth projective hash family homomorphic smooth projective hash family, and also call a smooth HPS homomorphic smooth HPS if the underlying smooth projective hash family has the homomorphic property.

Diverse Group System and Derived Projective Hash Family: Here, we recall the definition of diverse group systems introduced in [11], which were used to construct projective hash families. Let X, L, and Π be abelian groups, where L is a proper subgroup of X, and $\mathsf{Hom}(X, \Pi)$ be the group of all homomorphisms $\phi : X \to \Pi$. Let $\mathcal{H}$ be a subgroup of $\mathsf{Hom}(X, \Pi)$. Then $\mathbf{G} = (\mathcal{H}, X, L, \Pi)$ is called a *group system*. In addition, we say that $\mathbf{G}$ is *diverse* if for all $x \in X \setminus L$, there exists $\phi \in \mathcal{H}$ such that $\phi(L) = \langle 0 \rangle$, but $\phi(x) \neq 0$.

We recall the projective hash family $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ derived from a diverse group system $\mathbf{G}$ ([11, Definition 2]): Let $g_1, \ldots, g_d \in L$ be a set of generators of L (i.e., for all $x \in L$, there exist $\omega_1, \ldots, \omega_d \in \mathbb{Z}$ such that $x = \sum_{i=1}^{d} \omega_i g_i$). Set $S = \Pi^d$, and define $\alpha : K \to S$ by $\alpha(k) = (\phi(g_1), \ldots, \phi(g_d))$, where $\phi = H_k$. Note that $\mathbf{H}$ is a projective hash family because $H_k(x)$ for $x \in L$ is determined by $\alpha(k)$ such that $H_k(x) = \phi(\sum_{i=1}^{d} \omega_i g_i) = \sum_{i=1}^{d} \omega_i \phi(g_i)$. The following was shown by Cramer and Shoup [11, Theorem 2].

Lemma 1. *The projective hash family* $\mathbf{H}$ *derived from a diverse group system* $\mathbf{G}$ *as above is* $1/\widetilde{p}$*-universal*$_1$*, where* $\widetilde{p}$ *is the smallest prime dividing* $|X/L|$.

3 Definition of KH-PKE

In this section, we give the formal definitions of the syntax and the security requirements of KH-PKE.

3.1 Syntax of KH-PKE

Definition 1 (Syntax of KH-PKE for homomorphic operation $\odot$). *Let* $\mathcal{M}$ *be a message space. We require that for all* $M_1, M_2 \in \mathcal{M}$*, it holds that*

$M_1 \odot M_2 \in \mathcal{M}$. *A KH-PKE scheme* $\mathcal{KH}\text{-}\mathcal{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *for homomorphic operation* $\odot$ *consists of the following four algorithms:*

KeyGen: *This algorithm takes a security parameter* 1^ℓ ($\ell \in \mathbb{N}$) *as input, and returns a public key* pk*, a decryption key* sk_d*, and a homomorphic operation key* sk_h*.*

Enc: *This algorithm takes* pk*, and a message* $M \in \mathcal{M}$ *as input, and returns a ciphertext* C*.*

Dec: *This algorithm takes* sk_d *and* C *as input, and returns* M *or* $\perp$*.*

Eval: *This algorithm takes* sk_h*, two ciphertexts* C_1 *and* C_2 *as input, and outputs a ciphertext* C *or* $\perp$*.*

Note that the above definition for the evaluation algorithm Eval does not say anything about the homomorphic property, and its functionality is defined as a correctness requirement below. Let pk be a public key generated by the KeyGen algorithm, and $\mathcal{C}_{pk,M}$ be the set of all ciphertexts of $M \in \mathcal{M}$ under the public key pk, i.e., $\mathcal{C}_{pk,M} = \{C | \exists r \in \{0,1\}^* \text{ s.t. } C = \mathsf{Enc}(pk, M; r)\}$.

Definition 2 (Correctness). *A KH-PKE scheme for homomorphic operation* $\odot$ *is said to be* correct *if for all* $(pk, sk_d, sk_h) \leftarrow \mathsf{KeyGen}(1^\ell)$*, the following two conditions are satisfied: (1) For all* $M \in \mathcal{M}$*, and all* $C \in \mathcal{C}_{pk,M}$*, it holds that* $\mathsf{Dec}(sk_d, C) = M$*. (2) For all* $M_1, M_2 \in \mathcal{M}$*, all* $C_1 \in \mathcal{C}_{pk,M_1}$*, and all* $C_2 \in \mathcal{C}_{pk,M_2}$*, it holds that* $\mathsf{Eval}(sk_h, C_1, C_2) \in \mathcal{C}_{pk,M_1 \odot M_2}$*.*

If an operation $\odot$ is commutative, then the Eval algorithm is also called *commutative*, and we require that the distribution of $\mathsf{Eval}(sk_h, C_1, C_2)$ and that of $\mathsf{Eval}(sk_h, C_2, C_1)$ are identical. We instantiate DDH/DLIN/DBDH-based KH-PKEs with multiplicative homomorphic operations ($\odot := \times$), a DCR-based KH-PKE with additive homomorphic operations ($\odot := +$), and a DQR-based KH-PKE with XOR homomorphic operations ($\odot := \oplus$). Thus, our concrete instantiations are all commutative schemes.

Next, we define the security notion for KH-PKE, which we call *indistinguishability of message under adaptive chosen ciphertext attacks* (KH-CCA).

Definition 3 (KH-CCA). *A KH-PKE scheme is said to be KH-CCA secure if for any PPT adversary* $\mathcal{A}$*, the advantage*

$$Adv^{KH\text{-}CCA}_{\mathsf{KH\text{-}PKE},\mathcal{A}}(\ell) = \Big| \Pr[(pk, sk_d, sk_h) \leftarrow \mathsf{KeyGen}(1^\ell); \\ (M_0^*, M_1^*, State) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{find}, pk);\ \beta \xleftarrow{\$} \{0,1\}; \\ C^* \leftarrow \mathsf{Enc}(pk, M_\beta^*);\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{guess}, State, C^*);\ \beta = \beta'] - \frac{1}{2} \Big|$$

is negligible in ℓ*, where* $\mathcal{O}$ *consists of the three oracles* $\mathsf{Eval}(sk_h, \cdot, \cdot)$*,* RevHK*, and* $\mathsf{Dec}(sk_d, \cdot)$ *defined as follows. Let* $\mathcal{D}$ *be a list which is set as* $\mathcal{D} = \{C^*\}$ *right after the challenge stage (*$\mathcal{D}$ *is set as* $\emptyset$ *in the* find *stage).*

- *The evaluation oracle* $\mathsf{Eval}(sk_h, \cdot, \cdot)$*: If* RevHK *has already been queried before, then this oracle is not available. Otherwise, this oracle responds to a query* (C_1, C_2) *with the result of* $C \leftarrow \mathsf{Eval}(sk_h, C_1, C_2)$*. In addition, if* $C \neq \perp$ *and either* $C_1 \in \mathcal{D}$ *or* $C_2 \in \mathcal{D}$*, then the oracle updates the list by* $\mathcal{D} \leftarrow \mathcal{D} \cup \{C\}$*.*
- *The homomorphic key reveal oracle* RevHK*: Upon a request, this oracle responds with* sk_h*. (This oracle is available only once.)*
- *The decryption oracle* $\mathsf{Dec}(sk_d, \cdot)$*: This oracle is not available if* $\mathcal{A}$ *has queried to* RevHK *and* $\mathcal{A}$ *has obtained the challenge ciphertext* C^**. Otherwise, this oracle responds to a query* C *with the result of* $\mathsf{Dec}(sk_d, \cdot)$ *if* $C \notin \mathcal{D}$ *or returns* $\perp$ *otherwise.*

Here, let us remark on the definition of KH-CCA security. Throughout this paper, an adversary who has sk_h is called an *insider*, whereas an adversary who does not have sk_h is called an *outsider*.

In case $\mathcal{A}$ does not query the RevHK oracle (i.e., $\mathcal{A}$ is an outsider), $\mathcal{A}$ is allowed to adaptively issue decryption queries and evaluation queries of any ciphertexts. In particular, in order to capture the malleability in the presence of the homomorphic operation, the Eval oracle allows the challenge ciphertext C^* as input. To avoid an unachievable security definition, the Dec oracle immediately answers $\perp$ for "unallowable ciphertexts" that are the results of a homomorphic operation for C^* and any ciphertext of an adversary's choice. Such unallowable ciphertexts are maintained by the list $\mathcal{D}$.

The situation that the Dec oracle does not answer for ciphertexts that are derived from the challenge ciphertext C^* might seem somewhat analogous to the definition of RCCA security [6]. However, there is a critical difference between KH-CCA and RCCA: In the RCCA security game, the Dec oracle does not answer if a ciphertext C satisfies $\mathsf{Dec}(sk_d, C) \in \{M_0^*, M_1^*\}$. That is, the functionality of the Dec oracle is restricted regardless of the adversary's strategy. On the other hand, in the KH-CCA security game, in case an adversary selects the strategy that it does not submit C^* to the Eval oracle, the restriction on the Dec oracle is exactly the same as the CCA2 security for ordinary PKE scheme, and it is one of the adversary's possible strategies whether it submits C^* to the Eval oracle, and thus the adversary has more flexibility than in the RCCA game.

If an outsider $\mathcal{A}$ becomes an insider *after* $\mathcal{A}$ obtains the challenge ciphertext C^*, then $\mathcal{A}$ is not allowed to issue a decryption query *after* obtaining sk_h via the RevHK oracle. In other words, $\mathcal{A}$ is allowed to issue a decryption query until right before obtaining sk_h, even if C^* is given to $\mathcal{A}$. This restriction is again to avoid a triviality. (If $\mathcal{A}$ obtains sk_h, $\mathcal{A}$ can freely perform homomorphic operations over the challenge ciphertexts, and we cannot meaningfully define the "unallowable set" of ciphertexts.)

Note that we can show that any KH-CCA secure PKE scheme satisfies CCA1 (thus CPA also) security against an adversary who is given (pk, sk_h) in the setup phase. Showing this implication is possible mainly due to the RevHK oracle that returns sk_h to an adversary, and the Dec oracle in the KH-CCA game.

4 Generic Construction via Homomorphic Transitional Universal HPS

In this section, we give a generic construction of KH-PKE from an enhanced variant of universal HPS, which we call *homomorphic transitional universal HPS*. A homomorphic transitional universal HPS has, as its internal structure, a family of hash functions which we call *transitional universal projective hash family*.

4.1 Homomorphic Transitional Universal Projective Hash Families

Informally, a projective hash family $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ is said to be a transitional universal projective hash family if an index $k \in K$ for specifying a hash function from the family can be divided into two components as $(k', \widehat{k})$, and even if $\widehat{k}$ is exposed, it still yields the universal$_1$ property.

Definition 4 (Homomorphic Transitional (ϵ, ϵ')-Universal Projective Hash Families). *Let $\mathbf{H} = (H, K, X \times E, L \times E, \Pi, S, \alpha)$ be an ϵ-universal$_2$ hash family. We say that $\mathbf{H}$ is (ϵ, ϵ')-transitional if (1) The function index space K can be divided into two subspaces K_1 and K_2 such that $K = K_1 \times K_2$ (say $\overrightarrow{k} := (k', \widehat{k}) \in K_1 \times K_2$), and (2) Considering the probability space defined by choosing $k' \in K_1$ at random. Then for all $s \in S$, $x \in X \setminus L$, $\widehat{k} \in K_2$ and $\pi \in \Pi$, it holds that $\Pr_{k' \xleftarrow{\$} K_1}[H_{k',\widehat{k}}(x, e) = \pi \wedge \alpha(k', \widehat{k}) = s] \leq \epsilon' \cdot \Pr_{k' \xleftarrow{\$} K_1}[\alpha(k', \widehat{k}) = s]$. Especially, if ϵ and ϵ' are negligible, then $\mathbf{H}$ is called a transitional universal projective hash family. Moreover, if for all $(k', \widehat{k}) \in K_1 \times K_2$ and for all $(x_1, e_1), (x_2, e_2) \in X \times E$, $H_{k',\widehat{k}}(x_1 + x_2, e_1 + e_2)$ can be efficiently computed given $\widehat{k}$, $(x_1, e_1, H_{k',\widehat{k}}(x_1, e_1))$ and $(x_2, e_2, H_{k',\widehat{k}}(x_2, e_2))$, then $\mathbf{H}$ is called a homomorphic transitional universal projective hash family.*

Next, we show that the projective hash family [10, §7.43 Theorem 3] based on a diverse group system, satisfies the homomorphic transitional universal property as it is.

The Cramer-Shoup (CS) Projective Hash Family [10] : Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be a universal$_1$ projective hash family derived from a diverse group system $\mathbf{G} = (\mathcal{H}, X, L, \Pi)$ (see the last paragraph of Section 2), and E be an abelian group. Then the CS projective hash family $\widehat{\mathbf{H}} = (\widehat{H}, \widehat{K} = K^{n+1}, X \times E, L \times E, \widehat{\Pi}, \widehat{S} = S^{n+1}, \widehat{\alpha})$ is constructed as follows: Let $\Gamma : X \times E \to \{0, \ldots, \widetilde{p} - 1\}^n$ be an injective function, where $\widetilde{p}$ is the smallest prime dividing $|X/L|$, and n is sufficiently large enough for Γ to be injective. For $\overrightarrow{k} = (k', k_1, \ldots, k_n) \in K^{n+1}$, $x \in X$, and $e \in E$, $\widehat{H}$ is defined as: $\widehat{H}_{k',\widehat{k}}(x, e) := H_{k'}(x) + \sum_{i=1}^{n} \gamma_i H_{k_i}(x)$, and $\widehat{\alpha}(k', \widehat{k}) = (\alpha(k'), \alpha(k_1), \ldots, \alpha(k_n))$, where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e)$. Cramer and Shoup showed that the CS projective hash family $\widehat{\mathbf{H}}$ is $(1/\widetilde{p})$-universal$_2$. Note that since $H_k = \phi \in \mathsf{Hom}(X, \Pi)$, the basic projective hash family H derived from the diverse group system satisfies the homomorphic

property, namely for all $k \in K$, and $x_1, x_2 \in X$, we have $H_k(x_1) + H_k(x_2) = H_k(x_1 + x_2) \in \Pi$. Next, we show that it is in fact a homomorphic transitional universal projective hash family.

Lemma 2. *If an index $\overrightarrow{k} \in K^{n+1}$ is divided into $k' \in K$ and $\widehat{k} = (k_1, \ldots, k_n) \in K^n$, then the CS projective hash family $\widehat{\mathbf{H}}$ is a homomorphic transitional $(1/\widetilde{p}, 1/\widetilde{p})$-universal projective hash family.*

Proof: For $\overrightarrow{k} \in K^{n+1}$, fix $(k_1, \ldots, k_n) \in K^n$, and consider the probability space is defined by choosing $k' \in K$ at random. Then, $\widehat{\mathbf{H}}$ still provides the $(1/\widetilde{p})$-universal$_1$ property, because the projective hash family $\mathbf{H}$ is a $(1/\widetilde{p})$-universal$_1$ and the output of $\widehat{\mathbf{H}}$ is "masked" by the output of $\mathbf{H}$. Furthermore, for all $(x_1, e_1), (x_2, e_2) \in X \times E$, $H_{k',\widehat{k}}(x_1 + x_2, e_1 + e_2)$ can be efficiently computed given $\widehat{k} = (k_1, \ldots, k_n)$, $(x_1, e_1, H_{k',\widehat{k}}(x_1, e_1))$ and $(x_2, e_2, H_{k',\widehat{k}}(x_2, e_2))$ such that (1) compute $\sum_{i=1}^n \gamma_i^{(1)} H_{k_i}(x_1)$ and $\sum_{i=1}^n \gamma_i^{(2)} H_{k_i}(x_2)$, where $(\gamma_1^{(b)}, \ldots, \gamma_n^{(b)}) = \Gamma(x_b, e_b)$ for $b = 1, 2$, and (2) compute $\widehat{H}_{k',\widehat{k}}(x_1 + x_2, e_1 + e_2) \leftarrow \big(\widehat{H}_{k',\widehat{k}}(x_1, e_1) - \sum_{i=1}^n \gamma_i^{(1)} H_{k_i}(x_1)\big) + \big(\widehat{H}_{k',\widehat{k}}(x_2, e_2) - \sum_{i=1}^n \gamma_i^{(2)} H_{k_i}(x_2)\big) + \sum_{i=1}^n \gamma_i H_{k_i}(x_1 + x_2)$, where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x_1 + x_2, e_1 + e_2)$. □

Finally we define the notion of homomorphic transitional universal HPS.

Definition 5 (Homomorphic Transitional Universal HPS). *Let $\mathbf{M} = \{I_\ell\}_{\ell \geq 0}$ be a subset membership problem. We say that a HPS $\mathbf{P}$ for $\mathbf{M}$ is homomorphic transitional (ϵ, ϵ')-universal if for all $\ell > 0$ and for all $\Lambda = \Lambda[X, L, W, R] \in [I_\ell]$, the projective hash family $\mathbf{H}$ that $\mathbf{P}$ associates with Λ is homomorphic transitional (ϵ, ϵ')-universal.*

4.2 Generic Construction of KH-PKE

Here, we give the proposed construction of a KH-PKE scheme based on a homomorphic transitional universal HPS given in the previous subsection, a homomorphic smooth projective HPS, and a universal$_2$ projective HPS. We note that all of the projective hash families used in our construction can be constructed from a diverse group system [11]. Therefore, our proposed construction is fairly generic.

We set $E = \Pi$ (Π is an abelian group, for which we use additive notation) and $\Gamma : X \times \Pi \to \Pi^n$ is an injective function, where n is a natural number which is sufficiently large so that Γ is injective. Let $\mathbf{M} = \{I_\ell\}_{\ell \geq 0}$ be a subset membership problem which specifies an instance description $\Lambda = \Lambda[X, L, W, R] \in [I_\ell]$. We will use the following three kinds of projective hash families $\mathbf{H}$, $\widehat{\mathbf{H}}$ and $\widetilde{\mathbf{H}}$ and corresponding HPS (for $\mathbf{M}$). Using these building blocks, we construct a KH-PKE scheme as in Figure 1.

- $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ is a homomorphic smooth and projective hash family. Let $\mathbf{P} = (\mathsf{HPS.param}, \mathsf{HPS.priv}, \mathsf{HPS.pub})$ be a homomorphic smooth projective HPS for $\mathbf{M}$ which associates the instance Λ with $\mathbf{H}$.

$\mathsf{KeyGen}(1^\ell)$:
Pick $\Lambda = \Lambda[X, L, W, R] \leftarrow [I_\ell]$.
$(k, s) \leftarrow \mathsf{HPS.param}(1^\ell, \Lambda)$
$(\overrightarrow{\widehat{k}}, \widehat{s}) \leftarrow \widehat{\mathsf{HPS}}.\mathsf{param}(1^\ell, \Lambda)$
Parse $\overrightarrow{\widehat{k}} \in \widehat{K} = K \times K^n$ as $(k', \widehat{k})$
s.t. $k' \in K$ and
$\widehat{k} := (k_1, \ldots, k_n) \in K^n$
$(\widetilde{k}, \widetilde{s}) \leftarrow \widetilde{\mathsf{HPS}}.\mathsf{param}(1^\ell, \Lambda)$
$pk \leftarrow (s, \widehat{s}, \widetilde{s})$
$sk_d \leftarrow (k, (k', \widehat{k}), \widetilde{k})$; $sk_h \leftarrow (\widehat{k}, \widetilde{k})$
Return (pk, sk_d, sk_h)

$\mathsf{Dec}(sk_d, C)$:
Parse sk_d as $(k, (k', \widehat{k}), \widetilde{k})$
Parse C as $(x, e, \widehat{\pi}, \widetilde{\pi})$
$\widehat{\pi}' \leftarrow \widehat{\mathsf{HPS}}.\mathsf{priv}(1^\ell, \Lambda, (k', \widehat{k}), (x, e))$
$\widetilde{\pi}' \leftarrow \widetilde{\mathsf{HPS}}.\mathsf{priv}(1^\ell, \Lambda, \widetilde{k}, (x, e))$
If $\widehat{\pi} \neq \widehat{\pi}'$ or $\widetilde{\pi} \neq \widetilde{\pi}'$ then return $\perp$
$\pi \leftarrow \mathsf{HPS.priv}(1^\ell, \Lambda, k, x)$
Return $M \leftarrow e - \pi$

$\mathsf{Enc}(pk, M)$:
Choose $x \xleftarrow{\$} L$ and its witness $\omega \in W$
$\pi \leftarrow \mathsf{HPS.pub}(1^\ell, \Lambda, s, x, \omega)$; $e \leftarrow M + \pi$
$\widehat{\pi} \leftarrow \widehat{\mathsf{HPS}}.\mathsf{pub}(1^\ell, \Lambda, \widehat{s}, (x, e), \omega)$
$\widetilde{\pi} \leftarrow \widetilde{\mathsf{HPS}}.\mathsf{pub}(1^\ell, \Lambda, \widetilde{s}, (x, e), \omega)$
Return $C \leftarrow (x, e, \widehat{\pi}, \widetilde{\pi})$.

$\mathsf{Eval}(sk_h, C_1, C_2)$:
Parse sk_h as $(\widehat{k}, \widetilde{k})$ where $\widehat{k} = (k_1, \ldots, k_n)$
Parse C_b as $(x_b, e_b, \widehat{\pi}_b, \widetilde{\pi}_b)$ for $b = 1, 2$
$\widetilde{\pi}'_b \leftarrow \widetilde{\mathsf{HPS}}.\mathsf{priv}(1^\ell, \Lambda, \widetilde{k}, (x_b, e_b))$ for $b = 1, 2$
If $\widetilde{\pi}_1 \neq \widetilde{\pi}'_1$ or $\widetilde{\pi}_2 \neq \widetilde{\pi}'_2$ then return $\perp$
For $b = 1, 2$ Do:
$(\gamma_1^{(b)}, \ldots, \gamma_n^{(b)}) \leftarrow \Gamma(x_b, e_b)$
$H_{k_i}(x_b) \leftarrow \mathsf{HPS.priv}(1^\ell, \Lambda, k_i, x_b)$
for all $i \in [n]$
$\widehat{\pi}'_b \leftarrow \widehat{\pi}_b - \sum_{i \in [n]} \gamma_i^{(b)} H_{k_i}(x_b)$
End For
$x \leftarrow x_1 + x_2$; $e \leftarrow e_1 + e_2$
$(\gamma_1, \ldots, \gamma_n) \leftarrow \Gamma(x, e)$
$H_{k_i}(x) \leftarrow \mathsf{HPS.priv}(1^\ell, \Lambda, k_i, x)$
for all $i \in [n]$
$\widehat{\pi} \leftarrow \widehat{\pi}'_1 + \widehat{\pi}'_2 + \sum_{i \in [n]} \gamma_i H_{k_i}(x)$
$\widetilde{\pi} \leftarrow \widetilde{\mathsf{HPS}}.\mathsf{priv}(1^\ell, \Lambda, \widetilde{k}, (x, e))$
Return $C \leftarrow (x, e, \widehat{\pi}, \widetilde{\pi})$

Fig. 1. The proposed KH-PKE construction from HPS

- $\widehat{\mathbf{H}} = (\widehat{H}, \widehat{K} = K \times K^n, X \times \Pi, L \times \Pi, \widehat{\Pi}, \widehat{S} = S^{n+1}, \widehat{\alpha})$ is the CS (homomorphic transitional universal) projective hash family that we showed in the previous subsection (with the index space $\widehat{K}$ is divided into $K_1 = K$ and $K_2 = K^n$). Let $\widehat{\mathbf{P}} = (\widehat{\mathsf{HPS}}.\mathsf{param}, \widehat{\mathsf{HPS}}.\mathsf{priv}, \widehat{\mathsf{HPS}}.\mathsf{pub})$ be a homomorphic transitional universal HPS for $\mathbf{M}$ which associates Λ with $\widehat{\mathbf{H}}$.
- $\widetilde{\mathbf{H}} = (\widetilde{H}, \widetilde{K}, X \times \Pi, L \times \Pi, \widetilde{\Pi}, \widetilde{S}, \widetilde{\alpha})$ is a universal$_2$ projective hash family. Let $\widetilde{\mathbf{P}} = (\widetilde{\mathsf{HPS}}.\mathsf{param}, \widetilde{\mathsf{HPS}}.\mathsf{priv}, \widetilde{\mathsf{HPS}}.\mathsf{pub})$ be a universal$_2$ HPS for $\mathbf{M}$ which associates Λ with $\widetilde{\mathbf{H}}$.

One might think that in the contraction, $\widetilde{\mathbf{H}}$ is redundant, and thus is not necessary. However, this is not true. Namely, if $\widetilde{\mathbf{H}}$ is removed, then the adversary can extract meaningful information from the Eval oracle by submitting an invalid ciphertexts, and therefore, the resulting scheme becomes insecure. In other words, with the help of $\widetilde{\mathbf{H}}$, the Eval oracle can distinguish invalid ciphertexts from valid ones, and consequently, the above attack is prevented.

To see the correctness for the Eval algorithm, suppose that Eval receives correctly generated ciphertexts $C_1 = (x_1, e_1, \widehat{\pi}_1, \widetilde{\pi}_1)$ and $C_2 = (x_2, e_2, \widehat{\pi}_2, \widetilde{\pi}_2)$ of plaintexts M_1 and M_2, respectively. Let $M = M_1 + M_2$. Then, by recalling

the homomorphic and transitional properties, the following holds: $\widehat{\pi}'_b = \widehat{\pi}_b - \sum_{i=1}^{n} \gamma_i^{(b)} H_{k_i}(x_b) = H_{k'}(x_b)$ for $b = 1, 2$, $e_1 + e_2 = (M_1 + M_2) + (H_k(x_1) + H_k(x_2)) = (M_1 + M_2) + H_k(x_1 + x_2) = (M_1 + M_2) + H_k(x)$, $\widehat{\pi} = \widehat{\pi}'_1 + \widehat{\pi}'_2 + \sum_{i=1}^{n} \gamma_i H_{k_i}(x) = H_{k'}(x_1) + H_{k'}(x_2) + \sum_{i=1}^{n} \gamma_i H_{k_i}(x) = H_{k'}(x_1 + x_2) + \sum_{i=1}^{n} \gamma_i H_{k_i}(x) = H_{k'}(x) + \sum_{i=1}^{n} \gamma_i H_{k_i}(x) = \widehat{H}_{k',\widehat{k}}(x, e)$, which means that $C = (x, e, \widehat{\pi}, \widetilde{\pi})$ is a valid ciphertext of $M := M_1 + M_2$.

Since all of the projective hash families used in our construction can be constructed from a diverse group system, from the result of [19] (where CPA-secure homomorphic PKE (with cyclic ciphertext space) implies diverse group systems), the following corollary is given.

Corollary 1. *KH-PKE is implied by CPA-secure homomorphic PKE with cyclic ciphertext space.*

The proof of the following theorem is given in the Appendix.

Theorem 1. *Our construction is KH-CCA-secure if* $\mathbf{M}$ *is a hard subset membership problem,* $\mathbf{P}$ *is a homomorphic smooth projective HPS for* $\mathbf{M}$*,* $\widehat{\mathbf{P}}$ *is a homomorphic transitional universal HPS for* $\mathbf{M}$*, and* $\widetilde{\mathbf{P}}$ *is a universal*$_2$ *HPS for* $\mathbf{M}$*.*

5 Practical KH-PKE Construction from DDH

In this section, we present an efficient DDH-based KH-PKE construction. This scheme is not a mere combination of the generic construction of KH-PKE in Section 4 and the transitional HPS from DDH (which will appear in the full version), but introduces additional techniques for enhancing efficiency. Remarkably, efficiency of our scheme is only slightly lower than the Cramer-Shoup encryption in spite of its complicated functionality. In particular, ciphertext length of our scheme is only ℓ-bit larger than that of the Cramer-Shoup scheme, where ℓ is the security parameter. For example, for 128-bit security, ciphertext overhead of our scheme is 896-bit while that of the Cramer-Shoup scheme is 768-bit (assuming that these schemes are implemented over elliptic curves).

5.1 Techniques for Improving Efficiency

Before going into the concrete construction of our DDH-based KH-PKE scheme, we briefly explain two additional techniques for enhancing efficiency which are not mentioned in the previous sections. Both these techniques employ target collision resistant (TCR) hash functions [10], and can also be applicable to other various (standard) PKE schemes.

The first technique is just the same as the popular method for transforming hash-free variant of the Cramer-Shoup scheme into the TCR-based one (i.e., the standard Cramer-Shoup scheme). Due to it, the size of the public key is significantly reduced.

$\mathsf{KeyGen}(1^\ell)$:
$g_0, g_1 \xleftarrow{\$} \mathbb{G}$
$k_0, k_1, k'_0, k'_1, \widehat{k}_{1,0}, \widehat{k}_{1,1}, \widetilde{k}_0, \widetilde{k}_1, \widetilde{k}_{1,0}, \widetilde{k}_{1,1} \xleftarrow{\$} \mathbb{Z}_p$;
$s \leftarrow g_0^{k_0} g_1^{k_1}$; $s' \leftarrow g_0^{k'_0} g_1^{k'_1}$
$\widehat{s} \leftarrow g_0^{\widehat{k}_{1,0}} g_1^{\widehat{k}_{1,1}}$; $\widetilde{s} \leftarrow g_0^{\widetilde{k}_0} g_1^{\widetilde{k}_1}$
$\widetilde{s}_1 \leftarrow g_0^{\widetilde{k}_{1,0}} g_1^{\widetilde{k}_{1,1}}$
$pk \leftarrow (g_0, g_1, s, s', \widehat{s}, \widetilde{s}, \widetilde{s}_1)$
$sk_d \leftarrow ((k_0, k_1), (k'_0, k'_1, \widehat{k}_{1,0}, \widehat{k}_{1,1}), (\widetilde{k}_0, \widetilde{k}_1, \widetilde{k}_{1,0}, \widetilde{k}_{1,1}))$
$sk_h \leftarrow ((\widehat{k}_{1,0}, \widehat{k}_{1,1}), (\widetilde{k}_0, \widetilde{k}_1, \widetilde{k}_{1,0}, \widetilde{k}_{1,1}))$
Return (pk, sk_d, sk_h)

$\mathsf{Dec}(sk_d, C)$:
Parse C as $(x_0, x_1, e, \widehat{\pi}, \tau)$
$\gamma \leftarrow \mathsf{TCR}_1(x_0, x_1, e)$
$\widehat{\pi}' \leftarrow x_0^{k'_0 + \gamma \widehat{k}_{1,0}} x_1^{k'_1 + \gamma \widehat{k}_{1,1}}$
$\widetilde{\pi}' \leftarrow x_0^{\widetilde{k}_0 + \gamma \widetilde{k}_{1,0}} x_1^{\widetilde{k}_1 + \gamma \widetilde{k}_{1,1}}$
If either $\widehat{\pi} \neq \widehat{\pi}'$ or $\tau \neq \mathsf{TCR}_2(\widetilde{\pi}')$ then return $\perp$
$\pi \leftarrow x_0^{k_0} x_1^{k_1}$
Return $M \leftarrow e/\pi$

$\mathsf{Enc}(pk, M)$:
$\omega \xleftarrow{\$} \mathbb{Z}_p$; $x_0 \leftarrow g_0^\omega$; $x_1 \leftarrow g_1^\omega$
$\pi \leftarrow s^\omega$; $e \leftarrow M \cdot \pi$
$\gamma \leftarrow \mathsf{TCR}_1(x_0, x_1, e)$
$\widehat{\pi} \leftarrow (s' \cdot \widehat{s}^{\,\gamma})^\omega$; $\widetilde{\pi} \leftarrow (\widetilde{s} \cdot \widetilde{s}_1^{\,\gamma})^\omega$
$\tau \leftarrow \mathsf{TCR}_2(\widetilde{\pi})$
Return $C \leftarrow (x_0, x_1, e, \widehat{\pi}, \tau)$

$\mathsf{Eval}(sk_h, C_1, C_2)$:
Parse C_b as $(x_{b,0}, x_{b,1}, e_b, \widehat{\pi}_b, \tau_b)$ for $b = 1, 2$
$\gamma_b \leftarrow \mathsf{TCR}_1(x_{b,0}, x_{b,1}, e_b)$ for $b = 1, 2$
$\widetilde{\pi}'_b \leftarrow x_{b,0}^{\widetilde{k}_0 + \gamma_b \widetilde{k}_{1,0}} x_{b,1}^{\widetilde{k}_1 + \gamma_b \widetilde{k}_{1,1}}$ for $b = 1, 2$
If $\tau_1 \neq \mathsf{TCR}_2(\widetilde{\pi}'_1)$ or $\tau_2 \neq \mathsf{TCR}_2(\widetilde{\pi}'_2)$ then return $\perp$
$\widehat{\pi}'_b \leftarrow \widehat{\pi}_b / (x_{b,0}^{\gamma_b \widehat{k}_{1,0}} x_{b,1}^{\gamma_b \widehat{k}_{1,1}})$ for $b = 1, 2$
$x_0 \leftarrow x_{1,0} x_{2,0}$; $x_1 \leftarrow x_{1,1} x_{2,1}$
$e \leftarrow e_1 e_2$; $\gamma \leftarrow \mathsf{TCR}_1(x_0, x_1, e)$
$\widehat{\pi} \leftarrow \widehat{\pi}'_1 \widehat{\pi}'_2 x_0^{\gamma \widehat{k}_{1,0}} x_1^{\gamma \widehat{k}_{1,1}}$
$\widetilde{\pi} \leftarrow x_0^{\widetilde{k}_0 + \gamma \widetilde{k}_{1,0}} x_1^{\widetilde{k}_1 + \gamma \widetilde{k}_{1,1}}$
$\tau \leftarrow \mathsf{TCR}_2(\widetilde{\pi})$
Return $C \leftarrow (x_0, x_1, e, \widehat{\pi}, \tau)$

Fig. 2. Our DDH-based KH-PKE Scheme

The second technique is to compress the redundant part of the ciphertext by using a TCR hash function. Interestingly, our security proof still works even if one of ciphertext components (specifically, a component for validity checking upon the homomorphic operation) is hashed to be a smaller value. It is a bit surprising that this technique can be also applied to the original Cramer-Shoup scheme, but to the best of our knowledge, it has never explicitly been stated in the literatures. When applying our technique to the Cramer-Shoup scheme, ciphertext length of the resulting scheme becomes the same as that of the Kurosawa-Desmedt (KD) scheme [23] which is the best known DDH-based PKE scheme. We should also note that this technique is not applicable to other similar schemes such as the Cash-Kiltz-Shoup [7], Hanaoka-Kurosawa [18], and Kiltz schemes [21]. This fact is primarily due to the structure of HPS-based constructions, and thus, it is difficult to apply the above technique to PKE schemes from other methodology, e.g. [5,18,20].

5.2 Practical KH-PKE from DDH

Here, we give a description of our KH-PKE instantiation (using our technique of reducing the ciphertext size). First, we define the DDH assumption as follows.

Table 1. Comparison among the Cramer-Shoup (CS) scheme, the Kurosawa-Desmedt (KD) scheme, the KD + CS-lite (using the double encryption) scheme, and our DDH-based KH-PKE scheme, where $|C| - |M|$ denotes ciphertext overhead, $|\mathbb{G}|$ denotes the size of the underlying group element $\mathbb{G}$, and exp denotes exponentiation. We count 1 multi-exp equals as 1.2 regular exp, and the size of MAC and the hashed value of TCR as $0.5|\mathbb{G}|$.

	$\|C\| - \|M\|$	Cost (Enc)	Cost (Dec)	KH property
CS [9]	$3\|\mathbb{G}\|$	4.2 exp	2.4 exp	No
KD [23]	$2.5\|\mathbb{G}\|$	3.2 exp	1.2 exp	No
KD+CS-lite Double Enc	$5.5\|\mathbb{G}\|$	7.2 exp	3.6 exp	No?
Our DDH-based KH-PKE	$3.5\|\mathbb{G}\|$	5.4 exp	3.6 exp	Yes

Definition 6 (The Decisional Diffie-Hellman (DDH) Assumption). *Let $\mathbb{G}$ be a group with prime order p. We say that the DDH assumption holds in $\mathbb{G}$ if the advantage $Adv^{DDH}_{\mathbb{G},\mathcal{A}}(1^\ell) := |\Pr[\mathcal{A}(g_0, g_1, g_0^r, g_1^r) = 0] - \Pr[\mathcal{A}(g_0, g_1, g_0^r, g_1^{r'}) = 0]|$ is negligible for any PPT algorithm $\mathcal{A}$, where g_0 and g_1 are randomly chosen from $\mathbb{G}$, and r and r' are randomly chosen from $\mathbb{Z}_p$.*

Our DDH-Based KH-PKE Scheme : Let $\mathsf{TCR}_1 : \mathbb{G}\times\mathbb{G}\times\mathbb{G} \to \mathbb{Z}_p$ and $\mathsf{TCR}_2 : \mathbb{G} \to \{0,1\}^{\log p/2}$ be TCR hash functions. We give our DDH-based KH-PKE scheme in Figure 2. Here, we explain the usage of $sk_h = ((\widehat{k}_{1,0}, \widehat{k}_{1,1}), (\widetilde{k}_0, \widetilde{k}_1, \widetilde{k}_{1,0}, \widetilde{k}_{1,1}))$. $\widehat{\pi}'_1 = \widehat{\pi}_1/(x_{1,0}^{\gamma_1\widehat{k}_{1,0}} x_{1,1}^{\gamma_1\widehat{k}_{1,1}}) = x_{1,0}^{k'_0} x_{1,1}^{k'_1}$ and $\widehat{\pi}'_2 = \widehat{\pi}_2/(x_{2,0}^{\gamma_2\widehat{k}_{1,0}} x_{2,1}^{\gamma_2\widehat{k}_{1,1}}) = x_{2,0}^{k'_0} x_{2,1}^{k'_1}$ hold using $(\widehat{k}_{1,0}, \widehat{k}_{1,1})$. So, $\widehat{\pi} \leftarrow \widehat{\pi}'_1\widehat{\pi}'_2 x_0^{\gamma\widehat{k}_{1,0}} x_1^{\gamma\widehat{k}_{1,1}} = x_0^{k'_0+\gamma\widehat{k}_{1,0}} x_1^{k'_1+\gamma\widehat{k}_{1,1}}$ holds. Therefore, the Eval algorithm works. The other keys $(\widetilde{k}_0, \widetilde{k}_1, \widetilde{k}_{1,0}, \widetilde{k}_{1,1})$ (and TCR_2) are used for computing $\widetilde{\pi}'_1$ (resp. $\widetilde{\pi}'_2$) to check the validity of C_1 (resp. C_2).

The following theorem can be proved in the same way as Theorem 1.

Theorem 2. *The proposed DDH-based KH-PKE scheme is KH-CCA-secure if the DDH assumption holds, and TCR_1 and TCR_2 are TCR hash functions.*

In Table 1, we give an efficiency comparison of our DDH-based KH-PKE scheme with the CS PKE [9], the KD PKE [23], and the naive construction (See Section 1). We note that these three schemes do not yield keyed-homomorphic property and/or KH-CCA security. As seen in Table 1, our scheme is comparably efficient to the best known DDH-based (standard) PKE schemes, i.e. the CS and the KD schemes, in terms of both ciphertext overhead and computational costs. Especially, ciphertext overhead of our scheme is only ℓ-bit longer than that of the CS scheme for ℓ-bit security. It is somewhat surprising that it is possible to realize KH property with only significantly small additional cost. Furthermore, comparing with the naive construction (from KD and CS(-lite)) which appears to have KH property (but does not satisfy KH-CCA security), we see that our scheme is more efficient. This means that our methodology does not only yield KH property (and KH-CCA security) but also significantly high efficiency.

Acknolwdgement. We thank anonymous reviewers and the members of Shin-Akarui-Angou-Benkyou-Kai for their helpful comments. The 4th and 5th authors are supported by a JSPS Fellowship for Young Scientists. This work was supported by JSPS KAKENHI Grant Number 24700009.

References

1. An, J.H., Dodis, Y., Rabin, T.: On the Security of Joint Signature and Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 83–107. Springer, Heidelberg (2002)
2. 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)
3. Bernhard, D., Cortier, V., Pereira, O., Smyth, B., Warinschi, B.: Adapting Helios for Provable Ballot Privacy. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 335–354. Springer, Heidelberg (2011)
4. Boneh, D., Segev, G., Waters, B.: Targeted malleability: homomorphic encryption for restricted computations. In: ICTS, pp. 350–366 (2012)
5. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
6. Canetti, R., Krawczyk, H., Nielsen, J.B.: Relaxing Chosen-Ciphertext Security. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 565–582. Springer, Heidelberg (2003)
7. Cash, D.M., Kiltz, E., Shoup, V.: The Twin Diffie-Hellman Problem and Applications. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 127–145. Springer, Heidelberg (2008)
8. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable Proof Systems and Applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 281–300. Springer, Heidelberg (2012)
9. Cramer, R., Shoup, V.: A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
10. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. Cryptology ePrint Archive, Report 2001/085 (2001), `http://eprint.iacr.org/`
11. Cramer, R., Shoup, V.: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
12. Galindo, D., Villar, J.L.: An instantiation of the Cramer-Shoup encryption paradigm using bilinear map groups. In: Workshop on Mathematical Problems and Techniques in Cryptology (2005)
13. Gamal, T.E.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31(4), 469–472 (1985)
14. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
15. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)

16. Goldwasser, S., Micali, S.: Probabilistic encryption and how to play mental poker keeping secret all partial information. In: STOC, pp. 365–377 (1982)
17. Groth, J.: Rerandomizable and Replayable Adaptive Chosen Ciphertext Attack Secure Cryptosystems. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 152–170. Springer, Heidelberg (2004)
18. Hanaoka, G., Kurosawa, K.: Efficient Chosen Ciphertext Secure Public Key Encryption under the Computational Diffie-Hellman Assumption. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 308–325. Springer, Heidelberg (2008)
19. Hemenway, B., Ostrovsky, R.: On Homomorphic Encryption and Chosen-Ciphertext Security. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 52–65. Springer, Heidelberg (2012)
20. Kiltz, E.: Chosen-Ciphertext Security from Tag-Based Encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006)
21. Kiltz, E.: Chosen-Ciphertext Secure Key-Encapsulation Based on Gap Hashed Diffie-Hellman. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 282–297. Springer, Heidelberg (2007)
22. Kiltz, E., Pietrzak, K., Stam, M., Yung, M.: A New Randomness Extraction Paradigm for Hybrid Encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 590–609. Springer, Heidelberg (2009)
23. Kurosawa, K., Desmedt, Y.: A New Paradigm of Hybrid Encryption Scheme. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 426–442. Springer, Heidelberg (2004)
24. Loftus, J., May, A., Smart, N.P., Vercauteren, F.: On CCA-Secure Somewhat Homomorphic Encryption. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 55–72. Springer, Heidelberg (2012)
25. 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)
26. Prabhakaran, M., Rosulek, M.: Rerandomizable RCCA Encryption. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 517–534. Springer, Heidelberg (2007)
27. Prabhakaran, M., Rosulek, M.: Homomorphic Encryption with CCA Security. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 667–678. Springer, Heidelberg (2008)
28. Shacham, H.: A Cramer-Shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074 (2007), `http://eprint.iacr.org/`
29. Shoup, V.: A proposal for an ISO standard for public key encryption. Cryptology ePrint Archive, Report 2001/112 (2001), `http://eprint.iacr.org/`

Appendix: Proof of Theorem 1

Proof. Let $\mathcal{A}$ be an adversary who breaks KH-CCA security. Then, we construct an algorithm $\mathcal{B}$ that can break the hardness of $\mathbf{M}$. To later calculate the concrete advantage of $\mathcal{A}$, let $\epsilon(\ell)$, $\widehat{\epsilon}(\ell)$, and $\widetilde{\epsilon}(\ell)$ be negligible functions such that $\mathbf{P}$ be $\epsilon(\ell)$-smooth, and $\widehat{\mathbf{P}}$ be homomorphic transitional $(\widehat{\epsilon}(\ell), \widehat{\epsilon}(\ell))$-universal, and $\widetilde{\mathbf{P}}$ be $\widetilde{\epsilon}(\ell)$-universal$_2$.

We describe how $\mathcal{B}$ simulates the KH-CCA experiment for $\mathcal{A}$. First, $\mathcal{B}$ takes as input 1^ℓ along with $\Lambda[X, L, W, R] \in [I_\ell]$ and $x^* \in X$. $\mathcal{B}$ runs $(pk, sk_d, sk_h) \leftarrow \mathsf{KeyGen}(1^\ell)$ as usual using the given value of Λ, where $pk = (s, \widehat{s}, \widetilde{s})$, $sk_d = (k, \overrightarrow{k}, \widetilde{k}) = (k, (k', \widehat{k}), \widetilde{k})$, and $sk_h = (\widehat{k}, \widetilde{k})$. $\mathcal{B}$ sends pk to $\mathcal{A}$.

In find stage, $\mathcal{B}$ answers for each query as follows: For a decryption query C, $\mathcal{B}$ runs $\mathsf{Dec}(sk_d, C)$ as usual using sk_d, and returns the result of the decryption algorithm. For an evaluation query (C_1, C_2), $\mathcal{B}$ runs $\mathsf{Eval}(sk_h, C_1, C_2)$ as usual using sk_h, and returns the result of the evaluation algorithm. For the reveal homomorphic key query, $\mathcal{B}$ returns $sk_h = (\widehat{k}, \widetilde{k})$. In the challenge phase, $\mathcal{A}$ sends (M_0^*, M_1^*) to $\mathcal{B}$. $\mathcal{B}$ chooses $\beta \xleftarrow{\$} \{0, 1\}$, and computes $\pi^* \leftarrow H_k(x^*)$ using the private evaluation algorithm, $e^* = \pi^* + M_\beta^*$, and $\widehat{\pi}^* \leftarrow \widehat{H}_{k', \widehat{k}}(x^*, e^*)$ and $\widetilde{\pi}^* \leftarrow \widetilde{H}_{\widetilde{k}}(x^*, e^*)$ using the private evaluation algorithm, and sends $C^* = (x^*, e^*, \widehat{\pi}^*, \widetilde{\pi}^*)$ to $\mathcal{A}$. In addition, $\mathcal{B}$ sets a ciphertext dictionary $\mathcal{D}$ such that $\mathcal{D} = \{C^*\}$. In guess stage, $\mathcal{B}$ answers for each query as follows: For a decryption query C, if $C \in \mathcal{D}$, then return $\perp$. Otherwise, $\mathcal{B}$ runs $\mathsf{Dec}(sk_d, C)$ as usual using sk_d, and returns the result of the decryption algorithm. For an evaluation query (C_1, C_2), $\mathcal{B}$ runs $\mathsf{Eval}(sk_h, C_1, C_2)$ as usual using sk_h, and returns the result of the evaluation algorithm. If either $C_1 \in \mathcal{D}$ or $C_2 \in \mathcal{D}$, then $\mathcal{B}$ updates $\mathcal{D} \leftarrow \mathcal{D} \cup \{C_3\}$, where $C_3 = \mathsf{Eval}(sk_h, C_1, C_2)$. Note that if $C_3 = \perp$, then $\mathcal{B}$ does not update the dictionary $\mathcal{D}$. For the reveal homomorphic key query, $\mathcal{B}$ returns $sk_h = (\widehat{k}, \widetilde{k})$. Finally, $\mathcal{A}$ outputs a guessing bit β'. $\mathcal{B}$ outputs 1 if $\beta = \beta'$, and 0 otherwise.

Next, we define two experiments according to whether $x^* \in L$ or $x^* \in X \setminus L$. In the first experiment, $\mathcal{B}$ is given (Λ, x^*), where $\Lambda[X, L, W, R] \in [I_\ell]$ and $x^* \in L$. Let T'_ℓ be the event that $\mathcal{B}$ outputs 1 in this experiment. In the second experiment, $\mathcal{B}$ is given (Λ, x^*), where $\Lambda[X, L, W, R] \in [I_\ell]$ and $x^* \in X \setminus L$. Let T_ℓ be the event that $\mathcal{B}$ outputs 1 in this experiment. By definition of the subset membership problem, the advantage of $\mathcal{B}$ is defined as $\mathsf{AdvDist}(\ell) := |\Pr[T_\ell] - \Pr[T'_\ell]|$. Let $Q_{dec}(\ell)$ be the number of decryption queries and $Q_{eval}(\ell)$ be the number of evaluation queries. In the case of $x^* \in L$, the simulation of the KH-CCA game for the adversary $\mathcal{A}$ is perfect. Thus, we get $\left|\Pr[T'_\ell] - \frac{1}{2}\right| \geq Adv^{KH\text{-}CCA}_{\mathsf{KH\text{-}PKE},\mathcal{A}}(\ell)$ In the case of $x^* \in X \setminus L$, we define the sequences of games. We denote $T_\ell^{(0)}$, $T_\ell^{(1)}$, and $T_\ell^{(2)}$ as the event that $\mathcal{B}$ outputs 1 in the game 0, 1, and 2, respectively.

Game 0: The same as the KH-CCA simulation.

Game 1: Recall that in Game 0, the decryption oracle (and the evaluation oracle also) rejects a query $(x, e, \widehat{\pi}, \widetilde{\pi})$ if either $\widehat{H}_{k', \widehat{k}}(x, e) \neq \widehat{\pi}$ or $\widetilde{H}_{\widetilde{k}}(x, e) \neq \widetilde{\pi}$. In this game, we make these oracles reject a query that contains a ciphertext $(x, e, \widehat{\pi}, \widetilde{\pi})$ satisfying $x \notin L$. Let F_2 be the event that these oracles reject a query $(x, e, \widehat{\pi}, \widetilde{\pi})$ with $x \notin L$, but either $\widehat{H}_{k', \widehat{k}}(x, e) = \widehat{\pi}$ or $\widetilde{H}_{\widetilde{k}}(x, e) = \widetilde{\pi}$ holds. In the find phase, $\widehat{\alpha}(k', \widehat{k}) = \widehat{s}$ and $\widetilde{\alpha}(\widetilde{k}) = \widetilde{s}$ are fixed. Then, the probability that $\widehat{H}_{k', \widehat{k}}(x, e) = \widehat{\pi}$ is at most $\widehat{\epsilon}(\ell)$, since $\widehat{\mathbf{H}}$ is a $\widehat{\epsilon}$-universal$_2$ (or $\widehat{\epsilon}$-universal$_1$ projective, if $\mathcal{A}$ has been an insider via the RevHK oracle) hash family, and

the probability that $\widetilde{H}_{\widetilde{k}}(x,e) = \widetilde{\pi}$ is at most $\widetilde{\epsilon}(\ell)$, since $\widetilde{\mathbf{H}}$ is a $\widetilde{\epsilon}$-universal$_2$ hash family. In the challenge phase, $\widehat{\pi}^* = \widehat{H}_{k',\widehat{k}}(x^*,e^*)$ and $\widetilde{\pi}^* = \widetilde{H}_{\widetilde{k}}(x^*,e^*)$ are fixed. After this, in the guess stage, the probability that $\widehat{H}_{k',\widehat{k}}(x,e) = \widehat{\pi}$ is at most $\widehat{\epsilon}(\ell)$, since $\widehat{\mathbf{H}}$ is a $\widehat{\epsilon}$-universal$_2$. Note that if $\mathcal{A}$ has been an insider, then $\mathcal{A}$ does not issue the decryption query. In addition, the probability that $\widetilde{H}_{\widetilde{k}}(x,e) = \widetilde{\pi}$ is at most $\widetilde{\epsilon}(\ell)$, since $\widetilde{\mathbf{H}}$ is a $\widetilde{\epsilon}$-universal$_2$. To sum up, we get $\Pr[F_2] \leq Q_{dec}(\ell)(\widehat{\epsilon}(\ell)+\widetilde{\epsilon}(\ell))+2Q_{eval}(\ell)\widetilde{\epsilon}(\ell)$. The term $2Q_{eval}(\ell)$ is derived from the fact that an evaluation query contains two ciphertexts. In addition, from the fact that Game 0 and Game 1 are identical if the event F_2 does not occur, we get $\left|\Pr[T_\ell^{(1)}] - \Pr[T_\ell^{(0)}]\right| \leq \Pr[F_2] \leq Q_{dec}(\ell)(\widehat{\epsilon}(\ell)+\widetilde{\epsilon}(\ell))+2Q_{eval}(\ell)\widetilde{\epsilon}(\ell)$.

Game 2: In this game, $\mathcal{B}$ chooses $\pi^* \xleftarrow{\$} \Pi$ (instead of computing $\pi^* = H_k(x^*)$) and computes $e^* = \pi^* + M^*_\beta$. Since $\mathbf{H}$ is an $\epsilon(\ell)$-smooth projective hash family and β is hidden by π^*, we get $\left|\Pr[T_\ell^{(2)}] - \Pr[T_\ell^{(1)}]\right| \leq \epsilon(\ell)$ and $\Pr[T_\ell^{(2)}] = \frac{1}{2}$. By combining the inequalities, we get $Adv^{KH\text{-}CCA}_{\mathsf{KH\text{-}PKE},\mathcal{A}}(\ell) \leq \mathsf{AdvDist}(\ell) + Q_{dec}(\ell)\widehat{\epsilon}(\ell) + (Q_{dec}(\ell)+2Q_{eval}(\ell))\widetilde{\epsilon}(\ell)+\epsilon(\ell)$, which is negligible. □

Functional Encryption: Origins and Recent Developments

Brent Waters*

The University of Texas at Austin
bwaters@cs.utexas.edu

Abstract. In this talk, we will present the notion of functional encryption and recent progress in the area. We will begin by describing the concept and origins of functional encryption. Next, we will describe intuitively why current bilinear map based constructions appear to be "stuck" with boolean formula type functionality even in the public index setting. Finally, we will see some very recent work that uses multilinear forms to move beyond these barriers and achieve functionality for any circuit.

Overview

Encryption is a method to encode data such that it can only be understood by a recipient that holds a certain private key object. The traditional notion of public key encryption [10,11,21,13] is that a data owner will encrypt data to the public key of a specific targeted user to create a ciphertext. Later, a user possessing the corresponding private key can decrypt the ciphertext to obtain the original data. Ingrained in this notion is that: (1) *Encryption is a method to target to a specific user.* (2) *Decryption is an all or nothing operation; either a ciphertext is fully decrypted and the original data is recovered or else it fails and nothing is learned.*

Functional encryption is a new vision of encryption that moves pass these barriers. In a Functional Encryption system what a user learns from decryption is determined by a function of the encrypted data and the user's secret key descriptor (as issued by some authority). Briefly, in a functional encryption system with functionality $F(\cdot,\cdot)$ a user is issued a secret key sk_k for value k by some authority. Suppose that a ciphertext ct is the encryption of data x. The user can apply their secret key to learn $F(k,x)$.

Functional encryption for expressive functionalities open up a wide variety of applications. For instance, one might determine access to encrypted data based on an arbitrary policy over a user's credentials. Another possibility is that encrypted data could consist of images and a user's private key of their headshot.

* Supported by NSF CNS-0915361 and CNS-0952692, CNS-1228599 DARPA through the U.S. Office of Naval Research under Contract N00014-11-1-0382, DARPA N11AP20006, Google Faculty Research award, the Alfred P. Sloan Fellowship, and Microsoft Faculty Fellowship, and Packard Foundation Fellowship.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 51–54, 2013.

The user would be able to view the image only if their face appeared in it as determined by some vision recognition algorithm. Moreover, the functionality could blur out parts of the image not immediately surrounding the user's body. In a medical research environment, one could consider encrypting a large database containing medical histories of patients coupled with DNA sequencing. Later, if a researcher is granted permission to test a correlation between a certain type of cancer and genotype they could be given a secret key that divulges the correlation and nothing else.

Origins of Functional Encryption

The origins of functional encryption can be traced to the concept of Attribute-Based Encryption (ABE) [23] proposed by Sahai and Waters. In a (Key-Policy) ABE scheme a ciphertext contains a hidden message as well as (unhidden) metadata or attributes. A user's private key is associated with a formula ϕ. A user can decrypt a given ciphertext and recover the hidden message if and only if the formula is satisfied when its values are assigned according to the metadata. A technical lynchpin was the concept that any secure system must be *collusion resistant*. Suppose an attacker obtains multiple secret keys, e.g., $\mathrm{sk}_k, \mathrm{sk}_{k'}$. In particular, the attacker should not be able to combine two private keys to decrypt ciphertexts that neither private key was authorized for.

While Attribute-Based Encryption moves beyond the notion of encrypting to a particular user, decryption is still an all or nothing proposition. In subsequent works [6,16] the concept evolved to hide the metadata. The notion of Functional Encryption first appeared in presentation slides prepared by Sahai and Waters in 2008 [24] and was described during talks given by both authors. Significant conceptual work was done while both Sahai and Waters were researchers at IPAM for the 2006 Securing Cyberspace program. The term functional encryption first appeared in a published research paper by Lewko et. al. [17]. Finally, a definitional framework for functional encryption was put forward by Boneh, Sahai, and Waters [5] where they put forward both simulation and indistinguishability definitions.[1] The above work was influenced by concepts such as Identity-Based Encryption (IBE) [26,4,9] and Anonymous IBE [3,1].

Achieving Stronger Functionality

Over the past several years there has been significant research activity on a variety directions in functional encryption including proofs of adaptive security [17,19], revocation of secret keys [2,22], policies across multiple authorities [7,8,18], and investigation of definitions [5,20]. *Arguably, the most important question is what functionality can we achieve.* For several years the strongest form of expression we had was boolean formulas[2] in ABE cryptosystems. While

[1] Concurrently, with [5] and subsequent to discussions stemming from [24], O'Neill [20] also put forward general definitions for functional encryption.

[2] Technically, one can obtain span programs.

boolean formula ABE systems give rise to several interesting applications, they are still a far cry from being able to express access control in the form of any program or circuit.

In this talk we will first explore the techniques that give rise to ABE systems for boolean formulas. Our starting point will be the "Key-Policy" ABE system of Goyal et. al. [15]. We will see how they use bilinear maps as the primary mechanism for decryption blended with interpolation in the exponent techniques. Together these give the boolean formula functionality and the needed protection against collusion attacks. We also give insight into the difficulty of obtaining stronger functionality using bilinear maps by arguing why such constructions are "stuck" at the level of boolean formulas. Intuitively, the bilinear map mechanism is "used up" in pairing the ciphertext with the secret key to prevent collusions between different users. However, this leaves natural larger fanout generalizations of GPSW to so called backtracking attacks.

We will next describe some very recent progress [25] that obtains Attribute-Based Encryption for circuits. Obtaining ABE for circuits is a major jump in that circuits can express any program of fixed running time. The new result is obtained by applying the recent work of Garg, Gentry, and Halevi [12] which describes some approximation of groups with multilinear maps. The new ABE crypto leverages these multilinear forms to create a new "move forward and shift" technique for decryption that replaces and subsumes the prior methods. Independently, Gorbunov, Vaikuntanathan and Wee [14] obtained the same result under the Learning with Error (LWE) assumption. They create a set of novel and elegant techniques to combat the backtracking issue. We refer the reader to the introduction of [25] for further discussion of backtracking attacks and how these are circumvented by new techniques.

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. J. Cryptology 21(3), 350–391 (2008)
2. Boldyreva, A., Goyal, V., Kumar, V.: Identity-based encryption with efficient revocation. In: ACM Conference on Computer and Communications Security, pp. 417–426 (2008)
3. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
4. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. SIAM J. Comput. 32(3), 586–615, (2003); Extended abstract in: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boneh, D., Sahai, A., Waters, B.: Functional Encryption: Definitions and Challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011)
6. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)

7. Chase, M.: Multi-authority Attribute Based Encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
8. Chase, M., Chow, S.S.M.: Improving privacy and security in multi-authority attribute-based encryption. In: ACM Conference on Computer and Communications Security, pp. 121–130 (2009)
9. 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)
10. Diffie, W., Hellman, M.E.: Multiuser cryptographic techniques. In: AFIPS National Computer Conference, pp. 109–112 (1976)
11. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory IT-22(6), 644–654 (1976)
12. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices and applications. Cryptology ePrint Archive, Report 2012/610 (2012), http://eprint.iacr.org/
13. Goldwasser, S., Micali, S.: Probabilistic encryption. Jour. of Computer and System Science 28(2), 270–299 (1984)
14. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Personal Communication (2012)
15. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
16. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
17. 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)
18. Lewko, A., Waters, B.: Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011)
19. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
20. O'Neill, A.: Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556 (2010), http://eprint.iacr.org/2010/556
21. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21(2), 120–126 (1978)
22. Sahai, A., Seyalioglu, H., Waters, B.: Dynamic Credentials and Ciphertext Delegation for Attribute-Based Encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 199–217. Springer, Heidelberg (2012)
23. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
24. Sahai, A., Waters, B.: Slides on functional encryption. PowerPoint presentation (2008), http://www.cs.utexas.edu/ bwaters/presentations/files/functional.ppt
25. Sahai, A., Waters, B.: Attribute-based encryption for circuits from multilinear maps. Cryptology ePrint Archive, Report 2012/592 (2012), http://eprint.iacr.org/
26. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)

Vector Commitments and Their Applications

Dario Catalano[1] and Dario Fiore[2,⋆]

[1] Dipartimento di Matematica e Informatica, Università di Catania, Italy
catalano@dmi.unict.it
[2] Max Planck Institute for Software Systems (MPI-SWS)
fiore@mpi-sws.org

Abstract. We put forward the study of a new primitive that we call *Vector Commitment* (VC, for short). Informally, VCs allow to commit to an ordered sequence of q values $(m_1, \ldots, m_q)$ in such a way that one can later open the commitment at specific positions (e.g., prove that m_i is the i-th committed message). For security, Vector Commitments are required to satisfy a notion that we call *position binding* which states that an adversary should not be able to open a commitment to two different values at the same position. Moreover, what makes our primitive interesting is that we require VCs to be *concise*, i.e. the size of the commitment string and of its openings has to be independent of the vector length.

We show two realizations of VCs based on standard and well established assumptions, such as RSA, and Computational Diffie-Hellman (in bilinear groups). Next, we turn our attention to applications and we show that Vector Commitments are useful in a variety of contexts, as they allow for compact and efficient solutions which significantly improve previous works either in terms of efficiency of the resulting solutions, or in terms of "quality" of the underlying assumption, or both. These applications include: Verifiable Databases with Efficient Updates, Updatable Zero-Knowledge Databases, and Universal Dynamic Accumulators.

1 Introduction

Commitment schemes are one of the most important primitives in cryptography. Informally, they can be seen as the digital equivalent of a sealed envelope: whenever a party S wants to commit to a message m, she puts m in the envelope. At a later moment, S opens the envelope to publicly reveal the message she committed to. In their most basic form commitment schemes are expected to meet two requirements. A commitment should be *hiding*, meaning with this that it should not reveal information about the committed message, and *binding* which means that the committing mechanism should not allow S to change her mind about m. More precisely, this means that the commitment comes with an opening procedure that can be efficiently verified, i.e. one should be able to efficiently check that the opened message is the one S originally committed to.

⋆ Work done while at NYU supported by NSF grant CNS-1017471.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 55–72, 2013.

Thus, a commitment scheme typically involves two phases: a *committing* one, where a sender S creates a commitment C on some messages m, using some appropriate algorithm and a *decommitting* stage, where S reveals m and should "convince" a receiver R that C contains m. A commitment scheme is said to be non-interactive if each phase requires only one messages from S to R.

Commitment schemes turned out to be extremely useful in cryptography and have been used as a building block to realize highly non-trivial protocols and primitives. Because of this, the basic properties discussed above have often turned out to be insufficient for realizing the desired functionalities. This led researchers to investigate more complex notions realizing additional properties and features. Here we discuss a couple of these extensions, those more closely related to the results presented in this paper.

Trapdoor commitment schemes (also known as *chameleon* commitments) have a public key and a (matching) secret key (also known as the trapdoor). Knowledge of the trapdoor allows to completely destroy the binding property. On the other hand, the scheme remains binding for those who know only the public key. A special case of trapdoor commitments are (trapdoor) Mercurial commitments, a notion formalized by Chase *et al.* in [12]. Here the binding property is further relaxed to allow for two different decommitting procedures: a *hard* and a *soft* one. In the committing phase one can decide as whether to create a hard commitment or a soft one. A hard commitment is like a standard one: it is created to a specific message m, and it can be opened only to m. Instead, a soft commitment is initially created to "no message", and it can later be soft-opened (or *teased*) to any m, but it cannot be hard-opened.

Our Contributions. In this paper we introduce a new and simple, yet powerful notion of commitment, that we call *Vector Commitment* (VC, for short). Informally, VCs allow to commit to an ordered sequence of q values (i.e. a vector), rather than to single messages. This is done in a way such that it is later possible to open the commitment w.r.t. specific positions (e.g., to prove that m_i is the i-th committed message). More precisely, vector commitments are required to satisfy what we call *position binding*. Position binding states that an adversary should not be able to open a commitment to two different values at the same position. While this property, by itself, would be trivial to realize using standard commitment schemes, what makes our design interesting is that we require VCs to be *concise*, i.e., the size of the commitment string as well as the size of each opening have to be independent of the vector length.

Vector commitments can also be required to be *hiding*, in the sense that one should not be able to distinguish whether a commitment was created to a vector $(m_1, \ldots, m_q)$ or to $(m'_1, \ldots, m'_q)$, even after seeing some openings. We, however, notice that hiding is not a crucial property in the realization of vector commitments. Therefore, in our constructions we will not focus on it. While this might be surprising at first, we motivate it as follows. First, all the applications of VCs described in this paper do not require such a property. Second, hiding VCs can be easily obtained by composing a non-hiding VC with a standard commitment scheme (see Section 2 for more details).

Additionally, Vector Commitments need to be *updatable*. Very roughly, this means that they come equipped with two algorithms to update the commitment and the corresponding openings. The first algorithm allows the committer, who created a commitment $\texttt{Com}$ and wants to update it by changing the i-th message from m_i to m'_i, to obtain a (modified) $\texttt{Com}'$ containing the updated message. The second algorithm allows holders of an opening for a message at position j w.r.t. $\texttt{Com}$ to update their proof so as to become valid w.r.t. the new $\texttt{Com}'$.

Next, we turn our attention to the problem of realizing vector commitments. Our technical contributions are two realizations of VCs from standard and well established assumptions, namely RSA and Computational Diffie-Hellman (over bilinear groups)[1].

Finally, we confirm the power of this new primitive by showing several applications (see below) in which our notion of Vector Commitment allows for compact and efficient solutions, which significantly improve previous works either in terms of efficiency of the resulting solutions, or in terms of "quality" of the underlying assumption, or both.

Verifiable Databases with Efficient Updates. Very recently, Benabbas, Gennaro and Vahlis [3] formalized the notion of Verifiable Databases with Efficient Updates (VDB, for short). This primitive turns out to be extremely useful to solve the following problem in the context of verifiable outsourcing of storage. Assume that a client with limited resources wants to store a large database on a server so that it can later retrieve a database record, and update a record by assigning a new value to it. For efficiency, it is crucial that the computational resources invested by the client to perform such operations must not depend on the size of the database (except for an initial pre-processing phase). On the other hand, for security, the server should not be able to tamper with any record of the database without being detected by the client.

For the static case (i.e., the client does not perform any update) simple solutions can be achieved by using message authentication or signature schemes. For example, the client first signs each database record before sending it to the server, and then the server is requested to output the record together with its valid signature. However, this idea does not work well if the client performs updates on the database. The problem is that the client should have a mechanism to revoke the signatures given to the server for the previous values. To solve this issue, the client could keep track of every change locally, but this is in contrast with the main goal, i.e., using less resources than those needed to store the database locally.

Solutions to this problem have been addressed by works on accumulators [26,6,7], authenticated data structures [25,21,27,30], and the recent work on verifiable computation [3]. Also, other recent works have addressed a slightly different and more practical problem of realizing authenticated remote file systems [29]. However, as pointed out in [3], previous solutions based on accumulators and authenticated data structures either rely on non-constant size assumptions

[1] Precisely, our construction relies on the Square Computational Diffie-Hellman assumption which however has been shown equivalent to the standard CDH [22,1].

(such as q-Strong Diffie-Hellman), or they require expensive operations such as generation of prime numbers, and re-shuffling procedures. Benabbas *et al.* propose a nice solution with efficient query and update time [3]. Their scheme relies on a constant size assumption in bilinear groups of composite order, but does not support public verifiability (i.e., only the client owner of the database can verify the correctness of the proofs provided by the server).

In this work, we show that Vector Commitments can be used to build Verifiable Databases with efficient updates that allow for public verifiability. More importantly, if we instantiate this construction with our VC based on CDH, then we obtain an implementation of Verifiable Databases that relies on a standard constant-size assumption, and whose efficiency improves over the scheme of Benabbas *et al.* as we can use bilinear groups of *prime order*.

Updatable Zero Knowledge Elementary Databases. Zero Knowledge Sets allow a party P, called the *prover*, to commit to a secret set S in a way such that he can later produce proofs for statements of the form $x \in S$ or $x \notin S$. The required properties are the following. First, any user V (the *verifier*) should be able to check the validity of the received proofs without learning any information on S (not even its size) beyond the mere membership (or non-membership) of the queried elements. Second, the produced proofs should be reliable in the sense that no dishonest prover should be able to convince V of the validity of a false statement. Zero Knowledge Sets (ZKS) were introduced and constructed by Micali, Rabin and Kilian [23][2]. Micali *et al.*'s construction was abstracted away by Chase *et al.* [12], and by Catalano, Dodis and Visconti [8]. The former showed that ZKS can be built from trapdoor mercurial commitments and collision resistant hash functions, and also that ZKS imply collision-resistant hash functions. The latter showed generic constructions of (trapdoor) mercurial commitments from the sole assumptions that one-way functions exist. These results taken together [12,8], thus, show that collision-resistant hash functions are necessary and sufficient to build ZKS in the CRS model. From a practical perspective, however, none of the above solutions can be considered efficient enough to be used in practice. A reason is that all of them allow to commit to a set $S \subset \{0,1\}^k$ by constructing a Merkle tree of depth k, where each internal node is filled with a mercurial commitment (rather than the hash) of its two children. A proof that $x \in \{0,1\}^k$ is in the committed set consists of the openings of all the commitments in the path from the root to the leaf labeled by x (more details about this construction can be found in [23,12]). This implies that proofs have size linear in the height k of the tree. Now, since 2^k is an upper bound for $|S|$, to guarantee that no information about $|S|$ is revealed, k has to be chosen so that 2^k is much larger than any reasonable set size.

[2] More precisely, Micali *et al.* addressed the problem for the more general case of elementary databases (EDB), where each key x has associated a value $D(x)$ in the committed database. In the rest of this paper we will slightly abuse the notation and use the two acronyms ZKS and ZK-EDB interchangeably to indicate the same primitive.

Catalano, Fiore and Messina addressed in [10] the problem of building ZKS with shorter proofs. Their proposed idea was a construction that uses q-ary trees, instead of binary ones, and suggested an extension of mercurial commitment (that they called q-Trapdoor Mercurial Commitment) which allows to implement it. The drawback of the specific realization of qTMC in [10] is that it is not as efficient as one might want. In particular, while the size of soft openings is independent of q, hard openings grow linearly in q. This results in an "unbalanced" ZK-EDB construction where proofs of membership are much longer than proofs of non membership. In a follow-up work, Libert and Yung [19] proposed a very elegant solution to this problem. Specifically, they managed to construct a q-mercurial commitment (that they called *concise*) achieving constant-size (soft and hard) openings. This resulted in ZK-EDB with very short proofs, as by increasing q one can get an arbitrarily "flat" tree[3]. Similarly to [10], the scheme of Libert and Yung [19] also relies on a non-constant size assumption in bilinear groups: the q-Diffie-Hellman Exponent [5].

Our main application of VCs to ZKS is the proof of the following theorem:

Theorem 1 (informal). *A (concise) trapdoor q-mercurial commitment can be obtained from a vector commitment and a trapdoor mercurial commitment.*

The power of this theorem comes from the fact that, by applying the generic transform of Catalano *et al.* [10], we can immediately conclude that Compact ZKS (i.e. ZKS with short membership and non-membership proofs) can be built from mercurial commitments and vector commitments. Therefore, when combining our realizations of Vector Commitments with well known (trapdoor) mercurial ones (such as that of Gennaro and Micali [14] for the RSA case, or that from [23], for the CDH construction) we get concise qTMCs from RSA and CDH. Moreover, when instantiating the ZK-EDB construction of Catalano *et al.* [10] with such schemes, one gets the first compact ZK-EDB realizations which are provably secure under standard assumptions. Our CDH realization induces proofs whose length is comparable to that induced by Libert and Yung's commitment [19], while relying on more standard and better established assumptions.

Additionally, and more importantly, we show the first construction of *updatable* ZK-EDB with short proofs. The notion of Updatable Zero Knowledge EDB was introduced by Liskov [20] to extend ZK-EDB to the (very natural) case of "dynamic" databases. In an updatable ZK-EDB the prover is allowed to change the value of some element x in the database and then output a new commitment C' and some update information U. Users holding a proof π_y for a $y \neq x$ valid w.r.t. C, should be able to use U to produce an updated proof π'_y that is valid w.r.t. C'. In [20] is given a definition of *Updatable Zero Knowledge (Elementary) Databases* together with a construction based on mercurial commitments and Verifiable Random Functions [24] in the random oracle model. More precisely, Liskov introduced the notion of *updatable mercurial commitment* and proposed a construction, based on discrete logarithm, which is a variant of the mercurial commitment of Micali *et al.* [23].

[3] The only limitation is that the resulting CRS grows linearly in q.

Using Vector Commitments, we realize the *first* constructions of "compact" Updatable ZK-EDB whose proofs and updates are much shorter than those of Liskov [20]. In particular, we show how to use VCs to build Updatable ZK-EDB from updatable qTMCs (which we also define and construct) and Verifiable Random Functions in the random oracle model. We stress that our solutions, in addition to solving the open problem of realizing Updatable ZK-EDB with short proofs, further improve on previous work as they allow for much shorter updates as well[4].

Additional Applications of Vector Commitments. We leave to the full version of this paper [9] a description of additional applications of Vector Commitments to compact *Independent Zero-Knowledge Databases* [14], *Fully Dynamic Universal Accumulators* [4,7,17], and *pseudonymous credentials* [16]. Very recently, Libert, Peters and Yung also used our Vector Commitments to improve the efficiency of group signatures with revocation [18].

Preliminaries and Definitions. In what follows we will denote with $k \in \mathbb{N}$ the security parameter, and by $poly(k)$ any function which bounded by a polynomial in k. An algorithm $\mathcal{A}$ is said to be PPT if it is modeled as a probabilistic Turing machine that runs in time polynomial in k. Informally, we say that a function is *negligible* if it vanishes faster than the inverse of any polynomial. If S is a set, then $x \xleftarrow{\$} S$ indicates the process of selecting x uniformly at random over S (which in particular assumes that S can be sampled efficiently). If n is an integer, we denote with $[n]$, the set containing the integers $1, 2, \ldots, n$.

2 Vector Commitments

In this section we introduce the notion of *Vector Commitment*. Informally speaking, a vector commitment allows to commit to an ordered sequence of values in such a way that it is later possible to open the commitment only w.r.t. a specific position. We define Vector Commitments as a non-interactive primitive, that can be formally described via the following algorithms:

$\mathsf{VC.KeyGen}(1^k, q)$ Given the security parameter k and the size q of the committed vector (with $q = poly(k)$), the key generation outputs some public parameters pp (which implicitly define the message space $\mathcal{M}$).

$\mathsf{VC.Com}_{\mathsf{pp}}(m_1, \ldots, m_q)$ On input a sequence of q messages $m_1, \ldots, m_q \in \mathcal{M}$ and the public parameters pp, the committing algorithm outputs a commitment string C and an auxiliary information aux.

$\mathsf{VC.Open}_{\mathsf{pp}}(m, i, \mathsf{aux})$ This algorithm is run by the committer to produce a proof Λ_i that m is the i-th committed message.

$\mathsf{VC.Ver}_{\mathsf{pp}}(C, m, i, \Lambda_i)$ The verification algorithm accepts (i.e., it outputs 1) only if Λ_i is a valid proof that C was created to a sequence $m_1, \ldots, m_q$ such that $m = m_i$.

[4] This is because, in all known constructions, the size of the update information linearly depends on the height of the tree.

$\mathsf{VC.Update}_{\mathsf{pp}}(C, m, m', i)$ This algorithm is run by the committer who produced C and wants to update it by changing the i-th message to m'. The algorithm takes as input the old message m, the new message m' and the position i. It outputs a new commitment C' together with an update information U.

$\mathsf{VC.ProofUpdate}_{\mathsf{pp}}(C, \Lambda_j, m', i, U)$ This algorithm can be run by any user who holds a proof Λ_j for some message at position j w.r.t. C, and it allows the user to compute an updated proof Λ'_j (and the updated commitment C') such that Λ'_j will be valid w.r.t. C' which contains m' as the new message at position i. Basically, the value U contains the update information which is needed to compute such values.

For correctness, we require that $\forall k \in \mathbb{N}, q = poly(k)$, for all honestly generated parameters $\mathsf{pp} \xleftarrow{\$} \mathsf{VC.KeyGen}(1^k, q)$, if C is a commitment on a vector $(m_1, \ldots, m_q) \in \mathcal{M}^q$ (obtained by running $\mathsf{VC.Com}_{\mathsf{pp}}$ possibly followed by a sequence of updates), Λ_i is a proof for position i generated by $\mathsf{VC.Open}_{\mathsf{pp}}$ or $\mathsf{VC.ProofUpdate}_{\mathsf{pp}}$ ($\forall i = 1, \ldots, q$), then $\mathsf{VC.Ver}_{\mathsf{pp}}(C, m_i, i, \mathsf{VC.Open}_{\mathsf{pp}}(m_i, i, \mathsf{aux}))$ outputs 1 with overwhelming probability.

The attractive feature of vector commitments is that they are required to meet a very simple security requirement, that we call *position binding*. Informally, this says that it should be infeasible, for any polynomially bounded adversary having knowledge of pp, to come up with a commitment C and two different valid openings for the same position i. More formally:

Definition 2. [Position Binding] A vector commitment satisfies position binding if $\forall i = 1, \ldots, q$ and for every PPT adversary $\mathcal{A}$ the following probability (which is taken over all honestly generated parameters) is at most negligible in k:

$$Pr\left[\begin{array}{l}\mathsf{VC.Ver}_{\mathsf{pp}}(C, m, i, \Lambda) = 1 \wedge \\ \mathsf{VC.Ver}_{\mathsf{pp}}(C, m', i, \Lambda') = 1 \wedge m \neq m'\end{array} \mid (C, m, m', i, \Lambda, \Lambda') \leftarrow \mathcal{A}(\mathsf{pp})\right]$$

Moreover, we require a vector commitment to be *concise* in the sense that the size of the commitment string C and the outputs of $\mathsf{VC.Open}$ are both independent of q.

Vector commitments can also be required to be *hiding*. Informally, a vector commitment is hiding if an adversary cannot distinguish whether a commitment was created to a sequence $(m_1, \ldots, m_q)$ or to $(m'_1, \ldots, m'_q)$, even after seeing some openings (at positions i where the two sequences agree). We observe, however, that hiding is not a critical property in the realization of vector commitments. Indeed, any construction of vector commitments which does not satisfy hiding, can be easily fixed by composing it with a standard commitment scheme, i.e., first commit to each message separately using a standard commitment scheme, and then apply the VC to the obtained sequence of commitments. Moreover, neither the applications considered in this paper nor that considered in [18] require the underlying VC to be hiding. For these reasons, in our constructions we will only focus on the realization of the position binding property. We leave a formal definition of hiding for the full version of the paper.

2.1 A Vector Commitment Based on CDH

Here we propose an implementation of concise vector commitments based on the CDH assumption in bilinear groups. Precisely, the security of the scheme reduces to the Square Computational Diffie-Hellman assumption. Roughly speaking, the Square-CDH assumption says that it is computationally infeasible to compute the value g^{a^2}, given $g, g^a \in \mathbb{G}$. This has been shown equivalent to the standard CDH assumption [22,1]. Our construction is reminiscent of the incremental hash function by Bellare and Micciancio [2], even if we develop new techniques for creating our proofs that open the commitment at a specific position.

$\mathsf{VC.KeyGen}(1^k, q)$ Let $\mathbb{G}, \mathbb{G}_T$ be two bilinear groups of prime order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $g \in \mathbb{G}$ be a random generator. Randomly choose $z_1, \ldots, z_q \xleftarrow{\$} \mathbb{Z}_p$. For all $i = 1, \ldots, q$ set: $h_i = g^{z_i}$. For all $i, j = 1, \ldots, q$, $i \neq j$ set $h_{i,j} = g^{z_i z_j}$.
Set $\mathsf{pp} = (g, \{h_i\}_{i\in[q]}, \{h_{i,j}\}_{i,j\in[q],i\neq j})$. The message space is $\mathcal{M} = \mathbb{Z}_p$. [5]

$\mathsf{VC.Com_{pp}}(m_1, \ldots, m_q)$ Compute $C = h_1^{m_1} h_2^{m_2} \cdots h_q^{m_q}$ and output C and the auxiliary information $\mathsf{aux} = (m_1, \ldots, m_q)$.

$\mathsf{VC.Open_{pp}}(m_i, i, \mathsf{aux})$ Compute

$$\Lambda_i = \prod_{j=1, j\neq i}^{q} h_{i,j}^{m_j} = \left(\prod_{j=1, j\neq i}^{q} h_j^{m_j} \right)^{z_i}$$

$\mathsf{VC.Ver_{pp}}(C, m_i, i, \Lambda_i)$ If $e(C/h_i^{m_i}, h_i) = e(\Lambda_i, g)$ then output 1. Otherwise output 0.

$\mathsf{VC.Update_{pp}}(C, m, m', i)$ Compute the updated commitment $C' = C \cdot h_i^{m'-m}$. Finally output C' and $U = (m, m', i)$.

$\mathsf{VC.ProofUpdate_{pp}}(C, \Lambda_j, m', U)$ A client who owns a proof Λ_j, that is valid w.r.t. to C for some message at position j, can use the update information $U = (m, m', i)$ to compute the updated commitment C' and produce a new proof Λ'_j which will be valid w.r.t. C'. We distinguish two cases:

1. $i \neq j$. Compute the updated commitment $C' = C \cdot h_i^{m'-m}$ while the updated proof is $\Lambda'_j = \Lambda_j \cdot (h_i^{m'-m})^{z_j} = \Lambda_j \cdot h_{j,i}^{m'-m}$.
2. $i = j$. Compute the updated commitment as $C' = C \cdot h_i^{m'-m}$ while the updated proof remains the same Λ_i.

The correctness of the scheme can be easily verified by inspection. We prove its security via the following theorem whose proof appears in the full version.

Theorem 3. *If the CDH assumption holds, then the scheme defined above is a concise vector commitment.*

[5] The scheme can be easily extended to support arbitrary messages in $\{0,1\}^*$ by using a collision-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p$.

Efficiency and Optimizations. A drawback of our scheme is that the size of the public parameters pp is $O(q^2)$. This can be significant in those applications where the vector commitment is used with large datasets (e.g., verifiable databases and accumulators). However, we first notice that the verifier does not need the elements $h_{i,j}$. Furthermore, our construction can be easily optimized in such a way that the verifier does not need to store all the elements h_i of pp. The optimization works as follows. Who runs the setup signs each pair (i, h_i), includes the resulting signatures σ_i in the public parameters given to the committer pp, and publishes the signature's verification key. Next, the committer includes σ_i, h_i in the proof of an element at position i. This way the verifier can store only g and the verification key of the signature scheme. Later, each time it runs the verification of the vector commitment it has to check the validity of h_i by checking that σ_i is a valid signature on (i, h_i).

2.2 A Vector Commitment Based on RSA

Here we propose a realization of vector commitments from the RSA assumption.

$\mathsf{VC.KeyGen}(1^k, \ell, q)$ Randomly choose two $k/2$-bit primes p_1, p_2, set $N = p_1 p_2$, and then choose q $(\ell+1)$-bit primes $e_1, \ldots, e_q$ that do not divide $\phi(N)$. For $i = 1$ to q set $S_i = a^{\prod_{j=1, j\neq i}^{q} e_j}$. The public parameters pp are $(N, a, S_1, \ldots, S_q, e_1, \ldots, e_q)$. The message space is $\mathcal{M} = \{0,1\}^\ell$. [6]

$\mathsf{VC.Com_{pp}}(m_1, \ldots, m_q)$ Compute $C \leftarrow S_1^{m_1} \cdots S_q^{m_q}$ and output C and the auxiliary information $\mathsf{aux} = (m_1, \ldots, m_q)$.

$\mathsf{VC.Open_{pp}}(m, i, \mathsf{aux})$, Compute

$$\Lambda_i \leftarrow \sqrt[e_i]{\prod_{j=1, j\neq i}^{q} S_j^{m_j}} \bmod N$$

Notice that knowledge of pp allows to compute Λ_i efficiently *without* the factorization of N.

$\mathsf{VC.Ver_{pp}}(C, m, i, \Lambda_i)$ The verification algorithm returns 1 if $m \in \mathcal{M}$ and $C = S_i^m \Lambda_i^{e_i} \bmod N$ Otherwise it returns 0.

$\mathsf{VC.Update_{pp}}(C, m, m', i)$ Compute the updated commitment $C' = C \cdot S_i^{m'-m}$. Finally output C' and $U = (m, m', i)$.

$\mathsf{VC.ProofUpdate_{pp}}(C, \Lambda_j, m', i, U)$ A client who owns a proof Λ_j, that is valid w.r.t. to C for some message at position j, can use the update information U to compute the updated commitment C' and to produce a new proof Λ'_j which will be valid w.r.t. C'. We distinguish two cases:

1. $i \neq j$. Compute the updated commitment as $C' = C S_i^{m'-m}$ while the updated proof is $\Lambda'_j = \Lambda_j \sqrt[e_j]{S_i^{m'-m}}$ (notice that such e_j-th root can be efficiently computed using the elements in the public key).

[6] As in the CDH case, also this scheme can be extended to support arbitrary messages by using a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^\ell$.

2. $i = j$. Compute the updated commitment $C' = C \cdot S_i^{m'-m}$ while the updated proof remains the same Λ_i.

In order for the verification process to be correct, notice that one should also verify (only once) the validity of the public key by checking that the S_i's are correctly generated with respect to a and the exponents $e_1, \ldots, e_q$.

The correctness of the scheme can be easily verified by inspection. We prove its security via the following theorem.

Theorem 4. *If the RSA assumption holds, then the scheme defined above is a concise vector commitment.*

An optimization similar to the one suggested for CDH construction in the previous section applies to this scheme as well, and thus allows the verifier to store only a constant number of elements of the public parameters.

Achieving Constant-Size Public Parameters. In the full version of this work we show a variant of this RSA scheme that achieves *constant-size* public parameters. Very roughly, to do this we borrow some techniques from [15] that introduce a "special" pseudorandom function f that generates prime numbers, and we use such f to compute each prime e_i as $f(i)$. This new scheme is computationally less efficient compared to the other RSA and CDH constructions. Though, it shows that vector commitments with constant-size public parameters exist.

3 Verifiable Databases with Efficient Updates from Vector Commitments

In this section we show that vector commitments allow to build a verifiable database scheme. This notion has been formalized very recently by Benabbas, Gennaro and Vahlis [3]. Intuitively, a verifiable database allows a weak client to outsource the storage of a large database D to a server in such a way that the client can later retrieve the database records from the server and be convinced that the records have not been tampered with. In particular, since the main application is in the context of cloud computing services for storage outsourcing, it is crucial that the resources invested by the client after transmitting the database (e.g., to retrieve and update the records) must be independent of the database's size. While a solution for the static case in which the database is not updated can be obtained using standard techniques (e.g., digital signatures), the setting in which the client can update the values of the database records need different ideas.

Here we describe a solution based on our notion of Vector Commitments. Our construction, when instantiated with our CDH-based VC, allows for an efficient scheme, yet it is based on a standard constant-size assumption such as Computation Diffie-Hellman in bilinear groups. Furthermore, our scheme allows for public verifiability, that was not supported by the scheme in [3].

We begin by recalling the definition of Verifiable Databases. Our definition closely follows that in [3] except for some changes that we introduce because we

consider public verifiability. We denote a database D as a set of tuples (x, v_x) in some appropriate domain, where x is the key, and v_x is the corresponding value. We denote this by writing $D(x) = v_x$. In our case we consider keys that are integers in the interval $\{1, \ldots, q\}$, where $q = poly(k)$, whereas the DB values can be arbitrary strings $v \in \{0,1\}^*$.

A Verifiable Database scheme VDB is defined by the following algorithms:

$\mathsf{VDB.Setup}(1^k, D)$**.** On input the security parameter k and a database D, the setup algorithm is run by the client to generate a secret key SK that is kept private by the client, a database encoding S that is given to the server, and a public key PK that is distributed to all users (including the client itself) who wish to verify the proofs.

$\mathsf{VDB.Query}(\mathsf{PK}, S, x)$**.** On input a database key x, the query processing algorithm is run by the server, and returns a pair $\tau = (v, \pi)$.

$\mathsf{VDB.Verify}(\mathsf{PK}, x, \tau)$**.** The public verification algorithm outputs a value v if τ verifies correctly w.r.t. x (i.e., $D(x) = v$), and an error $\perp$ otherwise.

$\mathsf{VDB.ClientUpdate}(\mathsf{SK}, x, v')$**.** The client update algorithm is used by the client to change the value of the database record with key x, and it outputs a value t'_x and an updated public key PK'.

$\mathsf{VDB.ServerUpdate}(\mathsf{PK}, S, x, t'_x)$**.** The server update algorithm is run by the server to update the database according to the value t'_x produced by the client.

Before defining the notion of security we remark that a crucial requirement is that the size of the information stored by the client as well as the time needed to compute verifications and updates must be independent of the size $|D|$ of the database.

Roughly speaking, a Verifiable Database is secure if the server cannot convince users about the validity of false statements, i.e., that $D(x) = v$ where v is not the value v_x that the client wrote in the record with key x. We defer the interested reader to [3] and our full version for a more precise definition.

3.1 A Verifiable Database Scheme from Vector Commitments

Now we show how to build a verifiable database scheme VDB from a vector commitment VC. The construction follows.

$\mathsf{VDB.Setup}(1^k, D)$**.** Let $D = \{(i, v_i)\}_{i=1}^q$. Run $\mathsf{pp} \xleftarrow{\$} \mathsf{VC.KeyGen}(1^k, q)$. Compute $(C, \mathsf{aux}) \leftarrow \mathsf{VC.Com}_{\mathsf{pp}}(v_1, \ldots, v_q)$ and set $\mathsf{PK} = (\mathsf{pp}, C)$, $S = (\mathsf{pp}, \mathsf{aux}, D)$, $\mathsf{SK} = \perp$.

$\mathsf{VDB.Query}(\mathsf{PK}, S, x)$**.** Let $v_x = D(x)$. Compute $\Lambda_x \leftarrow \mathsf{VC.Open}_{\mathsf{pp}}(v_x, x, \mathsf{aux})$ and return $\tau = (v_x, \Lambda_x)$.

$\mathsf{VDB.Verify}(\mathsf{PK}, x, \tau)$**.** Parse τ as (v_x, Λ_x). If $\mathsf{VC.Ver}_{\mathsf{pp}}(C, x, v_x, \Lambda_x) = 1$, then return v_x. Otherwise return $\perp$.

$\mathsf{VDB.ClientUpdate}(\mathsf{SK}, x, v'_x)$**.** To update the record with key x, the client first retrieves the record x from the server (i.e., it asks the server for $\tau \leftarrow \mathsf{VDB.Query}(\mathsf{PK}, S, x)$ and checks that $\mathsf{VDB.Verify}(\mathsf{PK}, x, \tau) = v_x \neq \perp$). Then, it computes $(C', U) \xleftarrow{\$} \mathsf{VC.Update}_{\mathsf{pp}}(C, v_x, v'_x, x)$ and outputs $\mathsf{PK}' = (\mathsf{pp}, C')$ and $t'_x = (\mathsf{PK}', v'_x, U)$.

$\mathsf{VDB.ServerUpdate}(\mathsf{pk}, S, x, t'_x)$**.** Let $t'_x = (\mathsf{PK}', v'_x, U)$. The server writes v'_x in the database record with key x and adds the update information U to $\mathtt{aux}$ in S.

The security of our scheme follows from the following theorem whose proof appears in the full version.

Theorem 5. *If* VC *is a vector commitment, then the Verifiable Database scheme described above is secure.*

A NOTE ON THE SIZE OF THE PUBLIC KEY. If one looks at the concrete Verifiable Database scheme resulting by instantiating the vector commitment with one of our constructions in sections 2.1 and 2.2 a problem arises. In VDBs the public key must have size independent of the DB size, but this happens not to be the case in our CDH and RSA constructions where the public parameters pp depend on q. To solve this issue we thus require this transform to use vector commitments with constant size parameters. Concretely, we can use the variant of our RSA construction that has this property, or, for a better efficiency, our CDH/RSA constructions in Section 2 with the respective optimizations that enable the verifier to store only a constant number of elements of pp. A detailed description of this optimization was given in the previous section.

4 (Updatable) Zero-Knowledge Elementary Databases from Vector Commitments

In this section we show that Vector Commitments can be used to build Zero-Knowledge Elementary Databases (ZK-EDBs). In particular, following the approach of Catalano, Fiore and Messina [10], we can solve the open problem of building compact ZK-EDBs based on standard constant-size assumptions. Furthermore, in the next section we will show that the same approach can be extended to build *Updatable* ZK-EDBs, thus allowing for the first compact construction of this primitive. Since the updatable case is more interesting in practice, we believe that this can be a significant improvement.

ZERO-KNOWLEDGE ELEMENTARY DATABASES. We first recall the notion of Zero-Knowledge Elementary Databases. Let D be a database and $[D]$ be the set of all the keys in D. We assume that $[D]$ is a proper subset of $\{0,1\}^*$. If $x \in [D]$, we denote with $y = D(x)$ its associated value in the database D. If $x \notin [D]$ we let $D(x) = \perp$. A Zero Knowledge (Elementary) Database system is formally defined by a tuple of algorithms $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Prove}, \mathsf{V})$ that work as follows:

- $\mathsf{Setup}(1^k)$ takes as input the security parameter k and generates a common reference string CRS.
- $\mathsf{Commit}(CRS, D)$, the *committer* algorithm, takes as input a database D and the common reference string CRS and outputs a public key ZPK and a secret key ZSK.

- $\mathsf{Prove}(CRS, ZSK, x)$ On input the common reference string CRS, the secret key ZSK and an element x, the *prover* algorithm produces a proof π_x of either $D(x) = y$ or $D(x) = \perp$.
- $\mathsf{V}(CRS, ZPK, x, \pi_x)$ The *verifier* algorithm outputs y if $D(x) = y$, *out* if $D(x) = \perp$, and $\perp$ if the proof π_x is not valid.

We say that such a scheme is a Zero-Knowledge Elementary Database if it satisfies *completeness*, *soundness* and *zero-knowledge*. A precise description of such requirements can be found in [11]. Here we only explain them informally. In a nutshell, *completeness* requires that proofs generated by honest provers are correctly verified; *soundness* imposes that a dishonest prover cannot prove false statements about elements of the database; *zero-knowledge* guarantees that proofs do not reveal any information on the database (beyond their validity).

Towards Building Zero-Knowledge Elementary Databases. Chase *et al.* showed a general construction of ZK-EDB from a new primitive, that they called trapdoor mercurial commitment, and collision-resistant hash functions [12]. At a very high level, the idea of the construction is to build a Merkle tree in which each node is the mercurial commitment (instead of a hash) of its two children. This construction has been later generalized by Catalano *et al.* so as to work with q-ary trees instead of binary ones [10,11] in order to obtain more efficient schemes. This required the introduction of a new primitive called trapdoor q-mercurial commitments (qTMC), and it basically shows that the task of building ZK-EDBs can be reduced to that of building qTMCs. Therefore, in what follows we simply show how to build qTMCs using vector commitments. Then one can apply the generic methodology of Catalano *et al.* to obtain compact ZK-EDBs. We stress that in the construction of Catalano *et al.* the value q is the branching factor of the tree and is *not* related to the size of the database. Thus, even if vector commitments reveal q in the clear, this does not compromise the security of ZK-EDBs.

4.1 Trapdoor q-Mercurial Commitments from Vector Commitments and Mercurial Commitments

Here we show how to combine (concise) vector commitments and standard trapdoor commitment to obtain (concise) trapdoor qTMC. For lack of space, we defer the interested reader to [11] for the definitions of (trapdoor) mercurial commitments and trapdoor q-mercurial commitments.

Let $\mathsf{TMC} = (\mathsf{KeyGen}, \mathsf{HCom}, \mathsf{HOpen}, \mathsf{HVer}, \mathsf{SCom}, \mathsf{SOpen}, \mathsf{SVer}, \mathsf{Fake}, \mathsf{HEquiv}, \mathsf{SEquiv})$ be a trapdoor mercurial commitment and $\mathsf{VC} = (\mathsf{VC.KeyGen}, \mathsf{VC.Com}, \mathsf{VC.Open}, \mathsf{VC.Ver})$ be a vector commitment. We construct a trapdoor qTMC as follows:

$\mathsf{qKeyGen}(1^k)$**.** Run $\mathsf{pp} \xleftarrow{\$} \mathsf{VC.KeyGen}(1^k, q)$ and $(PK_{\mathsf{TMC}}, TK_{\mathsf{TMC}}) \xleftarrow{\$} \mathsf{KeyGen}(1^k)$ and set $pk = (\mathsf{pp}, PK_{\mathsf{TMC}})$ and $tk = TK_{\mathsf{TMC}}$.

$\mathsf{qHCom}_{pk}(m_1, \ldots, m_q)$**.** For $i = 1$ to q compute $(C_i, \mathsf{aux}^i_{\mathsf{TMC}}) \xleftarrow{\$} \mathsf{HCom}_{PK_{\mathsf{TMC}}}(m_i)$. Next, compute $(C, \mathsf{aux}_{\mathsf{VC}}) \leftarrow \mathsf{VC.Com}_{\mathsf{pp}}(C_1, \ldots C_q)$. The output is C and the auxiliary information is $\mathsf{aux} = (\mathsf{aux}_{\mathsf{VC}}, m_1, C_1, \mathsf{aux}^1_{\mathsf{TMC}}, \ldots, m_q, C_q, \mathsf{aux}^q_{\mathsf{TMC}})$.

$\mathsf{qHOpen}_{pk}(m_i, i, \mathsf{aux})$**.** Extract $(m_i, C_i, \mathsf{aux}^i_{\mathsf{TMC}})$ from aux and set $\Lambda_i \leftarrow \mathsf{VC.Open}_{\mathsf{pp}}(C_i, i, \mathsf{aux}_{\mathsf{VC}})$. The opening information is $\tau_i = (\Lambda_i, C_i, \pi_i)$ where π_i is the output of $\mathsf{HOpen}_{PK_{\mathsf{TMC}}}(m_i, \mathsf{aux}^i_{\mathsf{TMC}})$.

$\mathsf{qHVer}_{pk}(C, m_i, i, \tau_i)$**.** Parse τ_i as (Λ_i, C_i, π_i). The hard verification algorithm returns 1 if and only if both $\mathsf{HVer}_{PK_{\mathsf{TMC}}}(C_i, m_i, \pi_i)$ and $\mathsf{VC.Ver}_{\mathsf{pp}}(C, C_i, i, \Lambda_i)$ return 1.

$\mathsf{qSCom}_{pk}()$**.** For $i = 1$ to q compute $(C_i, \mathsf{aux}^i_{\mathsf{TMC}}) \leftarrow \mathsf{SCom}_{PK_{\mathsf{TMC}}}()$. Next, compute $(C, \mathsf{aux}_{\mathsf{VC}}) \leftarrow \mathsf{VC.Com}_{\mathsf{pp}}(C_1, \ldots, C_q)$. The output is C and the auxiliary information is $\mathsf{aux} = (\mathsf{aux}_{\mathsf{VC}}, m_1, C_1, \mathsf{aux}^1_{\mathsf{TMC}}, \ldots, m_q, C_q, \mathsf{aux}^q_{\mathsf{TMC}})$.

$\mathsf{qSOpen}_{pk}(m_i, i, \mathsf{flag}, \mathsf{aux})$**.** Extract $(m_i, C_i, \mathsf{aux}^i_{\mathsf{TMC}})$ from aux and set $\Lambda_i \leftarrow \mathsf{VC.Open}_{\mathsf{pp}}(C_i, i, \mathsf{aux}_{\mathsf{VC}})$. The opening information is $\tau_i = (\Lambda_i, C_i, \pi_i)$ where π_i is the output of $\mathsf{SOpen}_{PK_{\mathsf{TMC}}}(m_i, \mathsf{aux}^i_{\mathsf{TMC}})$.

$\mathsf{qSVer}_{pk}(C, m, i, \tau_i)$**.** Parse τ_i as (Λ_i, C_i, π_i). The soft verification algorithm returns 1 if and only if both $\mathsf{SVer}_{PK_{\mathsf{TMC}}}(C_i, m_i, \pi_i)$ and $\mathsf{VC.Ver}_{\mathsf{pp}}(C, C_i, i, \Lambda_i)$ return 1.

$\mathsf{qFake}_{pk,tk}()$**.** This is the same as the qSCom algorithm.

$\mathsf{qHEquiv}_{pk,tk}(m, i, \mathsf{aux})$**.** Extract $(C_i, \mathsf{aux}^i_{\mathsf{TMC}})$ (for all $i = 1$ to q) and set $\Lambda_i \leftarrow \mathsf{VC.Open}_{\mathsf{pp}}(C_i, i, \mathsf{aux}_{\mathsf{VC}})$. The hard equivocation is $\tau_i = (\Lambda_i, C_i, \pi_i)$ where π_i is the output of $\mathsf{HEquiv}_{PK_{\mathsf{TMC}}, tk_{\mathsf{TMC}}}(m, \mathsf{aux}^i_{\mathsf{TMC}})$

$\mathsf{qSEquiv}_{pk,tk}(m, i, \mathsf{aux})$**.** Extract $(C_i, \mathsf{aux}^i_{\mathsf{TMC}})$ (for all $i = 1$ to q) and set $\Lambda_i \leftarrow \mathsf{VC.Open}_{\mathsf{pp}}(C_i, i, \mathsf{aux}_{\mathsf{VC}})$. The soft equivocation is $\tau_i = (\Lambda_i, C_i, \pi_i)$ where π_i is the output of $\mathsf{SEquiv}_{PK_{\mathsf{TMC}}, tk_{\mathsf{TMC}}}(m, \mathsf{aux}^i_{\mathsf{TMC}})$

The correctness of the scheme easily follows from the correctness of the underlying building blocks. Its security follows from the following theorem (whose proof appears in the full version).

Theorem 6. *Assuming that* TMC *is a trapdoor mercurial commitment and* VC *is a vector commitment, then the scheme defined above is a trapdoor* q*-mercurial commitment.*

On the efficiency of the CDH instantiation. By instantiating the above scheme with our vector commitment based on CDH (and with the discrete log based TMC from [23]), one gets a qTMC based on CDH whose efficiency is roughly the same as that of the scheme in [19] based on q-DHE. For the sake of a fair comparison we notice that in our construction the reduction to CDH is not tight (due to the non-tight reduction from Square-DH to CDH [22]), and our scheme suffers from public parameters of size $O(q^2)$. In contrast, the scheme by Libert and Yung has a tight reduction to the q-DHE problem and achieves public parameters of size $O(q)$. However, we think that the CDH and the q-DHE assumptions are not easily comparable, especially given that the latter is defined with instances of size $O(q)$, where q is known to degrade the quality of the assumption (see [13,28] for some attacks). Furthermore, a more careful look shows that in our scheme the verifier *does not* need to store this many elements. This is because the $h_{i,j}$'s are not required for verification. Thus, from the verifier side, the space required is actually only $O(q)$. In the application of ZK-EDBs

such an optimization reflects on the size of the common reference string. More precisely, while the server still needs to store a CRS of size $O(q^2)$, the client is required to keep in memory only a portion of the CRS of size of $O(q)$ (thus allowing for comparable client-side requirements with respect to [19]).

4.2 Updatable ZK-EDBs with Short Proofs and Updates

In most practical applications databases are frequently updated. The constructions of ZK-EDBs described so far do not deal with this and the only way of updating a ZK-EDB is to actually recompute the entire commitment from scratch every time the database changes. This is highly undesirable as previously issued proofs can no longer be valid.

This problem was studied by Liskov in [20] where he showed how to build Updatable ZK EDB by appropriately modifying the basic approach of combining Merkle trees and mercurial commitments. In particular, rather than using standard mercurial commitments, Liskov employed a new primitive called *updatable mercurial commitment*. Very informally, updatable mercurial commitments are like standard ones with the additional feature that they allow for two update procedures. The committer can change the message inside the commitment and produce: a new commitment and an update information. These can later be used by verifiers to update their commitments and the associated proofs (that will be valid w.r.t. the new commitment). Therefore whenever the prover changes some value $D(x)$ in the database, first he has to update the commitment in the leaf labeled by x and then he updates all the commitments in the path from x to the root. The new database commitment is the updated commitment in the root node, while the database update information contains the update informations for all the nodes involved in the update.

A natural question raised by the methodology above is whether the zero-knowledge property remains preserved after an update occurs, as the latter reveals information about the updated key. To solve this issue Liskov proposed to "mask" the label of each key x (i.e. the paths in the tree) using a pseudorandom pseudonym $N(x)$ and he relaxed the zero-knowledge property to hold w.r.t. $N()$. Further details can be found in [20].

In [20] two constructions of updatable mercurial commitments are given. One is generic and uses both standard and mercurial commitments. The other one is direct and builds from the DL-based mercurial commitment of [23].

Our Result. We introduce the notion of *updatable* q-mercurial commitments, and then we show that these can be built from vector commitments and updatable mercurial commitments. Next, by applying the methodology of Liskov sketched above, adapted with the compact construction of Catalano *et al.* [10], we can build the first *compact* Updatable ZK-EDB. It is interesting to observe that by using the compact construction the resulting ZK-EDB improves over the scheme in [20] both in terms of proofs' size and length of the update information, as these both grow linearly in the height of the tree $log_q 2^k$ (which is strictly less than k for $q > 2$). For lack of space, we defer the interested reader to [20] for formal definitions of (basic) updatable mercurial commitments.

Updatable q-Mercurial Commitments. An updatable q-(trapdoor) mercurial commitment is defined like a qTMC with the following two additional algorithms:

$\mathsf{qUpdate}_{pk}(C, \mathsf{aux}, m', i)$**.** This algorithm is run by the committer who produced C (and holds the corresponding aux) and wants to change the i-th committed message with m'. The algorithm takes as input m' and the position i and outputs a new commitment C' and an update information U.

$\mathsf{qProofUpdate}_{pk}(C, \tau_j, m', i, U)$**.** This algorithm can be run by any user who holds a proof τ_j for some message at position j in C and allows the user to produce a new proof τ'_j (and the updated commitment C') which will be valid w.r.t. C' that contains m' as the new message at position i. The value U contains the update information which is needed to compute such values.

The q-mercurial binding property is defined as usual, namely for any PPT adversary it is computationally infeasible to open a commitment (even an updated one) to two different messages at the same position. Hiding and equivocations for updatable qTMCs easily follow from those of updatable mercurial commitments by extending them to the case of sequences of q messages.

Updatable q-mercurial Commitments from Vector Commitments. In this section we show that an updatable qTMC can be built using an updatable (trapdoor) mercurial commitment uTMC and a vector commitment VC. The construction is essentially the same as that given in Section 4.1 augmented with the following update algorithms:

$\mathsf{qUpdate}_{pk}(C, \mathsf{aux}, m', i)$**.** Parse aux as $(m_1, C_1, \mathsf{aux}^1_{\mathsf{uTMC}}, \ldots, m_q, C_q, \mathsf{aux}^q_{\mathsf{uTMC}}, \mathsf{aux}_{\mathsf{VC}})$, extract $(C_i, m_i, \mathsf{aux}^i_{\mathsf{uTMC}})$ from it and run $(C'_i, U_{\mathsf{uTMC}}) \leftarrow \mathsf{Update}_{pk_{\mathsf{uTMC}}}(C_i, \mathsf{aux}^i_{\mathsf{uTMC}}, m')$. Then update the vector commitment $(C', U') \leftarrow \mathsf{VC.Update}_{\mathsf{pp}}(C, C_i, C'_i, i)$ and output C' and $U = (U', C_i, C'_i, i, U^i_{\mathsf{uTMC}})$.

$\mathsf{qProofUpdate}_{pk}(C, \tau_j, m', i, U)$**.** The client who holds a proof $\tau_j = (\Lambda_j, C_j, \pi_j)$ that is valid w.r.t. to C for some message at position j, can use the update information U to compute the updated commitment C' and produce a new proof τ'_j which will be valid w.r.t. the new C'. We distinguish two cases:

1. $i \neq j$. Compute $(C', \Lambda'_j) \leftarrow \mathsf{VC.ProofUpdate}_{\mathsf{pp}}(C, \Lambda_j, C'_i, i, U')$ and output the updated commitment C' and the updated proof $\tau_j = (\Lambda'_j, C_j, \pi_j)$.
2. $i = j$. Let $(C'_i, \pi'_i) \leftarrow \mathsf{ProofUpdate}_{pk_{\mathsf{uTMC}}}(C_i, m', \pi_i, U^i_{\mathsf{uTMC}})$. Compute the updated commitment as $(C', \Lambda'_i) \leftarrow \mathsf{VC.ProofUpdate}_{\mathsf{pp}}(C, \Lambda_i, C'_i, i, U')$ and the updated proof is $\tau'_i = (\Lambda'_i, C'_i, \pi'_i)$.

Theorem 7. *If* uTMC *is an updatable trapdoor mercurial commitment and* VC *is an updatable vector commitment, then the scheme given above is an updatable concise trapdoor q-mercurial commitment.*

The proof is very similar to that of Theorem 6 and is omitted here.

References

1. Bao, F., Deng, R.H., Zhu, H.: Variations of Diffie-Hellman Problem. In: Qing, S., Gollmann, D., Zhou, J. (eds.) ICICS 2003. LNCS, vol. 2836, pp. 301–312. Springer, Heidelberg (2003)
2. Bellare, M., Micciancio, D.: A New Paradigm for Collision-Free Hashing: Incrementality at Reduced Cost. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 163–192. Springer, Heidelberg (1997)
3. 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)
4. Benaloh, J.C., de Mare, M.: One-Way Accumulators: A Decentralized Alternative to Digital Signatures (Extended Abstract). In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 274–285. Springer, Heidelberg (1994)
5. 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)
6. Camenisch, J., Kohlweiss, M., Soriente, C.: An Accumulator Based on Bilinear Maps and Efficient Revocation for Anonymous Credentials. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 481–500. Springer, Heidelberg (2009)
7. Camenisch, J., Lysyanskaya, A.: Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002)
8. Catalano, D., Dodis, Y., Visconti, I.: Mercurial Commitments: Minimal Assumptions and Efficient Constructions. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 120–144. Springer, Heidelberg (2006)
9. Catalano, D., Fiore, D.: Vector commitments and their applications. Cryptology ePrint Archive (2011), `http://eprint.iacr.org/2011/495`
10. Catalano, D., Fiore, D., Messina, M.: Zero-Knowledge Sets with Short Proofs. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 433–450. Springer, Heidelberg (2008)
11. Catalano, D., Di Raimondo, M., Fiore, D., Messina, M.: Zero-knowledge sets with short proofs. IEEE Transactions on Information Theory 57(4), 2488–2502 (2011)
12. Chase, M., Healy, A., Lysyanskaya, A., Malkin, T., Reyzin, L.: Mercurial Commitments with Applications to Zero-Knowledge Sets. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 422–439. Springer, Heidelberg (2005)
13. Cheon, J.H.: Security Analysis of the Strong Diffie-Hellman Problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 1–11. Springer, Heidelberg (2006)
14. Gennaro, R., Micali, S.: Independent Zero-Knowledge Sets. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 34–45. Springer, Heidelberg (2006)
15. Hohenberger, S., Waters, B.: Short and Stateless Signatures from the RSA Assumption. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 654–670. Springer, Heidelberg (2009)
16. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-Size Commitments to Polynomials and Their Applications. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 177–194. Springer, Heidelberg (2010)
17. Li, J., Li, N., Xue, R.: Universal Accumulators with Efficient Nonmembership Proofs. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 253–269. Springer, Heidelberg (2007)

18. Libert, B., Peters, T., Yung, M.: Group Signatures with Almost-for-Free Revocation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 571–589. Springer, Heidelberg (2012)
19. Libert, B., Yung, M.: Concise Mercurial Vector Commitments and Independent Zero-Knowledge Sets with Short Proofs. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 499–517. Springer, Heidelberg (2010)
20. Liskov, M.: Updatable Zero-Knowledge Databases. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 174–198. Springer, Heidelberg (2005)
21. Martel, C.U., Nuckolls, G., Devanbu, P.T., Gertz, M., Kwong, A., Stubblebine, S.G.: A general model for authenticated data structures. Algorithmica 39(1), 21–41 (2004)
22. Maurer, U.M., Wolf, S.: Diffie-Hellman Oracles. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 268–282. Springer, Heidelberg (1996)
23. Micali, S., Rabin, M.O., Kilian, J.: Zero-knowledge sets. In: 44th FOCS, pp. 80–91. IEEE Computer Society Press (October 2003)
24. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: 40th FOCS, pp. 120–130. IEEE Computer Society Press (October 1999)
25. Naor, M., Nissim, K.: Certificate revocation and certificate update. In: Proceedings of the 7th Conference on USENIX Security Symposium, vol. 7, p. 17 (1998)
26. Nguyen, L.: Accumulators from Bilinear Pairings and Applications. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 275–292. Springer, Heidelberg (2005)
27. Papamanthou, C., Tamassia, R.: Time and Space Efficient Algorithms for Two-Party Authenticated Data Structures. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 1–15. Springer, Heidelberg (2007)
28. Sakemi, Y., Hanaoka, G., Izu, T., Takenaka, M., Yasuda, M.: Solving a Discrete Logarithm Problem with Auxiliary Input on a 160-Bit Elliptic Curve. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 595–608. Springer, Heidelberg (2012)
29. Stefanov, E., van Dijk, M., Oprea, A., Juels, A.: Iris: A scalable cloud file system with efficient integrity checks. Cryptology ePrint Archive, Report 2011/585 (2011), `http://eprint.iacr.org/`
30. Tamassia, R., Triandopoulos, N.: Certification and authentication of data structures. In: Alberto Mendelzon Workshop on Foundations of Data Management (2010)

Efficient, Adaptively Secure, and Composable Oblivious Transfer with a Single, Global CRS

Seung Geol Choi[1,⋆], Jonathan Katz[2,⋆⋆], Hoeteck Wee[3,⋆⋆⋆], and Hong-Sheng Zhou[2,†]

[1] Columbia University
sgchoi@cs.columbia.edu
[2] University of Maryland
{jkatz,hszhou}@cs.umd.edu
[3] George Washington University
hoeteck@alum.mit.edu

Abstract. We present a general framework for efficient, universally composable oblivious transfer (OT) protocols in which a *single*, global, common reference string (CRS) can be used for multiple invocations of oblivious transfer by arbitrary pairs of parties. In addition:

- Our framework is round-efficient. E.g., under the DLIN or SXDH assumptions we achieve round-optimal protocols with static security, or 3-round protocols with adaptive security (assuming erasure).
- Our resulting protocols are more efficient than any known previously, and in particular yield protocols for string OT using $O(1)$ exponentiations and communicating $O(1)$ group elements.

Our result improves on that of Peikert et al. (Crypto 2008), which uses a CRS whose length depends on the number of parties in the network and achieves only static security. Compared to Garay et al. (Crypto 2009), we achieve adaptive security with better round complexity and efficiency.

1 Introduction

In this work we study the construction of efficient protocols for universally composable (UC) [5] oblivious transfer (OT). Our work is motivated by the fact that, although UC *commitments* are complete for UC multiparty

⋆ This work was done in part at the University of Maryland, and was supported in part by the Intelligence Advanced Research Project Activity (IARPA) via DoI/NBC contract #D11PC20194. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation herein. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies or endorsements, expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

⋆⋆ This work is supported in part by DARPA and by NSF awards #0447075, #1111599, and #1223623.

⋆⋆⋆ Supported by NSF CAREER Award CNS-1237429.

† Supported by an NSF CI postdoctoral fellowship.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 73–88, 2013.

computation [9], the most efficient multiparty computation protocols (e.g., [29,28]) rely on universally composable OT as a building block. Relative to UC commitments (see [27,16] and references therein), however, universally composable OT has received less attention.

There is a long series of work on efficient OT protocols in the stand-alone setting (e.g., [30,1,21,25]). Lindell [26] (also [23, Appendix A]) gave a generic transformation from static security to adaptive security (assuming erasure) that applied in the semi-honest setting and the stand-alone malicious setting, but not in the UC setting.

Constructions of UC oblivious transfer from general assumptions were given in [9]; these constructions are relatively inefficient. Garay, MacKenzie, and Yang [17] constructed a constant-round protocol for *committed* OT under the DDH and strong RSA assumptions. Their protocol yields *bit* OT rather than *string* OT, so results in protocols for string OT with complexity linear in the length of the sender's inputs. Jarecki and Shmatikov show a four-round protocol for committed string OT under the decisional composite residuosity (DCR) assumption [24]. A round-optimal OT protocol appears in [22].

The most efficient known protocol for UC oblivious transfer is that of Peikert et al. [33]. Their work, however, has several disadvantages. First, it requires an independent common reference string[1] (CRS) for *every party* in the network or, equivalently, a single CRS of length linear in the number of parties. (Any pair of parties can then run the protocol of Peikert et al. using the CRS of the receiver.) Their protocols also only achieve security against a *static* adversary who decides which parties to corrupt before the protocol begins (and even before the CRS is chosen). They do not handle an *adaptive* adversary who may choose which parties to corrupt during the course of the protocol execution.

Garay et al. [18] constructed efficient UC oblivious-transfer protocols that address both the above-mentioned drawbacks. In their constructions, the parties run a coin-tossing protocol whose outcome is then used as a common random string for an OT protocol. This approach is not entirely satisfactory. First, it increases the overall computation, communication, and round complexity; second, it can (in general) only be instantiated with OT protocols that work in the common *random* string model rather than the more general common *reference* string model. Choi et al. [11,10] showed other approaches for obtaining adaptively secure, constant-round UC oblivious transfer.

1.1 Our Results

Here, we present a new framework for constructing UC oblivious-transfer protocols that require only a *single*, global CRS. We aim for efficient protocols having low round complexity, and incurring only *constant* computation and communication even when the sender's inputs are long strings. We are also interested in achieving *adaptive* security, under the assumption that parties erase

[1] *Some* form of setup is known to be necessary for universally composable OT [7,8].

portions of their local state that are no longer needed. (Note, however, that the works of [11,18,10] do not make this assumption.)

Our framework is fairly general and can be instantiated from several assumptions. Specifically:

- We obtain efficient, *round-optimal* OT protocols with static security under the decisional linear (DLIN) [3] or symmetric external Diffie-Hellman (SXDH) assumptions [34,3]. These protocols can be modified to achieve adaptive security (assuming erasure) with one additional round and a slight increase in communication and computation.
- We obtain efficient, four-round OT protocols under the decisional Diffie-Hellman (DDH) or DCR [31] assumptions. Our basic constructions achieve static security, and we present variants that are secure against adaptive corruptions (assuming erasure) *without* any additional rounds, but with a slight increase in communication and computation.

We compare our constructions with previous work in Table 1[2]

Overview of Our Constructions. The starting point of our approach is the Halevi-Kalai construction [21] of 2-round OT based on smooth projective hashing. Their construction only achieves indistinguishability-based security (and not even stand-alone simulation-based security) against a malicious receiver. We show how to overcome this with the following modifications:

1. We require the receiver to commit to its input using CCA-secure encryption.
2. The receiver proves in zero knowledge that it is behaving consistently in the underlying OT protocol (with respect to the input it committed to).

A similar high-level approach was taken in [22], but using generic simulation-sound non-interactive zero knowledge [15]. Here, following recent constructions of efficient UC commitments [27,16], we rely instead on efficient zero-knowledge protocols that admit straight-line simulation in the CRS model. In particular, for our two-round OT protocols we instantiate the underlying zero-knowledge proofs using Groth-Sahai proofs [20], as in [16]. For our four-round OT protocols, we rely on Damgård's three-round zero-knowledge proof system [14].

Achieving Adaptive Security. To achieve adaptive security, we first modify our protocols so the final message is sent over an adaptively secure channel (cf. functionality $\mathcal{F}_{\text{SMT}}$ in [5]). The latter can be realized at low cost if erasure is assumed [2]. With this modification, security against adaptive corruption of the sender is achieved automatically by simply having the sender erase its local state at appropriate times. In our two-round protocols, security against adaptive corruption of the receiver is similarly achieved. For our 4-round protocols, we use techniques similar to those in [27,16]. Unlike this prior work, however, we do

[2] The numbers for the adaptively secure protocol of [33]+[18]+[27] in Table 1 are based on a preliminary version of [27], and could change once the author publishes the fix to a bug in the protocol.

Table 1. Efficient universally composable protocols for string OT. The number of parties is n. Communication complexity and CRS size are measured in terms of the number of group elements, with other values ignored. The numbers for [24] include the cost of the pre-processing stage.

Reference	Assumption	Rounds	Communication complexity	CRS size
[33]	DDH	2	6	n
[33]+[18]+[16]	DLIN	4	78	12
Protocol 1*	DLIN	2	54	12
[33]+[18]+[27]	DDH	6	38	7
Protocol 2	DDH	4	32	6
[24]	DCR	4	35 $(\mathbb{Z}_{N^2})$ + 16 $(\mathbb{Z}_N)$	10
Protocol 2	DCR	4	18 $(\mathbb{Z}_{N^2})$ + 7 $(\mathbb{Z}_N)$	12

Protocols with static security.

Reference	Assumption	Rounds	Communication complexity	CRS size
[33]+[18]+[16]	DLIN	4	83	12
Protocol 1*	DLIN	3	59	12
[33]+[18]+[27]	DDH	8	51	7
Protocol 2*	DDH	4	35	6
Protocol 2*	DCR	4	21 $(\mathbb{Z}_{N^2})$ + 7 $(\mathbb{Z}_N)$	12

Protocols with adaptive security (assuming erasure).

not introduce any additional overhead in communication or round complexity. (We incur a modest increase in computational cost.)

Organization. We review some preliminaries in Section 2. Our framework for 2-round OT with static security (resp., 3-round OT with adaptive security) is described in Section 3 Our framework for 4-round OT is given in Section 4. Due to space limitations, further details, proofs, and discussions about concrete instantiations have been deferred to the full version.

2 Preliminaries

We let λ be the security parameter. We let $\mathcal{F}_{\mathsf{MOT}}$ be the multi-session OT functionality [5], and $\mathcal{F}_{\mathsf{CRS}}^{\mathcal{P},\mathcal{D}}$ be the CRS functionality [6].

We use the standard notion of chosen-ciphertext security for labeled public-key encryption [4].

$\mathcal{HF} = \left\{h_k : \{0,1\}^* \to \{0,1\}^{\ell(\lambda)}\right\}_{k\in\{0,1\}^\lambda}$ is a family of collision-resistant hash functions if for any non-uniform PPT algorithm $\mathcal{A}$, it holds that

$$\Pr[k \leftarrow \{0,1\}^\lambda : \mathcal{A}(k) = (x_1, x_2) \text{ s.t. } x_1 \neq x_2 \text{ and } h_k(x_1) = h_k(x_2)] = \mathsf{negl}(\lambda).$$

2.1 Smooth Projective Hash Proof Systems

We recall the notion of a hard subset membership problem and smooth projective hashing defined by Cramer and Shoup [13], following the notation of [21]. A hash family $\mathcal{H}$ consists of the following PPT algorithms:

- The *parameter-generator* $\mathsf{HashPG}(1^\lambda) \to \text{PP}$. We assume that the security parameter λ can be inferred from PP. Let $\lambda(\text{PP})$ denote the security parameter corresponding to PP.
- A pair of disjoint sets Λ_{YES} and Λ_{NO} are associated to PP corresponding to YES and NO instances respectively. There exists a YES *instance-sampler* $\mathsf{SampYes}(\text{PP}) \to (x, w)$ where x is uniformly distributed over Λ_{YES} and w is the corresponding witness. There also exists a NO *instance-sampler* $\mathsf{SampNo}(\text{PP}) \to x'$ where x' is uniformly distributed over Λ_{NO}.
- The *hash-key generator* $\mathsf{HashKG}(\text{PP}) \to (\text{HK}, \text{PK})$. Here HK is the primary hashing key and PK is a projective key.
- The *primary hash algorithm* $\mathsf{Hash}(\text{HK}, x) \to y$ for all $x \in \Lambda_{\text{YES}} \cup \Lambda_{\text{NO}}$.
- The *projection hash algorithm* $\mathsf{pHash}(\text{PK}, x, w) \to y$ for all $(x, w) \leftarrow \mathsf{SampYes}(\text{PP})$.

We require that for all $\text{PP} \in \mathsf{support}(\mathsf{HashPG})$, every $(\text{HK}, \text{PK}) \leftarrow \mathsf{HashKG}(\text{PP})$, and every $(x, w) \leftarrow \mathsf{SampYes}(\text{PP})$, we have $\mathsf{pHash}(\text{PK}, x, w) = \mathsf{Hash}(\text{HK}, x)$.

Definition 1. $\mathcal{H} = (\mathsf{HashPG}, \mathsf{SampYes}, \mathsf{SampNo}, \mathsf{HashKG}, \mathsf{Hash}, \mathsf{pHash})$ *is a* smooth projective hash family *if*

Smoothness: *Let* $(\text{HK}, \text{PK}) \leftarrow \mathsf{HashKG}(\text{PP})$. *For all* $x \in \Lambda_{\text{NO}}$, *the distribution of* $\mathsf{Hash}(\text{HK}, x)$ *given* PK *is statistically close to uniform. That is, the statistical difference between the following two distributions is negligible in* $\lambda(\text{PP})$.

$$\{y \leftarrow \mathsf{Hash}(\text{HK}, x) : \ (\text{PK}, y, x)\} \stackrel{s}{\equiv} \{y \leftarrow \Gamma : \ (\text{PK}, y, x)\}$$

where Γ *denotes the set of possible hash values with parameter* PP.

Definition 2. *A smooth projective hash family* $\mathcal{H} = (\mathsf{HashPG}, \mathsf{SampYes}, \mathsf{SampNo}, \mathsf{HashKG}, \mathsf{Hash}, \mathsf{pHash})$ *is said to have a* hard subset membership *property if the following two ensembles are computationally indistinguishable:*

- $\left\{\text{PP} \leftarrow \mathsf{HashPG}(1^\lambda);\ (x, w) \leftarrow \mathsf{SampYes}(\text{PP}) :\ (\text{PP}, x)\right\}_{\lambda \in \mathbb{N}}$
- $\left\{\text{PP} \leftarrow \mathsf{HashPG}(1^\lambda);\ x \leftarrow \mathsf{SampNo}(\text{PP}) :\ (\text{PP}, x)\right\}_{\lambda \in \mathbb{N}}$.

2.2 Dual-Mode NIZK

Groth introduced non-interactive zero-knowledge (NIZK) proofs [19] that we call *dual-mode*. In such a proof system, a common reference string crs is generated in either a *soundness* mode or a *zero-knowledge* (ZK) mode; given crs, it is infeasible to determine the mode in which it was generated. When crs is generated in the soundness mode, the proof system is statistically sound. On the other hand, when crs is generated in the ZK mode, the simulation is perfect. Groth and Sahai [20] provide efficient dual-mode NIZK proofs for various equations in bilinear groups.

Definition 3. *A* non-interactive proof system *for a language $L \in \mathcal{NP}$ consists of three algorithms $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ where $\mathcal{K}$ is a CRS generation algorithm, $\mathcal{P}$ and $\mathcal{V}$ are a prover and a verifier algorithm respectively. The system is required to satisfy the following properties:*

Completeness: *For any λ, any $x \in L$, and any witness w for x, it holds that*

$$\Pr[\mathsf{crs} \leftarrow \mathcal{K}(1^\lambda);\ \pi \leftarrow \mathcal{P}(1^\lambda, \mathsf{crs}, x, w) :\ \mathcal{V}(1^\lambda, \mathsf{crs}, x, \pi) = 1] = 1.$$

Adaptive soundness: *For any λ and any adversary $\mathcal{A}$, it holds that*

$$\Pr[\mathsf{crs} \leftarrow \mathcal{K}(1^\lambda);\ (x, \pi) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}) :\ \mathcal{V}(1^\lambda, \mathsf{crs}, x, \pi) = 1\ \wedge x \notin L] = \mathsf{negl}(\lambda).$$

Definition 4. *A non-interactive proof system $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ for a language $L \in \mathcal{NP}$ is said to be* dual-mode NIZK *if there is a pair of efficient algorithms $(\mathcal{S}_1, \mathcal{S}_2)$ such that for any $\lambda \in \mathbb{N}$ and for all non-uniform polynomial time adversary $\mathcal{A}$, it holds the following:*

Indistinguishability of Modes:

$$\left| \Pr[\mathsf{crs} \leftarrow \mathcal{K}(1^\lambda) :\ \mathcal{A}(1^\lambda, \mathsf{crs}) = 1] - \Pr[(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\lambda) :\ \mathcal{A}(1^\lambda, \mathsf{crs}) = 1] \right| = \mathsf{negl}(\lambda).$$

Perfect Simulation in ZK Mode: *The following two probabilities are equal.*
- $\Pr[(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\lambda);\ (x, w) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}, \tau);\ \pi \leftarrow \mathcal{P}(1^\lambda, \mathsf{crs}, x, w) : \mathcal{A}(\pi) = 1]$
- $\Pr[(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\lambda);\ (x, w) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}, \tau);\ \pi \leftarrow \mathcal{S}_2(\tau, x) : \mathcal{A}(\pi) = 1]$

Here, $\mathcal{A}$ has to generate a pair (x, w) with w a witness for x.

2.3 Σ-Protocols

A Σ-protocol is a 3-round honest-verifier zero-knowledge protocol. We denote by (a, e, z) the messages exchanged between the prover $\mathcal{P}_\Sigma$ and the verifier $\mathcal{V}_\Sigma$. We say a transcript (a, e, z) is an *accepting transcript for x* if $\mathcal{V}_\Sigma$ would accept based on the values (x, a, e, z). We use the standard definitions of special soundness and special honest-verifier zero knowledge.

2.4 Equivocal Commitments

We define an equivocal commitment scheme as follows:

Definition 5. *Let $(\mathcal{K}_{com}, \mathsf{Com})$ be a non-interactive commitment scheme with CRS where $\mathcal{K}_{com}$ is a CRS generation algorithm, and* Com *is a commitment algorithm. The scheme is said to be* equivocal *if there exists a tuple of PPT algorithm $(\mathcal{S}_{com1}, \mathcal{S}_{com2}, \mathcal{S}_{com3})$ that satisfies the following properties:*

Computational Binding: *For any non-uniform polynomial time adversary $\mathcal{A}$ the following is negligible in λ:*

$$\Pr\left[\begin{array}{l}\mathsf{crs} \leftarrow \mathcal{K}_{com}(1^\lambda);\ (m, m', r, r') \leftarrow \mathcal{A}(\mathsf{crs}) : \\ \qquad m \neq m' \bigwedge \mathsf{Com}_{\mathsf{crs}}(m; r) = \mathsf{Com}_{\mathsf{crs}}(m'; r')\end{array}\right]$$

Indistinguishability of Modes:

$$\left\{\mathsf{crs}\leftarrow\mathcal{K}_{com}(1^\lambda):\ \mathsf{crs}\right\}_{\lambda\in\mathbb{N}} \overset{c}{\approx} \left\{(\mathsf{crs},t)\leftarrow\mathcal{S}_{com1}(1^\lambda):\ \mathsf{crs}\right\}_{\lambda\in\mathbb{N}}$$

Equivocality: *For any* $\lambda\in\mathbb{N}$, *any* $(\mathsf{crs},t)\in\mathsf{support}(\mathcal{S}_{com1}(1^\lambda))$, *and any adversary* $\mathcal{A}$, *the following distributions are identical.*

- $\left\{m\leftarrow\mathcal{A}(\mathsf{crs});\ r\leftarrow\mathcal{R};\ c=\mathsf{Com}_{\mathsf{crs}}(m;r):\ (m,r,c)\right\}$
- $\left\{m\leftarrow\mathcal{A}(\mathsf{crs});\ (c,s)\leftarrow\mathcal{S}_{com2}(t);\ r\leftarrow\mathcal{S}_{com3}(s,m):\ (m,r,c)\right\}$

3 A Generic Framework for Two-Round OT

In this section we describe **Protocol 1***, an adaptively secure, 2-round protocol. Let $(\mathcal{K},\mathcal{P},\mathcal{V})$ be a dual-mode NIZK proof system, $(\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})$ be a CCA-secure labeled public-key encryption scheme, and $\mathcal{H}=(\mathsf{HashPG},\mathsf{SampYes},\mathsf{SampNo},\mathsf{HashKG},\mathsf{Hash},\mathsf{pHash})$ be a smooth hash proof system with a hard subset membership property. We assume for simplicity that $\{0,1\}^\ell$ is the range of the hash functions in $\mathcal{H}$; known constructions can be modified to achieve this property. Based on these components, we construct an OT protocol between a sender P_i and a receiver P_j in the CRS model; refer also to Figure 1.

Common Reference String: Compute $\mathrm{PP}\leftarrow\mathsf{HashPG}(1^\lambda)$, $(pk,sk)\leftarrow\mathsf{Gen}(1^\lambda)$, and $\mathsf{crs}_{nizk}\leftarrow\mathcal{K}(1^\lambda)$. The common reference string is $\mathsf{crs}_{ot}=(\mathrm{PP},pk,\mathsf{crs}_{nizk})$.

Oblivious Transfer: The protocol starts by having the receiver, holding selection bit b, send two instances (x_0,x_1) for the hash proof system $\mathcal{H}$ with x_{1-b} a NO-instance; the receiver sends $\mathsf{Enc}_{pk}(b)$ and a NIZK proof that x_{1-b} is a NO-instance as well. In the second round, for $\sigma\in\{0,1\}$ the sender generates primary and projection hash keys $(\mathrm{HK}_\sigma,\mathrm{PK}_\sigma)$ and sends $(\mathrm{PK}_\sigma,\mathsf{Hash}(\mathrm{HK}_\sigma,x_\sigma)\oplus m_\sigma)$ to the receiver. The receiver recovers m_b in the standard way. In more detail:

- On input a selection bit b, the receiver P_j proceeds as follows:
  1. Compute $(x_b,w)\leftarrow\mathsf{SampYes}(\mathrm{PP})$ and $x_{1-b}=\mathsf{SampNo}(\mathrm{PP};\gamma)$ for uniform γ. Compute $\Phi=\mathsf{Enc}^L_{pk}(b;\xi)$ with uniformly random ξ, where $L=(\mathsf{sid},\mathsf{ssid},P_i,P_j)$. Generate an NIZK proof π that there exist (b,γ,ξ) such that $x_{1-b}=\mathsf{SampNo}(\mathrm{PP};\gamma)$ and $\Phi=\mathsf{Enc}^L_{pk}(b;\xi)$.
  2. Send $\langle x_0,x_1,\Phi,\pi\rangle$.
- On input $m_0,m_1\in\{0,1\}^\ell$, and after receiving the first-round message $\langle x_0,x_1,\Phi,\pi\rangle$ from the receiver, the sender P_i proceeds as follows:
  1. If the proof π does not verify, abort.
  2. For $\sigma\in\{0,1\}$ compute $(\mathrm{HK}_\sigma,\mathrm{PK}_\sigma)\leftarrow\mathsf{HashKG}(\mathrm{PP})$ and $Z_\sigma=m_\sigma\oplus\mathsf{Hash}(\mathrm{HK}_\sigma,x_\sigma)$.
  3. Send $\langle\mathrm{PK}_0,Z_0,\mathrm{PK}_1,Z_1\rangle$ to P_j.
- Upon receiving the second-round message $\langle\mathrm{PK}_0,Z_0,\mathrm{PK}_1,Z_1\rangle$, the receiver P_j computes the output $m_b=Z_b\oplus\mathsf{pHash}(\mathrm{PK}_b,x_b,w)$.

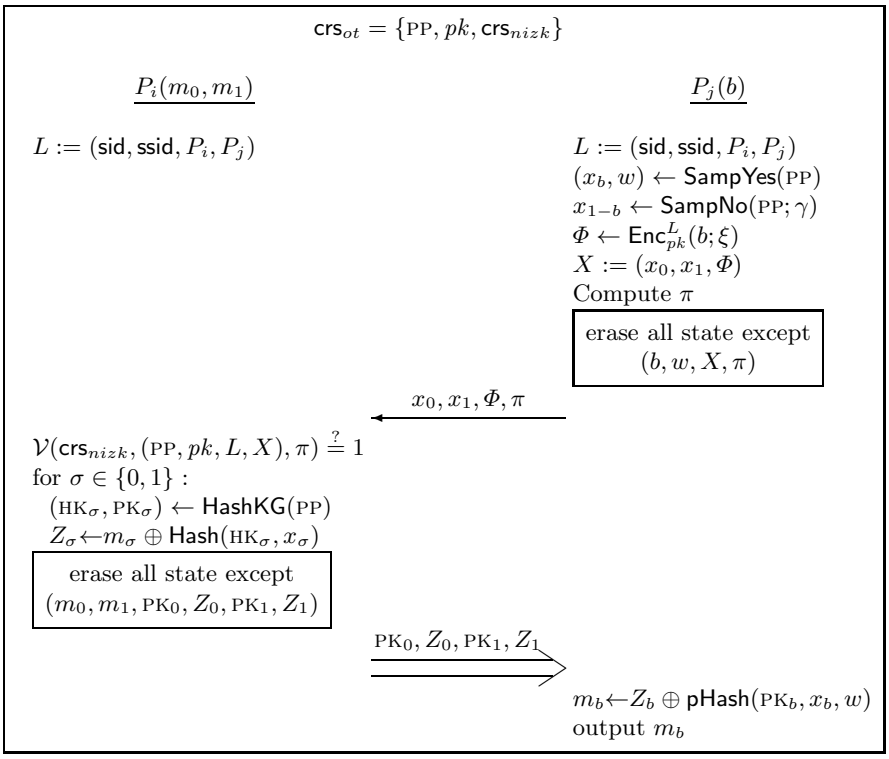


Fig. 1. An OT protocol in the $\mathcal{F}_{\mathsf{CRS}}$-hybrid model (**Protocol 1***). For adaptive security, the second-round message is sent over an adaptively secure channel.

Informally, security against a malicious sender holds because the sender cannot guess the receiver's selection bit due to the hard subset membership property. On the other hand, a malicious receiver gets no information about m_{1-b} if x_{1-b} is a NO-instance, and this property is enforced by the NIZK proof.

Theorem 1. *Say* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is a CCA-secure labeled public-key encryption scheme,* $(\mathsf{HashPG}, \mathsf{SampYes}, \mathsf{SampNo}, \mathsf{HashKG}, \mathsf{Hash}, \mathsf{pHash})$ *is a smooth projective hash proof system with hard subset membership property, and* $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ *is a dual-mode NIZK proof system. Then the protocol described above securely realizes* $\mathcal{F}_{\mathsf{MOT}}$ *in the* $\mathcal{F}_{\mathsf{CRS}}$*-hybrid model, for static corruptions. If the second round message is sent over an adaptively secure channel, the protocol securely realizes* $\mathcal{F}_{\mathsf{MOT}}$ *in the* $\mathcal{F}_{\mathsf{CRS}}$*-hybrid model, for adaptive corruptions (assuming erasure).*

In the full version of this work, we discuss concrete instantiations of this framework based on the DLIN and SXDH assumptions.

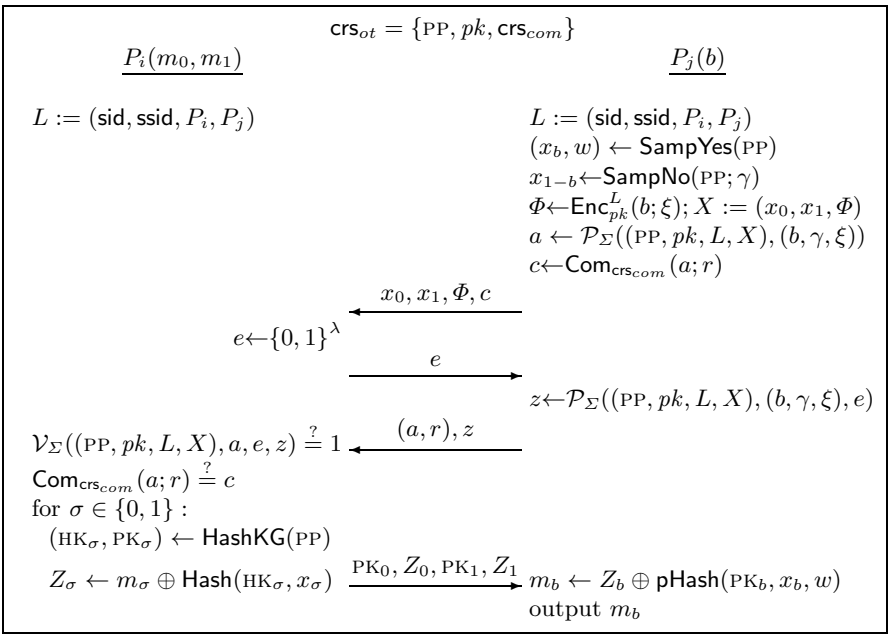


Fig. 2. A statically secure OT protocol in the $\mathcal{F}_{\mathsf{CRS}}$-hybrid model (**Protocol 2**)

4 A Generic Framework for Four-Round OT

In this section, we describe a generic framework for constructing four-round OT protocols. We begin by looking at the case of static security, and then show how the ideas can be extended to achieve security against adaptive adversaries.

4.1 Static Security (Protocol 2)

The main idea is to adapt our previous two-round framework by replacing the dual-mode NIZK proof with an interactive equivalent. In particular, the general structure of the protocol is as follows: the protocol starts by having the receiver send two instances (x_0, x_1) for hash proof system where x_{1-b} being a NO-instance; also, in protection against a malicious behavior, $\mathsf{Enc}_{pk}(b)$ and a Sigma protocol (augmented with an equivocal commitment) are attached. Then, the sender generates primary and projective hash keys $(\text{HK}_\sigma, \text{PK}_\sigma)$ for each instance x_σ and sends $(\text{PK}_\sigma, \mathsf{Hash}(\text{HK}_\sigma, x_\sigma) \oplus m_\sigma)$ to the receiver. The security can be shown similarly to the two-round OT case.

Here, instead of replicating all the details, we only describe how to combine a Sigma protocol with an equivocal commitment scheme in order to replace the NIZK part. The idea is having the prover commit to the first round message of the Sigma protocol, and reveal it in the third round. Refer to Figure 2 for the overall pictorial description of the protocol.

CRS. Compute $\mathrm{PP} \leftarrow \mathsf{HashPG}(1^\lambda)$, $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, and $\mathsf{crs}_{com} \leftarrow \mathcal{K}_{com}(1^\lambda)$. The common reference string is $\mathsf{crs}_{ot} = (\mathrm{PP}, pk, \mathsf{crs}_{com})$.

Replacing NIZK. Recall in the two-round OT case, the receiver generates a NIZK π to prove that (x_0, x_1, Φ) is valid message, i.e., Φ is an encryption of $b \in \{0,1\}$ for some b and x_{1-b} is NO-instance. In this protocol, the receiver proves it by running a Sigma protocol $(\mathcal{P}_\Sigma, \mathcal{V}_\Sigma)$, along with an equivocal commitment scheme $(\mathcal{K}_{com}, \mathsf{Com})$, with respect to the following language:

$$\mathcal{L}^* = \left\{ \begin{array}{l} (\mathrm{PP}, pk, L, x_0, x_1, \Phi) : \\ \quad \exists (b, \gamma, \xi) \text{ s.t. } x_{1-b} = \mathsf{SampNo}(\mathrm{PP}; \gamma), \Phi = \mathsf{Enc}^L_{pk}(b; \xi) \end{array} \right\},$$

where $L = (\mathsf{sid}, \mathsf{ssid}, P_i, P_j)$.

1. The receiver runs $a \leftarrow \mathcal{P}_\Sigma((\mathrm{PP}, pk, L, x_0, x_1, \Phi), (b, \gamma, \xi))$, and computes $c = \mathsf{Com}_{\mathsf{crs}_{com}}(a; r)$ with r chosen uniformly at random. It sends (x_0, x_1, Φ, c).
2. The sender sends the challenge message $e \leftarrow \{0,1\}^\lambda$ of the Sigma protocol.
3. Upon receiving the challenge e, the receiver generates an answer by running
$$z = \mathcal{P}_\Sigma((\mathrm{PP}, pk, L, x_0, x_1, \Phi), (b, \gamma, \xi), e).$$
It sends the sender the answer z along with the opening of the commitment, i.e., $((a, r), z)$.
4. The sender verifies (a, e, z) is an accepting transcript and (a, r) is a valid opening of c:
$$\mathcal{V}_\Sigma((\mathrm{PP}, pk, L, x_0, x_1, \Phi), a, e, z) \stackrel{?}{=} 1, \quad \mathsf{Com}_{\mathsf{crs}_{com}}(a; r) \stackrel{?}{=} c.$$

The security of the protocol can be proved similarly to the two-round case.

Theorem 2. *Say* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is a CCA-secure labeled public-key encryption scheme,* $(\mathsf{HashPG}, \mathsf{SampYes}, \mathsf{SampNo}, \mathsf{HashKG}, \mathsf{Hash}, \mathsf{pHash})$ *is a smooth projective hash proof system with hard subset membership property,* $(\mathcal{P}_\Sigma, \mathcal{V}_\Sigma)$ *is a* Σ*-protocol, and* $(\mathcal{K}_{com}, \mathsf{Com})$ *is an equivocal commitment scheme. Then the protocol of Figure 2 securely realizes* $\mathcal{F}_{\mathsf{MOT}}$ *in the* $\mathcal{F}_{\mathsf{CRS}}$*-hybrid model, for static corruptions.*

4.2 Adaptive Security (Protocol 2*)

As with the 2-round framework, the protocol first needs to be changed so that the last round message is sent over a secure channel. This modification (along with erasing the state appropriately), however, is not sufficient to deal with adaptive corruption in the four-round case. For the NIZK, the receiver can generate π and then erase the unnecessary internal state before sending out (x_0, x_1, Φ, π). However, if the statement is composed with the interactive Sigma protocol, some of the internal state cannot be erased until the last move. For example, in the Sigma protocol, the receiver cannot erase the randomness used for generating the NO-instance x_{1-b} until it receives the challenge e, since he has

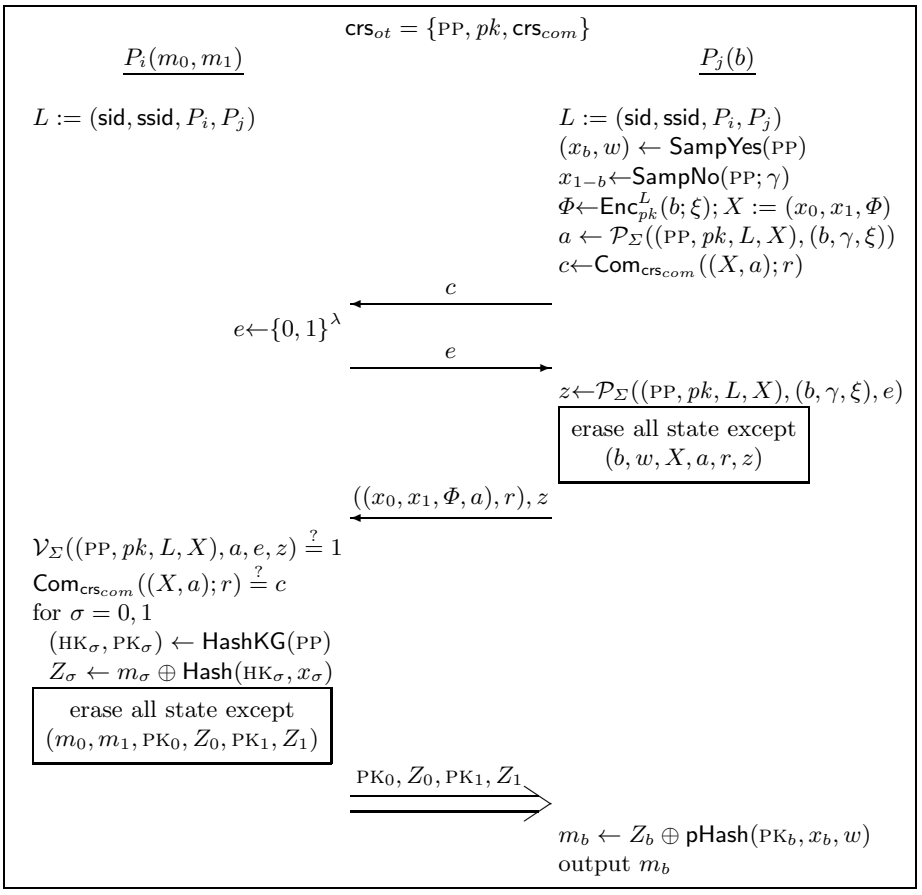


Fig. 3. An adaptively secure OT protocol in the $\mathcal{F}_{\mathsf{CRS}}$-hybrid model (**Protocol 2***). The final message is sent over an adaptively secure channel.

to use the randomness as part of the witness in order to finish the proof. However, recall that both x_0 and x_1 are YES instances in simulation; when the adversary corrupts the receiver right before sending e, the simulator cannot return a valid randomness for x_{1-b}, and so the simulation breaks down.

Changing the Order of Messages. As in the commitment scheme [27], we resolve this issue by switching the order of messages. That is, the message to be committed to is not only the first message a of the Sigma protocol but also the statement itself (i.e., (x_0, x_1, Φ)), and they are revealed at the last move of the Sigma protocol. Now, thanks to the equivocality of the commitment scheme, the protocol can achieve adaptive security. Refer to Figure 3 for the overall pictorial description. Here, we only describe the aforementioned modification in more detail. Recall in the statically secure protocol described in Section 4.1, the

receiver sends (x_0, x_1, Φ) and the commitment c to the first message a of the Sigma protocol $(\mathcal{P}_\Sigma, \mathcal{V}_\Sigma)$ for the language

$$\mathcal{L}^* = \left\{ \begin{array}{l} (\text{PP}, pk, L, x_0, x_1, \Phi) : \\ \quad \exists (b, \gamma, \xi) \text{ s.t. } x_{1-b} = \mathsf{SampNo}(\text{PP}; \gamma), \Phi = \mathsf{Enc}^L_{pk}(b; \xi) \end{array} \right\},$$

where $L = (\mathsf{sid}, \mathsf{ssid}, P_i, P_j)$. In this protocol, we change the order of messages as follows:

1. The receiver runs $a \leftarrow \mathcal{P}_\Sigma((\text{PP}, pk, L, x_0, x_1, \Phi), (b, \gamma, \xi))$, and then computes $c \leftarrow \mathsf{Com}_{\mathsf{crs}_{com}}((x_0, x_1, \Phi, a); r)$ with r chosen uniformly at random. It sends c.
2. The sender sends the challenge message $e \leftarrow \{0,1\}^\lambda$ of the Sigma protocol.
3. Upon receiving the challenge e, the receiver generates an answer by running

$$z = \mathcal{P}_\Sigma((\text{PP}, pk, L, x_0, x_1, \Phi), (b, \gamma, \xi), e).$$

   It sends the sender the answer z along with the opening of the commitment, i.e., $((x_0, x_1, \Phi, a), r, z)$.
4. The sender verifies (a, e, z) is an accepting transcript and $((x_0, x_1, \Phi, a), r)$ is a valid opening of c:

$$\mathcal{V}_\Sigma((\text{PP}, pk, L, x_0, x_1, \Phi), a, e, z) \stackrel{?}{=} 1, \quad \mathsf{Com}_{\mathsf{crs}_{com}}((x_0, x_1, \Phi, a); r) \stackrel{?}{=} c.$$

Theorem 3. *Under the same assumptions as in Theorem 2, the protocol in Figure 3 securely realizes $\mathcal{F}_{\mathsf{MOT}}$ in the $\mathcal{F}_{\mathsf{CRS}}$-hybrid model, for adaptive corruptions (assuming erasure).*

4.3 Instantiations from the DDH Assumption

We show a CCA-secure labeled public-key encryption scheme, a smooth hash proof system, and an equivocal commitment scheme under the DDH assumption. We then obtain a four-round OT protocol by combining these building blocks.

Decisional Diffie-Hellman Assumption. Let $\mathcal{G}_{\text{ddh}}$ be a randomized algorithm that takes a security parameter λ and outputs $\mathsf{desc} = (p, \mathbb{G}, g)$ such that $\mathbb{G}$ is the description of group of prime order p, and g is a generator of $\mathbb{G}$.

Definition 6. *The* DDH problem is hard relative to $\mathbb{G}$ *if for all* PPT *algorithms $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that*

$$\left|\Pr[\mathcal{A}(\mathbb{G}, p, g, g^a, g^b, g^c) = 1] - \Pr[\mathcal{A}(\mathbb{G}, p, g, g^a, g^b, g^{ab}) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where in each case the probabilities are taken over the experiment in which the group-generating algorithm outputs $(\mathbb{G}, p, g)$ and random $a, b, c \in \mathbb{Z}_p$ are chosen.

CCA-secure Labeled Public-Key Encryption. Since the DDH assumption holds in $\mathbb{G}_1$, we can use Cramer-Shoup encryption scheme [12]. As in the case for the DLIN assumption, we slightly change the scheme to support labels, that is, we use collision resistant hash functions instead of UOWHF and apply labels to hash functions when performing encryptions and decryptions.

Key generation $(pk, sk) \leftarrow \mathsf{Gen}(\mathsf{desc})$: Choose random generators $g_1 \leftarrow \mathbb{G}$ and exponents $\beta_1, \beta_2, \gamma_1, \gamma_2, \delta_1, \delta_2 \leftarrow \mathbb{Z}_p$ and compute $c = g_1^{\beta_1} g^{\beta_2}, d = g_1^{\gamma_1} g^{\gamma_2}, h = g_1^{\delta_1} g^{\delta_2}$. Choose a hash function $H \leftarrow \mathcal{HF}$ where $\mathcal{HF}$ is a family of collision-resistant hash functions. Now set $pk = (g_1, g, c, d, h, H)$ and $sk = (\beta_1, \beta_2, \gamma_1, \gamma_2, \delta_1, \delta_2)$.

Encryption $C \leftarrow \mathsf{Enc}_{pk}^{L}(m; r)$: Given the message $m \in \mathbb{G}$ under label L, choose $r \leftarrow \mathbb{Z}_p$ and compute $u_1 = g_1^r, u_2 = g^r, e = m \cdot h^r$. Then compute $\alpha = H(u_1, u_2, e, L) \in \mathbb{Z}_p$ and $v = (cd^{\alpha})^r$. The ciphertext is $C = (u_1, u_2, e, v)$.

Decryption $\mathsf{Dec}_{sk}^{L}(C)$: Parse $C = (u_1, u_2, e, v)$ and $sk = (\beta_1, \beta_2, \gamma_1, \gamma_2, \delta_1, \delta_2)$; compute $\alpha \leftarrow H(u_1, u_2, e, L)$ and test if $u_1^{\beta_1 + \alpha\gamma_1} \cdot u_2^{\beta_2 + \alpha\gamma_2} \stackrel{?}{=} v$. If it does not, output `reject`. Otherwise, output $m = e/(u_1^{\delta_1} u_2^{\delta_2})$.

Smooth Projective Hashing. We recall the smooth projective hashing based on the DDH assumption [12,13].

Parameter Generation. Choose $g_1, g \leftarrow \mathbb{G}$. Then $\text{PP} = (g_1, g, \mathbb{G})$.

Instance Sampling. To sample a YES instance, choose $t \leftarrow \mathbb{Z}_p$, and compute $z_1 = g_1^t$, $z_2 = g^t$, and then return $x = (z_1, z_2)$. To sample a NO instance, choose $t \leftarrow \mathbb{Z}_p$, and then $z_1 = g_1^t$, $z_2 = g^{t+1}$, and then return $x = (z_1, z_2)$.

Hash Key Generation. Choose $\theta_1, \theta_2 \leftarrow \mathbb{Z}_p$ and compute $f = g_1^{\theta_1} g^{\theta_2}$. Return $\text{HK} = (\theta_1, \theta_2)$, and $\text{PK} = f$.

Primary Hashing. Given $\text{HK} = (\theta_1, \theta_2)$ and $x = (z_1, z_2)$, return $y = z_1^{\theta_1} z_2^{\theta_2}$.

Projective Hashing. Given a projective hash key $\text{PK} = f$, an instance $x = (z_1, z_2)$, and its witness $w = t$ such that $z_1 = g_1^t$, $z_2 = g^t$, return $y = f^t$.

Equivocal Commitment. We use a variant of the Pedersen commitment scheme [32]. The main difference from the original Pedersen commitment is that collision resilient hash function $H : \{0,1\}^* \to \mathbb{Z}_p$ is used to commit to arbitrary long message very efficiently. In particular, given the CRS $(g, h_1) \in \mathbb{G}^2$, the commitment to a message m is $g^r h_1^{H(m)}$. We note that the binding property is under the DLOG assumption and the collision resilient property of the hash function. When a trapdoor ζ with $h_1 = g^{\zeta}$ is known, it easy to equivocate a commitment $c = g^s$ into any m by outputting $r = s - \zeta \cdot H(m)$.

By plugging these components into our generic framework for four-round OT, we obtain an OT protocol based on the DDH assumption. It is only left to show the concrete Σ-protocol that is used.

Protocol Details. Ignoring the description desc of the group $\mathbb{G}$, the CRS is $\mathsf{crs}_{ot} = (\text{PP}, pk, \mathsf{crs}_{com})$ where $\text{PP} = (g_1, g)$ $pk = (g_1, g, c, d, h, H)$ $\mathsf{crs}_{com} = (h_1, g)$. Therefore, the CRS can be represented with 6 group elements of $\mathbb{G}$ and one hash function index, along with the description of the group $\mathbb{G}$.

Let $x_0 = (z_{01}, z_{02})$, $x_1 = (z_{11}, z_{12})$, and $\Phi = (u_1, u_2, e, v)$ with $\alpha = H(u_1, u_2, e, (\mathsf{sid}, \mathsf{ssid}, P_i, P_j))$. Then, we use a standard Sigma protocol for the following language:

$$\mathcal{L}^* = \left\{ \begin{array}{l} (\mathsf{crs}_{ot}, pk, x_0, x_1, \Phi, \alpha) : \\ \exists (r,t) \text{ s.t. } u_1 = g_1^r, u_2 = g^r, e = h^r, v = (cd^{\alpha})^r, z_{11} = g_1^t, z_{12} = g^{t+1} \\ \qquad \text{or } u_1 = g_1^r, u_2 = g^r, e = gh^r, v = (cd^{\alpha})^r, z_{01} = g_1^t, z_{02} = g^{t+1} \end{array} \right\}.$$

1. Suppose that $\Phi = \mathsf{Enc}(g^b)$. Let $\bar{b} = 1 - b$. The prover chooses $R, T \leftarrow \mathbb{Z}_p$, $\eta \leftarrow [0, 2^\lambda)$, and $\rho, \tau \leftarrow \mathbb{Z}_p$. Then, it computes and sends the verifier the following:

$$\begin{array}{lll} U_{1b} = g_1^R, & U_{2b} = g^R, & E_b = h^R, \\ V_b = (cd^\alpha)^R, & Z_{1b} = g_1^T, & Z_{2b} = g^T \\ U_{1\bar{b}} = g_1^\rho / u_1^\eta, & U_{2\bar{b}} = g_1^\rho / u_2^\eta, & E_{\bar{b}} = h^\rho / (e/g^{\bar{b}})^\eta, \\ V_{\bar{b}} = (cd^\alpha)^\rho / v^\eta, & Z_{1\bar{b}} = g_1^\tau / z_{b1}^\eta, & Z_{2\bar{b}} = g^\tau / (z_{b2}/g)^\eta. \end{array}$$

2. The verifier chooses $\epsilon \leftarrow [0, 2^\lambda)$ and sends it to the prover.
3. The prover computes the following:

$$\begin{array}{ll} \epsilon_b = \epsilon - \eta \bmod 2^\lambda & \epsilon_{\bar{b}} = \eta \\ \rho_b = R + r\epsilon_b & \rho_{\bar{b}} = \rho \\ \tau_b = T + t\epsilon_b & \tau_{\bar{b}} = \tau. \end{array}$$

   Then, it sends $(\epsilon_0, \rho_0, \tau_0, \rho_1, \tau_1)$ to the verifier.
4. The verifier computes $\epsilon_1 = \epsilon - \epsilon_0 \bmod 2^\lambda$. It also checks if the following holds for $i \in \{0, 1\}$.

$$\begin{array}{lll} g_1^{\rho_i} = U_{1i} \cdot u_1^{\epsilon_i}, & g^{\rho_i} = U_{2i} \cdot u_2^{\epsilon_i}, & h^{\rho_i} = E_i \cdot (e/g^i)^{\epsilon_i}, \\ (cd^\alpha)^{\rho_i} = V_i \cdot v^{\epsilon_i}, & g_1^{\tau_i} = Z_{1i} \cdot z_{\bar{i}1}^{\epsilon_i}, & g^{\tau_i} = Z_{2i} \cdot (z_{\bar{i}2}/g)^{\epsilon_i}. \end{array}$$

Communication Complexity. The receiver message (x_0, x_1, Φ) needs $2 + 2 + 4 = 8$ group elements. The proof takes 13 elements in $\mathbb{G}$ and 7 elements in $\mathbb{Z}_p$. In particular, the first message has one commitment (i.e., one element in $\mathbb{G}$). The second message has one element[3] in $\mathbb{Z}_p$, and the third messages has 5 elements in $\mathbb{Z}_p$ along with the decommitment (i.e., 12 elements in $\mathbb{G}$ and 1 element in $\mathbb{Z}_p$). The sender message (pk_0, Z_0, pk_1, Z_1) needs $(1, 1, 1, 1) = 4$ group elements in $\mathbb{G}$. Therefore, the total communication complexity amounts to 25 elements in $\mathbb{G}$ and 7 elements in $\mathbb{Z}_p$.

Realizing an Adaptively Secure Channel. Note that the non-committing encryption given in [2] runs in three rounds and needs one public key and one ciphertext of a semantically secure public key encryption scheme. Since the NCE protocol UC-realizes an adaptively secure channel [5, Section 6.3], the NCE protocol messages can be overlapped with the OT protocol messages (aligning the first message of the NCE protocol with the second message of the OT protocol), and thus the final OT protocol runs in four rounds. We can use ElGamal encryption, and the communication overhead amounts to 3 group elements; the public key consists of one element excluding the generator in the CRS, and the ciphertext consists of two elements.

Acknowledgments. We would like to thank the anonymous reviewers for pointing out the need to transmit the sender's messages over an adaptively secure channel, and for additional helpful feedback.

[3] The second message is in $\{0, 1\}^\lambda$ but we count it as an element of $\mathbb{Z}_p$ for simplicity.

References

1. Aiello, W., Ishai, Y., Reingold, O.: Priced Oblivious Transfer: How to Sell Digital Goods. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 119–135. Springer, Heidelberg (2001)
2. Beaver, D., Haber, S.: Cryptographic Protocols Provably Secure against Dynamic Adversaries. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 307–323. Springer, Heidelberg (1993)
3. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
4. Camenisch, J., Shoup, V.: Practical Verifiable Encryption and Decryption of Discrete Logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
5. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd Annual Symposium on Foundations of Computer Science (FOCS), pp. 136–145. IEEE (2001)
6. Canetti, R.: Obtaining Universally Compoable Security: Towards the Bare Bones of Trust. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 88–112. Springer, Heidelberg (2007)
7. Canetti, R., Fischlin, M.: Universally Composable Commitments. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 19–40. Springer, Heidelberg (2001)
8. Canetti, R., Kushilevitz, E., Lindell, Y.: On the limitations of universally composable two-party computation without set-up assumptions. Journal of Cryptology 19(2), 135–167 (2006)
9. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: 34th Annual ACM Symposium on Theory of Computing (STOC), pp. 494–503. ACM Press (May 2002)
10. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Improved Non-committing Encryption with Applications to Adaptively Secure Protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 287–302. Springer, Heidelberg (2009)
11. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Simple, Black-Box Constructions of Adaptively Secure Protocols. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 387–402. Springer, Heidelberg (2009)
12. Cramer, R., Shoup, V.: A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
13. Cramer, R., Shoup, V.: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
14. Damgård, I.: Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 418–430. Springer, Heidelberg (2000)
15. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust Non-interactive Zero Knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Heidelberg (2001)
16. Fischlin, M., Libert, B., Manulis, M.: Non-interactive and Re-usable Universally Composable String Commitments with Adaptive Security. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 468–485. Springer, Heidelberg (2011)
17. Garay, J.A., MacKenzie, P., Yang, K.: Efficient and Universally Composable Committed Oblivious Transfer and Applications. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 297–316. Springer, Heidelberg (2004)

18. Garay, J.A., Wichs, D., Zhou, H.-S.: Somewhat Non-committing Encryption and Efficient Adaptively Secure Oblivious Transfer. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 505–523. Springer, Heidelberg (2009)
19. Groth, J.: Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (2006)
20. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
21. Halevi, S., Kalai, Y.T.: Smooth projective hashing and two-message oblivious transfer. Journal of Cryptology 25(1), 158–193 (2012)
22. Horvitz, O., Katz, J.: Universally-Composable Two-Party Computation in Two Rounds. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 111–129. Springer, Heidelberg (2007)
23. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding Cryptography on Oblivious Transfer – Efficiently. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 572–591. Springer, Heidelberg (2008)
24. Jarecki, S., Shmatikov, V.: Efficient Two-Party Secure Computation on Committed Inputs. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 97–114. Springer, Heidelberg (2007)
25. Lindell, A.Y.: Efficient Fully-Simulatable Oblivious Transfer. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 52–70. Springer, Heidelberg (2008)
26. Lindell, A.Y.: Adaptively Secure Two-Party Computation with Erasures. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 117–132. Springer, Heidelberg (2009)
27. Lindell, Y.: Highly-Efficient Universally-Composable Commitments Based on the DDH Assumption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 446–466. Springer, Heidelberg (2011)
28. Lindell, Y., Oxman, E., Pinkas, B.: The IPS Compiler: Optimizations, Variants and Concrete Efficiency. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 259–276. Springer, Heidelberg (2011)
29. Lindell, Y., Pinkas, B.: Secure Two-Party Computation via Cut-and-Choose Oblivious Transfer. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 329–346. Springer, Heidelberg (2011)
30. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: 12th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 448–457. ACM-SIAM (2001)
31. 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)
32. Pedersen, T.P.: Non-interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992)
33. Peikert, C., Vaikuntanathan, V., Waters, B.: A Framework for Efficient and Composable Oblivious Transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008)
34. Scott, M.: Authenticated ID-based key exchange and remote log-in with simple token and PIN. Cryptology ePrint Archive, Report 2002/164 (2002)

Cryptography Using Captcha Puzzles

Abishek Kumarasubramanian[1], Rafail Ostrovsky[1,⋆], Omkant Pandey[2], and Akshay Wadia[1]

[1] University of California, Los Angeles
{abishekk,rafail,awadia}@cs.ucla.edu
[2] University of Texas at Austin
omkant@cs.utexas.edu

Abstract. A Captcha is a puzzle that is easy for humans but hard to solve for computers. A formal framework, modelling Captcha puzzles (as hard AI problems), was introduced by Ahn, Blum, Hopper, and Langford ([1], Eurocrypt 2003). Despite their attractive features and wide adoption in practice, the use of Captcha puzzles for general cryptographic applications has been limited.

In this work, we explore various ways to formally model Captcha puzzles and their human component and explore new applications for Captcha. We show that by defining Captcha with additional (strong but realistic) properties, it is possible to broaden Captcha applicability, including using it to learning a machine's "secret internal state." To facilitate this, we introduce the notion of an human-extractable Captcha, which we believe may be of independent interest. We show that this type of Captcha yields a *constant round* protocol for *fully* concurrent non-malleable zero-knowledge. To enable this we also define and construct a Captcha-based commitment scheme which admits "straight line" extraction. We also explore Captcha definitions in the setting of Universal Composability (UC). We show that there are two (incomparable) ways to model Captcha within the UC framework that lead to different results. In particular, we show that in the so called *indirect access model*, for every polynomial time functionality $\mathcal{F}$ there exists a protocol that UC-realizes $\mathcal{F}$ using human-extractable Captcha, while for the so-called *direct access model*, UC is impossible, even with the help of human-extractable Captcha.

The security of our constructions using human-extractable Captcha is proven against the (standard) class of all polynomial time adversaries.

⋆ Department of Computer Science and Mathematics, UCLA, Email: rafail@cs.ucla.edu. Research supported in part by NSF grants CNS-0830803; CCF-0916574; IIS-1065276; CCF-1016540; CNS-1118126; CNS-1136174; US-Israel BSF grant 2008411, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. This material is also based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 89–106, 2013.

In contrast, most previous works guarantee security only against a very limited class of adversaries, called the *conservative* adversaries.

Keywords: CAPTCHA, concurrent non-malleable zero-knowledge, universal composability, human-extractable CAPTCHA.

1 Introduction

CAPTCHA is an acronym for *Completely Automated Public Turing test to tell Computers and Humans Apart*. These are puzzles that are easy for humans but hard to solve for automated computer programs. They are used to confirm the "presence of a human" in a communication channel. As an illustration of a scenario where such a confirmation is very important, consider the problem of spam. To carry out their nefarious activities, spammers need to create a large number of fake email accounts. Creating a new email account usually requires the filling-in of an online form. If the spammers were to manually fill-in all these forms, then the process would be too slow, and they would not be able to generate a number of fake addresses. However, it is relatively simple to write a script (or an automated *bot*) to quickly fill-in the forms automatically without human intervention. Thus, it is crucial for the email service provider to ensure that the party filling-in the form is an actual human, and not an automated script. This is achieved by asking the party to solve a CAPTCHA, which can only be sovled by a human[1]. A common example of a CAPTCHA puzzle involves the distorted image of a word, and the party is asked to identify the word in the image.

The definition of CAPTCHA stipulates certain limitations on the power of machines, in particular, that they cannot solve CAPTCHA puzzles efficiently. This gives rise to two distinct questions which are interesting from a cryptographic point of view. Firstly, what are the underlying hard problems upon which CAPTCHA puzzles can be based? Von Ahn, Blum, Hopper and Langford [1] study this question formally, and provide constructions based on the conjectured hardness of certain Artificial Intelligence problems.

The second direction of investigation, and the one which we are concerned with in this paper, is to use CAPTCHAs as a tool for achieving general cryptographic tasks. There have been only a few examples of use of CAPTCHAs in this regard. Von Ahn, Blum, Hopper and Langford [1] use CAPTCHAs for image-based steganography. Canetti, Halevi and Steiner construct a scheme to thwart off-line dictionary attacks on encrypted data using CAPTCHAs. And recently, Dziembowski [3] constructs a "human" key agreement protocol using only CAPTCHAs. We continue this line of work in the current paper, and investigate the use of CAPTCHAs in zero-knowledge and UC secure protocols. On the face of it, it is unclear how CAPTCHAs may be used for constructing such protocols, or even for constructing building blocks for these protocols, like commitment schemes.

[1] For many more uses of CAPTCHA, see [2]

However, motivated by current CAPTCHA theory, we define a new *extraction* property of CAPTCHAs that allows us to use them for designing these protocols.

We now give an overview of our contributions. We formally define CAPTCHAs in Section 3, but give an informal overview of the model here to make the following discussion cogent. Firstly, modelling CAPTCHA puzzles invariably involves modelling humans who are the key tenets in distinguishing CAPTCHAs from just another one-way function. Following [4] we model the presence of a human entity as an oracle H that is capable of solving CAPTCHA puzzles. A party generates a CAPTCHA puzzle by running a (standard) PPT generation algorithm denoted by G. This algorithm outputs a puzzle-solution pair (z, a). All parties have access to a "human" oracle denoted by H. To "solve" a CAPTCHA puzzle, a party simply queries its oracle with the puzzle and obtains the solution in response. This allows us to distinguish between two classes of machines. Standard PPT machines for which solving CAPTCHAs is a hard problem and oracle PPT machines with oracle access to H which may solve CAPTCHAs efficiently.

The starting point of our work is the observation that if a machine must solve a given CAPTCHA puzzle (called *challenge*), it *must* send one or more CAPTCHA-queries to a human. These queries are likely to be correlated to the challenge puzzle since otherwise they would be of no help in solving the challenge puzzle. Access to these queries, with the help of another human, may therefore provide us with some knowledge about the internal state of a (potentially) malicious machine! This is formulated in our definition of an human extractable CAPTCHA (Definition 3). Informally, we make the following assumption about CAPTCHA puzzles. Consider two randomly chosen CAPTCHA puzzles (p_0, p_1) of which an adversary obtains only one to solve, say p_b, where the value of b is not known to the challenger. Then by merely looking at his queries to a human oracle H, and with the help of a *human*, a challenger must be able to identify the value of b. More precisely, we augment the human oracle H to possess this added ability. We then model adversaries in our protocols as oracle PPT machines with access to a CAPTCHA solving oracle, but whose internal state can be "extracted" by another oracle PPT machine.

It is clear that this idea, i.e.—the idea of learning something non-trivial about a machine's secret by looking at its CAPTCHA-queries—connects CAPTCHA puzzles with main-stream questions in cryptography much more than ever. This work uses this feature present in CAPTCHAs to construct building blocks for zero-knowledge protocols which admit "straight-line" simulation. It is then natural to investigate that if we can get "straight-line" simulation, then perhaps we can answer the following questions as well: construction of *plain-text aware* encryption schemes [5], "straight-line" extractable commitment schemes, constant-round fully concurrent zero-knowledge for **NP** [6], fully concurrent two/multi-party computation [7–9], universal composition *without* trusted setup assumptions [10, 11], and so on.

Our Contribution. In section 4 (theorem 5), as the first main result of this work, we construct a commitment scheme which admits "straight-line" extraction.

That is, the committed value can be extracted by looking at the CAPTCHA-queries made by the committer to a human oracle.

The starting point (ignoring for a moment an important difficulty) behind our commitment protocol is the following. The receiver R chooses two independent CAPTCHA puzzles (z_0, z_1). To commit to a bit b, the sender C will select z_b using the 1-2-OT protocol and commit to its solution a_b using an ordinary (perfectly-binding) commitment scheme. Since the committer cannot solve the puzzle itself, it must query a human to obtain the solution. By looking at the puzzles C queries to the human, an extractor (with the help of another human oracle) can detect the bit being committed. Since the other puzzle z_{1-b} is computationally hidden from C, this should indeed be possible.

As alluded above, the main difficulty with this approach is that a cheating sender may not query the human on any of the two puzzles, but might still be able to commit to a correct value by obtaining solutions to some related puzzles. This is the issue of malleability that we discuss shortly, and also in section 3.

We then use this commitment scheme as a tool to obtain new results in protocol composition. First off, it is straightforward to see that given such a scheme, one can obtain a constant-round concurrent zero-knowledge protocol for all of **NP**. In fact, by using our commitment scheme in place of the "PRS-preamble" [12] in the protocol of Barak, Prabhakaran, and Sahai [13], we obtain a constant-round protocol for *concurrent non-malleable zero-knowledge* [13] (see appendix D of the full version [14]).[2]

As a natural extension, we investigate the issue of incorporating CAPTCHA puzzles in the UC framework introduced by Canetti [10]. The situation turns out to be very sensitive to the modelling of CAPTCHA puzzles in the UC framework. We discuss two different ways of incorporating CAPTCHA puzzles in the UC framework: [3]

- INDIRECT ACCESS MODEL: In this model, the environment $\mathcal{Z}$ is *not* given direct access to a human H. Instead, the environment is given access to H *only through the adversary* $\mathcal{A}$. This model was proposed in the work of Canetti et. al. [4], who constructed a UC-secure protocol for password-based key-generation functionality. We call this model the *indirect* access model.
- DIRECT ACCESS MODEL: In this model, the environment is given a direct access to H. In particular, the queries made by $\mathcal{Z}$ to H are not visible to the adversary $\mathcal{A}$, in this model.

In the indirect access model, we show how to construct UC-secure protocols for all functionalities. In section 5, as the second main result of this work, we

[2] For readers familiar with concurrent non-malleability, our protocol admits "straight-line" simulation, but the extraction of witnesses from a man-in-the-middle is not straight-line. Also, another modification is needed to the protocol of [13]: we need to use a constant round non-malleable commitment scheme and not that of [15]. We can use any of the schemes presented in [16–19].

[3] We assume basic familiarity with the model of universal composition, and briefly recall it in appendix C.1 of the full version [14] .

construct a constant-round *UC-puzzle* protocol as defined by Lin, Pass, and Venkitasubramaniam [20]. By the results of [20], UC-puzzles are sufficient to obtain UC-secure protocols for general functionalities. Our protocol for UC-puzzles is obtained by combining our commitment scheme with a "cut-and-choose" protocol and (standard) zero-knowledge proofs for **NP** [21, 22].

In contrast, in the direct access model, it is easy to show that UC-secure computation is impossible for most functionalities. A formal statement is obtained by essentially reproducing the Canetti-Fischlin impossibility result for UC-commitments [23] (details reproduced in appendix E.1 of the full version [14]). The situation turns out to be the same for concurrent self-composition of two-party protocols: by reproducing the steps of Lindell's impossibility results [24, 25], concurrent self-composition in this model can be shown equivalent to universal composition. This means that secure computation of (most) functionalities in the concurrent self-composition model is impossible even with CAPTCHA puzzles.

On modelling CAPTCHA *puzzles in the UC framework.* The fact that UC-computation is possible in the indirect access model but concurrent self-composition is impossible raises the question whether indirect access model is the "right" model. What does a positive result in this model mean? To understand this, let us compare the indirect access model to the other "trusted setup" models such as the Common-Random-String (CRS) model [26]. In the CRS-model, the simulator $\mathcal{S}$ is in control of generating the CRS in the ideal world—this enables $\mathcal{S}$ to have a "trapdoor" to continue its actions without having to "rewind" the environment. We can view the indirect access model as some sort of a setup (i.e., access to H) controlled by the simulator in the ideal world. The fact that $\mathcal{S}$ can see the queries made by $\mathcal{Z}$ to H in the indirect-access-model, is then analogous to $\mathcal{S}$ controlling the CRS in the CRS-model. The only difference between these two settings is that the indirect-access-model does *not* require any trusted third party. viewed this way, the indirect-access-model can be seen as a "hybrid" model that stands somewhere between a trusted setup (such as the CRS model) and the plain model.

Beyond Conservative Adversaries. An inherent difficulty when dealing with CAPTCHA puzzles, is that of *malleability*. Informally, this means that given a challenge puzzle z, it might be possible for an algorithm $\mathcal{A}$ to efficiently generate a new puzzle z' such that given the solution of z', $\mathcal{A}$ can efficiently solve z. Such a malleability attack makes it difficult to reduce the security of a cryptographic scheme to the "hardness" of solving CAPTCHA puzzles.

To overcome this, previous works [3, 4] only prove security against a very restricted class of adversaries called *conservative* adversaries. Such adversaries are essentially those who do not launch the 'malleability' attack: that is, they only query H on CAPTCHA instances that are provided to them by the system. In both of these works, it is possible that a PPT adversary, on input a puzzle z may produce a puzzle z' such that the solutions of z and z' are related. But both works consider only restricted adversaries which are prohibited from querying H

with such a mauled puzzle z'. As noted in [3, 4], this an unreasonable restriction, especially knowing that CAPTCHA puzzles are in fact easily malleable.

In contrast, in this work, we prove the security of our schemes against the standard class of all probabilistic polynomial time (PPT) adversaries. The key-idea that enables us to go beyond the class of conservative adversaries is the formulation of the notion of an *human-extractable* CAPTCHA puzzle. Informally speaking, an human-extractable CAPTCHA puzzle, has the following property: suppose that a PPT algorithm $\mathcal{A}$ can solve a challenge puzzle z, and makes queries $\bar{q}$ to the human H during this process; then there is a PPT algorithm which on input the queries $\bar{q}$, can distinguish with the help of the human that $\bar{q}$ are correlated to z and not to some other randomly generated puzzle, say z''.

We discuss this notion at length in section 3, and many other issues related to formalizing CAPTCHA puzzles. This section essentially builds and improves upon previous works of [1, 3, 4] to give a unified framework for working with CAPTCHA puzzles. We view the notion of human-extractable CAPTCHA puzzles as an important contribution to prove security beyond the class of conservative adversaries.

2 Preliminaries

In this work, to model "access to a human", we will provide some parties (modeled as interactive Turing machines–ITM) *oracle* access to a function H. An ITM M with oracle access to H is an ordinary ITM except that it has two special tapes: a write-only *query tape* and a read-only *answer tape*. When M writes a string q on its query tape, the value $H(q)$ is written on its answer tape. If q is not a valid query (i.e., not in the domain of H), a special symbol $\perp$ is written on the output tape. Such a query and answer step is counted as one step in the running time of M. We use the notation M^H to mean that M has oracle access to H. The reader is referred to [27, 28] for a detailed treatment of this notion.

Notation. The output of an oracle ITM M^H is denoted by a triplet $(\mathtt{out}, \bar{q}, \bar{a})$ where $\mathtt{out}, \bar{q}$, and $\bar{a}$ denote the contents of M's output tape, a vector of strings written to the query tape in the current execution, and the answer to the queries present in $\bar{q}$ respectively.

Let $k \in \mathbb{N}$ denote the security parameter, where $\mathbb{N}$ is the set of natural numbers. All parties are assumed to receive 1^k as an implicit input (even if not mentioned explicitly). When we say that an (I)TM M (perhaps with access to an oracle H) runs in polynomial time, we mean that there exists a polynomial $T(\cdot)$ such that for every input, the total number of steps taken by M are at most $T(k)$. For two strings a and b, their concatenation is denoted by $a \circ b$. The statistical distance between two distributions X, Y is denoted $\Delta(X, Y)$.

In all places, we only use standard notations (with their usual meaning) for describing algorithms, random variables, experiments, protocol transcripts and so on. We assume familiarity with standard concepts such as computational indistinguishability, negligible functions, and so on (see [27]).

Statistically Secure Oblivious Transfer We now recall the notion of a statistically secure, two message oblivious transfer (OT) protocol, as defined by Halevi and Kalai [29].

Definition 1. (Statistically Secure Oblivious Transfer), [29] *Let $\ell(\cdot)$ be a polynomial and $k \in \mathbb{N}$ the security parameter. A two-message, two-party protocol $\langle S_{\mathrm{OT}}, R_{\mathrm{OT}} \rangle$ is said to be a* statistically secure oblivious transfer *protocol for bit-strings of length $\ell(k)$ such that both the sender S_{OT} and the receiver R_{OT} are* PPT *ITMs receiving 1^k as common input; in addition, S_{OT} gets as input two strings $(m_0, m_1) \in \{0,1\}^{\ell(k)} \times \{0,1\}^{\ell(k)}$ and R_{OT} gets as input a choice bit $b \in \{0,1\}$. We require that the following conditions are satisfied:*

- Functionality: *If the sender and the receiver follow the protocol then for every $k \in \mathbb{N}$, every $(m_0, m_1) \in \{0,1\}^{\ell(k)} \times \{0,1\}^{\ell(k)}$, and every $b \in \{0,1\}$, the receiver outputs m_b.*
- Receiver security: *The ensembles $\{R_{\mathrm{OT}}(1^k, 0)\}_{k\in\mathbb{N}}$ and $\{R_{\mathrm{OT}}(1^k, 1)\}_{k\in\mathbb{N}}$ are computationally indistinguishable, where $\{R_{\mathrm{OT}}(1^k, b)\}_{k\in\mathbb{N}}$ denotes the (first and only) message sent by R_{OT} on input $(1^k, b)$. That is,*

$$\{R_{\mathrm{OT}}(1^k, 0)\}_{k\in\mathbb{N}} \stackrel{c}{\equiv} \{R_{\mathrm{OT}}(1^k, 1)\}_{k\in\mathbb{N}}$$

- Sender security: *There exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for every $(m_0, m_1) \in \{0,1\}^{\ell(k)} \times \{0,1\}^{\ell(k)}$, every first message $\alpha \in \{0,1\}^*$ (from an arbitrary and possibly unbounded malicious receiver), and every sufficiently large $k \in \mathbb{N}$, it holds that either*

$$\Delta_0(k) := \Delta(S_{\mathrm{OT}}(1^k, m_0, m_1, \alpha), S_{\mathrm{OT}}(1^k, m_0, 0^{\ell(k)}, \alpha)) \text{ or,}$$

$$\Delta_1(k) := \Delta(S_{\mathrm{OT}}(1^k, m_0, m_1, \alpha), S_{\mathrm{OT}}(1^k, 0^{\ell(k)}, m_1, \alpha))$$

is negligible, where $S_{\mathrm{OT}}(1^k, m_0, m_1, \alpha)$ denotes the (only) response of the honest sender S_{OT} with input $(1^k, m_0, m_1)$ when the receiver's first message is α.

Statistically secure OT can be constructed from a vareity of cryptographic assumptions. In [29], Halevi and Kalai construct protocols satisfying the above definition under the assumption that *verifiable smooth projective hash families with hard subset membership problem* exist (which in turn, can be constructed from a variety of standard assumptions such as the quadratic-residue problem). [30] show the equivalence of 2-message statistically secure oblivious transfer and lossy encryption.

3 Modeling Captcha Puzzles

As said earlier, CAPTCHA puzzles are problem instances that are easy for "humans" but hard for computers to solve. Let us first consider the "hardness" of such puzzles for computers. To model "hardness," one approach is to consider an

asymptotic formulation. That is, we envision a randomized generation algorithm G which on input a security parameter 1^k, outputs a puzzle from a (discrete and finite) set $\mathcal{P}_k$ called the *puzzle-space*. Indeed, this is the formulation that previous works [1, 3, 4] as well as our work here follow. assume that there is a fixed polynomial $\ell(\cdot)$ such that every puzzle instance $z \in \mathcal{P}_k$ is a bit string of length at most $\ell(k)$.

Of course, not all CAPTCHA puzzle systems satisfy such an asymptotic formulation. It is possible to have a (natural) non-asymptotic formulation to define CAPTCHA puzzles which takes into consideration this issue and defines hardness in terms of a "human population" [1]. However, a non-asymptotic formulation will be insufficient for cryptographic purposes. For many puzzles, typically hardness can be amplified by sequential or parallel repetition[31].

Usually, CAPTCHA puzzles have a unique and well defined solution associated with every puzzle instance. We capture this by introducing a discrete and finite set $\mathcal{S}_k$, called the *solution-space*, and a corresponding *solution function* $H_k : \mathcal{P}_k \to \mathcal{S}_k$ which maps a puzzle instance $z \in \mathcal{P}_k$ to its corresponding solution. Without loss of generality we assume that every element of $\mathcal{S}_k$ is a bit string of length k. We will require that G generates puzzles together with their solutions. This restriction is also required in previous works [1, 3]. To facilitate the idea that the puzzle-generation is a completely *automated* process, G will not be given "access to a human."

With this formulation, we can view "humans" as computational devices which can "efficiently" compute the solution function H_k. Therefore, to capture "access to a human", the algorithms can simply be provided with *oracle* access to the family of solution functions $H := \{H_k\}_{k\in\mathbb{N}}$. Recall that by definition, oracle-access to H means that algorithms can only provide an input z to some function $H_{k'}$ in the family H, and then read its output $H_{k'}(z)$; if z is not in the domain $\mathcal{P}_{k'}$, the response to the query is set to a special symbol, denoted $\perp$. Every query to $H_{k'}$ will be assumed to contribute one step to the running time of the querying algorithm. The discussion so far leads to the following definition for CAPTCHA puzzles.

Definition 2. (Captcha Puzzles) *Let $\ell(\cdot)$ be a polynomial, and $\mathcal{S} := \{\mathcal{S}_k\}_{k\in\mathbb{N}}$ and $\mathcal{P} := \{\mathcal{P}_k\}_{k\in\mathbb{N}}$ be such that $\mathcal{P}_k \subseteq \{0,1\}^{\ell(k)}$ and $\mathcal{S}_k \subseteq \{0,1\}^k$. A* CAPTCHA *puzzle system $\mathcal{C} := (G, H)$ over $(\mathcal{P}, \mathcal{S})$ is a pair such that G is a randomized polynomial time turing machine, called the* generation algorithm*, and $H := \{H_k\}_{k\in\mathbb{N}}$ is a collection of* solution functions *such that $H_k : \mathcal{P}_k \to \mathcal{S}_k$. Algorithm G, on input a security parameter $k \in \mathbb{N}$, outputs a tuple $(z, a) \in \mathcal{P}_k \times \mathcal{S}_k$ such that $H_k(z) = a$. We require that there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for every* PPT *algorithm $\mathcal{A}$, and every sufficiently large $k \in \mathbb{N}$, we have that:*

$$p_{inv}(k) := \Pr\left[(z, a) \leftarrow G(1^k); \mathcal{A}(1^k, z) = a\right] \leq \mathrm{negl}(k)$$

where the probability is taken over the randomness of both G and $\mathcal{A}$.

Turing Machines vs Oracle Turing Machines. We emphasize that the CAPTCHA puzzle generation algorithm G is an ordinary turing machine with no access to

any oracles. Furthermore, the security of a CAPTCHA system holds *only* against PPT adversaries $\mathcal{A}$ who are turing machines. It *does not* hold against oracle turing machines with oracle access to H. However, we use CAPTCHA systems defined as above in protocols which guarantee security against adversaries who may even have access to the oracle H. This distinction between machines which have access to an (human) oracle and machines which don't occurs throughout the text.

The Issue of Malleability. As noted earlier, CAPTCHA puzzles are usually easily *malleable* [15]. That is, given a challenge puzzle z, it might be possible for an algorithm $\mathcal{A}$ to efficiently generate a new puzzle $z' \neq z$ such that given the solution of z', $\mathcal{A}$ can efficiently solve z. It turns out that in all previous works this creates several difficulties in the security proofs. In particular, in reducing the "security" of a cryptographic scheme to the "hardness" of the CAPTCHA puzzle, it becomes unclear how to handle such an adversary.

Due to this, previous works [3, 4] only prove security against a very restricted class of adversaries called the *conservative* adversaries. Such adversaries are essentially those who do not query H_k on any CAPTCHA instances other than the ones that are provided to them by the system. To facilitate a proof against all PPT adversaries, we develop the notion of human-extractable CAPTCHA puzzles below.

Human-Extractable CAPTCHA *Puzzles.* The notion of human-extractable CAPTCHA puzzles stems from the intuition that if a PPT algorithm $\mathcal{A}$ can solve a random instance z produced by G, then it must make queries $\bar{q} = (q_1, q_2, \ldots)$ to (functions in) H that contain sufficient information about z. More formally, suppose that z_1 and z_2 are generated by two random and independent executions of G. If $\mathcal{A}$ is given z_1 as input and it produces the correct solution, then the queries $\bar{q}$ will contain sufficient information about z_1 and no information about z_2 (since z_2 is independent of z_1 and never seen by $\mathcal{A}$). Therefore, by looking at the queries $\bar{q}$, it should be possible with the help of the human to deduce which of the two instances is solved by $\mathcal{A}$. We say that a CAPTCHA puzzle system is human-extractable if there exists a PPT algorithm Extr which, by looking at the queries $\bar{q}$, can tell with the help of the human which of the two instances was solved by $\mathcal{A}$. The formal definition follows; recall the convention that output of oracle Turing machines includes the queries $\bar{q}$ they make to H and corresponding answers $\bar{a}$ received.

Definition 3. (Human-extractable Captcha) *A* CAPTCHA *puzzle system* $\mathcal{C} := (G, H)$ *is said to be* human-extractable *if there exists an oracle* PPT *algorithm* Extr^H*, called the* extractor*, and a negligible function* $\mathrm{negl}(\cdot)$*, such that for every oracle* PPT *algorithm* $\mathcal{A}^H$*, and every sufficiently large* $k \in \mathbb{N}$*, we have that:*

$$p_{\mathit{fail}}(k) := \Pr\left[\begin{array}{l}(z_0, s_0) \leftarrow G(1^k); (z_1, s_1) \leftarrow G(1^k); b \xleftarrow{\$} \{0,1\}; \\ (s, \bar{q}, \bar{a}) \leftarrow \mathcal{A}^H(1^k, z_b); b' \leftarrow \mathsf{Extr}^H(1^k, (z_0, z_1), \bar{q}); \\ s = s_b \wedge b' \neq b\end{array}\right] \leq \mathrm{negl}(k)$$

where the probability is taken over the randomness of $G, \mathcal{A}$*, and* Extr*.*

Observe that except with negligible probability, $s_0 \neq s_1$, since otherwise one can break the hardness of $\mathcal{C}$(definition 2).

We believe that the notion of human-extractable CAPTCHA puzzles is a very natural notion; it may be of independent interest and find applications elsewhere. We note that while assuming the existence of human-extractable CAPTCHA puzzles may be a strong assumption, it is very different from the usual extractability assumptions in the literature such as the Knowledge-Of-Exponent (KOE) assumption [32, 33]. In particular, often it might be possible to empirically test whether a given CAPTCHA system is human-extractable. For example, one approach for such a test is to just ask sufficiently many humans to correlate the queries $\bar{q}$ to one of the puzzles z_0 or z_1. If sufficiently many humans can correctly correlate $\bar{q}$ to z_b with probability noticeably better than 1/2, one can already conclude some form of weak extraction. Such weak extractability can then be amplified by using techniques from parallel repetition. In contrast, there is no such hope for KOE assumption (and other problems with similar "non-black-box" flavor) since they are not falsifiable [34].

In this work, we only concern ourselves with human-extractable CAPTCHA puzzles. Thus we drop the adjective human-extractable as convenient.

Drawbacks of Our Approach and Other Considerations. While our framework significantly improves upon previous works [3, 4], it still has certain drawbacks which are impossible to eliminate in an asymptotic formulation such as ours.

The first drawback is that as the value of k increases, the solution becomes larger. It is not clear if the humans can consistently answer such a long solution. Therefore, such a formulation can become unsuitable for even very small values of k. The second drawback is that the current formulation enforces strict "rules" on how a human and a Turing machine communicate via oracle access to H. This does not capture "malicious" humans who can communicate with their computers in arbitrary ways. It is not even clear how to formally define such "malicious" humans for our purpose.

Finally, definition 2 enforces the condition that $|\mathcal{S}_k|$ is super-polynomial in k. For many CAPTCHA puzzle systems in use today, $|\mathcal{S}_k|$ may be small (e.g., polynomial in k or even a constant). Such CAPTCHA puzzles are not directly usable in our setting. Observe that if $|\mathcal{S}_k|$ is small, clearly $\mathcal{A}$ can solve a given challenge puzzle with noticeable probability. Therefore, it makes sense to consider the following weaker variant in definition 2: instead of requiring $\mathrm{p}_{\mathsf{inv}}$ to be negligible, we can consider it to be a small constant ϵ. Likewise, we can also consider weakening the extractability condition by in definition 3 by requiring $\mathrm{p}_{\mathsf{fail}}$ to be only noticeably better than 1/2.

A subtle point to observe here is that while it might be possible to *individually* amplify $\mathrm{p}_{\mathsf{inv}}$ and $\mathrm{p}_{\mathsf{fail}}$ by using parallel or sequential repetitions, it may not be possible to amplify *both at the same time*. Indeed, when $|\mathcal{S}_k|$ is small, the adversary $\mathcal{A}$ can simply ask one CAPTCHA puzzle for every solution $a \in \mathcal{S}_k$ multiple times and "hide" the challenge puzzle z_b (in some mauled form z'_b) somewhere in this large list of queries. Such a list of queries might have sufficient correlation with *both* z_0 and z_1 simply because the solutions of these both are

in $\mathcal{S}_k$ and $\mathcal{A}$ has asked at least one puzzle for each solution in the whole space. In this case, even though parallel repetition may amplify $\mathrm{p}_{\mathsf{inv}}$, extraction might completely fail because the correlation corresponding to the challenge puzzle is not easy to observe in $\mathcal{A}$'s queries and answers.

As a consequence of this, our formulation essentially rules out the possibility of using such "weak" CAPTCHA puzzles for which both $\mathrm{p}_{\mathsf{inv}}$ and $\mathrm{p}_{\mathsf{fail}}$ are not suitable. This is admittedly a strong limitation, which seems to come at the cost of proving security beyond the class of conservative adversaries.

4 A Straight-Line Extractable Commitment Scheme

In this section we present a straight-line extractable commitment scheme which uses human-extractable CAPTCHA puzzles. The hiding and binding properties of this commitment scheme rely on standard cryptographic assumptions, and the straight-line extraction property relies on the extraction property of CAPTCHA puzzles.

We briefly recall the notion of secure commitment schemes, with emphasis on the changes from the standard definition and then define the notion of straight-line extractable commitments.

Commitment Schemes. First, we present a definition of commitment schemes augmented with CAPTCHA puzzles. Let $\mathcal{C} := (G, H)$ be a CAPTCHA puzzle system, and let $\mathrm{Com}_{\mathcal{C}} := \langle \mathsf{C}^H, \mathsf{R} \rangle$ be a two-party interactive protocol where (only) C has oracle access to the solution function family H[4]. We say that $\mathrm{Com}_{\mathcal{C}}$ is a commitment scheme if: both C and R are PPT (interactive) TM receiving 1^k as the common input; in addition, C receives a string $m \in \{0,1\}^k$. Further, we require C to *privately* output a *decommitment* string d, and R to *privately* output an auxiliary string aux. The transcript of the interaction is called the *commitment* string, denoted by c. During the course of the interaction, let $\bar{q}$ and $\bar{a}$ be the queries and answers obtained by C via queries to the CAPTCHA oracle H. To denote the sampling of an honest execution of $\mathrm{Com}_{\mathcal{C}}$, we use the following notation: $(c, (d, \bar{q}, \bar{a}), \mathsf{aux}) \leftarrow \langle \mathsf{C}^H(1^k, m), \mathsf{R}(1^k) \rangle$.

Notice that $(d, \bar{q}, \bar{a})$ is the output of oracle ITM C^H as defined in section 2. For convenience, we associate a polynomial time algorithm DCom which on input (c, d, aux) either outputs a message m, or $\perp$. It is required that for all honest executions where C commits to m, DCom always outputs m. We say that $\mathrm{Com}_{\mathcal{C}}$ is an *ordinary* commitment scheme if $\bar{q}$ (and hence $\bar{a}$) is an empty string.

Furthermore, our definition of a commitment scheme allows for stateful commitments. In particular the output aux might be necessary for a successful decommitment of the committed message.

[4] The reason we do not provide R with access to H, is because our construction does not need it, and therefore we would like to avoid cluttering the notation. In general, however, both parties can have access to H. Also, in our adversarial model, we consider all malicious receivers to have access to the oracle H

We assume that the reader is familiar with perfect/statistical binding and computational hiding properties of a commitment scheme. Informally, straight-line extraction property means that there exists an extractor $\mathsf{ComExtr}^H$ which on input the commitment string c (possibly from an interaction with a malicious committer), aux (from an honest receiver), and $\bar{q}$, outputs the committed message m (if one exists), except with negligible probability. If m is not well defined, there is no guarantee about the output of $\mathsf{ComExtr}$.

For any commitment, we use $\mathcal{M} = \mathcal{M}(c, \mathsf{aux})$ to denote a possible decommitment message defined by the commitment string c and the receiver state aux. If such a message is not well defined (say there could be multiple such messages or none at all) for a particular (c, aux), then define $\mathcal{M}(c, \mathsf{aux}) = \perp$.

Definition 4. (Straight-line Extractable Commitment) *A statistically-binding computationally-hiding commitment scheme* $\mathrm{Com}_{\mathcal{C}} := \langle \mathsf{C}^H, \mathsf{R} \rangle$ *defined over a human-extractable* CAPTCHA *puzzle system* $\mathcal{C} := (G, H)$ *is said to admit* straight-line extraction *if there exists a* PPT *algorithm* $\mathsf{ComExtr}^H$ *(the* extractor*) and a negligible function* $\mathrm{negl}(\cdot)$*, such that for every* PPT *algorithm* $\widehat{\mathsf{C}}$ *(a* malicious *committer whose input could be arbitrary), and every sufficiently large* $k \in \mathbb{N}$*, we have that:*

$$\Pr\left[\begin{array}{l}(c^*, (d^*, \bar{q}, \bar{a}), \mathsf{aux}) \leftarrow \langle \widehat{\mathsf{C}}^H(1^k, \cdot), \mathsf{R}(1^k) \rangle; \mathcal{M} = \mathcal{M}(c^*, aux); \\ m \leftarrow \mathsf{ComExtr}^H(1^k, \bar{q}, (c^*, \mathsf{aux})) : (\mathcal{M} \neq \perp) \wedge (m \neq \mathcal{M})\end{array}\right] \leq \mathrm{negl}(k)$$

where the probability is taken over the randomness of $\widehat{\mathsf{C}}, \mathsf{R}$*, and* $\mathsf{ComExtr}$*.*

The Commitment Protocol. At a high level, the receiver R of our protocol will choose two CAPTCHA puzzles (z^0, z^1) (along with their solutions s^0, s^1). To commit to bit b, the sender C will select z^b using the OT protocol and commit to its solution s^b using an ordinary (perfectly-binding) commitment scheme $\langle C_{\mathrm{PB}}, R_{\mathrm{PB}} \rangle$. The solution to the puzzle is obtained by querying H on z^b. To decommit, first decommit to s^b which the receiver verifies; and then the receiver accepts b as the decommitted bit if the solution it received is equal to s^b. To facilitate this task, the receiver outputs an auxiliary string aux which contains (z^0, z^1, s^0, s^1). To commit to a k-bit string $m \in \{0,1\}^k$, this atomic protocol is repeated in *parallel* k-times (with some minor modifications as in Figure 1)

For convenience we assume that $\langle C_{\mathrm{PB}}, R_{\mathrm{PB}} \rangle$ is *non-interactive* (i.e., C sends only one message to R) for committing strings of length k^2. The decommitment string then consists of the committed messages and the randomness of C_{PB}. The formal description of our protocol appears in figure 1.

Theorem 5. *Assume that* $\langle C_{\mathrm{PB}}, R_{\mathrm{PB}} \rangle$ *is an ordinary, non-interactive, perfectly-binding and computationally-hiding commitment scheme,* $\mathcal{C} = (G, H)$ *is a human-extractable* CAPTCHA *puzzle system, and* $\langle S_{\mathrm{OT}}, R_{\mathrm{OT}} \rangle$ *is a two-round statistically-secure oblivious transfer protocol. Then, protocol* $\mathrm{Com}_{\mathcal{C}} = \langle \mathsf{C}^H, \mathsf{R} \rangle$ *described in figure 1 is a 3-round perfectly-binding and computationally-hiding commitment scheme which admits straight-line extraction.*

Proof. A full proof may be found in Appendix A of the full version [14]. ∎

Let k be the security parameter, $\mathcal{C} := (G, H)$ a human-extractable CAPTCHA puzzle system, $\langle C_{\text{PB}}, R_{\text{PB}} \rangle$ a non-interactive perfectly-binding commitment scheme for strings of length k^2, and $\langle S_{\text{OT}}, R_{\text{OT}} \rangle$ a two-message two-party OT protocol.

Commitment. Let $m = (m_1, \ldots, m_k) \in \{0,1\}^k$ be the message to be committed.

1. CAPTCHA GENERATION: For every $i \in [k]$, R generates a pair of independent CAPTCHA puzzles: $(z_i^0, s_i^0) \leftarrow G(1^k)$ and $(z_i^1, s_i^1) \leftarrow G(1^k)$.
2. PARALLEL OT: C and R perform k parallel executions of OT, where the i^{th} execution proceeds as follows. Party R acts as the OT-sender S_{OT} on input (z_i^0, z_i^1) and party C acts the OT-receiver R_{OT} on input the bit m_i. At the end of the execution, let the puzzle instances obtained by C be $z_1^{m_1}, \ldots, z_k^{m_k}$.
3. COMMIT TO CAPTCHA SOLUTIONS: For every $i \in [k]$, C queries H_k on $z_i^{m_i}$ to obtain the solution $s_i^{m_i}$. Let $\bar{s} := s_1^{m_1} \circ \ldots \circ s_k^{m_k}$, which is of length k^2. C commits to $\bar{s}$ using protocol $\langle C_{\text{PB}}, R_{\text{PB}} \rangle$. Let r be the randomness used and c be the message sent by C in this step.
4. OUTPUTS: R sets $\mathsf{aux} = \{(z_i^0, z_i^1, s_i^0, s_i^1)\}_{i=1}^k$, and C sets $d = (\bar{s}, r)$.

Decommitment. On input the commitment transcript, and strings $d = (\bar{s}, r)$ and $\mathsf{aux} = \{(z_i^0, z_i^1, s_i^0, s_i^1)\}_{i=1}^k$ do the following: parse the transcript to obtain string c from the last step, and verify that $(\bar{s}, r)$ is a valid decommitment for c. If yes, parse $\bar{s} = a_1 \circ \ldots \circ a_k$ and test that for every $i \in [k]$, there exists a *unique* bit b_i such that $a_i = s_i^{b_i}$. If any test fails, output $\perp$; otherwise output $m = (b_1, \ldots, b_k)$.

Fig. 1. Straightline Extractable Commitment Protocol $\langle \mathsf{C}^H, \mathsf{R} \rangle$

5 Constructing UC-Puzzles Using Captcha

We provided a basic background in the section 1 to our results on protocol composition, and mentioned that there are two ways in which we can incorporate CAPTCHA puzzles in the UC-framework: the indirect access model, and the direct access model. This section is about constructing *UC puzzles* [20] in the indirect access model. Recall that in the indirect access model, the environment $\mathcal{Z}$ is not given direct access to a human (or the solution function family of the CAPTCHA system) H; instead, $\mathcal{Z}$ must access H exclusively through the adversary $\mathcal{A}$. This allows the simulator to look at the queries of $\mathcal{Z}$, which in turn allows for a positive result. Due to space constraints, we shall assume basic familiarity with the UC-framework [10], and directly work with the notion of UC-puzzles. A more detailed review of the UC framework, and concurrent composition, is given in appendix C of the full version [14] .

Lin, Pass and Venkitasubramaniam [20] defined the notion of a *UC puzzle*, and demonstrated that to obtain universal-composition in a particular model (e.g., the CRS model), it suffices to construct a UC puzzle in that model. We will

adopt this approach, and construct a UC puzzle using CAPTCHA. We recall the notion of a UC-puzzle with necessary details, and refer the reader to [20] for an extensive exposition. Our formulation directly incorporates CAPTCHA puzzles in the definition and hence does not refer to any setup $\mathcal{T}$; other than this semantic change, the description here is essentially identical to that of [20].

The UC-puzzle is a protocol which consists of two parties—a sender S, and a receiver R, and a PPT-relation $\mathcal{R}$. Let $\mathcal{C} := (G, H)$ be a CAPTCHA puzzle system. Only the sender will be given oracle access to H, and the resulting protocol will be denoted by $\langle S^H, R\rangle$. Informally, we want that the protocol be *sound*: no efficient receiver R^* can successfully complete an interaction with S and also obtain a "trapdoor" y such that $\mathcal{R}(\mathsf{TRANS}, y) = 1$, where TRANS is the transcript of that execution. We also require *statistical UC-simulation*: for every efficient adversary $\mathcal{A}$ participating as a sender in many executions of the protocol with multiple receivers $R_1, \ldots, R_m$, and communicating with an environment $\mathcal{Z}$ simultaneously, there exists a simulator Sim which can *statistically* simulate the view of $\mathcal{A}$ for $\mathcal{Z}$ and output trapdoors to all successfully completed puzzles at the same time.

Formally, we consider a concurrent execution of the protocol $\langle S^H, R\rangle$ for an adversary $\mathcal{A}$. In the concurrent execution, $\mathcal{A}$ exchanges messages with a puzzle-environment $\mathcal{Z}$ and participates as a sender concurrently in $m = \text{poly}(k)$ (puzzle)-protocols with honest receivers $R_1, \ldots, R_m$. At the onset of a execution, $\mathcal{Z}$ outputs a session identifier *sid* that all receivers receive as input. Thereafter, $\mathcal{Z}$ is allowed to exchange messages only with the adversary $\mathcal{A}$. In particular, for any queries to the CAPTCHA solving oracle, $\mathcal{Z}$ cannot query H; instead, it can send its queries to $\mathcal{A}$, who in turn, can query H for $\mathcal{Z}$, and report the answer back to $\mathcal{Z}$. We compare a real and an ideal execution.

REAL EXECUTION. The real execution consists of the adversary $\mathcal{A}$, which interacts with $\mathcal{Z}$, and participates as a sender in m concurrent interactions of $\langle S^H, R\rangle$. Further, the adversary and the honest receivers have access to H which they can query and receive the solutions over secure channels. The environment $\mathcal{Z}$ does not have access to H; it can query H, by sending its queries to $\mathcal{A}$, who queries H with the query and reports the answers back to $\mathcal{Z}$. Without loss of generality, we assume that after every interaction, $\mathcal{A}$ honestly sends TRANS to $\mathcal{Z}$, where TRANS is the transcript of execution. Let $\text{REAL}^H_{\mathcal{A},\mathcal{Z}}(k)$ be the random variable that describes the output of $\mathcal{Z}$ in the real execution.

IDEAL EXECUTION. The ideal execution consists of a PPT machine (the simulator) with oracle access to H, denoted Sim^H. On input 1^k, Sim^H interacts with the environment $\mathcal{Z}$. At the end of the execution, the environment produces an output. We denote the output of $\mathcal{Z}$ in the ideal execution by the random variable $\text{IDEAL}_{\mathsf{Sim}^H,\mathcal{Z}}(k)$.

Definition 6. (UC-Puzzle, adapted from [20]) *Let $\mathcal{C} := (G, H)$ be a* CAPTCHA *puzzle system. A pair $(\langle S^H, R\rangle, \mathcal{R})$ is called UC-puzzle for a polynomial*

time computable relation $\mathcal{R}$ *and the* CAPTCHA *puzzle system* $\mathcal{C}$*, if the following conditions hold:*

- SOUNDNESS. *There exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for every* PPT *receiver* $\mathcal{A}$*, and every sufficiently large* k*, the probability that* $\mathcal{A}$*, after an execution with the sender* S^H *on common input* 1^k*, outputs* y *such that* $y \in \mathcal{R}(\mathsf{TRANS})$ *where* TRANS *is the transcript of the message exchanged in the interaction, is at most* $\mathrm{negl}(k)$*.*
- STATISTICAL SIMULATION. *For every* PPT *adversary* $\mathcal{A}$ *participating in a concurrent puzzle execution, there exists an oracle* PPT *machine called the simulator,* Sim^H*, such that for every* PPT *environment* $\mathcal{Z}$ *and every sufficiently large* k*, the random variables* $\mathrm{REAL}^H_{\mathcal{A},\mathcal{Z}}(k)$ *and* $\mathrm{IDEAL}_{\mathsf{Sim}^H,\mathcal{Z}}(k)$ *are statistically close over* $k \in \mathbb{N}$*, and whenever* Sim *sends a message of the form* TRANS *to* $\mathcal{Z}$*, it outputs* y *in its special output tape such that* $y \in \mathcal{R}(\mathsf{TRANS})$*.*

The UC-puzzle System. Due to space constraints, here we only sketch the construction of our UC-puzzle, and defer the details to the full version [14]. A straightforward approach that does not quite work is to use our extractable commitment from Figure 1. That is, the sender of the UC puzzle picks random string s, which will serve as the trapdoor, and commits to it using our extractable commitment. Although this scheme allows extraction of the trapdoor s, it is not clear how, given a transcript and a purported trapdoor, it can be verified in PPT whether it is the correct trapdoor or not. Further, a malicious sender may commit to an invalid string (by using incorrect CAPTCHA solution, for example). The receiver can not detect this and will accept, while there is no well-defined trapdoor for such a transcript. Moreover, we can not use the standard trick of using zero-knowledge to enforce correct sender behaviour because checking validity of CAPTCHA solutions is not a PPT process.

We solve the first problem by making the sender additionally send $z := f(s)$ to the receiver, where $f(\cdot)$ is a one-way function. The idea is to make it easy to verify the trapdoor, by simply checking if z is the image of the trapdoor under $f(\cdot)$. However, for this to work, we must ensure that the pre-image of z and the string committed in the extractable commitment are the same.

To solve this problem, we use the following modified commitment scheme in the above protocol to commit to the trapdoor: to commit to a string s, the sender commits s twice, first using our straight line extractable commitment from Figure 1, and then using any non-interactive perfectly binding scheme $\langle C_{\mathrm{PB}}, R_{\mathrm{PB}} \rangle$ (which can be constructed from, for eg., one-way functions). Using this, we tackle the aforementioned problem in two steps:

1. First, the committer proves that the commitment is 'well-formed': that is, the string committed in both the commitments is the same. This is done by using secret sharing and cut-and-choose.
2. Thereafter, the committer gives a zero-knowledge proof that the string committed using the commitment scheme $\langle C_{\mathrm{PB}}, R_{\mathrm{PB}} \rangle$ is the same as the pre-image of z under $f(\cdot)$.

The first step ensures that the string committed using the commitment scheme of Figure 1 is the same as that committed by $\langle C_{\text{PB}}, R_{\text{PB}} \rangle$. As $\langle C_{\text{PB}}, R_{\text{PB}} \rangle$ is in the plain model and does not involve CAPTCHA, we can give a proof of correctness using standard zero-knowledge. For full details, please refer to the full version [14].

6 Conclusion

Open Questions and Future Work. Our work presents a basic technique using human-extractable CAPTCHA puzzles to enable straight-line extraction and shows how to incorporate it into the framework of protocol composition to obtain new and interesting feasibility results. However, many other important questions remain to be answered. For examples, can we obtain zero-knowledge proofs for **NP** in 3 or less rounds?[5] Can we obtain plain-text aware encryption-schemes? What about *non-interactive* non-malleable commitments *without* setup [15, 17, 35, 36]?

One interesting direction is to consider improving upon the recent work of Goyal, Jain, and Ostrovsky on generating a password-based session-keys in the concurrent setting [37]. One of the main difficulties in [37] is to get a control on the number of times the simulator rewinds any given session. They accomplish this by using the technique of precise-simulation [38, 39]. However, since we obtain straight-line simulation, it seems likely that our techniques could be used to improve the results in [37]. The reason we are not able to do this is that our techniques are limited to only simulation—they do *not* yield both straight-line simulation and extraction, whereas [37] needs a control over both.

Another interesting direction is to explore the design of extractable CAPTCHA puzzles. In general, investigating the feasibility and drawbacks of the asymptotic formulation for CAPTCHA puzzles presented here and in [1, 3, 4] is an interesting question in its own right. We presented a discussion of these details in section 3, however they still present numerous questions for future work.

References

1. von Ahn, L., Blum, M., Hopper, N.J., Langford, J.: CAPTCHA: Using Hard AI Problems for Security. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 294–311. Springer, Heidelberg (2003)
2. The Official CAPTCHA Site, `http://www.captcha.net`
3. Dziembowski, S.: How to Pair with a Human. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 200–218. Springer, Heidelberg (2010)
4. Canetti, R., Halevi, S., Steiner, M.: Mitigating Dictionary Attacks on Password-Protected Local Storage. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 160–179. Springer, Heidelberg (2006)

[5] By using standard techniques, e.g., coin-tossing using our commitment scheme along with Blum's protocol [22], we can obtain a 5-round (concurrent) zero-knowledge protocol. But we do not know how to reduce it to 3 rounds.

5. Bellare, M., Rogaway, P.: Optimal Asymmetric Encryption. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995)
6. Dwork, C., Naor, M., Sahai, A.: Concurrent zero-knowledge. In: STOC, pp. 409–418 (1998)
7. Lindell, Y.: Bounded-concurrent secure two-party computation without setup assumptions. In: STOC, pp. 683–692 (2003)
8. Pass, R., Rosen, A.: Bounded-concurrent secure two-party computation in a constant number of rounds. In: FOCS (2003)
9. Pass, R.: Bounded-concurrent secure multi-party computation with a dishonest majority. In: STOC, pp. 232–241 (2004)
10. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: FOCS, pp. 136–145 (2001)
11. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: STOC, pp. 494–503 (2002)
12. Prabhakaran, M., Rosen, A., Sahai, A.: Concurrent zero knowledge with logarithmic round-complexity. In: FOCS, pp. 366–375 (2002)
13. Barak, B., Prabhakaran, M., Sahai, A.: Concurrent non-malleable zero knowledge. In: FOCS, pp. 345–354. IEEE Computer Society (2006)
14. Kumarasubramanian, A., Ostrovsky, R., Pandey, O., Wadia, A.: Cryptography using captcha puzzles. Cryptology ePrint Archive, Report 2012/689 (2012), `http://eprint.iacr.org/`
15. Dolev, D., Dwork, C., Naor, M.: Non-Malleable Cryptography. SIAM J. on Computing 30(2), 391–437 (2000)
16. Pass, R., Rosen, A.: New and improved constructions of non-malleable cryptographic protocols. In: STOC, pp. 533–542 (2005)
17. Pandey, O., Pass, R., Vaikuntanathan, V.: Adaptive One-Way Functions and Applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 57–74. Springer, Heidelberg (2008)
18. Lin, H., Pass, R.: Constant-round non-malleable commitments from any one-way function. In: STOC, pp. 705–714 (2011)
19. Goyal, V.: Constant round non-malleable protocols using one way functions. In: STOC, pp. 695–704 (2011)
20. Lin, H., Pass, R., Venkitasubramaniam, M.: A unified framework for concurrent security: universal composability from stand-alone non-malleability. In: STOC, pp. 179–188 (2009)
21. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract). In: FOCS, pp. 174–187 (1986)
22. Blum, M.: How to prove a theorem so no one else can claim it. In: International Congress of Mathematicians, pp. 1444–1451 (1987)
23. Canetti, R., Fischlin, M.: Universally Composable Commitments. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 19–40. Springer, Heidelberg (2001)
24. Lindell, Y.: General composition and universal composability in secure multi-party computation. In: 44th FOCS, pp. 394–403 (2003)
25. Lindell, Y.: Lower bounds and impossibility results for concurrent self composition. Journal of Cryptology 21, 200–249 (2008), doi:10.1007/s00145-007-9015-5
26. Blum, M., Santis, A.D., Micali, S., Persiano, G.: Noninteractive zero-knowledge. SIAM J. Comput. 20(6), 1084–1118 (1991)
27. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press (2001)

28. Arora, S., Barak, B.: Computational Complexity: A Modern Approach. Cambridge University Press (2009)
29. Halevi, S., Kalai, Y.: Smooth projective hashing and two-message oblivious transfer. Journal of Cryptology, 1–36 (2010), doi:10.1007/s00145-010-9092-8
30. Hemenway, B., Ostrovsky, R.: Lossy trapdoor functions from smooth homomorphic hash proof systems. Electronic Colloquium on Computational Complexity (ECCC) 16, 127 (2009)
31. Canetti, R., Halevi, S., Steiner, M.: Hardness Amplification of Weakly Verifiable Puzzles. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 17–33. Springer, Heidelberg (2005)
32. Hada, S., Tanaka, T.: On the Existence of 3-Round Zero-Knowledge Protocols. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 408–423. Springer, Heidelberg (1998)
33. Bellare, M., Palacio, A.: The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 273–289. Springer, Heidelberg (2004)
34. Naor, M.: On Cryptographic Assumptions and Challenges. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 96–109. Springer, Heidelberg (2003)
35. Di Crescenzo, G., Ishai, Y., Ostrovsky, R.: Non-interactive and non-malleable commitment. In: STOC, pp. 141–150 (1998)
36. Di Crescenzo, G., Katz, J., Ostrovsky, R., Smith, A.: Efficient and Non-interactive Non-malleable Commitment. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 40–59. Springer, Heidelberg (2001)
37. Goyal, V., Jain, A., Ostrovsky, R.: Password-Authenticated Session-Key Generation on the Internet in the Plain Model. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 277–294. Springer, Heidelberg (2010)
38. Micali, S., Pass, R.: Local zero knowledge. In: STOC, pp. 306–315 (2006)
39. Pandey, O., Pass, R., Sahai, A., Tseng, W.-L.D., Venkitasubramaniam, M.: Precise Concurrent Zero Knowledge. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 397–414. Springer, Heidelberg (2008)

Improved Zero-Knowledge Proofs of Knowledge for the ISIS Problem, and Applications

San Ling[1], Khoa Nguyen[1], Damien Stehlé[2], and Huaxiong Wang[1]

[1] Division of Mathematical Sciences,
School of Physical and Mathematical Sciences,
Nanyang Technological University, Singapore
{lingsan,nguy0106,hxwang}@ntu.edu.sg
[2] ÉNS de Lyon, Laboratoire LIP (U. Lyon, CNRS, ENSL, INRIA, UCBL),
46 Allée d'Italie, 69364 Lyon Cedex 07, France
damien.stehle@ens-lyon.fr

Abstract. In all existing efficient proofs of knowledge of a solution to the infinity norm Inhomogeneous Small Integer Solution (ISIS$^\infty$) problem, the knowledge extractor outputs a solution vector that is only guaranteed to be $\widetilde{O}(n)$ times longer than the witness possessed by the prover. As a consequence, in many cryptographic schemes that use these proof systems as building blocks, there exists a gap between the hardness of solving the underlying ISIS$^\infty$ problem and the hardness underlying the security reductions. In this paper, we generalize Stern's protocol to obtain two statistical zero-knowledge proofs of knowledge for the ISIS$^\infty$ problem that remove this gap. Our result yields the potential of relying on weaker security assumptions for various lattice-based cryptographic constructions. As applications of our proof system, we introduce a concurrently secure identity-based identification scheme based on the worst-case hardness of the $\mathrm{SIVP}_{\widetilde{O}(n^{1.5})}$ problem (in the ℓ_2 norm) in general lattices in the random oracle model, and an efficient statistical zero-knowledge proof of plaintext knowledge with small constant gap factor for Regev's encryption scheme.

1 Introduction

Zero-knowledge proofs and proofs of knowledge are fundamental notions and powerful tools in cryptography. In a zero-knowledge proof system [GMR89], a prover convinces a verifier that some statement is true while leaking nothing but the validity of the assertion. In a proof of knowledge ([GMR89, BG93]), the prover also convinces the verifier that he indeed knows a satisfying "witness" for the given statement. In the last 25 years, zero-knowledge proofs of knowledge (**ZKPoK**) have been extensively studied ([FFS87, GQ90, FS89, RS92, Mau09],...). These proof systems are the building blocks in many cryptographic constructions (e.g., identification schemes, group signatures, anonymous credential systems, to name just a few). In this work, we focus on **ZKPoK** for an important hard-on-average problem in lattice-based cryptography - the Inhomogeneous

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 107–124, 2013.

Small Integer Solution (ISIS) problem, that was introduced in [GPV08] and has since then been used extensively ([ABB10a, ABB10b, CHKP10, Boy10],...).

In recent years, lattice-based cryptography has received much attention from the research community because it enjoys a unique combination of attractive features: provable security under worst-case hardness assumptions, conjectured resistance against quantum computers, and asymptotic efficiency. The rapid development of the field yields an interesting challenge of designing and improving proof systems for lattice problems. There exist several proof systems, both interactive and non-interactive ([GG98, MV03, GMR05, PV08]) that exploit the geometric structure of worst-case lattice problems. On the other hand, when designing lattice-based cryptographic protocols, one essentially has to deal with the average-case problems that enjoy worst-case to average-case reductions, such as the SIS and ISIS problems ([Ajt96, MR07, GPV08]) and the Learning With Errors (LWE) problem ([Reg05, Reg09, Pei09]). All existing proofs of knowledge for the ISIS problem ([MV03, Lyu08]) have some limitations, most notably the fact that there is a gap between the norm of the witness vector and the norm of the vector computed by the knowledge extractor: The latter is only guaranteed to be $\widetilde{O}(n)$ larger than the former in the case of the infinity norm, where n denotes the dimension of the corresponding worst-case lattice problem. As a consequence, cryptographic schemes using these proof systems as building blocks rely on a stronger security assumption than the assumed hardness of finding a witness for the ISIS instance, by a $\widetilde{O}(n)$ factor. This hints that the existing **ZKPoK** for the ISIS$^\infty$ problem are sub-optimal: Is it possible to design an efficient **ZKPoK** for ISIS$^\infty$ whose security provably relies on a weaker assumption than the existing ones? In this work, we reply positively, and describe such a **ZKPoK**, for which there is only a constant gap between the norm of the witness vector and the norm of the vector computed by the extractor. We also briefly describe a scheme with no gap (i.e., constant factor 1), but that is less efficient.

Notations. Throughout the paper, we assume that all vectors are column vectors. We denote vectors by bold lower-case letters (e.g., $\mathbf{x}$), and matrices by bold upper-case letters (e.g., $\mathbf{A}$). The Gram-Schmidt norm of a matrix $\mathbf{A}$ is denoted by $\|\widetilde{\mathbf{A}}\|$. We let the Hamming weight of a vector $\mathbf{x} \in \{0,1\}^m$ be denoted by $\mathrm{wt}(\mathbf{x})$. We let B_{3m} denote the set of all vectors $\mathbf{x} \in \{-1,0,1\}^{3m}$ having exactly m coordinates equal to -1; m coordinates equal to 0; and m coordinates equal to 1. The symmetric group of all permutations of k elements is denoted by $\mathcal{S}_k$. We use the notation $y \xleftarrow{\$} \mathrm{D}$ when y is sampled from the distribution D. When S is a finite set, $y \xleftarrow{\$} \mathrm{S}$ means that y is chosen uniformly at random from S. We let n denote the security parameter of our schemes. A function $\epsilon : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said negligible in n (denoted by $\mathrm{negl}(n)$) if it vanishes faster than the inverse of any polynomial. We say that an event happens with overwhelming probability if it happens with probability $1 - \epsilon(n)$ for some negligible function ϵ. We often use the soft-O notation: We write $f(n) = \widetilde{O}(g(n))$ if $f(n) = O(g(n) \log^c g(n))$ for some constant c. The statistical distance between two distributions X and Y over a countable domain D is $\frac{1}{2}\sum_{d \in \mathrm{D}} |\mathrm{X}(d) - \mathrm{Y}(d)|$. We say that X and Y are statistically close (denoted by $\mathrm{X} \approx_s \mathrm{Y}$) if their statistical distance is negligible.

1.1 Related Works

We briefly review some of the results related to proofs of knowledge for the ISIS problem. The $\mathrm{ISIS}^{p}_{n,m,q,\beta}$ problem in the ℓ_p norm with parameters (n, m, q, β) asks to find a vector $\mathbf{x} \in \mathbb{Z}^m$ such that and $\|\mathbf{x}\|_p \leq \beta$ and $\mathbf{Ax} = \mathbf{y} \bmod q$ for a uniformly chosen input matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a uniformly chosen input vector $\mathbf{y} \in \mathbb{Z}_q^n$. The hardness of the $\mathrm{ISIS}^{2}_{n,m,q,\beta}$ problem is established by a worst-case to average-case reduction from standard lattice problems, such as the Shortest Independent Vectors Problem (SIVP).

Theorem 1 ([GPV08]). *For any m, $\beta = \mathrm{poly}(n)$, and for any integer $q \geq \beta \cdot \omega(\sqrt{n \log n})$, solving a* random *instance of the $\mathrm{ISIS}^{2}_{n,m,q,\beta}$ problem with non-negligible probability is at least as hard as approximating the $\mathrm{SIVP}^{2}_{\gamma}$ problem on* any *lattice of dimension n to within certain $\gamma = \beta \cdot \widetilde{O}(\sqrt{n})$ factors.*

By the relationship between the ℓ_2 and ℓ_∞ norms (i.e., for any vector $\mathbf{x} \in \mathbb{R}^n$, we have $\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_2 \leq \sqrt{n} \cdot \|\mathbf{x}\|_\infty$), it follows that the $\mathrm{ISIS}^{\infty}_{n,m,q,\beta}$ problem is at least as hard as $\mathrm{SIVP}^{2}_{\gamma}$ (in the ℓ_2 norm) for some $\gamma = \beta \cdot \widetilde{O}(n)$. Without loss of generality, throughout this work, we will assume that β is a positive *integer*. We define the relation $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$ for this problem as

$$\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}} = \Big\{((\mathbf{A}, \mathbf{y}), \mathbf{x}) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^n \times \mathbb{Z}^m \colon (\|\mathbf{x}\|_\infty \leq \beta) \wedge (\mathbf{Ax} = \mathbf{y}\,[q])\Big\}.$$

Kawachi et al. [KTX08] adapted Stern's identification scheme [Ste96] to the lattice setting to obtain a **ZKPoK** for a *restricted* version of the ISIS^{∞} problem, with respect to the relation

$$\mathrm{R}_{\mathrm{KTX}_{n,m,q,w}} = \Big\{((\mathbf{A}, \mathbf{y}), \mathbf{x}) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^n \times \{0,1\}^m \colon (\mathrm{wt}(\mathbf{x}) = w) \wedge (\mathbf{Ax} = \mathbf{y}\,[q])\Big\}.$$

This restriction of $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$ does not seem to suffice for a wide range of applications. For some cryptographic schemes that allow many users, such as ID-based identification [Sha85] and group signature [CH91] schemes, the secret keys of the users are typically generated from the public keys by a trusted authority. For such schemes that rely on lattice-based hardness assumptions ([SSTX09, Rüc10a, CNR12, GKV10]), this task is performed by using a secret trapdoor possessed by the trusted authority, consisting in a relatively short basis of a publicly known lattice. As a result, a user secret key $\mathbf{x}$ is a *general* solution to the $\mathrm{ISIS}^{\infty}_{n,m,q,\beta}$ problem, where β is typically $\widetilde{O}(\sqrt{n})$. Whenever a user in the scheme wants to identify himself, he must prove that he knows such a vector $\mathbf{x}$. In other words, these schemes require a **PoK** for the relation $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$, for which, up to the best of our knowledge, there exist two options:

- A proof of knowledge for $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$ was introduced by Lyubashevsky [Lyu08]. His protocol is efficient with low communication cost, but suffers from several limitations: It is not proven zero-knowledge (it is only proven to be witness-indistinguishable - a weaker notion than zero-knowledge [FS90]); It has a constant completeness error in each round; And it relies on a relatively strong hardness assumption for the ISIS^{∞} problem, with a $\widetilde{O}(n)$ gap factor.

• Another proof system can be obtained by transforming the ISIS instance into a GapCVP instance, and adapting the Micciancio-Vadhan **ZKPoK** for GapCVP [MV03] to the infinity norm. Let $\mathbf{B}$ be any basis of the lattice $\Lambda_q^{\perp}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{Ax} = \mathbf{0} \bmod q\}$ and $\mathbf{t}$ be a vector in $\mathbb{Z}^m$ such that $\mathbf{At} = \mathbf{y} \bmod q$. Such $\mathbf{B}$ and $\mathbf{t}$ can be efficiently computed using linear algebra. Then run the Micciancio-Vadhan protocol for $\mathrm{GapCVP}_{\gamma}^{\infty}$ with common input $(\mathbf{B}, \mathbf{t}, \beta)$. The prover's auxiliary input is $\mathbf{e} = \mathbf{t} - \mathbf{x} \in \Lambda_q^{\perp}(\mathbf{A})$. We note that the knowledge extractor in [MV03] is only able to output a vector $\mathbf{e}' \in \Lambda_q^{\perp}(\mathbf{A})$, such that $\|\mathbf{t} - \mathbf{e}'\|_{\infty} \leq g \cdot \beta$ for some $g > 1$. This implies that $\mathbf{x}' = \mathbf{t} - \mathbf{e}'$ is a solution to the $\mathrm{ISIS}_{n,m,q,g\cdot\beta}^{\infty}$ problem with respect to $(\mathbf{A}, \mathbf{y})$. However, in the infinity norm, the smallest g that can be obtained is $\geq \Theta(n/\log n)$ while the bit complexity is relatively high. In more details, the gap factor g depends on some parameter k as follows: $g = m^{1+\Omega(1)}$ for $k = \omega(1)$; $g = \Omega(m)$ for $k = \omega(\log m)$; and $g = \Omega(m/\log m)$ for $k = \mathrm{poly}(m)$ - a sufficiently large polynomial. The communication cost of the protocol depends linearly on k. Alternatively, one could apply the ISIS-GapCVP transformation to the Micciancio-Vadhan protocol for the ℓ_2 norm, and then use the relationship between the ℓ_2 and ℓ_{∞} norms. However, in this case, the gap is slightly bigger (at least $\Theta(n/\sqrt{\log n})$).

We now shortly review a class of proof systems related to our work: zero-knowledge proofs of plaintext knowledge (**ZKPoPK**) for Regev's LWE-based cryptosystem ([Reg05, Reg09]). All known **ZKPoPK** ([BD10, BDOZ11, AJLA+12, DLA12]) were derived from Secure Multi-Party Computation protocols, via the [IKOS07] transformation from **MPC** to **ZK**. The proof systems are relatively inefficient and rely on the assumption that SIVP is hard for super-polynomial approximation factors (i.e., $\gamma = n^{\omega(1)}$). We observe (in Section 3.2) that a **PoPK** for Regev's cryptosystem can be obtained from a **PoK** for $\mathrm{R}_{\mathrm{ISIS}}$. Thus, a **ZKPoK** for the ISIS problem with lower communication cost and a weaker hardness assumption leads to a significant improvement in this direction.

1.2 Our Contributions and Techniques

The discussions above raise the question whether it is possible to design a **ZKPoK** for the *general* ISIS problem that completely removes the gap. Even a **ZKPoK** that has small constant gap factor while maintaining efficiency would be desirable. In this work, we answer this question positively. Specifically, we show that there exists a statistical **ZKPoK** (called **Naive SternExt**) for the relation $\mathrm{R}_{\mathrm{ISIS}_{n,m,q,\beta}^{\infty}}$ whose security relies on the assumed hardness of the $\mathrm{ISIS}_{n,m,q,\beta}^{\infty}$. This scheme achieves optimal gap, as the norm bounds for the witness and the security assumptions are identical. However, its communication cost depends linearly on β, which may be a significant drawback for large β. Our main result is a statistical **ZKPoK** called **SternExt** achieving both security and efficiency requirements: it has an almost optimal gap factor ($g \leq 2$), while the communication cost compares favorably to the Micciancio-Vadhan proof system. We believe that our result can be applied to many cryptographic primitives. In particular, we will describe two applications of the **SternExt** proof system:

1. A concurrently secure identity-based identification scheme that relies on worst-case hardness of the $\mathrm{SIVP}_{\widetilde{O}(n^{1.5})}$ problem (in the ℓ_2 norm) in general lattices. This is the weakest security assumption among contemporary lattice-based ID-based ID schemes ([SSTX09, Rüc10a]).
2. An efficient statistical **ZKPoPK** for Regev's cryptosystem with small constant gap factor between the sizes of a valid plaintext and the output of the knowledge extractor. In comparison with the results of [BD10, BDOZ11, AJLA+12, DLA12], our proof system offers a noticeable improvement in both security and efficiency points of view.

We now sketch our approach. While the [MV03] protocol exploits the geometric aspect of the ISIS problem, our protocol exploits its combinatorial and algebraic aspects. We first look at the scheme from [Ste96, KTX08], and investigate how to loosen the restrictions on the witness $\mathbf{x}$, which are $\mathbf{x} \in \{0,1\}^m$ and $\mathrm{wt}(\mathbf{x}) = w$. Note that these conditions are invariant under all permutations of coordinates: For $\pi \in \mathcal{S}_m$, a vector $\mathbf{x}$ satisfies those restrictions if and only if $\pi(\mathbf{x})$ also does. Thus, a witness $\mathbf{x}$ with such constraints can be verified in zero-knowledge thanks to the randomness of π. We then notice that the same statement still holds true for $\mathbf{x} \in \mathrm{B}_{3m}$, namely: for $\pi \in \mathcal{S}_{3m}$, $\mathbf{x} \in \mathrm{B}_{3m} \Leftrightarrow \pi(\mathbf{x}) \in \mathrm{B}_{3m}$. This basic fact allows us to generalize the proof system from [Ste96, KTX08]. Our generalization consists of two steps:

Step 1. *Removing the restriction on the Hamming weight.* Specifically, we observe that a **ZKPoK** for the relation

$$\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,1}} = \left\{((\mathbf{A},\mathbf{y}),\mathbf{x}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n \times \{-1,0,1\}^m \colon \mathbf{A}\mathbf{x} = \mathbf{y} \bmod q\right\}$$

can be derived from Stern's scheme by the following *extensions*: For any vector $\mathbf{x} \in \{-1,0,1\}^m$, append $2m$ coordinates from the set $\{-1,0,1\}$ to $\mathbf{x}$ to obtain $\mathbf{x}' \in \mathrm{B}_{3m}$. Next, append $2m$ zero-columns to matrix $\mathbf{A}$ to get $\mathbf{A}' \in \mathbb{Z}_q^{n\times 3m}$. We then have:

$$\mathbf{x}' \in \mathrm{B}_{3m} \Leftrightarrow \mathbf{x} \in \{-1,0,1\}^m,$$
$$\mathbf{A}'\mathbf{x}' = \mathbf{y} \bmod q \Leftrightarrow \mathbf{A}\mathbf{x} = \mathbf{y} \bmod q.$$

In other words, if a verifier is convinced that $\mathbf{x}' \in \mathrm{B}_{3m}$ and $\mathbf{A}'\mathbf{x}' = \mathbf{y} \bmod q$, then he is also convinced that $\mathbf{x}$ is a valid witness for the relation $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,1}}$.

Step 2. *Increasing the ℓ_∞ bound to β, for any $\beta > 0$.* The principle of **Step 1** can be generalized in a naive manner. For any $\mathbf{x} \in \{-\beta,\ldots,0,\ldots,\beta\}^m$, one can append $2\beta m$ coordinates to $\mathbf{x}$ to obtain an $\mathbf{x}^* \in \{-\beta,\ldots,0,\ldots,\beta\}^{(2\beta+1)m}$ that has exactly m coordinates equal to d for each $d \in \{-\beta,\ldots,0,\ldots,\beta\}$. The extended matrix $\mathbf{A}^* \in \mathbb{Z}_q^{n\times(2\beta+1)m}$ is obtained by appending $2\beta m$ zero-columns to matrix $\mathbf{A}$. Then $\mathbf{A}^*\mathbf{x}^* = \mathbf{A}\mathbf{x} \bmod q$. Moreover, the constraints of $\mathbf{x}^*$ can be verified in zero-knowledge by using a uniform $\pi \in \mathcal{S}_{(2\beta+1)m}$. Therefore, we obtain a **ZKPoK** for $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$, that we call **Naive SternExt**, where the extraction gap factor is completely removed. However, as mentioned earlier, the proof is inefficient for large β as its communication cost is $\beta \cdot \widetilde{O}(n \lg q)$.

A much more efficient method to achieve our goal is based on the idea of representing any vector $\mathbf{x} \in \{-\beta, \ldots, 0, \ldots, \beta\}^m$ by $k = \lfloor \lg \beta \rfloor + 1$ vectors $\tilde{\mathbf{u}}_0, \ldots, \tilde{\mathbf{u}}_{k-1}$ in $\{-1, 0, 1\}^m$ via a binary decomposition, namely: $\mathbf{x} = \sum_{j=0}^{k-1} 2^j \cdot \tilde{\mathbf{u}}_j$. Next we apply the extension of **Step 1**: Extend each $\tilde{\mathbf{u}}_j$ to $\mathbf{u}_j \in \mathrm{B}_{3m}$, and extend $\mathbf{A}$ to $\mathbf{A}' \in \mathbb{Z}_q^{n \times 3m}$. We then have:

$$\mathbf{A}'\big(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{u}_j\big) = \mathbf{y} \bmod q \Leftrightarrow \mathbf{A}\mathbf{x} = \mathbf{y} \bmod q.$$

This allows us to combine k proofs for $\mathrm{R}_{\mathrm{ISIS}^\infty_{n,m,q,1}}$ into one proof $\mathrm{R}_{\mathrm{ISIS}^\infty_{n,m,q,\beta}}$.[1] We thus obtain a statistical **ZKPoK** for the *general* ISIS^∞ problem, that we call **SternExt**, with the following properties:

- The knowledge extractor obtains an $\mathbf{x}'$ with $\|\mathbf{x}'\|_\infty \leq \beta'$, where $\beta \leq \beta' \leq 2\beta - 1$ (depending on the binary representation of β). Hence, the extraction gap factor satisfies $g < 2$.
- The communication cost is $\lg \beta \cdot \widetilde{O}(n \lg q)$. In particular, in most cryptographic applications q is $\mathrm{poly}(n)$, and we then have $\lg \beta \leq \lg q = \widetilde{O}(1)$.

Overall, **SternExt** provides a better proof system for $\mathrm{R}_{\mathrm{ISIS}^\infty_{n,m,q,\beta}}$ in both security and efficiency aspects than the one derived from the Micciancio-Vadhan protocol. We summarize the comparison among the **PoK** for $\mathrm{R}_{\mathrm{ISIS}^\infty_{n,m,q,\beta}}$ in Table 1. The comparison data are for one round of protocol, in which case all the considered proof systems admit a constant soundness error.

Table 1. Comparison among the proofs of knowledge for $\mathrm{R}_{\mathrm{ISIS}^\infty_{n,m,q,\beta}}$. See discussion in Section 1.1 for other security/efficiency trade-offs for the [MV03] scheme.

Schemes	[Lyu08]	[MV03]	**Naive SternExt**	**SternExt**
Zero-knowledge?	✗(WI)	✓	✓	✓
Perfect completeness?	✗	✓	✓	✓
Norm bound in the ISIS hardness assumption	$\beta \cdot \widetilde{O}(n)$	$\beta \cdot \widetilde{O}(n)$	β	$\leq 2\beta - 1$
Communication cost	$\widetilde{O}(n \lg q)$	$\widetilde{O}(n \lg q)$	$\beta \cdot \widetilde{O}(n \lg q)$	$\lg \beta \cdot \widetilde{O}(n \lg q)$

Outline. The rest of the paper is organized as follows: In Section 2, we present the **SternExt** proof system; and in Section 3, we describe two cryptographic applications. We refer the reader to [Gol04, Chap. 4] and [GPV08] for standard definitions of zero-knowledge proof systems and lattice problems, respectively. In the appendix, we adapt **SternExt** to the relation $\mathrm{R}_{\mathrm{SIS}^\infty}$ associated to the SIS problem: $\mathrm{R}_{\mathrm{SIS}^\infty}$ corresponds to setting $\mathbf{y} = \mathbf{0}$ and imposing $\mathbf{x} \neq \mathbf{0}$ in $\mathrm{R}_{\mathrm{ISIS}^\infty}$.

[1] This packing of proofs is akin to Jain et al.'s recent work on the Learning Parity with Noise problem [JKPT12, Section 4.2].

2 A Zero-Knowledge Proof of Knowledge for ISIS

Our scheme extends Stern's **ZKPoK** [Ste96] for the Syndrome Decoding Problem (SDP). Stern's proof system is a 3-move interactive protocol: the prover P computes three commitments and sends them to the verifier V; verifier V sends a uniformly random challenge to P; prover P reveals two of the three commitments according to the challenge. Kawachi et al. [KTX08] adapted Stern's scheme to the lattice setting, exploiting the similarity between the SDP and ISIS problems. Their construction makes use of a string commitment scheme that is statistically hiding and computationally binding.

Definition 1. *A statistically hiding, computationally binding string commitment scheme is a* PPT *algorithm* $\mathrm{COM}(s, \rho)$ *satisfying:*

- *For all* $s_0, s_1 \in \{0,1\}^*$*, we have (over the random coins of* COM*):*
$$\mathrm{COM}(s_0; \cdot) \approx_s \mathrm{COM}(s_1; \cdot),$$
- *For all* PPT *algorithm* $\mathcal{A}$ *returning* $(s_0, \rho_0); (s_1, \rho_1)$*, where* $s_0 \neq s_1$*, we have (over the random coins of* $\mathcal{A}$*):*
$$\Pr[\mathrm{COM}(s_0; \rho_0) = \mathrm{COM}(s_1; \rho_1)] = \mathrm{negl}(n).$$

2.1 Setup

For a security parameter n, let q be a positive integer. Let β be some positive integer, and $k = \lfloor \lg \beta \rfloor + 1$. Let COM be a statistically hiding and computationally binding string commitment scheme. It was shown in [KTX08] that such a scheme can be constructed based on the hardness of the $\mathrm{ISIS}^{\infty}_{n,m,q,\widetilde{O}(1)}$ problem. For simplicity, in the interactive protocol, we will not explicitly write the randomness ρ of the commitment scheme COM.

The common input is a pair $(\mathbf{A}, \mathbf{y})$ such that $\mathbf{y}$ belongs to the image of $\mathbf{A}$, and the prover's auxiliary input is vector $\mathbf{x}$. Prior to the interaction, both P and V form the extended matrix $\mathbf{A}' \in \mathbb{Z}_q^{n \times 3m}$ by appending $2m$ zero-columns to matrix $\mathbf{A}$. In addition, prover P performs the following preparation steps:

1. Decomposition. The goal is to represent vector $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ by k vectors in $\{-1, 0, 1\}^m$. For each $1 \leq i \leq m$, consider a binary representation of coordinate x_i, that is: $x_i = b_{i,0} \cdot 2^0 + b_{i,1} \cdot 2^1 + \ldots + b_{i,k-1} \cdot 2^{k-1}$, where $b_{i,j} \in \{-1, 0, 1\}$, for all $j = 0, \ldots, k-1$. Now for each index j, let $\tilde{\mathbf{u}}_j = (b_{1,j}, b_{2,j}, \ldots, b_{m,j}) \in \{-1, 0, 1\}^m$. We observe that $\mathbf{x} = \sum_{j=0}^{k-1} 2^j \cdot \tilde{\mathbf{u}}_j$.
2. Extension. For each index $j = 0, \ldots, k-1$, extend $\tilde{\mathbf{u}}_j$ to a vector $\mathbf{u}_j \in \mathrm{B}_{3m}$ as follows: If the numbers of coordinates -1, 0, and 1 in vectors $\tilde{\mathbf{u}}_j$ are $\lambda_j^{(-1)}$, $\lambda_j^{(0)}$ and $\lambda_j^{(1)}$ respectively, then choose a random vector $\mathbf{t}_j \in \{-1, 0, 1\}^{2m}$ that has exactly $(m - \lambda_j^{(-1)})$ coordinates -1, $(m - \lambda_j^{(0)})$ coordinates 0, and

$(m - \lambda_j^{(1)})$ coordinates 1; and append $\mathbf{t}_j$ to $\tilde{\mathbf{u}}_j$, i.e., set $\mathbf{u}_j = (\tilde{\mathbf{u}}_j \| \mathbf{t}_j)$. Since the last $2m$ columns of matrix $\mathbf{A}'$ are zero-columns, we have:

$$\mathbf{A}'\big(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{u}_j\big) = \mathbf{y} \bmod q \Leftrightarrow \mathbf{A}\mathbf{x} = \mathbf{y} \bmod q.$$

2.2 The Interactive Proof System

The prover P and the verifier V interact as described in Figure 1.

1. **Commitment.** Prover P samples k vectors $\mathbf{r}_0, \ldots, \mathbf{r}_{k-1} \xleftarrow{\$} \mathbb{Z}_q^{3m}$; k permutations $\pi_0, \ldots, \pi_{k-1} \xleftarrow{\$} \mathcal{S}_{3m}$, and sends the commitment $\mathrm{CMT} := (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$, where

$$\begin{cases} \mathbf{c}_1 = \mathrm{COM}(\pi_0, \ldots, \pi_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{r}_j) \bmod q) \\ \mathbf{c}_2 = \mathrm{COM}(\pi_0(\mathbf{r}_0), \ldots, \pi_{k-1}(\mathbf{r}_{k-1})) \\ \mathbf{c}_3 = \mathrm{COM}(\pi_0(\mathbf{u}_0 + \mathbf{r}_0), \ldots, \pi_{k-1}(\mathbf{u}_{k-1} + \mathbf{r}_{k-1})) \end{cases}$$

2. **Challenge.** Receiving CMT, verifer V sends a challenge $Ch \xleftarrow{\$} \{1, 2, 3\}$ to P.
3. **Response.** Prover P replies as follows:
   - If $Ch = 1$, then reveal $\mathbf{c}_2$ and $\mathbf{c}_3$. For each j, let $\mathbf{v}_j = \pi_j(\mathbf{u}_j)$, and $\mathbf{w}_j = \pi_j(\mathbf{r}_j)$. Send $\mathrm{RSP} := (\mathbf{v}_0, \ldots, \mathbf{v}_{k-1}, \mathbf{w}_0, \ldots, \mathbf{w}_{k-1})$.
   - If $Ch = 2$, then reveal $\mathbf{c}_1$ and $\mathbf{c}_3$. For each j, let $\phi_j = \pi_j$, and $\mathbf{z}_j = \mathbf{u}_j + \mathbf{r}_j$. Send $\mathrm{RSP} := (\phi_0, \ldots, \phi_{k-1}, \mathbf{z}_0, \ldots, \mathbf{z}_{k-1})$.
   - If $Ch = 3$, then reveal $\mathbf{c}_1$ and $\mathbf{c}_2$. For each j, let $\psi_j = \pi_j$, and $\mathbf{s}_j = \mathbf{r}_j$. Send $\mathrm{RSP} := (\psi_0, \ldots, \psi_{k-1}, \mathbf{s}_0, \ldots, \mathbf{s}_{k-1})$.

Verification. Receiving the response RSP, verifier V performs the following checks:

- If $Ch = 1$: Check that $\mathbf{v}_j \in \mathrm{B}_{3m}$ for all $j = 0, \ldots, k-1$, and

$$\begin{cases} \mathbf{c}_2 = \mathrm{COM}(\mathbf{w}_0, \ldots, \mathbf{w}_{k-1}) \\ \mathbf{c}_3 = \mathrm{COM}(\mathbf{v}_0 + \mathbf{w}_0, \ldots, \mathbf{v}_{k-1} + \mathbf{w}_{k-1}) \end{cases}$$

- If $Ch = 2$: Check that

$$\begin{cases} \mathbf{c}_1 = \mathrm{COM}(\phi_0, \ldots, \phi_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{z}_j) - \mathbf{y} \bmod q) \\ \mathbf{c}_3 = \mathrm{COM}(\phi_0(\mathbf{z}_0), \ldots, \phi_{k-1}(\mathbf{z}_{k-1})) \end{cases}$$

- If $Ch = 3$: Check that

$$\begin{cases} \mathbf{c}_1 = \mathrm{COM}(\psi_0, \ldots, \psi_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{s}_j) \bmod q) \\ \mathbf{c}_2 = \mathrm{COM}(\psi_0(\mathbf{s}_0), \ldots, \psi_{k-1}(\mathbf{s}_{k-1})) \end{cases}$$

In each case, verifier V outputs the decision $d = 1$ (Accept) if and only if all the conditions hold. Otherwise, he outputs $d = 0$ (Reject).

Fig. 1. The **SternExt** proof system

Completeness. We observe that if prover P has a valid witness $\mathbf{x}$ for the relation $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$ and follows the protocol, then he always gets accepted by V. Therefore, the proof system has perfect completeness.

Communication Cost. The size of the commitment scheme from [KTX08] is $\widetilde{O}(n \lg q)$. If $Ch = 1$, then the size of RSP is $3km + 3km \lg q$. If $Ch = 2$ or $Ch = 3$, then RSP consists of k vectors in $\mathbb{Z}_q^{3m}$ and k permutations. Note that in practice, instead of sending the permutations and vectors, one would send the *random seed* of the PRNG used to generate these data, and thus significantly reduce the communication cost. Overall, the total communication cost of the protocol is $\lg \beta \cdot \widetilde{O}(n \lg q)$.

2.3 Statistical Zero-Knowledge

We now prove that the proof system **SternExt** is statistically zero-knowledge, by exhibiting a transcript simulator.

Theorem 2. *If* COM *is a statistically hiding string commitment scheme, then the proof system* ***SternExt*** *from Figure 1 is statistically zero-knowledge.*

Proof. Adapting the techniques of [Ste96] and [KTX08], we construct a simulator $\mathcal{S}$ which has black-box access to a (possibly cheating) verifier $\widehat{V}$, such that on input the public parameters $\mathbf{A}$ (and implicitly its extension $\mathbf{A}'$) and $\mathbf{y}$, outputs with probability 2/3 a successful transcript (i.e., an accepted interaction), and the view of $\widehat{V}$ in the simulation is statistically close to that in the real interaction. The simulator $\mathcal{S}$ begins by selecting a random $\overline{Ch} \in \{1, 2, 3\}$ (a prediction of the challenge value that $\widehat{V}$ will *not* choose), and a random tape r' of $\widehat{V}$. We note that in all the cases we consider below, by the assumption on the commitment scheme COM, the distributions of $\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3$ are statistically close to the distributions of the commitments in the real interaction, and thus, the distributions of the challenge Ch from $\widehat{V}$ is also statistically close to that in the real interactions.

Case $\overline{Ch} = 1$: The simulator $\mathcal{S}$ computes $\mathbf{x}' \in \mathbb{Z}_q^m$ such that $\mathbf{A}\mathbf{x}' = \mathbf{y} \bmod q$ using linear algebra. It picks $k-1$ random vectors $\widetilde{\mathbf{u}}'_1, \ldots, \widetilde{\mathbf{u}}'_{k-1} \xleftarrow{\$} \mathbb{Z}_q^m$ and sets:

$$\widetilde{\mathbf{u}}'_0 := \mathbf{x}' - \sum_{j=1}^{k-1} 2^j \cdot \widetilde{\mathbf{u}}'_j \bmod q.$$

In other words, we have $\mathbf{x}' = \sum_{j=0}^{k-1} 2^j \cdot \widetilde{\mathbf{u}}'_j \bmod q$. Now for each j, the simulator extends $\widetilde{\mathbf{u}}'_j$ to $\mathbf{u}'_j \in \mathbb{Z}_q^{3m}$ by appending $2m$ random coordinates. It then picks k vectors $\mathbf{r}'_0, \ldots, \mathbf{r}'_{k-1} \xleftarrow{\$} \mathbb{Z}_q^{3m}$; k permutations $\pi'_0, \ldots, \pi'_{k-1} \xleftarrow{\$} \mathcal{S}_{3m}$; and uniformly random strings $\rho'_1, \rho'_2, \rho'_3$. It sends the following commitments to $\widehat{V}$:

$$\begin{cases} \mathbf{c}'_1 = \mathrm{COM}(\pi'_0, \ldots, \pi'_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{r}'_j) \bmod q; \rho'_1) \\ \mathbf{c}'_2 = \mathrm{COM}(\pi'_0(\mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{r}'_{k-1}); \rho'_2) \\ \mathbf{c}'_3 = \mathrm{COM}(\pi'_0(\mathbf{u}'_0 + \mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{u}'_{k-1} + \mathbf{r}'_{k-1}); \rho'_3). \end{cases}$$

Receiving a challenge Ch from $\widehat{V}$, simulator $\mathcal{S}$ provides a transcript as follows:

- If $Ch = 1$: Output $\perp$ and halt.
- If $Ch = 2$: Output

$$\Big(r', (\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3), 2, (\pi'_0, \pi'_1, \ldots, \pi'_{k-1}, \mathbf{u}'_0 + \mathbf{r}'_0, \ldots, \mathbf{u}'_{k-1} + \mathbf{r}'_{k-1}); \rho'_1, \rho'_3\Big).$$

- If $Ch = 3$: Output $\Big(r', (\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3), 3, (\pi'_0, \ldots, \pi'_{k-1}, \mathbf{r}'_0, \ldots, \mathbf{r}'_{k-1}); \rho'_1, \rho'_2\Big)$.

Case $\overline{Ch} = 2$: The simulator $\mathcal{S}$ picks $\mathbf{r}'_0, \ldots, \mathbf{r}'_{k-1} \xleftarrow{\$} \mathbb{Z}_q^{3m}$; $\mathbf{u}'_0, \ldots, \mathbf{u}'_{k-1} \xleftarrow{\$} \mathrm{B}_{3m}$; permutations $\pi'_0, \ldots, \pi'_{k-1} \xleftarrow{\$} \mathcal{S}_{3m}$; and uniformly random strings $\rho'_1, \rho'_2, \rho'_3$. It sends to $\widehat{V}$ the commitments:

$$\begin{cases} \mathbf{c}'_1 = \mathrm{COM}(\pi'_0, \ldots, \pi'_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{r}'_j) \bmod q; \rho'_1) \\ \mathbf{c}'_2 = \mathrm{COM}(\pi'_0(\mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{r}'_{k-1}); \rho'_2) \\ \mathbf{c}'_3 = \mathrm{COM}(\pi'_0(\mathbf{u}'_0 + \mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{u}'_{k-1} + \mathbf{r}'_{k-1}); \rho'_3). \end{cases}$$

Receiving a challenge Ch from $\widehat{V}$, simulator $\mathcal{S}$ computes the following transcript:

- If $Ch = 1$: Output

$$\Big(r', (\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3), 1, (\pi'_0(\mathbf{u}'_0), \ldots, \pi'_{k-1}(\mathbf{u}'_{k-1}), \pi'_0(\mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{r}'_{k-1})); \rho'_2, \rho'_3\Big).$$

- If $Ch = 2$: Output $\perp$ and halt.
- If $Ch = 3$: Output $\Big(r', (\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3), 3, (\pi'_0, \ldots, \pi'_{k-1}, \mathbf{r}'_0, \ldots, \mathbf{r}'_{k-1}); \rho'_1, \rho'_2\Big)$.

Case $\overline{Ch} = 3$: The simulator picks the uniformly random vectors, permutations, and strings exactly as in the case $\overline{Ch} = 2$ above, but sends the following:

$$\begin{cases} \mathbf{c}'_1 = \mathrm{COM}(\pi'_0, \ldots, \pi'_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot (\mathbf{u}'_j + \mathbf{r}'_j)) - \mathbf{y} \bmod q; \rho'_1) \\ \mathbf{c}'_2 = \mathrm{COM}(\pi'_0(\mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{r}'_{k-1}); \rho'_2) \\ \mathbf{c}'_3 = \mathrm{COM}(\pi'_0(\mathbf{u}'_0 + \mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{u}'_{k-1} + \mathbf{r}'_{k-1}); \rho'_3). \end{cases}$$

Receiving a challenge Ch from $\widehat{V}$, simulator $\mathcal{S}$ computes a transcript as follows:

- If $Ch = 1$: Output

$$\Big(r', (\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3), 1, (\pi'_0(\mathbf{u}'_0), \ldots, \pi'_{k-1}(\mathbf{u}'_{k-1}), \pi'_0(\mathbf{r}'_0), \ldots, \pi'_{k-1}(\mathbf{r}'_{k-1})); \rho'_2, \rho'_3\Big).$$

- If $Ch = 2$: Output

$$\Big(r', (\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3), 2, (\pi'_0, \ldots, \pi'_{k-1}, \mathbf{u}'_0 + \mathbf{r}'_0, \ldots, \mathbf{u}'_{k-1} + \mathbf{r}'_{k-1}); \rho'_1, \rho'_3\Big).$$

- If $Ch = 3$: Output $\perp$ and halt.

We observe that the probability that the simulator outputs $\perp$ is negligibly close to 1/3. Moreover, one can check that whenever $\mathcal{S}$ does not halt, it will provide a successful transcript, and the distribution of the transcript is statistically close to that of the prover in the real interaction. Hence, we have constructed a simulator that can successfully impersonate the honest prover with probability 2/3, and completed the proof.

2.4 Proof of Knowledge

The fact that anyone can run the simulator to convince the verifier with probability 2/3 implies that the **SternExt** proof system has soundness error $\geq 2/3$. In the following, we prove that it is indeed a proof of knowledge for the relation $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$ with knowledge error $\kappa = 2/3$.

Theorem 3. *Assume that* COM *is a computationally binding string commitment scheme. Then there exists a knowledge extractor $\mathcal{K}$ such that the following holds. If $\mathcal{K}$ has access to a cheating prover who convinces the verifier on input $(\mathbf{A}, \mathbf{y})$ with probability $2/3 + \epsilon$ for some $\epsilon > 0$ and in time T, then $\mathcal{K}$ outputs an $\mathbf{x}$ such that $((\mathbf{A}, \mathbf{y}); \mathbf{x}) \in \mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,2\beta-1}}$ with overwhelming probability and runtime $T \cdot \mathrm{poly}(n, m, \lg q, 1/\epsilon)$.*

As a corollary, **SternExt** is sound for uniform $(\mathbf{A}, \mathbf{y})$ under the assumption that the $\mathrm{ISIS}^{\infty}_{n,m,q,2\beta-1}$ problem is hard.

Proof. We apply the technique of [Vér96] relying on trees to model the probability space corresponding to the protocol execution. Suppose a cheating prover $\widehat{P}$ can convince the verifier with probability $2/3 + \epsilon$. Then by rewinding $\widehat{P}$ a number of times polynomial in $1/\epsilon$, the knowledge extractor $\mathcal{K}$ can find with overwhelming probability a node with 3 sons in the tree associated with the protocol between $\widehat{P}$ and the verifier. This node corresponds to the reception of all 3 values of the challenge. In other words, $\widehat{P}$ is able to answer correctly to all challenges for the same commitment. Therefore, $\mathcal{K}$ can get the following relations:

$$\mathrm{COM}(\phi_0, \ldots, \phi_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{z}_j) - \mathbf{y}) = \mathrm{COM}(\psi_0, \ldots, \psi_{k-1}, \mathbf{A}'(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{s}_j))$$
$$\mathrm{COM}(\mathbf{w}_0, \ldots, \mathbf{w}_{k-1}) = \mathrm{COM}(\psi_0(\mathbf{s}_0), \ldots, \psi_{k-1}(\mathbf{s}_{k-1}))$$
$$\mathrm{COM}(\phi_0(\mathbf{z}_0), \ldots, \phi_{k-1}(\mathbf{z}_{k-1})) = \mathrm{COM}(\mathbf{v}_0 + \mathbf{w}_0, \ldots, \mathbf{v}_{k-1} + \mathbf{w}_{k-1}),$$

and $\mathbf{v}_j \in \mathrm{B}_{3m}$ for all $j = 0, \ldots, k-1$. Since COM is computationally binding, it follows that:

$$\mathbf{A}'\Big(\sum_{j=0}^{k-1} 2^j \cdot (\mathbf{z}_j - \mathbf{s}_j)\Big) = \mathbf{y} \bmod q,$$

and for all j, we have $\phi_j = \psi_j; \mathbf{w}_j = \psi_j(\mathbf{s}_j); \mathbf{v}_j + \mathbf{w}_j = \phi_j(\mathbf{z}_j); \mathbf{v}_j \in \mathrm{B}_{3m}$. This implies that $\phi_j(\mathbf{z}_j - \mathbf{s}_j) = \mathbf{v}_j \in \mathrm{B}_{3m}$. Let $\mathbf{v}'_j := \mathbf{z}_j - \mathbf{s}_j = \phi_j^{-1}(\mathbf{v_j})$, then we obtain that $\mathbf{A}'\Big(\sum_{j=0}^{k-1} 2^j \cdot \mathbf{v}'_j\Big) = \mathbf{y} \bmod q$ and $\mathbf{v}'_j \in \mathrm{B}_{3m}$. Then for each $\mathbf{v}'_j$, we drop the last $2m$ coordinates to obtain $\widetilde{\mathbf{v}'_j} \in \{-1, 0, 1\}^m$. Now we have $\mathbf{A}\Big(\sum_{j=0}^{k-1} 2^j \cdot \widetilde{\mathbf{v}'_j}\Big) = \mathbf{y} \bmod q$. Let $\mathbf{x}' = \sum_{j=0}^{k-1} 2^j \cdot \widetilde{\mathbf{v}'_j}$. Then $\mathbf{A}\mathbf{x}' = \mathbf{y} \bmod q$, and

$$\|\mathbf{x}'\|_\infty \leq \sum_{j=0}^{k-1} 2^j \cdot \|\widetilde{\mathbf{v}'_j}\|_\infty \leq \sum_{j=0}^{k-1} 2^j = \sum_{j=0}^{\lfloor \lg \beta \rfloor} 2^j = 2^{\lfloor \lg \beta \rfloor + 1} - 1 \leq 2\beta - 1.$$

The knowledge extractor outputs $\mathbf{x}'$, which satisfies $((\mathbf{A}, \mathbf{y}; \mathbf{x}') \in \mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,2\beta-1}}$.

2.5 A Scheme Variant with No Gap

In a personal communication, D. Micciancio indicated to the authors a modification of the **SternExt** proof system that removes the extraction gap entirely. Instead of relying on powers of 2, one can use the following sequence of integers: $b_1 = \lceil \beta/2 \rceil$, $b_2 = \lceil (\beta - b_1)/2 \rceil$, $b_3 = \lceil (\beta - b_1 - b_2)/2 \rceil$, ... , and 1. One obtains a sequence of numbers of length $k = \lfloor \lg \beta \rfloor + 1$, whose subset sums are precisely the numbers between 0 and β. Finally, any integer in this interval can be efficiently expressed as a subset sum of the integers in the sequence.

3 Applications

Our results described in Section 2 yield the potential of enabling weaker security assumptions and lower complexities for various lattice-based cryptographic constructions. In this section, we will describe two applications of the **SternExt** proof system: an improved ID-based identification scheme and a new **ZKPoPK** for Regev's encryption scheme [Reg05, Reg09].

3.1 Identity-Based Identification

Definition 2 ([BNN09]). *An* identity-based identification (IBI) scheme *is a tuple of four* PPT *algorithms* $(\mathrm{MKg}, \mathrm{UKg}, \overline{\mathrm{P}}, \overline{\mathrm{V}})$*:*

- $\mathrm{MKg}(1^n)$*: On input* 1^n*, output a master public and master secret key pair* (mpk, msk).
- $\mathrm{UKg}(msk, id)$*: On input* msk *and a user identity* $id \in \{0,1\}^*$*, output a secret key* sk_{id} *for this user.*
- $\langle \overline{\mathrm{P}}, \overline{\mathrm{V}} \rangle$ *is an interactive protocol. The prover* $\overline{\mathrm{P}}$ *takes* (mpk, id, sk_{id}) *as input, the verifier* $\overline{\mathrm{V}}$ *takes* (mpk, id) *as input. At the end of the protocol,* $\overline{\mathrm{V}}$ *outputs* 1 *(accept) or* 0 *(reject).*

The completeness requirement for an IBI scheme is as follows: For any mpk generated by $\mathrm{MKg}(1^n)$, and sk_{id} extracted by $\mathrm{UKg}(msk, id)$, the decision of $\overline{\mathrm{V}}$ after interacting with $\overline{\mathrm{P}}$ is always 1. We refer the reader to [BNN09] for formal definitions of security notions for IBI schemes.

A common strategy in constructing IBI schemes consists in combining a signature scheme and a **PoK** in the following way: The trusted authority generates (mpk, msk) as a verification key - signing key pair of a signature scheme; whenever a user id queries for his secret key, the authority returns sk_{id} as a signature on id; for identification, the user plays the role of the prover, and runs a **PoK** to prove the possession of sk_{id}. If the signature scheme is strongly secure against existential forgery under chosen message attacks, and the **PoK** is at least witness-indistinguishable, then the resulting IBI scheme is secure against impersonation under concurrent attacks [BNN09]. This strategy is widely used for lattice-based IBI schemes. Stehlé et al. [SSTX09] combined the GPV signature scheme [GPV08], and the Micciancio-Vadhan [MV03] **PoK** to obtain an

IBI scheme based on the hardness of the $\mathrm{SIVP}_{\widetilde{O}(n^2)}$ problem (in the ℓ_2 norm). Rückert [Rüc10a] combined the Bonsai tree signature scheme [CHKP10] and Lyubashevsky's **PoK** [Lyu08] for ideal lattices to produce an IBI scheme based on the hardness of the restriction of $\mathrm{SVP}_{\widetilde{O}(n^{3.5})}$ to ideal lattices (in the ℓ_∞ norm).

Following the same approach, the **SternExt** proof system allows us to achieve better in terms of security assumption. Since **SternExt** is zero-knowledge, it has the witness-indistinguishability (**WI**) property. As **WI** is preserved under parallel composition [FS90], we can repeat the protocol $\omega(\log n)$ times in parallel to obtain a **WIPoK** with negligible soundness error. Combining with the GPV signature scheme, we obtain a secure IBI scheme in the random oracle model with hardness assumption $\mathrm{SIVP}_{\widetilde{O}(n^{1.5})}$. At first, we review the trapdoor generation and preimage sampling algorithms used in [GPV08], which will essentially serve as the $\mathrm{MKg}(1^n)$ and $\mathrm{UKg}(msk, id)$ algorithms in our IBI scheme. The following trapdoor generation algorithm was introduced in [Ajt99], improved in [AP11], and recently simplified in [MP12].

Lemma 1 ([AP11, MP12]). *Let $q \geq 2$ and $m \geq 6n \lg q$. There is a* PPT *algorithm* $\mathrm{TrapGen}(n, m, q)$ *that outputs a matrix $\mathbf{A}$ statistically close to uniform in $\mathbb{Z}_q^{n\times m}$, and a basis $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ for $\Lambda_q^\perp(\mathbf{A})$ satisfying $\|\widetilde{\mathbf{T_A}}\| \leq O(\sqrt{n \lg q})$.*

Given an integer lattice $\mathbf{L}$, the discrete Gaussian distribution $\mathrm{D}_{\mathbf{L},\sigma,\mathbf{c}}$ with parameter σ is the m-dimensional Gaussian distribution centered at $\mathbf{c}$, with support restricted to the lattice $\mathbf{L}$. Given a basis $\mathbf{B}$ for $\mathbf{L}$, the distribution $\mathrm{D}_{\mathbf{L},\sigma,\mathbf{c}}$ can be sampled efficiently for $\sigma \geq \|\widetilde{\mathbf{B}}\|\omega(\sqrt{\log m})$.

Lemma 2 ([GPV08]). *Let $q \geq 2$ and $m \geq n$. Let $\mathbf{A}$ be a matrix in $\mathbb{Z}_q^{n\times m}$ and $\mathbf{T_A}$ be a basis for $\Lambda_q^\perp(\mathbf{A})$. Then for $\mathbf{y}$ in the image of $\mathbf{A}$ and $\sigma \geq \|\widetilde{\mathbf{T_A}}\|\omega(\sqrt{\log m})$, there is a* PPT *algorithm* $\mathrm{SampleISIS}(\mathbf{A}, \mathbf{T_A}, \mathbf{y}, \sigma)$ *that outputs $\mathbf{x} \in \mathbb{Z}^m$ sampled from the distribution $\mathrm{D}_{\mathbb{Z}^m,\sigma,\mathbf{0}}$, conditioned on the event that $\mathbf{Ax} = \mathbf{y} \bmod q$.*

Let $\mathbf{x}$ be the output of $\mathrm{SampleISIS}(\mathbf{A}, \mathbf{T_A}, \mathbf{y}, \sigma)$. Gentry et al. [GPV08] noted that for any fixed function $t(m) \geq \omega(\sqrt{\log m})$, one has $\|\mathbf{x}\|_\infty \leq \sigma \cdot t$ with overwhelming probability. If $\mathbf{T_A}$ is a basis generated by $\mathrm{TrapGen}(n, m, q)$, then we can take $\sigma = O(\sqrt{n \lg q}) \cdot \omega(\sqrt{\log m})$. In this case, let $\beta = \lceil \sigma \cdot t \rceil = \widetilde{O}(\sqrt{n})$. Now let $\mathrm{H} : \{0,1\}^* \to \mathbb{Z}_q^n$ be the random oracle used in the GPV signature. For parameters (m, q, β, σ) as described above, we obtain the following IBI scheme:

- $\mathrm{MKg}(1^n)$: Run algorithm $\mathrm{TrapGen}(n, m, q)$ to output a master public key $mpk = \mathbf{A} \in \mathbb{Z}_q^{n\times m}$, and a master secret key $msk = \mathbf{T_A} \in \mathbb{Z}^{m\times m}$.
- $\mathrm{UKg}(msk, id)$: For $id \in \{0,1\}^*$, let $sk_{id} = \mathrm{SampleISIS}(\mathbf{A}, \mathbf{T_A}, \mathrm{H}(id), \sigma)$. If $\|sk_{id}\|_\infty > \beta$ (which happens with negligible probability) then restart. Otherwise, output sk_{id} as the secret key for identity id. We note that sk_{id} is the GPV signature for the message id, and is a solution to the $\mathrm{ISIS}^\infty_{n,m,q,\beta}$ instance $(\mathbf{A}, \mathrm{H}(id))$.
- $\langle \overline{\mathrm{P}}, \overline{\mathrm{V}} \rangle$: The common input is the pair $(\mathbf{A}, \mathrm{H}(id))$. The auxiliary input of $\overline{P}$ is sk_{id}. Then $\overline{P}$ and $\overline{V}$ play the roles of the prover and the verifier in the **SternExt** protocol. The protocol is repeated $l = \omega(\log n)$ times in parallel to make the soundness error negligibly small.

The completeness of the obtained IBI scheme follows from the perfect completeness of **SternExt**. Since the GPV signature scheme is strongly secure against existential forgery under chosen message attacks [GPV08], and the **SternExt** protocol is a **WIPoK**, the obtained IBI scheme is secure against impersonation under concurrent attacks. The scheme relies on the assumed hardness of the $\mathrm{ISIS}^{\infty}_{n,m,q,2\beta-1}$ problem, where $\beta = \widetilde{O}(\sqrt{n})$. It follows from Theorem 1 that solving the $\mathrm{ISIS}^{\infty}_{n,m,q,2\beta-1}$ problem is at least as hard as solving $\mathrm{SIVP}^{2}_{\gamma}$ (in the ℓ_2 norm) with $\gamma = (2\beta-1)\cdot\widetilde{O}(n) = \widetilde{O}(n^{1.5})$.

Theorem 4. *The obtained* IBI *scheme is concurrently secure in the random oracle model if the* $\mathrm{SIVP}_{\widetilde{O}(n^{1.5})}$ *problem is hard (in the worst-case).*

Similarly, combining the **SternExt** proof system with lattice-based signature schemes that are secure in the standard model (e.g., [CHKP10, Boy10, MP12]) we can obtain secure lattice-based IBI schemes in the standard model, with weaker security assumptions than in the contemporary schemes.

3.2 Proof of Plaintext Knowledge for Regev's Cryptosystem

Regev's LWE-based encryption scheme is as follows:

- **Parameters:** Integers n, m, q, an integer $p \ll q$ and a real $\alpha > 0$.
- **Private key:** The private key is $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$.
- **Public key:** Let $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and $\mathbf{e} \xleftarrow{\$} (\overline{\Psi}_\alpha(q))^m$, where $\overline{\Psi}_\alpha(q)$ is the LWE error distribution [Reg05, Reg09]. The public key is
$$(\mathbf{A}, \mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m.$$
- **Encryption:** The message space is $\{0, \ldots, p-1\}$. Given a message M, and the public key $(\mathbf{A}, \mathbf{b})$, choose a uniformly random[2] integer vector $\mathbf{r} \xleftarrow{\$} \{0, \ldots, p-1\}^m$, and output the ciphertext
$$(\mathbf{u}, c) = (\mathbf{A}\mathbf{r}, \mathbf{b}^T\mathbf{r} + M \cdot \lfloor q/p \rfloor) \in \mathbb{Z}_q^n \times \mathbb{Z}_q.$$
- **Decryption:** Given the ciphertext $(\mathbf{u}, c) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, and the private key $\mathbf{s} \in \mathbb{Z}_q^n$, output $M = \lfloor (c - \mathbf{s}^T\mathbf{u}) \cdot p/q \rceil$.

For the correctness, security, and parameters selection of this cryptosystem we refer to [Reg09]. We now show how to derive a **PoPK** for this encryption scheme from a **PoK** for the relation $\mathrm{R}_{\mathrm{ISIS}^\infty}$. A **PoPK** for Regev's cryptosystem is a **PoK** for the following relation:

$$\mathrm{R}_{\mathrm{Regev}} = \Big\{((\mathbf{A}, \mathbf{b}), (\mathbf{u}, c), \mathbf{r} \| M) \in (\mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m) \times (\mathbb{Z}_q^n \times \mathbb{Z}_q) \times \{0, \ldots, p-1\}^{m+1} : (\mathbf{u} = \mathbf{A}\mathbf{r}) \wedge (c = \mathbf{b}^T\mathbf{r} + M \cdot \lfloor q/p \rfloor)\Big\}.$$

[2] In fact, the proof system can be adapted to any nonce distribution, as long as $\|\mathbf{r}\|_\infty$ is bounded by some B sufficiently smaller than q.

We form the following matrix:

$$\mathbf{A}' = \left[\begin{array}{c|c} \multirow{2}{*}{$\mathbf{A}$} & \begin{matrix}0\\ \vdots \\ 0\end{matrix} \\ \hline \mathbf{b}^{\mathrm{T}} & \lfloor q/p \rceil \end{array}\right] \in \mathbb{Z}_q^{(n+1)\times(m+1)},$$

and let $\mathbf{y} = (\mathbf{u} \| c) \in \mathbb{Z}_q^{n+1}$. Let $\mathbf{x} = (\mathbf{r} \| M)$ be any witness of the relation $\mathrm{R}_{\mathrm{Regev}}$. Then we have $\mathbf{x} \in \mathbb{Z}^{m+1}$, and $\|\mathbf{x}\|_\infty \leq p-1$. Moreover, we observe that $\mathbf{A}'\mathbf{x} = \mathbf{y} \bmod q$. Therefore, vector $\mathbf{x}$ is a solution to the ISIS^∞ problem with parameters $(n+1, m+1, q, p-1)$ defined by $(\mathbf{A}', \mathbf{y})$. In other words, we have shown that the relation $\mathrm{R}_{\mathrm{Regev}}$ can be embedded into the relation $\mathrm{R}_{\mathrm{ISIS}^\infty_{n+1,m+1,q,p-1}}$. We then run the **SternExt** protocol for the relation $\mathrm{R}_{\mathrm{ISIS}^\infty_{n+1,m+1,q,p-1}}$ to obtain an efficient **ZKPoPK** for Regev's encryption scheme.

If a cheating prover succeeds in proving the knowledge of a plaintext $\mathbf{x} = (\mathbf{r} \| M)$, then we use the knowledge extractor to output a vector $\mathbf{x}' = (\mathbf{r}' \| M') \in \mathbb{Z}^{m+1}$ such that $\|\mathbf{x}'\|_\infty \leq 2 \cdot (p-1) - 1 = 2p-3$. In particular, we obtain $\mathbf{r}' \in \mathbb{Z}^m$ such that $\|\mathbf{r}'\|_\infty \leq 2p-3$ and $\mathbf{A}\mathbf{r}' = \mathbf{u} \bmod q$. Since $\mathbf{A}$ is chosen uniformly at random in $\mathbb{Z}_q^{n\times m}$, and the distribution of $\mathbf{u}$ is statistically close to uniform over $\mathbb{Z}_q^n$ (see [Reg09, Section 5]), the vector $\mathbf{r}'$ is a solution to the random $\mathrm{ISIS}^\infty_{n,m,q,2p-3}$ instance $(\mathbf{A}, \mathbf{u})$. This implies that the security of our **ZKPoPK** for Regev's encryption scheme relies on the assumed hardness of $\mathrm{SIVP}_{p\cdot\widetilde{O}(n)}$ (in the ℓ_2 norm).

Acknowledgements. The authors would like to thank D. Micciancio and the anonymous reviewers for their helpful comments. This work was supported by the LaBaCry MERLION grant. The research of S. Ling and H. Wang is also supported by the National Research Foundation Singapore under the Competitive Research Programme (NRF-CRP2-2007-03). Part of this research was undertaken while S. Ling was visiting ÉNS de Lyon as an invited professor. D. Stehlé was partly supported by the Australian Research Council Discovery Grant DP110100628.

References

[ABB10a] 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)

[ABB10b] Agrawal, S., Boneh, D., Boyen, X.: Lattice Basis Delegation in Fixed Dimension and Shorter-Ciphertext Hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (2010)

[AJLA+12] Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty Computation with Low Communication, Computation and Interaction via Threshold FHE. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer, Heidelberg (2012)

[Ajt96] Ajtai, M.: Generating Hard Instances of Lattice Problems (Extended Abstract). In: Proc. of STOC 1996, pp. 99–108. ACM (1996)

[Ajt99] Ajtai, M.: Generating Hard Instances of the Short Basis Problem. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999)

[AP11] Alwen, J., Peikert, C.: Generating Shorter Bases for Hard Random Lattices. Theory of Computing Systems 48(3), 535–553 (2011)

[BD10] Bendlin, R., Damgård, I.: Threshold Decryption and Zero-Knowledge Proofs for Lattice-Based Cryptosystems. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 201–218. Springer, Heidelberg (2010)

[BDOZ11] Bendlin, R., Damgård, I., Orlandi, C., Zakarias, S.: Semi-homomorphic Encryption and Multiparty Computation. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 169–188. Springer, Heidelberg (2011)

[BG93] Bellare, M., Goldreich, O.: On Defining Proofs of Knowledge. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 390–420. Springer, Heidelberg (1993)

[BNN09] Bellare, M., Namprempre, C., Neven, G.: Security Proofs for Identity-Based Identification and Signature Schemes. Journal of Cryptology 22(1), 1–61 (2009)

[Boy10] 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)

[CH91] Chaum, D., van Heyst, E.: Group Signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)

[CHKP10] 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)

[CNR12] Camenisch, J., Neven, G., Rückert, M.: Fully Anonymous Attribute Tokens from Lattices. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 57–75. Springer, Heidelberg (2012)

[DLA12] Damgård, I., López-Alt, A.: Zero-Knowledge Proofs with Low Amortized Communication from Lattice Assumptions. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 38–56. Springer, Heidelberg (2012)

[FFS87] Fiege, U., Fiat, A., Shamir, A.: Zero Knowledge Proofs of Identity. In: Proc. of STOC 1987, pp. 210–217. ACM (1987)

[FS89] Feige, U., Shamir, A.: Zero Knowledge Proofs of Knowledge in Two Rounds. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 526–544. Springer, Heidelberg (1990)

[FS90] Feige, U., Shamir, A.: Witness Indistinguishable and Witness Hiding Protocols. In: Proc. of STOC 1990, pp. 416–426. ACM (1990)

[GG98] Goldreich, O., Goldwasser, S.: On the Limits of Non-Approximability of Lattice Problems. In: Proc. of STOC 1998, pp. 1–9. ACM (1998)

[GKV10] 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)

[GMR89] Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof Systems. SIAM J. Comput. 18(1), 186–208 (1989)

[GMR05] Guruswami, V., Micciancio, D., Regev, O.: The Complexity of the Covering Radius Problem. Computational Complexity 14(2), 90–121 (2005)

[Gol04] Goldreich, O.: The Foundations of Cryptography. Basic Techniques, vol. 1. Cambridge University Press (2004)

[GPV08] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for Hard Lattices and New Cryptographic Constructions. In: Proc. of STOC 2008, pp. 197–206. ACM (2008)

[GQ90] Guillou, L.C., Quisquater, J.-J.: A "Paradoxical" Identity-Based Signature Scheme Resulting from Zero-Knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 216–231. Springer, Heidelberg (1990)

[IKOS07] Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-Knowledge from Secure Multiparty Computation. In: Proc. of STOC 2007, pp. 21–30. ACM (2007)

[JKPT12] Jain, A., Krenn, S., Pietrzak, K., Tentes, A.: Commitments and Efficient Zero-Knowledge Proofs from Learning Parity with Noise. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 663–680. Springer, Heidelberg (2012)

[KTX08] Kawachi, A., Tanaka, K., Xagawa, K.: Concurrently Secure Identification Schemes Based on the Worst-Case Hardness of Lattice Problems. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 372–389. Springer, Heidelberg (2008)

[Lyu08] Lyubashevsky, V.: Lattice-Based Identification Schemes Secure Under Active Attacks. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 162–179. Springer, Heidelberg (2008)

[Mau09] Maurer, U.: Unifying Zero-Knowledge Proofs of Knowledge. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 272–286. Springer, Heidelberg (2009)

[MP12] Micciancio, D., Peikert, C.: Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)

[MR07] Micciancio, D., Regev, O.: Worst-case to Average-case Reductions based on Gaussian Measures. SIAM J. Comput. 37(1), 267–302 (2007)

[MR09] Micciancio, D., Regev, O.: Lattice-Based Cryptography. In: Post-Quantum Cryptography, pp. 147–191. Springer (2009)

[MV03] 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)

[Pei09] Peikert, C.: Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem: Extended Abstract. In: Proc. of STOC 2009, pp. 333–342. ACM (2009)

[PV08] Peikert, C., Vaikuntanathan, V.: Noninteractive Statistical Zero-Knowledge Proofs for Lattice Problems. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 536–553. Springer, Heidelberg (2008)

[Reg05] Regev, O.: On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In: Proc. of STOC 2005, pp. 84–93. ACM (2005)

[Reg09] Regev, O.: On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. J. ACM 56(6) (2009)

[RS92] Rackoff, C., Simon, D.R.: Non-interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992)

[Rüc10a] Rückert, M.: Adaptively Secure Identity-Based Identification from Lattices without Random Oracles. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 345–362. Springer, Heidelberg (2010)

[Rüc10b] Rückert, M.: Strongly Unforgeable Signatures and Hierarchical Identity-Based Signatures from Lattices without Random Oracles. In: Sendrier, N. (ed.) PQCrypto 2010. LNCS, vol. 6061, pp. 182–200. Springer, Heidelberg (2010)

[Sha85] Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)

[SSTX09] Stehlé, D., Steinfeld, R., Tanaka, K., Xagawa, K.: Efficient Public Key Encryption Based on Ideal Lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 617–635. Springer, Heidelberg (2009)

[Ste96] Stern, J.: A New Paradigm for Public Key Identification. IEEE Transactions on Information Theory 42(6), 1757–1768 (1996)

[Vér96] Véron, P.: Improved Identification Schemes based on Error-Correcting Codes. Appl. Algebra Eng. Commun. Comput. 8(1), 57–69 (1996)

A A Zero-Knowledge Proof of Knowledge for SIS

We consider the relation associated to the $\mathrm{SIS}^{\infty}_{n,m,q,\beta}$ problem:

$$\mathrm{R}_{\mathrm{SIS}^{\infty}_{n,m,q,\beta}} = \left\{(\mathbf{A},\mathbf{x}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}^m \colon (0 < \|\mathbf{x}\|_\infty \leq \beta) \wedge (\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q)\right\}.$$

We now show how to modify the **SternExt** proof system for $\mathrm{R}_{\mathrm{ISIS}^{\infty}_{n,m,q,\beta}}$ in Section 2 to handle the additional requirement on the witness, i.e., $\mathbf{x} \neq \mathbf{0}$. In particular, the protocol must prevent a cheating prover using $\mathbf{x} = \mathbf{0}$ from passing the verification step. We look at the binary decomposition of $\mathbf{x}$, i.e., $\mathbf{x} = \sum_{j=0}^{k-1} 2^j \cdot \tilde{\mathbf{u}}_j$, and observe that $\mathbf{x} = \mathbf{0}$ is equivalent to $\forall j : \tilde{\mathbf{u}}_j = \mathbf{0}$. Our idea is to constrain the prover to prove in zero-knowledge that (at least) one of his $\tilde{\mathbf{u}}_j$'s is non-zero.

Now, observe that if $\mathbf{x} = (x_1, \ldots, x_m)$ is a valid witness for $\mathrm{R}_{\mathrm{SIS}^{\infty}_{n,m,q,\beta}}$, and 2^l is the highest power of 2 dividing $\gcd(x_1, \ldots, x_m)$, then $\mathbf{x}^* = (x_1/2^l, \ldots, x_m/2^l)$ is also a valid witness for $\mathrm{R}_{\mathrm{SIS}^{\infty}_{n,m,q,\beta}}$. Applying the binary decomposition to the vector $\mathbf{x}^*$, we note that the vector $\tilde{\mathbf{u}}_0^*$, whose coordinates are the least significant bits of $x_1/2^l, \ldots, x_m/2^l$, must be non-zero. To prove the knowledge of such a vector $\tilde{\mathbf{u}}_0^*$, the prover can use the extension trick, but in dimension $3m-1$ instead of dimension $3m$. More precisely, the prover appends $2m-1$ coordinates to $\tilde{\mathbf{u}}_0^*$ to get a vector $\mathbf{u}_0^*$ that has exactly m coordinates equal to 1; m coordinates equal to -1; and $m-1$ coordinates equal to 0. Seeing a permutation of $\mathbf{u}_0^*$ that has these constraints, the verifier will be convinced that the original vector $\tilde{\mathbf{u}}_0^*$ must have *at least* one coordinate equal to 1 or -1, and thus it must be non-zero.

In summary, the modified **SternExt** proof system for $\mathrm{R}_{\mathrm{SIS}^{\infty}_{n,m,q,\beta}}$ works as follows: The common input is a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$. The auxiliary input of the prover is $\mathbf{x}$. Prior to the interaction, both parties append $2m-1$ and $2m$ zero-columns to the matrix $\mathbf{A}$ to get a matrix $\mathbf{A}^*$, and a matrix $\mathbf{A}'$, respectively. In addition, the prover performs the following preparation steps:

- Shifting: Map $\mathbf{x}$ to $\mathbf{x}^*$, as described above.
- Binary decomposition: Write $\mathbf{x}^* = \sum_{j=0}^{k-1} 2^j \cdot \tilde{\mathbf{u}}_j^*$.
- Extensions: Append $(2m-1)$ coordinates to $\tilde{\mathbf{u}}_0^*$ as described above, and perform the usual extension to dimension $3m$ for the other vectors $\tilde{\mathbf{u}}_1^*, \ldots, \tilde{\mathbf{u}}_{k-1}^*$.

We note that $\mathbf{A}^*\mathbf{u}_0^* + \mathbf{A}'(\sum_{j=1}^{k-1} 2^j \cdot \mathbf{u}_j^*) = \mathbf{0} \bmod q$ is equivalent to $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$. Therefore, we can now apply the **SternExt** proof with a small tweak: The constraints of $\mathbf{u}_0^*$ are verified using a random permutation of $3m-1$ elements. This leads to a **ZKPoK** for the $\mathrm{SIS}^{\infty}_{n,m,q,\beta}$ problem.

Decentralized Attribute-Based Signatures

Tatsuaki Okamoto[1] and Katsuyuki Takashima[2]

[1] NTT
okamoto.tatsuaki@lab.ntt.co.jp
[2] Mitsubishi Electric
Takashima.Katsuyuki@aj.MitsubishiElectric.co.jp

Abstract. We present the first *decentralized* multi-authority attribute-based signature (DMA-ABS) scheme, in which no central authority and no trusted setup are required. The proposed DMA-ABS scheme for a large class of (non-monotone) predicates is fully secure (adaptive-predicate unforgeable and perfectly private) under a standard assumption, the decisional linear (DLIN) assumption, in the random oracle model. Our DMA-ABS scheme is comparably as efficient as the most efficient ABS scheme. As a by-product, this paper also presents an adaptively secure DMA functional encryption (DMA-FE) scheme under the DLIN assumption.

1 Introduction

1.1 Background

Recently a versatile and privacy-enhanced class of digital signatures have been studied as attribute-based signatures (ABS) [11, 14, 17, 18, 21–24, 27, 30, 32]. A signing (secret) key, $\mathsf{sk}_{\boldsymbol{x}}$, in ABS is parameterized by *attribute* $\boldsymbol{x}$, and the verification is executed using public key pk and predicate (or policy) Υ. A message m along with predicate Υ can be signed by signing key $\mathsf{sk}_{\boldsymbol{x}}$ (i.e., signature $\sigma := \mathsf{Sig}(\mathsf{sk}_{\boldsymbol{x}}, m, \Upsilon)$), if and only if $\boldsymbol{x}$ satisfies Υ. Signed message (m, Υ, σ) is verified by using public-key pk and predicate Υ, i.e., $\mathsf{Ver}(\mathsf{pk}, m, \Upsilon, \sigma) \in \{0, 1\}$. The *privacy* of a signer in this class of signatures requires that a signature (m, Υ, σ) generated by $\mathsf{sk}_{\boldsymbol{x}}$ (where $\boldsymbol{x}$ satisfies Υ) release no information regarding $\boldsymbol{x}$ except that $\boldsymbol{x}$ satisfies Υ.

There are many applications of ABS such as attribute-based messaging (ABM), attribute-based authentication, trust-negotiation and leaking secrets (see [24] for more details). For example, in a country (say country U), public comments on a new government's policy on scientific research are widely requested, especially to a class of people who should be responsible or heavily related to this topic from academia, government and industries. Comments from this class of people are requested to be signed (authenticated) to prove that the comments are from such people. In addition, the privacy of the people who send comments should be ensured. So there are contradictory requirements on authentication and privacy. The concept of ABS provides a nice solution to this type of problems. For example, a professor of University A sends a comment signed through ABS with a predicate

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 125–142, 2013.

such that ((Affiliation = University A OR B OR C) AND (Position = Professor OR Lecturer)) OR ((Affiliation = Government of Country U) AND (Qualification = PhD)) OR ((Affiliation = Company X OR Y OR Z) AND (Position = Chief Scientist OR Senior Manager)). A recipient of this signed comment can confirm that the signer of this comment is from the class of people, and the privacy is also preserved since there are too many people who satisfy the predicate and it is hard to identify the actual signer among so many possible signers due to the privacy condition of ABS.

The basic concept of ABS, however, has a serious problem that only a single authority exists in a system. Therefore, the single authority should issue to all users their secret keys (certificates/credentials) associated with all attributes in the system, i.e., all positions of all organizations (e.g., all positions of Universities A, B and C, Governments of Countries U, V and W, and Companies X, Y and Z). If the party is corrupted, the system will be totally broken.

To overcome the drawback, the concept of *multi-authority* (MA-)ABS, was introduced [23, 24, 27], in which there are multiple authorities and each authority is responsible for issuing a secret key associated with a category or sub-universe of attributes, i.e., a user obtains several secret keys, each of which is issued by each authority. For example, a professor of university A obtains a secret key (for the position) from university A, a secret key for the citizenship from country U, and a secret key for a consultant position from company X, where university A, country U and company X are individual authorities. An important requirement for MA-ABS is the security (unforgeability) against collusion attacks. For example, it is required that a professor of university A, Alice, with a secret key for her position and a student, Bob, with a secret key for his citizenship of country W cannot collude to forge a signature endorsed by a professor of university A with the citizenship of country W.

The existing MA-ABS schemes, however, still have a problem that a special central authority is required in addition to multiple authorities regarding attributes, and if the central authority is corrupted, the security (unforgeability) of the system will be totally broken. As a typical example, we show in the full version of this paper [26] that all MA-ABS schemes in [24] will be totally broken if the central authority is corrupted.

Any MA-ABS scheme with no central authority, *decentralized* MA-ABS (DMA-ABS) scheme, has not been proposed.

Recently, Lewko and Waters [20] presented the first DMA system for attribute-based encryption (ABE) (but not for ABS). Their scheme, however, still has a problem. It requires a trusted setup of a parameter, composite number $N := p_1p_2p_3$ (p_1, p_2, p_3 are primes) and a generator g_1 of secret subgroup G_{p_1}. That is, there exists a trapdoor, (p_1, p_2, p_3), and the security of the system will not be guaranteed by the security proof, if the trapdoor is compromised. In other words, their system requires a trusted setup. A generic conversion method from a composite-order-group-based system to a prime-order-group-based system has been presented by Lewko [19] and it may be applicable to the DMA-ABE scheme.

1.2 Our Results

- This paper proposes the first DMA-ABS scheme, which supports a large class of relations, non-monotone access structures, in which no central authority exists and no global coordination is required except for the setting of a parameter for a prime order bilinear group and hash functions. Note that parameters for a prime order bilinear group on supersingular and some ordinary elliptic curves and specification of hash functions such as the SHA families can be available from public documents, e.g., ISO and FIPS official documents [13, 16] and [12], or can be included in the specification of the scheme. That is, no trusted setup is necessary in the proposed DMA-ABS system.

  In the proposed DMA-ABS schemes, every process can be executed in a decentralized manner; any party can become an authority and issue a (piece of a) secret key to a user without interacting with any other party, and each user obtains a (piece of a) secret key from the associated authority without interacting with any other party. While enjoying such decentralized processes, the proposed schemes are still secure against collusion attacks. i.e., multiple pieces issued to a user by different authorities can form a (collusion resistant) single secret key, composed of the pieces, of the user.
- This paper also proposes a more general signature scheme, DMA functional signature (FS) scheme, which supports more general predicates, non-monotone access structures combined with inner-product relations [25]. The proposed DMA-ABS scheme is a special case of the DMA-FS scheme, where the underlying inner-product relations are specialized to be two-dimensional inner-product relations for equality.

  Remark: The general relations (non-monotone access structures combined with inner-product relations [25]) supported by the proposed DMA-FS scheme are: $\boldsymbol{x} := (\vec{x}_1, \ldots, \vec{x}_i) \in \mathbb{F}_q^{n_1+\cdots+n_i}$ for verification, and $\Upsilon := (\hat{M}, (\vec{v}_1, \ldots, \vec{v}_i) \in \mathbb{F}_q^{n_1+\cdots+n_i})$ for a secret key. The component-wise inner-product relations for attribute vector components, e.g., $\{\vec{x}_t \cdot \vec{v}_t = 0 \text{ or not }\}_{t\in\{1,\ldots,i\}}$, are input to span program $\hat{M}$, and $\boldsymbol{x}$ satisfies Υ iff the truth-value vector of $(\mathsf{T}(\vec{x}_1 \cdot \vec{v}_1 = 0), \ldots, \mathsf{T}(\vec{x}_i \cdot \vec{v}_i = 0))$ is accepted by span program $\hat{M}$. If the DMA-FS is specialized to DMA-ABS, then $n_t := 2$, i.e., $\vec{x}_t := (1, x_t)$ and $\vec{v}_t := (v_t, -1)$, where $\vec{x}_t \cdot \vec{v}_t = 0$ iff $x_t = v_t$.
- This paper proves that the proposed DMA-FS scheme is fully secure (adaptive-predicate unforgeable and perfectly private in the DMA security model) under the DLIN assumption in the random oracle model. It implies that the proposed DMA-ABS scheme is fully secure under the DLIN assumption in the random oracle model.
- The efficiency of the DMA-ABS scheme is comparable to those of the existing ABS schemes (e.g., [24, 27]). See Table 1 in Section 4.5.
- Although the main aim of this paper is to propose the first DMA-ABS scheme, there is a by-product, a new DMA-FE (or DMA-ABE) scheme, which is an adaptively secure DMA-FE scheme without a trusted setup under the DLIN assumption in the random oracle model.

Our DMA-ABS scheme is considered to be a natural extension of *ring signatures* [28, 29]. In ring signatures, no central authority and no trusted setup are required and every process is fully distributed. Our DMA-ABS also requires no central authority and no trusted setup and every process is fully distributed. In other words, ring signatures are a very special case of our DMA-ABS where the underlying predicate is just a disjunction and each authority is a user in a ring. For many applications of ring signatures, our DMA-ABS is more suitable. For example, in an application to whistle-blowing, an expose document on a financial scandal to a newspaper company would be better to be endorsed by someone with certain possible positions and qualifications related to the scandal than by someone in a list of real persons.

1.3 Key Techniques

There are two major requirements for DMA-ABS, *(collusion resistant) unforgeability* and *privacy* in the decentralized multi-authority model. Our target is to construct a DMA-ABS scheme that is secure (unforgeable and private in the decentralized multi-authority model) under a *standard assumption*, the DLIN assumption. It is a challenging task even in the random oracle model. For some notations hereafter, see Section 1.5.

To realize such a DMA-ABS scheme, the top level strategy is based on Naor's paradigm [4], which is originally a conversion from identity-based encryption (IBE) to (ordinary) digital signatures, but in our case, an encryption counterpart, DMA-ABE, is converted to DMA-ABS. Therefore, DMA-ABE scheme is designed first, and then DMA-ABS is constructed on it.

To construct a DMA-ABE (or more generally DMA-FE) scheme for this purpose, we follow several established key ideas; dual pairing vector spaces (DPVS) [25, 27], global identifier gid [9], (random oracle) hashing of gid [20], dual system encryption [20, 31], and the linear transformation technique to produce $(\delta\vec{x}_t, \ldots)_{\mathbb{B}_t^*}$ by using X_t (the master secret key of authority t) and $\delta G := H(\mathsf{gid}) \in \mathbb{G}$ [27], which is essentially different from the technique using $H(\mathsf{gid})$ in [20] (see Section 4.3 for the details). Note that, although our design strategy is based on Naor's paradigm, this paper directly proves the security of the proposed DMA-ABS scheme from the DLIN assumption.

A specific *central* space, $\mathbb{V}_0$ ($t = 0$), played an essential role in the security proof (based on the dual system encryption technique) of previous ABS and ABE (FS and FE) schemes in [25, 27]. No such a central space, however, is allowed in our DMA setting, where only spaces, $\mathbb{V}_t$ ($t = 1, \ldots$), generated by decentralized authorities are available. A crucial part of the key techniques in our DMA-ABS and DMA-ABE (DMA-FS and DMA-FE) schemes is to distribute the dual system encryption trick for the central space in the previous schemes into all the spaces.

More precisely, the secret-key and verification-text (where the negative term case in the span program, i.e., $\rho(i) = \neg(t, \vec{v}_i)$, is used, for simplicity of expression) are of the forms of $(\vec{x}_t, \delta\vec{x}_t, 0^{n_t}, 0^{n_t}, \ldots)_{\mathbb{B}_t^*}$ and $(s_i\vec{v}_i, s_i'\vec{v}_i, 0^{n_t}, 0^{n_t}, \ldots)_{\mathbb{B}_t}$, respectively. Here, s_i and s_i' are shares from an access structure with a signature.

Subspaces with $\{s_i\vec{v}_i\}$ and $\{\vec{x}_t\}$ are used for verification (or decryption), and subspaces with $\{s'_i\vec{v}_i\}$ and $\{\delta\vec{x}_t\}$ are for the *distributed* dual system encryption trick. To execute the trick over the subspaces, we develop a new technique, *swap and conceptual change*, in which 4-dimensional (in DMA-FS and DMA-FE, $2n_t$-dimensional) hidden subspaces are employed for *semi-functional* forms of secret-keys and verification-texts. In the previous dual system encryption tricks [25, 27], the semi-functional form of secret-keys and verification-texts in a central space $\mathbb{V}_0$ ($t = 0$) played a key role. In our *distributed* dual system encryption trick, the left 2-dimensional subspaces in the 4-dimensional hidden subspaces are used for a computational change of secret-keys from DLIN and a conceptual change on key query restrictions. The right 2-dimensional subspaces are swapped with the left ones through a computational change from DLIN, and these subspaces for all $\mathbb{V}_t$ ($t = 1, \ldots$) play the key role in a distributed manner that corresponds to that of $\mathbb{V}_0$ ($t = 0$) in the previous schemes (see the full version [26]).

A new idea is also required to achieve the *privacy* condition for DMA-ABS, since no privacy condition is required for DMA-ABE or included in Naor's paradigm. Moreover, a *new re-randomization* technique should be developed in this paper to achieve the privacy of DMA-ABS, since the re-randomization technique for privacy in [27] is not effective in the DMA-ABS setting due to the fully distributed structure (see Section 4.2).

For more details on the techniques in the security proofs of DMA-ABS, see the full version [26].

1.4 Related Works

1. The *mesh signatures* [5] are a variation of ring signatures, where the predicate is an access structure on a list of pairs comprising a message and public key (m_i, pk_i), and a valid mesh signature can be generated by a person who has enough standard signatures σ_i on m_i, each valid under pk_i, to satisfy the given access structure.

   A crucial difference between mesh signatures and DMA-ABS is the security against the collusion of users. In mesh signatures, several users can collude by pooling their signatures together and create signatures that none of them could produce individually. That is, such collusion is considered to be legitimate in mesh signatures. In contrast, the security against collusion attacks is one of the basic requirements in ABS and DMA-ABS.
2. Another related concept is *anonymous credentials (ACs)* [2, 3, 6–8, 10]. The notion of ACs also provides a functionality for users to demonstrate anonymously possession of attributes, but the goals of ACs and (DMA-)ABS differ in several points.

   As described in [24], ACs and (DMA-)ABS aim at different goals: ACs target very strong anonymity even in the registration phase, whereas under less demanding anonymity requirements in the registration phase, (DMA-)ABS aims to achieve more expressive functionalities, more efficient constructions

and new applications. In addition, (DMA-)ABS is a signature scheme and a simpler primitive compared with ACs. See the full version of this paper [26] for more details.

1.5 Notations

When A is a random variable or distribution, $y \xleftarrow{\mathsf{R}} A$ denotes that y is randomly selected from A according to its distribution. When A is a set, $y \xleftarrow{\mathsf{U}} A$ denotes that y is uniformly selected from A. We denote the finite field of order q by $\mathbb{F}_q$, and $\mathbb{F}_q \setminus \{0\}$ by $\mathbb{F}_q^\times$. A vector symbol denotes a vector representation over $\mathbb{F}_q$, e.g., $\vec{x}$ denotes $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$. For two vectors $\vec{x} = (x_1, \ldots, x_n)$ and $\vec{v} = (v_1, \ldots, v_n)$, $\vec{x} \cdot \vec{v}$ denotes the inner-product $\sum_{i=1}^n x_i v_i$. The vector $\vec{0}$ is abused as the zero vector in $\mathbb{F}_q^n$ for any n. X^{T} denotes the transpose of matrix X. I_ℓ and 0_ℓ denote the $\ell \times \ell$ identity matrix and the $\ell \times \ell$ zero matrix, respectively. A bold face letter denotes an element of vector space $\mathbb{V}$, e.g., $\boldsymbol{x} \in \mathbb{V}$. When $\boldsymbol{b}_i \in \mathbb{V}$ $(i = 1, \ldots, n)$, $\mathsf{span}\langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \rangle \subseteq \mathbb{V}$ (resp. $\mathsf{span}\langle \vec{x}_1, \ldots, \vec{x}_n \rangle$) denotes the subspace generated by $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ (resp. $\vec{x}_1, \ldots, \vec{x}_n$). For bases $\mathbb{B} := (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_N)$ and $\mathbb{B}^* := (\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_N^*)$, $(x_1, \ldots, x_N)_{\mathbb{B}} := \sum_{i=1}^N x_i \boldsymbol{b}_i$ and $(y_1, \ldots, y_N)_{\mathbb{B}^*} := \sum_{i=1}^N y_i \boldsymbol{b}_i^*$. For a format of attribute vectors $\vec{n} := (d; n_1, \ldots, n_d)$ that indicates dimensions of vector spaces, $\vec{e}_{t,j}$ denotes the canonical basis vector $(\overbrace{0 \cdots 0}^{j-1}, 1, \overbrace{0 \cdots 0}^{n_t - j}) \in \mathbb{F}_q^{n_t}$ for $t = 1, \ldots, d$ and $j = 1, \ldots, n_t$. $GL(n, \mathbb{F}_q)$ denotes the general linear group of degree n over $\mathbb{F}_q$.

2 Dual Pairing Vector Spaces by Direct Product of Symmetric Pairing Groups

Definition 1. *"Symmetric bilinear pairing groups"* $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ *are a tuple of a prime* q*, cyclic additive group* $\mathbb{G}$ *and multiplicative group* $\mathbb{G}_T$ *of order* q*,* $G \neq 0 \in \mathbb{G}$*, and a polynomial-time computable nondegenerate bilinear pairing* $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ *i.e.,* $e(sG, tG) = e(G, G)^{st}$ *and* $e(G, G) \neq 1$*. Let* $\mathcal{G}_{\mathsf{bpg}}$ *be an algorithm that takes input* 1^λ *and outputs a description of bilinear pairing groups* $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ *with security parameter* λ*.*

Definition 2. *"Dual pairing vector spaces (DPVS)"* $(q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e)$ *by a direct product of symmetric pairing groups* $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ *are a tuple of prime* q*,* N*-dimensional vector space* $\mathbb{V} := \overbrace{\mathbb{G} \times \cdots \times \mathbb{G}}^{N}$ *over* $\mathbb{F}_q$*, cyclic group* $\mathbb{G}_T$ *of order* q*, canonical basis* $\mathbb{A} := (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N)$ *of* $\mathbb{V}$*, where* $\boldsymbol{a}_i := (\overbrace{0, \ldots, 0}^{i-1}, G, \overbrace{0, \ldots, 0}^{N-i})$*, and pairing* $e : \mathbb{V} \times \mathbb{V} \to \mathbb{G}_T$*. (Symbol* e *is abused as pairing for* $\mathbb{G}$ *and for* $\mathbb{V}$*.) The pairing is defined by* $e(\boldsymbol{x}, \boldsymbol{y}) := \prod_{i=1}^N e(G_i, H_i) \in \mathbb{G}_T$ *where* $\boldsymbol{x} := (G_1, \ldots, G_N) \in \mathbb{V}$ *and* $\boldsymbol{y} := (H_1, \ldots, H_N) \in \mathbb{V}$*. This is nondegenerate bilinear i.e.,* $e(s\boldsymbol{x}, t\boldsymbol{y}) = e(\boldsymbol{x}, \boldsymbol{y})^{st}$ *and if* $e(\boldsymbol{x}, \boldsymbol{y}) = 1$ *for all* $\boldsymbol{y} \in \mathbb{V}$*, then* $\boldsymbol{x} = \boldsymbol{0}$*. For all*

i and j, $e(\boldsymbol{a}_i, \boldsymbol{a}_j) = e(G,G)^{\delta_{i,j}}$ where $\delta_{i,j} = 1$ if $i = j$, and 0 otherwise, and $e(G,G) \neq 1 \in \mathbb{G}_T$. DPVS generation algorithm $\mathcal{G}_{\mathsf{dpvs}}$ takes input 1^λ ($\lambda \in \mathbb{N}$) and $N \in \mathbb{N}$, and outputs a description of $\mathsf{param}_{\mathbb{V}} := (q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e)$ with security parameter λ and N-dimensional $\mathbb{V}$. It can be constructed by using $\mathcal{G}_{\mathsf{bpg}}$.

For the asymmetric version of DPVS, $(q, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \mathbb{A}, \mathbb{A}^*, e)$, see Appendix A.2 in the full version of [25].

3 Non-monotone Access Structures with Inner-Product Relations

3.1 Span Programs and Non-monotone Access Structures

Definition 3 (Span Programs [1]). *Let $\{p_1, \ldots, p_n\}$ be a set of variables. A span program over $\mathbb{F}_q$ is a labeled matrix $\hat{M} := (M, \rho)$ where M is a $(\ell \times r)$ matrix over $\mathbb{F}_q$ and ρ is a labeling of the rows of M by literals from $\{p_1, \ldots, p_n, \neg p_1, \ldots, \neg p_n\}$ (every row is labeled by one literal), i.e., $\rho : \{1, \ldots, \ell\} \to \{p_1, \ldots, p_n, \neg p_1, \ldots, \neg p_n\}$. A span program accepts or rejects an input by the following criterion. For every input sequence $\delta \in \{0,1\}^n$ define the submatrix M_δ of M consisting of those rows whose labels are set to 1 by the input δ, i.e., either rows labeled by some p_i such that $\delta_i = 1$ or rows labeled by some $\neg p_i$ such that $\delta_i = 0$. (i.e., $\gamma : \{1, \ldots, \ell\} \to \{0,1\}$ is defined by $\gamma(j) = 1$ if $[\rho(j) = p_i] \wedge [\delta_i = 1]$ or $[\rho(j) = \neg p_i] \wedge [\delta_i = 0]$, and $\gamma(j) = 0$ otherwise. $M_\delta := (M_j)_{\gamma(j)=1}$, where M_j is the j-th row of M.)*

The span program $\hat{M}$ accepts δ if and only if $\vec{1} \in \mathsf{span}\langle M_\delta \rangle$, i.e., some linear combination of the rows of M_δ gives the all one vector $\vec{1}$. (The row vector has the value 1 in each coordinate.) A span program computes a Boolean function f if it accepts exactly those inputs δ where $f(\delta) = 1$.

A span program is called monotone if the labels of the rows are only the positive literals $\{p_1, \ldots, p_n\}$. Monotone span programs compute monotone functions. (So, a span program in general is "non"-monotone.)

We assume that no row M_i ($i = 1, \ldots, \ell$) of the matrix M is $\vec{0}$. We now introduce a non-monotone access structure with evaluating map γ by using the inner-product of attribute vectors, that is employed in the proposed DMA-ABS (and DMA-FS, DMA-FE) scheme.

Definition 4 (Inner-Products of Attribute Vectors and Access Structures). *$\mathcal{U}_t$ ($t = 1, \ldots, d$ and $\mathcal{U}_t \subset \{0,1\}^*$) is a sub-universe, a set of attributes, each of which is expressed by a pair of sub-universe id and n_t-dimensional vector, i.e., $(t, \vec{v})$, where $t \in \{1, \ldots, d\}$ and $\vec{v} \in \mathbb{F}_q^{n_t} \setminus \{\vec{0}\}$.*

We now define such an attribute to be a variable p of a span program $\hat{M} := (M, \rho)$, i.e., $p := (t, \vec{v})$. An access structure $\mathbb{S}$ is a span program $\hat{M} := (M, \rho)$ along with variables $p := (t, \vec{v}), p' := (t', \vec{v}'), \ldots$, i.e., $\mathbb{S} := (M, \rho)$ such that $\rho : \{1, \ldots, \ell\} \to \{(t, \vec{v}), (t', \vec{v}'), \ldots, \neg(t, \vec{v}), \neg(t', \vec{v}'), \ldots\}$. Let Γ be a set of attributes,

i.e., $\Gamma := \{(t, \vec{x}_t) \mid \vec{x}_t \in \mathbb{F}_q^{n_t} \setminus \{\vec{0}\}, 1 \le t \le d\}$*, where* t *runs through some subset of* $\{1, \ldots, d\}$*, not necessarily the whole indices.*

When Γ *is given to access structure* $\mathbb{S}$*, map* $\gamma : \{1, \ldots, \ell\} \to \{0, 1\}$ *for span program* $\hat{M} := (M, \rho)$ *is defined as follows: For* $i = 1, \ldots, \ell$*, set* $\gamma(i) = 1$ *if* $[\rho(i) = (t, \vec{v}_i)] \wedge [(t, \vec{x}_t) \in \Gamma] \wedge [\vec{v}_i \cdot \vec{x}_t = 0]$ *or* $[\rho(i) = \neg(t, \vec{v}_i)] \wedge [(t, \vec{x}_t) \in \Gamma] \wedge [\vec{v}_i \cdot \vec{x}_t \neq 0]$*. Set* $\gamma(i) = 0$ *otherwise.*

Access structure $\mathbb{S} := (M, \rho)$ *accepts* Γ *iff* $\vec{1} \in \mathsf{span}\langle (M_i)_{\gamma(i)=1} \rangle$*.*

Remark 1. The simplest form of the inner-product relations in the above-mentioned access structures, that is for ABS and ABE, is a special case when $n_t = 2$ for all $t \in \{1, \ldots, d\}$, and $\vec{x} := (1, x)$ and $\vec{v} := (v, -1)$. Hence, $(t, \vec{x}_t) := (t, (1, x_t))$ and $(t, \vec{v}_i) := (t, (v_i, -1))$, but we often denote them shortly by (t, x_t) and (t, v_i). Then, $\mathbb{S} := (M, \rho)$ such that $\rho : \{1, \ldots, \ell\} \to \{(t, v), (t', v'), \ldots \neg(t, v), \neg(t', v'), \ldots\}$ $(v, v', \ldots \in \mathbb{F}_q)$, and $\Gamma := \{(t, x_t) \mid x_t \in \mathbb{F}_q, 1 \le t \le d\}$.

When Γ is given to access structure $\mathbb{S}$, map $\gamma : \{1, \ldots, \ell\} \to \{0, 1\}$ for span program $\hat{M} := (M, \rho)$ is defined as follows: For $i = 1, \ldots, \ell$, set $\gamma(i) = 1$ if $[\rho(i) = (t, v_i)] \wedge [(t, x_t) \in \Gamma] \wedge [v_i = x_t]$ or $[\rho(i) = \neg(t, v_i)] \wedge [(t, x_t) \in \Gamma] \wedge [v_i \neq x_t]$. Set $\gamma(i) = 0$ otherwise.

Remark 2. When a user has multiple attributes in a sub-universe (category) t, we can employ dimension $n_t > 2$. For instance, a professor (say Alice) in the science faculty of a university is also a professor in the engineering faculty of this university. If the attribute authority of this university manages sub-universe $t :=$ "faculties of this university", Alice obtains a secret key for $(t, \vec{x}_t := (1, -(a + b), ab) \in \mathbb{F}_q^3)$ with $a :=$ "science" and $b :=$ "engineering" from the authority. When a user verifies a signature for an access structure with a single negative attribute $\neg(t,$ "science"$)$, the verification text is encoded as $\neg(t, \vec{v}_i := (a^2, a, 1))$ with $a :=$ "science". Since $\vec{x}_t \cdot \vec{v}_i = 0$, Alice cannot make a valid signature for an access structure with the negative attribute $\neg(t,$ "science"$)$. For such a case with $n_t > 2$, see the full version [26] with our DMA-FS scheme.

We now construct a secret-sharing scheme for a span program.

Definition 5. *A secret-sharing scheme for span program* $\hat{M} := (M, \rho)$ *is:*

1. *Let* M *be* $\ell \times r$ *matrix. Let column vector* $\vec{f}^{\mathrm{T}} := (f_1, \ldots, f_r)^{\mathrm{T}} \xleftarrow{\cup} \mathbb{F}_q^r$*. Then,* $s_0 := \vec{1} \cdot \vec{f}^{\mathrm{T}} = \sum_{k=1}^{r} f_k$ *is the secret to be shared, and* $\vec{s}^{\mathrm{T}} := (s_1, \ldots, s_\ell)^{\mathrm{T}} := M \cdot \vec{f}^{\mathrm{T}}$ *is the vector of* ℓ *shares of the secret* s_0 *and the share* s_i *belongs to* $\rho(i)$*.*
2. *If span program* $\hat{M} := (M, \rho)$ *accept* δ*, or access structure* $\mathbb{S} := (M, \rho)$ *accepts* Γ*, i.e.,* $\vec{1} \in \mathsf{span}\langle (M_i)_{\gamma(i)=1} \rangle$ *with* $\gamma : \{1, \ldots, \ell\} \to \{0, 1\}$*, then there exist constants* $\{\alpha_i \in \mathbb{F}_q \mid i \in I\}$ *such that* $I \subseteq \{i \in \{1, \ldots, \ell\} \mid \gamma(i) = 1\}$ *and* $\sum_{i \in I} \alpha_i s_i = s_0$*. Furthermore, these constants* $\{\alpha_i\}$ *can be computed in time polynomial in the size of matrix* M*.*

4 Decentralized Multi-Authority Attribute-Based Signatures (DMA-ABS)

4.1 Definitions for DMA-ABS

Definition 6 (Decentralized Multi-Authority ABS : DMA-ABS). *A decentralized multi-authority ABS scheme consists of the following algorithms/ protocols.*

GSetup *A party runs the algorithm* $\mathsf{GSetup}(1^\lambda)$ *which outputs a global parameter* gparam. *The party publishes* gparam.

ASetup *An attribute authority* t $(1 \leq t \leq d)$ *who wishes to issue attributes runs* $\mathsf{ASetup}(\mathsf{gparam}, t, n_t)$ *which outputs an attribute-authority public key* apk_t *and an attribute-authority secret key* ask_t. *The attribute authority* t *publishes* apk_t *and stores* ask_t.

AttrGen *When an attribute authority* t *issues user* gid *a secret key associated with an attribute* x_t, *it runs* $\mathsf{AttrGen}(\mathsf{gparam}, t, \mathsf{ask}_t, \mathsf{gid}, x_t)$ *that outputs an attribute secret key* $\mathsf{usk}_{\mathsf{gid},(t,x_t)}$. *The attribute authority gives* $\mathsf{usk}_{\mathsf{gid},(t,x_t)}$ *to the user.*

Sig *This is a randomized algorithm. A user signs message* m *with claim-predicate (access structure)* $\mathbb{S} := (M, \rho)$, *only if there is a set of attributes* Γ *such that* $\mathbb{S}$ *accepts* Γ, *the user has obtained a set of keys* $\{\mathsf{usk}_{\mathsf{gid},(t,x_t)} \mid (t, x_t) \in \Gamma\}$ *from the attribute authorities. Then signature* σ *can be generated using* $\mathsf{Sig}(\mathsf{gparam}, \{\mathsf{apk}_t, \mathsf{usk}_{\mathsf{gid},(t,x_t)}\}, m, \mathbb{S})$, *where* $\mathsf{usk}_{\mathsf{gid},(t,x_t)} \xleftarrow{\mathsf{R}} \mathsf{AttrGen}(\mathsf{gparam}, t, \mathsf{ask}_t, \mathsf{gid}, x_t)$.

Ver *To verify signature* σ *on message* m *with claim-predicate (access structure)* $\mathbb{S}$, *using a set of public keys for relevant authorities* $\{\mathsf{apk}_t\}$, *a user runs* $\mathsf{Ver}(\mathsf{gparam}, \{\mathsf{apk}_t\}, m, \mathbb{S}, \sigma)$ *which outputs boolean value* $\mathsf{accept} := 1$ *or* $\mathsf{reject} := 0$.

Definition 7 (Perfect Privacy of DMA-ABS). *A DMA-ABS scheme is perfectly private, if, for all* $\mathsf{gparam} \xleftarrow{\mathsf{R}} \mathsf{GSetup}(1^\lambda)$, *for all* $(\mathsf{ask}_t, \mathsf{apk}_t) \xleftarrow{\mathsf{R}} \mathsf{ASetup}(\mathsf{gparam}, t)$ $(1 \leq t \leq d)$, *all messages* m, *all attribute sets* Γ_1 *associated with* gid_1 *and* Γ_2 *associated with* gid_2, *all signing keys* $\{\mathsf{usk}_{t,1} \xleftarrow{\mathsf{R}} \mathsf{AttrGen}(\mathsf{gparam}, t, \mathsf{ask}_t, \mathsf{gid}_1, x_{t,1})\}_{(t,x_{t,1})\in\Gamma_1}$ *and* $\{\mathsf{usk}_{t,2} \xleftarrow{\mathsf{R}} \mathsf{AttrGen}(\mathsf{gparam}, t, \mathsf{ask}_t, \mathsf{gid}_2, x_{t,2})\}_{(t,x_{t,2})\in\Gamma_2}$, *all access structures* $\mathbb{S}$ *such that* $\mathbb{S}$ *accepts* Γ_1 *and* $\mathbb{S}$ *accepts* Γ_2, *the distributions* $\mathsf{Sig}(\mathsf{gparam}, \{\mathsf{apk}_t, \mathsf{usk}_{t,1} \mid (t, x_{t,1}) \in \Gamma_1\}, m, \mathbb{S})$ *and* $\mathsf{Sig}(\mathsf{gparam}, \{\mathsf{apk}_t, \mathsf{usk}_{t,2} \mid (t, x_{t,2}) \in \Gamma_2\}, m, \mathbb{S})$ *are equal.*

Note that the above definition of perfect privacy is weaker than that in [24], since the attribute authorities are assumed to be honest in our definition, while they can be malicious in [24].

For a DMA-ABS scheme with perfect privacy, we define algorithm $\mathsf{AltSig}(\mathsf{gparam}, \{\mathsf{apk}_t, \mathsf{ask}_t\}, m, \mathbb{S})$ with $\mathbb{S}$ and master key ask_t instead of Γ and $\{\mathsf{usk}_{\mathsf{gid},(t,x_t)}\}_{(t,x_t)\in\Gamma}$: First, generate $\mathsf{usk}_{\mathsf{gid},(t,x_t)} \xleftarrow{\mathsf{R}} \mathsf{AttrGen}(\mathsf{gparam}, t, \mathsf{ask}_t, \mathsf{gid}, x_t)$ for arbitrary $\Gamma := \{(t, x_t)\}$ which satisfies $\mathbb{S}$, then $\sigma \xleftarrow{\mathsf{R}} \mathsf{Sig}(\mathsf{gparam}, \{\mathsf{apk}_t, \mathsf{usk}_{\mathsf{gid},(t,x_t)}\}, m, \mathbb{S})$. Return σ.

Let T be the set of authorities. We assume each attribute is assigned to one authority.

Definition 8 (Unforgeability of DMA-ABS). *For an adversary $\mathcal{A}$, we define $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DMA\text{-}ABS},\mathsf{UF}}(\lambda)$ to be the success probability in the following experiment for any security parameter λ. A DMA-ABS scheme is unforgeable if the success probability of any polynomial-time adversary $\mathcal{A}$ is negligible:*

1. *Run* $\mathsf{gparam} \xleftarrow{\mathsf{R}} \mathsf{GSetup}(1^\lambda)$ *and give* gparam *to adversary $\mathcal{A}$. For authorities $t \in T$, run* $(\mathsf{ask}_t, \mathsf{apk}_t) \xleftarrow{\mathsf{R}} \mathsf{ASetup}(\mathsf{gparam})$ *and give* $\{\mathsf{apk}_t\}_{t\in T}$ *to $\mathcal{A}$. Adversary $\mathcal{A}$ specifies a set $\widetilde{T} \subset T$ of corrupt attribute authorities, and gets* $\{\mathsf{ask}_t\}_{t\in\widetilde{T}}$.
2. *The adversary $\mathcal{A}$ is given access to oracles* $\mathsf{AttrGen}$ *and* AltSig *with queries including attribute authorities, t, from $S := T \setminus \widetilde{T}$ alone.*
3. *At the end, the adversary outputs $(m', \mathbb{S}', \sigma')$.*

Let $\Gamma_{\mathsf{gid}} := \{(t, x_t) \mid (t \in S,\ x_t,\ \mathsf{gid})$ *is queried to* $\mathsf{AttrGen}$ *oracle by $\mathcal{A}$}$\}$. *We say the adversary succeeds, if $(m', \mathbb{S}')$ was never queried to* AltSig *oracle, $\mathbb{S}'$ does not accept Γ_{gid} for any* gid, *$\mathbb{S}'$ includes attributes authorities, t, from S alone, and* $\mathsf{Ver}(\mathsf{pk}, m', \mathbb{S}', \sigma') = 1$.

Remark 3. The unforgeability defined above ensures that adversary $\mathcal{A}$ cannot forge a signature regarding uncorrupt authorities even if $\mathcal{A}$ makes key and signature queries to uncorrupt authorities. That is, the forging capability of any $\mathcal{A}$ is limited or localized to that of corrupt authorities as expected in DMA schemes (in contrast, it can be expanded to the whole system in MA schemes).

The model regarding *corrupt authorities* in this definition, however, is weaker than that in [24]. Roughly, the security on this model implies that no adversary $\mathcal{A}$ can forge a signature with a predicate $\mathbb{S}'_S$ unless $\mathcal{A}$ issues key queries for Γ_S such that $\mathbb{S}'_S$ accepts Γ_S, where $\mathbb{S}'_S$ and Γ_S are a predicate and attributes including uncorrupt parties from S alone. On the other hand, the security on the model in [24] implies that no adversary $\mathcal{A}$ can forge a signature with a predicate $\mathbb{S}'_{S\cup\widetilde{T}}$ unless $\mathcal{A}$ issues key queries for Γ_S such that, for some $\Gamma_{\widetilde{T}}$, $\mathbb{S}'_{S\cup\widetilde{T}}$ accepts $(\Gamma_S \cup \Gamma_{\widetilde{T}})$. Namely, the scope of forgery in [24] is wider (i.e., it covers a policy over $S \cup \widetilde{T}$) than that in our definition (i.e., it is limited to a policy over S). [1]

4.2 Construction Idea of the Proposed DMA-ABS Scheme

Here we will show some basic idea to construct the proposed DMA-ABS scheme, which is designed on the DMA-FE scheme (Appendix A) through Naor's paradigm. For the key techniques to construct DMA-FE from (non-decentralized) FE [25], we refer to Section 1.3. In the paradigm, collusion-resistant identity-based encryption (IBE) is transformed to unforgeable signatures, where (a hash

[1] The proposed scheme in this paper has been proven unforgeable only in our model due to some technical reason caused by no trusted setup (or no trapdoor) of our scheme.

value of) a message is used for an identity in IBE. To realize the Naor-like transformation in our DMA-FE, two-dimensional subspaces $\mathsf{span}\langle \boldsymbol{b}_{t,5}, \boldsymbol{b}_{t,6}\rangle$ (and their dual subspaces) are newly added for identity (message) embedding to all spaces $\mathbb{V}_t$ for $t > 0$. Note that the privacy condition is not included in Naor's paradigm.

In our variant of Naor's paradigm, a secret signing key sk_Γ with attribute set Γ and a verification text $\vec{\boldsymbol{c}}$ with access structure $\mathbb{S}$ (for signature verification) in our DMA-ABS scheme correspond to a secret decryption key sk_Γ with Γ and a ciphertext $\vec{\boldsymbol{c}}$ with $\mathbb{S}$ in the DMA-FE scheme, respectively. No counterpart of a signature $\vec{\boldsymbol{s}}^*$ in the DMA-ABS exists in the DMA-FE, and the privacy property for signature $\vec{\boldsymbol{s}}^*$ is also specific in DMA-ABS. Signature $\vec{\boldsymbol{s}}^*$ in DMA-ABS may be interpreted to be a decryption key specialized to decrypt a ciphertext with access structure $\mathbb{S}$, that is delegated from secret key sk_Γ. The algorithms of the proposed DMA-ABS scheme can be described in the light of such correspondence to the DMA-FE scheme:

GSetup. Almost the same as that in the DMA-FE scheme except that a hash function, H_2, is added in gparam. This is used for hashing of message and access structure in the signing and verification algorithms.

ASetup. Almost the same as that in the DMA-FE scheme except that $\widehat{\mathbb{B}}^*_t$ is *published* in our DMA-ABS, while it is *secret* in the DMA-FE scheme. They are used in our DMA-ABS for the signature generation procedure Sig to meet the *privacy* of signers (for randomization). This is an essential difference between DMA-FE and DMA-ABS.

Here, we remark an important difference in setup between (non-decentralized) ABS and DMA-ABS: While a part of $\widehat{\mathbb{B}}^*_0$, $\boldsymbol{b}^*_{0,1}$, is a master secret in ABS [27], there is no central space $\mathbb{V}_0$ in our DMA-ABS. To obtain unforgeability in our setting, the secret key $\boldsymbol{b}^*_{0,1}$ in ABS is distributed to all $(\boldsymbol{b}^*_{t,\iota})_{t>0;\iota=1,2}$. Therefore, we modify them to $(\widetilde{\boldsymbol{b}}^*_{t,\iota} := \pi \boldsymbol{b}^*_{t,\iota})_{t>0;\iota=1,2}$ with $\pi \xleftarrow{\cup} \mathbb{F}_q$ as a part of public key $\{\widehat{\mathbb{B}}^*_t\}_{t>0}$.

AttrGen. The same as that in the DMA-FE scheme.

Sig. Specific in DMA-ABS. To meet the privacy condition for $\vec{\boldsymbol{s}}^*$, a novel technique is employed to randomly generate a signature from sk_Γ and $\{\widehat{\mathbb{B}}^*_t\}_{(t,x_t)\in\Gamma}$. Since our DMA-FE (and DMA-ABS) lacks the central space $\mathbb{V}_0$, attribute vectors $(1, x_t)$ and $\delta(1, x_t)$ with $\delta \xleftarrow{\cup} \mathbb{F}_q$ are encoded in subspaces $\mathsf{span}\langle \boldsymbol{b}^*_{t,1}, \boldsymbol{b}^*_{t,2}\rangle$ and $\mathsf{span}\langle \boldsymbol{b}^*_{t,3}, \boldsymbol{b}^*_{t,4}\rangle$, for sk_Γ with $\Gamma := \{(t, x_t)\}$. In signature generation, both vectors are re-randomized independently using $(\widetilde{\boldsymbol{b}}^*_{t,\iota}, \boldsymbol{b}^*_{t,2+\iota})_{\iota=1,2}$, in a manner consistent with predicate $\mathbb{S}$.

Ver. The signature verification in our DMA-ABS checks whether a signature (or a specific decryption key) $\vec{\boldsymbol{s}}^*$ works as a decryption key to decrypt a verification text (or a ciphertext) associated with $\mathbb{S}$ and $\mathsf{H}_2(m, \mathbb{S})$.

4.3 Proposed DMA-ABS Scheme

For matrix $X := (\chi_{i,j})_{i,j=1,\ldots,N} \in \mathbb{F}_q^{N\times N}$ and element $\boldsymbol{g} := (G_1, \ldots, G_N)$ in N-dimensional $\mathbb{V}$, $\boldsymbol{g}X$ denotes $(\sum_{i=1}^N G_i\chi_{i,1}, \ldots, \sum_{i=1}^N G_i\chi_{i,N}) = (\sum_{i=1}^N \chi_{i,1}G_i, \ldots,$

$\sum_{i=1}^{N} \chi_{i,N} G_i$) by a natural multiplication of a N-dim. row vector and a $N \times N$ matrix. Thus, it holds that $e(\boldsymbol{g}X, \boldsymbol{h}(X^{-1})^{\mathrm{T}}) = e(\boldsymbol{g}, \boldsymbol{h})$ for any $\boldsymbol{g}, \boldsymbol{h} \in \mathbb{V}$. The proposed scheme is given as:

$\mathsf{GSetup}(1^\lambda)$: $\mathsf{param}_{\mathbb{G}} := (q, \mathbb{G}, \mathbb{G}_T, G, e) \xleftarrow{\mathsf{R}} \mathcal{G}_{\mathsf{bpg}}(1^\lambda)$,
$H_1 : \{0,1\}^* \to \mathbb{G}$; $H_2 : \{0,1\}^* \to \mathbb{F}_q$; return $\mathsf{gparam} := (\mathsf{param}_{\mathbb{G}}, H_1, H_2)$.

Remark : Given gparam, the following values can be computed by anyone and shared by all parties: $G_0 := H_1(0^\lambda) \in \mathbb{G}$, $G_1 := H_1(0^{\lambda-1}, 1) \in \mathbb{G}$, $G_2 := H_1(0^{\lambda-2}, 1, 0) \in \mathbb{G}$, $g_T := e(G_0, G_1)$.

$\mathsf{ASetup}(\mathsf{gparam}, t)$: $\mathsf{param}_{\mathbb{V}_t} := (q, \mathbb{V}_t, \mathbb{G}_T, \mathbb{A}_t, e) := \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, 13, \mathsf{param}_{\mathbb{G}})$,
$X_t \xleftarrow{\mathsf{U}} GL(13, \mathbb{F}_q)$, $(\widetilde{\varphi}_{t,\iota,1}, \widetilde{\varphi}_{t,\iota,2}) \xleftarrow{\mathsf{U}} \mathbb{F}_q^2$ for $\iota = 1, 2$,
$\boldsymbol{b}_{t,\iota} := (0^{\iota-1},\ G_0,\ 0^{13-\iota}) X_t$, $\boldsymbol{b}^*_{t,\iota} := (0^{\iota-1},\ G_1,\ 0^{13-\iota})(X_t^{-1})^{\mathrm{T}}$ for $\iota = 1, \ldots, 13$,

$$\widetilde{\boldsymbol{b}}^*_{t,1} := (\ \overbrace{G_2,\ 0,}^{2}\ \overbrace{0^8,}^{8}\ \overbrace{\widetilde{\varphi}_{t,1,1}G_1, \widetilde{\varphi}_{t,1,2}G_1,}^{2}\ \overbrace{0}^{1}\ )(X_t^{-1})^{\mathrm{T}},$$

$$\widetilde{\boldsymbol{b}}^*_{t,2} := (\ \overbrace{0,\ G_2,}^{2}\ \overbrace{0^8,}^{8}\ \overbrace{\widetilde{\varphi}_{t,2,1}G_1, \widetilde{\varphi}_{t,2,2}G_1,}^{2}\ \overbrace{0}^{1}\ )(X_t^{-1})^{\mathrm{T}},$$

$\mathbb{B}_t := (\boldsymbol{b}_{t,1}, \ldots, \boldsymbol{b}_{t,13})$, $\mathbb{B}^*_t := (\boldsymbol{b}^*_{t,1}, \ldots, \boldsymbol{b}^*_{t,13})$, $\widehat{\mathbb{B}}_t := (\boldsymbol{b}_{t,1}, \ldots, \boldsymbol{b}_{t,6}, \boldsymbol{b}_{t,13})$,
$\widehat{\mathbb{B}}^*_t := (\widetilde{\boldsymbol{b}}^*_{t,1}, \widetilde{\boldsymbol{b}}^*_{t,2}, \boldsymbol{b}^*_{t,3}, \ldots, \boldsymbol{b}^*_{t,6}, \boldsymbol{b}^*_{t,11}, \boldsymbol{b}^*_{t,12})$,
return $\mathsf{ask}_t := X_t$, $\mathsf{apk}_t := (\mathsf{param}_{\mathbb{V}_t}, \widehat{\mathbb{B}}_t, \widehat{\mathbb{B}}^*_t)$.

Remark : Let $\pi \in \mathbb{F}_q$ s.t. $G_2 = \pi G_1$,

$$\text{then } \widetilde{\boldsymbol{b}}^*_{t,1} = (\ \overbrace{\pi, 0}^{2},\ \overbrace{0^8}^{8},\ \overbrace{\widetilde{\varphi}_{t,1,1}, \widetilde{\varphi}_{t,1,2},}^{2}\ \overbrace{0}^{1}\ )_{\mathbb{B}^*_t},$$

$$\widetilde{\boldsymbol{b}}^*_{t,2} = (\ \overbrace{0, \pi}^{2},\ \overbrace{0^8}^{8},\ \overbrace{\widetilde{\varphi}_{t,2,1}, \widetilde{\varphi}_{t,2,2},}^{2}\ \overbrace{0}^{1}\ )_{\mathbb{B}^*_t}.$$

$\mathsf{AttrGen}(\mathsf{gparam},\ t,\ \mathsf{ask}_t,\ \mathsf{gid},\ x_t \in \mathbb{F}_q)$: $G_{\mathsf{gid}} := H_1(\mathsf{gid})$, $(\varphi_{t,1}, \varphi_{t,2}) \xleftarrow{\mathsf{U}} \mathbb{F}_q^2$,

$$\boldsymbol{k}^*_t := (\ \overbrace{G_1,\ x_t G_1,}^{2}\ \overbrace{G_{\mathsf{gid}},\ x_t G_{\mathsf{gid}},}^{2}\ \overbrace{0^6,}^{6}\ \overbrace{\varphi_{t,1}G_1,\ \varphi_{t,2}G_1,}^{2}\ \overbrace{0}^{1}\ )(X_t^{-1})^{\mathrm{T}},$$

return $\mathsf{usk}_{\mathsf{gid},(t,x_t)} := (\mathsf{gid}, (t, x_t), \boldsymbol{k}^*_t)$.

Remark : Let $\delta \in \mathbb{F}_q$ s.t. $G_{\mathsf{gid}} = \delta G_1$,

$$\text{then } \boldsymbol{k}^*_t = (\ \overbrace{(1,\ x_t),}^{2}\ \overbrace{\delta(1,\ x_t),}^{2}\ \overbrace{0^6}^{6},\ \overbrace{\varphi_{t,1},\ \varphi_{t,2},}^{2}\ 0\ )_{\mathbb{B}^*_t}.$$

$\mathsf{Sig}(\mathsf{gparam}, \{\mathsf{apk}_t, \mathsf{usk}_{\mathsf{gid},(t,x_t)} := (\mathsf{gid}, (t, x_t), \boldsymbol{k}^*_t)\},\ m,\ \mathbb{S} := (M, \rho))$:
If $\mathbb{S} := (M, \rho)$ accepts $\Gamma := \{(t, x_t) \in \mathsf{usk}_{\mathsf{gid},(t,x_t)}\}$, then compute I and $\{\alpha_i\}_{i \in I}$ such that $\vec{1} = \sum_{i \in I} \alpha_i M_i$, where M_i is the i-th row of M, and

$$I \subseteq \{i \in \{1,\dots,\ell\} \mid [\rho(i) = (t, v_i) \ \wedge\ (t, x_t) \in \Gamma \ \wedge\ v_i = x_t] \vee [\rho(i) = \neg(t, v_i) \ \wedge\ (t, x_t) \in \Gamma \ \wedge\ v_i \neq x_t]\},$$

$\psi \xleftarrow{\mathsf{U}} \mathbb{F}_q$, $\psi_i := \psi$ if $i \in I$, $\psi_i := 0$ if $i \notin I$ for $i = 1,\dots,\ell$,

for $i = 1,\dots,\ell$, $\zeta_i \xleftarrow{\mathsf{U}} \mathbb{F}_q$, $(\beta_{i,0}), (\beta_{i,1}) \xleftarrow{\mathsf{U}} \{(\beta_1,\dots,\beta_\ell) \mid \sum_{i=1}^{\ell} \beta_i M_i = \vec{0}\}$,

Remark : If $\mathrm{rank}(M) \geq \ell$, the set contains only 0^ℓ, i.e., $\beta_i = 0$ for $i = 1,..,\ell$.

$$\boldsymbol{s}_i^* := \gamma_i \cdot \boldsymbol{k}_t^* + \psi_i(\boldsymbol{b}_{t,3}^* + x_t \boldsymbol{b}_{t,4}^*) + \sum_{\iota=1}^{2} \left(y_{i,0,\iota} \widetilde{\boldsymbol{b}}_{t,\iota}^* + y_{i,1,\iota} \boldsymbol{b}_{t,2+\iota}^* \right) + \zeta_i \left(\boldsymbol{b}_{t,5}^* + H_2(m, \mathbb{S}) \boldsymbol{b}_{t,6}^* \right) + \boldsymbol{r}_i^*,$$

where $\boldsymbol{r}_i^* \xleftarrow{\mathsf{U}} \mathsf{span}\langle \boldsymbol{b}_{t,11}^*, \boldsymbol{b}_{t,12}^* \rangle$, and $\gamma_i, \vec{y}_{i,j} := (y_{i,j,1}, y_{i,j,2})$ for $j = 0, 1$, are defined as

if $i \in I \ \wedge\ \rho(i) = (t, v_i)$, $\gamma_i := \alpha_i$, $\vec{y}_{i,j} := \beta_{i,j}(1, v_i)$,

if $i \in I \ \wedge\ \rho(i) = \neg(t, v_i)$, $\gamma_i := \dfrac{\alpha_i}{v_i - x_t}$, $\vec{y}_{i,j} := \dfrac{\beta_{i,j}}{v_i - y_{i,j}}(1, y_{i,j})$ where $y_{i,j} \xleftarrow{\mathsf{U}} \mathbb{F}_q \setminus \{v_i\}$,

if $i \notin I \ \wedge\ \rho(i) = (t, v_i)$, $\gamma_i := 0$, $\vec{y}_{i,j} := \beta_{i,j}(1, v_i)$,

if $i \notin I \ \wedge\ \rho(i) = \neg(t, v_i)$, $\gamma_i := 0$, $\vec{y}_{i,j} := \dfrac{\beta_{i,j}}{v_i - y_{i,j}}(1, y_{i,j})$ where $y_{i,j} \xleftarrow{\mathsf{U}} \mathbb{F}_q \setminus \{v_i\}$,

return $\vec{\boldsymbol{s}}^* := (\boldsymbol{s}_1^*, \dots, \boldsymbol{s}_\ell^*)$.

$\mathsf{Ver}(\mathsf{gparam}, \{\mathsf{apk}_t\}, m, \mathbb{S} := (M, \rho), \vec{\boldsymbol{s}}^*)$: $\vec{f} \xleftarrow{\mathsf{U}} \mathbb{F}_q^r$, $\vec{s}^{\mathrm{T}} := (s_1, \dots, s_\ell)^{\mathrm{T}} := M \cdot \vec{f}^{\mathrm{T}}$,

$s_0 := \vec{1} \cdot \vec{f}^{\mathrm{T}}$, $\vec{f}' \xleftarrow{\mathsf{U}} \mathbb{F}_q^r$ s.t. $\vec{1} \cdot \vec{f}'^{\mathrm{T}} = 0$, $\vec{s}'^{\mathrm{T}} := (s'_1, \dots, s'_\ell)^{\mathrm{T}} := M \cdot \vec{f}'^{\mathrm{T}}$,

for $i = 1,\dots,\ell$, $\theta_i, \theta'_i, \theta''_i, \eta_i \xleftarrow{\mathsf{U}} \mathbb{F}_q$,

if $\rho(i) = (t, v_i)$,

$$\boldsymbol{c}_i := (\overbrace{s_i + \theta_i v_i,\ -\theta_i}^{2}, \overbrace{s'_i + \theta'_i v_i,\ -\theta'_i}^{2}, \overbrace{\theta''_i(H_2(m, \mathbb{S}), -1)}^{2}, \overbrace{0^6}^{6}, \overbrace{\eta_i}^{1})_{\mathbb{B}_t},$$

if $\rho(i) = \neg(t, v_i)$,

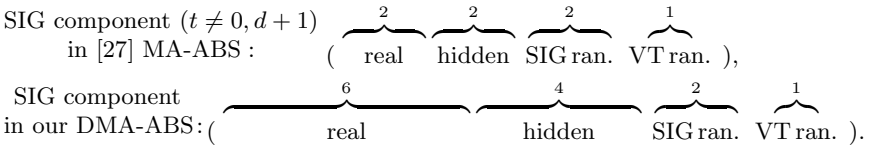
$$\boldsymbol{c}_i := (\overbrace{s_i(v_i,\ -1)}^{2}, \overbrace{s'_i(v_i,\ -1)}^{2}, \overbrace{\theta''_i(H_2(m, \mathbb{S}), -1)}^{2}, \overbrace{0^6}^{6}, \overbrace{\eta_i}^{1})_{\mathbb{B}_t},$$

$c_{d+1} := g_T^{s_0}$, return 1 if $\prod_{i=1}^{\ell} e(\boldsymbol{c}_i, \boldsymbol{s}_i^*) = c_{d+1}$, return 0 otherwise.

[Correctness] If $\mathbb{S} := (M, \rho)$ accepts $\Gamma := \{(t, x_t) \in \mathsf{usk}_{\mathsf{gid},(t,x_t)}\}$, $\prod_{i=1}^{\ell} e(\boldsymbol{c}_i, \boldsymbol{s}_i^*)$
$= \prod_{i \in I} \left(e(\boldsymbol{c}_i, \boldsymbol{k}_t^*)^{\gamma_i} e(\boldsymbol{c}_i, \boldsymbol{b}_3^* + x_{t,\iota} \boldsymbol{b}_4^*)^{\psi} \right) \cdot \prod_{i=1}^{\ell} \prod_{\iota=1}^{2} e(\boldsymbol{c}_i, \widetilde{\boldsymbol{b}}_\iota^*)^{y_{i,0,\iota}} e(\boldsymbol{c}_i, \boldsymbol{b}_{2+\iota}^*)^{y_{i,1,\iota}}$
$= \prod_{i \in I} g_T^{\alpha_i (s_i + (\delta+\psi) s'_i)} \cdot \prod_{i=1}^{\ell} g_T^{\pi \beta_{i,0} s_i + \beta_{i,1} s'_i} = g_T^{\sum_{i \in I} \alpha_i (s_i + (\delta+\psi) s'_i)} \cdot g_T^{\sum_{i=1}^{\ell} (\pi \beta_{i,0} s_i + \beta_{i,1} s'_i)}$
$= g_T^{s_0}$, since $\sum_{i \in I} \alpha_i s_i = s_0$ and $\sum_{i \in I} \alpha_i s'_i = \sum_{i=1}^{\ell} \beta_{i,0} s_i = \sum_{i=1}^{\ell} \beta_{i,1} s'_i = 0$.

Comparison with the MA-ABS Scheme in [27]. Okamoto-Takashima [27] gave a fully secure (non-decentralized) MA-ABS scheme on the DPVS framework. In their scheme, a signature (SIG) associated with a policy of size ℓ consists of $(\ell + 2)$ components, $(s_0^*, \ldots, s_{\ell+1}^*)$, which are categorized into three roles. The first one, $s_0^* \in \mathbb{V}_0$ (for $t = 0$), is for embedding/recovering a secret, the second, $(s_1^*, \ldots, s_\ell^*)$, for secret shares on the policy (access structure), and the last, $s_{\ell+1}^* \in \mathbb{V}_{d+1}$ (for $t = d + 1$), is for embedding/verifying the hashed value, $H_2(m, \mathbb{S})$. The secret share components, $(s_1^*, \ldots, s_\ell^*)$, are 7-dimensional $(7 = 2 + 2 + 2 + 1)$, where the first 2-dimensional part is the real-encoding part (real part, for short) for shared secrets, the second the hidden part for semi-functional signatures, the third the signature randomness part, and the last is the verification text (VT) randomness part.

In the DMA setting, we cannot use special (central) spaces, $\mathbb{V}_0$ and $\mathbb{V}_{d+1}$. Instead, we should distribute the roles of these spaces into the secret share components, $(s_1^*, \ldots, s_\ell^*)$. As a result, these components become 13-dimensional $(13 = 6 + 4 + 2 + 1)$, where the real part (hidden part, resp.) is expanded to 6-dimensions (4-dimensions, resp.) (see the figure below). The 6-dimensional real part consists of 2 dimensions to distribute the role of $\mathbb{V}_0$, 2 dimensions for secret shares, and 2 dimensions to distribute the role of $\mathbb{V}_{d+1}$. We also use additional 2 dimensions in the hidden part to execute the *swapping* technique in the security proof.



4.4 Security of the Proposed DMA-ABS

The (standard) DLIN assumption is given in the full version [26].

Theorem 1. *The proposed DMA-ABS scheme is perfectly private.*

Theorem 2. *The proposed DMA-ABS scheme is unforgeable (adaptive-predicate unforgeable) under the DLIN assumption in the random oracle model.*

The proofs of Theorems 1 and 2 are given in the full version of this paper [26].

4.5 Performance

In this section, we compare the efficiency and security of the proposed DMA-ABS scheme with the existing MA-ABS schemes in the standard model (instantiation 2 in [24] and MA-ABS in [27]) as well as the ABS scheme in the generic group model (instantiation 3 in [24]) as a benchmark. Since all of these schemes can be implemented over a *prime order* pairing group, the size of a group element can be around the size of $\mathbb{F}_q$ (e.g., 256 bits). In Table 1, ℓ and r represent the size of the underlying access structure matrix M for a predicate, i.e., $M \in \mathbb{F}_q^{\ell \times r}$.

Table 1. Comparison with the Existing MA-ABS Schemes

	MPR10 [24] Instantiation 3	MPR10 [24] Instantiation 2	OT11 [27]	Proposed
Signature size (# of group elts)	$\ell + r + 2$	$36\ell + 2r + 9\lambda + 12$	$7\ell + 11$	13ℓ
Decentralized	×	×	×	✓
Model	generic group model	standard model	standard model	random oracle model
Security	full	full	full	full
Authority Corruption Type	strong	strong	weak	weak
Assumptions	CR hash	DLIN	DLIN and CR hash	DLIN
Predicates	monotone	monotone	non-monotone	non-monotone
Sig. size example 1 ($\ell = 10, r = 5, \lambda = 128$)	17	1534	81	130
Sig. size example 2 ($\ell = 100, r = 50, \lambda = 128$)	152	4864	711	1300

For example, some predicate with 4 AND and 5 OR gates as well as 10 variables may be expressed by a 10×5 matrix, and a predicate with 49 AND and 50 OR gates as well as 100 variables may be expressed by a 100×50 matrix (see the appendix of [20]). λ is the security parameter (e.g., 128).

5 Concluding Remarks

We presented the first DMA-ABS scheme, in which no central authority and no trusted setup are required. An adaptively secure DMA-FE scheme with no trusted setup was also presented.

One of the most important remaining problems in this paper is to construct a DMA-ABS (and DMA-FE) scheme in the standard model (without random oracles). It would be also important to realize a DMA-ABS (and DMA-FE) scheme with no trusted setup in a stronger authority corruption model (like that in [24]), and to introduce a revocation mechanism in a DMA-ABS scheme.

References

1. Beimel, A.: Secure schemes for secret sharing and key distribution. PhD Thesis, Israel Institute of Technology, Technion, Haifa (1996)
2. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi (ed.) [15], pp. 108–125

3. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: P-signatures and Noninteractive Anonymous Credentials. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 356–374. Springer, Heidelberg (2008)
4. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boyen, X.: Mesh Signatures. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 210–227. Springer, Heidelberg (2007)
6. Camenisch, J., Groß, T.: Efficient attributes for anonymous credentials. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM Conference on Computer and Communications Security. pp. 345–356. ACM (2008)
7. Camenisch, J., Lysyanskaya, A.: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)
8. Camenisch, J., Lysyanskaya, A.: Signature Schemes and Anonymous Credentials from Bilinear Maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
9. Chase, M.: Multi-authority Attribute Based Encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
10. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. Commun. ACM 28(10), 1030–1044 (1985)
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)
12. Estibals, N.: Compact Hardware for Computing the Tate Pairing over 128-Bit-Security Supersingular Curves. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 397–416. Springer, Heidelberg (2010)
13. FIPS PUB 180-1, 180-2: Secure hash standard. NIST (1995, 2002)
14. Guo, S., Zeng, Y.: Attribute-based signature scheme. In: ISA, pp. 509–511. IEEE (2008)
15. Halevi, S. (ed.): CRYPTO 2009. LNCS, vol. 5677. Springer, Heidelberg (2009)
16. ISO/IEC 15946-5: Information technology - Security techniques - Cryptographic techniques based on elliptic curves - Part 5: Elliptic curve generation. ISO/IEC (2009)
17. Khader, D.: Attribute based group signature with revocation. IACR Cryptology ePrint Archive 2007, 241 (2007)
18. Khader, D.: Attribute based group signatures. IACR Cryptology ePrint Archive 2007, 159 (2007)
19. Lewko, A.: Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 318–335. Springer, Heidelberg (2012)
20. Lewko, A., Waters, B.: Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011)
21. Li, J., Au, M.H., Susilo, W., Xie, D., Ren, K.: Attribute-based signature and its applications. In: Feng, D., Basin, D.A., Liu, P. (eds.) ASIACCS, pp. 60–69. ACM (2010)
22. Li, J., Kim, K.: Attribute-based ring signatures. IACR Cryptology ePrint Archive 2008, 394 (2008)

23. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. IACR Cryptology ePrint Archive 2008, 328 (2008)
24. 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)
25. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010), full version is available at `http://eprint.iacr.org/2010/563`
26. Okamoto, T., Takashima, K.: Decentralized attribute-based signatures. IACR Cryptology ePrint Archive 2011, 701 (2011), full version of this paper, `http://eprint.iacr.org/2011/701`
27. 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), full version is available at `http://eprint.iacr.org/2011/700`
28. 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)
29. Shacham, H., Waters, B.: Efficient Ring Signatures Without Random Oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 166–180. Springer, Heidelberg (2007)
30. 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)
31. Waters, B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In: Halevi (ed.) [15], pp. 619–636
32. Yang, P., Cao, Z., Dong, X.: Fuzzy identity based signature. IACR Cryptology ePrint Archive 2008, 2 (2008)

A Proposed DMA-FE

We define function $\widetilde{\rho} : \{1, \ldots, \ell\} \to \{1, \ldots, d\}$ by $\widetilde{\rho}(i) := t$ if $\rho(i) = (t, \vec{v})$ or $\rho(i) = \neg(t, \vec{v})$, where ρ is given in access structure $\mathbb{S} := (M, \rho)$. In the proposed scheme, we assume that $\widetilde{\rho}$ is injective for $\mathbb{S} := (M, \rho)$ with ciphertext $\boldsymbol{c} = \boldsymbol{c}_{\mathbb{S}}$. We will show how to relax the restriction in the full version [26]. In the description of the scheme, we assume that input vector $\vec{x}_t := (x_{t,1}, \ldots, x_{t,n_t})$ is normalized such that $x_{t,1} := 1$. (If $\vec{x}_t$ is not normalized, change it to a normalized one by $(1/x_{t,1}) \cdot \vec{x}_t$ assuming that $x_{t,1}$ is non-zero). In addition, we assume that input vector $\vec{v}_i := (v_{i,1}, \ldots, v_{i,n_t})$ satisfies that $v_{i,n_t} \neq 0$. For matrix $X := (\chi_{i,j})_{i,j=1,\ldots,N} \in \mathbb{F}_q^{N \times N}$ and element $\boldsymbol{g} := (G_1, \ldots, G_N)$ in N-dimensional $\mathbb{V}$, for notation $\boldsymbol{g}X$, refer to Section 4.3.

$\mathsf{GSetup}(1^\lambda)$: $\mathsf{param}_{\mathbb{G}} := (q, \mathbb{G}, \mathbb{G}_T, G, e) \xleftarrow{\mathsf{R}} \mathcal{G}_{\mathsf{bpg}}(1^\lambda)$, $H : \{0,1\}^* \to \mathbb{G}$;
return $\mathsf{gparam} := (\mathsf{param}_{\mathbb{G}}, H)$.

Remark : Given gparam, the following values can be computed by

anyone and shared by all parties:

$G_0 := H_1(0^\lambda) \in \mathbb{G},\ \ G_1 := H_1(0^{\lambda-1}, 1) \in \mathbb{G},\ \ g_T := e(G_0, G_1),$

$\mathsf{ASetup}(\mathsf{gparam}, t, n_t) : \mathsf{param}_{\mathbb{V}_t} := (q, \mathbb{V}_t, \mathbb{G}_T, \mathbb{A}_t, e) := \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, 5n_t+1, \mathsf{param}_{\mathbb{G}}),$

$X_t \xleftarrow{\mathsf{U}} GL(5n_t+1, \mathbb{F}_q),\ \boldsymbol{b}_{t,i} := (0^{i-1}, G_0, 0^{5n_t+1-i})X_t$ for $i = 1, \ldots, 5n_t+1,$

$\widehat{\mathbb{B}}_t := (\boldsymbol{b}_{t,1}, \ldots, \boldsymbol{b}_{t,2n_t}, \boldsymbol{b}_{t,5n_t+1}),\ \ \mathsf{ask}_t := X_t,\ \ \mathsf{apk}_t := (\mathsf{param}_{\mathbb{V}_t}, \widehat{\mathbb{B}}_t),$

return $(\mathsf{ask}_t,\ \mathsf{apk}_t)$.

$\mathsf{AttrGen}(\mathsf{gparam},\ t,\ \mathsf{ask}_t,\ \mathsf{gid},\ \vec{x}_t := (x_{t,1}, \ldots, x_{t,n_t}) \in \mathbb{F}_q^{n_t} \setminus \{\vec{0}\} \text{ s.t. } x_{t,1} := 1) :$

$G_{\mathsf{gid}} := H(\mathsf{gid}) \in \mathbb{G},\ \vec{\varphi}_t := (\varphi_{t,1}, \ldots, \varphi_{t,n_t}) \xleftarrow{\mathsf{U}} \mathbb{F}_q^{n_t},$

$$\boldsymbol{k}_t^* := (\ \overbrace{x_{t,1}G_1, .., x_{t,n_t}G_1,}^{n_t}\ \overbrace{x_{t,1}G_{\mathsf{gid}}, .., x_{t,n_t}G_{\mathsf{gid}},}^{n_t}\ \overbrace{0^{2n_t},}^{2n_t}\ \overbrace{\varphi_{t,1}G_1, .., \varphi_{t,n_t}G_1,}^{n_t}\ \overbrace{0}^{1}\ )(X_t^{-1})^{\mathrm{T}},$$

return $\mathsf{usk}_{\mathsf{gid},(t,\vec{x}_t)} := (\mathsf{gid}, (t, \vec{x}_t), \boldsymbol{k}_t^*).$

Remark: Let $\boldsymbol{b}_{t,i}^* := (0^{i-1},\ G_1,\ 0^{5n_t+1-i})(X_t^{-1})^{\mathrm{T}},$
$\mathbb{B}_t^* := (\boldsymbol{b}_{t,1}^*, \ldots, \boldsymbol{b}_{t,5n_t+1}^*)$ and $\delta \in \mathbb{F}_q$ s.t. $G_{\mathsf{gid}} = \delta G_1$. Then $\boldsymbol{k}_t^*$ is

represented as $\boldsymbol{k}_t^* = (\ \overbrace{\vec{x}_t}^{n_t},\ \overbrace{\delta\vec{x}_t}^{n_t},\ \overbrace{0^{2n_t}}^{2n_t},\ \overbrace{\vec{\varphi}_t}^{n_t},\ 0\ )_{\mathbb{B}_t^*}.$

$\mathsf{Enc}(\mathsf{gparam}, \{\mathsf{apk}_t\},\ m,\ \mathbb{S} := (M, \rho)) :$

$\vec{f} \xleftarrow{\mathsf{U}} \mathbb{F}_q^r,\ \vec{s}^{\mathrm{T}} := (s_1, \ldots, s_\ell)^{\mathrm{T}} := M \cdot \vec{f}^{\mathrm{T}},\ s_0 := \vec{1} \cdot \vec{f}^{\mathrm{T}}, \vec{f}' \xleftarrow{\mathsf{R}} \mathbb{F}_q^r \text{ s.t. } \vec{1} \cdot \vec{f}'^{\mathrm{T}} = 0,$

$\vec{s}'^{\mathrm{T}} := (s_1', \ldots, s_\ell')^{\mathrm{T}} := M \cdot \vec{f}'^{\mathrm{T}},\ \eta_i, \theta_i, \theta_i' \xleftarrow{\mathsf{U}} \mathbb{F}_q\ (i = 1, .., \ell),$

for $i = 1, \ldots, \ell,$

if $\rho(i) = (t, \vec{v}_i := (v_{i,1}, \ldots, v_{i,n_t}) \in \mathbb{F}_q^{n_t} \setminus \{\vec{0}\} \text{ such that } v_{i,n_t} \neq 0),$

$$\boldsymbol{c}_i := (\ \overbrace{s_i\vec{e}_{t,1} + \theta_i\vec{v}_i,}^{n_t}\ \overbrace{s_i'\vec{e}_{t,1} + \theta_i'\vec{v}_i,}^{n_t}\ \overbrace{0^{2n_t},}^{2n_t}\ \overbrace{0^{n_t},}^{n_t}\ \overbrace{\eta_i}^{1}\ )_{\mathbb{B}_t},$$

if $\rho(i) = \neg(t, \vec{v}_i),\ \ \boldsymbol{c}_i := (\ \overbrace{s_i\vec{v}_i,}^{n_t}\ \overbrace{s_i'\vec{v}_i,}^{n_t}\ \overbrace{0^{2n_t},}^{2n_t}\ \overbrace{0^{n_t},}^{n_t}\ \overbrace{\eta_i}^{1}\ )_{\mathbb{B}_t},$

$c_{d+1} := g_T^{s_0} m,\ \ \mathsf{ct}_{\mathbb{S}} := (\mathbb{S}, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_\ell, c_{d+1}),\ \ \text{return } \mathsf{ct}_{\mathbb{S}}.$

$\mathsf{Dec}(\mathsf{gparam}, \{\mathsf{apk}_t, \mathsf{usk}_{\mathsf{gid},(t,\vec{x}_t)} := (\mathsf{gid}, (t, \vec{x}_t), \boldsymbol{k}_t^*)\}, \mathsf{ct}_{\mathbb{S}} := (\mathbb{S}, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_\ell, c_{d+1})) :$

If $\mathbb{S} := (M, \rho)$ accepts $\Gamma := \{(t, \vec{x}_t) \in \mathsf{usk}_{\mathsf{gid},(t,\vec{x}_t)}\}$, then compute I and $\{\alpha_i\}_{i \in I}$

such that $\vec{1} = \sum_{i \in I} \alpha_i M_i,$ where M_i is the i-th row of $M,$ and

$I \subseteq \{i \in \{1, \ldots, \ell\} |\ [\rho(i) = (t, \vec{v}_i) \ \wedge\ (t, \vec{x}_t) \in \Gamma \ \wedge\ \vec{v}_i \cdot \vec{x}_t = 0]$

$\qquad\qquad \vee\ \ [\rho(i) = \neg(t, \vec{v}_i) \ \wedge\ (t, \vec{x}_t) \in \Gamma \ \wedge\ \vec{v}_i \cdot \vec{x}_t \neq 0]\ \},$

$$K := \prod_{i \in I\ \wedge\ \rho(i) = (t, \vec{v}_i)} e(\boldsymbol{c}_i, \boldsymbol{k}_t^*)^{\alpha_i} \cdot \prod_{i \in I\ \wedge\ \rho(i) = \neg(t, \vec{v}_i)} e(\boldsymbol{c}_i, \boldsymbol{k}_t^*)^{\alpha_i / (\vec{v}_i \cdot \vec{x}_t)},$$

return $m' := c_{d+1}/K.$

On the Semantic Security of Functional Encryption Schemes

Manuel Barbosa[1] and Pooya Farshim[2]

[1] HASLab – INESC TEC and Universidade do Minho, Portugal
[2] Fachbereich Informatik, Technische Universität Darmstadt, Germany
mbb@di.uminho.pt, farshim@cased.de

Abstract. Functional encryption (FE) is a powerful cryptographic primitive that generalizes many asymmetric encryption systems proposed in recent years. Syntax and security definitions for FE were proposed by Boneh, Sahai, and Waters (BSW) (TCC 2011) and independently by O'Neill (ePrint 2010/556). In this paper we revisit these definitions, identify several shortcomings in them, and propose a new definitional approach that overcomes these limitations. Our definitions display good compositionality properties and allow us to obtain new feasibility and impossibility results for adaptive token-extraction attack scenarios that shed further light on the potential reach of general FE for practical applications.

Keywords. Functional encryption, Semantic security, Adaptive token extraction, Inner-product encryption, SIS problem.

1 Introduction

Functional encryption (FE) is a public-key primitive that generalizes many encryption systems, including public-key encryption (PKE), identity-based encryption (IBE), searchable encryption, attribute-based encryption (ABE), and all other variants of predicate encryption systems [5]. In such a system, each decryption key TK_f (called a *token*) is associated with a function f (which may be viewed as a circuit). When a token holder runs the decryption algorithm on a ciphertext encrypting a message m, it recovers the image $f(\mathsf{m})$. A trusted authority holding a master secret key is responsible for issuing tokens. This allows the TA to control which users can recover which images from encrypted data. Realizing such a powerful primitive for complex functionalities could revolutionize information security applications in a way that is comparable to the notable case of fully homomorphic encryption. Interestingly, very recent developments in this area indicate that this may indeed be within our reach. For example, a concrete realization of functional encryption for arbitrary functionalities has been recently proposed in [9], as well as functional encryption for regular languages in [15].

The intuitive security requirement for a functional encryption scheme is that no information should leak from a ciphertext bar that which can be recovered via legitimately obtained decryption tokens. As for other encryption primitives, there are various ways in which this intuition can be formalized. In the case of PKE, for example, the two standard formalizations are semantic security and ciphertext indistinguishability, which were shown to be equivalent in the seminal work of Goldwasser and Micali [8].

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 143–161, 2013.

Somewhat surprisingly, in independent works, Boneh, Sahai, and Waters [5] (BSW) and O'Neill [14] have shown that this is *not* the case for FE schemes supporting complex functionalities. Indeed, both works demonstrated limitations in the indistinguishability-based notion for functional encryption and proposed strictly stronger semantic security notions to overcome these problems. This highlights the importance of converging to a definition of semantic security for FE that can be adopted as a de facto standard by the community. However, the definitional approaches adopted in both works are significantly different and the relation between the two is not well understood. In particular, it is not clear whether there are fundamental differences between the two so as to determine which one of them should be favored in detriment of the other. The goal of this paper is to change this state of affairs. We analyze the positive and negative aspects of the definitions by BSW and O'Neill and find that both approaches have strengths that should be preserved, and yet they also have weaknesses that should be reconsidered. We propose a new balanced set of definitions incorporating these results.

ANALYSIS OF PREVIOUS DEFINITIONS. Boneh et al. [5] provide an elegant generalization of the syntax of FE schemes. The authors propose a natural indistinguishability-based security definition, but then present a counterexample showing that this notion of security is generally inadequate: a scheme that is intuitively insecure, but can be proven IND-CPA-secure. A notion of semantic security using black-box simulators is then proposed to address this problem. The paper concludes with a series of feasibility results. Most notably, BSW show that semantically secure schemes do not exist even for simple functionalities such as IBE. This result hinges on the adversary's capability to perform adaptive token-extraction queries. Nevertheless, in this work we show that the BSW definition is too *weak* in the sense that it also fails to exclude some intuitively insecure schemes. The problem is that the ideal-world simulator controls the generation of the global parameters for the FE scheme. We show that this renders the simulator unreasonably more powerful than the real-world adversary, as it can retain trapdoor information that permits recovering information from images $f(\mathsf{m})$ that is hidden in the real world.

Independently, O'Neill [14] proposed an alternative definitional approach to FE schemes for general functionalities. The author presents alternative syntax, correctness, and indistinguishability-based security notions that are conceptually close to those in [5], but proposes a significantly different semantic security definition. The paper then discusses the feasibility of achieving semantic security, by first presenting a separation to the indistinguishability notion, and then introducing a simple property for supported functionalities, called *preimage samplability*, under which the two notions are equivalent for non-adaptive token-extraction attacks. The fact that functionalities such as IBE and inner-product encryption [10] are shown to be preimage samplable provide positive results for semantically secure FE schemes for such functionalities.

The semantic security model proposed by O'Neill does not suffer from the same problem we identified for the BSW definition. Indeed, the ideal-world simulator in O'Neill's definition must work with honestly sampled global parameters. Nevertheless, we present other counterexamples for which the intuitive notion of security is not at all clear, but which can be proven secure under O'Neill's model. The crux of the matter here is that information is leaked via tokens, rather than by the ciphertext, which raises the question of whether such a scheme should be rejected by a semantic security

definition. However, one can also argue that a security definition for FE should reject schemes that fail to preserve the security properties of the supported functionalities. Finally, as acknowledged in [14], O'Neill's notion of semantic security does not suitably deal with adaptive token-extraction attacks.

RECENT WORK. In independent work, Bellare and O'Neill [4] proposed syntax and security definitions for FE that go in the same direction as those proposed here. Their notions of correctness and SS3 security are similar in spirit to ours, and their resamplability notion is akin to our notion of restricted preimage samplability. Gorbunov, Vaikuntanathan, and Wee [9] have also presented a new semantic security model. Similarly to our definition, their simulator does not control the generation of the global parameters. However, the simulators considered there are black-box and follow a specific simulation structure. We leave a detailed comparison to future work.

MAIN CONTRIBUTIONS. We now detail our main contributions.

Syntax. We start in Section 2 by tailoring the syntax and correctness definitions of functional encryption so as to capture the standard definitions for primitives such as IBE, ABE, PE, etc., as particular cases. This was not strictly the case with previous approaches. In particular, we identify a notion of *full correctness*, which maps to the notions adopted in [14,5], and imposes that the decryption operation explicitly returns a failure symbol when the functionality is undefined for a particular input value.

Indistinguishability. We modify the notion of *intentional leakage* [5] to the slightly different concept of *potential leakage* in Section 3. This allows us to dissociate syntactic aspects (e.g., we do not need to include a special empty token in the syntax of the primitive) from the security aspects of an FE scheme. Potential leakage captures the general restrictions that must be in place to ensure that various security models exclude attacks on functional encryption schemes based on information that the scheme is not designed to conceal, e.g., the length of messages or the identity of the receivers. Through this notion we are able to define indistinguishability-based security as a natural generalization of the equivalent notions for standard primitives, and automatically get feasibility results that do not require transforming the original schemes.

Semantic security. Having identified a number of limitations of the semantic security models proposed by BSW and O'Neill (Section 4), in Section 5 we propose a notion of semantic security that incorporates features from the definitions by BSW, and also by O'Neill. Again, our goal is to faithfully generalize the definitions of semantic security for primitives like PKE [7] and IBE [1]. We observe that full adaptive token extraction models are not typically considered in such schemes, and so we propose a *restricted adaptive* token-extraction attack model. The restriction we impose intuitively prevents an attacker from obtaining decryption tokens that would allow it to trivially corrupt an encrypted ciphertext a posteriori, in the style of non-committing encryption [13]. Put another way, our semantic security definition permits specifying the message distributions from which encrypted messages may be drawn, along with matching restrictions on the tokens that can be issued by the TA a posteriori, in order to provide FE security guarantees in a more flexible usage scenario. Using this strategy we circumvent impossibility results for unrestricted token extractions [5]. Finally, we show that our semantic security definition displays a desirable composition property: security against

single-message attacks implies security against multi-message attacks, even under restricted adaptive token-extraction attacks, thereby allowing us to present our results in the simpler single-message scenario.

Setup security. Our definition of semantic security preserves the resilience of O'Neill's definition in rejecting schemes that leak information to the adversary via the ciphertext. However, like all previous definitions, it does not provide any safeguards against leakage via decryption tokens or the master secret key. We therefore go on to introduce a new notion of *setup security* which enforces that tokens (or more strongly the setup procedure of the system) do not release any privileged information that might enable token holders or the trusted authority to obtain information which would otherwise be hidden by images values (Section 6). We show that setup security excludes all intuitively insecure schemes that we consider in the paper, while being inclusive enough to enable positive results for existing FE schemes. More precisely, we show that functionalities admitting a *conditional preimage sampling* procedure have an intrinsically secure setup procedure. We show PKE and IBE schemes, and more generally FE schemes supporting *all-or-nothing* functionalities are conditionally preimage samplable.

Adaptive equivalence. In Section 7 we present some positive feasibility results for our proposed notion of semantic security. There we extend O'Neill's results for non-adaptive token-extraction attacks and propose a variant of preimage samplability (PS) that enables us to obtain an equivalence between IND-CPA-secure and semantically secure functional encryption under *restricted* adaptive token extraction. Moreover, our requirement is *weaker* than that of O'Neill if we are only interested in the non-adaptive token extraction scenario. Finally, we show that conditional preimage samplability (as defined to establish setup security) also implies our stronger notion of preimage samplability. We immediately get that indistinguishability-based security is equivalent to semantic security under restricted adaptive token-extraction attacks for all-or-nothing functionalities. This gives a wide range of positive results for (multi-message) semantically secure functional encryption that extends previous known results.

Inner products. We conclude the paper in Section 8 by presenting negative results for inner-product encryption (IPE). These results bring a twist to our extension of O'Neill's work: it is *not* the case that all the equivalences between semantic security and indistinguishability established by O'Neill for non-adaptive token extractions carry over to our restricted adaptive scenario. Concretely, we show that although inner-product encryption is proven by O'Neill to satisfy the preimage sampling property [14], this functionality is provably *not* preimage samplable under the more restrictive PS notion that we introduce: for certain parameterizations of the inner-product functionality, a successful preimage sampler can be used to break the Small Integer Solution (SIS) problem. This leaves open the question of proving the semantic security of existing inner-product encryption schemes under restricted adaptive token-extraction attacks.

2 Functional Encryption Syntax and Correctness

NOTATION. We start by introducing notation. We denote assigning y to x by $x \leftarrow y$ and use $x \leftarrow\!\!\$\ X$ for sampling x from set X uniformly at random. If $\mathcal{A}$ is a probabilistic algorithm, $y \leftarrow\!\!\$\ \mathcal{A}(x_1, \ldots, x_n)$ denotes running $\mathcal{A}$ on $x_1, \ldots, x_n$ with random coins

chosen uniformly at random, assigning the result to y. We use ":" for appending to a list. We denote by $[X]$ the support of random variable X. We say $\nu(\lambda)$ is negligible if $|\nu(\lambda)| \in \lambda^{-\omega(1)}$. We use bold font to denote a vector, and abuse notation when applying a function to each element of a vector, writing $f(\mathbf{m})$. We use $[X]_i$ for the ith component of X, and $[X]_i^j$ for the ith to jth components. We denote the ℓ_2 norm of $\boldsymbol{x}$ by $\|\boldsymbol{x}\|_2$.

SYNTAX. We now define the syntax for a functional encryption (FE) scheme, where the function space may, in general, *depend* on the public parameters of the system; see the discussion below. Such a scheme is specified by four PPT algorithms as follows.

1. $\mathsf{Setup}(1^\lambda)$: This is the setup algorithm. On input a security parameter 1^λ, it outputs a master secret key Msk and a master public key Mpk. Implicitly included in Mpk are a function/circuit space description FunSp and a message space MsgSp. The function space FunSp consists of circuit descriptions $f : \mathsf{MsgSp} \to \mathsf{MsgSp} \cup \{\perp\}$.
2. $\mathsf{TKGen}(f, \mathsf{Msk})$: This is the token-generation algorithm. On input a function f and a master secret key Msk, it outputs a token TK for f.
3. $\mathsf{Enc}(\mathsf{m}, \mathsf{Mpk})$: This is the encryption algorithm. On input a message m and the master public key Mpk, it outputs a ciphertext c.
4. $\mathsf{Dec}(\mathsf{c}, \mathsf{TK})$: This is the deterministic decryption algorithm. On input a ciphertext c and a token TK, it outputs a message $\mathsf{m} \in \mathsf{MsgSp}$ or the special failure symbol $\perp$.

CORRECTNESS. The special symbol $\perp$ in the co-domain of functions accounts for functions that may be undefined on parts of their domain, or for which we do not expect the cryptosystem to behave correctly. We call an FE scheme *correct* if, for all $\lambda \in \mathbb{N}$, all $(\mathsf{Mpk}, \mathsf{Msk}) \in [\mathsf{Setup}(1^\lambda)]$, all $\mathsf{m} \in \mathsf{MsgSp}(\mathsf{Mpk})$, all $\mathsf{c} \in [\mathsf{Enc}(\mathsf{m}, \mathsf{Mpk})]$, all $f \in \mathsf{FunSp}(\mathsf{Mpk})$, and all $\mathsf{TK} \in [\mathsf{TKGen}(f, \mathsf{Msk})]$, we have that $f(\mathsf{m}) \neq \perp \implies \mathsf{Dec}(\mathsf{c}, \mathsf{TK}) = f(\mathsf{m})$. We call an FE scheme *fully correct* when the $f(\mathsf{m}) \neq \perp$ restriction is removed, i.e., when the decryption algorithm must return $\perp$ whenever $f(\mathsf{m}) = \perp$.

In the full version [2] we show that a number of standard cryptographic primitives can be seen as special cases of the FE syntax and correctness conditions defined above.

COMPARISON WITH THE PREVIOUS DEFINITIONS. There are two differences to the definition in [14]. First, O'Neill stipulates that the function space is indexed by the security parameter, yet it is fixed and independent of the setup algorithm. However, for a number of primitives such as inner-product encryption and attribute-based encryption, the function space may depend on the parameters generated by the setup algorithm. Second, we do not require correctness to hold when the function evaluates to $\perp$. As we will see, this weaker correctness notion allows us to see standard PKE, IBE, and other schemes as particular cases of functional encryption. Strictly speaking, this was not possible with the definition in [14]. A correct FE scheme according to [14] can be written as a fully correct scheme in our syntax, and vice versa.

One presentational difference between the definition in [5] and that of ours is that we treat functions explicitly whereas BSW define a general functionality $F(K, \cdot)$ indexed by keys K. This difference is inconsequential as the description of a function can be interpreted as a key. In both definitions various spaces can depend on the public parameters. Furthermore, we do not rely on a special empty key to model leakage of side information such as plaintext length. We will deal with this issue when defining

the security of an FE scheme later on. Finally, our definition of correctness differs from BSW in the same way as it differs from that of O'Neill.

3 Indistinguishability

We define an indistinguishability-based notion of security tailored to capture different gradings of security for the same functionality. The game is parameterized by a PPT relation R that defines the admissible set of challenge queries. This generalizes the restriction of choosing challenge messages with the same length in PKE. By requiring $\mathsf{R}(\mathsf{m}_0, \mathsf{m}_1)$ to hold in the challenge query, one acknowledges that challenge queries that violate this restriction *may* lead to a (trivial) break. This decouples security concerns from the correctness of the scheme. We refer to R as the *potential leakage relation*.

Definition 1 (IND-CPA **Security).** *Let game* $\mathsf{IND\text{-}CPA}_{\mathsf{FE},\mathsf{R},\mathcal{A}}$ *be as defined in Figure 1. The* IND-CPA *security of an FE scheme relative to potential leakage relation* R*, requires the advantage of any adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *to be negligible, when this is defined as*

$$\mathbf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\mathsf{IND\text{-}CPA}_{\mathsf{FE},\mathsf{R},\mathcal{A}}(\lambda) \Rightarrow \mathsf{T}\right] - 1.$$

Game $\mathsf{IND\text{-}CPA}_{\mathsf{FE},\mathsf{R},\mathcal{A}}(\lambda)$:	**oracle** $\mathbf{LR}(\mathsf{m}_0, \mathsf{m}_1)$:	**oracle** $\mathbf{Token}(f)$:
$b \leftarrow\!\!\$\ \{0,1\}$; $\mathsf{TKList} \leftarrow [\,]$	$\mathsf{c} \leftarrow\!\!\$\ \mathsf{Enc}(\mathsf{m}_b, \mathsf{Mpk})$	$\mathsf{TK} \leftarrow\!\!\$\ \mathsf{TKGen}(f, \mathsf{Msk})$
$(\mathsf{Msk}, \mathsf{Mpk}) \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda)$	Return c	$\mathsf{TKList} \leftarrow f : \mathsf{TKList}$
$b' \leftarrow\!\!\$\ \mathcal{A}^{\mathbf{O}}(\mathsf{Mpk})$		Return TK
Return $(b' = b)$		

Fig. 1. Game defining the IND-CPA security of an FE scheme. An adversary is legitimate if: 1) it calls $\mathbf{LR}$ once and with a pair $(\mathsf{m}_0, \mathsf{m}_1)$ such that $\mathsf{R}(\mathsf{m}_0, \mathsf{m}_1)$ holds; 2) for all $f \in \mathsf{TKList}$ have $f(\mathsf{m}_0) = f(\mathsf{m}_1)$; and 3) in the token non-adaptive model, it does not call $\mathbf{Token}$ after $\mathbf{LR}$.

SECURITY. Not only can we relate the syntax of functional encryption schemes to that of existing primitives but, under the appropriate potential leakage relations, we can also reduce IND-CPA security of an FE scheme to an existing primitive and vice versa. Our choices therefore lead to a notion of functional encryption scheme that is indeed a generalization of existing cryptographic primitives.

RELATION WITH O'NEILL'S DEFINITION. In [14] the implicit potential leakage relation is the equality of the message lengths, i.e., $\mathsf{R}(\mathsf{m}_0, \mathsf{m}_1) := (|\mathsf{m}_0| = |\mathsf{m}_1|)$. Although this is a natural choice, the resulting security definition fails to generalize those for IBE schemes (be it anonymous or non-anonymous). Our choice for the potential leakage relation deals with these issues in a cleaner way and is closer in spirit to that in [5]. It is straightforward to see that a feasibility result under O'Neill's definitional choices leads to a feasibility result in our setting with a fixed function space, full correctness, and with respect to the length equality relation. The converse also holds.

RELATION WITH BSW. Boneh et al. [5] define a special empty key (function) ϵ that is aimed at capturing information about encrypted messages that might be publicly

recoverable from ciphertexts (typically including the message length). However, this requirement implies that the standard syntax definitions for PKE, IBE, and other primitives do not naturally generalize to functional encryption and deviates from our goal. In BSW, for example, it is stated that an IBE should attach the message length and target identity to the ciphertext to strictly meet this requirement. We believe that our approach via relation R separates security issues from syntactic and correctness issues, while still maintaining the flexibility of the BSW definition. More formally, if the potential leakage relation is defined to be $\mathsf{R}(\mathsf{m}_0, \mathsf{m}_1) := (\epsilon(\mathsf{m}_0) = \epsilon(\mathsf{m}_1))$, queries that allow adversaries to exploit the empty token are excluded.

Given the discussion above, we can translate between feasibility results for the BSW definition and our definition. Any scheme that is secure under the BSW definition yields a fully correct and secure scheme under our definition, when one introduces the appropriate syntactic changes and adopts an adequate potential leakage relation. A conversion in the other direction implies transforming the scheme so that it explicitly leaks information through the empty token to match the restrictions imposed by R. In this case, full correctness and security in our model, yields a BSW-secure scheme.

4 Limitations of the Models by BSW and O'Neill

A closer look at the IND-CPA notion of security for FE schemes reveals that it is inadequate for general functionalities. The problem is as follows [5]. For some functionalities the restriction on the $\mathbf{LR}$ oracle imposing that $f(\mathsf{m}_0) = f(\mathsf{m}_1)$ can prevent the adversary from simultaneously extracting the token for f and launching a meaningful attack. For example, if the function is injective, any adversary extracting the token for this function will be prevented from querying anything other than $\mathsf{m}_0 = \mathsf{m}_1$ from the challenge oracle. However, in this case, the adversary will have no chance of winning.

Boneh et al. go on to turn this observation into a concrete functional encryption scheme supporting a one-way permutation that is intuitively insecure, but can be easily shown to satisfy the IND-CPA security definition. Roughly speaking, in this scheme one encrypts m under a standard PKE scheme. The token for the one-way permutation function f is the secret key for the PKE. Upon decryption, one first recovers m and then computes $f(\mathsf{m})$. The scheme is clearly correct. However, since $f(\mathsf{m})$ hides m computationally, the functional encryption scheme is not guaranteeing that the decryptor learns no more about the encrypted message than that which is leaked by $f(\mathsf{m})$. On the other hand, one can easily show that this FE scheme is IND-CPA-secure if the underlying PKE is itself IND-CPA-secure: if an adversary extracts the token for f, which is a permutation, then it is bound to calling the challenge oracle on $\mathsf{m}_0 = \mathsf{m}_1$; if it does not extract the token, then a simple reduction shows that it is attacking the PKE scheme. Boneh et al. also show that this scheme cannot be proven semantically secure, providing evidence that this is the correct notion of security for FE.

A Weakness in the BSW Model. We now follow the same approach to demonstrate an intuitively insecure scheme that can be proven BSW semantically secure. We restrict ourselves to the non-adaptive token extraction model so as not to fall within the range of impossibility results established in [5]. (This only strengthens our argument.)

Our argument also goes through for the weaker definition of semantic security that is used in [5, Definition 5] to present a (stronger) impossibility result.

Consider an FE scheme constructed from a PKE scheme and a one-way trapdoor permutation TDP. The scheme provides the expected functionality by encrypting an input message under a standard PKE, and evaluating the TDP upon decryption. The token corresponding to the TDP is simply the PKE secret key. The trapdoor for the TDP is *not* kept as part of the secret parameters, and to make the point clearer one should think of it as being "destroyed" upon generation. Consider also that the intentional leakage for this scheme is defined as $|\mathsf{m}|$. The scheme is clearly correct. Following the same reasoning as in the previous counterexample, this scheme leaks too much information to a decryptor holding a token for f: it will learn m, whereas only the image under the TDP should be leaked.

In the full version [2] we present a BSW simulator that always succeeds in simulating $\mathcal{A}$'s output, as long as the underlying PKE is $\mathsf{IND\text{-}CPA}$-secure. If the adversary extracts the token for f, then the simulator is able reconstruct a perfect simulation of the ciphertext in the real game using the trapdoor for the TDP that it (abusively) keeps in its state. On the other hand, if the adversary does not extract the token, then any adversary/distinguisher pair that distinguish the simulation can be used to break the $\mathsf{IND\text{-}CPA}$ security of the underlying PKE scheme. This counterexample can be extended to FE schemes where the function space is fixed and independent of the global parameters.

Potential Shortcomings in O'Neill's Model. There is a fundamental difference between O'Neill's definition of semantic security and that of BSW: the simulator is no longer in control of the generation of systems parameters. In return, a token-extraction oracle is provided in the ideal game. This means that the same strategy we presented above to argue for the inadequacy of the BSW definition does *not* directly apply. Nevertheless, other potential problems remain that we discuss next.

POTENTIALLY INSECURE SCHEME 1. We modify the BSW counterexample scheme as follows. The setup procedure is similar to before, except that the trapdoor for the randomly chosen permutation f is no longer destroyed but kept in the master secret key. The token-generation algorithm is modified so that the token for the TDP now also contains the trapdoor. The encryption and decryption routines are as before. This scheme can be proven secure under O'Neill's definition: although the simulator cannot generate the trapdoor information itself, this will become available once the adversary extracts the token for f. It is unclear if this scheme is intuitively insecure as the ciphertext does not leak any information beyond that leaked by images *and tokens*.

POTENTIALLY INSECURE SCHEME 2. Consider the following trivial construction of an FE scheme supporting its own encryption circuit. Take a PKE scheme and set the message space of the FE scheme to be (m, r) pairs. Take a PKE keys $(\mathsf{sk}, \mathsf{pk})$ and set the master secret key to be sk and the master public key to be pk. To functionally encrypt m re-encrypts under pk the ciphertext c resulting from $\mathsf{Enc}(\mathsf{m}, \mathsf{pk}; r)$. The decryption token is simply sk and decryption recovers and outputs c. This construction is correct and it can be shown to be semantically secure under the previous semantic security definitions. It is also unclear whether it should be classified as insecure. On the one hand, it is hard to argue that it is intuitively insecure. This is because the function is evaluated

on the sender's side *and* encrypted under a secure encryption scheme. Furthermore, the decryptor is the legitimate holder of the decryption key, and hence from its perspective there is no security property associated with the evaluated function. However, one can also consider things from the perspective of the encryptor, e.g., she might expect that token holders recover nothing but a ciphertext, but this is not the case.

5 The New Semantic Security Model

We now propose a new definition of semantic security that avoids the above problems while simultaneously achieving several other goals of interest. Our definition is designed to be strong enough to exclude the clearly intuitively insecure counterexample we presented for the BSW definition and capture adaptive token extractions, without being infeasible to achieve due to its excessive strength. Furthermore, the definition should be compatible with the standard definitional approaches for PKE and IBE schemes.

Definition 2 (Semantic Security). *Let games* $\mathsf{SS\text{-}Real}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{D}}$ *and* $\mathsf{SS\text{-}Ideal}_{\mathsf{FE},\mathsf{R},\mathcal{S},\mathcal{D}}$ *be as shown in Figure 2. The semantic security of an FE scheme relative to potential leakage relation* R *requires that for any PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, there exists a legitimate PPT simulator* $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ *such that for all PPT distinguishers* $\mathcal{D}$ *the following advantage function is negligible.*

$$\mathbf{Adv}^{\mathsf{ss\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{S},\mathcal{D}}(\lambda) := \Pr\left[\mathsf{SS\text{-}Real}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{D}}(\lambda) \Rightarrow \mathsf{T}\right] - \Pr\left[\mathsf{SS\text{-}Ideal}_{\mathsf{FE},\mathsf{R},\mathcal{S},\mathcal{D}}(\lambda) \Rightarrow \mathsf{T}\right]$$

Game $\mathsf{SS\text{-}Real}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{D}}(\lambda)$:
$\mathsf{FuncList} \leftarrow [\,]$
$(\mathsf{Msk}, \mathsf{Mpk}) \leftarrow\$\ \mathsf{Setup}(1^\lambda)$
$(\mathcal{M}, \mathsf{st}) \leftarrow\$\ \mathcal{A}_1^{\mathbf{Token}}(\mathsf{Mpk})$
$(\mathsf{m}, h, t) \leftarrow\$\ \mathcal{M}$
$\mathsf{c} \leftarrow\$\ \mathsf{Enc}(\mathsf{m}, \mathsf{Mpk})$
$v \leftarrow\$\ \mathcal{A}_2^{\mathbf{Token}}(\mathsf{c}, h, \mathsf{st})$
$\mathsf{trace} \leftarrow (\mathsf{Mpk}, \mathcal{M}, t, \mathsf{FuncList})$
Return $\mathcal{D}(\mathsf{trace}, v)$

Game $\mathsf{SS\text{-}Ideal}_{\mathsf{FE},\mathsf{R},\mathcal{S},\mathcal{D}}(\lambda)$:
$\mathsf{FuncList} \leftarrow [\,]; \mathsf{m} \leftarrow \perp$
$(\mathsf{Msk}, \mathsf{Mpk}) \leftarrow\$\ \mathsf{Setup}(1^\lambda)$
$(\mathcal{M}, \mathsf{st}) \leftarrow\$\ \mathcal{S}_1^{\mathbf{Eval}}(\mathsf{Mpk})$
$(\mathsf{m}, h, t) \leftarrow\$\ \mathcal{M}$
$\mathsf{ImgList} \leftarrow [f(\mathsf{m}) : f \in \mathsf{FuncList}]$
$v \leftarrow\$\ \mathcal{S}_2^{\mathbf{Eval}}(\mathsf{ImgList}, h, \mathsf{st})$
$\mathsf{trace} \leftarrow (\mathsf{Mpk}, \mathcal{M}, t, \mathsf{FuncList})$
Return $\mathcal{D}(\mathsf{trace}, v)$

oracle $\mathbf{Eval}(f)$:
$\mathsf{TK} \leftarrow\$\ \mathsf{TKGen}(f, \mathsf{Msk})$
$\mathsf{FuncList} \leftarrow f : \mathsf{FuncList}$
Return $(\mathsf{TK}, f(\mathsf{m}))$

oracle $\mathbf{Token}(f)$:
$\mathsf{TK} \leftarrow\$\ \mathsf{TKGen}(f, \mathsf{Msk})$
$\mathsf{FuncList} \leftarrow f : \mathsf{FuncList}$
Return TK

Fig. 2. Games defining the semantic security of an FE scheme. An adversary is legitimate if: 1) $\mathsf{R}(\mathsf{m}_0, \mathsf{m}_1)$ holds for every pair of messages in $[\mathcal{M}]_1$; 2) for all second-stage $\mathbf{Token}$ queries f, we have that $f(\mathsf{m}_0) = f(\mathsf{m}_1)$ for all $\mathsf{m}_0, \mathsf{m}_1 \in [\mathcal{M}]_1$; and 3) in the token non-adaptive model, $\mathcal{A}_2$ and $\mathcal{S}_2$ do not call $\mathbf{Token}$ and $\mathbf{Eval}$ respectively.

The intuition behind the definition is as in the previous definitional approaches: an adversary should learn no more about an encrypted message than that which is explicitly revealed by the functions associated to the decryption tokens that it obtains. To this end, we require the existence of a simulator that does not have access to the ciphertext, but only to the images of the encrypted message under the same set of functions. This simulator is bound to producing an output that essentially looks like that produced by the adversary in the real world, which implies the ciphertext indeed reveals no extra information. More in detail, the simulator must emulate $\mathcal{A}_1$'s output and produce an output v that matches the information recovered by $\mathcal{A}_2$ from the ciphertext. However,

the simulator is denied access to the ciphertext, and is bound to obtaining a set of images that matches those recovered by the real-world adversary via its $\mathbf{Token}$ oracle (this last restriction is imposed by including $\mathsf{FuncList}$ in trace). Like in the indistinguishability model, the potential leakage relation can be used to exclude trivial attacks whereby the real-world adversary would obtain information trivially leaked by the ciphertext (whereas this would not be available in the ideal world). Finally, we observe that the token-extraction queries performed by the adversary in the second stage are restricted to functions that are constant over the support of the message distribution. This allows us to generalize the feasibility results that are well known for particular instances of functional encryption, namely IBE schemes. For this reason, we call this model semantic security under *restricted adaptive token-extraction attacks*.

In the full version [2] we present a detailed justification of our definitional choices. Here we summarize the main features of our definition: 1) free simulators, 2) honest parameter generation (as in O'Neill), 3) use of general distinguishers (closer to BSW), 4) message generation via a message distribution (closer to O'Neill), 5) history information (closer to BSW), and 6) hint for distinguisher (present in both O'Neill and BSW).

COMPOSITION. Observe that the $\mathsf{IND\text{-}CPA}$ definition can be shown to compose from single to multiple $\mathbf{LR}$ queries (i.e., from a single-message to a multi-message attack scenario) using a standard hybrid argument [3]. One of the crucial features of our semantic security definition is that it also composes. Below we show that a multi-message variant of our definition where the message distribution outputs a *vector* of messages (see the full version [2] for the details) is equivalent to the definition above.

Theorem 1 (Composition). *Let* FE *be a functional encryption scheme that is semantically secure under the (single-message) definition in Figure 2. Suppose that there is a polynomial* poly *such that for any single-message adversary* $\mathcal{A}$*, there is a semantic security simulator* $\mathcal{S}[\mathcal{A}]$ *such that*

$$\mathsf{Time}_{\mathcal{S}[\mathcal{A}]}(\lambda) \leq \mathsf{Time}_{\mathcal{A}}(\lambda) + \mathsf{Time}_{\mathcal{M}}(\lambda) + \mathsf{poly}(\lambda)$$
$$\mathsf{Time}_{\mathcal{S}[\mathcal{M}]}(\lambda) \leq \mathsf{Time}_{\mathcal{M}}(\lambda) + \mathsf{poly}(\lambda)$$

where $\mathcal{M}$ *is the distribution output by* $\mathcal{A}$ *and* $\mathcal{S}[\mathcal{M}]$ *is the simulated message distribution. Then for every real-world multi-message PPT adversary* $\mathcal{A}'$*, there exist a real-world single-message PPT adversary* $\mathcal{A}$ *and a multi-message PPT simulator* $\mathcal{S}'$ *such that for any distinguisher* $\mathcal{D}'$*, there exists a distinguisher* $\mathcal{D}$ *for which*

$$\mathbf{Adv}^{\mathsf{m\text{-}ss\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{A}',\mathcal{S}',\mathcal{D}'}(\lambda) \leq \mathbf{Q}(\lambda) \cdot \mathbf{Adv}^{\mathsf{ss\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{S},\mathcal{D}}(\lambda)\,,$$

where $\mathcal{S}$ *is the simulator implied by the single-message semantic security and* $\mathbf{Q}(\lambda)$ *is an upper bound on the number of messages output by message distributions.*

Proof (Overview). We give an overview of the proof for the non-adaptive case here and leave the details to in the full version [2]. The proof is essentially a simulation-based hybrid argument. Consider the attack models where in the ith hybrid, for $i = 0, \ldots, q$, the adversary has access to $(q - i)$ ciphertexts and i image lists. In each step we change a ciphertext to the corresponding image list. We show that for any adversary in the ith

hybrid model, there is an adversary in the $(i+1)$st hybrid that does equally well. To see this, note that the adversary in the ith hybrid can be viewed as a *single-message* real-world adversary that also receives some extra auxiliary information consisting of ciphertexts and image lists. By the semantic security guarantees of the scheme we may replace this adversary by an equally good one that only gets the image list for the replaced ciphertext. This concludes the proof as the qth hybrid corresponds to the ideal-world multi-message semantic security game. Note that the running time of the final simulator in the ideal game, which recursively depends on the previous simulators, stays polynomial if the condition given in the theorem is satisfied. □

6 Setup Security

Similarly to O'Neill's model, our definition of semantic security fails to exclude the counterexamples from Section 4 (because the simulator also has access to decryption tokens). This raises the question of whether our model can be strengthened further so these schemes are also ruled out. One direct approach to achieve this would be to further restrict the simulator by denying it access to tokens (as well as the master secret key) in the ideal world. We present this model in the full version of this paper [2] and show that it is infeasible to achieve (essentially because a correct simulation of tokens would imply breaking the functional encryption scheme one is trying to prove semantically secure). We therefore take a different approach.

The first observation we make is that our definition accepts these counterexamples because indeed they do *not* leak information through the ciphertext. In fact, leakage is enabled by the combined information provided by *images* and decryption tokens (or more generally the master secret key). Intuitively, once a trapdoor for a TDP is provided to a token holder, the TDP circuit essentially becomes an efficiently invertible encoding function from messages onto images, offering no (intuitive) security whatsoever. From the point of view of the semantic security definition, where the aim is to exclude schemes where ciphertexts leak more information than that which is leaked by images, these counterexample schemes should therefore be considered secure.

However, it is still a reasonable security goal to expect that tokens do not help a token holder to extract information from images that would otherwise be hidden by the functionality. We therefore consider the stronger setting where the master secret key is required not to compromise the security properties of the supported functionalities. Informally, this means that even the trusted authority holding the master secret key should not be able to hold any "trapdoor" information on the functionalities supported by the functional encryption scheme.

A NEW NOTION OF SECURITY. Our approach to formalizing this security notion, which we call *setup security*, is as follows. We consider an attack scenario where an adversary is given the master secret key Msk and adaptively interacts with an evaluation oracle for the functionality in order to construct a trace that contains the master public key, a list of functions, a message distribution, a list of images, and a function func that models leaked information. This trace establishes the adversary's claim as to his ability to extract information from the images, which is specified by the leakage function. The FE scheme will then be setup-secure if a simulator given only the trace and the same

set of images provided to the adversary, i.e., without having access to the master secret key, can extract essentially the same information.

Definition 3 (Setup Security). *Let game* $\mathsf{SetSec}_{\mathsf{FE},\mathsf{R},\mathcal{A}}$ *be as shown in Figure 3. The setup security of an FE scheme relative to potential leakage relation* R *requires that for any adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, there exists a simulator* $\mathcal{S}$ *such that the following advantage function is negligible.*

$$\mathbf{Adv}^{\mathsf{setsec}}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{S}}(\lambda) := 2 \cdot \Pr\left[\mathsf{SetSec}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{S}}(\lambda) \Rightarrow \mathsf{T}\right] - 1\,.$$

Game $\mathsf{SetSec}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{S}}(\lambda)$:	**oracle** $\mathbf{Eval}(f)$:
$\mathsf{FuncList} \leftarrow [\,]$; $b \leftarrow_{\$} \{0,1\}$	$\mathsf{FuncList} \leftarrow f : \mathsf{FuncList}$
$(\mathsf{Msk}, \mathsf{Mpk}) \leftarrow_{\$} \mathsf{Setup}(1^\lambda)$	Return $f(\mathsf{m})$
$(\mathcal{M}, \mathsf{st}) \leftarrow_{\$} \mathcal{A}_1(\mathsf{Msk}, \mathsf{Mpk})$	
$\mathsf{m} \leftarrow_{\$} \mathcal{M}$	
$(\mathsf{func}, v_0) \leftarrow_{\$} \mathcal{A}_2^{\mathbf{Eval}}(\mathsf{st})$	
$\mathsf{ImgList} \leftarrow [f(\mathsf{m}) : f \in \mathsf{FuncList}]$	
$\mathsf{trace} \leftarrow (\mathsf{Mpk}, \mathcal{M}, \mathsf{FuncList}, \mathsf{ImgList}, \mathsf{func})$	
$v_1 \leftarrow_{\$} \mathcal{S}(\mathsf{trace})$	
Return $(v_b = \mathsf{func}(\mathsf{m}))$	

Fig. 3. Game defining the setup security of an FE scheme. An adversary is legitimate if $\mathsf{R}(\mathsf{m}_0, \mathsf{m}_1)$ holds for every pair of messages in $[\mathcal{M}]$.

SETUP SECURITY VIA CONDITIONAL PREIMAGE SAMPLING. It is easy to see that all the potentially insecure TDP-based counterexamples that we have introduced are excluded by the setup security definition. We now show that the definition allows natural classes of functionalities to be proven setup secure. To this end, we introduce a notion of *conditional preimage samplability*. Roughly speaking, this asserts that given a message distribution $\mathcal{M}$ and a list of functions $[f_i]_{i=1}^n$, it is possible to efficiently sample from $\mathcal{M}$ when this is conditioned on a set of images $[f_i(\mathsf{m})]_{i=1}^n$ for some $\mathsf{m} \in [\mathcal{M}]$. The actual definition is slightly more complex, as we need to deal with possible adversarial adaptiveness as well as define indistinguishability of conditional distributions in a meaningful way. We leave the details to the full version [2], where we also prove the results in this section. The following theorem shows that conditional preimage samplability is a sufficient condition for setup security.

Theorem 2 (CPS $\Rightarrow$ Secure Setup). *Any functional encryption supporting a CPS functionality relative to potential leakage relation* R *is setup-secure with respect to* R*.*

CONCRETE SETUP-SECURE SCHEMES. We now look at concrete functionalities and show that setup security is already achieved by many existing functional encryption schemes. We begin by defining a broad class of functionalities where an image either entirely reveals the encrypted message, or nothing at all.

Definition 4 (All-or-Nothing Functionality). *We say a functional encryption scheme supports an all-or-nothing (ANOT) functionality if for all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{Mpk}, \mathsf{Msk}) \in [\mathsf{Setup}(1^\lambda)]$*, all* $f \in \mathsf{FunSp}$*, and all* $\mathsf{m} \in \mathsf{MsgSp}$ *we have that* $f(\mathsf{m}) \in \{\mathsf{m}, \perp\}$*.*

As an example, consider predicate encryption systems [5] where the message space is partitioned into pairs $\mathsf{m} = (\mathsf{x}, \mathsf{idx})$. Here, x is a hidden payload and idx is extra information that determines which tokens can be used to recover x from a ciphertext encrypting m. More precisely, each secret key is associated with a predicate P, and the payload can be recovered whenever $P(\mathsf{idx}) = \mathsf{T}$. Formally,

$$f_P(\mathsf{x}, \mathsf{idx}) := \begin{cases} (\mathsf{x}, \mathsf{idx}) & \text{if } P(\mathsf{idx}) = \mathsf{T}\,; \\ \perp & \text{otherwise.} \end{cases}$$

Observe that we include idx in the output of the functionality when the predicate evaluates to T, thereby rendering the functionality all-or-nothing. It is easy to see that PKE and IBE schemes are examples of ANOT schemes. For PKE schemes this is obvious, since the functionality is the identity function and the index space is empty. For IBE, observe that whenever the output of the functionality is not $\perp$, the identity (i.e., the index) is also implicitly leaked by the functionality, thereby revealing the full message. Furthermore, this also includes variants of inner-product encryption, hidden vector encryption, etc., where a successful decryption operation explicitly reveals the encrypted index. Our first positive result for setup security is given by the following theorem.

Theorem 3 (ANOT $\Rightarrow$ CPS). *Any ANOT functionality is conditionally preimage samplable in* expected *polynomial time (for any potential leakage relation).*

The intuition behind the proof is as follows. Since the functionality is ANOT, there are two possible trace outcomes. In the first, the adversary queries a function which acts as the identity map on the message sampled from $\mathcal{M}$. Here the sampler can simply return the message. In the second, all the image values are $\perp$. Here the sampler will repeatedly sample a message from $\mathcal{M}$ until it maps to $\perp$ under all functions given to it in the trace. The number of retries, conditioned on a given trace, will depend on the probability that the image list is all $\perp$. However, the overall expected number of retries can be shown to be 1. In the full version [2] we discuss why the sampler we construct cannot be converted into a *strict* PPT black-box sampler by truncating the execution time. Combining Theorems 2 and 3 we obtain the following corollary.

Corollary 1. *Any FE scheme supporting an ANOT functionality has a secure setup procedure w.r.t. expected PPT simulators and arbitrary potential leakage relations.*

PUBLIC-INDEX PREDICATE ENCRYPTION. We now show that there exists a large class of FE schemes for which we can construct a strict PPT conditional preimage sampler. Intuitively, such schemes leak more information about encrypted messages which, when provided to the sampler, allows this stronger result to go through (in our framework, this is captured by the potential leakage relation.

Definition 5 (Jointly All-or-Nothing Functionality). *We say an FE scheme supports a jointly all-or-nothing (JNOT) functionality relative to potential leakage relation* R *if, for all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{Msk}, \mathsf{Mpk}) \in [\mathsf{Setup}(1^\lambda)]$*, all subsets* $\mathcal{F} \subseteq \mathsf{FunSp}$*, all message distributions* $\mathcal{M}$ *where* $\mathsf{R}(\mathsf{m}_0, \mathsf{m}_1) = \mathsf{T}$ *for all* $\mathsf{m}_0, \mathsf{m}_1 \in [\mathcal{M}]$*, we have that*

$$\forall \mathsf{m} \in [\mathcal{M}],\ \exists f \in \mathcal{F},\ f(\mathsf{m}) = \mathsf{m} \quad \vee \quad \forall \mathsf{m} \in [\mathcal{M}],\ \forall f \in \mathcal{F},\ f(\mathsf{m}) = \perp\ .$$

In this definition the all-or-nothing property is no longer formulated over a class of admissible message distributions defined by the potential leakage relation. Concretely, R constrains the support of the message distribution $\mathcal{M}$ in such a way that, for any subset of functions $\mathcal{F}$ extracted from the function space, the list of images will be guaranteed to, either totally reveal m, or to information theoretically preserve the entropy of the message distribution. We now show that this definition is satisfied by a large class of all-or-nothing functionalities, corresponding to predicate encryption systems with *public index* [5]. For such schemes, no claim about hiding idx is made. This means that their security is analyzed with respect to the special potential leakage relation

$$\mathsf{R}^*((\mathsf{x}_0, \mathsf{idx}_0), (\mathsf{x}_1, \mathsf{idx}_1)) := (|\mathsf{x}_0| = |\mathsf{x}_1| \ \wedge \ \mathsf{idx}_0 = \mathsf{idx}_1) \ .$$

The fact that message distributions are now restricted by R^* yields the following result.

Theorem 4. *All predicate encryption systems are JNOT with respect to* R^*.

The previous result includes primitives such as PKE, (non-anonymous) IBE, non-attribute-hiding ABE, and inner-product encryption that *reveals* the index in the ciphertext. We now state the final result of this section.

Theorem 5 (JNOT $\Rightarrow$ CPS). *Take an FE scheme supporting a JNOT functionality with respect to potential leakage relation* R. *Then this scheme is conditionally preimage samplable (in strict polynomial time) with respect to* R.

The intuition behind the proof of this theorem is exactly the same as in Theorem 3. The difference to all-or-nothing functionalities is that the JNOT property guarantees that, either the sampler gets the challenge message in the image list, or sampling a message from the message distribution yields a valid result. We obtain the following corollary.

Corollary 2. *All (public-index) predicate encryption systems are setup-secure with respect to potential leakage relation* R^*.

7 Preimage Samplability

Despite the shortcomings of indistinguishability models highlighted in [14,5], O'Neill shows that, for certain classes of functionalities, indistinguishability-based security is no less adequate than his proposed notion of semantic security. Indeed, it is shown in [14] that if an FE scheme is *preimage samplable* (see the full version [2]) then, in the non-adaptive token-extraction attack scenario, indistinguishability and semantic security are equivalent. Furthermore, functionalities such as those for IBE and inner-product encryption are shown to be preimage samplable. In light of the new syntactical and definitional approach introduced above, we propose a modified definition of preimage samplability and show that a similar result holds. Our definition, however, permits extending the equivalence result to the *multi-message and restricted adaptive* token extraction model, and hence generalizes known results for, e.g., IBE schemes, in this area. This is an important extension, as (restricted) adaptive extraction of secret keys is the standard attack model for all predicate encryption systems.

Definition 6 ((Un)Restricted Preimage Samplability). *Let game* $\mathsf{PS}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathsf{Samp},\mathsf{mode}}$ *be as defined in Figure 4. We call an FE (un)restricted preimage samplable for the potential leakage relation* R *if, for any algorithm* $\mathcal{A}$ *there exists a sampling algorithm* Samp *such that the following advantage function is negligible.*

$$\mathbf{Adv}^{\text{mode-ps}}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathsf{Samp}}(\lambda) := \Pr\left[\mathsf{PS}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathsf{Samp},\mathsf{mode}}(\lambda) \Rightarrow \mathsf{F}\right] .$$

Game $\mathsf{PS}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathsf{Samp},\mathsf{mode}}(\lambda)$:
$(\mathsf{Msk},\mathsf{Mpk}) \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda)$
$(\mathcal{M},[f_j]_{j=1}^n) \leftarrow\!\!\$\ \mathcal{A}(\mathsf{Msk},\mathsf{Mpk})$
$\mathsf{m}_0 \leftarrow\!\!\$\ \mathcal{M}$
$\mathsf{m}_1 \leftarrow\!\!\$\ \mathsf{Samp}(\mathcal{M},[(f_j,f_j(\mathsf{m}_0))]_{j=1}^n,\mathsf{Mpk})$
If $(\exists j\ :\ f_j(\mathsf{m}_0) \neq f_j(\mathsf{m}_1))$ Return F
If $\neg\mathsf{R}(\mathsf{m}_0,\mathsf{m}_1)$ Return F
If $(\mathsf{mode} = \mathsf{res} \wedge \mathsf{m}_1 \notin [\mathcal{M}])$ Return F
Return T

Fig. 4. Game defining (un)restricted preimage samplability, for mode $\mathsf{mode} \in \{\mathsf{res},\mathsf{unres}\}$. $\mathcal{A}$ is legitimate if $\mathsf{R}(\mathsf{m}_0,\mathsf{m}_1)$ holds for all $\mathsf{m}_0,\mathsf{m}_1 \in [\mathcal{M}]$.

COMPARISON WITH O'NEILL PS DEFINITION. Our definition differs from that in [14] in several aspects. First, the adversary now has access to $(\mathsf{Msk},\mathsf{Mpk})$ rather than 1^λ only. Access to Mpk is consistent with our syntax of FE schemes, which permits generation of function space together with the master public key. Access to Msk in needed when arguing that the actions of some IND-CPA adversary contradict preimage samplability. More precisely, the adversary may use information dependent on the Msk (i.e., decryption tokens) to come up with a non-samplable message. This issue seems to have been overlooked in [14]. Second, the definition is parameterized by a potential leakage relation R. This ensures that the sampler obtains as much information about the challenge message as a real-world semantic security adversary. Technically, this allows the equivalence proof to go through (for the non-adaptive case) for a larger class of functional encryption schemes than those covered by O'Neill. For example, our results cover those schemes that can be captured using our syntactic conventions, but not under those in [14]. (A simple example of this is standard (non-anonymous) IBE.) Finally, the adversary now outputs a message distribution rather than a single message. The unrestricted sampler, similar to O'Neill's, is only bound to producing a message that collides with m_0 on all functions. The (stronger) restricted sampler is bound to return an m_1 that is in the support of $\mathcal{M}$. As we shall see, this is necessary to enable extending the equivalence result to the *restricted adaptive* token extraction setting. For the unrestricted case this condition is dropped, and we end up with a definition which is implied by (and hence *weaker* than) O'Neill PS definition (the $\mathcal{M}$ and $\mathcal{A}$ can be merged). Consequently, all positive feasibility results in [14] carry over to our setting.

The following theorem, proven in the full version of this paper [2], establishes equivalence between our two notions of FE security for restricted preimage samplable schemes and restricted adaptive token extraction scenarios.

Theorem 6 (Equivalence under PS). *Fix potential leakage relation* R. *For every adversary* $\mathcal{A}$ *against the* IND-CPA *security of scheme* FE*, there exist a (single-message)* SS-Real *adversary* $\mathcal{B}$ *and a distinguisher* $\mathcal{D}$ *such that for any simulator* $\mathcal{S}$

$$\mathbf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{A}}(\lambda) \leq 2 \cdot \mathbf{Adv}^{\mathsf{ss\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{B},\mathcal{S},\mathcal{D}}(\lambda)\,.$$

Furthermore, for every single-message SS-Real *adversary* $\mathcal{A}$*, there is a* PS *adversary* $\mathcal{C}$ *with sampler* Samp*, and a* SS-Ideal *simulator* $\mathcal{S}$ *such that for every distinguisher* $\mathcal{D}$ *there is an* IND-CPA *adversary* $\mathcal{B}$ *with*

$$\mathbf{Adv}^{\mathsf{ss\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{A},\mathcal{S},\mathcal{D}}(\lambda) \leq \mathbf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{FE},\mathsf{R},\mathcal{B}}(\lambda) + \mathbf{Adv}^{\mathsf{res\text{-}ps}}_{\mathsf{FE},\mathsf{R},\mathcal{C},\mathsf{Samp}}(\lambda)\,.$$

The running time of $\mathcal{S}$ *in the ideal world is that of running* $\mathcal{A}$ *in the real world plus the running time of* Samp.

The guarantee on the running time of the simulator allows us to obtain semantic security in the multi-message scenario from single-message indistinguishability via Theorem 1, provided that the running time of the sampler is independent of the running time of the adversary. This is indeed the case in our feasibility results below.

REMARK. The above result can be extended to a setting where samplers, adversaries and simulators may execute in expected polynomial time. More precisely, for expected PPT preimage samplers, one can prove that IND-CPA security with respect to expected PPT adversaries is equivalent to semantic security when both the real-world adversary and the ideal-world simulator may run in expected polynomial time.

FEASIBILITY. We conclude this section with a discussion of the feasibility results we obtain with the new definition of preimage samplability. On the negative side, it is easy to see that no FE scheme supporting a one-way function can be preimage samplable with respect to O'Neill's or our definition. On the positive side, and on top of all of O'Neill's feasibility results for the non-adaptive token extraction scenario, the following theorem yields feasibility results for restricted preimage samplability for a large class of functionalities, which in turn immediately yield positive feasibility results for semantically secure functional encryption under restricted adaptive token extraction scenarios.

Theorem 7 (CPS $\Rightarrow$ PS). *Any conditionally preimage samplable functional encryption scheme is also restricted preimage samplable.*

Combining this theorem (proved in the full version [2]) with the previous results we get that any IND-CPA-secure FE scheme supporting an ANOT functionality is semantically secure under restricted adaptive token-extraction attacks and also enjoys setup security, both with respect to expected PPT simulators. For the special cases where the potential leakage relation allows us to construct a strict PPT preimage sampler (e.g., PKE, IBE, and other predicate encryption schemes that explicitly leak the index) the result holds for strict PPT simulators. Furthermore, since the sampler executes $\mathcal{M}$ once, this allows us to extend the implication from single-message IND-CPA security to multi-message semantic security under restricted adaptive token-extraction attacks.

8 Inner-Product Encryption

Inner-product encryption (IPE) [10] is a form of functional encryption where the index space corresponds to vectors $\boldsymbol{x}$ in $\mathbb{Z}_q^m$, and each secret key is also associated with a vector $\boldsymbol{y}$ in $\mathbb{Z}_q^m$. The associated predicate is given by

$$P_{\boldsymbol{y}}(\boldsymbol{x}) := \begin{cases} \mathsf{T} & \text{if } \langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0 \mod q\,; \\ \mathsf{F} & \text{otherwise.} \end{cases}$$

Without loss of generality, we will concentrate on the predicate-only version of inner-product encryption, where the payload is empty and the functionality is $f_{\boldsymbol{y}}(\mathsf{m}) = P_{\boldsymbol{y}}(\boldsymbol{x})$. Note that IPE is *not* an all-or-nothing functionality, since upon successful decryption one does not learn $\boldsymbol{x}$.

Our goal is to show that inner-product encryption is not restricted preimage samplable. To this end, we will rely on well-established intractable problems related to finding short solutions to linear equations. More precisely, we will be relying on the Small Integer Solution (SIS) problem and a decisional variant of it that we call DSIS [12,6,11] (see the full version [2] for the details). We will show that, for certain parameters q, m in the inner-product functionality, no restricted preimage sampler can be successful against the PS adversary $\mathcal{A}$ shown in Figure 5. This adversary is parameterized by four values n, m, q, and d, which we assume to be polynomial in the security parameter. This guarantees that the algorithm runs in PPT.

Algorithm $\mathcal{A}_{q,n,m,d}(\mathsf{Msk}, \mathsf{Mpk})$:
For i from 1 to n do
  $\boldsymbol{y}_i \leftarrow\!\!\$\ \mathbb{Z}_q^m$
Set $\mathcal{M}$ to uniform on $\mathrm{B}(d)^m$
Return $(\mathcal{M}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$

Fig. 5. PS adversary for the inner-product encryption

The formal statement of our result is as follows.

Theorem 8 (IPE Is Not Restricted PS). *Let $\mathcal{A}$ be the PS adversary in Figure 5. Then for any PPT sampler* Samp*, there exist PPT adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that*

$$1 - \mathbf{Adv}^{\mathsf{res\text{-}ps}}_{\mathsf{FE},\mathsf{R},\mathcal{A}_{q,n,m,d},\mathsf{Samp}}(\lambda) \leq \mathbf{Adv}^{(q,m,n,d)\text{-}\mathsf{dsis}}_{\mathcal{B}_1}(\lambda) + \mathbf{Adv}^{(q',m',n',\beta)\text{-}\mathsf{sis}}_{\mathcal{B}_2}(\lambda) + \nu(\lambda)\,,$$

where $d = q^{n/m}$, $q' = q$, $m' = m$, $n' = n/q - \sqrt{n/q}\log(n/q)$, $\beta = d\sqrt{m}$, and $\nu(\lambda)$ is a negligible function depending on q and n.

We note that this is a stronger result than what we need as it establishes that the sampler will fail with *overwhelming* probability. We leave the details of the proof to the full version [2], where we also briefly discuss how to extend the theorem to large values of q, and give a high-level overview here.

The main idea behind the proof is that a successful sampler should match the zero values in the image list it receives, while being restricted to outputting solutions in the

support of the message distribution, which consists of *small* vectors. In other words, the sampler is solving a system of linear equations with a small solution. This allows us to establish a connection with the SIS problem. Despite this, in order to solve a SIS problem instance using the sampler, we need to make sure that the sampler is forced to match sufficiently many zeros. (Note it cannot be the case that sampling a random message leads to only zero image values as otherwise we can preimage sample by repeated sampling as before.) Hence enough zero and nonzero values must be present in the image list. We achieve this by making sure the adversary $\mathcal{A}$ returns more vectors $\boldsymbol{y}_i$ than the SIS dimension n'. But now there is a problem as we do not know the image values for the newly generated vectors. This is where we appeal to the DSIS problem and simply assume these values are random: any change in the sampler's success probability would translate to a DSIS break. We are now in a position where we can reduce to the SIS problem. We assign the rows of the SIS matrix to (some of) the zeros in the randomly generated values, and assign the newly generated rows to the remaining ones. A successful sampler for this set of images would also solve the SIS problem instance.

Acknowledgements. We would like to thank Daniel Cabarcas, Angelo De Caro, Gottfried Herold, Vincenzo Iovino, Vadim Lyubashevsky, and Giuseppe Persiano for helpful discussions. Pooya Farshim is supported by grant Fi 940/4-1 of the German Research Foundation (DFG). This work was partly done while Manuel Barbosa was visiting École Normale Supérieure – Paris. This work is part-financed by National Funds through the FCT - Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project ENIAC/2224/2009 and by ENIAC Joint Undertaking under grant agreement number 120224.

References

1. Attrapadung, N., Cui, Y., Galindo, D., Hanaoka, G., Hasuo, I., Imai, H., Matsuura, K., Yang, P., Zhang, R.: Relations Among Notions of Security for Identity Based Encryption Schemes. In: Correa, J.R., Hevia, A., Kiwi, M. (eds.) LATIN 2006. LNCS, vol. 3887, pp. 130–141. Springer, Heidelberg (2006)
2. Barbosa, M., Farshim, P.: On the semantic security of functional encryption schemes. Cryptology ePrint Archive, Report 2012/474 (2012) Full version of this paper
3. Bellare, M., Desai, A., Pointcheval, D., Rogaway, P.: Relations among Notions of Security for Public-Key Encryption Schemes. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 26–45. Springer, Heidelberg (1998)
4. Bellare, M., O'Neill, A.: Semantically-secure functional encryption: possibility results, impossibility results and the quest for a general definition. Cryptology ePrint Archive, Report 2012/515 (2012)
5. Boneh, D., Sahai, A., Waters, B.: Functional Encryption: Definitions and Challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011)
6. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork, C. (ed.) STOC 2008, pp. 197–206. ACM (2008)
7. Goldreich, O.: The Foundations of Cryptography. Basic Applications, vol. 2. Cambridge University Press (2004)
8. Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Sciences 28(2), 270–299 (1984)

9. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional Encryption with Bounded Collusions via Multi-party Computation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 162–179. Springer, Heidelberg (2012)
10. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
11. Lyubashevsky, V.: Lattice Signatures without Trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012)
12. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing 37(1), 267–302 (2007)
13. Nielsen, J.B.: Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 111–126. Springer, Heidelberg (2002)
14. O'Neill, A.: Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556 (2010)
15. Waters, B.: Functional Encryption for Regular Languages. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 218–235. Springer, Heidelberg (2012)

Attribute-Based Encryption with Fast Decryption

Susan Hohenberger and Brent Waters

[1] Johns Hopkins University
susan@cs.jhu.edu
[2] University of Texas at Austin
bwaters@cs.utexas.edu

Abstract. Attribute-based encryption (ABE) is a vision of public key encryption that allows users to encrypt and decrypt messages based on user attributes. This functionality comes at a cost. In a typical implementation, the size of the ciphertext is proportional to the number of attributes associated with it and the decryption time is proportional to the number of attributes used during decryption. Specifically, many practical ABE implementations require one pairing operation per attribute used during decryption.

This work focuses on designing ABE schemes with fast decryption algorithms. We restrict our attention to expressive systems *without* system-wide bounds or limitations, such as placing a limit on the number of attributes used in a ciphertext or a private key. In this setting, we present the first key-policy ABE system where ciphertexts can be decrypted with a constant number of pairings. We show that GPSW ciphertexts can be decrypted with only 2 pairings by increasing the private key size by a factor of $|\Gamma|$, where Γ is the set of distinct attributes that appear in the private key. We then present a generalized construction that allows each system user to independently tune various efficiency tradeoffs to their liking on a spectrum where the extremes are GPSW on one end and our very fast scheme on the other. This tuning requires no changes to the public parameters or the encryption algorithm. Strategies for choosing an individualized user optimization plan are discussed. Finally, we discuss how these ideas can be translated into the ciphertext-policy ABE setting at a higher cost.

1 Introduction

Attribute-based encryption (ABE) [18] is an expansion of public key encryption that allows users to encrypt and decrypt messages based on user attributes. In a *key-policy* ABE (KP-ABE) system, an encrypted message can be tagged with a set of attributes, such as tagging an email with the metadata "from: Alice", "to: IACR board", "subject: voting", "date: October 1, 2012", etc. The master authority for the system can issue private decryption keys to users including an access policy, such as giving to Bob a decryption key that enables him to decrypt

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 162–179, 2013.

any ciphertexts that satisfy "to: Bob" OR ("to: IACR board" AND (January 1, 2011 $\leq$"date" $\leq$ December 31, 2012)).

This access control functionality can be very powerful, but also costly. In this work, we focus on the cost of decryption. In many key-policy ABE systems, such as that of Goyal, Pandey, Sahai and Waters (GPSW) [13], the decryption algorithm requires one pairing for each attribute used during decryption. (Encryption does not require any pairings, and is thus already fast by comparison.)

It seems conceivable that one might reduce the cost of decryption by making tradeoffs elsewhere. One tradeoff we allow ourselves in this work is to increase the private key size, although we ideally want to limit any increase as much as possible. We do not, however, consider tradeoffs that increase the ciphertext size or that place any limitations on how the ABE system can be used. That is, we focus on fast decryption for the most general setting possible – an expressive, large-universe system, where there are no bounds on, say, the number of attributes that can appear in a ciphertext or private key. While good progress has been made on efficient ABE in "bounded settings", as we discuss shortly, our focus is to develop techniques for improving efficiency in the most general setting and for applications where it is infeasible to trade system-wide usability for performance.

Our Contributions. We present the first expressive and "unbounded" key-policy ABE (KP-ABE) system in a pairing setting, which requires only a constant number of pairings to decrypt any ciphertext. It builds upon the GPSW system [13]. It reduces the decryption requirements to two pairings and two exponentiations (these exponents are 1 or 0 if the access policy is a boolean formula), while increasing the number of multiplications by a factor of $|\Delta|$, where Δ is the set of distinct attributes used during decryption. It also increases the private key size by a factor of $|\Gamma|$, where Γ is the set of distinct attributes used in the private key.

We discuss several variants of this system, including a method for reducing the ciphertext size from $O(|\Delta|)$ to three group elements (in the small universe setting only) at the cost of larger private keys. We also discuss the difficulties of achieving fast decryption for ciphertext-policy ABE (CP-ABE) systems in an unbounded setting, as well as progress in the bounded setting.

In Section 5, we present generalized decryption and "key storage" algorithms that allow each system user to independently tune various efficiency tradeoffs to their liking on a spectrum that ranges from GPSW (shorter keys, slow decryption) on one end to our main KP-ABE construction (longer keys, fast decryption) on the other. This tuning requires no changes to the setup, encryption or key generation algorithms of our base construction. Rather, it is managed transparently by each user, who can choose among many possible decryption algorithms where the amount of her private key that she needs to securely store scales accordingly. We conclude with some strategies for choosing an individualized user optimization plan.

1.1 Related Work

To our knowledge, the only prior work to achieve constant pairings in decryption for an expressive KP-ABE system is that of Attrapadung, Herranz, Laguillaumie, Libert, de Panafieu and Ráfols [3,2]. The goal of their work was on achieving short ciphertexts. They were able to achieve very short ciphertexts (constant number of group elements) using novel applications of aggregation techniques that have roots in those used to achieve hierarchical identity-based encryption [14,11] with constant size ciphertexts by Boneh, Boyen and Goh [6] and those used to achieve practical broadcast encryption by Boneh, Gentry and Waters [8]. Our ideas begin at this starting point as well. However, Attrapadung et al. brought in an inner product instance as a base building block which *fundamentally* demands a bound, n, on the maximum number of attributes that can appear in a ciphertext. This choice of n forces a tradeoff between flexibility and performance.

One needs to set n high to cover the "worst" case even if the typical encryption uses far fewer attributes. Unfortunately, their private key size blows up by a factor of n, whereas our key size only increases by a factor of the number of distinct attributes used in *that particular* key. Moreover, for a ciphertext that encrypts with $|S|$ attributes, they have an added cost of $|S|$ exponentiations during decryption. This could be very costly in some situations. Suppose Alice has the key policy (A_1 AND A_2) and receives a ciphertext with many attributes $A_1, A_2, \ldots, A_{1000}$. In their system, Alice must do 1000 exponentiations, which is actually much worse than if she was only required to do one pairing per attribute used in decryption.[1]

This work aims to avoid these "worst case" penalties. In particular, we want that a user's key size be related to the complexity of her key. In their scheme, it grows with n. In decryption, the computational cost should be related to how many rows of the LSSS one must use. In their scheme, the number of exponentiations grows with the number of attributes in the ciphertext. This is a tradeoff relative to GPSW [13]. Finally, in their scheme, the number of multiplications is roughly $|I| \cdot |S|$, the number of rows used in decryption times the number of attributes in a ciphertext. Ours is $|I| \cdot |\Delta|$, the number of rows used in decryption times the number of distinct attributes used in decryption. Note that $|\Delta| \leq |S|$.

While we have pointed out some of the areas for improvement in Attrapadung et al. for comparison's sake, we do wish to stress that this was a pioneering work that showed that short ciphertexts and fast decryption was possible at all for KP-ABE. We also refer the reader to that work for an excellent summary of the efficiency of prior ABE schemes. Our work compliments theirs by taking the study of fast decryption for ABE into the unbounded realm with tighter growth.

We note that Identity-Based Encryption [19,7,10] was an early forerunner of ABE and several technique from IBE impact ABE systems. Finally, in this work, we only consider selective security [9], however, we believe our techniques

[1] The polynomial related to Y in their construction grows with the number of attributes in the ciphertext.

could apply to more recent systems proven adaptively secure using dual system encryption [15,17].

2 Background

This section covers background information. We make use of the standard definitions for access structures and linear secret sharing schemes (LSSS), as well as the conventions and notation for these employed in several prior ABE works. These are included in Appendix A for reference.

2.1 Definitions of Security for Key Policy ABE Schemes

Definition 1 (KP-ABE Algorithm Specification). *A key-policy attribute-based encryption system for message space $\mathcal{M}$ and access structure space $\mathcal{G}$ is a tuple of the following algorithms:*

Setup$(\lambda, U) \rightarrow (\mathrm{PK}, \mathrm{MK})$**.** *The setup algorithm takes as input a security parameter λ and a universe description U, which defines the set of allowed attributes in the system. It outputs the public parameters* PK *and the master secret key* MK*.*

Encrypt$(\mathrm{PK}, M, S) \rightarrow \mathrm{CT}$**.** *The encryption algorithm takes as input the public parameters* PK*, a message M and a set of attributes S and outputs a ciphertext* CT *associated with the attribute set.*

KeyGen$(\mathrm{MK}, \mathbb{A}) \rightarrow \mathrm{SK}$**.** *The key generation algorithm takes as input the master secret key* MK *and an access structure $\mathbb{A}$ and outputs a private key* SK *associated with the attributes.*

Decrypt$(\mathrm{SK}, \mathrm{CT}) \rightarrow M$**.** *The decryption algorithm takes as input a private key* SK *associated with access structure $\mathbb{A}$ and a ciphertext* CT *associated with attribute set S and outputs a message M if S satisfies $\mathbb{A}$ or the error message $\perp$ otherwise.*

The correctness property requires that for all sufficiently large $\lambda \in \mathbb{N}$, all universe descriptions U, all $(\mathrm{PK}, \mathrm{MK}) \in \mathsf{Setup}(\lambda, U)$*, all $S \subseteq U$, all* $\mathrm{SK} \in \mathsf{KeyGen}(\mathrm{MK}, \mathbb{A})$*, all $M \in \mathcal{M}$, all $\mathbb{A} \in \mathcal{G}$ and all* $\mathrm{CT} \in \mathsf{Encrypt}(\mathrm{PK}, M, S)$*, if S satisfies $\mathbb{A}$, then* $\mathsf{Decrypt}(\mathrm{SK}, \mathrm{CT})$ *outputs M.*

Security Model for KP-ABE Let $\Pi = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{KeyGen}, \mathsf{Decrypt})$ be a KP-ABE scheme for message space $\mathcal{M}$ and access structure space $\mathcal{G}$, and consider the following experiment for an adversary $\mathcal{A}$, parameter λ and attribute universe U:

The KP-ABE Experiment $\mathsf{KP\text{-}ABE\text{-}Exp}_{\mathcal{A},\Pi}(\lambda, U)$:

Setup. The challenger runs the Setup algorithm and gives the public parameters, PK to the adversary.

Phase 1. The challenger initializes an empty table T, an empty set D and an integer counter $j = 0$. Proceeding adaptively, the adversary can repeatedly make any of the following queries:

- Create($\mathbb{A}$): The challenger sets $j := j + 1$. It runs the key generation algorithm on $\mathbb{A}$ to obtain the private key SK and stores in table T the entry $(j, \mathbb{A}, \text{SK})$.
  Note: Create can be repeatedly queried with the same input.
- Corrupt(i): If there exists an i^{th} entry in table T, then the challenger obtains the entry $(i, \mathbb{A}, \text{SK})$ and sets $D := D \cup \{\mathbb{A}\}$. It then returns to the adversary the private key SK. If no such entry exists, then it returns $\perp$.
- Decrypt(i, CT): If there exists an i^{th} entry in table T, then the challenger obtains the entry $(i, \mathbb{A}, \text{SK})$ and returns to the adversary the output of the decryption algorithm on input (SK, CT). If no such entry exists, then it returns $\perp$.

Challenge. The adversary submits two equal length messages M_0 and M_1. In addition the adversary gives a set of attributes S^* such that for all $\mathbb{A} \in D$, the set S^* does not satisfy the access structure $\mathbb{A}$. The challenger flips a random coin b, and encrypts M_b under S^*. The resulting ciphertext CT^* is given to the adversary.

Phase 2. Phase 1 is repeated with the restrictions that the adversary cannot
- trivially obtain a private key for the challenge ciphertext. That is, it cannot issue a Corrupt query that would result in an access structure $\mathbb{A}$ which S^* satisfies being added to D.
- issue a decryption query on the challenge ciphertext CT^*.

Guess. The adversary outputs a guess b' of b. The output of the experiment is 1 if and only if $b = b'$.

Definition 2 (KP-ABE Security). *A KP-ABE scheme Π is CCA-secure (or secure against* chosen-ciphertext attacks*) for attribute universe U if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there exists a negligible function negl such that:*

$$\Pr[\mathsf{KP\text{-}ABE\text{-}Exp}_{\mathcal{A},\Pi}(\lambda, U) = 1] \leq \frac{1}{2} + negl(\lambda).$$

CPA Security. We say that a system is *CPA*-secure (or secure against *chosen-plaintext attacks*) if we remove the Decrypt oracle in both Phase 1 and 2.

Selective Security. We say that a system is *selectively* secure if we add an Init stage before Start where the adversary outputs the challenge attribute set S^* (instead of waiting until Challenge).

2.2 Bilinear Maps

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map with the properties: (1) Bilinearity: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$ and (2) Non-degeneracy: $e(g, g) \neq 1$. We say that $\mathbb{G}$ is a bilinear group if the group operation in $\mathbb{G}$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ are both efficiently computable. We now state an assumption used in the constructions.

Definition 3 (Decisional BDHE [6]). *Let $a, s \in \mathbb{Z}_p$ be chosen at random and g be a generator of group $\mathbb{G}$ of prime order $p \in \Theta(2^\lambda)$. The decisional q-BDHE assumption is that all probabilistic polynomial-time algorithms $\mathcal{A}$ given the vector $\boldsymbol{y}=$*

$$\mathbb{G}, p, g, g^s, g^a, \ldots, g^{(a^q)}, g^{(a^{q+2})}, \ldots, g^{(a^{2q})}$$

have an advantage negligible in λ of distinguishing $e(g,g)^{a^{q+1}s} \in \mathbb{G}_T$ from a random element in $R \in \mathbb{G}_T$. The advantage of $\mathcal{A}$ is defined as

$$\left| \Pr\left[\mathcal{A}(\boldsymbol{y}, e(g,g)^{a^{q+1}s}) = 0\right] - \Pr\left[\mathcal{A}(\boldsymbol{y}, R) = 0\right] \right|$$

where the probability is taken over the random choice of a, s in $\mathbb{Z}_p$, R in $\mathbb{G}_T$ and the generator g, and the random bits consumed by $\mathcal{A}$.

3 ABE with Fast Decryption

3.1 The Base Construction: Small Universe KP-ABE

We first describe a system for a small universe U of attributes, where $|U|$ is a polynomial in 1^λ, and the attributes are the integers $1, \ldots, U$. Subsequently, we will describe how to alter this construction to accommodate a large universe $U = \{0,1\}^*$ of attributes in the random oracle model. When we refer to our base construction in a setting where large universes are assumed, we mean this close variant. The message space is $\mathbb{G}_T$.

Setup$(\lambda, U) \to (\mathrm{PK}, \mathrm{MK})$. The setup algorithm first chooses a bilinear group $\mathbb{G}$ of prime order $p \in \Theta(2^\lambda)$. It selects a random generator $g \in \mathbb{G}$. It next selects random values $h_1, \ldots, h_{|U|} \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$. It then sets the keys as:

$$\mathrm{PK} = (\mathbb{G}, p, g, e(g,g)^\alpha, h_1, \ldots, h_{|U|}), \quad \mathrm{MK} = (\mathrm{PK}, \alpha).$$

Encrypt$(\mathrm{PK}, M, S) \to \mathrm{CT}$. The encryption algorithm takes as input the public parameters PK, a message $M \in \mathbb{G}_T$ to encrypt, and a set of attributes S. It chooses a random $s \in \mathbb{Z}_p$. The ciphertext is published as $\mathrm{CT} = (S, C, \hat{C}, \{C_x\})$ where

$$C = M \cdot e(g,g)^{\alpha s}, \ \hat{C} = g^s, \ \{C_x = h_x^s\}_{x \in S}.$$

KeyGen$(\mathrm{MK}, \mathbb{A}) \to \mathrm{SK}$. The key generation algorithm takes as input the master secret key and an LSSS access structure (W, ρ). Let W be an $\ell \times n$ matrix. The function ρ associates rows of W to attributes. Let Γ denote the set of distinct attributes the appear in the access structure matrix W; that is, $\Gamma = \{d : \exists i \in [1, \ell], \rho(i) = d\}$. The algorithm first chooses a random vector $\boldsymbol{v} = (\alpha, y_2, ..., y_n) \in \mathbb{Z}_p^n$. These values will be used to share the master secret α. For $i = 1$ to ℓ, it calculates $\lambda_i = \boldsymbol{v} \cdot W_i$, where W_i is the vector corresponding to the ith row of

W. In addition, the algorithm chooses random $r_1, \ldots, r_\ell \in \mathbb{Z}_p$. It sets the private key SK as:

$$\text{PK}, \quad (D_1 = g^{\lambda_1} \cdot h_{\rho(1)}^{r_1},\ R_1 = g^{r_1},\ \forall d \in \Gamma/\rho(1),\ Q_{1,d} = h_d^{r_1}),\ \ldots\ ,$$
$$(D_\ell = g^{\lambda_\ell} \cdot h_{\rho(\ell)}^{r_\ell},\ R_\ell = g^{r_\ell},\ \forall d \in \Gamma/\rho(\ell),\ Q_{\ell,d} = h_d^{r_\ell}).$$

In our notation above, we slightly abuse the set minus notation, and by Γ/x, where Γ is a set and x is a single element, we mean $\Gamma/\{x\}$; i.e., the set Γ with the element x removed if present.

These keys contain GPSW [13] keys with the addition of the "helper values" $Q_{i,d}$. The key size is proportional to $|\Gamma| \cdot \ell$, which is the number of distinct attributes that appear in the access matrix times the number of rows in the matrix. Since $|\Gamma| \leq \ell$, we have $|\Gamma| \cdot \ell \leq \ell^2$.

Decrypt$(\text{SK}, \text{CT}) \to M$. The decryption algorithm takes as input a key $\text{SK} = (\text{PK}, (D_1, R_1, \{Q_{1,d}\}), \ldots, (D_\ell, R_\ell, \{Q_{\ell,d}\}))$ for access structure (W, ρ) and a ciphertext $\text{CT} = (C, \hat{C}, \{C_x\}_{x \in S})$ for set S. Let W be an $\ell \times n$ matrix. The function ρ associates rows of W to attributes. If S does not satisfy the access structure, it outputs $\perp$. Suppose that S satisfies the access structure and let $I \subseteq \{1, 2, \ldots, \ell\}$ be a set of indices and $\{\omega_i\}_{i \in I} \in \mathbb{Z}_p$ be a set of constants such that:

1. For all $i \in I$, $\rho(i) \in S$.
2. $\sum_{i \in I} \omega_i \cdot W_i = (1, 0, 0, \ldots, 0)$.

We then define $\Delta = \{x : \exists i \in I,\ \rho(i) = x\}$. That is, I is the set of indices corresponding to the rows used in one possible way to decrypt the ciphertext and Δ is the set of distinct attributes associated with these rows. In general, there can be multiple such I that satisfy the above constraints. Typically, one will wish to minimize the size of I. Note that $\Delta \subseteq S$, where S is the attributes used to encrypt the ciphertext, and $\Delta \subseteq \Gamma$, the set of attributes used to create the private key.

Next we define the function f which transforms a set of attributes into an element of $\mathbb{G}$ as:

$$f(\Delta) = \prod_{x \in \Delta} h_x.$$

To decrypt, the algorithm will first do a pre-processing step on the private key. For each $i \in I$, it will compute the value

$$\hat{D}_i = D_i \cdot \prod_{x \in \Delta/\rho(i)} Q_{i,x} = g^{\lambda_i} f(\Delta)^{r_i}.$$

Next, the algorithm will do a pre-processing step on the ciphertext by computing the value

$$L = \prod_{x \in \Delta} C_x = \prod_{x \in \Delta} h_x^s = f(\Delta)^s.$$

The algorithm now recovers the value $e(g,g)^{\alpha s}$ by computing

$$e(\hat{C}, \prod_{i \in I} \hat{D}_i^{\omega_i}) / e(\prod_{i \in I} R_i^{\omega_i}, L) =$$
$$e(g^s, \prod_{i \in I} g^{\lambda_i \omega_i} f(\Delta)^{r_i \omega_i}) / e(\prod_{i \in I} g^{r_i \omega_i}, f(\Delta)^s) =$$
$$e(g,g)^{\alpha s} \cdot e(g, f(\Delta))^{s \sum_{i \in I} r_i \omega_i} / e(g, f(\Delta))^{s \sum_{i \in I} r_i \omega_i} = e(g,g)^{\alpha s}.$$

The decryption algorithm can then divide out this value from C and obtain the message M. The decryption algorithm requires the computation of only two pairing operations.

3.2 Efficiency and Tradeoffs

The main feature of the above scheme is that decryption only requires two pairings. While decryption also requires two exponentiations per row used, if the LSSS is derived from an AND/OR tree then the exponents w_i will be 1. (That is, they will be either 0 or 1, but the $w_i = 0$ rows should not be used.) Thus, decryption can be very fast. There are two tradeoffs:

1. The private key size and generation time blows up by roughly a factor of $|\Gamma|$ compared to GPSW, where Γ is the set of distinct attributes used in making the key.
2. Decryption reduces the number of pairings, but requires modular multiplications of roughly a factor of $|\Delta|$ compared to GPSW, where Δ is the set of distinct attributes used in decryption.

Thus, while there is a blow-up, this increase is tied only to the number of distinct attributes "touched" by the corresponding operation, and not by a global bound. Depending on the application, one should take into consideration whether the blow up in key size is worth it. Moreover, the decryption time could actually increase over GPSW [13] once Δ becomes sufficiently large. However, it would have to be *so large* that 2 pairings plus $|I| \cdot |\Delta|$ multiplications dominates $|\Delta|$ pairings and $|I|$ multiplications. As one benchmark, it required 8.22ms to compute a pairing for a BN256 curve with the RELIC library on a modern PC while roughly 0.0034ms to compute a modular multiplication. Thus, in a setting where $|I| = |\Delta|$ (the number of rows of the access matrix touched during decryption is the same as the number of distinct attributes touched), the decryption algorithm would need to touch over 2416 distinct attributes before GPSW would be faster. Should this occur, however, it is worth noting that the above private keys actually contain a GPSW key; thus, if this threshold was ever reached, one could revert back to doing GPSW decryption.

In Section 5, we will provide a generalized construction for finer-grained tradeoff optimization.

3.3 Large Universe Realizations

The construction in Section 3.1 can be transformed so that *any* string can be a valid attribute; that is, $U = \{0,1\}^*$, as follows: Assume that all parties have access to a hash function $H : \{0,1\}^* \to \mathbb{G}$, which will be treated as a random oracle. Remove the values $h_1, \ldots, h_{|U|}$ from the public parameters PK. For any attribute $x \in \{0,1\}^*$, define the value $h_x = H(x)$. Otherwise, follow the construction as written. Thus, the efficiency is the same, modulo additional hash function evaluations. Regarding the proof of security, let q be the maximum number of unique queries made to the random oracle. Then, this large universe construction is selectively, CPA-secure under the Decisional q-BDHE assumption in the random oracle model. The proof will follow the outline of that in Section 4 except that Setup no longer outputs any h_x values and instead $\mathcal{B}$ must simulate the random oracle as follows. It should initialize an empty table T_{RO} at the beginning of the experiment. On each query for attribute x to the random oracle, $\mathcal{B}$ should first look to see if x is in T_{RO} and if so, return the value associated with it.

If x is not in T_{RO}, $\mathcal{B}$ creates a new table entry (x, i, h_x) for it as follows. Let i be the number of unique attributes queried to the random oracle (including x) at the time of this query. Let z_x be a random value in $\mathbb{Z}_p$. Then set

$$h_x := \begin{cases} g^{z_x} & \text{if } x \in S^*; \\ g^{z_x} g^{a^i} & \text{if } x \notin S^*. \end{cases}$$

An interesting question is whether one can achieve fast decryption for a large universe in the standard model. Lewko and Waters [16] gave a large universe construction for KP-ABE in the standard model. However, their technique requires that each attribute in the ciphertext have some "local" randomness associated with it. This does not work with our methods here which leverage the fact that there is only one random exponent that propagates through the ciphertext.

3.4 Short Ciphertext Realizations

In Section 3.1, we focused on optimizing decryption time. For some applications where bandwidth or storage space is a practical concern, one might prefer to optimize on ciphertext size. In the small universe construction, we can compress the ciphertext into only three group elements (as opposed to $2 + |S|$ group elements in Section 3.1) plus the description of the attribute set S. The main tradeoff is that private key sizes must scale by a factor of the size $|U|$ of the universe (compared to GPSW [13]), as opposed to scaling by only Γ as above.

We now sketch the main idea. During encryption, instead of including the set $\{C_x = h_x^s\}_{x \in S}$ in the ciphertext, it now includes the product of these values, the "aggregate" $\prod_{x \in S} C_x = f(S)^s$. Thus, the ciphertext contains three group elements of the form:

$$C = M \cdot e(g,g)^{\alpha s},\ \hat{C} = g^s,\ \prod_{x \in S} C_x = f(S)^s.$$

When generating the private key, replace Γ with U; that is, instead of only having "helper values" $Q_{i,x}$ for attributes x used in the access matrix, one now must include helpers for any attribute in the universe. This will allow the decryptor to handle an aggregate of any set of attributes. Thus, following the setup as before, the private keys are of the form:

$$\text{PK}, \quad (D_1 = g^{\lambda_1} \cdot h_{\rho(1)}^{r_1},\ R_1 = g^{r_1},\ \forall d \in U/\rho(1),\ Q_{1,d} = h_d^{r_1}),\ \ldots\ ,$$
$$(D_\ell = g^{\lambda_\ell} \cdot h_{\rho(\ell)}^{r_\ell},\ R_\ell = g^{r_\ell},\ \forall d \in U/\rho(\ell),\ Q_{\ell,d} = h_d^{r_\ell}).$$

The key size is proportional to $|U| \cdot \ell$, which is size of the attribute universe times the number of rows in the matrix. Due to the dependence on $|U|$, this aggregation unfortunately only works for small universes of attributes.

Finally, run the decryption algorithm as it is written, but understand that Δ will always be S due to the aggregate. This will increase the number of modular multiplications over Section 3.1, but not the number of pairings.

3.5 CP-ABE Variants

One might consider trying to apply these techniques in the CP-ABE setting [5,12] by analogously modifying an "unrestricted" CP-ABE construction, such as Waters [20, Section 3]. A natural analogy would arise in a CP-ABE system where the ciphertext size and encryption time blows up by a factor of X, where X is the number of distinct attributes used in the ciphertext access structure. In some applications, one might consider this to be a less palatable tradeoff. That is, one might not be willing to increase the transmission costs (i.e., ciphertext size) and encryption time, even if it meant faster decryption times.

We note, however, that if one is willing to consider "bounded" systems, where a value k_{max} can be set system-wide as the maximum number of times a single attribute can appear in a particular formula (or access structure), then one can achieve fast decryption without an increase in ciphertext size or encryption time. One such example is the CP-ABE construction of Waters [20, Section 5]. The critical part of the decryption algorithm appears in that paper as

$$e(C', K) / \left(\prod_{i \in I} (e(C_i, L) \cdot e(C', K_{\rho(i)}))^{\omega_i} \right)$$

which seems to require a non-constant $(2|I|+1)$ pairings. However, this equation is identical to:

$$e(\prod_{i \in I} C_i^{-\omega_i}, L) \cdot e(C', K \prod_{i \in I} K_{\rho(i)}^{-\omega_i})$$

which requires only two pairings. The observation that this bounded scheme offers fast decryption was previously made in its Charm [1] implementation.

4 Proof of the Base Construction

Theorem 1 (Security of the Small Universe KP-ABE). *The KP-ABE scheme Π in Section 3.1 for attribute universe U is selectively secure against chosen-plaintext attacks under the Decisional $|U|$-BDHE assumption in $\mathbb{G}$.*

Proof. Let U be an attribute universe, where $|U|$ is a polynomial in 1^λ. For notational convenience, we will assume each of the $|U|$ attributes is a unique integer between 1 and $|U|$.[2] Next suppose there exists a PPT adversary $\mathcal{A}$ that causes the selective, CPA security experiment $\mathsf{KP\text{-}ABE\text{-}Exp}^{\mathsf{sel\text{-}CPA}}_{\mathcal{A},\Pi}(\lambda, U)$ to output 1 with non-negligible probability. Then, we can construct a PPT adversary $\mathcal{B}$ that violates the Decisional $|U|$-BDHE assumption in $\mathbb{G}$ as follows:

Init: $\mathcal{A}$ outputs a set S^* of attributes for the challenge ciphertext.
Setup: $\mathcal{B}$ receives the Decisional $|U|$-BDHE challenge input

$$(\mathbb{G}, p, g, g^s, g^a, \ldots, g^{(a^{|U|})}, g^{(a^{|U|+2})}, \ldots, g^{(a^{2|U|})}, P)$$

for security parameter λ. It chooses random $\alpha', z_1, \ldots, z_{|U|} \in \mathbb{Z}_p$ and sets $e(g,g)^\alpha := e(g,g)^{\alpha'} \cdot e(g^a, g^{a^{|U|}})$ (implicitly defining α as $(\alpha' + a^{|U|+1})$) and

$$\text{for } x \in [1, |U|], \text{ sets } h_x := \begin{cases} g^{z_x} & \text{if } x \in S^*; \\ g^{z_x} g^{a^x} & \text{if } x \notin S^*. \end{cases}$$

It sets the public parameters PK as $(\mathbb{G}, p, g, e(g,g)^\alpha, h_1, \ldots, h_{|U|})$ and sends them to $\mathcal{A}$. Note that all parameters are well distributed due to the α' and z_x values.

Phase 1: $\mathcal{B}$ initializes an empty table T, an empty set D and a counter $j = 0$. $\mathcal{B}$ responds to $\mathcal{A}$'s queries as follows:

1. Create($\mathbb{A}$): $\mathcal{B}$ sets $j := j + 1$. It parses $\mathbb{A}$ as (W, ρ), where W is an $\ell \times n$ matrix. $\mathcal{B}$ will now work in two steps. First, it will create a valid private key, but not necessarily a well-distributed one. Then, it will re-randomize the key to ensure it is well distributed.

   Let K be the set of rows where the attributes are in S^* (i.e., for $i \in K, \rho(i) \in S^*$.) and K' be the rows where attributes are not in S^* (i.e., $K' = [1, \ell]/K$). Define an n dimensional vector $\boldsymbol{v}$ over $\mathbb{Z}_p$. Let $v_1 = 1$ (i.e., first element of $\boldsymbol{v}$ is 1) and for all $i \in K, \boldsymbol{v} \cdot W_i = 0$. (Here W_i is the n-dimensional vector that is row i of the matrix W.) To see that this is well-defined, consider that since S^* does not satisfy W, it must be the case that $(1, 0, 0..., 0)$ is *not* in the span of rows W_i for $i \in K$. It then follows from linear algebra that such a vector $\boldsymbol{v}$ exists.

   Next, $\mathcal{B}$ will make a private key for secret sharing with the vector $\alpha\boldsymbol{v}$ (i.e., the vector $\boldsymbol{v}$ with all the components scaled up by α.) This shares the secret

[2] A more proper notation would define an injective function $t()$ mapping attributes to integers from 1 to $|U|$ and then wherever we refer to an attribute x to instead refer to $t(x)$. However, this is more cumbersome.

α since $v_1 = 1$, although the key may not be well distributed. This will be addressed later through re-randomization. The shares λ_i for $i \in [1, \ell]$ are computed as $(\alpha \boldsymbol{v}) \cdot W_i$.

For $i \in K$, we have that all components are just the identity element (recall that the key is not re-randomized yet; we add the h_x components in shortly with the randomization): $D_i = R_i = H_{i,x} = g^0$ for $x \in \Gamma$ (recall Γ is the set of distinct attributes used in key generation). This follows because for all $i \in K, \lambda_i = 0$.

For $i \in K'$, we first compute $c_i = \boldsymbol{v} \cdot W_i$. Note that $\lambda_i = c_i \cdot \alpha = c_i \cdot (\alpha' + a^{|U|+1})$. To produce these key components, we need a cancellation technique. Set $R_i = g^{-c_i a^{(|U|+1)-\rho(i)}}$, which implicitly defines $r_i = -c_i \cdot a^{(|U|+1)-\rho(i)}$. This is computable from $\mathcal{B}$'s challenge input since $\rho(i)$ is between 1 and $|U|$. Set

$$\begin{aligned} D_i &= g^{c_i \alpha'} \cdot R_i^{z_i} \\ &= g^{c_i \alpha'} \cdot g^{-z_i c_i a^{(|U|+1)-\rho(i)}} \\ &= g^{c_i \alpha'} \cdot g^{c_i a^{|U|+1}} \cdot g^{-z_i c_i a^{(|U|+1)-\rho(i)}} \cdot g^{-c_i a^{|U|+1}} \\ &= g^{c_i \alpha} \cdot (g^{z_i})^{r_i} \cdot (g^{a^{\rho(i)}})^{r_i} \\ &= g^{c_i \alpha} \cdot h_{\rho(i)}^{r_i} \end{aligned}$$

Next, we turn to computing the helper values. For all $x \in \Gamma / \rho(i)$, set $Q_{i,x} = h_x^{r_i}$ by computing as follows:

$$h_x^{r_i} := \begin{cases} (g^{z_x})^{r_i} = g^{-z_x c_i a^{(|U|+1)-\rho(i)}} & \text{if } x \in S^*; \\ (g^{z_x})^{r_i} \cdot (g^{a^x})^{r_i} = g^{-z_x c_i a^{(|U|+1)-\rho(i)}} \cdot g^{-c_i a^{(|U|+1)-\rho(i)+x}} & \text{if } x \notin S^*. \end{cases}$$

This last part is computable since $\rho(i) \neq x$ for the helper values and recall that the z_x values were chosen by $\mathcal{B}$ during Setup.

At this point, $\mathcal{B}$ has constructed the components of a valid private key. Next, we give a public-key re-randomization algorithm that can be applied by $\mathcal{B}$ to *any* valid private key, before it sends the key to $\mathcal{A}$. To re-randomize, choose random $y_2, ..., y_n \in \mathbb{Z}_p$. Consider the vector $(0, y_2, y_3, ..., y_n)$. This will be used to secret share 0 to re-randomize the key. Let $\lambda_i' = (0, y_2, y_3, ..., y_n) \cdot W_i$ for $i \in [1, \ell]$.

For all $i \in [1, \ell]$, the first step to re-randomization is to let $D_i^{\#} := D_i \cdot g^{\lambda_i'}$. The next step is to re-randomize all the r_i values, which are used in all key components. To do this, choose a fresh $r_i' \in \mathbb{Z}_p$ and set

$$D_i' := D_i^{\#} \cdot h_{\rho(i)}^{r_i'} \; R_i' := R_i \cdot g^{r_i'} \; Q_{i,x}' := Q_{i,x} \cdot h_x^{r_i'}, \forall x \in \Gamma / \rho(i)$$

We claim that the above re-randomization procedure correctly re-randomizes any "valid" key. A valid key is one which is generated from some sharing of α, but not necessarily a well distributed one. The above algorithm propagates new random values $r_1', \ldots, r_\ell'$ completely through all key components, and then also generates a fresh secret sharing for α used in $D_1', \ldots, D_\ell'$.

This properly redistributes the only variable parts of the key. That is the distribution after applying this transformation to any valid key with policy (W, ρ) has the same distribution as a fresh key generated by running KeyGen for (W, ρ).

2. Corrupt(i): If there exists an ith entry in table T, then $\mathcal{B}$ obtains the entry $(i, \mathbb{A}, \mathrm{SK})$ and sets $D := D \cup \{\mathbb{A}\}$. It then sends SK to $\mathcal{A}$. If no such entry exists, then it returns $\perp$.

Challenge: $\mathcal{A}$ outputs two messages M_0, M_1 and $\mathcal{B}$ chooses a random bit b. $\mathcal{B}$ then constructs and sends to $\mathcal{A}$ the challenge ciphertext

$$\mathrm{CT}^* = (C^* := M_b \cdot P,\ C' := g^s,\ \forall x \in S^*, C_x^* := (g^s)^{z_x}).$$

Phase 2: $\mathcal{B}$ responds to $\mathcal{A}$'s queries in the same manner as in Phase 1, except that it refuses to answer any Corrupt query that would result in an access structure $\mathbb{A}$ which S^* satisfies being added to D.

Guess: Eventually, $\mathcal{A}$ outputs a bit b'. If $b = b'$, then $\mathcal{B}$ outputs 0 (guessing that $P = e(g, g)^{a^{|U|+1}s}$), else it outputs 1 (guessing that P is random.)

Thus, $\mathcal{B}$'s responses to $\mathcal{A}$ are distributed identically as in the $\mathsf{KP\text{-}ABE\text{-}Exp}^{\mathsf{sel\text{-}CPA}}_{\mathcal{A},\Pi}(\lambda, U)$ experiment. Whenever $\mathcal{A}$ causes the output of this experiment to be 1, $\mathcal{B}$ will also correctly answer its Decisional BDHE challenge.

5 Exploring a Spectrum of Efficiency Tradeoffs

We now focus on the tradeoff between private key size and decryption time. We generalize the construction ideas of the last section to give a spectrum of possible "unbounded" schemes, where GPSW is one extreme and Section 3 (longer keys, faster decryption) is the other. We do this in two steps. We first present a generalized decryption algorithm. We then show how the size of the private key scales depending on how one chooses to take advantage of this generalized decryption algorithm. We conclude with strategies for keeping both key size and decryption time low.

5.1 A Generalized Decryption Algorithm

To begin, we present a generalized decryption algorithm for the GPSW ciphertexts (which are the same as the encryption algorithm presented in Section 3.1). The main idea is to break Δ (the set of distinct attributes associated with the rows of the access matrix used in one chosen way to decrypt the ciphertext) into y disjoint subsets $\Delta_1, \Delta_2, \ldots, \Delta_y$. Recall that we defined the function f as

$$f(\Delta) = \prod_{x \in \Delta} h_x \in \mathbb{G}.$$

Further, let us establish the notation for a function w that (informally) takes in a attribute and outputs which set the attribute is in. More formally, let $w : \Delta \to [1, y]$ be a map such that $w(x) = j$ if and only if $x \in \Delta_j$.

The decryption algorithm then proceeds as follows. For each $i \in I$, it will compute the value

$$\hat{D}_i = D_i \cdot \prod_{x \in \Delta_{w(\rho(i))}/\rho(i)} Q_{i,x} = g^{\lambda_i} f(\Delta_{w(\rho(i))})^{r_i}.$$

Next, for each $j \in [1, y]$, it will compute the value

$$L_j = \prod_{x \in \Delta_j} C_x = \prod_{x \in \Delta_j} h_x^s = f(\Delta_j)^s.$$

The algorithm now recovers the value $e(g,g)^{\alpha s}$ by computing

$$e(\hat{C}, \prod_{i \in I} \hat{D}_i^{\omega_i}) / \left(\prod_{j=1}^{y} e(\prod_{i:\rho(i) \in \Delta_i} R_i^{\omega_i}, L_j) \right).$$

The decryption algorithm can then divide out this value from C and obtain the message M.

We observe that this will take $(1 + y)$ pairings and roughly $|\Delta_1|^2 + |\Delta_2|^2 + \cdots + |\Delta_y|^2$ modular multiplications. (Recall that the ω_i values will be either 0 or 1 when the access structure is a boolean formula, so no exponentiations come into play in this case.) Thus, when $y = 1$, we have the scheme from Section 3.1 (which can be trivially extended to large universes in the random oracle model as shown in Section 3.3) and when $y = |\Delta|$, this corresponds to GPSW.

5.2 Reducing Private Key Overhead

At this point, the private key still contains "helper" values such that each of the ℓ rows has helpers for all other attributes in Δ. However, we can reduce the size of the private key by eliminating all helper values $Q_{i,x}$ where where attribute $\rho(i) \in \Delta_d$, attribute $x \in \Delta_{d'}$ and $d \neq d'$, i.e., where the two attributes were separated into distinct subsets. This is because only helpers *within* subsets will be used in the generalized decryption algorithm. We note that the security of the base system trivially implies security of this system, since for each private key in this reduced setting the components given out are a strict subset of the private key components in the base system.

5.3 Different Tradeoff Strategies

We now discuss how one might take advantage of the generalized algorithm and corresponding private key reduction. The choice of y and the subsets $\Delta_1, \ldots, \Delta_y$ is critical to the decryption *performance of a particular user*. Roughly, one expects decryption time to increase with y, but the size of the private key to decrease as the size of each Δ_i decreases. However, a nice feature of this approach is that each user can tune their own performance based on how they think they are likely to use their private key. A user might choose to retain her entire private key on her PC, but upload only a portion of her private key to her mobile device (where secure storage may be more limited.)

1. *Group attributes by expected ciphertext attributes.* Group together any attributes that are likely to appear together in a ciphertext, such as attributes relating to a certain work project, activity, role or time period. For instance, one might group together the attributes "cryptography", "encryption", and "pairings" and form a distinct group for the attributes "audubon", "peregrine falcon", "glaucous gull". Of course, the subsets of Δ need not be distinct[3] and it could be more efficient to place an attribute into two or more groups, but this should be done with care or the decryption time will increase without reducing the private key size.
2. *Group attributes by observing the private key.* It may be possible to deduce from the access structure which attributes are likely to be used together during decryption. For instance, suppose the structure is a formula and the only time that attributes A and B appear, they appear as "A AND B". Then clearly one should place A and B into the same group Δ_i.
3. *Break into y equal sized subsets of attributes.* One could also try the simple approach of choosing a y and randomly creating y equal-sized subsets. In many practical applications, the average overhead incurred on future ciphertexts would be dependent on the overhead from past ciphertexts, so one could try a random setting and then observe performance.
4. *Benchmark and set experimentally.* One can also imagine starting with any combination of the three above techniques and then applying machine learning tools to evolve to a good balance point for any particular user.

We leave an evaluation of these strategies and their performance as an interesting open problem.

Acknowledgments. The authors thank Matthew Green for helpful input and discussions and the anonymous reviewers for helpful comments. This work was performed while the authors were at Zeutro, LLC.

References

1. Ayo Akinyele, J., Belvin, G., Garman, C., Pagano, M., Rushanan, M., Martin, P., Miers, I., Green, M., Rubin, A.: Charm: A tool for rapid cryptographic prototyping (2012), `http://www.charm-crypto.com/`
2. Attrapadung, N., Herranz, J., Laguillaumie, F., Libert, B., de Panafieu, E., Ràfols, C.: Attribute-based encryption schemes with constant-size ciphertexts. Theor. Comput. Sci. 422, 15–38 (2012)
3. 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)
4. Beimel, A.: Secure Schemes for Secret Sharing and Key Distribution. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel (1996)

[3] If the subsets are not distinct, then one needs to generalize the function w to output a subset of indices.

5. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334 (2007)
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)
7. 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)
8. 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)
9. Canetti, R., Halevi, S., Katz, J.: A Forward-Secure Public-Key Encryption Scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 255–271. Springer, Heidelberg (2003)
10. 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)
11. Gentry, C., Silverberg, A.: Hierarchical ID-Based Cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)
12. 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., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 579–591. Springer, Heidelberg (2008)
13. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
14. Horwitz, J., Lynn, B.: Toward Hierarchical Identity-Based Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 466–481. Springer, Heidelberg (2002)
15. 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)
16. Lewko, A., Waters, B.: Unbounded HIBE and Attribute-Based Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 547–567. Springer, Heidelberg (2011)
17. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
18. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
19. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
20. Waters, B.: 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)

A Access Structures and Notation

A.1 Access Structures

Definition 1 (Access Structure [4]) *Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An* access structure *(respectively, monotone access structure) is a collection (resp., monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the* authorized sets*, and the sets not in $\mathbb{A}$ are called the* unauthorized sets*.*

In our context, the role of the parties is taken by the attributes. Thus, the access structure $\mathbb{A}$ will contain the authorized sets of attributes. We restrict our attention to monotone access structures. However, it is also possible to (inefficiently) realize general access structures using our techniques by defining the "not" of an attribute as a separate attribute altogether. Thus, the number of attributes in the system will be doubled. From now on, unless stated otherwise, by an access structure we mean a monotone access structure.

A.2 Linear Secret Sharing Schemes

The construction will use linear secret sharing schemes, as slightly adapted from Beimel [4]:

Definition 2 (Linear Secret-Sharing Schemes (LSSS)) *A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called* linear *(over $\mathbb{Z}_p$) if*

1. *The shares of the parties form a vector over $\mathbb{Z}_p$.*
2. *There exists a matrix M with ℓ rows and n columns called the share-generating matrix for Π. There exists a function ρ which maps each row of the matrix to an associated party. That is for $i = 1, \ldots, \ell$, the value $\rho(i)$ is the party associated with row i. When we consider the column vector $v = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen, then Mv is the vector of ℓ shares of the secret s according to Π. The share $(Mv)_i$ belongs to party $\rho(i)$.*

It is shown in [4] that every linear secret sharing-scheme according to the above definition also enjoys the *linear reconstruction* property, defined as follows: Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subseteq \{1, 2, \ldots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \omega_i \lambda_i = s$. It is shown in [4] that these constants $\{\omega_i\}$ can be found in time polynomial in the size of the share-generating matrix M.

Like any secret sharing scheme, it has the property that for any unauthorized set $S \notin \mathbb{A}$, the secret s should be information theoretically hidden from the parties in S.

Note on Convention. We use the convention that vector $(1, 0, 0, \ldots, 0)$ is the "target" vector for any linear secret sharing scheme. For any satisfying set of rows I in M, we will have that the target vector is in the span of I.

For any unauthorized set of rows I the target vector is not in the span of the rows of the set I. Moreover, there will exist a vector w such that $w \cdot (1, 0, 0 \ldots, 0) = -1$ and $w \cdot M_i = 0$ for all $i \in I$.

Using Access Trees. Some prior ABE works (e.g., [13]) described access formulas in terms of binary trees. Using standard techniques [4] one can convert any monotonic boolean formula into an LSSS representation. An access tree of ℓ nodes will result in an LSSS matrix of ℓ rows.

Recovering RSA Secret Keys from Noisy Key Bits with Erasures and Errors

Noboru Kunihiro[1], Naoyuki Shinohara[2], and Tetsuya Izu[3]

[1] The University of Tokyo, Japan
kunihiro@k.u-tokyo.ac.jp
[2] NICT, Japan
[3] Fujitsu Labs, Japan

Abstract. We discuss how to recover RSA secret keys from noisy key bits with erasures and errors. There are two known algorithms recovering original secret keys from noisy keys. At Crypto 2009, Heninger and Shacham proposed a method for the case where an erroneous version of secret keys contains only erasures. Subsequently, Henecka et al. proposed a method for an erroneous version containing only errors at Crypto 2010. For physical attacks such as side-channel and cold boot attacks, we need to study key recovery from a noisy secret key containing both erasures and errors. In this paper, we propose a method to recover a secret key from such an erroneous version and analyze the condition for error and erasure rates so that our algorithm succeeds in finding the correct secret key in polynomial time. We also evaluate a theoretical bound to recover the secret key and discuss to what extent our algorithm achieves this bound.

Keywords: RSA, Key-recovery, Cold Boot Attack, Side-channel Attack, Maximal Likelihood.

1 Introduction

1.1 Background

RSA [12] is a widely used cryptosystem. In RSA a public modulus N is chosen to be a product of two distinct primes p and q. The key-pair $e, d \in \mathbb{Z}$ satisfies $ed \equiv 1 \pmod{(p-1)(q-1)}$. The encryption keys are (N, e) and the decryption keys are (N, d). The PKCS#1 standard [10] specifies that the RSA secret key includes the following information: $(p, q, d, d_p, d_q, q^{-1} \bmod p)$ in addition to d, which allows a fast decryption process using the Chinese Remainder Theorem.

Secret keys must be kept secret. Nevertheless, some fractional amounts of the secret information can be leaked by physical attacks such as side-channel and cold boot attacks [4]. If the amount of leaked bits for secret keys is quite small, it is impossible to recover the secret keys from the leaked information. Conversely, it might be possible to recover them by using their redundancy if a certain amount of bits are leaked. Note that all bits are not necessarily leaked. For example, Coppersmith [2] showed that RSA can be broken if the upper half

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 180–197, 2013.

of the secret key p is revealed. Herrmann and May [7] showed that RSA can be broken (in exponential time) if at least 70% of the bits for a prime factor p of N are leaked. Their methods are based on the lattice reduction technique. Note that the Herrmann-May method does not require that the leaked bits are consecutive.

At Crypto 2009, Heninger and Shacham [6] proposed an algorithm that efficiently recovers secret keys (p, q, d, d_p, d_q) given a random fraction of their bits. Concretely, they showed that if at least 27% of the secret key bits are leaked at random, the full secret keys can be recovered. Conversely, we can say that even if 73% of original secret bits are erased, the key recovery succeeds.

As opposed to the Heninger-Shacham algorithm correcting erasures, Henecka et al. [5] proposed an algorithm correcting error bits of secret keys at Crypto 2010. They showed that the secret key (p, q, d, d_p, d_q) can be fully recovered if the error probability is less than 0.237. They also showed the bound for the error probability is given by 0.084 if the involved secret key is (p, q).

Independently of our work, Paterson et al. proposed an algorithm correcting error bits which asymmetrically occurs at Asiacrypt 2012 [9]. Their algorithm works in a true cold boot setting. They took a coding theoretic approach for designing a new algorithm and analyzing its performance.

1.2 Motivation: Attack Scenario

All existing works concerning key recovery from noisy secret keys have discussed the erasure-only (error-free) case or error-only (erasure-free) case. This paper deals with the key recovery for a noisy secret key with both erasures and errors. We call the erroneous version of the secret key with both erasures and errors *noisy secret keys*. We denote the correct secret key by $\mathbf{sk}$, and the noisy secret key corresponding to $\mathbf{sk}$ by $\overline{\mathbf{sk}}$. Before discussing the details, we address the motivations of this study.

Cold Boot Attack Scenario: Under the cold boot attack scenario [4], (the degraded version of) secret keys are observed with (almost) unidirectional bit flipping. Assume that the flip of each bit occurs as completely unidirectional. For simplicity, we assume that only the bit flipping of $1 \to 0$ occurs. If the observed bit is 1, the corresponding bit of the correct secret key is definitely 1. In contract, if the observed bit is 0, we cannot determine whether the corresponding bit is 0 or 1. Therefore, the observed bit 0 can be considered *erasure*. Heninger and Shacham [6] proposed an efficient algorithm that recovers the secret key from the degraded version of the secret key with erasure. However, as Heninger and Shacham [6] pointed out, the bit flip with an opposite direction occurs with small but non-zero probability. If the observed bit sequence contains errors, Heninger-Shacham's algorithm can never recover the correct secret key. This algorithm is then no longer applicable for the noisy secret key containing both erasures and errors.

Side-channel Attack Scenario: Henecka et al. [5] proposed an efficient algorithm given a noisy secret key only with errors. The noisy keys are often

provided through a side-channel attack. Under some attack situations, each bit is provided with additional information: so-called *reliability*. Consider the following situation: some bits of secret keys are 0 (or 1) with very high reliability and others are 0 (or 1) with not so high reliability. One reasonable strategy is to set a bit value as the observed bit if its reliability is sufficiently high. How should we set a bit value with low reliability? We have two potential strategies. The first is to set a bit value as the observed bit, which will cause a high number of bit errors. The second strategy is to regard the bit as an *erasure* bit, which will involve the observed secret key with (fewer) errors and erasures. So then which of strategies is good for attackers? As Henecka et al. pointed out, the correction of errors seems to be a much more difficult problem than the correction problem. We therefore expect that the second strategy leads to a better algorithm. However, their algorithm is not applicable to a noisy secret key containing both erasures and errors.

For both cases, studies for the key recovery for noisy secret keys with both errors and erasures are important to maximize and evaluate the potential threat of physical attacks and to consider the possible countermeasures against them.

1.3 Our Contributions

This paper discusses secret key recovery from noisy secret key sequences with both errors and erasures. First, we present a polynomial time algorithm for recovering secret keys and show an explicit success condition for recovering the keys. We denote the erasure probability by δ and error probability by ϵ. We also denote by m the number of involved secret keys. For example, $m = 5$ if $\mathbf{sk} = (p, q, d, d_p, d_q)$ is involved. Our algorithm can asymptotically recover secret keys in polynomial time with high probability provided that

$$1 - \delta - 2\epsilon \geq \sqrt{\frac{2(1-\delta)\ln 2}{m}},$$

where we denote the natural logarithm of n to the base e by $\ln n$. In special case, our algorithm also includes previous methods. In fact, our algorithm achieves the upper bound of Heninger-Shacham [6] and that of Henecka et al. [5] for the error-free case ($\epsilon = 0$) and erasure-free case ($\delta = 0$), respectively. We ran experiments to verify our analysis. We achieved to the error rates of up to 0.6 and the erasure rate $\epsilon = 0.01$ for 1024-bit RSA with high success probability.

Second, we derive a theoretical bound for recovering the secret keys from the noisy secret keys. We first introduce a natural abstract algorithm (`meta-algorithm`) and derive a condition for δ and ϵ such that it needs exponential time for recovering keys. The binary Entropy function $H(x)$ [3] is defined by by $-x \log x - (1-x)\log(1-x)$, where $\log n$ is the binary logarithm of n to the base 2. Then, we prove that we cannot recover the secret keys in polynomial time under our `meta-algorithm` if it holds that

$$(1-\delta)\left(1 - H\left(\frac{\epsilon}{1-\delta}\right)\right) < \frac{1}{m}.$$

Finally, we discuss the relation between the condition where our algorithm can recover secret keys and the theoretical bound. We first see that there exists a small gap between the success condition and the theoretical bound. Then, we show that the proposed algorithm achieves the second order expansion of the theoretical bound.

2 Preliminaries

This section presents an overview of methods using binary trees to recover the secret key of the RSA cryptosystem [12]. In particular, we briefly explain two known methods: Heninger-Shacham method [6] and the method of Henecka et al. (abbreviated to HMM method) [5].

We use similar notations as [5]. For an n-bit sequence $\mathbf{x} = (x_{n-1}, \ldots, x_0) \in \{0,1\}^n$, we denote the i-th bit of $\mathbf{x}$ by $x[i] = x_i$, where $x[0]$ is the least significant bit of $\mathbf{x}$. Let $\tau(M)$ denote the largest exponent such that $2^{\tau(M)} | M$. As well as [5], Hoeffding's bound [8] is the main tool in our analysis.

Theorem 1 (Hoeffding's Bound). *Let $X_1, \ldots, X_k$ be a sequence of independent Bernoulli trials with identical success probability $\Pr[X_i = 1] = p$ for all i. Define $X := \sum_{i=1}^{k} X_i$. Then, for every $0 < \gamma < 1$ we have $\Pr[X \geq k(p+\gamma)] \leq \exp(-2k\gamma^2)$ and $\Pr[X \leq k(p-\gamma)] \leq \exp(-2k\gamma^2)$.*

2.1 Noise Models

We formalize (three) noise models discussed in this paper. Let ϵ and δ be real numbers satisfying $0 \leq \epsilon < 1/2$, $0 \leq \delta < 1$ and $0 \leq \epsilon + \delta < 1$. In our noise models, each bit in a secret bit sequence is either erased with probability δ or flipped with probability ϵ, or remains unchanged with probability $1-\delta-\epsilon$. Then, only the transformed sequence is observed. Nevertheless, the original sequence is not directly obtained. We refer to this noise model as the Binary Erasure-Error model (BEE model). The error-free model, that is $\epsilon = 0$ (but, $\delta > 0$), is referred to as the Binary Erasure model (BE model) and erasure-free model, that is $\delta = 0$ (but, $\epsilon > 0$), is referred to as the Binary Symmetric model (BS model).

Our target in this paper is to recover the original secret key from the observed noisy sequence. We can say that Heninger and Shacham [6] have studied key recovery from the noisy keys in the BE model and Henecka et al. [5] have studied it in the BS model.

2.2 Recovering RSA Secret Key by Using Binary Trees

First, we review the key setting of the RSA cryptosystem [12], especially of the PKCS #1 standard [10]. The public key is (N, e) and the secret key is $\mathbf{sk} = (p, q, d, d_p, d_q, q^{-1} \bmod p)$. As in the previous works, we also ignore the last component $q^{-1} \bmod p$ in the secret key. The public and secret keys have the following relations:

$$N = pq,\ ed \equiv 1 \pmod{(p-1)(q-1)},\ ed_p \equiv 1 \pmod{p-1},\ ed_q \equiv 1 \pmod{q-1}.$$

From the key setting, there exist some integers k, k_p, k_q such that

$$N = pq, ed = 1 + k(p-1)(q-1), ed_p = 1 + k_p(p-1), ed_q = 1 + k_q(q-1). \quad (1)$$

Suppose that we know the exact values of k, k_p, and k_q. There exist five unknowns (p, q, d, d_p, d_q) in Eq. (1). Then, if we know just one of the exact values of unknowns, we can easily obtain the others.

The small public exponent e is usually used in practical applications [13], so we suppose that e is small enough such that $e = 2^{16} + 1$ in the same manner as [5,6]. We need to find k, k_p, and k_q for small e. See the full version for how to compute k, k_p and k_q in our method.

In the Heninger-Shacham method [6], HMM method [5] and our new method, a secret key $\mathbf{sk}$ is recovered by using a binary-tree-based technique. Here we explain how to recover secret keys, taking $\mathbf{sk} = (p, q, d, d_p, d_q)$ as an example.

First we mention generating the tree. Since p and q are $n/2$ bit prime numbers and half of the most significant bit (MSB) of d is efficiently computable in the Heninger-Shacham or HMM methods [5,6], there exist at most $2^{n/2}$ candidates for each secret key in $\{p, q, d, d_p, d_q\}$.

Heninger and Shacham [6] define the i-th bit slice for each bit index i and we denote by

$$\mathbf{slice}(i) := (p[i], q[i], d[i + \tau(k)], d_p[i + \tau(k_p)], d_q[i + \tau(k_q)]).$$

Assume that we have computed a partial solution $\mathbf{sk}' = (p', q', d', d'_p, d'_q)$ up to $\mathbf{slice}(i-1)$. Heninger and Shacham [6] applied Hensel's lemma to Eq. (1) and presented the following equations

$$p[i] + q[i] = (N - p'q')[i] \bmod 2, \quad (2)$$

$$d[i + \tau(k)] + p[i] + q[i] = (k(N+1) + 1 - k(p' + q') - ed')[i + \tau(k)] \bmod 2, \quad (3)$$

$$d_p[i + \tau(k_p)] + p[i] = (k_p(p' - 1) + 1 - ed'_p)[i + \tau(k_p)] \bmod 2, \quad (4)$$

$$d_q[i + \tau(k_q)] + q[i] = ((k_q(q' - 1) + 1 - ed'_q)[i + \tau(k_q)] \bmod 2. \quad (5)$$

We can easily see that $p[i], q[i], d[i + \tau(k)], d_p[i + \tau(k_p)]$, and $d_q[i + \tau(k_q)]$ are not independent and the degree of freedom is 1. Therefore, each Hensel lift yields exactly two candidate solutions. Then, the number of all candidates is given by $2^{n/2}$. The root node is given by $\mathbf{slice}(0) = (1, 1, d[\tau(k)], d_p[\tau(k_p)], d_q[\tau(k_q)])$.

Next we explain a pruning step, in which we count the number of matching bits between a bit sequence given by a node sequence and the corresponding bit sequence of a noisy secret key. We then discard or leave each node according to given criteria.

Section 2.3 briefly overviews the Heninger-Shacham method, which is for the case of an erroneous version $\overline{\mathbf{sk}}$ with an erasure rate δ of $\mathbf{sk}$. And in section 2.4 we mention the HMM method for an erroneous version $\overline{\mathbf{sk}}$ with error rate ϵ.

2.3 Heninger-Shacham Method [6]

In the Heninger-Shacham method, a binary tree is constructed by iterating an *expansion phase* and a *pruning phase*. At the pruning step, we compare a bit sequence given by one node with the corresponding bit sequence given by $\overline{\mathbf{sk}}$. Then we discard a node containing a bit not matching with the corresponding bit of $\overline{\mathbf{sk}}$, skipping the bit corresponding to an erasure bit of $\overline{\mathbf{sk}}$.

In the Heninger-Shacham method, discarded nodes are exactly wrong nodes, so the node corresponding to the correct solution consistently remains. Therefore, the success probability of the Heninger-Shacham method is 1. The computational cost of Heninger-Shacham method is evaluated with the number of remaining nodes of the binary tree, and depends on erasure rate δ. Therefore, Heninger and Shacham estimated the upper bound of δ such that the expected number of remaining nodes yielded from one wrong node is less than 1 under the following assumption:

Assumption 1. *The bit slice corresponding to a wrong node consists of random bits.*

This assumption is also used in the analysis of [5] and our new method.

Heninger and Shacham showed that their method recovers the secret keys provided that $\delta \leq 0.73$ if the noisy secret key is of the form (p, q, d, d_p, d_q), namely $m = 5$. If we use the noisy secret information (p, q), the secret key (p, q) can be obtained provided that $\delta \leq 0.43$. For general m of the involved secret information, the secret key can be recovered provided that $\delta \leq 2^{\frac{m-1}{m}} - 1$. We can see that the right-hand side of the above inequality is approximated by $1 - \frac{2 \ln 2}{m}$ for large m.

The Heninger-Shacham algorithm requires that the non-erasure bit is correct. However, this requirement is too idealistic in the physical attacks such as a cold boot attack, as described in Section 1.2. If the observed secret key contains an error, the Heninger-Shacham algorithm never finds the correct secret keys. We provide a simple example. Assuming that $\epsilon = 0.001$, we can regard the error rate as extremely small. Nevertheless, the number of errors in secret keys is expected to be $512 \times 5 \times 0.001 = 2.56 (> 2)$. Due to there being only two errors, the Heninger-Shacham algorithm does not work.

2.4 Henecka-May-Meurer Method [5]

We briefly explain the HMM method. For an erroneous version $\overline{\mathbf{sk}}$ with error rate ϵ, if we discard every node having a bit not matching the corresponding bit in $\overline{\mathbf{sk}}$, $\mathbf{sk}$ is never recovered since the leaf node corresponding to the correct solution does not remain. Therefore, the binary tree is separated into partial trees whose depth is t, and then the pruning step is performed for each partial tree. Actually, mt bits of the node sequence from the root node of the partial tree to the leaf node of the partial tree are compared with the corresponding bit of $\overline{\mathbf{sk}}$. If the number of matches is less than $C \in [0, mt]$, the leaf node is discarded. Since the remaining nodes of the binary tree decrease if the threshold value C increases, the computational cost decreases and the success probability decreases.

Especially the pruning step is not practically performed when $C = 0$, and we never obtain **sk** when $C = mt$. Henecka et al. considered the two following restrictions, which help to decide how to choose parameters (t, C). Note that $\mathbb{E}[X]$ is the mean of a random variable X.

Restriction 1. *Let $Z_{b,i}$ be the number of bad candidates generated from one bad partial solution at the i-th pruning step. Then, we choose parameters (t, C) so that $\mathbb{E}[Z_{b,i}] \leq 1/2$ holds.*

Restriction 2. *For each pruning step, we choose parameters (t, C) so that the probability that the correct node is discarded is less than $1/n$.*

The HMM method recovers the secret keys (p, q, d, d_p, d_q) if the error rate ϵ of the noisy keys is not larger than 0.237. If we use the noisy secret information or (p, q), the secret key (p, q) can be obtained, provided that $\epsilon \leq 0.084$. For general m of the involved secret information, the secret key can be recovered provided that $\epsilon \leq 1/2 - \sqrt{\ln 2/2m}$.

2.5 Naive Method Based on HMM Method

As mentioned, our main purpose is to recover secret keys from the noisy keys with both erasures and errors (that is, obtained through the BEE model). The following naive algorithm, which is not described in the literature, is sufficient for merely achieving this purpose.

Naive Method

Input: Public key (N, e), observed secret key $\overline{\mathbf{sk}}$, erasure probability δ and error probability ϵ

Output: Correct secret key **sk**

Step 1: Transform $\overline{\mathbf{sk}}$ to $\overline{\mathbf{sk}}'$ by substituting random bits into erasure positions of $\overline{\mathbf{sk}}$.

Step 2: Perform the HMM method with the sequence $\overline{\mathbf{sk}}'$ and the error probability $\epsilon + \frac{\delta}{2}$ as inputs.

We evaluate the success condition of the algorithm. Each erasure bit will change a correct bit with probability $1/2$ and a wrong bit with $1/2$. The secret key sequence transformed in Step 1 can be considered a sequence with erasure probability 0 and error probability $\epsilon + \frac{\delta}{2}$. By applying the success condition for the BS model, we have the following condition for the naive method:

$$\epsilon + \delta/2 \leq 1/2 - \sqrt{\ln 2/(2m)}. \tag{6}$$

Although the algorithm does work for the noisy secret key for the BEE model, the above algorithm is not better than expected. There are some drawbacks to the naive method. Assuming that $\epsilon = 0$, the condition is described as $\delta \leq 1 - \sqrt{2 \ln 2/m}$. This condition is clearly worse than that of Heninger-Shacham:

$\delta \leq 1 - \frac{2\ln 2}{m}$. Next, we discuss the case where the error probability ϵ is very small but not zero, which is a natural situation in the cold boot attack scenario. For example, we assume that $m = 5$, $\delta = 0.6$ and $\epsilon = 0.001$. Considering that the Heninger-Shacham algorithm works well if $\delta = 0.73$ and $\epsilon = 0$, it is natural that we expect that the key recovery succeeds if $\delta = 0.6, \epsilon = 0.001$. However, the condition that $\delta = 0.6$ and $\epsilon = 0.001$ does not satisfy Eq. (6), and the naive method then cannot recover the secret key if $\delta = 0.6$ and $\epsilon = 0.001$. Our main goal in this paper is to propose a method that works in that case.

3 Recovering Secret Key from Noisy Secret Keys in BEE Model

Let $\overline{\mathbf{sk}}$ be an erroneous version of a secret key $\mathbf{sk}$ with erasure rate δ and error rate ϵ. The main purpose of our algorithm is to recover the original secret key from the observed $\overline{\mathbf{sk}}$ with the help of redundancy. We propose an algorithm to recover $\mathbf{sk}$ from $\overline{\mathbf{sk}}$ by using the binary-tree-based technique as in the Heninger-Shacham method [6] and HMM method [5]. Our algorithm is a combination of the two methods.

In our algorithm, the binary tree is separated into partial trees, and the pruning step is executed for every partial tree with threshold values as with the HMM method. Analysis of our algorithms then requires Assumption 1, Restrictions 1 and 2 in the same manner as with the HMM method.

Lesson Learned from Failure of Naive Method. In the naive method described in section 2.5, we transform the erasure bit to the error bit with probability 1/2. This worsens the success condition. Any erasure bit should then be handled as erasure not error.

3.1 Our Proposed Method

In the HMM method [5], the noisy secret key sequence $\overline{\mathbf{sk}}$ is divided in an mt-bit subsequence to construct a partial tree, where t is a fixed integer. On the other hand, in our new method we divide the sequence in a T-bit subsequence skipping erasure bits in $\overline{\mathbf{sk}}$. We show a small example for $m = 3$ and $T = 4$. First, we explain how to divide bits for the i-th pruning step. Let E be the error symbol in $\overline{\mathbf{sk}}$. Suppose that we have divided bits until the bit p_s in the s-th node $[p_s, q_s, E]$ at the $(i-1)$-th pruning step, and the following nodes are given: $[p_s, q_s, E]$, $[p_{s+1}, E, d_{s+1}], [p_{s+2}, q_{s+2}, d_{s+2}]$. Then, since the i-th pruning step will be performed for T bits skipping bits corresponding to E in $\overline{\mathbf{sk}}$, we check the bits corresponding to $q_s, p_{s+1}, d_{s+1}, p_{s+2}$. Here we denote by t_i the length of a node sequence that is newly generated for the i-th pruning step, and denote by Δ_i the number of E in $\overline{\mathbf{sk}}$ at the i-th pruning step. In the example, $t_i = 2$ and $\Delta_i = 2$. Since the condition $T \geq m$ practically holds, we have that

$$t_i = \lceil (T + \Delta_i)/m \rceil \text{ or } \lceil (T + \Delta_i)/m \rceil - 1. \tag{7}$$

In the HMM method, only one threshold value C is used. In contrast, we use threshold values $C_1, \ldots, C_\ell$ when $\overline{\mathbf{sk}}$ is separated into ℓ intervals. Theorem 2 in Section 3.2 provides how to set each C_i. Note that unknown values of k, k_p and k_q are efficiently computable from $\overline{\mathbf{sk}}$. We show the details of how to compute them in the full version.

New method

Input: Public key (N, e), noisy secret key $\overline{\mathbf{sk}}$, error probability ϵ and erasure probability δ

Output: Correct secret key $\mathbf{sk}$.

Step 1: Compute k, k_p, k_q and $\mathbf{slice}(0)$.

Step 2: Compute $(T, C_1, \ldots, C_\ell)$.

Step 3: From $i = 1$ to ℓ, perform the following computation. Set $t_0 = 0$: Compute t_i slices: $\mathbf{slice}(1 + \sum_{j=0}^{i-1} t_j), \mathbf{slice}(2 + \sum_{j=0}^{i-1} t_j), \ldots, \mathbf{slice}(\sum_{j=0}^{i} t_j)$ and generate a partial tree whose depth is $t_i + 1$. For T bits skipping erasure bits $\overline{\mathbf{sk}}$, count the number of matches of bits in partial solutions with the corresponding bits in $\overline{\mathbf{sk}}$. If it is not less than C_i, then set $i = i + 1$ and go to the generating of a partial tree step. Otherwise, discard the node.

Step4: For each remaining leaf node, check whether the nodes are indeed the valid secret key with the help of public information.

Remark 1. Suppose that $\epsilon = 0$. Our method for $T = C = 1$ is equivalent to the Heninger-Shacham method [6]. Suppose that $\delta = 0$. Our method with $(T, C, C, \ldots, C)$ is equivalent to the HMM method [5] with $(T/m, C)$. Our method includes both of the two methods.

3.2 Analysis of Our Proposed Method

This section provides the analysis of our proposed method. The proofs of theorem and corollary in this section are given in Appendix A.

Theorem 2. *Suppose that Assumption 1 holds. Let (N, e) be an RSA public key with n-bit N and fixed e. We choose*

$$T = \left\lceil \frac{\ln n}{2\epsilon'^2} \right\rceil, \ \gamma_i = \sqrt{\frac{t_i + 1}{T} \frac{\ln 2}{2}}, \ C_i = T\left(\frac{1}{2} + \gamma_i\right), \tag{8}$$

where t_i and Δ_i are defined in section 3.1. Furthermore, let $\overline{\mathbf{sk}} = (\overline{\mathbf{sk}}_1, \ldots, \overline{\mathbf{sk}}_m)$ be an RSA secret key with noise rate ϵ such that

$$\frac{1}{2} + \gamma_i \leq 1 - \frac{(T + \Delta_i)\epsilon}{T} - \epsilon' \tag{9}$$

for every i. Then, Restrictions 1 and 2 hold for every fixed $\epsilon' > 0$. Our method also corrects $\overline{\mathbf{sk}}$ in expected time $\mathcal{O}(n^{2+2(\frac{\ln 2}{2m\epsilon'^2} + \frac{\Delta}{m}\frac{\ln 2}{\ln n})})$ with success probability at least $1 - \left(\frac{(1-\delta)m\epsilon'^2}{\ln n} + \frac{1}{n}\right)$, where $\Delta = \max\{\Delta_i\}$ and $\delta mn/2 = \sum \Delta_i$.

By Theorem 2, we have the corollary.

Corollary 1. *Suppose that Assumption 1 holds and that the number of erasure bits is Δ for each block. We choose*

$$T = \left\lceil \frac{\ln n}{2\epsilon'^2} \right\rceil,\ t = \frac{T+\Delta}{m},\ \gamma = \sqrt{\left(1+\frac{1}{t}\right)\frac{\ln 2}{2m}},\ C = T\left(\frac{1}{2}+\gamma\right).$$

If δ and ϵ satisfy

$$\epsilon + \frac{\delta}{2} \leq \frac{1}{2} - \sqrt{\left(1+\frac{1}{t}\right)\frac{(1-\delta)\ln 2}{2m}} - (1-\delta)\epsilon', \tag{10}$$

then our method satisfies Restrictions 1 and 2 for every fixed $\epsilon' > 0$. It also corrects $\overline{\mathbf{sk}}$ in expected time $\mathcal{O}(n^{2+2(\frac{\ln 2}{2m\epsilon'^2}+\delta t\frac{\ln 2}{\ln n})})$ with success probability at least $1-\left(\frac{(1-\delta)m\epsilon'^2}{\ln n}+\frac{1}{n}\right)$.

Remark 2. For sufficiently large n, t goes to infinity and thus γ converges to $\sqrt{\frac{\ln 2}{2m}}$. This implies that our algorithm asymptotically works if

$$\epsilon + \frac{\delta}{2} \leq \frac{1}{2} - \sqrt{\frac{(1-\delta)\ln 2}{2m}} - \epsilon' \tag{11}$$

and succeeds with a probability close to 1. Hereafter, we ignore the term "$-\epsilon'$" for simplicity.

If the erasure rate δ is 0, then the new method is equivalent to the HMM method [5] by Corollary 1. Therefore, the new method naturally combines the results of the Heninger-Shacham and HMM methods. The upper bound of the new method coincides with that of Heninger-Shacham for $\epsilon = 0$ and that of the HMM method for $\delta = 0$. Finally, we confirm that our algorithm works well for $\delta = 0.6, \epsilon = 0.001$. Remember that our algorithm works provided that $\epsilon + \delta/2 \leq 1/2 - 0.263\sqrt{1-\delta}$. The left-hand side is given by 0.301 and the right-hand side is given by 0.334; the left-hand side is less than the right-hand side. Our algorithm works in that case.

4 Implementation and Experiments

We implemented our algorithm in the Risa/Asir [11] computer algebra system and used the program on an Intel Xeon X5570 at 2.93 GHz with 72 GB memory of DDR3 at 1333 MHz. In our experiments for 1024-bit RSA, we prepared 100 different tuples of secret keys $\mathbf{sk}$, e.g. $\mathbf{sk} = (p, q, d, d_p, d_q)$. For a fixed ϵ and δ, we generated one erroneous version $\overline{\mathbf{sk}}$ for each of $\mathbf{sk}$. For a given T, the threshold value C_i is determined by using Eq. (8).

Table 1 shows the experimental results for the case of $\mathbf{sk} = (p, q, d, d_p, d_q)$, $\epsilon = 0.01$, and $T = 40$. Note that the erasure rate δ was selected to be smaller than the theoretical bound 0.684 estimated by Eq. (11). Similarly, but for $T = 75$, we

Table 1. Experiments for $\mathbf{sk} = (p, q, d, d_p, d_q)$, $n = 1024$, $\epsilon = 0.01$, and $T = 40$

δ	0	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65
success rate	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.91	0.42	0.02
average time (s)	0.45	0.57	0.83	0.98	0.99	1.41	1.75	1.91	2.05	2.24	2.07	1.56	0.97	0.59

Table 2. Experiments for $\mathbf{sk} = (p, q)$, $n = 1024$, $\epsilon = 0.01$, and $T = 75$

δ	0	0.05	0.10	0.15	0.20	0.25
success rate	1.00	1.00	0.97	0.91	0.42	0.04
average time (s)	14.06	5.86	3.26	1.07	0.25	0.08

also conducted the experiments for the case of $\mathbf{sk} = (p, q)$ and the results are given in Table 2.

For fixed n, ϵ, and T, if an erasure rate δ becomes large, then the average of depth t_i becomes large with the increase in δ by Eq. (7). The average of threshold values C_i also increase because of the process of determining C_i, namely, Eq. (8). We determine these C_i's to satisfy Restriction 1 for the fixed T, so the success rate of our algorithm becomes small as Tables 1 and 2 show. If we use $T = 80$ instead of $T = 40$ for the case of Table 1, the success rate for $\delta = 0.65$ increases to 0.21 from 0.02 and the average time becomes 40.14 seconds.

5 Theoretical Bound

This section derives a theoretical upper bound for key recovery from noisy secret keys with errors and erasures in polynomial time.

First, we define the Hamming distance between two l-bit sequences; the symbol of one sequence (Sequence 1) is $\{0, 1\}$ and that of the other sequence (Sequence 2) is $\{0, 1, E\}$, where E is an erasure symbol. We denote the number of positions at which the corresponding symbols are different by h. We also denote the number of symbols E in Sequence 2 by a. We define the Hamming distance b between two sequences by $b := h - a$. We also have the equivalent definition of Hamming distance as follows. First, remove the bit of the position at which the symbol in Sequence 2 is E in Sequence 1 and remove the symbol E in Sequence 2. We define the Hamming distance between Sequences 1 and 2 by the ordinary Hamming weight between resulting sequences. For example, the Hamming weight between 1111 and 1E01 is 1.

We recall some known facts about the binary Entropy function. Remember that the binary Entropy function $H(x)$ is defined by $H(x) = -x \log x - (1 - x) \log(1 - x)$. It is well known that the following inequalities hold between the number of combinations and the binary Entropy [3].

Lemma 1. *For any positive integer n and $w(\leq n)$, it holds that*

$$\frac{1}{\sqrt{8w(1 - w/n)}} 2^{nH(w/n)} \leq \sum_{i=0}^{w} \binom{n}{i} \leq 2^{nH(w/n)}. \tag{12}$$

It is known that $H(x)$ can be represented by the following sum of an infinite series:

$$H(x) = 1 - \frac{1}{\ln 2} \sum_{u=1}^{\infty} \frac{1}{2u(2u-1)} (2x-1)^{2u}. \tag{13}$$

5.1 Maximal-Likelihood-Based Approach

We consider the following meta-algorithm.

`Meta-Algorithm` **for Recovering Keys**
Input: Public key (N, e) and noisy secret key $\overline{\mathbf{sk}}$, (ϵ, δ)
Output: Correct secret key $\mathbf{sk}$
Step 1: Expansion Phase (Virtually) generate a candidate set $\mathcal{C}$ by using the public information and Eqs. (2)–(5). Note that the number of elements of $\mathcal{C}$ is given by $2^{n/2-1}$.
Step 2: Pruning Phase Discard the candidate that is not consistent with $\overline{\mathbf{sk}}$. We denote the obtained set by $\mathcal{C}^*$.
Step 3: Finalization Phase Test whether each candidate solution in $\mathcal{C}^*$ is indeed the correct $\mathbf{sk}$ with the help of public information

The design of Step 2 is crucial for our algorithm. It is important to adequately determine criteria in Step 2 so that the correct solution $\boldsymbol{c}$ is not discarded during Step 2 in $\mathcal{C}^*$ and $|\mathcal{C}^*|$ is as small as possible. We discuss concrete criteria for discarding a candidate solution in Step 2. In order to do so, we adopt the maximal-likelihood-based approach.

Our analysis relies on a similar heuristic assumption as that in [5] and [6].

Assumption 2. *Every candidate solution in $\mathcal{C}$ is a bit-wise sum of $n/2-1$ randomly chosen bit and the correct sequence $\boldsymbol{c}$.*

We denote a candidate solution by $\boldsymbol{c} \in \mathcal{C}$. We discuss the conditional probability that we observed $\overline{\mathbf{sk}}$ under the condition that $\boldsymbol{c}$ is the correct solution. We denote the conditional probability by $\Pr(\overline{\mathbf{sk}}; \boldsymbol{c})$ and we refer to $\Pr(\overline{\mathbf{sk}}; \boldsymbol{c})$ as *likelihood*. In the maximal likelihood-based-approach, we decide that candidate that maximizes $\Pr(\overline{\mathbf{sk}}; \boldsymbol{c})$ is the correct solution.

This probability is simply evaluated as follows:

$$\Pr(\overline{\mathbf{sk}}; \boldsymbol{c}) = \delta^a \epsilon^b (1-\epsilon-\delta)^{mn/2-a-b} = (\delta/(1-\epsilon-\delta))^a \epsilon^b (1-\epsilon-\delta)^{mn/2-b},$$

where a is the number of erasure symbols in $\overline{\mathbf{sk}}$ and b is the Hamming distance between $\overline{\mathbf{sk}}$ and $\boldsymbol{c}$.

Since a does not depend on the choice of $\boldsymbol{c}$, it is sufficient to find b that maximizes the likelihood. If b is smaller, the likelihood is obviously bigger. Then, it is sufficient to find the solution $\boldsymbol{c}$ with the smallest Hamming distance to $\overline{\mathbf{sk}}$ for finding the solution that maximizes the likelihood. The Hamming distance b_{c} between the correct solution and $\overline{\mathbf{sk}}$ is $b_{\mathrm{c}} \approx \frac{mn\epsilon}{2}$ with high probability. Meanwhile, the Hamming distance b_{w} between the wrong solution and $\overline{\mathbf{sk}}$ is

$b_{\mathrm{w}} \approx m \times \frac{n}{2} \times \frac{1-\delta}{2} (> \frac{mn\epsilon}{2})$ with high probability. Then, it is sufficient to find the solution whose Hamming distance is $mn\epsilon/2$ in order to find the solution with maximal likelihood.

Remark 3. The computation of our proposed algorithm described in section 3 corresponds to finding the solution whose Hamming distance is less than $m \times \frac{n}{2} \times (1-\delta) \times (\frac{1}{2} - \gamma)$ for small positive γ. This implies that the correct solution is not discarded and falls within $\mathcal{C}^*$ with high probability. However, the size of $\mathcal{C}^*$ increases.

Remark 4. It is obviously impossible to execute Step 2 if the computational time is limited to a polynomial of n. In practice, we need to divide the candidate sequence into several sub-sequences and execute the expansion and pruning phase as in our proposed algorithm in section 3.

5.2 Deriving Theoretical Upper Bound

We derive the condition such that the `meta-algorithm` can never recover the secret key in polynomial time. This can be done by counting up the candidate solution that is not discarded during Step 2 and deriving the condition of (ϵ, δ) when the number of candidate solutions exceeds the polynomial of n.

We note that the candidate solution $\boldsymbol{c}$ is consistent with the observed solution $\overline{\mathbf{sk}}$ in Step 2 of the `meta-algorithm` if the following criteria hold.

CRITERIA. The Hamming distance between $\boldsymbol{c}$ and $\overline{\mathbf{sk}}$ is less than $mn\epsilon/2$.

Note that the expected bit length of the sequences removing erasures is given by $mn(1-\delta)/2$. The probability Pr that one candidate $\boldsymbol{c}$ is consistent with $\overline{\mathbf{sk}}$ is evaluated by

$$\Pr = \frac{\sum_{i=0}^{mn\epsilon/2} \binom{mn(1-\delta)/2}{i}}{2^{mn(1-\delta)/2}}. \tag{14}$$

From Lemma 1, Eq. (14) is lower bounded by

$$\Pr \geq 2^{-mn(1-\delta)(1-H(\epsilon/(1-\delta)))/2}. \tag{15}$$

We define $C(\epsilon, \delta)$ by $C(\epsilon, \delta) := (1-\delta)(1 - H(\epsilon/(1-\delta)))$. Then, the probability is larger than $2^{-mnC(\epsilon,\delta)/2}$. Since the number of candidate solutions is $2^{n/2}$, the expected number of candidate solutions consistent with the observed sequence $\overline{\mathbf{sk}}$ is lower bounded by $2^{n/2} 2^{-mnC(\epsilon,\delta)/2} = 2^{n(1-mC(\epsilon,\delta))/2}$.

Suppose that ϵ and δ satisfy the condition: $C(\epsilon, \delta) < 1/m$. This implies that $1 - mC(\epsilon, \delta) > 0$. Then, the expected number of candidate solutions consistent with $\overline{\mathbf{sk}}$ is an exponential function of n. Step 3 then requires the exponential testing of whether the candidate is indeed the secret key. Hence, the total computational time of the whole algorithm is actually exponential.

Conversely, suppose that $C(\epsilon, \delta) \geq 1/m$. Then, the number of candidate solutions is at most a polynomial of n and the total computational time dominates Step2. This means that it depends on $C(\epsilon, \delta)$ and $1/m$ whether there exists an algorithm that recovers in polynomial time of n. We show an information-theoretic view of our theoretical bound in the full version.

5.3 Discussion

Fig.1 shows achievable regions for the naive method and our proposed method in addition to the theoretical bound with $m = 5$. Note that all the values that lie below the respective line concerning the naive method and proposed method are vulnerable to each of the attacks and all the values that lie above the line about theoretical limitation are not solvable in polynomial time. We can see that the bound for our method nearly achieves the theoretical bound, but there is still a small gap.

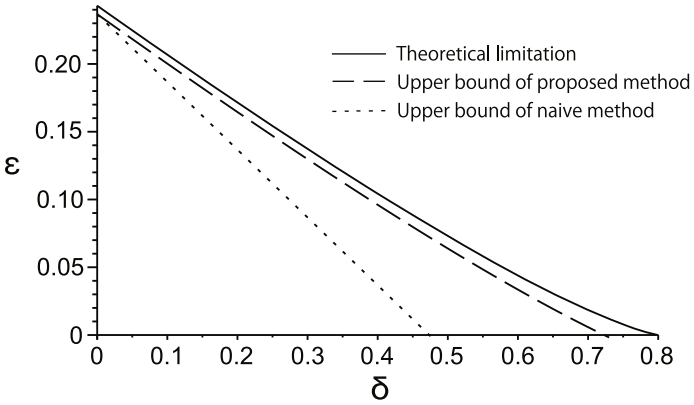


Fig. 1. Upper bounds of naive method and new method, and theoretical limitation

Table 3 shows the success conditions for three noise models; the upper is the bound the best-known algorithm achieves and the lower is the theoretical bound.

Table 3. Success Conditions of Heninger-Shacham, HMM, and our Proposed Methods

	model	BE model ($\epsilon = 0$)	BS model ($\delta = 0$)	BEE model
	best known algorithm	Heninger-Shacham [6]	HMM [5]	Proposed Method in Sec. 3
2	algorithm	$\delta \leq 0.43$	$\epsilon \leq 0.084$	$\epsilon + \delta/2 \leq \frac{1}{2} - 0.416\sqrt{1-\delta}$
2	theoretical bound	$\delta \leq 0.5$	$\epsilon \leq 0.110$	(ϵ, δ) s.t. $C(\epsilon, \delta) \geq 1/2$
5	algorithm	$\delta \leq 0.73$	$\epsilon \leq 0.237$	$\epsilon + \delta/2 \leq \frac{1}{2} - 0.263\sqrt{1-\delta}$
5	theoretical bound	$\delta \leq 0.8$	$\epsilon \leq 0.243$	(ϵ, δ) s.t. $C(\epsilon, \delta) \geq 1/5$
m	algorithm	$\delta \leq 1 - \frac{2\ln 2}{m}$	$\epsilon \leq \frac{1}{2} - \sqrt{\frac{\ln 2}{2m}}$	$\epsilon + \frac{\delta}{2} \leq \frac{1}{2} - \sqrt{\frac{(1-\delta)\ln 2}{2m}}$
m	theoretical bound	$\delta \leq 1 - \frac{1}{m}$	ϵ s. t. $H(\epsilon) \leq 1 - \frac{1}{m}$	(ϵ, δ) s.t. $C(\epsilon, \delta) \geq 1/m$

5.4 Our Algorithm Achieves Second-Order Expansion of Theoretical Bound

We present a strong bridge between the theoretical bound and achieved regions. We define the whole parameter space $\mathcal{I}$ by $\mathcal{I} := \{(\epsilon, \delta) | 0 \leq \epsilon < 1/2, 0 \leq \delta < 1\}$

and define $\mathcal{H}$ by

$$\mathcal{H} := \left\{ (\epsilon, \delta) | 0 \leq \epsilon < 1/2, 0 \leq \delta < 1, (1-\delta)\left(1 - H\left(\frac{\epsilon}{1-\delta}\right)\right) \geq \frac{1}{m} \right\}.$$

The discussion in Section 5.3 shows that we cannot recover the secret keys in polynomial time if $(\epsilon, \delta) \in \mathcal{I}/\mathcal{H}$. This argument suggests that we have a chance to recover the secret key in polynomial time if $(\epsilon, \delta) \in \mathcal{H}$. However, it does not guarantee that we can recover the secret keys if $(\epsilon, \delta) \in \mathcal{H}$. As shown in Fig. 1, there exists a small gap between our theoretical bound and the achieved regions. We give a strong relation between the two regions.

From Eq. (13), $C(\epsilon, \delta) < 1/m$ can be represented as follows:

$$\sum_{u=1}^{\infty} \frac{(1-\delta)}{2u(2u-1)} \left(\frac{1-\delta-2\epsilon}{1-\delta}\right)^{2u} \leq \frac{\ln 2}{m}, \tag{16}$$

which is a representation not explicitly used by the binary Entropy $H(\cdot)$. Consider the condition truncated by $u = k$ and denote the condition by $\mathcal{H}_k$

$$\mathcal{H}_k := \left\{ (\epsilon, \delta) | 0 \leq \epsilon < 1/2, 0 \leq \delta < 1, \sum_{u=1}^{k} \frac{(1-\delta)}{2u(2u-1)} \left(\frac{1-\delta-2\epsilon}{1-\delta}\right)^{2u} \leq \frac{\ln 2}{m} \right\}.$$

Obviously, it holds that $\mathcal{H}_i \subseteq \mathcal{H}_j$ if $i \leq j$ for any $i, j \in \mathbb{Z}$ and it holds that $\lim_{k\to\infty} \mathcal{H}_k = \mathcal{H}$.

We focus on the region $\mathcal{H}_1$. By simplifying the condition corresponding to $\mathcal{H}_1$, we have the equivalent condition:

$$1 - \delta - 2\epsilon \geq \sqrt{2(1-\delta)\ln 2/m}.$$

This is equivalent to the condition obtained in section 3: Eq. (11) if we neglect the small term ϵ. This implies that our proposed algorithm can recover the secret key in polynomial time if $(\epsilon, \delta) \in \mathcal{H}_1$.

Acknowledgement. We thank PKC 2012 and Asiacrypt 2012 reviewers for revising the manuscript. The first author was supported by JSPS Grant Number KAKENHI22700006.

References

1. Boneh, D., Durfee, G., Frankel, Y.: An Attack on RSA Given a Small Fraction of the Private Key Bits. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 25–34. Springer, Heidelberg (1998)
2. Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 178–189. Springer, Heidelberg (1996)
3. Cover, C.M., Thomas, J.A.: Elements of Information Theory, 2nd edn. Wiley-Interscience (2006)

4. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest We Remember: Cold Boot Attacks on Encryption Keys. In: Proc. of USENIX Security Symposium 2008, pp. 45–60 (2008)
5. Henecka, W., May, A., Meurer, A.: Correcting Errors in RSA Private Keys. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 351–369. Springer, Heidelberg (2010)
6. Heninger, N., Shacham, H.: Reconstructing RSA Private Keys from Random Key Bits. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 1–17. Springer, Heidelberg (2009)
7. Herrmann, M., May, A.: Solving Linear Equations Modulo Divisors: On Factoring Given Any Bits. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 406–424. Springer, Heidelberg (2008)
8. Hoeffding, W.: Probability Inequalities for Sums of Bounded Random Variables. Journal of the American Statistical Association 58(301), 13–30 (1963)
9. Paterson, K.G., Polychroniadou, A., Sibborn, D.L.: A Coding-Theoretic Approach to Recovering Noisy RSA Keys. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 386–403. Springer, Heidelberg (2012)
10. PKCS #1 Standard for RSA, `http://www.rsa.com/rsalabs/node.asp?id=2125`
11. Risa/Asir (Kobe Distribution) Download Page (2011), `http://www.math.kobe-u.ac.jp/Asir/asir.html`
12. Rivest, R., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
13. Yilek, S., Rescorla, E., Shacham, H., Enright, B., Savage, S.: When Private Keys are Public: Results from the 2008 Debian OpenSSL Vulnerability. In: IMC 2009, pp. 15–27. ACM Press (2009)

A Proofs of Theorem 2 and Corollary 1

A.1 Proofs of Theorem 2

First, we discuss how to determine the threshold value C_i satisfying Restriction 1 for a fixed T. Note that t_i and Δ_i are uniquely determined if T is fixed once.

In one i-th partial tree of the binary tree, there are 2^{t_i} candidates. Thus we defines 2^{t_i} variables $Z_{b,i}^j$ for $j = 1, \ldots, 2^{t_i}$ as

$$Z_{b,i}^j = \begin{cases} 1 \text{ (j-th bad candidate passes)} \\ 0 \text{ (otherwise.)} \end{cases}$$

Then, the number of bad candidates $Z_{b,i}$ given in Restriction 1 is described as $Z_{b,i} = \sum_{j=1}^{2^{t_i}} Z_{b,i}^j$. Since all $Z_{b,i}^j$ are identically distributed, there exists an integer j such that $\mathbb{E}[Z_{b,i}] = 2^{t_i}\mathbb{E}[Z_{b,i}^j]$.

Here we consider the number $X_{b,i}$ of matching bits between $\overline{\mathbf{sk}}$ and one bad candidate at the i-th pruning step skipping bits corresponding to erasure positions of $\overline{\mathbf{sk}}$. Since T bits of a bad candidate are compared with the corresponding bits of $\overline{\mathbf{sk}}$, we have that $X_{b,i} \sim \mathrm{Bin}(T, 1/2)$ by Assumption 1. The condition

$Z^j_{b,i} = 1$ is equivalent to that $X_{b,i} \geq C_i$, and thus $\mathbb{E}[Z^j_{b,i}] = \Pr[Z^j_{b,i} = 1] = \Pr[X_{b,i} \geq C_i]$. Supposing that

$$C_i = T\left(\frac{1}{2} + \gamma_i\right), \tag{17}$$

we have $\Pr[X_{b,i} \geq C_i] \leq \exp(-2T\gamma_i^2)$ from Theorem 1. Therefore, we obtain that $\mathbb{E}[Z_{b,i}]/2^{t_i} = \mathbb{E}[Z^j_{b,i}] \leq \exp(-2T\gamma_i^2)$. By setting

$$\gamma_i = \sqrt{\frac{t_i + 1}{T}\frac{\ln 2}{2}}, \tag{18}$$

we have $\exp(-2T\gamma_i^2) = 2^{-(t_i+1)}$. Restriction 1 holds since $\mathbb{E}[Z_{b,i}] \leq 2^{t_i}\exp(-2T\gamma_i^2) = 1/2$.

Let Y_i be the number of all bad candidates passing the i-th pruning step. Then, we have the following lemma.

Lemma 2. *Suppose that γ_i and C_i satisfy Eqs. (17) and (18) for a fixed T. Then, it holds that $\mathbb{E}[Y_i] < 2^{\max\{t_j\}_{j=1}^{i}+1}$.*

Proof. At the i-th pruning step, let $Z_{g,i}$ be the number of bad candidates generated from the correct solution, and $Z_{b,i}$ the number of bad candidates generated from one bad partial solution. Then, the following holds:

$$\mathbb{E}[Y_1] = \mathbb{E}[Z_{g,1}],\ \mathbb{E}[Y_i] = \mathbb{E}[Z_{g,i}] + \mathbb{E}[Z_{b,i}]\mathbb{E}[Y_{i-1}]. \tag{19}$$

Since the number of candidates is 2^{t_i}, we have $\mathbb{E}[Z_{g,i}] \leq 2^{t_i}$. For a given T, namely a fixed t_i, we determine γ_i and C_i so that Restriction 1 holds. From (19), we have

$$\mathbb{E}[Y_i] < 2^{t_i} + \frac{\mathbb{E}[Y_{i-1}]}{2} < 2^{\max\{t_j\}_{j=1}^{i}}\frac{1-(1/2)^i}{1-1/2} < 2^{\max\{t_j\}_{j=1}^{i}+1}.$$

Then, we have the lemma. □

Next we discuss T such that Restriction 2 holds. Let $X_{c,i}$ be the number of matching bits between $\overline{\mathbf{sk}}$ and the correct solution at the i-th pruning step without the bits corresponding to erasure positions of $\overline{\mathbf{sk}}$. Since we see total $(T + \Delta_i)$ bits and the T bits of them correspond to the non-erasure position of $\overline{\mathbf{sk}}$, the probability that a bit of a correct solution matches the corresponding bit of $\overline{\mathbf{sk}}$ is $(T - (T + \Delta_i)\epsilon)/T = 1 - (T + \Delta_i)\epsilon/T$. Therefore, since $X_{c,i} \sim \mathrm{Bin}(T, 1 - \frac{(T+\Delta_i)\epsilon}{T})$, we suppose that

$$\frac{1}{2} + \gamma_i \leq 1 - \frac{(T + \Delta_i)\epsilon}{T} - \epsilon',$$

for any i. From Theorem 1, we have that

$$\Pr[X_{c,i} < C_i] \leq \Pr\left[X_{c,i} < T\left(1 - \frac{(T + \Delta_i)\epsilon}{T} - \epsilon'\right)\right] \leq \exp(-2T\epsilon'^2).$$

Since we consider T such that Restriction 2 holds, $\exp(-2T\epsilon'^2) \leq 1/n$. Therefore, we have $T \geq \ln n/2\epsilon'^2$, and so we set $T = \lceil \ln n/(2\epsilon'^2) \rceil$.

By considering the above discussion, we have Theorem 2. The proof of Theorem 2 is given in detail below.

Proof. First we show that the total expected computational cost of the new method is $\mathcal{O}(n^{2+2(\frac{\ln 2}{2m\epsilon'^2}+\frac{\Delta}{m}\frac{\ln 2}{\ln n})})$. One node is computable in time $\mathcal{O}(n)$, so the partial tree is generated in time $\mathcal{O}(n2^{t_i})$ since there are $\sum_{j=0}^{t_i-1} 2^j (< 2^{t_i})$ nodes. The pruning step can be performed in time $\mathcal{O}(t_i)$ for each of 2^{t_i} candidates, and thus the total time complexity for pruning is $\mathcal{O}(t_i 2^{t_i})$. Therefore, the time complexity for one partial tree is $\mathcal{O}((n+t_i)2^{t_i}) = \mathcal{O}(n2^{t_i})$. For a given T, we suppose that the erroneous version $\overline{\mathbf{sk}}$ is separated into ℓ parts. By Eq. (7), t_i is bounded by $t_i^* = \lceil \frac{T+\Delta_i}{m} \rceil$. Let t^* be the maximum integer of $t_1^*, \ldots, t_\ell^*$. By Lemma 2, the upper bound for the expected total number $\mathbb{E}[Y]$ of partial trees is given by $\mathbb{E}[Y] < 1 + \sum_{j=1}^{\ell-1} \mathbb{E}[Y_j] < \ell 2^{t^*+1} \leq n2^{t^*+1} = \mathcal{O}(n2^{t^*})$. Let Δ be the Δ_i corresponding to t^*. Then, the total expected computational cost is

$$\mathcal{O}(n2^{t^*} \cdot n2^{t^*}) = \mathcal{O}(n^2 n^{2^{t^*}\frac{\ln 2}{\ln n}}) = \mathcal{O}(n^2 n^{2^{\frac{T+\Delta}{m}}\frac{\ln 2}{\ln n}}) = \mathcal{O}(n^{2+2(\frac{\ln 2}{2m\epsilon'^2}+\frac{\Delta}{m}\frac{\ln 2}{\ln n})}).$$

Next we discuss the success probability of the new method. Note that C_i, γ_i and T are determined so that Restriction 2 holds. Hence the success probability is given by

$$\prod_{i=1}^{\ell}(1 - \Pr[X_c < C_i]) \geq \left(1 - \frac{1}{n}\right)^{\ell} \geq 1 - \frac{\ell}{n} \geq 1 - \left(\frac{(1-\delta)m\epsilon'^2}{\ln n} + \frac{1}{n}\right)$$

since $\ell \leq \frac{\frac{n}{2}(1-\delta)m}{T} + 1$. □

A.2 Proofs of Corollary 1

To give the proof of Corollary 1, we begin with the discussion of Eq. (9) in the analysis of our method. For simplicity, we consider only the case where all δ_i's are the same[1], for example, $\delta_i = \delta$. Suppose that $\overline{\mathbf{sk}}$ is separated into ℓ fractions. Then, each part consists of $mn/2\ell$ bits. By letting $t = n/2\ell$, we have $\Delta = \delta tm$ and $T = tm - \Delta = (1-\delta)tm$, so we can describe γ_i in Theorem 2 as $\sqrt{\frac{t+1}{(1-\delta)tm}\frac{\ln 2}{2}}$. Hence, in this case, the upper bound (9) implies that

$$\epsilon + \frac{\delta}{2} \leq \frac{1}{2} - \sqrt{\left(1 + \frac{1}{t}\right)\frac{(1-\delta)\ln 2}{2m}} - (1-\delta)\epsilon'.$$

[1] For a large enough T, it holds with high probability. More precisely, all of δ_i takes the value close to $\delta T/(1-\delta)$ with overwhelming probability, which can be proved by the similar analysis of [6].

Combined Attack on CRT-RSA

Why Public Verification Must Not Be Public?

Guillaume Barbu[1], Alberto Battistello[1], Guillaume Dabosville[2], Christophe Giraud[1], Guénaël Renault[3], Soline Renner[1,4], and Rina Zeitoun[2,3]

[1] Oberthur Technologies – Security Group
4, allée du Doyen Georges Brus, 33 600 Pessac, France
[2] Oberthur Technologies – Crypto Group
71-73, rue des Hautes Pâtures, 92 726 Nanterre Cedex, France
[3] UPMC, Université Paris 6, INRIA, Centre Paris-Rocquencourt,
PolSys Project-team, CNRS, UMR 7606, LIP6
4 place Jussieu, 75252 Paris, Cedex 05, France
[4] Institut Mathématiques de Bordeaux, Université Bordeaux I
351, cours de la Libération, 33 405 Talence Cedex, France
f.lastname@oberthur.com,
guenael.renault@lip6.fr

Abstract. This article introduces a new Combined Attack on a CRT-RSA implementation resistant against Side-Channel Analysis and Fault Injection attacks. Such implementations prevent the attacker from obtaining the signature when a fault has been induced during the computation. Indeed, such a value would allow the attacker to recover the RSA private key by computing the *gcd* of the public modulus and the faulty signature. The principle of our attack is to inject a fault during the signature computation and to perform a Side-Channel Analysis targeting a sensitive value processed during the Fault Injection countermeasure execution. The resulting information is then used to factorize the public modulus, leading to the disclosure of the whole RSA private key. After presenting a detailed account of our attack, we explain how its complexity can be significantly reduced by using lattice reduction techniques. We also provide simulations that confirm the efficiency of our attack as well as two different countermeasures having a very small impact on the performance of the algorithm. As it performs a Side-Channel Analysis during a Fault Injection countermeasure to retrieve the secret value, this article recalls the need for Fault Injection and Side-Channel Analysis countermeasures as monolithic implementations.

Keywords: CRT-RSA, Combined Attacks, Fault Injection, Side-Channel Analysis, Coppersmith's methods.

1 Introduction

Since the seminal work of Kocher published in 1996 [1], Side-Channel Analysis (SCA) has raised a huge interest in both academic and industrial communities.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 198–215, 2013.

This kind of attack is based on the fact that side-channel leakages of embedded devices contain information on the values manipulated inside the device. Therefore any sensitive variable carelessly used can be recovered by an attacker using SCA. Originally, time execution was used as side-channel leakage but the exploitation of power consumption and electromagnetic radiation became quickly the most efficient way to attack embedded cryptography [2, 3]. Over the years, many improvements have been made leading to very efficient attacks and very ingenious countermeasures [4].

In parallel to SCA, Fault Injection (FI) provides the attacker with another way to attack embedded devices. Such attacks aim at disturbing cryptographic computations and the analysis of corresponding faulty outputs allows the attacker to recover the secret key [5]. Shortly after the original publication focusing on RSA implementation [6], many other articles have been published to present FI attacks on various cryptosystems such as DES, ElGamal or DSA signature schemes [7,8]. As for SCA, FI has been deeply studied over the last decade [9] and the consequences of both attacks on the industry are huge since secure products must now be certified to prove their resistance against such threats.

Over the last few years, the cryptographic community has investigated the possibility of combining the two previous kinds of attacks. This has resulted in a new class of attacks called Combined Attacks (CA) that can defeat implementations which are meant to resist both SCA and FI. However, as far as we know only four CA have been published since their introduction in 2007, proving the difficulty to conceive such attacks [10–13].

Nowadays, most embedded devices implement a large variety of cryptosystems to ensure the security of the sensitive assets they contain. As well as being the first practical public-key cryptosystem published, RSA [14] has also been the most widely used for many years. In particular, the RSA using the Chinese Remainder Theorem (CRT), providing a speed-up factor of four compared to the original implementation, is available on most ID, banking and mobile smart cards. Obviously, this cryptosystem has been the main target of SCA and FI attackers leading to the development of efficient countermeasures having the smallest impact on both memory consumption and time execution due to the constraints of embedded environment.

In this article, we describe a new Combined Attack against a CRT-RSA implementation resistant to SCA by using blinding countermeasures and protected against FI by verifying the signature using the public exponent. Such an implementation is known to resist each and every kind of attack published so far. However, we demonstrate that when injecting a fault during the signature computation, a value depending on the message and on a multiple of a secret prime is manipulated in plain during the public verification. Therefore, we notice that one can use SCA to gain some information on such a sensitive value. The recovered information can then be used to factorize the RSA modulus and thus to reveal the whole private key. Besides, we exploit lattice techniques, and in particular Coppersmith's methods for finding small solutions to polynomial equations [15, 16], to significantly reduce the complexity of our CA.

The rest of this paper is organised as follows. In Section 2 we briefly recall some basics on CRT-RSA signature as well as the corresponding attacks and countermeasures. In Section 3 we describe our new CA on a CRT-RSA implementation that is known to resist both SCA and FI attacks. In Section 4 we present the results of our simulations which prove the efficiency of our new attack. We then improve its complexity by using lattice reduction techniques in Section 5. Finally, we suggest in Section 6 possible countermeasures having a negligible penalty on the performance of the algorithm.

2 Previous Works

In this section we briefly recall the RSA signature, in particular the CRT mode. Secondly we present the principal attacks on such an algorithm as well as the main countermeasures.

2.1 RSA on Embedded Systems

Since its introduction in 1978, the RSA cryptosystem has become one of the most used public-key cryptosystems, especially in electronic signature schemes [14]. In the following we briefly recall how to compute the RSA signature in both standard and CRT modes.

Let N denote the public modulus being the product of two secret large prime integers p and q. Let d refer to the private exponent and e refer to the public exponent satisfying $de = 1 \bmod \varphi(N)$, where φ denotes Euler's totient function. The RSA signature of a message $m \in \mathbb{Z}_N$ is then obtained by computing $S = m^d \bmod N$. To verify the signature, one computes $S^e \bmod N$ and checks if the corresponding result is equal to m.

In embedded systems, most RSA implementations use the Chinese Remainder Theorem (CRT) which yields an expected speed-up factor of four [17]. Following the CRT-RSA algorithm, the signature generation is composed of two exponentiations $S_p = m^{d_p} \bmod p$ and $S_q = m^{d_q} \bmod q$, where $d_p = d \bmod p - 1$ and $d_q = d \bmod q - 1$. The signature is then obtained by recombining S_p and S_q, which is usually done by using Garner's formula [18]:

$$S = CRT(S_p, S_q) = S_q + q(i_q(S_p - S_q) \bmod p) \ , \tag{1}$$

where $i_q = q^{-1} \bmod p$.

2.2 Attacks and Countermeasures

Side-Channel Analysis. Side-Channel Analysis (SCA) has been introduced by the publication of the so-called timing attacks in 1996 [1]. SCA exploits the dependency between the manipulated data or the executed instruction and the side-channel leakages which can be monitored during the algorithm execution. Examples of such leakages are the power consumption or the electromagnetic

radiation of the device. When only one measure is required to exploit sensitive information, the attack is called Simple Passive Analysis (SPA) [2]. In the case of a straightforward *Square-and-Multiply* exponentiation, SPA consists for instance in observing if the squaring and multiplication operations have different patterns in the corresponding side-channel leakages [2]. Hence, the secret exponent can be directly extracted from one measurement. In the literature, a common countermeasures consists in using a so-called *regular* algorithm which performs the same operation whatever the exponent bit value such as the *Square-Always* or *Montgomery ladder* algorithms [19, 20].

Moreover, attacks based on side-channel leakages have evolved to a type of SCA called Differential Passive Analysis (DPA) [2] which requires a large number of measurements. This type of attack applies a statistical treatment on the curves to recover information on the manipulated values.Nowadays a common statistic tool used to perform such a statistical treatment, is the Pearson correlation coefficient:

$$\rho_k = \frac{\mathrm{cov}(\mathcal{L}, H)}{\sigma_{\mathcal{L}} \sigma_H}, \tag{2}$$

where $\mathcal{L}$ is the set of curves and H depends on a known value m and on a guess of a small part of a secret k. Such an attack is called Correlation Power Analysis (CPA) [21]. In the literature, many different CPAs have been published to attack the RSA cryptosystem [22]. For instance in the CRT-mode, an attacker can mount a CPA to recover the private parameter q during the CRT-recombination, *cf.* Rel. (1), by observing the leakage obtained during the manipulation of the value $i_q(S_p - S_q) \bmod p$ and by making a guess on a few bits of q. Hence, an attacker can obtain the whole secret q by performing a CPA for each of its subpart (typically for each byte). However, half of q is sufficient to recover the rest of q by using Coppersmith's attack [23]. Classical countermeasures to resist CPA consist in randomizing the modulus, the message and the exponent [22].

Fault Injection. RSA has been the first cryptosystem to succumb to Fault Injection [24]. In the following, we describe such an attack in the CRT case. Assume that a fault is injected during the computation of S_p leading to a faulty signature $\widetilde{S}$. Since $S \equiv S_p \bmod p$ and $S \equiv S_q \bmod q$, one can notice that $\widetilde{S} \equiv S \bmod q$ but $\widetilde{S} \not\equiv S \bmod p$. Therefore, the secret parameter q can be easily recovered by computing the *gcd* of $S - \widetilde{S}$ and N. The rest of the private key can then be straightforwardly deduced.

When it is not possible to sign the same message twice and if the message is known to the attacker, a variant of this attack consists in computing the *gcd* of $\widetilde{S}^e - m$ and N to obtain the secret value q [25].

Moreover, the effect of fault injections on CRT-RSA is not limited to the disturbance of S_p or S_q. Indeed, a fault injected in any part of the key parameters (i.e. p, q, d_p, d_q or i_q), in the message m at the beginning of either S_p or S_q computation, or even during the CRT-recombination can lead to a useful faulty signature.

In the literature, four different fault models are generally considered to define the attacker's capabilities [26]:

- the *random* fault model: the bits are changed to a uniformly distributed random value;
- the *bit-flip* fault model: in that case, affected bits are flipped to their complementary value;
- the *stuck-at* fault model: the fault sets the bits to 0 or to 1, depending on the underlying hardware;
- the *unknown constant* fault model: the fault always sets the bits to the same unknown value.

Moreover, these faults do not necessarily modify a whole temporary result. Indeed, it is generally considered that the number of bits affected by the fault is linked to the CPU word-size which is generally 8, 16 or 32 bits.

The most natural way to counteract fault injection on RSA-type signature is to check the correctness of the signature S before outputting it [24]. More precisely, the signature is returned iff $S^e \bmod N = m$. Moreover, such a method requires very little overhead since the public exponent e is usually small in practice (typically 3, 17 or $2^{16} + 1$).

Other methods getting rid of e have also been proposed but they do not offer the same level of security and are generally slower than the public verification [27–29].

Combined Attacks. The idea to combine SCA and FI appeared in 2007 when Amiel *et al.* proposed a so-called Combined Attack (CA) on an RSA implementation protected against FI and SPA [10]. They noticed that by setting to zero one of the temporary registers used in the Montgomery ladder, its structure becomes unbalanced, revealing the value of the secret exponent by SPA. Following this publication, three other papers have been published taking advantage of this new way of defeating embedded security. Two of them present a CA against a secured AES implementation [12,13]. The third one focuses on the elliptic curve scalar multiplication [11].

Despite its theoretical effectiveness, the combination of SCA and FI is very difficult in practice, explaining the lack of practical experiments in the current literature.

Lattices. Randomized RSA encoding schemes are usually considered to be resistant to traditional FI attacks since a part of the message is unknown to the attacker and varies for each signature computation. However this common assumption has to be mitigated regarding the works of [30] and [31] which defeat two randomised RSA encoding schemes. These attacks use Coppersmiths method to solve a bivariate polynomial whose coefficients are built thanks to the generated faulty signatures.

More recently in [32], the authors present an attack taking advantage of the disturbance of the public modulus. The generated faulty signatures allow them to build a lattice, which in turns leads to factorize the public modulus.

Although [30], [31] and [32] also apply lattice reduction techniques as carried out in this paper, the attack presented hereafter does not share the same context since we consider a secured implementation which never returns the faulty signature.

3 A New Combined Attack on CRT-RSA

3.1 Context and Principle

As stated in Section 2, several countermeasures have been developed to protect CRT-RSA embedded implementations against both SCA and FI. In the framework of this article, we consider an algorithm protected:

- against SCA by using message and exponent blinding as suggested in [33], a regular exponentiation algorithm such as the Square Always [20] and a mask refreshing method along the exponentiation such as the one presented in [34]. Moreover, the blinding is kept all along the CRT-recombination.
- against FI by verifying the signature using the public exponent e [24]. In addition, we also use the approach presented in [35] which mainly consists in checking the result of the verification twice to counteract double FI attacks.

Fig. 1 depicts the main steps of such an implementation where the k_i's are random values (typically of 64 bits) generated at each execution of the algorithm and S'_p, S'_q and S' represent the blinded version of S_p, S_q and S respectively.

In the following, we assume that the fault injected by the attacker follows either the *bit-fault*, the *stuck-at* or the *unknown constant* fault models (cf. Section 2.2). Moreover, we assume the attacker is able to choose which byte of the message is affected by the fault.

As mentioned in Section 2.2, injecting a fault during the signature computation leads to a faulty signature that allows the attacker to recover the private key. However in the implementation considered in this paper, the verification with the public exponent detects such a disturbance and the faulty signature is never revealed to the attacker. The main contribution of this paper is to show that in this case, an SCA can still allow the attacker to gain enough information on the faulty signature to recover the private key.

At first glance, it seems impossible to perform such an attack during the signature process due to the blinding countermeasure. However by observing Fig. 1, one may note that the faulty signature $\widetilde{S}$ remains blinded until the end of exponentiation with e modulo N. Therefore if we can express $\widetilde{S}^e \bmod N$ in terms of the message m and of the private key then we can perform an SCA on this value. In the next section, we exhibit such a relation allowing us to mount a CA on an SCA-FI-resistant CRT-RSA implementation.

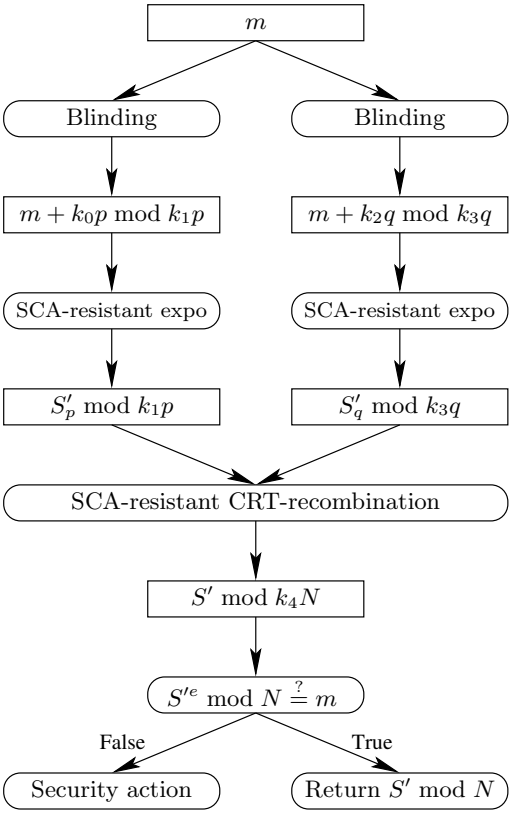


Fig. 1. Main steps of a CRT-RSA implementation secure against SCA and FI.

3.2 A Useful Relation

Proposition 1. *If a fault ε is induced in m such that the faulty message $\widetilde{m}$ is equal to $m + \varepsilon$ at the very beginning of the computation of S_p then*

$$\widetilde{S}^e \equiv m + \varepsilon q i_q \bmod N \ , \tag{3}$$

where $\widetilde{S}$ corresponds to the faulty signature.

Proof. By definition of the CRT-RSA signature, we have:

$$\begin{cases} \widetilde{S} \equiv (m + \varepsilon)^d \bmod p \\ \widetilde{S} \equiv m^d \bmod q \end{cases} \tag{4}$$

It comes then straightforwardly that:

$$\begin{cases} \widetilde{S}^e \equiv m + \varepsilon \bmod p \\ \widetilde{S}^e \equiv m \bmod q \end{cases} \tag{5}$$

Finally, applying Gauss recombination to (5) leads to (3) since:

$$\widetilde{S}^e \equiv pi_p m + qi_q(m+\varepsilon) \bmod N \quad (6)$$
$$\equiv (pi_p m + qi_q m) + \varepsilon qi_q \bmod N \quad (7)$$
$$\equiv m + \varepsilon qi_q \bmod N \ , \quad (8)$$

where $i_p = p^{-1} \bmod q$. □

One may note that a similar relation holds if m is disturbed at the very beginning of S_q computation due to the symmetrical roles of p and q in both branches of the CRT-RSA. For the sake of simplicity, we will use the case where S_p computation is disturbed in the rest of this paper.

3.3 Recovering the Private Key

Following the attack's principle depicted in Section 3.1 and using Proposition 1, we will now present in detail the main steps of our attack.

Firstly, the attacker asks the embedded device to sign several messages m_i through a CRT-RSA implemented as described in Section 3.1. For each signature, the computation of S_q is performed correctly and a constant additive error ε is injected on the message m_i at the beginning of each S_p computation. Then during each signature verification, the attacker monitors the corresponding side-channel leakage $\mathcal{L}_i$ which represents the manipulation of $\widetilde{S}_i^e \bmod N$.

From Proposition 1, we know that there exists a sensitive value k satisfying the relation $\widetilde{S}_i^e \bmod N = m_i + k$. Therefore, the attacker will perform a CPA to recover this sensitive value by computing $\rho_k(m_i + k, \mathcal{L}_i)$ for all the possible values of k (cf. Section 2.2).

Depending on the set $\{(m_i, \widetilde{S}_i^e \bmod N)\}_i$, it follows from Rel. (3) that k will be equal either to $\varepsilon qi_q \bmod N$ or to $\varepsilon qi_q \bmod N - N$. Therefore, the value $\widehat{k}$ producing the strongest correlation at the end of the CPA will be one of these two values. Once $\widehat{k}$ recovered, the attacker must then compute the *gcd* between $\widehat{k}$ and N, which leads to the disclosure of q. From this value, the private key is straightforwardly computed.

Regarding the practicality of our fault model (i.e. a constant additive fault), one may note that by fixing a small part of the message (e.g. a byte), the disturbance of such a part in either the stuck-at, the bit-flip or the unknown constant fault model results in a constant additive error during the different signature computations. Therefore our fault model is definitely valid if the attacker can choose the messages to sign, or even if she can only have the knowledge of the messages and attack only those with a given common part.

Finally, one may note that it is not possible to perform a statistical attack targeting the full value of k at once due to its large size (i.e. $\lceil \log_2(N) \rceil$ bits). However, one can attack each subpart of this value, for instance by attacking byte per byte starting with the least significant one in order to be able to propagate easily the carry. It is worth noticing that CPA only applies when the corresponding part of the message varies. Therefore, if the attacker fixes the MSB of the

message, then the corresponding set of measurements can be used to recover the whole but last byte of $\widehat{k}$. In such a case, a brute force search can be used to recover the missing byte.

In the next section, we present simulations of our attack which prove the efficiency of our method and which are based on the attacker's capability to inject the same fault and on the noise of the side-channel measurements.

4 Experiments

The success of the attack presented in Section 3 relies on the ability of the attacker to both measure the side-channel leakage of the system during the signature verification and induce the same fault ε on the different manipulated messages.

In order to evaluate the effectiveness of this attack, we have experimented it on simulated curves of the side-channel leakage $\mathcal{L}$, according to the following leakage model:

$$\mathcal{L}(d) = HW(d) + \mathcal{N}(\mu, \sigma) \tag{9}$$

with $\mathcal{N}(\mu, \sigma)$ a Gaussian noise of mean μ and standard deviation σ, and $HW(d)$ the Hamming weight function evaluated for the manipulated data d. In the framework of our experiments, we consider that the processor manipulates 8-bit words and we use three different levels of noise, namely $\sigma = 0.1$, 1 and 5.

As well as the side-channel leakage, the faults were also simulated by setting the most significant word of the message m to all-0 at the very beginning of the S_p computation. These faults were induced with a given success rate r, varying in our different experiment campaigns (namely 50%, 10% and 1%).

Depending on the experimental settings, all the different words of the secret value will be equivalently correlated with the simulated curves. The graphs presented in Fig. 2 present the convergence of the correlation for each possible value k of one particular byte (the 5^{th} least-significant byte) of the secret depending on the number of side-channel measurements with different simulation settings σ and r.

As exposed in Fig. 2, the number of traces required to recover the secret value depends essentially on the fault injection success rate. This comes from the fact that every wrongly-faulted computation can be considered as noise in the scope of our statistical analysis. The number of curves required to retrieve the secret word grows as the fault injection success rate decreases and to a fewer extent as the noise of the side-channel leakage increases.

With regards to the results obtained when $\sigma = 5$ and $r = 10\%$, which appear to be plausible values in practice, it took us 3.35 seconds to retrieve one byte of the secret value by performing the CPA on $15,000$ curves of 128 points each[1]. Assuming a genuine curve should be made of at least $5,000$ points, we can estimate the time required to practically perform the attack to about 1 minute

[1] The execution time given here and in Section 5.2 have been obtained on a 32-bit CPU @3.2GHz.

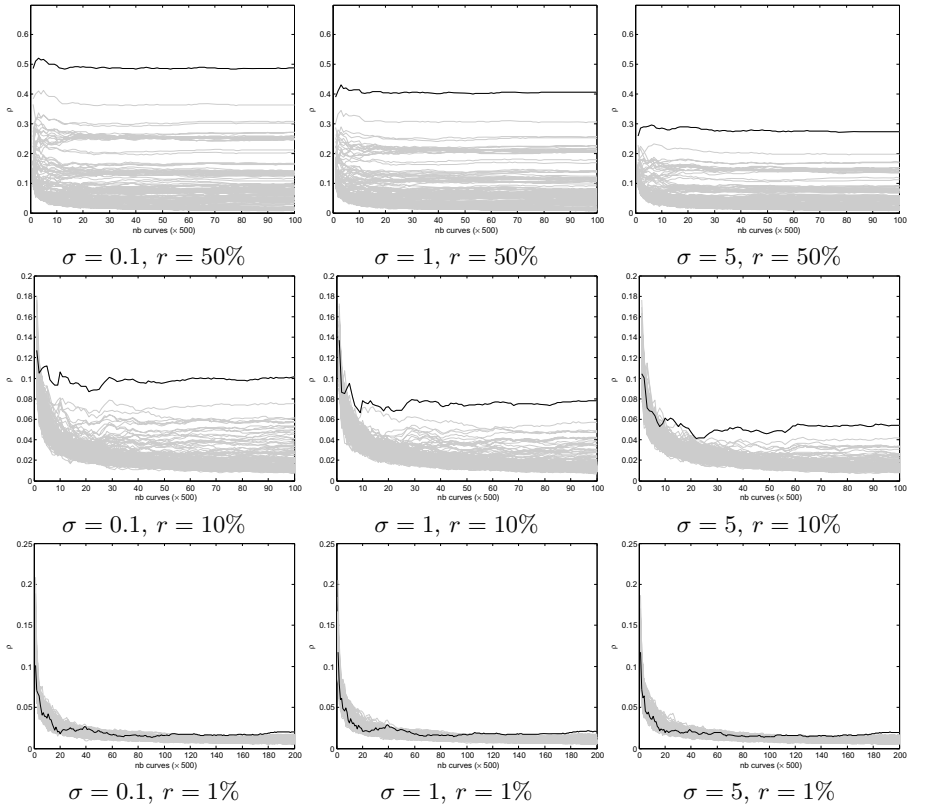


Fig. 2. Convergence of the correlation for the 256 possible values k_i for the secret (the correct one being depicted in black) depending on the number of side-channel measurements ($\times 500$) for different levels of noise σ and fault injection success rates r

5 seconds per byte. That is to say, it takes about 2 hours 20 minutes to recover the complete secret value if we consider a 1024-bit RSA module.

For the sake of clarity, we restrained the experiments presented here to the case where the processor manipulates 8-bit words, and thus ε is an 8-bit error. The same experiments have been run for processor word-size up to 32 bits with success. Besides, about the same number of curves were necessary for the CPA to highlight the correct secret byte.

Section 5 shows how it is possible to considerably reduce the complexity of our attack thanks to the use of lattice techniques.

5 Reducing the Attack Complexity Using Coppersmith's Methods

This section aims at improving the attack complexity using Coppersmith's methods. It is in line with the problem of factorizing N knowing half part of prime p

(or q), that was solved in [16]. With respect to our case, we highlight that if the CA presented in Section 3 provides about half of the secret $\varepsilon q i_q \bmod N$, then the other half part can be straightforwardly computed by solving a well-designed modular polynomial equation that we elaborate in the sequel. Besides, we deal with two cases (ε known and unknown), depending on the fault model that is considered.

5.1 Bringing Up the Original Problem to Solving a Modular Equation

Suppose we are given the t least significant bits (LSB) of the secret $\varepsilon q i_q \bmod N$. The latter value can be rewritten as follows:

$$\varepsilon q i_q \equiv 2^t x_0 + k \bmod N \ , \tag{10}$$

where t and k are known values, and x_0 is the $\lceil \log_2(N) - t \rceil$-bit unknown integer that is to be recovered.

Lemma 1. *The unknown secret part x_0 is solution of the polynomial $P_\varepsilon(x)$:*

$$P_\varepsilon(x) = x^2 + c\ (2^{t+1}k - 2^t\varepsilon)\ x + c\ (k^2 - k\varepsilon) \equiv 0 \bmod N \ , \tag{11}$$

where $c = (2^{2t})^{-1} \bmod N$, k, t, N are known, and ε is the induced fault.

Proof. The Bézout identity applied to our context yields that primes p and q interrelate with integers $i_p = p^{-1} \bmod q$ and $i_q = q^{-1} \bmod p$ by the following relation:

$$p i_p + q i_q \equiv 1 \bmod N \ . \tag{12}$$

Multiplying (12) by ε leads to the relation $\varepsilon p i_p + \varepsilon q i_q \equiv \varepsilon \bmod N$, or equivalently to $\varepsilon p i_p \equiv \varepsilon - \varepsilon q i_q \bmod N$. Therefore, replacing $\varepsilon q i_q$ using (10) allows us to deduce an equivalence for $\varepsilon p i_p$:

$$\varepsilon p i_p \equiv \varepsilon - 2^t x_0 - k \bmod N \ . \tag{13}$$

As $N = pq$, we then multiply (10) by (13), to get the relation:

$$\varepsilon q i_q \cdot \varepsilon p i_p \equiv (2^t x_0 + k) \cdot (\varepsilon - 2^t x_0 - k) \equiv 0 \bmod N \ . \tag{14}$$

Eventually, developing the right-hand side of (14), and multiplying it by $c = (2^{2t})^{-1} \bmod N$ leads to the obtention of the monic polynomial $P_\varepsilon(x)$. □

The initial problem of retrieving the unknown part of $\varepsilon q i_q \bmod N$ is thereby altered in solving the modular polynomial equation (11). In the sequel, we deal with two possible cases regarding ε, whether it is known to the adversary or not.

Case 1. **The fault ε is known to the adversary**

This case corresponds to the *bit-flip* and *stuck-at* fault models (Section 2.2) since the message is known to the attacker and the fault location can be chosen. In both cases, since the fault ε is known, the problem is reduced to solving a univariate modular polynomial equation, *cf.* Rel. (11). This problem is known to be hard. However, when the integer solution x_0 is small, Coppersmith showed [23] that it can be retrieved using the well-known LLL algorithm. Accordingly, we induce the following proposition:

Proposition 2. *Given $N = pq$ and the low order $1/2\ \log_2(N)$ bits of $\varepsilon q i_q \bmod N$ and assuming ε is known, one can recover in time polynomial in $(\log_2(N), d)$ the factorization of N.*

Proof. From Coppersmith's Theorem [16], we know that, given a monic polynomial $P(x)$ of degree d, modulo an integer N of unknown factorization, and an upper bound X on the desired solution x_0, one can find in polynomial time all integers x_0 such that

$$P(x_0) \equiv 0 \bmod N \qquad \text{and} \qquad |X| < N^{1/d} \ . \tag{15}$$

In our case we have $d = 2$, and since x_0 is a $\lceil \log_2(N) - t \rceil$-bit integer, we know that $|x_0| < X = 2^{\lceil \log_2(N) - t \rceil}$. Thus, the condition in (15) becomes $2^{\lceil \log_2(N) - t \rceil} < N^{1/2}$, *i.e.*

$$t > \frac{1}{2} \log_2(N) \ . \tag{16}$$

Therefore, knowing at least half part of the secret $\varepsilon q i_q \bmod N$ allows to recover the whole secret. As previously done, computing $gcd(\varepsilon q i_q \bmod N, N)$ provides the factorization of N. □

Note that the method is deterministic, and as will be seen further (Table 1), it is reasonably fast.

Case 2. **The fault ε is unknown to the adversary**

This case is met in the *unknown constant* fault model (Section 2.2). In such a case, one can consider the polynomial $P_\varepsilon(x)$ as a bivariate modular polynomial equation with unknown values x and ε. This specific scheme has also been studied by Coppersmith and includes an additional difficulty of algebraic dependency of vectors which induces the heuristic characteristic of the method [15]. As depicted in Section 5.2, in our experiments nearly 100% of the tests verified the favorable property of independency. Accordingly, in this vast majority of cases, the following proposition holds:

Proposition 3. *Under an hypothesis of independency (see discussion above), given $N = pq$ and the low order $1/2\ \log_2(N) + s$ bits of $\varepsilon q i_q \bmod N$, where s denotes the bitsize of ε, and assuming ε is unknown, one can recover in time polynomial in $(\log_2(N), d)$ the factorization of N.*

Proof. Coppersmith's Theorem for the bivariate modular case [15] notifies that given a polynomial $P(x, \varepsilon)$ of total degree d, modulo an integer N of unknown factorization, and upper bounds X and E on the desired solutions x_0, ε_0, it may be possible (heuristic) to find in polynomial time all integer couples (x_0, ε_0) such that

$$P(x_0, \varepsilon_0) \equiv 0 \bmod N \qquad \text{and} \qquad |X \cdot E| < N^{1/d} \ . \tag{17}$$

In our case, we have $d = 2$ and $E = 2^s$. The integer x_0 is $\lceil \log_2(N) - t \rceil$-bit long, therefore we have $X = 2^{\lceil \log_2(N) - t \rceil}$. Thus, the condition in (17) becomes $2^{\lceil \log_2(N) - t \rceil} \cdot 2^s < N^{1/2}$, *i.e.*

$$t > \frac{1}{2} \log_2(N) + s \ . \tag{18}$$

This means that knowing s more bits of the secret $\varepsilon q i_q \bmod N$ than before, would allow the recovering of the whole secret. □

Remark 1. The bound of success in Proposition 3 can actually be slightly improved using results of [36]. Indeed, Coppersmith's bound applies to polynomials whose monomials shape is rectangular, while in our case the monomial ε^2 does not appear in $P(x, \varepsilon)$ which corresponds to what they called an *extended rectangle* in [36]. For the sake of simplicity, we only mentioned Coppersmith's bound since practical results are similar.

5.2 Results from Our Implementation

We have implemented this lattice-based improvement using Magma Software [37], with N a 1024-bit integer *i.e.* 128 bytes long, in the cases where ε is an 8-bit known value (for Case 1) and a 32-bit unknown value (for Case 2). We chose Howgrave-Graham's method [38] for the univariate case, and its generalization by Jochemsz *et al.* [39] for the bivariate case since both have the same bound of success as Coppersmith's method (sometimes even better for [39]) and they are easier to implement. As we know, the theoretical bound given in Coppersmith's method is only asymptotic [16]. Thus, we report in Table 1 (for Case 1) and in Table 2 (for Case 2) the size t (in bytes) of the secret $\varepsilon q i_q \bmod N$ that is known to the attacker before applying Coppersmith's method, the lattice dimension used to solve (11) and finally the timings of our attack.

As depicted in Table 1, and combining these results with the experiments of Section 4, the best trade-off is to perform a CPA on the 66 first bytes, taking $66 \times 1m05s = 1h11m30s$, and to retrieve the 62 remaining bytes using lattices in $34.25s$, bringing the total time up to 1 hour 12 minutes, instead of the previous 2 hours 20 minutes.

In order to illustrate Case 2, we have chosen to rather show our results for ε being a 32-bit value, since when ε is 8-bit long, we obtained slightly better results by considering the 255 possible values of the variable ε together with their corresponding polynomials $P_\varepsilon(x)$, and by running the method on each of the polynomials until finding the solution x_0 that allows to factorize N.

Table 1. Size t required (in bytes) for the method to work and timings (Magma V2.17-1), as a function of the lattice dimension in Case 1 (ε known, being an 8-bit integer)

Size t required (bytes)	86	72	70	69	68	67	**66**	65	64
Dimension of the lattice	3	9	11	15	17	23	**37**	73	Theoretical
Time for LLL (seconds)	< 0.01	0.03	0.07	0.29	0.52	2.63	**34.25**	2587.7	bound

Table 2. Size t required (in bytes) for the method to work and timings (Magma V2.17-1), as a function of the lattice dimension in Case 2 (ε unknown, being a 32-bit integer)

Size t required (bytes)	86	78	76	74	73	**72**	71	70	69
Dimension of the lattice	5	12	22	35	51	**70**	117	201	Theoretical
Time for LLL (seconds)	< 0.01	0.02	0.16	1.17	5.88	**30.22**	605.9	12071.1	bound

This indeed leads to a best trade-off of 70 bytes required from the CPA and the 58 remaining bytes computed with lattices by performing 255 times the LLL algorithm in the worst case, for a total of $68 \times 1m05s + 255 \times 0.52s$, *i.e.* 1 hour 16 minutes instead of 2 hours 20 minutes. Besides, this exhaustive search can be performed in parallel and it also has the advantage to be deterministic.

However, when ε is 32-bit long, an exhaustive search becomes impractical and, as depicted in Table 2, the best trade-off would be to perform a CPA on 72 bytes and to compute the 56 remaining bytes with lattices (even if heuristic, it worked in nearly 100% of the tests in practice), resulting in a total of $72 \times 1m05s + 30.22s$, *i.e.* 1 hour 18 minutes instead of the previous 2 hours 20 minutes.

6 Countermeasures

In this section, we describe different countermeasures to protect an implementation against the CA presented in Section 3.

6.1 Blind before Splitting

Our first proposition consists in avoiding the possibility to inject the same fault during several signature computations. To do so, we deport the blinding of the input message m before executing the two exponentiations modulo p and q:

$$m' = m + k_0 N \bmod k_1 N \ , \tag{19}$$

with k_0 and k_1 two n-bit random values generated at each algorithm execution (n being typically 64). Hence $S'_p = m'^{d_p} \bmod k_2 p$ and $S'_q = m'^{d_q} \bmod k_3 q$.

This countermeasure prevents an attacker from injecting always the same error during the signature computation. Indeed if the fault is injected on m at the very beginning of one exponentiation, then the corresponding error cannot be fixed due to the blinding injected by Rel. (19).

Moreover, if the fault is injected when the message m is manipulated during (19), then the error ε impacts the computation of both S'_p and S'_q, leading to inexploitable faulty outputs.

Such a countermeasure induces a small overhead in terms of memory space since m' must be kept in memory during the first exponentiation but the execution time remains the same.

6.2 Verification Blinding

Our second countermeasure aims at annihilating the second hypothesis of our attack: a predictive variable is manipulated in plain during the verification. To do so, we inject a $\lceil \log_2(N) \rceil$-bit random r before performing the final reduction with N, *cf.* Rel. (20). Therefore, each and every variable manipulated during the verification is blinded.

$$((\widetilde{S}^e + r - m) \bmod k_1 N) \bmod N \stackrel{?}{=} r \ . \tag{20}$$

One may note that the final comparison should be performed securely with regards to the attack described in [40] since information on $\varepsilon q i_q$ could leak if such a comparison was performed through a substraction.

The cost of such a countermeasure is negligible since it mainly consists in generating a $\lceil \log_2(N) \rceil$-bit random variable.

7 Conclusion

This paper introduces a new Combined Attack on CRT-RSA. Even if a secure implementation does not return the faulty signature when the computation is disturbed, we show how to combine FI with SCA during the verification process to obtain information on the faulty signature. Such information allows us to factorize the public modulus and thus to recover the whole private key. We also show that Coppersmith's methods to solve univariate and bivariate modular polynomial equations can be used to significantly reduce the complexity of our new attack. Finally, we provide simulations to confirm the efficiency of our method and we present two countermeasures which have a very small penalty on the performance of the algorithm.

Our main objective was to prove that stacking several countermeasures does not provide global security despite addressing each and every attack separately. Therefore, the main consequence of this paper is that fault injection countermeasures must also be designed to resist SCA and vice versa.

Acknowledgements. The authors would like to thank Emmanuelle Dottax for her useful comments on the preliminary version of this article. We would also like to thank Jean-Sébastien Coron and the anonymous reviewers of PKC'13 for their valuable comments and suggestions.

References

1. Kocher, P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
2. Kocher, P.C., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
3. Quisquater, J.J., Samyde, D.: A New Tool for Non-intrusive Analysis of Smart Cards Based on Electro-magnetic Emissions, the SEMA and DEMA Methods. Presented at EUROCRYPT 2000 Rump Session (2000)
4. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks – Revealing the Secrets of Smartcards. Springer (2007)
5. Giraud, C., Thiebeauld, H.: A Survey on Fault Attacks. In: CARDIS 2004, pp. 159–176. Kluwer Academic Publishers (2004)
6. Bellcore: New Threat Model Breaks Crypto Codes. Press Release (1996)
7. Biham, E., Shamir, A.: Differential Fault Analysis of Secret Key Cryptosystems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)
8. Bao, F., Deng, R.H., Han, Y., Jeng, A., Narasimhalu, A.D., Ngair, T.: Breaking Public Key Cryptosystems an Tamper Resistance Devices in the Presence of Transient Fault. In: Christianson, B., Crispo, B., Lomas, M., Roe, M. (eds.) Security Protocols 1997. LNCS, vol. 1361, pp. 115–124. Springer, Heidelberg (1998)
9. Joye, M., Tunstall, M.: Fault Analysis in Cryptography. Information Security and Cryptography. Springer (2012)
10. Amiel, F., Feix, B., Marcel, L., Villegas, K.: Passive and Active Combined Attacks – Combining Fault Attacks and Side Channel Analysis. In: FDTC 2007, pp. 92–99. IEEE Computer Society (2007)
11. Fan, J., Gierlichs, B., Vercauteren, F.: To Infinity and Beyond: Combined Attack on ECC Using Points of Low Order. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 143–159. Springer, Heidelberg (2011)
12. Roche, T., Lomné, V., Khalfallah, K.: Combined Fault and Side-Channel Attack on Protected Implementations of AES. In: Prouff, E. (ed.) CARDIS 2011. LNCS, vol. 7079, pp. 65–83. Springer, Heidelberg (2011)
13. Robisson, B., Manet, P.: Differential Behavioral Analysis. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 413–426. Springer, Heidelberg (2007)
14. Rivest, R., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21, 120–126 (1978)
15. Coppersmith, D.: Finding a Small Root of a Univariate Modular Equation. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 155–165. Springer, Heidelberg (1996)
16. Coppersmith, D.: Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. Journal of Cryptology 10, 233–260 (1997)
17. Couvreur, C., Quisquater, J.J.: Fast Decipherment Algorithm for RSA Public-Key Cryptosystem. Electronics Letters 18, 905–907 (1982)

18. Garner, H.: The Residue Number System. IRE Transactions on Electronic Computers 8, 140–147 (1959)
19. Joye, M., Yen, S.M.: The Montgomery Powering Ladder. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 291–302. Springer, Heidelberg (2003)
20. Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., Verneuil, V.: Square Always Exponentiation. In: Bernstein, D.J., Chatterjee, S. (eds.) INDOCRYPT 2011. LNCS, vol. 7107, pp. 40–57. Springer, Heidelberg (2011)
21. 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)
22. Amiel, F., Feix, B., Villegas, K.: Power Analysis for Secret Recovering and Reverse Engineering of Public Key Algorithms. In: Adams, C., Miri, A., Wiener, M. (eds.) SAC 2007. LNCS, vol. 4876, pp. 110–125. Springer, Heidelberg (2007)
23. Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 178–189. Springer, Heidelberg (1996)
24. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the Importance of Checking Cryptographic Protocols for Faults. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg (1997)
25. Lenstra, A.: Memo on RSA Signature Generation in the Presence of Faults. Manuscript (1996)
26. Blömer, J., Otto, M., Seifert, J.P.: A New RSA-CRT Algorithm Secure against Bellcore Attacks. In: CCS 2003 ACM Conference, pp. 311–320. ACM Press (2003)
27. Giraud, C.: An RSA Implementation Resistant to Fault Attacks and to Simple Power Analysis. IEEE Transactions on Computers 55, 1116–1120 (2006)
28. Vigilant, D.: RSA with CRT: A New Cost-Effective Solution to Thwart Fault Attacks. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 130–145. Springer, Heidelberg (2008)
29. Rivain, M.: Securing RSA against Fault Analysis by Double Addition Chain Exponentiation. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 459–480. Springer, Heidelberg (2009)
30. Coron, J.-S., Joux, A., Kizhvatov, I., Naccache, D., Paillier, P.: Fault Attacks on RSA Signatures with Partially Unknown Messages. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 444–456. Springer, Heidelberg (2009)
31. Coron, J.-S., Naccache, D., Tibouchi, M.: Fault Attacks Against EMV Signatures. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 208–220. Springer, Heidelberg (2010)
32. Brier, É., Naccache, D., Nguyen, P.Q., Tibouchi, M.: Modulus Fault Attacks against RSA-CRT Signatures. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 192–206. Springer, Heidelberg (2011)
33. Witteman, M.F., van Woudenberg, J.G.J., Menarini, F.: Defeating RSA Multiply-Always and Message Blinding Countermeasures. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 77–88. Springer, Heidelberg (2011)
34. Dupaquis, V., Venelli, A.: Redundant Modular Reduction Algorithms. In: Prouff, E. (ed.) CARDIS 2011. LNCS, vol. 7079, pp. 102–114. Springer, Heidelberg (2011)
35. Dottax, E., Giraud, C., Rivain, M., Sierra, Y.: On Second-Order Fault Analysis Resistance for CRT-RSA Implementations. In: Markowitch, O., Bilas, A., Hoepman, J.-H., Mitchell, C.J., Quisquater, J.-J. (eds.) WISTP 2009. LNCS, vol. 5746, pp. 68–83. Springer, Heidelberg (2009)

36. Blömer, J., May, A.: A Tool Kit for Finding Small Roots of Bivariate Polynomials over the Integers. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 251–267. Springer, Heidelberg (2005)
37. Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system. I. The user language. J. Symbolic Comput. 24, 235–265 (1997)
38. Howgrave-Graham, N.: Finding Small Roots of Univariate Modular Equations Revisited. In: Darnell, M.J. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg (1997)
39. Jochemsz, E., May, A.: A Strategy for Finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variants. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 267–282. Springer, Heidelberg (2006)
40. Lomne, V., Roche, T., Thillard, A.: On the Need of Randomness in Fault Attack Countermeasures – Application to AES. In: FDTC 2012, pp. 85–94. IEEE Computer Society (2012)

Revocable Identity-Based Encryption Revisited: Security Model and Construction

Jae Hong Seo and Keita Emura

National Institute of Information and Communications Technology (NICT), 4-2-1, Nukui-kitamachi, Koganei, Tokyo, 184-8795, Japan
{jaehong,k-emura}@nict.go.jp

Abstract. In ACM CCS 2008, Boldyreva et al. proposed an elegant way of achieving an Identity-based Encryption (IBE) with *efficient* revocation, which we call revocable IBE (RIBE). One of the significant benefit of their construction is scalability, where the overhead of the trusted authority is logarithmically increased in the number of users, whereas that in the Boneh-Franklin naive revocation way is linearly increased. All subsequent RIBE schemes follow the Boldyreva et al. security model and syntax. In this paper, we first revisit the Boldyreva et al. security model, and aim at capturing the exact notion for the security of the naive but non-scalable Boneh-Franklin RIBE scheme. To this end, we consider a realistic threat, which we call *decryption key exposure*. We also show that all prior RIBE constructions except for the Boneh-Franklin one are vulnerable to decryption key exposure. As the second contribution, we revisit approaches to achieve (efficient and adaptively secure) scalable RIBE schemes, and propose a simple RIBE scheme, which is the first scalable RIBE scheme with decryption key exposure resistance, and is more efficient than previous (adaptively secure) scalable RIBE schemes. In particular, our construction has the shortest ciphertext size and the fastest decryption algorithm even compared with all scalable RIBE schemes without decryption key exposure resistance.

Keywords: Identity-based encryption with revocation, decryption key exposure.

1 Introduction

Identity-based Encryption (IBE) provides an important alternative way to avoid the need for a public key infrastructure (PKI). Revocation capability is very important for IBE setting as well as PKI setting. An efficient way to revoke users in the traditional PKI setting has been studied in numerous studies [2, 17, 18, 20, 30–33]. In contrast to PKI setting, there are only a few studies on IBE setting. First, Boneh and Franklin [11] consider one naive revocation way as follows. Let ID be a receiver's identity and T be a time to be decrypted. An encryptor uses (ID, T) as the public key, and a trusted authority, called key generation center (KGC), issues private keys $\mathsf{pvk}_{(ID,T)}$ for all non-revoked user

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 216–234, 2013.

IDs on each time period T via secure channels.[1] We call the Boneh-Franklin revocable IBE (RIBE) scheme BF-RIBE. However, the BF-RIBE does not scale well; that is, the overhead on KGC is linearly increased in the number of users. Recently, some studies [7, 13, 28] have aimed at offering scalability in the RIBE scheme while preserving the same security level as the BF-RIBE.

For the first time, Boldyreva et al. [7] formalized the security model of RIBE by capturing possible threats, and proposed the first *scalable* RIBE (BGK-RIBE) scheme by combining Fuzzy IBE [36] with a binary tree data structure, which was previously used in a revocation scheme [32]; Each user is given a long-term secret key sk_{ID} from KGC (via a secure channel as in IBE), and KGC broadcasts key update ku_T in each time period T (i.e., no secure channel is required in this phase). Only a non-revoked user can generate a short-term decryption key $\mathsf{dk}_{ID,T}$ from sk_{ID} and ku_T, which can be used to decrypt ciphertexts in time T. By using a binary tree data structure, the size of ku_T can be much smaller than the overhead of KGC in the BF-RIBE scheme.[2] Several scalable RIBE schemes have been proposed and those are provably secure in the Boldyreva et al. security model.

Our Contribution. Our contribution consists of two parts. First, we separate the Boldyreva et al. security model and the security level of the BF-RIBE by introducing a new realistic threat, which we call *decryption key exposure*, and also show that all previous RIBE schemes, except the BF-RIBE, are vulnerable to decryption key exposure.[3] That is, we show that the Boldyreva et al. security model does not fully capture the exact notion for security of the BF-RIBE scheme. Roughly speaking, the Boldyreva et al. security model allows an adversary to obtain any secret keys of a chosen identity. The only one restriction is that if the adversary obtains sk_{ID^*} of the challenge identity ID^*, then ID^* should be revoked before the challenge time T^*. This model is a natural extension of the security of the ordinary IBE scheme. However, *does this security model formalize all realistic threats?* For example, if the short-term decryption key $\mathsf{dk}_{ID,T}$ ($T \neq T^*$) is leaked, is the RIBE scheme still secure? The answer to this question may naturally appear to be 'yes' since the adversary can obtain secret keys of any chosen identity, and the decryption key can be generated from a secret key and (public) key update. But to show this thinking is wrong, we give an exceptional attack (decryption key exposure), wherein an adversary is allowed to obtain a decryption key $\mathsf{dk}_{ID^*,T}$ with the condition $T \neq T^*$. This setting is based on the similar attitude of key-insulated PKE [16], where it is desired that no information of the plaintext is revealed from a ciphertext even if

[1] Boldyreva et al. [7] provided an alternative way for this naive solution to avoid a secure channel, wherein the previous-time key is used to establish a public channel. However, this does not match the framework of RIBE, and thus we do not discuss it in this paper.

[2] In fact, the size of ku_T is $O(R\log(N/R))$ if $R \leq N/2$, and $O(N-R)$, otherwise, where N is the number of users and R is the number of revoked users.

[3] We do not contradict the security proofs given in previous schemes. Our attack is positioned in outside of their security models.

Table 1. (Pairing-based) Revocable IBE schemes

	CT size	Dec. cost	Mpk size	Model	Scalability	DKE resistance	Assumption
BF [11]	$1\tau_{\mathbb{G}} + 1\tau_H$	1p	$3\tau_{\mathbb{G}}$ +2hash ft.	RO, Adaptive		$\surd$	BDH
BGK [7]	$3\tau_{\mathbb{G}} + 1\tau_{\mathbb{G}_t}$	4p	$6\tau_{\mathbb{G}}$	Standard, Selective	$\surd$		DBDH
LV [28]	$3\tau_{\mathbb{G}} + 2\tau_{\mathbb{G}_t}$	3p	$(n+6)\tau_{\mathbb{G}}$	Standard, Adaptive	$\surd$		DBDH
Ours	$3\tau_{\mathbb{G}} + 1\tau_{\mathbb{G}_t}$	3p	$(n+6)\tau_{\mathbb{G}}$	Standard, Adaptive	$\surd$	$\surd$	DBDH

$\tau_{\mathbb{G}}$ and $\tau_{\mathbb{G}_t}$ are the sizes of groups $\mathbb{G}$ and $\mathbb{G}_t$, respectively, over which a bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_t$ is defined. τ_H is the range-size of a hash function. p is the cost for performing a bilinear pairing e. n is the size of the identity space. RO (Standard, respectively) is a random oracle model (standard model, respectively). Selective (Adaptive, respectively) means a selective-security model (adaptive-security model, respectively). 'DKE' means decryption key exposure. (D)BDH is (Decisional) Bilinear Diffie-Hellman assumption.

all (short-term) decryption keys of a "different time period" are exposed. This kind of attack is not covered by the Boldyreva et al. security model; that is, the adversary may obtain not a secret key sk_{ID^*} but a decryption key $\mathsf{dk}_{ID^*,T}$, and ID^* can still be alive in the system in the challenge time period $T^* \neq T$. However, we can easily show that the BF-RIBE is still secure against decryption key exposure since every decryption key in the BF-RIBE is a private key with a distinct identity (ID, T) in the Boneh-Franklin IBE scheme.

Next, we revisit approaches to achieve (adaptively secure) scalable RIBE schemes, and propose a simple RIBE scheme by combining the (adaptively secure) Waters IBE scheme [38] and the (selectively secure) Boneh-Boyen IBE scheme [8]. This is the first scalable RIBE scheme with decryption key exposure resistance, and is more efficient than previous (adaptively secure) scalable RIBE schemes. Surprisingly, our construction does not require any additional efficiency cost for achieving decryption key exposure resistance. In particular, our construction has the shortest ciphertext size and a fastest decryption algorithm even compared with all scalable RIBE schemes without decryption key exposure resistance. Table 1 gives a detailed comparison with previous (efficient pairing-based) schemes. From our standard model RIBE construction, we can easily obtain more efficient RIBE construction in the random oracle model, by replacing both the Waters hash and the Boneh-Boyen hash into cryptographic hash functions that are modeled as random oracles.

Our construction is natural in the sense that its security can be reduced to the original (non-revocable) Waters IBE scheme. However, in [28], Libert and Vergnaud mentioned that this kind of simple construction using the original Waters IBE scheme will face with the difficulty in the security proof, and they

circumvented this by using a variant of the Waters IBE scheme [29] instead of the original.[4] We resolve this difficulty by carefully dealing with the means of assigning nodes of a binary tree to each user, which we call *random node assignment* technique. This allows us to circumvent the difficulty, and is explained in section 4. Surprisingly, such a simple construction is secure against decryption key exposure. The main difference between ours and previous constructions is the *re-randomizable* property of the decryption key, whereas decryption keys use the same randomness used in the secret key in all previous constructions.

Related Work. After the Boneh-Franklin RIBE scheme [11] and the Boldyreva et al. *scalable* (but *selectively secure*) RIBE scheme [7], there were some results. Libert and Vergnaud [28] proposed the first *adaptively secure* RIBE scheme (LV-RIBE) without assuming any stronger assumption compared with that of the Boldyreva et al.[5] Later an RIBE scheme from lattices [13] also have proposed. All these RIBE schemes are proven secure in the security model proposed by Boldyreva et al. [7].

Revocable IBE with mediators [5, 10, 14, 26] has been considered, where a special semi-trusted authority called a mediator who helps users to decrypt each ciphertext. However, this essentially requires communication between users and the mediator at each decryption and so is not totally satisfactory in some practical circumstances.

Recently, several functional encryption (FE) schemes, which are generalizations of the IBE scheme, have been proposed [23, 36], and the revocation capability in FE has also been studied [3, 4, 34]. The revocation method used in [4, 34] differs from RIBE contexts; the senders carry out the revocation, so it does not require any private key update procedures on the recipient's side. In [3] Attrapadung and Imai considered two different ways for revocation method; one is similar to that in [4, 34], and the other is similar to that in RIBE schemes. However, decryption key exposure is not considered in [3], so achieving revocation capabilities in FE with decryption key exposure resistance would be an interesting future area of study.

All revocable IBE schemes use a strategy in which only decryption keys of users who are not revoked in a time period T can be updated in time period T. This strategy is similar to those for cryptosystems against key exposure such as key-insulated PKE [6, 16, 27] and IBE [21, 22, 39, 40], forward secure encryption [12], and intrusion-resilient PKE [15]. However, these systems require a secure channel between a user and a key issuer or do not support scalability.

Outline. The next section gives preliminaries. In Section 3, we provide definitions for the RIBE scheme and explain the vulnerability of previous RIBE schemes against decryption key exposure. Section 4 gives our construction for a

[4] In a footnote of [28], there is a remark that a two-level hierarchical version of the Waters original IBE seems to work, however the details are not provided.

[5] Both schemes are secure under the Decisional Bilinear Diffie-Hellman (DBDH) assumption.

scalable RIBE scheme (with decryption key exposure resistance) and a high-level description of its security proof. We discuss about a room for extention of our RIBE scheme in Section 5. Finally, we summarize our result and leave several interesting open problems in Section 6.

2 Bilinear Groups and Waters IBE Scheme

Definition 1 (Bilinear Groups). *A bilinear group generator $\mathcal{G}(\cdot)$ is an algorithm that takes as input a security parameter λ and outputs a bilinear group $(p, \mathbb{G}, \mathbb{G}_t, e)$, where p is a prime of size 2λ, $\mathbb{G}$ and $\mathbb{G}_t$ are cyclic groups of order p, and e is an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with*

- *Bilinearity : for all $u, u', v, v' \in \mathbb{G}$, $e(uu', v) = e(u, v)e(u', v)$ and $e(u, vv') = e(u, v)e(u, v')$,*
- *Non-degeneracy : for a generator g of $\mathbb{G}$, $e(g, g) \neq 1_{\mathbb{G}_t}$, where $1_{\mathbb{G}_t}$ is the identity element in $\mathbb{G}_t$.*

In the security proof of our RIBE construction, we provide a reduction from our IND-RID-CPA secure RIBE scheme to the IND-ID-CPA secure Waters IBE scheme, which is secure under the DBDH assumption. We give the definition of the DBDH assumption, description of the Waters IBE scheme, and IND-ID-CPA security of ordinary IBE schemes.

Definition 2 (Decision Bilinear Diffie-Hellman (DBDH) Assumption). *Given a bilinear group $(p, \mathbb{G}, \mathbb{G}_t, e)$ generated by $\mathcal{G}(\lambda)$, define two distributions $\mathcal{D}_0(\lambda) = (g, g^a, g^b, g^c, e(g, g)^{abc}) \in \mathbb{G}^4 \times \mathbb{G}_T$ and $\mathcal{D}_1(\lambda) = (g, g^a, g^b, g^c, e(g, g)^z) \in \mathbb{G}^4 \times \mathbb{G}_T$, where $g \xleftarrow{\$} \mathbb{G}$ and $a, b, c, z \xleftarrow{\$} \mathbb{Z}_p$. The DBDH problem in the bilinear group $(p, \mathbb{G}, \mathbb{G}_t, e)$ is to decide a bit b from given $\mathcal{D}_b$, where $b \xleftarrow{\$} \{0, 1\}$. The advantage of $\mathcal{A}$ in solving the DBDH problem in the bilinear group $(p, \mathbb{G}, \mathbb{G}_t, e)$ is defined by*

$$Adv^{DBDH}_{\mathcal{G},\mathcal{A}}(\lambda) = \Big|\Pr[\mathcal{A}(\mathcal{D}_0(\lambda)) \to 1] - \Pr[\mathcal{A}(\mathcal{D}_1(\lambda)) \to 1]\Big|.$$

We say that the DBDH assumption holds in the bilinear group $(p, \mathbb{G}, \mathbb{G}_t, e)$ if no Probabilistic Polynomial Time (PPT) algorithm has a non-negligible advantage in solving the DBDH problem in the bilinear group $(p, \mathbb{G}, \mathbb{G}_t, e)$.

Definition 3 (Waters IBE). *The Waters IBE consists of four algorithms* $\mathsf{Setup}_{\mathsf{Wat}}$, $\mathsf{PKG}_{\mathsf{Wat}}$, $\mathsf{Enc}_{\mathsf{Wat}}$, *and* $\mathsf{Dec}_{\mathsf{Wat}}$.

- $\mathsf{Setup}_{\mathsf{Wat}}(\lambda)$: *Generate a bilinear group $(p, \mathbb{G}, \mathbb{G}_t, e) \leftarrow \mathcal{G}(\lambda)$. Choose g, g_2, u', $u_1, \ldots, u_n \xleftarrow{\$} \mathbb{G}$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p$. Set $g_1 = g^\alpha$. Publish a master public key $\mathsf{mpk}_{\mathsf{Wat}} = \{g, g_1, g_2, u', u_1, \ldots, u_n\}$ and keep a master secret key $\mathsf{msk}_{\mathsf{Wat}} = \{g_2^\alpha\}$.*

– $\mathsf{PKG}_{\mathsf{Wat}}(\mathsf{mpk}_{\mathsf{Wat}}, \mathsf{msk}_{\mathsf{Wat}}, ID)$: *Parse* $ID = (b_1, \ldots, b_n) \in \{0,1\}^n$, *where for all* $i \in [1, n]$, $b_i \in \{0,1\}$. *Choose* $r \xleftarrow{\$} \mathbb{Z}_p$ *and return the private key*

$$\mathsf{pvk}_{ID} = (g_2^{\alpha}(u' \prod_{i=1}^{n} u_i^{b_i})^r,\ g^r).$$

– $\mathsf{Enc}_{\mathsf{Wat}}(\mathsf{mpk}_{\mathsf{Wat}}, ID, M)$: *Parse* $ID = (b_1, \ldots, b_n) \in \{0,1\}^n$. *Choose* $t \xleftarrow{\$} \mathbb{Z}_p$ *and return a ciphertext*

$$\mathsf{CT}_{\mathsf{Wat}} := (M \cdot e(g_1, g_2)^t,\ g^{-t},\ (u' \prod_{i=1}^{n} u_i^{b_i})^t).$$

– $\mathsf{Dec}_{\mathsf{Wat}}(\mathsf{mpk}_{\mathsf{Wat}}, \mathsf{pvk}_{ID}, \mathsf{CT}_{\mathsf{Wat}})$: *Parse* $\mathsf{CT}_{\mathsf{Wat}} = (C_0, C_1, C_2)$ *and* $\mathsf{pvk}_{ID} = (d_0, d_1)$ *and return*

$$C_0 \cdot e(C_1, d_0) \cdot e(C_2, d_1).$$

Definition 4 (IND-ID-CPA). *Let* $\mathcal{IBE} = (\mathsf{Setup}, \mathsf{PKG}, \mathsf{Enc}, \mathsf{Dec})$ *be an IBE scheme. For adversary* $\mathcal{A}$ *define the following experiment:*

$\mathbf{Exp}^{IND\text{-}ID\text{-}CPA}_{\mathcal{IBE},\mathcal{A}}(\lambda)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(\lambda)$;
$(M_0^*, M_1^*, ID^*, \mathsf{st}) \leftarrow \mathcal{A}^{\mathsf{PKG}(\cdot)}(\mathsf{mpk})$ *such that* $|M_0^*| = |M_1^*|$;
$b \xleftarrow{\$} \{0,1\}$;
$\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{mpk}, ID^*, M_b^*)$;
$b' \leftarrow \mathcal{A}^{\mathsf{PKG}(\cdot)}(\mathsf{CT}^*, \mathsf{st})$;
If $b = b'$ *return* 1 *else return* 0.

In the above experiment, $\mathsf{PKG}(\cdot)$ *is an oracle, which returns a private key* pvk_{ID} *of given identity* ID, *and* $\mathcal{A}$ *is not allowed to send* ID^* *to* $\mathsf{PKG}(\cdot)$.

An IBE scheme is said to be IND-ID-CPA if for all PPT adversaries $\mathcal{A}$ *the following advantage is negligible in the security parameter* λ.

$$Adv^{IND\text{-}ID\text{-}CPA}_{\mathcal{IBE},\mathcal{A}}(\lambda) = \left| \Pr\left[\mathbf{Exp}^{IND\text{-}ID\text{-}CPA}_{\mathcal{IBE},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2} \right|.$$

Theorem 1 ([38]). *The Waters IBE scheme is IND-ID-CPA secure under DBDH assumption. More precisely, if there exists an adversary* $\mathcal{A}$ *breaking IND-ID-CPA security of the Waters IBE scheme with* ϵ *advantage, then by using* $\mathcal{A}$, *we can construct an algorithm* $\mathcal{B}$ *solving DBDH problem in the same bilinear group, over which the Waters IBE scheme is defined, with* $O(\frac{\epsilon}{nq})$ *advantage, where* q *is the maximum number of key extraction queries issued by* $\mathcal{A}$.

3 Definition of RIBE Scheme

In this subsection, we give the formal definition of the syntax and the security model of our RIBE construction. First, we give the syntax of RIBE scheme.

Our syntax of RIBE scheme is slightly different from previous one [7, 28][6] ; Our DKG algorithm is probabilistic, whereas the Boldyreva et al. one is deterministic.[7] A RIBE scheme $\mathcal{RIBE}$ consists of seven algorithms (Setup, PKG, KeyUp, DKG, Enc, Dec, Revoke). Let $\mathcal{M}$, $\mathcal{I}$, and $\mathcal{T}$ be a message space, an identity space, and a time space, respectively.

Definition 5 (Syntax of RIBE).

Setup *: This is the (stateful) setup algorithm which takes as input the security parameter λ and the number of users N, and outputs the public parameter* mpk*, the master secret key* msk*, the initial revocation list $RL = \emptyset$, and a state* st*.*

PKG *: This is the (stateful) private key generation algorithm which takes as input* mpk*,* msk*, an identity $ID \in \mathcal{I}$, and outputs a secret key sk_{ID} associated with ID and an updated state* st*.*

KeyUp *: This is the key update generation algorithm which takes as input* mpk*,* msk*, the key update time $T \in \mathcal{T}$, the current revocation list RL, and* st*, and outputs the key update ku_T.*

DKG *: This is the probabilistic decryption key algorithm which takes as input* mpk*, sk_{ID}, and ku_T, and outputs a decryption key $\mathsf{dk}_{ID,T}$ or $\perp$ if ID has been revoked.*

Enc *: This is the probabilistic encryption algorithm which takes as input* mpk*, $ID \in \mathcal{I}$, $T \in \mathcal{T}$, and a message $M \in \mathcal{M}$, and outputs a ciphertext* CT*.*

Dec *: This is the deterministic decryption algorithm which takes as input* mpk*, $\mathsf{dk}_{ID,T}$, and* CT*, and outputs M or $\perp$ if* CT *is an invalid ciphertext.*

Revoke *: This is the stateful revocation algorithm which takes as input an identity to be revoked $ID \in \mathcal{I}$, a revocation time $T \in \mathcal{T}$, the current revocation list RL, and a state* st*, and outputs an updated RL.*

Every RIBE scheme should satisfy the following correctness condition: For any $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(\lambda)$, $M \in \mathcal{M}$, all possible state st, and a revocation list RL, if $ID \in \mathcal{I}$ is not revoked on a time $T \in \mathcal{T}$, then for $(\mathsf{sk}_{ID}, \mathsf{st}) \leftarrow \mathsf{PKG}(\mathsf{mpk}, \mathsf{msk}, ID, \mathsf{st})$, $\mathsf{ku}_T \leftarrow \mathsf{KeyUp}(\mathsf{mpk}, \mathsf{msk}, T, RL, \mathsf{st})$, and $\mathsf{dk}_{ID,T} \leftarrow \mathsf{DKG}(\mathsf{mpk}, \mathsf{sk}_{ID}, \mathsf{ku}_T)$,

$$\mathsf{Dec}(\mathsf{mpk}, \mathsf{dk}_{ID,T}, \mathsf{Enc}(\mathsf{mpk}, ID, T, M)) = M \quad \text{holds.}$$

Next, we provide a security definition of RIBE scheme that captures realistic threats including decryption key exposure.

Definition 6 (IND-RID-CPA). *Let $\mathcal{RIBE}$ =* (Setup, PKG, KeyUP, DKG, Enc, Dec, Revoke) *be a RIBE scheme. For an adversary $\mathcal{A}$ define the following experiment:*

[6] Boldyreva et al. [7] define the *selective* security, and Libert-Vergnaud [28] extends it to *adaptive* security.

[7] All DKG algorithms in the previous constructions are deterministic and invertible in the sense that a secret key can be recovered from a corresponding decryption key (and key update), and so previous schemes are vulnerable against decryption key exposure. To prevent such an attack by inversion, we define DKG to be probabilistic.

$\mathbf{Exp}^{IND\text{-}RID\text{-}CPA}_{\mathcal{RIBE},\mathcal{A}}(\lambda, N)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(\lambda, N)$;
$(M_0^*, M_1^*, ID^*, T^*, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{mpk})$ *such that* $|M_0^*| = |M_1^*|$;
$b \xleftarrow{\$} \{0,1\}$;
$\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{mpk}, ID^*, T^*, M_b^*)$;
$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{CT}^*, \mathsf{st})$;
If $b = b'$ *return* 1 *else return* 0.

In the above experiment, $\mathcal{O}$ *is a set of oracles* $\{\mathsf{PKG}(\cdot), \mathsf{KeyUp}(\cdot), \mathsf{Revoke}(\cdot,\cdot), \mathsf{DKG}(\cdot,\cdot)\}$ *defined as follows:*

$\mathsf{PKG}(\cdot)$ *: For* $ID \in \mathcal{I}$*, it returns* sk_{ID} *(by running* $\mathsf{PKG}(\mathsf{mpk}, \mathsf{msk}, ID, \mathsf{st}) \rightarrow \mathsf{sk}_{ID}$*).*

$\mathsf{KeyUp}(\cdot)$ *: For* $T \in \mathcal{T}$*, it returns* ku_T *(by running* $\mathsf{KeyUp}(\mathsf{mpk}, \mathsf{msk}, T, RL, \mathsf{st}) \rightarrow \mathsf{ku}_T$*).*

$\mathsf{Revoke}(\cdot,\cdot)$ *: For* $ID \in \mathcal{I}$ *and* $T \in \mathcal{T}$*, it returns the updated revocation list* RL *(by running* $\mathsf{Revoke}(\mathsf{mpk}, ID, T, RL, \mathsf{st}) \rightarrow RL$*).*

$\mathsf{DKG}(\cdot,\cdot)$ *: For* $ID \in \mathcal{I}$ *and* $T \in \mathcal{T}$*, it returns* $\mathsf{dk}_{ID,T}$ *(by running* $\mathsf{PKG}(\mathsf{mpk}, \mathsf{msk}, ID, \mathsf{st})) \rightarrow \mathsf{sk}_{ID}$ *and* $\mathsf{DKG}(\mathsf{mpk}, \mathsf{sk}_{ID}, \mathsf{ku}_T) \rightarrow \mathsf{dk}_{ID,T}$*).*

$\mathcal{A}$ *is allowed to issue the above oracles with the following restrictions:*[8]

1. $\mathsf{KeyUp}(\cdot)$ *and* $\mathsf{Revoke}(\cdot,\cdot)$ *can be queried on time which is greater than or equal to the time of all previous queries.*
2. $\mathsf{Revoke}(\cdot,\cdot)$ *cannot be queried on time* T *if* $\mathsf{KeyUp}(\cdot)$ *was queried on* T*.*
3. *If* $\mathsf{PKG}(ID^*)$ *was queried, then* $\mathsf{Revoke}(ID^*, T)$ *must be queried for* $T \leq T^*$*.*
4. $\mathsf{DKG}(\cdot,\cdot)$ *cannot be queried on time* T *before* $\mathsf{KeyUp}(\cdot)$ *was queried on* T*.*
5. $\mathsf{DKG}(ID^*, T^*)$ *cannot be queried.*

A RIBE scheme is said to be IND-RID-CPA if for all PPT adversaries $\mathcal{A}$ *and polynomials* N*, the following advantage is negligible in the security parameter* λ*.*

$$Adv^{IND\text{-}RID\text{-}CPA}_{\mathcal{RIBE},\mathcal{A}}(\lambda, N) = \left| \Pr\left[\mathbf{Exp}^{IND\text{-}RID\text{-}CPA}_{\mathcal{RIBE},\mathcal{A}}(\lambda, N) = 1\right] - \frac{1}{2} \right|.$$

3.1 Security Analysis of Previous RIBE Schemes

In this section, we analyze the security of previous RIBE schemes in our security model, which assumes a stronger adversary than Boldyreva et al. adversarial model: our adversary can access the decryption key oracle, which is not given in the Boldyreva et al. model. First, we show that the (simple but non-scalable) BF-RIBE scheme is secure in the new adversarial model. Next, we show that all previous RIBE schemes except for the BF-RIBE are vulnerable against

[8] The fourth and fifth restrictions are the difference between our definition and Boldyreva et al.'s one [7].

decryption key exposure. More precisely, we can construct polynomial time adversaries using decryption key oracles.[9] We briefly explain the BGK-RIBE scheme and the LV-RIBE scheme are vulnerable against decryption key exposure.

Boneh-Franklin RIBE Scheme: To fit the BF-RIBE scheme into our syntax of the RIBE scheme, BF-RIBE can be instantiated with the $IND\text{-}CPA$ secure symmetric encryption scheme. Let $\mathsf{Enc}_{\mathsf{BF}}(\mathsf{mpk}_{\mathsf{BF}}, ID, M)$ and $\mathsf{PKG}_{\mathsf{BF}}(\mathsf{msk}_{\mathsf{BF}}, ID)$ be an encryption algorithm and private key generation algorithm for the BF-IBE scheme, respectively, where $(\mathsf{mpk}_{\mathsf{BF}}, \mathsf{msk}_{\mathsf{BF}})$ is a pair of a master public key and master secret key, ID is a receiver's identity, and M is a message. Let $\mathsf{SE} = (\mathsf{SEnc}, \mathsf{SDec})$ be a symmetric encryption scheme. For each user ID, KGC randomly chooses a secret key of SE and gives it to the user ID as sk_{ID}. The encryption algorithm of the (modified) BF-RIBE scheme is defined as $\mathsf{Enc}(\mathsf{mpk}_{\mathsf{BF}}, ID, T, M) := \mathsf{Enc}_{\mathsf{BF}}(\mathsf{mpk}_{\mathsf{BF}}, (ID, T), M)$. In each time period T, KGC runs $\mathsf{PKG}_{\mathsf{BF}}(\mathsf{msk}_{\mathsf{BF}}, (ID, T)) \rightarrow \mathsf{pvk}_{(ID,T)}$, where $\mathsf{pvk}_{(ID,T)}$ will be a decryption key for ID on time T, that is, $\mathsf{pvk}_{(ID,T)} = \mathsf{dk}_{ID,T}$. Then KGC posts $\mathsf{ku}_T = \{\mathsf{SEnc}_{\mathsf{sk}_{ID}}(\mathsf{dk}_{ID,T}) | ID \text{ is a non-revoked user in time period } T\}$. Then, only non-revoked users can recover $\mathsf{dk}_{ID,T}$. If the BF-IBE scheme is IND-ID-CPA secure and the SE is IND-CPA secure, then the (modified) BF-RIBE scheme is IND-RID-CPA secure, which can easily be proven by using the standard hybrid argument.

Boldyreva-Goyal-Kumar RIBE Scheme: Boldyreva et al. [7] proposed the first scalable but selectively secure RIBE scheme by using Fuzzy IBE [36]. Due to the collusion resistance of the Fuzzy IBE scheme, no revoked user can compute its decryption key. The user's decryption keys are associated with two attributes: identity ID and time period T. The decryption key is split into two components corresponding to ID and T. A secret key sk_{ID} is associated with ID and key update ku_T is associated with T. The DKG algorithm is only to put parts of sk_{ID} and ku_T together.[10] More concretely, the PKG algorithm returns $\{(x, D_x, d_x)\}_{x \in \mathsf{I}}$ which is a private key of a user who is assigned to a leaf node η, and the KeyUp algorithm returns $\{(y, E_y, e_y)\}_{y \in \mathsf{J}}$. If a user is not revoked on T, then there exist x where $x \in \mathsf{I} \cap \mathsf{J}$. The DKG algorithm finds such a x, and returns (D_x, E_x, d_x, e_x) which is a decryption key $\mathsf{dk}_{ID,T}$ of this user on time T. Therefore, an adversary that has $\mathsf{dk}_{ID^*,T}$ and ku_T can always recover a part (D_{x^*}, d_{x^*}) of sk_{ID^*} for some x^* if ID^* is not revoked in time T, and can always compute $\mathsf{dk}_{ID^*,T^*} = (D_{x^*}, E_{x^*}, d_{x^*}, e_{x^*})$ from the parts (D_{x^*}, d_{x^*}) of sk_{ID^*} and (E_{x^*}, e_{x^*}) of ku_{T^*} if ID^* is still not revoked in the challenge time T^*.

[9] As we mentioned in the introduction, a goal of this subsection is to not contradict the security proofs of previous RIBE schemes. Our attack is positioned outside of their security models.

[10] Chen et al. [13] proposed an RIBE scheme based on lattices by applying the Agrawal et al. lattice-based IBE [1]. They used the same methodology as that of BGK-RIBE, where a private key itself is contained in a corresponding decryption key. Therefore, the same attack works.

Libert-Vergnaud RIBE Scheme: Libert and Vergnaud [28] proposed the first adaptively secure RIBE scheme without random oracles. In the LV-RIBE scheme, the process of $\mathsf{DKG}(\mathsf{mpk}, \mathsf{sk}_{ID}, \mathsf{ku}_T)$ is component-wise multiplications or additions between sk_{ID} and ku_T. Since ku_T is public information, if an adversary obtains a decryption key $\mathsf{dk}_{ID^*,T}$, where $T \neq T^*$, it can then recover sk_{ID} in polynomial-time by performing the inverse process of DKG, that is, divisions or subtractions. More concretely, the PKG algorithm returns $\mathsf{sk}_{ID} = \{(i, d_{ID,i})\}_{i \in \mathsf{I}}$, where $d_{ID,i} := (d_{1,i}, d_{2,i}, r_{1,i})$, and the KeyUp algorithm returns $\mathsf{ku}_T = \{\mathsf{ku}_{T,j}\}_{j \in \mathsf{J}}$, where $\mathsf{ku}_{T,j} := (\mathsf{ku}_{1,j}, \mathsf{ku}_{2,j}, r_{2,j})$. The DKG algorithm parses $\mathsf{sk}_{ID} = \{(i, d_{ID,i})\}_{i \in \mathtt{I}}$ and $\mathsf{ku}_T = \{(j, \mathsf{ku}_{T,j}\}_{j \in \mathtt{J}}$. If there is no pair $(i, j) \in \mathtt{I} \times \mathtt{J}$ such that $i = j$, then return $\perp$. Otherwise, choose such pair $i = j$. Return $\mathsf{dk}_{ID,T} = (d_{T,1}, d_{T,2}, d_{T,3}, d_{T,4}) = (d_{1,i} \cdot \mathsf{ku}_{1,i}, d_{2,i}, \mathsf{ku}_{2,i}, r_{1,i} + r_{2,i})$. Therefore, anyone can easily compute $(d_{1,i}, d_{2,i}, r_{1,i}) \in \mathsf{sk}_{ID}$ from "both" $\mathsf{dk}_{ID,T}$ and $\mathsf{ku}_{T,i}$ such that $d_{1,i} = d_{T,1}/\mathsf{ku}_{1,i}$ and $r_{1,i} = d_{T,4} - r_{2,i}$. Moreover, $d_{2,i}$ is directly contained in $\mathsf{dk}_{ID,T}$.

One may expect that we can impede this attack by adding a randomization process in DKG, but it does not seem easy to prove the security of such a modification of the LV-RIBE scheme. LV-RIBE scheme is based on a variant of the Waters IBE (LV-IBE) scheme proposed in [29]. The security strategy of the LV-IBE is somewhat similar to the Gentry IBE [19] such that the simulator can compute a private key for any identity, even the challenge identity, in the proof. The simulator can generate a private key using "fixed" randomness and this fixed randomness is also used in making the challenge ciphertext.[11] Therefore, in the adversarial view, the challenge ciphertext is uniformly generated since it cannot obtain the corresponding private key of the challenge ciphertext. Since the LV-RIBE is based on the LV-IBE, the LV-RIBE does not support full re-randomization of the decryption key. Therefore, decryption key exposure reveals randomness used in the secret key. Since ku_{T^*} is public, if dk_{ID^*,T^*} is leaked, the randomness of sk_{ID^*} will be also leaked. (Not all parts of the randomness, but fixed randomness, which is essentially used in the security proof, will be leaked.) As mentioned, the simulation of this type of IBE scheme, such as the LV-IBE and the Gentry IBE scheme, succeeds only when the fixed randomness of the private key is hidden from the adversary's view. Therefore, we cannot construct a simulator for the LV-RIBE scheme when we directly follow the same strategy used in the LV-IBE scheme.

4 Our Construction

In this section, we propose an RIBE scheme. For the revocation process, we basically follow previous RIBE schemes' strategy using a binary tree structure; that is, to reduce the key update costs, we apply a binary tree structure and

[11] The term "fixed" means that the simulator can generate only one private key using fixed randomness per each identity after publishing mpk. A decryption key may have other flexible randomness, but at least a part of the randomness should be fixed according to the identity.

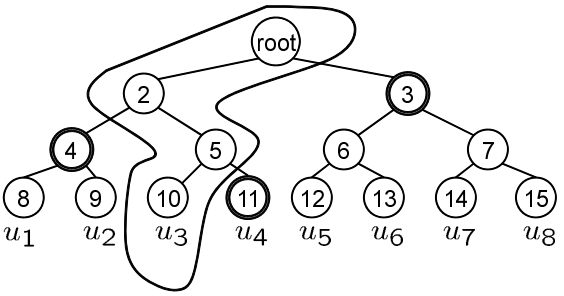


Fig. 1. Example of KUNode

define the KUNode algorithm. In the actual schemes, this algorithm is used in a black-box manner.

4.1 KUNode Algorithm

We introduce the KUNode algorithm and Boldyreva et al.'s idea for efficient revocation.

Definition 7 (KUNode **Algorithm [7]).** *This algorithm takes as input a binary tree* BT*, revocation list* RL*, and time* T*, and outputs a set of nodes. A formal description of this algorithm is as follows: If* η *is a non-leaf node, then* η_{left} *and* η_{right} *denote the left and right child of* η*, respectively. Each user is assigned to a leaf node. If a user (assigned to* η*) is revoked on time* T*, then* $(\eta, T) \in RL$*.* Path(η) *denotes the set of nodes on the path from* η *to* root*. The description of* KUNode *is given below.*

KUNode(BT, RL, T) :
  X, Y $\leftarrow \emptyset$;
  $\forall(\eta_i, T_i) \in RL$
    If $T_i \leq T$ *then add* Path(η_i) *to* X
  $\forall x \in$ X
    If $x_{left} \notin$ X *then add* x_{left} *to* Y
    If $x_{right} \notin$ X *then add* x_{right} *to* Y
  If Y $= \emptyset$ *then add* root *to* Y
  Return Y

Figure 1 gives a simple example to help the readers easily understand KUNode(BT, RL, T). In the example, let a user u_3 (assigned to x_{10}) be revoked.

Then, $X = \mathsf{Path}(x_{10}) = \{x_{10}, x_5, x_2, \mathsf{root} = x_1\}$, and $Y = \{x_3, x_4, x_{11}\}$. Intuitively, all users, except u_3, have a node $x \in Y$ that is contained in the set of nodes on the path from their assigned node to `root`: e.g., x_4 for u_1 and u_2, x_{11} for u_4, and x_3 for u_5, u_6, u_7, and u_8, whereas $Y \cap \mathsf{Path}(x_{10}) = \emptyset$.

When a user joins the system, KGC assigns it to the leaf node η of a complete binary tree, and issues a set of keys, wherein each key is associated with each node on $\mathsf{Path}(\eta)$. At time period T, KGC publishes key updates for a set $\mathsf{KUNode}(\mathtt{BT}, RL, T)$. Then, only non-revoked users have at least one key corresponding to a node in $\mathsf{KUNode}(\mathtt{BT}, RL, T)$ and are able to generate decryption keys on time T.

4.2 Our Construction

For a simple description of our RIBE scheme, we use notation F_{Wat} and F_{BB} to denote the respective hash functions used in the Waters IBE scheme and Boneh-Boyen IBE scheme. More precisely, for an identity space $\mathcal{I}$ and time space $\mathcal{T}$, define $F_{\mathsf{Wat}} : \mathcal{I} \to \mathbb{G}$ and $F_{\mathsf{BB}} : \mathcal{T} \to \mathbb{G}$ by

$$F_{\mathsf{Wat}}(ID) = u' \prod_{i=1}^{n} u_i^{b_i} \text{ and } F_{\mathsf{BB}}(T) = v' v^T, \text{ respectively,}$$

where $ID = (b_1, b_2, \ldots, b_n) \in \{0,1\}^n$.

Before describing our construction, we will explain the intuition behind it. As mentioned, we need a different approach to achieve an (adaptively secure) RIBE scheme with decryption key exposure. To this end, we begin with a simple two-level HIBE scheme (without delegating property). More precisely, the first level is assigned for identity and the second level is assigned for the time period. Since we consider only polynomially bounded time (as all previous RIBE schemes), we combine the adaptively secure Waters IBE scheme (for the first level) and the selectively secure Boneh-Boyen IBE scheme (for the second level). The decryption key of our RIBE scheme is exact second level secret key of the HIBE scheme, that is,

$$\mathsf{dk}_{ID,T} = (g_2^{\alpha} F_{\mathsf{Wat}}(ID)^r F_{\mathsf{BB}}(T)^s,\ g^r,\ g^s).$$

The above decryption key allows user to re-randomize r and s in the exponent without knowing master key g_2^{α}.[12] Decryption key exposure will then not be helpful for the adversary in obtaining information about the challenge ciphertext since this combined two-level hierarchical extension can be considered as a secure HIBE scheme (for exponentially many identities and polynomially many time periods) in the sense that it has resistance against collusion attacks. To generate secret keys and key updates, we use a technique similar to that used in [28].

[12] As mentioned in section 3.1, the LV-RIBE does not support such re-randomization process in decryption keys, and this is the essential difference between ours and previous schemes.

The master secret key is randomly divided into two parts, which are respectively contained in the secret key and key updates, that is,

$$(g_\theta^\alpha F_{\mathsf{Wat}}(ID)^{r_\theta},\ g^{r_\theta}) \in \mathsf{sk}_{ID} \text{ and } (\tilde{g}_\theta^\alpha F_{\mathsf{BB}}(T)^{s_\theta},\ g^{s_\theta}) \in \mathsf{ku}_T,$$

where $g_\theta \cdot \tilde{g}_\theta = g_2$. Therefore, if the adversary cannot obtain both the secret key and key updates, which will contribute to computing the target decryption key dk_{ID^*,T^*}, then in the security proof we can simulate either the secret key or key update. (We can assume that the part not given to the adversary contains information about the master key and the other part is a random element.)

Even if the above intuition explains the decryption key exposure resistance of a combination of the Waters IBE and the Boneh-Boyen IBE, we need an additional technique to circumvent the difficulty pointed out in [28]. The difficulty occurs when the adversary issues a secret key query for the target identity ID^*. For each node θ in the binary tree, a random value g_θ is assigned. Whenever PKG is run, the identity ID is assigned in the leaf node and the value g_θ on the path to the root node is used in the secret key sk_{ID}. In the security proof, whenever a secret key query or key update query regarding ID is issued, the simulator should decide which of two shares g_θ^α and $\tilde{g}_\theta^\alpha$ will contain the master secret key, where θ is on the path to the root of tree; that is, one share is $g_2^\alpha S_\theta$ and the other is S_θ^{-1} for random group element S_θ. However, when the target identity ID^* is unknown and has yet to be assigned a leaf, the other path regarding the different identity ID may or may not have connection with the path regarding ID^* (except the root node). To address this issue, Libert and Vergnaud used a variant of the Waters IBE wherein the simulator can generate at least one valid decryption key for each identity and can answer queries regardless of whether nodes are on the path from ID^* to the root node. However, our construction does not support a simulation strategy such as is used in [28, 29] since the original Waters IBE scheme uses a different proof strategy, called *partitioning*, in the security proof. Instead, we carefully deal with the method to assign identity into the tree. In our RIBE scheme, whenever a new identity joins the system, KGC assigns a random leaf node among the undefined nodes.[13] In the security proof, this simple random node assignment technique allows the simulator to pre-assign a random leaf node for the target identity (therefore nodes on the path to the root node are also pre-determined) and to simulate for a secret key, decryption key, and key update queries before receiving queries regarding ID^*. When the first query regarding ID^* is issued, the simulator can use the pre-assigned leaf node for ID^*. We can show that this simulation for node assignment is identically distributed to that in the real protocol.

We describe the proposed RIBE scheme below.

$\mathsf{Setup}(\lambda, N)$: Randomly choose $g, g_2, u', u_1, \ldots, u_n, v', v \xleftarrow{\$} \mathbb{G}$ and $\alpha \xleftarrow{\$} \mathbb{Z}_p$. Set $\mathsf{mpk} = \{g, g_1 = g^\alpha, g_2, u', u_1, \ldots, u_n, v', v\}$, $\mathsf{msk} = \{g_2^\alpha\}$, $RL = \emptyset$, and $\mathsf{st} = \mathsf{BT}$, where BT is a binary tree with N leaves.

[13] KGC can use a pseudorandom generator for this process.

$\mathsf{PKG}(\mathsf{mpk}, \mathsf{msk}, ID, \mathsf{st})$: Randomly choose an unassigned leaf η from $\mathtt{BT}$, and store ID in the node η. For each node $\theta \in \mathsf{Path}(\eta)$,

1. Recall g_θ if it was defined. Otherwise, $g_\theta \xleftarrow{\$} \mathbb{G}$ and store $(g_\theta, \tilde{g}_\theta = g_2/g_\theta)$ in the node θ.[14]
2. Choose $r_\theta \xleftarrow{\$} \mathbb{Z}_p$.
3. Compute $(D_{\theta,0}, D_{\theta,1}) := (g_\theta^\alpha F_{\mathsf{Wat}}(ID)^{r_\theta}, g^{r_\theta})$.

Return $\mathsf{sk}_{ID} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_{\theta \in \mathsf{Path}(\eta)}$.

$\mathsf{KeyUp}(\mathsf{mpk}, \mathsf{msk}, T, RL, \mathsf{st})$: Parse $\mathsf{st} = \mathtt{BT}$. For each node $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)$,

1. Retrieve $\tilde{g}_\theta$ (note that $\tilde{g}_\theta$ is always pre-defined in the PKG algorithm).
2. Choose $s_\theta \xleftarrow{\$} \mathbb{Z}_p$.
3. Compute $(\tilde{D}_{\theta,0}, \tilde{D}_{\theta,1}) := (\tilde{g}_\theta^\alpha F_{\mathsf{BB}}(T)^{s_\theta}, g^{s_\theta})$.

Return $\mathsf{ku}_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)}$.

$\mathsf{DKG}(\mathsf{mpk}, \mathsf{sk}_{ID}, \mathsf{ku}_T)$: Parse $\mathsf{sk}_{ID} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_{\theta \in \mathtt{I}}$ and $\mathsf{ku}_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathtt{J}}$. If $\mathtt{I} \cap \mathtt{J} = \emptyset$, then return $\perp$. Otherwise, choose $\theta \in \mathtt{I} \cap \mathtt{J}$ and $r, s \xleftarrow{\$} \mathbb{Z}_p$ and return

$$\mathsf{dk}_{ID,T} = (D_{\theta,0} \cdot \tilde{D}_{\theta,0} \cdot F_{\mathsf{Wat}}(ID)^r \cdot F_{\mathsf{BB}}(T)^s, \quad D_{\theta,1} \cdot g^r, \quad \tilde{D}_{\theta,1} \cdot g^s).$$

$\mathsf{Enc}(\mathsf{mpk}, ID, T, M)$: Choose a random integer $t \xleftarrow{\$} \mathbb{Z}_p$ and return

$$\mathsf{CT} = (M \cdot e(g_1, g_2)^t, \quad g^{-t}, \quad F_{\mathsf{Wat}}(ID)^t, \quad F_{\mathsf{BB}}(T)^t).$$

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{dk}_{ID,T}, \mathsf{CT})$: Parse $\mathsf{CT} = (C_0, C_1, C_2, C_3)$ and $\mathsf{dk}_{ID,T} = (D_1, D_2, D_3)$ and return

$$C_0 \prod_{i=1}^{3} e(C_i, D_i).$$

$\mathsf{Revoke}(\mathsf{mpk}, ID, T, RL, \mathsf{st})$: Let η be the leaf node associated with ID. Update the revocation list by $RL \leftarrow RL \cup \{(\eta, T)\}$ and return the updated revocation list.

We should check the correctness of our scheme: Assume that mpk, msk, $\mathcal{M}$, $\mathcal{I}$, $\mathcal{T}$, st, and RL are normally generated and fixed. Moreover, assume that $\mathsf{sk}_{ID} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_{\theta \in \mathsf{Path}(\eta)}$ is a secret key of a non-revoked user ID on time T and $\mathsf{ku}_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)}$. Then, for some $\theta \in \mathsf{Path}(\eta) \cap \mathsf{KUNode}(\mathtt{BT}, RL, T)$, DKG should output

$$\begin{aligned}\mathsf{dk}_{ID,T} &= (D_{\theta,0} \cdot \tilde{D}_{\theta,0} \cdot F_{\mathsf{Wat}}(ID)^r \cdot F_{\mathsf{BB}}(T)^s, \quad D_{\theta,1} \cdot g^r, \quad \tilde{D}_{\theta,1} \cdot g^s) \\ &= (g_2^\alpha F_{\mathsf{Wat}}(ID)^{r_\theta + r} F_{\mathsf{BB}}(T)^{s_\theta + s}, \quad g^{r_\theta + r}, \quad g^{s_\theta + s}).\end{aligned}$$

For an encryption of M, $\mathsf{CT} = (M \cdot e(g_1, g_2)^t, \quad g^{-t}, \quad F_{\mathsf{Wat}}(ID)^t, \quad F_{\mathsf{BB}}(T)^t)$,

$$\begin{aligned}&\mathsf{Dec}(\mathsf{mpk}, \mathsf{dk}_{ID,T}, \mathsf{CT}) \\ &= M \cdot e(g_1, g_2)^t e(g^{-t}, g_2^\alpha F_{\mathsf{Wat}}(ID)^{r_\theta + r} F_{\mathsf{BB}}(T)^{s_\theta + s}) e(F_{\mathsf{Wat}}(ID)^t, g^{r_\theta + r}) \\ &\quad \cdot e(F_{\mathsf{BB}}(T)^t, g^{s_\theta + s}) \\ &= M.\end{aligned}$$

[14] As in the Libert-Vergnaud scheme, KGC can use a pseudorandom generator instead of storing g_θ.

We provide a (polynomial-time) reduction to the Waters IBE scheme, which is a non-revocable IBE secure under the DBDH assumption. Therefore, our scheme is secure under the DBDH assumption.

Theorem 2. *If there exists an adversary $\mathcal{A}$ attacking IND-RID-CPA security of the proposed RIBE scheme, then there exists another adversary $\mathcal{B}$ breaking IND-ID-CPA security of the Waters IBE scheme.*

Because of space constraints, we relegate the proof of Theorem 2 in the full version. Note that the reduction loss in our security proof is $2q|\mathcal{T}|$. Since the security proof to show that the Waters IBE scheme is secure under DBDH assumption losses $O(nq)$ [38], our RIBE scheme is secure under DBDH assumption with $O(nq^2|\mathcal{T}|)$ reduction loss. Although our proof is loose, we note that the previous adaptively secure LV-RIBE scheme lose the same factor $O(nq^2|\mathcal{T}|)$ in the security proof.

5 Discussion

In this section, we discuss several issues related to RIBE schemes.

Short Public Parameters: In high level explanation, our technique is to add the revocation capability to the IBE scheme without sacrificing efficiency, and we show that the underlying IBE scheme supporting key re-randomization can be provably secure against decryption key exposure. In fact, we essentially used the Water IBE as the underlying IBE scheme of our RIBE construction. Therefore, we may construct an RIBE scheme from other IBE schemes. We expect that a scalable RIBE scheme with decryption key exposure resilience can be constructed from the Lewko-Waters IBE [25], which supports key re-randomization. Then, we can reduce the size of public parameter, though we need to use composite-order bilinear groups and other complexity assumptions. Note that Lewko have shown a Lewko-Waters IBE scheme under the prime-order group setting [24], however, this scheme does not support key-rerandomization. It would be interesting to construct a RIBE scheme with decryption key exposure resilience and short public parameters in the prime-order group setting.

Better Efficiency from Random Oracle Heuristic: In our construction, we used two level hierarchical construction by combining the Water IBE and the Boneh-Boyen IBE, where both schemes are secure in the standard model. Both schemes use hash functions F_{Wat} and F_{BB}, respectively. The role of both hash functions is to apply *partitioning* technique; that is, in the security proof, the simulator divides the domain of hash functions into two subsets, one for the challenge query and the other for key extraction queries. By changing these two hash functions into cryptographic hash functions that are modeled as random oracles, we can achieve better efficiency since the random oracles allow such a partitioning technique but require low computational cost. Furthermore, by using one more hash function and standard techniques for random oracle model schemes, we can reduce the security of the random oracle model RIBE scheme to the BDH problem.

Revocable Identity-Based Signature (RIBS): We basically used two level hierarchical construction and it is not difficult to extend our construction to three level hierarchical construction (but revocation capability is allowed only for the first level). Therefore, we can apply the well-known Naor transformation from an IBE scheme to a signature scheme. More precisely, from the three level hierarchical construction, we can obtain a scalable identity-based signature scheme, where the first level is for identity, the second level is for time period, and the third level is for message. For better efficiency, we can apply the same transformation used in the previous paragraph to here, and then obtain an efficent RIBS scheme in the random oracle model.

Chosen Ciphertext Security: Due to the property of the underlying Waters IBE scheme, we can extend our RIBE scheme to a HIBE scheme with efficient revocation only for the first level users. There is a well-known transformation from a two-level HIBE scheme to a CCA-secure IBE scheme [9]. Therefore, we can obtain CCA-secure RIBE scheme by applying this transformation.

6 Conclusion

We revisited both the security model and construction methodology for RIBE schemes. First, we pointed out a gap between the Boldyreva et al. security model and the trivial but non-scalable BF-RIBE construction. We introduced a new security model for RIBE scheme by capturing realistic threat, called decryption key exposure, and proposed the first scalable RIBE construction in the new security model.

There are several interesting remaining problems. From a theoretical point of view, one natural question is how to construct a generic transformation from IBE to RIBE. In the practice, revocation is a necessary functionality in the public key encryption schemes. Therefore, finding efficient revocation methods in other encryption schemes such as (hierarchical) inner-product encryption [35] and attribute-based encryption [36] are also important. In this paper, we only focused on the pairing-based schemes, but it is interesting to construct schemes based on other mathematical structure such as lattice that are secure in our security model. Recently, a revocable hierarchical IBE (RHIBE) scheme is proposed, but its security is proven only in the weaker security notion, selective security [37]. Achieving full security in RHIBE construction is a direct open problem. To the best of our knowledge, all scalable RIBE use the *Complete Subtree* [32] method for revocation capability. Therefore, it is interesting to combine IBE with different revocation methods such as the *Subset Difference* [32].

Acknowledgements. We thank anonymous reviewers of PKC 2013 and members of Shin-Akarui-Angou-Benkyou-Kai for their helpful comments.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient Lattice (H)IBE in the Standard Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)
2. Aiello, W., Lodha, S., Ostrovsky, R.: Fast Digital Identity Revocation (Extended Abstract). In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 137–152. Springer, Heidelberg (1998)
3. Attrapadung, N., Imai, H.: Attribute-Based Encryption Supporting Direct/Indirect Revocation Modes. In: Parker, M.G. (ed.) Cryptography and Coding 2009. LNCS, vol. 5921, pp. 278–300. Springer, Heidelberg (2009)
4. 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)
5. Baek, J., Zheng, Y.: Identity-Based Threshold Decryption. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 262–276. Springer, Heidelberg (2004)
6. Bellare, M., Palacio, A.: Protecting against key exposure: strongly key-insulated encryption with optimal threshold. IACR Cryptology ePrint Archive 2002:064 (2002)
7. Boldyreva, A., Goyal, V., Kumar, V.: Identity-based encryption with efficient revocation. In: ACM CCS 2008, pp. 417–426 (2008)
8. Boneh, D., Boyen, X.: Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
9. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. SIAM J. Comput. 36(5), 1301–1328 (2007)
10. Boneh, D., Ding, X., Tsudik, G., Wong, C.-M.: A method for fast revocation of public key certificates and security capabilities. In: USENIX Security Symposium 2001. USENIX (2001)
11. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. SIAM J. Comput. 32(3), 586–615 (2003)
12. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. Journal of Cryptology 20, 265–294 (2007)
13. Chen, J., Lim, H.W., Ling, S., Wang, H., Nguyen, K.: Revocable Identity-Based Encryption from Lattices. In: Susilo, W., Mu, Y., Seberry, J. (eds.) ACISP 2012. LNCS, vol. 7372, pp. 390–403. Springer, Heidelberg (2012)
14. Ding, X., Tsudik, G.: Simple Identity-Based Cryptography with Mediated RSA. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 193–210. Springer, Heidelberg (2003)
15. Dodis, Y., Franklin, M., Katz, J., Miyaji, A., Yung, M.: A Generic Construction for Intrusion-Resilient Public-Key Encryption. In: Okamoto, T. (ed.) CT-RSA 2004. LNCS, vol. 2964, pp. 81–98. Springer, Heidelberg (2004)
16. Dodis, Y., Katz, J., Xu, S., Yung, M.: Key-Insulated Public Key Cryptosystems. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 65–82. Springer, Heidelberg (2002)
17. Elwailly, F.F., Gentry, C., Ramzan, Z.: QuasiModo: Efficient Certificate Validation and Revocation. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 375–388. Springer, Heidelberg (2004)
18. Gentry, C.: Certificate-Based Encryption and the Certificate Revocation Problem. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 272–293. Springer, Heidelberg (2003)

19. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
20. Goyal, V.: Certificate Revocation Using Fine Grained Certificate Space Partitioning. In: Dietrich, S., Dhamija, R. (eds.) FC 2007 and USEC 2007. LNCS, vol. 4886, pp. 247–259. Springer, Heidelberg (2007)
21. Hanaoka, G., Weng, J.: Generic Constructions of Parallel Key-Insulated Encryption. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 36–53. Springer, Heidelberg (2010)
22. Hanaoka, Y., Hanaoka, G., Shikata, J., Imai, H.: Identity-Based Hierarchical Strongly Key-Insulated Encryption and Its Application. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 495–514. Springer, Heidelberg (2005)
23. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
24. Lewko, A.: Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 318–335. Springer, Heidelberg (2012)
25. Lewko, A., Waters, B.: New Techniques for Dual System Encryption and Fully Secure HIBE with Short Ciphertexts. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 455–479. Springer, Heidelberg (2010)
26. Libert, B., Quisquater, J.-J.: Efficient revocation and threshold pairing based cryptosystems. In: PODC 2003, pp. 163–171. ACM (2003)
27. Libert, B., Quisquater, J.-J., Yung, M.: Parallel Key-Insulated Public Key Encryption Without Random Oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 298–314. Springer, Heidelberg (2007)
28. Libert, B., Vergnaud, D.: Adaptive-ID Secure Revocable Identity-Based Encryption. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 1–15. Springer, Heidelberg (2009)
29. Libert, B., Vergnaud, D.: Towards Black-Box Accountable Authority IBE with Short Ciphertexts and Private Keys. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 235–255. Springer, Heidelberg (2009)
30. Micali, S.: Efficient certificate revocation. Technical Report MIT/LCS/TM-542b (1996)
31. Micali, S.: Novomodo: Scalable certificate validation and simplified PKI management. In: PKI Research Workshop (2002)
32. 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)
33. Naor, M., Nissim, K.: Certificate revocation and certificate update. IEEE Journal on Selected Areas in Communications 18, 561–570 (2000)
34. González-Nieto, J.M., Manulis, M., Sun, D.: Fully Private Revocable Predicate Encryption. In: Susilo, W., Mu, Y., Seberry, J. (eds.) ACISP 2012. LNCS, vol. 7372, pp. 350–363. Springer, Heidelberg (2012)
35. Okamoto, T., Takashima, K.: Hierarchical Predicate Encryption for Inner-Products. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 214–231. Springer, Heidelberg (2009)
36. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)

37. Seo, J.H., Emura, K.: Efficient Delegation of Key Generation and Revocation Functionalities in Identity-Based Encryption. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 343–358. Springer, Heidelberg (2013)
38. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
39. Weng, J., Liu, S., Chen, K., Ma, C.: Identity-Based Parallel Key-Insulated Encryption Without Random Oracles: Security Notions and Construction. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. LNCS, vol. 4329, pp. 409–423. Springer, Heidelberg (2006)
40. Weng, J., Liu, S., Chen, K., Zheng, D., Qiu, W.: Identity-Based Threshold Key-Insulated Encryption without Random Oracles. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 203–220. Springer, Heidelberg (2008)

Improved (Hierarchical) Inner-Product Encryption from Lattices

Keita Xagawa

NTT Secure Platform Laboratories
xagawa.keita@lab.ntt.co.jp

Abstract. Inner-product encryption (IPE) provides fine-grained access control and has attractive applications. Agrawal, Freeman, and Vaikuntanathan (Asiacrypt 2011) proposed the first IPE scheme from lattices by twisting the identity-based encryption (IBE) scheme by Agrawal, Boneh, and Boyen (Eurocrypt 2010). Their IPE scheme supports inner-product predicates over R^μ, where the ring is $R = \mathbb{Z}_q$. Several applications require the ring R to be exponentially large and, thus, they set $q = 2^{O(n)}$ to implement such applications. This choice results in the AFV IPE scheme with public parameters of size $O(\mu n^2 \lg^3 q) = O(\mu n^5)$ and ciphertexts of size $O(\mu n \lg^3 q) = O(\mu n^4)$, where n is the security parameter. Hence, this makes the scheme impractical, as they noted.

We address this efficiency issue by "untwisting" their twist and providing another twist. Our scheme supports inner-product predicates over R^μ where $R = \mathrm{GF}(q^n)$ instead of $\mathbb{Z}_q$. Our scheme has public parameters of size $O(\mu n^2 \lg^2 q)$ and ciphertexts of size $O(\mu n \lg^2 q)$. Since the cardinality of $\mathrm{GF}(q^n)$ is inherently exponential in n, we have no need to set q as the exponential size for applications.

As side contributions, we extend our IPE scheme to a hierarchical IPE (HIPE) scheme and propose a fuzzy IBE scheme from IPE. Our HIPE scheme is more efficient than that developed by Abdalla, De Caro, and Mochetti (Latincrypt 2012). Our fuzzy IBE is secure under a much weaker assumption than that employed by Agrawal et al. (PKC 2012), who constructed the first lattice-based fuzzy IBE scheme.

Keywords: predicate encryption, (hierarchical) inner-product encryption, lattices, learning with errors, full-rank difference encoding, pseudo-commutativity.

1 Introduction

Background: Predicate encryption (PE) gives fine-grained access control beyond identity-based encryption (IBE). In a PE scheme, a receiver corresponding to a *key attribute* v can decrypt a ciphertext corresponding to a *ciphertext attribute* w if and only if $\mathcal{P}(v, w) = 1$, where $\mathcal{P}$ is a predicate.

Katz, Sahai, and Waters [30] introduced inner-product encryption (IPE), which is PE that supports the inner-product predicate: that is, predicate $\mathcal{P}^{\mathsf{IPE}} : R^\mu \times R^\mu \to \{0, 1\}$, where R is a finite ring, defined as $\mathcal{P}^{\mathsf{IPE}}(\vec{v}, \vec{w}) = 1$ if and only if $\vec{w}^\top \vec{v} = 0$. They showed that several predicates, for example, equalities, hidden-vector predicates, polynomial evaluations, and CNF/DNF formulae, can be encoded as an inner product that exemplifies the serviceability of IPE. Following their work and by exploiting the

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 235–252, 2013.

properties of the pairing on composite-number or prime order groups, recent studies on IPE have enhanced security or introduced compact schemes [30,36,31,37,10,40,38,39], and have left an open problem of constructing IPE from other assumptions, say, factoring, decisional Diffie-Hellman (DDH), or the learning with errors (LWE) assumptions.

In 2011, Agrawal, Freeman, and Vaikuntanathan [6] overcame the hurdle, i.e., the problem of constructing IPE *without pairing*. They proposed the first IPE scheme based on the LWE assumption [46] and left three open problems: improving security, efficiency, and functionality.

Let us focus on the second problem, the efficiency issue. Their scheme supports an inner-product predicate over $R = \mathbb{Z}_q$, and has public parameters of size $\Theta(\mu n^2 \lg^3 q)$ and ciphertexts of size $\Theta(\mu n \lg^3 q + \ell \lg q)$, where n is the security parameter, μ is the dimension of the vector space, and ℓ is the length of a message. (In what follows, we will ignore $\Theta(\ell \lg q)$.) This seems satisfactory for actual use.

In several applications of IPE, we require exponentially large R (see below). To implement such applications, Agrawal et al. set $q = 2^{O(n)}$ [6, Section 6]. This setting results in the length of ciphertext $\Theta(\mu n^4)$, which shows the impracticality of the scheme in the real world.

Motivated by applications: We were motivated to improve the efficiency by applications of IPE that require large R, which we discuss here. Roughly speaking, we require the ring R to be exponentially large in order to implement an application when we have to take AND (logical conjunction) of predicates by using the technique proposed by Katz, Sahai, and Waters [30][1]. Since the existing pairing-based IPE serves inner products over the ring $R = \mathbb{Z}_q$ or $\mathbb{Z}_N$, where q, N is an exponential of a security parameter, there are no problematic issues. Unfortunately, the exponential magnitude of R makes the AFV IPE scheme impractical, since the length of the ciphertext is the cubic order of $\lg(\#R) = \lg q$.

We have several attractive applications that are implemented by IPE with logical conjunctions. These include CNF formulae [30], hidden vector encryption [15], which serves a comparison and a range query on a small set, and wild-carded IBE [1]. We will review and discuss the applications of IPE schemes in the full version.

Moreover, for a realistic scenario, we will treat a colossal set as a domain of the predicate, e.g., one billion users ($10^9 \approx 2^{30}$), addresses of IPv6 (2^{128}), verification keys of one-time signature (2^{128}–), and hash values of SHA3 candidates (2^{256}–2^{512}). In such a situation, even the equality predicate requires logical conjunctions to split them into chunks in R.

Hence, we should make IPE efficient even for exponentially large R for the IPE applications.

[1] Suppose that we have two implementations of two predicates f and g; one is embedded as $\vec{v}_f$ and $\vec{w}_f$, and the other is embedded as $\vec{v}_g$ and $\vec{w}_g$. The Katz-Sahai-Waters (KSW) technique embeds $f \wedge g$ into two vectors $\vec{v}_{f\wedge g} = (\vec{v}_f, \vec{w}_g)$ and $\vec{w}_{f\wedge g} = (r_f \vec{v}_f, r_g \vec{w}_g)$, where r_f, r_g are chosen uniformly at random from R. The inner product of $\vec{v}_{f\wedge g}$ and $\vec{w}_{f\wedge g}$ is $r_f \vec{w}_f^{\top} \vec{v}_f + r_g \vec{w}_g^{\top} \vec{v}_g$. If the two inner products are 0, that is, two predicates f and g are true, then the inner product becomes 0. The inversion is not true; if not, then the inner product is not 0 without probability, say, $1/\#R$. This shows that R should be exponentially large, say, at least 2^{80} from the security requirement.

1.1 Our Contribution

Our main contribution is to improve the efficiency of the AFV IPE scheme. More formally, we construct an IPE scheme under the LWE assumption, which supports an inner-product predicate over the field $\mathrm{GF}(q^n)$ instead of $\mathbb{Z}_q$ and has public parameters of size $\Theta(\mu n^2 \lg^2 q)$ and ciphertexts of size $\Theta(\mu n \lg^2 q + \ell \lg q)$. Since the cardinality of $\mathrm{GF}(q^n)$ is $q^n = 2^{\Omega(n)}$, and $\mathrm{GF}(q^n)$ is a field, we can set $q = \mathrm{poly}(n)$ even for the above applications. We note that Agrawal et al. [6, Section 6] expected the *ring-LWE* assumption [32] to resolve the issue, but we solve it *without the ring-LWE assumption*.

In addition, we have two side contributions; One is an extension of *hierarchical inner-product encryption* (HIPE) [36], which implies spatial encryption (SE) [14,29,21]. We apply our techniques to again drastically improve the existing HIPE scheme from lattices [3] in the case of exponentially large R. The other is a fuzzy IBE (FIBE) scheme over a small universe $\{0,1\}$ from IPE under the LWE assumption with *conservative* parameters, whereas the existing fuzzy IBE scheme from lattices are under the LWE assumption with *sub-exponential* parameters [5].

Comparison: Since the description size of the public parameters is n times that of ciphertexts, we compare the efficiency of the schemes by the length of the ciphertext. For simplicity, we let L_{ours} and L_{AFV} denote the lengths of ciphertexts of our scheme and the AFV scheme, respectively.

When $q = \mathrm{poly}(n)$, our scheme improves the size by only a factor of $\lg q = O(\lg n)$ ($L_{\mathrm{ours}} = \Theta(\mu n \lg^2 q)$ and $L_{\mathrm{AFV}} = \Theta(\mu n \lg^3 q)$). Moreover, if we restrict $\vec{v}$ in a small domain, say, $\{0,1\}^\mu$, then $L_{\mathrm{AFV}} = \Theta(\mu n \lg^2 q)$, and there is no improvement.

On the other hand, if we set $\#R = 2^{\Theta(n)}$ to implement applications, the improvement is drastic: $L_{\mathrm{ours}} = \Theta(\mu n \lg^2 q)$ since $\#\,\mathrm{GF}(q^n) = 2^{\Omega(n)}$ and $L_{\mathrm{AFV}} = \Theta(\mu n \lg^3 q) = \Theta(\mu n^4)$ since they need to set $q = 2^{\Theta(n)}$. In this case, efficiency is improved by a factor of $\tilde{O}(n^3)$.

Next, we compare the FIBE schemes with a small universe; that is, their identities are binary vectors of length N. Agrawal, Boyen, Vaikuntanathan, Voulgaris, and Wee [5] proposed a FIBE scheme based on the LWE assumption with *sub-exponential* parameters. They restricted $N = n^\epsilon$ with $\epsilon \in (0, 1/2)$ in order to obtain the security under the hardness of lattice problems. Their scheme is based on the worst-case hardness of lattice problems of approximation factor $2^{O(N)} = 2^{O(n^\epsilon)}$ with a subexponential-time algorithm. The length of their ciphertext is $\Theta(Nm \lg q) = \Theta(N^2 n \lg^2 n)$.

On the contrary, our scheme enjoys flexible $N = \mathrm{poly}(n)$ and a weaker assumption, which is the worst-case hardness of lattice problem of approximation factor $\tilde{O}(n^{4.5})$. The length of our ciphertext is $O(N^2 n \lg^2 n)$, which is the same as theirs.

We note that Agrawal et al. also extended their scheme to support identity space $(\mathbb{Z}_q^n)^N$ without changing parameters or the assumption (see [5, Appendix B]).

On the ring-LWE assumption: We finally note that there are the variants of the ABB IBE schemes based on the ring-LWE assumption and the variants of the AFV IPE scheme also [35,34], which yield certain exponentially large R. Our technique is orthogonal to their techniques and improves the variants of the AFV IPE scheme by a factor of $\lg q$. We will describe concrete schemes in the full version.

1.2 Related Works

IPE was introduced by Katz, Sahai, and Waters [30], who gave a fully attribute-hiding but selectively secure IPE scheme based on the composite-order pairings. Following them, several researchers proposed (H)IPE schemes based on the pairings [36,31,37,10,38,39].

On IPE based on lattices, Agrawal et al. [6] constructed the first IPE scheme which is selectively secure and weakly attribute hiding under the LWE assumption. Another study on a lattice-based HIPE scheme was done by Abdalla, De Caro, and Mochetti [3], who extended the AFV IPE scheme to a HIPE scheme. They also proposed two extensions of the HIPE scheme, a wild-carded IBE scheme and a CCA secure HIPE scheme. The CCA2 construction exemplifies the requirement of large R, since, in the construction, the attribute space of the basic scheme includes a one-time verification key as required for the CHK conversion [13].

Another line of study of IPE is initiated as spatial encryption (SE) defined by Boneh and Hamburg [14]. Hamburg [29] observed that HIPE and SE are strongly related, and Chen, Lim, Ling, and Wang [21] gave explicit property-preserving conversions between them, which enable us to treat SE schemes as (H)IPE schemes. For SE, see Hamburg's thesis [29].

From the perspective of lattice-based encryption beyond IBE, we refer to fuzzy IBE schemes by Agrawal et al. [5], a revocable IBE scheme by Chen, Lim, Ling, Wang, and Ngyuen [22], and an attribute-based encryption scheme by Boyen [16].

1.3 Overview of Our Construction

We give an overview of our construction. For simplicity, we focus on the construction of IPE and omit HIPE. After briefly explaining the basics and the AFV IPE, we present our ideas for "half untwisting" and "half twisting."

The basics: We first review the "dual" public-key encryption (PKE) scheme proposed by Gentry, Peikert, and Vaikuntanathan [26] (or the Peikert KEM [41]). Their public key is a random matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$. The ciphertext is a vector close to the lattice $\Lambda_q(\boldsymbol{A}) = \{\boldsymbol{z} \in \mathbb{Z}^m : \boldsymbol{z} \equiv \boldsymbol{A}^\top \boldsymbol{s} \text{ for some } \boldsymbol{s} \in \mathbb{Z}_q^n\}$. The secret key is a short basis of $\Lambda_q^\perp(\boldsymbol{A}) = \{\boldsymbol{z} \in \mathbb{Z}^m : \boldsymbol{A}\boldsymbol{z} \equiv \boldsymbol{0}\}$, which enables us to recover a lattice vector in $\Lambda_q(\boldsymbol{A})$ from a vector close to $\Lambda_q(\boldsymbol{A})$. Cash, Hofheinz, Kiltz, and Peikert [20] proposed the first IBE scheme based on the lattices in the standard model.

After that, Agrawal, Boneh, and Boyen [4] proposed a lattice analogue of the Boneh–Boyen IBE [12] (and that of the Waters IBE [48]). Let $id = \mathfrak{w} = w_0 + w_1 X + \cdots + w_{n-1}X^{n-1} \in \mathrm{GF}(q^n)$. Let H be an invertible (or full-rank) difference encoding [23] that maps a polynomial in $\mathrm{GF}(q^n)$ to an n by n matrix of elements in $\mathbb{Z}_q$. [2] In the ABB IBE scheme, the public parameters consist of $\boldsymbol{A}_0$, $\boldsymbol{A}_1$, and $\boldsymbol{B}$, and the encryption lattice for id is

[2] Originally, H is called as "full-rank difference" encoding [23]. Recently, this concept was generalized for composite q and others [35,8]: The one of reviews suggested to call it "invertible difference" and the author follows. We say that $H : R \to \mathbb{Z}_q^{n\times n}$ is invertible difference if for any distinct $\mathfrak{w} \neq \mathfrak{w}' \in R$, matrix $H(\mathfrak{w}) - H(\mathfrak{w}') \in \mathbb{Z}_q^{n\times n}$ is invertible (rather than the matrix has rank n). See Section 4 for a concrete construction.

$$\Lambda_{id} = \Lambda_q(\boldsymbol{A}_0 \mid \boldsymbol{A_1} + H(id) \cdot \boldsymbol{B}).$$

The master has a short basis for $\Lambda_q^{\perp}(\boldsymbol{A}_0)$, and it can generate secret keys for $\Lambda_{id}^{\perp}$ using the basis sampling techniques as in [20]. For the security proof, we require H to be invertible difference.

The "twist" by Agrawal, Freeman, and Vaikuntanathan: Agrawal, Freeman, and Vaikuntanathan [6] gave a novel twist on the ABB IBE [4] and obtained an IPE scheme.

In the AFV IPE scheme, the encryption lattice for ciphertext-attribute vector $\vec{w} \in \mathbb{Z}_q^{\mu}$ is defined as

$$\Lambda_{\vec{w}} = \Lambda_q\left(\boldsymbol{A}_0 \mid \boldsymbol{A}_1 + w_1\boldsymbol{B} \mid \cdots \mid \boldsymbol{A}_{\mu} + w_{\mu}\boldsymbol{B}\right).$$

The ciphertext is a vector $\boldsymbol{c} = (\boldsymbol{c}_0, \ldots, \boldsymbol{c}_{\mu}) \in (\mathbb{Z}_q^m)^{\mu+1}$ close to $\Lambda_{\vec{w}}$.

They define the mapping $F_{\vec{v}} : (\mathbb{Z}_q^m)^{\mu+1} \to (\mathbb{Z}_q^m)^2$ as

$$F_{\vec{v}}(\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_{\mu}) = \left(\boldsymbol{c}_0, \textstyle\sum_{i=1}^{\mu} v_i \boldsymbol{c}_i\right) \in \mathbb{Z}_q^{2m}$$

for decryption, where F means "fold." Notice that, if $\vec{v}$ is a *short* vector, e.g., $\vec{v} \in \{0,1\}^{\mu} \subset \mathbb{Z}_q^{\mu}$, then $F_{\vec{v}}(\boldsymbol{c})$ is a vector close to the lattice

$$\Lambda_{\vec{v},\vec{w}} = \Lambda_q\Big(\boldsymbol{A}_0 \,\Big|\, \textstyle\sum_{i=1}^{\mu} v_i(\boldsymbol{A}_i + w_i\boldsymbol{B})\Big) = \Lambda_q\Big(\boldsymbol{A}_0 \,\Big|\, \textstyle\sum_{i=1}^{\mu} v_i\boldsymbol{A}_i + (\vec{w}^{\top}\vec{v})\boldsymbol{B}\Big).$$

If $\vec{w}^{\top}\vec{v} = 0$ then the masking term, $(\vec{w}^{\top}\vec{v})\boldsymbol{B}$, vanishes. The secret key for $\vec{v} \in \mathbb{Z}_q^{\mu}$ is defined as a short basis of $\Lambda_q^{\perp}(\boldsymbol{A}_0 \mid \sum_{i=1}^{\mu} v_i\boldsymbol{A}_i)$.

They also gave a binary decomposition technique for $\vec{v}$ of long norm, which expands the public parameters and ciphertext by a factor of $\lg q$: they replaced $\vec{w}$ and $\vec{v}$ with $\vec{w}' = (1, 2, \ldots, 2^{k-1}) \otimes \vec{w}$, where $k = \lceil \lg q \rceil$ and $\otimes$ denotes the standard tensor product, and $\vec{v} \in \mathbb{Z}_q^{\mu}$ with $\vec{v}' \in \{0,1\}^{\mu k}$ such that $\vec{w}^{\top}\vec{v} = (\vec{w}')^{\top}\vec{v}'$. This technique is already exploited in the constructions of fully homomorphic encryption [19,18,17].

Our Ideas: Here, we present our two ideas for changing R from $\mathbb{Z}_q$ to $\mathrm{GF}(q^n)$.

Half untwist: We first change the domain of attributes $\vec{w}$ from $\mathbb{Z}_q^{\mu}$ to $\mathrm{GF}(q^n)^{\mu}$, while $\vec{v}$'s domain is the same as the original, $\mathbb{Z}_q^{\mu} \subset \mathrm{GF}(q^n)^{\mu}$.

Let us turn back to the invertible difference encoding H, which appeared in the ABB IBE but was omitted from the AFV IPE. We have the following facts on the typical construction of $H : \mathrm{GF}(q^n) \to \mathbb{Z}_q^{n \times n}$:

- Fact 1: H maps $w \in \mathbb{Z}_q \subseteq \mathrm{GF}(q^n)$ to $H(w) = w\boldsymbol{I}_n$, where $\boldsymbol{I}_n$ is the n-dimensional identity matrix.
- Fact 2: H is $\mathbb{Z}_q$-linear and is an isomorphism from $\mathrm{GF}(q^n)$ to a field contained in $\mathbb{Z}_q^{n \times n}$.

From Fact 1, we have $w\boldsymbol{B} = w\boldsymbol{I}_n\boldsymbol{B} = H(w) \cdot \boldsymbol{B}$. We can rewrite the encryption lattice of the AFV IPE for $\vec{w} \in \mathbb{Z}_q^{\mu} \subseteq \mathrm{GF}(q^n)^{\mu}$ as

$$\Lambda_{\vec{w}} = \Lambda_q(\boldsymbol{A}_0 \mid \boldsymbol{A}_1 + H(w_1 + 0X + \cdots + 0X^{n-1})\boldsymbol{B} \mid \cdots \mid \boldsymbol{A}_\mu + H(w_\mu + 0X + \cdots + 0X^{n-1})\boldsymbol{B}).$$

We discover the hidden H in the AFV IPE and find $(n-1)$ empty slots for each $i \in [\mu]$.

Now, we can change the domain of ciphertext-attribute vector $\vec{w}$ from $\mathbb{Z}_q^\mu$ to $\mathrm{GF}(q^n)^\mu$. We naturally define $\Lambda_{\vec{w}}$ for $\vec{w} = (\mathfrak{w}_1, \ldots, \mathfrak{w}_\mu)^\top \in \mathrm{GF}(q^n)^\mu$ as

$$\Lambda_{\vec{w}} = \Lambda_q(\boldsymbol{A}_0 \mid \boldsymbol{A}_1 + H(\mathfrak{w}_1)\boldsymbol{B} \mid \cdots \mid \boldsymbol{A}_\mu + H(\mathfrak{w}_\mu)\boldsymbol{B}).$$

For a short key vector $\vec{v} \in \mathbb{Z}_q^\mu \subset \mathrm{GF}(q^n)^\mu$ and a ciphertext $\boldsymbol{c} = (\boldsymbol{c}_0, \ldots, \boldsymbol{c}_\mu)$ close to $\Lambda_{\vec{w}}$, we observe that $F_{\vec{v}}(\boldsymbol{c}) = (\boldsymbol{c}_0, \sum_{i=1}^\mu v_i \boldsymbol{c}_v)$ is close to

$$\Lambda_{\vec{w},\vec{v}} = \Lambda_q\Big(\boldsymbol{A}_0 \,\Big|\, \textstyle\sum_{i=1}^\mu v_i(\boldsymbol{A}_i + H(\mathfrak{w}_i)\cdot\boldsymbol{B})\Big) = \Lambda_q\Big(\boldsymbol{A}_0 \,\Big|\, \textstyle\sum_{i=1}^\mu v_i\boldsymbol{A}_i + H(\vec{w}^\top\vec{v})\cdot\boldsymbol{B}\Big),$$

where the latter equality follows from the linearity of H (Fact 2). If $\vec{w}^\top\vec{v} = 0$ then $H(\vec{w}^\top\vec{v}) = \boldsymbol{O}$. By using a short basis of $\Lambda_q^\perp(\boldsymbol{A}_0 \mid \sum_{i=1}^\mu v_i\boldsymbol{A}_i)$, one can decrypt the ciphertext if the inner product is 0. Otherwise, the masking matrix $H(\vec{w}^\top\vec{v})\cdot\boldsymbol{B}$ survives and $H(\vec{w}^\top\vec{v})$ is invertible.

Half twist: We next change the domain of $\vec{v}$ from $\mathbb{Z}_q^\mu$ to $\mathrm{GF}(q^n)^\mu$.

We observe that the proof of security by Agrawal et al. [6] does not require randomness of $\boldsymbol{B}$. Hence, we can safely replace a random matrix $\boldsymbol{B}$ with a very structured matrix $\boldsymbol{G} = \boldsymbol{I}_n \otimes (1, 2, 2^2, \ldots, 2^{k-1})$ as in Micciancio and Peikert [35], where $\otimes$ denotes the Kronecker product, and $k = \lceil \lg q \rceil$,

We exploit the structure of $\boldsymbol{G}$ and define a new encoding, $H' : \mathrm{GF}(q^n) \to \{0,1\}^{m\times m}$ (see Section 4), which gives *pseudo-commutativity* with respect to $\boldsymbol{G}$ and H, that is, for any $\mathfrak{v} \in \mathrm{GF}(q^n)$, it holds that $\boldsymbol{G}\cdot H'(\mathfrak{v}) = H(\mathfrak{v})\cdot\boldsymbol{G}$.

We apply the above idea and new encoding to the half untwist version of the AFV IPE. The encryption lattice for $\vec{w}$ is $\Lambda_{\vec{w}} = \Lambda_q(\boldsymbol{A}_0 \mid \boldsymbol{A}_1 + H(\mathfrak{w}_1)\cdot\boldsymbol{G} \mid \cdots \mid \boldsymbol{A}_\mu + H(\mathfrak{w}_\mu)\cdot\boldsymbol{G})$ as in the previous version. We modify the key-extraction and decryption algorithms for $\vec{v} = (\mathfrak{v}_1, \ldots, \mathfrak{v}_\mu) \in \mathrm{GF}(q^n)^\mu$. In decryption, a ciphertext $(\boldsymbol{c}_0, \ldots, \boldsymbol{c}_\mu)$ is folded up by $H'(\mathfrak{v}_i)$ instead of v_i, that is,

$$F'_{\vec{v}}(\boldsymbol{c}_0, \ldots, \boldsymbol{c}_\mu) = \Big(\boldsymbol{c}_0, \textstyle\sum_{i=1}^\mu H'(\mathfrak{v}_i)^\top \cdot \boldsymbol{c}_i\Big).$$

By the pseudo-commutativity, the sum $F'_{\vec{v}}(\boldsymbol{c}_0, \ldots, \boldsymbol{c}_\mu)$ is a vector close to the lattice

$$\begin{aligned}
\Lambda_{\vec{w},\vec{v}} &= \Lambda_q\Big(\boldsymbol{A}_0 \,\Big|\, \textstyle\sum_{i=1}^\mu (\boldsymbol{A}_i + H(\mathfrak{w}_i)\cdot\boldsymbol{G})\cdot H'(\mathfrak{v}_i)\Big) \\
&= \Lambda_q\Big(\boldsymbol{A}_0 \,\Big|\, \textstyle\sum_{i=1}^\mu \boldsymbol{A}_i\cdot H'(\mathfrak{v}_i) + \sum_{i=1}^\mu H(\mathfrak{w}_i)\cdot H(\mathfrak{v}_i)\cdot\boldsymbol{G}\Big) \\
&= \Lambda_q\Big(\boldsymbol{A}_0 \,\Big|\, \textstyle\sum_{i=1}^\mu \boldsymbol{A}_i\cdot H'(\mathfrak{v}_i) + H(\vec{w}^\top\vec{v})\cdot\boldsymbol{G}\Big),
\end{aligned}$$

since the matrix norm of $H'(\mathfrak{v}_i) \in \{0,1\}^{m\times m}$ is at most m. The secret key is a short basis of lattice $\Lambda_q^\perp(\boldsymbol{A}_0 \mid \sum_{i=1}^\mu \boldsymbol{A}_i\cdot H'(\mathfrak{v}_i))$.

We note that the binary-decomposition technique is built into our new encoding H' and the structured matrix $\boldsymbol{G}$. Therefore, we can save the $\lg q$ factor introduced by the binary decomposition in the AFV IPE scheme.

2 Preliminaries

A security parameter is denoted by κ. We use the standard O-notations, O, Θ, Ω, and ω. We use capital bold symbols $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ for matrices. In particular, $\boldsymbol{I}_n$ denotes an n by n identity matrix. We use lower-case bold symbols $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ for vectors. In addition, we use over-arrows to denote ciphertext- and key-attribute vectors as $\vec{w}$, $\vec{v}$. We use lower-case fraktur symbols $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ for polynomials and elements of $\mathrm{GF}(q^n)$. The abbreviations DPT and PPT stand for deterministic polynomial time and probabilistic polynomial time. For any integer $q \geq 3$, we write $\mathbb{Z}_q$ for the ring $\{-(q-1)/2, \ldots, -1, 0, 1, \ldots, (q-1)/2\}$ with addition and multiplication modulo q.

A function $f(\kappa)$ is said to be negligible if $f(\kappa) = \kappa^{-\omega(1)}$. We denote a set of negligible functions by $\mathsf{negl}(\kappa)$. For a positive integer n, $[n]$ denotes $\{1, 2, \ldots, n\}$. For $x \in \mathbb{R}$, we define $\lfloor x \rceil = \lceil x - 1/2 \rceil$ as the integer closest to x. For $\boldsymbol{x} = (x_1, \ldots, x_\ell) \in \mathbb{R}^\ell$, we define $\lfloor \boldsymbol{x} \rceil$ as $(\lfloor x_1 \rceil, \ldots, \lfloor x_\ell \rceil) \in \mathbb{Z}^\ell$. For two matrices $\boldsymbol{X} \in \mathbb{R}^{m \times n_1}$ and $\boldsymbol{Y} \in \mathbb{R}^{m \times n_2}$, $[\boldsymbol{X} \mid \boldsymbol{Y}] \in \mathbb{R}^{m \times (n_1 + n_2)}$ is the concatenation of the columns of $\boldsymbol{X}$ and $\boldsymbol{Y}$. For two matrices $\boldsymbol{X} \in \mathbb{R}^{m_1 \times n}$ and $\boldsymbol{Y} \in \mathbb{R}^{m_2 \times n}$, $[\boldsymbol{X}; \boldsymbol{Y}] \in \mathbb{R}^{(m_1 + m_2) \times n}$ is the concatenation of the rows of $\boldsymbol{X}$ and $\boldsymbol{Y}$. For a vector $\boldsymbol{x} \in \mathbb{R}^m$, $\|\boldsymbol{x}\|_p$ denotes the ℓ_p norm of $\boldsymbol{x}$. For ease of notation, we omit the subscript if $p = 2$.

For matrix $\boldsymbol{X} = [\boldsymbol{x}_1 \ \ldots \ \boldsymbol{x}_n]$, $\tilde{\boldsymbol{X}}$ denotes the Gram-Schmidt orthogonalization of $\boldsymbol{X}$. For a matrix $\boldsymbol{X} = [\boldsymbol{x}_1; \ldots; \boldsymbol{x}_m] \in \mathbb{R}^{m \times n}$, $\|\boldsymbol{X}\|_{\mathrm{row}} = \max_i \|\boldsymbol{x}_i\|$. For a matrix $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, $s_1(\boldsymbol{X})$ denotes the largest singular value of $\boldsymbol{X}$; we have that $s_1(\boldsymbol{X}) = \sup_{\boldsymbol{u} \in \mathbb{R}^n, \|\boldsymbol{u}\| = 1} \|\boldsymbol{X}\boldsymbol{u}\| = \sup_{\boldsymbol{u}' \in \mathbb{R}^m, \|\boldsymbol{u}'\| = 1} \|\boldsymbol{X}^\top \boldsymbol{u}'\|$. For two matrices $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{Y} \in \mathbb{R}^{m \times k}$, we have $s_1(\boldsymbol{X}\boldsymbol{Y}) \leq s_1(\boldsymbol{X}) \cdot s_1(\boldsymbol{Y})$. We also have for any $\boldsymbol{X} \in \mathbb{R}^{n \times m}$, $\|\boldsymbol{X}\|_{\mathrm{row}}, \|\boldsymbol{X}^\top\|_{\mathrm{row}} \leq s_1(\boldsymbol{X})$. Finally, for ring R and positive integer n, $\mathrm{GL}_n(R)$ denotes the set of n by n invertible matrices whose entries in R.

Distribution: We recall distributions in the lattice-based cryptography. For a distribution χ, we often write $x \leftarrow \chi$, which indicates that we take a sample x from χ. For a finite set S, $U(S)$ denotes the uniform distribution over S. The Gaussian distribution with mean 0 and variance s^2, denoted by $N(0, s^2)$, is defined by density function $(1/s\sqrt{2\pi}) \cdot \exp(-x^2/2s^2)$ over $\mathbb{R}$. For $\alpha \in (0, 1)$ and positive integer q, we define the *discretized* Gaussian $\bar{\Psi}_\alpha$ as: take sample x from $N(0, \alpha^2/2\pi)$ and output $\lfloor qx \rceil \bmod q$. For positive real s, the n-dimensional Gaussian function is defined as $\rho_s(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x}\|^2/s^2)$. For positive real s and countable set A, the *discrete* Gaussian distribution $D_{A,s}$ is defined by $D_{A,s}(\boldsymbol{x}) = \frac{\rho_s(\boldsymbol{x})}{\sum_{\boldsymbol{y} \in A} \rho_s(\boldsymbol{y})}$.

2.1 Lattices

A (full-rank) lattice in $\mathbb{R}^n$ is $\Lambda = \{\sum_{i=1}^n x_i \boldsymbol{b}_i : x_i \in \mathbb{Z}\}$, where $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^n$ are linearly independent over $\mathbb{R}^n$. Matrix $\boldsymbol{B} = [\boldsymbol{b}_1 \ \ldots \ \boldsymbol{b}_n]$ is a basis of lattice Λ. For $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ and $\boldsymbol{u} \in \mathbb{Z}_q^n$, we define lattices and their shift:

$$\Lambda_q(\boldsymbol{A}) = \{\boldsymbol{y} \in \mathbb{Z}^m : \exists \boldsymbol{s} \in \mathbb{Z}_q^n \text{ such that } \boldsymbol{y} \equiv \boldsymbol{A}^\top \boldsymbol{s} \pmod q\},$$
$$\Lambda_q^\perp(\boldsymbol{A}) = \{\boldsymbol{e} \in \mathbb{Z}^m : \boldsymbol{A}\boldsymbol{e} \equiv \boldsymbol{0} \pmod q\},$$
$$\Lambda_q^{\boldsymbol{u}}(\boldsymbol{A}) = \{\boldsymbol{e} \in \mathbb{Z}^m : \boldsymbol{A}\boldsymbol{e} \equiv \boldsymbol{u} \pmod q\}.$$

We recall the property of very structured matrix $\boldsymbol{G}$.

Theorem 2.1 (Adapted version of [35, Theorem 4.1]). *Let $q \geq 2$, $n \geq 1$, $k = \lceil \lg q \rceil$, and $\bar{m} = nk$ be integers. Let $\boldsymbol{g} = (1, 2, \ldots, 2^{k-1}) \in \mathbb{Z}^k$ and $\boldsymbol{G} = \boldsymbol{I}_n \otimes \boldsymbol{g}$. Then the lattice $\Lambda_q^{\perp}(\boldsymbol{G})$ has a known basis $\boldsymbol{S} \in \mathbb{Z}^{\bar{m} \times \bar{m}}$ with $\|\tilde{\boldsymbol{S}}\| \leq \sqrt{5}$ and $\|\boldsymbol{S}\| \leq \max\{\sqrt{5}, \sqrt{k}\}$.*

Recently, Micciancio and Peikert introduced a new notion of "trapdoors" for lattices. Let $m = \bar{m} + nk$, where $k = \lceil \lg q \rceil$. We review their notion of trapdoors.

Definition 2.1 (Adapted, [35, Definition 5.2]). *Let $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ and $\boldsymbol{G} \in \mathbb{Z}_q^{n \times w}$ be matrices with $m \geq w \geq n$. We say a matrix $\boldsymbol{R} \in \mathbb{Z}^{(m-w) \times w}$ is a $\boldsymbol{G}$-trapdoor with tag $\boldsymbol{H} \in \mathrm{GL}_n(\mathbb{Z}_q) \subseteq \mathbb{Z}_q^{n \times n}$ if $\boldsymbol{A}\,[\boldsymbol{R}; \boldsymbol{I}_w] = \boldsymbol{HG}$. The* quality *of the trapdoor is measured by $s_1(\boldsymbol{R})$.*

Theorem 2.2. *We borrow the following algorithms in [35], which are improvements of those in the literature [7,26,9,4,42]. We set $k = \lceil \lg q \rceil$ and $m = \bar{m} + nk$ for simplicity of notation.*

$\mathsf{GenTrap}^D(\bar{\boldsymbol{A}}, \boldsymbol{H})$**:** *Given a matrix $\bar{\boldsymbol{A}} \in \mathbb{Z}_q^{n \times \bar{m}}$, an invertible matrix $\boldsymbol{H} \in \mathrm{GL}_n(\mathbb{Z}_q)$, and a distribution D over $\mathbb{Z}_q$, it outputs $\boldsymbol{A} = [\bar{\boldsymbol{A}} \mid \boldsymbol{HG} - \bar{\boldsymbol{A}}\boldsymbol{R}] \in \mathbb{Z}_q^{n \times (\bar{m}+nk)}$ and its trapdoor $\boldsymbol{R} \in \mathbb{Z}_q^{\bar{m} \times nk}$ with tag $\boldsymbol{H}$, where $\boldsymbol{R}$ is chosen from distribution D.*
In particular, we often set q as an odd prime, $\bar{m} = n \lg q + \omega(\lg \kappa)$, $D = U(\{-1, +1\})$, and choose $\bar{\boldsymbol{A}}$ from $\mathbb{Z}_q^{n \times \bar{m}}$ uniformly at random. These settings yield the obtained matrix $\boldsymbol{A}$ as $\mathsf{negl}(\kappa)$-uniform and $s_1(\boldsymbol{R}) \leq C(\sqrt{\bar{m}} + \sqrt{nk})$ with overwhelming probability.

$\mathsf{SampleD}(\boldsymbol{R}, \boldsymbol{A}, \boldsymbol{H}, \boldsymbol{u}, s)$**:** *The input is $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, its trapdoor $\boldsymbol{R} \in \mathbb{Z}_q^{\bar{m} \times nk}$ with tag $\boldsymbol{H} \in \mathrm{GL}_n(\mathbb{Z}_q)$, and a target vector $\boldsymbol{u}$, and Gaussian parameter $s > \sqrt{s_1(\boldsymbol{R})^2 + 1} \cdot \sqrt{7} \cdot \omega(\sqrt{\lg n})$. It outputs $\boldsymbol{x}$ according to a distribution statistically close to $D_{\Lambda_q^{\boldsymbol{u}}(\boldsymbol{A}), s}$; roughly speaking, it samples $\boldsymbol{x}$ from $D_{\mathbb{Z}, s}^m$ conditioned on $\boldsymbol{Ax} = \boldsymbol{u}$.*

2.2 Assumption

The learning with errors (LWE) problem proposed by Regev [46] is a generalization of the learning parity noise (LPN) problem.

For vector $\boldsymbol{s} \in \mathbb{Z}_q^n$ and distribution χ over $\mathbb{Z}_q$, let $A(\boldsymbol{s}, \chi)$ be a distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ defined by taking samples $\boldsymbol{a} \leftarrow \mathbb{Z}_q^n$ and $x \leftarrow \chi$, and outputting $(\boldsymbol{a}, \boldsymbol{a}^{\top}\boldsymbol{s} + x)$.

Definition 2.2 (The LWE problem and assumption). *For integer $q = q(n)$ and distribution χ over $\mathbb{Z}_q$, the learning with errors problem,* $\mathrm{LWE}(q, \chi)$*, distinguishes oracle $A(\boldsymbol{s}, \chi)$ from oracle $U(\mathbb{Z}_q^n \times \mathbb{Z}_q)$ for uniformly random $\boldsymbol{s} \in \mathbb{Z}_q^n$.*

We say the LWE assumption holds if for any PPT adversary $\mathcal{A}$, its advantage

$$\mathsf{Adv}_{\mathcal{A}, \mathrm{LWE}(q,\chi)}(n) = \left|\Pr[\mathcal{A}^{A(\boldsymbol{s},\chi)}(1^n) = 1] - \Pr[\mathcal{A}^{U(\mathbb{Z}_q^n \times \mathbb{Z}_q)}(1^n) = 1]\right| = \mathsf{negl}(n),$$

where $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$.

It is well-known that under (quantum) reductions, solving the LWE problem *on average* is as hard as the *worst* case of the approximation version of the shortest independent vector problem, SIVP_γ, and the decision version of the shortest vector problem, GapSVP_γ, where γ is an approximation factor for appropriate parameters. In particular, we have a reduction with parameter $\chi = \bar{\Psi}_\alpha$, $\alpha q \geq 2\sqrt{n}$, and $\gamma = \tilde{O}(n/\alpha)$. See [46,41] for details.

3 Predicate Encryption

We review the syntax of predicate encryption.

Definition 3.1. *Let $\mathcal{P} : \Phi \times \Sigma \to \{0,1\}$ be a predicate where Φ and Σ denote "key attribute" and "ciphertext attribute" spaces. A predicate encryption scheme for $\mathcal{P}$ is a fourtuplet of algorithms.*

$\mathsf{Setup}(1^\kappa) \to (pp, msk)$**:** *The setup algorithm takes as input security parameter 1^κ and outputs public parameters pp and master secret key msk.*

$\mathsf{Extract}(msk, \phi) \to dk_\phi$**:** *The extraction algorithm takes as input msk and key attribute $\phi \in \Phi$. It outputs decryption key dk_ϕ.*

$\mathsf{Enc}(pp, \sigma, M) \to ct$**:** *The encryption algorithm takes as input pp, ciphertext attribute $\sigma \in \Sigma$, and message $M \in \mathcal{M}$. It outputs ciphertext ct.*

$\mathsf{Dec}(pp, dk_\phi, ct) \to M$ **or** $\perp$**:** *The decryption algorithm takes as input decryption key dk_ϕ and ciphertext ct. It outputs either $M \in \mathcal{M}$ or rejection symbol $\perp$.*

We define slightly weak correctness for decryption. For any $\phi \in \Phi$, $\sigma \in \Sigma$, and $M \in \mathcal{M}$, if $\mathcal{P}(\phi, \sigma) = 1$ then

$$\Pr\left[M = \tilde{M} : \begin{array}{l}(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa); dk_\phi \leftarrow \mathsf{Extract}(msk, \phi);\\ ct \leftarrow \mathsf{Enc}(pp, \sigma, M); \tilde{M} \leftarrow \mathsf{Dec}(pp, dk_\phi, ct);\end{array}\right]$$

is overwhelming probability and if $\mathcal{P}(\phi, \sigma) = 0$ then

$$\Pr\left[\tilde{M} = \perp : \begin{array}{l}(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa); dk_\phi \leftarrow \mathsf{Extract}(msk, \phi);\\ ct \leftarrow \mathsf{Enc}(pp, \sigma, M); \tilde{M} \leftarrow \mathsf{Dec}(pp, dk_\phi, ct);\end{array}\right]$$

is overwhelming probability. As in [6], our construction satisfies the different correctness condition: the latter condition is replaced with the condition that if $\mathcal{P}(\phi, \sigma) = 0$ then $\mathsf{Dec}(pp, dk_\phi, ct)$ is computationally indistinguishable from a uniformly random element in $\mathcal{M}$. One can use a suitable message padding to obtain the original correctness, if an IPE scheme has message space $\{0,1\}^\ell$ for sufficiently large ℓ.

We next review the security definition of predicate encryption. Roughly speaking, we say that a PE scheme is weakly attribute hiding in a selective attribute setting against chosen-plaintext attacks (wAH-sA-CPA), if any adversary cannot distinguish $\mathsf{Enc}(pp, \sigma_0, M_0)$ or $\mathsf{Enc}(pp, \sigma_1, M_1)$, where σ_0 and σ_1 are declared at the initialization, even if the adversary can query the decryption key dk_ϕ for $\mathcal{P}(\sigma_0, \phi) = \mathcal{P}(\sigma_1, \phi) = 0$. The precise definition follows:

Definition 3.2 (wAH-sA-CPA security). *Let* PE *be a predicate encryption scheme, $\mathcal{A}$ an adversary, and κ a security parameter. The experiment between a challenger and adversary $\mathcal{A}$, $\mathsf{Expt}^{\text{wah-sa-cpa}}_{\mathcal{A},\mathsf{PE}}(1^\kappa)$, is defined as follows:*

Initialization: *Given security parameter* 1^κ*, run adversary* $\mathcal{A}$ *with* 1^κ*. Receive two ciphertext attributes* $\sigma_0, \sigma_1 \in \Sigma$ *from* $\mathcal{A}$*. Run* $(pp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$*. Flip a coin* $b \leftarrow \{0,1\}$*.*

Learning Phase: *Feed pp to adversary* $\mathcal{A}$*. Adversary* $\mathcal{A}$ *could issue queries to the following oracles in any order and many times except for the constraint regarding oracle* Challenge*.*

- *Oracle* Extract *receives key attribute* $\phi \in \Phi$ *subject to the restriction that* $\mathcal{P}(\phi, \sigma_0) = \mathcal{P}(\phi, \sigma_1) = 0$*. If so, it obtains* $dk_\phi \leftarrow \mathsf{Extract}(msk, \phi)$ *and provides* $\mathcal{A}$ *with* dk_ϕ*.*
- *Oracle* Challenge *receives two messages* M_0 *and* M_1*. It obtains* $C \leftarrow \mathsf{Enc}(pp, \sigma_b, M_b)$ *and provides* $\mathcal{A}$ *with* C*.*

Eventually, $\mathcal{A}$ *halts after it outputs its decision,* $b' \in \{0,1\}$*.*

Finalization: *Output* 1 *if* $b' = b$*. Otherwise, output* 0*.*

We define the advantage of $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\text{wah-sa-cpa}}_{\mathcal{A},\mathsf{PE}}(\kappa) = \left|\Pr[\mathsf{Expt}^{\text{wah-sa-cpa}}_{\mathcal{A},\mathsf{PE}}(1^\kappa) = 1 \mid b = 0] - \Pr[\mathsf{Expt}^{\text{wah-sa-cpa}}_{\mathcal{A},\mathsf{PE}}(1^\kappa) = 1 \mid b = 1]\right|.$$

We say that PE *is weakly attribute hiding against chosen-plaintext attacks in selective attribute setting (*wAH-sA-CPA*-secure) if* $\mathsf{Adv}^{\text{wah-sa-cpa}}_{\mathcal{A},\mathsf{PE}}(\kappa)$ *is negligible for every PPT adversary* $\mathcal{A}$*.*

4 Pseudo-commutativity of Invertible Difference Encoding

In this section, we define H and $H_{\boldsymbol{g}}$ such that, for any $\mathfrak{a}$, $H(\mathfrak{a}) \cdot \boldsymbol{G} = \boldsymbol{G} \cdot H_{\boldsymbol{g}}(\mathfrak{a})$ holds and $s_1(H_{\boldsymbol{g}}(\mathfrak{a}))$ is small. We first recall the polynomial rings. After a reminder of the invertible difference encoding, we define its companion $H_{\boldsymbol{g}}$.

4.1 Quick Reminder of Rings

Consider a finite ring $R = \mathbb{Z}_q[X]/\langle \mathfrak{g} \rangle$, where $\mathfrak{g} \in \mathbb{Z}_q[X]$ is monic and of degree n. If q is prime and $\mathfrak{g}$ is irreducible over $\mathbb{Z}_q$, ring R is the field $\mathrm{GF}(q^n)$.

We define the mapping $\tau : R \to \mathbb{Z}_q^n$ by $\mathfrak{a} = a_0 + a_1 X + \ldots a_{n-1} X^{n-1} \mapsto (a_0, \ldots, a_{n-1})^\top$. By this mapping (as known as "coefficient embedding"), we can identify a polynomial in R with a vector in $\mathbb{Z}_q^n$. We next define $\mathrm{Rot} : R \to \mathbb{Z}_q^{n \times n}$ by

$$\mathfrak{a} = a_0 + a_1 X + \ldots a_{n-1} X^{n-1} \mapsto [\tau(\mathfrak{a})\, \tau(\mathfrak{a}X) \ \ldots \ \tau(\mathfrak{a}X^{n-1})],$$

which is borrowed from Micciancio [33]. We note that

$$\mathrm{Rot}(\mathfrak{a}) \cdot \tau(\mathfrak{b}) = \tau(\mathfrak{a}\mathfrak{b}), \mathrm{Rot}(\mathfrak{a}) \cdot \mathrm{Rot}(\mathfrak{b}) = \mathrm{Rot}(\mathfrak{a}\mathfrak{b}), \text{ and } \mathrm{Rot}(\mathfrak{a}) + \mathrm{Rot}(\mathfrak{b}) = \mathrm{Rot}(\mathfrak{a} + \mathfrak{b}),$$

and, thus, Rot is a ring-homomorphism from R into $\mathbb{Z}_q^{n \times n}$.

4.2 Invertible Difference Encoding H

Lattice-based cryptography often employs an encoding $H : \mathrm{GF}(q^n) \to \mathbb{Z}_q^{n\times n}$ for prime q, e.g., [43, due to Micciancio] and [4]. Hereafter we stick to prime q.

We say that H is an invertible difference if for any two distinct polynomials $\mathfrak{a} \neq \mathfrak{a}' \in \mathrm{GF}(q^n)$, the difference of outputs, $H(\mathfrak{a}) - H(\mathfrak{a}')$, is always invertible.

In this paper, we employ explicit H defined by $H(\mathfrak{a}) := \mathrm{Rot}(\mathfrak{a})$. It holds that $H(\mathfrak{a}) - H(\mathfrak{a}') = H(\mathfrak{a}-\mathfrak{a}')$ for any $\mathfrak{a} \neq \mathfrak{a}'$. If $\mathfrak{a}-\mathfrak{a}'$ is a unit, that is, $\mathfrak{a} \neq \mathfrak{a}' \in \mathrm{GF}(q^n)$, then $H(\mathfrak{a}-\mathfrak{a}')$ is also a unit in $\mathbb{Z}_q^{n\times n}$. In addition, we note that for any constant $a \in \mathbb{Z}_q \subset \mathrm{GF}(q^n)$, $H(a) = a\boldsymbol{I}_n$.

4.3 New Encoding $H_{\boldsymbol{g}}$

We define a new encoding, denoted by $H_{\boldsymbol{g}}$, that maps an element in $\mathrm{GF}(q^n)$ to matrices in $\{0,1,\ldots,b-1\}^{nk\times nk}$ and gives pseudo-commutativity with $\boldsymbol{G}$ and H.

Let $b \geq 2$ be a positive integer and let B be the range $\{0,1,\ldots,b-1\} \subset \mathbb{Z}_q$. We define $k = \lceil \log_b q \rceil$ and $\boldsymbol{g} = (1, b, \ldots, b^{k-1})$. The gadget matrix $\boldsymbol{G}$ in [35] is defined by

$$\boldsymbol{G} = \boldsymbol{I}_n \otimes \boldsymbol{g} = \begin{bmatrix} \text{—}\boldsymbol{g}\text{—} & & & \\ & \text{—}\boldsymbol{g}\text{—} & & \\ & & \ddots & \\ & & & \text{—}\boldsymbol{g}\text{—} \end{bmatrix} = \begin{bmatrix} 1\ b \ldots\ b^{k-1} & & & \\ & 1\ b \ldots\ b^{k-1} & & \\ & & \ddots & \\ & & & 1\ b \ldots\ b^{k-1} \end{bmatrix} \in \mathbb{Z}_q^{n\times nk}.$$

For $a \in \mathbb{Z}_q$, we define b-ary decomposition of a by $d_{\boldsymbol{g}}(a) = (a_1, \ldots, a_k)^\top \in B^k$, on which we have that $\boldsymbol{g} \cdot d_{\boldsymbol{g}}(a) = \sum_{i=1}^{k} a_i \cdot b^{i-1} = a$.

We define $H_{\boldsymbol{g}}(\mathfrak{a})$ as the b-ary decomposition of $H(\mathfrak{a})$. More formally, we first define the mapping $D_{\boldsymbol{g}}$ by

$$D_{\boldsymbol{g}} : a \in \mathbb{Z}_q \mapsto [d_{\boldsymbol{g}}(a)\ d_{\boldsymbol{g}}(ba)\ \ldots\ d_{\boldsymbol{g}}(b^{k-1}a)] \in B^{k\times k}.$$

By the definition of $D_{\boldsymbol{g}}$, we have that $\boldsymbol{g} \cdot D_{\boldsymbol{g}}(a) = (a, ba, \ldots, b^{k-1}a) = a \cdot \boldsymbol{g}$, which is a source of the pseudo-commutativity. Next, we extend the domain of $D_{\boldsymbol{g}}$ to any matrix $\boldsymbol{A} = \{a_{i,j}\} \in \mathbb{Z}_q^{n\times m}$ as follows:

$$D_{\boldsymbol{g}}(\boldsymbol{A}) = \begin{bmatrix} D_{\boldsymbol{g}}(a_{1,1}) & D_{\boldsymbol{g}}(a_{1,2}) & \ldots & D_{\boldsymbol{g}}(a_{1,m}) \\ D_{\boldsymbol{g}}(a_{2,1}) & D_{\boldsymbol{g}}(a_{2,2}) & \ldots & D_{\boldsymbol{g}}(a_{2,m}) \\ \vdots & \vdots & \ddots & \vdots \\ D_{\boldsymbol{g}}(a_{n,1}) & D_{\boldsymbol{g}}(a_{n,2}) & \ldots & D_{\boldsymbol{g}}(a_{n,m}) \end{bmatrix} \in B^{nk\times mk}.$$

Finally, we define $H_{\boldsymbol{g}}$ that maps a polynomial into a matrix as follows:

$$\mathfrak{a} = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1} \in \mathrm{GF}(q^n) \mapsto D_{\boldsymbol{g}}(\mathrm{Rot}(\mathfrak{a})) \in B^{nk\times nk}.$$

The mapping $H_{\boldsymbol{g}}$ has two properties that are crucial for our construction. One is pseudo-commutativity with $\boldsymbol{G}$ and H and the other is a small matrix norm.

Lemma 4.1. *Let $\boldsymbol{G} = \boldsymbol{I}_n \otimes \boldsymbol{g} \in \mathbb{Z}_q^{n\times nk}$. It holds that, for any $\mathfrak{a} \in \mathrm{GF}(q^n)$, $\boldsymbol{G} \cdot H_{\boldsymbol{g}}(\mathfrak{a}) = H(\mathfrak{a}) \cdot \boldsymbol{G}$.*

Proof. We show that for any matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times n}$, $\boldsymbol{G} \cdot D_{\boldsymbol{g}}(\boldsymbol{A}) = \boldsymbol{A} \cdot \boldsymbol{G}$, where $\boldsymbol{A} = [\boldsymbol{a}_1 \mid \cdots \mid \boldsymbol{a}_n]$. We divide the matrices into k submatrices; $\boldsymbol{G} \cdot D_{\boldsymbol{g}}(\boldsymbol{A}) = [\boldsymbol{L}_1 \mid \cdots \mid \boldsymbol{L}_k]$ and $\boldsymbol{A} \cdot \boldsymbol{G} = [\boldsymbol{R}_1 \mid \cdots \mid \boldsymbol{R}_k]$, where $\boldsymbol{L}_i, \boldsymbol{R}_i \in \mathbb{Z}_q^{n\times n}$. It is easy to check that $\boldsymbol{L}_i = \boldsymbol{a}_i \otimes \boldsymbol{g} = \boldsymbol{R}_i$ for any i. □

Lemma 4.2. *For any* $\mathfrak{a} \in \mathrm{GF}(q^n)$, $\|H_{\boldsymbol{g}}(\mathfrak{a})\|_{\mathrm{row}} \le (b-1)\cdot\sqrt{nk}$ *and* $s_1(H_{\boldsymbol{g}}(\mathfrak{a})) \le (b-1)nk$.

Proof. Since $H_{\boldsymbol{g}} \in B^{nk\times nk}$, the maximal length of the rows is at most $(b-1)\sqrt{nk}$. The latter bound is obtained by the upper bound on the length of $(b-1)\cdot\mathbf{1}\cdot\boldsymbol{u}$, where $\mathbf{1}$ is an nk-dimensional all-1 matrix and $\boldsymbol{u}$ is a unit vector. □

4.4 On the Case Composite q

Although we have stuck to prime q here, lattice-based cryptography often employs $q = p^e$, say $q = 2^k$, or $q = \prod_i p_i$ for small prime p_i for the sake of easiness and speed of implementations. Therefore, one would extend our technique into such cases.

Micciancio and Peikert [35, Section 6.1 of the ePrint version] and Alperin-Sheriff and Peikert [8, Section 5.1] defined an encoding $H : \mathbb{Z}_q[X]/\langle \mathfrak{g}\rangle \to \mathbb{Z}_q^{n\times n}$, where $\mathfrak{g}$ is a monic degree-n polynomial in $\mathbb{Z}[X]$ and irreducible modulo every prime p dividing q. In their constructions, H is a ring homomorphism from $R = \mathbb{Z}_q[X]/\langle \mathfrak{g}\rangle$ into $\mathbb{Z}_q^{n\times n}$. Thus, if $\mathfrak{u} \in R$ is a unit, then $H(\mathfrak{u})$ is invertible. In general, $H(\mathfrak{u})$ is not invertible even for non-zero $\mathfrak{u} \in R \setminus R^*$.

This property suffices for public-key encryption, IBE, and signature, but, may trouble designers of predicates. If one can ensure that the inner product results in either a unit or zero of R, one can employ the above techniques.

4.5 On the Ring-LWE Setting

When q is a prime, we can extend our new encoding into the ring-LWE setting [32]. Let us consider the cyclotomic ring $\mathcal{R} = \mathbb{Z}[X]/\langle \Phi_m(X)\rangle$, where $\Phi_m(X)$ denotes the m-th cyclotomic polynomial. Let n be the degree of $\Phi_m(X)$. For any poly(n)-bounded prime q, we let $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$.

The Micciancio–Peikert algorithm in the ring-LWE setting: Let $\boldsymbol{g} = (1, b, \ldots, b^{k-1}) \in \mathcal{R}_q^k$. In the ring setting, we will use $\boldsymbol{g}$ directly instead of $\boldsymbol{G}$. Let us set $R = \mathbb{Z}_q[X]/\langle \mathfrak{g}\rangle \simeq \mathrm{GF}(q^n)$ as in the LWE case.

Micciancio and Peikert [34] define $\mathcal{R}$ to be R-module [3] by extending the ideas in [35, Section 6.1 of the ePrint version] and [8, Section 5.1]. Formally speaking, for $\mathfrak{a} \in R$ and $b \in \mathcal{R}$, scalar multiplication $\mathfrak{a} \odot b \in \mathcal{R}$ is defined by $\sigma^{-1}(\mathfrak{a}\cdot\sigma(b)) \in \mathcal{R}$, where $\sigma : \mathcal{R} \to R$ is an additive isomorphism. (Notice that R and $\mathcal{R}$ are additively isomorphic to $\mathbb{Z}_q^n$.) By the construction, $\mathcal{R}$ is an R-module and $\mathfrak{a}$ acts as the linear transformation over $\mathcal{R}$. Now, the trapdoor of $\boldsymbol{a} \in \mathcal{R}_q^{\bar{m}+k}$ with tag $\mathfrak{h} \in R$ is short $\boldsymbol{R} \in \mathcal{R}_q^{\bar{m}+k}$ satisfying $\boldsymbol{a} = [\bar{\boldsymbol{a}} \mid \mathfrak{h}\odot\boldsymbol{g} - \bar{\boldsymbol{a}}\boldsymbol{R}]$. We can define the new encoding $h_{\boldsymbol{g}} : R \to R^{k\times k}$ in a similar way to the LWE case. We defer the details to the full version.

[3] We say $\mathcal{R}$ is R-module if for any $\mathfrak{r}, \mathfrak{s} \in R$ and any $x, y \in \mathcal{R}$, we have $\mathfrak{r}(x+y) = \mathfrak{r}x + \mathfrak{r}y$, $(\mathfrak{r}+\mathfrak{s})x = \mathfrak{r}x + \mathfrak{s}x$, $(\mathfrak{r}\mathfrak{s})x = \mathfrak{r}(\mathfrak{s}x)$, and $1_R x = x$.

Langlois and Stehlé [47] also pointed out another way. Let us consider the case that n is even and $\Phi_m(X)$ is split into two polynomials $\mathfrak{f}_1$ and $\mathfrak{f}_2$ of degree $n/2$ which are irreducible over $\mathbb{Z}_q$. In such a case, we have $\mathcal{R}_q \simeq \mathbb{Z}_q[X]/\langle \mathfrak{f}_1 \rangle \times \mathbb{Z}_q[X]/\langle \mathfrak{f}_2 \rangle \simeq \mathrm{GF}(q^{n/2})^2$. We let $R = \mathrm{GF}(q^{n/2})$ and consider $H : R \to \mathcal{R}_q$ as follows:

We first define a duplicating function $\mathrm{dp} : \mathrm{GF}(q^{n/2}) \to \mathrm{GF}(q^{n/2})^2$ as $\mathfrak{a} \mapsto (\mathfrak{a}, \mathfrak{a})$. By the Chinese remainder theorem, we have invertible mapping $\tau : \mathbb{Z}_q[X]/\langle \Phi_m(X) \rangle \to \mathbb{Z}_q[X]/\langle \mathfrak{f}_1 \rangle \times \mathbb{Z}_q[X]/\langle \mathfrak{f}_2 \rangle$ as $\mathfrak{a} \mapsto (\mathfrak{a} \bmod \mathfrak{g}_1, \mathfrak{a} \bmod \mathfrak{g}_2)$. Then, we define the full-rank difference encoding from $\mathrm{GF}(q^{n/2})$ to R_q as $H = \tau^{-1} \circ \mathrm{dp}$. By the construction, H is an isomorphism from $\mathrm{GF}(q^{n/2})$ to a sub-ring of $\mathcal{R}_q$, which is a field.

Now, the trapdoor of $\boldsymbol{a} \in \mathcal{R}_q^{\bar{m}+k}$ with tag $\mathfrak{h} \in R$ is $\boldsymbol{R} \in \mathcal{R}_q^{\bar{m}+k}$ satisfying $\boldsymbol{a} = [\bar{\boldsymbol{a}} \mid H(\mathfrak{h})\boldsymbol{g} - \bar{\boldsymbol{a}}\boldsymbol{R}]$. We can define the new encoding $H_{\boldsymbol{g}} : R \to \mathcal{R}_q^{k \times k}$ in a similar way to the LWE case. We defer the details to the full version.

5 Our Construction

We describe our IPE scheme that supports inner-product predicates over $\mathrm{GF}(q^n)^\mu$. The scheme is obtained by applying our ideas in the introduction to the AFV IPE scheme. The extension to HIPE is deferred to the full version.

Let $\kappa \in \mathbb{N}$ be a security parameter. Let μ be the length of predicate and attribute vectors. Let n be a dimension of lattices and let q and m be the parameters that define the matrices. Let $\mathfrak{g} = \mathfrak{g}(x) \in \mathbb{Z}_q[x]$ be a monic, irreducible polynomial of degree n that explicitly defines $\mathrm{GF}(q^n)$.

For simplicity, we set $b = 2$, $B = \{0, 1\}$, and $k = k(\kappa, \mu) = \lceil \lg q \rceil$. Other choices are possible. For simplicity, we let $\zeta = \zeta(n)$ denote a fixed $\omega(\sqrt{\lg n})$ function. Let $s = s(\kappa, \mu)$ and $\alpha = \alpha(\kappa, \mu)$ be positive reals that define the Gaussians.

5.1 Construction

$\mathsf{Setup}(1^\kappa, n, q, m, \ell, s, \alpha, \mathfrak{g}, k)$**:** On input a security parameter 1^κ and additional parameters:

1. Generate a random matrix with a trapdoor by running $(\boldsymbol{A}, \boldsymbol{R}_{\boldsymbol{A}}) \leftarrow \mathsf{GenTrap}(1^\kappa, q, n, m)$.
2. Choose μ uniformly random matrices $\boldsymbol{B}_i \leftarrow \mathbb{Z}_q^{n \times nk}$ for $i \in [\mu]$.
3. Choose a random matrix $\boldsymbol{U} = [\boldsymbol{u}_1 \mid \cdots \mid \boldsymbol{u}_\ell] \leftarrow \mathbb{Z}_q^{n \times \ell}$.

Output $pp = ((n, q, m, \ell, s, \alpha, \mathfrak{g}, k), \boldsymbol{A}, \{\boldsymbol{B}_i\}, \boldsymbol{U})$ and $msk = (\boldsymbol{R}_{\boldsymbol{A}}, pp)$.

$\mathsf{Extract}(pp, msk, \vec{v})$**:** On input a key-attribute vector $\vec{v} = (\mathfrak{v}_1, \ldots, \mathfrak{v}_\mu)^\top \in \mathrm{GF}(q^n)^\mu$:

1. Define the matrices $\boldsymbol{B}_{\vec{v}} = \sum_{i=1}^{\mu} \boldsymbol{B}_i \cdot H_{\boldsymbol{g}}(\mathfrak{v}_i) \in \mathbb{Z}_q^{n \times nk}$ and $\boldsymbol{A}_{\vec{v}} = [\boldsymbol{A} \mid \boldsymbol{B}_{\vec{v}}] \in \mathbb{Z}_q^{n \times (m+nk)}$.
2. Sample vectors $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_\ell$ by using the master secret key $\boldsymbol{R}_{\boldsymbol{A}}$; Formally, for $i = 1, \ldots, \ell$, take sample $\boldsymbol{e}_i \leftarrow \mathsf{SampleD}(\boldsymbol{R}_{\boldsymbol{A}}, \boldsymbol{A}_{\vec{v}}, \boldsymbol{I}, \boldsymbol{u}_i, s)$.
3. Set $\boldsymbol{E}_{\vec{v}} = [\boldsymbol{e}_1 \mid \cdots \mid \boldsymbol{e}_\ell]$. (Notice that $\boldsymbol{A}_{\vec{v}} \cdot \boldsymbol{E}_{\vec{v}} = \boldsymbol{U}$.)

Output $dk_{\vec{v}} = \boldsymbol{E}_{\vec{v}}$.

$\mathsf{Enc}(pp, \vec{w}, \boldsymbol{m})$**:** On input pp, a ciphertext-attribute vector $\vec{w} = (\mathfrak{w}_1, \ldots, \mathfrak{w}_\mu)^\top \in \mathrm{GF}(q^n)^\mu$, and a message $\boldsymbol{m} \in \{0, 1\}^\ell$:

1. Choose a random vector $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$.

2. Set $\boldsymbol{c}_0 \leftarrow \boldsymbol{A}^\top \boldsymbol{s} + \boldsymbol{x}_0$, where $\boldsymbol{x}_0 \leftarrow \chi^m$.
3. Set $\boldsymbol{c}' \leftarrow \boldsymbol{U}^\top \boldsymbol{s} + \boldsymbol{x}' + \lfloor q/2 \rfloor \boldsymbol{m}$, where $\boldsymbol{x}' \leftarrow \chi^\ell$.
4. For $i = 1, \ldots, \mu$; sample $\boldsymbol{R}_i \leftarrow \{-1, +1\}^{m \times nk}$ and set $\boldsymbol{c}_i \leftarrow (\boldsymbol{B}_i + H(\mathfrak{w}_i) \cdot \boldsymbol{G})^\top \boldsymbol{s} + \boldsymbol{R}_i^\top \boldsymbol{x}_0 \in \mathbb{Z}_q^{nk}$.
5. Output $ct = (\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_\mu, \boldsymbol{c}')$.

$\mathsf{Dec}(pp, dk_{\vec{v}}, ct)$**:** On input pp, a decryption key $\boldsymbol{E}_{\vec{v}}$, and $ct = (\boldsymbol{c}_0, \ldots, \boldsymbol{c}_\mu, \boldsymbol{c}')$:

1. Compute $\boldsymbol{c}_{\vec{v}} \leftarrow \sum_{i=1}^{\mu} H'_{\boldsymbol{g}}(\mathfrak{v}_i)^\top \cdot \boldsymbol{c}_i$.
2. Let $\boldsymbol{c} \leftarrow [\boldsymbol{c}_0; \boldsymbol{c}_{\vec{v}}] \in \mathbb{Z}_q^{m+\bar{m}}$.
3. Compute $\boldsymbol{d} \leftarrow \boldsymbol{c}' - \boldsymbol{E}_{\vec{v}}^\top \boldsymbol{c} \bmod q$ and output $\lfloor (2/q)\boldsymbol{d} \rceil \bmod 2$.

Remark 5.1. In the following, we will take the noise of $\boldsymbol{c}_0$ and $\boldsymbol{c}'$ from $\chi = \bar{\Psi}_\alpha$. We note that $\bar{\Psi}_\alpha$ has a good tail bound on the inner product and this is why we employ the conservative distribution $\bar{\Psi}_\alpha$.

We can replace the distribution $\bar{\Psi}_\alpha$ with $D_{\mathbb{Z},\sigma}$. We then change the noises $\boldsymbol{x}_i$ with $\boldsymbol{x}_i \leftarrow D_{\mathbb{Z},r}^{nk}$ where $r = \sqrt{\|\boldsymbol{x}_0\|^2 + nk\sigma^2} \cdot \zeta$ as in the CCA2 secure PKE scheme in [35]. The problem $\mathrm{LWE}(n, q, D_{\mathbb{Z}, \sqrt{2} \cdot \alpha q})$ is as hard as $\mathrm{LWE}(n, q, \bar{\Psi}_\alpha)$, which is shown by Gordon, Katz, and Vaikuntanathan [28, Lemma 1] employing [42, Theorem 3.1]. Hence, even if we change the noise distribution from $\bar{\Psi}_\alpha$ to $D_{\mathbb{Z}, \sqrt{2}\alpha q}$, we can reduce the security to the lattice problems.

5.2 Correctness, Security, and Parameters

The scheme is correct and secure as the following theorems.

Theorem 5.1. *Let $\chi = \bar{\Psi}_\alpha$. Suppose that $s > 4Cm \cdot \omega(\sqrt{\lg n})$ and $(\alpha q \cdot \omega(\sqrt{\lg \kappa}) + \sqrt{m}/2) \cdot 4C\mu s m^2 < q/5$. Then our scheme is correct.*

Theorem 5.2. *Let $m = 2n \lg q + \omega(\lg \kappa)$ and $s \geq 3C\mu m^{1.5} \cdot \omega(\sqrt{\lg n})$. Suppose that the* $\mathrm{LWE}(n, q, \chi)$ *assumption holds. Then, the scheme is selectively and weakly attribute hiding.*

The proofs are obtained by merging those of [6] and [35]. We defer the proofs to the full version of the paper.

Parameter settings: Let us summarize the constraints on the parameters:

- To satisfy the correctness (Theorem 5.1), we require that $\chi = \bar{\Psi}_\alpha$, $s > 4Cm\omega(\sqrt{\lg n})$, and $(\alpha q \cdot \omega(\sqrt{\lg \kappa}) + \sqrt{m}/2) \cdot 4C\mu s m^2 < q/5$. For example, we can take $q = \Omega(\mu m^{5/2} s)$ and $\alpha \leq \left(\mu m^2 s \cdot \omega(\sqrt{\lg \kappa})\right)^{-1}$ to satisfy the above condition with $q\alpha \cdot \omega(\sqrt{\lg \kappa}) = \sqrt{m}/2$.
- From the security (Theorem 5.2), we obtain the bound that $m = 2n \lg q + \omega(\lg \kappa)$ and $s \geq 3C\mu m^{1.5} \cdot \omega(\sqrt{\lg n})$.
- In order to reduce the security to the worst-case hardness of lattice problems, we require that $q\alpha > 2\sqrt{n}$ and $1/\alpha = \mathrm{poly}(n)$.

For example, the following setting fulfills the above requirements:

$$k = \lceil \lg q \rceil, \qquad \zeta = \omega(\sqrt{\lg(2m)}), \qquad m = 3n \lg q,$$
$$s = 3C\mu m^{1.5} \cdot \zeta, \qquad q = 60C^2\mu^2 \cdot m^4 \cdot \zeta, \qquad \alpha = (120C^2\mu^2 \cdot m^{3.5} \cdot \zeta^2)^{-1}.$$

By these settings, the security is based on the worst-case hardness of GapSVP_γ or SIVP_γ, where $\gamma = \tilde{\Omega}(\mu^2 n^{4.5})$, while the AFV scheme is based on that with $\gamma = \tilde{\Omega}(\mu^2 n^4)$. (We note that if the AFV scheme also employs the Micciancio–Peikert trapdoor [35] as we did, γ is reduced to $\tilde{\Omega}(\mu^2 n^{3.5})$.)

In our scheme, the size of the public parameter is $nm \lg q + \mu n^2 k \lg q + \ell n \lg q = \Theta(\mu n^2 \lg^2 q) = \tilde{\Theta}(\mu n^2)$, and the size of the ciphertext is $m \lg q + \mu n k \lg q + \ell \lg q = \Theta(\mu n \lg^2 q)$, where ℓ denotes the length of plaintexts.

6 Fuzzy Identity-Based Encryption

In this section, we construct a FIBE scheme from a weakly attribute-hiding IPE scheme in general way. We first review the embedding of exact threshold by Katz, Sahai, and Waters. If the IPE scheme hides attribute weakly, we can take logical disjunction in a lazy way as Waters pointed out [6, Remark 5.1 of the ePrint version].

Exact threshold: For binary vector $\vec{x} \in \{0,1\}^N$, $H_w(\vec{x})$ denotes a Hamming weight of $\vec{x}$, that is, the number of 1 in $\vec{x}$. For binary vectors $\vec{a}, \vec{x} \in \{0,1\}^\mu$, the exact threshold predicate is denoted by $\mathcal{P}^{\mathsf{th}}_{=t}(\vec{a}, \vec{x})$ and outputs 1 if and only if $H_w(\vec{a} \,\&\, \vec{x}) = t$, where & denotes the logical conjunction. Suppose that $t < q$. Katz, Sahai, and Waters [30] gave an embedding $\mathcal{P}^{\mathsf{th}}_{=t}$ into $\mathcal{P}^{\mathsf{ipe}}$ as follows:

$$\mu = N + 1, \vec{v} = (\vec{a}, 1) \in \mathbb{Z}_q^\mu, \text{ and } \vec{w} = (\vec{x}, -t) \in \mathbb{Z}_q^\mu.$$

We have that $\vec{w}^\top \vec{v} = 0$ if and only if $H_w(\vec{a} \,\&\, \vec{x}) = t$.

6.1 Construction

Now, we implement FIBE on small universe $\{0,1\}$ from IPE. Let $\{0,1\}^N$ be a space of identities. The threshold predicate over $\{0,1\}^N$ is defined by $\mathcal{P}^{\mathsf{th}}_{\geq t}(\vec{a}, \vec{x}) = 1$ if and only if $H_w(\vec{a} \,\&\, \vec{x}) \geq t$.

We observe that the above predicate can be written as $\bigvee_{i=t}^{N} \mathcal{P}^{\mathsf{th}}_{=i}(\vec{a}, \vec{x})$. Hence, repeating ciphertexts of an IPE scheme that supports the relations $\mathcal{P}^{\mathsf{th}}_{=i}$ for $i = t, \ldots, N$, we can implement a FIBE scheme by the relation $\mathcal{P}^{\mathsf{th}}_{\geq t}$.

When we employ our IPE scheme, the obtained scheme has a ciphertext of length $(N-t+1) \cdot O(Nm \lg q) = O(N^2 n \lg^2 q)$ and enjoys the security reduced to the worst-case hardness of lattice problems with approximation factor $\tilde{O}(n^{4.5})$.

Comparison: Agrawal et al. already presented FIBE schemes from lattices [5]. Their small-universe construction is defined with the identity space $\{0,1\}^N$ as in our case. They gave concrete parameter settings for $\epsilon \in (0, 1/2)$ as follows:

$$N = n^\epsilon, q \in [n^6 2^{5N}, 2n^6 2^{5N}], m = n^{1.5} \geq 5n \lg q, \text{ and } \alpha = 2\sqrt{m}/q = 1/(2^{5n^\epsilon} \cdot \mathrm{poly}(n)).$$

The length of the ciphertext is $N \cdot O(m \lg q) = O(n^{1.5+2\epsilon} \lg n)$. The security is reduced to the worst-case hardness of $2^{O(n^\epsilon)}$-approximating GapSVP or SIVP using $2^{O(n^\epsilon)}$-time algorithms, which is stronger assumption than that we employ.

We finally note that, their scheme allows identity space $(\mathbb{Z}_q^n)^N$ without drastic changes of parameters whereas our scheme cannot.

Acknowledgments. The author thanks Damien Stehlé, Chris Peikert, Daniele Micciancio, and reviewers for helpful discussions and comments.

References

1. Abdalla, M., Birkett, J., Catalano, D., Dent, A.W., Malone-Lee, J., Neven, G., Schuldt, J.C.N., Smart, N.P.: Wildcarded identity-based encryption. Journal of Cryptology 24(1), 42–82 (2011), combined and extended of two papers [2, 11]
2. Abdalla, M., Catalano, D., Dent, A.W., Malone-Lee, J., Neven, G., Smart, N.P.: Identity-Based Encryption Gone Wild. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006, Part II. LNCS, vol. 4052, pp. 300–311. Springer, Heidelberg (2006), the full version is available at `http://eprint.iacr.org/2006/30`
3. Abdalla, M., De Caro, A., Mochetti, K.: Lattice-Based Hierarchical Inner Product Encryption. In: Hevia, A., Neven, G. (eds.) LATINCRYPT 2012. LNCS, vol. 7533, pp. 121–138. Springer, Heidelberg (2012)
4. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert (ed.) [27], pp. 553–572
5. Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or, fuzzy IBE) from lattices. In: Fischlin, et al. (eds.) [25], pp. 280–297
6. Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional Encryption for Inner Product Predicates from Learning with Errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011), the full version is available at `http://eprint.iacr.org/2011/410`
7. Ajtai, M.: Generating Hard Instances of the Short Basis Problem. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999)
8. Alperin-Sheriff, J., Peikert, C.: Circular and KDM security for identity-based encryption. In: Fischlin, et al. (eds.) [25], pp. 334–352
9. Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: Albers, S., Marion, J.Y. (eds.) STACS 2009. LIPIcs, vol. 3, pp. 75–86. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany (2009)
10. Attrapadung, N., Libert, B.: Functional Encryption for Inner Product: Achieving Constant-Size Ciphertexts with Adaptive Security or Support for Negation. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 384–402. Springer, Heidelberg (2010)
11. Birkett, J., Dent, A.W., Neven, G., Schuldt, J.C.N.: Efficient Chosen-Ciphertext Secure Identity-Based Encryption with Wildcards. In: Pieprzyk, J., Ghodosi, H., Dawson, E. (eds.) ACISP 2007. LNCS, vol. 4586, pp. 274–292. Springer, Heidelberg (2007)
12. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. Journal of Cryptology 24(4), 659–693 (2011), a preliminary version appeared In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)

13. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. SIAM Journal on Computing 36(5), 1301–1328 (2006)
14. Boneh, D., Hamburg, M.: Generalized Identity Based and Broadcast Encryption Schemes. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 455–470. Springer, Heidelberg (2008)
15. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
16. Boyen, X.: Attribute-based encryption from lattices (2012) (to appear TCC 2013)
17. 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), see also http://eprint.iacr.org/2012/078
18. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: Goldwasser, S. (ed.) ITCS 2012, pp. 309–325. ACM (2012), see also http://eprint.iacr.org/2011/277
19. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS 2011, pp. 97–106. IEEE Computer Society (2011), see also http://eprint.iacr.org/2011/344
20. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert (ed.) [27], pp. 523–552
21. Chen, J., Lim, H.W., Ling, S., Wang, H.: The relation and transformation between hierarchical inner product encryption and spatial encryption. Designs, Codes and Cryptography, Online First (2012), see also http://eprint.iacr.org/2011/455
22. Chen, J., Lim, H.W., Ling, S., Wang, H., Nguyen, K.: Revocable Identity-Based Encryption from Lattices. In: Susilo, W., Mu, Y., Seberry, J. (eds.) ACISP 2012. LNCS, vol. 7372, pp. 390–403. Springer, Heidelberg (2012), the full version is available at http://eprint.iacr.org/2011/583
23. Cramer, R., Damgård, I.: On the Amortized Complexity of Zero-Knowledge Protocols. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 177–191. Springer, Heidelberg (2009)
24. Dwork, C. (ed.): Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20. ACM (2008)
25. Fischlin, M., Buchmann, J., Manulis, M. (eds.): PKC 2012. LNCS, vol. 7293. Springer, Heidelberg (2012)
26. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork (ed.) [24], pp. 197–206, see also http://eprint.iacr.org/2007/432
27. Gilbert, H. (ed.): EUROCRYPT 2010. LNCS, vol. 6110. Springer, Heidelberg (2010)
28. 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)
29. Hamburg, M.: Spatial Encryption. Ph.D. thesis, Stanford University (2011), http://eprint.iacr.org/2011/389
30. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008), the full version is available at http://eprint.iacr.org/2007/404
31. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: Gilbert (ed.) [27], pp. 62–91, the full version is available at http://eprint.iacr.org/2010/110
32. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert (ed.) [27], pp. 1–23

33. Micciancio, D.: Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. Computational Complexity 16, 365–411 (2007), a preliminary version appeared in FOCS 2002 (2002), See also ECCC TR04-095
34. Micciancio, D., Peikert, C.: Private communication (December 12, 2012)
35. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, Johansson (eds.) [44], pp. 700–718, http://eprint.iacr.org/2011/501
36. Okamoto, T., Takashima, K.: Hierarchical Predicate Encryption for Inner-Products. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 214–231. Springer, Heidelberg (2009)
37. Okamoto, T., Takashima, K.: Fully secure functional encryption with general relations from the decisional linear assumption. In: Rabin (ed.) [45], pp. 191–208, the full version is available at http://eprint.iacr.org/2010/563
38. Okamoto, T., Takashima, K.: Achieving Short Ciphertexts or Short Secret-Keys for Adaptively Secure General Inner-Product Encryption. In: Lin, D., Tsudik, G., Wang, X. (eds.) CANS 2011. LNCS, vol. 7092, pp. 138–159. Springer, Heidelberg (2011), the full version is available at http://eprint.iacr.org/2010/648
39. Okamoto, T., Takashima, K.: Adaptively attribute-hiding (hierarchical) inner product encryption. In: Pointcheval, Johansson (eds.) [44], pp. 591–608, the full version is available at http://eprint.iacr.org/2010/543
40. Park, J.H.: Inner-product encryption under standard assumptions. Designs, Codes and Cryptography 58(3), 235–257 (2011)
41. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Mitzenmacher, M. (ed.) STOC 2009, pp. 333–342. ACM (2009)
42. Peikert, C.: An efficient and parallel Gaussian sampler for lattices. In: Rabin (ed.) [45], pp. 80–97
43. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: Dwork (ed.) [24], pp. 187–196
44. Pointcheval, D., Johansson, T. (eds.): EUROCRYPT 2012. LNCS, vol. 7237. Springer, Heidelberg (2012)
45. Rabin, T. (ed.): CRYPTO 2010. LNCS, vol. 6223. Springer, Heidelberg (2010)
46. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM 56(6), Article 34 (2009), a preliminary version appeared in STOC 2005 (2005)
47. Stehlé, D.: Private communication (December 12, 2012)
48. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005), the full version is available at http://eprint.iacr.org/2004/180

Techniques for Efficient Secure Computation Based on Yao's Protocol*

Yehuda Lindell

Dept. of Computer Science
Bar-Ilan University, Israel
lindell@biu.ac.il

Abstract. In the setting of secure two-party computation, two parties wish to securely compute a function of their joint private inputs. The theoretical foundations of this problem were laid down in the 1980s, and it has been heavily studied due to its generality and many applications. However, until recently, secure computation was considered a theoretical problem of purely theoretical interest. This has changed, and progress on the question of efficient secure computation has been extraordinarily fast in the past five years. In this talk, we survey some of this recent progress and describe the main techniques used for obtaining fast two-party computation, based on Yao's garbled circuit protocol. We will present the main algorithmic/protocol improvements, as well as implementation issues that have turned out to be a big factor in obtaining concrete efficiency. In addition, we will relate to the settings of semi-honest, covert and malicious adversaries, and will describe the challenges that arise for each along with the solutions and major open questions.

* This work was funded by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 239868.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, p. 253, 2013.

Non-Interactive Key Exchange

Eduarda S.V. Freire[1,⋆], Dennis Hofheinz,[2,⋆⋆], Eike Kiltz[3,⋆⋆⋆], and Kenneth G. Paterson[1,†]

[1] Royal Holloway, University of London
[2] Karlsruhe Institute of Technology
[3] Ruhr-Universität Bochum

Abstract. Non-interactive key exchange (NIKE) is a fundamental but much-overlooked cryptographic primitive. It appears as a major contribution in the ground-breaking paper of Diffie and Hellman, but NIKE has remained largely unstudied since then. In this paper, we provide different security models for this primitive and explore the relationships between them. We then give constructions for secure NIKE in the Random Oracle Model based on the hardness of factoring and in the standard model based on the hardness of a variant of the decisional Bilinear Diffie Hellman Problem for asymmetric pairings. We also study the relationship between NIKE and public key encryption (PKE), showing that a secure NIKE scheme can be generically converted into an IND-CCA secure PKE scheme. Our conversion also illustrates the fundamental nature of NIKE in public key cryptography.

Keywords: non-interactive key exchange, public-key cryptography, pairings.

1 Introduction

Non-interactive key exchange (NIKE) is a cryptographic primitive which enables two parties, who know each others' public keys, to agree on a symmetric shared key without requiring any interaction. The canonical example of a NIKE scheme can be found in the seminal paper by Diffie and Hellman [1]: let G be a group of prime order p with generator g, and assume Alice has public key $g^x \in G$ and private key $x \in \mathbb{Z}_p$, while Bob has public key $g^y \in G$ and private key $y \in \mathbb{Z}_p$. Then Alice and Bob can both compute the value $g^{xy} \in G$ without exchanging any messages. More properly, Alice and Bob should hash this key together with their identities in order to derive a symmetric key $H(\texttt{Alice}, \texttt{Bob}, g^{xy})$.

⋆ Eduarda S.V. Freire was supported by CAPES Foundation/Brazil on grant 0560/09-0 and Royal Holloway, University of London.
⋆⋆ Dennis Hofheinz was supported by a DFG grant (GZ HO 4534/2-1).
⋆⋆⋆ Eike Kiltz was funded by a Sofja Kovalevskaja Award of the Alexander von Humboldt Foundation and the German Federal Ministry for Education and Research.
† Kenneth G. Paterson was supported by EPSRC Leadership Fellowship EP/H005455/1.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 254–271, 2013.

This example encapsulates in a nutshell all the basic features required of a NIKE scheme: users should agree on some common parameters (p, G and g here), then create their key pairs. Once these are computed and the public keys distributed, any pair of users can set up a shared key without further exchange of messages. The security properties desired of NIKE are, informally at least, clear: compromise of one user's private key should not affect the security of shared keys between pairs of uncorrupted users; compromise of one shared key should not undermine the security of other shared keys. Naturally, since the primitive is non-interactive, one cannot hope to obtain any kind of forward security properties. In practice, the public keys will be certified, and consideration needs to be given to modelling the key registration process.

NIKE has real-world applications. In wireless and sensor networks, conserving battery power is a prime concern, and so the energy cost of communication must be minimised. Thus using key establishment methods that minimise the number of bits that need to be transmitted is of fundamental importance. In particular, when faced with a jamming adversary, reducing the total number of rounds of interaction needed to establish a key is particularly helpful. NIKE is an excellent option in solving this problem, since a key can be established with minimal communication and interaction: assuming the public keys are pre-distributed, all that is needed is an exchange of identifiers for those keys, and often this exchange must take place anyway, in order to establish communications. A recent paper [2] gives a detailed evaluation of the energy costs of interactive and non-interactive key exchange protocols in the ID-based and PKI settings for wireless communications with a jamming adversary, demonstrating that significant energy savings can be made by adopting a non-interactive approach to key establishment. Its non-interactive nature makes NIKE an abstract building block that is *qualitatively* different from interactive key exchange: e.g., to achieve deniable authentication, [3] explicitly requires a *non-interactive* key exchange. But NIKE can also be used as a basis for *interactive* key exhange [4]: one can use the shared key in a MAC to authenticate an exchange of ephemeral Diffie-Hellman values. Finally, NIKE can be used to build very simple non-interactive designated verifier signature schemes [5], again using the shared key in a MAC to authenticate messages. Thus NIKE appears in various guises throughout the literature.

Despite its appearing in the very first paper on public key cryptography, the NIKE primitive has so far received scant attention as a primitive in its own right. Cash, Kiltz and Shoup (CKS) [6] provided a basic security model for NIKE and analysed the Diffie-Hellman-based scheme above, as well as a twinned variant of it, in the Random Oracle Model (ROM). There is also some work in the ID-based setting [7,8,9,10], also all restricted to the ROM.

Our Contributions: Our contention is that NIKE is long overdue for more serious attention and development. In this paper, we initiate the systematic study of NIKE in the public key setting, providing: models and their relationships; constructions for secure NIKE in the Random Oracle Model and in the standard model in the challenging setting where the adversary can introduce arbitrary

public keys into the system; and a construction for IND-CCA secure public key encryption (PKE) from any secure NIKE. Let us expand on each of these contributions in turn.

Models: It would seem that definitions and security models for interactive key exchange (e.g., [11,12,13,14]) could provide a natural starting point for formalising NIKE. However, here we take the CKS definition [6] for NIKE as our starting point. One reason for using a case-tailored NIKE definition is simplicity: existing security models for interactive key exchange give considerable attention to properties which are irrelevant in the NIKE setting. (For instance, forward security, multiple sessions, and in particular the pairing of sessions play no role in a non-interactive setting.) Another reason for a case-tailored NIKE definition is that we can focus on adversarial key registration queries; these are usually only implicitly [14] (or not at all [11,13]) considered in the standard models for interactive key exchange[1]. However, in our setting, adversarial key registrations pose the main technical obstacle to achieve NIKE security, as we will explain below.

The CKS security model for NIKE uses an indistinguishability- and game-based approach to define security, with the adversary being required to distinguish real from random keys in responses to its test queries. The model does allow the adversary to register public keys of his choice in the system and then to make queries for the shared keys between these "corrupted" users and honest (non-adversarially controlled) users, so-called *corrupt reveal queries*. This translates in the real world to minimising the assumptions made about certification procedures followed by the Certification Authority (CA) in the PKI supporting the NIKE: it means that the CA is not assumed to check that a public key submitted for certification has not been submitted before, and does not check that the party submitting the public key knows the corresponding private key. The model for NIKE in [6] is similar to, and presumably inspired by, the early work of Shoup [12] on interactive key exchange, where capturing so-called PKI attacks, also known as rogue-key attacks, was intrinsic to the security modelling. This modelling approach is referred to elsewhere in the literature as the *plain* setting (see [16,17] and the references therein) or the *bare PKI* setting [3]. The CKS model is certainly more challenging than settings where proofs of knowledge or proofs of possession of private keys are assumed to be given during registration, or where the adversary must reveal its secret key directly (as with the knowledge of secret key assumption used in [18,19]). However, the CKS model has some shortcomings: the adversary is not allowed to directly query for the shared keys held between pairs of honest users, but instead only gets to see real or random values for these via test queries. Moreover the model does not allow an adversary to query for the private keys of honestly registered users.

Therefore, as a necessary precursor to the further development of NIKE, we start by exploring different models for NIKE and their relationships (Section 2).

[1] We mention that some security analyses (e.g., [15]) and Shoup's security model [12] do explicitly consider adversarial key registration queries.

In summary, we introduce three new security models for NIKE and show that they are all polynomially equivalent to one another and to the original CKS model from [6]. One of our models, the *m-CKS-heavy* model, augments the CKS model and effectively allows all conceivable queries, without allowing the adversary to win trivially. It is our preferred security model for NIKE. Another of our models, *CKS-light*, allows only two honest users, no corruption of honest users, and a single test query. Thus it is particularly simple and so easy to use when analyzing specific NIKE schemes; moreover our results showing equivalence between the models ensure that security in this model implies security in the preferred *m-CKS-heavy* model.

We stress that all these models allow the adversary to register public keys of his choice in the system, so are in the plain setting.

Constructions for NIKE: In Section 4, we give two concrete constructions for NIKE schemes meeting our *CKS-light* security definition, and hence secure in our preferred *m-CKS-heavy* model (*with* dishonest key registrations).

Our two constructions are inspired by public key encryption (PKE) schemes which are secure against chosen-ciphertext attacks (IND-CCA secure). We note that dealing with *corrupt reveal* queries requires techniques to guard against active attacks, which in part explains the connection to IND-CCA security. Indeed, we will also show how to go in the reverse direction, converting any secure NIKE scheme into an IND-CCA secure PKE scheme, see below. We stress, however, that we cannot simply take any IND-CCA secure PKE scheme and directly interpret it as a NIKE scheme.[2] Rather, our constructions for NIKE exploit specific properties of the underlying PKE schemes. In fact, our belief is that a generic construction for secure NIKE from PKE is unlikely to be forthcoming.

The first scheme acts as a warm-up. It is provably secure under the factoring assumption in the Random Oracle Model (ROM) and uses ideas from [20] to analyse the basic Diffie-Hellman scheme, where keys are of the form $H(\mathtt{Alice}, \mathtt{Bob}, g^{xy})$, in the group of signed quadratic residues. We note that closely related schemes were analysed in [6], but in different groups and under different assumptions. Specifically, a twinned version of the scheme was proved secure under the CDH assumption, while it is stated that the basic Diffie-Hellman scheme is secure under the Strong DH assumption.

We remark that the latter claim of [6] is problematic. Concretely, the Strong DH assumption is not (directly) sufficient to show that the basic Diffie-Hellman scheme is secure. Namely, the corresponding security reduction requires *two* DDH oracles – one for each of the two users sharing the key on which the adversary wants to be challenged – while the Strong DH assumption supplies only one. Certainly this problem could be solved instead by appealing to a suitable gap-DH assumption. We show how to overcome this problem in the group

[2] One reason is that it is not clear what should correspond to the NIKE public key: a PKE public key, a PKE ciphertext, or a combination of both? Besides, the corresponding security experiments for NIKE and PKE schemes are rather different: there usually is one challenge ciphertext in a PKE security experiment, while there are at least two challenge users in a NIKE security experiment.

of signed quadratic residues without the need to rely on a gap assumption. We then proceed to sketch how to transport this scheme to the standard model, under the additional assumption that the adversary only registers valid public keys. Because of the extra assumption, this scheme does not strictly speaking meet our security definitions, and would require validity to be enforced by some means in an interactive registration protocol (for example, via a proof of correctness of the public key). This limitation of our standard model, factoring-based solution reflects the technical challenge involved in achieving our "bare PKI" security notions.

Our second NIKE scheme is provably secure in the standard model and combines a specific weak Programmable Hash Function [21] whose output lies in a pairing group and a Chameleon hash function [22]. This enables the simulation in our security proof for the scheme to handle the tricky queries for shared keys involving an honestly generated public key and an adversarially chosen public key. Similar ideas were used in the context of HIBE in [23]. We also make use of the pairing to provide a means of checking that public keys coming from the adversary are in some sense well-formed. We work with asymmetric pairings for efficiency at high security levels (and because it does not add any real complexity to the description of our scheme). The scheme's security relies on a natural variant of the Decisional Bilinear Diffie-Hellman (DBDH) assumption for the asymmetric setting.

From NIKE to PKE: In Section 5, we explore the connections between NIKE and public key encryption (PKE). That such connections exist should not be too much of a surprise: it is folklore that the ElGamal encryption scheme [24] can be seen as arising from the Diffie-Hellman NIKE scheme by making the sender's key pair (g^x, x) ephemeral and using the receiver's public key g^y to create the basis for a shared key g^{xy}. Similar connections were explored in the ID-based setting in [10].

In our setting with dishonest key registrations, we provide a simple, generic construction for PKE from NIKE that is also in the spirit of the original Diffie-Hellman–to–ElGamal conversion. The construction takes a NIKE scheme that is secure in our *CKS-light* model (with dishonest key registrations) and a strongly one-time secure signature scheme as inputs, and produces from these components a Key Encapsulation Mechanism (KEM) that we prove to be IND-CCA secure. A secure PKE from such a KEM can be obtained using standard results. At a high level, the key pair for the KEM is a randomly generated key pair (pk, sk) from the NIKE scheme, ciphertexts are also randomly generated public keys pk' from the NIKE scheme (together with a one-time signature that binds the public key to an identity), while the encapsulated key is the shared key computed from sk' and pk; the receiver computes the same key from sk and pk', assuming the one-time signature verifies. In order to prove the KEM to be IND-CCA secure, we exploit the presence of corrupt reveal queries in the NIKE security model in an essential way to handle certain decapsulation queries. The resulting KEM is almost as efficient as the underlying NIKE scheme.

The fact that secure NIKE implies IND-CCA-secure PKE, one of the most important primitives in cryptography, illustrates the fundamental role and utility of NIKE. We believe that this connection should spur further research on the topic.

2 Non-Interactive Key Exchange and Security Models

2.1 Non-Interactive Key Exchange

Following [6], we formally define a Non-Interactive Key Exchange (NIKE) scheme in the public key setting to be a collection of three algorithms: `CommonSetup`, `NIKE.KeyGen` and `SharedKey` together with an identity space $\mathcal{IDS}$ and a shared key space $\mathcal{SHK}$. Note that identities in the scheme and security model are merely used to track which public keys are associated with which users – we are *not* in the identity-based setting.

- `CommonSetup`: On input 1^k, outputs *params*, a set of system parameters.
- `NIKE.KeyGen`: On input *params* and an identity $\mathsf{ID} \in \mathcal{IDS}$, outputs a public key/secret key pair (pk, sk). This algorithm is probabilistic and can be executed by any user. We assume, without loss of generality, that *params* is included in pk.
- `SharedKey`: On input an identity $\mathsf{ID}_1 \in \mathcal{IDS}$ and a public key pk_1 along with another identity $\mathsf{ID}_2 \in \mathcal{IDS}$ and a secret key sk_2, outputs either a shared key in $\mathcal{SHK}$ for the two identities, or a failure symbol $\perp$. This algorithm is assumed to always output $\perp$ if $\mathsf{ID}_1 = \mathsf{ID}_2$.

For correctness, we require that, for any pair of identities $\mathsf{ID}_1, \mathsf{ID}_2$, and corresponding key pairs (pk_1, sk_1) and (pk_2, sk_2), algorithm `SharedKey` satisfies the constraint:

$$\texttt{SharedKey}(\mathsf{ID}_1, pk_1, \mathsf{ID}_2, sk_2) = \texttt{SharedKey}(\mathsf{ID}_2, pk_2, \mathsf{ID}_1, sk_1).$$

2.2 Definitions of Security for Non-Interactive Key Exchange

Cash, Kiltz and Shoup [6] proposed a security model for NIKE schemes in the public key setting, denoted here by the *CKS* model. This model abstracts away all considerations concerning certification and PKI in a particularly nice way. It allows an adversary to obtain honestly generated public keys, but also to then associate such public keys with other identities, and to register dishonestly generated public keys (for which the adversary need not know the corresponding private keys). This dishonest key registration (DKR) setting (abstractly) models a PKI where minimal assumptions are made about the actions of the Certificate Authority (CA): the CA is not assumed to check that a public key has not been previously registered to another user, and does not demand a proof of knowledge or possession of the private key when issuing a certificate on a public key. This conservative approach to modelling is fully appropriate given the great diversity

in how CAs operate in the real world. The model can be seen as a natural adaptation of the approach of Shoup [12] for modelling interactive key exchange to the NIKE setting and is analogous to the plain setting studied in [16,17].

However, there are some obvious omissions from the model, including the ability of an adversary to "corrupt" honestly generated public keys to learn the corresponding private keys, and the ability of a user to directly learn the key shared between two honest parties in the system (which could be possible, for example, because of cryptanalysis of a scheme making use of the shared key). Equivalent queries in the ID-based setting were permitted in the model introduced in [10].

For this reason, we augment the original *CKS* model with the "missing" queries, introducing the *m-CKS-heavy* model. We regard this as providing the "correct" model for NIKE. We also introduce two further models, the *CKS-heavy* and *CKS-light* models. These differ from *m-CKS-heavy* and the original *CKS* model only in the numbers and types of query that the adversary is allowed to make. Next we present in detail the *m-CKS-heavy* model. Then in Table 1 we summarize the differences between these security models in the DKR setting.

The m-CKS-heavy model: Our model is stated in terms of a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. In this game, $\mathcal{C}$ takes as input the security parameter 1^k, runs algorithm `CommonSetup` of the NIKE scheme and gives $\mathcal{A}$ *params*. The challenger takes a random bit b and answers oracle queries for $\mathcal{A}$ until $\mathcal{A}$ outputs a bit $\hat{b}$. The challenger answers the following types of queries for $\mathcal{A}$:

- *Register honest user ID*: $\mathcal{A}$ supplies an identity $\mathsf{ID} \in \mathcal{IDS}$. On input *params* and ID, the challenger runs `NIKE.KeyGen` to generate a public key/secret key pair (pk, sk) and records the tuple $(honest, \mathsf{ID}, pk, sk)$. The challenger returns pk to $\mathcal{A}$.
- *Register corrupt user ID*: In this type of query, $\mathcal{A}$ supplies both an identity $\mathsf{ID} \in \mathcal{IDS}$ and a public key pk. The challenger records the tuple $(corrupt, \mathsf{ID}, pk, \perp)$. We stress that $\mathcal{A}$ may make multiple "Register corrupt user ID" queries for the same ID during the experiment. In that case, only the most recent $(corrupt, \mathsf{ID}, pk, \perp)$ entry is kept.
- *Extract queries*: Here $\mathcal{A}$ supplies an identity ID that was registered as an honest user. The challenger looks for a tuple $(honest, \mathsf{ID}, pk, sk)$ containing ID and returns sk to $\mathcal{A}$.
- *Reveal queries*: Here $\mathcal{A}$ supplies a pair of registered identities $\mathsf{ID}_1, \mathsf{ID}_2$, subject only to the restriction that at least one of the two identities was registered as *honest*. The challenger runs `SharedKey` using the secret key of one of the *honest* identities and the public key of the other identity and returns the result to $\mathcal{A}$. Note that here the adversary is allowed to make reveal queries between two users that were originally registered as honest users. We denote by *honest reveal* the queries involving two honest users and by *corrupt reveal* the queries involving an honest user and a corrupt user.

Table 1. Types of queries for different security models in the dishonest key registration (DKR) PKI model (aka plain/bare model). Notation: ✓means that an adversary is allowed to make an arbitrary number of queries; ✗ means that no queries can be made; numbers represent the number of queries allowed to an adversary.

Model	Register Honest	Register Corrupt	Extract	Honest Reveal	Corrupt Reveal	Test
CKS-light	2	✓	✗	✗	✓	1
CKS	✓	✓	✗	✗	✓	✓
CKS-heavy	✓	✓	✓	✓	✓	1
m-CKS-heavy	✓	✓	✓	✓	✓	✓

– *Test queries*: Here $\mathcal{A}$ supplies two distinct identities $\mathsf{ID}_1, \mathsf{ID}_2$ that were both registered as honest. The challenger returns $\perp$ if $\mathsf{ID}_1 = \mathsf{ID}_2$. Otherwise, it uses the bit b to answer the queries. If $b = 0$, the challenger runs `SharedKey` using the public key for ID_1 and the secret key for ID_2 and returns the result to $\mathcal{A}$. If $b = 1$, the challenger generates a random key, records it for later, and returns that key to the adversary. In this case, to keep things consistent, the challenger returns the same random key for the pair $\mathsf{ID}_1, \mathsf{ID}_2$ every time $\mathcal{A}$ queries for their paired key, in either order.

$\mathcal{A}$'s queries may be made adaptively and are arbitrary in number. To prevent trivial wins for the adversary, no query to the *reveal* oracle is allowed on any pair of identities selected for *test* queries (in either order), and no *extract* query is allowed on any of the identities involved in *test* queries. Also, we demand that no identity registered as corrupt can later be the subject of a *register honest user ID* query, and vice versa.

When the adversary finally outputs $\hat{b}$, it wins the game if $\hat{b} = b$. For an adversary $\mathcal{A}$, we define its advantage in this security game as:

$$\mathrm{Adv}^{\text{m-CKS-heavy}}_{\mathcal{A}}(k, q_H, q_C, q_E, q_{HR}, q_{CR}, q_T) = |\Pr[\hat{b} = b] - 1/2|$$

where q_H, q_C, q_E, q_{HR}, q_{CR} and q_T are the numbers of *register honest user ID* queries, *register corrupt user ID* queries, *extract* queries, *honest reveal* queries, *corrupt reveal* queries and *test* queries made by $\mathcal{A}$, respectively. We say that a NIKE scheme is $(t, \epsilon, q_H, q_C, q_E, q_{HR}, q_{CR}, q_T)$-secure in the *m-CKS-heavy* model if there is no adversary with advantage at least ϵ that runs in time t and makes at most q_H *register honest user ID* queries, etc. Informally, we say that a NIKE scheme is *m-CKS-heavy secure* if there is no efficient adversary having non-negligible advantage in k, where efficient means that the running time and numbers of queries made by the adversary are bounded by polynomials in k.

Comparing the models: Table 1 outlines the properties of our other security models in the DKR setting, in terms of restrictions on the queries that can be made by the adversary. It is apparent that the *m-CKS-heavy* model is the strongest model. It differs from the *CKS-heavy* model only in allowing multiple

test queries. The *m-CKS-heavy* model represents a strengthening of the original *CKS* model by allowing *extract* and *honest reveal* queries, whereas the *CKS* model only allows the adversary to gain information about honestly generated shared keys via *test* queries. The *CKS-light* model is simplest of all, involving only two honestly registered identities, removing the *extract* and *honest reveal* queries, and allowing only a single *test* query. We prove that it is polynomially equivalent to the *m-CKS-heavy* model. In fact, we prove the following theorem:

Theorem 1. *The* m-CKS-heavy*,* CKS-heavy*,* CKS *and* CKS-light *security models are all polynomially equivalent.*

Proof. See the full version [25].

Thus, while the *m-CKS-heavy* model is our preferred model, it suffices to analyse schemes in the *CKS-light* model if one is not overly concerned about concrete security. However, we note that various factors are involved in the reductions. In particular a factor of $q_T {q_H}^2$ is lost in going from the *m-CKS-heavy* to the *CKS-light* model. This reflects the proof techniques used in establishing the bounds, specifically the use of hybrid arguments. It is an interesting open problem to either prove tighter relations between the models, or to prove that such results are not possible.

3 Intractability Assumptions

3.1 The Group of Signed Quadratic Residues, the BBS generator, and the Strong Diffie-Hellman Assumption

The factoring assumption: Let $n(k)$ be a function and δ a constant with $0 \leq \delta < 1/2$. Let $\mathtt{RSAgen}$ be an algorithm with input 1^k that generates elements (N, P, Q) such that $N = PQ$ is an n-bit Blum integer and all prime factors of $\phi(N)/4$ are pairwise distinct and have at least δn bits. These conditions ensure that $(\mathbb{J}_N, \cdot)$ is cyclic and that the square g of a random element in $\mathbb{Z}_N^*$, generates $\mathbb{QR}_N$ with high probability. That is, $\langle g \rangle = \mathbb{QR}_N$. For such N, we recall the definition of the *group of signed quadratic residues* $\mathbb{QR}_N^+$ from [20] (see also [26,27]) which is defined as the set $\{|x| : x \in \mathbb{QR}_N\}$, where $|x|$ is the absolute value when representing elements of $\mathbb{Z}_N$ as the set $\{-(N-1)/2, \ldots, (N-1)/2\}$. $(\mathbb{QR}_N^+, \cdot)$ is a cyclic group of order $\phi(N)/4$ whose elements are efficiently recognisable given only N as input.

For any algorithm $\mathcal{A}$, we write

$$\mathrm{Adv}^{\mathrm{fac}}_{\mathcal{A},\mathtt{RSAgen}}(k) = \Pr[\{P, Q\} \xleftarrow{\$} \mathcal{A}(N) : (N, P, Q) \xleftarrow{\$} \mathtt{RSAgen}(1^{\mathtt{k}})].$$

The factoring assumption for $\mathtt{RSAgen}$ is that $\mathrm{Adv}^{\mathrm{fac}}_{\mathcal{A},\mathtt{RSAgen}}(k)$ is negligible for all PPT algorithms $\mathcal{A}$.

The BBS generator: Let $\mathrm{BBS}_N : \mathbb{QR}_N^+ \to \{0,1\}^k$ be the Blum-Blum-Shub pseudorandom number generator. (That is, $\mathrm{BBS}_N(X) = (\mathrm{lsb}_N(X), \mathrm{lsb}_N(X^2), \ldots, \mathrm{lsb}_N(X^{2^{k-1}}))$, where $\mathrm{lsb}_N(X)$ denotes the least significant bit of $X \in \mathbb{QR}_N^+$.) Recall that the factoring assumption implies the computational indistinguishability of the distributions

$$(N, X^{2^k}, \mathrm{BBS}_N(X)) \text{ and } (N, X^{2^k}, R),$$

where $N \xleftarrow{\$} \mathtt{RSAgen}(1^{\mathtt{k}})$, and $X \xleftarrow{\$} \mathbb{QR}_N^+$ and $R \xleftarrow{\$} \{0,1\}^k$ are chosen uniformly. (See also [28, Theorem 2] for a summary why this holds.) Concretely, under the factoring assumption, the advantage

$$\mathrm{Adv}^{\mathrm{BBS}}_{\mathcal{B},\mathtt{RSAgen}}(k) := \left|\Pr[\mathcal{B}(N, X^{2^k}, \mathrm{BBS}_N(X)) = 1] - \Pr[\mathcal{B}(N, X^{2^k}, R) = 1]\right|$$

is negligible for any PPT adversary $\mathcal{B}$.

The Strong DH assumption: In [20] it is shown that if the factoring assumption holds, then the *Strong DH assumption* holds relative to RSAgen. This assumption is that there is no PPT algorithm having non-negligible advantage in solving the CDH problem on input (N, g, X, Y) when given an oracle for $\mathrm{DDH}_{g,X}(\cdot,\cdot)$. Here g is a randomly selected generator of $\mathbb{QR}_N^+$, X and Y are selected uniformly from $\mathbb{QR}_N^+$, the solution to the CDH problem is defined as $g^{(\mathrm{dlog}_g X)(\mathrm{dlog}_g Y)}$, and the DDH oracle $\mathrm{DDH}_{g,X}(\hat{Y}, \hat{Z})$ returns 1 if $\hat{Y}^{\mathrm{dlog}_g X} = \hat{Z}$ and 0 otherwise.

We will require a variant of the Strong DH assumption, which we name the *Double Strong DH (DSDH)* assumption. This can be stated as follows. Let $(N, P, Q) \leftarrow \mathtt{RSAgen}(1^{\mathtt{k}})$ and let g be a randomly selected generator of $\mathbb{QR}_N^+$, and X, Y be selected uniformly from $\mathbb{QR}_N^+$. Then the Double Strong DH problem is to solve the CDH problem on input (N, g, X, Y), that is to compute $g^{(\mathrm{dlog}_g X)(\mathrm{dlog}_g Y)}$, when given oracles for $\mathrm{DDH}_{g,X}(\cdot,\cdot)$ and $\mathrm{DDH}_{g,Y}(\cdot,\cdot)$. The DSDH assumption relative to RSAgen is that there is no PPT algorithm having non-negligible advantage in solving this problem.

Theorem 2. *If the factoring assumption holds relative to* RSAgen*, then the DSDH assumption also holds relative to* RSAgen*. In particular, for every algorithm $\mathcal{A}$ solving the Double Strong DH problem, there exists a factoring algorithm $\mathcal{B}$ (with roughly the same running time as $\mathcal{A}$) such that*

$$\mathrm{Adv}^{\mathrm{dsdh}}_{\mathcal{A},\mathtt{RSAgen}}(k) \leq \mathrm{Adv}^{\mathrm{fac}}_{\mathcal{B},\mathtt{RSAgen}}(k) + O(2^{-\delta n(k)}).$$

Proof. The original proof of [20, Theorem 2] shows how to handle a single DDH oracle $\mathrm{DDH}_{g,X}(\cdot,\cdot)$. By symmetry of the set-up used in the proof, the same procedure can also be used to (simultaneously) handle the oracle $\mathrm{DDH}_{g,Y}(\cdot,\cdot)$.

3.2 Parameter Generation Algorithms for Asymmetric Pairings

Our pairing based scheme will be parameterized by a *type 2 pairing parameter generator*, denoted by $\mathcal{G}2$. This is a polynomial time algorithm that on input

a security parameter 1^k, returns the description of three multiplicative cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of the same prime order p, generators g_1, g_2 for $\mathbb{G}_1$, $\mathbb{G}_2$ respectively, and a bilinear non-degenerate and efficiently computable pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We assume that $\mathcal{G}2$ also outputs the description of an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ and that $g_1 = \psi(g_2)$. Throughout, we write $\mathcal{PG}2 = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, p, e, \psi)$ for a set of groups and other parameters with the properties just described.

3.3 The Decisional Bilinear Diffie-Hellman Assumption for Type 2 Pairings (DBDH-2)

Let $\mathcal{PG}2 = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, p, e, \psi)$ as above. We consider the following version of the Decisional Bilinear Diffie-Hellman problem for type 2 pairings, as introduced by Galindo in [29]: Given $(g_2, g_2^a, g_2^b, g_1^c, T) \in \mathbb{G}_2^3 \times \mathbb{G}_1 \times \mathbb{G}_T$ as input, the problem is to decide whether or not $T = e(g_1, g_2)^{abc}$, where $g_1 = \psi(g_2)$. More formally, we associate the following experiment to a type 2 pairing parameter generator $\mathcal{G}2$ and an adversary $\mathcal{B}$.

$$
\begin{array}{l}
\text{Experiment } \mathrm{Exp}^{\text{dbdh-2}}_{\mathcal{B},\mathcal{G}2}(k) \\
\quad \mathcal{PG}2 \xleftarrow{\$} \mathcal{G}2(1^k) \\
\quad a, b, c, z \xleftarrow{\$} \mathbb{Z}_p \\
\quad \beta \xleftarrow{\$} \{0,1\} \\
\quad \text{If } \beta = 1 \text{ then } T \leftarrow e(g_1, g_2)^{abc} \text{ else } T \leftarrow e(g_1, g_2)^z \\
\quad \beta' \xleftarrow{\$} \mathcal{B}(1^k, \mathcal{PG}2, g_2^a, g_2^b, g_1^c, T) \\
\quad \text{If } \beta = \beta' \text{ then return 0 else return 1}
\end{array}
$$

The advantage of $\mathcal{B}$ in the above experiment is defined as

$$
\mathrm{Adv}^{\text{dbdh-2}}_{\mathcal{B},\mathcal{G}2}(k) = \left| \Pr[\mathrm{Exp}^{\text{dbdh-2}}_{\mathcal{B},\mathcal{G}2}(k) = 1] - \frac{1}{2} \right|.
$$

We say that the DBDH-2 assumption relative to $\mathcal{G}2$ holds if $\mathrm{Adv}^{\text{dbdh-2}}_{\mathcal{B},\mathcal{G}2}$ is negligible in k for all PPT algorithms $\mathcal{B}$.

4 Constructions for Non-Interactive Key Exchange

4.1 A Construction in the Random Oracle Model from Factoring

We specify how to build a NIKE scheme, $\mathrm{NIKE}_{\mathrm{fac}}$, that is secure in the *CKS-light* security model under the factoring assumption relative `RSAgen` in the ROM. Our scheme makes use of a hash function $H : \{0,1\}^* \to \{0,1\}^k$ which is modelled as a random oracle in the security proof. The component algorithms of the scheme $\mathrm{NIKE}_{\mathrm{fac}}$ are defined as follows:

$\underline{\texttt{CommonSetup}(1^k)}$
$(N, P, Q) \xleftarrow{\$} \texttt{RSAgen}(1^k)$
$g \xleftarrow{\$} \mathbb{QR}_N^+$, where $\langle g \rangle = \mathbb{QR}_N^+$
$params \leftarrow (H, N, g)$
Return $params$

$\underline{\texttt{NIKE.KeyGen}(params, \mathsf{ID})}$
$x \xleftarrow{\$} \mathbb{Z}_{\lfloor N/4 \rfloor}$;
$X \leftarrow g^x$
$pk \leftarrow X$; $sk \leftarrow x$
Return (pk, sk)

$\underline{\texttt{SharedKey}(\mathsf{ID}_1, pk_1, \mathsf{ID}_2, sk_2)}$
If $(\mathsf{ID}_1 = \mathsf{ID}_2)$ or $pk_1 \notin \mathbb{QR}_N^+$ or $pk_2 \notin \mathbb{QR}_N^+$ return $\perp$
else if $\begin{cases} \mathsf{ID}_1 < \mathsf{ID}_2 \text{ return } H(\mathsf{ID}_1, \mathsf{ID}_2, {pk_1}^{sk_2}) \\ \mathsf{ID}_2 < \mathsf{ID}_1 \text{ return } H(\mathsf{ID}_2, \mathsf{ID}_1, {pk_1}^{sk_2}) \end{cases}$

Here we are assuming that the identities ID come from a space with a natural ordering $<$.

Theorem 3. *The scheme* NIKE_{fac} *is secure in the ROM under the factoring assumption relative to* $\texttt{RSAgen}$. *In particular, suppose* $\mathcal{A}$ *is an adversary against* NIKE_{fac} *in the* CKS-light *security model. Then there exists a factoring adversary* $\mathcal{C}$ *with:*

$$\text{Adv}^{\text{CKS-light}}_{\mathcal{A},\text{NIKE}_{\text{fac}}}(k) \leq \text{Adv}^{\text{fac}}_{\mathcal{C},\texttt{RSAgen}}(k) + O(2^{-\delta n(k)}).$$

Proof. See the full version [25].

4.2 Towards a Factoring-Based Scheme in the Standard Model

The security proof of NIKE_{fac} above crucially uses the statistical properties of the random oracle H. If we accept an *interactive* key registration, we can however give a factoring-based NIKE scheme in the standard model. The basis of this scheme is the factoring-based IND-CCA secure encryption scheme of Hofheinz and Kiltz [28]. However, in adapting their scheme to the NIKE setting, we will have to find a way to *simultaneously* cope with two challenge ciphertexts (which correspond to the public keys of the challenge identities). To cope with this modified setting, we will set up a simulation that is able to decrypt all but *two* ciphertexts (resp. NIKE public keys).

In our description, let $\texttt{RSAgen}$ as before, let $\text{ChamH} : \{0,1\}^* \times \mathcal{R}_{\text{Cham}} \to \mathbb{Z}_{2^k}$ be a chameleon hash function [22]. Now consider the following scheme $\text{NIKE}_{\text{fac-int}}$:

$\underline{\texttt{CommonSetup}(1^k)}$
$(N, P, Q) \xleftarrow{\$} \texttt{RSAgen}(1^k)$
$g, u_0, u_1, u_2 \xleftarrow{\$} \mathbb{QR}_N^+$,
where $\langle g \rangle = \mathbb{QR}_N^+$
$\mathsf{hk}, \mathsf{ck} \xleftarrow{\$} \texttt{Cham.KeyGen}(1^k)$
$params \leftarrow (N, g, u_0, u_1, u_2, \mathsf{hk})$
Return $params$

$\underline{\texttt{NIKE.KeyGen}(params, \mathsf{ID})}$
$x \xleftarrow{\$} \mathbb{Z}_{\lfloor N/4 \rfloor}$; $r \xleftarrow{\$} \mathcal{R}_{\text{Cham}}$
$Z \leftarrow g^{x \cdot 2^{3k}}$;
$t \leftarrow \text{ChamH}_{\mathsf{hk}}(Z || \mathsf{ID}; r)$
$Y \leftarrow u_0 u_1^t {u_2}^{t^2}$; $X \leftarrow Y^x$
$pk \leftarrow (Z, X, r)$; $sk \leftarrow x$
Return (pk, sk)

$\texttt{SharedKey}(\mathsf{ID}_1, pk_1, \mathsf{ID}_2, sk_2)$
If $(\mathsf{ID}_1 = \mathsf{ID}_2)$ or $pk_1 \notin \mathbb{QR}_N^+ \times \mathbb{QR}_N^+ \times \mathcal{R}_{\text{Cham}}$ or $sk_2 \notin \mathbb{Z}_{\lfloor N/4 \rfloor}$ return $\perp$
Parse $pk_1 =: (Z_1, X_1, r_1)$ and $sk_2 =: x_2$
Return $\text{BBS}_N(Z_1^{x_2 \cdot 2^{2^k}})$

Note that correctness of the scheme follows from $Z_1^{x_2 \cdot 2^{2^k}} = g^{x_1 \cdot x_2 \cdot 2^{5k}} = Z_2^{x_1 \cdot 2^{2^k}}$. To prove security, we need to rely on the *consistency* of public keys. Concretely, the security reduction we will give can only authentically answer corrupt reveal queries for corrupt user keys $pk = (Z, X, r)$ that satisfy $Z = g^{x \cdot 2^{3k}}, X = (u_0 u_1^t u_2^{t^2})^x$ for $t = \text{ChamH}_{\mathsf{hk}}(Z||\mathsf{ID}; r)$ and some x. Unlike in our upcoming pairing-based scheme, this kind of consistency is not (obviously) efficiently verifiable. Hence, the key registration process must ensure that only consistent user keys are registered, e.g., by having the user prove consistency in zero-knowledge (interactively, using x as witness).

On top of assuming consistent keys, we will also have to make an assumption about the distribution of (or rather, the ability to generate) primes. Namely, we will need to assume a PPT algorithm $\texttt{PrimeGen}$ that, on input a $2k$-bit prime ρ, outputs a prime α such that $\alpha \bmod \rho$ has statistical distance $O(2^{-k})$ from the uniform distribution over $\mathbb{Z}_\rho$. Such an algorithm $\texttt{PrimeGen}$ exists. This is an easy consequence of Dirichlet's theorem on the distribution of primes in arithmetic progressions: our generator simply samples integers of the form $\alpha_0 + i \cdot \rho$ for uniformly chosen $\alpha_0 \in \mathbb{Z}_\rho$ and $i = 1, 2, \ldots$, and checks them for primality. This algorithm can be rigorously proven to be efficient under the Generalized Riemann Hypothesis.

Theorem 4. *Under the factoring assumption relative to* $\texttt{RSAgen}$*, given an algorithm* $\texttt{PrimeGen}$ *as above, and assuming that the chameleon hash function ChamH is collision-resistant, the scheme* $\text{NIKE}_{\text{fac-int}}$ *is secure against all adversaries that only register consistent (in the sense above) user keys. In particular, suppose* $\mathcal{A}$ *is such an adversary against* NIKE_{fac} *in the* CKS-light *security model. Then there exists a BBS distinguisher* $\mathcal{B}$ *and a collision-finder* $\mathcal{A}_{\mathcal{CH}}$ *with:*

$$\text{Adv}^{\text{CKS-light}}_{\mathcal{A},\text{NIKE}_{\text{fac-int}}}(k) \leq \text{Adv}^{\text{BBS}}_{\mathcal{B},\texttt{RSAgen}}(k) + \text{Adv}^{\text{coll}}_{\mathcal{A}_{\mathcal{CH}},\text{ChamH}}(k) + O(2^{-k}). \quad (1)$$

Proof. See the full version [25].

4.3 A Construction in the Standard Model from Pairings

We specify how to build a NIKE scheme, $\text{NIKE}_{\text{dbdh-2}}$, that is secure in the *CKS-light* security model under the DBDH-2 assumption in the standard model. Our construction makes use of a tuple $\mathcal{PG}2 = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, p, e, \psi)$, output by a parameter generator $\mathcal{G}2$, and a chameleon hash function $\text{ChamH} : \{0,1\}^* \times \mathcal{R}_{\text{Cham}} \to \mathbb{Z}_p$. This can be instantiated efficiently using the discrete-log based construction from [22]. The component algorithms of the scheme $\text{NIKE}_{\text{dbdh-2}}$ are defined as follows:

$\texttt{CommonSetup}(1^k)$
$\mathcal{PG}2 \xleftarrow{\$} \mathcal{G}2(1^k)$,
where $\mathcal{PG}2 = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, p, e, \psi)$
$u_0, u_1, u_2, S \xleftarrow{\$} \mathbb{G}_1^*$
$\mathsf{hk}, \mathsf{ck} \xleftarrow{\$} \texttt{Cham.KeyGen}(1^k)$
$params \leftarrow (\mathcal{PG}2, u_0, u_1, u_2, S, \mathsf{hk})$
Return $params$

$\texttt{NIKE.KeyGen}(params, \mathsf{ID})$
$x \xleftarrow{\$} \mathbb{Z}_p$; $r \xleftarrow{\$} \mathcal{R}_{\text{Cham}}$
$Z \leftarrow g_2^x$;
$t \leftarrow \text{ChamH}_{\mathsf{hk}}(Z||\mathsf{ID}; r)$;
$Y \leftarrow u_0 u_1^t {u_2}^{t^2}$; $X \leftarrow Y^x$
$pk \leftarrow (X, Z, r)$; $sk \leftarrow x$
Return (pk, sk)

$\texttt{SharedKey}(\mathsf{ID}_1, pk_1, \mathsf{ID}_2, sk_2)$
If $\mathsf{ID}_1 = \mathsf{ID}_2$ return $\perp$
Parse pk_1 as (X_1, Z_1, r_1) and sk_2 as x_2
$t_1 \leftarrow \text{ChamH}_{\mathsf{hk}}(Z_1||\mathsf{ID}_1; r_1)$
If $e(X_1, g_2) \neq e(u_0 u_1^{t_1} {u_2}^{t_1^2}, Z_1)$
then $K_{1,2} \leftarrow \perp$
else $K_{1,2} \leftarrow e(S^{x_2}, Z_1)$
Return $K_{1,2}$

The check in the SharedKey algorithm for valid public keys can be implemented by evaluating the bilinear map twice. It is clear that SharedKey defined in this way satisfies the requirement that entities ID_1 and ID_2 are able to compute a common key. To see this, note that $e(S^{x_2}, Z_1) = e(S, g_2)^{x_1, x_2}$. The identity space for this construction, $\mathcal{IDS}$, is $\{0,1\}^*$, while the space of shared keys is $\mathcal{SHK} = \mathbb{G}_T$. Public keys and parameters are compact. For example, at the 128-bit security level, using BN curves [30] and point compression, public keys consist of 768 bits plus an element from $\mathcal{R}_{\text{Cham}}$.

As stated before, we can prove the above NIKE scheme to be secure under the DBDH-2 assumption in the sense of the *CKS-light* security model. Interestingly, our scheme can be generalised to use any *weak* $(2, \text{poly})$-PHF [21] in combination with a chameleon hash function. That is, Y (in the NIKE.KeyGen algorithm) would be the output of the *weak* $(2, \text{poly})$-PHF on input t, where t is the output of the chameleon hash function. We have given a specific construction here because suitable weak PHFs are currently rare. A further generalisation of our scheme could use any randomised $(2, \text{poly})$-PHF and avoid the chameleon hash, but no constructions for these are currently known.

Theorem 5. *Assume* ChamH *is a family of chameleon hash functions. Then* $\text{NIKE}_{\text{dbdh-2}}$ *is secure under the DBDH-2 assumption relative to generator* $\mathcal{G}2$. *In particular, suppose* $\mathcal{A}$ *is an adversary against* $\text{NIKE}_{\text{dbdh-2}}$ *in the* CKS-light *security model. Then there exists a DBDH-2 adversary* $\mathcal{B}$ *with:*

$$\text{Adv}^{\text{dbdh-2}}_{\mathcal{B},\mathcal{G}2}(k) \geq \text{Adv}^{\text{CKS-light}}_{\mathcal{A},\text{NIKE}_{\text{dbdh-2}}}(k) - \text{Adv}^{\text{coll}}_{\mathcal{A}_{\mathcal{CH}},\text{ChamH}}(k).$$

Proof. See the full version [25].

5 From Non-Interactive Key Exchange to Public Key Encryption

We give a conversion that takes a NIKE scheme that is secure in the *CKS-light* security model plus a strongly one-time secure signature (OTS) scheme, and produces from it a KEM that is IND-CCA secure. From such a KEM, it is easy to construct an IND-CCA secure public key encryption scheme [31].

The formal definitions of KEM and OTS schemes and their security can be found in the full version [25].

5.1 The Conversion from NIKE to KEM

We now present our conversion from a NIKE scheme to a KEM. For a NIKE scheme $\texttt{NIKE}$ and an OTS scheme $\texttt{OTS}$, we construct a KEM $\texttt{KEM}(\texttt{NIKE}, \texttt{OTS})$ with the following algorithms:

- $\texttt{KEM.KeyGen}(1^k)$: This algorithm runs the algorithm $\texttt{CommonSetup}(1^k)$ of $\texttt{NIKE}$ to obtain a set of system parameters, *params*. Then it picks $\mathsf{ID} \in \mathcal{IDS}$ uniformly and runs $\texttt{NIKE.KeyGen}(params, \mathsf{ID})$ to obtain a key pair (pk, sk). It sets $pk_{\texttt{KEM}} = (params, \mathsf{ID}, pk)$ and $sk_{\texttt{KEM}} = (\mathsf{ID}, sk)$.
- $\texttt{Enc}(pk_{\texttt{KEM}})$: This algorithm parses $pk_{\texttt{KEM}}$ as $(params, \mathsf{ID}, pk)$, runs $\texttt{OTSKeyGen}$ to obtain a pair $(vk, sigk)$. This is repeated until $vk \neq \mathsf{ID}$. Next, it runs $\texttt{NIKE.KeyGen}(params, \mathsf{ID}' = vk)$ of $\texttt{NIKE}$ to obtain a key pair (pk', sk') and runs $\texttt{OTSSign}(sigk, pk')$ to obtain σ, a signature on pk'. It then runs $\texttt{SharedKey}(\mathsf{ID}, pk, \mathsf{ID}' = vk, sk')$ of scheme $\texttt{NIKE}$ to obtain a key $K \in \mathcal{SHK}$. The output is $(K, C = (vk, pk', \sigma))$.
- $\texttt{Dec}(sk_{\texttt{KEM}}, C)$: This algorithm first parses C as (vk, pk', σ) and $sk_{\texttt{KEM}}$ as (ID, sk). Next, it runs $\texttt{OTSVfy}(vk, pk', \sigma)$ and returns $\perp$ if the output is $\texttt{reject}$ or if $vk = \mathsf{ID}$. Otherwise, it runs $\texttt{SharedKey}(\mathsf{ID}' = vk, pk', \mathsf{ID}, sk)$ and outputs the result, which may be $\perp$.

Notice that the ciphertexts in this scheme consist of a verification key from the OTS scheme, a public key from the NIKE scheme, and a one-time signature, while the encapsulated keys are elements of $\mathcal{SHK}$. As our next result shows, the resulting KEM is automatically IND-CCA secure if the NIKE scheme is secure in the *CKS-light* security model.

Theorem 6. *Suppose the NIKE scheme* $\texttt{NIKE}$ *is secure in the* CKS-light *security model and* $\texttt{OTS}$ *is a strongly secure one-time signature scheme. Then* $\texttt{KEM}(\texttt{NIKE}, \texttt{OTS})$ *is an IND-CCA secure KEM. More precisely, for any adversary* $\mathcal{A}$ *against* $\texttt{KEM}(\texttt{NIKE}, \texttt{OTS})$*, there exists an adversary* $\mathcal{B}$ *against* $\texttt{NIKE}$ *in the* CKS-light *security model or an adversary* $\mathcal{C}$ *against* $\texttt{OTS}$ *having the same advantage. Moreover, if* $\mathcal{A}$ *makes* q_D *decapsulation queries, then* $\mathcal{B}$ *makes* q_D *register corrupt user queries and* q_D *corrupt reveal queries, while* $\mathcal{B}$*'s running time is roughly the same as that of* $\mathcal{A}$*.*

Proof. See the full version [25].

Applying the above construction to the pairing-based NIKE scheme from the previous section results in an IND-CCA secure KEM with public keys (ID, pk) that consist of an identity string, two group elements (one in $\mathbb{G}_1$ and one in $\mathbb{G}_2$), and a key for the Chameleon hash function. Ciphertexts are slightly longer, containing in addition a verification key and a signature from the one-time signature scheme[3].

6 Conclusions and Open Problems

We provided different security models for NIKE and explored the relationships between them. We then gave constructions for secure NIKE in the ROM and in the standard model. We also studied the relationship between NIKE and PKE, showing that a secure NIKE implies an IND-CCA secure PKE scheme.

There are several interesting open problems that arise from our work. One is to construct pairing-free NIKE schemes in the standard model. A challenge to doing so is that our pairing-based construction uses the pairing in a fundamental way in order to provide a publicly computable check on the validity of public keys. The RSA/factoring setting seems particularly challenging in this respect – we recall that our standard model, factoring-based scheme required that the adversary only register valid public keys, a condition that could be enforced in practice by having an interactive key registration protocol and insisting on proofs of validity during that protocol. Clearly, it is desirable from both a practical and a theoretical perspective to obtain schemes that are secure in the plain setting, where no such protocol is required.

Another open problem is to construct ID-based NIKE schemes that are provably secure in the standard model, moving beyond the ROM schemes analysed in [8,10]. Starting with known IBE schemes may be profitable, but the fact that these generally have randomised private key generation algorithms seems to make it hard to work backwards from IBE to ID-based NIKE.

Finally, it would be interesting to consider three-party NIKE schemes based on Joux's protocol [32]. Currently, there is no security model for such schemes, and no constructions which can handle adversarially-generated public keys.

References

1. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
2. Çapar, Ç., Goeckel, D., Paterson, K.G., Quaglia, E.A., Towsley, D., Zafer, M.: Signal-flow-based analysis of wireless security protocols. Information and Computation (to appear)
3. Dodis, Y., Katz, J., Smith, A., Walfish, S.: Composability and On-Line Deniability of Authentication. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 146–162. Springer, Heidelberg (2009)

[3] Arguably, one might also include the public parameters *params* when evaluating the public key size.

4. Boyd, C., Mao, W., Paterson, K.G.: Key Agreement Using Statically Keyed Authenticators. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 248–262. Springer, Heidelberg (2004)
5. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
6. Cash, D., Kiltz, E., Shoup, V.: The Twin Diffie-Hellman Problem and Applications. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 127–145. Springer, Heidelberg (2008)
7. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. In: The 2000 Symposium on Cryptography and Information Security, pp. 26–28 (2000)
8. Dupont, R., Enge, A.: Provably secure non-interactive key distribution based on pairings. Discrete Applied Mathematics 154(2), 270–276 (2006)
9. Gennaro, R., Halevi, S., Krawczyk, H., Rabin, T., Reidt, S., Wolthusen, S.D.: Strongly-Resilient and Non-interactive Hierarchical Key-Agreement in MANETs. In: Jajodia, S., Lopez, J. (eds.) ESORICS 2008. LNCS, vol. 5283, pp. 49–65. Springer, Heidelberg (2008)
10. Paterson, K.G., Srinivasan, S.: On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups. Des. Codes Cryptography 52(2), 219–241 (2009)
11. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
12. Shoup, V.: On formal models for secure key exchange (version 4) (1999)
13. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure against Dictionary Attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
14. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
15. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
16. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) ACM Conference on Computer and Communications Security, pp. 390–399. ACM (2006)
17. Ristenpart, T., Yilek, S.: The Power of Proofs-of-Possession: Securing Multiparty Signatures against Rogue-Key Attacks. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 228–245. Springer, Heidelberg (2007)
18. Boldyreva, A.: Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2002)
19. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential Aggregate Signatures and Multisignatures Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006)
20. Hofheinz, D., Kiltz, E.: The Group of Signed Quadratic Residues and Applications. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 637–653. Springer, Heidelberg (2009)
21. Hofheinz, D., Jager, T., Kiltz, E.: Short Signatures from Weaker Assumptions. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 647–666. Springer, Heidelberg (2011)

22. Krawczyk, H., Rabin, T.: Chameleon signatures. In: NDSS (2000)
23. Chatterjee, S., Sarkar, P.: Generalization of the Selective-ID Security Model for HIBE Protocols. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 241–256. Springer, Heidelberg (2006)
24. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31(4), 469–472 (1985)
25. Freire, E.S.V., Hofheinz, D., Kiltz, E., Paterson, K.G.: Non-interactive key exchange. Cryptology ePrint Archive, Report 2012/xxx (2012), http://eprint.iacr.org/
26. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing 18(1), 186–208 (1989)
27. Fischlin, R., Schnorr, C.P.: Stronger security proofs for RSA and Rabin bits. Journal of Cryptology 13(2), 221–244 (2000)
28. Hofheinz, D., Kiltz, E.: Practical Chosen Ciphertext Secure Encryption from Factoring. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 313–332. Springer, Heidelberg (2009)
29. Galindo, D.: Boneh-Franklin Identity Based Encryption Revisited. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 791–802. Springer, Heidelberg (2005)
30. 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)
31. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing 33, 167–226 (2003)
32. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. In: Bosma, W. (ed.) ANTS 2000. LNCS, vol. 1838, pp. 385–394. Springer, Heidelberg (2000)

Efficient UC-Secure Authenticated Key-Exchange for Algebraic Languages

Fabrice Ben Hamouda[1], Olivier Blazy[2], Céline Chevalier[3], David Pointcheval[1], and Damien Vergnaud[1]

[1] ENS, Paris, France*
[2] Ruhr-Universität Bochum, Germany
[3] Université Panthéon-Assas, Paris, France

Abstract. *Authenticated Key Exchange* (AKE) protocols enable two parties to establish a shared, cryptographically strong key over an insecure network using various authentication means, such as cryptographic keys, short (*i.e.*, low-entropy) secret keys or *credentials*. In this paper, we provide a general framework, that encompasses several previous AKE primitives such as *(Verifier-based) Password-Authenticated Key Exchange* or *Secret Handshakes*, we call *LAKE* for *Language-Authenticated Key Exchange*.

We first model this general primitive in the *Universal Composability* (UC) setting. Thereafter, we show that the Gennaro-Lindell approach can efficiently address this goal. But we need *smooth projective hash functions* on new languages, whose efficient implementations are of independent interest. We indeed provide such hash functions for languages defined by combinations of linear pairing product equations.

Combined with an efficient commitment scheme, that is derived from the highly-efficient UC-secure Lindell's commitment, we obtain a very practical realization of Secret Handshakes, but also *Credential-Authenticated Key Exchange protocols*. All the protocols are UC-secure, in the standard model with a common reference string, under the classical Decisional Linear assumption.

1 Introduction

The main goal of an *Authenticated Key Exchange* (AKE) protocol is to enable two parties to establish a shared cryptographically strong key over an insecure network under the complete control of an adversary. AKE is one of the most widely used and fundamental cryptographic primitives. In order for AKE to be possible, the parties must have authentication means, *e.g.* (public or secret) cryptographic keys, short (*i.e.*, low-entropy) secret keys or *credentials* that satisfy a (public or secret) policy.

Motivation. PAKE, for *Password-Authenticated Key Exchange*, was formalized by Bellovin and Merritt [5] and followed by many proposals based on different

* ENS, CNRS & INRIA – UMR 8548.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 272–291, 2013.

cryptographic assumptions (see [1, 8] and references therein). It allows users to generate a strong cryptographic key based on a shared "human-memorable" (*i.e.* low-entropy) password without requiring a public-key infrastructure. In this setting, an adversary controlling all communication in the network should not be able to mount an off-line dictionary attack.

The concept of *Secret Handshakes* has been introduced in 2003 by Balfanz, Durfee, Shankar, Smetters, Staddon and Wong [3] (see also [2,19]). It allows two members of the same group to identify each other secretly, in the sense that each party reveals his affiliation to the other only if they are members of the same group. At the end of the protocol, the parties can set up an ephemeral session key for securing further communication between them and an outsider is unable to determine if the handshake succeeded. In case of failure, the players do not learn any information about the other party's affiliation.

More recently, *Credential-Authenticated Key Exchange* (CAKE) was presented by Camenisch, Casati, Groß and Shoup [8]. In this primitive, a common key is established if and only if a specific relation is satisfied between credentials hold by the two players. This primitive includes variants of PAKE and Secret Handshakes, and namely Verifier-based PAKE, where the client owns a password pw and the server knows a one-way transformation v of the password only. It prevents massive password recovering in case of server corruption. The two players eventually agree on a common high entropy secret if and only if pw and v match together, and off-line dictionary attacks are prevented for third-party players.

Our Results. We propose a new primitive that encompasses most of the previous notions of authenticated key exchange. It is closely related to CAKE and we call it LAKE, for *Language-Authenticated Key-Exchange*, since parties establish a common key if and only if they hold credentials that belong to specific (and possibly independent) languages. The definition of the primitive is more practice-oriented than the definition of CAKE from [8] but the two notions are very similar. In particular, the new primitive enables privacy-preserving authentication and key exchange protocols by allowing two members of the same group to secretly and privately authenticate to each other without revealing this group beforehand.

In order to define the security of this primitive, we use the UC framework and an appropriate definition for languages that permits to dissociate the public part of the policy, the private common information the users want to check and the (possibly independent) secret values each user owns that assess the membership to the languages. We provide an ideal functionality for LAKE and give efficient realizations of the new primitive (for a large family of languages) secure under classical mild assumptions, in the standard model (with a common reference string – CRS), with static corruptions.

We significantly improve the efficiency of several CAKE protocols [8] for specific languages and we enlarge the set of languages for which we can construct practical schemes. Notably, we obtain a very practical realization of Secret Handshakes and a Verifier-based Password-Authenticated Key Exchange.

Our Techniques. A general framework to design PAKE in the CRS model was proposed by Gennaro and Lindell [17] in 2003. This approach was applied to the UC framework by Canetti, Halevi, Katz, Lindell, and MacKenzie [11], and improved by Abdalla, Chevalier and Pointcheval [1]. It makes use of the *smooth projective hash functions* (SPHF), introduced by Cramer and Shoup [14]. Such a hashing family is a family of hash functions that can be evaluated in two ways: using the (secret) hashing key, one can compute the function on every point in its domain, whereas using the (public) *projection* key one can only compute the function on a special subset of its domain. Our first contribution is the description of smooth projective hash functions for new interesting languages: Abdalla, Chevalier and Pointcheval [1] explained how to make disjunctions and conjunctions of languages, we study here languages defined by linear pairing product equations on committed values.

In 2011, Lindell [20] proposed a highly-efficient commitment scheme, with a non-interactive opening algorithm, in the UC framework. We will not use it in black-box, but instead we will patch it to make the initial Gennaro and Lindell's approach to work, without zero-knowledge proofs [11], using the equivocability of the commitment.

Language Definition. In [1], Abdalla *et al.* already formalized languages to be considered for SPHF. But, in the following, we will use a more simple formalism, which is nevertheless more general: we consider any efficiently computable binary relation $\mathcal{R} : \{0,1\}^* \times \mathcal{P} \times \mathcal{S} \to \{0,1\}$, where the additional parameters $\mathsf{pub} \in \{0,1\}^*$ and $\mathsf{priv} \in \mathcal{P}$ define a language $L_{\mathcal{R}}(\mathsf{pub}, \mathsf{priv}) \subseteq \mathcal{S}$ of the words W such that $\mathcal{R}(\mathsf{pub}, \mathsf{priv}, W) = 1$:

- pub are public parameters;
- priv are private parameters the two players have in mind, and they should think to the same values: they will be committed to, but never revealed;
- W is the word the sender claims to know in the language: it will be committed to, but never revealed.

Our LAKE primitive, specific to two relations $\mathcal{R}_a$ and $\mathcal{R}_b$, will allow two users, Alice and Bob, owning a word $W_a \in L_{\mathcal{R}_a}(\mathsf{pub}, \mathsf{priv}_a)$ and $W_b \in L_{\mathcal{R}_b}(\mathsf{pub}, \mathsf{priv}_b)$ respectively, to agree on a session key under some specific conditions: they first both agree on the public parameter pub, Bob will think about priv'_a for his expected value of priv_a, Alice will do the same with priv'_b for priv_b; eventually, if $\mathsf{priv}'_a = \mathsf{priv}_a$ and $\mathsf{priv}'_b = \mathsf{priv}_b$, and if they both know words in the languages, then the key agreement will succeed. In case of failure, no information should leak about the reason of failure, except the inputs did not satisfy the relations $\mathcal{R}_a$ or $\mathcal{R}_b$, or the languages were not consistent.

We stress that each LAKE protocol will be specific to a pair of relations $(\mathcal{R}_a, \mathcal{R}_b)$ describing the way Alice and Bob will authenticate to each other. This pair of relations $(\mathcal{R}_a, \mathcal{R}_b)$ specifies the sets $\mathcal{P}_a$, $\mathcal{P}_b$ and $\mathcal{S}_a$, $\mathcal{S}_b$ (to which the private parameters and the words should respectively belong). Therefore, the formats of priv_a, priv_b and W_a and W_b are known in advance, but not their values. When $\mathcal{R}_a$ and $\mathcal{R}_b$ are clearly defined from the context (e.g., PAKE), we omit them in the notations. For example, these relations can formalize:

- Password authentication: The language is defined by $\mathcal{R}(\mathsf{pub}, \mathsf{priv}, W) = 1 \Leftrightarrow W = \mathsf{priv}$, and thus $\mathsf{pub} = \emptyset$. The classical setting of PAKE requires the players A and B to use the same password W, and thus we should have $\mathsf{priv}_a = \mathsf{priv}'_b = \mathsf{priv}_b = \mathsf{priv}'_a = W_a = W_b$;
- Signature authentication: $\mathcal{R}(\mathsf{pub}, \mathsf{priv}, W) = 1 \Leftrightarrow \mathsf{Verif}(\mathsf{pub}_1, \mathsf{pub}_2, W) = 1$, where $\mathsf{pub} = (\mathsf{pub}_1 = \mathsf{vk}, \mathsf{pub}_2 = M)$ and $\mathsf{priv} = \emptyset$. The word W is thus a signature of M valid under vk, both specified in pub;
- Credential authentication: we can consider any mix for vk and M in pub or priv, and even in W, for which the relation $\mathcal{R}$ verifies the validity of the signature. When M and vk are in priv or W, we achieve *affiliation-hiding* property.

In the two last cases, the parameter pub can thus consist of a message on which the user is expected to know a signature valid under vk: either the user knows the signing key and can generate the signature on the fly to run the protocol, or the user has been given signatures on some messages (credentials). As a consequence, we just assume that, after having publicly agreed on a common pub, the two players have valid words in the appropriate languages. The way they have obtained these words does not matter.

Following our generic construction, private elements will be committed using encryption schemes, derived from Cramer-Shoup's scheme, and will thus have to be first encoded as n-tuples of elements in a group $\mathbb{G}$. In the case of PAKE, authentication will check that a player knows an appropriate password. The relation is a simple equality test, and accepts for one word only. A random commitment (and thus of a random group element) will succeed with negligible probability. For signature-based authentication, the verification key can be kept secret, but the signature should be unforgeable and thus a random word W should quite unlikely satisfy the relation. We will often make this assumption on useful relations $\mathcal{R}$: for any pub, $\{(\mathsf{priv}, W) \in \mathcal{P} \times \mathcal{S}, \mathcal{R}(\mathsf{pub}, \mathsf{priv}, W) = 1\}$ is sparse (negligible) in $\mathcal{P} \times \mathcal{S}$, and *a fortiori* in the set $\mathbb{G}^n$ in which elements are first embedded.

2 Definitions

In this section, we first briefly recall the notations and the security notions of the basic primitives we will use in the rest of the paper, and namely public key encryption and signature. More formal definitions, together with the classical computational assumptions (CDH, DDH, and DLin) are provided in the full version [6]: A public-key encryption scheme is defined by four algorithms: $\mathsf{param} \leftarrow \mathsf{Setup}(1^k)$, $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$, $c \leftarrow \mathsf{Encrypt}(\mathsf{ek}, m; r)$, and $m \leftarrow \mathsf{Decrypt}(\mathsf{dk}, c)$. We will need the classical notion of IND-CCA security. A signature scheme is defined by the four following algorithms: $\mathsf{param} \leftarrow \mathsf{Setup}(1^k)$, $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$, $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m; s)$, and $\mathsf{Verif}(\mathsf{vk}, m, \sigma)$. We will need the classical notion of EUF-CMA security. In both cases, the global parameters param will be ignored, included in the CRS. We will furthermore make use of collision-resistant hash function families.

2.1 Universal Composability

Our main goal will be to provide protocols with security in the universal composability framework. The interested reader is referred to [10, 11] for details. More precisely, we will work in the UC framework with joint state proposed by Canetti and Rabin [12] (with the CRS as the joint state). Since players are not individually authenticated, but just afterward if the credentials are mutually consistent with the two players' languages, the adversary will be allowed to interact on behalf of any player from the beginning of the protocol, either with the credentials provided by the environment (static corruption) or without (impersonation attempt). As with the Split Functionality [4], according to whom sends the first flow for a player, either the player itself or the adversary, we know whether this is an honest player or a dishonest player (corrupted or impersonation attempt, but anyway controlled by the adversary). Then, our goal will be to prove that the best an adversary can do is to try to play against one of the other players, as an honest player would do, with a credential it guessed or obtained in any possible way. This is exactly the so-called one-line dictionary attack when one considers PAKE protocols. In the adaptive corruption setting, the adversary could get complete access to the private credentials and the internal memory of an honest player, and then get control of it, at any time. But we will restrict to the static corruption setting in this paper. It is enough to deal with most of the concrete requirements: related credentials, arbitrary compositions, and forward-secrecy. To achieve our goal, for a UC-secure LAKE, we will use some other primitives which are secure in the classical setting only.

2.2 Commitment

Commitments allow a user to commit to a value, without revealing it, but without the possibility to later change his mind. It is composed of three algorithms: $\mathsf{Setup}(1^k)$ generates the system parameters, according to a security parameter k; $\mathsf{Commit}(\ell, m; r)$ produces a commitment c on the input message $m \in \mathcal{M}$ using the random coins $r \xleftarrow{\$} \mathcal{R}$, under the label ℓ, and the opening information d; while $\mathsf{Decommit}(\ell, c, m, d)$ opens the commitment c with the message m and the opening information d that proves the correct opening under the label ℓ.

Such a commitment scheme should be both *hiding*, which says that the commit phase does not leak any information about m, and *binding*, which says that the decommit phase should not be able to open to two different messages. Additional features will be required in the following, such as non-malleability, extractability, and equivocability. We also included a label ℓ, which can be empty or an additional public information that has to be the same in both the commit and the decommit phases. A labeled commitment that is both non-malleable and extractable can be instantiated by an IND-CCA labeled encryption scheme (see the full version [6]). We will use the Linear Cramer-Shoup encryption scheme [13,21]. We will then patch it, using a technique inspired from [20], to make it additionally equivocable (see Section 3). It will have an interactive commit phase, in two

rounds: $\mathsf{Commit}(\ell, m; r)$ and a challenge ε from the receiver, which will define an implicit full commitment to be open latter.

2.3 Smooth Projective Hash Functions

Smooth projective hash function (SPHF) systems have been defined by Cramer and Shoup [14] in order to build a chosen-ciphertext secure encryption scheme. They have thereafter been extended [1,7,17] and applied to several other primitives. Such a system is defined on a language L, with five algorithms:

- $\mathsf{Setup}(1^k)$ generates the system parameters, according to a security parameter k;
- $\mathsf{HashKG}(L)$ generates a hashing key hk for the language L;
- $\mathsf{ProjKG}(\mathsf{hk}, L, W)$ derives the projection key hp, possibly depending on a word W;
- $\mathsf{Hash}(\mathsf{hk}, L, W)$ outputs the hash value from the hashing key;
- $\mathsf{ProjHash}(\mathsf{hp}, L, W, w)$ outputs the hash value from the projection key and the witness w that $W \in L$.

The correctness of the scheme assures that if W is in L with w as a witness, then the two ways to compute the hash values give the same result: $\mathsf{Hash}(\mathsf{hk}, L, W) = \mathsf{ProjHash}(\mathsf{hp}, L, W, w)$. In our setting, these hash values will belong to a group $\mathbb{G}$. The security is defined through two different notions: the *smoothness* property guarantees that if $W \notin L$, the hash value is *statistically* indistinguishable from a random element, even knowing hp; the *pseudo-randomness* property guarantees that even for a word $W \in L$, but without the knowledge of a witness w, the hash value is *computationally* indistinguishable from a random element, even knowing hp.

3 Double Linear Cramer-Shoup Encryption (DLCS)

As explained earlier, any IND-CCA labeled encryption scheme can be used as a non-malleable and extractable labeled commitment scheme: we will focus on the DLin-based primitives, and thus the Linear Cramer-Shoup scheme (see the full version [6]), we call LCS. Committed/encrypted elements will either directly be group elements, or bit-strings on which we apply a reversible mapping $\mathcal{G}$ from $\{0,1\}^n$ to $\mathbb{G}$. In order to add the equivocability, one can use a technique inspired from [20]. See the full version [6] for more details, but we briefly present the commitment scheme we will use in the rest of this paper in conjunction with SPHF.

Linear Cramer-Shoup Commitment Scheme. The parameters, in the CRS, are a group $\mathbb{G}$ of prime order p, with three independent generators denoted by $(g_1, g_2, g_3) \xleftarrow{\$} \mathbb{G}^3$, a collision-resistant hash function $\mathfrak{H}_K$, and possibly an additional reversible mapping $\mathcal{G}$ from $\{0,1\}^n$ to $\mathbb{G}$ to commit bit-strings. From 9 scalars $(x_1, x_2, x_3,\ y_1, y_2, y_3,\ z_1, z_2, z_3) \xleftarrow{\$} \mathbb{Z}_p^9$, one also sets, for $i = 1, 2,$

$c_i = g_i^{x_i} g_3^{x_3}$, $d_i = g_i^{y_i} g_3^{y_3}$, and $h_i = g_i^{z_i} g_3^{z_3}$. The public parameters consist of the encryption key $\mathsf{ek} = (\mathbb{G}, g_1, g_2, g_3, c_1, c_2, d_1, d_2, h_1, h_2, \mathfrak{H}_K)$, while the trapdoor for extraction is $\mathsf{dk} = (x_1, x_2, x_3, y_1, y_2, y_3, z_1, z_2, z_3)$. One can define the encryption process:

$$\mathsf{LCS}(\ell, \mathsf{ek}, M; r, s) \stackrel{\mathsf{def}}{=} (\mathbf{u} = (g_1^r, g_2^s, g_3^{r+s}), e = M \cdot h_1^r h_2^s, v = (c_1 d_1^{\xi})^r (c_2 d_2^{\xi})^s)$$

where $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$. When ξ is specified from outside, one additionally denotes it $\mathsf{LCS}^*(\ell, \mathsf{ek}, M, \xi; r, s)$. The commitment to a message $M \in \mathbb{G}$, or $M = \mathcal{G}(m)$ for $m \in \{0,1\}^n$, encrypts M under ek: $\mathsf{LCSCom}(\ell, M; r, s) \stackrel{\mathsf{def}}{=} \mathsf{LCS}(\ell, \mathsf{ek}, M; r, s)$. The decommit process consists of M and (r, s) to check the correctness of the encryption. It is possible to do implicit verification, without any decommit information, but just an SPHF on the language of the ciphertexts of M that is privately shared by the two players. Since the underlying encryption scheme is IND-CCA, this commitment scheme is non-malleable and extractable.

Double Linear Cramer-Shoup Commitment Schemes. To make it equivocable, we double the commitment process, in two steps. The CRS additionally contains a scalar $\aleph \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, one also sets, $\zeta = g_1^{\aleph}$. The trapdoor for equivocability is $\aleph$. The Double Linear Cramer-Shoup encryption scheme, denoted DLCS and detailed in the full version [6] is

$$\mathsf{DLCS}(\ell, \mathsf{ek}, M, N; r, s, a, b) \stackrel{\mathsf{def}}{=} (\mathcal{C} \leftarrow \mathsf{LCS}(\ell, \mathsf{ek}, M; r, s), \mathcal{C}' \leftarrow \mathsf{LCS}^*(\ell, \mathsf{ek}, N, \xi; a, b))$$

where $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$ is computed during the generation of $\mathcal{C}$ and transfered for the generation of $\mathcal{C}'$. As above, we denote $\mathsf{DLCSCom}$ denotes the use of DLCS with the encryption key ek. The usual commit/decommit processes are described in the full version [6]. On Figure 1, one can find the $\mathsf{DLCSCom}'$ scheme where one can implicitly check the opening with an SPHF. These two constructions essentially differ with $\chi = \mathfrak{H}_K(\mathcal{C}')$ (for the SPHF implicit check) instead of $\chi = \mathfrak{H}_K(M, \mathcal{C}')$ (for the explicit check). We stress that with this alteration, the $\mathsf{DLCSCom}'$ scheme is not a real commitment scheme (not formally extractable/binding): in $\mathsf{DLCSCom}'$, the sender can indeed encrypt M in $\mathcal{C}$ and $N \neq 1_{\mathbb{G}}$ in $\mathcal{C}'$, and then, the global ciphertext $\mathcal{C} \times \mathcal{C}'^{\varepsilon}$ contains $M' = MN^{\varepsilon} \neq M$, whereas one would have extracted M from $\mathcal{C}$. But M' is unknown before ε is sent, and thus, if one checks the membership of M' to a sparse language, it will unlikely be true.

Multi-message Schemes. One can extend these encryption and commitment schemes to vectors of n messages (see the full version [6]). We will denote them n-$\mathsf{DLCSCom}'$ or n-$\mathsf{DLCSCom}$ for the commitment schemes. They consist in encrypting each message with independent random coins in $\mathcal{C}_i = (\mathbf{u}_i, e_i, v_i)$ but the same $\xi = \mathfrak{H}_K(\ell, (\mathbf{u}_i), (e_i))$, together with independent companion ciphertexts $\mathcal{C}'_i$ of $1_{\mathbb{G}}$, still with the same ξ for the doubled version. In the latter case, n independent challenges $\varepsilon_i \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$ are then sent to lead to the full commitment $(\mathcal{C}_i \times \mathcal{C}_i'^{\varepsilon_i})$ with random coins $z_{r_i} = r_i + \varepsilon_i a_i$ and $z_{s_i} = s_i + \varepsilon_i b_i$. Again, if one of the companion ciphertext $\mathcal{C}'_i$ does not encrypt $1_{\mathbb{G}}$, the full commitment encrypts a vector with at least one unpredictable component M'_i. Several non-unity components

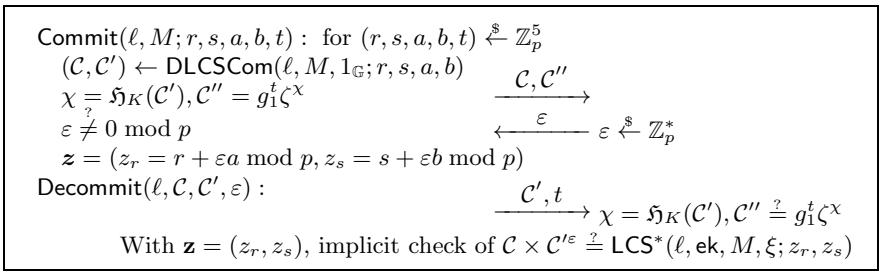


Fig. 1. DLCSCom′ Commitment Scheme for SPHF

in the companion ciphertexts would lead to independent components in the full commitment. For languages sparse enough, this definitely turns out not to be in the language.

4 SPHF for Implicit Proofs of Membership

In [1], Abdalla *et al.* presented a way to compute a conjunction or a disjunction of languages by some simple operations on their projection keys. Therefore all languages presented afterward can easily be combined together. However as the original set of manageable languages was not really developed, we are going to present several steps to extend it, and namely in order to cover some languages useful in various AKE instantiations.

We will show that almost all the vast family of languages covered by the Groth-Sahai methodology [18] can be addressed by our approach too. More precisely, we can handle all the linear pairing product equations, when witnesses are committed using our above (multi-message) DLCSCom′ commitment scheme, or even the non-equivocable LCSCom version. This will be strong enough for our applications. For using them in black-box to build our LAKE protocol, one should note that the projection key is computed from the ciphertext $\mathcal{C}$ when using the simple LCSCom commitment, but also when using the DLCSCom′ version. The full commitment $\mathcal{C} \times \mathcal{C}'^\varepsilon$ is not required, but ξ only, which is known as soon as $\mathcal{C}$ is given (or the vector $(\mathcal{C}_i)_i$ for the multi-message version). Of course, the hash value will then depend on the full commitment (either $\mathcal{C}$ for the LCSCom commitment, or $\mathcal{C} \cdot \mathcal{C}'^\varepsilon$ for the DLCSCom′ commitment).

This will be relevant to our AKE problem: equality of two passwords, in PAKE protocols; corresponding signing/verification keys associated with a valid signature on a pseudonym or a hidden identity, in secret handshakes; valid credentials, in CAKE protocols. All those tests are quite similar: one has to show that the ciphertexts are valid and that the plaintexts satisfy the expected relations in a group. We first illustrate that with commitments of Waters signatures of a public message under a committed verification key. We then explain the general method. The formal proofs are provided in the full version [6].

4.1 Commitments of Signatures

Let us consider the Waters signature [22] in a symmetric bilinear group, and then we just need to recall that, in a pairing-friendly setting $(p, \mathbb{G}, \mathbb{G}_T, e)$, with public parameters $(\mathcal{F}, g, h)$, and a verification key vk, a signature $\sigma = (\sigma_1, \sigma_2)$ is valid with respect to the message M under the key vk if it satisfies $e(\sigma_1, g) = e(h, \mathsf{vk}) \cdot e(\mathcal{F}(M), \sigma_2)$.

A similar approach has already been followed in [7], however not with a Linear Cramer-Shoup commitment scheme, nor with such general languages. We indeed first consider the language of the signatures $(\sigma_1, \sigma_2) \in \mathbb{G}^2$ of a message $M \in \{0,1\}^k$ under the verification key $\mathsf{vk} \in \mathbb{G}$, where M is public but vk is private: $L(\mathsf{pub}, \mathsf{priv})$, where $\mathsf{priv} = \mathsf{vk}$ and $\mathsf{pub} = M$. One will thus commit the pair $(\mathsf{vk}, \sigma_1) \in \mathbb{G}^2$ with the label $\ell = (M, \sigma_2)$ using a 2-$\mathsf{DLCSCom}'$ commitment and then prove the commitment actually contains (vk, σ_1) such that $e(\sigma_1, g) = e(h, \mathsf{vk}) \cdot e(\mathcal{F}(M), \sigma_2)$. We insist on the fact that σ_1 only has to be encrypted, and not σ_2, in order to hide the signature, since the latter σ_2 is a random group element. If one wants unlinkability between signature commitments, one simply needs to re-randomize (σ_1, σ_2) before encryption. Hence σ_2 can be sent in clear, but bounded to the commitment in the label, together with the pub part of the language. In order to prove the above property on the committed values, we will use conjunctions of SPHF: first, to show that each commitment is well-formed (valid ciphertexts), and then that the associated plaintexts verify the linear pairing equation, where the committed values are underlined: $e(\underline{\sigma_1}, g) = e(h, \mathsf{vk}) \cdot e(\mathcal{F}(M), \sigma_2)$ Note that vk is not used as a committed value for this verification of the membership of σ to the language since this is the verification key expected by the verifier, specified in the private part priv, which has to be independently checked with respect to the committed verification key. This is enough for the affiliation-hiding property. We could consider the similar language where $M \in \{0,1\}^k$ is in the word too: $e(\underline{\sigma_1}, g) = e(h, \mathsf{vk}) \cdot e(\underline{\mathcal{F}(M)}, \sigma_2)$, and then one should commit M, bit-by-bit, and then use a $(k+2)$-$\mathsf{DLCSCom}'$ commitment.

4.2 Linear Pairing Product Equations

Instead of describing in details the SPHF for the above examples, let us show it for a more general framework: we considered

$$e(\underline{\sigma_1}, g) = e(h, \mathsf{vk}) \cdot e(\mathcal{F}(M), \sigma_2) \text{ or } e(\underline{\sigma_1}, g) = e(h, \mathsf{vk}) \cdot e(\underline{\mathcal{F}(M)}, \sigma_2),$$

where the unknowns are underlined. These are particular instantiations of t simultaneous equations

$$\left(\prod_{i \in A_k} e(\underline{\mathcal{Y}_i}, \mathcal{A}_{k,i}) \right) \cdot \left(\prod_{i \in B_k} \underline{\mathcal{Z}_i}^{\mathfrak{z}_{k,i}} \right) = \mathcal{B}_k, \text{ for } k = 1, \ldots, t,$$

where $\mathcal{A}_{k,i} \in \mathbb{G}$, $\mathcal{B}_k \in \mathbb{G}_T$, and $\mathfrak{z}_{k,i} \in \mathbb{Z}_p$, as well as $A_k \subseteq \{1, \ldots, m\}$ and $B_k \subseteq \{m+1, \ldots, n\}$ are public, but the $\mathcal{Y}_i \in \mathbb{G}$ and $\mathcal{Z}_i \in \mathbb{G}_T$ are simultaneously committed using the multi-message $\mathsf{DLCSCom}'$ or LCSCom commitments

scheme, in $\mathbb{G}$ or $\mathbb{G}_T$ respectively. This is more general than the relations covered by [8], since one can also commit scalars bit-by-bit. In the full version [6], we detail how to build the corresponding SPHF, and prove the soundness of our approach. For the sake of clarity, we focus here to a single equation only, since multiple equations are just conjunctions. We can even consider the simpler equation $\prod_{i=1}^{i=m} \underline{\mathcal{Z}_i}^{\mathfrak{z}_i} = \mathcal{B}$, since one can lift any ciphertext from $\mathbb{G}$ to a ciphertext in $\mathbb{G}_T$, setting $\mathcal{Z}_i = e(\mathcal{Y}_i, \mathcal{A}_i)$, as well as, for $j = 1, 2, 3$, $G_{i,j} = e(g_j, \mathcal{A}_i)$ and for $j = 1, 2$, $H_{i,j} = e(h_j, \mathcal{A}_i)$, $C_{i,j} = e(c_j, \mathcal{A}_i)$, $D_{i,j} = e(d_j, \mathcal{A}_i)$, to lift all the group basis elements. Then, one transforms $\mathcal{C}_i = \mathsf{LCS}^*(\ell, \mathsf{ek}, \mathcal{Y}_i, \xi; \mathbf{z}_i) = (\boldsymbol{u_i} = (g_1^{z_{r_i}}, g_2^{z_{s_i}}, g_3^{z_{r_i}+z_{s_i}}), e_i = h_1^{z_{r_i}} h_2^{z_{s_i}} \cdot \mathcal{Y}_i, v_i = (c_1 d_1^{\xi})^{z_{r_i}} \cdot (c_2 d_2^{\xi})^{z_{s_i}})$ into $(\boldsymbol{U_i} = (G_{i,1}^{z_{r_i}}, G_{i,2}^{z_{s_i}}, G_{i,3}^{z_{r_i}+z_{s_i}}), E_i = H_{i,1}^{z_{r_i}} H_{i,2}^{z_{s_i}} \cdot \mathcal{Z}_i, V_i = (C_{i,1} D_{i,1}^{\xi})^{z_{r_i}} \cdot (C_{i,2} D_{i,2}^{\xi})^{z_{s_i}})$. Encryptions of $\mathcal{Z}_i$ originally in $\mathbb{G}_T$ use constant basis elements for $j = 1, 2, 3$, $G_{i,j} = G_j = e(g_j, g)$ and for $j = 1, 2$, $H_{i,j} = H_j = e(h_j, g)$, $C_{i,j} = C_j = e(c_j, g)$, $D_{i,j} = D_j = e(d_j, g)$.

The commitments have been generated in $\mathbb{G}$ and $\mathbb{G}_T$ simultaneously using the m-$\mathsf{DLCSCom}'$ version, with a common ξ, where the possible combination with the companion ciphertext to the power ε leads to the above $\mathcal{C}_i$, thereafter lifted to $\mathbb{G}_T$. For the hashing keys, one picks random scalars $(\lambda, (\eta_i, \theta_i, \kappa_i, \mu_i)_{i=1,\ldots,m}) \xleftarrow{\$} \mathbb{Z}_p^{4m+1}$, and sets $\mathsf{hk}_i = (\eta_i, \theta_i, \kappa_i, \lambda, \mu_i)$. One then computes the projection keys as $\mathsf{hp}_i = (g_1^{\eta_i} g_3^{\kappa_i} h_1^{\lambda} (c_1 d_1^{\xi})^{\mu_i}, g_2^{\theta_i} g_3^{\kappa_i} h_2^{\lambda} (c_2 d_2^{\xi})^{\mu_i}) \in \mathbb{G}^2$. The hash value is

$$\prod_i e(u_{i,1}^{\eta_i} \cdot u_{i,2}^{\theta_i} \cdot u_{i,3}^{\kappa_i} \cdot e_i^{\lambda} \cdot v_i^{\mu_i}, \mathcal{A}_i) \times \mathcal{B}^{-\lambda} = \prod_i e(\mathsf{hp}_{i,1}^{z_{r_i}} \mathsf{hp}_{i,2}^{z_{s_i}}, \mathcal{A}_i),$$

where $\mathcal{A}_i$ is the constant used to compute $\mathcal{Z}_i = e(\mathcal{Y}_i, \mathcal{A}_i)$ and to lift ciphertexts from $\mathbb{G}$ to $\mathbb{G}_T$, or $\mathcal{A}_i = g^{\mathfrak{z}_i}$ if the ciphertext was already in $\mathbb{G}_T$. These evaluations can be computed either from the commitments and the hashing keys, or from the projection keys and the witnesses. We insist on the fact that, whereas the hash values are in $\mathbb{G}_T$, the projection keys are in $\mathbb{G}$ even if the ciphertexts are initially in $\mathbb{G}_T$. We stress again that the projection keys require the knowledge of ξ only: known from the LCSCom commitment or the first part $\mathcal{C}$ of the $\mathsf{DLCSCom}'$ commitment.

5 Language-Authenticated Key Exchange

5.1 The Ideal Functionality

We generalize the Password-Authenticated Key Exchange functionality $\mathcal{F}_{\text{PAKE}}$ (first provided in [11]) to more complex languages: the players agree on a common secret key if and only if they own words that lie in the languages the partners have in mind. More precisely, after an agreement on pub between P_i and P_j (modeled here by the use of the split functionality, see below), player P_i uses a word W_i belonging to $L_i = L_{\mathcal{R}_i}(\mathsf{pub}, \mathsf{priv}_i)$ and it expects its partner P_j to use a word W_j belonging to the language $L'_j = L_{\mathcal{R}_j}(\mathsf{pub}, \mathsf{priv}'_j)$, and vice-versa for P_j and P_i. We assume relations $\mathcal{R}_i$ and $\mathcal{R}_j$ to be specified by the kind of protocol we study (PAKE, Verifier-based PAKE, secret handshakes, ...) and

The functionality $\mathcal{F}_{\text{LAKE}}$ is parametrized by a security parameter k and a public parameter pub for the languages. It interacts with an adversary $\mathcal{S}$ and a set of parties $P_1,\ldots,P_n$ via the following queries:

- New Session: Upon receiving a query $(\mathsf{NewSession} : \mathsf{sid}, P_i, P_j, W_i, L_i = L(\mathsf{pub}, \mathsf{priv}_i), L'_j = L(\mathsf{pub}, \mathsf{priv}'_j))$ from P_i,
  - If this is the first $\mathsf{NewSession}$-query with identifier sid, record the tuple $(P_i, P_j, W_i, L_i, L'_j, \mathsf{initiator})$. Send $(\mathsf{NewSession}; \mathsf{sid}, P_i, P_j, \mathsf{pub}, \mathsf{initiator})$ to $\mathcal{S}$ and P_j.
  - If this is the second $\mathsf{NewSession}$-query with identifier sid and if there is a record $(P_j, P_i, W_j, L_j, L'_i, \mathsf{initiator})$, then record the tuple $(P_j, P_i, W_j, L_j, L'_i, \mathsf{initiator}, W_i, L_i, L'_j, \mathsf{receiver})$ and send the answer $(\mathsf{NewSession}; \mathsf{sid}, P_i, P_j, \mathsf{pub}, \mathsf{receiver})$ to $\mathcal{S}$ and P_j.
- Key Computation: Upon receiving a query $(\mathsf{NewKey} : \mathsf{sid})$ from $\mathcal{S}$, if there is a record of the form $(P_i, P_j, W_i, L_i, L'_j, \mathsf{initiator}, W_j, L_j, L'_i, \mathsf{receiver})$ and this is the first NewKey-query for session sid, then
  - If $(L'_i = L_i$ and $W_i \in L_i)$ and $(L'_j = L_j$ and $W_j \in L_j)$, then pick a random key sk of length k and store $(\mathsf{sid}, \mathsf{sk})$. If one player is corrupted, send $(\mathsf{sid}, \mathsf{success})$ to the adversary.
  - Else, store $(\mathsf{sid}, \perp)$, and send $(\mathsf{sid}, \mathsf{fail})$ to the adversary if one player is corrupted.
- Key Delivery: Upon receiving a query $(\mathsf{SendKey} : \mathsf{sid}, P_i, \mathsf{sk})$ from $\mathcal{S}$, then
  - if there is a record of the form $(\mathsf{sid}, \mathsf{sk}')$, then, if both players are uncorrupted, output $(\mathsf{sid}, \mathsf{sk}')$ to P_i. Otherwise, output $(\mathsf{sid}, \mathsf{sk})$ to P_i.
  - if there is a record of the form $(\mathsf{sid}, \perp)$, then pick a random key sk' of length k and output $(\mathsf{sid}, \mathsf{sk}')$ to P_i.

Fig. 2. Ideal Functionality $\mathcal{F}_{\text{LAKE}}$

so the languages are defined by the additional parameters pub, priv_i and priv_j only: they both agree on the public part pub, to be possibly parsed in a different way by each player for each language according to the relations. Note however that the respective languages do not need to be the same or to use similar relations: authentication means could be totally different for the 2 players. The key exchange should succeed if and only if the two following pairs of equations hold: $(L'_i = L_i$ and $W_i \in L_i)$ and $(L'_j = L_j$ and $W_j \in L_j)$.

Description. In the initial $\mathcal{F}_{\text{PAKE}}$ functionality [11], the adversary was given access to a $\mathsf{TestPwd}$-query, which modeled the on-line dictionary attack. But it is known since [4] that it is equivalent to use the split functionality model [4], generate the $\mathsf{NewSession}$-queries corresponding to the corrupted players and tell the adversary (on behalf of the corrupted player) whether the protocol should succeed or not. Both methods enable the adversary to try a credential for a player (on-line dictionary attack). The second method (that we use here) implies allowing $\mathcal{S}$ to ask $\mathsf{NewSession}$-queries on behalf of the corrupted player, and letting it to be aware of the success or failure of the protocol in this case: the adversary learns this information only when it plays on behalf of a player

Given the functionality $\mathcal{F}_{\text{LAKE}}$, the split functionality $s\mathcal{F}_{\text{LAKE}}$ proceeds as follows:

- Initialization:
  - Upon receiving $(\mathsf{Init}, \mathsf{sid}, \mathsf{pub}_i)$ from party P_i, send $(\mathsf{Init}, \mathsf{sid}, P_i, \mathsf{pub}_i)$ to the adversary.
  - Upon receiving a message $(\mathsf{Init}, \mathsf{sid}, P_i, H, \mathsf{pub}, \mathsf{sid}_H)$ from $\mathcal{S}$, where $H = \{P_i, P_j\}$ is a set of party identities, check that P_i has already sent $(\mathsf{Init}, \mathsf{sid}, \mathsf{pub}_i)$ and that for all recorded $(H', \mathsf{pub}', \mathsf{sid}_{H'})$, either $H = H'$, $\mathsf{pub} = \mathsf{pub}'$ and $\mathsf{sid}_H = \mathsf{sid}_{H'}$ or H and H' are disjoint and $\mathsf{sid}_H \neq \mathsf{sid}_{H'}$. If so, record the pair $(H, \mathsf{pub}, \mathsf{sid}_H)$, send $(\mathsf{Init}, \mathsf{sid}, \mathsf{sid}_H, \mathsf{pub})$ to P_i, and invoke a new functionality $(\mathcal{F}_{\text{LAKE}}, \mathsf{sid}_H, \mathsf{pub})$ denoted as $\mathcal{F}_{\text{LAKE}}^{(H,\mathsf{pub})}$ and with set of honest parties H.
- Computation:
  - Upon receiving $(\mathsf{Input}, \mathsf{sid}, m)$ from party P_i, find the set H such that $P_i \in H$, the public value pub recorded, and forward m to $\mathcal{F}_{\text{LAKE}}^{(H,\mathsf{pub})}$.
  - Upon receiving $(\mathsf{Input}, \mathsf{sid}, P_j, H, m)$ from $\mathcal{S}$, such that $P_j \notin H$, forward m to $\mathcal{F}_{\text{LAKE}}^{(H,\mathsf{pub})}$ as if coming from P_j.
  - When $\mathcal{F}_{\text{LAKE}}^{(H,\mathsf{pub})}$ generates an output m for party $P_i \in H$, send m to P_i. If the output is for $P_j \notin H$ or for the adversary, send m to the adversary.

Fig. 3. Split Functionality $s\mathcal{F}_{\text{LAKE}}$

(corruption or impersonation attempt). This is any way an information it would learn at the end of the protocol. We insist that third parties will not learn whether the protocol succeeded or not, as required for secret handshakes. To this aim, the NewKey-query informs in this case the adversary whether the credentials are consistent with the languages or not. In addition, the split functionality model guarantees from the beginning which player is honest and which one is controlled by the adversary. This finally allows us to get rid of the TestPwd-query. The $\mathcal{F}_{\text{LAKE}}$ functionality is presented in Figure 2 and the corresponding split functionality $s\mathcal{F}_{\text{LAKE}}$ in Figure 3, where the languages are formally described and compared using the pub and priv parts.

The security goal is to show that the best attack for the adversary is a basic trial execution with a credential of its guess or choice: the proof will thus consist in emulating any real-life attack by either a trial execution by the adversary, playing as an honest player would do, but with a credential chosen by the adversary or obtained in any way; or a denial of service, where the adversary is clearly aware that its behavior will make the execution fail.

5.2 A Generic UC-Secure LAKE Construction

Intuition. Using smooth projective hash functions on commitments, one can generically define a LAKE protocol as done in [1]. The basic idea is to make the player commit to their private information (for the expected languages and the owned words), and eventually the smooth projective hash functions will be used to make implicit validity checks of the global relation.

To this aim, we use the commitments and associated smooth projective hash functions as described in Sections 3 and 4. More precisely, all examples of SPHF in Section 4 can be used on extractable commitments divided into one or two parts (the non-equivocable LCSCom or the equivocable $\mathsf{DLCSCom}'$ commitments, see Figure 1). The relations on the committed values will not be explicitly checked, since the values will never be revealed, but will be implicitly checked using SPHF. It is interesting to note that in both cases (one-part or two-part commitment), the projection key will only depend on the first part of the commitment.

As it is often the case in the UC setting, we need the initiator to use stronger primitives than the receiver. They both have to use non-malleable and extractable commitments, but the initiator will use a commitment that is additionally equivocable, the $\mathsf{DLCSCom}'$ in two parts ($(\mathcal{C}_i, \mathcal{C}'_i)$ and $\mathsf{Com}_i = \mathcal{C}_i \cdot \mathcal{C}'^{\boldsymbol{\epsilon}}_i$), while the receiver will only need the basic LCSCom commitment in one part ($\mathsf{Com}_j = \mathcal{C}_j$).

As already explained, SPHF will be used to implicitly check whether ($L'_i = L_i$ and $W_i \in L_i$) and ($L'_j = L_j$ and $W_j \in L_j$). But since in our instantiations private parameters priv and words W will have to be committed, the structure of these commitments will thus be publicly known in advance: commitments of $\mathcal{P}$-elements and $\mathcal{S}$-elements. Section 6 discusses on the languages captured by our definition, and illustrates with some AKE protocols. However, while these $\mathcal{P}$ and $\mathcal{S}$ sets are embedded in $\mathbb{G}^n$ from some n, it might be important to prove that the committed values are actually in $\mathcal{P}$ and $\mathcal{S}$ (e.g., one can have to prove it commits bits, whereas messages are first embedded as group elements in $\mathbb{G}$ of large order p). This will be an additional language-membership to prove on the commitments.

This leads to a very simple protocol described on Figure 4. Note that if a player wants to make external adversaries think he owns an appropriate word, as it is required for Secret Handshakes, he can still play, but will compute everything with dummy words, and will replace the $\mathsf{ProjHash}$ evaluation by a random value, which will lead to a random key at the end.

Security Analysis. Since we have to assume common pub, we make a first round (with flows in each direction) where the players send their contribution, to come up with pub. These flows will also be used to know if there is a player controlled by the adversary (as with the Split Functionality [4]). In case the languages have empty pub, these additional flows are not required, since the Split Functionality can be applied on the committed values. The signing key for the receiver is not required anymore since there is one flow only from its side. This LAKE protocol is secure against static corruptions. The proof is provided in the full version [6], and is in the same vein as the one in [1,11]. However, it is a bit more intricate:

– in PAKE, when one is simulating a player, and knows the adversary used the correct password, one simply uses this password for the simulated player. In LAKE, when one knows the language expected by the adversary for the

Execution between P_i and P_j, with session identifier sid.

- Preliminary Round: each user generates a pair of signing/verification keys $(\mathsf{SK}, \mathsf{VK})$ and sends VK together with its contribution to the public part of the language.

We denote by ℓ_i the label $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, \mathsf{pub}, \mathsf{VK}_i, \mathsf{VK}_j)$ and by ℓ_j the label $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, \mathsf{pub}, \mathsf{VK}_j, \mathsf{VK}_i)$, where pub is the combination of the contributions of the two players. The initiator now uses a word W_i in the language $L(\mathsf{pub}, \mathsf{priv}_i)$, and the receiver uses a word W_j in the language $L(\mathsf{pub}, \mathsf{priv}_j)$, possibly re-randomized from their long-term secrets (*). We assume commitments and associated smooth projective hash functions exist for these languages.

- First Round: user P_i (with random tape ω_i) generates a multi-$\mathsf{DLCSCom}'$ commitment on $(\mathsf{priv}_i, \mathsf{priv}'_j, W_i)$ in $(\mathcal{C}_i, \mathcal{C}'_i)$, where W_i has been randomized in the language, under the label ℓ_i. It also computes a Pedersen commitment on $\mathcal{C}'_i$ in $\mathcal{C}''_i$ (with random exponent t). It then sends $(\mathcal{C}_i, \mathcal{C}''_i)$ to P_j;
- Second Round: user P_j (with random tape ω_j) computes a multi-LCS commitment on $(\mathsf{priv}_j, \mathsf{priv}'_i, W_j)$ in $\mathsf{Com}_j = \mathcal{C}_j$, with witness $\boldsymbol{r}$, where W_j has been randomized in the language, under the label ℓ_j. It then generates a challenge $\boldsymbol{\varepsilon}$ on $\mathcal{C}_i$ and hashing/projection keys (**) hk_i and hp_i associated to $\mathcal{C}_i$ (which will be associated to the future Com_i). It finally signs all the flows using SK_j in σ_j, and sends $(\mathcal{C}_j, \boldsymbol{\varepsilon}, \mathsf{hp}_i, \sigma_j)$ to P_i;
- Third Round: user P_i first checks the signature σ_j, computes $\mathsf{Com}_i = \mathcal{C}_i \times \mathcal{C}_i'^{\boldsymbol{\varepsilon}}$ and witness $\mathbf{z}$ (from $\boldsymbol{\varepsilon}$ and ω_i), it generates hashing/projection keys hk_j and hp_j associated to Com_j. It finally signs all the flows using SK_i in σ_i, and sends $(\mathcal{C}'_i, t, \mathsf{hp}_j, \sigma_i)$ to P_j;
- Hashing: P_j first checks the signature σ_i and the correct opening of $\mathcal{C}''_i$ into $\mathcal{C}'_i$, it computes $\mathsf{Com}_i = \mathcal{C}_i \times \mathcal{C}_i'^{\boldsymbol{\varepsilon}}$.
  P_i computes K_i and P_j computes K_j as follows:

$$\begin{aligned} K_i &= \mathsf{Hash}(\mathsf{hk}_j, \{(\mathsf{priv}'_j, \mathsf{priv}_i)\} \times L(\mathsf{pub}, \mathsf{priv}'_j), \ell_j, \mathsf{Com}_j) \\ &\quad \times \mathsf{ProjHash}(\mathsf{hp}_i, \{(\mathsf{priv}_i, \mathsf{priv}'_j)\} \times L(\mathsf{pub}, \mathsf{priv}_i), \ell_i, \mathsf{Com}_i; \mathbf{z}) \\ K_j &= \mathsf{ProjHash}(\mathsf{hp}_j, \{(\mathsf{priv}_j, \mathsf{priv}'_i)\} \times L(\mathsf{pub}, \mathsf{priv}_j), \ell_j, \mathsf{Com}_j; \boldsymbol{r}) \\ &\quad \times \mathsf{Hash}(\mathsf{hk}_i, \{(\mathsf{priv}'_i, \mathsf{priv}_j)\} \times L(\mathsf{pub}, \mathsf{priv}'_i), \ell_i, \mathsf{Com}_i) \end{aligned}$$

(*) As explained in Section 1, recall that the languages considered depend on two possibly different relations, namely $L_i = L_{\mathcal{R}_i}(\mathsf{pub}, \mathsf{priv}_i)$ and $L_j = L_{\mathcal{R}_j}(\mathsf{pub}, \mathsf{priv}_j)$, but we omit them for the sake of clarity. We assume they are both self-randomizable.
(**) Recall that the SPHF is constructed in such a way that this projection key does not depend on $\mathcal{C}'_i$ and is indeed associated to the future whole Com_i.

Fig. 4. Language-based Authenticated Key Exchange from a Smooth Projective Hash Function on Commitments

simulated player and has to simulate a successful execution (because of success announced by the NewKey-query), one has to actually include a correct word in the commitment: smooth projective hash functions do not allow the simulator to cheat, equivocability of the commitment is the unique trapdoor, but with a valid word. The languages must allow the simulator to produce a valid word W in $L(\mathsf{pub}, \mathsf{priv})$, for any pub and $\mathsf{priv} \in \mathcal{P}$ provided by the adversary or the environment. This will be the case in all the interesting applications of our protocol (see Section 6): if priv defines a Waters' verification key $\mathsf{vk} = g^x$, with the master key s such that $h = g^s$, the signing key is $\mathsf{sk} = h^x = \mathsf{vk}^s$, and thus the simulator can sign any message; if such a master key does not exist, one can restrict $\mathcal{P}$, and implicitly check it with the SPHF (the additional language-membership check, as said above). But since a random word is generated by the simulator, we need the real player to derive a random word from his own word, and the language to be *self-randomizable*.
- In addition, as already noted, our commitment $\mathsf{DLCSCom}'$ is not formally binding (contrarily to the much less efficient one used in [1]). The adversary can indeed make the extraction give $\boldsymbol{M}$ from $\mathcal{C}_i$, whereas Com_i will eventually contain $\boldsymbol{M}'$ if $\mathcal{C}'_i$ does not encrypt $(1_{\mathbb{G}})^n$. However, since the actual value $\boldsymbol{M}'$ depends on the random challenge $\boldsymbol{\varepsilon}$, and the language is assumed sparse (otherwise authentication is easy), the protocol will fail: this can be seen as a denial of service from the adversary.

Theorem 1. *Our LAKE scheme from Figure 4 realizes the* $s\mathcal{F}_{\text{LAKE}}$ *functionality in the* $\mathcal{F}_{\text{CRS}}$*-hybrid model, in the presence of static adversaries, under the* DLin *assumption and the security of the One-Time Signature.*

Actually, from a closer look at the full proof, one can notice that $\mathsf{Com}_j = \mathcal{C}_j$ needs to be extractable, but $\mathsf{IND} - \mathsf{CPA}$ security is enough, which leads to a shorter ciphertext (2 group elements less if one uses a Linear ciphertext instead of LCS). Similarly, one will not have to extract W_i from $\mathcal{C}_i$ when simulating sessions where P_i is corrupted. As a consequence, only the private parts of the languages have to be committed to in Com_i in the first and third rounds, whereas W_i can be encrypted independently with an $\mathsf{IND} - \mathsf{CPA}$ encryption scheme in the third round only (5 group elements less in the first round, and 2 group elements less in the third round if one uses a Linear ciphertext instead of LCS).

6 Concrete Instantiations and Comparisons

In this section, we first give some concrete instantiations of several AKE protocols, using our generic protocol of LAKE, and compare the efficiencies of those instantiations.

6.1 Possible Languages

As explained above, our LAKE protocol is provably secure for *self-randomizable* languages only. While this notion may seem quite strong, most of the usual languages fall into it. For example, in a PAKE or a Verifier-based PAKE scheme,

the languages consist of a single word and so trivially given a word, each user is able to deduce all the words in the language. One may be a little more worried about Waters Signature in our Secret Handshake, and/or Linear pairing equations. However the *self-randomizability* of the languages is easy to show:

- Given a Waters signature $\sigma = (\sigma_1, \sigma_2)$ over a message m valid under a verification key vk, one is able to randomize the signature into any signature over the same message m valid under the same verification key vk simply by picking a random s and computing $\sigma' = (\sigma_1 \cdot \mathcal{F}(m)^s, \sigma_2 \cdot g^s)$.
- For linear pairing equations, with public parameters $\mathcal{A}_i$ for $i = 1, \ldots, m$ and γ_i for $i = m+1, \ldots, n$, and $\mathcal{B}$, given $(\mathcal{X}_1, \ldots, \mathcal{X}_m, \mathcal{Z}_{m+1}, \ldots, \mathcal{Z}_n)$ verifying $\prod_{i=1}^{m} e(\mathcal{X}_i, \mathcal{A}_i) \cdot \prod_{i=m+1}^{n} \mathcal{Z}_i^{\gamma_i} = \mathcal{B}$, one can randomize the word in the following way:
  - If $m < n$, one simply picks random $(\mathcal{X}'_1, \ldots, \mathcal{X}'_m)$, $(\mathcal{Z}'_{m+1}, \ldots, \mathcal{Z}'_{n-1})$ and sets $\mathcal{Z}'_n = (\mathcal{B}/(\prod_{i=1}^{m} e(\mathcal{X}'_i, \mathcal{A}_i) \cdot \prod_{i=m+1}^{n-1} \mathcal{Z}_i'^{\gamma_i}))^{1/\gamma_n}$,
  - Else, if $m = n > 1$, one picks random $r_1, \ldots, r_{n-1}$ and set $\mathcal{X}'_i = \mathcal{X}_i \cdot \mathcal{A}_n^{r_i}$, for $i = 1, \ldots, m-1$ and $\mathcal{X}'_m = \mathcal{X}_m \cdot \prod_{i=1}^{m-1} \mathcal{A}_i^{-r_i}$,
  - Else $m = n = 1$, this means only one word satisfies the equation. So we already have this word.

As we can see most of the common languages manageable with a SPHF are already *self-randomizable*. We now show how to use them in concrete instantiations.

6.2 Concrete Instantiations

Password-Authenticated Key Exchange. Using our generic construction, we can easily obtain a PAKE protocol, as described on Figure 5, where we optimize from the generic construction, since $\mathsf{pub} = \emptyset$, removing the agreement on pub, but still keeping the one-time signature keys $(\mathsf{SK}_i, \mathsf{VK}_i)$ to avoid man-in-the-middle attacks since it has another later flow: P_i uses a password W_i and expects P_j to own the same word, and thus in the language $L'_j = L_i = \{W_i\}$; P_j uses a password W_j and expects P_i to own the same word, and thus in the language $L'_i = L_j = \{W_j\}$; The relation is the equality test between priv_i and priv_j, which both have no restriction in $\mathbb{G}$ (hence $\mathcal{P} = \mathbb{G}$). As the word W_i, the language private parameters priv_i of a user and priv'_j of the expected language for the other user are the same, each user can commit in the protocol to only one value: its password.

We kept the general description and notations in Figure 5, but $\mathcal{C}_j$ can be a simply $\mathsf{IND} - \mathsf{CPA}$ encryption scheme. It is quite efficient and relies on the DLin assumption, with DLCS for $(\mathcal{C}_i, \mathcal{C}'_i)$ and thus 10 group elements, but a Linear encryption for $\mathcal{C}_j$ and thus 3 group elements. Projection keys are both 2 group elements. Globally, P_i sends 13 groups elements plus 1 scalar, a verification key and a one-time signature, while P_j sends 5 group elements and 1 scalar: 18 group elements and 2 scalars in total. We can of course instantiate it with the Cramer-Shoup and ElGamal variants, under the DDH assumption: P_i sends 8

P_i uses a password W_i and P_j uses a password W_j. We denote $\ell = (\mathsf{sid}, \mathsf{ssid}, P_i, P_j)$.

- First Round: P_i (with random tape ω_i) first generates a pair of signing/verification keys $(\mathsf{SK}_i, \mathsf{VK}_i)$ and a $\mathsf{DLCSCom}'$ commitment on W_i in $(\mathcal{C}_i, \mathcal{C}'_i)$, under $\ell_i = (\ell, \mathsf{VK}_i)$. It also computes a Pedersen commitment on $\mathcal{C}'_i$ in $\mathcal{C}''_i$ (with random exponent t). It then sends $(\mathsf{VK}_i, \mathcal{C}_i, \mathcal{C}''_i)$ to P_j;
- Second Round: P_j (with random tape ω_j) computes a LCSCom commitment on W_j in $\mathsf{Com}_j = \mathcal{C}_j$, with witness $\boldsymbol{r}$, under the label ℓ. It then generates a challenge ε on $\mathcal{C}_i$ and hashing/projection keys hk_i and the corresponding hp_i for the equality test on Com_i ("Com_i is a valid commitment of W_j", this only requires the value ξ_i computable thanks to $\mathcal{C}_i$). It then sends $(\mathcal{C}_j, \varepsilon, \mathsf{hp}_i)$ to P_i;
- Third Round: user P_i can compute $\mathsf{Com}_i = \mathcal{C}_i \times \mathcal{C}_i'^{\varepsilon}$ and witness $\mathbf{z}$ (from ε and ω_i), it generates hashing/projection keys hk_j and hp_j for the equality test on Com_j. It finally signs all the flows using SK_i in σ_i and send $(\mathcal{C}'_i, t, \mathsf{hp}_j, \sigma_i)$ to P_j;
- Hashing: P_j first checks the signature and the validity of the Pedersen commitment (thanks to t), it computes $\mathsf{Com}_i = \mathcal{C}_i \times \mathcal{C}_i'^{\varepsilon}$. P_i computes K_i and P_j computes K_j as follows:

$$K_i = \mathsf{Hash}(\mathsf{hk}_j, L'_j, \ell, \mathsf{Com}_j) \cdot \mathsf{ProjHash}(\mathsf{hp}_i, L_i, \ell_i, \mathsf{Com}_i; \mathbf{z})$$
$$K_j = \mathsf{ProjHash}(\mathsf{hp}_j, L_j, \ell, \mathsf{Com}_j; \boldsymbol{r}) \cdot \mathsf{Hash}(\mathsf{hk}_i, L'_i, \ell_i, \mathsf{Com}_i)$$

Fig. 5. Password-based Authenticated Key Exchange

groups elements plus 1 scalar, a verification key and a one-time signature, while P_j sends 3 group elements and 1 scalar (all group elements can be in the smallest group): 11 group elements and 2 scalars in total.

Verifier-Based PAKE. The above scheme can be modified into an efficient PAKE protocol that is additionally secure against *server compromise*: the so-called verifier-based PAKE, where the client owns a password pw, while the server knows a verifier only, such as g^{pw}, so that in case of break-in to the server, the adversary will not immediately get all the passwords.

To this aim, as usually done, one first does a PAKE with g^{pw} as common password, then asks the client to additionally prove it can compute the Diffie-Hellman value h^{pw} for a basis h chosen by the server. Ideally, we could implement this trick, where the client P_j just considers the equality test between the g^{pw} and the value committed by the server for the language $L'_i = L_j$, while the server P_i considers the equality test with $(g^{\mathsf{pw}}, h^{\mathsf{pw}})$, where h is sent as its contribution to the public part of the language by the server $L_i = L'_j$. Since the server chooses h itself, it chooses it as $h = g^{\alpha}$, for an ephemeral random α, and can thus compute $h^{\mathsf{pw}} = (g^{\mathsf{pw}})^{\alpha}$. On its side, the client can compute this value since it knows pw. The client could thus commit to $(g^{\mathsf{pw}}, h^{\mathsf{pw}})$, in order to prove its knowledge of pw, whereas the server could just commit to g^{pw}. Unfortunately, from the extractability of the server commitment, one would just get g^{pw}, which is not enough to simulate the client.

To make it in a provable way, the server chooses an ephemeral h as above, and they both run the previous PAKE protocol with $(g^{\mathsf{pw}}, h^{\mathsf{pw}})$ as common password,

and mutually checked: h is seen as the pub part, hence the preliminary flows are required.

Credential-Authenticated Key Exchange. In [8], the authors proposed instantiations of the CAKE primitive for conjunctions of atomic policies that are defined algebraically by relations of the form $\prod_{j=1}^{k} g_j^{F_j} = 1$ where the g_j's are elements of an abelian group and F_j's are integer polynomials in the variables committed by the users.

The core of their constructions relies on their practical UC zero-knowledge proof. There is no precise instantiation of such proof, but it is very likely to be inefficient. Their proof technique indeed requires to transform the underlying Σ-protocols into corresponding Ω-protocols [16] by verifiably encrypting the witness. An Ω-protocol is a Σ-protocol with the additional property that it admits a polynomial-time straight-line extractor. Since the witnesses are scalars in their algebraic relations, their approach requires either inefficient bit-per-bit encryption of these witnesses or Paillier encryption in which case the problem of using group with different orders in the representation and in the encryption requires additional overhead.

Even when used with Σ-protocols, their PAKE scheme without UC-security, requires at least two proofs of knowledge of representations that involve at least 30 group elements (if we assume the encryption to be linear Cramer Shoup), and some extra for the last proof of existence (*cf.* [9]), where our PAKE requires less than 20 group elements. Anyway they say, their PAKE scheme is less efficient than [11], which needed 6 rounds and around 30 modular exponentiations per user, while our efficient PAKE requires less than 40 exponentiations, in total, in only 3 rounds. Our scheme is therefore more efficient than the scheme from [11] for the same security level (*i.e.* UC-security with static corruptions).

Secret-Handshakes. We can also instantiate a (linkable) Secret Handshakes protocol, using our scheme with two different languages: P_i will commit to a valid signature σ_i on a message m_i (his identity for example), under a private verification key vk_i, and expects P_j to commit to a valid signature on a message m'_j under a private verification key vk'_j; but P_j will do analogously with a signature σ_j on m_j under vk_j, while expecting a signature on m'_i under vk'_i. The public parts of the signature (the second component) are sent in clear with the commitments.

In a regular Secret Handshakes both users should use the same languages. But here, we have a more general situation (called *dynamic matching* in [2]): the two participants will have the same final value if and only if they both belong to the organization the other expects. If one lies, our protocol guarantees no information leakage. Furthermore, the semantic security of the session is even guaranteed with respect to the authorities, in a forward-secure way (this property is also achieved in [19] but in a weaker security model). Finally, our scheme supports revocation and can handle roles as in [2].

Standard secret handshakes, like [2], usually work with credentials delivered by a unique authority, this would remove our need for a hidden verification key, and private part of the language. Both users would only need to commit

to signatures on their identity/credential, and show that they are valid. This would require a dozen of group elements with our approach. Their construction requires only 4 elements under BDH, however it relies on the asymmetric Waters IBE with only two elements, whereas the only security proof known for such IBE [15] requires an extra term in $\mathbb{G}_2$ which would render their technique far less efficient, as several extra terms would be needed to expect a provably secure scheme. While sometimes less effective, our LAKE approach can manage Secret Handshakes, and provide additional functionalities, like more granular control on the credential as part of them can be expressly hidden by both the users. More precisely, we provide affiliation-hiding property and let third parties unaware of the success/failure of the protocol.

Unlinkable Secret-Handshakes. Moving the users' identity from the public pub part to individual private priv part, and combining our technique with [7], it is also possible to design an *unlinkable* Secret Handshakes protocol [19] with practical efficiency. It illustrates the case where committed values have to be proven in a strict subset of $\mathbb{G}$, as one has to commit to bits: the signed message M is now committed and not in clear, it thus has to be done bit-by-bit since the encoding $\mathcal{G}$ does not allow algebraic operations with the content to apply the Waters function on the message. It is thus possible to prove the knowledge of a Waters signature on a private message (identity) valid under a private verification key. Additional relations can be required on the latter to make authentication even stronger.

Acknowledgments. This work was supported in part by the European Commission through the FP7-ICT-2011-EU-Brazil Program under Contract 288349 SecFuNet and the ICT Program under Contract ICT-2007-216676 ECRYPT II.

References

1. Abdalla, M., Chevalier, C., Pointcheval, D.: Smooth Projective Hashing for Conditionally Extractable Commitments. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 671–689. Springer, Heidelberg (2009)
2. Ateniese, G., Kirsch, J., Blanton, M.: Secret handshakes with dynamic and fuzzy matching. In: NDSS 2007. The Internet Society (February/March 2007)
3. Balfanz, D., Durfee, G., Shankar, N., Smetters, D.K., Staddon, J., Wong, H.-C.: Secret handshakes from pairing-based key agreements. In: IEEE Symposium on Security and Privacy, pp. 180–196. IEEE Computer Society (2003)
4. Barak, B., Canetti, R., Lindell, Y., Pass, R., Rabin, T.: Secure Computation Without Authentication. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 361–377. Springer, Heidelberg (2005)
5. Bellovin, S.M., Merritt, M.: Encrypted key exchange: Password-based protocols secure against dictionary attacks. In: 1992 IEEE Symposium on Security and Privacy, pp. 72–84. IEEE Computer Society Press (May 1992)
6. Ben Hamouda, F., Blazy, O., Chevalier, C., Pointcheval, D., Vergnaud, D.: Efficient UC-Secure Authenticated Key-Exchange for Algebraic Languages. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 272–291. Springer, Heidelberg (2013), Full version available from the web page of the authors or from http://eprint.iacr.org/2012/284

7. Blazy, O., Pointcheval, D., Vergnaud, D.: Round-Optimal Privacy-Preserving Protocols with Smooth Projective Hash Functions. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 94–111. Springer, Heidelberg (2012)
8. Camenisch, J., Casati, N., Gross, T., Shoup, V.: Credential Authenticated Identification and Key Exchange. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 255–276. Springer, Heidelberg (2010)
9. Camenisch, J., Krenn, S., Shoup, V.: A Framework for Practical Universally Composable Zero-Knowledge Protocols. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 449–467. Springer, Heidelberg (2011)
10. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd FOCS, pp. 136–145. IEEE Computer Society Press (October 2001)
11. Canetti, R., Halevi, S., Katz, J., Lindell, Y., MacKenzie, P.: Universally Composable Password-Based Key Exchange. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 404–421. Springer, Heidelberg (2005)
12. Canetti, R., Rabin, T.: Universal Composition with Joint State. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 265–281. Springer, Heidelberg (2003)
13. Cramer, R., Kiltz, E., Padró, C.: A Note on Secure Computation of the Moore-Penrose Pseudoinverse and Its Application to Secure Linear Algebra. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 613–630. Springer, Heidelberg (2007)
14. Cramer, R., Shoup, V.: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
15. Ducas, L.: Anonymity from Asymmetry: New Constructions for Anonymous HIBE. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 148–164. Springer, Heidelberg (2010)
16. Garay, J.A., MacKenzie, P.D., Yang, K.: Strengthening zero-knowledge protocols using signatures. Journal of Cryptology 19(2), 169–209 (2006)
17. Gennaro, R., Lindell, Y.: A Framework for Password-Based Authenticated Key Exchange. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 524–543. Springer, Heidelberg (2003)
18. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
19. Jarecki, S., Liu, X.: Private Mutual Authentication and Conditional Oblivious Transfer. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 90–107. Springer, Heidelberg (2009)
20. Lindell, Y.: Highly-Efficient Universally-Composable Commitments Based on the DDH Assumption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 446–466. Springer, Heidelberg (2011)
21. Shacham, H.: A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074 (2007)
22. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

Tighter Reductions for Forward-Secure Signature Schemes

Michel Abdalla, Fabrice Ben Hamouda, and David Pointcheval

Departement d'Informatique, École normale supérieure, Paris, France
{Michel.Abdalla,Fabrice.Ben.Hamouda,David.Pointcheval}@ens.fr
http://www.di.ens.fr/users/{mabdalla,fbenhamo,pointche}

Abstract. In this paper, we revisit the security of factoring-based signature schemes built via the Fiat-Shamir transform and show that they can admit tighter reductions to certain decisional complexity assumptions such as the quadratic-residuosity, the high-residuosity, and the ϕ-hiding assumptions. We do so by proving that the underlying identification schemes used in these schemes are a particular case of the lossy identification notion recently introduced by Abdalla *et al.* at Eurocrypt 2012. Next, we show how to extend these results to the forward-security setting based on ideas from the Itkis-Reyzin forward-secure signature scheme. Unlike the original Itkis-Reyzin scheme, our construction can be instantiated under different decisional complexity assumptions and has a much tighter security reduction. Finally, we show that the tighter security reductions provided by our proof methodology can result in concrete efficiency gains in practice, both in the standard and forward-security setting, as long as the use of stronger security assumptions is deemed acceptable. All of our results hold in the random oracle model.

1 Introduction

A common paradigm for constructing signature schemes is to apply the Fiat-Shamir transform [9] to a secure three-move canonical identification protocol. In these protocols, the prover first sends a commitment to the verifier, which in turn chooses a random string from the challenge space and sends it back to the prover. Upon receiving the challenge, the prover sends a response to the verifier, which decides whether or not to accept based on the conversation transcript and the public key. To obtain the corresponding signature scheme, one simply makes the signing and verification algorithms non-interactive by computing the challenge as the hash of the message and the commitment. As shown by Abdalla *et al.* in [1], the resulting signature scheme can be proven secure in the random oracle model as long as the identification scheme is secure against passive adversaries and the commitment has large enough min-entropy. Unfortunately, the reduction to the security of the identification scheme is not tight and loses a factor q_h, where q_h denotes the number of queries to the random oracle.

If one assumes additional properties about the identification scheme, one can avoid impossibility results such as those in [10,27,31] and obtain a signature

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 292–311, 2013.

scheme with a tighter proof of security. For instance, in [22], Micali and Reyzin introduced a new method for converting identification schemes into signature schemes, known as the "swap method", in which they reverse the roles of the commitment and challenge. More precisely, in their transform, the challenge is chosen uniformly at random from the challenge space and the commitment is computed as the hash of the message and the challenge. Although they only provided a tight security proof for the modified version of Micali's signature scheme [20], their method generalizes to any scheme in which the prover can compute the response given only the challenge and the commitment, such as the factoring-based schemes in [8,9,12,24,25]. This is due to the fact that the prover in these schemes possesses a trapdoor (such as the factorization of the modulus in the public key) which allows it to compute the response. On the other hand, their method does not apply to discrete-log-based identification schemes in which the prover needs to know the discrete log with respect to the commitment when computing the response, such as in [30].

In 2003, Katz and Wang [17] showed that tighter security reductions can be obtained even with respect to the Fiat-Shamir transform, by relying on a proof of membership rather than a proof of knowledge. In particular, using this idea, they proposed a signature scheme with a tight security reduction to the hardness of the DDH problem. They also informally mentioned that one could obtain similar results based on the quadratic-residuosity problem by relying on a proof that shows that a set of elements in $\mathbb{Z}_N^*$ are all quadratic residues. This result was recently extended to other settings by Abdalla *et al.* [3], who presented three new signature schemes based on the hardness of the short exponent discrete log problem [28,32], on the worst-case hardness of the shortest vector problem in ideal lattices [18,29], and on the hardness of the Subset Sum problem [14,23]. Additionally, they also formalized the intuition in [17] by introducing the notion of lossy identification schemes and showing that any such schemes can be transformed into a signature scheme via the Fiat-Shamir transform while preserving the tightness of the reduction.

Tight security from lossy identification. In light of these recent results, we revisit in this paper the security of factoring-based signature schemes built via the Fiat-Shamir transform. Even though the swap method from [22] could be applied in this setting (resulting in a slightly different scheme), our first contribution is to show that these signature schemes admit tight security reductions to certain decisional complexity assumptions such as the quadratic-residuosity, the high-residuosity [26], and the ϕ-hiding [6] assumptions. We do so by showing that the underlying identification schemes used in these schemes are a particular case of a lossy identification scheme [3]. As shown in Section 4.1 in the case of the Guillou-Quisquater signature scheme [12], our tighter security reduction can result in concrete efficiency gains with respect to the swap method. However, this comes at the cost of relying on a stronger security assumption, namely the ϕ-hiding [6] assumption.

Tighter reductions for forward-secure signatures. Unlike the swap method of Micali and Reyzin, the prover in factoring-based signature schemes

built via the Fiat-Shamir transform does not need to know the factorization of the modulus in order to be able to compute the response. Using this crucial fact, the second main contribution of this paper is to extend our results to the forward-security setting. To achieve this goal, we first introduce in Section 3 the notion of lossy key-evolving identification schemes and show how the latter can be turned into forward-secure signature schemes using a generalized version of the Fiat-Shamir transform. As in the case of standard signature schemes, this transformation does not incur a loss of factor of q_h in the security reduction. Nevertheless, we remark that the reduction is not entirely tight as we lose a factor T corresponding to the total number of time periods.

After introducing the notion of lossy key-evolving identification schemes, we show in Section 4.2 that a variant of the Itkis-Reyzin forward-secure signature scheme [15] (which can be seen as an extension of the Guillou-Quisquater scheme to the forward-security setting) admits a much tighter security reduction, albeit to a stronger assumption, namely the ϕ-hiding assumption.

Concrete security. As in the case of standard signature schemes, the tighter security reductions provided by our proof methodology can result in concrete efficiency gains in practice. More specifically, as we show in Section 5, our variant of the Itkis-Reyzin scheme outperforms the original scheme for most concrete choices of parameters.

Generic factoring-based signatures and forward-secure signatures. As an additional contribution, we show in Section 6 that all the above-mentioned schemes can be seen as straightforward instantiations of a generic factoring-based forward-secure signature scheme. This enables us to not only easily prove the security properties of these schemes, but to also design a new forward-secure scheme based on a new assumption, the 2^t-strong-residuosity.

Organization. After recalling some definitions in Section 2, we introduce the notion of key-evolving lossy identification scheme and show how to transform such a scheme into a forward-secure signature scheme in Section 3. Then, in Section 4, we apply our security proof methodology to two cases: the Guillou-Quisquater scheme and its extension to the forward-secure case (i.e., our variant of the Itkis-Reyzin scheme). In Section 5, we compare this second scheme with the original Itkis-Reyzin scheme and the MMM scheme by Malkin, Micciancio and Miner [19]. Finally, we introduce our generic lossy key-evolving identification scheme and show various instantiations of it in Section 6.

2 Definitions

2.1 Notation and Conventions

Let $\mathbb{N}$ denote the set of natural numbers. If $n \in \mathbb{N}$, then $\{0,1\}^n$ denotes the set of n-bit strings, and $\{0,1\}^*$ is the set of all bit strings. The empty string is denoted $\perp$. If x is a string then $|x|$ denotes its length, and if S is a set then $|S|$ denotes its size. If S is finite, then $x \xleftarrow{\$} S$ denotes the assignment to x of an element chosen

uniformly at random from S. If $\mathcal{A}$ is an algorithm, then $y \leftarrow \mathcal{A}(x)$ denotes the assignment to y of the output of $\mathcal{A}$ on input x, and if $\mathcal{A}$ is randomized, then $y \xleftarrow{\$} \mathcal{A}(x)$ denotes that the output of an execution of $\mathcal{A}(x)$ with fresh coins assigned to y. Unless otherwise indicated, an algorithm may be randomized. We denote by $k \in \mathbb{N}$ the security parameter. Let $\mathbb{P}$ denote the set of primes and $\mathbb{P}_{\ell_e}$ denote the set of primes of length ℓ_e. All our schemes are in the random oracle model [5].

2.2 Complexity Assumptions

The security of the signature schemes being analyzed in this paper will be based on decisional assumptions over composite-order groups: the e-residuosity assumption, the ϕ-hiding assumption and a new assumption called the strong-2^t-residuosity. We also need to recall the strong-RSA assumption to be able to compare our scheme with the Itkis-Reyzin scheme [15].

Let N be the product of distinct large primes p_1 and p_2. We call such N an RSA modulus. Informally, the **e-residuosity** assumption states that the problem of deciding whether a given element y in $\mathbb{Z}_N^*$ is an e-residue or not is intractable without knowing the factorization of N. Remember that an element $y \in \mathbb{Z}_N^*$ is said to be an e-residue if there exists an element $x \in \mathbb{Z}_N^*$ such that $y = x^e \bmod N$. If $e = 2$, this assumption is called the **quadratic-residuosity** assumption. Furthermore, if we extend it to $N = e^2$, with e an RSA modulus, this is called the **high-residuosity** assumption [26]. Likewise, the **ϕ-hiding** assumption, introduced by Cachin, Micali, and Stadler in [6], states that it is hard for an adversary to tell whether a prime number e divides the order of the group $\mathbb{Z}_N^*$ or not. Next, we introduce the **strong-2^t-residuosity** assumption that states that it is hard for an adversary to decide whether a given element y in $\mathbb{Z}_N^*$ is a 2^t-residue or is even not a 2-residue, when 2^t divides $p_1 - 1$ and $p_2 - 1$. Finally, the **strong-RSA** assumption states that, given an element $y \in \mathbb{Z}_N^*$, it is hard for an adversary to find an integer $e \geq 2$ and an element $x \in \mathbb{Z}_N^*$ such that $y = x^e \bmod N$.

For each of these assumptions, the underlying problem is said to be (t, ε)-hard, if no adversary running in time at most t is able to solve the problem with probability at least ε. Formal descriptions of the assumptions can be found in the full version [2].

2.3 Forward-Secure Signature Schemes

A forward-secure signature scheme is a key-evolving signature scheme in which the secret key is updated periodically while the public key remains the same throughout the lifetime of the scheme [4]. Each time period has a secret signing key associated with it, which can be used to sign messages with respect to that time period. The validity of these signatures can be checked with the help of a verification algorithm. At the end of each time period, the signer in possession of the current secret key can generate the secret key for the next time period via an update algorithm. Moreover, old secret keys are erased after a key update.

Formally, a key-evolving signature scheme is defined by a tuple of algorithms $\mathcal{FS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ and a message space $\mathcal{M}$, providing the following functionality. Via $(pk, sk) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$, a user can run the probabilistic key generation algorithm KG to obtain a pair (pk, sk_1) of public and secret keys for a given security parameter k and a given total number of periods T. sk_1 is the secret key associated with time period 1. Via $sk_{i+1} \leftarrow \mathsf{Update}(sk_i)$, the user in possession of the secret key sk_i associated with time period $i \leq T$ can generate a secret key sk_{i+1} associated with time period $i+1$. By convention, $sk_{T+1} = \perp$. Via $\langle \sigma, i \rangle \xleftarrow{\$} \mathsf{Sign}(sk_i, M)$, the user in possession of the secret key sk_i associated with time period $i \leq T$ can generate a signature $\langle \sigma, i \rangle$ for a message $M \in \mathcal{M}$ for period i. Finally, via $d \leftarrow \mathsf{Ver}(pk, \langle \sigma, i \rangle, M)$, one can run the deterministic verification algorithm to check if σ is a valid signature for a message $M \in \mathcal{M}$ for period i and public key pk, where $d = 1$ if the signature is correct and 0 otherwise. For correctness, it is required that for all honestly generated keys $(sk_1, \ldots, sk_T)$ and for all messages $M \in \mathcal{M}$, $\mathsf{Ver}(pk, \mathsf{Sign}(sk_i, M), M) = 1$ holds with all but negligible probability.

Informally, a key-evolving signature scheme is **existentially forward-secure** under adaptive chosen-message attack (EUF-CMA), if it is infeasible for an adversary —also called forger— to forge a signature σ^* on a message M^* for a time period i^*, even with access to the secret key for a period $i > i^*$ (and thus to all the subsequent secret keys; this period i is called the breakin period) and to signed messages of his choice for any period (via a signing oracle), as long as he has not requested a signature on M^* for period i^* to the signing oracle. This notion is a generalization of the existential unforgeability under adaptive chosen-message attacks (EUF-CMA for signature schemes) [11] to key-evolving signature scheme and a slightly stronger variant of the definition in [4]. In particular, we do not restrict the adversary to only perform signing queries with respect to the current time period.

In the remainder of the paper, we also use a stronger notion: **forward security** (SUF-CMA). In this notion, the forger is allowed to produce a signature σ^* on a message M^* for a period i^*, such that the triple (M^*, i^*, σ^*) is different from all the triples produced by the signing oracle. More formally, a key-evolving signature scheme is $(t, q_h, q_s, \varepsilon)$-(existentially)-forward-secure if no adversary running in time at most t and making at most q_h queries to the random oracle and q_s queries to the signing oracle can break the (existential) forward security with probability at least ε. All the formal security notions and the comparison with [4], together with other security notions (used for detailed comparisons), can be found in the full version [2].

3 Lossy Key-Evolving Identification and Signature Schemes

In this section, we present a new notion, called lossy key-evolving identification scheme, which combines the notions of lossy identification schemes [3], which

can be transformed to tightly secure signature scheme, and key-evolving identification schemes [4], which can be transformed to forward-secure signature via a generalized Fiat-Shamir transform (not necessarily tight, and under some conditions). Although this new primitive is not very useful for practical real-world applications, it is a tool that will enable us to construct forward-secure signatures with tight reductions, via the generalized Fiat-Shamir transform described in Section 3.2.

3.1 Lossy Key-Evolving Identification Scheme

The operation of a key-evolving identification scheme is divided into time periods $1, \ldots, T$, where a different secret is used in each time period, and such that the secret key for a period $i+1$ can be computed from the secret key for the period i. The public key remains the same in every time period. In this paper, a key-evolving identification scheme is a three-move protocol in which the prover first sends a **commitment** cmt to the verifier, then the verifier sends a **challenge** ch uniformly at random, and finally the prover answers by a **response** rsp. The verifier's final decision is a deterministic function of the conversation with the prover (the triple (cmt, ch, rsp)), of the public key, and of the index of the current time period.

Informally, a lossy key-evolving identification scheme has $T+1$ kinds of public keys: normal public keys, which are used in the real protocol, and i-lossy public keys, for $i \in \{1, \ldots, T\}$, which are such that no prover (even not computationally bounded) should be able to make the verifier accept for the period i with non-negligible probability. Furthermore, for each period i, it is possible to generate a i-lossy public key, such that the latter is indistinguishable from a normal public key even if the adversary is given access to any secret key for period $i' > i$.

More formally, a lossy key-evolving identification scheme $I\mathcal{D}$ is defined by a tuple $(\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \ell_c, \mathsf{Ver})$ such that:

- KG is the normal key generation algorithm which takes as input the security parameter k and the number of periods T and outputs a pair (pk, sk_1) containing the public key and the prover's secret key for the first period.
- LKG is the lossy key generation algorithm which takes as input the security parameter k and the number of periods T and a period i and outputs a pair (pk, sk_{i+1}) containing a i-lossy public key pk and a prover's secret key for period $i+1$ ($sk_{T+1} = \perp$).
- Update is the deterministic secret key update algorithm which takes as input a secret key sk_i for period i and outputs a secret key sk_{i+1} for period $i+1$ if sk_i is a secret key for some period $i < T$, and $\perp$ otherwise. We write Update^j the function Update composed j times with itself ($\mathsf{Update}^j(sk_i)$ is a secret key sk_{i+j} for period $i+j$, if $i+j \leq T$).
- Prove is the prover algorithm which takes as input the secret key for the current period, the current conversation transcript (and the current state st associated with it, if needed) and outputs the next message to be sent to

the verifier, and the next state (if needed). We suppose that any secret key sk_i for period i always contains i, and so i is not an input of Prove.

- ℓ_c is a polynomial; $\ell_c(k)$ (often simply denoted ℓ_c) is the length of the challenge sent by the verifier.
- Ver is the deterministic verification algorithm which takes as input the conversation transcript and the period i and outputs 1 to indicate acceptance, and 0 otherwise.

A randomized transcript generation oracle $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$ is associated to each $\mathcal{ID}$, k, and (pk, sk_i). $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$ takes no inputs and returns a random transcript of an "honest" execution for period i. More precisely, the transcript generation oracle $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$ is defined as follows:

function $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$
$(cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)$; $ch \xleftarrow{\$} \{0,1\}^{\ell_c}$; $rsp \xleftarrow{\$} \mathsf{Prove}(sk_i, cmt, ch, st)$
return (cmt, ch, rsp)

An identification scheme is said to be lossy if it has the following properties:

(1) **Completeness of normal keys.** $\mathcal{ID}$ is said to be complete, if for every period i, every security parameter k and all honestly generated keys $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k)$, $\mathsf{Ver}(pk, cmt, ch, rsp, i) = 1$ holds with probability 1 when $(cmt, ch, rsp) \xleftarrow{\$} \mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}()$, with $sk_i = \mathsf{Update}^{i-1}(sk_1)$.

(2) **Simulatability of transcripts.** Let (pk, sk_1) be the output of $\mathsf{KG}(1^k)$ for a security parameter k, and sk_i be the output of $\mathsf{Update}^{i-1}(sk_1)$. Then, $\mathcal{ID}$ is said to be ε-simulatable if there exists a probabilistic polynomial time algorithm $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,i,k}$ with no access to any secret key, which can generate transcripts $\{(cmt, ch, rsp)\}$ whose distribution is statistically indistinguishable from the transcripts output by $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$, where ε is an upper-bound for the statistical distance. When $\varepsilon = 0$, then $\mathcal{ID}$ is said to be simulatable.

(3) **Indistinguishability of keys.** Consider the two following experiments $\mathbf{Exp}^{\text{ind-keys-real}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$ and $\mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$ $(i \in \{1, \ldots, T\})$:

$\mathbf{Exp}^{\text{ind-keys-real}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$	$\mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$
$(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$	$(pk, sk_{i+1}) \xleftarrow{\$} \mathsf{LKG}(1^k, 1^T, i)$
$sk_{i+1} \xleftarrow{\$} \mathsf{Update}^i(sk_1)$	return $\mathsf{D}_i(pk, sk_{i+1})$
return $\mathsf{D}_i(pk, sk_{i+1})$	

D is said to (t, ε)-solve the key-indistinguishability problem for period i if it runs in time t and

$$\left|\Pr\left[\mathbf{Exp}^{\text{ind-keys-real}}_{\mathcal{ID},k,i}(\mathsf{D}_i) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathcal{ID},k,i}(\mathsf{D}_i) = 1\right]\right| \geq \varepsilon.$$

Furthermore, we say that $\mathcal{ID}$ is (t, ε)-key-indistinguishable, if, for any i, no algorithm (t, ε)-solves the key-indistinguishability problem for period i.

(4) **Lossiness.** Let I_i be an impersonator for period i $(i \in \{1, \ldots, T\})$, st be its state. We consider the experiment $\mathbf{Exp}^{\text{los-imp-pa}}_{\mathcal{ID},k,i}(\mathsf{I}_i)$ played between I_i and a hypothetical challenger:

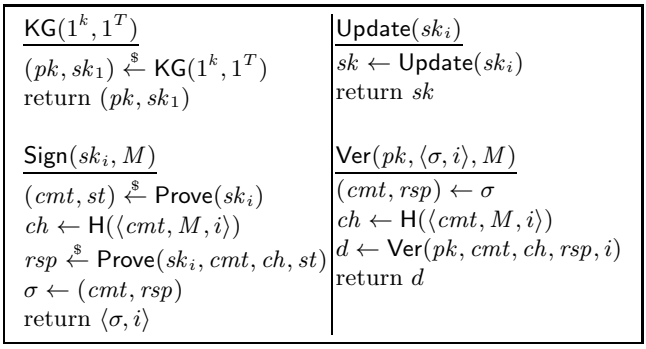


Fig. 1. Generalized Fiat-Shamir transform for forward-secure signature

$$\mathbf{Exp}^{\text{los-imp-pa}}_{I\!\mathcal{D},k,i}(\mathsf{I}_i)$$
$$(pk, sk_{i+1}) \xleftarrow{\$} \mathsf{LKG}(1^k, 1^T, i)\,;\,(cmt, st) \xleftarrow{\$} \mathsf{I}_i(pk, sk_{i+1})$$
$$ch \xleftarrow{\$} \{0,1\}^{\ell_c}\,;\,rsp \xleftarrow{\$} \mathsf{I}_i(ch, st)$$
$$\text{return } \mathsf{Ver}(pk, cmt, ch, rsp, i)$$

I_i is said to ε-solve the impersonation problem with respect to i-lossy public keys if $\Pr\left[\mathbf{Exp}^{\text{los-imp-pa}}_{I\!\mathcal{D},k,i}(\mathsf{I}_i) = 1\right] \geq \varepsilon$. Furthermore, $I\!\mathcal{D}$ is said to be ε-lossy if, for any period $i \in \{1, \ldots, T\}$, no (computationally unrestricted) algorithm ε-solves the impersonation problem with respect to i-lossy keys.

We remark that, for $T = 1$, a key-evolving lossy identification scheme becomes a standard lossy identification scheme[1], described in [3].

Finally, we say that $I\!\mathcal{D}$ is **response-unique** if for all normal public keys pk or for all lossy keys pk, for all periods $i \in \{1, \ldots, T\}$, for all messages M, for all bit strings cmt[2], and for all challenges ch, there exists at most one response rsp such that $\mathsf{Ver}(pk, cmt, ch, rsp, i) = 1$.

3.2 Generalized Fiat-Shamir Transform

The forward-secure signature schemes considered in this paper are built from a key-evolving identification scheme via a straightforward generalization of the Fiat-Shamir transform [9], depicted in Fig. 1. More precisely, the signature for period i is just the signature obtained from a Fiat-Shamir transform with secret key $sk_i = \mathsf{Update}^{i-1}(sk_1)$ (with the period i included in the random oracle input).

Let $\mathcal{FS}[I\!\mathcal{D}] = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be the signature scheme obtained via this generalized Fiat-Shamir transform. The following theorem is a generalization of (a special case of) Theorem 1 in [3], where we assume perfect completeness.

[1] Contrary to the definition of lossiness given in [3], the impersonator I_1 does not have access to an oracle $\widetilde{\mathsf{Tr}}^{I\!\mathcal{D}}_{pk,1,k}$ in $\mathbf{Exp}^{\text{los-imp-pa}}_{I\!\mathcal{D},k,1}(\mathsf{I}_1)$. However, we remark that this has no impact on the security definition as the execution of $\widetilde{\mathsf{Tr}}^{I\!\mathcal{D}}_{pk,1,k}$ does not require any secret information.

[2] Not necessarily a correctly generated commitment, but any bit string.

Theorem 1. *Let* $\mathcal{ID} = (\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \ell_c, \mathsf{Ver})$ *be a key-evolving lossy identification scheme whose commitment space has min-entropy at least* β *(for every period* i*), let* H *be a random oracle, and let* $\mathcal{FS}[\mathcal{ID}] = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ *be the signature scheme obtained via the generalized Fiat-Shamir transform. If* $\mathcal{ID}$ *is* ε_s*-simulatable, complete,* (t', ε_k)*-key-indistinguishable, and* ε_ℓ*-lossy, then* $\mathcal{FS}[\mathcal{ID}]$ *is* $(t, q_h, q_s, \varepsilon)$*-existentially-forward-secure in the random oracle model for:*

$$\varepsilon = T\ (\varepsilon_k + (q_h + 1)\varepsilon_\ell) + q_s\varepsilon_s + (q_h + 1)q_s/2^\beta$$
$$t \approx t' - (q_s\, t_{\mathsf{Sim-Sign}} + (T-1)\, t_{\mathsf{Update}})$$

where $t_{\mathsf{Sim-Sign}}$ *denotes the average time of a query to the simulated transcript function* $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,i,k}$ *and* t_{Update} *denotes the average time of a query to* Update*. Furthermore, if* $\mathcal{ID}$ *is response-unique,* $\mathcal{FS}[\mathcal{ID}]$ *is also* $(t, q_h, q_s, \varepsilon)$*-forward-secure.*

Actually, if we choose $T = 1$ in the previous theorem, we get a slightly improved special case of Theorem 1 in [3], since the forward security for $T = 1$ is exactly the strong unforgeability for a signature scheme. The proof of this theorem can be found in the full version [2] and is very similar to the proof in [3], except that we need to guess the period i^* of the signature output by the adversary, in order to choose the correct lossy key. That is why we lose a factor T in the reduction.

Remark 2. As in the standard Fiat-Shamir transform, the signature obtained via the generalized transform consists of a commitment-response pair. However, in all schemes proposed in this paper, the commitment can be recovered from the challenge and the response. Hence, since the challenge is often shorter than the commitment, it is generally better to use the challenge-response pair as the signature in our schemes. Obviously, this change does not affect the security of our schemes.

4 Tighter Security Reductions for Guillou-Quisquater-Like Schemes

In this section, we prove tighter security reductions for the Guillou-Quisquater scheme (GQ, [12]) and for a slight variant of the Itkis-Reyzin scheme (IR, [15]), which can also be seen as a forward-secure extension of the GQ scheme. We analyze the practical performance of this new scheme in the next section of this article. Detailed proofs for these schemes are available in the full version [2].

4.1 Guillou-Quisquater Scheme

Let us describe the identification scheme corresponding to the GQ signature scheme, before presenting our tight reduction and comparing it with the swap method.

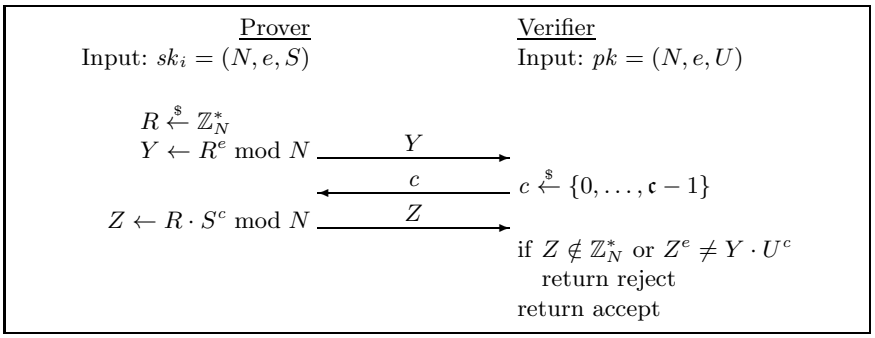


Fig. 2. Description of the GQ identification scheme ($U = S^e \bmod N$)

SCHEME. Let N be a product of two distinct ℓ_N-bit primes p_1, p_2 and let e be a ℓ_e-bit prime, coprime with $\phi(N) = (p_1 - 1)(p_2 - 1)$, chosen uniformly at random. Let S be an element chosen uniformly at random in $\mathbb{Z}_N^*$ and let $U = S^e \bmod N$. Let $\mathfrak{c} = 2^{\ell_e}$. The public key is $pk = (N, e, U)$ and the secret key is $sk = (N, e, S)$.

The goal of the identification scheme is to prove U is a e-residue. The identification scheme is depicted in Fig. 2 and works as follows. First, the prover chooses a random element $R \in \mathbb{Z}_N^*$, computes $Y \leftarrow R^e \bmod N$. It sends Y to the verifier, which in turn chooses $c \in \{0, \ldots, \mathfrak{c} - 1\}$ and returns it to the prover. Upon receiving c, the prover computes $Z \leftarrow R \cdot S^c \bmod N$ and sends this value to the verifier. Finally, the verifier checks whether $Z \in \mathbb{Z}_N^*$ and $Z^e = Y \cdot U^c$ and accepts only in this case[3].

SECURITY. The previous proofs of the GQ schemes looses a factor q_h in the reduction. In this paragraph, we prove the previously described identification scheme $I\mathcal{D}$ is a lossy identification scheme, under the ϕ-hiding assumption. This yields a security proof of the strong unforgeability of the GQ scheme, with a tight reduction to this assumption.

The algorithm LKG chooses e and $N = p_1 p_2$ such that e divides $p_1 - 1$, instead of being coprime with $\phi(N)$, and chooses U uniformly at random among the non-e-residue modulo N. In the full version [2], we show that if U is chosen uniformly at random in $\mathbb{Z}_N^*$, it is not an e-residue with probability $1 - 1/e$ and that it is possible to efficiently check whether U is an e-residue or not if the factorization of N is known: U is a e-residue if and only if, for any $k \in \{1, 2\}$, e does not divide $p_k - 1$ or $U^{(p_k - 1)/e} = 1 \bmod p_k$.

The proof that $I\mathcal{D}$ is **complete** follows immediately from the fact that, if $U = S^e \bmod N$, an honest execution of the protocol will always result in acceptance as $Z^e = (R \cdot S^c)^e = R^e \cdot (S^e)^c = Y \cdot U^c$.

The **simulatability** of $I\mathcal{D}$ follows from the fact that, given $pk = (N, e, U)$, we can easily generate transcripts whose distribution is perfectly indistinguishable from the transcripts output by an honest execution of the protocol. This is

[3] The test $Z \in \mathbb{Z}_N^*$ can be replaced by the less expensive test $Z \bmod N \neq 0$, as explained in the full version [2].

done by choosing Z uniformly at random in $\mathbb{Z}_N^*$ and c uniformly at random in $\{0, \ldots, \mathfrak{c} - 1\}$, and setting $Y = Z^e / U^c$.

Let us prove the **key indistinguishability**. The distribution of normal public keys is indistinguishable from the one where e divides $\phi(N)$ and U is chosen uniformly at random, according to the ϕ-hiding assumption. And in this latter distribution, U is not a e-residue with probability $1 - 1/e$, so this distribution is statistically close to the distribution of lossy keys. Therefore, $I\mathcal{D}$ is key indistinguishable.

To show that $I\mathcal{D}$ is **lossy**, we note that, when the public key is lossy, for every element Y chosen by the adversary, there exists only one value of $c \in \{0, \ldots, \mathfrak{c} - 1\}$ for which there exists a response Z which is considered valid by the verifier. To see why, assume for the sake of contradiction that there exist two different values c_1 and c_2 in $\{0, \ldots, \mathfrak{c} - 1\}$ for which there exists a valid response. Denote by Z_1 and Z_2 one of the valid responses in each case. Without loss of generality, assume that $c_1 < c_2$. Since $Z_1^e = Y \cdot U^{c_1}$ and $Z_2^e = Y \cdot U^{c_2}$, we have that $(Z_2/Z_1)^e = U^{c_2 - c_1}$. As $c_2 - c_1$ is a positive number smaller than 2^{ℓ_e}, it is coprime with e (since e is a prime and $e \geq 2^{\ell_e}$). Therefore, according to Bezout theorem, there exists two integers u, v such that: $ue + v(c_1 - c_2) = 1$. So:

$$U = U^{ue + v(c_1 - c_2)} = (U^u)^e (U^{c_2 - c_1})^v = (U^u (Z_2/Z_1)^v)^e$$

and U is a e-residue, which is impossible. This means that the probability that a valid response Z_i exists in the case where U is not a e-residue is at most $1/\mathfrak{c}$. It follows that $I\mathcal{D}$ is $1/\mathfrak{c}$-lossy.

Comparison with the swap method. Applying the swap method [22] to the GQ identification scheme can also provide a signature with a tight reduction, to the RSA problem. However, in this case, the signing algorithm needs to compute the e-root of the output of the random oracle modulo N. Therefore, instead of requiring two exponentiation modulo N with a ℓ_e-bit exponent, the signing algorithm requires one such exponentiation and one exponentiation modulo N with a ℓ_N-bit exponent. And our signing algorithm will be $\ell_N/(2\ell_e)$ faster, for the same parameters and the same security level, if we consider the ϕ-hiding problem is as hard as the RSA problem. Furthermore, the swap method cannot be directly extended to the forward-secure extension of the GQ scheme, described in the next section, because the prover has to know the factorization of N.

A slight variant of the scheme. We can also chooses e uniformly at random among the ℓ_e-bit primes (without forcing that e is coprime with $\phi(N)$ in KG), because, with high probability, such a prime number will be coprime with $\phi(N)$.

4.2 Variant of the Itkis-Reyzin Scheme

Scheme. The idea of this forward-secure extension of the GQ scheme consists in using a different e for each period. More precisely, let $e_1, \ldots, e_T$ be T distinct ℓ_e-bit primes chosen uniformly at random. Let $f_i = e_{i+1} \ldots e_T$, $f_T = 1$ and $E = e_1 \ldots e_T$. Let S be an element chosen uniformly at random in $\mathbb{Z}_N^*$ and let

$U = S^E \bmod N$. Let $S_i = S^{E/e_i}$ and $S'_i = S^{E/f_i}$. Then the public key is $pk = (N, e_1, \ldots, e_T, U)$ and the secret key for period i is $sk_i = (N, e_i, \ldots, e_T, S_i, S'_i)$. We remark we can easily compute sk_{i+1} from sk_i, since $S_{i+1} = {S'_i}^{f_{i+1}} \bmod N$ and ${S'}_{i+1} = {S'_i}^{e_{i+1}} \bmod N$.

For period i, the identification scheme works exactly as the previous one with public key $pk = (N, e_i, U)$ and secret key $sk = (N, e_i, S_i)$.

For the sake of simplicity, in this naive description of the scheme, we store the exponents $e_1, \ldots, e_T$ in the public key and in the secret key. Therefore, the keys are linear in T, the number of periods. It is possible to have constant-size key, either by using fixed exponents, or by computing the exponents using a random oracle. This will be discussed in Section 5.1.

Security. The security proof is similar to the one for the previous scheme, with the main difference being the description of the lossy key generation algorithm LKG. More precisely, on input $(1^k, 1^T, i)$, the algorithm LKG generates e_i and $N = p_1 p_2$ such that e_i divides $p_1 - 1$, instead of being coprime with $\phi(N)$, and chooses U' uniformly at random among the non-e_i-residues modulo N. Then it chooses $T - 1$ distinct random ℓ_e-bit primes $e_1, \ldots, e_{i-1}, e_{i+1}, \ldots, e_T$, and sets $U = U'^{e_{i+1} \cdots e_T} \bmod N$, $S_{i+1} = U'^{e_{i+2} \cdots e_T} \bmod N$ and $S'_{i+1} = U'^{e_{i+1}} \bmod N$. The public key is $pk = (N, e_1, \ldots, e_T, U)$ and the secret key for period $i + 1$ is $sk_{i+1} = (N, e_{i+1}, \ldots, e_T, S_{i+1}, S'_{i+1})$ (or $\perp$ if $i = T$). We remark that, since U' is a non-e_i-residue, U is also a non-e_i-residue and so the public key pk is i-lossy.

5 Analysis of Our Variant of the Itkis-Reyzin Scheme

In this section, we analyze our variant of the IR scheme and compare it with the original IR scheme [15] and the MMM scheme [19].

5.1 Computation of the Exponents $e_1, \ldots, e_T$

As explained before, storing the exponents $e_1, \ldots, e_T$ in the keys is not a good idea since the key size becomes linear in T. Since we need $e_1, \ldots, e_T$ to be random primes to be able to do the reduction of key indistinguishability to the ϕ-hiding assumption, we can use a second random oracle H' which outputs prime numbers of length ℓ_e, and set $e_i = \mathsf{H}'(i)$.

An implementation of a random oracle for prime numbers using a classical random oracle is presented in the full version [2]. The construction is close to the construction of a PRF mapping to prime numbers in [13]. The idea is to hash the input value concatenated to a counter and to increment the counter until we get a prime number. One can prove that it behaves like a random oracle uniform over all primes, and that we can program it efficiently (property which is needed for the security reductions).

We finally remark that, we can always store e_i in the secret key for period i. The secret key length is increased only by a small amount and the signing algorithm becomes faster, since it does not need to recompute e_i.

Table 1. Choice of parameters

k	q_h	q_s	ℓ_e	ε_p	ℓ_N
80	2^{80}	2^{30}	123	2^{-80}	≥ 1248
128	2^{128}	2^{46}	171	2^{-128}	≥ 3248

5.2 Choice of Parameters

In order to be able to compare the original IR scheme with our scheme, we need to choose various parameters. In Table 1, we show our choice of parameters for two security levels: $k = 80$ bits and $k = 128$ bits. When choosing these parameters, we considered a value of $T = 2^{20}$, as it enables to update the key every hour for up to 120 years (please refer to the full version [2] for more details). In both cases, ε_p denotes the maximum error probability of the probabilistic primality test used in the random oracle for primes numbers H', whereas q_h and q_s specify the maximum number of queries to the random oracle and to the signing oracle, respectively, in the forward-security game. In the sequel, all the parameters are fixed except the length ℓ_N of the modulus.

5.3 Comparison with Existing Schemes

Comparison with the Itkis-Reyzin scheme. In this section, we compare the original IR scheme without optimization with our scheme (in which e_i is stored in the secret key sk_i, as in the IR scheme). The original IR scheme is very close to our scheme. The only differences are that the IR scheme requires that the factors p_1 and p_2 of the modulus N are safe primes[4] and that IR signatures for period i contain the used exponent e_i. Therefore the IR verification algorithm does not need to recompute the exponent, and is faster. In order to prevent an adversary from using an exponent for the breakin period to sign messages for an older period, the exponent has to be in a different set for each period. The security of the scheme comes from the strong-RSA assumption. Unfortunately, we cannot use such an optimization with our security reduction for our scheme, because we need to know which exponent the adversary will use to make the key lossy for this exponent. However, we remark in the full version [2] that the other optimizations of the original IR scheme can also be applied to our scheme.

Let us now compare the two schemes with the same security parameters (k, ℓ_e, ℓ_N), before analyzing the exact security. We first remark that for the same security parameters, our key generation algorithm is slightly faster since it does not require safe primes; and our signing and key update algorithms are as fast as the IR ones. The key and signature lengths of the signatures are nearly the same as the IR ones (IR signatures are only ℓ_e-bits longer than our signatures). The real difference is the verification time since our verification algorithm needs to recompute the e_i, contrary to the IR scheme. Verification consists of

[4] A safe prime p is an odd prime such that $(p-1)/2$ is also prime. This assumption is needed for the security reduction of the IR scheme.

Table 2. Time of verification algorithm (using parameters of Table 1)

		exponentiation		prime generation		verification orig.[a]		verification new[b]	
k	ℓ_N	mul.[c]	ms[d]	mul.[c]	ms[d]	mul.[c]	ms[d]	mul.[c]	ms[d]
k	ℓ_N	$\frac{3}{2}\,\ell_e\,{\ell_N}^2$	n/a	$(\frac{3}{2}\,kp + 2\,\ell_e)\,{\ell_e}^3$	n/a	$3\,\ell_e\,{\ell_N}^2$	n/a	$3\,\ell_e\,{\ell_N}^2 + (\frac{3}{2}\,kp + 2\,\ell_e)\,{\ell_e}^3$	n/a
80	1248	$0.29 \cdot 10^9$	0.15	$0.68 \cdot 10^9$	0.26	$0.58 \cdot 10^9$	0.30	$1.26 \cdot 10^9$	0.56
80	1920	$0.68 \cdot 10^9$	0.34	$0.68 \cdot 10^9$	0.26	$1.36 \cdot 10^9$	**0.68**	$2.04 \cdot 10^9$	**0.94**
80	6848	$8.65 \cdot 10^9$	3.09	$0.68 \cdot 10^9$	0.26	$17.3 \cdot 10^9$	**6.18**	$1.26 \cdot 10^9$	6.44
128	3248	$2.71 \cdot 10^9$	1.19	$2.67 \cdot 10^9$	0.82	$5.42 \cdot 10^9$	2.38	$8.09 \cdot 10^9$	3.10

[a] verification time of the original scheme (also equal to the signature time for both schemes), estimated using the time of the two exponentiations.

[b] verification time of our scheme, estimated using the time of the two exponentiations and of the prime generation.

[c] approximate theoretical complexity (see the full version [2]).

[d] time on an Intel Core i5 750 (2.67 GHz), using GMP version 5.0.4 (`http://gmplib.org`, a pseudo-random number generator is used as a random oracle.

two exponentiations (modulo N with a ℓ_e-bit exponent) for the original scheme and two exponentiations and an evaluation of the random prime oracle (roughly equivalent to a random prime generation) for our scheme.

Let us now focus on the exact security of the two schemes. As explained by Kakvi and Kiltz in [16], the best known attacks against the ϕ-hiding problems are the factorization of N. Let us also consider it is true for the strong RSA problem (since it just strengthens our result if it is not the case). As shown in the full version [2], with our choice of parameters, if we want $k = 80$ bits of security, we need to choose a modulo length ℓ_N such that the factorization is $k + \log_2(T) = 100$-bit hard (for our scheme) and $k + \log_2(Tq_h) = 180$-bit hard (for the original scheme). This corresponds to about $\ell_N \approx 1920$ and $\ell_N \approx 6848$ respectively, according to Ecrypt II [7]. In this case, according to Table 2, our verification algorithm is about 6 times faster (0.94ms vs 6.18ms) and our signing algorithm is about 9 times faster (0.68ms vs 6.18ms). And our scheme generates 3.5 times shorter signatures.

Comparison with the MMM scheme. The MMM scheme [19] is one of the most efficient generic constructions of forward-secure signatures (from any signature scheme), to the best of our knowledge. Furthermore, it does not require to fix the number of periods T. However, in the security proof, we have to bound the number of periods T the adversary can use (as query for the oracles **Sign** and **Breakin**). Its forward security can be reduced to the strong unforgeability of the underlying signature scheme with a loss of a factor T.

If we want to compare the MMM scheme with our variant of the IR scheme, the fairest solution is to instantiate the MMM scheme with the GQ scheme. Then we can use our tight reduction of the GQ scheme to the ϕ-hiding problem, to

prove that the resulting MMM scheme is forward-secure with a relatively tight (losing only a factor T) reduction to the ϕ-hiding problem. In this setting, the MMM scheme and our scheme have approximatively the same proven security. And the comparison of the MMM scheme with our scheme is roughly the same as the comparison in [19] between the IR scheme and the MMM scheme (which did not take into account the tightness of the reduction).

Very roughly, we can say that the MMM key generation and key update algorithms are faster (about T times faster). However, MMM private keys are longer. And, even if MMM public keys are shorter (more than 30 times for $k = 80, \ell_N = 1248$), in most cases, it is not really useful since signatures with the MMM scheme are about four times longer than signatures with our scheme ($4\ell_N + (\log(k) + \log T)k$ compared to $\ell_N + k$), and also about twice as long as the sum of the length of a public key of our scheme and a signature. Therefore, since the public key is used for verification, the total memory needed to store input data needed for the verification of a signature with the MMM scheme is still twice the amount of the one needed with our scheme. Furthermore, our scheme outperforms the MMM scheme with respect to verification time (considering Table 2, since the MMM verification algorithm verifies two classical GQ signatures). This means that, if verification time, signing time, and signature size are critical (for example, if verification or signing has to be performed on a smartcard), our scheme is better than the MMM scheme. And, even more generally, if key updates are not performed often and if T can be bounded by a reasonable constant (for example, if keys are updated each day and are expected to last 3 years, $T = 2^{10}$, and key update time is not really a problem), our scheme is also better than the MMM scheme.

6 Generic Factoring-Based Forward-Secure Signature Scheme

In this section, we show that all our previous results on the GQ scheme and its forward-secure extension can be generalized and applied to several other schemes. To do so, we first introduce a new generic factoring-based key-evolving lossy identification scheme and then show that several factoring-based signature and forward-secure signature schemes can be seen as simple instantiations of this generic scheme.

6.1 Generic Factoring-Based Forward-Secure Signature Scheme

Let ℓ be a security parameter, let N be a product of large primes, and let $e_1, \ldots, e_T$ be T integers and E be the least common multiple of $e_1, \ldots, e_T$. Let $S_1, \ldots, S_\ell$ be a set of elements in ${\mathbb{Z}_N^*}^\ell$ and let $U_1, \ldots, U_\ell \in {\mathbb{Z}_N^*}^\ell$ be the set of elements containing the corresponding E-powers. That is, for each $j \in \{1, \ldots, \ell\}$, $U_j = S_j^E \bmod N$. The public key is $pk = (N, e_1, \ldots, e_T, U_1, \ldots, U_\ell)$ (as for our variant of the IR scheme, we can use a random oracle to avoid storing the exponents in the keys, as explained in Section 5.1). Let f_i be the least common

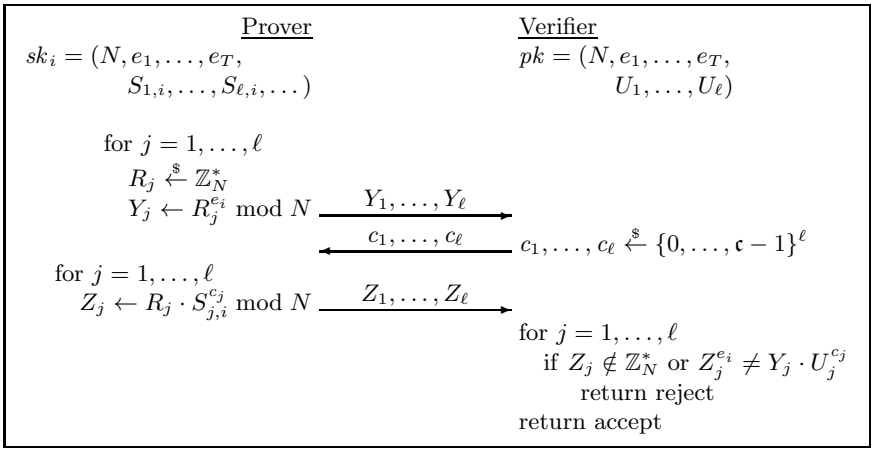


Fig. 3. Description of the generic identification scheme $I\mathcal{D}$ for proving that the elements $U_1, \ldots, U_\ell$ in pk are all e_i-residues (for each $j \in \{1, \ldots, \ell\}$, $U_j = S_{j,i}^e \bmod N$)

multiple of $e_{i+1}, \ldots, e_T$ for each $i \in \{1, \ldots, T\}$ ($f_T = 1$) and let $S_{j,i} = S_j^{E/e_i}$ and $S'_{j,i} = S_j^{E/f_i}$, for each $1 \leq i \leq T$ and each $1 \leq j \leq \ell$. Then, the secret key for period $1 \leq i \leq T$ is $sk_i = (i, N, e_i, \ldots, e_T, S_{1,i}, \ldots, S_{\ell,i}, S'_{1,i}, \ldots, S'_{\ell,i})$. We remark that it is possible to compute sk_{i+1} from sk_i by computing: $S_{j,i+1} = S_{j,i}^{\prime f_i/e_{i+1}} \bmod N$ and $S'_{j,i+1} = S_{j,i}^{\prime f_i/f_{i+1}} \bmod N$.

The identification scheme is depicted in Fig. 3 and is a straightforward extension of the one of our variant of the IR scheme in Section 4.2. For period i, its goal is to prove that the elements $U_1, \ldots, U_\ell$ are all e_i-residues, and works as follows. First, the prover chooses an element $R_j \in \mathbb{Z}_N^*$ and computes $Y_j \leftarrow R_j^{e_i} \bmod N$, for $j \in \{1, \ldots, \ell\}$. It then sends $Y_1, \ldots, Y_\ell$ to the verifier, which in turn chooses $c_1, \ldots, c_\ell \in \{0, \ldots, \mathfrak{c}-1\}^\ell$ and returns it to the prover. Upon receiving $c_1, \ldots, c_\ell$, the prover computes $Z_j \leftarrow R_j \cdot S_{j,i}^{c_j} \bmod N$ for $j \in \{1, \ldots, \ell\}$ and sends these values to the verifier. Finally, the verifier checks whether $Z_j \in \mathbb{Z}_N^*$ and $Z_j^{e_i} = Y_j \cdot U_j^{c_j}$ for $j \in \{1, \ldots, \ell\}$ and accepts only if this is the case. The corresponding factoring-based forward-secure signature scheme is depicted in Fig. 4.

In the full version [2], we prove that the previous scheme is existentially forward-secure, under the following condition:

Condition 3. *There exists a normal key generation algorithm* KG *and a lossy key generation algorithm* LKG *which takes as input the security parameter and the period i and outputs a pair (pk, sk'_{i+1}) such that, for every $i \in \{1, \ldots, T\}$:*

- *(pk, sk'_{i+1}) is indistinguishable from a pair (pk, sk_{i+1}) generated by* KG *and i calls to* Update *(to get sk_{i+1} from sk_1);*
- *for all $c \in \{0, \ldots, \mathfrak{c}-1\}$, none of $U_1, \ldots, U_\ell$ is a $e'(e, c, N)$-residue, where $e'(e, c, N)$ is:*

$$e'(e, c, N) = \operatorname*{gcd}_{i \in \{1, \ldots, m\}} \frac{e \wedge (p_i^{k_i} - p_i^{k_i - 1})}{c \wedge e \wedge (p_i^{k_i} - p_i^{k_i - 1})} e'_i,$$

$\underline{\mathsf{KG}(1^k, 1^T)}$
Generate $N, e_1, \ldots, e_T$
$E \leftarrow \mathrm{lcm}(e_1, \ldots, e_T)$
for $i = 1, \ldots, T$
  $f_i \leftarrow \mathrm{lcm}(e_{i+1}, \ldots, e_T)$
for $j = 1, \ldots, \ell$
  $S_j \xleftarrow{\$} \mathbb{Z}_N^*$
  $S_{j,1} \leftarrow S_j^{E/e_1} \bmod N$
  $S'_{j,1} \leftarrow S_j^{E/f_1} \bmod N$
  $U_j \leftarrow S_j^E \bmod N$
$pk \leftarrow (N, e_1, \ldots, e_T, U_1, \ldots, U_\ell)$
$sk_1 \leftarrow (1, N, e_1, \ldots, e_T, S_{1,1}, \ldots, S_{\ell,1}, S'_{1,1}, \ldots, S'_{\ell,1})$
return (pk, sk_1)

$\underline{\mathsf{Update}(sk, M)}$
$(i, N, e_1, \ldots, e_T, S_{1,i}, \ldots, S_{\ell,i}, S'_{1,i}, \ldots, S'_{\ell,i}) \leftarrow sk$
if $i = T$ then
  return $\perp$
$f_i \leftarrow \mathrm{lcm}(e_{i+1}, \ldots, e_T)$
$f_{i+1} \leftarrow \mathrm{lcm}(e_{i+2}, \ldots, e_T)$
for $j = 1, \ldots, \ell$
  $S_{j,i+1} \leftarrow S_{j,i}^{\prime f_i/e_{i+1}}$
  $S'_{j,i+1} \leftarrow S_{j,i}^{\prime f_i/f_{i+1}}$
$sk_{i+1} \leftarrow (i+1, N, e_{i+1}, \ldots, e_T, S_{1,i+1}, \ldots, S_{\ell,i+1}, S'_{1,i+1}, \ldots, S'_{\ell,i+1})$
return sk_{i+1}

$\underline{\mathsf{Ver}(pk, \langle \sigma, i \rangle, M)}$
$(N, e_1, \ldots, e_T, U1, \ldots, U_\ell) \leftarrow pk$
$((Y_1, \ldots, Y_\ell), (Z_1, \ldots, Z_\ell)) \leftarrow \sigma$
$(c_1, \ldots, c_\ell) \leftarrow \mathsf{H}(\langle (Y_1, \ldots, Y_\ell), M, i \rangle)$
for $j = 1, \ldots, \ell$
  if $Z_j \notin \mathbb{Z}_N^*$ or $Z_j^{e_i} \neq Y_j \cdot U_j^{c_j}$ then
    return reject
return accept

$\underline{\mathsf{Sign}(sk, M)}$
$(i, N, e_i, \ldots, e_T, S_{1,i}, \ldots, S_{\ell,i}, S'_{1,i}, \ldots, S'_{\ell,i}) \leftarrow sk$
for $j = 1, \ldots, \ell$
  $R_j \xleftarrow{\$} \mathbb{Z}_N^*$
  $Y_j \leftarrow R_j^{e_i} \bmod N$
$(c_1, \ldots, c_\ell) \leftarrow \mathsf{H}(\langle (Y_1, \ldots, Y_\ell), M, i \rangle)$
for $j = 1, \ldots, \ell$
  $Z_j \leftarrow R_j \cdot S_j^{c_j} \bmod N$
$\sigma \leftarrow ((Y_1, \ldots, Y_\ell), (Z_1, \ldots, Z_\ell))$
return $\langle \sigma, i \rangle$

Fig. 4. Factoring-based forward-secure signature scheme

with $N = p_1^{k_1} \ldots p_m^{k_m}$ the prime decomposition of N and e'_i the greatest divisor of e coprime with $p_i^{k_i} - p_i^{k_i - 1}$, and where $a \wedge b$ is the greatest common divisor (gcd) of a and b.

The second part of the condition ensures that the scheme is $1/\mathfrak{c}^\ell$-lossy.

6.2 Some Instantiations

In addition to the GQ scheme and our variant of the IR scheme, there are other possible instantiations of our generic scheme.

Quadratic-Residuosity-Based Signature Scheme. The case where $e = \mathfrak{c} = 2$ and $T = 1$ is an important instantiation of the generic scheme as it coincides with the quadratic-residuosity-based scheme informally suggested by Katz and Wang in [17]. This scheme is existentially unforgeable based on the

hardness of the quadratic-residuosity problem as long as ℓ is large enough to make the term $q_h/2^\ell$ negligible.

2^t-Root Signature Scheme by Ong and Schnorr. The case where $e = \mathfrak{c} = 2^t$, $\ell = 1$, and $T = 1$ coincides with the 2^t-root identification scheme by Ong and Schnorr [25]. If $N = p_1 p_2$ is an RSA modulus such that 2^t divides $p_1 - 1$ and $p_2 - 1$, this scheme is existentially unforgeable based on the hardness of the strong-2^t-residuosity problem as long as t is large enough to make the term $q_h/2^t$ negligible.

Paillier Signature Scheme. The case where $\ell = 1$, $T = 1$, and $e = p_1 p_2$ is an RSA modulus, $N = e^2 = p_1^2 p_2^2$ and $\mathfrak{c} \leq \min(p_1, p_2)$ coincides with the Paillier signature scheme [26]. This scheme is existentially unforgeable based on the hardness of the high-residuosity problem of [26].

2^t-Root Forward-Secure Signature Scheme. The case in which $e_i = 2^{t(T-i+1)}$ with t a positive integer and $\mathfrak{c} = 2^i$ is a generalization of the quadratic-residuosity-based scheme and the 2^t-root scheme. In this case, $f_i = e_i$, and we do not need to store $S'_{1,i}$. If $N = p_1 p_2$ is an RSA modulus such that 2^{tT} divides $p_1 - 1$ and $p_2 - 1$, this scheme is existentially forward-secure based on the hardness of a variant of the strong-2^{tT}-assumption, as long as the exponents t and ℓ are large enough to make the term $q_h/2^{t\ell}$ negligible. Although this scheme appears to be new, it is of limited interest as its public key and secret key sizes are linear in the number T of time periods.

Proof details for the above instantiations can be found in the full version [2].

Acknowledgments. We would like to thank Mihir Bellare and Eike Kiltz for their helpful comments on a preliminary version of this paper and the anonymous referees of PKC 2013 for their valuable input.

This work was supported in part by the French ANR-10-SEGI-015 PRINCE Project and in part by the European Commission through the FP7-ICT-2011-EU-Brazil Program under Contract 288349 SecFuNet and the ICT Program under Contract ICT-2007-216676 ECRYPT II.

References

1. Abdalla, M., An, J.H., Bellare, M., Namprempre, C.: From Identification to Signatures via the Fiat-Shamir Transform: Minimizing Assumptions for Security and Forward-Security. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 418–433. Springer, Heidelberg (2002)
2. Abdalla, M., Ben Hamouda, F., Pointcheval, D.: Tighter Reductions for Forward-Secure Signature Schemes. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 292–311. Springer, Heidelberg (2013), full version available from the webpage of the authors
3. Abdalla, M., Fouque, P.-A., Lyubashevsky, V., Tibouchi, M.: Tightly-Secure Signatures from Lossy Identification Schemes. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 572–590. Springer, Heidelberg (2012)

4. Bellare, M., Miner, S.K.: A Forward-Secure Digital Signature Scheme. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 431–448. Springer, Heidelberg (1999)
5. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Ashby, V. (ed.) ACM CCS 1993, pp. 62–73. ACM Press (November 1993)
6. Cachin, C., Micali, S., Stadler, M.: Computationally Private Information Retrieval with Polylogarithmic Communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999)
7. ECRYPT II yearly report on algorithms and keysizes (2011)
8. Feige, U., Fiat, A., Shamir, A.: Zero-knowledge proofs of identity. Journal of Cryptology 1(2), 77–94 (1988)
9. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
10. Garg, S., Bhaskar, R., Lokam, S.V.: Improved Bounds on Security Reductions for Discrete Log Based Signatures. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 93–107. Springer, Heidelberg (2008)
11. Goldwasser, S., Micali, S., Rivest, R.L.: A "Paradoxical" Solution to the Signature Problem. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, p. 467. Springer, Heidelberg (1985)
12. Guillou, L.C., Quisquater, J.-J.: A "Paradoxical" Identity-Based Signature Scheme Resulting from Zero-Knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 216–231. Springer, Heidelberg (1990)
13. Hohenberger, S., Waters, B.: Short and Stateless Signatures from the RSA Assumption. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 654–670. Springer, Heidelberg (2009)
14. Impagliazzo, R., Naor, M.: Efficient cryptographic schemes provably as secure as subset sum. Journal of Cryptology 9(4), 199–216 (1996)
15. Itkis, G., Reyzin, L.: Forward-Secure Signatures with Optimal Signing and Verifying. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 332–354. Springer, Heidelberg (2001)
16. Kakvi, S.A., Kiltz, E.: Optimal Security Proofs for Full Domain Hash, Revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 537–553. Springer, Heidelberg (2012)
17. Katz, J., Wang, N.: Efficiency improvements for signature schemes with tight security reductions. In: Jajodia, S., Atluri, V., Jaeger, T. (eds.) ACM CCS 2003, pp. 155–164. ACM Press (October 2003)
18. Lyubashevsky, V., Micciancio, D.: Generalized Compact Knapsacks Are Collision Resistant. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006, Part II. LNCS, vol. 4052, pp. 144–155. Springer, Heidelberg (2006)
19. Malkin, T., Micciancio, D., Miner, S.K.: Efficient Generic Forward-Secure Signatures with an Unbounded Number of Time Periods. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 400–417. Springer, Heidelberg (2002)
20. Micali, S.: A secure and efficient digital signature algorithm. Technical Memo MIT/LCS/TM-501b, Massachusetts Institute of Technology, Laboratory for Computer Science (April 1994)
21. Micali, S., Reyzin, L.: Improving the Exact Security of Fiat-Shamir Signature Schemes. In: Baumgart, R. (ed.) CQRE 1999. LNCS, vol. 1740, pp. 167–182. Springer, Heidelberg (1999)
22. Micali, S., Reyzin, L.: Improving the exact security of digital signature schemes. Journal of Cryptology 15(1), 1–18 (2002), full version of [21]

23. Micciancio, D., Mol, P.: Pseudorandom Knapsacks and the Sample Complexity of LWE Search-to-Decision Reductions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 465–484. Springer, Heidelberg (2011)
24. Ohta, K., Okamoto, T.: A Modification of the Fiat-Shamir Scheme. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 232–243. Springer, Heidelberg (1990)
25. Ong, H., Schnorr, C.-P.: Fast Signature Generation with a Fiat Shamir-like Scheme. In: Damgård, I.B. (ed.) EUROCRYPT 1990. LNCS, vol. 473, pp. 432–440. Springer, Heidelberg (1991)
26. 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)
27. Paillier, P., Vergnaud, D.: Discrete-Log-Based Signatures May Not Be Equivalent to Discrete Log. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 1–20. Springer, Heidelberg (2005)
28. Patel, S., Sundaram, G.S.: An Efficient Discrete Log Pseudo Random Generator. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 304–317. Springer, Heidelberg (1998)
29. Peikert, C., Rosen, A.: Efficient Collision-Resistant Hashing from Worst-Case Assumptions on Cyclic Lattices. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 145–166. Springer, Heidelberg (2006)
30. Schnorr, C.-P.: Efficient Identification and Signatures for Smart Cards. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 688–689. Springer, Heidelberg (1990)
31. Seurin, Y.: On the Exact Security of Schnorr-Type Signatures in the Random Oracle Model. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 554–571. Springer, Heidelberg (2012)
32. van Oorschot, P.C., Wiener, M.J.: On Diffie-Hellman Key Agreement with Short Exponents. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 332–343. Springer, Heidelberg (1996)

Tagged One-Time Signatures: Tight Security and Optimal Tag Size

Masayuki Abe[1], Bernardo David[2], Markulf Kohlweiss[3], Ryo Nishimaki[1], and Miyako Ohkubo[4]

[1] NTT Secure Platform Laboratories
{abe.masayuki,nishimaki.ryo}@lab.ntt.co.jp
[2] University of Brasilia
bernardo.david@aluno.unb.br
[3] Microsoft Research, Cambridge
markulf@microsoft.com
[4] Security Architecture Laboratory, NSRI, NICT
m.ohkubo@nict.go.jp

Abstract. We present an efficient structure-preserving tagged one-time signature scheme with tight security reductions to the decision-linear assumption. Our scheme features short tags consisting of a single group element and gives rise to the currently most efficient structure-preserving signature scheme based on the decision-liner assumption with constant-size signatures of only 14 group elements, where the record-so-far was 17 elements.

To demonstrate the advantages of our scheme, we revisit the work by Hofheinz and Jager (CRYPTO 2012) and present the currently most efficient tightly secure public-key encryption scheme. We also obtain the first structure-preserving public-key encryption scheme featuring both tight security and public verifiability.

Keywords: Tagged One-Time Signatures, Structure-Preserving Signatures, Tight Security Reduction, Decision Linear Assumption.

1 Introduction

Background. A tagged one-time signature (TOS, [1]) scheme is a signature scheme that includes a fresh random tag in each signature. It is unforgeable if creating a signature on a new message but with an old tag picked by an honest signer is hard. A TOS is a special type of partial one-time signature (POS, [1]), that involves one-time keys and long-term keys.[1] Namely, a TOS is a POS with an empty one-time secret-key. For this reason the one-time public-key is called a tag.

A TOS is structure-preserving [2] if its long-term public-keys, tags, messages, and signatures consist only of elements of the base bilinear groups and the verification only evaluates pairing product equations. Structure-preservation grants interoperability among building blocks over the same bilinear groups and allows modular constructions of conceptually complex cryptographic schemes, in particular when combined

[1] POS also known as two-tier signatures [8].

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 312–331, 2013.

with the Groth-Sahai (GS) proof system [25]. So far, structure-preserving constructions have been developed for signature [23,14,2,3,15,11,1] commitments [4,5], and public-key encryption schemes [12]. The growing list of their applications include universally composable adaptive oblivious transfer [22,21], anonymous proxy signatures [18], delegatable anonymous credentials [7], transferable e-cash [19], compact verifiable shuffles [16], and network coding [6].

Efficiency and tight security is of general interest for cryptographic primitives. In [26], a tightly-secure structure-preserving POS is used as a central building block for constructing pubic-key encryption scheme secure against adaptive chosen-ciphertext attacks with multiple challenges and users. Replacing the POS with a TOS gives an immediate improvement. It is however seemingly more difficult to construct a TOS with high efficiency and tight security at the same time due to the absence of one-time secrets. To the best of our knowledge, the scheme in [1] is the only structure-preserving TOS in the literature bases on the decision-linear assumption (DLIN, [9]). Unfortunately, their reduction is not tight. For q_s signing queries, it suffers a factor of $1/q_s$. Moreover, a tag requires two group elements for technical reasons. This contrasts to the case of POS, where tight reductions to DLIN or SXDH are known, and the one-time public-key can be a single group element [1].

Our Contribution. The main contribution of this paper is a structure-preserving TOS with 1) optimally short tags consisting only of one group element, and 2) a tight security reduction to a computational assumption tightly implied by DLIN. Thus, when compared with the TOS scheme in [1], our scheme improves both tag size and tightness. The first application of our new TOS is a more efficient structure-preserving signature (SPS) scheme based on DLIN. The signature consists of 14 group elements and the verification evaluates 7 pairing product equations. It saves 3 group elements and 2 equations over previous SPS in [1]. Our second application is a more efficient tightly secure public-key encryption scheme. As a stepping stone we also obtain a more efficient and tight secure structure-preserving tree-based signature schemes. We obtain these results by revisiting the framework of [26]. In addition to the efficiency and key-management improvements, our contributions include the first structure-preserving CCA-secure encryption schemes featuring a tight security reduction, public verifiability, and leakage resilience which we inherit from [17].

The combined length of a tag and the long-term public-key in the new TOS is shorter than the one-time public-key of other structure-preserving one-time signature schemes (OTS) in the literature. (It saves 2 group elements over the OTS in [4].) Using our TOS as OTS is therefore beneficial even for applications that use the whole public-key only once. Typical examples include the IBE-to-PKE transformation [13], NIZK-to-SS-NIZK transformation [23], CCA-secure Group Signatures [24] where one-time signatures are used to add non-malleability, and delegatable anonymous credentials [7]. Though the improvement in this direction is small, it is often considerably amplified in applications, e.g., in delegatable anonymous credentials where the whole public key needs to be concealed in Groth-Sahai commitments.

Many of the applications in this paper require hundreds of group elements and are not necessarily practical. Nevertheless, the concrete efficiency assessment should serve as a reference that shows how efficient instantiation of generic modular constructions

can be. In particular, as the constants in generic constructions can be large, we observed that small gains in the building blocks can result in significant efficiency improvements in applications.

2 Preliminaries

2.1 Bilinear Groups

We work in a setting with a symmetric bilinear pairing (the Type-I setting of [20]) and use multiplicative notation. Let $\mathcal{G}$ be a bilinear group generator that takes security parameter λ as input and outputs a description of bilinear groups $\Lambda := (p, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of prime order p, and e is an efficient and non-degenerating bilinear map $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We count the number of group elements to measure the size of cryptographic objects such as keys, messages, and signatures. By $\mathbb{Z}_p$ and $\mathbb{Z}_p^*$, we denote $\mathbb{Z}/p\mathbb{Z}$ and $\mathbb{Z}/p\mathbb{Z} \setminus \{0\}$, respectively. We abuse the notation and denote $\mathbb{G} \setminus \{1_{\mathbb{G}}\}$ by $\mathbb{G}^*$.

The security of our schemes is based on the following computational assumption.

Definition 1 (Simultaneous Double Pairing Assumption : SDP [14]). *For the bilinear group generator $\mathcal{G}$ and any polynomial time $\mathcal{A}$ the probability*

$$\mathrm{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{ll} \Lambda \leftarrow \mathcal{G}(1^\lambda) & Z \in \mathbb{G}^* \wedge \\ (G_z, G_r, H_z, H_s) \leftarrow \mathbb{G}^{*4} & : 1 = e(G_z, Z)\, e(G_r, R) \wedge \\ (Z, R, S) \leftarrow \mathcal{A}(\Lambda, G_z, G_r, H_z, H_s) & 1 = e(H_z, Z)\, e(H_s, S) \end{array}\right]$$

is negligible in λ.

SDP is random-self reducible. Given (G_z, G_r, H_z, H_s), another random instance $(G_z^a G_r^b, G_r^c, H_z^a H_s^d, H_s^e)$ can be generated by choosing a, b, c, d, and e uniformly from $\mathbb{Z}_p^*$. Given an answer (Z, R, S) to the new instance, $(Z^a, R^c Z^b, S^e Z^d)$ is the answer to the original instance. Furthermore, SDP is tightly reduced from DLIN as observed in [14]. For a DLIN instance $(G_1, G_2, G_3, G_1^a, G_2^b, G_3^c)$ for deciding $c = a+b$ or not, construct an SDP instance (G_1^a, G_1, G_2^b, G_2). Then, given an answer (Z, R, S) that satisfies $1 = e(G_1^a, Z)\ e(G_1, R)$ and $1 = e(G_2^b, Z)\ e(G_2, S)$, one can conclude that $c = a + b$ if $e(G_3, R \cdot S) = e(G_3^c, Z)$ since $R = Z^a$ and $S = Z^b$. We restate this observation as a lemma below.

Lemma 1 (DLIN $\Rightarrow$ SDP). *If there exists adversary $\mathcal{A}$ that solves SDP, then there exists adversary $\mathcal{B}$ that solves DLIN with the same advantage and a runtime overhead of a few exponentiations and pairings.*

2.2 Syntax and Security Notions

We follow the syntax and security notions for TOS in [1]. Let $\mathsf{Setup}(1^\lambda)$ be an algorithm that takes security parameter λ and outputs common parameter gk. Parameters gk are (sometimes implicit) input to all algorithms.

Definition 2 (Tagged One-Time Signature Scheme). *A tagged one-time signature scheme* TOS *is a set of polynomial-time algorithms* TOS.{Key, Tag, Sign, Vrf} *that takes* gk *generated by* Setup. *Each function works as follows.*

- $\mathsf{TOS.Key}(gk)$ *generates a long-term public-key* pk *and a secret-key* sk. *Message space* $\mathcal{M}_t$ *and tag space* $\mathcal{T}$ *are determined by* gk.
- $\mathsf{TOS.Tag}(gk)$ *takes* gk *as input and outputs* $tag \in \mathcal{T}$.
- $\mathsf{TOS.Sign}(sk, msg, tag)$ *outputs signature* σ *for message* msg *based on secret-key* sk *and tag* tag.
- $\mathsf{TOS.Vrf}(pk, tag, msg, \sigma)$ *outputs* 1 *for acceptance, or* 0 *for rejection.*

For any key $(pk, sk) \leftarrow \mathsf{TOS.Key}(\mathsf{Setup}(1^\lambda))$, *any message* $msg \in \mathcal{M}_t$, *any tag* $tag \leftarrow \mathsf{TOS.Tag}(gk)$, *and any signature* $\sigma \leftarrow \mathsf{TOS.Sign}(sk, msg, tag)$, *verification* $\mathsf{TOS.Vrf}(pk, tag, msg, \sigma)$ *outputs* 1.

TOS is called uniform-tag if the output distribution of tag is uniform over $\mathcal{T}$. TOS is structure-preserving over Λ if gk contains Λ and the public-keys, messages, tags, and signatures consist only of elements of base groups of Λ and TOS.Vrf consists of evaluating pairing product equations.

Definition 3 (Unforgeability against One-Time Tag Chosen-Message Attacks). *For tagged one-time signature scheme* TOS *and algorithm* $\mathcal{A}$, *let* $\mathrm{Expr}^{\mathsf{ot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}}$ *be an experiment that:*

$$
\begin{aligned}
&\mathrm{Expr}^{\mathsf{ot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}}(1^\lambda) := \\
&\quad gk \leftarrow \mathsf{Setup}(1^\lambda),\ (pk, sk) \leftarrow \mathsf{TOS.Key}(gk) \\
&\quad (tag^\dagger, \sigma^\dagger, msg^\dagger) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{tag}}, \mathcal{O}_{\mathsf{sig}}}(pk) \\
&\quad \textit{If } \exists (tag, msg, \sigma) \in Q_m \textit{ s.t.} \\
&\qquad tag^\dagger = tag \ \wedge\ msg^\dagger \neq msg \ \wedge\ 1 = \mathsf{TOS.Vrf}(pk, tag^\dagger, \sigma^\dagger, msg^\dagger) \\
&\quad \textit{return 1. Return 0, otherwise.}
\end{aligned}
$$

$\mathcal{O}_{\mathsf{tag}}$ *and* $\mathcal{O}_{\mathsf{sig}}$ *are tag and signature generation oracles, respectively. On receiving* i*-th query,* $\mathcal{O}_{\mathsf{tag}}$ *returns tag* tag_i *generated by* $\mathsf{TOS.Tag}(gk)$. *On receiving* j*-th query with message* msg_j *as input (if at this point* $\mathcal{O}_{\mathsf{tag}}$ *has been received* $i < j$ *requests,* $\mathcal{O}_{\mathsf{tag}}$ *is invoked to generate* tag_j*),* $\mathcal{O}_{\mathsf{sig}}$ *performs* $\sigma_j \leftarrow \mathsf{TOS.Sign}(sk, msg_j, tag_j)$, *appends* (tag_j, msg_j, σ_j) *to* Q_m, *and returns* σ_j *(and* tag_j *if generated) to* $\mathcal{A}$.

A tagged one-time signature scheme is unforgeable against one-time tag adaptive chosen message attacks (OT-CMA) if for all polynomial-time oracle algorithms $\mathcal{A}$ *the advantage function* $\mathrm{Adv}^{\mathsf{ot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}} := \Pr[\mathrm{Expr}^{\mathsf{ot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}}(1^\lambda) = 1]$ *is negligible in* λ.

Strong unforgeability is a variation on this definition obtained by replacing the condition $msg^\dagger \neq msg$ in the experiment with $(msg^\dagger, \sigma^\dagger) \neq (msg, \sigma)$. Another variation is non-adaptive attack unforgeability (OT-NACMA) defined by integrating $\mathcal{O}_{\mathsf{tag}}$ into $\mathcal{O}_{\mathsf{sig}}$ so that tag_j and σ_j are returned to $\mathcal{A}$ at the same time. Namely, $\mathcal{A}$ must submit msg_j before seeing tag_j. It is obvious that if a scheme is secure in the sense of OT-CMA,

the scheme is also secure in the sense of OT-NACMA. By $\mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{TOS},\mathcal{A}}(\lambda)$ we denote the advantage of $\mathcal{A}$ in this non-adaptive case. We use labels sot-cma and sot-nacma for adaptive and non-adaptive strong unforgeability respectively.

For signatures we follow the standard syntax of digital signatures with common setup. Namely, a signature scheme consists of three algorithms $\mathsf{SIG}.\{\mathsf{Key}, \mathsf{Sign}, \mathsf{Vrf}\}$ that take gk generated by Setup as additional input. SIG.Key is a key generation algorithm, SIG.Sign is a signing algorithm and SIG.Vrf is a verification algorithm. We also follow the standard security notion of existential unforgeability against adaptive chosen message attacks.

2.3 A Framework of TOS + RMA-SIG

We review the framework of combining TOS and RMA signatures in [1] to obtain a CMA-secure signature scheme. Let rSIG be a signature scheme with message space $\mathcal{M}_{\mathsf{r}}$, and TOS be a tagged one-time signature scheme with tag space $\mathcal{T}$ such that $\mathcal{M}_{\mathsf{r}} = \mathcal{T}$. We construct a signature scheme SIG from rSIG and TOS. Let gk be a global parameter generated by $\mathsf{Setup}(1^\lambda)$.

[Generic Construction: SIG**]**

$\mathsf{SIG.Key}(gk)$: Run $(pk_t, sk_t) \leftarrow \mathsf{TOS.Key}(gk)$, $(vk_r, sk_r) \leftarrow \mathsf{rSIG.Key}(gk)$. Output $vk := (pk_t, vk_r)$ and $sk := (sk_t, sk_r)$.

$\mathsf{SIG.Sign}(sk, msg)$: Parse sk into (sk_t, sk_r). Output $\sigma := (tag, \sigma_t, \sigma_r)$ where $tag \leftarrow \mathsf{TOS.Tag}(gk)$, $\sigma_t \leftarrow \mathsf{TOS.Sign}(sk_t, msg, tag)$, and $\sigma_r \leftarrow \mathsf{rSIG.Sign}(sk_r, tag)$.

$\mathsf{SIG.Vrf}(vk, \sigma, msg)$: Parse vk and σ accordingly. Output 1, if $1 = \mathsf{TOS.Vrf}(pk_t, tag, \sigma_t, msg)$ and $1 = \mathsf{rSIG.Vrf}(vk_r, \sigma_r, tag)$. Output 0, otherwise.

The following theorems are due to [1].

Theorem 1. SIG *is unforgeable against adaptive chosen message attacks (UF-CMA) if* TOS *is uniform-tag and unforgeable against one-time non-adaptive chosen message attacks (OT-NACMA), and* rSIG *is unforgeable against random message attacks (UF-RMA). In particular,* $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{TOS},\mathcal{B}}(\lambda) + \mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{rSIG},\mathcal{C}}(\lambda)$. *The overhead of adversary* $\mathcal{B}$ *against* rSIG *and* $\mathcal{C}$ *against* TOS *is proportional to the running time of the key generation and signing operations of* rSIG *and* TOS *respectively.*

Theorem 2. *If* TOS.Tag *produces constant-size tags and signatures in the size of input messages, the resulting* SIG *produces constant-size signatures as well. Furthermore, if* TOS *and* rSIG *are structure-preserving, so is* SIG.

3 Tightly-Secure TOS Based on DLIN

Let gk be a global parameter that specifies $\Lambda = (p, \mathbb{G}, \mathbb{G}_T, e)$ generated by group generator $\mathcal{G}(1^\lambda)$. It also includes a generator $G \in \mathbb{G}^*$. We construct $\mathsf{TOS}.\{\mathsf{Key}, \mathsf{Tag}, \mathsf{Sign}, \mathsf{Vrf}\}$ as shown in Fig. 1.

[Scheme TOS]

$\mathsf{TOS.Key}(gk)$: Parse $gk = (\Lambda, G)$. Choose $w_z, w_r, \mu_z, \mu_s, \tau$ randomly from $\mathbb{Z}_p^*$ and compute $G_z := G^{w_z}$, $G_r := G^{w_r}$, $H_z := G^{\mu_z}$, $H_s := G^{\mu_s}$, $G_t := G^{\tau}$ and For $i = 1, \ldots, k$, uniformly choose $\chi_i, \gamma_i, \delta_i$ from $\mathbb{Z}_p$ and compute

$$G_i := G_z^{\chi_i} G_r^{\gamma_i}, \quad \text{and} \quad H_i := H_z^{\chi_i} H_s^{\delta_i}. \tag{1}$$

Output $pk := (G_z, G_r, H_z, H_s, G_t, G_1, \ldots, G_k, H_1, \ldots, H_k) \in \mathbb{G}^{2k+5}$ and $sk := (\chi_1, \gamma_1, \delta_1, \ldots, \chi_k, \gamma_k, \delta_k, w_z, w_r, \mu_z, \mu_s, \tau) \in \mathbb{Z}_p^{3k+5}$.

$\mathsf{TOS.Tag}(gk)$: Choose $t \leftarrow \mathbb{Z}_p^*$ and output $tag := T = G^t \in \mathbb{G}$.

$\mathsf{TOS.Sign}(sk, msg, tag)$: Parse msg into $(M_1, \ldots, M_k) \in \mathbb{G}^k$. Take T_1 from tag. Parse sk accordingly. Output $\sigma := (Z, R, S) \in \mathbb{G}^3$ that, for $\zeta \leftarrow \mathbb{Z}_p$,

$$Z := G^{\zeta} \prod_{i=1}^{k} M_i^{-\chi_i}, \quad R := (T^{\tau} G_z^{-\zeta})^{\frac{1}{w_r}} \prod_{i=1}^{k} M_i^{-\gamma_i}, \text{ and}$$
$$S := (H_z^{-\zeta})^{\frac{1}{\mu_s}} \prod_{i=1}^{k} M_i^{-\delta_i}.$$

$\mathsf{TOS.Vrf}(pk, tag, msg, \sigma)$: Parse σ as $(Z, R, S) \in \mathbb{G}^3$, msg as $(M_1, \ldots, M_k) \in \mathbb{G}^k$, and take T from tag. Return 1 if the following equations hold. Return 0, otherwise.

$$e(T, G_t) = e(G_z, Z)\, e(G_r, R) \prod_{i=1}^{k} e(G_i, M_i) \tag{2}$$

$$1 = e(H_z, Z)\, e(H_s, S) \prod_{i=1}^{k} e(H_i, M_i) \tag{3}$$

Fig. 1. Tagged One-Time Signature Scheme

Correctness is verified by inspecting the following relations.

$$\text{For (2): } e(G_z, G^{\zeta} \prod_{i=1}^{k} M_i^{-\chi_i})\, e(G_r, (T^{\tau} G_z^{-\zeta})^{\frac{1}{w_r}} \prod_{i=1}^{k} M_i^{-\gamma_i}) \prod_{i=1}^{k} e(G_z^{\chi_i} G_r^{\gamma_i}, M_i)$$
$$= e(G_z, G^{\zeta})\, e(G, T^{\tau}) e(G, G_z^{-\zeta}) = e(G, T^{\tau}) = e(T, G_t)$$

$$\text{For (3): } e(H_z, G^{\zeta} \prod_{i=1}^{k} M_i^{-\chi_i})\, e(H_s, (H_z^{-\zeta})^{\frac{1}{\mu_s}} \prod_{i=1}^{k} M_i^{-\delta_i}) \prod_{i=1}^{k} e(H_z^{\chi_i} H_s^{\delta_i}, M_i)$$
$$= e(H_z, G^{\zeta})\, e(G, H_z^{-\zeta}) = 1$$

We state the following theorems, of which the first one is immediate from the construction.

Theorem 3. *Above* TOS *is structure-preserving, and yields uniform tags and constant-size signatures.*

Theorem 4. *Above* TOS *is strongly unforgeable against one-time tag adaptive chosen message attacks (SOT-CMA) if the SDP assumption holds. In particular,* $\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}} \leq \mathrm{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{B}} + 1/p$ *and the runtime overhead of the reduction* $\mathcal{B}$ *is a small number of multi-exponentiations per signing query.*

Proof. Given successful forger $\mathcal{A}$ against TOS as a black-box, we construct $\mathcal{B}$ that breaks SDP as follows. Let $I_{\mathsf{sdp}} = (\Lambda, G_z, G_r, H_z, H_s)$ be an instance of SDP. Algorithm $\mathcal{B}$ simulates the attack game against TOS as follows. It first build $gk := (\Lambda, G, G)$ by choosing G randomly from $\mathbb{G}^*$. This yields a gk in the same distribution as produced by Setup. Next $\mathcal{B}$ simulates TOS.Key by taking (G_z, G_r, H_z, H_s) from I_{sdp} and computing $G_t := H_s^{\tau}$ for random τ in $\mathbb{Z}_p^*$. It then generates G_i and H_i according to (1). This perfectly simulates TOS.Key.

On receiving the j-th query to $\mathcal{O}_{\mathsf{tag}}$, algorithm $\mathcal{B}$ computes

$$T := (G_z^{\zeta} G_r^{\rho})^{\frac{1}{\tau}} \tag{4}$$

for $\zeta, \rho \leftarrow \mathbb{Z}_p^*$. If $T = 1$, $\mathcal{B}$ sets $Z^\star := H_s$, $S^\star := H_z^{-1}$, and $R^\star := (Z^\star)^{\rho/\zeta}$, outputs $(Z^\star, R^\star, S^\star)$ and stop. Otherwise, $\mathcal{B}$ stores (ζ, ρ) and returns $tag_j := T$ to $\mathcal{A}$.

On receiving signing query $msg_j = (M_1, \ldots, M_k)$, algorithm $\mathcal{B}$ takes ζ and ρ used for computing tag_j (if they are not yet defined, invoke the procedure for $\mathcal{O}_{\mathsf{tag}}$) and computes

$$Z := H_s^{\zeta} \prod_{i=1}^{k} M_i^{-\chi_i}, \quad R := H_s^{\rho} \prod_{i=1}^{k} M_i^{-\gamma_i}, \quad \text{and} \quad S := H_z^{-\zeta} \prod_{i=1}^{k} M_i^{-\delta_i}. \tag{5}$$

Then $\mathcal{B}$ returns $\sigma_j := (Z, R, S)$ to $\mathcal{A}$ and record (tag_j, σ_j, msg_j).

When $\mathcal{A}$ outputs a forgery $(tag^\dagger, \sigma^\dagger, msg^\dagger)$, algorithm $\mathcal{B}$ searches the records for (tag, σ, msg) such that $tag^\dagger = tag$ and $(msg^\dagger, \sigma^\dagger) \neq (msg, \sigma)$. If no such entry exists, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ computes

$$Z^\star := \frac{Z^\dagger}{Z} \prod_{i=1}^{k} \left(\frac{M_i^\dagger}{M_i}\right)^{\chi_i}, \quad R^\star := \frac{R^\dagger}{R} \prod_{i=1}^{k} \left(\frac{M_i^\dagger}{M_i}\right)^{\gamma_i}, \text{ and } \quad S^\star := \frac{S^\dagger}{S} \prod_{i=1}^{k} \left(\frac{M_i^\dagger}{M_i}\right)^{\delta_i}$$

where (Z, R, S), $(M_1, \ldots, M_k)$ and their dagger counterparts are taken from (σ, msg) and $(\sigma^\dagger, msg^\dagger)$, respectively. $\mathcal{B}$ finally outputs $(Z^\star, R^\star, S^\star)$ and stops. This completes the description of $\mathcal{B}$.

We claim that $\mathcal{B}$ solves the problem by itself or the view of $\mathcal{A}$ is perfectly simulated. The correctness of key generation has been already inspected. In the simulation of $\mathcal{O}_{\mathsf{tag}}$, there is a case of $T = 1$ that happens with probability $1/q$. If it happens, $\mathcal{B}$ outputs a correct answer to I_{sdp}, which is inspected by observing $G_z = G_r^{-\rho/\zeta}$, $Z^\star = H_s \neq 1$, $e(G_z, Z^\star)e(G_r, R^\star) = e(G_r^{-\rho/\zeta}, Z^\star)e(G_r, (Z^\star)^{\rho/\zeta}) = 1$ and $e(H_z, Z^\star)e(H_s, S^\star) = e(H_z, H_s)e(H_s, H_z^{-1}) = 1$. Otherwise, tag T uniformly distributes over $\mathbb{G}^*$ and the simulation is perfect.

Oracle $\mathcal{O}_{\mathsf{sig}}$ is simulated perfectly as well. Correctness of simulated $\sigma_j = (Z,\ R,\ S)$ can be verified by inspecting the following relations.

$$\begin{aligned}
\text{(Right-hand of (2))} &= e(G_z, H_s^{\zeta} \prod_{i=1}^{k} M_i^{-\chi_i})\ e(G_r, H_s^{\rho} \prod_{i=1}^{k} M_i^{-\gamma_i})\ \prod_{i=1}^{k} e(G_z^{\chi_i} G_r^{\gamma_i}, M_i) \\
&= e(G_z^{\zeta} G_r^{\rho}, H_s) = e((G_z^{\zeta} G_r^{\rho})^{\frac{1}{\tau}}, H_s^{\tau}) = e(T_1, G_t) \\
\text{(Right-hand of (3))} &= e(H_z, H_s^{\zeta} \prod_{i=1}^{k} M_i^{-\chi_i})\ e(H_s, H_z^{-\zeta} \prod_{i=1}^{k} M_i^{-\delta_i})\ \prod_{i=1}^{k} e(H_z^{\chi_i} H_s^{\delta_i}, M_i) \\
&= e(H_z, H_s^{\zeta})\ e(H_s, H_z^{-\zeta})\ = 1
\end{aligned}$$

Every Z distributes uniformly over $\mathbb{G}$ due to the uniform choice of ζ. Then R and S are uniquely determined by following the distribution of Z.

Accordingly, $\mathcal{A}$ outputs successful forgery with noticeable probability and $\mathcal{B}$ finds a corresponding record (tag, σ, msg). We show that output $(Z^{\star}, R^{\star}, S^{\star})$ from $\mathcal{B}$ is a valid solution to I_{sdp}. First, equation (2) is satisfied because

$$\begin{aligned}
1 &= e\left(G_z, \frac{Z^{\dagger}}{Z}\right)\ e\left(G_r, \frac{R^{\dagger}}{R}\right) \prod_{i=1}^{k} e\left(G_z^{\chi_i} G_r^{\gamma_i}, \frac{M_i^{\dagger}}{M_i}\right) \\
&= e\left(G_z, \frac{Z^{\dagger}}{Z} \prod_{i=1}^{k} \left(\frac{M_i^{\dagger}}{M_i}\right)^{\chi_i}\right)\ e\left(G_r, \frac{R^{\dagger}}{R} \prod_{i=1}^{k} \left(\frac{M_i^{\dagger}}{M_i}\right)^{\gamma_i}\right) \\
&= e\left(G_z, Z^{\star}\right)\ e\left(G_r, R^{\star}\right),
\end{aligned}$$

holds. Equation (3) is verified similarly.

It remains to prove that $Z^{\star} \neq 1$. Since $msg^{\dagger} \neq msg$, there exists $\ell \in \{1, \ldots, k\}$ such that $M_{\ell}^{\dagger}/M_{\ell} \neq 1$. We claim that, parameter $\chi_1, \ldots, \chi_k$ are independent of the view of $\mathcal{A}$. We prove it by showing that, for every possible assignment to $\chi_1, \ldots, \chi_k$, there exists an assignment to other coins, i.e., $(\gamma_1, \ldots, \gamma_k,\ \delta_1, \ldots, \delta_k)$ and $(\zeta^{(1)}, \rho^{(1)}, \ldots, \zeta^{(q_s)}, \rho^{(q_s)})$ for q_s queries, that is consistent to the view of $\mathcal{A}$. (By $\zeta^{(j)}$, we denote ζ with respect to the j-th query. We follow this convention hereafter. Without loss of generality, we assume that $\mathcal{A}$ makes q_s tag queries and the same number of signing queries.) Observe that the view of $\mathcal{A}$ consists of independent group elements $(G, G_z, G_r, H_z, H_s, G_t, G_1, H_1, \ldots, G_k, H_k)$ and $(T^{(j)}, Z^{(j)}, M_1^{(j)}, \ldots, M_k^{(j)})$ for $j = 1, \ldots, q_s$. (Note that $R^{(j)}$ and $S^{(j)}$ are not in the view since they are uniquely determined from other components.) We represent the view by the discrete-logarithms of these group elements with respect to base G. Namely, the view is $(1, w_z, w_r, \mu_z, \mu_s, \tau, w_1, \mu_1, \ldots, w_k, \mu_k)$ and $(t^{(j)}, z^{(j)}, m_1^{(j)}, \ldots, m_k^{(j)})$ for $j = 1, \ldots, q_s$. The view and the random coins follow relations from (1), (4), and (5) translated to

$$w_i = w_z \chi_i + w_r \gamma_i, \quad \mu_i = \mu_z \chi_i + \mu_s \delta_i \quad \text{for } i = 1, \ldots, k, \tag{6}$$

$$\tau t^{(j)} = w_z \zeta^{(j)} + w_r \rho^{(j)}, \text{ and} \tag{7}$$

$$z^{(j)} = \mu_s \, \zeta^{(j)} - \sum_{i=1}^{k} m_i^{(j)} \chi_i \quad \text{for } j = 1, \ldots, q_s. \tag{8}$$

Consider χ_ℓ for some $\ell \in \{1, \ldots, k\}$. For every value of χ_ℓ in $\mathbb{Z}_p$, the linear equations in (6) determine γ_ℓ and δ_ℓ. Then, if $m_\ell^{(j)} \neq 0$, equations in (8) determine $\zeta^{(j)}$, $\rho^{(j)}$. If $m_\ell^{(j)} = 0$, then $\zeta^{(j)}$, $\rho^{(j)}$ can be assigned independently from χ_ℓ. The above holds for every ℓ in $\{1, \ldots, k\}$. Thus, if $\chi_1, \ldots, \chi_k$ distributes uniformly over $\mathbb{Z}_p^k$, then other coins distribute uniformly as well retaining the consistency with the view of $\mathcal{A}$.

Now we see that $\left(M_\ell^\dagger / M_\ell\right)^{\chi_\ell}$ distributes uniformly over $\mathbb{G}$. Therefore $Z^\star = 1$ happens only with probability $1/p$. Thus, $\mathcal{B}$ outputs correct answer with probability $\mathrm{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{B}} = 1/p + (1-1/p)(1-1/p)\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}}$, which leads to $\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}} \leq \mathrm{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{B}} + 1/p$ as claimed. □

Remark 1. On tag extension. The tag can be easily extended to the form $(G^t, G_1^t, G_2^t, ...)$ for extra bases $G_1, G_2, ...$ provided as a part of gk. (In the security proof, the extended part is computed from the first element by using $\log_G G_i$. This is possible since the extra generators in gk are chosen by the reduction algorithm.) Such an extension is in particular needed when the TOS is coupled with other signature schemes whose message space is structured as above. Indeed, it is the case for an application in Section 4.

Remark 2. Signing lengthy messages. The TOS can be used to sign messages of unbound length by chaining the signatures. Every message block except for the last one is followed by a tag used to sign the next block. The signature consists of all internal signatures and tags. The initial tag is considered as the tag. For a message consisting of m group elements, it repeats $\tau := 1 + \max(0, \lceil \frac{m-k}{k-1} \rceil)$ times. The signature consists of $4\tau - 2$ elements.

4 Efficient SPS Based on DLIN

As the first application of our TOS, we present an efficiet structure-preserving signature scheme. The construction follows the framework suggested in Theorem 1. We begin with introducing an RMA-secure SPS as a building block. The scheme in Fig. 2 is an RMA-secure SPS for messages in the form $(C^m, F^m, U^m) \in \mathbb{G}^3$ defined by generators (C, F, U) provided in gk. The scheme is a modification of the one in Sec.5.3 of [1] that signs longer message of the form $\{(C^{m_1}, C^{m_2}, F^{m_1}, F^{m_2}, U^{m_1}, U^{m_2})\}$. Our scheme is obtained by restricting $m_2 = 0$ and removing useless operations relevant to m_2. The security is stated in Theorem 5 below, whose proof is obtained as a trivial modification of the proof of Theorem 24 in [1].

Theorem 5. *The above* rSIG *scheme is secure against random message attacks under the DLIN assumption. In particular, for any polynomial-time adversary $\mathcal{A}$ against* rSIG *that makes at most q_s signing queries, there exists polynomial-time algorithm $\mathcal{B}$ for DLIN such that* $\mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{rSIG},\mathcal{A}}(\lambda) \leq (q_s + 2) \cdot \mathrm{Adv}^{\mathsf{dlin}}_{\mathcal{G},\mathcal{B}}(\lambda)$.

[Scheme rSIG]

Let gk be a common parameter that consists of group description $\Lambda = (p, \mathbb{G}, \mathbb{G}_T, e)$ and default generator G. It also includes randomly chosen generators C, F, and U.

$\mathsf{rSIG.Key}(gk)$**:** Given $gk := (\Lambda, G, C, F, U)$ as input, uniformly select V, V_1, V_2, H from $\mathbb{G}^*$ and a_1, a_2, b, α, and ρ from $\mathbb{Z}_p^*$. Then compute and output $vk := (B, A_1, A_2, B_1, B_2, R_1, R_2, W_1, W_2, V, V_1, V_2, H, X_1, X_2)$ and $sk := (vk, K_1, K_2)$ where

$$B := G^b, \quad A_1 := G^{a_1}, \quad A_2 := G^{a_2}, \quad B_1 := G^{b \cdot a_1}, \quad B_2 := G^{b \cdot a_2}$$
$$R_1 := V V_1^{a_1}, \quad R_2 := V V_2^{a_2}, \quad W_1 := R_1^b, \quad W_2 := R_2^b,$$
$$X_1 := G^{\rho}, \quad X_2 := G^{\alpha \cdot a_1 \cdot b/\rho}, \quad K_1 := G^{\alpha}, \quad K_2 := G^{\alpha \cdot a_1} .$$

$\mathsf{rSIG.Sign}(sk, msg)$**:** Parse msg into (M_1, M_2, M_3). Pick random $r_1, r_2, z_1, z_2 \in \mathbb{Z}_p$. Let $r = r_1 + r_2$. Compute and output signature $\sigma := (S_0, S_1, \ldots S_7)$ where

$$S_0 := (M_3 H)^{r_1}, \quad S_1 := K_2 V^r, \quad S_2 := K_1^{-1} V_1^r G^{z_1}, \quad S_3 := B^{-z_1},$$
$$S_4 := V_2^r G^{z_2}, \quad S_5 := B^{-z_2}, \quad S_6 := B^{r_2}, \quad S_7 := G^{r_1}.$$

$\mathsf{rSIG.Vrf}(vk, \sigma, msg)$**:** Parse msg into (M_1, M_2, M_3) and σ into $(S_0, S_1, \ldots, S_7)$. Also parse vk accordingly. Verify the following pairing product equations:

$$e(S_7, M_3 H) = e(G, S_0),$$
$$e(S_1, B)\, e(S_2, B_1)\, e(S_3, A_1) = e(S_6, R_1)\, e(S_7, W_1),$$
$$e(S_1, B)\, e(S_4, B_2)\, e(S_5, A_2) = e(S_6, R_2)\, e(S_7, W_2)\, e(X_1, X_2),$$
$$e(F, M_1) = e(C, M_2), \quad e(U, M_1) = e(C, M_3)$$

Fig. 2. RMA-secure SPS for 1 message block based on DLIN

According to Theorem 1, combining TOS in Section 3 and rSIG in Fig. 2 results in a chosen-message-secure SPS. (Note that tags of TOS are extended as explained in the remark in the end of Section 3 so that they fit to the message space of rSIG. Concretely, by using generator C from rSIG as G in the description of TOS, and also using extra generators F and U, a tag is defined as $(T_1, T_2, T_3) := (C^t, F^t, U^t)$.) The resulting SPS yields signatures consisting of 14 group elements $(T_1, T_2, T_3, Z, R, S, S_0, \ldots, S_7)$ and evaluates 7 pairing product equations in the verification. Since both TOS and rSIG are based on DLIN, the resulting SPS is secure under DLIN as well. (They are actually based on SDP that is a seemingly weaker computational assumption.)

The efficiency is summarised in Table 1. It is compared to existing efficient structure-preserving schemes over symmetric bilinear groups. We measure efficiency by counting the number of group elements and the number of pairing product equations for verifying a signature. The figures do not count default generator G in gk.

To see how a small difference in the size of signatures and the number of PPEs impacts the efficiency in applications, we assess the cost of proving possession of valid signatures and messages by using Groth-Sahai NIWI proof system. Column "Proof Cost σ" shows the number of group elements in the commitment of a signature and the proof. If there are randomizable parts in a signature, they are put in the clear. It is the case for the scheme in [2]. Similarly, column "Proof Cost (σ, msg)" shows the size when both messages and signatures are committed as witnesses.

Table 1. Comparison of constant-size SPS over symmetric bilinear groups. "Reduction Cost" shows the loss factor to the underlying assumptions. "Proof Cost" is the number of group elements in the Groth-Sahai NIWI proof of knowledge about a valid signature.

Scheme	$\|msg\|$	$\|gk\| + \|vk\|$	$\|\sigma\|$	#(PPE)	Assumption	Reduction Cost	Proof Cost σ	Proof Cost (msg, σ)
[2]	k	$2k + 12$	7	2	q-SFP	1	19	3k+19
[1]	k	$2k + 25$	17	9	DLIN	$(2q)^{-1}$	84	3k+84
this paper	k	$2k + 20$	14	7	DLIN	$(q+1)^{-1}$	69	3k+69

5 Chosen-Ciphertext Secure Public-Key Encryption

5.1 Simulation Extractable NIZK

A non-interactive zero-knowledge argument system $\mathsf{NIZK} = \mathsf{NIZK}.\{\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}\}$ for a relation R consists of three algorithms: $\mathsf{NIZK.Crs}$ that takes a common setup parameter and generates a common reference string crs, the proof algorithm $\mathsf{NIZK.Prv}$ which on input crs, an instance x and a witness w for the truth of the statement R, outputs a proof π, and the verification algorithm $\mathsf{NIZK.Vrf}$ that on input crs, an instance x, and a proof π either accepts or rejects the proof. It is equipped with a pair of algorithms, $\mathsf{NIZK.CrsSim}$ and $\mathsf{NIZK.PrvSim}$, that simulates $\mathsf{NIZK.Crs}$ and $\mathsf{NIZK.Prv}$, respectively. $\mathsf{NIZK.CrsSim}$ outputs crs and a simulation-trapdoor, τ_{zk}, and $\mathsf{NIZK.PrvSim}$ produces proofs by using the trapdoor. NIZK is (unbounded multi-theorem) zero-knowledge, if given oracle access to either $\mathsf{NIZK.PrvSim}(\tau_{\mathsf{zk}}, \cdot)$ or $\mathsf{NIZK.Prv}(crs, \cdot, \cdot)$, with true statements as inputs, any polynomial-time adversary trying to distinguish the oracles has advantage upper bounded by a negligible function, ϵ_{zk}, in the security parameter. A NIZK is strongly simulation-sound if adversary $\mathcal{A}$ is given oracle access to $\mathsf{NIZK.PrvSim}(\tau_{\mathsf{zk}}, \cdot)$ and outputs valid (x, π) only with negligible probability. It can be relaxed to standard simulation soundness by requiring that only x is not reused.

A NIZK is a non-interactive proof of knowledge [28] if $\mathsf{NIZK.Crs}$ additionally outputs an extraction trapdoor, τ_{ex}, and there exists an efficient algorithm, $\mathsf{NIZK.Ext}$, that extracts a correct witness w from any (x, π) that $1 = \mathsf{NIZK.Vrf}(crs, x, \pi)$ with probability $1 - \epsilon_{\mathsf{ks}}$ for some negligible function ϵ_{ks}. This property is called knowledge soundness. A simulation-extractable NIZK extends a NIZK proof of knowledge so that $\mathsf{NIZK.Crs}$ outputs extraction trapdoor τ_{ex} and simulation trapdoor τ_{zk} at the same time. Then it is simulation-extractable if $\mathsf{NIZK.Ext}$ works even if an adversary is given oracle access to $\mathsf{NIZK.PrvSim}(\tau_{\mathsf{zk}}, \cdot)$. More precisely,

$$\Pr\left[\begin{array}{l}(crs,\tau_{\mathsf{ex}},\tau_{\mathsf{zk}})\leftarrow\mathsf{NIZK.Crs}\\(x,\pi)\leftarrow\mathcal{A}^{\mathsf{NIZK.PrvSim}(\tau_{\mathsf{zk}},\cdot)}(crs)\\w\leftarrow\mathsf{NIZK.Ext}(crs,x,\pi,\tau_{\mathsf{ex}})\end{array}\;\middle|\;\begin{array}{l}\mathsf{NIZK.Vrf}(crs,x,\pi)=1\wedge\\R(x,w)\neq 1\end{array}\right]<\epsilon_{\mathsf{se}} \qquad (9)$$

holds for a negligible function ϵ_{se}.

Recall that simulation soundness only guarantees that x is a true statement whereas simulation extractability additionally guarantees that the witness be efficiently extractable. When the number of oracle access is unlimited (limited to only once, resp.), it is called unbounded (one-time, resp.) simulation extractability.

We show that the simulation-sound NIZK of [26] is simulation extractable if the underlying NIZK is a proof of knowledge system. Let $\mathsf{POK} = \mathsf{POK}.\{\mathsf{Crs},\mathsf{Prv},\mathsf{Vrf},\mathsf{Ext}\}$ be a NIZK proof of knowledge system, $\mathsf{SIG} = \mathsf{SIG}.\{\mathsf{Key},\mathsf{Sign},\mathsf{Vrf}\}$ be a signature scheme, and $\mathsf{OTS} = \mathsf{OTS}.\{\mathsf{Key},\mathsf{Sign},\mathsf{Vrf}\}$ be a one-time signature scheme. Their construction of $\mathsf{SE\text{-}NIZK} = \mathsf{SE\text{-}NIZK}.\{\mathsf{Crs},\mathsf{Prv},\mathsf{Vrf},\mathsf{PrvSim},\mathsf{Ext}\}$ is shown in Fig. 3

[Scheme SE-NIZK]

$\mathsf{SE\text{-}NIZK.Crs}(gk)$: It takes gk and runs $(crs_{\mathsf{pok}},\tau_{\mathsf{ex}})\leftarrow\mathsf{POK.Crs}(gk)$, $(vk,sk)\leftarrow\mathsf{SIG.Key}(gk)$. It then outputs $crs := (gk, crs_{\mathsf{pok}}, vk)$, $\tau_{\mathsf{ex}} := \tau_{\mathsf{ex}}$, and $\tau_{\mathsf{zk}} := sk$.

$\mathsf{SE\text{-}NIZK.Prv}(crs,x,w)$: Run $opk\leftarrow\mathsf{OTS.Key}(gk)$. Set $\sigma=\perp$. Let $x_{\mathsf{se}} := (x, opk)$ and $w_{\mathsf{se}} := (w,\sigma)$. Set relation R_{se} be

$$R_{\mathsf{se}}(x_{\mathsf{se}},w_{\mathsf{se}}) := (R(x,w)=1)\vee(\mathsf{SIG.Vrf}(vk,\sigma,opk)=1)\,.$$

Run $\pi\leftarrow\mathsf{POK.Prv}(crs_{\mathsf{pok}},x_{\mathsf{se}},w_{\mathsf{se}})$, and $\sigma_o\leftarrow\mathsf{OTS.Sign}(osk,\pi)$. Output $\pi_{\mathsf{se}} := (\pi, opk, \sigma_o)$.

$\mathsf{SE\text{-}NIZK.Vrf}(crs,x,\pi_{\mathsf{se}})$: Parse $(\pi, opk, \sigma_o)\leftarrow\pi_{\mathsf{se}}$. Verify both σ_o and π.

$\mathsf{SE\text{-}NIZK.PrvSim}(crs,\tau_{\mathsf{zk}},x)$: Parse $(gk, crs_{\mathsf{pok}}, vk)\leftarrow crs$ and $sk\leftarrow\tau_{\mathsf{zk}}$. Run $opk\leftarrow\mathsf{OTS.Key}(gk)$ and $\sigma\leftarrow\mathsf{SIG.Sign}(sk,opk)$. Set $w_{\mathsf{se}} := (\perp,\sigma)$. Run $\pi\leftarrow\mathsf{POK.Prv}(crs_{\mathsf{pok}},x_{\mathsf{se}},w_{\mathsf{se}})$ and $\sigma_o\leftarrow\mathsf{OTS.Sign}(osk,\pi)$. Output $\pi_{\mathsf{se}} := (\pi, opk, \sigma_o)$.

$\mathsf{SE\text{-}NIZK.Ext}(crs,\tau_{\mathsf{ex}},x,\pi_{\mathsf{se}})$: Parse $(gk, crs_{\mathsf{pok}}, vk)\leftarrow crs$ and $(\pi, opk, \sigma_o)\leftarrow\pi_{\mathsf{se}}$. Run $w_{\mathsf{se}}\leftarrow\mathsf{POK.Ext}(crs_{\mathsf{pok}},\tau_{\mathsf{ex}},\pi,(x,opk))$ and return w in $w_{\mathsf{se}}=(w,\sigma)$.

Fig. 3. Simulation-Extractable Non-Interactive Zero-Knowledge Proof System

Theorem 6. *If* POK *is a witness indistinguishable proof of knowledge system with knowledge-soundness error* ϵ_{ks}, SIG *is unforgeable against non-adaptive chosen message attacks with advantage* ϵ_{sig}, *and* OTS *is strongly one-time unforgeable against chosen message attacks with advantage* ϵ_{ots}, *then* SE-NIZK *is strongly simulation-extractable NIZK with simulation-extraction error* $\epsilon_{\mathsf{se}}\leq\epsilon_{\mathsf{ots}}+\epsilon_{\mathsf{ks}}+\epsilon_{\mathsf{sig}}$.

Proof. Correctness of the scheme and zero-knowledge property is verified by inspecting the construction. Computational zero-knowledge is not hard to verify due to the witness indistinguishability of POK and the construction of SE-NIZK.PrvSim.

We focus on showing simulation extractability. Suppose that adversary $\mathcal{A}$ accesses $\mathsf{SE\text{-}NIZK.PrvSim}(crs, \tau_{\mathsf{zk}}, \cdot)$ as an oracle and eventually outputs $x^\star$ and $\pi^\star = (\pi^\star, opk^\star, \sigma^\star)$ that passes SE-NIZK.Vrf. For $\mathcal{A}$ to be successful, it must be the case that $(x^\star, \pi^\star) \notin \{x_i, \pi_i\}$ and $(x^\star, \pi^\star) \notin R$. Recall that $\pi^\star_{\mathsf{se}} = (\pi^\star, opk^\star, \sigma^\star)$. We distinguish two cases:

Case 1: $opk^\star = opk_i$ happens for opk_i returned from the oracle. In this case, $(x^\star, \pi^\star, \sigma^\star) \neq (x_i, \pi_i, \sigma_i)$ and we have a valid forgery for OTS. This happens with probability at most ϵ_{ots} due to the strong one-time unforgeability of OTS.

Case 2: $opk^\star \neq opk_i$ for all opk_i. By executing $\mathsf{SE\text{-}NIZK.Ext}(crs, \tau_{\mathsf{ex}}, x^\star, \pi^\star)$, we have $w_{\mathsf{se}} = (w, \sigma)$ that either $R(x^\star, w) = 1$ or $\mathsf{SIG.Vrf}(vk, \sigma, opk^\star) = 1$ for vk included in crs. The extraction is successful with probability $1 - \epsilon_{\mathsf{ks}}$ due to the knowledge-soundness of NIZK. Then, if the former happens, we have extracted correct witness for $x^\star$ and $\mathcal{A}$ is unsuccessful. Otherwise, we have a valid forgery for SIG since its message $opk^\star$ is fresh. This happens with probability at most ϵ_{sig} due to the unforgeability against non-adaptive chosen-message attacks for SIG. (The non-adaptiveness is due to the fact that all opk_i can be generated in advance.)

In total, the extraction is successful with probability $(1 - \epsilon_{\mathsf{se}}) = (1 - \epsilon_{\mathsf{ots}})(1 - \epsilon_{\mathsf{ks}})(1 - \epsilon_{\mathsf{sig}})$ which leads to $\epsilon_{\mathsf{se}} \leq \epsilon_{\mathsf{ots}} + \epsilon_{\mathsf{ks}} + \epsilon_{\mathsf{sig}}$ as stated. □

Instantiating SE-NIZK. We instantiate the above generic SE-NIZK in several ways. The result is several SE-NIZKs that have different sets of properties as summarised in Table 2.

SE-NIZK0: The original instantiation in [26]. SIG is a tree-based signature scheme with their original one-time signature scheme, and OTS is instantiated with the Pedersen commitment as a one-time signature that is not structure-preserving. The result is a unbounded SE-NIZK.

SE-NIZK1: SIG remains a tree-based scheme but we replace the internal one-time signatures with our TOS in plug-in manner. The result is a more efficient unbounded SE-NIZK. This shows how plug-in replacement of low-level building block impacts to the efficiency.

SE-NIZK2: The same as SE-NIZK1 but we instantiate OTS with our TOS as well. Since that OTS is the only non-structure-preserving component in SE-NIZK1, the result is structure-preserving unbounded SE-NIZK. A problem is that the TOS must be able to sign the entire proof that linearly grows in the size of the public-key of the TOS itself. We therefore use the technique of chaining the signatures as mentioned in Remark 2 in Section 3. The same technique is used when the one-time key is signed at the bottom of the tree-based signing. The resulting SE-NIZK is used in constructing structure-preserving publicly verifiable CCA-secure PKE tightly-secure with multiple challenges.

SE-NIZK3: We use TOS for SIG. No tree-based construction here. This means the signature can be generated only once for simulation and the result is structure-preserving one-time SE-NIZK. As well as SE-NIZK2, we use the signatrue

chaining. The resulting scheme can be used in constructing efficient structure-preserving publicly verifiable CCA-secure PKE. We can add leakage resilience (LR) if desired.

SE-NIZK4: As well as SE-NIZK3 we instantiate SIG with our TOS but leave OTS with the one based on the Pedersen commitment for the sake of efficiency. In exchange of losing structure-preservation, it results in a very efficient one-time SE-NIZK. It will be used for publicly verifiable CCA-secure PKE (with LR if desired).

Table 2. Properties of the instantiations of SE-NIZK. Efficiency is presented in subjective term. Objective evaluation of efficiency is in Table 3.

scheme	efficiency	simulatability	structure-preservance
SE-NIZK0	less efficient	unbounded	no
SE-NIZK1	moderate	unbounded	no
SE-NIZK2	less efficient	unbounded	yes
SE-NIZK3	efficient	one-time	yes
SE-NIZK4	very efficient	one-time	no

We give a general formula that evaluate the cost of the generic construction. The generic SE-NIZK uses the S_0-or-S_1 structure so that real proof is done for statement S_0 whereas simulation is done with a witness for statement S_1. It is however believed that the OR structure with Groth-Sahai proof system is as costly as doubling the number of elements in a proof. It is true for general statements. But for the specific construction of SE-NIZK, it can be done much less costly. taking the advantage of the fact that there is no common witnesses shared by statements S_0 and S_1.

Regarding the proof of disjunction, we sketch the construction of [10] and refer to [10] for details. The prover commits to $1_{\mathbb{G}}$ or G with X, and show its correctness by proving a single non-linear relation $e(X, X) = e(X, G)$. We call X a switcher as it switches the statement that is really proven. Let $X_0 = X$ and $X_1 = G \cdot X^{-1}$. Then for every pairing product equation in S_b, if pairing $e(A, B)$ with some constants A and B is involved, one of them say A is transformed to variable Y and prove its correctness by showing $e(Y, G) = e(A, X_b)$ holds. (Observe that if $X_b = G$, it guarantees that $Y = A$. Otherwise, if $X_b = 1$, it holds for $Y = 1$.) After that, every pairing in every relation in S_b includes at least one variable. Now, if $X_b = G$, one can still satisfy the relations with the legitimate witnesses. Otherwise, if $X_b = 1_{\mathbb{G}}$, they can be satisfied by setting $1_{\mathbb{G}}$ to all variables, which allows zero-knowledge simulation.

Now the number of group elements in a proof of SE-NIZK is counted as follows. Let $S_0 : (R(x, w) = 1)$ and $S_1 : (\mathsf{SIG.Vrf}(vk, \sigma, opk) = 1)$ be the statements represented by pairing product equations. The proof size of SE-NIZK is as follows:

$$
\begin{aligned}
&(\text{cost for } S_0) + (\text{cost for switcher}) + (\text{cost for } S_1) + (\text{cost for } \mathsf{OTS}) \\
&= (\text{cost for } S_0) && (10)\\
&+ (|com| \times 1 + |\pi_{NL}| \times 1) && (11)\\
&+ (|com| \times (|\sigma_{\mathsf{sig}}| + S_1(C)) + |\pi_L| \times (S_1(L) + S_1(C)) + |\pi_{NL}| \times S_1(NL) && (12)\\
&+ (|opk_o| + |\sigma_o|) && (13)
\end{aligned}
$$

Here, parameters $|\pi_{L/NL}|$, $|opk_o|$, $|\sigma_o|$, $|\sigma_{\mathsf{sig}}|$, $|com|$ are the size of a proof for a linear/non-linear relation, a one-time public-key of OTS, a signature of OTS, a signature of SIG, and commitment per variable, respectively. Also, $S_1(L/NL)$ and $S_1(C)$, denote the number of linear/non-linear relations and constant pairings, respectively, in SIG.Vrf where signatures are considered as variables. By "overhead", we mean the size of (11)+(12)+ (13) since it is the cost for achieving simulation extractability on top of simply proving the original statement S_0.

With the Groth-Sahai proof over the DLIN setting, we have $(|com|, |\pi_L|, |\pi_{NL}|) = (3, 3, 9)$. Other parameters $(|\sigma_{\mathsf{sig}}|, S_1(C), S_1(L), S_1(NL), |opk_o|, |\sigma_o|)$ differ in every instantiation and summarised as in Table 3. For SE-NIZK2,3 that uses the signature chaining, let k_1 and k_2 be block size of a message for SIG and OTS, respectively. Also let τ_1 and τ_2 be the length of the chains determined by $\tau_2 := 1 + \max(0, \lceil \frac{m_2 - k_2}{k_2 - 1} \rceil)$ and $\tau_1 := 1 + \max(0, \lceil \frac{m_1 - k_1}{k_1 - 1} \rceil)$ where $m_2 := |opk_o|$ and $m_1 :=$ (cost for S_0) + (cost for switcher) + (cost for S_0). Then the overhead in a proof is $\psi_2 := 21d + 18\tau_2 + 4\tau_1 + 14k_1 + 45$ for SE-NIZK2 and $\psi_3 := 18\tau_2 + 4\tau_1 + 14k_1 + 45$ for SE-NIZK3. When those schemes are used, parameters k_1 and k_2 should be chosen to minimize the overhead. Unfortunately, the general assessment in Table 3 is not intuitive enough to see the difference of efficiency due to the several parameters involved. One can see their difference in more concrete manner in the next section.

Table 3. Parameterized costs for simulation extractable NIZKs. See the main text for the meaning of parameters.

scheme	$\|\sigma_{\mathsf{sig}}\|$	$S_1(C)$	$S_1(L)$	$S_1(NL)$	$\|opk_o\|$	$\|\sigma_o\|$	overhead
SE-NIZK0	$10d + 2$	5	3	$3d$	2	2	$57d + 61$
SE-NIZK1	$5d + 1$	3	$2(d + 1)$	0	2	2	$21d + 43$
SE-NIZK2	$5d + 4\tau_2 - 2$	$2k_1 + 6$	$2(d + \tau_2)$	0	$2k_1 + 5$	$4\tau_1 - 2$	ψ_2
SE-NIZK3	$4\tau_2 - 2$	$2k_1 + 6$	$2\tau_2$	0	$2k_1 + 5$	$4\tau_1 - 2$	ψ_3
SE-NIZK4	3	2	3	0	2	2	46

We note that the instantiations follow the generic construction rigorously. Some hand-crafted optimization is possible in reality by carefully choosing variables and constants in GS-proofs. In particular, it is not necessary to commit the entire signature when we compute π. The tag and Z in every signature can be sent in the clear. Such optimization saves considerable number of group elements. The impact of optimization will be discussed in the next section with concrete numbers.

5.2 Tight/Structure-Preserving CCA-Secure Encryption from SE-NIZK

In [26], the SS-NIZK is used to construct a chosen-ciphertext-secure (CCA) PKE that is secure against multiple challenges retaining the tightness property. It follows the Naor-Yung paradigm that combines *two* chosen-plaintext-secure public-key encryption schemes (CPA-secure PKE) with an SS-NIZK. As we now know that their instantiation of SS-NIZK actually gives SE-NIZK, we rather follow more efficient generic construction by Dodis, et. al.,[17] that combines *one* CPA-secure PKE with SE-NIZK. This

results in more efficient CCA PKE. Since slightly different components is used in [17] for their purpose of adding leakage resilience and no quantified evaluation was presented, we restate their theorem in a simplified form with a proof in the following.

Let CPA be a CPA-secure encryption scheme and SE-NIZK be simulation extractable NIZK. We construct CCA-secure encryption scheme $\mathsf{PKE} := \mathsf{PKE}.\{\mathsf{Key}, \mathsf{Enc}, \mathsf{Dec}\}$ by combining CPA and SE-NIZK as shown in Fig. 4. Let gk be a common parameter generated by $\mathsf{Setup}(1^\lambda)$. Underlying encryption scheme CPA must satisfy the following property. There exists efficiently computable function W and efficiently verifiable relation R such that

$$(R((ek_{\mathsf{cpa}}, c_{\mathsf{cpa}}), (msg, W(r)) = 1) \iff (c_{\mathsf{cpa}} = \mathsf{CPA.Enc}(ek_{\mathsf{cpa}}, msg; r)). \quad (14)$$

Function W is understood as a converter that transforms random coin r into a form that is easily handled in verifying relation R. In our instantiation with Groth-Sahai proof system, W transforms $r \in \mathbb{Z}_p$ to a vector of group elements.

[Scheme PKE**]**

$\mathsf{PKE.Key}(gk)$**:** Run $(crs_{\mathsf{nizk}}, \tau_{\mathsf{zk}}, \tau_{\mathsf{ex}}) \leftarrow \mathsf{SE\text{-}NIZK.Crs}(gk)$, $(ek_{\mathsf{cpa}}, dk_{\mathsf{cpa}}) \leftarrow \mathsf{CPA.Key}(gk)$. Set $ek := (crs_{\mathsf{nizk}}, ek_{\mathsf{cpa}})$ and $dk := dk_{\mathsf{cpa}}$.

$\mathsf{PKE.Enc}(ek, msg)$**:** Run $c_{\mathsf{cpa}} \leftarrow \mathsf{CPA.Enc}(ek_{\mathsf{cpa}}, msg; r)$ and $\pi \leftarrow \mathsf{SE\text{-}NIZK.Prv}(crs_{\mathsf{nizk}}, c_{\mathsf{cpa}}, (msg, r))$. The proof is for relation $1 = R((ek_{\mathsf{cpa}}, c_{\mathsf{cpa}}), (msg, r))$. Output ciphertext $c := (c_{\mathsf{cpa}}, \pi)$.

$\mathsf{PKE.Dec}(dk, c)$**:** Parse c into (c_{cpa}, π). If $0 \leftarrow \mathsf{SE\text{-}NIZK.Vrf}(crs_{\mathsf{nizk}}, c_{\mathsf{cpa}}, \pi)$, return $\perp$. Otherwise, output $msg := \mathsf{CPA.Dec}(dk_{\mathsf{cpa}}, c_{\mathsf{cpa}})$.

Fig. 4. CCA-secure PKE from SE-NIZK

In addition to the use of only one CPA-secure encryption, the construction in Fig. 4 is different from [26] in the following sense. In [26], crs_{nizk} is included in gk and common for all users. Hence the security of resulting CCA PKE fully relies on the secrecy of the trapdoors behind crs_{nizk}. In our case, fresh crs_{nizk} is selected for every public-key. If gk includes no trapdoors (as is usually the case in the certified group model where only the group description Λ is included in gk), the security of the resulting CCA PKE is reduced to complexity assumptions defined over gk. In fact, when gk does not include trapdoors, and the underlying complexity assumption is random self reducible, it is rather trivial to preserve tightness when extending the security reduction from the single-user to the multi-user setting because no secret information is shared between users. On the contrary, it is not trivial to preserve tightness in the multi-challenge setting since every challenge is related to the same public-key which involves a trapdoor. We therefore focus on security in the multi-challenge and single-user setting in the following argument.

Theorem 7. *If* CPA *is left-or-right CPA secure encryption scheme with advantage* ϵ_{cpa} *and* SE-NIZK *be unbounded (or one-time, resp.) simulation-extractable NIZK with zero-knowledge error* ϵ_{zk} *and simulation-extraction error* ϵ_{se}*, then* PKE *is multi-challenge (or standard single-challenge, resp.) CCA-secure with advantage* $\epsilon_{\mathsf{cca}} \leq 2 \cdot (\epsilon_{\mathsf{zk}} + \epsilon_{\mathsf{se}}) + \epsilon_{\mathsf{cpa}}$.

Proof. The proof structure follows [17]. Games are numbered by the combination of an idealization step counter and a bit indicating whether to encrypt the left or the right side to visualize its inherent symmetry.

Game 0.0. This is the IND-CCA security experiment from [26], executed with $b = 0$. The challenger always returns encryptions of msg_0.

Game 1.0. This game is identical to Game 0.0, except that we use the zero-knowledge simulator of SE-NIZK to generate proofs in the challenge ciphertexts. (If SE-NIZK is one-time simulation extractable, this is limited to a single challenge.) We have $[Pr[Win_{1.0}] - Pr[Win_{0.0}] \leq \epsilon_{\mathsf{zk}}$.

Game 2.0 This game is identical to Game 1.0, except that decryption queries $c = (c_{\mathsf{cpa}}, \pi)$ are answered by running SE-NIZK.Ext on π to extract msg. (This modification accommodates with the previous one since SE-NIZK.Crs outputs trapdoors for simulation and extraction at the same time.) We have $\Pr[Win_{2.0}] - \Pr[Win_{1.0}] \leq \epsilon_{\mathsf{se}}$.

Game 2.1 This game is identical to Game 2.0, except that the challenger always returns encryptions of msg_1. As we do not use dk_{cpa} anywhere we can do a reduction to IND-CPA security and have $\Pr[Win_{2.1}] - \Pr[Win_{2.0}] \leq \epsilon_{\mathsf{cpa}}$.

Game 1.1. This game is identical to Game 2.1, except that decryption queries $c = (c_{\mathsf{cpa}}, \pi)$ are no longer answered by running the extractor but by decrypting c_{cpa} to obtain msg. We have $\Pr[Win_{1.1}] - \Pr[Win_{2.1}] \leq \epsilon_{\mathsf{se}}$.

Game 0.1. This game is identical to Game 1.1, except that we no longer use the zero-knowledge simulator of SE-NIZK to generate all proofs but generate them honestly. We have $\Pr[Win_{0.1}] - \Pr[Win_{1.1}] \leq \epsilon_{\mathsf{zk}}$. This is the IND-CCA security experiment executed with $b = 1$.

By accumulating the differences, we have $\epsilon_{\mathsf{cca}} \leq 2 \cdot (\epsilon_{\mathsf{zk}} + \epsilon_{\mathsf{se}}) + \epsilon_{\mathsf{cpa}}$ as stated. □

We instantiate CPA with the linear encryption scheme [27,29] shown in Fig. 5. It is IND-CPA secure and tightly reducible to DLIN in the multi-challenge and multi-user setting as formally proven in [26]. Well formness of a ciphertext can be proven by providing a GS proof for relations

$$e(C_1, \underline{G}) = e(G_1, \underline{W_1}), \qquad e(C_2, \underline{G}) = e(G_2, \underline{W_2}),$$
$$e(\underline{W_1}\,\underline{W_2}, G) = e(C_3/\underline{M}, G), \qquad e(\underline{G}, G) = e(G, \underline{X_0}).$$

The underlined variables $G, W_1 := G^{r_1}, W_2 := G^{r_2}, M$ are witneses and X_0 is a switcher as explained in Section 5.1. Accordingly, the "cost for S_0" in (10) is 24 group elements (12 for four commitments and 12 for proof of four linear equations).

The SE-NIZK in the construction of PKE can be instantiated with any SE-NIZKi in Section 5.1. The efficiency and properties of the resulting PKE is shown in Table 4. For SE-NIZK1,2,3 that uses a tree-based signature scheme, we set the depth of the tree

[Scheme CPA**]**
Let gk include $\Lambda = (p, \mathbb{G}, \mathbb{G}_T, e)$ and generator $G \in \mathbb{G}$ as global parameters.

CPA.Key(gk)**:** Uniformly select y_1, y_2 from $\mathbb{Z}_p^*$. Compute $G_1 = G^{y_1}$ and $G_2 = G^{y_2}$, And then output $ek := (\Lambda, G_1, G_2)$ and $dk := (ek, y_1, y_2)$. The message space is $\mathbb{G}$.

CPA.Enc(ek, msg)**:** Parse msg into $M \in \mathbb{G}$ and ek accordingly. Pick random $r_1, r_2 \in \mathbb{Z}_p$. Compute and output signature $c := (C_1, C_2, C_3)$ where $C_1 := G_1^{r_1}$, $C_2 := G_2^{r_2}$, and $C_3 := M\, G^{r_1+r_2}$.

CPA.Dec(dk, c)**:** Parse c into (C_1, C_2, C_3), and dk into (y_1, y_2). Then output $M := C_3\, C_1^{-1/y_1}\, C_2^{-1/y_2}$.

Fig. 5. The Linear Encryption Scheme

Table 4. Properties and ciphertext size of CCA PKE constructed with SE-NIZKi. Tight security is for multiple challenges and users.

SE-NIZKi	Cipheretxt Size	Properties			Parameter
	Size	Publicly-Verifiable	Tightly-Secure	Strucure-Preserving	Setting
0	1228	yes	yes	no	d=20
1	490	yes	yes	no	d=20
2	916	yes	yes	yes	d=20, k_1=31, k_2=13
3	304	yes	no	yes	k_1=19, k_2=7
4	73	yes	no	no	

to $d = 20$, which allows up to 2^{20} simulations. (If one demands virtually unbounded simulatability, d should equal to the security parameter as suggested in [26].) For SE-NIZK3,4 that uses TOS as OTS, we seek for optimal value for parameter k_1 and k_2 that minimizes the size of the cipehrtext. As originally stated in [17], leakage resilience can be added by using a leakage resilient CPA encryption from [17] while retaining other properties.

We finally remark that the ciphertext size is assessed with non-optimized instantiations of SE-NIZKi. Following the already mentioned observation that only a part of a simulated signature in NIZK must be committed, one can optimize the GS proofs and reduce the size of ciphertext to 398 from 490 with SE-NIZK1 at $d = 20$, 731 from 916 with with SE-NIZK2 at $d = 20, k_1 = 27, k_2 = 11$, and 273 from 304 with SE-NIZK3 at $d = 20, k_1 = 17, k_2 = 6$.

6 Conclusion

We present a new efficient tagged one-time signature scheme that features tight reduction to DLIN and optimal tag size. We then revisit several generic constructions where (tagged) one-time signatures play a central role, and build structure preserving signature and public-key encryption schemes that for the first time simultaneously achieve several desirable properties. Although many of our instantiations are not necessarily

practical with hundreds of group elements, the concrete efficiency assessment should serve as a reference and as a first step.

Our construction uses the symmetry of the pairing in an essential way. It is left as an open problem to construct TOS schemes with optimal tag size and a tight security reduction over asymmetric pairings.

References

1. Abe, M., Chase, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Constant-Size Structure-Preserving Signatures: Generic Constructions and Simple Assumptions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 4–24. Springer, Heidelberg (2012)
2. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-Preserving Signatures and Commitments to Group Elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
3. Abe, M., Groth, J., Haralambiev, K., Ohkubo, M.: Optimal Structure-Preserving Signatures in Asymmetric Bilinear Groups. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 649–666. Springer, Heidelberg (2011)
4. Abe, M., Haralambiev, K., Ohkubo, M.: Signing on group elements for modular protocol designs. IACR ePrint Archive, 2010/133 (2010), `http://eprint.iacr.org`
5. Abe, M., Haralambiev, K., Ohkubo, M.: Group to Group Commitments Do Not Shrink. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 301–317. Springer, Heidelberg (2012)
6. Attrapadung, N., Libert, B., Peters, T.: Computing on Authenticated Data: New Privacy Definitions and Constructions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 367–385. Springer, Heidelberg (2012)
7. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable Proofs and Delegatable Anonymous Credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (2009)
8. Bellare, M., Shoup, S.: Two-Tier Signatures, Strongly Unforgeable Signatures, and Fiat-Shamir Without Random Oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 201–216. Springer, Heidelberg (2007)
9. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
10. Camenisch, J., Chandran, N., Shoup, V.: A Public Key Encryption Scheme Secure against Key Dependent Chosen Plaintext and Adaptive Chosen Ciphertext Attacks. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 351–368. Springer, Heidelberg (2009)
11. Camenisch, J., Dubovitskaya, M., Haralambiev, K.: Efficient Structure-Preserving Signature Scheme from Standard Assumptions. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 76–94. Springer, Heidelberg (2012)
12. Camenisch, J., Haralambiev, K., Kohlweiss, M., Lapon, J., Naessens, V.: Structure Preserving CCA Secure Encryption and Applications. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 89–106. Springer, Heidelberg (2011)
13. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
14. Cathalo, J., Libert, B., Yung, M.: Group Encryption: Non-interactive Realization in the Standard Model. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 179–196. Springer, Heidelberg (2009)

15. Chase, M., Kohlweiss, M.: A New Hash-and-Sign Approach and Structure-Preserving Signatures from DLIN. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 131–148. Springer, Heidelberg (2012)
16. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable Proof Systems and Applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 281–300. Springer, Heidelberg (2012)
17. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Efficient Public-Key Cryptography in the Presence of Key Leakage. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 613–631. Springer, Heidelberg (2010)
18. Fuchsbauer, G., Pointcheval, D.: Anonymous Proxy Signatures. In: Ostrovsky, R., De Prisco, R., Visconti, I. (eds.) SCN 2008. LNCS, vol. 5229, pp. 201–217. Springer, Heidelberg (2008)
19. Fuchsbauer, G., Pointcheval, D., Vergnaud, D.: Transferable Constant-Size Fair E-Cash. In: Garay, J.A., Miyaji, A., Otsuka, A. (eds.) CANS 2009. LNCS, vol. 5888, pp. 226–247. Springer, Heidelberg (2009)
20. Galbraith, S.D., Peterson, K.G., Smart, N.P.: Pairings for cryptographers. Discrete Applied Mathematics 156(16), 3113–3121 (2008)
21. Green, M., Hohenberger, S.: Universally Composable Adaptive Oblivious Transfer. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 179–197. Springer, Heidelberg (2008)
22. Green, M., Hohenberger, S.: Practical Adaptive Oblivious Transfer from Simple Assumptions. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 347–363. Springer, Heidelberg (2011)
23. 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)
24. Groth, J.: Fully Anonymous Group Signatures Without Random Oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007)
25. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
26. Hofheinz, D., Jager, T.: Tightly Secure Signatures and Public-Key Encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 590–607. Springer, Heidelberg (2012)
27. Hofheinz, D., Kiltz, E.: Secure Hybrid Encryption from Weakened Key Encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (2007)
28. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust Non-interactive Zero Knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Heidelberg (2001)
29. Shacham, H.: A Cramer-Shoup encryption scheme from the linear assumption and from progressively weaker linear variants. IACR ePrint Archive, 2007/074 (2007), `http://eprint.iacr.org`

Key Encapsulation Mechanisms from Extractable Hash Proof Systems, Revisited

Takahiro Matsuda and Goichiro Hanaoka

Research Institute for Secure Systems,
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{t-matsuda,hanaoka-goichiro}@aist.go.jp

Abstract. In CRYPTO 2010, Wee proposed the notion of "extractable hash proof systems" (XHPS), and its richer version, "all-but-one XHPS" (ABO-XHPS), and showed that chosen ciphertext secure (CCA secure) key encapsulation mechanisms (KEM) can be constructed from them. This elegantly explains several recently proposed practical KEMs constructed based on the "all-but-one" simulation paradigm in a unified framework. Somewhat frustratingly, however, there still exist popular KEMs whose construction and security proofs are not captured by this framework. In this paper, we revisit the framework of the ABO-XHPS-based KEM. Firstly, we show that to prove CCA security of the ABO-XHPS-based KEM, some requirements can be relaxed. This relaxation widens the applicability of the original framework, and explains why many known practical KEMs can be proved CCA secure. Moreover, we introduce new properties for ABO-XHPS, and show how one of the properties leads to KEMs that achieve "constrained" CCA security, which is a useful security notion of KEMs for obtaining CCA secure public key encryption via hybrid encryption. Thirdly, we investigate the relationships among computational properties that we introduce in this paper, and derive a useful theorem that enables us to understand the structure of KEMs of a certain type in a modular manner. Finally, we show that the ABO-XHPS-based KEM can be extended to efficient multi-recipient KEMs. Our results significantly extend the framework for constructing a KEM from ABO-XHPS, enables us to capture and explain more existing practical CCA secure schemes (most notably those based on the decisional Diffie-Hellman assumption) in the framework, and leads to a number of new instantiations of (single- and multi-recipient) KEMs.

Keywords: key encapsulation mechanism, extractable hash proof system, chosen ciphertext security, constrained chosen ciphertext security.

1 Introduction

Background and Motivation. Studies on constructing and understanding practical public key encryption (PKE) schemes secure against chosen ciphertext attacks (CCA security) [24,9] are important research themes in the area of cryptography. Among several approaches towards practical CCA secure PKE schemes, the promising approach is to construct a PKE scheme via the hybrid encryption methodologies using a key encapsulation mechanism (KEM) and a data encapsulation mechanism (DEM). Cramer and

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 332–351, 2013.

Shoup [8] show that if we combine a CCA secure KEM and a CCA secure DEM, then we obtain a hybrid PKE scheme which is CCA secure. Hofheinz and Kiltz [17] introduce a security notion called *constrained CCA* security (CCCA security), and show that a CCA secure PKE scheme can be constructed by combining a CCCA secure KEM and a DEM satisfying the security of (one-time) authenticated encryption [2]. These results enable us to concentrate on studying practical constructions of (C)CCA secure KEMs, for obtaining practical PKE schemes.

Seeing in a larger perspective, there are two general paradigms towards CCA secure PKE schemes: the first paradigm uses non-interactive proofs of "well-formedness" [10], which includes the constructions with non-interactive zero-knowledge proofs [22,9,25] that cover generic constructions from cryptographic primitives, and the constructions with *universal hash proof systems* [7,17] that cover practical and efficient schemes based on specific intractability of decision problems.; The second paradigm uses the so-called "all-but-one" simulation technique, (e.g. [3,5,19,17,23,12,18,27]). In fact, [9] can also be seen to be included in this paradigm. These two paradigms in fact cover almost all known constructions of CCA secure PKE schemes and KEMs. Our focus in this paper is on KEMs constructed based on the second paradigm.

In CRYPTO'10, Wee [27] introduced the notion of "*extractable hash proof systems*" (XHPS) and its richer version "*all-but-one XHPS*" (ABO-XHPS), which are both a special kind of non-interactive proof system for a family of *one-way relations* (which defines a hard search problem, such as the computational Diffie-Hellman problem), and showed that CCA secure KEMs can be constructed from them. This framework elegantly explains the constructions and the security proofs of several (variants of) recently proposed KEMs (e.g. [6,18]) based on hardness of "search" problems (not only "decision" problems), which are proved with the "all-but-one" simulation paradigm.

Somewhat frustratingly, however, there still exist several popular KEMs (e.g. [17,6,12]) whose construction and (C)CCA security are not explained by the framework in [27], although those that cannot be explained by the framework in [27] are quite similar to those that can be explained. The main motivation of this work is to extend the framework of KEMs based on ABO-XHPS to capture a wider class of constructions and security proofs of CCA secure, and even CCCA secure, KEMs, so that it works as a more general framework capturing a wider class of constructions based on the "all-but-one" simulation paradigm as we categorized above. Such general framework can be expected to lead to deeper understanding of constructions and security proofs of KEMs and be useful for future design of (C)CCA secure practical KEMs and PKE schemes, and higher level primitives/protocols that use those as building blocks.

Our Contribution. In this paper, we revisit and extend the framework for constructing a KEM based on ABO-XHPS in [27] in several different aspects:

Firstly, we show that to prove CCA security of the ABO-XHPS-based KEM, some requirement of ABO-XHPS and its associated one-way relation family can be relaxed. More specifically, the original definition of an ABO-XHPS in [27] requires some unnecessarily strong "correctness" requirement and a underlying one-way relation family with which the ABO-XHPS is associated needs to satisfy "*gap*"-type one-wayness, which requires that one-wayness holds even in the presence of the decision oracle, and thus is a stronger type of one-wayness. Instead, we show that as long as the ABO-XHPS

satisfies the property which we call *computational soundness* (CS security, for short), the ABO-XHPS-based KEM can be shown to be CCA secure with a weaker correctness requirement for the underlying ABO-XHPS and a weaker (non-gap) one-way relation. (The formal definitions of an ABO-XHPS and a family of one-way relations are given in Section 3.) Due to these relaxations, we can treat a wider class of computational assumptions, and the class of CCA secure KEMs that can be captured by the framework becomes significantly wider. Most notably, we can now treat the decisional Diffie-Hellman (DDH) assumption as a one-way relation family, and thus several practical DDH-based KEMs (e.g. [6,12]), which was not possible by the original framework because of the requirement of the "gap"-type one-wayness.

Secondly, we propose another computational property of ABO-XHPS which we call "*pseudorandom extraction property*" (PR−Ext security, for short), and show that if an ABO-XHPS satisfies the property, then the ABO-XHPS-based KEM achieves CCCA security. This result enables us to explain CCCA security of the KEMs whose construction and security proof can be understood in the "all-but-one" simulation paradigm. This enables us to cast CCCA secure KEMs proposed in [17] and in [13, Sect. 6] in our extended framework.

Thirdly, we study the computational properties of ABO-XHPS themselves. Specifically, we introduce yet another computational property which we call *weak computational soundness* (wCS security, for short), and show that wCS security is implied by both CS security and PR−Ext security. Furthermore, we show how to combine a PR−Ext secure ABO-XHPS and a wCS secure ABO-XHPS to obtain a CS secure ABO-XHPS. This "transformation," together with the above mentioned results, enables us to understand the constructions and CCA security of KEMs in a modular manner. For example, this provides us with an alternative security proof of the Cash et al. KEM [6, Sect. 5.2], without the "trapdoor test" theorem [6, Theorem 2] that was originally used to prove its CCA security. Moreover, combined with the above mentioned results, this result enables us to derive a number of new variants of KEMs [8,19,17,6,12] that can be shown to be CCA secure under the DDH or the Hashed DH (HDH) assumption [11].

Finally, we show that the ABO-XHPS-based KEM can be extended to be a multi-recipient KEM (MR-KEM) [26,16]. Here, by MR-KEM we mean the one formalized by Smart [26] in which all recipients recover a same session-key. (This differs from multi-recipient PKE by Bellare et al. [1] in which each receiver may recover different message.) From this result, we derive a number of new practical (C)CCA secure MR-KEMs.

The results in this paper are summarized in Fig. 1. Our results enable us to capture more existing practical CCA secure schemes than the original framework [27], derive a number of new practical instantiations of (C)CCA secure (MR-)KEMs, and understand the structures and security proofs of these schemes. (See Section 6 for more details.) We believe that the framework of ABO-XHPS extended by our results widely captures the constructions of KEMs based on the "all-but-one" simulation paradigm and leads to deeper understanding of the constructions and security proofs of practical KEMs, and is useful for future design of (C)CCA secure practical (MR-)KEMs.

Due to space limitations, the full proofs of the theorems in this paper will be given in the full version. We instead give intuitive explanations for each theorem.

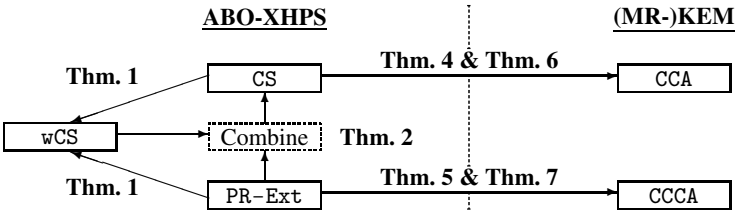


Fig. 1. Summary of our results. Each box with label "X" denotes an X-secure primitive. The arrow (X $\rightarrow$ Y) indicates that an X-secure primitive can be used to construct a Y-secure primitive.

Related Work. The relevant general framework of constructions of PKE schemes and KEMs is be the framework using *universal hash proof systems* introduced by Cramer and Shoup [7]. This framework, as we mentioned above, can be seen as one of the general paradigms using non-interactive proof of "well-formedness", and captures a wide class of practical constructions of PKE schemes and KEMs, such as Cramer-Shoup PKE scheme [8]. Kurosawa and Desmedt [20] showed how to construct CCA secure KEM directly from hash proof systems. The requirements in the original definition of a universal hash proof system in [7] (and in [20]) were all statistical (information-theoretic) ones. Hofheinz and Kiltz [17] introduced computational relaxation for a universal hash proof system, and showed that the KEM based on a hash proof system in [20] can be shown to be CCCA secure if the underlying hash proof system satisfies some computational property.

Wee [28] recently proposed the notion of *threshold extractable hash proof system*, which can be seen as a generalization of an ABO-XHPS, from "all-but-one" to "all-but-t." From it, he showed how to construct threshold signature schemes, threshold encryption schemes, and broadcast encryption schemes.

2 Preliminaries

In this section, we review the basic notation and the definitions for a (multi-recipient) KEM. Due to space limitation, the definitions for other basic primitives and computational intractability assumptions will be given in the full version.

Basic Notation. $\mathbb{N}$ denotes the set of all natural numbers, and if $n \in \mathbb{N}$ then $[n] = \{1, \ldots, n\}$. "$x \leftarrow y$" denotes that x is chosen uniformly at random from y if y is a finite set, or y is assigned to x otherwise. If S is a set, then "$|S|$" denotes its size. "PPTA" denotes a *probabilistic polynomial time algorithm*. Unless otherwise stated, k denotes the security parameter. If $\mathcal{A}$ is an algorithm and $\mathcal{O}$ is a function, then "$\mathcal{A}^{\mathcal{O}}$" denotes that $\mathcal{A}$ has oracle access to $\mathcal{O}$. A function $f(k) : \mathbb{N} \rightarrow [0, 1]$ is said to be *negligible* if for all positive polynomials $p(k)$ and all sufficiently large $k \in \mathbb{N}$, we have $f(k) < 1/p(k)$.

Multi-Recipient KEM. Here, we review the definition of a multi-recipient KEM (MR-KEM). We use the definition formalized by Smart [26], where all recipients recover a same session-key. A MR-KEM Γ consists of the following five PPTAs:

MSetup**:** The setup algorithm that takes 1^k as input, and outputs a set of public parameters pub. pub specifies the session-key space $\mathcal{K}$.

MKG**:** The (user's) key generation algorithm that takes pub as input, and outputs a public/secret key pair (pk, sk). Without loss of generality, we assume that the information on pub is contained in pk and sk, and we do not write pub for the inputs of the following algorithms.

MEnc**:** The encapsulation algorithm that takes a set of public keys $\mathbf{pk} = (pk_1, \ldots, pk_n)$ as input, and outputs a ciphertext c and a session-key $K \in \mathcal{K}$.

MExt**:** The (deterministic) user's ciphertext extraction algorithm that takes a user i's public key pk_i, and a ciphertext c (which is output from MEnc) as input, and outputs the user i's ciphertext c_i.

MDec**:** The (deterministic) decapsulation algorithm that takes a user i's secret key sk_i and a user i's ciphertext c_i as input, and outputs a session-key K which could be a special symbol $\perp$ meaning "invalid".

We say that a MR-KEM satisfies *correctness* (resp. *almost-correctness*), if for all $\mathsf{pub} \leftarrow \mathsf{MSetup}(1^k)$ and all polynomials $n = n(k)$, the following probability is zero (resp. negligible).

$$\Pr[\ (pk_i, sk_i) \leftarrow \mathsf{MKG}(\mathsf{pub}) \text{ for } i \in [n];\ (c, K) \leftarrow \mathsf{MEnc}(\mathbf{pk} = (pk_1, \ldots, pk_n)) : \mathsf{MDec}(sk_i, \mathsf{MExt}(pk_i, c)) \neq K \text{ for some } i \in [n]\ ]$$

Security Notions. Here, we recall the definitions of indistinguishability against chosen ciphertext attacks ($\mathtt{CCA}$ security) and against constrained chosen ciphertext attacks ($\mathtt{CCCA}$ security) [17].

Let $\mathtt{ATK} \in \{\mathtt{CCA}, \mathtt{CCCA}\}$ and $n \in \mathbb{N}$. For a MR-KEM $\Gamma = (\mathsf{MSetup}, \mathsf{MKG}, \mathsf{MEnc}, \mathsf{MExt}, \mathsf{MDec})$, we define the experiment $\mathsf{Expt}^{\mathtt{ATK}}_{\Gamma,\mathcal{A},n}(k)$ that an adversary $\mathcal{A}$ attacks Γ under the attack type $\mathtt{ATK}$ as follows:

$$\mathsf{Expt}^{\mathtt{ATK}}_{\Gamma,\mathcal{A},n}(k) : [\ \mathsf{pub} \leftarrow \mathsf{MSetup}(1^k);\ (pk_i, sk_i) \leftarrow \mathsf{MKG}(\mathsf{pub}) \text{ for } i \in [n];\ \mathbf{pk} \leftarrow (pk_1, \ldots, pk_n);\ (c^*, K_1^*) \leftarrow \mathsf{MEnc}(\mathbf{pk});\ K_0^* \leftarrow \mathcal{K};\ b \leftarrow \{0,1\};\ b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pub}, \mathbf{pk}, c^*, K_b^*);\ \text{If } b' = b \text{ then return } 1 \text{ else return } 0\ ],$$

where the oracle $\mathcal{O}$ is determined by $\mathtt{ATK}$ in the following ways: If $\mathtt{ATK} = \mathtt{CCA}$, then the oracle $\mathcal{O}$ is the decapsulation oracle $\mathcal{O}(\cdot,\cdot)$ which takes a user index/ciphertext pair (i, c) as input, and outputs the result of $\mathsf{tMDec}(sk_i, \mathsf{MExt}(pk_i, c))$. If $\mathtt{ATK} = \mathtt{CCCA}$ then the oracle $\mathcal{O}$ is the *constrained decapsulation (CDEC) oracle* $\mathcal{O}_{cdec}(\cdot,\cdot,\cdot)$, which takes a user index i, a predicate $\mathsf{pred} : \mathcal{K} \to \{0,1\}$, and a ciphertext c as input, and outputs a response that is calculated as follows:

$$\mathcal{O}_{cdec}(i, \mathsf{pred}, c) = \begin{cases} K & \text{If } \mathsf{MDec}(sk_i, \mathsf{MExt}(pk_i, c)) = K \neq \perp \wedge \mathsf{pred}(K) = 1 \\ \perp & \text{Otherwise} \end{cases}$$

Moreover, in both cases $\mathtt{ATK} \in \{\mathtt{CCA}, \mathtt{CCCA}\}$, $\mathcal{A}$ is not allowed to submit a query that contains a user index/ciphertext pair (i, c) satisfying $\mathsf{MExt}(pk_i, c) = \mathsf{MExt}(pk_i, c^*)$ to the oracle.

Let $\mathcal{A}$ be an adversary that runs in a CCCA experiment and makes in total q queries, and let $(i_j, \mathsf{pred}_j, c_j)$ be $\mathcal{A}$'s j-th CDEC query. "*The running time of $\mathcal{A}$ in the* CCCA *experiment*" is defined as the sum of $\mathcal{A}$'s running time and the total of the maximum running time for evaluating each pred_j submitted by $\mathcal{A}$. "*The running time of the* CCCA *experiment*" is defined as the total running time of $\mathsf{Expt}^{\mathtt{CCCA}}_{\Gamma,\mathcal{A}}(k)$ minus "the running time of $\mathcal{A}$ in the CCCA experiment." For a CCCA adversary $\mathcal{A}$ and an experiment $\mathcal{E}$ (not necessarily $\mathsf{Expt}^{\mathtt{CCCA}}_{\Gamma,\mathcal{A}}(k)$) that $\mathcal{A}$ runs in, we define the parameter called (plaintext) *uncertainty* $\mathsf{uncert}_{\mathcal{A},\mathcal{E}}(k)$ by:

$$\mathsf{uncert}_{\mathcal{A},\mathcal{E}}(k) = \frac{1}{q}\sum_{j\in[q]} \Pr[\mathcal{E}; K \leftarrow \mathcal{K} : \mathsf{pred}_j(K) = 1].$$

Finally, we say that an adversary $\mathcal{A}$ is a *valid* CCCA *adversary* if (1) "the running time of $\mathcal{A}$ in the CCCA experiment" is polynomial in k, and (2) $\mathsf{uncert}_{\mathcal{A},\mathcal{E}}(k)$ is negligible for all experiments $\mathcal{E}$ whose running time is at most that of "the running time of the CCCA experiment" that $\mathcal{A}$ runs in.

For a KEM Γ, an adversary $\mathcal{A}$, $\mathtt{ATK} \in \{\mathtt{CCA}, \mathtt{CCCA}\}$, and $n \in \mathbb{N}$ we define ATK advantage $\mathsf{Adv}^{\mathtt{ATK}}_{\Gamma,\mathcal{A},n}(k)$ of $\mathcal{A}$ by $\mathsf{Adv}^{\mathtt{ATK}}_{\Gamma,\mathcal{A},n}(k) = |\Pr[\mathsf{Expt}^{\mathtt{ATK}}_{\Gamma,\mathcal{A},n}(k) = 1] - 1/2|$.

Definition 1. *We say that a MR-KEM Γ is* CCA *secure if $\mathsf{Adv}^{\mathtt{CCA}}_{\Gamma,\mathcal{A},n}(k)$ is negligible for any PPTA $\mathcal{A}$ and any polynomial $n = n(k)$. Furthermore, we say that a MR-KEM Γ is* CCCA *secure if $\mathsf{Adv}^{\mathtt{CCCA}}_{\Gamma,\mathcal{A},n}(k)$ is negligible for any valid* CCCA *adversary $\mathcal{A}$ and any polynomial $n = n(k)$.*

Single-Recipient KEM. When we talk about ordinary "single-recipient" KEMs, we need not consider the setup and user key generation algorithms separately. Therefore, in order to clarify the difference between multi-recipient KEMs and ordinary KEMs, we write the key generation, the encapsulation, and the decapsulation algorithms of a single-recipient KEM by KG, Enc, and Dec, respectively (without the prefix "M"). The syntax and the security notions for single-recipient KEMs are defined similarly to those of MR-KEMs.

3 Definitions for All-But-One Extractable Hash Proof Systems

In this section, we define an ABO-XHPS and one-way relations which are necessary for ABO-XHPS, following the definitions in [27]. However, our definitions here are slightly different from ones in [27], and we also highlight the difference.

3.1 One-Way Relation Families

A family of relations (relation family, for short) $\mathcal{R}$ (that supports a PRG) is associated with the following three PPTAs (RSetup, RSamp, G):

RSetup**:** The setup algorithm that takes 1^k as input, and outputs a public/private parameter pair $(\mathsf{pub}, \mathsf{pri})$. pub contains the description of sets $\mathcal{U}$, $\mathcal{S}$, $\mathcal{W}$, and $\mathcal{K}$, from which we can efficiently sample elements uniformly. pub also fixes one relation

$\mathcal{R}_{\mathsf{pub}}$ over $\mathcal{U} \times \mathcal{S}$. We require that: (1) for all u, there is at most one s such that $(u, s) \in \mathcal{R}_{\mathsf{pub}}$ (with overwhelming probability over the choice of pub), and (2) given pri (corresponding to pub) and $(u, s) \in \mathcal{U} \times \mathcal{S}$, whether $(u, s) \in \mathcal{R}_{\mathsf{pub}}$ or not is efficiently decidable. For notational convenience, we assume that pub is provided as input to the following algorithms, and do not write it explicitly.

RSamp**:** The sampling algorithm that (takes pub as input, and) outputs a pair $(u, s) \in \mathcal{R}_{\mathsf{pub}}$ so that u is distributed uniformly over $\mathcal{U}$. The randomness space of RSamp is $\mathcal{W}$, and when we need to make the randomness used to sample (u, s) explicit, we write this process as "$(u, s) \leftarrow \mathsf{RSamp}(w)$" (in this case, RSamp is treated as a deterministic algorithm).

G**:** The (pseudorandom) generator that takes (pub and) an element $s \in \mathcal{S}$ as input, and outputs $K \in \mathcal{K}$.

Hereafter, we identify a relation family $\mathcal{R}$ with the associated PPTAs (RSetup, RSamp, G), and in particular, write $\mathcal{R} = (\mathsf{RSetup}, \mathsf{RSamp}, \mathsf{G})$.

Definition 2. *We say that* $\mathcal{R} = (\mathsf{RSetup}, \mathsf{RSamp}, \mathsf{G})$ *is a* one-way relation family *if the advantage* $\mathsf{Adv}^{\mathtt{PRG}}_{\mathcal{R},\mathcal{A}}(k)$ *defined below is negligible for any PPTA* $\mathcal{A}$*:*

$$\mathsf{Adv}^{\mathtt{PRG}}_{\mathcal{R},\mathcal{A}}(k) = |\Pr[(\mathsf{pub}, \mathsf{pri}) \leftarrow \mathsf{RSetup}(1^k); (u, s) \leftarrow \mathsf{RSamp}; K_1^* \leftarrow \mathsf{G}(s); K_0^* \leftarrow \mathcal{K}; b \leftarrow \{0,1\}; b' \leftarrow \mathcal{A}(\mathsf{pub}, u, K_b^*) : b' = b] - \frac{1}{2}|.$$

Furthermore, we say that $\mathcal{R}$ *is a* gap one-way relation family *if the advantage is negligible for any PPTA adversary that is given access to the "relation" oracle which takes* $(u, s) \in \mathcal{U} \times \mathcal{S}$ *as input and tells if* $(u, s) \in \mathcal{R}_{\mathsf{pub}}$ *or not.*

Difference from the Definition in [27]. The original definition of one-way relation families in [27] is the "gap" version here. The definition of (non-gap-)one-way relation family is clearly weaker, thus potentially easier to achieve and captures wider class of relation families than the gap version. For example, the "gap" one-way relation of [27] does not capture the HDH-based Diffie-Hellman relation family we introduce below.[1]

Concrete Example of One-Way Relation Families: Diffie-Hellman Relation. Let $\mathbb{G}$ be a group of prime order p and let $H : \mathbb{G} \to \mathcal{K}$ be a hash function. We say that the hashed Diffie-Hellman (HDH) assumption holds in $(\mathbb{G}, H)$ if the distributions of $(g, g^a, g^b, H(g^{ab}))$ and (g, g^a, g^b, K) are computationally indistinguishable, where $g \in \mathbb{G}$, $a, b \in \mathbb{Z}_p$, and $K \in \mathcal{K}$ are chosen randomly.[2]

The Diffie-Hellman relation family (that supports a PRG H) $\mathcal{R}^{\mathsf{DH}}$, indexed by $\mathsf{pub} = (g, g^\alpha) \in (\mathbb{G})^2$, is defined by $\mathcal{R}^{\mathsf{DH}}_{(g,g^\alpha)} = \{(u, s) \in (\mathbb{G})^2 | s = u^\alpha\}$. The associated algorithms $(\mathsf{RSetup}, \mathsf{RSamp}, \mathsf{G})$ are as follows: RSetup sets $\mathcal{U} = \mathcal{S} = \mathbb{G}$ and $\mathcal{W} = \mathbb{Z}_p$,

[1] We note that in [28], Wee introduced the definition of one-way relation families in the same sense as the one defined here.

[2] The DDH assumption is the special case of the HDH assumption in which H is the identity function. It is possible that the DDH assumption in $\mathbb{G}$ is false while the HDH assumption in $(\mathbb{G}, H)$ holds for some H. For more details about the HDH assumption, see [11,19,6,12] and the full version of this paper.

picks random elements $g \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, and sets $\mathsf{pub} = (g, h) = (g, g^\alpha)$ and $\mathsf{pri} = \alpha$. $\mathsf{RSamp}(w) := (g^w, h^w)$. $\mathsf{G}(s) := H(s)$. It is straightforward to see that $\mathcal{R}^{\mathsf{DH}}$ is a one-way relation family under the HDH assumption in $(\mathbb{G}, H)$.

3.2 All-But-One Extractable Hash Proof Systems

An ABO-XHPS is always associated with a relation family. Thus, for notational convenience, we denote by "$\mathcal{X}^{\mathcal{R}}$" an ABO-XHPS $\mathcal{X}$ associated with a relation family $\mathcal{R}$. (If $\mathcal{R}$ is clear from the context, we often omit $\mathcal{R}$ and just write $\mathcal{X}$.) Informally, an ABO-XHPS is a special type of "designated-verifier non-interactive zero-knowledge proof of knowledge," and it has, as its internal structure, a family of "tag-based" hash functions $\mathsf{H}_{pk} : \mathcal{T} \times \mathcal{U} \to \{0,1\}^*$ indexed by a public key pk (where $\mathcal{T}$ is the tag space) which represents the relation of an instance $u \in \mathcal{U}$ and a (tag-based) "proof" $\pi = \mathsf{H}_{pk}(\mathsf{tag}, u)$ (with some $\mathsf{tag} \in \mathcal{T}$). If π is in a valid form, we can "extract" the answer s to the instance u satisfying $(u, s) \in \mathcal{R}_{\mathsf{pub}}$, using the secret key corresponding to pk. It is possible that H itself is not efficiently computable. Furthermore, $\mathcal{X}$ has "simulation" algorithms for key generation, extraction, and generating a proof. The first two algorithms work normally as above, except for one particular tag tag^* (used for the simulated key generation process) under which one can generate a valid proof without a witness (hence the name "all-but-one").

Formally, an ABO-XHPS $\mathcal{X}$, associated with a relation family $\mathcal{R} = (\mathsf{RSetup}, \mathsf{RSamp}, \mathsf{G})$, consists of six PPTAs (XKG, Pub, Ext, $\widehat{\mathsf{XKG}}$, $\widehat{\mathsf{Priv}}$, $\widehat{\mathsf{Ext}}$) that satisfy the following "functional requirements" (correctness) with overwhelming probability over the choice of $(\mathsf{pub}, \mathsf{pri}) \leftarrow \mathsf{RSetup}(1^k)$:

Extraction Mode. For all $(pk, sk) \leftarrow \mathsf{XKG}(\mathsf{pub}, \mathsf{pri})$ and all tuples (tag, u, π): If $\pi = \mathsf{H}_{pk}(\mathsf{tag}, u)$ then $(u, \mathsf{Ext}(sk, \mathsf{tag}, u, \pi)) \in \mathcal{R}_{\mathsf{pub}}$, and if $\pi \neq \mathsf{H}_{pk}(\mathsf{tag}, u)$ then $\mathsf{Ext}(sk, \mathsf{tag}, u, \pi) = \perp$.

All-But-One Mode. For all tag^* and all $(pk, \widehat{sk}) \leftarrow \widehat{\mathsf{XKG}}(\mathsf{pub}, \mathsf{tag}^*)$:
Private Evaluation under tag^*: For all $(u, s) \in \mathcal{R}_{\mathsf{pub}}$: $\widehat{\mathsf{Priv}}(\widehat{sk}, u) = \mathsf{H}_{pk}(\mathsf{tag}^*, u)$.
Extraction: For all $\mathsf{tag} \neq \mathsf{tag}^*$ and all (u, π): If $\pi = \mathsf{H}_{pk}(\mathsf{tag}, u)$ then $(u, s) \in \mathcal{R}_{\mathsf{pub}}$, where $s = \widehat{\mathsf{Ext}}(\widehat{sk}, \mathsf{tag}, u, \pi)$. (The case of $\pi \neq \mathsf{H}_{pk}(\mathsf{tag}, u)$ is unspecified.)

Public Evaluation. For all pk (output from either XKG or $\widehat{\mathsf{XKG}}$), tag, and $(u, s) = \mathsf{RSamp}(w)$: $\mathsf{Pub}(pk, \mathsf{tag}, w) = \mathsf{H}_{pk}(\mathsf{tag}, u)$.

Indistinguishability of Two Modes. For all tag^*, the two distributions, $\{(pk, sk) \leftarrow \mathsf{XKG}(\mathsf{pub}, \mathsf{pri}) : pk\}$ and $\{(pk, \widehat{sk}) \leftarrow \widehat{\mathsf{XKG}}(\mathsf{pub}, \mathsf{tag}^*) : pk\}$, are statistically indistinguishable.

In this paper, we also consider a slight relaxation of the extraction property of the "all-but-one" mode. We say that an ABO-XHPS satisfies *almost-correctness* if for all $(\mathsf{pub}, \mathsf{pri}) \leftarrow \mathsf{RSetup}(1^k)$, all $(u, s) = \mathsf{RSamp}(w)$, and all $(\mathsf{tag}, \mathsf{tag}^*)$ such that $\mathsf{tag} \neq \mathsf{tag}^*$, the following probability is overwhelming: $\Pr[(pk, \widehat{sk}) \leftarrow \widehat{\mathsf{XKG}}(\mathsf{pub}, \mathsf{tag}^*) : \widehat{\mathsf{Ext}}(\widehat{sk}, \mathsf{tag}, u, \mathsf{H}_{pk}(\mathsf{tag}, u)) = s]$.

We note that the indistinguishability of the two modes implies that the information on a tag tag^* is statistically hidden from pk output from $\widehat{\mathsf{XKG}}(\mathsf{pub}, \mathsf{tag}^*)$.

Difference from the Definition in [27]. Here, we explain the difference of our definition of ABO-XHPS and the definition by Wee [27, Sect. 3.4]. Firstly, XKG algorithm in [27] does not take the private parameter pri as input (while ours does). However, this restriction is unnecessary for proving (C)CCA security of the ABO-XHPS-based KEM, and thus we allow XKG to take pri as input.

Secondly, the correctness requirements of Ext and $\widehat{\mathsf{Ext}}$ algorithms in [27] are defined in an "if-and-only-if" style. More specifically, the correctness requirements of Ext and $\widehat{\mathsf{Ext}}$ algorithms in [27] are: (i) "$\pi = \mathsf{H}_{pk}(\mathsf{tag}, u) \Leftrightarrow (u, \mathsf{Ext}(sk, \mathsf{tag}, u, \pi)) \in \mathcal{R}_{\mathsf{pub}}$," and (ii) "$\pi = \mathsf{H}_{pk}(\mathsf{tag}, u) \Leftrightarrow (u, \widehat{\mathsf{Ext}}(\widehat{sk}, \mathsf{tag}, u, \pi)) \in \mathcal{R}_{\mathsf{pub}}$." Regarding (i), since the definition of [27] does not specify what is output from Ext when $\mathsf{H}_{pk}(\mathsf{tag}, u) \neq \pi$, we require that it output $\perp$. We stress that this is without loss of generality because given pri, it is possible to tell whether $(u, \mathsf{Ext}(sk, \mathsf{tag}, u, \pi)) \in \mathcal{R}_{\mathsf{pub}}$ or not, and pri can be contained in sk in our definition. The main difference from the definition in this paper and the one in [27] is regarding (ii), i.e. correctness of $\widehat{\mathsf{Ext}}$ algorithm. It is clear that ours requires weaker correctness since we do not specify the behavior of $\widehat{\mathsf{Ext}}$ in case $\mathsf{H}_{pk}(\mathsf{tag}, u) \neq \pi$, while the definition in [27] does. As will be shown later, this relaxation is the main reason that makes the framework of the ABO-XHPS-based KEM much wider, and makes it possible to capture most known practical CCA secure KEMs, and even CCCA secure schemes.

4 Computational Properties of ABO-XHPS

In this section, we introduce three computational properties of ABO-XHPS which are all related to the behavior of the extraction algorithm for the all-but-one mode, i.e. $\widehat{\mathsf{Ext}}$, and play important roles for proving (C)CCA security of the ABO-XHPS-based KEMs in the next section. We also show the relationships among these properties.

4.1 Computational Soundness (CS)

"*Computational soundness*" (CS security) captures soundness of the $\widehat{\mathsf{Ext}}$ algorithm, and roughly means that it is hard to find an "invalid proof" π from which $\widehat{\mathsf{Ext}}$ extracts some value that is not $\perp$. This is, it is hard to find a tuple (tag, u, π) satisfying $\mathsf{tag} \neq \mathsf{tag}^*$, $\mathsf{H}_{pk}(\mathsf{tag}, u) \neq \pi$, and $\widehat{\mathsf{Ext}}(\widehat{sk}, \mathsf{tag}, u, \pi) \neq \perp$, where $(pk, sk) \leftarrow \widehat{\mathsf{XKG}}(\mathsf{pub}, \mathsf{tag}^*)$. Formally, consider the experiment $\mathsf{Expt}^{\mathsf{CS}}_{\mathcal{X},\mathcal{A}}(k)$ that an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ runs in as in Fig. 2 (top-left).

Definition 3. *We say that an ABO-XHPS $\mathcal{X}$ satisfies* computational soundness (CS *secure, for short*)*, if the advantage* $\mathsf{Adv}^{\mathsf{CS}}_{\mathcal{X},\mathcal{A}}(k) = \Pr[\mathsf{Expt}^{\mathsf{CS}}_{\mathcal{X},\mathcal{A}}(k) = 1]$ *is negligible for any PPTA $\mathcal{A}$.*

Concrete CS *Secure ABO-XHPS.* The factoring-based ABO-XHPS [27, Sect. 4.2], the (non-twin-)Diffie-Hellman-based one [27, Sect. 5.1] in case instantiated with bilinear groups, and the twin Diffie-Hellman-based one [27, Sect. 5,2] shown by Wee, are in fact all CS secure. The $\widehat{\mathsf{Ext}}$ algorithm of these ABO-XHPS satisfy the "if-and-only-if"-style

$\mathsf{Expt}^{\mathtt{CS}}_{\mathcal{X},\mathcal{A}}(k)$:
$(\mathsf{pub},\mathsf{pri}) \leftarrow \mathsf{RSetup}(1^k)$;
$(\mathsf{tag}^*,\mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{pub})$;
$(pk,\widehat{sk}) \leftarrow \widehat{\mathsf{XKG}}(\mathsf{pub},\mathsf{tag}^*)$;
$\mathcal{A}_2^{\mathcal{O}_{\mathtt{CS}}}(pk,\mathsf{st})$;
If $\mathcal{A}_2$ submits to oracle $\mathcal{O}_{\mathtt{CS}}$
at least one query
(tag',u',π') such that
$\mathsf{tag}' \neq \mathsf{tag}^*$
$\wedge\ \mathsf{H}_{pk}(\mathsf{tag}',u') \neq \pi'$
$\wedge\ \widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag}',u',\pi') \neq \bot$
then return 1 else return 0

The oracle in $\mathsf{Expt}^{\mathtt{CS}}_{\mathcal{X},\mathcal{A}}(k)$:
$$\mathcal{O}_{\mathtt{CS}}(\mathsf{tag},u,\pi) = \begin{cases} \widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\pi) & \text{If } \mathsf{tag} \neq \mathsf{tag}^* \\ \bot & \text{Otherwise} \end{cases}$$

The oracle in $\mathsf{Expt}^{\mathtt{PR\text{-}Ext}}_{\mathcal{X},\mathcal{A}}(k)$ and $\mathsf{Expt}^{\mathtt{wCS}}_{\mathcal{X},\mathcal{A}}(k)$:
$$\mathcal{O}_{\mathtt{PR\text{-}Ext}}(\mathsf{tag},u,\pi) = \mathcal{O}_{\mathtt{wCS}}(\mathsf{tag},u,\pi) = \begin{cases} \widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\pi) & \text{If } \mathsf{tag} \neq \mathsf{tag}^* \wedge \mathsf{H}_{pk}(\mathsf{tag},u) = \pi \\ \bot & \text{Otherwise} \end{cases}$$

$\mathsf{Expt}^{\mathtt{PR\text{-}Ext}}_{\mathcal{X},\mathcal{A}}(k)$:
$(\mathsf{pub},\mathsf{pri}) \leftarrow \mathsf{RSetup}(1^k)$;
$(\mathsf{tag}^*,\mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{pub})$;
$(pk,\widehat{sk}) \leftarrow \widehat{\mathsf{XKG}}(\mathsf{pub},\mathsf{tag}^*)$;
$(\mathsf{tag}',u',\pi',\mathsf{st}') \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathtt{PR\text{-}Ext}}}(pk,\mathsf{st})$;
$s'_1 \leftarrow \widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag}',u',\pi')$;
$s'_0 \leftarrow \mathcal{S}$;
$b \leftarrow \{0,1\}$;
$b' \leftarrow \mathcal{A}_3(s'_b,\mathsf{st}')$;
If $b' = b$ then return 1 else return 0

$\mathsf{Expt}^{\mathtt{wCS}}_{\mathcal{X},\mathcal{A}}(k)$:
$(\mathsf{pub},\mathsf{pri}) \leftarrow \mathsf{RSetup}(1^k)$;
$(\mathsf{tag}^*,\mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{pub})$;
$(pk,\widehat{sk}) \leftarrow \widehat{\mathsf{XKG}}(\mathsf{pub},\mathsf{tag}^*)$;
$(\mathsf{tag}',u',\pi',s') \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathtt{wCS}}}(pk,\mathsf{st})$;
If $\mathsf{tag}' \neq \mathsf{tag}^*$
$\wedge\ \mathsf{H}_{pk}(\mathsf{tag}',u') \neq \pi'$
$\wedge\ s' = \widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag}',u',\pi')$
$= \widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag}',u',\mathsf{H}_{pk}(\mathsf{tag}',u'))$
then return 1 else return 0

Fig. 2. The CS experiment (top-left), the PR–Ext experiment (bottom-left), the wCS experiment (bottom-right), and the definitions of the oracles (top-right)

correctness, and additionally have the property that invalid proofs $\pi \neq \mathsf{H}_{pk}(\mathsf{tag},u)$ can be detected publicly or by using a secret key of the ABO-XHPS. Furthermore, the recently proposed practical CCA secure KEMs based on the HDH and the DBDH assumptions can be understood as CS secure ABO-XHPS. These include (a simplified version of) the KEM in [5], [6, Sect. 5.2], and [13, Sect. 4]. Concretely, here we show the ABO-XHPS $\mathcal{X}_{\mathtt{CKS}}$ based on the KEM by Cash et al. [6, Sect. 5.2], which is associated with the HDH-based Diffie-Hellman relation family $\mathcal{R}^{\mathsf{DH}}$, as in Fig. 3. $\mathcal{X}_{\mathtt{CKS}}$ can be proved to be CS secure because the truth value of the validity check in the $\widehat{\mathsf{Ext}}$ algorithm of $\mathcal{X}_{\mathtt{CKS}}$ is the same as the truth value of the validity check in the Ext algorithm with overwhelming probability, due to the "trapdoor test" [6, Theorem 2]. In the full version, we also show ABO-XHPS based on the KEMs in [5] and [13, Sect. 4].

4.2 Pseudorandom Extraction Property (PR–Ext)

The "*pseudorandom extraction property*" (PR–Ext security) guarantees that if the $\widehat{\mathsf{Ext}}$ algorithm is given (tag,u,π) such that $\mathsf{H}_{pk}(\mathsf{tag},u) \neq \pi$ and $\mathsf{tag} \neq \mathsf{tag}^*$, then the extracted value $s = \widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\pi)$ looks pseudorandom. In the context of the ABO-XHPS-based KEMs (that will be shown later), this property means that when $c = (u,\pi)$ is an inconsistent ciphertext, if we extract s from $\widehat{\mathsf{Ext}}$, then the seed s of the session-key

$\mathsf{XKG}(\mathsf{pub}=(g,h),\mathsf{pri}=\alpha)$: $x,y_1,y_2 \leftarrow \mathbb{Z}_p;\ X \leftarrow g^x$ $Y_i \leftarrow g^{y_i}$ for $i \in [2]$ $pk \leftarrow (g,h,X,Y_1,Y_2)$ $sk \leftarrow (\alpha,x,y_1,y_2)$ Return (pk,sk)	$\widehat{\mathsf{XKG}}(\mathsf{pub}=(g,h),\mathsf{tag}^*)$: $z',z_1,z_2,z_3 \leftarrow \mathbb{Z}_p;\ X \leftarrow g^{z'}h^{-\mathsf{tag}^*}$ $Y_1 \leftarrow g^{z_1}h^{-z_2};\ Y_2 \leftarrow g^{z_3}Y_1^{-\mathsf{tag}^*}$ $pk \leftarrow (g,h,X,Y_1,Y_2)$ $\widehat{sk} \leftarrow (z',z_1,z_2,z_3,\mathsf{tag}^*)$ Return $(pk,\widehat{sk})$
$\mathsf{Pub}(pk,\mathsf{tag},w)$: $\pi_1 \leftarrow (h^{\mathsf{tag}}X)^w;\ \pi_2 \leftarrow (Y_1^{\mathsf{tag}}Y_2)^w$ Return $\pi \leftarrow (\pi_1,\pi_2)$	$\widehat{\mathsf{Priv}}(\widehat{sk},u)$: $\pi_1 \leftarrow u^{z'};\ \pi_2 \leftarrow u^{z_3}$ Return $\pi \leftarrow (\pi_1,\pi_2)$
$\mathsf{Ext}(sk,\mathsf{tag},u,\pi)$: If $u^{\alpha\cdot\mathsf{tag}+x}=\pi_1$ and $u^{y_1\cdot\mathsf{tag}+y_2}=\pi_2$ then return $s \leftarrow u^\alpha$ else return $\perp$	$\widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\pi)$: $s \leftarrow (\pi_1\cdot u^{-z'})^{\frac{1}{\mathsf{tag}-\mathsf{tag}^*}};\ s' \leftarrow (\pi_2\cdot u^{-z_3})^{\frac{1}{\mathsf{tag}-\mathsf{tag}^*}}$ If $s^{z_2}s'=u^{z_1}$ then return s else return $\perp$

Fig. 3. The CS secure ABO-XHPS $\mathcal{X}_{\mathtt{CKS}}$. The internal hash function family is defined by $\mathsf{H}_{pk}(\mathsf{tag},u)=((h^{\mathsf{tag}}X)^w,(Y_1^{\mathsf{tag}}Y_2)^w)$ where $u=g^w$.

$K=\mathsf{G}(s)$ looks like a uniformly random value. This property is like *computational universal*$_2$ [17] for a "Cramer-Shoup" type HPS [7], and plays a key role for showing CCCA security of the ABO-XHPS-based KEMs that will be given in the next section. Formally, consider the experiment $\mathsf{Expt}^{\mathtt{PR\text{-}Ext}}_{\mathcal{X},\mathcal{A}}(k)$ that an adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2,\mathcal{A}_3)$ runs in as in Fig. 2 (bottom-left). In the experiment, it is required that (tag',u',π') in $\mathcal{A}_2$'s output satisfy $\mathsf{tag}' \neq \mathsf{tag}^*$ and $\mathsf{H}_{pk}(\mathsf{tag}',u') \neq \pi'$.

Definition 4. *We say that an ABO-XHPS $\mathcal{X}$ has the* pseudorandom extraction *property* (PR-Ext *secure, for short*)*, if the advantage* $\mathsf{Adv}^{\mathtt{PR\text{-}Ext}}_{\mathcal{X},\mathcal{A}}(k)=|\Pr[\mathsf{Expt}^{\mathtt{PR\text{-}Ext}}_{\mathcal{X},\mathcal{A}}(k)=1]-1/2|$ *is negligible for any PPTA $\mathcal{A}$.*

Concrete PR-Ext *Secure ABO-XHPS.* Here, we show a concrete ABO-XHPS based on the KEM by Hofheinz and Kiltz [17] and the KEM by Hanaoka and Kurosawa [13, Sect. 6], both of which are associated with the HDH-based Diffie-Hellman relation $\mathcal{R}^{\mathsf{DH}}$. The ABO-XHPS $\mathcal{X}_{\mathtt{HoKi}}$ based on [17] and the ABO-XHPS $\mathcal{X}_{\mathtt{HaKu}}$ based on [13, Sect. 6] are constructed as in Fig. 4. $\mathcal{X}_{\mathtt{HoKi}}$ can be proved PR-Ext secure roughly because the value z_2 generated in $\widehat{\mathsf{XKG}}$ is information-theoretically hidden from pk and values s extracted from a "correct" proof $\pi=\mathsf{H}_{pk}(\mathsf{tag},u)$ using $\widehat{\mathsf{Ext}}$, while it appears in a value s extracted from an "invalid proof π satisfying $\pi \neq \mathsf{H}_{pk}(\mathsf{tag},u)$ and makes the extracted value s look like a random value in $\mathbb{G}$. The value β generated in $\widehat{\mathsf{XKG}}$ of $\mathcal{X}_{\mathtt{HaKu}}$ plays a similar role. We also note that $\mathcal{X}_{\mathtt{HaKu}}$ satisfies only almost-correctness, as $\widehat{\mathsf{Ext}}$ cannot extract a value when $\mathsf{tag}=\beta$. However, it suffices for showing CCCA security of the ABO-XHPS-based KEM shown in the next section.

4.3 Weak Computational Soundness (wCS)

"*Weak computational soundness*" (wCS security) guarantees that it is hard to find an "invalid" proof $\pi \neq \mathsf{H}_{pk}(\mathsf{tag},u)$ such that if we extract a value s with $\widehat{\mathsf{Ext}}$ from the invalid π, then the value s is the same as the value that is extracted from a "correct" proof

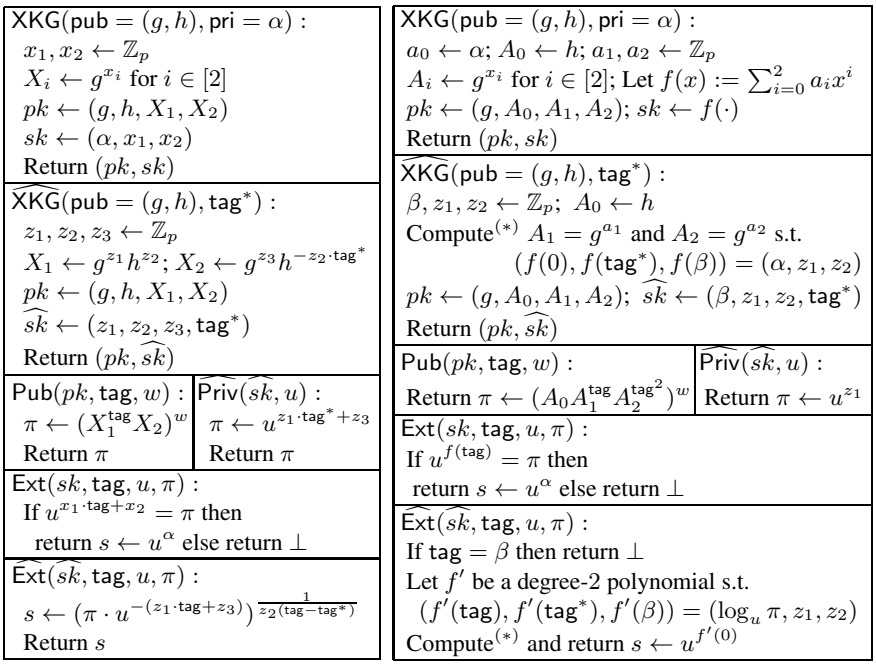


Fig. 4. The PR–Ext secure ABO-XHPS $\mathcal{X}_{\mathtt{HoKi}}$ (left) and $\mathcal{X}_{\mathtt{HaKu}}$ (right). The internal hash function family of $\mathcal{X}_{\mathtt{HoKi}}$ is defined by $\mathsf{H}_{pk}(\mathsf{tag},u)=(X_1^{\mathsf{tag}}X_2)^w$, and that of $\mathcal{X}_{\mathtt{HaKu}}$ is defined by $\mathsf{H}_{pk}(\mathsf{tag},u)=(A_0A_1^{\mathsf{tag}}A_2^{\mathsf{tag}^2})^w$, where $u=g^w$. $^{(*)}$ In $\mathcal{X}_{\mathtt{HaKu}}$, The values A_1 and A_2 in $\widehat{\mathsf{XKG}}$ and the value $u^{f'(0)}$ in $\widehat{\mathsf{Ext}}$ can be computed by Lagrange interpolation in the exponent [13].

$\pi' = \mathsf{H}_{pk}(\mathsf{tag},u)$. Formally, consider the experiment $\mathsf{Expt}^{\mathtt{wCS}}_{\mathcal{X},\mathcal{A}}(k)$ that an adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ runs in as in Fig. 2 (bottom-right).

Definition 5. *We say that an ABO-XHPS $\mathcal{X}$ satisfies* weak computational soundness (wCS *secure, for short*)*, if the advantage* $\mathsf{Adv}^{\mathtt{wCS}}_{\mathcal{X},\mathcal{A}}(k)=\Pr[\mathsf{Expt}^{\mathtt{wCS}}_{\mathcal{X},\mathcal{A}}(k)=1]$ *is negligible for any PPTA $\mathcal{A}$.*

We show that wCS security is indeed weaker than both CS and PR–Ext security.

Theorem 1. *Let $\mathcal{R}$ be a relation family and let $\mathcal{X}$ be an ABO-XHPS associated with $\mathcal{R}$. Assume that $\mathcal{R}$ is a one-way relation family, and $\mathcal{X}$ is either* CS *secure or* PR–Ext *secure. Then $\mathcal{X}$ is* wCS *secure.*

Intuition. If $\mathcal{X}$ is CS secure, then it is hard to find an invalid proof $\pi \neq \mathsf{H}_{pk}(\mathsf{tag},u)$ from which we can extract some value that is not $\perp$, and thus wCS security is satisfied. If $\mathcal{X}$ is PR–Ext secure, then an extracted value s from an invalid proof $\pi \neq \mathsf{H}_{pk}(\mathsf{tag},u)$ is pseudorandom, which will be different from the value $\widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\mathsf{H}_{pk}(\mathsf{tag},u))$ with overwhelming probability, and thus wCS security is satisfied.

Concrete wCS *Secure ABO-XHPS.* By definition, any ABO-XHPS whose $\widehat{\mathsf{Ext}}$ algorithm satisfies the "if-and-only-if"-style correctness of [27], is automatically wCS secure (and

$\mathsf{XKG}(\mathsf{pub}=(g,h),\mathsf{pri}=\alpha)$: $x \leftarrow \mathbb{Z}_p$; $X \leftarrow g^x$ Return $pk \leftarrow (g,h,X)$ and $sk \leftarrow (\alpha, x)$	$\widehat{\mathsf{XKG}}(\mathsf{pub}=(g,h),\mathsf{tag}^*)$: $z \leftarrow \mathbb{Z}_p$; $X \leftarrow g^z h^{-\mathsf{tag}^*}$ Return $pk \leftarrow (g,h,X)$ and $\widehat{sk} \leftarrow (z,\mathsf{tag}^*)$
$\mathsf{Pub}(pk,\mathsf{tag},w)$: Return $\pi \leftarrow (h^{\mathsf{tag}}X)^w$	$\widehat{\mathsf{Priv}}(\widehat{sk},u)$: Return $\pi \leftarrow u^z$
$\mathsf{Ext}(sk,\mathsf{tag},u,\pi)$: If $u^{\alpha\cdot\mathsf{tag}+x}=\pi$ then return $s \leftarrow u^\alpha$ else return $\perp$	$\widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\pi)$: Return $s \leftarrow (\pi \cdot u^{-z})^{\frac{1}{\mathsf{tag}-\mathsf{tag}^*}}$

Fig. 5. The wCS secure ABO-XHPS $\mathcal{X}_{\mathtt{Kiltz}}$. The internal hash function family is defined by $\mathsf{H}_{pk}(\mathsf{tag},u)=(h^{\mathsf{tag}}X)^w$ where $u=g^w$.

hence all XHPS shown in [27] is wCS secure). Here, we show another concrete example of a wCS secure ABO-XHPS, which is based on the KEM by Kiltz [19] and is associated with the Diffie-Hellman relation family $\mathcal{R}^{\mathsf{DH}}$. (This is a variant of the (non-twin-)Diffie-Hellman-based ABO-XHPS in [27, Sect. 5.1].) Specifically, the example of the ABO-XHPS, which we call $\mathcal{X}_{\mathtt{Kiltz}}$, is as in Fig. 5. $\mathcal{X}_{\mathtt{Kiltz}}$ can be shown to be wCS secure because there is no tuple (tag,u,π,s) that satisfies the winning condition of an adversary $\mathcal{A}$ in the wCS experiment. Namely, if $\mathsf{tag}\neq\mathsf{tag}^*$ and $\pi\neq\mathsf{H}_{pk}(\mathsf{tag},u)$, then it is guaranteed that $\widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\pi)\neq\widehat{\mathsf{Ext}}(\widehat{sk},\mathsf{tag},u,\mathsf{H}_{pk}(\mathsf{tag},u))$.

4.4 Combining PR–Ext and wCS to Obtain CS

Here, we propose a "transformation" for obtaining a CS secure ABO-XHPS from PR–Ext secure one and wCS secure one. Let $\mathcal{R}$ be a relation family, and for $i\in[2]$, let $\mathcal{X}_i=(\mathsf{XKG}_i,\mathsf{Pub}_i,\mathsf{Ext}_i,\widehat{\mathsf{XKG}}_i,\widehat{\mathsf{Priv}}_i,\widehat{\mathsf{Ext}}_i)$ be an ABO-XHPS which is associated with $\mathcal{R}$. Furthermore, let $\mathsf{H}^{(i)}$ be the internal hash function family of $\mathcal{X}_i$. Then, using $\mathcal{X}_1$ and $\mathcal{X}_2$ as building blocks, we construct another ABO-XHPS $\mathcal{X}'=(\mathsf{XKG}',\mathsf{Pub}',\mathsf{Ext}',\widehat{\mathsf{XKG}}',\widehat{\mathsf{Priv}}',\widehat{\mathsf{Ext}}')$, which is associated with the same $\mathcal{R}$, as in Fig. 6. Let $PK=(pk_1,pk_2)$ be a public key of $\mathcal{X}'$. Then the internal hash function family H' of $\mathcal{X}'$ is defined by $\mathsf{H}'_{PK}(\mathsf{tag},u)=(\pi_1,\pi_2)=(\mathsf{H}^{(1)}_{pk_1}(\mathsf{tag},u),\mathsf{H}^{(2)}_{pk_2}(\mathsf{tag},u))$.

The following theorem holds.

Theorem 2. *Let $\mathcal{R}$ be a relation family and let $\mathcal{X}_1$ and $\mathcal{X}_2$ be ABO-XHPS associated with $\mathcal{R}$. Assume that $\mathcal{R}$ is a one-way relation family, $\mathcal{X}_1$ and $\mathcal{X}_2$ are* PR–Ext *secure and* wCS *secure, respectively. Then the ABO-XHPS $\mathcal{X}'$ constructed as in Fig. 6 is* CS *secure.*

Intuition. In order for an adversary $\mathcal{A}$ against the CS security of $\mathcal{X}'$ to win, it has to make a query $(\mathsf{tag},u,\pi=(\pi_1,\pi_2))$ of either of the following types: (1) $\mathsf{tag}\neq\mathsf{tag}^*\wedge\mathsf{H}^{(1)}_{pk_1}(\mathsf{tag},u)\neq\pi_1\wedge\widehat{\mathsf{Ext}}_1(\widehat{sk}_1,\mathsf{tag},u,\pi_1)=\widehat{\mathsf{Ext}}_2(\widehat{sk}_2,\mathsf{tag},u,\pi_2)\neq\perp$, or (2) $\mathsf{tag}\neq\mathsf{tag}^*\wedge\mathsf{H}^{(1)}_{pk_1}(\mathsf{tag},u)=\pi_1\wedge\mathsf{H}^{(2)}_{pk_2}(\mathsf{tag},u)\neq\pi_2\wedge\widehat{\mathsf{Ext}}_1(\widehat{sk}_1,\mathsf{tag},u,\pi_1)=\widehat{\mathsf{Ext}}_2(\widehat{sk}_2,\mathsf{tag},u,\pi_2)\neq\perp$. However, a tuple of the first type is hard to find due to the PR–Ext security of $\mathcal{X}_1$, because if the query is of first type, then the extracted value $s_1=\widehat{\mathsf{Ext}}_1(\widehat{sk}_1,\mathsf{tag},u,\pi_1)$ is a pseudorandom and is different from $s_2=\widehat{\mathsf{Ext}}_2(\widehat{sk}_2,\mathsf{tag},u,\pi_2)$

$\mathsf{XKG}'(\mathsf{pub}, \mathsf{pri})$: $(pk_i, sk_i) \leftarrow \mathsf{XKG}_i(\mathsf{pub}, \mathsf{pri})$ for $i \in [2]$ $PK \leftarrow (pk_1, pk_2)$; $SK \leftarrow (sk_1, sk_2)$ Return (PK, SK)	$\widehat{\mathsf{XKG}}'(\mathsf{pub}, \mathsf{tag}^*)$: $(pk_i, \widehat{sk_i}) \leftarrow \widehat{\mathsf{XKG}}_i(\mathsf{pub}, \mathsf{tag}^*)$ for $i \in [2]$ $PK \leftarrow (pk_1, pk_2)$; $\widehat{SK} \leftarrow (\widehat{sk_1}, \widehat{sk_2})$ Return $(PK, \widehat{SK})$
$\mathsf{Pub}'(PK, \mathsf{tag}, w)$: $\pi_i \leftarrow \mathsf{Pub}_i(pk_i, \mathsf{tag}, w)$ for $i \in [2]$ Return $\pi \leftarrow (\pi_1, \pi_2)$	$\widehat{\mathsf{Priv}}'(\widehat{SK}, u)$: $\pi_i \leftarrow \widehat{\mathsf{Priv}}_i(\widehat{sk_i}, u)$ for $i \in [2]$ Return $\pi \leftarrow (\pi_1, \pi_2)$
$\mathsf{Ext}'(SK, \mathsf{tag}, u, \pi)$: $s_i \leftarrow \mathsf{Ext}_i(sk_i, \mathsf{tag}, u, \pi_i)$ for $i \in [2]$ If $s_1 = s_2 \neq \perp$ then return s_1 else return $\perp$	$\widehat{\mathsf{Ext}}'(\widehat{SK}, \mathsf{tag}, u, \pi)$: $s_i \leftarrow \widehat{\mathsf{Ext}}_i(\widehat{sk_i}, \mathsf{tag}, u, \pi_i)$ for $i \in [2]$ If $s_1 = s_2 \neq \perp$ then return s_1 else return $\perp$

Fig. 6. The transformation for obtaining a CS secure ABO-XHPS $\mathcal{X}'$ from a PR–Ext secure ABO-XHPS $\mathcal{X}_1$ and a wCS secure ABO-XHPS $\mathcal{X}_2$

with overwhelming probability, regardless of the value s_2. Furthermore, a query of the second type is also hard to find because such tuple can be directly used to break the wCS security of $\mathcal{X}_2$. More specifically, the condition $\mathsf{H}^{(1)}_{pk_1}(\mathsf{tag}, u) = \pi_1$ guarantees $s_1 = \widehat{\mathsf{Ext}}_1(\widehat{sk}_1, \mathsf{tag}, u, \pi_1) = \widehat{\mathsf{Ext}}_2(\widehat{sk}_2, \mathsf{tag}, u, \mathsf{H}^{(2)}_{pk_2}(\mathsf{tag}, u))$ due to the correctness of the all-but-one mode of ABO-XHPS. Therefore, the tuple $(\mathsf{tag}, u, \pi_2, s_1)$ with $\mathsf{tag} \neq \mathsf{tag}^*$ and $\mathsf{H}^{(2)}_{pk_2}(\mathsf{tag}, u) \neq \pi_2$ satisfies the winning condition of the wCS experiment.

5 KEMs Based on ABO-XHPS

In this section, we show our results regarding the KEMs based on ABO-XHPS. Specifically, we show that CCA security of the ABO-XHPS-based KEM can be shown without using gap version of one-way relation families and the stronger correctness requirement defined in [27], and instead a (non-gap) one-way relation family and our weaker correctness, together with CS security, suffices. Furthermore, we show that the KEM can be shown to be CCCA secure if the ABO-XHPS satisfies PR–Ext security. Finally, we show that using the ABO-XHPS in a slightly different way, the ABO-XHPS-based KEM can be extended to be a (C)CCA secure MR-KEM.

5.1 Single-Recipient KEM

Let $\mathcal{R} = (\mathsf{RSetup}, \mathsf{RSamp}, \mathsf{G})$ be a relation family, $\mathcal{X} = (\mathsf{XKG}, \mathsf{Pub}, \mathsf{Ext}, \widehat{\mathsf{XKG}}, \widehat{\mathsf{Priv}}, \widehat{\mathsf{Ext}})$ be an ABO-XHPS associated with $\mathcal{R}$, and $\mathsf{TCR} : \mathcal{U} \to \mathcal{T}$ be a target collision resistant hash function (TCRHF).[3] Then we construct a KEM $\Gamma_1 = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ based on the ABO-XHPS $\mathcal{X}$ as in Fig. 7 (left).

[3] Roughly, an efficiently computable function TCR is said to be a TCRHF if given a random input x, it is hard to find another input x' such that $\mathsf{TCR}(x) = \mathsf{TCR}(x') \wedge x \neq x'$. The formal definition can be found in the full version or in the papers [19,17,12,18,27].

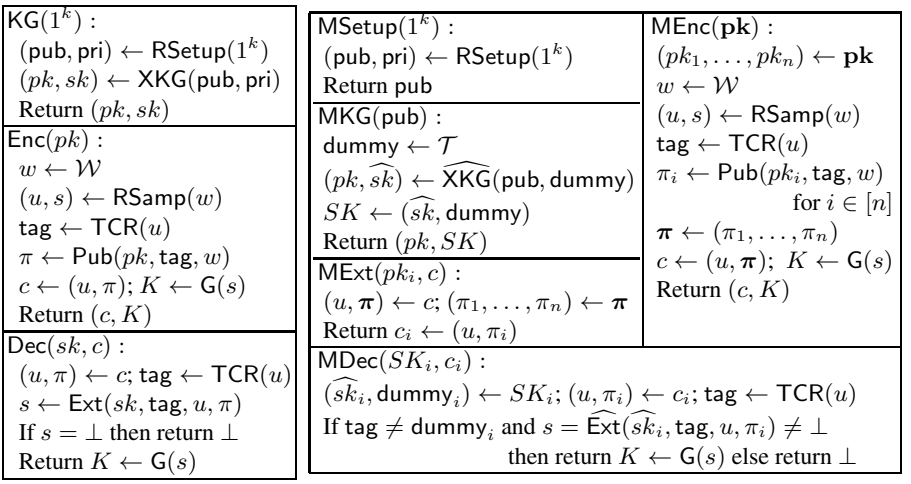


Fig. 7. The (single-recipient) KEM Γ_1 (left) and the MR-KEM Γ_M (right)

CCA Security. Wee [27] showed the following.[4]

Theorem 3. *([27]) If $\mathcal{R}$ is a gap one-way relation family, $\mathcal{X}^{\mathcal{R}}$ is an ABO-XHPS, and* TCR *is a TCRHF, then the KEM Γ_1 is* CCA *secure.*

We show that the same KEM Γ_1 can be proved in the following way, without using a "gap" one-way relation family.

Theorem 4. *If $\mathcal{R}$ is a one-way relation family, $\mathcal{X}^{\mathcal{R}}$ is an ABO-XHPS which satisfies* CS *security, and* TCR *is a TCRHF, then the KEM Γ_1 is* CCA *secure.*

Intuition. To ensure that the real challenge key $K_1^* = \mathsf{G}(s^*)$ looks random for a CCA adversary $\mathcal{A}$, we have to use pseudorandomness of the generator G of the one-way relation $\mathcal{R}$. However, the reduction algorithm $\mathcal{B}$, who attacks pseudorandomness of G, needs to simulate the CCA experiment for $\mathcal{A}$ without knowing the private parameter pri or the randomness w^* used to sample $(u^*, s^*) \in \mathcal{R}_{\mathsf{pub}}$. $\mathcal{B}$ therefore simulates the CCA experiment for $\mathcal{A}$ by using the all-but-one mode of the ABO-XHPS $\mathcal{X}$. The $\widehat{\text{Priv}}$ algorithm enables $\mathcal{B}$ to generate the challenge ciphertext $c^* = (u^*, \pi^*)$ correctly, using $\widehat{sk}$ output from $\widehat{\text{XKG}}(\text{pub}, \text{tag}^*)$ where $\text{tag}^* = \text{TCR}(u^*)$. However, since we do not use "gap" one-way relation family, $\mathcal{B}$ does not have access to the relation oracle $\mathcal{R}_{\mathsf{pub}}$, and thus cannot check inconsistency of a ciphertext by itself. Here, CS security of $\mathcal{X}$ guarantees that even if $\mathcal{A}$ submits an invalid ciphertext $c = (u, \pi)$ with $\mathsf{H}_{pk}(\text{TCR}(u), u) \neq \pi$, the $\widehat{\text{Ext}}$ algorithm almost perfectly works like the Ext algorithm in the real decapsulation algorithm in Dec of Γ_1. In doing so, the TCRHF TCR enables $\mathcal{B}$ to always use $\widehat{\text{Ext}}$,

[4] As we have mentioned, Wee's definition of ABO-XHPS in [27] requires stronger correctness for $\widehat{\text{Ext}}$ algorithm. However, CCA security of the ABO-XHPS-based KEM can be shown without this requirement.

so that the problematic situation where $\mathsf{tag} = \mathsf{TCR}(u) = \mathsf{tag}^*$ never occurs. Then, indistinguishability of two modes guarantees that $\mathcal{A}$'s behavior cannot be non-negligibly different between the case in which the experiment is simulated by $\mathcal{B}$ with the all-but-one mode, and the case in which $\mathcal{A}$ is in the original CCA experiment.

CCCA Security. We show that the KEM Γ_1 based on the ABO-XHPS $\mathcal{X}$ is CCCA secure, when $\mathcal{X}$ is PR–Ext secure.

Theorem 5. *If $\mathcal{R}$ is a one-way relation family, $\mathcal{X}^{\mathcal{R}}$ is an ABO-XHPS which satisfies* PR–Ext *security, and* TCR *is a TCRHF, then the KEM Γ_1 is* CCCA *secure.*

Intuition. The intuitive explanation on the proof of this theorem is very close to that of Theorem 4. The difference is that we can no longer expect that the $\widehat{\mathsf{Ext}}$ algorithm can be used to reject an invalid ciphertext $c = (u, \pi)$ with $\pi \neq \mathsf{H}_{pk}(\mathsf{TCR}(u), u)$, because $\mathcal{X}$ is not guaranteed to be CS secure. However, recall that PR–Ext security of $\mathcal{X}$ guarantees that an extracted value s from an invalid input is a pseudorandom value in $\mathcal{S}$, which in turn guarantees that $K = \mathsf{G}(s) \in \mathcal{K}$ is also pseudorandom and thus unpredictable to the adversary $\mathcal{A}$. Recall also that a valid CCCA adversary has to control its "uncertainty" to be negligible. These help that $\mathcal{A}$'s CDEC query with an invalid ciphertext is "implicitly" rejected, and thus the main reduction algorithm $\mathcal{B}$'s simulation of the CCCA experiment for $\mathcal{A}$ are guaranteed to be almost perfect.

5.2 Multi-Recipient KEM

Here, we show how to construct a MR-KEM using ABO-XHPS. Using the same building blocks ($\mathcal{R}$, $\mathcal{X}$, and TCR) as in Γ_1, we construct a MR-KEM $\Gamma_M = (\mathsf{MSetup}, \mathsf{MKG}, \mathsf{MEnc}, \mathsf{MExt}, \mathsf{MDec})$ as in Fig. 7 (right).

The main feature of the MR-KEM Γ_M is that we use the all-but-one mode of the underlying ABO-XHPS $\mathcal{X}$ even for normal operations, namely, each user's key is setup with $\widehat{\mathsf{XKG}}$ using a "dummy tag" dummy. This is to setup users' keys without using the private parameter pri corresponding to pub, which makes it possible to share pub with many users. Since $\widehat{\mathsf{Ext}}$ cannot extract a value when it is invoked with the tag that is used to generate $\widehat{sk}$, the decapsulation algorithm MDec rejects a user i's ciphertext $c = (u, \pi_i)$ satisfying $\mathsf{TCR}(u) = \mathsf{dummy}$, even if c is honestly generated by using MEnc. Therefore, our MR-KEM Γ_M does not have perfect correctness. However, it satisfies *almost-correctness*: The information on dummy in a user's secret key is information-theoretically hidden from entities other than the user who holds dummy. Therefore, it is hard to find a ciphertext $c = (u, \boldsymbol{\pi})$ that satisfies $\mathsf{TCR}(u) = \mathsf{dummy}$, regardless of the validity of c.

Hiwatari et al. [16] proposed two MR-KEMs. Their first scheme, which is CCA secure, is based on the KEM by [6, Sect. 5.2], while their second scheme, which is CCCA secure, is based on the KEM by [13, Sect. 5]. Both of their schemes can be seen as concrete instantiations of the MR-KEM Γ_M: Their first one is based on the ABO-XHPS $\mathcal{X}_{\mathtt{CKS}}$, while their second one is based on the ABO-XHPS $\mathcal{X}_{\mathtt{HaKu}}$. From another viewpoint, our MR-KEM based on ABO-XHPS is a generalization of Hiwatari et al.

Theorem 6. *If $\mathcal{R}$ is a one-way relation, $\mathcal{X}^{\mathcal{R}}$ is an ABO-XHPS which satisfies* CS *security, and* TCR *is a TCRHF, then the MR-KEM Γ_M is* CCA *secure.*

Theorem 7. *If $\mathcal{R}$ is a one-way relation, $\mathcal{X}^{\mathcal{R}}$ is an ABO-XHPS which satisfies* PR–Ext *security, and* TCR *is a TCRHF, then the MR-KEM Γ_M is* CCCA *secure.*

The proofs proceed similarly to those of Theorems 4 and 5. The difference is that here, we start from the situation in which each user's key is generated by $\widehat{\mathsf{XKG}}$ with dummy tag dummy, while in the proofs of Theorems 4 and 5, we started from the situation in which each user's key is generated by XKG. We also have to deal with the difference between multi-recipient (n users) and single-recipient environments, but this can be essentially dealt with users' key-wise hybrid argument.

6 Discussion

Capturing a Wider Class of Constructions and Security Proofs. We see that by our results, the framework of KEMs based on ABO-XHPS captures most practical (C)CCA secure KEMs. Concretely, many existing CCA secure KEMs can be seen as concrete instantiations derived from our extended framework, which include KEMs by Boyen et al. [5], Cash et al. [6, Sect. 5.2], and Hanaoka and Kurosawa [13, Sect. 4], and the CCCA secure KEMs by Hofheinz and Kiltz [17] and Hanaoka and Kurosawa [13, Sect. 6].

Interestingly, the extraction mode of the ABO-XHPS $\mathcal{X}_{\mathtt{CKS}}$ based on the Cash et al. KEM [6, Sect. 5.2] is exactly the same as that of the CS secure ABO-XHPS obtained via the transformation (Theorem 2) using the PR–Ext secure ABO-XHPS $\mathcal{X}_{\mathtt{HoKi}}$ and the wCS secure ABO-XHPS $\mathcal{X}_{\mathtt{Kiltz}}$. Therefore, Theorems 2 and 4 provide us with an alternative proof of CCA security of Cash et al. KEM, without using the trapdoor test theorem [6, Theorem 2]. We see that this is a concrete evidence that our results are useful for understanding constructions and security proofs of practical CCA secure KEMs in a modular manner.

As is the same with the original framework [27], our results also work for k-wise product relation (i.e. k-independent copies of relation families). This extension is useful to capture hardcore bit-based constructions of KEMs in the framework of ABO-XHPS. However, the clear disadvantage of this approach is that the ciphertext size of the KEM derived from the ABO-XHPS for the k-wise product relation becomes linear in k.

Strictly speaking, ours (and the original framework in [27]) still does not capture the CCA secure KEMs whose session-key is derived using hardcore bits but whose ciphertext size is constant (e.g. [13,14,15,29]). Technically, the security proofs of these KEMs require hybrid argument to replace the real session-key bit-by-bit to finally reach the game in which the real session-key is truly random (and thus an adversary has zero advantage), while the security proofs of the ABO-XHPS-based KEMs in our work and in [27], do not allow this approach. Moreover, it seems to us that how to derive many hardcore-bits in each scheme is quite dependent on the algebraic structure of the constructions. However, we note that at least the "basic structures" of the KEMs in [15,29], which do not consider hardcore-bit-based session-key derivation but derive key by considering "the corresponding (hashed version of) decisional problems, can be seen as concrete instantiations from our extended framework. To extend the framework of ABO-XHPS-based KEMs further to capture these constructions will be worth tackling.

New Instantiations of (MR-)KEMs. Due to Theorems 4, 5, 6, and 7, we can derive a number of new (C)CCA secure (MR-)KEMs. Specifically, due to Theorem 2, we can construct a CS secure ABO-XHPS from a PR−Ext secure ABO-XHPS and a wCS secure ABO-XHPS, or from two PR−Ext secure ABO-XHPS via Theorem 1 (i.e. one of the two ABO-XHPS is treated as a wCS secure ABO-XHPS). Therefore, using the ABO-XHPS we show in Section 4, we can derive a number of variants of KEMs [8,6,12]: we can obtain a CS secure ABO-XHPS by the combination of $\mathcal{X}_{\mathtt{HoKi}}$ and $\mathcal{X}_{\mathtt{Kiltz}}$ (which happens to be essentially identical to $\mathcal{X}_{\mathtt{CKS}}$ as mentioned above) and the combination of $\mathcal{X}_{\mathtt{HaKu}}$ and $\mathcal{X}_{\mathtt{Kiltz}}$. We can also obtain a new CS ABO-XHPS by combining $\mathcal{X}_{\mathtt{HoKi}}$ and $\mathcal{X}_{\mathtt{HaKu}}$, two independent instances of $\mathcal{X}_{\mathtt{HoKi}}$, and two independent instances of $\mathcal{X}_{\mathtt{HaKu}}$. Then, from these CS secure ABO-XHPS, we derive new CCA secure KEMs and MR-KEMs, due to Theorems 4 and 6, respectively.

Furthermore, we can also obtain a number of practical MR-KEMs from existing ABO-XHPS. For example, from the CS secure ABO-XHPS based on the KEM by Boyen et al. [5] (which can be found the full version), we obtain a CCA secure MR-KEM based on the DBDH assumption whose ciphertext size is $n + 1$ group elements when sending to n recipients. This construction is the most efficient CCA secure MR-KEM in terms of ciphertext size. Moreover, by using the factoring-based ABO-XHPS shown in [27, Sect. 4.2] (which is CS secure), we obtain a CCA secure factoring-based MR-KEM which is more efficient than the construction that naively concatenates the ciphertexts from a single-recipient KEM by Hofheinz and Kiltz [18].

Finally, we stress that the advantages of our results are not only the efficiency of the concretely derived (MR-)KEMs, but also the strengthening of the framework of [27], which we believe is useful for future design of (C)CCA secure (MR-)KEMs.

Acknowledgement. The authors would like to thank anonymous reviewers and the members of Shin-Akarui-Angou-Benkyou-Kai for their invaluable comments and suggestions. The first author is supported by the JSPS Fellowships for Young Scientists.

References

1. Bellare, M., Boldyreva, A., Kurosawa, K., Staddon, J.: Multirecipient encryption schemes: How to save on bandwidth and computation without sacrificing security. IEEE Trans. Inf. Theory 53(11), 3927–3943 (2007)
2. 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)
3. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. SIAM J. Comput. 36(5), 1301–1328 (2007)
4. Boyen, X., Mei, Q., Waters, B.: Direct chosen ciphertext security from identity-based techniques. In: ACMCCS 2005, pp. 320–329 (2005)
5. Boyen, X., Mei, Q., Waters, B.: Direct chosen ciphertext security from identity-based techniques. Updated version of [4]. Cryptology ePrint Archive: Report 2005/288 (2005), http://eprint.iacr.org/2005/288/

6. Cash, D., Kiltz, E., Shoup, V.: The Twin Diffie-Hellman Problem and Applications. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 127–145. Springer, Heidelberg (2008)
7. Cramer, R., Shoup, V.: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
8. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. 33(1), 167–226 (2003)
9. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography. In: STOC 1991, pp. 542–552 (1991)
10. Elkind, E., Sahai, A.: A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive: Report 2002/042 (2002), http://eprint.iacr.org/2002/042/
11. Gennaro, R., Krawczyk, H., Rabin, T.: Secure Hashed Diffie-Hellman over Non-DDH Groups. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 361–381. Springer, Heidelberg (2004)
12. Hanaoka, G., Kurosawa, K.: Efficient Chosen Ciphertext Secure Public Key Encryption under the Computational Diffie-Hellman Assumption. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 308–325. Springer, Heidelberg (2008)
13. Hanaoka, G., Kurosawa, K.: Efficient chosen ciphertext secure public key encryption under the computational Diffie-Hellman assumption. Full version of [12]. Cryptology ePrint Archive: Report 2008/211 (2008), http://eprint.iacr.org/2008/211/
14. Hanaoka, G., Kurosawa, K.: Between hashed DH and computational DH: Compact encryption from weaker assumption. IEICE Transactions E93-A(11), 1994–2006 (2010)
15. Haralambiev, K., Jager, T., Kiltz, E., Shoup, V.: Simple and Efficient Public-Key Encryption from Computational Diffie-Hellman in the Standard Model. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 1–18. Springer, Heidelberg (2010)
16. Hiwatari, H., Tanaka, K., Asano, T., Sakumoto, K.: Multi-recipient Public-Key Encryption from Simulators in Security Proofs. In: Boyd, C., González Nieto, J. (eds.) ACISP 2009. LNCS, vol. 5594, pp. 293–308. Springer, Heidelberg (2009)
17. Hofheinz, D., Kiltz, E.: Secure Hybrid Encryption from Weakened Key Encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (2007)
18. Hofheinz, D., Kiltz, E.: Practical Chosen Ciphertext Secure Encryption from Factoring. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 313–332. Springer, Heidelberg (2009)
19. Kiltz, E.: Chosen-Ciphertext Secure Key-Encapsulation Based on Gap Hashed Diffie-Hellman. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 282–297. Springer, Heidelberg (2007)
20. Kurosawa, K., Desmedt, Y.: A New Paradigm of Hybrid Encryption Scheme. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 426–442. Springer, Heidelberg (2004)
21. Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: STOC 1989, pp. 33–43 (1989)
22. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC 1990, pp. 427–437 (1990)
23. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: STOC 2008, pp. 187–196 (2008)
24. Rackoff, C., Simon, D.R.: Non-interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992)

25. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: FOCS 1999, pp. 543–553 (1999)
26. Smart, N.P.: Efficient Key Encapsulation to Multiple Parties. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 208–219. Springer, Heidelberg (2005)
27. Wee, H.: Efficient Chosen-Ciphertext Security via Extractable Hash Proofs. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 314–332. Springer, Heidelberg (2010)
28. Wee, H.: Threshold and Revocation Cryptosystems via Extractable Hash Proofs. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 589–609. Springer, Heidelberg (2011)
29. Yamada, S., Kawai, Y., Hanaoka, G., Kunihiro, N.: Public key encryption schemes from the (B)CDH assumption with better efficiency. IEICE Transactions 93-A(11), 1984–1993 (2010)

Robust Encryption, Revisited

Pooya Farshim[1], Benoît Libert[2],
Kenneth G. Paterson[3], and Elizabeth A. Quaglia[4]

[1] Fachbereich Informatik, Technische Universität Darmstadt, Germany
[2] Technicolor, France
[3] Information Security Group, Royal Holloway, University of London, UK
[4] Département d'Informatique, École Normale Supérieure – Paris, France

Abstract. We revisit the notions of robustness introduced by Abdalla, Bellare, and Neven (TCC 2010). One of the main motivations for the introduction of strong robustness for public-key encryption (PKE) by Abdalla et al. is to prevent certain types of attack on Sako's auction protocol. We show, perhaps surprisingly, that Sako's protocol is still vulnerable to attacks exploiting robustness problems in the underlying PKE scheme, even when it is instantiated with a *strongly* robust scheme. This demonstrates that current notions of robustness are insufficient even for one of its most natural applications. To address this and other limitations in existing notions, we introduce a series of new robustness notions for PKE and explore their relationships. In particular, we introduce *complete* robustness, our strongest new notion of robustness, and give a number of constructions for completely robust PKE schemes.

Keywords: Robustness, Anonymity, Public-key encryption, Security proofs.

1 Introduction

A commonly pursued goal in cryptography is message privacy, which is typically achieved by means of encryption. In recent years, the privacy of users has become an equally relevant concern. It has led the research community to strive for anonymity properties when designing cryptographic primitives. In public-key encryption, in particular, *key-privacy* (a.k.a. receiver anonymity) was introduced in [4] to capture the idea that a ciphertext does not leak any information about the public key under which it was created, thereby making the communication anonymous. In this context, Abdalla, Bellare, and Neven [2] raised a fundamental question: how does a legitimate user know if an anonymous ciphertext is intended for him? Moreover, what happens if he uses his secret key on a ciphertext *not* created under his public key? To address this question, Abdalla et al. formalized a property called *robustness*, which (informally speaking) guarantees that decryption attempts fail with high probability if the "wrong" private key is used. They argued that, in all applications requiring anonymous public-key encryption, robustness is usually needed as well. These applications include auction protocols with bid privacy [23], consistency [1] in searchable encryption [7]

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 352–368, 2013.

and anonymous broadcast encryption [3,21]. As shown by Mohassel [22], robustness is also important in guaranteeing the anonymity of hybrid encryption schemes resulting from the combination of anonymous asymmetric and symmetric components.

1.1 Robust Public-Key Encryption

Robustness ensures that a ciphertext cannot correctly decrypt under two different secret keys. This notion has (often implicitly) been present in the literature (e.g., [23,7,9,19,3]), but formal definitions remained lacking until the recent foundational work of Abdalla et al. [2]. In particular, Abdalla et al. introduced two flavors of encryption robustness: *weak* and *strong* robustness.

Weak robustness is modeled as a game in which a winning adversary outputs a valid message M and two distinct public keys pk_0 and pk_1 such that the encryption of M under pk_0 decrypts to a valid message under sk_1, the secret key corresponding to pk_1. This notion is of interest since it precisely addresses the issue of *using the wrong key* that arises in anonymity contexts (such as anonymous broadcast encryption [3,21], for instance), but it is also useful in achieving the stronger notion of strong robustness.

Strong robustness—also called SROB-CCA when the adversary has access to a decryption oracle—allows for a more powerful adversary which chooses a ciphertext C (as opposed to a message which will be honestly encrypted) and two distinct public keys, and wins if C decrypts to a valid message under both corresponding secret keys. In [2] the need for this notion is motivated by scenarios where ciphertexts can be adversarially chosen. The authors of [2] give Sako's auction protocol [23] as an example of such a situation, explaining that strong robustness is required in order to prevent an attack on the fairness of this protocol by a cheating bidder and a colluding auctioneer.

As pointed out by Abdalla et al. [2], merely appending the receiver's public key to the ciphertext is not an option for providing robustness, since it destroys key-privacy properties. Abdalla et al. also showed that the seemingly natural solution of using an unkeyed redundancy function to modify the message before encryption does not achieve even weak robustness, thus demonstrating the non-triviality of the problem. They then gave several anonymity-preserving constructions to obtain both weak and strong robustness for public-key encryption. Using a simple tweak, they also showed how to render the Cramer–Shoup cryptosystem [12] strongly robust without introducing any overhead.

More recently, Mohassel [22] studied robustness in the context of hybrid encryption [13]. He showed that weak robustness (and not only anonymity) is needed in the asymmetric part of a hybrid encryption scheme to ensure anonymity of the overall scheme. Mohassel also considered relaxations, called *collision-freeness*, of both weak and strong robustness. He showed that many constructions in the literature are natively collision-free and showed how to generically turn any weakly (resp., strongly) collision-free scheme into a weakly (resp., strongly) robust one.

1.2 Our Contributions

The need for stronger definitions. In this paper, we argue that some applications require even stronger forms of robustness than those considered in [2,22]. The first such application is, perhaps surprisingly, the construction of auction protocols with bid privacy, like that of Sako [23]. Recall that this was one of the initial motivations for analyzing robustness in [2]. Strong robustness actually turns out *not* to suffice for thwarting attacks against the fairness of Sako's auction protocol [23]: strong robustness assumes honestly generated public keys whereas, if the auctioneer can collude with cheating bidders (as assumed in [2]), what really needs to be considered is an adversary who can maliciously generate ciphertexts *and* the public keys. To illustrate this, we show an attack on the fairness of Sako's protocol when instantiated with $\mathcal{CS}^{\star}$, a variant of the Cramer–Shoup encryption scheme which was proven to be key-private and strongly robust in [2]. This observation, then, motivates us to introduce notions of robustness where keys may be maliciously generated. We do not offer a full treatment of the delicate issue of fairness in auction protocols and its relation to robustness, since that is beyond the scope of this paper. Rather, as with [2], we use Sako's protocol as a motivation for introducing and studying stronger robustness notions.

The limitations of existing robustness notions, and therefore the motivation for this work, are not solely restricted to Sako's protocol. For instance, existing notions are not necessarily strong enough to provide robustness guarantees if the scheme is used to encrypt *key-dependent* messages [6] or messages encrypted under *related keys* [5]. This is because the adversary is denied access to the secret keys in these notions. The strongest of our new notions gives the adversary sufficient power and automatically provides robustness in these more challenging settings.

New notions of robustness and their relations. Our strongest new notion is called *complete* robustness (CROB) and is obtained by progressively removing various restrictions on adversarial capabilities in the strong robustness security model. First, we give access to honestly generated secret keys and arrive at an intermediate notion which we term *unrestricted (strong) robustness* (US-ROB). Next, we also remove the honest key-generation requirement to get to the notion of *full robustness* (or FROB for short). We then view robustness in terms of the behavior of the encryption and decryption algorithms with respect to each other, and obtain our CROB notion. Roughly speaking, in CROB, the adversary should not be able to find "collisions" in the scheme beyond those which are already implied by the correctness property of the scheme. For example, he should not be able to "explain" a ciphertext C of his choice as an encryption under two different adversarially chosen public keys pk_0, pk_1 by revealing the plaintext and the encryption coins for pk_0 and the secret key sk_1 for pk_1. As we will see, full robustness can be viewed as the "decryption-only part" of CROB. Another natural notion of robustness, which we call *key-less robustness* (KROB), arises as the dual notion corresponding to the "encryption-only part" of CROB, and is also implied by CROB. Finally, XROB is a "mixed" notion derived from FROB

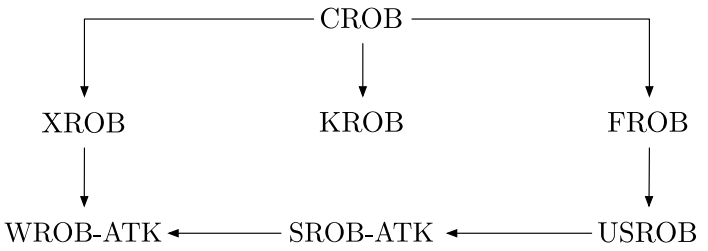


Fig. 1. Relations among notions of robustness

and KROB that has no natural interpretation but is a useful tool in establishing results about these notions.

We next study how these new notions of robustness relate to each other and to existing notions. Figure 1 summarizes the main relations that we prove between our new and existing robustness notions. In this figure, the lack of an implication between two notions should be interpreted as meaning that we prove a separation. Thus, for example, we will show that CROB is strictly stronger than FROB. It is apparent from the figure that we provide a complete account of the pairwise relations between the various robustness notions. In addition to these relations, we can prove several pairwise separations. For example, we will show that no two of the three notions from $\{\mathrm{FROB}, \mathrm{KROB}, \mathrm{XROB}\}$ are sufficient to prove CROB, but that their combination is. Thus we obtain a characterization of CROB in terms of the three intermediate notions. These separations are not displayed in the figure for ease of visual presentation.

That robustness can come in so many flavors may be unsettling to some readers. Certainly, one should not seek to clutter the definitional landscape unnecessarily. Yet, with the exception of XROB, all of our notions arise as natural generalizations of the existing notions. Exploring their relations is then a natural endeavor. This is not so different from the situation for, say, confidentiality and anonymity notions for public-key encryption, where we now have many different security definitions and developing an understanding of their relations has taken several years.

Constructions of completely robust encryption. Having defined CROB and its weaker relatives, we prove it to be achievable via a variety of efficient and natural constructions.

We first show that the generic construction for strong robustness presented in [2] is *already* powerful enough as to also achieve CROB. Further, we observe that a slight modification of this transformation allows dispensing with the weak robustness assumption—which was necessary in [2]—on the underlying PKE scheme. Moreover, we point out that the random-oracle-based generic transformation of Mohassel [22] also achieves CROB.

In the standard model, we also answer in a positive sense a question left open in [2] as to whether the Canetti–Halevi–Katz [11] (CHK) paradigm—which is known to provide chosen-ciphertext secure cryptosystems from weakly secure identity-based encryption (IBE) schemes—can be leveraged to construct systems that are simultaneously anonymous and offer message privacy under chosen-ciphertext attacks (AI-CCA security) *and* are robust in a strong sense. Answering this question is non-trivial: Abdalla et al. pinpointed that applying the one-time-signature-based CHK transformation to, say, the Boyen–Waters IBE [10] does not provide SROB-CCA or even SROB-CPA. Here, we show how to obtain AI-CCA-secure, completely robust PKE schemes from weakly secure IBE schemes. Our construction is a variant of the Boneh–Katz construction for chosen-ciphertext security [8], and it only requires the underlying IBE to satisfy a weak level of security under chosen-plaintext attacks. In comparison, the most powerful transformation of [2] must start from a scheme that is already AI-CCA-secure to achieve a comparable result. Because our technique simultaneously provides complete robustness *and* AI-CCA security, it enjoys better efficiency than applying the strongest robustness-conferring transformation of [2] to an AI-CCA-secure scheme obtained from the original Boneh–Katz transformation.

Finally, we also ask whether we can improve upon the efficiency of generic constructions with concrete schemes whose security rests on specific computational assumptions. By giving a concrete construction of a scheme that is CROB and AI-CCA-secure, we present a different and potentially more efficient way of directly achieving CROB for certain hybrid encryption schemes such as the Hofheinz–Kiltz [17] or Kurosawa–Desmedt [20] schemes. To do so, we take advantage of certain properties in the underlying symmetric components. Namely, we consider hybrid schemes that build on the *encrypt-then-MAC* paradigm in their symmetric part to obtain a suitably secure symmetric cipher. We show that, if the message authentication code (MAC) is what we call *committing*, then a simple modification in the hybrid scheme gives complete robustness without any significant computational overhead. The use of committing MACs readily extends as a tool to design AI-CCA-secure CROB hybrid constructions via the KEM/DEM framework [13]. Concretely, Mohassel [22] showed that the KEM/DEM framework gives an AI-CCA-secure hybrid encryption scheme when the KEM component is weakly robust and AI-CCA, and the DEM component is an authenticated symmetric encryption scheme. If the latter part is furthermore realized using the encrypt-then-MAC approach with a committing MAC, we easily obtain complete robustness as well. As we will see, the committing MAC technique can also offer certain advantages.

Taken altogether, our constructions achieving CROB rely on different building blocks and, when fully instantiated, allow us to rely on a variety of different hardness assumptions. They demonstrate that CROB, while providing strong guarantees, is attainable in an efficient and flexible manner.

Organization. We start by highlighting the limitations of previous notions of robustness in Section 2. Section 3 presents our new notions. In Section 4,

we study the relations among notions of robustness. We describe our generic constructions in Section 5 and give an efficient construction in Section 6. We close by some concluding remarks in Section 7. Many details and all proofs are deferred to the full version [15].

2 Strong Robustness Does Not Suffice for Auction Protocols

Sako's auction protocol [23] was the first practical protocol to ensure *bid privacy*, i.e., to hide the bids of losers. The basic idea is as follows. Let $V = \{v_1, ..., v_N\}$ be the set of possible bid values. The auctioneer prepares N key-pairs $(sk_i, pk_i)_{i\in\{1,...,N\}}$ and publishes the N public keys. To *bid* for a value v_i a bidder encrypts a pre-determined message M under the public key pk_i. This is signed and posted by the bidder. To *open* a bid the auctioneer attempts to decrypt the encrypted bids one by one using sk_N. If at least one decrypts to M, the auctioneer publishes the winning bid v_N, a list of all the winning bidders and the secret key sk_N for the bidders to verify correctness of the result. If no decryption returns M, the auctioneer repeats the procedure using sk_{N-1}, and so on. For the auction to hide the bid values, the underlying public-key encryption scheme needs to be key-private, in the sense of [4].

In [23], Sako provided an example of an auction protocol scheme based on the ElGamal public-key encryption scheme, which is key-private. In [2], Abdalla et al. gave an attack which allows a cheating bidder and a colluding auctioneer to break the fairness of the protocol. This attack is based on the fact that the ElGamal scheme is not robust and therefore the auctioneer can open the cheating bidder's bid to an arbitrary (winning) value. To prevent this attack, the authors of [2] suggest using any *strongly* robust scheme (strong robustness, instead of simply weak robustness, is required since the ciphertexts are generated adversarially; see [2,15] for the details).

We show that strong robustness is *not* sufficient to prevent an attack of the above type on Sako's protocol. More precisely, in [15, Appendix C] we present an attack on the protocol when it is instantiated with a variant of the Cramer–Shoup encryption scheme, $\mathcal{CS}^\star$, which is known to be key-private and strongly robust (the latter result was proved in [2]). Just as with the attack of Abdalla et al. [2], the attack we present assumes a cheating bidder and a colluding auctioneer. The key idea behind the attack is that an auctioneer can maliciously prepare the public keys so that the cheating bidder's encryption decrypts to M under *any* secret key.

This attack shows that strong robustness is not enough to guarantee fairness in Sako's auction protocol. Intuitively what is needed here is a form of robustness wherein all the public keys and ciphertexts in the system may be adversarially generated. In the coming sections we will formalize stronger notions of robustness which rule out such attacks.

3 New Notions of Robustness

3.1 A Direct Strengthening: Full Robustness

Recall that an SROB adversary has to output a ciphertext C and two public keys pk_0 and pk_1 such that C decrypts to a message M_0 under (sk_0, pk_0) and a message M_1 under (sk_1, pk_1). The notion poses three restrictions on the adversary: (1) pk_0 and pk_1 have to be distinct; (2) The corresponding secret keys cannot have been queried by the adversary; (3) The public keys are honestly generated.

The first condition is *inherent* to modeling the behavior of an encryption scheme when used on different public keys, and removing it would make it trivial for an adversary to win.

We now look at the notion resulting from the removal of the second restriction, i.e., when the adversary is allowed to query secret keys even for the finally output public keys. We call this notion unrestricted strong robustness (USROB). This notion is powerful enough to model scenarios where keys are honestly generated, but an adversary may know the secret keys. This, for example, includes robustness for the encryption of key-dependent messages as discussed in the introduction.

However, as we have seen in the previous section, if an adversary can control the generation of keys, it may be unreasonable to assume that it can only generate the keys honestly. We therefore can strengthen USROB further by removing the third restriction on the adversary. We, however, ask the adversary to return secret keys for the public keys that it chooses. Two points deserve further attention at this point. First, returning the secret keys is to allow for a polynomial-time game definition which is not excessively strong. Second, we do not require the secret keys to be valid. Indeed, it is the responsibility of the decryption algorithm to check that the key-pair it receives is valid. Note that as a result of removing the two restrictions, the adversary has now full control over the keys, and we no longer need to provide the adversary with the oracles present in the SROB and USROB games. These modifications result in a simple, but strong, notion we call *full robustness* (FROB), and formalize in Figure 2.

proc Initialize	**proc Finalize**$(C, pk_0, pk_1, sk_0, sk_1)$ // FROB
$pars \leftarrow\!\!\$\ \mathsf{PG}$	If $(pk_0 = pk_1)$ Then Return F
Return $pars$	$M_0 \leftarrow \mathsf{Dec}(pars, pk_0, sk_0, C)$
	$M_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, C)$
	Return $(M_0 \neq \perp) \wedge (M_1 \neq \perp)$

Fig. 2. Game defining full robustness

3.2 A Unified Approach: Complete Robustness

At this point it can be asked if there are attacks which fall outside the FROB model. To answer this question, we take a somewhat different approach towards robustness and view it in terms of the behavior of the encryption and decryption routines of a scheme with respect to each other. In fact, this is the underlying intuition behind not only the original weak robustness notion,[1] but also the standard correctness criterion for a PKE scheme (albeit for a single key). This leads us to a new notion which we term *complete robustness* (CROB). In this game the shared parameters of the system are passed to an adversary, which then arbitrarily interacts with the encryption and decryption routines on plaintexts, ciphertexts, keys, and even random coins of its choice. Its goal is to find an "unexpected collision" in the cryptosystem (i.e., one outside that imposed by the correctness criterion). We formalize the CROB game in Figure 3.

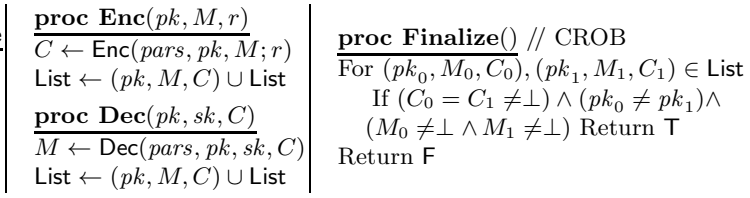


Fig. 3. Game defining complete robustness

Key-less robustness. It can be seen through an easy inspection that full robustness is a sub-case of complete robustness where the adversary is restricted to querying the **Dec** oracle. One can also consider the dual case where the adversary only queries the **Enc** oracle. This results in a new notion which we call *key-less robustness* (KROB). Key-less robustness differs from full robustness in that an adversary no longer needs to return any secret keys, but instead "opens" a ciphertext by providing the random coins and the message used in the encryption. More precisely, the adversary outputs two messages, two distinct public keys and two sets of random coins, and its goal is to invoke a collision in the encryption algorithm. The game is shown in Figure 4.

In the next section we give a complete treatment of relations among different notions.

Identity-based encryption. In the IBE setting the identities (analogous to public keys in the PKE setting) are already chosen maliciously, while the natural extension of our notions would allow the adversary to also choose the IBE master keys maliciously. In particular, the identity-based analogue of FROB would be strong enough to guarantee *well-addressedness* according to the definition proposed by Hofheinz and Weinreb [18] (see also [15, Appendix B]), whereas

[1] This then disappears in the SROB game as the adversary outputs ciphertexts.

proc Initialize	**proc Finalize**$(M_0, M_1, pk_0, pk_1, r_0, r_1)$ // KROB
$pars \leftarrow\!\!\$\ \mathsf{PG}$	If $(pk_0 = pk_1)$ Then Return F
Return $pars$	$C_0 \leftarrow \mathsf{Enc}(pars, M_0, pk_0; r_0)$
	$C_1 \leftarrow \mathsf{Enc}(pars, M_1, pk_1; r_1)$
	Return $(C_0 = C_1 \neq \perp)$

Fig. 4. Game defining key-less robustness

SROB-CCA may not always do so. We leave the further development of the ID-based setting to future work.

4 Relations among Notions of Robustness

We now study how the various notions of robustness relate to each other. Starting with complete robustness, it may be asked if KROB and FROB are strong enough together to jointly imply CROB. We show that this is *not* the case. Indeed, there is a third "mixed" notion of robustness implicit in CROB, which we term XROB and formalize in Figure 5. As the next theorem shows, the XROB notion is necessary in the sense that it is not implied by KROB and FROB together.

In fact, no pair of the notions from {FROB, KROB, XROB} implies the third. Furthermore, the conjunction of all three notions is sufficient to imply CROB.

Theorem 1 (CROB **characterization).** *A PKE scheme is* CROB *if and only if it is simultaneously* FROB*,* KROB*, and* XROB*. Furthermore, no combination of at most two of* FROB*,* KROB*, and* XROB *is sufficient to provide the* CROB *guarantees.*

We prove the theorem via a sequence of propositions in [15, Appendix E].

proc Initialize	**proc Finalize**$(M_0, pk_0, r_0, C_1, pk_1, sk_1)$ // XROB
$pars \leftarrow\!\!\$\ \mathsf{PG}$	If $(pk_0 = pk_1)$ Then Return F
Return $pars$	$C_0 \leftarrow \mathsf{Enc}(pars, M_0, pk_0; r_0)$
	$M_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, C_1)$
	Return $(C_0 = C_1) \wedge (M_0 \neq \perp) \wedge (M_1 \neq \perp)$

Fig. 5. Game defining mixed robustness

As a next step we study how our new notions relate to the existing notions from Abdalla et al. [2]. Since USROB is a natural intermediate notion, for the sake of completeness, we also investigate where it stands in relation to existing notions. We start by observing that FROB $\implies$ USROB $\implies$ SROB-CCA as the adversary becomes progressively more restricted in each game. Moreover, in the first part of the following theorem, we show that USROB is strictly stronger than SROB-CCA, and that FROB is strictly stronger than USROB. In the second part of the theorem we show that KROB does not even imply WROB-CPA,

separating this notion from all notions other than complete robustness. Finally, we show XROB implies WROB-CCA but *not* SROB-CPA. Hence XROB can be seen a strengthened version of weak robustness in a direction orthogonal to strong robustness.

Theorem 2 (Relation with WROB **and** SROB**).** *Let $\mathcal{PKE}$ be a PKE scheme. We have the following.*

- **FROB***: If $\mathcal{PKE}$ is* FROB*, then it is also* USROB*. If $\mathcal{PKE}$ is* USROB *then it is also* SROB-CCA*. Moreover, these implications are strict.*
- **KROB***:* KROB *does not imply* WROB-CPA *and* SROB-CCA *does not imply* KROB*.*
- **XROB***: If $\mathcal{PKE}$ is* XROB*, then it is also* WROB-CCA*. Furthermore,* XROB *does not imply* SROB-CPA *and* SROB-CCA *does not imply* XROB*.*

We prove the theorem in [15, Appendix F]. The results of [2] together with Theorems 1 and 2 resolve all the relations between any pair of robustness notions as we have summarized in Figure 1. For example, to see that KROB $\not\Longrightarrow$ FROB, we use the facts that FROB $\Longrightarrow$ SROB-ATK but KROB $\wedge$ XROB $\not\Longrightarrow$ SROB-ATK. Moreover, although we do not formally prove it here, all our separating examples are designed such that they preserve the AI-ATK security of the underlying PKE schemes. Hence Figure 1 also applies in the presence of AI-ATK security.

5 Generic Constructions of Completely Robust Public-Key Encryption

5.1 Mohassel's Transformation

Mohassel [22] gives a generic transformation in the random-oracle model that converts an AI-ATK encryption scheme into one which is SROB-CCA without compromising its AI-ATK security. This construction also achieves complete robustness. In this construction, the hash value $H(pk, r, M)$, where r is the randomness used in the encryption, is attached to ciphertexts. This immediately rules out all forms of collisions between ciphertexts, as the hash values are unlikely to collide on two distinct public keys.

5.2 The ABN Transformation

In [2, Theorem 4.2] the authors give a generic construction for a scheme $\overline{\mathcal{PKE}}$ which confers strong robustness and preserves the AI-ATK security of the starting scheme $\mathcal{PKE}$, provided that the latter scheme is additionally WROB. We briefly describe how the transformation works, and refer the reader to the original work for the details. At setup, include in *pars* for $\overline{\mathcal{PKE}}$ the parameters of a commitment scheme (see [15, Appendix G] for the definitions). When encrypting, commit to the public key, and encrypt the *de-commitment* along with the

message. Also include the commitment as a ciphertext component. Decryption checks the commitment/de-commitment pair for consistency and rejects if this is not the case. We strengthen the result of [2], showing that this construction achieves complete robustness:

Theorem 3 (The ABN transformation achieves CROB**).** *Let $\mathcal{A}$ be a PPT* CROB *adversary against $\overline{\mathcal{PKE}}$. Then there exist PPT adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$ against the binding property of $\mathcal{CMT}$ such that*

$$\mathbf{Adv}^{\mathrm{crob}}_{\overline{\mathcal{PKE}}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(\mathcal{B}_1) + \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(\mathcal{B}_2) + \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(\mathcal{B}_3).$$

The proof of this theorem is given in [15, Appendix H], where we show scheme $\overline{\mathcal{PKE}}$ is simultaneously FROB, KROB, and XROB.

5.3 A Modification of the ABN Transformation

While the original transformation [2] *does* provide AI-ATK and CROB guarantees, the AI-ATK security of the transformed scheme $\overline{\mathcal{PKE}}$ relies on the weak robustness of the underlying encryption scheme $\mathcal{PKE}$ in the case of chosen-ciphertext adversaries (i.e., when ATK = CCA). We show that, if the underlying encryption scheme supports labels [24] (in which case the encryption and decryption algorithms both take an additional public string L as input; see [15, Appendix A]), this assumption can be eliminated and we only need $\mathcal{PKE}$ to be AI-ATK-secure.

Although the weak robustness assumption is not too demanding in theory (since any encryption scheme can be made weakly robust by means of a keyed redundancy-based transformation [2]), our construction provides better efficiency in some settings since many AI-CCA encryption schemes, such as the Cramer–Shoup or the Kurosawa–Desmedt scheme, natively support labels.[2]

Our transformation, which relies on a commitment scheme $\mathcal{CMT}$ consisting of algorithms $(\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$, is as follows.

$\overline{\mathsf{PG}}(1^\lambda)$**:** Run $pars \leftarrow\!\!\$\ \mathsf{PG}(1^\lambda)$ to obtain public parameters $pars$ for $\mathcal{PKE}$. Then, generate $cpars \leftarrow\!\!\$\ \mathsf{CPG}(1^\lambda)$ for $\mathcal{CMT}$. Finally, return $(pars, cpars)$.

$\overline{\mathsf{KG}}(pars, cpars)$**:** Compute and return $(sk, pk) \leftarrow\!\!\$\ \mathsf{KG}(pars)$.

$\overline{\mathsf{Enc}}\big((pars, cpars), pk, M\big)$**:** The algorithm proceeds in two steps.

1. Commit to pk by computing a pair $(com, dec) \leftarrow\!\!\$\ \mathsf{Com}(cpars, pk)$.
2. Encrypt $M \| dec$ under the label $L = com$ by setting the ciphertext C to be $\mathsf{Enc}(pars, pk, M \| dec, L)$.

Return (C, com) as the final ciphertext.

$\overline{\mathsf{Dec}}\big((pars, cpars), pk, sk, (C, com)\big)$**:** The algorithm proceeds in two steps.

[2] In the worst case, labeled public-key encryption schemes can always be obtained by appending the label to the encrypted plaintext and checking whether the correct label is recovered at decryption.

1. Compute $M' \leftarrow \mathsf{Dec}\big(pars, pk, sk, (C, com), L\big)$, with $L = com$. Then, parse M' as $M \| dec$ (and return $\perp$ if M' cannot be parsed properly).
2. Return M if $\mathsf{Ver}(cpars, pk, com, dec) = 1$. Else return $\perp$.

Theorem 4, whose proof is in [15, Appendix I], shows that thanks to the use of labels, we do not have to rely on any weaker form of robustness of $\mathcal{PKE}$ when proving the AI-ATK security of $\overline{\mathcal{PKE}}$.

Theorem 4. *If $\mathcal{PKE}$ is* AI-ATK*-secure and $\mathcal{CMT}$ is a hiding commitment, then $\overline{\mathcal{PKE}}$ is* AI-ATK*-secure. More precisely, for any PPT* AI-ATK *adversary $\mathcal{A}$ against $\overline{\mathcal{PKE}}$, there exists a PPT* AI-ATK *adversary $\mathcal{B}_1$ against $\mathcal{PKE}$ and a PPT distinguisher $\mathcal{B}_2$ against $\mathcal{CMT}$ such that*

$$\mathbf{Adv}^{\text{ai-atk}}_{\overline{\mathcal{PKE}}}(\mathcal{A}) \leq 2 \cdot \mathbf{Adv}^{\text{ai-atk}}_{\mathcal{PKE}}(\mathcal{B}_1) + \mathbf{Adv}^{\text{hide}}_{\mathcal{CMT}}(\mathcal{B}_2).$$

Furthermore, the above construction is CROB *if $\mathcal{CMT}$ is a binding commitment. More precisely, for any PPT* CROB *adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ against the binding property of the commitment scheme such that*

$$\mathbf{Adv}^{\text{crob}}_{\overline{\mathcal{PKE}}}(\mathcal{A}) \leq \mathbf{Adv}^{\text{bind}}_{\mathcal{CMT}}(\mathcal{B}).$$

5.4 Completely Robust AI-CCA-Secure PKE from Selectively Secure IBE

Next, we present a modification of the Boneh–Katz approach [8] which provides both CROB and AI-CCA security when applied to any IBE scheme that only provides TA anonymity in the multi-authority selective-ID setting (or sID-TAA-CPA security, as defined in [15, Appendix J]). In particular, this positively answers the question of whether CHK-like techniques can be used to achieve a strong flavor of robustness from weakly secure IBE.

Let $\mathcal{IBE}$ be an sID-TAA-CPA secure IBE scheme. We obtain a completely robust AI-CCA-secure public-key encryption scheme $\overline{\mathcal{PKE}}$ by combining $\mathcal{IBE}$ with a strongly secure message authentication code $\mathcal{MAC}$ and a trapdoor commitment scheme $\mathcal{TCMT}$.

Recall that a trapdoor commitment scheme $\mathcal{TCMT}$ consists of efficient algorithms $(\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver}, \mathsf{Equiv})$ where $(\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$ function as in an ordinary commitment except that CPG outputs public parameters *cpars* and a trapdoor td. In addition, Equiv allows equivocating a commitment using the trapdoor td: for any two messages m_1, m_2 and any tuple (com, dec_1) produced as $(com, dec_1) \leftarrow\!\!\$\ \mathsf{Com}(cpars, m_1)$, the trapdoor td allows computing the value $dec_2 \leftarrow\!\!\$\ \mathsf{Equiv}(td, com, m_1, dec_1, m_2)$ such that $\mathsf{Ver}(cpars, com, m_2, dec_2) = 1$. Moreover, (com, dec_2) has the same distribution as $\mathsf{Com}(cpars, m_2)$.

Our IBE-based construction $\overline{\mathcal{PKE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ is as follows.

$\overline{\mathsf{PG}}(1^\lambda)$**:** Run $pars \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{PG}(1^\lambda)$ to obtain common public parameters *pars*. Also run $cpars \leftarrow\!\!\$\ \mathsf{CPG}(1^\lambda)$ to obtain public parameters for a trapdoor commitment scheme $\mathcal{TCMT}$. Then, choose a message authentication code $\mathcal{MAC}$ with key length $\ell \in \text{poly}(\lambda)$. Finally, return $(pars, cpars, \mathcal{MAC})$.

$\overline{\mathsf{KG}}(pars, cpars, \mathcal{MAC})$**:** Generate $(msk, mpk) \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{MPG}(pars)$ for $\mathcal{IBE}$. Return the key pair $(sk, pk) := (msk, mpk)$.

$\overline{\mathsf{Enc}}\big((pars, cpars, \mathcal{MAC}), pk, M\big)$**:** To encrypt M under $pk = mpk$, the algorithm proceeds as follows.

1. Choose a random MAC key $k \leftarrow\!\!\$\ \{0,1\}^{\ell}$.
2. Commit to $mpk\|k$ by computing $(com, dec) \leftarrow\!\!\$\ \mathsf{Com}(cpars, mpk\|k)$.
3. Encrypt $M\|k\|dec$ under the identity com by setting C to the output of $\mathcal{IBE}.\mathsf{Enc}(pars, mpk, com, M\|k\|dec)$.
4. Compute $tag = \mathsf{MacGen}_k(C)$ and return (C, com, tag) as the final ciphertext.

$\overline{\mathsf{Dec}}\big((pars, cpars, \mathcal{MAC}), pk, sk, (C, com, tag)\big)$**:** Given $pk = mpk$ and $sk = msk$, conduct the following steps.

1. Compute $dk_{com} \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{KG}(pars, msk, com)$ and then set M' to be $\mathcal{IBE}.\mathsf{Dec}\big(pars, mpk, dk_{com}, com, C\big)$. Then, parse M' as $M\|k\|dec$ (and return $\perp$ if $M' = \perp$ or if M' cannot be parsed properly).
2. If $\mathsf{MacVer}_k\big(C, tag\big) = 1$ and $\mathsf{Ver}(cpars, mpk\|k, com, dec) = 1$, return M. Otherwise, return $\perp$.

A difference with the original Boneh–Katz construction—which can use a weak form of commitment called *encapsulation*—is that our construction requires a full-fledged commitment scheme. This is because, in order to achieve complete robustness, we need to commit to the master public key of the scheme at the same time as the MAC key in the encryption algorithm. Moreover, the proof of AI-CCA security requires the commitment to be a *trapdoor* commitment: the trapdoor plays an essential role when we reduce the sID-TAA-CPA security of the IBE to the AI-CCA security of the encryption scheme.

The proof of the following theorem can be found in [15, Appendix J].

Theorem 5. *If $\mathcal{IBE}$ is* sID-TAA-CPA*-secure, $\mathcal{MAC}$ is strongly unforgeable, and $\mathcal{TCMT}$ is a computationally binding trapdoor commitment scheme, then $\overline{\mathcal{PKE}}$ is* AI-CCA*-secure. Moreover, the scheme $\overline{\mathcal{PKE}}$ is* CROB *if $\mathcal{TCMT}$ is computationally binding.*

6 A Concrete CROB Scheme

In this section, we describe a simple way to achieve complete robustness using hybrid encryption where the symmetric component uses the encrypt-then-MAC approach. To this end, we require the MAC to satisfy a "MAC analogue" of the notion of committing symmetric encryption [16]. Informally this notion requires that a given MAC tag is valid for a single message regardless of the key used.

Committing MAC. We say $\mathcal{MAC} = (\mathsf{MacGen}, \mathsf{MacVer})$ is *committing* if for any message m and any key k, there exists no message-key pair (m', k') such that $m' \neq m$ and $\mathsf{MacVer}_{k'}(m', \mathsf{MacGen}_k(m)) = 1$.

We also need the MAC to computationally hide the message. Note that the following definition is implied by the definition of message-hiding security used in [14, Definition 2.2].

INDISTINGUISHABLE MAC. We say a message authentication code $\mathcal{MAC} = (\mathsf{MacGen}, \mathsf{MacVer})$ with key space KSp provides *indistinguishability* if, for any two messages m_0, m_1, it is computationally infeasible to distinguish the distributions $\mathcal{D}_b := \{tag \leftarrow\!\!\$\ \mathsf{MacGen}_k(m_b) : k \leftarrow\!\!\$\ \mathsf{KSp}\}$ for $b \in \{0,1\}$.

For our purposes, the MAC only has to provide one-time strong unforgeability. Namely, the adversary is allowed to see one pair of the form (m, tag), where $tag = \mathsf{MacGen}_k(m)$, and should not be able to produce a pair (m', tag') such that $(m', tag') \neq (m, tag)$ and $\mathsf{MacVer}_k(m', tag') = 1$.

Using ideas from [16], it is easy to construct a MAC which is simultaneously committing, indistinguishable, and strongly unforgeable. The idea is to use a family of *injective* and *key-binding* pseudorandom functions: for any distinct keys k_1, k_2, the functions $f_{k_1}(\cdot)$ and $f_{k_2}(\cdot)$ have disjoint ranges, i.e., there exist no two pairs $(k_1, x_1), (k_2, x_2)$ such that $k_1 \neq k_2$ and $f_{k_1}(x_1) = f_{k_2}(x_2)$. The key space of the MAC is that of the PRF. For any message $m \neq 1^\lambda$, the MAC generation computes and outputs the pair $(f_k(1^\lambda), f_k(m))$. The first component serves as a perfectly binding commitment to the key k while the injectivity of $f_k(\cdot)$ guarantees that the MAC is only valid for one message. In addition, its strong unforgeability and indistinguishability properties are both implied by the pseudorandomness of $\{f_k\}_k$ as long as the message space of the MAC, $\mathsf{MSp}^{\mathrm{mac}}$, does not include 1^λ (the proof is straightforward).

We show a simple variant of the Hofheinz–Kiltz (HK) hybrid encryption scheme [17] that provides CROB and AI-CCA security when the underlying authenticated symmetric encryption scheme uses a MAC with the aforementioned properties. Besides providing new ways to achieve robustness, our scheme comes with the advantage that its computational efficiency is the same as the original HK scheme and in particular it is more efficient than combining HK with a commitment using the ABN transformation.

$\mathsf{PG}(1^\lambda)$**:** Choose a group $\mathbb{G}$ of prime order $p > 2^\lambda$ with $g \leftarrow\!\!\$\ \mathbb{G}$. Also, choose a symmetric encryption scheme (E, D) of key length ℓ_0 and a message authentication code $\mathcal{MAC} = (\mathsf{MacGen}, \mathsf{MacVer})$ of key length ℓ_1. Finally, choose a key-derivation function $\mathsf{KDF} : \mathbb{G} \to \{0,1\}^{\ell_0+\ell_1}$, a target collision-resistant hash function[3] $\mathsf{TCR} : \mathbb{G} \to \mathbb{Z}_p$, and a collision-resistant hash function $H : \{0,1\}^* \to \mathsf{MSp}^{\mathrm{mac}}$, where $\mathsf{MSp}^{\mathrm{mac}}$ is the message space of $\mathcal{MAC}$. The public parameters consist of $pars := \big(\mathbb{G},\ p,\ g,\ (\mathsf{E}, \mathsf{D}),\ \mathcal{MAC},\ \mathsf{TCR},\ \mathsf{KDF},\ H\big)$.

$\mathsf{KG}(pars)$**:** Choose $x, y, z \leftarrow\!\!\$\ \mathbb{Z}_p^*$ and compute $u = g^x$, $v = g^y$, and $h = g^z$. The public key is $pk = \big(u, v, h\big)$ and the private key is $sk = (x, y, z) \in (\mathbb{Z}_p^*)^3$.

$\mathsf{Enc}\big(pars, pk, M\big)$**:** Choose $s \leftarrow\!\!\$\ \mathbb{Z}_p^*$ and compute

$$C_1 = g^s, C_2 = (u^\tau \cdot v)^s, C_3 \leftarrow\!\!\$\ \mathsf{E}_{K_0}(M), tag = \mathsf{MacGen}_{K_1}(H(C_3, u, v, h))$$

where $\tau = \mathsf{TCR}(C_1) \in \mathbb{Z}_p^*$ and $(K_0, K_1) = \mathsf{KDF}(h^s) \in \{0,1\}^{\ell_0+\ell_1}$. Return $C = (C_1, C_2, C_3, tag)$.

[3] As in [17], this function can be replaced by an injective encoding from $\mathbb{G}$ to $\mathbb{Z}_p$.

$\mathsf{Dec}(pars, pk, sk, C)$: Given $C = (C_1, C_2, C_3, tag)$, return $\perp$ if $C_2 \neq C_1^{\tau \cdot x + y}$, where $\tau = \mathsf{TCR}(C_1)$. Else, compute $(K_0, K_1) = \mathsf{KDF}(C_1^z)$ and $M \leftarrow \mathsf{D}_{K_0}(C_3)$. Return M if $\mathsf{MacVer}_{K_1}(H(C_3, pk), tag) = 1$. Else, return $\perp$.

The scheme was known to be IND-CCA-secure. We are also able to prove that it provides AI-CCA security, essentially because the ciphertexts can be shown to be indistinguishable from dummy ciphertexts that are statistically independent of the public key, even in the presence of a decryption oracle. Proofs of the following results may be found in [15, Appendix K].

Theorem 6. *The scheme provides* AI-CCA *security assuming that: (1) The DDH assumption holds in* $\mathbb{G}$*; (2)* (E, D) *is a semantically secure symmetric encryption scheme; (3)* KDF *is a secure key-derivation function;*[4] *(4)* $\mathcal{MAC}$ *is one-time strongly unforgeable and provides indistinguishability; (5)* H *and* TCR *are collision-resistant and target collision-resistant, respectively. Furthermore, the scheme is* CROB *if* H *is collision-resistant and* $\mathcal{MAC}$ *is committing.*

Interestingly, if the construction of Section 5.4 is modified to use a committing MAC, it can be instantiated using any commitment scheme and in particular a perfectly binding commitment or even an encapsulation scheme (as in the original Boneh–Katz construction) also work. In this case, the sender no longer needs to commit to the master public key: (com, dec) is generated by committing to the MAC key only. Instead, the sender computes tag as $tag = \mathsf{MacGen}_k(H(C, mpk))$ using a collision-resistant hash function H. If the MAC is committing, the resulting construction is easily seen to provide complete robustness. It also remains AI-CCA-secure provided the MAC satisfies the notion of indistinguishability.

7 Closing Remarks

Motivated in part by the shortcomings of existing definitions of robustness, we have made a thorough exploration of the landscape of robustness definitions and their relations, and given a suite of flexible and efficient methods for obtaining completely robust AI-CCA-secure public-key encryption schemes. In future work, one could explore the situation in the ID-based setting. Another open question, well beyond the remit of this paper, is to formalize the fairness of auctions and formally prove that our CROB notion is strong enough to ensure this property for Sako's protocol or its variants.

Acknowledgments. The authors would like to thank Mihir Bellare for his valuable comments. Pooya Farshim is supported by grant Fi 940/4-1 of the German Research Foundation (DFG). Part of this work was done while Benoît Libert was an F.R.S.-F.N.R.S. scientific collaborator at the Université catholique de Louvain (Belgium). Kenneth G. Paterson was supported by an EPSRC Leadership Fellowship, EP/H005455/1.

[4] The standard KDF security requires that no distinguisher can tell if it is given the output of the KDF for a random input or just a random element in the range of the KDF.

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions. Journal of Cryptology 21(3), 350–391 (2008)
2. Abdalla, M., Bellare, M., Neven, G.: Robust Encryption. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 480–497. Springer, Heidelberg (2010)
3. 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)
4. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-Privacy in Public-Key Encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (2001)
5. Bellare, M., Kohno, T.: A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 491–506. Springer, Heidelberg (2003)
6. Black, J., Rogaway, P., Shrimpton, T.: Encryption-Scheme Security in the Presence of Key-Dependent Messages. In: Nyberg, K., Heys, H. (eds.) SAC 2002. LNCS, vol. 2595, pp. 62–75. Springer, Heidelberg (2003)
7. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
8. Boneh, D., Katz, J.: Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 87–103. Springer, Heidelberg (2005)
9. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
10. Boyen, X., Waters, B.: Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
11. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
12. Cramer, R., Shoup, V.: A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
13. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing 33, 167–226 (2003)
14. Dodis, Y., Kiltz, E., Pietrzak, K., Wichs, D.: Message Authentication, Revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 355–374. Springer, Heidelberg (2012)
15. Farshim, P., Libert, B., Paterson, K.G., Quaglia, E.A.: Robust encryption, revisited. Cryptology ePrint Archive, Report 2012/673 (2012), Full version of this paper, `http://eprint.iacr.org/`
16. Fischlin, M.: Pseudorandom Function Tribe Ensembles Based on One-Way Permutations: Improvements and Applications. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 432–445. Springer, Heidelberg (1999)

17. Hofheinz, D., Kiltz, E.: Secure Hybrid Encryption from Weakened Key Encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (2007)
18. Hofheinz, D., Weinreb, E.: Searchable encryption with decryption in the standard model. Cryptology ePrint Archive, Report 2008/423 (2008), http://eprint.iacr.org/
19. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
20. Kurosawa, K., Desmedt, Y.: A New Paradigm of Hybrid Encryption Scheme. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 426–442. Springer, Heidelberg (2004)
21. Libert, B., Paterson, K.G., Quaglia, E.A.: Anonymous Broadcast Encryption: Adaptive Security and Efficient Constructions in the Standard Model. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 206–224. Springer, Heidelberg (2012)
22. Mohassel, P.: A Closer Look at Anonymity and Robustness in Encryption Schemes. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 501–518. Springer, Heidelberg (2010)
23. Sako, K.: An Auction Protocol Which Hides Bids of Losers. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 422–432. Springer, Heidelberg (2000)
24. Shoup, V.: A proposal for an ISO standard for public key encryption (version 2.1). Cryptology ePrint Archive, Report 2001/112 (2001), http://eprint.iacr.org/

Sender-Equivocable Encryption Schemes Secure against Chosen-Ciphertext Attacks Revisited*

Zhengan Huang[1], Shengli Liu[1], and Baodong Qin[1,2]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[2] College of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621000, China
zhahuang.sjtu@gmail.com, {slliu,qinbaodong}@sjtu.edu.cn

Abstract. In Eurocrypt 2010, Fehr et al. proposed the first sender-equivocable encryption scheme secure against chosen-ciphertext attacks (NC-CCA) and proved that NC-CCA security implies security against selective opening chosen-ciphertext attacks (SO-CCA). The NC-CCA security proof of the scheme relies on security against substitution attacks of a new primitive, "cross-authentication code". However, the security of cross-authentication code can not be guaranteed when all the keys used in the code are exposed. Our key observation is that in the NC-CCA security game, the randomness used in the generation of the challenge ciphertext is exposed to the adversary. This random information can be used to recover all the keys involved in the cross-authentication code, and forge a ciphertext (like a substitution attack of cross-authentication code) that is different from but related to the challenge ciphertext. And the response of the decryption oracle, with respect to the forged ciphertext, leaks information. This leaked information can be employed by an adversary to spoil the NC-CCA security proof of Fehr et al.'s scheme encrypting multi-bit plaintexts. We also show that Fehr et al.'s scheme encrypting single-bit plaintexts can be refined to achieve NC-CCA security, free of any cross-authentication code.

Keywords: sender-equivocable encryption, chosen-ciphertext attack, cross-authentication code.

1 Introduction

The notion of sender equivocability for a public-key encryption (PKE) scheme was formalized by Fehr et al. [7] in Eurocrypt 2010. It is an important tool to construct PKE schemes secure against chosen-plaintext/ciphertext selective opening attacks (SO-CPA/CCA). Sender equivocability focuses on the ability

* Funded by NSFC (Nos. 61170229, 61133014), Innovation Project (No.12ZZ021) of Shanghai Municipal Education Commission, and Specialized Research Fund (No. 20110073110016) for the Doctoral Program of Higher Education.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 369–385, 2013.

of a PKE scheme to generate some "equivocable" ciphertexts which can be efficiently opened arbitrarily. More specifically, a PKE scheme is called sender-equivocable, if there is a simulator which can generate non-committing ciphertexts and later open them to any requested plaintexts by releasing some randomness, such that the simulation and real encryption are indistinguishable. This notion is similar to non-committing encryption [5]. In fact, Fehr et al. [7] have pointed out that sender-equivocable encryption secure under chosen-plaintext attacks (CPA) is a variant of non-committing encryption defined in [5]. Following the notations in [7], security of a sender-equivocable encryption scheme against chosen-plaintext/ciphertext attacks is denoted by *NC-CPA/CCA security*.

As proved in [7], NC-CPA/CCA security implies simulation-based selective opening security against chosen-plaintext/ciphertext attacks (SIM-SO-CPA/CCA security). This fact suggests an alternative way of constructing PKE secure against selective opening attacks, besides the construction from lossy encryption proposed in [3].

Discussion and Related Work. In Eurocrypt 2009, Bellare et al. [3] formalized the notion of security against selective opening attacks (SOA security) for sender corruptions. This security notion captures a situation that n senders encrypt their own messages and send the ciphertexts to a single receiver. Some subset of the senders can be corrupted by an adversary, exposing their messages and randomness to the adversary. SOA security requires that the unopened ciphertexts remain secure.

In [3], Bellare et al. proposed two kinds of SOA security: simulation-based selective opening (SIM-SO) security and indistinguishability-based selective opening (IND-SO) security. The relations between the two notions are figured out by Böhl et al. [2]. Bellare et al. [1] showed that the standard security of PKE does not imply SIM-SO security. Bellare et al. [3] proposed that IND-SO-CPA security and SIM-SO-CPA security can be achieved through a special class of encryption named lossy encryption, and lossy encryption can be constructed from lossy trapdoor functions [13]. Hemenway et al. [10] showed more constructions of lossy encryption, which achieved IND-SO-CCA security with a-priori bounded number of challenge ciphertexts. In Eurocrypt 2012, Hofheinz [9] proposed a new primitive called all-but-many lossy trapdoor functions, which were employed to construct IND-SO-CCA secure and SIM-SO-CCA secure PKE with unbounded number of challenge ciphertexts. In [4], Bellare et al. extended SOA security from PKE to IBE.

In [7], Fehr et al. presented a totally different way of achieving SIM-SO-CCA security, also with unbounded number of challenge ciphertexts. They formalized the security notion of sender equivocability under chosen-plaintext/ciphertext attacks (NC-CPA/CCA security), and proved that NC-CPA (resp. NC-CCA) security implies SIM-SO-CPA (resp. SIM-SO-CCA) security. In [7], two PKE schemes were proposed. The first one, constructed from trapdoor one-way permutations, is NC-CPA secure, so it is SIM-SO-CPA secure. The second one (denoted by the FHKW scheme) is constructed from an extended hash proof

system [6] and a new primitive, "cross-authentication code". They proved that the FHKW scheme is NC-CCA secure.

With help of similar techniques as those in the FHKW scheme, Gao et al. [8] presented a deniable encryption scheme in 2012. The CCA security of their scheme was guaranteed mainly by an extended hash proof system of [6] and a cross-authentication code of [7].

In this paper, we will analyze the security proof of the FHKW scheme and show that its NC-CCA security can not be guaranteed by their proof. The GXW scheme suffers from the similar security problem. We also offer a refined version of the FHKW scheme for single bit with NC-CCA security.

Our Contribution. In this paper, we focus on NC-CCA security.

- We provide an analysis of the security proof of the FHKW scheme in [7], and show the proof of NC-CCA security in [7] is flawed by showing an attack. The key observation is: In the definition of NC-CCA security, the randomness used in the generation of the challenge ciphertext C^* is offered to the adversary. The adversary is able to use the randomness to forge a ciphertext and obtain useful information by querying the forged ciphertext to the decryption oracle. Assume that the plaintext consists of L bits. We present a PPT adversary who can always distinguish the real experiment and the simulated experiment for $L > 1$. We also show that the security requirement of "L-cross-authentication codes" is not enough in the proof of NC-CCA security in [7] for any positive integer L.
- We refine the FHKW scheme encrypting one bit. Although we showed that "L-cross-authentication codes" are generally not sufficient to prove NC-CCA security, some specific instances of "1-cross-authentication codes" are helpful to finish the proof of NC-CCA security of the FHKW scheme [7], but limited to encryption of a single bit. We provide a simpler encryption scheme for single-bit plaintexts, free of any cross-authentication code.

Organization. We start by notations and definitions in Section 2. We recall the FHKW scheme in Section 3, and then provide a security analysis of it in Section 4. We present a refined version of the FHKW scheme for single-bit plaintexts in Section 5 and leave the proof in the Appendix. Finally, we give a summary of our work in Section 6.

2 Preliminaries

2.1 Notations

Let $\mathbb{N}$ denote the set of natural numbers. We use $k \in \mathbb{N}$ as the security parameter throughout the paper. For $n \in \mathbb{N}$, let $[n]$ denote the set $\{1, 2, \cdots, n\}$ and $\{0,1\}^n$ the set of bitstrings of length n. For a finite set S, let $s \leftarrow S$ denote the process of sampling s uniformly at random from S. If A is a probabilistic algorithm,

we denote by $\mathcal{R}_A$ the randomness set of A. Let $y \leftarrow A(x_1, x_2, \cdots, x_t)$ denote the process of running A on inputs $\{x_1, x_2, \cdots, x_t\}$ and inner randomness $R \leftarrow \mathcal{R}_A$, and outputting y. If the running time of probabilistic algorithm A is polynomial in k, then A is a probabilistic polynomial time (PPT) algorithm.

2.2 Sender-Equivocable Encryption Schemes

The notion of Sender Equivocability was formalized by Fehr et al. [7] in 2010. For a public-key encryption scheme $\prod = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, let $A = (A_1, A_2)$ denote a stateful adversary, $S = (S_1, S_2)$ denote a stateful simulator, and M denote a plaintext. Let *state* denote some state information output by A_1 and then is passed to A_2. Sender equivocability under adaptive chosen-ciphertext attacks is defined through the following two experiments.

Experiment $\mathbf{Exp}^{\text{NC-CCA-Real}}_{\prod,A}(k)$**:**
$(pk, sk) \leftarrow \mathsf{Gen}(1^k)$
$(M, state) \leftarrow A_1^{\mathsf{Dec}_{sk}(\cdot)}(pk)$
$R \leftarrow \mathcal{R}_{\mathsf{Enc}}$
$C \leftarrow \mathsf{Enc}_{pk}(M; R)$
return $A_2^{\mathsf{Dec}_{sk}(\cdot)}(M, C, R, state)$

Experiment $\mathbf{Exp}^{\text{NC-CCA-Sim}}_{\prod,A}(k)$**:**
$(pk, sk) \leftarrow \mathsf{Gen}(1^k)$
$(M, state) \leftarrow A_1^{\mathsf{Dec}_{sk}(\cdot)}(pk)$
$C \leftarrow S_1(pk, 1^{|M|})$
$R \leftarrow S_2(M)$
return $A_2^{\mathsf{Dec}_{sk}(\cdot)}(M, C, R, state)$

In both experiments, $A = (A_1, A_2)$ is allowed to access to a decryption oracle $\mathsf{Dec}_{sk}(\cdot)$ with constraint that A_2 is not allowed to query C.

The advantage of adversary A is defined as follows.

$$\mathbf{Adv}^{\text{NC-CCA}}_{\prod,A,S}(k) := \left|\Pr\left[\mathsf{Exp}^{\text{NC-CCA-Real}}_{\prod,A}(k) = 1\right] - \Pr\left[\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\prod,A}(k) = 1\right]\right|.$$

Definition 1 (NC-CCA security). *A public-key encryption scheme* $\prod = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* sender-equivocable under adaptive chosen-ciphertext attacks (NC-CCA secure)*, if there is a stateful PPT algorithm S (the simulator), such that for any PPT algorithm A (the adversary), the advantage* $\mathbf{Adv}^{\text{NC-CCA}}_{\prod,A,S}(k)$ *is negligible.*

2.3 Building Blocks of the FHKW Scheme

In [7], Fehr et al. presented a construction of PKE with NC-CCA security. We will call their scheme the FHKW scheme. The FHKW scheme was built from the following cryptographic primitives: a collision-resistant hash function (informally, a function is collision-resistant if any PPT adversary cannot find two distinct inputs hashing to the same output except with negligible probability), a subset membership problem, an extended version of hash proof system [6], and a cross-authentication code [7].

Definition 2 (Subset membership problem). *A* subset membership problem *consists of the following PPT algorithms.*

- $\mathsf{SmpGen}(1^k)$: *On input* 1^k*, algorithm* SmpGen *outputs a parameter* Λ*, which specifies a set* $\mathcal{X}_\Lambda$ *and its subset* $\mathcal{L}_\Lambda \subseteq \mathcal{X}_\Lambda$*. Set* $\mathcal{X}_\Lambda$ *is required to be easily recognizable with* Λ*.*
- $\mathsf{SampleL}(\mathcal{L}_\Lambda; W)$: *Algorithm* SampleL *samples* $X \in \mathcal{L}_\Lambda$ *using randomness* $W \in \mathcal{R}_{\mathsf{SampleL}}$*.*

A subset membership problem SMP *is* hard*, if for any PPT distinguisher* D*,* D*'s advantage*

$$\begin{aligned}\mathbf{Adv}_{\mathsf{SMP},D}(k) := \mid \Pr[\Lambda \leftarrow \mathsf{SmpGen}(1^k), X \leftarrow \mathcal{L}_\Lambda : D(X) = 1] \\ - \Pr[\Lambda \leftarrow \mathsf{SmpGen}(1^k), X \leftarrow \mathcal{X}_\Lambda : D(X) = 1] \mid\end{aligned}$$

is negligible.

Definition 3 (Subset sparseness). *A subset membership problem* SMP *has the property of* subset sparseness*, if the probability* $\Pr[\Lambda \leftarrow \mathsf{SmpGen}(1^k), X \leftarrow \mathcal{X}_\Lambda : X \in \mathcal{L}_\Lambda]$ *is negligible.*

Definition 4 (Hash Proof System and Extended Hash Proof System). *A* hash proof system HPS *for a subset membership problem* SMP *associates each* $\Lambda \leftarrow \mathsf{SmpGen}(1^k)$ *with an efficiently recognizable key space* $\mathcal{K}_\Lambda$ *and the following PPT algorithms:*

- $\mathsf{HashGen}(\Lambda)$: *On input* Λ*,* HashGen *outputs a public key* hpk *and a secret key* hsk*, both containing the parameter* Λ*.*
- $\mathsf{SecEvl}(hsk, X)$: *It is a deterministic algorithm. On input a secret key* hsk *and an element* $X \in \mathcal{X}_\Lambda$*,* SecEvl *outputs a key* $K \in \mathcal{K}_\Lambda$*.*
- $\mathsf{PubEvl}(hpk, X, W)$: *It is a deterministic algorithm. On input a public key* hpk*, an element* $X \in \mathcal{X}_\Lambda$ *and a witness* W *for* $X \in \mathcal{L}_\Lambda$*,* PubEvl *outputs a key* $K \in \mathcal{K}_\Lambda$*. The correctness requires that* $\mathsf{PubEvl}(hpk, X, W) = \mathsf{SecEvl}(hsk, X)$ *for all* $\Lambda \leftarrow \mathsf{SmpGen}(1^k)$*,* $(hpk, hsk) \leftarrow \mathsf{HashGen}(\Lambda)$ *and* $X \leftarrow \mathsf{SampleL}(\mathcal{L}_\Lambda; W)$*.*

An extended hash proof system EHPS *is a variation of a hash proof system* HPS*, extending the sets* $\mathcal{X}_\Lambda$ *and* $\mathcal{L}_\Lambda$ *by taking the Cartesian product of these sets with an efficiently recognizable tag space* $\mathcal{T}_\Lambda$*. Hence, the tuple of the three algorithms* (HashGen, SecEvl, PubEvl) *of* EHPS *is changed to* $(hpk, hsk) \leftarrow \mathsf{HashGen}(\Lambda)$*,* $K \leftarrow \mathsf{SecEvl}(hsk, X, t)$ *and* $K \leftarrow \mathsf{PubEvl}(hpk, X, W, t)$*, with* $t \in \mathcal{T}_\Lambda$*.*

The public key hpk in a hash proof system HPS uniquely determines the action of algorithm SecEvl for all $X \in \mathcal{L}_\Lambda$. However, the action of SecEvl for $X \in \mathcal{X}_\Lambda \backslash \mathcal{L}_\Lambda$ is still undetermined by hpk. This is defined by a *perfectly* 2*-universal* property.

Definition 5 (perfectly 2-universal). *A hash proof system* HPS *for* SMP *is* perfectly 2-universal *if for any* $\Lambda \leftarrow \mathsf{SmpGen}(1^k)$*, any* hpk *from* $\mathsf{HashGen}(\Lambda)$*, any distinct* $X_1, X_2 \in \mathcal{X}_\Lambda \setminus \mathcal{L}_\Lambda$*, and any* $K_1, K_2 \in \mathcal{K}_\Lambda$*,*

$$\Pr[\mathsf{SecEvl}(hsk, X_2) = K_2 \mid \mathsf{SecEvl}(hsk, X_1) = K_1] = \frac{1}{|\mathcal{K}_\Lambda|},$$

where the probability is taken over all possible hsk *with* $(hpk, hsk) \leftarrow \mathsf{HashGen}(\Lambda)$*.*

Definition 6 (Efficiently samplable and explainable domain). *A domain $\mathcal{D}$ is* efficiently samplable and explainable*, if there exists two PPT algorithms:*

- $\mathsf{Sample}(\mathcal{D}; R)$: *On input a random coin $R \leftarrow \mathcal{R}_{\mathsf{Sample}}$ and a domain $\mathcal{D}$, it outputs an element uniformly distributed over $\mathcal{D}$.*
- $\mathsf{Explain}(\mathcal{D}, x)$: *On input $\mathcal{D}$ and $x \in \mathcal{D}$, this algorithm outputs R that is uniformly distributed over the set $\{R \in \mathcal{R}_{\mathsf{Sample}} \mid \mathsf{Sample}(\mathcal{D}; R) = x\}$.*

Definition 7 (L-Cross-Authentication Code [7]). *For any $L \in \mathbb{N}$, an L-*cross-authentication code XAC*, associated with a key space $\mathcal{XK}$ and a tag space $\mathcal{XT}$, consists of three PPT algorithms (XGen, XAuth, XVer). Algorithm $\mathsf{XGen}(1^k)$ generates a uniformly random key $K \in \mathcal{XK}$, $\mathsf{XAuth}(K_1, \cdots, K_L)$ produces a tag $T \in \mathcal{XT}$, and $\mathsf{XVer}(K, i, T)$ outputs $b \in \{0, 1\}$. The following properties are required:*

Correctness. *The function*

$$\mathsf{fail}^{correct}_{\mathsf{XAC}}(k) := \max_{i \in [L]} \Pr[\mathsf{XVer}(K_i, i, \mathsf{XAuth}(K_1, \cdots, K_L)) \neq 1]$$

is negligible in k, where the max *is over all $i \in [L]$ and the probability is taken over all possible $K_1, \cdots, K_L \leftarrow \mathsf{XGen}(1^k)$.*

Security against impersonation and substitution attacks. *The advantages $\mathbf{Adv}^{imp}_{\mathsf{XAC}}(k)$ and $\mathbf{Adv}^{sub}_{\mathsf{XAC}}(k)$, defined as follows, are both negligible.*

$$\mathbf{Adv}^{imp}_{\mathsf{XAC}}(k) := \max_{i, T'} \Pr[K \leftarrow \mathsf{XGen}(1^k) : \mathsf{XVer}(K, i, T') = 1]$$

where the max *is over all $i \in [L]$ and $T' \in \mathcal{XT}$.*

$$\mathbf{Adv}^{sub}_{\mathsf{XAC}}(k) := \max_{i, K_{\neq i}, \mathsf{Func}} \Pr\left[\begin{array}{l} K_i \leftarrow \mathsf{XGen}(1^k) \\ T \leftarrow \mathsf{XAuth}(K_1, \cdots, K_L) \\ T' \leftarrow \mathsf{Func}(T) \end{array} : \begin{array}{l} T' \neq T \wedge \\ \mathsf{XVer}(K_i, i, T') = 1 \end{array}\right]$$

where the max *is over all $i \in [L]$, all $K_{\neq i} := (K_j)_{j \neq i} \in \mathcal{XK}^{L-1}$ and all possibly randomized functions $\mathsf{Func} : \mathcal{XT} \to \mathcal{XT}$.*

3 Review on the FHKW Scheme in [7]

With the above cryptographic primitives, we now present the FHKW scheme [7].

Let SMP be a hard subset membership problem that has the property of subset sparseness. Let $\mathcal{X}_\Lambda$, with $\Lambda \leftarrow \mathsf{SmpGen}(1^k)$, be efficiently samplable and explainable. Let EHPS be a perfectly 2-universal extended hash proof system for SMP with tag space $\mathcal{T}_\Lambda$ and key space (range) $\mathcal{K}_\Lambda$, which is efficiently samplable and explainable as well. Let $\mathcal{H} : (\mathcal{X}_\Lambda)^L \to \mathcal{T}_\Lambda$ be a family of collision-resistant hash

functions, and XAC be an L-cross-authentication code with key space $\mathcal{XK} = \mathcal{K}_\Lambda$ and tag space $\mathcal{XT}$.

The FHKW Scheme

$\mathsf{Gen}(1^k)$: On input 1^k, algorithm Gen runs $\Lambda \leftarrow \mathsf{SmpGen}(1^k)$, $(hpk, hsk) \leftarrow \mathsf{HashGen}(\Lambda)$, $\mathsf{H} \leftarrow \mathcal{H}$, and outputs (pk, sk), where $pk = (hpk, \mathsf{H})$ and $sk = (hsk, \mathsf{H})$.

$\mathsf{Enc}(pk, M; R)$: To encrypt a plaintext $M = (M_1, \cdots, M_L) \in \{0,1\}^L$ under a public key $pk = (hpk, \mathsf{H})$ with randomness $R = (W_i, R_i^{\mathcal{X}_\Lambda}, R_i^{\mathcal{K}_\Lambda})_{i\in[L]} \in (\mathcal{R}_{\mathsf{SampleL}} \times \mathcal{R}_{\mathsf{Sample}} \times \mathcal{R}_{\mathsf{Sample}})^L$, algorithm Enc runs as follows:
For $i \in [L]$, set

$$X_i := \begin{cases} \mathsf{Sample}(\mathcal{X}_\Lambda; R_i^{\mathcal{X}_\Lambda}) & \text{if } M_i = 0 \\ \mathsf{SampleL}(\mathcal{L}_\Lambda; W_i) & \text{if } M_i = 1 \end{cases}$$

and $t := \mathsf{H}(X_1, \cdots, X_L)$. Then for $i \in [L]$, set the keys

$$K_i := \begin{cases} \mathsf{Sample}(\mathcal{K}_\Lambda; R_i^{\mathcal{K}_\Lambda}) & \text{if } M_i = 0 \\ \mathsf{PubEvl}(hpk, X_i, W_i, t) & \text{if } M_i = 1 \end{cases}$$

and the tag $T := \mathsf{XAuth}(K_1, \cdots, K_L)$. Finally, return $C = (X_1, \cdots, X_L, T)$ as the ciphertext.

$\mathsf{Dec}(sk, C)$: To decrypt a ciphertext $C = (X_1, \cdots, X_L, T) \in \mathcal{X}_\Lambda^L \times \mathcal{XT}$ under a secret key $sk = (hsk, \mathsf{H})$, algorithm Dec computes $t = \mathsf{H}(X_1, \cdots, X_L)$, for $i \in [L]$ sets $\overline{K_i} := \mathsf{SecEvl}(hsk, X_i, t)$ and $M_i = \mathsf{XVer}(\overline{K_i}, i, T)$, and returns $M = (M_1, \cdots, M_L)$ as the plaintext.

The correctness of the FHKW scheme is proved by [7], which we omit here.

4 Security Analysis of the FHKW Scheme

According to the definition of NC-CCA security, the FHKW scheme is NC-CCA secure, if and only if there exists a simulator S such that for any PPT algorithm A, the two experiments $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$, defined in Section 2, are indistinguishable.

In order to prove NC-CCA security of the FHKW scheme, Fehr et al. [7] constructed the following simulator $S = (S_1, S_2)$.

Simulator S:

– $S_1(pk, 1^{|M|})$: Parse $pk = (hpk, \mathsf{H})$. For $i \in [L]$, choose $\widetilde{W_i} \leftarrow \mathcal{R}_{\mathsf{SampleL}}$ and set $X_i := \mathsf{SampleL}(\mathcal{L}_\Lambda; \widetilde{W_i})$. Compute $t := \mathsf{H}(X_1, \cdots, X_L)$. For $i \in [L]$, set $K_i := \mathsf{PubEvl}(hpk, X_i, \widetilde{W_i}, t)$. Set $T \leftarrow \mathsf{XAuth}(K_1, \cdots, K_L)$. Return the ciphertext $C = (X_1, \cdots, X_L, T)$.

– $S_2(M)$: Parse $M = (M_1, \cdots, M_L)$. For $i \in [L]$, if $M_i = 1$, set $W_i := \widetilde{W_i}$, and choose $R_i^{\mathcal{X}_\Lambda} \leftarrow \mathcal{R}_{\mathsf{Sample}}$, $R_i^{\mathcal{K}_\Lambda} \leftarrow \mathcal{R}_{\mathsf{Sample}}$; else, choose $W_i \leftarrow \mathcal{R}_{\mathsf{SampleL}}$, and set $R_i^{\mathcal{X}_\Lambda} \leftarrow \mathsf{Explain}(\mathcal{X}_\Lambda, X_i)$, $R_i^{\mathcal{K}_\Lambda} \leftarrow \mathsf{Explain}(\mathcal{K}_\Lambda, K_i)$. Return the randomness $R = (W_i, R_i^{\mathcal{X}_\Lambda}, R_i^{\mathcal{K}_\Lambda})_{i \in [L]}$.

With simulator S, Fehr et al. [7] proved that the FHKW scheme is NC-CCA secure. However, we will show that this specific simulator S does not guarantee NC-CCA security of the FHKW scheme for any positive integer L.

4.1 The Problem of Security Proof in [7]

To prove NC-CCA security, it is essential to show that the decryption oracle will not leak any useful information to any PPT adversary. As to the FHKW scheme, given a challenge ciphertext $C = (X_1, \cdots, X_L, T)$, an adversary A comes up with a decryption query $C' = (X_1, \cdots, X_L, T')$ where $T' \neq T$. NC-CCA security expects the decryption of C' by the oracle will not help the adversary to distinguish the two experiments $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$(see the proof of [7, Lemma 5]). This strongly relies on the security against substitution attacks of cross-authentication code, which requires that "given T and $K_{\neq i}$, it is difficult to output a $T' \neq T$ such that $\mathsf{XVer}(K_i, i, T') = 1$, where K_i is uniformly distributed". However, in the NC-CCA game, adversary A KNOWs K_i for any $i \in [L]$! The reason is as follows. Upon returning a plaintext M, adversary A receives not only a challenge ciphertext C, but also some related random coins R which are supposed to have been consumed in the challenge ciphertext generation. With R and M, adversary A can recover K_i for any $i \in [L]$. Then, it is possible for A to output a $T' \neq T$ such that $\mathsf{XVer}(K_i, i, T') = 1$. Hence, the XAC's security against substitution attacks is not sufficient to guarantee the aforementioned property. That is why the security proof of [7] fails (more precisely, the proof of [7, Lemma 5] fails).

In fact, this kind of adversary, which can output a $T' \neq T$ such that $\mathsf{XVer}(K_i, i, T') = 1$ given T and K_i for any $i \in [L]$, does exist. In Section 4.2, we will present such an adversary A to destroy the security proof of the FHKW scheme for $L > 1$.

Gao et al.'s Deniable Scheme in [8]. In [8], Gao et al. utilized exactly the same technique as that in the FHKW scheme to construct a deniable encryption scheme and "proved" the CCA security. The similar problem we pointed out above also exists in their security proof (more specifically, the proof of [8, Claim 1]). As a result, our following attack in Section 4.2 applies to their scheme and ruins their proof, too.

4.2 Security Analysis of the FHKW Scheme - $L > 1$

Before going into a formal statement and its proof, we briefly give a high-level description of our security analysis for $L > 1$.

With the aforementioned simulator S, for any $L > 1$, our aim is to construct an adversary $A = (A_1, A_2)$ to distinguish the two experiments $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$. The construction of adversary A is as follows.

In an experiment environment (either $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ or $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$), upon receiving pk, A_1 returns $M = (0, \cdots, 0)$. Then, upon receiving a ciphertext $C = (X_1, \cdots, X_L, T)$ and randomness R, A_2 returns $C' = (X_1, \cdots, X_L, T')$ as his decryption query, where $T' \leftarrow \mathsf{XAuth}(K'_1, K_2, \cdots, K_L)$, K'_1 is uniformly random chosen from $\mathcal{K}_\Lambda$ and $K_2, \cdots, K_L$ are all recovered from R. Finally, if the decryption oracle returns $M' = (0, \cdots, 0)$, A_2 will output $b = 1$, and otherwise, A_2 will output $b = 0$.

Now, we consider the probabilities that A outputs 1 in the two experiments, respectively. In $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$, for $i \in [L]$, X_i (resp. K_i) is chosen uniformly random from $\mathcal{X}_\Lambda$ (resp. $\mathcal{K}_\Lambda$), so the subset sparseness of SMP and the perfect 2-universality of HPS guarantee that for $i \in [L]$, $\overline{K'_i} = \mathsf{SecEvl}(hsk, X_i, t)$ is uniformly random in $\mathcal{K}_\Lambda$ from A's point of view. Due to the security of XAC, the decryption oracle returns $M' = (0, 0, ..., 0)$ for the queried ciphertext C'. Consequently, A outputs $b = 1$ with overwhelming probability in $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$. On the other hand, in $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$, for $i \in [L]$, X_i is chosen uniformly random from $\mathcal{L}_\Lambda$ and $K_i = \mathsf{PubEvl}(hpk, X_i, W_i, t)$, so the property of HPS guarantees that for $i \in [L]$, $\overline{K'_i} = \mathsf{SecEvl}(hsk, X_i, t) = K_i$. Due to the correctness of XAC and the facts that $T' \leftarrow \mathsf{XAuth}(K'_1, K_2, \cdots, K_L)$ and $M'_i = \mathsf{XVer}(\overline{K'_i}, i, T') = 1$ for $i \in \{2, 3, \cdots, L\}$, the decryption oracle returns $M' = (0, 1, \cdots, 1)$ with overwhelming probability. As a result, A outputs $b = 1$ with negligible probability in $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$. The two experiments $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$ have been distinguished by A with overwhelming advantage.

A formal statement of the result and its corresponding proof are as follows.

Theorem 1. *With the aforementioned simulator S, the* FHKW *scheme cannot be proved to be NC-CCA secure for any $L > 1$. More specifically, there exists an adversary A distinguishing the real and the simulated NC-CCA experiments, with advantage*

$$\mathbf{Adv}^{\text{NC-CCA}}_{\text{FHKW},A,S}(k) \geq 1 - 2\mathbf{Adv}^{imp}_{\mathsf{XAC}}(k) - \mathsf{fail}^{correct}_{\mathsf{XAC}}(k).$$

Proof. For simplicity, we consider the case of $L = 2$. We note that this attack is applicable to any $L > 1$.

Our aim is to construct a specific adversary $A = (A_1, A_2)$ to distinguish the two experiments $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$ with non-negligible advantage.

Specifically, given an experiment environment (either $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ or $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$), the adversary $A = (A_1, A_2)$ behaves as follows.

- Upon receiving $pk = (hpk, \mathsf{H})$, A_1 returns $M = (0, 0)$, i.e. $M_1 = M_2 = 0$.
- Upon receiving a ciphertext $C = (X_1, X_2, T)$ and randomness $R = ((W_1, R_1^{\mathcal{X}_\Lambda}, R_1^{\mathcal{K}_\Lambda}), (W_2, R_2^{\mathcal{X}_\Lambda}, R_2^{\mathcal{K}_\Lambda}))$, A_2 creates a new ciphertext C' according to C.

- Set $X_1' := X_1$, $X_2' := X_2$.
- Set $K_1' \leftarrow \mathcal{K}_\Lambda, K_2' \leftarrow \mathsf{Sample}(\mathcal{K}_\Lambda; R_2^{\mathcal{K}_\Lambda})$.
- Compute $T' \leftarrow \mathsf{XAuth}(K_1', K_2')$.
- Check that $T' \neq T$. If $T' = T$, choose another random value for K_1' and repeat the above steps, until $T' \neq T$.
- Set $C' := (X_1', X_2', T')$.

Then A_2 submits C' to the decryption oracle.

– Let $M' \leftarrow \mathsf{Dec}(sk, C')$. A_2 outputs b, where

$$b = \begin{cases} 1 & \texttt{if } M' = (0,0); \\ 0 & \texttt{if } M' \neq (0,0). \end{cases}$$

Now we analyze the probabilities that A_2 outputs $b = 1$ in the real experiment and the simulated experiment, respectively.

In both experiments, A_2 receives a ciphertext $C = (X_1, X_2, T)$ and randomness $R = ((W_1, R_1^{\mathcal{X}_\Lambda}, R_1^{\mathcal{K}_\Lambda}), (W_2, R_2^{\mathcal{X}_\Lambda}, R_2^{\mathcal{K}_\Lambda}))$. The ciphertext created and submitted to the decryption oracle by A_2 is $C' = (X_1', X_2', T') = (X_1, X_2, T')$, where $T' = \mathsf{XAuth}(K_1', K_2') = \mathsf{XAuth}(K_1', K_2)$ (due to $K_2' = K_2$) and $T' \neq T$.

The Real Experiment. The challenge ciphertext $C = (X_1, X_2, T)$ satisfies $X_1 \leftarrow \mathsf{Sample}(\mathcal{X}_\Lambda; R_1^{\mathcal{X}_\Lambda})$, $X_2 \leftarrow \mathsf{Sample}(\mathcal{X}_\Lambda; R_2^{\mathcal{X}_\Lambda})$, and $T = \mathsf{XAuth}(K_1, K_2)$, where $K_1 \leftarrow \mathsf{Sample}(\mathcal{K}_\Lambda; R_1^{\mathcal{K}_\Lambda})$ and $K_2 \leftarrow \mathsf{Sample}(\mathcal{K}_\Lambda; R_2^{\mathcal{K}_\Lambda})$.
The decryption of C' by the decryption oracle $\mathsf{Dec}(sk, \cdot)$ involves the computation of $t' := \mathsf{H}(X_1', X_2') = \mathsf{H}(X_1, X_2) = t$ and $\overline{K_i'} := \mathsf{SecEvl}(hsk, X_i', t') = \mathsf{SecEvl}(hsk, X_i, t)$, for $i \in \{1, 2\}$.
Due to the perfect 2-universality of EHPS, $\overline{K_i'}$ is uniformly random distributed in $\mathcal{K}_\Lambda$. Hence, for $i \in \{1, 2\}$,

$$\Pr\left[\mathsf{XVer}(\overline{K_i'}, i, T') = 1 \mid \text{in } \mathsf{Exp}_{\mathrm{FHKW},A}^{\text{NC-CCA-Real}}(k)\right] \leq \mathbf{Adv}_{\mathsf{XAC}}^{imp}(k).$$

Let $M' = (M_1', M_2')$ denote the decryption result of C' by the decryption oracle $\mathsf{Dec}(sk, \cdot)$. Then for $i \in \{1, 2\}$,

$$\begin{aligned} &\Pr\left[M_i' = 1 \mid \text{in } \mathsf{Exp}_{\mathrm{FHKW},A}^{\text{NC-CCA-Real}}(k)\right] \\ =\ &\Pr\left[\mathsf{XVer}(\overline{K_i'}, i, T') = 1 \mid \text{in } \mathsf{Exp}_{\mathrm{FHKW},A}^{\text{NC-CCA-Real}}(k)\right] \\ \leq\ &\mathbf{Adv}_{\mathsf{XAC}}^{imp}(k). \end{aligned}$$

The probability that A_2 outputs $b = 1$ in the real experiment is given by

$$\begin{aligned} &\Pr\left[\mathsf{Exp}_{\mathrm{FHKW},A}^{\text{NC-CCA-Real}}(k) = 1\right] \\ =\ &\Pr\left[M' = (0,0) \mid \text{in } \mathsf{Exp}_{\mathrm{FHKW},A}^{\text{NC-CCA-Real}}(k)\right] \\ =\ &1 - \Pr\left[M' \neq (0,0) \mid \text{in } \mathsf{Exp}_{\mathrm{FHKW},A}^{\text{NC-CCA-Real}}(k)\right] \\ =\ &1 - \Pr\left[M_1' = 1 \vee M_2' = 1 \mid \text{in } \mathsf{Exp}_{\mathrm{FHKW},A}^{\text{NC-CCA-Real}}(k)\right] \\ \geq\ &1 - 2\mathbf{Adv}_{\mathsf{XAC}}^{imp}(k). \end{aligned}$$

The Simulated Experiment. The ciphertext $C = (X_1, X_2, T)$ satisfies $X_1 \leftarrow \mathsf{SampleL}(\mathcal{L}_\Lambda; \widetilde{W_1})$, $X_2 \leftarrow \mathsf{SampleL}(\mathcal{L}_\Lambda; \widetilde{W_2})$, and $T = \mathsf{XAuth}(K_1, K_2)$, where for $i \in \{1, 2\}$, $\widetilde{W_i} \leftarrow \mathcal{R}_{\mathsf{SampleL}}$ and $K_i = \mathsf{PubEvl}(hpk, X_i, \widetilde{W_i}, t)$ with $t = \mathsf{H}(X_1, X_2)$.

The decryption of C' by the decryption oracle $\mathsf{Dec}(sk, \cdot)$ involves the computation of $t' = \mathsf{H}(X_1', X_2') = \mathsf{H}(X_1, X_2) = t$ and $\overline{K_i'} = \mathsf{SecEvl}(hsk, X_i', t') = \mathsf{SecEvl}(hsk, X_i, t)$, for $i \in \{1, 2\}$. On the other hand, we know that $K_2' = K_2$ and $K_2 = \mathsf{PubEvl}(hpk, X_2, W_2, t)$. Since $X_2 \in \mathcal{L}_\Lambda$, the property of EHPS guarantees that $\mathsf{SecEvl}(hsk, X_2, t) = \mathsf{PubEvl}(hpk, X_2, W_2, t)$, which means that $\overline{K_2'} = K_2 = K_2'$. Note that $M_2' = \mathsf{XVer}(\overline{K_2'}, 2, T')$. Hence, we have

$$\begin{aligned}
&\Pr\left[\ M_2' = 1 \mid \text{in } \mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)\right] \\
&= \Pr\left[\mathsf{XVer}(\overline{K_2'}, 2, T') = 1 \mid \text{in } \mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)\right] \\
&= \Pr\left[\mathsf{XVer}(K_2', 2, T') = 1 \mid \text{in } \mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)\right] \\
&\geq 1 - \mathsf{fail}^{correct}_{\mathsf{XAC}}(k).
\end{aligned}$$

The probability that A_2 outputs $b = 1$ in the simulated experiment is given by

$$\begin{aligned}
&\Pr\left[\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k) = 1\right] \\
&= \Pr\left[M' = (0,0) \mid \text{in } \mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)\right] \\
&= 1 - \Pr\left[M' \neq (0,0) \mid \text{in } \mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)\right] \\
&\leq 1 - \Pr\left[M_2' = 1 \mid \text{in } \mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)\right] \\
&\leq \mathsf{fail}^{correct}_{\mathsf{XAC}}(k).
\end{aligned}$$

The advantage of adversary A is given by

$$\begin{aligned}
\mathbf{Adv}^{\text{NC-CCA}}_{\text{FHKW},A,S}(k) &= \left|\Pr\left[\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k) = 1\right] - \Pr\left[\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k) = 1\right]\right| \\
&\geq 1 - 2\mathbf{Adv}^{imp}_{\mathsf{XAC}}(k) - \mathsf{fail}^{correct}_{\mathsf{XAC}}(k).
\end{aligned}$$

Note that both $\mathbf{Adv}^{imp}_{\mathsf{XAC}}(k)$ and $\mathsf{fail}^{correct}_{\mathsf{XAC}}(k)$ are negligible. So A's advantage $\mathbf{Adv}^{\text{NC-CCA}}_{\text{FHKW},A,S}(k)$ is non-negligible (in fact, it is overwhelming), i.e., the security proof of the FHKW scheme in [7] is incorrect. □

4.3 Security Analysis of the FHKW Scheme - $L = 1$

Note that our attack in the previous section does not apply to the case $L = 1$. In the previous section, upon receiving the ciphertext C and randomness R, the adversary A recovers K and switches the first element of K with a random one. If $L = 1$, A will get a new $K' = K_1'$ and then $T' = \mathsf{XAuth}(K_1')$. Afterwards, A

will return $C' = (X_1, T')$ as his decryption query. Then, A will receive $M' = 0$ with overwhelming probability in both $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$. Hence, the two experiments are still indistinguishable for A.

As we have pointed out earlier, the security of L-cross-authentication code against substitution attacks is not sufficient for the security proof of the FHKW scheme for any value of L. But our above attack only works for $L > 1$. Therefore, the remaining problem is whether it is possible for the FHKW scheme to achieve NC-CCA security for $L = 1$, still with the aforementioned simulator S.

Before solving the problem, we claim that algorithm XAuth of XAC in the FHKW scheme is deterministic (this is not explicitly expressed in [7]). That's because $R = (W_i, R_i^{\mathcal{X}_\Lambda}, R_i^{\mathcal{K}_\Lambda})_{i\in[L]}$ is the only randomness used in the encryption process. In other words, if XAuth is probabilistic, the inner random number used by XAuth should be contained in the randomness R (and then passed to the adversary, according to the definition of NC-CCA security). On the other hand, if algorithm XAuth of XAC in the FHKW scheme is probabilistic, with the aforementioned simulator S, the FHKW scheme *cannot* be proved secure in the sense of NC-CCA for any positive integer L. (See Appendix A for the proof.)

In fact, the security proof of the FHKW scheme expected such a property from L-cross-authentication code: "given $(K_1, K_2, \cdots, K_L)$ and $T = \mathsf{XAuth}(K_1, \cdots, K_L)$, it is difficult to output a $T' \neq T$ such that $\mathsf{XVer}(K_i, i, T') = 1$ for some $i \in [L]$". This property generally does not hold for L-cross-authentication code. However, it is true for some special 1-cross-authentication code, for example, the instance of L-cross-authentication code given by Fehr et al. [7] when constricted to $L = 1$. For that special instance, when $L = 1$, given $K = K_1$ and $T = \mathsf{XAuth}(K_1)$ (note that XAuth is deterministic), it is *impossible* to find a $T' \neq T$ such that $\mathsf{XVer}(K_1, 1, T') = 1$, since only $T = \mathsf{XAuth}(K_1)$ itself could pass the verification. Therefore, with the special 1-cross-authentication code instance (or other instance with similar property) as ingredient, the FHKW scheme is NC-CCA secure for $L = 1$.

5 Sender-Equivocable Encryption Scheme for Single Bit

In this section, we will refine the FHKW scheme for $L = 1$. Specifically, we will present a PKE scheme with NC-CCA security for $L = 1$ without any L-cross-authentication code.

Our scheme can be seen as a simplified version of the FHKW scheme instantiated with a special 1-cross-authentication code. As we pointed earlier, the special property of 1-cross-authentication code requires that each K determines a unique tag T satisfying $\mathsf{XVer}(K, T) = 1$. In our scheme, the encryption algorithm replaces the tag T by the key K directly. In the decryption, whether the plaintext is 1 or 0 depends on the equality of K in the ciphertext and $\overline{K}$ computed by $\mathsf{SecEvl}(hsk, X)$, while in the FHKW scheme the plaintext bit is determined by whether $\mathsf{XVer}(K, T') = 1$ or not.

Below describes our scheme $\mathcal{E} = (\mathsf{Gen}_\mathcal{E}, \mathsf{Enc}_\mathcal{E}, \mathsf{Dec}_\mathcal{E})$. The scheme consists of a hard subset membership problem SMP, with subset sparseness, and its corresponding perfectly 2-universal hash proof system HPS. We require that for

any $\Lambda \leftarrow \mathsf{SmpGen}(1^k)$, both $\mathcal{X}_\Lambda$ (with respect to SMP) and $\mathcal{K}_\Lambda$ (with respect to HPS) are efficiently explainable. As suggested in [7], the requirement of efficient samplability and explainability on $\mathcal{K}_\Lambda$ imposes no real restriction, and it has shown in [6] that both the above ingredients can be constructed based on some standard number-theoretic assumptions, such as the DDH, DCR and QR assumptions.

Scheme $\mathcal{E} = (\mathsf{Gen}_\mathcal{E}, \mathsf{Enc}_\mathcal{E}, \mathsf{Dec}_\mathcal{E})$

$\mathsf{Gen}_\mathcal{E}(1^k)$: On input 1^k, algorithm $\mathsf{Gen}_\mathcal{E}$ runs $\Lambda \leftarrow \mathsf{SmpGen}(1^k)$, $(hpk, hsk) \leftarrow \mathsf{HashGen}(\Lambda)$, and outputs (pk, sk), where $pk = hpk$ and $sk = hsk$.

$\mathsf{Enc}_\mathcal{E}(pk, M; R)$: To encrypt a plaintext $M \in \{0,1\}$ under a public key $pk = hpk$ with randomness $R = (W, R^{\mathcal{X}_\Lambda}, R^{\mathcal{K}_\Lambda}) \in \mathcal{R}_{\mathsf{SampleL}} \times \mathcal{R}_{\mathsf{Sample}} \times \mathcal{R}_{\mathsf{Sample}}$, algorithm $\mathsf{Enc}_\mathcal{E}$ sets

$$X := \begin{cases} \mathsf{Sample}(\mathcal{X}_\Lambda; R^{\mathcal{X}_\Lambda}) & \text{if } M = 0 \\ \mathsf{SampleL}(\mathcal{L}_\Lambda; W) & \text{if } M = 1 \end{cases}$$

and

$$K := \begin{cases} \mathsf{Sample}(\mathcal{K}_\Lambda; R^{\mathcal{K}_\Lambda}) & \text{if } M = 0 \\ \mathsf{PubEvl}(hpk, X, W) & \text{if } M = 1 \end{cases}$$

then returns ciphertext $C = (X, K)$.

$\mathsf{Dec}_\mathcal{E}(sk, C)$: To decrypt a ciphertext $C = (X, K) \in \mathcal{X}_\Lambda \times \mathcal{K}_\Lambda$ under a secret key $sk = hsk$, algorithm $\mathsf{Dec}_\mathcal{E}$ sets $\overline{K} := \mathsf{SecEvl}(hsk, X)$. If $\overline{K} = K$, return $M = 1$; else, return $M = 0$.

Correctness: On one hand, if $C = (X, K)$ is a ciphertext of $M = 1$, then $\overline{K} = \mathsf{SecEvl}(hsk, X) = \mathsf{PubEvl}(hpk, X, W) = K$ due to the property of HPS. So $\mathsf{Dec}_\mathcal{E}(sk, C)$ returns $M = 1$. On the other hand, if $C = (X, K)$ is a ciphertext of $M = 0$, then $X \leftarrow \mathcal{X}_\Lambda$, $K \leftarrow \mathcal{K}_\Lambda$ and $\overline{K} = \mathsf{SecEvl}(hsk, X)$. So $\Pr[\overline{K} = K] = \frac{1}{|\mathcal{K}_\Lambda|}$. Hence, with probability $1 - \frac{1}{|\mathcal{K}_\Lambda|}$, $\mathsf{Dec}_\mathcal{E}(sk, C)$ returns $M = 0$.

Security: As for the security of scheme $\mathcal{E}$, we have the following Theorem 2. The proof is similar to that of the FHKW scheme in [7]. But the key observation is: given $C = (X, K)$, it is impossible to create $C' = (X, K')$, $K \neq K'$, such that $K' = \overline{K'}$. Note that the security proof of our scheme doesn't involve any cross-authentication code. Details of the proof are in Appendix B.

Theorem 2. *Scheme* $\mathcal{E} = (\mathsf{Gen}_\mathcal{E}, \mathsf{Enc}_\mathcal{E}, \mathsf{Dec}_\mathcal{E})$ *is NC-CCA secure.*

6 Conclusion

We provided a security analysis of the FHKW scheme in [7] and showed that the original simulator constructed in [7] is not sufficient to prove NC-CCA security. However, some specific instances of 1-cross-authentication codes help the FHKW scheme to obtain NC-CCA security for encryption of single-bit plaintexts. We

provided a refined version of the FHKW scheme for single bit and proved its NC-CCA security. Our scheme does not involve any cross-authentication code, avoiding the security problem that annoys the FHKW scheme.

Open Questions. (1) The failure of the simulator proposed in [7] does not rule out the existence of other simulators working properly for the NC-CCA security proof of the FHKW scheme. Therefore, it is still open whether the FHKW scheme is NC-CCA secure or not. (2) Even if the FHKW scheme is not NC-CCA secure, it might still possess SIM-SO-CCA security. Hence, another question is whether it is SIM-SO-CCA secure or not. (3) Now that an NC-CCA secure PKE encrypting single bits is available in this paper, it may be interesting to construct an NC-CCA secure PKE encrypting multiple bits from an NC-CCA secure PKE encrypting single bits. This question in the relaxed setting of IND-CCA2 has been answered by Myers and Shelat [12]. But the selective opening scenario is much more complicated and we believe that the problem is much harder. (4) The last open question is how to construct a public-key encryption scheme that is NC-CCA secure for multi-bit plaintexts directly. We believe that with some extra property, the underlying cross-authentication code might be sufficient for the NC-CCA security proof of the FHKW scheme. We are working on this question. See [11] for details.

References

1. Bellare, M., Dowsley, R., Waters, B., Yilek, S.: Standard Security Does Not Imply Security against Selective-Opening. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 645–662. Springer, Heidelberg (2012)
2. Böhl, F., Hofheinz, D., Kraschewski, D.: On Definitions of Selective Opening Security. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 522–539. Springer, Heidelberg (2012)
3. Bellare, M., Hofheinz, D., Yilek, S.: Possibility and Impossibility Results for Encryption and Commitment Secure under Selective Opening. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 1–35. Springer, Heidelberg (2009)
4. Bellare, M., Waters, B., Yilek, S.: Identity-Based Encryption Secure against Selective Opening Attack. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 235–252. Springer, Heidelberg (2011)
5. Canetti, R., Friege, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: 28th ACM STOC, pp. 639–648. ACM Press, New York (1996)
6. Cramer, R., Shoup, V.: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
7. Fehr, S., Hofheinz, D., Kiltz, E., Wee, H.: Encryption Schemes Secure against Chosen-Ciphertext Selective Opening Attacks. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 381–402. Springer, Heidelberg (2010)
8. Gao, C.-Z., Xie, D., Wei, B.: Deniable Encryptions Secure against Adaptive Chosen Ciphertext Attack. In: Ryan, M.D., Smyth, B., Wang, G. (eds.) ISPEC 2012. LNCS, vol. 7232, pp. 46–62. Springer, Heidelberg (2012)
9. Hofheinz, D.: All-But-Many Lossy Trapdoor Functions. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 209–227. Springer, Heidelberg (2012)

10. Hemenway, B., Libert, B., Ostrovsky, R., Vergnaud, D.: Lossy Encryption: Constructions from General Assumptions and Efficient Selective Opening Chosen Ciphertext Security. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 70–88. Springer, Heidelberg (2011)
11. Huang, Z., Liu, S., Qin, B.: Sender equivocable encryption schemes secure against chosen-ciphertext attacks revisited. Cryptology ePrint Archive, Report 2012/473 (2012)
12. Myers, S., Shelat, A.: Bit encryption is complete. In: FOCS 2009, pp. 607–616. IEEE Computer Society Press (2009)
13. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: STOC 2008, pp. 187–196. ACM, New York (2008)

A In Case Algorithm XAuth Is Probabilistic

In Section 4.3, we have claimed that if algorithm XAuth of XAC in the FHKW scheme is probabilistic, with the aforementioned simulator S in Section 4, the FHKW scheme can not be proved NC-CCA secure for any positive integer L. Now we show the reason.

Firstly, a slight modification to XAuth is needed. Because XAuth is probabilistic, there exists an inner random number R^{XAuth} used by XAuth during the encryption process (i.e., $T \leftarrow \mathsf{XAuth}(K_1, \cdots, K_L; R^{\mathsf{XAuth}})$). Note that the aforementioned simulator S should output randomness $R = ((W_i, R_i^{\mathcal{X}_\Lambda}, R_i^{\mathcal{K}_\Lambda})_{i\in[L]}, R^{\mathsf{XAuth}})$ according to the ciphertext C and its related plaintext M. In the mean time, the original simulator S can recover $(W_i, R_i^{\mathcal{X}_\Lambda}, R_i^{\mathcal{K}_\Lambda})_{i\in[L]}$. Therefore, S should generate R^{XAuth} according to T and $(K_1, \cdots, K_L)$, which can be recovered from $R = (W_i, R_i^{\mathcal{X}_\Lambda}, R_i^{\mathcal{K}_\Lambda})_{i\in[L]}$. Now we make a modification to XAuth: we require that XAuth is efficiently "explainable", which means that there is an efficient algorithm $\mathsf{Explain}_{\mathsf{XAuth}}$ such that $R^{\mathsf{XAuth}} \leftarrow \mathsf{Explain}_{\mathsf{XAuth}}((K_1, \cdots, K_L), T)$. For simplicity, we still use the original notations S and XAuth after this modification.

Secondly, with the above modification, consider our main conclusion of this Appendix. As the proof of Theorem 1, our aim is to construct an adversary $A = (A_1, A_2)$ to distinguish the two experiments $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$. The adversary A is the same as the one in the proof of Theorem 1, except that in the decryption query stage, instead of choosing a random K_1', the adversary A uses the original K_1, which can be recovered from randomness $R = ((W_i, R_i^{\mathcal{X}_\Lambda}, R_i^{\mathcal{K}_\Lambda})_{i\in[L]}, R^{\mathsf{XAuth}})$. More specifically, in the first stage, A_1 returns $M = (0, \cdots, 0)$ to the challenger, and in the second stage, upon receiving the ciphertext $C = (X_1, \cdots, X_L, T)$ and randomness R, A_2 recovers $(K_1, \cdots, K_L)$ from R, computes $T' \leftarrow \mathsf{XAuth}(K_1, \cdots, K_L; \widetilde{R}^{\mathsf{XAuth}})$, where $\widetilde{R}^{\mathsf{XAuth}}$ is uniformly random chosen from $\mathcal{R}_{\mathsf{XAuth}}$, and returns $C' = (X_1, \cdots, X_L, T')$ as his decryption query. Because XAuth is probabilistic, it is very easy for A to get a $T' \neq T$ with the above method. As a result, with overwhelming probability, A_2 will receive $M' = (0, \cdots, 0)$ as the decryption result of C' in $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$, and receive $M' = (1, \cdots, 1)$ in $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$. Hence, A can distinguish $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\text{FHKW},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\text{FHKW},A}(k)$.

B Proof of Theorem 2

Proof. First, we construct a simulator $S_{\mathcal{E}}$ for scheme $\mathcal{E} = (\mathsf{Gen}_{\mathcal{E}}, \mathsf{Enc}_{\mathcal{E}}, \mathsf{Dec}_{\mathcal{E}})$.

Simulator $S_{\mathcal{E}}$:

- $S_{\mathcal{E}1}(pk, 1)$: With $pk = hpk$, choose $\widetilde{W} \leftarrow \mathcal{R}_{\mathsf{SampleL}}$ and set $X := \mathsf{SampleL}(\mathcal{L}_{\Lambda}; \widetilde{W})$. Then set $K := \mathsf{PubEvl}(hpk, X, \widetilde{W})$. Return the ciphertext $C = (X, K)$.
- $S_{\mathcal{E}2}(M)$: If $M = 1$, set $W := \widetilde{W}$ and choose $R^{\mathcal{X}_{\Lambda}} \leftarrow \mathcal{R}_{\mathsf{Sample}}$, $R^{\mathcal{K}_{\Lambda}} \leftarrow \mathcal{R}_{\mathsf{Sample}}$; otherwise choose $W \leftarrow \mathcal{R}_{\mathsf{SampleL}}$, and set $R^{\mathcal{X}_{\Lambda}} \leftarrow \mathsf{Explain}(\mathcal{X}_{\Lambda}, X)$, $R^{\mathcal{K}_{\Lambda}} \leftarrow \mathsf{Explain}(\mathcal{K}_{\Lambda}, K)$. Return the randomness $R = (W, R^{\mathcal{X}_{\Lambda}}, R^{\mathcal{K}_{\Lambda}})$.

With simulator $S_{\mathcal{E}}$, we will show that for any PPT adversary A, the two experiments $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\mathcal{E},A}(k)$ and $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\mathcal{E},A}(k)$ are computationally indistinguishable through a series of indistinguishable games. Technically, we denote the challenge ciphertext and its corresponding plaintext by C^* and M^*, and write $C^* := (X^*, K^*)$. Without loss of generality, we assume that A always makes q decryption queries, where $q = poly(k)$. For $j \in [q]$, denote A's j-th decryption query by $C^j := (X^j, K^j)$ and let its corresponding plaintext be M^j. At the same time, we define $\overline{K^*} := \mathsf{SecEvl}(hsk, X^*)$, $\overline{K^j} := \mathsf{SecEvl}(hsk, X^j)$ for $j \in [q]$, and denote the final output of A in Game i by $output_{A,i}$.

Game 0: Game 0 is the real experiment $\mathsf{Exp}^{\text{NC-CCA-Real}}_{\mathcal{E},A}(k)$. By our above notations,

$$\Pr\left[output_{A,0} = 1\right] = \Pr\left[\mathsf{Exp}^{\text{NC-CCA-Real}}_{\mathcal{E},A}(k) = 1\right].$$

Game 1: Game 1 is the same as Game 0, except for the decryption oracle. In Game 1, for any decryption query $C^j = (X^j, K^j)$ made by A, if $X^j \notin \mathcal{L}_{\Lambda}$, the challenger will return $M^j = 0$ directly, and if $X^j \in \mathcal{L}_{\Lambda}$, the challenger will answer the query as in Game 0: compute $\overline{K^j} = \mathsf{SecEvl}(hsk, X^j)$, and if $\overline{K^j} = K^j$, return $M^j = 1$, else return $M^j = 0$. Note that the decryption oracle in Game 1 is inefficient and it doesn't leak any information on hsk beyond hpk. Let bad_i denote the event that in Game i, A makes some decryption query $C^j = (X^j, K^j)$ such that $X^j \notin \mathcal{L}_{\Lambda}$ and $K^j = \overline{K^j}$. Note that $\Pr[\mathsf{bad}_1] = \Pr[\mathsf{bad}_0]$ and that Game 1 and Game 0 are identical unless events bad_1 or bad_0 occurs. By the perfect 2-universality of HPS and a union bound, $\Pr[\mathsf{bad}_1] = \Pr[\mathsf{bad}_0] \leq \frac{q}{|\mathcal{K}_{\Lambda}|}$. So we have

$$|\Pr\left[output_{A,1} = 1\right] - \Pr\left[output_{A,0} = 1\right]| \leq \Pr\left[\mathsf{bad}_1\right] = \frac{q}{|\mathcal{K}_{\Lambda}|}.$$

Game 2: Game 2 is the same as Game 1, except that in the challenge ciphertext generation, set $K^* = \mathsf{SecEvl}(hsk, X^*)$ for $M^* = 0$ and then the randomness of K^* is opened as $\mathsf{Explain}(\mathcal{K}_{\Lambda}, K^*)$. In Game 1 if $M^* = 0$, K^* also can be seen as being opened by $\mathsf{Explain}(\mathcal{K}_{\Lambda}, K^*)$. In Game 2, since the only information on hsk beyond hpk is released in the computation of K^*, the perfect 2-universality of HPS implies that if $X^* \notin \mathcal{L}_{\Lambda}$, K^* is uniformly

distributed in $\mathcal{K}_\Lambda$. Let sub_i denote the event that in Game i when $M^* = 0$, $X^* \in \mathcal{L}_\Lambda$. Note that $\Pr[\mathsf{sub}_2] = \Pr[\mathsf{sub}_1]$ and that Game 2 and Game 1 are the same unless events sub_2 or sub_1 occurs. So we have

$$|\Pr\left[output_{A,2} = 1\right] - \Pr\left[output_{A,1} = 1\right]| \leq \Pr\left[\mathsf{sub}_2\right] = \frac{|\mathcal{L}_\Lambda|}{|\mathcal{X}_\Lambda|}.$$

Game 3: Game 3 is the same as Game 2, except that the decryption oracle works with the original decryption rule. In Game 3, for any decryption query $C^j = (X^j, K^j)$, the challenger sets $\overline{K^j} = \mathsf{SecEvl}(hsk, X^j)$, then returns $M^j = 1$ if $\overline{K^j} = K^j$, or returns $M^j = 0$ if $\overline{K^j} \neq K^j$. Note that the decryption oracle in Game 3 is efficient. Similarly, bad_i denotes the event that in Game i, A makes some decryption query $C^j = (X^j, K^j)$ such that $X^j \notin \mathcal{L}_\Lambda$ and $K^j = \overline{K^j}$. Note that $\Pr[\mathsf{bad}_3] = \Pr[\mathsf{bad}_2]$ and that Game 3 and Game 2 are identical unless events bad_3 or bad_2 occurs. Since the only information on hsk beyond hpk is released in the computation of K^*, by the perfect 2-universality of HPS and a union bound, $\Pr[\mathsf{bad}_3] = \Pr[\mathsf{bad}_2] = \frac{q}{|\mathcal{K}_\Lambda|}$. So

$$|\Pr\left[output_{A,3} = 1\right] - \Pr\left[output_{A,2} = 1\right]| \leq \Pr\left[\mathsf{bad}_3\right] = \frac{q}{|\mathcal{K}_\Lambda|}.$$

Game 4: Game 4 is the same as Game 3, except that in the challenge ciphertext generation, the challenger chooses $X^* \leftarrow \mathcal{L}_\Lambda$ if $M^* = 0$. That is to say, choose $X^* \leftarrow \mathcal{L}_\Lambda$ no matter whether M^* is 0 or 1, and X^* is opened as $\mathsf{Explain}(\mathcal{X}_\Lambda, X^*)$ if $M^* = 0$. Since SMP is hard,

$$|\Pr\left[output_{A,4} = 1\right] - \Pr\left[output_{A,3} = 1\right]| \leq \mathbf{Adv}_{\mathsf{SMP},A}(k).$$

Combining all the above results, we have

$$|\Pr\left[output_{A,0} = 1\right] - \Pr\left[output_{A,4} = 1\right]| \leq \frac{2q}{|\mathcal{K}_\Lambda|} + \frac{|\mathcal{L}_\Lambda|}{|\mathcal{X}_\Lambda|} + \mathbf{Adv}_{\mathsf{SMP},A}(k).$$

Note that Game 4 is just the experiment $\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\mathcal{E},A}(k)$. So we have

$$\begin{aligned}\mathbf{Adv}^{\text{NC-CCA}}_{\mathcal{E},A,S}(k) &= |\Pr\left[\mathsf{Exp}^{\text{NC-CCA-Real}}_{\mathcal{E},A}(k) = 1\right] - \Pr\left[\mathsf{Exp}^{\text{NC-CCA-Sim}}_{\mathcal{E},A}(k) = 1\right]| \\ &\leq \tfrac{2q}{|\mathcal{K}_\Lambda|} + \tfrac{|\mathcal{L}_\Lambda|}{|\mathcal{X}_\Lambda|} + \mathbf{Adv}_{\mathsf{SMP},A}(k).\end{aligned}$$

□

Efficient Completely Context-Hiding Quotable and Linearly Homomorphic Signatures

Nuttapong Attrapadung[1,⋆], Benoît Libert[2,⋆⋆], and Thomas Peters[3,⋆⋆⋆]

[1] Research Center for Information Security, AIST, Japan
[2] Technicolor, France
[3] Université Catholique de Louvain, ICTEAM Institute, Belgium

Abstract. Homomorphic signatures are primitives that allow for public computations for a class of specified predicates over authenticated data. An enhanced privacy notion, called complete context-hiding security, was recently motivated by Attrapadung *et al.* (Asiacrypt'12). This notion ensures that a signature derived from any valid signatures is perfectly indistinguishable from a newly generated signatures (on the same message), and seems desirable in many applications requiring to compute on authenticated data. In this paper, we focus on two useful predicates – namely, substring quotation predicates and linear dependency predicates – and present the first completely context-hiding schemes for these in the standard model. Moreover, our new quotable signature scheme is the first such construction with signatures of linear size. In comparison with the initial scheme of Ahn *et al.* (TCC 2012), we thus reduce the signature size from $O(n \log n)$ to $O(n)$, where n is the message size. Our scheme also allows signing messages of arbitrary length using constant-size public keys.

Keywords: Homomorphic signatures, provable security, privacy, unlinkability, standard model.

1 Introduction

The recent years, much attention has been paid to homomorphic cryptographic primitives, which make it possible to publicly compute over encrypted [24,34] or signed [30,10,12] datasets.

In the latter case, anyone holding signatures $\{\sigma_i = \mathsf{Sign}(\mathsf{sk}, m_i)\}_{i=1}^k$ on messages $\{m_i\}_{i=1}^k$ can publicly derive pairs $(m, \sigma) = \mathsf{Evaluate}(\mathsf{pk}, \{(m_i, \sigma_i)\}_{i=1}^k, f)$ such that $\mathsf{Verify}(\mathsf{pk}, m, \sigma) = 1$, where $m = f(m_1, \ldots, m_k)$ for certain functions f. This has been possible for arithmetic functions [10,22,11,12], logical predicates [33,26,14,15,13] and other kinds of algebraic signatures [32,8,27,28]. In the case of arithmetic manipulations, homomorphic signatures notably allow untrusted

⋆ This author is supported by KAKENHI (Grant-in-Aid for Young Scientists B) No. 22700020. This work was done while the author visited ENS Paris.

⋆⋆ Part of this work was done while this author was a F.R.S.-F.N.R.S. scientific collaborator at the Université catholique de Louvain (Belgium).

⋆⋆⋆ Supported by the IUAP B-Crypt Project and the Walloon Region Camus Project.

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 386–404, 2013.

remote parties (e.g. storage servers in cloud computing services) to authenticate their calculations on the clients' data. They also proved useful to prevent pollution attacks in network coding [10,3,22].

At TCC 2012, Ahn *et al.* [4] defined the general notion of P-homomorphic signature – for a predicate P – that captures all the aforementioned forms of homomorphic signatures. Specifically, it allows anybody who sees a signature on a message m to publicly obtain signatures on messages m' such that $P(m, m') = 1$. Informally, a P-homomorphic signature is said unforgeable when a signature on m only makes it possible to publicly derive signatures on messages m' such that $P(m, m') = 1$. Ahn *et al.* also formalized a strong privacy property, called *strong context hiding*, which mandates that original and derived signatures be unconditionally unlinkable.

Quite recently, Attrapadung, Libert and Peters [6] suggested even stronger privacy notions, of which the strongest one is termed *complete context-hiding* security. The difference between the definition of Ahn *et al.* [4] and the one of [6] lies in that the former requires the unlinkability of derived signatures to only *honestly generated* signatures. In contrast, the stronger *complete* context hiding property [6] requires unlinkability with respect to *any valid* signatures, including those signatures that might have been somehow maliciously re-randomized by the adversary. Not achieving this kind of security may raise some concerns in certain applications such as collusion attacks in network coding, as motivated in [6].

So far, in the standard model, complete context-hiding security has been achieved for only one specific kind of predicates, namely subset predicates [6]. For other predicates, completely context-hiding constructions are currently lacking. In particular, this is true for substring quotations – which were addressed by the main construction of [4] – and linear homomorphisms, that have been extensively studied in recent years [10,22,11,5,16,17,20]. This paper aims at filling these gaps by proposing the first completely context-hiding schemes for these predicates. Along the way, we also improve upon the best previously achieved efficiency for quoting predicates.

1.1 Related Work

Homomorphic signatures were first suggested by Desmedt [19] and further studied by Johnson, Molnar, Song and Wagner [30]. Later on, they were considered by Boneh, Freeman, Katz and Waters [10] who used them to sign linear subspaces so as to thwart pollution attacks in network coding. In the random oracle model, Boneh *et al.* [10] described a pairing-based scheme with short per-vector signatures. In a follow-up work, Gennaro, Katz, Krawczyk and Rabin [22] gave an RSA-based linearly homomorphic system [22] over the integers in the random oracle model. Boneh and Freeman [11] suggested to work over binary fields using lattices. They also motivated a notion, termed *weak privacy*, which requires derived signatures not to leak the original dataset they were derived from.

Constructions in the standard model came out in two independent papers by Attrapadung and Libert [5] and Catalano, Fiore and Warinschi [16,17]. The

construction of [5] was extended by Freeman [20] who defined a framework for the design of linearly homomorphic signatures satisfying a stronger definition of unforgeability. The latter framework of [20] was notably instantiated under standard assumptions like RSA, Diffie-Hellman and, more efficiently, the Strong Diffie-Hellman assumption. In the random oracle model, Boneh and Freeman [12] designed lattice-based homomorphic signatures for multivariate polynomial functions. Except [10,5], all the aforementioned constructions are only weakly context-hiding in the sense of [11].

Strongly context-hiding P-homomorphic signatures were recently given by Ahn *et al.* [4] for both quoting and subset predicates. In [4], linearly homomorphic signatures [10,11,16,20] were also shown to imply P-homomorphic signatures allowing for the computation of weighted averages and Fourier transforms. It was pinpointed in [4] that the Boneh *et al.* [10] system is strongly context-hiding thanks to the uniqueness of its signatures (in the random oracle model).

In the standard model, the construction of Attrapadung and Libert [5] can be proved strongly context hiding as well (unlike the schemes of [16,17,20]) but, as discussed in [6], it is demonstrably not completely context-hiding. Attrapadung *et al.* [6] came close to filling this gap by describing a more efficient strongly context-hiding realization simultaneously satisfying another privacy notion which had been elusive so far. Still, their use of the dual system technique [36,23] prevented them from reaching the desired complete context-hiding level. In the standard model, no completely context-hiding linearly homomorphic signature has ever been reported to date.

1.2 Our Contributions

Linear-Size Homomorphic Signatures for Quoting Substring. Given a signature on a message m, quotable signatures allow for the public derivation of signatures on any substring of m. Ahn *et al.* [4] gave a system where signatures have quasi-linear size: for a message consisting of n symbols, each signature contains $O(n \log n)$ group elements[1]. Their construction is known to be only strongly context-hiding (in the sense of [4]) and selectively unforgeable in the random oracle model. It was argued that their scheme can be modified so as to be proved fully unforgeable in the standard model using the dual system encryption technique of Waters [36] (or, more precisely, its signature analogue [23]). The latter inherently involves two distinct distributions of signatures satisfying the verification algorithm. The very existence of an alternative distribution of valid signatures implies that the resulting system can hardly be completely context-hiding.

The first contribution of this paper is a quotable signature scheme whose design principle is very different from [4]. The new scheme is proved fully unforgeable in the standard model and also turns out to be the first completely

[1] In the signature derivation algorithm of [4], two kinds of signatures can be produced. Apart from Type I signatures, which are distributed as original signatures, Type II signatures have $O(\log n)$-size signatures but cannot be quoted any further.

context-hiding quotable signature. Moreover, it improves upon the worst-case efficiency of [4] in that a n-symbol message can be signed using $O(n)$ group elements.

Our construction builds on the structure-preserving signature of Abe, Haralambiev and Ohkubo [1], which is used to sign individual message symbols. An important property of the structure-preserving signature in [1] is that certain signature components can serve as a commitment to the message. Our quotable signature exploits this property to link signatures on individual symbols: each symbol is signed with a commitment to the next symbol. Quotable signatures are then obtained as a sequence of perfectly hiding commitments to these underlying signatures and non-interactive randomizable arguments of their validity.

Beyond its asymptotically shorter signatures, our scheme also allows signing messages of arbitrary length using a constant-size public key. In contrast, [4] requires the key generation algorithm to define a logarithmic bound on the maximal number of symbols in messages to be signed.

Completely Context-Hiding Linearly Homomorphic Signatures. We provide the first completely context-hiding linearly homomorphic signature in the standard model. So far, the random-oracle-based construction of Boneh *et al.* [10] was the only linearly homomorphic signatures satisfying that level of privacy. The scheme of [5] is strongly context-hiding in the standard model but, as pointed out in [6], it falls short of the enhanced privacy level advocated by [6].

To bypass the latter limitation – which seems inherent to all signature schemes [5,23] based on the dual system technique – we take further advantage of the malleability properties [7,21] of Groth-Sahai proofs [25] and build on a linearly homomorphic signature proposed by Attrapadung *et al.* [6]. The latter scheme is only weakly context-hiding (*i.e.*, the original message remains hidden as long as the original signature is not given) as its signatures contain components that cannot be randomized at each derivation and thus carry information about the original signatures. Our idea is to replace these signature components by perfectly hiding commitments to these values. The commitments are accompanied with non-interactive (randomizable) witness indistinguishable arguments that committed values satisfy appropriate algebraic relations.

One difficulty to solve is that, in the underlying weakly context-hiding construction [6], the "problematic" signature components are actually exponents that the reduction has to compute in the security proof. When Groth-Sahai proofs are used in their extractable mode, committed exponents cannot be fully extracted from their commitments. To solve this problem, we need to modify the weakly context-hiding scheme of [6] in such a way that its signatures only consist of group elements. We were able to do this at the expense of relying on a slightly stronger assumption in the security proof: instead of the standard Diffie-Hellman assumption, the unforgeability now relies on the Flexible Diffie-Hellman assumption [29], which is still a simple assumption.

2 Background

2.1 Definitions for Homomorphic Signatures

Definition 1 ([4]). *Let $\mathcal{M}$ be a message space and $2^{\mathcal{M}}$ be its powerset. Let $P : 2^{\mathcal{M}} \times \mathcal{M} \to \{0,1\}$ be a predicate. A message m' is said* **derivable** *from $M \subset \mathcal{M}$ if $P(M,m') = 1$. As in [4], $P^i(M)$ is the set of messages derivable from $P^{i-1}(M)$, where $P^0(M) := \{m' \in \mathcal{M} \mid P(M,m') = 1\}$. Finally, $P^*(M) := \cup_{i=0}^{\infty} P^i(M)$ denotes the set of messages derivable from M by iterated derivation.*

Definition 2 ([4]). *A P-homomorphic signature for a predicate $P : 2^{\mathcal{M}} \times \mathcal{M} \to \{0,1\}$ is a triple of algorithms* $(\mathsf{Keygen}, \mathsf{SignDerive}, \mathsf{Verify})$ *with the following properties.*

$\mathsf{Keygen}(\lambda)$: *takes as input a security parameter $\lambda \in \mathbb{N}$ and outputs a key pair $(\mathsf{sk}, \mathsf{pk})$. As in [4], the private key sk is seen as a signature on the empty tuple $\varepsilon \in \mathcal{M}$.*

$\mathsf{SignDerive}\big(\mathsf{pk}, (\{\sigma_m\}_{m\in M}, M), m'\big)$: *is a possibly randomized algorithm that takes as input a public key pk, a set of messages $M \subset \mathcal{M}$, a corresponding set of signatures $\{\sigma_m\}_{m\in M}$ and a derived message $m' \in \mathcal{M}$. If $P(M,m') = 0$, it returns $\perp$. Otherwise, it outputs a derived signature σ'*

$\mathsf{Verify}(\mathsf{pk}, \sigma, m)$: *is a deterministic algorithm that takes as input a public key pk, a signature σ and a message m. It outputs 0 or 1.*

Note that the empty tuple $\varepsilon \in \mathcal{M}$ satisfies $P(\varepsilon, m) = 1$ for each message $m \in \mathcal{M}$. Similarly to [4], we define the algorithm $\mathsf{Sign}(\mathsf{pk}, \mathsf{sk}, m)$ that runs[2] $\mathsf{SignDerive}(\mathsf{pk}, (\mathsf{sk}, \varepsilon), m)$ and returns the output. For any $M = \{m_1, \ldots, m_k\} \subset \mathcal{M}$, we define $\mathsf{Sign}(\mathsf{sk}, M) := \{\mathsf{Sign}(\mathsf{sk}, m_1), \ldots, \mathsf{Sign}(\mathsf{sk}, m_k)\}$. Also, we write $\mathsf{Verify}(\mathsf{pk}, M, \{\sigma_m\}_{m\in M}) = 1$ to express that $\mathsf{Verify}(\mathsf{pk}, m, \sigma_m) = 1$ for each $m \in M$.

Correctness. It is required that, for all key pairs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\lambda)$, for any message set $M \subset \mathcal{M}$, any message $m' \in \mathcal{M}$ such that $P(M,m') = 1$, the following conditions must be satisfied: (i) $\mathsf{SignDerive}(\mathsf{pk}, (\mathsf{Sign}(\mathsf{sk}, M), M), m') \neq \perp$; (ii) $\mathsf{Verify}\big(\mathsf{pk}, m', \mathsf{SignDerive}(\mathsf{pk}, (\mathsf{Sign}(\mathsf{sk}, M), M), m')\big) = 1$.

Definition 3 ([4]). *A P-homomorphic signature* $(\mathsf{Keygen}, \mathsf{SignDerive}, \mathsf{Verify})$ *is said* **unforgeable** *if no probabilistic polynomial-time (PPT) adversary has non-negligible advantage in this game:*

1. *The challenger generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\lambda)$ and gives pk to the adversary $\mathcal{A}$. It initializes two initially empty tables T and Q.*
2. *$\mathcal{A}$ adaptively interleaves the following queries.*
   - *Signing queries: $\mathcal{A}$ chooses a message $m \in \mathcal{M}$. The challenger replies by choosing a handle h, runs $\sigma \leftarrow \mathsf{Sign}(sk, m)$ and stores (h, m, σ) in a table T. The handle h is returned to $\mathcal{A}$.*

[2] The intuition is that any message can be derived when the original message contains the signing key.

- *Derivation queries:* $\mathcal{A}$ *chooses a vector of handles* $\vec{\mathsf{h}} = (\mathsf{h}_1, \ldots, \mathsf{h}_k)$ *and a message* $m' \in \mathcal{M}$. *The challenger retrieves the tuples* $\{(\mathsf{h}_i, m_i, \sigma_i)\}_{i=1}^{k}$ *from* T *and returns* $\perp$ *if one of these does not exist. Otherwise, it defines* $M := (m_1, \ldots, m_k)$ *and* $\{\sigma_m\}_{m \in M} = \{\sigma_1, \ldots, \sigma_k\}$. *If* $P(M, m') = 1$, *the challenger runs* $\sigma' \leftarrow \mathsf{SignDerive}\big(\mathsf{pk}, (\{\sigma_m\}_{m \in M}, M), m'\big)$, *chooses a handle* h', *stores* $(\mathsf{h}', m', \sigma')$ *in* T *and returns* h' *to* $\mathcal{A}$.
- *Reveal queries:* $\mathcal{A}$ *chooses a handle* h. *If no tuple of the form* $(\mathsf{h}, m', \sigma')$ *exists in* T, *the challenger returns* $\perp$. *Otherwise, it returns* σ' *to* $\mathcal{A}$ *and adds* (m', σ') *to the set* Q.

3. $\mathcal{A}$ *outputs a pair* (σ', m') *and wins if: (i)* $\mathsf{Verify}(\mathsf{pk}, m', \sigma') = 1$; *(ii) If* $M \subset \mathcal{M}$ *is the set of messages in* Q, *then* $m' \notin P^*(M)$.

Ahn *et al.* [4] formalized a strong notion of privacy that captures the inability of distinguishing derived signatures from original ones, *even* when these are given along with the private key. In [4], it was shown that, if a scheme is strongly context hiding, then Definition 3 can be simplified by only providing the adversary with an ordinary signing oracle.

As noted in [6], specific applications may require an even stronger definition. In particular, the following definition makes sense when homomorphic signature schemes are randomizable and/or the verification algorithm accepts several distributions of valid-looking signatures.

Definition 4 ([6]). *A homomorphic signature* $(\mathsf{Keygen}, \mathsf{Sign}, \mathsf{SignDerive}, \mathsf{Verify})$ *is* **completely context hiding** *for the predicate* P *if, for all key pairs* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\lambda)$, *for all message sets* $M \subset \mathcal{M}^*$ *and* $m' \in \mathcal{M}$ *such that* $P(M, m') = 1$, *for all* $\{\sigma_m\}_{m \in M}$ *such that* $\mathsf{Verify}(\mathsf{pk}, M, \{\sigma_m\}_{m \in M}) = 1$, *the following distributions are statistically close*

$$\{(\mathsf{sk},\ \{\sigma_m\}_{m \in M},\ \mathsf{Sign}(sk, m'))\}_{sk, M, m'}\ ,$$
$$\big\{\big(\mathsf{sk},\ \{\sigma_m\}_{m \in M},\ \mathsf{SignDerive}\big(\mathsf{pk}, (\{\sigma_m\}_{m \in M}, M), m'\big)\big)\big\}_{sk, M, m'}\ .$$

2.2 Hardness Assumptions

We consider bilinear maps $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ over groups of prime order p. In these groups, we rely on the following hardness assumptions.

Definition 5 ([9]). *The* **Decision Linear Problem** *(DLIN) in* $\mathbb{G}$ *consists in distinguishing the distributions* $(g^a, g^b, g^{ac}, g^{bd}, g^{c+d})$ *and* $(g^a, g^b, g^{ac}, g^{bd}, g^z)$, *with* $a, b, c, d \xleftarrow{R} \mathbb{Z}_p^*$, $z \xleftarrow{R} \mathbb{Z}_p^*$. *The* **Decision Linear Assumption** *is the intractability of DLIN for any PPT distinguisher* $\mathcal{D}$.

We also use a weaker variant of an assumption used in [29,31]. The latter is a variant of the Diffie-Hellman assumption, which posits the infeasibility of finding a pair $(g^\mu, g^{ab \cdot \mu})$ given $(g, g^a, g^b) \in \mathbb{G}^3$.

Definition 6. *The* **Flexible Diffie-Hellman Problem** *(FlexDH) in* $\mathbb{G}$, *is given* (g, g^a, g^b), *where* $a, b \xleftarrow{R} \mathbb{Z}_p$, *to find a triple* $(g^\mu, g^{a \cdot \mu}, g^{ab \cdot \mu}) \in \mathbb{G}^3$ *such that* $\mu \neq 0$.

The FlexDH assumption is known to imply the intractability of distinguishing g^{abc} from random given (g, g^a, g^b, g^c). For this reason, it can be seen as a *simple* assumption.

Finally, we also use the following q-type assumption.

Definition 7 ([1]). *In a group $\mathbb{G}$, the* **q-Simultaneous Flexible Pairing Problem** *(q-SFP) is, given $\left(g_z, h_z, g_r, h_r, a, \tilde{a}, b, \tilde{b}\right) \in \mathbb{G}^8$ as well as a set of q tuples $(z_j, r_j, s_j, t_j, u_j, v_j, w_j) \in \mathbb{G}^7$ such that*

$$\begin{aligned} e(a, \tilde{a}) &= e(g_z, z_j) \cdot e(g_r, r_j) \cdot e(s_j, t_j), \\ e(b, \tilde{b}) &= e(h_z, z_j) \cdot e(h_r, u_j) \cdot e(v_j, w_j), \end{aligned} \tag{1}$$

to find a new tuple $(z^\star, r^\star, s^\star, t^\star, u^\star, v^\star, w^\star) \in \mathbb{G}^7$ satisfying (1) and such that $z^\star \notin \{1_\mathbb{G}, z_1, \ldots, z_q\}$.

2.3 Structure-Preserving Signatures

Many protocols require to sign elements of bilinear groups while preserving their structure and, in particular, without hashing them. Abe, Haralambiev and Ohkubo [1,2] (AHO) described such a signature. The description below assumes common public parameters $\mathsf{pp} = \left((\mathbb{G}, \mathbb{G}_T),\ g\right)$ consisting of symmetric bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$, where $\lambda \in \mathbb{N}$ and a generator $g \in \mathbb{G}$.

Keygen(pp, n)**:** given an upper bound $n \in \mathbb{N}$ on the number of group elements per message to be signed, choose generators $G_r, H_r \xleftarrow{R} \mathbb{G}$. Pick $\gamma_z, \delta_z \xleftarrow{R} \mathbb{Z}_p$ and $\gamma_i, \delta_i \xleftarrow{R} \mathbb{Z}_p$, for $i = 1$ to n. Then, compute $G_z = G_r^{\gamma_z}$, $H_z = H_r^{\delta_z}$ and $G_i = G_r^{\gamma_i}$, $H_i = H_r^{\delta_i}$ for each $i \in \{1, \ldots, n\}$. Finally, choose $\alpha_a, \alpha_b \xleftarrow{R} \mathbb{Z}_p$ and define $A = e(G_r, g^{\alpha_a})$ and $B = e(H_r, g^{\alpha_b})$. The public key is defined to be

$$pk = \left(G_r,\ H_r,\ G_z,\ H_z,\ \{G_i, H_i\}_{i=1}^n,\ A,\ B\right) \in \mathbb{G}^{2n+4} \times \mathbb{G}_T^2$$

while the private key is $sk = \left(\alpha_a, \alpha_b, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^n\right)$.

Sign$(sk, (M_1, \ldots, M_n))$**:** to sign a vector $(M_1, \ldots, M_n) \in \mathbb{G}^n$ using the private key $sk = (\alpha_a, \alpha_b, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^n)$, choose $\zeta, \rho_a, \rho_b, \omega_a, \omega_b \xleftarrow{R} \mathbb{Z}_p$ and compute $\theta_1 = g^\zeta$ as well as

$$\theta_2 = g^{\rho_a - \gamma_z \zeta} \cdot \prod_{i=1}^n M_i^{-\gamma_i}, \qquad \theta_3 = G_r^{\omega_a}, \qquad \theta_4 = g^{(\alpha_a - \rho_a)/\omega_a},$$

$$\theta_5 = g^{\rho_b - \delta_z \zeta} \cdot \prod_{i=1}^n M_i^{-\delta_i}, \qquad \theta_6 = H_r^{\omega_b}, \qquad \theta_7 = g^{(\alpha_b - \rho_b)/\omega_b},$$

The signature consists of $\sigma = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7) \in \mathbb{G}^7$.

Verify$(pk, \sigma, (M_1, \ldots, M_n))$**:** given a signature $\sigma = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7) \in \mathbb{G}^7$, return 1 if and only if these values satisfy the equalities

$$A = e(G_z, \theta_1) \cdot e(G_r, \theta_2) \cdot e(\theta_3, \theta_4) \cdot \prod_{i=1}^{n} e(G_i, M_i)$$

$$B = e(H_z, \theta_1) \cdot e(H_r, \theta_5) \cdot e(\theta_6, \theta_7) \cdot \prod_{i=1}^{n} e(H_i, M_i).$$

The scheme is known [1,2] to be existentially unforgeable under chosen-message attacks under the q-SFP assumption, where q is the number of signing queries.

As pointed out in [1,2], signature components $\{\theta_i\}_{i=2}^{7}$ can be publicly re-randomized so as to obtain a different signature $\{\theta'_i\}_{i=1}^{7} \leftarrow \mathsf{ReRand}(pk, \sigma)$ on $(M_1, \ldots, M_n)$. After each randomization, we have $\theta'_1 = \theta_1$ whereas $\{\theta'_i\}_{i=2}^{7}$ are uniformly distributed among the set of group elements $(\theta_2, \ldots, \theta_7)$ for which the equalities $e(G_r, \theta'_2) \cdot e(\theta'_3, \theta'_4) = e(G_r, \theta_2) \cdot e(\theta_3, \theta_4)$ and $e(H_r, \theta'_5) \cdot e(\theta'_6, \theta'_7) = e(H_r, \theta_5) \cdot e(\theta_6, \theta_7)$ hold. As a result, $\{\theta'_i\}_{i \in \{3,6\}}$ are statistically independent of the message and other signature components.

It was also observed [1,2] that signature components (θ_3, θ_6) can be used as a commitment to the message. Under the q-SFP assumption, it is infeasible to find signatures $\sigma = (\theta_1, \ldots, \theta_7)$, $\sigma' = (\theta'_1, \ldots, \theta'_7)$ on two distinct messages M, M' such that $(\theta_3, \theta_6) = (\theta'_3, \theta'_6)$. This is true even if the adversary has access to a signing oracle and obtains signatures on both M and M'.

2.4 Groth-Sahai Proof Systems

In [25], Groth and Sahai described efficient non-interactive witness indistinguishable (NIWI) proof systems that can be based on the DLIN assumption. In this case, they use prime order groups and a common reference string containing three vectors $\vec{f_1}, \vec{f_2}, \vec{f_3} \in \mathbb{G}^3$, where $\vec{f_1} = (f_1, 1, g)$, $\vec{f_2} = (1, f_2, g)$ for some $f_1, f_2 \in \mathbb{G}$. To commit to a group element $X \in \mathbb{G}$, the prover chooses $r, s, t \xleftarrow{R} \mathbb{Z}_p^*$ and computes $\vec{C} = (1, 1, X) \cdot \vec{f_1}^{\,r} \cdot \vec{f_2}^{\,s} \cdot \vec{f_3}^{\,t}$. On a perfectly sound common reference string, we have $\vec{f_3} = \vec{f_1}^{\,\xi_1} \cdot \vec{f_2}^{\,\xi_2}$ where $\xi_1, \xi_2 \in \mathbb{Z}_p^*$. Commitments $\vec{C} = (f_1^{r+\xi_1 t}, f_2^{s+\xi_2 t}, X \cdot g^{r+s+t(\xi_1+\xi_2)})$ are extractable commitments whose distribution is that of Boneh-Boyen-Shacham (BBS) ciphertexts [9]: committed values can be extracted using $\beta_1 = \log_g(f_1)$, $\beta_2 = \log_g(f_2)$. In the witness indistinguishability (WI) setting, vectors $\vec{f_3}$ is chosen outside the span of $(\vec{f_1}, \vec{f_2})$, so that $\vec{C}$ is a perfectly hiding commitment. Under the DLIN assumption, the two kinds of CRS are computationally indistinguishable.

To provide evidence that committed variables satisfy a set of relations, the prover computes one commitment per variable and one proof element per relation. Such efficient NIWI proofs are available for pairing-product equations, which are relations of the type.

$$\prod_{i=1}^{n} e(\mathcal{A}_i, \mathcal{X}_i) \cdot \prod_{i=1}^{n} \cdot \prod_{j=1}^{n} e(\mathcal{X}_i, \mathcal{X}_j)^{a_{ij}} = t_T, \tag{2}$$

for variables $\mathcal{X}_1, \ldots, \mathcal{X}_n \in \mathbb{G}$ and constants $t_T \in \mathbb{G}_T$, $\mathcal{A}_1, \ldots, \mathcal{A}_n \in \mathbb{G}$, $a_{ij} \in \mathbb{Z}_p$, for $i, j \in \{1, \ldots, n\}$.

In pairing-product equations, proofs for quadratic equations require 9 group elements whereas linear equations (*i.e.*, where $a_{ij} = 0$ for all i, j in equation (2)) only cost 3 group elements each.

Belenkiy *et al.* [7] showed that Groth-Sahai proofs are perfectly randomizable. Given commitments $\{\vec{C}_{\mathcal{X}_i}\}_{i=1}^n$ and a NIWI proof $\vec{\pi}_{\text{PPE}}$ that committed variables $\{\mathcal{X}\}_{i=1}^n$ satisfy (2), anyone can publicly (*i.e.*, without knowing the witnesses) compute re-randomized commitments $\{\vec{C}_{\mathcal{X}'_i}\}_{i=1}^n$ and a re-randomized proof $\vec{\pi}'_{\text{PPE}}$ of the same statement. Moreover, $\{\vec{C}_{\mathcal{X}'_i}\}_{i=1}^n$ and $\vec{\pi}'_{\text{PPE}}$ are distributed as freshly generated commitments and proof. This property was notably used in [21,18].

3 Linear-Size Quotable Signatures

In quotable signatures, given a signature on some message, one should only be able to derive signatures on arbitrary substrings of the original message. The message space $\mathcal{M}$ is also defined as the set of strings $\mathcal{M} := \Sigma^*$, where Σ is a set of symbols. The predicate P is univariate (*i.e.*, $|M| = 1$) and defined to have $P\big(\{\mathsf{Msg}_1\}, \mathsf{Msg}_2\big) = 1$ whenever Msg_2 is a substring of Msg_1.

The scheme bears resemblance with the homomorphic signature for subset predicates of [6] which also builds on structure-preserving signatures. In fact, the construction is itself a structure-preserving quotable signature as it allows signing sequences of group elements.

We actually use a variant of the unbounded AHO signature scheme which allows signing messages of arbitrary length with a public key of fixed size. In [1], this is achieved by taking advantage of the property called "signature binding" (and proved in [1, Lemma 3]), which informally says that signature components (θ_3, θ_6) can be used as a commitment to the message: namely, given only the public key and access to a signing oracle, unless the scheme is existentially forgeable under chosen-message attacks, it is infeasible to come up with two *distinct* messages $(M_1, \ldots, M_n), (M'_1, \ldots, M'_n)$ with corresponding valid signatures $\sigma = (\theta_1, \ldots, \theta_7)$ and $\sigma' = (\theta'_1, \ldots, \theta'_7)$ such that $\theta_3 = \theta'_3$ and $\theta_6 = \theta'_6$. This remains true *even* if $(M_1, \ldots, M_n)$ and $(M'_1, \ldots, M'_n)$ are both submitted to the signing oracle during the game. Using this observation, a basic signature scheme where the message space is $\mathbb{G}^3$ can be turned into an "unbounded" structure-preserving signature, where the signer can sign messages of arbitrary length. The idea is to use signature components $\{(\theta_{i,3}, \theta_{i,6})\}_{i=1}^n$ to link adjacent message blocks together: each block $m_i \in \mathbb{G}$ is signed along with the $(\theta_{i-1,3}, \theta_{i-1,6})$ components of the signature on the previous block $m_{i-1} \in \mathbb{G}$. In our scheme, we proceed in the same way but, unlike [1], we do not encode the total number of

blocks within the message. This modification allows anyone to quote signatures by removing portions of the chain in its extremities. In order to prevent illegal combinations of two different chains, the signer processes the last block m_n of each message $(m_1, \ldots, m_n)$ by signing it with a pair of random group elements $(\tilde{\theta}_3, \tilde{\theta}_6)$ which are part of the private key. This allows us to prove security using the same arguments as in [1].

For the sake of privacy, the components of $\{\sigma_i\}_{i=1}^n$ are not explicitly given out but only appear within perfectly hiding Groth-Sahai commitments accompanied with appropriate NIWI arguments. At each signature derivation, commitments and NIWI arguments are suitably re-randomized.

An important difference with the construction for subset predicates in [6], is that underlying AHO signatures entirely appear in committed form. The reason is that using $(\theta_{i,3}, \theta_{i,6})$ in the chaining process prevents their re-randomization. For this reason, they also have to be committed so that we need to work with quadratic pairing-product equations.

In the following, when $X \in \mathbb{G}$ (resp. $X \in \mathbb{G}_T$), the notation $\iota(X)$ (resp. $\iota_{\mathbb{G}_T}(X)$) will be used to denote the vector $(1, 1, X) \in \mathbb{G}^3$ (resp. the 3×3 matrix containing X in position $(3,3)$ and $1_{\mathbb{G}_T}$ everywhere else). Finally, we also use a symmetric bilinear map $F : \mathbb{G}^3 \times \mathbb{G}^3 \to \mathbb{G}_T^9$ such that, for any two vectors $\vec{X} = (X_1, X_2, X_3) \in \mathbb{G}^3$ and $\vec{Y} = (Y_1, Y_2, Y_3) \in \mathbb{G}^3$, $F(\vec{X}, \vec{Y})$ is defined to be $F(\vec{X}, \vec{Y}) = \tilde{F}(\vec{X}, \vec{Y})^{1/2} \cdot \tilde{F}(\vec{Y}, \vec{X})^{1/2}$, where the non-commutative mapping $\tilde{F} : \mathbb{G}^3 \times \mathbb{G}^3 \to \mathbb{G}_T^9$ sends $(\vec{X}, \vec{Y})$ onto the matrix $\tilde{F}(\vec{X}, \vec{Y})$ of entry-wise pairings (*i.e.*, containing $e(X_i, Y_j)$ in its entry (i, j)).

Keygen(λ)**:** given a security parameter $\lambda \in \mathbb{N}$, choose bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$.

1. Choose a Groth-Sahai CRS $\mathbf{f} = (\vec{f}_1, \vec{f}_2, \vec{f}_3)$ for the perfect WI setting. More precisely, choose $\vec{f}_1 = (f_1, 1, g)$, $\vec{f}_2 = (1, f_2, g)$, and $\vec{f}_3 = \vec{f}_1^{\,\xi_1} \cdot \vec{f}_2^{\,\xi_2} \cdot (1, 1, g)^{-1}$, with $f_1, f_2, g \xleftarrow{R} \mathbb{G}$, $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p$.
2. Generate a key pair $(sk_{\mathsf{aho}}, pk_{\mathsf{aho}})$ for the AHO signature in order to sign messages consisting of three group elements. This key pair consists of $sk_{\mathsf{aho}} = \big(\alpha_a, \alpha_b, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^3\big)$ and

$$pk_{\mathsf{aho}} = \Big(G_r,\ H_r,\ G_z = G_r^{\gamma_z},\ H_z = H_r^{\delta_z},\ \{G_i = G_r^{\gamma_i},\ H_i = H_r^{\delta_i}\}_{i=1}^3,\ A,\ B\Big).$$

3. Choose two uniformly random group elements $\tilde{\theta}_3, \tilde{\theta}_6 \xleftarrow{R} \mathbb{G}$.

The public key consists of $\mathsf{pk} := \Big((\mathbb{G}, \mathbb{G}_T),\ \mathbf{f},\ pk_{\mathsf{aho}}\Big)$ whereas the private key is defined to be $\mathsf{sk} = \big(sk_{\mathsf{aho}},\ (\tilde{\theta}_3, \tilde{\theta}_6)\big)$. The public key defines the set of symbols $\Sigma = \mathbb{G}$.

Sign$(\mathsf{sk}, \mathsf{Msg})$**:** given $\mathsf{sk} = \big(sk_{\mathsf{aho}},\ (\tilde{\theta}_3, \tilde{\theta}_6)\big)$ and a length-n message $\mathsf{Msg} = (m_1, \ldots, m_n) \in \mathbb{G}^n$, for some $n \in \mathsf{poly}(\lambda)$ and where $m_i \in \mathbb{G}$ for each $i \in \{1, \ldots, n\}$, do the following.

1. Define $(\theta_{n+1,3}, \theta_{n+1,6}) = (\tilde{\theta}_3, \tilde{\theta}_6)$. Then, for $k \in \{3,6\}$, compute Groth-Sahai commitments $\vec{C}_{\theta_{n+1,k}} = \iota(\theta_{n+1,k}) \cdot \vec{f}_1^{\,r_{\theta_{n+1,k}}} \cdot \vec{f}_2^{\,s_{\theta_{n+1,k}}} \cdot \vec{f}_3^{\,t_{\theta_{n+1,k}}}$.
2. For each $j = n$ down to 1, generate an AHO signature $(\theta_{j,1}, \ldots, \theta_{j,7})$ on the message $(m_j, \theta_{j+1,3}, \theta_{j+1,6}) \in \mathbb{G}^3$. For each $k \in \{1, \ldots, 7\}$ and $j \in \{1, \ldots, n\}$, generate commitments

$$\vec{C}_{\theta_{j,k}} = \iota(\theta_{j,k}) \cdot \vec{f}_1^{\,r_{\theta_{j,k}}} \cdot \vec{f}_2^{\,s_{\theta_{j,k}}} \cdot \vec{f}_3^{\,t_{\theta_{j,k}}}.$$

   Next, generate NIWI aruments $\vec{\pi}_{\mathsf{aho},j,1}, \vec{\pi}_{\mathsf{aho},j,2} \in \mathbb{G}^9$ that committed variables $(\theta_{j,1}, \theta_{j,2}, \theta_{j,3}, \theta_{j,4}, \theta_{j,5}, \theta_{j,6}, \theta_{j,7})$ satisfy

$$\begin{aligned} A \cdot e(G_1, m_j)^{-1} &= e(G_z, \theta_{j,1}) \cdot e(G_r, \theta_{j,2}) \cdot e(\theta_{j,3}, \theta_{j,4}) \\ &\quad \cdot e(G_2, \theta_{j+1,3}) \cdot e(G_3, \theta_{j+1,6}) \\ B \cdot e(H_1, m_j)^{-1} &= e(H_z, \theta_{j,1}) \cdot e(H_r, \theta_{j,5}) \cdot e(\theta_{j,6}, \theta_{j,7}) \\ &\quad \cdot e(H_2, \theta_{j+1,3}) \cdot e(H_3, \theta_{j+1,6}) \end{aligned} \tag{3}$$

   These equations are quadratic, so that $\{\vec{\pi}_{\mathsf{aho},j,1}, \vec{\pi}_{\mathsf{aho},j,2}\}_{j=1}^{n}$ consist of 9 group elements each.
3. Return the signature

$$\sigma = \Big(\{\vec{C}_{\theta_{n+1,k}}\}_{k\in\{3,6\}}, \big\{\{\vec{C}_{\theta_{j,k}}\}_{k=1}^{7}, \vec{\pi}_{\mathsf{aho},j,1}, \vec{\pi}_{\mathsf{aho},j,2}\big\}_{j=1}^{n}\Big). \tag{4}$$

SignDerive$(\mathsf{pk}, \mathsf{Msg}, \mathsf{Msg}', \sigma)$**:** given the public key pk as well as two messages $\mathsf{Msg} = (m_1, \ldots, m_n) \in \mathbb{G}^n$ and $\mathsf{Msg}' = (m'_1, \ldots, m'_{n'}) \in \mathbb{G}^{n'}$, return $\perp$ if Msg' is not a substring of Msg. Otherwise, there exists $i \in \{1, \ldots, n-n'+1\}$ such that $\mathsf{Msg}' = (m'_1, \ldots, m'_{n'}) = (m_i, \ldots, m_{i+n'-1})$. Then, parse σ as in (4) and, for each $i \in \{1, \ldots, n'\}$, conduct the following steps.

1. Define the sub-signature

$$\tilde{\sigma} = \big(\{\vec{C}_{\theta_{i+n',k}}\}_{k\in\{3,6\}}, \big\{\{\vec{C}_{\theta_{i+j,k}}\}_{k=1}^{7}, \vec{\pi}_{\mathsf{aho},i+j,1}, \vec{\pi}_{\mathsf{aho},i+j,2}\big\}_{j=0}^{n'-1}\big).$$

2. Re-randomize $\vec{C}'_{\theta_{i+j,k}} = \vec{C}_{\theta_{i+j,k}} \cdot \vec{f}_1^{\,r'_{\theta_{i+j,k}}} \cdot \vec{f}_2^{\,s'_{\theta_{i+j,k}}} \cdot \vec{f}_3^{\,t'_{\theta_{i+j,k}}}$ for $j = 0$ to $n'-1$ and $k = 1$ to 7. Likewise, compute re-randomized versions $\{\vec{C}'_{\theta_{i+n',k}}\}_{k\in\{3,6\}}$ of $\{\vec{C}_{\theta_{i+n',k}}\}_{k\in\{3,6\}}$. Finally, re-randomize the proofs

$$\{\vec{\pi}_{\mathsf{aho},i+j,1} = (\vec{\pi}_{i+j,1}, \vec{\pi}_{i+j,2}, \vec{\pi}_{i+j,3})\}_{j=0}^{n'-1}$$

   and

$$\{\vec{\pi}_{\mathsf{aho},i+j,2} = (\vec{\pi}_{i+j,4}, \vec{\pi}_{i+j,5}, \vec{\pi}_{i+j,6})\}_{j=0}^{n'-1}$$

   as suggested in [7].
3. Return the signature

$$\sigma' = \Big(\{\vec{C}'_{\theta_{i+n',k}}\}_{k\in\{3,6\}}, \big\{\{\vec{C}'_{\theta_{i+j,k}}\}_{k=1}^{7}, \vec{\pi}'_{\mathsf{aho},i+j,1}, \vec{\pi}'_{\mathsf{aho},i+j,2}\big\}_{j=0}^{n'-1}\Big). \tag{5}$$

Verify$(\mathsf{pk}, \mathsf{Msg}, \sigma)$**:** given pk, a signature σ and a message $\mathsf{Msg} = (m_1, \ldots, m_n) \in \mathbb{G}^n$, parse σ as per (4) and do the following. For $j = 1$ to n, return 0 if $\vec{\pi}_{\mathsf{aho},j,1} = (\vec{\pi}_{j,1}, \vec{\pi}_{j,2}, \vec{\pi}_{j,3})$ and $\vec{\pi}_{\mathsf{aho},j,2} = (\vec{\pi}_{j,4}, \vec{\pi}_{j,5}, \vec{\pi}_{j,6})$ do not satisfy the equations below. Otherwise, return 1.

$$\iota_{\mathbb{G}_T}(A) \,/\, F\big(\iota(G_1), \iota(m_j)\big) = F\big(\iota(G_z), \vec{C}_{\theta_{j,1}}\big) \cdot F\big(\iota(G_r), \vec{C}_{\theta_{j,2}}\big) \cdot F\big(\vec{C}_{\theta_{j,3}}, \vec{C}_{\theta_{j,4}}\big) \cdot F\big(\iota(G_2), \vec{C}_{\theta_{j+1,3}}\big) \cdot F\big(\iota(G_3), \vec{C}_{\theta_{j+1,6}}\big) \cdot \prod_{k=1}^{3} F\big(\vec{\pi}_{j,k}, \vec{f}_k\big) \tag{6}$$

$$\iota_{\mathbb{G}_T}(B) \,/\, F\big(\iota(H_1), \iota(m_j)\big) = F\big(\iota(H_z), \vec{C}_{\theta_{j,1}}\big) \cdot F\big(\iota(H_r), \vec{C}_{\theta_{j,5}}\big) \cdot F\big(\vec{C}_{\theta_{j,6}}, \vec{C}_{\theta_{j,7}}\big) \cdot F\big(\iota(H_2), \vec{C}_{\theta_{j+1,3}}\big) \cdot F\big(\iota(H_3), \vec{C}_{\theta_{j+1,6}}\big) \cdot \prod_{k=1}^{3} F\big(\vec{\pi}_{j,k+1}, \vec{f}_k\big).$$

Unlike the scheme of [4], the above system allows signing arbitrarily long messages with a public key of constant size whereas [4] requires to set a logarithmic bound on the length of signed messages at key generation. The signature length is asymptotically optimal: a n-symbol message can be signed using $39n+6$ group elements.

On the other hand, we lose a useful feature of the construction in [4]. The latter allows the derivation algorithm to produce two kinds of derived signatures: when the message m' consists of ℓ symbols, Type I signatures contain $O(\ell \log \ell)$ group elements and support subsequent quoting. Alternatively, the quoting algorithm can derive a much shorter Type II signature, which comprises $O(\log \ell)$ elements, but cannot be quoted any further. In our scheme, the quoter can only produce Type I signatures and does not have the same flexibility as in [4].

We now turn to the security of the scheme and first observe that it is clearly completely context-hiding due to the use of a witness indistinguishable Groth-Sahai CRS.

Theorem 1. *The above quotable signature scheme is completely context hiding.*

Proof. The proof follows from the fact that each signature only consists of perfectly hiding commitments and perfectly NIWI arguments, which can be perfectly re-randomized at each derivation. □

The unforgeability relies on the DLIN assumption and the security properties of AHO signatures, as established by Theorem 2.

Theorem 2. *The scheme is existentially unforgeable against chosen-message attacks under the $(q \cdot L + 1)$-SFP and DLIN assumptions, where q denotes the maximal number of signing queries and L is the maximal number of symbols per signing query.*

Proof. Since the scheme is completely context hiding, we only need to prove unforgeability using the simpler definition where the adversary $\mathcal{A}$ only has a signing oracle. The proof uses a sequence of games where, for each i, S_i stands for the event that $\mathcal{A}$ produces a valid forgery in Game_i.

Game$_0$: This is the real game. We denote by S_0 the event that the adversary $\mathcal{A}$ manages to output a successful forgery. Obviously, $\mathcal{A}$'s advantage is $\Pr[S_0]$.

Game$_1$: We change the generation of the public key and set up $\mathbf{f} = (\vec{f_1}, \vec{f_2}, \vec{f_3})$ as a perfectly sound Groth-Sahai CRS. Concretely, the challenger $\mathcal{B}$ chooses $\vec{f_3}$ in the span of $\vec{f_1} = (f_1, 1, g)$ and $\vec{f_2} = (1, f_2, g)$, where $f_1 = g^{\phi_1}$ and $f_2 = g^{\phi_2}$, for random chosen $\phi_1, \phi_2 \xleftarrow{R} \mathbb{Z}_p$. Signing queries are answered as in Game$_0$, using the private key $(sk_{\mathsf{aho}}, (\tilde{\theta}_3, \tilde{\theta}_6))$ and generating NIWI arguments faithfully. Under the DLIN assumption, this change should not significantly affect $\mathcal{A}$'s behavior and we have $|\Pr[S_1] - \Pr[S_0]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\mathcal{B})$. Note that the reduction is immediate as $\mathcal{B}$ does not need the trapdoor (ϕ_1, ϕ_2) at any time. In Game$_1$, perfectly hiding Groth-Sahai commitments (and NIWI arguments) are traded for perfectly binding commitments (and perfectly sound proofs).

Game$_2$: This game is identical to Game 1 except that we bring a conceptual change in the generation of sk. Instead of merely choosing $(\tilde{\theta}_3, \tilde{\theta}_6)$ at random, the challenger $\mathcal{B}$ picks a uniformly random group element $\tilde{m} \xleftarrow{R} \mathbb{G}$ and computes an AHO signature $\{\tilde{\theta}_k\}_{k=1}^{7}$ on the "dummy" message $(\tilde{m}, 1, 1)$. The resulting $(\tilde{\theta}_3, \tilde{\theta}_6)$ are included in the private key sk whereas $\tilde{m}$ and $\{\tilde{\theta}_k\}_{k\in\{1,2,4,5,7\}}$ are retained by $\mathcal{B}$. We argue that this change does not alter $\mathcal{A}$'s view whatsoever since $(\tilde{\theta}_3, \tilde{\theta}_6)$ have the same distribution either way. Indeed, in Game$_2$, they remain uniformly distributed in $\mathbb{G}^2$ and statistically independent of the message $\tilde{m}$ and other signature components. We have $\Pr[S_2] = \Pr[S_1]$.

In Game$_2$, $\mathcal{B}$ uses the values $(\phi_1, \phi_2) = (\log_g(f_1), \log_g(f_2))$ that were defined in Game$_1$. When $\mathcal{A}$ outputs a forgery $\sigma^\star$ on a message $(m_1^\star, \ldots, m_{n^\star}^\star)$, $\mathcal{B}$ uses (ϕ_1, ϕ_2) to extract $(\theta_{n^\star+1,3}^\star, \theta_{n^\star+1,6}^\star)$ as well as a sequence of AHO signatures $\{\sigma_j^\star = (\theta_{j,1}^\star, \ldots, \theta_{j,7}^\star)\}_{j=1}^{n^\star}$ from the Groth-Sahai commitments contained in $\sigma^\star$. The perfect soundness of $\{\vec{\pi}_{\mathsf{aho},j,1}^\star, \vec{\pi}_{\mathsf{aho},j,2}^\star\}_{j=1}^{n^\star}$ guarantees that extracted values $(m_1^\star, \ldots, m_{n^\star}^\star)$, $\{\sigma_j^\star\}_{j=1}^{n^\star}$ and $(\theta_{n^\star+1,3}^\star, \theta_{n^\star+1,6}^\star)$ satisfy equations (3).

In Game$_2$, we can prove that event S_2 occurs with negligible probability if the $(q \cdot L + 1)$-SFP assumption holds. Indeed, if $\mathcal{A}$ is successful in Game$_3$, $\{\sigma_j^\star = (\theta_{j,1}^\star, \ldots, \theta_{j,7}^\star)\}_{j=1}^{n^\star}$ is a sequence of valid AHO signatures on the messages $\{(m_j^\star, \theta_{j+1,3}^\star, \theta_{j+1,6}^\star)\}_{j=1}^{n^\star}$ but $(m_1^\star, \ldots, m_{n^\star}^\star)$ is not a subsequence involved in any of the signing queries. We can thus distinguish two situations.

Case A. There exists $j^\dagger \in \{1, \ldots, n\}$ such that $\mathcal{B}$ never had to sign the message $(m_{j^\dagger}^\star, \theta_{j^\dagger+1,3}^\star, \theta_{j^\dagger+1,6}^\star)$ in any signing query.

Case B. The messages $\{(m_j^\star, \theta_{j+1,3}^\star, \theta_{j+1,6}^\star)\}_{j=1}^{n^\star}$ were all signed by $\mathcal{B}$ at some point of the game but not all of them were involved in the same query. This covers the case of an adversary mixing substrings of two different messages for which it received signatures.

In Case A, it is immediate that $\mathcal{A}$ necessarily broke the chosen-message security of the AHO signature: the reduction $\mathcal{B}$ simply outputs $(m_{j^\dagger}^\star, \theta_{j^\dagger+1,3}^\star, \theta_{j^\dagger+1,6}^\star)$ and the signature $\sigma_{j^\dagger}^\star$.

We are thus left with Case B for which we know that $(m_1^\star, \theta_{2,3}^\star, \theta_{2,6}^\star)$ was involved in the κ-th signing query $\mathsf{Msg}_\kappa = (m_{\kappa,1}, \ldots, m_{\kappa,n_\kappa})$, for some integers $\kappa \in \{1, \ldots, q\}$ and $n_\kappa \in \{1, \ldots, L\}$. Let $\{(\theta_{\kappa,j,1}, \ldots, \theta_{\kappa,j,7})\}_{j=1}^{n_\kappa}$ be the AHO signatures that were used to answer the κ-th signing query. Let also $t \in \{1, \ldots, n_\kappa\}$ be such that $(m_{\kappa,t}, \theta_{\kappa,t+1,3}, \theta_{\kappa,t+1,6}) = (m_1^\star, \theta_{2,3}^\star, \theta_{2,6}^\star)$.

We now define $j^\star$ to be the largest index in $\{1, \ldots, n^\star - 1\}$ such that

$$(m_{\kappa,t+j^\star-1}, \theta_{\kappa,t+j^\star,3}, \theta_{\kappa,t+j^\star,6}) = (m_{j^\star}^\star, \theta_{j^\star+1,3}^\star, \theta_{j^\star+1,6}^\star).$$

At this step, we further consider two sub-cases of Case B:

Case $t + j^\star < n_\kappa + 1$**:** Since $m_{\kappa,t+j^\star} \neq m_{j^\star+1}$ or $(\theta_{\kappa,t+j^\star+1,3}, \theta_{\kappa,t+j^\star+1,6}) \neq (\theta_{j^\star+2,3}^\star, \theta_{j^\star+2,6}^\star)$, the signature binding property of the AHO signature is broken since we have two distinct messages whose signatures share the same $\theta_{j^\star+1,3}^\star, \theta_{j^\star+1,6}^\star$ components. As implied by the results of [1], this contradicts the $(q \cdot L + 1)$-SFP assumption since $\mathcal{B}$ computes at most $q \cdot L + 1$ AHO signatures.

Case $t + j^\star = n_\kappa + 1$**:** We have the equality

$$(\theta_{j^\star+1,3}^\star, \theta_{j^\star+1,6}^\star) = (\theta_{\kappa,n_{\kappa+1},3}, \theta_{\kappa,n_{\kappa+1},6}) = (\tilde{\theta}_3, \tilde{\theta}_6),$$

which means that $(m_{j^\star}^\star, \theta_{j^\star+1,3}^\star, \theta_{j^\star+1,6}^\star)$ was the message of an "end-of-chain" signature produced by $\mathcal{B}$. Said otherwise, this is a forgery where $(m_1^\star, \ldots, m_{n^\star}^\star)$ is a super-string of $(m_{\kappa,t}, \ldots, m_{\kappa,n_k})$. In this case, thanks to the modification introduced in Game$_2$, $\mathcal{B}$ knows $\{\tilde{\theta}_k\}_{k\in\{1,2,4,5,7\}}$ as well as a dummy message $\tilde{m}$ such that $(\tilde{\theta}_1, \ldots, \tilde{\theta}_7)$ is a valid AHO signature on $(\tilde{m}, 1, 1)$. With overwhelming probability, we obtain distinct messages $(\tilde{m}, 1, 1)$ and $(m_{j^\star+1}^\star, \theta_{j^\star+2,3}^\star, \theta_{j^\star+2,6}^\star)$ that share the same signature components $(\theta_{j^\star+1,3}^\star, \theta_{j^\star+1,6}^\star) = (\tilde{\theta}_3, \tilde{\theta}_6)$. Indeed, the pair $(\tilde{\theta}_3, \tilde{\theta}_6)$ is statistically independent of the dummy message $\tilde{m}$ and the latter was uniformly chosen in $\mathbb{G}$. It comes that we can only have $m_{j^\star+1}^\star = \tilde{m}$ by pure chance.

In Case B, the signature binding property of AHO signatures is thus broken either way and we can eventually write $\Pr[S_2] \leq 2 \cdot \mathbf{Adv}^{(q\cdot L+1)\text{-SFP}}(\mathcal{B})$, where the factor 2 accounts for the fact that the reduction has to guess beforehand which of Case A or Case B will come about. Depending on this guess, $\mathcal{B}$ undertakes to either attack the standard unforgeability of AHO signatures or, alternatively, break their signature-binding property. In either case, $\mathcal{B}$ answers $\mathcal{A}$'s queries by invoking the signing oracle in its interaction with the appropriate challenger.

Putting the above altogether, we find the upper bound

$$\Pr[S_0] \leq \mathbf{Adv}^{\text{DLIN}}(\mathcal{B}) + 2 \cdot \mathbf{Adv}^{(q\cdot L+1)\text{-SFP}}(\mathcal{B})$$

on the forger's advantage. □

4 Completely Context-Hiding Linearly Homomorphic Signatures

We now turn to linearly homomorphic signatures for which the syntax and the security definitions of Section 2 can be simplified as explained in [6].

Our starting point is the weakly context-hiding linearly homomorphic signature of [6]. Its public key includes group elements g^α, v and $\{g_i\}_{i=1}^n$, where n is the dimension of vectors to be signed. Signatures of vectors $\vec{v} = (v_1, \ldots, v_n)$ are of the form $(\sigma_1, \sigma_2, s) = \big((\prod_{i=1}^n g_i^{v_i} \cdot v^s)^\alpha \cdot H_{\mathbb{G}}(\tau)^r, g^r, s\big)$, where $r, s \in_R \mathbb{Z}_p$ and τ identifies the linear subspace.

The reason why the scheme is only weakly context-hiding is that the signature component s cannot be re-randomized. Hence, it always allows linking a derived signature to those it was obtained from. To render the scheme completely context-hiding, we need to modify the signing algorithm so as to hide $s \in \mathbb{Z}_p$. In signatures, the exponent s is replaced by Groth-Sahai commitments to group elements $(g^s, g^{\alpha \cdot s})$, where g^α is the public key, together with NIWI arguments that these are correctly formed. Then, the randomizability properties of Groth-Sahai proofs come in handy to guarantee that derived signatures will be statistically independent of original signatures.

In the notations hereunder, for any $h \in \mathbb{G}$ and $\vec{g} = (g_1, g_2, g_3) \in \mathbb{G}^3$, $E(h, \vec{g})$ stands for the vector $\big(e(h, g_1), e(h, g_2), e(h, g_3)\big) \in \mathbb{G}_T^3$.

Keygen(λ, n)**:** given a security parameter $\lambda \in \mathbb{N}$ and an integer $n \in \mathsf{poly}(\lambda)$, choose bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$.

1. Choose $\alpha \xleftarrow{R} \mathbb{Z}_p$, $g, v \xleftarrow{R} \mathbb{G}$ and $u_0, u_1, \ldots, u_L \xleftarrow{R} \mathbb{G}$, for some $L \in \mathsf{poly}(\lambda)$. Elements $(u_0, \ldots, u_L) \in \mathbb{G}^{L+1}$ will define hash function $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ mapping any L-bit string $m = m[1] \ldots m[L] \in \{0,1\}^L$ onto a hash value $H_{\mathbb{G}}(m) = u_0 \cdot \prod_{i=1}^L u_i^{m[i]}$.
2. Pick $g_i \xleftarrow{R} \mathbb{G}$ for $i = 1$ to n. Also, define the identifier space $\mathcal{T} := \{0,1\}^L$.
3. Generate Groth-Sahai common reference string $\mathbf{f} = (\vec{f_1}, \vec{f_2}, \vec{f_3})$ for the perfect WI setting. Namely, choose vectors $\vec{f_1} = (f_1, 1, g)$, $\vec{f_2} = (1, f_2, g)$, as well as $\vec{f_3} = \vec{f_1}^{\xi_1} \cdot \vec{f_2}^{\xi_2} \cdot (1, 1, g)^{-1}$, with $f_1, f_2 \xleftarrow{R} \mathbb{G}$, $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p$.

The private key is $\mathsf{sk} := \alpha$ and the public key consists of

$$\mathsf{pk} := \Big((\mathbb{G}, \mathbb{G}_T),\ g,\ g^\alpha,\ v,\ \{g_i\}_{i=1}^n,\ \{u_i\}_{i=0}^L,\ \mathbf{f}\Big).$$

Sign$(\mathsf{sk}, \tau, \vec{v})$**:** given a vector $\vec{v} = (v_1, \ldots, v_n) \in \mathbb{Z}_p^n$, a file identifier $\tau \in \{0,1\}^L$ and the private key $\mathsf{sk} = \alpha \in \mathbb{Z}_p$, do the following.

1. Choose $r, s \xleftarrow{R} \mathbb{Z}_p$ and compute

$$\sigma_1 = (g_1^{v_1} \cdots g_n^{v_n} \cdot v^s)^\alpha \cdot H_{\mathbb{G}}(\tau)^r, \qquad \sigma_2 = g^r, \qquad \sigma_3 = g^s, \qquad \sigma_4 = g^{\alpha \cdot s}.$$

2. Compute commitments to $(\sigma_1, \sigma_3, \sigma_4)$. Namely, for each $j \in \{1, 3, 4\}$, choose $r_{\sigma_j}, s_{\sigma_j}, t_{\sigma_j} \xleftarrow{R} \mathbb{Z}_p$ and compute $\vec{C}_{\sigma_j} = (1, 1, \sigma_j) \cdot \vec{f_1}^{r_{\sigma_j}} \cdot \vec{f_2}^{s_{\sigma_j}} \cdot \vec{f_3}^{t_{\sigma_j}}$.
3. Generate a NIWI proof that $(\sigma_1, \sigma_3, \sigma_4) \in \mathbb{G}^3$ satisfy the linear equations

$$e(\sigma_1, g) = e(\prod_{i=1}^n g_i^{v_i}, g^\alpha) \cdot e(v, \sigma_4) \cdot e(H_{\mathbb{G}}(\tau), \sigma_2), \tag{7}$$

$$e(\sigma_3, g^\alpha) = e(g, \sigma_4). \tag{8}$$

These proofs are obtained as

$$\vec{\pi}_1 = (\pi_{1,1}, \pi_{1,2}, \pi_{1,3}) = \big(g^{r_{\sigma_1}} \cdot v^{-r_{\sigma_4}},\ g^{s_{\sigma_1}} \cdot v^{-s_{\sigma_4}},\ g^{t_{\sigma_1}} \cdot v^{-t_{\sigma_4}}\big)$$
$$\vec{\pi}_2 = (\pi_{2,1}, \pi_{2,2}, \pi_{2,3}) = \big((g^{\alpha})^{r_{\sigma_3}} \cdot g^{-r_{\sigma_4}},\ (g^{\alpha})^{s_{\sigma_3}} \cdot g^{-s_{\sigma_4}},\ (g^{\alpha})^{t_{\sigma_3}} \cdot g^{-t_{\sigma_4}}\big),$$

which satisfy the equations

$$E(g, \vec{C}_{\sigma_1}) = E\big(\prod_{i=1}^{n} g_i^{v_i}, (1,1,g^{\alpha})\big) \cdot E(v, \vec{C}_{\sigma_4}) \cdot E\big(H_{\mathbb{G}}(\tau), (1,1,\sigma_2)\big) \cdot \prod_{j=1}^{3} E(\pi_{1,j}, \vec{f}_j) \tag{9}$$

$$E(g^{\alpha}, \vec{C}_{\sigma_3}) = E(g, \vec{C}_{\sigma_4}) \cdot \prod_{j=1}^{3} E(\pi_{2,j}, \vec{f}_j). \tag{10}$$

The signature consists of $\sigma = \big(\vec{C}_{\sigma_1}, \sigma_2, \vec{C}_{\sigma_3}, \vec{C}_{\sigma_4}, \vec{\pi}_1, \vec{\pi}_2\big) \in \mathbb{G}^{16}$.

SignDerive$(\mathsf{pk}, \tau, \{(\beta_i, \sigma^{(i)})\}_{i=1}^{\ell})$**:** given pk, a file identifier τ and ℓ tuples $(\beta_i, \sigma^{(i)})$, parse each signature $\sigma^{(i)}$ as $\sigma^{(i)} = \big(\vec{C}_{\sigma_{i,1}}, \sigma_{i,2}, \vec{C}_{\sigma_{i,3}}, \vec{C}_{\sigma_{i,4}}, \vec{\pi}_{i,1}, \vec{\pi}_{i,2}\big) \in \mathbb{G}^{16}$.

1. Choose $\tilde{r} \xleftarrow{R} \mathbb{Z}_p$. Then, compute $\sigma_2 = \prod_{i=1}^{\ell} \sigma_{i,2}^{\beta_i} \cdot g^{\tilde{r}}$ and

$$\vec{C}_{\sigma_1} = \prod_{i=1}^{\ell} \vec{C}_{\sigma_{i,1}}^{\beta_i} \cdot (1,1,H_{\mathbb{G}}(\tau)^{\tilde{r}}) \qquad \vec{C}_{\sigma_3} = \prod_{i=1}^{\ell} \vec{C}_{\sigma_{i,3}}^{\beta_i} \qquad \vec{C}_{\sigma_4} = \prod_{i=1}^{\ell} \vec{C}_{\sigma_{i,4}}^{\beta_i}$$

   as well as $\vec{\pi}_1 = \prod_{i=1}^{\ell} \vec{\pi}_{i,1}^{\beta_i}$ and $\vec{\pi}_2 = \prod_{i=1}^{\ell} \vec{\pi}_{i,2}^{\beta_i}$.
2. Re-randomize commitments $\vec{C}_{\sigma_1}, \vec{C}_{\sigma_3}, \vec{C}_{\sigma_4}$ and the proofs $\vec{\pi}_1, \vec{\pi}_2$. Finally, return the re-randomized signature $\sigma' = \big(\vec{C}'_{\sigma_1}, \sigma'_2, \vec{C}'_{\sigma_3}, \vec{C}'_{\sigma_4}, \vec{\pi}'_1, \vec{\pi}'_2\big)$.

Verify$(\mathsf{pk}, \tau, \vec{y}, \sigma)$**:** given pk, a signature $\sigma = \big(\vec{C}_{\sigma_1}, \sigma_2, \vec{C}_{\sigma_3}, \vec{C}_{\sigma_4}, \vec{\pi}_1, \vec{\pi}_2\big) \in \mathbb{G}^{16}$ and a message $(\tau, \vec{y})$, where $\tau \in \{0,1\}^L$ and $\vec{y} = (y_1, \ldots, y_n) \in (\mathbb{Z}_p)^n$, return $\perp$ if $\vec{y} = \vec{0}$. Otherwise, return 1 if and only if equations (9)-(10) are satisfied.

The properties of Groth-Sahai proofs guarantee that the scheme is completely hiding as established by Theorem 3.

Theorem 3. *The scheme is completely context hiding.*

Proof. The statement follows from the fact that, on a perfectly hiding CRS $(\vec{f}_1, \vec{f}_2, \vec{f}_3)$, all commitments are perfectly hiding and arguments are perfectly WI. Moreover, signature components σ_2, commitments $\vec{C}_{\sigma_1}, \vec{C}_{\sigma_3}, \vec{C}_{\sigma_4}$ and $\vec{\pi}_1, \vec{\pi}_2$ are perfectly re-randomized by the derivation algorithm. For this reason, the output of $\mathsf{SignDerive}$ has the same distribution as a fresh signature. □

In the proof of unforgeability, we will need a slightly stronger (but still simple) assumption than the standard CDH assumption.

The proof assumes that the adversary only obtains signatures on linearly independent vectors. This is not a limitation since, in practice, one usually augments the signed vectors (e.g., by unit vectors) so that they are always linearly independent. As in [20] and [6, Appendix F], we also assume that a given pair $(\tau, \vec{v})$ is always signed using the same s. This can be enforced by deriving s from a pseudo-random function of τ and $\vec{v}$.

Theorem 4. *The scheme is unforgeable assuming that the DLIN and FlexDH assumption both hold in the group* $\mathbb{G}$. (The proof is given in the full version of the paper).

Acknowledgements. The authors thank the anonymous reviewers for useful comments.

References

1. Abe, M., Haralambiev, K., Ohkubo, M.: Signing on Elements in Bilinear Groups for Modular Protocol Design. Cryptology ePrint Archive: Report 2010/133 (2010)
2. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-Preserving Signatures and Commitments to Group Elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
3. Agrawal, S., Boneh, D., Boyen, X., Freeman, D.M.: Preventing Pollution Attacks in Multi-source Network Coding. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 161–176. Springer, Heidelberg (2010)
4. Ahn, J.H., Boneh, D., Camenisch, J., Hohenberger, S., Shelat, A., Waters, B.: Computing on Authenticated Data. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 1–20. Springer, Heidelberg (2012)
5. 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)
6. Attrapadung, N., Libert, B., Peters, T.: Computing on Authenticated Data: New Privacy Definitions and Constructions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 367–385. Springer, Heidelberg (2012)
7. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable Proofs and Delegatable Anonymous Credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (2009)
8. Bellare, M., Neven, G.: Transitive Signatures Based on Factoring and RSA. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 397–414. Springer, Heidelberg (2002)
9. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
10. Boneh, D., Freeman, D., Katz, J., Waters, B.: Signing a Linear Subspace: Signature Schemes for Network Coding. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 68–87. Springer, Heidelberg (2009)
11. 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)

12. 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)
13. Brzuska, C., Busch, H., Dagdelen, O., Fischlin, M., Franz, M., Katzenbeisser, S., Manulis, M., Onete, C., Peter, A., Poettering, B., Schröder, D.: Redactable Signatures for Tree-Structured Data: Definitions and Constructions. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 87–104. Springer, Heidelberg (2010)
14. Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schröder, D., Volk, F.: Security of Sanitizable Signatures Revisited. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 317–336. Springer, Heidelberg (2009)
15. Brzuska, C., Fischlin, M., Lehmann, A., Schröder, D.: Unlinkability of Sanitizable Signatures. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 444–461. Springer, Heidelberg (2010)
16. 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)
17. 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)
18. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable Proof Systems and Applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 281–300. Springer, Heidelberg (2012)
19. Desmedt, Y.: Computer security by redefining what a computer is. In: New Security Paradigms Workshop (NSPW 1993), pp. 160–166 (1993)
20. 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)
21. Fuchsbauer, G.: Commuting Signatures and Verifiable Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 224–245. Springer, Heidelberg (2011)
22. Gennaro, R., Katz, J., Krawczyk, H., Rabin, T.: Secure Network Coding over the Integers. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 142–160. Springer, Heidelberg (2010)
23. Gerbush, M., Lewko, A., O'Neill, A., Waters, B.: Dual Form Signatures: An Approach for Proving Security from Static Assumptions. Cryptology ePrint Archive: Report 2012/261 (May 2012)
24. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC 2009, pp. 169–178 (2009)
25. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
26. Haber, S., Hatano, Y., Honda, Y., Horne, W., Miyazaki, K., Sander, T., Tezoku, S., Yao, D.: Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In: AsiaCCS 2008, pp. 353–362 (2008)
27. Hevia, A., Micciancio, D.: The Provable Security of Graph-Based One-Time Signatures and Extensions to Algebraic Signature Schemes. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 379–396. Springer, Heidelberg (2002)
28. Kiltz, E., Mityagin, A., Panjwani, S., Raghavan, B.: Append-Only Signatures. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 434–445. Springer, Heidelberg (2005)

29. Kunz-Jacques, S., Pointcheval, D.: About the Security of MTI/C0 and MQV. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 156–172. Springer, Heidelberg (2006)
30. Johnson, R., Molnar, D., Song, D., Wagner, D.: Homomorphic Signature Schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 244–262. Springer, Heidelberg (2002)
31. Libert, B., Vergnaud, D.: Multi-use unidirectional proxy re-signatures. In: ACM-CCS 2008, pp. 511–520 (2008)
32. Micali, S., Rivest, R.L.: Transitive Signature Schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 236–243. Springer, Heidelberg (2002)
33. Miyazaki, K., Hanaoka, G., Imai, H.: Digitally signed document sanitizing scheme based on bilinear maps. In: AsiaCCS 2006, pp. 343–354 (2006)
34. 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)
35. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
36. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

Verifiably Encrypted Signatures with Short Keys Based on the Decisional Linear Problem and Obfuscation for Encrypted VES

Ryo Nishimaki and Keita Xagawa

NTT Secure Platform Laboratories,
3-9-11 Midori-cho, Musashino-shi, Tokyo, 180-8585 Japan
{nishimaki.ryo,xagawa.keita}@lab.ntt.co.jp

Abstract. Verifiably encrypted signatures (VES) are signatures encrypted by a public key of a trusted third party and we can verify their validity without decryption. This paper proposes a new VES scheme which is secure under the decisional linear (DLIN) assumption in the standard model. We also propose new obfuscators for encrypted signatures (ES) and encrypted VES (EVES) which are secure under the DLIN assumption.

All previous efficient VES schemes in the standard model are either secure under standard assumptions (such as the computational Diffie-Hellman assumption) with large verification (or secret) keys or secure under *(non-standard) dynamic q-type assumptions* (such as the q-strong Diffie-Hellman extraction assumption) with short verification keys. Our construction is the first efficient VES scheme with short verification (and secret) keys secure under *a standard assumption (DLIN)*.

As by-products of our VES scheme, we construct new obfuscators for ES/EVES based on our new VES scheme. They are more efficient than previous obfuscators with respect to the public key size. Previous obfuscators for EVES are secure under non-standard assumption and use zero-knowledge (ZK) proof systems and Fiat-Shamir heuristics to obtain non-interactive ZK, i.e., its security is considered in the random oracle model. Thus, our construction also has an advantage with respect to assumptions and security models. Our new obfuscator for ES is obtained from our new obfuscator for EVES.

Keywords: verifiably encrypted signature, obfuscation, encrypted verifiably encrypted signature, decisional linear assumption.

1 Introduction

1.1 Background

In verifiably encrypted signature (VES) schemes, there are a signer, verifiers, and *a trusted third party, called the adjudicator*. The signer generates a signature, encrypts it under the public key of the adjudicator, and adds extra contents

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 405–422, 2013.

to make it verifiable without decryption. The adjudicator can recover ordinary signatures from encrypted ones by using his/her decryption key.

VES was introduced by Asokan, Shoup, and Waidner [2] and Boneh, Gentry, Lynn, and Shacham proposed an efficient (non-interactive) VES scheme based on Boneh-Lynn-Shacham signature scheme in the random oracle model (ROM) [8,9]. VES has useful and important applications such as online contract signing and optimistic fair exchange [2,3]. Suppose a situation that a user, say Alice, wants to buy digital goods from a company online. Alice gives the company her VES for a contract instead of paying money and the company returns the requested digital goods if it receive a valid VES. Alice sends an ordinary signature as effective one to the company if she receives the goods. If a malicious company does not return the requested goods when it receives a VES, Alice can claim that the VES is of no use for the contract since it is encrypted. If malicious Alice does not return a ordinary signature when she receives the goods, the company sends the encrypted signature together with the transcript to the adjudicator and the adjudicator extracts an ordinary signature from the VES by using the secret key of the adjudicator and returns it to the company. The adjudicator is offline, that is, it should be active only when malicious Alice cheats the company. As another application, Fuchsbauer used a certain kind of VES to construct delegatable anonymous credentials [16]. Anonymous credentials are very useful for access control [5]. In some system with access control, users must prove to have the required credential issued by an authority to use the system. The authority may want to delegate its right to other entities to avoid centralization of power.

Lu, Ostrovsky, Sahai, Shacham, and Waters proposed a VES scheme which is secure under the computational Diffie-Hellman (CDH) assumption in the standard model, but the verification key size is quite large [24]. Rückert and Schröder proposed a VES scheme with short verification keys, but its security relies on a non-standard q-type assumption, called q-strong DH extraction assumption [27]. They did not prove its hardness in the generic group model [28]. Thus, there is no VES scheme that achieves constant size verification key and signature based on standard assumptions.

Program Obfuscation and Encrypted Signature/VES. Encrypted VES (EVES) is an extension of encrypted signature (ES) proposed by Hada [23]. ES/EVES functionalities output encryption of signatures/VES. They do not encrypt messages but signatures, and can be used as building blocks of signcryption functionalities as Hada pointed out [23]. If Alice uses free web-mail services to send a mail to Bob on low computational power devices such as smart-phones and her web browsers do not have enough resources to sign messages and encrypt them with Bob's public key, then she wants web-mail providers to carry out its process instead of her. However, she does not want to reveal her signing key. The obfuscation for ES/EVES will give a solution. A program obfuscator is an algorithm which transforms a program into a completely unintelligible program whose functionality is the same as the original one [4,22]. Informally speaking, obfuscators should guarantee that what is efficiently computed given an obfuscated program is nothing more than what is computed given *black-box access* to the original

program. If Alice gives an obfuscated program for ES/EVES functionalities, then she can securely delegate her signing capability to web-mail providers. Moreover, in a situation that president Alice on vacation want to have vice president Carol sign contracts for Bob (only Alice to Bob) instead of her, Alice can give Carol an obfuscated program for EVES functionality. In the case of the obfuscator for ES by Hada, if a malicious party has access to Bob's decryption key, then Alice's signing key is extracted from the obfuscated program [23]. However, in the case of our obfuscator for EVES, even such a malicious party cannot extract Alice's key due to the existence of the adjudicator's key. Thus, obfuscators for EVES have useful applications.

Hada proposed a secure obfuscator for an ES functionality and its application to signcryption [23]. His scheme is secure under the DLIN assumption in the standard model, but the verification key size is quite large. Cheng, Zhang, and Zhang proposed a secure obfuscator for an EVES functionality at ProvSec'11 [13]. Their VES scheme and obfuscator for EVES use zero-knowledge (ZK) proofs and Fiat-Shamir heuristics to crash ZK proofs into non-interactive zero-knowledge (NIZK) proofs. That is, their scheme and obfuscator are *secure in the ROM*. Furthermore, they used a non-standard assumption, called exponent 3-weak DH assumption to prove the unforgeability of their scheme and did not prove opacity (explained in the next section), which is required for secure VES schemes, of their scheme.

In general, obfuscators for ES/EVES can be obtained from fully homomorphic encryption (FHE) schemes [17]. However, existing FHE schemes are still inefficient [11,12,14,18–20,29]. so we do not rely on expensive FHE schemes but directly construct obfuscators for ES/EVES.

1.2 Our Contributions and Constructions

We propose a new efficient VES scheme based on the *decisional linear assumption (DLIN)* in the standard model. Our main advantages over previous VES schemes are as follows:

1. It is efficient and secure under a standard (i.e., not q-type) assumption in the standard model.
2. The verification key and signature size is small (constant).

As a by-product of our VES scheme, we construct secure obfuscators for ES/EVES functionality based on the DLIN assumption in the standard model. Main advantages of our obfuscators for ES/EVES over previous obfuscators for ES/EVES are as follows: They are *secure under the DLIN assumption in the standard model with short verification keys.*

Comparison and Related Works. Comparisons of our results and previous results of VES schemes and obfuscators for ES/EVES are shown in Table 1 and in Table 2, respectively. Let λ denote the security parameter. In this paper, the CDH assumption is considered in bilinear groups. There is no efficient VES scheme and obfuscator for ES/EVES which are secure under standard assumptions in

Table 1. A summary of previous efficient schemes and ours for VES

Reference	Key size (vk/sk)	VES size	Assumptions	ROM
BGLS [8]	$1\mathbb{G}/1\mathbb{Z}_p$	$2\mathbb{G}$	CDH	Yes
ZSS [32]	$2\mathbb{G}/2\mathbb{Z}_p$	$1\mathbb{G}$	CDH	Yes
LOSSW [24]	$O(\lambda)\mathbb{G}(> 160\mathbb{G})/1\mathbb{Z}_p$	$3\mathbb{G}$	CDH	No
RS [27]	$4\mathbb{G}/2\mathbb{Z}_p$	$2\mathbb{G} + 1\mathbb{Z}_p$	q-strong DH extraction	No
This work	$16\mathbb{G} + 1\mathbb{G}_T/3\mathbb{G}$	$12\mathbb{G} + 2\mathbb{Z}_p$	*DLIN*	No

Table 2. A summary of previous obfuscation for encrypted ES/EVES

Reference	ES/EVES	Key size (vk)	ROM	Assumptions
Hada [23]	ES	$O(\lambda)$	No	DLIN
CZZ [13]	EVES	$O(\lambda)$	Yes	DLIN and Exponent 3-weak DH
This work	ES	$O(1)$	*No*	*DLIN*
This work	EVES	$O(1)$	*No*	*DLIN*

the standard model with short verification keys prior to our work. The VES scheme by Lu et al. needs a quite large verification key but its signature size is small and its security is based on a standard CDH assumption, so one may think that the scheme of Lu et al. is better than our scheme in terms of signature size. However, we think it is incomparable with our new scheme and we showed a tradeoff between the verification key size and signature size. Rückert proposed a VES scheme based on full-domain hash RSA signature, but it is secure in the ROM [25]. Rückert, Schneider, and Schröder proposed *generic* constructions for VES without NIZKs, pairings, and ROM. Their construction is very insightful, but their schemes use an extra adjudication setup phase and Merkle trees, so they need to setup large parameters and have large keys (non-constant size), that is, they are *inefficient* [26].

Our Construction Technique. Loosely speaking, a VES scheme consists of a signature scheme and a encryption scheme as Lu et al. and Rückert and Schröder [24, 27]. We use a signature scheme presented by Waters at CRYPTO'09 [31] as an underlying signature scheme. We call it the Waters dual signature in this paper to distinguish from Waters' signature at Eurocrypt'05 [30]. Someone may think that a combination of the Waters dual signature and ElGamal encryption easily yields a secure VES scheme under the DLIN assumption, *but that is not the case*. The reason is as follows: We can prove unforgeability of VES by relying on unforgeability of the underlying signature scheme as previous schemes [8, 24, 27], *but opacity is non-trivial*. Opacity means that it is difficult to extract ordinary signatures from VES, i.e., decrypt VES. Moreover, it is highly non-trivial whether we can prove opacity from standard assumptions or not. The reason is as follows: The VES scheme of Lu et al. is a combination of Waters' signature (Eurocrypt'05) [30] and the ElGamal encryption scheme and they proved its opacity from the aggregate extraction assumption [8] (fortunately, it is equivalent to the CDH assumption [15]). On the other hand, the VES scheme of

Rückert and Schröder is a combination of Boneh-Boyen signature scheme [6] and the ElGamal encryption scheme, but they proved its opacity from q-strong DH extraction assumption, which is a *stronger assumption than that of underlying Boneh-Boyen signature scheme* [27].

Our construction is a combination of the Waters dual signature scheme and the ElGamal encryption scheme. We encrypt only signature elements *related to signing keys*. The security proof of the Waters dual signature is different from that of many known secure signature schemes such as Boneh-Boyen [6], Waters [30], so we must employ a different proof strategy from that of Lu et al. and Rückert and Schröder. The Waters dual signature has two types of signature, standard signature (which is called type A) and *semi-functional* signature (which is called type B). Semi-functional signatures also pass the verification algorithm as standard ones and are *indistinguishable from standard ones* [21,31]. We extend the proof strategy of this dual form signature technique to prove opacity. First, we employ type B signatures as normal signatures output by a normal signing algorithm and type A signatures are used for simulation. Both type A and B signatures are valid signatures and there is no essential difference in terms of functionality as long as a normal verification algorithm is used. We employ this swap of role since we do not know how to prove that the adversary cannot extract a valid type A signature from given VES when the oracle answers type A signatures.

In the experiment of opacity, the adversary can output a pair of a signature and a message such that the message was queried to an oracle which returns a VES for the queried message. This causes the main difficulty for proving the opacity since the adversary may output a re-randomized signature obtained by using valid signatures from oracles. Unfortunately, the Waters dual signature is re-randomizable. Thus, we modify the Waters dual signature scheme to make it *strongly* unforgeable. Strong unforgeability guarantees that the adversary cannot output a forgery even for a queried message, so it must hold that if the adversary output valid signature for queried message in the experiment of opacity, then the signature is identical to the signature generated by the VES creation oracle (otherwise, contradict to strong unforgeability). This fact can be used to prove the opacity of our scheme.

In the proof of opacity, we must simulate two oracles. One is the creation oracle, which answers VES for queried messages. The other is the adjudication oracle, which extracts ordinary signatures from queried message/VES pairs and returns them. When we answer only encryption of type B signature for VES creation queries of the adversary, we can prove that the adversary cannot extract type B signature from VES under the aggregate extraction assumption. This is the reason why we swap the role of type A signatures for that of type B signature. We have no way to prove that when we answer only encryption of type A signature for VES creation queries of the adversary, adversary cannot extracts type A signature from VES.

Thus, it is showed that the adversary cannot output a valid signature for *queried message* to the VES creation oracle. For non-queried messages, we can

use the proof technique for unforgeability of dual form signatures. We show that the adversary cannot output a type A signature when the oracle returns type B signatures (VES).

Next, we change the type of signatures used to generate VES which are answered by the VES creation oracle. Answers of the adjudication oracle depend on the type of the VES creation oracle. Thus, we show that the view of the adversary is indistinguishable even if the type of answers are changed from type B to type A one-by-one for each query. This order of change is reverse to the original proof, but it is not essential difference. Lastly, we show that the adversary cannot output a type B signature when the oracle returns type A signatures (VES).

Secure obfuscations for ES and EVES based on the Waters dual signature scheme are also non-trivial because the signing keys of the Waters dual signature scheme consist of multiple group elements and the signing algorithm computes exponentiation of the signing keys with randomness in contrast to Waters' signature presented at Eurocrypt'05, whose signing key is only one group element and signing algorithm only multiplies it by other group elements [30]. We overcome this hurdle by using additive homomorphic property of ElGamal and the linear encryption schemes [7]. Cheng et al. use the linear encryption scheme for not only encryption of VES but also the construction of VES itself, so their VES scheme cannot check the validity of ciphertext by using only the pairing technique and they need (NI)ZK [13]. We do not need (NI)ZK because our new VES scheme uses the ElGamal encryption scheme and can verify the validity of VES by using only pairings.

Remark. In this extended abstract, we do not have enough space to write complete proofs and all definitions, so we omitted some of them.

2 Preliminaries

Notations and Conventions. For any $n \in \mathbb{N} \setminus \{0\}$, let $[n]$ be the set $\{1, \ldots, n\}$. When D is a random variable or distribution, $y \xleftarrow{R} D$ denote that y is randomly selected from D according to its distribution. If S is a set, then $x \xleftarrow{U} S$ denotes that x is uniformly selected from S. $y := z$ denotes that y is set, defined or substituted by z. When b is a fixed value, $A(x) \to b$ (e.g., $A(x) \to 1$) denotes the event that machine (or algorithm) A outputs a on input x. We say that function $f : \mathbb{N} \to \mathbb{R}$ is negligible in $\lambda \in \mathbb{N}$ if for every constant $c \in \mathbb{N}$ there exists $k_c \in \mathbb{N}$ such that $f(\lambda) < \lambda^{-c}$ for any $\lambda > k_c$. Hereafter, we use $f < \mathsf{negl}(\lambda)$ to mean that f is negligible in λ. Let $\Gamma := (p, \mathbb{G}, \mathbb{G}_T, e, g)$ be a description of groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p equipped with efficient bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We often omit common parameters Γ. Let $\mathcal{G}_{\mathsf{bmp}}$ be a standard parameter generation algorithm for bilinear maps that outputs Γ.

Definition 1 (DLIN assumption). *The DLIN problem is to guess $\beta \in \{0,1\}$, given $(\Gamma, g, f, \nu, g^x, f^y, Q_\beta) \xleftarrow{R} \mathcal{G}_\beta^{\mathsf{dlin}}(1^\lambda)$, where $\mathcal{G}_\beta^{\mathsf{dlin}}(1^\lambda)$: $\Gamma := (p, \mathbb{G}, \mathbb{G}_T, e, g) \xleftarrow{R} \mathcal{G}_{\mathsf{bmp}}(1^\lambda)$, $f, \nu \xleftarrow{U} \mathbb{G}, x, y \xleftarrow{U} \mathbb{Z}_p, Q_0 := \nu^{x+y}, Q_1 \xleftarrow{U} \mathbb{G}$, return $(\Gamma, g, f, \nu, g^x, f^y, Q_\beta)$.*

The advantage is $\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathcal{A}(\mathcal{I}) \to 1 \;\middle|\; \mathcal{I} \xleftarrow{R} \mathcal{G}_0^{\mathsf{dlin}}(1^\lambda)\right] - \Pr\left[\mathcal{A}(\mathcal{I}) \to 1 \;\middle|\; \mathcal{I} \xleftarrow{R} \mathcal{G}_1^{\mathsf{dlin}}(1^\lambda)\right]\right|$. *We say that the DLIN assumption holds if for all probabilistic polynomial-time (PPT) adversary* $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A}}(\lambda) < \mathsf{negl}(\lambda)$.

Definition 2 (Aggregate Extraction (AgExt) assumption [8, 15]). *The AgExt problem in bilinear groups is to compute* g^{xy}, *given* $\Gamma := (p, \mathbb{G}, \mathbb{G}_T, e, g) \xleftarrow{R} \mathcal{G}_{\mathsf{bmp}}(1^\lambda)$ *and* $(g^x, g^y, g^\beta, g^\delta, g^{xy+\beta\delta})$ *for* $x, y, \beta, \delta \xleftarrow{U} \mathbb{Z}_p$. *The advantage is* $\mathsf{Adv}^{\mathsf{AgExt}}_{\mathcal{A}}(\lambda) := \Pr[z = g^{xy} \mid \Gamma \xleftarrow{U} \mathcal{G}_{\mathsf{bmp}}(1^\lambda); x, y, \beta, \delta \xleftarrow{U} \mathbb{Z}_p; z \xleftarrow{R} \mathcal{A}(\Gamma, g^x, g^y, g^\beta, g^\delta, g^{xy+\beta\delta})]$. *We say that the AgExt assumption holds in bilinear groups if for any PPT* $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{AgExt}}_{\mathcal{A}}(\lambda) < \mathsf{negl}(\lambda)$.

Definition 3 (CDH assumption). *The CDH problem in bilinear groups is to compute* g^{xy}, *given* $\Gamma := (p, \mathbb{G}, \mathbb{G}_T, e, g) \xleftarrow{R} \mathcal{G}_{\mathsf{bmp}}(1^\lambda)$ *and* (g^x, g^y) *for* $x, y \xleftarrow{U} \mathbb{Z}_p$. *The advantage is* $\mathsf{Adv}^{\mathsf{CDH}}_{\mathcal{A}}(\lambda) := \Pr[z = g^{xy} \mid \Gamma \xleftarrow{U} \mathcal{G}_{\mathsf{bmp}}(1^\lambda); x, y \xleftarrow{U} \mathbb{Z}_p; z \xleftarrow{R} \mathcal{A}(\Gamma, g^x, g^y)]$. *We say that the CDH assumption holds in bilinear groups if for any PPT* $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CDH}}_{\mathcal{A}}(\lambda) < \mathsf{negl}(\lambda)$.

The AgExt assumption is equivalent to computational Diffie-Hellman (CDH) assumption, which is implied by the DLIN assumption.

Theorem 1 ([15]). *The AgExt and CDH problems are Karp reducible to each other with* $O(1)$ *computation.*

Verifiably Encrypted Signature (VES). A VES scheme consists of following seven algorithms $\mathsf{VES} = \{\mathsf{AdjGen}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy}, \mathsf{Create}, \mathsf{VesVrfy}, \mathsf{Adj}\}$:

Adjudicator Key Generation: Algorithm AdjGen takes as input security parameter 1^λ and outputs a pair of key for an adjudicator, that is, $(apk, ask) \xleftarrow{R} \mathsf{AdjGen}(1^\lambda)$.

Key Generation: Algorithm Gen takes as input 1^λ and outputs a pair of keys for a signer, that is, $(vk, sk) \xleftarrow{R} \mathsf{Gen}(1^\lambda)$. They are called the verification key and the signing key, respectively.

Signing: Algorithm Sign takes as input a signing key and a message and outputs signature σ. That is, $\sigma \xleftarrow{R} \mathsf{Sign}(sk, M)$, where $M \in \mathcal{M}_{vk}$ and $\mathcal{M}_{vk}$ is a message space defined by vk.

Verification: Algorithm Vrfy is deterministic and takes as input vk, M, and σ and outputs bit b. If $b = 1$ then the signature is valid. Else, it is invalid. That is, $\mathsf{Vrfy}(vk, \sigma, m) \to b$.

VES Creation: Algorithm Create takes as input sk, apk, and M and outputs VES ω on M. That is, $\omega \xleftarrow{R} \mathsf{Create}(sk, apk, M)$.

VES Verification: Algorithm $\mathsf{VesVrfy}$ is deterministic and takes as input apk, vk, ω, and M and outputs bit b, $\mathsf{VesVrfy}(apk, vk, \omega, M) \to b$.

Adjudication: Algorithm Adj takes as input ask, apk, vk, ω, and M. If ω is valid, it extracts an ordinary signature σ on M and returns σ, that is $\sigma \xleftarrow{R} \mathsf{Adj}(ask, apk, vk, \omega, M)$ if $\mathsf{VesVrfy}(apk, vk, \omega, M) \to 1$.

For correctnes, it is required that $\forall\lambda\ \forall(apk, ask) \xleftarrow{R} \mathsf{AdjGen}(1^\lambda)\ \forall(vk, sk) \xleftarrow{R} \mathsf{Gen}(1^\lambda)\ \forall m \in \mathcal{M}_{vk}\ \ \mathsf{VesVrfy}(apk, pk, \mathsf{Create}(sk, apk, M), M) \to 1$ and $\mathsf{Vrfy}(vk, \mathsf{Adj}(ask, apk, vk, \mathsf{Create}(sk, apk, M)), M) \to 1$.

Experiments $\mathsf{VesForge}_{\mathcal{A}}(\lambda)$ and $\mathsf{Opac}_{\mathcal{A}}(\lambda)$ are defined as follows:

Experiment $\mathsf{VesForge}_{\mathcal{A}}(\lambda)$
$(apk, ask) \xleftarrow{R} \mathsf{AdjGen}(1^\lambda)$;
$(vk, sk) \xleftarrow{R} \mathsf{Gen}(1^\lambda)$;
$(M^*, \omega^*) \xleftarrow{R} \mathcal{A}^{\mathcal{CO}(sk,apk,\cdot),\mathcal{AO}(ask,apk,vk,\cdot,\cdot)}(vk, apk)$;
Return 1 iff
$\mathsf{VesVrfy}(apk, vk, \omega^*, M^*) \to 1$ and
$M^* \notin Q_{\mathsf{C}}$ and $M^* \notin Q_{\mathsf{A}}$.

Experiment $\mathsf{Opac}_{\mathcal{A}}(\lambda)$
$(apk, ask) \xleftarrow{R} \mathsf{AdjGen}(1^\lambda)$;
$(vk, sk) \xleftarrow{R} \mathsf{Gen}(1^\lambda)$;
$(M^*, \sigma^*) \xleftarrow{R} \mathcal{A}^{\mathcal{CO}(sk,apk,\cdot),\mathcal{AO}(ask,apk,vk,\cdot,\cdot)}(vk, apk)$;
Return 1 iff
$\mathsf{Vrfy}(vk, \sigma^*, M^*) \to 1$ and
$M^* \notin Q_{\mathsf{A}}$.

where the creation oracle, $\mathcal{CO}(sk, apk, \cdot)$, returns a VES for a queried message, the adjudication oracle, $\mathcal{AO}(ask, apk, vk, \cdot, \cdot)$, extracts and returns a signature for a queried message/VES pair, and Q_{C} and Q_{A} are sets of messages queried by the adversary to $\mathcal{CO}$ and $\mathcal{AO}$, respectively.

Definition 4 (Secure VES [8]). *A VES scheme is secure if it satisfies unforgeability and opacity, i.e., it holds for any PPT* $\mathcal{A}$, $\Pr[\mathsf{VesForge}_{\mathcal{A}}(\lambda) \to 1] < \mathsf{negl}(\lambda)$ *and* $\Pr[\mathsf{Opac}_{\mathcal{A}}(\lambda) \to 1] < \mathsf{negl}(\lambda)$.

Collision Resistant Hash Functions (CRHF). Let $\mathcal{H} := \{H_k\}$ be a keyed hash family of functions $H_k : \{0,1\}^* \to \{0,1\}^n$ indexed by $k \in \mathcal{K}_\lambda$ where λ is a security parameter.

Definition 5. *We say that* $\mathcal{H}$ *is* (t, ϵ)*-collision-resistant if for any adversary* $\mathcal{A}$ *running in time* t*, we have that* $\mathsf{Adv}^{\mathsf{CRHF}}_{\mathcal{A},\mathcal{H}}(\lambda) := \Pr[m_0 \neq m_1 \wedge H_k(m_0) = H_k(m_1) \mid (m_0, m_1) \xleftarrow{R} \mathcal{A}(k)] < \epsilon$ *where the probability is taken over the random choice of* $k \in \mathcal{K}_\lambda$ *and random coins of* $\mathcal{A}$.

3 Strongly Unforgeable Waters Dual Signature

Waters Dual Signature Scheme. We review a signature scheme presented by Waters [31] since we use it as a essential building block. However, we add a few minor changes to fit the scheme to this paper. We will explain the differences between the original scheme and modified scheme WdSig.

$\mathsf{Wd.Gen}(1^\lambda, \Gamma)$**:** On input security parameter λ and $\Gamma := (p, \mathbb{G}, \mathbb{G}_T, e, g) \xleftarrow{R} \mathcal{G}_{\mathsf{bmp}}(1^\lambda)$, it chooses generators $v, v_1, v_2, w, u, h \xleftarrow{U} \mathbb{G}$ and exponent $a_1, a_2, b, \alpha \xleftarrow{U} \mathbb{Z}_p$, computes $\tau_1 := vv_1^{a_1}, \tau_2 := vv_2^{a_2}$, and sets $VK := (\Gamma, g^b, g^{a_1}, g^{a_2}, g^{ba_1}, g^{ba_2}, v, v_1, v_2, \tau_1, \tau_2, \tau_1^b, \tau_2^b, w, u, h, e(g,g)^{\alpha a_1 b})$ and $SK := (VK, g^\alpha, g^{\alpha a_1}, g^{a_1 a_2})$. Hereafter we often omit input 1^λ.

$\mathsf{Wd.Sign}(SK, M)$**:** On input message $M \in \mathbb{Z}_p$, it selects $r_1, r_2, z_1, z_2, \gamma, \mathsf{stag} \xleftarrow{\cup} \mathbb{Z}_p$, sets $r := r_1 + r_2$, computes $\mathsf{sig} := (\sigma_0, \sigma_1, \ldots, \sigma_7, \mathsf{stag})$, where

$$\sigma_0 := (u^M w^{\mathsf{stag}} h)^{r_1}, \quad \sigma_1 := g^{\alpha a_1} v^r g^{-a_1 a_2 \gamma}, \quad \sigma_2 := g^{-\alpha} v_1^r g^{z_1} g^{a_2 \gamma},$$
$$\sigma_3 := (g^b)^{-z_1}, \quad \sigma_4 := v_2^r g^{z_2} g^{a_1 \gamma}, \quad \sigma_5 := (g^b)^{-z_2},$$
$$\sigma_6 := (g^b)^{r_2}, \quad \sigma_7 := g^{r_1}.$$

$\mathsf{Wd.Vrfy}(VK, \mathsf{sig}, M)$**:** On input VK, M, and sig, it outputs 1 if and only if it holds that

$$e(u^M w^{\mathsf{stag}} h, \sigma_7) = e(g, \sigma_0),$$
$$e(g^b, \sigma_1) e(g^{b a_1}, \sigma_2) e(g^{a_1}, \sigma_3) = e(\tau_1, \sigma_6)\, e(\tau_1^b, \sigma_7),$$
$$e(g^b, \sigma_1)\, e(g^{b a_2}, \sigma_4)\, e(g^{a_2}, \sigma_5) = e(\tau_2, \sigma_6)\, e(\tau_2^b, \sigma_7)\, e(g, g)^{\alpha a_1 b}.$$

The differences are as follows: In the original Waters dual signature scheme, (1) the verification equation is only one equation and probabilistic, (2) values v, v_1, v_2 are included in secret keys, (3) value $g^{a_1 a_2}$ is not included in the signing key, (4) the (normal) signing algorithm does not multiply $g^{-a_1 a_2 \gamma}$, $g^{a_2 \gamma}$, $g^{a_1 \gamma}$ in σ_1, σ_2, σ_4, respectively.

There are two types of signatures in the Waters dual signature scheme, type A (if $\gamma = 0$) and type B (if $\gamma \neq 0$) signatures. The modified three verification equations above are introduced by Abe et al. [1]. They proved that if a signature passes the equations, then the signature is either type A or B. The original equations use ciphertexts and the decryption procedure of the Waters dual encryption scheme, so it is probabilistic and has a semi-functional verification algorithm that uses semi-functional ciphertexts [31]. Type A signatures are signatures with $\gamma = 0$ and pass both the normal and semi-functional verification equations. Type B signatures are signatures with $\gamma \neq 0$ and cannot pass the semi-functional verification equations (Gerbush, Lewko, O'Neill, and Waters defined them as backdoor verification tests [21]). As long as the verification equations are normal, both type A and type B signatures are valid signatures and there is no essential difference. Thus, we employ type B signatures in the normal signing algorithm.

Even if v, v_1, v_2 are disclosed, we cannot compute v_2^b (and semi-functional ciphertexts of the dual system encryption of Waters [31]). Thus, we add (v, v_1, v_2) to the verification key and this does not affect its security since g^α and $g^{\alpha a_1}$ (and v_2^b) are kept secret and they are essential secret signing keys. This is observed by Abe et al. [1]. For the minor changed version above, the following theorem holds [1, 31].

Theorem 2. *If the DLIN assumption holds, then* $\mathsf{WdSig} := \mathsf{Wd}.\{\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy}\}$ *is existentially unforgeable against adaptive chosen message attacks (EUF-CMA).*

The original Waters dual signature is not strongly unforgeable since it is re-randomizable. "Strong" means that the adversary cannot forge a signature even for a queried message to the signing oracle. In order to make our VES scheme

satisfy opacity, we modify the Waters dual signature. We extend the technique by Boneh, Shen, and Waters [10]. They introduced a property called 2-partitioned to convert unforgeable signature schemes into *strongly* unforgeable signature schemes. We extend 2-partitioned to 3-partitioned.

Definition 6. *A signature scheme is* 3*-partitioned if it satisfies the following two properties:*

- *The signing algorithm consists of three deterministic algorithms* F_1*,* F_2*, and* F_3
  1. *chooses random* $R \in \mathcal{R}$ *(*$\mathcal{R}$ *is a space for randomness),*
  2. *computes* $\Sigma_1 := F_1(M, R, VK)$*,* $\Sigma_2 := F_2(R, VK)$*,* $\Sigma_3 := F_3(R, SK)$*,*
  3. *and outputs signature* $\sigma := (\Sigma_1, \Sigma_2, \Sigma_3)$*.*
- *Given* M *and* Σ_2 *there is at most one* (Σ_1, Σ_3) *such that* $(\Sigma_1, \Sigma_2, \Sigma_3)$ *is a valid signature on* M *under* VK*.*

A 2-partitioned signature is $\sigma = (\Sigma_1', \Sigma_2')$ where $\Sigma_1' = F_1'(M, R, SK)$ and $\Sigma_2' = F_2'(R, SK)$ [10]. Value Σ_2' binds all randomness R, so M and R fully determine Σ_1'. For VES, signature elements related to the secret signing key (i.e., Σ_3) should be encrypted, so we cannot use such elements as inputs to hash functions (we will use hash functions to obtain strongly secure signature) and want to isolate the secret signing key from Σ_2'. Otherwise, encrypted signatures are not verifiable. If Σ_3 is not used as an input of hash function, then hash values are not changed even if Σ_3 is encrypted. This is the reason why we introduced 3-partitioned and Σ_1 and Σ_2 are independent of the secret signing key.

Let $\Pi := (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be an existentially unforgeable signature scheme. New signature scheme $\Pi' := (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ is as follows:

$\mathsf{Gen}'(1^\lambda)$**:** It generates $(VK, SK) \xleftarrow{R} \mathsf{Gen}(1^\lambda)$, chooses $\bar{h} \xleftarrow{U} \mathbb{G}$ and random hash key $k \in \mathcal{K}$, and sets $(VK', SK') := ((VK, \bar{h}, k), SK)$.

$\mathsf{Sign}'(SK', M)$**:** On input message $M \in \{0,1\}^\ell$, it chooses exponent $\varphi \xleftarrow{U} \mathbb{Z}_p$ and randomness $R \in \mathcal{R}$, computes $\Sigma_2 := F_2(R, VK)$, $\vartheta := H_k(M \parallel \Sigma_2)$ (view ϑ as an element in $\mathbb{Z}_p$), $m := H_k(g^\vartheta \bar{h}^\varphi)$, $\Sigma_1 := F_1(m, R, VK)$ and $\Sigma_3 := F_3(R, SK)$, and outputs a signature $\mathsf{sig} := (\Sigma_1, \Sigma_2, \Sigma_3, \varphi)$.

$\mathsf{Vrfy}'(VK', \mathsf{sig}, M)$**:** On input VK', M, and signature $\mathsf{sig} = (\Sigma_1, \Sigma_2, \Sigma_3, \varphi)$, it computes $\vartheta' := H_k(M \parallel \Sigma_2)$ (view ϑ' as an element $\mathbb{Z}_p$), $m' := H_k(g^{\vartheta'} \bar{h}^\varphi)$, It outputs 1 if and only if $\mathsf{Vrfy}(VK, (\Sigma_1, \Sigma_2, \Sigma_3), m') \to 1$.

Theorem 3. *Signature scheme* Π' *is* (t, q, ϵ)*-strongly existentially unforgeable if* Π *is* $(t, q, \epsilon/3)$*-existentially unforgeable, the* $(t, \epsilon/3)$*-DL assumption holds in* $\mathbb{G}$*, and* H *is* $(t, \epsilon/3)$*-collision-resistant.*

This is easily proved by extending the proof of Boneh, Shen, and Waters [10]. The DL assumption means the discrete logarithm assumption. The essential point is that given message M and partial signature Σ_2, the randomness which is used to generate the whole signature is determined and there is at most one (Σ_1, Σ_3) such that $(\Sigma_1, \Sigma_2, \Sigma_3)$ is a valid signature on M under VK. Intuitively, in the construction of Π', we sign not only message M but also randomness R to bind

the randomness and prevent re-randomization. Moreover, in order to prevent message m being determined by randomness R, new randomness φ is introduced and chameleon hash functions $(g^{\vartheta}\bar{h}^{\varphi})$ are used. Value m will be signed.

Theorem 4. *The Waters dual signature is* 3*-partitioned.*

Proof. Let $\mathcal{R} := \{(r_1, r_2, z_1, z_2, \mathsf{stag}, \gamma) |\ r_1, r_2, z_1, z_2, \mathsf{stag}, \gamma \xleftarrow{\cup} \mathbb{Z}_p\}$, then functions F_1, F_2, and F_3 are defined as follows: $R \xleftarrow{R} \mathcal{R}$, $F_1(M, R, VK) := \sigma_0 = (u^M w^{\mathsf{stag}} h)^{r_1}$, $F_2(R, VK) := (\sigma_3, \ldots, \sigma_7, \mathsf{stag}) = (g^{-bz_1}, v_2^r g^{z_2} \cdot g^{a_1\gamma}, g^{-bz_2}, g^{br_2}, g^{r_1}, \mathsf{stag})$, $F_3(R, SK) := (\sigma_1, \sigma_2) = (g^{\alpha a_1} v^r \cdot g^{-a_1 a_2 \gamma}, g^{-\alpha} v_1^r g^{z_1} \cdot g^{a_2\gamma})$ where $\gamma \xleftarrow{\cup} \mathbb{Z}_p$ is chosen for type B signatures. If the signature is type A, then $\gamma := 0$. We can interpret $\sigma_3, \sigma_5, \sigma_6, \sigma_7$ (outputs of F_2) as $g^{-bz_1}, g^{-bz_2}, g^{br_2}, g^{r_1}$, respectively and it follows $\sigma_0 = (u^M w^{\mathsf{stag}} h)^{r_1}$ from the first verification equation, that is, the output of F_1 is fixed. If we interpret σ_4 as $v_2^r g^{z_2} \cdot g^{a_1\gamma}$, then by the second and third equations two unknowns σ_1 and σ_2 are fixed to $g^{\alpha a_1} v^r \cdot g^{-a_1 a_2 \gamma}$ and $g^{-\alpha} v_1^r g^{z_1}$, respectively, that is, the output of F_3 is fixed. Thus, if the output of F_2 and M are fixed, then the outputs of F_1 and F_3 are also fixed.

We can see that even if (σ_1, σ_2) is encrypted by the ElGamal encryption, hash value $\vartheta = H_k(M \parallel (\sigma_3, \ldots, \sigma_7, \mathsf{stag}))$ is not changed, so it can be fitted to VES schemes. Note that we assume that each element $g \in \mathbb{G}$ has a unique encoding. We can obtain strongly secure scheme sWdSig:

$\mathsf{sWd.Gen}(1^\lambda, \Gamma)$**:** It generates $(VK', SK') \xleftarrow{R} \mathsf{Wd.Gen}(1^\lambda, \Gamma)$, chooses $\bar{h} \xleftarrow{\cup} \mathbb{G}$ and random hash key $k \in \mathcal{K}$, and sets $(VK, SK) := ((VK', \bar{h}, k), SK')$.

$\mathsf{sWd.Sign}(SK, M)$**:** On input message $M \in \mathbb{Z}_p$, it selects $r_1, r_2, z_1, z_2, \gamma, \mathsf{stag}, \varphi \xleftarrow{\cup} \mathbb{Z}_p$, sets $r := r_1 + r_2$, computes $\sigma_1 := g^{\alpha a_1} v^r \cdot g^{-a_1 a_2 \gamma}$, $\sigma_2 := g^{-\alpha} v_1^r g^{z_1} \cdot g^{a_2\gamma}$, $\sigma_3 := (g^b)^{-z_1}$, $\sigma_4 := v_2^r g^{z_2} \cdot g^{a_1\gamma}$, $\sigma_5 := (g^b)^{-z_2}$, $\sigma_6 := (g^b)^{r_2}$, $\sigma_7 := g^{r_1}$, $\vartheta := H_k(M \parallel \Sigma_2)$ where $\Sigma_2 = (\sigma_3, \ldots, \sigma_7, \mathsf{stag})$ and view ϑ as an element in $\mathbb{Z}_p$, $m := H_k(g^{\vartheta}\bar{h}^{\varphi})$, $\sigma_0 := (u^m w^{\mathsf{stag}} h)^{r_1}$, and outputs $\mathsf{sig} := (\sigma_0, \sigma_1, \ldots, \sigma_7, \mathsf{stag}, \varphi)$.

$\mathsf{sWd.Vrfy}(VK, \mathsf{sig}, M)$**:** On input VK, M, and signature $\mathsf{sig} = (\sigma_0, \sigma_1, \ldots, \sigma_7, \mathsf{stag}, \varphi)$, it computes $\vartheta' := H_k(M \parallel (\sigma_3, \ldots, \sigma_7, \mathsf{stag}))$, $m' := H_k(g^{\vartheta'}\bar{h}^{\varphi})$, and $\mathsf{Wd.Vrfy}(VK', \mathsf{sig}', m') \to b$ where $\mathsf{sig}' := (\sigma_0, \ldots, \sigma_7, \mathsf{stag})$, and outputs b.

Corollary 1. *The scheme above is strongly unforgeable against adaptive chosen message attacks if the DLIN assumption holds. In particular, for any PPT adversary* $\mathcal{F}$ *against* sWdSig *that makes at most* q *signing queries, there exists PPT algorithm* $\mathcal{B}'$ *for DLIN and* $\mathcal{C}$ *for CRHF,* $\mathsf{Adv}^{\mathsf{sEUF\text{-}CMA}}_{\mathcal{F},\mathsf{sWdSig}}(\lambda) \leq \{(q+3)/3\}\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{B}'} + (1/3)\mathsf{Adv}^{\mathsf{CRHF}}_{\mathcal{C},\mathcal{H}}$ *where* $\mathsf{Adv}^{\mathsf{sEUF\text{-}CMA}}_{\mathcal{F},\mathsf{sWdSig}}(\lambda)$ *and* $\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{F}',\mathsf{WdSig}}(\lambda)$ *is the advantage of the adversary for* sWdSig.

Note that the DL assumption is implied by the DLIN assumption.

4 Construction of Our VES

We present our VES scheme, sWdVES, based on the strongly secure variant the Waters dual signature scheme in this section. The proposed scheme is basically the same as the strongly unforgeable Waters dual signature scheme in Section 3 except that we encrypt signature elements which include secret keys $(g^{\alpha}, g^{\alpha a_1}, g^{a_1 a_2})$ by the ElGamal encryption scheme. That is, in our creation algorithm, only σ_1 and σ_2 are encrypted. In order to verify encrypted signatures, we add extra elements and cancel out group elements which are generated by pairing computation of encrypted signatures in the verification equation. sWdVES is as follows:

$\mathsf{AdjGen}(1^\lambda)$**:** It selects $\beta \xleftarrow{\cup} \mathbb{Z}_p$ and sets $apk := \zeta := g^\beta$ and $ask := \beta$.

$\mathsf{Gen}(1^\lambda)$**:** It generates $(VK', SK') := ((g, g^b, g^{a_1}, g^{a_2}, g^{ba_1}, g^{ba_2}, \tau_1, \tau_2, \tau_1^b, \tau_2^b, v, v_1, v_2, w, u, h, \bar{h}, k, e(g,g)^{\alpha a_1 b}), (g^\alpha, g^{\alpha a_1}, g^{a_1 a_2})) \xleftarrow{R} \mathsf{sWd.Gen}(1^\lambda)$ and sets $vk := VK'$ and $sk := (VK', SK')$.

Sign and Vrfy**:** Same as $\mathsf{sWd.\{Sign, Vrfy\}}$ in Section 3, respectively.

$\mathsf{Create}(sk, apk, M)$**:** It generates $(\sigma_0, \ldots, \sigma_7, \mathsf{stag}, \varphi) \xleftarrow{R} \mathsf{sWd.Sign}(SK', M)$, selects $\rho_1, \rho_2 \xleftarrow{\cup} \mathbb{Z}_p$, outputs $\omega := (K_0, \ldots, K_7, K_1', K_2', \hat{K}_1, \hat{K}_2, \mathsf{stag}, \varphi)$, where $(K_0, K_3, \ldots, K_7) := (\sigma_0, \sigma_3, \ldots, \sigma_7)$ and

$$K_1 := \sigma_1 \cdot \zeta^{\rho_1}, \qquad K_1' := g^{\rho_1}, \qquad \hat{K}_1 := (g^b)^{\rho_1},$$
$$K_2 := \sigma_2 \cdot \zeta^{\rho_2}, \qquad K_2' := g^{\rho_2}, \qquad \hat{K}_2 := (g^{ba_1})^{\rho_2}.$$

$\mathsf{VesVrfy}(apk, vk, \omega, M)$**:** It parses $\omega = (K_0, \ldots, K_7, K_1', K_2', \hat{K}_1, \hat{K}_2, \mathsf{stag}, \varphi)$, and computes $\vartheta' := H_k(M \parallel (K_3, \ldots, K_7, \mathsf{stag}))$, $m' := H_k(g^{\vartheta'} \bar{h}^{\varphi})$, It outputs 1 if and only if it holds that

$$e(K_1', g^b) = e(g, \hat{K}_1) \ , \ e(K_2', g^{ba_1}) = e(g, \hat{K}_2)$$
$$e(u^{m'} w^{\mathsf{stag}} h, K_7) = e(g, K_0)$$
$$\frac{e(g^b, K_1)}{e(\zeta, \hat{K}_1)} \cdot \frac{e(g^{ba_1}, K_2)}{e(\zeta, \hat{K}_2)} \cdot e(g^{a_1}, K_3) = e(\tau_1, K_6)\, e(\tau_1^b, K_7)$$
$$\frac{e(g^b, K_1)}{e(\zeta, \hat{K}_1)} \cdot e(g^{ba_2}, K_4)\, e(g^{a_2}, K_5) = e(\tau_2, K_6)\, e(\tau_2^b, K_7)\, e(g,g)^{\alpha a_1 b}$$

$\mathsf{Adj}(ask, apk, vk, \omega, M)$**:** It parses $\omega = (K_0, \ldots, K_7, \mathsf{stag}, \varphi)$ and computes $\sigma_1 := K_1 \cdot (K_1')^{-\beta}$, $\sigma_2 := K_2 \cdot (K_2')^{-\beta}$, $\sigma_3 := K_3$, $\sigma_4 := K_4$, $\sigma_5 := K_5$, $\sigma_6 := K_6$, $\sigma_7 := K_7$, $\sigma_0 := K_0$. If $\mathsf{VesVrfy}(apk, vk, \omega, M) \to 1$, then it outputs $(\sigma_0, \ldots, \sigma_7, \mathsf{stag}, \varphi)$. These are valid signatures.

Intuitively, the scheme above is secure because underlying signature scheme is strongly unforgeable. The adversary has no choice but to decrypt valid VES given by oracles to output a valid signature, but it contradicts to the one-wayness of the ElGamal encryption scheme.

Rückert and Schröder defined key-independence and extractability of VES to prove unforgeability and collusion-resistance of VES in a modular way [26, 27]. Key-independence means that a VES creation algorithm consists of a signature generation part and a transformation (into VES) part and they are independent. Extractability means that if VES ω is valid, then the adjudicator can extract a valid (ordinary) signature σ with except negligible probability. Collusion-resistance means that no adversary can forge VES even if the adjudicator is corrupted, i.e., adversary obtains the secret decryption key of the adjudicator. Rückert and Schröder showed the following theorem.

Theorem 5 ([27]). *Let* VES *be an extractable and key-independent verifiably encrypted signature scheme.* VES *is unforgeable if and only if the underlying signature scheme* Sig *is unforgeable.*

As a corollary, sWdVES is unforgeable under the DLIN assumption since we can easily show that our sWdVES based on sWdSig is key-independent and extractable though we omit proofs in this extended abstract.

Theorem 6. sWdVES *is opaque if the DLIN assumption holds and there exists CRHF.*

Proof. If adversary $\mathcal{A}$ outputs forgery $\sigma^* = (\sigma_0^*, \ldots, \sigma_7^*, \mathsf{stag}^*, \varphi^*)$ and M^* such that M^* is not queried to $\mathcal{AO}$, then it means that $\mathcal{A}$ breaks opacity of sWdVES. $\mathcal{A}$ directly forges a signature of underlying sWdSig or extracts a signature by breaking the one-wayness of the ElGamal encryption scheme. In order to prove opacity, we introduce the following games: Let Game-(i) denote a game where $\mathcal{CO}$ answers encryption of type A signatures for the first i ($i \in [q_{\mathsf{C}}]$ and q_{C} is the number of creation query by $\mathcal{A}$) queries and encryption of type B signatures for the remaining $(q_{\mathsf{C}} - i)$ queries and $\mathcal{AO}$ answers signatures extracted from queried VES for all q_{A} (the number of adjudication query) queries. Let $\mathsf{Adv}_i^{\mathsf{forge\text{-}A}}$ (resp. $\mathsf{Adv}_i^{\mathsf{forge\text{-}B}}$) denote the advantage of the adversary in Game-(i) for outputting type A (resp. B) forgery for a non-queried message (a message which is not queried to $\mathcal{CO}$). Let $\mathsf{Adv}_0^{\mathsf{extract\text{-}B}}$ denote the advantage of the adversary in Game-0 for extracting a type B signature from a VES for a queried message (a message which is queried to $\mathcal{CO}$).

1. In Game-(0), $\mathcal{CO}$ returns encryption of type B signature and $\mathcal{AO}$ returns type B signature. First, we show Lemma 1: If $\mathcal{A}$ outputs a valid type B signature for message M_i which has been *already queried to* $\mathcal{CO}$, then we can construct algorithm $\mathcal{E}$ which solves the AgExt problem. Thus, in the remaining games, we only consider $\mathcal{A}$ which outputs forgery for message M^* such that $M^* \neq M_i$ for all $i \in [q]$. We can show that if $\mathcal{A}$ outputs forgery of type A signature, then we can construct algorithm $\mathcal{B}_1$ which solves the CDH problem.
2. Next, we consider Game-(i). We can show that if $\mathcal{A}$ detects the change from type B answer to type A answer by $\mathcal{CO}$, then we can construct algorithm $\mathcal{B}_2$ which solves the DLIN problem.

3. Last, we consider $\mathsf{Game}\text{-}(q_{\mathsf{C}})$, where all answers for VES queries of $\mathcal{A}$ are encryption of type A signature. We can show that if $\mathcal{A}$ outputs a forgery of type B signature, then we can construct algorithm $\mathcal{B}_3$ which solves the DLIN problem.

Thus, if the DLIN assumption holds, the signature scheme is opaque. The core part is Lemma 1. By statements described above except Lemma 1, we can show $\mathsf{Adv}^{\mathsf{Opac}}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathsf{forge\text{-}A}}_{0} + \mathsf{Adv}^{\mathsf{extract\text{-}B}}_{0} + \mathsf{Adv}^{\mathsf{forge\text{-}B}}_{0} < \mathsf{Adv}^{\mathsf{extract\text{-}B}}_{0} + \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{F},\mathsf{WdSig}} < \mathsf{Adv}^{\mathsf{extract\text{-}B}}_{0} + (q_{\mathsf{C}}+2)\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{B}}$. By Lemma 1, we can show

$$\begin{aligned}\mathsf{Adv}^{\mathsf{extract\text{-}B}}_{0} &< q_{\mathsf{C}}\mathsf{Adv}^{\mathsf{AgExt}}_{\mathcal{E}} + \mathsf{Adv}^{\mathsf{sEUF\text{-}CMA}}_{\mathcal{F}',\mathsf{sWdSig}} + \mathsf{Adv}^{\mathsf{CRHF}}_{\mathcal{C}} \\ &< \frac{4q_{\mathsf{C}}+3}{3}\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{B}} + \frac{4}{3}\mathsf{Adv}^{\mathsf{CRHF}}_{\mathcal{C}}.\end{aligned}$$

Thus, it holds $\mathsf{Adv}^{\mathsf{Opac}}_{\mathcal{A}}(\lambda) < ((7q_{\mathsf{C}}+9)/3)\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{B}} + (4/3)\mathsf{Adv}^{\mathsf{CRHF}}_{\mathcal{C}}$.

Lemma 1. *If there exists adversary $\mathcal{A}$ that outputs a type B forgery for a queried message M_i in $\mathsf{Game}\text{-}(0)$, then we can construct algorithm $\mathcal{E}$ that solves the AgExt problem.*

Proof of lemma. $\mathcal{E}$ is given instance $(\Gamma, g^x, g^y, g^\beta, g^\delta, g^{xy+\beta\delta})$ of the AgExt problem. $\mathcal{E}$ generates the verification key as follows: Chooses exponents $a_1, b, y_v, y_{v_1}, y_{v_2}, y_w, y_h, y_u, \eta \xleftarrow{\cup} \mathbb{Z}_p$ and hash key $k \in \mathcal{K}$, computes $g := g$, $g^b := g^b$, $g^{a_1} := g^{a_1}$, $g^{a_2} := g^y$, $g^{ba_2} := (g^y)^b$, $g^{ba_1} := g^{ba_1}$, $v := g^{y_v}$, $v_1 := g^{y_{v_1}}$, $v_2 := g^{y_{v_2}}$, $w := g^{y_w}$, $u := g^{y_u}$, $h := g^{y_h}$, $\bar{h} := g^\eta$, $\zeta := g^\beta$, $e(g,g)^{\alpha a_1 b} := e(g^x, g^y)^{a_1 \cdot b}$ (it implicitly holds $\alpha = xy$ though $\mathcal{E}$ does not have α), $\tau_1 := vv_1^{a_1}$, τ_1^b, $\tau_2 := v(g^y)^{y_{v_2}}$, and τ_2^b, and sets $VK := (g, g^b, g^{a_1}, g^{a_2}, g^{ba_1}, g^{ba_2}, \tau_1, \tau_2, \tau_1^b, \tau_2^b, w, u, h, \bar{h}, k, e(g,g)^{\alpha a_1 b})$ and $apk := \zeta = g^\beta$. Note that $\mathcal{E}$ does not have $a_2 = y$ and $g^\alpha = g^{xy}$, so $\mathcal{E}$ cannot directly compute Type B signature.

Simulation of Creation Oracle: $\mathcal{E}$ initializes list $\mathsf{QList} := \emptyset$. $\mathcal{E}$ chooses random index $j \xleftarrow{\cup} [q_{\mathsf{C}}]$, i.e., guesses which VES $\mathcal{A}$ selects and outputs its extraction. $\mathcal{E}$ outputs encryption of Type B signatures for i-th VES creation query M_i as follows: If $i \neq j$, then chooses $r_1, r_2, z_1, z_2, \gamma', \mathsf{stag}, \varphi_i, \rho_1, \rho_2 \xleftarrow{\cup} \mathbb{Z}_p$, sets $r := r_1 + r_2$ (we want to set $\gamma := x + \gamma'$), computes $\sigma_{i,1} := (g^y)^{-\gamma' a_1} \cdot v^r = (g^{\alpha a_1} v^r) \cdot g^{-a_1 a_2 \gamma}$ (where $a_2 = y$ and $xy = \alpha$), $\sigma_{i,2} := (g^y)^{\gamma'} v_1^r g^{z_1} = (g^\alpha v_1^r g^{z_1}) \cdot g^{a_2 \gamma}$, $K_3 := \sigma_{i,3} := (g^b)^{-z_1}$, $K_4 := \sigma_{i,4} := (g^x)^{a_1} g^{a_1 \gamma'} v_2^r g^{z_2} = (v_2^r g^{z_2}) \cdot g^{a_1 \gamma}$, $K_5 := \sigma_{i,5} := (g^b)^{-z_2}$, $K_6 := \sigma_{i,6} := g^{r_2 b}$, $K_7 := \sigma_{i,7} := g^{r_1}$, $\vartheta_i := H_k(M_i \parallel \Sigma_{i,2})$ where $\Sigma_{i,2} := (\sigma_{i,3}, \ldots, \sigma_{i,7})$, $m_i := H_k(g^{\vartheta_i} \bar{h}^{\varphi_i})$, $K_0 := \sigma_{i,0} := (u^{m_i} w^{\mathsf{stag}} h)^{r_1}$, $K_1 := \sigma_{i,1} \cdot \zeta^{\rho_1}$, $K_1' := g^{\rho_1}$, $\hat{K}_1 := (g^b)^{\rho_1}$, $K_2 := \sigma_{i,2} \cdot \zeta^{\rho_2}$, $K_2' := g^{\rho_2}$, $\hat{K}_2 := (g^{ba_1})^{\rho_2}$, stores $(M_i, \sigma_i, R_i := (r_1, r_2, z_1, z_2, \mathsf{stag}, \gamma_i := \gamma'))$ in QList where $\sigma_i := (\sigma_{i,0}, \ldots, \sigma_{i,7}, \mathsf{stag}, \varphi_i)$ and outputs $\omega := (K_0, \ldots, K_7, K_1', K_2', \hat{K}_1, \hat{K}_2, \mathsf{stag}, \varphi_i)$ for M_i. We can verify σ_i is a correct type B signature.

Embedding Instance: If $i = j$, then $\mathcal{E}$ chooses $r_1^*, r_2^*, z_1^*, z_2^*, \gamma^*, \mathsf{stag}^*, \varphi^*$, $\rho_1^*, \rho_2^* \xleftarrow{\cup} \mathbb{Z}_p$, sets $r^* = r_1^* + r_2^*$, answers $K_1^* := (g^{xy+\beta\delta})^{a_1} \cdot v^{r^*} \cdot (g^y)^{-a_1\gamma^*} \zeta^{\rho_1^*} = (g^{\alpha a_1} v^{r^*}) \cdot g^{-a_1 a_2 \gamma^*} \zeta^{\rho_1'}$ (where $a_2 = y, xy = \alpha, \rho_1' := a_1\delta + \rho_1^*$), $K_1^{*\prime} := (g^\delta)^{a_1} g^{\rho_1^*} = g^{\rho_1'}$, $\hat{K}_1^* := (g^\delta)^{ba_1} g^{b\rho_1^*} = (g^b)^{\rho_1'}$, $K_2^* := (g^{xy+\beta\delta})^{-1} v_1^{r^*} g^{z_1^*} \cdot (g^{a_2})^{\gamma^*} (g^\beta)^{\rho_2^*} = (g^{-\alpha} v_1^{r^*} g^{z_1^*}) \cdot g^{a_2\gamma^*} \zeta^{\rho_2'}$ (where $\rho_2' := -\delta + \rho_2^*$), $K_2^{*\prime} := (g^\delta)^{-1} g^{\rho_2^*} = g^{\rho_2'}$, $\hat{K}_2^* := (g^\delta)^{-ba_1} g^{ba_1\rho_2^*} = (g^{ba_1})^{\rho_2'}$, $K_3^* := (g^b)^{-z_1^*}$, $K_4^* := v_2^{r^*} g^{z_2^*} g^{a_1\gamma^*}$, $K_5^* := (g^b)^{-z_2^*}$, $K_6^* := g^{r_2^* b}$, $K_7^* := g^{r_1^*}$, $\vartheta^* := H_k(K_3^*, \ldots, K_7^*, \mathsf{stag}^*)$, $m^* := H_k(g^{\vartheta^*} \bar{h}^{\varphi^*})$, and $K_0^* := (u^{m^*} w^{\mathsf{stag}^*} h)^{r_1^*}$ and records $(M_j, \omega^* := (K_0^*, \ldots, \varphi^*), j, \gamma^*)$ as the challenge instance. It can be verified ω^* is a correct encryption of type B signature.

Simulation of Adjudication Oracle: When $\mathcal{A}$ makes ℓ-th adjudication query (M_ℓ, ω_ℓ), then we know that $\mathcal{A}$ must have queried M_ℓ to $\mathcal{CO}$ by the theorem of Rückert and Schröder (Otherwise, it is a forgery. This is the same argument by Rückert and Schröder in [27]). First, $\mathcal{E}$ verifies the query and returns $\perp$ if it is invalid. Otherwise, $\mathcal{E}$ acts as follows: If $M_\ell = M_j$, that is, the guessed index $((M_j, \ldots) \notin \mathsf{QList})$, then $\mathcal{E}$ aborts. Otherwise, there exists $(M_i, \sigma_i, R_i) \in \mathsf{QList}$ for some $i \neq j$ such that $M_\ell = M_i$ and the signature is Type B. In this case $(M_\ell = M_i)$, for query $(M_\ell, \omega = (K_0, \ldots, K_7, K_1', K_2', \hat{K}_1, \hat{K}_2, \mathsf{stag}, \varphi))$, if $\varphi \neq \varphi_i$, then $\mathcal{A}$ breaks strong unforgeability of our modified Waters dual signature. We consider an intermediate game where if $\varphi \neq \varphi_i$, then $\mathcal{E}$ aborts. The probability $\mathcal{E}$ aborts with this condition is less than the success probability of breaking strong unforgeability of sWdSig. That is, it holds $\varphi = \varphi_i$ without negligible probability. If $\varphi = \varphi_i$, then it holds $K_3 = \sigma_{i,3}$, $K_4 = \sigma_{i,4}$, $K_5 = \sigma_{i,5}$, $K_6 = \sigma_{i,6}$, $K_7 = \sigma_{i,7}$, $\mathsf{stag} = \mathsf{stag}_i$ since otherwise it means $\mathcal{A}$ outputs $(K_3, \ldots, K_7, \mathsf{stag})$ such that $H_k(\sigma_{i,3}, \ldots, \sigma_{i,7}, \mathsf{stag}_i) = \varphi_i = \varphi = H_k(K_3, \ldots, K_7, \mathsf{stag})$ and $(K_3, \ldots, K_7, \mathsf{stag}) \neq (\sigma_{i,3}, \ldots, \sigma_{i,7}, \mathsf{stag}_i)$. This is a collision of the hash function and contradicts to the collision-resistant property. We consider an intermediate game where if $H_k(\sigma_{i,3}, \ldots, \sigma_{i,7}, \mathsf{stag}_i) = \varphi_i = \varphi = H_k(K_3, \ldots, K_7, \mathsf{stag})$ and $(K_3, \ldots, K_7, \mathsf{stag}) \neq (\sigma_{i,3}, \ldots, \sigma_{i,7}, \mathsf{stag}_i)$, then $\mathcal{E}$ aborts. The probability $\mathcal{E}$ aborts with this condition is less than the success probability of breaking the CRHF. That is, randomness of $(K_3, \ldots, K_7, \mathsf{stag})$ is the same as that of $(\sigma_{i,3}, \ldots, \sigma_{i,7}, \mathsf{stag}_i)$ without negligible probability.

By using $K_3 = g^{-bz_1}$, $K_5 = g^{-bz_2}$, $K_6 = g^{br_2}$, $K_7 = g^{r_1}$, $\mathcal{E}$ can compute $g^{r_2} = (K_6)^{1/b}$, $g^{r_1} = K_7$, $g^{z_1} = (K_3)^{-1/b}$, $g^{z_2} = (K_5)^{-1/b}$, $v^r = (g^{r_1} \cdot g^{r_2})^{y_v}$, $v_1^r = (g^{r_1} \cdot g^{r_2})^{y_{v_1}}$, and $v_2^r = (g^{r_1} \cdot g^{r_2})^{y_{v_2}}$ since $\mathcal{E}$ has b, y_v, y_{v_1}, y_{v_2} and it holds that $v = g^{y_v}$ $v_1 = g^{y_{v_1}}$ $v_2 = g^{y_{v_2}}$. $\mathcal{E}$ can use the same computation procedure in the simulation of $\mathcal{CO}$ above by using γ_i stored in QList. Therefore, $\mathcal{E}$ can return valid Type B signature $(\sigma_0, \ldots, \sigma_7, \mathsf{stag}, \varphi)$ such that the randomness r in σ_1, σ_2 is the same as that in K_1, K_2 by using stored information σ_i. That is, $\mathcal{AO}$ is perfectly simulated by $\mathcal{E}$.

Solving the Problem: At some point, $\mathcal{A}$ outputs a Type B extraction, $M^* = M_j$, stag^*, $\sigma_1^* = g^{\alpha a_1} v^{r^*} g^{-a_1 a_2 \gamma^*}$, $\sigma_2^* = g^{-\alpha} v_1^{r^*} g^{z_1^*} g^{a_2\gamma^*}$, $\sigma_3^* = (g^b)^{-z_1^*}$, $\sigma_4^* = v_2^{r^*} g^{z_2^*} g^{a_1\gamma^*}$, $\sigma_5^* = (g^b)^{-z_2^*}$, $\sigma_6^* = g^{r_2^* b}$, $\sigma_7^* = g^{r_1^*}$, $\vartheta^* = H_k(\sigma_3, \ldots, \sigma_7, \mathsf{stag}^*)$, $m^* = H_k(g^{\vartheta^*} \bar{h}^\varphi)$, and $\sigma_0^* = (u^{m^*} w^{\mathsf{stag}^*} h)^{r_1^*}$ (not queried to $\mathcal{AO}$ but $\mathcal{CO}$) such that *randomness are the same as those used when $\mathcal{B}$ embedded the problem instance at j-th query* and $(M_j, \omega^*, j, \gamma^*)$ is recorded as the challenge instance.

This is guaranteed by the strong unforgeability and collision-resistance as we discussed above. By using these values, $\mathcal{E}$ can compute $g^{r_2^*} = (\sigma_6^*)^{1/b}$, $g^{r_1^*} = \sigma_7^*$, $g^{z_1^*} = (\sigma_3^*)^{-1/b}$, $g^{z_2^*} = (\sigma_5^*)^{-1/b}$, $v^{r^*} = (g^{r_1^*} \cdot g^{r_2^*})^{y_v}$, $v_1^{r^*} = (g^{r_1^*} \cdot g^{r_2^*})^{y_{v_1}}$, $v_2^{r^*} = (g^{r_1^*} \cdot g^{r_2^*})^{y_{v_2}}$, since $\mathcal{E}$ has b, y_v, y_{v_1}, y_{v_2} and it holds that $v = g^{y_v}$ $v_1 = g^{y_{v_1}}$ $v_2 = g^{y_{v_2}}$. Thus, $\mathcal{E}$ can compute $g^{z_1^*} \cdot v_1^{r^*} g^{a_2\gamma^*}/\sigma_2^* = g^{\alpha} = g^{xy}$ since $\mathcal{E}$ has a_1 and γ^* is recorded. That is, $\mathcal{E}$ can output solution g^{xy} of the AgExt problem if the adversary outputs a Type A extraction for queried message M_j to $\mathcal{CO}$. $\mathcal{E}$ guesses index j, so its success probability is degraded by a factor of $1/q_{\mathsf{C}}$. However, it still breaks the AgExt problem with non-negligible probability ϵ/q_{C} where ϵ is the success probability of $\mathcal{A}$. ■

5 Application to Obfuscators for ES and EVES

Our VES scheme can be used to construct new obfuscators for ES and EVES. Hada constructed an obfuscator for ES by combining Waters's signature (2005) and the linear encryption scheme [23]. The linear encryption scheme proposed by Boneh, Boyen, and Shacham [7] and is as follows:

$\mathsf{L.Gen}(1^\lambda)$**:** It generates $\Gamma \xleftarrow{\mathsf{R}} \mathcal{G}_{\mathsf{bmp}}(1^\lambda)$, selects exponents $x_e, y_e \xleftarrow{\mathsf{U}} \mathbb{Z}_p$, and outputs $pk := (f_e, h_e) := (g^{x_e}, g^{y_e})$, $dk := (x_e, y_e)$.

$\mathsf{L.Enc}(pk_e, m)$**:** On input $m \in \mathbb{G}$ and $pk = (f_e, h_e)$ it selects $r, s \xleftarrow{\mathsf{U}} \mathbb{Z}_p$ and outputs $c := (f_e^r, h_e^s, g^{r+s}m)$.

Hada's idea is as follows: Suppose that signature σ is computed as $\sigma = sk \cdot G(m)$ where $sk \in \mathbb{G}$ is the signing key, $m \in \mathbb{Z}_p$ is the message and $G : \mathbb{Z}_p \to \mathbb{G}$ is an efficiently computable function. Then, for ciphertext $c = \mathsf{L.Enc}(pk, sk)$, we can compute $\tilde{c} := c \cdot G(m) = \mathsf{L.Enc}(pk, sk \cdot G(m))$ by homomorphic property of the linear encryption scheme. This is exactly an encrypted signature. The ciphertext of sk can be seen as an obfuscated circuit for encrypted signatures since the linear encryption scheme is semantically secure and no information about sk is revealed. We extend Hada's construction, that is, we combine our VES scheme based on the strongly unforgeable Waters dual signature and the linear encryption scheme. However, our VES scheme is based on the Waters dual signature, which is more complex than Waters' signature at Eurocrypt'05, so it is non-trivial whether we can use Hada's technique directly or not. Especially, in Waters' signature at Eurocrypt'05, the signing algorithm does not exponentiate sk, but in the Waters dual signature, it does. We can resolve this problem by using the multiplicatively homomorphic property of the linear encryption scheme, that is, we can compute $c^r \cdot G(m) = \mathsf{L.Enc}(pk, sk^r \cdot G(m))$. Therefore, if we encrypt $sk = (g^\alpha, g^{\alpha a_1}, g^{a_1 a_2})$ by linear encryption, then we can construct an obfuscator for ES/EVES. We omit details of these constructions since we do not have space to present them. We will present them in a full version.

References

1. Abe, M., Chase, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Constant-Size Structure-Preserving Signatures: Generic Constructions and Simple Assumptions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 4–24. Springer, Heidelberg (2012), full version available from `http://eprint.iacr.org/2012/285`
2. Asokan, N., Shoup, V., Waidner, M.: Optimistic Fair Exchange of Digital Signatures (Extended Abstract). In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)
3. Bao, F., Deng, R.H., Mao, W.: Efficient and practical fair exchange protocols with off-line TTP. In: IEEE Symposium on Security and Privacy, pp. 77–85. IEEE Computer Society (1998)
4. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. J. ACM 59(2), 6 (2012)
5. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable Proofs and Delegatable Anonymous Credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (2009)
6. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. J. Cryptology 21(2), 149–177 (2008)
7. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
8. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
9. 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)
10. Boneh, D., Shen, E., Waters, B.: Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 229–240. Springer, Heidelberg (2006)
11. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: ITCS, pp. 309–325. ACM (2012)
12. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS, pp. 97–106. IEEE (2011)
13. Cheng, R., Zhang, B., Zhang, F.: Secure Obfuscation of Encrypted Verifiable Encrypted Signatures. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 188–203. Springer, Heidelberg (2011)
14. Coron, J.-S., Mandal, A., Naccache, D., Tibouchi, M.: Fully Homomorphic Encryption over the Integers with Shorter Public Keys. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 487–504. Springer, Heidelberg (2011)
15. Coron, J.-S., Naccache, D.: Boneh *et al.*'s k-Element Aggregate Extraction Assumption Is Equivalent to the Diffie-Hellman Assumption. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 392–397. Springer, Heidelberg (2003)
16. Fuchsbauer, G.: Commuting Signatures and Verifiable Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 224–245. Springer, Heidelberg (2011)
17. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178. ACM (2009)

18. Gentry, C., Halevi, S.: Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In: FOCS, pp. 107–109. IEEE (2011)
19. 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)
20. Gentry, C., Halevi, S., Smart, N.P.: Fully Homomorphic Encryption with Polylog Overhead. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 465–482. Springer, Heidelberg (2012)
21. Gerbush, M., Lewko, A., O'Neill, A., Waters, B.: Dual Form Signatures: An Approach for Proving Security from Static Assumptions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 25–42. Springer, Heidelberg (2012)
22. Hada, S.: Zero-Knowledge and Code Obfuscation. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 443–457. Springer, Heidelberg (2000)
23. Hada, S.: Secure Obfuscation for Encrypted Signatures. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 92–112. Springer, Heidelberg (2010)
24. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential Aggregate Signatures and Multisignatures Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006)
25. Rückert, M.: Verifiably Encrypted Signatures from RSA without NIZKs. In: Roy, B., Sendrier, N. (eds.) INDOCRYPT 2009. LNCS, vol. 5922, pp. 363–377. Springer, Heidelberg (2009)
26. Rückert, M., Schneider, M., Schröoder, D.: Generic Constructions for Verifiably Encrypted Signatures without Random Oracles or NIZKs. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 69–86. Springer, Heidelberg (2010)
27. Rückert, M., Schröder, D.: Security of Verifiably Encrypted Signatures and a Construction without Random Oracles. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 17–34. Springer, Heidelberg (2009)
28. Shoup, V.: Lower Bounds for Discrete Logarithms and Related Problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997)
29. 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)
30. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
31. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009), full version available from `http://eprint.iacr.org/2009/385`
32. Zhang, F., Safavi-Naini, R., Susilo, W.: Efficient Verifiably Encrypted Signature and Partially Blind Signature from Bilinear Pairings. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT 2003. LNCS, vol. 2904, pp. 191–204. Springer, Heidelberg (2003)

Sequential Aggregate Signatures with Short Public Keys: Design, Analysis and Implementation Studies

Kwangsu Lee[1,⋆], Dong Hoon Lee[2,⋆⋆], and Moti Yung[3]

[1] Columbia University, NY, USA
kwangsu@cs.columbia.edu
[2] Korea University, Seoul, Korea
donghlee@korea.ac.kr
[3] Google Inc. and Columbia University, NY, USA
moti@cs.columbia.edu

Abstract. The notion of aggregate signature has been motivated by applications and it enables any user to compress different signatures signed by different signers on different messages into a short signature. Sequential aggregate signature, in turn, is a special kind of aggregate signature that only allows a signer to add his signature into an aggregate signature in sequential order. This latter scheme has applications in diversified settings, such as in reducing bandwidth of a certificate chains, and in secure routing protocols. Lu, Ostrovsky, Sahai, Shacham, and Waters presented the first sequential aggregate signature scheme in the standard (non idealized ROM) model. The size of their public key, however, is quite large (i.e., the number of group elements is proportional to the security parameter), and therefore they suggested as an open problem the construction of such a scheme with short keys. Schröder recently proposed a sequential aggregate signature (SAS) with short public keys using the Camenisch-Lysyanskaya signature scheme, but the security is only proven under an interactive assumption (which is considered a relaxed notion of security). In this paper, we propose the first sequential aggregate signature scheme with short public keys (i.e., a constant number of group elements) in prime order (asymmetric) bilinear groups which is secure under static assumptions in the standard model. Technically, we start with a public key signature scheme based on the recent dual system encryption technique of Lewko and Waters. This technique cannot give directly an aggregate signature scheme since, as we observed, additional elements should be published in the public key to support aggregation. Thus, our construction is a careful augmentation technique for the dual system technique to allow it to support a sequential aggregate signature scheme. We further implemented our scheme and conducted a performance study and implementation optimization.

⋆ Supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2012-H0301-12-3007) supervised by the NIPA (National IT Industry Promotion Agency).
⋆⋆ Supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2010-0029121).

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 423–442, 2013.

1 Introduction

Aggregate signature is a relatively new type of public key signature which enables any user to combine n signatures signed by different n signers on different n messages into a short signature. The concept of public key aggregate signature (PKAS) was introduced by Boneh, Gentry, Lynn, and Shacham [9], and they proposed an efficient PKAS scheme in the random oracle model using the bilinear groups. After that, numerous PKAS schemes were proposed using bilinear groups [14,22,6,7,1,15] or using trapdoor permutations [24,3,25].

One application of aggregate signature is the certificate chains of the public key infrastructure (PKI) [9]. The PKI system has a tree structure, and a certificate for a user consists of a certificate chain from a root node to a leaf node, each node in the chain signing its predecessor. If the signatures in the certificate chain are replaced with a single aggregate signature, then the bandwidth for signatures transfer can be significantly saved. Another application is to the secure routing protocol of the internet protocol [9]. If each router which participates in the routing protocol uses PKAS instead of a public key signature (PKS), then the communication overload of signature transfer can be dramatically reduced. Further, aggregate signatures have other applications such as reducing bandwidth in sensor networks or ad-hoc networks, and in software authentication in the presence of software update [1].

1.1 Previous Methods

Aggregate signature schemes are categorized as *full* aggregate signature, *synchronized* aggregate signature, and *sequential* aggregate signature depending on the type of signature aggregation. They have also been applied to regular signatures in the PKI model, and to ID-based signatures (with trusted key server).

The first type of aggregate signature is *full aggregate signature* which enables any user to freely aggregate different signatures of different signers. This full aggregate signature is the most flexible aggregate signature since it does not require any restriction on the aggregation step (though, restriction may be needed at times for certain applications). However, there is only one full aggregate signature scheme that was proposed by Boneh et al. [9]. Since this scheme is based on the short signature scheme of Boneh et al. [10], the signature length it provides is also very short. However, the security of the scheme is just proven in the idealized random oracle model and the number of pairing operations in the aggregate signature verification algorithm is proportional to the number of signers in the aggregate signature.

The second type of aggregate signature is *synchronized aggregate signature* which enables any user to combine different signatures with the same synchronizing information into a single signature. The synchronized aggregate signature has a demerit which dictates that all signers should share the same synchronizing information (like a time clock or other shared value). Gentry and Ramzan introduced the concept of synchronized aggregate signature, they proposed an identity-based synchronized aggregate signature scheme using bilinear groups,

and they proved its security in the random oracle model [14]. We note that identity-based aggregate signature (IBAS) is an ID-based scheme and thus relies on a trusted server knowing all private keys (i.e., its trust structure is different than in regular PKI). However, it also has a notable advantage such that it is not required to retrieve the public keys of signers in the verification algorithm since an identity string plays the role of a public key (the lack of public key is indicated in our comparison table as public key of no size!). Recently, Ahn et al. presented a public key synchronized aggregate signature scheme without relying on random oracles [1].

The third type of aggregate signature is *sequential aggregate signature* (SAS) that enables each signer to aggregate his signature to a previously aggregated signature in a sequential order. The sequential aggregate signature has the obvious limitation of signers being ordered to aggregate their signatures in contrast to the full aggregate signature and the synchronized aggregate signature. However, it has an advantage such that it is not required to share synchronized information among signers in contrast to the synchronized aggregate signature, and many natural applications lead themselves to this setting. The concept of sequential aggregate signature was introduced by Lysyanskaya et al., and they proposed a public key sequential aggregate signature scheme using the certified trapdoor permutations in the random oracle model [24]. Boldyreva et al. presented an identity-based sequential aggregate signature scheme in the random oracle model using an interactive assumption [6], but it was shown that their construction is not secure by Hwang et al. [17]. After that, Boldyreva et al. proposed a new identity-based sequential aggregate signature by modifying their previous construction and proved its security in the generic group model [7]. Recently, Gerbush et al. showed that the modified IBAS scheme of Boldyreva et al. is secure under static assumptions using the dual form signatures framework [15]. The first sequential aggregate signature scheme without the random oracle idealization was proposed by Lu et al. [22]. They converted the PKS scheme of Waters [28] to the PKAS scheme, and proved its security under the well known CDH assumption. However, the scheme of Lu et al. has a demerit since the number of group elements in the public key is proportional to the security parameter (for a security of 2^{80} they need 160 elements or about 80 elements in a larger group); they left as an open question to design a scheme with shorter public key. Schröder proposed a PKAS scheme with short public keys relying on the Camenisch-Lysyanskaya signature scheme [27], however the scheme's security is proven under an interactive assumption (which typically, is a relaxation used when designs based on static assumptions are hard to find).[1] Therefore, the construction of sequential aggregate signature scheme with short public keys without relaxations like random oracles or an interactive assumptions was left as an open question.

[1] Gerbush et al. showed that a modified Camenisch-Lysyanskaya signature scheme in composite order groups is secure under static assumptions [15]. However, it is unclear whether the construction of Schröder can be directly applied to this modified Camenisch-Lysyanskaya signature scheme.

Table 1. Comparison of aggregate signature schemes

Scheme	Type	ROM	PK Size	AS Size	Sign Time	Verify Time	Assumption
BGLS [9]	Full	Yes	$1k_p$	$1k_p$	1E	lP	CDH
GR [14]	IB, Sync	Yes	–	$2k_p + \lambda$	3E	3P + lE	CDH
AGH [1]	Sync	Yes	$1k_p$	$2k_p + 32$	6E	4P + lE	CDH
AGH [1]	Sync	No	$1k_p$	$2k_p + 32$	10E	8P + lE	CDH
LMRS [24]	Seq	Yes	$1k_f$	$1k_f$	lE	lE	cert TDP
Neven [25]	Seq	Yes	$1k_f$	$1k_f + 2\lambda$	1E + 2lM	2lM	uncert CFP
BGOY [7]	IB, Seq	Yes	–	$3k_p$	4P + lE	4P + lE	Interactive
GLOW [15]	IB, Seq	Yes	–	$5k_f$	10P + 2lE	10P + 2lE	Static
LOSSW [22]	Seq	No	$2\lambda k_p$	$2k_p$	2P + 4λlM	2P + 2λlM	CDH
Schröder [27]	Seq	No	$2k_p$	$4k_p$	lP + 2lE	lP + lE	Interactive
Ours	Seq	No	$11k_p$	$8k_p$	8P + 5lE	8P + 4lE	Static

ROM = random oracle model, IB = identity based, λ = security parameter
k_p, k_f = the bit size of element for pairing and factoring, l = the number of signers
P = pairing computation, E = exponentiation, M = multiplication

1.2 Our Contributions

Challenged by the above question, the motivation of our research is to construct an efficient sequential aggregate signature scheme secure in the standard model (i.e., without employing assumptions like random oracle or interactive assumptions as part of the proof) with short public keys (e.g., constant number of group elements). To achieve this goal, we use the public key signature scheme derived from the identity-based encryption (IBE) scheme that adopts the innovative dual system encryption techniques of Waters [29,21]. That is, an IBE scheme is first converted to a PKS scheme by the clever observation of Naor [8]. The PKS schemes that adopt the dual system encryption techniques are the scheme of Waters [29] which includes a random tag in a signature and the scheme of Lewko and Waters [21] which does not include a random tag in a signature. The scheme of Waters is not appropriate to aggregate signature since the random tags in signatures cannot be compressed into a single value. The scheme of Lewko and Waters in composite order groups is easily converted to an aggregate signature scheme if the element of $\mathbb{G}_{p_3}$ is moved from a private key to a public key, but it is inefficient because of composite order groups.[2] Therefore, we start the construction from the IBE scheme in prime order (asymmetric) bilinear groups of Lewko and Waters. However, this PKS scheme which is directly derived from the IBE scheme of Lewko and Waters is not easily converted to a sequential aggregate

[2] Lewko obtained a prime order IBE scheme by translating the Lewko-Waters composite order IBE scheme using the dual pairing vector spaces [20]. One may consider to construct an aggregate signature scheme using this IBE scheme. However, it is not easy to aggregate individual signatures since the dual orthonormal basis vectors of each users are randomly generated.

signature scheme (as far as we see). The reason is that we need a PKS scheme that supports multi-user setting and public re-randomization to construct a SAS scheme by using the randomness reuse technique of Lu et al. [22], but this PKS scheme does not support these two properties.

Here we first construct a PKS scheme in prime order (asymmetric) bilinear groups which supports multi-user seting and public re-randomization by modifying the PKS scheme of Lewko and Waters, and we prove its security using the dual system encryption technique. Next, we convert the modified PKS scheme to a SAS scheme with short public keys by using the randomness reuse technique of Lu et al. [22], and we prove its security without random oracles and based on the traditional static assumptions. Our security proof crucially relies on the fact that we add additional randomization elements to the SAS verification algorithm, so that we can expand these elements to a semi-functional space; this allows us to introduce in the SAS scheme public-key elements used in aggregation. Note that Table 1 gives a comparison of past schemes to ours. Finally, to support our claim of efficiency, we implemented our SAS scheme using the PBC library and we measured the performance of the scheme. Additionally, as part of the implementation we provide a computational preprocessing method which improves the amortized performance of our scheme.

1.3 Additional Related Work

There are some work on aggregate signature schemes which allow signers to communicate with each other or schemes which compress only partial elements of a signature in the aggregate algorithm [4,2,16,11]. Generally, communication resources of computer systems are very expensive compared to the computation resources. Thus, it is preferred to perform several expensive computational operations instead of a single communication exchange. Additionally, a signature scheme with added communications does not correspond to a pure public key signature schemes, but corresponds more to a multi-party protocol. In addition, signature schemes which compress just partial elements of signatures cannot be an aggregate signature since the total size of signatures is still proportional to the number of signers.

Another research area related to aggregate signature is multi-signature [5,22]. Multi-signature is a special type of aggregate signature in which all signers generate signatures on the same message, and then any user can combine these signature to a single signature. Aggregate message authentication code (AMAC) is the symmetric key analogue of aggregate signature: Katz and Lindell introduced the concept of AMAC and showed that it is possible to construct AMAC schemes based on any message authentication code schemes [18].

2 Preliminaries

We first define public key signature and sequential aggregate signature, and then give the definition of their correctness and security.

2.1 Public Key Signature

A public key signature (PKS) scheme consists of three PPT algorithms **KeyGen**, **Sign**, and **Verify**, which are defined as follows: The key generation algorithm $\mathbf{KeyGen}(1^\lambda)$ takes as input the security parameters 1^λ, and outputs a public key PK and a private key SK. The signing algorithm $\mathbf{Sign}(M, SK)$ takes as input a message M and a private key SK, and outputs a signature σ. The verification algorithm $\mathbf{Verify}(\sigma, M, PK)$ takes as input a signature σ, a message M, and a public key PK, and outputs either 1 or 0 depending on the validity of the signature. The correctness requirement is that for any (PK, SK) output by **KeyGen** and any $M \in \mathcal{M}$, we have that $\mathbf{Verify}(\mathbf{Sign}(M, SK), M, PK) = 1$. We can relax this notion to require that the verification is correct with overwhelming probability over all the randomness of the experiment.

The security notion of existential unforgeability under a chosen message attack is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$: $\mathcal{C}$ first generates a key pair (PK, SK) by running **KeyGen**, and gives PK to $\mathcal{A}$. Then $\mathcal{A}$, adaptively and polynomially many times, requests a signature query on a message M under the challenge public key PK, and receives a signature σ. Finally, $\mathcal{A}$ outputs a forged signature σ^* on a message M^*. $\mathcal{C}$ then outputs 1 if the forged signature satisfies the following two conditions, or outputs 0 otherwise: 1) $\mathbf{Verify}(\sigma^*, M^*, PK) = 1$ and 2) M^* was not queried by $\mathcal{A}$ to the signing oracle. The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{PKS} = \Pr[\mathcal{C} = 1]$ where the probability is taken over all the randomness of the experiment. A PKS scheme is existentially unforgeable under a chosen message attack if all PPT adversaries have at most a negligible advantage in the above experiment (for large enough security parameter).

2.2 Sequential Aggregate Signature

A sequential aggregate signature (SAS) scheme consists of four PPT algorithms **Setup**, **KeyGen**, **AggSign**, and **AggVerify**, which are defined as follows: The setup algorithm $\mathbf{Setup}(1^\lambda)$ takes as input a security parameter 1^λ and outputs public parameters PP. The key generation algorithm $\mathbf{KeyGen}(PP)$ takes as input the public parameters PP, and outputs a public key PK and a private key SK. The aggregate signing algorithm $\mathbf{AggSign}(AS', \mathbf{M}, \mathbf{PK}, M, SK)$ takes as input an aggregate-so-far AS' on messages $\mathbf{M} = (M_1, \ldots, M_l)$ under public keys $\mathbf{PK} = (PK_1, \ldots, PK_l)$, a message M, and a private key SK, and outputs a new aggregate signature AS. The aggregate verification algorithm $\mathbf{AggVerify}(AS, \mathbf{M}, \mathbf{PK})$ takes as input an aggregate signature AS on messages $\mathbf{M} = (M_1, \ldots, M_l)$ under public keys $\mathbf{PK} = (PK_1, \ldots, PK_l)$, and outputs either 1 or 0 depending on the validity of the sequential aggregate signature. The correctness requirement is that for each PP output by **Setup**, for all (PK, SK) output by **KeyGen**, any M, we have that $\mathbf{AggVerify}(\mathbf{AggSign}(AS', \mathbf{M}', \mathbf{PK}', M, SK), \mathbf{M}'||M, \mathbf{PK}'||PK) = 1$ where AS' is a valid aggregate-so-far signature on messages $\mathbf{M}'$ under public keys $\mathbf{PK}'$.

The security notion of existential unforgeability under a chosen message attack is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$:

Setup: $\mathcal{C}$ first initializes a certification list CL as empty. Next, it runs **Setup** to obtain public parameters PP and **KeyGen** to obtain a key pair (PK, SK), and gives PK to $\mathcal{A}$.

Certification Query: $\mathcal{A}$ adaptively requests the certification of a public key by providing a key pair (PK, SK). Then $\mathcal{C}$ adds the key pair (PK, SK) to CL if the key pair is a valid one.

Signature Query: $\mathcal{A}$ adaptively requests a sequential aggregate signature (by providing an aggregate-so-far AS' on messages $\mathbf{M}'$ under public keys $\mathbf{PK}'$), on a message M to sign under the challenge public key PK, and receives a sequential aggregate signature AS.

Output: Finally (after a sequence of the above queries), $\mathcal{A}$ outputs a forged sequential aggregate signature AS^* on messages $\mathbf{M}^*$ under public keys $\mathbf{PK}^*$. $\mathcal{C}$ outputs 1 if the forged signature satisfies the following three conditions, or outputs 0 otherwise: 1) $\mathbf{AggVerify}(AS^*, \mathbf{M}^*, \mathbf{PK}^*) = 1$, 2) The challenge public key PK must exists in $\mathbf{PK}^*$ and each public key in $\mathbf{PK}^*$ except the challenge public key must be in CL, and 3) The corresponding message M in $\mathbf{M}^*$ of the challenge public key PK must not have been queried by $\mathcal{A}$ to the sequential aggregate signing oracle.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{SAS} = \Pr[\mathcal{C} = 1]$ where the probability is taken over all the randomness of the experiment. A SAS scheme is existentially unforgeable under a chosen message attack if all PPT adversaries have at most a negligible advantage (for large enough security parameter) in the above experiment.

2.3 Asymmetric Bilinear Groups

Let $\mathbb{G}, \hat{\mathbb{G}}$ and $\mathbb{G}_T$ be multiplicative cyclic groups of prime order p. Let $g, \hat{g}$ be generators of $\mathbb{G}, \hat{\mathbb{G}}$. The bilinear map $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ has the following properties:

1. Bilinearity: $\forall u \in \mathbb{G}, \forall \hat{v} \in \hat{\mathbb{G}}$ and $\forall a, b \in \mathbb{Z}_p$, $e(u^a, \hat{v}^b) = e(u, \hat{v})^{ab}$.
2. Non-degeneracy: $\exists g, \hat{g}$ such that $e(g, \hat{g})$ has order p, that is, $e(g, \hat{g})$ is a generator of $\mathbb{G}_T$.

We say that $\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T$ are bilinear groups with no efficiently computable isomorphisms if the group operations in $\mathbb{G}, \hat{\mathbb{G}}$, and $\mathbb{G}_T$ as well as the bilinear map e are all efficiently computable, but there are no efficiently computable isomorphisms between $\mathbb{G}$ and $\hat{\mathbb{G}}$.

2.4 Complexity Assumptions

We employ three static assumptions in prime order bilinear groups. Assumptions 1 and 3 have been used extensively, while Assumption 2 was introduced by Lewko and Waters [21].

Assumption 1 (Symmetric eXternal Diffie-Hellman). Let $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$ be a description of the asymmetric bilinear group of prime order p. Let $g, \hat{g}$ be generators of $\mathbb{G}, \hat{\mathbb{G}}$ respectively. The assumption is that if the challenge values

$$D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e), g, \hat{g}, \hat{g}^a, \hat{g}^b) \text{ and } T,$$

are given, no PPT algorithm $\mathcal{B}$ can distinguish $T = T_0 = \hat{g}^{ab}$ from $T = T_1 = \hat{g}^c$ with more than a negligible advantage. The advantage of $\mathcal{B}$ is defined as $\mathbf{Adv}^{A1}_{\mathcal{B}}(\lambda) = \left| \Pr[\mathcal{B}(D, T_0) = 0] - \Pr[\mathcal{B}(D, T_1) = 0] \right|$ where the probability is taken over the random choice of $a, b, c \in \mathbb{Z}_p$.

Assumption 2 (LW2). Let $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$ be a description of the asymmetric bilinear group of prime order p. Let $g, \hat{g}$ be generators of $\mathbb{G}, \hat{\mathbb{G}}$ respectively. The assumption is that if the challenge values

$$D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e), g, g^a, g^b, g^c, \hat{g}, \hat{g}^a, \hat{g}^{a^2}, \hat{g}^{bx}, \hat{g}^{abx}, \hat{g}^{a^2 x}) \text{ and } T,$$

are given, no PPT algorithm $\mathcal{B}$ can distinguish $T = T_0 = g^{bc}$ from $T = T_1 = g^d$ with more than a negligible advantage. The advantage of $\mathcal{B}$ is defined as $\mathbf{Adv}^{A2}_{\mathcal{B}}(\lambda) = \left| \Pr[\mathcal{B}(D, T_0) = 0] - \Pr[\mathcal{B}(D, T_1) = 0] \right|$ where the probability is taken over the random choice of $a, b, c, x, d \in \mathbb{Z}_p$.

Assumption 3 (Decisional Bilinear Diffie-Hellman). Let $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$ be a description of the asymmetric bilinear group of prime order p. Let $g, \hat{g}$ be generators of $\mathbb{G}, \hat{\mathbb{G}}$ respectively. The assumption is that if the challenge values

$$D = ((p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e), g, g^a, g^b, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, \hat{g}^c) \text{ and } T,$$

are given, no PPT algorithm $\mathcal{B}$ can distinguish $T = T_0 = e(g, \hat{g})^{abc}$ from $T = T_1 = e(g, \hat{g})^d$ with more than a negligible advantage. The advantage of $\mathcal{B}$ is defined as $\mathbf{Adv}^{A3}_{\mathcal{B}}(\lambda) = \left| \Pr[\mathcal{B}(D, T_0) = 0] - \Pr[\mathcal{B}(D, T_1) = 0] \right|$ where the probability is taken over the random choice of $a, b, c, d \in \mathbb{Z}_p$.

3 Aggregate Signature

We construct a SAS scheme in prime order (asymmetric) bilinear groups and prove its existential unforgeability under a chosen message attack. The main idea is to modify a PKS scheme to support multi-user setting and signature aggregation by using the "randomness reuse" technique of Lu et al. [22]. To support multi-user setting, it is required for all users to share common elements in the public parameters. To use the randomness reuse technique, it is crucial for a signer to publicly re-randomize a sequential aggregate signature to prevent a forgery attack. Thus we need a PKS scheme with short public key that supports "multi-user setting" and "public re-randomization".

Before we present a SAS scheme, we first construct a PKS scheme with short public key that will be augmented to support multi-user setting and public re-randomization. One method to build a PKS scheme is to use the observation of

Naor [8] that private keys of fully secure IBE are easily converted to signatures of PKS. Thus we can convert the prime order IBE scheme of Lewko and Waters [21] to a prime order PKS scheme. However, this directly converted PKS scheme does not support multi-user setting and public re-randomization since it needs to publish additional public key components: Specifically, we need to publish an element g for multi-user setting and elements u, h for public re-randomization. Note that $\hat{g}, \hat{u}, \hat{h}$ are already in the public key, but g, u, h are not. One may try to publish g, u, h in the public key. The technical difficulty arising in this case is that the simulator of the security proof can easily distinguish the changes of the verification algorithm that checks the validity of the forged signature from the normal verification algorithm to the semi-functional one, without using an adversary.

To solve this problem, we devise a method that allows a PKS scheme to safely publish elements g, u, h in the public key for multi-user setting and public re-randomization. The main idea is to additionally randomize the verification components using $\hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi}$ in the verification algorithm. If a valid signature is given in the verification algorithm, then the additionally added randomization elements $\hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi}$ are canceled. Otherwise, the added randomization components prevent the verification of an invalid signature. Therefore, the simulator of the security proof cannot distinguish the changes of the verification algorithm even if g, u, h are published, since the additional elements $\hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi}$ prevent the signature verification.

3.1 Our PKS Scheme

The PKS scheme in prime order bilinear groups is described as follows:

PKS.KeyGen(1^λ): This algorithm first generates the asymmetric bilinear groups $\mathbb{G}, \hat{\mathbb{G}}$ of prime order p of bit size $\Theta(\lambda)$. It chooses random elements $g, w \in \mathbb{G}$ and $\hat{g}, \hat{v} \in \hat{\mathbb{G}}$. Next, it chooses random exponents $\nu_1, \nu_2, \nu_3, \phi_1, \phi_2, \phi_3 \in \mathbb{Z}_p$ and sets $\tau = \phi_1 + \nu_1\phi_2 + \nu_2\phi_3, \pi = \phi_2 + \nu_3\phi_3$. It selects random exponents $\alpha, x, y \in \mathbb{Z}_p$ and sets $u = g^x, h = g^y, \hat{u} = \hat{g}^x, \hat{h} = \hat{g}^y$. It outputs a private key $SK = (\alpha, x, y)$ and a public key PK as

$$g, u, h,\ w_1 = w^{\phi_1}, w_2 = w^{\phi_2}, w_3 = w^{\phi_3}, w,\ \hat{g}, \hat{g}^{\nu_1}, \hat{g}^{\nu_2}, \hat{g}^{-\tau},$$
$$\hat{u}, \hat{u}^{\nu_1}, \hat{u}^{\nu_2}, \hat{u}^{-\tau},\ \hat{h}, \hat{h}^{\nu_1}, \hat{h}^{\nu_2}, \hat{h}^{-\tau},\ \hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi},\ \Omega = e(g, \hat{g})^\alpha.$$

PKS.Sign(M, SK): This algorithm takes as input a message $M \in \{0,1\}^k$ where $k < \lambda$ and a private key $SK = (\alpha, x, y)$. It selects random exponents $r, c_1, c_2 \in \mathbb{Z}_p$ and outputs a signature σ as

$$W_{1,1} = g^\alpha (u^M h)^r w_1^{c_1}, W_{1,2} = w_2^{c_1}, W_{1,3} = w_3^{c_1}, W_{1,4} = w^{c_1},$$
$$W_{2,1} = g^r w_1^{c_2}, W_{2,2} = w_2^{c_2}, W_{2,3} = w_3^{c_2}, W_{2,4} = w^{c_2}.$$

PKS.Verify(σ, M, PK): This algorithm takes as input a signature σ on a message $M \in \{0,1\}^k$ under a public key PK. It first chooses random exponents

$t, s_1, s_2 \in \mathbb{Z}_p$ and computes verification components as

$$V_{1,1} = \hat{g}^t, V_{1,2} = (\hat{g}^{\nu_1})^t \hat{v}^{s_1}, V_{1,3} = (\hat{g}^{\nu_2})^t (\hat{v}^{\nu_3})^{s_1}, V_{1,4} = (\hat{g}^{-\tau})^t (\hat{v}^{-\pi})^{s_1},$$
$$V_{2,1} = (\hat{u}^M \hat{h})^t, V_{2,2} = ((\hat{u}^{\nu_1})^M \hat{h}^{\nu_1})^t \hat{v}^{s_2}, V_{2,3} = ((\hat{u}^{\nu_2})^M \hat{h}^{\nu_2})^t (\hat{v}^{\nu_3})^{s_2},$$
$$V_{2,4} = ((\hat{u}^{-\tau})^M \hat{h}^{-\tau})^t (\hat{v}^{-\pi})^{s_2}.$$

Next, it verifies that $\prod_{i=1}^{4} e(W_{1,i}, V_{1,i}) \cdot \prod_{i=1}^{4} e(W_{2,i}, V_{2,i})^{-1} \stackrel{?}{=} \Omega^t$. If this equation holds, then it outputs 1. Otherwise, it outputs 0.

We first note that the inner product of $(\phi_1, \phi_2, \phi_3, 1)$ and $(1, \nu_1, \nu_2, -\tau)$ is zero since $\tau = \phi_1 + \nu_1\phi_2 + \nu_2\phi_3$, and the inner product of $(\phi_1, \phi_2, \phi_3, 1)$ and $(0, 1, \nu_3, -\pi)$ is zero since $\pi = \phi_2 + \nu_3\phi_3$. Using these facts, the correctness requirement of the above PKS scheme is easily verified as

$$\prod_{i=1}^{4} e(W_{1,i}, V_{1,i}) \cdot \prod_{i=1}^{4} e(W_{2,i}, V_{2,i})^{-1} = e(g^{\alpha}(u^M h)^r, \hat{g}^t) \cdot e(g^r, (\hat{u}^M \hat{h})^t)^{-1} = \Omega^t.$$

Theorem 1. *The above PKS scheme is existentially unforgeable under a chosen message attack if Assumptions 1, 2, and 3 hold. That is, for any PPT adversary $\mathcal{A}$, there exist PPT algorithms $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ such that $\boldsymbol{Adv}_{\mathcal{A}}^{PKS}(\lambda) \leq \boldsymbol{Adv}_{\mathcal{B}_1}^{A1}(\lambda) + q\boldsymbol{Adv}_{\mathcal{B}_2}^{A2}(\lambda) + \boldsymbol{Adv}_{\mathcal{B}_3}^{A3}(\lambda)$ where q is the maximum number of signature queries of $\mathcal{A}$.*

The proof of this theorem is given in Section 4.1.

3.2 Our SAS Scheme

The SAS scheme in prime order bilinear groups is described as follows:

SAS.Setup(1^λ): This algorithm first generates the asymmetric bilinear groups $\mathbb{G}, \hat{\mathbb{G}}$ of prime order p of bit size $\Theta(\lambda)$. It chooses random elements $g, w \in \mathbb{G}$ and $\hat{g}, \hat{v} \in \hat{\mathbb{G}}$. Next, it chooses random exponents $\nu_1, \nu_2, \nu_3, \phi_1, \phi_2, \phi_3 \in \mathbb{Z}_p$ and sets $\tau = \phi_1 + \nu_1\phi_2 + \nu_2\phi_3, \pi = \phi_2 + \nu_3\phi_3$. It publishes public parameters PP as

$$g,\ w_1 = w^{\phi_1}, w_2 = w^{\phi_2}, w_3 = w^{\phi_3}, w,\ \hat{g}, \hat{g}^{\nu_1}, \hat{g}^{\nu_2}, \hat{g}^{-\tau},\ \hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi}.$$

SAS.KeyGen(PP): This algorithm takes as input the public parameters PP. It selects random exponents $\alpha, x, y \in \mathbb{Z}_p$ and computes $u = g^x, h = g^y, \hat{u} = \hat{g}^x, \hat{u}^{\nu_1} = (\hat{g}^{\nu_1})^x, \hat{u}^{\nu_2} = (\hat{g}^{\nu_2})^x, \hat{u}^{-\tau} = (\hat{g}^{-\tau})^x, \hat{h} = \hat{g}^y, \hat{h}^{\nu_1} = (\hat{g}^{\nu_1})^y, \hat{h}^{\nu_2} = (\hat{g}^{\nu_2})^y, \hat{h}^{-\tau} = (\hat{g}^{-\tau})^y$. It outputs a private key $SK = (\alpha, x, y)$ and a public key PK as

$$u, h,\ \hat{u}, \hat{u}^{\nu_1}, \hat{u}^{\nu_2}, \hat{u}^{-\tau},\ \hat{h}, \hat{h}^{\nu_1}, \hat{h}^{\nu_2}, \hat{h}^{-\tau},\ \Omega = e(g, \hat{g})^{\alpha}.$$

SAS.AggSign$(AS', \mathbf{M}', \mathbf{PK}', M, SK)$: This algorithm takes as input an aggregate-so-far $AS' = (S'_{1,1}, \ldots, S'_{2,4})$ on messages $\mathbf{M}' = (M_1, \ldots, M_{l-1})$ under public keys $\mathbf{PK}' = (PK_1, \ldots, PK_{l-1})$ where $PK_i = (u_i, h_i, \ldots, \Omega_i)$, a message

$M \in \{0,1\}^k$ where $k < \lambda$, a private key $SK = (\alpha, x, y)$ with $PK = (u, h, \ldots, \Omega)$ and PP. It first checks the validity of AS' by calling $\mathbf{AggVerify}(AS', \mathbf{M}', \mathbf{PK}')$. If AS' is not valid, then it halts. If the public key PK of SK does already exist in $\mathbf{PK}'$, then it halts. Next, it selects random exponents $r, c_1, c_2 \in \mathbb{Z}_p$ and outputs an aggregate signature AS as

$$
\begin{aligned}
&S_{1,1} = S'_{1,1} g^{\alpha} (S'_{2,1})^{xM+y} \cdot \prod_{i=1}^{l-1} (u_i^{M_i} h_i)^r (u^M h)^r w_1^{c_1},\\
&S_{1,2} = S'_{1,2} (S'_{2,2})^{xM+y} \cdot w_2^{c_1},\ S_{1,3} = S'_{1,3} (S'_{2,3})^{xM+y} \cdot w_3^{c_1},\\
&S_{1,4} = S'_{1,4} (S'_{2,4})^{xM+y} \cdot w^{c_1},\ S_{2,1} = S'_{2,1} \cdot g^r w_1^{c_2},\\
&S_{2,2} = S'_{2,2} \cdot w_2^{c_2},\ S_{2,3} = S'_{2,3} \cdot w_3^{c_2},\ S_{2,4} = S'_{2,4} \cdot w^{c_2}.
\end{aligned}
$$

SAS.AggVerify$(AS, \mathbf{M}, \mathbf{PK})$: This algorithm takes as input a sequential aggregate signature AS on messages $\mathbf{M} = (M_1, \ldots, M_l)$ under public keys $\mathbf{PK} = (PK_1, \ldots, PK_l)$ where $PK_i = (u_i, h_i, \ldots, \Omega_i)$. It first checks that any public key does not appear twice in $\mathbf{PK}$ and that any public key in $\mathbf{PK}$ has been certified. If these checks fail, then it outputs 0. If $l = 0$, then it outputs 1 if $S_1 = S_2 = 1$, 0 otherwise. It chooses random exponents $t, s_1, s_2 \in \mathbb{Z}_p$ and computes verification components as

$$
\begin{aligned}
&C_{1,1} = \hat{g}^t,\ C_{1,2} = (\hat{g}^{\nu_1})^t \hat{v}^{s_1},\ C_{1,3} = (\hat{g}^{\nu_2})^t (\hat{v}^{\nu_3})^{s_1},\ C_{1,4} = (\hat{g}^{-\tau})^t (\hat{v}^{-\pi})^{s_1},\\
&C_{2,1} = \prod_{i=1}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^t,\ C_{2,2} = \prod_{i=1}^{l} ((\hat{u}_i^{\nu_1})^{M_i} \hat{h}_i^{\nu_1})^t \hat{v}^{s_2},\\
&C_{2,3} = \prod_{i=1}^{l} ((\hat{u}_i^{\nu_2})^{M_i} \hat{h}_i^{\nu_2})^t (\hat{v}^{\nu_3})^{s_2},\ C_{2,4} = \prod_{i=1}^{l} ((\hat{u}_i^{-\tau})^{M_i} \hat{h}_i^{-\tau})^t (\hat{v}^{-\pi})^{s_2}.
\end{aligned}
$$

Next, it verifies that $\prod_{i=1}^{4} e(S_{1,i}, C_{1,i}) \cdot \prod_{i=1}^{4} e(S_{2,i}, C_{2,i})^{-1} \stackrel{?}{=} \prod_{i=1}^{l} \Omega_i^t$. If this equation holds, then it outputs 1. Otherwise, it outputs 0.

The aggregate signature AS is a valid sequential aggregate signature on messages $\mathbf{M}'||M$ under public keys $\mathbf{PK}'||PK$ with randomness $\tilde{r} = r' + r$, $\tilde{c}_1 = c'_1 + c'_2(xM + y) + c_1$, $\tilde{c}_2 = c'_2 + c_2$ where r', c'_1, c'_2 are random values in AS'. The sequential aggregate signature has the following form

$$
\begin{aligned}
&S_{1,1} = \prod_{i=1}^{l} g^{\alpha_i} \prod_{i=1}^{l} (u_i^{M_i} h_i)^{\tilde{r}} w_1^{\tilde{c}_1},\ S_{1,2} = w_2^{\tilde{c}_1},\ S_{1,3} = w_3^{\tilde{c}_1},\ S_{1,4} = w^{\tilde{c}_1},\\
&S_{2,1} = g^{\tilde{r}} w_1^{\tilde{c}_2},\ S_{2,2} = w_2^{\tilde{c}_2},\ S_{2,3} = w_3^{\tilde{c}_2},\ S_{2,4} = w^{\tilde{c}_2}.
\end{aligned}
$$

Theorem 2. *The above SAS scheme is existentially unforgeable under a chosen message attack if the PKS scheme is existentially unforgeable under a chosen message attack. That is, for any PPT adversary $\mathcal{A}$ for the above SAS scheme, there exists a PPT algorithm $\mathcal{B}$ for the PKS scheme such that $\boldsymbol{Adv}_{\mathcal{A}}^{SAS}(\lambda) \leq \boldsymbol{Adv}_{\mathcal{B}}^{PKS}(\lambda)$.*

The proof of this theorem is given in Section 4.2.

3.3 Extensions

In this section, we discuss various extensions of our SAS scheme.

Multiple Messages. To support multiple signing per one signer, we can use the method of Lu et al. [22]. The basic idea of Lu et al. is to apply a collision resistant hash function H to a message M before performing the signing algorithm. If a signer wants to add a signature on a message M_2 into the aggregate signature, he first removes his previous signature on $H(M_1)$ from the aggregate signature using his private key, and then he adds the new signature on the $H(M_1||M_2)$ to the aggregate signature.

Multi-signatures. The SAS scheme of this paper can be easily converted to a multi-signature scheme. In case of multi-signature, some elements of public keys in SAS can be moved to the public parameters since multi-signature only allows signers to sign on the same message. Compared to the multi-signature scheme of Lu et al. [22], our multi-signature scheme has short size public parameters.

4 Security Analysis

In this section, we analyze the security of the basic PKS scheme and our SAS scheme.

4.1 Proof of Theorem 1

To prove the security of our PKS scheme, we use the dual system encryption technique of Lewko and Waters [21]. We describe a semi-functional signing algorithm and a semi-functional verification algorithm. They are not used in a real system, rather they are used in the security proof. When comparing our proof to that of Lewko and Waters, we employ a different assumption since we have published additional elements g, u, h used in aggregation (in fact, direct adaptation of the earlier technique will break the assumption and thus the proof). A crucial idea in our proof is that we have added elements $\hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi}$ in the public key which are used in randomization of the verification algorithm. In the security proof when moving from normal to semi-functional verification, it is the randomization elements $\hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi}$ which are expanded to the semi-functional space; this enables deriving semi-functional verification as part of the security proof under our assumption, without being affected by the publication of the additional public key elements used for aggregation.

For the semi-functional signing and verification we set $f = g^{y_f}, \hat{f} = \hat{g}^{y_f}$ where y_f is a random exponent in $\mathbb{Z}_p$.

PKS.SignSF. The semi-functional signing algorithm first creates a normal signature using the private key. Let $(W'_{1,1}, \ldots, W'_{2,4})$ be the normal signature of a message M with random exponents $r, c_1, c_2 \in \mathbb{Z}_p$. It selects random exponents $s_k, z_k \in \mathbb{Z}_p$ and outputs a semi-functional signature σ as

$$W_{1,1} = W'_{1,1}(f^{\nu_1\nu_3-\nu_2})^{s_k z_k}, W_{1,2} = W'_{1,2}(f^{-\nu_3})^{s_k z_k}, W_{1,3} = W'_{1,3} f^{s_k z_k}, W_{1,4} = W'_{1,4},$$
$$W_{2,1} = W'_{2,1}(f^{\nu_1\nu_3-\nu_2})^{s_k}, \ W_{2,2} = W'_{2,2}(f^{-\nu_3})^{s_k}, \ W_{2,3} = W'_{2,3} f^{s_k}, \ W_{2,4} = W'_{2,4}.$$

PKS.VerifySF. The semi-functional verification algorithm first creates a normal verification components using the public key. Let $(V'_{1,1}, \ldots, V'_{2,4})$ be the normal verification components with random exponents $t, s_1, s_2 \in \mathbb{Z}_p$. It chooses random exponents $s_c, z_c \in \mathbb{Z}_p$ and computes semi-functional verification components as

$$V_{1,1} = V'_{1,1},\ V_{1,2} = V'_{1,2},\ V_{1,3} = V'_{1,3}\hat{f}^{s_c},\ V_{1,4} = V'_{1,4}(\hat{f}^{-\phi_3})^{s_c},$$
$$V_{2,1} = V'_{2,1},\ V_{2,2} = V'_{2,2},\ V_{2,3} = V'_{2,3}\hat{f}^{s_c z_c},\ V_{2,4} = V'_{2,4}(\hat{f}^{-\phi_3})^{s_c z_c}.$$

Next, it verifies that $\prod_{i=1}^{4} e(W_{1,i}, V_{1,i}) \cdot \prod_{i=1}^{4} e(W_{2,i}, V_{2,i})^{-1} \stackrel{?}{=} \Omega^t$. If this equation holds, then it outputs 1. Otherwise, it outputs 0.

Note that if the semi-functional verification algorithm verifies a semi-functional signature, then the left part of the above verification equation contains an additional random element $e(f, \hat{f})^{s_k s_c (z_k - z_c)}$. If $z_k = z_c$, then the semi-functional verification algorithm succeeds. In this case, we say that the signature is *nominally* semi-functional.

The security proof uses a sequence of games $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2, \mathbf{G}_3$: The first game $\mathbf{G}_0$ will be the original security game and the last game $\mathbf{G}_3$ will be a game such that an adversary $\mathcal{A}$ has no advantage. Formally, the hybrid games are defined as follows:

Game $\mathbf{G}_0$. In this game, the signatures that are given to $\mathcal{A}$ are normal and the challenger use the normal verification algorithm **PKS.Verify** to check the validity of the forged signature of $\mathcal{A}$.

Game $\mathbf{G}_1$. This game is almost identical to $\mathbf{G}_0$ except that the challenger use the semi-functional verification algorithm **PKS.VerifySF** to check the validity of the forged signature of $\mathcal{A}$.

Game $\mathbf{G}_2$. This game is the same as the $\mathbf{G}_1$ except that the signatures that are given to $\mathcal{A}$ will be semi-functional. At this moment, the signatures are semi-functional and the challenger use the semi-functional verification algorithm **PKS.VerifySF** to check the validity of the forged signature. Suppose that $\mathcal{A}$ makes at most q signature queries. For the security proof, we define a sequence of hybrid games $\mathbf{G}_{1,0}, \ldots, \mathbf{G}_{1,k}, \ldots, \mathbf{G}_{1,q}$ where $\mathbf{G}_{1,0} = \mathbf{G}_1$. In $\mathbf{G}_{1,k}$, a normal signature is given to $\mathcal{A}$ for all j-th signature queries such that $j > k$ and a semi-functional signature is given to $\mathcal{A}$ for all j-th signature queries such that $j \leq k$. It is obvious that $\mathbf{G}_{1,q}$ is equal to $\mathbf{G}_2$.

Game $\mathbf{G}_3$. Finally, we define a new game $\mathbf{G}_3$. This game differs from $\mathbf{G}_2$ in that the challenger always rejects the forged signature of $\mathcal{A}$. Therefore, the advantage of this game is zero since $\mathcal{A}$ cannot win this game.

For the security proof, we show the indistinguishability of each hybrid games. We informally describe the meaning of each indistinguishability as follows:

- Indistinguishability of $\mathbf{G}_0$ and $\mathbf{G}_1$: This property shows that $\mathcal{A}$ cannot forge a semi-functional signature if it is only given normal signatures. That is, if $\mathcal{A}$ forges a semi-functional signature, then it can distinguish $\mathbf{G}_0$ from $\mathbf{G}_1$.

- Indistinguishability of $\mathbf{G}_1$ and $\mathbf{G}_2$: This property shows that the probability of $\mathcal{A}$ to forge a normal signature is almost the same when the signatures given to the adversary are changed from normal type to semi-functional type. That is, if the probability of $\mathcal{A}$ to forge a normal signature is different in $\mathbf{G}_1$ and $\mathbf{G}_2$, then $\mathcal{A}$ can distinguish two games.
- Indistinguishability of $\mathbf{G}_2$ and $\mathbf{G}_3$: This property shows that $\mathcal{A}$ cannot forge a normal signature if it is only given semi-functional signatures. That is, if $\mathcal{A}$ forges a normal signature, then it can distinguish $\mathbf{G}_2$ from $\mathbf{G}_3$.

The security (unforgeability) of our PKS scheme follows from a hybrid argument. We first consider an adversary $\mathcal{A}$ to attack our PKS scheme in the original security game $\mathbf{G}_0$. By the indistinguishability of $\mathbf{G}_0$ and $\mathbf{G}_1$, we have that $\mathcal{A}$ can forge a normal signature with a non-negligible ϵ probability, but it can forge a semi-functional signature with only a negligible probability. Now we should show that the ϵ probability of $\mathcal{A}$ to forge a normal signature is also negligible. By the indistinguishability of $\mathbf{G}_1$ and $\mathbf{G}_2$, we have that the ϵ probability of $\mathcal{A}$ to forge a normal signature is almost the same when the signatures given to $\mathcal{A}$ are changed from normal type to semi-functional type. Finally, by the indistinguishability of $\mathbf{G}_2$ and $\mathbf{G}_3$, we have that $\mathcal{A}$ can forge a normal signature with only a negligible probability. Summing up, we obtain that the probability of $\mathcal{A}$ to forge a semi-functional signature is negligible (from the indistinguishability of $\mathbf{G}_0$ and $\mathbf{G}_1$) and the probability of $\mathcal{A}$ to forge a normal signature is also negligible (from the indistinguishability of $\mathbf{G}_2$ and $\mathbf{G}_3$).

Let $\mathbf{Adv}_{\mathcal{A}}^{G_j}$ be the advantage of $\mathcal{A}$ in $\mathbf{G}_j$ for $j = 0, \ldots, 3$. Let $\mathbf{Adv}_{\mathcal{A}}^{G_{1,k}}$ be the advantage of $\mathcal{A}$ in $\mathbf{G}_{1,k}$ for $k = 0, \ldots, q$. It is clear that $\mathbf{Adv}_{\mathcal{A}}^{G_0} = \mathbf{Adv}_{\mathcal{A}}^{PKS}(\lambda)$, $\mathbf{Adv}_{\mathcal{A}}^{G_{1,0}} = \mathbf{Adv}_{\mathcal{A}}^{G_1}$, $\mathbf{Adv}_{\mathcal{A}}^{G_{1,q}} = \mathbf{Adv}_{\mathcal{A}}^{G_2}$, and $\mathbf{Adv}_{\mathcal{A}}^{G_3} = 0$. From the following three Lemmas, we prove that it is hard for $\mathcal{A}$ to distinguish $\mathbf{G}_{i-1}$ from $\mathbf{G}_i$ under the given assumptions. Therefore, we have that

$$\begin{aligned}
&\mathbf{Adv}_{\mathcal{A}}^{PKS}(\lambda) \\
&= \mathbf{Adv}_{\mathcal{A}}^{G_0} + \sum_{i=1}^{2} \left(\mathbf{Adv}_{\mathcal{A}}^{G_i} - \mathbf{Adv}_{\mathcal{A}}^{G_i}\right) - \mathbf{Adv}_{\mathcal{A}}^{G_3} \leq \sum_{i=1}^{3} \left|\mathbf{Adv}_{\mathcal{A}}^{G_{i-1}} - \mathbf{Adv}_{\mathcal{A}}^{G_i}\right| \\
&= \mathbf{Adv}_{\mathcal{B}_1}^{A1}(\lambda) + \sum_{k=1}^{q} \mathbf{Adv}_{\mathcal{B}_2}^{A2}(\lambda) + \mathbf{Adv}_{\mathcal{B}_3}^{A3}(\lambda).
\end{aligned}$$

This completes our proof.

Lemma 1. *If Assumption 1 holds, then no polynomial-time adversary can distinguish between $\boldsymbol{G}_0$ and $\boldsymbol{G}_1$ with non-negligible advantage. That is, for any adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}_1$ such that $\left|\boldsymbol{Adv}_{\mathcal{A}}^{G_0} - \boldsymbol{Adv}_{\mathcal{A}}^{G_1}\right| = \boldsymbol{Adv}_{\mathcal{B}_1}^{A1}(\lambda)$.*

Lemma 2. *If Assumption 2 holds, then no polynomial-time adversary can distinguish between $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$ with non-negligible advantage. That is, for any adversary $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}_2$ such that $\left|\boldsymbol{Adv}_{\mathcal{A}}^{G_{1,k-1}} - \boldsymbol{Adv}_{\mathcal{A}}^{G_{1,k}}\right| = \boldsymbol{Adv}_{\mathcal{B}_2}^{A2}(\lambda)$.*

Lemma 3. *If Assumption 3 holds, then no polynomial-time adversary can distinguish between* $\boldsymbol{G}_2$ *and* $\boldsymbol{G}_3$ *with non-negligible advantage. That is, for any adversary* $\mathcal{A}$*, there exists a PPT algorithm* $\mathcal{B}_3$ *such that* $\left|\boldsymbol{Adv}_{\mathcal{A}}^{G_2} - \boldsymbol{Adv}_{\mathcal{A}}^{G_3}\right| = \boldsymbol{Adv}_{\mathcal{B}_3}^{A3}(\lambda)$.

The proofs of these lemmas are given in the full version of this paper [19].

4.2 Proof of Theorem 2

Our overall proof strategy for this part follows Lu et al. [22] and adapts it to our setting. The proof uses two properties: the fact that the aggregated signature result is independent of the order of aggregation, and the fact that the simulator of the SAS system possesses the private keys of all but the target PKS.

Suppose there exists an adversary $\mathcal{A}$ that forges the above SAS scheme with non-negligible advantage ϵ. A simulator $\mathcal{B}$ that forges the PKS scheme is first given: a challenge public key $PK_{PKS} = (g, u, h, w_1, \ldots, w, \hat{g}, \ldots, \hat{g}^{-\tau}, \hat{u}, \ldots, \hat{u}^{-\tau}, \hat{h}, \ldots, \hat{h}^{-\tau}, \hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi}, \Omega)$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows: $\mathcal{B}$ first constructs $PP = (g, w_1, \ldots, w, \hat{g}, \ldots, \hat{g}^{-\tau}, \hat{v}, \hat{v}^{\nu_3}, \hat{v}^{-\pi})$ and $PK^* = (u, h, \hat{u}, \ldots, \hat{u}^{-\tau}, \hat{h}, \ldots, \hat{h}^{-\tau}, \Omega = e(g, \hat{g})^{\alpha})$ from PK_{PKS}. Next, it initializes a certification list CL as an empty one and gives PP and PK^* to $\mathcal{A}$. $\mathcal{A}$ may adaptively requests certification queries or sequential aggregate signature queries. If $\mathcal{A}$ requests the certification of a public key by providing a public key $PK_i = (u_i, h_i, \ldots, \Omega_i)$ and its private key $SK_i = (\alpha_i, x_i, y_i)$, then $\mathcal{B}$ checks the private key and adds the key pair (PK_i, SK_i) to CL. If $\mathcal{A}$ requests a sequential aggregate signature by providing an aggregate-so-far AS' on messages $\mathbf{M}' = (M_1, \ldots, M_{l-1})$ under public keys $\mathbf{PK}' = (PK_1, \ldots, PK_{l-1})$, and a message M to sign under the challenge private key of PK^*, then $\mathcal{B}$ proceeds the aggregate signature query as follows:

1. It first checks that the signature AS' is valid and that each public key in $\mathbf{PK}'$ exits in CL.
2. It queries its signing oracle that simulates **PKS.Sign** on the message M for the challenge public key PK^* and obtains a signature σ.
3. For each $1 \leq i \leq l-1$, it constructs an aggregate signature on message M_i using **SAS.AggSign** since it knows the private key that corresponds to PK_i. The result signature is an aggregate signature for messages $\mathbf{M}'||M$ under public keys $\mathbf{PK}'||PK^*$ since this scheme does not check the order of aggregation. It gives the result signature AS to $\mathcal{A}$.

Finally, $\mathcal{A}$ outputs a forged aggregate signature $AS^* = (S^*_{1,1}, \ldots, S^*_{2,4})$ on messages $\mathbf{M}^* = (M_1, \ldots, M_l)$ under public keys $\mathbf{PK}^* = (PK_1, \ldots, PK_l)$ for some l. Without loss of generality, we assume that $PK_1 = PK^*$. $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ first checks the validity of AS^* by calling **SAS.AggVerify**. Additionally, the forged signature should not be trivial: the challenge public key PK^* must be in $\mathbf{PK}^*$, and the message M_1 must not be queried by $\mathcal{A}$ to the signature query oracle.

2. For each $2 \leq i \leq l$, it parses $PK_i = (u_i, h_i, \ldots, \Omega_i)$ from $\mathbf{PK}^*$, and it retrieves the private key $SK_i = (\alpha_i, x_i, y_i)$ of PK_i from CL. It then computes

$$W_{1,1} = S^*_{1,1} \cdot \prod_{i=2}^{l} \left(g^{\alpha_j} (S^*_{2,1})^{x_i M_i + y_i}\right)^{-1},\ W_{1,2} = S^*_{1,2} \cdot \prod_{i=2}^{l} \left((S^*_{2,2})^{x_i M_i + y_i}\right)^{-1},$$
$$W_{1,3} = S^*_{1,3} \cdot \prod_{i=2}^{l} \left((S^*_{2,3})^{x_i M_i + y_i}\right)^{-1},\ W_{1,4} = S^*_{1,4} \cdot \prod_{i=2}^{l} \left((S^*_{2,4})^{x_i M_i + y_i}\right)^{-1},$$
$$W_{2,1} = S^*_{2,1},\ W_{2,2} = S^*_{2,2},\ W_{2,3} = S^*_{2,3},\ W_{2,4} = S^*_{2,4}.$$

3. It outputs $\sigma = (W_{1,1}, \ldots, W_{2,4})$ as a non-trivial forgery of the PKS scheme since it did not make a signing query on M_1.

To finish the proof, we first show that the distribution of the simulation is correct. It is obvious that the public parameters and the public key are correctly distributed. The sequential aggregate signatures is correctly distributed since this scheme does not check the order of aggregation. Finally, we can show that the result signature $\sigma = (W_{1,1}, \ldots, W_{2,4})$ of the simulator is a valid signature for the PKS scheme on the message M_1 under the public key PK^* since it satisfies the following equation:

$$\begin{aligned}
&\prod_{i=1}^{4} e(W_{1,i}, V_{1,i}) \cdot \prod_{i=1}^{4} e(W_{2,i}, V_{2,i})^{-1} \\
&= e(S^*_{1,1}, \hat{g}^t) \cdot e(S^*_{1,2}, \hat{g}^{\nu_1 t} \hat{v}^{s_1}) \cdot e(S^*_{1,3}, \hat{g}^{\nu_2 t} \hat{v}^{\nu_3 s_1}) \cdot e(S^*_{1,4}, \hat{g}^{-\tau t} \hat{v}^{-\pi s_1}) \cdot e(\prod_{i=2}^{l} g^{\alpha_i}, \hat{g}^t)^{-1} \cdot \\
&\quad e(S^*_{2,1}, \prod_{i=2}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^t)^{-1} \cdot e(S^*_{2,2}, \prod_{i=2}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^{\nu_1 t} \hat{v}^{\delta_i s_1})^{-1} \cdot e(S^*_{2,3}, \prod_{i=2}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^{\nu_2 t} \hat{v}^{\delta_i s_1})^{-1} \cdot \\
&\quad e(S^*_{2,4}, \prod_{i=2}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^{-\tau t} \hat{v}^{-\pi \delta_i s_1})^{-1} \cdot e(S^*_{2,1}, (\hat{u}^{M_1} \hat{h})^t)^{-1} \cdot e(S^*_{2,2}, (\hat{u}^{M_1} \hat{h})^{\nu_1 t} \hat{v}^{s_2})^{-1} \cdot \\
&\quad e(S^*_{2,3}, (\hat{u}^{M_1} \hat{h})^{\nu_2 t} \hat{v}^{\nu_3 s_2})^{-1} \cdot e(S^*_{2,4}, (\hat{u}^{M_1} \hat{h})^{-\tau t} \hat{v}^{-\pi s_2})^{-1} \\
&= e(S^*_{1,1}, C_{1,1}) \cdot e(S^*_{1,2}, C_{1,2}) \cdot e(S^*_{1,3}, C_{1,3}) \cdot e(S^*_{1,4}, C_{1,4}) \cdot e(\prod_{i=2}^{l} g^{\alpha_i}, \hat{g}^t)^{-1} \cdot \\
&\quad e(S^*_{2,1}, \prod_{i=1}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^t)^{-1} \cdot e(S^*_{2,2}, \prod_{i=1}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^{\nu_1 t} \hat{v}^{\tilde{s}_2})^{-1} \cdot e(S^*_{2,3}, \prod_{i=1}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^{\nu_2 t} \hat{v}^{\tilde{s}_2})^{-1} \cdot \\
&\quad e(S^*_{2,4}, \prod_{i=1}^{l} (\hat{u}_i^{M_i} \hat{h}_i)^{-\tau t} \hat{v}^{-\pi \tilde{s}_2})^{-1} \cdot \\
&= \prod_{i=1}^{4} e(S^*_{1,i}, C_{1,i}) \cdot \prod_{i=1}^{4} e(S^*_{2,i}, C_{2,i})^{-1} \cdot e(\prod_{i=2}^{l} g^{\alpha_i}, \hat{g}^t)^{-1} = \prod_{i=1}^{l} \Omega_i^t \cdot \prod_{i=2}^{l} \Omega_i^{-t} = \Omega_1^t
\end{aligned}$$

where $\delta_i = x_i M_i + y_i$ and $\tilde{s}_2 = \sum_{i=2}^{l} (x_i M_i + y_i) s_1 + s_2$. This completes our proof.

5 Implementation

In this section, we report on the implementation of our SAS scheme and analysis of its performance.

We used the Pairing Based Cryptography (PBC) library of Ben Lynn [23] to implement our SAS scheme. According to the NIST recommendations for the 80-bit security [26], the key size of elliptic curve systems should be at least 160 bits and the key size of discrete logarithm systems should be at least 1024 bits. For 80-bit security, we, therefore, selected the Miyaji-Nakabayashi-Takano (MNT) curve with embedding degree 6. In the MNT curve with embedding degree 6, the group size of $\mathbb{G}$ should be at least 171 bits and the group size of $\mathbb{G}_T$ should be at least 1024 bits since the security of the $\mathbb{G}_T$ group is related to the security of the discrete logarithm [13]. Therefore, we used a 175-bit MNT curve that is generated by the MNT parameter generation program in the PBC library.

5.1 Signature and Public Key Size

We compare the signature size and the public key size of Lu et al.'s SAS scheme (the earlier scheme with non relaxed-model proof, based on a static assumption and standard model) with our SAS scheme. The original SAS scheme of Lu et al. is described using symmetric bilinear groups, but it can also be described using asymmetric bilinear groups. In the 175-bit MNT curve with point compression, the group size of $\mathbb{G}$ is about 175 bits, the group size of $\hat{\mathbb{G}}$ is about 525 bits, and the group size of $\mathbb{G}_T$ is 1050 bits respectively.

In Lu et al. system, the size of an aggregate signature is about 350 bits and the size of a public key is about 113,000 bits. Alternately, one may consider to use the method of Chatterjee and Sarkar [12] to reduce the public key size of the SAS scheme of Lu et al. However, this method obtains shorter public key size by sacrificing the security reduction of the scheme. Thus, it should use a larger size of prime for the order of groups to support the same security level of the original scheme.

5.2 Performance Measurements

We implemented and measured the performance of our SAS scheme on a notebook computer with an Intel Core i5-460M 2.53 GHz CPU. The PBC library on the test machine can compute a pairing operation in 14.0 ms, an exponentiation operation of $\mathbb{G}$ and $\hat{\mathbb{G}}$ in 1.7 ms and 20.3 ms respectively. We assume that there are 100 users who participate in the sequential aggregate signature system (indexed 1 to 100).

At first, the setup algorithm takes about 0.159 seconds to generate the public parameters and the key generation algorithm for each user takes about 0.185 seconds. The aggregate signing algorithm mainly consists of verifying the previous aggregate signature and adding its own signature into the aggregate signature. The time to generate an aggregate signature is proportional to the index number of the user who participates in the aggregate signing algorithm. Furthermore,

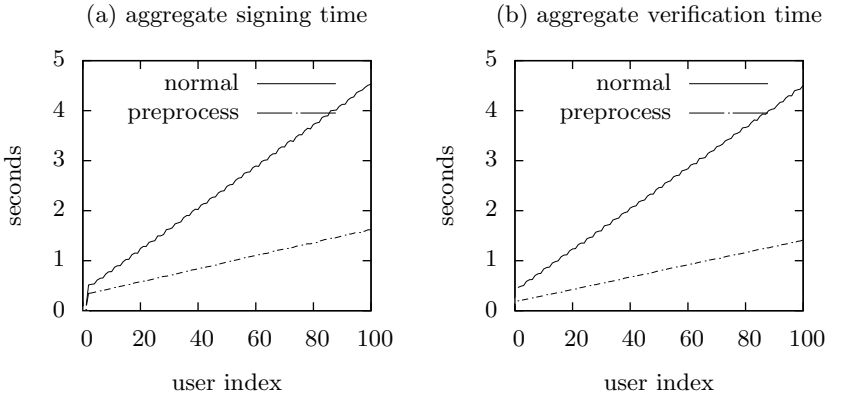


Fig. 1. Performance of our SAS scheme

this algorithm spends nearly 98 percent of its time on verifying the previous aggregate signature since it should compute $4l + 14$ numbers of exponentiation in $\hat{\mathbb{G}}$ where l is the number of previous signers.

Optimization: We can improve the performance of the aggregate verification algorithm by preprocessing the exponentiations in $\hat{\mathbb{G}}$. To use the preprocessing method, users should keep the public keys of the previous users. If the set of users who participate in the aggregate signature system is not changed or changed a little (as in the routing and the certification cases), then users can preprocess the public keys of previous users after running the first aggregate signing algorithm.

6 Conclusion

In this paper, we proposed a sequential aggregate signature scheme with a proof of security in the standard model and with no relaxation of assumptions (i.e., employing neither random oracles nor interactive assumptions). The proposed scheme is the first of this kind which has short (constant number of group elements) size public keys and constant number of pairing operations per message in the verification algorithm. Also, we provided an implementation and performance measurements of our scheme.

Acknowledgements. We thank Adam O'Neill and Benoît Libert for their helpful comments. We gratefully thank the anonymous reviewers of PKC 2013 for their valuable comments.

References

1. Ahn, J.H., Green, M., Hohenberger, S.: Synchronized aggregate signatures: new definitions, constructions and applications. In: ACM Conference on Computer and Communications Security, pp. 473–484 (2010)

2. Bagherzandi, A., Jarecki, S.: Identity-Based Aggregate and Multi-Signature Schemes Based on RSA. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 480–498. Springer, Heidelberg (2010)
3. Bellare, M., Namprempre, C., Neven, G.: Unrestricted Aggregate Signatures. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 411–422. Springer, Heidelberg (2007)
4. Bellare, M., Neven, G.: Identity-Based Multi-signatures from RSA. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 145–162. Springer, Heidelberg (2006)
5. Boldyreva, A.: Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2002)
6. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In: Ning, P., di Vimercati, S.D.C., Syverson, P.F. (eds.) ACM Conference on Computer and Communications Security, pp. 276–285. ACM (2007)
7. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. Cryptology ePrint Archive, Report 2007/438 (2010), http://eprint.iacr.org/2007/438
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)
9. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
10. 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)
11. Brogle, K., Goldberg, S., Reyzin, L.: Sequential Aggregate Signatures with Lazy Verification from Trapdoor Permutations (Extended Abstract). In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 644–662. Springer, Heidelberg (2012)
12. Chatterjee, S., Sarkar, P.: Trading Time for Space: Towards an Efficient IBE Scheme with Short(er) Public Parameters in the Standard Model. In: Won, D.H., Kim, S. (eds.) ICISC 2005. LNCS, vol. 3935, pp. 424–440. Springer, Heidelberg (2006)
13. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for cryptographers. Discrete Applied Mathematics 156(16), 3113–3121 (2008)
14. Gentry, C., Ramzan, Z.: Identity-Based Aggregate Signatures. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 257–273. Springer, Heidelberg (2006)
15. Gerbush, M., Lewko, A., O'Neill, A., Waters, B.: Dual Form Signatures: An Approach for Proving Security from Static Assumptions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 25–42. Springer, Heidelberg (2012)
16. Herranz, J.: Deterministic identity-based signatures for partial aggregation. Comput. J. 49(3), 322–330 (2006)
17. Hwang, J.Y., Lee, D.H., Yung, M.: Universal forgery of the identity-based sequential aggregate signature scheme. In: Li, W., Susilo, W., Tupakula, U.K., Safavi-Naini, R., Varadharajan, V. (eds.) ASIACCS, pp. 157–160. ACM (2009)
18. Katz, J., Lindell, A.Y.: Aggregate Message Authentication Codes. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 155–169. Springer, Heidelberg (2008)

19. Lee, K., Lee, D.H., Yung, M.: Sequential aggregate signatures with short public keys: Design, analysis, and implementation studies. Cryptology ePrint Archive, Report 2012/518 (2012), http://eprint.iacr.org/2012/518
20. Lewko, A.: Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 318–335. Springer, Heidelberg (2012)
21. Lewko, A., Waters, B.: New Techniques for Dual System Encryption and Fully Secure HIBE with Short Ciphertexts. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 455–479. Springer, Heidelberg (2010)
22. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential Aggregate Signatures and Multisignatures Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006)
23. Lynn, B.: The pairing-based cryptography library, http://crypto.stanford.edu/pbc/
24. Lysyanskaya, A., Micali, S., Reyzin, L., Shacham, H.: Sequential Aggregate Signatures from Trapdoor Permutations. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 74–90. Springer, Heidelberg (2004)
25. Neven, G.: Efficient Sequential Aggregate Signed Data. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 52–69. Springer, Heidelberg (2008)
26. NIST: Recommendation for key management (2011), http://csrc.nist.gov/publications/PubsSPs.html
27. Schröder, D.: How to Aggregate the CL Signature Scheme. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 298–314. Springer, Heidelberg (2011)
28. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
29. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

New Constructions and Applications of Trapdoor DDH Groups

Yannick Seurin

ANSSI, Paris, France
yannick.seurin@m4x.org

Abstract. Trapdoor Decisional Diffie-Hellman (TDDH) groups, introduced by Dent and Galbraith (ANTS 2006), are groups where the DDH problem is hard, unless one is in possession of a secret trapdoor which enables solving it efficiently. Despite their intuitively appealing properties, they have found up to now very few cryptographic applications. Moreover, among the two constructions of such groups proposed by Dent and Galbraith, only a single one based on hidden pairings remains unbroken. In this paper, we extend the set of trapdoor DDH groups by giving a construction based on composite residuosity. We also introduce a more restrictive variant of these groups that we name *static* trapdoor DDH groups, where the trapdoor only enables to solve the DDH problem with respect to a fixed pair (G, G^x) of group elements. We give two constructions for such groups whose security relies respectively on the RSA and the factoring assumptions. Then, we show that static trapdoor DDH groups yield elementary constructions of convertible undeniable signature schemes allowing delegatable verification. Using our constructions of static trapdoor DDH groups from the RSA or the factoring assumption, we obtain slightly simpler variants of the undeniable signature schemes of respectively Gennaro, Rabin, and Krawczyk (J. Cryptology, 2000) and Galbraith and Mao (CT-RSA 2003). These new schemes are conceptually more satisfying since they can strictly be viewed as instantiations, in an adequate group, of the original undeniable signature scheme of Chaum and van Antwerpen (CRYPTO '89).

1 Introduction

The CDH and DDH Problems. Given a group $\mathbb{G}$ and an element $G \in \mathbb{G}$ of large order, the Computational Diffie-Hellman (CDH) problem is to compute G^{xy}, given G^x and G^y for random integers x, y. The Decisional Diffie-Hellman (DDH) problem is to distinguish the two distributions (G^x, G^y, G^{xy}) and (G^x, G^y, G^z) for random and independent integers x, y, z. Usually, when considering the status of various groups with respect to the CDH and DDH problems, one of the following two cases arises: either the CDH and DDH problems are both presumably hard (this is the case for example for subgroups of large prime order of $\mathbb{Z}_p^*$, p prime), or the group is a so-called *gap group*: the CDH problem is (presumably) hard while the DDH problem is universally easy (*i.e.* easy given only the description of the group law, which seems to be the

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 443–460, 2013.

minimal publicly available information to obtain useful applications). The latter case typically arises in certain elliptic curve groups equipped with bilinear pairings [35,22], and has given rise to many important applications in cryptography [32,3,4].

Trapdoor DDH Groups. Trapdoor DDH groups (*TDDH groups* for short), introduced by Dent and Galbraith [18], lie somewhere between the above two cases. These are groups where the DDH problem is hard, except if one possesses a trapdoor for solving it efficiently. Dent and Galbraith gave two candidates for such groups based on the concept of *hidden pairings*, one in elliptic curves over the ring $\mathbb{Z}_N$, where N is hard to factor, and the other one based on Frey's idea of disguising an elliptic curve [21]. Subsequently, the second proposal was broken by Morales [38]. Since the DDH problem is the basis of so many cryptosystems [2], the concept of trapdoor DDH groups is very attractive. Indeed, it should enable to control more precisely who is able to solve the DDH problem in a system. This may help in situations where there is a conflict between security, which requires a group where the DDH problem is hard, and some interesting additional functionalities that could be achieved thanks to an algorithm for solving the DDH problem. One example that comes to mind is threshold ElGamal encryption. In threshold ElGamal encryption [19], given a secret/public key pair $(x, X = G^x)$, each decryption server is given a share x_i of the secret key, to which is associated a "partial" public key G^{x_i}. In order to decrypt a ciphertext $(R, Y) = (G^r, MX^r)$, each server participating to decryption must compute a decryption share $S_i = R^{x_i}$. Hence, checking whether a decryption share from a server is correct or not amounts to deciding whether (X_i, R, S_i) is a DDH tuple or not. Yet IND-CPA-security of ElGamal encryption is equivalent to the hardness of the DDH problem in the underlying group $\mathbb{G}$ [46]. Hence, there seems to be no other choice than using a group where the DDH problem is hard, thereby condemning other participants to be unable to distinguish correct decryption shares from incorrect ones. We do not claim that TDDH groups are the best way to solve this problem (this can be more easily achieved by having each server provide a non-interactive zero-knowledge proof that his decryption share is correctly computed), and this example only serves to argue that sometimes, one may want that only some authorized party be able to solve the DDH problem. Despite these considerations, TDDH groups have found up to now very few cryptographic applications. In their original paper, Dent and Galbraith gave only one example, namely an identification scheme. To the best of our knowledge, the only previous paper proposing a non-trivial application of TDDH groups (namely the construction of statistically hiding sets, a variant of zero-knowledge sets) is due to Prabhakaran and Xue [43].

Contributions of This Work. The contributions of this paper can be summarized as follows. First, at a conceptual level, we refine the definition of TDDH groups of Dent and Galbraith by requiring that the CDH problem remain hard even given the trapdoor for solving the DDH problem. This was not made explicit in the formalization by Dent and Galbraith, yet we think that this is probably

a key feature for many interesting applications, such as undeniable signatures for example. We also broaden the set of constructions of trapdoor DDH groups. We propose a new construction based on composite residuosity in $\mathbb{Z}_{N^2}^*$ (similar considerations have been made by [6], albeit not in the formalism of TDDH groups), and identify under which hardness assumptions this group satisfies our definition. A drawback of this construction is that it lacks what we call *perfect soundness*, meaning that the algorithm solving the DDH problem with the trapdoor can sometimes err and declare valid a non-DH tuple.

Then, we introduce a variant of trapdoor DDH groups that we name *static* trapdoor DDH groups. Their definition is very similar to the one of trapdoor DDH groups, except that the trapdoor for solving the DDH problem is now dedicated to a specific pair of group elements (G, G^x), hence the name *static*. We then show that such groups can be easily constructed from the RSA and the factoring problems. This concept abstracts some of the ideas underlying the work of Hofheinz and Kiltz [31], who showed that the Strong Diffie-Hellman (SDH) problem (*i.e.* solving the CDH problem given access to a static DDH oracle) is hard in the so-called group of signed quadratic residues under the factoring assumption.

Finally, we describe a very natural application of (static or not) TDDH groups to convertible undeniable signature schemes. Namely, the construction we propose is exactly the original undeniable signature scheme proposed by Chaum and van Antwerpen [12] (for which deciding the validity of a signature is equivalent to solving the DDH problem), but in a TDDH group rather than simply a group where the DDH problem is hard. The trapdoor for solving the DDH problem can then be used to universally convert or delegate verification of signatures. Once instantiated with our proposals of static TDDH groups based on the RSA or the factoring problems, we obtain schemes similar to previous RSA-based undeniable signature schemes due to Gennaro, Rabin, and Krawczyk [26] and Galbraith and Mao [23]. However, these new schemes are conceptually simpler and easier to analyze. Moreover, since they are strict instantiations of the Chaum and van Antwerpen scheme, their confirmation and disavowal protocols can use classical proofs of equality or inequality of discrete logarithms, which are simpler and more efficient than what was proposed previously for the schemes of [26,23].

Open Problems. Two key features of TDDH groups are perfect soundness (the property that the algorithm for solving the DDH problem with the trapdoor perfectly distinguishes DH tuples from non-DH tuples), and the possibility to securely hash into the group (see discussion in Section 2.3). However, none of the two candidates for TDDH groups (the hidden pairing based proposal of [18], and our proposal in Section 3.2) fulfills both requirements. We think that providing a plausible candidate possessing both properties is the key to enable powerful applications of TDDH groups.[1] A related open problem is whether there exists a (plausible construction of a) TDDH group with publicly known (ideally prime) order, since they are usually simpler to use in cryptography.

[1] Our examples of *static* TDDH groups do fulfill both requirement, however non-static TDDH groups would allow more flexibility in cryptographic applications.

Organization. In Section 2 we give some basic definitions and introduce some of the tools we will need in the remainder of the paper. In Section 3, we define trapdoor DDH groups, and give a construction based on composite residuosity. In Section 4, we introduce static trapdoor DDH groups, and give two constructions based on respectively the RSA and the factoring assumptions. Finally, in Section 5, we show how to obtain convertible undeniable signature schemes from static TDDH groups, and discuss their instantiation with the constructions described previously.

2 Preliminaries

2.1 Notation and Definitions

The set of integers i such that $a \leq i \leq b$ will be denoted $[a;b]$. The security parameter will be denoted k. A function f of the security parameter is said *negligible* if for any $c > 0$, $f(k) \leq 1/k^c$ for sufficiently large k. When S is a non-empty finite set, we write $s \leftarrow_{\$} S$ to mean that a value is sampled uniformly at random from S and assigned to s. By $z \leftarrow \mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2,\cdots}(x,y,\ldots)$ we denote the operation of running the (possibly probabilistic) algorithm $\mathcal{A}$ on inputs $x, y, \ldots$ with access to oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$ (possibly none), and letting z be the output. PPT will stand for probabilistic polynomial-time. Given two Interactive Turing Machines $\mathcal{P}$ and $\mathcal{V}$, we denote $w \leftarrow \langle \mathcal{P}(x), \mathcal{V}(y)\rangle(z)$ to mean that the output of the interaction of $\mathcal{P}$ with private input x and $\mathcal{V}$ with private input y on common input z is w.

Given an integer N, the multiplicative group of integers modulo N is denoted $\mathbb{Z}_N^*$. This group has order $\phi(N)$ where $\phi(\cdot)$ is the Euler function and exponent $\lambda(N)$ where $\lambda(\cdot)$ is the Carmichael function. We denote $\mathbb{J}_N$ the subgroup of $\mathbb{Z}_N^*$ of all elements $x \in \mathbb{Z}_N^*$ with Jacobi symbol $\left(\frac{x}{N}\right) = 1$. This subgroup has index 2 and order $\phi(N)/2$ in $\mathbb{Z}_N^*$. Moreover it is efficiently recognizable even without the factorization of N since the Jacobi symbol is efficiently computable given only N. We also denote $\mathbb{QR}_N$ the subgroup of quadratic residues of $\mathbb{Z}_N^*$. This subgroup is widely believed not to be efficiently recognizable when N is composite and its factorization is unknown (this is the Quadratic Residuosity assumption). We call a prime number p such that $(p-1)/2$ is prime a *safe prime*.

In all the following, given a group $\mathbb{G}$, we use the notation $[\mathbb{G}]$ to denote a description of the group, *i.e.* an efficient algorithm for computing the group operation. This notation always implies that $\mathbb{G}$ is efficiently recognizable. We assume that it is always possible to derive from the description of the group a negligibly close upper bound on the order $|\mathbb{G}|$ of the group (in some cases the exact order may be efficiently computable), and we use the notation $|\mathbb{G}|^+$ to denote this upper bound.[2] Given an element $G \in \mathbb{G}$, we denote $\texttt{ord}(G)$ its order, $\langle G\rangle$ the group generated by G, $\texttt{Dlog}_G(X)$ the discrete logarithm in base

[2] *E.g.* when $\mathbb{G} = \mathbb{Z}_p^*$ for some prime number p, $|\mathbb{G}|^+ = p-1$, while when $\mathbb{G} = \mathbb{Z}_N^*$, where the factorization of N is secret, $|\mathbb{G}|^+ = N$.

G of an element $X \in \langle G \rangle$, and $\mathtt{CDH}_G(X,Y) = G^{\mathtt{Dlog}_G(X)\mathtt{Dlog}_G(Y)}$. We also denote $\mathcal{DH}_G \subset \langle G \rangle^3$ the set of Diffie-Hellman (DH) tuples with respect to G:

$$\mathcal{DH}_G = \{(G^x, G^y, G^{xy}), x, y \in [0; \mathtt{ord}(G) - 1]\} \quad .$$

A group generator $\mathtt{Gen}$ is a PPT algorithm which on input a security parameter 1^k, outputs a tuple $([\mathbb{G}], G, \gamma)$ where $[\mathbb{G}]$ is the description of a group $\mathbb{G}$, $G \in \mathbb{G}$ is an element of order $2^{\Theta(k)}$, and γ is some arbitrary side information. We say that the CDH problem is hard for $\mathtt{Gen}$ if for any PPT adversary $\mathcal{A}$, the following probability is negligible:

$$\Pr\left[([\mathbb{G}], G, \gamma) \leftarrow \mathtt{Gen}(1^k), (X,Y) \leftarrow_{\$} \langle G \rangle^2, Z \leftarrow \mathcal{A}([\mathbb{G}], G, \gamma; X, Y) : Z = \mathtt{CDH}_G(X,Y)\right] \quad .$$

We say that the DDH problem is hard for $\mathtt{Gen}$ if for any PPT adversary $\mathcal{A}$, the following advantage is negligible:

$$\Big|\Pr\left[([\mathbb{G}], G, \gamma) \leftarrow \mathtt{Gen}(1^k), (X,Y) \leftarrow_{\$} \langle G \rangle^2, Z \leftarrow \mathtt{CDH}_G(X,Y) : 1 \leftarrow \mathcal{A}([\mathbb{G}], G, \gamma; X, Y, Z)\right] - \Pr\left[([\mathbb{G}], G, \gamma) \leftarrow \mathtt{Gen}(1^k), (X,Y,Z) \leftarrow_{\$} \langle G \rangle^3 : 1 \leftarrow \mathcal{A}([\mathbb{G}], G, \gamma; X, Y, Z)\right]\Big| \quad .$$

2.2 Proofs of Equality and Inequality of Discrete Logarithms

Protocols for proving, given $(G, X, Y, Z) \in \mathbb{G}$, the equality of discrete logarithms (EDL) $\mathtt{Dlog}_G(X) = \mathtt{Dlog}_Y(Z)$ or the inequality of discrete logarithms (IDL) constitute (among many other applications) the heart of respectively the confirmation and disavowal protocols for many undeniable signature schemes, and have therefore been the subject of many works. They vary depending on the exact kind of zero-knowledge property one wants to achieve. The basic honest-verifier zero-knowledge (HVZK) proof of EDL is due to Chaum and Pedersen [11], while the simplest HVZK proof of IDL is due to Camenish and Shoup [9]. These protocols are usually described for ambient groups $\mathbb{G}$ with publicly known prime order, in which case recognizing $\langle G \rangle$ is trivial, so that these protocols are actually proofs that a tuple $(X, Y, Z) \in \mathbb{G}^3$ is in $\mathcal{DH}_G$ or not. They can be adapted to the case where the order of the ambient group is composite and secret using well-known techniques [27,28], with the caveat that if $\langle G \rangle$ is not efficiently recognizable, the verifier must be promised that $X, Y, Z \in \langle G \rangle$ since these proofs do not in general ensure membership of X, Y, Z in $\langle G \rangle$ with negligible soundness.[3] Stated differently, if $\mathbb{G}'$ is a cyclic and efficiently recognizable subgroup of $\mathbb{G}$ (*e.g.* $\mathbb{G} = \mathbb{Z}_N^*$ and $\mathbb{G}' = \mathbb{J}_N$ when $\mathbb{J}_N$ is cyclic), these protocols are actually proofs that a tuple $(X, Y, Z) \in \mathbb{G}'$ is a DH tuple with respect to G or not, assuming that the verifier is guaranteed that G is indeed a generator of

[3] The soundness of the Schnorr protocol [44], seen as a proof of membership in $\langle G \rangle$, is $1/\ell$, where ℓ is the smallest prime factor of the order of the ambient group $\mathbb{G}$.

$\mathbb{G}'$ (which may not be efficiently checkable). The HVZK protocols for EDL and IDL are described in the full version of the paper [45]. They can be strengthen to achieve various notions of zero-knowledge (against cheating verifiers) using known techniques [29,14,16,25] that we do not discuss in this paper.

The HVZK proofs of EDL and IDL can be made non-interactive in the Random Oracle Model using the Fiat-Shamir transformation [20], *i.e.* by having the prover compute the challenge (first message from the verifier) by itself by applying a hash function to the commitment (first message from the prover). Note that these proofs then become universally convincing.

2.3 Hashing into Groups

For many applications (and in particular for undeniable signatures based on the Chaum and van Antwerpen scheme [12]), it is required to securely hash into the subgroup $\langle G \rangle$ specified by the group generator $\texttt{Gen}$. We discuss this in more details in the full version of the paper [45].

3 Trapdoor DDH Groups

We start by defining trapdoor DDH groups. Our definition is a refinement of the one of Dent and Galbraith [18] in that we explicitly require that the CDH problem remain hard even given the trapdoor τ enabling to solve the DDH problem.

3.1 Definition

Definition 1. *A* trapdoor DDH group $\mathcal{TDDH}$ *is a pair of algorithms* $(\texttt{Gen}, \texttt{Solve})$ *with the following properties. The trapdoor DDH group generator algorithm* $\texttt{Gen}$ *is a PPT algorithm which takes as input a security parameter* 1^k *and outputs a tuple* $([\mathbb{G}], G, \tau)$ *where* $[\mathbb{G}]$ *is the description of a group* $\mathbb{G}$*,* $G \in \mathbb{G}$ *is a group element of order* $2^{\Theta(k)}$*, and* τ *is a trapdoor information, such that:*

i) hardness of DDH without the trapdoor*: the DDH problem is hard for the group generator* $\texttt{Gen}'$ *which outputs only* $([\mathbb{G}], G)$*;*
ii) hardness of CDH with the trapdoor*: the CDH problem is hard for* $\texttt{Gen}$*.*

$\texttt{Solve}$ *is a deterministic polynomial-time algorithm which takes as input* $([\mathbb{G}], G, \tau)$ *and a tuple* $(X, Y, Z) \in \mathbb{G}^3$*, either accepts (outputs 1) or rejects (outputs 0), and satisfies the following:*

iii) completeness*: for all* $([\mathbb{G}], G, \tau)$ *possibly output by* $\texttt{Gen}$*,* $\texttt{Solve}$ *always accepts on input a DH tuple* $(X, Y, Z) \in \mathcal{DH}_G$*;*
iv) soundness*: for any PPT adversary* $\mathcal{A}$*, the following probability is negligible:*

$$\Pr\left[([\mathbb{G}], G, \tau) \leftarrow \texttt{Gen}(1^k), (X, Y) \leftarrow_{\$} \langle G \rangle^2, Z \leftarrow \mathcal{A}([\mathbb{G}], G; X, Y) : 1 \leftarrow \texttt{Solve}([\mathbb{G}], G, \tau; X, Y, Z) \wedge (X, Y, Z) \notin \mathcal{DH}_G\right] .$$

We say that $\mathcal{TDDH}$ has perfect soundness when `Solve` *always rejects on input a non-DH tuple (X, Y, Z), so that the above probability is zero.*

Note that the soundness condition implies in particular that `Solve`, on input a *uniformly random* tuple $(X, Y, Z) \in \mathbb{G}^3$, accepts only with negligible probability. We silently assumed in the above definition that `Solve` is always run with a correctly generated trapdoor. This is safe for all examples presented below since there is an efficient way, given $([\mathbb{G}], G, \tau)$, to check whether the trapdoor is correct. We assume that `Solve` outputs a special symbol $\perp$ when this is not the case. We recall the original proposal of a TDDH group based on hidden pairings by Dent and Galbraith [18] in the full version of the paper [45].

3.2 A TDDH Group Based on Composite Residuosity

In this section, we describe a TDDH group $\mathcal{TDDH}_{\texttt{BCP}}$ based on the group of quadratic residues modulo N^2, where N is an RSA modulus. This group was first considered by Bresson, Catalano, and Pointcheval [6], who noticed that when the factorization of N is publicly available, this constitutes a *gap group*, *i.e.* a group where the CDH problem is hard and the DDH problem is easy. Here, we show that it constitutes in fact a TDDH group when the factorization of N is kept secret and used as the trapdoor.

We first recall some basic facts about the group of quadratic residues modulo N^2, where N is an RSA modulus. Let p, q be two safe primes where $p = 2p' + 1$ and $q = 2q' + 1$ (p' and q' primes), and $N = pq$. The group $\mathbb{QR}_{N^2}$ of quadratic residues modulo N^2 is a cyclic group of order $m = Np'q'$. We define the notion of *partial discrete logarithm.*

Definition 2 (Partial Discrete Logarithm). *Given a generator G of $\mathbb{QR}_{N^2}$, the partial discrete logarithm of a group element $X \in \mathbb{QR}_{N^2}$ is defined as* $\texttt{PDlog}_G(X) = \texttt{Dlog}_G(X) \bmod N$.

Computing the partial discrete logarithm is believed to be hard without the factorization of N.[4] However, it can be efficiently computed given the prime factors of N (or simply $\lambda(N)$) as follows [42]:

1. input: N, $\lambda(N)$, generator G of $\mathbb{QR}_{N^2}$ and $X \in \mathbb{QR}_{N^2}$; output: $\texttt{PDlog}_G(X)$
2. for integers $u \in [0; N^2 - 1]$ such that $u = 1 \bmod N$, define the function (having integer values) $\mathcal{L}(u) = (u - 1)/N$
3. return
$$\frac{\mathcal{L}(X^{\lambda(N)} \bmod N^2)}{\mathcal{L}(G^{\lambda(N)} \bmod N^2)} \bmod N \ .$$

We now formally describe the TDDH group $\mathcal{TDDH}_{\texttt{BCP}}$. On input the security parameter 1^k, $\texttt{Gen}_{\texttt{BCP}}$ selects two k-bit safe primes $p = 2p' + 1$ and $q = 2q' + 1$,

[4] As noted by Paillier [42] and in [6], the Partial Discrete Logarithm problem can be shown equivalent to the Composite Residuosity Class problem in the particular case considered here.

sets $N = pq$, selects a random generator G of $\mathbb{QR}_{N^2}$, and outputs $([\mathbb{Z}^*_{N^2}], G, \tau = (p, q))$. The $\mathtt{Solve}_{\mathtt{BCP}}$ algorithm works as follows: on input a tuple $(X, Y, Z) \in (\mathbb{Z}^*_{N^2})^3$ (as well as the trapdoor $\tau = (p, q)$), it checks whether $X, Y, Z \in \mathbb{QR}_{N^2}$, computes $x' = \mathtt{PDlog}_G(X)$, $y' = \mathtt{PDlog}_G(Y)$, and $z' = \mathtt{PDlog}_G(Z)$ as described above, and checks whether $z' = x'y' \bmod N$. It accepts if this holds and rejects otherwise. The security of this TDDH group relies on a "partial" version of the CDH problem, defined as follows.

Definition 3 (Partial CDH Problem). *We say that the* Partial CDH problem *is hard if for any PPT algorithm $\mathcal{A}$, the following probability is negligible:*

$$\Pr[([\mathbb{Z}^*_{N^2}], G, \tau) \leftarrow \mathtt{Gen}_{\mathtt{BCP}}(1^k), (X, Y) \leftarrow_{\$} \langle G \rangle^2, Z \leftarrow \mathcal{A}([\mathbb{Z}^*_{N^2}], G; X, Y) : \mathtt{Dlog}_G(Z) \equiv \mathtt{Dlog}_G(X)\mathtt{Dlog}_G(Y) \bmod N] \ .$$

Theorem 1. *Assuming that the DDH problem (without the factorization of N), the CDH problem (with the factorization of N), and the Partial CDH problem (without the factorization of N) are hard for $\mathbb{QR}_{N^2}$, $\mathcal{TDDH}_{\mathtt{BCP}}$ is a trapdoor DDH group.*

Proof. We prove that properties *i*) to *iv*) of Definition 1 are satisfied. Properties *i*) and *ii*) follow directly from the assumptions that respectively the DDH (without the factorization of N) and the CDH (with the factorization of N) problems are hard in $\mathbb{QR}_{N^2}$. Property *iii*) is straightforward to verify by definition of $\mathtt{Solve}_{\mathtt{BCP}}$. Finally, property *iv*) follows from the hardness of the Partial CDH problem. □

Note that this TDDH group does not have perfect soundness. In particular, on input a random tuple $(X, Y, Z) \in (\mathbb{QR}_{N^2})^3$, there is a negligible probability that $\mathtt{Solve}_{\mathtt{BCP}}$ accepts and yet $(X, Y, Z) \notin \mathcal{DH}_G$ (this probability can easily be seen to be $\mathcal{O}(1/N)$ [6]). Moreover, given the trapdoor $\tau = (p, q)$, and two random elements $(X, Y) \in (\mathbb{QR}_{N^2})^2$, it is easy to generate Z such that $(X, Y, Z) \notin \mathcal{DH}_G$ and yet $\mathtt{Solve}_{\mathtt{BCP}}$ accepts on input (X, Y, Z): simply compute $x' = \mathtt{PDlog}_G(X)$ and $y' = \mathtt{PDlog}_G(Y)$ and output $G^{x'y' \bmod N}$. Alternatively, given two random elements $(X, Y) \in (\mathbb{QR}_{N^2})^2$ and $Z = \mathtt{CDH}_G(X, Y)$, it is easy to compute $Z' \neq Z$ such that $\mathtt{Solve}_{\mathtt{BCP}}$ accepts on input (X, Y, Z'): simply compute $Z' = ZU^N$ for some random $U \in \mathbb{QR}_{N^2}$. This may be of concern in some applications, especially for undeniable signature schemes where $\mathtt{Solve}$ is typically used to check the validity of signatures (see Section 5).[5]

4 Static Trapdoor DDH Groups

In this section, we define and construct *static* trapdoor DDH groups. They are similar to trapdoor DDH groups as defined in Section 3, except that the trapdoor only allows to solve the DDH problem with respect to a specific pair of group elements (G, G^x).

[5] We note however that imperfect soundness is not a problem for the identification scheme outlined in [18].

4.1 Definition

Definition 4. *A* static trapdoor DDH group $\mathcal{STDDH}$ *is a tuple of algorithms* $(\mathtt{Gen}, \mathtt{Samp}, \mathtt{Solve})$ *with the following properties. The static trapdoor DDH group generator algorithm* $\mathtt{Gen}$ *is a PPT algorithm which takes as input a security parameter* 1^k *and outputs a tuple* $([\mathbb{G}], G, \tau)$ *where* $[\mathbb{G}]$ *is the description of a group* $\mathbb{G}$*,* $G \in \mathbb{G}$ *is a group element of order* $2^{\Theta(k)}$*, and* τ *is a (master) trapdoor information, such that:*

i) hardness of DDH without the trapdoor*: the DDH problem is hard for the group generator* $\mathtt{Gen}'$ *which outputs only* $([\mathbb{G}], G)$*.*

$\mathtt{Samp}$ *is a PPT algorithm which on input* $([\mathbb{G}], G, \tau)$*, samples uniformly at random a group element* $X \leftarrow_{\$} \langle G \rangle$*, and outputs*[6] (X, x, τ_x) *where* $x = \mathtt{Dlog}_G(X)$ *and* τ_x *is a (static) trapdoor information, such that:*

ii) hardness of CDH with the static trapdoor*: for any PPT algorithm* $\mathcal{A}$*, the following probability is negligible:*

$$\Pr\left[([\mathbb{G}], G, \tau) \leftarrow \mathtt{Gen}(1^k), (X, x, \tau_x) \leftarrow \mathtt{Samp}([\mathbb{G}], G, \tau), Y \leftarrow_{\$} \langle G \rangle, \right.$$
$$\left. Z \leftarrow \mathcal{A}([\mathbb{G}], G; X, Y; \tau_x) : Z = \mathtt{CDH}_G(X, Y)\right] .$$

$\mathtt{Solve}$ *is a deterministic polynomial-time algorithm which takes as input* $([\mathbb{G}], G)$*, a tuple* $(X, Y, Z) \in \langle G \rangle \times \mathbb{G}^2$*, and the trapdoor* τ_x *for* X*, either accepts (outputs 1) or rejects (outputs 0), and satisfies the following:*

iii) completeness*: for all* $([\mathbb{G}], G, \tau)$ *and* (X, x, τ_x) *possibly output by* $\mathtt{Gen}$ *and* $\mathtt{Samp}$*, and any* $(Y, Z) \in \mathbb{G}^2$*,* $\mathtt{Solve}$ *always accepts when* $(X, Y, Z) \in \mathcal{DH}_G$*;*

iv) soundness*: for any PPT adversary* $\mathcal{A}$*, the following probability is negligible:*

$$\Pr\left[([\mathbb{G}], G, \tau) \leftarrow \mathtt{Gen}(1^k), (X, x, \tau_x) \leftarrow \mathtt{Samp}([\mathbb{G}], G, \tau), Y \leftarrow_{\$} \langle G \rangle, \right.$$
$$\left. Z \leftarrow \mathcal{A}([\mathbb{G}], G; X, Y) : 1 \leftarrow \mathtt{Solve}([\mathbb{G}], G; X, Y, Z; \tau_x) \wedge (X, Y, Z) \notin \mathcal{DH}_G\right]$$

We say that $\mathcal{STDDH}$ *has perfect soundness when* $\mathtt{Solve}$ *always rejects on input a non-DH tuple* (X, Y, Z)*, so that the above probability is zero.*

Again, we silently assumed that $\mathtt{Solve}$ is always run with the correct trapdoor τ_x because in all examples below this can be checked efficiently. In the remainder of this section, we propose two constructions of static TDDH groups based respectively on the RSA problem and the factoring problem.

4.2 A Construction Based on the RSA Problem

We first show how a static TDDH group can be obtained from the RSA problem. Let $N = pq$ be an RSA modulus. When $(p-1)/2$ and $(q-1)/2$ are coprime,

[6] We stress that in typical applications, x is retained by an authorized user and is never made available to the adversary.

then the subgroup $\mathbb{J}_N$ of $\mathbb{Z}_N^*$ is cyclic. Moreover, when p and q are distinct safe primes, the DDH problem is widely believed to be hard in $\mathbb{J}_N$ [2]. We define the static TDDH group $\mathcal{STDDH}_{\mathtt{RSA}}$ as follows. On input 1^k, the group generator $\mathtt{Gen}_{\mathtt{RSA}}$ selects two k-bit safe primes $p = 2p' + 1$ and $q = 2q' + 1$, defines $N = pq$ and $m = (p-1)(q-1)/2 = 2p'q'$, selects a generator G of $\mathbb{J}_N$, and outputs $([\mathbb{J}_N], G, \tau = m)$. The $\mathtt{Samp}_{\mathtt{RSA}}$ algorithm, on input $([\mathbb{J}_N], G, m)$, draws a random $x \leftarrow_{\$} \mathbb{Z}_m^*$, computes $X = G^x$, $\tau_x = 1/x \bmod m$, and outputs (X, x, τ_x) (note that we slightly deviate from Definition 4 here since X is not uniformly random in $\langle G \rangle$, but the statistical distance is negligible). Algorithm $\mathtt{Solve}_{\mathtt{RSA}}$, on input $([\mathbb{J}_N], G; X, Y, Z; \tau_x)$, first checks that $X, Y, Z \in \mathbb{J}_N$, that the trapdoor is correct by verifying whether $X^{\tau_x} = G$ (it outputs $\perp$ if this does not hold), and outputs 1 iff $Z^{\tau_x} = Y$.

Definition 5. *We say that the RSA problem is hard for $\mathbb{J}_N$ if for any PPT adversary $\mathcal{A}$, the following probability is negligible:*

$$\Pr\left[([\mathbb{J}_N], G, m) \leftarrow \mathtt{Gen}_{\mathtt{RSA}}(1^k), e \leftarrow_{\$} \mathbb{Z}_m^*, Y \leftarrow_{\$} \mathbb{J}_N, Z \leftarrow \mathcal{A}([\mathbb{J}_N], Y, e) : Z^e = Y\right]$$

Theorem 2. *Assuming that the DDH problem and the RSA problem are hard in $\mathbb{J}_N$ (for N the product of two distinct safe primes), $\mathcal{STDDH}_{\mathtt{RSA}}$ is a static TDDH group with perfect soundness.*

Proof. We show that properties *i*) to *iv*) of Definition 4 hold. Property *i*) holds by assumption that DDH is hard for $\mathbb{J}_N$. We now prove property *ii*). Assume that there is an adversary $\mathcal{A}$ breaking property *ii*). We construct a reduction $\mathcal{R}$ that solves the RSA problem as follows. The reduction is given the product $N = pq$ of two safe primes, a random e coprime with $m = (p-1)(q-1)/2$, and a random challenge $Y \in \mathbb{J}_N$ of which it must compute the e-th root. The reduction draws a random $X \leftarrow_{\$} \mathbb{J}_N$. With overwhelming probability, X is a generator of $\mathbb{J}_N$ since p and q are safe primes. The reduction defines $G = X^e$, and runs $\mathcal{A}$ on input $([\mathbb{J}_N], G; X, Y; e)$. The statistical distance between inputs (G, X, Y) in the simulated experiment and in the real CDH experiment defining property *ii*) is negligible (the difference coming from cases where X does not generate $\mathbb{J}_N$). Moreover, e is the correct trapdoor for X since $G = X^e$ implies $e = 1/x \bmod m$, where $x = \mathtt{Dlog}_G(X)$. Hence, $\mathcal{A}$ returns the correct value $Z = \mathtt{CDH}_G(X, Y)$ with probability negligibly close to its advantage, in which case $Z = Y^x$, which implies $Z^e = Y$, so that Z is indeed the e-th root of Y. The running time of $\mathcal{R}$ is similar to the one of $\mathcal{A}$ and its success probability is negligibly close to the one of $\mathcal{A}$. Property *iii*) is clear, and $\mathcal{STDDH}_{\mathtt{RSA}}$ has perfect soundness since by definition of $\mathtt{Samp}_{\mathtt{RSA}}$, x is coprime to m so that $Z^{\tau_x} = Y \Leftrightarrow Z^{x\tau_x} = Y^x \Leftrightarrow Z = Y^x$. □

4.3 A Construction Based on Signed Quadratic Residues

In this section, we describe a static TDDH group based on signed quadratic residues, whose usefulness for cryptography was first noticed by Hofheinz and Kiltz [31]. This can be seen as a variant of $\mathcal{STDDH}_{\mathtt{RSA}}$ described above, whose security relies on the factoring problem rather than the RSA problem. We first give some definitions.

Definition 6. *Let N be an odd positive integer such that $-1 \in \mathbb{J}_N$. We denote $\mathbb{J}_N^+$ the quotient group $\mathbb{J}_N/\{-1,1\}$. We identify $\mathbb{J}_N^+$ with the set $\mathbb{J}_N \cap [1;(N-1)/2]$ equipped with the group operation $\circ$ defined as $a \circ b = |ab \bmod N|$, where $|x \bmod N|$ is defined as the absolute value of $x \bmod N$ when representing elements of $\mathbb{Z}_N$ as integers in $[-(N-1)/2;(N-1)/2]$.*

To be completely rigorous, the mapping which to an element $\{-x,x\} \in \mathbb{J}_N^+$ associates $|x|$ is a group isomorphism between $\mathbb{J}_N^+$ and $(\mathbb{J}_N \cap [1,(N-1)/2],\circ)$.

Let $N = pq$ be a Blum integer (*i.e.* p and q are two primes such that $p \equiv q \equiv 3 \bmod 4$). Then $-1 \in \mathbb{J}_N$ so that we can define $\mathbb{J}_N^+$, which in this particular case is named the group of signed quadratic residues and denoted $\mathbb{QR}_N^+$.[7] Its order is $\phi(N)/4 = (p-1)(q-1)/4$. The most interesting points to notice about this group is that it is efficiently recognizable (since it is isomorphic to $\mathbb{J}_N \cap [1;(N-1)/2]$), and that the squaring operation is one-to-one so that any $x \in \mathbb{QR}_N^+$ has a unique square root in $\mathbb{QR}_N^+$ (more precisely, for any $x \in \mathbb{QR}_N^+$, either x or $-x \bmod N$ is a quadratic residue mod N, and exactly one corresponding square root is in $\mathbb{QR}_N^+$). Moreover, when $(p-1)/2$ and $(q-1)/2$ are coprime, then $\mathbb{J}_N$ is cyclic and so is $\mathbb{QR}_N^+$. See [31] for proofs of these basic facts.

In the following, we restrict ourselves for simplicity to the special case where N is the product of two distinct safe primes. This implies that N is a Blum integer, and that $(p-1)/2$ and $(q-1)/2$ are coprime so that $\mathbb{QR}_N^+$ is cyclic. Moreover, a uniformly random element of $\mathbb{QR}_N^+$ is a generator with overwhelming probability since the number of generators of $\mathbb{QR}_N^+$ is $\phi((p-1)(q-1)/4) = (p-3)(q-3)/4$.

Let G be a generator of $\mathbb{QR}_N^+$, and denote $m = |\mathbb{QR}_N^+| = (p-1)(q-1)/4$. Let $x \in [0;m-1]$ and $X = G^x$. To build a trapdoor enabling to solve the static DDH problem for (G,X), we use the following idea: the trapdoor will be $t = 2x \pm m$ (computed over $\mathbb{Z}$), *i.e.* the value $2x$ masked with the group order m. Since computing the group order m is as hard as factoring N, t does not reveal x. Now, given a group element $Y = G^y \in \mathbb{G}$, t enables computing $Y^t = G^{2xy} = \mathtt{CDH}_G(X,Y)^2$. This enables testing whether an element Z is a correct solution to the static CDH problem (in other words to solve the static DDH problem) by simply checking whether $Z^2 = Y^t$. However, as we will see, the static CDH problem remains as hard as computing square roots in $\mathbb{QR}_N^+$, which in turn is equivalent to factoring N. For what follows, we will also make the assumption that the DDH problem is hard in $\mathbb{QR}_N^+$. The DDH problem in $\mathbb{QR}_N^+$ can easily shown to be equivalent to the DDH problem in $\mathbb{J}_N$, which as already pointed out is widely believed to be hard when N is the product of two distinct safe primes [2].

We now formally define the static TDDH group $\mathcal{STDDH}_{\mathtt{SQR}}$. For ease of exposition, given an odd integer m, we define the function ξ from $[0;m-1]$ to $\{1,3,5,\ldots,2m-3,2m-1\}$ as:

$$\begin{cases} \xi(x) = 2x+m \text{ if } x \in [0;(m-1)/2] \\ \xi(x) = 2x-m \text{ if } x \in [(m+1)/2;m-1] \end{cases}.$$

[7] We warn that $\mathbb{QR}_N^+$ is *not* equal to $\mathbb{QR}_N/\{-1,1\}$ for the good reason that $-1 \notin \mathbb{QR}_N$ when N is a Blum integer.

$\xi(x)$ is the unique odd integer $t \in [1; 2m-1]$ such that $t = 2x \pm m$.

On input the security parameter 1^k, $\mathtt{Gen}_{\mathtt{SQR}}$ selects two k-bit safe primes $p = 2p'+1$ and $q = 2q'+1$, sets $N = pq$, $m = p'q'$, selects a generator G of the group of signed quadratic residues $\mathbb{QR}_N^+$, and outputs $([\mathbb{QR}_N^+], G, m)$. Algorithm $\mathtt{Samp}_{\mathtt{SQR}}$, on input $([\mathbb{QR}_N^+], G, m)$, selects a random $x \in [0; m-1]$, sets $X = G^x$, $\tau_x = \xi(x)$, and outputs (X, x, τ_x). The algorithm $\mathtt{Solve}_{\mathtt{SQR}}$, on input $([\mathbb{QR}_N^+], G; X, Y, Z; \tau_x)$, first checks that the trapdoor is correct by verifying whether $G^{\tau_x} = X^2$ (it outputs $\perp$ if this does not hold), and outputs 1 iff $Y^{\tau_x} = Z^2$. We now formally prove that this constitutes a static TDDH group under appropriate assumptions (the proof of property *ii*) is reminiscent of the one of Theorem 3.2 in [31]).

Theorem 3. *Under the factoring assumption (for the product of safe primes) and the DDH assumption for $\mathbb{QR}_N^+$, $\mathcal{STDDH}_{\mathtt{SQR}}$ is a static TDDH group with perfect soundness.*

Proof. Deferred to the full version of the paper [45] for reasons of space. □

4.4 Relation to the Strong Diffie-Hellman Problem

We note that in a static TDDH group with perfect soundness, the Strong Diffie-Hellman (SDH) problem [1] is always hard.[8] The SDH problem is to compute $\mathtt{CDH}_G(X, Y)$ given $X, Y \in \langle G \rangle$, and being granted access to a static DDH oracle which on input $(Y', Z') \in \mathbb{G}^2$ outputs 1 iff $(X, Y', Z') \in \mathcal{DH}_G$. Clearly, an adversary $\mathcal{A}$ breaking the SDH problem can be turned into an adversary $\mathcal{B}$ breaking property *ii*) of the static TDDH group ($\mathcal{B}$ can answer queries of $\mathcal{A}$ to the static DDH oracle thanks to the trapdoor τ_x it is given as input). Applying this observation to $\mathcal{STDDH}_{\mathtt{SQR}}$, we recover Theorem 3.2 of [31] which states that SDH is hard in $\mathbb{QR}_N^+$ under the factoring assumption. Hence, the concept of static TDDH group allows to cast the result of [31] in a more general framework. In particular, Theorem 2 directly implies that under the RSA assumption, the SDH problem is hard in $\mathbb{J}_N$, which complements the result of [31].[9] As an immediate consequence of the results of [1,15], we obtain that Hybrid ElGamal encryption over $\mathbb{J}_N$ is IND-CCA2-secure in the ROM under the RSA assumption.

5 Convertible Undeniable Signatures

5.1 Background on Undeniable Signatures

In this section, we show how TDDH groups can be used to build simple and natural undeniable signature schemes with attractive properties such as universal convertibility and delegation. Undeniable signatures, introduced by Chaum

[8] More precisely, the SDH problem is hard for the group generator which only outputs $([\mathbb{G}], G)$.

[9] Note that, by inspection of the proof of property *ii*), this result holds in fact for all RSA moduli N such that $\mathbb{J}_N$ is cyclic, not only the product of safe primes.

and van Antwerpen [12], are signatures that cannot be universally verified: confirmation (or disavowal) of a signature requires the cooperation of the signer (however a signer cannot deny the validity of a correct signature, hence the name undeniable). Later, Boyar *et al.* [5] proposed the refined notion of *convertible* undeniable signature (CUS) scheme, where a mechanism allows the signer to selectively or globally transform undeniable signatures into self-authenticating signatures. The particular scheme proposed in [5] was later broken in [36]. Subsequently, schemes based on usual signatures such as ElGamal [17], Schnorr [37], and RSA [26,24,23] were proposed.

We first recall the basic Chaum and van Antwerpen undeniable signature scheme [12] (in its Full Domain Hash version [41,39]). Let $\mathbb{G}$ be a group, $\mathbb{G}'$ be a cyclic and efficiently recognizable subgroup of $\mathbb{G}$, G be a (certified) generator of $\mathbb{G}'$, and $\boldsymbol{H} : \{0,1\}^* \to \mathbb{G}'$ be a hash function (modeled as a random oracle in security proofs). Assume the DDH problem is hard for $\mathbb{G}'$. The secret and public keys of a user are $x \in \mathbb{Z}_{|\mathbb{G}'|^+}$ and $X = G^x$ respectively. To sign a message $\mu \in \{0,1\}^*$, the signer computes $M = \boldsymbol{H}(\mu) \in \mathbb{G}'$, and $S = M^x$. The signature is S. A signature S on μ is valid iff $(X, \boldsymbol{H}(\mu), S)$ is a valid DH tuple (with respect to G). Since we assumed that the DDH problem is hard, checking the validity of a signature cannot be done without knowledge of x.[10] Hence, the signer must cooperate with the verifier in order to confirm or disavow a purported signature. The confirmation protocol is a proof that $(X, \boldsymbol{H}(\mu), S) \in \mathcal{DH}_G$ (*i.e.* a proof of EDL since G is guaranteed to be a generator of $\mathbb{G}'$), whereas the disavowal protocol is a proof that $(X, \boldsymbol{H}(\mu), S) \notin \mathcal{DH}_G$ (*i.e.* a proof of IDL). The security of this scheme (depending on which type of EDL and IDL proofs are used) has been studied in [41,34,39,33].

The idea to allow efficient universal conversion of signatures is simply to use a Chaum and van Antwerpen undeniable signature with a (static or not) TDDH group, and to use the trapdoor to delegate the ability to verify undeniable signatures and to universally convert them. In the following, we describe the construction using static TDDH groups since the instantiations using constructions of Sections 4.2 and 4.3 are particularly interesting.

5.2 Construction of a CUS scheme from a Static TDDH Group

Let $\mathcal{STDDH} = (\mathtt{Gen}, \mathtt{Samp}, \mathtt{Solve})$ be a static TDDH group with perfect soundness. For this part, we assume that $\mathtt{Gen}$ outputs a tuple $([\mathbb{G}], G, \tau)$ such that $\mathbb{G}$ is cyclic and efficiently recognizable, and G is a generator of $\mathbb{G}$. This assumption is satisfied by $\mathcal{STDDH}_{\mathtt{RSA}}$ and $\mathcal{STDDH}_{\mathtt{SQR}}$. Note that there is not necessarily an efficient way to check that G is indeed a generator; we come back on this issue later. We construct a CUS scheme $\mathcal{CUS}$ as follows (see the full version of the paper [45] for a more formal description). To construct his public/secret key pair, the signer runs $\mathtt{Gen}(1^k)$ to obtain $([\mathbb{G}], G, \tau)$ and then $\mathtt{Samp}([\mathbb{G}], G, \tau)$ to

[10] When the DDH problem is easy in $\mathbb{G}'$, signatures can be universally verified. For example, using bilinear groups (where a pairing can be used to solve the DDH problem), one obtains the Boneh-Lynn-Shacham signature scheme [4].

obtain (X, x, τ_x). It also selects a hash function $\boldsymbol{H} : \{0,1\}^* \to \mathbb{G}$. The public key of the signer is $\mathtt{pk} = ([\mathbb{G}], G, X, \boldsymbol{H})$ and its secret key is $\mathtt{sk} = (x, \tau_x)$. To sign a message $\mu \in \{0,1\}^*$, the signer computes $M = \boldsymbol{H}(\mu)$, and $S = M^x$. The signature is S. The signer can confirm or disavow a signature by running a proof of EDL or IDL respectively with the verifier. To individually convert a signature, the signer produces a NIZK proof of EDL (using an independent hash function $\boldsymbol{H}_{\mathrm{FS}}$ to apply the Fiat-Shamir transform). To universally convert signatures, the signer releases τ_x as universal receipt. A signature S for message μ can then be verified by running $\mathtt{Solve}([\mathbb{G}], G; X, \boldsymbol{H}(\mu), S; \tau_x)$.

Informally, the two main security properties of a CUS scheme (beside soundness of the confirmation and disavowal protocols) are (see the full version of the paper [45] for details):

- *security against existential forgery under chosen-message attacks (EF-CMA-security)*: any PPT attacker, given the receipt for universal verification τ_x, and with access to a signing oracle, can forge a new signature with only negligible probability (note that access to confirmation or disavowal oracles is unnecessary here since the adversary is given the universal receipt τ_x for checking signatures);
- *invisibility under chosen-message attacks (INV-CMA-security)*: any PPT adversary can distinguish a valid signature for a message of its choice from a string sampled uniformly at random from the signature space with only negligible probability. The adversary is granted access to the signing oracle, the confirmation and disavowal protocols, and the individual signature conversion oracle (with the restriction that they cannot be queried on the challenge message).

We stress that formalizing the invisibility notion is quite subtle (many variations appear in the literature [13,17,8,23]), and that the exact property that is achieved is dependent on the nature of the confirmation and disavowal protocols [39,33].

Theorem 4. *When instantiated with a static TDDH group with perfect soundness, and when the confirmation and disavowal protocols are zero-knowledge, the CUS scheme described above is EF-CMA-secure and INV-CMA-secure in the ROM (for $\boldsymbol{H}$ and $\boldsymbol{H}_{\mathrm{FS}}$).*

Proof. Deferred to the full version of the paper [45] for reasons of space. □

Delegation. The ability to verify (confirm or disavow) and convert (either individually or universally) signatures can easily be delegated to a semi-trusted party by simply giving him the trapdoor τ_x. Since the CDH problem remains hard even with the trapdoor, the third party cannot forge signatures on behalf of the signer. It can however prove in zero-knowledge whether a signature is valid or invalid (since it knows the witness τ_x for this). We avoid using the term *designated confirmer signatures* [10] here since this usually refers to schemes (mostly following the "encryption of a signature" paradigm [40,8]) where the signer can create designated confirmer undeniable signatures without having beforehand to transmit some secret information to the confirmer (in our case the trapdoor τ_x).

Instantiation with $\mathcal{STDDH}_{\mathtt{RSA}}$ and $\mathcal{STDDH}_{\mathtt{SQR}}$. The CUS scheme described above can be instantiated with the two static TDDH groups described in Sections 4.2 and 4.3. The schemes obtained this way are similar respectively to the scheme of Gennaro, Rabin, and Krawczyk [26] and Galbraith and Mao [23], with important distinctions though. Both schemes work over $\mathbb{Z}_N^*$, but without explicitly restricting in which subgroup. As a consequence, they cannot be exactly seen as an instantiation of the Chaum and van Antwerpen scheme, and specific confirmation and disavowal protocols were therefore proposed for them (see also [24]). On the contrary, our schemes are strict instantiations of the Chaum and van Antwerpen scheme, and in particular the confirmation and disavowal protocols can use zero-knowledge proofs of EDL and IDL derived from the HVZK protocols described in the full version of the paper [45]. This is conceptually simpler and more efficient (especially for the disavowal protocol).

Certifying Signers Public Keys. Correct key generation is of primary importance in factoring-based undeniable signatures, since a cheating signer may generate its secret/public key in a different way than the one expected by verifiers, which may enable him to confirm invalid signatures or disavow valid ones (see [24]). Hence, the signer, when registering his public key, must prove to the certification authority (CA) that it was generated according to the specification of the static TDDH group generator. We now discuss this issue with respect to $\mathcal{STDDH}_{\mathtt{RSA}}$ and $\mathcal{STDDH}_{\mathtt{SQR}}$. For both schemes, the signer must first prove to the CA that its modulus N is the product of two safe primes. A zero-knowledge protocol for this was proposed by Camenish and Michels [7]. Though expensive, this protocol must be run only once at key registration time. Then, the signer must prove that G is indeed a generator of either $\mathbb{J}_N$ or $\mathbb{QR}_N^+$. The situation is slightly different in the two cases. Denote $N = pq$ with $p = 2p' + 1$ and $q = 2q' + 1$. When p and q are safe primes, then an integer $g \in \mathbb{Z}_N^*$ such that $g^2 \neq 1 \bmod N$ and $\gcd(g^2 - 1, N) = 1$ necessarily has order in $\{p'q', 2p'q'\}$ [26, Lemma 1]. Hence, an ad-hoc solution for ensuring that the element G provided by the signer is a generator of the intended group is as follows. Restrict the scheme to moduli N such that $N \equiv 1 \bmod 8$ and fix $g_0 = 2$ so that $g_0 \in \mathbb{J}_N$. Since an element $g \in \mathbb{QR}_N^+$ generates $\mathbb{QR}_N^+$ exactly when g has multiplicative order modulo N in $\{p'q', 2p'q'\}$, we see by the previous remark that g_0 is always a generator of $\mathbb{QR}_N^+$. Hence, when using $\mathcal{STDDH}_{\mathtt{SQR}}$, we can impose to the signer to always use $G = g_0$. Things are a bit more complicated when using $\mathcal{STDDH}_{\mathtt{RSA}}$, since for an element $g \in \mathbb{Z}_N^*$ with order in $\{p'q', 2p'q'\}$ to generate $\mathbb{J}_N$, one has to check that it is a quadratic non-residue. What we propose for this is that the signer proves in zero-knowledge to the CA whether $g_0 \in \mathbb{QR}_N$ or not [30]. If it is in $\mathbb{QR}_N$, then the signer tries with $g_0 + 1$, $g_0 + 2$, etc. until a quadratic non-residue in $\mathbb{J}_N$ is found. The signer then has to use $G = g_0 + i$ for the smallest $i \geq 0$ such that $g_0 + i \in \mathbb{J}_N \setminus \mathbb{QR}_N$.

As a matter of fact, there seems to be no reason to instantiate the CUS scheme with $\mathcal{STDDH}_{\mathtt{RSA}}$ rather than $\mathcal{STDDH}_{\mathtt{SQR}}$ since both schemes are almost identical, except that the key registration step is simpler for $\mathcal{STDDH}_{\mathtt{SQR}}$.

References

1. Abdalla, M., Bellare, M., Rogaway, P.: The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 143–158. Springer, Heidelberg (2001)
2. Boneh, D.: The Decision Diffie-Hellman Problem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg (1998)
3. 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)
4. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. Journal of Cryptology 17(4), 297–319 (2004)
5. Boyar, J., Chaum, D., Damgård, I., Pedersen, T.P.: Convertible Undeniable Signatures. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 189–205. Springer, Heidelberg (1991)
6. Bresson, E., Catalano, D., Pointcheval, D.: A Simple Public-Key Cryptosystem with a Double Trapdoor Decryption Mechanism and Its Applications. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 37–54. Springer, Heidelberg (2003)
7. Camenisch, J., Michels, M.: Proving in Zero-Knowledge that a Number Is the Product of Two Safe Primes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 107–122. Springer, Heidelberg (1999)
8. Camenisch, J., Michels, M.: Confirmer Signature Schemes Secure against Adaptive Adversaries (Extended Abstract). In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 243–258. Springer, Heidelberg (2000)
9. Camenisch, J., Shoup, V.: Practical Verifiable Encryption and Decryption of Discrete Logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
10. Chaum, D.: Designated Confirmer Signatures. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 86–91. Springer, Heidelberg (1995)
11. Chaum, D., Pedersen, T.P.: Wallet Databases with Observers. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer, Heidelberg (1993)
12. Chaum, D., van Antwerpen, H.: Undeniable Signatures. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 212–216. Springer, Heidelberg (1990)
13. Chaum, D., van Heijst, E., Pfitzmann, B.: Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 470–484. Springer, Heidelberg (1992)
14. Cramer, R., Damgård, I., MacKenzie, P.D.: Efficient Zero-Knowledge Proofs of Knowledge without Intractability Assumptions. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 354–373. Springer, Heidelberg (2000)
15. Cramer, R., Shoup, V.: Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack. SIAM Journal on Computing 33(1), 167–226 (2003)
16. Damgård, I.: Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 418–430. Springer, Heidelberg (2000)
17. Damgård, I., Pedersen, T.P.: New Convertible Undeniable Signature Schemes. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 372–386. Springer, Heidelberg (1996)

18. Dent, A.W., Galbraith, S.D.: Hidden Pairings and Trapdoor DDH Groups. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 436–451. Springer, Heidelberg (2006)
19. Desmedt, Y., Frankel, Y.: Threshold Cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
20. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
21. Frey, G.: How to disguise an elliptic curve (Weil descent). In: Elliptic Curve Cryptography - ECC 1998 (1998), `http://cacr.uwaterloo.ca/conferences/1998/ecc98/frey.ps`
22. Frey, G., Müller, M., Rück, H.-G.: The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. IEEE Transactions on Information Theory 45(5), 1717–1719 (1999)
23. Galbraith, S.D., Mao, W.: Invisibility and Anonymity of Undeniable and Confirmer Signatures. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 80–97. Springer, Heidelberg (2003)
24. Galbraith, S.D., Mao, W., Paterson, K.G.: RSA-Based Undeniable Signatures for General Moduli. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 200–217. Springer, Heidelberg (2002)
25. Gennaro, R.: Multi-trapdoor Commitments and Their Applications to Proofs of Knowledge Secure Under Concurrent Man-in-the-Middle Attacks. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 220–236. Springer, Heidelberg (2004)
26. Gennaro, R., Rabin, T., Krawczyk, H.: RSA-Based Undeniable Signatures. Journal of Cryptology 13(4), 397–416 (2000)
27. Girault, M.: An Identity-Based Identification Scheme Based on Discrete Logarithms Modulo a Composite Number. In: Damgård, I.B. (ed.) EUROCRYPT 1990. LNCS, vol. 473, pp. 481–486. Springer, Heidelberg (1991)
28. Girault, M.: Self-Certified Public Keys. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 490–497. Springer, Heidelberg (1991)
29. Goldreich, O., Sahai, A., Vadhan, S.P.: Honest-Verifier Statistical Zero-Knowledge Equals General Statistical Zero-Knowledge. In: Vitter, J.S. (ed.) Symposium on the Theory of Computing - STOC 1998, pp. 399–408. ACM (1998)
30. Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof Systems. SIAM Journal on Computing 18(1), 186–208 (1989)
31. Hofheinz, D., Kiltz, E.: The Group of Signed Quadratic Residues and Applications. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 637–653. Springer, Heidelberg (2009)
32. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. In: Bosma, W. (ed.) ANTS 2000. LNCS, vol. 1838, pp. 385–394. Springer, Heidelberg (2000)
33. Kurosawa, K., Furukawa, J.: Universally Composable Undeniable Signature. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 524–535. Springer, Heidelberg (2008)
34. Kurosawa, K., Heng, S.-H.: 3-Move Undeniable Signature Scheme. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 181–197. Springer, Heidelberg (2005)
35. Menezes, A., Okamoto, T., Vanstone, S.A.: Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Transactions on Information Theory 39(5), 1639–1646 (1993)

36. Michels, M., Petersen, H., Horster, P.: Breaking and Repairing a Convertible Undeniable Signature Scheme. In: Gong, L., Stearn, J. (eds.) ACM Conference on Computer and Communications Security - CCS 1996, pp. 148–152. ACM (1996)
37. Michels, M., Stadler, M.: Efficient Convertible Undeniable Signature Schemes. In: Selected Areas in Cryptography - SAC 1997, pp. 231–244 (1997)
38. Morales, D.J.M.: An attack on disguised elliptic curves. Journal of Mathematical Cryptology 2(1), 1–8 (2008), http://eprint.iacr.org/2006/469.pdf
39. Ogata, W., Kurosawa, K., Heng, S.-H.: The Security of the FDH Variant of Chaum's Undeniable Signature Scheme. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 328–345. Springer, Heidelberg (2005)
40. Okamoto, T.: Designated Confirmer Signatures and Public-Key Encryption Are Equivalent. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 61–74. Springer, Heidelberg (1994)
41. Okamoto, T., Pointcheval, D.: The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In: Kim, K. (ed.) PKC 2001. LNCS, vol. 1992, pp. 104–118. Springer, Heidelberg (2001)
42. 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)
43. Prabhakaran, M., Xue, R.: Statistically Hiding Sets. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 100–116. Springer, Heidelberg (2009)
44. Schnorr, C.-P.: Efficient Signature Generation by Smart Cards. Journal of Cryptology 4(3), 161–174 (1991)
45. Seurin, Y.: New Constructions and Applications of Trapdoor DDH Groups. Full version of this paper. Available from the author or from http://eprint.iacr.org
46. Tsiounis, Y., Yung, M.: On the Security of ElGamal Based Encryption. In: Imai, H., Zheng, Y. (eds.) PKC 1998. LNCS, vol. 1431, pp. 117–134. Springer, Heidelberg (1998)

Rate-Limited Secure Function Evaluation: Definitions and Constructions

Özgür Dagdelen[1], Payman Mohassel[2], and Daniele Venturi[3]

[1] Technische Universität Darmstadt, Germany
[2] University of Calgary, Canada
[3] Aarhus University, Denmark

Abstract. We introduce the notion of rate-limited secure function evaluation (RL-SFE). Loosely speaking, in an RL-SFE protocol participants can monitor and limit the number of distinct inputs (i.e., *rate*) used by their counterparts in multiple executions of an SFE, in a private and verifiable manner. The need for RL-SFE naturally arises in a variety of scenarios: e.g., it enables service providers to "meter" their customers' usage without compromising their privacy, or can be used to prevent oracle attacks against SFE constructions.

We consider three variants of RL-SFE providing different levels of security. As a stepping stone, we also formalize the notion of commit-first SFE (cf-SFE) wherein parties are committed to their inputs before each SFE execution. We provide compilers for transforming any cf-SFE protocol into each of the three RL-SFE variants. Our compilers are accompanied with simulation-based proofs of security in the standard model and show a clear tradeoff between the level of security offered and the overhead required. Moreover, motivated by the fact that in many client-server applications clients do not keep state, we also describe a general approach for transforming the resulting RL-SFE protocols into *stateless* ones.

As a case study, we take a closer look at the oblivious polynomial evaluation (OPE) protocol of Hazay and Lindell, show that it is commit-first and instantiate efficient rate-limited variants of it.

Keywords: secure function evaluation, foundations, secure metering, oracle attacks, oblivious polynomial evaluation.

1 Introduction

Secure function evaluation (SFE) allows a set of mutually distrustful parties to securely compute a function f of their private inputs. Roughly speaking, SFE protocols guarantee that the function is computed correctly and that the parties will not learn any information from the interaction other than their output and what is inherently leaked from it. Seminal results in SFE show that one can securely compute any functionality [29,30,15,8,2]. There has been a large number of follow-up work improving the security, strengthening adversarial models, and studying efficiency. Recent work on practical SFE has also led to real-world deployments [7,6], and the design and implementation of several SFE frameworks [24,5,12,20,21].

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 461–478, 2013.

In practice, most applications of SFE considered in the literature need to accommodate *multiple executions* of a protocol.[1] Consider a client that searches for multiple patterns in a large text via a secure pattern matching protocol [17,19], searches several keywords in a private database via an oblivious keyword search [27,13], or an individual who needs to run a software diagnostic program, or an intrusion detection system (IDS) to analyze data via an oblivious branching program (OBP) or an automaton evaluation (OAE) protocol [22,28].

Invoking an SFE protocol multiple times raises important practical issues that are outside the scope of standard SFE, and hence are not addressed by the existing solutions. We point out two such issues and introduce *rate-limited* SFE as a means to address them. The reason for the choice of name is that rate-limiting is commonly used in network and web applications to refer to restrictions put on clients' usage (on a per user, or a per IP address basis). In this work we consider similar restrictions on a user's inputs to services that maybe implemented using SFE.

Secure Metering of SFE. Service providers tend to charge their clients according to their level of usage: a location-based service may wish to charge its clients based on the number of locations they use the service from; a database owner based on the number of distinct search queries; an IDS provider based on the number of suspicious files sent for vulnerability analysis. Service providers would be more willing to adopt SFE protocols if it is possible to efficiently enforce such a metering mechanism. The challenge is to do so without compromising the client's privacy, or allowing the server or the client to cheat the metering system.

Oracle attacks. Consider multiple executions of a two-party SFE protocol (such as those mentioned above), where the first party's input stays the same in different executions but the second party's input varies. A malicious second party who "adaptively" uses different inputs in each execution, can gradually learn significant information about the first party's input, and, in the worst case, fully recover it. For instance, consider an oblivious polynomial evaluation (OPE) protocol (e.g., used in oblivious keyword search) wherein the server holds a polynomial p while the client holds a private point x and wants to learn $p(x)$, but cannot learn more than it. Evaluating the polynomial p on sufficiently many points allows a malicious client to interpolate and recover p. A similar attack can be applied to OBP and OAE protocols to learn the private branching program or automaton which may embed propriety information. Learning attacks of this sort are well-understood and have been previously identified as important threats in the context of SFE; they are sometimes referred to as *oracle attacks* since the attacker has black-box access to input/output values from multiple executions (e.g., see the discussion in [1]).

A naïve solution to the problems discussed above is to limit the total number of executions of an SFE protocol, ignoring the actual input values. However, this approach does not provide a satisfactory solution in most scenarios. For

[1] Depending on the application, a subset of the participants may use the same input in different executions.

example, in case of secure metering, fixing an a priori upper bound on the total number of executions would mean charging legitimate clients multiple times for using the service with the same input; a disadvantage for clients who may need to use the same input from multiple devices, or reproduce a result due to communication errors, device upgrades, or perhaps to prove the validity of the outcome to a third-party by re-running the protocol. Similarly, in case of oracle attacks, clients need not be disallowed to use the same input multiple times since querying the same input many times does not yield new information to an attacker.

Rate-limited SFE. A more accurate (and challenging) solution is to *limit* and/or *monitor* the number of distinct inputs used by an SFE participant in multiple executions. Obviously, this should be done in a secure and efficient manner, i.e., a party should not be able to exceed an agreed-upon limit, and its counterpart should not learn any additional information about his private inputs, or impose a lower limit than the one they agreed on.[2] We refer to the number of distinct inputs used by a participant as his *rate*, and call a SFE protocol that monitors/limits this number, a rate-limited SFE.

Of course, achieving RL-SFE is more costly than the naïve solution discussed above. However, at a minimum we require the proposed solution to avoid storing and/or processing the complete transcripts of all previous executions. (We discuss the exact overhead of our solutions in detail below.)

We note that the complementary question of what functions are *unsafe* for use in SFE (leak too much information) has also been studied, e.g., by combining SFE and differential privacy [3,26], or belief tracking techniques [25]. These works are orthogonal to ours, and can potentially be used in conjunction with rate-limited SFE as an *enforcement mechanism*. For instance, the former works can be invoked to negotiate on a function f with a measurable "safeness" from which the rate for each user can be derived. Subsequently, the abidance of this rate can be enforced through our rate-limited SFE.

Our Contribution. Motivated by the discussion above, we initiate the study of rate-limited SFE. For simplicity, in this paper we focus on the two-party case, but point out that the definitions and some of the constructions are easily extendible to the multiparty setting. Our main contributions are as follows.

Definitions. We introduce three definitions for rate-limited secure function evaluation: (i) rate-hiding, (ii) rate-revealing and (iii) pattern-revealing. All our definitions are in the real-world/ideal-world simulation paradigm and are concerned with *multiple* sequential executions of an SFE protocol. They reduce to the standard simulation-based definition (stand alone) for SFE, when applied to a single execution.

In a *rate-hiding* RL-SFE, in each execution, the only information revealed to the parties is whether the agreed-upon rate limit has been exceeded or not.

[2] In fact the problem becomes significantly easier when the parties are assumed to be semi-honest.

In a *rate-revealing* RL-SFE, the parties additionally learn the current rate (i.e., the number of distinct inputs used by their counterpart so far). In a *pattern-revealing* RL-SFE, parties also learn the pattern of occurrences of each other's inputs during the previous executions. These notions provide a useful spectrum of tradeoffs between security and efficiency: our constructions become more efficient as we move to the more relaxed notions, to the extent that *our pattern-revealing transformation essentially adds no overhead* to the underlying SFE protocol.

Commit-first SFE. In order to design rate-limited SFE protocols, we formalize the auxiliary notion of commit-first SFE (cf-SFE). Roughly speaking, a protocol is commit-first if it can be naturally divided into a (i) *committing phase*, where each party becomes committed to its input for the second phase, and (ii) a *function evaluation phase*, where the function f is computed on the inputs committed to in the first phase.[3]

We utilize cf-SFE as a stepping stone to design rate-limited SFE. It turns out that the separation between the input commitment phase and the function evaluation phase facilitates the design of efficient rate-limited SFE. In particular, now a party only needs to provide some evidence of a particular relation between the committed inputs in the first phase. In contrast, if we had not started with a commit-first protocol, such an argument would have involved the complete history of the transcripts for all the previous executions, rendering such an approach impractical.

The related notion of "evaluating on committed inputs" is well-known (e.g. see [15,23]), but we need and put forth a formal (and general) definition for cf-SFE in order to prove our RL-SFE protocols secure. We then show that several existing SFE constructions are either commit-first or can be efficiently transformed into one. Examples include variants of Yao's garbled circuit protocol, the oblivious polynomial evaluation of Hazay and Lindell [16], the private set intersection protocol of Hazay and Nissim [18], and oblivious automaton evaluation of Gennaro *et al.* [14]. We also show that the GMW compiler [15], outputs a commit-first protocol. This is of theoretical interest as it provides a general compiler for transforming a semi-honest SFE protocol into a malicious cf-SFE (and eventually a rate-limited SFE using the compilers in this paper). We elaborate on these cf-SFE instantiations in the full version of this paper [11].

Compilers & Techniques. We design three compilers for transforming a cf-SFE into each of the three variants of RL-SFE discussed above, and provide simulation-based proofs of their security. All our compilers start from a cf-SFE protocol and add a "proof of repeated-input phase" between the committing phase and the function evaluation phase. An exception is our pattern-revealing compiler, where a proof of repeated-input is implicit given that we force the commitments to be deterministic. In our first compiler (rate-hiding), whenever the j-th execution begins, party P_1 first checks whether its input is "fresh" or has

[3] Note that adding input commitments to the beginning of a protocol does not automatically yield a cf-SFE, since parties are not necessarily bound to using the committed inputs in their evaluation.

already been used in a previous run. In the former case, P_1 encrypts the value "1" and, otherwise, the value "0" using a semantically secure public-key encryption scheme (E, D) for which it holds the secret key sk. Denote the resulting ciphertext with c^j. Party P_1 forwards to P_2 a ZK proof of the following statement:

$$(\text{"committed to old input"} \wedge \mathsf{E}(0) \vee (\text{"committed to new input"} \wedge \mathsf{E}(1) \wedge \text{"}\textstyle\sum_{i \leq j} \mathsf{D}(\mathsf{sk}, c^i) \leq \text{rate"}).$$

Intuitively, the proof above only leaks the fact that the rate is not exceeded in the current execution, but nothing else. In order to generate this proof (resp. verify the proof generated by the counterpart), P_1 needs to store all the commitments and ciphertexts sent to (resp. received from) P_2 in previous executions.

For our second compiler (rate-revealing), we can do without the encryptions. Parties can instead prove a simpler statement giving evidence that the current (committed) input corresponds to one of the commitments the other party received earlier. Clearly, this approach reveals the current rate, but as we prove nothing more.

Finally, our third compiler (pattern-revealing) exploits a PRF to generate the randomness used in the committing phase of the underlying cf-SFE protocol. In this way, the commitment becomes deterministic (given the input), allowing the other party to check whether the current input has already been used and *in which runs*. This approach discloses the pattern of inputs used by the parties; on the other hand, it is extremely efficient adding little computational overhead (merely one invocation of a PRF) to the original cf-SFE protocol.

Making RL-SFE stateless. The above compilers suffer from the limitation that the parties need to keep a state which grows linearly in the total number of executions of the underlying SFE protocol. In many applications, clients do not keep state (and outsource this task to the servers), either due to lack of resources or because they need to use the service from multiple locations/devices. We show a general approach for transforming the stateful RL-SFE protocols generated above into *stateless* ones. Here, the client keeps merely a small secret (whose size is independent of the total number of executions), but is still able to prevent cheating by a malicious server, and preserve privacy of his inputs. At a high level, the transformation requires the client to store its *authenticated* (MACed) state information on the server side and retrieve/verify/update it on-the-fly as needed. We show how to apply this transformation to our rate-revealing compiler to obtain a stateless variant and prove its security. A similar technique can be applied to our rate-hiding compiler. Our pattern-revealing compiler is already stateless (client only needs to store a PRF key) for the party who plays the role of the client.

Case Study. We take a closer look at the oblivious polynomial evaluation protocol of Hazay and Lindell [16]. Their protocol is secure against malicious adversaries. We show that it is also a commit-first OPE, by observing that a homomorphic encryption of the parties' inputs together with ZK proofs of their validity, can be interpreted as a commitment to their inputs. This immediately

yields an efficient pattern-revealing RL-SFE for the OPE problem, based on the compiler we design. We also provide an efficient rate-hiding and rate-revealing RL-OPE by instantiating the ZK proofs for membership in the necessary languages, efficiently.

Roadmap. We start introducing notations and our model for commit-first SFE in Section 2 and 3. In Section 4 we give the definition of rate-limited SFE. A fortaste of our rate-hiding, rate-revealing and pattern-revealing compilers are given in Section 5, whereas Section 6 describes the stateless version of the rate-revealing compiler. Finally, Section 7 deals with concrete instantiations for the case of OPE.

2 Preliminaries

Throughout the paper, we denote the security parameter by λ. A function $negl(\lambda)$ is negligible in λ (or just negligible) if it decreases faster than the inverse of every polynomial in λ. A machine is said to run in polynomial-time if its number of steps is polynomial in the security parameter.

Let $X = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $Y = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ be two distribution ensembles. We say X and Y are computationally indistinguishable (and we write $X \equiv_c Y$) if for every non-uniform polynomial-time adversary $\mathcal{A}$ there exists a negligible function $negl$ such that $|\Pr[\mathcal{A}(X) = 1] - \Pr[\mathcal{A}(Y) = 1]| \leq negl(\lambda)$. Note that all our security statements can be straightforwardly proven for uniform polynomial-time adversaries, as well.

If x is a string, $|x|$ denotes the length of x. Vectors are denoted boldface; given vector $\mathbf{x}$, we write $\mathbf{x}[j]$ for the j-th element of $\mathbf{x}$. If $\mathcal{X}$ is a set, $\#\mathcal{X}$ represents the number of elements in $\mathcal{X}$. When x is chosen randomly in $\mathcal{X}$, we write $x \leftarrow \mathcal{X}$. When $\mathcal{A}$ is an algorithm, $y \leftarrow \mathcal{A}(x)$ denotes a run of $\mathcal{A}$ on input x and output y; if $\mathcal{A}$ is randomized, then y is a random variable and $\mathcal{A}(x; r)$ denotes a run of $\mathcal{A}$ on input x and random coins r.

Our compilers make use of standard cryptographic primitives. Due to space limitations, we assume familiarity of these primitives and define them formally in the full version of this paper [11].

3 Commit-First SFE

In this section, we formally define the notion of *commit-first secure function evaluation* (cf-SFE). Our three compilers $\Psi_{\mathsf{RH}}, \Psi_{\mathsf{RR}}$ and Ψ_{PR} for designing rate-limited SFE, leverage commit-first protocols as a building block. We call a protocol π commit-first if it can be naturally divided into two phases. In the first phase (committing phase), both parties P_1 and P_2 become committed to their inputs. At the end of this phase, no information about the parties' inputs is revealed (the hiding property), and neither party can use a different input than what it is committed to in the remainder of the protocol (the binding property). In the

second phase (function evaluation phase), the function f will be computed on the inputs committed to in the last phase.

We now specify the two separate phases. Consider a polynomial-time functionality $f = (f_1, f_2)$ with $f_i : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$. Then, a cf-SFE protocol $\pi = (\pi_1, \pi_2)$ for evaluating f on parties' inputs x_1 and x_2 proceeds as follows.

Committing Phase: Parties P_1 and P_2 execute π_1 which is defined by the functionality $((x_1, r_1), (x_2, r_2)) \mapsto (\mathsf{C}_2(x_2, r_2), \mathsf{C}_1(x_1, r_1))$. Note that the commitment schemes $\mathsf{C}_1, \mathsf{C}_2$ can be arbitrary schemes (often different for each cf-SFE protocol), as long as they satisfy the hiding and the binding properties required.

Function Evaluation Phase: Afterwards, P_1 and P_2 execute π_2 on the same inputs as in the committing phase; π_2 is defined by the functionality $((x_1, \mathsf{C}_2(x_2)), (x_2, \mathsf{C}_1(x_1))) \mapsto (f_1(x_1, x_2), f_2(x_1, x_2))$. Note that P_1 and P_2, can use their state information from the previous phase in the function evaluation phase, too.

Next, we formalize the security definition for a cf-SFE using the real/ideal world simulation paradigm.

The real world. In each execution, a non-uniform adversary $\mathcal{A}$ following an arbitrary polynomial-time strategy can send messages in place of the corrupted parties (whereas the honest parties continue to follow π). Let $i \in \{1, 2\}$ be the index of the corrupted party. A $\mathsf{real\ execution}$ of $\pi = (\pi_1, \pi_2)$ on inputs (x_1, x_2), auxiliary input z to $\mathcal{A}$ and the security parameter λ, denoted by $\text{REAL}^{\mathsf{cf-SFE}}_{\pi,\mathcal{A}(z),i}(x_1, x_2, \lambda)$ is defined as the output of the honest party and the adversary upon execution of π.

The ideal world. Let $i \in \{1, 2\}$ be the index of the corrupted party. We define the ideal world in two steps. During the ideal execution, the honest party sends its input x_{3-i}, and a uniformly random string r_{3-i} used by the commitment scheme, to the trusted party. Party P_i which is controlled by the ideal adversary $\mathcal{S}$, called the simulator, may either abort (sending a special symbol $\perp$) or send input x'_i, and an arbitrary randomness r'_i (not necessarily uniform) chosen based on the auxiliary input z, and P_i's original input x_i. Denote by $((x'_1, r'_1), (x'_2, r'_2))$ the values received by the trusted party. If the trusted party receives $\perp$, the value $\perp$ is forwarded to both P_1 and P_2 and the ideal execution terminates; else the trusted party computes $\gamma_1 = \mathsf{C}_1(x'_1; r'_1)$ and $\gamma_2 = \mathsf{C}_2(x'_2; r'_2)$, respectively. The TTP sends γ_{3-i} to $\mathcal{S}$, which can either continue or abort by sending $\perp$ to the TTP. In case of an abort, the TTP sends $\perp$ to the honest party; otherwise, it sends γ_i.

In the second phase, the honest party continues the ideal execution by sending to the TTP a $\mathsf{continue}$ flag, or abort by sending $\perp$. $\mathcal{S}$ sends either $\perp$ or $\mathsf{continue}$ based on the auxiliary input z, P_i's original input, and the value γ_{3-i}. If the trusted party receives $\perp$, the value $\perp$ is forwarded to both P_1 and P_2 and the ideal execution terminates; else the trusted party computes $y_1 = f_1(x'_1, x'_2)$ (resp. $y_2 = f_2(x'_1, x'_2)$).

The TTP sends y_i to $\mathcal{S}$. At this point, $\mathcal{S}$ can decide whether the trusted party should continue, and thus send the output y_{3-i} to the honest party, or halt, in which case the honest party receives $\perp$. The honest party outputs the received value. The simulator $\mathcal{S}$ outputs an arbitrary polynomial-time computable function of (z, x_i, y_i).

The ideal execution of f on inputs (x_1, x_2), auxiliary input z to $\mathcal{S}$ and security parameter λ, denoted by $\text{IDEAL}^{\mathsf{cf-SFE}}_{f,\mathsf{C}_1,\mathsf{C}_2,\mathcal{S}(z),i}(x_1, x_2, \lambda)$ is defined as the output of the honest party and the simulator.

Emulating the ideal world. We define a secure commit-first protocol π as follows:

Definition 1 (Commit-first Protocols). *Let π and f be as above. We say that π is* a commit-first protocol for computing $f = (f_1, f_2)$ in the presence of malicious adversaries with abort *if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ in the real world there exists a non-uniform probabilistic polynomial-time simulator $\mathcal{S}$ in the ideal world, such that for every $i \in \{1, 2\}$,*

$$\left\{\text{REAL}^{\mathsf{cf-SFE}}_{\pi,\mathcal{A}(z),i}(x_1, x_2, \lambda)\right\}_{x_1,x_2,z,\lambda} \equiv_c \left\{\text{IDEAL}^{\mathsf{cf-SFE}}_{f,\mathsf{C}_1,\mathsf{C}_2,\mathcal{S}(z),i}(x_1, x_2, \lambda)\right\}_{x_1,x_2,z,\lambda}$$

where $x_1, x_2, z \in \{0,1\}^*$ and $\lambda \in \mathbb{N}$.

4 Rate-Limited Secure Function Evaluation

In this section, we introduce three notions for rate-limited secure function evaluation (RL-SFE). In particular, we augment the standard notion of two-party SFE by allowing each player to monitor and/or limit, the number of distinct inputs (the *rate*) the other player uses in multiple executions. The idea is that each party can abort the protocol if the number of distinct inputs used in the previous executions raises above a threshold $\mathit{ri} \in \mathbb{N}$. We call this threshold the *rate limit*, i.e. the maximum number of allowable executions with distinct inputs.

Naturally, our security definitions for RL-SFE are concerned with *multiple* executions of an SFE protocol and reduce to the standard simulation-based definition (stand alone) for SFE, when applied to a single run. We call a sequence of executions of a protocol π $(\mathit{ri}_1, \mathit{ri}_2)$-limited if party P_1 (resp. P_2) can use at most ri_1 (resp. ri_2) distinct inputs in the executions. In this work, we assume that the executions take place *sequentially*, i.e. one execution after the other. We emphasize that the inputs used by the parties in each execution can depend on the transcripts of the previous executions, but honest parties will always use fresh randomness in their computation.

We provide three security definitions for rate-limited SFE: (i) rate-hiding, (ii) rate-revealing and (iii) pattern-revealing. In a *rate-hiding* RL-SFE, at the end of each execution, the only information revealed to the parties (besides the output from the function being computed), is whether the agreed-upon rate limit (threshold) has been exceeded or not, but nothing else. In a *rate-revealing* RL-SFE, in addition to the above, parties also learn the current rates

(i.e., the number of distinct inputs used by their counterpart so far). Finally, in a *pattern-revealing* RL-SFE, parties further learn the pattern of occurrences of each others' inputs in the previous executions. In particular, each party learns which executions were invoked by the same input and which ones used different ones, but nothing else.

High Level Description. Let $f = (f_1, f_2)$ be a pair of polynomial-time functions such that f_i is of type $f_i : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$. Consider an arbitrary number ℓ of sequential executions of two-party SFE protocol π for evaluating f on parties' inputs. During the j-th execution, party P_i has input x_i^j and should learn $y_i^j = f_i(x_1^j, x_2^j)$. We will define rate-limited SFE in the general case where *both* parties are allowed to change their input in each execution. The case of oracle attacks and secure metering, where one party's input is fixed and the other party's input changes, are found as a special case. (In the case of secure metering one can also think that a change in the service provider's input reflects a software update.)

In the ideal world, during the j-th execution, each party sends its input to a trusted authority. The following is then performed for both $i = 1, 2$. The trusted party checks whether value x_i^j was already sent in a previous execution; in case it was not, a new entry (x_i^j, j) is stored in an initially empty set $\mathcal{X}_i$. Otherwise, the index $j' < j$ corresponding to such input is recovered. Whenever $\#\mathcal{X}_i$ exceeds $\wp_i$ the trusted party aborts. Otherwise, the current outputs $y_i^j = f_i(x_1^j, x_2^j)$ are computed. Finally: (i) in the rate-hiding definition party P_i learns only y_i^j; (ii) in the rate-revealing definition party P_i learns also $\#\mathcal{X}_{3-i}$, i.e. the (partial) total number of distinct inputs used by P_{3-i} until the j-th execution; (iii) in the pattern-revealing definition party P_i learns j', i.e. the index corresponding to the query where x_i^j was asked for the first time. Note that if the rate is exceeded, the trusted party aborts here, but, equivalently, we could simply ignore this execution and still allow to query previous inputs in subsequent executions.

We formalize the above intuitive security notions for all three flavors using the simulation-based ideal/real world paradigm. We first review the real execution which all three notions share.

The real world. In each execution, a non-uniform adversary $\mathcal{A}$ following an arbitrary polynomial-time strategy can send messages in place of the corrupted party (whereas the honest party continues to follow π). Let $i \in \{1, 2\}$ be the index of the corrupted party. The j-th real execution of π on inputs (x_1^j, x_2^j), auxiliary input z^j to $\mathcal{A}$ and security parameter λ, denoted by $\text{REAL}^{\wp}_{\pi,\mathcal{A}(z^j),i}(x_1^j, x_2^j, \lambda)_j$ is defined as the output of the honest party and the adversary in the j-th real execution of π. We denote by $\text{REAL}^{\wp}_{\pi,\mathcal{A}(\mathbf{z}),i}(\mathbf{x}_1, \mathbf{x}_2, \lambda, \ell)$ the accumulative distribution at the end of the ℓ-th execution, i.e.,

$$\text{REAL}^{\wp}_{\pi,\mathcal{A}(\mathbf{z}),i}(\mathbf{x}_1, \mathbf{x}_2, \lambda, \ell) = \text{REAL}^{\wp}_{\pi,\mathcal{A}(z^1),i}(x_1^1, x_2^1, \lambda)_1, \ldots, \text{REAL}^{\wp}_{\pi,\mathcal{A}(z^\ell),i}(x_1^\ell, x_2^\ell, \lambda)_\ell$$

where $\mathbf{x}_1 = (x_1^1, \ldots, x_1^\ell)$, $\mathbf{x}_2 = (x_2^1, \ldots, x_2^\ell)$ and $\mathbf{z} = (z^1, \ldots, z^\ell)$.

The ideal world. The trusted party keeps two sets $\mathcal{X}_1$, and $\mathcal{X}_2$ initially set to $\emptyset$. Let $i \in \{1, 2\}$ be the index of the corrupted party. During the j-th ideal

execution, the honest party sends its input to the trusted party. Party P_i, which is controlled by the ideal adversary $\mathcal{S}$, called the simulator, may either abort (sending a special symbol $\perp$) or send input $x_i'^j$ to the trusted party chosen based on the auxiliary input z^j, P_i's original input x_i^j, and its view in the previous $j-1$ ideal executions. Denote with $(x_1'^j, x_2'^j)$ the values received by the trusted party (note that if $i = 2$ then $x_1'^j = x_1^j$).

If the trusted party receives $\perp$, the value $\perp$ is forwarded to both P_1 and P_2 and the ideal execution terminates; else when the trusted party receives $x_1'^j$ as the first party's input, it checks whether an entry $(x_1'^j, j') \in \mathcal{X}_1$ already exists; if so, it sets $J_1 = j'$. Otherwise, it creates a new entry $(x_1'^j, j)$, adds it to $\mathcal{X}_1$, and sets $J_1 = j$. An identical procedure is applied to input of the second party $x_2'^j$ to determine an index J_2. At the end of the j-th ideal execution if $\sigma_1 := \#\mathcal{X}_1 \geq \nu_1$ *or* $\sigma_2 := \#\mathcal{X}_2 > \nu_2$, the value $\perp$ is forwarded to both P_1 and P_2 and the ideal execution terminates. Otherwise, the pair $(y_1^j, y_2^j) = (f_1(x_1'^j, x_2'^j), f_2(x_1'^j, x_2'^j))$ is computed.

At this point, the ideal executions will be different depending on the variant of RL-SFE being considered.

Rate-Hiding. The trusted party forwards to the malicious party P_i the output y_i^j. At this point, $\mathcal{S}$ can decide whether the trusted party should continue, and thus send the pair y_{3-i} to the honest party, or halt, in which case the honest party receives $\perp$.

Rate-Revealing. The trusted party forwards to the malicious party P_i the pair (y_i^j, σ_{3-i}). At this point, $\mathcal{S}$ can decide whether the trusted party should continue, and thus send the pair (y_{3-i}^j, σ_i) to the honest party, or halt, in which case the honest party receives $\perp$.

Pattern-Revealing. The trusted party forwards to the malicious party P_i the pair (y_i^j, J_{3-i}). The integer $1 \leq J_{3-i} \leq j$ represents the index of the first execution where the input x_{3-i}^j has been used. At this point, $\mathcal{S}$ can decide whether the trusted party should continue, and thus send the pair (y_{3-i}^j, J_i) to the honest party, or halt, in which case the honest party receives $\perp$.

The honest party outputs the received value. The simulator $\mathcal{S}$ outputs an arbitrary polynomial-time computable function of (z^j, x_i^j, y_i^j).

The j-th ideal execution of f on inputs (x_1^j, x_2^j), auxiliary input z^j to $\mathcal{S}$ and security parameter λ, denoted by $\text{IDEAL}^{\nu-\mathsf{X}}_{f,\mathcal{S}(z^j),i}(x_1^j, x_2^j, \lambda)_j$ is defined as the output of the honest party and the simulator in the above j-th ideal execution. Here, $\mathsf{X} \in \{\mathsf{RH}, \mathsf{RR}, \mathsf{PR}\}$ determines the flavor of rate-limited SFE. We denote by $\text{IDEAL}^{\nu-\mathsf{X}}_{f,\mathcal{S}(\mathbf{z}),i}(\mathbf{x}_1, \mathbf{x}_2, \lambda, \ell)$ the accumulative distribution at the end of the ℓ-th execution, i.e.,

$$\text{IDEAL}^{\nu-\mathsf{X}}_{f,\mathcal{S}(\mathbf{z}),i}(\mathbf{x}_1, \mathbf{x}_2, \lambda, \ell) = \text{IDEAL}^{\nu-\mathsf{X}}_{f,\mathcal{S}(z^1),i}(x_1^1, x_2^1, \lambda)_1, \ldots, \text{IDEAL}^{\nu-\mathsf{X}}_{f,\mathcal{S}(z^\ell),i}(x_1^\ell, x_2^\ell, \lambda)_\ell$$

where $\mathbf{x}_1 = (x_1^1, \ldots, x_1^\ell)$, $\mathbf{x}_2 = (x_2^1, \ldots, x_2^\ell)$ and $\mathbf{z} = (z^1, \ldots, z^\ell)$.

Emulating the ideal world. Roughly speaking, ℓ sequential executions of a protocol π are secure under the rate limit $\nu = (\nu_1, \nu_2)$ if the real executions

can be simulated in the above mentioned ideal world. More formally, we define a secure $(\mathcal{R}_1, \mathcal{R}_2)$-limited protocol π as follows:

Definition 2 (RL-SFE). *Let π and f be as above, and consider $\ell = poly(\lambda)$ sequential executions of protocol π. For $\mathsf{X} \in \{\mathsf{RH}, \mathsf{RR}, \mathsf{PR}\}$, we say protocol π is a* secure X $\mathcal{R}$-limited SFE *for computing $f = (f_1, f_2)$,* in presence of malicious adversaries with abort *with $\mathcal{R} = (\mathcal{R}_1, \mathcal{R}_2)$, if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ there exists a non-uniform probabilistic polynomial-time simulator $\mathcal{S}$, such that for every $i \in \{1, 2\}$,*

$$\left\{\mathrm{REAL}^{\mathcal{R}}_{\pi,\mathcal{A}(\mathbf{z}),i}(\mathbf{x}_1, \mathbf{x}_2, \lambda, \ell)\right\}_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{z},\lambda} \equiv_c \left\{\mathrm{IDEAL}^{\mathcal{R}-\mathsf{X}}_{f,\mathcal{S}(\mathbf{z}),i}(\mathbf{x}_1, \mathbf{x}_2, \lambda, \ell)\right\}_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{z},\lambda}$$

where $\mathbf{x}_1, \mathbf{x}_2, \mathbf{z} \in (\{0,1\}^*)^\ell$, such that $|\mathbf{x}_1[j]| = |\mathbf{x}_2[j]|$ for all j, and $\lambda \in \mathbb{N}$.

It is easy to see that the rate-hiding notion is strictly stronger than the rate-revealing notion, which in turn is strictly stronger than the pattern-revealing notion. A proof to this fact can be found in the full version [11].

5 Compilers for Rate-Limited SFE

In this section, we introduce our three compilers to transform an arbitrary (two-party) cf-SFE protocol into a *rate-limited* protocol for the same functionality.

Our first compiler Ψ_{RH} achieves the notion of rate-hiding RL-SFE through the use of general ZK proofs and (additively) homomorphic public key encryption. Our second compiler Ψ_{RR} achieves the notion of rate-revealing RL-SFE and is more efficient in that it needs to prove a simpler statement and does not rely on homomorphic encryption. Our last compiler Ψ_{PR} introduces essentially no overhead and avoids the use of general ZK proofs, yielding our third notion of pattern-revealing RL-SFE.

Let π_f be a two-party (single-run) commit-first protocol for secure function evaluation of a function $f = (f_1, f_2)$ (cf. Definition 1). Our compilers get as input (a description of) π_f, together with the rate $\mathcal{R} = (\mathcal{R}_1, \mathcal{R}_2)$, and the number of executions ℓ, and output (a description of) $\widehat{\pi}_f \leftarrow \Psi(\pi_f, \mathcal{R}, \ell)$. The compilers are functionality preserving, meaning that protocol $\widehat{\pi}$ repeatedly computes the same functionality f.

Due to space limitations, in this section we only provide a full description and analysis for the rate-revealing compiler. The other two compilers (rate-hiding Ψ_{RH}, and pattern-revealing Ψ_{PR}) are only covered at a high level here. The complete descriptions and analyses are given in the full version of this paper [11].

5.1 A Rate-Hiding Compiler

The Overview. We naturally divide the cf-SFE protocol into a committing phase and a function evaluation phase and introduce a new phase in between where P_1 and P_2 convince each other that they have not exceeded the rate limit.

The latter step is achieved as follows. Whenever one of the parties is going to use a "fresh" input, it transmits an encryption of "1" to the other party; otherwise, it sends an encryption of "0". The encryptions are obtained using a CPA-secure (homomorphic) PKE scheme $(\tilde{\mathsf{G}}, \tilde{\mathsf{E}}, \tilde{\mathsf{D}})$. Then, the party proves in ZK that "the last commitment transmitted hides an already used input *and* it encrypted 0, *or* the last commitment transmitted hides a fresh input *and* it encrypted 1 *and* the sum of all the plaintexts, encrypted until now, does not exceed the rate". A successful verification of this proof convinces the other party that the rate is not exceeded, leaking nothing more than this. We instantiate such ZK proofs for the OPE problem in Section 7. Notice that to generate such a proof each party needs to store all the ciphertexts transmitted to the other player, together with all the inputs and randomness used to generate the previous commitments. On the other hand, to verify the other party's proof, one needs to store the ciphertexts and the commitments received in all earlier executions. The remainder of the messages exchanged during each execution, however, can be discarded.

Theorem 1. *Let π_f be a cf-SFE securely evaluating function f and $(\tilde{\mathsf{G}}, \tilde{\mathsf{E}}, \tilde{\mathsf{D}})$ be a CPA-secure PKE scheme. Then $\widehat{\pi}_f \leftarrow \Psi_{\mathsf{RH}}(\pi_f, \Lambda_1, \Lambda_2, \ell)$ is a secure* rate-hiding *(Λ_1, Λ_2)-limited protocol for the function f.*

5.2 A Rate-Revealing Compiler

The Overview. Once again, we divide the cf-SFE protocol into a committing phase and a function evaluation phase and introduce a new phase in between where P_1 and P_2 convince each other that the current input has already been used in a previous execution. Note that the parties need to maintain a state variable Γ collecting the input commitments sent and received in all earlier executions. During the j-th execution, given a list of input commitments (and the corresponding inputs and randomness) for all the previous executions, party P_i can prove in ZK that the input commitment generated in the current execution is for the same value as one of the commitments collected previously. Party P_{3-i} also needs to collect the same set of commitments in order to verify the statement proven by P_i. The remainder of the messages exchanged during each execution, however, can be discarded. We note that while in general efficient ZK proofs of repeated inputs might be hard to find, for discrete-logarithm based statements, there exist efficient techniques for proving such statements. We refer the reader to the full version for more details. We also instantiate such ZK proofs for the OPE problem in Section 7. A complete description of the compiler is depicted in Figure 1. We prove the following result:

Theorem 2. *Let π_f be a cf-SFE securely evaluating function f. Then $\widehat{\pi}_f \leftarrow \Psi_{\mathsf{RR}}(\pi_f, \Lambda_1, \Lambda_2, \ell)$ is a secure* rate-revealing *(Λ_1, Λ_2)-limited protocol for f.*

5.3 A Pattern-Revealing Compiler

In this section, we introduce a more efficient compiler Ψ_{PR} for designing rate-limited SFE. Given as input a cf-SFE protocol, our compiler Ψ_{PR} outputs a

Rate-Revealing Compiler Ψ_{RR}:

Given as input a commit-first protocol $\pi_f = (\pi_f^1, \pi_f^2)$, a rate $\rho = (\rho_1, \rho_2)$, and a number of executions ℓ, the compiled protocol $\widehat{\pi}_f$ is made of three phases, described below. Party P_1 and P_2 keep the state variables $\Gamma_1, \Gamma_2 := \emptyset$, respectively. For each execution $j \in [\ell]$, $\widehat{\pi}_f$ proceeds as follows.

Committing Phase: Parties P_1 and P_2, holding respectively inputs x_1^j and x_2^j, run the protocol π_f^1 yielding the output $(\gamma_2^j = \mathsf{C}(\mathsf{pk}_2, x_2^j; r_2^j), \gamma_1^j = \mathsf{C}(\mathsf{pk}_1, x_1^j; r_1^j))$.

Proof of Repeated-Input Phase: Consider the following language: $\mathcal{L}_i = \{\gamma \in \mathcal{C}_{\mathsf{pk}_i} : \exists (x, r, r') \text{ s.t. } \gamma = \mathsf{C}(\mathsf{pk}_i, x; r) \wedge \mathsf{C}(\mathsf{pk}_i, x; r') \in \Gamma_{3-i}\}$, and let $(\mathsf{Pr}, \mathsf{Vr})$ be a ZK proof system for $\mathcal{L}_i$. The following is executed for all $i \in \{1, 2\}$. When the input x_i^j of party P_i is *not* fresh—i.e., it has already been used in a previous execution—P_i plays the role of the prover in $(\mathsf{Pr}, \mathsf{Vr})$. When the input x_i^j is fresh, P_i just forwards the empty string ε. Also, party P_i plays the role of the verifier in $(\mathsf{Pr}, \mathsf{Vr})$ (with P_{3-i} being the prover and $\mathcal{L}_{3-i}$ being the underlying language). If the value ε is received or if the verification of the proof fails, P_i updates the rate by letting $\rho_i := \rho_i - 1$ and the state by letting $\Gamma_i := \Gamma_i \cup \{\gamma_{3-i}^j\}$. Otherwise, if the verification is successful, the state and rate information will not be modified.

Protocol Emulation Phase: P_1 and P_2 run the protocol π_f^2 on the same inputs as in the committing phase, yielding the output (y_1^j, y_2^j).

Fig. 1. A compiler for rate-revealing rate-limited SFE

weaker form of rate-limited SFE where each party not only learns the current rate for its counterpart during each execution, but also the pattern of already used inputs. The main advantage is that this new compiler adds very little overhead to the original cf-SFE.

The Overview. The idea is as follows. Besides their input, each party also stores a secret key for a PRF (a different key for each party). Before invoking the commit-first SFE protocol, each player generates the randomness it needs for the committing phase by applying the PRF on the *chosen input* for this execution. With this modification in place, the committing phase for each party becomes deterministic. If a party uses the same input in two executions, the two commitments its counterpart receives will be identical. As a result, to prove a repeated-input, each party can compare the commitment for the current execution with those used in the previous ones, and determine if the input is new or being repeated (hence also revealing the pattern). Note that the commitments still provide the required hiding and binding properties. The only overhead imposed by this compiler is the application of a PRF to generate the randomness for the committing phase.

Theorem 3. *Let π_f be a cf-SFE securely evaluating function f. Then $\widehat{\pi}_f \leftarrow \Psi_{\mathsf{PR}}(\pi_f, \pi_1, \pi_2, \ell)$ is a secure* pattern-revealing *(π_1, π_2)-limited SFE for f.*

6 Making the Compilers Stateless

One drawback of the compilers described in the previous section is that both P_1 and P_2 need to maintain state. To some extent, this assumption is necessary. It is not too hard to see that RL-SFE is impossible to achieve if neither party is keeping any information about the previous executions (we omit a formal

argument of this statement). However, as discussed earlier, in many natural client-server applications of SFE in the real world, it is reasonable to assume that the servers keep state, while the clients typically do not.

In this section, we show how to modify the compilers from Section 5 in such a way that only one of the parties needs to keep state. Our solution is efficient and works for all three compilers we discussed earlier. Throughout this section, we assume P_1 is the client and P_2 is the server. Server P_2 receives no output (as it is usually the case in the client-server setting) and wants to enforce the rate limit ρ for the client. Although P_1 does not maintain any state, it needs to make sure that P_2 handles the rate, honestly. On the other hand, the server also needs to be convinced that the client is not cheating, by exceeding the rate limit ρ.

The Overview. Note that in the stateful versions of our compilers, P_1 needs to keep state in order to generate a ZK proof of repeated inputs, and verify the corresponding statement being proven by P_2. Since we are only enforcing the rate for P_1, we can eliminate the latter ZK proofs, and focus on the first one. Although our approach is general, for the sake of simplicity, we describe it in relation to our rate-revealing compiler from Section 5.2. The same idea can be applied to make the other compilers stateless. The basic idea is simple: We ask the server to store the list of all the commitments previously sent by P_1 sends the list to the client, during each run. For this simple approach to work, we need to address several important issues:

- For the client to learn the current rate and the previously queried inputs before each execution, it needs to store these values on the server side in a secure way. This can be easily addressed by having P_1 encrypt the message and randomness for each commitment (using a symmetric-key encryption) and send it along with the commitment itself. P_1 will just keep the private key for the encryption scheme.
- The client needs to verify that the list of commitments it receives from the server are the original commitments it sent in the previous executions. To do so, in each run P_1 computes a MAC ϕ of the string obtained by hashing all the commitments (i.e., the concatenation of the list it obtains from the server and the one it creates in the current execution) and sends it to the server.[4] In each execution, it requests this MAC, the list of commitments along with the ciphertext storing the inputs and random coins from the server. Due to the unforgeability of the MAC, the server will only be able to use a correct list of commitments, previously issued and MACed by the client itself.
- It may seem that the above solution still allows the server to cheat and only send a subset of the commitment list along with a tag generated for that subset in one of the earlier executions, to the client. This would potentially make the input to the current execution look "new" and allow the server to decrease the rate. The client would not be able to detect this attack since it does not keep state and does not know the total number of commitments.

[4] To save on computation, one could let the client obtain the previous hash value and compute the new one via *incremental* hashing [9,4].

However, a more careful inspection shows that the above does not really constitute an attack. In fact, the tag ϕ already *binds* the current rate to the current list of commitments, and prevents the server from decreasing the rate in this fashion. In particular, it is hard for the server to cook-up a state such that the verification of the tag is successful, and the client will think its rate is already exceeded when it is not. Essentially, coming up with such a state requires to find a collision in H or forging a tag for a fake list of commitments.

A detailed description of the compiler Ψ_{RR} and a proof of the following theorem are given in our full version [11].

Theorem 4. *Let π_f be a cf-SFE securely evaluating function $f = (f_1, -)$ and $(\mathsf{G}, \mathsf{T}, \mathsf{V})$ be a* UNF-CMA *MAC scheme, $(\tilde{\mathsf{G}}, \tilde{\mathsf{E}}, \tilde{\mathsf{D}})$ be a* CPA*-secure PKE scheme, and H being picked from a family of CRHFs. Then, $\widehat{\pi}_f \leftarrow \Psi_{\mathsf{RR}}(\pi_f, \pi, \ell)$ is a secure* rate-revealing *π-limited protocol for f.*

7 Rate-Limited OPE

Hazay and Lindell [16] design an efficient two-party protocol for oblivious polynomial evaluation (OPE) with security against malicious adversaries. In an OPE protocol, the first party holds a value t while the second party holds a polynomial p of degree d. Their goal is to let the first party learn $p(t)$ without revealing anything else. The protocol takes advantage of an additively homomorphic encryption scheme (Paillier's encryption) and efficient ZK proofs of a few statements related to the encryption scheme. While the authors (only) prove security against malicious adversaries, we observe that, with a small modification, their construction is indeed a commit-first protocol for OPE as well.

First party's commitment. Consider an additively homomorphic encryption scheme $(\mathsf{G}, \mathsf{E}, \mathsf{D})$. The first few steps performed by the first party (the party holding the value t) are as follows: (i) it runs the key generation for the encryption scheme to generate a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{G}(1^\lambda)$, accompanied by a ZK proof of knowledge of the secret key; (ii) then, it encrypts powers of t, i.e. $\mathsf{E}(\mathsf{pk}, t), \mathsf{E}(\mathsf{pk}, t^2), \ldots, \mathsf{E}(\mathsf{pk}, t^d)$, and sends the resulting ciphertexts along with a ZK proof of the validity of the ciphertexts to the other party.

We observe that sending $\mathsf{E}(\mathsf{pk}, t)$ and a ZK proof of its validity constitutes a commitment by the first party to its input t. This commitment scheme realizes the ideal functionality of the first phase in our definition of commit-first protocols. (Recall that this means the simulator can extract both the input and the randomness used to generate the commitment.) In particular, a careful inspection of the security proof of [16] reveals that the simulator can extract both t and the randomness used to encrypt it during the simulation. Extracting the randomness is possible since in Paillier's encryption scheme, given the secret key sk and a ciphertext c, one can recover both the randomness and the message.

Second party's commitment. The commitment of the second party to its input polynomial is slightly more subtle, and requires a small modification to the

original design. In the first few steps, the second party does the following: (i) it runs the key generation to generate a key pair $(\mathsf{pk}', \mathsf{sk}') \leftarrow \mathsf{G}(1^\lambda)$, accompanied by a ZK proof of knowledge of the secret key; (ii) it computes $((\mathsf{E}(\mathsf{pk}', q_1), \mathsf{E}(\mathsf{pk}', p - q_1)), \ldots, (\mathsf{E}(\mathsf{pk}', q_s), \mathsf{E}(\mathsf{pk}', p - q_s))$ where q_i's are random polynomials of degree d for some security parameter s; (iii) it sends all the ciphertext pairs along with ZK proofs of the fact that the homomorphic addition of every pair encrypts the same polynomial (i.e., p), to the first party. We need to slightly modify this step to realize our ideal commitment functionality: For the first pair of ciphertexts, the second party will also include a ZK proof of validity of $(\mathsf{E}(\mathsf{pk}', q_1), \mathsf{E}(\mathsf{pk}', p - q_1))$.

The pair of ciphertexts $(\mathsf{E}(\mathsf{pk}', q_1), \mathsf{E}(\mathsf{pk}', p - q_1))$ and the accompanied ZK proof of their validity, constitute the commitment by the second party to its input polynomial p. Once again, we note that the simulator in the proof is able to extract q_1, p, and the randomness used in the two encryptions, due to the randomness recovering property of Paillier's encryption. The proof of security provided in [16] can be easily modified to show the commit-first property of the above-mentioned variant of their OPE construction.

Claim. The modified oblivious polynomial evaluation protocol of [16] is a commit-first SFE with security against malicious adversaries.

7.1 ZK Proofs for Rate-Limited OPE

We now explain how to derive rate-limited OPE protocols from the scheme of [16], by giving concrete instantiation of our compilers from Section 5 and 6.

Rate-Revealing OPE. Consider first our rate-revealing compiler from Figure 1. A proof of repeated-input, here, is equivalent to proving a statement for the following language:

$$\mathcal{L}^{\text{ope}}(n) = \left\{ \begin{array}{l} (\mathsf{pk}, \hat{c}, c_1, \ldots, c_n) : \exists \lambda, r \text{ s.t. } (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{G}(1^\lambda, r) \text{ and} \\ (\mathsf{D}(\mathsf{sk}, \hat{c}) = \mathsf{D}(\mathsf{sk}, c_1) \vee \ldots \vee \mathsf{D}(\mathsf{sk}, \hat{c}) = \mathsf{D}(\mathsf{sk}, c_n)) \end{array} \right\},$$

where the ciphertexts $c_1, \ldots, c_n$ are encryptions of the inputs for n previous executions of the OPE protocol. The ciphertext $\hat{c}$ is the encryption of the input for the current execution.

Such a proof can be obtained by exploiting ZK proofs for the languages $\mathcal{L}^{\text{zero}}$ and $\mathcal{L}^{\text{mult}}$ defined in [16]. Informally, a valid proof of a statement in the language $\mathcal{L}^{\text{zero}}$ says that a ciphertext is an encryption of 0. Language $\mathcal{L}^{\text{mult}}$ allows us to prove that given three ciphertexts, one of them decrypts to the product of the other two underlying plaintexts. Denote the plaintext for each c_i by m_i and the one for $\hat{c}$ by $\hat{m}$. The high level idea is to have the prover compute $\mathsf{E}(\mathsf{pk}, (\hat{m} - m_1) \cdots (\hat{m} - m_n))$, prove correctness of this computation and show that the final ciphertext is an encryption of zero. This clearly ensures correctness when the current input equals one of the inputs used in previous executions. See the full version [11] for a complete description.

Rate-Hiding OPE. In the rate-hiding case, besides a standard proof of the statement "a ciphertext is a valid encryption of bit b", the prover also needs

to prove that: (i) "the current commitment corresponds to a fresh input"; (ii) "given a collection of ciphertexts, the sum of the corresponding plaintexts is below some threshold π".

Note that a proof for the first statement is equivalent to proving that an element is *not* in $\mathcal{L}^{\text{ope}}$ (denoted by $\mathcal{L}^{\overline{\text{ope}}}$). Moreover, we show that a proof for the second statement can also be reduced to a proof of membership in $\mathcal{L}^{\overline{\text{ope}}}$ by relying on the homomorphic properties of the underlying encryption scheme. It remains to show ZK proofs for $\mathcal{L}^{\overline{\text{ope}}}$. It is possible to do so using range proofs, but we show a simple and more efficient construction.

Using techniques of [10], the proofs discussed above can be combined (via conjuctive/disjunctive formulas) to generate a ZK proof of membership for the language used in our rate-hiding compiler. A detailed description of the compiler and the proof of the theorem below is given in this paper's full version [11].

Acknowledgements. Daniele Venturi acknowledges support from the Danish National Research Foundation and The National Science Foundation of China (under the grant 61061130540) for the Sino-Danish Center for the Theory of Interactive Computation, and also from the CFEM research center (supported by the Danish Strategic Research Council) within which part of this work was performed. Özgür Dagdelen was supported by CASED (`http://www.cased.de`).

References

1. Barni, M., Failla, P., Kolesnikov, V., Lazzeretti, R., Sadeghi, A.-R., Schneider, T.: Secure Evaluation of Private Linear Branching Programs with Medical Applications. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 424–439. Springer, Heidelberg (2009)
2. Beaver, D., Goldwasser, S.: Multiparty Computation with Faulty Majority. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 589–590. Springer, Heidelberg (1990)
3. Beimel, A., Nissim, K., Omri, E.: Distributed private data analysis: On simultaneously solving how and what. CoRR, abs/1103.2626 (2011)
4. Bellare, M., Goldreich, O., Goldwasser, S.: Incremental Cryptography: The Case of Hashing and Signing. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 216–233. Springer, Heidelberg (1994)
5. Bogdanov, D., Laur, S., Willemson, J.: Sharemind: A Framework for Fast Privacy-Preserving Computations. In: Jajodia, S., Lopez, J. (eds.) ESORICS 2008. LNCS, vol. 5283, pp. 192–206. Springer, Heidelberg (2008)
6. Bogdanov, D., Talviste, R., Willemson, J.: Deploying secure multiparty computation for financial data analysis. Technical report, Cryptology ePrint, 2011/662
7. Bogetoft, P., Christensen, D.L., Damgård, I., Geisler, M., Jakobsen, T., Krøigaard, M., Nielsen, J.D., Nielsen, J.B., Nielsen, K., Pagter, J., Schwartzbach, M., Toft, T.: Secure Multiparty Computation Goes Live. In: Dingledine, R., Golle, P. (eds.) FC 2009. LNCS, vol. 5628, pp. 325–343. Springer, Heidelberg (2009)
8. Chaum, D., Crépeau, C., Damgard, I.: Multiparty unconditionally secure protocols. In: ACM Symposium on Theory of Computing, pp. 11–19. ACM (1988)
9. Chaum, D., van Heijst, E., Pfitzmann, B.: Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 470–484. Springer, Heidelberg (1992)

10. Cramer, R., Damgård, I., Schoenmakers, B.: Proof of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
11. Dagdelen, Ö., Mohassel, P., Venturi, D.: Rate-Limited Secure Function Evaluation: Definitions and Constructions. Cryptology ePrint, Report 201X/XXX
12. Damgård, I., Geisler, M., Krøigaard, M., Nielsen, J.B.: Asynchronous Multiparty Computation: Theory and Implementation. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 160–179. Springer, Heidelberg (2009)
13. Freedman, M.J., Ishai, Y., Pinkas, B., Reingold, O.: Keyword Search and Oblivious Pseudorandom Functions. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 303–324. Springer, Heidelberg (2005)
14. Gennaro, R., Hazay, C., Sorensen, F.: Automata evaluation and text search protocols with simulation based security. Cryptology ePrint, Report 2010/484
15. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: ACM Symposium on Theory of Computing, pp. 218–229. ACM (1987)
16. Hazay, C., Lindell, Y.: Efficient oblivious polynomial evaluation with simulation-based security. Technical report, Cryptology ePrint, Report 2009/459
17. Hazay, C., Lindell, Y.: Efficient Protocols for Set Intersection and Pattern Matching with Security Against Malicious and Covert Adversaries. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 155–175. Springer, Heidelberg (2008)
18. Hazay, C., Nissim, K.: Efficient Set Operations in the Presence of Malicious Adversaries. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 312–331. Springer, Heidelberg (2010)
19. Hazay, C., Toft, T.: Computationally Secure Pattern Matching in the Presence of Malicious Adversaries. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 195–212. Springer, Heidelberg (2010)
20. Henecka, W., Kogl, S., Sadeghi, A.-R., Schneider, T., Wehrenberg, I.: TASTY: tool for automating secure two-party computations. In: ACM CCS 2007 (2010)
21. Huang, Y., Evans, D., Katz, J., Malka, L.: Faster secure two-party computation using garbled circuits. In: USENIX Security (2011)
22. Ishai, Y., Paskin, A.: Evaluating Branching Programs on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 575–594. Springer, Heidelberg (2007)
23. Jarecki, S., Shmatikov, V.: Efficient Two-Party Secure Computation on Committed Inputs. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 97–114. Springer, Heidelberg (2007)
24. Malkhi, D., Nisan, N., Pinkas, B., Sella, Y.: Fairplay—a secure two-party computation system. In: USENIX Security (2004)
25. Mardziel, P., Hicks, M., Katz, J., Srivatsa, M.: Knowledge-oriented secure multiparty computation. In: ACM SIGPLAN – PLAS (June 2012)
26. McGregor, A., Mironov, I., Pitassi, T., Reingold, O., Talwar, K., Vadhan, S.: The limits of two-party differential privacy. ECCC 18:106 (2011)
27. Ogata, W., Kurosawa, K.: Oblivious keyword search. Technical report, Cryptology ePrint, Report 2002/182
28. Troncoso-Pastoriza, J., Katzenbeisser, S., Celik, M.: Privacy preserving error resilient DNA searching through oblivious automata. In: ACM CCS, pp. 519–528 (2007)
29. Yao, A.C.: Protocols for secure computations (extended abstract). In: FOCS, pp. 160–164 (1982)
30. Yao, A.C.: How to generate and exchange secrets (extended abstract). In: FOCS, pp. 162–167 (1986)

Verifiable Elections That Scale for Free

Melissa Chase[1], Markulf Kohlweiss[2], Anna Lysyanskaya[3], and Sarah Meiklejohn[4]

[1] Microsoft Research Redmond
melissac@microsoft.com
[2] Microsoft Research Cambridge
markulf@microsoft.com
[3] Brown University
anna@cs.brown.edu
[4] UC San Diego
smeiklej@cs.ucsd.edu

Abstract. In order to guarantee a fair and transparent voting process, electronic voting schemes must be verifiable. Most of the time, however, it is important that elections also be anonymous. The notion of a *verifiable shuffle* describes how to satisfy both properties at the same time: ballots are submitted to a public bulletin board in encrypted form, verifiably shuffled by several mix servers (thus guaranteeing anonymity), and then verifiably decrypted by an appropriate threshold decryption mechanism. To guarantee transparency, the intermediate shuffles and decryption results, together with proofs of their correctness, are posted on the bulletin board throughout this process.

In this paper, we present a verifiable shuffle and threshold decryption scheme in which, for security parameter k, L voters, M mix servers, and N decryption servers, the proof that the end tally corresponds to the original encrypted ballots is only $O(k(L + M + N))$ bits long. Previous verifiable shuffle constructions had proofs of size $O(kLM + kLN)$, which, for elections with thousands of voters, mix servers, and decryption servers, meant that verifying an election on an ordinary computer in a reasonable amount of time was out of the question.

The linchpin of each construction is a *controlled-malleable proof* (cm-NIZK), which allows each server, in turn, to take a current set of ciphertexts and a proof that the computation done by other servers has proceeded correctly so far. After shuffling or partially decrypting these ciphertexts, the server can also update the proof of correctness, obtaining as a result a *cumulative* proof that the computation is correct so far. In order to verify the end result, it is therefore sufficient to verify just the proof produced by the last server.

1 Introduction

Electronic voting is one of the most compelling applications of cryptography [3]. An approach popular in cryptographic literature is voting via a *verifiable shuffle* [22,11,16,17,19], which consists of L voters $V_1, \ldots, V_L$, M mix servers $S_1, \ldots,$

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 479–496, 2013.

S_M (that are needed for the election to be anonymous) and N threshold decryption servers $D_1, \ldots, D_N$ (that are responsible for setting up the system and, in the end, tallying the results). This approach requires a secure rerandomizable encryption scheme, in which given the public key and a ciphertext c for some message m, one can efficiently find a random ciphertext c' for the same message m. Further, it requires that there be a threshold realization of the cryptosystem [13,24,6]; i.e., the secret key can be split up into "shares" such that each server can use its share to partially decrypt a ciphertext, and the correct decryption can be obtained by putting all the decryption shares together.

On a high level, once the decryption servers set up the system, a verifiable election works in the following three phases [21,4]: first, each voter V_i submits to a public bulletin board a ciphertext $c_i^{(0)}$ containing his or her encrypted ballot (in one variation [22,23,2], a trusted device submits this ciphertext on the user's behalf, so that the user does not know the randomness that went into forming the encryption and thus is unable to demonstrate that he voted a certain way). Next, in the ballot processing phase, each mix server S_i in turn takes as input the set of encrypted ballots $(c_1^{(i-1)}, \ldots, c_L^{(i-1)})$ and randomizes and permutes (i.e., shuffles) them, posting to the public bulletin board the ciphertexts $(c_1^{(i)}, \ldots, c_L^{(i)})$ together with a zero-knowledge proof π_i that this was done correctly. Finally, in the tallying phase, on input $(c_1^{(M)}, \ldots, c_L^{(M)})$, each decryption server D_i publicly outputs its decryption shares $(d_1^{(i)}, \ldots, d_L^{(i)})$, together with a zero-knowledge proof π_i' that this was done correctly. The tally is now publicly computable by putting together the decryption shares for each ciphertext.

How much data does an elections monitor have to process in order to verify the tally? Suppose the monitor observes and verifies *every* step of both mixing and decrypting. This means verifying that, in the ballot processing step, the mix servers correctly formed LM ciphertexts, and then that the decryption servers correctly computed LN decryption shares. This multiplicative blow-up is very unfortunate if these algorithms are used on a large scale; indeed, the very vision of universally verifiable elections is that it should be easy for anyone, including the voters themselves, to participate in guaranteeing both the anonymity and the correctness of the election. This means that it should scale well as the number of mix and decryption servers grows. Can the work of the elections monitors be reduced to $O(k(L + M + N))$ for security parameter k?

(Note that a verifiable shuffle has the attractive property that the set of ballots output in the end is the same as the set of ballots that were encrypted and submitted to the bulletin board. In particular, this allows for write-in candidates. If an election is simply binary, then an encrypted tally can be computed if the underlying cryptosystem is additively homomorphic, and the resulting ciphertext can be decrypted by the decryption servers.)

In a recent result [7], we (referred to ask CKLM in what follows to distinguish between our current and prior work) proposed an idea for overcoming this blow-up as far as the ballot processing phase was concerned. Before, all known aggregation results [1,15] required complex interactions between shuffling authorities and, for non-interactive verification, were based on the Fiat-Shamir [14]

heuristic and thus the random oracle model. The crucial observation of CKLM is that the monitor does not need to verify every step of the shuffle: it is sufficient to just verify the last set of ciphertexts $(c_1^{(M)}, \ldots, c_L^{(M)})$, as long as the proof π_M produced by the last mix server S_M attests to the fact that these were correctly computed from the *original* ballots $(c_1^{(0)}, \ldots, c_L^{(0)})$. Of course, the last mix server S_M does not have the witness to this statement: it knows only the randomness it used to randomize and shuffle the ciphertexts $(c_1^{(M-1)}, \ldots, c_L^{(M-1)})$. To nevertheless allow π_M to suffice for the entire shuffle, CKLM proposed a cryptographic tool, called *controlled-malleable proofs* (cm-NIZKs), that allows each server S_i to build on the proof π_{i-1} that attests to the validity of the ciphertexts $(c_1^{(i-1)}, \ldots, c_L^{(i-1)})$ in order to obtain the proof π_i attesting to the validity of $(c_1^{(i)}, \ldots, c_L^{(i)})$; importantly, cm-NIZKs allow π_i to be the same size as π_{i-1}. As a result, the proof π_M suffices, and the elections monitor need not verify any of the intermediate ciphertexts and proofs. CKLM then gave a construction of cm-NIZKs by taking advantage of certain convenient properties of GS proofs [20].

The CKLM result came with a significant caveat that made it almost irrelevant in practice as far as verifiable shuffles are concerned: they used permutation matrices to represent the statement that there exists a permutation and a randomization that, when applied to $(c_1^{(i-1)}, \ldots, c_L^{(i-1)})$, result in $(c_1^{(i)}, \ldots, c_L^{(i)})$. A permutation matrix is $L \times L$, and so, by necessity, each proof π_i was $\Theta(L^2 k)$ bits, for the security parameter k. The elections monitor would thus have to read $\Theta(k(L^2 + M))$ bits in order to verify the correctness of a shuffle, rather than $\Theta(LMk)$ bits when using, for example, the verifiable shuffle of Groth and Lu [19], which does require the monitor to check intermediate ciphertexts and proofs (hence the factor of M), but in which each proof is only of size $\Theta(Lk)$ because Groth and Lu represent a permutation as a list rather than a matrix. The CKLM solution is therefore asymptotically superior only in the case where there are more mix servers than voters. In recent follow-up work, CKLM extended their results [8] in a way that would allow permutations to be represented as lists rather than matrices, but the extension does not apply for the scenario at hand because it can only tolerate a constant number of mix servers. A natural question, therefore, is the following: Is it possible to combine the CKLM techniques with the Groth-Lu techniques to get a cm-NIZK for the correctness of a shuffle of size $\Theta(k(L + M))$? In this paper, we answer it in the affirmative, obtaining a verifiable shuffle construction in which elections monitors only read $\Theta(k(L + M))$ bits to verify that the ballot processing step was done correctly.

Next, we focus on the application of cm-NIZKs to the verification of threshold decryption (i.e., the tallying phase). In a naïve approach, each decryption server D_i, on input the ciphertexts $(c_1^{(M)}, \ldots, c_L^{(M)})$, outputs the decryption shares $(d_1^{(i)}, \ldots, d_L^{(i)})$ and the proofs $(\pi_1^{(i)}, \ldots, \pi_L^{(i)})$ that these decryption shares were correct. It is natural to ask whether, by taking turns processing these ciphertexts and using cm-NIZK techniques, it is possible to achieve *compact* verifiable threshold decryption, in which each server builds on the decryption share

and proof of the previous server to arrive, at the end, at the vector of decryptions $(m_1, \ldots, m_L)$ for the original L ciphertexts and a single vector of proofs $(\pi_1^{(N)}, \ldots, \pi_L^{(N)})$ that attests to the correct decryption and requires $\Theta(k(L+N))$ bits to verify. In this paper we answer this question in the affirmative as well. Rather than have each decryption server produce its own decryption share and proof of correctness, we instead have the decryption servers pass around a single *cumulative* share, along with a malleable proof of correctness. When one authority receives the share and proof from the previous authority, it can therefore fold in its own share, and update, or "maul", the proof to obtain a new proof of correctness that takes into account this new share.

To the best of our knowledge, the question of compact verifiable threshold decryption has not been previously considered: the standard approach in threshold cryptography [13,24,6] is that, on input the ciphertext and a share of the secret key, each decryption server computes a share of the decryption and a proof that this share was computed correctly. These shares are then publicly output, and the decryption can be computed; one can verify that the decryption is correct by verifying the proofs. In a t-out-of-N threshold cryptosystem, $t+1$ correct shares are sufficient, while no malicious coalition of t servers can break the security of the cryptosystem or cause incorrect decryption. An advantage of this approach is that no communication need be required between servers; in the public bulletin board model of electronic voting, however, this is not as important as compact verification. Our approach, instead, has the decryption servers communicate via the public bulletin board. Each server, in turn, takes as input the cumulative decryptions and their proofs of correctness so far (if any), carries out its share of the decryption, and outputs the resulting cumulative decryption shares and the resulting cumulative cm-NIZK proof of their correctness. The overall process results in the correct decryption if no server fails to produce a valid proof.

2 Definitions and Notation

In this section, we present building blocks and definitions for our voting scheme. First, we recall the malleable proof system due to CKLM [7] used by both our shuffle and threshold decryption constructions. Then, we give the CKLM definition of a verifiable shuffle, which takes into account that one proof is used to prove correctness of the entire shuffle. Next, we give a new definition, analogous to the definition for the shuffle, of compact threshold encryption; here, the malleable proof is used to prove correct partial decryption. Finally, in order to show that these two notions fit together, we present a simple definition of a secure voting scheme.

2.1 Controlled Malleable Proofs (cm-NIZKs)

As defined by CKLM, a controlled malleable proof for a relation R and transformation class $\mathcal{T}$ consists of four algorithms $(\mathsf{CRSSetup}, \mathcal{P}, \mathcal{V}, \mathsf{ZKEval})$: $\mathsf{CRSSetup}$ generates a common reference string crs, the prover $\mathcal{P}$ takes as input the crs,

the instance x, and a witness w for the truth of the statement $(x, w) \in R$ and outputs a proof π, and the verifier $\mathcal{V}$ takes as input the crs, an instance x, and a proof π and either accepts or rejects the proof.

These three algorithms constitute a regular non-interactive proof (which we define formally in the full version of the paper [9]); such a proof is further called *zero knowledge* (NIZK) if there exists a PPT simulator (S_1, S_2) such that an adversary can't distinguish between proofs formed by the prover and proofs formed by the simulator, and a *proof of knowledge* (NIZKPoK) if there exists a PPT extractor (E_1, E_2) that can produce a valid witness from any accepting proof.

The fourth algorithm, specific to malleable proof systems, is ZKEval, which, on input crs, a transformation $T = (T_{\text{inst}}, T_{\text{wit}})$ (in some transformation class $\mathcal{T}$), an instance x, and a proof π, outputs a mauled proof π' for instance $T_{\text{inst}}(x)$. The main definition of CKLM for controlled malleable proofs then reconciles malleability with extractability (specifically, simulation-sound extractability [12,18]) and requires that, for any instance x, if an adversary can produce a valid proof π that $x \in L_R$ then an extractor can extract from π either a witness w such that $(x, w) \in R$ or a previously proved instance x' and transformation $T \in \mathcal{T}$ such that $x = T_{\text{inst}}(x')$. Intuitively this guarantees that any proof that the adversary produces is either generated from scratch using a valid witness, or formed by applying a transformation from the class $\mathcal{T}$ to an existing proof. They define this formally as follows:

Definition 2.1. [7] *Let* $(\mathsf{CRSSetup}, \mathcal{P}, \mathcal{V}, \mathsf{ZKEval})$ *be a NIZKPoK system for an efficient relation* R*, with a simulator* (S_1, S_2) *and an extractor* (E_1, E_2)*. Let* $\mathcal{T}$ *be an allowable set of unary transformations for the relation* R *such that membership in* $\mathcal{T}$ *is efficiently testable. Let* SE_1 *be an algorithm that, on input* 1^k*, outputs* $(\mathsf{crs}, \tau_s, \tau_e)$ *such that* (crs, τ_s) *is distributed identically to the output of* S_1*. Let* $\mathcal{A}$ *be given, and consider the following game:*

- *Step 1.* $(\mathsf{crs}, \tau_s, \tau_e) \xleftarrow{\$} SE_1(1^k)$.
- *Step 2.* $(x, \pi) \xleftarrow{\$} \mathcal{A}^{S_2(\mathsf{crs}, \tau_s, \cdot)}(\mathsf{crs}, \tau_e)$.
- *Step 3.* $(w, x', T) \leftarrow E_2(\mathsf{crs}, \tau_e, x, \pi)$.

The proof system satisfies controlled-malleable simulation-sound extractability *(CM-SSE, for short) with respect to* $\mathcal{T}$ *if for all PPT algorithms* $\mathcal{A}$ *there exists a negligible function* $\nu(\cdot)$ *such that the probability (over the choices of* SE_1*,* $\mathcal{A}$*, and* S_2*) that* $\mathcal{V}(\mathsf{crs}, x, \pi) = 1$ *and* $(x, \pi) \notin Q$ *(where* Q *is the set of queried statements and their responses) but either (1)* $w \neq \perp$ *and* $(x, w) \notin R$*; (2)* $(x', T) \neq (\perp, \perp)$ *and either* $x' \notin Q_x$ *(the set of queried instances),* $x \neq T_{inst}(x')$*, or* $T \notin \mathcal{T}$*; or (3)* $(w, x', T) = (\perp, \perp, \perp)$ *is at most* $\nu(k)$*.*

In addition, CKLM define the notion of *strong derivation privacy* for such proofs, in which simulated proofs are indistinguishable from those formed via transformation. This is defined formally as follows:

Definition 2.2. [7] *For a malleable NIZK* $(\mathsf{CRSSetup}, \mathcal{P}, \mathcal{V}, \mathsf{ZKEval})$ *with an associated simulator* (S_1, S_2)*, a given adversary* $\mathcal{A}$*, and a bit* b*, let* $p_b^{\mathcal{A}}(k)$ *be the probability of the event that* $b' = 0$ *in the following game:*

- *Step 1.* $(\sigma_{sim}, \tau_s) \xleftarrow{\$} S_1(1^k)$.
- *Step 2.* $(\mathsf{state}, x_1, \pi_1, \ldots, x_q, \pi_q, T) \xleftarrow{\$} \mathcal{A}(\sigma_{sim}, \tau_s)$.
- *Step 3. If* $\mathcal{V}(\sigma_{sim}, x_i, \pi_i) = 0$ *for some* i*,* $(x_1, \ldots, x_q)$ *is not in the domain of* T_{inst}*, or* $T \notin \mathcal{T}$*, abort and output* $\perp$*. Otherwise, form*

$$\pi \xleftarrow{\$} \begin{cases} S_2(\sigma_{sim}, \tau_s, T_{inst}(x_1, \ldots, x_q)) & \text{if } b = 0 \\ \mathsf{ZKEval}(\sigma_{sim}, T, \{x_i, \pi_i\}_i) & \text{if } b = 1. \end{cases}$$

- *Step 4.* $b' \xleftarrow{\$} \mathcal{A}(\mathsf{state}, \pi)$.

The proof system is strongly derivation private *if for all PPT algorithms* $\mathcal{A}$ *there exists a negligible function* $\nu(\cdot)$ *such that* $|p_0^{\mathcal{A}}(k) - p_1^{\mathcal{A}}(k)| < \nu(k)$.

Putting these two definitions together, if a proof system is CM-SSE, strongly derivation private, and zero knowledge, then CKLM call it a cm-NIZK.

2.2 Compactly Verifiable Shuffles

In a compact verifiable shuffle, as defined by CKLM, a single (malleable) proof is used to prove the correctness of an entire multi-step shuffle. Formally, a compact verifiable shuffle $(\mathsf{Setup}, \mathsf{ShuffleKg}, \mathsf{Shuffle}, \mathsf{Verify})$ is parameterized by a re-randomizable encryption scheme $(\mathsf{EncKg}, \mathsf{Enc}, \mathsf{Dec})$: Setup generates parameters *params*; $\mathsf{ShuffleKg}$ outputs key pairs (pk_j, sk_j) chosen from a hard relation R_{pk} that are used by mix servers as a stamp of participation; $\mathsf{Shuffle}$ takes the original ciphertexts $\{c_i, \pi_i\}_i$, the shuffled ciphertexts $\{c_i'\}_i$ and proof thus far $(\pi, \{pk_j\}_j)$, and a pair of keys $(pk_m, sk_m) \in R_{pk}$, and outputs $(\{c_i''\}_i, \pi', \{pk_j\}_j \cup \{pk_m\})$; and Verify ensures that the shuffle has been performed correctly.

Before giving the compact verifiability definition, we recall the relation that is proved by the shuffle. Instances are of the form $(pk, \{c_i\}_i, \{c_i'\}_i, \{pk_j\}_j)$, where pk is a public key produced by EncKg, $\{c_i\}_i$ are ciphertexts produced by Enc through the voting process, $\{c_i'\}_i$ are the shuffled ciphertexts, and $\{pk_j\}_j$ are the public keys for R_{pk} that are used to identify the mix servers that have participated in the shuffle thus far. Witnesses are of the form $(\varphi, \{R_i\}_i, \{sk_j\}_j)$, where φ is a permutation, $\{R_i\}_i$ are re-randomization factors, and $\{sk_j\}_j$ are the secret keys for the mix servers. Then the relation R is defined by

$$\begin{aligned} &((pk, \{c_i\}_i, \{c_i'\}_i, \{pk_j\}_j), (\varphi, \{R_i\}_i, \{sk_j\}_j)) \in R \\ \Leftrightarrow &\{c_i'\}_i = \{\mathsf{ReRand}(pk, \varphi(c_i); R_i)\}_i \wedge \forall j (pk_j, sk_j) \in R_{pk}. \end{aligned}$$

Definition 2.3. [7] *Let* $(\mathsf{Setup}, \mathsf{ShuffleKg}, \mathsf{Shuffle}, \mathsf{Verify})$ *be a verifiable shuffle with respect to an encryption scheme* $(\mathsf{EncKg}, \mathsf{Enc}, \mathsf{Dec})$*. For an adversary* $\mathcal{A}$ *and a bit* $b \in \{0, 1\}$*, let* $p_b^{\mathcal{A}}(k)$ *be the probability that* $b' = 0$ *in the following experiment:*

- *Step 1. params* $\xleftarrow{\$}$ $\mathsf{Setup}(1^k)$, $(pk, sk) \xleftarrow{\$} \mathsf{EncKg}(params)$, *and* $(T = \{pk_i\}_i, \{sk_i\}_i) \xleftarrow{\$} \mathsf{ShuffleKg}(1^k)$.
- *Step 2.* $\mathcal{A}$ *gets params,* pk, T, *and access to the following two oracles: an initial shuffle oracle that, on input* $(\{c_i, \pi_i\}_i, pk_\ell)$ *for* $pk_\ell \in T$, *outputs* $(\{c'_i\}_i, \pi, \{pk_\ell\}_\ell)$ *(if all the proofs of knowledge* π_i *verify), where* π *is a proof that the* $\{c'_i\}_i$ *constitute a valid shuffle of the* $\{c_i\}_i$ *performed by the user corresponding to* pk_ℓ *(i.e., the user who knows* sk_ℓ*); and a shuffle oracle that, on input* $(\{c_i, \pi_i\}_i, \{c'_i\}_i, \pi, \{pk_j\}_j, pk_m)$ *for* $pk_m \in T$, *outputs* $(\{c''_i\}_i, \pi', \{pk_j\}_j \cup \{pk_m\})$.
- *Step 3. Eventually,* $\mathcal{A}$ *outputs a tuple* $(\{c_i, \pi_i\}_i, \{c'_i\}_i, \pi, T' = \{pk_j\}_j)$.
- *Step 4. If* $\mathsf{Verify}(params, (\{c_i, \pi_i\}_i, \{c'_i\}_i, \pi, \{pk_j\}_j)) = 1$ *and* $T \cap T' \neq \emptyset$ *then continue; otherwise simply abort and output* $\perp$. *If* $b = 0$ *give* $\mathcal{A}$ $\{\mathsf{Dec}(sk, c'_i)\}_i$, *and if* $b = 1$ *then give* $\mathcal{A}$ $\varphi(\{\mathsf{Dec}(sk, c_i)\}_i)$, *where* φ *is a random permutation.*
- *Step 5.* $\mathcal{A}$ *outputs a guess bit* b'.

Then the shuffle is compactly verifiable *if for all PPT algorithms* $\mathcal{A}$ *there exists a negligible function* $\nu(\cdot)$ *such that* $|p_0^{\mathcal{A}}(k) - p_1^{\mathcal{A}}(k)| < \nu(k)$.

In addition to defining such a shuffle, CKLM also provide a generic construction using a hard relation [10], a proof of knowledge, and a cm-NIZK. Since we use this generic construction as a template for our shuffle construction in Section 3, for completeness we provide an outline of it in the full version of the paper.

2.3 Threshold Encryption

As discussed in the introduction, the previous model for threshold encryption had each participant generate a share and proof of correctness separately; the proofs of correctness would then be verified separately, and the shares would all be combined at the end to produce the decrypted ciphertext. As we now assume that the participants compute a single share and proof cumulatively (by computing their own shares and then folding them into a single one that gets passed around and mauling the accompanying proof appropriately), the model must be changed to reflect these differences.

With this in mind, we define a threshold encryption scheme to be a tuple of four algorithms $(\mathsf{EncKg}, \mathsf{Enc}, \mathsf{ShareDec}, \mathsf{ShareVerify})$. The first, EncKg, generates a public encryption key pk, a verification key vk that is used to check the validity of a share, and a set of secret key shares $\{sk_i\}_i$. The next, Enc, performs regular public-key encryption. The next, $\mathsf{ShareDec}$, takes in a share sk_i of the secret key, a ciphertext c, and the decryption share/proof thus far. It first computes its own partial decryption of c, and then folds this value into the cumulative share and outputs this new share; it also mauls the proof to take into account that the value it has folded in is correct, and thus the new share is the correct cumulative share for the participants thus far. Finally, $\mathsf{ShareVerify}$ takes in the cumulative share and proof and verifies that the share is indeed correct. In this paper we focus on n-out-of-n threshold decryption, in which all n parties must participate

in the decryption; our results should generalize to the t-out-of-n case as well, but we leave that as an open problem.

There are a number of desirable properties of a threshold encryption scheme. Functionally, we require completeness, which says that if everyone is behaving honestly then the scheme works as it should; i.e., the proofs of correctness verify and the ciphertexts decrypt correctly. Completeness therefore requires that the threshold encryption scheme also yields a regular encryption scheme: the Dec algorithm would take as input $sk := \{sk_j\}_j$ and compute the cumulative shares; it would then output the final cumulative share, which by completeness is equal to the message m. This essentially means $\mathsf{Dec}(sk, c) = \mathsf{ShareDec}(pk, vk, sk, c, (\perp, \perp, \perp))$.

In terms of security properties, we would first like our scheme to satisfy IND-CPA security; to capture this, we can use the usual IND-CPA security experiment, in which an adversary $\mathcal{A}$ outputs message (m_0, m_1) such that $|m_0| = |m_1|$ and is asked to guess which one of them a challenge ciphertext c^* encrypts. In addition to IND-CPA security, in the threshold setting we would also like to guarantee that partial decryption shares do not reveal anything about the secret key shares, even in the face of malicious participants (which also means that these malicious participants should not be able to recover the message without a sufficient number of collaborators). To capture this requirement, which we call *share simulatability*, we have the following definition:

Definition 2.4. *Let* $(\mathsf{EncKg}, \mathsf{Enc}, \mathsf{ShareDec}, \mathsf{ShareVerify})$ *be a threshold encryption scheme with* N *decryption participants. For an adversary* $\mathcal{A}$ *and a bit* b*, let* $p_b^{\mathcal{A}}(k)$ *be the probability of the event that* $b' = 0$ *in the following game:*

- *Step 1.* $\{1, \ldots, N\} \supset S \xleftarrow{\$} \mathcal{A}(1^k, N)$.
- *Step 2.* $(pk, vk, \{sk_i\}_i) \xleftarrow{\$} K(1^k, N, S)$.
- *Step 3.* $b' \xleftarrow{\$} \mathcal{A}^{SD}(pk, vk, \{sk_i\}_{i \in S})$,

where (K, SD) *are defined as* $(\mathsf{EncKg}, \mathsf{ShareDec})$ *if* $b = 0$ *and the following algorithms if* $b = 1$*:*

Procedure $K(1^k, n, S)$	*Procedure* $SD(t := (i, c, s, I, \pi))$
$(pk, vk', \{sk'_j\}_{j=1}^N) \xleftarrow{\$} \mathsf{EncKg}(1^k, N)$	$m \leftarrow \mathsf{Dec}(\{sk_j\}_{j=1}^N, c)$
$(vk, \{sk_j\}_{j \in S}, \tau) \xleftarrow{\$} \mathsf{SimKg}(pk, vk', N, S)$	$(s', \pi') \xleftarrow{\$} \mathsf{SimShareDec}(pk, vk, \tau, t, m)$
output $(pk, vk, \{sk_j\}_{j \in S} \cup \{sk'_j\}_{j \in [N] \setminus S})$	*output* $(s', I \cup \{i\}, \pi')$

Then the threshold encryption scheme is share simulatable *if there exist PPT algorithms* SimKg *and* $\mathsf{SimShareDec}$ *as used above such that for all PPT algorithms* $\mathcal{A}$ *there exists a negligible function* $\nu(\cdot)$ *such that* $|p_0^{\mathcal{A}}(k) - p_1^{\mathcal{A}}(k)| < \nu(k)$.

As $\mathsf{SimShareDec}$ can therefore simulate the decryption process without access to the secret key, we can argue that the shares produced by $\mathsf{ShareDec}$ do not reveal anything more than what an honest decryption would reveal. Finally, we require that the proof of correctness is meaningful; i.e., that it is hard for an adversary to produce a ciphertext c, a message m and an accepting proof π such that $m \neq \mathsf{Dec}(sk, c)$. More formally:

Definition 2.5. *Let* $(\mathsf{EncKg}, \mathsf{Enc}, \mathsf{ShareDec}, \mathsf{ShareVerify})$ *be a threshold encryption scheme with* N *decryption participants. For an adversary* $\mathcal{A}$*, define the following game:*

- *Step 1.* $(pk, vk, \{sk_i\}_i) \xleftarrow{\$} \mathsf{EncKg}(1^k, N)$.
- *Step 2.* $(c, m, \pi) \xleftarrow{\$} \mathcal{A}(pk, vk, \{sk_i\}_i)$,

Then the threshold encryption scheme is sound *if for all PPT algorithms* $\mathcal{A}$ *there exists a negligible function* $\nu(\cdot)$ *such that the probability that* $\mathsf{ShareVerify}(pk, vk, c, (m, [N], \pi)) = 1$ *but* $m \neq \mathsf{Dec}(\{sk_i\}_i, c)$ *is at most* $\nu(k)$.

Putting everything together, we say that a threshold encryption scheme is secure if it satisfies IND-CPA security, share simulatability, and soundness.

2.4 Compactly Verifiable Voting

In order for ballots to be cast and elections to be publicly verifiable, verifiable voting schemes use a public space (in practice, an append-only authenticated storage system) commonly referred to as a *bulletin board*. To describe an election, we break it up into several phases, which we describe here. To ease exposition, we implicitly assume that all parties are informed and agree about when a particular phase ends and the next one starts; e.g., by a particular symbol being written on the bulletin board.

- Setup. All authorities meet and jointly compute the public parameters of the election, while also keeping some correlated secrets private. All public parameters are published on the bulletin board.
- Voting. Each voter now uses these public parameters to encrypt his vote v and produce a ballot. All ballots are written on the bulletin board.
- Ballot processing. Next, once all ballots have been cast, they are examined to weed out invalid or duplicate ballots, and a set of mix authorities shuffle the remaining valid ballots.
- Tallying. Finally, a set of decryption authorities work together to decrypt the shuffled ballots. After decryption, the actual count of the votes can be performed publicly.

This multi-phase model of elections is inspired by the work of Juels et al. [21] and Bernhard et al. [4], although with some crucial modifications: unlike the former, we do not address coercion resistance, and unlike the latter we consider both shuffling and threshold decryption.

As far as security is concerned, there are a wide variety of properties we might want a voting scheme to satisfy; e.g., keeping users' votes private, coercion resistance, end-to-end verifiability, etc. In this paper, we focus mainly on this first property. As did Benaloh [3], we observe that we can provide voter privacy only up to a certain point; for example, if the election consisted of only one vote

(or only one vote not controlled by some adversary), then voter privacy would be quite difficult to enforce! We therefore follow Benaloh's approach in requiring that votes can be private only in elections in which different assignments of honest votes still lead to the same outcome. To capture this property formally, we say that an election with N decryption authorities, L voters, and M mix authorities satisfies basic vote privacy if, for a random bit $b \xleftarrow{\$} \{0,1\}$, no PPT adversary $\mathcal{A}$ can win the following game with more than negligible advantage:

- Setup. First, a random bit $b \xleftarrow{\$} \{0,1\}$ is chosen. Then, $\mathcal{A}$ picks the decryption authorities to corrupt as $[N] \supset S \xleftarrow{\$} \mathcal{A}(1^k)$. Then, $params \xleftarrow{\$} \mathsf{Setup}(1^k)$, $(\{pk_i\}_i, \{sk_i\}_i) \xleftarrow{\$} \mathsf{ShuffleKg}(params)$, $(pk, vk, \{dk_j\}_j) \xleftarrow{\$} \mathsf{EncKg}(params)$. At the end of the setup phase $(params, pk, vk)$ are added to the bulletin board, and $\mathcal{A}$ gets to see $\{dk_j\}_{j \in S}$ and $T := \{pk_i\}_i$.
- Voting. Proceeding adaptively, the adversary can either provide his own ballot B, or a vote pair (v_0, v_1). For the former, the ballot is simply added to the bulletin board, while for the latter he gets back the ballot B_b (i.e., the ballot corresponding to either v_0 or v_1), which is also added to the bulletin board. At the end of the phase (i.e., once there are L votes on the board), $\mathcal{A}$ automatically loses if the election outcome differs between $b = 0$ and $b = 1$.
- Ballot processing. In this phase, in addition to access to the bulletin board, we give the adversary access to two shuffle oracles: an initial shuffle oracle that, on input pk_ℓ for $pk_\ell \in T$, writes $(\{c'_i\}_i, \pi, \{pk_\ell\}_\ell)$ on the bulletin board (if all the ballots on the bulletin board are valid), where π is a proof that the $\{c'_i\}_i$ constitute a valid shuffle of the initial ballots $\{B_i\}_i$ performed by the user corresponding to pk_ℓ (i.e., the user who knows sk_ℓ); and a regular shuffle oracle that, on input $(\{c'_i\}_i, \pi, \{pk_j\}_j, pk_\ell)$ for $pk_\ell \in T$, adds both $(\{c'_i\}_i, \pi, \{pk_j\}_j)$ (if it cannot be found there already) and the shuffled $(\{c''_i\}_i, \pi', \{pk_j\}_j \cup \{pk_\ell\})$ to the bulletin board. The phase ends when the final shuffle $(\{c'_i\}_i, \pi, \{pk_j\}_j)$ such that $|\{pk_j\}_j| = M$ and $\{pk_j\}_j \cap T \neq \emptyset$ is written to the bulletin board, either by the shuffle oracle or by the adversary.
- Tallying. The adversary can ask for decryption shares for the shuffled $\{c'_i\}_i$ through an oracle that, on input (j, k, s_k, I, ϕ_k), computes $(s'_k, I \cup \{j\}, \phi'_k) \xleftarrow{\$} \mathsf{ShareDec}(pk, vk, dk_j, c'_k, (s_k, I, \phi_k))$ and posts both (s_k, I, ϕ_k) (if it cannot be found there already; $s_k = \perp$ and $I = \emptyset$ denotes an initial decryption) and the share $(s'_k, I \cup \{j\}, \phi'_k)$ with its new contribution. The phase ends when, for every i, $1 \leq i \leq L$, the final decryption share $(s_i, [N], \phi_i)$ is written to the bulletin board, either by the share decryption oracle or by the adversary.
- Winning the game. The adversary outputs b', and wins if $b' = b$.

While the above definition explicitly captures vote privacy, we could also attempt to extend it to deal with verifiability by requiring that, if π and ϕ_i verify for all i, then the expected outcome (based on the v_i and the decryption of c_i in the adversary's ballots) should match the real outcome. While the soundness of the

proofs used in our construction in Section 5 should guarantee that this holds, we focus solely on privacy in this work and leave a formal proof of verifiability as an interesting open problem.

3 A Compactly Verifiable Shuffle

In this section, we show how to achieve a compactly verifiable shuffle, as defined in Definition 2.3, with parameter size $O(L)$ and proof size $O(L + M)$ by using the verifiable shuffle due to Groth and Lu [19]. To do this, we use the following outline: first, we show that an adapted version of the Groth-Lu construction is what CKLM call *CM-friendly*, meaning that a pairing-based cm-NIZK can be constructed based on it. We then observe that, once we have a cm-NIZK, we can plug it into the generic construction of CKLM to obtain a compactly verifiable shuffle.

In the definition of CM-friendliness as proposed by CKLM [7, Definition 4.3], they assigned the property of CM-friendliness to a relation and transformation; in the case of a shuffle, this relation and the set of transformations describe the permutation and randomization of ciphertexts, as we saw formally in Section 2.2. We propose a useful weakening of this definition that shifts the assignation of CM-friendliness from the relation to its specific instantiation using a sound proof system; as we will see, this allows the definition to accomodate computationally sound proofs (i.e., arguments) as well as the perfectly sound proofs that the previous definition required. We capture the previous definition as *perfect* CM-friendliness.

Due to space constraints, we present here only an informal version of our definition; the formal definition can be found in the full version of the paper.

Definition 3.1. *(Informal.) For sets S and S' of pairing product equations and a PPT setup algorithm params $\xleftarrow{\$}$ $\mathsf{CRSSetup}(1^k)$ that specifies some group G, we say that $(S, S', \mathsf{CRSSetup})$ is a* CM-friendly instantiation *for a relation R and transformation class $\mathcal{T}$ if the following six properties hold: (1) representable statements: any instance and witness of R can be represented as a set of group elements; (2) representable transformations: any transformation in $\mathcal{T}$ can be represented as a set of group elements; (3) provable statements: proving satisfaction of S constitutes a computationally sound proof for the statement "$(x, w) \in R$" using the above representations for x and w; (4) provable transformations: proving satisfaction of S' constitutes a computationally sound proof for the statement "$T_{inst}(x') = x$ for $T \in \mathcal{T}$" using the above representations for x and T; (5) transformable statements: for any $T \in \mathcal{T}$ the statement "$(x, w) \in R$" (phrased using S as above) can be transformed into the statement "$(T_{inst}(x), T_{wit}(w)) \in R$"; and (6) transformable transformations: for any $T, T' \in \mathcal{T}$, the statement "$T_{inst}(x') = x$ for $T \in \mathcal{T}$" (phrased using S' as above) can be transformed using valid transformations into the statement "$\hat{T}_x(x') = \hat{x}$ for $\hat{T} \in \mathcal{T}$" where $\hat{T} = T' \circ T$ and $\hat{x} = \hat{T}_x(x)$. We say that $(S, S', \mathsf{CRSSetup})$ is a* perfect *CM-friendly instantiation if the probabilities in the third and fourth*

properties are zero. A relation and transformation class $(R, \mathcal{T})$ are (perfectly) CM-friendly, if they have a (perfect) CM-friendly instantiation.

To instantiate the shuffle relation and transformations from Section 2.2, we combine the proof of hard relation instances of CKLM and an adapted version of the Groth-Lu protocol for the permutation proof. We omit the proof that the $\{pk_j\}_j$ are the public keys for R_{pk} in our exposition as it is unchanged from the original CKLM shuffle.

Our adapted version Groth-Lu protocol is slightly less efficient than theirs and achieves a weaker notion of zero knowledge (theirs is perfect whereas ours is computational) but a stronger notion of soundness (theirs achieves the slightly non-standard notion of L_{co}-soundness, whereas ours is computationally sound). These tradeoffs seem necessary, as it is not clear how to accomodate the definition of CM-friendliness (or of a cm-NIZK or compact shuffle) to allow for L_{co}-soundness.

Following Groth and Lu, the instantiation we use for the shuffle encryption scheme is Boneh-Boyen-Shacham (BBS) encryption [5], which uses a prime-order bilinear group setting (p, G, G_T, g, e) with public keys of the form $pk := (f, h)$ for $f := g^\alpha$ and $h := g^\beta$ (for random $\alpha, \beta \xleftarrow{\$} \mathbb{F}_p$) and ciphertexts of the form $c := (u, v, w)$ for $u := f^r$, $v := h^s$, and $w := g^{r+s}m$ (for the message m and $r, s \xleftarrow{\$} \mathbb{F}_p$). Using this, we can show how to satisfy CM-friendliness, starting with $\mathsf{CRSSetup}(1^k)$:

- $\mathsf{CRSSetup}(1^k)$: First generate a prime-order bilinear group (p, G, G_T, e, g). To allow for a shuffle over L ciphertexts, pick $x_1, \ldots, x_L \xleftarrow{\$} \mathbb{F}_p$ and set $g_i := g^{x_i}$ and $\gamma_i := g^{x_i^2}$ for all i. Output $\mathsf{crs} := (p, G, G_t, e, g, \{g_i\}_i, \{\gamma_i\}_i)$.

With this in place, we now describe how the six properties of CM-friendliness are met; in what follows, we highlight the involvement of the permutation by using $\varphi(g_i)$ in place of $g_{\varphi(i)}$ (and similarly for other variables):

1. Representable statements. Because we are using BBS encryption, instances will use $pk = (f, h)$, $c_i = (u_i, v_i, w_i)$, and $c'_i = (u'_i, v'_i, w'_i)$. We represent the witness as follows: to represent φ, we use $(\{a_i\}_i, \{b_i\}_i)$, where $a_i = \varphi(g_i)$ and $b_i = \varphi(\gamma_i)$ for all i, $1 \le i \le L$, and to represent R_i we use $(f^{r'_i}, h^{s'_i}, g^{r'_i}, g^{s'_i})$ for random $r'_i, s'_i \xleftarrow{\$} \mathbb{F}_p$.
2. Representable transformations. We represent $T_{(\varphi, \{R_i\}_i)} = (T_{\text{inst}}, T_{\text{wit}})$ in the same form as witnesses; i.e., $(\{a_i\}_i, \{b_i\}_i)$ for φ and $(f^{r'_i}, h^{s'_i}, g^{r'_i}, g^{s'_i})$ for all R_i.
3. Provable statements. To prove that, under the public key $pk = (f, h)$, the set of ciphertexts $\{(u'_i, v'_i, w'_i)\}_i$ is a shuffle of $\{(u_i, v_i, w_i)\}_i$ using the permutation represented by $(\{a_i\}_i, \{b_i\}_i)$ and re-randomization represented by $\{(f^{r'_i}, h^{s'_i}, g^{r'_i}, g^{s'_i})\}_i$, we use the set S of pairing product equations defined as follows:

$$\textbf{(1)}\ \prod_{i=1}^{L} e(a_i, u_i') = \prod_{i=1}^{L} e(a_i, f^{r_i'}) e(g_i, u_i),\ \textbf{(2)}\ \prod_{i=1}^{L} e(b_i, u_i') = \prod_{i=1}^{L} e(b_i, f^{r_i'}) e(\gamma_i, u_i),$$

$$\textbf{(3)}\ \prod_{i=1}^{L} e(a_i, v_i') = \prod_{i=1}^{L} e(a_i, h^{s_i'}) e(g_i, v_i),\ \textbf{(4)}\ \prod_{i=1}^{L} e(b_i, v_i') = \prod_{i=1}^{L} e(b_i, h^{s_i'}) e(\gamma_i, v_i),$$

$$\textbf{(5)}\ \prod_{i=1}^{L} e(a_i, w_i') = \prod_{i=1}^{L} e(a_i, g^{r_i'} g^{s_i'}) e(g_i, w_i),$$

$$\textbf{(6)}\ \prod_{i=1}^{L} e(b_i, w_i') = \prod_{i=1}^{L} e(b_i, g^{r_i'} g^{s_i'}) e(\gamma_i, w_i),$$

$$\textbf{(7)}\ \prod_{i=1}^{L} a_i g_i^{-1} = 1,\ \textbf{(8)}\ \prod_{i=1}^{L} b_i \gamma_i^{-1} = 1, \textbf{(9)}\ e(a_i, a_i) = e(g, b_i) \text{ for all } i,\ 1 \le i \le L,$$

$$\textbf{(10)}\ e(f^{r_i'}, g) = e(f, g^{r_i'}) \text{ for all } i, \text{and } \textbf{(11)}\ e(h^{s_i'}, g) = e(h, g^{s_i'}) \text{ for all } i.$$

4. Provable transformations. To prove $T_{\text{inst}}(x') = x$ for $T \in \mathcal{T}$, we use the same equations from the above set S. We must additionally prove that the transformation does not change pk or $\{c_i\}_i$; to do this, we form an augmented set S', which consists of all the equations in S as well as equations to check that these values stay fixed. More formally, if we represent X as $(pk, \{(u_i, v_i, w_i)\}_i, \{(u_i', v_i', w_i')\}_i)$ and X' as $(pk', \{(U_i, V_i, W_i)\}_i, \{U_i', V_i', W_i'\}_i)$, then our extra checks ensure that $pk = pk'$ and $u_i = U_i$, $v_i = V_i$, $w_i = W_i$ for all i, $1 \le i \le L$. We can then run the checks in S using T_{inst} as the witness and $X_T := (pk, \{(u_i', v_i', w_i')\}_i, \{(U_i', V_i', W_i')\}_i)$ as the instance.
5. Transformable statements. CKLM already show how to permute variables by a permutation φ and multiply re-randomization factors into ciphertexts using valid transformations; we therefore assume these operations exist and are valid. To change the statement $(x, w) \in R$ into $(T_{\text{inst}}(x), T_{\text{wit}}(w)) \in R$ for $X = (pk, \{(u_i, v_i, w_i)\}_i, \{(u_i', v_i', w_i')\}_i)$, $W = ((\{a_i\}_i, \{b_i\}_i), \{R_i\}_i)$, and $T = (\varphi', \{R_i'\}_i)$, we therefore begin by permuting the values $\{(u_i', v_i', w_i')\}_i$, $\{a_i\}_i$, and $\{b_i\}_i$ by φ'; this operation affects Equations 1 through 9 in S. We then multiply the additional randomness $\{R_i'\}_i$ into Equations 1 through 6, as well as Equations 10 and 11.
6. Transformable transformations. To change the statement $T_{\text{inst}}(x') = x$ into $T'_{\text{inst}} \circ T_{\text{inst}}(x') = T'_{\text{inst}}(x)$, we leave the additional checks in S' (i.e., the checks that ensure that pk and $\{c_i\}_i$ go unchanged) as they are. We then transform S as we did above using the values $(\varphi', \{R_i'\}_i)$ specified in T'_{inst}, so that we permute the values $\{(u_i', v_i', w_i')\}_i$, $\{a_i\}_i$, and $\{b_i\}_i$ by φ' and multiply the additional randomness into Equations 1 through 6 and 10 and 11.

Due to space constraints, a proof of the following theorem can be found in the full version of the paper:

Theorem 3.1. *If both the Permutation Pairing and Simultaneous Pairing assumptions hold, then* $(S, S', \mathsf{CRSSetup})$ *as defined above is a CM-friendly instantiation for the shuffle relation* R *(defined in Section 2.2) and the transformation class* $\mathcal{T}$ *consisting of all valid shuffles.*

Now that we have a CM-friendly instantiation for the shuffle relation, we can use the results of CKLM to construct a cm-NIZK for this relation. As we slightly weakened the notion of CM-friendliness, we argue in the full version of the paper that their results still carry through to produce a cm-NIZK; we mention here that our proof is nearly identical, as the notion of soundness used for cm-NIZKs is already computational.

Armed with our cm-NIZK, we now plug it into the generic verifiable shuffle construction of CKLM , which they already proved secure. We can even use the same representation of mix server keys as CKLM, which means $pk_j := g^{\alpha_j}$ and $sk_j := h^{\alpha_j}$ for $\alpha_j \xleftarrow{\$} \mathbb{F}_p$ and $h := g^{\beta}$ for some $\beta \xleftarrow{\$} \mathbb{F}_p$. As for the size, looking at the construction above we see that the CRS must contain the g_i and γ_i elements for all i (and adding in the parameters for R_{pk} adds only the single group element h), which means the parameters are of size $O(L)$. For the proofs, Equations 9, 10, and 11 in S are required for every i, so the size of the proof is also $O(L)$. In addition, a constant number of equations is required to check that $(pk_j, sk_j) \in R_{pk}$ for every value of j; if the number of mix authorities is M, then this adds a proof component of size $O(M)$ and thus our total proof size is $O(L + M)$.

4 Threshold Decryption

In this section, we provide our construction of a threshold encryption scheme that satisfies the notions of security defined in Section 2.3; i.e., IND-CPA security, share simulatability, and soundness. We provide first a construction using a generic malleable NIZK proof of knowledge (NIZKPoK), and then describe in the full version of the paper [9]how to instantiate this proof system concretely.

4.1 Our Construction

In threshold decryption, the statement that each participant i wants to prove is that the share s he produces is a correct partial decryption of some ciphertext c. Formally, we represent instances as $x = (vk_c, c, s)$, where c is a ciphertext, and s is the cumulative decryption share produced by the combined user represented in vk_c, and witnesses as $(t, open)$, where t is a secret token (in our case, a bijection applied to the cumulative secret key) used to prove correctness of partial decryption, and $vk_c = \mathsf{Com}(t; open)$ for some commitment scheme Com. The statement we want to prove is then

$$((vk_c, c, s),(t, open)) \in R \Leftrightarrow \exists sk_c : vk_c = \mathsf{Com}(t; open) \wedge t = F(sk_c) \wedge s = \mathsf{Dec}(sk_c, c), \quad (1)$$

where F is the bijection between cumulative secret keys and tokens.

Transformations for this relation correspond to a new set of users J folding in their shares. This means we represent transformations as $T = (\hat{s}, \hat{t}, \widehat{open})$, where $T_{\text{inst}}(vk_c, c, s) := (vk_c \cdot \mathsf{Com}(\hat{t}; \widehat{open}), c, s \cdot \hat{s})$ and $T_{\text{wit}}(t, open) = (t \cdot \hat{t}, open \cdot \widehat{open})$; the transformation is considered allowable if $\hat{s}$ is a valid share using the token $\hat{t}$; i.e., $\hat{s}$ was computed using the secret key $\widehat{sk}$ corresponding to $\hat{t}$.

Our concrete instantiation uses BBS encryption [5], which is multiplicatively homomorphic; this is why we multiply both the shares and the tokens to combine them. We also use a commitment scheme Com and a strongly derivation-private malleable NIZK proof of knowledge $(\mathsf{CRSSetup}, \mathcal{P}, \mathcal{V}, \mathsf{ZKEval})$. We will see later how to instantiate the NIZK concretely; for the commitment scheme (which we use to commit to the two components of t) we can use the instantiation of Groth-Sahai commitments under Decision Linear, which are almost identical to BBS encryption (and thus also multiplicatively homomorphic). We thus usually keep these parameters implicit.

- $\mathsf{EncKg}(1^k)$: Generate $\mathsf{crs} \xleftarrow{\$} \mathsf{CRSSetup}(1^k)$ and $\mathsf{par} \xleftarrow{\$} \mathsf{ComSetup}(1^k)$; these are defined over a shared bilinear group (p, G, G_T, e, g). Pick random $\alpha, \beta \xleftarrow{\$} \mathbb{F}_p$, set $f := g^\alpha$ and $h := g^\beta$, and set $pk := (f, h)$. Next, to allow N parties to partake in decryption, compute $a_1, b_1, \ldots, a_{N-1}, b_{N-1} \xleftarrow{\$} \mathbb{F}_p$ and $a_N := -1/\alpha - \sum_i a_i$ and $b_N := -1/\beta - \sum_i b_i$. Next, for all i, set $t_{1i} := g^{a_i}$, $t_{2i} := g^{b_i}$, and form commitments $A_i \xleftarrow{\$} \mathsf{Com}(t_{1i}; open_{1i})$ and $B_i \xleftarrow{\$} \mathsf{Com}(t_{2i}; open_{2i})$ using random openings. Set $vk' := \{(A_i, B_i)\}_i$ and $sk_i := (a_i, b_i, t_{1i}, t_{2i}, open_{1i}, open_{2i})$ for all i, $1 \le i \le n$. Output pk, $vk := (\mathsf{crs}, \mathsf{par}, vk')$, and $\{sk_i\}_i$.
- $\mathsf{Enc}(pk, m)$: Parse $pk = (f, h)$ and pick random $r, s \xleftarrow{\$} \mathbb{F}_p$. Set $u := f^r$, $v := h^s$, $w := g^{r+s} m$, and output $c := (u, v, w)$.
- $\mathsf{Dec}(\{sk_i\}_{i=1}^n, c)$: Parse $c = (u, v, w)$ and $sk_i = (a_i, b_i, t_{1i}, t_{2i}, open_{1i}, open_{2i})$ for all i, and compute $a := \sum_i a_i$ and $b := \sum_i b_i$. Output $m := u^a \cdot v^b w$. (By definition, $a = -1/\alpha$ and $b = -1/\beta$, so this is just standard BBS decryption with a reconstructed key.)
- $\mathsf{ShareDec}(pk, vk, sk_j, c, (s, I, \pi))$: Parse $sk_j = (a_j, b_j, t_{1j}, t_{2j}, open_{1j}, open_{2j})$. If $(s, I, \pi) = (\bot, \bot, \bot)$, then this is the initial decryption. Compute the share $s_j := u^{a_j} v^{b_j} w$ and $\pi \xleftarrow{\$} \mathcal{P}(\mathsf{crs}, (vk_j, c, s_j), (t_{1j}, t_{2j}, open_{1j}, open_{2j}))$, and output $(s_j, \{j\}, \pi)$.
  Otherwise, define $vk_c := \prod_{i \in I} vk'_i$ and check that $\mathcal{V}(\mathsf{crs}, (pk, vk_c, c, s), \pi) = 1$; abort and output $\bot$ if not. Otherwise continue and compute $s_j := u^{a_j} v^{b_j}$ and $s' := s \cdot s_j$; then set $T := (s_j, (t_{1j}, t_{2j}), (open_{1i}, open_{2i}))$. Compute $\pi' \xleftarrow{\$} \mathsf{ZKEval}(\mathsf{crs}, T, (vk_c, c, s), \pi)$, and output $(s', I' := I \cup \{j\}, \pi')$.
- $\mathsf{ShareVerify}(pk, vk, c, (s, I, \pi))$: Parse $vk = (\mathsf{crs}, \mathsf{par}, vk')$ and output $\mathcal{V}(\mathsf{crs}, (pk, \prod_{i \in I} vk'_i, c, s), \pi)$.

As the security of both BBS encryption and our cm-NIZK come from Decision Linear, we obtain the following theorem.

Theorem 4.1. *If Decision Linear holds in G then we can instantiate the above construction to obtain a secure threshold decryption scheme, as defined in Section 2.3.*

To prove this, we must prove that four properties are satisfied: completeness, IND-CPA security, soundness, and share simulatability. The first of these, completeness, follows directly by inspection; similarly, for IND-CPA security, as we use BBS encryption, IND-CPA follows directly from their result and holds under Decision Linear.

For the latter two, we prove them using the security of the commitment scheme and NIZK. Interestingly, while the proof system is required to be malleable, strongly derivation private, and zero knowledge, for soundness we require not the strong notion of CM-SSE for cm-NIZKs, but instead regular extractability (i.e., we require the proof to be a proof of knowledge). Intuitively, the reason for this is that in the soundness game the adversary is not provided with simulated proofs, and we can therefore always expect to be able to extract a witness (rather than just a transformation as we do with CM-SSE).

Lemma 4.1. *If* $(\mathsf{CRSSetup}, \mathcal{P}, \mathcal{V}, \mathsf{ZKEval})$ *is extractable and* Com *is binding, the threshold encryption scheme describe above is sound, as defined in Definition 2.5.*

Lemma 4.2. *If* $(\mathsf{CRSSetup}, \mathcal{P}, \mathcal{V}, \mathsf{ZKEval})$ *is zero knowledge and strongly derivation private, and* Com *is hiding, the threshold encryption scheme described above is share simulatable, as defined in Definition 2.4.*

Due to space constraints, the proofs of these lemmas and the concrete implementation of the cm-NIZK, using Groth-Sahai proofs, can be found in the full version of the paper. For our concrete instantiation, we mention here that we follow the same outline as in Section 3 to show that R has a CM-friendly instantiation. In fact, as we encode x, w, and T directly without relying on any computational assumptions, we can achieve *perfect* CM-friendliness.

5 A Secure Voting Scheme

In this section, we bring together the components constructed in the previous two sections to construct an electronic voting scheme from a compactly verifiable shuffle $(\mathsf{Setup}, \mathsf{ShuffleKg}, \mathsf{Shuffle}, \mathsf{Verify})$, a secure threshold decryption scheme $(\mathsf{EncKg}, \mathsf{Enc}, \mathsf{ShareDec}, \mathsf{ShareVerify})$, and a simulation-sound extractable proof $(\mathsf{CRSSetup}, \mathcal{P}, \mathcal{V})$.

- Setup. The voting authorities jointly compute the parameters $params \xleftarrow{\$} \mathsf{Setup}(1^k)$ and threshold keys $(pk, vk, \{dk_j\}_j) \xleftarrow{\$} \mathsf{EncKg}(params)$. The mix authorities compute the shuffling keys $(\{pk_i\}_i, \{sk_i\}_i) \xleftarrow{\$} \mathsf{ShuffleKg}(params)$, and the values $params$, pk, and vk are added to the bulletin board.
- Voting. Each voter i forms $c_i \xleftarrow{\$} \mathsf{Enc}(pk, v_i)$ (using some randomness r_i) and proves knowledge of his vote by computing $\pi_i \xleftarrow{\$} \mathcal{P}(\mathsf{crs}, (pk, c), (v_i, r_i))$. The resulting ballot (c_i, π_i) is added to the bulletin board.

– Ballot processing. The mix authority with public key pk_k picks the most recent valid shuffle $(\{c'_i\}_i, \pi, \{pk_j\}_j)$; e.g., the one with the most public keys, or the one that has used the correct sequence of public keys (if an order has been imposed). It performs $(\{c''_i\}_i, \pi') \xleftarrow{\$} \mathsf{Shuffle}(params, \{c_i, \pi_i\}_i, \{c'_i\}_i, \pi, \{pk_j\}_j, (pk_k, sk_k))$ and posts $(\{c''_i\}_i, \pi', \{pk_j\}_j \cup \{pk_k\})$ to the bulletin board. The ballot processing phase ends once there is a valid sequence of shuffle proofs with sufficiently many mix authorities.
– Tallying. Let $(\{c'_i\}_i, \pi, \{pk_j\}_j)$ be the completed shuffle. Each decryption authority looks for the valid decryption shares (s_i, I, ϕ_i) with the largest set I. The k-th decryption authority performs $(s'_i, I \cup \{k\}, \phi'_i) \xleftarrow{\$} \mathsf{ShareDec}(pk, vk, dk_k, c_i, (s_i, I, \phi_i))$ for all i and posts $(s'_i, I \cup \{k\}, \phi'_i)$ on the bulletin board.

Theorem 5.1. *The voting scheme outlined above satisfies basic voter privacy, as defined in Section 2.4.*

To prove this, we proceed through a series of game transformations; due to space constraints, the transformations and proofs of their indistinguishability can be found in the full version of the paper.

Acknowledgments. We thank Stephan Neumann for spurring our interest in the application of malleable proofs to threshold decryption. Anna Lysyanskaya was supported by NSF grants 1012060, 0964379, 0831293, and by a Sloan Foundation fellowship, and Sarah Meiklejohn was supported by a MURI grant administered by the Air Force Office of Scientific Research.

References

1. Abe, M.: Universally Verifiable Mix-Net with Verification Work Independent of the Number of Mix-Servers. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 437–447. Springer, Heidelberg (1998)
2. Adida, B., Neff, C.A.: Efficient Receipt-Free Ballot Casting Resistant to Covert Channels. Cryptology ePrint Archive, Report 2008/207 (2008), `http://eprint.iacr.org/2008/207`
3. Benaloh, J.D.C.: Verifiable Secret-Ballot Elections. PhD thesis, Yale University (1987)
4. Bernhard, D., Cortier, V., Pereira, O., Smyth, B., Warinschi, B.: Adapting Helios for Provable Ballot Privacy. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 335–354. Springer, Heidelberg (2011)
5. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
6. Canetti, R., Goldwasser, S.: An Efficient *Threshold* Public Key Cryptosystem Secure against Adaptive Chosen Ciphertext Attack. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 90–106. Springer, Heidelberg (1999)
7. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable Proof Systems and Applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 281–300. Springer, Heidelberg (2012)

8. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Succinct Malleable NIZKs and an Application to Compact Shuffles. Cryptology ePrint Archive, Report 2012/506 (2012), http://eprint.iacr.org/2012/506
9. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Verifiable Elections that Scale for Free. Cryptology ePrint Archive (2012), http://eprint.iacr.org/
10. Damgård, I.: On Sigma Protocols, http://www.daimi.au.dk/~ivan/Sigma.pdf
11. Damgård, I., Groth, J., Salomonsen, G.: The Theory and Implementation of an Electronic Voting System. In: Proceedings of Secure Electronic Voting (SEC), pp. 77–100 (2003)
12. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust Non-interactive Zero Knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Heidelberg (2001)
13. Desmedt, Y., Frankel, Y.: Threshold Cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
14. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
15. Furukawa, J., Imai, H.: An Efficient Aggregate Shuffle Argument Scheme. In: Dietrich, S., Dhamija, R. (eds.) FC 2007 and USEC 2007. LNCS, vol. 4886, pp. 260–274. Springer, Heidelberg (2007)
16. Groth, J.: A Verifiable Secret Shuffle of Homomorphic Encryptions. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 145–160. Springer, Heidelberg (2003)
17. Groth, J.: Non-interactive Zero-Knowledge Arguments for Voting. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 467–482. Springer, Heidelberg (2005)
18. 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)
19. Groth, J., Lu, S.: A Non-interactive Shuffle with Pairing Based Verifiability. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 51–67. Springer, Heidelberg (2007)
20. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
21. Juels, A., Catalano, D., Jakobsson, M.: Coercion-Resistant Electronic Elections. In: Chaum, D., Jakobsson, M., Rivest, R.L., Ryan, P.Y.A., Benaloh, J., Kutylowski, M., Adida, B. (eds.) Towards Trustworthy Elections. LNCS, vol. 6000, pp. 37–63. Springer, Heidelberg (2010)
22. Neff, C.A.: A Verifiable Secret Shuffle and its Application to E-Voting. In: Proceedings of ACM CCS 2001, pp. 116–125. ACM Press (November 2001)
23. Sandler, D., Derr, K., Wallach, D.S.: Votebox: A Tamper-evident, Verifiable Electronic Voting System. In: USENIX Security Symposium, pp. 349–364 (2008)
24. Shoup, V., Gennaro, R.: Securing Threshold Cryptosystems against Chosen Ciphertext Attack. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 1–16. Springer, Heidelberg (1998)

On the Connection between Leakage Tolerance and Adaptive Security

Jesper Buus Nielsen, Daniele Venturi, and Angela Zottarel

Aarhus University

Abstract. We revisit the context of leakage-tolerant interactive protocols as defined by Bitanski, Canetti and Halevi (TCC 2012). Our contributions can be summarized as follows:

1. For the purpose of secure message transmission, any encryption protocol with message space $\mathcal{M}$ and secret key space $\mathcal{SK}$ tolerating poly-logarithmic leakage on the secret state of the receiver must satisfy $|\mathcal{SK}| \geq (1-\epsilon)|\mathcal{M}|$, for every $0 < \epsilon \leq 1$, and if $|\mathcal{SK}| = |\mathcal{M}|$, then the scheme must use a fresh key pair to encrypt each message.
2. More generally, we show that any n party protocol tolerates leakage of $\approx poly(\log \kappa)$ bits from one party at the end of the protocol execution, *if and only if* the protocol has passive adaptive security against an adaptive corruption of one party at the end of the protocol execution. This shows that as soon as a little leakage is tolerated, one needs full adaptive security.
3. In case more than one party can be corrupted, we get that leakage tolerance is equivalent to a weaker form of adaptivity, which we call *semi-adaptivity*. Roughly, a protocol has semi-adaptive security if there exist a simulator which can simulate the internal state of corrupted parties, however, such a state is not required to be indistinguishable from a real state, only that it would have lead to the simulated communication.

All our results can be based on the solely assumption that collision-resistant function ensembles exist.

Keywords: simulation-based security, leakage tolerance, adaptive security, arguments of knowledge.

1 Introduction

Would you trust your partner when you don't trust his secrets? Suppose that Alice has a confidential message m she wants to communicate to Bob, in a way that the content of m is protected from outsiders. In a world where public key cryptography exists, Bob can sample a fresh public key pk and hands it to Alice via an authenticated channel (while keeping the corresponding secret key sk). Now Alice can use pk to encrypt m and send the resulting ciphertext c to Bob, who in turn can decrypt using sk and recover the message.

The problem sketched above, also known as the problem of *secure message transmission*, is one of the most basic questions in cryptography. For instance we

K. Kurosawa and G. Hanaoka (Eds.): PKC 2013, LNCS 7778, pp. 497–515, 2013.

know that when Bob's secret key is uniform and "well protected", any semantically secure encryption scheme would suffice for the purpose of secure message transmission. But what if (part of) Bob's secrets can leak to an outsider? Even when Bob's secret is not exposed, what if the randomness Alice used to encrypt can leak? Can Alice still trust the protocol above?

Leakage-resilient cryptography. In the last few years, questions of this kind gained momentum in the cryptographic community due to the spread of *side-channel attacks*. Starting from the early 90s', it has become clear that an adversary can potentially gain partial information on the secret state of uncorrupted players in a variety of ways, e.g. by measuring time [26], power [27] and electromagnetic emission [34]. This information, often called *leakage*, can be powerful knowledge in the hands of an adversary, putting security of the cryptographic primitive under attack on edge.

Indeed, cryptographic algorithms are typically analyzed in a black-box fashion where secrets are assumed to be completely oblivious to an adversary; in particular they offer no guarantees in the presence of side-channel attacks. To change the above state of affairs, researchers started to investigate the possibility of constructing schemes which preserve both their functionality and their security properties even in the presence of (an as large as possible class of) leakage. As a result, we now possess a rich list of leakage-resilient (a.k.a. leakage-tolerant) schemes, e.g., for pseudorandomness generation [14,32], storage [8,11], encryption [30,9], signatures [24,15] and general non-interactive circuits [22,16,13].

However, in order to have a scheme Π which maintains (in the presence of leakage) exactly the same security guarantees it has in a leak-free setting, some restriction on the leakage itself must be placed as to escape trivial attacks. Examples include putting a bound on the total information leaked, assuming that "only computation leaks information" [29], that different parts of the memory leak independently [8,11,12], that leakage occurs only in specific times or that the leakage is "hard to invert" [10]. Two general approaches have emerged:

- In the *game-based* approach, one augments the standard cryptographic game for Π by giving the adversary $\mathbb{A}$ access to an auxiliary interface from which she can input some function (within a set of admissible leakage functions) and receive back the value of the function applied to the secret state of Π.
- In the *simulation-based* approach, one shows that Π (augmented with a leakage interface) achieves the same properties of an ideal execution where a simulator $\mathbb{S}$ interacts with a functionality $\mathbf{F}$ (augmented with a leakage interface) and no communication between parties takes place. Hence, security is achieved if $\mathbb{A}$ can be simulated in the UC framework [6], i.e. for any $\mathbb{A}$ attacking Π there exists a simulator $\mathbb{S}$ such that no environment $\mathbb{Z}$ can tell whether it is interacting with $\mathbb{A}$ and Π or with $\mathbb{S}$ and $\mathbf{F}$.

Both approaches have advantages and disadvantages. Sometimes, game-based notions do not exactly capture the realistic security threats they wish to model and do not come in general with easy composition rules. Simulation-based notions are harder to achieve and often require the use of expensive tools.

The model of Bitanski, Canetti and Halevi. In this paper we focus on the second approach, building upon previous work of Bitanski et al. [5]. In their model, leakage queries from an adversary $\mathbb{A}$ are viewed as a form of partial corruption, where $\mathbb{A}$ does not receive the complete state of the chosen party but just some function $f_{\mathbb{A}}(\cdot)$ of it.

Note that without any help the simulator $\mathbb{S}$ would have a very hard life. Consider for instance the case of secure message transmission: Already a single bit of arbitrary leakage, say the first bit of the transmitted message, makes it impossible to achieve semantic security! The solution is to allow also the simulator to leak on the "ideal state" of the protocol, by specifying some function $f_{\mathbb{S}}(\cdot)$. Now, security means that a real world attacker leaking λ bits from the entire secret state of the implementation can be simulated given λ bits of leakage on the corresponding ideal state (i.e., on the message alone in case of secure message transmission).

The functionality is also able to react to leakage, in the sense that it can be asked to "give-up" on security when too much leakage occurred. This feature allows us to model relaxed security notions of protocols in the presence of leakage, and in particular to specify how the security degrades with the leakage.

1.1 Our Contribution

We revisit the context of leakage-tolerant interactive protocols. Our results give strong evidence that leakage tolerance in the simulation-based setting requires expensive tools already when a small amount of leakage needs to be tolerated. Our main contributions are outlined below:

1. For the concrete case of secure message transmission, we show that any encryption protocol Π tolerating a poly-logarithmic amount of leakage in the definition of Bitanski et al. [5] must satisfy $|\mathcal{SK}| \geq (1-\epsilon)|\mathcal{M}|$ for all $0 < \epsilon \leq 1$, where $\mathcal{M}$ is the message space and $\mathcal{SK}$ is the space of secret keys. In other words, the decryption key must be essentially as long as the message being encrypted. Furthermore, if the messages and the secret keys have the same length, then a fresh key must be used to encrypt every message.
2. We prove that Π is secure against one adaptive corruption of the receiver at the end of the protocol execution *if and only if* Π is secure against leakage of $\approx poly(\log \kappa)$ bits from the receiver's internal state at the end of the protocol execution. More in general, we prove that any n-party protocol tolerates leakage of $\approx poly(\log \kappa)$ bits from one party at the end of the protocol execution, *if and only if* the protocol has passive security against an adaptive corruption. This shows that simulation-based leakage tolerance becomes identical to full adaptive security already for very little leakage, as long as at most one party can be corrupted.
3. We further explain how to generalize our result from 2 to adaptive corruption of an *arbitrary* number of parties in a leakage-tolerant protocol.

All our results can be based on the solely assumption that collision-resistant function ensembles exist.

1.2 Our Techniques

At the heart of our results there is a novel technique exploiting succinct interactive arguments for **NP**. These are argument systems where the total amount of communication is at most poly-logarithmic in the length of the witness and the instance being proven. Succinct interactive arguments (with a constant number of rounds) are known to exist given any collision-resistant function ensemble [25,36].

Proof outline We now sketch the proof of our main result. Since protocol Π is leakage-tolerant, there exists a simulator $\mathbb{S}$ producing a "convincing" view of the protocol for $\mathbb{A}$. In addition, $\mathbb{S}$ can handle leakage queries from $\mathbb{A}$.

We exhibit an environment $\mathbb{Z}$ for which the existence of a simulator yields our bound. The environment inputs a uniformly random $m \in \mathcal{M}$. Then, it lets the protocol terminate without making any leakage query or any corruption, i.e. it simply delivers all messages between Alice (the sender) and Bob (the receiver). As part of this, $\mathbb{Z}$ learns pk and the ciphertext c from observing the communication on the authenticated channel. After the protocol terminates, $\mathbb{Z}$ asks the receiver to prove the following **NP**-statement via a succinct argument system: "There exists some sk that explains c as an encryption of m". Notice that the receiver can do this as it knows the secret key (i.e., a valid witness).

The crux of the strategy above is that $\mathbb{Z}$ can play the role of the verifier in the interactive argument by using the leakage queries on the state of the receiver to "extract" the massages of the prover.

Now, by completeness of the argument system, in the real world the proof will be accepting with overwhelming probability. On the other hand, leakage tolerance of Π implies that the simulator must cook-up an indistinguishable output in the ideal world. However, $\mathbb{S}$ has to choose c beforehand to simulate $\mathbb{A}$'s view, and later answer leakage queries consistently by "explaining" c as an encryption of m for decryption key sk. It follows from (computational) soundness of the proof system that this is only possible if for a large fraction of the messages in $\mathcal{M}$ there exists a secret key sk' which explains c consistently. From this, a simple counting argument shows that $|\mathcal{M}|$ must be negligibly close to $|\mathcal{SK}|$.

Extracting the state Let $H(\cdot)$ be a collision-resistant hash function with range μ bits. When the argument system from above is an *argument of knowledge* (i.e., there exists a knowledge extractor which is able to extract a valid witness for a statement when given access to a successful prover with respect to that statement), we are able to show that Π is leakage-tolerant against $2poly(\log \kappa) + \mu + 1$ bits of leakage from the receiver's internal state, if and only if Π has semi-honest adaptive security. The second direction follows directly from the result of [5] that adaptive (semi-honest) security is sufficient to obtain leakage tolerance for a broad class of functionalities.

To prove the first direction, one has to construct a simulator $\mathbb{S}'$ which simulates first the communication (pk, c) of the protocol to adversary $\mathbb{A}'$, and then after being given m simulates the internal state of the receiver consistently. Roughly, we do this as follows. We start by considering an adversary $\mathbb{A}$ against leakage tolerance of Π; from the definition of leakage tolerance, we know there exists a simulator $\mathbb{S}$. Hence, we use $\mathbb{S}$ to construct $\mathbb{S}'$. The adversary starts by leaking the value h obtained by applying $H(\cdot)$ on the final state of the receiver; then $\mathbb{A}$ uses an argument of knowledge to ask for a proof that there is a consistent state inside the receiver which could be extracted (consistent also with the above value of h). Now, $\mathbb{A}$ uses an additional leakage query to "send" a distinguisher $\mathbb{Z}'$ (attacking adaptive security of Π) inside the receiver, have a look at the state and output its guess b. Finally, the adversary leaks a proof that the bit b was actually computed by $\mathbb{Z}'$ from the *same* state which could be extracted from the first argument of knowledge. Note that the latter can be achieved by using the same value of h in both arguments.

It follows that if we later use a simulator $\mathbb{S}'$ for this attack and extract from its first argument of knowledge some state, this state will have to look indistinguishable from a real state to any $\mathbb{Z}'$ (as long as finding collisions in $H(\cdot)$ is hard). Adaptive security follows.

Semi-adaptive security The above proof technique is quite general, and in fact it can be applied to any leakage-tolerant interactive n-party protocol, where at most one party gets corrupted. The n-party case with arbitrary corruptions is more subtle as now a distinguisher for the adaptive security game should have access to the state of all parties when it makes its guess, and it is not clear how to simulate this given short leakages from each state. In particular, we cannot "send" a distinguisher $\mathbb{Z}'$ into each of the parties one by one, as sending $\mathbb{Z}'$ out of the parties again could require too much leakage. Indeed, in this case we do not know how to force the extracted internal states from the parties to be indistinguishable from the internal state in the real world. All that is guaranteed is that the states are consistent with the simulated public communication.

We say that Π has *semi-adaptive* security if there exist a simulator which can simulate the internal state of corrupted parties, in the sense that it can output some internal state consistent with what the party has sent and received. Notice that the state may not look indistinguishable from a real state, but it would have lead to the simulated communication. Hence, one can show that if an arbitrary interactive n-party protocol Π is able to tolerate a little leakage from t parties at the end of the execution of the protocol, then Π must be semi-adaptive secure against a semi-honest adversary which is allowed to do t adaptive corruptions.

1.3 Related Work

Simulation-based notions of leakage tolerance have been considered also for public key encryption schemes by Halevi and Lin [20] and in the context of zero-knowledge protocols by Garg et al. [18].

We mention a few other papers exploiting argument systems for negative results. The first one is the work on "seed-incompressible functions" of Halevi, Myers and Rackoff [21], who use CS proofs [28] to show that no pseudorandom function exists which remains secure after one leaks a "compressed" key. Another example is the work of [33] on parallel repetition of computationally sound proofs and the work of Jain and Pietrzak [23], who show that (game-based) leakage resilience for natural primitives like signatures and encryption does not always amplify in case of parallel repetition. The first and the last results rely on random oracles, whereas the second one is based on universal arguments [1].

We stress that the techniques used in all the above works are substantially different than ours.

2 Preliminaries

2.1 Notation

We let $\mathbf{N}$ denote the naturals and $\mathbf{R}$ denote the reals. For $a, b \in \mathbf{R}$, we let $[a, b] = \{x \in \mathbf{R} \ ; \ a \leq x \leq b\}$; for $a \in \mathbf{N}$ we let $[a] = \{1, 2, \ldots, a\}$. If x is a string, we denote its length by $|x|$; if $\mathcal{X}$ is a set, $|\mathcal{X}|$ represents the number of elements in $\mathcal{X}$. When x is chosen randomly in $\mathcal{X}$, we write $x \leftarrow \mathcal{X}$. When $\mathbb{A}$ is an algorithm, we write $y \leftarrow \mathbb{A}(x)$ to denote a run of $\mathbb{A}$ on input x and output y; if $\mathbb{A}$ is randomized, then y is a random variable and $\mathbb{A}(x; r)$ denotes a run of $\mathbb{A}$ on input x and randomness r. An algorithm $\mathbb{A}$ is *probabilistic polynomial-time* (PPT) if $\mathbb{A}$ is allowed to use randomness as part of its logic (i.e., $\mathbb{A}$ is probabilistic) and for any input $x \in \{0,1\}^*$ the computation of $\mathbb{A}(x)$ terminates in at most $poly(|x|)$ steps.

Let κ be a security parameter. A function *negl* is called *negligible* in κ (or simply negligible) if it vanishes faster than the inverse of any polynomial in κ. For a relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$, the language associated with $\mathcal{R}$ is $\mathcal{L}_{\mathcal{R}} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}\}$.

For two ensembles $\mathcal{X} = \{X_\kappa\}_{\kappa \in \mathbf{N}}, \mathcal{Y} = \{Y_\kappa\}_{\kappa \in \mathbf{N}}$, we write $\mathcal{X} \approx \mathcal{Y}$, meaning that every probabilistic polynomial-time distinguisher has negligible advantage in distinguishing $\mathcal{X}$ and $\mathcal{Y}$.

2.2 Interactive Argument Systems

Our results are based on the existence of round-efficient interactive argument systems. The definition below is taken from [36].

Definition 1 (Round-efficient interactive argument system). *An interactive protocol (P, V) is an interactive argument system for a language $\mathcal{L}$ if there is a relation $\mathcal{R}$ such that $\mathcal{L} = \mathcal{L}_{\mathcal{R}}$, and functions $\nu, s : \mathbf{N} \to [0, 1]$ such that $1 - \nu(\kappa) > s(\kappa) + 1/poly(\kappa)$ and the following holds:*

- *(Efficiency): The length of all the exchanged messages is polynomially bounded and both P and V are computable in probabilistic polynomial time;*

- *(Completeness): If* $(x, w) \in \mathcal{R}$*, then* V *accepts in* $(P(w), V)(x)$ *with probability at least* $1 - \nu(|x|)$*.*
- *(Computational soundness): If* $x \notin \mathcal{L}$*, then for every non-uniform probabilistic polynomial-time* P^* *and for all sufficiently long* $x \notin \mathcal{L}$*, the verifier* V *accepts in* $(P^*, V)(x)$ *with probability at most* $s(|x|)$*.*

The value $\nu(\cdot)$ is called the *completeness error* and the value $s(\cdot)$ is called the *soundness error*. We say (P, V) has perfect completeness if $\nu = 0$. The communication complexity of the argument system is the total length of all messages exchanged during an execution; the round complexity is the total number of exchanged messages. The protocol is called *public-coin* when the verifier's moves consist merely of tossing coins and sending their outcomes to the prover. We write $\mathbf{AM}_{\nu,s}(\rho(\kappa), \lambda(\kappa))$ to denote public-coin interactive argument systems with completeness error ν, soundness error s, round-complexity $\rho(\kappa)$ and communication complexity $\lambda(\kappa)$. Sometimes we also write $\lambda(\kappa) = \lambda_P(\kappa) + \lambda_V(\kappa)$ to differentiate between the communication complexity of the prover and of the verifier. We say (P, V) is *succinct* if $\lambda(\kappa)$ is poly-logarithmic in the length of the witness and the statement being proven.

We get an argument of knowledge whenever it is possible to extract a witness from any successful prover:

Definition 2 (Argument of knowledge). *An interactive protocol* (P, V) *is an interactive argument of knowledge for a language* $\mathcal{L}$ *if it is an interactive argument system, where the computational soundness condition is replaced by the following:*

- *(Argument of knowledge): For every non-uniform probabilistic polynomial-time* P^* *such that* V *accepts in* $(P^*, V)(x)$ with overwhelming probability*, there exists a non-uniform probabilistic polynomial-time extractor* E_{P^*} *outputting* (x, w) *such that* $(x, w) \in \mathcal{R}$ *with overwhelming probability.*

There are other forms of extractability, where from any prover succeeding to convince V with probability $p(\cdot)$, one can extract a witness with probability which is polynomially related to $p(\cdot)$ [2,35]. Here we only need the weak notion above, where extraction is only guaranteed if the prover convinces the verifier with probability close to 1. The technical reason is that in the real world we will ask a party to "leak" an argument of knowledge of its internal state, which will succeed with overwhelming probability by completeness.

Instantiations Kilian [25] constructs a 4-round public-coin succinct argument of knowledge for **NP** based on a probabilistically checkable proof (PCP) system for **NP** and a collision-resistant function ensemble. Gentry and Wichs [19] prove that non-interactive succinct arguments, so called SNARGs, cannot exist given a black-box reduction to any falsifiable assumption. In fact, the only constructions of SNARGs we know of are either based on the random oracle model of Bellare and Rogaway [3] (as shows Micali [28] by applying the Fiat-Shamir transform [17] to Kilian's protocol) or under so-called "knowledge of exponent" assumptions [4].

We remark that for our results interactive arguments are sufficient; in particular our theorems can be based on the assumption that collision-resistant function ensembles exist.

2.3 Leakage-Tolerant Secure Message Transmission

Syntax of public-key encryption A public-key encryption (PKE) scheme is a tuple of algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ defined as follows. The key generation algorithm Gen takes as input a security parameter κ and outputs $(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$; we let $\mathcal{PK} \times \mathcal{SK}$ be the key space. The encryption algorithm takes as input a message $m \in \mathcal{M}$ and outputs a ciphertext $c \leftarrow \mathsf{Enc}(pk, m)$ in some ciphertext space $\mathcal{C}$. The decryption algorithm takes as input a ciphertext $c \in \mathcal{C}$ and a secret key $sk \in \mathcal{SK}$ and outputs $m \leftarrow \mathsf{Dec}(sk, c)$.

Since we aim to apply our result to arbitrary encryption schemes, we will assume that decryption is also randomized. We say that $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has negligible completeness error if it holds that $Pr[\mathsf{Dec}(sk, (\mathsf{Enc}(pk, m)) \rightarrow m]$ with overwhelming probability over the coin tosses of $(\mathsf{Enc}, \mathsf{Dec})$ and the choices of $(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$ and $m \in \mathcal{M}$.

Leakage-tolerant PKE We recall the simulation-based notion of leakage tolerance introduce by Bitansky et al. [5]. Informally, leakage queries from an adversary $\mathbb{A}$ are viewed as a form of partial corruptions, where $\mathbb{A}$ does not received the complete state of the chosen party but just some function of it. Security is then achieved if such an adversary can be simulated in the UC framework. Without loss of generality we will consider only dummy adversaries — adversaries which just carry out the commands of the environment. I.e., it is the environment which specifies all leakage queries. We will therefore completely drop the adversary in the notation for clarity.

Let Π be a protocol implementing an ideal functionality $\mathbf{F}$. Let $\mathbb{Z}$ be an environment trying to "break" security of Π. The environment specifies all inputs to the protocol, sees all messages sent, schedules all message deliveries, sees all outputs and is in addition allowed to make leakage queries during the run of the protocol. Such queries are modelled in the following way: When $\mathbb{Z}$ wants to leak from the state of player X, it sends a leakage request $(X, f_\mathbb{Z})$ upon which it receives $f_\mathbb{Z}(\sigma_X)$, where σ_X is the current secret state of X. The function $f_\mathbb{Z}$ can be any function within a set of admissible leakage functions $\mathcal{F}$, which is a parameter in the definition.

In the ideal world, a trusted party is running $\mathbf{F}$ and a simulator $\mathbb{S}$ is interacting with it. The simulator must then simulate the protocol to the environment $\mathbb{Z}$. All inputs specified by $\mathbb{Z}$ go directly to $\mathbf{F}$; the simulator only sees the input of corrupted parties. The simulator must then simulate the communication of the protocol to $\mathbb{Z}$. In addition, all leakage queries $(\texttt{leak}, X, f_\mathbb{Z})$ from $\mathbb{Z}$ goes to the simulator. When a query $(\texttt{leak}, X, f_\mathbb{Z})$ arrives, the simulator is allowed to make its own leakage query $(\texttt{leak}, X, f_\mathbb{S})$ to the ideal functionality, under the restriction that the length of the leakage requested by $\mathbb{S}$ does not exceed the length of the leakage requested by $\mathbb{Z}$.

Functionality $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$

Running with parties R, S and adversary $\mathbb{S}$, the functionality $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$ is parametrized by the security parameter κ, message space $\mathcal{M}$ and the set of all admissible leakage functions $\mathcal{F}$. Hence, $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$ behaves as follows:

- Upon input $(\mathtt{send}, S, R, m)$ send a message $(\mathtt{send}, S, R, |m|)$ to $\mathbb{S}$. Once $\mathbb{S}$ allows to forward the message, send $(\mathtt{sent}, S, m)$ to R.
- Upon input $(\mathtt{leak}, X, f_{\mathbb{Z}})$ for $X \in \{S, R\}$ and $f_{\mathbb{Z}} \in \mathcal{F}$ send a message $(\mathtt{leak}, X)$ to $\mathbb{S}$. Receive $(\mathtt{leak}, X', f_{\mathbb{S}})$ from $\mathbb{S}$, check that $f_{\mathbb{S}} \in \mathcal{F}$, and that $|f_{\mathbb{Z}}(\cdot)| = |f_{\mathbb{S}}(\cdot)|$ and $X' = X$. Send $(\mathtt{leak}, f_{\mathbb{S}}(m))$ to $\mathbb{S}$ and $(\mathtt{leaked}, |f_{\mathbb{S}}(m)|)$ to X'.

Fig. 1. Ideal functionality $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$ for secure message transmission with leakage

We say that Π is a leakage-tolerant secure implementation of $\mathbf{F}$ if there exists a simulator $\mathbb{S}$ such that no environment can distinguish between the real life protocol Π and $\mathbb{S}$ interacting with the ideal functionality $\mathbf{F}$. More formally, consider the ideal functionality $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$, depicted in Figure 1. Denote with $\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}},\mathbb{S},\mathbb{Z}}(\mathcal{F}, \kappa)$ the output of the environment $\mathbb{Z}$ when interacting with simulator $\mathbb{S}$ in the simulation.

Consider the following protocol Π between a sender S and a receiver R, supposed to realize $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$ via a public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ and key space $\mathcal{PK} \times \mathcal{SK}$, assuming authenticated channels:

1. S transmits to R its willing to forward a message $m \in \mathcal{M}$;
2. R samples $(pk, sk) = \mathsf{Gen}(1^{\kappa}; r_G)$, where $pk \in \mathcal{PK}$ and $sk \in \mathcal{SK}$, and sends pk to S;
3. S computes $c = \mathsf{Enc}(pk, m; r_E)$ and forwards the result to R;
4. R outputs $m' = \mathsf{Dec}(sk, c; r_D)$.

Note that at the end of the execution of Π the state of S is $\sigma_S = (m, r_E)$ whereas the state of R is $\sigma_R = (sk, r_G, r_D, m')$. Denote with $\mathbf{REAL}_{\Pi,\mathbb{Z}}(\mathcal{F}, \kappa)$ the output of the environment $\mathbb{Z}$ after interacting with parties R, S in a real execution of Π.

Definition 3 (Leakage-tolerant PKE protocol). *We say that Π is a leakage-tolerant public-key encryption protocol (w.r.t. a set of leakage functions $\mathcal{F}$) if Π securely implements $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$, i.e., there exists a probabilistic polynomial-time simulator $\mathbb{S}$ such that for any environment $\mathbb{Z}$ it holds that*

$$\{\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}},\mathbb{S},\mathbb{Z}}(\mathcal{F}, \kappa)\}_{\kappa \in \mathbf{N}} \approx \{\mathbf{REAL}_{\Pi,\mathbb{Z}}(\mathcal{F}, \kappa)\}_{\kappa \in \mathbf{N}}.$$

When the total amount of leaked information is $\lambda = \sum_i |f_{\mathbb{Z}}^{(i)}(\cdot)|$, we say that Π tolerates λ bits of leakage.

3 Upper Bounds on Leakage-Tolerant PKE

In this section we present a result regarding the complexity of encryption schemes that are leakage-resilient according to Definition 3. Looking ahead, we will prove that it is not possible to achieve security in this setting without relying on an encryption scheme having similar properties to non-committing encryption [7].

Theorem 1 (Definition 3 requires long keys). *Assume the existence of* $\mathbf{AM}_{negl(\kappa),negl(\kappa)}(O(1),\lambda(\kappa))$ *argument systems for* **NP***, where* $\lambda(\kappa)=\lambda_P(\kappa)+\lambda_V(\kappa)$*. Let* Π *be a leakage-tolerant public-key encryption protocol with key space* $\mathcal{PK}\times\mathcal{SK}$ *and message space* $\mathcal{M}$*. Then, whenever* Π *tolerates* $\lambda'(\kappa)=\lambda_P(\kappa)$ *bits of leakage it must be that* $|\mathcal{SK}|\geq(1-\epsilon)|\mathcal{M}|$ *for all* $1\geq\epsilon>0$*. In particular, if* $\ell(\mathcal{SK})$ *and* $\ell(\mathcal{M})$ *are resp. the bit length of the secret key and of the messages, we have* $\ell(\mathcal{SK})\geq\ell(\mathcal{M})-1$*, i.e. to encrypt a message of length* ℓ *bits one needs a key of length at least* $\ell-1$ *bits.*

Proof. Assume first that the decryption algorithm is deterministic and that the encryption scheme has perfect correctness, i.e., $\mathsf{Dec}(sk,\mathsf{Enc}(pk,m;r_E))=m$ for all r_E when $(pk,sk)\leftarrow\mathsf{Gen}(1^\kappa)$.

Since protocol Π is leakage-tolerant, we know that there exist a simulator $\mathbb{S}$ producing a "convincing" view of the protocol. Moreover, $\mathbb{S}$ can handle requests of the kind $(\texttt{leak},X,f_{\mathbb{Z}})$, where X is either S or R and $f_{\mathbb{Z}}$ is a leakage function (chosen by the environment) to be applied to the internal state σ_X of X.

We construct an environment $\mathbb{Z}$ which uses λ_P bits of leakage on the receiver's state after the execution of Π, for which the existence of simulator $\mathbb{S}$ implies our bound. Consider the following relation:

$$\mathcal{R}:=\{((pk,c,m),(sk,r_G)):\ (pk,sk)=\mathsf{Gen}(1^\kappa;r_G)\wedge\mathsf{Dec}(sk,c)=m\}\ , \quad (1)$$

and let (P,V) be an $\mathbf{AM}_{negl(\kappa),negl(\kappa)}(O(1),\lambda(\kappa))$ argument system for $\mathcal{L}=\mathcal{L}(\mathcal{R})$. The main idea will be to let $\mathbb{Z}$ play the role of the verifier in the argument system, while running the prover with the help of the leakage queries on the state of the receiver. The environment $\mathbb{Z}$ works as follows:

1. Input a uniformly random $m\in\mathcal{M}$ to S.
2. Let the protocol terminate without any leakage queries or any corruptions, i.e., simply deliver all messages between S and R. As part of this $\mathbb{Z}$ learns pk and c from observing the authenticated channel between S and R.
3. After the protocol terminates, let R prove via leakage queries that $x=(pk,c,m)\in\mathcal{L}$. Notice that R can do this as it knows the witness $w=(sk,r_G)$. Details follow.

We now show how to generate an interactive argument for $\mathcal{L}$, by letting $\mathbb{Z}$ (holding the instance $x=(pk,c,m)$) play the role of the verifier and using the leakage queries on the receiver's state $w=(sk,r_G)$ to generate the interaction with the prover. Wlog. assume the verifier talks first, and denote with $\rho(\kappa)=poly(\kappa)$ the total number of rounds. (The case where the prover talks first can be derived similarly.)

We introduce some auxiliary notation. Let r_P (r_V) be a random string long enough to specify all random choices done by the prover (verifier), such that for fixed r_P (r_V), the prover (verifier) is deterministic. For all $i = 0, \ldots, \rho/2 - 1$, denote with $y_{2i+1} = V(x, 2i+1, view_{2i}; r_V)$ the next message sent by the verifier, where the variable $view_j$ is defined as the entire view until round $j \in [\rho]$. Similarly, the next message computed by the prover is computed as $y_{2i} = P(x, w, 2i, view_{2i-1}; r_P)$ for all $i = 1, \ldots, \rho/2$. Note that, with this notation, the complete view consists of $(y_1, y_2, \ldots, y_\rho)$. At the end the verifier computes a judgement $J(x, view_\rho; r_V) \in \{0, 1\}$, where 1 indicates accept.

Therefore, it suffices to specify how $\mathbb{Z}$ (holding only (pk, c, m)) can generate the messages of the prover. It proceeds as follows:

1. $\mathbb{Z}$ samples uniformly random r_P and r_V.
2. $\mathbb{Z}$ computes $y_1 = V(x, 1, \perp; r_V)$ and then sets the leakage function $f_{\mathbb{Z}}^{(1)}$ to be the function $f_{\mathbb{Z}}^{(1)}(w) = P(x, w, 2, y_1; r_P)$. (This can be done by "hard-wiring" the values x and y_1 into the leakage function.)
3. In general, given $view_{2i} = (y_1, y_2, \ldots, y_{2i})$, the adversary $\mathbb{Z}$ can compute y_{2i+1}, hard-wire this value into $f_{\mathbb{Z}}^{(i)}$ and get $y_{2i+2} = P(x, w, 2i+2, y_{2i+1}; r_P)$. This can be done for all $i \in [\rho]$, until the last message y_ρ of the argument system is obtained.
4. Then $\mathbb{Z}$ outputs $J(x, view_\rho; r_V)$ as its guess.

Note that the total amount of leaked information is the communication complexity of the prover in (P, V), i.e., λ_P bits. By completeness of the argument system, we know that $\mathbf{REAL}_{\Pi,\mathbb{Z}}(\mathcal{F}, \kappa) = 1$, except with negligible probability. From this we conclude that $\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}},\mathbb{S},\mathbb{Z}}(\mathcal{F}, \kappa) = 1$ except with negligible probability, by security of the protocol. We write out what this means. The simulation proceeds as follows:

1. First $\mathbb{Z}$ inputs a uniformly random $m \in \mathcal{M}$ to the ideal functionality on behalf of S. As a result $\mathbb{S}$ is given $(\texttt{send}, S, R, |m|)$.
2. Then $\mathbb{S}$ must simulate the communication of the protocol, which in particular means that it must output some pk and c to $\mathbb{Z}$.
3. After the simulation of the protocol terminates, the environment makes the leakage queries with which R proves that $x = (pk, c, m) \in \mathcal{L}$. The leakage queries are answered by $\mathbb{S}$. In more detail:
   (a) $\mathbb{Z}$ samples uniformly random r_P and r_V.
   (b) $\mathbb{Z}$ sets the leakage function $f_{\mathbb{Z}}^{(1)}$ to be the function $f_{\mathbb{Z}}^{(1)}(w) = P(x, w, 2, y_1; r_P)$. The function is sent to $\mathbb{S}$, who must choose some function $f_{\mathbb{S}}$ producing value y_2.
   (c) In general, given $view_{2i} = (y_1, y_2, \ldots, y_{2i})$, the environment $\mathbb{Z}$ specify $f_{\mathbb{Z}}^{(i)}$ and sends the same $f_{\mathbb{Z}}^{(i)}$ as in the protocol to $\mathbb{S}$ which in turn chooses $f_{\mathbb{S}}^{(i)}$ defining some y_{2i+1}. This is done for all $i \in [\rho]$, until the last message y_ρ of the argument system is obtained.
   (d) Then $\mathbb{Z}$ outputs $J(x, view_\rho; r_V)$ as its guess.

Since $\mathbb{Z}$ is computing its own messages y_{2i+1} as the verifier of (P, V) would have done, and the messages y_{2i} are computed by $\mathbb{S}$ which is PPT, and $J(x, view_\rho; r_V) = 1$, it follows from soundness that $x \in \mathcal{L}$ except with negligible probability. This means that there exist (sk, r_G) such that $(pk, sk) = \mathsf{Gen}(1^\kappa; r_G)$ and $m = \mathsf{Dec}(sk, c)$. In particular, there exist $sk \in \mathcal{SK}$ such that $m = \mathsf{Dec}(sk, c)$. Let $M_{pk,c} \subset \mathcal{M}$ denote the subset of $m' \in \mathcal{M}$ for which there exist $sk' \in \mathcal{SK}$ such that $m' = \mathsf{Dec}(sk', c)$. We have that $m \in M_{pk,c}$. Notice, that if it was the case that $m \notin M_{pk,c}$, then it would be the case that $(pk, c, m) \notin \mathcal{L}$ and hence $\mathbb{S}$ would not be able to answer the leakage queries such that $J(x, view_\rho; r_V) = 1$, except with negligible probability, by soundness. Hence, it follows from $\{\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}, \mathbb{S}, \mathbb{Z}}(\mathcal{F}, \kappa)\}_{\kappa \in \mathbf{N}} \approx \{\mathbf{REAL}_{\Pi, \mathbb{Z}}(\mathcal{F}, \kappa)\}_{\kappa \in \mathbf{N}}$ that the probability that $m \in M_{pk,c}$ is overwhelming. This implies that $|M_{pk,c}|/|\mathcal{M}|$ is negligibly close to 1, in particular $|M_{pk,c}| \geq (1 - \epsilon)|\mathcal{M}|$ for all $0 < \epsilon \leq 1$. Take two $m_0 \neq m_1 \in M_{pk,c}$. By definition there exist $sk_0, sk_1 \in \mathcal{SK}$ such that $m_0 = \mathsf{Dec}(sk_0, c)$ and $m_1 = \mathsf{Dec}(sk_1, c)$. From $m_0 \neq m_1$, we conclude that $sk_0 \neq sk_1$, so $|\mathcal{SK}| \geq |M_{pk,c}|$. From this we get the theorem.

To handle randomized decryption functions, we let the environment pick the randomness which should be used for decryption. I.e., $\mathbb{Z}$ hard-wires a random string r_D into the instance x and asks the receiver to prove that there exists r_G, sk such that $(pk, sk) = \mathsf{Gen}(1^\kappa; r_G)$ and $\mathsf{Dec}(sk, c; r_D) = m$. In the real world, this will hold with overwhelming probability, and hence in the ideal world we can, along the lines above, conclude that for any two messages m_0 and m_1, there exists $sk_0, sk_1 \in \mathcal{SK}$ such that $m_0 = \mathsf{Dec}(sk_0, c; r_D)$ and $m_1 = \mathsf{Dec}(sk_1, c; r_D)$. This again allows to conclude that $sk_0 \neq sk_1$. Note that it is important that $\mathbb{Z}$ picks r_D. If it was considered part of the witness, we would only get that there exists $sk_0, sk_1 \in \mathcal{SK}$ and r_D^0, r_D^1 such that $m_0 = \mathsf{Dec}(sk_0, c; r_D^0)$ and $m_1 = \mathsf{Dec}(sk_1, c; r_D^1)$, from which we cannot conclude that $sk_0 \neq sk_1$, as $r_D^0 \neq r_D^1$ might be enough to give different decryptions for a fixed $sk_0 = sk_1$. □

Remark 1. Assuming the existence of collision-resistant function ensembles (which implies an argument system for $\mathbf{AM}_{negl(\kappa), negl(\kappa)}(4, poly(\log \kappa))$), we get that Theorem 1 holds for any leakage-tolerant public-key encryption protocol tolerating poly-logarithmic leakage on the receiver's state.

On re-using keys One could still hope that it is possible to use the same key to encrypt more than one message. Below, we prove that this hope is also vacuous.

Corollary 1 (Fresh key for every message). *If Π is a leakage-tolerant public-key encryption protocol tolerating poly-logarithmic leakage and such that $2\ell(\mathcal{M}) - 1 > \ell(\mathcal{SK}) \geq \ell(\mathcal{M}) - 1$, then a fresh key must be used to encrypt every message.*

Proof. We prove this by contradiction to Theorem 1. Namely, assume $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has message space $\mathcal{M}$, key space $\mathcal{PK} \times \mathcal{SK}$ and uses a single pair $(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$ to encrypt two messages m' and m'' sequentially. Denote with $c' \leftarrow \mathsf{Enc}(pk, m')$ and $c'' \leftarrow \mathsf{Enc}(pk, m'')$ the corresponding ciphertexts.

Now consider the following public-key encryption scheme $\overline{\Pi} = (\overline{\mathsf{Gen}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$. The key generation algorithm $\overline{\mathsf{Gen}}$ simply runs $(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$. The encryption algorithm takes as input a message $m \in \mathcal{M}^2$, writes it as $m = m'||m''$ and outputs

$$\overline{\mathsf{Enc}}(pk, m) = \mathsf{Enc}(pk, m')||\mathsf{Enc}(pk, m'') = c'||c'' = c.$$

The decryption algorithm $\overline{\mathsf{Dec}}$ parses c as $c'||c''$ and outputs $m \leftarrow \mathsf{Dec}(sk, c')||\mathsf{Dec}(sk, c'')$.

Since Π securely realizes $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$ in the presence of λ bits of leakage, Theorem 1 implies $\ell(\mathcal{SK}) \geq \ell(\mathcal{M}) - 1$. On the other hand, the notion of leakage tolerance composes sequentially, so that $\overline{\Pi}$ securely realizes $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$ (with the same leakage bound). However, $\overline{\Pi}$ has message space $\overline{\mathcal{M}} = \mathcal{M}^2$ and key space $\overline{\mathcal{SK}} = \mathcal{SK}$. Hence, Theorem 1 yields

$$\ell(\mathcal{SK}) = \ell(\overline{\mathcal{SK}}) \geq \ell(\overline{\mathcal{M}}) - 1 = 2\ell(\mathcal{M}) - 1,$$

a contradiction.

Connection with Bitanski et al. The authors in [5] show that any non-committing encryption protocol [7] suffices to securely realize $\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}$. It is understood that every non-committing encryption protocol must satisfy the property that both the public and the secret key are as long as the total number of message bits ever encrypted [31].

4 Generalizing Our Result

It is possible to make generalizations of our results in two directions.

1. We can show that being secure against a semi-honest adversary which is allowed to do one adaptive corruption after the execution of the protocol is equivalent to being secure against a little leakage from a single party after the execution of the protocol.
2. Furthermore, say that a protocol has *semi-adaptive* security if there exists a simulator which can simulate the internal state of corrupted parties in the sense that it can output *some* internal state consistent with what the party has sent and received (but not necessarily distributed as a real-world state would be).

   We can show that for a protocol being secure against a little leakage from t parties after the execution of the protocol implies that it is semi-adaptive secure against a semi-honest adversary which is allowed to do t adaptive corruptions.

4.1 Equivalence to Adaptive Security

Assume that there exists an $\mathbf{AM}_{negl(\kappa),negl(\kappa)}(O(1), \lambda(\kappa))$ argument system, which is also an argument of knowledge. Also assume there exists a family of collision resistant hash functions $\mathcal{H} = \{H_s\}_s$ with output length $\mu(\kappa)$.

We now prove that it holds for any leakage-tolerant PKE protocol Π, as in the above section, that Π is secure against one adaptive corruption of R after the protocol execution *if and only if* Π is secure against leakage of $\approx \lambda(\kappa)+\mu(\kappa)$ bits from R after the protocol execution. Note that the above statement is clearly true when λ is large, as this would mean that the adversary is essentially leaking the entire state. Interestingly, we prove that also for a small amount of leakage (how small depends on the communication complexity of the underlying argument of knowledge) simulation-based leakage tolerance becomes identical to adaptive security.

Assume that Π is secure against one adaptive corruption of R after the protocol execution. In that case Π is also secure against any leakage queries from R after the protocol execution. This follows from [4], as leakage is weaker than adaptive corruption. We therefore focus on the other direction.

Theorem 2 (Equivalence to adaptive security). *Assume the existence of* $\mathbf{AM}_{negl(\kappa),negl(\kappa)}(O(1),\lambda(\kappa))$ *argument of knowledge systems for* **NP**, *where* $\lambda(\kappa) = \lambda_P(\kappa) + \lambda_V(\kappa)$. *Let* $\mathcal{H}$ *be afamily of collision-resistant hash functions with range* μ *and* Π *be a leakage-tolerant public-key encryption protocol. If* Π *tolerates* $\lambda'(\kappa) = 2\lambda_P(\kappa)+\mu(\kappa)+1$ *bits of leakage from* R *after the protocol execution, then* Π *is passive secure against an adaptive corruption of* R *after the protocol execution.*

Proof. For simplicity we prove the theorem in the case where decryption is deterministic. One can handle randomized decryption using the same technique as in the proof of Theorem 1.

Let $\mathbf{F}_{\mathsf{SMT}}$ be the ideal functionality for secure message transmission without leakage (featuring simulator $\mathbb{S}'$), and denote with $\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}},\mathbb{S}',\mathbb{Z}'}(\kappa)$ and $\mathbf{REAL}_{\Pi,\mathbb{Z}'}(\kappa)$ the real and ideal distributions in the adaptive security game. To prove that Π is secure against one adaptive corruption of R after the protocol execution, we have to construct a simulator $\mathbb{S}'$ such that for all environments $\mathbb{Z}'$ (corrupting R at the end of the protocol execution) and for all $\kappa \in \mathbf{N}$ it holds that $\mathbf{REAL}_{\Pi,\mathbb{Z}'}(\kappa) \approx \mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}},\mathbb{S}',\mathbb{Z}'}(\kappa)$.

Note that $\mathbb{S}'$ needs to simulate first the communication (pk, c) of the protocol, and then after being given m simulates the internal state (sk, r_G) of R. We will build $\mathbb{S}'$ by constructing an environment $\mathbb{Z}$ attacking Π in the leakage game. Then we will get a simulator $\mathbb{S}$ which can simulate the attack of $\mathbb{Z}$ in the ideal world, by the assumption that Π is secure. From $\mathbb{S}$ we will then construct $\mathbb{S}'$. For later use, $\mathbb{Z}$ will depend on an environment $\mathbb{Z}'$ for the adaptive security game. Specifically we will assume that $\mathbb{Z}'$ does a normal adaptive corruption of R after the execution of the protocol. The environment $\mathbb{Z}(\mathbb{Z}')$ runs as follows.

1. $\mathbb{Z}(\mathbb{Z}')$ runs an internal copy of $\mathbb{Z}'$.
2. Until the protocol Π is running $\mathbb{Z}$ simply runs $\mathbb{Z}'$, using the same inputs to Π and delivering messages in the same way. This is possible as the real world for leakage tolerance and adaptive security are identical as long as no leakage queries and no corruption queries are issued.
3. If $\mathbb{Z}'$ does not make an adaptive corruption of R after the execution of Π terminated, then $\mathbb{Z}$ just terminates with the same guess as $\mathbb{Z}'$.

4. If $\mathbb{Z}'$ makes an adaptive corruption of R, then $\mathbb{Z}$ proceeds as follows.
   (a) Ask R to leak $h = H_s(w)$, where $w = (sk, r_G)$ and s is a random seed for the hash family $\mathcal{H}$.
   (b) Ask R to leak an argument of knowledge of $w = (sk, r_G)$ such that $h = H_s(w)$ and $(pk, sk) = \mathsf{Gen}(1^\kappa; r_G)$ and $\mathsf{Dec}(sk, c) = m$. (This can be done exactly in the same way as in the proof of Theorem 1, by letting $\mathbb{Z}(\mathbb{Z}')$ play the role of the verifier and simulating the interaction with the prover via leakage queries.)
   (c) Let σ be the current state of $\mathbb{Z}'$. We can without loss of generality assume that $\mathbb{Z}'$ is deterministic and that it terminates with its guess b after seeing the internal state (sk, r_G, m) of R; we write $b = \mathbb{Z}'(\sigma, sk, r_G, m)$. Now $\mathbb{Z}$ leaks $f(sk, r_G) = \mathbb{Z}'(\sigma, sk, r_G, m)$. Note that $\mathbb{Z}$ knows m as this was a value it input to Π itself, and that it knows σ as it is $\mathbb{Z}$ which is running $\mathbb{Z}'$ (so these values can be hard-wired into the leakage function).
   (d) Finally ask R to leak an argument of knowledge for $w = (sk, r_G)$ such that $h = H_s(w)$ and $b = \mathbb{Z}'(\sigma, sk, r_G, m)$.
   (e) Output b.

Note that the total amount of leakage is twice the communication complexity of the prover for the arguments of knowledge, plus μ bits of H_s's output and one additional bit for the output of $\mathbb{Z}'$, i.e., $\lambda' = 2\lambda_P + \mu + 1$. By leakage tolerance, there exists a simulator $\mathbb{S}$ for the above $\mathbb{Z}(\mathbb{Z}')$. Since $\mathbb{S}$ is required to work for *all* environments, it in particular works for $\mathbb{Z}(\mathbb{Z}')$ for all $\mathbb{Z}'$, from which we get

$$\{\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}}, \mathbb{S}, \mathbb{Z}(\mathbb{Z}')}(\mathcal{F}, \kappa)\}_{\kappa \in \mathbf{N}} \approx \{\mathbf{REAL}_{\Pi, \mathbb{Z}(\mathbb{Z}')}(\mathcal{F}, \kappa)\}_{\kappa \in \mathbf{N}}, \tag{2}$$

which we use later. Note, first, however, that by leakage resilience, it holds that in the view simulated by $\mathbb{S}$, the arguments of knowledge accept with probability negligibly close to 1, or we could easily construct a distinguisher between the real world and the simulation. Furthermore, the distributions of the bit b in the real world and in the simulation are computationally indistinguishable.

Consider now the following simulator $\mathbb{S}'$, interacting with $\mathbf{F}_{\mathsf{SMT}}$ in the adaptive security game.

1. Until the protocol Π is running, simulate using $\mathbb{S}$.
2. When $\mathbb{Z}'$ adaptively corrupted R, receive m from the ideal functionality.
3. Give the leakage function $H_s(\cdot)$ to $\mathbb{S}$ to make it generate a simulated value h. Note that $\mathbb{S}$ is a simulator for the ideal world in the definition of leakage tolerance, i.e., it might issue leakage queries $f_{\mathbb{S}}$ to the ideal functionality. Answer these with $f_{\mathbb{S}}(m)$ — the trick is that $\mathbb{S}'$ at this point knows m.
4. Similarly, make $\mathbb{S}$ give an argument of knowledge of $w = (sk, r_G)$ such that $h = H_s(w)$ and $(pk, sk) = \mathsf{Gen}(1^\kappa; r_G)$ and $\mathsf{Dec}(sk, c) = m$.
5. By an above comment we know that this argument accepts except with negligible probability, so $\mathbb{S}'$ can extract from $P^* := \mathbb{S}$ a witness $w = (sk, r_G)$ such that $h = H_s(w)$ and $(pk, sk) = \mathsf{Gen}(1^\kappa; r_G)$ and $\mathsf{Dec}(sk, c) = m$.
6. Output w.

It only remains to argue that the w output by $\mathbb{S}'$ has a distribution computationally indistinguishable from the internal state of R in the real world. Assume for the sake of contradiction that it is not. Then there exists an environment $\mathbb{Z}'$ which can distinguish. This means that $b = \mathbb{Z}'(w)$ has distinguishable distributions in the real world and the simulation (for the adaptive security game). Consider then the adversary $\mathbb{Z}(\mathbb{Z}')$ for the leakage resilience game.

Claim. $\{\mathbf{REAL}_{\Pi,\mathbb{Z}(\mathbb{Z}')}(\mathcal{F},\kappa)\}_{\kappa\in\mathbf{N}} \equiv \{\mathbf{REAL}_{\Pi,\mathbb{Z}'}(\kappa)\}_{\kappa\in\mathbf{N}}$.

Proof (of claim). In words, the output distribution of $\mathbb{Z}(\mathbb{Z}')$ in the real world of the leakage game and $\mathbb{Z}'$ in the real world of the adaptive security game are the same. This follows simply by construction of $\mathbb{Z}(\mathbb{Z}')$, which runs $\mathbb{Z}'$ on the internal state w of R. □

Claim. $\{\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}}^{+\mathsf{lk}},\mathbb{S},\mathbb{Z}(\mathbb{Z}')}(\mathcal{F},\kappa)\}_{\kappa\in\mathbf{N}} \approx \{\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}},\mathbb{S}',\mathbb{Z}'}(\kappa)\}_{\kappa\in\mathbf{N}}$.

Proof (of claim). In words, the output distribution of $\mathbb{Z}(\mathbb{Z}')$ in the ideal world of the leakage game and $\mathbb{Z}'$ in the ideal world of the adaptive security game are computationally indistinguishable.

The output distribution of $\mathbb{Z}(\mathbb{Z}')$ in the ideal world of the leakage game is the value b simulated by $\mathbb{S}$. The output distribution of $\mathbb{Z}'$ in the ideal world of the adaptive security game is $\mathbb{Z}'$ applied to the value w extracted from $P^* := \mathbb{S}$. We need to prove that these two distributions are indistinguishable. To analyze the distribution of the b returned by $\mathbb{S}$ in the simulation of the leakage game, notice that since both the arguments of knowledge given by $\mathbb{S}$ are accepting, we can extract $w = (sk, r_G)$ and $w' = (sk', r'_G)$ such that $h = H_s(w)$ and $(pk, sk) = \mathsf{Gen}(1^\kappa; r_G)$ and $\mathsf{Dec}(sk, c) = m$, and $h = H_s(w')$ and $b = \mathbb{Z}'(\sigma, sk', r'_G, m)$. From $H_s(\cdot)$ being collision resistant we can assume that $w = w'$, so we conclude that it holds from the w extracted from the first argument of knowledge generated by $\mathbb{S}$ that $w = (sk, r_G)$, $(pk, sk) = \mathsf{Gen}(1^\kappa; r_G)$, $\mathsf{Dec}(sk, c) = m$ and $b = \mathbb{Z}'(\sigma, sk, r_G, m)$. This means that unless the collision resistance of $H_s(\cdot)$ is broken, the output distribution of $\mathbb{Z}(\mathbb{Z}')$ in the ideal world of the leakage game and $\mathbb{Z}'$ in the ideal world of the adaptive security game are the same. □

The two claims above together with the assumption that $\mathbb{Z}'$ can distinguish, imply that that $\mathbb{Z}(\mathbb{Z}')$ has distinguishable outputs in the real world and the ideal world for the leakage game, contradicting Eq. (2) above. From this we conclude that $\{\mathbf{REAL}_{\Pi,\mathbb{Z}'}(\kappa)\}_{\kappa\in\mathbf{N}} \approx \{\mathbf{IDEAL}_{\mathbf{F}_{\mathsf{SMT}},\mathbb{S}'\mathbb{Z}'}(\kappa)\}_{\kappa\in\mathbf{N}}$ for all environments $\mathbb{Z}'$, which proves the theorem. □

4.2 Equivalence to Semi-adaptive Security for Many Parties

We note that the proof technique from the previous section can be easily generalized to show that an arbitrary two-party protocol Π is secure against one adaptive corruption after the protocol execution *if and only if* Π tolerates $\approx poly(\log \kappa)$ bits of leakage from one of the parties after the protocol execution.

A variant of the above proof technique works also for an arbitrary protocol and if we allow that many parties can be corrupted/leaked from after the protocol execution. The environment will ask each party to leak an argument of knowledge of an internal state consistent with its inputs and outputs. A simulator which can simulate such an argument could also "by extracting itself" have output the entire internal state. We cannot, however, perform the trick where we send the distinguisher $\mathbb{Z}'$ into the parties to leak $\mathbb{Z}'(w)$, as now a distinguisher for the adaptive security game should have access to $(w_1, \ldots, w_n)$, where w_i is the internal state of party i, and $(w_1, \ldots, w_n)$ is not sitting inside a single party, so $\mathbb{Z}'(w_1, \ldots, w_n)$ cannot *per se* be computed using short leakages $f_1(w_1), \ldots, f_n(w_n)$. Hence we cannot force the extracted internal state to be indistinguishable from the internal state in the real world, all that is guaranteed is that the state is consistent with the simulated public communication.

Acknowledgements. The authors acknowledge support from the Danish National Research Foundation and The National Science Foundation of China (under the grant 61061130540) for the Sino-Danish Center for the Theory of Interactive Computation, and also from the CFEM research center (supported by the Danish Strategic Research Council) within which part of this work was performed.

References

1. Barak, B., Goldreich, O.: Universal arguments and their applications. SIAM J. Comput. 38(5), 1661–1694 (2008)
2. Bellare, M., Goldreich, O.: On Defining Proofs of Knowledge. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 390–420. Springer, Heidelberg (1993)
3. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security, pp. 62–73 (1993)
4. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: ITCS, pp. 326–349 (2012)
5. Bitansky, N., Canetti, R., Halevi, S.: Leakage-Tolerant Interactive Protocols. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 266–284. Springer, Heidelberg (2012)
6. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: FOCS, pp. 136–145 (2001)
7. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: STOC, pp. 639–648 (1996)
8. Davì, F., Dziembowski, S., Venturi, D.: Leakage-Resilient Storage. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 121–137. Springer, Heidelberg (2010)
9. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Efficient Public-Key Cryptography in the Presence of Key Leakage. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 613–631. Springer, Heidelberg (2010)
10. Dodis, Y., Kalai, Y.T., Lovett, S.: On cryptography with auxiliary input. In: STOC, pp. 621–630 (2009)

11. Dodis, Y., Lewko, A.B., Waters, B., Wichs, D.: Storing secrets on continually leaky devices. In: FOCS, pp. 688–697 (2011)
12. Dziembowski, S., Faust, S.: Leakage-Resilient Cryptography from the Inner-Product Extractor. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 702–721. Springer, Heidelberg (2011)
13. Dziembowski, S., Faust, S.: Leakage-Resilient Circuits without Computational Assumptions. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 230–247. Springer, Heidelberg (2012)
14. Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: FOCS, pp. 293–302 (2008)
15. Faust, S., Kiltz, E., Pietrzak, K., Rothblum, G.N.: Leakage-Resilient Signatures. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 343–360. Springer, Heidelberg (2010)
16. Faust, S., Rabin, T., Reyzin, L., Tromer, E., Vaikuntanathan, V.: Protecting Circuits from Leakage: the Computationally-Bounded and Noisy Cases. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 135–156. Springer, Heidelberg (2010)
17. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
18. Garg, S., Jain, A., Sahai, A.: Leakage-Resilient Zero Knowledge. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 297–315. Springer, Heidelberg (2011)
19. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: STOC, pp. 99–108 (2011)
20. Halevi, S., Lin, H.: After-the-Fact Leakage in Public-Key Encryption. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 107–124. Springer, Heidelberg (2011)
21. Halevi, S., Myers, S., Rackoff, C.: On Seed-Incompressible Functions. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 19–36. Springer, Heidelberg (2008)
22. Ishai, Y., Sahai, A., Wagner, D.: Private Circuits: Securing Hardware against Probing Attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (2003)
23. Jain, A., Pietrzak, K.: Parallel Repetition for Leakage Resilience Amplification Revisited. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 58–69. Springer, Heidelberg (2011)
24. Katz, J., Vaikuntanathan, V.: Signature Schemes with Bounded Leakage Resilience. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 703–720. Springer, Heidelberg (2009)
25. Kilian, J.: A note on efficient zero-knowledge proofs and arguments (extended abstract). In: STOC, pp. 723–732 (1992)
26. Kocher, P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
27. Kocher, P.C., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
28. Micali, S.: Computationally sound proofs. SIAM J. Comput. 30(4), 1253–1298 (2000)
29. Micali, S., Reyzin, L.: Physically Observable Cryptography (Extended Abstract). In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 278–296. Springer, Heidelberg (2004)
30. Naor, M., Segev, G.: Public-key cryptosystems resilient to key leakage. IACR Cryptology ePrint Archive, 2009:105 (2009)

31. Nielsen, J.B.: Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 111–126. Springer, Heidelberg (2002)
32. Pietrzak, K.: A Leakage-Resilient Mode of Operation. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 462–482. Springer, Heidelberg (2009)
33. Pietrzak, K., Wikström, D.: Parallel repetition of computationally sound protocols revisited. J. Cryptology 25(1), 116–135 (2012)
34. Quisquater, J.-J., Samyde, D.: ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards. In: Attali, S., Jensen, T. (eds.) E-smart 2001. LNCS, vol. 2140, pp. 200–210. Springer, Heidelberg (2001)
35. Unruh, D.: Quantum Proofs of Knowledge. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 135–152. Springer, Heidelberg (2012)
36. Wee, H.: On Round-Efficient Argument Systems. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 140–152. Springer, Heidelberg (2005)

Author Index

Abdalla, Michel 292
Abe, Masayuki 312
Attrapadung, Nuttapong 386

Barbosa, Manuel 143
Barbu, Guillaume 198
Battistello, Alberto 198
Ben Hamouda, Fabrice 272, 292
Blazy, Olivier 272
Brakerski, Zvika 1

Catalano, Dario 55
Chase, Melissa 479
Chevalier, Céline 272
Choi, Seung Geol 73

Dabosville, Guillaume 198
Dagdelen, Özgür 461
David, Bernardo 312

Emura, Keita 32, 216

Farshim, Pooya 143, 352
Fiore, Dario 55
Freire, Eduarda S.V. 254

Gentry, Craig 1
Giraud, Christophe 198

Halevi, Shai 1
Hanaoka, Goichiro 32, 332
Hofheinz, Dennis 254
Hohenberger, Susan 162
Huang, Zhengan 369

Izu, Tetsuya 180

Katz, Jonathan 14, 73
Kiltz, Eike 254
Kohlweiss, Markulf 312, 479
Kumarasubramanian, Abishek 89
Kunihiro, Noboru 180

Lee, Dong Hoon 423
Lee, Kwangsu 423
Libert, Benoît 352, 386
Lindell, Yehuda 253
Ling, San 107
Liu, Shengli 369
Lysyanskaya, Anna 479

Matsuda, Takahiro 32, 332
Meiklejohn, Sarah 479
Mohassel, Payman 461

Nguyen, Khoa 107
Nielsen, Jesper Buus 497
Nishimaki, Ryo 312, 405

Ohkubo, Miyako 312
Ohtake, Go 32
Okamoto, Tatsuaki 125
Ostrovsky, Rafail 89

Pandey, Omkant 89
Paterson, Kenneth G. 254, 352
Peters, Thomas 386
Pointcheval, David 272, 292

Qin, Baodong 369
Quaglia, Elizabeth A. 352

Renault, Guénaël 198
Renner, Soline 198

Seo, Jae Hong 216
Seurin, Yannick 443
Shinohara, Naoyuki 180
Stehlé, Damien 107

Takashima, Katsuyuki 125
Thiruvengadam, Aishwarya 14

Venturi, Daniele 461, 497
Vergnaud, Damien 272

Wadia, Akshay 89
Wang, Huaxiong 107
Waters, Brent 51, 162
Wee, Hoeteck 73

Xagawa, Keita 235, 405

Yamada, Shota 32
Yung, Moti 423

Zeitoun, Rina 198
Zhou, Hong-Sheng 14, 73
Zottarel, Angela 497