Phong Q. Nguyen
David Pointcheval (Eds.)

LNCS 6056

Public Key Cryptography – PKC 2010

13th International Conference
on Practice and Theory in Public Key Cryptography
Paris, France, May 2010, Proceedings

INTERNATIONAL ASSOCIATION FOR CRYPTOLOGIC RESEARCH

Springer

Phong Q. Nguyen   David Pointcheval (Eds.)

Public Key Cryptography – PKC 2010

13th International Conference
on Practice and Theory in Public Key Cryptography
Paris, France, May 26-28, 2010
Proceedings

Springer

Volume Editors

Phong Q. Nguyen
David Pointcheval
École Normale Supérieure
Département d'Informatique
45 rue d'Ulm, 75230 Paris Cedex 05, France
E-mail: {phong.nguyen, david.pointcheval}@ens.fr

Library of Congress Control Number: 2010926287

CR Subject Classification (1998): E.3, K.6.5, C.2, D.4.6, K.4.4, E.4

LNCS Sublibrary: SL 4 – Security and Cryptology

ISSN	0302-9743
ISBN-10	3-642-13012-7 Springer Berlin Heidelberg New York
ISBN-13	978-3-642-13012-0 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

The 13th International Conference on Practice and Theory in Public Key Cryptography (PKC 2010) was held May 26–28, 2010, at the École Normale Supérieure (ENS) in Paris, France. PKC 2010 was sponsored by the International Association for Cryptologic Research (IACR), in cooperation with the *École Normale Supérieure* (ENS) and the *Institut National de Recherche en Informatique et en Automatique* (INRIA). The General Chairs of the conference were Michel Abdalla and Pierre-Alain Fouque.

The conference received a record number of 145 submissions and each submission was assigned to at least 3 committee members. Submissions co-authored by members of the Program Committee were assigned to at least five committee members. Due to the large number of high-quality submissions, the review process was challenging and we are deeply grateful to the 34 committee members and the 163 external reviewers for their outstanding work. After extensive discussions, the Program Committee selected 29 submissions for presentation during the conference and these are the articles that are included in this volume. The best paper was awarded to Petros Mol and Scott Yilek for their paper "Chosen-Ciphertext Security from Slightly Lossy Trapdoor Functions." The review process was run using the iChair software, written by Thomas Baignères and Matthieu Finiasz from EPFL, LASEC, Switzerland, and we are indebted to them for letting us use their software.

The program also included two invited talks: it was a great honor to have Daniele Micciancio and Jacques Stern as invited speakers. Their talks were entitled, respectively, "Duality in Lattice Based Cryptography" and "Mathematics, Cryptography, Security." We would like to genuinely thank them for accepting our invitation and for contributing to the success of PKC 2010.

Finally, we would like to thank our sponsors Google, Ingenico, and Technicolor for their financial support and all the people involved in the organization of this conference. In particular, we would like to thank the Office for Courses and Colloquiums (*Bureau des Cours-Colloques*) from INRIA and Gaëlle Dorkeld, as well as Jacques Beigbeder and Joëlle Isnard from ENS, for their diligent work and for making this conference possible. We also wish to thank Springer for publishing the proceedings in the *Lecture Notes in Computer Science* series.

May 2010

Phong Q. Nguyen
David Pointcheval

PKC 2010

13th International Conference on
Practice and Theory in Public Key Cryptography

Paris, France, May 26–28, 2010

General Chairs

Michel Abdalla	CNRS and ENS, Paris, France
Pierre-Alain Fouque	ENS, Paris, France

Program Chairs

Phong Q. Nguyen	INRIA and ENS, Paris, France
David Pointcheval	CNRS, ENS and INRIA, Paris, France

Program Committee

Alexandra Boldyreva	Georgia Institute of Technology, USA
Xavier Boyen	University of Liege, Belgium
Dario Catalano	University of Catania, Italy
Jung Hee Cheon	Seoul National University, South Korea
Jean-Sébastien Coron	University of Luxembourg
Marc Fischlin	TU Darmstadt, Germany
Eiichiro Fujisaki	NTT Labs, Japan
Craig Gentry	IBM, USA
Maria Isabel Gonzalez Vasco	Universidad Rey Juan Carlos, Madrid, Spain
Stanislaw Jarecki	UC Irvine, California, USA
Jonathan Katz	University of Maryland, USA
Eike Kiltz	CWI, The Netherlands
Fabien Laguillaumie	University of Caen, France
Dong Hoon Lee	Korea University, Seoul, South Korea
Reynald Lercier	DGA/CELAR and University of Rennes, France
Benoît Libert	Université Catholique de Louvain, Belgium
Vadim Lyubashevsky	University of Tel-Aviv, Israel
Mark Manulis	TU Darmstadt and CASED, Germany
Alfred Menezes	University of Waterloo, Canada
Kenny Paterson	Royal Holloway, University of London, UK
Duong Hieu Phan	University of Paris 8, France
Benny Pinkas	University of Haifa, Israel
Alon Rosen	IDC Herzliya, Israel
Kazue Sako	NEC, Japan

Hovav Shacham	UC San Diego, California, USA
Igor Shparlinski	University of Macquarie, Sydney, Australia
Martijn Stam	EPFL, Switzerland
Keisuke Tanaka	Tokyo Institute of Technology, Japan
Ramarathnam Venkatesan	Microsoft Research, Bangalore and Redmond, India and USA
Damien Vergnaud	ENS, Paris, France
Ivan Visconti	University of Salerno, Italy
Bogdan Warinschi	Bristol University, UK
Brent Waters	University of Texas, USA
Duncan Wong	City University of Hong Kong, China

External Reviewers

Michel Abdalla
Divesh Aggarwal
Shweta Agrawal
Adi Akavia
Koichiro Akiyama
Frederik Armknecht
Ali Bagherzandi
Aurélie Bauer
Amos Beimei
Daniel J. Bernstein
Raghav Bhaskar
James Birkett
Jens-Matthias Bohli
Joppe Bos
Charles Bouillaguet
John Boxall
Emmanuel Bresson
Jin Wook Byun
David Cash
Guilhem Castagnos
Julien Cathalo
Pierre-Louis Cayrel
Sanjit Chatterjee
Céline Chevalier
Kwantae Cho
Kyu Young Choi
Raymond Choo
Ji Young Chun
Cas Cremers
Maria Cristina Onete
Özgür Dagdelen
Vanesa Daza
Sebastiaan de Hoogh
Cécile Delerablée
Olivier de Marneffe
Breno de Medeiros
Alexander W. Dent
Claus Diem
Mario Di Raimondo
Vivien Dubois
Laila El Aimani
Nadia El Mrabet
Pooya Farshim
Anna Lisa Ferrara
Dario Fiore
Jun Furukawa
David Galindo
Nicolas Gama
Essam Ghadafi
Domingo Gomez Perez
Choudary Gorantla
Vipul Goyal
Robert Granger
Matthew Green
Thomas Gross
Jens Groth
Jaime Gutierrez
Daewan Han
Darrel Hankerson
Carmit Hazay
Brett Hemenway
Javier Herranz

Mathias Herrmann
Dennis Hofheinz
Thomas Holenstein
Jeongdae Hong
Qiong Huang
Jung Yeon Hwang
Thomas Icart
Toshiyuki Isshiki
Malika Izabachène
Tibor Jager
Ayman Jarrous
Haimin Jin
Seny Kamara
Koray Karabina
Akinori Kawachi
Yutaka Kawai
Mitsuru Kawazoe
Jihye Kim
Kitak Kim
Minkyu Kim
Myungsun Kim
Woo Kwon Koo
Takeshi Koshiba
Hugo Krawczyk
Virendra Kumar
Robin Künzler
Benoît Larroque
Hyung Tae Lee
Ji-Seon Lee
Kwangsu Lee
Munkyu Lee
Anja Lehmann
Arjen K. Lenstra
Allison Lewko
Yehuda Lindell
Xiaomin Liu
Satya Lokam
Julio Lopez
Xizhao Luo
Lior Malka
Toshihide Matsuda
Payman Mohassel
Tal Moran
Michael Naehrig
Toru Nakanishi
Gregory Neven
Ryo Nishimaki
Yasuyuki Nogami
Tatsuaki Okamoto
Josh Olsen
Adam O'Neill
Claudio Orlandi
Alina Ostafe
Adriana Palacio
Omkant Pandey
C. Pandu Rangan
Hyun-A Park
Jehong Park
Jong Hwan Park
Sylvain Pasini
Chris Peikert
Olivier Pereira
Angel L. Perez del Pozo
Bertram Poettering
Hyun Sook Rhee
Maike Ritzenhofen
Ben Riva
Francisco Rodriguez-Henriquez
Yannis Rouselakis
Ahmad-Reza Sadeghi
Alessandra Scafuro
Thomas Schneider
Berry Schoenmakers
Dominique Schröder
Michael Scott
Jae Hong Seo
Elaine Shi
Thomas Sirvent
William Skeith
Damien Stehlé
Mario Strefler
Willy Susilo
Koutarou Suzuki
Tamir Tassa
Edlyn Teske-Wilson
Berkant Ustaoglu
Vinod Vaikuntanathan
Carmine Ventre
Jorge L. Villar
Panagiotis Voulgaris

Christian Wachsmann
Christopher Wolf
Keita Xagawa
Xiaokang Xiong
Guomin Yang
Scott Yilek
Kazuki Yoneyama
Tsz Hon Yuen
Aaram Yun
Zongyang Zhang
Vassilis Zikas

Sponsors

Financial support by the following sponsors is gratefully acknowledged:

- ENS
- Google
- Ingenico
- Technicolor

Table of Contents

Simple and Efficient Public-Key Encryption from Computational Diffie-Hellman in the Standard Model

Kristiyan Haralambiev[1,⋆], Tibor Jager[2], Eike Kiltz[3,⋆⋆], and Victor Shoup[4,⋆⋆⋆]

[1] Dept. of Computer Science, New York University, Courant Institute, 251 Mercer Street, New York, NY 10012, USA
kkh@cs.nyu.edu

[2] Horst Görtz Institute for IT Security, Ruhr-University Bochum, Germany
tibor.jager@rub.de

[3] Cryptology & Information Security Group, CWI, Amsterdam, The Netherlands
kiltz@cwi.nl

[4] Dept. of Computer Science, New York University, Courant Institute, 251 Mercer Street, New York, NY 10012, USA
shoup@cs.nyu.edu

Abstract. This paper proposes practical chosen-ciphertext secure public-key encryption systems that are provably secure under the *computational* Diffie-Hellman assumption, in the standard model. Our schemes are conceptually simpler and more efficient than previous constructions. We also show that in bilinear groups the size of the public-key can be shrunk from n to $2\sqrt{n}$ group elements, where n is the security parameter.

1 Introduction

Security against chosen-ciphertext attack (CCA) is nowadays considered to be the standard security notion for public-key encryption. In this work we are interested in practical schemes with proofs of security under mild security assumptions (such as the computational Diffie-Hellman assumption), without relying on heuristics such as the random oracle model [2].

ElGamal Encryption. Let $\mathbb{G}$ be a cyclic group generated by g. The ElGamal encryption scheme, described as a key-encapsulation mechanism $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, is as follows

$$\mathsf{Gen} : sk = z, pk = Z = g^z, \quad \mathsf{Enc}(pk) : C = g^r, K = Z^r,$$
$$\mathsf{Dec}(sk, C) : K = C^z \in \mathbb{G},$$

where all appearing exponents are chosen at random. It can be proved one-way (OW-CPA) secure under the computational Diffie-Hellman (DH) assumption,

⋆ Supported by NSF award number CNS-0716690.

⋆⋆ Supported by the research program Sentinels.

⋆⋆⋆ Supported by NSF award number CNS-0716690.

but its semantic (IND-CPA) security is equivalent to the stronger DDH assumption. To obtain an IND-CPA secure variant from the DH assumption one commonly uses the Goldreich-Levin [13] hard-core predicate $f_{\mathrm{gl}}(\cdot, R)$ with randomness R to extract a pseudorandom bit from the Diffie-Hellman seed. By a standard randomness-reusing technique one obtains a scheme that encapsulates n-bit keys:

$$\begin{array}{ll}\mathsf{Gen}_{\mathsf{dh}}: & sk_{\mathsf{dh}} = (z_1, \ldots, z_n), \quad pk_{\mathsf{dh}} = (Z_1 = g^{z_1}, \ldots, Z_n = g^{z_n}), \\ \mathsf{Enc}(pk): & C_{\mathsf{dh}} = g^r, \quad K_{\mathsf{dh}} = (f_{\mathrm{gl}}(Z_1^r, R), \ldots, f_{\mathrm{gl}}(Z_n^r, R)) \in \{0,1\}^n,\end{array} \tag{1}$$

where decapsulation reconstructs the seed values Z_i^r by computing $Z_i^r = C_{\mathsf{dh}}^{z_i}$. Combined with a one-time pad it yields an IND-CPA secure encryption scheme.

IND-CCA SECURITY FROM DECISIONAL ASSUMPTIONS. Whereas CPA-secure schemes can be constructed generically, building CCA-secure schemes seems more difficult and usually requires stronger hardness assumptions. The first practical CCA-secure encryption scheme (without random oracles) was proposed in a seminal paper by Cramer and Shoup [10]. Their construction was later generalized to hash proof systems [9]. However, the Cramer-Shoup encryption scheme and all its variants [22,7,20,21,16,17] inherently rely on *decisional assumption*, e.g., the Decisional Diffie-Hellman (DDH) assumption or the quadratic residuosity assumption. Moreover, there are groups, such as certain elliptic curve groups with bilinear pairing map, where the DDH assumption does not hold, but the DH problem appears to be hard.

IND-CCA SECURITY FROM COMPUTATIONAL ASSUMPTIONS. The DDH assumption has often been criticized as being too strong [3,12] and in general wrong in certain cryptographically relevant groups [19]. Schemes based on the DH assumption are preferred but, surprisingly, even with strong tools such as the Cramer Shoup framework [10] such schemes seem to be hard to obtain.

Canetti, Halevi and Katz [5] proposed the first practical public-key encryption scheme based on a computational assumption, namely the Bilinear DH assumption in bilinear groups. Later, as a general tool to construct secure cryptographic primitives against active attacks, Cash *et al.* [8] proposed the Twin Diffie-Hellman (2DH) assumption. Though seemingly a stronger assumption, the *interactive* Strong 2DH assumption (which is the 2DH assumption where the adversary is additionally given an oracle that solves the 2DH problem for fixed bases) is implied by the standard DH assumption. Building on "IBE techniques" [4,5], Cash *et al.* obtained the first practical encryption scheme which is CCA-secure assuming the strong 2DH assumption, and therefore also assuming the standard DH assumption. Here the decisional 2DH oracle provided by Strong 2DH assumption plays a crucial role in distinguishing consistent from non-consistent ciphertexts. However, to prove IND-CCA security, [8] had to add n group elements to the ciphertext of the scheme from Equation (1) which renders the scheme quite impractical. In independent work, Hanaoka and Kurosawa [14] used a different approach based on broadcast encryption, and could thereby reduce the number of group elements in the ciphertexts to a constant. According to [14], their approach is not based on the twinning framework.

Recently, Hofheinz and Kiltz gave a CCA-secure encryption scheme based on the factoring assumption [18].

1.1 Our Contributions

In this paper we propose a number of new encryption schemes that are CCA-secure assuming the standard DH assumption. We apply the Twin Diffie-Hellman framework from [8] to the CPA-secure scheme given in Equation (1). Therefore our schemes are simple and intuitive. As summarized in [15, Table 1], they improve efficiency of prior schemes from [8,14].

A scheme from Strong DH. To illustrate our main ideas we first give a toy scheme that is IND-CCA secure assuming the *Strong DH assumption* [1] (The Strong DH assumption is that the DH assumption holds when the adversary is equipped with a (fixed-base) DDH oracle.) This is essentially the same scheme as ElGamal from Equation (1), but one more group element is added to the ciphertext.

$$\begin{array}{ll}\mathsf{Gen}_{\mathsf{sdh}}: & sk = (sk_{\mathsf{dh}}, x, x'), \quad pk = (pk_{\mathsf{dh}}, X = g^x, X' = g^{x'}) \\ \mathsf{Enc}_{\mathsf{sdh}}(pk): & C = (C_{\mathsf{dh}}, (X^t X')^r),\ K = K_{\mathsf{dh}},\end{array} \tag{2}$$

where $t = \mathsf{T}(C_{\mathsf{dh}})$ is the output of a target collision resistant hash function. Decryption only returns K if the ciphertext $C = (C_0, C_1)$ is consistent, i.e., if $C_0^{xt+x'} = C_1$. In all other cases it rejects and returns $\perp$. The additional element $(X^t X')^r$ from the ciphertext is used as a handle for an all-but-one simulation technique (based on techniques from identity-based encryption [4]) to be able to simulate the decryption oracle for all ciphertexts, except the challenge ciphertext. The above simulation technique works only if consistent ciphertexts can be distinguished from inconsistent ones, which is why we need the DDH oracle provided by the Strong DH assumption.

First scheme from DH. Our first scheme, which is secure under the (standard) DH assumption, applies the twinning framework to the above idea by adding an additional element $(Y^t Y')^r$ to the ciphertext.

$$\begin{array}{ll}\mathsf{Gen}_{\mathsf{dh1}}: & sk = (sk_{\mathsf{dh}}, x, x', y, y'), \\ & pk = (pk_{\mathsf{dh}}, X = g^x, X' = g^{x'}, Y = g^y, Y' = g^{y'}) \\ \mathsf{Enc}_{\mathsf{dh1}}(pk): & C = (C_{\mathsf{dh}}, (X^t X')^r, (Y^t Y')^r), \\ & K = K_{\mathsf{dh}}.\end{array} \tag{3}$$

Again, decryption only returns K if the ciphertext is consistent, and $\perp$ otherwise. By analogy to the scheme from Equation (2) it is IND-CCA secure under the Strong 2DH assumption which, by the Twinning theorem from [8], is implied by the standard DH assumption. Again, the Decisional 2DH oracle provided by the Strong DH assumption is crucial for distinguishing consistent from inconsistent ciphertexts in the reduction.

Second scheme from DH. Our second scheme from the DH assumption applies an “implicit rejection technique” to remove the second element from the ciphertext.

$$\begin{aligned} \mathsf{Gen}_{\mathsf{dh2}}: \quad & sk = (sk_{\mathsf{dh}}, x, x', y, y'), \\ & pk = (pk_{\mathsf{dh}}, X = g^x, X' = g^{x'}, Y = g^y, Y' = g^{y'}) \\ \mathsf{Enc}_{\mathsf{dh2}}(pk): & C = (C_{\mathsf{dh}}, (X^t X')^r), \\ & K = K_{\mathsf{G}} \oplus K_{\mathsf{dh}}, \text{ where } K_{\mathsf{G}} = \mathsf{G}((Y^t Y')^r), \end{aligned} \tag{4}$$

where $\mathsf{G} : \mathbb{G} \to \{0,1\}^n$ is a secure pseudorandom generator. Decryption only returns K if the ciphertext $C = (C_0, C_1)$ is consistent, i.e., if $C_0^{xt+x'} = C_1$. In that case K_{G} is computed as $K_{\mathsf{G}} = \mathsf{G}(C_0^{yt+y'})$. Unfortunately, we are not able to show full CCA security of this KEM but, instead, we are able to prove the weaker constrained CCA (CCCA) security [16] under the DH assumption. A CCCA-secure KEM plus a symmetric authenticated encryption scheme (i.e., a MAC plus a one-time pad) yields CCA-secure encryption. The intuition behind the security is similar to the scheme from Equation (3) with the difference that, during the simulation, the values Y and Y' are set-up such that, if the ciphertext is inconsistent, then the simulated decryption will produce K_{G} that is uniform in the adversary's view and therefore $K = K_{\mathsf{G}} \oplus K_{\mathsf{dh}}$ is also uniform. Consequently, when combined with symmetric authenticated encryption such inconsistent decryption queries will get rejected by the symmetric cipher.

Reducing the size of the Public-Keys. Our schemes are quite practical, except for the large public-key which consists of $\approx n$ group elements. We also propose two methods to reduce the size of the public-key when our schemes are instantiated over bilinear groups. Most interestingly, we note that the public-key can be shrunk from n to $2\sqrt{n}$ elements by "implicitly defining" the n elements of pk_{dh} as $Z_{i,j} := \hat{e}(Z_i, Z'_j)$, for $i, j \in [1, \sqrt{n}]$. (Here $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a symmetric bilinear map.) Note that now only the $2\sqrt{n}$ elements Z_i, Z'_j need to be stored in the public-key.[1] Furthermore, in bilinear groups it is also possible to move the n values $Z_1, \ldots, Z_n$ from the public-key pk_{dh} into the system parameter that can be shared among many users. In that case the public-key only contains one group element, but the system parameters are still of size $\approx n$. We remark that the observation of putting public-key elements into the systems parameters is not new and has been made before, e.g., for Water's IBE scheme [24]. Finally, we also sketch how our ideas can be extended to construct an IBE scheme. All our bilinear constructions are CCA secure under the Bilinear DH (BDH) assumption.

2 Preliminaries

2.1 Notation

In the following we let $(\mathbb{G}_\kappa)_{\kappa\in\mathbb{N}}$ be a family of prime-order groups, indexed by security parameter κ. Occasionally we write $\mathbb{G}$ shorthand for some group $\mathbb{G}_\kappa \in (\mathbb{G}_\kappa)_{\kappa\in\mathbb{N}}$, when the reference to the security parameter κ is clear. We denote with

[1] We remark that this is a generic technique that may also be applied to other Diffie-Hellman based constructions suffering from large public keys, such as the DDH-based lossy trapdoor functions in [23,11].

$\text{poly}(\kappa)$ an unspecified positive integer-valued polynomial, and with $\text{negl}(\kappa)$ a negligible function in κ, that is, $|\text{negl}(\kappa)| < o(1/\kappa^c)$ for every positive integer c. For a positive integer n, we denote with $[n]$ the set $[n] = \{1, \dots, n\}$.

2.2 Key Encapsulation Mechanisms

Let $n = n(\kappa)$ be a polynomial. A *key-encapsulation mechanism* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with key-space $\{0,1\}^n$ consists of three polynomial-time algorithms (PTAs). Via $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$ the randomized key-generation algorithm produces public/secret keys for security parameter $\kappa \in \mathbb{N}$; via $(C, K) \leftarrow \mathsf{Enc}(pk)$ the randomized encapsulation algorithm creates an uniformly distributed symmetric key $K \in \{0,1\}^n$, together with a ciphertext C; via $K \leftarrow \mathsf{Dec}(sk, C)$ the possessor of secret key sk decrypts ciphertext C to get back a key K which is an element in $\{0,1\}^n$ or a special rejection symbol $\perp$. For consistency, we require that for all $\kappa \in \mathbb{N}$, and all $(C, K) \leftarrow \mathsf{Enc}(pk)$ we have $\Pr[\mathsf{Dec}(sk, C) = K] = 1$, where the probability is taken over the choice of $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$, and the coins of all the algorithms in the expression above.

Chosen-Ciphertext Security. The common requirement for a KEM is indistinguishability against chosen-ciphertext attacks (IND-CCA) [10] where an adversary is allowed to adaptively query a decapsulation oracle with ciphertexts to obtain the corresponding session key. More formally, for an adversary $\mathcal{A}$ we define the advantage function

$$\mathsf{AdvCCA}^{\mathcal{A}}_{\mathsf{KEM}_{\mathsf{dh1}}}(\kappa) := \Pr\left[b = b' \; : \; \begin{array}{l} (pk, sk) \leftarrow \mathsf{Gen}(1^n) \\ (C, K_0) \leftarrow \mathsf{Enc}(pk) \\ K_1 \leftarrow \{0,1\}^n;\; b \leftarrow \{0,1\} \\ b' \leftarrow \mathcal{A}^{\mathsf{Dec}(\cdot)}(pk, K_b, C) \end{array} \right] - \frac{1}{2},$$

where oracle $\mathsf{Dec}(C_i)$ returns $K_i \leftarrow \mathsf{Dec}(sk, C_i)$. The restriction is that $\mathcal{A}$ is only allowed to query $\mathsf{Dec}(\cdot)$ on ciphertexts C_i different from the challenge ciphertext C. A key encapsulation mechanism is said to be *indistinguishable against chosen ciphertext attacks* (IND-CCA) if for all PTA adversaries $\mathcal{A}$, the advantage $\mathsf{AdvCCA}^{\mathcal{A}}_{\mathsf{KEM}_{\mathsf{dh1}}}(\kappa)$ is a negligible function in κ.

It was proved in [10] that an IND-CCA secure KEM and a CCA-secure symmetric encryption scheme yields an IND-CCA secure hybrid encryption scheme.

Constrained Chosen-Ciphertext Security. Chosen-ciphertext security can be relaxed to indistinguishability against constrained chosen-ciphertext attacks (IND-CCCA) [16]. Intuitively, one only allows the adversary to make a decapsulation query if it already has some “a priori knowledge” about the decapsulated key. This partial knowledge about the key is modeled implicitly by letting the adversary additionally provide an efficiently computable Boolean predicate $pred : \{0,1\}^n \to \{0,1\}$. If $pred(K) = 1$ then the decapsulated key K is returned, and $\perp$ otherwise. The amount of uncertainty the adversary has about the session key (denoted as *plaintext uncertainty* $uncert_{\mathcal{A}}$) is measured by the fraction of keys for which the predicate evaluates to 1. We require this fraction to be negligible for every query, i.e. the adversary has to have a high a priori knowledge about the decapsulated key when making a decapsulation query. More formally, for an adversary $\mathcal{A}$ we define the advantage function

$$\mathsf{AdvCCCA}^{\mathcal{A}}_{\mathsf{KEM}_{\mathsf{dh2}}}(\kappa) := \Pr\left[b = b' \ : \ \begin{array}{l} (pk, sk) \leftarrow \mathsf{Gen}(1^n) \\ (C, K_0) \leftarrow \mathsf{Enc}(pk) \\ K_1 \leftarrow \{0,1\}^n;\ b \leftarrow \{0,1\} \\ b' \leftarrow \mathcal{A}^{\mathsf{CDec}(\cdot,\cdot)}(pk, K_b, C) \end{array} \right] - \frac{1}{2},$$

where oracle $\mathsf{CDec}(pred_i, C_i)$ first computes $K_i \leftarrow \mathsf{Dec}(sk, C_i)$. If $K_i = \perp$ or $pred_i(K_i) = 0$ then return $\perp$. Otherwise, return K_i. The restriction is that $\mathcal{A}$ is only allowed to query $\mathsf{CDec}(pred_i, C_i)$ on predicates $pred_i$ that are provided as PTA and on ciphertexts C_i different from the challenge ciphertext C.

To adversary $\mathcal{A}$ in the above experiment we also associate $\mathcal{A}$'s plaintext uncertainty $uncert_{\mathcal{A}}(\kappa)$ when making Q decapsulation queries, measured by

$$uncert_{\mathcal{A}}(\kappa) := \frac{1}{Q} \sum_{1 \le i \le Q} \Pr_{K \in \{0,1\}^n} [pred_i(K) = 1] \ ,$$

where $pred_i : \mathbb{G} \to \{0,1\}$ is the predicate $\mathcal{A}$ submits in the ith decapsulation query. Finally, a key encapsulation mechanism is said to be *indistinguishable against constrained chosen ciphertext attacks* (IND-CCCA) if for all PTA adversaries $\mathcal{A}$ with negligible $uncert_{\mathcal{A}}(\kappa)$, the advantage $\mathsf{AdvCCCA}^{\mathcal{A}}_{\mathsf{KEM}_{\mathsf{dh2}}}(n)$ is a negligible function in κ.

It was proved in [16] that an IND-CCCA secure KEM plus a symmetric encryption scheme secure in the sense of authenticated encryption yields an IND-CCA secure hybrid encryption scheme.

We refer to the full version [15, Appendix A] for other definitions of standard cryptographic primitives such as hash functions and pseudorandom generators.

2.3 Diffie-Hellman Assumptions

Let $\mathbb{G} = \mathbb{G}_\kappa$ be a cyclic group generated by g. Define

$$\mathrm{dh}(A, B) := C, \quad \text{where } A = g^a,\ B = g^b, \text{ and } C = g^{ab}. \tag{5}$$

The problem of computing $\mathrm{dh}(A, B)$ given random $A, B \in \mathbb{G}$ is the *computational Diffie-Hellman (DH) problem*. The *DH assumption* asserts that this problem is hard, that is, $\Pr[\mathcal{A}(A, B) = \mathrm{dh}(A, B)] \le \mathrm{negl}(\kappa)$ for all probabilistic polynomial-time algorithms $\mathcal{A}$. The *DH predicate* is defined as

$$\mathrm{dhp}(A, \hat{B}, \hat{C}) := \quad \mathrm{dh}(A, \hat{B}) \stackrel{?}{=} \hat{C}.$$

The Strong DH assumption states that it is hard to compute $\mathrm{dh}(A, B)$, given random $A, B \in \mathbb{G}$, along with access to a *decision oracle* for the predicate $\mathrm{dhp}(A, \cdot, \cdot)$, which on input $(\hat{B}, \hat{C})$, returns $\mathrm{dhp}(A, \hat{B}, \hat{C})$.

Let dh be defined as in (5). Define the function

$$\begin{aligned} 2\mathrm{dh} : \quad & \mathbb{G}^3 \to \mathbb{G}^2 \\ & (A_1, A_2, B) \mapsto (\mathrm{dh}(A_1, B), \mathrm{dh}(A_2, B)). \end{aligned}$$

This function, introduced in [8], is called the *twin DH function*. One can also define a corresponding *twin DH predicate*:

$$\mathrm{2dhp}(A_1, A_2, \hat{B}, \hat{C}_1, \hat{C}_2) := \quad \mathrm{2dh}(A_1, A_2, \hat{B}) \overset{?}{=} (\hat{C}_1, \hat{C}_2).$$

The *twin Diffie-Hellman assumption* states it is hard to compute $\mathrm{2dh}(A_1, A_2, B)$, given random $A_1, A_2, B \in \mathbb{G}$. The *strong twin DH assumption* states that it is hard to compute $\mathrm{2dh}(A_1, A_2, B)$, given random $A_1, A_2, B \in \mathbb{G}$, along with access to a *decision oracle* for the predicate $\mathrm{2dhp}(A_1, A_2, \cdot, \cdot, \cdot)$, which on input $(\hat{B}, \hat{C}_1, \hat{C}_2)$, returns $\mathrm{2dhp}(A_1, A_2, \hat{B}, \hat{C}_1, \hat{C}_2)$. It is clear that the (strong) twin DH assumption implies the DH assumption.

We will make use of a result from [8], which essentially states that the DH assumption implies the *strong twin Diffie-Hellman assumption*.

Lemma 1 (Theorem 3 of [8]). *Let $\mathbb{G}$ be a group of prime order p, $\log_2 p = \mathrm{poly}(\kappa)$. Suppose $\mathcal{A}$ is an adversary against the strong twin Diffie-Hellman problem in $\mathbb{G}$, running in polynomial-time in κ and having non-negligible success probability. Then there exists a polynomial-time adversary $\mathcal{B}$ against the computational Diffie-Hellman problem in $\mathbb{G}$ having non-negligible success probability.*

2.4 Hard-Core Functions

In the following we denote with $f_{\mathrm{gl}} : \mathbb{G} \times \{0,1\}^u \to \{0,1\}^\nu$ a Goldreich-Levin hard-core function [13] for $\mathrm{dh}(A, B)$ with randomness space $\{0,1\}^u$ and range $\{0,1\}^\nu$, where u and ν are suitable integers (depending on the given group representation).

The following lemma is from [8, Theorem 9].

Lemma 2. *Let $\mathbb{G} = \mathbb{G}_\kappa$ be a prime-order group generated by g. Let $A_1, A_2, B \overset{\$}{\leftarrow} \mathbb{G}$ be random group elements, $R \overset{\$}{\leftarrow} \{0,1\}^u$, and let $K = f_{\mathrm{gl}}(\mathrm{dh}(A_1, B), R)$. Let $U_\nu \overset{\$}{\leftarrow} \{0,1\}^\nu$ be uniformly random. Suppose there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ having access to an oracle computing $\mathrm{2dhp}(A_1, A_2, \cdot, \cdot, \cdot)$ and distinguishing the distributions*

$$\Delta_{\mathsf{dh}} = (g, A_1, A_2, B, K, R) \quad \text{and} \quad \Delta_{\mathsf{rand}} = (g, A_1, A_2, B, U_\nu, R)$$

with non-negligible advantage. Then there exists a probabilistic polynomial-time algorithm computing $\mathrm{dh}(A, B)$ on input (A, B) with non-negligible success probability.

3 Chosen-Ciphertext Secure Key Encapsulation

In this section we build our first CCA-secure key-encapsulation mechanism whose security is based on the DH assumption.

Let $\mathbb{G} = \mathbb{G}_\kappa$ be a group of prime order p and let $n = n(\kappa)$ be a polynomial. Let $\mathsf{T}_s : \mathbb{G} \to \mathbb{Z}_p$ be a hash function with key s that is assumed to be target collision resistant (see [15, Appendix A] for a formal definition). Let $\mathsf{KEM}_{\mathsf{dh1}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be defined as follows.

$\mathsf{Gen}(1^\kappa)$ Choose a random generator $g \xleftarrow{\$} \mathbb{G}$ and randomness $R \xleftarrow{\$} \{0,1\}^u$ for f_{gl}. Choose a random seed s for the hash function T_s, choose random integers $x, x', y, y', z_1, \ldots, z_n \xleftarrow{\$} \mathbb{Z}_p$, and set $X = g^x$, $X' = g^{x'}$, $Y = g^y$, $Y' = g^{y'}$, $Z_1 = g^{z_1}, \ldots, Z_n = g^{z_n}$. Set

$$pk = (g, X, X', Y, Y', Z_1, \ldots, Z_n, R, s) \text{ and } sk = (pk, x, x', y, y', z_1, \ldots, z_n)$$

and return (pk, sk).

$\mathsf{Enc}(pk)$ On input of public key pk, sample $r \xleftarrow{\$} \mathbb{Z}_p$. Set $C_0 = g^r$, $t = \mathsf{T}_s(C_0)$, $C_1 = (X^t X')^r$, $C_2 = (Y^t Y')^r$, and

$$K = (f_{\mathrm{gl}}(Z_1^r, R), \ldots, f_{\mathrm{gl}}(Z_n^r, R))$$

Return $((C_0, C_1, C_2), K)$.

$\mathsf{Dec}(sk, (C_0, C_1, C_2))$ Set $t = \mathsf{T}_s(C_0)$. If $C_1 \neq C_0^{xt+x'}$ or $C_2 \neq C_0^{yt+y'}$ then return $\perp$. Otherwise compute and return

$$K = (f_{\mathrm{gl}}(C_0^{z_1}, R), \ldots, f_{\mathrm{gl}}(C_0^{z_n}, R)).$$

Theorem 1. *Let T_s be a target collision-resistant hash function and suppose that the computational Diffie-Hellman assumption holds in $\mathbb{G}$. Then $\mathsf{KEM}_{\mathsf{dh1}}$ is* IND-CCA *secure.*

In the proof we use a trick from [4] to set up the public key and challenge ciphertext in a way to perform an all-but-one simulation. This enables the simulator to embed the given Diffie-Hellman challenge, while at the same time being able to decapsulate any ciphertext submitted by the adversary. We combine this technique with the twinning technique from [8], to be able to check for consistency of submitted ciphertexts.

PROOF. In the following we write (C_0^*, C_1^*, C_2^*) to denote the challenge ciphertext with corresponding key K_0^*, denote with K_1^* the random key chosen by the IND-CCA experiment, and set $t^* = \mathsf{T}_s(C_0^*)$.

We proceed in a sequence of games. We start with a game where the challenger proceeds like the standard IND-CCA game (i.e., K_0^* is a real key and K_1^* is a random key), and end up with a game where both K_0^* and K_1^* are chosen uniformly random. Then we show that all games are computationally indistinguishable under the computational Diffie-Hellman assumption. Let W_i denote the event that $\mathcal{A}$ outputs b' such that $b' = b$ in Game i.

Game 0. This is the standard IND-CCA game. By definition we have

$$\Pr[W_0] = \frac{1}{2} + \mathsf{AdvCCA}^{\mathcal{A}}_{\mathsf{KEM}_{\mathsf{dh1}}}(\kappa)$$

Game 1. We proceed as in Game 0, except that the challenger returns $\perp$ if the adversary queries to decapsulate a ciphertext (C_0', C_1', C_2') with $C_0' = C_0^*$. Note that the probability that the adversary submits a ciphertext such that $C_0' = C_0^*$ *before* seeing the challenge ciphertext is bounded by q/p, where q is the number of chosen-ciphertext queries issued by $\mathcal{A}$. Since $q = \text{poly}(\kappa)$, we have $q/p \leq \text{negl}(\kappa)$. Moreover, a ciphertext is inconsistent, thus gets rejected, if $C_0' = C_0^*$ and $C_1' \neq C_1^*$ or $C_2' \neq C_2^*$, and is rejected by definition if $C_1' = C_1^*$ and $C_2' = C_2^*$. Therefore

$$|\Pr[W_1] - \Pr[W_0]| \leq \text{negl}(\kappa).$$

Game 2. We define Game 2 like Game 1, except for the following. Now the challenger aborts, if the adversary asks to decapsulate a ciphertext (C_0', C_1', C_2') with $C_0' \neq C_0^*$ and $\mathsf{T}_s(C_0') = \mathsf{T}_s(C_0^*)$. By the target collision resistance of T_s, we have

$$|\Pr[W_2] - \Pr[W_1]| \leq \text{negl}(\kappa).$$

Game 3. We define Game 3 like Game 2, except that we sample $K_0^* \xleftarrow{\$} \{0,1\}^{n\nu}$ uniformly random. Note that now both K_0^* and K_1^* are chosen uniformly random, thus we have

$$\Pr[W_3] = \frac{1}{2}.$$

We claim that

$$|\Pr[W_3] - \Pr[W_2]| \leq \text{negl}(\kappa)$$

under the computational Diffie-Hellman assumption. We prove this by a hybrid argument. To this end, we define a sequence of hybrid games $H_0, \ldots, H_n$, such that H_0 equals Game 2 and H_n equals Game 3. Then we argue that hybrid H_i is indistinguishable from hybrid H_{i-1} for $i \in \{1, \ldots, n\}$ under the computational Diffie-Hellman assumption. The claim follows, since $n = n(\kappa)$ is a polynomial. We define H_0 exactly like Game 2. Then, for i from 1 to n, in hybrid H_i we set the first $i\nu$ bits of K_0^* to independent random bits, and proceed otherwise exactly like in hybrid H_{i-1}. Thus, hybrid H_n proceeds exactly like Game 3.

Let E_i denote the event that $\mathcal{A}$ outputs 1 in Hybrid i. Suppose

$$|\Pr[E_0] - \Pr[E_n]| = 1/\text{poly}_0(\kappa), \tag{6}$$

that is, the success probability of $\mathcal{A}$ in Hybrid 0 is not negligibly close to the success probability in Hybrid n. Note that then there must exist an index i such that $|\Pr[E_{i-1}] - \Pr[E_i]| = 1/\text{poly}(\kappa)$ (since if $|\Pr[E_{i-1}] - \Pr[E_i]| \leq \text{negl}(\kappa)$ for all i, then we would have $|\Pr[E_0] - \Pr[E_n]| \leq \text{negl}(\kappa)$).

Suppose there exists an algorithm $\mathcal{A}$ for which (6) holds. Then we can construct an adversary $\mathcal{B}$ having access to a 2dhp oracle and distinguishing the distributions Δ_{dh} and Δ_{rand}, which by Lemma 2 is sufficient to prove security under the computational Diffie-Hellman assumption in $\mathbb{G}$. Adversary $\mathcal{B}$ receives a challenge $\delta = (g, A_1, A_2, B, L, R)$ as input, and has access to an oracle evaluating $\text{2dhp}(A_1, A_2, \cdot, \cdot, \cdot)$. $\mathcal{B}$ guesses an index $i \in [n]$, which with probability at least $1/n$ corresponds to the index i such that $|\Pr[E_{i-1}] - \Pr[E_i]| = \max_i |\Pr[E_{i-1}] - \Pr[E_i]|$, and proceeds as follows.

Set-up of the public key. $\mathcal{B}$ picks random integers $d, e, f \xleftarrow{\$} \mathbb{Z}_p$, and sets $X = A_1^e$, $X' = A_1^{-et^*} g^d$, $Y = A_2$, $Y' = A_2^{-t^*} g^f$, and $Z_i = A_1$, where $t^* = \mathsf{T}_s(B)$. R is used as randomness for $f_{\text{gl}}(\cdot, R)$, the rest of the public key is generated as in Game 0. Note that X, X', Y, Y', Z_i are independent and uniformly distributed group elements.

Handling decapsulation queries. When $\mathcal{A}$ issues a decapsulation query $(C_0 = g^r, C_1, C_2)$, $\mathcal{B}$ computes $t = \mathsf{T}_s(C_0)$, $\tilde{X} = (C_1/C_0^d)^{1/(et-et^*)}$, and $\tilde{Y} = (C_2/C_0^f)^{1/(t-t^*)}$. Assuming $t \neq t^*$ and that the ciphertext is formed correctly (that is, $C_0 = g^r$, $C_1 = (X^t X')^r$, and $C_2 = (Y^t Y')^r$) we have

$$\begin{aligned}\tilde{X} &= ((X^t X')^r/(g^r)^d)^{1/(et-et^*)} = (A_1^{er(t-t^*)} g^{rd}/g^{rd})^{1/(et-et^*)} \\ &= A_1^r = \text{dh}(A_1, C_0),\end{aligned}$$

and likewise $\tilde{Y} = A_2^r = \text{dh}(A_2, C_0)$. $\mathcal{B}$ tests consistency of ciphertexts by querying $\text{2dhp}(A_1, A_2, C_0, \tilde{X}, \tilde{Y})$, which returns 1 if and only if $\tilde{X} = \text{dh}(A_1, C_0)$ and $\tilde{Y} = \text{dh}(A_2, C_0)$.

If this test is passed, then $\mathcal{B}$ sets $K_0^* = (K_{0,1}^*, \ldots, K_{0,n}^*)$ as $K_{0,i}^* = f_{\text{gl}}(\tilde{X}, R)$ and $K_{0,j}^* = f_{\text{gl}}(C_0^{z_j}, R)$ for $j \in [n] \setminus \{i\}$. Since by Game 2 we have $t \neq t^*$ for all queries issued by $\mathcal{A}$, $\mathcal{B}$ can answer all decapsulation queries correctly.

Set-up of the challenge ciphertext. $\mathcal{B}$ sets $C_0^* = B$, $C_1^* = B^d$, and $C_2^* = B^f$. Note that, by the set-up of X, X', Y, Y', this is a consistent ciphertext, since we have

$$(X^{t^*} X')^{\log_g B} = ((A_1^e)^{t^*} A_1^{-et^*} g^d)^{\log_g B} = B^d$$

and (similarly) $(Y^{t^*} Y')^{\log_g B} = B^f$. Then $\mathcal{B}$ samples $i-1$ uniformly random bits $K_1, \ldots, K_{i-1}$, sets $K_i = L$, $K_j = f_{\text{gl}}((C_0^*)^{z_j}, R)$ for j from $i+1$ to n, and outputs the challenge $((C_0^*, C_1^*, C_2^*), (K_1, \ldots, K_n))$.

Now, if $\delta \xleftarrow{\$} \Delta_{\mathsf{dh}}$ then $L = f_{\text{gl}}(\text{dh}(B, Z_i), R)$. Thus $\mathcal{A}$'s view when interacting with $\mathcal{B}$ is identical to Hybrid H_{i-1}. If $\delta \xleftarrow{\$} \Delta_{\mathsf{rand}}$, then $\mathcal{A}$'s view is identical to Hybrid H_i. Thus $\mathcal{B}$ can use $\mathcal{A}$ to distinguish $\delta \in \Delta_{\mathsf{dh}}$ from $\delta \in \Delta_{\mathsf{rand}}$. $\square$

We remark that the same proof strategy can be used to prove that the KEM given in equation (2) (Section 1) is CCA-secure under the Strong DH assumption.

4 Constrained Chosen-Ciphertext Secure Key Encapsulation

In this section we build a more efficient variant of our first CCA-secure key-encapsulation mechanism, which we cannot prove CCA-secure. However, we can prove that it is secure in the sense of constrained CCA security, which is sufficient to obtain CCA-secure hybrid encryption. Again the security is based on the DH assumption.

Let $\mathbb{G} = \mathbb{G}_\kappa$ be a group of prime order p and let $n = n(\kappa)$ be a polynomial. Let $\mathsf{KEM}_{\mathsf{dh2}} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be defined as follows.

$\mathsf{Gen}(1^\kappa)$ Choose a random generator $g \xleftarrow{\$} \mathbb{G}$ and randomness $R \xleftarrow{\$} \{0,1\}^u$ for f_{gl}. Choose a random seed s for the hash function $\mathsf{T}_s : \mathbb{G} \to \mathbb{Z}_p$, choose random integers $x, x', y, y', z_1, \ldots, z_n \xleftarrow{\$} \mathbb{Z}_p$, and set $X = g^x$, $X' = g^{x'}$, $Y = g^y$, $Y' = g^{y'}$, $Z_1 = g^{z_1}, \ldots, Z_n = g^{z_n}$. Let $\mathsf{G} : \mathbb{G} \to \{0,1\}^n$ be a pseudorandom generator. Set

$$pk = (g, X, X', Y, Y', Z_1, \ldots, Z_n, R, s, \mathsf{G}) \text{ and } sk = (pk, x, x', y, y', z_1, \ldots, z_n)$$

and return (pk, sk).

$\mathsf{Enc}(pk)$ On input of public key pk, sample $r \xleftarrow{\$} \mathbb{Z}_p$. Set $C_0 = g^r$, $t = \mathsf{T}_s(C_0)$, $C_1 = (X^t X')^r$, $K_{\mathsf{G}} = \mathsf{G}((Y^t Y')^r)$, and

$$K_{\mathsf{dh}} = (f_{\mathrm{gl}}(Z_1^r, R), \ldots, f_{\mathrm{gl}}(Z_n^r, R))$$

Set $K = K_{\mathsf{G}} \oplus K_{\mathsf{dh}}$ and return $((C_0, C_1), K)$.

$\mathsf{Dec}(sk, (C_0, C_1))$ Set $t = \mathsf{T}_s(C_0)$. If $C_1 \neq C_0^{xt+x'}$ then return $\perp$. Otherwise compute $K_{\mathsf{G}} = \mathsf{G}(C_0^{yt+y'})$ and

$$K_{\mathsf{dh}} = (f_{\mathrm{gl}}(C_0^{z_1}, R), \ldots, f_{\mathrm{gl}}(C_0^{z_n}, R)),$$

and return $K = K_{\mathsf{G}} \oplus K_{\mathsf{dh}}$.

Theorem 2. *Let T_s be a target collision-resistant hash function, G be a pseudo-random generator, and suppose that the computational Diffie-Hellman assumption holds in $\mathbb{G}$. Then $\mathsf{KEM}_{\mathsf{dh2}}$ is IND-CCCA secure.*

Since we removed one element from the ciphertext (which was crucial to apply the twinning technique from the proof of Theorem 1 to check for consistency of ciphertexts) we have to use different means to prove the constrained chosen-ciphertext security of $\mathsf{KEM}_{\mathsf{dh2}}$. Here we exploit the new set-up of the encapsulated key, which allows us to reject invalid ciphertexts "implicitly." Due to space restrictions, the proof is deferred to the full version [15].

5 Reducing the Size of the Public Key

Let $(\mathbb{G}, \mathbb{G}_T)$ be a bilinear group that is equiped with an efficiently computable pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. (See, e.g., [6,4].) In this section we show that by instantiating our scheme from Equation (2) (Section 1) in bilinear groups we are able to reduce the size of the public-key considerably.

5.1 Bilinear Diffie-Hellman Assumption

Let

$$\mathrm{bdh}(A,B,C) := D, \quad \text{where } A = g^a,\ B = g^b,\ C = g^c, \text{ and } D = \hat{e}(g,g)^{abc}. \quad (7)$$

The problem of computing $\mathrm{bdh}(A,B,C)$ given random $A,B,C \in \mathbb{G}$ is the *computational Bilinear Diffie-Hellman (DH) problem.* The *BDH assumption* [6] asserts that this problem is hard, that is, $\Pr[\mathcal{A}(A,B,C) = \mathrm{bdh}(A,B,C)] \leq \mathrm{negl}(\kappa)$ for all probabilistic polynomial-time algorithms $\mathcal{A}$.

In the bilinear setting, the Goldreich-Levin theorem [13] gives us the following lemma for a $f_{\mathrm{gl}} : \mathbb{G}_T \times \{0,1\}^u \to \{0,1\}^\nu$.

Lemma 3. *Let $\mathbb{G} = \mathbb{G}_\kappa$ be a prime-order group generated by g equipped with a pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $A,B,C \overset{\$}{\leftarrow} \mathbb{G}$ be random group elements, $R \overset{\$}{\leftarrow} \{0,1\}^u$, and let $K = f_{\mathrm{gl}}(\mathrm{bdh}(A,B,C),R)$. Let $U_\nu \overset{\$}{\leftarrow} \{0,1\}^\nu$ be uniformly random. Suppose there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ distinguishing the distributions*

$$\Delta_{\mathsf{bdh}} = (g,A,B,C,K,R) \quad \text{and} \quad \Delta_{\mathsf{rand}} = (g,A,B,C,U_\nu,R)$$

with non-negligible advantage. Then there exists a probabilistic polynomial-time algorithm computing $\mathrm{bdh}(A,B,C)$ on input (A,B,C) with non-negligible success probability, hence breaking the BDH assumption.

5.2 Public-Key Encryption with Public Keys of Size $\mathcal{O}(1)$

Our first idea is a variant where the elements $sys = (g,X,X',Z_1,\ldots,Z_n) \in \mathbb{G}^{n+3}$ can be put into the system parameters (that can be shared among many users) and the public-key to contain only one single group element Y. Our encryption scheme can be viewed as a BDH-variant of a Decisional BDH scheme from [7,20]. It is defined as follows.

$\mathsf{Gen}(1^\kappa)$ Given the system parameters sys choose a random integer $y \overset{\$}{\leftarrow} \mathbb{Z}_p$, and set $Y = g^y$. Set

$$pk = Y \qquad \text{and} \qquad sk = y$$

and return (pk, sk).

$\mathsf{Enc}(pk)$ On input of public key pk, sample $r \overset{\$}{\leftarrow} \mathbb{Z}_p$. Set $C_0 = g^r$, $t = \mathsf{T}(C_0)$, $C_1 = (X^t X')^r$, and $K = (K_1,\ldots,K_n)$, where

$$K_i = f_{\mathrm{gl}}(\hat{e}(Y^r, Z_i), R), \text{ for } i \in [1,n].$$

Return $((C_0,C_1),K)$.

$\mathsf{Dec}(sk,(C_0,C_1))$ If $\hat{e}(C_0, X^t X') \neq \hat{e}(g, C_1)$ then return $\perp$. Otherwise, compute, for each $i \in [1,n]$,

$$K_i = f_{\mathrm{gl}}(\hat{e}(C_0^y, Z_i), R)$$

and return $K = (K_1,\ldots,K_n) \in \{0,1\}^{n\nu}$.

Note that the consistency of the ciphertext is publicly verifiable, i.e., anyone could verify a ciphertext being consistent or not.

Theorem 3. *Let T be a target collision-resistant hash function and suppose that the computational Bilinear Diffie-Hellman assumption holds in $\mathbb{G}$. Then the above scheme is an* IND-CCA *secure* KEM.

Proof. We proceed in a sequence of games similarly to Theorem 1.

As before, we write (C_0^*, C_1^*) to denote the challenge ciphertext with corresponding key K_0^*, denote with K_1^* the random key chosen by the IND-CCA experiment, and set $t^* = \mathsf{T}_s(C_0^*)$.

We start with a game where the challenger proceeds like the standard IND-CCA game (i.e., K_0^* is a real key and K_1^* is a random key), and end up with a game where both K_0^* and K_1^* are chosen uniformly random. Then we show that all games are computationally indistinguishable under the computational Bilinear Diffie-Hellman assumption. Let W_i denote the event that $\mathcal{A}$ outputs b' such that $b' = b$ in Game i.

Game 0. This is the standard IND-CCA game. By definition we have

$$\Pr[W_0] = \frac{1}{2} + \mathsf{AdvCCA}^{\mathcal{A}}_{\mathsf{KEM}_{\mathsf{bdh1}}}(\kappa)$$

Game 1. We proceed as in Game 0, except that the challenger aborts, if the adversary queries to decapsulate a ciphertext (C_0', C_1') with $C_0' = C_0^*$. Note that the probability that the adversary submits a ciphertext such that $C_0' = C_0^*$ *before* seeing the challenge ciphertext is bounded by q/p, where q is the number of chosen-ciphertext queries issued by $\mathcal{A}$. Since $q = \mathrm{poly}(\kappa)$, we have $q/p \leq \mathrm{negl}(\kappa)$. Moreover, a ciphertext is inconsistent, thus gets rejected, if $C_0' = C_0^*$ and $C_1' \neq C_1^*$, and is rejected by definition if $C_0' = C_0^*$ and $C_1' = C_1^*$. Therefore

$$|\Pr[W_1] - \Pr[W_0]| \leq \mathrm{negl}(\kappa).$$

Game 2. We define Game 2 like Game 1, except for the following. Now the challenger aborts, if the adversary asks to decapsulate a ciphertext (C_0', C_1') with $C_0' \neq C_0^*$ and $\mathsf{T}_s(C_0') = \mathsf{T}_s(C_0^*)$. By the target collision resistance of T_s, we have

$$|\Pr[W_2] - \Pr[W_1]| \leq \mathrm{negl}(\kappa).$$

Game 3. We define Game 3 like Game 2, except that we sample $K_0^* \xleftarrow{\$} \{0,1\}^{n\nu}$ uniformly random. Note that now both K_0^* and K_1^* are chosen uniformly random, thus we have

$$\Pr[W_3] = \frac{1}{2}.$$

We claim that

$$|\Pr[W_3] - \Pr[W_2]| \leq \mathrm{negl}(\kappa)$$

under the computational Bilinear Diffie-Hellman assumption. We prove this by a hybrid argument. To this end, we define a sequence of hybrid games $H_0, \ldots, H_n$, such that H_0 equals Game 2 and H_n equals Game 3. Then we argue that hybrid H_i is indistinguishable from hybrid H_{i-1} for $i \in \{1, \ldots, n\}$ under the computational Bilinear Diffie-Hellman assumption. The claim follows, since $n = n(\kappa)$ is a polynomial. We define H_0 exactly like Game 2. Then, for i from 1 to n, in hybrid H_i we set the first $i\nu$ bits of K_0^* to independent random bits, and proceed otherwise exactly like in hybrid H_{i-1}. Thus, hybrid H_n proceeds exactly like Game 3.

Let E_i denote the event that $\mathcal{A}$ outputs 1 in Hybrid i. Suppose that

$$|\Pr[E_0] - \Pr[E_n]| = 1/\mathrm{poly}_0(\kappa), \tag{8}$$

that is, the success probability of $\mathcal{A}$ in Hybrid 0 is not negligibly close to the success probability in Hybrid n. Note that then there must exist an index i such that $|\Pr[E_{i-1}] - \Pr[E_i]| = 1/\mathrm{poly}(\kappa)$ (since if $|\Pr[E_{i-1}] - \Pr[E_i]| \leq \mathrm{negl}(\kappa)$ for all i, then we would have $|\Pr[E_0] - \Pr[E_n]| \leq \mathrm{negl}(\kappa)$).

Suppose that there exists an algorithm $\mathcal{A}$ for which (8) holds. Then we can construct an adversary $\mathcal{B}$ distinguishing the distributions Δ_{bdh} and Δ_{rand}, which by Lemma 3 is sufficient to prove security under the computational Bilinear Diffie-Hellman assumption in $\mathbb{G}$. Adversary $\mathcal{B}$ receives a challenge $\delta = (g, A, B, C, L, R)$ as input, guesses an index $i \in [n]$, which with probability at least $1/n$ corresponds to the index i such that $|\Pr[E_{i-1}] - \Pr[E_i]| = \max_i |\Pr[E_{i-1}] - \Pr[E_i]|$, and proceeds as follows:

Set-up of the system parameters. $\mathcal{B}$ picks random integers $d, e, f \overset{\$}{\leftarrow} \mathbb{Z}_p$, and sets $X = A^e$, $X' = A^{-et^*} g^d$, and $Z_i = A$, where $t^* = \mathsf{T}(C)$. The rest of the public key is generated as in Game 0. Note that C, X, X', Z_i are independent and uniformly distributed group elements.

Set-up of the public key. $\mathcal{B}$ sets $Y = B$.

Handling decapsulation queries. When $\mathcal{A}$ issues a decapsulation query $(C_0 = g^r, C_1)$, $\mathcal{B}$ computes $t = \mathsf{T}_s(C_0)$ and tests the consistency of the ciphertext by verifying

$$\hat{e}(C_0, X^t X') \overset{?}{=} \hat{e}(g, C_1).$$

If the equality holds, then $\mathcal{B}$ sets $K = (K_1, \ldots, K_n)$ as $K_j = f_{\mathrm{gl}}(\hat{e}(C_0^{z_j}, Y), R)$ for $j \in [n] \setminus \{i\}$ and $K_i = f_{\mathrm{gl}}(\hat{e}(\tilde{X}, Y), R)$, where $\tilde{X} := (C_1/C_0^d)^{1/(et-et^*)}$. Note that

$$\begin{aligned}\tilde{X} &= ((X^t X')^r/(g^r)^d)^{1/(et-et^*)} = (A^{r(et-et^*)} g^{rd}/g^{rd})^{1/(et-et^*)} \\ &= A^r = \mathrm{dh}(A, C_0).\end{aligned}$$

Since by Game 2 we have $t \neq t^*$, $\mathcal{B}$ can answer all decapsulation queries correctly for all queries issued by $\mathcal{A}$.

Set-up of the challenge ciphertext. $\mathcal{B}$ sets $C_0^* = C$ and $C_1^* = C^d$. Note that, by the set-up of X, X', this is a consistent ciphertext, since we have

$$(X^{t^*} X')^{\log_g C} = ((A_1^e)^{t^*} A_1^{-et^*} g^d)^{\log_g C} = C^d$$

Then $\mathcal{B}$ samples $i-1$ uniformly random groups of ν bits $K_1^*, \ldots, K_{i-1}^*$, sets $K_i^* = L$, $K_j^* = f_{\mathrm{gl}}(\hat{e}(C_0^*, Y)^{z_j}, R)$ for j from $i+1$ to n, and outputs the challenge $((C_0^*, C_1^*), (K_1^*, \ldots, K_n^*))$.

Now, if $\delta \overset{\$}{\leftarrow} \Delta_{\mathsf{bdh}}$ then we have $L = f_{\mathrm{gl}}(\mathrm{bdh}(A,B,C), R)$. Thus $\mathcal{A}$'s view when interacting with $\mathcal{B}$ is identical to Hybrid H_{i-1}. If $\delta \overset{\$}{\leftarrow} \Delta_{\mathsf{rand}}$, then $\mathcal{A}$'s view is identical to Hybrid H_i. Thus $\mathcal{B}$ can use $\mathcal{A}$ to distinguish $\delta \in \Delta_{\mathsf{bdh}}$ from $\delta \in \Delta_{\mathsf{rand}}$. □

5.3 Public-Key Encryption with Public-Key of Size $\mathcal{O}(\sqrt{n})$

Our second idea reduces the size of the public-key from $\approx n$ to $\approx 2\sqrt{n}$ group elements (and no systems parameters). Assume n is a square and set $\eta := \sqrt{n}$. The public key contains elements $Z_1, Z_1', \ldots, Z_\eta, Z_\eta' \in \mathbb{G}$ which implicitly define $\eta^2 = n$ distinct elements $Z_{i,j} = \hat{e}(Z_i, Z_j')$ in the target group $\mathbb{G}_T$. In our new scheme these elements can be used in place of $Z_1, \ldots, Z_n$.

$\mathsf{Gen}(1^\kappa)$ Choose a random generator $g \overset{\$}{\leftarrow} \mathbb{G}$ and randomness $R \overset{\$}{\leftarrow} \{0,1\}^u$ for f_{gl}. Choose a random seed s for the hash function T_s, choose random integers $x, x', z_1, z_1', \ldots, z_\eta, z_\eta' \overset{\$}{\leftarrow} \mathbb{Z}_p$, and set $X = g^x$, $X' = g^{x'}$, $Z_1 = g^{z_1}$, $Z_1' = g^{z_1'}$, ... , $Z_\eta = g^{z_\eta}$, $Z_\eta' = g^{z_\eta'}$. Set

$$pk = (g, X, X', Z_1, Z_1', \ldots, Z_\eta, Z_\eta', R, s) \text{ and } sk = (pk, x, x', z_1, z_1', \ldots, z_\eta, z_\eta')$$

and return (pk, sk).

$\mathsf{Enc}(pk)$ On input of public key pk, sample $r \overset{\$}{\leftarrow} \mathbb{Z}_p$. Set $C_0 = g^r$, $t = \mathsf{T}_s(C_0)$, $C_1 = (X^t X')^r$, and $K = (K_{1,1}, \ldots, K_{\eta,\eta})$, where

$$K_{i,j} = f_{\mathrm{gl}}(\hat{e}(Z_i^r, Z_j'), R), \text{ for } i,j \in [1,\eta].$$

Return $((C_0, C_1), K)$.

$\mathsf{Dec}(sk, (C_0, C_1))$ First reject if $\hat{e}(C_0, X^t X') \neq \hat{e}(g, C_1)$. Otherwise, for each $i, j \in [1,\eta]$ compute

$$K_{i,j} = f_{\mathrm{gl}}(\hat{e}(C_0^{z_i}, Z_j'), R).$$

and return $K = (K_{1,1}, \ldots, K_{\eta,\eta}) \in \{0,1\}^{n\nu}$.

Like in the previous scheme, the consistency of the ciphertext is publicly verifiable. Furthermore, decryption can alternatively check consistency of the ciphertext by testing if $C_0^{xt+x'} = C_1$.

Theorem 4. *Let T_s be a target collision-resistant hash function and suppose that the computational Bilinear Diffie-Hellman assumption holds in $\mathbb{G}$. Then the above scheme is an* IND-CCA *secure* KEM.

Proof. The proofs goes analogously to that of Theorem 3 with Game 3 defining hybrid games $H_{1,0}$, $H_{1,1}$, $H_{1,2}$, ..., $H_{1,\eta}$, $H_{2,1}$, $H_{2,2}$, ..., $H_{2,\eta}$, $H_{3,1}$, ..., $H_{\eta,\eta}$ (for convenience, we denote with $H_{i,j}^-$ the game preceding $H_{i,j}$ in this ordering, e.g. $H^-_{3,1} = H_{2,\eta}$). Assuming that each two consecutive hybrid games are indistinguishable by $\mathcal{A}$, Game 2 (which is the same as $H_{1,0}$) is indistinguishable from $H_{\eta,\eta}$ (which is the same as Game 3). But when both K_0^* and K_1^* are chosen uniformly random then we have

$$\Pr[W_3] = \frac{1}{2}.$$

So all we have to show is that indeed the hybrid games are indistinguishable.

Suppose that there exists an algorithm $\mathcal{A}$ for which

$$|\Pr[E_{\eta,\eta}] - \Pr[E_{1,0}]| = 1/\mathrm{poly}_0(\kappa), \tag{9}$$

where $E_{i,j}$ denotes the event that $\mathcal{A}$ outputs 1 in $H_{i,j}$. Then there are $i^*, j^* \in \{1 \ldots \eta\}$ such that $\Pr[E_{i^*,j^*}] - \Pr[E^-_{i^*,j^*}] = 1/\mathrm{poly}(\kappa)$, where $E^-_{i,j}$ denotes the event that $\mathcal{A}$ outputs 1 in $H^-_{i,j}$. (If no such indices exist and the difference is negligible for all (i,j), then $|\Pr[E_{\eta,\eta}] - \Pr[E_{1,0}]| = \mathrm{negl}(\kappa)$.)

Then we can construct an adversary $\mathcal{B}$ distinguishing the distributions Δ_{bdh} and Δ_{rand}, which by Lemma 3 is sufficient to prove security under the computational Bilinear Diffie-Hellman assumption in $\mathbb{G}$. Adversary $\mathcal{B}$ receives a challenge $\delta = (g, A, B, C, L, R)$ as input, guesses indices $i, j \in [\eta]$, which with probability at least $1/\eta^2$ correspond to the indices i^*, j^* such that $\left|\Pr[E^-_{i^*,j^*}] - \Pr[E_{i^*,j^*}]\right| = \max_{i,j} \left|\Pr[E^-_{i,j}] - \Pr[E_{i,j}]\right|$, and proceeds as follows:

Set-up of the public-key. $\mathcal{B}$ picks random integers $d, e, f \xleftarrow{\$} \mathbb{Z}_p$, and sets $X = A^e$, $X' = A^{-et^*} g^d$, $Z_{i^*} = A$, and $Z'_{j^*} = B$, where $t^* = \mathsf{T}_s(C)$. The rest of the public key is generated as in scheme definition. Note that $C, X, X', Z_{i^*}, Z'_{j^*}$ are independent and uniformly distributed group elements.

Handling decapsulation queries. When $\mathcal{A}$ issues a decapsulation query $(C_0 = g^r, C_1)$, $\mathcal{B}$ computes $t = \mathsf{T}_s(C_0)$ and tests the consistency of the ciphertext by verifying

$$\hat{e}(C_0, X^t X') \stackrel{?}{=} \hat{e}(g, C_1).$$

If the equality holds, then $\mathcal{B}$ sets $K = (K_{1,1}, \ldots, K_{\eta,\eta})$ as:
- $K_{i,j} = f_{\mathrm{gl}}(\hat{e}(C_0, Z'_j)^{z_i}, R)$ for $i \in [\eta] \setminus \{i^*\}$ and $j \in [\eta]$,
- $K_{i^*,j} = f_{\mathrm{gl}}(\hat{e}(C_0, Z_{i^*})^{z'_j}, R)$ for $j \in [\eta] \setminus \{j^*\}$, and
- $K_{i^*,j^*} = f_{\mathrm{gl}}(\hat{e}(\tilde{X}, B), R)$, where $\tilde{X} := (C_1/C_0^d)^{1/(et-et^*)}$.

Note that

$$\begin{aligned}\tilde{X} &= ((X^t X')^r/(g^r)^d)^{1/(et-et^*)} = (A^{r(et-et^*)} g^{rd}/g^{rd})^{1/(et-et^*)} \\ &= A^r = \mathrm{dh}(A, C_0).\end{aligned}$$

Since by Game 2 we have $t \neq t^*$, $\mathcal{B}$ can answer all decapsulation queries correctly for all queries issued by $\mathcal{A}$.

Set-up of the challenge ciphertext. $\mathcal{B}$ sets $C_0^* = C$ and $C_1^* = C^d$. Note that, by the set-up of X, X', this is a consistent ciphertext, since we have

$$(X^{t^*} X')^{\log_g C} = ((A_1^e)^{t^*} A_1^{-et^*} g^d)^{\log_g C} = C^d$$

Then $\mathcal{B}$ sets the key $K^* = (K_{1,1}^*, K_{1,2}^*, \ldots, K_{i^*,j^*}^*, \ldots, K_{\eta,\eta}^*)$ accordingly:

– the bits before K_{i^*,j^*}^* uniformly at random;
– $K_{i^*,j^*}^* = L$;
– and $K_{i,j}^* = f_{\mathrm{gl}}(\mathrm{bdh}(C, Z_i, Z_j'), R)$ for the remaining ν-bit blocks $K_{i,j}^*$, i.e. $i > i^*$ or $(i = i^* \wedge j > j^*)$, which is possible because $\mathcal{B}$ knows z_i or z_j';

and outputs the challenge $((C_0^*, C_1^*), K^*)$.

Now, if $\delta \xleftarrow{\$} \Delta_{\mathsf{bdh}}$ then we have $L = f_{\mathrm{gl}}(\mathrm{bdh}(A, B, C), R)$. Thus $\mathcal{A}$'s view when interacting with $\mathcal{B}$ is identical to Hybrid H_{i^*,j^*}^-. If $\delta \xleftarrow{\$} \Delta_{\mathsf{rand}}$, then $\mathcal{A}$'s view is identical to Hybrid $H_{i,j}$. Thus $\mathcal{B}$ can use $\mathcal{A}$ to distinguish $\delta \in \Delta_{\mathsf{bdh}}$ from $\delta \in \Delta_{\mathsf{rand}}$. $\square$

We remark that the above construction also extends to a Boneh-Boyen-style [4] identity-based encryption scheme selective-identity secure under the computational Bilinear Diffie-Hellman assumption. The IBE scheme has the same parameters as the above scheme, a user secret key for an identity id contains $2n$ group elements of the form $(g^{z_i z_j'} \cdot (X^{id} X')^{s_{i,j}}, g^{s_{i,j}}) \in \mathbb{G}^2$.

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. 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, New York (November 1993)
3. Boneh, D.: The decision Diffie-Hellman problem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg (1998)
4. 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)
5. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. SIAM Journal on Computing 36(5), 915–942 (2006)
6. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
7. Boyen, X., Mei, Q., Waters, B.: Direct chosen ciphertext security from identity-based techniques. In: ACM CCS 2005, pp. 320–329. ACM Press, New York (November 2005)
8. Cash, D., Kiltz, E., Shoup, V.: The twin Diffie-Hellman problem and applications. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 127–145. Springer, Heidelberg (2008)

9. 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)
10. 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)
11. Freeman, D.M., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: More constructions of lossy and correlation-secure trapdoor functions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 282–298. Springer, Heidelberg (2010)
12. Goldreich, O.: Foundations of Cryptography: Basic Applications, vol. 2. Cambridge University Press, Cambridge (2004)
13. Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: 21st ACM STOC, pp. 25–32. ACM Press, New York (May 1989)
14. 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)
15. Haralambiev, K., Jager, T., Kiltz, E., Shoup, V.: Simple and efficient public-key encryption from Computational Diffie-Hellman in the standard model. Cryptology ePrint Archive, Report 2010/033 (2010), `http://eprint.iacr.org/`
16. 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)
17. 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)
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. Joux, A.: A one round protocol for tripartite Diffie-Hellman. Journal of Cryptology 17(4), 263–276 (2004)
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. 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)
23. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: 40th ACM STOC, pp. 187–196. ACM Press, New York (2008)
24. Waters, B.R.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

Constant Size Ciphertexts in Threshold Attribute-Based Encryption

Javier Herranz[1], Fabien Laguillaumie[2], and Carla Ràfols[1]

[1] Dept. Matemàtica Aplicada IV, Universitat Politècnica de Catalunya, C. Jordi Girona 1-3, Mòdul C3, 08034, Barcelona, Spain
{jherranz,crafols}@ma4.upc.edu

[2] GREYC - Université de Caen Basse-Normandie, Boulevard du Maréchal Juin, BP 5186, 14032 Caen Cedex, France
fabien.laguillaumie@unicaen.fr

Abstract. Attribute-based cryptography has emerged in the last years as a promising primitive for digital security. For instance, it provides good solutions to the problem of anonymous access control. In a ciphertext-policy attribute-based encryption scheme, the secret keys of the users depend on their attributes. When encrypting a message, the sender chooses which subset of attributes must be held by a receiver in order to be able to decrypt.

All current attribute-based encryption schemes that admit reasonably expressive decryption policies produce ciphertexts whose size depends at least linearly on the number of attributes involved in the policy. In this paper we propose the first scheme whose ciphertexts have constant size. Our scheme works for the threshold case: users authorized to decrypt are those who hold at least t attributes among a certain universe of attributes, for some threshold t chosen by the sender. An extension to the case of weighted threshold decryption policies is possible. The security of the scheme against selective chosen plaintext attacks can be proven in the standard model by reduction to the augmented multi-sequence of exponents decisional Diffie-Hellman ($\mathsf{aMSE\text{-}DDH}$) problem.

Keywords: attribute-based encryption, provable security, pairings.

1 Introduction

Encryption is the cryptographic primitive which provides confidentiality to digital communications. In a traditional public key encryption scheme, a message is encrypted with the public key of the intended receiver, who is the only person able to decrypt. This level of confidentiality is enough for many real-life applications, including e-mail and key escrow. However, new situations requiring different cryptographic functionalities appear constantly.

Let us consider for example the case of *anonymous access control*: a system must be accessible only to those who have received the appropriate rights, which are defined by the system administrator. Let us imagine how such a process could be implemented with a standard public key encryption scheme. First, a user A claims that he is actually user A. Second, the system sends to this user a challenge: a ciphertext computed with the public key of A (obtained from a certification authority, maybe), for some random plaintext. Third, A decrypts and sends back the plaintext. Fourth, if the plaintext is correct, the system checks if user A must have access to the system, and if so, A is accepted. This solution has some weaknesses, the main one being the lack of anonymity, as user A must reveal his identity to the system. Furthermore, each time the system wants to change its access control policy, it has to update the database containing all the users that have the right to access the system.

A more desirable solution, employing encryption, would be as follows. First, in a (possibly interactive, physical) registration process, every potential user receives a secret key that depends on his age, his job, his company, his expertise, etc., in short, on his *attributes*. Later, the system defines his policy for access control as a (monotonic) family of subsets of attributes: attributes in one of such subsets must be held by a user in order to have the right to access the system; in particular, in an extreme case, this policy can contain a unique subset with the unique attribute 'right to access system X'. When a user tries to access the system, he receives as a challenge a ciphertext computed by the system, on a random message, using the current access policy. If the policy changes, the system administrator just has to take into account the new policy for generating the future challenges. A user is able to decrypt the challenge only if his attributes satisfy the considered policy. In this way, if a user answers such a challenge correctly, he does not leak who he is, only the fact that his attributes satisfy the access control policy.

Ciphertext-policy *attribute-based encryption* (ABE for short, from now on) is the cryptographic primitive which precisely realizes the functionality described in the previous paragraph. This primitive can be traced back to identity-based encryption [Sha84] (which can be seen as the particular case of ABE where the policy contains a single subset with a single attribute) and to fuzzy identity-based encryption [SW05] (the particular case of ABE where the policy is always defined by a predetermined threshold t: only users holding at least t attributes can decrypt).

Related work. The first paper dealing explicitly with ABE was [GPSW06]. Two different and complementary notions of ABE were defined there: key-policy ABE, where a ciphertext is associated to a list of attributes, and a secret key is associated to a policy for decryption; and ciphertext-policy ABE, where secret keys are associated to a list of attributes (i.e. credentials of that user) and ciphertexts are associated to policies for decryption. It seems that ciphertext-policy ABE can be more useful for practical applications than key-policy ABE. Another related notion is that of fuzzy identity-based encryption [SW05], which can be seen as a particular case of both key-policy and ciphertext-policy ABE.

A construction of a key-policy ABE scheme was provided in [GPSW06], while the first ciphertext-policy ABE scheme was proposed in [BSW07], but its security was proved in the generic group model. Later, a generic construction to transform a key-policy ABE scheme into a ciphertext-policy ABE scheme was given in

[GJPS08], with the drawback that the size of the ciphertexts is $\mathcal{O}(s^3)$, if s is the number of attributes involved in the decryption policy.

The most efficient ciphertext-policy ABE schemes in terms of ciphertext size can be found in [Wat08, DHMR08], the size of a ciphertext depending linearly on the number of attributes involved in the specific policy for that ciphertext. For example, in the case of (t, s)-threshold decryption policies, where there are s involved attributes and a user can decrypt only if he holds t or more attributes, the size of the ciphertexts in one of the schemes in [Wat08] is $s + \mathcal{O}(1)$, whereas the size of the ciphertexts in the scheme in [DHMR08] is $2(s - t) + \mathcal{O}(1)$. Both schemes admit however general policies (general monotonic access structures) and make use of secret sharing techniques.

All the constructions mentioned so far only achieve security under selective attacks, a model in which the attacker specifies the challenge access structure before the setup phase. The first CP-ABE scheme with full security has appeared very recently [LO+10]. The size of the ciphertexts in this scheme is $2s + \mathcal{O}(1)$.

A concept which is more generic than attribute-based encryption is that of predicate encryption [KSW08]: the decryption policy, chosen by the sender of the message, is hidden in the ciphertext, in such a way that even the receiver gets no information on this policy, other than the fact that his attributes satisfy it or not. Because of this additional strong privacy requirement, current proposals for predicate encryption consider quite simple (not very expressive) policies.

We stress that all the existing proposals for ABE schemes produce ciphertexts whose size depends (at least) linearly on the number of attributes involved in the policy for that ciphertext. An exception is the scheme in [EM+09], where ciphertexts have constant size; but this scheme admits only (s, s)-threshold decryption policies. Note that for this particular threshold case where $t = s$, the scheme in [DHMR08] already achieved constant-size ciphertexts. For more expressive or general decryption policies, no existing scheme has short ciphertexts. This fact can limit the applications of ABE in real life, if we consider for example the case of anonymous access control, with a low bandwidth available for the communication between the user and the system administrator.

An essential feature of ABE schemes is their collusion resistance property, which guarantees that a ciphertext can leak no information about the plaintext to users whose attributes do not satisfy the considered policy, even if the union of the attributes of these colluding users satisfies the policy. This property is essential to guarantee a reasonable level of security in many of the applications of ABE schemes, like anonymous access control or access to encrypted data.

A notion similar to ciphertext-policy ABE but without this collusion resistance property has been considered under different names: policy-based encryption [BM05], cryptographic work flow [AMS06], etc. This notion is actually equivalent to the primitive of dynamic distributed identity-based encryption [CCZ06, DHMR07, DP08, DHMR08]: the sender chooses ad-hoc a set of identities and a monotonic access structure defined on this set; the ciphertext can be decrypted only if users associated to the identities of some subset in the access structure cooperate.

Our contribution. In this paper we propose the first collusion-resistant ABE scheme which produces constant size ciphertexts and which admits reasonably expressive decryption policies. Our scheme is inspired by the dynamic threshold (identity-based) encryption scheme from [DP08], in which the ciphertext's size was constant as well. As we have just said, this scheme directly leads to a weak ABE scheme, without the collusion resistance property. The challenge was to modify this scheme in order to achieve collusion resistance without losing the other security and efficiency properties, in particular that of constant size ciphertexts. The resulting scheme works for threshold policies: the sender chooses ad-hoc a set S of attributes and a threshold t, and only users who hold at least t of the attributes in S can decrypt. An extension is possible in order to support also weighted threshold policies.

Our new scheme achieves security against selective chosen plaintext attacks (sCPA), in the standard model, under the assumption that the augmented multi-sequence of exponents decisional Diffie-Hellman (aMSE-DDH) problem is hard to solve. This is essentially the same level of security that was proved for the scheme in [DP08]. Using well-known techniques, it is possible to obtain security against chosen ciphertext attacks (CCA), in the random oracle model.

Organization of the paper. We define the syntactics of attribute-based encryption and the required security properties in Section 2, where we also describe the aMSE-DDH problem, on which the security of our scheme will be based. Section 3 contains the description of our scheme, the details on its correctness and consistency checking, and finally the formal proof of its security. In Section 4 we discuss how to extend our threshold scheme to the case of weighted threshold decryption policies, and the (im)possibility to achieve CCA security from CPA security in the standard model using a generic conversion due to [Wat08]. The work is concluded in Section 5.

2 Preliminaries

In this section we describe the algorithms that form an attribute-based encryption scheme which supports threshold decryption policies, as well as the basic security requirements for such schemes. We also introduce the computational problem called aMSE-DDH problem, to which we will relate the security of our scheme.

2.1 Attribute-Based Encryption

In a ciphertext-policy attribute-based encryption (ABE, for short) system, each user receives from a master entity a secret key which depends on the attributes that he satisfies (to soften the natural limitation of the *unique* trusted authority, the possibility to distribute the key extraction among several authorities has been investigated in [Cha07]). A sender can encrypt a message so that it can be decrypted only by users whose attributes satisfy some policy of his choice, and which may depend of the message. Since the basic scheme that we propose in Section 3 works for *threshold* decryption policies, we describe the protocols and security model with respect to these threshold policies: the sender chooses a subset S of attributes and a threshold t such that $1 \leq t \leq |S|$, and encrypts a message m for the pair (S, t). A particular user will be able to decrypt the ciphertext only if he holds t or more attributes in S. The protocols and security model for ABE schemes supporting more general decryption policies can be described in a very similar way.

Syntactic Definition. A *ciphertext-policy attribute-based encryption scheme* $\mathsf{ABE} = (\mathsf{Setup}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ supporting threshold decryption policies consists of four probabilistic polynomial-time algorithms:

– The randomized *setup* algorithm Setup takes a security parameter λ and a universe of attributes $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_m\}$ as inputs and outputs some public parameters params, containing in particular the set $\mathcal{P}$, which will be common to all the users of the system, along with a secret key msk for the master entity. The public parameters will be an input of all the following algorithms. We write $(\mathsf{params}, \mathsf{msk}) \leftarrow \mathsf{ABE.Setup}(1^\lambda, \mathcal{P})$ to denote an execution of this algorithm.

– The *key extraction* algorithm Ext is an interaction between a user and the master entity. The user proves to the master entity that he enjoys a subset $A \subset \mathcal{P}$ of attributes. After verifying that this is actually the case, the master entity uses his master secret key msk to generate a secret key sk_A (which depends on the subset A of attributes), and gives it to the user. We refer to an execution of this protocol as $\mathsf{sk}_A \leftarrow \mathsf{ABE.Ext}(\mathsf{params}, A, \mathsf{msk})$.

– The *encryption* algorithm Enc takes a subset of attributes $S \subset \mathcal{P}$, a threshold t such that $1 \leq t \leq |S|$, and a message M as inputs. The output is a ciphertext C. We denote an execution of the encryption algorithm as $C \leftarrow \mathsf{ABE.Enc}(\mathsf{params}, S, t, M)$.

– The *decryption* algorithm Dec takes a ciphertext C for the pair (S, t) and a secret key sk_A corresponding to some subset A of attributes as inputs. The output is a message $\tilde{M}$. We write $\tilde{M} \leftarrow \mathsf{ABE.Dec}(\mathsf{params}, C, (S, t), \mathsf{sk}_A)$ to refer to an execution of this protocol.

For correctness, it is required that

$$\mathsf{ABE.Dec}(\mathsf{params}, \mathsf{ABE.Enc}(\mathsf{params}, S, t, M), (S, t), \mathsf{sk}_A) \;=\; M,$$

whenever $|A \cap S| \geq t$ and the values $\mathsf{params}, \mathsf{msk}, \mathsf{sk}_A$ have been obtained by properly executing the protocols $\mathsf{ABE.Setup}$ and $\mathsf{ABE.Ext}$.

Security Model for ABE Schemes. Most previous schemes (all but the one in [LO+10]) consider only security under selective chosen plaintext attacks. This is also the security level that will be provably achieved by our scheme. *Indistinguishability under selective chosen plaintext attacks* ($\mathsf{IND\text{-}sCPA}$ security, for short) for an attribute-based encryption scheme ABE supporting threshold decryption policies and for a security parameter $\lambda \in \mathbb{N}$ is defined by considering the following game that an attacker $\mathcal{A}$ plays against a challenger:

1. The challenger specifies a universe of attributes $\mathcal{P}$ of size m and gives it to the attacker $\mathcal{A}$.
2. $\mathcal{A}$ selects a subset $S \subset \mathcal{P}$ of s attributes and a threshold t such that $1 \leq t \leq s$.
3. The challenger runs $(\mathsf{params}, \mathsf{msk}) \leftarrow \mathsf{ABE.Setup}(1^\lambda, \mathcal{P})$ and gives params to $\mathcal{A}$.
4. [Secret key queries:] $\mathcal{A}$ adaptively sends subsets of attributes $B \subset \mathcal{P}$, with the restriction $|B \cap S| < t$, and must receive $\mathsf{sk}_B \leftarrow \mathsf{ABE.Ext}(\mathsf{params}, B, \mathsf{msk})$ as the answer.
5. $\mathcal{A}$ outputs two messages M_0, M_1 of the same length.
6. [Challenge:] The challenger picks a random bit $b^\star \in \{0,1\}$, computes $C^\star \leftarrow \mathsf{ABE.Enc}(\mathsf{params}, S, t, M_{b^\star})$ and gives $C^\star$ to $\mathcal{A}$.
7. Step 4 is repeated.
8. $\mathcal{A}$ outputs a bit b.

The *advantage* of such an adversary $\mathcal{A}$ in breaking the IND-sCPA security of the ABE scheme is defined as

$$\mathsf{Adv}^{\mathsf{IND\text{-}sCPA}}_{\mathcal{A},\mathsf{ABE}}(\lambda) = |2\Pr[b = b^\star] - 1| .$$

An attribute-based encryption scheme ABE is said to be IND-sCPA secure if $\mathsf{Adv}^{\mathsf{IND\text{-}sCPA}}_{\mathcal{A},\mathsf{ABE}}(\lambda)$ is negligible with respect to the security parameter λ, for any polynomial time adversary $\mathcal{A}$.

Note also that collusion resistance follows from the fact that the adversary can make multiple adaptive secret key queries both before and after the challenge phase.

This is not the strongest security notion that one can consider for ABE schemes. On the one hand, the attacker $\mathcal{A}$ can be allowed to make decryption queries, for ciphertexts C' of his choice (corresponding to pairs (S', t')), with the restriction that the challenge ciphertext C^* is never queried for the challenge pair (S, t). On the other hand, $\mathcal{A}$ can be allowed to choose the challenge pair (S, t) not at the beginning of the game, but at the same time when he chooses the two messages M_0, M_1. In this case, we say that $\mathcal{A}$ is a chosen ciphertext attacker, and that his goal is to break the CCA security of the ABE scheme.

2.2 The Augmented Multi-sequence of Exponents Diffie-Hellman Problem

Our scheme uses an admissible bilinear map (or pairing) as an ingredient and its security relies on the hardness of a problem that we call the *augmented multi-sequence of exponents decisional Diffie-Hellman problem*, which is a slight modification of the multi-sequence of exponents decisional Diffie-Hellman problem considered in [DP08]. The generic complexity of these two problems is covered by the analysis in [BBG05], because the problems fit their *general Diffie-Hellman exponent problem* framework.

Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be three groups of the same prime order p (this is called a *bilinear group triple* in the sequel), and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ be a non-degenerate and efficiently computable bilinear map. Let g_0 be a generator of

$\mathbb{G}_1$ and let h_0 be a generator of $\mathbb{G}_2$. In practice, the bilinear map e can be implemented on any pairing-friendly (hyper-)elliptic curve [FST10]; no more assumptions are made on the groups $\mathbb{G}_1$ and $\mathbb{G}_2$, or on the hypothetical existence of an efficient isomorphism from the one to the other.

Let $\tilde{\ell}, \tilde{m}, \tilde{t}$ be three integers. The *$(\tilde{\ell}, \tilde{m}, \tilde{t})$-augmented multi-sequence of exponents decisional Diffie-Hellman problem* ($(\tilde{\ell}, \tilde{m}, \tilde{t})$-aMSE-DDH) related to the group triplet $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ is as follows:

Input: the vector $\overrightarrow{x}_{\tilde{\ell}+\tilde{m}} = (x_1, \ldots, x_{\tilde{\ell}+\tilde{m}})$ whose components are pairwise distinct elements of $(\mathbb{Z}/p\mathbb{Z})^\star$ which define the polynomials

$$f(X) = \prod_{i=1}^{\tilde{\ell}} (X + x_i) \text{ and } g(X) = \prod_{i=\tilde{\ell}+1}^{\tilde{\ell}+\tilde{m}} (X + x_i),$$

the values

$$\begin{cases} g_0, g_0^{\gamma}, \ldots, g_0^{\gamma^{\tilde{\ell}+\tilde{t}-2}}, & g_0^{\kappa \cdot \gamma \cdot f(\gamma)}, \quad (1.1) \\ g_0^{\omega\gamma}, \ldots, g_0^{\omega\gamma^{\tilde{\ell}+\tilde{t}-2}}, & \quad (1.2) \\ g_0^{\alpha}, g_0^{\alpha\gamma}, \ldots, g_0^{\alpha\gamma^{\tilde{\ell}+\tilde{t}}}, & \quad (1.3) \\ h_0, h_0^{\gamma}, \ldots, h_0^{\gamma^{\tilde{m}-2}}, & h_0^{\kappa \cdot g(\gamma)} \quad (1.4) \\ h_0^{\omega}, h_0^{\omega\gamma}, \ldots, h_0^{\omega\gamma^{\tilde{m}-1}}, & \quad (1.5) \\ h_0^{\alpha}, h_0^{\alpha\gamma}, \ldots, h_0^{\alpha\gamma^{2(\tilde{m}-\tilde{t})+3}} & \quad (1.6) \end{cases}$$

where $\kappa, \alpha, \gamma, \omega$ are unknown random elements of $(\mathbb{Z}/p\mathbb{Z})^\star$, and finally an element $T \in \mathbb{G}_T$.

Output: a bit b.

The problem is correctly solved if the output is $b = 1$ when $T = e(g_0, h_0)^{\kappa \cdot f(\gamma)}$ or if the output is $b = 0$ when T is a random value from $\mathbb{G}_T$. In other words, the goal is to distinguish if T is a random value or if it is equal to $e(g_0, h_0)^{\kappa \cdot f(\gamma)}$.

More formally, let us denote by real the event that T is indeed equal to $T = e(g_0, h_0)^{\kappa \cdot f(\gamma)}$, by random the event that T is a random element from $\mathbb{G}_T$ and by $\mathcal{I}(\overrightarrow{x}_{\tilde{\ell}+\tilde{m}}, \kappa, \alpha, \gamma, \omega, T)$ the input of the problem. Then, we define the *advantage* of an algorithm $\mathcal{B}$ in solving the $(\tilde{\ell}, \tilde{m}, \tilde{t})$-aMSE-DDH problem as

$$\begin{aligned} \mathsf{Adv}_{\mathcal{B}}^{(\tilde{\ell},\tilde{m},\tilde{t})-\mathsf{aMSE\text{-}DDH}}(\lambda) = \Big| & \Pr\left[\mathcal{B}(\mathcal{I}(\overrightarrow{x}_{\tilde{\ell}+\tilde{m}}, \kappa, \alpha, \gamma, \omega, T)) = 1 \middle| \mathsf{real}\right] \\ & - \Pr\left[\mathcal{B}(\mathcal{I}(\overrightarrow{x}_{\tilde{\ell}+\tilde{m}}, \kappa, \alpha, \gamma, \omega, T)) = 1 \middle| \mathsf{random}\right] \Big| \end{aligned}$$

where the probability is taken over all random choices and over the random coins of $\mathcal{B}$.

The only difference with the multi-sequence of exponents decisional Diffie-Hellman problem from [DP08] is the presence in the input of two additional lines (1.2) and (1.5). The generic hardness of this problem is a consequence of Theorem A.2 from [BBG05]. It is stated in the next proposition whose proof follows (almost exactly) that of Corollary 3 in [DP08].

Proposition 1. *For any probabilistic algorithm $\mathcal{B}$ making at most q_G queries to the the oracle that computes the group operations (in groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of order p) and the bilinear pairing $e(\cdot,\cdot)$, its advantage in solving the* aMSE-DDH *problem satisfies*

$$\mathsf{Adv}_{\mathcal{B}}^{(\tilde{\ell},\tilde{m},\tilde{t})-\mathsf{aMSE\text{-}DDH}}(\lambda) \leq \frac{(q_G+2s+2)^2 \cdot d}{2p}$$

where $s = 4\tilde{m} + 3\tilde{\ell} + \tilde{t} + 3$ *and* $d = \max\{2(\tilde{\ell}+2), 2(\tilde{m}+2), 4(\tilde{m}-\tilde{t})+10\}$.

3 The New ABE Scheme

This section is dedicated to the presentation of our ciphertext-policy attribute-based encryption scheme.

In the decryption process, we will use the algorithm Aggregate of [DP08]. Given a list of values $\{g^{\frac{r}{\gamma+x_i}}, x_i\}_{1\leq i\leq n}$, where $r, \gamma \in (\mathbb{Z}/p\mathbb{Z})^\star$ are unknown and $x_i \neq x_j$ if $i \neq j$, the algorithm computes the value

$$\mathsf{Aggregate}(\{g^{\frac{r}{\gamma+x_i}}, x_i\}_{1\leq i\leq n}) = g^{\frac{r}{\prod_{i=1}^{n}(\gamma+x_i)}}.$$

using $O(n^2)$ exponentiations.

Although the algorithm Aggregate of [DP08] is given for elements in $\mathbb{G}_T$, it is immediate to see that it works in any group of prime order. Running Aggregate for elements in $\mathbb{G}_1$ results in our case in a more efficient decryption algorithm.

3.1 Description of the Scheme

Setup, $\mathsf{ABE.Setup}(1^\lambda, \mathcal{P})$.
The master entity chooses a suitable encoding τ sending each of the m attributes $\mathsf{at} \in \mathcal{P}$ onto a (different) element $\tau(\mathsf{at}) = x \in (\mathbb{Z}/p\mathbb{Z})^\star$. He also chooses a bilinear group triple $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of prime order p (such that p is λ bits long) and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$. He selects a generator g of $\mathbb{G}_1$ and a generator h of $\mathbb{G}_2$.

After that, he chooses a set $\mathcal{D} = \{d_1, \ldots, d_{m-1}\}$ consisting of $m-1$ pairwise different elements of $(\mathbb{Z}/p\mathbb{Z})^\star$, which must also be different to the values $x = \tau(\mathsf{at})$, for all $\mathsf{at} \in \mathcal{P}$. For any integer i lower or equal to $m-1$, we denote as $\mathcal{D}_i$ the set $\{d_1, \ldots, d_i\}$. Next, the master entity picks at random $\alpha, \gamma \in (\mathbb{Z}/p\mathbb{Z})^\star$ and sets $u = g^{\alpha\gamma}$ and $v = e(g^\alpha, h)$. The master secret key is then $\mathsf{msk} = (g, \alpha, \gamma)$ and the public parameters are

$$\mathsf{params} = \left\{\mathcal{P}, m, u, v, \left\{h^{\alpha\gamma^i}\right\}_{i=0,\ldots,2m-1}, \mathcal{D}, \tau\right\}.$$

Key Extraction, $\mathsf{ABE.Ext}(\mathsf{params}, A, \mathsf{msk})$.
Given any subset $A \subset \mathcal{P}$ of attributes, the master entity picks $r \in (\mathbb{Z}/p\mathbb{Z})^\star$ at random and computes $\mathsf{sk}_A = \left\{\left\{g^{\frac{r}{\gamma+\tau(\mathsf{at})}}\right\}_{\mathsf{at}\in A}, \left\{h^{r\gamma^i}\right\}_{i=0,\ldots,m-2}, h^{\frac{r-1}{\gamma}}\right\}.$

Encryption, $\mathsf{ABE.Enc}(\mathsf{params}, S, t, M)$.
Given a subset $S \subset \mathcal{P}$ with $s = |S|$ attributes, a threshold t satisfying $1 \leq t \leq s$, and a message $M \in \mathbb{G}_T$, the sender picks at random $\kappa \in (\mathbb{Z}/p\mathbb{Z})^\star$ and computes

$$\begin{cases} C_1 = u^{-\kappa}, \\ C_2 = h^{\kappa \cdot \alpha \cdot \prod_{\mathsf{at} \in S} (\gamma + \tau(\mathsf{at})) \prod_{d \in \mathcal{D}_{m+t-1-s}} (\gamma + d)}, \\ K \ = v^\kappa. \end{cases}$$

The value C_2 is computed from the set $\{h^{\alpha\gamma^i}\}_{i=0,\ldots,2m-1}$ that can be found in the public parameters. The ciphertext is then (C_1, C_2, C_3), where $C_3 = K \cdot M$.

Decryption, $\mathsf{ABE.Dec}(\mathsf{params}, (C_1, C_2, C_3), (S, t), \mathsf{sk}_A)$.
Any user with a set of attributes A such that $|A \cap S| \geq t$ can use the secret key sk_A to decrypt the ciphertext, as follows. Let A_S be any subset of $A \cap S$ with $|A_S| = t$. The user computes, from all $\mathsf{at} \in A_S$, the value

$$\mathsf{Aggregate}(\{g^{\frac{r}{\gamma + \tau(\mathsf{at})}}, \tau(\mathsf{at})\}_{\mathsf{at} \in A_S}) = g^{\frac{r}{\prod_{\mathsf{at} \in A_S} (\gamma + \tau(\mathsf{at}))}}.$$

With the output of the algorithm $\mathsf{Aggregate}$ the user computes

$$L = e(g^{\frac{r}{\prod_{\mathsf{at} \in A_S} (\gamma + \tau(\mathsf{at}))}}, C_2) = e(g, h)^{r \cdot \kappa \cdot \alpha \cdot \prod_{\mathsf{at} \in S \setminus A_S} (\gamma + \tau(\mathsf{at})) \prod_{d \in \mathcal{D}_{m+t-1-s}} (\gamma + d)}.$$

For simplicity we define $\tau(d) = d$ for all $d \in \mathcal{D}$ and given a set $A_S \subset S$, $P_{(A_S, S)}(\gamma)$ is

$$P_{(A_S,S)}(\gamma) = \frac{1}{\gamma} \left(\prod_{\mathsf{at} \in (S \cup \mathcal{D}_{m+t-1-s}) \setminus A_S} (\gamma + \tau(\mathsf{at})) - \prod_{\mathsf{at} \in (S \cup \mathcal{D}_{m+t-1-s}) \setminus A_S} \tau(\mathsf{at}) \right).$$

The crucial point is that, since $|A_S| \geq t$, the degree of the polynomial $P_{(A_S,S)}(X)$ is lower or equal to $m - 2$. Therefore, from the values included in sk_A, the user can compute $h^{rP_{(A_S,S)}(\gamma)}$.

After that, the user calculates

$$e(C_1, h^{rP_{(A_S,S)}(\gamma)}) \cdot L = e(g, h)^{\kappa \cdot r \cdot \alpha \cdot \prod_{\mathsf{at} \in (S \cup \mathcal{D}_{m+t-1-s}) \setminus A_S} \tau(\mathsf{at})} \quad (1)$$

and

$$e(C_1, h^{\frac{r-1}{\gamma}}) = e(g, h)^{-\kappa \cdot \alpha \cdot r} \cdot e(g, h)^{\kappa \cdot \alpha} \quad (2)$$

From Equation (1) the user can obtain

$$e(g, h)^{\kappa \cdot r \cdot \alpha} = \left(e(C_1, h^{rP_{(A_S,S)}(\gamma)}) \cdot L \right)^{1 / \prod_{\mathsf{at} \in (S \cup \mathcal{D}_{m+t-1-s}) \setminus A_S} \tau(\mathsf{at})}$$

and multiply this value in Equation (2). The result of this multiplication leads to $K = e(g, h)^{\kappa \cdot \alpha}$. Finally, the user recovers the message by computing $M = C_3 / K$.

3.2 Consistency Checking and Efficiency Considerations

It is not hard to prove that the new ABE scheme satisfy the correctness property: if all the protocols are correctly executed, and if $|A \cap S| \geq t$, then sk_A allows to recover plaintexts that have been encrypted for the pair (S, t).

It is worth noting that, by adding g^α to the public parameters (this modification does not affect the security proof that we present in the next section), the users can check the consistency of the secret key they receive from the master entity. To do so, they must verify that, for all their attributes $\mathsf{at} \in A$,

$$e\left(g^{\frac{r}{\gamma+\tau(\mathsf{at})}}, h^{\alpha\gamma} \cdot (h^\alpha)^{\tau(\mathsf{at})}\right) = e\left(g^\alpha, h^r\right)$$

and then that, for $i = 1, \ldots, m-2$,

$$e\left(g^\alpha, h^{r\gamma^i}\right) = e\left(u, h^{r\gamma^{i-1}}\right)$$

Finally, they have to check that $e(u, h^{\frac{r-1}{\gamma}}) = e\left(g^\alpha, h^r\right)/v$.

In terms of efficiency, the main contribution of this new scheme is the constant size of the ciphertext, which consists of one element of each group $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$. The encryption requires no pairing computations, but $m+t+1$ exponentiations. The decryption process requires 3 pairing evaluations and $O(t^2+m)$ exponentiations. The size of the secret key is linear in the number of attributes, as in all existing ABE schemes.

3.3 Security Analysis

We are going to prove that our scheme is $\mathsf{IND\text{-}sCPA}$ secure, assuming that the $\mathsf{aMSE\text{-}DDH}$ problem is hard to solve.

Theorem 1. *Let λ be an integer. For any adversary $\mathcal{A}$ against the $\mathsf{IND\text{-}sCPA}$ security of our attribute-based encryption scheme, for a universe of m attributes $\mathcal{P}$, and a challenge pair (S, t) with $s = |S|$, there exists a solver $\mathcal{B}$ of the $(\tilde{\ell}, \tilde{m}, \tilde{t})$-$\mathsf{aMSE\text{-}DDH}$ problem, for $\tilde{\ell} = m - s$, $\tilde{m} = m + t - 1$ and $\tilde{t} = t + 1$, such that*

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{aMSE\text{-}DDH}}(\lambda) \;\geq\; \frac{1}{2} \cdot \mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}sCPA}}(\lambda).$$

Proof. We are going to construct an algorithm $\mathcal{B}$ that uses the adversary $\mathcal{A}$ as a black-box and that solves the $(m-s, m+t-1, t+1)$-augmented multi-sequence of exponents decisional Diffie-Hellman problem. The main trick in the proof will be to use the input of the $\mathsf{aMSE\text{-}DDH}$ problem to compute evaluations of some polynomials in γ "in the exponent".

Let $\mathcal{I}(\overrightarrow{x}_{2m+t-1-s}, \kappa, \alpha, \gamma, \omega, T)$ be the input of the algorithm $\mathcal{B}$. First, $\mathcal{B}$ specifies a universe of attributes, $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_m\}$. Next, the adversary $\mathcal{A}$ chooses a set $S \subset \mathcal{P}$ of cardinal s that he wants to attack, and a threshold t such that $1 \leq t \leq s$. Without loss of generality, we assume $S = \{\mathsf{at}_{m-s+1}, \ldots, \mathsf{at}_m\} \subset \mathcal{P}$. From now on, we will denote by A_S the subset $A \cap S$, for any subset of attributes A.

Simulation of the setup. The algorithm $\mathcal{B}$ defines the encoding of the attributes as $\tau(\mathsf{at}_i) = x_i$ for $i = 1, \ldots, m$. Observe that the encodings of the first $m - s$ elements are the opposite of the roots of $f(X)$, and the encodings of the attributes in S are the opposite of some roots of $g(X)$.

The values corresponding to the "dummy" attributes $\mathcal{D} = \{d_1, \ldots, d_{m-1}\}$ are defined as $d_j = x_{m+j}$ if $j = 1 \ldots m + t - 1 - s$. For $j = m + t - s, \ldots, m - 1$, the d_j's are picked uniformly at random in $(\mathbb{Z}/p\mathbb{Z})^\star$ until they are distinct from $\{x_1, \ldots, x_{2m+t-1-s}, d_{m+t-s}, \ldots, d_{j-1}\}$.

The algorithm $\mathcal{B}$ defines $g := g_0^{f(\gamma)}$. Note that $\mathcal{B}$ can compute g with the elements of line (1.1) of its input, since f is a polynomial of degree $\tilde{\ell}$. To complete the setup phase, $\mathcal{B}$ sets $h = h_0$ and computes

- $u = g^{\alpha\gamma} = g_0^{\alpha\cdot\gamma\cdot f(\gamma)}$ with line (1.3) of its input, which is possible since $Xf(X)$ is a polynomial of degree $\tilde{\ell} + 1$. Indeed, $\alpha \cdot \gamma \cdot f(\gamma)$ is a linear combination of $\{\alpha\gamma, \ldots, \alpha\gamma^{\tilde{\ell}+1}\}$ and the coefficients of this linear combination are known to $\mathcal{B}$, so the value u can be computed from line (1.3).
- $v = e(g, h)^\alpha = e(g_0^{f(\gamma)\alpha}, h_0)$ with line (1.3) for $g_0^{f(\gamma)\alpha}$. Note that the value g^α could be computed by $\mathcal{B}$ and added to the public parameters, in case the verification of the consistency of the secret keys is desired for the scheme.

The algorithm $\mathcal{B}$ can compute the values $\{h^{\alpha\gamma^i}\}_{i=0,\ldots,2m-1}$ from line (1.6) of its input. Eventually, $\mathcal{B}$ gives to $\mathcal{A}$ the resulting

$$\mathsf{params} = \{\mathcal{P}, m, u, v, \{h^{\alpha\gamma^i}\}_{i=0,\ldots,2m-1}, \mathcal{D}, \tau\}.$$

Simulation of key extraction queries. Whenever the adversary $\mathcal{A}$ makes a key extraction query for a subset of attributes $A = \{\mathsf{at}_{i_1}, \ldots, \mathsf{at}_{i_n}\} \subset \mathcal{P}$ satisfying that $0 \leq |A_S| \leq t - 1$, the algorithm $\mathcal{B}$ must produce a tuple of the form

$$\mathsf{sk}_A = \left\{ \left\{ g^{\frac{r}{\gamma+\tau(\mathsf{at})}} \right\}_{\mathsf{at}\in A}, \left\{ h^{r\gamma^i} \right\}_{i=0,\ldots,m-2}, h^{\frac{r-1}{\gamma}} \right\},$$

for some random value $r \in (\mathbb{Z}/p\mathbb{Z})^\star$. To do so, $\mathcal{B}$ implicitly defines $r = (\omega y_A \gamma + 1)Q_A(\gamma)$, where y_A is randomly picked in $(\mathbb{Z}/p\mathbb{Z})^\star$, and the polynomial $Q_A(X)$ is defined as $Q_A(\gamma) = 1$ when $|A_S| = 0$, or $Q_A(X) = \lambda_A \cdot \prod_{\mathsf{at}\in A_S} (X + \tau(\mathsf{at}))$ otherwise, in which case $\lambda_A = (\prod_{\mathsf{at}\in A_S} \tau(\mathsf{at}))^{-1}$.

The elements which form sk_A are then computed as follows:

- For any $\mathsf{at} \in A_S$, $\mathcal{B}$ defines

$$Q_{\mathsf{at}}(\gamma) = Q_A(\gamma)/(\gamma + \tau(\mathsf{at})) = \lambda_A \cdot \prod_{\tilde{\mathsf{at}}\in A_S,\ \tilde{\mathsf{at}}\neq\mathsf{at}} (\gamma + \tau(\tilde{\mathsf{at}})).$$

  Then $g^{\frac{r}{\gamma+\tau(\mathsf{at})}} = g_0^{f(\gamma)\omega y_A \gamma Q_{\mathsf{at}}(\gamma)} \cdot g_0^{f(\gamma)Q_{\mathsf{at}}(\gamma)}$. The first factor of the product (whose exponent is a polynomial in γ of degree at most $(m - s) + 1 + t - 2$) can be computed from line (1.2), whereas the second factor (whose exponent is a polynomial in γ of degree at most $(m - s) + t - 2$) can be computed from line (1.1).

– For any $\mathsf{at} \in A \setminus A_S$, $\mathcal{B}$ defines the polynomial $f_{\mathsf{at}}(X) = f(X)/(X + \tau(\mathsf{at}))$. Then $g^{\frac{r}{\gamma+\tau(\mathsf{at})}} = g_0^{f_{\mathsf{at}}(\gamma)\omega y_A \gamma Q_A(\gamma)} \cdot g_0^{f_{\mathsf{at}}(\gamma) Q_A(\gamma)}$. Again, the first factor of this product can be computed from line (l.2), and the second factor can be computed from line (l.1).
– The values $\left\{h^{r\gamma^i}\right\}_{i=0,\ldots,m-2}$ can be computed from line (l.4) and (l.5), since $h^{r\gamma^i} = h^{Q_A(\gamma)\omega y_A \gamma^{i+1}} \cdot h^{Q_A(\gamma)\gamma^i}$.
– Finally, $\mathcal{B}$ has to compute $h^{\frac{r-1}{\gamma}} = h^{Q_A(\gamma)\omega y_A} \cdot h^{\frac{Q_A(\gamma)-1}{\gamma}}$. The first factor of the product can be computed from line (l.5) and the second factor can be computed from line (l.4), since by definition of λ_A, $Q_A(X)$ is a polynomial with independent term equal to 1 and thus $\frac{Q_A(\gamma)-1}{\gamma}$ is a linear combination of $\{1, \gamma, \ldots, \gamma^{t-2}\}$.

Note that $Q_A(\gamma) \neq 0$ (otherwise $\gamma = \tau(at)$ for some $\mathsf{at} \in A_S$ and γ is public), in which case it is not hard to see that r is uniformly distributed in $\mathbb{Z}/p\mathbb{Z}$. If the choice of y_A leads to $r = 0$ (which occurs only with negligible probability anyhow), it suffices to pick a different value for y_A. That is, in the simulation r is uniformly distributed in $(\mathbb{Z}/p\mathbb{Z})^\star$.

Simulation of the challenge. Once $\mathcal{A}$ sends to $\mathcal{B}$ the two messages M_0 and M_1, $\mathcal{B}$ flips a coin $b \in \{0, 1\}$, and sets $C_3^\star = T \cdot M_b$. To simulate the rest of the challenge ciphertext, $\mathcal{B}$ implicitly defines the randomness for the encryption as $\kappa' = \kappa/\alpha$, and sets $C_2^\star = h_0^{\kappa \cdot g(\gamma)}$ (given in line (l.4) of the aMSE-DDH input). To complete the ciphertext, $\mathcal{B}$ computes $C_1^\star = \left(g_0^{\kappa \cdot \gamma f(\gamma)}\right)^{-1}$ from line (l.1) of the input, which is equal to $u^{-\kappa'}$.

After the challenge step $\mathcal{A}$ may make other key extraction queries, which are answered as before.

Guess. Finally, $\mathcal{A}$ outputs a bit b'. If $b' = b$, $\mathcal{B}$ answers 1 as the solution to the given instance of the aMSE-DDH problem, meaning that $T = e(g_0, h_0)^{\kappa \cdot f(\gamma)}$. Otherwise, $\mathcal{B}$ answers 0, meaning that T is a random element.

We now have to analyze the advantage of the algorithm $\mathcal{B}$:

$$\begin{aligned}\mathsf{Adv}_{\mathcal{B}}^{\mathsf{aMSE\text{-}DDH}}(\lambda) &= \Big| \Pr\left[\mathcal{B}(\mathcal{I}(\overrightarrow{x}_{\tilde{\ell}+\tilde{m}}, \kappa, \alpha, \gamma, \omega, T)) = 1 \middle| \mathsf{real}\right] - \\ &\qquad \Pr\left[\mathcal{B}(\mathcal{I}(\overrightarrow{x}_{\tilde{\ell}+\tilde{m}}, \kappa, \alpha, \gamma, \omega, T)) = 1 \middle| \mathsf{random}\right] \Big| \\ &= \Big| \Pr\left[b = b' \middle| \mathsf{real}\right] - \Pr\left[b = b' \middle| \mathsf{random}\right] \Big|.\end{aligned}$$

When the event real occurs, then $\mathcal{A}$ is playing a real attack and therefore $|\Pr\left[b = b' \middle| \mathsf{real}\right] - 1/2| = \frac{1}{2}\mathsf{Adv}_{\mathcal{A},\Pi}^{\mathsf{IND\text{-}sCPA}}(\lambda)$. During the random event, the view of $\mathcal{A}$ is completely independent of the bit b; in this case, the probability $\Pr[b = b']$ is equal to 1/2. Summing up, we obtain

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{aMSE\text{-}DDH}}(\lambda) \geq \frac{1}{2}\mathsf{Adv}_{\mathcal{A},\Pi}^{\mathsf{IND\text{-}sCPA}}(\lambda). \qquad \square$$

4 Extensions

In this section we discuss two possible extensions of the basic scheme that we have described and analyzed in the previous section. First, we study the possibility of supporting more general decryption policies, not only threshold ones. After that, we discuss the options to obtain security against chosen ciphertext attacks.

4.1 More General Decryption Policies

Although we have considered in this paper the special case of threshold decryption policies, attribute-based encryption schemes can be defined for general decryption policies. Such a policy is determined by a monotone increasing family $\Gamma \subset 2^{\mathcal{P}}$ of subsets of attributes, in $\mathcal{P} = \{\mathsf{at}_1, \dots, \mathsf{at}_n\}$. This family (or *access structure*) is chosen by the sender at the time of encryption, in such a way that only users whose subset of attributes A belong to Γ can decrypt. Even if many users collude, each of them having a subset of attributes out of Γ, the encryption scheme must remain secure.

The threshold ABE scheme that we have described and analyzed in this paper is inspired on the dynamic threshold identity-based encryption scheme of [DP08]. It is claimed in [DP08] that the threshold scheme there can be extended to admit "all the classical cases" of more general access structures. However, this is not completely true, because their extension only applies to a sub family of access structures, *weighted threshold* ones. A family $\Gamma \subset 2^{\mathcal{P}}$ is a weighted threshold access structure if there exist a threshold t and an assignment of weights $\omega : \mathcal{P} \to \mathbb{Z}^+$ such that $A \in \Gamma \iff \sum\limits_{\mathsf{at} \in A} \omega(\mathsf{at}) \geq t$. Of course, there are many access structures which are not weighted threshold, for example $\Gamma = \{\{\mathsf{at}_1, \mathsf{at}_2\}, \{\mathsf{at}_2, \mathsf{at}_3\}, \{\mathsf{at}_3, \mathsf{at}_4\}\}$ in the set $\mathcal{P} = \{\mathsf{at}_1, \mathsf{at}_2, \mathsf{at}_3, \mathsf{at}_4\}$.

The same extension proposed in [DP08] works for our threshold ABE scheme. Let K be an upper bound for $\omega(\mathsf{at})$, for all $\mathsf{at} \in \mathcal{P}$ and for all possible assignments of weights that realize weighted threshold decryption policies. During the setup of the ABE scheme, the new universe of attributes will be $\mathcal{P}' = \{\mathsf{at}_1||1, \mathsf{at}_1||2, \dots, \mathsf{at}_1||K, \dots, \mathsf{at}_n||1, \dots, \mathsf{at}_n||K\}$. During the secret key request phase, if an attribute at belongs to the requested subset $A \subset \mathcal{P}$, the secret key sk_A will contain the elements $g^{\frac{r}{\gamma + \tau(\mathsf{at}^{(j)})}}$ corresponding to $\mathsf{at}^{(j)} = \mathsf{at}||j$, for all $j = 1, \dots, K$.

Later, suppose a sender wants to encrypt a message for a weighted threshold decryption policy Γ, defined on a subset of attributes $S = \{\mathsf{at}_1, \dots, \mathsf{at}_s\}$ (without loss of generality). Let t and $\omega : S \to \mathbb{Z}^+$ be the threshold and assignment of weights that realize Γ. The sender can use the threshold ABE encryption routine described in Section 3.1, with threshold t, but applied to the set of attributes $S' = \{\mathsf{at}_1||1, \dots, \mathsf{at}_1||\omega(\mathsf{at}_1), \dots, \mathsf{at}_s||1, \dots, \mathsf{at}_s||\omega(\mathsf{at}_s)\}$. In this way, if a user holds a subset of attributes $A \in \Gamma$, he will have $\omega(\mathsf{at})$ valid elements in his secret key, for each attribute $\mathsf{at} \in A$. In total, he will have $\sum\limits_{\mathsf{at} \in A} \omega(\mathsf{at}) \geq t$ valid elements, so he will be able to run the decryption routine of the threshold ABE scheme and decrypt the ciphertext.

The security analysis can be extended to this more general case, as well. Therefore, we can conclude that our ABE scheme with constant size ciphertexts also admits weighted threshold decryption policies.

4.2 Security under Chosen Ciphertext Attacks

Some ABE schemes proposed in the literature [BSW07, CN07, Wat08] achieve security under selective chosen ciphertext attacks (sCCA security). This is done in two steps. Firstly sCPA security is proved, and secondly the scheme is shown to admit delegation of secret keys: it is possible to compute a valid secret key $\mathsf{sk}_{A'}$ from a valid secret key sk_A, for any $A' \subset A$. If this is the case, the basic ABE scheme can be viewed as a hierarchical ABE scheme, where the hierarchy is the classical one: a user holding attributes A is over a user holding attributes A', if $A' \subset A$. Finally, the techniques developed in [CHK04] can be applied to this sCPA secure hierarchical ABE scheme, which results in a sCCA secure ABE scheme, in the standard model.

Unfortunately our scheme does not seem to admit delegation of secret keys. Therefore, it is still an open problem to come up with an ABE scheme with constant size ciphertexts, achieving sCCA security in the standard model. In contrast, if one requires security in the random oracle model only, such a result is easily obtained by applying to our scheme (a variant of) some classical CPA to CCA transformation, such as the Fujisaki-Okamoto one [FuOk99].

5 Conclusion

We have proposed in this paper the first (reasonably expressive) attribute-based encryption scheme with constant size ciphertexts. The design of the scheme is inspired by the dynamic threshold encryption scheme in [DP08]. Our ABE scheme works for threshold policies: the sender chooses, at the time of encryption, the involved set of attributes and a threshold, in such a way that only those users holding (at least) this threshold of the involved attributes can decrypt. However, the scheme can be easily extended to admit weighted threshold decryption policies, as well.

Although finding attribute-based encryption schemes with short ciphertexts supporting even more expressive decryption policies is an important open problem, weighted threshold decryption policies are quite expressive and can cover a wide range of applications. Therefore, we think that our proposal achieves a fair trade-off between expressiveness and efficiency.

Our scheme employs bilinear pairings, and its security is based on the assumption that a newly introduced problem, the augmented Multi-Sequence of Exponents Decisional Diffie-Hellman (aMSE-DDH) problem, is hard. It remains an open problem to obtain a scheme with constant ciphertext's length whose security is based on a more standard algorithmic problem and which achieves full security (i.e. not only selective security).

Acknowledgments

This work was partially done while Javier Herranz and Carla Ràfols were visiting Université de Caen Basse-Normandie.

The work of Javier Herranz is supported by a *Ramón y Cajal* grant, partially funded by the European Social Fund (ESF) of the Spanish MICINN Ministry. Carla Ràfols holds an *FPI* grant of the Spanish MICINN Ministry. The work of both these authors is partially supported by the Spanish MICINN Ministry under project MTM2009-07694. The work of Fabien Laguillaumie is supported by the French ANR-07-TCOM-013-04 PACE Project.

References

[AMS06] Al-Riyami, S., Malone-Lee, J., Smart, N.P.: Escrow-free encryption supporting cryptographic workflow. International Journal of Information Security 5(4), 217–229 (2006)

[BM05] Bagga, W., Molva, R.: Policy-based cryptography and applications. In: S. Patrick, A., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 72–87. Springer, Heidelberg (2005)

[BSW07] Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 321–334. IEEE Society Press, Los Alamitos (2007)

[BBG05] 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)

[CG99] 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)

[CHK04] Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)

[CCZ06] Chai, Z., Cao, Z., Zhou, Y.: Efficient ID-based broadcast threshold decryption in ad hoc network. In: Proceedings of IMSCCS 2006, vol. 2, pp. 148–154. IEEE Computer Society, Los Alamitos (2006)

[CN07] Cheung, L., Newport, C.C.: Provably secure ciphertext policy ABE. In: Proceedings of Computer and Communications Security, CCS 2007, pp. 456–465. ACM, New York (2007)

[Cha07] Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)

[DHMR07] Daza, V., Herranz, J., Morillo, P., Ràfols, C.: CCA2-secure threshold broadcast encryption with shorter ciphertexts. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 35–50. Springer, Heidelberg (2007)

[DHMR08] Daza, V., Herranz, J., Morillo, P., Ràfols, C.: Extended access structures and their cryptographic applications. To appear in Applicable Algebra in Engineering, Communication and Computing (2008), http://eprint.iacr.org/2008/502

[DP08] Delerablée, C., Pointcheval, D.: Dynamic threshold public-key encryption. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 317–334. Springer, Heidelberg (2008)

[EM+09] Emura, K., Miyaji, A., Nomura, A., Omote, K., Soshi, M.: A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In: Bao, F., Li, H., Wang, G. (eds.) ISPEC 2009. LNCS, vol. 5451, pp. 13–23. Springer, Heidelberg (2009)

[FST10] Freeman, D., Scott, M., Teske, E.: A taxonomy of pairing-friendly elliptic curves. Journal of Cryptology 23(2), 224–280 (2010)

[FuOk99] Fujisaki, E., Okamoto, T.: How to enhance the security of public-key encryption at minimum cost. In: Imai, H., Zheng, Y. (eds.) PKC 1999. LNCS, vol. 1560, pp. 53–68. Springer, Heidelberg (1999)

[GJPS08] 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)

[GPSW06] Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of Computer and Communications Security, CCS 2006, pp. 89–98. ACM, New York (2006)

[KSW08] 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)

[LO+10] Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. To appear in Proceedings of Eurocrypt 2010 (2010), http://eprint.iacr.org/2010/110

[SW05] Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)

[Sha79] Shamir, A.: How to share a secret. Communications of the ACM 22, 612–613 (1979)

[Sha84] 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)

[Wat08] Waters, B.: Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. (2008), manuscript available at, http://eprint.iacr.org/2008/290

Algebraic Cryptanalysis of the PKC'2009 Algebraic Surface Cryptosystem

Jean-Charles Faugère and Pierre-Jean Spaenlehauer

UPMC, Université Paris 06, LIP6
INRIA Centre Paris-Rocquencourt, SALSA Project
CNRS, UMR 7606, LIP6
Boîte courrier 169 – 4, place Jussieu, 75252 Paris Cedex 05, France
`Jean-Charles.Faugere@inria.fr`, `Pierre-Jean.Spaenlehauer@lip6.fr`

Abstract. In this paper, we fully break the Algebraic Surface Cryptosystem (ASC for short) proposed at PKC'2009 [3]. This system is based on an unusual problem in multivariate cryptography: the Section Finding Problem. Given an algebraic surface $X(x, y, t) \in \mathbb{F}_p[x, y, t]$ such that $\deg_{xy} X(x, y, t) = w$, the question is to find a pair of polynomials of degree d, $u_x(t)$ and $u_y(t)$, such that $X(u_x(t), u_y(t), t) = 0$. In ASC, the public key is the surface, and the secret key is the section. This asymmetric encryption scheme enjoys reasonable sizes of the keys: for recommended parameters, the size of the secret key is only 102 bits and the size of the public key is 500 bits. In this paper, we propose a message recovery attack whose complexity is quasi-linear in the size of the secret key. The main idea of this algebraic attack is to decompose ideals deduced from the ciphertext in order to avoid to solve the section finding problem. Experimental results show that we can break the cipher for recommended parameters (the security level is 2^{102}) in 0.05 seconds. Furthermore, the attack still applies even when the secret key is very large (more than 10000 bits). The complexity of the attack is $\widetilde{\mathcal{O}}(w^7 d \log(p))$ which is polynomial with respect to all security parameters. In particular, it is quasi-linear in the size of the secret key which is $(2d + 2)\log(p)$. This result is rather surprising since the algebraic attack is often more efficient than the legal decryption algorithm.

Keywords: Multivariate Cryptography, Algebraic Cryptanalysis, Section Finding Problem (SFP), Gröbner bases, Decomposition of ideals.

1 Introduction

In 1994, Shor designed a quantum algorithm to compute efficiently discrete logarithm and factorization [16]. Hence, if one could construct a quantum computer, a huge number of well established public key cryptosystems – for instance, RSA or Elliptic Curve based systems – would be seriously threatened. Therefore, cryptographers are continuously searching for post-quantum alternatives. The first step to design new cryptosystems is to identify hard problems to use as trapdoors. So far, most of the problems used in post-quantum cryptology can be classified into three main categories: Multivariate cryptography, Code-based cryptography and Lattice-based cryptography.

In this context, Akiyama, Goto, and Miyake propose a new multivariate public-key algorithm at PKC'2009: the Algebraic Surface Cryptosystem (*ASC* for short) [3]. Interestingly, its security is based on a difficult problem which is not common:

Section Finding Problem (SFP). Given an algebraic surface defined by the polynomial $X(x, y, t) \in \mathbb{F}_p[x, y, t]$ (where $\mathbb{F}_p$ denotes the finite field of cardinality p), the question is to find two polynomials $u_x(t), u_y(t) \in \mathbb{F}_p[t]$ of degree d, such that $X(u_x(t), u_y(t), t) = 0$.

As stated in [3], this problem is computationally hard: the only algorithm known so far induces to find roots of a huge multivariate polynomial system. Hence the idea of ASC is to use the surface as public key and the knowledge of a section of this surface as the trapdoor. In comparison to HFE [15] or other multivariate systems, ASC has some interesting and unusual properties. In particular, the keys are unexpectedly short. The security of multivariate systems is usually related to the difficulty of finding a zero of a system of low degree polynomials (often quadratic) in a huge number of variables. For instance, in the case of HFE, the size of the public key is precisely the size of the multivariate system: 265680 bits for a security of 2^{80}. In contrast with HFE, ASC enjoys a small public key of 500 bits for a security of 2^{102}. More generally, for a security level of 2^d, the size of the public key of HFE is $\mathcal{O}(d^3)$. In comparison, the public key of ASC is a unique high degree polynomial in only three variables: its size is $\mathcal{O}(d)$ bits for a security of 2^d. Actually, the authors explains that the keys of ASC are among the shortest of known post-quantum cryptosystems. More precisely, let w denote the degree of the public surface X in x and y. For a security level of p^{2d}, the size of the secret key is $2d \log(p)$ bits and the size of the public key is about $wd \log(p)$. The main observation is that the sizes of the keys are linear in $d \log(p)$, which is the logarithm of the security level.

Although a completely different version of ASC [2] has been attacked by Ivanov and Voloch [11], by Uchiyama and Tokunaga [17] and by Iwami [12], the new version of ASC, presented at PKC'2009, is resistant to all known attacks. We would like to mention that the decryption algorithm raises some questions. Indeed, one step of this algorithm is to recover some factors of given degree D of a univariate polynomial. In order to find those factors, the designers propose to recombine the irreducible factors of the polynomial by solving a knapsack. However, this problem is known to be NP-hard [10]. Therefore, it is not clear if the cryptosystem remains practical for high security parameters.

Main Results. In this paper, we describe a message recovery attack which can break ASC in polynomial time. One important step of the legal decryption algorithm is the factorization of a univariate polynomial. The key idea of the algebraic attack is to perform this factorization step *implicitly* by decomposing ideals deduced from the ciphertext. Indeed, decomposition of ideals can be seen as a generalisation of the standard factorization of polynomials. Hence, this technique allows us to bypass the Section Finding Problem, which is hard.

We present three versions of this attack. The Level 1 Attack is high-level, deterministic, offers a good view of the mechanisms involved and can be implemented straightforwardly into a Computer Algebra System such as Magma (code given in Appendix B). However, this version is not very efficient and cannot break ASC for the recommended parameters. The Level 2 Attack is based on the following observation: the polynomials occurring in ASC have a high degree in t and a rather low degree in x and y. Thus, it is natural to see expressions in t as *coefficients* instead of polynomials in t; in other words, in order to speed up the attack, we have to perform the computations in the ring $\mathbb{F}_p(t)[x, y]$ (where $\mathbb{F}_p(t)$ is the field of fractions) instead of $\mathbb{F}_p[x, y, t]$. In the Level 3 Attack, we replace the ground field $\mathbb{F}_p(t)$ by a finite field $\mathbb{F}_{p^D} \approx \mathbb{F}_p[t]/(P(t))$ for a large enough D to avoid the swelling of the intermediate coefficients and to recover the initial message modulo $P(t)$. Even more efficiently, we can split $P(t)$ into several irreducible factors $P_i(t)$ of small degree; the Chinese Remainder Theorem is then used to recombine the congruences and retrieve the original message. In this third version of the attack, the size of the plaintext determines the number of congruences required as well as the size of the finite fields considered. Therefore, the complexity of the Level 3 Attack is expected to be *quasi-linear* in the size of the secret key. This behaviour is confirmed by experimental results together with a complexity analysis. The binary complexity[1] of the Level 3 Attack is (Theorem 1):

$$\widetilde{\mathcal{O}}(w^6 size(m))$$

where $size(m)$ denotes the binary size of the plaintext, w is the degree of X in the variables x and y and $\widetilde{\mathcal{O}}()$ is the "soft Oh" notation (see e.g. [18, Definition 25.8]). Since the size of the secret key is smaller than $size(m)$, the attack is also quasi-linear in the size of the secret key. In practice, $size(m) \approx dw \log(p)$ (where d is the degree of the secret section). Thus the complexity of the attack is

$$\widetilde{\mathcal{O}}(w^7 d \log(p)).$$

This can be compared with a lower bound on the binary complexity (see page 47) of the decryption algorithm:

$$\widetilde{\mathcal{O}}(\log(p)(w^3 d^3 + dw \log(p))).$$

It can be noted that the decryption algorithm is cubic in the size of the secret key. Therefore, increasing the size of the secret key does not secure the system, since the cost of the decryption algorithm increases faster than the cost of the attack.

We implemented in Magma 2.15-7 the three variants. The Level 3 Attack can break ASC with parameters recommended in [3] ($d = 50$, $p = 2$, $w = 5$) in only 0.05 seconds. Experiments confirm that increasing the size of the secret key with the parameters p and d does not really increase the security of the

[1] The binary complexity is the number of arithmetic operations on bits, whereas the arithmetic complexity is the number of arithmetic operations in the base ring.

system. We are still able to break it in few seconds, even when the size of the secret key is more than 10000 bits! We also try to increase the parameter w (the degree in x and y of the public surface). For a reasonable size of the public key (less than 4000 bits), the message can be recovered in few hours. Finally, we try to figure out whether it is possible to secure the system by increasing the size of the support of the surface (the parameter k). However, as predicted by the complexity analysis, this parameter has very few effect on the complexity of the attack. Thereby, we can consider the system as fully broken.

Structure of the Paper
After this introduction, the paper is organized as follows. In Section 2, we give a short description of the ASC cryptosystem as it is presented in [3]. Then, we explain the theoretical foundations of the attack. In Section 3, we describe the three variants of the attack and we show a concrete example by applying it to the toy example given in [3]. We also perform a precise complexity analysis in Section 5. Finally, we give some experimental results showing that the attack is scalable.

2 Description of the Cryptosystem

We give here a short description of ASC. For a more detailed presentation of this cryptosystem, we refer the reader to [3]. We consider the ring of polynomials $\mathbb{F}_p[x, y, t]$ where p is a prime number. For any polynomial $P \in \mathbb{F}_p[x, y, t]$, Λ_P denotes its support in $\mathbb{F}_p(t)[x, y]$ (that is to say the set of couples $(i, j) \in \mathbb{N}^2$ such that $t^\ell x^i y^j$ is a monomial of P).

2.1 Parameters

The cryptosystem ASC has four parameters: p is the cardinality of the ground field, and d is the degree of the secret section. These two parameters are especially important for the security. They have a direct impact on the binary size of the secret key, which is $2d \log p$. Another parameter is w the degree in x and y of the public surface X. The last parameter is k, the cardinality of Λ_X (which is the support of X in $\mathbb{F}_p(t)[x, y]$). The parameters w, d and p have an impact on the size of the public key which is approximatively $dw \log(p)$ bits.

2.2 Keys

The secret key is a pair of polynomials $(u_x(t), u_y(t))$ of degree d.

The public key is given by:

- A surface described by an irreducible polynomial $X(x, y, t) \in \mathbb{F}_p[x, y, t]$ such that $X(u_x(t), u_y(t), t) = 0$ and $\mathsf{card}(\Lambda_X) = k$.
- Λ_m the support of the plaintext polynomial and $\{d_{ij}^{(m)} \in \mathbb{N}\}_{(i,j)\in\Lambda_m}$ the degrees of the coefficients.
- Λ_f the support of the divisor polynomial and $\{d_{ij}^{(f)} \in \mathbb{N}\}_{(i,j)\in\Lambda_f}$ the degrees of the coefficients.

For encryption/decryption it is required that:

$$\begin{gathered}\Lambda_m \subset \Lambda_f \Lambda_X = \{(i_1+i_2, j_1+j_2) : (i_1,j_1) \in \Lambda_f, (i_2,j_2) \in \Lambda_X\}.\\ \max\{i : (i,j) \in \Lambda_X\} < \max\{i : (i,j) \in \Lambda_m\} < \max\{i : (i,j) \in \Lambda_f\}.\\ \max\{j : (i,j) \in \Lambda_X\} < \max\{j : (i,j) \in \Lambda_m\} < \max\{j : (i,j) \in \Lambda_f\}.\\ \deg_t(X(x,y,t)) < \max\{d_{ij}^{(m)}\}_{(i,j)\in\Lambda_m} < \max\{d_{ij}^{(f)}\}_{(i,j)\in\Lambda_f}.\end{gathered}$$

2.3 Encryption/Decryption

Encryption. Consider a plaintext embedded into a polynomial

$$m(x,y,t) = \sum_{(i,j)\in\Lambda_m} m_{ij}(t)x^i y^j$$

where $\deg(m_{ij}(t)) = d_{ij}^{(m)}$. Choose a random divisor polynomial

$$f(x,y,t) = \sum_{(i,j)\in\Lambda_f} f_{ij}(t)x^i y^j$$

where $\deg(f_{ij}(t)) = d_{ij}^{(f)}$. Then select four random polynomials r_0, r_1, s_0, s_1 such that, for $\ell \in \{0,1\}$,

$$r_\ell(x,y,t) = \sum_{(i,j)\in\Lambda_f} r_{ij}^{(\ell)}(t)x^i y^j, \quad s_\ell(x,y,t) = \sum_{(i,j)\in\Lambda_X} s_{ij}^{(\ell)}(t)x^i y^j$$

and $\forall i,j, \deg(r_{ij}^{(\ell)}(t)) = \deg(f_{ij}(t)), \deg(s_{ij}^{(\ell)}(t)) = \deg(X_{ij}(t))$. Finally, construct the ciphertext $(F_0(x,y,t), F_1(x,y,t))$ where

$$\begin{aligned}F_0(x,y,t) &= m(x,y,t) + f(x,y,t)s_0(x,y,t) + X(x,y,t)r_0(x,y,t),\\ F_1(x,y,t) &= m(x,y,t) + f(x,y,t)s_1(x,y,t) + X(x,y,t)r_1(x,y,t).\end{aligned}$$

Decryption. Consider $h_\ell(t) = F_\ell(u_x(t), u_y(t), t), \ell \in \{0,1\}$ and compute the difference $h_0(t)-h_1(t) = f(u_x(t),u_y(t),t)(s_0(u_x(t),u_y(t),t)-s_1(u_x(t),u_y(t),t))$. Next, find a factor of $h_0(t)-h_1(t)$ whose degree matches $\deg(f(u_x(t),u_y(t),t))$. Let $\widetilde{f}(t)$ denote this factor. Then computes $\widetilde{m}(u_x(t),u_y(t),t) = h_0(t) \mod \widetilde{f}(t)$. Finally, retrieve $\widetilde{m}(x,y,t)$ by solving the linear system:

$$\widetilde{m}(u_x(t),u_y(t),t) = \sum \widetilde{m}_{ijk} u_x(t)^i u_y(t)^j t^k.$$

There are potentially several factors of $h_0(t)-h_1(t)$ whose degree is equal to $\deg(f(u_x(t),u_y(t),t))$. So, we have to verify that we picked the good one. To do so, the designers of ASC propose to use a MAC to verify that $\widetilde{m}(x,y,t) = m(x,y,t)$. If the verification fails, we start again by considering another factor of $h_0(t)-h_1(t)$.

To find factors of $h_0(t)-h_1(t)$ whose degree matches $\deg(f(u_x(t),u_y(t),t))$, the designers propose to factor $h_0(t)-h_1(t)$, then recombine its irreducible factors by solving a knapsack problem. However, the knapsack problem is NP-hard [10]. Therefore, as pointed out in [3], it is not clear if the decryption algorithm remains practicable when the security parameters are high.

2.4 Security of the System

The designers of the cryptosystem propose the following parameters:

- $p = 2$.
- d should be greater than 50.
- $w = \deg_{xy}(X) = \max\{i + j : (i, j) \in \Lambda_X\}$ should be greater than 5.
- The lower bound on k is 3.

The size of the secret key is around 100 bits and the size of the public key is close to 500 bits. According to the designers of ASC, there is so far no known attack faster than exhaustive search for these parameters. Therefore, the security level of ASC is expected to be the cost of exhaustive search of the secret key, namely p^{2d+2}.

3 Description of the Attack

Overview of the Attack
In this section, we propose a message recovery attack on the cryptosystem described above.

The main point of the attack is to decompose ideals, instead of factoring the univariate polynomial obtained by evaluating $F_0 - F_1$ in the section (u_x, u_y). This way, we can implicitly manipulate the so-called *divisor polynomial* f occurring in the decryption process. Consequently, we can avoid to solve the underlying Section Finding Problem, and we obtain a polynomial attack on ASC.

First, we present a high-level and deterministic version of the attack (Algorithm 1) based on two fundamental lemmas. Then, we speed-up the algorithm by considering the field of fractions $\mathbb{F}_p(t)$ (Algorithm 2). Indeed, polynomials occurring in ASC have a high degree in t. Since the complexity of Gröbner bases algorithms is linear in the complexity of the arithmetic in the ground field, it seems natural to compute in the field of fractions $\mathbb{F}_p(t)$. Finally, we use a modular approach to implement efficiently the attack: we perform computations in some well-chosen finite fields $\mathbb{F}_p[t]/(P)$ and recombine the results by using the Chinese Remainder Theorem (Algorithm 3). Doing this, the size of the coefficients of intermediate values are bounded (these coefficients can be huge when computations are performed in the field of fractions). This allows us to break bigger instances of ASC. In particular, we are able to break the system with recommended parameters in 0.05 seconds. Furthermore, this will allow us to perform a precise complexity analysis and to show that this attack is quasi-linear in the size of the secret key. Experimentally, we are able to break with this technique some instances where the size of the secret key is greater than 10000 bits.

Now we compare the efficiency of the three versions of the attack on a small example. For instance, we consider the following parameters $p = 11$, $d = 8$, $w = 5$ and $k = 3$ and we use our Magma implementation. The Level 1 Attack (code given in Appendix) recover the plaintext in 136 seconds. As predicted, the Level 2 Attack is faster and can break the system in 74 seconds. Using the modular approach in the Level 3 Attack really speeds up the computations: it retrieves the plaintext in 0.06 seconds.

3.1 Level 1 Attack: Decomposition of Ideals

The two following lemmas are the key elements of the attack.

Lemma 1. *Let I be the ideal $I = \langle F_0 - F_1, X\rangle \subset \mathbb{F}_p[x, y, t]$. Then $I = I_1 \cap I_2$ where $I_1 = \langle f, X\rangle$ and $I_2 = \langle s_0 - s_1, X\rangle$. Generically, the ideals I_1 and I_2 are prime ideals of $\mathbb{F}_p[x, y, t]$.*

Proof. $I = \langle F_0 - F_1, X\rangle = \langle f\,(s_0 - s_1), X\rangle = I_1 \cap I_2$.

Lemma 1 shows that, once we managed to decompose the ideal $\langle F_0 - F_1, X\rangle = \langle f\,(s_0 - s_1), X\rangle$, we can manipulate implicitly the polynomial f through I_1.

Remark 1. In order to decompose I, a strategy is to eliminate x from I by computing a Gröbner basis of $I \cap \mathbb{F}_p[y, t]$. Generically, this Gröbner basis contains only one polynomial Q. If p is big enough, Q has in general two factors which depend on y and t (we do not consider the factors which are in $\mathbb{F}_p[t]$). This fact is confirmed experimentally. The two factors correspond to I_1 and I_2. Then, we can construct I_1 (resp. I_2) by adding to I an appropriate factor of Q. Since $\deg_y(f) > \deg_y(s_1 - s_0)$, the factor of Q with the highest degree in y is the one corresponding to I_1. To factor efficiently the bivariate polynomial Q, we can use for instance the algorithm in [14].

Lemma 2. *Let J be the ideal of $\mathbb{F}_p[x, y, t]$ generated by $J = \langle F_0, F_1, X\rangle + I_1$. Then $m(x, y, t) \in J$. Moreover, J is a zero-dimensional ideal.*

Proof. $J = \langle F_0, F_1, X\rangle + I_1 = \langle F_0, F_1, X, f\rangle = \langle m, f, X\rangle$.

Remark 2. Lemma 2 shows that we can compute explicitly a multivariate ideal which contains m. Since we know $\varLambda_m$, we can recover m by solving the following linear system:

$$NF_J(m) = \sum_{(i,j)\in\varLambda_m} \sum_{k=0}^{d_{ij}^{(m)}} m_{ijk} NF_J(x^i y^j t^k) = 0$$

where NF_J denotes the normal form with respect to the ideal J for a chosen monomial ordering (the definition of the normal form is given in Appendix). Since $\lambda m \in J$ for all $\lambda \in \mathbb{F}_p$, we retrieve m up to multiplication by a scalar.

Remark 3. For efficiency purpose, we compute the Gröbner bases with respect to the *graded reverse lexicographical ordering* (Definition 1 in appendix). Instead of computing the Gröbner basis of $\langle F_0 - F_1, X\rangle \cap \mathbb{F}_p[y, t]$, it is also possible to compute a resultant to eliminate the variable x.

Remark 4. The normal form NF_J is a linear application from $\mathbb{F}_p[x, y, t]$ onto $\mathbb{F}_p[x, y, t]/J$. In the last step of the attack, we are searching for the intersection of its kernel with the $\mathbb{F}_p$-linear subspace generated by $\varGamma_m$ (where $\varGamma_m$ denotes the support of m in $\mathbb{F}_p[x, y, t]$). Therefore, the linear system has $\mathsf{card}(\varGamma_m)$ unknowns

Algorithm 1. Level 1 Attack

```
1: Compute a Gröbner basis of the ideal $\langle F_0 - F_1, X\rangle \cap \mathbb{F}_p[y,t]$. Generically this
   Gröbner basis contains only one polynomial $Q(y,t)$.
2: Factor $Q = \prod Q_i(y,t)$. Let $Q_0(y,t) \in \mathbb{F}_p[y,t]$ denote an irreducible factor with
   highest degree with respect to $y$.
3: Compute a Gröbner basis of the ideal $J = \langle F_0, F_1, X, Q_0\rangle$.
4: To retrieve the plaintext (up to multiplication by a scalar in $\mathbb{F}_p$), solve the linear
   system over $\mathbb{F}_p$
       $\sum_{(i,j)\in\Lambda_m} \sum_{k=0}^{d_{ij}^{(m)}} m_{ijk} NF_J(x^i y^j t^k) = 0.$
   If the system has no solution, go back to Step 2 and pick another factor of $Q$.
```

and $\deg(J)$ equations ($\deg(J) = \dim(\mathbb{F}_p[x,y,t]/J)$ when $\mathbb{F}_p[x,y,t]/J$ is seen as a $\mathbb{F}_p$-vector space). From the Bézout bound [13], $\deg(J) \approx \deg(m)\deg(X)\deg(f)$. The decryption algorithm requires that $\deg(m(u_x,u_y,t)) \geq \mathsf{card}(\Gamma_m)$ (in order to solve the final linear system) and one can remark that $\deg(X)\deg(f) > \deg(m(u_x,u_y,t)) \approx d\deg_{xy}(m)+\deg_t(m)$ (since $\deg_{xy}(f) > \deg_{xy}(m)$, $\deg_t(f) > \deg_t(m)$ and $\deg(X) > d$). Therefore, the linear system has more equations than unknowns: $\mathsf{card}(\Gamma_m) \leq \deg(m(u_x,u_y,t)) \leq \deg(X)\deg(f) \leq \deg(J)$.

3.2 Level 2 Attack: Computing in the Field of Fractions $\mathbb{F}_p(t)$

Polynomials appearing in ASC have a high total degree, but their degree in the variables x and y is low. Hence, it is natural to consider these polynomials as bivariate polynomials in x and y over the field of fractions $\mathbb{F}_p(t)$. Indeed, the degree in x and y are completely independent of the security parameter d. In this section, we explain how to adapt the attack in this context. Doing that, we expect to have a lower complexity. Indeed, many operations on ideals – for instance Gröbner basis computations – are linear in the complexity of the arithmetic in the ground field.

From now on, $\mathbb{K}$ denotes the field of fractions $\mathbb{F}_p(t)$.

First, we need to transpose the key lemmas in this new context. This can be done for Lemma 1 without any major modification:

Lemma 3. *Let I be the ideal $I = \langle F_0 - F_1, X\rangle$ (seen as an ideal of $\mathbb{K}[x,y]$). Then there exists I_1 and I_2 two strict ideals of $\mathbb{K}[x,y]$ such that $I = I_1 \cap I_2$ and $\langle f, X\rangle \subset I_1$.*

Unfortunately, Lemma 2 cannot be directly transposed in the context of the field of fractions. Indeed, the variety of the ideal $J = \langle F_0, F_1, X\rangle + I_1 = \langle m, f, X\rangle$ (seen as an ideal of $\mathbb{K}[x,y]$) is generically empty since it is generated by three independent equations. Therefore we have to introduce a new variable z if we want to keep the ideal zero-dimensional and strictly included in $\mathbb{K}[x,y,z]$. Roughly speaking, the role of z is to deform the ideal $\langle m, f, X\rangle$ in order to introduce new elements in the variety:

Algorithm 2. Level 2 Attack: computing in the field of fractions $\mathbb{K} = \mathbb{F}_p(t)$

```
1: Compute the resultant $\mathsf{Res}_x(F_0 - F_1, X) \in \mathbb{K}[y]$.
2: Factor the resultant $\mathsf{Res}_x(F_0 - F_1, X) = \prod Q_i(y)$. Let $Q_0(y) \in \mathbb{K}[y]$ denote an
   irreducible factor of highest degree in $y$.
3: Compute a grevlex-Gröbner basis of the ideal $J = \langle F_0 + z, F_1 + z, X, Q_0 \rangle \subset$
   $\mathbb{K}[x, y, z]$.
4: Consider the following linear system over $\mathbb{K}$:
       $NF_J(z) + \sum_{(i,j) \in \Lambda_m} m_{ij}(t) NF_J(x^i y^j) = 0.$
   If the system has no solution, then go back to Step 2 and choose another factor of
   the resultant.
5: Return $m = \sum_{(i,j) \in \Lambda_m} m_{ij}(t) x^i y^j$ where $(m_{ij}(t))$ is the unique solution of the
   linear system.
```

Lemma 4. *Let $J \subset \mathbb{K}[x, y, z]$ be the ideal $J = \langle F_0 + z, F_1 + z, X \rangle + I_1$. Then $m(x, y, t) + z \in J$. Moreover, J is a zero-dimensional ideal.*

Proof. $\langle F_0 + z, F_1 + z, X \rangle + I_1 = \langle F_0 + z, F_1 + z, X, f \rangle = \langle m + z, f, X \rangle$.

To be able to recover the plaintext, we need to solve a linear system with $\mathsf{card}(\Lambda_m)$ unknowns and $\deg(J)$ equations. In practice, there are more equations than unknowns. Thus, if we choose a wrong factor of the resultant (a factor which is not a divisor of f), then the linear system has generically no solution, and we just have to restart from Step 2 until we find an appropriate factor. In practice, the irreducible factor of the resultant with the highest degree in y is almost always a good choice.

Remark 5. It is also possible to combine the two versions of the attack by computing a Gröbner basis of the elimination ideal and factoring it in $\mathbb{F}_p[x, y, t]$, as in Level 1 attack (Steps 1 and 2 in Algorithm 1). Then, once we found $Q_0 \in \mathbb{F}_p[x, y, t]$, we retrieve the message by computing a Gröbner Basis of $J = \langle F_0 + z, F_1 + z, X, Q_0 \rangle \subset \mathbb{K}[x, y, z]$ in the field of fractions (Steps $3, 4, 5$ in Algorithm 2).

3.3 Level 3 Attack: Computing in Finite Fields $\mathbb{F}_{p^m}$

In this section, we study how to implement efficiently the attack in practice. In order to speed up the attack and to compute efficiently in the field of fractions, we perform all computations modulo polynomials of $\mathbb{F}_p[t]$. Indeed, a bound on the degree of m with respect to t is known since $\deg_t(m) \le \max\{d_{i,j}^{(m)}\}$.

We choose a constant C and $n = \deg_t(m) \log(p)/C$ irreducible polynomials $P_1, \ldots, P_n$ of degree close to $C/\log(p)$ such that $\sum \deg(P_i) > \deg_t(m)$. Then for each P_i, we consider

$$\mathbb{F}_p[t]/(P_i) = \mathbb{F}_{p^{\deg(P_i)}}.$$

Algorithm 3. Level 3 Attack: computing in the finite fields $\mathbb{K} = \mathbb{F}_p[t]/(P)$

```
1: Choose $n \approx \deg_t(m)\log(p)/C$ irreducible polynomials of degree $\approx C/\log(p)$ such
   that $\sum \deg(P_i) > \deg_t(m)$.
2: for $i$ from 1 to $n$ do
3:    Consider $\mathbb{K} = \mathbb{F}_p[t]/(P_i)$.
4:    Compute the resultant $\mathsf{Res}_x(F_0 - F_1, X) \in \mathbb{K}[y]$.
5:    Factor the resultant $\mathsf{Res}_x(F_0 - F_1, X) = \prod Q_i(y)$. Let $Q_0(y) \in \mathbb{K}[y]$ denote an
      irreducible factor of highest degree in $y$.
6:    Compute a grevlex-Gröbner basis of the ideal $J = \langle F_0 + z, F_1 + z, X, Q_0 \rangle \subset$
      $\mathbb{K}[x, y, z]$.
7:    Consider the following linear system over $\mathbb{K}$:
          $$NF_J(z) + \sum_{(i,j)\in\Lambda_m} m_{ij}(t) NF_J(x^i y^j) = 0.$$
      If the system has no solution, then go back to Step 2 and choose another factor
      of the resultant.
8:    Retrieve a congruence $m \mod P_i = \sum_{(i,j)\in\Lambda_m} m_{ij}(t) x^i y^j$ where $(m_{ij}(t))$ is the
      solution of the linear system.
9: end for
10: Use the CRT to retrieve $m = m \mod \prod P_i$.
```

Considering all computations in $\mathbb{K} = \mathbb{F}_p[t]/(P_i)$ instead of $\mathbb{F}_p(t)$, the attack yields $m \mod P_i$. Finally we use the Chinese Remainder Theorem (CRT) to recover $m \mod \prod P_i$. Since $\deg(\prod P_i) > \deg_t(m)$, we retrieve the plaintext.

Remark 6. The linear system at step 7 in Algorithm 3 has only $\mathsf{card}(\Lambda_m)$ unknowns and $\deg(J) \approx \deg_{xy}(m)\deg_{xy}(f)\deg_{xy}(X)$ equations. For practical parameters, $\mathsf{card}(\Lambda_m) \approx k$ is smaller than $\deg(J)$, thus the linear system is overdetermined and has in general only one solution. This fact is confirmed by experiments.

The value $\sum \deg(P_i) \approx \deg_t(m)$ is only dependent of the size of the plaintext. Therefore, the number of times we have to run the main loop of Algorithm 3 is linear in the size of the plaintext. Since the cost of arithmetic operations in $\mathbb{F}_p[t]/(P)$ only depends on C (which is a constant chosen by the attacker), we expect this Level 3 Attack to be linear or quasi-linear in the size of the plaintext. This expectation will be confirmed by a complexity analysis and by experimental results. Besides, we would also like to mention that the main loop of Algorithm 3 can be easily parallelized.

4 A Concrete Example

We consider here the toy example given in [3]. We have

- $p = 17$.
- The secret key is $(u_x, u_y) = (14t^3 + 12t^2 + 5t + 1, 11t^3 + 3t^2 + 5t + 4)$.

- The public surface is
  $X = (t+10)x^3y^2 + (16t^2+7t+4)xy^2 + (3t^{16}+8t^{15}+13t^{14}+8t^{13}+3t^{12}+12t^{11}+4t^{10}+8t^9+7t^8+4t^7+13t^6+2t^5+5t^4+4t^3+14t^2+9t+14)$.
- The support of m and f are

$$\Lambda_m = \{(4,4),(0,0)\}, d^m_{00} = 17, d^m_{44} = 17,$$
$$\Lambda_f = \{(5,5),(1,2),(0,0)\}, d^f_{00} = 13, d^f_{12} = 11, d^f_{55} = 18.$$

Here we show how to recover the message m from the ciphertext (F_0, F_1) (given in [3]) with the Level 3 Attack:

1. Since $\deg_t(m) = 17$, we choose (for instance) $P_1, P_2, P_3, P_4 \in \mathbb{F}_p[t]$ irreducible such that $\sum \deg(P_i) \geq 18$. In particular,
$$P_1 = t^5 + t + 14,$$
$$P_2 = t^5 + 14t^4 + 4t^3 + 4t + 4,$$
$$P_3 = t^5 + 9t^4 + 15t^3 + 8t^2 + 4t + 8,$$
$$P_4 = t^5 + 11t^4 + 11t^3 + 8t^2 + 7t + 8.$$
First, we consider the finite field $\mathbb{K} = \mathbb{F}_p[t]/(P_1)$.
2. Compute the resultant in $\mathbb{K}[y]$:
$\mathsf{Res}_x(F_0 - F_1, X) = (9t^4+14t^3+4t^2+6t+13)y^{30} + (5t^4+t^3+14t^2+15t+8)y^{27} + (6t^4+9t^3+10t^2+7t+14)y^{26} + (7t^4+4t^3+8t^2+5t+8)y^{25} + (8t^4+4t^3+7t^2+7t+6)y^{24} + (12t^4+9t^3+8t^2+13t)y^{23} + (9t^4+4t^3+9t^2+15t+6)y^{22} + (3t^4+6t^3+10t^2+6t+6)y^{21} + (9t^4+9t^3+13t^2+15t+6)y^{20} + (4t^4+4t^3+15t^2)y^{19} + (2t^4+11t^3+2t^2+5t+2)y^{16}$.
3. Then factor it in $\mathbb{K}[y]$:
$\mathsf{Res}_x(F_0 - F_1, X) = y^{16}(y+8t^4+3t^3+16t^2+8t+2)(y^2+2t^4+14t^3+14t^2+6t+10)(y^2+15t^4+3t^3+3t^2+11t+7)(y^2+(14t^4+7t^3+4t)y+13t^4+10t^3+7t^2+8t+1)(y^7+(12t^4+7t^3+t^2+5t+15)y^6+(t^4+5t^3+7t^2+12t+11)y^5+(9t^4+14t^3+5t^2+10t+10)y^4+(4t^4+7t^3+t^2+7t+14)y^3+(11t^4+13t^3+12t^2+8t+4)y^2+(15t^4+9t^3+16t^2+14t+14)y+14t^4+3t^3+9t^2+15t+8)$.
4. Consider Q_0 an irreducible factor with highest degree:
$Q_0 = y^7+(12t^4+7t^3+t^2+5t+15)y^6+(t^4+5t^3+7t^2+12t+11)y^5+(9t^4+14t^3+5t^2+10t+10)y^4+(4t^4+7t^3+t^2+7t+14)y^3+(11t^4+13t^3+12t^2+8t+4)y^2+(15t^4+9t^3+16t^2+14t+14)y+(14t^4+3t^3+9t^2+15t+8)$.
5. Compute a Gröbner basis G with respect to the grevlex ordering of the ideal $J = \langle F_0 + z, F_1 + z, X, Q_0 \rangle \subset \mathbb{K}[x,y,z]$.
6. Since $\Lambda_m = \{(0,0),(4,4)\}$ compute $NF_J(x^4y^4)$:
$NF_J(x^4y^4) = N_1 z + N_2 = (15t^4+3t^3+t^2+13t+16)z + (5t^4+11t^2+t+7)$.
7. Solve the linear system $z + m_{44}NF_J(x^4y^4) + m_{00} = 0$ over $\mathbb{K}$:
$$\begin{cases} m_{00} = N_2/N_1 \mod P_1 \\ m_{44} = -1/N_1 \mod P_1. \end{cases}$$
8. Recover a congruence: $m = m_{00} + m_{44}x^4y^4 \mod P_1$.
9. Repeat the process with P_2, P_3 and P_4.

10. Use the CRT to retrieve $m = m \mod \prod P_i$:
$m = (5t^{17} + 15t^{16} + 4t^{15} + 9t^{14} + 7t^{13} + 2t^{12} + 3t^{11} + 8t^{10} + 11t^9 + 6t^{17} + 6t^8 + 3t^{16} + 10t^7 + 11t^{15} + 7t^6 + t^5 + t^{13} + 14t^4 + 10t^{12} + 3t^3 + 3t^{11} + 12t^2 + 8t^{10} + 11t + 6t^9 + 2)x^4y^4 + (13t^8 + 2t^7 + 2t^6 + 10t^5 + 5t^4 + 2t^3 + 15t^2 + 3t + 11)$.

5 Complexity Analysis

In this part, we investigate the complexity of the Level 3 Attack. To simplify the notations, we suppose here that the complexity of multiplying two $n \times n$ matrices is $\mathcal{O}(n^3)$. We note that C is a parameter chosen by the attacker. This parameter fixes the size of the finite fields considered. Indeed, we choose finite fields $\mathbb{K} = \mathbb{F}_p/(P_i)$ with $\deg(P_i) \approx C/\log(p)$. Hence, $\log(\mathsf{card}(\mathbb{K})) \approx C$.

1. First, we estimate the complexity of the computation of the resultant with respect to x in $\mathbb{K}[x, y]$ (where $\mathbb{K} = \mathbb{F}_p[t]/(P_i)$). According to [18] (Corollary 11.18), this can be done in $\widetilde{\mathcal{O}}(w^3)$ operations in $\mathbb{K}$, and the degree of the resultant is $\mathcal{O}(w^2)$.
2. The probabilistic Cantor-Zassenhaus algorithm [18] factors a polynomial of degree n over a finite field $\mathbb{F}_q$ in $\widetilde{\mathcal{O}}(n^2 + n\log(q))$ arithmetic operations in $\mathbb{F}_q$. Therefore the arithmetic complexity in $\mathbb{K}$ of the factorization of the resultant is
$$\widetilde{\mathcal{O}}(w^4 + w^2 \log(\mathsf{card}(\mathbb{K}))) = \widetilde{\mathcal{O}}(w^4 + w^2 C).$$
3. The degree of regularity of an ideal is an important indicator of the complexity of computing its Gröbner basis with respect to the grevlex ordering: it is the highest degree of the polynomials occuring in the F_5 Algorithm. According to [13,5,4], if an ideal is spanned by m generic equations in n variables, then the complexity of computing a Gröbner basis is:
$$\mathcal{O}\left(m^3 \binom{\mathsf{d_{reg}} + n - 1}{n - 1}^3\right).$$
Since the ideal $J = \langle m + z, f, X\rangle$ is generated by three independent equations, its degree of regularity can be estimated from the Macaulay bound (see [13]) as
$$\mathsf{d_{reg}}(J) = (\deg_{xy}(m + z) - 1) + (\deg_{xy}(f) - 1) + (\deg(X)_{xy} - 1) + 1.$$
For practical parameters, $\deg_{xy}(m+z) \approx \deg_{xy}(f) \approx \deg(X)_{xy} \approx w$. Therefore, $\mathsf{d_{reg}} \approx 3w$. The arithmetic complexity in $\mathbb{K}$ of the Gröbner basis computation is then:
$$\mathcal{O}\left(3^3 \binom{\mathsf{d_{reg}}(J) + 2}{2}^3\right) = \mathcal{O}(w^6).$$

4. Finally we have a linear system to solve. The number of variables is $\mathsf{card}(\Lambda_m)$. For practical parameters, $\mathsf{card}(\Lambda_m) \approx k$, which is less than 1000 (the recommended parameter is $k = 3$). Hence, this step is negligible in practice compared to the Gröbner basis computation, since an overdetermined linear system with less than 1000 variables in a finite field can be easily solved. Furthermore, this step is analog to the linear system which is solved in the legal decryption algorithm. Therefore this step of the attack is faster than the decryption algorithm which has to be efficient for practical parameters.

The cost of an arithmetic operation in $\mathbb{K}$ is quasi-linear in $\log(\mathsf{card}(\mathbb{K})) \approx C$. The number of times we have to run the main loop of the attack is $size(m)/C$. The complexity of the CRT is $\widetilde{\mathcal{O}}(size(m)\log(size(m)))$ [18]. Putting all the steps together, we find the total complexity of the attack:

Theorem 1. *The total binary complexity of the Level 3 Attack is*

$$\underbrace{\widetilde{\mathcal{O}}(size(m)w^3)}_{resultant} + \underbrace{\widetilde{\mathcal{O}}(size(m)(w^4 + w^2C))}_{factorization} + \underbrace{\widetilde{\mathcal{O}}(size(m)w^6)}_{Gröbner} + \underbrace{\widetilde{\mathcal{O}}(size(m))}_{CRT}.$$

Hence, the total binary asymptotic complexity of the attack is upper bounded by

$$\widetilde{\mathcal{O}}(w^6 size(m)).$$

Corollary 1. *If we assume that $size(m) \approx wd\log(p)$ (which is the case in practice), then the binary complexity of the attack is: $\widetilde{\mathcal{O}}(dw^7\log(p))$.*

Consequently, the attack is polynomial in all the security parameters and it is quasi-linear in the size of the secret key which is $2d\log(p)$. It can be noted that the parameter k has few effect on the complexity of the attack.

A Lower Bound on the Complexity of the Decryption Algorithm
The complexity of this attack has to be compared with a lower bound on the cost of the decryption process. During the decryption algorithm, one has to factor $(F_0 - F_1)(u_x(t), u_y(t), t)$ over $\mathbb{F}_p[t]$. The degree of this polynomial is at least dw. To the best of our knowledge, the best probabilistic factorization algorithms have an arithmetic complexity of $\widetilde{\mathcal{O}}(d^2w^2 + dw\log(p))$ [18]. Moreover, there is also a knapsack to solve after the factorization. The complexity of this step is difficult to estimate so we do not consider it here (remember that we try to establish a lower bound). The last step of the decryption process is the resolution of a linear system with $\mathcal{O}(dw)$ variables: the arithmetic complexity of this step is $\mathcal{O}(w^3d^3)$. Finally, the total binary complexity of the decryption algorithm is unsharply lower bounded by $\widetilde{\mathcal{O}}(\log(p)(w^3d^3 + dw\log(p)))$ which is cubic in the parameters d and w, and quadratic in $\log(p)$. In comparison, the attack is quasi-linear in d and $\log(p)$, and polynomial of degree 7 in w.

6 Experimental Results

Workstation. The experimental results have been obtained with a Xeon bi-processor 3.2 GHz, with 64 GB of RAM. The instances of ASC have been generated with MAGMA2.15-7. To compute the Gröbner basis, we use the F_4 [7] implementation in MAGMA.

To generate our instances, we pick $\ell, d \in \mathbb{N}$ and we consider the following parameters:

– $w = 2\ell + 5$.
– $\Lambda_m = \{(4+\ell, 4+\ell), (0,0)\}$.
– $\Lambda_X = \{(3+\ell, 2+\ell), (1+\ell, 2+\ell), (0,0)\}$.
– $\Lambda_f = \{(5+\ell, 5+\ell), (1+\ell, 2+\ell), (1,2), (0,0)\}$.
– $\forall (i,j) \in \Lambda_m, d_{ij}^{(m)} = (2\ell+5)d + 21$.
– $\forall (i,j) \in \Lambda_m, d_{ij}^{(f)} = (2\ell+5)d + 22$.

Construction of X, u_x and u_y
$u_x, u_y \in \mathbb{F}_p[t]$ are random polynomials of degree d.

To construct $X(x,y,t)$, we pick two random polynomials $R_1, R_2 \in \mathbb{F}_p[t]$ of degree 20 and we consider

$$X = R_1(t)(x^{3+\ell}y^{2+\ell} - u_x(t)^{3+\ell}u_y(t)^{2+\ell}) + R_2(t)(x^{1+\ell}y^{2+\ell} - u_x(t)^{1+\ell}u_y(t)^{2+\ell}).$$

Then we verify that $X(x,y,t)$ is irreducible. If not, we restart by picking another R_1 and another R_2.

Table 1 shows the complexity of the Level 3 Attack for different values of p and d. Each entry in the table is obtained by considering the average results over 20 random instances of the cryptosystem.

Table Notations
t_{res} denotes the time used for the computation of the resultant. t_{fact} is the time used by the factorization of the resultant, whereas t_{GB} denotes the cost of the Gröbner basis computation. The time for solving the linear system and for the recombination by the CRT is negligible and hence are not given in the table. According to [3], there were no known attack better than exhaustive search when $d \geq 50$ and $w \geq 5$. Therefore the security bound is the cost of the exhaustive search of the secret section, namely p^{2d+2}.

Interpretation of the Results
It is worth remarking that the first line of Table 1 corresponds to the parameters recommended by the designers [3] and are broken in 0.05 seconds. The major observation is that the complexity of the attack behaves as predicted by the complexity analysis: it is quasi-linear in the parameter d. We also ran some experiments to study the impact of the parameter k (the cardinality of the support of the surface X) on the complexity: as expected, increasing k has very few effect on the cost of the attack. To summarize, we see in Table 1 that trying to secure the system by increasing the size of the secret key (that is to say by increasing the parameters p and d) is pointless: even when the size of the secret key is bigger than 10000 bits, the system can be broken in few seconds.

Table 1. Level 3 Attack – Experimental results with $w = 5$

p	d	w	k	size of public key	size of secret key	t_{res}	t_{fact}	t_{GB}	t_{total}	security bound
2	50	5	3	310 bits	102 bits	0.02s	0.02s	0.01s	0.05s	2^{102}
2	100	5	3	560 bits	202 bits	0.03s	0.02s	0.02s	0.07s	2^{202}
2	200	5	3	1060 bits	402 bits	0.05s	0.05s	0.05s	0.15s	2^{402}
2	400	5	3	2060 bits	802 bits	0.1s	0.1s	0.1s	0.30s	2^{802}
2	800	5	3	4060 bits	1602 bits	0.2s	0.2s	0.2s	0.65s	2^{1602}
2	1600	5	3	8060 bits	3202 bits	0.3s	0.3s	0.4s	1.0s	2^{3202}
2	2000	5	3	10060 bits	4002 bits	0.45s	0.4s	0.4s	1.3s	2^{4002}
2	5000	5	3	25060 bits	10002 bits	0.8s	1.3s	0.8s	3.0s	2^{10002}
17	50	5	3	1267 bits	409 bits	0.2s	2.4s	0.4s	3.0s	2^{409}
17	100	5	3	2289 bits	818 bits	0.3s	5.1s	0.6s	3.0s	2^{818}
17	400	5	3	8420 bits	3270 bits	1.45s	27.7s	3.9s	33.1s	2^{3270}
17	800	5	3	16595 bits	6500 bits	3.1s	70s	9.5s	83s	2^{6500}
10007	500	5	3	34019 bits	13289 bits	29s	217s	64s	310s	2^{13289}

The Parameter w

In order to secure the system, one can think of increasing the parameter w since the attack is in $\mathcal{O}(w^7)$. However, we showed that the complexity decryption algorithm is lower bounded by $\mathcal{O}(w^3)$. Consequently, the parameter w should not be too high if the owner of the secret key wants to be able to decrypt. Table 2 gives the experimental results of the attack when w increases.

Interpretation of the Results

The main observation is that the complexity of the attack still behaves as predicted: when w is increased, the Gröbner basis computation is the most expensive step. Increasing w seems to be the best counter-measure against the attack. However, it should be noted that the attack is still feasible in practice, even when the public key is big.

Table 2. Level 3 Attack – Experimental results: increasing w

p	d	w	k	size of public key	size of secret key	t_{res}	t_{fact}	t_{GB}	t_{LinSys}	t_{total}	security bound
2	50	5	3	310 bits	102 bits	0.02s	0.02s	0.01s	0.001s	0.05s	2^{102}
2	50	15	3	810 bits	102 bits	0.7s	0.3s	4.4s	0.03s	5.4s	2^{102}
2	50	25	3	1310 bits	102 bits	3s	1s	32s	0.2s	37s	2^{102}
2	50	35	3	1810 bits	102 bits	10s	3s	260s	1s	274s	2^{102}
2	50	45	3	2310 bits	102 bits	30s	7s	1352s	4s	1393s	2^{102}
2	50	55	3	2810 bits	102 bits	70s	12s	4619s	13s	4714s	2^{102}
2	50	65	3	3310 bits	102 bits	147s	22s	12408s	27s	12604s	2^{102}
2	50	75	3	3810 bits	102 bits	288s	38s	37900s	56s	38280s	2^{102}

7 Conclusion

In this paper, we analyze the security of the PKC'2009 Algebraic Surface Cryptosystem. We provide three variants of a message recovery attack. We also estimate very precisely the complexity of the Level 3 Attack and we show that it is polynomial in all the parameters of the system. Furthermore, it is quasi-linear in the size of the secret key, whereas the decryption algorithm proposed in [3] is cubic.

Experimental results confirm the theoretical analysis. We show that the attack can easily break ASC with recommended parameters. The best choice to try to secure ASC against the attack is to take p and d as small as possible ($p = 2$ and $d = 50$) and increase w. However our implementation is polynomial in w and can break the system in few hours, even when $w = 75$ (this value should be compared to the initial recommended $w = 5$).

Thereby, we consider that the system is fully broken, but we believe that the section finding problem is still an interesting problem; in this paper, we have simply shown how to avoid to solve it in the context of ASC.

Acknowledgements. We wish to thank the anonymous referees for their helpful comments and suggestions. We are also thankful to Maki Iwami for useful discussions.

References

1. Adams, W.W., Loustaunau, P.: An introduction to Gröbner bases. American Mathematical Society (1994)
2. Akiyama, K., Goto, Y.: An Algebraic Surface Public-key Cryptosystem. IEIC Technical Report (Institute of Electronics, Information and Communication Engineers) 104(421), 13–20 (2004)
3. Akiyama, K., Goto, Y., Miyake, H.: An Algebraic Surface Cryptosystem. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, p. 442. Springer, Heidelberg (2009)
4. Bardet, M., Faugere, J.-C., Salvy, B.: On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In: Proceedings of the International Conference on Polynomial System Solving, pp. 71–74 (2004)
5. Bardet, M., Faugere, J.-C., Salvy, B., Yang, B.-Y.: Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. In: Proceedings of the Eight International Symposium on Effective Methods in Algebraic Geometry, MEGA (2005)
6. Cox, D.A., Little, J.B., O'Shea, D.: Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra. Springer, Heidelberg (1997)
7. Faugère, J.-C.: A new efficient algorithm for computing Gröbner bases (F4). Journal of Pure and Applied Algebra 139, 61–88 (1999)
8. Faugère, J.-C.: A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In: Proceedings of the 2002 international symposium on symbolic and algebraic computation, pp. 75–83. ACM, New York (2002)

9. Faugère, J.-C., Gianni, P., Lazard, D., Mora, T.: Efficient computation of zero-dimensional Gröbner bases by change of ordering. Journal of Symbolic Computation 16(4), 329–344 (1993)
10. Garey, M.R., Johnson, D.S., et al.: Computers and Intractability: A Guide to the Theory of NP-completeness. W.H. Freeman, San Francisco (1979)
11. Ivanov, P., Voloch, J.F.: Breaking the Akiyama-Goto cryptosystem. Arithmetic, Geometry, Cryptography and Coding Theory 487 (2009)
12. Iwami, M.: A Reduction Attack on Algebraic Surface Public-Key Cryptosystems. In: Workshop of Research Institute for Mathematical Sciences (RIMS) Kyoto University, New development of research on Computer Algebra, RIMS Kokyuroku, vol. 1572. Springer, Heidelberg (2007)
13. Lazard, D.: Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In: EUROCAL, vol. 162, pp. 146–156. Springer, Heidelberg (1983)
14. Lecerf, G.: New recombination algorithms for bivariate polynomial factorization based on Hensel lifting. To appear in AAECC (2007)
15. Patarin, J.: Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 33–48. Springer, Heidelberg (1996)
16. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: SFCS 1994: Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Washington, DC, USA, pp. 124–134. IEEE Computer Society Press, Los Alamitos (1994)
17. Uchiyama, S., Tokunaga, H.: On the Security of the Algebraic Surface Public-key Cryptosystems. In: Proceedings of SCIS (2007)
18. Von Zur Gathen, J., Gerhard, J.: Modern computer algebra. Cambridge University Press, Cambridge (2003)

A Gröbner Bases and Normal Form

In this section, we shortly describe some fundamental tools from commutative algebra, which are useful for the attack presented in this paper. For a more complete presentation of those tools, the reader can refer to [6,1].

From now on, $\mathbb{K}$ is a field and R denotes the ring $\mathbb{K}[x_1, \ldots, x_n]$. We suppose given an admissible monomial ordering $<$: for the attack we consider the *grevlex* (graded reverse lexicographical) ordering.

Definition 1 (Grevlex ordering). *The* grevlex *ordering is defined as follows. Let* $m_1 = x_1^{\alpha_1} \ldots x_n^{\alpha_n}, m_2 = x_1^{\beta_1} \ldots x_n^{\beta_n}$ *be two monomials. Then* $m_1 > m_2$ *if*

- $\sum_{i=1}^{n} \alpha_i > \sum_{i=1}^{n} \beta_i$ *or*
- $\sum_{i=1}^{h} \alpha_i = \sum_{i=1}^{h} \beta_i$ *and the rightmost nonzero entry of* $(\alpha_1 - \beta_1, \ldots, \alpha_n - \beta_n)$ *is negative.*

For any polynomial $P \in R$, $LM(P)$ denotes its leading monomial with respect to $<$. For any ideal $I \subset R$, $LM(I)$ denotes the ideal generated by $\langle\{LM(P) : P \in I\}\rangle$.

Definition 2 (Normal form). *Let I be an ideal of R, and $f \in R$ be a polynomial. Then there exist unique polynomials $h, g \in R$ such that h is monic, $g \in I$, $f = h + g$ and no monomial of h is in $LM(I)$. Then h is called the* normal form *of f with respect to I and $<$, and is noted $NF_I(f)$.*

The normal form is a $\mathbb{K}$-linear application and its main property is:

Proposition 1. *Let I be an ideal of R, and $f \in R$ be a polynomial. Then $f \in I$ if and only if $NF_I(f) = 0$.*

To be able to compute the normal form, we need another fundamental tool: Gröbner bases.

Definition 3 (Gröbner basis). *Let I be an ideal of R. A finite subset of polynomials $G \subset I$ is called a* Gröbner basis *of I (with respect to the monomial ordering $<$) if $\langle LM(G)\rangle = LM(I)$.*

B Magma Code for the Level 1 Attack

In the following piece of code, `p` and `d` are the parameters of the system. `deg_t` is the degree of m with respect to t and `Lambda_m` denotes the support of m (these values are public). `F0` and `F1` are the ciphertext, and `X` is the public surface.

```
R<x,y,t>:=PolynomialRing(GF(p),3,"grevlex");
Res:=Resultant(R!(F0-F1),R!X,x);  // Eliminate x
F:=Factorization(Res); // Factor the resultant
// Pick the irreducible factor of highest degree in y
maxdeg:=Max([Degree(R!f[1],R!y) : f in F]);
exists(Q0){f[1]:f in F| Degree(R!f[1],R!y) eq maxdeg};
J:=Ideal([R!Q0,R!X,R!F0,R!F1]);
Groebner(J);  // Compute the Gröbner basis of J
Coeffm:=PolynomialRing(GF(p),#Lambda_m*(deg_t+1));
R2<x,y,t>:=PolynomialRing(Coeffm,3);
// Construct the linear system
plaintext:=&+[Coeffm.((i-1)*(deg_t+1)+j)*
              R2!NormalForm(R!x^Lambda_m[i][1]*
              R!y^Lambda_m[i][2]*R!t^(j-1),J) :
              i in [1..#Lambda_m], j in [1..deg_t+1]];
// Solve the linear system:
V:=Variety(Ideal(Coefficients(plaintext)));
```

Maximizing Small Root Bounds by Linearization and Applications to Small Secret Exponent RSA

Mathias Herrmann and Alexander May[⋆]

Horst Görtz Institute for IT-Security
Faculty of Mathematics
Ruhr University Bochum, Germany
mathias.herrmann@rub.de, alex.may@rub.de

Abstract. We present an elementary method to construct optimized lattices that are used for finding small roots of polynomial equations. Former methods first construct some large lattice in a generic way from a polynomial f and then optimize via finding suitable smaller dimensional sublattices. In contrast, our method focuses on optimizing f first which then directly leads to an optimized small dimensional lattice.

Using our method, we construct the first elementary proof of the Boneh-Durfee attack for small RSA secret exponents with $d \leq N^{0.292}$. Moreover, we identify a sublattice structure behind the Jochemsz-May attack for small CRT-RSA exponents $d_p, d_q \leq N^{0.073}$. Unfortunately, in contrast to the Boneh-Durfee attack, for the Jochemsz-May attack the sublattice does not help to improve the bound asymptotically. Instead, we are able to attack much larger values of d_p, d_q in practice by LLL reducing smaller dimensional lattices.

Keywords: linearization, lattices, small roots, small secret exponent, RSA, CRT-RSA.

1 Introduction

The RSA cryptosystem is currently the most widely deployed cryptosystem. To perform a decryption or signature generation, an element $x \in \mathbb{Z}_N$ is raised to the d-th power, where $d \in \mathbb{Z}^*_{\phi(N)}$ is the secret key. In order to speed up this process, one might be tempted to use a small value of d. However, once $d \leq N^{\frac{1}{4}}$, Wiener [Wie90] showed using a continued fraction approach that d can be reconstructed from just the public parameters e and N in polynomial time. This result has been further improved by Boneh and Durfee to $d \leq N^{0.292}$ using a lattice based technique [BD99].

Another possibility to speed up the decryption and signature generation has been proposed by Quisquater and Couvreur [QC82]. They make use of the knowledge of the prime factorization of $N = pq$ to compute x^d modulo p and modulo q

[⋆] This research was supported by the German Research Foundation (DFG) as part of the project MA 2536/3-1 and by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II.

and finally combine the result using the Chinese Remainder Theorem. The running time of this process is approx. 4 times faster than a standard decryption. To further lower the number of required operations, one can additionally use *small CRT exponents*, i.e. one can choose d such that $d_p = d \bmod p$ and $d_q = d \bmod q$ are both small.

At Crypto '07, Jochemsz and May [JM07] proposed the first polynomial time attack on CRT exponents that are smaller than $N^{0.073}$. However, the experimental results of Jochemsz and May for small dimensional lattices are much better than theoretically predicted. For example, using a lattice dimension of 56, theoretically the attack should not work at all, while in practice this lattice dimension is sufficient to reconstruct private keys up to a size of $N^{0.01}$. Such a discrepancy between theoretically predicted and practically achieved results is a strong indication that the involved lattice structure is not optimal. This led Jochemsz and May to conjecture that an analysis of sublattice structures could lead to a theoretically superior bound.

In this paper we propose a method that can be applied to attack small CRT-exponents. Our new approach leads to smaller dimensional lattices than in the Jochemsz-May attack and fully explains the gap between the practical results of Jochemsz and May and their theoretical analysis. Unfortunately, our analysis shows that our smaller dimensional lattices asymptotically lead to the same bound $N^{0.073}$ as in [JM07], thereby answering the conjecture of Jochemsz and May that sublattices improve the bound in the negative.

Although we do not achieve an asymptotic improvement, our new approach enables us to attack much larger values of d_p, d_q in practice, compared to [JM07], by using smaller dimensional lattices. We implemented our algorithm and showed that e.g. for a 2000-bit N we can efficiently recover 47-bit d_p, d_q, whereas the technique of [JM07] only allows to recover about 35-bit d_p, d_q in a comparable amount of time.

Our method is lattice-based and uses the technique of *unravelled linearization* introduced by Herrmann and May at Asiacrypt '09 [HM09], which can be seen as a hybrid method between usual linearization and Coppersmith's method [Cop97]. The central idea of unravelled linearization is to perform as a first step a linearization on the initial polynomial and keep the *induced relations* of the linearization in mind. These relations are afterwards used in a second step where we back-substitute in order to eliminate some monomials, thereby partially unravelling the first linearization step. In order to explicitly compute the induced relations, we propose to use a Gröbner basis computation.

We illustrate the technique of unravelled linearization by showing the first elementary proof of the Boneh-Durfee bound $d \leq N^{0.292}$ for small secret RSA exponents. Optimization of bounds is in our framework a simple task. Therefore, we conjecture that the Boneh-Durfee bound cannot be improved unless a different polynomial equation is used.

The rest of the paper is organized as follows: In Section 2 we will review some basic results from lattice theory. Section 3 will describe the method of unravelled linearization for the case of small RSA exponents d with a proof of $d \leq N^{0.292}$.

We will then apply our method to attack small CRT exponents in Section 4, where we achieve the Jochemsz-May bound of $N^{0.073}$ with smaller dimensional lattices. In Section 5, we demonstrate that our improved lattices allow for much better practical results in attacking small CRT-exponents.

2 Basics

Before we explain the details of unravelled linearization and how to use it to improve the analysis of small CRT-exponents, we want to give some necessary background information on lattice theory and the lattice-based method of Coppersmith [Cop97].

A lattice is a discrete additive subgroup of $\mathbb{R}^n$. That is, for a set of linearly independent basis vectors $b_1, \dots, b_{dim} \in \mathbb{R}^n$, $dim \leq n$, the set

$$L := \left\{ x \in \mathbb{R}^n \mid x = \sum_{i=0}^{dim} a_i b_i \text{ with } a_i \in \mathbb{Z} \right\}$$

is called a lattice. One can describe a lattice by its basis matrix B, where we write the vectors b_i as row vectors.

Let L be a lattice with basis $b_1, \dots, b_{dim}$, and let $b_1^*, \dots, b_{dim}^*$ be the result of applying Gram-Schmidt orthogonalization to the basis vectors. Then the determinant of L is defined as $\det(L) = \prod_{i=1}^{dim} \|b_i^*\|$. For a lattice of full rank, i.e. $dim = n$, the determinant of a lattice equals the absolute value of the determinant of a lattice basis matrix.

Lattices have proved to be very useful in cryptanalysis mostly because of a powerful and efficient lattice reduction algorithm due to Lenstra, Lenstra and Lovász [LLL82]. This so-called LLL algorithm outputs an approximation of a shortest lattice vector in time polynomial in the bit-length of the entries of the basis matrix and in the dimension of the lattice *dim*. Using the LLL algorithm as a building block, Coppersmith [Cop96a, Cop96b] designed a rigorous algorithm that allows to efficiently compute *small* roots of bivariate polynomials over the integers or univariate modular polynomials. Additionally, he gave a heuristic extension to multivariate polynomials.

Coppersmith's idea is to construct, on input some polynomial f, a set of coprime polynomials which contain the same roots over the integers. Then one can use standard elimination and root finding techniques to extract these roots. Howgrave-Graham [HG97] gave a simple reformulation of Coppersmith's method that defines the following condition.

Theorem 1 (Howgrave-Graham). *Let $g(x_1, \dots, x_k)$ be a polynomial in k variables with n monomials. Furthermore, let m be a positive integer. Suppose that*

1. $g(r_1, \dots, r_k) = 0 \bmod b^m$*, where* $|r_i| \leq X_i, i = 1, \dots, k$ *and*
2. $||g(x_1X_1, \dots, x_kX_k)|| \leq \frac{b^m}{\sqrt{n}}$,

where the norm of g is defined as the Euclidean norm of its coefficient vector.

Then $g(r_1, \dots, r_k) = 0$ holds over the integers.

3 Unravelled Linearization and the Boneh-Durfee Attack

In this section, we will apply the method of unravelled linearization, introduced by Herrmann and May [HM09], to attack RSA with small secret exponent d. This will lead to an elementary proof of the Boneh-Durfee bound $d \leq N^{0.292}$.

In 1999, Boneh and Durfee [BD99] showed with a lattice-based Coppersmith-type attack, that private RSA keys smaller than $N^{0.284-\epsilon}$ can be recovered in polynomial time. The attack's running time is dominated by LLL-reducing some large dimensional lattice basis B, whose dimension depends on $\frac{1}{\epsilon}$. It turns out that the associated lattice $L(B)$ contains a smaller dimensional sublattice L' that allows to show an improved bound of $N^{0.292-\epsilon}$.

The identification and analysis of this sublattice L', however, is a complicated task due to the fact that its lattice basis is no longer triangular and, therefore, the computation of the lattice determinant $\det(L')$ is much more involved. Boneh and Durfee developed for the analysis of $\det(L')$ a notion called *geometrically progressive matrices* that allowed for handling these non-triangular lattice bases. Blömer and May [BM01] followed a different approach and showed that asymptotically it does not influence the determinant if some specific columns are removed. This allowed them to rebuild some triangular structure of the basis matrix. Both approaches are, however, quite complex methods for optimizing lattice bases.

As opposed to the methods of [BD99] and [BM01] our new approach will not manipulate a basis matrix but rather it will manipulate the underlying polynomial from which a basis matrix is derived. This will directly lead to a low-dimensional sublattice with a basis of triangular structure that allows for an easy determinant calculation.

The method of our choice for this task is the technique of unravelled linearization [HM09]. However, before we introduce our method we briefly recall the original Boneh-Durfee attack in order to illustrate the similarities and differences.

The polynomial to be analyzed is derived from the RSA key equation $ed = 1 \bmod \phi(N)$. Rewrite this as

$$\begin{aligned} ed &= 1 + x\phi(N) \\ \Leftrightarrow ed &= 1 + x(\underbrace{N+1}_{A} + \underbrace{(-p-q)}_{y}) \end{aligned}$$

and search for small modular roots of the polynomial

$$f(x,y) := 1 + x(A+y) \bmod e.$$

Therefore, we fix an integer m and define the polynomials

$$g_{i,k}(x,y) := x^i f^k e^{m-k} \quad \text{and} \quad h_{j,k}(x,y) := y^j f^k e^{m-k}.$$

A lattice basis is constructed by using the coefficient vectors of the so-called x-shifts $g_{i,k}(xX, yY)$ for $k = 0, \ldots, m$ and $i = 0, \ldots, m-k$ as basis vectors.

$$\begin{array}{c|ccccccccc}
 & 1 & x & xy & x^2 & x^2y & x^2y^2 & y & xy^2 & x^2y^3 \\
e^2 & e^2 & & & & & & & & \\
xe^2 & & e^2X & & & & & & & \\
fe & e & eAX & eXY & & & & & & \\
x^2e^2 & & & & e^2X^2 & & & & & \\
xfe & & eX & & eAX^2 & eX^2Y & & & & \\
f^2 & 1 & 2AX & 2XY & A^2X^2 & 2AX^2Y & X^2Y^2 & & & \\
\hline
\boldsymbol{ye^2} & & & & & & & e^2Y & & \\
\boldsymbol{yfe} & & & eAXY & & & & eY & eXY^2 & \\
\hline
yf^2 & & & 2AXY & & A^2X^2Y & 2AX^2Y^2 & Y & 2XY^2 & X^2Y^3
\end{array}$$

Fig. 1. Boneh-Durfee basis matrix for $m = 2, t = 1$

The values X and Y denote upper bounds on the sizes of the solutions. Additionally, we use the so-called y-shifts $h_{j,k}(xX, yY)$ for $k = 0, \ldots, m$ and $j = 1, \ldots, t$, where t is some parameter that has to be optimized. Figure 1 shows an example for the parameters $m = 2$ and $t = 1$. Note that the coefficient vectors of the shift polynomials $g_{i,k}(xX, yY)$ and $h_{j,k}(xX, yY)$ are written as row vectors.

Boneh and Durfee's improved analysis showed that one obtains superior values for X and Y, if one takes only a subset of the y-shifts. For our example this means we exclude ye^2 and yfe. Hence, the resulting lattice basis is no longer triangular and, therefore, deriving a closed determinant formula for general m and t is a complex task.

We now use the technique of unravelled linearization to construct a lattice basis which yields the best known asymptotic bound $N^{0.292}$ and yet retains a triangular lattice basis.

The first step in the process is to perform a suitable linearization of the original polynomial. In our case, we glue together the monomials in the following way

$$\underbrace{1 + xy}_{u} + Ax \bmod e.$$

This leaves us with the linear polynomial $\bar{f}(u, x) = u + Ax$ and additionally a *relation* $xy = u - 1$ derived from the substitution. Although Coppersmith's method is a construction method suited for polynomial equations and does not give improved bounds in the case of linear equations, we now construct a lattice basis using exactly the same x-shifts as in the original Boneh-Durfee attack. I.e., we construct polynomials

$$\bar{g}_{i,k}(u, x) := x^i \bar{f}^k e^{m-k} \quad \text{for} \quad k = 0, \ldots, m \text{ and } i = 0, \ldots, m - k, \qquad (1)$$

and use their coefficient vectors as basis vectors. One can show that this leads to the Wiener bound of $N^{0.25}$.

However, if we also include y-shifts of the form $\bar{h}_{j,k}(u, x, y) := y^j \bar{f}^k e^{m-k}$, then we obtain a benefit. This may sound strange at first glance since the monomial y is not even present in our new polynomial $\bar{f}(u, x)$. The reason for the improved

$$\begin{array}{c|ccccccc}
 & 1 & x & u & x^2 & ux & u^2 & u^2y \\
e^2 & e^2 & & & & & & \\
xe^2 & & e^2X & & & & & \\
\bar{f}e & & eAX & eU & & & & \\
x^2e^2 & & & & e^2X^2 & & & \\
x\bar{f}e & & & & eAX^2 & eUX & & \\
\bar{f}^2 & & & & A^2X^2 & 2AUX & U^2 & \\
\hline
y\bar{f}^2 & & -A^2X & -2AU & & A^2UX & 2AU^2 & U^2Y
\end{array}$$

Fig. 2. Boneh-Durfee lattice for $m = 2, t = 1$ using unravelled linearization

bound becomes clear, when we incorporate the induced relation $xy = u - 1$ and use it to substitute each occurrence of xy by the term $u - 1$.

The advantage can be seen by comparing the shift yf^2 from the original analysis with the new shift $y\bar{f}^2$. As noted previously, the improved analysis uses only the shift yf^2 and neither yfe nor ye^2. But yf^2 introduces three new monomials y, xy^2 and x^2y^3 in the Boneh-Durfee lattice basis – thereby destroying the triangular structure.

Let us compare this with our new unravelled linearization approach, which we depicted in Figure 2 for the same parameters $m = 2$ and $t = 1$. The shift $y\bar{f}^2$ introduces the monomials x^2y, uxy and u^2y. We replace each occurrence of xy by $u - 1$, i.e., we replace x^2y by $ux - x$ and uxy by $u^2 - u$. But the monomials ux, x, u^2 and u are already present in the lattice bases. Thus, the only new monomial that comes from the shift $y\bar{f}^2$ is u^2y, thereby retaining the triangular structure.

In order to keep the triangular structure in general, we look at an arbitrary shift $y^i\bar{f}^\ell$. Notice that for the ease of notation we will omit the factor $e^{m-\ell}$ as it does not influence the set of monomials. Since $\bar{f} = u + Ax$ we can expand $y^i\bar{f}^\ell$ by the binomial theorem

$$u^\ell y^i + \binom{\ell}{1} A u^{\ell-1} x y^i + \ldots + \binom{\ell}{\ell} A^\ell x^\ell y^i.$$

The first term introduces a new monomial $u^\ell y^i$. However, we will now derive a certain restriction under which all other monomials are already present in the lattice basis. Let us therefore look at the monomials of the second term after the substitution of xy

$$u^{\ell-1}xy^i = u^{\ell-1}(u-1)y^{i-1} = u^\ell y^{i-1} - u^{\ell-1}y^{i-1}.$$

The monomials $u^\ell y^{i-1}$ and $u^{\ell-1}y^{i-1}$ appear in $y^{i-1}\bar{f}^\ell$ and $y^{i-1}\bar{f}^{\ell-1}$, respectively. In general, the $(j+1)^{th}$ term of the binomial expansion contains monomials that appear in $y^{i-j}\bar{f}^{\ell-k}$ for $k = 0, \ldots, j$.

Therefore, the shift $y^i\bar{f}^\ell$ introduces exactly one new monomial $u^\ell y^i$ if all shifts $y^{i-j}\bar{f}^{\ell-k}$ for $j = 1, \ldots, i-1$ and $k = 0, \ldots j$ were used in the construction of the lattice basis. This is exactly to the restriction that was called *increasing pattern* in [BM01].

Since the y-shifts $h_{j,k}$ in the original Boneh-Durfee attack satisfy this *increasing pattern* restriction as shown in [BM01], we take in our analysis the y-shifts $\bar{h}_{j,k}$ for the same set of indices (j,k) as in [BD99]. I.e., we define the y-shifts

$$\bar{h}_{j,k} = y^j \bar{f}^k e^{m-k} \text{ for } j = 1, \ldots, t \text{ and } k = \left\lfloor \frac{m}{t} \right\rfloor j, \ldots, m. \tag{2}$$

We show that this set of y-shifts $\bar{h}_{j,k}$ satisfies our requirement, i.e. we show that if $y^i \bar{f}^\ell$ is a y-shift, then all of $y^{i-j} \bar{f}^{\ell-k}$ for $j = 1, \ldots, i-1$ and $k = 0, \ldots, j$ are also used as shifts. Notice that it is sufficient to show $y^{i-j} \bar{f}^{\ell-j}$ is used as a shift.

Since $y^i \bar{f}^\ell$ is in the set of y-shifts, we know that $\ell \in \{\lfloor \frac{m}{t} \rfloor i, \ldots, m\}$ and therefore $\ell - j \in \{\lfloor \frac{m}{t} \rfloor i - j, \ldots, m - j\}$. For $y^{i-j} \bar{f}^{\ell-j}$ on the other hand, we have $\ell - j \in \{\lfloor \frac{m}{t} \rfloor (i-j), \ldots, m\}$. Our requirement is thus fulfilled if the condition

$$\left\lfloor \frac{m}{t} \right\rfloor (i-j) \leq \left\lfloor \frac{m}{t} \right\rfloor i - j$$

holds. We can rewrite this as $\lfloor \frac{m}{t} \rfloor \geq 1$, which holds if $m \geq t$.

Given the set of shift polynomials, we proceed with the computation of the determinant. For the following asymptotic analysis we let $t = \tau m$. Further, for the optimization we omit roundings as their contribution is negligible for sufficiently large m.

We are able to directly compute the contributions of the shift polynomials from (1) and (2). Here, we denote by s_x the contribution of X to the determinant.

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

Using these values together with the upper bounds $X = N^\delta, Y = N^{\frac{1}{2}}, U = N^{\delta + \frac{1}{2}}$ on the variables in the usual enabling condition $\det(L) = X^{s_x} Y^{s_y} U^{s_u} e^{s_e} \leq e^{m \dim(L)}$, we obtain an optimized value of $\tau = (1 - 2\delta)$ and finally derive the desired Boneh-Durfee bound[1]

$$\delta \leq \frac{1}{2}\left(2 - \sqrt{2}\right) \approx 0.292.$$

[1] The given bound is for full size e, i.e. we set $e \approx N$.

Notice that our choice of τ fulfills our previous restriction $m \geq t$. To summarize, the method of unravelled linearization provides a simple and elegant way to capture the sublattice structure in the Boneh-Durfee attack. In the following section, we will use the same method to recover the hidden sublattice structure in the Jochemsz-May attack on small CRT-RSA exponents. This sublattice was previously unknown and was conjectured to be the key for improving the CRT-RSA attack bound.

4 CRT Exponents

The task of attacking small CRT exponents was first mentioned as an open problem in Wiener [Wie90]. At PKC '06, Bleichenbacher and May [BM06] gave an attack that worked in the case where e is significantly smaller than N. They started with the CRT-RSA equations $ed_p = 1 + k(p-1)$ and $ed_q = 1 + l(q-1)$, and derived a single polynomial in the unknowns (d_p, d_q, k, l) by setting $q = \frac{N}{p}$ and eliminating p:

$$e^2 d_p d_q - e(d_p + d_q) + e(d_q k + d_p l) - (k + l - 1) - (N-1)kl = 0. \tag{3}$$

This equation can be linearized to

$$e^2 x_1 + e x_2 - (N-1)x_3 - x_4 = 0 \tag{4}$$

with unknowns

$$x_1 = d_p d_q, x_2 = d_q k + d_p l - d_p - d_q, x_3 = kl, x_4 = (k + l - 1).$$

For $d_p, d_q \leq N^{\delta}$ we get $k, l \leq N^{\frac{1}{2}+\delta}$ and Eq. (4) directly leads to a lattice attack provided that $\delta \leq \min\{\frac{1}{4}, \frac{2}{5} - \frac{2}{5}\alpha\}$, where $\alpha = \log_N e$. However, for a full size e, i.e. $\alpha = 1$, this attack does not work.

In 2007, Jochemsz and May [JM07] improved the analysis by exploiting the full algebraic structure of Eq. (3) with a Coppersmith-type attack. For the case $\alpha = 1$, they showed that it is possible to find small solutions if $\delta \leq 0.073$. However, in their experiments they noticed a big gap between the theoretically predicted bound and the experimentally observed bound. Namely, the experiments were far better than theoretically expected indicating the possibility of a better bound.

E.g., using their analysis, a lattice dimension of 56 should not suffice for attacking small CRT-exponents, while practically it allows for solving up to $d_p, d_q \leq N^{0.01}$. Jochemsz and May reported that the smallest LLL vectors came from a sublattice and conjectured that identifying the sublattice structure would improve the bound – analogous to the case of the Boneh-Durfee attack where the sublattice lifts the bound from $N^{0.284}$ to $N^{0.292}$.

In this section, we show that this conjecture is false. By using the method of unravelled linearization, we will capture the sublattice structure behind the Jochemsz-May attack. This will completely explain the experimental behavior in [JM07] and therefore close the gap between practice and theoretical analysis.

As a result, we construct lattices of much smaller dimension than in [JM07], whose theoretical analysis exactly matches the experiments that we present in the subsequent section.

Very disappointingly from a cryptanalytic point of view, the size of the CRT-exponents d_p, d_q that we are able to attack in polynomial time converges for growing lattice dimension to the same bound $N^{0.073}$ as in [JM07]. Thus, asymptotically we are unable to improve on the bound although we fully exploit the sublattice structure. Nevertheless, we think that our method is of independent interest and will prove to be useful for other attacks since it is simple and leads to an easy analysis.

Let us describe the attack in detail. Starting point is the polynomial equation (3). We proceed similar to [BM06] and perform an (almost) identical linearization.

$$e^2 \underbrace{d_p d_q}_{u} - e \underbrace{(d_p + d_q)}_{v} + e \underbrace{(d_q k + d_p l)}_{w} - (\underbrace{k + l}_{x} - 1) - (N - 1) \underbrace{kl}_{y} = 0 \qquad (5)$$

We now use the method of unravelled linearization with the linear polynomial $f = e^2 u - ev + ew - x - Ay + 1$, where $A = N - 1$. The next step is to build up a lattice following the extended strategy from [JM06]. This means we use the monomials of f^{m-1} as shifts and furthermore include extrashifts in the variables u and v up to some parameter t which has to be optimized later.

The benefit in unravelled linearization comes from the fact that the variables u, v, w, x, y are related. Namely, we have

$$\begin{aligned} vwx &= (d_p + d_q)(d_p l + d_q k)(k + l) \\ &= d_p^2 kl + d_p^2 l^2 + d_p d_q (k + l)^2 + d_q^2 k^2 + d_q^2 kl \\ &= (d_p^2 + d_q^2) kl + (d_p^2 l^2 + d_q^2 k^2) + d_p d_q (k + l)^2 \\ &= ((d_p + d_q)^2 - 2 d_p d_q) kl + ((d_p l + d_q k)^2 - 2 d_p d_q kl) + d_p d_q (k + l)^2 \\ &= (v^2 - 2u) y + w^2 - 2uy + ux^2. \end{aligned} \qquad (6)$$

This non-obvious relation can be computed easily using a Gröbner basis computation. Recall the equations given by the linearization. These are 5 linearization equations in 9 unknowns, so we can eliminate via Gröbner basis computation the four variables d_p, d_q, k, l and obtain Eq. (6) in the unknowns u, v, w, x, y only. This equation now serves in the back-substitution step of unravelled linearization, where we replace each occurrence of vwx by the monomials $v^2 y, uy, w^2$ and ux^2.

To exemplify our method, we use the parameters $m = 2$ and $t = 1$. This is the smallest choice where Jochemsz and May [JM07] found positive experimental results. In the framework of unravelled linearization, it is obvious why we do not obtain a positive result for smaller parameters. In order to improve upon the bound from Bleichenbacher, May [BM06], we have to use relation (6). However, the lattice parameters $m = 2$ and $t = 1$ are the smallest ones for which the monomial vwx appears.

A lattice basis B for $(m,t) = (2,1)$ is given in Figure 3. We use here the notation from the original Coppersmith method over the integers – as opposed to the modular approach taken in Section 3. That is, we construct a lattice basis with the coefficient vectors of the shift polynomials as column vectors (refer to [Cop97] for details). For simplicity we omit the left hand side of the basis matrix, which contains just the inverses of the corresponding upper bounds of the monomials on its diagonal. The entries that come from the substitution are printed in bold letters.

For the lattice attack to work, we require the enabling condition $\det(L) > 1$ (see [Cop97]). In our example, computation of the determinant of the basis matrix yields

$$\begin{array}{c|ccccccccccccccc}
 & f & uf & vf & xf & wf & yf & u^2f & uvf & uwf & uxf & yuf & v^2f & vwf & vxf & yvf \\
\hline
u & e^2 & 1 & & & & & & & & & & & & & \\
v & e & & 1 & & & & & & & & & & & & \\
w & e & & & & 1 & & & & & & & & & & \\
x & 1 & & & 1 & & & & & & & & & & & \\
1 & 1 & & & & & & & & & & & & & & \\
u^2 & & e^2 & & & & & 1 & & & & & & & & \\
uv & & e & e^2 & & & & & 1 & & & & & & & \\
uw & & e & & & e^2 & & & & 1 & & & & & & \\
ux & & 1 & & e^2 & & & & & & 1 & & & & & \\
v^2 & & & e & & & & & & & & & 1 & & & \\
vw & & & e & & e & & & & & & & & 1 & & \\
vx & & & 1 & e & & & & & & & & & & 1 & \\
wx & & & & e & 1 & & & & & & & & & & \\
x^2 & & & & 1 & & & & & & & & & & & \\
w^2 & & & & & e & & & & & & & & \mathbf{1} & \mathbf{e} & \\
u^3 & & & & & & & e^2 & & & & & & & & \\
u^2v & & & & & & & e & e^2 & & & & & & & \\
u^2w & & & & & & & e & & e^2 & & & & & & \\
u^2x & & & & & & & 1 & & & e^2 & & & & & \\
uv^2 & & & & & & & & e & & & & e^2 & & & \\
uvw & & & & & & & & e & e & & & & e^2 & & \\
uvx & & & & & & & & 1 & & e & & & & e^2 & \\
uw^2 & & & & & & & & & e & & & & & & \\
uwx & & & & & & & & & 1 & e & & & & & \\
ux^2 & & & & & & & & & & 1 & & & \mathbf{1} & \mathbf{e} & \\
v^3 & & & & & & & & & & & & e & & & \\
v^2w & & & & & & & & & & & & e & e & & \\
v^2x & & & & & & & & & & & & 1 & & e & \\
vw^2 & & & & & & & & & & & & & e & & \\
vx^2 & & & & & & & & & & & & & & 1 & \\
\hline
y & A & & & & & 1 & & & & & & & & & \\
yu & & A & & & & e^2 & & & & & 1 & & \mathbf{-4} & \mathbf{-4e} & \\
yv & & & A & & & e & & & & & & & & & 1 \\
yx & & & & A & & 1 & & & & & & & & & \\
yw & & & & & A & e & & & & & & & & & \\
y^2 & & & & & & A & & & & & & & & & \\
yu^2 & & & & & & & A & & & & e^2 & & & & \\
yuv & & & & & & & & A & & & e & & & & e^2 \\
yuw & & & & & & & & & A & & e & & & & \\
yux & & & & & & & & & & A & 1 & & & & \\
y^2u & & & & & & & & & & & A & & & & \\
yv^2 & & & & & & & & & & & & A & \mathbf{1} & \mathbf{e} & e \\
yvw & & & & & & & & & & & & & A & & e \\
yvx & & & & & & & & & & & & & & A & 1 \\
y^2v & & & & & & & & & & & & & & & A
\end{array}$$

Fig. 3. Matrix of unravelled linearized polynomial for $m = 2$, $t = 1$

$$\det(B) = U^{-21}V^{-20}W^{-14}X^{-14}A^{15}.$$

We have upper bounds $(U, V, W, X) = (N^{2\delta}, N^{\delta}, N^{\frac{1}{2}+2\delta}, N^{\frac{1}{2}+\delta})$ for the unknowns $(d_pd_q, d_p + d_q, d_qk + d_pl, k + l)$, respectively. Thus, with $A \approx N$ the enabling condition $\det(L) > 1$ reduces to $\delta < \frac{1}{104} \approx 0.01$. This perfectly matches the experimental results of Jochemsz and May for parameters $(m, t) = (2, 1)$.

We now proceed to the asymptotic analysis and start by analyzing the simpler case without any extrashifts. I.e., we shift in the monomials of f^{m-1} only, but we have to exclude all monomials that are divisible by vwx, since these can be written as the linear combination from Eq. (6).

To compute the value of the determinant we begin by counting the number of shift polynomials as each one contributes with a factor of A to the determinant. The number of shift polynomials equals the number of monomials in the set

$$\left\{u^{e_1}v^{e_2}w^{e_3}x^{e_4}y^{e_5} \mid e_i \in \mathbb{N}_0, \sum_{i=1}^{5} e_i \le m-1, e_2 = 0 \text{ or } e_3 = 0 \text{ or } e_4 = 0\right\}.$$

Their number can be computed as

$$\left|\left\{(e_1,\dots,e_5) \in \mathbb{N}_0^5 \mid \sum_{i=1}^{5} e_i \le m-1\right\}\right| - \left|\left\{(e_1,\dots,e_5) \in \mathbb{N}_0^5 \mid \sum_{i=1}^{5} e_i \le m-1, e_2, e_3, e_4 \ge 1\right\}\right|.$$

Let us derive the size of the first set by counting. Write $e_1+e_2+e_3+e_4+e_5+h = m-1$ for some slack variable $h \in \{0,\dots,m-1\}$ to transform the inequality into an equality. If we set $e_i' = e_i + 1$ and $h' = h + 1$ then the number of tuples that fulfill the equation

$$e_1' + e_2' + e_3' + e_4' + e_5' + h' = (m-1) + 6 \qquad \text{with } e_i', h_i' \ge 1$$

is exactly the number of ordered partitions of $m+5$ in 6 partitions. Let us write $m + 5 = 1 + 1 + \ldots + 1$, then one obtains an ordered 6-partition of $m + 5$ by choosing 5 out of the $m+4$ signs as breakpoints for the partition. We have $\binom{m+4}{5}$ possibilities for this choice.

The size of the second set is derived in a similar fashion, where we require $e_1' + e_2 + e_3 + e_4 + e_5' + h' = m + 2$. In this case, the number of tuples is $\binom{m+1}{5}$. Summing up, we obtain for the number of shifts

$$\#\text{shifts} = \binom{m+4}{5} - \binom{m+1}{5} = \frac{1}{8}m^4 + o(m^4).$$

The second part contributing to the determinant comes from the monomials that occur in the lattice basis. This is the product of the diagonal entries in the submatrix on the left that has been omitted in Figure 3. As mentioned before, the diagonal entries consist of the inverses of the upper bound of the monomial corresponding to that row. The explicit computation is given in Appendix A, while we only state the results here.

$$\#u = \binom{m+1}{2} + 3\binom{m+2}{4} = \frac{1}{8}m^4 + o(m^4)$$
$$\#v = \#w = \#x = \binom{m+2}{3} + 2\binom{m+2}{4} = \frac{1}{12}m^4 + o(m^4).$$

Recall that the enabling condition for the lattice attack is $\det(L) > 1$. With the previously derived values and neglecting low order terms as well as setting $A = N$, we are able to write the determinant as

$$\det(L) = U^{-\frac{1}{8}m^4} V^{-\frac{1}{12}m^4} W^{-\frac{1}{12}m^4} X^{-\frac{1}{12}m^4} N^{\frac{1}{8}m^4}.$$

If we use the upper bounds $(U, V, W, X) = (N^{2\delta}, N^{\delta}, N^{\frac{1}{2}+2\delta}, N^{\frac{1}{2}+\delta})$ on the sizes of the variables, we derive the condition

$$\delta < \frac{1}{14} \approx 0.071.$$

This is the same asymptotic bound that was obtained by Jochemsz and May [JM07] without extrashifts. So, unfortunately, our new lattice does not improve the asymptotic bound of [JM07]. But, as opposed to [JM07], our approach requires smaller lattice dimensions. Asymptotically, [JM07] need to LLL-reduce a lattice of size m^3, while our approach requires only lattice dimension $\frac{1}{2}m^3$. Figure 4 shows a comparison of the two methods in terms of the size of d_p, d_q that can be attacked.

While our approach clearly allows for attacking larger values of CRT-exponents in practice, we would also like to stress the fact that as opposed

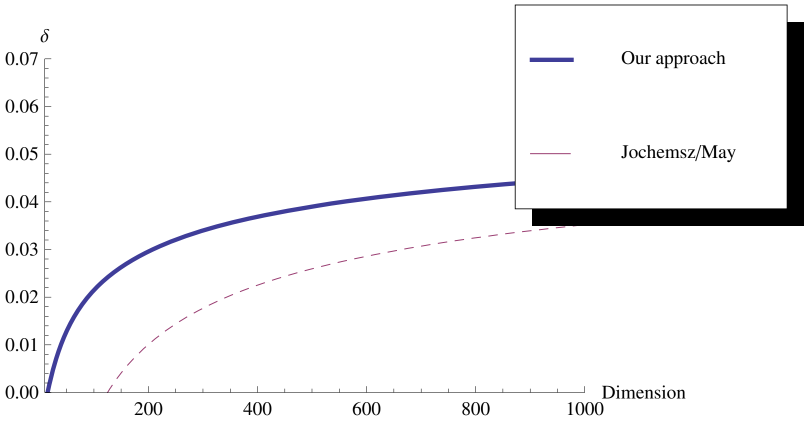


Fig. 4. Comparison of the achievable bound depending on the lattice dimension

to [JM07] the experimental behavior of our attack can be completely explained by our theoretical analysis – thereby also explaining the experimental behavior of [JM07]. We will show this in the subsequent section.

If we also use so-called extrashifts then we end up with a slightly improved bound of $d_p, d_q \leq N^{0.073}$ as in [JM07]. The analysis can be done in a similar fashion to the case without extrashifts. We carry out the calculations in Appendix B.

5 Experiments

The reason for carrying out various experiments for attacking CRT-RSA is twofold. First, we want to show that our analysis from Section 4 is indeed optimal. That is, the experimental behavior can be perfectly predicted by the analysis and there is no hope to improve the bound by this approach. Second, as our lattice-based approach is heuristic, we have to verify that the polynomials that we obtain after the lattice reduction are indeed coprime and thus allow for efficient recovery of their roots.

Table 1. Experimental Results

N	d_p, d_q	δ	lattice parameters	dim JM	LLL-time JM	LLL-time(s)
1000 bit	11 bit	0.0096	$m = 2, t = 1, dim = 30$	56	14	2
1000 bit	18 bit	0.0178	$m = 3, t = 1, dim = 60$	115	6100	258
1000 bit	22 bit	0.0226	$m = 3, t = 2, dim = 93$	–	–	3393
1000 bit	24 bit	0.0244	$m = 4, t = 1, dim = 105$	–	–	7572
1000 bit	29 bit	0.0291	$m = 4, t = 2, dim = 154$	–	–	61298
2000 bit	21 bit	0.0096	$m = 2, t = 1, dim = 30$	56	40	4
2000 bit	35 bit	0.0178	$m = 3, t = 1, dim = 60$	115	20700	613
2000 bit	45 bit	0.0226	$m = 3, t = 2, dim = 93$	–	–	13516
2000 bit	47 bit	0.0244	$m = 4, t = 1, dim = 105$	–	–	34305
5000 bit	48 bit	0.0096	$m = 2, t = 1, dim = 30$	56	379	39
5000 bit	89 bit	0.0178	$m = 3, t = 1, dim = 60$	–	–	5783
5000 bit	113 bit	0.0226	$m = 3, t = 2, dim = 93$	–	–	74417
10000 bit	96 bit	0.0096	$m = 2, t = 1, dim = 30$	56	2500	360
10000 bit	179 bit	0.0178	$m = 3, t = 1, dim = 60$	–	–	31226

We reimplemented the attack of [JM07] and used in the experiments the same modulus sizes and lattice parameters as done in [JM07]. Table 1 clearly shows the speedup for the LLL reduction. For example with parameters $m = 3$ and $t = 1$ our method is 20 to 30 times faster than the one of Jochemsz and May. As previously mentioned, this is due to the reduced lattice dimension[2]. While Jochemsz and May required the reduction of a lattice of dimension 115, our lattice only has dimension 60. Because of this smaller lattice dimension we were

[2] The lattice we are considering here is the one that serves as input to the LLL reduction routine. That is the sublattice containing zeros in the coordinates corresponding to the shift polynomials.

able to perform experiments on parameter sets that have been out of reach before.

Notice that the experimental results on the achievable sizes of d_p and d_q perfectly match the theoretically predicted bound δ. This is a strong indication that our approach is indeed optimal.

We ran our experiments using sage 4.1.1. and used the L^2 reduction algorithm from Nguyen and Stehlé [NS09]. The calculations were performed on an Quad Core Intel Xeon processor running at 2.66 GHz.

References

[BD99] Boneh, D., Durfee, G.: Cryptanalysis of RSA with Private Key d Less than $N^{0.292}$. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 1–11. Springer, Heidelberg (1999)

[BM01] Blömer, J., May, A.: Low Secret Exponent RSA Revisited. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 4–19. Springer, Heidelberg (2001)

[BM06] Bleichenbacher, D., May, A.: New Attacks on RSA with Small Secret CRT-Exponents. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 1–13. Springer, Heidelberg (2006)

[Cop96a] Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In: Maurer [Mau96], pp. 178–189 (1996)

[Cop96b] Coppersmith, D.: Finding a Small Root of a Univariate Modular Equation. In: Maurer [Mau96], pp. 155–165 (1996)

[Cop97] Coppersmith, D.: Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. J. Cryptology 10(4), 233–260 (1997)

[HG97] 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)

[HM09] Herrmann, M., May, A.: Attacking Power Generators Using Unravelled Linearization: When Do We Output Too Much? In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 487–504. Springer, Heidelberg (2009)

[JM06] 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)

[JM07] Jochemsz, E., May, A.: A Polynomial Time Attack on RSA with Private CRT-Exponents Smaller Than $N^{0.073}$. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 395–411. Springer, Heidelberg (2007)

[LLL82] Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring Polynomials with Rational Coefficients. Mathematische Annalen 261(4), 515–534 (1982)

[Mau96] Maurer, U.M. (ed.): EUROCRYPT 1996. LNCS, vol. 1070. Springer, Heidelberg (1996)

[NS09] Nguyen, P.Q., Stehlé, D.: An LLL Algorithm with Quadratic Complexity. SIAM J. Comput. 39(3), 874–903 (2009)

[QC82] Quisquater, J.J., Couvreur, C.: Fast Decipherment Algorithm for RSA Public-key Cryptosystem. Electronics Letters 18, 905 (1982)

[Wie90] Wiener, M.J.: Cryptanalysis of Short RSA Secret Exponents. IEEE Transactions on Information Theory 36(3), 553–558 (1990)

A Counting $\#u$, $\#v$, $\#w$, $\#x$

The monomials that contribute to the determinant are exactly the monomials of f^m that do not contain the variable y. Denote such a monomial by $u^{e_1}v^{e_2}w^{e_3}x^{e_4}$. In order to count the number of u's that contribute to the determinant we proceed as follows.

Let $e_1 = 0$. We have $e_2 + e_3 + e_4 \leq m$ with $e_i \in \mathbb{N}_0$, which transform into $e'_2 + e'_3 + e'_4 + h' \leq m + 4$ for a slack variable $h' \in \{1, \ldots, m+1\}$ and $e'_i = e_i + 1$. The number of such tuples is just the number of 4-partitions of $m + 4$, which is $\binom{m+3}{3}$. From these tuples we have to remove the ones with $e_i \geq 1$ for $i = 2, 3, 4$, because of the substitutions of vwx. The number of these tuples is $\binom{m}{3}$. For $e_1 = 1$, we proceed similarly and obtain $\binom{m+2}{3} - \binom{m-1}{3}$. We carry this out for all possibilities of e_1 and end up with $e_1 = m$, where we get $\binom{3}{3} - \binom{0}{3}$.

Now we know the number of occurences for each power u^i, $i = 0, \ldots, m$. In order to count the total number of u we compute the weighted sum as follows.

$$\begin{aligned}
\#\mathrm{u} &= \sum_{i=3}^{m+3}(m+3-i)\binom{i}{3} - \sum_{i=0}^{m}(m-i)\binom{i}{3} \\
&= \sum_{i=m+1}^{m+3}(m+3-i)\binom{i}{3} + \sum_{i=3}^{m}((m+3-i)-(m-i))\binom{i}{3} \\
&= 2\binom{m+1}{3} + \binom{m+2}{3} + 3\sum_{i=3}^{m}\binom{i}{3} = \binom{m+2}{3} - \binom{m+1}{3} + 3\sum_{i=3}^{m+1}\binom{i}{3}
\end{aligned}$$

Using the identities $\binom{n}{k} - \binom{n-1}{k} = \binom{n-1}{k-1}$ and $\sum_{i=0}^{n}\binom{i}{k} = \binom{n+1}{k+1}$ we eventually obtain

$$\#\mathrm{u} = \binom{m+1}{2} + 3\binom{m+2}{4}.$$

Thus, $\#u = \frac{1}{8}m^4 + o(m^4)$.

Counting the number of occurrences of v, w and x can be done in a similar way and we obtain

$$\begin{aligned}
\#v = \#w = \#x &= \sum_{i=3}^{m+3}(m+3-i)\binom{i}{3} - \sum_{i=1}^{m+1}(m+1-i)\binom{i}{3} \\
&= 2\sum_{i=0}^{m+1}\binom{i}{3} + \binom{m+2}{3} = 2\binom{m+2}{4} + \binom{m+2}{3} \\
&= \frac{1}{12}m^4 + o(m^4).
\end{aligned}$$

B Improving the Bound Using Extrashifts

In the following we will show that it is possible to improve the bound $\delta < \frac{1}{14} \approx 0.0714$ to $\delta \approx 0.0734$ by using so-called extrashifts. In this case, we use the set of shifts

$$S = \bigcup_{t_1=0}^{t} \bigcup_{t_2=0}^{t-t_1} \{u^{e_1+t_1} v^{e_2+t_2} w^{e_3} x^{e_4} y^{e_5} \mid u^{e_1} v^{e_2} w^{e_3} x^{e_4} y^{e_5} \text{ is monomial of } f^{m-1}\}.$$

To estimate the number of shifts, one may use a combinatorial proof as in Section 4 and count the number of all monomials minus the monomials having $e_2, e_3, e_4 \geq 1$. However, we choose to use a computational approach here and simply evaluate a series of sums.

The shift monomials can be characterized by the set $S_1 \setminus S_2$, where S_1 is the set of all shifts and S_2 are the shifts that have to be removed due to the substitution of vwx.

$$u^{e_1} v^{e_2} w^{e_3} x^{e_4} y^{e_5} \in S_1 \Leftrightarrow \begin{cases} e_5 = 0, \ldots, m-1 \\ e_4 = 0, \ldots, m-1-e_5 \\ e_3 = 0, \ldots, m-1-e_5-e_4 \\ e_2 = 0, \ldots, m-1-e_5-e_4-e_3+t \\ e_1 = 0, \ldots, m-1-e_5-e_4-e_3-e_2+t \end{cases}$$

$$u^{e_1} v^{e_2} w^{e_3} x^{e_4} y^{e_5} \in S_2 \Leftrightarrow \begin{cases} e_5 = 0, \ldots, m-1 \\ e_4 = 1, \ldots, m-1-e_5 \\ e_3 = 1, \ldots, m-1-e_5-e_4 \\ e_2 = 1, \ldots, m-1-e_5-e_4-e_3+t \\ e_1 = 0, \ldots, m-1-e_5-e_4-e_3-e_2+t \end{cases}$$

Setting $t = \tau m$, the resulting number of shifts is

$$|S_1 \setminus S_2| = \left(\frac{1}{8} + \frac{\tau}{2} + \frac{\tau^2}{2}\right) m^4 + o(m^4).$$

In a similar fashion we derive the exponents of the variables u, v, w and x contributing to the determinant. For example, to calculate the number of occurrences of u, we compute

$$\begin{aligned} s_u &= \sum_{e_4=0}^{m} \sum_{e_3=0}^{m-e_4} \sum_{e_2=0}^{m-e_4-e_3+t} \sum_{e_1=0}^{m-e_4-e_3-e_2+t} e_1 \\ &\quad - \sum_{e_4=1}^{m} \sum_{e_3=1}^{m-e_4} \sum_{e_2=1}^{m-e_4-e_3+t} \sum_{e_1=0}^{m-e_4-e_3-e_2+t} e_1 \\ &= \left(\frac{1}{8} + \frac{\tau}{2} + \frac{3\tau^2}{4} + \frac{\tau^3}{3}\right) m^4 + o(m^4). \end{aligned}$$

For the other values we obtain

$$\begin{aligned}
s_v &= \left(\frac{1}{12} + \frac{\tau}{3} + \frac{\tau^2}{2} + \frac{\tau^3}{3}\right) m^4 + o(m^4) \\
s_w &= \left(\frac{1}{12} + \frac{\tau}{3} + \frac{\tau^2}{4}\right) m^4 + o(m^4) \\
s_x &= \left(\frac{1}{12} + \frac{\tau}{3} + \frac{\tau^2}{4}\right) m^4 + o(m^4).
\end{aligned}$$

We use these values together with the upper bounds $(U, V, W, X) = (N^{2\delta}, N^{\delta}, N^{\frac{1}{2}+2\delta}, N^{\frac{1}{2}+\delta})$ to compute the determinant of the lattice. After that, we are able to solve the enabling condition $\det(L) > 1$ for δ and optimize the value of τ to maximize δ. We obtain $\tau \approx 0.381788$, which finally leads to the bound

$$\delta \leq 0.0734142.$$

Implicit Factoring with Shared Most Significant and Middle Bits

Jean-Charles Faugère, Raphaël Marinier, and Guénaël Renault

UPMC, Université Paris 06, LIP6
INRIA, Centre Paris-Rocquencourt, SALSA Project-team
CNRS, UMR 7606, LIP6
4, place Jussieu
75252 Paris, Cedex 5, France
jean-charles.faugere@inria.fr, raphael.marinier@polytechnique.edu,
guenael.renault@lip6.fr

Abstract. We study the problem of integer factoring given *implicit* information of a special kind. The problem is as follows: let $N_1 = p_1q_1$ and $N_2 = p_2q_2$ be two RSA moduli of same bit-size, where q_1, q_2 are α-bit primes. We are given the *implicit* information that p_1 and p_2 share t most significant bits. We present a novel and rigorous lattice-based method that leads to the factorization of N_1 and N_2 in polynomial time as soon as $t \geq 2\alpha + 3$. Subsequently, we heuristically generalize the method to k RSA moduli $N_i = p_iq_i$ where the p_i's all share t most significant bits (MSBs) and obtain an improved bound on t that converges to $t \geq \alpha + 3.55\ldots$ as k tends to infinity. We study also the case where the k factors p_i's share t contiguous bits in the middle and find a bound that converges to $2\alpha + 3$ when k tends to infinity. This paper extends the work of May and Ritzenhofen in [9], where similar results were obtained when the p_i's share least significant bits (LSBs). In [15], Sarkar and Maitra describe an alternative but heuristic method for only two RSA moduli, when the p_i's share LSBs and/or MSBs, or bits in the middle. In the case of shared MSBs or bits in the middle and two RSA moduli, they get better experimental results in some cases, but we use much lower (at least 23 times lower) lattice dimensions and so we obtain a great speedup (at least 10^3 faster). Our results rely on the following surprisingly simple algebraic relation in which the shared MSBs of p_1 and p_2 cancel out: $q_1N_2 - q_2N_1 = q_1q_2(p_2 - p_1)$. This relation allows us to build a lattice whose shortest vector yields the factorization of the N_i's.

Keywords: implicit factorization, lattices, RSA.

1 Introduction

Efficient factorization of large integers is one of the most fundamental problem of Algorithmic Number Theory, and has fascinated mathematicians for centuries. It has been particularly intensively studied over the past 35 years, all the more that efficient factorization leads immediately to an attack of the RSA Cryptosystem. In the 1970's, the first general-purpose sub-exponential algorithm for factoring was developed by Morrison and Brillhart in [11] (improving a method described for the first time in [7]), using continued fraction techniques. Several faster general-purpose algorithms have been proposed over the past years, the most recent and efficient being the general number field sieve (GNFS) [8], proposed in 1993. It is not known whether factoring integers can be done in polynomial time on a classical Turing machine. On quantum machines, Shor's algorithm [16] allows polynomial-time factoring of integers. However, it is still an open question whether a capable-enough quantum computer can be built.

At the same time, the problem of factoring integers given additional information about their factors has been studied since 1985. In [14], Rivest and Shamir showed that $N = pq$ of bit-size n and with balanced factors ($\log_2(p) \approx \log_2(q) \approx \frac{n}{2}$) can be factored in polynomial time as soon as we have access to an *oracle* that returns the $\frac{n}{3}$ most significant bits (MSBs) of p. Beyond its theoretical interest, the motivation behind this is mostly of cryptographic nature. In fact, during an attack of an RSA-encrypted exchange, the cryptanalyst may have access to additional information beyond the RSA public parameters (e,N), that may be gained for instance through side-channel attacks revealing some of the bits of the secret factors. Besides, some variations of the RSA Cryptosystem purposely leak some of the secret bits (for instance, [17]). In 1996, Rivest and Shamir's results were improved in [2] by Coppersmith applying lattice-based methods to the problem of finding small integer roots of bivariate integer polynomials (the now so-called *Coppersmith's method*). It requires only half of the most significant bits of p to be known to the cryptanalyst (that is $\frac{n}{4}$).

In PKC 2009, May and Ritzenhofen [9] significantly reduced the power of the oracle. Given an RSA modulus $N_1 = p_1q_1$, they allow the oracle to output a new and different RSA modulus $N_2 = p_2q_2$ such that p_1 and p_2 share at least t least significant bits (LSBs). Note that the additional information here is only *implicit*: the attacker does not know the actual value of the t least significant bits of the p_i's, he only knows that p_1 and p_2 share them. In the rest of the paper, we will refer to this problem as the problem of *implicit factoring*. When q_1 and q_2 are α-bit primes, May and Ritzenhofen's lattice-based method rigorously finds in quadratic time the factorization of N_1 and N_2 when $t \geq 2\alpha + 3$. Besides, their technique heuristically generalizes to $k-1$ oracle queries that give access to k different RSA moduli $N_i = p_iq_i$ with all the p_i's sharing t least significant bits. With $k-1$ queries the bound on t improves to: $t \geq \frac{k}{k-1}\alpha$. Note that these results are of interest for unbalanced RSA moduli: for instance, if $N_1 = p_1q_1$, $N_2 = p_2q_2$ are 1000-bit RSA moduli and the q_i's are 200-bit primes, knowing that p_1 and p_2 share at least 403 least significant bits out of 800 is enough to factorize N_1 and N_2 in polynomial time. Note also that the method absolutely requires that the shared bits be the least significant ones. They finally apply their method to factorize k n-bit balanced RSA moduli $N_i = p_iq_i$ under some conditions and with an additional exhaustive search of $2^{\frac{n}{4}}$.

Very recently, in [15], Sarkar and Maitra applied Coppersmith and Gröbner-basis techniques on the problem of implicit factoring, and improved heuristically the bounds in some of the cases. Contrary to [9], their method applies when either (or both) LSBs or MSBs of p_1, p_2 are shared (or when bits in the middle are shared). Namely, in the case of shared LSBs they obtain better theoretical bounds on t than [9] as soon as $\alpha \geq 0.266n$. Besides, their experiments often perform better than their theoretical bounds, and they improve in practice the bound on t of [9] when $\alpha \geq 0.21n$. Note finally that their bounds are very similar in the two cases of shared MSBs and shared LSBs. Readers interested in getting their precise bounds may refer to their paper [15].

Unfortunately, Sarkar and Maitra's method is heuristic even in the case of two RSA moduli, and does not generalize to $k \geq 3$ RSA moduli. In fact, when the p_i's share MSBs and/or LSBs, their method consists in building a polynomial f_1 in three variables, whose roots are $(q_2+1, q_1, \frac{p_1-p_2}{2^\gamma})$, where γ is the number of shared LSBs between p_1 and p_2. That is, $\frac{p_1-p_2}{2^\gamma}$ represents the part of p_1-p_2 where the shared bits do not cancel out. To find the integer roots of f_1, they use the Coppersmith-like technique of [5] which consists in computing two (or more) new polynomials $f_2, f_3, \dots$ sharing the same roots as f_1. If the variety defined by $f_1, f_2, f_3, \dots$ is 0-dimensional, then the roots can be easily recovered computing resultants or Gröbner basis. However, with an input polynomial with more than two variables, the method is heuristic: there is no guarantee for the polynomials $f_1, f_2, f_3, \dots$ to define a 0-dimensional variety. We reproduced the results of Sarkar and Maitra and we observed that $f_1, f_2, f_3, \dots$ almost never defined a 0-dimensional variety. They observed however that it was possible to recover the roots of the polynomials directly by looking at the coefficients of the polynomials in the Gröbner basis of the ideal generated by the f_i's, even when the ideal was of positive dimension. The assumption on which their work relies is that it will always be possible. For instance, in the case of shared MSBs between p_1 and p_2, they found in their experiments that the Gröbner basis contained a polynomial multiple of $x - \frac{q_2}{q_1}y - 1$ whose coefficients lead immediately to the factorization of N_1 and N_2. They support their assumption by experimental data: in most cases their experiments perform better than their theoretical bounds. It seems nevertheless that their assumption is not fully understood.

Our contribution consists of a novel and rigorous lattice-based method that address the implicit factoring problem when p_1 and p_2 share *most* significant bits. That is, we obtained an analog of May and Ritzenhofen's results for shared MSBs, and our method is rigorous contrary to the work of Sarkar and Maitra in [15]. Namely, let $N_1 = p_1q_1$ and $N_2 = p_2q_2$ be two RSA moduli of same bit-size n. If q_1, q_2 are α-bit primes and p_1, p_2 share t most significant bits, our method provably factorizes N_1 and N_2 as soon as $t \geq 2\alpha + 3$ (which is the same as the bound on t for least significant bits in [9]). This is the first rigorous bound on t when p_1 and p_2 share most significant bits. From this method, we deduce a new heuristic lattice-based for the case when p_1 and p_2 share t bits in the middle. Moreover, contrary to [15], these methods heuristically generalize to an arbitrary number k of RSA moduli and do not depend on the position of the shared bits in the middle, allowing us to factorize k RSA moduli as soon as $t \geq \frac{k}{k-1}\alpha + 6$ (resp. $t \geq \frac{2k}{k-1}\alpha + 7$) most significant bits (resp. bits in the middle) are shared between the p_i's (more precise bounds are stated later in this paper). A summary of the comparison of our method with the methods in [9] and [15] can be found in table 1.

Let's give the main idea of our method with 2 RSA moduli in the case of shared MSB's. Consider the lattice L spanned by the row vectors $\mathbf{v_1}$ and $\mathbf{v_2}$ of the following matrix:

$$\begin{pmatrix} K & 0 & N_2 \\ 0 & K & -N_1 \end{pmatrix} \quad \text{where } K = \lfloor 2^{n-t+\frac{1}{2}} \rfloor$$

Table 1. Comparison of our results against the results of [9] and [15] with k RSA moduli

	May, Ritzenhofen's Results [9]	Sarkar, Maitra's Results [15]	Our results
$k = 2$	When p_1, p_2 share t LSBs: rigorous bound of $t \geq 2\alpha + 3$ using 2-dimensional lattices of $\mathbb{Z}^2$.	When p_1, p_2 share either t LSBs or MSBs: heuristic bound better than $t \geq 2\alpha + 3$ when $\alpha \geq 0.266n$, and experimentally better when $\alpha \geq 0.21n$. In the case of t shared bits in the middle, better bound than $t \geq 4\alpha + 7$ but depending on the position of the shared bits. Using 46-dimensional lattices of $\mathbb{Z}^{46}$	When p_1, p_2 share t MSBs: rigorous bound of $t \geq 2\alpha + 3$ using 2-dimensional lattices of $\mathbb{Z}^3$. In the case of t bits shared in the middle: heuristic bound of $t \geq 4\alpha + 7$ using 3-dimensional lattices of $\mathbb{Z}^3$.
$k \geq 3$	When the p_i's all share t LSBs: heuristic bound of $t \geq \frac{k}{k-1}\alpha$ using k-dimensional lattices of $\mathbb{Z}^k$.	Cannot be directly applied.	When the p_i's all share t MSBs (resp. bits in the middle): heuristic bound of $t \geq \frac{k}{k-1}\alpha + \delta_k$ (resp. $t \geq \frac{2k}{k-1}\alpha + \delta_k$), with $\delta_k \leq 6$ (resp. ≤ 7) and using k-dimensional ($\frac{k(k+1)}{2}$-dimensional) lattices of $\mathbb{Z}^{\frac{k(k+1)}{2}}$.

Consider also the following vector in L:

$$\mathbf{v_0} = q_1\mathbf{v_1} + q_2\mathbf{v_2} = (q_1K, q_2K, q_1q_2(p_2 - p_1))$$

The key observation is that the t shared significant bits of p_1 and p_2 cancel out in the algebraic relation $q_1N_2 - q_2N_1 = q_1q_2(p_2 - p_1)$. Furthermore, we choose K in order to force the coefficients of a shortest vector of L on the basis $(\mathbf{v_1}, \mathbf{v_2})$ to be of the order of $2^\alpha \approx q_1 \approx q_2$. We prove in the next section that $\mathbf{v_0}$ is indeed a shortest vector of L (thus N_1 and N_2 can be factored in polynomial time) as soon as $t \geq 2\alpha + 3$. Besides, we generalized this construction to an arbitrary number of k RSA moduli such that a small vector of the lattice harnesses the same algebraic relation, and to shared middle bits. However, the generalized constructions in both cases become heuristic: we use the Gaussian heuristic to find a condition on t for this vector to be a shortest of the lattice.

Applications of implicit factoring have not yet been extensively studied, and we believe that they will develop. The introduction of [9] gives some ideas for possible applications. They include destructive applications with malicious manipulation of public key generators, as well as possibly constructive ones. Indeed, our work shows that when $t \geq 2\alpha + 3$, it is as hard to factorize $N_1 = p_1q_1$, as generating $N_2 = p_2q_2$ with p_2 sharing t most significant bits with p_1. This problem could form the basis of a cryptographic primitive.

Throughout this paper, we heavily use common results on euclidean lattice. A summary of these results can be found in appendix A. The paper is organized as follows. In section 2, we present our rigorous method in the case of shared MSB's and two RSA moduli, we generalize it to k RSA moduli in section 3. In section 4, we present our method in the case of shared bits in the middle. Finally, in section 5 we present our experiments that strongly support the assumption we made in the case of k RSA moduli and of shared middle bits.

2 Implicit Factoring of Two RSA Moduli with Shared MSBs

In this section, we study the problem of factoring two n-bit RSA moduli: $N_1 = p_1q_1$ and $N_2 = p_2q_2$, where q_1 and q_2 are α-bit primes, given only the implicit hint that p_1 and p_2 share t most significant bits (MSBs) that are *unknown* to us. We will show that N_1 and N_2 can be factored in quadratic time as soon as $t \geq 2\alpha + 3$. By saying that the primes p_1, p_2 of maximal bit-size $n - \alpha + 1$ share t MSBs, we really mean that $|p_1 - p_2| \leq 2^{n-\alpha-t+1}$.

Let's consider the lattice L spanned by the row vectors (denoted by $\mathbf{v_1}$ and $\mathbf{v_2}$) of the following matrix:

$$M = \begin{pmatrix} K & 0 & N_2 \\ 0 & K & -N_1 \end{pmatrix} \quad \text{where } K = \lfloor 2^{n-t+\frac{1}{2}} \rfloor$$

We have the following immediate lemma that makes our method work:

Lemma 1. *Let $\mathbf{v_0}$ be the vector of L defined by $\mathbf{v_0} = q_1\mathbf{v_1} + q_2\mathbf{v_2}$. Then $\mathbf{v_0}$ can be rewritten as $\mathbf{v_0} = (q_1K, q_2K, q_1q_2(p_2 - p_1))$.*

Note that the shared MSBs of p_1 and p_2 cancel each other out in the difference $p_2 - p_1$. Each of the coefficients of $\mathbf{v_0}$ are thus integers of roughly $(n+\alpha-t)$ bits. Provided that t is sufficiently large, $\pm\mathbf{v_0}$ may be a shortest vector of L that can be found using Lagrange reduction on L. Moreover, note that as soon as we retrieve $\mathbf{v_0}$ from L, factoring N_1 and N_2 is easily done by dividing the first two coordinates of $\mathbf{v_0}$ by K (which can be done in quadratic time in n). Proving that $\mathbf{v_0}$ is a shortest vector of L under some conditions on t is therefore sufficient to factorize N_1 and N_2.

We first give an intuition on the bound on t that we can expect, and we give after that a proof that $\pm\mathbf{v_0}$ is indeed the shortest vector of L under a similar condition.

The volume of L is the square root of the determinant of the Gramian matrix of L given by $MM^t = \begin{pmatrix} K^2+N_2^2 & -N_1N_2 \\ -N_1N_2 & K^2+N_1^2 \end{pmatrix}$. That is, $vol(L) = K\sqrt{N_1^2+N_2^2+K^2}$ which can be approximated by 2^{2n-t} because $K^2 \approx 2^{2(n-t)}$ is small compared to the $N_i^2 \approx 2^{2n}$. The norm of $\mathbf{v_0}$ is approximately $2^{n+\alpha-t}$, because each of its coefficients have roughly $n+\alpha-t$ bits. If $\mathbf{v_0}$ is a shortest vector of L, it must be smaller than the Minkowski bound applied to L: $2^{n+\alpha-t} \approx \|\mathbf{v_0}\| \leq \sqrt{2}\,\mathrm{Vol}(L)^{1/2} \approx 2^{n-t/2}$, which happens when $t \geq 2\alpha$. The following lemma affirms that $\mathbf{v_0}$ is indeed a shortest vector of L under a similar condition on t.

Lemma 2. *Let L be the lattice generated by the row vectors $\mathbf{v_1}$ and $\mathbf{v_2}$ of M and let $\mathbf{v_0} = q_1\mathbf{v_1} + q_2\mathbf{v_2} = (q_1K, q_2K, q_1q_2(p_2 - p_1))$ as defined in Lemma 1. The vector $\pm\mathbf{v_0}$ is the shortest vector of the lattice L as soon as $t \geq 2\alpha + 3$.*

Proof. Let $(\mathbf{b_1}, \mathbf{b_2})$ be the resulting basis from the Lagrange reduction on L. This reduced basis verifies $\|\mathbf{b_1}\| = \lambda_1(L), \|\mathbf{b_2}\| = \lambda_2(L)$, and, by Hadamard's inequality one have: $\|\mathbf{b_1}\|\|\mathbf{b_2}\| \geq \mathrm{Vol}(L)$. As $\mathbf{v_0}$ is in the lattice, $\|\mathbf{b_1}\| = \lambda_1(L) \leq \|\mathbf{v_0}\|$. Hence we get $\|\mathbf{b_2}\| \geq \frac{\mathrm{Vol}(L)}{\|\mathbf{v_0}\|}$. Moreover, if $\mathbf{v_0}$ is strictly shorter that $\mathbf{b_2}$, $\mathbf{v_0}$ is a multiple of $\mathbf{b_1}$; for otherwise $\mathbf{b_2}$ would not be the second minimum of the lattice. In this case, $\mathbf{v_0} = a\mathbf{b_1} = a(b\mathbf{v_1} + c\mathbf{v_2}), a, b, c \in \mathbb{Z}$, and looking at the first two coefficients of $\mathbf{v_0}$, we get that $ab = q_1$ and $ac = q_2$. Since the q_i's are prime, we conclude that $a = \pm 1$, that is, $\mathbf{v_0} = \pm \mathbf{b_1}$. Using the previous inequality, a condition for $\mathbf{v_0}$ to be strictly shorter than $\mathbf{b_2}$ is:

$$\|\mathbf{v_0}\|^2 < \mathrm{Vol}(L) \tag{1}$$

Let's upper-bound the norm of $\mathbf{v_0}$ and lower-bound $\mathrm{Vol}(L)$. We first provide simple bounds that proves the lemma when $t \geq 2\alpha + 4$ and derive secondly tighter bounds that require only $t \geq 2\alpha + 3$.

The p_i's have at most $n - \alpha + 1$ bits, and they share their t most significant bits so $|p_2 - p_1| \leq 2^{n-\alpha+1-t}$. We thus have the inequality $\|\mathbf{v_0}\|^2 \leq 2^{2(n-t)+1}(q_1^2 + q_2^2) + q_1^2 q_2^2 (p_1 - p_2)^2$ which implies

$$\|\mathbf{v_0}\|^2 \leq 2^{2(n+\alpha-t)+2} + 2^{2(\alpha+n+1-t)} \leq 2^{2(n+\alpha-t)+3} \tag{2}$$

We can lower-bound the volume of L, using that $N_1, N_2 \geq 2^{n-1}$ and $K^2 \geq 2^{2(n-t)}$:

$$\mathrm{Vol}(L)^2 = K^2(N_1^2 + N_2^2 + 2^{2(n-t)}) > 2^{4n-2t-1} \tag{3}$$

Using inequalities (2) and (3), the inequality (1) is true provided that: $2^{2(n+\alpha-t)+3} \leq 2^{2n-t-\frac{1}{2}}$ which is equivalent to (as t and α are an integers):

$$t \geq 2\alpha + 4 \tag{4}$$

We have thus proved the lemma under condition (4). We now refine the bounds on $\|\mathbf{v_0}\|$ and $\mathrm{Vol}(L)$ in order to prove the tight case.

The integers q_1 and q_2 are α-bit primes, therefore $q_i \leq 2^\alpha - 1$, $(i = 1, 2)$. Define ε_1 by $2^\alpha - 1 = 2^{\alpha - \varepsilon_1}$. We get $q_i^2 \leq 2^{2\alpha - 2\varepsilon_1}, (i = 1, 2)$. Moreover, since $K = \lfloor 2^{n-t+\frac{1}{2}} \rfloor$, we have $K^2 \leq 2^{2(n-t)+1}$. From these inequalities, we can upper-bound $K^2 q_i^2$

$$K^2 q_i^2 \leq 2^{2(n-t+\alpha)+1-2\varepsilon_1},\ (i = 1, 2) \tag{5}$$

The p_i's have at most $n - \alpha + 1$ bits and they share t bits, so $(p_2 - p_1)^2 \leq 2^{2(n-\alpha+1-t)}$. Thus, using the upper-bound on the q_i^2, we have

$$q_1^2 q_2^2 (p_2 - p_1)^2 \leq 2^{2(n-t+\alpha+1-2\varepsilon_1)} \tag{6}$$

We can finally bound $\|\mathbf{v_0}\|^2 = K^2(q_1^2 + q_2^2) + q_1^2 q_2^2 (p_2 - p_1)^2$ using (5) and (6):

$$\|\mathbf{v_0}\|^2 \leq 2^{2(n+\alpha-t)+2-2\varepsilon_1} + 2^{2(n-t+\alpha+1-2\varepsilon_1)} \leq 2^{2(n+\alpha-t)+3-\varepsilon_1} \tag{7}$$

Let's now define ε_2 by the equality $2^{n-t+1/2}-1 = 2^{n-t+1/2-\varepsilon_2}$. We have that $K = \lfloor 2^{n-t+\frac{1}{2}} \rfloor \geq 2^{n-t+1/2-\varepsilon_2}$ and $N_i^2 \geq 2^{2n-2}$, we can therefore lower-bound $\mathrm{Vol}(L)^2$:

$$\mathrm{Vol}(L)^2 = K^2(N_1^2+N_2^2+2^{2(n-t)}) > K^2(N_1^2+N_2^2) \geq 2^{4n-2t-2\varepsilon_2} \tag{8}$$

Using the inequalities (7) and (8), the condition (1) is true under the new condition $2^{2(n+\alpha-t)+3-\varepsilon_1} \leq 2^{2n-t-\varepsilon_2}$ which is equivalent to $t \geq 2\alpha+3+\varepsilon_2-\varepsilon_1$.

Since $\varepsilon_1 = \log_2(\frac{1}{1-\frac{1}{2^\alpha}})$, $\varepsilon_2 = \log_2(\frac{1}{1-\frac{1}{2^{n-t+\frac{1}{2}}}})$ and $\alpha \leq n-t$, we have $\varepsilon_2 \leq \varepsilon_1$ and the result follows.

From the preceding Lemmas 1 and 2, one can deduce the following result.

Theorem 1. *Let $N_1 = p_1q_1, N_2 = p_2q_2$ be two n-bit RSA moduli, where the q_i's are α-bit primes and the p_i's are primes that share t most significant bits. If $t \geq 2\alpha+3$, then N_1 and N_2 can be factored in quadratic time in n.*

Proof. Let L be the lattice generated by $\mathbf{v_1}$ and $\mathbf{v_2}$ as above. Since the norms of $\mathbf{v_1}$ and $\mathbf{v_2}$ are bounded by 2^{n+1}, computing the reduced basis $(\mathbf{b_1}, \mathbf{b_2})$ takes a quadratic time in n. By Lemma 2 we know that $\mathbf{b_0} = \pm\mathbf{v_0}$ as soon as $t \geq 2\alpha+3$. The factorization of N_1 of N_2 follows from the description of $\mathbf{v_0}$ given by the lemma 1.

Remark 1. For our analysis, the value $K = \lfloor 2^{n-t+\frac{1}{2}} \rfloor$ is indeed the best possible value. If we use $K = \lfloor 2^{n-t+\gamma} \rfloor$, we obtain the bound $t \geq 2\alpha + f(\gamma)$ with $f(\gamma) = \frac{3}{2} - \gamma + \log_2(2+2^{2\gamma})$. The minimum of f is 3 and is attained in $\gamma = \frac{1}{2}$.

3 Implicit Factoring of *k* RSA Moduli with Shared MSBs

The construction of the lattice for 2 RSA moduli naturally generalizes to an arbitrary number k of moduli. Similarly, we show that a short vector $\mathbf{v_0}$ of the lattice allows us to recover the factorization of the N_i's. This vector takes advantage of the relations $q_iN_j - q_jN_i = q_iq_j(p_j-p_i)$ for all $i,j \in \{1,\dots,k\}$. However, we were unable to prove that $\mathbf{v_0}$ is a shortest vector of the lattice. Therefore, our method relies on the Gaussian heuristic to estimate the conditions under which $\mathbf{v_0}$ should be a shortest vector of the lattice. Experimental data in section 5 confirms that this heuristic is valid in nearly all the cases.

In this section, we are given k RSA moduli of n bits $N_1 = p_1q_1, \dots, N_k = p_kq_k$ where the q_i's are α-bit primes and the p_i's are primes that all share t most significant bits.

Let us construct a matrix M whose row vectors will form a basis of a lattice L; this matrix will have k rows and $k + \binom{k}{2} = \frac{k(k+1)}{2}$ columns. Denote by $s_1, \dots, s_m$ with $m = \binom{k}{2}$ all the subsets of cardinality 2 of $\{1,2,\dots,k\}$. To each of the s_i's, associate a column vector $\mathbf{c_i}$ of size k the following way. Let a,b be the two elements of s_i, with $a < b$. We set the a-th element of $\mathbf{c_i}$ to N_b, the b-th element of $\mathbf{c_i}$ to $-N_a$, and all other elements to zero. Finally, one forms M by concatenating column-wise the matrix $KI_{k\times k}$, where $I_{k\times k}$ is the identity matrix of size k, along with the matrix C_m composed by the m column vectors $\mathbf{c_1}, \dots, \mathbf{c_m}$. K is chosen to be $\lfloor 2^{n-t+\frac{1}{2}} \rfloor$. We will call $\mathbf{v_1}, \dots, \mathbf{v_k}$ the row vectors of M.

To make things more concrete, consider the example of $k = 4$. Up to a reordering of the columns (that changes nothing to the upcoming analysis),

$$M = \begin{pmatrix} K & 0 & 0 & 0 & N_2 & N_3 & N_4 & 0 & 0 & 0 \\ 0 & K & 0 & 0 & -N_1 & 0 & 0 & N_3 & N_4 & 0 \\ 0 & 0 & K & 0 & 0 & -N_1 & 0 & -N_2 & 0 & N_4 \\ 0 & 0 & 0 & K & 0 & 0 & -N_1 & 0 & -N_2 & -N_3 \end{pmatrix} \text{ where } K = \lfloor 2^{n-t+\frac{1}{2}} \rfloor \tag{9}$$

Notice that the columns $k+1$ to $k+m$ correspond to all the 2-subsets of $\{1,2,3,4\}$.

Similarly to the case of 2 RSA moduli (lemma 1), L contains a short vector that allows us to factorize all the N_i's:

Lemma 3. *Let* $\mathbf{v_0}$ *be the vector of L defined by* $\mathbf{v_0} = \sum_{i=1}^{k} q_i \mathbf{v_i}$. *Then* $\mathbf{v_0}$ *can be rewritten as follows:*

$$\mathbf{v_0} = (q_1 K, \ldots, q_k K, \underbrace{\ldots, q_a q_b (p_b - p_a), \ldots}_{\forall \{a,b\} \subset \{1,\ldots,k\}})$$

Proof. For $1 \le i \le m$, let a,b be such that $s_i = \{a,b\}$ and $a < b$. By the construction of the $\mathbf{c_i}$'s, we get that the $(k+i)$-th coordinate of $\mathbf{v_0}$ is equal to $q_a N_b - q_b N_a = q_a q_b$ $(p_b - p_a)$. □

Remark that $\mathbf{v_0}$ is short because its m last coordinates harness the cancellation of the t most significant bits between the p_i's. Retrieving $\pm\mathbf{v_0}$ from L leads immediately to the factorization of all the N_i's, dividing its first k coordinates by K.

Assumption 1. *If* $\pm\mathbf{v_0}$ *is shorter than the Gaussian heuristic* $\lambda_1(L) \approx \sqrt{\frac{d}{2\pi e}} \mathrm{Vol}(L)^{\frac{1}{d}}$ *applied to the d-dimensional lattice L then it is a shortest vector of L.*

This assumption is supported by experimental data in the section 5. We found it to be almost always true in practice. This condition can be seen as an analog of condition 1 of section 2 in the case of two RSA moduli.

Let's derive a bound on t so that $\mathbf{v_0}$ is smaller than the Gaussian heuristic applied to L. The norm of $\mathbf{v_0}$ can be computed and upper-bounded easily: $\|\mathbf{v_0}\|^2 = K^2 \left(\sum_{i=1}^{k} q_i^2\right)$ $+ \sum_{\{i,j\} \subset \{1,\ldots,k\}} q_i^2 q_j^2 (p_i - p_j)^2 \le k^2 2^{2(n+\alpha-t)+1}$. Computing the volume of L is a bit more involved, we refer to Lemma 5 of appendix B: $\mathrm{Vol}(L) = K\left(K^2 + \sum_{i=1}^{k} N_i^2\right)^{\frac{k-1}{2}}$ and thus $\mathrm{Vol}(L) \ge 2^{n-t}\left(\sqrt{k} 2^{n-1}\right)^{k-1}$.

We now seek the condition on t for the norm of $\mathbf{v_0}$ to be smaller than the Gaussian heuristic. Using the two previous inequalities on $\|\mathbf{v_0}\|$ and $\mathrm{Vol}(L)$, we get the stricter condition:

$$k^2 2^{2(n+\alpha-t)+1} \le \frac{k}{2\pi e}\left(2^{n-t}\left(\sqrt{k}2^{n-1}\right)^{k-1}\right)^{\frac{2}{k}}$$

Expanding everything and extracting t, we get the following condition:

$$t \ge \frac{k}{k-1}\alpha + 1 + \frac{k}{2(k-1)}\left(2 + \frac{\log_2(k)}{k} + \log_2(\pi e)\right) \tag{10}$$

When $k \ge 3$, we can derive a simpler and stricter bound on t: $t \ge \frac{k}{k-1}\alpha + 6$

Finally, as $\pm\mathbf{v_0}$ is now the shortest vector of L under Assumption 1, it can be found in time $\mathscr{C}(k, \frac{k(k+1)}{2}, n)$ where $\mathscr{C}(k,s,B)$ is the time to find a shortest vector of a k-dimensional lattice of $\mathbb{Z}^s$ given by B-bit basis vectors. We just proved the following theorem:

Theorem 2. *Let $N_1 = p_1q_1, \dots, N_k = p_kq_k$ be k n-bit RSA moduli, with the q_i's being α-bit primes, and the p_i's being primes that all share t most significant bits. Under Assumption 1, the N_i's can be factored in time $\mathscr{C}(k, \frac{k(k+1)}{2}, n)$, as soon as t verifies equation* (10).

Remark 2. Note that we can find a shortest vector of the lattice of Theorem 2 using Kannan's algorithm (Theorem 6 in appendix A) in time $\mathscr{O}(\mathscr{P}(n,k)k^{\frac{k}{2e}+o(k)})$ where $\mathscr{P}$ is a polynomial. It implies that we can factorize all $N_1, \dots, N_k$ in time polynomial in n as soon as k is constant or k^k is a polynomial in n. Unfortunately, to the best of our knowledge, this algorithm is not implemented in the computer algebra system Magma [1] on which we implemented the methods. In our experiments, to compute a shortest vector of the lattice, we used instead the Schnorr-Euchner's enumeration algorithm which is well known (see [4,3]) to perform well beyond small dimension (≤ 50) and this step in Magma took less than 1 minute for $k \leq 40$. One may also reduce the lattice using LLL algorithm instead of Schnorr-Euchner's enumeraion. If t is not too close to the bound of Theorem 2, the Gaussian heuristic suggests that the gap (see Definition 1 in the appendix) of the lattice is large, and thus LLL may be able to find a shortest vector of L even in medium dimension (50–200).

Similarly to the case of 2 RSA moduli, $K = \lfloor 2^{n-t+\frac{1}{2}} \rfloor$ is optimal for our analysis. Indeed, if we redo the analysis with $K = \lfloor 2^{n-t+\gamma} \rfloor$, we find that the optimal value for γ is the one that minimizes the function $f_k = \gamma \mapsto \frac{1}{2}k\log_2(k-1+2^{2\gamma-1}) - \gamma$, which is $\gamma = \frac{1}{2}$ regardless of k.

Finally, note that a slightly tighter bound (differing to equation 10 by a small additive constant) may be attained by bounding $\|\mathbf{v_0}\|$ and $\mathrm{Vol}(L)$ more precisely.

4 Implicit Factoring with Shared Bits in the Middle

In this section, we are given k RSA moduli of n bits $N_1 = p_1q_1, \dots, N_k = p_kq_k$ where the q_i's are α-bit primes and the p_i's are primes that all share t bits from position t_1 to $t_2 = t_1 + t$. More precisely, these RSA moduli all verify:

$$N_i = p_iq_i = (p_{i_2}2^{t_2} + p2^{t_1} + p_{i_0})q_i$$

where p is the integer part shared by all the moduli. Contrary to the LSB case presented in [9] and the MSB one developed in the previous sections, the method we present here is heuristic even when $k = 2$. We sketch now our method when $k = 2$ and present the details on the general result later. When $k = 2$, we have a system of two equations in four variables p_1, q_1, p_2, q_2: $N_1 = p_1q_1 = (p_{1_2}2^{t_2} + p2^{t_1} + p_{1_0})q_1$ and $N_2 = p_2q_2 = (p_{2_2}2^{t_2} + p2^{t_1} + p_{2_0})q_2$. Similarly to the LSB's case (see [9]), this system can be reduced modulo 2^{t_2}. One obtains a system of two equations with 5 variables p, p_{1_0}, p_{2_0}, q_1, q_2:

$$\begin{cases} (p2^{t_1} + p_{1_0})q_1 = N_1 \mod 2^{t_2} \\ (p2^{t_1} + p_{2_0})q_2 = N_2 \mod 2^{t_2} \end{cases} \tag{11}$$

The problem can now be seen as a modular implicit factorization of N_1 and N_2 with shared MSBs. Thus, we adapt the method we proposed in section 2 to the modular case. More precisely, we consider the lattice L defined by the rows of the matrix

$$M = \begin{pmatrix} K & 0 & N_2 \\ 0 & K & -N_1 \\ 0 & 0 & 2^{t_2} \end{pmatrix} \tag{12}$$

Let $\mathbf{v_0}$ be the vector (q_1K, q_2K, r) with r being the unique remainder of $q_1N_2 - q_2N_1$ modulo 2^{t_2} in $]-2^{t_2-1}, 2^{t_2-1}]$. Clearly, $\mathbf{v_0}$ is in L. As in the section 3, we search for a condition on the integer t under which $\pm\mathbf{v_0}$ is the shortest vector in L under Assumption 1 (here, the dimension of the lattice L is 3). The integer K will be set at the end of the analysis.

We have $\|\mathbf{v_0}\|^2 = K^2(q_1^2+q_2^2)+r^2$ and $]-2^{t_2-1}, 2^{t_2-1}] \ni r = q_1N_2 - q_2N_1 \mod 2^{t_2}$ $= q_1q_2(p_{2_0} - p_{1_0}) \mod 2^{t_1+t}$ with $|p_{2_0} - p_{1_0}| \leq 2^{t_1}$ and $q_i \leq 2^{\alpha}$. Thanks to the upper-triangular shape of M, the volume of L is easily computed: $\mathrm{Vol} L = K^2 2^{t_2}$. Thus, we can respectively upper-bound and lower-bound $\|\mathbf{v_0}\|^2$ and $\mathrm{Vol} L$ by $2^{2\alpha+1}K^2 + 2^{2t_1+4\alpha}$ and $K^2 2^{t_2}$; a condition on t so that $\mathbf{v_0}$ is smaller than the Gaussian heuristic follows: $2^{2\alpha+1}K^2 + 2^{2t_1+4\alpha} \leq \frac{3}{2\pi e}(K^2 2^{t_2})^{\frac{2}{3}}$. This condition is equivalent to

$$t \geq \frac{3}{2}\left[\log_2(2^{2\alpha+1-\frac{2}{3}t_1}K^{\frac{2}{3}} + 2^{\frac{4}{3}t_1+4\alpha}K^{-\frac{4}{3}}) + \log_2(\frac{2\pi e}{3})\right]$$

and the integer value of K which minimizes the right-hand of this inequality is $K = 2^{\alpha+t_1}$. Hence, under Assumption 1, one can factorize N_1, N_2 in polynomial-time as soon as

$$t \geq 4\alpha + \frac{3}{2}(1+\log_2(\pi e)) \tag{13}$$

A stricter and simpler condition on t is: $t \geq 4\alpha + 7$.

We now inspect when Assumption 1 is not verified, that is we study the possible existence of exceptional short vectors in L that are smaller than $\mathbf{v_0}$. These vectors may appear when there exists small coefficients c_1, c_2 ($< 2^{\alpha}$) such that $c_1N_1 - c_2N_2 \mod 2^{t_2}$ is small (say $\approx 2^{t_2-\gamma}$). In particular, to make easier the analysis, we examine the case when the simple vector $\mathbf{v_1}$ defined with $c_1 = c_2 = 1$ is smaller than $\mathbf{v_0}$. The inequality $\|\mathbf{v_1}\|^2 < \|\mathbf{v_0}\|^2$ is equivalent to $t - \gamma < 2\alpha$. So this inequality is possible only for small t and large γ which can be considered as an exception. In our experiments, these exceptional shorts vectors (and, in particular, simple vectors $\mathbf{v_1}$) almost never appear in the $k = 2$ case with t verifying the bound 13.

The method for $k \geq 3$ is a straightforward generalization of the $k = 2$ case by using the results of section 3. Let's consider the lattice L defined by the rows of the matrix M given by

$$M = \left(\begin{array}{c|c} K\,\mathrm{I}_{k\times k} & C_m \\ \hline 0 & 2^{t_2}\,\mathrm{I}_{m\times m} \end{array}\right)$$

where C_m is the matrix defined in section 3 and formed by the concatenation of $m = \binom{k}{2}$ column vectors of k rows and $I_{k\times k}$ (resp. $I_{m\times m}$) is the identity matrix of size $k\times k$ (resp. $m\times m$). Thus, M is a square upper triangular matrix of size $(m+k)\times(m+k)$ and the volume of the $m+k$-dimensional lattice L is easily computed: $\mathrm{Vol}\,L = K^k 2^{mt_2}$.

The vector

$$\mathbf{v_0} = (q_1K,\dots,q_kK,\underbrace{\dots,r_{(a,b)},\dots}_{\forall\{a,b\}\subset\{1,\dots,k\}})$$

with $r_{(a,b)}$ defined as the unique remainder of $q_aq_b(p_b-p_a) = q_aN_a - q_bN_b$ modulo 2^{t_2} in $]-2^{t_2-1},2^{t_2-1}]$, is clearly a vector of L. As we do above, we search for a condition on the integer t under which $\pm\mathbf{v_0}$ is the shortest vector in L under Assumption 1. The integer K will be set at the end of the analysis to be optimal.

We have $\|\mathbf{v_0}\|^2 = K^2(q_1^2+\cdots+q_k^2) + \sum_{\{a,b\}\subset\{1,\dots,k\}} r_{(a,b)}^2$, that we can bound by $\|\mathbf{v_0}\|^2 \le k2^{2\alpha}K^2 + m2^{2t_1+4\alpha}$. A condition on t, under Assumption 1, follows:

$$k2^{2\alpha}K^2 + m2^{4\alpha+2t_1} \le \frac{m+k}{2\pi e}(K^k2^{mt_2})^{\frac{2}{m+k}}.$$

This condition is equivalent to

$$t \ge \frac{m+k}{2m}\left[\log_2\left(k2^{2\alpha-\frac{2m}{m+k}t_1}K^{\frac{2m}{m+k}} + m2^{4\alpha+\frac{2k}{m+k}t_1}K^{-\frac{2k}{m+k}}\right) + \log_2\left(\frac{2\pi e}{m+k}\right)\right] \tag{14}$$

The value of K which minimizes the right-hand of this inequality is given by the zero of the derivative of the function $K \mapsto k2^{2\alpha-\frac{2m}{m+k}t_1}K^{\frac{2m}{m+k}} + m2^{4\alpha+\frac{2k}{m+k}t_1}K^{-\frac{2k}{m+k}}$. Actually, K is given by the solution of the equation

$$\frac{2mk}{m+k}2^{2\alpha-\frac{2m}{m+k}t_1}K^{\frac{m-k}{m+k}} = \frac{2km}{m+k}2^{4\alpha+\frac{2k}{m+k}t_1}K^{-\frac{m+3k}{m+k}}$$

and thus, after simplification, $K = 2^{\alpha+t_1}$ which is an integer value. A general condition on t becomes

$$t \ge \frac{m+k}{2m}\left[\log_2\left((m+k)2^{2\alpha\frac{2m+k}{m+k}}\right) + \log_2\left(\frac{2\pi e}{m+k}\right)\right]$$

and the general result immediately follows.

Theorem 3. *Let $N_1 = p_1q_1,\dots,N_k = p_kq_k$ be k n-bit RSA moduli, where the q_i's are α-bit primes and the p_i's are primes that all share t bits from the position t_1 to $t_2 = t_1+t$. Under Assumption 1, the N_i's can be factored in time $\mathscr{C}(\frac{k(k+1)}{2},\frac{k(k+1)}{2},n)$, as soon as*

$$t \ge 2\alpha + \frac{2}{k-1}\alpha + \frac{k+1}{2(k-1)}\log_2(2\pi e)$$

As in the case of $k=2$, we inspect the general case $k\ge 3$ for the existence of exceptional vectors $\mathbf{v_1} = (c_1K,\dots,c_kK,\dots,c_iN_i - c_jN_j \mod 2^{t_2},\dots)$ which will disprove Assumption 1, that is, with c_i's ($< 2^\alpha$) and $c_iN_i - cjN_j \mod 2^{t_2}$ small (say $\approx 2^{t_2-\gamma}$).

The condition under which the simple vector $\mathbf{v_1}$ with $c_1 = c_2 = \cdots = c_k = 1$ verify $\|\mathbf{v_1}\|^2 < \|\mathbf{v_0}\|^2$ is given by

$$t - \gamma < \alpha + \frac{1}{2} + \frac{1}{2}\log(\frac{(k+1)2^{2\alpha-1} - 1}{(k-1)}) \approx 2\alpha$$

Thus, as in the case of $k = 2$, for t and α small and γ large enough, this type of simple vectors may appear. Moreover, the degree of liberty for choosing the c_i increases with k, thus, exceptional vectors may appear more frequently when k grows. This fact was observed during our experiments.

Remark 3. During our first experiments, in few cases, our method fails to factor the N_i's. After analysis of the random generation functions used in our code, it turns out that the q_i where randomly generated in the interval $\left]2^{\alpha-1}, 2^{\alpha}\right]$. Thus, the probability that a lot of q_i's have exactly size α is high. If, moreover, α is small enough compared to t_2 ($\alpha < t_2 = t + t_1$), the corresponding $N_i - N_j \mod 2^{t_2}$ may be very small. This could be explained by the following fact: some of the most significant bits (and at least the highest bit) of $N_i \mod 2^{t_2}$ and $N_j \mod 2^{t_2}$ will be a part of the shared bits between the p_i's and thus they cancel themselves in $(N_i - N_j) \mod 2^{t_2}$. Hence, in this case, we have an exceptional short vector in L and our method fails; on the other hand, if one use these moduli then an attacker may use this extra information to easily factor them with another method.

5 Experimental Results

In order to check the validity of Assumption 1 and the quality of our bounds on t, we implemented the methods on Magma 2.15 [1].

5.1 Shared MSBs

We generated many random 1024-bit RSA moduli, for various values of α and t. We observed that the results were similar for other values of n. In the case where $k = 2$, we used the Lagrange reduction to find with certainty a shortest vector of the lattice, and for $3 \leq k \leq 40$ we compared Schnorr-Euchner's algorithm (that provably outputs a shortest vector of the lattice) with LLL (that gives an exponential approximation of a shortest vector). We used only LLL for $k = 80$.

Table 2. Results for $k = 2$ and 1024-bit RSA moduli with shared MSBs

α (bit-size of the q_i's)	Bound of Theorem 1 $t \geq 2\alpha + 3$	Best experimental t
150	303	302
200	403	402
250	503	502
300	603	602

Table 3. Results for $k = 3, 10, 40$ and 1024-bit RSA moduli with shared MSBs

α (bit-size of the q_i's)	Theoretical bound t	Best experimental t using LLL algo.	Best experimental t using Schnorr-Euchner's algo.	Failure rate of Assumption 1
Results for $k = 3$ (Theoretical bound of Theorem 2: $t \geq \frac{3}{2}\alpha + 5.2\ldots$)				
150	231	228	228	0% ($t = 227$)
200	306	303	303	0% ($t = 302$)
250	381	378	378	0% ($t = 377$)
300	456	453	453	0% ($t = 452$)
350	531	528	528	0% ($t = 527$)
400	606	603	603	0% ($t = 602$)
Results for $k = 10$ (Theoretical bound of Theorem 2: $t \geq \frac{10}{9}\alpha + 4.01\ldots$)				
150	171	169	169	0% ($t = 168$)
200	227	225	225	3% ($t = 224$)
250	282	280	280	3% ($t = 279$)
300	338	336	336	1% ($t = 335$)
350	393	391	391	2% ($t = 390$)
400	449	447	447	0% ($t = 446$)
Results for $k = 40$ (Theoretical bound of Theorem 2: $t \geq \frac{40}{39}\alpha + 3.68\ldots$)				
150	158	156	155	2% ($t = 154$)
200	209	208	207	3% ($t = 206$)
250	261	259	258	1% ($t = 257$)
300	312	310	309	1% ($t = 308$)
350	363	362	361	0% ($t = 360$)
400	414	413	412	2% ($t = 411$)

We conducted experiments for $k = 2, 3, 10, 40$ and 80, and for several values for α. For specific values of k, α and t, we said that a test was successful when the first vector of the reduced basis of the lattice was of the form $\pm\mathbf{v_0}$ (that is, it satisfies Assumption 1 in the heuristic case $k \geq 3$). For each k and each α, we generated 100 tests and found experimentally the best (lowest) value of t that had 100% success rate. We compared this experimental value to the bounds we obtained in Theorems 2 and 1. For the first value of t that does not have 100% success rate and for $k \geq 3$, we analyzed the rate of failures due to Assumption 1 not being valid. Note that failures can be of two different kinds: the first possibility is that $\|\mathbf{v_0}\|$ is greater than the Gaussian heuristic, and the second one is that $\|\mathbf{v_0}\|$ is smaller than the Gaussian heuristic yet $\mathbf{v_0}$ is not a shortest vector of the lattice (that is, Assumption 1 does not hold). We wrote down the percentage of the cases where Assumption 1 was not valid among all the cases where $\|\mathbf{v_0}\|$ was smaller than the Gaussian heuristic. These results are shown in tables 2 and 3. Let's take an example. For $k = 10$ and $\alpha = 200$ (second line of the part corresponding to $k = 10$ in table 3), Theorem 2 predicts that $\mathbf{v_0}$ is a shortest vector of the lattice as soon as $t \geq 227$. It turned out that it was always the case as soon as $t \geq 225$, which is better than expected. For $t = 224$, Assumption 1 was not valid in 3% of the cases.

Table 4. Results for $k = 5$ and 1024-bit RSA moduli with shared bits in the middle ($\alpha \in \{99, 100\}$, $t_1 = 20$, theoretical bound $t \geq 254$)

Experimental t	Failure rate of $\|\mathbf{v_0}\| <$ Gaussian heuristic	Failure rate with Schnorr-Euchner's algo.	Failure rate with LLL's algo.
261	0%	0%	0%
260	0%	1%	1%
259	0%	1%	1%
258	0%	1%	0%
257	0%	3%	2%
256	0%	6%	5%
255	0%	17%	10%
254	0%	33%	19%
253	0%	58%	28%
252	2%	90%	58%
251	96%	100%	89%

Let's analyze the results now. In the rigorous case $k = 2$, we observe that the attack consistently goes one bit further with 100% success rate than our bound in Theorem 1.

In all our experiments concerning the heuristic cases $k \geq 3$, we observed that we had 100% success rate (thus, Assumption 1 was always true) when t was within the bound (10) of Theorem 2. That means that Theorem 2 was always true in our experiments. Moreover, we were often able to go a few bits (up to 3) beyond the theoretical bound on t. When the success rate was not 100% (that is, beyond our experimental bounds on t), we found that Assumption 1 was not true in a very limited number of the cases (less than 3%). Finally, up to dimension 80, LLL was always sufficient to find $\mathbf{v_0}$ when t was within the bound of Theorem 2, and Schnorr-Euchner's algorithm allowed us to go one bit further than LLL in dimension 40.

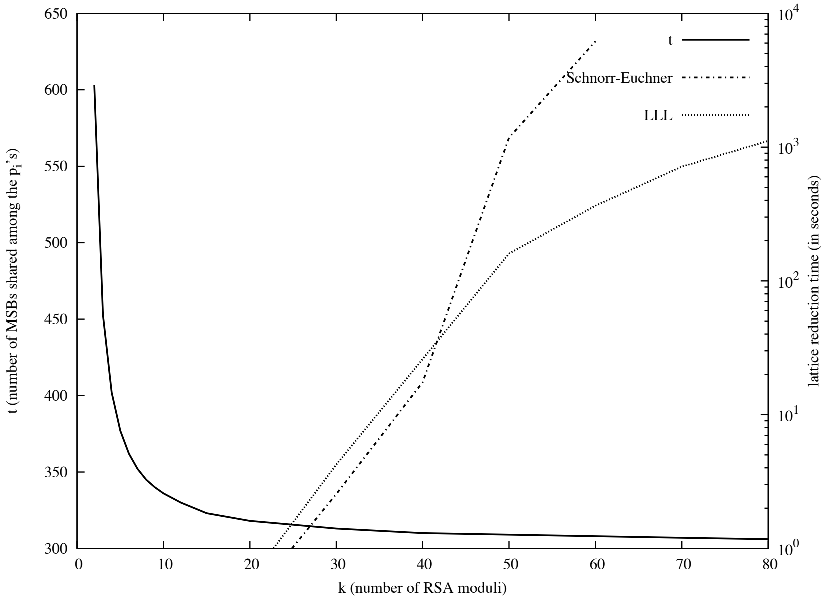
5.2 Shared Bits in the Middle

Contrary to the case of shared MSBs, Assumption 1 may fail when we apply our method with shared bits in the middle (see section 4). When $k = 2$ the phenomenon of exceptional short vectors rarely appeared when t was within the bound of Theorem 3 (less than 1% of failure and did not depend on the position t_1, moreover, we were generally allowed to go 2 or 3 bits further with 90% of success). When $k \geq 3$ it was not still the case. When Schnorr-Euchner's algorithm did not return $\mathbf{v_0}$, we tried to find it in a reduced basis computed by LLL. If neither of these algorithms was able to find $\mathbf{v_0}$ then our method failed. The table 4 shows the result of our experiments for $k = 5$ RSA moduli of size $n = 1024$ and q_i's of size $\alpha \in \{100, 99\}$ (see Remark 3). As one can see, our method can be successfully applied in this case. During these experiments, the failure rate of our method was equal to the failure rate of finding $\mathbf{v_0}$ in a reduced basis computed by LLL. More generally, our experiments showed that for the same size of problems the rate of success is approximately 80% when t was within the bound of Theorem 3 and allowed us to go one or two bits further with success rate ≈50%.

5.3 Efficiency Comparisons

Additionally, we show in table 5 the lowest value of t with 100% success rate and the running-time of LLL and Schnorr-Euchner's algorithm for several values of k (k RSA moduli with p_i's factors sharing t MSBs). For each k, we show the worst running-time we encountered when running 10 tests on an Intel Xeon E5420 at 2.5Ghz. We see that all individual tests completed in less than 1 second for $2 \le k \le 20$. We used Schnorr-Euchner's algorithm up to $k = 60$ where it took at most 6200 seconds. LLL completes under one minute for $20 \le k \le 40$ and in less than 30 minutes for $40 \le k \le 80$.

Table 5. Running time of LLL and Schnorr-Euchner's algorithm, and bound on t as k grows. (Shared MSBs with $\alpha = 300$ and $n = 1024$)



6 Conclusion

In this article we have studied the problem of integers factorization with implicit hints. We have presented new lattice based methods in order to factorize $k \ge 2$ RSA moduli $N_i = p_i q_i$ with polynomial complexity in $\log(N_i)$ when p_i's share unknown MSBs or contiguous bits in the middle. In the case $k = 2$ and shared MSBs, our method is the first one to be completely rigorous. These new results can be seen as an extension of the ones presented in [9] and [15] where, respectively, May and Ritzenhofen gave same type of results in the case where the p_i's share LSBs and Sarkar and Maitra presented heuristic methods based on the Coppersmith's algorithm for finding small roots of polynomials for $k = 2$ moduli with shared MSBs (and/or LSBs) or bits in the middle . Our method gives comparable theoretical results as the one of May and Ritzenhofen and it is more efficient than the Sarkar and Maitra's method.

Whether the method can be applied for $k \geq 3$ N_i's RSA moduli with p_i's sharing MSBs and LSBs remains an open issue. In this case, the problem has much more variables and our method can not be directly applied. One possible way to follow for attacking this problem is to use algebraic techniques, in particular elimination theory, jointly with lattice based methods. This would be an interesting focus for future research.

Acknowledgments

We would like to thank the referees for their valuable comments. We thank Alexander May and Maike Ritzenhofen for their very helpful comments on a draft version of this article and, more particularly, for those which initiated the results of section 4.

References

1. Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system I: The user language. J. Symbolic Comput. 24(3-4), 235–265 (1997); Computational algebra and number theory, London (1993)
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. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
4. Hanrot, G., Stehlé, D.: Improved analysis of Kannan's shortest lattice vector algorithm. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 170–186. Springer, Heidelberg (2007)
5. 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)
6. Kannan, R.: Improved algorithms for integer programming and related lattice problems. In: STOC, pp. 193–206. ACM, New York (1983)
7. Lehmer, D.H., Powers, R.E.: On factoring large numbers. Bulletin of the AMS 37, 770–776 (1931)
8. Lenstra, A.K., Lenstra Jr., H.W.: The development of the number field sieve. Lecture Notes in Mathematics, vol. 1554. Springer, Berlin (1993)
9. May, A., Ritzenhofen, M.: Implicit factoring: On polynomial time factoring given only an implicit hint. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 1–14. Springer, Heidelberg (2009)
10. Micciancio, D., Goldwasser, S.: Complexity of Lattice Problems: a cryptographic perspective. The Kluwer International Series in Engineering and Computer Science, vol. 671. Kluwer Academic Publishers, Boston (2002)
11. Morrison, M.A., Brillhart, J.: A method of factoring and the factorization of F_7. Mathematics of Computation 29(129), 183–205 (1975)
12. Nguyen, P.Q., Stehlé, D.: Floating-point lll revisited. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 215–233. Springer, Heidelberg (2005)
13. Pujol, X., Stehlé, D.: Rigorous and efficient short lattice vectors enumeration. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 390–405. Springer, Heidelberg (2008)
14. Rivest, R.L., Shamir, A.: Efficient factoring based on partial information. In: Pichler, F. (ed.) EUROCRYPT 1985. LNCS, vol. 219, pp. 31–34. Springer, Heidelberg (1986)

15. Sarkar, S., Maitra, S.: Further Results on Implicit Factoring in Polynomial Time. Advances in Mathematics of Communications 3(2), 205–217 (2009)
16. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: FOCS, pp. 124–134. IEEE, Los Alamitos (1994)
17. Vanstone, S.A., Zuccherato, R.J.: Short rsa keys and their generation. J. Cryptology 8(2), 101–114 (1995)

A Common Results on Lattice

An integer lattice L is an additive subgroup of $\mathbb{Z}^n$. Equivalently, it can be defined as the set of all integer linear combinations of d independent vectors $\mathbf{b_1}, \ldots, \mathbf{b_d}$ of $\mathbb{Z}^n$. The integer d is called the *dimension* of L, and $B = (\mathbf{b_1}, \ldots, \mathbf{b_d})$ is one of its *bases*. All the bases of L are related by a unimodular transformation. The *volume* (or *determinant*) of L is the d-dimensional volume of the parallelepiped spanned by the vectors of a basis of L and is equal to the square root of the determinant of the Gramian matrix of B. It does not depend upon the choice of B. We denote it by $\mathrm{Vol}(L)$.

We state (without proofs) common results on lattices that will be used throughout this paper. Readers interested in getting more details and proofs can refer to [10].

Definition 1. *For $1 \leq r \leq d$, let $\lambda_r(L)$ be the least real number such that there exist at least r linearly independent vectors of L of euclidean norm smaller than or equal to $\lambda_r(L)$. We call $\lambda_1(L), \ldots, \lambda_d(L)$ the d* minima *of L, and we call $g(L) = \frac{\lambda_2(L)}{\lambda_1(L)} \geq 1$ the* gap *of L.*

Lemma 4 (Hadamard). *Let $B = (\mathbf{b_1}, \ldots, \mathbf{b_d})$ be a basis of a d-dimensional integer lattice of $\mathbb{Z}^n$. Then the inequality $\prod_{i=1}^d \|\mathbf{b_i}\| \geq Vol(L)$ holds.*

Theorem 4 (Minkowski). *Let L be a d-dimensional lattice of $\mathbb{Z}^n$. Then there exists a non zero vector $\mathbf{v}$ in L which verifies $\|\mathbf{v}\| \leq \sqrt{d}\,\mathrm{Vol}(L)^{\frac{1}{d}}$. An immediate consequence is that $\lambda_1(L) \leq \sqrt{d}\,\mathrm{Vol}(L)^{\frac{1}{d}}$*

Theorem 5 (Lagrange reduction). *Let L be a 2-dimensional lattice of $\mathbb{Z}^n$, given by a basis $B = (\mathbf{b_1}, \mathbf{b_2})$. Then one can compute a Lagrange-reduced basis $B' = (\mathbf{v_1}, \mathbf{v_2})$ of L in time $\mathscr{O}(n \log^2(\max(\|\mathbf{b_1}\|, \|\mathbf{b_2}\|)))$. Besides, it verifies $\|\mathbf{v_1}\| = \lambda_1(L)$ and $\|\mathbf{v_2}\| = \lambda_2(L)$. More information about the running time of the Lagrange reduction may be found in [10].*

Theorem 6 (Kannan's algorithm, see [6,13,4]). *Let L be a d-dimensional lattice of $\mathbb{Z}^n$ given by a basis $(\mathbf{b_1}, \ldots, \mathbf{b_d})$. One can compute a shortest vector of L (with norm equal to $\lambda_1(L)$) in time $\mathscr{O}(\mathscr{P}(\log B, n)\, d^{\frac{d}{2e} + o(d)})$ where $\mathscr{P}$ is a polynomial and $B = \max_i(\|\mathbf{b_i}\|)$. This is done by computing a HKZ-reduced basis of L.*

Theorem 7 (LLL). *Let L be a d-dimensional lattice of $\mathbb{Z}^n$ given by a basis $(\mathbf{b_1}, \ldots, \mathbf{b_d})$. Then LLL algorithm computes a reduced basis $(\mathbf{v_1}, \ldots, \mathbf{v_d})$ that approximates a shortest vector of L within an exponential factor $\|\mathbf{v_1}\| \leq 2^{\frac{d-1}{4}}\,\mathrm{Vol}(L)^{\frac{1}{d}}$. The running time of Nguyen and Stehlé's version is $\mathscr{O}(d^5(d + \log B)\log B)$ where $B = \max_i(\|\mathbf{b_i}\|)$, see [12].*

In practice, LLL algorithm is known to perform much better than expected. It has been experimentally established in [3] that we can expect the bound $\|\mathbf{v_1}\| \leq 1.0219^d \operatorname{Vol}(L)^{\frac{1}{d}}$ on $\|\mathbf{v_1}\|$ on random lattices and that finding a shortest vector of a lattice with gap greater than 1.0219^d should be easy using LLL.

B Exact Computation of the Volume of Lattice L of Section 3

In this section, we compute exactly the volume of the lattice L defined at the beginning of section 3. As a visual example of the construction of this lattice, the reader may take a look at the matrix defined in equation (9) in the case of $k = 4$. We use the notations of section 3.

Lemma 5. *Let L be the lattice whose construction is described at the beginning of section 3. Then its volume is equal to* $\operatorname{Vol}(L) = K\left(K^2 + \sum_{i=1}^{k} N_i^2\right)^{\frac{k-1}{2}}$.

Proof. Let G be the Gramian matrix (of size $k \times k$) of L. Its diagonal terms are $\langle \mathbf{v_i}, \mathbf{v_i} \rangle = K^2 + \sum_{\substack{u=1 \\ u \neq i}}^{k} N_u^2$ and its other terms are: $\langle \mathbf{v_i}, \mathbf{v_j} \rangle = -N_i N_j$. Observe that we can rewrite G as follows $G = \left(K^2 + \sum_{i=1}^{k} N_i^2\right) I_{k\times k} + J$ where $I_{k\times k}$ is the identity matrix of size k and J is the $k \times k$ matrix with terms $-N_i N_j$. If we let χ_J be the characteristic polynomial of J and $\lambda_0 = K^2 + \sum_{i=1}^{k} N_i^2$, we observe that $\det(G) = \chi_J(-\lambda_0)$.

All the columns of J are multiples of $(N_1, N_2, \dots, N_k)^t$. The rank of J is thus 1. The matrix J has therefore the eigenvalue 0 with multiplicity $k-1$. The last eigenvalue is computed using its trace: $\operatorname{Tr}(J) = -\sum_{i=1}^{k} N_i^2$. Therefore, up to a sign $\chi_J(X) = X^{k-1}\left(X + \sum_{i=1}^{k} N_i^2\right)$. We conclude that $\det(G) = \chi_J\left(-K^2 - \sum_{i=1}^{k} N_i^2\right)$, hence $\det(G) = K^2\left(K^2 + \sum_{i=1}^{k} N_i^2\right)^{k-1}$ and $\operatorname{Vol}(L) = \sqrt{\det(G)} = K\left(K^2 + \sum_{i=1}^{k} N_i^2\right)^{\frac{k-1}{2}}$ □

On the Feasibility of Consistent Computations

Sven Laur[1] and Helger Lipmaa[2,3]

[1] University of Tartu, Estonia
[2] Cybernetica AS, Estonia
[3] Tallinn University, Estonia

Abstract. In many practical settings, participants are willing to deviate from the protocol only if they remain undetected. Aumann and Lindell introduced a concept of covert adversaries to formalize this type of corruption. In the current paper, we refine their model to get stronger security guarantees. Namely, we show how to construct protocols, where malicious participants cannot learn anything beyond their intended outputs and honest participants can detect malicious behavior that alters their outputs. As this construction does not protect honest parties from selective protocol failures, a valid corruption complaint can leak a single bit of information about the inputs of honest parties. Importantly, it is often up to the honest party to decide whether to complain or not. This potential leakage is often compensated by gains in efficiency—many standard zero-knowledge proof steps can be omitted. As a concrete practical contribution, we show how to implement consistent versions of several important cryptographic protocols such as oblivious transfer, conditional disclosure of secrets and private inference control.

Keywords: Consistency, equivocal and extractable commitment, oblivious transfer, private inference control.

1 Introduction

Although classical results assure the existence of secure two- and multi-party protocols for any functionality in the presence of malicious adversaries, the computational overhead is often prohibitively large in practice. Hence, cryptographers have sought more restricted models for malicious behavior, which are still realistic but facilitate more efficient protocol construction. A model of covert adversaries [2] proposed by Aumann and Lindell considers a setting, where corrupted parties are unwilling to deviate from the protocol unless they remain uncaught. More precisely, they defined a hierarchy of security models, where malicious behavior that alters the outputs of honest parties is detectable with high probability. However, none of these models guarantee input-privacy because a malicious adversary might potentially issue a detectable attack that completely reveals inputs of all honest parties. We extend their hierarchy with a new security model (*consistent computing*), which guarantees that malicious participants cannot learn anything beyond their intended outputs and honest participants can

Table 1. Comparison of various security objectives in a malicious model

Objective	Input-privacy	Output-privacy	Complaint handling	Detectability
Multi-party protocols				
Security	Yes	Yes	Secure	Optional
Consistency	Limited leaks	Limited leaks	Possible	Optional
K-leakage	Limited leaks	Limited leaks	Possible	No
Covert Model	No	No	Impossible	Partial
Privacy	No	No	Impossible	No
Two-party protocols				
Security	Yes	Yes	Secure	Yes
Consistency	Yes	Yes	Possible	Yes
K-leakage	Limited leaks	Limited leaks	Possible	No
Covert Model	No	No	Impossible	Yes
Privacy	No	No	Impossible	No

detect malicious behavior that alters their outputs. As a result, a valid corruption complaint leaks only *a single bit* of information about the inputs of honest parties as opposed to *the complete disclosure*. Moreover, an honest participant can often decide whether to complain or not. If a complaint is not filed, then no information will be leaked at all unless the adversary learns it indirectly.

Our security model also guarantees that no participant can change their input during a multi-round protocol, which consists of many sub-protocols, i.e., there exists an input that is consistent with all outputs. Additionally, the client can prove cheating to third parties without active participation from the server, since the protocol failure together with a proof that shows correctness of client's actions is sufficient. Hence, our security model is sufficient for many client-server applications, where a server's long-term reputation is more valuable than information revealed by corruption complaints.

Finally, note that the ability to detect cheating from legitimate protocol failures can be important, as well. A good example is private inference control [31], where the client makes queries to the server's database. To protect the server's privacy, certain query patterns are known to be forbidden and should be rejected, though without the server necessarily getting to know which one of the "forbidden" query patterns was used. Hence, a client really needs to know whether the query failed due to insufficient privileges or the server just cheated.

Our Contributions. Our main contribution is the new security model, which provides more strict security guarantees than the semihonest model, all flavors of covert models [2], and the k-leakage model [23] as depicted in Table 1.

We also present concrete, efficient protocols for consistent adaptive oblivious transfer and consistent conditional disclosure of secrets. Notably, all our constructions are much more efficient than their fully secure counterparts. For instance, the new consistent oblivious transfer protocol is secure against unbounded malicious clients, uses 2 messages per query, and has communication and computation comparable to that of the underlying private oblivious transfer protocol. As a main technical tool, we use list commitment schemes, which allow to commit to a list of elements so that, given a short certificate, one can later verify the value of a single element of the committed list. Besides conventional hiding and binding properties, we need equivocality and extractability. See Sect. 3 for details and constructions.

Notation. Throughout this paper, k denotes the security parameter, $\{\mathcal{A}_k\}$ is a shorthand for a non-uniform adversary. The shorthand $t(k) \in \mathrm{poly}(k)$ denotes that $t(k)$ can be bounded by a polynomial and $\varepsilon(k) \in \mathrm{negl}(k)$ means that $\varepsilon(k)$ decreases asymptotically faster than any reciprocal of a polynomial.

Full Version and History. Full version of this paper can be found at [20]. The first version of this eprint from the March of 2006 already defines consistency (although under a different name). The 2-message argument system from [14] was influenced by the first version of the eprint.

2 Definition of Consistent Computations

Achieving security against malicious behavior usually involves a large computational overhead, since one must provide a universal fraud detection mechanism such that honest parties can detect a fraud even if it does not affect their concrete private outputs. As a possible trade-off between efficiency and security, we could protect honest parties only against such actions that alter their outputs. As a result, malicious adversaries might still cause selective protocol failures, where honest parties fail if their inputs are in a specific range. In the following, we use the standard ideal versus real world paradigm to formalize this concept of consistent computations for various protocols. Note that we use standard security definitions [8,18] with modified ideal world implementations, which give additional power to the adversary, see Fig. 1 as an example.

For clarity and brevity, we present the definitions without delving into subtle technical issues. In particular, we have omitted all low-level details of the ideal and real world executions, as these are thoroughly discussed in common reference materials [8,18]. Other more model specific details are separately discussed at the end of the section.

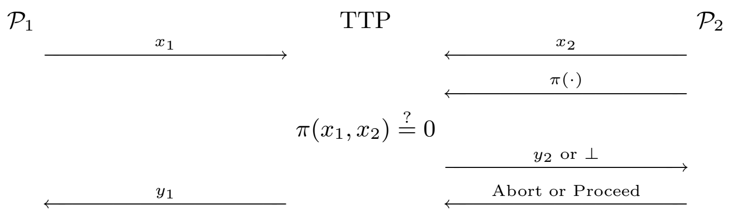


Fig. 1. Ideal world model for consistent two-party computations. A corrupted participant $\mathcal{P}_2$ can cause selective halting by specifying a predicate $\pi(\cdot)$. In the standard model, the dominant party $\mathcal{P}_2$ can cause only a premature abortion.

Idealized Implementations. In an idealized two-party protocol corresponding to consistent computing, both parties send their inputs x_1, x_2 to the trusted third party TTP, which computes the corresponding outputs y_1, y_2. Next, a corrupted participant sends the description of a randomized halting predicate $\pi(\cdot)$ to TTP, who internally computes $\pi(x_1, x_2)$. If $\pi(x_1, x_2) = 1$, then TTP halts the computations and sends $\perp$ to the honest participant. If $\pi(x_1, x_2) = 0$, then TTP sends back the outputs y_i exactly the same way as in the standard ideal model. In particular, the corrupted party can still cause a premature abortion and thus still learn its output.

Generalization to the multi-party setting is straightforward. However, there are two subtle issues connected with *fairness* and *detectability*. A protocol guarantees *fair selective abortion* if an adversary can specify only a single predicate $\pi(\cdot)$ such that TTP halts the computations and sends $\perp$ to all participants iff $\pi(x_1, \ldots, x_n) = 1$. Alternatively, corrupted participants can separately specify different halting predicates $\pi_i(\cdot)$ for each party $\mathcal{P}_i$ and thus some parties might get their outputs while others do not. Also, note that the identity of the malicious coalition might remain hidden for multi-party protocols, whereas this cannot happen in a two-party protocol. A consistent protocol provides *detectability* if TTP sends $\perp$ to $\mathcal{P}_i$ together with the identity of a corrupted participant who specified the halting predicate whenever $\pi_i(x_1, \ldots, x_n) = 1$.

Consistency can also be formalized for adaptive computations, where the outputs of each round can depend on the inputs submitted in previous rounds. For the sake of brevity, we define consistency only for client-server protocols, where the server initially commits to his or her input, and after that the client can issue various oblivious queries. This model covers many practical settings such as selling digital goods and private inference control [1,31]. To start such a protocol, a server sends his or her input x to TTP. After that the client(s) can adaptively issue various queries q_i to TTP. When a query q_i arrives, TTP sends a notification message to the server who can then specify a description of a halting predicate $\pi_i(x_1, \ldots, x_i)$. Next, TTP evaluates the predicate and sends $f(q_i, x)$ back to the client if $\pi_i(q_1, \ldots, q_i) = 0$, otherwise the client receives $\perp$. As a small subtlety, note that the ability to issue halting predicates one-by-one is needed only in the adaptive corruption model, where there are many clients and the server might become corrupted in the middle of computations.

Formal Security Definitin. As usual, we define security of a protocol by comparing its output distribution in the standalone setting with the corresponding output distribution in the ideal world. More formally, fix a security parameter k and let $\mathfrak{D}_k$ denote the input distribution of all parties including the adversary $\mathcal{A}_k$. W.l.o.g. we assume that each input is a pair (ϕ_i, x_i), where the auxiliary input ϕ_i models the internal state of the participant before the protocol and x_i is the actual protocol input. Now, if we fix the exact details how protocols are executed and what the plausible attacks are, then a protocol instance Π_k and an adversary $\mathcal{A}_k$ together determine uniquely a joint output distribution $\textsc{Real}_{\mathfrak{D}_k}(\mathcal{A}_k, \Pi_k)$ of all parties including $\mathcal{A}_k$. Let $\textsc{Ideal}_{\mathfrak{D}_k}(\mathcal{A}_k^\circ, \Pi_k^\circ)$ denote the joint output distribution determined by the ideal world adversary and the corresponding ideal world implementation Π_k°. We say that the protocol family $\{\Pi_k\}$ *securely implements* $\{\Pi_k^\circ\}$ if for any non-uniform polynomial-time adversary $\{\mathcal{A}_k\}$ there exists a non-uniform polynomial-time adversary $\{\mathcal{A}_k^\circ\}$ such that for any input distribution family $\{\mathfrak{D}_k\}$, the output distributions $\textsc{Real}_{\mathfrak{D}_k}(\mathcal{A}_k, \Pi_k)$ and $\textsc{Ideal}_{\mathfrak{D}_k}(\mathcal{A}_k^\circ, \Pi_k^\circ)$ are computationally indistinguishable. If the output distributions are statistically indistinguishable or coincide, then we can talk about statistical and perfect security. Finally, a protocol family $\{\Pi_k\}$ *correctly implements* $\{\Pi_k^\circ\}$ if for any input distribution family $\{\mathfrak{D}_k\}$ the output distributions coincide provided that the adversaries remain inactive (corrupt nobody) in both worlds.

Basic Properties. Note that the only difference between the formal definitions of consistency and security in the malicious model is in the description of the ideal world execution. Hence, we can treat a consistent protocol as a secure implementation of a modified functionality that allows explicit specification of halting predicates. As a result, standard composability results carry over and each consistent protocol in a sequential composition can be replaced with the ideal implementation. However, the resulting hybrid protocol does not necessarily correspond to a consistent ideal world execution. For instance, if we execute two client-server protocols in a row, then the server's input is not guaranteed to be the same for both ideal implementations. Also, a malicious server can specify two halting predicates instead of a single one.

The main advantage of consistent computations over other weakened security models is an explicit correctness guarantee. By the construction of the idealized model of consistent computations, an honest participant reaches the accepting state iff his or her output is consistent with the inputs submitted in the beginning of the protocol. Hence, a successful protocol run provides consistency guarantees in the real world, as well. Consequently, a non-accepting honest participant can prove without the help of other participants that a malicious attack was carried out. Moreover, any consistent protocol can be augmented with a complaint handling mechanism that reveals nothing beyond the validity of the complaint.

Theorem 1. *Let a protocol family $\{\Pi_k\}$ be a correct and consistent implementation of a functionality $\{\Pi_k^\circ\}$ such that all messages are signed by their creators. Then an honest participant can prove the existence of a malicious attack that alters his or her output without help from others provided that the signature scheme is secure. This proof can be converted to a zero-knowledge proof if the messages received by the honest participant reveal nothing about his or her input.*

Proof (Sketch). For the proof, note that by our security assumptions no participant can forge messages sent by others. Hence, if an honest party reveals his or her input and randomness together with all received messages, then anybody can verify correctness of her computations. Since the protocol implements correctly $\{\Pi_k^\circ\}$, semi-honestly behaving participants cannot cause a non-accepting output. This proof can be converted to a zero-knowledge proof, since it is sufficient to present all received messages and then prove that there exists a valid input and randomness that leads to the non-accepting state. The corresponding statement belongs to an **NP**-language and thus has an efficient zero-knowledge proof. The claim follows as messages in the proof reveal nothing about his or her input. □

Note that the last assumption in Theorem 1 is not a real restriction and can be easily met by using a secure public key cryptosystem. Namely, if all messages are encrypted with public keys of corresponding recipients, then messages leak no information to outside observers but the protocol remains consistent.

However, differently from secure computations, a valid complaint reveals additional information, namely, the adversary learns that the corresponding halting predicate $\pi_i(x_1, \ldots, x_n)$ holds. On the other hand, a honest party does not have to issue a complaint and thus the adversary is *not guaranteed* to learn halting predicates—in some applications, the honest parties can untraceably recover from protocol failures. For all consistent and detectable protocols, such a complaint also reveals the identity of the maliciously behaving participant. Hence, there is a trade-off between the utility of a single bit $\pi_i(x_1, \ldots, x_n)$ and a long-term reputation of a participant. As a result, consistency of computations is an adequate protection mechanism for all settings, where participants are unwilling to cheat if they are caught with high probability or a single bit leakage is much smaller compared to the amount of information revealed by legitimate protocol outputs. For instance, the intended output of privacy-preserving data-aggregation is usually several kilobytes (if not megabytes) long, and therefore the effect of a single bit leakage is likely to be irrelevant.

The same argumentation holds also for consistent protocols without accountability. However, finding the identity of the culprit is difficult in such settings, because everybody must prove the correctness of their actions and the corresponding zero-knowledge proofs can be intractable in practice. Finally, note that the potential damage of a valid complaint depends on the set of possible halting predicates π_i. In Section 6, we study this question explicitly and show how to restrict the class of enforceable predicates.

Relation to Other Security Definitions. The concept of covert corruption is rather old and has been discussed in many contexts. The earliest definitions were given for the multi-party setting [15,9] and only recently modified to work in two-party settings by Aumann and Lindell [2]. In particular, note that the definition of *t-detectability* given in [15] and various definitions of *ε-detectability* given in [2] guarantee only that malicious behavior, which alters the outputs of honest parties, is detected with notable probability. However, none of the definitions limit the amount of information acquired during a successful fraud attempt. Thus, our definition of consistent computations is a natural *strengthening* of these definitions, which also guarantees the privacy of inputs. Another related security notion is the *k-leakage model* [23], where the adversary can learn up to k bits of auxiliary information about the inputs of honest parties. Similarly to consistent computations, the adversary cannot alter outputs without being detected. However, differently from consistent computations, the information is *guaranteed* to reach adversary and such an attack is undetectable. Hence, the k-leakage model provides less strict security guarantees. See Table 1 for a comprehensive summary.

Subtle Details. Note that halting predicates must be efficiently computable. Otherwise, participation in an idealized computation can provide significant gains to the adversary. Hence, we require that for any adversary $\{\mathcal{A}_k\}$, the time needed to evaluate halting predicates is polynomial in the running-time of $\{\mathcal{A}_k\}$.

Also, observe that many important cryptographic protocols are not secure in the strict sense. The problem starts with classical zero-knowledge proofs, for which, we know only how to construct simulators $\mathcal{A}^\circ$ that work in expected polynomial time. However, a model where ideal world adversaries have expected polynomial running time causes many technical and philosophical drawbacks [19]. For instance, we loose sequential composability guarantees. Hence, we use an alternative formalization. A protocol $\{\Pi_k\}$ is secure in a *weak polynomial security model*, if for any time bound $t(k) \in \text{poly}(k)$, for any notable difference $\varepsilon(k) \in \Omega(k^{-c})$, and for any polynomial-time real world adversary $\{\mathcal{A}_k\}$, there exists an polynomial-time ideal world adversary $\{\mathcal{A}_k^\circ\}$ such that no non-uniform distinguisher $\{\mathcal{B}_k\}$ with running-time $t(k)$ that can distinguish $\textsc{Real}_{\mathfrak{D}_k}(\mathcal{A}_k, \Pi_k)$ and $\textsc{Ideal}_{\mathfrak{D}_k}(\mathcal{A}_k^\circ, \Pi_k^\circ)$ with advantage more than $\varepsilon(k)$. This definition has the virtue of being formalized with strict time bounds and thus free of technical issues. In particular, it is sequentially composable and formalizes our knowledge about the reductions as precisely as possible.

3 List Commitment Schemes

To achieve consistency, a corrupted participant must be unable to change his or her input during the protocol without being caught. The latter can be achieved by forcing participants to commit to their inputs. More precisely, we need commitment schemes for lists of elements, such that individual elements can later be decommitted by presenting short certificates. A *list commitment scheme* is a quadruple of probabilistic polynomial-time algorithms $(\mathsf{gen}, \mathsf{com}, \mathsf{cert}, \mathsf{open})$ with the following semantics. The key-generator algorithm $\mathsf{gen}(1^k)$ is used to generate public parameters ck that fix the message space $\mathcal{M}_k$ and the maximal number of list elements $N_k \in \text{poly}(k)$. Given a list $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathcal{M}^n$ with $n \leq N_k$, the commitment algorithm $\mathsf{com}_{\mathsf{ck}}(\boldsymbol{x})$ outputs a pair (c, d) of commitment and decommitment values. The certificate generation algorithm $\mathsf{cert}_{\mathsf{ck}}(d, i)$ returns a partial decommitment value (*certificate*) s_i for the ith element. The verification algorithm $\mathsf{open}_{\mathsf{ck}}(c, s)$ returns either a pair (i, x_i) or $\perp$. It is required that $\mathsf{open}_{\mathsf{ck}}(c, \mathsf{cert}_{\mathsf{ck}}(d, i)) = (i, x_i)$ for all possible values of $\mathsf{ck} \leftarrow \mathsf{gen}(1^k)$ and $(c, d) \leftarrow \mathsf{com}_{\mathsf{ck}}(\boldsymbol{x})$. We now define various (optional) security properties through games that are played between a challenger and a nonuniform adversary.

Binding and Hiding. A list commitment scheme is *computationally binding* if every polynomial-time adversary $\{\mathcal{A}_k\}$ wins the following game with negligible probability:

1. Challenger generates $\mathsf{ck} \leftarrow \mathsf{gen}(1^k)$ and sends ck to $\mathcal{A}_k$.
2. $\mathcal{A}_k$ generates a commitment $\hat{c}$ and two certificates $\widehat{s}_0$ and $\widehat{s}_1$.
3. $\mathcal{A}_k$ wins if the commitment can be opened to different values of x_i. That is, locations coincide $i_0 = i_1$ but $\perp \neq x_0 \neq x_1 \neq \perp$ for the openings $(i_0, x_0) \leftarrow \mathsf{open}_{\mathsf{ck}}(\hat{c}, \widehat{s}_0)$ and $(i_1, x_1) \leftarrow \mathsf{open}_{\mathsf{ck}}(\hat{c}, \widehat{s}_1)$.

A list commitment scheme is *statistically hiding* if for any non-uniform adversary $\mathcal{A}$ the probability that $\mathcal{A}_k$ wins the following game is negligibly close to one half:

1. Challenger generates $\mathsf{ck} \leftarrow \mathsf{gen}(1^k)$ and sends ck to $\mathcal{A}_k$.
2. $\mathcal{A}_k$ sends two lists $\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)} \in \mathcal{M}^n$ with $n \leq N_k$ to the challenger.
3. Challenger generates a random bit $b \leftarrow \{0,1\}$, computes $(c, d) \leftarrow \mathsf{com}_{\mathsf{ck}}(\boldsymbol{x}^{(b)})$ and sends the commitment value c to $\mathcal{A}_k$.
4. In the next phase, $\mathcal{A}_k$ can make a number of oracle queries to $\mathsf{cert}_{\mathsf{ck}}(d, \cdot)$ provided that $x_i^{(0)} = x_i^{(1)}$ for any queried index i.
5. $\mathcal{A}_k$ wins the game if he or she correctly guesses the bit b.

Equivocality. In several proofs, we use simulators that send a fake commitment value $\hat{c}$ to the adversary and then gradually open parts of it according to the instructions sent by TTP. To preserve the closeness of real and simulated executions in such a setting, the commitment scheme must be equivocal. A list commitment scheme lc is *perfectly equivocal* if there exist three additional algorithms gen°, com° and equiv, such that no unbounded adversary $\{\mathcal{A}_k\}$ can distinguish between the following two experiments:

Normal Execution:

1. Challenger generates $\mathsf{ck} \leftarrow \mathsf{gen}(1^k)$ and sends ck to $\mathcal{A}_k$.
2. $\mathcal{A}_k$ sends $\boldsymbol{x} = (x_1, \ldots, x_n)$ to the oracle $\mathcal{O}$ who computes $(c, d) \leftarrow \mathsf{com}_{\mathsf{ck}}(\boldsymbol{x})$, $s_i \leftarrow \mathsf{cert}_{\mathsf{ck}}(d, i)$ and replies with $(c, s_1, \ldots, s_n)$.

Simulated Execution:

1. Challenger generates $(\mathsf{ek}, \mathsf{ck}) \leftarrow \mathsf{gen}^\circ(1^k)$ and sends ck to $\mathcal{A}_k$.
2. The oracle $\hat{\mathcal{O}}$ computes $(\hat{c}, \eta) \leftarrow \mathsf{com}^\circ_{\mathsf{ek}}(n)$ and, given $\boldsymbol{x} = (x_1, \ldots, x_n)$ from $\mathcal{A}_k$, computes $\hat{s}_i \leftarrow \mathsf{equiv}_{\mathsf{ek}}(\hat{c}, \eta, i, x_i)$ and replies with $(\hat{c}, \widehat{s}_1, \ldots, \widehat{s}_n)$.

One can build non-interactive equivocal commitment schemes based on any one-way functions in the common reference string (CRS) model [12]. In the standard model, 3 messages are needed to implement an equivocal commitment scheme. Thus, all subsequent results that use equivocal commitment schemes require at least 3 messages. However, as the initialization phase can be shared between different runs, the round complexity is not a problem in practice.

Extractability. Many commitment schemes have an explicit extraction mechanism such that a person who possesses some extra information sk can open commitments without decommitment value, see for instance [29,13]. These commitments are often used in simulator constructions, where one has to extract inputs for committed values. For obvious reasons, such trapdoors do not exist when the final commitment value is shorter than the length of a committed string.

Buldas and Laur showed that if the creator of a commitment gets no additional information besides the commitment parameters ck, then all committed elements are efficiently extractable given white-box access to the committing algorithm and to the randomness used by it. See the definition of *knowledge-binding* and corresponding proofs in [5]. However, in the context of two- and multi-party computations, an adversary always gets additional inputs and thus we must amend the definition. A list commitment scheme is *white-box extractable* if for any polynomial-time adversary $\{\mathcal{A}_k\}$ there exists a polynomial-time extractor machine $\{\mathcal{K}_{A_k}\}$ such that for any input distribution $\mathfrak{D}_k$ and for any family of advice strings $\{a_k\}$ the adversary $\mathcal{A}_k$ can win the following game with negligible probability. The family of advice strings $\{a_k\}$ in the game models unknown future events, which might help the adversary to open the commitments.

1. Challenger generates $\mathsf{ck} \leftarrow \mathsf{gen}(1^k)$, $\phi \leftarrow \mathfrak{D}_k$ and a new random tape ω.
2. $\mathcal{A}_k$ gets ϕ and ck as inputs and ω as the random tape and outputs a list commitment c together with size n, $(c, n) \leftarrow \mathcal{A}_k(\phi, \mathsf{ck}; \omega)$.
3. $\mathcal{K}_{\mathcal{A}_k}$ gets ϕ, ck and ω as inputs and outputs $(\hat{x}_1, \ldots, \hat{x}_n) \leftarrow \mathcal{K}_{\mathcal{A}_k}(\phi, \mathsf{ck}; \omega)$.
4. Given advice a_k, $\mathcal{A}_k$ outputs certificates $(s_1, \ldots, s_m) \leftarrow \mathcal{A}_k(a_k)$.
5. The adversary wins if $\mathcal{A}_k$ outputs at least one certificate that is consistent with the commitment and that corresponds to a list element, not correctly guessed by the extractor, i.e., if $\exists j : \perp \neq (i, x_*) = \mathsf{open}_{\mathsf{ck}}(c, s_j) \wedge x_* \neq \hat{x}_i$.

Currently it is not know how to construct a non-interactive compressing commitment scheme that is provably white-box extractable.[1] However, if we consider interactive commitment schemes, where a sender executes a zero-knowledge proof of knowledge that he or she knows how to open all elements under the list commitment, we can construct such a knowledge extractor by definition. By using suitable zero-knowledge techniques as detailed in [24], the total communication between the receiver and the sender can be made sublinear, although the computational overhead might be too large for practical applications.

As the security of proofs of knowledge is often defined in a weaker model [3], we also relax other definitions to be compatible. A list commitment scheme is *weakly white-box extractable* if for any polynomial-time adversary $\{\mathcal{A}_k\}$ and an error bound $\varepsilon(k)$, there exists a extractor machine $\{\mathcal{K}_{\mathcal{A}_k}\}$ such that, for any input distribution $\mathfrak{D}_k$ and for any family of advice strings $\{a_k\}$, the adversary $\mathcal{A}_k$ wins the extractability game with a probability at most $\varepsilon(k)$ and the running-time of $\mathcal{K}_{\mathcal{A}_k}$ is at most $\mathrm{O}(\mathrm{poly}(k)/\varepsilon(k))$ times slower than $\mathcal{A}_k$.

Double-Layered Commitments. There are two principally different ways how to construct a list commitment scheme with extractability and equivocality properties. First, one can commit elements individually using ordinary

[1] The results of [5] assure existence of extractors $\mathcal{K}_{\mathcal{A}_K, \mathfrak{D}_k}$ that depend on $\mathfrak{D}_k$.

commitment scheme with these properties, such as [13]. As a result, we get strong extractability guarantees but cannot get beyond linear communication complexity. Alternatively, we can first build a double-layered equivocal commitment scheme, and then add extractability by an interactive proof of knowledge. A *double-layered commitment scheme* dlc is specified by a conventional commitment scheme cs and a list commitment scheme lc. The key-generator procedure $\mathsf{dlc.gen}$ runs both key generation procedures and outputs a pair of resulting public parameters $(\mathsf{ck}_1, \mathsf{ck}_2)$. To commit to $\boldsymbol{x} = (x_1, \ldots, x_n)$, one first computes conventional commitments $(c_i, d_i) \leftarrow \mathsf{cs.com}_{\mathsf{ck}_1}(x_i)$ for $i \in \{1, \ldots, n\}$ and then outputs $(c_*, d_*) \leftarrow \mathsf{lc.com}_{\mathsf{ck}_2}(c_1, \ldots, c_n)$. To decommit x_i one has to first decommit c_i by giving $\mathsf{lc.cert}_{\mathsf{ck}_2}(d_*, i)$ and then also reveal d_i so that the receiver could compute $\mathsf{cs.open}_{\mathsf{ck}_1}(c_i, d_i)$. Other operations are defined analogously.

Theorem 2. *Let* lc *be a binding commitment scheme and* cs *be a conventional commitment scheme. Then* dlc *inherits statistical hiding, perfect hiding; computational binding; statistical equivocality and perfect equivocality from* cs.

Proof. Hiding and binding are evident. For the equivocality, note that given the equivocation key ek for cs, it is possible to use $\mathsf{cs.com}^\circ$ to generate a list $\hat{c}_1, \ldots, \hat{c}_n$ of fake commitments for lower level that can be later opened to any values using the function $\mathsf{cs.equiv}_{\mathsf{ek}}$ and the claim follows. □

The list commitment scheme does not have to be hiding. Hence, we can use hash trees to compress large lists into succinct digests. The corresponding construction is based on a collision-resistant hash function family $\{\mathcal{H}_k\}$ and the length of certificates is known to be of size $\mathrm{O}(k \log n)$. The Pedersen commitment scheme [27] is a good candidate of the conventional commitment scheme, as it is perfectly equivocal in the CRS model and can be easily set up in the standard model. More precisely, the public parameter is a uniformly chosen group element $y \in \langle g \rangle$ and the corresponding equivocality trapdoor is the discrete log of y. As the first option, parameters can be generated jointly by the sender and the receiver by using a secure three-message multiplication protocol to multiply two random group elements. Alternatively, the client may specify y since the Pedersen commitment scheme is perfectly hiding. Then, we lose equivocality unless we are willing to find the discrete logarithm of y in exponential time.

4 Consistent Adaptive Oblivious Transfer

Oblivious transfer protocols are often used as building blocks for complex protocols. In an *adaptive oblivious transfer protocol*, a server has an input database $\boldsymbol{x} = (x_1, \ldots, x_n)$ of ℓ-bit strings and a client can adaptively query up to m elements from this database. The client should learn nothing beyond $x_{q_1}, \ldots, x_{q_m}$ and the the server should learn nothing. In particular, the client should learn $\perp$ if its query is not in the range $\{1, \ldots, n\}$. In the asymptotic setting, all parameters m, n, ℓ must be polynomial in the security parameter k. Two standard security notions for the oblivious transfer protocol in the malicious model are

```
Client's inputs: adaptively chosen indexes $q_1, \dots, q_m$.
Server's inputs: a database $\boldsymbol{x} = (x_1, \dots, x_n)$.
Common inputs: a description of $\mathsf{lc}$ and $\mathsf{ot}$.

TRUSTED SETUP
    If needed, the trusted dealer executes the shared setup phase for $\mathsf{ot}$.
    The trusted dealer broadcasts public parameters $\mathsf{ck} \leftarrow \mathsf{lc.gen}(1^k)$ to everybody.

COMMITMENT PHASE
    The server computes $(c, d) \leftarrow \mathsf{lc.com}_{\mathsf{ck}}(\boldsymbol{x})$ and sends the commitment $(c, n)$ to
    the client. Then the server computes $s_i \leftarrow \mathsf{lc.cert}_{\mathsf{ck}}(d, i)$ for each $i \in \{1, \dots, n\}$,
    and stores the database of partial decommitment values $\boldsymbol{s} \leftarrow (s_1, \dots, s_n)$.

QUERY PHASE. To fetch the $q_i$th element form the database:
    1. The client sends $Q_i \leftarrow \mathsf{ot.query}(q_i)$ to the server.
    2. The server returns $R_i \leftarrow \mathsf{ot.reply}(\boldsymbol{s}, Q_i)$.
    3. The client computes $A_i \leftarrow \mathsf{ot.decode}(q_i, R_i)$ and $(j, x_*) \leftarrow \mathsf{lc.open}_{\mathsf{ck}}(c, A_i)$.
       If $j = q_i$ then the client outputs $x_*$, otherwise the client outputs $\perp$.
```

Protocol 1. The new consistent adaptive oblivious transfer protocol

security and privacy. In brief, private protocols guarantee only that a malicious client cannot learn anything beyond $x_{q_1}, \dots, x_{q_m}$ but do not assure that an honest client indeed learns $x_{q_1}, \dots, x_{q_m}$ if the server is malicious. As such they are inapplicable for many practical applications.

In most adaptive oblivious transfer protocols that are secure in the malicious model, the server first commits to his or her individual database elements, and then at every query helps the client to "decrypt" a single database element, see for example [7,28]. A natural alternative is to use a sublinear-length commitment scheme together with suitable zero-knowledge techniques as detailed in [24]. However, the resulting low-communication protocol is only a theoretical solution with computational overhead that is too large for practical applications.

As a practical solution, we show how to convert any private oblivious transfer protocol into a consistent protocol with low computational and communicational overhead, see Prot. 4. By using protocols [17,22] for oblivious transfer, we get an efficient protocol with almost optimal communication. For the sake of simplicity, we assume that the underlying private 1-out-of-n oblivious transfer protocol ot has 2 moves and is determined by a triple of algorithms ($\mathsf{query}, \mathsf{reply}, \mathsf{decode}$) such that for any $q_i \in \{1, \dots, n\}$ and $\boldsymbol{x} \in \{0, 1\}^{\ell \times n}$, we have $\mathsf{decode}(q_i, \mathsf{reply}(\boldsymbol{x}, \mathsf{query}(q_i))) = x_{q_i}$. This assumption is not a big restriction, since most practical oblivious transfer protocols are in this form, and generalization to multi-round protocols is obvious. As a second simplification, we use a trusted setup phase for generating the public parameters. One can eliminate the need for a trusted dealer by running a secure multiparty protocol, but the explicit use of the trusted setup makes security proofs more modular.

The underlying idea behind the protocol is rather simple. First, the server uses a list commitment scheme lc to commit all inputs $\boldsymbol{x}$. Then the server computes an intermediate database $\boldsymbol{s} = (s_1, \dots, s_n)$ of certificates corresponding to every x_j. In an query phase, the client and the server execute the oblivious transfer protocol ot with the server's input $\boldsymbol{s}$ to fetch the q_jth certificate s_{q_j}. If this value opens a database element x_* that is consistent with the commitment of $\boldsymbol{x}$ and the query q_j, then we output x_*, otherwise we return $\perp$.

Theorem 3. *If the oblivious transfer protocol* ot *is computationally private in the shared setup model and the list commitment scheme* lc *is binding and equivocal, then Prot. 4 is a consistent adaptive m-out-of-n oblivious transfer protocol in the polynomial security model provided that* $n^m \in \mathrm{poly}(k)$*.*

Proof. For the proof, we fix a security parameter k, consider an adversary $\mathcal{A}_k$ and show how to convert it into an ideal world adversary $\mathcal{A}_k^\circ$ such that the output distributions are close enough for any input pair (ϕ_c, ϕ_s).

Security of honest client. Let $\mathcal{A}_k$ be a corrupted server and $\mathcal{C}_k$ an honest client. As the number of potential queries is polynomial, we can construct a black-box extractor $\mathcal{K}^{\mathcal{A}_k,\mathcal{C}}$ that fixes random coins of the client and the malicious server, and makes all n^m queries in order to recover all valid openings $(j, \hat{x}_j)$. As the slowdown is polynomial and the commitment scheme is binding, double openings $\hat{x}_j \neq \hat{x}'_j$ are revealed with negligible probability. Hence, we can use $\mathcal{K}^{\mathcal{A}_k,\mathcal{C}}$ in the construction of ideal world server. By the definition, the oblivious transfer protocol ot is private in the shared setup model if for any adaptively chosen inputs vectors $q = (q_1, \dots, q_m)$ and $\overline{q} = (\overline{q}_1, \dots, \overline{q}_m)$ the output distribution of $\mathcal{A}_k$ is computationally indistinguishable. Hence, we can replace the missing messages in the ideal world by simulating the honest receiver with input $\overline{q} = (1, \dots, 1)$. We can combine these results and consider the following ideal world adversary $\mathcal{A}_k^\circ$:

1. Run the setup phase to obtain public parameters for lc and ot.
2. Choose randomness ω and store $(\hat{x}_1, \dots, \hat{x}_n) \leftarrow \mathcal{K}^{\mathcal{A}_k,\mathcal{C}_k}(\phi_s, \mathsf{ck}; \omega)$.
3. Send $(\hat{x}_1, \dots, \hat{x}_n)$ to TTP and specify halting predicates $\pi_1, \dots, \pi_m$ through the interaction between the client $\mathcal{C}_k(q)$ and the adversary $\mathcal{A}_k(\phi_s, \mathsf{ck}; \omega)$, that is, $\pi_i(q_1, \dots, q_i) = 1$ iff $\mathcal{C}_k$ with input $q_1, \dots, q_i$ obtains $x_{q_i} \neq \perp$.
4. Output whatever $\mathcal{A}_k(\phi_s, \mathsf{ck}; \omega)$ outputs in interaction with $\mathcal{C}_k(\overline{q})$.

Let (ψ_c, ψ_s) denote the outputs of the real execution and $(\psi_c^\circ, \psi_s^\circ)$ the outputs of the ideal execution. W.l.o.g. we can assume that the output of $\mathcal{A}_k$ contains $\phi_s, \mathsf{ck}, \omega$ and thus given the advice ϕ_c we can efficiently compute ψ_c form ψ_s or ψ_s°. Hence, the distributions (ψ_c, ψ_s) and (ψ_c, ψ_s°) must be computationally indistinguishable, or otherwise we can distinguish ψ_s form ψ_s°, which violates the privacy of ot. Now, note that for fixed $(\phi_s, \mathsf{ck}, \omega)$ the corresponding outputs ψ_c and ψ_c° can differ only if the client recovers $x_{q_j} \neq \hat{x}_{q_j}$. As this can happen with negligible probability, the distributions (ψ_c, ψ_s°) and $(\psi_c^\circ, \psi_s^\circ)$ must be computationally indistinguishable and thus also (ψ_c, ψ_s) and $(\psi_c^\circ, \psi_s^\circ)$.

Security of honest server. Since the output of the server in the ideal and real model is empty, only the output of a malicious client $\mathcal{A}_k$ must be analyzed. Consider a hybrid implementation of the protocol, where all instances of ot are replaced with ideal implementations of oblivious transfer protocol with the database $\boldsymbol{s}$. Then as the ot protocol is private in the shared setup model, there exists an adversary $\mathcal{A}_k^*$ such that the output distributions of $\mathcal{A}_k$ and $\mathcal{A}_k^*$ are computationally indistinguishable.

To complete the proof, we construct a true ideal world adversary $\mathcal{A}_k^\circ$ and show that the outputs of $\mathcal{A}_k^\circ$ and $\mathcal{A}_k^*$ are computationally indistinguishable. Indeed, let the ideal world adversary $\mathcal{A}_k^\circ$ proceed as follows:

1. Generate the equivocality key $(\mathsf{ek}, \mathsf{ck}) \leftarrow \mathsf{lc.gen}^\circ(1^k)$ and broadcast ck.
2. Compute $(\hat{c}, \eta) \leftarrow \mathsf{lc.com}^\circ_{\mathsf{ck},\mathsf{ek}}(n)$ and send $\hat{c}$ to the adversary $\mathcal{A}_k^*$.
3. If $\mathcal{A}_k^*$ queries q_j, obtain x_{q_j} from TTP and reply $\hat{s}_j \leftarrow \mathsf{lc.equiv}_{\mathsf{ek}}(\hat{c}, \eta, q_j, x_{q_j})$.
4. Return whatever the adversary $\mathcal{A}_k^*$ finally outputs.

Then it is easy to see that in the hybrid world $\mathcal{A}_k^*$ plays the first equivocality game with the honest server and in the ideal world $\mathcal{A}_k^*$ plays the second equivocality game with a challenger consisting of the simulator $\mathcal{A}_k^\circ$, TTP and the honest server. To nitpick, $\mathcal{A}_k^*$ does not query all faked decommitment values at once, but clearly we can write a wrapper that queries all decommitments and then gradually releases them to $\mathcal{A}_k^*$. Thus, the outputs of $\mathcal{A}_k^*$ and $\mathcal{A}_k^\circ$ must be computationally indistinguishable or otherwise $\mathcal{A}_k^*$ together with the distinguisher would break the equivocality property. □

Corollary 1. *If* ot *is (weakly) statistically server-private and* lc *is statistically equivocal, then Prot. 4 is (weakly) statistically server-private.*

Proof. If ot is statistically private, then for each $\mathcal{A}_k$ there exists poly(k) times slower $\mathcal{A}_k^*$ such that the output distributions are statistically close. Weak statistical privacy guarantees only the existence of $\mathcal{A}_k^*$ without bounds on the running time. Both claims follow as $\mathcal{A}_k^\circ$ is only poly(k) times slower than $\mathcal{A}_k^*$. □

The limitation that the number of potential queries must be polynomial in k seems to be essential for getting a low-communication solution with a small computational overhead. To bypass this restriction, we can either use list commitment schemes that are both extractable and equivocal.

Corollary 2. *If the oblivious transfer protocol* ot *is computationally private in the shared setup model and the list commitment scheme* lc *is (weakly) white-box extractable and equivocal, then Prot. 4 is a consistent adaptive m-out-of-n oblivious transfer protocol in the (weak) polynomial security model.*

Proof. Note that the algorithm pair $(\mathcal{A}_k, \mathcal{C}_k)$ can be treated as a compound adversary, which generates a list commitment (c, n) and then later opens m elements according to the advice $a = (q_1, \ldots, q_m)$. As the commitment scheme is white-box extractable, there exists an extractor machine $\mathcal{K}_{\mathcal{A}_k,\mathcal{C}_k}$ that, given the parameters ck, the server's input ϕ_s and the random tape ω, outputs a list of candidate elements $\hat{\boldsymbol{x}} = (\hat{x}_1, \ldots, \hat{x}_n)$ such that at the end of the execution $\mathcal{C}_k$ accepts $x_{q_j} \neq \hat{x}_{q_j}$ with negligible probability. This extractor can be used in the simulator construction of Thm. 3 instead of $\mathcal{K}^{\mathcal{A}_k,\mathcal{C}_k}$.

Weak extractability. The same construction is valid for a weakly extractable commitment scheme. However, in this case for any notable error bound $\varepsilon(k)$, we can choose $\mathcal{K}_{\mathcal{A}_k,\mathcal{C}_k}$ such that (ψ_c, ψ_s°) and $(\psi_c^\circ, \psi_s^\circ)$ are $\varepsilon(k)$-close. As (ψ_c, ψ_s) and (ψ_c, ψ_s°) are computationally indistinguishable, we can guarantee that, for a large enough k, distributions (ψ_c, ψ_s) and $(\psi_c^\circ, \psi_s^\circ)$ are computationally $2\varepsilon(k)$-close. As the slowdown is $\mathrm{O}(\mathrm{poly}(k)/\varepsilon(k))$, we have established that for any notable error bound $\varepsilon(k)$, we can construct a polynomial-time ideal world adversary, i.e., the correspondence $\{\mathcal{A}_k\} \mapsto \{\mathcal{A}_k^\circ\}$ is valid in the weak polynomial model. □

Comparison with Other Protocols. If n^m is polynomial in k, then we can use very communication efficient list commitments that stretch the input $\mathrm{O}(k \log n)$. By combining it with the most efficient private oblivious transfer protocol [17] we get a protocol with a communication complexity $\mathrm{O}(k \cdot m \log^2 n)$. Moreover, if we neglect the setup, then for the amortized round complexity is two messages per query. The latter is significantly better than the communication complexity $\Omega(mn)$ of the secure adaptive oblivious transfer protocols [11,10,6,25] relying on zero-knowledge proofs. With an explicit use of the PCP theorem one can achieve polylogarithmic communication [24] but this approach is only optimal in the asymptotic sense.

As for the computational complexity, note that additional computational overhead (compared to private protocols) comes from the commitment phase. For a hash tree based list commitment scheme, this computational overhead is $\mathrm{O}(n)$ hashing and commitment operations. If the number of queries is bounded or $n^m \in \mathrm{poly}(k)$, then there are no additional costs besides computing the commitment. If the server must handle an unbounded number of queries, the server has to prove that he or she knows how to open the commitment. In a communication inefficient version proof, the server sends all lower level commitment values $c_1, \ldots, c_n$ to the client and proves knowledge of each decommitment value. The client first checks that the root of the Merkle tree is correct and then verifies individual proofs. Such zero-knowledge proofs are particularly efficient for Petersen commitments. Again the overhead is $\mathrm{O}(n)$ operations. By using suitable conversion methods [24] we can achieve polylogarithmic communication by increasing the computational overhead by a polynomial factor. Although the construction still relies on the PCP theorem, the underlying proof is much simpler—the server does not have to prove correctness in the query phases.

Aumann and Lindell described a 1-out-of-2 oblivious transfer protocol [2], which is secure in the covert model. Although the resulting security guarantees are weaker than for the consistent protocol, see Table 1, their protocol still has 7 messages and a much higher communication complexity. To be fair, three of those messages are used to implement trusted setup for the private oblivious transfer but there are still 4 messages per query and a malicious sender can change its input during the protocol.

5 Consistent Conditional Disclosure of Secrets

Let $\boldsymbol{q} = (q_1, \ldots, q_n)$ denote the client's vector of inputs and let x be a secret possessed by the server. Then *conditional disclosure of secrets* (CDS) for a predicate ρ is a protocol, where the client should learn

$$\mathsf{cds}_\rho(\boldsymbol{q}, x) = \begin{cases} x, & \text{if } \rho(\boldsymbol{q}) = 1 \ , \\ \perp, & \text{otherwise} \ , \end{cases}$$

and the server should learn nothing. CDS protocols are often used to convert client-server protocols secure in a semihonest model to protocols that preserve the privacy of inputs in the malicious model, see [1,21] for the details.

In the context of the current work, we are more interested in the direct application of CDS protocols. Namely, note that a CDS protocol provides a way to distribute a secret only if the client's input satisfies certain condition, i.e., the client has credentials to access the secret. As an example, consider a video on demand service, where a client should obtain a key to a video stream only if his or her balance is non-negative: *credit* > 0. However, the server should be unable to tell the client's exact balance. The CDS protocols described in [1,21] consist of two moves and the client's *query* consists of ciphertexts of $q_1, \ldots, q_n$. As the CDS protocols of say [21] are based on an additively homomorphic cryptosystem, the server can do a limited amount of cryptocomputing to form ciphertexts that decrypt to the secret if the condition ρ is met. Thus, the client must often send some additional encryptions of auxiliary inputs $w_1, \ldots, w_n$ to help the server, i.e., $\boldsymbol{q} = (\mathit{credit}, w_1 \ldots, w_n)$ for our example. Since the solutions [1,21] provide only privacy in the malicious model, it is difficult to prove that the server maliciously declines access and the server cannot easily refute false accusations.

Now consider an extended CDS protocol, where the server first publicly commits to x and the CDS protocol is executed to recover the corresponding decommitment value. As the proof of Thm. 3 and Cor. 2 directly generalizes, the resulting protocol is consistent under the same assumptions. If the set of plausible client inputs is exponential, the exhaustive knowledge extraction technique from Thm. 3 becomes infeasible and the construction, where the server proves that he or she knows how to open the commitment is the only option. The latter is not a big problem, as many conventional commitment schemes have efficient proofs of knowledge for this. For instance, the equivocal Pedersen commitment scheme has this property. Also, note that the server does not have to prove knowledge of the decommitment value to everybody. It is sufficient, if the server proves it to a respected peer during an initialization phase. If we can guarantee that such *auditors* are semihonest, then we can further optimize the proof.

Moreover, Thm. 1 assures that the client can prove that the server acts maliciously to third parties. As anybody can repeat the second phase of a CDS protocol enlisted in [1,21] with a different secret $\overline{x}$, the corresponding honest-verifier zero-knowledge proof is very efficient. The complaining client has to reveal $\overline{x}$ to the prover and then additionally prove (in zero-knowledge if necessary) that the reply of the server is invalid.

The ability to complain makes CDS protocols very appealing in TV or military broadcasts with complex access policy, where credentials are granted by giving out random keys. This problem is commonly known as *private inference control* [31]. In this setting, a server holds a database of private keys that are used to encrypt various content, e.g., documents with different confidentiality levels. Clients have acquired different credentials and the server's task is to release correct keys. For security reasons, the server should not learn which documents are accessed by different clients. At the same time, the server should deny access for clients who do not have appropriate credentials. However, the client should be able to distinguish between denial of service attacks, where the server acts maliciously, and legitimate denials, where the client has no right to obtain a corresponding key. Moreover, to make the service *accountable* against inside attacks the client should be able to prove to third parties that the denial is illegitimate.

We emphasize that the proofs of knowledge can be skipped if it is possible to force the server to construct commitments of keys semi-honestly during the initialization phase either by organizational means or by auditing. As efficient CDS protocols exists for all **NP/poly** predicates [21], we have established that accountable private inference control is possible. More importantly, the solution is really practical if a complaining client is willing to reveal his input.

6 Discussion and Open Problems

Both solutions for oblivious transfer and conditional disclosure of secrets are based on a simple principle: the server first creates a list of possible answers and commits to it. Since all answers are independent of each other and a client can verify that the answer is correct, the server has to prove only that he knows how to decommit and not that all answers are consistent with some server's input. As soon as the answers must satisfy a certain constraint or the client cannot check whether he or she obtained a decommitment value for a correct answer, the construction of a consistent protocol becomes much more complicated.

Nevertheless, any such protocol must give rise to a list commitment scheme. Indeed, we can view any client-server protocol for computing $f(q, x)$ as a compact commitment to a list with elements $x_q = f(q, x)$ where q takes all plausible values. For three-move protocols, the first message is the commitment and the second message together with the third corresponds to interactive opening procedure. The second and third message can be compacted into a single decommitment value provided that a colluding client and server cannot fool third parties who know the first message. As the query should not leak information about other entries, construction of such commitment schemes with implicit correctness guarantees seems a highly non-trivial task. Hence, the question whether one can construct three-move consistent protocols for other tasks is an interesting question, which might shed a light on what type of restrictions are implicitly enforceable by the design of a list commitment scheme.

Another open question is how much can be learned from the complaints and whether is it possible to limit this exposure. By the definition of consistency the complaint leaks an output of a polynomial-time randomized predicate. In practice, we can further restrict the set of enforceable predicates π. For instance, one can force memoryless consistency in the oblivious transfer protocol. Namely, a client-server protocol is *memoryless-consistent* if the halting predicates $\pi_1, \dots, \pi_m$ are independent from previous queries, i.e., $\pi_i(q_1, \dots, q_i) = \pi_i(q_i)$ and the server cannot relate results of different queries.

Theorem 4. *Prot. 4 is memoryless consistent if no instantiations of* ot *protocols share random variables.*

Proof. Assume that an adversary A breaks the memoryless-consistent property of Prot. 4. That is, it can force the client to abort iff a predicate π_i holds on client's queries $(q_1, \dots, q_i)$, where π_i is a non-trivial function of at least two different values q_a and q_b for $a < b \leq i$. Since the protocol is stateless then the adversary can play the role of the client in round $b > a$, to breach the privacy of the client in round a: given its knowledge of whether the client aborted in round b, it will have some advantage in guessing q_a, given the value $\pi_i(q_a, q_b)$. $\square$

Analogous results can be stated for protocols consisting of several CDS protocols. However, memoryless consistency has a certain cost. Many efficient protocols for oblivious transfer [30,1,22] and CDS [1,21] are based on homomorphic encryption. In these protocols, the trusted setup phase assures that the client indeed knows the secret key. This setup phase is replaced with a corresponding proof of knowledge in practice. Now, if each sub-protocol has a different key pair, the preprocessing phase becomes rather complex. Hence, it is beneficial to share the key among many protocol instances, see [21] for further details.

By doing so we loose memoryless consistency and thus a natural question arises: can we still bound the set of enforceable halting predicates. As all of these protocols send the client input in an encrypted form to the server and the replies are also encryptions, it is easy to force affine predicates. Given a list of encryptions $\mathsf{Enc}(q_1), \dots, \mathsf{Enc}(q_\ell)$, the server can multiply all replies with

$$\mathsf{Enc}((q_1\alpha_1 + \cdots + q_i\alpha_i - \beta)r) = (\mathsf{Enc}(q_1)^{\alpha_1} \cdots \mathsf{Enc}(q_i)^{\alpha_i}\mathsf{Enc}(-\beta))^r$$

for a random message space element r. As a result, the replies are unaltered when $q_1\alpha_1 + \cdots + q_i\alpha_i = \beta$ and uniformly distributed otherwise. Consequently, the server can easily force halting predicates corresponding to *affine combinations* of received ciphertexts $\pi_i(q_1, \dots, q_i) = [q_1\alpha_1 + \cdots + q_i\alpha_i = \beta]$. By multiplying replies with several such ciphertexts, the server can also force conjunctions of such affine combinations.

Note that these attacks are applicable for any additively homomorphic encryption scheme. Hence, one can ask whether this is a *complete* description of halting predicates or not. Of course, this question makes sense only for deterministic predicates, as any client server interaction can be formalized as a randomized predicate. For all deterministic predicates, it is reasonable to compare the behavior of a concrete cryptosystem with its idealized counterpart that is implemented through encryption, decryption and ciphertext-addition oracles. We say that a homomorphic cryptosystem has *special cryptocomputing properties* if the malicious server can force deterministic predicates that cannot be forced if the underlying cryptosystem is ideal. As there are cryptosystems that allow to cryptocompute quadratic polynomials [4] and even polynomials of any length [16], cryptosystems with special properties exist. However, in all of these cases these properties follow directly from the design of a cryptosystem. Thus, it is reasonable to assume that standard additively homomorphic cryptosystems, such as Paillier [26], are without special properties and the set of enforceable predicates is limited to affine tests and their conjunctions. Any provable rejection to this fact would be interesting by itself as it would advance the set of cryptocomputable predicates.

Acknowledgments. Both authors were supported by the European Regional Development Fund through the Estonian Centre of Excellence in Computer Science, EXCS. The second author was also supported by Estonian Science Foundation, grant #8058.

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. Aumann, Y., Lindell, Y.: Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 137–156. Springer, Heidelberg (2007)
3. 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)
4. Boneh, D., Goh, E.J., Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)
5. Buldas, A., Laur, S.: Knowledge-Binding Commitments with Applications in Time-Stamping. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 150–165. Springer, Heidelberg (2007)
6. Cachin, C., Camenisch, J.: Optimistic Fair Secure Computation. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 93–111. Springer, Heidelberg (2000)
7. Camenisch, J., Neven, G., Shelat, A.: Simulatable Adaptive Oblivious Transfer. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 573–590. Springer, Heidelberg (2007)
8. Canetti, R.: Security and Composition of Multiparty Cryptographic Protocols. Journal of Cryptology 13(1), 143–202 (2000)
9. Canetti, R., Ostrovsky, R.: Secure Computation with Honest-Looking Parties: What If Nobody Is Truly Honest? In: Proc. of STOC 1999, pp. 255–264. ACM Press, New York (1999)
10. Cramer, R., Damgård, I.: Linear zero-knowledge – a note on efficient zero-knowledge proofs and arguments. In: Proc. of STOC 1997, pp. 436–445. ACM Press, New York (1997)
11. Crépeau, C., van de Graaf, J., Tapp, A.: Committed Oblivious Transfer and Private Multi-Party Computation. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 110–123. Springer, Heidelberg (1995)
12. Crescenzo, G.D., Ishai, Y., Ostrovsky, R.: Non-Interactive and Non-Malleable Commitment. In: Proc. of STOC 1998, pp. 141–150. ACM Press, New York (1998)
13. Di Crescenzo, G.: Equivocable And Extractable Commitment Schemes. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, pp. 74–87. Springer, Heidelberg (2003)

14. Di Crescenzo, G., Lipmaa, H.: Succinct NP Proofs from An Extractability Assumption. In: Beckmann, A., Dimitracopoulos, C., Löwe, B. (eds.) CiE 2008. LNCS, vol. 5028, pp. 175–185. Springer, Heidelberg (2008)
15. Franklin, M.K., Yung, M.: Communication complexity of secure computation. In: Proc. of STOC 1992, pp. 699–710. ACM Press, New York (1992)
16. Gentry, C.: Fully Homomorphic Encryption Using Ideal Lattices. In: Proc. of STOC 2009, pp. 169–178. ACM Press, New York (2009)
17. Gentry, C., Ramzan, Z.: Single-Database Private Information Retrieval with Constant Communication Rate. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 803–815. Springer, Heidelberg (2005)
18. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press, Cambridge (2001)
19. Goldreich, O.: On Expected Probabilistic Polynomial-Time Adversaries: A Suggestion for Restricted Definitions and Their Benefits. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 174–193. Springer, Heidelberg (2007)
20. Laur, S., Lipmaa, H.: On the Feasibility of Consistent Computations. Eprint 2006/088
21. Laur, S., Lipmaa, H.: A New Protocol for Conditional Disclosure of Secrets And Its Applications. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 207–225. Springer, Heidelberg (2007)
22. Lipmaa, H.: An Oblivious Transfer Protocol with Log-Squared Communication. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 314–328. Springer, Heidelberg (2005)
23. Mohassel, P., Franklin, M.K.: Efficiency Tradeoffs for Malicious Two-Party Computation. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 458–473. Springer, Heidelberg (2006)
24. Naor, M., Nissim, K.: Communication Preserving Protocols for Secure Function Evaluation. In: Proc. of STOC 2001, pp. 590–599. ACM Press, New York (2001)
25. Ogata, W., Kurosawa, K.: Oblivious Keyword Search. Journal of Complexity 20(2–3), 356–371 (2004)
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. 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)
28. 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)
29. Santis, A.D., Crescenzo, G.D., Persiano, G.: Necessary and Sufficient Assumptions for Non-iterative Zero-Knowledge Proofs of Knowledge for All NP Relations. In: Welzl, E., Montanari, U., Rolim, J.D.P. (eds.) ICALP 2000. LNCS, vol. 1853, pp. 451–462. Springer, Heidelberg (2000)
30. Stern, J.P.: A New And Efficient All Or Nothing Disclosure of Secrets Protocol. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 357–371. Springer, Heidelberg (1998)
31. Woodruff, D.P., Staddon, J.: Private Inference Control. In: Proc. of ACMCCS 2004, pp. 188–197. ACM Press, New York (2004)

Multi-query Computationally-Private Information Retrieval with Constant Communication Rate

Jens Groth[1], Aggelos Kiayias[2], and Helger Lipmaa[3,4]

[1] University College London, UK
j.groth@ucl.ac.uk
[2] Department of Informatics, University of Athens, Greece
aggelos@di.uoa.gr
[3] Cybernetica AS, Estonia
lipmaa@research.cyber.ee
[4] Tallinn University, Estonia

Abstract. A fundamental privacy problem in the client-server setting is the retrieval of a record from a database maintained by a server so that the computationally bounded server remains oblivious to the index of the record retrieved while the overall communication between the two parties is smaller than the database size. This problem has been extensively studied and is known as computationally private information retrieval (CPIR). In this work we consider a natural extension of this problem: a multi-query CPIR protocol allows a client to extract m records of a database containing n ℓ-bit records. We give an information-theoretic lower bound on the communication of any multi-query information retrieval protocol. We then design an efficient non-trivial multi-query CPIR protocol that matches this lower bound. This means we settle the multi-query CPIR problem optimally up to a constant factor.

Keywords: Computationally private information retrieval, multi-query CPIR, lower bound on communication.

1 Introduction

A (single-server) computationally-private information retrieval (CPIR) protocol enables a client to query a database without revealing which data it is extracting. Several cryptographic techniques based on various computational hardness assumptions have been proposed for CPIR with sublinear communication. In this paper, we go beyond the well-studied single-query case and investigate communication-efficient CPIR protocols for the case where the client has multiple queries. A multi-query CPIR protocol allows a client to extract m records of a database containing n records of ℓ bits each. We give an information-theoretic lower bound on the communication and design a multi-query CPIR protocol that matches this lower bound (up to a constant factor). Our focus in this paper is on the theory of multi-query CPIR giving an asymptotically communication-optimal multi-query CPIR protocol under a reasonable cryptographic assumption; we leave the tasks of optimizing the computational overhead and the constant factor gap between the upper and lower bounds on the communication as open problems.

oblivious transfer (OT) or symmetric CPIR (SCPIR) protocol. A two-message SCPIR protocol is usually required to be secure in the sense of semisimulatability, first defined by Naor and Pinkas [12].

1.1 Background

Private Information Retrieval was introduced in [3]. Kushilevitz and Ostrovsky [9] showed that it is possible to do CPIR with sublinear communication. Cachin, Micali and Stadler [2] gave the first CPIR-protocol for retrieving one bit out a database where the communication complexity is polylogarithmic in the database size. The communication-wise best single-query CPIR protocols up-to-date are by Gentry and Ramzan [6] and Lipmaa [10] that allow the retrieval of an ℓ-bit record from the database.

Turning to our problem, there are three trivial solutions to m-query CPIR: One option is parallel repetition of a single-query CPIR. In the case of repeating Gentry and Ramzan's CPIR [6] this would result in a communication of $\Theta(m \cdot \log n + m \cdot \ell + m \cdot k)$, where k is a security parameter specifying the length of an RSA-modulus. As we will see this is *not* optimal.

Another option is to use a single-query CPIR protocol to fetch one $m\ell$-bit element from an $\binom{n}{m}$-element database. As our lower bound shows, this solution has asymptotically optimal communication $\Theta(m \cdot \log_2(n/m) + m \cdot \ell + k)$ when combined with Gentry and Ramzan's CPIR, but unfortunately increases the server's computation to $\Omega(\binom{n}{m})$, which for many choices of n and m is superpolynomial in the security parameter.

A third option is to transmit the entire database to the client and is inefficient in terms of communication.

Ishai, Kushilevitz, Ostrovsky and Sahai [7] proposed batch-codes for encoding a database over many separate blocks such that a client can extract m records by querying only a smaller number of records from each block. Our solution uses a related strategy and part of this paper consists of encoding the database in separate blocks that can be queried by separate smaller CPIR protocols. The batch-codes by Ishai, Kushilevitz, Ostrovsky and Sahai, however, do not apply directly to our problem. One reason is that their batch-codes are optimized with respect to keeping the total number of records in all the blocks low in order to keep the computational complexity low, whereas our solution actually uses an encoding where the total number of records in all the blocks becomes quite large. Another difference between the works is that they only consider the case where the database and the blocks use the same alphabet, for instance ℓ-bit strings, while we in some instances will encode the database into blocks of records from a different alphabet.

1.2 Our Contribution

We design a computationally efficient two-message multi-query CPIR protocol with $\Theta(m\ell + m \cdot \log_2(n/m) + k)$ bits of communication, where k is a security parameter specifying the size of an RSA modulus. Server computation is dominated by $\Theta(n\ell)$ group operations, where in both cases the constant in big-Θ is reasonably small. The client's privacy is based on a variant of the Φ-hiding assumption [2,6]. In our construction, we use a multi-query CPIR variant of Gentry and Ramzan's single-query CPIR [6] that works for a restricted set of parameters (m, n, ℓ). We present a reduction demonstrating that any multi-query CPIR protocol that works for a restricted set of parameters can be used as a building block to construct a communication-optimal CPIR protocol for *any* set of parameters.

We also prove that any perfectly correct multi-query (non-private) information retrieval protocol has an information theoretical lower bound of $\Omega(m \cdot \log_2(n/m) + m\ell)$ bits of communication. Thus, our proposed multi-query CPIR has optimal communication complexity up to a constant factor.

1.3 Challenges and Techniques

Known techniques suffice for communication-optimal CPIR in the extreme cases, where the number of queries is very small or very large. If $m = \Omega(n)$ the server can send the entire database to the client in the clear, giving a communication complexity of $n\ell = O(m\ell + m\log(n/m))$ bits. If $m = O(1)$ we can invoke Gentry and Ramzan's single-query CPIR m times in parallel to get a communication complexity of $O(\ell + k) = O(m\ell + m\log(n/m) + k)$ bits. We are interested in finding a communication-efficient CPIR protocol for the case where m is in between the two extremes. Indeed, if $m = o(n)$, then downloading the entire database at a cost of $n\ell$ bits would be sub-optimal and when $m = \omega(1)$ simply repeating Gentry and Ramzan's protocol has an additive overhead of $\Omega(mk)$ bits, which would make it a sub-optimal choice.

A first step towards resolving this issue is Gentry and Ramzan's observation [6] that while they focused on the single-query case, it is also possible to get a restricted multi-query CPIR protocol with their techniques. We will use such a restricted multi-query CPIR scheme as a building block in our construction. The restricted protocol is only communication-optimal for certain choices of (m, n, ℓ) though. It encodes the queries as hidden prime-powers, however, when n grows, the size of these primes grows as well. When $\ell = \Omega(\log n)$ or $m = O(n^\epsilon)$ for a constant $\epsilon > 0$ this turns out not to be a problem, but when ℓ is small and $m = \omega(n^\epsilon)$ the increase of the prime size causes a loss of bandwidth of up to a factor $\log n$.

To eliminate the up to a factor $\log n$ overhead in the communication complexity we will encode the database in such a way that it can be split into smaller pieces that can be processed by the restricted multi-query CPIR protocol. One part of this encoding consists of dividing the database into smaller blocks that will be treated separately. With smaller blocks, we need smaller primes in the

Gentry-Ramzan CPIR to specify a particular index of a record and this improves the communication complexity. To spread the queries evenly on the blocks, we first let the client choose a random permutation of the database. To preserve the sublinear communication complexity, the client does this by sending the server a seed for a pseudorandom number generator from which the longer full permutation can be generated.

Another part of our encoding is best explained by an example. Suppose we have a database of 4 one-bit records and the client wants retrieve two records. We can encode the database as a 6-record database containing 2-bit elements for each possible pair of queries $(1, 2)$, $(1, 3)$, $(1, 4)$, $(2, 3)$, $(2, 4)$, $(3, 4)$ the client could have. This encoding increases the size of the database and the size of the records, but reduces the number of queries the client needs to make. When the client's encoding of queries as primes permits the extraction of many bits at a time, then this encoding improves bandwidth since fewer queries are needed. Batch-codes [7] address a related encoding problem, however, as explained in the section on related work neither of their batch-codes suffice for minimizing communication in our scheme.

With respect to the lower bound on communication, the challenge is that we must consider all possible multi-query CPIR protocols. Most known CPIR protocols consist of encoding the queries in a single message that is sent to the server, which information-theoretically leads to a lower bound of $\log_2(\binom{n}{m}) = \Omega(m \cdot \log(m/n))$. Furthermore, obviously m ℓ-bit strings cannot be communicated using less than $m\ell$ bits. For these protocols it is therefore straightforward to get a lower bound of $\Omega(m \cdot \log_2(n/m) + m\ell)$ bits. However, the lower bound also needs to cover the case of multi-query CPIR protocols that work in a different way and may use more rounds. This makes proving the lower bound non-trivial; we do not know of prior work giving such a lower bound even in the single-query case.

1.4 Roadmap

In Sect. 2, we present the necessary preliminaries. In Sect. 3, we prove a lower bound for the communication complexity of multi-query CPIR (even when privacy is not required). In Sect. 4, we construct a basic restricted multi-query CPIR protocol based on Gentry and Ramzan's work [6]. In Sect. 5, we design a new multi-query CPIR protocol for any parameter values.

2 Preliminaries

Notation. All our algorithms take as input a security parameter k. In the following we say a function f is negligible if $f(k) = k^{-\omega(1)}$. We write $f(k) \approx g(k)$ if $|f(k) - g(k)|$ is negligible. We write $(\mathrm{out}_C, \mathrm{out}_D) \leftarrow \langle C(x), D(y) \rangle$ if C on input x and D on input y output respectively out_C and out_D after interacting with each other.

Multi-query CPIR. Consider a database with records $x_1, \ldots, x_n \in \{0, 1\}^\ell$. Informally, a multi-query CPIR protocol is a protocol that allows a client to extract m different records $x_{i_1}, \ldots, x_{i_m}$ from the database, without revealing which records it extracted. Formally, a multi-query CPIR protocol consists of two interactive polynomial time Turing machines C and D that we call respectively the client and the server. Both parties get as input a security parameter k written in unary and additional parameters m, n, ℓ. The server takes as an input n elements $x_1, \ldots, x_n \in \{0,1\}^\ell$. The client takes as an input a set of m different indexes $i_1, \ldots, i_n \in \{1, \ldots, n\}$. (Note that since we are interested in minimal communication in the case of fixed m, we can assume that all m indexes are different.) The client and server interact, and in the end the client outputs $y_1, \ldots, y_m \in \{0,1\}^\ell$ or a special failure symbol $\perp$. (C, D) is a multi-query CPIR protocol if it satisfies the standard correctness and privacy properties as defined below. Intuitively, correctness means that in the case of an honest client and an honest server, the client always retrieves correct elements $x_{i_1}, \ldots, x_{i_m}$. Privacy is defined in the sense of indistinguishability: given two input tuples $\boldsymbol{i^0}$, $\boldsymbol{i^1}$ chosen by a malicious server, the server should not be able to guess which of the two tuples the client actually uses.

Definition 1 (Perfect correctness). *A multi-query CPIR protocol* (C, D) *has perfect correctness if for any* k*, any* $m, n, \ell = \mathrm{poly}(k)$ *and any* $\boldsymbol{i} = (i_1, \ldots, i_n)$ *and* $\boldsymbol{x} = (x_1, \ldots, x_n)$ *with* $i_j \in \{1, \ldots, n\}$ *and* $x_j \in \{0,1\}^\ell$*, we have that if* $(\mathrm{out}_C, \mathrm{out}_D) \leftarrow \langle C(1^k, m, n, \ell, \boldsymbol{i}), D(1^k, m, n, \ell, \boldsymbol{x})\rangle$*, then* $\mathrm{out}_C = (x_{i_1}, \ldots, x_{i_m})$*.*

Definition 2 (Computational privacy). *A multi-query CPIR protocol has computational privacy if for all non-uniform polynomial time adversaries* $\mathcal{A}$ *we have*

$$\Pr\left[\begin{array}{l} b \leftarrow \{0,1\}, (m, n, \ell, \boldsymbol{i^0}, \boldsymbol{i^1}, \mathrm{state}) \leftarrow \mathcal{A}(1^k), \\ (\mathrm{out}_C, \mathrm{out}_\mathcal{A}) \leftarrow \langle C(1^k, m, n, \ell, \boldsymbol{i^b}), \mathcal{A}(\mathrm{state})\rangle : \mathrm{out}_\mathcal{A} = b \end{array}\right] \approx \frac{1}{2},$$

where $m, n, \ell = \mathrm{poly}(k)$ *and* $\boldsymbol{i^j} = (i_1^j, \ldots, i_m^j)$ *with* $1 \le i_1^j < \ldots < i_m^j \le n$*.*

3 Lower Bound on (m, n, ℓ)-CPIR Communication

Let the database contain n records of size ℓ. An (m, n, ℓ) information retrieval is a two-party protocol between a client and a server that enables the client to receive any m out of the n records. In this section we will establish a lower bound for any perfectly correct (m, n, ℓ) information retrieval protocol, private or not. If the protocol consists of the user indicating the desired indices and the server sending the elements of those indices a straightforward lower bound of $\log_2 \binom{n}{m} + m\ell$ bits applies. Establishing that the same lower bound applies also in the general case requires more work. We show that $\Omega(m \cdot \log(n/m) + m \cdot \ell)$ is in fact the lower bound for any perfectly correct information retrieval protocol. The lower bound is information theoretical and holds even when the client and server are computationally unbounded. The lower bound assumes the protocol to have perfect correctness, so any choice of fixed random tapes also gives a perfectly correct protocol, which means it suffices to prove the lower bound for any fixed pair of random tapes. We will therefore in the following without loss of generality assume that the client and server are deterministic.

Denote by X the set of all subsets of n elements of size m and by Y the set of all n-tuples over the alphabet $\Sigma = \{0,1\}^{\ell}$. The output of any (m,n,ℓ) information retrieval protocol belongs to the set Z of m-tuples over Σ. We denote by $f : X \times Y \to Z$ the output of the information retrieval.

Any protocol computing f can be represented by a binary tree (cf. [8]) so that each internal node v is labeled by a function $c_v : X \to \{0,1\}$ or $s_v : Y \to \{0,1\}$. The root of the tree is the initial node of the protocol and an execution involves following a path from root to leaf according to the functions of the nodes. Note that the program of the client is determined by all the $c_v(\cdot)$ functions where the program of the server is determined by all the $s_v(\cdot)$ functions. For the purpose of obtaining the most general lower bound we make no assumptions on the complexity of these functions. Finally, each leaf holds a value $z \in Z = \{0,1\}^{\ell m}$ which is the output of the client.

For any such protocol we define the equivalence relation between two inputs $(x_1,y_1) \sim (x_2,y_2)$ if they lead the protocol to the same output leaf. For each leaf λ there is a different equivalence class R_λ, and the set of all equivalence classes of $\sim$ is thus parameterized by the set of all leaves. It holds that for any λ, the set R_λ is a combinatorial rectangle: $(x_1,y_1) \in R_\lambda, (x_2,y_2) \in R_\lambda$ implies that $(x_1,y_2),(x_2,y_1) \in R_\lambda$. This follows from the fact that the leaves define unique paths from the root and in each node the path the protocol takes only depends on one of the two inputs. A *fooling set* F_z for some $z \in Z$, on the other hand, is a subset of $X \times Y$ for which it holds that for any $(x_1,y_1),(x_2,y_2) \in F_z$ with $f(x_1,y_1) = f(x_2,y_2) = z$ we have that either $f(x_1,y_2) \neq z$, or $f(x_2,y_1) \neq z$. Fooling sets are useful for lower bounds as they can only be covered by as many f-monochromatic rectangles as their cardinality (a rectangle R is f-monochromatic iff $\exists z : (x,y) \in R \Rightarrow f(x,y) = z$). The number of monochromatic rectangles in turn yields a lower bound on the number of leaves in any protocol tree which then implies a lower bound on the tree's height (which is equal to the total communication) [8]. In a nutshell, the number of leaves in the protocol tree for any protocol computing the function f must be at least $\sum_{z\in Z} |F_z|$ where F_z is a fooling set for the output value $z \in Z$.

Lemma 1. *Fix n,m,ℓ. Let $z = (z_1,\dots,z_m)$ be such that $\{z_1,\dots,z_m\} \subsetneq \{0,1\}^{\ell}$. Define $L(z) := \operatorname{lexmin}\left(\{0,1\}^{\ell} \setminus \{z_j\}_{j=1}^{m}\right)$, where the function $\operatorname{lexmin}(A)$ denotes the lexicographically smallest element of the set of strings A. The set*

$$F_z = \left\{(I,y_1,\dots,y_n) \mid I = \{i_1,\dots,i_m\} \subseteq \{1,\dots,n\}, y_{i_j} = z_j, y_{i'} = L(z)\right\}$$

is a fooling set of size $\binom{n}{m}$, where the indexes have the ranges $j = 1,\dots,m$, $i' \in \{1,\dots,n\} \setminus \{i_1,\dots,i_m\}$, and $1 \le i_1 < \dots < i_m \le n$.

Proof. It is obvious that $|F_z| = \binom{n}{m}$ and that any input $(x,y) \in F_z$ satisfies that $f(x,y) = z$. We next show that F_z is a fooling set. Let $(I;y_1,\dots,y_n),(I';y'_1,\dots,y'_n) \in F_z$. Observe that it should be $I \neq I'$, thus there is at least one location in I that is not in I'. Without loss of generality, say $i_1 \in I \setminus I'$. It follows that $f(I; y'_1, \ldots, y'_n) = (z'_1, \ldots, z'_m) \neq z$ since $z'_1 = L(z) \neq z_1$. □

Observe that if $2^\ell > m$ then trivially $\{z_1, \ldots, z_m\} \subsetneq \{0,1\}^\ell$ for any possible output tuple $z \in Z = \{0,1\}^{\ell m}$, i.e., there would be $2^{\ell m}$ possible outputs for which the lemma above applies. On the other hand, when $2^\ell \leq m$ the number of possible outputs for which the lemma applies is at least $(2^\ell - 1)^m$. This follows from the fact that there are at least that many m tuples that ommit a specific ℓ-bitstring. While this lower bound can be made more tight it will be sufficient for our communication complexity argument.

Theorem 1. *Consider parameters n, m, ℓ with $n \geq m$. The communication complexity of any protocol solving the (m, n, ℓ) information retrieval problem is $\Omega(m \cdot \log_2(n/m) + m \cdot \ell)$.*

Proof. Consider first the case $\ell > 1$. There are at least $(2^\ell - 1)^m$ possible outputs of the information retrieval protocol for which the corresponding fooling set has cardinality $\binom{n}{m}$ according to lemma 1. It follows that the communication complexity is lower-bounded by

$$\left\lceil \log_2 \left(\binom{n}{m} \cdot (2^\ell - 1)^m \right) \right\rceil \geq \left\lceil \log_2 \binom{n}{m} + m \cdot \log_2(2^\ell - 1) \right\rceil$$
$$\geq \left\lceil \log_2 \binom{n}{m} \right\rceil + m(\ell - 1) \ .$$

In order to obtain the asymptotic bound, we use the fact $\binom{tm}{m} \geq t^m$ for any $t, m \geq 1$ and by setting $t = n/m$ we obtain the statement of the theorem. Next, consider the case $\ell = 1$ and $2m < n$. In this case, there are 2 choices where the fooling set has cardinality $\binom{n}{m}$. It follows that the communication complexity would be at least $\log_2 \binom{n}{m} = \Omega(m \cdot \log(n/m))$. Finally in case $\ell = 1$ and $2m \geq n$ trivially the communication is at least m bits, from which the statement of the theorem follows. □

4 Restricted Multi-query CPIR

In Sect. 5, we will construct a multi-query CPIR with communication complexity $O(m\ell + m \cdot \log(n/m) + k)$, where k is a security parameter. As a building block, it will use a multi-query CPIR protocol (C, D) with communication complexity k. Since communication is bounded by k, such a CPIR cannot handle all choices of (m, n, ℓ). What we will need from the building block is that it can be used whenever

$$m \leq \frac{\alpha k}{\ell + \log_2 n} \ , \tag{1}$$

for *some* constant $0 < \alpha < 1$.

In this section, we provide a construction of such a building block—that we call a *restricted multi-query CPIR*—based on the previous single-query CPIR of Gentry and Ramzan [6] and their observation that it can be extended to the multi-query setting. The security of the restricted multi-query CPIR relies on the Φ-hiding assumption by Cachin, Micali and Stadler [2]. It is a 2-message multi-query CPIR protocol, so we will describe it by three algorithms $(\mathrm{Query}, \mathrm{Response}, \mathrm{Extract})$ that generate respectively the client's query, the server's response, and finally allow the client to extract the records from the response.

In the following, Π will be a deterministic polynomial-time algorithm that takes n as input and generates n (small) prime numbers. K will be a probabilistic polynomial time key generator that takes as input the security parameter k and an integer $\pi < 2^{2\alpha k}$ with factors in the list generated by Π, where α is a constant parameter. On such an input it generates a triple (G, g, q), such that G is a group with efficiently computable operations, q is a positive integer, and g is an element of this group with $\mathrm{ord}(g) = \pi q$. We require that the description of G and group elements are at most $k/3$ bits each and that K satisfies the following assumption:

Definition 3 (Decision Subgroup Assumption). *There exists some $\alpha \in (0, 1)$ such that for all probabilistic polynomial-time $\mathcal{A}$,*

$$\Pr\left[\begin{matrix} b \leftarrow \{0,1\}, (1^n, \pi_0, \pi_1, \mathrm{state}) \leftarrow \mathcal{A}(1^k), (G, g, q) \leftarrow K(1^k, \pi_b) : \\ \mathcal{A}(G, g, \mathrm{state}) = b \end{matrix}\right] \approx \frac{1}{2} \enspace ,$$

where the adversary outputs positive integers $\pi_0, \pi_1 < 2^{2\alpha k}$ with factors in $\Pi(n)$.

As an example, we may choose $N = PQ$ as a $k/3$-bit RSA modulus, where $P = 2\pi r + 1$ and $Q = 2st + 1$ and r, s, and t are large random positive integers, and select g as a random element of $\mathbb{Z}_N^*$ that satisfies $\mathrm{ord}(g) = \pi q$ for some q with $\gcd(\pi, q) = 1$. When Π generates a set $p_1, \ldots, p_n$ where $2n < p_1 < \cdots < p_n$ it is shown by Gentry and Ramzan [6] that the assumption above reduces to a variant of the Φ-hiding assumption[1]:

$$\Pr\left[\begin{matrix} b \leftarrow \{0,1\}, (1^n, \pi_0, \pi_1, \mathrm{state}) \leftarrow \mathcal{A}(1^k), N \leftarrow \mathrm{RSAK}(1^k, \pi_b) : \\ \mathcal{A}(N, \mathrm{state}) = b \end{matrix}\right] \approx \frac{1}{2} \enspace ,$$

where $\mathcal{A}$ outputs π_0 and π_1 as described above, and RSAK outputs a $k/3$-bit RSA-modulus N. To avoid factorization attacks due to Coppersmith when a large factor of $\phi(N)$ is known, in this instantiation, the parameter α should be appropriately selected. Specifically, due to the fact that when a factor of $\phi(N)$ that is larger than $N^{1/4}$ is known then it is possible to factor N (Coppersmith [5,4], cf. [1] and the related discussion in [6]) we need to choose

$$\alpha \leq 1/25 \enspace .$$

[1] Strictly speaking Gentry and Ramzan only show this for π being a prime power, however, their proof carries over without change to the more general case where π can be a composite number.

Indeed, with this choice we have that $\pi < 2^{2k/25}$ which is smaller than $2^{(k/3-1)/4} \leq N^{1/4}$ as long as $k \geq 75$.

Given (Π, K), we can construct a 2-message (m, n, ℓ)-CPIR following the protocol of Gentry and Ramzan. This restricted (m, n, ℓ)-CPIR protocol works for choices of the parameters that satisfy Eq. (1):

Query: Let $p_1, \ldots, p_n$ be the primes generated by Π. Let $\pi_1, \ldots, \pi_n$ be the smallest prime powers of $p_1, \ldots, p_n$ that are larger than 2^ℓ, i.e., $\pi_i = p_i^{\lceil \ell / \log_2 p_i \rceil}$. Let $i_1, \ldots, i_m$ be different indexes of the elements the client wants to extract from the database. Define $\pi = \prod_{j=1}^{m} \pi_{i_j}$ and run $(G, g, q) \leftarrow K(1^k, \pi)$. Send (G, g) to the server, and store q for later use.

Response: Given a database of ℓ-bit elements $x_1, \ldots, x_n$, use the Chinese remainder theorem to compute x' so $x' \equiv x_i \mod \pi_i$ for $1 \leq i \leq n$ and send $c = g^{x'}$ to the client.

Extract: For each $1 \leq j \leq m$, compute $c_j = c^{q\pi/\pi_{i_j}}$ and $g_j = g^{q\pi/\pi_{i_j}}$, and find x_{i_j} so that $c_j = g_j^{x_{i_j}}$. Output $(x_{i_1}, \ldots, x_{i_m})$.

Observe that the extraction step requires solving m instances of the discrete logarithm problem within the cyclic groups $\langle g_j \rangle$ for $j = 1, \ldots, m$, where the order of each such subgroup is π_{i_j}. Given that π_{i_j} is a power of the prime p_{i_j}, the extraction requires $O(m\sqrt{p_n}(\ell / \log_2 p_1))$ steps using Giant-Step Baby-Step techniques when the user computes the x_i-s as is done in the Pohlig-Hellman algorithm [13]. The server's computation consists of one $\Theta(\ell n)$-bit exponentiation as in [6].

Now that we have given the protocol, let us explain the constraints on the parameters. By definition we have $1 \leq m, n, \ell = k^{O(1)}$ but we need more limiting constraints since we use only k bits of communication. Since π_i are chosen as the smallest prime powers of p_i that are larger than 2^ℓ we have $\pi_i < 2^\ell p_n = 2^{\ell + \log_2 p_n}$. This means $\pi < 2^{m(\ell + \log_2 p_n)}$ so we have $\pi < 2^{2\alpha k}$ whenever

$$m \leq \frac{2\alpha k}{\ell + \log_2 p_n} \ .$$

Using the constraints in the example given by Gentry and Ramzan based on RSA moduli, we may use Π that generates the first n primes larger than $2n$. We can use the following crude bound on the primes $2n < p_1 < \cdots < p_n < 2n^2$ for $n \geq 2$. For $n \geq 2$ we therefore can use the restricted multi-query CPIR protocol whenever Eq. (1) holds.

We observe that when $n = 1$ we do not need any security assumption, since the client only has one choice of index to query and therefore privacy is not a concern. We also remark that if the key generation algorithm has negligible failure probability, we still have computational privacy if the client reveals the indices $i_1, \ldots, i_m$ on key generation failure. This means we can get perfect correctness in the CPIR. In conclusion, we have the following theorem:

Theorem 2. *If the decision subgroup assumption (definition 3) holds for a constant* $0 < \alpha < 1$*, there exists a 2-message multi-query CPIR with perfect* *correctness, computational privacy and k bits of communication for parameters (m, n, ℓ) satisfying Eq.* (1).

Proof. Follows from discussion above. $\square$

Discussion. Recall that by Eq. (1), $k = \Omega(m \log_2 n + m\ell)$, thus the restricted protocol has communication $\Omega(m \log_2 n + m\ell)$. It may seem that we achieve no gain over the m-times parallel repetition of Gentry-Ramzan's CPIR protocol that has communication $\Theta(m \log_2 n + m\ell + mk)$ for some security parameter k. However, the gain is actually quite significant, especially when $k \gg \log n$.

For example, consider the case $\ell = 1$. Then, m-times repetition of the Gentry-Ramzan protocol gives us a multi-query protocol with communication $m \cdot k$. Now suppose that $m = \sqrt{k}$ and $n = k^{2/3}$. The number of bits used in the transcript is $k^{3/2}$. On the other hand, when we use the restricted multi-query protocol, $\sqrt{k} \leq \frac{\alpha k}{1+\log_2 k}$ and thus we get a protocol with communication k. Thus for large values of m the protocol of this section outperforms the m-times repetition of Gentry-Ramzan's single-query CPIR protocol.

5 Communication-Optimal Perfectly Correct Multi-query CPIR

Our optimal communication reduction of arbitrary multi-query CPIR to the restricted multi-query CPIR will use the restricted CPIR protocol from [6] described above and a pseudorandom number generator PRG. Our transformation operates in four different modes depending on the choice of the parameters (m, n, ℓ). We examine these modes of operation in the following four subsections. The most challenging case is the one that m is relatively large but not as large as to enable the trivial protocol that sends the whole database to be a good solution. We start with the easier cases first.

5.1 Multi-query (m, n, ℓ)-CPIR for Constant n/m

When $n = O(m)$ it is asymptotically communication-optimal to send the entire database to the client. For concreteness, we fix the implicit constant in the big-O notation to be 9 and send the entire database to the client whenever $n \leq 9m$. It is obvious this is a 1-message multi-query CPIR protocol that has perfect correctness and privacy, and optimal communication of $n\ell \leq 9m\ell = O(m\ell)$ bits. In this case, the server does not do any computation except what is needed for the transmission of the database.

5.2 Multi-query (m, n, ℓ)-CPIR for Small m

We will now give a simple extension of the restricted multi-query CPIR that is communication-optimal when $m \leq k^{2/3}$ and $n > 9m$. We do this by chopping the ℓ-bit records into smaller pieces of size e. This gives us $\lceil \ell/e \rceil$ databases containing e-bit strings. We run the restricted multi-query CPIR protocol (C, D) to extract

m records in each of these databases. In order to do this we have to select the parameter e suitably so that the parameter restriction for the restricted CPIR is satisfied.

1. Define $e = \min(\ell, \lfloor \alpha k/m - \log_2 n \rfloor)$.
2. The server splits $(x_1, \ldots, x_n)$ into $\lceil \ell/e \rceil$ databases $\{(x_{h,1}, \ldots, x_{h,n})\}_{h=1}^{\lceil \ell/e \rceil}$, where all $x_{h,i}$ are e-bit strings and x_i is the concatenation of $x_{1,i}, \ldots, x_{\lceil \ell/e \rceil, i}$.
3. The client and server run $\lceil \ell/e \rceil$ restricted multi-query CPIR protocols in parallel for $h \in \{1, \ldots, \lceil \ell/e \rceil\}$: $(x_{h,i_1}, \ldots, x_{h,i_m}) \leftarrow \langle C(1^k, m, n, e, i_1, \ldots, i_m), D(1^k, m, n, e, x_{h,1}, \ldots, x_{h,n}) \rangle$.
4. The client computes $x_{i_1}, \ldots, x_{i_m}$ by concatenating the restricted multi-query CPIR outputs $\{(x_{h,i_1}, \ldots, x_{h,i_m})\}_{h=1}^{\lceil \ell/e \rceil}$ for each index.
5. The client outputs $(x_{i_1}, \ldots, x_{i_m})$.

The above protocol runs in the same number of rounds as the restricted multi-query CPIR protocol. If $\ell \le \lfloor \alpha k/m - \log_2 n \rfloor$ we just need one copy of the restricted protocol, so we get a communication complexity of k bits. If $\ell > \lfloor \alpha k/m - \log_2 n \rfloor$ we get a communication complexity of

$$\lceil \ell / (\lfloor \alpha k/m - \log_2 n \rfloor) \rceil \cdot k < \Big(\frac{\ell}{\alpha k/2m} + 1 \Big) k = \frac{2m\ell}{\alpha} + k = O(k + m\ell) \ ,$$

provided

$$\lfloor \frac{\alpha k}{m} - \log_2 n \rfloor \ge \frac{\alpha k}{2m} \ .$$

The latter condition holds for large enough k, because $m \le k^{2/3}$ and $\log_2 n = O(\log_2 k)$ implies

$$k \ge \frac{2m(1 + \log_2 n)}{\alpha}$$

asymptotically, which in turn implies

$$\frac{\alpha k}{m} - \log_2 n - 1 \ge \frac{\alpha k}{2m} \ .$$

Note that one can further optimize this protocol, since for each h the client's uses the same indices and therefore may choose to use the same initial query every time.

Lemma 2. *The multi-query (m, n, ℓ)-CPIR protocol described above for $m \le k^{2/3}$ is correct and private under the assumption that the underlying restricted CPIR protocol satisfies these properties. Moreover, if the restricted CPIR protocol has perfect correctness the CPIR protocol above has perfect correctness as well.*

Proof. By the choice of e we guarantee that

$$m \le \frac{\alpha k}{e + \log_2 n}$$

as required by Eq. (1). A hybrid argument shows that (perfect) correctness follows from the (perfect) correctness of the restricted multi-query CPIR protocol. Another hybrid argument shows that if an adversary has advantage ϵ in breaking the privacy of the CPIR protocol, then we can break the restricted CPIR protocol with probability

$$\frac{\epsilon}{\lceil \ell/e \rceil} > \frac{\epsilon}{(\ell+1)}$$

where the last inequality holds for values of

$$k \geq \frac{2m(1+\log_2 n)}{\alpha} \enspace .$$

5.3 Multi-query (m,n,ℓ)-CPIR for Large Values of m and $\ell \leq \log_2(n/m)$

We will now consider the case, where $9m < n \wedge k^{2/3} < m \wedge \ell \leq \log_2(n/m)$. Let $b, d \in \mathbb{N}$ be two parameters to be specified below. We split the database into $\lceil \frac{n}{bd} \rceil$ blocks of size bd, and on each of these blocks we will use the restricted multi-query CPIR protocol. Note that if it happens that the clients' queries are evenly distributed then we only need to extract an average of $\frac{mbd}{n}$ records from each of these blocks.

To ensure the uniformity of its queries, the client will choose a seed $s \leftarrow \{0,1\}^k$ for a pseudorandom number generator. From this pseudorandomness seed, the client and the server can generate a pseudorandom permutation of the n elements. From now on we can therefore assume that the client's indices $i_1, \ldots, i_m$ are randomly distributed. Still, we cannot expect that each block has exactly $\frac{mbd}{n}$ records that need to be extracted. We will therefore choose $a = bm/n$ and extract $2ad$ records from each block. We will choose b, d such that ad is large enough to give us negligible probability that the pseudorandom permutation places more than $2ad$ records in any single block.

Recall that the restricted multi-query CPIR lets us extract $2ad$ records from each block provided that, following Eq. (1),

$$2ad \leq \frac{\alpha k}{\ell + \log_2(bd)} \enspace .$$

When ℓ is small, for instance when $\ell = 1$, this means that we need k bits to extract

$$2ad \leq \frac{\alpha k}{\ell + \log_2(bd)} < \frac{\alpha k}{\log_2(bd)}$$

database bits, giving us a non-constant communication rate.

We will get around this problem by using an encoding of the block that gives a more efficient utilization of the bandwidth. The encoding divides each block of size bd into d segments of b records. We then encode each segment by enumerating all possible combinations of a elements that can be drawn from this segment. This gives us a segment of $\binom{b}{a}$ strings of length $a\ell$. On average we desire to extract two $(a\ell)$-bit records from each of the d segments. In reality, the $2ad$ records we need to extract from the block are pseudorandomly distributed on the d segments, but by extracting $3d$ $(a\ell)$-bit strings from the d segments, we are guaranteed to cover any distribution of $2ad$ records in the block. This is an immediate corollary of the following simple counting lemma:

Lemma 3. *Let $a, b, d \in \mathbb{N}$ and let $S_1, \ldots, S_d$ be disjoint sets with $|S_i| = b$. For any $A \subseteq \cup_{i=1}^{d} S_i$ with $|A| = 2ad$, there exists a family of sets $G_1, \ldots, G_t$ such that (i) for each G_j there is some S_i with $G_j \subseteq S_i$, (ii) $|G_j| = a$, (iii) $A \subseteq \cup_{j=1}^{t} G_j$ and (iv) $t \leq 3d$.*

Proof. Let $A_1, \ldots, A_d$ be the partition of A across $S_1, \ldots, S_d$ with $|A_i| = a_i$ and $\sum_{i=1}^{d} a_i = 2ad$. Each A_i can be covered by $\lceil \frac{a_i}{a} \rceil$ subsets of size a from S_i. It follows that we can cover A with a number of sets that equals $\sum_{i=1}^{d} \lceil \frac{a_i}{a} \rceil \leq d + (\sum_{i=1}^{d} a_i)/a = 3d$.

In conclusion, on each block we use the restricted multi-query CPIR to extract $3d$ out of $d \cdot \binom{b}{a}$ possible $(a\ell)$-bit strings. According to Eq. (1), we can use the restricted multi-query CPIR protocol to do this if we choose b, d such that

$$3d \leq \frac{\alpha k}{a\ell + \log_2(d\binom{b}{a})} \quad . \tag{2}$$

Let us now give the constraints we have on the choices of b, d and give a possible choice of variables that gives us optimal communication complexity:

– We want $ad = mb/n \cdot d$ to be so large that there is negligible probability of more than $2ad$ records falling into the same block.
– We need Eq. (2) in order to use the restricted multi-query CPIR protocol.
– Finally, we want $d \cdot \binom{b}{a}$ to be polynomial in k so that the encoded database contains $k^{O(1)}$ elements and hence it is processed in polynomial time in k.

We first use a Chernoff-bound on the probability that for any given bd block there will be more than $2ad$ records that we want to extract. For a fixed bd block, the probability of more than $2ad$ indices needing extraction is smaller than the probability of more than $2ad$ indices ending up in the same block if we allow repetition. The latter probability is $\Pr[X > 2ad]$ where X is a random variable with $X = \sum_{i=1}^{bd} X_i$ and $X_1, \ldots, X_{bd}$ are independent Bernoulli trials with probability $p = m/n$. By using a Chernoff bound we get

$$\Pr[X > 2ad] < e^{-ad/3} \quad . \tag{3}$$

For the latter condition to hold, we choose $a = \lceil \frac{\log k}{\log(n/m)} \rceil$ and $b = a\lceil n/m \rceil$ giving

$$\binom{b}{a} \leq (e\frac{b}{a})^a \leq (e\lceil \frac{n}{m} \rceil)^{\frac{\log k}{\log(n/m)}+1} < e^{\log k + 1} \cdot 2^{\log(n/m) \cdot (\frac{\log k}{\log(n/m)}+1)} = k^{O(1)}.$$

When including $d = k^{O(1)}$ we will therefore have $ad = k^{O(1)}$ so the server will run in polynomial time. We observe for future use that at the same time

$$\binom{b}{a} \geq (\frac{b}{a})^a \geq \lceil \frac{n}{m} \rceil^{\frac{\log k}{\log(n/m)}} \geq \max(k, \frac{n}{m}) \geq k.$$

As we will see the above constraints on the parameters will be sufficient to get an optimal communication complexity. Note that in order to get perfect correctness, the client can check whether indeed all blocks need extraction of at most $2ad$ records. In the unlikely case this is not the case, the client can send the indices that it wants to extract in the clear. This latter protocol is obviously not private, but is only invoked with negligible probability. We have the following protocol construction:

1. Set $a = \lceil \frac{\log k}{\log(n/m)} \rceil$ and $b = a\lceil n/m \rceil$ and $d = \lceil \min(\frac{m}{a}, \frac{\alpha k/4}{a\ell + 2\log\binom{b}{a}}) \rceil$.
2. The client generates a seed $s \leftarrow \{0,1\}^k$ for the pseudorandom generator and checks that $\psi = \text{PRG}(s)$ is a permutation of the indices so at most $2ad$ records need to be extracted from each block of size bd.
3. In the unlikely event ψ does place more than $2ad$ records to be extracted in the same block, the client sends $i_1, \ldots, i_m$ in clear to the server (encoded so it uses approximately $\log\binom{n}{m}$ bits of communication). The server responds with $(x_{i_1}, \ldots, x_{i_m})$, which the client outputs and halts.
4. The client sends s to the server and the server permutes the indices according to $\psi = \text{PRG}(s)$.
5. The server divides the database into blocks of bd consecutive records and encodes each block as a database consisting of $d\binom{b}{a}$ records of length $a\ell$ such that each segment of $\binom{b}{a}$ records contains all possible choices of a ℓ-bit records from the corresponding segment of b records in the block.
6. The client and the server run the restrict multi-query CPIR protocol (C, D) on the $\lceil n/bd \rceil$ encoded blocks of $d\binom{b}{a}$ records to get $3d$ $(a\ell)$-bit strings. This corresponds to extracting the up to $2ad$ records from each of the original blocks.
7. The client decodes the output and reverses the permutation of the indices to get the output $(x_{i_1}, \ldots, x_{i_m})$.

First, the bound on the error probability given in Eq. 3 is negligible as it is bounded by $e^{-ad/3}$ and it holds that $ad = a \cdot \lceil \min(m/a, (\alpha k/4)/(a\ell + 2\log\binom{b}{a})) \rceil > k^{2/3}$ since $m > k^{2/3}$ and $\ell \leq \log(n/m)$ and $\log\binom{b}{a} = \log(k^{O(1)})$.

Regarding communication complexity, let us first compute it when

$$\frac{m}{a} > \frac{\alpha k/4}{a\ell + 2\log\binom{b}{a}}$$

so $d = \lceil (\alpha k/4)/(a\ell + 2\log\binom{b}{a})\rceil$. We send the pseudorandom seed of length k and run the CPIR protocol $\lceil n/bd\rceil$ times for a total communication of

$$\begin{aligned}(\lceil n/bd\rceil + 1)\,k <& \frac{nk}{bd} + 2k \le nk/(b \cdot \frac{\alpha k/4}{a\ell + 2\log_2\binom{b}{a}}) + 2k\\ =& \frac{4n}{\alpha b}\cdot (a\ell + 2\log_2\binom{b}{a}) + 2k \le \frac{20}{\alpha}\cdot\frac{na}{b}\cdot\log_2\frac{n}{m} + 2k\\ \le& \frac{40}{\alpha}\cdot m\log_2\frac{n}{m} + 2k\ ,\end{aligned}$$

where we have used that $a\ell + 2\log\binom{b}{a} \le a\log(n/m) + 2\log((eb/a)^a) \le a\log(n/m) + 2a\log(e\lceil n/m\rceil) \le 5a\log(n/m)$.

Next, we look at the case $d = \lceil m/a\rceil$. We have a communication complexity of

$$(\lceil n/bd\rceil + 1)\,k < \frac{nk}{bd} + 2k \le \frac{nka}{bm} + 2k \le 4k.$$

Also, in the rare cases where the client ends up sending the indices in the clear we have a communication complexity of $\log\binom{n}{m} + m\cdot\ell = O(m\cdot\log_2(n/m) + k)$.

Lemma 4. *The CPIR protocol for $n > 9m, m > k^{2/3}, \ell \le \log_2(n/m)$ is correct and private. It has perfect correctness if the restricted multi-query CPIR protocol has perfect correctness.*

Proof. The protocol is perfectly correct because the restricted CPIR protocol is correct. We just need to verify that the restricted protocol can actually be applied, i.e., for sufficiently large k we have

$$3d \le \frac{\alpha k}{a\ell + \log_2 d\cdot\binom{b}{a}}.$$

To see this holds, observe $d \le k$ because

$$\alpha/4 + \frac{a\ell + 2\log_2\binom{b}{a}}{k} \le 1$$

which follows from $\alpha/4 < 1$ and $a\ell + 2\log_2\binom{b}{a} = O(\log^2 k)$ (for sufficiently large k). From the choice of d in the protocol we now get

$$d \le \left\lceil\frac{\alpha k/4}{a\ell + 2\log\binom{b}{a}}\right\rceil \le \left\lceil\frac{\alpha k/4}{a\ell + \log_2 d\cdot\binom{b}{a}}\right\rceil < \frac{\alpha k/3}{a\ell + \log_2 d\cdot\binom{b}{a}}$$

where the second inequality follows from $d \le k \le \binom{b}{a}$ (for sufficiently large choice of k.)

With the choice of parameters we are guaranteed that the restricted multi-query CPIR of communication complexity k bits can be used on each block of size bd. An adversary with a probability of ϵ of breaking the privacy of the protocol can therefore be converted into an adversary that breaks the restricted multi-query CPIR with probability $\frac{\epsilon}{\lceil n/bd \rceil}$ except for the negligible probability that the privacy breach is due to a bad pseudorandom seed. Similarly, a hybrid argument shows that the multi-query protocol is correct. When the pseudorandom seed is bad, we step down to a non-private but perfectly correct CPIR. Therefore, if the restricted multi-query CPIR has perfect correctness, then we have perfect correctness of our CPIR. □

5.4 Multi-query CPIR for $\ell > \log_2(n/m)$

The final case is where $9m < n \wedge k^{2/3} < m \wedge \ell > \log_2(n/m)$. We split each database record into $\ell' := \lceil \ell / \lceil \log_2(n/m) \rceil \rceil$ records of length $\lceil \log_2(n/m) \rceil$ bits each. We now need to extract $\ell' \cdot m$ out of $\ell' \cdot n$ records of length $\lceil \log_2(n/m) \rceil$. Using the previous construction, we get a multi-query CPIR protocol that can do this with communication complexity $O\left(\ell' m \cdot \log_2\left(\frac{\ell' n}{\ell' m}\right) + k\right) = O(m\ell + k)$.

5.5 Summary: Communication-Optimal Multi-query CPIR

Combining the four protocols, we get a communication-optimal multi-query CPIR:

1. If $n \leq 9m$ send the entire database to the client
2. Else if $m \leq k^{2/3}$ use the CPIR protocol from Section 5.2 with communication complexity $O(m\ell + k)$
3. Else if $\ell \leq \log_2(n/m)$ use the CPIR protocol from Section 5.3 with communication complexity $O(m \cdot \log_2(n/m) + k)$
4. Else if $\ell > \log_2(n/m)$ use the CPIR protocol from Section 5.4 with communication complexity $O(m\ell + k)$

For sufficiently large k this protocol works for all choices of (m, n, ℓ). The communication complexity is $O(m\ell + m \cdot \log_2(n/m) + k)$, which is optimal up to a constant for perfectly correct CPIR. As a corollary to the lemmas in this section, we get the following:

Theorem 3. *The CPIR protocol given above is correct and private. It has perfect correctness if the restricted multi-query CPIR protocol has perfect correctness.*

Acknowledgments. Several reviewers have offered helpful comments on this paper. We would like in particular to thank an anonymous reviewer from ICALP 2009 for a long and insightful review.

The first author was supported by Engineering and Physical Sciences Research Council grant number EP/G013829/1. The second author performed the work at the University of Connecticut, Department of Computer Science and Engineering, partly supported by NSF grants 0447808,0831304,0831306. The third author was supported by Estonian Science Foundation grant #8058 and the European Union through the European Regional Development Fund.

References

1. 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)
2. Cachin, C., Micali, S., Stadler, M.: Computational Private Information Retrieval with Polylogarithmic Communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999)
3. Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M.: Private Information Retrieval. In: 36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, October 23–25, pp. 41–50. IEEE, Los Alamitos (1995)
4. Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In: Maurer [11], pp. 178–189
5. Coppersmith, D.: Finding a Small Root of a Univariate Modular Equation. In: Maurer [11], pp. 155–165
6. Gentry, C., Ramzan, Z.: Single-Database Private Information Retrieval with Constant Communication Rate. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 803–815. Springer, Heidelberg (2005)
7. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Batch codes and their applications. In: Proceedings of the Thirty-Fifth Annual ACM Symposium on the Theory of Computing, Chicago, IL, USA, June 13–16, pp. 262–271. ACM Press, New York (2004)
8. Kushilevitz, E., Nisan, N.: Communication Complexity. Cambridge University Press, Cambridge (1997)
9. Kushilevitz, E., Ostrovsky, R.: Replication is Not Needed: Single Database, Computationally-Private Information Retrieval. In: 38th Annual Symposium on Foundations of Computer Science, Miami Beach, Florida, October 20–22, pp. 364–373. IEEE Computer Society, Los Alamitos (1997)
10. Lipmaa, H.: An Oblivious Transfer Protocol with Log-Squared Communication. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 314–328. Springer, Heidelberg (2005)
11. Maurer, U. (ed.): EUROCRYPT 1996. LNCS, vol. 1070. Springer, Heidelberg (1996)
12. Naor, M., Pinkas, B.: Oblivious Transfer and Polynomial Evaluation. In: Proceedings of the Thirty-First Annual ACM Symposium on the Theory of Computing, Atlanta, Georgia, USA, May 1–4, pp. 245–254. ACM Press, New York (1999)
13. Pohlig, S., Hellman, M.: An Improved Algorithm for Computing Logarithms over GF(p) and Its Cryptographic Significance. IEEE Transactions on Information Theory 24, 106–110 (1978)

Further Observations on Optimistic Fair Exchange Protocols in the Multi-user Setting

Xinyi Huang[1], Yi Mu[2], Willy Susilo[2], Wei Wu[2], and Yang Xiang[3]

[1] School of Information Systems,
Singapore Management University, Singapore
xyhuang@smu.edu.sg

[2] Centre for Computer and Information Security Research,
School of Computer Science and Software Engineering,
University of Wollongong, Australia
{ymu,wsusilo,ww986}@uow.edu.au

[3] School of Information Technology, Deakin University, Australia
yxiang2@gmail.com

Abstract. Recent research has shown that the single-user security of optimistic fair exchange cannot guarantee the multi-user security. This paper investigates the conditions under which the security of optimistic fair exchange in the single-user setting is preserved in the multi-user setting. We first introduce and define a property called "Strong Resolution-Ambiguity". Then we prove that in the certified-key model, an optimistic fair exchange protocol is secure in the multi-user setting if it is secure in the single-user setting and has the property of strong resolution-ambiguity. Finally we provide a new construction of optimistic fair exchange with strong resolution-ambiguity. The new protocol is setup-free, stand-alone and multi-user secure without random oracles.

1 Introduction

In a fair exchange protocol, two parties can exchange their items in a fair way so that no one can gain any advantage in the process. A simple way to realize fair exchange is to introduce an *online* trusted third party who acts as a mediator: earth party sends the item to the trusted third party, who upon verifying the correctness of both items, forwards each item to the other party. A drawback of this approach is that the trusted third party is always involved in the exchange even if both parties are honest and no fault occurs. In practice, the trusted third party could become a bottleneck of the system and is vulnerable to the denial-of-service attack.

Optimistic Fair Exchange (also known as off-line fair exchange) was introduced by Asokan *et al.* [1]. An optimistic fair exchange protocol also needs a third party called "arbitrator", who is not required to be online all the time. Instead, the arbitrator only gets invoked when something goes wrong (e.g., one party attempts to cheat or other faults occur). An optimistic fair exchange protocol involves three participants, namely the signer, the verifier and the arbitrator.

The signer (say, Alice) first issues a verifiable "partial signature" σ' to the verifier (say, Bob). Bob verifies the validity of σ' and fulfills his obligation if σ' is valid. After that, Alice sends Bob a "full signature" σ to complete the transaction. Thus, if no problem occurs, the arbitrator does not participate in the exchange. However, if Bob does not receive the full signature σ from Alice, Bob can send σ' (and the proof of fulfilling his obligation) to the arbitrator, who will convert σ' to σ for Bob.

An optimistic fair exchange protocol can be *setup-driven* or *setup-free* [23]. An optimistic fair exchange protocol is called setup-driven if an initial-key-setup procedure between a signer and the arbitrator is involved. On the other hand, an optimistic fair exchange protocol is called setup-free if the signer does not need to contact the arbitrator, except that the signer can obtain and verify the arbitrator's public key certificate and vice versa. As shown in [10], setup-free is more desirable for the realization of optimistic fair exchange in the multi-user setting. Another notion of optimistic fair exchange is *stand-alone* [23], which requires that the full signature be an ordinary signature.

1.1 Previous Work

As one of the fundamental problems in secure electronic transactions and digital rights management, fair exchange has been studied intensively since its introduction. It is known that optimistic fair exchange can be constructed (in a generic way) using "two signatures" construction [11], verifiably encrypted signature [2,3,8,9,15,20,18], the sequential two-party multisignature (first introduced by Park *et al.* [17], and then broken and repaired by Dodis and Reyzin [11]), the OR-proof [10], and conventional signature and ring signature [14]. In the following, we only review some results which are most relevant to this paper.

Optimistic Fair Exchange in the Single-user Setting
There are three parties involved in an optimistic fair exchange protocol, which are signer(s), verifier(s) and arbitrator(s). Most work about optimistic fair exchange was considered only in the single-user setting, namely there is only one signer. The first formal security model of optimistic fair exchange was proposed in [2,3]. Dodis and Reyzin [11] defined a more generalized and unified model for non-interactive optimistic fair exchange, by introducing a new cryptographic primitive called *verifiably committed signature.* In [11], the security of a verifiably committed signature scheme (equivalently, an optimistic fair exchange protocol) in the single-user setting consists of three aspects: security against the signer, security against the verifier and security against the arbitrator. While the arbitrator is not fully trusted, it is still assumed to be semi-trusted in the sense that the arbitrator will not collude with the signer or the verifier. *In the remainder of this paper, an optimistic fair exchange protocol is single-user secure (or, secure in the single-user setting) means that it is secure in the single-user setting defined in [11].* Notice that their definition does not include all security notions of optimistic fair exchange (e.g., abuse-free [12], non-repudiation [16,21], timely-termination [2,3] and signer-ambiguity [13]), but it does not affect the point we

want to make in this paper. Dodis and Reyzin [11] proposed a stand-alone but setup-driven verifiably committed signature scheme from Gap Diffie-Hellman problem. Constructions of stand-alone and setup-free verifiably committed signature were proposed in [22,23].

Optimistic Fair Exchange in the Multi-user Setting
Recently the security of non-interactive optimistic fair exchange in the multi-user setting was independently studied in [10] and [24]. Optimistic fair exchange in the multi-user setting refers to the scenario where there are two or more signers in the system, but items are still exchanged between two parties. This is different from the multi-party exchange which considers the exchange among three or more parties.

In [10], Dodis, Lee and Yum pointed out that the single-user security of optimistic fair exchange cannot guarantee the multi-user security. They presented a simple counterexample which is secure in the single-user setting but is insecure in a multi-user setting. (In the counterexample, a dishonest verifier in the multi-user setting can obtain a full signature without fulfilling the obligation.) Dodis, Lee and Yum defined the multi-user security model of optimistic fair exchange and provided a generic setup-free construction of optimistic fair exchange secure in the multi-user setting [10]. The security of their construction relies on one-way functions in the random oracle model and trapdoor one-way permutations in the standard model. The analysis in [10] shows that two well-known techniques of optimistic fair exchange (namely, constructions based on verifiably encrypted signatures and sequential two-party signatures) remain secure in the multi-user setting if the underlying primitives satisfy some security notions. Independently, Zhu, Susilo and Mu [24] also demonstrated a verifiably committed signature scheme which is secure in the model defined in [11] but is insecure in the multi-user setting. They defined the security notions of verifiably committed signature in the multi-user setting and proposed a concrete construction of multi-user secure stand-alone and setup-free verifiably committed signature [24]. The non-interactive version of their scheme uses the Fiat-Shamir technique and requires a hash function, which is viewed as the random oracle in security analysis. Due to [10], multi-user secure stand-alone and setup-free optimistic fair exchange protocols without random oracles can be constructed from verifiably encrypted signature schemes without random oracles [15,20,18].

Certified-Key Model and Chosen-Key Model
Most optimistic fair exchange protocols are considered in the *certified-key model* where the user must prove the knowledge of the private key at the key registration phase. Therefore, the adversary is only allowed to make queries about certified public keys. Huang *et al.* [14,13] considered the multi-user security of optimistic fair exchange in the *chosen-key model*, where the adversary can make queries about public keys arbitrarily without requiring to show its knowledge of the corresponding private keys. Optimistic fair exchange protocols secure in the certified-key model may not be secure in the chosen-key model [14].

Huang *et al.* [14] proposed another generic construction for optimistic fair exchange. Their construction can lead to efficient setup-free optimistic fair exchange protocols secure in the standard model and the chosen-key model. Very recently, the first efficient ambiguous optimistic fair exchange protocol was proposed in [13]. The new protocol is proven secure in the multi-user setting and chosen-key model without relying on the random oracle assumption. Without any doubt, it is more desirable if cryptographic protocols can be proven secure in the chosen-key model. However, in this paper, the security of optimistic fair exchange is considered in the certified-key model (as defined in [10]), since certified-key model is reasonable and has been widely used in the research of public key cryptography. *In the remainder of this paper, when we say an optimistic fair exchange protocol is multi-user secure (or, secure in the multi-user setting), it refers that the protocol is secure in the multi-user setting defined in [10] (which is in the certified-key model).*

1.2 Motivation

The research on optimistic fair exchange has shown that:

– The single-user security of optimistic fair exchange does not guarantee the multi-user security [10,24].
– Not all single-user secure optimistic fair exchange protocols are insecure in the multi-user setting [10]. Several single-user secure protocols can be proven secure in the multi-user setting [10].

However, it remains unknown *under which conditions single-user secure optimistic fair exchange protocols will be secure in the multi-user setting*? We believe the investigation of this question not only will provide a further understanding on the security of optimistic fair exchange in the multi-user setting, but also can introduce new constructions of multi-user secure optimistic fair exchange.

1.3 Our Contributions

This paper focuses on both theory investigations and new construction of optimistic fair exchange in the multi-user setting.

1. In Section 3, we introduce and define a new property of optimistic fair exchange, which we call *Strong Resolution-Ambiguity.* Briefly speaking, an optimistic fair exchange protocol has the property of strong resolution-ambiguity if one can transform a partial signature σ' into a full signature σ using signer's private key or arbitrator's private key, and given such a pair (σ', σ), it is infeasible to tell which key is used in the conversion. While there are some optimistic fair exchange protocols satisfying strong resolution-ambiguity, it is the first time this notion is addressed and formally defined.
2. *For an optimistic fair exchange protocol with strong resolution-ambiguity, we prove that its security in the single-user setting is preserved in the multi-user setting.* More precisely, we show that: (1) the security against the signer and the security against the verifier in the single-user setting are preserved in the multi-user setting for optimistic fair exchange protocols with strong

resolution-ambiguity, and (2) the security against the arbitrator in the single-user setting is preserved in the multi-user setting (for optimistic fair exchange protocols either with or without strong resolution-ambiguity).

While strong resolution-ambiguity is not a necessary property for (multi-user secure) optimistic fair exchange protocols, our result provides a new approach for the security analysis of optimistic fair exchange protocols in the multi-user setting: One only needs to analyze the security in the single-user setting (rather than the more complex multi-user setting) for optimistic fair exchange protocols with strong resolution-ambiguity.

3. *In Section 4, we provide a new construction of optimistic fair exchange with strong resolution-ambiguity.* Our construction is a variant of the optimistic fair exchange protocol from the verifiably encrypted signature scheme proposed in [15]. The protocol in [15] has several desirable properties, e.g., setup-free, stand-alone and multi-user secure without random oracles under computational Diffie-Hellman assumption. Our protocol retains all these properties and is more efficient in generating, transmitting and verifying partial signatures. This however is achieved at the cost of larger key size.

2 Definitions of Optimistic Fair Exchange in the Multi-user Setting

This section reviews the syntax and security definitions of optimistic fair exchange in the multi-user setting [10].

2.1 Syntax of Optimistic Fair Exchange

A setup-free non-interactive optimistic fair exchange protocol involves three parties: the signer, the verifier and the arbitrator. It is defined by the following efficient algorithms. An algorithm is called efficient if it is a probabilistic polynomial-time Turing machine.

- $\mathsf{Setup}^{\mathsf{TTP}}$. The arbitrator setup algorithm takes as input a parameter Param, and gives as output a secret arbitration key ASK and a public partial verification key APK.
- $\mathsf{Setup}^{\mathsf{User}}$. The user setup algorithm takes as input Param and (optionally) APK, and gives as output a private signing key SK and a public verification key PK.
- Sig and Ver. These are similar to signing and verification algorithms in an ordinary digital signature scheme.
  - The signing algorithm Sig, run by a signer U_i, takes as input $(m, \mathsf{SK}_{U_i}, \mathsf{APK})$ and gives as output a signature σ_{U_i} on the message m. In fair exchange protocols, signatures generated by Sig are called as *full signatures.*
  - The verification algorithm Ver, run by a verifier, takes as input $(m, \sigma_{U_i}, \mathsf{PK}_{U_i}, \mathsf{APK})$ and returns `valid` or `invalid`. A signature σ_{U_i} is said to be a valid full signature of m under PK_{U_i} if $\mathsf{Ver}(m, \sigma_{U_i}, \mathsf{PK}_{U_i}, \mathsf{APK}) = \texttt{valid}$.

– PSig and PVer. These are partial signing and verification algorithms, where PSig together with Res (which will be defined soon) are functionally equivalent to Sig.
  - The partial signing algorithm PSig, run by a signer U_i, takes as input $(m, \mathsf{SK}_{U_i}, \mathsf{APK})$ and gives as output a signature σ'_{U_i} on m. To distinguish from those produced by Sig, signatures generated by PSig are called as *partial signatures*.
  - The partial verification algorithm PVer, run by a verifier, takes as input $(m, \sigma'_{U_i}, \mathsf{PK}_{U_i}, \mathsf{APK})$ and returns `valid` or `invalid`. A signature σ'_{U_i} is said to be a valid partial signature of m under PK_{U_i} if $\mathsf{PVer}(m, \sigma'_{U_i}, \mathsf{PK}_{U_i}, \mathsf{APK}) = \texttt{valid}$.

– Res. The resolution algorithm Res takes as input a valid partial signature σ'_{U_i} of m under PK_{U_i} and the secret arbitration key ASK, and gives as output a signature σ_{U_i}. This algorithm is run by the arbitrator for a party U_j, who does not receive the full signature from U_i, but possesses a valid partial signature of U_i and a proof that he/she has fulfilled the obligation to U_i.

Correctness. If each signature is generated according to the protocol specification, then it should pass the corresponding verification algorithms. Namely,

1. $\mathsf{Ver}(m, \mathsf{Sig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}), \mathsf{PK}_{U_i}, \mathsf{APK}) = \texttt{valid}$.
2. $\mathsf{PVer}(m, \mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}), \mathsf{PK}_{U_i}, \mathsf{APK}) = \texttt{valid}$.
3. $\mathsf{Ver}(m, \mathsf{Res}(m, \mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}), \mathsf{ASK}, \mathsf{PK}_{U_i}), \mathsf{PK}_{U_i}, \mathsf{APK}) = \texttt{valid}$.

Resolution-Ambiguity [10,11,14,16,24]. Any "resolved signature" $\mathsf{Res}(m, \mathsf{PSig}(m, \mathsf{SK}_{U_i}, \mathsf{APK}), \mathsf{ASK}, \mathsf{PK}_{U_i})$ is (at least computationally) indistinguishable from the "actual signature" $\mathsf{Sig}(m, \mathsf{SK}_{U_i}, \mathsf{APK})$.

Security of Optimistic Fair Exchange. Intuitively, the fairness of an exchange requires that two parties exchange their items in a fair way so that either each party obtains the other's item or neither party does. This requirement consists of the security against signer(s), the security against verifier(s) and the security against the arbitrator, which will be defined by the game between the adversary and the challenger. During the game, the challenger will maintain three initially empty lists: (1) *PK-List* contains the public keys of created users; (2) *PartialSign-List* contains the partial signing queries made by the adversary; and (3) *Resolve-List* contains the resolution queries made by the adversary.

The definitions in the following sections are inspired by those in [10], with modifications which we believe can demonstrate the difference between the single-user security and the multi-user security of optimistic fair exchange.

2.2 Security against Signer(s)

In an optimistic fair exchange protocol, the signer should not be able to generate a valid partial signature which cannot be converted into a valid full signature by the arbitrator. This property is defined by the following game.

– **Setup**. The challenger generates the parameter Param and the arbitrator's key pair $(\mathsf{APK}, \mathsf{ASK})$ by running $\mathsf{Setup}^{\mathsf{TTP}}$. The adversary $\mathcal{A}$ is given Param and APK.
– **Queries**. Proceeding adaptively, $\mathcal{A}$ can make following queries.
Creating-User-Queries. $\mathcal{A}$ can create a user U_i by making a creating-user query (U_i, PK_{U_i}). In order to convince the challenger to accept PK_{U_i} (i.e., add PK_{U_i} to the *PK-List*), $\mathcal{A}$ must prove its knowledge of the legitimate private key SK_{U_i}. This can be realized by requiring the adversary to hand over the private key as suggested in [15], or generate a proof of knowledge [4] of the private key[1].
Resolution-Queries. For a resolution-query $(m, \sigma', \mathsf{PK})$ satisfying $\mathsf{PVer}(m, \sigma', \mathsf{PK}, \mathsf{APK}) = \texttt{valid}$, the challenger first browses *PK-List*. If $\mathsf{PK} \notin$ *PK-List*, an error symbol "$\top$" will be returned to the adversary. Otherwise, the challenger adds (m, PK) to the *Resolve-List* (if the pair (m, PK) is not there) and responds with an output of $\mathsf{Res}(m, \sigma', \mathsf{ASK}, \mathsf{PK})$.
– **Output**. Eventually, $\mathcal{A}$ outputs a triple $(m_f, \sigma'_f, \mathsf{PK}^*)$ and wins the game if $\mathsf{PK}^* \in$ *PK-List*, $\mathsf{PVer}(m_f, \sigma'_f, \mathsf{PK}^*, \mathsf{APK}) = \texttt{valid}$, and $\mathsf{Ver}(m_f, \mathsf{Res}(m_f, \sigma'_f, \mathsf{ASK}, \mathsf{PK}^*), \mathsf{PK}^*, \mathsf{APK}) = \texttt{invalid}$.

Let $\mathsf{Adv\ OFE}_{\mathcal{A}}$ be the probability that $\mathcal{A}$ wins in the above game, taken over the coin tosses made by $\mathcal{A}$ and the challenger. An adversary $\mathcal{A}$ is said to $(t, q_{CU}, q_R, \epsilon)$-break the security against signer(s) if in time t, $\mathcal{A}$ makes at most q_{CU} *Creating-User-Queries*, q_R *Resolution-Queries* and $\mathsf{Adv\ OFE}_{\mathcal{A}}$ is at least ϵ.

Definition 1 (Security against Signer(s)). *An optimistic fair exchange protocol is $(t, q_{CU}, q_R, \epsilon)$-secure against signer(s) if no adversary $(t, q_{CU}, q_R, \epsilon)$-breaks it.*

By setting $q_{CU} = 1$, we can define the security against the signer in the single-user setting, namely an optimistic fair exchange protocol is (t, q_R, ϵ)-secure against the signer in the single-user setting if no adversary $(t, 1, q_R, \epsilon)$-breaks it.

2.3 Security against Verifier(s)

Briefly speaking, the security against verifier(s) requires that the verifier should not be able to generate a valid partial signature of a new message or generate a valid full signature without the assistance from the signer or the arbitrator.

The first requirement is ensured by the security against the arbitrator, namely even the arbitrator (knowing more than the verifier) cannot succeed in that attack. This will be defined shortly in Section 2.4. The second requirement is defined as below.

– **Setup**. The challenger generates the parameter Param and the arbitrator's key pair $(\mathsf{APK}, \mathsf{ASK})$ by running $\mathsf{Setup}^{\mathsf{TTP}}$. The challenger also generates a key pair $(\mathsf{PK}^*, \mathsf{SK}^*)$ by running $\mathsf{Setup}^{\mathsf{User}}$, and adds PK^* to *PK-List*. The adversary $\mathcal{B}$ is given Param, APK and PK^*.

[1] We will use the latter approach in the proof.

– **Queries**. Proceeding adaptively, $\mathcal{B}$ can make all queries defined in Section 2.2 and Partial-Signing-Queries defined as follows.
  Partial-Signing-Queries. For a partial-signing query (m, PK^*), the challenger responds with an output of $\mathsf{PSig}(m, \mathsf{SK}^*, \mathsf{APK})$. After that, (m, PK^*) is added to the *PartialSign-List*. ($\mathcal{B}$ is allowed to make Partial-Signing-Queries only about PK^* as other public keys are created by $\mathcal{B}$.)
– **Output**. Eventually, $\mathcal{B}$ outputs a pair (m_f, σ_f) and wins the game if $(m_f, \mathsf{PK}^*) \notin$ *Resolve-List* and $\mathsf{Ver}(m_f, \sigma_f, \mathsf{PK}^*, \mathsf{APK}) = \texttt{valid}$.

Let $\mathsf{Adv\ OFE}_{\mathcal{B}}$ be the probability that $\mathcal{B}$ wins in the above game, taken over the coin tosses made by $\mathcal{B}$ and the challenger. An adversary $\mathcal{B}$ is said to $(t, q_{CU}, q_{PS}, q_R, \epsilon)$-break the security against verifier(s) if in time t, $\mathcal{B}$ makes at most q_{CU} *Creating-User-Queries*, q_{PS} *Partial-Signing-Queries*, q_R *Resolution-Queries* and $\mathsf{Adv\ OFE}_{\mathcal{B}}$ is at least ϵ.

Definition 2 (Security against Verifier(s)). *An optimistic fair exchange protocol is $(t, q_{CU}, q_{PS}, q_R, \epsilon)$-secure against verifier(s) if no adversary $(t, q_{CU}, q_{PS}, q_R, \epsilon)$-breaks it.*

Similarly, we can obtain the definition of the security against the verifier in the single-user setting, namely an optimistic fair exchange protocol is $(t, q_{PS}, q_R, \epsilon)$-secure against the verifier in the single-user setting if no adversary $(t, 0, q_{PS}, q_R, \epsilon)$-breaks it.

2.4 Security against the Arbitrator

In this section, we will define the security against the arbitrator and prove that the security against the arbitrator in the single-user setting is preserved in the multi-user setting.

The security against the arbitrator requires that the arbitrator, without the partial signature on a message m, should not be able to produce a valid full signature on m[2]. This notion is defined as follows.

– **Setup**. The challenger generates the parameter Param, which is given to the adversary $\mathcal{C}$.
– **Output-I**. $\mathcal{C}$ generates the arbitrator's public key APK and sends it to the challenger. ($\mathcal{C}$ is required to prove the knowledge of the legitimate private key ASK.) In response, the challenger generates a key pair $(\mathsf{PK}^*, \mathsf{SK}^*)$ by running $\mathsf{Setup}^{\mathsf{User}}$ and adds PK^* to *PK-List*. The adversary $\mathcal{C}$ is given PK^*.
– **Queries**. Proceeding adaptively, $\mathcal{C}$ can make *Creating-User-Queries* (defined in Section 2.2) and *Partial-Signing-Queries* (defined in Section 2.3).
– **Output-II**. Eventually, $\mathcal{C}$ outputs a pair (m_f, σ_f) and wins the game if $(m_f, \mathsf{PK}^*) \notin$ *PartialSign-List* and $\mathsf{Ver}(m_f, \sigma_f, \mathsf{PK}^*, \mathsf{APK}) = \texttt{valid}$.

[2] As almost all previous work about optimistic fair exchange, we assume that signer-arbitrator collusion or verifier-arbitrator collusion will not occur. Please refer to [3,11] for discussions of those attacks.

Let $\mathsf{Adv\ OFE}_{\mathcal{C}}$ be the probability that $\mathcal{C}$ wins in the above game, taken over the coin tosses made by $\mathcal{C}$ and the challenger. An adversary $\mathcal{C}$ is said to $(t, q_{CU}, q_{PS}, \epsilon)$-break the security against the arbitrator if in time t, $\mathcal{C}$ makes at most q_{CU} *Creating-User-Queries*, q_{PS} *Partial-Signing-Queries* and $\mathsf{Adv\ OFE}_{\mathcal{C}}$ is at least ϵ.

Remark 1. In the game, the adversary must first generate the arbitrator's public key APK before obtaining PK^* or making other queries. This reflects the definition of optimistic fair exchange as APK could be an input of algorithms $\mathsf{Setup}^{\mathsf{User}}$ and PSig. For concrete protocols where these algorithms do not require APK as the input, the adversary can obtain PK^* and/or make partial-signing-queries of PK^* before generating APK.

Definition 3 (Security against the Arbitrator). *An optimistic fair exchange protocol is $(t, q_{CU}, q_{PS}, \epsilon)$-secure against the arbitrator in the multi-user setting if no adversary $(t, q_{CU}, q_{PS}, \epsilon)$-breaks it.*

We can obtain the definition of the security against the arbitrator in the single-user setting, namely an optimistic fair exchange protocol is (t, q_{PS}, ϵ)-secure against the arbitrator in the single-user setting if no adversary $(t, 0, q_{PS}, \epsilon)$-breaks it. The following theorem shows that *the security against the arbitrator in the single-user setting is preserved in the multi-user setting.*

Theorem 1. *An optimistic fair exchange protocol is $(t, q_{CU}, q_{PS}, \epsilon)$-secure against the arbitrator in the multi-user setting if it is $(t + t_1 q_{CU}, q_{PS}, \epsilon)$-secure against the arbitrator in the single-user setting. Here, t_1 denotes the time unit to respond to one creating-user query.*

Proof. We denote by $\mathcal{C}_S$ the adversary in the single-user setting and $\mathcal{C}_M$ in the multi-user setting. We will show how to convert a successful $\mathcal{C}_M$ to a successful $\mathcal{C}_S$. At the beginning, $\mathcal{C}_S$ obtains Param from its challenger in the single-user setting.

- **Setup**. Param is given to $\mathcal{C}_M$.
- **Output-I**. Let APK be the arbitrator's public key created by $\mathcal{C}_M$ in the multi-user setting. APK will be sent to $\mathcal{C}_S$'s challenger in the single-user setting. $\mathcal{C}_S$ will make use of $\mathcal{C}_M$ to generate a proof of knowledge, namely $\mathcal{C}_S$ will act as a relay in the proof by forwarding all messages from its challenger to $\mathcal{C}_M$ (or, from $\mathcal{C}_M$ to its challenger). At the end of this phase, $\mathcal{C}_S$ will be given a public key PK^*, which will be forwarded to $\mathcal{C}_M$ as its challenging public key in the multi-user setting.
- **Queries**. We show how $\mathcal{C}_S$ can correctly answer $\mathcal{C}_M$'s queries.
  Creating-User-Queries. For a creating-user query (U_i, PK_{U_i}), $\mathcal{C}_S$ will add PK_{U_i} to *PK-List* if $\mathcal{C}_M$ can generate a proof of knowledge of the legitimate private key.
  Partial-Signing-Queries. For a partial-signing query (m, PK^*), $\mathcal{C}_S$ forwards it to its own challenger and sends the response to $\mathcal{C}_M$.

– **Output-II**. Eventually, $\mathcal{C}_M$ will output a pair (m_f, σ_f). $\mathcal{C}_S$ will set (m_f, σ_f) as its own output in the single-user setting.

$\mathcal{C}_S$ will win the game in the single-user setting if $\mathcal{C}_M$ wins the game in the multi-user setting. It follows that the success probability of $\mathcal{C}_S$ will be ϵ if $\mathcal{C}_M$ can $(t, q_{CU}, q_{PS}, \epsilon)$-break the security against the arbitrator in the multi-user setting.

It remains to show the time consumption in the proof. $\mathcal{C}_S$'s running time is the same as $\mathcal{C}_M$'s running time plus the time it takes to answer creating-user-queries, which we assume each query takes time at most t_1. Therefore, the total time consumption is $t + t_1 q_{CU}$.

We have shown that for an optimistic fair exchange protocol, if there is an adversary $(t, q_{CU}, q_{PS}, \epsilon)$-breaks the security against the arbitrator in the multi-user setting, then there is an adversary $(t + t_1 q_{CU}, q_{PS}, \epsilon)$-breaks the security against the arbitrator in the single-user setting. This completes the proof of Theorem 1. □

Section 3 will investigate the conditions under which the security against the signer and the security against the verifier in the single-user setting will remain in the multi-user setting.

3 Strong Resolution-Ambiguity

This section investigates a new property of optimistic fair exchange, which we call "Strong Resolution-Ambiguity". We will give the definition of strong resolution-ambiguity and prove that for optimistic fair exchange protocols with that property, the security against the signer and the security against the verifier in the single-user setting are preserved in the multi-user setting. Before giving the formal definition, we first review a generic construction of optimistic fair exchange [11].

Optimistic Fair Exchange from Sequential Two-Party Multisignature A multisignature scheme allows any subgroup of users to jointly sign a document such that a verifier is convinced that each user of the subgroup participated in the signing. To construct an optimistic fair exchange protocol, one can use a simple type of multisignature, which is called sequential two-party multisignature. In this construction, the signer first generates two key pairs (pk, sk) and $(\mathsf{APK}, \mathsf{ASK})$, where $(pk, \mathsf{APK}, \mathsf{ASK})$ are sent to the arbitrator through a secured channel. The signer's private key SK is the pair (sk, ASK) and the arbitrator's private key is ASK. The partial signature σ' of a message m is an ordinary signature generated using sk, and the full signature σ is the multisignature generated using σ' and ASK. Given a valid partial signature, both the arbitrator and the signer can convert it to a full signature using ASK. (Recall that ASK is the arbitrator's private key and part of the signer's private key.) It is thus virtually infeasible to tell who (the signer or the arbitrator) converted the partial signature to the full signature. This is the essential requirement of optimistic fair exchange with strong resolution-ambiguity, which is formally defined as follows.

3.1 Definition of Strong Resolution-Ambiguity

We first introduce a probabilistic polynomial-time algorithm $\mathsf{Convert}$ which allows the signer to convert a partial signature to a full one. The definition of $\mathsf{Convert}$ is given as below.

– $\mathsf{Convert}$. This algorithm takes as input the signer's private key SK_{U_i}, (optionally) arbitrator's public key APK, a message m and its valid partial signature σ'. The output is the signer's full signature σ on m.

In a trivial case, each optimistic fair exchange protocol has an algorithm $\mathsf{Convert} = \mathsf{Sig}$. (In this case the full signature generated by $\mathsf{Convert}$ could be totally independent of the partial signature.) Our interest here is to investigate non-trivial $\mathsf{Convert}$ and compare it with the resolution algorithm Res. Recall that, with the knowledge of ASK, one can also convert a partial signature to a full one using Res. This makes the following question interesting: *Given a valid partial signature σ', what are the differences between full signatures produced by* $\mathsf{Convert}$ *and those produced by* Res? The answer to this question inspires the definition of strong resolution-ambiguity.

To formally define the strong resolution-ambiguity, we assume the arbitrator's key pair satisfies an NP-relation R_{TTP}, and users' key pairs satisfy another NP-relation R_{U}. An NP-relation R is a subset of $\{0,1\}^* \times \{0,1\}^*$ for which there exists a polynomial f such that $|y| \leq f(|x|)$ for all $(x, y) \in R$, and there exists a polynomial-time algorithm for deciding membership in R.

In an optimistic fair exchange protocol defined in Section 2, let $(\mathsf{APK}, \mathsf{ASK})$ be any pair in R_{TTP}, and let $(\mathsf{PK}_{U_i}, \mathsf{SK}_{U_i})$ be any pair in R_{U}. For any pair (m, σ') satisfying $\mathsf{PVer}(m, \sigma', \mathsf{PK}_{U_i}, \mathsf{APK}) = \texttt{valid}$, we define

$\mathbb{D}^{(m,\sigma')}_{\mathsf{Convert}}$: probability distribution of full signatures produced by $\mathsf{Convert}(m, \sigma', \mathsf{SK}_{U_i}, \mathsf{APK})$.

$\mathbb{D}^{(m,\sigma')}_{\mathsf{Res}}$: probability distribution of full signatures produced by $\mathsf{Res}(m, \sigma', \mathsf{PK}_{U_i}, \mathsf{ASK})$.

Definition 4 (Strong Resolution-Ambiguity). *An optimistic fair exchange protocol is said to satisfy strong resolution-ambiguity if there exists an algorithm* $\mathsf{Convert}$ *as defined above such that $\mathbb{D}^{(m,\sigma')}_{\mathsf{Convert}}$ is identical to $\mathbb{D}^{(m,\sigma')}_{\mathsf{Res}}$.*

Strong Resolution-Ambiguity and Resolution-Ambiguity: A Brief Comparison

An optimistic fair exchange protocol with strong resolution-ambiguity will satisfy resolution-ambiguity if Sig is defined as $(\mathsf{PSig} + \mathsf{Convert})$, namely the signer first generates a partial signature and then converts it to a full one using $\mathsf{Convert}$. In this case, actual signatures (generated by Sig) are indistinguishable from resolved signatures (generated by Res). However, resolution-ambiguity cannot ensure strong resolution-ambiguity which requires that one can use the signer's private key to convert a partial signature to a full one and the conversion is indistinguishable from that using the arbitrator's private key.

3.2 Optimistic Fair Exchange Protocols with/without Strong Resolution-Ambiguity

It is evident that the generic construction of optimistic fair exchange from sequential two-party multisignature [11] (reviewed at the beginning of Section 3) has the strong resolution-ambiguity property by defining $\mathsf{Convert} = \mathsf{Res}$. Below are some other concrete examples of optimistic fair exchange with/without strong resolution-ambiguity.

Optimistic Fair Exchange from Verifiably Encrypted Signatures
Let OFE-VES be optimistic fair exchange protocols constructed from verifiably encrypted signatures. If the algorithm Sig is deterministic (e.g., the verifiably encrypted signature scheme in [8]), then OFE-VES will have the strong resolution-ambiguity property. For any valid partial signature of m, there is only one output of the algorithm Res, namely the unique full signature of m. By defining $\mathsf{Convert} = \mathsf{Sig}$, $\mathbb{D}^{(m,\sigma')}_{\mathsf{Convert}}$ and $\mathbb{D}^{(m,\sigma')}_{\mathsf{Res}}$ will be identical and the protocols satisfy strong resolution-ambiguity. OFE-VES with probabilistic Sig algorithms could also have the strong resolution-ambiguity property. One example is the optimistic fair exchange protocol from the verifiably encrypted signature scheme proposed in [15]. In [15], the Sig algorithm is the signing algorithm in Waters signature [19], and the partial signature σ' is the encryption of the full signature σ using APK. After extracting σ from σ', the arbitrator will randomize σ such that the output of Res is a full signature uniformly distributed in the full signature space. This makes the distribution of full signatures produced by Res the same as that of full signatures generated by $\mathsf{Convert} = \mathsf{Sig}$.

A Concrete Instance of the Generic Construction in [14]
The generic construction of optimistic fair exchange in [14] is based on a conventional signature scheme and a ring signature scheme, both of which can be constructed efficiently without random oracles. In the protocol, the signer and the arbitrator first generate their own key pairs. The full signature of a message m is a pair (s_1, s_2), where s_1 is the signer's conventional signature on the message m, and s_2 is a ring-signature on m and s_1. Either the signer or the arbitrator is able to generate s_2. This construction will satisfy strong resolution-ambiguity if the distribution of ring signatures generated by the signer is the same as that of ring signatures generated by the arbitrator (e.g., 2-User ring signature scheme without random oracles [5]).

A Concrete Protocol without Strong Resolution-Ambiguity
One example of optimistic fair exchange protocols *without* strong resolution-ambiguity is the single-user secure but multi-user *insecure* optimistic fair exchange protocol proposed in [10]. In this protocol, the full signature of a message m is $\sigma = (r, \delta)$, where δ is the signer's conventional signature on "$m \| y$", $y = f(r)$, and f is a trapdoor one-way permutation. The partial signature is defined as $\sigma' = (y, \delta)$. To convert (y, δ) to a full signature, the arbitrator uses his/her private key f^{-1} to compute $r = f^{-1}(y)$ and obtain the full signature (r, δ). Given a message m and its full signature (r, δ), it is hard to tell if (r, δ) is produced by Sig directly, or first generated by PSig and then by Res. Thus,

as shown in [10], the property "resolution-ambiguity" is satisfied. On the other hand, this protocol does not have strong resolution-ambiguity as f is a trapdoor one-way permutation. Suppose, otherwise, there is an algorithm $\mathsf{Convert}$ such that for a partial signature σ', the outputs of $\mathsf{Convert}(m, \sigma', \mathsf{SK}_{U_i}, f)$ have the same probability distribution as those of $\mathsf{Res}(m, \sigma', \mathsf{PK}_{U_i}, f^{-1})$. Note that for $\sigma' = (y, \delta)$, Res will output a pair (r, δ) such that $y = f(r)$. It follows that $\mathsf{Convert}(m, \sigma', \mathsf{SK}_{U_i}, f)$ must also output (r, δ) satisfying $y = f(r)$ if the protocol has strong resolution-ambiguity. This breaks the one-wayness of f, namely given y, there is an efficient algorithm $\mathsf{Convert}$ which can find r such that $f(r) = y$ without the trapdoor f^{-1}.

Notice that given a partial signature σ', the signer can generate a full signature σ such that σ is indistinguishable from the one converted by the arbitrator. To do that, the signer needs to maintain a list $\{(r, y) : y = f(r)\}$ when he/she produces the partial signature $\sigma' = (y, \delta)$. Later on, for a partial signature (y, δ), the signer can search the list and find the matching pair (r, y). In this case, the signer can generate a full signature (r, δ) which is indistinguishable from the one converted by the resolution algorithm Res. However, this approach does not satisfy the definition of $\mathsf{Convert}$ since it requires an additional input r. (Recall that the inputs of $\mathsf{Convert}$ are only SK_{U_i}, (m, σ') and APK.)

3.3 Security of Optimistic Fair Exchange Protocols with Strong Resolution-Ambiguity

Theorem 1 has shown that the security against the arbitrator in the single-user setting is preserved in the multi-user setting. This section considers the other two security notions, and we will prove that:

1. For optimistic fair exchange protocols with strong resolution-ambiguity, the security against the signer in the single-user setting remains in the multi-user setting (Theorem 2).
2. For optimistic fair exchange protocols with strong resolution-ambiguity, the security against the verifier in the single-user setting remains in the multi-user setting (Theorem 3).

Theorem 2. *An optimistic fair exchange protocol with strong resolution ambiguity is $(t, q_{CU}, q_R, \epsilon)$-secure against signers in the multi-user setting, if it is $(t + t_1 q_{CU} + t_2 q_R, q_R, \epsilon/q_{CU})$-secure against the singer in the single-user setting. Here, t_1 is the time unit depends on the validity of the proof of knowledge and t_2 is the time unit depends on the algorithm* $\mathsf{Convert}$ *in the protocol.*

Proof. We denote by $\mathcal{A}_S$ the adversary in the single-user setting and $\mathcal{A}_M$ in the multi-user setting. In the proof, we use the standard method by showing that for an optimistic fair exchange protocol with strong resolution-ambiguity, a successful $\mathcal{A}_M$ can be converted into a successful $\mathcal{A}_S$. We first give a high-level description of the proof.

$\mathcal{A}_S$ will act as the challenger of $\mathcal{A}_M$ in the proof and answer all queries from the latter. $\mathcal{A}_S$ will set the challenging public key PK^* of $\mathcal{A}_M$ as its own challenging public key, and set $\mathcal{A}_M$'s output as its own output. The most difficult part in the proof is how $\mathcal{A}_S$ can correctly answer resolution queries from $\mathcal{A}_M$. For resolution queries related to PK^*, $\mathcal{A}_S$ can use its own challenger to generate correct responses. However, this is not feasible for resolution queries about other public keys (since $\mathcal{A}_S$'s challenger only responds to queries about PK^*). Fortunately, such queries can be correctly answered by $\mathcal{A}_S$ if the optimistic fair exchange protocol has strong resolution-ambiguity. For a resolution query $(m, \sigma', \mathsf{PK}_{U_i})$, $\mathcal{A}_S$ can convert σ' to a full signature σ using the algorithm Convert and the private key SK_{U_i}. Due to Def. 4, this perfectly simulates the real game between $\mathcal{A}_M$ and the challenger in the multi-user setting. The private key SK_{U_i} can be extracted by $\mathcal{A}_S$ due to the validity of the proof of knowledge required in the creating-user phase.

The details of the proof appear in the full version of this paper. □

Theorem 3. *An optimistic fair exchange protocol with strong resolution ambiguity is $(t, q_{CU}, q_{PS}, q_R, \epsilon)$-secure against verifiers in the multi-user setting, if it is $(t + t_1 q_{CU} + t_2 q_R, q_{PS}, q_R, \epsilon)$-secure against the verifier in the single-user setting. Here, t_1 is the time unit depends on the validity of the proof of knowledge and t_2 is the time unit depends on the algorithm* Convert *in the protocol.*

Proof. The details of the proof appear in the full version of this paper.

Remark 2. Our analysis only shows that strong resolution-ambiguity is a sufficient condition for single-user secure optimistic fair exchange protocols remaining secure in the multi-user setting. It is not a necessary property for (multi-user secure) optimistic fair exchange protocols.

4 A New Optimistic Fair Exchange Protocol with Strong Resolution-Ambiguity

A new optimistic fair exchange protocol with strong resolution-ambiguity is proposed in this section. The protocol is based on Waters signature [19] from bilinear mappings. Definitions of bilinear mappings and computational Diffie-Hellman assumption can be found in [19].

4.1 The Proposed Protocol

Let $(\mathbb{G}, \mathbb{G}_T)$ be bilinear groups of prime order p and let g be a generator of $\mathbb{G}$. e denotes the bilinear mapping $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Let n be the bit-string length of the message to be signed. For an element m in $\{0,1\}^n$, let $\mathcal{M} \subseteq \{1, 2, \cdots, n\}$ be the set of all i for which the i^{th} bit m_i is 1. The parameter Param is $(\mathbb{G}, \mathbb{G}_T, p, g, e, n)$.

- $\mathsf{Setup}^{\mathsf{TTP}}$. Given Param, the arbitrator chooses a random number $w \in \mathbb{Z}_p$ and calculates $W = g^w$. The arbitrator's public key APK is W, and the private key ASK is w.
- $\mathsf{Setup}^{\mathsf{User}}$. Given Param, this algorithm outputs a private signing key $\mathsf{SK}_{U_i} = (x_{U_i}, y_{U_i})$ and a public verification key $\mathsf{PK}_{U_i} = (X_{U_i}, Y_{U_i}, \boldsymbol{v}_{U_i})$, where

1. x_{U_i} and y_{U_i} are randomly chosen in $\mathbb{Z}_p$;
2. $X_{U_i} = e(g,g)^{x_{U_i}}$ and $Y_{U_i} = g^{y_{U_i}}$; and
3. $\boldsymbol{v}_{U_i}$ is a vector consisting of $n+1$ elements $V_0, V_1, V_2, \cdots, V_n$. All these elements are randomly selected in $\mathbb{G}$.

– Sig. Given a message m, the signer U_i uses the private key x_{U_i} to generate a Waters signature $\sigma = (\sigma_1, \sigma_2)$, where $\sigma_1 = g^{x_{U_i}} \cdot (V_0 \prod_{i \in \mathcal{M}} V_i)^r$, $\sigma_2 = g^r$ and r is a random number in $\mathbb{Z}_p$.
– Ver. Given a message-signature pair (m, σ) and U_i's public key $\mathsf{PK}_{U_i} = (X_{U_i}, Y_{U_i}, \boldsymbol{v}_{U_i})$, this algorithm outputs `valid` if $e(\sigma_1, g) = X_{U_i} \cdot e(V_0 \prod_{i \in \mathcal{M}} V_i,\ \sigma_2)$. Otherwise, this algorithm outputs `invalid`.
– PSig. Given a message m and the arbitrator's public key W, the signer U_i first runs Sig to obtain a full signature (σ_1, σ_2). After that, U_i calculates $\sigma_1' = \sigma_1 \cdot W^{y_{U_i}}$ and $\sigma_2' = \sigma_2$. The partial signature σ' is (σ_1', σ_2').
– PVer. Given a pair (m, σ'), U_i's public key PK_{U_i} and arbitrator's public key APK (which is W), one parses σ' as (σ_1', σ_2'). This algorithm outputs `valid` if $e(\sigma_1', g) = X_{U_i} \cdot e(Y_{U_i}, W) \cdot e(V_0 \prod_{i \in \mathcal{M}} V_i, \sigma_2')$. Otherwise, it outputs `invalid`.
– Res. Given a valid partial signature σ' of the message m under a public key $\mathsf{PK}_{U_i} = (X_{U_i}, Y_{U_i}, \boldsymbol{v}_{U_i})$, the arbitrator first parses σ' as (σ_1', σ_2'). After that, the arbitrator uses the private key w to calculate $\sigma_1 = \sigma_1' \cdot (Y_{U_i})^{-w}$ and $\sigma_2 = \sigma_2'$. The arbitrator then chooses a random number $r' \in \mathbb{Z}_p$ and calculates $\sigma_1^R = \sigma_1 \cdot (V_0 \prod_{i \in \mathcal{M}} V_i)^{r'}$ and $\sigma_2^R = \sigma_2 \cdot g^{r'}$. The output of the algorithm Res is (σ_1^R, σ_2^R).

Analysis of Our Protocol. It is evident that our protocol is setup-free and stand-alone. We show that it also satisfies strong resolution-ambiguity.

One can find an algorithm $\mathsf{Convert}$, which is the same as Sig, such that given any partial signature σ', the outputs of $\mathsf{Convert}$ are indistinguishable from those produced by Res, both of which are uniformly distributed in the valid signature space of Waters signature. Thus, the proposed protocol also satisfies strong resolution-ambiguity.

The following theorem shows that the protocol is secure in the multi-user setting.

Theorem 4. *The proposed protocol is multi-user secure under computational Diffie-Hellman assumption.*

Proof. The details of the proof appear in the full version of this paper. □

4.2 Comparison to Previous Protocols

Table. 1 compares the known optimistic fair exchange protocols which have the same properties as the newly proposed one (namely, non-interactive, setup-free, stand-alone and multi-user secure without random oracles). The comparison is made from the following aspects: (1) underlying complexity assumption, (2) partial signature size and full signature size, and (3) the computational cost of signing and verifying partial signatures and full signatures. We consider the cost of signing and verifying partial signatures since the signer must generate

Table 1. Multi-user Secure Stand-Alone and Setup-Free Optimistic Fair Exchange Protocols without Random Oracles

	Our Protocol	[15]	[20]	[18]
Complexity Assumption	CDH	CDH	CT-CDH	SDH
Full Signature	Waters [19]	Waters [19]	Waters [19]	BB [7]
Signature Size$^{\mathsf{PSig}}$	$2\|\mathbb{G}\|$	$3\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$2\|\mathbb{G}\| + \|\mathbb{Z}_p\|$
Signing Cost$^{\mathsf{PSig}}$	$\mathsf{C_W}$+ $1\mathsf{Exp}_{\mathbb{G}}$	$\mathsf{C_W}$+ $2Exp_{\mathbb{G}}$	$\mathsf{C_W}$	$\mathsf{C_{BB}}$+$2Exp_{\mathbb{G}}$
Verification Cost$^{\mathsf{PVer}}$	$2BM$+$1\mathsf{BM}$	$3BM$	$2BM$+$1\mathsf{BM}$	$2BM$+$4\mathsf{BM}$

Notations.
CDH: Computational Diffie-Hellman assumption.
CT-CDH: Chosen-target computational Diffie-Hellman assumption [6].
SDH: Strong Diffie-Hellman assumption.
$|\mathbb{G}|$: bit length of an element in $\mathbb{G}$, $|\mathbb{Z}_p|$: bit length of an element in $\mathbb{Z}_p$.
$\mathsf{C_W}$: Computational cost of generating one Waters signature [19].
$\mathsf{C_{BB}}$: Computational cost of generating one BB signature [7].
$Exp_{\mathbb{G}}$: Exponentiation in $\mathbb{G}$, $\mathsf{Exp}_{\mathbb{G}}$: Pre-computable exponentiation in $\mathbb{G}$.
BM: Bilinear mapping, BM: Pre-computable bilinear mapping.

a partial signature in each exchange, which will be verified by the verifier and could also be checked again by the arbitrator. Therefore, the efficiency of signing and verifying partial signatures is at least as important as that of full signatures. In Table. 1, the most efficient one is the protocol constructed from the verifiably encrypted signature scheme in [18], whose security assumption is strong Diffie-Hellman assumption (SDH). The other three protocols are all based on Waters signature, but the security of the protocol in [20] can only be reduced to a stronger assumption: chosen-target computational Diffie-Hellman assumption (CT-CDH). Our protocol and the one proposed in [15] are designed in a similar manner. When compared with [15], our protocol has a shorter partial signature size and is more efficient in signing and verifying partial signatures. This is achieved at the cost of larger key size (one more pair (y_{U_i}, Y_{U_i}) in $\mathbb{Z}_p \times \mathbb{G}$).

5 Conclusion

This paper shows several new results about optimistic fair exchange in the multi-user setting. We formally defined the *Strong Resolution-Ambiguity* in optimistic fair exchange and demonstrated several concrete optimistic fair exchange protocols with that property. In the certified-key model, we prove that for optimistic fair exchange protocols with strong resolution-ambiguity, the security in the single-user setting can guarantee the security in the multi-user setting. In addition to theoretical investigations, a new construction of optimistic fair exchange with strong resolution-ambiguity was proposed. The new protocol is setup-free, stand-alone, and provably secure in the multi-user setting without random oracles.

References

1. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for fair exchange. In: Proceedings of the 4th ACM conference on Computer and Communications Security, pp. 7–17. ACM Press, New York (1997)
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. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. IEEE Journal on Selected Areas in Communication 18(4), 593–610 (2000)
4. 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)
5. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 60–79. Springer, Heidelberg (2006)
6. 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)
7. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
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. Camenisch, J., Damgård, I.: Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 331–345. Springer, Heidelberg (2000)
10. Dodis, Y., Lee, P.J., Yum, D.H.: Optimistic fair exchange in a multi-user setting. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 118–133. Springer, Heidelberg (2007)
11. Dodis, Y., Reyzin, L.: Breaking and repairing optimistic fair exchange from PODC 2003. In: Proceedings of the 3rd ACM Workshop on Digital Rights Management, pp. 47–54. ACM, New York (2003)
12. Garay, J.A., Jakobsson, M., MacKenzie, P.: Abuse-free optimistic contract signing. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 449–466. Springer, Heidelberg (1999)
13. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Ambiguous optimistic fair exchange. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 74–89. Springer, Heidelberg (2008)
14. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles. In: Malkin, T.G. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 106–120. Springer, Heidelberg (2008)
15. 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)
16. Markowitch, O., Kremer, S.: An optimistic non-repudiation protocol with transparent trusted third party. In: Davida, G.I., Frankel, Y. (eds.) ISC 2001. LNCS, vol. 2200, pp. 363–378. Springer, Heidelberg (2001)

17. Park, J.M., Chong, E.K.P., Siegel, H.J.: Constructing fair-exchange protocols for E-commerce via distributed computation of RSA signatures. In: Proceedings of the twenty-second annual symposium on Principles of distributed computing, pp. 172–181. ACM, New York (2003)
18. Rückert, M., Schröder, D.: Security of verifiably encrypted signatures and a construction without random oracles. In: Shacham, H. (ed.) Pairing 2009. LNCS, vol. 5671, pp. 17–34. Springer, Heidelberg (2009)
19. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
20. Zhang, J., Mao, J.: A novel verifiably encrypted signature scheme without random oracle. In: Dawson, E., Wong, D.S. (eds.) ISPEC 2007. LNCS, vol. 4464, pp. 65–78. Springer, Heidelberg (2007)
21. Zhou, J., Gollmann, D.: A fair non-repudiation protocol. In: Proceedings of the 1996 IEEE Symposium on Security and Privacy, Washington DC, pp. 55–61. IEEE, Los Alamitos (1996)
22. Zhu, H., Bao, F.: More on stand-alone and setup-free verifiably committed signatures. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 148–158. Springer, Heidelberg (2006)
23. Zhu, H., Bao, F.: Stand-alone and setup-free verifiably committed signatures. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 159–173. Springer, Heidelberg (2006)
24. Zhu, H., Susilo, W., Mu, Y.: Multi-party stand-alone and setup-free verifiably committed signatures. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 134–149. Springer, Heidelberg (2007)

Secure Network Coding over the Integers[⋆]

Rosario Gennaro[1], Jonathan Katz[2,⋆⋆], Hugo Krawczyk[1], and Tal Rabin[1]

[1] IBM T.J. Watson Research Center, Hawthorne, NY
{rosario,talr}@us.ibm.com, hugo@ee.technion.ac.il
[2] Department of Computer Science, University of Maryland
jkatz@cs.umd.edu

Abstract. Network coding offers the potential to increase throughput and improve robustness without any centralized control. Unfortunately, network coding is highly susceptible to "pollution attacks" in which malicious nodes modify packets improperly so as to prevent message recovery at the recipient(s); such attacks cannot be prevented using standard end-to-end cryptographic authentication because network coding mandates that intermediate nodes modify data packets in transit.

Specialized "network coding signatures" addressing this problem have been developed in recent years using homomorphic hashing and homomorphic signatures. We contribute to this area in several ways:

– We show the first homomorphic signature scheme based on the RSA assumption (in the random oracle model).
– We give a homomorphic hashing scheme that is more efficient than existing schemes, and which leads to network coding signatures based on the hardness of factoring (in the standard model).
– We describe variants of existing schemes that reduce the communication overhead for moderate-size networks, and improve computational efficiency (in some cases quite dramatically – e.g., we achieve a 20-fold speedup in signature generation at intermediate nodes).

Underlying our techniques is a modified approach to random linear network coding where instead of working in a vector space over a *field*, we work in a module over the *integers* (with small coefficients).

1 Introduction

Network coding [2,18] offers an alternative, decentralized approach to traditional multicast routing. We consider a network setting where a *source node* has a file that it wants to distribute to a set of *target nodes*. The source partitions the file into m packets which it transmits to its neighboring nodes. Further transmission happens through intermediate nodes who receive packets via incoming links and produce modified packets sent over outgoing links. These outgoing packets are

[⋆] This work was supported by the US Army Research Laboratory and the UK Ministry of Defence under agreement number W911NF-06-3-0001.

[⋆⋆] Work done while visiting IBM, and supported also by NSF CAREER award #0447075, NSF grant #0627306, and the US DoD/ARO MURI program.

computed as *linear combinations* of incoming packets, where packets are viewed as vectors in a vector space over some field. (See further discussion in Section 2.1.) We focus on the case of *random linear network coding* [10,13], where scalars are chosen by each intermediate node *at random* from the underlying field. This strategy induces a fully decentralized solution to the routing problem since nodes do not need to coordinate their actions.

Target nodes reconstruct the original file sent by the source using the packets they receive. This can be done if the intermediate nodes *augment* each vector they send with m additional *coding coordinates* that encode the linear combination that resulted in that vector. A target that receives a set of augmented vectors for which the coding coordinates induce a full rank matrix can recover the file sent by the source via simple matrix inversion. (See Section 2.1.) A fundamental question is: what is the *decoding probability* at the targets; i.e., what is the probability with which a target is able to reconstruct the original file? The network coding literature shows that small-size fields (e.g., $\mathbb{F}_{2^8}$) provide good decoding probability for sufficiently connected networks.

Although network coding can increase throughput and reliability relative to alternative techniques, it is susceptible to *pollution attacks* in which malicious nodes inject invalid packets that prevent reconstruction of the file at the targets. (An invalid packet is any packet that is not in the linear span of the original augmented vectors sent by the source.) Due to the way vectors are propagated and combined in the network, a single invalid packet injected by an attacker can invalidate many more packets further downstream. This constitutes a serious denial of service attack which can be mounted effortlessly.

Two naive solutions to this problem are easily seen to be inapplicable. Having the source sign the file prevents a target node from reconstructing an incorrect file, but does not enable the target to efficiently reconstruct the correct file in the first place. (Moreover, it does not provide any way for intermediate nodes to drop invalid packets they receive.) Having the source sign each augmented vector it sends (using a standard signature scheme) is also of no help, since intermediate nodes are *supposed* to modify vectors in transit. Prior work has shown, however, that dedicated *network coding signatures* can be used to address pollution attacks. Such signatures have been based on two primitives: *homomorphic hash functions* [17,21] or *homomorphic signatures* [16,7,6]. In both cases, homomorphic properties ensure that the signature (or hashing) operation on a linear combination of vectors results in a corresponding homomorphic combination of signatures (or hash values). See Section 2.2 for further details.

Constructions of homomorphic hash functions are well known, and can be implemented over any prime-order group where the discrete logarithm problem is hard. Building homomorphic signatures is more challenging. So far the only known construction is based on bilinear groups [6] and involves costly pairing operations. In particular, network coding signatures based on homomorphic signatures are computationally more expensive than those built from homomorphic hashing. However, the latter are less communication-efficient since they require each packet transmitted to be sent along with some "authentication data" whose length is proportional to m (the number of file vectors). One drawback of both approaches is that they replace the small fields used in "standard" network coding with very large fields appropriate for cryptography. For example, instead of using vectors over an 8-bit field as in traditional network coding, the cryptographic approaches use vectors over a 160-bit field instead. This increases both the communication and computational overhead.

Our Contributions. We present new and improved network coding signatures. First, we show the first homomorphic signature scheme based on the RSA assumption in the random oracle model.[1] In particular, it offers more efficient processing at the intermediate nodes as compared to the scheme of [6] that is based on bilinear groups and pairings. The bandwidth overhead is also lower for networks of moderate size (e.g., where the maximum path length between source and target nodes is 20–30 hops).

We also present a new homomorphic hashing scheme which is quite efficient. Treating each information vector v as a single (large) integer, we define our hash function simply as $H_N(v) = 2^v \bmod N$ for a composite N. This hash function is homomorphic over the integers and can be proven collision resistant based on the hardness of factoring. This constructions leads to a network coding signature scheme based on the factoring assumption and without random oracles.

A core technique we use for both the above constructions is to apply network coding in a module over the *integers* rather than in a vector space over a *field* as is traditionally done. By working over the integers we enable the homomorphic properties of the above two schemes (where the group order is unknown), and furthermore can work with *small coefficients* (that need not be cryptographically large). This has the immediate effect of improving the computation at intermediate nodes, and it also reduces the total bandwidth overhead for networks with moderate-length paths between source and targets.

We must analyze how this change from working over a field to working over the integers affects the decoding probability. We show that if the integer coefficients are taken from a set $Q = \{0, \ldots, q-1\}$ for prime q, then the decoding probability is at least as good as working over the field $\mathbb{F}_q$; thus we conclude that using 8-bit coefficients is good enough for most applications.

The ability to perform with network coding with small integer coefficients allows us also to improve the performance of existing schemes. We show that by choosing coefficients from a small set Q as above (but still performing computations modulo the large prime p as required by prior schemes) we can significantly improve performance: e.g., we obtain roughly a 20-fold improvement in signature generation time at intermediate nodes and a reduction in the communication overhead as well.

[1] Yu et al. [20] recently proposed an RSA-based homomorphic signature scheme, but their scheme is essentially flawed (e.g., no signature, even one produced by an honest source, ever passes verification). The problem is that Yu et al. *incorrectly* assume (cf. equations (11) and (12) in Section III-B of [20]) that for integers A, b, d, a prime e, and RSA composite N, it holds that $(A^b \bmod e)^d \bmod N = (A \bmod e)^{bd} \bmod N$.

Organization. Section 2 reviews network coding and existing network coding signature schemes. In Section 3, we discuss network coding over the integers and show how this translates into performance improvements for existing network coding signature schemes. We present our RSA-based homomorphic signature scheme in Section 4, and our factoring-based homomorphic hashing scheme in Section 5.

2 Background

2.1 Network Coding

We present a high-level description of *linear* network coding (the only type with which we are concerned in this work); for further details see [12]. In this setting, we have a network with a distinguished node S, called the *source*, and a subset of nodes known as *targets*. The objective is for S to transmit a *file* $\bar{F}$ to all the target nodes, where $\bar{F}$ is represented as a matrix containing the m (row) vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)} \in \mathbb{F}^n$ over some finite field $\mathbb{F}$.

The source first creates m *augmented vectors* $\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}$ defined as

$$\bar{w}^{(i)} = (\overbrace{\underbrace{0, \ldots, 0, 1}_{i}, 0, \ldots, 0}^{m} \parallel \bar{v}^{(i)}) \in \mathbb{F}^{m+n};$$

i.e., each original vector $\bar{v}^{(i)}$ of the file is pre-pended with the vector of length m containing a single '1' in the ith position. These augmented vectors are sent by the source to its neighboring nodes.

Each (well-behaved) intermediate node I in the network processes packets (i.e., incoming vectors) as follows. Upon receiving packets $w^{(1)}, \ldots, w^{(\ell)} \in \mathbb{F}^{m+n}$ on its ℓ incoming communication edges, I computes a packet w for each of its outgoing links as a linear combination of the packets that it received. That is, each outgoing packet w transmitted by I takes the form $w = \sum_{i=1}^{\ell} \alpha_i w^{(i)}$, where $\alpha_i \in \mathbb{F}$. We say a vector w transmitted in the network (in the scenario above) is *valid* if it lies in the linear span of the original augmented vectors $\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}$. It is easy to see that if all nodes follow the protocol honestly, then every packet transmitted in the network is valid.

Different strategies for choosing the coefficients α_i yield different variants of network coding. When the $\{\alpha_i\}$ are chosen randomly and independently by each intermediate node, for each of its outgoing communication links, the resulting scheme is referred to as *random* linear network coding [8,10,13]. When analyzing efficiency, we assume random linear network coding is used; our constructions, however, ensure security regardless of how the coefficients are chosen.

To recover the original file, a target node must receive m (valid) vectors $\{w^{(i)} = (u^{(i)} \| v^{(i)})\}_{i=1}^{m}$ for which $u^{(1)}, \ldots, u^{(m)}$ are linearly independent. If we define a matrix U whose rows are the vectors $u^{(1)}, \ldots, u^{(m)}$ and a matrix V whose rows are the vectors $v^{(1)}, \ldots, v^{(m)}$, the original file can be recovered as

$$\bar{F} = U^{-1} V. \tag{1}$$

Assuming the coefficients are chosen randomly and independently by the intermediate nodes, the *decoding probability* — i.e., the probability with which a given target node will be able to recover the file (or, equivalently, the probability with which a given target node will receive m linearly independent vectors, in the sense required above) — is determined by the network topology and the size of the field $\mathbb{F}$. To minimize the communication overhead (due to the first m coordinates of every transmitted vector), it is desirable to keep $|\mathbb{F}|$ as small as possible; on the other hand, choosing $|\mathbb{F}|$ too small would reduce the decoding probability too much. For typical networks encountered in practice, taking $|\mathbb{F}| \approx 256$ has been shown to give a decoding probability of better than 99%.

2.2 Network Coding Signatures

We have already discussed the problem of pollution attacks, and why standard cryptographic mechanisms are incapable of preventing them. Early efforts to deal with pollution attacks focused on *information-theoretic* solutions [11,14,15] that use error-correction techniques to ensure that targets can reconstruct the file as long as the ratio of valid to invalid vectors they receive is sufficiently high. Unfortunately, these techniques (inherently) impose limitations on the number of nodes the adversary can corrupt, the number of packets that can be modified, and/or the number of links on which the adversary can eavesdrop. Researchers have more recently turned to *cryptographic* approaches that place no restrictions on the adversary (other than assuming that the adversary is computationally bounded) [17,7,21,6]. These approaches give *network coding signature schemes* that allow anyone holding the public key[2] of the source to determine whether a given vector is valid. This allows target nodes to reject invalid vectors before reconstructing the file; it also allows intermediate nodes to filter out invalid vectors when generating their outgoing messages. For formal definitions of network coding signatures and their security requirements, see [6].

Two classes of network coding signature schemes are known: those based on *homomorphic hashing*, and those using *homomorphic signatures*. We describe these now at a high level.

Schemes based on homomorphic hashing [17,21,6]. A homomorphic hash function H is a collision-resistant hash function with the property that for any vectors a, b and scalars α, β it holds that $H(\alpha a + \beta b) = H(a)^{\alpha} H(b)^{\beta}$. Collision resistance implies (via standard arguments) that if one knows vectors a, b, c for which $H(c) = H(a)^{\alpha} H(b)^{\beta}$ then it must be the case that $c = \alpha a + \beta b$.

A concrete example [17] of a homomorphic hash function is given by what we call the *exponential homomorphic hash (EHH)* scheme. Let $\mathbb{G}$ be a cyclic group of order p, and let the public key contain random generators $g_1, \ldots, g_n \in \mathbb{G}$. Define a function $\mathbf{H}$ on vectors $v = (v_1, \ldots, v_n) \in \mathbb{Z}_p^n$ as

$$\mathbf{H}(v) = \prod_{j=1}^{n} g_j^{v_j}. \tag{2}$$

[2] A symmetric-key analogue is also possible [9,1], but this allows only a (single) target to verify validity of vectors.

The homomorphic property is easily verified, and collision resistance is implied by the discrete logarithm assumption in $\mathbb{G}$.

Homomorphic hash functions can be used for network coding as follows. For each original vector $\bar{v}^{(i)}$, the source S computes $h_i = H(\bar{v}^{(i)})$; it then signs $(h_1, \ldots, h_m)$ (together with a unique file identifier fid) using a standard signature scheme. The $\{h_i\}$ and their signature are then appended to every packet sent in the network.[3] A node can determine whether a vector $w = (u \parallel v)$ is valid by checking the signature on the $\{h_i\}$ (and the fid), and then verifying whether $\prod_{i=1}^{m} h_i^{u_i} \stackrel{?}{=} H(v)$. In particular, for the EHH scheme $h_i = \mathbf{H}(\bar{v}^{(i)})$ and verification takes the form:

$$\prod_{i=1}^{m} h_i^{u_i} \stackrel{?}{=} \mathbf{H}(v) \stackrel{\text{def}}{=} \prod_{j=1}^{n} g_j^{v_j}. \tag{3}$$

The resulting network coding signature scheme can be proven secure without random oracles based on the discrete logarithm assumption [17,6].

When using homomorphic hashing, the only change in the processing done by intermediate nodes is to verify the hash and forward the authentication information. However, the linear network coding operations performed by intermediate nodes are now done over the (large) field $\mathbb{F} = \mathbb{Z}_p$.

Homomorphic signature schemes [16,7,6]. Here, the full signature (and not just the hash) is homomorphic. Namely, the signature scheme has the property that for any vectors a, b and scalars α, β, it holds that $\mathsf{Sign}(\alpha a + \beta b) = \mathsf{Sign}(a)^{\alpha}\mathsf{Sign}(b)^{\beta}$. The security property, roughly speaking, is that given the signatures of vectors $\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}$ it is *only* feasible to generate signatures on vectors in the linear span of $\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}$. The application to network coding is immediate: The source S signs each augmented vector $\bar{w}^{(i)}$ and transmits each $\bar{w}^{(i)}$ together with its signature $\mathsf{Sign}(\bar{w}^{(i)})$. An intermediate node I that receives a set of incoming vectors with their corresponding signatures will (i) verify the signatures (discarding any vector whose signature is invalid) and (ii) compute (using the homomorphic property) a valid signature on each outgoing vector that it generates. Thus, in addition to the normal network coding processing, intermediate nodes must now generate a signature on each outgoing packet. On the other hand, the per-packet communication overhead due to the signature is now *constant* rather than linear in m as in the case of homomorphic hashing.

A concrete example of a homomorphic signature scheme (the *BFKW scheme*) was given by Boneh et al. [6]; the scheme can be proven secure based on the CDH assumption in the random oracle model. We provide a description here for future reference. To begin, the source S establishes a public key as follows:

1. Generate $\mathcal{G} = (\mathbb{G}, \mathbb{G}_T, p, e)$ where $\mathbb{G}, \mathbb{G}_T$ are groups of prime order p, and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_t$ is a bilinear map. Choose random $g_1, \ldots, g_n, h \in \mathbb{G}$.
2. Choose $s \leftarrow \mathbb{Z}_p$, and set $f := h^s$.
3. Let $H : \{0,1\}^* \times \mathbb{Z} \to \mathbb{G}$ be a hash function, modeled as a random oracle.
4. Output the public key $PK = (\mathcal{G}, H, g_1, \ldots, g_n, h, f)$ and the private key s.

[3] In some settings, there may be alternate ways to distribute the $\{h_i\}$ authentically.

To sign a vector $w = (u \parallel v) \in \mathbb{Z}_p^{m+n}$ associated with the file identifier fid, the source S computes the signature

$$\sigma := \left(\prod_{i=1}^{m} H(\mathsf{fid}, i)^{u_i} \prod_{j=1}^{n} g_j^{v_j} \right)^{s} .$$

(Note that the above can be viewed as applying a cryptographic operation [that depends on the secret key] to a homomorphic hash of w.) An intermediate node who knows PK can verify validity of a vector $w = (u \parallel v)$ with associated signature σ by checking whether

$$e\left(\sigma, h\right) \overset{?}{=} e\left(\prod_{i=1}^{m} H(\mathsf{fid}, i)^{u_i} \prod_{j=1}^{n} g_j^{v_j},\ f \right) . \qquad (4)$$

Upon receiving vectors $w^{(1)}, \ldots, w^{(\ell)}$ with valid signatures $\sigma_1, \ldots, \sigma_\ell$, an intermediate node can generate a valid signature on any linear combination $w = \sum_i \alpha_i w^{(i)}$ by computing $\sigma := \prod_{i=1}^{\ell} \sigma_i^{\alpha_i}$.

3 Network Coding over the Integers

In this section we describe the idea of implementing network coding *over the integers* rather than over a finite field. This approach is essential for the cryptographic schemes we propose in the following sections, and also results in efficiency improvements for existing schemes as we describe here.

Let us first motivate this departure from traditional network coding. Examining the signature schemes described in the previous section, one can see that they result in significant performance penalties relative to basic (insecure) network coding, in terms of both communication and computation. The increase in communication is due to the fact that instead of working over a small (e.g., 8-bit) field as in basic network coding, the cryptographic schemes work modulo a 160-bit prime p. Each file vector is thus augmented by 160-bit coordinates, rather than 8-bit coordinates as in basic network coding — a 20-fold increase in the communication overhead. This also impacts computation; for example, the time required to verify signatures when using the EHH scheme (cf. Eq. (3)) is proportional to the bit-length of the exponents (i.e., the coefficients $\{u_i\}$). A similar effect can be observed in the time required to compute signatures at intermediate nodes when using the BFKW scheme (cf. Eq. (4)).

To alleviate these performance costs, our approach will be to choose small integer coefficients as opposed to 160-bit scalars as in previous schemes. In more detail: We now view the file $\bar{F}$ transmitted by the source S as a sequence of vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)}$ with *integer* coordinates. (At this point we do not specify the dimension of these vectors or the range of the coordinates – these details will depend on the specific cryptographic scheme used). These vectors are augmented with unit vectors $\bar{u}^{(1)}, \ldots, \bar{u}^{(m)}$ as described in Section 2.1. Intermediate nodes will again compute outgoing packets as random linear combinations of incoming vectors, except that now these combinations are taken over the integers and the coefficients α_i are chosen uniformly from $Q = \{0, \ldots, q-1\}$ for some small prime q (e.g., $q = 257$). (A hybrid approach using small integer coefficients but with linear combinations performed modulo a large prime is studied in Section 3.1. In no case are the computations done modulo q.) We stress that the coordinates of the file vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)}$ need not lie in Q.

Recall from Section 2.1 that the usefulness of random linear network coding depends on the decoding probability, namely, the probability with which a recipient can correctly reconstruct the file transmitted by the source. Technically, this is the probability that the recipient collects m vectors whose u-portions form an invertible matrix U (see Eq. (1)). For the setting described above, where operations are performed over the integers, we must re-analyze the decoding probability since existing bounds hold only when network coding is performed over a finite field. Fortunately, we show that working over the integers can only improve the decoding probability in a sense we make formal now.

Lemma 1. *Fix q prime. For any network, the decoding probability when network coding is performed over the integers with intermediate nodes choosing coefficients uniformly from $Q = \{0, \ldots, q-1\}$, is at least the decoding probability when network coding is performed modulo q (with intermediate nodes choosing coefficients uniformly from Q).*

Proof. Fix a sequence of coefficients $\alpha_i \in Q$ chosen by the intermediate nodes during a run of the network coding protocol when operations are performed modulo q. Assume these coefficients lead to successful recovery of the file in this case. This means that the target receives m vectors such that the u-portions of these vectors give a matrix U with $\det(U) \neq 0$ (computed modulo q). Note that $\det(U) \bmod q$ is unchanged if no modular reductions are performed in the network, but instead all modular reductions are 'delayed' and performed only by the target. But if U is an integer matrix, $\det(U) \neq 0 \bmod q$ implies $\det(U) \neq 0$ over the integers; thus, successful recovery would occur for these same coefficients if all operations were performed over the integers. □

The lemma implies that in order to get a good decoding probability when working over the integers, it suffices to choose q such that the decoding probability when working modulo q is sufficiently good. This puts us back in the setting of standard network coding, where the required size of the underlying field is well-studied. Appropriate choice of q depends on the network topology, required fault tolerance, etc., but in most practical applications an 8-bit q suffices.

In fact, we expect that working over the integers with coefficients chosen from $Q = \{0, \ldots, q-1\}$ will induce a decoding probability that is *noticeably better* than working over a field of size q. If so, one could further save in bandwidth and computation by reducing the size of q. Another variant to investigate is choosing coefficients from the set $\{-q/2, \ldots, q/2\}$.

Coordinate growth. When we work over the integers without any modular reduction, the size of the coordinates of the vectors transmitted in the network increases with each traversed hop. Specifically, each hop through some node increases the maximal coordinate of some vector in the network by a factor of at most $\min\{mq, \ell q\}$, where ℓ is the in-degree of that node. (Note that even if $\ell > m$, the incoming vectors contain a set of at most m linearly independent vectors.) So, after L hops the first m coordinates each have magnitude at most $(mq)^L$ (since the initial m coordinates in the augmented vectors sent by the source are 0/1-valued), while the remaining coordinates have magnitude at most $M(mq)^L$, where M is the maximal size of coordinates in the original file vectors $\bar{v}^{(i)}$. As we will see, by working over the integers we obtain bandwidth improvements (in typical networks) in spite of this coordinate growth.

We remark also that an attacker can generate large *valid* packets by choosing large coefficients, thus countering some of the bandwidth gains achieved by having honest nodes use small coefficients. (Note, however, that nodes may be able to reject suspiciously large packets; e.g., those that deviate significantly from the average packet size received at the node or packets whose coefficients exceed an upper bound derived from the distance between the node and the source.) Network coding signatures cannot and do not prevent all forms of denial of service; their purpose is to prevent pollution attacks that are easy for an attacker to carry out yet have devastating effect.

3.1 Improvements to Existing Schemes

We consider here a "hybrid" variant where intermediate nodes choose small integer coefficients but operations are performed modulo a large prime p. This approach will allow us to significantly improve the performance of the schemes described in Section 2.2, while keeping their security guarantees intact.

In the schemes described in Section 2.2, network coding is done modulo a large prime p. That is, the original vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)}$ transmitted by the source are in $\mathbb{Z}_p^n$; the coefficients for the linear combinations are chosen at random from $\mathbb{Z}_p$; and the linear combinations are performed modulo p. Here we suggest to keep these schemes unchanged except that the random coefficients chosen by each intermediate node will be taken from the set $Q = \{0, ..., q-1\}$ for some small prime q (we stress that linear combinations are still computed modulo p).

We first analyze the effect of this change on the decoding probability, showing that the decoding probability remains high as long as (1) p is a random k-bit prime, and (2) m and the maximal path length L from the source to the target are negligible relative to 2^k (the latter is the case in our applications where k is typically 160 or larger).

Lemma 2. *Fix q prime. For any network, the decoding probability of the "hybrid" scheme described above (where intermediate coefficients are chosen at random from $Q = \{0, \ldots, q-1\}$ and the linear combinations are performed modulo a random k-bit prime p) is at least the decoding probability when network coding is performed modulo q (with intermediate nodes choosing coefficients uniformly from Q), up to an $O((Lm \log mq)/2^k)$ additive term.*

Proof. As in the case of Lemma 1, we may assume that all linear combinations in the network are performed over the integers, and all modular reductions are performed only at the end by the target node. Fix some set of coefficients, chosen by all intermediate nodes, for which reconstruction of the file (when operations are performed mod q) succeeds. Letting U^* denote the integer matrix computed at the target, this means that $\det(U^*) \neq 0 \bmod q$ which, in turn, implies $\det(U^*) \neq 0$ (over the integers). We now show that except with probability $O(Lm \log mq/2^k)$ over choice of p, it also holds that $\det(U^*) \neq 0 \bmod p$.

Let d denote the bit-length of $\det(U^*)$. The number of primes of length k dividing $\det(U^*)$ is at most d/k, and the number of primes of length k is $O(2^k/k)$. Thus the probability that p divides $\det(U^*)$ is at most $O(d/2^k)$. It remains to bound $d = O(\log|\det(U^*)|)$.

The matrix U^* is composed of the u-portion of vectors received by the target. As seen before, the u-coordinates of such vectors have magnitude at most $(mq)^L$, where L is the maximal path length from the source to the target. So U^* is an $m \times m$ matrix with each entry having magnitude at most $(mq)^L$. Thus, $\det(U^*) \leq m!(mq)^{Lm} \leq (mq)^{m(L+1)}$ and $d = O(Lm \log mq)$. $\square$

We proceed to examine how using small integer coefficients can improve the performance of the network coding signature schemes discussed in Section 2.2.

Saving bandwidth. In the two schemes reviewed in Section 2.2, all vectors transmitted in the network are pre-pended with a u-portion consisting of m coordinates each 160 bits in length. (For simplicity, we assume here that p is a 160-bit prime.) Using our approach, all vectors are pre-pended with a u-portion consisting of m integer coordinates each of whose length is at most 160 bits (since we are still performing reduction modulo p). On average, however, the length of these coordinates can be much smaller.[4] For example, assume the maximum path length is 16 hops and u-coordinates increase by at most 10 bits per hop (this is the case, e.g., if $\ell = 4$ and $q = 253$). After the first hop the u-coordinates are at most 10-bits long; after the second hop they are at most 20-bits long, etc. Thus, in the worst-case we use (on average over all hops) 80 bits per coordinate which reduces the bandwidth of the u-components by a factor of two as compared to the case when intermediate nodes choose coefficients from $\mathbb{Z}_p$. Better improvements are obtained when average path lengths are shorter; even when average path lengths are longer, our approach can never perform worse than the basic approach.

Saving computation. Reducing the bit-length of the u-coordinates yields computational savings as well, due to the use of shorter exponents during verification (cf. Eqs. (3) and (4)). A major improvement is also obtained in the computation required by intermediate nodes in generating the signatures of their outgoing vectors when using the BFKW scheme, exactly due to the use of small coefficients. This gives a 20-fold improvement for this operation, regardless of the average path length in the network. See also the following remark.

[4] Note that the coordinates of the v-portion of the vectors are not affected by the use of small coefficients; in both cases these are always 160-bit values.

Remark 1. Signature verification can be done on an opportunistic basis by intermediate nodes (e.g., for a random subset of vectors). In contrast, signature computation must be done by *all* intermediate nodes for each outgoing packet.

4 An RSA-Based Network Coding Signature Scheme

In this section we present an RSA-based network coding signature scheme that enjoys a proof of security in the random oracle model under the RSA assumption, and relies on the ability to perform random linear network coding over the integers as described in Section 3. The scheme is similar to the BFKW scheme and adapts ideas from [3,4] in the same way the BFKW scheme borrows from [19]. We construct a homomorphic signature scheme by applying a multiplicatively homomorphic signature to a homomorphic hash of the vector being signed. The homomorphic hash we use is similar to the EHH scheme except that we work modulo an RSA composite rather than modulo a prime. We take as our multiplicatively homomorphic signature the "textbook RSA" scheme where a signature on x is just $x^d \bmod N$. The resulting scheme is presented in Section 4.1.

In order to use the resulting scheme for network coding, it is essential that the linear operations being performed by the nodes "work" relative to an unknown modulus (that arises in our case because $\phi(N)$ is unknown). To achieve this, we have intermediate nodes perform network coding over the integers. We describe this in detail in Section 4.2.

4.1 An RSA-Based Homomorphic Signature Scheme

We start by defining an RSA-based homomorphic signature scheme denoted Bsig.

• Public and secret keys: Let N be a product of two safe primes; in particular, the subgroup of quadratic residues $\mathcal{QR}_N$ is cyclic and random elements of $\mathcal{QR}_N$ are generators of this subgroup with overwhelming probability. The public key is $(N, e, g_1, \ldots, g_n)$ and the secret key is d, where $ed = 1 \bmod \phi(N)$ and $g_1, \ldots, g_n$ are random generators of $\mathcal{QR}_N$.

• Signature generation: The signature on $v = (v_1, \ldots, v_n) \in \mathbb{Z}^n$ is given by

$$\mathsf{Bsig}(v) = \left(\prod_{i=1}^{n} g_i^{v_i} \right)^d \bmod N. \tag{5}$$

Verification is done in the obvious way. It is easy to see that this scheme is homomorphic: for any $v, v' \in \mathbb{Z}^n$ and $\alpha, \beta \in \mathbb{Z}$, we have $\mathsf{Bsig}(\alpha v + \beta v') = (\mathsf{Bsig}(v))^\alpha \cdot (\mathsf{Bsig}(v'))^\beta$.

4.2 An RSA-Based Network Coding Signature Scheme

Here we describe how the above scheme Bsig can be extended to give a network coding signature scheme Nsig. We first review the underlying network coding being performed, focusing on details not already covered in Section 3. The file held by the source S is a sequence of vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)}$, where each $\bar{v}^{(i)} \in \mathbb{Z}^n$ for some value n. Note that once the size of the file and the number m of vectors is fixed, a lower bound $|v|$ on the bit-length of each of the vectors $\bar{v}^{(i)}$ is determined, and n can take on any value between 1 and $|v|$. As we will see, smaller values of n reduce communication while larger values of n reduce computation (very often $n = 1$ will provide the most practical trade-off).

As usual, before sending the $\bar{v}^{(i)}$ vectors to the network, the source pre-pends them with unit vectors $\bar{u}^{(i)}$ thus producing $\bar{w}^{(1)}, \ldots, \bar{w}^{(m)} \in \mathbb{Z}^{m+n}$. Everything else is carried out as already described in Section 3: in particular, intermediate nodes generate random linear combinations (over the integers) of incoming packets, using coefficients chosen uniformly from $Q = \{0, ..., q-1\}$ for prime q.

Let L be an upper bound on the path length from the source to any target. (Looking ahead, the Nsig scheme defined below may reject packets that traverse more than L hops.) Given L we define a bound $B = (mq)^L$ which represents the largest possible value of a u-coordinate in any (honestly generated) vector; cf. Section 3. If M denotes an upper bound on the magnitude of the coordinates of the initial vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)}$, then the maximal magnitude of any coordinate in an honestly generated vector is $B^* = BM$.

We now introduce our scheme Nsig.

• Parameters: m, n, M, B, and B^*.

• Public and secret keys: The public key $(N, e, g_1, \ldots, g_n)$ and the secret key d are as in Bsig, except that e is chosen to be prime with $e > mB^*$ (for efficiency reasons e can be chosen to have low Hamming weight). In addition, the scheme uses a public hash function $H : \{0,1\}^* \to \mathcal{QR}_N$ that will be modeled as a random oracle.

• Signature generation by source S: On input a file given by m vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)} \in \mathbb{Z}^n$, the source S generates the augmented vectors $\bar{w}^{(i)} = \bar{u}^{(i)} \parallel \bar{v}^{(i)} \in \mathbb{Z}^{m+n}$ in the usual way. S chooses random $\mathsf{fid} \in \{0,1\}^k$, and computes $h_i = H(i, \mathsf{fid})$ for $i = 1, \ldots, m$. The signature on each vector $w = (u_1, \ldots, u_m, v_1, \ldots, v_n)$ is:

$$\mathsf{Nsig}(w) = \left(\prod_{i=1}^{m} h_i^{u_i} \prod_{j=1}^{n} g_j^{v_j} \right)^d \bmod N. \tag{6}$$

S transmits each $\bar{w}^{(i)}$ along with its signature and fid.

• Signature verification: Given $w = u \parallel v = (u_1, \ldots, u_m, v_1, \ldots, v_n) \in \mathbb{Z}^{m+n}$, a file identifier fid, and a signature σ, verification is done as follows. Reject immediately if any of the u-coordinates is negative or larger than B, or any of the v-coordinates is negative or larger than B^*. Otherwise, compute $h_i = H(i, \mathsf{fid})$ for $i = 1, \ldots, m$ and accept the signature if and only if

$$\sigma^e \stackrel{?}{=} \prod_{i=1}^{m} h_i^{u_i} \prod_{j=1}^{n} g_j^{v_j} \bmod N. \tag{7}$$

(An optimized batch verification procedure for testing multiple incoming vectors is presented at the end of this subsection.)

• Signature combination at intermediate nodes. Upon receiving $w^{(1)}, \ldots, w^{(\ell)}$ associated with the same fid and with valid signatures $\sigma_1, \ldots, \sigma_\ell$, an intermediate node proceeds as follows. It first discards any $w^{(i)}$ having a u-coordinate larger than B/mq or a v-coordinates larger than B^*/mq.[5] For simplicity we continue to denote the non-discarded vectors by $w^{(1)}, \ldots, w^{(\ell)}$. The intermediate node then chooses random coefficients $\alpha_1, \ldots, \alpha_\ell \in Q$, sets $w = \sum_{i=1}^{\ell} \alpha_i w^{(i)}$, and computes the signature on w as:

$$\sigma = \prod_{i=1}^{\ell} \sigma_i^{\alpha_i} \bmod N. \tag{8}$$

We prove security of this scheme in the following subsection. First, we compare the performance of our scheme to the original BFKW scheme and the variant BFKW scheme (using small integer coefficients) described in Section 3.1.

Bandwidth. The lengths of the coordinates of the vectors w transmitted in the network increase by at most $s = \log(mq)$ bits for each traversed hop. Thus, after t hops each u-coordinate has bit-length at most ts. If $s = 10$, for example, then it will take 32 hops before the total communication overhead due to the u-coordinates exceeds that of the original BFKW scheme (where u-coordinates are always of size 160 bits). For most networks, where maximum path-lengths are expected to be much less than 32 hops, Nsig therefore incurs lower overhead. Comparing Nsig to the variant BFKW scheme described earlier in this work, we see that the two schemes have the same overhead until coordinates reach 160 bits; after that the variant BFKW scheme performs better (since in Nsig coordinates keep growing while in BFKW they do not). This, however, does not take into account the fact that in Nsig the v-coordinates also increase while in BFKW they do not. Fortunately, we can choose n to be small (e.g., $n = 1$), thus making this overhead insignificant (see more below regarding the choice of n).

Computation. The most critical operation is signature generation at intermediate nodes (see Remark 1 in Section 3.1). In Nsig this operation is extremely efficient since the exponents α_i in Eq. (8) are small (say, 8 bits each). Thus, we can expect this operation to be roughly 20 times faster in Nsig than in the original BFKW scheme. (The variant BFKW scheme is expected to perform about as well as Nsig.) Verification is more expensive. Looking at Eq. (7), we see that verification in Nsig requires an exponentiation using $(|v| + (m + n)\log B)$-bit exponents and a $(\log m + \log B + |v|/n)$-bit exponent (i.e., the bit-length of e). Since the impact of n is more significant with regard to bandwidth than computation, in most cases it makes sense to choose $n = 1$. The resulting cost of verification is still better than that of the original BFKW scheme due to the pairing operation and the cost of hashing onto the bilinear groups in the latter. The cost of a hashing operation in this case is equivalent to a full exponentiation

[5] These bounds are more restrictive than those required by signature verification, and are intended to ensure that signature verification will succeed at the *next* hop.

and it is needed for computing each of the m values $h_i = H(\mathsf{fid}, i)$ (and also to compute the generators $g_1, \ldots, g_n$ in the case that one implements BFKW with fixed-size public key). In contrast, in the case of Nsig the computational cost of the hashing operations is negligible. Moreover, if one uses $n = 1$ the resultant public key has a single generator while in BFKW one needs $|v|/160$ of them (e.g., for a 4 Kbyte $|v|$, BFKW requires 200 generators). The cost of computation can be further improved by resorting to a batch verification of incoming vectors as described next.

Batch verification. The most expensive operation in the Nsig scheme is the verification of incoming signatures. Here we show that instead of verifying each incoming signature it suffices to *verify just one outgoing signature.* The probability that the verification of this outgoing vector succeeds but one of the incoming vectors was invalid is at most $1/q$. This is fine in most cases since even if a node forwards an invalid vector this will be caught with high probability by subsequent nodes (i.e., the probability that t consecutive honest nodes do not discover a forgery is at most $1/q^t$). To achieve this optimization we modify the actions of intermediate nodes as follows.

Upon receiving $w^{(1)}, \ldots, w^{(\ell)}$ associated with the same fid and with alleged signatures $\sigma_1, \ldots, \sigma_\ell$, intermediate node I discards any vector that has too large coordinates as described above. Then, I generates one outgoing vector as usual, i.e., chooses random coefficients $\alpha_1, \ldots, \alpha_\ell \in Q$ and sets $w = \sum_{i=1}^{\ell} \alpha_i w^{(i)}$. It then sets $\sigma = \prod_{i=1}^{\ell} \sigma_i^{\alpha_i} \bmod N$ and verifies (using Eq. (7)) that $\mathsf{Nsig}(w)$ equals σ or $-\sigma$. If this verification succeeds, then no further verifications are needed. That is, I outputs w on one of its outgoing edges and proceeds to compute other outgoing vectors as in the case that all incoming vectors and their signatures were valid (with the usual random linear combinations and using Eq. (8) to generate outgoing signatures but without additional verifications). In this way, the number of signature verifications at any intermediate node is 1 regardless of the number of incoming or outgoing vectors. (Note: If the above verification of outgoing w fails, I may decide to discard its incoming vectors or test each one separately to find the valid ones – the important point is that under normal operation, i.e., without adversarial activity, a single verification suffices).

A proof of correctness of the above batch verification technique follows [5] and is presented in the full version.

4.3 Proof of Security

We now prove security of Nsig relative to the definition given in [6]

Theorem 1. *Under the RSA assumption,* Nsig *is a secure network coding signature scheme when the hash function H is modeled as a random oracle.*

Proof. Given a forger $\mathcal{F}$ attacking Nsig, we build an algorithm $\mathcal{S}$ that solves the RSA problem. Here $\mathcal{S}$ stands for *simulator* and also for *source* since $\mathcal{S}$ will be simulating the actions of the source being attacked by $\mathcal{F}$.

Algorithm $\mathcal{S}$ receives input N, e, C where N, e are distributed as in an Nsig public key and $C \in_R \mathcal{QR}_N$. Its goal is to output $C^{1/e} \bmod N$. (Note that if $\mathcal{S}$ computes $C^{1/e} \bmod N$ for $C \in_R \mathcal{QR}_N$ with non-negligible probability, this contradicts the standard RSA assumption where C is chosen uniformly from $\mathbb{Z}_N^*$.) Algorithm $\mathcal{S}$ begins by choosing $i_0 \in_R \{1, \ldots, n\}$ and then setting $g_{i_0} := C$. For $i \neq i_0$, algorithm $\mathcal{S}$ chooses $r_i \in_R \mathcal{QR}_N$ and sets $g_i := r_i^e \bmod N$. Then $\mathcal{S}$ calls $\mathcal{F}$ on the public key $(N, e, g_1, \ldots, g_n)$.

$\mathcal{F}$ chooses a file, represented as a set of vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)}$, and requests a signature on it. In response, algorithm $\mathcal{S}$ chooses $\sigma_1, \ldots, \sigma_m \in_R \mathcal{QR}_N$ and $\mathsf{fid} \in \{0,1\}^k$, and then sets (using the programmability of the random oracle H)

$$h_i \overset{\text{def}}{=} H(i, \mathsf{fid}) := \sigma_i^e \prod_{j=1}^{n} g_j^{-\bar{v}_j^{(i)}} \bmod N. \tag{9}$$

(If fid was used previously to sign another file, $\mathcal{S}$ aborts. This occurs with negligible probability and we ignore it from here on.) Finally, $\mathcal{S}$ gives to $\mathcal{F}$ the signature σ_i on the augmented vector $\bar{w}^{(i)} = \bar{u}^{(i)} \parallel \bar{v}^{(i)}$ (for $i = 1, \ldots, m$), along with fid. It is easy to see that signatures are distributed exactly in the real experiment.

Say $\mathcal{F}$ outputs a forgery, i.e., a file id fid^*, a vector $w^* \notin \mathrm{span}\{\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}\}$ (where $\{\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}\}$ is the unique set of augmented vectors signed using fid^*), and a valid signature $\sigma^* = \mathsf{Nsig}(w^*)$ on w^*. We show how $\mathcal{S}$ can use this to solve its given RSA instance.

Denote $w^* = u^* \parallel v^* = (u_1^*, \ldots, u_m^*, v_1^*, \ldots, v_n^*)$, and define the vector

$$z^* = w^* - \sum_{i=1}^{m} u_i^* \bar{w}^{(i)}. \tag{10}$$

Note $z = (0, \ldots, 0, z_1, \ldots, z_n)$; that is, its first m coordinates are all zero. Moreover, since $w^* \notin \mathrm{span}\{\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}\}$ at least one of the values z_i is non-zero. With probability at least $1/n$ we thus have $z_{i_0} \neq 0$, and we assume this to be the case from now on.

By definition of z^* and the homomorphic property of Nsig we have:

$$\begin{aligned}\mathsf{Nsig}(z^*) &= \mathsf{Nsig}(w^* - \sum_{i=1}^{m} u_i^* \bar{w}^{(i)}) \\ &= \mathsf{Nsig}(w^*) \prod_{i=1}^{m} \mathsf{Nsig}(\bar{w}^{(i)})^{-u_i^*} = \sigma^* \prod_{i=1}^{m} \sigma_i^{-u_i^*} \bmod N.\end{aligned} \tag{11}$$

On the other hand, we can also represent $\mathsf{Nsig}(z^*)$ as

$$\begin{aligned}\mathsf{Nsig}(z^*) &= \left(\prod_{i=1}^{m} h_i^0 \prod_{i=1}^{n} g_i^{z_i} \right)^{1/e} \\ &= (C^{z_{i_0}})^{1/e} \prod_{i \neq i_0} (g_i^{z_i})^{1/e} = (C^{z_{i_0}})^{1/e} \prod_{i \neq i_0} r_i^{z_i} \bmod N.\end{aligned} \tag{12}$$

Combining Eqs. (11) and (12) we get that

$$(C^{z_{i_0}})^{1/e} = \sigma^* \prod_{i=1}^{m} \sigma_i^{-u_i^*} \prod_{i \neq i_0} r_i^{-z_i} \bmod N,$$

from which $\mathcal{S}$ can compute a value x such that $x^e = C^{z_{i_0}} \bmod N$. Using a standard trick, $\mathcal{S}$ can then compute $C^{1/e} \bmod N$ provided that $\gcd(z_{i_0}, e) = 1$. But this is the case since $e > mBM$ is prime and

$$-mBM \leq v_{i_0}^* - \sum_{i=1}^{m} u_i^* \bar{w}_{i_0}^{(i)} = z_{i_0} = v_{i_0}^* - \sum_{i=1}^{m} u_i^* \bar{w}_{i_0}^{(i)} \leq MB.$$

(Since w^* passes verification we have $0 \leq u_i^* \leq B$ and $0 \leq v_i^* \leq MB$; it always holds that $0 \leq \bar{w}_j^{(i)} \leq M$.) □

Remark 2. The above proof uses the fact that e is larger than the coordinates of valid vectors. Indeed, if coordinates larger than e are allowed then given a valid vector $w = u||v$ with signature σ an attacker can output the forged signature $\sigma' = \sigma \cdot g_1$ on the vector $w' = u||v'$ with $v' = v + (e, 0, \ldots, 0)$.

5 Homomorphic Hashing Modulo a Composite

Network coding signatures based on homomorphic hashing can offer significant computational advantages relative to constructions based on homomorphic signature schemes since, when using the former, a node that chooses not to verify an incoming vector (cf. Remark 1) need not perform any cryptographic operations. On the other hand, constructions based on homomorphic hashing consume more bandwidth since nodes now need to obtain the (authenticated) hash values of the original file vectors. If delivery of the hash values to nodes can be done in some out-of-band fashion, however, this drawback is mitigated.

Here we introduce a homomorphic hashing scheme, denoted $\mathbf{H}_N$, that is similar to the EHH scheme described in Section 2.2 but where operations are performed modulo a composite N. This results in the homomorphic properties holding over a group of unknown order; hence this scheme can only be applied when the underlying network coding is done over the integers. $\mathbf{H}_N$ has better computational efficiency than the EHH scheme (over prime-order groups) from Section 2.2; although $\mathbf{H}_N$ produces larger hash values, it requires a smaller public key than the EHH scheme.

In this section we once again assume linear network coding being performed over the integers as described in Section 3. Namely, the file to be transmitted is represented by vectors $\bar{v}^{(1)}, \ldots, \bar{v}^{(m)} \in \mathbb{Z}^n$, and intermediate nodes choose coefficients uniformly from a set $Q = \{0, \ldots, q-1\}$ (for some small prime q) and compute all linear combinations without any modular reduction.

Let N be the product of two safe primes so that the group $\mathcal{QR}_N$ of quadratic residues modulo N is cyclic, and let $g_1, \ldots, g_n$ be generators of $\mathcal{QR}_N$. For $v = (v_1, \ldots, v_n) \in \mathbb{Z}^n$, define

$$\mathbf{H}_N(v) = \prod_{j=1}^{n} g_j^{v_j} \bmod N.$$

This is a homomorphic hash function that is collision resistant if factoring N is hard (proof omitted). Thus, $\mathbf{H}_N$ can serve as a basis for a network coding signature scheme as discussed in Section 2.2. In particular, a node receiving a vector $w = (u_1, \ldots, u_m, v_1, \ldots, v_n)$ can verify it by checking whether

$$\prod_{i=1}^{m} h_i^{u_i} \stackrel{?}{=} \mathbf{H}_N(v) \stackrel{\text{def}}{=} \prod_{j=1}^{n} g_j^{v_j} \bmod N. \tag{13}$$

where $h_i = \mathbf{H}_N(\bar{v}^{(i)}), i = 1, \ldots, m$. Below we show a batch verification optimization that allows an intermediate vector to verify *all* of its incoming vectors with a *single* application of Eq. (13).

Bandwidth considerations for this scheme, which uses integer coefficients that grow over time, are similar to those of the RSA-based scheme from Section 4. An additional benefit of $\mathbf{H}_N$ is that there is no need to determine an *a priori* bound on these coefficients. As in the case of the RSA-based scheme from Section 4, one way to limit the effect of coordinate growth on the total communication is to set $n = 1$. As we now discuss, this not only reduces bandwidth overhead but also improves computational performance significantly.

Fix $n = 1$ so that each block of information $\bar{v}^{(i)}$ is a *single* (long) integer. Choosing N appropriately[6], we can take 2 as a generator of $\mathcal{QR}_N$, thus obtaining:

$$\mathbf{H}_N(v) = 2^v \bmod N.$$

This achieves the most salient advantage of the $\mathbf{H}_N$ scheme: *fast exponentiation.* Another advantage of this homomorphic hash is that it considerably improves the size of the public parameters relative to the EHH scheme. To see this, observe that in the EHH scheme the total length of the set of generators $g_1, \ldots, g_n$ included in the public parameters is (at least) $n \log p$ which is (at least) as large as each information vector $\bar{v}^{(i)}$. Moreover, the number of generators is usually very large; e.g., for vectors $\bar{v}^{(i)}$ of size 4KB and 160-bit p the EHH scheme needs 200 random generators.[7] In the case of $\mathbf{H}_N$, on the other hand, only one generator is needed and furthermore this generator can be fixed to 2; the public parameters need only include N.

Batch verification. The use of $\mathbf{H}_N$ for network coding can be further optimized by using batch verification at intermediate nodes similarly to the procedure described in Section 4.2 for the Nsig signature. Specifically, instead of verifying each incoming vector using Eq. (13), an intermediate node can just generate one outgoing vector as usual (i.e., as a random linear combination over $\{0, \ldots, q-1\}$ of the incoming vectors) and then apply Eq. (13) to the resultant vector. It can be shown that the probability that this single verification passes but one of the

[6] Choose $N = p_1 p_2$ $p_1 = 2p'_1 + 1, p_2 = 2p'_2 + 1$, and p_1, p_2, p'_1, p'_2 all prime; $p'_1, p'_2 = 3 \bmod 8$; and $p_1, p_2 = 7 \bmod 8$.

[7] The size of the public parameters can be reduced by using a hash function to compute the generators "on the fly." Besides necessitating the use of the random oracle model, this also introduces additional computational overhead.

incoming vectors was invalid (not in the span of $\bar{w}^{(1)}, \ldots, \bar{w}^{(m)}$) is at most $1/q$. The probability that t consecutive nodes will be foiled to accept invalid vectors is at most $1/q^t$. Thus a single verification per intermediate node suffices regardless of the number of incoming or outgoing vectors.

In all, we have shown that the homomorphic hashing scheme $\mathbf{H}_N$ leads to a computationally efficient network coding signature scheme whose security can be proven based on the factoring assumption in the standard model (and assuming the security of the signature scheme used to sign the hash values $h_1, \ldots, h_n$).

Acknowledgments. Research was sponsored by US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either 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 of this work for Government purposes, notwithstanding any copyright notation hereon.

References

1. Agrawal, S., Boneh, D.: Homomorphic MACs: MAC-based integrity for network coding. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536. Springer, Heidelberg (2009)
2. Ahlswede, R., Cai, N., Li, S., Yeung, R.: Network information flow. IEEE Transactions on Information Theory 46(4), 1204–1216 (2000)
3. Ateniese, G., Burns, R.C., Curtmola, R., Herring, J., Kissner, L., Peterson, Z.N.J., Song, D.X.: Provable data possession at untrusted stores. In: ACM Conference on Computer and Communications Security, pp. 598–609 (2007)
4. Ateniese, G., Kamara, S., Katz, J.: Proofs of storage from homomorphic identification protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 319–333. Springer, Heidelberg (2009)
5. Bellare, M., Garay, J., Rabin, T.: Fast batch verification for modular exponentiation and digital signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 236–250. Springer, Heidelberg (1998)
6. 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)
7. Charles, D., Jain, K., Lauter, K.: Signatures for network coding. In: 40th Annual Conference on Information Sciences and Systems, CISS 2006 (2006); To appear in International Journal of Information and Coding Theory
8. Chou, P.A., Wu, Y., Jain, K.: Practical network coding. In: 41st Allerton Conference on Communication, Control, and Computing (2003)
9. Gkantsidis, C., Rodriguez, P.: Cooperative security for network coding file distribution. In: Proc. of IEEE INFOCOM 2006, pp. 1–13 (2006)
10. Ho, T., Koetter, R., Médard, M., Karger, D., Effros, M.: The benefits of coding over routing in a randomized setting. In: Proc. of International Symposium on Information Theory, ISIT (2003)

11. Ho, T., Leong, B., Koetter, R., Médard, M., Effros, M., Karger, D.: Byzantine modification detection in multicast networks using randomized network coding. In: Proc. Intl. Symposium on Information Theory (ISIT), pp. 144–152 (2004)
12. Ho, T., Lun, D.: Network Coding: An Introduction. Cambridge University Press, Cambridge (2008)
13. Ho, T., Médard, M., Koetter, R., Karger, D.R., Effros, M., Shi, J., Leong, B.: A random linear network coding approach to multicast. IEEE Trans. Inform. Theory 52(10), 4413–4430 (2006)
14. Jaggi, S.: Design and Analysis of Network Codes. PhD thesis, California Institute of Technology (2006)
15. Jaggi, S., Langberg, M., Katti, S., Ho, T., Katabi, D., Médard, M., Effros, M.: Resilient network coding in the presence of Byzantine adversaries. IEEE Trans. on Information Theory 54(6), 2596–2603 (2008)
16. 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)
17. Krohn, M., Freedman, M., Mazieres, D.: On the-fly verification of rateless erasure codes for efficient content distribution. In: Proc. IEEE Symposium on Security & Privacy, pp. 226–240 (2004)
18. Li, S.-Y.R., Yeung, R.W., Cai, N.: Linear network coding. IEEE Trans. Inform. Theory 49(2), 371–381 (2003)
19. Shacham, H., Waters, B.: Compact proofs of retrievability. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 90–107. Springer, Heidelberg (2008)
20. Yu, Z., Wei, Y., Ramkumar, B., Guan, Y.: An efficient signature-based scheme for securing network coding against pollution attacks. In: INFOCOM (2008)
21. Zhao, F., Kalker, T., Médard, M., Han, K.: Signatures for content distribution with network coding. In: Proc. Intl. Symp. on Information Theory ISIT (2007)

Preventing Pollution Attacks in Multi-source Network Coding

Shweta Agrawal[1,⋆], Dan Boneh[2,⋆⋆],
Xavier Boyen[3,⋆⋆⋆], and David Mandell Freeman[2,†]

[1] University of Texas at Austin, USA
shweta.a@gmail.com
[2] Stanford University, USA
{dabo,dfreeman}@cs.stanford.edu
[3] Université de Liège, Belgium
xb@boyen.org

Abstract. Network coding is a method for achieving channel capacity in networks. The key idea is to allow network routers to linearly mix packets as they traverse the network so that recipients receive linear combinations of packets. Network coded systems are vulnerable to pollution attacks where a single malicious node floods the network with bad packets and prevents the receiver from decoding correctly. Cryptographic defenses to these problems are based on homomorphic signatures and MACs. These proposals, however, cannot handle mixing of packets from multiple sources, which is needed to achieve the full benefits of network coding. In this paper we address integrity of multi-source mixing. We propose a security model for this setting and provide a generic construction.

1 Introduction

Network coding [3,16] is an elegant technique that replaces the traditional "store and forward" paradigm of network routing by a method that allows routers to transform the received data before re-transmission. It has been established that for certain classes of networks, *random linear* coding is sufficient to improve throughput [11]. In addition, linear network codes offer robustness and adaptability and have many practical applications (in wireless and sensor networks, for example) [10]. Due to these advantages, network coding has become very popular.

On the other hand, networks using network coding are exposed to problems that traditional networks do not face. A particularly important instance of this is the *pollution* problem: if some routers in the network are malicious and forward invalid combinations of received packets, then these invalid packets get

⋆ Supported by DARPA IAMANET.
⋆⋆ Supported by DARPA IAMANET and NSF.
⋆⋆⋆ Work done in part at Stanford, with support from DARPA IAMANET and NSF.
† Work done in part at CWI and Univ. Leiden. Supported by an NSF International Research Fellowship and the NSF Office of Multidisciplinary Activities.

mixed with valid packets downstream and quickly pollute the whole network. In addition, the receiver who obtains multiple packets has no way of ascertaining which of these are valid and should be used for decoding. Indeed, using even one invalid packet during the decoding process causes all the messages to be decoded wrongly. For a detailed discussion of pollution attacks, we refer the reader to [4,19,12].

To prevent the network from being flooded with invalid packets, it is desirable to have "hop-by-hop containment." This means that even if a bad packet gets injected into the network, it is detected and discarded at the very next hop. Thus, it can be dropped before it is combined with any other packets, preventing its pollution from spreading.

Hop-by-hop containment cannot be achieved by standard signatures or MACs. As pointed out in [1], signing the message packets does not help since recipients do not have the original message packets and therefore cannot verify the signature. Nor does signing the entire message prior to transmission work, because it forces the recipient to decode exponentially many subsets of received packets to find a decoded message with a consistent signature. Thus, new integrity mechanisms are needed to mitigate pollution attacks.

Previous Work. Security of network coding has been considered from both the information-theoretic and cryptographic perspectives. In the former, the adversary is modelled as having control over a limited number of links in the network. Such approaches, though useful for wireline networks, have limited application in wireless networks. For a detailed discussion of these techniques, see e.g. [6,9,13,14]. Cryptographic techniques have also been proposed, e.g. in [7,17,19,4]. These authors construct digital signatures for signing a linear subspace. If V is a subspace and σ its signature, then there is a verification algorithm which accepts the pair $(\mathbf{v}, \sigma)$ for all $\mathbf{v} \in V$, but it is difficult to construct a vector $\mathbf{y} \notin V$ for which the pair $(\mathbf{y}, \sigma)$ verifies. An alternative approach is to use a MAC (instead of a signature) for integrity of a linear subspace; see [1,18].

While the signature and MAC schemes in [7,17,19,4,1] are elegant, they are quite limited: they only allow routers to combine vectors from a single sender. (Furthermore, the constructions of [7,17,19] require a new public key to be generated for each file, thus hurting efficiency.) Traditional network coding assumes a network where many senders simultaneously send messages and network routers linearly combine vectors from multiple senders. This setting is essential in showing that network coding can improve the efficiency of 802.11 wireless networks [15].

Our Contribution. Our goal is to construct a signature mechanism that provides integrity when network routers combine packets from many sources. This problem is considerably harder than the single source problem. First, defining security is more difficult. It is necessary to model "insider" attacks where the attacker controls network routers as well as some senders. The attacker's goal is to generate valid signatures on mixed packets; after decoding these packets the recipient believes that an honest sender sent a message M^* that was never sent by the honest sender.

More precisely, if there are s senders in the network, we allow the attacker to control $s-1$ of them. Furthermore, the attacker can mount a chosen message attack on the single honest sender. The attacker's goal is to generate a mixed packet with a valid signature that after decoding corresponds to an existential forgery on the single honest sender.

In Section 3 we show that a natural generalization of the single-sender security model in [4] to the multi-sender setting results in a model that cannot be satisfied. We do this by constructing a generic attack against an abstract multi-source network coding signature scheme. In Section 4 we present a security model that captures the constraints of the multi-sender problem. Our model retains the desirable properties of the single-source model, such as hop-by-hop containment of forged packets, and is achievable.

In Section 5 we present a construction satisfying our security model. We give a generic construction from a new primitive called a *vector hash*, which captures the properties of homomorphic hashing that are necessary to produce secure signatures. In the full version of this paper [2] we show how to instantiate the construction based on the discrete logarithm assumption. We also prove a lower bound that shows that our model necessitates a relatively space-inefficient construction; our discrete log scheme (asymptotically) achieves this lower bound.

2 Network Coding

We refer the reader to [16] for a detailed introduction to network coding. Here we present a brief overview for completeness; this description describes the operation of a network coding system and is independent of any security model. We model a network as a directed graph consisting of a set of vertices (or *nodes*) V and a set of edges E. We assume the graph is connected. A node that only transmits data is called a *source node*. We start with the basic model, in which one source wishes to transmit one file F through the network. The source interprets the data in F as a set of m vectors $\hat{\mathbf{v}}_1, \ldots, \hat{\mathbf{v}}_m$ in an n-dimensional vector space over a finite field $\mathbb{F}_p$. (The prime p and the dimensions n and m are fixed parameters in the system.) We sometimes refer to individual vectors as *blocks* or *packets*. The source then appends a unit vector of length m to the vectors $\hat{\mathbf{v}}_i$ to create m *augmented vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_m$ given by

$$\mathbf{v}_i = (\text{—}\hat{\mathbf{v}}_i\text{—}, \overbrace{\underbrace{0, \ldots, 0, 1}_{i}, 0, \ldots, 0}^{m}) \in \mathbb{F}_p^{n+m}.$$

The augmented vectors comprise the data to be transmitted through the network. We call the first n entries of the vector $\mathbf{v}_i$ the *data component* and the last m entries the *augmentation component*.

The "coding" part of network coding works as follows: an intermediate node in the network receives some set of vectors $\mathbf{w}_1, \ldots, \mathbf{w}_\ell$, chooses ℓ random elements $\beta_i \in \mathbb{F}_p$, and transmits the vector $\mathbf{y} = \sum_{i=1}^{\ell} \beta_i \mathbf{w}_i$ along its outgoing edges. The key property of the augmentation is that the augmentation component contains exactly the linear combination coefficients used to construct $\mathbf{y}$. That is, we know that $\mathbf{y} = \sum_{i=1}^{m} y_{n+i}\mathbf{v}_i$ even though the intermediate node may never see the $\mathbf{v}_i$. This property allows any node that receives a set of m linearly independent vectors $\mathbf{y}_1, \ldots, \mathbf{y}_m$ to recover the original $\mathbf{v}_i$. Specifically, if we let D be $m \times n$ matrix whose ith row consists of the data component of $\mathbf{y}_i$, and A be the $m \times m$ matrix whose ith row consists of the augmentation component of $\mathbf{y}_i$, then the rows of $A^{-1}D$ are exactly the initial vectors $\hat{\mathbf{v}}_i$.

Since network coding consists of linearly combining vectors, the subspace spanned by the (augmented) vectors of a file remains invariant under network operations. Hence we can equivalently consider a file to be represented by the subspace spanned by the vectors that comprise it.

Notation. We use n to denote the dimension of the data space and m to denote the dimension of a vector subspace that represents a single file. The number of files in the system is denoted by f. For $\mathbf{v} \in \mathbb{F}_p^{n+\ell}$, we will use $\hat{\mathbf{v}}$ to denote the *data component* of $\mathbf{v}$, i.e., the first n coordinates of $\mathbf{v}$, and $\overline{\beta}_{\mathbf{v}}$ to denote the *augmentation component* of $\mathbf{v}$, i.e., the remaining ℓ coordinates. When we use a vector space V as input to or output of an algorithm we assume that V is described by an explicit basis $\{\mathbf{v}_1, \ldots, \mathbf{v}_\ell\}$. Such a basis is *properly augmented* if for $i = 1, \ldots, \ell$, the augmentation component $\overline{\beta}_{\mathbf{v}_i}$ is the unit vector $\mathbf{e}_i$ with a 1 in the ith place.

We will refer to the augmented vectors that the source wishes to transmit as *primitive* vectors. Here, "primitive" alludes to the fact that these vectors have not been mixed with any other; their augmentation components are unit vectors. *Aggregate* vectors, on the other hand, refer to vectors that have been formed as a result of linearly combining primitive or other aggregate vectors.

2.1 Multiple Sources, Multiple Files

In general networks may have multiple sources, each of which can transmit multiple files into the network. We now describe this situation, assuming that all nodes in the network are honest. In principle the network coding setup is the same as in the single-source situation described above, but there is some more bookkeeping to do. This bookkeeping is implicit in previous work that considers multiple sources (e.g. [16]); here we give an explicit description that we will use in our discussion of security. The complication arises from the fact that the intermediate nodes wish to combine vectors from files produced by different sources, but each source knows nothing of what the other sources are doing.

In the single source case, each file is associated with a file identifier id. The identifier allows the receiver to group together packets that belong to the same file. This prevents, for example, delayed honest packets from a previous file transmission from being decoded along with the current file's vectors. Hence each vector (primitive or aggregate) that traverses the system carries with it the identifier of the file it belongs to.

In the multi-source case, the file identifier id plays an even more crucial role — it allows the intermediate nodes to combine vectors arising from different files. In this scenario, an aggregate vector may be associated with multiple files, and the identifier attached to an aggregate vector $\mathbf{v}$ must carry with it the identifiers of all of the files whose vectors went into making $\mathbf{v}$. Upon receiving two vectors, where each vector contains a (probably different) list of identifiers $\overline{\mathsf{id}}$, an intermediate node will need to "merge" the lists of identifiers to a common list and adjust the two vectors' augmentation components so that they can be linearly combined.

For example, suppose a node receives two vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{F}_p^{n+m}$ with identifiers id_1 and id_2, respectively. Splitting $\mathbf{v}_i$ into its data and augmentation components, we write $\mathbf{v}_i = (\hat{\mathbf{v}}_i, \mathbf{a}_i)$. If $\mathsf{id}_1 = \mathsf{id}_2$ then the vectors come from the same file and the situation is analogous to the single source case and no additional adjustment is needed. However, if $\mathsf{id}_1 \neq \mathsf{id}_2$ then the vectors came from different files and we must introduce additional augmentation before we can linearly combine the vectors. In this case we define $\mathbf{v}_1' = (\hat{\mathbf{v}}_1, \mathbf{a}_1, \mathbf{0})$ and $\mathbf{v}_2' = (\hat{\mathbf{v}}_2, \mathbf{0}, \mathbf{a}_2) \in \mathbb{F}_p^{n+2m}$, where $\mathbf{0}$ denotes a length-m zero vector. Thus when we compute a linear combination $\mathbf{v} = a\mathbf{v}_1' + b\mathbf{v}_2'$, the data components are mixed together but the augmentation coefficients remain separate. We can then use the identifier $\overline{\mathsf{id}} = (\mathsf{id}_1, \mathsf{id}_2)$ to indicate which set of augmentation coefficients correspond to which file.

More generally, we define an algorithm Merge that merges the lists of identifiers contained in aggregate vectors and adjusts the vectors' augmentations. This algorithm is intrinsic to the multiple-source setting: the algorithm does not itself linearly combine vectors, but rather it *prepares* aggregate vectors (coming from different sources, made up of different files) to be mixed together. If $\mathbf{v} \in \mathbb{F}_p^{n+mf}$ is an aggregate vector, we continue to call the first n entries of $\mathbf{v}$ the data component; we call the rest of $\mathbf{v}$ the augmentation component, and we divide the augmentation component into f *augmentation blocks* of length m. (Here and in the remainder of the paper we assume for simplicity the dimension m is the same for each file and is publicly known; the generalization to variable dimension is straightforward.)

Algorithm 1. (Merge)
Input: lists of identifiers $\overline{\mathsf{id}}_1, \overline{\mathsf{id}}_2$ of lengths f_1, f_2, respectively, with no repeated entries, and vectors $\mathbf{w}_i \in \mathbb{F}_p^{n+mf_i}$ for $i = 1, 2$.
Output: vectors $\mathbf{w}_1', \mathbf{w}_2' \in \mathbb{F}_p^{n+mf'}$ and a list of identifiers $\overline{\mathsf{id}}'$ of length f'.

1. Let $\overline{\mathsf{id}}'$ be the list whose entries are the union of the elements of $\overline{\mathsf{id}}_1$ and $\overline{\mathsf{id}}_2$, ordered in some pre-determined way (e.g. lexicographically). Let f' be the length of $\overline{\mathsf{id}}'$.
2. For i in $1, 2$, define $\mathbf{w}_i' \in \mathbb{F}_p^{n+mf'}$ by setting the data component of $\mathbf{w}_i'$ equal to the data component of $\mathbf{w}_i$, and for j in $1, \dots, f'$, setting the jth augmentation block of $\mathbf{w}_i'$ as follows:
   - If the jth element of $\overline{\mathsf{id}}'$ is the kth element of $\overline{\mathsf{id}}_i$, the jth augmentation block of $\mathbf{w}_i'$ is equal to the kth augmentation block of $\mathbf{w}_i$.
   - If the jth element of $\overline{\mathsf{id}}'$ is not an element of $\overline{\mathsf{id}}_i$, the jth augmentation block of $\mathbf{w}_i'$ is $\mathbf{0}$.
3. Output the list $\overline{\mathsf{id}}'$ and the vectors $\mathbf{w}_1', \mathbf{w}_2'$.

The intermediate node can now compute a random linear combination $\mathbf{y}$ of the $\mathbf{w}_1'$ and $\mathbf{w}_2'$ and use the list $\overline{\mathsf{id}}'$ as the identifier component of the signature on $\mathbf{y}$. (In the example above we executed this algorithm on two vectors each with an identifier list of length $f_i = 1$.)

We also define an algorithm called MergeSpaces that uses the Merge algorithm to combine two files described as vector spaces.

Algorithm 2. (MergeSpaces)
Input: disjoint lists of identifiers $\overline{\mathsf{id}}_1, \overline{\mathsf{id}}_2$ and two vector spaces $V = \text{span}(\mathbf{v}_1, \ldots, \mathbf{v}_k) \subset \mathbb{F}_p^{n+k}$ and $W = \text{span}(\mathbf{w}_1, \ldots, \mathbf{w}_\ell) \subset \mathbb{F}_p^{n+\ell}$.
Output: a subspace $Z \subset \mathbb{F}_p^{n+k+\ell}$ and an identifier $\overline{\mathsf{id}}'$.

1. Let B be the set of nonzero vectors produced by

$$\mathsf{Merge}(\overline{\mathsf{id}}_1, \overline{\mathsf{id}}_2, \mathbf{v}_1, \mathbf{0}), \ldots, \mathsf{Merge}(\overline{\mathsf{id}}_1, \overline{\mathsf{id}}_2, \mathbf{v}_k, \mathbf{0}),$$
$$\mathsf{Merge}(\overline{\mathsf{id}}_1, \overline{\mathsf{id}}_2, \mathbf{0}, \mathbf{w}_1), \ldots, \mathsf{Merge}(\overline{\mathsf{id}}_1, \overline{\mathsf{id}}_2, \mathbf{0}, \mathbf{w}_\ell).$$

2. Let $\overline{\mathsf{id}}'$ be the identifier output by any of the calls to Merge in Step (1).
3. Output $Z = \text{span}(B)$ and $\overline{\mathsf{id}}'$.

By applying MergeSpaces repeatedly using concatenated lists of identifiers, the algorithm generalizes to take any number of vector spaces and identifiers as input. The decoding operation works as before: given a set of vectors whose (merged) augmentation components form a full-rank matrix, we can recover the original data vectors by inverting this matrix.

3 Signatures and File Identifiers

For single sources, a network coding signature scheme consists of three algorithms, Setup, Sign, and Verify, whose functionality correspond to the usual notions for a signature scheme. In this setting, the Sign algorithm produces signatures on a *vector space*, and the Verify algorithm checks whether the signature is valid on a given *vector*. In addition, both Sign and Verify take as additional input a file identifier id, which binds a signature to a file. Informally, the correctness condition is that if σ is a signature on a vector space V with identifier id, then for all $\mathbf{v} \in V$, $\mathsf{Verify}(\mathsf{id}, \mathbf{v}, \sigma)$ outputs "accept." (For formal definitions, see [4, Section 3.1].)

For multiple sources, we need to add an additional algorithm Combine that will be used by intermediate routers to produce signatures on vectors that are linear combinations of vectors from different files. More precisely, Combine takes as input two tuples $(\mathbf{v}_i, \overline{\mathsf{id}}_i, \sigma_i, a_i)$ for $i = 1, 2$, where $\mathbf{v}_i$ are vectors, $\overline{\mathsf{id}}_i$ are (lists of) identifiers, σ_i are signatures, and a_i are network coding coefficients. The algorithm outputs a signature σ'. The correctness condition is that if σ_i is a valid signature on $\mathbf{v}_i$ with identifier $\overline{\mathsf{id}}_i$ for $i = 1, 2$, then σ' is a valid signature on $a_1\mathbf{v}_1' + a_2\mathbf{v}_2'$ with identifier $\overline{\mathsf{id}}'$, where $\mathbf{v}_1', \mathbf{v}_2', \overline{\mathsf{id}}'$ are output by $\mathsf{Merge}(\overline{\mathsf{id}}_1, \overline{\mathsf{id}}_2, \mathbf{v}_1, \mathbf{v}_2)$.

In the single-source setting, the Sign algorithm takes id as input. Thus, a vector $\mathbf{v}$ carries a pair (id, σ) where id is the file identifier chosen arbitrarily, and σ is generated by Sign. In the multi-source case however, allowing senders to pick file identifiers gives them the ability to frame other users in the system, so that receiver Bob can be made to believe that user Alice sent him a packet which, in fact, Alice did not. In most network coded systems with multiple senders, such as BitTorrent [8], insider attacks form the real threat, so this attack has significant practical implications. Fortunately, this attack can be thwarted by enforcing that the file identifiers be *cryptographically verifiable*. In the subsequent sections, we will formalize these notions. We first describe the attack, and then use the intuition gained from the attack to construct a framework that can circumvent it.

3.1 Generic Attack (For Arbitrary File Identifiers)

Here we construct an attack against an abstract multi-source network coding signature scheme that consists of the algorithms Setup, Sign, $\mathsf{Combine}$, Verify discussed above. We make no assumptions about these algorithms beyond their functionality. We show that it is impossible to achieve hop-by-hop containment if the identifier id is chosen arbitrarily by the sender and is given as input to the Sign algorithm. We construct a generic attack in which an intermediate node is fooled into accepting invalid packets as valid. As mentioned before, the attack is an "insider" attack where one of the senders is malicious. The malicious sender can assign two different vector spaces the same id and sign both using his secret key. An intermediate node has no hope of ever detecting this, since two packets constructed using these two vector spaces are both individually valid, but they are not pairwise valid, and can cause the receiver to incorrectly decode an honest user's message. We make this formal below.

We explain the attack with subspace dimension $m = 1$; the attack easily generalizes to arbitrary m. In our system, the honest sender is Alice, the receiver is Bob, and the malicious user is Mallet.

Honest User Alice. Alice wishes to send a file described as a single nonzero vector $\hat{\mathbf{v}}_1 \in \mathbb{F}_p^n$. She sets $\mathbf{v}_1 = (\hat{\mathbf{v}}_1, 1)$, chooses a file identifier id_α and uses her secret key sk_α to create a signature τ_1 on the one-dimensional subspace $V_1 \subset \mathbb{F}_p^{n+1}$ spanned by $\mathbf{v}_1$, with identifier id_α. Then she transmits the packet $P_1 = (\mathbf{v}_1, \mathsf{id}_\alpha, \tau_1)$.

Malicious User Mallet. Mallet receives P_1 and does the following:

1. Generate a key pair $(\mathsf{sk}_\mu, \mathsf{pk}_\mu)$.
2. Pick two vectors $\hat{\mathbf{v}}_2, \hat{\mathbf{v}}_3 \in \mathbb{F}_p^n$ such that the set $\{\hat{\mathbf{v}}_1, \hat{\mathbf{v}}_2, \hat{\mathbf{v}}_3\}$ are linearly independent. Let V_2, V_3 be the subspaces of $\mathbb{F}_p^{n+1}$ spanned by $\mathbf{v}_2 = (\hat{\mathbf{v}}_2, 1)$ and $\mathbf{v}_3 = (\hat{\mathbf{v}}_3, 1)$, respectively.
3. Choose an identifier $\mathsf{id}_\mu \neq \mathsf{id}_\alpha$, and use the key sk_μ to compute signatures τ_2, τ_3 on subspaces V_2, V_3 with identifier id_μ. Create the packets $P_2 = (\mathbf{v}_2, \mathsf{id}_\mu, \tau_2)$, $P_3 = (\mathbf{v}_3, \mathsf{id}_\mu, \tau_3)$.
4. Run Merge on $(\mathbf{v}_1, \mathbf{v}_2)$ and $(\mathsf{id}_\alpha, \mathsf{id}_\mu)$ to obtain $\overline{\mathsf{id}} = (\mathsf{id}_\alpha, \mathsf{id}_\mu)$ and vectors $\mathbf{v}'_1 = (\hat{\mathbf{v}}_1, 1, 0)$, $\mathbf{v}'_2 = (\hat{\mathbf{v}}_2, 0, 1)$.

5. Run $\mathsf{Combine}\big((\mathbf{v}_1, \mathsf{id}_\alpha, \tau_1, 1), (\mathbf{v}_2, \mathsf{id}_\mu, \tau_2, 1)\big)$ to produce a signature τ_4 on the vector $\mathbf{v}_4 = \mathbf{v}_1' + \mathbf{v}_2' = (\hat{\mathbf{v}}_1 + \hat{\mathbf{v}}_2, 1, 1) \in \mathbb{F}_p^{n+2}$. Let $P_4 = (\mathbf{v}_4, \overline{\mathsf{id}}, \tau_4)$.
6. Send P_3 and P_4 to Bob.

Receiver Bob. Bob receives P_3 and P_4, each of which pass the verification test (by the correctness of Sign and Combine). Bob then tries to decode the received data to recover Alice's file.

The identifier $\overline{\mathsf{id}} = (\mathsf{id}_\alpha, \mathsf{id}_\mu)$ indicates that $\mathbf{v}^* = \mathbf{v}_4 - (\hat{\mathbf{v}}_3, 0, 1)$ is a primitive vector sent by Alice, since the augmentation component of $\mathbf{v}^*$ is $(1, 0)$. However, the data part of $\mathbf{v}^*$ is $\hat{\mathbf{v}}_1 + \hat{\mathbf{v}}_2 - \hat{\mathbf{v}}_3$, which cannot be in the subspace spanned by $\hat{\mathbf{v}}_1$ since $\{\hat{\mathbf{v}}_1, \hat{\mathbf{v}}_2, \hat{\mathbf{v}}_3\}$ are linearly independent. Thus $\mathbf{v}^*$ is an invalid vector accepted by Bob.

In the above attack, Mallet was able to frame Alice by secretly reusing id_μ for two different vector spaces. Note that this attack is more insidious than simply inserting data with identifier id_α, which would have the same effect of corrupting Alice's data. We see from this attack that arbitrary file identifiers provide a malicious insider too much power. It is thus necessary to tie the identifiers *cryptographically* to the files they represent, in a way that is verifiable at every node in the network. In particular, the Sign algorithm should output both an identifier id and a signature σ. To verify the identifier we use an algorithm IdTest that takes as input a public key pk, a vector $\mathbf{y}$, and a list of identifiers $\overline{\mathsf{id}}$, and outputs "accept" if $\mathbf{y}$ is in the subspace V identified by $\overline{\mathsf{id}}$. To avoid the above attack, the following tasks must be infeasible for Mallet:

1. Given a public key pk_α, find an identifier id_α and a vector $\mathbf{y}$ such that $\mathsf{IdTest}(\mathsf{pk}_\alpha, \mathbf{y}, \overline{\mathsf{id}}_\alpha)$ outputs "accept." (This is a type of "collision-resistance" property.)
2. Given a vector space V, a public key pk_α, and $(\mathsf{id}_\alpha, \sigma) := \mathsf{Sign}(\mathsf{sk}_\alpha, V)$ (where sk_α is the secret key corresponding to pk_α), find a $\mathbf{y} \notin V$ such that $\mathsf{IdTest}(\mathsf{pk}_\alpha, \mathbf{y}, \mathsf{id}_\alpha)$ outputs "accept." (This property is unique to the network coding scenario.)

If Mallet can succeed at either task, then Bob is convinced that the vector $\mathbf{y}$ belongs to a file sent by Alice, when in fact it does not. (Indeed, in the first case Alice didn't even send a file!)

These two tasks are quite familiar: they are analogous to the two ways of breaking a single-source network coding signature scheme [4, Section 3.1]. This analysis leads to our key observation: **the file identifier produced by Sign must itself be a vector space signature**. It follows that all the security properties of the system are carried in the identifier id, so we can set the "signature" part σ equal to id or eliminate σ entirely. We formalize these ideas in the following section.

Remark 3. One can show that allowing the use of arbitrary file identifiers not only makes hop-by-hop containment impossible, but also forces the receiver to solve the clique problem for proper decoding. Specifically, there is a formal reduction from the clique problem to decoding in multi-source network coding; details are in the full paper [2].

4 Network Coding Signatures

We formally define the multi-source network coding signature scheme. Here the Sign algorithm generates an element σ that is used both as a signature and a file identifier. The Verify algorithm implements the functionality of the IdTest algorithm in the previous section and allows every node to validate the identifier/signature of an incoming packet. Since signatures and identifiers play the same role, the $\mathsf{Combine}$ algorithm provides the same functionality as the Merge algorithm of Section 2, while also keeping track of the public keys involved. Note that in contrast to traditional signatures, the Verify algorithm does *not* take as input the original message (i.e., vector space).

Definition 4. A multi-source network coding signature scheme is a tuple of five PPT algorithms, $\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Combine}, \mathsf{Verify}$, with the following properties:

$\mathsf{Setup}(1^\lambda, n, m)$**:** On input the unary representation of a security parameter 1^λ, a data space dimension n, and a subspace dimension m, outputs a description of system parameters params. This description includes the prime p used to define the field over which vector spaces are defined, as well as n and m.

$\mathsf{KeyGen}(\mathsf{params})$**:** Outputs a randomly generated user key pair $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{params}, \mathsf{sk}, V)$**:** On input a secret key sk and a subspace $V \subset \mathbb{F}_p^{n+m}$, outputs a signature σ.

$\mathsf{Combine}\big(\mathsf{params}, (\mathbf{v}_1, \boldsymbol{\sigma}_1, \overline{\mathsf{pk}}_1, a_1), (\mathbf{v}_2, \boldsymbol{\sigma}_2, \overline{\mathsf{pk}}_2, a_2)\big)$**:** Takes as input two vectors $\mathbf{v}_1 \in \mathbb{F}_p^{n+mf_1}$ and $\mathbf{v}_2 \in \mathbb{F}_p^{n+mf_2}$, two lists of signatures $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2$, two lists of public keys $\overline{\mathsf{pk}}_1, \overline{\mathsf{pk}}_2$, and two coefficients $a_1, a_2 \in \mathbb{F}_p$. The algorithm outputs a list of signatures $\boldsymbol{\sigma}$ and a list of public keys $\overline{\mathsf{pk}}$.

$\mathsf{Verify}\big(\mathsf{params}, \overline{\mathsf{pk}}, \mathbf{v}, \boldsymbol{\sigma}\big)$**:** On input a list of public keys $\overline{\mathsf{pk}}$, a vector $\mathbf{v} \in \mathbb{F}_p^{n+mf}$, and a list of signatures $\boldsymbol{\sigma}$, outputs $\top$ (accept) or $\perp$ (reject).

Correctness. We require that for any set of system parameters determined by $\mathsf{Setup}(1^\lambda, n, m)$, the following hold:

1. For primitive signatures: Consider a key pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{params})$ and a vector space $V \subset \mathbb{F}_p^{n+m}$. Let σ be the output of $\mathsf{Sign}(\mathsf{params}, \mathsf{sk}, V)$. Let $\overline{\mathsf{pk}} = \{\mathsf{pk}\}$ and $\boldsymbol{\sigma} = \{\sigma\}$. Then for all $\mathbf{v} \in V$, we require that $\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}, \mathbf{v}, \boldsymbol{\sigma}) = \top$.
2. Recursively, for combined signatures: Consider two lists of public keys $\overline{\mathsf{pk}}_1, \overline{\mathsf{pk}}_2$, two vectors $\mathbf{v}_1, \mathbf{v}_2$, two lists of signatures $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2$ such that

$$\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}_1, \mathbf{v}_1, \boldsymbol{\sigma}_1) = \mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}_2, \mathbf{v}_2, \boldsymbol{\sigma}_2) = \top.$$

   Let $\mathbf{v}_1', \mathbf{v}_2', \boldsymbol{\sigma}'$ be the output of $\mathsf{Merge}(\mathbf{v}_1, \mathbf{v}_2, \boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$. For any $a_1, a_2 \in \mathbb{F}_p$, we require that if $\boldsymbol{\sigma}, \overline{\mathsf{pk}}$ is the output of the $\mathsf{Combine}$ algorithm on inputs $(\mathbf{v}_1, \boldsymbol{\sigma}_1, \overline{\mathsf{pk}}_1, a_1), (\mathbf{v}_2, \boldsymbol{\sigma}_2, \overline{\mathsf{pk}}_2, a_2)$, then:

(a) $\boldsymbol{\sigma}' = \boldsymbol{\sigma}$,
(b) For j in $1, \ldots, f = |\boldsymbol{\sigma}|$, if the jth element of $\boldsymbol{\sigma}$ is the kth element of $\boldsymbol{\sigma}_i$ for $i \in \{1, 2\}$, then the jth element of $\overline{\mathsf{pk}}$ is the kth element of $\overline{\mathsf{pk}}_i$.
(c) $\mathsf{Verify}\big(\mathsf{params}, \overline{\mathsf{pk}}', a_1\mathbf{v}_1' + a_2\mathbf{v}_2', \boldsymbol{\sigma}\big) = \top$.

In the second correctness condition, (a) tells us that identifiers and signature play the same role, while (b) requires that the list of public keys produced by Combine corresponds (in a natural way) to the list of identifiers produced by Merge.

4.1 Security

The security game captures the fact that if the system is secure, even an attacker who controls all sources but one and is given a chosen message oracle for the honest source cannot create an existential forgery on the honest source. The game between a challenger and an adversary $\mathcal{A}$ with respect to a signature scheme $\mathcal{S}$ proceeds as follows.

Init. The challenger runs $\mathsf{Setup}(1^\lambda, n, m)$ to obtain system parameters params and runs $\mathsf{KeyGen}(\mathsf{params})$ to obtain sk^* and pk^*. It sends pk^* and params to $\mathcal{A}$. It keeps sk^* to itself.

Signature queries. $\mathcal{A}$ adaptively requests signatures for vector spaces $V_1, \ldots, V_\ell \subset \mathbb{F}_p^{n+m}$. The challenger responds by computing $\mathsf{Sign}(\mathsf{params}, \mathsf{sk}^*, V_i)$ for $i = 1, \ldots, \ell$ and sends the resulting signatures to $\mathcal{A}$.

Forgery attempt. $\mathcal{A}$ eventually outputs a 4-tuple $(\overline{\mathsf{pk}}^\dagger, \mathbf{v}^\dagger, \boldsymbol{\sigma}^\dagger, W^\dagger)$, where $\overline{\mathsf{pk}}^\dagger$ is a list of f (not necessarily distinct) public keys $\overline{\mathsf{pk}}^\dagger = (\mathsf{pk}_1, \ldots, \mathsf{pk}_f)$ that contains the challenge public key pk^*, $\mathbf{v}^\dagger$ is a nonzero vector in $\mathbb{F}_p^{n+mf}$, $\boldsymbol{\sigma}^\dagger$ is list of f signatures, and $W^\dagger = \text{span}\{\mathbf{w}_1, \ldots, \mathbf{w}_t\} \subset \mathbb{F}_p^{n+t}$ for some t.

Adjudication. Let $\boldsymbol{\sigma}^\dagger = (\sigma_1, \ldots, \sigma_f)$ be the list of (distinct) identifiers output by $\mathcal{A}$, where, w.l.o.g. we assume the first k components $\sigma_1, \ldots, \sigma_k$ are returned as the signatures for the chosen message queries $V_1, \ldots, V_k$, $k \le \ell$. Let $\boldsymbol{\sigma}_w$ be the last $f - k$ elements of $\boldsymbol{\sigma}^\dagger$. Let V^* be the vector space output by $\mathsf{MergeSpaces}(V_1, \ldots, V_k, W^\dagger, \sigma_1, \ldots, \sigma_k, \boldsymbol{\sigma}_w)$.

The forger wins the game if $\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}^\dagger, \mathbf{v}^\dagger, \boldsymbol{\sigma}^\dagger) = \top$ and at least one of the following two conditions holds:
1. There exists i in $1, \ldots, f$ such that the ith component of $\mathsf{pk}^\dagger$ is equal to pk^*, but σ_i is not any of the signatures obtained in response to chosen message queries.
2. For $i = 1, \ldots, t$, we have $\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}_w, \mathbf{w}_i, \boldsymbol{\sigma}_w) = \top$, but $\mathbf{v}^\dagger \notin V^*$.

Definition 5. The *advantage* NC-Adv$[\mathcal{A}, \mathcal{S}]$ of $\mathcal{A}$ is defined to be the probability that $\mathcal{A}$ wins the security game. A multi-source network coding scheme $\mathcal{S}$ is *secure* if for all probabilistic, polynomial-time adversaries $\mathcal{A}$ the advantage NC-Adv$[\mathcal{A}, \mathcal{S}]$ is negligible in the security parameter λ.

In the security game, the attacker requests signatures for files $V_1, \ldots, V_k$ and creates his own file $W^\dagger$. Intuitively, $W^\dagger$ corresponds to the vector space (the set of files) whose data the adversary mixes with the honest user's data in order to frame the honest user. Winning condition (1) implies that the attacker can create a valid fake signature for one of the files that he requests signatures for, i.e., for a file signed with sk^*. Winning condition (2) implies that the attacker can produce a fake file $W^\dagger$ whose basis vectors pass the verification test, and a vector $\mathbf{v}^\dagger$ that passes the verification test but lives outside the subspace V^* that is the span of network coding combinations of the files he requested and created. A receiver that decodes the basis vectors of $W^\dagger$ together with the vector $\mathbf{v}^\dagger$ will be fooled into accepting a vector from the user with public key pk^* that this user never sent.

Implied properties. The security model implies that even given the secret key sk, no PPT adversary can construct distinct vector spaces $V_1, V_2 \in \mathbb{F}_p^{n+m}$ such that $\mathsf{Sign}(\mathsf{params}, \mathsf{sk}, V_1) = \mathsf{Sign}(\mathsf{params}, \mathsf{sk}, V_2)$. Note, however, that this is no ordinary collision resistance property. During signature verification the vector space V is not available and therefore the Verify algorithm must validate the signature given only $\mathbf{y} \in V$.

This collision resistance property is crucial during decoding. The decoder collects all incoming packets with a specific identifier into a full rank matrix and runs the decoding procedure. Collision resistance ensures that all packets with the same signature belong to the same vector space.

To see that this collision resistance property follows from our definition, it is not difficult to give a generic attack that works on any scheme for which this property is not satisfied. The attack, in fact, is essentially the same as the attack presented in Section 3.1.

The vector space $W^\dagger$. Recall that the forgery attempt by the adversary consists of the 4-tuple $(\overline{\mathsf{pk}}^\dagger, \mathbf{v}^\dagger, \boldsymbol{\sigma}^\dagger, W^\dagger)$ where $\overline{\mathsf{pk}}^\dagger$ is a vector of public keys containing the challenge public key pk^*. The other public keys in the vector $\overline{\mathsf{pk}}^\dagger$ are invented by the adversary and it is therefore possible that the adversary knows the corresponding private keys.

The vector $\mathbf{v}^\dagger$ and the signature $\boldsymbol{\sigma}^\dagger$ are the adversary's existential forgery. Suppose that $(\mathbf{v}^\dagger, \boldsymbol{\sigma}^\dagger)$ verify as a valid vector-signature pair with respect to $\overline{\mathsf{pk}}^\dagger$. We require the adversary to output the vector space $W^\dagger$ to prove that he is capable of exploiting $\mathbf{v}^\dagger$ to fool a recipient to incorrectly accept a vector from the single honest sender. To fool the recipient, the attacker can generate valid vector-signature pairs for all basis vectors of $W^\dagger$ using the secret keys at his disposal. Since all these vectors have valid signatures, a recipient might try to decode the basis of $W^\dagger$ along with the vector $\mathbf{v}^\dagger$. If $\mathbf{v}^\dagger \notin V^*$, after decoding this set of vectors (i.e. after subtracting from $\mathbf{v}^\dagger$ the projection of $\mathbf{v}^\dagger$ onto $W^\dagger$), the recipient obtains a vector $\mathbf{u}$ that he believes came from the honest sender, but which the honest sender never sent since $\mathbf{u}$ is not in $\mathsf{MergeSpaces}(V_1, \ldots, V_k, \sigma_1, \ldots, \sigma_k)$.

Hence, if the attacker is capable of producing a forgery for which condition (2) of adjudication holds, then an adversary can fool a recipient by sending it a sequence of properly signed vectors. We would like to require that for a secure signature scheme it should be impossible to produce a valid forgery where $\mathbf{v}^\dagger \notin V^*$. Unfortunately, this strong requirement appears to be unsatisfiable. We therefore weaken it to require that $\mathbf{v}^\dagger \notin V^*$ only when there is a possibility that the vectors in $W^\dagger$ will be jointly decoded with $\mathbf{v}^\dagger$, namely when $\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}_w, \mathbf{w}_i, \boldsymbol{\sigma}_w) = \top$ for all basis vectors $\mathbf{w}_i$ of $W^\dagger$. This is an acceptable weakening of the security requirement since the decoder will never group together vectors with different identifiers. In Section 5 we show that the resulting definition is satisfiable.

We note that requiring the adversary to output $W^\dagger$ is analogous to the security model of aggregate signatures [5] where the attacker outputs an aggregate signature from s public keys, where $s-1$ of them are invented by the attacker. Moreover, the attacker must output the list of $s-1$ messages that went into the aggregate forgery, for each of the public keys the attacker invented. Our vector space $W^\dagger$ plays the same role as the $s-1$ messages in the aggregate forgery.

5 Construction of a Multi-source Signature Scheme

In this section, we construct an explicit multi-source network coding signature scheme satisfying Definition 4. In order to give a generic construction, we first define an auxiliary primitive called *vector hash.* This primitive captures the properties of the homomorphic hashes used by Krohn et al. [17] that are necessary for secure signatures.

5.1 Vector Hashes

A vector hash consists of three algorithms, $\mathsf{Setup}, \mathsf{Hash}, \mathsf{Test}$, with the following properties:

$\mathsf{HashSetup}(1^\lambda, n)$**:** Input: unary representation of a security parameter λ and dimension of the data space n. Output: public parameters pp.

$\mathsf{Hash}(\mathsf{pp}, \mathbf{v})$**:** Input: public parameters pp and a vector $\mathbf{v} \in \mathbb{F}_p^n$. Output: hash h of the vector $\mathbf{v}$. We require that this algorithm be deterministic.

$\mathsf{Test}(\mathsf{pp}, \mathbf{y}, \overline{\beta}, \mathbf{h})$**:** Input: Public parameters pp, a vector $\mathbf{y} \in \mathbb{F}_p^n$, a vector of coefficients $\overline{\beta} \in \mathbb{F}_p^m$ and a vector of m hash values $\mathbf{h}$. Output: $\top$ (true) or $\perp$ (false).

Let $\mathbf{h}$ be a set of hashes of a basis $\mathbf{v}_1, \ldots, \mathbf{v}_m$ of a vector space V. Intuitively, we want the Test algorithm to tell us whether $\mathbf{y}$ was constructed correctly from the basis, i.e., whether $\mathbf{y} = \sum \beta_i \mathbf{v}_i$. This means that Test should output $\top$ whenever $\mathbf{y}$ is constructed correctly, and it should be difficult for an adversary to find a vector $\mathbf{y} \notin V$ and a $\overline{\beta}$ such that Test outputs $\top$. We now formalize these correctness and security conditions.

Correctness. For correctness, we require the following for all public parameters $\mathsf{pp} \leftarrow \mathsf{HashSetup}(1^\lambda)$:

1. For all $\mathbf{v} \in \mathbb{F}_p^n$, if $h \leftarrow \mathsf{Hash}(\mathsf{pp}, \mathbf{v})$ then we have $\mathsf{Test}(\mathsf{pp}, \mathbf{v}, 1, h) = \top$.
2. Let $\mathbf{v} \in \mathbb{F}_p^n$, let $\overline{\beta} \in \mathbb{F}_p^\ell$ for some ℓ, and let $\mathbf{h}$ be a list of hashes of length ℓ. Fix $i \in \{0, \dots, \ell\}$, let $\overline{\beta}' \in \mathbb{F}_p^{\ell+1}$ be the vector $\overline{\beta}$ with a zero inserted between the ith and $(i+1)$th place, and let $\mathbf{h}'$ be the vector $\mathbf{h}$ with any hash value inserted between the ith and $(i+1)$th place. We require that if $\mathsf{Test}(\mathsf{pp}, \mathbf{v}, \overline{\beta}, \mathbf{h}) = \top$, then $\mathsf{Test}(\mathsf{pp}, \mathbf{v}, \overline{\beta}', \mathbf{h}') = \top$.
3. Let $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{F}_p^n$, let $\overline{\beta}_1, \overline{\beta}_2 \in \mathbb{F}_p^\ell$ for some ℓ, let $\mathbf{h}$ be a list of hashes of length ℓ. Let $a, b \in \mathbb{F}_p$, let $\mathbf{y} = a\mathbf{v}_1 + b\mathbf{v}_2$, and $\overline{\beta} = a\overline{\beta}_1 + b\overline{\beta}_2$. We require that if $\mathsf{Test}(\mathsf{pp}, \mathbf{v}_i, \overline{\beta}_i, \mathbf{h}) = \top$ for $i = 1, 2$ then $\mathsf{Test}(\mathsf{pp}, \mathbf{y}, \overline{\beta}, \mathbf{h}) = \top$.

Security. Let $\mathcal{VH} = (\mathsf{HashSetup}, \mathsf{Hash}, \mathsf{Test})$ be a vector hash. Let $\mathcal{A}$ be a PPT algorithm that takes as input public parameters $\mathsf{pp} \leftarrow \mathsf{HashSetup}(1^\lambda, n)$ and outputs a vector $\mathbf{v}^* \in \mathbb{F}_p^n$, an m-dimensional vector space $V \subset \mathbb{F}_p^n$ (for some m) represented as basis vectors $\mathbf{v}_1, \dots, \mathbf{v}_m$, an m-tuple of coefficients $\overline{\beta}$, and a vector of hashes $\mathbf{h} = (h_1, \dots, h_m)$.

Definition 6. With notation as above, we say that $\mathcal{A}$ *breaks* the vector hash scheme $\mathcal{VH}$ if $\mathbf{v}^* \notin V$, $\mathsf{Test}(\mathsf{pp}, \mathbf{v}^*, \overline{\beta}, \mathbf{h}) = \top$, and $\mathsf{Test}(\mathsf{pp}, \hat{\mathbf{v}}_i, \mathbf{e}_i, h_i) = \top$ for $i = 1, \dots, m$. (Recall that $\hat{\mathbf{v}}_i$ is the data component of $\mathbf{v}_i$.) We define the *advantage* Hash-Adv$[\mathcal{A}, \mathcal{VH}]$ of $\mathcal{A}$ to be the probability that $\mathcal{A}$ breaks $\mathcal{VH}$. We say that a vector hash $\mathcal{VH}$ is *secure* if for all PPT algorithms $\mathcal{A}$ the advantage Hash-Adv$[\mathcal{A}, \mathcal{VH}]$ is negligible in the security parameter λ.

In the full paper [2] we give an example vector hash using a finite cyclic group $\mathbb{G}$ of order p. This vector hash is secure if the discrete logarithm problem is infeasible in $\mathbb{G}$.

5.2 The Construction

For this construction, we use as a black box a vector hash as defined in Section 5.1.

Signature Scheme $\mathcal{NS}$. Let $\mathcal{VH} = (\mathsf{HashSetup}_h, \mathsf{Hash}_h, \mathsf{Test}_h)$ be a vector hash and let $\mathcal{S} = (\mathsf{Setup}_s, \mathsf{KeyGen}_\mathsf{s}, \mathsf{Sign}_\mathsf{s}, \mathsf{Verify}_\mathsf{s})$ be a signature scheme for signing messages in $\{0,1\}^*$. Our network coding signature scheme is as follows:

$\mathsf{Setup}(1^\lambda, n, m)$**:** Run $\mathsf{HashSetup}_h(1^\lambda, n)$ to obtain hash parameters and $\mathsf{Setup}_s(1^\lambda)$ to obtain signature parameters. Let params contain m, n, and the outputs of these algorithms.

$\mathsf{KeyGen}(\mathsf{params})$**:** Run $\mathsf{KeyGen}_\mathsf{s}$ to obtain public key pk and the private key sk. Output $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Sign}(\mathsf{params}, \mathsf{sk}, \{\mathbf{v}_1, \dots, \mathbf{v}_m\})$**:** For $i = 1, \dots, m$, set $h_i := \mathsf{Hash}_h(\mathsf{params}, \hat{\mathbf{v}}_i)$. Set $\mathbf{h} = (h_1, \dots, h_m)$, $\eta := \mathsf{Sign}_s(\mathsf{sk}, \mathbf{h})$, and $\sigma := \big(\mathbf{h}, \eta\big)$. Output σ.

$\mathsf{Combine}\big(\mathsf{params}, (\mathbf{v}_1, \boldsymbol{\sigma}_1, \overline{\mathsf{pk}}_1, a_1),\ (\mathbf{v}_2, \boldsymbol{\sigma}_2, \overline{\mathsf{pk}}_2, a_2)\big)$**:**

1. Let $\boldsymbol{\sigma}', \mathbf{v}_1', \mathbf{v}_2' := \mathsf{Merge}(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2, \mathbf{v}_1, \mathbf{v}_2)$.
2. To create a list $\overline{\mathsf{pk}}'$, do:
   For j in $1, \dots, k = |\boldsymbol{\sigma}|$, if the jth element of $\boldsymbol{\sigma}$ is the kth element of $\boldsymbol{\sigma}_i$ for $i \in \{1, 2\}$, then the jth element of $\overline{\mathsf{pk}}'$ is the kth element of $\overline{\mathsf{pk}}_i$.
3. Output $\boldsymbol{\sigma}'$ and $\overline{\mathsf{pk}}'$.

$\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}, \mathbf{y}, \boldsymbol{\sigma})$**:** Interpret $\boldsymbol{\sigma}$ as a list of f signatures where each $\sigma_i = (\mathbf{h}_i, \eta_i)$. Write $\mathbf{H} = (\mathbf{h}_1, \dots, \mathbf{h}_f)$. Do the following:

1. For i in $1, \dots, f$, compute $\mathsf{Verify}_s(\mathsf{pk}_i, \mathbf{h}_i, \eta_i)$.
2. Compute $\mathsf{Test}_h(\mathsf{params}, \hat{\mathbf{y}}, \overline{\beta}_{\mathbf{y}}, \mathbf{H})$. (Recall $\overline{\beta}_{\mathbf{y}}$ is the augmentation component of $\mathbf{y}$.)

If all steps output $\top$, output $\top$; else output $\perp$.

The only difference between the $\mathsf{Combine}$ algorithm in our signature scheme and the Merge algorithm of Section 2.1 is that the $\mathsf{Combine}$ algorithm also keeps track of the public keys associated with the signatures.

Instead of sending a separate hash signature η_i in each σ_i, we can aggregate these signatures together for space efficiency. In the full paper [2] we describe an instantiation of the system where a signature on f files is of length $(fm+1)\log_2 p$ bits. We also prove a lower bound showing that for large values of f and m this length is optimal.

Correctness. We verify the correctness conditions of Definition 4.

1. For primitive signatures: Consider a key pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{params})$ and a vector space $V \subset \mathbb{F}_p^{n+m}$ described by a properly augmented basis $\mathbf{v}_1, \dots, \mathbf{v}_m$. Let σ be the output of $\mathsf{Sign}(\mathsf{params}, \mathsf{sk}, V = \{\mathbf{v}_1, \dots, \mathbf{v}_m\})$. Interpret the signature σ as $\sigma = (\mathbf{h}, \eta)$.
   For primitive signatures, there is only one file $f = 1$. We examine each step of Verify in turn:
   1. Since $\eta = \mathsf{Sign}_s(\mathsf{sk}, \mathbf{h})$, we have $\mathsf{Verify}_s(\mathsf{pk}, \mathbf{h}, \eta) = \top$ by correctness of $\mathcal{S}$.
   2. Since $h_i = \mathsf{Hash}_h(\mathsf{params}, \hat{\mathbf{v}}_i)$, and $\beta_{\mathbf{v}_i}$ is the unit vector $\mathbf{e}_i$ since we are using a properly augmented basis, correctness conditions (1) and (3) of $\mathcal{VH}$ imply that $\mathsf{Test}_h(\mathsf{params}, \hat{\mathbf{v}}_i, \beta_{\mathbf{v}_i}, \mathbf{h}) = \top$.

   It follows that every basis vector $\mathbf{v}_i$ passes the signature verification test, i.e., $\mathsf{Verify}(\mathsf{params}, \mathsf{pk}, \mathbf{v}_i, \sigma) = \top$.
2. Recursively, for combined signatures: Consider two lists of public keys $\overline{\mathsf{pk}}_1, \overline{\mathsf{pk}}_2$, two augmented vectors $\mathbf{v}_1, \mathbf{v}_2$, two lists of signatures $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2$ such that

$$\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}_1, \mathbf{v}_1, \boldsymbol{\sigma}_1) = \mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}_2, \mathbf{v}_2, \boldsymbol{\sigma}_2) = \top. \tag{5.1}$$

   Let $\mathbf{v}_1', \mathbf{v}_2', \boldsymbol{\sigma}'$ be the output of $\mathsf{Merge}(\mathbf{v}_1, \mathbf{v}_2, \boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$ and $f = |\boldsymbol{\sigma}|$. Let $\mathbf{H}_i$ be the list of all the hash elements in $\boldsymbol{\sigma}_i$ for $i = 1, 2$. Let $a_1, a_2 \in \mathbb{F}_p$ be network combination coefficients, and let $\mathbf{y} = a_1\mathbf{v}'_1 + a_2\mathbf{v}'_2$. Let $\boldsymbol{\sigma}, \overline{\mathsf{pk}}$ be the output of the $\mathsf{Combine}$ algorithm on inputs $(\mathbf{v}_1, \boldsymbol{\sigma}_1, \overline{\mathsf{pk}}_1, a_1), (\mathbf{v}_2, \boldsymbol{\sigma}_2, \overline{\mathsf{pk}}_2, a_2)$.

Conditions (a) and (b) are now immediate. For (c), we note that in our scheme, $\sigma'_j = (\mathbf{h}_j, \eta_j)$ for $j = 1, \ldots, f$. Let $\mathbf{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_f)$. We examine each step of the Verify algorithm:

1. By the assumption (5.1) and the way we have set up the correspondence between indices of $\overline{\mathsf{pk}}$ and $\boldsymbol{\sigma}$, we have $\mathsf{Verify}_s(\mathsf{pk}_j, \mathbf{h}_j, \eta_j) = \top$ for j in $1, \ldots, f$.
2. By assumption (5.1) we know that $\mathsf{Test}_h(\mathsf{params}, \hat{\mathbf{v}}_i, \overline{\beta}_{\mathbf{v}_i}, \mathbf{H}_i) = \top$ for $i = 1, 2$. By correctness property (2) of $\mathcal{VH}$, for $i = 1, 2$ we have $\mathsf{Test}_h(\mathsf{params}, \hat{\mathbf{v}}'_i, \overline{\beta}_{\mathbf{v}'_i}, \mathbf{H}) = \top$. Then, by correctness property (3) of $\mathcal{VH}$, we have $\mathsf{Test}_h(\mathsf{params}, \hat{\mathbf{y}}, \overline{\beta}_{\mathbf{y}}, \mathbf{H}) = \top$.

Thus, we have that $\mathsf{Verify}(\mathsf{params}, \overline{\mathsf{pk}}', \mathbf{y}, \boldsymbol{\sigma}) = \top$.

We have the following security theorem; the proof is in the full paper [2].

Theorem 7. *The network coding signature scheme $\mathcal{NS}$ is secure assuming that $\mathcal{VH}$ is a secure vector hash, and assuming $\mathcal{S}$ is a secure signature scheme.*

References

1. Agrawal, S., Boneh, D.: Homomorphic MACs: MAC-based integrity for network coding. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536. Springer, Heidelberg (2009)
2. Agrawal, S., Boneh, D., Boyen, X., Freeman, D.M.: Preventing pollution attacks in multi-source network coding. Cryptology ePrint Archive (2010), Full version of this paper, available at `http://eprint.iacr.org`
3. Ahlswede, R., Cai, N., Li, S., Yeung, R.: Network information flow. IEEE Transactions on Information Theory 46(4), 1204–1216 (2000)
4. Boneh, D., Freeman, D., Katz, J., Waters, B.: Signing a linear subspace: Signature schemes for network coding. In: Jarecki, S., Tsudik, G. (eds.) Public Key Cryptography – PKC 2009. LNCS, vol. 5443, pp. 68–87. Springer, Heidelberg (2009)
5. 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)
6. Cai, N., Yeung, R.: Secure network coding. In: Proceedings of the 2002 IEEE International Symposium on Information Theory (2002)
7. Charles, D., Jain, K., Lauter, K.: Signatures for network coding. In: 40th Annual Conference on Information Sciences and Systems, CISS 2006 (2006)
8. Cohen, B.: Incentives build robustness in BitTorrent (2003), `http://www.bittorrent.org/bittorrentecon.pdf`
9. Feldman, J., Malkin, T., Stein, C., Servedio, R.: On the capacity of secure network coding. In: Proc. 42nd Annual Allerton Conference on Communication, Control, and Computing (2004)
10. Fragouli, C., Le Boudec, J.-Y., Widmer, J.: Network coding: an instant primer. SIGCOMM Comput. Commun. Rev. 36(1), 63–68 (2006)
11. Fragouli, C., Soljanin, E.: Network Coding Fundamentals. Now Publishers Inc., Hanover (2007)

12. Han, K., Ho, T., Koetter, R., Médard, M., Zhao, F.: On network coding for security. In: Military Communications Conference, Milcom (2007)
13. Ho, T., Leong, B., Koetter, R., Médard, M., Effros, M., Karger, D.: Byzantine modification detection in multicast networks using randomized network coding. In: Proceedings of the 2004 IEEE International Symposium on Information Theory ISIT (June 2004)
14. Jaggi, S., Langberg, M., Katti, S., Ho, T., Katabi, D., Médard, M., Effros, M.: Resilient network coding in the presence of Byzantine adversaries. IEEE Trans. on Information Theory 54(6), 2596–2603 (2008)
15. Katti, S., Rahul, H., Hu, W., Katabi, D., Médard, M., Crowcroft, J.: XORs in the air: practical wireless network coding. IEEE/ACM Trans. Netw. 16(3), 497–510 (2008)
16. Koetter, R., Médard, M.: An algebraic approach to network coding. IEEE/ACM Transactions on Networking, 782–795 (2003)
17. Krohn, M., Freedman, M., Mazieres, D.: On the-fly verification of rateless erasure codes for efficient content distribution. In: Proc. of IEEE Symposium on Security and Privacy, pp. 226–240 (2004)
18. Li, Y., Yao, H., Chen, M., Jaggi, S., Rosen, A.: Ripple authentication for network coding. To appear in IEEE INFOCOM (2010), http://home.ie.cuhk.edu.hk/~mhchen/papers/ripple.infocom10.pdf
19. Zhao, F., Kalker, T., Médard, M., Han, K.: Signatures for content distribution with network coding. In: Proc. Intl. Symp. Info. Theory ISIT (2007)

Groth–Sahai Proofs Revisited

Essam Ghadafi, Nigel P. Smart, and Bogdan Warinschi

Dept. Computer Science,
University of Bristol,
Merchant Venturers Building,
Woodland Road,
Bristol, BS8 1UB,
United Kingdom
{ghadafi,nigel,bogdan}@cs.bris.ac.uk

Abstract. Since their introduction in 2008, the non-interactive zero-knowledge (NIZK) and non-interactive witness indistinguishable (NIWI) proofs designed by Groth and Sahai have been used in numerous applications. In this paper, we offer two contributions to the study of these proof systems. First, we identify and correct some errors, present in the oringal online manuscript, that occur in two of the three instantiations of the Groth-Sahai NIWI proofs for which the equation checked by the verifier is not valid for honest executions of the protocol. In particular, implementations of these proofs would not work correctly. We explain why, perhaps surprisingly, the NIZK proofs that are built from these NIWI proofs do not suffer from a similar problem. Secondly, we study the efficiency of existing instantiations and note that only one of the three instantiations has the potential of being practical. We therefore propose a natural extension of an existing assumption from symmetric pairings to asymmetric ones which in turn enables Groth-Sahai proofs based on new classes of efficient pairings.

1 Introduction

Background. Interactive proofs allow a prover who possesses some witness ω to convince a verifier that a certain statement $x \in L$ is true, where L is some language and ω is a witness that attests to this fact. A particularly fascinating class of interactive proofs are those where the interaction does not reveal information about the witness, even if the verifier behaves maliciously. Two popular flavors of witness privacy are witness-indistinguishability [14], when it is unfeasible for an adversary to decide which of the possible witnesses is used by the prover, and zero-knowledge[19,20], when it is possible to simulate the interaction between the prover and the verifier without access to a witness. The two notions share many commonalities, but are also different in important respects and suitable for different applications. For example, WI proofs can be executed in parallel while preserving the privacy of the witness, while ZK proofs may fail in this scenario.

A variant of zero-knowledge proofs useful in multiple application scenarios are the non-interactive ones [6] (NIZK). In such proofs the interaction between the prover and the verifier is minimal: the prover simply sends the verifier a single message after which the latter verifies correctness of the proof without any further interaction with the prover. It is not difficult to see that NIZK proofs are impossible in the plain model [18], so some additional setup assumptions are required. Originally, such proofs were constructed in a setting where parties share a common random string (CRS) [15]. Later, non-interactive protocols were also constructed by eliminating interaction through the use of random oracles [5].

Unsurprisingly, both zero-knowledge and witness-indistinguishable proofs have found countless applications in cryptography. The power and versatility of such proofs is based on general results that show how to construct zero-knowledge proof systems for any language in NP [21]. For example, with zero-knowledge proofs, a party can prove that he/she is following a certain protocol, without revealing any information about its internal state, and thus can be used to compile protocols secure for honest-but-curious adversaries into protocols secure against arbitrary adversaries. Witness indistinguishable proofs can be used, for instance, in the Yao garbled-circuit protocol, to show that public commitments are commitments to elements in $\{0, 1\}$. The usability of proofs is tightly tied to the class of languages to which they apply, and to the efficiency of the associated proof systems. Clearly, these two requirements are contradictory. Indeed, the approach of [21] is quite general, but the combination of general NP-reductions to problems along with ZK protocols leads to highly impractical protocols even for the simplest languages.

A crucial step towards more efficient non-interactive zero-knowledge proofs was the breakthrough work of Groth and Sahai [25]. The authors show how to give NIWI and NIZK proofs for a large class of languages, without going through the use of a general NP reduction. Numerous cryptographic results use GS proofs to obtain efficient implementations for various primitives, see the related work section for a very partial list of such works. In this paper, we contribute to the understanding of these proofs in two different ways. We extend the range of implementations to new, potentially more efficient settings and we fix an inconspicuous flaw that affects an important part of the original online manuscript [26]. To explain our contributions, we recall some details of the settings used by [25].

In the original (conference) version of the Groth–Sahai paper [25], the authors give a general, abstract framework for the construction of NIWI/NIZK proofs based on cryptographic pairings. We note that none of the errors we identify occur in [25]. Proofs and details for three different instantiations are given in the e-print archive version of the paper [26]. The first instantiation uses pairings over groups of large composite order; the other two use pairings over prime order groups. The cryptographic assumptions on which the results rely are: the subgroup decisional problem [8] in the first case, the decisional linear assumption (DLIN) [7], and the symmetric external Diffie–Hellman assumption (SXDH) [1], for the remaining two instantiations, respectively. To obtain the later instantiations the authors essentially use a general procedure [16] of converting protocols from the subgroup decision setting for composite order pairing groups, into protocols for the DLIN and SXDH assumptions in prime order pairing groups.

Efficient implementations based on a new assumption. From a practical perspective, pairings for groups of composite order are likely to have little practical impact, due to their inherent inefficiency. The same holds true for symmetric pairings, i.e. Type-1 pairings in the vocabulary of [17], which are the pairings used in the second instantiation. Therefore, the only practical instantiation proposed in [26] remains the one based on SXDH in Type-3 curves. In this paper, we propose new GS proofs which can be used with the most efficient curves for pairing based cryptography. Our proposals are based on a natural extension of the DLIN assumption from the symmetric setting to the asymmetric one. We thus give DLIN-based GS proofs that work for all of the asymmetric pairing types. In particular, our proofs are the first GS proofs that work for Type-2 pairings.

We wish to warn readers against judging the efficiency of the proof systems based on Type-1 curves versus those based on Type-2 and Type-3 solely based on the number of group elements needed. The efficiency of the former curves is only illusory since the key sizes for these curves grow faster, and the benefits are immediately lost. Also, we note that the relative merits of the SXDH assumption versus the DLIN assumption are a matter of debate in pairing based cryptography; some people prefer the DLIN assumption as it applies to both symmetric and asymmetric settings, although the latter is never formally stated and we need to formalise the underlying hard problem in this paper. On the other hand the SXDH assumption only applies to Type 3 pairings, which produce the most efficient pairings known. The SXDH assumption also usually results in cleaner and simpler protocol, with Groth–Sahai proofs being no exception. In addition the SXDH assumption is more closely related to a long standing natural number theoretic problem, i.e. decisional Diffie–Hellman, than the DLIN assumption.

Fixing the inconspicuous flaw. The construction of Groth–Sahai NIZK proofs in [25,26] is done in two stages. First, the authors show how to construct NIWI proofs, and then following a trick they turn these proofs into full zero-knowledge ones. Unfortunately, the NIWI proofs based on DLIN and SXDH presented in [26] are actually invalid: the verification equation is not always satisfied when the execution is between honest provers and verifiers. As such, these proofs do not apply for many rather simple but quite useful statements. The details are somewhat technical and we explain this point later in the paper. These errors were introduced during the translation from the construction based on the subgroup decisional problem to the DLIN and SXDH settings [27]. Interestingly, this problem does not affect the construction of NIZK proofs out of NIWI proofs, since in this case the verification equation is always satisfied! Again, we elaborate on this point later in the paper.

We believe that the reason why this error had not been discovered so far is twofold. On the one hand, as explained above, GS NIZK proofs are actually correct. On the other hand, when used in applications, GS (NIWI) proofs are usually treated in a black-box way: the actual proofs are never spelled out, and the associated equations are never verified. Clearly, the problem would immediately show up in an implementation. We fix these problems by giving the correct versions of the proofs.

Finally, we note that in an effort to encourage further study of the Groth-Sahai proofs we depart from the notation in the original paper and use some notation that we believe is more expressive and easier to follow.

Related work. Despite their recent introduction, Groth–Sahai proofs have been widely used. Since Groth–Sahai proofs apply to bilinear groups, they are mainly used to design cryptographic primitives that do not rely on the random oracle assumption. The proofs are used to prove a knowledge of some secret witnesses or as a proof of membership. The scenarios in which the Groth–Sahai proofs are used in the literature include: proving the possession of some signature without actually revealing the signature, proving that two ciphertexts encrypt the same message, etc. For instance, they were used by Camenisch et al. [10] to build an encryption scheme that is KDM-CCA2 secure. Also, the NIWI and NIZK proofs were used by Belenkiy et al.[2,3] to design p-signatures and anonymous credentials. Groth and Lu[24] used the NIZK proofs to prove the correctness of a shuffle. Huang et al. [28] used Groth–Sahai NIWI and NIZK proofs to construct optimistic fair exchange protocol. In [31], Phong et al. used the NIZK proofs to construct undeniable signatures. Belenkiy et al. in[4] have extensively used both the NIWI and NIZK proofs to construct many cryptographic primitives such as p-signatures, verifiable random functions and compact e-cash system. Groth–Sahai proofs have also been used to construct group-signatures [23,30]. In [13,22] the proofs are used to design universally composable oblivious transfer protocols. The first of these is particularly interesting from our perspective; in [13] the authors use a NIWI proof to prove that set of linear equations holds. When this protocol is instantiated with the DLIN or SXDH protocols from [26] one would not obtain a proof which verifies. This is an example of an instance where the verification equations of the GS NIWI proofs are not valid.

Many of the previous applications of Groth–Sahai proofs for prime order groups, are assumed to be in the (inefficient) symmetric pairing setting, as they wish to use protocols based on the DLIN assumption; or they are in the asymmetric setting and need to make a DLIN assumption related to their scheme and then an additional SXDH assumption to apply Groth–Sahai proofs. By extending the DLIN setting to both Type-2 and Type-3 pairings we hope to simplify future applications of Groth–Sahai proofs, in addition by providing a mechanism for implementing Groth–Sahai proofs in the Type-2 setting other applications may open up.

2 Bilinear Groups

Bilinear groups are a set of three groups $\mathbb{G}_1$,$\mathbb{G}_2$ and G_T of prime order q along with a bilinear map (deterministic function) $\hat{t}$ which takes as input an element in $\mathbb{G}_1$ and an element in $\mathbb{G}_2$ and outputs an element in $\mathbb{G}_T$. We shall write $\mathbb{G}_1$

and $\mathbb{G}_2$ additively, and $\mathbb{G}_T$ multiplicatively, and write $\mathbb{G}_1 = \langle P_1 \rangle, \mathbb{G}_2 = \langle P_2 \rangle$, for two explicitly given generators P_1 and P_2.

The function $\hat{t}$ must have the following three properties:

1. Bilinearity: $\forall Q_1 \in \mathbb{G}_1$, $Q_2 \in \mathbb{G}_2$ $x, y \in \mathbb{Z}$, we have
$$\hat{t}([x]Q_1, [y]Q_2) = \hat{t}(Q_1, Q_2)^{xy}.$$
2. Non-Degeneracy: The value $\hat{t}(P_1, P_2)$ generates $\mathbb{G}_T$.
3. The function $\hat{t}$ is efficiently computable.

In [17], pairings were categorized into three Types:

– **Type-1**: This is the symmetric pairing setting in which $\mathbb{G}_1 = \mathbb{G}_2$.
– **Type-2**: Here we have $\mathbb{G}_1 \neq \mathbb{G}_2$, but there is an efficiently computable isomorphism $\psi : \mathbb{G}_2 \longrightarrow \mathbb{G}_1$ where $\psi(P_2) = P_1$.
– **Type-3**: Again $\mathbb{G}_1 \neq \mathbb{G}_2$, but now there is no known efficiently computable isomorphism.

In the Type-1 setting the decision Diffie–Hellman problem is easy in $\mathbb{G}_1$, and hence in $\mathbb{G}_2$. In the Type-2 setting the decision Diffie–Hellman problem is easy in $\mathbb{G}_2$, but believed to be hard in $\mathbb{G}_1$. In the Type-3 setting the decision Diffie–Hellman problem is believed to be hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$. This last belief is often formalised as the symmetric external Diffie–Hellman assumption:

Definition 1. Symmetric External Diffie-Hellman (SXDH) Assumption: *In Type-3 pairings the Decisional Diffie-Hellman (DDH) problem is hard in both groups $\mathbb{G}_1$ and $\mathbb{G}_2$.*

As a note on naming, the “external” part relates to the fact we are talking about DDH in $\mathbb{G}_1$ and $\mathbb{G}_2$, as opposed to the pairing based BDDH problem. The “symmetric” part is related to the fact that we are talking about DDH being hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$. It is perhaps unfortunate terminology that this symmetry only applies in the asymmetric pairing setting!

As the SXDH problem only applies to Type-3 pairings, it is common to make the following assumption for Type-1 pairings, as a natural strengthening of the normal DDH assumption, which no longer applies in Type-1 pairings:

Definition 2. Decisional Linear Problem (DLIN) Assumption: *For Type-1 pairings with $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ and $P = P_1 = P_2$, given the tuple $([a]P, [b]P, [ra]P, [sb]P, [t]P)$ where $a, b, r, s, t \in \mathbb{F}_q$ are unknowns, it is hard to tell whether $t = r + s$ or t is random.*

To extend this definition to the Type-2 or Type-3 setting one could insist that DLIN is hard in either $\mathbb{G}_1$ or $\mathbb{G}_2$, however we will require that it is hard in *both* $\mathbb{G}_1$ and $\mathbb{G}_2$. We call this latter notion, in following the naming of the SXDH assumption, as the symmetric DLIN (SDLIN) assumption.

Definition 3. Symmetric Decisional Linear Problem (SDLIN) Assumption: *SDLIN is said to hold if DLIN is hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$.*

This is a stronger form of a version of the asymmetric DLIN problem considered in other works such as [22], where a single problem with some variable instances in $\mathbb{G}_1$ and some in $\mathbb{G}_2$ is considered.

We end this section by noting that in [9], Boneh et al. showed that the existence of the isomorphism in the Type-2 setting can affect the security of some cryptographic primitives. On the other hand, Chatterjee and Menezes [11] show that a protocol which is secure in Type-2 setting can almost always be transfered to one which is secure in Type-3 setting.

3 Groth–Sahai Proofs

In [25,26] Groth and Sahai presented a way to construct efficient non-interactive witness-indistinguishable and zero-knowledge proofs for a wide variety of statements in the common reference string model. In this section, we recap on their notation, and point out the problems with their presentation.

The NIZK proof systems allow the same methodology to be applied to four distinct types of equations, or three distinct types in the case of Type-1 pairings. In this section the four different types are presented in one go using the abstraction of Groth-Sahai. Later we present the specialisations to the different settings.

Let q be the order of $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ as above. We first create $\mathbb{F}_q$-vector spaces $\mathbb{A}_1, \mathbb{A}_2, \mathbb{A}_T, \mathbb{B}_1, \mathbb{B}_2$ and $\mathbb{B}_T$. In [26] these are $\mathbb{Z}_n$-modules and not $\mathbb{F}_q$-vector spaces since n may be composite, in our situations we always have $n = q$, a prime. We assume these vector spaces are equiped with bilinear maps $f : \mathbb{A}_1 \times \mathbb{A}_2 \to \mathbb{A}_T$ and $F : \mathbb{B}_1 \times \mathbb{B}_2 \to \mathbb{B}_T$. In addition, there are inclusion and projection maps for each pair, i.e. we have maps $\iota_1 : \mathbb{A}_1 \to \mathbb{B}_1$, $\iota_2 : \mathbb{A}_2 \to \mathbb{B}_2$, $\iota_T : \mathbb{A}_T \to \mathbb{B}_T$, and $p_1 : \mathbb{B}_1 \to \mathbb{A}_1$, $p_2 : \mathbb{B}_2 \to \mathbb{A}_2$, $p_T : \mathbb{B}_T \to \mathbb{A}_T$. Note, that the ι maps are required to be computable, but that the p maps will not be computable in general. The maps are extended to vectors of elements in a componentwise fashion.

All these maps need to satisfy the following commutative properties:

$$\begin{aligned}\forall x \in \mathbb{A}_1, \forall y \in \mathbb{A}_2 :& F(\iota_1(x), \iota_2(y)) = \iota_T(f(x, y)),\\ \forall \mathcal{X} \in \mathbb{B}_1, \forall \mathcal{Y} \in \mathbb{B}_2 :& f(p_1(\mathcal{X}), p_2(\mathcal{Y})) = p_T(F(\mathcal{X}, \mathcal{Y})).\end{aligned}$$

The essential problem in the DLIN and SXDH settings from [26] is that the specific values of these maps, for three of the four equation types, do not result in the first of these commutative properties holding. In particular the given presentation of ι_T is incorrect. This leads to the resulting verification of the NIWI proofs being invalid.

The CRS we use in our proofs is a set of $\hat{m}_1$ and $\hat{m}_2$ elements of $\mathbb{B}_1$ and $\mathbb{B}_2$, which we will denote by $\mathcal{U}_1^{(1)}, \ldots, \mathcal{U}_1^{(\hat{m}_1)} \in \mathbb{B}_1$ and $\mathcal{U}_2^{(1)}, \ldots, \mathcal{U}_2^{(\hat{m}_2)} \in \mathbb{B}_2$. To commit to an element $x \in \mathbb{A}_i$ one picks $\underline{r} = (r_1, \ldots, r_{\hat{m}_i}) \in \mathbb{F}_q^{\hat{m}_i}$ and computes

$$\begin{aligned}\mathrm{comm_i}(x) &= \iota_i(x) + \sum_{j=1}^{\hat{m}_i} [r_j] \mathcal{U}_i^{(j)}\\ &= \iota_i(x) + \underline{r} \cdot \underline{\mathcal{U}_i}.\end{aligned}$$

Now suppose we wish to produce a NIWI proof for the equation,

$$\underline{a} \otimes \underline{y} + \underline{x} \otimes \underline{b} + \underline{x} \otimes \Gamma \underline{y} = t, \tag{1}$$

where we use the shorthand $x \otimes y$ for $f(x, y)$, with an obvious extension to vectors. In the above equation; $\underline{x} \in \mathbb{A}_1^n$, $\underline{y} \in \mathbb{A}_2^m$ are the secret witnesses, with $\underline{a} \in \mathbb{A}_1^m$, $\underline{b} \in \mathbb{A}_2^n$, $\Gamma \in \mathrm{Mat}_{n\times m}(\mathbb{F}_q)$, and $t \in \mathbb{A}_T$ the known constants.

We commit to $\underline{x}$ and $\underline{y}$ using the random values given by $R \in \mathrm{Mat}_{n\times \hat{m}_1}(\mathbb{F}_q)$ and $S \in \mathrm{Mat}_{m\times \hat{m}_2}(\mathbb{F}_q)$ via

$$\underline{c} = \iota_1(\underline{x}) + R\,\underline{\mathcal{U}_1} \text{ and } \underline{d} = \iota_2(\underline{y}) + S\,\underline{\mathcal{U}_2}.$$

The NIWI proof is then given by the following two vector values; one picks $T \in \mathrm{Mat}_{\hat{m}_2\times \hat{m}_1}(\mathbb{F}_q)$ uniformly at random and computes

$$\underline{\pi} = R^{\mathrm{T}}\iota_2(\underline{b}) + R^{\mathrm{T}}\Gamma\iota_2(\underline{y}) + R^{\mathrm{T}}\Gamma S\,\underline{\mathcal{U}_2} - T^{\mathrm{T}}\underline{\mathcal{U}_2} \in \mathbb{B}_2^{\hat{m}_1},$$
$$\underline{\theta} = S^{\mathrm{T}}\iota_1(\underline{a}) + S^{\mathrm{T}}\Gamma^{\mathrm{T}}\iota_1(\underline{x}) + T\,\underline{\mathcal{U}_1} \in \mathbb{B}_1^{\hat{m}_2}.$$

Verification of the proof $(\underline{\pi}, \underline{\theta})$ is performed by checking whether

$$\iota_1(\underline{a}) \bullet \underline{d} + \underline{c} \bullet \iota_2(\underline{b}) + \underline{c} \bullet \Gamma \underline{d} = \iota_T(t) + \underline{\mathcal{U}_1} \bullet \underline{\pi} + \underline{\theta} \bullet \underline{\mathcal{U}_2}$$

holds. Here we use $\mathcal{X} \bullet \mathcal{Y}$ as a shorthand for $F(\mathcal{X}, \mathcal{Y})$, again with an obvious extension for vectors.

Notes. There are four possible instantiations of the equations:

- $\mathbb{A}_1 = \mathbb{G}_1$, $\mathbb{A}_2 = \mathbb{G}_2$, $f(P, Q) = \hat{t}(P, Q)$: This case is called *pairing product equations.*
- $\mathbb{A}_1 = \mathbb{G}_1$, $\mathbb{A}_2 = \mathbb{F}_q$, $f(P, y) = [y]P$: This case is called *multi-scalar multiplication in* $\mathbb{G}_1$.
- $\mathbb{A}_1 = \mathbb{F}_q$, $\mathbb{A}_2 = \mathbb{G}_2$, $f(x, Q) = [x]Q$: This case is called *multi-scalar multiplication in* $\mathbb{G}_2$.
- $\mathbb{A}_1 = \mathbb{F}_q$, $\mathbb{A}_2 = \mathbb{F}_q$, $f(x, y) = x \cdot y$: This case is called *quadratic equation in* $\mathbb{F}_q$.

In the DLIN and SXDH cases, the formulaes for ι_T for the last three types of equations are given incorrectly in [26]. ¿From examining the above methods for NIWI proofs, we see that the NIWI proofs would not verify, unless the value t was the trivial element.

We note that in the simpler, yet very common, setting of having $\Gamma = 0$ and either $\underline{a} = \underline{0}$ or $\underline{b} = \underline{0}$ in equation (1), the proofs can be simplified further by setting the random matrix T to be zero.

The CRS, and hence the commitment scheme used to commit to elements in $\mathbb{A}_1$ and $\mathbb{A}_2$, comes in two flavours: either we have a *binding key*, or a *hiding key.*

- **Binding key:** This setting requires that for $i = 1$ and $i = 2$, $p_i(\iota_i(x)) = x$ and $p_i(\mathcal{U}_i^{(j)}) = 0$ for all j. Hence we have $p_i(\mathrm{comm_i}(x)) = x$ which gives us a perfectly binding, computationally hiding commitment scheme. When used in the proof, this results in perfectly-sound proofs with computational witness indistinguishablity.

- **Hiding key:** This setting requires that $\{\mathcal{U}_i^{(1)}, \dots, \mathcal{U}_i^{(\hat{m}_i)}\}$, i.e. the set of commitment keys, generate the entire space $\mathbb{B}_i$, and hence we have $\iota_i(\mathbb{A}_i) \subseteq \langle \mathcal{U}_i^{(1)}, \dots, \mathcal{U}_i^{(\hat{m}_i)} \rangle$. Therefore, if the randomness vector, $\underline{r}$, is uniformly chosen, the commitment scheme is computationally binding and perfectly hiding. If this setting is used, the resulting proofs are computationally sound and perfectly witness-indistinguishable.

The security of the whole system is ensured as long as the adversary is unable to distinguish between a hiding and a binding key. The security proofs can be found in [26]. When producing a real system, one relies on a trusted third party to produce a binding key, however when producing a simulated proof etc. one relies on a hiding key, which essentially provides a trapdoor for the simulator in the CRS model.

For the DLIN assumption in the Type-1 setting in [26], a method is given to make the map F symmetric, in the sense that $F(\mathcal{X}, \mathcal{Y}) = F(\mathcal{Y}, \mathcal{X})$. We shall see when F is instantiated below, that such a symmetry is not possible for Type-2 and Type-3 pairings. When F is symmetric the associated proofs can be made much simpler, we leave the reader to consult [26] for details.

To convert the above method for NIWI proofs into a method for NIZK proofs, we first reorganize the above equation as

$$\underline{a} \otimes \underline{y} + (-1 \otimes t) + \underline{x} \otimes \underline{b} + \underline{x} \otimes \Gamma \underline{y} = \begin{cases} 0 & \text{If } \mathbb{A}_T = \mathbb{F}_q, \\ \mathcal{O} & \text{If } \mathbb{A}_T = \mathbb{G}_1 \text{ or } \mathbb{G}_2, \\ 1 & \text{If } \mathbb{A}_T = \mathbb{G}_T. \end{cases}$$

The vector of commitments $\underline{c}$ is extended to include a commitment to the element one, this is done to deal with the extra term in the left hand side of the above equation. Then the above NIWI method is applied. This results in the NIZK proofs in the pairing product equation subcase only applying when either $t = 1$ in equation (1), or one knows $P_1, \dots, P_n$ and $Q_1, \dots, Q_n$ such that $t = \hat{t}(P_1, Q_1) \cdots \hat{t}(P_n, Q_n)$, since only then can the above transform be applied. This is the only restriction in the method for obtaining NIZK proofs.

In all cases, to obtain NIZK proofs we apply the method for NIWI proofs in the case where the equation is homogeneous, i.e. has a trivial right hand side. This latter point is crucial in understanding why the NIZK proofs from [26] work but the NIWI proofs do not. Hence, even though ι_T was presented incorrectly in [26], since the method to produce NIZK proofs will result in a trivial value of ι_T, the method for NIZK is sound.

4 Equations for ι and p

From the last section, it is seen that the whole system depends on the choice of the ι and p maps, plus the CRS. The maps must be chosen so that they have the required commutativity property over f and F. In this section, we give such maps and the relevant CRS for the SXDH and SDLIN examples in the asymmetric pairing setting.

We present the data in the following way, for each setting we first present the hiding and binding CRS, along with the map F and the groups $\mathbb{B}_i$ and $\mathbb{B}_T$. Then we present the maps ι_i and p_i for the cases $\mathbb{A}_i = \mathbb{F}_q$ and $\mathbb{A}_i = \mathbb{G}_i$. At this point we overload the symbols ι_i and p_i, with the precise maps being obtained by type-checking. This helps simplify our notation somewhat.

Once the maps are defined we can proceed to produce the commitment schemes, and the NIWI and NIZK proofs. Then for the four types of equation being proved, we present the maps ι_T and p_T, which result in the maps being commutative. With these maps one can then verify the resulting NIWI proofs. Again we overload ι_T and p_T, with the precise map being determined by type checking.

4.1 SXDH-Based Proofs

Setup. We set $\mathbb{B}_1 = \mathbb{G}_1^2$, $\mathbb{B}_2 = \mathbb{G}_2^2$ and $\mathbb{B}_T = \mathbb{G}_T^4$, all with operations performed componentwise. We let

$$F : \begin{cases} \mathbb{B}_1 \times \mathbb{B}_2 \longrightarrow \mathbb{B}_T \\ (X_1, Y_1), (X_2, Y_2) \longmapsto \left(\hat{t}(X_1, X_2),\ \hat{t}(X_1, Y_2),\ \hat{t}(Y_1, X_2),\ \hat{t}(Y_1, Y_2) \right) \end{cases}$$

Since the underlying pairing $\hat{t}$ is bilinear, it follows that the map F is also bilinear. To generate the CRS, the trusted party generates, for $i = 1, 2$, $a_i, t_i \in \mathbb{F}_q^*$ at random and defines

$$Q_i = [a_i]P_i, \quad U_i = [t_i]P_i, \quad V_i = [t_i]Q_i.$$

We now set

$$\mathcal{U}_i^{(1)} = (P_i, Q_i) \in \mathbb{B}_i,$$
$$\mathcal{U}_i^{(2)} = \begin{cases} [t_i]\mathcal{U}_i^{(1)} = (U_i, V_i) & \text{Binding Case} \\ [t_i]\mathcal{U}_i^{(1)} - (\mathcal{O}, P_i) = (U_i, V_i - P_i) & \text{Hiding Case} \end{cases} \in \mathbb{B}_i.$$

The CRS is then the set $\{\mathcal{U}_1, \mathcal{U}_2\}$ where $\mathcal{U}_1 = \{\mathcal{U}_1^{(1)}, \mathcal{U}_1^{(2)}\}$, and $\mathcal{U}_2 = \{\mathcal{U}_2^{(1)}, \mathcal{U}_2^{(2)}\}$. Under the SXDH assumption one cannot tell a binding key from a hiding key. To aid what follows, we first set $\mathcal{W}_i = \mathcal{U}_i^{(2)} + (\mathcal{O}, P_i) = (W_{i,1}, W_{i,2}) \in \mathbb{B}_i$.

***ι_i*, *p_i* and comm$_\mathbf{i}$.** We now define the maps $\iota_i : \mathbb{A}_i \to \mathbb{B}_i$, $p_i : \mathbb{B}_i \to \mathbb{A}_i$ and the commitment scheme comm$_\mathrm{i}$. There are two cases we need to consider; $\mathbb{A}_i = \mathbb{F}_q$ and $\mathbb{A}_i = \mathbb{G}_i$.

$\underline{A_i = \mathbb{F}_q}$: We define, in this case, the maps via

$$\iota_i : \begin{cases} \mathbb{F}_q \longrightarrow \mathbb{B}_i \\ x \longmapsto [x]\mathcal{W}_i \end{cases} \qquad p_i : \begin{cases} \mathbb{B}_i \longrightarrow \mathbb{F}_q \\ \mathcal{X} = ([c_1]P_i, [c_2]P_i) \longmapsto c_2 - a_i c_1 \end{cases}$$

Note, that computing p_i requires one to solve discrete logarithms. This is not an issue since we at no point will compute p_i, we simply need to know it exists and it has the correct properties.

The commitment scheme $\mathrm{comm_i}$ is obtained as before, except we select $\hat{m}_i = 1$, as opposed to $\hat{m}_i = 2$, this simplifies the equations somewhat. Hence we have

$$\mathrm{comm_i} : \begin{cases} \mathbb{F}_q \times \mathbb{F}_q \longrightarrow \mathbb{B}_i \\ (x, r) \longmapsto \iota_i(x) + [r]\mathcal{U}_i^{(1)} \end{cases}$$

$\underline{A_i = \mathbb{G}_i}$: In this case we define

$$\iota_i : \begin{cases} \mathbb{G}_i \longrightarrow \mathbb{B}_i \\ X \longmapsto (\mathcal{O}, X) \end{cases} \qquad p_i : \begin{cases} \mathbb{B}_i \longrightarrow \mathbb{G}_i \\ \mathcal{X} = (C_1, C_2) \longmapsto C_2 - [a_i]C_1 \end{cases}$$

The commitment scheme $\mathrm{comm_i}$ is obtained as in our main discussion, i.e. with $\hat{m}_i = 2$. Hence we have

$$\mathrm{comm_i} : \begin{cases} \mathbb{G}_i \times \mathbb{F}_q \times \mathbb{F}_q \longrightarrow \mathbb{B}_i \\ (X, r_1, r_2) \longmapsto \iota_i(X) + [r_1]\mathcal{U}_i^{(1)} + [r_2]\mathcal{U}_i^{(2)} \end{cases}$$

***ι_T* and *p_T*.** Here we have four cases, depending on which of the four types of equation we are dealing with

Pairing Product Equations.

$$\iota_T : \begin{cases} \mathbb{G}_T \longrightarrow \mathbb{B}_T \\ \zeta \longmapsto (1, 1, 1, \zeta) \end{cases} \quad p_T : \begin{cases} \mathbb{B}_T \longrightarrow \mathbb{G}_T \\ (\zeta_{1,1}, \zeta_{1,2}, \zeta_{2,1}, \zeta_{2,2}) \longmapsto \zeta_{2,2}\zeta_{1,2}^{-a_1}(\zeta_{2,1}\zeta_{1,1}^{-a_1})^{-a_2} \end{cases}$$

Multi-Scalar Multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$.

In both of these cases we have

$$p_T : \begin{cases} \mathbb{B}_T \longrightarrow \mathbb{G}_i \\ (\zeta^{s_1}, \zeta^{s_2}, \zeta^{s_3}, \zeta^{s_4}) \longmapsto [s_4 - a_1 s_2 - a_2 s_3 + a_1 a_2 s_1]P_i \end{cases}$$

where $\zeta = \hat{t}(P_1, P_2)$. For multi-scalar multiplication in $\mathbb{G}_1$ the map ι_T is defined by

$$\iota_T : \begin{cases} \mathbb{G}_1 \longrightarrow \mathbb{B}_T \\ X \longmapsto (1, 1, \hat{t}(X, W_{2,1}), \hat{t}(X, W_{2,2})) \end{cases}$$

Whilst for multi-scalar multiplication in $\mathbb{G}_2$ the map ι_T is defined by

$$\iota_T : \begin{cases} \mathbb{G}_2 \longrightarrow \mathbb{B}_T \\ X \longmapsto (1, \hat{t}(W_{1,1}, X), 1, \hat{t}(W_{1,2}, X)). \end{cases}$$

Note, these are different definitions from those given in [26]. The above definitions produce the required commutative properties.

Quadratic Equations in $\mathbb{F}_q$.

In this case we have

$$p_T : \begin{cases} \mathbb{B}_T \longrightarrow \mathbb{F}_q \\ (\zeta^{s_1}, \zeta^{s_2}, \zeta^{s_3}, \zeta^{s_4}) \longmapsto s_4 - a_1 s_2 - a_2 s_3 + a_1 a_2 s_1 \end{cases}$$

where $\zeta = \hat{t}(P_1, P_2)$. The function ι_T is given by

$$\iota_T(z) : \begin{cases} \mathbb{F}_q \longrightarrow \mathbb{B}_T \\ z \longmapsto F(\mathcal{W}_1, \mathcal{W}_2)^z. \end{cases}$$

Again this is different from the map given in [26].

4.2 SDLIN-Based Proofs

We now perform a similar analysis when we wish to base security on the SDLIN problem. Recall in [26] this situation is only described for the Type-1 pairing situation. What we describe below can be used in both the Type-2 and Type-3 situations. In addition by specialising it to the Type-1 situation, and applying the optimization of [26], to produce a symmetric version of $F(\mathcal{X}, \mathcal{Y})$, one obtains more efficient NIZK proofs for Type-1 pairings as well.

Setup. We set $\mathbb{B}_1 = \mathbb{G}_1^3$, $\mathbb{B}_2 = \mathbb{G}_2^3$ and $\mathbb{B}_T = \mathbb{G}_T^9$, all with operations performed componentwise. We let

$$F : \begin{cases} \mathbb{B}_1 \times \mathbb{B}_2 & \longrightarrow & \mathbb{B}_T \\ (X_1, Y_1, Z_1), (X_2, Y_2, Z_2) & \longmapsto & \begin{pmatrix} \hat{t}(X_1, X_2) & \hat{t}(X_1, Y_2) & \hat{t}(X_1, Z_2) \\ \hat{t}(Y_1, X_2) & \hat{t}(Y_1, Y_2) & \hat{t}(Y_1, Z_2) \\ \hat{t}(Z_1, X_2) & \hat{t}(Z_1, Y_2) & \hat{t}(Z_1, Z_2) \end{pmatrix} \end{cases}$$

Since the underlying pairing $\hat{t}$ is bilinear, it follows that the map F is also bilinear. To generate the CRS the trusted party generates, for $i = 1, 2$ $a_i, r_i, s_i, t_i \in \mathbb{F}_q^*$ at random and defines

$$U_i = [a_i]P_i, \quad V_i = [t_i]P_i.$$

We now set

$$\begin{aligned} \mathcal{U}_i^{(1)} &= (U_i, \mathcal{O}, P_i) \in \mathbb{B}_i, \\ \mathcal{U}_i^{(2)} &= (\mathcal{O}, V_i, P_i) \in \mathbb{B}_i, \\ \mathcal{U}_i^{(3)} &= \begin{cases} [r_i]\mathcal{U}_i^{(1)} + [s_i]\mathcal{U}_i^{(2)} & \\ \quad = ([r_i]U_i, [s_i]V_i, [r_i + s_i]P_i) & \text{Binding Case} \\ [r_i]\mathcal{U}_i^{(1)} + [s_i]\mathcal{U}_i^{(2)} - (\mathcal{O}, \mathcal{O}, P_i) & \\ \quad = ([r_i]U_i, [s_i]V_i, [r_i + s_i - 1]P_i) & \text{Hiding Case} \end{cases} \end{aligned}$$

The CRS is then the set $\{\mathcal{U}_1, \mathcal{U}_2\}$ where $\mathcal{U}_1 = \{\mathcal{U}_1^{(1)}, \mathcal{U}_1^{(2)}, \mathcal{U}_1^{(3)}\}$, and $\mathcal{U}_2 = \{\mathcal{U}_2^{(1)}, \mathcal{U}_2^{(2)}, \mathcal{U}_2^{(3)}\}$. Under the SDLIN assumption one cannot tell a binding key from a hiding key. To aid notation in what follows, we first set $\mathcal{W}_i = \mathcal{U}_i^{(3)} + (\mathcal{O}, \mathcal{O}, P_i) = (W_{i,1}, W_{i,2}, W_{i,3}) \in \mathbb{B}_i$.

***ι_i*, *p_i* and comm$_\mathrm{i}$.** We now define the maps $\iota_i : \mathbb{A}_i \to \mathbb{B}_i$, $p_i : \mathbb{B}_i \to \mathbb{A}_i$ and the commitment scheme comm$_\mathrm{i}$. There are two cases; $\mathbb{A}_i = \mathbb{F}_q$ and $\mathbb{A}_i = \mathbb{G}_i$.

$\underline{A_i = \mathbb{F}_q}$: We define the maps via

$$\iota_i : \begin{cases} \mathbb{F}_q \longrightarrow \mathbb{B}_i \\ x \longmapsto [x]\mathcal{W}_i \end{cases} \qquad p_i : \begin{cases} \mathbb{B}_i & \longrightarrow & \mathbb{F}_q \\ \mathcal{X} = ([c_1]P_i, [c_2]P_i, [c_3]P_i) & \longmapsto & c_3 - \frac{1}{a_i}c_1 - \frac{1}{t_i}c_2 \end{cases}$$

The commitment scheme comm$_\mathrm{i}$ is obtained as before, except we select $\hat{m}_i = 2$, as opposed to $\hat{m}_i = 3$, this again simplifies the equations. Hence we have

$$\mathrm{comm_i} : \begin{cases} \mathbb{F}_q \times \mathbb{F}_q \times \mathbb{F}_q & \longrightarrow & \mathbb{B}_i \\ (x, r_1, r_2) & \longmapsto & \iota_i(x) + [r_1]\mathcal{U}_i^{(1)} + [r_2]\mathcal{U}_i^{(2)} \end{cases}$$

$\underline{A_i = \mathbb{G}_i}$: We define

$$\iota_i : \begin{cases} \mathbb{G}_i \longrightarrow \mathbb{B}_i \\ X \longmapsto (\mathcal{O}, \mathcal{O}, X) \end{cases} \qquad p_i : \begin{cases} \mathbb{B}_i \longrightarrow \mathbb{G}_i \\ \mathcal{X} = (C_1, C_2, C_3) \longmapsto C_3 - [\frac{1}{a_i}]C_1 - [\frac{1}{t_i}]C_2 \end{cases}$$

The commitment scheme $\mathrm{comm_i}$ is obtained as in our main discussion, i.e. with $\hat{m}_i = 3$. Hence we have

$$\mathrm{comm_i} : \begin{cases} \mathbb{G}_i \times \mathbb{F}_q \times \mathbb{F}_q \times \mathbb{F}_q \longrightarrow \mathbb{B}_i \\ (X, r_1, r_2, r_3) \longmapsto \iota_i(X) + [r_1]\mathcal{U}_i^{(1)} + [r_2]\mathcal{U}_i^{(2)} + [r_3]\mathcal{U}_i^{(3)} \end{cases}$$

$\boldsymbol{\iota_T}$ **and** $\boldsymbol{p_T}$**.** Here we have four cases, depending on which of the four types of equation we are dealing with

Pairing Product Equations.

$$\iota_T : \begin{cases} \mathbb{G}_T \longrightarrow \mathbb{B}_T \\ \zeta \longmapsto \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & \zeta \end{pmatrix} \end{cases} \qquad p_T : \begin{cases} \mathbb{B}_T \longrightarrow \mathbb{G}_T \\ \begin{pmatrix} \zeta_{1,1} & \zeta_{1,2} & \zeta_{1,3} \\ \zeta_{2,1} & \zeta_{2,2} & \zeta_{2,3} \\ \zeta_{3,1} & \zeta_{3,2} & \zeta_{3,3} \end{pmatrix} \longmapsto \gamma_1^{-1/a_2}\gamma_2^{-1/t_2}\gamma_3 \end{cases}$$

where $\gamma_i = \zeta_{1,i}^{-1/a_1}\zeta_{2,i}^{-1/t_1}\zeta_{3,i}$.

Multi-Scalar Multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$.

In both of these cases we have

$$p_T : \begin{cases} \mathbb{B}_T \longrightarrow \mathbb{G}_i \\ \begin{pmatrix} \zeta^{s_{1,1}} & \zeta^{s_{1,2}} & \zeta^{s_{1,3}} \\ \zeta^{s_{2,1}} & \zeta^{s_{2,2}} & \zeta^{s_{2,3}} \\ \zeta^{s_{3,1}} & \zeta^{s_{3,2}} & \zeta^{s_{3,3}} \end{pmatrix} \longmapsto [s_3 - \frac{1}{a_2}s_1 - \frac{1}{t_2}s_2]P_i \end{cases}$$

where $\zeta = \hat{t}(P_1, P_2)$ and

$$s_i = s_{3,i} - \frac{1}{a_1}s_{1,i} - \frac{1}{t_1}s_{2,i}.$$

For multi-scalar multiplication in $\mathbb{G}_1$ the map ι_T is defined by

$$\iota_T : \begin{cases} \mathbb{G}_1 \longrightarrow \mathbb{B}_T \\ X \longmapsto \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ \hat{t}(X, W_{2,1}) & \hat{t}(X, W_{2,2}) & \hat{t}(X, W_{2,3}) \end{pmatrix} \end{cases}$$

Whilst for multi-scalar multiplication in $\mathbb{G}_2$ the map ι_T is defined by

$$\iota_T : \begin{cases} \mathbb{G}_2 \longrightarrow \mathbb{B}_T \\ X \longmapsto \begin{pmatrix} 1 & 1 & \hat{t}(W_{1,1}, X) \\ 1 & 1 & \hat{t}(W_{1,2}, X) \\ 1 & 1 & \hat{t}(W_{1,3}, X) \end{pmatrix} \end{cases}$$

When specialised to the symmetric case these are different definitions of ι_T to those given in [26]. The above definitions produce the required commutative properties.

Quadratic Equations in $\mathbb{F}_q$.
In this case we have

$$p_T : \begin{cases} \mathbb{B}_T & \longrightarrow \quad \mathbb{F}_q \\ \begin{pmatrix} \zeta^{s_{1,1}} & \zeta^{s_{1,2}} & \zeta^{s_{1,3}} \\ \zeta^{s_{2,1}} & \zeta^{s_{2,2}} & \zeta^{s_{2,3}} \\ \zeta^{s_{3,1}} & \zeta^{s_{3,2}} & \zeta^{s_{3,3}} \end{pmatrix} & \longmapsto s_3 - \frac{1}{a_2} s_1 - \frac{1}{t_2} s_2 \end{cases}$$

where again we have $\zeta = \hat{t}(P_1, P_2)$ and

$$s_i = s_{3,i} - \frac{1}{a_1} s_{1,i} - \frac{1}{t_1} s_{2,i}.$$

The function ι_T is given by

$$\iota_T(z) : \begin{cases} \mathbb{F}_q \longrightarrow \mathbb{B}_T \\ \ z \ \longmapsto F(\mathcal{W}_1, \mathcal{W}_2)^z. \end{cases}$$

Again this is different from the mapping given in [26].

4.3 Combining SXDH and SDLIN

We end this section by noting an extension which was pointed out to us by J. Groth [27]. If one wanted to work in Type-2 pairings and one wanted a more efficient instantiation one could implement a system using DDH in $\mathbb{G}_1$ and DLIN in $\mathbb{G}_2$. We do not expand on the details of this construction, but remark that this would imply that elements in $\mathbb{B}_1$ would consist of two elements in $\mathbb{G}_1$ and elements in $\mathbb{B}_2$ would consist of three elements in $\mathbb{G}_2$, with $\mathbb{B}_T$ consisting of six elements in $\mathbb{G}_T$. This added efficiency is at the expense of having to assume DDH in $\mathbb{G}_1$, which defeats the benefit which some people (although not the current authors) see behind the DLIN assumption based constructions in pairing based cryptography; namely that a single protocol description can apply in all main three pairing types.

5 Performance Comparison

In this section we compare the relative commitment sizes of the different instantiations, the resulting proof sizes can be deduced from these and show a similar relative comparison. ¿From the size of the elements in the groups $\mathbb{B}_1$ and $\mathbb{B}_2$ one can also easily estimate the relative computational performance figures, as group operations are essentially a quadratic function of the bit length. Before proceeding we also note that Groth–Sahai proofs will usually be used in the context of another protocol or scheme which is likely to dictate the exact pairing type one is using, hence the following comparison is only for illustrative purposes.

To provide concrete numbers we assume a security level equivalent to 128-bits of symmetric key security. Using "standard" comparisons of different key sizes this equates to a minimum size of $\mathbb{G}_T$ of 3072-bits and a minimum size of elements in $\mathbb{G}_1$ of 256-bits. We let k denote the pairing embedding degree. For Type-1 pairings the value of k is bounded by two for elliptic curves defined over fields of large prime characteristic and by six for curves which are defined over fields of characteristic three. For Type-2 and Type-3 curves the "optimal" value of k at this security level is $k = 12$. A crucial observation is that for Type-3 curves we have the ability to compress the elements in $\mathbb{G}_2$ by a factor of six at this security level by using BN curves.

We summarize the commitment sizes (in bits), i.e. the size of elements in $\mathbb{B}_1$ and $\mathbb{B}_2$, as well as the proof sizes (also in bits), in Table 1. From the table it would appear that using the SDLIN setting as introduced in this paper gives no advantage. However, this overlooks the fact that the point of Groth–Sahai proofs is to use them in other protocols and schemes. These protocols and schemes may require one to work in the Type-2 setting, or to base ones security on the SDLIN assumption. Thus in these situations it makes more sense to use Groth–Sahai proofs suited to the particular protocol. In addition some researchers prefer the SDLIN setting to the SXDH setting as they prefer not to use the "special" pairing setting of Type-3, where there is no computable isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$.

For the Type-1 setting we give two figures to represent the case of large prime characteristic and characteristic three. Note in all cases the size of a proof is equal to $\hat{m}_1$ elements of $\mathbb{B}_2$ and $\hat{m}_2$ elements of $\mathbb{B}_1$, except in the case of Type-1 pairings where due to the symmetric nature of the map $F(\mathcal{X}, \mathcal{Y})$ one can simplify this to $\max(\hat{m}_1, \hat{m}_2)$ elements of $\mathbb{B}_1 = \mathbb{B}_2$.

Table 1. Summary of the different instantiations

Pairing Type	1	2	3	3
Hard Problems	DLIN	SDLIN	SDLIN	SXDH
$\lvert\mathbb{G}_1\rvert$ $\lvert\mathbb{G}_2\rvert$	1536/512 1536/512	256 3072	256 512	256 512
$\lvert\mathbb{B}_1\rvert$ $\lvert\mathbb{B}_2\rvert$	$3 \cdot \lvert\mathbb{G}_1\rvert = 4608/1536$ $3 \cdot \lvert\mathbb{G}_2\rvert = 4608/1536$	$3 \cdot \lvert\mathbb{G}_1\rvert = 768$ $3 \cdot \lvert\mathbb{G}_2\rvert = 9216$	$3 \cdot \lvert\mathbb{G}_1\rvert = 768$ $3 \cdot \lvert\mathbb{G}_2\rvert = 1536$	$2 \cdot \lvert\mathbb{G}_1\rvert = 512$ $2 \cdot \lvert\mathbb{G}_2\rvert = 1024$
Pairing Product Equations				
$(\hat{m}_1, \hat{m}_2)$ Size	(3,3) 13824/4608	(3,3) 29952	(3,3) 6912	(2,2) 3072
Multi-scalar multiplication in $\mathbb{G}_1$				
$(\hat{m}_1, \hat{m}_2)$ Size	(3,2) 13824/4608	(3,2) 29184	(3,2) 6144	(2,1) 2560
Multi-scalar multiplication in $\mathbb{G}_2$				
$(\hat{m}_1, \hat{m}_2)$ Size	(2,3) 13824/4608	(2,3) 20736	(2,3) 5376	(1,2) 2048
Quadratic Equations in $\mathbb{F}_q$				
$(\hat{m}_1, \hat{m}_2)$ Size	(2,2) 9216/3072	(2,2) 19968	(2,2) 4608	(1,1) 1536

6 Summary

We have extended the Groth–Sahai techniques to pairings in the Type-2 setting, and to using the DLIN assumption in the Type-3 setting. This required us to introduce a minor extension to the DLIN hardness assumption. In doing so we corrected a number of mistakes in the formulae presented in [26]. Using our formulae all valid NIWI proofs in both the DLIN and SXDH settings will now verify.

Acknowledgements. The authors work was partially funded by the EU FP7 projects CACE and eCrypt-2. The work of the second author was supported by a Royal Society Wolfson Merit Award. The authors would like to thank Jens Groth and Amit Sahai for useful feedback on an earlier version of this manuscript.

References

1. Ateniese, G., Camenisch, J., de Medeiros, B., Hohenberger, S.: Practical group signatures without random oracles. Cryptology ePrint Archive, Report 2005/385 (2005)
2. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (2009)
3. 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. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: Compact E-Cash and simulatable VRFs revisited. In: Shacham, H. (ed.) Pairing 2009. LNCS, vol. 5671, pp. 114–131. Springer, Heidelberg (2009)
5. Bellare, M., Rogaway, P.: Random oracles are practical: A Paradigm for Designing Efficient Protocols. In: Computer and Communications Security – CCS 1993, pp. 62–73. ACM, New York (1993)
6. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications. In: Symposium on Theory of Computing – STOC 1988, pp. 103–112. ACM, New York (1988)
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., Goh, E., Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)
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. 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, vol. 5479, pp. 351–368. Springer, Heidelberg (2009)
11. Chatterjee, S., Menezes, A.: On cryptographic protocols employing asymmetric pairings – The role of Ψ revisited. Cryptology ePrint Archive, Report 2009/480 (2009)
12. Damgård, I.: Non-interactive circuit based proofs and non-interactive proofs of knowledge with preprocessing. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 341–355. Springer, Heidelberg (1993)

13. Damgård, I., Nielsen, J.B., Orlandi, C.: Essentially optimal universally composable oblivious transfer. In: Lee, P.J., Cheon, J.H. (eds.) ICISC 2008. LNCS, vol. 5461, pp. 318–335. Springer, Heidelberg (2009)
14. Feige, U., Shamir, A.: Witness indistinguishable and witness hidding protocols. In: Symposium on Theory of Computing, pp. 416–426. ACM, New York (1990)
15. Feige, U., Lapidot, D., Shamir, A.: Non-interactive zero-knowledge proofs based on a single random string. In: Foundations of Computer Science – FOCS 1990, pp. 308–317. ACM, New York (1990)
16. Freeman, D.M.: Converting pairing-based cryptosystems from composite-order groups to prime-order groups. Cryptology ePrint Archive, Report 2009/540 (2009)
17. Galbraith, S., Paterson, K., Smart, N.P.: Pairings for cryptographers. Discrete Applied Mathematics 156, 3113–3121 (2008)
18. Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. Journal of Cryptology 7, 1–32 (1994)
19. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: Symposium on Theory of Computing – STOC 1985, pp. 291–304. ACM, New York (1985)
20. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing 18, 186–208 (1989)
21. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity. Journal of the ACM 38(3), 690–728 (1991)
22. Green, M., Hohenberger, S.: Universally composable adaptive oblivious transfer. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 179–197. Springer, Heidelberg (2008)
23. Groth, J.: Fully anonymous group signatures without random oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007)
24. 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)
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. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups (full version), http://www.brics.dk/~jg/WImoduleFull.pdf
27. Groth, J., Sahai, A.: Private Communication (December 2009)
28. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Ambiguous optimistic fair exchange. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 74–89. Springer, Heidelberg (2008)
29. Kilian, J., Petrank, E.: An efficient non-interactive proof system for NP with general assumptions. Journal of Cryptology 11, 1–27 (1998)
30. Liang, X., Cao, Z., Shao, J., Lin, H.: Short group signature without random oracles. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 69–82. Springer, Heidelberg (2007)
31. Phong, L.T., Kurosawa, K., Ogata, W.: New DLOG-based convertible undeniable signature schemes in the standard model. Cryptology ePrint Archive, Report 2009/394
32. De Santis, A., Di Crescenzo, G., Persiano, G.: Randomness-optimal characterization of two NP proof systems. In: Rolim, J.D.P., Vadhan, S.P. (eds.) RANDOM 2002. LNCS, vol. 2483, pp. 179–193. Springer, Heidelberg (2002)

Constant-Round Concurrent Non-Malleable Statistically Binding Commitments and Decommitments

Zhenfu Cao[1], Ivan Visconti[2,⋆], and Zongyang Zhang[1]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, P.R. China
{zfcao,zongyangzhang}@sjtu.edu.cn

[2] Dipartimento di Informatica ed Applicazioni, University of Salerno, Italy
visconti@dia.unisa.it

Abstract. When commitment schemes are used in complex environments, e.g., the Internet, the issue of malleability appears, i.e., a concurrent man-in-the-middle adversary might generate commitments to values related to ones committed to by honest players. In the plain model, the current best solution towards resolving this problem in a constant number of rounds is the work of Ostrovsky, Persiano and Visconti (TCC' 09). They constructed a constant-round commitment scheme that is concurrent non-malleable with respect to both commitment and decommitment. However, the scheme is only computationally binding. For application scenarios where the security of receivers is of a great concern, computational binding may not suffice.

In this work, we follow the line of their work and give a construction of statistically binding commitment scheme which is concurrent non-malleable with respect to both commitment and decommitment. Our work can be seen as a complement of the work of Ostrovsky et al. in the plain model. Our construction relies on the existence of a family of pairs of claw-free permutations and only needs a constant number of communication rounds in the plain model. Our proof of security uses non-black-box techniques and satisfies the (most powerful) simulation-based definitions of non-malleability.

Keywords: commitment schemes, statistically binding, non-malleability.

1 Introduction

A commitment scheme is a two-phase interactive protocol between two parties, the committer, who holds a value, and the receiver. It enables the committer to commit itself a value while keeping it secret from the receiver. Two basic properties of a commitment scheme are the hiding property (the receiver can not learn the committed value before the decommitment phase) and the binding

⋆ Work done while Ivan Visconti was visiting UCLA, USA.

property (the committer is bounded to one value after the commitment phase). In the literature, two fundamental types of commitment schemes, statistical hiding and statistical binding, are considered.

It is well known that the basic properties of commitment schemes can not prevent "malleability" attacks mounted by a probabilistic polynomial-time (PPT) man-in-the-middle (MIM) adversary who has full control of the communication channel between the committer and the receiver. The concept of non-malleability was first introduced by Dolev et al. [1] to capture security concerns in such settings. Loosely speaking, a commitment scheme is non-malleable if one can not transform the commitment of a value into a commitment of a related value. This kind of non-malleability is called *non-malleability with respect to commitment* (NMc for short) [1]. This definition is based on the independence of the committed messages played by the MIM adversary with respect to the ones played by the committer. The notion of non-malleability used by Di Crescenzo et al. [2] is called *non-malleability with respect to decommitment or opening* (NMd for short), i.e., the adversary can not construct a commitment from a given one, such that after having seen the opening of the original commitment, the adversary is able to correctly open his commitment with a related value. This definition requires that the success probability of a MIM adversary is maintained by a stand-alone simulator. Subsequent NMc definitions are modified in a similar way [3,4,5,6,7,8]. Simulation-based definitions are much more useful when a commitment scheme is used as a building block in a larger protocol since the existence of a simulator heavily simplifies the task of proving the security of the larger protocol.

Intuitively, it seems that NMc is stronger than NMd. However, this depends on the subtleties of the definitions. Indeed this does not necessarily always hold at least with respect to non-malleability definitions in [2,1,3]. In a journal version of [3], the authors [9] presented a stringent definition of non-malleability w.r.t commitment in order to imply the notion of non-malleability w.r.t opening.

Several previous results focused on designing statistically hiding commitment schemes which are NMd. Based on number-theoretic assumptions, NMd commitment schemes were designed in [10,3] assuming the existence of a common reference string (CRS) that is shared by the two players before the protocol execution. Thus, their schemes do not work in the plain model (i.e., without setup assumptions). Recently, Pass and Rosen [4,5] presented a slightly different definition of NMd.[1] They then constructed a commitment scheme under their NMd definition based on a family of collision-resistant hash functions in the plain model. Their scheme is round-efficient and needs only constant-round communication. More recently, based on the work of [11,12], Zhang et al. [13] presented a non-malleable commitment scheme under the weakest assumption, i.e., the existence of one-way functions.

[1] More precisely, the NMd definitions in [2,10] do not take into account possible a priori information the adversary might have about the commitment received in the left interaction, while the definitions in [3,4,5] do. The definitions in [2,10,3] do not provide the stand-alone simulator the value committed in the left interaction after the commitment phase is finished, while the definitions in [4,5] do.

Before the work of [8], it was commonly believed that NMd (compared with NMc) is the only notion that makes sense in a computationally binding commitment scheme [1]. However, Ostrovsky et al. [8] argued that by slightly relaxing the NMc definition,[2] NMc can also be achieved for computationally binding commitment schemes. They considered concurrent MIM attacks where the adversary can simultaneously participate in any polynomial number of executions as a receiver and as a committer. Based on the work of [6,7], and using some techniques already introduced in [14,15] they gave a computationally hiding and computationally binding commitment scheme which is both concurrent NMc and concurrent NMd. In a full version of [8], they [16] further gave a construction of a constant-round statistically hiding commitment scheme which is concurrent NMd and that actually consists of a simplified protocol with respect to the one presented in [8]. The above schemes assume the existence of a family of pairs of claw-free permutations, require constant number of communication rounds only and assume that commitment phase and decommitment phase do not overlap in time.

For statistically binding commitment schemes, the first NMc one was designed by Dolev et al. [1] assuming the existence of one-way functions. However, the scheme requires $O(\log n)$ rounds, where n is the security parameter. In the CRS model, Di Crescenzo et al. [10] constructed very efficient NMc commitment schemes based on any public-key cryptosystem that is non-malleable under chosen plaintext attacks in addition to any shared-key cryptosystem that enjoys indistinguishability under plaintext oracle CCA-post attack. In the plain model, Pass and Rosen [4,5] first constructed a *constant-round* NMc commitment scheme assuming the existence of collision resistant hash functions. Pass and Rosen [6,7] then showed the NMc scheme of [4,5] is actually a concurrent NMc one under a stronger simulation-based definition.[3] The security proofs of [4,5,6,7] requires a non-black-box use of the code of the adversary and moreover the one of [6,7] assumes that commitment phase and decommitment phase do not overlap in time. Lin et al. [12] reconsidered the scheme of [1] and presented a concurrent NMc commitment scheme using only black-box techniques. Their scheme requires a polynomial number of communication rounds and is based on the minimal assumption, i.e., existence of one-way functions. In addition to the above results focusing on NMc, the only one that *explicitly* claimed NMd commitment schemes was designed in [9] (see Sec. 3) in the CRS model.

Before the clarification of [8], another folklore belief about a statistically binding commitment scheme is that if it is NMc then it is NMd. However, at least

[2] The values committed to by the adversary in a MIM execution are uniquely defined for all algorithms in the NMc definition [1,3], but only for PPT algorithms in the relaxed definition. More recently, the NMc definition formulated in [9] can also be applied to computationally binding commitment scheme.

[3] The NMd definition in [6,7] is stronger than that in [2]. The former is a indistinguishability-based definition, i.e., there exists a PPT stand-alone simulator that commits to a value which is computationally indistinguishable from the value committed to by the MIM adversary. The latter is a relation-based definition, i.e., the stand-alone simulator is less likely to commit to a value satisfying any polynomial-time computable relation than the value committed to by the MIM adversary.

this can not be deduced just from the simulation-based definitions in [4,5,6,7] in the plain model.[4] The main problem is that the success probability of the stand-alone simulator is required to be only negligible close to the success probability of the MIM adversary [8]. Recall in the NMc proof [4,5,6,7], a stand-alone simulator will internally simulate the left interaction for the MIM adversary by committing to a bogus value 0^n. It seems that this simulator can not handle the NMd proof, because after receiving a committed value m, the simulator is stuck to open the bogus commitment to m.

Therefore, achieving simultaneously concurrent NMc and NMd in a constant number of rounds and under the simulation-based notions, the work of [8] achieves the strongest security for commitment schemes in the plain model. However, the scheme is only computationally binding. When the security of receivers is of a great concern in some application scenarios, it may not be sufficient. Thus, there remains an open problem as to whether or not constant-round statistically binding commitment scheme that is both concurrent NMc and concurrent NMd exists in the plain model, under the stronger simulation-based definition [6,5,6,7,8].

1.1 Our Contribution

We solve the above problem by presenting a round-efficient protocol for concurrent non-malleable statistically binding commitment scheme. We show the following theorem.

Theorem 1. *Suppose that there exists a family of pairs of claw-free permutations. Then there exists a constant-round statistically binding commitment scheme that is both concurrent* NMc *and concurrent* NMd*.*

On a high level view, the commitment phase of our scheme is almost identical with that in [8]. The technique used in this phase is also the same. More precisely, in addition to the technique used by [4,5,6,7], the two-witness technique of Feige [17] is also employed. Our contribution lies in the modification of the open phase in order to simultaneously achieve concurrent NMd and statistical binding property. We borrow the idea of [18] in designing concurrent zero-knowledge *proofs*, i.e., we let the committer guess the private values committed to by the receiver in the commitment phase, and then use a witness-indistinguishable *proof* system to prove a carefully designed statement. In this way, the scheme is guaranteed to prevent any unbounded adversary from opening the commitment in two different ways.

Our work can be viewed as a complement of the work of [8]. Both of the work resolve the non-malleability issues against concurrent man-in-the-middle attacks and achieve the same-level of security in the plain model. The main difference between the two results lies in that the work of [8] focuses mainly on computationally binding commitment schemes, whereas our work considers statistically binding ones. Compared with the work of [6,7], our work also achieves both concurrent NMc and concurrent NMd, whereas they only achieve concurrent NMc.

[4] There is no problem in the CRS model. The reader is refereed to [8] for more details.

We emphasize here that our scheme inherits the limitation from [6,7,8], i.e., the non-malleability proof heavily relies on the assumption that commitment phase and the decommitment phase do not overlap in time.

2 Preliminaries

We assume the reader is familiar with witness-indistinguishable protocols, zero-knowledge protocols and commitment schemes. For more details, the reader is refereed to [19] for references.

2.1 Concurrent Non-Malleable Commitments and Decommitments

Next, we formulate the definitions of concurrent NMc and concurrent NMd. As stated in [6,7] we formalize the notion of non-malleability by a comparison between a *man-in-the-middle* execution and a *simulated* execution. Let $\langle C, R\rangle$ be a commitment scheme. Let $n \in \mathbb{N}$ be a security parameter.

The man-in-the-middle execution. In the MIM execution, the adversary $\mathcal{A}$ is simultaneously participating in $m(n) = \mathsf{poly}(n)$ left and $m(n)$ right interactions (WLOG, the number of commitments is the same in the left and right execution). In the i^{th} left interaction, $\mathcal{A}$ interacts with the committer C to receive a commitment to a value v_i. In the i^{th} right interaction, $\mathcal{A}$ interacts with the receiver R and tries to commit to a value $\tilde{v}_i$ of its choice. After the execution of the commitments in all interactions, $\mathcal{A}$ executes the decommitments with C and the decommitments with R. Prior to the interaction, the value vector $\mathbb{V} = (v_1, \ldots, v_m)$ is given to C as local inputs. $\mathcal{A}$ also receives an auxiliary input z, which might contain a priori information about $\mathbb{V}$.

Let the random variable $\mathsf{mim}^{\mathcal{A}}_{\mathsf{com}}(\mathbb{V}, z)$ denote the values $\tilde{v}_1, \ldots, \tilde{v}_m$ to which the adversary has committed in the right interactions. If the i^{th} right commitment fails, or its transcript (commitment phase) equals to the transcript of any left interaction, the value $\tilde{v}_i$ is set to $\perp$.

Similarly, we let the random variable $\mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(\mathbb{V}, z)$ denote the values $\tilde{v}_1, \ldots, \tilde{v}_m$ to which the adversary has opened in the right interactions. If the i^{th} right commitment or decommitment fails, or its transcript (both commitment phase and decommitment phase) equals to the transcript of any left interaction, the value $\tilde{v}_i$ is set to $\perp$.

The simulated execution. In the simulated execution, a simulator S directly interacts with an honest receiver R in $m(n)$ interactions. As in the MIM execution, the value vector $\mathbb{V} = (v_1, \ldots, v_m)$ is chosen prior to the interaction, and S receives some a prior information about $\mathbb{V}$ as part of its auxiliary input z. S first executes the commitment phases with R. Once all the commitment phases have been completed, S receives the value vector $\mathbb{V}$ and attempts to decommit to values $\tilde{v}_1, \ldots, \tilde{v}_m$.

Let the random variable $\mathsf{sim}^{S}_{\mathsf{com}}(\mathbb{V}, z)$ denote the values $\tilde{v}_1, \ldots, \tilde{v}_m$ committed to by S. The value $\tilde{v}_i$ is set to $\perp$ if S fails in the i^{th} commitment phase. Let the random variable $\mathsf{sim}^S_{\mathsf{open}}(\mathbb{V}, z)$ denote the values $\tilde{v}_1, \ldots, \tilde{v}_m$ opened by S. The value $\tilde{v}_i$ is set to $\perp$ if S fails in the i^{th} commitment phase or decommitment phase.

Definition 1 (Concurrent Non-Malleable Commitment w.r.t Commitment [6,7]). *A commitment scheme $\langle C, R\rangle$ is said to be* concurrent non-malleable with respect to commitment *if for every PPT man-in-the-middle adversary $\mathcal{A}$ that participates in at most $m(n)$ left and $m(n)$ right interactions, there exists a PPT simulator S such that the following two ensembles are computationally indistinguishable:*

- $\{\mathsf{mim}^{\mathcal{A}}_{\mathsf{com}}(\mathbb{V}, z)\}_{\mathbb{V}=(v_1,\ldots,v_m)\in\{0,1\}^{n*m}, z\in\{0,1\}^*}$
- $\{\mathsf{sim}^{S}_{\mathsf{com}}(\mathbb{V}, z)\}_{\mathbb{V}=(v_1,\ldots,v_m)\in\{0,1\}^{n*m}, z\in\{0,1\}^*}$

Definition 2 (Concurrent Non-Malleable Commitment w.r.t Decommitment). *A commitment scheme $\langle C, R\rangle$ is said to be* concurrent non-malleable with respect to decommitment *if for every PPT man-in-the-middle adversary $\mathcal{A}$ that participates in at most $m(n)$ left and $m(n)$ right interactions, there exists an expected PPT simulator S such that the following two ensembles are computationally indistinguishable:*

- $\{\mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(\mathbb{V}, z)\}_{\mathbb{V}=(v_1,\ldots,v_m)\in\{0,1\}^{n*m}, z\in\{0,1\}^*}$
- $\{\mathsf{sim}^{S}_{\mathsf{open}}(\mathbb{V}, z)\}_{\mathbb{V}=(v_1,\ldots,v_m)\in\{0,1\}^{n*m}, z\in\{0,1\}^*}$

A commitment scheme that is non-malleable according to Definition 2 is liberal non-malleable rather than strict non-malleable [1,3]. Note we follow [4,5,8] in that non-malleability is guaranteed only if the commitment phase and the decommitment phase do not overlap in time.

Strong signature schemes. A signature scheme $\mathsf{SS} = (\mathsf{Sgen}, \mathsf{Ssig}, \mathsf{Sver})$ is said to be *strongly unforgeable* under adaptive chosen-message attack if no efficient adversary, with access to signature oracle with respect to the verification key VK, can output a valid message/signature pair (m, σ) with non-negligible probability. Here "valid" means that $\mathsf{Sver}(\mathsf{VK}, \mathsf{m}, \sigma) = 1$ and (m, σ) does not correspond to any message/signature pair that was output by the signature oracle. A *strong signature scheme* is a signature scheme that is strongly unforgeable.

3 Constant-Round Statistically Binding Concurrent NMc and Concurrent NMd

In this section, we present a constant-round statistically binding commitment scheme that is concurrent NMc and concurrent NMd. Denote by SBCom the statistically binding commitment scheme from any one-way function [20]. Denote by SHCom the statistically hiding commitment scheme from any collection of claw-free permutation with an efficiently-recognizable index set [21]. Denote by $\{\langle \mathcal{P}_{\mathrm{tag}}, \mathcal{V}_{\mathrm{tag}}\rangle\}_{\mathrm{tag}}$ the constant-round tag-based perfect non-malleable zero-knowledge argument of knowledge (NMZKAOK) for NP [4,5]. Denote by $\langle \mathrm{swi}\mathcal{P}, \mathrm{swi}\mathcal{V} \rangle$ the constant-round statistically witness-indistinguishable argument of knowledge (WIAOK) for NP [22,23].[5] Let $\langle \mathrm{cwi}\mathcal{P}, \mathrm{cwi}\mathcal{V} \rangle$ be a constant-round computationally witness-indistinguishable proof of knowledge (WIPOK) for NP. Let SS = (Sgen, Ssig, Sver) be a strong signature scheme. The commitment scheme is shown in Fig. 1. Note that all the tools used above can be achieved assuming the existence of a family of pairs of claw-free permutations.

Our commitment scheme is a statistically binding variant of the one in [8]. The commitment phase is almost identical with that of the commitment scheme in [8] with the following exception: in Stage 2, the receiver R uses a statistically hiding commitment scheme SHCom instead of a statistically binding one. It also invokes the statistical WIAOK $\langle \mathrm{swi}\mathcal{P}, \mathrm{swi}\mathcal{V} \rangle$ instead of a computational WIPOK. Roughly, in Stage 1, the committer generates a commitment c to v and proves knowledge of opening of c. In Stage 2, the receiver generates two commitments c_0, c_1 to two secretes v_0, v_1 respectively and proves knowledge of either secret. In Stage 3, the committer generates a signature to the transcripts up to now and the receiver then verifies the correctness of the signature.

The decommitment phase is more involved and needs more careful design. The main difficulty lies in simultaneously achieving concurrent NMd and statistical binding properties. We are inspired by the work of [18] on concurrent zero-knowledge *proofs*. We modify the scheme in [8] by letting the committer guess the private values committed to in the commitment phase and then use a WIPOK to prove a carefully designed OR statements. The construction employs the two-witness technique by Feige [17] and the well known FLS-technique [24]. Roughly, in Stage $1'$, the committer first generates a commitment c' to a dummy value 0^n. After receiving c', the receiver then opens the values v_0, v_1 committed to in the commitment phase and proves knowledge of opening of either commitment c_0 or c_1. In Stage $2'$, the committer sends the committed value v and runs a computational WIPOK to prove the statements that either c is a commitment to v, or c' is a commitment to v_0 or v_1. In Stage $3'$, the committer proves that c is a commitment to v, or it knows opening of c_{b^*} to v_{b^*} for some $b^* \in \{0, 1\}$. In Stage $4'$, the committer generates a signature to the transcripts up to now and the receiver then verifies the correctness of the signature. Note that the FLS-technique is used both in Stage $2'$ and Stage $3'$.

Theorem 2. *Suppose that* SBCom *is a statistically binding commitment scheme,* SHCom *is a statistically hiding commitment scheme and* SS = (Sgen, Ssig, Sver) *is a strong signature scheme. Suppose that* $\{\langle \mathcal{P}_{\mathrm{tag}}, \mathcal{V}_{\mathrm{tag}} \rangle\}_{\mathrm{tag}}$ *is an one-many concurrent perfect* NMZKAOK *for* NP*,* $\langle \mathrm{swi}\mathcal{P}, \mathrm{swi}\mathcal{V} \rangle$ *is a statistical* WIAOK *for* NP *and* $\langle \mathrm{cwi}\mathcal{P}, \mathrm{cwi}\mathcal{V} \rangle$ *is a computational* WIPOK *for* NP*. Then* $\langle C, R \rangle$ *is a statistically binding commitment scheme that is both concurrent* NMc *and concurrent* NMd*.*

[5] Blum's basic protocol for Hamiltonicity [22] is only computational zero-knowledge with soundness error $\frac{1}{2}$. Moreover, the protocol includes three rounds of interaction. By running the basic protocol polynomial times in parallel, we get a computational WIPOK for Hamiltonicity. If the prover uses a statistically hiding commitment scheme [21] in the first round, then we get a statistical WIAOK for Hamiltonicity.

Protocol $\langle C, R\rangle$

Security Parameter: 1^n

String to be committed: $v \in \{0,1\}^n$

Commitment Phase:

Stage 1:

- $C \to R$: Let $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Sgen}(1^{\mathsf{n}})$. Pick uniformly $r \in \{0,1\}^n$ and compute $c \leftarrow \mathsf{SBCom}(v; r)$. Send pk, c.
- $C \Leftrightarrow R$: C uses witness (v, r) and proves using $\langle \mathcal{P}_{\mathsf{pk}}, \mathcal{V}_{\mathsf{pk}}\rangle$ (with tag pk) the statement that there exist values $v, r \in \{0,1\}^n$ such that $c = \mathsf{SBCom}(v; r)$.
- R : Abort if the above proof fails.

Stage 2:

- $R \to C$: Pick uniformly $v_0, r_0, v_1, r_1 \in \{0,1\}^n$ and compute $c_0 = \mathsf{SHCom}(v_0; r_0), c_1 = \mathsf{SHCom}(v_1; r_1)$. Send c_0, c_1.
- $R \to C$: Pick a random bit $b \in \{0,1\}$. R uses witness (v_b, r_b) and proves using $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V}\rangle$ that there exist values $v^*, r^* \in \{0,1\}^n$ such that $c_0 = \mathsf{SHCom}(v^*; r^*)$ or $c_1 = \mathsf{SHCom}(v^*; r^*)$.
- C : Abort if the above proof fails.

Stage 3:

- $C \to R$: Let tr_0 be the transcript of the above interaction. Compute $\sigma_0 \leftarrow \mathsf{Ssig}(\mathsf{sk}, \mathsf{tr}_0)$ and send σ_0.
- R : Verify that $\mathsf{Sver}(\mathsf{pk}, \mathsf{tr}_0, \sigma_0) = 1$.

Decommitment Phase:

Stage $1'$:

- $C \to R$: Pick uniformly $r' \in \{0,1\}^n$. Compute $c' = \mathsf{SBCom}(0^n; r')$ and send c'.
- $R \to C$: Send v_0, v_1.
- $R \Leftrightarrow C$: R uses witness r_b and proves using $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V}\rangle$ (with tag pk) the statement that there exists a value $r^* \in \{0,1\}^n$ such that $c_0 = \mathsf{SHCom}(v_0; r^*)$ or $c_1 = \mathsf{SHCom}(v_1; r^*)$.
- C : Abort if the above proof fails.

Stage $2'$:

- $C \to R$: Send v.
- $C \Leftrightarrow R$: C uses witness r and proves using $\langle \text{cwi}\mathcal{P}, \text{cwi}\mathcal{V}\rangle$ the OR of the following statements
  1. $\exists\ r \in \{0,1\}^n$ s.t $c = \mathsf{SBCom}(v; r)$,
  2. $\exists\ b^* \in \{0,1\}, r^* \in \{0,1\}^n$ s.t $c' = \mathsf{SBCom}(v_{b^*}; r^*)$.
- R**:** Abort if the above proof fails.

Stage $3'$:

- $C \Leftrightarrow R$: C uses witness r and proves using $\langle \mathcal{P}_{\mathsf{pk}}, \mathcal{V}_{\mathsf{pk}}\rangle$ (with tag pk) the statement that either there exists $r \in \{0,1\}^n$ such that $c = \mathsf{SBCom}(v; r)$, or there exist $b^* \in \{0,1\}, r^* \in \{0,1\}^n$ such that $c_{b^*} = \mathsf{SHCom}(v_{b^*}; r^*)$.
- R : Abort if the above proof fails.

Stage $4'$:

- $C \to R$: Let tr_1 be the transcript of the above interaction. Compute $\sigma_1 \leftarrow \mathsf{Ssig}(\mathsf{sk}, \mathsf{tr}_1)$ and send σ_1.
- R : Verify that $\mathsf{Sver}(\mathsf{pk}, \mathsf{tr}_1, \sigma_1) = 1$.

Fig. 1. Concurrent non-malleable statistically binding commitment scheme $\langle C, R\rangle$

Proof. We need to prove the scheme satisfies the following three properties: computational hiding, statistical binding, and concurrent NMc and concurrent NMd.

Computational hiding. Intuitively, the hiding property follows from the hiding property of SBCom and perfect zero-knowledge property of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}} \rangle$. Suppose, on the contrary, there exists an adversary R^* that violates the hiding property of $\langle C, R \rangle$. Then we design an efficient adversary R' that breaks the hiding property of SBCom. R' proceeds as follows. On input a challenge com (i.e., a commitment to m_0 or m_1) from the committer of SBCom, R' internally incorporates R^* and forwards the external commitment com to R^* in $\mathsf{Stage\ 1}$. All other executions are emulated by R' by following the honest committer strategy except that R' runs the simulator for $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}} \rangle$ in $\mathsf{Stage\ 1}$. Finally, R' outputs whatever R^* outputs. From the perfect zero-knowledge property of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}} \rangle$, if R^* distinguishes the commitment made using $\langle C, R \rangle$, then R' distinguishes the commitment made using SBCom.

Statistical binding. The proof of binding property is more subtle. We show that any malicious adversary C^* can not violate the binding property of $\langle C, R \rangle$. Intuitively, if C^* can open the commitment in two different ways, then due to the soundness property of $\langle \text{cwi}\mathcal{P}, \text{cwi}\mathcal{V} \rangle$ and the statistical binding property of SBCom, C^* must use a fake witness in the execution of $\langle \text{cwi}\mathcal{P}, \text{cwi}\mathcal{V} \rangle$ in Stage $2'$, i.e., it knows the witness to the statement that c' is a commitment to v_0 or v_1. Note that the only place that C^* might learn v_0 or v_1 before $\mathsf{Stage}\ 1'$ is in $\mathsf{Stage\ 2}$ of the commitment phase. Since both the commitments c_0, c_1 are statistically hiding and $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V} \rangle$ is statistical WI, C^* learns v_0 or v_1 only with negligible probability in $\mathsf{Stage\ 2}$. Thus, C^* makes a commitment c' to the value v_0 or v_1 only with negligible probability in $\mathsf{Stage}\ 1'$. Moreover, C^* commits using SBCom. So the second statement proved in $\mathsf{Stage}\ 2'$ is a false statement (even to an unbounded machine). According to the property of $\langle \text{cwi}\mathcal{P}, \text{cwi}\mathcal{V} \rangle$, even an unbounded C^* can not successfully execute the proof with non-negligible probability in $\mathsf{Stage}\ 2'$. This reaches a contradiction.

More in details, assume for contradiction that there exists some adversary (not necessarily PPT) $\mathcal{A}$ that is able to violate the binding property of $\langle C, R \rangle$. We show how to construct an algorithm (not necessarily PPT) A' that violates the binding property of SBCom or the hiding property of SHCom or the WI property of $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V} \rangle$. A' interacts with $\mathcal{A}$ and follows honest receiver strategy. Once the decommitment phase is finished, A' runs the extractor of $\langle \text{cwi}\mathcal{P}, \text{cwi}\mathcal{V} \rangle$ in $\mathsf{Stage}\ 2'$. According to the property of the extractor of $\langle \text{cwi}\mathcal{P}, \text{cwi}\mathcal{V} \rangle$, with overwhelming probability, A' gets a witness w. Then it must be the case that (1) $w = r$ s.t $c = \mathsf{SBCom}(v; r)$. (2) $w = r^*$ s.t $c' = \mathsf{SBCom}(v_b; r^*)$. (3) $w = r^*$ s.t $c' = \mathsf{SBCom}(v_{1-b}; r^*)$.

We now show however that case 1 happens with negligible probability only. Assume by contradiction, that it can happen with non-negligible probability. Then we can design an algorithm $\mathcal{B}$ that breaks the binding property of SBCom. $\mathcal{B}$ proceeds exactly as A'. $\mathcal{B}$ then succeeds extracting r such that $c = \mathsf{SBCom}(v; r)$. Next, we let $\mathcal{B}$ keep rewinding $\mathcal{A}$ to the beginning of the decommitment phase until case 1 happens again. $\mathcal{B}$ again extracts a witness and we denote by r^* the extracted witness. Let v^* be the opened value by $\mathcal{A}$. Now we get $c = \mathsf{SBCom}(v^*; r^*)$. According to the assumption of $\mathcal{A}$, $v \neq v^*$ with non-negligible probability. Now we find a commitment c that can be opened in two different ways. Thus, we break the binding property of SBCom.

Next we show case 2 happens with negligible probability. Suppose on the contrary, with some non-negligible probability it happens that $c' = \mathsf{SBCom}(v_b; r^*)$. We can design an algorithm $\mathcal{B}$ that breaks the WI property of $\langle \mathrm{swi}\mathcal{P}, \mathrm{swi}\mathcal{V} \rangle$. $\mathcal{B}$ then internally executes all the interactions with $\mathcal{A}$ and proceeds exactly as A' with the only exception that the proof of $\langle \mathrm{swi}\mathcal{P}, \mathrm{swi}\mathcal{V} \rangle$ in Stage 2 is generated by relaying all the messages with an external prover ($\mathcal{B}$ submits opening information of c_0 and c_1 to the external prover. The external prover then proves using a witness for c_{b^*} for some $b^* \in \{0, 1\}$). We emphasize here that $\mathcal{B}$ generates the proof of Stage $2'$ itself. Then $\mathcal{B}$ successfully simulates the interactions with $\mathcal{A}$ in the decommitment, and $\mathcal{B}$ runs the extractor of $\langle \mathrm{cwi}\mathcal{P}, \mathrm{cwi}\mathcal{V} \rangle$. By looking at the extracted witness, $\mathcal{B}$ will guess the witness used by the external prover.

Finally, we show case 3 happens with negligible probability. Suppose on the contrary, with non-negligible probability it happens that $c' = \mathsf{SBCom}(v_{1-b}; r^*)$. We then design an algorithm $\mathcal{B}$ that breaks the hiding property of SHCom. On input a challenge commitment c^* (to value $\hat{v}_0, \hat{v}_1$), $\mathcal{B}$ has to decide which value corresponds to c^*. $\mathcal{B}$ proceeds exactly as A' with the following two exceptions. The first exception lies in the handling of interaction in Stage 2 of the commitment. Here $\mathcal{B}$ first picks a random bit $b \in \{0, 1\}$, a random string $v_b \in \{0, 1\}^n$ and a uniform random string $r_b \in \{0, 1\}^n$. $\mathcal{B}$ then computes $c_b = \mathsf{SHCom}(v_b; r_b)$ and sets $c_{1-b} = c^*$. Next $\mathcal{B}$ continues the execution of Stage 2 of commitment by following the honest prover strategy of $\langle \mathrm{swi}\mathcal{P}, \mathrm{swi}\mathcal{V} \rangle$ using (v_b, r_b) as witness. The second exception lies in the handling of interaction in Stage $1'$ of decommitment. Here $\mathcal{B}$ randomly chooses a bit $b^* \in \{0, 1\}$, sends $v_b, v_{1-b} = \hat{v}_{b^*}$ to $\mathcal{A}$ and then uses witness r_b to complete the proof $\langle \mathrm{swi}\mathcal{P}, \mathrm{swi}\mathcal{V} \rangle$. Finally, if the witness r^* extracted satisfies $c' = \mathsf{SBCom}(v_{1-b}; r^*)$, $\mathcal{B}$ then outputs v_{1-b}; otherwise, $\mathcal{B}$ outputs $\hat{v}_{b'}$ for randomly chosen $b' \in \{0, 1\}$. Therefore the probability that $\mathcal{B}$ breaks the hiding property of SHCom is also non-negligible.

Concurrent non-malleability. We need to show that the scheme is concurrent NMc and concurrent NMd. The proof of concurrent NMc is almost identical with that of the proof in [8]. Note NMc only concerns the commitment phase and as we discussed previously, the commitment phase of our scheme only deviates from that of [8] when R sends commitments and plays the WIAOK in Stage 2. Indeed, in our scheme we use statistical versions of these tools while the protocol of [8] only needs the computational versions. The proof however goes through precisely as the one of [8]. We omit the details here and defer the proof in the full version.

Next, we show it is concurrent NMd. We show that for every PPT man-in-the-middle adversary $\mathcal{A}$ that participates in $m(n)$ left commitments and $m(n)$ right commitments, there exists an expected PPT simulator S such that for every PPT distinguisher D and every negligible function μ, for every value vector $\mathbb{V} = (v_1, \ldots, v_m)$ where $v_i \in \{0, 1\}^n$ and every $z \in \{0, 1\}^*$, it holds that

$$\left|\Pr[D(\mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(\mathbb{V},z))=1]-\Pr[D(\mathsf{sim}^{S}_{\mathsf{open}}(\mathbb{V},z))=1]\right|\le\mu(n). \quad (1)$$

Denote by $\mathcal{A}_{\mathsf{dec}}$ the state of $\mathcal{A}$ after the commitment phase, i.e., $\mathcal{A}_{\mathsf{dec}}$ contains $\mathcal{A}$'s description along with its configuration at that time just before the decommitment phase starts.

We proceed by giving the description of the simulator S. S on input z and security parameter 1^n interacts with external honest receivers and runs the adversary $\mathcal{A}$ internally. During the commitment phases, on a high level, S internally incorporates $\mathcal{A}$ and emulates the commitment phases of all left interactions for adversary $\mathcal{A}$ by honestly committing to 0^n, while internally emulating the right interactions as honest receivers. After all the commitment phases end, S invokes the extractors for all the proofs provided by $\mathcal{A}$ in the left and right commitments to extract all the corresponding witnesses. More precisely, for each right commitment, S runs the extractor of $\langle\mathcal{P}_{\text{tag}},\mathcal{V}_{\text{tag}}\rangle$ and we denote by $(\tilde{v}_i,\tilde{r}_i)$ the witness extracted in the i^{th} right commitment. For each left commitment, S runs the extractor of $\langle\text{swi}\mathcal{P},\text{swi}\mathcal{V}\rangle$ to get witness $(v_{b_i,i},r_{b_i,i})$ $(b_i\in\{0,1\})$. Next, S plays the commitment phases with external receivers. S follows the honest committer strategy and commits to $\tilde{v}_i$ in the i^{th} commitment phase.

Once all the commitment phases are finished, S receives a value vector $\mathbb{V}=(v_1,\ldots,v_m)$ and has to perform the decommitment phases internally with $\mathcal{A}_{\mathsf{dec}}$. S follows the honest receiver strategy in all right decommitments. The simulation of the i^{th} left decommitment is as follows. In $\mathsf{Stage}\ 1'$, S acts identically as an honest committer with the exception that S commits to $v_{b_i,i}$ instead of 0^n (using randomness $r^*_{b_i,i}$). In $\mathsf{Stage}\ 2'$, S follows the honest committer strategy with the exception that it uses the "fake" witness $r^*_{b_i,i}$ to open the commitment to v_i. In $\mathsf{Stage}\ 3'$, S uses the fake witness $r_{b_i,i}$ to complete the proof. S follows the honest committer strategy in $\mathsf{Stage}\ 4'$. Finally, for each i, if $\mathcal{A}_{\mathsf{dec}}$ has successfully completed the i^{th} right decommitment, then S completes the decommitment phase of the external execution with honest receivers by opening the commitment to $\tilde{v}_i$.

Running time of S. From the construction of S, we know that S performs at most $2m$ extraction procedures in the commitment phases. Note that the running time of the extractions in both $\langle\mathcal{P}_{\text{tag}},\mathcal{V}_{\text{tag}}\rangle$ and $\langle\text{swi}\mathcal{P},\text{swi}\mathcal{V}\rangle$ are all expected PPT. Since the extractions are executed sequentially, the running time of all extractions is also expected PPT. Furthermore, the MIM adversary $\mathcal{A}$ is a PPT algorithm and therefore invoking a copy of $\mathcal{A}$ also requires PPT. Thus, S runs in expected PPT in the commitment phases. In the decommitment phases, since S runs in a straight-line manner and no rewinding is involved, the running time of S is strict PPT. Finally, we conclude that the overall running time of S is expected PPT.

Next, we prove that the distribution of the messages opened by $\mathcal{A}$ when interacting with honest committers and honest receivers is indistinguishable from the distribution of the messages opened by $\mathcal{A}$ when interacting with S.

Indistinguishability of the simulation. We first consider the case when there is only one left commitment and $m(n)$ right commitments. Towards of showing

Equation (1) (note $\mathbb{V}$ contains only a value v), we define a sequence of hybrid experiments $\{\mathsf{HYB}_i(v,z)\}_{1\le i\le 7}$ that receive v and z as auxiliary inputs. The output of each experiment is the output of a PPT distinguisher D on input a value v and a vector of values $\tilde{V}$ whose i^{th} element is defined as follows. If the i^{th} right decommitment completes successfully and its transcript is different from the left interaction, then $\tilde{v}_i$ is the value opened in the i^{th} right interaction. Otherwise, $\tilde{v}_i$ is set to $\perp$. Let $p_i = \Pr[\mathsf{HYB}_i(v,z) = 1]$.

$\mathsf{HYB}_1(v,z)$ proceeds exactly as S except that in $\mathsf{Stage\ 1}$ of the left commitment phase, it runs the simulator of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}}\rangle$. Since the simulation is perfect we conclude that $p_1 = \Pr[D(\mathsf{sim}^S_{\mathsf{open}}(v,z)) = 1]$.

$\mathsf{HYB}_2(v,z)$ proceeds exactly as HYB_1 except that in the left commitment phase, instead of feeding $\mathcal{A}$ a commitment to 0^n in Stage 1, HYB_2 feeds $\mathcal{A}$ a commitment to v using SBCom. Since both HYB_1 and HYB_2 are efficiently computable, that $|p_1 - p_2|$ is negligible follows directly from the computational hiding property of SBCom.

$\mathsf{HYB}_3(v,z)$ proceeds exactly as HYB_2 except that it runs the simulator of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}}\rangle$ in Stage $3'$ of the left decommitment. It follows from the perfect zero-knowledge property of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}}\rangle$ that $p_3 = p_2$.

$\mathsf{HYB}_4(v,z)$ differs from HYB_3 in that it uses the real witness (i.e., decommitment of c) to complete the proof in Stage $2'$ of the left decommitment. It follows from the computational WI property of $\langle \text{cwi}\mathcal{P}, \text{cwi}\mathcal{V}\rangle$ that $|p_4 - p_3|$ is negligible.

$\mathsf{HYB}_5(v,z)$ differs from HYB_4 in that it commits to 0^n in Stage $1'$ of the left decommitment. It follows from the computational hiding property of SBCom that $|p_5 - p_4|$ is negligible.

$\mathsf{HYB}_6(v,z)$ proceeds exactly as HYB_5 except that it uses the real witness (i.e., decommitment of c) to complete the proof in Stage $3'$ of the left decommitment. It follows from the perfect zero-knowledge property of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}}\rangle$ that $p_6 = p_5$.

$\mathsf{HYB}_7(v,z)$ proceeds exactly as HYB_6 except that it does not need to run the extractor of $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V}\rangle$ in Stage 2 of left commitment phase. Since the extraction fails with negligible probability we have that $|p_7 - p_6|$ is negligible.

Note that HYB_7 differs from the real game in that it runs the simulator-extractor of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}}\rangle$ in Stage 1 of the commitment phase. Following the description of HYB_7 we know that it opens to external receivers the values it extracts from $\mathcal{A}$ at the end of the commitment phase in the simulated game. Moreover, in the real game the adversary $\mathcal{A}$ can not open its commitments in a different way (cf. Claim 1). Thus, $\mathcal{A}$ opens to external receivers the values it commits to in the commitment phase. It follows from the simulation-extractability property of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}}\rangle$ that the simulation is perfect and the extraction fails with negligible probability in each right commitment where the tag is different from that in the left decommitment (when tags are the same, the security of signature scheme is violated). Therefore, except with negligible probability, we have that HYB_7 opens to external receivers the same values opened by $\mathcal{A}$ in the real game, i.e., we have that $\big|p_7 - \Pr[D(\mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(v,z)) = 1]\big|$ is negligible.

Finally we conclude Equation (1). This completes the proof of Theorem 2.

Claim 1. *In the real game $\mathcal{A}$ can not open in a different way.*

Proof. Assume that, with some probability p, there exists $i \in [m]$ such that the value $\tilde{v}'$ opened in the i^{th} right decommitment is different from the value $\tilde{v}$ committed in the i^{th} right commitment.[6] We denote by c_0, c_1 the commitments made by R in Stage 2 of the i^{th} right commitment. Denote by b the bit such that R uses the decommitment information corresponding to the commitment c_b to complete the proof of $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V} \rangle$ in Stage 2 and Stage $1'$ of i^{th} right interaction. If $\mathcal{A}$ successfully completes the i^{th} right decommitment, then we consider the following experiment B. B on inputs i, v and z interacts with $\mathcal{A}$ and works as follows. We let B commit to v in the left commitment phase. Furthermore, B follows the honest committer strategy in the left interaction and the honest receiver strategy in all right interactions, except in Stage $1'$ of i^{th} decommitment B sends v_b and a randomly chosen v^*_{1-b} (With overwhelming probability v^*_{1-b} will be different from the v_{1-b} chosen in the commitment phase. This is important for the proof of Case 3 below.). Once the i^{th} right decommitment is over, B runs the extractor of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}} \rangle$ in Stage $3'$. Note that the views of $\mathcal{A}$ in a real execution and an execution of B are identical. So the probability that $\mathcal{A}$ opens in a different way in the execution of B is also p. According to the property of the extractor of $\langle \mathcal{P}_{\text{tag}}, \mathcal{V}_{\text{tag}} \rangle$, with probability $p' = p - \epsilon(n)$ where ϵ is a negligible function, B gets a witness $\tilde{w}$, and one of the following three cases must happen:

1. $\tilde{w} = \tilde{r}$ s.t $\tilde{c} = \mathsf{SBCom}(\tilde{v}'; \tilde{r})$.($\tilde{c}$ is the commitment generated by $\mathcal{A}$ in the i^{th} right commitment.)
2. $\tilde{w} = r^*$ s.t $c_b = \mathsf{SHCom}(v_b; r^*)$. ($v_b, v^*_{1-b}$ are the values opened by B in Stage $1'$ of i^{th} right decommitment.)
3. $\tilde{w} = r^*$ s.t $c_{1-b} = \mathsf{SHCom}(v^*_{1-b}; r^*)$.

Let $p' = p_1 + p_2 + p_3$, where p_i is the probability that Case i happens (for $i = 1, 2, 3$). By the statistical binding property of SBCom, we have that $\tilde{v}'$ must correspond to $\tilde{v}$ and thus in this case $\mathcal{A}$ does not open in a different way, therefore p_1 must be negligible. Recall in the i^{th} right commitment, B already generates commitments c_0 and c_1 to two different values v_0 and v_1 respectively and B uses knowledge of a decommitment of v_b for a random bit b in both $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V} \rangle$ of Stage 2 and $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V} \rangle$ of Stage $1'$. By the statistical hiding property of SHCom and statistical WI property of $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V} \rangle$, we have that p_2 is essentially identical to p_3, and therefore both p_2 and p_3 roughly correspond to $p'/2 - \epsilon(n)$ where ϵ is a negligible function.

We can now conclude the proof showing that p_3 must be negligible, and thus summing up p is negligible as well, therefore $\mathcal{A}$ can not open in a different way with non-negligible probability. Indeed, notice that when Case 3 happens, we have that $\mathcal{A}$ committed to some value v^*_{1-b} in c_{1-b} without having never used any opening of c_{1-b} in the two executions of $\langle \text{swi}\mathcal{P}, \text{swi}\mathcal{V} \rangle$. Now in addition to the opening in the commitment phase (note B generates the commitment itself), we

[6] The committed value is the one uniquely specified by the statistically binding commitment scheme SBCom.

get two openings for SHCom. Therefore being SHCom computationally binding, this happens with negligible probability.

Extending to many-many concurrent NMd. Next, we present the proof sketch for the many-many concurrent case. We show that the two ensembles $\{\mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(\mathbb{V}, z)\}$ and $\{\mathsf{sim}^{S}_{\mathsf{open}}(\mathbb{V}, z)\}$ are computationally indistinguishable. Suppose, for contradiction, this is not the case. That is, there exists a PPT distinguisher D and a polynomial $p(n)$ such that for infinitely many $n \in \mathbb{N}$, there exists a value vector $\mathbb{V} = (v_1, \ldots, v_m), z \in \{0,1\}^*$ such that D distinguishes $\mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(\mathbb{V}, z)$ and $\mathsf{sim}^{S}_{\mathsf{open}}(\mathbb{V}, z)$ with probability at least $\frac{1}{p(n)}$. For a generic n for which this happens. We design a sequence of hybrid experiments $\{\mathsf{HYB}_i(\mathbb{V}, z)\}_{0 \le i \le m}$ where $\mathsf{HYB}_i(\mathbb{V}, z)$ is defined as follows. HYB_i proceeds as S except that it emulates the i^{th} left commitment phase by committing to v_i, if $j \le i$, and 0^n otherwise. Moreover, it emulates the i^{th} left decommitment by using a legal witness to v_i, if $j \le i$, and a false witness otherwise. It directly follows that $\mathsf{sim}^{\mathsf{HYB}_m}_{\mathsf{open}}(\mathbb{V}, z) = \mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(\mathbb{V}, z)$ and $\mathsf{sim}^{\mathsf{HYB}_0}_{\mathsf{open}}(\mathbb{V}, z) = \mathsf{sim}^{S}_{\mathsf{open}}(\mathbb{V}, z)$. By a standard hybrid argument there exists an $i \in [m]$ such that

$$\left|\Pr[D(\mathsf{sim}^{\mathsf{HYB}_{i-1}}_{\mathsf{open}}(\mathbb{V}, z)) = 1] - \Pr[D(\mathsf{sim}^{\mathsf{HYB}_i}_{\mathsf{open}}(\mathbb{V}, z)) = 1]\right| > \frac{1}{m \cdot p(n)} \qquad (2)$$

Note that the only difference between experiment $\mathsf{HYB}_{i-1}(\mathbb{V}, z)$ and $\mathsf{HYB}_i(\mathbb{V}, z)$ is that in the former $\mathcal{A}$ receives a commitment to v_i and its corresponding decommitment generated using a valid witness in the i^{th} interaction, whereas in the latter it receives a commitment to 0^n and its corresponding decommitment generated using a false witness.

Then we design an efficient MIM adversary $\tilde{A}$ that breaks the one-many concurrent non-malleability of $\langle C, R\rangle$. $\tilde{A}$ on auxiliary inputs $z' = (n, i, \mathbb{V}, z)$ proceeds as follows. $\tilde{A}$ internally incorporates $A(z)$ and emulates the left and right interactions for $\mathcal{A}$. $\tilde{A}$ relays the messages in all right interactions between $\mathcal{A}$ and external receivers. In the i^{th} left interaction, $\tilde{A}$ relays either messages between an external committer and $\mathcal{A}$, or messages between the simulator of the one-many concurrent case and $\mathcal{A}$. For $j \in [m]$ and $j \neq i$, $\tilde{A}$ internally emulates the i^{th} left commitment phase for A by committing to v_i, if $j \le i$, and 0^n otherwise. Moreover, $\tilde{A}$ emulates the i^{th} left decommitment for $\mathcal{A}$ by using a valid witness, if $j \le i$, and a false witness otherwise. By construction, it follows that $\mathsf{sim}^{S}_{\mathsf{open}}(\mathbb{V}, z) = \mathsf{sim}^{\mathsf{HYB}_{i-1}}_{\mathsf{open}}(z')$ and $\mathsf{mim}^{\mathcal{A}}_{\mathsf{open}}(\mathbb{V}, z) = \mathsf{sim}^{\mathsf{HYB}_i}_{\mathsf{open}}(z')$.

Therefore, $\tilde{A}$ breaks the one-many concurrent non-malleability of $\langle C, R\rangle$.

4 Concluding Remarks

Our result on top of previous work shows that there exist constant-round commitment schemes that are secure also against very powerful adversaries, as long as there is a barrier in time between commitment and decommitment phase. An interesting open question concerns the possibility of achieving commitment schemes that remain secure even without such a barrier. The question is interesting even without requiring a constant round complexity.[7]

Acknowledgments. We thank the anonymous reviewers for their helpful suggestions. The work of the first and third authors are supported in part by the National Natural Science Foundation of China under grant No.60773086, No.60972034 and No.60970110. The work of the second author has been supported in part by the European Commission through the EU ICT program under Contract ICT-2007-216646 ECRYPT II, and in part by a research grant received from Provincia di Salerno.

References

1. Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIAM J. Comput. 30(2), 391–437 (2000)
2. Crescenzo, G.D., Ishai, Y., Ostrovsky, R.: Non-interactive and non-malleable commitment. In: STOC 1998: Proceedings of the thirtieth annual ACM symposium on Theory of computing, pp. 141–150. ACM, New York (1998)
3. Fischlin, M., Fischlin, R.: Efficient non-malleable commitment schemes. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 413–431. Springer, Heidelberg (2000)
4. Pass, R., Rosen, A.: New and improved constructions of non-malleable cryptographic protocols. In: Gabow, H.N., Fagin, R. (eds.) STOC, pp. 533–542. ACM, New York (2005)
5. Pass, R., Rosen, A.: New and improved constructions of nonmalleable cryptographic protocols. SIAM J. Comput. 38(2), 702–752 (2008)
6. Pass, R., Rosen, A.: Concurrent non-malleable commitments. In: FOCS, pp. 563–572. IEEE Computer Society, Los Alamitos (2005)
7. Pass, R., Rosen, A.: Concurrent nonmalleable commitments. SIAM J. Comput. 37(6), 1891–1925 (2008)
8. Ostrovsky, R., Persiano, G., Visconti, I.: Simulation-based concurrent non-malleable commitments and decommitments. In: Reingold, O. (ed.) Theory of Cryptography. LNCS, vol. 5444, pp. 91–108. Springer, Heidelberg (2009)
9. Fischlin, M., Fischlin, R.: Efficient non-malleable commitment schemes. J. Cryptology 22(4), 530–571 (2009)
10. 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)
11. Haitner, I., Reingold, O.: Statistically-hiding commitment from any one-way function. In: Johnson, D.S., Feige, U. (eds.) STOC, pp. 1–10. ACM, New York (2007)
12. Lin, H., Pass, R., Venkitasubramaniam, M.: Concurrent non-malleable commitments from any one-way function. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 571–588. Springer, Heidelberg (2008)
13. Zhang, Z., Cao, Z., Ding, N., Ma, R.: Non-malleable statistically hiding commitment from any one-way function. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 303–318. Springer, Heidelberg (2009)

[7] Indeed, in this case one could also use the concurrent non-malleable zero knowledge argument of [25], and the one of [26] when some efficiency is also required.

14. Ostrovsky, R., Persiano, G., Visconti, I.: Concurrent non-malleable witness indistinguishability and its applications. Electronic Colloquium on Computational Complexity (ECCC) 13(95) (2006)
15. Ostrovsky, R., Persiano, G., Visconti, I.: Constant-round concurrent non-malleable zero knowledge in the bare public-key model. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 548–559. Springer, Heidelberg (2008)
16. Ostrovsky, R., Persiano, G., Visconti, I.: Concurrent non-malleable commitments and decommitments. Full version, unpublished manuscript (2009)
17. Feige, U.: Alternative Models for Zero Knowledge Interactive Proofs. PhD thesis, The Weizmann Institute of Science, Rehovot, Israel (1990)
18. Richardson, R., Kilian, J.: On the concurrent composition of zero-knowledge proofs. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 415–431. Springer, Heidelberg (1999)
19. Goldreich, O.: Foundations of Cryptography Volume II Basic Applications. Cambridge University Press, Cambridge (2004)
20. Naor, M.: Bit commitment using pseudorandomness. J. Cryptology 4(2), 151–158 (1991)
21. Goldreich, O., Kahan, A.: How to construct constant-round zero-knowledge proof systems for NP. J. Cryptology 9(3), 167–190 (1996)
22. Blum, M.: How to prove a theorem so no one else can claim it. In: Proceedings of the International Congress of Mathematicians, pp. 1444–1451 (1986)
23. Feige, U., Shamir, A.: Witness indistinguishable and witness hiding protocols. In: STOC, pp. 416–426. ACM, New York (1990)
24. Feige, U., Lapidot, D., Shamir, A.: Multiple noninteractive zero knowledge proofs under general assumptions. SIAM J. Comput. 29(1), 1–28 (1999)
25. Barak, B., Prabhakaran, M., Sahai, A.: Concurrent non-malleable zero knowledge. In: FOCS, pp. 345–354. IEEE Computer Society, Los Alamitos (2006)
26. Ostrovsky, R., Pandey, O., Visconti, I.: Efficiency preserving transformations for concurrent non-malleable zero knowledge. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 535–552. Springer, Heidelberg (2010)

Faster Squaring in the Cyclotomic Subgroup of Sixth Degree Extensions

Robert Granger and Michael Scott*

Claude Shannon Institute
School of Computing, Dublin City University
Glasnevin, Dublin 9, Ireland
{rgranger,mike}@computing.dcu.ie

Abstract. This paper describes an extremely efficient squaring operation in the so-called 'cyclotomic subgroup' of $\mathbb{F}_{q^6}^{\times}$, for $q \equiv 1 \bmod 6$. Our result arises from considering the Weil restriction of scalars of this group from $\mathbb{F}_{q^6}$ to $\mathbb{F}_{q^2}$, and provides efficiency improvements for both pairing-based and torus-based cryptographic protocols. In particular we argue that such fields are ideally suited for the latter when the field characteristic satisfies $p \equiv 1 \pmod 6$, and since torus-based techniques can be applied to the former, we present a compelling argument for the adoption of a single approach to efficient field arithmetic for pairing-based cryptography.

Keywords: Pairing-based cryptography, torus-based cryptography, finite field arithmetic.

1 Introduction

Pairing-based cryptography has provoked a wealth of research activity since the first cryptographically constructive application of pairings was proposed by Joux in 2000 [21]. Since then, numerous further applications of pairings have been proposed and their place in the modern cryptographers' toolkit is now well established. As a result, much research activity has focused on algorithmic, arithmetic and implementation issues in the computation of pairings themselves, in order to ensure the viability of such systems [3,12,2,18].

In practise, pairings are typically instantiated using an elliptic or a hyperelliptic curve over a finite field, via the Weil or Tate pairing (see [6]) - or a variant of the latter such as the ate [18], or R-ate pairing [25]. These pairings map pairs of points on such curves to elements of a subgroup of the multiplicative group of an extension field, which is contained in the so-called cyclotomic subgroup.

Properties of the cyclotomic subgroup can be exploited to obtain faster arithmetic or more compact representations than are possible for general elements of the extension field. Cryptosystems such as LUC [33] and XTR [26], and the observations of Stam and Lenstra [34] and Granger, Page and Stam [16], all exploit

* Research supported by the Claude Shannon Institute, Science Foundation Ireland Grant No. 06/MI/006.

membership of this subgroup to achieve fast exponentiation. Many pairing-based protocols require exponentiation in the cyclotomic subgroup, as does the 'hard' part of the final exponentiation of a pairing computation, and so these ideas can naturally be applied in this context [31,17].

Currently there is a huge range of parametrisation options and algorithmic choices to be made when implementing pairings, and in order to facilitate a simple and unified approach to the construction of extension fields used in pairings, in 2005 Koblitz and Menezes introduced the concept of Pairing-Friendly Fields (PFFs) [24]. These are extension fields $\mathbb{F}_{p^k}$ with $p \equiv 1 \pmod{12}$ and $k = 2^a 3^b$, with $a \geq 1$ and $b \geq 0$. Such specialisation enables algorithms and implementations to be highly optimised. Indeed for ordinary elliptic curves the 2008 IEEE 'Draft Standard for Identity-based Public-key Cryptography using Pairings' (P1636.3/D1) deals exclusively with fields of this form [19].

In 2006 Granger, Page and Smart proposed a method for fast squaring in the cyclotomic subgroup of PFFs [15]. However even for degree six extensions the method was almost 50% slower than the Stam-Lenstra result [34]; the latter however does not permit the use of the highly efficient sextic twists available to the former, and so is not practical in this context. Both of these methods rely on taking the Weil restriction of scalars of the equation that defines membership of the cyclotomic subgroup, in order to obtain a variety over $\mathbb{F}_p$. The defining equations of this variety are then exploited to improve squaring efficiency. Rather than descend to the base field $\mathbb{F}_p$, in this paper we show that descending to only a cubic subfield enables one to square with the same efficiency as Stam-Lenstra for degree six extensions, and for between 60% and 75% the cost of the next best method for the cryptographically interesting extension degrees $12, 18$ and 24.

In tandem with the results of [5] which show that PFFs are not always the most efficient field constructions for pairing-based cryptography, we present a compelling argument for the adoption of a single approach to efficient field arithmetic for pairing-based cryptography, based on the use of fields of the form $\mathbb{F}_{q^6}^{\times}$, for $q \equiv 1 \bmod 6$. While these fields intersect with those listed in [19] - lending strong support to their possible standardisation - since these recommendations can be improved upon and since in the latest draft of this standard the recommended security parameters section is empty [20], we believe that our proposed fields should now be given serious consideration for inclusion.

The sequel is organised as follows. In §2 we describe our field construction and in §3 present our fast squaring formulae. Then in §4 we compare our approach with previous results, and in §5 and §6 apply our result to pairing-based and torus-based cryptography respectively. We conclude in §7.

2 Pairing, Towering and Squaring-Friendly Fields

Pairing-friendly fields were introduced to allow the easy construction of, and efficient arithmetic within extension fields relevant to pairing-based cryptography (PBC), and are very closely related to Optimal Extension Fields [1]. In particular we have the following result from [24]:

Theorem 1. *Let $\mathbb{F}_{p^k}$ be a PFF, and let β be an element of $\mathbb{F}_p$ that is neither a square nor a cube in $\mathbb{F}_p$. Then the polynomial $X^k - \beta$ is irreducible over $\mathbb{F}_p$.*

Observe that for 'small' β, reduction modulo $X^k - \beta$ can be implemented very efficiently. Observe also that the form of the extension degree is important for applications. When $6 \mid k$ the presence of sextic twists for elliptic curves with discriminant $D = 3$ allows for very efficient pairing computation, while for $4 \mid k$ one can use the slightly less efficient quartic twists. Such extensions also permit the use of compression methods based on taking traces [33,26], or utilising the rationality of algebraic tori [30]. Furthermore $\mathbb{F}_{p^k}$ may be constructed as a sequence of Kummer extensions, by successively adjoining the square or cube root of β, then the square or cube root of that, as appropriate, until the full extension is reached.

As shown in [5], the condition $p \equiv 1 \pmod{12}$ is somewhat spurious in that PFFs do not always yield the most efficient extension towers, and does not allow for families of pairing-friendly curves that have since been discovered [22]. For the Barreto-Naehrig curves for example [4], which have embedding degree twelve, $p \equiv 3 \bmod 4$ is preferred since one can use the highly efficient quadratic subfield $\mathbb{F}_{p^2} = \mathbb{F}_p[x]/(x^2 + 1)$. To allow for the inclusion of such fields, Benger and Scott introduced the following concept [5]:

Definition 1. *A Towering-Friendly Field (TFF) is a field of the form $\mathbb{F}_{q^m}$ for which all prime divisors of m also divide $q - 1$.*

As with PFFs, TFFs allow a given tower of field extensions to be constructed via successive root extractions, but importantly stipulate less exclusive congruency conditions on the base field cardinality. For example, as above for BN-curves with $p \equiv 3 \pmod 4$, the extension $\mathbb{F}_{p^{12}}$ is not a PFF, whereas the degree six extension of $\mathbb{F}_{p^2}$ is towering-friendly, since $p^2 - 1 \equiv 0 \pmod 6$, cf. §5.2. This definition thus captures those considerations relevant to pairing-based cryptography (PBC). We refer the reader to [5] for details of the construction of efficient TFFs.

All of the fields for PBC that follow shall be TFFs of special extension degree $k = 2^a 3^b$, with $a, b \geq 1$, i.e., with $6 \mid k$. Should it not cause confusion, we also refer to any field of the form $\mathbb{F}_{q^6}$ for which $q \equiv 1 \bmod 6$ as a *Squaring-Friendly Field (SFF)*, a name whose aptness will become clear in §3. Thus all SFFs are TFFs and all TFFs used for PBC in this paper are SFFs.

3 New Fast Squaring in the Cyclotomic Subgroup

In this section we derive efficient squaring formulae for elements of the cyclotomic subgroup of TFFs, when the extension degree is of the form $k = 2^a 3^b$, with $a, b \geq 1$, i.e., for SFFs. This is the subgroup of $\mathbb{F}_{p^k}^{\times}$ of order $\Phi_k(p)$, where Φ_k is the k-th cyclotomic polynomial, which for $6 \mid k$ is always of the form:

$$\Phi_{2^a 3^b}(x) = x^{2 \cdot 2^{a-1} 3^{b-1}} - x^{2^{a-1} 3^{b-1}} + 1.$$

We denote the cyclotomic subgroup by $G_{\Phi_k(p)}$, the membership of which can be defined as follows:

$$G_{\Phi_k(p)} = \{\alpha \in \mathbb{F}_{p^k} \mid \alpha^{\Phi_k(p)} = 1\}. \tag{1}$$

The condition on α in (1) defines a variety V over $\mathbb{F}_{p^k}$. For $d \mid k$ let $\mathbb{F}_{p^d} \subset \mathbb{F}_{p^k}$. We write $\mathrm{Res}_{\mathbb{F}_{p^k}/\mathbb{F}_{p^d}} V$ for the Weil restriction of scalars of V from $\mathbb{F}_{p^k}$ to $\mathbb{F}_{p^d}$. Then $\mathrm{Res}_{\mathbb{F}_{p^k}/\mathbb{F}_{p^d}} V$ is a variety defined over $\mathbb{F}_{p^d}$ for which we have a morphism

$$\eta : \mathrm{Res}_{\mathbb{F}_{p^k}/\mathbb{F}_{p^d}} V \to V$$

defined over $\mathbb{F}_{p^k}$ that induces an isomorphism

$$\eta : (\mathrm{Res}_{\mathbb{F}_{p^k}/\mathbb{F}_{p^d}} V)(\mathbb{F}_{p^d}) \to V(\mathbb{F}_{p^k}).$$

We refer the reader to Section 1.3 of [37] for more on the restriction of scalars.

While not stated explicitly, all prior results for fast squaring in $G_{\Phi_k(p)}$ exploit the form of the Weil restriction of this variety to a subfield. Stam and Lenstra restrict $G_{\Phi_6(p)}$ from $\mathbb{F}_{p^6}$ to $\mathbb{F}_p$ and $G_{\Phi_2(p)}$ from $\mathbb{F}_{p^2}$ to $\mathbb{F}_p$ [34], and similarly Granger *et al.* restrict $G_{\Phi_k(p)}$ from $\mathbb{F}_{p^k}$ to $\mathbb{F}_p$ [15].

Observe that $\Phi_{2^a 3^b}(x) = \Phi_6(x^{2^{a-1}3^{b-1}})$ and so we have the following simplification:

$$G_{\Phi_k(p)} = G_{\Phi_6(p^{k/6})}.$$

We therefore need only consider $G_{\Phi_6(q)}$ where $q = p^{k/6}$. Observe also that $\Phi_6(q) \mid \Phi_2(q^3)$ and so

$$G_{\Phi_6(q)} \subset G_{\Phi_2(q^3)}.$$

Hence one can alternatively employ the simplest non-trivial restriction of $G_{\Phi_6(q)}$, or rather of $G_{\Phi_2(q^3)}$, from $\mathbb{F}_{q^6}$ to $\mathbb{F}_{q^3}$, as in [34]. This reduces the cost of squaring in $G_{\Phi_2(q^3)}$, and hence in $G_{\Phi_6(q)}$, from two $\mathbb{F}_{q^3}$-multiplications to two $\mathbb{F}_{q^3}$-squarings, as we shall see in §3.1.

Our simple idea is to use the next non-trivial Weil restriction of $G_{\Phi_6(q)}$, which is from $\mathbb{F}_{q^6}$ to $\mathbb{F}_{q^2}$. This rather fortuitously provides the fastest squaring formulae yet discovered for the cyclotomic subgroups of SFFs, making an even greater efficiency gain than the Stam-Lenstra formulae for $G_{\Phi_2(q^3)}$ (cf. Table 1), while providing a systematic and more general framework than the more ad-hoc method of [34]. Restrictions to other subfields for higher extension degrees of interest do not seem to yield better results, however we leave this as an open problem.

3.1 Fast Squaring in $\mathrm{Res}_{\mathbb{F}_{q^2}/\mathbb{F}_q} G_{\Phi_2(q)}$

Let $\mathbb{F}_{q^2} = \mathbb{F}_q[x]/(x^2 - i)$ with i a quadratic non-residue in $\mathbb{F}_q$, and consider the square of a generic element $\alpha = a + bx$:

$$\alpha^2 = (a+xb)^2 = a^2+2abx+b^2x^2 = a^2+ib^2+2abx = (a+ib)(a+b)-ab(1+i)+2abx.$$

This operation can be performed at the cost of two $\mathbb{F}_q$-multiplications, and a few additions.

If however $\alpha \in G_{\Phi_2(q)}$, we have $\alpha^{q+1} = 1$, or $\alpha^q \cdot \alpha = 1$. Observe that:

$$\alpha^q = (a + xb)^q = a + bx^q = a + bx^{2(q-1)/2} \cdot x = a + bi^{(q-1)/2} \cdot x = a - bx,$$

since i is a quadratic non-residue. Hence the variety defined by the cyclotomic subgroup membership equation (1) is $(a + xb)(a - xb) = 1$, or $a^2 - x^2b^2 = 1$, or $a^2 - ib^2 = 1$. Note that this results in just one equation over $\mathbb{F}_q$, rather than two. Substituting from this equation into the squaring formula, one obtains

$$\alpha^2 = (a + xb)^2 = 2a^2 - 1 + [(a + b)^2 - a^2 - (a^2 - 1)/i]x,$$

where now the main cost of computing this is just two $\mathbb{F}_q$-squarings. Observe that if i is 'small' (for example if $i = -1$ for $p \equiv 3 \pmod 4$ when $\mathbb{F}_q = \mathbb{F}_p$), then the above simplifies considerably.

3.2 Fast Squaring in $\mathrm{Res}_{\mathbb{F}_{q^6}/\mathbb{F}_{q^2}} G_{\Phi_6(q)}$

Let $\mathbb{F}_{q^6} = \mathbb{F}_q[z]/(z^6 - i)$, with $i \in \mathbb{F}_q$ a sextic non-residue. The standard representation for a general element of this extension is

$$\alpha = \alpha_0 + \alpha_1 z + \alpha_2 z^2 + \alpha_3 z^3 + \alpha_4 z^4 + \alpha_5 z^5.$$

However, in order to make the subfield structure explicit, we write elements of $\mathbb{F}_{q^6}$ in two possible ways, each of which will be convenient depending on the context: firstly as a compositum of $\mathbb{F}_{q^2}$ and $\mathbb{F}_{q^3}$, and secondly as cubic extension of a quadratic extension.

$\mathbb{F}_{q^6}$ as a compositum. Let

$$\alpha = (a_0 + a_1 y) + (b_0 + b_1 y)x + (c_0 + c_1 y)x^2 = a + bx + cx^2, \tag{2}$$

where $\mathbb{F}_{q^2} = \mathbb{F}_q[y]/(y^2 - i)$ with $y = z^3$, and $\mathbb{F}_{q^3} = \mathbb{F}_q[x]/(x^3 - i)$ with $x = z^2$. Note that $a, b, c \in \mathbb{F}_{q^2}$. One can therefore regard this extension as the compositum of the stated degree two and degree three extensions of $\mathbb{F}_q$:

$$\mathbb{F}_{q^6} = \mathbb{F}_q(z) = \mathbb{F}_{q^3}(y) = \mathbb{F}_{q^2}(x),$$

with the isomorphisms as given above. Viewing α in the latter form its square is simply:

$$\begin{aligned}\alpha^2 &= (a + bx + cx^2)^2 = a^2 + 2abx + (2ac + b^2)x^2 + 2bcx^3 + c^2x^4 \\ &= (a^2 + 2ibc) + (2ab + ic^2)x + (2ac + b^2)x^2 = A + Bx + Cx^2 \end{aligned} \tag{3}$$

As before we use the characterising equation (1) for membership of $G_{\Phi_6(q)}$, which in this case is $\alpha^{q^2-q+1} = 1$. To Weil restrict to $\mathbb{F}_{q^2}$, we first calculate how the Frobenius automorphism acts on our chosen basis. Firstly, since i is a quadratic non-residue, we have

$$y^q = y^{2(q-1)/2} \cdot y = i^{(q-1)/2} \cdot y = -y.$$

Hence $a^q = (a_0 + a_1 y)^q = a_0 - a_1 y$, which for simplicity we write as $\bar{a}$, and similarly for b^q and c^q. Furthermore, since i is a cubic non-residue we have

$$x^q = x^{3(q-1)/3} \cdot x = i^{(q-1)/3} \cdot x = \omega x,$$

where ω is a primitive cube root of unity in $\mathbb{F}_q$. Applying the Frobenius again gives $x^{q^2} = \omega^2 x$. Note that the above computations necessitate $q \equiv 1 \pmod 6$, which is satisfied thanks to the definition of SFFs.

The cyclotomic subgroup membership equation, rewritten as $\alpha^{q^2} \cdot \alpha = \alpha^q$ is therefore:

$$(a + b\omega^2 x + c\omega^4 x^2)(a + bx + cx^2) = \bar{a} + \bar{b}\omega x + \bar{c}\omega^2 x^2,$$

which upon expanding, reducing modulo $x^3 - i$, and modulo $\Phi_3(\omega) = \omega^2 + \omega + 1$, becomes

$$(a^2 - \bar{a} - bci) + \omega(ic^2 - \bar{b} - ab)x + \omega^2(b^2 - \bar{c} - ac)x^2 = 0. \tag{4}$$

This equation defines the variety $\mathrm{Res}_{\mathbb{F}_{q^6}/\mathbb{F}_{q^2}} G_{\Phi_6(q)}$, as each $\mathbb{F}_{q^2}$ coefficient of x^i equals zero. Solving for bc, ab, ac, one obtains:

$$\begin{aligned} bc &= (a^2 - \bar{a})/i \\ ab &= ic^2 - \bar{b} \\ ac &= b^2 - \bar{c} \end{aligned}$$

Substituting these into the original squaring formula (3) then gives

$$\begin{aligned} A &= a^2 + 2ibc = a^2 + 2i(a^2 - \bar{a})/i = 3a^2 - 2\bar{a}, \\ B &= ic^2 + 2ab = ic^2 + 2(ic^2 - \bar{b}) = 3ic^2 - 2\bar{b}, \\ C &= b^2 + 2ac = b^2 + 2(b^2 - \bar{c}) = 3b^2 - 2\bar{c}. \end{aligned}$$

$\mathbb{F}_{q^6}$ as a cubic over a quadratic extension. As before let $\mathbb{F}_{q^6} = \mathbb{F}_q[z]/(z^6 - i)$, with $i \in \mathbb{F}_q$ a sextic non-residue. Let the tower of extensions be given explicitly by $\mathbb{F}_{q^2} = \mathbb{F}_q[y]/(y^2 - i)$, and $\mathbb{F}_{q^6} = \mathbb{F}_{q^2}[x]/(x^3 - \sqrt{i})$, with elements represented in the basis:

$$\alpha = (a_0 + a_1 y) + (b_0 + b_1 y)x + (c_0 + c_1 y)x^2 = a + bx + cx^2,$$

which is superficially the same as equation (2), but where now the isomorphism is given by $y = z^3, x = z$. The squaring formula is identical to (3) with $i \leftarrow \sqrt{i}$.

With this representation one can see that the Frobenius automorphism acts on x as multiplication by a sixth root of unity in $\mathbb{F}_q$, which we shall also call ω. Noting that $q \equiv 1 \pmod 6$ observe that:

$$x^q = x^{q-1} \cdot x = x^{3(q-1)/3} \cdot x = \sqrt{i}^{(q-1)/3} \cdot x = i^{(q-1)/6} \cdot x.$$

Since i is a sextic non-residue in $\mathbb{F}_q$, we have that $\omega = i^{(q-1)/6}$ is a primitive sixth root of unity in $\mathbb{F}_q$. Hence $x^q = \omega x$, and similarly $x^{q^2} = (\omega x)^q = \omega^2 x$.

This simplifies the cyclotomic subgroup membership equation (1) to:

$$(a + b\omega^2 x + c\omega^4 x^2)(a + bx + cx^2) = \bar{a} + \bar{b}\omega x + \bar{c}\omega^2 x^2,$$

which upon expanding, reducing modulo $x^3 - \sqrt{i}$, and modulo $\Phi_6(\omega) = \omega^2 - \omega + 1$, becomes

$$(a^2 - \bar{a} - bc\sqrt{i}) - \omega(\sqrt{i}c^2 + \bar{b} - ab)x + \omega^2(b^2 - \bar{c} - ac)x^2 = 0. \tag{5}$$

Solving for bc, ab, ac, one obtains:

$$\begin{aligned} bc &= (a^2 - \bar{a})/\sqrt{i} \\ ab &= \sqrt{i}c^2 + \bar{b} \\ ac &= b^2 - \bar{c} \end{aligned}$$

Substituting these into the revised squaring formula gives

$$\begin{aligned} A &= a^2 + 2\sqrt{i}bc = a^2 + 2\sqrt{i}(a^2 - \bar{a})/\sqrt{i} = 3a^2 - 2\bar{a} \\ B &= \sqrt{i}c^2 + 2ab = \sqrt{i}c^2 + 2(\sqrt{i}c^2 + \bar{b}) = 3\sqrt{i}c^2 + 2\bar{b} \\ C &= b^2 + 2ac = b^2 + 2(b^2 - \bar{c}) = 3b^2 - 2\bar{c} \end{aligned}$$

3.3 Observations

Both sets of formulae for these degree six extensions are remarkably simple, requiring just three $\mathbb{F}_{q^2}$-squarings to square an element of $G_{\Phi_6(q)}$, which is analogous to the result in §3.1 that requires two $\mathbb{F}_q$-squarings to square an element of $G_{\Phi_2(q)}$. Combining the result from §3.1 for squaring a generic element of $\mathbb{F}_{q^2}$, means that for SFFs squaring in $G_{\Phi_6(q)}$ requires only six $\mathbb{F}_q$-multiplications, which matches the result of Stam and Lenstra.

Strictly speaking, the formulae require knowledge of the action of the Frobenius on elements of $\mathbb{F}_{q^2}$, which although as simple as it is, does entail Weil restriction of equations (4) and (5) to $\mathbb{F}_q$. However, if one ignores the arithmetic of $\mathbb{F}_{q^2}$ and restricts directly to $\mathbb{F}_q$ as in [15], then the above formulae are obscured and indeed were missed by Granger *et al.* So it is in the sense that the formulae were discovered in this way that we mean the restriction is to $\mathbb{F}_{q^2}$ only.

Observe also that for the extension tower, one needs to multiply by $\sqrt{i} \in \mathbb{F}_{q^2}$, whereas for the compositum one has $i \in \mathbb{F}_q$. The cost of the former is however not much more than the latter since the basis for $\mathbb{F}_{q^2}/\mathbb{F}_q$ is $\{1, \sqrt{i}\}$ and so multiplication by $\sqrt{i}$ of a value in $\mathbb{F}_{q^2}$ involves just a component swap and a multiplication by i.

4 Comparison with Prior Work

In this section we compare the efficiency of the squaring formulae derived in §3 with the most efficient results in the literature.

4.1 Operation Counts

Let m and s be the time required to perform an $\mathbb{F}_q$-multiplication and an $\mathbb{F}_q$-squaring respectively. Since the cost of computing a squaring using our formulae or others reduces to computing squarings in a subfield, we use the notation S_d and M_d to denote the time required to compute the square of one or the product of two generic elements of $\mathbb{F}_{q^d}$. In our estimates we do not include the time for modular additions and subtractions since although not negligible, are not the dominant operations. This assumes that multiplication by the elements i, j used in §3.1 and §3.2 can be effected with very few modular shifts and additions when needed, see [5] for justification of this assumption.

We focus on TFFs with extension degrees $6, 12, 18$ and 24 over $\mathbb{F}_q$, which are the main extension degrees of interest in PBC. However one can easily extrapolate cost estimates for any field whose extension degree is of the form $k = 2^a 3^b$. To estimate the cost of a multiplication in $\mathbb{F}_{q^k}$, we use the function $\nu(k)m$ where $\nu(k) = 3^a 6^b$. The 3 and 6 in this estimate arise from the use of Karatsuba-Ofman multiplication [23] for each quadratic and each cubic extension respectively. Our cost function differs from that in [24] and [15], which is $3^a 5^b$, because these assume that the Toom-Cook multiplication [36] of two degree three polynomials is more efficient than Karatsuba-Ofman multiplication, however this is not usually the case [9]. Hence the cost of an $\mathbb{F}_{q^k}$ multiplication for the given extension degrees is $18m, 54m, 108m, 162m$ respectively.

The cost of squaring a generic element of $\mathbb{F}_{q^k}$ is more complicated, since there are several squaring techniques and one needs to determine which is faster for a given application. Using the observation in [34], one can deduce that $S_k = 2M_{k/2} = 2 \cdot 3^{a-1} 6^b$. In addition, the results due to Chung and Hasan [8] give three alternative formulae for squaring using the final degree three polynomial at a cost of $3M_{k/3} + 2S_{k/3}$, $2M_{k/3} + 3S_{k/3}$ and $M_{k/3} + 4S_{k/3}$. For simplicity we use the second of the Chung-Hasan formulae, which incidentally for the above extension degrees requires exactly the same number of $\mathbb{F}_q$-multiplications as when using [34].

Table 1 contains counts of the number of $\mathbb{F}_q$-multiplications and $\mathbb{F}_q$-squarings that are required to perform a squaring in $\mathbb{F}_{q^k}$ and $G_{\Phi_k(q)}$, via the methods arising from Weil restriction to the quadratic, cubic and $\mathbb{F}_q$ subfields respectively.

As is clear from the table, with the present result we have reduced the squaring cost for generic elements in each of these fields by a factor of two for every

Table 1. Operation counts for squaring in various Weil restrictions of $G_{\Phi_k(q)}$ for $6 \mid k$

k	$\mathbb{F}_{q^k}$	$\mathrm{Res}_{\mathbb{F}_{q^k}/\mathbb{F}_{q^{k/2}}} G_{\Phi_2(q^{k/2})}$ (Stam-Lenstra [34])	$\mathrm{Res}_{\mathbb{F}_{q^k}/\mathbb{F}_{q^{k/3}}} G_{\Phi_6(q^{k/6})}$ (Present result)	$\mathrm{Res}_{\mathbb{F}_{q^k}/\mathbb{F}_q} G_{\Phi_6(q^{k/6})}$ (Granger *et al.* [15])
6	$12m$	$2S_3 = 4m + 6s$	$3S_2 = 6m$	$3m + 6s$
12	$36m$	$2S_6 = 24m$	$3S_4 = 18m$	$18m + 12s$
18	$72m$	$2S_9 = 24m + 30s$	$3S_6 = 36m$	
24	$108m$	$2S_{12} = 72m$	$3S_8 = 54m$	$84m + 24s$

degree, which greatly improves the speed of an exponentiation. If we assume for the moment also that $m \approx s$, then our squaring takes approximately 2/3-rds the time of the Stam-Lenstra result in the third column. In comparison with the final column, one sees that we beat this comprehensively; indeed for $k = 24$ the result from [15] is worse than when using $\mathrm{Res}_{\mathbb{F}_{q^k}/\mathbb{F}_{q^{k/2}}} G_{\Phi_2(q^{k/2})}$, and is barely better than Karatsuba-Ofman. Hence restricting to the cubic subfield is clearly the most efficient for fields of this form.

Remark 1. Note that we have not included the Stam-Lenstra squaring cost for $k = 6$ because this requires $q \equiv 2$ or $5 \pmod 9$ whereas the use of the sextic twist requires $D = 3$ and hence $p \equiv q \equiv 1 \pmod 3$, thus making them less desirable for pairings. An open problem posed in [15] asked for a generalisation of the Stam-Lenstra result to cyclotomic fields of degree different from six, for pairings. We have shown that our formula for squaring in the cyclotomic subgroup of $\mathbb{F}_{q^6}^{\times}$ when $q \equiv 1 \pmod 6$ matches the extremely efficient degree six squaring of [34] (while also permitting the use of sextic twists), and extends efficiently to higher degree extensions. Hence in the sense that we have provided an efficient tailor-made solution for pairings, we believe we have answered this question affirmatively.

4.2 Applicability of Method to Higher Powerings

As we have shown, all of the techniques to date for producing faster arithmetic in the cyclotomic subgroup result from an application of the Weil restriction of scalars of the equation defining membership of this group. A natural question to ask is whether this will work for extensions of any other degree? The answer is that it does, but that it appears very unlikely to provide a faster alternative to squaring.

Let $\delta(k)$ be the degree of the equation $\alpha^{\Phi_k(q)} = 1$, once expanded and the linear Frobenius operation has been incorporated. If $\delta = 2$, then the variety resulting from the Weil restriction down to any intermediate subfield may help with squaring. If $\delta > 2$ then the resulting equations may help when raising an element of the cyclotomic subgroup to the δ-th power [34]. However this is unlikely to be faster than sequential squaring for an exponentiation, even when squaring is slow. For example, for $G_{\Phi_3(q)}$, one finds that $\delta = 3$ and the resulting equations aid cubing. However the ratio of the cost of a cubing to a squaring is $> \log_2 3$, and thus it better to square than cube during an exponentiation in this case.

Complementary to this is the fact that $\delta \leq 2$ only for extensions of degree $k = 2^a 3^b$ for $a \geq 1, b \geq 0$. Hence pairings with embedding degrees of this form are ideally suited to exploit our, and the Stam-Lenstra fast squaring technique.

5 Application to Pairing-Based Cryptography

In this section we apply our squaring formula to extension field arithmetic required in the final exponentiation of a pairing computation, and post-pairing exponentations, for two concrete examples. We here assume that $\mathbb{F}_q = \mathbb{F}_p$.

The use of our formulae is possible because for any pairing, the codomain is a subgroup of $G_{\Phi_k(p)} \subset \mathbb{F}_{p^k}^\times$ where k is the embedding degree of the curve. For instance, the Tate pairing on an elliptic curve has the following form: for r coprime to p we have

$$e_r : E(\mathbb{F}_p)[r] \times E(\mathbb{F}_{p^k})/rE(\mathbb{F}_{p^k}) \rightarrow \mathbb{F}_{p^k}^\times/(\mathbb{F}_{p^k}^\times)^r.$$

In order to obtain a unique coset representative the output is usually powered by $(p^k - 1)/r$. Since

$$(p^k - 1)/r = (p^k - 1)/\Phi_k(p) \cdot \Phi_k(p)/r,$$

the first term $(p^k - 1)/\Phi_k(p)$ can be computed easily using the Frobenius and a few multiplications and a division, while the remaining 'hard' part must be computed as a proper exponentiation. Since for any element $\alpha \in \mathbb{F}_{p^k}^\times$, we have $\alpha^{(p^k-1)/\Phi_k(p)} \in G_{\Phi_k(p)}$, fast arithmetic for this group can be used.

5.1 MNT Curves

MNT curves were discovered in 2001 by Miyaji *et al.* and consist of three families of ordinary elliptic curves with embedding degrees 3, 4 and 6 [28]. For efficiency reasons, of most interest are the latter, for which the parametrisation of the base field, group cardinality and trace of Frobenius are given by:

$$\begin{aligned} p(x) &= x^2 + 1 \\ r(x) &= x^2 - x + 1 \\ t(x) &= x + 1 \end{aligned}$$

Using the method of Scott *et al.* [32] the final exponentiation reduces to the powering of an element of G_{p^2-p+1} by x. The maximum twist available has degree two and so for efficiency one would like to use $\mathbb{F}_{p^3}$ arithmetic with a quadratic extension of this field to give $\mathbb{F}_{p^6}$. This implies that one should use the composite construction of §3.2. One can alternatively use a quadratic extension of a cubic extension for the Miller loop computation, and then switch to the isomorphic tower construction of §3.2 for the final exponentiation. This isomorphism is just a permutation of basis elements, and so switching between representations, even during the Miller loop, is viable, and therefore permits the use of the fast multiplication results of [9].

The one condition that must be satisfied in order for our method to apply is that $p \equiv 1 \pmod 6$ which requires $x \equiv 0 \pmod 6$, which eliminates 2/3-rds of potential MNT curves. While this is restrictive, the benefits of ensuring this condition are clear.

5.2 BN Curves

The Barreto-Naehrig family of pairing-friendly curves were reported in 2005 and have embedding degree 12 [4]. The parametrisation of the base field, group cardinality and trace of Frobenius are given by:

$$\begin{aligned}p(x) &= 36x^4 + 36x^3 + 24x^2 + 6x + 1\\ r(x) &= 36x^4 + 36x^3 + 18x^2 + 6x + 1\\ t(x) &= 6x^2 + 1\end{aligned}$$

Note that for odd x, $\mathbb{F}_{p^{12}}$ is not pairing-friendly. However, choosing $p \equiv 3 \pmod 4$ enables the use of the initial extension $\mathbb{F}_{p^2} = \mathbb{F}_p[x]/(x^2+1)$ which permits highly efficient arithmetic. Since BN curves possess a sextic twist, such efficient subfield arithmetic is very desirable. With this choice of extension, $\mathbb{F}_{q^6}$ is towering-friendly with $\mathbb{F}_q = \mathbb{F}_{p^2}$ since $p^2 \equiv 1 \pmod 6$ for all primes $p > 3$, and is also squaring-friendly.

Again the efficient final exponentiation of Scott *et al.* [32] can be applied to reduce the final powering to essentially just three exponentiations by x. In practise, it is recommended that x should be chosen to have as low a Hamming weight as possible, to minimise the resulting cost of the Miller loop [10]. Hence for the final exponentiation, the entries in Table 1 imply that this cost will be $\approx 75\%$ the cost of the previous fastest. Indeed using our squaring method with the degree six extension of $\mathbb{F}_{p^2}$ given by the tower in §3.2 - i.e., a tower having extensions of $\mathbb{F}_p$ of degrees $1-2-4-12$ - a simple estimate of the cost of performing the final powering with Scott *et al.*'s method for a 256-bit prime, i.e., at AES 128-bit security, is 4856 $\mathbb{F}_p$-multiplications. In contrast using the tower extension with degrees $1-2-6-12$ and using the Stam-Lenstra result of §3.1 for the final extension, this figure is 5971 $\mathbb{F}_p$-multiplications, so our method should be approximately 20% faster in practise. Furthermore by excluding $p \equiv 3 \pmod 4$, the arithmetic for PFFs would be even slower for both towers.

With regard to post-pairing exponentiation, one is free to use the method of [13] which uses a clever application of the GLV decomposition [14]. For BN curves one obtains a four-dimensional decomposition and hence uses quadruple exponentiation to achieve this speed-up. Since there will be more multiplications than for the final powering the impact of our squaring formulae on the cost of exponentiation will be less pronounced, but still significant.

Another factor to consider for post-pairing exponentiations is that the trace-based methods of LUC [33], XTR [26,35] and XTR over extension fields [27], are known to be faster than [34] and [16] for a single exponentiation. However we expect the Galbrith-Scott method to be superior since the resulting exponents in the quadruple exponentiation are one quarter the size, and for the trace-based methods efficient algorithms appear to be known only for single and double exponentiation [13], ruling out their application in this context. The same reasoning applies to schemes that require a product of pairings each with individual post-pairing exponentiation, such as [7].

The trace methods are also ruled out of Scott *et al.*'s final-powering method since many multiplications are required, whereas the trace methods only permit exponentiation. Hence, barring any improvements in trace-based multi-exponentiation algorithms, we expect our formulae to feature in the most efficient way to implement pairings, their products and exponentiation.

6 Application to Torus-Based Cryptography

Our central result may also be applied to torus-based cryptography (TBC), which is based on the mathematics of algebraic tori, which were introduced to cryptography by Rubin and Silverberg in 2003 [30]. While for degree six extensions of prime fields, our squaring formulae only match the fastest implementation of CEILIDH [16] - which uses [34] - for nearly all sixth degree extensions of non-prime fields our squaring method is the most efficient known.

The implementation of $T_{30}(\mathbb{F}_p) = G_{\Phi_{30}(p)}$ by van Dijk *et al.* used the Stam-Lenstra result for $p \equiv 2$ or $5 \pmod 9$ [11]. This condition implies that $q = p^k \equiv 2$ or $5 \pmod 9$ whenever $k = 5^m$. Hence the family of fields of extension degree $6 \cdot 5^m$ over $\mathbb{F}_q$ for $q \equiv 2$ or $5 \pmod 9$ matches our squaring efficiency for the cyclotomic subgroup. On the other hand, the condition $q \equiv 1 \pmod 6$ for SFFs is far less restrictive and in fact can be said to apply to 3/4's of all finite fields.

With regard to compression of torus elements, which is the central function of torus-based cryptography, let $p \equiv 1 \pmod 6$ and let the field construction for $\mathbb{F}_{q^6}$ be the compositum given in §3.2. We reorder the basis as so:

$$\alpha = (a_0 + a_1 x + a_2 x^2) + (b_0 + b_1 x + b_2 x^2) y = a + by.$$

Assuming $\alpha \in G_{q^2-q+1}$, then as in [17] and explicitly in [29], a straightforward analysis of condition (1) yields that such elements - excepting the identity - can be represented by two elements of $\mathbb{F}_q$. To compress, one writes $\alpha \neq 1$ as

$$\alpha = a + by = \frac{c - y}{c + y},$$

where $c = -(a+1)/b$ for $b \neq 0$ and $c = 0$ if $b = 0$. Condition (1) now becomes

$$\left(\frac{c-y}{c+y}\right)^{q^2-q+1} = 1,$$

and leads to the equation $3c_0^2 + i - 3ic_1c_2 = 0$, where $c = c_0 + c_1 x + c_2 x^2$. Therefore there is redundancy between the c_i's. One can eliminate c_2 for instance which can be recovered from c_0 and c_1. The decompression map is just the inverse of this:

$$\psi : \mathbb{A}^2(\mathbb{F}_q) \to T_6(\mathbb{F}_q) \setminus \{1\} : (c_0, c_1) \mapsto \frac{3ic_0c_1 + 3ic_1^2 x + (3c_0^2 + i)x^2 - 3ic_1 y}{3ic_0c_1 + 3ic_1^2 x + (3c_0^2 + i)x^2 + 3ic_1 y},$$

with the condition $c_1 \neq 0$, which therefore represents all $q^2 - q$ non-identity elements in G_{q^2-q+1}.

Since this compression method works for all fields for which $p \equiv 1 \pmod 6$, achieves the maximum known compression for any algebraic torus, and has the fastest squaring available, we propose that such fields should be considered ideal candidates TBC.

Furthermore, as stated in [13], TBC parameters can be easily generated from pairing-friendly elliptic curves. The multi-exponentiation techniques stated in §5.2 that one acquires from PBC when TBC parameters are generated in this way mean that exponentiation in $T_6(\mathbb{F}_q)$ should be extremely efficient, and indeed faster than all other known methods.

Therefore while it could be argued that the main application of TBC is to PBC - in terms of offering faster arithmetic and compression mechanisms for systems that may be used in practise - here TBC really benefits from PBC, thus demonstrating a neat symbiosis between the two application areas.

7 Conclusion

We have presented a method to perform squaring extremely efficiently in the cyclotomic subgroup of $\mathbb{F}_{q^6}^{\times}$, for $q \equiv 1 \pmod 6$. We have shown how to apply this result to fields of interest in pairing-based cryptography to obtain the fastest final- and post-pairing exponentiation algorithms, and also detailed why these fields are ideally suited for torus-based cryptography, when $p \equiv 1 \pmod 6$.

Since these fields include those listed in the IEEE's P1363.3/D1 draft standard for identity-based public-key cryptography, which use pairings over ordinary elliptic curves that permit the fastest pairing via a maximal twist, our result strongly supports their standardisation, but also demonstrates that the more general squaring-friendly fields introduced here warrant serious consideration for inclusion.

We leave it as an open problem to find similarly efficient squaring formulae for the remaining case $q \equiv -1 \bmod 6$.

Acknowledgements

The authors would like to thank Martijn Stam for answering some questions.

References

1. Bailey, D.V., Paar, C.: Optimal Extension Fields for Fast Arithmetic in Public-Key Algorithms. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 472–485. Springer, Heidelberg (1998)
2. Barreto, P., Galbraith, S.D., ÓhÉigeartaigh, C., Scott., M.: Efficient Pairing Computation on Supersingular Abelian Varieties. Designs, Codes and Cryptography 42(3), 239–271 (2007)
3. Barreto, P., Kim, H., Lynn, B., Scott, M.: Efficient Algorithms for Pairing-Based Cryptosystems. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 354–368. Springer, Heidelberg (2002)
4. Barreto, P., 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)
5. Benger, N., Scott, M.: Constructing Tower Extensions for the implementation of Pairing-Based Cryptography (Preprint)
6. Blake, I.F., Seroussi, G., Smart, N.P.: Advances in Elliptic Curves in Cryptography. Cambridge University Press, Cambridge (2005)

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. Chung, J., Hasan, M.A.: Asymmetric Squaring Formulae. In: IEEE Symposium on Computer Arithmetic, pp. 113–122 (2007)
9. Devegili, A.J., ÓhÉigeartaigh, C., Scott, M., Dahab, R.: Multiplication and Squaring on Pairing-Friendly Fields, http://eprint.iacr.org/2006/471
10. Devegili, A.J., Scott, M., Dahab, R.: Implementing Cryptographic Pairings over Barreto-Naehrig Curves. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 197–207. Springer, Heidelberg (2007)
11. van Dijk, M., Granger, R., Page, D., Rubin, K., Silverberg, A., Stam, M., Woodruff, D.: Practical cryptography in high dimensional tori. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 234–250. Springer, Heidelberg (2005)
12. Galbraith, S.D., Harrison, K., Soldera, D.: Implementing the Tate pairing. In: Fieker, C., Kohel, D.R. (eds.) ANTS 2002. LNCS, vol. 2369, pp. 324–337. Springer, Heidelberg (2002)
13. Galbraith, S.D., Scott, M.: Exponentiation in Pairing-Friendly Groups Using Homomorphisms. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 211–224. Springer, Heidelberg (2008)
14. Gallant, R., Lambert, J., Vanstone, S.: Faster Point Multiplication on Elliptic Curves with Efficient Endomorphisms. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 190–200. Springer, Heidelberg (2001)
15. Granger, R., Page, D., Smart, N.P.: High Security Pairing-Based Cryptography Revisited. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 480–494. Springer, Heidelberg (2006)
16. Granger, R., Page, D., Stam, M.: A Comparison of CEILIDH and XTR. In: Buell, D.A. (ed.) ANTS 2004. LNCS, vol. 3076, pp. 235–249. Springer, Heidelberg (2004)
17. Granger, R., Page, D., Stam, M.: On Small Characteristic Algebraic Tori in Pairing-based Cryptography. LMS Journal of Computation and Mathematics 9, 64–85 (2006)
18. Hess, F., Vercauteren, F., Smart, N.P.: The Eta Pairing Revisited. IEEE Transactions on Information Theory 52(10), 4595–4602 (2006)
19. IEEE Draft Standard for Identity-based Public-key Cryptography using Pairings, P1636.3/D1 (2008), http://grouper.ieee.org/groups/1363/IBC/material/P1363.3-D1-200805.pdf
20. IEEE Draft Standard for identity-based cryptographic techniques using pairings, P1363.3/D3 (2009), http://grouper.ieee.org/groups/1363/IBC/index.html
21. 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)
22. Kachisa, E.J., Schaefer, E.F., Scott, M.: Constructing Brezing-Weng Pairing-Friendly Elliptic Curves Using Elements in the Cyclotomic Field. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 126–135. Springer, Heidelberg (2008)
23. Karatsuba, A., Ofman, Y.: Multiplication of Many-Digital Numbers by Automatic Computers. Soviet Physics Doklady 7, 595–596 (1963)
24. Koblitz, N., Menezes, A.J.: Pairing-Based Cryptography at High Security Levels. In: Smart, N.P. (ed.) Cryptography and Coding 2005. LNCS, vol. 3796, pp. 13–36. Springer, Heidelberg (2005)

25. Lee, E., Lee, H.S., Park, C.M.: Efficient and Generalized Pairing Computation on Abelian Varieties. IEEE Transactions on Information Theory 55(4), 1793–1803 (2009)
26. Lenstra, A.K., Verheul, E.: The XTR Public Key System. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 1–19. Springer, Heidelberg (2000)
27. Lim, S., Kim, S., Yie, I., Kim, J., Lee, H.: XTR extended to GF(p^{6m}). In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 301–312. Springer, Heidelberg (2001)
28. Miyaji, A., Nakabayashi, M., Takano, S.: New explicit conditions of elliptic curve traces for FR-reduction. IEICE Trans. Fundamentals E84-A (5), 1234–1243 (2001)
29. Naehrig, M., Barreto, P.S.L.M., Schwabe, P.: On Compressible Pairings and their Computation. In: Vaudenay, S. (ed.) AFRICACRYPT 2008. LNCS, vol. 5023, pp. 371–388. Springer, Heidelberg (2008)
30. Rubin, K., Silverberg, A.: Torus-Based Cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 349–365. Springer, Heidelberg (2003)
31. Scott, M., Barreto, P.: Compressed Pairings. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 140–156. Springer, Heidelberg (2004)
32. Scott, M., Benger, N., Charlemagne, M., Perez, L.J.D., Kachisa, E.J.: On the Final Exponentiation for Calculating Pairings on Ordinary Elliptic Curves. In: Shacham, H. (ed.) Pairing 2009. LNCS, vol. 5671, pp. 78–88. Springer, Heidelberg (2009)
33. Smith, P., Skinner, C.: A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms. In: Safavi-Naini, R., Pieprzyk, J.P. (eds.) ASIACRYPT 1994. LNCS, vol. 917, pp. 357–364. Springer, Heidelberg (1995)
34. Stam, M., Lenstra, A.K.: Efficient Subgroup Exponentiation in Quadratic and Sixth Degree Extensions. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 318–332. Springer, Heidelberg (2003)
35. Stam, M., Lenstra, A.K.: Speeding Up XTR. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 125–143. Springer, Heidelberg (2001)
36. Toom, A.L.: The Complexity of a Scheme of Functional Elements realizing the Multiplication of Integers. Soviet Mathematics 4(3), 714–716 (1963)
37. Weil, A.: Adeles and algebraic groups. Progress in Mathematics, vol. 23. Birkhäuser, Boston (1982)

Faster Pairing Computations on Curves with High-Degree Twists*

Craig Costello[1], Tanja Lange[2], and Michael Naehrig[2]

[1] Information Security Institute
Queensland University of Technology, GPO Box 2434, Brisbane QLD 4001, Australia
craig.costello@qut.edu.au
[2] Department of Mathematics and Computer Science
Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, Netherlands
tanja@hyperelliptic.org, michael@cryptojedi.org

Abstract. Research on efficient pairing implementation has focussed on reducing the loop length and on using high-degree twists. Existence of twists of degree larger than 2 is a very restrictive criterion but luckily constructions for pairing-friendly elliptic curves with such twists exist. In fact, Freeman, Scott and Teske showed in their overview paper that often the best known methods of constructing pairing-friendly elliptic curves over fields of large prime characteristic produce curves that admit twists of degree $3, 4$ or 6.

A few papers have presented explicit formulas for the doubling and the addition step in Miller's algorithm, but the optimizations were all done for the Tate pairing with degree-2 twists, so the main usage of the high-degree twists remained incompatible with more efficient formulas.

In this paper we present efficient formulas for curves with twists of degree $2, 3, 4$ or 6. These formulas are significantly faster than their predecessors. We show how these faster formulas can be applied to Tate and ate pairing variants, thereby speeding up all practical suggestions for efficient pairing implementations over fields of large characteristic.

Keywords: Pairings, Miller functions, explicit formulas, Tate pairing, ate pairing, twists, Weierstrass curves.

1 Introduction

Many new protocols are based on pairings and so the construction of pairing-friendly curves and the efficiency of pairing computation has become a field of active research. The first wave of this research exhausted many tricks that can be applied inside a Miller iteration, resulting in significant computational speed ups [4,6,7,34]. The second wave of improvements focussed on constructing pairing-friendly elliptic curves [5,11,37,16,8,22,9,17,28], and this research is extended

* This work has been supported in part by a Queensland Government Smart State PhD Scholarship and in part by the European Commission through the ICT Programme under Contract ICT–2007–216646 ECRYPT II. Part of this work was done while the second author visited QUT.

and collected in [18]. The third and more recent wave of research has focussed on reducing the loop length of Miller's algorithm [35,26,3,32] to be as short as possible [42,25]. Along the way, there have been several other clever optimizations that give faster pairings in certain scenarios, including compressed pairings [36], single coordinate pairings [21], efficient methods of hashing to pairing-friendly groups [38], and techniques that achieve a faster final exponentiation [24,39].

After the introduction of projective coordinates for pairing computations in [12], very little was heard about low level optimizations. This started to become more interesting lately for alternative curve shapes such as Edwards curves, studied in [14,27,2], and curves of the form $y^2 = x^3 + c^2$, studied in [13].

All of these improvements are presented in the context of the Tate pairing on curves with even embedding degrees and using only quadratic twists, since the nature of the Tate pairing allows for a relatively simple exposition and improves efficiency through denominator elimination. At the same time, curves with larger degree twists give much more efficient pairings and choosing special curve shapes was risking this larger benefit. On top of that, Galbraith [20] studied the group orders of curves and their twists and showed that for Edwards curves only quadratic twists could be used, in the sense that the only twist which preserves the existence of a point of order 4 is a quadratic twist. This deterred further research on ate pairings and other variants for special curves. In this paper we show that it is possible to compute a small power of the ate pairing entirely on the twisted curve; so the curve can be chosen so that the *twist* of the curve admits a particular shape. We show the fields of definition for the respective coordinates. This provides a framework for converting Tate-like pairing computation formulas and operation counts to their ate-like analogues.

For BN curves [8], Akane, Nogami, and Morikawa showed in [1] that the ate pairing itself can be computed on the twisted curve. Our result covers more general curves but computes the ate pairing only up to a power. Furthermore, the idea of using twists in order to cover curves of special shapes is new. In the context of Weierstrass curves, our result gives an easy way of computing the cost of evaluating the Miller function.

For all practically useful embedding degrees, the best methods of constructing pairing-friendly curves mostly produce elliptic curves of the form $y^2 = x^3 + ax + b$ with $a = 0$ or $b = 0$ (see [18]). In this paper we consider these two cases separately to give specialized pairing formulas in both scenarios. In particular, we achieve the fastest known formulas for computing pairings on general curves with $b = 0$ in weight-$(1, 2)$ coordinates. In addition, the point doubling formulas we derive for curves of this form are currently the fastest published point doubling formulas [10] across all forms of elliptic curves. For pairings on general curves with $a = 0$, we use standard projective coordinates. The doubling step on these curves is two field multiplications faster than the previous record for such curves. Furthermore, we also consider the case of computing pairings on curves with odd embedding degrees that employ cubic twists, where we present formulas which are significantly faster than their predecessors. Lastly, we also suggest an improvement to the formulas presented in [13]. Note that for ate pairings, speed ups in the doubling and addition step save computations in fields whose sizes grow proportionately to the embedding degree. This means that applying these faster formulas to the ate pairing variants will give relative speed ups which are consistent across all embedding degrees and savings which do not suffer as the extension field arithmetic becomes more complex.

The rest of this paper is organized as follows. Section 2 provides a brief background on pairings. In Section 3, we present a modified method of computing the ate pairing where all operations involve points only on the twisted curve. This theoretical result is a key ingredient for efficient computation of the ate pairing and has applications outside the scope of this paper, e.g. for Edwards curves. We then show how Tate pairing formulas and operation counts can be easily modified to this method of computing the ate pairing. In Sections 4 and 5, we present faster formulas for pairing computations that employ quadratic, quartic or sextic twists. In Section 6, we present faster formulas for pairings on curves with odd embedding degrees divisible by 3. We compare our results with the state-of-the-art pairing formulas in Section 7.

2 Background on Pairings

Let $p > 3$ be a prime, and let E be an elliptic curve over $\mathbb{F}_q$, $\mathrm{char}(\mathbb{F}_q) = p$, with short Weierstrass equation $E : y^2 = x^3 + ax + b$ and point at infinity $\mathcal{O}$. Let $r \neq p$ be a prime divisor of $n = \#E(\mathbb{F}_q) = q + 1 - t$ and let $k > 1$ be the embedding degree of E with respect to r, i. e. k is minimal with $r \mid q^k - 1$. For the r-torsion subgroup, we have $E[r] \subseteq E(\mathbb{F}_{q^k})$. Let $\mu_r \subseteq \mathbb{F}_{q^k}^*$ be the group of r-th roots of unity. For $m \in \mathbb{Z}$ and $P \in E[r]$, let $f_{m,P}$ be a function with divisor $\mathrm{div}(f_{m,P}) = m(P) - ([m]P) - (m-1)(\mathcal{O})$. The reduced Tate pairing is defined as

$$\tau_r : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/[r]E(\mathbb{F}_{q^k}) \to \mu_r,\ (P, Q) \mapsto f_{r,P}(Q)^{\frac{q^k-1}{r}}.$$

In practice one restricts the arguments to groups of prime order r. If $r^2 \nmid n$, the most common choice is to take the groups

$$G_1 = E[r] \cap \ker(\phi_q - [1]) = E(\mathbb{F}_q)[r],\ G_2 = E[r] \cap \ker(\phi_q - [q]) \subseteq E(\mathbb{F}_{q^k}),$$

where ϕ_q is the q-power Frobenius endomorphism on E. The groups G_1 and G_2 are the eigenspaces of ϕ_q on $E[r]$ and we have $E[r] = G_1 \oplus G_2$. From now on, we consider e_r, the reduced Tate pairing restricted to $G_1 \times G_2$, i. e.

$$e_r : G_1 \times G_2 \to \mu_r,\ (P, Q) \mapsto f_{r,P}(Q)^{\frac{q^k-1}{r}}.$$

Let $T = t - 1$. Restricting the Tate pairing to $G_2 \times G_1$ leads to the ate pairing [26]

$$a_T : G_2 \times G_1 \to \mu_r,\ (Q, P) \mapsto f_{T,Q}(P)^{\frac{q^k-1}{r}}.$$

Note that the parameter r is changed to T. The group G_2 consists of points defined over $\mathbb{F}_{q^k}$. Often G_2 can be represented by a subgroup G_2' of a curve isomorphic to E over $\mathbb{F}_{q^k}$. Let $d \mid k$; an elliptic curve E' over $\mathbb{F}_{q^{k/d}}$ is called a twist of degree d of $E/\mathbb{F}_{q^{k/d}}$ if there is an isomorphism $\psi : E' \to E$ defined over $\mathbb{F}_{q^k}$, and this is the smallest extension of $\mathbb{F}_{q^{k/d}}$ over which ψ is defined. Depending on the j-invariant $j(E)$ of E, there exist twists of degree at most 6. Pairing-friendly curves with twists of degree higher than 2 arise from constructions with j-invariants $j(E) = 0$ and $j(E) = 1728$.

A twist of E is given by E': $y^2 = x^3 + a\omega^4 x + b\omega^6$ for some $\omega \in \mathbb{F}_{q^k}$. The isomorphism between E' and E is $\Psi : E' \to E : (x', y') \to (x'/\omega^2, y'/\omega^3)$ with inverse $\Psi^{-1} : E \to E' : (x, y) \to (\omega^2 x, \omega^3 y)$. Depending on $j(E)$ and ω, we obtain the possible degrees of a twist E' as summarized in Table 1. The isomorphism Ψ induces a group isomorphism $G_2' \to G_2$, where $G_2' = E'(\mathbb{F}_{q^{k/d}})[r]$. Thus, points in G_2 can be represented by their image under Ψ^{-1}. In what follows, we write P' for the point on the twist E' corresponding to a point $P \in E$, i.e. $P' = \Psi^{-1}(P)$ and $P = \Psi(P')$. The last two columns in Table 1 show the subfields of $\mathbb{F}_{q^k}$ in which the coordinates of the specific points are contained. For example $(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^k})$ means that the x-coordinate is in $\mathbb{F}_{q^{k/2}}$ and the y-coordinate is in $\mathbb{F}_{q^k}$. The last column illustrates that the coordinates of P' lie in the same fields as the coordinates of Q. The importance of this becomes evident in Section 3. Since the points in G_2' are defined over a smaller field than those in G_2, curve arithmetic is more efficient in G_2'.

Table 1. The nature of the twist isomorphisms for twists of degree d

d	$j(E)$ a, b	fields of definition for powers of ω	$Q' = (x_{Q'}, y_{Q'})$ $P = (x_P, y_P)$	$Q = \Psi(Q')$ $P' = \Psi^{-1}(P)$
2	$\notin \{0, 1728\}$ $a \neq 0$, $b \neq 0$	$\omega^2, \omega^4, \omega^6 \in \mathbb{F}_{q^{k/2}}$ $\omega^3 \in \mathbb{F}_{q^k} \setminus \mathbb{F}_{q^{k/2}}$	$(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^{k/2}})$ $(\mathbb{F}_q, \mathbb{F}_q)$	$(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^k})$ $(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^k})$
3	0 $a = 0$, $b \neq 0$	$\omega^6, \omega^3 \in \mathbb{F}_{q^{k/3}}$ $\omega^2 \in \mathbb{F}_{q^k} \setminus \mathbb{F}_{q^{k/3}}$	$(\mathbb{F}_{q^{k/3}}, \mathbb{F}_{q^{k/3}})$ $(\mathbb{F}_q, \mathbb{F}_q)$	$(\mathbb{F}_{q^k}, \mathbb{F}_{q^{k/3}})$ $(\mathbb{F}_{q^k}, \mathbb{F}_{q^{k/3}})$
4	1728 $a \neq 0$, $b = 0$	$\omega^4 \in \mathbb{F}_{q^{k/4}}$, $\omega^2 \in \mathbb{F}_{q^{k/2}}$ $\omega^3 \in \mathbb{F}_{q^k} \setminus \mathbb{F}_{q^{k/2}}$	$(\mathbb{F}_{q^{k/4}}, \mathbb{F}_{q^{k/4}})$ $(\mathbb{F}_q, \mathbb{F}_q)$	$(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^k})$ $(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^k})$
6	0 $a = 0$, $b \neq 0$	$\omega^6 \in \mathbb{F}_{q^{k/6}}$, $\omega^3 \in \mathbb{F}_{q^{k/3}}$ $\omega^2 \in \mathbb{F}_{q^{k/2}}$	$(\mathbb{F}_{q^{k/6}}, \mathbb{F}_{q^{k/6}})$ $(\mathbb{F}_q, \mathbb{F}_q)$	$(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^{k/3}})$ $(\mathbb{F}_{q^{k/2}}, \mathbb{F}_{q^{k/3}})$

Assume that E has a twist of degree d and that $d \mid k$. Let $e = k/d$, $T_e = T^e$ mod r. The twisted ate pairing is defined as

$$\eta_{T_e} : G_1 \times G_2 \to \mu_r,\ (P, Q) \mapsto f_{T_e,P}(Q)^{\frac{q^k-1}{r}}.$$

The reduced Tate and the twisted ate pairing are both defined on $G_1 \times G_2$, while the ate pairing is defined on $G_2 \times G_1$. We aim to simultaneously treat both concepts of pairings by respectively fixing R and S as the first and second arguments of either pairing. For both variants, we thus write $f_{m,R}(S)^{(q^k-1)/r}$, where m, R, S are chosen according to the desired pairing. Miller's algorithm is used to compute the pairing as follows: Let $m = (m_{l-1}, \ldots, m_1, m_0)_2$ be the binary representation of m, initialize $U = R$, $f = 1$ and compute

```
1. for $i = l - 2$ to 0 do
    (a) $f \leftarrow f^2 \cdot f_{\mathrm{DBL}(U)}(S),\ U \leftarrow [2]U,$          //doubling step (DBL)
    (b) if $m_i = 1$ then $f \leftarrow f \cdot f_{\mathrm{ADD}(U,R)}(S),$          //addition step (ADD)
        $U \leftarrow U + R.$
2. $f \leftarrow f^{(q^k-1)/r}.$
```

The function $f_{\mathrm{DBL}(U)}$ is defined as $f_{\mathrm{DBL}(U)} = l_{\mathrm{DBL}(U)}/v_{\mathrm{DBL}(U)}$, where $l_{\mathrm{DBL}(U)}$ is the function of the line tangent to E at the point U and $v_{\mathrm{DBL}(U)}$ is the function of the vertical line through $[2]U$. Analogously, the function $f_{\mathrm{ADD}(U,R)}$ is defined as $f_{\mathrm{ADD}(U,R)} = l_{\mathrm{ADD}(U,R)}/v_{\mathrm{ADD}(U,R)}$, where $l_{\mathrm{ADD}(U,R)}$ is the function of the line through the points U and R and $v_{\mathrm{ADD}(U,R)}$ is the function of the vertical line through $U+R$. If one of the inputs to the addition is given in affine representation we speak of a "mixed addition" and use the abbreviation mADD.

Step 1 in the above algorithm is called the Miller loop; it computes the function value $f_{m,R}(S)$ up to r-th powers. Step 2, the final exponentiation, determines the final pairing value.

The number of iterations of the Miller loop is equal to $l - 1$, where l is the bitlength of m. Therefore, reducing the bitlength of m reduces the number of iterations in the Miller loop which reduces the cost of the pairing computation. Several papers have proposed methods for loop shortening [30,32,43,42,25]. For example, for the twisted ate pairing one can replace T_e by any of its powers modulo r and choose the smallest of those. A good choice for the ate pairing is to use the R-ate pairing [30], which often achieves an optimal loop length of $\log(r)/\varphi(k)$, yielding an optimal pairing [42].

3 Computing the Ate Pairing Entirely on the Twisted Curve

Several authors have presented new formulas that achieve faster iterations of the Miller loop on certain curves [12,14,27,2,13]. The operation counts presented in these papers are given in the context of Tate pairing computations on curves with even embedding degrees, where all elliptic curve operations occur in the base field $\mathbb{F}_q$ and the functions in the Miller loop are evaluated at a point which has one coordinate in $\mathbb{F}_{q^{k/2}}$ and one in the full extension field $\mathbb{F}_{q^k}$. This allows for a relatively simple exposition. However, the ate pairing reverses the roles of the points involved and employs twisted curves. This means that some of the optimizations can not be applied in the same fashion. The purpose of this section is to tidy up this discussion and to show how operation counts for the Tate pairing can be easily modified to give the analogous ate pairing count.

The usual practice when computing the ate pairing $a_T(Q, P)$ of the points $P \in E(\mathbb{F}_q)$ and $Q \in E(\mathbb{F}_{q^k})$ is to map the point Q to the twisted curve using the isomorphism Ψ^{-1}, so that the point operations (doubling/addition) in the Miller loop can be performed more efficiently using the point $Q' = \Psi^{-1}(Q) \in E'(\mathbb{F}_{q^{k/d}})$, whose coordinates are defined over the smaller field $\mathbb{F}_{q^{k/d}}$. When it is time to compute the Miller line, Q' is "untwisted" back to the full extension field via $Q = \Psi(Q')$. Operation counts for the Tate pairing do not carry over directly to the ate pairing. In particular, for the Tate pairing it is the y-coordinate of the second argument that is in the full extension field $\mathbb{F}_{q^k}$, whereas one of the coordinates of the first argument in the ate pairing is in $\mathbb{F}_{q^k}$. This means that all optimizations that were based on eliminating subfield elements have to be revised.

Furthermore, pairings on special curves such as Edwards curves and the curves in [13] pose conditions on cofactors of the group order. Galbraith [20] pointed out to us that for twists of degree larger than 2, E and E' can not both simultaneously be in Edwards form. His arguments also apply to the curves in [13] with sextic twists. So far this meant that the formulas used for the point operations and the formulas derived for the Miller functions must be treated separately which usually results in a greater overall operation count.

We show that a small (≤ 6) power of the ate pairing can be computed entirely on the twisted curve, rendering the above concerns obsolete. Our pairing can make use of loop shortening techniques just like the ate pairing, but only requires one curve (the twisted curve) to have particular properties. Furthermore, Table 1 shows that most coordinates of the twisted points P' and Q' are defined over subfields. Note that the computation of a small power of pairings for efficiency reasons has been addressed in previous work, see for example [15].

Theorem 1. *Let $E/\mathbb{F}_q : y^2 = x^3+ax+b$ and let $E'/\mathbb{F}_{q^{k/d}} : y^2 = x^3+a\omega^4x+b\omega^6$, a degree-$d$ twist of E. Let Ψ be the associated twist isomorphism $\Psi : E' \to E : (x', y') \to (x'/\omega^2, y'/\omega^3)$. Let $P \in G_1$, $Q \in G_2$, and let $Q' = \Psi^{-1}(Q)$ and $P' = \Psi^{-1}(P)$. Let $a_T(Q,P)$ be the ate pairing of Q and P. Then*

$$a_T(Q,P)^{\gcd(d,6)} = a_T(Q',P')^{\gcd(d,6)},$$

where $a_T(Q',P') = f_{T,Q'}(P')^{(q^k-1)/r}$ uses the same loop parameter as $a_T(Q,P)$ on E, but takes the two twisted points Q' and P' as inputs, instead of Q and P.

Proof. Since all factors of the Miller values that lie in a proper subfield of $\mathbb{F}_{q^k}$ vanish under the final exponentiation, it suffices to show that the Miller function updates at each iteration are equal, up to a constant defined over any proper subfield of $\mathbb{F}_{q^k}$. The computation of $a_T(Q,P)$ is composed of addition and doubling steps. Consider the gradients of the lines at either the addition or doubing stage of the Miller loop respectively. We have

$$\frac{y_2'-y_1'}{x_2'-x_1'} = \frac{\omega^3(y_2-y_1)}{\omega^2(x_2-x_1)} = \omega\frac{y_2-y_1}{x_2-x_1} \text{ and } \frac{3x_1'^2+a\omega^4}{2y_1'} = \frac{\omega^4(3x_1^2+a)}{2\omega^3y_1} = \omega\frac{3x_1^2+a}{2y_1}$$

for addition and doubling. We write the update to the Miller function at the doubling step, $f_{\mathtt{DBL}(U')}(P')$, as

$$\begin{aligned}(l_{\mathtt{DBL}(U')}(P'))/(v_{\mathtt{DBL}(U')}(P')) &= (y_{U'} - y_{P'} - \lambda'(x_{U'} - x_{P'}))/(x_{P'} - x_{[2]U'}) \\ = (\omega^3y_U - \omega^3y_P - \omega\lambda(\omega^2x_U - \omega^2x_P))&/(\omega^2x_P - \omega^2x_{[2]U}) = \omega\cdot f_{\mathtt{DBL}(U)}(P),\end{aligned}$$

where λ and λ' are the gradients determined before. We also have $f_{\mathtt{ADD}(U',Q')}(P') = \omega \cdot f_{\mathtt{ADD}(U,Q)}(P)$. For twists of degree $d = 2$ and $d = 4$, observe that $\omega^2 = \omega^{\gcd(d,6)}$ is in a subfield of $\mathbb{F}_{q^k}$ and thus vanishes in the final exponentiation. Similarly, for $d = 3$ and $d = 6$, ω^3 and ω^6 are both in subfields of $\mathbb{F}_{q^k}$ so that introducing a factor of 3 and 6 respectively to the exponent of $a_T(Q', P')$ will give an identical result to the computation of the same power of $a_T(Q, P)$. □

Corollary 2. *If $a_T(Q, P)$ is bilinear and non-degenerate, then so is $a_T(Q', P')$.*

Remark 3. Note that for $d = 6$ both ω^2 and ω^3 are in proper subfields of $\mathbb{F}_{q^k}$. Thus their contributions to the denominator and numerator vanish in the final exponentiation, so there is no need to introduce a factor of 6 to the final exponent. That is, for sextic twists it is actually always the case that $a_T(Q, P) = a_T(Q', P')$. If denominator elimination is used for $d = 6$, the values differ by ω^3 which lies in a subfield. For $k = 12$ and BN curves this case was considered by Akane, Nogami, and Morikawa [1] who showed that up to constants from subfields $a_T(Q, P) = a_T(Q', P')$.

For the other cases either ω^2 or ω^3 lie in a proper subfield $\mathbb{F}_{q^e}$ of $\mathbb{F}_{q^k}$. If 4 or 9 divides $\prod_{d|k} \Phi_d(q)/(q^e - 1)$, respectively, we obtain $\omega^{(q^k-1)/r} = 1$ and thus automatically $a_T(Q, P) = a_T(Q', P')$. However, in general these conditions are not satisfied, and the extra power of 2 or 3 is needed to obtain the same result.

Computing the ate pairing as $a_T(Q', P')$ and using twists as in Table 1 implies (for $d < 6$) that the only coordinate that lies in the full extension field $\mathbb{F}_{q^k}$ belongs to the second argument; for $d = 6$ all coordinates are defined over subfields. In this sense, the field operations encountered in computing the ate pairing $a_T(Q', P')$ on E' mimic the field operations encountered in computing the Tate pairing $e_r(P, Q)$ on E. Thus, point operation and line computation formulas that work in the Tate pairing can directly be applied to the ate pairing.

Inversions in $\mathbb{F}_{q^k}$ are prohibitively expensive and so we will show for all curve types a way to eliminate denominators. Therefore, at the doubling or addition stage of a Miller iteration the update function is given by a polynomial $f = \sum_{i,j} L_{i,j} \cdot x_S^i y_S^j$, where the $L_{i,j}$ are functions solely of the intermediate point U (doubling) or of the intermediate point U and the base point R (addition). In the Tate pairing computation of $e_r(P, Q)$, the $L_{i,j}$ are functions of some multiple of the point $P \in E(\mathbb{F}_q)$ and therefore all calculations required to compute the $L_{i,j}$ are performed in the base field $\mathbb{F}_q$. Similarly, in the modified definition of the ate pairing computation of $a_T(Q', P')$, the $L_{i,j}$ are functions of some multiple of the point $Q' \in E'(\mathbb{F}_{q^e})$ and therefore all calculations required to compute the $L_{i,j}$ in this case are performed in the subfield $\mathbb{F}_{q^e}$. Thus, if the computations of the $L_{i,j}$ in an iteration of the Tate pairing require $m\mathbf{m}_1 + s\mathbf{s}_1$, where $\mathbf{m}_1$ and $\mathbf{s}_1$ denote multiplication and squaring in $\mathbb{F}_q$, then the equivalent computations in an iteration of the ate pairing will require $m\mathbf{m}_e + s\mathbf{s}_e$, where $\mathbf{m}_e$ and $\mathbf{s}_e$ denote multiplication and squaring in $\mathbb{F}_{q^e}$; a multiplication by the curve constant a costs $\mathbf{d}_a$.

For even embedding degrees (admitting quadratic, sextic or quartic twists) the function update always simplifies to $f = L_{1,0}x + L_{0,1}y + L_{0,0}$, so that we have two extra multiplications required here ($L_{1,0}$ by x and $L_{0,1}$ by y). In the Tate pairing as well as in the ate pairing each of these multiplications costs $e = k/d$ base field multiplications if field extensions are represented in a suitable way. If k is odd and divisible by three and if the curve admits a cubic twist, the function update requires more terms. For comparison, let there be h_{ADD} non-zero terms (excluding $L_{0,0}$) in the addition step and h_{DBL} in the doubling step, each of which costs $e = k/3$ base field multiplications. We summarize the situation for different twists in Table 2.

Table 2. Converting operation counts for single addition and doubling steps in the Tate pairing $e_r(P,Q)$ and ate pairing $a_T(Q',P')$

k even	DBL	ADD/ mADD
Tate: $e_r(P,Q)$	$m_1\mathbf{m}_1 + s_1\mathbf{s}_1 + 2e\mathbf{m}_1 + \mathbf{m}_k + \mathbf{s}_k$	$m_2\mathbf{m}_1 + s_2\mathbf{s}_1 + 2e\mathbf{m}_1 + \mathbf{m}_k$
Ate: $a_T(Q',P')$	$m_1\mathbf{m}_e + s_1\mathbf{s}_e + 2e\mathbf{m}_1 + \mathbf{m}_k + \mathbf{s}_k$	$m_2\mathbf{m}_e + s_2\mathbf{s}_e + 2e\mathbf{m}_1 + \mathbf{m}_k$
k odd, $3 \mid k$	DBL	ADD/ mADD
Tate: $e_r(P,Q)$	$m_1\mathbf{m}_1 + s_1\mathbf{s}_1 + h_{\mathrm{DBL}}e\mathbf{m}_1 + \mathbf{m}_k + \mathbf{s}_k$	$m_2\mathbf{m}_1 + s_2\mathbf{s}_1 + h_{\mathrm{ADD}}e\mathbf{m}_1 + \mathbf{m}_k$
Ate: $a_T(Q',P')$	$m_1\mathbf{m}_e + s_1\mathbf{s}_e + h_{\mathrm{DBL}}e\mathbf{m}_1 + \mathbf{m}_k + \mathbf{s}_k$	$m_2\mathbf{m}_e + s_2\mathbf{s}_e + h_{\mathrm{ADD}}e\mathbf{m}_1 + \mathbf{m}_k$

In what follows, whenever we omit the subscripts from the operation costs and write $\mathbf{m}$ and $\mathbf{s}$, we mean $\mathbf{m}_1$, $\mathbf{s}_1$ for Tate pairing computation and $\mathbf{m}_e$, $\mathbf{s}_e$ for ate pairing computation.

Remark 4. Note that by Theorem 1 the computation of $a_T(Q',P')^{\gcd(d,6)}$ can be done entirely on the twisted curve. This means that Edwards curves can be employed in the ate setting if we choose the original curve such that the twisted curve can be written in Edwards form.

All curves we consider in the following are defined over the prime field $\mathbb{F}_p$. We therefore restrict to the case $q = p$ from now on.

4 Pairings on $y^2 = x^3 + ax$ with Even Embedding Degrees

The only curves which admit quartic twists over $\mathbb{F}_p$ are of the form $E : y^2 = x^3 + ax$. In this section we assume that the embedding degree k is even and so by Table 1 we can use that the x-coordinates of Q (used in the Tate pairing) and of P' (used in our modified ate pairing) are defined over a subfield of $\mathbb{F}_{p^k}$. Using the naming convention introduced in Section 2, x_S is defined over a subfield of $\mathbb{F}_{p^k}$ while y_S is minimally defined over $\mathbb{F}_{p^k}$.

Curves of the form $E : y^2 = x^3 + ax$ have not received much attention, even for simple elliptic curve arithmetic, e.g. no special formulas were reported in the EFD [10] before our paper. We present new formulas for addition and doubling in a new coordinate system, which we call "weight-$(1,2)$ coordinates". The point $(X : Y : Z)$ corresponds to the affine point (x,y), where $x = X/Z$ and $y = Y/Z^2$.

The projective curve equation for these weights is $Y^2 = X^3Z + aXZ^3$. Lopez and Dahab studied such coordinates in the context of elliptic curves over binary fields but these weights have not been used in the context of curves over odd-characteristic fields.

It is quite remarkable that our doubling formulas are faster than any doubling formulas reported for elliptic curves in the EFD.

We extend the explicit formulas for curve operations to compute the doubling and the addition step on these curves. The resulting pairing computations are also significantly faster than their predecessors.

Doubling formulas. For this curve shape the affine doubling formulas to compute $(x_3, y_3) = [2]U = [2](x_1, y_1)$ simplify to $x_3 = \lambda^2 - 2x_1,\ y_3 = \lambda(x_1 - x_3) - y_1$, where $\lambda = (3x_1^2 + a)/(2y_1)$. In weight-$(1, 2)$ coordinates the doubling formulas to compute $(X_3 : Y_3 : Z_3) = [2](X_1 : Y_1 : Z_1)$ become

$$X_3 = (X_1^2 - aZ_1^2)^2,\ Y_3 = 2Y_1(X_1^2 - aZ_1^2)((X_1^2 + aZ_1^2)^2 + 4aZ_1^2X_1^2),\ Z_3 = 4Y_1^2.$$

The point doubling needs $1\mathbf{m} + 6\mathbf{s} + 1\mathbf{d}_a$ using the following sequence of operations.

$$\begin{aligned} &A = X_1^2,\ B = Y_1^2,\ C = Z_1^2,\ D = aC,\ X_3 = (A - D)^2, \\ &E = 2(A + D)^2 - X_3,\ F = ((A - D + Y_1)^2 - B - X_3),\ Y_3 = E \cdot F,\ Z_3 = 4B. \end{aligned} \tag{1}$$

These formulas are now the fastest doubling formulas reported in the EFD [10]. They are faster by 1 **s-m** tradeoff. than the previous champion, "dbl-20090311-hwcd" due to Hisil, Wong, Carter, and Dawson. Those formulas are optimized for "Doubling-oriented XXYZZR coordinates for Jacobi quartics" and need $2\mathbf{m} + 5\mathbf{s} + 1\mathbf{d}_a$, where a is some curve constant.

Line computation for doubling. In the doubling step of the pairing computation we need to compute $[2]U$ and to compute the line function at U and evaluate it at $S = (x_S, y_S)$. The affine formula for the computation of $f_{\mathrm{DBL}(U)}(S)$ is given as $\frac{\lambda(X_1/Z_1 - x_S) + y_S - Y_1/Z_1^2}{x_S - (\lambda^2 - 2X_1/Z_1)} = -\frac{2Y_1(-(3X_1^2Z_1 + aZ_1^3)\cdot x_S + (2Y_1Z_1)\cdot y_S + X_1^3 - aZ_1^2X_1)}{-(4Y_1^2Z_1)\cdot x_S + 9X_1^4Z_1 + 6aX_1^2Z_1^3 + a^2Z_1^5 - 8X_1Y_1^2}$. Since any element except for y_S is in a proper subfield of $\mathbb{F}_{p^k}$, we can omit computing the entire denominator and also the multiplication by $-Y_1$. We leave the factor of 2 to obtain an **s-m** tradeoff. The simplified line function is

$$f'_{\mathrm{DBL}(U)}(S) = -2(3X_1^2Z_1 + aZ_1^3)\cdot x_S + (4Y_1Z_1)\cdot y_S + 2(X_1^3 - aZ_1^2X_1).$$

We write $f'_{\mathrm{DBL}(U)}(S)$ as $f'_{\mathrm{DBL}(U)}(S) = L_{1,0}\cdot x_S + L_{0,1}\cdot y_S + L_{0,0}$ and compute $L_{1,0}$, $L_{0,1}$ and $L_{0,0}$ as

$$L_{1,0} = -2Z_1\cdot(3\cdot A + D),\ L_{0,1} = 2((Y_1 + Z_1)^2 - B - C),\ L_{0,0} = (X_1 + A - D)^2 - X_3 - A,$$

using the values computed in (1) at an additional cost of $1\mathbf{m} + 2\mathbf{s}$, so that the total operation count for point doubling with line computation is $2(k/d)\mathbf{m}_1 + 2\mathbf{m} + 8\mathbf{s} + 1\mathbf{d}_a$.

Addition and mixed addition. In affine coordinates, the sum $(x_3, y_3) = U + R = (x_1, y_1) + (x_2, y_2)$ is given by $x_3 = \lambda^2 - x_1 - x_2,\ y_3 = \lambda(x_1 - x_3) - y_1$, where $\lambda = (y_1 - y_2)/(x_1 - x_2)$. In weight-$(1,2)$ coordinates this becomes $(X_3 : Y_3 : Z_3) = (X_1 : Y_1 : Z_1) + (X_2 : Y_2 : Z_2)$

$$\begin{aligned} X_3 &= (Y_1Z_2^2 - Y_2Z_1^2)^2 - (X_1Z_2 + X_2Z_1)T, \\ Y_3 &= ((Y_1Z_2^2 - Y_2Z_1^2)(X_1Z_2T - X_3) - Y_1Z_2^2TU)UZ_1Z_2, \\ Z_3 &= (UZ_1Z_2)^2, \end{aligned}$$

where $T = (X_1Z_2 - X_2Z_1)^2 Z_1Z_2$ and $U = (X_1Z_2 - X_2Z_1)$. This addition can be computed in $10\mathbf{m} + 7\mathbf{s}$ using

$$\begin{aligned} &A = Z_1^2,\ B = Z_2^2,\ C = (Z_1 + Z_2)^2 - A - B,\ D = X_1 \cdot Z_2,\ E = X_2 \cdot Z_1, \\ &F = Y_1 \cdot B,\ G = Y_2 \cdot A,\ H = (D - E),\ I = 2(F - G),\ II = I^2,\ J = C \cdot H, \\ &K = 4J \cdot H,\ X_3 = 2II - (D + E) \cdot K,\ Z_3 = J^2, \\ &Y_3 = ((J + I)^2 - Z_3 - II) \cdot (D \cdot K - X_3) - F \cdot K^2,\ Z_3 = 2Z_3. \end{aligned}$$

For mixed addition, i.e. $Z_2 = 1$, the number of operations reduces to $8\mathbf{m} + 5\mathbf{s}$ omitting computation of B, C, D and F.

Line computation for addition and mixed addition. For affine points U, R, and S the line function is given by $f_{\mathrm{ADD}(U,R)}(S) = \frac{\lambda(x_2 - x_S) + y_S - y_2}{x_S - (\lambda^2 - x_1 - x_2)}$. Again, we can omit the denominator because it is entirely defined over a subfield of $\mathbb{F}_{p^k}$. In weight-$(1,2)$ coordinates the modified line function becomes $f'_{\mathrm{ADD}(U,R)}(S) = I \cdot X_2Z_2 - I \cdot x_S Z_2^2 + J \cdot y_S Z_2^2 - J \cdot Y_2$. The values $X_2Z_2, x_SZ_2^2$, and $y_SZ_2^2$ do not change during the computation and can thus be precomputed. For the Tate pairing the cost of one addition step (computation of addition and line function) therefore is $(2k/d)\mathbf{m}_1 + 12\mathbf{m} + 7\mathbf{s}$. If $d = 2$ it is possible to save $1\mathbf{m}$ by computing $I \cdot (X_2Z_2 - x_SZ_2^2)$. When computing the ate pairing, the multiplications in $I \cdot X_2Z_2 - I \cdot x_SZ_2^2 + J \cdot y_SZ_2^2 - J \cdot Y_2$ cost $1\mathbf{m}$ each, given the shape of x_S and y_S. The cost of one addition step (computation of addition and line function) in the ate pairing therefore is $14\mathbf{m} + 7\mathbf{s}$.

For mixed additions ($Z_2 = 1$) this simplifies to $f'_{\mathrm{mADD}(U,R)}(S) = I \cdot X_2 - I \cdot x_S + J \cdot y_S - J \cdot Y_2$, costing $(2k/d)\mathbf{m}_1 + 10\mathbf{m} + 5\mathbf{s}$ for both the ate and the Tate pairing for a complete mixed addition step. For $d = 2$ again $1\mathbf{m}$ can be saved in the Tate pairing.

If R is reused several times in the Tate pairing it might be worthwhile to precompute $1/Y_2$ for longterm usage. At the beginning of a pairing computation $\tilde{X}_2 = X_2/Y_2, \tilde{x}_S = x_S/Y_2$ and $\tilde{y}_S = y_S/Y_2$ are computed. Since Y_2 lies in $\mathbb{F}_p$, so $f'_{\mathrm{mADD}(U,R)}(S)$ can be replaced by

$$f'_{\mathrm{mADD}(U,R)}(S)/Y_2 = I \cdot \tilde{X}_2 - I \cdot \tilde{x}_S + J \cdot \tilde{y}_S - J$$

without changing the pairing value. Note also that Table 1 shows that $\tilde{x}_S$ and $\tilde{y}_S$ are defined over the same fields as x_S and y_S are. In this case a mixed addition step costs only $(2k/d)\mathbf{m}_1 + 9\mathbf{m} + 5\mathbf{s}$.

If instead S is reused several times in the ate pairing, similar savings are possible. It is useful to precompute $1/y'_S$ and update the function by

$$\bar{f}_{\mathrm{mADD}(U,R)}(S)/\bar{y}_S = I \cdot \bar{X}_2 - I \cdot \bar{x}_S + J\omega^3 - J \cdot \bar{Y}_2,$$

where $\bar{X}_2 = X_2/\bar{y}_S, \bar{x}_S = x_S/\bar{y}_S$ and $\bar{Y}_2 = Y_2/\bar{y}_S$, and $y_S = \bar{y}_S\omega^3$ with $\bar{y}_S \in \mathbb{F}_p$. In this case a mixed addition step costs only $(2k/d)\mathbf{m}_1 + 9\mathbf{m} + 5\mathbf{s}$.

Note that these savings are compatible with the saving for $d = 2$.

Depending on the representation of $\mathbb{F}_{p^k}$ over $\mathbb{F}_{p^{k/2}}$ and $\mathbb{F}_{p^{k/d}}$ it is possible to save operations in the other cases.

5 Pairings on $y^2 = x^3 + b$ with Even Embedding Degrees

The only curves which can have sextic twists over $\mathbb{F}_p$ are of the form $E : y^2 = x^3 + b$. In this section we assume that the embedding degree k is even and so by Table 1 we can use that the x-coordinate of Q (in $e_r(P,Q)$) and of P' (in $a_T(Q',P')$) is defined over a subfield of $\mathbb{F}_{p^k}$. Using the naming convention introduced in Section 2, x_S is defined over a subfield of $\mathbb{F}_{p^k}$ while y_S might be defined over $\mathbb{F}_{p^k}$. Note that if $d = 6$, y_S is also defined over a proper subfield, namely $\mathbb{F}_{p^{k/3}}$. For these curves we obtained the best results in standard projective coordinates where the curve equation $y^2 = x^3 + b$ becomes $Y^2Z = X^3 + bZ^3$.

When b is a square in $\mathbb{F}_p$, the curve E always has a point of order 3, otherwise such a point never exists in $E(\mathbb{F}_p)$. The former case was extensively studied in [13] in the context of the Tate pairing. The addition formulas are independent of the nature of the curve constant b and can therefore also be used for non-square b. We slightly improve these addition formulas in the second half of this section and use these formulas for all curves with $a = 0$. The first part of this section focuses on achieving faster operation counts at the Miller doubling stage on general curves of the form $E : y^2 = x^3 + b$, where we make no assumptions about the nature of the curve constant b (and consequently the order of E).

Point doubling and line computation. The affine doubling formulas differ from those in Section 4 in the definition of λ. We have $\lambda = 3x_1^2/2y_1$. In projective coordinates and after eliminating powers of X_1^3 via the curve equation, we obtain $(X_3 : Y_3 : Z_3) = [2](X_1 : Y_1 : Z_1)$ as

$$X_3 = 2X_1Y_1(Y_1^2 - 9bZ_1^2),\ Y_3 = Y_1^4 + 18bY_1^2Z_1^2 - 27b^2Z_1^4,\ Z_3 = 8Y_1^3Z_1.$$

We homogenize the affine doubling line using $x_1 = X_1/Z_1$ and $y_1 = Y_1/Z_1$ and get

$$f'_{\mathrm{DBL}(U)}(S) = 3X_1^2 \cdot x_S - 2Y_1Z_1 \cdot y_S + 3bZ_1^2 - Y_1^2.$$

We write $f'_{\mathrm{DBL}(U)}(S) = L_{1,0} \cdot x_S + L_{0,1} \cdot y_S + L_{0,0}$ and compute $L_{1,0}$, $L_{0,1}$, $L_{0,0}$ and the point $(X_3 : Y_3 : Z_3)$ using the following sequence of operations.

$$\begin{aligned}
&A = X_1^2,\ B = Y_1^2,\ C = Z_1^2,\ D = 3bC,\ E = (X_1 + Y_1)^2 - A - B,\\
&F = (Y_1 + Z_1)^2 - B - C,\ G = 3D,\ X_3 = E \cdot (B - G),\\
&Y_3 = (B + G)^2 - 12D^2,\ Z_3 = 4B \cdot F,\ L_{1,0} = 3A,\ L_{0,1} = -F,\ L_{0,0} = D - B.
\end{aligned}$$

The total count for the above sequence of operations is $2\mathbf{m}+7\mathbf{s}+1\mathbf{d}_b$ in addition to the multiplications by x_S and y_S. Note that doubling outside the context of pairings would omit the computation of A and would obtain $E = 2X_1Y_1$, needing a total of $3\mathbf{m} + 5\mathbf{s} + 1\mathbf{d}_b$. As doubling formulas they are not competitive with those in the EFD but they are almost the fastest for the doubling step in pairings, second only to $y^2 = x^3 + c^2$ in [13].

Addition, mixed addition and line computation. For the addition of points on $y^2 = x^3 + b$, we adopt the formulas obtained in [13] for curves of the form $y^2 = x^3 + c^2$. These addition and line computation formulas are independent of b being a square. The cost for an addition is $12\mathbf{m} + 2\mathbf{s}$ The addition line in [13] can be written as $f'_{\mathrm{ADD}(U,R)}(S) = (Y_1Z_2 - Y_2Z_1) \cdot X_2 - (Y_1Z_2 - Y_2Z_1) \cdot x_SZ_2 + (X_1Z_2 - X_2Z_1) \cdot y_SZ_2 - (X_1Z_2 - X_2Z_1) \cdot Y_2$. Note that the coefficients appear as subexpressions in the mixed addition of U and R, so computing $f'_{\mathrm{ADD}(U,R)}(S)$ as above costs an extra $(2k/d)\mathbf{m}_1 + 2\mathbf{m}$ for the Tate pairing and an extra $4\mathbf{m}$ the ate pairing.

If $R = (X_2 : Y_2 : 1)$, the addition $U + R$ becomes a mixed addition and costs $9\mathbf{m} + 2\mathbf{s}$. Computing the addition and the line as $f'_{\mathrm{mADD}(U,R)}(S) = (Y_1 - Y_2Z_1) \cdot X_2 - (Y_1 - Y_2Z_1) \cdot x_S + (X_1 - X_2Z_1) \cdot y_S - (X_1 - X_2Z_1) \cdot Y_2$ costs an extra $(2k/d)\mathbf{m}_1 + 2\mathbf{m}$ for both the Tate and the ate pairing.

If R or S is fixed in the mixed addition, similar comments to Section 4 apply, reducing the extra costs to only $(2k/d)\mathbf{m}_1 + \mathbf{m}$.

6 Fast Formulas for Pairing Computations with Cubic Twists

For an odd embedding degree k, the only possible non-trivial twists are cubic twists and these only exist for curves of the form $y^2 = x^3 + b$, requiring also that $3|k$. Table 1 shows that in this scenario the point $S = (x_S, y_S)$ has x_S defined over the full extension field $\mathbb{F}_{p^k}$ and y_S defined over a subfield. The formulas obtained in most publications including the previous sections use denominator elimination based on x_S being in a subfield.

In this section we present fast formulas for addition and doubling steps for $y^2 = x^3 + b$ and optimize them using the fact that y_S, y_U and x_U are in a proper subfield of $\mathbb{F}_{p^k}$, while x_S is not. Our results are significantly faster than other studies of this case, but nevertheless the cases with even embedding degree offer more advantages. For curves of the form $y^2 = x^3 + b$, Lin *et al.* [31] observed that $1/v_{\mathrm{DBL}(U)}(S)$ can be written as

$$\frac{1}{v_{\mathrm{DBL}(U)}(S)} = \frac{1}{x_S - x_{[2]U}} = \frac{x_S^2 + x_Sx_{[2]U} + x_{[2]U}^2}{(y_S - y_{[2]U})(y_S + y_{[2]U})}.$$

Since $(y_S - y_{[2]U})(y_S + y_{[2]U})$ lies in a subfield, the line function can be multiplied by $x_S^2 + x_Sx_{[2]U} + x_{[2]U}^2$, instead of dividing it by $v_{\mathrm{DBL}(U)}(S)$. Analogously, the addition step becomes $f'_{\mathrm{ADD}(U,R)}(S) = l_{\mathrm{ADD}(U,R)}(S) \cdot (x_S^2 + x_Sx_{U+R} + x_{U+R}^2)$.

Point doubling and line computation. In projective coordinates $x_U = X_1/Z_1$ and $y_U = Y_1/Z_1$, we replace $X_1^3 = Y_1^2 Z_1 - bZ_1^3$ and factor $f'_{\mathrm{DBL}(U)}(S)$ to see that $f'_{\mathrm{DBL}(U)}(S)$ equals

$$\alpha \cdot \left(X_1 Z_1 (Y_1^2 - 9bZ_1^2) \cdot x_S + (4Y_1^2 Z_1^2) \cdot x_S^2 - (6X_1^2 Y_1 Z_1) \cdot y_S + X_1^2 (Y_1^2 + 9bZ_1^2)\right),$$

where $\alpha = (18bY_1^2 Z_1^2 - 27b^2 Z_1^4 + Y_1^4 + 8Y_1^3 Z_1 \cdot y_S)/(32Y_1^5 Z_1^3) \in \mathbb{F}_{p^{k/3}}$ does not contain x_S and can be discarded. The values for X_1 and Z_1 are defined over subfields of $\mathbb{F}_{p^k}$ and we obtain more efficient formulas by computing $f''_{\mathrm{DBL}(U)}(S) = f'_{\mathrm{DBL}(U)}(S) X_1/(Z_1 \alpha)$ as

$$f''_{\mathrm{DBL}(U)}(S) = X_1^2 (Y_1^2 - 9bZ_1^2) \cdot x_S + 4X_1 Y_1^2 Z_1 \cdot x_S^2 - 6X_1^3 Y_1 \cdot y_S + (Y_1^2 - bZ_1^2)(Y_1^2 + 9bZ_1^2).$$

For cubic twists, the term $x_S^2 \in \mathbb{F}_{p^k}$ appears in the simplified doubling line function so we write $f''_{\mathrm{DBL}(U)}(S) = L_{1,0} \cdot x_S + L_{2,0} \cdot x_S^2 + L_{0,1} \cdot y_S + L_{0,0}$. We compute $(X_3 : Y_3 : Z_3) = [2](X_1 : Y_1 : Z_1)$ and the necessary $L_{i,j}$ coefficients using $6\mathbf{m} + 7\mathbf{s} + 1\mathbf{d}_b$ in addition to the multiplications by x_S, x_S^2, and y_S.

$$\begin{aligned}
&A = X_1^2,\ B = Y_1^2,\ C = Z_1^2,\ D = bC,\ E = 3D,\ F = (X_1 + Y_1)^2 - A - B,\\
&G = (Y_1 + Z_1)^2 - B - C,\ H = 3E,\ X_3 = F \cdot (B - H),\\
&Y_3 = (B + H)^2 - 3(2E)^2,\ Z_3 = 4B \cdot G,\ L_{1,0} = A \cdot (B - H),\ L_{2,0} = F \cdot G,\\
&L_{0,1} = -3A \cdot F,\ L_{0,0} = (B - D) \cdot (B + H).
\end{aligned}$$

Note that the formulas in [33] require $8\mathbf{m} + 9\mathbf{s} + 1\mathbf{d}_b$ in addition to the multiplications by $x_S, x_S^2, y_S, y_S^2, x_S y_S$, and $x_S^2 y_S$, i.e. they need 6 multiplications costing $k/3$ base field multiplications each while we only need 3 such multiplications. This means that the overall saving is $2\mathbf{m} + 2\mathbf{s} + k\mathbf{m}_1$.

Addition and line computation. For additions we break with the conventional wisdom that the line function should be given in terms of the base point. For even embedding degrees where denominator elimination does not require further adjustment, that approach is suitable and particularly helps if the base point is given in affine coordinates. For the curves in this section we show that building the line function on the resulting point $(X_3 : Y_3 : Z_3)$ gives better operation counts in spite of Z_3 not being equal to 1.

The default line function is given by $\left(\frac{(y_1 - y_2)}{(x_1 - x_2)} \cdot (x_1 - x_S) + y_S - y_1\right)/(x_3 - x_S)$. Using the above denominator elimination technique this gets transformed to

$$\left(\frac{(y_2 - y_1)}{(x_2 - x_1)} \cdot (x_1 - x_S) + y_S - y_1\right) \cdot \left(x_3^2 + x_3 x_S + x_S^2\right)/(y_3^2 - y_S^2).$$

This approach leads to a polynomial of the form $L_{2,0} \cdot x_S^2 + L_{1,0} \cdot x_S + L_{1,1} \cdot x_S y_S + L_{2,1} \cdot x_S^2 y_S + L_{0,2} \cdot y_S + L_{0,1} \cdot y_S + L_{0,0}$ which requires $(6k/3)\mathbf{m}_1$ after the computation of the coefficients $L_{i,j}$.

In the representation

$$\left(\frac{(y_1 - y_2)}{(x_1 - x_2)} \cdot (x_3 - x_S) + y_S + y_3\right) \cdot \left(x_3^2 + x_3 x_S + x_S^2\right)/(y_3^2 - y_S^2)$$

using the coordinates x_3, y_3 instead of x_1, y_1, it becomes obvious that the factor $(x_3 - x_S)(x_3^2 + x_3x_S + x_S^2) = y_3^2 - y_S^2$ appears in the left term of the numerator and that thus the whole numerator is divisible by the subfield element $y_S + y_3$. (Note the sign change on y_3 because the line goes through $(x_3, -y_3)$ by the geometric addition law on E.) This means that the line function is of the form $L_{2,0} \cdot x_S^2 + L_{1,0} \cdot x_S + L_{0,1} \cdot y_S + L_{0,0}$, requiring only $(3k/3)\mathbf{m}_1$ after the computation of the coefficients $L_{i,j}$.

We obtain in projective coordinates that $f'_{\text{ADD}(U,R)}(S)$ equals

$$\frac{(Y_1Z_2 - Y_2Z_1)Z_3(Y_3 - y_SZ_3) + (X_3^2 + X_3Z_3x_S + Z_3^2x_S^2)(X_1Z_2 - X_2Z_1)}{(Y_3 - y_SZ_3)(X_1Z_2 - X_2Z_1)Z_3}.$$

The denominator can be discarded. To compute the numerator more efficiently we observe that $Z_3 = Z_1Z_2(X_1Z_2 - X_2Z_1)^3$ so that we can divide by $(X_1Z_2 - X_2Z_1)$; furthermore we scale the function by 2 to allow an **s-m** tradeoff. This gives

$$\begin{aligned} f''_{\text{ADD}(U,R)}(S) = {} & 2Z_3^2x_S^2 + 2X_3Z_3x_S - 2Z_1Z_2(X_1Z_2 - X_2Z_1)^2(Y_1Z_2 - Y_2Z_1)Z_3y_S \\ & + 2X_3^2 + 2Z_1Z_2(X_1Z_2 - X_2Z_1)^2(Y_1Z_2 - Y_2Z_1)Y_3. \end{aligned}$$

We compute the addition and line computation using the following sequence of operations.

$$\begin{aligned} & A = X_1 \cdot Z_2,\ B = Y_1 \cdot Z_2,\ C = Z_1 \cdot Z_2,\ D = Z_1 \cdot X_2 - A,\ E = B - Z_1 \cdot Y_2, \\ & F = D^2,\ G = E^2,\ H = -D \cdot F,\ I = F \cdot A,\ J = H + C \cdot G - 2I, \\ & K = C \cdot F \cdot E;\ X_3 = -D \cdot J,\ Y_3 = E \cdot (I - J) - (H \cdot B),\ Z_3 = C \cdot H, \\ & L = X_3^2,\ M = Z_3^2,\ N = (X_3 + Z_3)^2 - L - M,\ L_{2,0} = 2M,\ L_{1,0} = N, \\ & L_{0,0} = 2(L + K \cdot Y_3),\ L_{0,1} = -2K \cdot Z_3. \end{aligned}$$

The explicit formulas for computing $(X_3 : Y_3 : Z_3)$ are the same as in the EFD [10]; they use $12\mathbf{m} + 2\mathbf{s}$ and use the intermediate variables $A, \ldots, J$; the values $K, \ldots, N$ are used in the computation of the line function. The total operation count for the above sequence of operations is $16\mathbf{m} + 5\mathbf{s}$ in addition to the multiplications by x_S^2, x_S, and y_S. Mixed addition is cheaper saving one $\mathbf{m}$ in each of A, B, and C and needing only $13\mathbf{m} + 5\mathbf{s}$.

In the pairing computation each addition is followed by a doubling. Thus $L = X_3^2$ and $M = Z_3^2$ should be cached and used in the doubling computation. This reuse reduces the effective costs of the addition step by $2\mathbf{s}$ and similarly for the mixed-addition step. Accordingly we report $16\mathbf{m} + 3\mathbf{s}$ and $13\mathbf{m} + 3\mathbf{s}$ in the comparison in Section 7.

7 Comparisons

This section compares the speed of our pairing formulas with the literature in the following categories:

- **(i)** Curves of the form $y^2 = x^3 + ax$ have twists of degree $d = 2$ and 4. We compare operation counts with the results given by Ionica and Joux [27] and Arène *et al.* [2]; note that those papers cover general Weierstrass curves but we are not aware of any other study covering this case.
- **(ii)** Curves of the form $y^2 = x^3 + c^2$ have a point of order 3 and admit twists of degrees $d = 2$ and 6. These curves were studied in detail very recently by Costello *et al.* in [13] and we only found faster mixed addition formulas than those originally proposed.
- **(iii)** Curves of the form $y^2 = x^3 + b$ do not necessarily have a point of order 3. We study operation counts for twists of degree 2 and 6. These curves cover in particular BN curves [8]. We compare our new formulas with those given for the same curve shape in [2].
- **(iv)** Curves of the form $y^2 = x^3 + b$ also have twists of degree 3. This case requires very different optimizations and has not been studied much in the literature. The first paper studying pairing computation on curves admitting cubic twists [31] did not pay close attention to the operation count itself, so we compare our formulas with the results presented by El Mrabet *et al.* in [33], although that paper did not present addition formulas.

The above papers for even d give $k\mathbf{m}_1$ for evaluating the line function. This can almost always be done in $(2k/d)\mathbf{m}_1$, so we adjust their results accordingly. In the general addition case, Table 3 only gives counts for the Tate pairing. For $d = 2$ it is possible to save $1\mathbf{m}$ in each ADD and each mADD. For the ate pairing in this case the costs are different and the operation counts should be modified by $-(2k/d)\mathbf{m}_1 + 2\mathbf{m}$.

For mixed additions we use our improved precomputations, assuming that one of the input points is fixed.

Table 3. Comparisons of our pairing formulas with the previous fastest formulas

Curve Curve order Twist deg.	Best Coord.	DBL ADD mADD	Prev. best Coord.	DBL ADD mADD
$y^2 = x^3 + ax$ - $d = 2, 4$	Sec. 4 $\mathcal{W}_{(1,2)}$	$(2k/d)\mathbf{m}_1 + 2\mathbf{m} + 8\mathbf{s} + 1\mathbf{d}_a$ $(2k/d)\mathbf{m}_1 + 12\mathbf{m} + 7\mathbf{s}$ $(2k/d)\mathbf{m}_1 + 9\mathbf{m} + 5\mathbf{s}$	[27], [2] $\mathcal{J}$	$(2k/d)\mathbf{m}_1 + 1\mathbf{m} + 11\mathbf{s} + 1\mathbf{d}_a$ $(2k/d)\mathbf{m}_1 + 10\mathbf{m} + 6\mathbf{s}$ $(2k/d)\mathbf{m}_1 + 7\mathbf{m} + 6\mathbf{s}$
$y^2 = x^3 + c^2$ $3 \mid \#E$ $d = 2, 6$	Sec. 5 & [13] $\mathcal{P}$	$(2k/d)\mathbf{m}_1 + 3\mathbf{m} + 5\mathbf{s}$ $(2k/d)\mathbf{m}_1 + 14\mathbf{m} + 2\mathbf{s} + 1\mathbf{d}_c$ $(2k/d)\mathbf{m}_1 + 10\mathbf{m} + 2\mathbf{s} + 1\mathbf{d}_c$	[13] $\mathcal{P}$	$(2k/d)\mathbf{m}_1 + 3\mathbf{m} + 5\mathbf{s}$ $(2k/d)\mathbf{m}_1 + 14\mathbf{m} + 2\mathbf{s} + 1\mathbf{d}_c$ $(2k/d)\mathbf{m}_1 + 11\mathbf{m} + 2\mathbf{s} + 1\mathbf{d}_c$
$y^2 = x^3 + b$ $3 \nmid \#E$ $d = 2, 6$	Sec. 5 & [13] $\mathcal{P}$	$(2k/d)\mathbf{m}_1 + 2\mathbf{m} + 7\mathbf{s} + 1\mathbf{d}_b$ $(2k/d)\mathbf{m}_1 + 14\mathbf{m} + 2\mathbf{s}$ $(2k/d)\mathbf{m}_1 + 10\mathbf{m} + 2\mathbf{s}$	[2] $\mathcal{J}$	$(2k/d)\mathbf{m}_1 + 3\mathbf{m} + 8\mathbf{s}$ $(2k/d)\mathbf{m}_1 + 10\mathbf{m} + 6\mathbf{s}$ $(2k/d)\mathbf{m}_1 + 7\mathbf{m} + 6\mathbf{s}$
$y^2 = x^3 + b$ - $d = 3$	Sec. 6 $\mathcal{P}$	$k\mathbf{m}_1 + 6\mathbf{m} + 7\mathbf{s} + 1\mathbf{d}_b$ $k\mathbf{m}_1 + 16\mathbf{m} + 3\mathbf{s}$ $k\mathbf{m}_1 + 13\mathbf{m} + 3\mathbf{s}$	[33] $\mathcal{P}$	$2k\mathbf{m}_1 + 8\mathbf{m} + 9\mathbf{s} + 1\mathbf{d}_b$ *ADD/mADD* *not reported*

We point out that all new doublings are faster than the previous ones. In (i) and (iii) this comes at the expense of somewhat slower additions. In the Miller loop, doublings are significantly more frequent than additions so that this disadvantage is amply mitigated by the faster doublings. Note that the doublings save entire field operations, and do not just present **s**–**m** tradeoffs.

In Table 4 we determine the operation counts for both the Tate and ate pairings in a typical iteration of Miller's algorithm, based on the fastest operation counts summarized in Table 3. In optimized pairing implementations, the loop parameter is chosen to have a low Hamming weight so that only few additions are encountered throughout the loop. Thus, the operation counts presented in Table 4 are for the doubling stage of Miller's algorithm. The column titled *Tate* gives the equivalent number of total base field operations (multiplications and squarings in $\mathbb{F}_p$) for a Miller iteration, based on the fact that the first argument is $R \in E(\mathbb{F}_p)$ and the second argument is $S \in E(\mathbb{F}_{p^k})$; for the fields of the individual coordinates see Table 1. The column titled *ate* gives the equivalent number of base field operations for an iteration where the first argument is $R \in E'(\mathbb{F}_{p^e})$ and the second argument is $S \in E'(\mathbb{F}_{p^k})$. If $s = 2^i 3^j$, then we can quantify the cost of a multiplication in the field $\mathbb{F}_{p^s}$ as $3^i 5^j$ multiplications in $\mathbb{F}_p$ using Karatsuba and/or Toom-Cook multiplication, and we do the same for squarings, cf. [29] for details. To compare across operations we follow the EFD [10] and report two sets of numbers: the first ones are assuming that $1\mathbf{s} = 1\mathbf{m}$ and the second ones are assuming that $1\mathbf{s} = 0.8\mathbf{m}$. In the second case, we assume that squarings in $\mathbb{F}_{p^k}$ do not make use of special properties of the field extension. Thus we approximate the ratio of squaring to multiplication costs to be 0.8 as well. In both cases we assume multiplications by curve constants to be virtually free.

We use the optimal methods of curve construction for each embedding degree, which were originally presented in [18], to determine which categories (**(i)-(iv)**) E and E' belong to. We note that constructions 6.11-6.14 in [18] are due to [28].

Table 4. Comparison of optimal ate pairing and twisted ate pairing

k	Const. [18]	$\varphi(k)$	ρ	d	E	E'	$m_{opt} : T_e : r$ (log)	Tate : ate $\mathbf{s}=\mathbf{m}$	Tate : ate $\mathbf{s}=0.8\mathbf{m}$	$a_{m_{opt}}$ vs. η_{T_e}
4	6.4	2	2.000	4	**(i)**	**(i)**	1 : 1 : 2	30 : 30	26.6 : 26.6	Even
6	6.6	2	2.000	6	**(ii)**	**(iii)**	1 : 1 : 2	40 : 41	36 : 36.6	η_{T_e} (1.02)
8	6.10	4	1.500	4	**(i)**	**(i)**	3 : 3 : 4	68 : 88	61 : 77.8	η_{T_e} (1.3)
9	6.6	6	1.333	3	**(iv)**	**(iv)**	1 : 3 : 6	72 : 124	65.6 : 112	$a_{m_{opt}}$ (1.7)
12	6.8	4	1.000	6	**(iii)**	**(iii)**	1 : 2 : 4	103 : 121	92.6 : 107.8	$a_{m_{opt}}$ (1.7)
16	6.11	8	1.250	4	**(i)**	**(i)**	1 : 4 : 8	180 : 260	162.2 : 229.4	$a_{m_{opt}}$ (2.8)
18	6.12	6	1.333	6	**(iii)**	**(ii)**	1 : 3 : 6	165 : 196	148.6 : 176	$a_{m_{opt}}$ (2.5)
24	6.6	8	1.250	6	**(ii)**	**(iii)**	1 : 4 : 8	286 : 359	258 : 319.4	$a_{m_{opt}}$ (3.2)
27	6.6	18	1.111	3	**(iv)**	**(iv)**	1 : 9 : 18	290 : 602	263.6 : 542	$a_{m_{opt}}$ (4.4)
32	6.13	16	1.125	4	**(i)**	**(i)**	1 : 8 : 16	512 : 772	461.8 : 680.2	$a_{m_{opt}}$ (5.3)
36	6.14	12	1.167	6	**(iii)**	**(iii)**	1 : 6 : 12	471 : 597	424.6 : 531	$a_{m_{opt}}$ (4.7)
48	6.6	16	1.125	6	**(ii)**	**(iii)**	1 : 8 : 16	834 : 1069	752 : 950.2	$a_{m_{opt}}$ (6.2)

The construction of BN curves for $k = 12$ was given in [8] and construction 6.10 for $k = 8$ curves is due to [41]. For each embedding degree, we also present the loop length ratios $m_{opt} : T_e : r$, where m_{opt} is the loop parameter of the optimal ate pairing, T_e is the loop parameter of the twisted ate pairing and r is the loop parameter of the standard Tate pairing. For all construction methods shown in Table 4 there is an optimal ate pairing achieving the minimal loop length in Miller's algorithm. For the twisted ate pairing we used the shortest loop length found by considering the powers of $(t-1)^e \mod r$. In the last column, we compare the optimal ate pairing and twisted ate pairing and present a factor that approximates how many times faster the computation of the Miller loop is under the faster pairing option.

References

1. Akane, M., Nogami, Y., Morikawa, Y.: Fast ate pairing computation of embedding degree 12 using subfield-twisted elliptic curve. IEICE Transactions 92-A(2), 508–516 (2009)
2. Arene, C., Lange, T., Naehrig, M., Ritzenthaler, C.: Faster pairing computation. Cryptology ePrint Archive, Report 2009/155 (2009), http://eprint.iacr.org/
3. Barreto, P.S.L.M., Galbraith, S.D., O'hEigeartaigh, C., Scott, M.: Efficient pairing computation on supersingular abelian varieties. Des. Codes Cryptography 42(3), 239–271 (2007)
4. Barreto, P.S.L.M., Kim, H.Y., Lynn, B., Scott, M.: Efficient algorithms for pairing-based cryptosystems. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 354–368. Springer, Heidelberg (2002)
5. Barreto, P.S.L.M., Lynn, B., Scott, M.: Constructing elliptic curves with prescribed embedding degrees. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, pp. 257–267. Springer, Heidelberg (2003)
6. Barreto, P.S.L.M., Lynn, B., Scott, M.: On the selection of pairing-friendly groups. In: Matsui, M., Zuccherato, R.J. (eds.) SAC 2003. LNCS, vol. 3006, pp. 17–25. Springer, Heidelberg (2004)
7. Barreto, P.S.L.M., Lynn, B., Scott, M.: Efficient implementation of pairing-based cryptosystems. J. Cryptology 17(4), 321–334 (2004)
8. 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)
9. Benits Jr., W.D., Galbraith, S.D.: Constructing pairing-friendly elliptic curves using Gröbner basis reduction. In: Galbraith [19], pp. 336–345
10. Bernstein, D.J., Lange, T.: Explicit-formulas database, http://www.hyperelliptic.org/EFD
11. Brezing, F., Weng, A.: Elliptic curves suitable for pairing based cryptography. Des. Codes Cryptography 37(1), 133–141 (2005)
12. Chatterjee, S., Sarkar, P., Barua, R.: Efficient computation of Tate pairing in projective coordinate over general characteristic fields. In: Park, C.-s., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 168–181. Springer, Heidelberg (2005)
13. Costello, C., Hisil, H., Boyd, C., Nieto, J.M.G., Wong, K.K.-H.: Faster pairings on special Weierstrass curves. In: Shacham, Waters [40], pp. 89–101
14. Das, M.P.L., Sarkar, P.: Pairing computation on twisted Edwards form elliptic curves. In: Galbraith, Paterson (eds.) [23], pp. 192–210

15. Eisenträger, K., Lauter, K., Montgomery, P.L.: Improved Weil and Tate pairings for elliptic and hyperelliptic curves. In: Buell, D.A. (ed.) ANTS 2004. LNCS, vol. 3076, pp. 169–183. Springer, Heidelberg (2004)
16. Freeman, D.: Constructing pairing-friendly elliptic curves with embedding degree 10. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 452–465. Springer, Heidelberg (2006)
17. Freeman, D.: A generalized Brezing-Weng algorithm for constructing pairing-friendly ordinary abelian varieties. In: Galbraith, Paterson [23], pp. 146–163
18. Freeman, D., Scott, M., Teske, E.: A taxonomy of pairing-friendly elliptic curves. J. Cryptology 23(2), 224–280 (2010)
19. Galbraith, S.D. (ed.): Cryptography and Coding 2007. LNCS, vol. 4887. Springer, Heidelberg (2007)
20. Galbraith, S.D.: Twists of Edwards curves. unpublished manuscript (2009)
21. Galbraith, S.D., Lin, X.: Computing pairings using x-coordinates only. Des. Codes Cryptography 50(3), 305–324 (2009)
22. Galbraith, S.D., McKee, J.F., Valença, P.C.: Ordinary abelian varieties having small embedding degree. Finite Fields and their Applications 13, 800–814 (2007)
23. Galbraith, S.D., Paterson, K.G. (eds.): Pairing 2008. LNCS, vol. 5209. Springer, Heidelberg (2008)
24. Galbraith, S.D., Scott, M.: Exponentiation in pairing-friendly groups using homomorphisms. In: Galbraith, Paterson [23], pp. 211–224
25. Hess, F.: Pairing lattices. In: Galbraith, Paterson [23], pp. 18–38
26. Hess, F., Smart, N.P., Vercauteren, F.: The eta pairing revisited. IEEE Transactions on Information Theory 52(10), 4595–4602 (2006)
27. Ionica, S., Joux, A.: Another approach to pairing computation in Edwards coordinates. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 400–413. Springer, Heidelberg (2008), http://eprint.iacr.org/2008/292
28. Kachisa, E.J., Schaefer, E.F., Scott, M.: Constructing Brezing-Weng pairing-friendly elliptic curves using elements in the cyclotomic field. In: Galbraith, Paterson [23], pp. 126–135
29. Koblitz, N., Menezes, A.: Pairing-based cryptography at high security levels. In: Smart, N.P. (ed.) Cryptography and Coding 2005. LNCS, vol. 3796, pp. 13–36. Springer, Heidelberg (2005)
30. Lee, E., Lee, H.-S., Park, C.-M.: Efficient and generalized pairing computation on abelian varieties. Cryptology ePrint Archive, Report 2008/040 (2008), http://eprint.iacr.org/2008/040
31. Lin, X., Zhao, C., Zhang, F., Wang, Y.: Computing the ate pairing on elliptic curves with embedding degree $k = 9$. IEICE Transactions 91-A(9), 2387–2393 (2008)
32. Matsuda, S., Kanayama, N., Hess, F., Okamoto, E.: Optimised versions of the ate and twisted ate pairings. In: Galbraith [19], pp. 302–312
33. Mrabet, N.E., Guillermin, N., Ionica, S.: A study of pairing computation for elliptic curves with embedding degree 15. Cryptology ePrint Archive, Report 2009/370 (2009), http://eprint.iacr.org/
34. Scott, M.: Computing the Tate pairing. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 293–304. Springer, Heidelberg (2005)
35. Scott, M.: Faster pairings using an elliptic curve with an efficient endomorphism. In: Maitra, S., Veni Madhavan, C.E., Venkatesan, R. (eds.) INDOCRYPT 2005. LNCS, vol. 3797, pp. 258–269. Springer, Heidelberg (2005)
36. Scott, M., Barreto, P.S.L.M.: Compressed pairings. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 140–156. Springer, Heidelberg (2004)

37. Scott, M., Barreto, P.S.L.M.: Generating more MNT elliptic curves. Des. Codes Cryptography 38(2), 209–217 (2006)
38. Scott, M., Benger, N., Charlemagne, M., Dominguez Perez, L.J., Kachisa, E.J.: Fast hashing to G_2 on pairing-friendly curves. In: Shacham, Waters [40], pp. 102–113
39. Scott, M., Benger, N., Charlemagne, M., Dominguez Perez, L.J., Kachisa, E.J.: On the final exponentiation for calculating pairings on ordinary elliptic curves. In: Shacham, Waters [40], pp. 78–88
40. Shacham, H., Waters, B. (eds.): Pairing 2009. LNCS, vol. 5671. Springer, Heidelberg (2009)
41. Tanaka, S., Nakamula, K.: Constructing pairing-friendly elliptic curves using factorization of cyclotomic polynomials. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 136–145. Springer, Heidelberg (2008)
42. Vercauteren, F.: Optimal pairings. IEEE Transactions on Information Theory 56(1), 455–461 (2010)
43. Zhao, C.-A., Zhang, F., Huang, J.: A note on the ate pairing. International Journal of Information Security 7(6), 379–382 (2008)

Efficient Arithmetic on Hessian Curves

Reza R. Farashahi[1,2] and Marc Joye[3]

[1] Macquarie University, Department of Computing
Sydney, NSW 2109, Australia
reza@science.mq.edu.au
[2] Isfahan University of Technology, Department of Mathematical Sciences
P.O. Box 85145 Isfahan, Iran
[3] Technicolor, Security Competence Center
1 avenue de Belle Fontaine, 35576 Cesson-Sévigné Cedex, France
marc.joye@technicolor.com
http://www.thlab.net/~joyem/

Abstract. This paper considers a generalized form for Hessian curves. The family of generalized Hessian curves covers more isomorphism classes of elliptic curves. Over a finite field $\mathbb{F}_q$, it is shown to be equivalent to the family of elliptic curves with a torsion subgroup isomorphic to $\mathbb{Z}/3\mathbb{Z}$.

This paper provides efficient unified addition formulas for generalized Hessian curves. The formulas even feature completeness for suitably chosen parameters.

This paper also presents extremely fast addition formulas for generalized binary Hessian curves. The fastest projective addition formulas require $9\mathbf{M} + 3\mathbf{S}$, where $\mathbf{M}$ is the cost of a field multiplication and $\mathbf{S}$ is the cost of a field squaring. Moreover, very fast differential addition and doubling formulas are provided that need only $5\mathbf{M} + 4\mathbf{S}$ when the curve is chosen with small curve parameters.

Keywords: Elliptic curves, Hessian curves, cryptography.

1 Introduction

An elliptic curve E over a field $\mathbb{F}$ can be given by the *Weierstraß equation*

$$E: \ Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6 \,,$$

where the coefficients $a_1, a_2, a_3, a_4, a_6 \in \mathbb{F}$. Koblitz [26] and Miller [30] were the first to show that the group of rational points on an elliptic curve E over a finite field $\mathbb{F}_q$ can be used for the discrete logarithm problem in a public-key cryptosystem.

There are many other ways to represent elliptic curves such as Legendre equation, cubic equations, quartic equations and intersection of two quadratic surfaces [2,32,35]. Several forms of elliptic curves over finite fields with several coordinate systems have been studied to improve the efficiency and the speed of the arithmetic on the group law (mainly addition and doubling formulas) [2,4].

Some *unified* addition formulas that also work for the point doubling have been presented for several forms of elliptic curves, see e.g. [23,27,8,11,10,5]. Overviews can be found in [2,9]. Moreover, *complete* addition formulas that work for all pairs of inputs have been presented for Edwards curves over odd characteristic fields [5], and for binary Edwards curves [6].

A Hessian curve over a field $\mathbb{F}$ is defined by a symmetric cubic equation

$$X^3 + Y^3 + Z^3 = dXYZ\,,$$

where $d \in \mathbb{F}$ and $d^3 \neq 27$. The use of Hessian curves in cryptography has been studied in [13,23,33,21,22]. The Hessian addition formulas, the so-called Sylvester formulas, can also be used for point doubling after a permutation of input coordinates, providing a weak form of unification. Moreover, the same formulas can be used to double, add, and subtract points, which makes Hessian curves interesting against side-channel attacks [23].

In this paper, we consider the family of curves, referred to as *generalized Hessian curves*, over a field $\mathbb{F}$ defined by the equation

$$X^3 + Y^3 + cZ^3 = dXYZ\,,$$

where $c, d \in \mathbb{F}$, $c \neq 0$ and $d^3 \neq 27c$. Clearly, this family covers more isomorphism classes of elliptic curves than Hessian curves. Notice that the Sylvester addition formulas work for the family of *generalized Hessian.* But these formulas are not unified. From the Sylvester formulas and after suitable transformation of inputs coordinates, we present fast and efficient *unified* addition formulas for generalized Hessian curves.

Nevertheless, the unified formulas for Hessian curves are not complete. In other words, there are some exceptional cases where the formulas fail to give the output. We study the exceptional cases of the addition formulas for generalized Hessian curves. We observe that the unified formulas are *complete* for many generalized Hessian curves, i.e., the addition formulas work for all pairs of inputs. In particular, the group of $\mathbb{F}$-rational points on a generalized Hessian curve has *complete* addition formulas if and only if c is not a cube in $\mathbb{F}$. Also, the unified formulas are valid for all input points in rational subgroups $\mathcal{H}$ of generalized Hessian curves over finite fields $\mathbb{F}_q$ whenever $\gcd(\#\mathcal{H}, 3) = 1$.

For generalized binary Hessian curves, the *unified* addition formulas are the fastest known addition formulas on binary elliptic curves; for example $9\mathbf{M} + 3\mathbf{S}$ for extended projective addition, $8\mathbf{M} + 3\mathbf{S}$ for extended mixed affine-projective addition, and $5\mathbf{M} + 4\mathbf{S}$ for mixed addition and doubling, when curves are chosen with small parameters. As usual, we use $\mathbf{M}$ to denote a field multiplication and $\mathbf{S}$ to denote a field squaring. Furthermore, the addition formulas are complete for generalized Hessian curves over $\mathbb{F}_{2^n}$ when c is not a cube in $\mathbb{F}_{2^n}$. The mixed differential addition and doubling formulas are also complete.

Note. In [7], Bernstein, Kohel, and Lange define the *twisted* Hessian form. The twisted form is similar to the above form up to the order of the coordinates. Both forms present advantages. The neutral element on the twisted form is a finite point. In affine coordinates, the generalized form is fully symmetric and features a simpler inverse. See also [12, Exerc. 6.2].

2 Generalized Hessian Curves

A *Hessian curve* over a field $\mathbb{F}$ is given by the cubic equation

$$\mathrm{H}_d : \; x^3 + y^3 + 1 = dxy \,,$$

for some $d \in \mathbb{F}$ with $d^3 \neq 27$ [19]. This section considers the family of *generalized* Hessian curves which cover more isomorphism classes of elliptic curves than Hessian curves. As will be shown, this family provides efficient *unified* addition formulas. Moreover, the unified formulas are *complete* for some generalized Hessian curves, i.e., the addition formulas work for all pairs of inputs.

2.1 Definition

Definition 1. *Let c, d be elements of $\mathbb{F}$ such that $c \neq 0$ and $d^3 \neq 27c$. The generalized Hessian curve $\mathrm{H}_{c,d}$ over $\mathbb{F}$ is defined by the equation*

$$\mathrm{H}_{c,d} \; : \; x^3 + y^3 + c = dxy \; .$$

Clearly, a Hessian curve H_d is a *generalized Hessian* curve $\mathrm{H}_{c,d}$ with $c = 1$. Moreover, the *generalized Hessian* curve $\mathrm{H}_{c,d}$ over $\mathbb{F}$, via the map $(x, y) \mapsto (\widetilde{x}, \widetilde{y})$ defined by

$$\widetilde{x} = x/\zeta \quad \text{and} \quad \widetilde{y} = y/\zeta \tag{1}$$

with $\zeta^3 = c$, is isomorphic over $\overline{\mathbb{F}}$ to the Hessian curve $\mathrm{H}_{\frac{d}{\zeta}} \; : \; \widetilde{x}^3 + \widetilde{y}^3 + 1 = \frac{d}{\zeta}\widetilde{x}\widetilde{y}$. Therefore, for the j-invariant of $\mathrm{H}_{c,d}$, we have

$$j(\mathrm{H}_{c,d}) = j(\mathrm{H}_{\frac{d}{\zeta}}) = \frac{1}{c}\left(\frac{d(d^3 + 6^3 c)}{d^3 - 3^3 c}\right)^3 \; . \tag{2}$$

We see that the curve $\mathrm{H}_{c,d}$ over $\mathbb{F}$ is isomorphic to the curve $\mathrm{H}_{\frac{d}{\zeta}}$ over $\mathbb{F}$ if $\zeta \in \mathbb{F}$. In other words, a generalized Hessian curve over $\mathbb{F}$ is isomorphic over $\mathbb{F}$ to a Hessian curve if and only if c is a cube in $\mathbb{F}$.

It is easy to adapt the addition and doubling formulas for generalized Hessian curves (see e.g. [12, Formulary], a.k.a. Sylvester formulas). The sum of two (different) points (x_1, y_1), (x_2, y_2) on $\mathrm{H}_{c,d}$ is the point (x_3, y_3) given by

$$x_3 = \frac{{y_1}^2 x_2 - {y_2}^2 x_1}{x_2 y_2 - x_1 y_1} \quad \text{and} \quad y_3 = \frac{{x_1}^2 y_2 - {x_2}^2 y_1}{x_2 y_2 - x_1 y_1} \; . \tag{3}$$

The doubling of the point (x_1, y_1) on $\mathrm{H}_{c,d}$ is the point (x_3, y_3) given by

$$x_3 = \frac{y_1(c - {x_1}^3)}{{x_1}^3 - {y_1}^3} \quad \text{and} \quad y_3 = \frac{x_1(c - {y_1}^3)}{{x_1}^3 - {y_1}^3} \; . \tag{4}$$

Furthermore, the inverse of the point (x_1, y_1) on $\mathrm{H}_{c,d}$ is the point (y_1, x_1).

The projective closure of the curve $\mathrm{H}_{c,d}$ is

$$\mathbf{H}_{c,d} : \;\; X^3 + Y^3 + cZ^3 = dXYZ \;\; .$$

It has the points $(1 : -\omega : 0)$ with $\omega^3 = 1$ at infinity. The neutral element of the group of $\mathbb{F}$-rational points of $\mathbf{H}_{c,d}$ is the point at infinity $(1 : -1 : 0)$ that we denote by $\mathcal{O}$. For the point $P = (X_1 : Y_1 : Z_1)$ on $\mathbf{H}_{c,d}$, we have $-P = (Y_1 : X_1 : Z_1)$.

Point addition. Using the addition formulas [3], whenever defined, the sum of the points $(X_1 : Y_1 : Z_1)$, $(X_2 : Y_2 : Z_2)$ on $\mathbf{H}_{c,d}$ is the point $(X_3 : Y_3 : Z_3)$ with

$$X_3 = X_2Z_2{Y_1}^2 - X_1Z_1{Y_2}^2, \quad Y_3 = Y_2Z_2{X_1}^2 - Y_1Z_1{X_2}^2, \\ Z_3 = X_2Y_2{Z_1}^2 - X_1Y_1{Z_2}^2 \;\; . \quad (5)$$

The cost of point addition algorithms in [13,23,33] is 12$\mathbf{M}$. Moreover, these addition formulas can be performed in a parallel way, see [33]. In particular, one can perform the addition formulas (5) in a parallel environment using $3, 4$ or 6 processors with the cost of $4\mathbf{M}$, $3\mathbf{M}$ or $2\mathbf{M}$, respectively. To gain speedup, one can use the extended coordinates $(X : Y : Z : X^2 : Y^2 : Z^2 : 2XY : 2XZ : 2YZ)$. The addition algorithm in [22] uses this modified system of coordinates for the Hessian curves over the field $\mathbb{F}$ of characteristic $p > 3$. This algorithm requires $6\mathbf{M} + 6\mathbf{S}$.

Point doubling. The doubling of the point $(X_1 : Y_1 : Z_1)$ on $\mathbf{H}_{c,d}$ is the point $(X_3 : Y_3 : Z_3)$ given by

$$X_3 = Y_1(c{Z_1}^3 - {X_1}^3), \quad Y_3 = X_1({Y_1}^3 - c{Z_1}^3), \quad Z_3 = Z_1({X_1}^3 - {Y_1}^3) \;\; . \quad (6)$$

From the doubling algorithm in [13], we have the following algorithm that needs $6\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$, where $\mathbf{D}$ is the cost of a multiplication by the constant c:

$$A = {X_1}^2,\ B = {Y_1}^2,\ C = {Z_1}^2,\ D = X_1A,\ E = Y_1B,\ F = cZ_1C, \\ X_3 = Y_1(F - D), \quad Y_3 = X_1(E - F), \quad Z_3 = Z_1(D - E) \;\; . \quad (7)$$

Moreover, the cost of the following doubling algorithm for curves $\mathbf{H}_{c,d}$ over a field $\mathbb{F}$ of characteristic $p \neq 2$ is $7\mathbf{M} + 1\mathbf{S} + 1\mathbf{D}$:

$$A = X_1Y_1,\ B = (X_1 + Y_1)^2 - 2A,\ C = (X_1 + Y_1)(B - A), \\ D = (X_1 - Y_1)(B + A),\ E = 3C - 2dAZ_1, \\ X_3 = Y_1(E + D), \quad Y_3 = X_1(D - E), \quad Z_3 = -2Z_1D \;\; . \quad (8)$$

Also, one can perform the doubling formulas (6) with a cost of $3\mathbf{M} + 3\mathbf{C} + 1\mathbf{D}$, where $\mathbf{C}$ denotes a field cubing. Furthermore, for Hessian curves $\mathbf{H}_{1,d}$ over the field $\mathbb{F}$ of characteristic $p \neq 2$, the doubling algorithms in [21,22] use the extended coordinates which require $3\mathbf{M} + 6\mathbf{S}$.

2.2 Universality of the Model

We study the correspondence between generalized Hessian curves and elliptic curves having a torsion subgroup isomorphic to $\mathbb{Z}/3\mathbb{Z}$. In particular, we show that every elliptic curve over a finite field with a torsion subgroup isomorphic to $\mathbb{Z}/3\mathbb{Z}$ has an isomorphic generalized Hessian model.

Theorem 1. *Let E be an elliptic curve over a field $\mathbb{F}$. If the group $E(\mathbb{F})$ has a point of order 3 then E is isomorphic over $\overline{\mathbb{F}}$ to a generalized Hessian curve. Moreover, if $\mathbb{F}$ has an element ω with $\omega^2+\omega+1=0$, then the group $E(\mathbb{F})$ has a point of order 3 if and only if E is isomorphic over $\mathbb{F}$ to a generalized Hessian curve.*

Proof. We note that the elliptic curve E over $\mathbb{F}$ has a point of order 3 if and only if it has a Weierstraß model $\mathbf{E}_{a_1,a_3} : y^2z + a_1xyz + a_3yz^2 = x^3$ (see e.g. [25]). Let $\omega \in \overline{\mathbb{F}}$ with $\omega^2+\omega+1=0$. Let p be the characteristic of $\mathbb{F}$.

1. If $p \neq 3$, the elliptic curve $\mathbf{E}_{a_1,a_3}$ via the map $(x,y,z) \mapsto (X,Y,Z)$ given by

$$X = \omega a_1 x + (\omega-1)y + (2\omega+1)a_3 z,$$
$$Y = -(\omega+1)a_1x - (\omega+2)y - (2\omega+1)a_3z,\ Z = x$$

is isomorphic over $\mathbb{F}(\omega)$ to the generalized Hessian curve $\mathbf{H}_{c,d}$ with $c = {a_1}^3 - 27a_3$ and $d = 3a_1$. On the other hand, the generalized Hessian curve $\mathbf{H}_{c,d}$ is isomorphic over $\mathbb{F}(\omega)$ to the Weierstraß curve $\mathbf{E}_{a_1,a_3}$ with $a_1 = d/3$, $a_3 = (d^3-27c)/3^6$.

2. If $p = 3$, the elliptic curve $\mathbf{E}_{a_1,a_3}$ via the map $(x,y,z) \mapsto (X,Y,Z)$ given by

$$X = -{a_3}^2 z,\ Y = a_3(a_1x + y + a_3z),\ Z = -y$$

is isomorphic over $\mathbb{F}$ to the generalized Hessian curve $\mathbf{H}_{c,d}$ with $c = {a_3}^3$ and $d = {a_1}^3$. Conversely, every generalized Hessian curve $\mathbf{H}_{c,d}$ is isomorphic over $\mathbb{F}$ to the Weierstraß curve $\mathbf{E}_{a_1,a_3}$ with $a_1 = \sqrt[3]{d}$, $a_3 = \sqrt[3]{c}$. □

Remark 1. Consider the elliptic curve $\mathbf{E}_{a_1,a_3}$ defined in the proof of Theorem 1. If $p \neq 3$ and ${a_1}^3 - 27a_3$ is a cube in $\mathbb{F}$, we let $c = 1$ and $d = 3(a_1+2\delta)/(a_1-\delta)$, where $\delta^3 = {a_1}^3 - 27a_3$. Then, the map $(x,y,z) \mapsto (X,Y,Z)$ given by

$$X = (2a_1+\delta)x + 3y + 3a_3z,\ Y = -(a_1-\delta)x - 3y,\ Z = -(a_1-\delta)x - 3a_3z$$

is an isomorphism over $\mathbb{F}$ between $\mathbf{E}_{a_1,a_3}$ and $\mathbf{H}_{c,d}$.

Theorem 2. *Let E be an elliptic curve over a finite field $\mathbb{F}_q$. Then, the group $E(\mathbb{F}_q)$ has a point of order 3 if and only if E is isomorphic over $\mathbb{F}_q$ to a generalized Hessian curve.*

Proof. If $q \equiv 0, 1 \pmod 3$ then the theorem is a direct consequence of Theorem 1.

Next, we assume that $q \equiv 2 \pmod 3$. So, every element of $\mathbb{F}_q$ is a cube. If the elliptic curve E has an $\mathbb{F}_q$-rational point of order 3 then Remark 1 provides an isomorphism between E and a generalized Hessian curve. Moreover, every generalized Hessian curve $\mathbf{H}_{c,d}$ over $\mathbb{F}_q$ has the point $(-\zeta : 0 : 1)$ of order 3, where $\zeta^3 = c$ (see Section 4). □

3 Unified Addition Formulas

Let $\mathbf{H}_{c,d}$ be a generalized Hessian curve over $\mathbb{F}$. We recall that the addition formulas (5) do not work to double a point. Hereafter, we give some *unified* addition formulas for $\mathbf{H}_{c,d}$ where the doubling formulas can be derived directly from the addition formulas. The unified addition formulas make generalized Hessian curves interesting against side-channel attacks [2,9].

Let $P_1 = (X_1 : Y_1 : Z_1)$ and $P_2 = (X_2 : Y_2 : Z_2)$ be two points of $\mathbf{H}_{c,d}(\mathbb{F})$. Let also $T = (-\zeta : 0 : 1) \in \mathbf{H}_{c,d}(\overline{\mathbb{F}})$ with $\zeta^3 = c$. Letting $Q_1 = P_1 + T$ and $Q_2 = P_2 - T$, we have $Q_1 = (\zeta Y_1 : \zeta^2 Z_1 : X_1)$ and $Q_2 = (\zeta^2 Z_2 : \zeta X_2 : Y_2)$. Clearly, $P_1 + P_2 = Q_1 + Q_2$. To compute $P_1 + P_2$, we use the addition formulas (5) with inputs Q_1 and Q_2. Doing so, we see that the sum of the points $(X_1 : Y_1 : Z_1)$ and $(X_2 : Y_2 : Z_2)$ on $\mathbf{H}_{c,d}$ is the point $(X_3 : Y_3 : Z_3)$ given by

$$X_3 = cY_2Z_2{Z_1}^2 - X_1Y_1{X_2}^2, \quad Y_3 = X_2Y_2{Y_1}^2 - cX_1Z_1{Z_2}^2,$$
$$Z_3 = X_2Z_2{X_1}^2 - Y_1Z_1{Y_2}^2 \ . \quad (9)$$

These formulas work for doubling, i.e., they are unified addition formulas. We note that, by the swapping the order of the points in the addition formulas (9), one can obtain the following unified formulas:

$$X_3 = cY_1Z_1{Z_2}^2 - X_2Y_2{X_1}^2, \quad Y_3 = X_1Y_1{Y_2}^2 - cX_2Z_2{Z_1}^2,$$
$$Z_3 = X_1Z_1{X_2}^2 - Y_2Z_2{Y_1}^2 \ . \quad (10)$$

The next algorithm evaluates the addition formulas (9) with $12\mathbf{M} + 1\mathbf{D}$, where $1\mathbf{D}$ denotes the multiplication by constant c, which may be chosen small:

$$A = X_1X_2,\ B = Y_1Y_2,\ C = cZ_1Z_2,\ D = X_1Z_2,\ E = Y_1X_2,\ F = Z_1Y_2,$$
$$X_3 = CF - AE, \quad Y_3 = BE - CD, \quad Z_3 = AD - BF \ . \quad (11)$$

It turns out that a mixed addition requires $10\mathbf{M} + 1\mathbf{D}$ by setting $Z_2 = 1$. Moreover, the addition formulas (9) can be performed in a parallel way, similarly to the algorithm proposed for the addition formulas (5) in [33].

When $\mathbb{F}$ is of characteristic $p \neq 2$, one can use the modified system of coordinates presented in [22, §2.4]. Applying it to addition formulas (9), the sum of two points on $\mathbf{H}_{c,d}$ represented by $(X_1 : Y_1 : Z_1 : A_1 : B_1 : C_1 : D_1 : E_1 : F_1)$ and $(X_2 : Y_2 : Z_2 : A_2 : B_2 : C_2 : D_2 : E_2 : F_2)$ with

$$A_1 = {X_1}^2,\ B_1 = {Y_1}^2,\ C_1 = {Z_1}^2,\ D_1 = 2X_1Y_1,\ E_1 = 2X_1Z_1,\ F_1 = 2Y_1Z_1,$$
$$A_2 = {X_2}^2,\ B_2 = {Y_2}^2,\ C_2 = {Z_2}^2,\ D_2 = 2X_2Y_2,\ E_2 = 2X_2Z_2,\ F_2 = 2Y_2Z_2,$$

is the point represented by $(X_3 : Y_3 : Z_3 : A_3 : B_3 : C_3 : D_3 : E_3 : F_3)$ given by

$$X_3 = cC_1F_2 - D_1A_2,\ Y_3 = B_1D_2 - cE_1C_2,\ Z_3 = A_1E_2 - F_1B_2,$$
$$A_3 = {X_3}^2,\ B_3 = {Y_3}^2,\ C_3 = {Z_3}^2,\ D_3 = (X_3 + Y_3)^2 - A_3 - B_3, \quad (12)$$
$$E_3 = (X_3 + Z_3)^2 - A_3 - C_3,\ F_3 = (Y_3 + Z_3)^2 - B_3 - C_3 \ .$$

This algorithm requires $6\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$, where $2\mathbf{D}$ represent the two multiplications by constant c, which can be chosen small. Furthermore, the mixed addition formulas can be obtained by setting $Z_2 = 1$ which need $5\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$.

4 Complete Addition Formulas

Again, we let $\mathbf{H}_{c,d}$ denote a generalized Hessian curve over $\mathbb{F}$. In this section, we study the exceptional cases of the addition formulas (5), (9) and (10). In particular, we show that addition formulas (9), (10) work for all pairs of $\mathbb{F}$-rational points on $\mathbf{H}_{c,d}$ whenever c is not a cube in $\mathbb{F}$.

We consider the set of $\overline{\mathbb{F}}$-rational points at infinity on $\mathbf{H}_{c,d}$, denoted by ∞,

$$\infty = \left\{(1 : -\omega : 0) \mid \omega \in \overline{\mathbb{F}},\ \omega^3 = 1\right\} \ .$$

We note that ∞ is a subgroup of the group of $\overline{\mathbb{F}}$-rational points on $\mathbf{H}_{c,d}$. Further, ∞ is a subgroup of the 3-torsion group $\mathbf{H}_{c,d}[3]$, where

$$\mathbf{H}_{c,d}[3] = \left\{P \mid P \in \mathbf{H}_{c,d}(\overline{\mathbb{F}}),\ 3P = \mathcal{O}\right\} \ .$$

Let $\mathcal{T}_1$, $\mathcal{T}_2$ be the set of $\overline{\mathbb{F}}$-rational points $P = (X : Y : Z)$ of $\mathbf{H}_{c,d}[3]$ with $Y = 0$, $X = 0$, respectively. Namely,

$$\mathcal{T}_1 = \left\{(-\zeta : 0 : 1) \mid \zeta \in \overline{\mathbb{F}},\ \zeta^3 = c\right\} \quad \text{and} \quad \mathcal{T}_2 = \{-P \mid P \in \mathcal{T}_1\} \ .$$

Clearly, $\mathbf{H}_{c,d}[3]$ is partitioned into $\infty \cup \mathcal{T}_1 \cup \mathcal{T}_2$.

The following proposition describes the exceptional cases of the addition formulas (5).

Proposition 1. *The addition formulas* (5) *work for all pairs of points P_1, P_2 on $\mathbf{H}_{c,d}$ if and only if $P_1 - P_2$ is not a point at infinity.*

Proof. Let $P_1 = (X_1 : Y_1 : Z_1)$ and $P_2 = (X_2 : Y_2 : Z_2)$ be points in $\mathbf{H}_{c,d}(\overline{\mathbb{F}})$.

First, assume that the addition formulas (5) do not work for the inputs P_1, P_2, i.e., we have $X_3 = Y_3 = Z_3 = 0$, where $X_3 = X_2Z_2{Y_1}^2 - X_1Z_1{Y_2}^2$, $Y_3 = Y_2Z_2{X_1}^2 - Y_1Z_1{X_2}^2$ and $Z_3 = X_2Y_2{Z_1}^2 - X_1Y_1{Z_2}^2$. We distinguish two cases to show that $P_1 - P_2 \in \infty$.

1. If $Z_1 = 0$ then $Z_3 = -X_1Y_1{Z_2}^2$. We see that $X_1Y_1 \neq 0$, since $P_1 \in \mathbf{H}_{c,d}$. So, $Z_2 = 0$. That means P_1, P_2 are in ∞. Therefore, $P_1 - P_2$ is a point at infinity.
2. Assume now that $Z_1 \neq 0$ and $Z_2 \neq 0$. We write $P_1 = (x_1 : y_1 : 1)$ and $P_2 = (x_2 : y_2 : 1)$, where $x_i = X_i/Z_i$ and $y_i = Y_i/Z_i$ $(i = 1, 2)$. From $X_3 = Y_3 = Z_3 = 0$, we have $x_2{y_1}^2 = x_1{y_2}^2$, $y_2{x_1}^2 = y_1{x_2}^2$ and $x_1y_1 = x_2y_2$. So, $y_1y_2({x_1}^3 - {x_2}^3) = 0$ and $x_1x_2({y_1}^3 - {y_2}^3) = 0$. Moreover, from the equation of $\mathrm{H}_{c,d}$, we have ${x_1}^3 + {y_1}^3 = {x_2}^3 + {y_2}^3$.

   If $x_1x_2 \neq 0$ then ${y_1}^3 = {y_2}^3$. Next, we assume that $x_1x_2 = 0$. If $x_1 = 0$ then $y_1 \neq 0$. From $X_3 = 0$, we remark that $x_2 = 0$. Then, $x_1 = x_2 = 0$

implies that ${y_1}^3 = {y_2}^3$. Therefore, in all cases, we obtain ${y_1}^3 = {y_2}^3$ and ${x_1}^3 = {x_2}^3$. So, we can write $y_2 = \omega_1 y_1$ and $x_2 = \omega_2 x_1$, where ω_1, ω_2 are third roots of unity. The condition $x_1 y_1 = x_2 y_2$ becomes $(\omega_1\omega_2 - 1)x_1 y_1 = 0$. If $x_1 y_1 \neq 0$ then $\omega_2 = {\omega_1}^{-1}$ and thus $P_1 - P_2 = (1 : -\omega_1 : 0)$. If $x_1 = 0$ then $x_2 = 0$ and $P_1 - P_2 = (1 : -\omega_1 : 0)$. Finally, if $y_1 = 0$ then $y_2 = 0$ and $P_1 - P_2 = (\omega_2 : -1 : 0)$. Summing up, we always have $P_1 - P_2 \in \infty$.

Now, we study the other direction. We assume that $P_1 - P_2 \in \infty$ where $P_1, P_2 \in \mathbf{H}_{c,d}(\overline{\mathbb{F}})$. Then $P_1 = P_2 + (1 : -\omega : 0) = (\omega X_2 : \omega^{-1} Y_2 : Z_2)$, where ω is a third root of unity. It is easily seen that the addition formulas (5) do not work for such P_1, P_2. □

We note that the addition formulas (5) work for all *distinct* pairs of $\mathbb{F}$-rational inputs if the curve $\mathbf{H}_{c,d}$ over $\mathbb{F}$ has only one $\mathbb{F}$-rational point at infinity, i.e., if $\mathbb{F}$ has only one third root of unity. This happens for Hessian curves $\mathbf{H}_{c,d}$ over $\mathbb{F}_q$ with $q \not\equiv 1 \pmod 3$ and, in particular, for binary curves $\mathbf{H}_{c,d}$ over $\mathbb{F}_{2^n}$ with odd integers n.

Proposition 2. *The addition formulas* (9) *work for all pairs of points P_1, P_2 on $\mathbf{H}_{c,d}$ if and only if $P_1 - P_2 \notin \mathcal{T}_1$.*

Proof. Let P_1, P_2 be points on $\mathbf{H}_{c,d}$. Let T_1 be a point of $\mathcal{T}_1$. Let $Q_1 = P_1 + T_1$ and $Q_2 = P_2 - T_1$. We note that the output of formulas (9) for the pair of points P_1, P_2 is equal to the output of formulas (5) for the pair of points Q_1, Q_2. From Proposition 1, we see that the formulas (9) do not work for the pair of points P_1, P_2 if and only if $Q_1 - Q_2 \in \infty$. This is equivalent to $P_1 - P_2 \in \mathcal{T}_1$. □

Similarly, the addition formulas (10) work for all pairs of points P_1, P_2 on $\mathbf{H}_{c,d}$ with $P_1 - P_2 \notin \mathcal{T}_2$. Since the sets $\mathcal{T}_1$ and $\mathcal{T}_2$ are disjoint, if the addition formulas (9) fail to compute the sum of two points, then the addition formulas (10) work to compute this sum. Clearly, this is true for the other way round. In other words, if the addition formulas (9) do not work for the pair of inputs P_1, P_2, then they work for the pair of inputs P_2, P_1.

Corollary 1. *The doubling formulas* (6) *for the generalized Hessian curve $\mathbf{H}_{c,d}$ work for all inputs.*

Proof. The doubling formulas (6) can be obtained from the addition formulas (9) by letting $P_2 = P_1$. Then, from Proposition 2, we see that these doubling formulas work for all points on $\mathbf{H}_{c,d}$. □

Corollary 2. *Assume $\mathcal{H}$ is a subgroup of $\mathbf{H}_{c,d}(\overline{\mathbb{F}})$ which is disjoint from $\mathcal{T}_1$. Then, the addition formulas* (9) *and* (10) *work for all pairs of points in $\mathcal{H}$.*

Proof. Clearly, $\mathcal{H}$ and $\mathcal{T}_2$ are disjoint as well. Then, Proposition 2 concludes the proof. □

Here, we express the family of *complete generalized Hessian* curves. By a *complete* curve, we mean a curve with complete addition formulas, i.e., a curve over a field $\mathbb{F}$ with addition formulas that are valid for every pair of $\mathbb{F}$-rational points.

Theorem 3. *Let c, d be elements of $\mathbb{F}$ such that $d^3 \neq 27c$. Let $\mathbf{H}_{c,d}$ be the generalized Hessian curve over $\mathbb{F}$ with the addition formulas* (9)*. Then, $\mathbf{H}_{c,d}$ is complete over $\mathbb{F}$ if and only if c is not a cube in $\mathbb{F}$.*

Proof. By definition of $\mathcal{T}_1$, we see that the set of $\mathbb{F}$-rational points of $\mathcal{T}_1$ is empty if and only if c is not a cube in $\mathbb{F}$. By Proposition 2, the addition formulas (9) work for all pairs of $\mathbb{F}$-rational points if and only if the set of $\mathbb{F}$-rational points of $\mathcal{T}_1$ is empty, which completes the proof. $\square$

Below, we give two examples of generalized Hessian curves over finite fields with complete addition formulas.

Example 1. Let c, d be elements of the finite field $\mathbb{F}_q$ with $q \equiv 1 \pmod 3$ such that $d^3 \neq 27c$ and c is not a cube in $\mathbb{F}_q$. Then, the generalized Hessian curve $\mathrm{H}_{c,d}$ over $\mathbb{F}_q$ is complete with the addition formulas (9) or (10).

Example 2. Let c, d be elements of $\mathbb{F}_q$ such that $c \neq 0$ and $d^3 \neq 27c$. Let $\mathcal{H}$ be a subgroup of $\mathbf{H}_{c,d}(\mathbb{F}_q)$ with $\gcd(\#\mathcal{H}, 3) = 1$. Then, $\mathcal{H}$ is complete over $\mathbb{F}_q$ with the addition formulas (9) or (10).

5 Explicit Formulas in Characteristic 2

In this section, we present fast and efficient addition, doubling, tripling and differential addition formulas for generalized binary Hessian curves over a field $\mathbb{F}$ of characteristic $p = 2$.

5.1 Addition

We recall that the cost of point addition algorithms in [13,33] for the addition formulas (5) is $12\mathbf{M}$. Also, the addition algorithm (11) requires $12\mathbf{M} + 1\mathbf{D}$. One may choose the constant c small to reduce the cost of this algorithm to $12\mathbf{M}$. Further, the addition algorithm (11) is *unified*. Furthermore, it features *completeness* for generalized binary Hessian curve $\mathbf{H}_{c,d}$ over $\mathbb{F}_{2^n}$, where n is even and c is not a cube in $\mathbb{F}_{2^n}$.

Moreover, one can use the extended coordinates $(X : Y : Z : X^2 : Y^2 : Z^2 : XY : XZ : YZ)$. Here, the sum of two points on $\mathbf{H}_{c,d}$ represented by $(X_1 : Y_1 : Z_1 : A_1 : B_1 : C_1 : D_1 : E_1 : F_1)$ and $(X_2 : Y_2 : Z_2 : A_2 : B_2 : C_2 : D_2 : E_2 : F_2)$ where

$$A_1 = {X_1}^2,\ B_1 = {Y_1}^2,\ C_1 = {Z_1}^2,\ D_1 = X_1Y_1,\ E_1 = X_1Z_1,\ F_1 = Y_1Z_1,$$
$$A_2 = {X_2}^2,\ B_2 = {Y_2}^2,\ C_2 = {Z_2}^2,\ D_2 = X_2Y_2,\ E_2 = X_2Z_2,\ F_2 = Y_2Z_2$$

is the point represented by $(X_3 : Y_3 : Z_3 : A_3 : B_3 : C_3 : D_3 : E_3 : F_3)$ given by

$$\begin{aligned} X_3 = cC_1F_2 + D_1A_2,\ Y_3 = B_1D_2 + cE_1C_2,\ Z_3 = A_1E_2 + F_1B_2, \\ A_3 = {X_3}^2,\ B_3 = {Y_3}^2,\ C_3 = {Z_3}^2,\ D_3 = X_3Y_3,\ E_3 = X_3Z_3,\ F_3 = Y_3Z_3\ . \end{aligned} \tag{13}$$

This algorithm requires $9\mathbf{M} + 3\mathbf{S} + 2\mathbf{D}$, where the two $\mathbf{D}$ are multiplication by the constant c. We note that the algorithm (13) is obtained from the addition formulas (9), so it is *unified* and works for point doublings as well. Moreover, it works for all pairs of inputs on a *complete* curve (cf. Theorem 3). Furthermore, the mixed addition formulas need $8\mathbf{M}+3\mathbf{S}+2\mathbf{D}$ by setting $Z_2 = 1$. If c is small, then one can obtain the addition algorithm in a parallel environment using $3, 4$ or 6 processors which needs $3\mathbf{M} + 1\mathbf{S}$, $3\mathbf{M}$ or $2\mathbf{M}$, respectively.

Table 1 lists the complexities of addition formulas for different shapes of binary elliptic curves and different coordinate systems. As Table 1 shows, the generalized Hessian curves provide the fastest addition formulas for binary elliptic curves. Moreover, our formulas for Hessian curves are unified. They are even complete for many generalized Hessian curves. We note that all addition formulas for short Weierstraß curve are not even unified. But, binary Edwards curves provide unified and even complete formulas.

Table 1. Cost of addition formulas for different families of binary elliptic curves

Curve shape	Representation	Projective addition	Mixed addition
Short Weierstraß	Projective [4]	$14\mathbf{M} + 1\mathbf{S} + 1\mathbf{D}$	$11\mathbf{M} + 1\mathbf{S} + 1\mathbf{D}$
$y^2 + xy = x^3 + a_2x^2 + a_6$	Jacobian [4]	$14\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$	$10\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$
	Lopez-Dahab [1,4,20]	$13\mathbf{M} + 4\mathbf{S}$	$8\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$
	Extended Lopez-Dahab		
	with $a_2 = 0$ [1,4,20]	$14\mathbf{M} + 3\mathbf{S}$	$9\mathbf{M} + 4\mathbf{S} + 1\mathbf{D}$
	with $a_2 = 1$ [1,4,20,24]	$13\mathbf{M} + 3\mathbf{S}$	$8\mathbf{M} + 4\mathbf{S}$
Binary Edwards	Projective [6]	$18\mathbf{M} + 2\mathbf{S} + 7\mathbf{D}$	$13\mathbf{M} + 3\mathbf{S} + 3\mathbf{D}$
$d_1(x + y) + d_2(x^2 + y^2)$	Projective		
$= xy + xy(x + y) + x^2y^2$	with $d_1 = d_2$ [6]	$16\mathbf{M} + 1\mathbf{S} + 4\mathbf{D}$	$13\mathbf{M} + 3\mathbf{S} + 3\mathbf{D}$
Hessian	Projective [13,23,33]	$12\mathbf{M}$	$10\mathbf{M}$
$x^3 + y^3 + 1 = dxy$	Projective, formulas (11)	$12\mathbf{M}$	$10\mathbf{M}$
	Extended, formulas (13)	$9\mathbf{M} + 3\mathbf{S}$	$8\mathbf{M} + 3\mathbf{S}$
Generalized Hessian	Projective [12]	$12\mathbf{M}$	$10\mathbf{M}$
$x^3 + y^3 + c = dxy$	Projective, formulas (11)	$12\mathbf{M} + 1\mathbf{D}$	$10\mathbf{M} + 1\mathbf{D}$
	Extended, formulas (13)	$9\mathbf{M} + 3\mathbf{S} + 2\mathbf{D}$	$8\mathbf{M} + 3\mathbf{S} + 2\mathbf{D}$

5.2 Doubling

We recall that the doubling algorithm (7) needs $6\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$ to perform the doubling formulas (6). Furthermore, from the doubling formulas (6), we see that the doubling of the point $(X_1 : Y_1 : Z_1)$ on $\mathbf{H}_{c,d}$ is the point $(X_3 : Y_3 : Z_3)$ with

$$X_3 = {Y_1}^4 + dX_1{Y_1}^2Z_1,\ Y_3 = {X_1}^4 + d{X_1}^2Y_1Z_1,\ Z_3 = c{Z_1}^4 + dX_1Y_1{Z_1}^2\ . \quad (14)$$

The following algorithm performs the doubling formulas (14) which requires $5\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$:

$$A = {X_1}^2,\ B = {Y_1}^2,\ C = {Z_1}^2,\ D = X_1Y_1,\ G = DZ_1,\ H = dG,$$
$$X_3 = B^2 + Y_1H,\ Y_3 = A^2 + X_1H,\ Z_3 = cC^2 + Z_1H\ .$$

Moreover, the doubling of the point $(X_1 : Y_1 : Z_1)$ on a binary curve $\mathbf{H}_{c,d}$, using the representation $(X_1 : Y_1 : Z_1 : A_1 : B_1 : C_1 : D_1 : E_1 : F_1)$, where $A_1 = {X_1}^2$, $B_1 = {Y_1}^2$, $C_1 = {Z_1}^2$, $D_1 = X_1Y_1$, $E_1 = X_1Z_1$, $F_1 = Y_1Z_1$, is the point represented by $(X_3 : Y_3 : Z_3 : A_3 : B_3 : C_3 : D_3 : E_3 : F_3)$ given by

$$X_3 = B_1(B_1 + dE_1),\ Y_3 = A_1(A_1 + dF_1),\ Z_3 = (A_1 + B_1 + D_1)(E_1 + F_1),$$
$$A_3 = {X_3}^2,\ B_3 = {Y_3}^2,\ C_3 = {Z_3}^2,\ D_3 = X_3Y_3,\ E_3 = X_3Z_3,\ F_3 = Y_3Z_3\ .$$

The cost of above doubling algorithm is $6\mathbf{M} + 3\mathbf{S} + 2\mathbf{D}$. We also note that, the coordinates D_3, E_3 and F_3 can be given by

$$D_3 = {D_1}^4 + cd{E_1}^2{F_1}^2,\ E_3 = c{F_1}^4 + d{D_1}^2{E_1}^2,\ F_3 = c{E_1}^4 + d{D_1}^2{F_1}^2\ .$$

The following doubling algorithm needs less field multiplications:

$$G = {A_1}^2,\ H = {B_1}^2,\ I = {C_1}^2,\ J = D_1E_1,\ K = D_1F_1,\ L = E_1F_1,$$
$$X_3 = H + dK,\ Y_3 = G + dJ,\ Z_3 = cI + dL,\ A_3 = {X_3}^2,\ B_3 = {Y_3}^2,\ C_3 = {Z_3}^2,$$
$$R = {D_1}^2 + \sqrt{cd}L,\ S = \sqrt{c}{F_1}^2 + \sqrt{d}J,\ T = \sqrt{c}{E_1}^2 + \sqrt{d}K,$$
$$D_3 = R^2,\ E_3 = S^2,\ F_3 = T^2\ .$$

Above doubling algorithm needs $3\mathbf{M} + 12\mathbf{S} + 9\mathbf{D}$. This algorithm requires $3\mathbf{M} + 12\mathbf{S} + 6\mathbf{D}$ if c is small and $3\mathbf{M} + 12\mathbf{S} + 4\mathbf{D}$ if d is small.

Our doubling formulas slightly improve the current speed of doublings on Hessian curves. Moreover, the doubling formulas for generalized Hessian curves are faster than doubling formulas using projective coordinates in short Weierstraß form, see [2]. But, they are slower than various doubling formulas using Jacobian [2], Lopez-Dahab representations of short Weierstraß form [2,29,24,6] and projective representation of binary Edwards [6].

We note that the only *complete* doubling formulas are presented by binary Edwards [6] and generalized binary Hessian curves (see Corollary 1).

5.3 Tripling

Here, we present fast tripling formulas for generalized binary Hessian curves. The tripling formulas can be used in double based number systems, DBNS; see e.g., [14,3,15]. For a point $(X_1 : Y_1 : Z_1)$ on $\mathbf{H}_{c,d}$, we have $3(X_1 : Y_1 : Z_1) = (X_3 : Y_3 : Z_3)$ with

$$X_3 = d({Y_1}^3({Z_1}^3 + {X_1}^3)({X_1}^3 + {Y_1}^3) + {X_1}^3({Y_1}^3 + {Z_1}^3)({Y_1}^3 + {Z_1}^3))\,,$$
$$Y_3 = d({X_1}^3({Y_1}^3 + {Z_1}^3)({X_1}^3 + {Y_1}^3) + {Y_1}^3({Z_1}^3 + {X_1}^3)({Z_1}^3 + {X_1}^3))\,,$$
$$Z_3 = ({X_1}^3 + {Y_1}^3 + {Z_1}^3)(({Y_1}^3 + {Z_1}^3)({Z_1}^3 + {X_1}^3) + ({X_1}^3 + {Y_1}^3)^2)\ .$$

For generalized binary Hessian curves, we suggest the following formulas. If $d \neq 0$, let $e = d^{-1}$. The following algorithm computes $(X_3 : Y_3 : Z_3)$ and requires $7\mathbf{M} + 6\mathbf{S} + 3\mathbf{D}$ (and $7\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ if either c or e is small),

$$A = {X_1}^3,\ B = {Y_1}^3,\ C = c{Z_1}^3,\ E = A^2,\ F = B^2,\ G = C^2,$$
$$H = (A + C)(F + G),\ I = (B + C)(E + G),\ J = (A + B)(E + F),$$
$$K = (A + B + C)(E + F + G),\ L = H + I + (1 + ce^3)K,$$
$$X_3 = H + J + L,\ Y_3 = I + J + L,\ Z_3 = eL\ .$$

5.4 Differential Addition

We now devise differential addition formulas on binary Hessian curves using w-coordinates, where for a point (x, y) on the binary curve $\mathrm{H}_{c,d}$, $w(x, y)$ is defined by a *symmetric* function in terms of the coordinates x, y.

The w-coordinates for differential addition require computing $w(P+Q)$ given $w(P)$, $w(Q)$ and $w(P-Q)$; and the w-coordinates for differential doubling require computing $w(2P)$ given $w(P)$. We recall, [31,6], that using w-coordinate differential addition and doubling formulas, one can recursively compute $w((2m+1)P)$ and $w(2mP)$ given $w(mP)$ and $w((m+1)P)$.

Let (x_2, y_2) be a point on $\mathrm{H}_{c,d}$ and let $(x_4, y_4) = 2(x_2, y_2)$. Write $u_i = x_i + y_i$ and $v_i = x_i y_i$ for $i = 2, 4$. From doubling formulas (4), we obtain

$$u_4 = \frac{{u_2}^4 + cd}{d{u_2}^2 + c} \quad \text{and} \quad v_4 = \frac{{v_2}^4 + cd{v_2}^2}{d^2{v_2}^2 + c^2}\ . \tag{15}$$

Assume that $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_5, y_5)$ are affine points on $\mathrm{H}_{c,d}$ satisfying $(x_1, y_1) = (x_3, y_3) - (x_2, y_2)$ and $(x_5, y_5) = (x_2, y_2) + (x_3, y_3)$. Write $u_i = x_i + y_i$ and $v_i = x_i y_i$ for $i = 1, 2, 3, 5$. Using the addition formulas (3), we obtain

$$u_1 + u_5 = \frac{{u_2}^2{u_3}^2 + du_2u_3(u_2 + u_3) + d^2u_2u_3}{d({u_2}^2 + {u_3}^2) + u_2u_3(u_2 + u_3 + d) + c},$$
$$u_1u_5 = \frac{d{u_2}^2{u_3}^2 + c({u_2}^2 + {u_3}^2 + d^2)}{d({u_2}^2 + {u_3}^2) + u_2u_3(u_2 + u_3 + d) + c}\ .$$

Furthermore, we have

$$\begin{aligned} v_1 + v_5 &= \frac{(c + dv_2)(c + dv_3)}{(v_2 + v_3)^2}, \\ v_1v_5 &= \frac{{v_2}^2{v_3}^2 + cdv_2v_3 + c^2(v_2 + v_3)}{(v_2 + v_3)^2}\ . \end{aligned} \tag{16}$$

Using above affine formulas one can obtain fast projective and mixed differential addition and doubling formulas. In order to speed up these formulas, we consider the following w-coordinates. We write $w_i = c + dv_i$ for $i = 1, 2, \ldots, 5$. In other words, $w_i = {x_i}^3 + {y_i}^3$. Here, $d \neq 0$. From (15), we have

$$w_4 = \frac{{w_2}^4 + c^3(d^3 + c)}{d^3{w_2}^2}\ .$$

Using the formulas (16), we obtain

$$w_1 + w_5 = \frac{d^3w_2w_3}{(w_2 + w_3)^2} \quad \text{and} \quad w_1w_5 = \frac{{w_2}^2{w_3}^2 + c^3(d^3 + c)}{(w_2 + w_3)^2}\ .$$

To have projective formulas, we assume that w_i are given by the fractions W_i/Z_i for $i = 1, 2, 3$. The following explicit formulas give the output w_5 defined by W_5/Z_5:

$$\begin{gathered} A = W_2 Z_3,\ B = W_3 Z_2,\ C = AB,\ U = d^3 C,\ V = (A+B)^2, \\ Z_5 = Z_1 V,\ W_5 = Z_1 U + W_1 V\ . \end{gathered} \tag{17}$$

These formulas require $6\mathbf{M}+1\mathbf{S}+1\mathbf{D}$. Furthermore, the cost of mixed differential addition with w-coordinates is $4\mathbf{M}+1\mathbf{S}+1\mathbf{D}$ by setting $Z_1 = 1$.

Moreover, we write w_4 by the fraction W_4/Z_4. Then, the explicit doubling formulas

$$\begin{gathered} A = {W_2}^2,\ B = {Z_2}^2,\ C = A + \sqrt{c^3(d^3+c)}B,\ D = d^3 B, \\ W_4 = C^2,\ Z_4 = AD \end{gathered} \tag{18}$$

use $1\mathbf{M}+3\mathbf{S}+2\mathbf{D}$. If $c = 1$, i.e., $\mathbf{H}_{c,d}$ is a Hessian curve, then the explicit doubling formulas use $1\mathbf{M}+3\mathbf{S}+1\mathbf{D}$:

$$\begin{gathered} A = {W_2}^2,\ B = {Z_2}^2,\ C = A + B,\ D = (1/\sqrt{d^3})C, \\ W_4 = (B+D)^2,\ Z_4 = AB\ . \end{gathered} \tag{19}$$

As a result, the total cost of projective w-coordinate differential addition and doubling is $7\mathbf{M}+4\mathbf{S}+3\mathbf{D}$. Also, the mixed w-coordinate differential addition and doubling formulas use $5\mathbf{M}+4\mathbf{S}+3\mathbf{D}$. For Hessian curves $\mathbf{H}_{1,d}$, the total costs of projective and mixed w-coordinate differential addition and doubling are $7\mathbf{M}+4\mathbf{S}+2\mathbf{D}$ and $5\mathbf{M}+4\mathbf{S}+2\mathbf{D}$, respectively. Furthermore, if the parameter d of the curve $\mathbf{H}_{c,d}$ is chosen small then the total costs of projective and mixed w-coordinate differential addition and doubling reduces to $7\mathbf{M}+4\mathbf{S}+1\mathbf{D}$ and $5\mathbf{M}+4\mathbf{S}+1\mathbf{D}$, respectively. Moreover, from Proposition 1, we can see that the mixed w-coordinate addition and doubling formulas are *complete*.

Table 2. Cost of differential addition and doubling for families of binary elliptic curves

Curve shape	Representation	Projective differential addition+doubling	Mixed differential addition+doubling
Short Weierstraß $y^2 + xy = x^3 + a_2x^2 + a_6$	$XZ(x=X/Z)$ [28]	$7\mathbf{M}+5\mathbf{S}+1\mathbf{D}$	$5\mathbf{M}+5\mathbf{S}+1\mathbf{D}$
	$XZ(x=X/Z)$ [18]	$6\mathbf{M}+5\mathbf{S}+1\mathbf{D}$	$5\mathbf{M}+5\mathbf{S}+1\mathbf{D}$
	$XZ(x=X/Z)$ [34, §3.1]	$7\mathbf{M}+4\mathbf{S}+1\mathbf{D}$	$5\mathbf{M}+4\mathbf{S}+1\mathbf{D}$
	$XZ(x=X/Z)$ [34, §3.2]	$6\mathbf{M}+5\mathbf{S}+2\mathbf{D}$	$5\mathbf{M}+5\mathbf{S}+2\mathbf{D}$
Binary Edwards $d_1(x+y)+d_2(x^2+y^2)$ $= xy + xy(x+y) + x^2y^2$	$WZ(x+y=W/Z)$ [6]	$8\mathbf{M}+4\mathbf{S}+4\mathbf{D}$	$6\mathbf{M}+4\mathbf{S}+4\mathbf{D}$
	WZ with $d_1 = d_2$ [6]	$7\mathbf{M}+4\mathbf{S}+2\mathbf{D}$	$5\mathbf{M}+4\mathbf{S}+2\mathbf{D}$
Hessian $x^3+y^3+1=dxy$	$WZ(1+dxy=W/Z)$ formulas (17), (19)	$7\mathbf{M}+4\mathbf{S}+2\mathbf{D}$	$5\mathbf{M}+4\mathbf{S}+2\mathbf{D}$
Generalized Hessian $x^3+y^3+c=dxy$	$WZ(c+dxy=W/Z)$ formulas (17), (18)	$7\mathbf{M}+4\mathbf{S}+3\mathbf{D}$	$5\mathbf{M}+4\mathbf{S}+3\mathbf{D}$
	WZ with small d formulas (17), (18)	$7\mathbf{M}+4\mathbf{S}+1\mathbf{D}$	$5\mathbf{M}+4\mathbf{S}+1\mathbf{D}$

Table 2 shows the cost of differential addition and doubling for different coordinate systems on binary elliptic curves. From Table 2, we see that our w-coordinate representations for generalized Hessian curves are competitive with other representations for binary elliptic curves.

6 Conclusion

In this paper, the family of *generalized Hessian curves* has been presented. This family covers more isomorphism classes of elliptic curves than Hessian curves. For every elliptic curve E over a finite field $\mathbb{F}_q$, the group $E(\mathbb{F}_q)$ has a point of order 3 if and only if E is isomorphic over $\mathbb{F}_q$ to a generalized Hessian curve.

Unified addition formulas have been presented for generalized Hessian curves $\mathbf{H}_{c,d}$ over a field $\mathbb{F}$, see formulas (9), (10). In particular, these formulas are unified for Hessian curves $\mathbf{H}_{1,d}$. Further, the formulas are *complete* if c is not a cube in $\mathbb{F}$.

The cost of projective formulas using algorithm (11) is $12\mathbf{M} + 1\mathbf{D}$. Also, the mixed addition formulas require $10\mathbf{M} + 1\mathbf{D}$. For generalized Hessian curves $\mathbf{H}_{c,d}$ over $\mathbb{F}$ with characteristic $p \neq 2$, the projective addition formulas (12) using extended coordinates has a cost of $6\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$. The mixed formulas require $5\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$.

When $p = 2$, the generalized binary Hessian curves provide very fast and efficient addition formulas. Projective formulas (11) require $12\mathbf{M} + 1\mathbf{D}$ and the mixed addition formulas need $10\mathbf{M} + 1\mathbf{D}$. Moreover, using the extended coordinates, formulas (13) perform a projective addition using $9\mathbf{M} + 3\mathbf{S} + 2\mathbf{D}$ and a mixed addition using $8\mathbf{M} + 3\mathbf{S} + 2\mathbf{D}$. Several doubling and tripling formulas have been presented for generalized Hessian curves which improve the previous doubling and tripling formulas on Hessian curves. Also, very competitive differential addition and doubling formulas have been presented for generalized binary Hessian curves.

References

1. Al-Daoud, E., Mahmod, R., Rushdan, M., Kiliçman, A.: A new addition formula for elliptic curves over GF(2^n). IEEE Trans. Computers 51(8), 972–975 (2002)
2. Avanzi, R., Cohen, H., Doche, C., Frey, G., Lange, T., Nguyen, K., Vercauteren, F.: Handbook of Elliptic and Hyperelliptic Curve Cryptography. CRC Press, Boca Raton (2005)
3. Avanzi, R.M., Dimitrov, V.S., Doche, C., Sica, F.: Extending scalar multiplication using double bases. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 130–144. Springer, Heidelberg (2006)
4. Bernstein, D.J., Lange, T.: Explicit-formulas database, `http://www.hyperelliptic.org/EFD/`
5. Bernstein, D.J., Lange, T.: Faster addition and doubling on elliptic curves. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 29–50. Springer, Heidelberg (2007)

6. Bernstein, D.J., Lange, T., Farashahi, R.R.: Binary Edwards curves. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 244–265. Springer, Heidelberg (2008)
7. Bersntein, D.J., Kohel, D., Lange, T.: Twisted Hessian curves, http://www.hyperelliptic.org/EFD/g1p/auto-twistedhessian.html
8. Billet, O., Joye, M.: The Jacobi model of an elliptic curve and side-channel analysis. In: Fossorier, M.P.C., Høholdt, T., Poli, A. (eds.) AAECC 2003. LNCS, vol. 2643, pp. 34–42. Springer, Heidelberg (2003)
9. Blake, I.F., Seroussi, G., Smart, N.P.: Advances in Elliptic Curve Cryptography. Cambridge University Press, Cambridge (2005)
10. Brier, É., Déchène, I., Joye, M.: Unified point addition formulæ for elliptic curve cryptosystems. In: Embedded Cryptographic Hardware: Methodologies & Architectures, pp. 247–256. Nova Science Publishers (2004)
11. Brier, É., Joye, M.: Weierstraß elliptic curves and side-channel attacks. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 335–345. Springer, Heidelberg (2002)
12. Cassels, J.W.S.: Lectures on Elliptic Curves. Cambridge University Press, Cambridge (1991)
13. Chudnovsky, D.V., Chudnovsky, G.V.: Sequences of numbers generated by addition in formal groups and new primality and factorization tests. Advances in Applied Mathematics 7(4), 385–434 (1986)
14. Dimitrov, V.S., Imbert, L., Mishra, P.K.: Efficient and secure elliptic curve point multiplication using double-base chains. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 59–78. Springer, Heidelberg (2005)
15. Doche, C., Imbert, L.: Extended double-base number system with applications to elliptic curve cryptography. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. LNCS, vol. 4329, pp. 335–348. Springer, Heidelberg (2006)
16. Farashahi, R.R.: On the number of distinct Legendre, Jacobi and Hessian curves (Preprint)
17. Farashahi, R.R., Shparlinski, I.E.: On the number of distinct elliptic curves in some families. Designs, Codes and Cryptography 54(1), 83–99 (2010)
18. Gaudry, P., Lubicz, D.: The arithmetic of characteristic 2 Kummer surfaces. Finite Fields and Applications 15, 246–260 (2009)
19. Hesse, O.: Über die Elimination der Variabeln aus drei algebraischen Gleichungen vom zweiten Grade mit zwei Variabeln. Journal für die reine und angewandte Mathematik 10, 68–96 (1844)
20. Higuchi, A., Takagi, N.: A fast addition algorithm for elliptic curve arithmetic in GF(2^n) using projective coordinates. Inf. Process. Lett. 76(3), 101–103 (2000)
21. Hisil, H., Carter, G., Dawson, E.: New formulæ for efficient elliptic curve arithmetic. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 138–151. Springer, Heidelberg (2007)
22. Hisil, H., Wong, K.K.-H., Carter, G., Dawson, E.: Faster group operations on elliptic curves. In: Brankovic, L., Susilo, W. (eds.) Australasian Information Security Conference (AISC 2009). Conferences in Research and Practice in Information Technology (CRPIT), vol. 98, pp. 7–19 (2009)
23. Joye, M., Quisquater, J.-J.: Hessian elliptic curves and side-channel attacks. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 402–410. Springer, Heidelberg (2001)
24. Kim, K.H., Kim, S.I.: A new method for speeding up arithmetic on elliptic curves over binary fields. Cryptology ePrint Archive, Report 2007/181 (2007)

25. Knapp, A.: Elliptic Curves. Princeton University Press, Princeton (1992)
26. Koblitz, N.: Elliptic curve cryptosystems. Mathematics of Computation 48(177), 203–209 (1987)
27. Liardet, P.-Y., Smart, N.P.: Preventing SPA/DPA in ECC systems using the Jacobi form. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 391–401. Springer, Heidelberg (2001)
28. López, J., Dahab, R.: Fast multiplication on elliptic curves over GF(2^n) without precomputation. In: Koç, Ç.K., Paar, C. (eds.) CHES 1999. LNCS, vol. 1717, pp. 316–327. Springer, Heidelberg (1999)
29. López, J., Dahab, R.: Improved algorithms for elliptic curve arithmetic in GF(2^n). In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 201–212. Springer, Heidelberg (1999)
30. Miller, V.S.: Use of elliptic curves in cryptography. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
31. Montgomery, P.L.: Speeding the Pollard and elliptic curve methods of factorization. Mathematics of Computation 48(177), 243–264 (1987)
32. Silverman, J.H.: The Arithmetic of Elliptic Curves. Springer, Heidelberg (1986)
33. Smart, N.P.: The Hessian form of an elliptic curve. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 118–125. Springer, Heidelberg (2001)
34. Stam, M.: On Montgomery-like representationsfor elliptic curves over GF(2^n). In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 240–253. Springer, Heidelberg (2002)
35. Washington, L.C.: Elliptic Curves: Number Theory and Cryptography. CRC Press, Boca Raton (2005)

A On the Number of Distinct Generalized Hessian Curves

A.1 The Number of Distinct j-Invariants

We recall from [16] that the number of distinct Hessian curves over the finite field $\mathbb{F}_q$, up to isomorphism over $\overline{\mathbb{F}}_q$, is $q-1$, $\lfloor (q+11)/12 \rfloor$ and $\lfloor q/2 \rfloor$ if $q \equiv 0, 1, 2 \pmod 3$, respectively. Using the similar method described in [16,17], we give explicit formulas for the number of distinct generalized Hessian curves over the finite field $\mathbb{F}_q$ up to isomorphism over $\overline{\mathbb{F}}_q$.

From Equation (2), the j-invariant of $\mathrm{H}_{c,d}$ is $j(\mathrm{H}_{c,d}) = \frac{1}{c}\left(\frac{d(d^3+216c)}{d^3-27c}\right)^3$. We use J_{H} to denote the set of distinct j-invariants of the family of generalized Hessian curves over $\mathbb{F}_q$ and we let $J_{\mathrm{H}}(q) = \#J_{\mathrm{H}}$. For c in $\mathbb{F}_q$, with $c \neq 0$, we let

$$J_{\mathrm{H}_c} = \left\{ j \mid j = j(\mathrm{H}_{c,d}),\ d \in \mathbb{F}_q, d^3 \neq 27c \right\} .$$

Clearly, $J_{\mathrm{H}} = \bigcup_{c \in \mathbb{F}_q^*} J_{\mathrm{H}_c}$.

Lemma 1. *Let $c_1, c_2 \in \mathbb{F}_q^*$ and let $c = c_1/c_2$. If c is a cube in $\mathbb{F}_q$, then $J_{\mathrm{H}_{c_1}} = J_{\mathrm{H}_{c_2}}$. If c is not a cube in $\mathbb{F}_q$, then we have $J_{\mathrm{H}_{c_1}} \cap J_{\mathrm{H}_{c_2}} = \{0\}$.*

Proof. Suppose $c = \zeta^3$ is a cube in $\mathbb{F}_q$. For all $d \in \mathbb{F}_q$ with $d^3 \neq 27c$, we have $j(\mathrm{H}_{c_1,d}) = j(\mathrm{H}_{c_2,d/\zeta})$ and similarly $j(\mathrm{H}_{c_2,d}) = j(\mathrm{H}_{c_1,\zeta d})$. Therefore, $J_{\mathrm{H}_{c_1}} = J_{\mathrm{H}_{c_2}}$.

Now, suppose that c is not a cube in $\mathbb{F}_q$. Let $j \in J_{\mathrm{H}_{c_1}} \cap J_{\mathrm{H}_{c_2}}$. Then, $j = \frac{1}{c_1}\left(\frac{d_1({d_1}^3+216c_1)}{{d_1}^3-27c_1}\right)^3 = \frac{1}{c_2}\left(\frac{d_2({d_2}^3+216c_2)}{{d_2}^3-27c_2}\right)^3$ for some $d_1, d_2 \in \mathbb{F}_q$. If $j \neq 0$, we see that $c = c_1/c_2$ is a cube in $\mathbb{F}_q$, a contradiction. So, $J_{\mathrm{H}_{c_1}} \cap J_{\mathrm{H}_{c_2}} = \{0\}$. □

Lemma 2. *For $q \equiv 1 \pmod 3$, if c is not a cube in $\mathbb{F}_q$, we have $\#J_{\mathrm{H}_c} = (q+2)/3$.*

Proof. For $d \in \mathbb{F}_q$ with $d^3 \neq 27c$, we let $j(\mathrm{H}_{c,d}) = \frac{1}{c}\,(F(d))^3$ where $F(U) = \frac{U(U^3+216c)}{U^3-27c}$. We consider the bivariate rational function $F(U) - F(V)$. We obtain

$$F(U) - F(V) = \frac{U-V}{U^3-27c}\prod_{i=1}^{3}\left(U - \frac{3\zeta_i(V+6\zeta_i)}{V-3\zeta_i}\right),$$

where, $\zeta_1, \zeta_2, \zeta_3$ are three cubic roots of c in $\overline{\mathbb{F}}_q$. For all $u, v \in \mathbb{F}_q$ with $u^3 \neq 27c$, $v^3 \neq 27c$, we see that $F(u) = F(v)$ if and only if $u = v$. Hence, F is injective over $\mathbb{F}_q$ and we have $F(\mathbb{F}_q) = \mathbb{F}_q$. Now, consider the map $\kappa : \mathbb{F}_q^* \to \mathbb{F}_q^*$ by $\kappa(x) = \frac{1}{c}x^3$. This map is $3:1$, if $q \equiv 1 \pmod 3$. So, $\#J_{\mathrm{H}_c} = (q-1)/3 + 1$. □

Theorem 4. *For any prime power q, for the number $J_{\mathrm{H}}(q)$ of distinct values of the j-invariant of the family of generalized Hessian curves over the finite field $\mathbb{F}_q$, we have*

$$J_{\mathrm{H}}(q) = \begin{cases} q-1, & \text{if } q \equiv 0 \pmod 3 \\ \lfloor (3q+1)/4 \rfloor, & \text{if } q \equiv 1 \pmod 3 \\ \lfloor q/2 \rfloor, & \text{if } q \equiv 2 \pmod 3 \end{cases}.$$

Proof. If $q \not\equiv 1 \pmod 3$, every element of $\mathbb{F}_q$ is a cube in $\mathbb{F}_q$. Next, Lemma 1 implies that, for all $c \in \mathbb{F}_q^*$, we have $J_{\mathrm{H}_c} = J_{\mathrm{H}_1}$. Therefore, $J_{\mathrm{H}} = J_{\mathrm{H}_1}$. Then, from [16, Theorem 14], we have

$$J_{\mathrm{H}}(q) = \begin{cases} q-1, & \text{if } q \equiv 0 \pmod 3 \\ \lfloor q/2 \rfloor, & \text{if } q \equiv 2 \pmod 3 \end{cases}.$$

For $q \equiv 1 \pmod 3$, we fix a value $c \in \mathbb{F}_q$ that is not a cube in $\mathbb{F}_q$. Following Lemma 1, we write $J_{\mathrm{H}} = J_{\mathrm{H}_c} \cup J_{\mathrm{H}_{c^2}} \cup J_{\mathrm{H}_1}$, where $J_{\mathrm{H}_c} \cap J_{\mathrm{H}_{c^2}} = J_{\mathrm{H}_c} \cap J_{\mathrm{H}_1} = J_{\mathrm{H}_{c^2}} \cap J_{\mathrm{H}_1} = \{0\}$. By Lemma 2, we have $\#J_{\mathrm{H}_c} = \#J_{\mathrm{H}_{c^2}} = (q+2)/3$. Moreover, from [16, Theorem 14], we have

$$\#J_{\mathrm{H}_1} = \begin{cases} (q+11)/12, & \text{if } q \equiv 1 \pmod{12} \\ (q+8)/12, & \text{if } q \equiv 4 \pmod{12} \\ (q+5)/12, & \text{if } q \equiv 7 \pmod{12} \end{cases}.$$

Therefore, we have

$$J_{\mathrm{H}}(q) = \begin{cases} (3q+1)/4, & \text{if } q \equiv 1 \pmod{12} \\ 3q/4, & \text{if } q \equiv 4 \pmod{12} \\ (3q-1)/4, & \text{if } q \equiv 7 \pmod{12} \end{cases},$$

which completes the proof. □

A.2 The Number of $\mathbb{F}_q$-Isomorphism Classes

We recall from [16] that the number of $\mathbb{F}_q$-isomorphism classes of Hessian curves over $\mathbb{F}_q$ is $\lfloor (q+11)/12 \rfloor$ if $q \equiv 1 \pmod 3$ and $q-1$ if $q \not\equiv 1 \pmod 3$. The following theorem gives explicit formulas for the number of distinct generalized Hessian curves, up to $\mathbb{F}_q$-isomorphism, over the finite field $\mathbb{F}_q$.

Theorem 5. *For any prime power q, the number of $\mathbb{F}_q$-isomorphism classes of the family of generalized Hessian curves over the finite field $\mathbb{F}_q$ is*

$$\begin{cases} \lfloor (3(q+3)/4 \rfloor, & \textit{if } q \equiv 1 \pmod 3 \\ q-1, & \textit{if } q \equiv 0,2 \pmod 3 \end{cases}.$$

Proof. We use $I_{\mathrm{H}}(q)$ to denote the number of $\mathbb{F}_q$-isomorphism classes of the family of generalized Hessian curves over $\mathbb{F}_q$.

If $q \equiv 0,2 \pmod 3$, then every generalized Hessian curve is $\mathbb{F}_q$-isomorphic to a Hessian curve via the map given by Equations (1). So, $I_{\mathrm{H}}(q)$ equals the number of $\mathbb{F}_q$-isomorphism classes of the family of Hessian curves over $\mathbb{F}_q$. Then, from [16, Theorem 15], we have $I_{\mathrm{H}}(q) = q-1$ if $q \not\equiv 1 \pmod 3$.

Now, suppose that $q \equiv 1 \pmod 3$. For $a \in \mathbb{F}_q$, let $i_H(a)$ be the set of $\mathbb{F}_q$-isomorphism classes of generalized Hessian curves $\mathrm{H}_{c,d}$ with $j(\mathrm{H}_{c,d}) = a$. So, $\#i_H(a)$ is the number of distinct generalized Hessian curves with j-invariant a that are twists of each other. Clearly, $\#i_H(a) = 0$, if $a \notin J_{\mathrm{H}}$. We note that, for all elliptic curve E over $\mathbb{F}_q$, we have $\#E(\mathbb{F}_q) + \#E_t(\mathbb{F}_q) = 2q+2$, where E_t is the nontrivial quadratic twist of E. We also recall that the order of the group of $\mathbb{F}_q$-rational points of a generalized Hessian curve is divisible by 3 (see Theorem 2). Since $q \equiv 1 \pmod 3$, if the isomorphism class of $\mathrm{H}_{c,d}$ is in $i_H(a)$ then the isomorphism class of the nontrivial quadratic twist of $\mathrm{H}_{c,d}$ is not in $i_H(a)$. So, $\#i_H(a) = 1$ if $a \in J_{\mathrm{H}}$ and $a \neq 0, 1728$. Moreover, one can show that $\#i_H(a) = 3$ if $a = 0$ and $\#i_H(a) = 1$ if $a = 1728$, $a \neq 0$ and $a \in J_{\mathrm{H}}$. Therefore, we have

$$I_{\mathrm{H}}(q) = \sum_{a \in \mathbb{F}_q} i_H(a) = \sum_{a \in J_{\mathrm{H}}} i_H(a) = 2 + \sum_{a \in J_{\mathrm{H}}} 1 = 2 + J_{\mathrm{H}}(q) .$$

From the proof of Theorem 4, we have

$$I_{\mathrm{H}}(q) = \begin{cases} (3q+9)/4, & \text{if } q \equiv 1 \pmod{12} \\ (3q+8)/4, & \text{if } q \equiv 4 \pmod{12} \\ (3q+7)/4, & \text{if } q \equiv 7 \pmod{12} \end{cases},$$

which completes the proof. □

CCA Proxy Re-Encryption without Bilinear Maps in the Standard Model

Toshihide Matsuda[1], Ryo Nishimaki[2], and Keisuke Tanaka[1]

[1] Department of Mathematical and Computing Sciences, Tokyo Institute of Technology,W8-55, 2-12-1 Ookayama Meguro-ku, Tokyo 152-8552, Japan
{matsuda5,keisuke}@is.titech.ac.jp
[2] NTT Laboratories, 3-9-11 Midori-cho, Musashino-shi, Tokyo, 180-8585, Japan
nishimaki.ryo@lab.ntt.co.jp

Abstract. Proxy re-encryption (PRE) is a cryptographic application proposed by Blaze, Bleumer, and Strauss. It is an encryption system with a special property in which the semi-honest third party, *the proxy*, can re-encrypt ciphertexts for Alice into other ciphertexts for Bob without using Alice's secret key. We can classify PRE into bidirectional and unidirectional schemes. Canetti and Hohenberger formalized the semantic security under chosen ciphertext attack for PRE, the PRE-CCA security. Several schemes satisfy the PRE-CCA security as a bidirectional or unidirectional scheme. However, some PRE schemes need a bilinear map in the standard model, and the other PRE schemes are PRE-CCA secure in the random oracle model before our work. In this paper, we construct a bidirectional PRE-CCA proxy re-encryption *without bilinear maps in the standard model*. We study lossy trapdoor functions (LTDFs) based on the decisional Diffie-Hellman (DDH) assumption proposed by Peikert and Waters. We define a new variant of LTDFs, *re-applicable LTDFs*, which are specialized LTDFs for PRE, and use them for our scheme.

1 Introduction

1.1 Background

Proxy re-encryption (PRE) is a cryptographic application proposed by Blaze, Bleumer, and Strauss [4]. It is an encryption system with a special property in which the semi-honest third party, *the proxy*, can re-encrypt ciphertexts for Alice into other ciphertexts for Bob without using Alice's secret key. In the other words, if the proxy has a re-encryption key $rk_{A\leftrightarrow B}$ from Alice to Bob, the proxy can translate a ciphertext C_A under Alice's public key pk_A into another ciphertext C_B under Bob's public key pk_B. This translation requires only $rk_{A\leftrightarrow B}$ and keeps the message m secret for the proxy. There are many PRE cryptographic applications, such as email-forwarding, secure file systems, DRM, and secure mailing lists [2,4,11,21].

Ivan and Dodis classified PRE into two types, *bidirectional* and *unidirectional* [10]. The former means that, if the proxy has the re-encryption key $rk_{A\leftrightarrow B}$, the proxy cannot only re-encrypt ciphertexts from Alice to Bob, but also in the opposite direction. That is, we can assume that the re-encryption key $rk_{A\leftrightarrow B}$ from Alice to Bob is identical to

$rk_{B\leftrightarrow A}$ from Bob to Alice. On the other hand, the latter means that, the re-encryption key $rk_{A\rightarrow B}$ from Alice to Bob never helps re-encryption of the opposite direction. They also considered the notions of *single-hop* and *multi-hop*. The former means that a re-encrypted ciphertext cannot be further re-encrypted. In contrast, the later means that a ciphertext can be re-encrypted from Alice to Bob to Carol and so on. We discuss only bidirectional schemes in this paper.

The PRE-CCA Security. The security notion of indistinguishability against chosen cipher text attacks (IND-CCA) on encryption systems was proposed by Naor and Yung [15]. Many IND-CCA secure encryption systems have been proposed.

Canetti and Hohenberger applied the notion of IND-CCA to PRE for the bidirectional scheme [6]. They formalized the security notion as the (bidirectional) PRE-CCA security. They also investigated simulation-based security definitions that guarantee universally composable security. They constructed a bidirectional and multi-hop PRE-CCA scheme based on the decisional bilinear Diffie-Hellman (DBDH) assumption, which requires bilinear maps. Later, some research groups proposed bidirectional or unidirectional PRE-CCA schemes with bilinear maps or in the random oracle model [7,12,20,22].

1.2 Our Contribution

We construct a bidirectional and multi-hop PRE-CCA scheme without *bilinear maps* in *the standard model*. All previous PRE-CCA secure schemes in the standard model use bilinear maps. Our scheme is constructed in three steps.

First, we define a new cryptographic primitive, *re-applicable lossy trapdoor functions* (re-applicable LTDFs), which are specialized lossy trapdoor functions for PRE. They consist of nine algorithms and a set of tags, (ParGen, $\mathsf{LossyGen}$, $\mathsf{LossyEval}$, $\mathsf{LossyInv}$, $\mathsf{ReIndex}$, ReEval, $\mathsf{PrivReEval}$, Trans, $\mathsf{FakeKey}$) and $\mathcal{T}$. Second, we construct a bidirectional PRE-CCA scheme by using re-applicable LTDFs. We modify the original Peikert and Waters encryption scheme on several points. Third, we construct re-applicable LTDFs based on the *decisional Diffie-Hellman* (DDH) assumption.

Our Techniques. As stated above, we modify the original Peikert and Waters encryption scheme on several points for PRE. Our PRE scheme uses an index of all-but-one functions as the public parameter. The original scheme uses this index as a part of a public key. Our scheme generates a signature of a part of a ciphertext (c_2, c_3) in the encryption scheme. The original scheme generates a signature of all the main parts of a ciphertext (c_1, c_2, c_3) in the encryption process. The most different point is that our scheme re-encrypts c_1 from pk_A to pk_B by using $rk_{A\leftrightarrow B}$.

We modify LTDFs proposed by Peikert and Waters on one point for construction of re-applicable LTDFs. It is that an injective index is not $\mathsf{Enc}_{pk}(\boldsymbol{I})$, but $\mathsf{Enc}_{pk}(\tau\boldsymbol{I})$, where τ is a tag and $\boldsymbol{I}$ is the identity matrix. This modification is a technical change that satisfies the definition of re-applicable LTDFs.

The Peikert and Waters Encryption. Peikert and Waters proposed LTDFs and constructed an IND-CCA public-key encryption by using LTDFs [17]. We briefly review their encryption scheme. Let f_s and $f_{s'}$ be functions with a domain $\{0,1\}^n$, where f_s is an injective function with a trapdoor td, and $f_{s'}$ is a lossy function of which the size of a range is 2^{n-k} at most. LTDFs have the following property: Given f, the distinguisher cannot distinguish whether f is f_s or $f_{s'}$. They constructed them as $f_s(\boldsymbol{x}) = \boldsymbol{x}\mathsf{Enc}_{pk}(\boldsymbol{I})$ and $f_{s'}(\boldsymbol{x}) = \boldsymbol{x}\mathsf{Enc}_{pk}(\boldsymbol{0})$ where $\boldsymbol{I}$ and $\boldsymbol{0}$ is the identity and the zero matrix, and Enc is a homomorphic matrix encryption. From the homomorphism of Enc, we obtain $f_s(\boldsymbol{x}) = \boldsymbol{x}\mathsf{Enc}_{pk}(\boldsymbol{I}) = \mathsf{Enc}_{pk}(\boldsymbol{xI}) = \mathsf{Enc}_{pk}(\boldsymbol{x})$ and $f_{s'}(\boldsymbol{x}) = \boldsymbol{x}\mathsf{Enc}_{pk}(\boldsymbol{0}) = \mathsf{Enc}_{pk}(\boldsymbol{x0}) = \mathsf{Enc}_{pk}(\boldsymbol{0})$. We can reconstruct $\boldsymbol{x}$ from $f_s(\boldsymbol{x})$ with a secret key sk. On the other hand, we never obtain $\boldsymbol{x}$ from $f_{s'}(\boldsymbol{x})$ *information-theoretically*. They also proposed *all-but-one trapdoor functions* which have similar property to LTDFs. Let g_{s',b^*} be a function with a domain $B \times \{0,1\}^n$, where B is a finite set and $b^* \in B$. For every $b \neq b^*$, $g_{s',b^*}(b,\cdot)$ is an invertible function with a trapdoor td. On the other hand, $g_{s',b^*}(b^*,\cdot)$ is a lossy function.

Peikert and Waters constructed their IND-CCA encryption scheme by using f_s and g_{s',b^*} as follows. The encryption algorithm randomly chooses $x \in \{0,1\}^n$ and selects a key pair of a one-time signature (vk, sk_σ). Then, it computes a ciphertext as

$$C = (vk, c_1, c_2, c_3, \sigma) = (vk, f_s(x), g_{s',b^*}(vk, x), h(x) \oplus m, \mathsf{SigSign}(sk_\sigma, c_1, c_2, c_3)),$$

where h is a pair-wise independent hash and $\mathsf{SigSign}$ is a signing algorithm in a signature scheme. They proved that this scheme satisfied the IND-CCA security.

Observation of the Peikert and Waters Encryption: Free Part for Signature. The above encryption algorithm must sign all (c_1, c_2, c_3) and make the signature σ. The signature σ and vk guarantees that (c_1, c_2, c_3) is signed with the signing key sk_σ. That is, (c_1, c_2, c_3) is fixed by σ and vk. For this fixed (c_1, c_2, c_3), the Peikert and Waters encryption achieves the IND-CCA security.

However, we find that it is not necessary to sign all (c_1, c_2, c_3) in order to achieve the IND-CCA security. Moreover, we find that it does not need to sign c_1. The reason is as follows. If (c_1, c_2, c_3) is fixed by σ and vk, randomness x and a message m also are fixed because of the injectivity of f_s and g_{s',b^*}. That is, we can consider σ as a signature of x and m as well as (c_1, c_2, c_3). In addition, if x and m are determined, (c_1, c_2, c_3) is determined. We understand that it is necessary to sign x and m, not (c_1, c_2, c_3). We replace a signature of (c_1, c_2, c_3) with a signature of (c_2, c_3). A pair of x and m is fixed similarly to the case of (c_1, c_2, c_3) because of the injectivity of g_{s',b^*}. That is, the signature of (c_2, c_3) performs tasks of a signature of (c_1, c_2, c_3). We do not need the signature of c_1 to achieve the IND-CCA security. This free c_1 is very important in constructing our proposed scheme, which satisfies the bidirectional PRE-CCA security.

Remarks on Our Scheme. We discuss two points of our scheme: 1. Comparison of efficiency and 2. Construction based on other assumptions.

Our scheme uses an index of re-applicable LTDFs as a public key. We represent this key as an $n \times n$ matrix, which has n^2 group elements. The public parameter contains $n \times n$ matrix as well as public keys since all-but-one trapdoor functions are based on the DDH assumption. One ciphertext and one re-encryption key have $O(n)$ group elements. However, in most of the previous schemes, they consist of a constant number of group elements. For example, one public key is one group element, and one ciphertext consists of five group elements, a verification key, and a signature in the Canetti and Hohenberger's bidirectional scheme, which satisfies the PRE-CCA security as well as ours [6]. Time complexity as well as space complexity are larger than others. Therefore, future work is to construct more efficient schemes.

LTDFs are constructed from various assumptions. Therefore, one may think that we can construct PRE-CCA secure schemes from other assumptions. However, we do not know how to construct PRE-CCA secure schemes from other assumptions now. Factoring, quadratic residue, RSA, or Paillier-based LTDFs do not clearly satisfy the definition of re-applicable lossy trapdoor functions. One might think that decision linear or lattice-based LTDFs work in our proposal, but this is not clear. In other words, they do not guarantee several properties of re-applicable LTDFs. However, we might be able to use LTDFs based on them with other techniques for PRE.

1.3 Related Work

We review previous work on PRE and LTDFs.

Proxy Re-Encryptions. Mambo and Okamoto first proposed the concept of proxy encryption, which delegates the ability of decryption through an interaction [13]. Based on their concept, Blaze, Bleumer, and Strauss proposed the notion of proxy cryptography. PRE is one concept in proxy cryptography [4]. Their construction is nearly similar to the ElGamal encryption and satisfies the CPA security. A re-encryption key is made from the division of secret keys in the scheme. Later, Ivan and Dodis point out that Blaze et.al.'s scheme is bidirectional and multi-hop. Ateniese, Fu, Green, and Hohenberger proposed the first unidirectional scheme, which was single-hop and satisfied the CPA security with a bilinear map [2]. Deng, Weng, Liu, and Chen constructed a bidirectional and single-hop PRE scheme without a bilinear map in the random oracle model [7]. Libert and Vergnaud discussed the unidirectional PRE-CCA security and constructed an unidirectional and single-hop PRE-RCCA[1] scheme with a bilinear map [12]. Shao and Cao proposed an unidirectional and single-hop PRE-CCA scheme in the random oracle model [20][2]. Weng, Chow, Yang, and Deng improved Shao and Cao's scheme, and their scheme also is in the random oracle model [22]. We put this previous work in chronological order in Figure 1. ROM stands for the random oracle model.

Hohenberger, Rothblum, shelat, and Vaikuntanathan argued the existence of obfuscators with PRE and practically constructed an obfuscator as a PRE scheme [9]. Their scheme is CPA-secure.

Recently, Ateniese, Benson, and Hohenberger introduced an additional property on PRE, which is key-privacy (or anonymous) [1]. It is desirable to have this property, if the proxy can freely re-encrypt ciphertexts. Our scheme does not have the key-private property. They constructed a key-private scheme with bilinear maps in the standard model, which satisfies the PRE-CPA security, not CCA.

[1] This security notion is truly weaker than the CCA security and stronger than the CPA security. An adversary can use a decryption oracle in a restricted way. If he sends messages (m_0, m_1) to the challenger and obtains a challenge c, the adversary cannot query ciphertexts of challenge messages m_0 and m_1 to the decryption oracle.

[2] Two research groups posed questions on the security model of this paper and published their discussions on ePrint Archive [22,23]. However, they do not effect our results and we do not mention this in this paper.

Authors	Direction	Hop	Assumption	Security	Bilinear Map	ROM
Blaze et.al. [4]	$\leftrightarrow$	multi	DDH on $\mathbb{G}_p$	PRE-CPA	no	no
Ateniese et.al. [2]	$\rightarrow$	single	eDBDH	PRE-CPA	yes	no
Canetti and Hohenberger [6]	$\leftrightarrow$	multi	DBDH	PRE-CCA	yes	no
Libert and Vergnaud [12]	$\rightarrow$	single	3-wDBDHI	PRE-RCCA	yes	no
Deng et.al. [7]	$\leftrightarrow$	single	mCDH on $\mathbb{G}_p$	PRE-CCA	no	yes
Shao and Cao [20]	$\rightarrow$	single	DDH on $\mathbb{Z}_{N^2}$	PRE-CCA	no	yes
Weng et.al. [22]	$\rightarrow$	single	CDH on $\mathbb{G}_p$	PRE-CCA	no	yes
This paper	$\leftrightarrow$	multi	DDH on $\mathbb{G}_p$	PRE-CCA	*no*	*no*

Fig. 1. Comparison of our work with previous work

Lossy Trapdoor Functions. We review previous work related to LTDFs. Peikert and Waters proposed notions of LTDFs [17]. They showed cryptographic applications based on LTDFs, such as a (ordinary) trapdoor function, a collision-resistant hash function, an oblivious transfer, and an IND-CCA encryption scheme. They also showed the constructions of LTDFs based on the DDH assumption and the LWE assumption. They mentioned that the Paillier encryption realized LTDFs by the similar methodology to the construction based on the DDH assumption. Later, Rosen and Segev noted that the Damgård-Jurik encryption scheme simply satisfied the definition of LTDFs because of its number-theoretic property [18]. The Damgård-Jurik encryption scheme is considered as a generalized Paillier encryption. In addition to [18], Freedman, Goldreich, Kiltz, Rosen, and Segev proposed more constructions of LTDFs [8]. They are based on the d-Linear assumption and the QR assumption. Mol and Yilek showed that slightly LTDFs are sufficient for constructing a IND-CCA secure public-key encryption scheme [14]. Slightly LTDFs lost a $(1 - \omega(\log n))$ fraction of all its input bits.

Other applications of LTDFs have been proposed. Rosen and Segev proposed a new primitive, *a one-way function under correlated products* [19]. They showed the construction of a one-way function under correlated products from LTDFs and the IND-CCA secure encryption by using this primitive. Boldyreva, Fehr, and O'Neill applied LTDFs to the construction of *deterministic encryption* [5]. They constructed a CCA-secure deterministic encryption scheme in the standard model, where the CCA security meant the sense of the semantic security on a message, not indistinguishability of messages. Bellare, Hofheinz, and Yilek formalized a new security notion of encryption, *selective opening attack*, which meant that it kept secret even if an adversary selectively obtained *messages* and *randomness* of ciphertexts [3]. They used LTDFs for the above purpose. Nishimaki, Fujisaki, and Tanaka used all-but-one trapdoor functions for the universally composable commitment scheme [16]. They first proposed a non-interactive string-commitment scheme, which is universally composable.

Organization

In Section 2, we show preliminaries to describe our scheme. In Section 3, we define re-applicable LTDFs. In Section 4, we review the definition of bidirectional PRE schemes, propose our scheme, and describe a sketch of a proof of it. In Section 5, we construct re-applicable LTDFs based on the DDH assumption.

2 Preliminaries

In this section, we show preliminaries to describe our scheme.

2.1 Notation

Let S be a finite set. $s \in_R S$ denotes that the element s is chosen from S uniformly at random. For probabilistic algorithm A, $y \leftarrow A(x)$ denotes that A outputs y on input x with uniform randomness. If A runs in time polynomial in the security parameter, then A is a probabilistic polynomial-time (PPT) algorithm. We say that function $f : \mathbb{N} \to [0, 1]$ 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 $\lambda > k_c$. We say that function $g : \mathbb{N} \to [0, 1]$ is overwhelming in $\lambda \in \mathbb{N}$ if function $f(\lambda) = 1 - g(\lambda)$ is negligible in $\lambda \in \mathbb{N}$. Let X_λ and Y_λ denote random variables over a finite set $Z_\lambda \subset \{0, 1\}^\lambda$, where $\lambda \in \mathbb{N}$ is the security parameter. We say that X_λ and Y_λ are (computationally) indistinguishable if, for every distinguisher D, $|\Pr[D(X_\lambda) = 1] - \Pr[D(Y_\lambda) = 1]|$ is negligible in $\lambda \in \mathbb{N}$. We say that X_λ and Y_λ are statistically indistinguishable if $\sum_{z \in Z_\lambda} |\Pr[X_\lambda = z] - \Pr[Y_\lambda = z]$ is negligible in $\lambda \in \mathbb{N}$.

2.2 DDH Assumption

We review the DDH assumption. Let $\mathcal{G}$ be an algorithm that takes as input a security parameter λ and outputs a tuple $(p, \mathbb{G}, g)$, where p is a prime with $2^{\lambda-1} \leq p < 2^\lambda$, $\mathbb{G}$ is a cyclic group of prime order p, and g is a generator of $\mathbb{G}$.

Assumption 1 (The Decisional Diffie-Hellman Assumption). For any PPT adversary A, the advantage $\mathsf{Adv}_A(k)$ is negligible in the security parameter k.

$$\mathsf{Adv}_A(k) = |\Pr[A((p, \mathbb{G}, g), g^a, g^b, g^{ab}) = 1] - \Pr[A((p, \mathbb{G}, g), g^a, g^b, g^c) = 1]|$$

The probability is over the random choices of $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(\lambda)$, the random choices of $a, b, c \in \mathbb{Z}_p$ and the random coin of A.

2.3 All-But-One Trapdoor Functions

We review all-but-one trapdoor functions to describe our scheme. All-but-one trapdoor functions are made from the DDH assumption[17].

Definition 1 (All-but-one trapdoor functions). A collection of (n, k)-all-but-one trapdoor functions is a tuple of PPT algorithms $(\mathsf{G}_{\mathrm{abo}}, \mathsf{F}_{\mathrm{abo}}, \mathsf{F}_{\mathrm{abo}}^{-1})$ and sequence of branch sets $B = \{B_\lambda\}$ such that:

All-but-one property: Given a lossy branch $b^* \in B_\lambda$, the algorithm $\mathsf{G}_{\mathrm{abo}}(1^\lambda, b^*)$ outputs a pair (s, td). For every $b \in B_\lambda \backslash \{b^*\}$, the algorithm $\mathsf{F}_{\mathrm{abo}}(s, b, \cdot)$ computes an injective function $f_{s,b}(\cdot)$ over $\{0, 1\}^n$, and $\mathsf{F}_{\mathrm{abo}}^{-1}(td, b, \cdot)$ computes $f_{s,b}^{-1}(\cdot)$. For the lossy branch b^*, $\mathsf{F}_{\mathrm{abo}}(s, b^*, \cdot)$ computes a lossy function $f_{s,b^*}(\cdot)$ over $\{0, 1\}^n$, where $|f_{s,b^*}(\{0, 1\}^n)| \leq 2^{n-k}$.

Indistinguishability: For every b_1^* and $b_2^* \in B_\lambda$, the first output s_0 of $\mathsf{G}_{\mathrm{abo}}(1^\lambda, b_0^*)$ and the first output s_1 of $\mathsf{G}_{\mathrm{abo}}(1^\lambda, b_1^*)$ are computationally indistinguishable.

3 Re-applicable Lossy Trapdoor Functions

In this section, we propose a new primitive, *re-applicable LTDFs*, which is an extension of LTDFs. Peikert and Waters proposed *LTDFs* and *all-but-one functions* in STOC'08 [17].

For our purpose, we transform LTDFs in several points. First, we add one algorithm, the *parameter-generation* algorithm ParGen. This algorithm generates public parameters which is common to every algorithm and applied in every evaluation. We introduce ParGen since PRE schemes are used in the multi-user setting, not single-user setting. In addition, the validity checks of ciphertexts require that each ciphertext of the ElGamal encryption in an index of LTDFs has common randomness for every user.

Second, we modify LTDFs so that the function-generation algorithm receives a *tag* in a set of tags $\mathcal{T}$, not injective or lossy commands. For every tag τ, except one special lossy tag τ_{los}, the function-generation algorithm outputs an index that represents an injective function. On the other hand, the function-generation algorithm given τ_{los} outputs an index that represents a lossy function.

Third, we define five new algorithms $\mathsf{ReIndex}, \mathsf{ReEval}, \mathsf{PrivReEval}, \mathsf{Trans}$, and $\mathsf{FakeKey}$. $\mathsf{ReIndex}$, ReEval, $\mathsf{PrivReEval}$, and Trans are deterministic, and $\mathsf{FakeKey}$ is probabilistic. We apply $\mathsf{ReIndex}$ for generating re-encryption keys, ReEval for evaluation of re-encryption, and $\mathsf{PrivReEval}$ for a validity check of ciphertexts. The algorithms Trans and $\mathsf{FakeKey}$ are only used in the proof. The algorithm Trans guarantees the transitivity between re-encryption keys. In other word, we can make a re-encryption key $rk_{j\leftrightarrow k}$ from $rk_{i\leftrightarrow j}$ and $rk_{i\leftrightarrow k}$. The algorithm $\mathsf{FakeKey}$ generates a pair of public and re-encryption keys $(pk_j, rk_{i\leftrightarrow j})$ from another public key pk_i. Moreover, even if pk_i represents a lossy function, $\mathsf{FakeKey}$ always outputs pk_j, which represents an *injective* function. This property is necessary for the last modification in our proof. We introduce $\mathcal{T}$ to provide this property.

We call this new primitive, *a collection of re-applicable LTDFs*. They are specialized LTDFs for PRE. If we unify ParGen and $\mathsf{LossyGen}$, ignore the other new algorithms, and define $\mathcal{T} = \{\tau_{\mathrm{inj}}, \tau_{\mathrm{los}}\}$, we can consider this new primitive as (ordinary) LTDFs proposed by Peikert and Waters.

Definition 2 (Re-applicable LTDFs with respect to function indices). Let (ParGen, $\mathsf{LossyGen}$, $\mathsf{LossyEval}$, $\mathsf{LossyInv}$, $\mathsf{ReIndex}$, ReEval, $\mathsf{PrivReEval}$, Trans, $\mathsf{FakeKey}$) be a tuple of PPT algorithms, and $\mathcal{T}$ be a set of tags that contains one lossy element τ_{los}. The algorithm $\mathsf{ParGen}(1^\lambda)$ outputs a public parameter par. The other algorithms apply the parameter par to their computations. Hereafter, we omit the input of the public parameter par for the algorithms.

A collection of re-applicable (n, k)-lossy trapdoor functions with respect to function indices is a tuple of the PPT algorithms (ParGen, $\mathsf{LossyGen}$, $\mathsf{LossyEval}$, $\mathsf{LossyInv}$, $\mathsf{ReIndex}$, ReEval, $\mathsf{PrivReEval}$, Trans, $\mathsf{FakeKey}$) such that:

Injectivity; For every public parameter $\mathsf{par} \leftarrow \mathsf{ParGen}(1^\lambda)$ and every tag $\tau \in \mathcal{T}\backslash\{\tau_{\mathrm{los}}\}$, $\mathsf{LossyGen}(\tau)$ outputs a pair of a function index and its trapdoor (s, td), $\mathsf{LossyEval}(s, \cdot)$ computes an injective function $f_{s,\tau}(\cdot)$ over $\{0, 1\}^n$, and $\mathsf{LossyInv}(td, \tau, \cdot)$ computes $f_{s,\tau}^{-1}(\cdot)$.

(We represent the function $f_{s,\tau}$, not f_s, in order to clarify a tag τ. If we do not need to clarify a tag, we represent a function as $f_{s,\star}$)

Lossiness: For every public parameter $\mathsf{par} \leftarrow \mathsf{ParGen}(1^\lambda)$, the algorithm $\mathsf{LossyGen}(\tau_{\mathrm{los}})$ outputs $(s, \bot)$ and $\mathsf{LossyEval}(s, \cdot)$ computes a function $f_{s,\tau_{\mathrm{los}}}(\cdot)$ over $\{0,1\}^n$, where $|f_{s,\tau_{\mathrm{los}}}(\{0,1\}^n)| \le 2^{n-k}$.

Indistinguishability between injective and lossy indices: Let X_λ denote the distribution of $(\mathsf{par}, s_{\mathrm{inj}}, \tau)$, and let Y_λ denote the distribution of $(\mathsf{par}, s_{\mathrm{los}}, \tau')$, where par is a public parameter from $\mathsf{ParGen}(1^\lambda)$, τ and τ' are random elements in $\mathcal{T}$, and the function indices s_{inj} and s_{los} are the first element outputs from $\mathsf{LossyGen}(\tau)$ and $\mathsf{LossyGen}(\tau_{\mathrm{los}})$. Then, $\{X_\lambda\}$ and $\{Y_\lambda\}$ are computationally indistinguishable.

Re-applying with respect to function indices: Let τ_i and τ_j be any tags with $\tau_i \neq \tau_{\mathrm{los}}$ and $\tau_j \neq \tau_{\mathrm{los}}$. The algorithm $\mathsf{ReIndex}(td_i, td_j)$ outputs $s_{i\leftrightarrow j}$, where td_i and td_j are the second elements of $\mathsf{LossyGen}(\tau_i)$ and $\mathsf{LossyGen}(\tau_j)$. Then, for every $x \in \{0,1\}^n$, $x = \mathsf{LossyInv}(td_j, \tau_i, \mathsf{ReEval}(s_{i\leftrightarrow j}, \mathsf{LossyEval}(s_i, x)))$. We remark that $\mathsf{LossyInv}$ takes τ_i as one of the inputs, not τ_j.

Generating proper outputs: Let c be an output from $\mathsf{ReEval}(s_{i\leftrightarrow j}, \mathsf{LossyEval}(s_i, x))$, where $s_{i\leftrightarrow j}$ and s_i have the same meaning as that in the above paragraph. Then, $\mathsf{PrivReEval}(x, \tau_i, \tau_j, s_j)$ outputs the same c, where x, τ_i, τ_j, and s_j have the same meaning as that in the above paragraph. That is, $\mathsf{ReEval}(s_{i\leftrightarrow j}, \mathsf{LossyEval}(s_i, \cdot))$ and $\mathsf{PrivReEval}(\cdot, \tau_i, \tau_j, s_j)$ are equivalent as a function (i.e. Any output of $\mathsf{ReEval}(s_{i\leftrightarrow j}, \mathsf{LossyEval}(s_i, \cdot))$ is independent of s_i.).

Transitivity: Let (s_i, td_i), (s_j, td_j) and (s_k, td_k) be outputs from $\mathsf{LossyGen}(\tau_i)$, $\mathsf{LossyGen}(\tau_j)$, and $\mathsf{LossyGen}(\tau_k)$, and let $s_{i\leftrightarrow j}$ and $s_{i\leftrightarrow k}$ be outputs from $\mathsf{ReIndex}(td_i, td_j)$ and $\mathsf{ReIndex}(td_i, td_k)$, respectively. Then, $\mathsf{Trans}(s_{i\leftrightarrow j}, s_{i\leftrightarrow k})$ outputs $s_{j\leftrightarrow k}$ which is the same output from $\mathsf{ReIndex}(td_j, td_k)$.

Statistical indistinguishability of the fake key: The algorithm $\mathsf{FakeKey}(s_i, \tau_i)$ outputs $(s'_j, s'_{i\leftrightarrow j}, \tau'_j)$, where s_i is the first element of an output from $\mathsf{LossyGen}(\tau_i)$. Let X_λ denote the distribution of $(\mathsf{par}, s_i, s_j, s_{i\leftrightarrow j}, \tau_i, \tau_j)$, and let Y_λ denote the distribution of $(\mathsf{par}, s_i, s'_j, s'_{i\leftrightarrow j}, \tau_i, \tau'_j)$, where each par, s_j, $s_{i\leftrightarrow j}$, and τ_j has the same meaning as that in the above paragraph. Then, $\{X_\lambda\}$ and $\{Y_\lambda\}$ are statistically indistinguishable.

Generation of injective functions from lossy functions: Let s be the first element of an output from $\mathsf{FakeKey}(s_{\mathrm{los}}, \tau)$, where τ is a tag and s_{los} is the first element of an output from $\mathsf{LossyGen}(\tau_{\mathrm{los}})$. Then, for every τ, $\mathsf{LossyEval}(s, \cdot)$ represents an injective function $f_{s,\star}$ with overwhelming probability, where a random variable is the randomness of $\mathsf{FakeKey}(s_{\mathrm{los}}, \tau)$. (We do not require other properties of index s if $f_{s,\star}$ is injective. The function $f_{s,\star}$ cannot have any trapdoor information.)

4 Bidirectional and Multi-Hop PRE-CCA Scheme

In this section, we first review the definition of a bidirectional PRE scheme. Then, we describe our scheme and show a sketch of the proof.

A bidirectional PRE scheme consists of six algorithms Π = (Setup, KeyGen, Enc, Dec, $\mathsf{ReKeyGen}$, and ReEnc) as follows. $PP \leftarrow \mathsf{Setup}(1^\lambda)$: Given a security parameter 1^λ, the setup algorithm outputs a public parameter PP. This algorithm is executed by a trusted third party. $(pk, sk) \leftarrow \mathsf{KeyGen}(PP)$: Given a public parameter PP, the key generation algorithm outputs a public key pk and a secret key sk. $C \leftarrow \mathsf{Enc}(PP, pk, m)$: Given a public key pk and a message $m \in \mathcal{M}$, the encryption algorithm outputs a ciphertext C, where $\mathcal{M}$ is a message space. $rk_{i\leftrightarrow j} \leftarrow \mathsf{ReKeyGen}(PP, sk_i, sk_j)$: Given a pair of secret keys sk_i, sk_j, where $i \neq j$, this algorithm outputs a re-encryption key $rk_{i\leftrightarrow j}$. We call $rk_{i\leftrightarrow j}$ the re-encryption key between i and j. $C_j \leftarrow \mathsf{ReEnc}(PP, rk_{i\leftrightarrow j}, C_i)$: Given a re-encryption key $rk_{i\leftrightarrow j}$ between i and j and a ciphertext C_i for i, this algorithm outputs another ciphertext C_j for j or the error symbol $\perp$. $m \leftarrow \mathsf{Dec}(PP, sk, C)$: Given a public key sk and a ciphertext C, the decryption algorithm outputs a message m or the error symbol $\perp$.

If the following two conditions holds, we say that the PRE scheme Π satisfies correctness. For every PP which is output from $\mathsf{Setup}(1^\lambda)$, every (pk, sk) which is output from $\mathsf{KeyGen}(PP)$ and every message $m \in \mathcal{M}$, the probability $\Pr[C \leftarrow \mathsf{Enc}(PP, pk, m) : \mathsf{Dec}(PP, sk, C) = m]$ is overwhelming. For every natural number $n \in \mathbb{N}$, every PP which is output from $\mathsf{Setup}(1^\lambda)$, every $(pk_1, sk_1) \ldots (pk_n, sk_n)$ which are outputs from $\mathsf{KeyGen}(PP)$, every message $m \in M$, and every $rk_{1\leftrightarrow 2} \ldots rk_{n-1\leftrightarrow n}$ which are outputs from $\mathsf{ReKeyGen}(rk_i, rk_{i+1})$ for each $i \in [1, n-1]$, the probability $\Pr[C_1 \leftarrow \mathsf{Enc}(PP, pk_1, m) : \mathsf{Dec}(PP, sk_n, \mathsf{ReEnc}(PP, rk_{n-1\leftrightarrow n}, \ldots \mathsf{ReEnc}(PP, rk_{1\leftrightarrow 2}, C_1) \ldots)) = m]$ is overwhelming.

4.1 Bidirectional and Multi-Hop PRE-CCA Security

We prove that our scheme satisfies the PRE-CCA security in the full version of this paper. This security notion was proposed by Canetti and Hohenberger [6].

Definition 3 (Bidirectional and Multi-Hop PRE-CCA Security). Let λ be the security parameter, A be an oracle TM, representing the adversary, and Γ_U and Γ_C be date structures. Date structures Γ_U and Γ_C are first initialized as empty in the game. The game consists of an execution of A with the following oracles, which can be invoked multiple times in any order, subject to the constraint below:

Setup Oracle: This oracle can be queried first in the game only once. This oracle makes a public parameter as $PP \leftarrow \mathsf{Setup}(1^\lambda)$. A is given PP.

Uncorrupted key generation: This oracle generates a new key pair $(pk, sk) \leftarrow \mathsf{KeyGen}(PP)$ and adds pk in Γ_U, where PP is generated from the setup oracle. A is given pk.

Corrupted key generation: This oracle generates a new key pair $(pk, sk) \leftarrow \mathsf{KeyGen}(PP)$ and adds pk in Γ_C, where PP is generated from the setup oracle. A is given (pk, sk).

Challenge oracle: This oracle can be queried only once. On input (pk^*, m_0, m_1), the oracle chooses a bit $b \leftarrow \{0, 1\}$ and returns $C^* = \mathsf{Enc}(PP, pk^*, m_b)$. We call pk^* the *challenge key* and C^* the *challenge ciphertext*. (We require the challenge key $pk^* \in \Gamma_U$ for A to win.)

Re-encryption key generation: On input (pk_i, pk_j) from the adversary, this oracle return the re-encryption key $rk_{i\leftrightarrow j} = \mathsf{ReKeyGen}(sk_i, sk_j)$, where sk_i and sk_j are the secret keys that correspond to pk_i and pk_j, respectively.

We require that pk_i and pk_j are in Γ_C, or alternatively both are in Γ_U. We do not allow for re-encryption key generation queries between a corrupted key and an uncorrupted key.

Re-encryption oracle: On input (pk_i, pk_j, C_i), if $pk_j \in \Gamma_C$ and (pk_i, C_i) is a *derivative* of (pk^*, C^*), then return a special symbol $\perp$, which is not in the domain of messages or ciphertext. Else, return the re-encrypted ciphertext $C_j = \mathsf{ReEnc}(\mathsf{ReKeyGen}(sk_i, sk_j), C_j)$. Derivatives of (pk^*, C^*) are defined inductively as follows.

- (pk^*, C^*) is a derivative of itself.
- If (pk, C) is a derivative of (pk^*, C^*), and (pk', C') is a derivative of (pk, C), then (pk', C') is a derivative of (pk^*, C^*).
- If A has queried the re-encryption oracle on input (pk, pk', C) and obtained response C', then (pk', C') is a derivative of (pk, C).
- If A has queried the re-encryption key generation oracle on input (pk, pk') or (pk', pk), and $C' = \mathsf{ReEnc}(\mathsf{ReKeyGen}(sk, sk'), C)$, then (pk', C') is a derivative of (pk, C), where sk and sk' are the secret keys that correspond to pk and pk', respectively.

Decryption oracle: On input (pk, C), if the pair (pk, C) is a derivative of the challenge key and ciphertext (pk^*, C^*), or pk is not in $\Gamma_U \cup \Gamma_C$, then return a special symbol $\perp$ which is not in the domain of messages. Else, return $\mathsf{Dec}(sk, C)$, where sk is the secret key that corresponds to pk.

Decision oracle: This oracle can be queried at the end of the game. On input b': If $b' = b$ and the challenge key $pk^* \in \Gamma_U$, then output 1. Else, output 0:

We describe the output of the decision oracle in the above game as $\mathsf{Expt}^{\text{bid-PRE-CCA}}_{\Pi,A}(\lambda) = b$ for an adversary A and a scheme Π. We define the advantage of adversary A as

$$\mathsf{Adv}^{\text{bid-PRE-CCA}}_{\Pi,A}(\lambda) \overset{\text{def}}{=} \left|\Pr[\mathsf{Expt}^{\text{bid-PRE-CCA}}_{\Pi,A}(\lambda) = 1] - \frac{1}{2}\right|,$$

where the probability is over the random choices of A and oracles. We say that the scheme Π is secure under the bidirectional PRE-CCA attack, if, for every adversary A, $\mathsf{Adv}^{\text{bid-PRE-CCA}}_{\Pi,A}(\lambda)$ is negligible in the security parameter λ.

4.2 Description of Our Scheme

We next describe our scheme. Let λ be the security parameter, and let n, k, k', k'' and v be parameters depending on λ. Let $(\mathsf{SigGen}, \mathsf{SigSign}, \mathsf{SigVer})$ be a strongly unforgeable one-time signature scheme where verification keys are in $\{0,1\}^v$. Let (ParGen, $\mathsf{LossyGen}$, $\mathsf{LossyEval}$, $\mathsf{LossyInv}$, $\mathsf{ReIndex}$, ReEval, $\mathsf{PrivReEval}$, Trans, $\mathsf{FakeKey}$) be a collection of re-applicable (n, k)-LTDFs and $\mathcal{T}$ be a set of tags. Let $(\mathsf{G}_{\text{abo}}, \mathsf{F}_{\text{abo}}, \mathsf{F}^{-1}_{\text{abo}})$ be a collection of (n, k')-ABO trapdoor functions with branches $B_\lambda = \{0,1\}^v$, which contains the set of signature verification keys. Let $\mathcal{H}$ be a family of pairwise independent hash functions from $\{0,1\}^n$ to $\{0,1\}^{k''}$. We require that the above parameters are $(k + k') - (k'' + n) \geq \delta = \delta_1 + \delta_2$ for some $\delta_1 = \omega(\log\lambda)$ and $\delta_2 = \omega(\log\lambda)$. Our cryptosystem has message space $\{0,1\}^{k''}$.

The algorithm Setup generates a public parameter $PP = (s_{\text{abo}}, \mathsf{par}, h)$, and the algorithm KeyGen makes a pair of keys $(pk, sk) = ((s_{\text{rltdf}}, \tau), (td_{\text{rltdf}}, s_{\text{rltdf}}, \tau))$. Except a tag τ, we can consider that both Setup and KeyGen are the same algorithm as the key generation algorithm in the Peikert and Waters encryption. The algorithm Enc is also the same algorithm as the encryption algorithm in the Peikert and Waters encryption, except that c_1 is not signed for the re-encryption. The algorithm ReKeyGen makes a re-encryption key, and the ReEnc re-encrypts a ciphertext into another ciphertext. These algorithms only use ReIndex and ReEval in re-applicable LTDFs. The algorithm Dec is the same algorithm as the decryption algorithm in the Peikert and Waters encryption, expect that, if a ciphertext is re-encrypted, it applies PrivReEval for the validity check of ciphertexts.

Setup(1^λ): Setup(1^λ) first generates an index of all-but-one trapdoor functions with lossy branch 0^v: $(s_{\text{abo}}, td_{\text{abo}}) \leftarrow \mathsf{G}_{\text{abo}}(1^\lambda, 0^v)$. Then, it generates a public parameter of re-applicable LTDFs: $\mathsf{par} \leftarrow \mathsf{ParGen}(1^\lambda)$. Finally, it chooses a hash function $h \leftarrow \mathcal{H}$. It outputs a public parameter as $PP = (s_{\text{abo}}, \mathsf{par}, h)$.

(The algorithm Setup erases the trapdoor td_{abo} because the following algorithms do not use td_{abo}.)

KeyGen(PP): KeyGen takes $PP = (s_{\text{abo}}, \mathsf{par}, h)$ as input. It chooses a tag $\tau \in \mathcal{T} \backslash \{\tau_{\text{los}}\}$ and generates an injective index of re-applicable LTDFs: $(s_{\text{rltdf}}, td_{\text{rltdf}}) \leftarrow \mathsf{LossyGen}(\tau)$. A public key consists of the injective function index and the tag, and a secret key consisting of the trapdoor of s_{rltdf} and the tag: $pk = (s_{\text{rltdf}}, \tau)$, and $sk = (td_{\text{rltdf}}, s_{\text{rltdf}}, \tau)$.

Enc(PP, pk, m): Enc takes (PP, pk, m) as input, where $PP = (s_{\text{abo}}, \mathsf{par}, h)$ is a tuple of public parameters, $pk = (s_{\text{rltdf}}, \tau)$ is a public key, and $m \in \{0,1\}^\ell$ is a message. It chooses $x \in \{0,1\}^n$ uniformly at random. It generates a key-pair for the one-time signature scheme: $(vk, sk_\sigma) \leftarrow \mathsf{SigGen}(1^\lambda)$, then computes

$$c_1 = \mathsf{LossyEval}(s_{\text{rltdf}}, x),\ c_2 = \mathsf{F}_{\text{abo}}(s_{\text{abo}}, vk, x),\ \text{and } c_3 = h(x) \oplus m.$$

Finally, it signs a tuple (c_2, c_3, τ) as $\sigma \leftarrow \mathsf{SigSign}(sk_\sigma, (c_2, c_3, \tau))$. Then, a ciphertext C is output as $C = (vk, c_1, c_2, c_3, \tau, \sigma)$.

ReKeyGen(PP, sk_i, sk_j) : ReKeyGen takes as input (PP, sk_i, sk_j), where $(sk_i, sk_j) = ((td_i, s_i, \tau_i), (td_j, s_j, \tau_j))$. It computes $s_{i \leftrightarrow j} \leftarrow \mathsf{ReIndex}(td_i, td_j)$, then outputs a re-encryption key as $rk_{i \leftrightarrow j} = s_{i \leftrightarrow j}$.

ReEnc($PP, rk_{i \leftrightarrow j}, C_i$) : ReEnc takes $(rk_{i \leftrightarrow j}, C_i)$ as input, where $rk_{i \leftrightarrow j} = s_{i \leftrightarrow j}$ is a re-encryption key and $C_i = (vk, c_{1,i}, c_2, c_3, \tau, \sigma)$ is a ciphertext. It computes $c_{1,j} \leftarrow \mathsf{ReEval}(s_{i \leftrightarrow j}, c_{1,i})$. It then outputs $C_j = (vk, c_{1,j}, c_2, c_3, \tau, \sigma)$ as a new ciphertext for the user with sk_j.

Dec(PP, sk, C) : Dec takes (PP, sk, C) as input, where $PP = (s_{\text{abo}}, \mathsf{par}, h)$ is a tuple of public parameters, $sk = (td_{\text{rltdf}}, s_{\text{rltdf}}, \tau)$ is a secret key, and $C = (vk, c_1, c_2, c_3, \tau', \sigma)$ is a ciphertext. It first checks $\mathsf{SigVer}(vk, (c_2, c_3, \tau'), \sigma) = 1$; if not, it outputs $\perp$.

It then compute $x = \mathsf{LossyInv}(td_{\text{rltdf}}, \tau', c_1)$. If $\tau = \tau'$ then it checks $c_1 = \mathsf{LossyEval}(s_{\text{rltdf}}, x)$, otherwise, it checks $\mathsf{PrivReEval}(x, \tau', \tau, s_{\text{rltdf}}) = c_1$; if not, it outputs $\perp$. It also checks $c_2 = \mathsf{F}_{\text{abo}}(s_{\text{abo}}, vk, x)$; if not, it outputs $\perp$. Finally, it outputs $m = c_3 \oplus h(x)$. (We note that, if C was not re-encrypted, then $\tau = \tau'$. On the other hand, if C was re-encrypted, $\tau \neq \tau'$.)

4.3 Security of Our Scheme

In this section, we claim the following theorem and describe a sketch of the proof. We give the detail proof in the full version of this paper.

Theorem 1. The above proposed scheme satisfies the PRE-CCA security.

We now show modifications of games from Game_0 to Game_{10} to prove Theorem 1. Game_0 is identical to the PRE-CCA game. In Game_{10}, no adversary can win with meaningful probability. Every modification from Game_i to Game_{i+1} is perfect, statistically or computationally indistinguishable for each $i \in [0, n-1]$. Therefore, we conclude that no adversary also can win with meaningful probability in Game_0.

In every game, let $C^* = (vk^*, c_1^*, c_2^*, c_3^*, \tau^*, \sigma^*)$ and $pk^* = (s^*_{\mathrm{rltdf}}, \tau^*)$ be the challenge ciphertext and the challenge public key, respectively.

Game_0**:** This game is identical to the PRE-CCA game.

Game_1**:** Let x^* denote a random input applied to making the challenge ciphertext. (i.e. $c_1^* = \mathsf{LossyEval}(s^*_{\mathrm{rltdf}}, x^*)$, $c_2^* = \mathsf{F}_{\mathrm{abo}}(s_{\mathrm{abo}}, vk^*, x^*)$, $c_3^* = h(x^*) \oplus m_b$.)

Then, we modify the decryption oracle as follows: The decryption oracle is given a decryption query $(pk, C) = ((s, \tau), (vk, c_1, c_2, c_3, \tau', \sigma))$, where pk is limited to an output by the corrupted or the uncorrupted oracles. If (pk, C) is a derivative of (pk^*, C^*), then it outputs $\perp$. Else if the decryption query satisfies that $(vk, c_2, c_3, \tau', \sigma) = (vk^*, c_2^*, c_3^*, \tau^*, \sigma^*)$ and $\mathsf{PrivReEval}(x^*, \tau', \tau, s) = c_1$, then it outputs $m \leftarrow c_3 \oplus h(x^*)$. Otherwise, it outputs $\mathsf{Dec}(sk, C)$ in the ordinary decryption processes.

This modification does not affect any success probability of an adversary. From the injectivity of $\mathsf{PrivReEval}(\cdot, \tau', \tau, s)$, if $\mathsf{PrivReEval}(x^*, \tau', \tau, s) = c_1$, then we obtain $\mathsf{LossyInv}(td, \tau', c_1) = x^*$. In fact, the probability that the above queries satisfy $\mathsf{PrivReEval}(x^*, \tau', \tau, s) = c_1$ is negligible. We discuss this fact in the modification between Game_9 and Game_{10}. However, this check is necessary for the following modifications.

Game_2**:** We add the following check to the decryption oracle after checking a derivative: The decryption oracle is given a decryption query $(pk, C) = ((s, \tau), (vk, c_1, c_2, c_3, \tau', \sigma))$. The decryption oracle always outputs $\perp$, if $vk = vk^*$ and $(c_2, c_3, \tau', \sigma) \neq (c_2^*, c_3^*, \tau^*, \sigma^*)$.

This modification is negligible for the success probability of an adversary from the strongly existential unforgeability of the signature scheme.

Game_3**:** Let vk^* be the verification key used by the challenge oracle. We modify the setup oracle on a lossy branch as follows: We replace generates $(s_{\mathrm{abo}}, td_{\mathrm{abo}}) \leftarrow \mathsf{G}_{\mathrm{abo}}(1^\lambda, 0^\nu)$ with $(s_{\mathrm{abo}}, td_{\mathrm{abo}}) \leftarrow \mathsf{G}_{\mathrm{abo}}(1^\lambda, vk^*)$. The lossy branch is changed the verification key vk^* from a zero-padding 0^ν.

This modification is negligible for the success probability of an adversary from the computational indistinguishability of all-but-one trapdoor functions.

Game_4: Let $(s_{\text{abo}}, td_{\text{abo}})$ be an index of all-but-one trapdoor functions and its trapdoor. We modify the decryption oracle as follows: The decryption oracle uses the trapdoor td_{abo} to decrypt ciphertext, when it receives a ciphertext $C = (vk, c_1, c_2, c_3, \tau', \sigma)$. That is, in the case of $(vk, c_2, c_3, \tau', \sigma) \neq (vk^* c_2^*, c_3^*, \tau^*, \sigma^*)$ and $vk \neq vk^*$, we replace $x \leftarrow \mathsf{LossyInv}(td_{\text{rltdf}}, \tau', c_1)$ with $x \leftarrow \mathsf{F}_{\text{abo}}^{-1}(td_{\text{abo}}, vk, c_2)$, and proceed with the decryption processes (In the other cases, the decryption oracle executes the operations defined in Game_1 and Game_2.).

This modification does not affect any success probability of an adversary because of the injectivity of $\mathsf{LossyEval}(s, \cdot)$, $\mathsf{F}_{\text{abo}}(s_{\text{abo}}, vk, \cdot)$, and $\mathsf{PrivReEval}(\cdot, \tau', \tau, s)$. In this modification, $\mathsf{F}_{\text{abo}}(s_{\text{abo}}, vk, \cdot)$ is an injective function since vk is not the lossy branch vk^*.

In the following games, the challenger manages uncorrupted re-encryption keys to apply the table. That is, in the first of this game, the challenger makes an empty table, which memorizes uncorrupted re-encryption keys. We set that a pair of keys (pk_1, sk_1) is the first output by the uncorrupted oracle. At n-th query, the uncorrupted oracle outputs a pair of keys (pk_n, sk_n) and makes the re-encryption keys $rk_{1\leftrightarrow n}, \cdots, rk_{n-1\leftrightarrow n}$ and $rk_{n\leftrightarrow 1}, \cdots, rk_{n\leftrightarrow n-1}$. The challenger adds re-encryption keys to the table.

Game_5: We modify the re-encryption oracle as follows: The re-encryption oracle takes as query $(pk_a, pk_b, C_a) = ((s_a, \tau_a), (s_b, \tau_b), (vk, c_{1,a}, c_2, c_3, \tau'_a, \sigma))$ in the game. If pk_a is corrupted, then it evaluates $x \leftarrow \mathsf{LossyInv}(td_a, \tau'_a, c_{1,a})$. Then, it makes $c_{1,b} = \mathsf{PrivReEval}(x, \tau_a, \tau_b, s_b)$. If pk_b is uncorrupted, it searches $rk_{a\leftrightarrow b}$ in the re-encryption keys table and evaluates $c_{1,b} \leftarrow \mathsf{ReEnc}(rk_{a\leftrightarrow b}, c_{1,a})$. Then, it outputs $C_b = (vk, c_{1,b}, c_2, c_3, \tau'_a, \sigma)$ as a re-encrypted ciphertext for pk_b.

This modification does not affect any success probability of an adversary because of the equivalence between $\mathsf{PrivReEval}(\cdot, \tau_a, \tau_b, s_b)$ and $\mathsf{ReEnc}(rk_{a\leftrightarrow b}, \mathsf{LossyEval}(s_a., \cdot))$.

Game_6: We modify the re-encryption key generation oracle as follows: Given a pair (pk_a, pk_b), it searches the re-encryption keys $rk_{a\leftrightarrow b}$ from the table. Then, it outputs this re-encryption key $rk_{a\leftrightarrow b}$.

This modification does not affect any success probability of an adversary.

Game_7: We define the number $q_{A,\text{unc}}$ as the maximum number of times that an adversary A queries the uncorrupted oracle in the game. We modify the challenge oracle as follows.

First, the challenger chooses a random number $r \in \{1, \ldots, q_{A,\text{unc}}\}$. If the challenge oracle receives the challenge key $pk^* \neq pk_r$, the challenger outputs a random bit b and aborts this game, where pk_r is the r-th public key output by the uncorrupted oracle. Otherwise, it proceeds with this game.

This modification reduces the success probability of an adversary to $1/q_{A,\text{unc}}$ fraction. However, this is not an important reduction since $q_{A,\text{unc}}$ is polynomial of the security parameter λ.

Game_8: We modify the uncorrupted key generation oracle as follows.

First, it executes the following preprocessing. It choose a random number $r \in \{1, \ldots, q_{A,\text{unc}}\}$ and generates a pair of keys $(pk_r, sk_r) \leftarrow \mathsf{KeyGen}(PP)$. We describe

$pk_r = (s_r, \tau_r)$ and $sk_r = (td_r, s_r, \tau_r)$. Then, for every $i \in \{1, \ldots, q_{A,\mathrm{unc}}\} \setminus \{r\}$, it uses the fake key generation algorithm to generate a public key and the re-encryption key: it computes $(s_i, s_{r \leftrightarrow i}, \tau_i) \leftarrow \mathsf{FakeKey}(s_r, \tau_r)$ and sets $pk = (s_i, \tau_i)$, $rk_{r \leftrightarrow i}$. Then, it computes that $rk_{i \leftrightarrow j} = s_{i \leftrightarrow j} \leftarrow \mathsf{Trans}(s_{r \leftrightarrow i}, s_{r \leftrightarrow j})$ for every $i, j \in \{1, \ldots, q_{A,\mathrm{unc}}\}$ and $i \neq j$. It adds every above-mentioned re-encryption key $\{rk_{i \leftrightarrow j}\}_{i,j \in \{1,\ldots,q_{A,\mathrm{unc}}\}, i \neq j}$ to the re-encryption keys table.

At $j \in \{1, \ldots, q_{A,unc}\}$ times query for the uncorrupted oracle, it outputs the public key $pk_j = s_j$ generated by the above preprocessing. The re-encryption key oracle and the re-encryption oracle execute their processes with the preprocessed table. The challenge oracle applies the above number r to the check $pk^* = pk_r$.

This modification is negligible for the success probability of an adversary from the statistical indistinguishability of the fake key algorithm.

Game_9**:** We modify the above preprocessing as follows: We replace the first key generation $pk_r = s_r \leftarrow \mathsf{LossyGen}(\tau_r)$ with $pk_r = s_r \leftarrow \mathsf{LossyGen}(\tau_{\mathrm{los}})$, where τ_{los} is a lossy tag.

This modification is negligible for the success probability of an adversary from the computational indistinguishability of LTDFs.

Game_{10}**:** We modify the decryption oracle as follows: The decryption oracle always outputs $\perp$, when it receives the query $C = (vk^*, c_1, c_2^*, c_3^*, \sigma^*)$ and $pk = s_{\mathrm{rltdf}}$.

This modification is negligible for the success probability of an adversary from the fact that the average-case min-entropy of x^* is high, and every public key $pk = s$, except a challenge key, represents an injective function $f_{s,\star}$. That is, adversary never compute c_1 such that $f_{s,\star}^{-1}(c_1) = x^*$ information-theoretically. This fact implies that Game_{10} is statistically close to Game_9. Due to this modification, we attach tags to re-applicable LTDFs.

From the above sketch, we transform the PRE-CCA game (i.e. Game_0) into the last game (i.e. Game_{10}). In the last game, Game_{10}, we conclude that any adversary does not detect which message is encrypted. The reason is, $h(x^*)$ is statistically close to U_ℓ since x^* has high average-case min-entropy and $h(\cdot)$ can extracts a random string from x^*.

5 Realization of Re-applicable LTDFs Based on DDH Assumption

In this section, we describe the realization of re-applicable LTDFs from the DDH assumption. We modify the construction proposed by Peikert and Waters.

We now describe LTDFs based on the DDH assumption. We modify the construction as the proposed by Peikert and Waters on two points. One is a division of one function generation algorithm into two algorithms. The other is a change on an encrypting matrix in the injective function generator. Peikert and Waters proposed the function-generation algorithm which creates a function index as a ciphertext of a matrix. This ciphertext is encrypted with the ElGamal encryption on matrices. When generating an injective-function index, it encrypts the matrix $\boldsymbol{I} = (g^{\delta_{i,j}})_{i,j}$ on $\mathbb{G}^{n \times n}$, where $\delta_{i,j}$ is the Kronecker delta. When generating a lossy-function index, it encrypts the matrix $\boldsymbol{0} = (e)_{i,j}$ on $\mathbb{G}^{n \times n}$, where e is the identity in $\mathbb{G}$ (i.e. $e = g^0$). We also do the same for generating a lossy-function index, but we do another procedure for generating an injective-function index. We define a set of tags $\mathcal{T}$ as a group $\mathbb{G}$ and a special element τ_{los} as the identity $e \in \mathbb{G}$. We use a matrix $\boldsymbol{M} = (\tau^{\delta_{i,j}})_{i,j}$ on $\mathbb{G}^{n \times n}$, where τ is any element in $\mathbb{G}$. When generating a injective-function index, we set τ with $\tau \neq \tau_{\mathrm{los}}$. When generating a lossy-function index, we set $\tau = \tau_{\mathrm{los}}$.

– *Generation of a public parameter.* A parameter generator ParGen first executes $\mathcal{G}$, and $\mathcal{G}$ outputs a tuple $(p, \mathbb{G}, g)$. It next selects random numbers $r_1, \ldots, r_n \in_R \mathbb{Z}_p$, then makes a public parameter $\boldsymbol{C}_1$ as

$$\boldsymbol{C}_1 = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} g^{r_1} \\ \vdots \\ g^{r_n} \end{pmatrix}.$$

– *Generation of function indices.* A function generator LossyGen takes as input $\boldsymbol{C}_1$ and a tag τ, where $\boldsymbol{C}_1$ is the public parameter and τ is an element in $\mathbb{G}$. (We note that if $\tau = e$, it means execution of the lossy mode, otherwise, execution of the injective mode.) It first selects random elements $z_1, z_2, \ldots, z_n \in_R \mathbb{Z}_p$, then computes a function index as

$$\boldsymbol{C}_2 = \begin{pmatrix} c_{1,1} & \cdots & c_{1,n} \\ \vdots & \ddots & \vdots \\ c_{n,1} & \cdots & c_{n,n} \end{pmatrix} = \begin{pmatrix} c_1^{z_1} \cdot \tau & \cdots & c_1^{z_n} \\ \vdots & \ddots & \vdots \\ c_n^{z_1} & \cdots & c_n^{z_n} \cdot \tau \end{pmatrix} = \begin{cases} c_{i,j} = c_i^{z_j} \cdot \tau & \text{if } i = j, \\ c_{i,j} = c_i^{z_j} & \text{otherwise.} \end{cases}$$

The function index consists of $(\boldsymbol{C}_1, \boldsymbol{C}_2)$. A trapdoor consists of the random elements $z = (z_1, \ldots, z_n)$.

– *Evaluation algorithm.* An evaluation algorithm LossyEval takes as input $(\boldsymbol{C}_1, \boldsymbol{C}_2, \boldsymbol{x})$, where $(\boldsymbol{C}_1, \boldsymbol{C}_2)$ is a function index, and $\boldsymbol{x} = (x_1, \ldots, x_n) \in \{0, 1\}^n$ is an n-bit input interpreted as a vector.

It evaluates the linear product of $\boldsymbol{x}$ and $\boldsymbol{C}_1$: That is, $y_1 = \boldsymbol{x}\boldsymbol{C}_1 = \prod_{i=1}^n (c_i)^{x_i}$.Next, it evaluates the product of $\boldsymbol{x}$ and $\boldsymbol{C}_2$: That is, $\boldsymbol{y}_2 = \boldsymbol{x}\boldsymbol{C}_2 = (\prod_{i=1}^n c_{i,1}^{x_i}, \cdots, \prod_{i=1}^n c_{i,n}^{x_i}) = ((\prod_{i=1}^n c_i^{z_1 x_i})\tau^{x_1}, \cdots, (\prod_{i=1}^n c_i^{z_n x_i})\tau^{x_n})$. Finally, it outputs $(y_1, \boldsymbol{y}_2)$.

– *Inversion algorithm.* An inversion algorithm LossyInv takes as input $(td, \tau, (y_1, \boldsymbol{y}_2))$, where trapdoor information td consists of $z = (z_1, \ldots, z_n)$, τ, which is an element in $\mathbb{G}\backslash\{e\}$, and $\boldsymbol{y}_2 = (y_{2,1}, \cdots, y_{2,n}) \in \mathbb{G}^{1 \times n}$. It computes that $\boldsymbol{w} = (y_{2,1} \cdot y_1^{-z_1}, y_{2,2} \cdot y_1^{-z_2}, \cdots, y_{2,n} \cdot y_1^{-z_n})$. Then, if j-th element of $\boldsymbol{w}$ is the identity element of $\mathbb{G}$, then it sets $x_j = 0$, else if j-th element of $\boldsymbol{w}$ is τ then it sets $x_j = 1$; otherwise, it output $\perp$. Finally, it outputs $x = (x_1, x_2, \ldots, x_n)$.

We can show that the above four algorithms satisfy injectivity, $(n, n - \log p)$-lossiness, and indistinguishability based on the DDH assumption. These proofs are nearly similar to the proof of Peikert and Waters [17]; therefore, we omit them in this paper.

Next, we show that the above algorithms satisfy the definition of re-applicable LTDFs.

Re-applying with respect to function indices: Let τ_i and τ_j be tags different from τ_{los} in $\mathcal{T}$. Let (s_i, td_i) and (s_j, td_j) be outputs from $\mathsf{LossyGen}(\tau_i)$ and $\mathsf{LossyGen}(\tau_j)$. We now define an algorithm $\mathsf{ReIndex}$, which takes td_i and td_j as inputs and outputs $s_{i\leftrightarrow j} = td_j - td_i = (z'_1 - z_1, z'_2 - z_2, \ldots, z'_n - z_n) = (z_{1,i\leftrightarrow j}, \ldots, z_{n,i\leftrightarrow j})$.

We next define an algorithm ReEval. The algorithm ReEval takes as input $(s_{i\leftrightarrow j}, (y_1, \boldsymbol{y}_2))$, where $(y_1, \boldsymbol{y}_2) = (y_1, (y_{2,1}, y_{2,2}, \cdots, y_{2,n}))$ is an output from $\mathsf{LossyEval}(s_i, \boldsymbol{x})$. It computes that $\boldsymbol{y}'_2 = (y'_{2,1}, y'_{2,2}, \ldots, y'_{2,n}) = (y_{2,1} \cdot y_1^{z_{1,i\leftrightarrow j}}, y_{2,2} \cdot y_1^{z_{2,i\leftrightarrow j}}, \cdots, y_{2,n} \cdot y_1^{z_{n,i\leftrightarrow j}})$. Then, it outputs $(y_1, \boldsymbol{y}'_2)$.

For the above elements, we can describe that $(y_1, \boldsymbol{y}_2) = (g^{\sum_{i=1}^n x_i r_i}, (g^{z_1 \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_1}, \cdots, g^{z_n \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_n}))$ and $(y_1, \boldsymbol{y}'_2) = (g^{\sum_{i=1}^n x_i r_i}, (g^{z'_1 \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_1}, \cdots, g^{z'_n \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_n}))$. Therefore, we have $\boldsymbol{w}' = (\tau_i^{x_1}, \cdots, \tau_i^{x_n})$ in the algorithm $\mathsf{LossyInv}(td_j, \tau_i, (y_1, \boldsymbol{y}'_2))$. That is, we obtain that $\boldsymbol{x} = \mathsf{LossyInv}(td_j, \tau_i, (y_1, \boldsymbol{y}'_2))$.

Generating proper outputs: Let $\tau_i, \tau_j, (s_i, td_i), (s_j, td_j), s_{i\leftrightarrow j}$ $\boldsymbol{x}$, and $(y_1, \boldsymbol{y}'_2)$ be defined similarly to the above paragraph. We call $(y_1, \boldsymbol{y}'_2)$ a *proper* output for $\boldsymbol{x}$, τ_i, τ_j, and s_j, if they satisfy $(y_1, \boldsymbol{y}'_2) = \mathsf{ReEval}(s_{i\leftrightarrow j}, \mathsf{LossyEval}(s_i, \boldsymbol{x}))$ for some s_i made from a tag τ_i and $s_{i\leftarrow j}$. We can uniquely describe a proper $(y_1, \boldsymbol{y}'_2)$ as $(g^{\sum_{i=1}^n x_i r_i}, (g^{z'_1 \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_1}, \cdots, g^{z'_n \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_n}))$ from the above algorithms, where $td_j = (z'_1, \cdots, z'_n)$.

We define a new algorithm $\mathsf{PrivReEval}$, which takes $\boldsymbol{x}$, τ_i, τ_j, and s_j as input, where $\boldsymbol{x} = (x_1, \cdots, x_n)$ is n-bits input. It computes $(\hat{y}_1, \hat{\boldsymbol{y}}_2) \leftarrow \mathsf{LossyEval}(s_j, \boldsymbol{x})$. It makes $\hat{\boldsymbol{y}}'_2$ from $\hat{\boldsymbol{y}}_2$ in the following process: for each $i \in [1, n]$, if $x_i = 1$ then $\hat{y}'_{2,i} \leftarrow \hat{y}_{2,i} \tau_j^{-1} \tau_i$, else $\hat{y}_{2,i} \leftarrow \hat{y}_{2,i}$, where $\hat{y}_{2,i}$ and $\hat{y}'_{2,i}$ are the i-th elements of $\hat{\boldsymbol{y}}_2$ and $\hat{\boldsymbol{y}}'_2$. Finally it outputs $(\hat{y}_1, \hat{y}'_2)$. The algorithm $\mathsf{PrivReEval}(\boldsymbol{x}, \tau_i, \tau_j, s_j)$ always computes a proper output for $\boldsymbol{x}$, τ_i, τ_j, and s_j. The reason is that, from an output $(\hat{y}_1, \hat{\boldsymbol{y}}_2) = (g^{\sum_{i=1}^n x_i r_i}, (g^{z'_1 \sum_{i=1}^n x_i r_i} \cdot \tau_j^{x_1}, \cdots, g^{z'_n \sum_{i=1}^n x_i r_i} \cdot \tau_j^{x_n}))$, we verify $(y_1, \hat{\boldsymbol{y}}'_2) = (g^{\sum_{i=1}^n x_i r_i}, (g^{z'_1 \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_1}, \cdots, g^{z'_n \sum_{i=1}^n x_i r_i} \cdot \tau_i^{x_n}))$ that τ_j is replaced with τ_i. This means that $\mathsf{PrivReEval}(\cdot, \tau_i, \tau_j, s_j)$ is equivalent to $\mathsf{ReEval}(s_{i\leftrightarrow j}, \mathsf{LossyEval}(s_i, \cdot))$ as a function.

Transitivity: We define an algorithm Trans, which takes $s_{i\leftrightarrow j}$, $s_{i\leftrightarrow k}$ and outputs $s_{i\leftrightarrow k} - s_{i\leftrightarrow j} = (td_k - td_i) - (td_j - td_i) = td_k - td_j = s_{j\leftrightarrow k}$. That is, $\mathsf{Trans}(s_{i\leftrightarrow j}, s_{i\leftrightarrow k}) \rightarrow s_{j\leftrightarrow k}$.

Statistical indistinguishability of the fake key: Now, we define an algorithm $\mathsf{FakeKey}$, which takes a function index s_i and a tag τ_i and makes a fake index s_j, a fake re-key $s_{i\leftrightarrow j}$, and a fake tag τ_j. The fake key generator $\mathsf{FakeKey}$ takes as input $s_i = (\boldsymbol{C}_1, \boldsymbol{C}_2)$ and $\tau_i \in \mathbb{G}$, where $(\boldsymbol{C}_1, \boldsymbol{C}_2)$ is a function index. It then selects a random element $t \in \mathbb{G}$. It next chooses a random number $s_{i\leftrightarrow j} = (z_{1,i\leftrightarrow j}, \ldots, z_{n,i\leftrightarrow j}) \in_R \mathbb{Z}_p^n$, and makes a new matrix $\boldsymbol{C}'_2$ as follows.

$$\boldsymbol{C}'_2 = \begin{pmatrix} c_{1,1} \cdot c_1^{z_{1,i\leftrightarrow j}} \cdot t & \cdots & c_{1,n} \cdot c_1^{z_{n,i\leftrightarrow j}} \\ \vdots & \ddots & \vdots \\ c_{n,1} \cdot c_n^{z_{1,i\leftrightarrow j}} & \cdots & c_{n,n} \cdot c_n^{z_{n,i\leftrightarrow j}} \cdot t \end{pmatrix} = \begin{cases} c'_{k,\ell} = c_{k,\ell} \cdot c_k^{z_{l,i\leftrightarrow j}} \cdot t & \text{if } k = \ell, \\ c'_{k,\ell} = c_{k,\ell} \cdot c_k^{z_{l,i\leftrightarrow j}} & \text{otherwise,} \end{cases}$$

where c_k is the k entry of $\boldsymbol{C}_1$, and $c_{k,l}$ is the (k, l) entry of $\boldsymbol{C}_2$. Finally, it output $s_j = (\boldsymbol{C}_1, \boldsymbol{C}'_2)$, $s_{i\leftrightarrow j} = (z_{1,i\leftrightarrow j}, \ldots, z_{n,i\leftrightarrow j})$, and $\tau_j = \tau_i \cdot t$.

From the abode description, outputs of $\mathsf{FakeKey}(s_i, \tau_i)$ and the proper index have the same distribution. The reason is that, when s_i and τ_i is made from the proper way, we can describe $\boldsymbol{C}_2'$ by

$$\boldsymbol{C}_2' = \begin{pmatrix} g^{r_1(z_1+z_{1,i\leftrightarrow j})} \cdot \tau_i \cdot t & \cdots & g^{r_1(z_n+z_{n,i\leftrightarrow j})} \\ \vdots & \ddots & \vdots \\ g^{r_n(z_1+z_{1,i\leftrightarrow j})} & \cdots & g^{r_n(z_n+z_{n,i\leftrightarrow j})} \cdot \tau_i \cdot t \end{pmatrix} = \begin{cases} c'_{k,\ell} = g^{r_k(z_\ell+z_{\ell,i\leftrightarrow j})} \cdot \tau_i \cdot t & \text{if } k = \ell, \\ c'_{k,\ell} = g^{r_k(z_\ell+z_{\ell,i\leftrightarrow j})} & \text{otherwise,} \end{cases}$$

where $(z_1, \cdots, z_n)$ is the trapdoor of s_i. This means that, conditioned on $t \neq (\tau_i)^{-1}$, the distribution of $\{\tau_i, \tau_j, (s_i, td_j), (s_j, td_j), s_{i\leftrightarrow j}\}$ is identical to $\{\tau_i, \tau_j, \mathsf{LossyGen}(\tau_i), \mathsf{LossyGen}(\tau_j), \mathsf{ReIndex}(td_i, td_j)\}$. That is, distributions between them are statistically indistinguishable since the probability of $t = (\tau_i)^{-1}$ is $1/p$.

Generation of injective functions from lossy functions: Next, we consider FakeKey which $s_i = (\boldsymbol{C}_1, \boldsymbol{C}_2)$ is output from $\mathsf{LossyGen}(\tau_{\mathrm{los}})$. In this case, we can describe $\boldsymbol{C}_2'$ as

$$\boldsymbol{C}_2' = \begin{pmatrix} g^{r_1(z_1+z_{1,i\leftrightarrow j})} \cdot t & \cdots & g^{r_1(z_n+z_{n,i\leftrightarrow j})} \\ \vdots & \ddots & \vdots \\ g^{r_n(z_1+z_{1,i\leftrightarrow j})} & \cdots & g^{r_n(z_n+z_{n,i\leftrightarrow j})} \cdot \cdot t \end{pmatrix},$$

where $(z_1, \cdots, z_n)$ are logarithms between $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$. Then, assuming that $t \neq e$, $\mathsf{LossyEval}(s_j, \cdot)$ represents an injective function $f_{s_j,t}$. This probability is $1 - 1/p$ which is overwhelming in the security parameter λ.

References

1. Ateniese, G., Benson, K., Hohenberger, S.: Key-Private Proxy Re-encryption. In: Fischlin, M. (ed.) RSA Conference 2009. LNCS, vol. 5473, pp. 279–294. Springer, Heidelberg (2009)
2. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage. In: Network and Distributed System Security Symposium, NDSS. The Internet Society (2005)
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. Blaze, M., Bleumer, G., Strauss, M.: Divertible Protocols and Atomic Proxy Cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998)
5. Boldyreva, A., Fehr, S., O'Neill, A.: On Notions of Security for Deterministic Encryption, and Efficient Constructions without Random Oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 335–359. Springer, Heidelberg (2008)
6. Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption. In: Ning, P., di Vimercati, S.D.C., Syverson, P.F. (eds.) ACM Conference on Computer and Communications Security, Alexandria, Virginia, USA, Octorber 2007, pp. 185–194. ACM, New York (2007)
7. Deng, R.H., Weng, J., Liu, S., Chen, K.: Chosen-Ciphertext Secure Proxy Re-encryption without Pairings. In: Franklin, M.K., Hui, L.C.K., Wong, D.S. (eds.) CANS 2008. LNCS, vol. 5339, pp. 1–17. Springer, Heidelberg (2008)
8. Freeman, D.M., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: More Constructions of Lossy and Correlation-Secure Trapdoor Functions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 279–295. Springer, Heidelberg (2010)

9. Hohenberger, S., Rothblum, G.N., Shelat, A., Vaikuntanathan, V.: Securely Obfuscating Re-encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 233–252. Springer, Heidelberg (2007)
10. Ivan, A., Dodis, Y.: Proxy Cryptography Revisited. In: NDSS, The Internet Society (2003)
11. Khurana, H., Heo, J., Pant, M.: From Proxy Encryption Primitives to a Deployable Secure-Mailing-List Solution. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, pp. 260–281. Springer, Heidelberg (2006)
12. Libert, B., Vergnaud, D.: Unidirectional Chosen-Ciphertext Secure Proxy Re-encryption. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 360–379. Springer, Heidelberg (2008)
13. Mambo, M., Okamoto, E.: Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts. IEICE transactions on fundamentals of electronics, Communications and computer sciences 80(1), 54–63 (1997)
14. Mol, P., Yilek, S.: Chosen-Ciphertext Security from Slightly Lossy Trapdoor Functions. In: PKC (2010)
15. Naor, M., Yung, M.: Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In: STOC, New Orleans, Louisiana, USA, May 1990, pp. 427–437. ACM, New York (1990)
16. Nishimaki, R., Fujisaki, E., Tanaka, K.: Efficient Non-interactive Universally Composable String-Commitment Schemes. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 3–18. Springer, Heidelberg (2009)
17. Peikert, C., Waters, B.: Lossy Trapdoor Functions and Their Applications. In: Ladner, R.E., Dwork, C. (eds.) STOC, Victoria, British Columbia, Canada, May 2008, pp. 187–196. ACM, New York (2008)
18. Rosen, A., Segev, G.: Efficient Lossy Trapdoor Functions based on the Composite Residuosity Assumption. Cryptology ePrint Archive, Report 2008/134 (2008), http://eprint.iacr.org/
19. Rosen, A., Segev, G.: Chosen-Ciphertext Security via Correlated Products. In: Reingold, O. (ed.) Theory of Cryptography. LNCS, vol. 5444, pp. 419–436. Springer, Heidelberg (2009)
20. Shao, J., Cao, Z.: CCA-Secure Proxy Re-encryption without Pairings. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 357–376. Springer, Heidelberg (2009)
21. Taban, G., Cárdenas, A.A., Gligor, V.D.: Towards a secure and interoperable DRM architecture. In: Yung, M., Kurosawa, K., Safavi-Naini, R. (eds.) Digital Rights Management Workshop, pp. 69–78. ACM, New York (2006)
22. Weng, J., Chow, S.S., Yang, Y., Deng, R.H.: Efficient Unidirectional Proxy Re-Encryption. Cryptology ePrint Archive, Report 2009/189 (2009), http://eprint.iacr.org/
23. Zhang, X., Chen, M.-R., Li, X.: Comments on Shao-Cao's Unidirectional Proxy Re-Encryption Scheme from PKC 2009. Cryptology ePrint Archive, Report 2009/344 (2009), http://eprint.iacr.org

More Constructions of Lossy and Correlation-Secure Trapdoor Functions

David Mandell Freeman[1], Oded Goldreich[2], Eike Kiltz[3], Alon Rosen[4], and Gil Segev[2]

[1] Stanford University, USA
dfreeman@cs.stanford.edu
[2] Weizmann Institute of Science, Rehovot 76100, Israel
{oded.goldreich,gil.segev}@weizmann.ac.il
[3] CWI, Netherlands
kiltz@cwi.nl
[4] Efi Arazi School of Computer Science, Herzliya Interdisciplinary Center (IDC), Herzliya 46150, Israel
alon.rosen@idc.ac.il

Abstract. We propose new and improved instantiations of lossy trapdoor functions (Peikert and Waters, STOC '08), and correlation-secure trapdoor functions (Rosen and Segev, TCC '09). Our constructions widen the set of number-theoretic assumptions upon which these primitives can be based, and are summarized as follows:

– Lossy trapdoor functions based on the quadratic residuosity assumption. Our construction relies on modular squaring, and whereas previous such constructions were based on seemingly stronger assumptions, we present the first construction that is based solely on the quadratic residuosity assumption.
– Lossy trapdoor functions based on the composite residuosity assumption. Our construction guarantees essentially any required amount of lossiness, where at the same time the functions are more efficient than the matrix-based approach of Peikert and Waters.
– Lossy trapdoor functions based on the d-Linear assumption. Our construction both simplifies the DDH-based construction of Peikert and Waters, and admits a generalization to the whole family of d-Linear assumptions without any loss of efficiency.
– Correlation-secure trapdoor functions related to the hardness of syndrome decoding.

Keywords: Public-key encryption, lossy trapdoor functions, correlation-secure trapdoor functions.

1 Introduction

In this paper, we describe new constructions of lossy trapdoor functions and correlation-secure trapdoor functions. These primitives are strengthened variants of the classical notion of trapdoor functions, and were introduced with the main goal of enabling simple and black-box constructions of public-key encryption schemes that are secure against chosen-ciphertext attacks. At a high level, they are defined as follows:

Lossy trapdoor functions [25]: A collection of lossy trapdoor functions consists of two families of functions. Functions in one family are injective and can be efficiently inverted using a trapdoor. Functions in the other family are "lossy," which means that the size of their image is significantly smaller than the size of their domain. The only computational requirement is that a description of a randomly chosen function from the family of injective functions is computationally indistinguishable from a description of a randomly chosen function from the family of lossy functions.

Correlation-secure trapdoor functions [26]: The classical notion of a one-way function asks for a function that is efficiently computable but is hard to invert given the image of a uniformly chosen input. Correlation security generalizes the one-wayness requirement by considering k-wise products of functions and any specified input distribution, not necessarily the uniform distribution. Given a collection of functions $\mathcal{F}$ and a distribution $\mathcal{C}$ over k-tuples of inputs, we say that $\mathcal{F}$ is secure under $\mathcal{C}$-correlated inputs if the function $(f_1(x_1), \ldots, f_k(x_k))$ is one-way, where $f_1, \ldots, f_k$ are independently chosen from $\mathcal{F}$ and $(x_1, \ldots, x_k)$ are sampled from $\mathcal{C}$.

Lossy trapdoor functions were introduced by Peikert and Waters [25], who showed that they imply fundamental cryptographic primitives, such as trapdoor functions, collision-resistant hash functions, oblivious transfer, and CCA-secure public-key encryption. In addition, lossy trapdoor functions have already found various other applications, including deterministic public-key encryption [3], OAEP-based public-key encryption [17], "hedged" public-key encryption for protecting against bad randomness [1], security against selective opening attacks [2], and efficient non-interactive string commitments [22].

The notion of correlation security was introduced by Rosen and Segev [26], who showed that any collection of injective trapdoor functions that is one-way under a natural input distribution can be used to construct a CCA-secure public-key encryption scheme.[1] They showed that any collection of lossy trapdoor functions that are sufficiently lossy is in fact also correlation-secure. This result was recently refined by Mol and Yilek [19] who showed that even lossiness of any polynomial fraction of a single bit suffices.

These applications motivate us to investigate new constructions of lossy and correlation-secure functions. Such constructions would enable us to widen the basis upon which one can achieve the above cryptographic tasks in a simple and modular way.

[1] Any distribution where $(x_1, \ldots, x_k)$ are $(1-\epsilon)k$-wise independent, for a constant $\epsilon < 1$, can be used in their framework. In particular, this includes the case where x_1 is uniformly distributed and $x_1 = \cdots = x_k$.

1.1 Our Contributions

We propose new and improved constructions of lossy and correlation-secure trapdoor functions based on well-established number-theoretic assumptions (some of which were previously not known to imply either of the primitives). By directly applying the results of [25,26,19], we obtain new CCA-secure public-key encryption schemes based on these assumptions. Concretely, we present the following constructions:

1. *Lossy trapdoor permutations based on the quadratic residuosity assumption.* Our construction relies on Rabin's modular squaring function and is based solely on the quadratic residuosity assumption. More precisely, the function is defined as $f(x) = x^2 \cdot \delta_{r,s}(x) \bmod N$, where $N = PQ$ is an RSA modulus and $\delta_{r,s}(\cdot)$ is a function indexed by two public elements $r, s \in \mathbb{Z}_N$ serving two independent purposes. First, it extends the modular squaring function to a permutation over $\mathbb{Z}_N$. Second, $f(x)$ loses the information about the sign of x if and only if s is a quadratic residue. Therefore, under the quadratic residuosity assumption f has one bit of lossiness. We note that although a function with only one bit of lossiness (or, more generally, with only a non-negligible amount of lossiness) is not necessarily a (strong) one-way function, it nevertheless can be used as a building block for constructing even a CCA-secure public-key encryption scheme (see [19,26]).
2. *Lossy trapdoor functions based on the composite residuosity assumption.* Our construction is based on the Damgård-Jurik encryption scheme [8] with additional insights by Damgård and Nielsen [9,10]. The Damgård-Jurik scheme is based on computations in the group $\mathbb{Z}_{N^{s+1}}$, where $N = PQ$ is an RSA modulus and $s \geq 1$ is an integer (it contains Paillier's encryption scheme [23] as a special case by setting $s = 1$). At a high level, each function is described by a pair (pk, c), where pk is a public key for the encryption scheme, and c is either an encryption of 1 (injective mode) or an encryption of 0 (lossy mode). By using the homomorphic properties of the encryption scheme, given such a ciphertext c and an element x, it is possible to compute either an encryption of x in the injective mode, or an encryption of 0 in the lossy mode. We note that this construction was concurrently and independently proposed by Boldyreva et al. [3]. We also give an "all-but-one" version of the construction.
3. *Lossy trapdoor functions based on the d-Linear assumption.* Our construction both simplifies and generalizes the DDH-based construction of Peikert and Waters [25, Section 5]. (Recall that DDH is the 1-Linear assumption.) Let $\mathbb{G}$ be a finite group of order p and choose an $n \times n$ matrix M over $\mathbb{F}_p$ that has either rank d (lossy mode) or rank n (injective mode). We "encrypt" $M = (a_{ij})$ as the matrix $g^M = (g^{a_{ij}}) \in \mathbb{G}^{n \times n}$, where g is a generator of $\mathbb{G}$. If $\vec{x}$ is a binary vector of length n, then given g^M we can efficiently evaluate the function $f_M(\vec{x}) = g^{M\vec{x}} \in \mathbb{G}^n$. If M has rank n, then given M we can efficiently invert f_M on the image of $\{0,1\}^n$. On the other hand, if M has rank d and $p < 2^{n/d}$, then f is lossy. The d-Linear assumption implies that the lossy and injective modes cannot be efficiently distinguished. We also give an "all-but-one" version of the function f_M based on the DDH assumption.

4. *Correlation-secure trapdoor functions based on the hardness of syndrome decoding.* Our construction is based on Niederreiter's coding-based encryption system [21] which itself is the dual of the McEliece encryption system [18]. Our trapdoor function is defined as $f(x) = Hx$, where H is a binary $(n-k) \times n$ matrix (of a certain distribution that allows for embedding a trapdoor) and x is bit string of small Hamming weight. We show that the function's correlation security is directly implied by a result of Fischer and Stern [12] about the pseudorandomness of the function f. Interestingly, the related McEliece trapdoor function (which can be viewed as the dual of the Niederreiter function) is not correlation-secure.[2] It is however possible to extend the framework of correlation security in a natural way to obtain a direct construction of a CCA-secure encryption scheme from the McEliece trapdoor function. This was recently demonstrated by Dowsley et al [11] (who proposed the first coding-based encryption scheme that is CCA-secure in the standard model) and, for the related lattice case, independently by Peikert [24] and Goldwasser and Vaikuntanathan [14]. Our contribution is to show that the Niederreiter function admits a simple construction of correlation-secure trapdoor functions based on the same security assumptions as [11].[3] The resulting CCA-secure encryption scheme is as efficient as the one from [11].

1.2 Related Work

Most of the known constructions and applications of lossy and correlation-secure trapdoor functions are already mentioned above; here we include a few more. Besides their construction based on DDH, Peikert and Waters [25] also present a construction of lossy trapdoor functions based on the worst-case hardness of lattice problems. The construction does not enjoy the same amount of lossiness as their DDH-based one, but it still suffices for their construction of a CCA-secure public-key encryption scheme. The worst-case hardness of lattice problems is also used by Peikert [24] and by Goldwasser and Vaikuntanathan [14] to construct a CCA-secure encryption scheme using a natural generalization of correlation-secure trapdoor functions.

Kiltz et al. [17] show that the RSA trapdoor permutation is lossy under the Φ-Hiding assumption of Cachin et al. [6]. (Concretely, it has $\log_2(e)$ bits of lossiness, where e is the public RSA exponent.) Furthermore, they propose multi-prime hardness assumptions under which RSA has greater lossiness.

In concurrent and independent work, Mol and Yilek [19] propose a lossy trapdoor function based on the modular squaring function. Though this construction

[2] The McEliece trapdoor function is defined as $f'_H(x,e) := Hx \oplus e$, where H is a binary $k \times n$ matrix, x is a k-bit string and e is a error vector of small Hamming weight. Given H_1, H_2 and two evaluations $y_1 = H_1 x \oplus e$ and $y_2 = H_2 x \oplus e$ one can reconstruct the unique x by solving $(H_1 \oplus H_2)x = y_1 \oplus y_2$ for x.

[3] We remark that our construction of a correlation-secure trapdoor function from coding theory does not carry over to the lattice case since the "dual" of the one-way function used in [24,14] is not injective.

is related to ours, its security is based on the stronger assumption that a random two-prime RSA modulus is indistinguishable from a random three-prime RSA modulus. In another concurrent and independent work, Hemenway and Ostrovsky [15] generalize the framework of Peikert and Waters [25] to rely on any homomorphic hash proof system, a natural generalization of hash proof systems introduced by Cramer and Shoup [7]. Hemenway and Ostrovsky then show that homomorphic hash proof systems can be constructed based on either the quadratic residuosity assumption or the composite residuosity assumption. Their approach is significantly different than ours, and the resulting constructions seem incomparable when considering the trade-off between efficiency and lossiness.

1.3 Paper Organization

The remainder of this paper is organized as follows. In Section 2 we review the definitions of lossy and correlation-secure trapdoor functions. In Sections 3, 4, and 5 we present our constructions that are based on the quadratic residuosity assumption, the d-Linear assumption, and the hardness of syndrome decoding, respectively. Due to space constraints, the construction based on the composite residuosity assumption is only given in the full version [13].

2 Preliminaries

2.1 Lossy Trapdoor Functions

A *collection of lossy trapdoor functions* consists of two families of functions. Functions in one family are injective and can be efficiently inverted using a trapdoor. Functions in the other family are "lossy," which means that the size of their image is significantly smaller than the size of their domain. The only computational requirement is that a description of a randomly chosen function from the family of injective functions is computationally indistinguishable from a description of a randomly chosen function from the family of lossy functions.

Definition 2.1 (Lossy trapdoor functions). *A collection of (n, ℓ)-lossy trapdoor functions is a 4-tuple of probabilistic polynomial-time algorithms $(\mathsf{G}_0, \mathsf{G}_1, \mathsf{F}, \mathsf{F}^{-1})$ such that:*

1. **Sampling a lossy function:** $\mathsf{G}_0(1^n)$ *outputs a function index $\sigma \in \{0,1\}^*$.*
2. **Sampling an injective function:** $\mathsf{G}_1(1^n)$ *outputs a pair $(\sigma, \tau) \in \{0,1\}^* \times \{0,1\}^*$. (Here σ is a function index and τ is a trapdoor.)*
3. **Evaluation of lossy functions:** *For every function index σ produced by G_0, the algorithm $\mathsf{F}(\sigma, \cdot)$ computes a function $f_\sigma : \{0,1\}^n \mapsto \{0,1\}^*$, whose image is of size at most $2^{n-\ell}$.*
4. **Evaluation of injective functions:** *For every pair (σ, τ) produced by G_1, the algorithm $\mathsf{F}(\sigma, \cdot)$ computes an injective function $f_\sigma : \{0,1\}^n \mapsto \{0,1\}^*$.*
5. **Inversion of injective functions:** *For every pair (σ, τ) produced by G_1 and every $x \in \{0,1\}^n$, we have $\mathsf{F}^{-1}(\tau, \mathsf{F}(\sigma, x)) = x$.*

6. **Security:** *The two ensembles* $\{\sigma : \sigma \leftarrow \mathsf{G}_0(1^n)\}_{n\in\mathbb{N}}$ *and* $\{\sigma : (\sigma, \tau) \leftarrow \mathsf{G}_1(1^n)\}_{n\in\mathbb{N}}$ *are computationally indistinguishable.*

Note that ℓ can be a function of n. Note also that we do not specify the output of F^{-1} on inputs not in the image of f_σ.

A *collection of all-but-one lossy trapdoor functions* is a more general primitive. Such a collection is associated with a set B, whose members are referred to as *branches*. (If $B = \{0, 1\}$ then we obtain the previous notion of lossy trapdoor functions.) The sampling algorithm of the collection receives an additional parameter $b^* \in B$, and outputs a description of a function $f(\cdot, \cdot)$ together with a trapdoor τ and a set of lossy branches β. The function f has the property that for any branch $b \notin \beta$ the function $f(b, \cdot)$ is injective (and can be inverted using τ), and the function $f(b^*, \cdot)$ is lossy. Moreover, the description of f hides (in a computational sense) the set of lossy branches β.

Our definition is slightly more general than that of Peikert and Waters [25, Section 3.2], which allows only one lossy branch (i.e., $\beta = \{b^*\}$). We allow possibly many lossy branches (other than b^*), and require that given a description of a function and b^* it is computationally infeasible to find another lossy branch. The proof of security of the Peikert-Waters CCA-secure public-key encryption scheme [25, Section 4.3] can easily be adapted to our more general context. (We are currently not aware of other applications of all-but-one lossy trapdoor functions).

Definition 2.2 (All-but-one lossy trapdoor functions). *A collection of* (n, ℓ)*-all-but-one lossy trapdoor functions is a* 4*-tuple of probabilistic polynomial-time algorithms* $(\mathsf{B}, \mathsf{G}, \mathsf{F}, \mathsf{F}^{-1})$ *such that:*

1. **Sampling a branch:** $\mathsf{B}(1^n)$ *outputs a value* $b \in \{0, 1\}^*$.
2. **Sampling a function:** *For every value* b *produced by* $\mathsf{B}(1^n)$*, the algorithm* $\mathsf{G}(1^n, b)$ *outputs a triple* $(\sigma, \tau, \beta) \in \{0, 1\}^* \times \{0, 1\}^* \times \{0, 1\}^*$ *consisting of a function index* σ*, a trapdoor* τ*, and a set of lossy branches* β *with* $b^* \in \beta$.
3. **Evaluation of lossy functions:** *For every value* b^* *produced by* $\mathsf{B}(1^n)$ *and for every* (σ, τ, β) *produced by* $\mathsf{G}(1^n, b^*)$*, the algorithm* $\mathsf{F}(\sigma, b^*, \cdot)$ *computes a function* $f_{\sigma,b^*} : \{0, 1\}^n \mapsto \{0, 1\}^*$*, whose image is of size at most* $2^{n-\ell}$.
4. **Evaluation of injective functions:** *For any* b^* *and* b *produced by* $\mathsf{B}(1^n)$ *and for every* (σ, τ, β) *produced by* $\mathsf{G}(1^n, b^*)$*, if* $b \notin \beta$*, then the algorithm* $\mathsf{F}(\sigma, b, \cdot)$ *computes an injective function* $f_{\sigma,b} : \{0, 1\}^n \to \{0, 1\}^*$.
5. **Inversion of injective functions:** *For any* b^* *and* b *produced by* $\mathsf{B}(1^n)$ *and for every* (σ, τ, β) *produced by* $\mathsf{G}(1^n, b^*)$*, if* $b \notin \beta$ *then we have*

$$\mathsf{F}^{-1}(\tau, b, \mathsf{F}(\sigma, b, x)) = x.$$

6. **Security:** *For any two sequences* $\{(b_n^*, b_n)\}_{n\in\mathbb{N}}$ *such that* b_n^* *and* b_n *are distinct values in the image of* $\mathsf{B}(1^n)$*, the two ensembles* $\{\sigma : (\sigma, \tau, \beta) \leftarrow \mathsf{G}(1^n, b_n^*)\}_{n\in\mathbb{N}}$ *and* $\{\sigma : (\sigma, \tau, \beta) \leftarrow \mathsf{G}(1^n, b_n)\}_{n\in\mathbb{N}}$ *are computationally indistinguishable.*

7. **Hiding lossy branches:** *Any probabilistic polynomial-time algorithm $\mathcal{A}$ that receives as input (σ, b^*), where $b^* \leftarrow \mathsf{B}(1^n)$ and $(\sigma, \tau, \beta) \leftarrow \mathsf{G}(1^n, b^*)$, has only a negligible probability of outputting an element $b \in \beta \setminus \{b^*\}$ (where the probability is taken over the randomness of* B, G, *and* $\mathcal{A}$*).*

2.2 Correlation-Secure Trapdoor Functions

A *collection of efficiently computable functions* is a pair of algorithms $\mathcal{F} = (\mathsf{G}, \mathsf{F})$, where G is a key-generation algorithm used for sampling a description of a function, and F is an evaluation algorithm used for evaluating a function on a given input. The following definition formalizes the notion of a *k-wise product*, which is a collection $\mathcal{F}_k$ consisting of all k-tuples of functions from $\mathcal{F}$.

Definition 2.3 (k-wise product). *Let $\mathcal{F} = (\mathsf{G}, \mathsf{F})$ be a collection of efficiently computable functions. For any integer k, we define the k-*wise product *$\mathcal{F}_k = (\mathsf{G}_k, \mathsf{F}_k)$ as follows:*

– *The key-generation algorithm G_k on input 1^n invokes k independent instances of $\mathsf{G}(1^n)$ and outputs $(\sigma_1, \ldots, \sigma_k)$. That is, a function is sampled from $\mathcal{F}_k$ by independently sampling k functions from $\mathcal{F}$.*
– *The evaluation algorithm F_k on input $(\sigma_1, \ldots, \sigma_k, x_1, \ldots, x_k)$ invokes F to evaluate each function σ_i on x_i. I.e., $\mathsf{F}_k(\sigma_1, \ldots, \sigma_k, x_1, \ldots, x_k) = (\mathsf{F}(\sigma_1, x_1), \ldots, \mathsf{F}(\sigma_k, x_k))$.*

A one-way function is a function that is efficiently computable but is hard to invert given the image of a uniformly chosen input. This notion extends naturally to one-wayness under any specified input distribution, not necessarily the uniform distribution. Specifically, we say that a function is one-way with respect to an input distribution $\mathcal{I}$ if it is efficiently computable but hard to invert given the image of a random input sampled according to $\mathcal{I}$.

In the context of k-wise products, a straightforward argument shows that for any collection $\mathcal{F}$ which is one-way with respect to some input distribution $\mathcal{I}$, the k-wise product $\mathcal{F}_k$ is one-way with respect to the input distribution that samples k independent inputs from $\mathcal{I}$. The following definition formalizes the notion of one-wayness under correlated inputs, where the inputs for $\mathcal{F}_k$ may be correlated.

Definition 2.4 (One-wayness under correlated inputs). *Let $\mathcal{F} = (\mathsf{G}, \mathsf{F})$ be a collection of efficiently computable functions with domain $\{D_n\}_{n \in \mathbb{N}}$, and let $\mathcal{C}$ be a distribution where $\mathcal{C}(1^n)$ is distributed over $D_n^k = D_n \times \cdots \times D_n$ for some integer $k = k(n)$. We say that $\mathcal{F}$ is* one-way under $\mathcal{C}$-correlated inputs *if $\mathcal{F}_k$ is one-way with respect to the input distribution $\mathcal{C}$.*

For the special case that distribution $\mathcal{C}$ is the uniform k-repetition distribution (i.e., $\mathcal{C}$ samples a uniformly random input $x \in D_n$ and outputs k copies of x), we say that $\mathcal{F}$ is *one-way under k-correlated inputs*. Rosen and Segev [26, Theorem 3.3] show that a collection of (n, ℓ)-lossy trapdoor functions can be used to construct a collection $\mathcal{F}$ that is one-way under k-correlated inputs for any $k < \frac{n - \omega(\log n)}{n - \ell}$.

3 A Construction Based on the Quadratic Residuosity Assumption

Our construction is based on the modular squaring function $x \mapsto x^2 \bmod N$, where $N = PQ$ for prime numbers $P \equiv Q \equiv 3 \bmod 4$ (i.e., Blum integers). This is a 4-to-1 mapping on $\mathbb{Z}_N^*$, and the construction is obtained by embedding additional information in the output that reduces the number of preimages to either 2 (these are the lossy functions) or 1 (these are the injective functions) in a computationally indistinguishable manner. Although this results in one bit of lossiness when the functions are defined over $\mathbb{Z}_N^*$, all lossy trapdoor functions in a collection are required to share the same domain (i.e., the domain should depend only on the security parameter). We overcome this difficulty with a simple domain extension, which results in lossiness of $\log_2(4/3)$ bits.

For any odd positive integer N, we denote by $\mathsf{JS}_N : \mathbb{Z} \to \{-1, 0, 1\}$ the Jacobi symbol mod N. We define functions $h, j : \mathbb{Z} \to \{0, 1\}$ as follows:

$$h(x) = \begin{cases} 1, & \text{if } x > N/2 \\ 0, & \text{if } x \leq N/2 \end{cases}$$

$$j(x) = \begin{cases} 1, & \text{if } \mathsf{JS}_N(x) = -1 \\ 0, & \text{if } \mathsf{JS}_N(x) = 0 \text{ or } 1 \end{cases}$$

We define h and j on $\mathbb{Z}_N$ by representing elements of $\mathbb{Z}_N$ as integers between 0 and $N - 1$.

Fact 3.1 *Let $N = PQ$ where $P \equiv Q \equiv 3 \bmod 4$, and let $y \in \mathbb{Z}_N^*$ be a quadratic residue. Denote by $\{\pm x_0, \pm x_1\}$ the distinct solutions of the equation $x^2 = y \bmod N$. Then, $\mathsf{JS}_P(-1) = \mathsf{JS}_Q(-1) = -1$ and therefore*

1. $\mathsf{JS}_N(x_0) = \mathsf{JS}_N(-x_0)$ *and* $\mathsf{JS}_N(x_1) = \mathsf{JS}_N(-x_1)$.
2. $\mathsf{JS}_N(x_0) = -\mathsf{JS}_N(x_1)$.

In particular, the four square roots of y take all four values of $(h(x), j(x))$.

The construction. We define a 4-tuple $\mathcal{F} = (\mathsf{G}_0, \mathsf{G}_1, \mathsf{F}, \mathsf{F}^{-1})$ (recall Definition 2.1) as follows:

1. **Sampling a lossy function:** On input 1^n the algorithm G_0 chooses an n-bit modulus $N = PQ$, where $P \equiv Q \equiv 3 \bmod 4$ are $n/2$-bit prime numbers. Then it chooses random $r \in \mathbb{Z}_N^*$ such that $\mathsf{JS}_N(r) = -1$, and a random $s \in \mathbb{Z}_N^*$ such that $\mathsf{JS}_N(s) = 1$ and s is a *quadratic residue*. The function index is $\sigma = (N, r, s)$.
2. **Sampling an injective function:** On input 1^n the algorithm G_1 chooses an n-bit modulus $N = PQ$, where $P \equiv Q \equiv 3 \bmod 4$ are $n/2$-bit prime numbers. Then it chooses random $r \in \mathbb{Z}_N^*$ such that $\mathsf{JS}_N(r) = -1$, and a random $s \in \mathbb{Z}_N^*$ such that $\mathsf{JS}_N(s) = 1$ and s is a *quadratic non-residue*. The function index is $\sigma = (N, r, s)$, and the trapdoor is $\tau = (P, Q)$.

3. **Evaluation:** Given a function index $\sigma = (N, r, s)$ and $x \in \{0,1\}^n$, the algorithm F interprets x as an integer in the set $\{1, \ldots, 2^n\}$ and outputs
$$f_{N,r,s}(x) = \begin{cases} x^2 \cdot r^{j(x)} \cdot s^{h(x)} \bmod N, & \text{if } 1 \le x < N \\ x, & \text{if } N \le x \le 2^n \end{cases}$$
4. **Inversion:** Given a description of an injective function $\sigma = (N, r, s)$ together with its trapdoor $\tau = (P, Q)$ and $y = f_{N,r,s}(x)$, the algorithm F^{-1} retrieves x as follows. If $N \le y \le 2^n$, then the algorithm outputs y. Otherwise,
   (a) Find $j(x)$ by computing $\mathsf{JS}_N(f_{N,r,s}(x))$ (note that $\mathsf{JS}_N(f_{N,r,s}(x)) = \mathsf{JS}_N(x)$). Let $y' = yr^{-j(x)}$.
   (b) Find $h(x)$ by checking whether y' is a quadratic residue mod N (note that $h(x) = 1$ if and only if y' is not a quadratic residue). Let $y'' = y's^{-h(x)}$.
   (c) Find all square roots of y'' in $\mathbb{Z}_N$, and output the one that agrees with both $j(x)$ and $h(x)$. (We use Fact 3.1 if $y'' \in \mathbb{Z}_N^*$, and note that if $1 < \gcd(y'', N) < N$, then y'' has two square roots that are negatives of each other.)

We now prove that the above construction is indeed lossy based on the *quadratic residuosity assumption*. Let $\mathcal{J}_N = \{x \in \mathbb{Z}_N^* : \mathsf{JS}_N(x) = 1\}$, and let $\mathcal{Q}_N$ be the subgroup of squares in $\mathbb{Z}_N^*$. Then the quadratic residuosity assumption states that the two distributions obtained by sampling uniformly at random from $\mathcal{Q}_N$ and from $\mathcal{J}_N \setminus \mathcal{Q}_N$ are computationally indistinguishable.

Theorem 3.2. *Under the quadratic residuosity assumption, $\mathcal{F}$ is a collection of $(n, \log_2(4/3))$-lossy trapdoor functions.*

Proof. First, it follows from the correctness of the inversion algorithm that G_1 outputs permutations on the set $\{1, \ldots, 2^n\}$. Next, we claim that G_0 outputs functions that are 2-to-1 on the set $\{1, \ldots, N-1\}$. Suppose $y \in \mathcal{Q}_N$. Since s is a quadratic residue, Fact 3.1 implies that for each $(\eta, \iota) \in \{0,1\}^2$ there is an $x_{\eta,\iota}$ satisfying
$$x_{\eta,\iota}^2 = ys^{-\eta}, \quad h(x_{\eta,\iota}) = \eta, \quad j(x_{\eta,\iota}) = \iota.$$
Then for each $\eta \in \{0,1\}$ we have $f_{N,r,s}(x_{\eta,0}) = y$ and $f_{N,r,s}(x_{\eta,1}) = ry$. Thus each element in the set $\mathcal{Q}_N \cup r\mathcal{Q}_N$ has at least two preimages in $\mathbb{Z}_N^*$, and since this set has cardinality half that of $\mathbb{Z}_N^*$ we deduce that $f_{N,r,s}$ is 2-to-1 on $\mathbb{Z}_N^*$. A similar argument shows that every square in the group $X_P = \{x \in \mathbb{Z}_N : \gcd(x, N) = P\}$ has two preimages in X_P, and the same for X_Q. Since $\{1, \ldots, N-1\} = \mathbb{Z}_N^* \cup X_P \cup X_Q$, the function $f_{N,r,s}$ is 2-to-1 on this whole set.

Since N is an n-bit modulus (i.e., $2^{n-1} < N < 2^n$), the lossy functions are 2-to-1 on at least half of their domain, which implies that their image is of size at most $3/4 \cdot 2^n = 2^{n - \log_2(4/3)}$. In addition, descriptions of lossy functions and injective functions differ only in the element s, which is a random element of the subgroup of $\mathbb{Z}_N^*$ with Jacobi symbol 1 that is a quadratic residue in the lossy case and a quadratic non-residue in the injective case. Therefore, the quadratic residuosity assumption implies that lossy functions are computationally indistinguishable from injective functions. □

4 A Construction Based on the d-Linear Assumption

The *d-Linear assumption* [16,27] is a generalization of the decision Diffie-Hellman assumption that may hold even in groups with an efficiently computable d-linear map. The 1-Linear assumption is DDH, while the 2-Linear assumption is also known as the *Decision Linear* assumption [4]. The assumption is as follows:

Definition 4.1. *Let $d \geq 1$ be an integer, and let $\mathbb{G}$ be a finite cyclic group of order q. We say the d*-Linear assumption *holds in $\mathbb{G}$ if the distributions*

$$\begin{array}{ll} \{(g_1, \ldots, g_d, g_1^{r_1}, \ldots, g_d^{r_d}, h, h^{r_1+\cdots+r_d}) & : g_1, \ldots, g_d, h \xleftarrow{R} \mathbb{G},\ r_1, \ldots, r_d \xleftarrow{R} \mathbb{Z}_q\}, \\ \{(g_1, \ldots, g_d, g_1^{r_1}, \ldots, g_d^{r_d}, h, h^{s}) & : g_1, \ldots, g_d, h \xleftarrow{R} \mathbb{G},\ r_1, \ldots, r_d, s \xleftarrow{R} \mathbb{Z}_q\} \end{array}$$

are computationally indistinguishable.

For any $d \geq 1$, the d-linear assumption implies the $(d+1)$-linear assumption [16, Lemma 3].

Peikert and Waters [25, Section 5] give lossy and all-but-one lossy trapdoor functions based on the DDH assumption. In the Peikert-Waters construction, the function index is an ElGamal encryption of an $n \times n$ matrix M which is either the zero matrix (lossy mode) or the identity matrix (injective mode) using a finite cyclic group $\mathbb{G}$ of order p. The DDH assumption in $\mathbb{G}$ implies that these two encryptions cannot be distinguished. The construction can be generalized to d-linear assumptions using generalized ElGamal encryption, but such schemes are less efficient since ElGamal based on the d-Linear assumption produces $d+1$ group elements per ciphertext (see e.g. [27]).

Our construction is based on the following basic observation from linear algebra: if M is an $n \times n$ matrix over a finite field $\mathbb{F}_p$ and $\vec{x}$ is a length-n column vector, then the map $f_M : \vec{x} \mapsto M\vec{x}$ has image of size $p^{\mathrm{Rk}(M)}$. If we restrict the domain to only *binary* vectors (i.e., those with entries in $\{0, 1\}$), then the function f_M is injective when $\mathrm{Rk}(M) = n$, and its inverse can be computed by $f_M^{-1} : \vec{y} \mapsto M^{-1}\vec{y}$. If on the other hand we have $\mathrm{Rk}(M) < n/\log_2(p)$, then f_M is not injective even when the domain is restricted to binary vectors, since the image is contained in a subgroup of size less than 2^n.

By performing the above linear algebra "in the exponent" of a group of order p, we can create lossy trapdoor functions based on DDH and the related d-Linear assumptions. In particular, for any n the size of the function index is the same for all d.

We will use the following notation: we let $\mathbb{F}_p$ denote a field of p elements and $\mathrm{Rk}_d(\mathbb{F}_p^{n \times n})$ the set of $n \times n$ matrices over $\mathbb{F}_p$ of rank d. If we have a group $\mathbb{G}$ of order p, an element $g \in \mathbb{G}$, and a vector $\vec{x} = (x_1, \ldots, x_n) \in \mathbb{F}_p^n$, then we define $g^{\vec{x}}$ to be the column vector $(g^{x_1}, \ldots, g^{x_n}) \in \mathbb{G}^n$. If $M = (a_{ij})$ is an $n \times n$ matrix over $\mathbb{F}_p$, we denote by g^M the $n \times n$ matrix over $\mathbb{G}$ given by $(g^{a_{ij}})$. Given a matrix $M = (a_{ij}) \in \mathbb{F}_p^{n \times n}$ and a column vector $\mathbf{g} = (g_1, \ldots, g_n) \in \mathbb{G}^n$, we define $\mathbf{g}^M$ by

$$\mathbf{g}^M = \left(\prod_{j=1}^n g_j^{a_{1j}}, \ldots, \prod_{j=1}^n g_j^{a_{nj}} \right).$$

Similarly, given a matrix $\mathbf{S}=(g_{ij})\in\mathbb{G}^{n\times n}$ and a column vector $\vec{x}=(x_1,\ldots,x_n)\in\mathbb{F}_p^n$, we define $\mathbf{S}^{\vec{x}}$ by

$$\mathbf{S}^{\vec{x}} = \left(\prod_{j=1}^n g_{1j}^{x_j},\ldots,\prod_{j=1}^n g_{nj}^{x_j}\right).$$

With these definitions, we have $(g^M)^{\vec{x}} = (g^{\vec{x}})^M = g^{(M\vec{x})}$.

The construction. For any positive integer d and any real number $\epsilon\in(0,1)$, we define a 4-tuple $\mathcal{F}=(\mathsf{G}_0,\mathsf{G}_1,\mathsf{F},\mathsf{F}^{-1})$ (recall Definition 2.1) as follows:

1. **Sampling a lossy function:** On input 1^n, the algorithm G_0 chooses at random a $\lceil\epsilon n/d\rceil$-bit prime p, a group $\mathbb{G}$ of order p, and a generator g of $\mathbb{G}$. Then it chooses a matrix $M\overset{\mathrm{R}}{\leftarrow}\mathrm{Rk}_d(\mathbb{F}_p^{n\times n})$ and computes $\mathbf{S}=g^M\in\mathbb{G}^{n\times n}$. The function index is $\sigma=\mathbf{S}$.
2. **Sampling an injective function:** On input 1^n, the algorithm G_1 chooses at random a $\lceil\epsilon n/d\rceil$-bit prime p, a group $\mathbb{G}$ of order p, and a generator g of $\mathbb{G}$. Then it chooses a matrix $M\overset{\mathrm{R}}{\leftarrow}\mathrm{Rk}_n(\mathbb{F}_p^{n\times n})$ and computes $\mathbf{S}=g^M\in\mathbb{G}^{n\times n}$. The function index is $\sigma=\mathbf{S}$, and the trapdoor is $\tau=(g,M)$.
3. **Evaluation:** Given a function index $\mathbf{S}$ and $x\in\{0,1\}^n$, we interpret x as a binary column vector $\vec{x}=(x_1,\ldots,x_n)\in\mathbb{F}_2^n$. The algorithm F computes the function $f_{\mathbf{S}}(x)=\mathbf{S}^{\vec{x}}$.
4. **Inversion:** Given a function index $\mathbf{S}$, a trapdoor $\tau=(g,M)$, and a vector $\mathbf{g}\in\mathbb{G}^n$, we define $\mathsf{F}^{-1}(\tau,\mathbf{g})$ as follows:
   (a) Compute $\mathbf{h}=(h_1,\ldots,h_n)\leftarrow\mathbf{g}^{M^{-1}}$.
   (b) Let $x_i=\log_g(h_i)$ for $i=1,\ldots,n$.
   (c) Output $\vec{x}=(x_1,\ldots,x_n)$.

Theorem 4.1. *Suppose $\epsilon n > d$. If the d-Linear assumption holds for $\mathbb{G}$, then the above family is a collection of $(n,(1-\epsilon)n)$-lossy trapdoor functions.*

Proof. We first note that in the lossy case, when M is of rank d, the image of $f_{\mathbf{S}}$ is contained in a subgroup of $\mathbb{G}^n$ of size $p^d < 2^{\epsilon n}$. The condition $\epsilon n > d$ guarantees $p\geq 3$, so when M is of rank n the function $f_{\mathbf{S}}$ is in fact injective. It is straightforward to verify that the inversion algorithm performs correctly for injective functions. Finally, by [20, Lemma A.1], the d-Linear assumption implies that the matrix $\mathbf{S}$ when M is of rank n is computationally indistinguishable from the matrix $\mathbf{S}$ when M is of rank d. □

Note that the system's security scales with the bit size of p, i.e., as $\epsilon n/d$. In addition, note that the discrete logarithms in the inversion step can be performed efficiently when $\vec{x}$ is a binary vector. (Here we take advantage of the fact that the output of F^{-1} is unspecified on inputs not in the image of F.)

We now describe the extension of the system to all-but-one lossy trapdoor functions, in the case where the parameter d in the above construction is equal to 1. Let I_n denote the $n\times n$ identity matrix. For any real number $\epsilon\in(0,1)$, we define a 4-tuple $\mathcal{F}=(\mathsf{G}_0,\mathsf{G}_1,\mathsf{F},\mathsf{F}^{-1})$ (recall Definition 2.2) as follows:

1. **Sampling a branch:** On input 1^n, the algorithm B outputs a uniformly distributed $b \in \{1, \ldots, 2^{\lfloor \epsilon n \rfloor}\}$.
2. **Sampling a function:** On input 1^n and a lossy branch b^*, the algorithm G chooses at random a $\lceil \epsilon n \rceil$-bit prime p, a group $\mathbb{G}$ of order p, and a generator g of $\mathbb{G}$. Then it chooses a matrix $A \overset{\text{R}}{\leftarrow} \text{Rk}_1(\mathbb{F}_p^{n \times n})$ Let $M = A - b^* I_n \in \mathbb{F}_p^{n \times n}$ and $\mathbf{S} = g^M \in \mathbb{G}^{n \times n}$. The function index is $\sigma = \mathbf{S}$, the trapdoor is $\tau = (g, M)$, and the set of lossy branches is $\beta = \{b^*, b^* - \text{Tr}(A)\}$.
3. **Evaluation:** Given a function index $\mathbf{S}$, a branch b, and an input $x \in \{0,1\}^n$, we interpret x as a binary column vector $\vec{x} = (x_1, \ldots, x_n)$. The algorithm F computes the function $f_{\mathbf{S},b}(\vec{x}) = \mathbf{S}^{\vec{x}} * g^{b\vec{x}}$, where $*$ indicates the componentwise product of elements of $\mathbb{G}^n$.
4. **Inversion:** Given a function index $\mathbf{S}$, a trapdoor $\tau = (g, M)$, a branch b, and a vector $\mathbf{g} \in \mathbb{G}^n$, we define $\mathsf{F}^{-1}(\tau, b, \mathbf{g})$ as follows:
   (a) If $M + bI_n$ is not invertible, output $\perp$.
   (b) Compute $\mathbf{h} = (h_1, \ldots, h_n) \leftarrow \mathbf{g}^{(M+bI_n)^{-1}}$.
   (c) Let $x_i = \log_g(h_i)$ for $i = 1, \ldots, n$.
   (d) Output $\vec{x} = (x_1, \ldots, x_n)$.

Theorem 4.2. *Suppose $\epsilon n > 1$. If the DDH assumption holds for $\mathbb{G}$, then the above family is a collection of $(n, (1-\epsilon)n)$-all-but-one lossy trapdoor functions.*

Proof. We first observe that if A is the rank 1 matrix computed by $\mathsf{G}(1^n, b^*)$, then

$$f_{\mathbf{S},b}(\vec{x}) = g^{(A-(b^*-b)I_n)\vec{x}}. \tag{4.1}$$

We now verify each property of Definition 2.2. Properties (1) and (2) are immediate. To verify property (3), note that (4.1) implies that $f_{\mathbf{S},b^*}(\vec{x}) = g^{A\vec{x}}$. Since A has rank 1, the image of $f_{\mathbf{S},b^*}$ is contained in a subgroup of $\mathbb{G}^n$ of size $p < 2^{\epsilon n}$.

To check property (4), we observe that the condition $\epsilon n > 1$ guarantees $p \geq 3$, so when $A - (b^* - b)I_n$ is invertible the function $f_{\mathbf{S},b}$ is injective. The condition $A - (b^* - b)I_n$ being not invertible is equivalent to $(b^* - b)$ being an eigenvalue of A. Since A has rank 1, its eigenvalues are 0 and $\text{Tr}(A)$. Thus $(b^* - b)$ is an eigenvalue of A if and only if $b \in \beta$, and $f_{\mathbf{S},b}$ is injective for all $b \notin \beta$. It is straightforward to verify that the inversion algorithm performs correctly whenever $b \notin \beta$, so property (5) holds.

Properties (6) and (7) follow from the DDH assumption for $\mathbb{G}$. We show property (6) by constructing a sequence of games:

Game_0**:** This is the real security game. The adversary is given b_0, b_1, and $g^{A-b_\omega I_n}$ for $\omega \overset{\text{R}}{\leftarrow} \{0,1\}$ and $A \overset{\text{R}}{\leftarrow} \text{Rk}_1(\mathbb{F}_p^{n \times n})$, and outputs a bit ω'. The adversary wins if $\omega' = \omega$.

Game_1**:** The same as Game_0, except the challenge is $g^{A'-b_\omega I_n}$ for some full rank matrix $A' \overset{\text{R}}{\leftarrow} \text{Rk}_n(\mathbb{F}_p^{n \times n})$.

Game_2**:** The same as Game_1, except the challenge is $g^{U-b_\omega I_n}$ for some uniform matrix $U \overset{\text{R}}{\leftarrow} \mathbb{F}_p^{n \times n}$.

Game_3**:** The same as Game_2, except the challenge is g^U.

Since the Game_3 challenge is independent of ω, the advantage of any adversary playing Game_3 is zero. We now show that if the DDH assumption holds for $\mathbb{G}$, then for $i = 0, 1, 2$, no polynomial-time adversary $\mathcal{A}$ can distinguish Game_i from Game_{i+1} with non-negligible advantage.

$i = 0$**:** Any algorithm that distinguishes Game_0 from Game_1 can be used to distinguish the distributions $\{g^A : A \overset{\text{R}}{\leftarrow} \mathrm{Rk}_1(\mathbb{F}_p^{n\times n})\}$ and $\{g^{A'} : A' \overset{\text{R}}{\leftarrow} \mathrm{Rk}_n(\mathbb{F}_p^{n\times n})\}$. By [5, Lemma 1], any algorithm that distinguishes these distributions can solve the DDH problem in $\mathbb{G}$.

$i = 1$**:** Since the proportion of full-rank matrices to all matrices in $\mathbb{F}_p^{n\times n}$ is $(p-1)/p$, even an unbounded adversary can distinguish Game_1 from Game_2 with probability at most $1/p$.

$i = 2$**:** Since the matrix U is uniform in $\mathbb{F}_p^{n\times n}$, the matrix $U - b_\omega I_n$ is also uniform in $\mathbb{F}_p^{n\times n}$, so Game_2 and Game_3 are identical.

We conclude that for any b_0, b_1, no polynomial-time adversary can win Game_0 with non-negligible advantage.

Finally, to demonstrate property (7) we show that any adversary $\mathcal{A}$ that produces an element of β given $\mathbf{S}$ and b^* can be used to compute discrete logarithms in $\mathbb{G}$, contradicting the DDH assumption. Choose a matrix $A \overset{\text{R}}{\leftarrow} \mathrm{Rk}_1(\mathbb{F}_p^{n\times n})$, and let $A'(X)$ be the $n \times n$ matrix over $\mathbb{F}_p[X]$ that is the matrix A with the first row multiplied by X. For any value $X = t \neq 0$, the matrix $A'(t)$ is uniformly distributed in $\mathrm{Rk}_1(\mathbb{F}_p^{n\times n})$.

Let (g, g^t) be a discrete logarithm challenge for $\mathbb{G}$. For any b^* we compute the matrix $\mathbf{S} = g^{A'(t) - b^* I_n}$ and give $(\mathbf{S}, b^*)$ to the adversary $\mathcal{A}$. If the adversary outputs $b \in \beta$ with $b \neq b^*$, then we can compute $\mathrm{Tr}(A'(t))$ since this is the only nonzero eigenvalue of $A'(t)$. If a_{ii} is the ith diagonal entry of A, this gives us an equation

$$a_{11}t + a_{22} + \cdots + a_{nn} = \lambda. \tag{4.2}$$

Since $a_{11} = 0$ with probability $1/p$, we can solve for t with all but negligible probability. □

If we choose any integer $d \geq 1$ and repeat the above construction with p a $\lceil \epsilon n/d \rceil$-bit prime and A a rank d matrix, then we expect to obtain an all-but-one lossy trapdoor function under the d-Linear assumption. Indeed, the proofs of properties (1)–(6) carry through in a straightforward way. However, the above proof of property (7) does not seem to generalize. In particular, the generalization of (4.2) is the equation $\det(A'(t) - \lambda I_n) = 0$, which can be written as $ut + v = 0$ for some (known) $u, v \in \mathbb{F}_p$. When $d = 1$ the element $u = a_{11}$ is independent of λ, so we can conclude that it is nonzero with high probability; however when $d \geq 2$ this is not the case. We thus leave as an open problem the completion of the proof for $d \geq 2$.

5 Correlated Input Security from Syndrome Decoding

Our construction is based on Niederreiter's coding-based encryption system [21] which itself is the dual of the McEliece encryption system [18].

Let $0 < \rho = \rho(n) < 1$ and $0 < \delta = \delta(n) < 1/2$ be two functions in the security parameter n. We set the domain $D_{n,\delta}$ to be the set of all n-bit strings with Hamming weight δn. Note that D_n is efficiently samplable (see e.g. [12]). The Niederreiter trapdoor function $\mathcal{F} = (\mathsf{G}, \mathsf{F}, \mathsf{F}^{-1})$ is defined as follows.

– **Key generation:** On input 1^n the algorithm G chooses at random a non-singular binary $\rho n \times \rho n$ matrix S, a $(n, n - \rho n, \delta n)$-linear binary Goppa code capable of correcting up to δn errors (given by its $\rho n \times n$ binary parity check matrix G), and a $n \times n$ permutation matrix P. It sets $H := SGP$, which is a binary $\rho n \times n$ matrix. The description of the function is $\sigma = H$, the trapdoor is $\tau = (S, G, P)$.
– **Evaluation:** Given a description H of a function and $x \in \{0,1\}^n$ with Hamming weight δn, the algorithm F computes the function $f_H(x) = Hx \in \{0,1\}^{\rho n}$.
– **Inversion:** Given the trapdoor (S, G, P) and $y = Hx$, the algorithm F^{-1} computes $S^{-1}y = GPx$, applies a syndrome decoding algorithm for G to recover $\hat{y} = Px$, and computes $x = P^{-1}\hat{y}$.

The Niederreiter trapdoor function can be proved one-way under the indistinguishability and syndrome decoding assumptions which are indexed by the parameters $0 < \rho < 1$ and $0 < \delta < 1/2$.

Indistinguishability assumption. The binary $\rho n \times n$ matrix H output by $\mathsf{G}(1^n)$ is computationally indistinguishable from a uniform matrix of the same dimensions.

Syndrome decoding assumption. The collection of functions which is defined as $f_U(x) := Ux$ for a uniform $\rho n \times n$ binary matrix U is one-way on domain $D_{n,\delta}$.

Choosing the weight δ to be close to the Gilbert-Warshamov bound is commonly believed to give hard instances for the syndrome decoding problem. The Gilbert-Warshamov bound for a $(n, k, \delta n)$ linear code with $\delta < 1/2$ is given by the equation $k/n \leq 1 - H_2(\delta)$, where $H_2(\delta) := -\delta \log_2 \delta - (1-\delta)\log_2(1-\delta)$. It is therefore assumed that the syndrome decoding assumption holds for all $0 < \delta < 1/2$ satisfying $H_2(\delta) < \rho$ [12]. Note that one-wayness also implies that the cardinality of $D_{n,\delta}$ is super-polynomial in n.

The following theorem was proved in [12].

Theorem 5.1 ([12]). *If the syndrome decoding assumption holds for $\tilde{\rho}$ and δ then the ensembles $\{(M, Mx) : M \overset{\mathrm{R}}{\leftarrow} \{0,1\}^{\tilde{\rho} n \times n};\ x \overset{\mathrm{R}}{\leftarrow} D_{n,\delta})\}_{n \in \mathbb{N}}$ and $\{(M, y) : M \overset{\mathrm{R}}{\leftarrow} \{0,1\}^{\tilde{\rho} n \times n};\ y \overset{\mathrm{R}}{\leftarrow} \{0,1\}^{\tilde{\rho} n}\}_{n \in \mathbb{N}}$ are computationally indistinguishable.*

This theorem implies that the Niederreiter trapdoor function is one-way under k-correlated inputs.

Theorem 5.2. *Suppose ρ, δ, and k are chosen such that $\tilde{\rho} := \rho k < 1$, and the indistinguishability and the syndrome decoding assumptions hold for parameters $\tilde{\rho}$ and δ. Then the Niederreiter trapdoor function is one-way under k-correlated inputs.*

Proof. Fix a probabilistic polynomial-time adversary $\mathcal{A}$ that plays the security game for one-wayness under k-correlated inputs. Define

$$\varepsilon = \Pr[\mathcal{A}(H_1, \dots, H_k, H_1(x), \dots, H_k(x)) = x],$$

where $H_i \overset{\mathrm{R}}{\leftarrow} \mathsf{G}(1^n)$ and $x \overset{\mathrm{R}}{\leftarrow} D_{n,\delta}$. We now exchange all the matrices H_i for uniform matrices U_i of the same dimension. By the indistinguishability assumption and a hybrid argument, we have that

$$\Big| \Pr[\mathcal{A}(H_1, \dots, H_k, H_1(x), \dots, H_k(x)) = x] \\ - \Pr[\mathcal{A}(U_1, \dots, U_k, U_1(x), \dots, U_k(x)) = x] \Big| \in \mathrm{negl}(n).$$

For $\tilde{\rho} := \rho k$, define the $\tilde{\rho} n \times n$ matrix U by concatenating the columns of the matrices U_i. Then the distributions $(U_1, \dots, U_k, U_1(x), \dots, U_k(x))$ and (U, Ux) are identical. Since $H_2(\delta) \leq \rho/k = \tilde{\rho}$ we can apply Theorem 5.1 to obtain

$$|\Pr[\mathcal{A}(U, Ux) = x] - \Pr[\mathcal{A}(M, u_{\tilde{\rho}n}) = x]| \in \mathrm{negl}(n),$$

where $u_{\tilde{\rho}n}$ is a uniform bit-string in $\{0,1\}^{\tilde{\rho}n}$. Observing that $\Pr[\mathcal{A}(U, u_{\tilde{\rho}n}) = x] = 1/|D_{n,\delta}| \in \mathrm{negl}(n)$ (since the Niederreiter function is assumed to be one-way) implies that ε is negligible. □

We remark that the above proof implies that the Niederreiter trapdoor function has linearly many hard-core bits, which greatly improves efficiency of the CCA-secure encryption scheme obtained by using the construction from [26].

Acknowledgements. We thank Ivan Damgård and Chris Peikert for useful discussions. David Mandell Freeman's research was conducted at CWI and Universiteit Leiden, Netherlands, and supported by a National Science Foundation International Research Fellowship, with additional support from the Office of Multidisciplinary Activities in the NSF Directorate for Mathematical and Physical Sciences. Oded Goldreich is partially supported by the Israel Science Foundation (grant No. 1041/08). Eike Kiltz is supported by the research program Sentinels. Sentinels is being financed by Technology Foundation STW, the Netherlands Organization for Scientific Research (NWO), and the Dutch Ministry of Economic Affairs. Alon Rosen is partially supported by the Israel Science Foundation (grant No. 334/08). Gil Segev is supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities.

References

1. Bellare, M., Brakerski, Z., Naor, M., Ristenpart, T., Segev, G., Shacham, H., Yilek, S.: Hedged public-key encryption: How to protect against bad randomness. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 232–249. Springer, Heidelberg (2009)

2. 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)
3. Boldyreva, A., Fehr, S., O'Neill, A.: On notions of security for deterministic encryption, and efficient constructions without random oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 335–359. Springer, Heidelberg (2008)
4. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
5. Boneh, D., Halevi, S., Hamburg, M., Ostrovsky, R.: Circular-secure encryption from decision Diffie-Hellman. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 108–125. Springer, Heidelberg (2008)
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. 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. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001); Full version (with additional co-author Nielsen, J. B.), available at www.daimi.au.dk/~ivan/GenPaillier_finaljour.ps
9. Damgård, I., Nielsen, J.B.: Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 581–596. Springer, Heidelberg (2002)
10. Damgård, I., Nielsen, J.B.: Universally composable efficient multiparty computation from threshold homomorphic encryption. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 247–264. Springer, Heidelberg (2003)
11. Dowsley, R., Müller-Quade, J., Nascimento, A.C.A.: A CCA2 secure public key encryption scheme based on the McEliece assumptions in the standard model. In: Fischlin, M. (ed.) RSA Conference 2009. LNCS, vol. 5473, pp. 240–251. Springer, Heidelberg (2009)
12. Fischer, J.-B., Stern, J.: An efficient pseudo-random generator provably as secure as syndrome decoding. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 245–255. Springer, Heidelberg (1996)
13. Freeman, D.M., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: More constructions of lossy and correlation-secure trapdoor functions. Cryptology ePrint Archive, Report 2009/590 (2009), http://eprint.iacr.org/2009/590
14. Goldwasser, S., Vaikuntanathan, V.: New constructions of correlation-secure trapdoor functions and CCA-secure encryption schemes. Manuscript (2008)
15. Hemenway, B., Ostrovsky, R.: Lossy trapdoor functions from smooth homomorphic hash proof systems. Electronic Colloquium on Computational Complexity, Report TR09-127 (2009)
16. 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)
17. Kiltz, E., O'Neill, A., Smith, A.: Lossiness of RSA and the instantiability of OAEP. Manuscript (2009)
18. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. DSN Prog. Rep., Jet Prop. Lab., 114–116 (January 1978)

19. Mol, P., Yilek, S.: Chosen-ciphertext security from slightly lossy trapdoor functions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 296–311. Springer, Heidelberg (2010), http://eprint.iacr.org/2009/524
20. Naor, M., Segev, G.: Public-key cryptosystems resilient to key leakage. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 18–35. Springer, Heidelberg (2009), Full version http://eprint.iacr.org/2009/105.
21. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Problems of Control and Information Theory [Problemy Upravlenija i Teorii Informacii] 15, 159–166 (1986)
22. Nishimaki, R., Fujisaki, E., Tanaka, K.: Efficient non-interactive universally composable string-commitment schemes. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 3–18. Springer, Heidelberg (2009)
23. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
24. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: 41st ACM Symposium on Theory of Computing, pp. 333–342 (2009)
25. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: 40th ACM Symposium on Theory of Computing, pp. 187–196 (2008), Full version http://eprint.iacr.org/2007/279
26. Rosen, A., Segev, G.: Chosen-ciphertext security via correlated products. In: Reingold, O. (ed.) Theory of Cryptography. LNCS, vol. 5444, pp. 419–436. Springer, Heidelberg (2009)
27. 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/2007/074

Chosen-Ciphertext Security from Slightly Lossy Trapdoor Functions

Petros Mol and Scott Yilek

Department of Computer Science & Engineering
University of California, San Diego
{pmol,syilek}@cs.ucsd.edu

Abstract. Lossy Trapdoor Functions (LTDFs), introduced by Peikert and Waters (STOC 2008) have been useful for building many cryptographic primitives. In particular, by using an LTDF that loses a $(1 - 1/\omega(\log n))$ fraction of all its input bits, it is possible to achieve CCA security using the LTDF as a black-box. Unfortunately, not all candidate LTDFs achieve such a high level of lossiness. In this paper we drastically lower the lossiness required to achieve CCA security, showing that an LTDF that loses *only a noticeable fraction of a single bit* can be used in a black-box way to build CCA-secure PKE. To show our result, we build on the recent result of Rosen and Segev (TCC 2009) that showed how to achieve CCA security from functions whose products are one-way on particular types of correlated inputs. Lastly, we give an example construction of a slightly lossy TDF based on the assumption that it is hard to distinguish the product of two primes from the product of three primes.

1 Introduction

Lossy Trapdoor Functions (LTDFs), recently introduced by Peikert and Waters [15], have proven to be a useful tool both for giving new constructions of traditional cryptographic primitives and also for constructing new primitives. Specifically, Peikert and Waters used LTDFs to construct one-way injective trapdoor functions, collision-resistant hash functions, CPA and CCA-secure encryption[1], and more. More recently, LTDFs were used to construct deterministic PKE schemes secure in the standard model [3], as well as PKE schemes secure under selective-opening attack [1].

Informally, an LTDF is an injective trapdoor function with a function description g that is (computationally) indistinguishable from the description $\hat{g}$ of another function that statistically loses information about its input. In other words, the function $\hat{g}$ is non-injective, with some images having potentially many preimages. We say an LTDF g (computationally) loses ℓ bits if the effective range size of the indistinguishable function $\hat{g}$ is at most a $1/2^{\ell}$-fraction of its domain

[1] By CCA-secure we mean CCA2-secure. See [8] for a good overview of all the ways currently used to achieve CCA security.

size. LTDFs allow a useful and simple proof technique: in the honest execution of a protocol we use the injective function to get the correct functionality, while in the proof the "challenge" given to an adversary will use the lossy function. One can then do a statistical argument to complete the proof.

Using LTDFs and this proof technique, Peikert and Waters show that an LTDF f with input size a polynomial $n(\lambda)$ (where λ is the security parameter) that loses $\omega(\log \lambda)$ bits is one-way. This is easy to see since if an inverter is given $\hat{g}(x)$, where $\hat{g}$ is the indistinguishable lossy function, then there are on average $2^{\omega(\log \lambda)}$ possible preimages; thus the adversary has only a negligible probability of outputting the correct one. Applying known results, these one-way TDFs immediately give CPA secure encryption using generic hardcore predicates [7]. Additionally, Peikert and Waters go on to show that LTDFs admit simple hardcore *functions*, resulting in efficient multi-bit encryption schemes.

To achieve CCA security from LTDFs, Peikert and Waters then show that any LTDF with enough lossiness can be used to construct an all-but-one trapdoor function (ABO), which can then be used to achieve CCA security. "Enough" lossiness turns out to be almost all of the input bits, which can be difficult to achieve. Peikert and Waters get enough lossiness from a DDH-based construction, however their lattice-based construction only loses a constant fraction of the input bits which turns out to be insufficient for the general construction. Thus, to get CCA security from lattice-based assumptions, they need to give a direct construction of an ABO.

Since the original paper, more constructions of LTDFs have been proposed. Rosen and Segev [20] and Boldyreva, Fehr, and O'Neill [3] both gave a construction based on the decisional composite residuosity (DCR) assumption, while Kiltz, O'Neill, and Smith [9] show that the RSA trapdoor permutation is lossy under the phi-hiding assumption of [4]. While the DCR-based LTDF has enough lossiness to construct ABOs and achieve CCA security, RSA only loses a constant fraction (less than one-half) of the input bits and thus cannot be used to construct an ABO using the general construction.

Correlated Products. Rosen and Segev [21] recently generalized the ABO technique for achieving CCA security by giving a sufficient, strictly computational assumption on the underlying TDFs. They called their notion one-wayness under correlated products. It is well known that for a polynomially-bounded w, sampling w functions independently from a family of one-way functions and applying them to independent uniform inputs still results in a one-way function, and even amplifies the one-wayness. Rosen and Segev investigated the case when the inputs are not necessarily independent and uniform, but are instead correlated in some way. They went on to show how to get CCA security from a function family that is one-way with respect to specific distributions $\mathcal{C}_w$ of w correlated inputs. Specifically, the distributions they use have the property that given any $d < w$ of the inputs the entire input vector can be reconstructed. (We call such distributions (d, w)-subset reconstructible; see Section 3 for details.) The simplest such distribution happens when $d = 1$, which Rosen and Segev call the w-repetition distribution. In this case, independently sampled functions are each applied to the *same* input[2].

Of course, this notion is useful only if there exist TDFs that are one-way under such correlations. Rosen and Segev show that LTDFs with enough lossiness satisfy the requirements. The amount of lossiness they require turns out to be approximately the same amount needed by Peikert and Waters to go from an LTDF to an ABO. This amount, as we said, is more than any constant fraction of the input bits, ruling out numerous LTDFs.

Our Results. We extend the results of [15] and [21] and show that *only a noticeable fraction of a single bit of lossiness is sufficient* for building IND-CCA secure encryption. Our results lower the required lossiness from a $(1-1/\omega(\log \lambda))$-fraction of *all* the input bits to just a $1/\text{poly}$ fraction of *one* bit. This solves an open problem from (the most recent version [14] of) [15] and additionally further confirms the usefulness of the correlated product formalization of Rosen and Segev. Our result also immediately implies that the LTDF construction based on the RSA function from [9] as well as the lattice-based construction from [15] can now be used in a black-box way to achieve CCA security.

To achieve our result, we first prove a straightforward theorem bounding the amount of lossiness required of an LTDF in order to argue that its w-wise product is one-way with respect to a correlated input distribution $\mathcal{C}_w$ with min-entropy μ. We then show that if we instantiate the error-correcting code in the Rosen-Segev construction with Reed-Solomon codes and carefully choose the parameters, then we can use a correlated input distribution $\mathcal{C}_w$ with enough min-entropy μ that we only need an LTDF that loses about two bits. Since it is easy to amplify the *quantity* of lossiness (not the rate), we can get an LTDF that loses two bits from any LTDF that loses only a noticeable fraction of a bit.

Since we have significantly lowered the amount of lossiness needed for CCA security, we hope that it will be possible to achieve CCA security via LTDFs from a wider variety of assumptions. Towards this goal, we give an example of how to build a slightly lossy TDF using an assumption from which it is not clear how to build an LTDF with significantly more lossiness. Our LTDF is based on modular squaring and it loses a constant fraction of one bit under the assumption that it is hard to distinguish the product of two primes from the product of three primes [2]. Our results described above immediately give us CCA security from this assumption[3]. Interestingly, Freeman, Goldreich, Kiltz, Rosen, and Segev [6] independently describe an LTDF that loses one bit under the quadratic residuosity assumption. Our result allows them to achieve CCA security from this slightly lossy TDF in a black-box way.

A Closer Look. To see why slightly lossy TDFs are sufficient for building a variety of cryptographic primitives, let us first focus on building CPA-secure

[2] Rosen and Segev focused on the w-repetition case in the proceedings version of their paper [21]. See their full version [19] for details on the more general case.

[3] It should be noted that this assumption is clearly stronger than other assumptions from which we already know how to achieve CCA security (e.g., factoring [8]).

encryption. For simplicity, say that we have a family $\mathcal{F}$ of LTDFs with domain $\{0,1\}^n$ that (computationally) loses 1 bit. Now consider a new family of LTDFs which is simply the w-wise product of $\mathcal{F}$ for $w = \text{poly}(\lambda)$, where λ is the security parameter. This means that to sample a function from the product family we independently sample w functions from $\mathcal{F}$; the domain of the product family is $\{0,1\}^{nw}$. It is easy to see that such a family computationally loses $w = \text{poly}(\lambda)$ bits and, applying the results of [15], is thus one-way. Applying generic hardcore predicates, this immediately gives us a CPA-secure encryption scheme.

The Rosen-Segev encryption scheme is similar, but one important difference is the input distribution to the function chosen from the product family is no longer uniform over $\{0,1\}^{nw}$, but instead correlated (recall that it is what we call (d,w)-subset reconstructible). This helps provide the ability to answer decryption queries in the proof. Rosen and Segev focused on the case $d = 1$, which means each of the w functions that make up the product function is applied to the same input. If such functions are not *very* lossy, too much information about the input will leak. We show that by choosing an appropriate error-correcting code in the RS construction and by carefully setting the parameters, we can instead set d large relative to w and thus get enough entropy in the input distribution to argue one-wayness and achieve CCA security when using only slightly lossy TDFs in the w-wise product function.

Open Directions. An interesting open question is whether we can achieve CCA-security based on other hardness assumptions. For example, is it possible to construct slightly lossy trapdoor functions from hardness assumptions from which we don't already know how to achieve CCA security? Another interesting question is whether LTDFs with small amount of lossiness are sufficient for constructing other primitives such as collision resistant hash functions. Lastly, another challenging direction is developing techniques for amplifying the lossiness *rate*, i.e., increase the lossiness to input-size ratio.

2 Preliminaries

Notation. Throughout the paper, λ denotes a security parameter. For a random variable X, we let $x \leftarrow\!\!\$\, X$ denote choosing a value uniformly at random according to (the distribution of) X and assigning it to x. We say a function $\mu(\cdot)$ is negligible if $\mu(\lambda) \in \lambda^{-\omega(1)}$ and is noticeable if $\mu(\lambda) \in \lambda^{-O(1)}$. We let negl$(\lambda)$ denote an arbitrary negligible function, poly(λ) a polynomially bounded function and $\frac{1}{poly(\lambda)}$ denote an arbitrary noticeable function.

Probability Background. Let X, Y be two (discrete) random variables distributed over a countable set $\mathcal{V}$ according to $\mathcal{D}_X$ and $\mathcal{D}_Y$ respectively. The statistical distance between X and Y (or between $\mathcal{D}_X$ and $\mathcal{D}_Y$) is defined as

$$\Delta(X,Y) = \frac{1}{2}\sum_{v\in\mathcal{V}} |\Pr[X = v] - \Pr[Y = v]|$$

For two random variable ensembles $\mathcal{X} = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ indexed by a (security) parameter λ, we say that $\mathcal{X}$ and $\mathcal{Y}$ are statistically indistinguishable (denoted $\mathcal{X} \overset{s}{\approx} \mathcal{Y}$) if $\Delta(X_\lambda, Y_\lambda) = \text{negl}(\lambda)$. Likewise, $\mathcal{X}$ and $\mathcal{Y}$ are computationally indistinguishable (denoted $\mathcal{X} \overset{c}{\approx} \mathcal{Y}$) if

$$|\Pr[\,\mathcal{A}(X_\lambda) = 1\,] - \Pr[\,\mathcal{A}(Y_\lambda) = 1\,]| = \text{negl}(\lambda)$$

for any PPT algorithm $\mathcal{A}$ (where the probability is taken over the randomness of $\mathcal{A}$ and the random variables X_λ, Y_λ).

For a random variable X taking values in a domain $\mathcal{X}$, we define its *min-entropy* as

$$\text{H}_\infty(X) = -\log(\max_{x \in \mathcal{X}} \Pr[\,X = x\,]).$$

where $\max_{x \in \mathcal{X}} \Pr[\,X = x\,] = 2^{-\text{H}_\infty(X)}$ denotes the *predictability* of the random variable X.

Another useful notion of entropy is the *average min-entropy* (defined in [5]) of a random variable X (given Y) which is defined as follows:

$$\tilde{\text{H}}_\infty(X|Y) = -\log\left(\mathop{\mathbf{E}}_{y \leftarrow Y}\left[2^{-\text{H}_\infty(X \mid Y=y)}\right]\right)$$

The average min-entropy expresses the average maximum probability of predicting X given Y. The following lemma gives a useful bound on the remaining entropy of a random variable X conditioned on the values of side information.

Lemma 1 ([5], Lemma 2.2b). *Let X, Y, Z be random variables such that Y takes at most 2^k values. Then*

$$\tilde{\text{H}}_\infty(X \mid (Y, Z)) \geq \tilde{\text{H}}_\infty((X, Y) \mid Z) - k \geq \tilde{\text{H}}_\infty(X|Z) - k.$$

In particular, if X is independent of Z then $\tilde{\text{H}}_\infty(X \mid (Y, Z)) \geq \text{H}_\infty(X) - k$.

Trapdoor Functions. A collection of injective trapdoor functions is a tuple of PT algorithms $\mathcal{F} = (G, F, F^{-1})$ such that (probabilistic) algorithm G outputs a pair (s, t) consisting of function index s and a corresponding trapdoor t. Deterministic algorithm F, on input a function index s and $x \in \{0,1\}^n$ outputs $f_s(x)$. Algorithm F^{-1}, given the trapdoor t, computes the inverse function $f_s^{-1}(\cdot)$. Consider a collection of injective trapdoor functions $\mathcal{F}$ with domain $\{0,1\}^{n(\lambda)}$ and let $X(1^\lambda)$ be a distribution over $\{0,1\}^{n(\lambda)}$. We say $\mathcal{F}$ is *one-way with respect to* X if for all PPT adversaries A and every polynomial $p(\cdot)$ it follows that for all sufficiently large λ

$$\Pr\left[\,A(1^\lambda, s, F(s,x)) = F^{-1}(t, F(s,x))\,\right] < \frac{1}{p(\lambda)}\ ,$$

where $(s,t) \leftarrow_\$ G(1^\lambda)$ and $x \leftarrow_\$ X(1^\lambda)$.

We say that $\mathcal{F}$ is $(n(\lambda), \ell(\lambda))$-lossy if there exists a PPT algorithm $\hat{G}$ that, on input security parameter 1^λ, outputs $\hat{s}$ and $\hat{t}$ such that

– The first outputs of G and $\hat{G}$ are computationally indistinguishable.
– For any $(\hat{s}, \hat{t})$ outputted by $\hat{G}$, the map $F(\hat{s}, \cdot)$ has image size at most $2^{n-\ell}$. We call ℓ the lossiness.

We will sometimes call a TDF that is lossy a lossy trapdoor function (LTDF).

Public-Key Encryption. A public-key encryption scheme is a triple $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ of PPT algorithms. The key generation algorithm $\mathcal{K}$, on input the security parameter 1^λ, outputs a pair of keys (pk, sk). The encryption algorithm $\mathcal{E}$ gets as input the public key pk and a message $m \in \mathcal{M}$ (for some message space $\mathcal{M}$) and outputs a ciphertext c. The decryption algorithm $\mathcal{D}$ on input the secret key sk and a ciphertext c, outputs a message m or $\perp$ (failure). It is required that $\Pr[\,\mathcal{D}(sk, \mathcal{E}(pk, m)) \neq m\,] = \mathrm{negl}(\lambda)$, where the probability is taken over the randomness of $\mathcal{K}, \mathcal{E}$ and $\mathcal{D}$.

The strong notion of security for a public key cryptosystem $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ we consider in this paper is indistinguishability of ciphertexts under a chosen ciphertext attack (IND-CCA) [13,17]. We define IND-CCA security as a game between and adversary $\mathcal{A}$ and an environment as follows. The environment runs $\mathcal{K}(1^n)$ to get a keypair (pk, sk) and flips a bit b. It gives pk to $\mathcal{A}$. $\mathcal{A}$ outputs a pair of messages $m_0, m_1 \in \mathcal{M}$ with $|m_0| = |m_0|$. The environment returns the challenge ciphertext $c \leftarrow_{\$} \mathcal{E}(pk, m_b)$ to $\mathcal{A}$. Additionally, throughout the entire game the adversary also has access to a decryption oracle **Dec** that, on input c, outputs $\mathcal{D}(sk, c)$. The one restriction we place on the adversary is that it may not query the challenge ciphertext to the decryption oracle, as this would lead to a trivial win. At the end of the game the adversary $\mathcal{A}$ returns a guess bit b'. We define the IND-CCA advantage of an adversary $\mathcal{A}$ as

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{A},\mathcal{AE}}(\lambda) = 2 \cdot \Pr[\,\mathcal{A}\ \textit{wins}\,] - 1\,.$$

We say that $\mathcal{AE}$ is CCA-secure if $\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{A},\mathcal{AE}}(\lambda)$ is negligible in λ for all PPT adversaries $\mathcal{A}$.

Error Correcting Codes. We use error correcting codes for the construction of the CCA secure scheme. [4] In this section we review some basic definitions and facts from coding theory. We focus only on the material that is required for the security proof of our CCA construction. The reader is referred to [10] for a detailed treatment of the subject.

Let Σ be a set of symbols (alphabet) with $|\Sigma| = q$. For two strings $\mathbf{x}, \mathbf{y} \in \Sigma^w$, the *Hamming distance* $d_H(\mathbf{x}, \mathbf{y})$ is defined as the number of coordinates where $\mathbf{x}$ differs from $\mathbf{y}$. Consider now an encoding map $\mathsf{ECC} : \Sigma^k \to \Sigma^w$. A *code* $\mathcal{C}$ is simply the image of such a map (that is $\mathcal{C} \subseteq \Sigma^w$), with $|\mathcal{C}| = q^k$. The *minimum distance* of a code $\mathcal{C}$ is defined as

$$d(\mathcal{C}) = \min_{\substack{\mathbf{x},\mathbf{y}\in\mathcal{C} \\ \mathbf{x}\neq\mathbf{y}}} \{d_h(\mathbf{x}, \mathbf{y})\}$$

We use $[w, k, d]_q$ to denote a code $\mathcal{C}$ with *block length* w ($\mathcal{C} \subseteq \Sigma^w$), *message length* $k = \log_q |\mathcal{C}|$, minimum distance $d(\mathcal{C}) = d$ and alphabet size $|\Sigma| = q$.

For the CCA construction we need a code whose words are as "far apart" as possible. In particular, for a fixed k, we need a code which maximizes d/w under the restriction that w is polynomial in k. By the Singleton bound [23],

[4] For the purposes of the construction, we only need an appropriate encoding scheme and not a full -fledged error correcting scheme, in the sense that the ability to decode is unnecessary for the construction.

$d \leq w - k + 1$ for any code and alphabet size which immediately gives an upper bound $1 - \frac{k-1}{w}$ for d/w. Codes that meet the Singleton bound are called Maximum Distance Separable (MDS) codes.

Reed-Solomon Codes. Reed-Solomon codes (introduced in [18]) are an example of MDS codes. We describe a (simplified) construction of a family of asymptotic Reed-Solomon codes. Let $RS^q_{w,k}$ denote a Reed-Solomon code (or more precisely a family of RS codes) with message length k, block length w and alphabet size $|\Sigma| = q$ (with $q \geq w$). The construction works as follows:

- *Generation:* Pick a field $\mathbb{F}_q$ (for convenience we use $\mathbb{Z}_q$ as the underlying field where q is the smallest prime such that $q \geq w$). Pick also w *distinct* elements $\alpha_1, ..., \alpha_w \in \mathbb{Z}_q$ (evaluation points).
- *Encoding:* Let $\mathbf{m} = (m_0, ..., m_{k-1}) \in \Sigma^k$ be a message and let $m(x) = \sum_{j=0}^{k-1} m_j x^j$ be the corresponding polynomial. The encoding of the message is defined as

$$\mathsf{ECC}(\mathbf{m}) = \langle m(\alpha_1), ..., m(\alpha_w) \rangle \in \mathbb{Z}_q$$

  where the evaluation takes place over $\mathbb{Z}_q$.

Lemma 2. *The Reed-Solomon code $RS^q_{w,k}$ has minimum distance $d = w - k + 1$. Also both the code length and the time complexity of the encoding are polynomial in w.*

3 Products and Correlated Inputs

In this section we define w-wise products, prove the lossiness amplification lemma that we use throughout the paper, and finally present the types of correlated input distributions we are interested in for our CCA result.

3.1 Products and Lossiness Amplification

We first define the w-wise product of a collection of functions

Definition 1 (w-wise product, Definition 3.1 in [21]). *Let $\mathcal{F} = (G, F)$ be a collection of efficiently computable functions. For any integer w, we define the w-wise product $\mathcal{F}_w = (G_w, F_w)$ as follows:*

– *The generation algorithm G_w on input 1^λ invokes $G(1^\lambda)$ for w times independently and outputs $(s_1, \ldots, s_w)$. That is, a function is sampled from $\mathcal{F}_w$ by independently sampling w functions from $\mathcal{F}$.*

– *The evaluation algorithm F_w on input $(s_1, \ldots, s_w, x_1, \ldots, x_w)$ invokes F to evaluate each function s_i on x_i. That is, $F_w(s_1, \ldots, s_w, x_1, \ldots, x_w) = (F(s_1, x_1), \ldots, F(s_w, x_w))$.*

We will use the following lemma throughout the rest of the paper. It states that w-wise products (for $w = poly(\lambda)$) amplify the *absolute amount* of lossiness[5].

Lemma 3 (Lossiness Amplification). *Let λ be a security parameter. For any family of TDFs $\mathcal{F} = (G, F, F^{-1})$ with message space $n(\lambda)$, if $\mathcal{F}$ is $(n(\lambda), \ell(\lambda))$-lossy, then the $w(\cdot)$-wise product family $\mathcal{F}_w$ (defined above) built from $\mathcal{F}$ is $(n(\lambda)\cdot w(\lambda), \ell(\lambda)\cdot w(\lambda))$-lossy for all $w = poly(\lambda)$.*

Proof. First, if there exists an efficient lossy key generation algorithm $\hat{G}$ that outputs indistinguishable function indices from G, then by a standard hybrid argument it follows that $\hat{G}_w$, which runs $\hat{G}$ independently w times to get $(s_1, t_1), \ldots, (s_w, t_w)$ and outputs $(\mathbf{s}, \mathbf{t})$ where $\mathbf{s} = (s_1, \ldots, s_w)$ and $\mathbf{t} = (t_1, \ldots, t_w)$, outputs indistinguishable keys from G_w.

Second, since for each s_i outputted by $\hat{G}$ the map $F(s_i, \cdot)$ has range size at most $2^{n-\ell}$, it follows that for each $\mathbf{s}$ outputted by $\hat{G}_w$, map $F_w(\mathbf{s}, \cdot)$ has range size at most $(2^{n-\ell})^w = 2^{nw-\ell w}$. □

An immediate implication of Lemma 3 is that $(n, \frac{1}{poly(\lambda)})$-LTDFs imply injective trapdoor one-way functions and CPA-secure encryptions (the proofs of these statements are rather straightforward and hence omitted). We simply state this observation as a corollary for completeness.

Corollary 1. *Let $p(\cdot)$ be a polynomial. Then $(n, \frac{1}{p(\lambda)})$-LTDFs imply injective trapdoor one-way functions and CPA-secure encryption schemes.*

3.2 Subset Reconstructible Distributions

While it is well-known that if $\mathcal{F}$ is one-way with respect to the uniform distribution on $\{0,1\}^n$, then the product $\mathcal{F}_w$ is one-way with respect to the uniform distribution over $\{0,1\}^{nw}$, we will be interested in the security of products when the inputs are correlated and not necessarily uniform. We will be interested in input distributions that are what we call (d, w)-subset reconstructible.

Definition 2 ((d, w)- Subset Reconstructible Distribution (SRD)). *Let $d, w \in \mathbb{N}$ such that $d \leq w$, $\mathcal{S}$ be a domain and $\mathcal{D}$ a distribution with support $Supp(\mathcal{D}) \subseteq \mathcal{S}^w$. We say that $\mathcal{D}$ is (d, w)- Subset Reconstructible (and denote $\mathcal{SRD}_{d,w}$) if, each w-tuple $(x_1, ..., x_w) \in Supp(\mathcal{D})$ is fully and uniquely reconstructible from any subset $\{x_{i_1}, ..., x_{i_d}\}$ of d distinct elements of the tuple.*

It is easy to see that the special case where $d = 1$ and $\mathcal{S} = \{0,1\}^n$ gives the uniform w-repetition distribution used in the simplified construction of the CCA secure cryptosystems in [21]. For our CCA-construction, we need to choose a value for d smaller than w (this is necessary for almost perfect simulation of

[5] We use the term "absolute amount of lossiness" to explicitly distinguish it from "rate of lossiness" defined as $\frac{k}{n}$ for a (n, k)-LTDF. Amplifying the rate of lossiness seems to be a much harder problem than amplifying the absolute amount of lossiness.

the decryption oracle) but as close to w as possible in order to minimize the required lossiness of the TDF (the closer to 1 the value $\frac{d}{w}$ is, the less lossiness we need for the CCA construction). We note that the SRD notion is similar to other well-known notions in coding theory and cryptography; we compare and contrast in [11].

Sampling via Polynomial Interpolation. We use polynomial interpolation as a way to sample efficiently from $\mathcal{SRD}_{d,w}$ for any value of d and w. The construction is identical to the one used by Shamir [22] for a (d, w)-threshold secret sharing scheme. On input a prime Q (with $\log Q = O(\text{poly}(\lambda)))$ and integers d, w, the sampling algorithm picks independently d values $p_0, ..., p_{d-1}$ uniformly at random from $\mathbb{Z}_Q$ (these correspond to the d coefficients of a $(d-1)$-degree polynomial $p \in \mathbb{Z}_Q[x]$). The algorithm then simply outputs $(x_1, ..., x_w) = (p(1), ..., p(w))$ where evaluation takes place in $\mathbb{Z}_Q$ and x_i's are represented by binary strings of length at most $\log Q$. [6] The following lemma (proved in [11]) states that the output distribution of polynomial interpolation sampling is a (d, w)- subset reconstructible distribution with sufficient entropy.

Lemma 4. *Let* $w = \text{poly}(\lambda)$. *Then the above algorithm is a* $\text{poly}(\lambda)$*-sampling algorithm for* $\mathcal{SRD}_{d,w}$. *Also the min-entropy of the distribution* $\mathcal{SRD}_{d,w}$ *is* $d \cdot \log Q$.

4 CCA Security from Functions with Small Lossiness

In this section we prove our main result: lossy TDFs that lose a noticeable fraction of a bit imply CCA-secure encryption. We start by describing the encryption scheme of Rosen and Segev [21] that shows that CCA security is implied by the security (one-wayness) of trapdoor injective functions under certain correlated products. We then show that $(n, 2)$-lossy TDFs imply injective trapdoor functions that are secure under these correlated products. We complete the proof by observing that $(n, 2)$-lossy TDFs can be constructed in a black-box way from LTDFs that lose a $\frac{1}{poly(\lambda)}$ fraction of a single bit (this is clear by a straightforward lossiness amplification argument).

For ease of presentation, we describe a single-bit encryption scheme. Due to a recent result [12], this directly implies the existence of multi-bit CCA-secure schemes. We mention however that one can get a multi-bit encryption scheme directly by simply replacing the hardcore predicate h with a universal hash function, as in the PKE schemes of [15].

4.1 The Rosen-Segev Construction

We recall the cryptosystem from [21]. Let $\mathcal{F} = (G, F, F^{-1})$ be a collection of injective trapdoor functions, $\mathcal{C}_w$ be an input distribution such that any $\mathbf{x} = (x_1, \ldots, x_w)$ outputted by $\mathcal{C}_w(1^\lambda)$ can be reconstructed given any size $d < w$

[6] Any (fixed and public) distinct values $a_1, ..., a_w \in \mathbb{Z}_q$ instead of $1, ..., w$ would work just fine.

subset of $\mathbf{x}$. Let also $h : \{0,1\}^* \rightarrow \{0,1\}$ be a predicate, $\mathsf{ECC} : \Sigma^k \rightarrow \Sigma^w$ be the PT encoding function for an error-correcting code with distance d and $\Pi = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Ver})$ be a one-time signature scheme whose verification keys are elements from Σ^k. The RS encryption scheme works as follows:

Key Generation: On input security parameter 1^λ, for each $\sigma \in \Sigma$ and each $1 \leq i \leq w$, run $(s_i^\sigma, t_i^\sigma) \leftarrow_{\$} G(1^\lambda)$, the key generation for the injective trapdoor function family. Return the pair (pk, sk) where

$$pk = (\{s_1^\sigma\}_{\sigma\in\Sigma}, \ldots, \{s_w^\sigma\}_{\sigma\in\Sigma})$$
$$sk = (\{t_1^\sigma\}_{\sigma\in\Sigma}, \ldots, \{t_w^\sigma\}_{\sigma\in\Sigma})$$

Encryption: On input public key pk and one-bit message m, run $\mathsf{Kg}(1^\lambda)$ to generate (VK, SK) and sample $(x_1, \ldots, x_w)$ from $\mathcal{C}_w(1^\lambda)$. Apply the error correcting code to VK to get $\mathsf{ECC}(VK) = (\sigma_1, \ldots, \sigma_w)$. Then output

$$c = (VK, y_1, \ldots, y_w, c_1, c_2) ,$$

where VK is as above and

$$y_i = F(s_i^{\sigma_i}, x_i),\ 1 \leq i \leq w$$
$$c_1 = m \oplus h(s_1^{\sigma_1}, \ldots, s_w^{\sigma_w}, x_1, \ldots, x_w)$$
$$c_2 = \mathsf{Sign}(SK, (y_1, \ldots, y_w, c_1)) .$$

Decryption: On input secret key sk and ciphertext $(VK, y_1, \ldots, y_w, c_1, c_2)$ check if $\mathsf{Ver}(VK, (y_1, \ldots, y_w, c_1), c_2)$ equals 1. If not output $\perp$. Otherwise, compute $\mathsf{ECC}(VK) = (\sigma_1, \ldots, \sigma_w)$ and pick d distinct indices $i_1, ..., i_d$. Use the trapdoors $t_{i_1}^{\sigma_{i_1}}, ..., t_{i_d}^{\sigma_{i_d}}$ to compute

$$x_{i_1} = F^{-1}(t_{i_1}^{\sigma_{i_1}}, y_{i_1}), \ldots, x_{i_d} = F^{-1}(t_{i_d}^{\sigma_{i_d}}, y_{i_d}) .$$

Use these x_i's to reconstruct the entire vector $x_1, \ldots, x_w$. If $y_j = F(s_j^{\sigma_j}, x_j)$ for all $1 \leq j \leq w$ output $c_1 \oplus h(s_1^{\sigma_1}, \ldots, s_w^{\sigma_w}, x_1, \ldots, x_w)$ and otherwise output $\perp$.
Rosen and Segev proved the following theorem:

Theorem 1 (Theorem 5.1 in [19]). *If Π is a one-time strongly unforgeable signature scheme, $\mathcal{F}$ is secure under a $\mathcal{C}_w$-correlated product, and h is a hardcore predicate for $\mathcal{F}_w$ with respect to $\mathcal{C}_w$, then the above PKE scheme is IND-CCA secure.*

4.2 Our Result

In this section we establish the following result

Theorem 2 (Main Theorem). *CCA-secure schemes can be constructed in a black-box way from LTDFs that lose $\frac{1}{\mathrm{poly}(\lambda)}$ bits.*

The proof proceeds in two steps. In the first step (Lemma 5), we show that lossy TDFs give rise to families of injective trapdoor functions that are secure under correlated product distributions with sufficiently large entropy. Moreover, the more entropy the underlying distribution has, the less lossiness is needed from our LTDFs. In the second and final step (Lemma 6), we show that, by choosing an appropriate error correcting code and a correlated input distribution with high entropy in the Rosen-Segev scheme, we can achieve one-wayness under correlated products (and hence CCA-security) starting from lossy TDFs with minimal lossiness requirements. More specifically, using the uniform $\mathcal{SRD}_{d,w}$ (which has high entropy, see Lemma 4) as our underlying distribution and Reed-Solomon codes for ECC, we show that $(n, 2)$-lossy TDFs suffice for CCA-secure encryption. We then derive Theorem 2 by observing that $(n, 2)$-lossy TDFs can be constructed by $(n', \frac{1}{\text{poly}(\lambda)})$-lossy functions (where $n = \text{poly}(n')$) (see Lemma 3).

Lemma 5. *Let $\mathcal{F} = (G, F, F^{-1})$ be a collection of (n, ℓ)-lossy trapdoor functions and let $\mathcal{F}_w = (G_w, F_w)$ be its w-wise product for $w = \text{poly}(\lambda)$. Let $\mathcal{C}_w$ be an input distribution with min-entropy μ. Then $\mathcal{F}_w$ is secure under a $\mathcal{C}_w$-correlated product as long as*

$$\ell \geq n - \frac{\mu}{w} + \frac{\omega(\log \lambda)}{w}.$$

Proof. The proof is similar with a proof from [15]. Assume for the contrary that there exists an inverter $\mathcal{I}$ that succeeds at inverting $\mathcal{F}_w$ with probability $1/p(\lambda)$ for some polynomial p. We will build an adversary $\mathcal{A}$ that can distinguish between the lossy keys and real keys. Because of a standard hybrid argument, it suffices to show that there exists an adversary $\mathcal{A}$ that can distinguish with non-negligible probability the case where it is given $w = \text{poly}(\lambda)$ lossy keys (generated with $\hat{G}$) from the case where it is given $w = \text{poly}(\lambda)$ real keys (generated with G). Adversary $\mathcal{A}$, on input keys $\mathbf{s} = (s_1, \ldots, s_w)$, samples $\mathbf{x} = (x_1, \ldots, x_w)$ from $\mathcal{C}_w(1^\lambda)$ and runs the inverter $\mathcal{I}(1^\lambda, \mathbf{s}, F_w(\mathbf{s}, \mathbf{x}))$. If the $\mathbf{s}$ are real keys generated from G, then $\mathcal{I}$ will output $\mathbf{x}$ with probability $\frac{1}{p(\lambda)}$. If, however, $\mathbf{s}$ come from $\hat{G}$, then the probability of success for $\mathcal{I}$ is at most $2^{-\tilde{\mathrm{H}}_\infty(\mathbf{x} \mid (\mathbf{s}, F_w(\mathbf{s},\mathbf{x})))}$.

To bound this probability, we use Lemma 1 to see that

$$\tilde{\mathrm{H}}_\infty(\mathbf{x} \mid (\mathbf{s}, F_w(\mathbf{s}, \mathbf{x}))) \geq \mathrm{H}_\infty(\mathbf{x} \mid \mathbf{s}) - w(n - \ell) \; . \tag{1}$$

Since the choice of the functions is independent from the choices of $\mathbf{x}$, the first term on the right of the above equation is simply $\mathrm{H}_\infty(\mathbf{x})$ and thus μ. Combining with (1), we get that

$$\tilde{\mathrm{H}}_\infty(\mathbf{x} \mid (\mathbf{s}, F_w(\mathbf{s}, \mathbf{x}))) \geq \mu - w(n - \ell) \geq \omega(\log \lambda)$$

where in the last inequality we used the bound for ℓ. It follows that the probability $\mathcal{I}$ succeeds in the case when $\mathcal{A}$ is given lossy keys is upper bounded by $2^{-\omega(\log \lambda)} = \text{negl}(\lambda)$. Therefore, for that choice of ℓ the inverter has negligible success probability and thus $\mathcal{A}$ can distinguish between keys from G and keys from $\hat{G}$ which gives us our contradiction. □

Lemma 6. *CCA-secure schemes can be constructed in a black-box way from $(n, 2)$-lossy TDFs.*

Proof. Let $n = \text{poly}(\lambda)$. Let also $\mathsf{ECC} \in RS^q_{w,k}$ be a Reed-Solomon code with $k = n^\epsilon$ (for some constant ϵ with $0 < \epsilon < 1$) , $w = n^c$ for some constant $c > 1+\epsilon$, q the smallest prime such that $q \geq w$ and distance $d = w - k + 1$. Let also C_w be the distribution $\mathcal{SRD}_{d,w}$ sampled via polynomial interpolation (see Section 3.2) for some prime Q such that $n - 1 \leq \log Q \leq n$. Let finally $\mathcal{F} = (G, F, F^{-1})$ be a collection of $(n, 2)$-lossy trapdoor functions and $\mathcal{F}_w = (G_w, F_w)$ be its w-wise product. By construction (Lemma 4, Section 3.2) C_w has min-entropy $\mu = \text{H}_\infty(C_w) = d \cdot \log Q$ and can be sampled in time $\text{poly}(w) = \text{poly}(\lambda)$. In addition, by properties of the Reed-Solomon codes we have

$$\frac{d}{w} = \frac{w-k+1}{w} \geq 1 - \frac{k}{w} = 1 - \frac{1}{n^{c-\epsilon}}$$

and hence

$$\frac{\mu}{w} = \frac{d}{w} \log Q \geq (n-1) \cdot \left(1 - \frac{1}{n^{c-\epsilon}}\right) = n - 1 - \frac{1}{n^{c-\epsilon-1}} + \frac{1}{n^{c-\epsilon}}$$

Therefore, we have that

$$\begin{aligned} n - \frac{\mu}{w} + \frac{\omega(\log\lambda)}{w} &\leq n - \left(n - 1 - \frac{1}{n^{c-\epsilon-1}} + \frac{1}{n^{c-\epsilon}}\right) + \frac{\omega(\log\lambda)}{w} \\ &= 1 + \frac{1}{n^{c-\epsilon-1}} - \frac{1}{n^{c-\epsilon}} + \frac{\omega(\log\lambda)}{n^c} < 2 \end{aligned}$$

for some $\omega(\log\lambda)$- function. Applying Lemma 5, we get that $\mathcal{F}$ is secure under the aforementioned C_w-correlated product. Let h be a hardcore predicate for the w-wise product $\mathcal{F}_w$ (with respect to C_w). Applying the Rosen-Segev construction along with Theorem 1 from Section 4.1, we conclude that $(n, 2)$-lossy TDFs imply CCA-security (in a black-box sense). □

5 An Explicit Construction of a Slightly Lossy TDF

The Idea. In this section we construct an LTDF that loses 1/4 bits. At a high level, our construction works as follows: the basic component is a trapdoor function g (with trapdoor t) that statistically loses ℓ bits ($\ell \geq 0$ and $\ell = 0$ corresponds to an injective trapdoor function). Let also $\hat{g}$ be a deterministic function such that $\hat{g} \overset{c}{\approx} g$ (under some computational assumption $\mathcal{CA}$) and $\hat{g}$ loses $\hat{\ell}$ bits (that is $|Img(Dom(\hat{g}))| \leq \frac{Dom(\hat{g})}{2^{\hat{\ell}}}$) for some $\hat{\ell} > \ell$. Consider now a function h such that $\|h(x)\| = \ell$ (where $\|\cdot\|$ denotes bitsize) and $(g(x), h(x))$ uniquely determines the preimage x (which can be efficiently recovered given the trapdoor t) for all inputs x. The descriptions of the injective trapdoor function and the lossy function are $s = (g, h)$ and $\hat{s} = (\hat{g}, h)$ respectively. It is not hard to see that $\hat{s}$ corresponds to an $(\hat{\ell} - \ell)$-lossy function. Indeed $|Img(\hat{s})| \leq |Img(Dom(\hat{g}))| \cdot 2^\ell \leq \frac{Dom(\hat{g})}{2^{\hat{\ell}-\ell}}$. Finally the indistinguishability of $\hat{g}$ and g implies that $s \overset{c}{\approx} \hat{s}$.

Below we give an example on how to instantiate our technique using as a core trapdoor function the squaring over a composite modulus N. We believe that our technique might serve as a paradigm for the construction of LTDFs from other hardness assumptions in the future.

Hardness assumption. Consider the following two distributions (where $n = \text{poly}(\lambda)$).

$$\begin{aligned}
\mathit{2Primes}_n &= \{N = pq \mid \|N\| = n;\ p, q \text{ distinct primes}; p \equiv q \equiv 3 \pmod 4\}\\
\mathit{3Primes}_n &= \{N = pqr \mid \|N\| = n;\ p, q, r \text{ distinct primes}; pqr \equiv 1 \pmod 4\}
\end{aligned}$$

where $\|N\|$ denotes the bitsize of N and $\|N\| = n$ implies that the most significant bit of N is 1.

Assumption 1 **(2v3Primes).** *For any PPT algorithm $\mathcal{D}$ and any polynomial $p(\cdot)$*

$$\left|\Pr\left[\,\mathcal{D}(\mathit{2Primes}_n) = 1\,\right] - \Pr\left[\,\mathcal{D}(\mathit{3Primes}_n) = 1\,\right]\right| \leq \frac{1}{p(n)}$$

where the probability is taken over the randomness of sampling N and the internal randomness of $\mathcal{D}$.

This assumption (in a slightly different form) was introduced in [2] under the name 2OR3A.

The Construction. For our function g we use squaring modulo the product N of two large (balanced) primes p and q. This function was the basis for the Rabin cryptosystem [16]. We define a family of injective trapdoor functions $\mathcal{F} = (G, F, F^{-1})$ as follows:

$\underline{G(1^\lambda)}$: $N \leftarrow pq$, with $p \equiv q \equiv 3 \pmod 4$ and pq has bitsize $n+1$. That is, $N \leftarrow_{\$} \mathit{2Primes}_{n+1}$. Return (s, t) where $s = N$ and $t = (p, q)$.

$\underline{\hat{G}(1^\lambda)}$: $N \leftarrow pqr$ with $pqr \equiv 1 \pmod 4$[7] and pqr has bitsize $n+1$. That is, $N \leftarrow_{\$} \mathit{3Primes}_{n+1}$. Return $(s, \perp)$ where $s = N$.

$\underline{F(s, x)}$: Parse s as N. On input $x \in \{0,1\}^n$ compute $y = x^2 \bmod N$. Define $\mathcal{P}_N(x) = 1$ if $x > N/2$ and $\mathcal{P}_N(x) = 0$ otherwise and $\mathcal{Q}_N(x) = 1$ if $\mathcal{J}_N(x) = 1$ and $\mathcal{Q}_N(x) = 0$ otherwise where $\mathcal{J}_N(x)$ is the Jacobi symbol of x modulo N. Return $(y, \mathcal{P}_N(x), \mathcal{Q}_N(x))$.

$\underline{F^{-1}(t, y')}$: Parse t as (p, q) and y' as (y, b_1, b_2). Compute the square roots $x_1, ..., x_k$ of y using p and q. Compute also $\mathcal{P}_N(x_i)$ and $\mathcal{Q}_N(x_i)$ for all $i \in [k]$ and output the (unique) x_i such that $\mathcal{P}_N(x_i) = b_1$ and $\mathcal{Q}_N(x_i) = b_2$.

Note that even though the modulus N has bitsize $n+1$ (that is $N > 2^n$) the domain of the functions is $\{0,1\}^n$.

Theorem 3. *$\mathcal{F}$ as given above is a family of $(n, \frac{1}{4})$-lossy trapdoor functions under the* 2v3Primes *assumption.*

[7] The requirement $pqr \equiv 1 \pmod 4$ is essential since otherwise there exists a trivial algorithm that distinguishes between Ns sampled according to G and those sampled according to $\hat{G}$.

Proof. We prove the properties one by one:

Injectivity/Trapdoor: First, $F(s, x)$ is efficiently computable ($\mathcal{J}_N(x)$ can be efficiently computed even if the factorization of N is unknown). Let now $s = N$ (N being a Blum integer) and $y' = F(s, x) = (y, b_1, b_2)$.

If $y \in \mathbb{Z}_N^*$ then it has 4 square roots modulo N which can be recovered using the trapdoor (p, q) (by first recovering the pairs of square roots modulo p and q separately and then combining them using the Chinese Remainder Theorem). Let $\pm x, \pm z$ be the 4 square roots of y. Since $\mathcal{P}_N(x) = -\mathcal{P}_N(-x)\ \forall x$, only one of $x, -x$ and one of $z, -z$ is consistent with b_1. Assume wlog that x, z are consistent with b_1. Also since $x \neq \pm z$, $\mathcal{J}_N(z) = -\mathcal{J}_N(x)$ [8] and hence only one of x, z is consistent with b_2.

If $\gcd(y, N) > 1$ (wlog $\gcd(y, N) = p$), then y has exactly 2 square roots (preimages) x and $-x$ (which can be recovered using the CRT) out of which, only one is consistent with b_1.

This means that for all $(n + 1)$-bit Blum Integers N and all $x \in \{0, 1\}^n$ the triple $(x^2 \bmod N, \mathcal{P}_N(x), \mathcal{Q}_N(x))$ uniquely determines x, which, given (p, q), can be efficiently recovered. This concludes that $\mathcal{F}$ (defined over $\{0, 1\}^n$) is a collection of injective trapdoor functions.

Lossiness: Let $(\hat{s} = N, \perp) \leftarrow \hat{G}(1^\lambda)$. Consider the sets

$$S_1 = \left\{ x \in \{0,1\}^n \middle| \, x \in \mathbb{Z}_N^* \text{ and } x < \frac{N}{2} \right\}$$
$$S_2 = \left\{ x \in \{0,1\}^n \middle| \, \gcd(x, N) > 1 \text{ and } x < \frac{N}{2} \right\}$$
$$S_3 = \left\{ x \in \{0,1\}^n \middle| \, x \geq \frac{N}{2} \right\}$$

which form a partition of $\{0, 1\}^n$. Squaring modulo $N = pqr$ is an 8-to-1 function over $\mathbb{Z}_N^*$ which means that y takes at most $\frac{\phi(N)}{8}$ values. Also for all $x \in S_1$, $\mathcal{P}_N(x) = 0$ by definition. Hence $(x^2 \bmod N, \mathcal{P}_N(x), \mathcal{Q}_N(x))$ for $x \in S_1$ takes at most $\frac{\phi(N)}{8} \cdot 2$ values, that is

$$|Img(S_1)| \leq \frac{\phi(N)}{4} \quad (2)$$

Also, $|S_2| = \frac{N - \phi(N)}{2}$ (there are $N - \phi(N)$ elements that are not coprime with N and exactly half of them are smaller than $N/2$). Finally, $|S_3| \leq 2^n - \frac{N}{2}$. We then have that

$$|Img(S_2)| \leq |S_2| \leq \frac{N - \phi(N)}{2} \quad \text{and} \quad |Img(S_3)| \leq |S_3| \leq 2^n - \frac{N}{2}. \quad (3)$$

[8] It is easy to prove that if N is a Blum integer and $x, z \in \mathbb{Z}_N^*$ such that $x \neq \pm z$ and $x^2 \equiv z^2 \equiv y \pmod{N}$, then $\mathcal{J}_N(x) = -\mathcal{J}_N(z)$.

Combining equations (2) and (3) we get

$$|Img(\{0,1\}^n)| \leq \sum_{i=1}^{3} |Img(S_i)| \leq \frac{\phi(N)}{4} + \frac{N-\phi(N)}{2} + 2^n - \frac{N}{2}$$
$$= 2^n - \frac{\phi(N)}{4} \leq 2^n - \frac{2^n}{5} = \frac{4}{5}2^n \leq 2^n 2^{-\frac{1}{4}}$$

where in the penultimate inequality we used the fact that (for balanced primes p, q, r) $\phi(N) = N - O(N^{\frac{2}{3}})$ and hence $\frac{\phi(N)}{4} > \frac{N}{5} > \frac{2^n}{5}$. Therefore the image of $\{0,1\}^n$ when N is a product of 3 primes is at most $\frac{2^n}{2^{\frac{1}{4}}}$ which implies that in this case $F(\hat{s}, \cdot)$ loses (at least) $\frac{1}{4}$-bits.

Indistinguishability: The fact that $s \overset{c}{\approx} \hat{s}$ (where $(s, \cdot) \leftarrow \mathrm{G}(1^\lambda)$ and $(\hat{s}, \cdot) \leftarrow \hat{G}(1^\lambda)$) follows directly from the 2v3Primes assumption. □

Acknowledgements

We would like to thank Mihir Bellare, Russell Impagliazzo, Eike Kiltz, Daniele Micciancio, Chris Peikert, Gil Segev, and Brent Waters for useful discussions. Scott Yilek is supported by NSF grants CNS–0831536 and CNS–0627779. Petros Mol is supported by NSF grants CNS–0716790 and CCF–0634909.

References

1. 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)
2. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: STOC, pp. 103–112. ACM, New York (1988)
3. Boldyreva, A., Fehr, S., O'Neill, A.: On notions of security for deterministic encryption, and efficient constructions without random oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 335–359. Springer, Heidelberg (2008)
4. 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)
5. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. SIAM J. Comput. 38(1), 97–139 (2008); Cachin, C., Camenisch, J.L. (eds.): EUROCRYPT 2004. LNCS, vol. 3027, pp. 523–540. Springer, Heidelberg (2004)
6. Freeman, D., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: Number-theoretic constructions of lossy and correlation-secure trapdoor functions. In: PKC 2010. Springer, Heidelberg (to appear, 2010)
7. Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: Proceedings of the 21st Annual ACM Symposium on Theory of Computing – STOC 1989, pp. 25–32. ACM, New York (1989)

8. Hofheinz, D., Kiltz, E.: Practical Chosen Ciphertext Secure Encryption from Factoring. In: Joux, A. (ed.) EUROCRYPT 2009, vol. 5479, pp. 313–332. Springer, Heidelberg (2009)
9. Kiltz, E., O'Neill, A., Smith, A.: Lossiness of RSA and the Chosen-Ciphertext Security of OAEP without Random Oracles (2009) (manuscript)
10. Macwilliams, F., Sloane, N.: The Theory of Error-Correcting Codes. North-Holland, Amsterdam (January 1983)
11. Mol, P., Yilek, S.: Chosen-Ciphertext Security from Slightly Lossy Trapdoor Functions. Cryptology ePrint Archive, Report 2009/524 (2009), http://eprint.iacr.org/
12. Myers, S., Shelat, A.: Bit Encryption Is Complete. In: FOCS, pp. 607–616. IEEE Computer Society, Los Alamitos (2009)
13. Naor, M., Yung, M.: Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In: STOC, pp. 427–437. ACM, New York (1990)
14. Peikert, C., Waters, B.: Lossy Trapdoor Functions and Their Applications (October 5, 2009), Latest Version availbale at http://www.cc.gatech.edu/~cpeikert/
15. Peikert, C., Waters, B.: Lossy Trapdoor Functions and Their Applications. In: STOC 2008, pp. 187–196. ACM, New York (2008)
16. Rabin, M.O.: Digitalized Signatures and Public-Key Functions as Intractable as Factorization. Technical report, Massachusetts Institute of Technology (1979)
17. 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 (1991)
18. Reed, I.S., Solomon, G.: Polynomial Codes Over Certain Finite Fields. SIAM J. Comput. 8(2), 300–304 (1960)
19. Rosen, A., Segev, G.: Chosen-Ciphertext Security via Correlated Products. IACR ePrint Archive, Report 2008/116
20. Rosen, A., Segev, G.: Efficient lossy trapdoor functions based on the composite residuosity assumption. IACR ePrint Archive, Report 2008/134
21. Rosen, A., Segev, G.: Chosen-Ciphertext Security via Correlated Products. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 419–436. Springer, Heidelberg (2009)
22. Shamir, A.: How to Share a Secret. Commun. ACM 22(11), 612–613 (1979)
23. Singleton, R.C.: Maximum Distance q-nary Codes. IEEE Transactions on Information Theory 10, 116–118 (1964)

Efficient Set Operations in the Presence of Malicious Adversaries

Carmit Hazay[1,⋆] and Kobbi Nissim[2,⋆⋆]

[1] Dept. of Computer Science and Applied Mathematics,
Weizmann Institute and Interdisciplinary Center (IDC) Herzliya
carmit.hazay@weizmann.ac.il

[2] Dept. of Computer Science, Ben-Gurion University and Microsoft AI,
Israel
kobbi@cs.bgu.ac.il

Abstract. We revisit the problem of constructing efficient secure two-party protocols for set-intersection and set-union, focusing on the model of malicious parties. Our main results are constant-round protocols that exhibit linear communication and a linear number of exponentiations with simulation based security. In the heart of these constructions is a technique based on a combination of a perfectly hiding commitment and an oblivious pseudorandom function evaluation protocol. Our protocols readily transform into protocols that are UC-secure.

Keywords: Secure two-party computation, Simulation based security, Set intersection, Set union, Oblivious pseudorandom function evaluation.

1 Introduction

Secure function evaluation (SFE) allows two distrusting parties to jointly compute a function of their respective inputs as if the computation is executed in an ideal setting where the parties send inputs to a trusted party that performs the computation and returns its result. Starting with the work of [37,20,6,4], it is by now well known that (in various settings, and considering semi-honest and malicious adversaries) any polynomial-time computation can be generically compiled into a secure function evaluation protocol with polynomial complexity. However, more often than not, the resulting protocols are inefficient for practical uses and hence attention was given to constructing efficient protocols for specific functions. This approach that proved quite successful for the semi-honest setting (see, e.g., [26,13,28,1,17,25,5,30,24,27]), while the malicious setting remained, at large, elusive (a notable exception is [1]).

We focus on the secure computation of basic set operations (intersection and union) where the parties P_1, P_2, holding input sets X, Y, respectively, wish to compute $X \cap Y$ or $X \cup Y$. These problems have been widely looked at by researchers in the last few years, and our main goal is to come up with protocols

⋆ Research supported by an Eshkol scholarship.

⋆⋆ Research partly supported by the Israel Science Foundation (grant No. 860/06).

for set-intersection and set-union that are secure in the malicious setting and are of better complexity to those known.

We begin by briefly surveying the current constructions of two-party secure computation for set intersection and union that are most relevant to our work:

– Freedman, Nissim and Pinkas studied set intersection in [17]. They represent a set by a polynomial that zeros exactly on the set elements. Their construction for the semi-honest setting utilizes oblivious polynomial evaluation and a balanced allocation scheme and exhibits linear communication (counting field elements) and (almost) linear computation (counting modular exponentiations). See Section 3.

  They also present variants of the above protocol, for the cases where one of the parties is malicious and the other is semi-honest. For efficiency, generic zero-knowledge proofs of adherence to the protocol are avoided. The protocol for malicious P_1 (denoted *client* in [17]) and semi-honest P_2 (*server*) utilizes a cut-and-choose strategy and hence communication is inflated by a statistical security parameter. The protocol for malicious P_2 and semi-honest P_1 is in the random oracle model. A protocol that is secure in the fully malicious setup, that combines both techniques, is sketched in Section 3.1.
– Kissner and Song [25] used polynomials to represent multi-sets. Letting the roots of $Q_X(\cdot)$ and $Q_Y(\cdot)$ coincide with elements of the multi-sets X and Y, they observed that if $r(\cdot), s(\cdot)$ are polynomials chosen at random then the roots of $r(\cdot) \cdot Q_X(\cdot) + s(\cdot) \cdot Q_Y(\cdot)$ coincide with high probability with the multi-set $X \cap Y$. This beautiful observation yields a set-intersection protocol for the semi-honest case, where the parties use an additively homomorphic encryption scheme (the Paillier scheme is suggested in [25]) to perform the polynomial multiplication, introducing quadratic computation costs in the set sizes. For the security of the protocol, it is crucial that no party should be able to decrypt on her own. Hence, the secret key should be shared and joint decryption should be deployed. Assuming a trusted setup for the encryption scheme, the communication costs for the two-party case are as in the protocol for semi-honest parties of [17].

  For malicious parties [25] introduced generic zero-knowledge proofs for proving adherence to the prescribed protocol (e.g., zero-knowledge proofs of knowledge for the multiplication of the encrypted $Q_x(\cdot)$ with a randomly selected $r(\cdot)$). While this change seems to be of dire consequences to the protocol efficiency, the analysis in [25] ignores its effects. Furthermore, the costs of setting the shared key for the Paillier scheme are ignored in the analysis. To the best of our knowledge, there are currently no efficient techniques for generating the shared Paillier keys, which do not incorporate an external trusted dealer (the latter schemes include [14,15] referenced in [25]).

  In addition to that, Kissner and Song presented a protocol for the threshold set-union problem, where only the elements that appear in the combined inputs more than t times are learnt by the parties. Their protocol employs the same technique of polynomial multiplication and thus introduces quadratic computation costs as above.

– Hazay and Lindell [21] revisited secure set intersection, with the aim of achieving efficient protocols in presence of a more realistic adversarial behavior than in the benign semi-honest model, and under standard cryptographic assumptions. Two protocols were presented, one achieves security in the presence of malicious adversaries with one-sided simulatability, the other is secure in the presence of covert adversaries [2]. The main tool used in these protocols is a secure implementation of oblivious pseudorandom function evaluation.

  Having P_1, P_2 hold sets of sizes m_X, m_Y respectively, both protocols in [21] are constant round, and incur the communication of $O(m_X \cdot p(n)+m_Y)$ group elements and the computation of $O(m_X \cdot p(n) + m_Y)$ modular exponentiations, where set elements are taken from $\{0,1\}^{p(n)}$.

  We note that the protocols in [21] can be made secure in the malicious setup, e.g., by introducing a secure key selection step for the oblivious PRF and by adding zero-knowledge proofs of knowledge to show correctness at each step. Namely, for proving that the *same* PRF key is indeed used by party P_1 in each iteration and to enable extraction of its input (as a pseudorandom function is not necessarily invertible). While this would preserve the complexity of these protocols asymptotically (in m_X, m_Y), the introduction of such proofs would probably make them inefficient for practical use since there is no effieicnt known way to construct them.
– Recently, Jarecki, and Liu [23] presented a very efficient protocol for computing a pseudorandom function with a committed key (informally, this means that the same key is used in all invocations), and showed that it yields an efficient set-intersection protocol. The main restriction of this construction is that the input domain size of the pseudorandom function should be polynomial in the security parameter (curiously, the proof of security for the set-intersection protocol makes use of the ability to exhaustively search over the input domain, so removing the restriction on the input domain of the pseudorandom function does not immediately yield a set-intersection protocol for a super-polynomial domain).
– Finally, Dachman-Soled et al. [11] present a protocol for set-intersection in the presence of malicious adversaries *without* restricting the domain. Their construction uses polynomial evaluation but takes a different approach than ours by incorporating a secret sharing of the inputs to the polynomials. They avoid generic zero-knowledge by utilizing the fact that Shamir's secret sharing implies Reed Solomon code. Their protocol incurs communication of $O(mk^2 \log^2 n + kn)$ group elements and computation of $O(mnk \log n + mk^2 \log^2 n)$.

1.1 Our Contributions

Our main contributions are efficient set-intersection and set-union protocols that are secure in the setup of malicious parties. Our constructions are in the standard model, and are based on standard cryptographic assumptions (in particular, no random oracle or a trusted setup). We begin by briefly describing the construction of Freedman et al. [17] for semi-honest parties that serves as our starting point.

Secure Set Intersection with Semi-Honest Parties. The main tool used in the construction of [17] is oblivious polynomial evaluation. The basic protocol works as follows:

1. Party P_1 chooses encryption/decryption keys $(pk, sk) \leftarrow G(1^n)$ for a homomorphic encryption scheme (G, E, D) and sends pk to P_2.
2. P_1 computes the coefficients of a polynomial $Q(\cdot)$ of degree m_X, with roots set to the m_X elements of X, and sends the encrypted coefficients to P_2.
3. For each element $y \in Y$ (in random order), party P_2 chooses a random value r (taken from an appropriate set depending on the encryption scheme), and uses the homomorphic properties of the encryption scheme to compute an encryption of $r \cdot Q(y) + y$. P_2 sends the encrypted values to P_1.
4. Upon receiving these encrypted values, P_1 extracts $X \cap Y$ by decrypting each value and then checking if the result is in X. Note that if $y \in X \cap Y$ then by the construction of the polynomial $Q(\cdot)$ we get that $r \cdot Q(y) + y = r \cdot 0 + y = y$. Otherwise, $r \cdot Q(y) + y$ is a random value that reveals no information about y and (with high probability) is not in X.

Note that the communication complexity of this simple scheme is linear in $m_X + m_Y$, as $m_X + 1$ encrypted values are sent from P_1 to P_2 (these are the encrypted coefficients of $Q(\cdot)$), and m_Y encrypted values are sent from P_2 to P_1 (i.e., $Q(y)$ for every $y \in Y$). However, the work performed by P_2 is high, as each of the m_Y oblivious polynomial evaluations includes performing $O(m_X)$ exponentiations, totaling in $O(m_X \cdot m_Y)$ exponentiations.

To save on computational work, Freedman et al. introduced a balanced allocation scheme into the protocol. Loosely speaking, they used the balanced allocation scheme of [3] with $B = \frac{m_X}{\log \log m_X}$ bins, each of size $M = O(m_X/B + \log \log B) = O(\log \log m_X)$. Party P_1 now uses the balanced allocation scheme to hash every $x \in X$ into one of the B bins resulting (with high probability) with each bin's load being at most M. Instead of a single polynomial of degree m_X party P_1 now constructs a degree-M polynomial for each of the B bins, i.e., polynomials $Q_1(\cdot), \ldots, Q_B(\cdot)$ such that the roots of $Q_i(\cdot)$ are the elements put in bin i. As some of the bins contain less than M elements, P_1 pads each polynomial with zero coefficients up to degree M. Upon receiving the encrypted polynomials, party P_2 obliviously evaluates the encryption of $r_0 \cdot Q_{h_0(y)}(y) + y$ and $r_1 \cdot Q_{h_1(y)}(y) + y$ for each of the two bins $h_0(y), h_1(y)$ in which y can be allocated, enabling P_1 to extract $X \cap Y$ as above.

Neglecting constant factors, the communication complexity is not affected as P_1 now sends $BM = O(m_X)$ encrypted values and P_2 replies with $2m_Y$ encrypted values. There is, however, a dramatic reduction in the work performed by P_2 as each of the oblivious polynomial evaluations amounts now to performing just $O(M)$ exponentiations, and P_2 performs $O(m_Y \cdot M) = O(m_Y \cdot \log \log m_X)$ exponentiations overall.

Our main goal is to come up with protocols that exhibit low asymptotic communication and computation costs in the presence of malicious behavior. Noting that asymptotic complexity does not reveal everything about a protocol's efficiency or practicality, we avoid using generic zero-knowledge proofs of adherence to the prescribed protocols, even when they involve relatively short statements, and costly set up commutations that make the efficient only for very large inputs. Our contributions are realized as follows,

Preventing the Players from Deviating from the Protocol: We inherit the oblivious polynomial evaluation and balanced allocation techniques used in [17]. On top of these we introduce an efficient zero-knowledge proof that P_1 uses to show that her encrypted polynomials were correctly produced (unlike in [17], our proof does not use a cut-and-choose strategy), and a technique preventing player P_2 from deviating meaningfully from the protocol. This technique combines a perfectly hiding commitment scheme with an oblivious pseudorandom function evaluation protocol.

Eliminating the Random Oracle: In some sense, our construction replaces the random oracle used in [17] in the case of a malicious sender with a PRF, but this 'replacement' is only in a very weak sense: In our construction P_2 holds the key for the pseudorandom function, and hence the function does not look random to P_2, nor does P_2 does not need to invoke the oblivious pseudorandom evaluation protocol to compute it. The consequence is that, unlike with the simulator for the protocol in the random oracle model that can easily monitor all invocations of the oracle, our simulator cannot extract P_2's input to the pseudorandom function.

We note that the protocols of [21] also use an oblivious pseudorandom function evaluation primitive, where the player analogous to P_2 knows the key for the function. Their usage of this primitive is, however, very unlike in our protocols. In the protocols of [21] the pseudorandom function is evaluated on the set of elements that P_2 holds, using the *same* PRF key for all evaluations. Whereas in our protocols it is evaluated on a random payload using (possibly) different keys. A payload s_y is a random element that is chosen independently for each element $y \in Y$ with the aim to fix the randomness used by P_2. Meaning that the randomness for the oblivious polynomial evaluation of y is determined by the PRF evaluation of s_y. Furthermore, the protocols in [21] are designed for the covert adversary model and for the one-sided simulatability model, and hence a technique enabling full simulation of P_2 is not needed, whereas our constructions allow simulation of both parties.

Choosing the Underlying Encryption Scheme: Our protocols make extensive use of a homomorphic encryption scheme, and would remain secure (with only small modifications) under a variety of choices. We chose to work with the El Gamal scheme (that is *multiplicatively* homomorphic) although it may seem that the more natural choice is the Paillier scheme [32], that is additively homomorphic (indeed, our initial constructions were based on the Paillier scheme).

Using the Paillier scheme, a subtle problem emerges (this was overlooked, e.g., in [17]). Recall that for the Paillier scheme $pk = N, sk = \phi(N)$. Now, if P_1 knows sk when she constructs her polynomials, then she may construct a polynomial $Q(\cdot)$ such that $Q(y) \notin \mathbb{Z}_N^*$ for some specific 'target' value y. This would allow her to learn about P_2's input beyond the intended protocol output. A possible solution is that P_1, P_2 would first engage in a protocol to jointly generate pk and shares of sk, whereas P_1 would learn sk only *after* committing to her polynomials. This, however, introduces high key setup costs, and the result is a protocol that exhibits low *asymptotic* costs, but, because of its high setup costs, its efficiency is gained only for very large inputs.

Efficiency: Our protocols for set intersection and set union $\pi_\cap, \pi_\cup$ are constant round, work in the standard model and do not require a trusted setup. The underlying encryption scheme is El Gamal where the keys are selected by party P_1. Both protocols do not employ any generic zero-knowledge proof.

Assuming the protocols of [16,21] (that require $p(n)$ oblivious transfers for realizing the oblivious pseudorandom function evaluation), we get that for sets $X, Y \subset \{0,1\}^{p(n)}$ of m_X, m_Y elements respectively, the costs of $\pi_\cap, \pi_\cup$ are of sending $O(m_X + m_Y \cdot p(n))$ group elements, and the computation of $O(m_X + m_Y \cdot (\log\log m_X + p(n)))$ modular exponentiations. Note that this is significantly better than $O(m_X \cdot m_Y)$.

A significant improvement can be achieved by using a more efficient pseudorandom function evaluation instead of using the function of [29] which requires a single oblivious transfer for every input bit. This is due to the fact that our protocol uses oblivious pseudorandom function evaluation as a black box. Furthermore, for set intersection, another significant improvement can be achieved if the size of the intersection $m_{X\cap Y}$ is allowed to be leaked (to P_2). The resulting protocol is of sending $O(m_X + m_{X\cap Y} \cdot p(n))$ and computing $O(m_X + m_Y \cdot \log\log m_X + m_{X\cap Y} \cdot p(n))$. When $m_{X\cap Y} \ll m_Y$ we get a protocol that is more efficient than that of [21]. This type of improvement does not apply for [21] as well since the parties apply the PRF directly on the input set Y and thus cannot deduce $m_{X\cap Y}$ before that.

UC Security: Our protocols readily transform into the UC framework as all our simulators are straight-line in an hybrid model with access to some specific zero-knowledge proofs. We show how to modify our set intersection protocol to one that is secure in the UC framework (in the common reference string model).

For lack of space, we focus only on the protocol for secure set intersection in the malicious model, and omit the more standard details of the construction, and its proof of security. The missing details and proofs can be found in the full version of this paper [22].

2 Preliminaries

Throughout the paper, we denote the security parameter by n, and, although not explicitly specified, input lengths are always assumed to be bounded by some polynomial in n. A probabilistic machine is said to run in *polynomial-time* (PPT) if it runs in time that is polynomial in the security parameter n alone. A function $\mu(n)$ is *negligible* (in n) if for every polynomial $p(\cdot)$ there exists a value N such that $\mu(n) < \frac{1}{p(n)}$ for all $n > N$; i.e., $\mu(n) = n^{-\omega(1)}$.

2.1 Secure Two-Party Computation – Definitions

We use standard definitions of security for two party commputation in the malicious model, which we now briefly review. The reader is referred to [18, Chapter 7] for more details and motivating discussion.

We prove the security of our protocols in the setting of *malicious adversaries*, that may arbitrarily deviate from the specified protocol. Security is analyzed by comparing what an adversary can do in a *real* protocol execution to what it can do in an *ideal* scenario. In the ideal scenario, the computation involves an incorruptible *trusted third party* to whom the parties send their inputs. The trusted party computes the functionality on the inputs and returns to each party its respective output. Informally, the protocol is secure if any adversary interacting in the real protocol (i.e., where no trusted third party exists) can do no more harm than what it could do in the ideal scenario. We consider the *static* setting where the adversary is only able to corrupt a party at the outset of the protocol. There are technical issues that arise, such as that it may be impossible to achieve fairness or guaranteed output delivery. E.g., it is possible for the an adversarial party to prevent an honest party from receiving outputs.

2.2 The El Gamal Encryption Scheme

The El Gamal encryption scheme operates on a cyclic group $\mathbb{G}$ of prime order q. We will work in the group $\mathbb{Z}_{q'}^*$ where $q' = 2q + 1$ is prime, and set $\mathbb{G}$ to be the subgroup of $\mathbb{Z}_{q'}$ of quadratic residues modulo q' (note that membership in $\mathbb{G}$ can be easily checked). Let g denote a random generator in $\mathbb{G}$, then the public and secret keys are $\langle \mathbb{G}, q, g, h\rangle$ and $\langle \mathbb{G}, q, g, x\rangle$ where $x \leftarrow_R \mathbb{Z}_q$ and $h = g^x$. A message $m \in \mathbb{G}$ is encrypted by choosing $y \leftarrow_R \mathbb{Z}_q$ and the ciphertext is $\langle g^y, h^y \cdot m\rangle$. A ciphertext $c = \langle \alpha, \beta\rangle$ is decrypted as $m = \beta/\alpha^x$. We use the property that given $y = \log_g \alpha$ one can reconstruct $m = \beta/h^y$ and hence a party encrypting m can prove knowledge of m by proving knowledge of y.

The semantic security of the El Gamal scheme follows from the hardness of decisional Diffie-Hellman (DDH) in $\mathbb{G}$. The El Gamal scheme is homomorphic relative to multiplication. I.e., if $\langle \alpha_1, \beta_1\rangle$ encrypts m_1 and $\langle \alpha_2, \beta_2\rangle$ encrypts m_2 then $\langle \alpha_1 \cdot \alpha_2, \beta_1 \cdot \beta_2\rangle$ encrypts $m_1 m_2$. We additionally consider a modified version of El Gamal where the encryption is performed by choosing $y \leftarrow_R \mathbb{Z}_q$ and computing $\langle g^y, h^y \cdot g^m\rangle$. Decryption of a ciphertext $c = \langle a, b\rangle$ is performed by computing $g^m = b \cdot a^{-x}$. The fact that m cannot be efficiently recovered is not problematic for the way El Gamal is incorporated in our protocols. Moreover, this variant of El Gamal is additively homomorphic and can be used to perform oblivious linear computations (e.g., polynomial evaluation) in the exponent.

2.3 Perfectly Hiding Commitment

We use a perfectly-hiding commitment scheme $(\mathsf{com}, \mathsf{dec})$ with a zero-knowledge proof of knowledge π_{COM} for the relation

$$\mathcal{R}_{\text{COM}} = \Big\{ \Big(c, (r, m) \Big) \mid c = \mathsf{com}(m; r) \Big\},$$

where $\mathsf{com}(m; r)$ denotes the commitment to a message m using random coins r. We instantiate $\mathsf{com}(\cdot; \cdot)$ with Pedersen's commitment scheme [33], using the same underlying group $\mathbb{G}$ used for the El Gamal scheme. I.e., let $q' = 2q + 1$ where q', q are primes and let g, h be generators of the subgroup $\mathbb{G}$ of quadratic residues modulo q'. A commitment to m is then defined as $\mathsf{com}(m; r) = g^m h^r$ where $r \leftarrow_R \mathbb{Z}_{q-1}$. The scheme is perfectly hiding as for every m, r, m' there exists a single r' such that $g^m h^r = g^{m'} h^{r'}$. The scheme is binding assuming hardness of computing $\log_g h$. However, given $\log_g h$, it is possible to decommit any commitment c into any message $m \in \mathbb{Z}_q$. We instantiate π_{COM} with the proof of knowledge from [31] (this proof is not a zero-knowledge proof, yet can be modified using standard techniques [19]).

2.4 Zero-Knowledge Proofs

Our protocols employ zero-knowledge proofs of knowledge for the following relations (in the following, $\mathbb{G}$ is a group of prime order):

Type	Protocol	Relation/Language	Reference
ZKPK	π_{DL}	$\mathcal{R}_{\text{DL}} = \{((\mathbb{G}, g, h), x) \mid h = g^x\}$	[35]
ZKPK	π_{DDH}	$\mathcal{R}_{\text{DDH}} = \{((\mathbb{G}, g, g_1, g_2, g_3), x) \mid g_1 = g^x \wedge g_3 = g_2^x\}\}$	[7]
ZK	π_{NZ}	$\text{L}_{\text{NZ}} = \left\{ (\mathbb{G}, g, h, \langle \alpha, \beta \rangle) \mid \begin{matrix} \exists\ (m \neq 0, r) \text{ s.t.} \\ \alpha = g^r \wedge \beta = h^r g^m \end{matrix} \right\}$	Sec. 2.4

Zero-Knowledge Proof for $\mathbf{L_{NZ}}$. We use standard techniques for constructing a zero-knowledge proof for the language of encryptions $\langle \alpha, \beta \rangle$ of non-zero exponents of g:

$$\text{L}_{\text{NZ}} = \{(\mathbb{G}, g, h, \langle \alpha, \beta \rangle) \mid \exists\ (m \neq 0, r) \text{ s.t. } \alpha = g^r \wedge\ \beta = h^r g^m\}.$$

The construction is based on a zero-knowledge protocol π_{MULT} for the language

$$\text{L}_{\text{MULT}} = \left\{ (\mathbb{G}, g, h, c_1, c_2, c_3) \mid \begin{matrix} \exists\ m, m' \in \mathbb{Z}_q \text{ s.t. } c_1, c_2, c_3 \text{ are} \\ \text{encryptions of } g^m, g^{m'}, g^{mm'} \text{ resp.} \end{matrix} \right\}.$$

π_{MULT} is a modification of a protocol by Damgård and M. Jurik [9] designed for the Paillier encryption scheme.

2.5 Balanced Allocation

We employ a scheme for randomly mapping elements into bins, as suggested in [17]. We use the balanced allocation scheme of [3] where elements are inserted into B bins as follows. Let $h_0, h_1 : \{0,1\}^{p(n)} \to [B]$ be two randomly chosen hash functions mapping elements from $\{0,1\}^{p(n)}$ into bins $1, \ldots, B$. An element $x \in \{0,1\}^{p(n)}$ is inserted into the less occupied bin from $\{h_0(x), h_1(x)\}$, where ties are broken arbitrarily. If m elements are inserted, then except with negligible probability over the choice of the hash functions h_0, h_1, the maximum number of elements allocated to any single bin is at most $M = O(m/B + \log\log B)$. Setting $B = \frac{m}{\log\log m}$ we get that $M = O(\log\log m)$.[1] In the protocol we deviate insignificantly from the description above, and let P_1 choose seeds for two pseudorandom functions, that are used as the hash functions h_0, h_1.

2.6 Oblivious PRF Evaluation

We use a protocol π_{PRF} that obliviously evaluates a pseudorandom function in the presence of a malicious adversary. Let I_{PRF} be the indexing algorithm for a pseudorandom function ensemble, and let $k \leftarrow_R I_{\text{PRF}}(1^n)$ be a sampled key. The functionality F_{PRF} is defined as

$$(k, x) \mapsto (\lambda, F_{\text{PRF}}(k, x)). \tag{1}$$

The PRF may be instantiated with the Naor-Reingold pseudorandom function [29] with the protocol presented in [16] (and proven in [21]). The function is defined as

$$F_{\text{PRF}}((a_0, \ldots, a_n), x) = g^{a_0 \prod_{i=1}^{n} a_i^{x[i]}},$$

where $\mathbb{G}$ is a group of prime order q, g is a generator of $\mathbb{G}$, $a_i \in \mathbb{Z}_q$ and $x = (x[1], \ldots, x[n]) \in \{0,1\}^n$. The protocol involves executing an oblivious transfer for every bit of the input x. Combining this with the fact that n oblivious transfers runs require $11n + 29$ exponentiations using the protocol in [34] (the analysis in [34] includes the cost for generating a common reference string), one gets a constant-round protocol that securely computes $\mathcal{F}_{\text{PRF}}$ in the presence of malicious players using a constant number of exponentiations for every bit of the input x.

3 Secure Set Intersection

We now consider the functionality of *set intersection*, where each party's input consists of a *set*, and the size of the other party's input set. If the set sizes match, then the functionality outputs the intersection of these input sets to P_1. Otherwise P_1 is given $\perp$. More formally:

Definition 1. *Let X and Y be subsets of a predetermined domain (w.l.o.g., we assume $X, Y \subset \{0,1\}^{p(n)}$ for some polynomial $p()$ such that $2^{p(n)}$ is*

[1] A constant factor improvement is achieved using the *Always Go Left* scheme in [36] where $h_0 : \{0,1\}^{p(n)} \to [1, \ldots, \frac{B}{2}], h_1 : \{0,1\}^{p(n)} \to [\frac{b}{2} + 1, \ldots, B]$. An element x is inserted into the less occupied bin from $\{h_0(x), h_1(x)\}$; in case of a tie x is inserted into $h_0(x)$.

super-polynomial in n, and that the set elements can be represented as elements of some finite group), the functionality $\mathcal{F}_\cap$ is:

$$((X, m_Y), (Y, m_X)) \mapsto \begin{cases} (X \cap Y, \lambda) \text{ if } |X| = m_X,\ |Y| = m_Y \text{ and } X, Y \subseteq \{0,1\}^{p(n)} \\ (\perp, \lambda) \qquad \text{otherwise} \end{cases}$$

In the rest of this section we present in detail our construction for a protocol realizing $\mathcal{F}_\cap$ in the presence of malicious adversaries. For completeness we include a description of the protocol by Freedman et al. for semi-honest parties:

Protocol 1. (set-intersection protocol secure in the presence of semi-honest parties):

- **Inputs:** *The input of P_1 is m_Y and a set $X \subseteq \{0,1\}^{p(n)}$ containing m_X items; the input of P_2 is m_X and a set $Y \subseteq \{0,1\}^{p(n)}$ containing m_Y items.*
- **Auxiliary inputs:** *A security parameter 1^n.*
- **The protocol:**
  1. **Key setup:** *P_1 chooses the secret and public keys (sk, pk) for the underlying homomorphic encryption scheme (e.g., Paillier or El Gamal). She sends pk to P_2.*
  2. **Setting the balanced allocation scheme:** *P_1 computes the parameters B, M for the scheme and chooses the seeds for two (pseudo-)random hash functions $h_0, h_1 : \{0,1\}^{p(n)} \to [B]$. She sends B, M, h_0, h_1 to P_2.*
  3. **Creating polynomials for the set X:** *For every $x \in X$, P_1 maps x into the less occupied bin from $\{h_0(x), h_1(x)\}$ (ties broken arbitrarily). Let $\mathcal{B}_i$ denote the set of elements mapped into bin i and let $Q_i(x) \stackrel{\text{def}}{=} \sum_{j=0}^{M} Q_{i,j} \cdot x^j$ denote a polynomial with the set of roots $\mathcal{B}_i$. P_1 encrypts the coefficients of the polynomials and sends the encrypted coefficients to P_2.*
  4. **Substituting in the polynomials:** *Let $y_1, \ldots, y_{m_Y}$ be a random ordering of the elements of set Y. P_2 does the following for all $\alpha \in \{1, \ldots, m_Y\}$:*
     (a) *He sets $\hat{h}_0 = h_0(y_\alpha)$, $\hat{h}_1 = h_1(y_\alpha)$.*
     (b) *He chooses two random elements in the underlying group of the homomorphic encryption scheme r_0, r_1. He then uses the homomorphic properties of the encryption scheme to compute an encryption of $r_0 \cdot Q_{\hat{h}_0}(y_\alpha) + y_\alpha$ and $r_1 \cdot Q_{\hat{h}_1}(y_\alpha) + y_\alpha$. Both encrypted values are sent to P_1.*
  5. **Computing the intersection:** *P_1 decrypts each received value. If the decrypted value is in X then P_1 records as part of her local output.*

Note that, since the parties are semi-honest, P_1 outputs $X \cap Y$ with probability negligibly close to 1: (i) For elements $y_\alpha \in X \cap Y$ we get that $Q_{h_0(y_\alpha)}(y_\alpha) = 0$ or $Q_{h_1(y_\alpha)}(y_\alpha) = 0$, hence one of the corresponding encrypted values is y_α itself, and P_1 would record it in its local output. (ii) For $y_\alpha \notin X \cap Y$ we get that $Q_{h_0(y_\alpha)}(y_\alpha) \neq 0$ and $Q_{h_1(y_\alpha)}(y_\alpha) \neq 0$ and hence corresponding encrypted values are two random values $r_0 + y$ and $r_1 + y$ that fall within X with only a negligible probability.

Efficiency. The protocol runs in a constant number of rounds. The communication costs are of sending the encrypted polynomials (BM values) and the encrypted $r_0 \cdot Q_{\hat{h}_0}(y_\alpha) + y_\alpha$ and $r_1 \cdot Q_{\hat{h}_1}(y_\alpha) + y_\alpha$ ($2m_Y$ values). Using the El Gamal or Paillier encryption schemes, the computation costs are of encrypting the polynomials ($O(BM)$ exponentiations) and of obliviously computing the encryptions of $r_0 \cdot Q_{\hat{h}_0}(y_\alpha) + y_\alpha$ and $r_1 \cdot Q_{\hat{h}_1}(y_\alpha) + y_\alpha$ ($O(Mm_Y)$ exponentiations). Overall, we get that the overall communication costs are of sending $O(m_X + m_Y)$ encryptions, and the computation costs are of performing $O(m_X + m_Y \log\log n)$ modular exponentiations.

3.1 Constructing a Protocol for Malicious Parties

We note a couple of issues that need to be addressed in transforming the above protocol for semi-honest parties to a protocol for malicious parties:

1. It is easy for P_1 to construct the B polynomials such that it would learn about elements that are not in the intersection $X \cap Y$. For instance, if $Q_i(\cdot)$ is identically zero then P_1 learns all elements $\{y \in Y : h_0(y) = i \text{ or } h_1(y) = i\}$. Similarly, if the sum of degrees of $Q_1, \ldots, Q_B$ exceeds m_X then P_1 may learn about more than m_X elements in P_2's input.

   To resolve these problems we introduce a zero-knowledge protocol for verifying that $Q_i \not\equiv 0$ for all $i \in \{1, \ldots, B\}$, and $\sum_{i \in \{1,\ldots,B\}} \mathsf{deg}(Q_i) = m_X$.
2. While party P_2 is supposed to send m_Y pairs of encryptions resulting from substituting a value y (known to P_2) in the (encrypted) polynomials $Q_{h_0(y)}$ and $Q_{h_1(y)}$ it may deviate from his prescribed computation. Thus, P_2's input to the protocol may be ill defined. A solution suggested in [17] solves this problem partially, as it deals with the case where each element y is substituted in a single polynomial. This solution avoids the standard usage of zero-knowledge proofs by P_2 that it indeed followed the protocol. Instead, it enables party P_1 to redo the entire computation supposedly carried out by P_2 on y and verify that its outcome is consistent with the messages received from P_2 (this is where the construction uses a random oracle).

   We remove the dependency on the random oracle and present a solution to the case where y is substituted in two polynomials.

3.2 Checking the Polynomials

Our set-intersection protocol utilizes a zero-knowledge proof of knowledge for the relation[2]

$$\mathcal{R}_{\text{POLY}} = \left\{ \left(\left(\{q_{i,j}\}_{i,j}, m_X, pk\right), \left(\{Q_{i,j}, r_{i,j}\}_{i,j}\right) \right) \middle| \begin{array}{c} \forall i,j \; q_{i,j} = E_{pk}(Q_{i,j}; r_{i,j}) \wedge \\ \sum_i \mathsf{deg}(Q_i(\cdot)) = m_X \wedge \\ \forall i, \; Q_i(\cdot) \not\equiv 0 \end{array} \right\}$$

where $i \in \{1, \ldots, B\}, j \in \{0, \ldots, M\}$.

[2] We will use the convention that the degree of a polynomial $Q_i(\cdot)$ can be chosen to be any integer j' such that $Q_{i,j} = 0$ for all $j \geq j'$, hence equality with m_X can always be achieved.

3.3 Secure Set-Intersection in the Presence of Malicious Adversaries

We now get to the main contribution of this work – a protocol that securely computes $\mathcal{F}_\cap$ in the presence of malicious adversaries, in the standard model; see Figure 1. The main ingredient is a subtle combination of an oblivious pseudo-random function evaluation protocol and a perfectly hiding commitment scheme. Somewhat counter intuitively, the oblivious PRF need not be committed (in a sense that the same key is being reused) – the proof of security shows that although party P_2 that controls the key may change it between invocations, this does not get him any advantage.

The oblivious PRF is used to save on using a (generic) zero-knowledge protocol for party P_2's adherence to the protocol. Recall that in the protocol of Freedman et al. P_1 learns for every $y \in Y$ two values: $r_0 \cdot Q_{h_0(y)}(y) + y$ and $r_1 \cdot Q_{h_1(y)}(y) + y$ where r_0, r_1 are randomly distributed. If $y \notin X$ then both values are random, and reveal no information about y. If, $y \in X$ then one of these values equals y. In our protocol, the 'payload' y of this computation is replaced by a secret s (meaning, P_1 learns two values: $r_0 \cdot Q_{h_0(y)}(y) + s$ and $r_1 \cdot Q_{h_1(y)}(y) + s$). The result of the polynomial evaluation step is, hence, that if $y \notin X$ then P_1 learns no information about s, and if $y \in X$ then P_1 learns s.

The crux of our construction is that the strings r_0, r_1 (as well as other) are not really random. These are pseudorandom strings that are directly derived from $F_{\text{PRF}}(k, s)$. What we get, is that if $y \notin X$ then P_1 learns nothing about s or y. If, on the other hand, $y \in X$ then P_1 learns s, and furthermore after P_1 invokes the oblivious PRF protocol, she can recover y and check that the computations P_2 performed based on the other 'random' strings were performed correctly.

A complication arises as P_2 (who selects the key k for the PRF) computes $F_{\text{PRF}}(k, s)$ by himself, and hence it is impossible for the simulator to extract s from this computation. We thus provide the simulator with an alternative means of extracting s (and also the corresponding y value) by having P_2 commit to both. To guarantee independence of inputs (i.e., that P_1 would not be able to choose his inputs depending on P_2's commitment or vise versa), this commitment is perfectly hiding and is performed *before* P_1 sends the encrypted polynomials representing her input set X.

We continue with a formal description of the protocol.

Protocol 2. ($\pi_\cap$ – secure set-intersection):

- **Inputs:** *The input of P_1 is m_Y and a set $X \subseteq \{0,1\}^{p(n)}$ containing m_X items; the input of P_2 is m_X and a set $Y \subseteq \{0,1\}^{p(n)}$ containing m_Y items (hence, both parties know m_X and m_Y).*
- **Auxiliary inputs:** *A security parameter 1^n, a prime q' such that $q' = 2q+1$ for a prime q. The group $\mathbb{G}$ is the subgroup of quadratic residues modulo q' and g is a generator of $\mathbb{G}$.*
- **Convention:** *Both parties check every received ciphertext for validity (i.e, that it is in $\mathbb{G}$), and abort otherwise.*

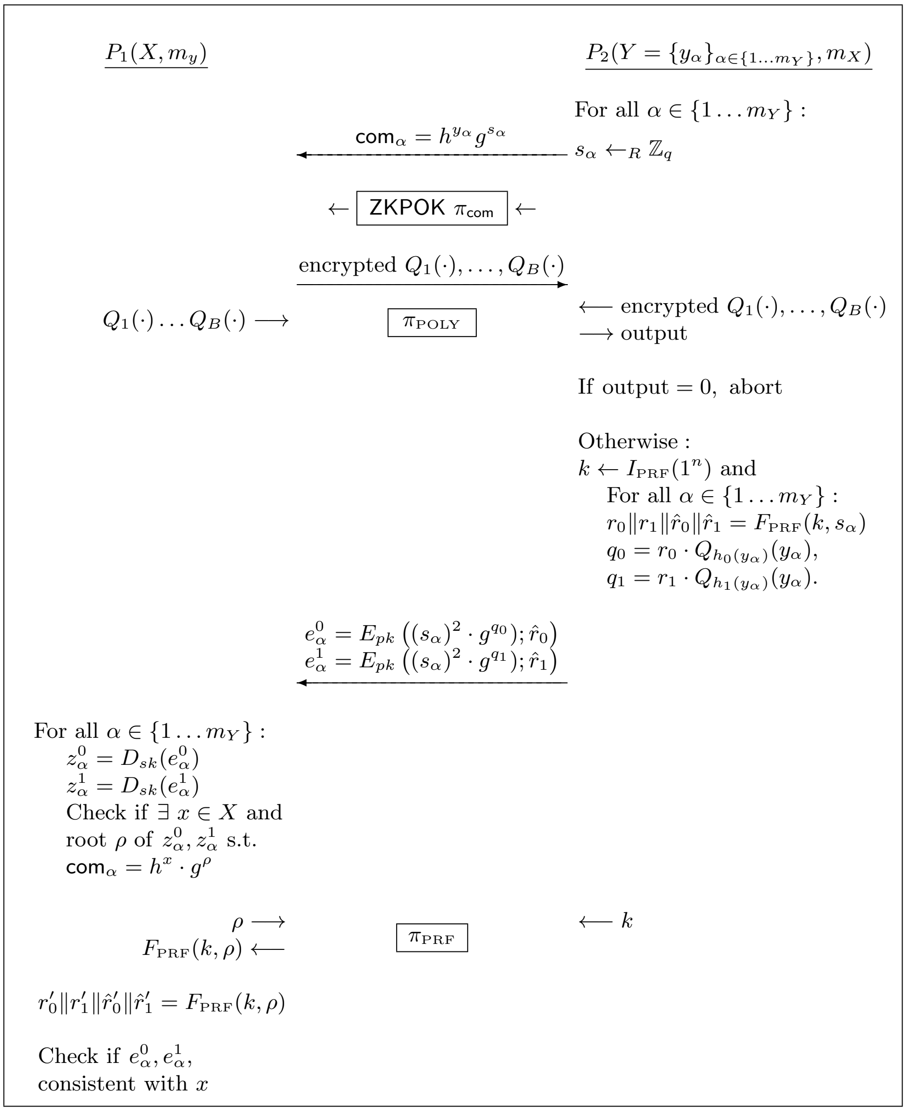

$P_1(X, m_y)$
$P_2(Y = \{y_\alpha\}_{\alpha \in \{1 \dots m_Y\}}, m_X)$
For all $\alpha \in \{1 \dots m_Y\}$:
$\mathsf{com}_\alpha = h^{y_\alpha} g^{s_\alpha}$
$s_\alpha \leftarrow_R \mathbb{Z}_q$
$\leftarrow \boxed{\text{ZKPOK } \pi_{\mathsf{com}}} \leftarrow$
encrypted $Q_1(\cdot), \dots, Q_B(\cdot)$
$Q_1(\cdot) \dots Q_B(\cdot) \longrightarrow$
$\boxed{\pi_{\text{POLY}}}$
$\longleftarrow$ encrypted $Q_1(\cdot), \dots, Q_B(\cdot)$
$\longrightarrow$ output
If output $= 0$, abort
Otherwise :
$k \leftarrow I_{\text{PRF}}(1^n)$ and
For all $\alpha \in \{1 \dots m_Y\}$:
$r_0 \| r_1 \| \hat{r}_0 \| \hat{r}_1 = F_{\text{PRF}}(k, s_\alpha)$
$q_0 = r_0 \cdot Q_{h_0(y_\alpha)}(y_\alpha)$,
$q_1 = r_1 \cdot Q_{h_1(y_\alpha)}(y_\alpha)$.
$e^0_\alpha = E_{pk}\left((s_\alpha)^2 \cdot g^{q_0}); \hat{r}_0\right)$
$e^1_\alpha = E_{pk}\left((s_\alpha)^2 \cdot g^{q_1}); \hat{r}_1\right)$
For all $\alpha \in \{1 \dots m_Y\}$:
$z^0_\alpha = D_{sk}(e^0_\alpha)$
$z^1_\alpha = D_{sk}(e^1_\alpha)$
Check if $\exists\, x \in X$ and
root ρ of z^0_α, z^1_α s.t.
$\mathsf{com}_\alpha = h^x \cdot g^\rho$
$\rho \longrightarrow$
$F_{\text{PRF}}(k, \rho) \longleftarrow$
$\boxed{\pi_{\text{PRF}}}$
$\longleftarrow k$
$r'_0 \| r'_1 \| \hat{r}'_0 \| \hat{r}'_1 = F_{\text{PRF}}(k, \rho)$
Check if e^0_α, e^1_α,
consistent with x

Fig. 1. A high-level diagram of $\pi_\cap$

- **The protocol:**
  1. **Key setup for the encryption and commitment schemes:** P_1 *chooses* $t, t' \leftarrow_R \mathbb{Z}_q$, *sets* $h = g^t, h' = g^{t'}$ *and sends* h, h' *to* P_2. *The key for the Pedersen commitment scheme is* h. *The public and private keys for the El Gamal scheme are* $pk = h'$ *and* $sk = t'$. P_1 *proves knowledge of* $\log_g h$ *and* $\log_g h'$ *using the zero-knowledge proof of knowledge for* $\mathcal{R}_{\text{DL}}$.

2. **Setting the balanced allocation scheme:** *P_1 computes the parameters B, M for the scheme and chooses the seeds for two (pseudo-)random hash functions $h_0, h_1 : \{0,1\}^{p(n)} \to [B]$. She sends B, M, h_0, h_1 to P_2 that checks that the parameters B, M were computed correctly, and aborts otherwise.*
3. **P_2 commits to his input set:** *Let $y_1, \ldots, y_{m_Y}$ be a random ordering of the elements of set Y. For all $\alpha \in \{1, \ldots, m_Y\}$, P_2 chooses $s_\alpha \leftarrow_R \mathbb{Z}_q$ and sends $\mathsf{com}_\alpha = \mathsf{com}(y_\alpha; s_\alpha) = h^{y_\alpha} g^{s_\alpha}$ to P_1. P_2 then proves the knowledge of y_α and s_α by invoking the zero-knowledge proof of knowledge for $\mathcal{R}_{\text{COM}}$.*
4. **P_1 Creates the polynomials representing her input set:** *For every $x \in X$, P_1 maps x into the less occupied bin from $\{h_0(x), h_1(x)\}$ (ties broken arbitrarily). Let $\mathcal{B}_i$ denote the set of elements mapped into bin i. P_1 constructs a polynomial $Q_i(x) \overset{\text{def}}{=} \sum_{j=0}^{M} Q_{i,j} \cdot x^j$ of degree at most M whose set of roots is $\mathcal{B}_i$.[3] P_1 encrypts the polynomials' coefficients, setting $q_{i,j} = E_{pk}(g^{Q_{i,j}}; r_{i,j})$, and sends the encrypted coefficients to P_2.*
5. **Checking the polynomials:** *P_1 and P_2 engage in a zero-knowledge execution π_{POLY} for which P_1 proves that the sets $\{q_{i,j}\}_{i\in\{1,\ldots,B\}, j\in\{0,\ldots,M\}}$ were computed correctly, using its witness $\{Q_{i,j}, r_{i,j}\}_{i\in\{1,\ldots,B\}, j\in\{0,\ldots,M\}}$. If the outcome is not 1 then P_2 aborts.*
6. **Evaluating the polynomials:** *P_2 chooses $k \leftarrow I_{\text{PRF}}(1^n)$. Then, P_2 performs the following for all $\alpha \in \{1, \ldots, m_Y\}$:*
   (a) P_2 computes $F_{\text{PRF}}(k, s_\alpha)$ and parses the result to obtain pseudorandom strings r_0, r_1, $\hat{r}_0$, $\hat{r}_1$ of appropriate lengths for their usage below (i.e., $r_0 \| r_1 \| \hat{r}_0 \| \hat{r}_1 = F_{\text{PRF}}(k, s_\alpha)$).
   (b) He sets $\hat{h}_0 = h_0(y_\alpha)$ and $\hat{h}_1 = h_1(y_\alpha)$.
   (c) He uses the homomorphic properties of the encryption scheme to evaluate $e_\alpha^0 = E_{pk}(((s_\alpha)^2 \bmod q') \cdot g^{r_0 \cdot Q_{\hat{h}_0}(y_\alpha)}; \hat{r}_0)$ and $e_\alpha^1 = E_{pk}(((s_\alpha)^2 \bmod q') \cdot g^{r_1 \cdot Q_{\hat{h}_1}(y_\alpha)}; \hat{r}_1)$ (where $\hat{r}_0, \hat{r}_1$ denote here the randomness used in the re-encryptions). Then he sends e_α^0, e_α^1 to P_1.
7. **Computing the intersection:** *For each $\alpha \in \{1, \ldots, m_Y\}$:*
   (a) P_1 computes $z_\alpha^0 = D_{sk}(e_\alpha^0)$ and $z_\alpha^1 = D_{sk}(e_\alpha^1)$. For each of the (up to four) roots ρ of z_α^0, z_α^1 (roots are computed modulo $q' = 2q+1$ and the result is considered only if it falls within $\mathbb{Z}_q$), she checks if $\mathsf{com}_\alpha / g^\rho$ coincides with h^{x_α} for some $x_\alpha \in X$ (this can be done efficiently by creating a hash table for set $\{h^x : x \in X\}$), and if this is the case sets $\hat{s}_\alpha$ to the corresponding root and marks α.
   (b) P_1 and P_2 engage in an execution of the protocol for F_{PRF}. If α is marked, then P_1 enters $\hat{s}_\alpha$ as input, and otherwise she enters a zero. P_2 enters k as input. Let $r_0' \| r_1' \| \hat{r}_0' \| \hat{r}_1'$ denote P_1's output from this execution.

[3] If $B_i = \emptyset$ then P_1 sets $Q_i(x) = 1$. Otherwise, if $|\mathcal{B}_i| < M$ then P_1 sets the $M+1-|\mathcal{B}_i|$ highest-degree coefficients of $Q_i(\cdot)$ to zero.

(c) *If α is marked, then P_1 checks that $e^0_\alpha = E_{pk}((\hat{s}_\alpha)^2 \cdot g^{r'_0 \cdot Q_{h_0(x_\alpha)}(x_\alpha)}; \hat{r}'_0)$, and $e^1_\alpha = E_{pk}((\hat{s}_\alpha)^2 \cdot g^{r'_1 \cdot Q_{h_1(x_\alpha)}(x_\alpha)}; \hat{r}'_1)$ result from applying the homomorphic operations on the encrypted polynomials and randomness r'_0, r'_1. If all checks succeed P_1 records x_α as part of her output.*

A word of explanation is needed for the computation done in Step 6. A natural choice for the payload is s_α itself. However, $s_\alpha \in \mathbb{Z}_q$ whereas the message space of the El Gamal encryption is $\mathbb{G}$. Noting that $\mathbb{Z}_q \subset \mathbb{Z}^*_{q'}$ (neglecting $0 \in \mathbb{Z}_q$), and that by squaring an element of $\mathbb{Z}_{q'}$ we get an element of $\mathbb{G}$, we get that treating s_α as an element of $\mathbb{Z}^*_{q'}$ and computing $(s_\alpha)^2 \bmod q'$ we get an element of $\mathbb{G}$. In this mapping, (up to) two elements of $\mathbb{Z}_q$ share an image in $\mathbb{G}$, and hence in Step 7a we need to recover and check both pre-images.

Before getting into the proof of security, we observe that if both parties are honest, then P_1 outputs $X \cap Y$ with probability negligibly close to one. In this case, if for an element $y_\alpha \in Y$ is holds that $y_\alpha \in X$ then one of $Q_{h_0(y_\alpha)}(y_\alpha)$, $Q_{h_1(y_\alpha)}(y_\alpha)$ is zero, and otherwise none of $Q_{h_0(y_\alpha)}(y_\alpha)$, $Q_{h_1(y_\alpha)}(y_\alpha)$ is zero. We get:

1. If $y_\alpha \in X \cap Y$ then one of e^0_α, e^1_α encrypts $(s_\alpha)^2 \bmod q'$. Hence, there exists a root ρ of z^0_α, z^1_α such that $\mathsf{com}_\alpha / g^\rho$ coincides with h^{x_α} for some $x_\alpha \in X$, resulting in P_1 marking α. Furthermore, as $r_0, r_1, \hat{r}_0, \hat{r}_1$ are derived from $F_{\text{PRF}}(k, s_\alpha)$, the check done by P_1 in Step 7 succeeds and P_1 records y_α in her output.
2. If $y_\alpha \notin X \cap Y$ then none of e^0_α, e^1_α encrypts $(s_\alpha)^2 \bmod q'$ and (except for a negligible probability) $\mathsf{com}_\alpha / g^\rho$ coincides with h^{x_α} for no root ρ of z^0_α, z^1_α and $x_\alpha \in X$. Hence, P_1 does not mark α, and y_α is not considered in Step 7, and not included in P_1's output.

Theorem 1. *Assume that π_{DL}, π_{PRF} and π_{POLY} are as described above, that (G, E, D) is the El Gamal encryption scheme, and that* com *is a perfectly-hiding commitment scheme. Then $\pi_\cap$ (Protocol 2) securely computes $\mathcal{F}_\cap$ in the presence of malicious adversaries.*

Efficiency. We first note that the protocol is constant round (as all its zero-knowledge proofs and subprotocols are constant round). The costs of using current implementations of F_{PRF} on inputs of length $p(n)$ is that of $p(n)$ oblivious transfer invocations [21], and hence of $O(p(n))$ modular exponentiations. We get that the overall communication costs are of sending $O(m_X + m_Y p(n))$ group elements, and the computation costs are of performing $O(m_X + m_Y(\log\log m_X + p(n)))$ modular exponentiations.

Optimizations. Note first that if the functionality is changed to allow P_2 learn the size of the intersection $m_{X\cap Y}$, then, in Step 7b, it is possible to avoid invoking π_{PRF} when α is not marked. This yields a protocol where $O(m_X + m_{X\cap Y} \cdot p(n))$ group elements are sent, and $O(m_X + m_Y \cdot \log\log m_X + m_{X\cap Y} \cdot p(n))$ exponentiation are computed. When $m_{X\cap Y} \ll m_Y$, this protocol is significantly more efficient than those suggested in [21] for weaker adversarial models. Furthermore, an improved scheme for oblivious pseudorandom function evaluation with overall complexity which is independent of the input length yields a better efficiency as well.

3.4 A Very Efficient Heuristic Construction

Note that we can now modify protocol $\pi_\cap$ to get a protocol in the random oracle model $\pi_\cap^{\mathrm{RO}}$ by performing the following two changes: (1) the computation of $F_{\mathrm{PRF}}(k, s_\alpha)$ performed by P_2 in Step 6a of $\pi_\cap$ is replaced with an invocation of the random oracle, i.e., P_2 computes $\mathcal{H}(s_\alpha)$; and (ii) the execution of the secure protocol for evaluating F_{PRF} by P_1 and P_2 in Step 7 of $\pi_\cap$ is replaced with an invocation of the random oracle by P_1, i.e., no communication is needed, and instead of providing s' to the protocol for F_{PRF}, P_1 computes $\mathcal{H}(s')$.

A typical proof of security in the random oracle model relies on the simulator's ability to record the inputs on which the random oracle is invoked, and the recorded information is used by the simulator for malicious P_2 while recovering his input. In other words, the proof of security relies on the property of the random oracle that the only way to learn any information about $\mathcal{H}(s)$ is to apply $\mathcal{H}$ on a *well defined input* s. Should $\pi_\cap^{\mathrm{RO}}$ be implemented such that the invocations of the random oracle are replaced by a concrete computation of some function, it seems that this proof of security would collapse, even if very strong hardness assumptions are made with respect to this implementation.

Nevertheless, the situation in protocol $\pi_\cap$ is very different. Note, in particular, that the simulator for malicious P_2 cannot monitor P_2's input to F_{PRF} (nor is this notion of inputs to the function well defined). Instead, the simulator extracts s from the zero-knowledge proof of knowledge for the commitment on P_2's inputs in Step 3 of $\pi_\cap$. This is inherited by the modified protocol $\pi_\cap^{\mathrm{RO}}$. Hence, should the random oracle calls in $\pi_\cap^{\mathrm{RO}}$ be replaced with some primitive Gen, the proof of security may still hold with small modifications, given the hardness assumption on Gen (intuitively, some functions of the outcome of $\mathsf{Gen}(s)$ and s should look random). $\pi_\cap^{\mathrm{RO}}$ can hence be viewed as an intermediate step between the protocol in [17] that utilizes a random oracle to cope with malicious parties, and the protocol suggested in the current paper. If the primitive Gen is realized efficiently (e.g., if its computation incurs a constant number of exponentiations), we get an extremely efficient protocol for $\mathcal{F}_\cap$, where the communication costs are of sending $O(m_X + m_Y)$ group elements, and the number of exponentiations is $O(m_X + m_Y \log\log m_X)$.

For the sake of completeness we include a formal description of protocol $\pi_\cap^{\mathsf{Gen}}$, that is identical to protocol $\pi_\cap$ except for the replacing every invocation of $F_{\mathrm{PRF}}(k, \cdot)$ by a computation of $\mathsf{Gen}(\cdot)$. Note that unlike in an invocation of $F_{\mathrm{PRF}}(k, \cdot)$, no communication is needed for computing $\mathsf{Gen}(\cdot)$.

Protocol 3. ($\pi_\cap^{\mathsf{Gen}}$ – secure set-intersection with a "generator"):

– **Inputs:** *The input of P_1 is m_Y and a set $X \subseteq \{0,1\}^{p(n)}$ containing m_X items; the input of P_2 is m_X and a set $Y \subseteq \{0,1\}^{p(n)}$ containing m_Y items (hence, both parties know m_X and m_Y).*

– **Auxiliary inputs:** *A security parameter 1^n, a prime q' such that $q' = 2q+1$ for a prime q. The group $\mathbb{G}$ is the subgroup of quadratic residues modulo q' and g is a generator of $\mathbb{G}$.*

- **Convention:** *Both parties check every received ciphertext for validity (i.e, that it is in $\mathbb{G}$), and abort otherwise.*
- **The protocol:**
  1. **Key setup for the encryption and commitment schemes:** P_1 *chooses $t, t' \leftarrow_R \mathbb{Z}_q$, sets $h = g^t, h' = g^{t'}$ and sends h, h' to P_2. The key for the Pedersen commitment scheme is h. The public and private keys for the El Gamal scheme are $pk = h'$ and $sk = t'$. P_1 proves knowledge of $\log_g h$ and $\log_g h'$ using the zero-knowledge proof of knowledge for $\mathcal{R}_{\text{DL}}$.*
  2. **Setting the balanced allocation scheme:** P_1 *computes the parameters B, M for the scheme and chooses the seeds for two (pseudo-)random hash functions $h_0, h_1 : \{0,1\}^{p(n)} \rightarrow [B]$. She sends B, M, h_0, h_1 to P_2 that checks that the parameters B, M were computed correctly, and aborts otherwise.*
  3. P_2 **commits to his input set:** *Let $y_1, \ldots, y_{m_Y}$ be a random ordering of the elements of set Y. For all $\alpha \in \{1, \ldots, m_Y\}$, P_2 chooses $s_\alpha \leftarrow_R \mathbb{Z}_q$ and sends $\mathsf{com}_\alpha = \mathsf{com}(y_\alpha; s_\alpha) = h^{y_\alpha} g^{s_\alpha}$ to P_1. P_2 then proves the knowledge of y_α and s_α by invoking the zero-knowledge proof of knowledge for $\mathcal{R}_{\text{COM}}$.*
  4. P_1 **Creates the polynomials representing her input set:** *For every $x \in X$, P_1 maps x into the less occupied bin from $\{h_0(x), h_1(x)\}$ (ties broken arbitrarily). Let $\mathcal{B}_i$ denote the set of elements mapped into bin i. P_1 constructs a polynomial $Q_i(x) \overset{\text{def}}{=} \sum_{j=0}^{M} Q_{i,j} \cdot x^j$ of degree at most M whose set of roots is $\mathcal{B}_i$.*[4] *P_1 encrypts the polynomials' coefficients, setting $q_{i,j} = E_{pk}(g^{Q_{i,j}}; r_{i,j})$, and sends the encrypted coefficients to P_2.*
  5. **Checking the polynomials:** *P_1 and P_2 engage in a zero-knowledge execution π_{POLY} for which P_1 proves that the sets $\{q_{i,j}\}_{i \in \{1,\ldots,B\}, j \in \{0,\ldots,M\}}$ were computed correctly, using its witness $\{Q_{i,j}, r_{i,j}\}_{i \in \{1,\ldots,B\}, j \in \{0,\ldots,M\}}$. If the outcome is not 1 then P_2 aborts.*
  6. **Evaluating the polynomials:** *For all $\alpha \in \{1, \ldots, m_Y\}$ P_2 performs the following :*
     (a) *He sets $\hat{h}_0 = h_0(y_\alpha)$ and $\hat{h}_1 = h_1(y_\alpha)$.*
     (b) *He parses $\mathsf{Gen}(s_\alpha)$ to obtain pseudorandom strings $r_1, r_2, \hat{r}_0, \hat{r}_1$ of appropriate lengths for their usage below (i.e., $r_1 \| r_2 \| \hat{r}_0 \| \hat{r}_1 = \mathsf{Gen}(s_\alpha)$).*
     He uses the homomorphic properties of the encryption scheme to evaluate $e_\alpha^0 = E_{pk}(((s_\alpha)^2 \bmod q') \cdot g^{r_0 \cdot Q_{\hat{h}_0}(y_\alpha)}; \hat{r}_0)$ and $e_\alpha^1 = E_{pk}(((s_\alpha)^2 \bmod q') \cdot g^{r_1 \cdot Q_{\hat{h}_1}(y_\alpha)}; \hat{r}_1)$ (where $\hat{r}_0, \hat{r}_1$ denote here the randomness used in the re-encryptions). Then he sends e_α^0, e_α^1 to P_1.
  7. **Computing the intersection:** *For each $\alpha \in \{1, \ldots, m_Y\}$:*
     (a) *P_1 computes $z_\alpha^0 = D_{sk}(e_\alpha^0)$ and $z_\alpha^1 = D_{sk}(e_\alpha^1)$. For each of the (up to four) roots ρ of z_α^0, z_α^1 (roots are computed modulo $q' = 2q+1$ and the*

[4] If $B_i = \emptyset$ then P_1 sets $Q_i(x) = 1$. Otherwise, if $|\mathcal{B}_i| < M$ then P_1 sets the $M+1-|\mathcal{B}_i|$ highest-degree coefficients of $Q_i(\cdot)$ to zero.

result is considered only if it falls within $\mathbb{Z}_q$), she checks if $\mathsf{com}_\alpha/g^\rho$ coincides with h^{x_α} for some $x_\alpha \in X$ (this can be done efficiently by creating a hash table for set $\{h^x : x \in X\}$), and if this is the case sets $\hat{s}_\alpha$ to the corresponding root and marks α.

(b) *If α is marked, then P_1 parses $\mathsf{Gen}(\hat{s}_\alpha)$ to obtain $r'_0, r'_1, \hat{r}'_0, \hat{r}'_1$.*

(c) *P_1 checks that $e^0_\alpha = E_{pk}((\hat{s}_\alpha)^2 \cdot g^{r'_0 \cdot Q_{h_0(x_\alpha)}(x_\alpha)}; \hat{r}'_0)$, and $e^1_\alpha = E_{pk}((\hat{s}_\alpha)^2 \cdot g^{r'_1 \cdot Q_{h_1(x_\alpha)}(x_\alpha)}; \hat{r}'_1)$ result from applying the homomorphic operations on the encrypted polynomials and randomness r'_0, r'_1. If all checks succeed P_1 records x_α as part of her output.*

References

1. Aggarwal, G., Mishra, N., Pinkas, B.: Secure Computation of the kth-Ranked Element. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 40–55. Springer, Heidelberg (2004)
2. Aumann, Y., Lindell, Y.: Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 137–156. Springer, Heidelberg (2007)
3. Azar, Y., Broder, A.Z., Karlin, A.R., Upfal, E.: Balanced Allocations. SIAM Journal on Computing 29(1), 180–200 (1999)
4. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non cryptographic fault tolerant distributed computations. In: 20th STOC, pp. 1–10 (1988)
5. Boneh, D., Goh, E., Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)
6. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols. In: 20th STOC, pp. 11–19 (1988)
7. 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)
8. 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)
9. Damgård, I., Jurik, M.: A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001)
10. Damgård, I., Nielsen, J.B.: Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 3–42. Springer, Heidelberg (2002)
11. Dachman-Soled, D., Malkin, T., Raykova, M., Yung, M.: Efficient Robust Private Set Intersection. In: Ghilardi, S. (ed.) ANCS 2009. LNCS, vol. 5479, pp. 125–142. Springer, Heidelberg (2009)
12. El Gamal, T.: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1984)
13. Feigenbaum, J., Ishai, Y., Malkin, T., Nissim, K., Strauss, M.J., Wright, R.N.: Secure multiparty computation of approximations. ACM Transactions on Algorithms (TALG) 2(3), 435–472 (2006)

14. Fouque, P., Pointcheval, D.: Threshold cryptosystems secure against chosen-ciphertext attacks. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 573–584. Springer, Heidelberg (2000)
15. Fouque, P., Poupard, G., Stern, J.: Sharing decryption in the context of voting of lotteries. In: Dingledine, R., Golle, P. (eds.) FC 2009. LNCS, vol. 5628, pp. 90–104. Springer, Heidelberg (2009)
16. 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)
17. Freedman, M., Nissim, K., Pinkas, B.: Efficient Private Matching and Set-Intersection. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 1–19. Springer, Heidelberg (2004)
18. Goldreich, O.: Foundations of Cryptography: Volume 2 – Basic Applications. Cambridge University Press, Cambridge (2004)
19. Goldreich, O., Kahan, A.: How To Construct Constant-Round Zero-Knowledge Proof Systems for NP. Journal of Cryptology 9(3), 167–190 (1996)
20. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: 19th STOC, pp. 218–229 (1987)
21. 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)
22. Hazay, C., Nissim, K.: Efficient Set Operations in the Presence of Malicious Adversaries. Cryptology ePrint Archive, Report 2009/594 (2009), http://eprint.iacr.org/
23. Jarecki, S., Liu, X.: Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 577–594. Springer, Heidelberg (2009)
24. Kiltz, E., Mohassel, P., Weinreb, E., Franklin, M.K.: Secure Linear Algebra Using Linearly Recurrent Sequences. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 291–310. Springer, Heidelberg (2007)
25. Kissner, L., Song, D.X.: Privacy-Preserving Set Operations. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 241–257. Springer, Heidelberg (2005); See technical report CMU-CS-05-113 for the full version
26. Lindell, Y., Pinkas, B.: Privacy Preserving Data Mining. Journal of Cryptology 15(3), 177–206 (2002)
27. Mohassel, P., Weinreb, E.: Efficient Secure Linear Algebra in the Presence of Covert or Computationally Unbounded Adversaries. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 481–496. Springer, Heidelberg (2008)
28. Naor, M., Nissim, K.: Communication preserving protocols for secure function evaluation. In: 33th STOC, pp. 590–599 (2001)
29. Naor, M., Reingold, O.: Number-Theoretic Constructions of Ecient Pseudo-Random Functions. In: 38th FOCS, pp. 231–262 (1997)
30. Nissim, K., Weinreb, E.: Communication Efficient Secure Linear Algebra. In: 4th TCC, pp. 522–541 (2006)
31. Okamoto, T.: Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg (1993)
32. 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)

33. Pedersen, T.P.: Non-Interactive and Information-Theoretical Secure Verifiable Secret Sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992)
34. 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)
35. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
36. Vocking, B.: How Asymmetry Helps Load Balancing. Journal of the ACM 50(4), 568–589 (2003)
37. Yao, A.C.: Protocols for secure computations. In: 23rd FOCS, pp. 160–164 (1982)

Text Search Protocols with Simulation Based Security

Rosario Gennaro[1], Carmit Hazay[2], and Jeffrey S. Sorensen[1]

[1] IBM T.J. Watson Research Center, Hawthorne, New York, USA
{rosario,sorenj}@us.ibm.com
[2] Dept. of Computer Science and Applied Mathematics,
Weizmann Institute and IDC, Israel
carmit.hazay@weizmann.ac.il

Abstract. This paper presents an efficient protocol for securely computing the fundamental problem of *pattern matching*. This problem is defined in the two-party setting, where party P_1 holds a pattern and party P_2 holds a text. The goal of P_1 is to learn where the pattern appears in the text, without revealing it to P_2 or learning anything else about P_2's text. Our protocol is the first to address this problem with *full security* in the face of malicious adversaries. The construction is based on a novel protocol for *secure oblivious automata evaluation* which is of independent interest. In this problem party P_1 holds an automaton and party P_2 holds an input string, and they need to decide if the automaton accepts the input, without learning anything else.

1 Introduction

Secure two-party computation is defined as the joint computation of some function over private inputs using a communications protocol, satisying at least *privacy* (no other information is revealed beyond the output of the function) and *correctness* (the correct output is computed). Today's standard definition (cf. [1] following [2,3,4]) formalizes security by comparing the execution of such a protocol to an "ideal execution" where a trusted third party helps the parties compute the function. Specifically, in the ideal world the parties just send their inputs over perfectly secure communication lines to a trusted party, who then computes the function honestly and sends the output to the designated party. Informally, the real protocol is defined to be secure if all adversarial attacks on a real protocol can also be carried out in the ideal world. In the ideal world, the adversary can do almost nothing, and this guarantees that the same is also true in the real world. This definition of security is often called *simulation-based* because security is demonstrated by showing that a real protocol execution can be "simulated" in the ideal world.

Secure two-party computation has been extensively studied, and it is known that any efficient two-party functionality can be securely computed [5,6,7]. However, these are just feasibility results that demonstrate secure computation is possible, in principle, though not necessarily in practice. One reason is that the results mentioned above are generic, i.e. they do not exploit any structural properties of the specific function being computed. A long series of research efforts has been focused on finding efficient protocols for specific functions: constructing such protocols is crucial if secure computation is ever to be used in practice.

Our Contribution. In this paper we focus on the following problems:

– *Secure Pattern Matching.* We look at the basic problem of *pattern matching.* In this problem, one party holds a text T and the other party holds a pattern p, but $|T|$ and $|p|$ are mutually known. The aim is for the party holding the pattern to learn all the locations of the pattern in the text (and there may be many) while the other party learns nothing about the pattern.
– *Oblivious Automata Evaluation.* To solve the above problem we consider the approach of [8] which reduces the pattern matching problem to the composition of a pattern-specific automaton Γ with the text T. We develop a protocol for securely computing the evaluation of Γ on T.

The problem of pattern matching has been widely studied for decades due to its numerous applications. Yet, the problem of pattern matching in a secure setting has not received similar attention. Our starting point is an extremely efficient protocol that computes this function in the "honest-but-curious" setting.[1] This solution can be extended for one-sided simulation, or security even with corruption of the party with the pattern. This first protocol is independent of out protocol for the malicious setting, and is comparable to the one-sided simulatable protocol of [9]. However, while both protocols reach the same asymptotic complexity our protocol is much more practical since the concrete constants that are involved are much smaller ([9] requires $|p|$ oblivious transfers). Moreover, our protocol can be easily extended to address related problems such as approximate text search or text search with wildcards. The malicious setting introduces many subtleties beyond those considered in the previous settings and requires the use of a different technique. This includes the introduction of novel sub-protocols such as, for example, a protocol to prove that a correct pattern-specific automaton was constructed. We note that our protocols are the first efficient ones in the literature to achieve full simulatability for these problems with malicious adversaries. Security is based on the El Gamal encryption scheme [10,11] and thus requires a relatively small security parameter (although any additively homomorphic threshold encryption with secure two-party distributed protocols to generate shared keys and perform decryptions would work).

Motivation. Consider a hospital holding a DNA database of all the participants in a research study, and a researcher wanting to determine the frequency of the occurrence of a specific gene. This is a classical pattern matching application, which is however complicated by privacy considerations. The hospital may be

[1] In this setting, an adversary follows the protocol specification but may try to examine the messages it receives to learn more than it should.

forbidden from releasing the DNA records to a third party. Likewise, the researcher may not want to reveal what specific gene she is looking for, nor trust the hospital to perform the search correctly.

It would seem that basic honest-but-curious solutions (already present in the literature, see below) would work here. However, the parties may be motivated to produce invalid results, so a proof of accurate output might be as important as the output itself. Moreover, there is also a need to make sure that the data on which the protocol is run is valid. For example, a rogue hospital could sell "fake" DNA databases for research purposes. Perhaps some trusted certification authorities might one day pre-certify a database as being valid for certain applications. Then, the security properties of our protocol could guarantee that only valid data is used in the pattern matching protocol. (The first step of our protocol is for the hospital to publish an encryption of the data, this could be replaced by publication of encrypted data that was certified as correct.)

Related work. The problem of secure pattern matching was studied by Hazay and Lindell in [9] who used oblivious pseudorandom function (PRF) evaluation to evaluate every block of size m bits. However, their protocol achieves only a weaker notion of security called one-sided simulatability which guarantees privacy in all cases and requires that one of the two parties is never corrupted to guarantee correctness. It is tempting to think that a protocol for computing oblivious PRF evaluation with a committed key (where it is guaranteed that the same key is used for all PRF evaluations) for malicious adversaries [12] suffices for malicious security. Unfortunately, this is not the case since the inputs for the PRF must be consistent and it is not clear how to enforce this. Namely, for every i, the last $m-1$ bits of the ith block are supposed to be the first $m-1$ bits of the following block. The idea to use oblivious automata evaluation to achieve secure pattern matching originates in [13]. Their protocols, however, are only secure in the honest-but-curious setting. We improve their results to tolerate a malicious adversary.

The efficiency of our protocol. When presenting a two-party protocol for the secure computation of a specific function, one has to make sure that the resulting protocol is indeed more efficient than the known "generic" solutions for secure two-party computation of *any* function. We compare our protocols to the two most efficient generic two-party protocols secure against malicious adversaries, both based on the circuit garbling-technique by Yao [5]. Recall that Yao's protocol (which is secure against semi-honest players) uses a Boolean circuit to compute the fuction, and its computational complexity is linear in the size of the circuit.

– One result, in [14], to make Yao resistant to malicious adversaries uses a binary cut-and-choose strategy. This requires running s copies of Yao's protocol, where s is a statistical security parameter that must be large enough so that $2 * 2^{\frac{-s}{17}}$ is sufficiently small. This requires $O(s|C| + s^2 m)$ symmetric-key encryptions, and this communications overhead, as shown by [15], is a major obstacle.

- The other general result from [16] uses a special form of encryption for garbling and performs efficient zero-knowledge (ZK) proofs over that encryption scheme. The protocol requires a common reference string (CRS) which consists of a strong RSA modulus. To the best of our knowledge, there are currently no efficient techniques for generating a shared strong RSA modulus without incorporating external help. Furthermore, their protocol requires approximately 720 RSA exponentiations per gate where these operations are computed modulo 2048 due to the use of Paillier's encryption scheme. This means that the bandwidth of [16] is relatively high as well.

2 Tools and Definitions

Throughout the paper, we denote the security parameter by n. Although not explicitly specified, input lengths are always assumed to be bounded by some polynomial in n. A probabilistic machine is said to run in *polynomial-time* (PPT) if it runs in time that is polynomial in the security parameter n alone.

A function $\mu(\cdot)$ is *negligible* in n (or simply *negligible*) if for every polynomial $p(\cdot)$ there exists a value N such that $\mu(n) < \frac{1}{p(n)}$ for all $n > N$; i.e., $\mu(n) = n^{-\omega(1)}$. Let $X = \{X(n)\}_{n \in N, a \in \{0,1\}^*}$ and $Y = \{Y(n)\}_{n \in N, a \in \{0,1\}^*}$ be distribution ensembles. We say that X and Y are *computationally indistinguishable*, denoted $X \overset{c}{\equiv} Y$, if for every polynomial non-uniform distinguisher D there exists a negligible $\mu(\cdot)$ such that

$$\Big|\Pr[D(X(n,a)) = 1] - \Pr[D(Y(n,a)) = 1]\Big| < \mu(n)$$

for every $n \in N$ and $a \in \{0,1\}^*$.

Due to space considerations we defer the formal definitions for two-party secure computations in the presence of malicious adversaries and one-sided simulation security to the extended version of this paper.

Zero-knowledge proofs. Our protocols use the several standard zero-knowledge proofs, as summarized in Table 1. We also employ the following additional zero-knowledge proofs.

1. A zero-knowledge proof of knowledge π_{ENC} for the following relation: Let $C_i = [c_{i,1}, ..., c_{i,m}]$ for $i \in \{0,1\}$ and $C' = [c'_1, ..., c'_m]$ be three vectors of m ciphertexts each. We want to prove that C' is the "re-encryption" of the

Table 1. Zero knowledge proofs referenced by our protocols

Protocol	Relation/Language	Reference
π_{DL}	$\mathcal{R}_{\text{DL}} = \{((\mathbb{G}, g, h), x) \mid h = g^x\}$	[17]
π_{DDH}	$\mathcal{R}_{\text{DDH}} = \{((\mathbb{G}, g, g_1, g_2, g_3), x) \mid g_1 = g^x \wedge g_3 = g_2^x\}\}$	[18]
π_{NZ}	$\text{L}_{\text{NZ}} = \{(\mathbb{G}, g, h, \langle\alpha, \beta\rangle) \mid \exists\ (m \neq 0, r) \text{ s.t. } \alpha = g^r,\ \beta = h^r g^m\}$	[19]

same messages encrypted in either C_0 or C_1, or, in other words, that there exists an index $i \in \{0,1\}$ such that for all j, c'_j was obtained by multiplying $c_{i,j}$ by a random encryption of 0. More formally,

$$\mathcal{R}_{\text{ENC}} = \left\{(\mathbb{G}, g, m, C_0, C_1, C'), (i, \{r_j\}_j)\right| \text{s.t. for all } j : c'_j = c_{i,j} \cdot_G E_{pk}(0; r_j)\right\}.$$

This involves the parties computing the sets $c_0 = \left\{\prod (c_{0,j} \cdot_G (1/c'_j))^{r_{0,j}}\right\}_{j=1}^{Q}$ and $c_1 = \left\{\prod (c_{1,j} \cdot_G (1/c'_j))^{r_{1,j}}\right\}_{j=1}^{Q}$, where the sets $\{r_{0,j}\}_j$ and $\{r_{1,j}\}_j$ are public randomness. The prover then proves that either $\langle pk, c_0\rangle$ or $\langle pk, c_1\rangle$ is a Diffie-Hellman tuple.

2. Let $C = \{c_{i,j}\}_{j,i}$ and $C' = \{c'_{i,j}\}_{j,i}$ be two sets of encryptions, where $j \in \{1, \ldots, |Q|\}$ and $i \in \{0,1\}$. Then we consider a zero-knowledge proof of knowledge π_{PERM} for proving that C and C' correspond to the same decryption vector up to some random permutation. Meaning that,

$$\mathcal{R}_{\text{PERM}} = \left\{\left(pk, C, C'\right), \left(\pi, \{r_{j,i}\}_{j,i}\right) | \forall\ i,\ \{c_{j,i} = c'_{\pi(j),i} \cdot E_{pk}(0; r_{j,i})\}_j\right\}$$

where π is a random one-to-one mapping over the elements $\{1, \ldots, |Q|\}$. Basically we prove that C' is obtained from C by randomizing all the ciphertexts and permuting the indices (i.e., the columns). We require that the same permutation is applied for both vectors. The problem in which a single a vector of ciphertexts is randomized and permuted is defined by

$$\mathcal{R}^1_{\text{PERM}} = \left\{\left((c_1, \ldots, c_Q), (\tilde{c}_1, \ldots, \tilde{c}_Q), pk\right),\right.$$
$$\left.\left(\pi, (r_1, \ldots, r_Q)\right) | \forall\ i, \tilde{c}_j = c_{\pi(j)} \cdot E_{pk}(0; r_j)\right\}.$$

and has been widely studied. The state-of-the-art protocol is in [20]. We use a simpler, though slightly less efficient (but still good for our purposes) protocol, from [21], which is an efficient zero-knowledge proof π^1_{PERM} for $\mathcal{R}^1_{\text{PERM}}$ with linear computation and communication complexity and constant number of rounds. We use this slightly less efficient protocol because its proof proof applies to the case where the same permutation is applied to multiple vectors of ciphertexts.

3 Secure Text Search Protocols

Text search involves scanning a text sequentially, looking for instances of a particular pattern. Efficient text search requires analysis of the pattern string to enable $O(\ell)$ scanning that skips over regions of text whenever possible matches are provably not possible, as in KMP [8] which we employ here.

Pattern matching is defined as follows: given a binary string T of length ℓ and a binary pattern p of length m, find all the locations in the text where pattern p appears in the text. Stated differently, for every $i = 1, \ldots, \ell - m + 1$, let T_i

be the substring of length m that begins at the ith position in T. Then, the basic problem of pattern matching is to return the set $\{i \mid T_i = p\}$. Formally, we consider the functionality $\mathcal{F}_{\text{PM}}$ defined by

$$(p,(T,m)) \mapsto \begin{cases} (\{i \mid T_i = p\}, \lambda) & \text{if } |p| = m \\ (\{i \mid T_i = p_1 \ldots p_m\}, \lambda) & \text{otherwise} \end{cases}$$

Note that P_2, who holds the text, learns nothing about the pattern held by P_1, and the only thing that P_1 learns about the text held by P_2 is the locations where its pattern appears. Our starting point is an extremely efficient protocol that computes $\mathcal{F}_{\text{PM}}$ in the "honest-but-curious" setting, where the adversary follows the protocol specification but tries to gain useful information about the honest party's input.

3.1 "Honest-But-Curious" Secure Text Search

The protocol shown in Fig. 1 uses homomorphic encryption to ensure the privacy of the two parties' inputs. Informally, party P_1 computes a matrix Φ of size $2 \times m$ that includes an encryption of zero in position (i,j) if $p_j = i$ or an encryption of one otherwise. Given Φ, party P_2 creates a new encryption e_k for every text location k that corresponds to the product of the encryptions at locations (t_{k+j-1}, j) for all $j \in \{1,\ldots,m\}$. Since e_k is the Hamming distance between p and T_k, iff p matches T_k, e_k will be a random encryption of zero. We remark that even though we consider here the problem of exact matching, this solution can be easily applied to the problem of text search with mismatches, or for larger alphabets.

Formally,

Protocol 1. $\boldsymbol{\pi}_{\text{SIMPLE}}$

- **Inputs:** The input of P_1 is a binary search string $p = p_1,\ldots,p_m$ and P_2 a binary text string $T = t_1,\ldots,t_\ell$

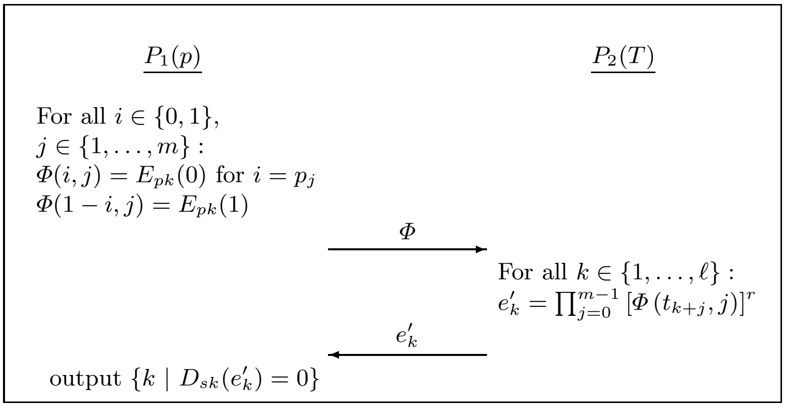

$P_1(p)$
$P_2(T)$
For all $i \in \{0,1\}$,
$j \in \{1,\ldots,m\}$:
$\Phi(i,j) = E_{pk}(0)$ for $i = p_j$
$\Phi(1-i,j) = E_{pk}(1)$
Φ
For all $k \in \{1,\ldots,\ell\}$:
$e'_k = \prod_{j=0}^{m-1} [\Phi(t_{k+j}, j)]^r$
e'_k
output $\{k \mid D_{sk}(e'_k) = 0\}$

Fig. 1. Text search in the honest-but-curious setting

– **Conventions:** The parties jointly agree on a group $\mathbb{G}$ of prime order q and a generator g for the El Gamal encryption. Party P_1 generates a key pair $(pk, sk) \leftarrow G$ and publishes pk. Finally, unless written differently, $j \in \{1, \ldots, m\}$ and $i \in \{0, 1\}$.
– **The protocol:**
  1. **Encryption of pattern.** Party P_1 builds a $2 \times m$ matrix of ciphertexts Φ defined by,
  $$\Phi(i,j) = \begin{cases} E_{pk}(0;r) & p_j = i \\ E_{pk}(1;r) & \text{otherwise} \end{cases}$$
  where each r is a uniformly chosen random value of appropriate length. The matrix Φ is sent to party P_2.
  2. **Scanning of text.** For each offset $k \in \{1, \ldots, \ell - m + 1\}$, P_2 computes
  $$e_k = \prod_{j=1}^{m} \Phi\left(t_{k+j-1}, j\right)$$
  Then for each offset k, it holds that T_k matches pattern p if and only if $e_k = E_{pk}(0)$.
  3. **Masking of terms.** Due to the fact that the decryption of e_k reveals the number of matched elements at text location k, party P_2 masks this result through scalar multiplication. In particular, P_2 sends the set $\{e'_k = (e_k)^{r_k}\}_k$ where r_k is a random string chosen independently for each k.
  4. **Obtaining result.** P_1 uses sk to decrypt the values of e'_k and obtains
  $$\{k \mid D_{sk}(e'_k) = 0\}$$

Clearly, if both parties are honest then P_1 outputs a correct set of indexes with overwhelming probability (an error may occur with negligible probability if $(e_k)^r$ is an encryption of zero even though e_k is not), as the parties execute a naive solution for $\mathcal{F}_{\text{PM}}$. Then we state the following,

Theorem 1. *Assume that (G, E, D) is the semantically secure El Gamal encryption scheme. Then protocol π_{SIMPLE} securely computes $\mathcal{F}_{\text{PM}}$ in the presence of honest-but-curious adversaries.*

The proof is straightforward via a reduction to the security of (G, E, D) and is therefore omitted.

Furthermore, if party P_1 proves that it computed matrix Φ correctly, we can also guarantee full simulation with respect to a corrupted P_1. This can be achieved by having P_1 prove, for every j, $\Phi(0,j), \Phi(1,j)$ is a permuted pair of the encryptions $E_{pk}(0), E_{pk}(1)$, using π_{PERM}. Constructing a simulator for the case of a corrupted P_2 is more challenging since the protocol does not guarantee that P_2 computes $\{e'_k\}_k$ relative to a well defined bit string p. In particular, it may compute every encryption e'_k using a different length m string. Thus, only privacy is guaranteed for this case. Let π'_{SIMPLE} denote the modified version of π_{SIMPLE} with the additional zero-knowledge proof of knowledge π_{PERM} of P_1. We conclude with the following claim,

Theorem 2. *Assume that (G, E, D) is the semantically secure El Gamal encryption scheme. Then protocol π'_{SIMPLE} securely computes $\mathcal{F}_{\text{PM}}$ with one-sided simulation.*

The proof sketch is in the extended version of this paper.

Efficiency. We first note that the protocol π'_{SIMPLE} is constant round. The overall communication costs are of sending $O(m+\ell)$ group elements, and the computation costs are of performing $O(m+\ell)$ modular exponentiations, as P_1 sends the table Φ and P_2 replies with a collection of ℓ encryptions. The additional cost of π_{PERM} is linear in the length of the pattern.

The fact that this protocol does not seem to be naturally extendable for the malicious setting has led us to search for the techniques presented in the next section.

3.2 Secure Text Search in the Presence of Malicious Adversaries

We consider a secure version of the KMP algorithm [8], that searches for occurrences of a pattern p within a text T by employing the observation that when a mismatch occurs, the pattern itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters. More formally, P_1, whose input is a pattern p, constructs an automaton Γ_p as follows: We denote by $p_{\langle i \rangle}$ the length-i prefix $p_1, \ldots, p_i$ of p. P_1 first constructs a table Υ with m entries where its ith entry denotes the largest prefix of p that matches a suffix of $p_{\langle i-1 \rangle}$. This table (as in Fig. 2) maintains the appropriate partial match state if a mismatch occurs in the ith bit of p. The algorithm indicates for each bit of the text, every input that reaches the final state, or one bit for each text location.

			Fail	$\Gamma(q_i, j)$	
State	Prefix	$\Upsilon(q_i)$	State	$j=0$	$j=1$
q_1			q_1	q_1	q_2
q_2	1		q_1	$\Gamma(q_1, 0)$	q_3
q_3	11	1	q_2	q_4	$\Gamma(q_2, 1)$
q_4	110		q_1	q_5	$\Gamma(q_1, 1)$
q_5	1100		q_1	q_6	$\Gamma(q_1, 1)$
q_6	11000		q_1	$\Gamma(q_1, 0)$	q_7
q_7	110001	1	q_2	$\Gamma(q_2, 0)$	q_8
q_8	1100011	11	q_3	q_9	$\Gamma(q_3, 1)$
q_9	11000110	110	q_4	q_{10}	$\Gamma(q_4, 1)$
q_{10}	110001100	1100	q_5	$\Gamma(q_5, 0)$	q_{11}
q_{11}	1100011001	1	q_2	$\Gamma(q_2, 0)$	$\Gamma(q_2, 1)$

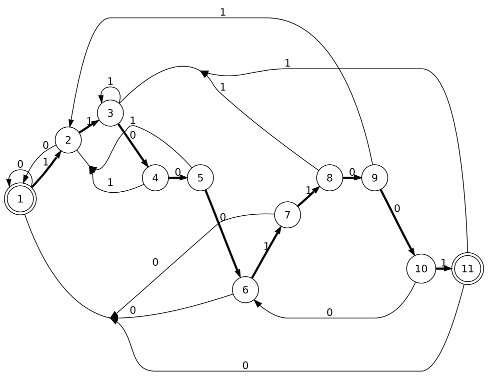


Fig. 2. Construction of determinized KMP automaton for pattern 1100011001

We remark that Υ can be easily constructed in time $O(m^2)$ by comparing p against itself at every alignment. P_1 constructs its automaton Γ_p based on Υ. It first sets $|Q| = |p| + 1$ and constructs the transition table Δ as follows. For all $i \in \{1, \ldots, m\}$, $\Delta(q_{i-1} \times p_i) \rightarrow q_i$ and $\Delta(q_{i-1} \times (1 - p_i)) \rightarrow \Upsilon(i)$ where $\Upsilon(i)$ denotes the ith entry in Υ (we denote the labels of the states in Q by the sequential integers starting from 1 to $m + 1$. This way, if there is no matching prefix in the ith entry, the automaton goes back to the initial state q_1). P_1 concludes the construction by setting $F = q_m$.

In the next section we show a general protocol evalue to perform a *secure* and *oblivious* evaluation of P_1's automaton on P_2's text. The protocol works for any automaton (not just a KMP one) and therefore may be of independent interest. After showing the automata evaluation protocol, we also show how to prove in zero-knowledge that the automaton P_1 constructs is a correct KMP automaton.

3.3 Secure Oblivious Automata Evaluation

In this section we present a secure protocol for oblivious automata evaluation in the presence of malicious adversaries. In this functionality P_1 inputs a description of an automaton Γ, and P_2 inputs a string t. The result of the protocol is that P_1 receives $\Gamma(t)$, while P_2 learns nothing. Formally, we define this problem via the functionality

$$\mathcal{F}_{\text{AUTO}} : (\Gamma = (Q, \Sigma, \Delta, q_1, F), t) \mapsto \begin{cases} (\mathsf{accept}, \lambda) & \text{if } \Gamma(t) \in F \\ (\mathsf{no-accept}, \lambda) & \text{otherwise} \end{cases}$$

where λ is the empty string (denoting that P_2 does not receive an output) and $\Gamma(t)$ denotes the final state in the evaluation of Γ on t. For a binary inpuy, the transition table contains $|Q|$ rows and two columns. Furthermore, we assume that the names of the states are the integers $\{1, \ldots, |Q|\}$. For simplicity, we assume that $|Q|$ and $|F|$ are public. This is due to the fact that this information is public anyway when reducing the problem of pattern matching into oblivious polynomial evaluation. For the sake of generality we note making $|F|$ private again can be easily dealt by having P_1 send a vector of encryptions for which the ith encryption is a zero encryption only if $q_i \notin F$. Otherwise, it is an encryption of q_i (this can be verified using a simple zero-knowledge proof). The final verification can be done by checking membership in this set using techniques from below.

Recall that our starting point is the protocol from [13]. Their idea is to have the parties share the current machine state, such that by the end of the kth iteration the party with the automaton knows a random string r^k whereas the party with the text learns $q^k + r^k$. The parties complete each iteration by running an oblivious transfer in which the next state is now shared between them. The honest-but-curious setting significantly simplifies their construction. Unfortunately, we cannot see any natural way to extend their technique into the malicious case (even using oblivious transfers resilient to malicious attacks).

A high level description. We begin by briefly motivating our construction; see Fig. 3 as well. Loosely speaking, at the beginning of the protocol P_1 and P_2 jointly

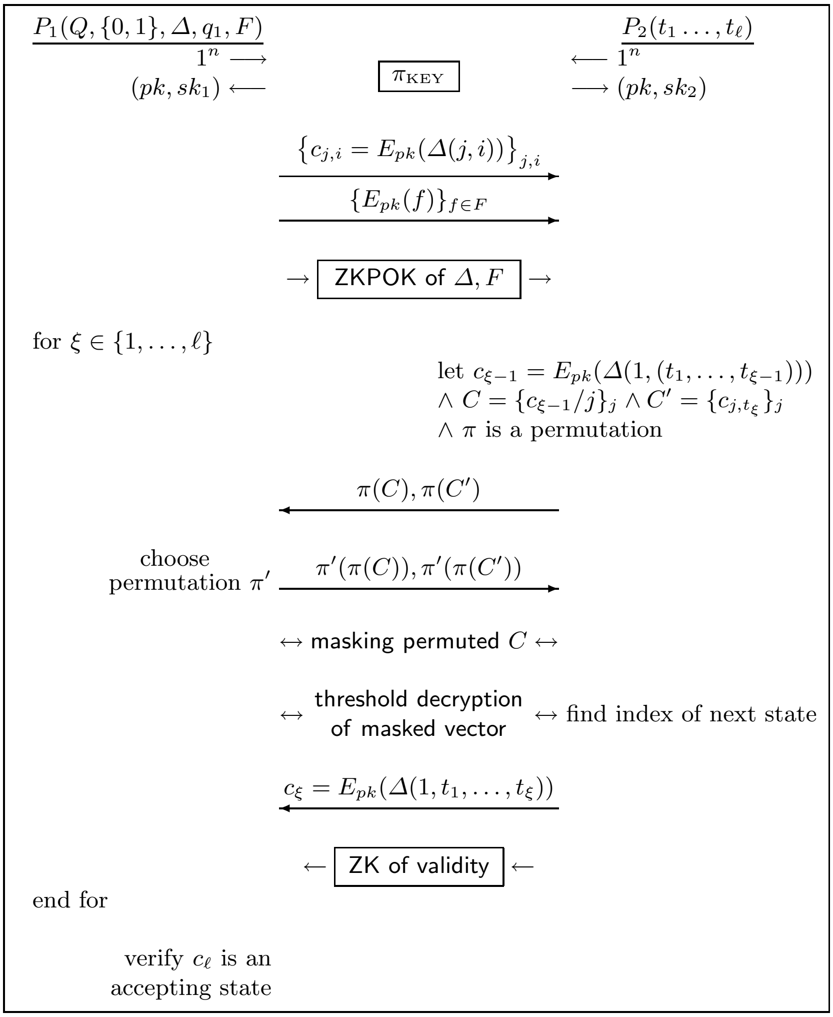

$P_1(Q, \{0,1\}, \Delta, q_1, F)$
$P_2(t_1 \ldots, t_\ell)$
$1^n \longrightarrow$
$(pk, sk_1) \longleftarrow$
π_{KEY}
$\longleftarrow 1^n$
$\longrightarrow (pk, sk_2)$
$\{c_{j,i} = E_{pk}(\Delta(j,i))\}_{j,i}$
$\{E_{pk}(f)\}_{f \in F}$
ZKPOK of Δ, F
for $\xi \in \{1, \ldots, \ell\}$
let $c_{\xi-1} = E_{pk}(\Delta(1, (t_1, \ldots, t_{\xi-1})))$
$\wedge\ C = \{c_{\xi-1}/j\}_j \wedge C' = \{c_{j,t_\xi}\}_j$
$\wedge\ \pi$ is a permutation
$\pi(C), \pi(C')$
choose permutation π'
$\pi'(\pi(C)), \pi'(\pi(C'))$
masking permuted C
threshold decryption of masked vector
find index of next state
$c_\xi = E_{pk}(\Delta(1, t_1, \ldots, t_\xi))$
ZK of validity
end for
verify c_ℓ is an accepting state

Fig. 3. A high-level diagram of π_{AUTO}

generate a public-key (G, E, D) for the threshold El Gamal encryption scheme (denoted by the sub-protocol π_{KEY}). Next, party P_1 encrypts its transition table Δ and the set of accepting states F, and sends it to P_2. This allows P_2 to find the encryption of the next state $c_1 = \Delta(1, t_1)$, by selecting it from the encrypted matrix. P_2 re-randomizes this encryption and shows it to P_1. The protocol continues in this fashion for the whole text with ℓ iterations.[2]

[2] Unfortunately, these iterations are not independent. Each requires an encryption of the current state, and thus cannot be performed in parallel. We show in Sect. 4 how to minimize the number of rounds into $O(|Q|)$ when performing a secure text search, which is typically quite small.

At the beginning of each iteration, the parties know a randomized encryption of the current state, and their goal is to find an encryption of the next state. At iteration i, P_2 selects from the matrix the entire encrypted column of all possible $|Q|$ next states for its input t_i (only knowing an encryption of the current state). Then, using the homomorphic properties of El Gamal , the parties obliviously select the correct next state: Let $c_{\xi-1}$ denote an encryption of $\Delta(1, t_1, \ldots, t_{\xi-1})$. The parties compute first the set $C = \{c_{\xi-1}/c_{j,\epsilon}\}_j$, where $\{c_{j,\epsilon}\}_j$ correspond to encryptions of the labels in Q; see below for more details. Only one ciphertext in this set will be an encryption of 0, it indicates the position corresponding to the current state. The protocol concludes by the parties jointly checking if the encrypted state that is produced within the final iteration is in the encrypted list of accepting states.

There are several technical challenges in constructing such a secure protocol. In particular the identification of the next encrypted state without leaking additional information requires a couple of rounds of interaction between the parties in which they mask and permute the ciphertext vector containing all possible states, in order to "destroy any link" between their input and the next encrypted state. Moreover, in order to protect against malicious behavior, zero-knowledge proofs are included at each step to make sure parties behave according to the protocol specifications. We are now ready to present a formal description of our protocol.

For a final remark we denote that due to technicalities that arise in the security proof, our protocol employs an unnatural masking technique, where instead of multiplying or adding a random value to each encryption, it uses both. The reason for this becomes clear in the proof.

Protocol 2. π_{AUTO}

- **Inputs:** The input of P_1 is a description of an automaton $\Gamma = (Q, \{0,1\}, \Delta, q_1, F)$, and the input of P_2 is a binary string $t = t_1, \ldots, t_\ell$.
- **Auxiliary Inputs:** $|Q|$ and $|F|$ for P_2, ℓ for P_1, and the security parameter 1^n for both.
- **Conventions:** We assume that the parties jointly agree on a group $\mathbb{G}$ of prime order q and a generator g for the threshold El Gamal encryption scheme. Both parties check every received ciphertext for validity, and abort if an invalid ciphertext is received.

  The transition table Δ is defined as $\Delta(j, i)$ which denotes the state that follows state j if the input letter is i. We augment this table by adding a column to Δ labeled by ϵ: we define $\Delta(j, \epsilon) = j$ (the reader can think of it as the "label" of the jth row of the transition table).

  Since we are assuming a binary alphabet, unless written differently, $j \in \{1, \ldots, |Q|\}$ and $i \in \{\epsilon, 0, 1\}$. Finally we assume that the initial state is labeled 1.
- **The protocol:**
  1. **El Gamal key setup:**
     (a) Party P_1 chooses a random value $r_1 \leftarrow_R \mathbb{Z}_q$ and sends party P_2 the value $g_1 = g^{r_1}$. It proves knowledge of r_1 using π_{DL}.

(b) Party P_2 chooses a random value $r_2 \leftarrow_R \mathbb{Z}_q$ and sends party P_1 the value $g_2 = g^{r_2}$. It proves knowledge of r_2 using π_{DL}. The parties set $pk = \langle \mathbb{G}, q, g, h = g_1 \cdot g_2 \rangle$ (i.e., the secret key is $(r_1 + r_2) \bmod q$).

2. **Encrypting P_1 transition table and accepting states:**

   (a) P_1 encrypts its (augmented) transition table $\varDelta$ under pk component-wise; $\varDelta_{\mathsf{E}} = \left\{c_{j,i} = E_{pk}(\varDelta(j,i))\right\}_{j,i}$. Notice that $c_{j,\epsilon}$ is an encryption of the state j. P_1 also sends the list of encrypted accepting states denoted by $\mathsf{E}(F) = \{E_{pk}(f)\}_{f \in F}$.

   (b) For every encryption $\langle c_1, c_2 \rangle \in \varDelta_{\mathsf{E}} \cup \mathsf{E}(F)$, P_1 proves the knowledge of $\log_g c_1$ using π_{DL}.

   (c) *Proving the validity of the encrypted transition matrix.* P_1 proves that $\varDelta_E$ is a set of encryptions for values from the set $\{1, \ldots, |Q|\}$. It first sorts the encryptions according to their encrypted values, denoted by $c_1, \ldots, c_{3 \cdot |Q|}$. P_1 multiplies every encryption in this set with a random encryption of 0, sends it to P_2 and proves: firstly, that this vector is a permutation of $\varDelta_E$, using π_{PERM}, further that $\bar{c}_i = c_i / c_{i-1} \in \{E_{pk}(0), E_{pk}(1)\}$ by proving that either $(pk, \bar{c}_i)$ or $(pk, \bar{c}_i / E_{pk}(1))$ is a Diffie-Hellman tuple, and, finally, that $\prod_i \bar{c}_i = E_{pk}(|Q|)$. P_1 also decrypts $c_{3 \cdot |Q|}$, which is always an encryption of $|Q|$.

3. **First iteration:**

   (a) P_2 chooses the encryption of the next state $c_{1,t_1} = E_{pk}(\varDelta(1, t_1))$. It then defines $c_1 = c_{1,t_1} \cdot_G E_{pk}(0)$, i.e. a random encryption of the next state and sends it to P_1.

   (b) P_2 proves that $D_{sk}(c_1) \in \{D_{sk}(c_{1,0}), D_{sk}(c_{1,1})\}$ using the zero-knowledge proof π_{ENC} for $m = 1$.

4. **Iterations $\{2, \ldots, \ell\}$:** for every $\xi \in \{2, \ldots, \ell\}$, let $c_{\xi-1}$ denotes the encryption of $\varDelta(1, (t_1, \ldots, t_{\xi-1}))$; the parties continue as follows:

   (a) **Subtracting the current state from the state labels in $\varDelta$:** The parties compute the vector of encryptions $C = \{c_{\xi-1} / c_{j,\epsilon}\}$ for all j. Note that only one ciphertext will denote an encryption of 0, and that indicates the position corresponding to the current state.

   (b) P_2 **permutes C and column t_ξ:**

   - P_2 computes $C' = \{c_{j,t_\xi} \cdot_G E_{pk}(0)\}$ for all j (note that C' corresponds to column t_ξ in the transition matrix – i.e. the encryptions of all the possible next states given input bit t_ξ) and sends C' to P_1. It also proves that C' were computed correctly using π_{ENC}.
   - P_2 chooses a random permutation π over $\{1, \ldots, |Q|\}$ and sends P_1 a randomized version of $\{\pi(C), \pi(C')\}$ That is, the ciphertexts are permuted and randomized by multiplication with $E_{pk}(0)$. The parties engage in zero-knowledge proof π_{PERM} where P_2 proves that it computed this step correctly.

(c) P_1 **permutes** $\pi(C)$ **and column** t_ξ**:** Let $C^2_\pi, {C'}^2_\pi$ denote the permuted columns that P_2 sent. If P_1 is accepts the proof π_{PERM} it continues similarly by permuting and randomizing $C^2_\pi, {C'}^2_\pi$ using a new random permutation π'. P_1 proves its computations using π_{PERM}.

(d) **Multiplicative masking:** Let $C^1_\pi, {C'}^1_\pi$ denote the permuted columns from the previous step. C^1_π corresponds to the permuted ϵ column from the transition matrix, which contains the labels of the states minus the label of the current state. The parties take turns in masking C^1_π as follows: for every $\langle c_{j,a}, c_{j,b}\rangle \in C^1_\pi$, P_2 chooses $x \leftarrow_R \mathbb{Z}_q$ and computes $c'_j = \langle c^x_{j,a}, c^x_{j,b}\rangle$. It then proves that $(\mathbb{G}, c_{j,a}, c^x_{j,a}, c_{j,b}, c^x_{j,b})$ is a Diffie-Hellman tuple using π_{DDH}. (Encryptions of zero will be unaffected by this step, while non-zero values will be mapped to random values.) P_1 repeats this step and masks the result. Let $\widetilde{C}_1, \widetilde{C}_2$ be the resulting masked columns for P_1 and P_2 respectively.

(e) **Additive masking:** $\widetilde{C}_1$ corresponds to the permuted ϵ column from the transition matrix, which has by now been masked by both parties. P_2 chooses $|Q|$ random values $\mu^{P_2}_1, \ldots, \mu^{P_2}_{|Q|}$ and encrypts them; $\gamma^{P_2}_i = E_{pk}(\mu^{P_2}_i)$. P_2 also computes $\bar{c}^{P_2}_i = (\tilde{c}_i \cdot_G \gamma^{P_2}_i) \cdot_G E_{pk}(0)$ for every $\tilde{c}_i \in \widetilde{C}_2$, and proves in zero-knowledge that the masking is correct by proving that for every i

$$\left\langle \frac{\bar{c}_{i,a}}{\tilde{c}_{i,a} \cdot \gamma_{i,a}}, \frac{\bar{c}_{i,b}}{\tilde{c}_{i,b} \cdot \gamma_{i,b}} \right\rangle$$

is an encryption of zero, using π_{DDH}. The ciphertext $\tilde{c}_i$ that denotes an encryption of zero is now mapped to the ciphertext $\bar{c}_i$ that contains an encryption of μ_i. The others are mapped to random values. Let $\overline{C}_2 = [\bar{c}_1, \ldots, \bar{c}_{|Q|}]$ denote this masked vector. P_1 performs this step as well to obtain $\overline{C}_1$.

(f) **Decrypting column** ϵ**:** The parties decrypt it using their shared knowledge of sk. In particular, for every $\bar{c}_j \in \overline{C}$ in which $\bar{c}_j = \langle \bar{c}_{j,a}, \bar{c}_{j,b}\rangle$, P_1 computes $c'_j = \bar{c}^{r_1}_{j,a}$ and proves that $(\mathbb{G}, g, g^{r_1}, c'_j, \bar{c}_{j,a})$ is a Diffie-Hellman tuple. Next P_2 computes $c''_j = \bar{c}^{r_2}_{j,a}$ and proves that $(\mathbb{G}, g, g^{r_2}, c''_j, \bar{c}_{j,a})$ is a Diffie-Hellman tuple. The parties decrypt $\bar{c}_j$ by computing $D_{sk}(\bar{c}_j) = \bar{c}_{j,b}/(c'_j \cdot c''_j)$. Each party P_i sends its additive shares; $\mu^{P_i}_1, \ldots, \mu^{P_i}_{|Q|}$, and proves their correctness via π_{DDH}. The parties chooses the index j for which there exists $D_{sk}(\bar{c}_j) = \mu^{P_1}_j + \mu^{P_2}_j$ (with high probability there will be only one such index).

5. **Checking the output:** After the ℓth iteration c_ℓ contains the encryption of $\Delta(1, t)$. To check if this is an accepting state without revealing any other information (in particular which state it is) the parties do the following:

(a) They compute the ciphertext vector $C_F = \{c_\ell / c\}_{c \in \mathsf{E}(F)}$. Notice that $\Delta(1, t)$ is accepting if and only if one of these ciphertexts is an encryption of 0.

(b) P_2 masks the ciphertexts as in Steps 4d and 4e. Let C'_F be the resulting vector.

(c) P_2 randomizes and permutes C'_F. It also proves correctness using π_{PERM}. Let C''_F be the resulting vector.

(d) The parties decrypt all the ciphertexts in C''_F with the result going only to P_1; for every ciphertext $c = \langle c_a, c_b \rangle \in C''_F$, the party P_2 sends $c'_a = c_a^{r_2}$ and proves that $(\mathbb{G}, g, g^{r_2}, c_a, c'_a)$ is a Diffie-Hellman tuple. This information allows P_1 to decrypt the ciphertexts, and P_1 accepts if one of the decryptions equals one of the additive mask of P_2.

Before turning to the security proof we show that if both parties are honest then P_1 outputs $\Gamma(t)$ with probability negligibly close to 1. This is because with each iteration ξ the parties agree upon the correct encrypted state c_ξ with probability close to 1. We continue with the following claim,

Theorem 3. *Assume that π_{DL}, π_{DDH}, π_{ENC} and π_{PERM} are as described above and that (G, E, D) is the semantically secure El Gamal encryption scheme. Then π_{AUTO} securely computes $\mathcal{F}_{\text{AUTO}}$ in the presence of malicious adversaries.*

Intuitively it should be clear that the automaton and the text remain secret, if the encryption scheme is secure. However, a formal proof of Theorem 3 is actually quite involved. Consider, for example, the case in which P_1 is corrupted and we need to simulate P_2. The simulator is going to choose a random input and run P_2's code on it. Then, to prove that this view is indistinguishable, we need a reduction to the encryption scheme. A straightforward reduction does not work for the following reason: in order for the simulator to finish the execution correctly it must "know" the current state, at every iteration i; but when we do a reduction to El Gamal we need to plug in a ciphertext for the current state for which we do not know a decryption, and this prevents us from going forward to iteration $i + 1$. A non-trivial solution to this problem is to prove that the real and simulated views are indistinguishable via a sequence of hybrid games, in which indistinguishable changes are introduced to the way the simulator works, but still allowing it to finish the simulated execution.

As for the case that P_2 is corrupted, the proof is rather simple mainly because P_2 does not receive an output. Specifically, the simulator extracts P_2's input in every iteration using the extractor for π_{ENC}.

Efficiency. We present an analysis of our protocol and compare its efficiency to the generic protocols of [14,16] for secure two-party computation in the presence of malicious adversaries. We introduced the efficency of the generic protocols in Sec. 1. A circuit that computes $\mathcal{F}_{\text{AUTO}}$ would require $O(\ell Q \log Q)$ gates. While there exists a sequential circuit of size $O(\ell Q)$ that computes a Q-state automaton over a binary input of size ℓ, the generic protocol applies only to combinatorial circuits. So, our protocol improves the computational complexity over the generic solution by a factor of $n \log Q$ or $\log Q$ operations compared to [14] and [16] respectively.[3] In comparison to [14,16], we have the following:

[3] Although not presented here, our protocol generalizes to strings over a larger alphabet, and this provides another $\log \Sigma$ factor improvement where Σ is the size of the alphabet.

1. *Rounds of Communication:* Our protocol runs $O(\ell)$ rounds where ℓ is the length of the text, compared to the constant round complexity of [14,16]. This is due to the fact that the parties cannot initiate a new iteration before completing the previous one. By applying known techniques, the round complexity can be reduced to $O(m)$ (i.e., the length of the pattern) by breaking the text into blocks of size $2m$; see Sect. 4 for more details. We note that the length of the pattern is typically very small, usually up to a constant. An additional improvement can be achieved if some leakage is allowed. In particular, since the number of rounds are determined by the length of the pattern, the pattern can be broken into smaller blocks and the output combined out of these results. This means that additional information about the text is released, as it is then possible to identify appearances in the text that correspond to substrings of the input pattern.
2. *Asymmetric computations:* The overall number of exponentiations in protocol $\pi_{\text{VALIDAUTO}}$, including the zero-knowledge proofs is $O(m \cdot \ell)$. As for [14], the number of such computations depends on the number of oblivious transfer executions, which is bounded by $\max(4\ell, 8s)$ where and the number of commitments $2\ell s(s+1)$, where s must be large enough so that $2 * 2^{\frac{-s}{17}}$ is sufficiently small. Finally, [16] requires $O(|C|)$ such operations. However, after carefully examining these costs, it seems that the parties compute approximately 720 RSA exponentiations per gate, where the number of gates is $O(m \cdot \ell)$, which is clearly not practical. Furthermore, the protocol of [16] also requires a security parameter, usually of size of at least 1024, due to the strong RSA assumption. Consequently, all the public-key operations are performed over groups modulus this value (or higher, such as N^2). In contrast, our protocol uses the El Gamal encryption scheme which can be implemented over elliptic curve group, typically, using only 160 bit keys.
3. *Symmetric operations:* [14], due to the use of a symmetric encryption scheme, also requires $O(s|C| + s^2 \cdot n)$ such symmetric computations.
4. *Bandwidth:* In our protocol, the parties send each other $O(m \cdot \ell)$ encryptions. The bandwidth of the protocol in [16] is similar (again, with relatively high constants), whereas in [14], the bandwidth is dominated by the $O(s|C| + s^2 n)$ symmetric-key encryptions. In [15], it is shown that communication is the main bottleneck when implementing the malicious protocol of [14].

In order to conclude the comparison, we note that implementing a circuit that solves the basic pattern matching problem may be significantly harder than implementing our protocol.

3.4 A Zero-Knowledge Proof of Knowledge for a KMP Automaton

Space does not permit a full treatment of our protocol for $\pi_{\text{VALIDAUTO}}$, but we we include a description of the protocol here:

Protocol 3. $\pi_{\text{VALIDAUTO}}$

- **Auxiliary input for the prover:** A collection $\{Q_{i,j}, r_{i,j}\}_{i,j}$ of Q sets, each set is of size 2, such that $c_{i,j} = E_{pk}(Q_{i,j}; r_{i,j})$ for all $i \in \{0,1\}$ and $j \in \{1, \ldots, |Q|\}$.
- **Auxiliary inputs for both:** A prime p such that $p - 1 = 2q$ for a prime q, the description of a group $\mathbb{G}$ of order q for which the DDH assumption holds, and a generator g of $\mathbb{G}$.
- **The protocol:**
  1. For every $c_{i,j} = \langle \alpha_{i,j}, \beta_{i,j} \rangle$, the prover P proves the knowledge of $\log_g \alpha_{i,j}$ using π_{DL}.
  2. **For every row $\Delta_j = \{c_{j,0}, c_{j,1}\}_{j=1}^{|Q|}$ in the transition matrix P proves the following:**
     (a) It first randomly permutes $c_{j,0}$ and $c_{j,1}$ and employs π_{PERM} to prove its computations.
     (b) It proves that there exists $b \in \{0,1\}$ in which $c_{j,b} = E_{pk}(j+1)$ by proving that $(pk, c_{j,b}/E_{pk}(j+1))$ is a Diffie-Hellman tuple.
     (c) P **proves the correctness of** $c_{j,1-b}$ **in two steps:** (i) It first proves that it corresponds to a valid prefix of $p_{\langle j-1 \rangle}$, (ii) then it proves the maximality of this prefix ($p_{\langle r \rangle}$ denotes the rth length prefix $p_1, \ldots, p_r$ of p).
        i. Let $|Q| - 1 = m$, then the verifier V chooses m random elements $u_\alpha \leftarrow_R \mathbb{Z}_q$ and sends $\{u_\alpha\}_\alpha$ to P. Next, both parties use the homomorphic properties of the encryption scheme to compute an encryption $v_{\alpha',\alpha''} = E_{pk}(\sum_{k=\alpha'}^{\alpha''} u_k \cdot p_k)$ for all $\alpha', \alpha'' \in \{1, \ldots, m\}$ with $\alpha' < \alpha''$.
        ii. **Proving the existence of a prefix that matches a suffix of** $p_{\langle j-1 \rangle}$**:** For all $1 \leq k \leq j-1$ the parties compute an encryption $v'_k = (v_{j-k,j-1}/v_{1,k}) \cdot (c_{j,1-b}/k)$.
        P then proves that there exists k for which v'_k is a Diffie-Hellman tuple.
        iii. **Proving that** $p_{\langle D_{sk}(c_{i,b}) \rangle}$ **corresponds to the longest suffix of** $p_{\langle j-1 \rangle}$**:**
        Next P proves that there does not exist an index $D_{sk}(c_{i,1-b}) < k \leq j-1$ in which $v_{j-k,j-1}/v_{1,k} = 0$ yet $c_{j,1-b}/k \neq 0$, as this would imply that $p_{\langle k \rangle}$ is a larger suffix of $p_{\langle j-1 \rangle}$ that matches however, $D_{sk}(c_{j,1-b}) \neq k$.
        - Therefore, for every $1 \leq k \leq j-2$ and for every $2 \leq k' \leq j-1$ the parties compute an encryption $e_{k,k'} = v'_k \cdot (v_{j-k',j-1}/v_{1,k'})$ for which P then proves that $e_{k,k'}$ is not an encryption of zero using π_{NZ}.
  3. If all the proofs are successfully completed, V outputs 1. Otherwise it outputs 0.

We refer the reader to the extended version of this paper for a detailed definition and proof.

4 Text Search Protocol with Simulation Based Security

In this section we present our complete and main construction for securely evaluating the pattern matching functionality $\mathcal{F}_{\text{PM}}$ defined by

$$(p, (T, m)) \mapsto \begin{cases} (\{i \mid T_i = p\}, \lambda) & \text{if } |p| = m \\ (\{i \mid T_i = p_1 \dots p_m\}, \lambda) & \text{otherwise} \end{cases}$$

Recall that our construction is presented in the malicious setting with full simulatability and is modular in the sub-protocols π_{AUTO} and $\pi_{\text{VALIDAUTO}}$. Having described the sub-protocols incorporated in the our scheme we are now ready to describe it formally. Our protocol is comprised out of two main phases: first the parties first engage in an execution of $\pi_{\text{VALIDAUTO}}$ for which P_1 proves that it indeed sent a valid KMP automaton, followed by an execution of π_{AUTO} which in an evaluation of Γ on P_2's private input.

In order to reduce the round complexity of the protocol, long texts are partitioned into $2m$ pieces and are handled separately so that the KMP algorithm is employed on each block independently (thus all these executions can be run in parallel). That is, let $T = t_1, \dots, t_\ell$ then the text is partitioned into the blocks $(t_1, \dots, t_{2m})$,$(t_m, \dots, t_{3m})$,$(t_{2m}, \dots, t_{4m})$ and so on, such that every two consecutive blocks overlap in m bits. This ensures that all the matches will be found. Therefore, the total number of blocks is ℓ/m.

Protocol 4. $\boldsymbol{\pi_{\text{PM}}}$

- **Inputs:** The input of P_1 is a binary pattern $p = p_1, \dots, p_m$, and the input of P_2 is a binary string $T = t_1, \dots, t_\ell$.
- **Auxiliary Inputs:** the security parameter 1^n, and the input sizes ℓ and m.

- **The protocol:**
  1. P_1 constructs its automaton $\Gamma = (Q, \Sigma, \Delta, q_1, F)$ according to the KMP specifications based on its input p and sends P_2 encryptions of the transition matrix Δ and the accepting states F, denoted by E_Δ and E_F.
  2. The parties engage in an execution of the zero-knowledge proof $\pi_{\text{VALIDAUTO}}$ where P_1 proves that Γ was constructed correctly. That is, P_1 proves that the set E_Δ corresponds to a valid KMP automaton and a well defined input pattern of length m. If P_2's output from this execution is 1 the parties continue to the next step. Otherwise P_2 aborts.
  3. P_2 sends an encryption of T to P_1 and the parties partition T into ℓ/m blocks of length $2m$ in which every two consecutive blocks overlap in m bits.
  4. The parties engage in ℓ/m parallel executions of π_{AUTO} on these blocks.[4] For every $1 \leq i \leq \ell/m$, let $\{\mathsf{output}^{\mathsf{i}}_{\mathsf{j}}\}_{j=1}^{m+1}$ denotes the set of P_1's outputs from the ith execution. Then P_1 returns $\{j \mid \mathsf{output}^{\mathsf{i}}_{\mathsf{j}} = \text{``}\mathsf{accept}\text{''}\}$.

[4] The parties run a slightly modified version of π_{AUTO} where they carry out Step 5 for verifying acceptance $m+1$ times, as each block contains potentially $m+1$ matches. This step can be executed in parallel for all block locations.

Theorem 4. *Assume that π_{AUTO} and $\pi_{\text{VALIDAUTO}}$ are as described above and that (G, E, D) is the semantically secure El Gamal encryption scheme. Then π_{PM} securely computes $\mathcal{F}_{\text{PM}}$ in the presence of malicious adversaries.*

The security proof for π_{PM} is a combination of the proofs described for for π_{AUTO} and $\pi_{\text{VALIDAUTO}}$ and is therefore omitted here.

Efficiency. Since the costs are dominated by the costs of π_{AUTO}, we refer the reader to the detailed analysis presented in Sect. 3.3. The overall costs are amount to $O(m \cdot \ell + m^2) = O(m \cdot \ell)$ since in most cases $m << \ell$.

5 Conclusion

Our protocols for oblivious automata evaluation and pattern matching operate in the standard model and require no CRS nor apply a cut-and-choose strategy. Having P_1, P_2 hold strings of lengths ℓ, m for the text and the pattern respectively, both protocols incur communication and computation costs of $O(\ell \cdot m)$ which is even asymptotically better then the general construction for the oblivious automata evaluation that requires a circuit with $O(\ell \cdot m \log m)$ gates. Table 2 summarizes and compares the computational and communications costs of each of these schemes.

Table 2. Summary of results

	Round Complexity	Communication Complexity	Asymmetric Computations	Symmetric Computations
Lindell-Pinkas [14]	constant	$O(s\|C\| + s^2 m)$ times 128/256 bits	$\max(4m, 8s)$ OT's	$O(s\|C\| + s^2 \cdot m)$
Jarecki-Shmatikov [16]	constant	$O(\ell \cdot m)$ times 2048 bits	720 exp. per gate	None
Our Protocol	$O(m)$	$O(\ell \cdot m)$ times 160 bits	$O(\ell \cdot m)$	None

References

1. Canetti, R.: Security and composition of multi-party cryptographic protocols. Journal of Cryptology 13, 2000 (1998)
2. Goldwasser, S., Levin, L.A.: Fair computation of general functions in presence of immoral majority. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 77–93. Springer, Heidelberg (1991)
3. Beaver, D.: Foundations of secure interactive computing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 377–391. Springer, Heidelberg (1992)
4. Micali, S., Rogaway, P.: Secure computation (abstract). In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 392–404. Springer, Heidelberg (1992)
5. Yao, A.C.C.: How to generate and exchange secrets. In: SFCS 1986: Proceedings of the 27th Annual Symposium on Foundations of Computer Science, Washington, DC, USA, pp. 162–167. IEEE Computer Society, Los Alamitos (1986)

6. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: STOC 1987: Proceedings of the nineteenth annual ACM symposium on Theory of computing, pp. 218–229. ACM, New York (1987)
7. Goldreich, O.: Foundations of Cryptography: Basic Applications, vol. 2. Cambridge University Press, New York (2004)
8. Knuth Jr., D.E., Morris, J.H., Pratt, V.R.: Fast pattern matching in strings. SIAM J. Comput. 6(2), 323–350 (1977)
9. 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)
10. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory IT-22(6), 644–654 (1976)
11. El Gamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985)
12. Jarecki, S., Xiaomin, L.: Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 577–594. Springer, Heidelberg (2009)
13. Troncoso-Pastoriza, J.R., Katzenbeisser, S., Celik, M.: Privacy preserving error resilient dna searching through oblivious automata. In: CCS 2007: Proceedings of the 14th ACM conference on Computer and communications security, pp. 519–528. ACM, New York (2007)
14. Lindell, Y., Pinkas, B.: An efficient protocol for secure two-party computation in the presence of malicious adversaries. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 52–78. Springer, Heidelberg (2007)
15. Pinkas, B., Schneider, T., Smart, N.P., Williams, S.C.: Secure two-party computation is practical. In: ASIACRYPT 2009, Tokyo, Japan. LNCS, pp. 250–267. Springer, Heidelberg (2009)
16. 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)
17. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
18. 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)
19. Hazay, C., Nissim, K.: Efficient set operations in the presence of malicious adversaries (2010)
20. Groth, J., Ishai, Y.: Sub-linear zero-knowledge argument for correctness of a shuffle. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 379–396. Springer, Heidelberg (2008)
21. Groth, J., Lu, S.: Verifiable shuffle of large size ciphertexts. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 377–392. Springer, Heidelberg (2007)

Solving a 676-Bit Discrete Logarithm Problem in GF(3^{6n})

Takuya Hayashi[1,⋆], Naoyuki Shinohara[2], Lihua Wang[2], Shin'ichiro Matsuo[2], Masaaki Shirase[3], and Tsuyoshi Takagi[1,⋆]

[1] Graduate School of Mathematics, Kyushu University, 744, Motooka, Nishi-ku, Fukuoka, 819-0395, Japan
[2] Information Security Research Center, National Institute of Information and Communications Technology, 4-2-1, Nukui-Kitamachi, Koganei, Tokyo 184-9795, Japan
[3] School of Systems Information Science, Future University Hakodate, 116-2, Kamedanakano-cho, Hakodate, Hokkaido, 041-0806, Japan

Abstract. Pairings on elliptic curves over finite fields are crucial for constructing various cryptographic schemes. The η_T pairing on supersingular curves over GF(3^n) is particularly popular since it is efficiently implementable. Taking into account the Menezes-Okamoto-Vanstone (MOV) attack, the discrete logarithm problem (DLP) in GF(3^{6n}) becomes a concern for the security of cryptosystems using η_T pairings in this case. In 2006, Joux and Lercier proposed a new variant of the function field sieve in the medium prime case, named JL06-FFS. We have, however, not yet found any practical implementations on JL06-FFS over GF(3^{6n}). Therefore, we first fulfill such an implementation and we successfully set a new record for solving the DLP in GF(3^{6n}), the DLP in GF($3^{6\cdot 71}$) of 676-bit size. In addition, we also compare JL06-FFS and an earlier version, named JL02-FFS, with practical experiments. Our results confirm that the former is several times faster than the latter under certain conditions.

Keywords: function field sieve, discrete logarithm problem, pairing-based cryptosystems.

1 Introduction

Based on pairings, many novel cryptographic protocols have been successively constructed, such as identity-based encryptions [8], forward-secure cryptosystems, proxy cryptosystems, keyword searchable PKEs [7]. As a result, two requirements arose: efficient pairing computation and security parameter selection.

The η_T pairing [5] on supersingular curves over GF(3^n) has been efficiently implemented both in software and hardware [6,13,14][1]. Along with the increase in computation speed on the η_T pairing, one may ask whether cryptosystems

⋆ This work was done when authors belonged to Future University Hakodate.
[1] Here, n is a prime number such as $n = 97$, 163 and 193 [25].

based on the η_T pairing are still secure. It is well known that a discrete logarithm problem (DLP) on supersingular curves over GF(q) can be converted to a DLP in GF(q^m) (where q is a prime power and m is not larger than 6) [24]. Therefore, the DLP in GF(3^{6n}) is one of the most important problems in analyzing the cryptosystems constructed with the η_T pairing on supersingular curves over GF(3^n).

The function field sieve (FFS) is the most efficient algorithm for solving the DLP in finite fields of small characteristic. The complexity of the FFS for solving the DLP in GF(3^{6n}) is $L_{3^{6n}}[1/3, c]$ with constant c, where

$$L_{3^{6n}}[1/3, c] = \exp((c + o(1))(\log 3^{6n})^{1/3}(\log\log 3^{6n})^{2/3}).$$

Here $o(1)$ stands for a function that converges to zero as n approaches infinity.

The first FFS was proposed by Adleman [1] in 1994. Five years later, Adleman and Huang proposed an improved FFS (AH-FFS) with $c = (32/9)^{1/3}$ [2]. In 2002, Joux and Lercier proposed a practical improvement of the FFS (JL02-FFS) [16]. Since a definition polynomial of the function field in JL02-FFS can select more flexibly, JL02-FFS is more practical than AH-FFS, though its asymptotic complexity is the same as that of AH-FFS. Furthermore, by using JL02-FFS, Joux and Lercier succeeded in solving the DLP in GF(2^{613}). This refreshed the record for solving the DLP in finite fields of characteristic two with regard to bit size [15]. In 2006, Joux and Lercier proposed another new variant of the FFS (JL06-FFS) [18]. JL06-FFS has the same asymptotic complexity with JL02-FFS for solving the DLP in GF(3^{6n}), where n is a prime number[2]. This work implied that JL06-FFS might be efficient for solving the DLP in extension fields of GF(3^6) of degree n. However, to our knowledge, there have been no practical experiments. Note that JL02-FFS can also be applied to extension fields of GF(3^6) of degree n, but [12] showed no advantage using GF(3^6) as the base field.

Our contributions. We have first conducted experiments on JL06-FFS in GF(3^{6n}). In JL06-FFS, GF(3^{6n}) is constructed as extension fields of GF(3^6) of degree n, and thus the Galois action can be dealt for reducing required relations. By our implementation, we succeeded in solving the DLP in GF$(3^{6\cdot 71})$ of 676-bit size with about 33 days computation, which is the new record for solving the DLP in GF(3^{6n}). Our work contributes to the selecting of security parameters. Additionally, we compared JL06-FFS [18] with JL02-FFS [16], and according to the experimental results, we confirmed that JL06-FFS is several times faster than JL02-FFS with $n = 19, 61$.

The rest of the paper is organized as follows. In Section 2, we briefly review the FFS algorithm. In Section 3, we compare JL02-FFS with JL06-FFS according to the polynomial selection method and experimental results. In Section 4, we describe our implementation on how to solve the DLP in GF$(3^{6\cdot 71})$ in detail, which is based on JL06-FFS. Concluding remarks are made in Section 5.

[2] When n is a composite number, this variant may have complexity $L_{3^{6n}}[1/3, 3^{1/3}]$ for solving the DLP in GF(3^{6n}) (When JL06-FFS has complexity $L_{q^m}[1/3, 3^{1/3}]$, we call it JL06-FFS-2). We do not deal with this case in this paper.

2 Outline of Function Field Sieve

In this section, we describe an overview of the FFS [1], which consists of four steps: polynomial selection, collection of relations, linear algebra, and individual logarithm. We particularly deal with the FFS for solving the DLP in extension fields of GF(3^6) of degree n and describe the four steps below. For more details, refer to related work as [1,12,16,18].

Throughout this paper, let γ be a generator of the multiplicative group of GF(3^{6n}) and $\alpha \in \langle\gamma\rangle$, then we try to find the smallest positive integer $\log_\gamma \alpha$ such that $\gamma^{\log_\gamma \alpha} = \alpha$, which is called the discrete logarithm.

1. Polynomial selection: Select $f \in \mathrm{GF}(3^6)[x]$ such that f is a monic irreducible polynomial of degree n, then $\mathrm{GF}(3^{6n}) \cong \mathrm{GF}(3^6)[x]/(f)$. Next, find a polynomial $H(x, y) \in \mathrm{GF}(3^6)[x, y]$ satisfying the eight conditions proposed by Adleman [1]. Then there is a surjective homomorphism

$$\varPhi : \begin{cases} \mathrm{GF}(3^6)[x, y]/(H) \rightarrow \mathrm{GF}(3^{6n}) \cong \mathrm{GF}(3^6)[x]/(f) \\ \qquad\quad y \qquad\qquad \mapsto \qquad\qquad\quad m, \end{cases}$$

   where m is in $\mathrm{GF}(3^6)[x]$ such that $H(x, m) \equiv 0 \pmod f$. Here we select the smoothness bound B and define a rational factorbase B_R and an algebraic factorbase B_A as follows:

$$B_R = \{\mathfrak{p} \in \mathrm{GF}(3^6)[x] \mid \deg(\mathfrak{p}) \le B,\ \mathfrak{p} \text{ is irreducible}\},$$
$$B_A = \{\langle \mathfrak{p}, y - t\rangle \in \mathrm{Div}(\mathrm{GF}(3^6)[x,y]/(H)) \mid \mathfrak{p} \in B_R,\ t \equiv m \pmod{\mathfrak{p}}\},$$

   where $\mathrm{Div}(\mathrm{GF}(3^6)[x, y]/(H))$ is the divisor group of $\mathrm{GF}(3^6)[x, y]/(H)$ and $\langle \mathfrak{p}, y - t\rangle$ is a divisor generated by $\mathfrak{p}$ and $y - t$.
2. Collection of relations: For $r, s \in \mathrm{GF}(3^6)[x]$ of degree not larger than B, find at least $(\#B_R + \#B_A)$ relatively prime pairs (r, s) such that

$$rm + s = \prod_{\mathfrak{p}_i \in B_R} \mathfrak{p}_i^{a_i}$$
$$\langle ry + s\rangle = \sum_{\langle \mathfrak{p}_j, t_j\rangle \in B_A} b_j \langle \mathfrak{p}_j, y - t_j\rangle. \tag{1}$$

   Such a pair (r, s) is called a double smooth pair. For each (r, s), compute

$$rm + s, \tag{2}$$

$$(-r)^d H(x, -s/r). \tag{3}$$

   Polynomial (3) is said to be B-smooth if it is factorized into irreducible polynomials of degree not larger than B, and then we have

$$(-r)^d H(x, -s/r) = \prod_{\langle \mathfrak{p}_j, t_j\rangle \in B_A} \mathfrak{p}_j^{b_j}, \tag{4}$$

where t_j is uniquely determined by r, s and $\mathfrak{p}_j$. Then the b_j in Equation (4) is exactly the same as the one in Equation (1). When both Polynomials (2) and (3) are B-smooth, a pair (r, s) is a double smooth pair. Eventually, we obtain the following relation:

$$\sum_{\mathfrak{p}_i \in B_R} a_i \log_\gamma \mathfrak{p}_i \equiv \sum_{\langle \mathfrak{p}_j, t_j \rangle \in B_A} b_j \log_\gamma \kappa_j \pmod{(3^{6n}-1)/(3^6-1)}, \tag{5}$$

where

$$\kappa_j = \Phi(\lambda_j)^{1/h}, \ \langle \lambda_j \rangle = h \langle \mathfrak{p}_j \ y - t_j \rangle, \tag{6}$$

for the class number h of the quotient field of $\mathrm{GF}(3^6)(x)[y]/(H)$.

3. Linear algebra: For the number R of relations, construct an $R \times (\#B_R + \#B_A)$ matrix M from the relations in Equation (5) and $(\#B_R + \#B_A)$ dimensional column vector $\boldsymbol{v}$ as follows:

$$M = \begin{pmatrix} a_1^{(1)} & \dots & a_{\#B_R}^{(1)} & -b_1^{(1)} & \dots & -b_{\#B_A}^{(1)} \\ \vdots & & \vdots & \vdots & & \vdots \\ a_1^{(R)} & \dots & a_{\#B_R}^{(R)} & -b_1^{(R)} & \dots & -b_{\#B_A}^{(R)} \end{pmatrix}, \ \boldsymbol{v} = \begin{pmatrix} \log_\gamma \mathfrak{p}_1 \\ \vdots \\ \log_\gamma \mathfrak{p}_{\#B_R} \\ \log_\gamma \kappa_1 \\ \vdots \\ \log_\gamma \kappa_{\#B_A} \end{pmatrix}.$$

Then we solve the linear equation

$$M\boldsymbol{v} \equiv 0 \pmod{(3^{6n}-1)/(3^6-1)}. \tag{7}$$

4. Individual logarithm: Find integers e_i, f_j such that

$$\log_\gamma \alpha \equiv \sum_{\mathfrak{p}_i \in B_R} e_i \log_\gamma \mathfrak{p}_i + \sum_{\langle \mathfrak{p}_j, t_j \rangle \in B_A} f_j \log_\gamma \kappa_j \pmod{(3^{6n}-1)/(3^6-1)},$$

then compute the discrete logarithm $\log_\gamma \alpha$. This is done using the special-$\mathfrak{q}$ descent method [16,18,19].

3 Comparison of Polynomial Selection on JL02-FFS and JL06-FFS

The two most efficient variants of the FFS for solving the DLP in $\mathrm{GF}(3^{6n})$ are JL02-FFS and JL06-FFS. Although they have the same asymptotic complexity, there is a considerable difference between them in the fixed extension degree for practical use. The time complexities of JL02-FFS and JL06-FFS depend on the size of each sieving area, which is the number of pairs (r, s), and each size is explained in the following subsections. Note that our comparison is done merely by the size of the sieving area, and the detailed analysis should incorporate the non-integer smoothness bound estimated by Granger [11].

3.1 Polynomial Selection of JL02-FFS and Its Sieving Area

At first we describe an outline of the polynomial selection of JL02-FFS, after that we estimate the size of the sieving area. In order to distinguish from previous section, we set the subindex "02" after the symbols.

Let $H_{02}(x, y)$ of degree d_{02} in y be formed as C_{ab} curves [23]:

$$H_{02}(x, y) = h_{a,0}y^a + h_{0,b}x^b + \sum_{ib+ja<ab} h_{i,j}y^i x^j \quad (h_{i,j} \in \mathrm{GF}(3),\, h_{a,0},\, h_{0,b} \neq 0).$$

Randomly choose polynomials $u_1, u_2 \in \mathrm{GF}(3)[x]$ of degree at most $\lfloor 6n/d_{02} \rfloor$, and try to find an irreducible polynomial $f_{02} = u_2^{d_{02}} H_{02}(x, -u_1/u_2) \in \mathrm{GF}(3)[x]$ of degree $6n$ such that $\gcd(u_2, f_{02}) = 1$, then u_2 is invertible modulo f_{02}. Then, there is a surjective homomorphism

$$\Phi_{02} : \begin{cases} \mathrm{GF}(3)[x, y]/(H_{02}) \rightarrow \mathrm{GF}(3^{6n}) \cong \mathrm{GF}(3)[x]/(f_{02}) \\ \qquad\qquad y \qquad\qquad \mapsto \qquad\qquad -u_1/u_2, \end{cases}$$

where $H_{02}(x, y)$ holds $H_{02}(x, -u_1/u_2) \equiv 0 \pmod{f_{02}}$. In this polynomial selection, we need to modify Polynomial (2) to $su_2 - ru_1$. Note that r and s are chosen in GF(3)[x] of degree not larger than B_{02} in JL02-FFS, the size of the sieving area in the collection of relation step is

$$3^{B_{02}+1} \cdot 3^{B_{02}+1}. \tag{8}$$

From heuristic analysis in [16], JL02-FFS becomes optimized when we choose the smoothness bound B_{02} as

$$B_{02} = \lceil (4/9)^{1/3}(6n)^{1/3} \log_3(6n)^{2/3} \rfloor. \tag{9}$$

and the extension degree d_{02} of $H_{02}(x, y)$ as $d_{02} = \lceil \sqrt{6n/(B_{02}+1)} \rfloor$. For example, for $n = 97, 163, 193$, we have $(n, B_{02}) = (97, 21), (163, 26), (193, 28)$.

3.2 Polynomial Selection of JL06-FFS and Its Sieving Area

Next we describe an outline of the polynomial selection of JL06-FFS and estimate the size of the sieving area of JL06-FFS.

For each extension degree n of GF(3^6), we choose the smallest smoothness bound B_{06} in JL06-FFS satisfying the following condition,

$$(B_{06}+1)\log(3^6) \geq \sqrt{n/B_{06}}\log(n/B_{06}) \tag{10}$$

For example, for $n = 97, 163, 193$, we have $(n, B_{06}) = (97, 3), (163, 4), (193, 4)$. Next, we choose positive integers d and d' such that $d \approx \sqrt{n/B_{06}}$ and $d' \approx \sqrt{nB_{06}}$, where $dd' \geq n$. After that, we randomly generate $g(y) \in \mathrm{GF}(3^6)[y]$ of degree d and set $H(x, y) = g(y) + x$. Finally, we try to find an irreducible polynomial f in GF(3^6)[x] of degree n, which divides $H(x, m)$, where $m \in \mathrm{GF}(3^6)[x]$ of degree d' is chosen randomly. In this polynomial selection, each of the leading coefficients of Polynomials (2) and (3) depends on r, so we avoid obtaining duplicate relations by fixing the leading coefficient of r as a monic polynomial. Therefore, the size of the sieving area in the collection of relations step is at most

$$(3^6)^{B_{06}+1} \cdot (3^6)^{B_{06}}. \tag{11}$$

3.3 Comparison of Sieving Area

We compare JL06-FFS with JL02-FFS with respect to the size of the sieving area in the collection of relations step in three classes of extension degree n: *experimental class* as $\{19, 31, 47, 61\}$, *medium-security class* as $\{97, 163, 193\}$, and *high-security class* as $\{239, 313, 353, 509\}$. Table 1 lists the smoothness bound and size of the sieving area in each variant. For each n, we obtain the smoothness bound B_{02} in Equation (9) and B_{06} in Equation (10), and estimate the size of the sieving area by Form (8) in JL02-FFS and by Form (11) in JL06-FFS.

Table 1. Parameters and sieving area

	n	Polynomial selection in JL02-FFS			Polynomial selection in JL06-FFS		
		$6n$	B_{02}	Size of sieving area	n	B_{06}	Size of sieving area
Experimental class	19	114	10	3.1×10^{10}	19	1	3.9×10^{8}
	31	186	12	2.5×10^{12}	31	2	2.1×10^{14}
	47	282	15	1.9×10^{15}	47	2	2.1×10^{14}
	61	366	17	1.5×10^{17}	61	2	2.1×10^{14}
Medium-security class	97	582	21	9.8×10^{20}	97	3	1.1×10^{20}
	163	978	26	5.8×10^{25}	163	4	5.8×10^{25}
	193	1158	28	4.7×10^{27}	193	4	5.8×10^{25}
High-security class	239	1434	30	3.8×10^{29}	239	4	5.8×10^{25}
	313	1878	34	2.5×10^{33}	313	5	3.1×10^{31}
	353	2118	36	2.0×10^{35}	353	5	3.1×10^{31}
	509	3054	42	1.1×10^{41}	509	6	1.6×10^{37}

Figure 1 shows the size of the required sieving area over GF(3^{6n}). The sieving area in JL06-FFS is much smaller than that in JL02-FFS when $n \neq 31, 163$. Moreover, the differences between the sieving areas in JL06-FFS and in JL02-FFS increase along with the increase in n. The computational cost in the collection of relations step is closely related to the size of the sieving area, so the collection of relations step in JL06-FFS might be several times faster than that in JL02-FFS.

We have conducted experiments on the collection of relations step in JL02-FFS and JL06-FFS to confirm the difference between their computational costs of that step. Parameters in JL02-FFS and JL06-FFS are listed in Table 2. The curves that we used in our experiments are superelliptic ones, but not C_{ab} curves

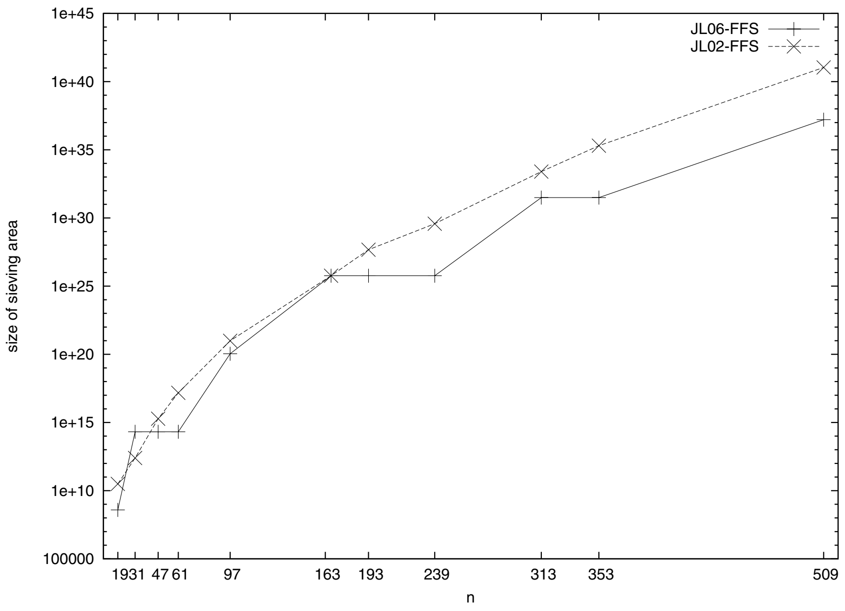


Fig. 1. Size of sieving area over GF(3^{6n}) in JL02-FFS and JL06-FFS

Table 2. Parameters in our experiments

n	Bit size of GF(3^{6n})	Experiments with JL02-FFS			Experiments with JL06-FFS		
		$6n$	B_{02}	$H_{02}(x, y)$	n	B_{06}	$H(x, y)$
19	181	114	10	$y^4 + x$	19	1	$y^5 + x$
31	295	186	12	$y^4 + x$	31	2	$y^4 + x$
47	447	282	15	$y^4 + x$	47	2	$y^5 + x$
61	581	366	17	$y^5 + x$	61	2	$y^6 + x$

as [12]. Note that we have only experimented with the experimental class as $n \in \{19, 31, 47, 61\}$, not with medium and high-security classes.

In our experiments, we used 96 cores, each of which had the same performance about Intel 2.83GHz Xeon. We implemented the lattice sieve [26] in JL02-FFS as [12,15,16]. On the other hand, we implemented the polynomial sieve [10] in JL06-FFS, since we fixed r as a monic poynomial in the collection of relations step and so the lattice sieve might not be efficient. The details of our implementation in JL06-FFS are described in Section 4.

Figure 2 shows the time complexity of JL02-FFS and JL06-FFS to compute the entire sieving area in the collection of relations step in GF(3^{6n}) with $n = 19, 31, 47, 61$, respectively. Note that we estimated the time when the computation lasts over one hour.

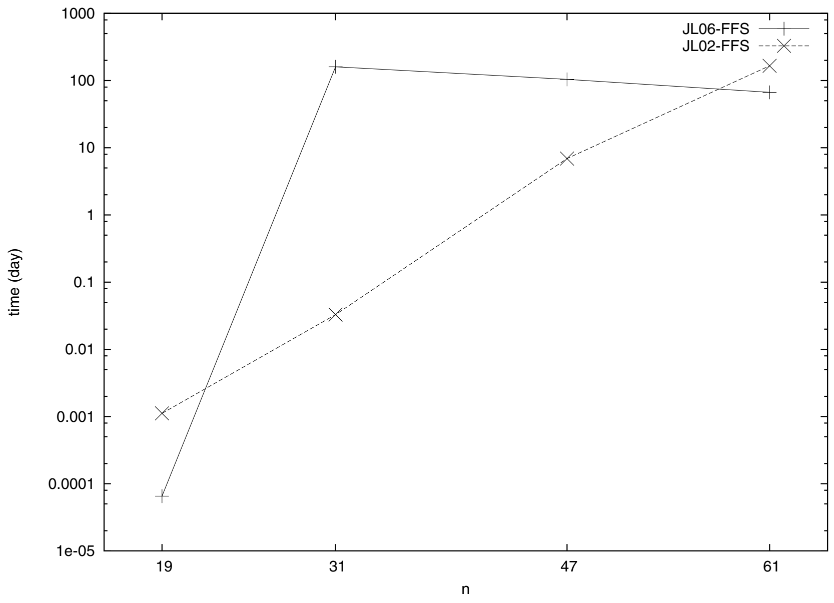


Fig. 2. Estimated time taken to compute entire sieving area in the collection of relations step over GF(3^{6n}) in JL02-FFS and JL06-FFS

When $n = 19, 61$, our implementation on JL06-FFS is faster than that on JL02-FFS, and we confirm that JL06-FFS is more efficient than JL02-FFS for solving the DLP in GF(3^{6n}). In particular, when $n = 61$, our implementation of JL06-FFS takes about 66 days for the collection of relations step, but our implementation of JL02-FFS takes about 165 days for the same step. Therefore, the former is 2.5 times faster than the latter. Accordingly, we expect that JL06-FFS will be efficient for solving the DLP in GF(3^{6n}) for larger n.

4 Solving the DLP in GF($3^{6\cdot 71}$)

In this section, we report that the DLP in GF($3^{6\cdot 71}$) of 676-bit size is solved by improving JL06-FFS. In our implementation, we deal with four practical improvements, polynomial sieve, free relation, Galois action, and parallel Lanczos method.

Particularly, by using the polynomial $H(x, y) = y^6 + x$, we only need to find about 1/8 of the originally required relations in the collection of relations step. Furthermore, via the Galois action, the size of the matrix given by the relations is also decreased to 1/6 of the original. To the best of our knowledge, the 676-bit size is currently the record for solving the DLP in GF(3^{6n}).

4.1 Collection of Relations

In the collection of relations step, we collect many double smooth pairs (r, s). The simple idea for collecting them is factoring Polynomials (2) and (3) for all pairs

(r, s). This is not practical since we have to factor them about $(3^6)^B \times (3^6)^{B+1}$ times. In order to reduce the number of factorings, we use a sieving method. The idea of sieving is merely factoring Polynomials (2) and (3) of the pair (r, s), which has a high probability of becoming a double smooth pair. Such a pair is called a candidate.

The polynomial sieve [10] and the lattice sieve [26] are well-known sieving algorithms. Although the lattice sieve has been implemented in some experiments of the FFS [12,15,16], we implemented the polynomial sieve since r is fixed as a monic polynomial by the polynomial sieve in JL06-FFS, whereas neither r nor s is able to be fixed by the lattice sieve.

Polynomial Sieve. We describe the polynomial sieve in Polynomial (2), namely, $rm+s$. Notice that we can also sieve in Polynomial (3) with the same procedure. Moreover, we discuss the case where s is fixed and omit the details when r is fixed. By fixed s, we can lead r such that $rm+s$ is divisible by $\mathfrak{p} \in B_R$ or its power, where the degree of $\mathfrak{p}$ is not larger than B. Additionally, $(rm+s)+k\mathfrak{p}$ with $k \in \mathrm{GF}(3^6)[x]$ is also divisible by $\mathfrak{p}$. Hence, we can obtain all r of degree less than or equal to B such that $rm+s$ is divisible by $\mathfrak{p}$. After computing such all r for each $\mathfrak{p}$, we can obtain the pair (r, s) such that $rm+s$ is divisible by some $\mathfrak{p}$. If the summation of the degree of all $\mathfrak{p}$, which divide $rm+s$, reaches $\deg(rm+s)$, then $rm+s$ has a high probability of becoming B-smooth and the pair (r, s) becomes a candidate.

In this procedure, the most time-consuming work is to compute $r+k\mathfrak{p}$ for all $k \in \mathrm{GF}(3^6)[x]$ whose degree is not larger than B. In characteristic two, Gordon and McCurley proposed a method using binary gray codes [10] to compute these $r+k\mathfrak{p}$. Using ternary gray codes, we can also compute them efficiently in characteristic three.

In the polynomial sieve, we sieve with all powers of $\mathfrak{p}$ whose degree is not larger than B. Since B is very small, such as 1 or 2 in our experiments, the power of $\mathfrak{p}$ is only $\mathfrak{p}^2$ when $\deg(\mathfrak{p}) = 1$. Such polynomials are exceptional since there are 3^6 monic irreducible polynomials of degree 1 in $\mathrm{GF}(3^6)[x]$. In this way, we can obtain only candidates each of which generates a relation in Equation (5) (except that r and s are not relatively prime). Thus, we only check the greatest common divisor of r and s, but not the smoothness of Polynomials (2) and (3) using the B-smooth test [10].

Free Relation. By considering how a divisor $\langle \mathfrak{p} \rangle$ where $\mathfrak{p} \in B_R$ is factorized into divisors in $\mathrm{GF}(3^6)[x, y]/(H)$, namely, obtaining the following congruent expression that

$$H(x, y) \equiv \prod_{i=1}^{d} (y - t_i) \pmod{\mathfrak{p}},$$

where d is the degree of $H(x, y)$ on y, we can obtain a relation virtually for free, without the sieving procedure. We call such a relation a free relation.

The number of free relations depends on the degree d of $H(x, y)$ on y and the characteristic of the field treated in the FFS. In fact, there are about $\#B_A/d$ free relations in many cases and, furthermore, they increase when the characteristic is small. For example, in the case of $\mathrm{GF}(3^{6n})$ and $H(x, y) = y^6 + x$, there are about $\#B_A/2$ free relations since $y^6 + x$ is generally factored as $(y - t_1)^3(y - t_2)^3$ modulo $\mathfrak{p}$.

4.2 Linear Algebra

In the linear algebra step, we solve the linear equation depending on the relations. Specifically, we construct a matrix from the relations and reduce it to a much smaller one using the Galois action. After that, we solve the reduced linear equation modulo $(3^{6n} - 1)/(3^6 - 1)$, by applying the parallel Lanczos method described as [3]. In this section, we describe the Galois action and our ideas about parallel computation of the matrix operation.

Galois Action. Here, we consider to reduce unknowns of linear equations, using the Galois action which was presented in [18].

Let M' be the matrix given by the relations, whose row ${M'}_{(i)}$ means the i-th relation and j-th column ${M'}^{(j)}$ corresponds to the factorbase $\mathfrak{p}_j$. In order to use the Galois action, we choose the polynomial $f \in \mathrm{GF}(3^6)[x]$ satisfying that all coefficients of f are in $\mathrm{GF}(3)$ and $\deg(f) = n$, then we construct $\mathrm{GF}(3^{6n})$ as $\mathrm{GF}(3^6)[x]/(f)$. Let ϕ be the Frobenius power such that $\phi(\xi) = \xi^{3^n}$. As ϕ fixes the element x in $\mathrm{GF}(3)[x]/(f)$, we also have $\phi(x) = x$ in $\mathrm{GF}(3^6)[x]/(f)$ by the assumption of f. However, for an element $c \in \mathrm{GF}(3^6)\backslash\mathrm{GF}(3)$, ϕ does not fix c in $\mathrm{GF}(3^6)[x]/(f)$ by the above assumption that n is coprime to 6. The monic irreducible polynomial $\mathfrak{p}_j \in B_R$ of degree not larger than B, and we assume that $B = 1$ for convenience. In fact, $\mathfrak{p}_j = x + c_j$ where $c_j \in \mathrm{GF}(3^6)$ since $B = 1$, so we have

$$\phi(\mathfrak{p}_j) = \phi(x + c_j) = x + \phi(c_j)$$

in $\mathrm{GF}(3^6)[x]/(f)$. If c_j is not in $\mathrm{GF}(3)$, $c_j \neq \phi(c_j)$ in $\mathrm{GF}(3^6)[x]/(f)$. This fact implies that there are ordinarily many unknowns of linear equations, which can be rewritten by the other one via the Galois action. Clearly, for such $\mathfrak{p}_j$, there exists $\mathfrak{p}_{j'}$ satisfying that

$$\log_\gamma \mathfrak{p}_{j'} = \log_\gamma \phi(\mathfrak{p}_j) = 3^n \log_\gamma \mathfrak{p}_j \tag{12}$$

where $\mathfrak{p}_j \neq \mathfrak{p}_{j'}$. Therefore, we can remove the j'-th column ${M'}^{(j')}$ and set the j-th column ${M'}^{(j)}$ as ${M'}^{(j)} + 3^n {M'}^{(j')}$. Then we denote the new matrix M^* as the reduced M'. Notice that this technique is also used for the algebraic factorbase. Consequently, the number of unknowns is about 1/6 of the original; thus, the number of relations is reduced to about 1/6. In our implementation, we do not reduce the factorbase in the sieving phase (the computation is the same as the case without the Galois action). After sieving, we compress obtained relations using rewritable elements of the factorbase via the Galois action as Equation (12), and so the factorbase is reduced to about 1/6. Using this procedure, we almost do not lose the probability of obtaining the relation. Hence, this technique enables us to perform computations for the collection of relations step about 6 times as fast as before, and the linear algebra step can be also done about 6^2 times faster.

Parallel Lanczos method. The reduced matrix M^* is reconstructed to optimize first, then we apply the parallel Lanczos method to it. Before explaining the reconstruction, we begin with the explanation of the parallel computation. Assume that there are four nodes written as $N_{1,1}, N_{1,2}, N_{2,1}, N_{2,2}$ and each node has 4 or 8 cores. As the Figure 3, we partition the reconstructed matrix M into four matrices $M_{i,j}$, and each $M_{i,j}$ is allotted to node $N_{i,j}$ respectively. The given vector $\boldsymbol{v}$ is also partitioned into $\boldsymbol{v}_1, \boldsymbol{v}_2$, and $\boldsymbol{v}_j$ is given to nodes $N_{i,j}$, $N_{i',j}$ where $i \neq i'$. Moreover, $M_{i,j}$ is partitioned into L matrices A_ℓ when $N_{i,j}$ has L cores.

$$M\boldsymbol{v} = \left(\begin{array}{c|c} M_{1,1} & M_{1,2} \\ \hline M_{2,1} & M_{2,2} \end{array}\right) \left(\begin{array}{c} \boldsymbol{v}_1 \\ \hline \boldsymbol{v}_2 \end{array}\right). \qquad M_{i,j}\boldsymbol{v}_j := A\boldsymbol{v}_j = \left(\begin{array}{c} A_1 \\ \hline A_2 \\ \hline \vdots \\ \hline A_L \end{array}\right) \boldsymbol{v}_j.$$

Fig. 3. Partitioning M into four matrices $M_{i,j}$ and $M_{i,j}$ into L matrices A_ℓ

We now give the notation of the Lanczos method. The Lanczos method can operate only a symmetric matrix; however, the given matrix M is usually non-symmetric. Therefore, we try to solve the linear equation of the form $M^T M\boldsymbol{v} = \boldsymbol{\alpha}$, where $\boldsymbol{v}$ is an unknown column vector consisting of the logarithms of the factorbase and $\boldsymbol{\alpha}$ is the given column vector. Note that computing $M^T M$ is not efficient, so we compute the vector $\boldsymbol{u} = M\boldsymbol{v}$ and $M^T\boldsymbol{u}$. For more details about this computation is in [22].

After partitioning M, we perform a parallel computation for $\boldsymbol{u} := M\boldsymbol{v}$ and $\boldsymbol{w} := M^T\boldsymbol{u}$ with $M_{i,j}$. Let $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{u}_1$, and $\boldsymbol{u}_2$ be the partitioned vectors such that $\boldsymbol{v} = \boldsymbol{v}_1 \oplus \boldsymbol{v}_2$ and $\boldsymbol{u} = \boldsymbol{u}_1 \oplus \boldsymbol{u}_2$. From Algorithm 1, we obtain the partitioned vector $\boldsymbol{w}_i$ such that $\boldsymbol{w} = \boldsymbol{w}_i \oplus \boldsymbol{w}_{i'}$ in node $N_{i,j}$, where $i \in \{1, 2\}$ and $i' = 3 - i$. The symbol j' also means that $j' = 3 - j$ for $j \in \{1, 2\}$.

Lines 4, 5, and 6 describe the computation of $M^T u$. Note that in each node $N_{i,j}$, by regarding the column of $M_{i,j}$ as the row of $M^T_{j,i}$, we do not have to trade $M_{i,j}$ with $M^T_{j,i}$, namely, we can cut unnecessary operations.

Algorithm 1. (Computation with node $N_{i,j}$.)
Input : the partitioned matrix $M_{i,j}$ and the partitioned vector $\boldsymbol{v}_j$.
Output : the partitioned vector $\boldsymbol{w}_j$ such that $\boldsymbol{w}_1 \oplus \boldsymbol{w}_2 = M^T M\boldsymbol{v}$, where j is equal to 1 or 2.

```
[Step for computation of $u := Mv$]
1. $\boldsymbol{u}_{i,j} := M_{i,j}\boldsymbol{v}_j$.
2. Give $\boldsymbol{u}_{i,j}$ to Node $N_{i,j'}$ and receive $\boldsymbol{u}_{i,j'}$ from $N_{i,j'}$.
3. $\boldsymbol{u}_i := \boldsymbol{u}_{i,j} + \boldsymbol{u}_{i,j'}$.
[Step for computation of $w := M^T u$]
4. $\boldsymbol{w}_{i,j} := M^T_{i,j}\boldsymbol{u}_i$.
5. Give $\boldsymbol{w}_{i,j}$ to Node $N_{i',j}$ and receive $\boldsymbol{w}_{i',j}$ from $N_{i',j}$.
6. $\boldsymbol{w}_j := \boldsymbol{w}_{i,j} + \boldsymbol{w}_{i',j}$.
```

We have discussed the parallel computations among nodes, and now we move on to the parallel computations among cores in one node. Here, A_ℓ denotes the partitioned matrix of $M_{i,j}$ such that $M_{i,j} = \oplus_{\ell=1}^{L} A_\ell$. From Algorithm 2, we can easily obtain $A_\ell \boldsymbol{v}_j$, and then we set the new vector $\boldsymbol{u}_{i,j} = (A_1\boldsymbol{v}_j, \ldots, A_L\boldsymbol{v}_j)^T$, where L is the number of cores in the same node. Similarly, we can easily obtain $A_\ell^T \boldsymbol{u}_i$ and compute $\boldsymbol{w}_{i,j} = \sum_{\ell=1}^{L} A_\ell^T \boldsymbol{u}_i$ by using Algorithm 3.

Algorithm 2. (Parallel computation of $M_{i,j}\boldsymbol{v}_j$ among L cores in the same node.)
Input : the partitioned matrix $A := M_{i,j}$ whose size is $s \times t$ and the partitioned t-vector $\boldsymbol{v}_j$.
Output : the partitioned vector $\boldsymbol{u}_{i,j}$ such that $\boldsymbol{u}_{i,j} = A\boldsymbol{v}_j$.
```
1. Compute $\boldsymbol{b}_\ell := A_\ell \boldsymbol{v}_j$ for $\ell = 1$ to $\ell = L$ in parallel.
2. $\boldsymbol{u}_{i,j} = \oplus_{\ell=1}^{L} \boldsymbol{b}_\ell$.
```

Algorithm 3. (Parallel computation of $M_{i,j}^T\boldsymbol{u}_i$ among L cores in the same node.)
Input : the partitioned matrix $A := M_{i,j}$ whose size is $s \times t$ and the partitioned s-vector $\boldsymbol{u}_i$.
Output : the partitioned vector $\boldsymbol{w}_{i,j}$ such that $\boldsymbol{w}_{i,j} = A^T\boldsymbol{u}_i$.
```
1. Compute $\boldsymbol{c}_\ell := A_\ell^T \boldsymbol{u}_i$ for $\ell = 1$ to $\ell = L$ in parallel.
2. $\boldsymbol{w}_{i,j} = \sum_{\ell=1}^{L} \boldsymbol{c}_\ell$.
```

From the parallel computations of $M_{i,j}\boldsymbol{v}_j$ and so on, we obtain the vector $M^T M\boldsymbol{v}$ from Algorithm 1 and 2. Therefore, we need to reconstruct M so that each node has the balanced calculation amount of computing $M_{i,j}\boldsymbol{v}_j$ and so on. It is clear that the calculation amount depends on the number of non-zero elements in the allotted matrix, and the distribution of non-zero elements in M is not uniformity. In fact, the number of non-zero elements in a column of M is not balanced, but that in a row is balanced. Thus, we reconstruct the new matrix M so that the number of non-zero elements in $M_{1,1}$ and $M_{2,1}$ is almost equal to that in $M_{1,2}$ and $M_{2,2}$ by sorting columns of M^* defined in the section of the Galois action. We perform a similar strategy as above for the parallel computation among cores in the same node, namely, A is partitioned into 4 or 8 smaller matrices A_ℓ so that each A_ℓ has almost the same number of non-zero elements.

4.3 Computation Results

In this section, we describe our computation results of the 676-bit DLP in $\mathrm{GF}(3^{6\cdot 71})$, which contains a multiplicative subgroup whose order is a 112-bit prime. We construct $\mathrm{GF}(3^6)$ as $\mathrm{GF}(3)[z]/(z^6 + 2z + 2)$ and define a mapping $\psi : \mathbb{Z} \to \mathrm{GF}(3^6)[x]$, such that $\psi^{-1} : z \mapsto 3, x \mapsto 3^6$, in order to represent the element in $\mathrm{GF}(3^6)[x]$.

In the polynomial selection step, we set $H(x, y) = y^6 + x$ in order to use the Galois action. Moreover, we select $m \in \mathrm{GF}(3^6)[x]$ such that all its coefficients are in $\mathrm{GF}(3)$ to construct f whose coefficients are also in $\mathrm{GF}(3)$. By an easy computation, we obtain proper m and f as follows,

$m = \psi$ (0x456bc 60e76c11 1e679735 c929fc55)

$f = \psi$ (0x9 2d3e5daf 5ac01130 4e6909f7 09cc8833 baa757d3
17dc6f99 9c8b98b5 ab8baa01 d68ec151 aec39e2e ed081c79
d851066b 3ffb2a4f a3e19c1e cef46675 0918a26d 9c7cacd4
8d74ccfe 2c1d3b79 e81e6138 ab06aef4).

Then, GF(3^{6n}) is constructed as GF(3^6)$[x]/(f)$. When we set the smoothness bound $B = 2$, there are 266,085 elements in the rational factorbase and 265,721 elements in the algebraic factorbase, so we need to collect at least 531,806 relations. However, the size of the sieving area when $B = 2$ is too small to collect enough relations.

We settle this problem by using the Galois action, since we can considerably reduce the number of required elements in the factorbase described in Section 4.2. In fact, we need only 88,674 relations, and so this number is about 1/6 the number of the originally required relations.

Moreover, we deal with free relations which are obtained without sieving. If we choose $H(x, y)$ as $y^6 + x$, then it is fortunately factored as $(y - t_1)^3(y - t_2)^3$ (mod $\mathfrak{p}$) for most of elements $\mathfrak{p}$ in the factorbase, and so there are 132,860 ($\approx \#B_A/2$) free relations. Even if we delete many duplicates which are produced by using the Galois action, 22,155 free relations remain. Thus, we only have to find at least 66,519 relations in the collection of relations step, and this number is about 1/8 that of the originally required relations.

In the collection of relations step, we use the polynomial sieve described in Section 4.1 and compute relations using five nodes, each consisting of Intel Quad-Core Xeon E5440 (2.83 GHz) $\times$ 2 CPUs with 16-GB RAM, one node consisting of Intel Quad-Core Xeon X5355 (2.66 GHz) $\times$ 2 CPUs with 16-GB RAM, and twelve nodes, each consisting of Intel Quad-Core Xeon L5420 (2.33 GHz) $\times$ 1 CPU with 4-GB RAM, total of 96 cores. In 18 days of computation, after removing duplicates, we found 66,646 relations. Thus, we obtained a total of 88,801 relations, which are enough to solve the linear equation in Equation (7).

The linear equation constructed from the relations has to be solved modulo $(3^{6\cdot 71} - 1)/(3^6 - 1)$; however, the Lanczos method may fail when the modulus has a small prime factor. Therefore, we work modulo the factor N_i of $(3^{6\cdot 71} - 1)/(3^6 - 1)$,

$$\begin{aligned}
N_1 &= (3^{2\cdot 71} + 3^{71} + 1)/(13 \cdot 5113),\\
N_2 &= (3^{2\cdot 71} - 3^{71} + 1)/(7 \cdot 210019 \cdot 49682251 \cdot 55126531),\\
N_3 &= (3^{71} + 1)/(2^2 \cdot 853 \cdot 2131 \cdot 82219),\\
N_4 &= (3^{71} - 1)/2.
\end{aligned}$$

where every prime factor of N_i is larger than 30 bits and N_i is relatively prime to each other.

We use a cluster with four nodes, each consisting of Intel Quad-Core Xeon E5440 (2.83 GHz) $\times$ 2 CPUs with 16-GB RAM, and three clusters with four nodes, each consisting of Intel Quad-Core Xeon L5420 (2.33 GHz) $\times$ 1 CPU with

4-GB RAM. With about 12 hours computation, we solve the linear equation modulo N_i via the parallel Lanczos method with the four nodes described in Section 4.2 on each cluster. With the Chinese remainder theorem and the Galois action of ϕ, we solved discrete logarithms of the elements in the factorbase modulo $N = \prod_{i=1}^{4} N_i$.

In the individual logarithm step, our target of computing the logarithm is the element

$$\begin{aligned}\pi(x) &= \psi(\lfloor \pi \times 10^{202} \rfloor)\\ &= (z^4 + z^3 + 2z^2 + 1)x^{70} + \cdots + (z^5 + 2z^4 + 2z^3 + z^2 + 2)\end{aligned}$$

in basis $\gamma = \psi(\texttt{0x456})$. We choose the representation of $\pi(x)$ as a product of elements of degree at most 7 as follows:

$$\gamma^{\tau}\pi(x) \equiv z_1/z_2 \pmod f),\ \textit{where}$$

$$\begin{aligned}z_1 &= \psi(\texttt{0x333}) \times \psi(\texttt{0x345}) \times \psi(\texttt{0x427}) \times \psi(\texttt{0x43b}) \times \psi(\texttt{0x4c3})\\ &\quad \times\psi(\texttt{0xd909 66c7e3ec}) \times \psi(\texttt{0x293996d cc380672})\\ &\quad \times\psi(\texttt{0x3ff378e 3d4659d0}) \times \psi(\texttt{0x6 27d6c281 0a0fc5a2})\\ &\quad \times\psi(\texttt{0x8 f4797e29 a9ec3b4a}),\\ z_2 &= \psi(\texttt{0x318}) \times \psi(\texttt{0x45 4c6fbfd4}) \times \psi(\texttt{0x54 c69e6f97})\\ &\quad \times\psi(\texttt{0x1686d 42782189}) \times \psi(\texttt{0x3cf67a5 84055cd8})\\ &\quad \times\psi(\texttt{0x8 f68ab2e2 5d2bc04f}) \times \psi(\texttt{0xb cc56922c f651b383}),\\ \tau &= \quad \texttt{0x2 0f822e8c ac48792a e2aea337 c9002b49 bbf1b864}\\ &\quad \texttt{43a6111b 24c5593d e44daf43 e26de26e 1f85f982 1ba485b3}\\ &\quad \texttt{beda74bd f782626d 6cd38bb2 8f829867 5dc04adc f8741c24},\end{aligned}$$

and z_1, z_2 are 7-smooth. Then, we compute the logarithms of z_1 and z_2 in basis γ using the special-$\mathfrak{q}$ descent technique [16,18]. With about 14 days computation using five nodes, each consisting of Intel Quad-Core Xeon E5440 (2.83 GHz) × 2 CPUs with 16-GB RAM, and one node consisting of Intel Quad-Core Xeon X5355 (2.66 GHz) × 2 CPUs with 16-GB RAM, we compute the logarithms,

$$\begin{aligned}\log_\gamma z_1 \equiv &\quad \texttt{0x3 fc71c577 10be8e3f e7af0fba e00e711f 0ad6dd50}\\ &\texttt{38fb8f26 c0fadb3b 448cab2f 67671247 285f9e95 dc501717}\\ &\texttt{d9def844 a75f9e58 f04a9bd2 3a5d0fdb 8f8ebb9f fea4deea},\\ \log_\gamma z_2 \equiv &\quad \texttt{0x4 82febaec ae4382e0 e651f577 09df4e7d 99d99d34}\\ &\texttt{03db5d5e 521c4e2b da89ec33 6c9d45d6 2dd1f982 2f198fb2}\\ &\texttt{6c069414 3b0b1544 ece8e4b1 5304872f 6ff261fd 03b271c7}.\end{aligned}$$

modulo N, and so we obtain $\log_\gamma \pi(x) \bmod N$.

The logarithm in multiplicative subgroups of less than 30 bits are computed using the Pollard's ρ method in a minute. Using the Pohlig-Hellman method, we compute the logarithm $\log_\gamma \pi(x)$:

Table 3. Records for solving the DLP in finite fields

Finite Fields	GF(p)	GF(2^n)	GF(p^3)	GF(p^{30})	**GF(3^{6n})**
Reference	[21]	[15]	[20]	[18]	**This Work**
Date	Feb. 5, 2007	Sep. 22, 2005	Aug. 23, 2006	Nov. 9, 2005	**Dec. 9, 2009**
Algorithm	NFS*	JL02-FFS	JLSV06-NFS†	JL06-FFS-2‡	**JL06-FFS**
Collection of Relations	Many CPUs¶	4 nodes of 16 Itanium 2 (1.3GHz)	16 Alpha processors (1.15GHz)	16 Alpha processors (1.15GHz)	**Xeon (2.83GHz) 96 cores**
Linear Algebra	12–24 Xeon (3.2GHz)	4 nodes of 16 Itanium 2 (1.3GHz)	16 Alpha processors (1.15GHz)	16 Alpha processors (1.15GHz)	**Xeon (2.83GHz) 80 cores**
Timing	33 days	17 days	19 days	12 hours	**33 days**
Bit Size	532	613	394	556	**676**

*NFS: Number Field Sieve [9,17]. †JLSV06-NFS: NFS in the medium prime case [20].
‡See footnote ^{2} on page 2. ¶There are no detailed descriptions of computational resources in [21].

$\log_\gamma \pi(x) =$ `0x8 78b54797 2fb6ff9b 57add5d5 11f69de6 a3853f98`
`68d53cc0 5b531076 2872ac6a 320874bf ba6d66d6 8e5e245f`
`39778f02 31ae791a acbab8c7 5ee6850c 9f5df0e5 f6b8ab0b`
`95d8bdb1 aea95b1f bad82465 25590f66`

and completely solve the DLP in GF($3^{6\cdot 71}$) of 676-bit.

4.4 For Larger Extension Degrees

We have solved the DLP in GF(3^{6n}) for n in the experimental class, where the smoothness bound B (i.e., B_{06}) is less than or equal to 2 (ref. Table 1). Note that the size of the sieving area increases $(3^6)^2$-fold if the smoothness bound B increases by one (see Form (11)). However, we expect that, if we set $B = 3$, the DLP in GF($3^{6\cdot 97}$) might be computed for several years by using dozens of our computational resources through various techniques such as large prime variation, block sieving and sieving via bucket sort [29,4], and SIMD implementation.

5 Concluding Remarks

In this study, we implemented a new variant of the FFS in GF(3^{6n}) (n is a prime), proposed by Joux and Lercier in 2006 [18], and compared it with the earlier variant, which was also proposed by Joux and Lercier in 2002 [16] with practical experiments. In solving the DLP in GF(3^{6n}), these two variants of the FFS have the same asymptotic complexity, but we expected the new variant to be more efficient than the earlier one in some extension degrees n. From our experimental results, we confirmed this forecast when the extension degree $n = 19, 61$. Moreover, with our implementations, we succeeded in solving the DLP in GF($3^{6\cdot 71}$) of 676-bit size with about 33 days computation.

We have experimented with the DLP in GF(3^{6n}) required for pairing-based cryptosystems. The security of pairing-based cryptosystems relies on the difficulty of the DLP in various finite fields, for example, GF(2^{4n}) and GF(p^{12}). Table 3 presents the current records for solving the DLP in various finite fields. All the DLPs used for pairing-based cryptosystems have not examined yet. It is an open problem to analyze the hardness of the DLP with practical key sizes in such finite fields.

References

1. Adleman, L.M.: The function field sieve. In: Huang, M.-D.A., Adleman, L.M. (eds.) ANTS 1994. LNCS, vol. 877, pp. 108–121. Springer, Heidelberg (1994)
2. Adleman, L.M., Huang, M.-D.A.: Function field sieve method for discrete logarithms over finite fields. Inform. and Comput. 151, 5–16 (1999)
3. Aoki, K., Shimoyama, T., Ueda, H.: Experiments on the linear algebra step in the number field sieve. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC 2007. LNCS, vol. 4752, pp. 58–73. Springer, Heidelberg (2007)
4. Aoki, K., Ueda, H.: Sieving using bucket sort. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 92–102. Springer, Heidelberg (2004)
5. Barreto, P.S.L.M., Galbraith, S., ÓhÉigeartaigh, C., Scott., M.: Efficient pairing computation on supersingular abelian varieties. Des. Codes Cryptogr. 42(3), 239–271 (2007)
6. Beuchat, J.-L., Brisebarre, N., Detrey, J., Okamoto, E., Shirase, M., Takagi, T.: Algorithms and arithmetic operators for computing the η_T pairing in characteristic three. IEEE Trans. Comput. 57(11), 1454–1468 (2008)
7. Boneh, D., Crescenzo, D., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
8. Boneh, D., Franklin, M.: Identity based encryption from the Weil pairing. SIAM J. Comput. 32(3), 586–615 (2003)
9. Gordon, D.M.: Discrete logarithms in GF(p) using the number field sieve. SIAM J. Discrete Math. 6(1), 124–138 (1993)
10. Gordon, D.M., McCurley, K.S.: Massively parallel computation of discrete logarithms. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 312–323. Springer, Heidelberg (1993)
11. Granger, R.: Estimates for discrete logarithm computations in finite fields of small characteristic. In: Paterson, K.G. (ed.) Cryptography and Coding 2003. LNCS, vol. 2898, pp. 190–206. Springer, Heidelberg (2003)
12. Granger, R., Holt, A.J., Page, D., Smart, N.P., Vercauteren, F.: Function field sieve in characteristic three. In: Buell, D.A. (ed.) ANTS 2004. LNCS, vol. 3076, pp. 223–234. Springer, Heidelberg (2004)
13. Granger, R., Page, D., Stam, M.: Hardware and software normal basis arithmetic for pairing-based cryptography in characteristic three. IEEE Trans. Comput. 54(7), 852–860 (2005)
14. Hankerson, D., Menezes, A., Scott, M.: Software implementation of pairings. In: Identity Based Cryptography, pp. 188–206 (2009)
15. Joux, A., et al.: Discrete logarithms in GF(2^{607}) and GF(2^{613}). Posting to the Number Theory List (2005), http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0509&L=nmbrthry&T=0&P=3690

16. Joux, A., Lercier, R.: The function field sieve is quite special. In: Fieker, C., Kohel, D.R. (eds.) ANTS 2002. LNCS, vol. 2369, pp. 431–445. Springer, Heidelberg (2002)
17. Joux, A., Lercier, R.: Improvements to the general number field sieve for discrete logarithms in prime fields. A comparison with the Gaussian integer method. Math. Comp. 72(242), 953–967 (2002)
18. Joux, A., Lercier, R.: The function field sieve in the medium prime case. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 254–270. Springer, Heidelberg (2006)
19. Joux, A., Lercier, R., Naccache, D., Thome, E.: Oracle-assisted static Diffie-Hellman is easier than discrete logarithms. In: Parker, M.G. (ed.) IMACC 2009. LNCS, vol. 5921, pp. 351–367. Springer, Heidelberg (2009)
20. Joux, A., Lercier, R., Smart, N.P., Vercauteren, F.: The number field sieve in the medium prime case. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 326–344. Springer, Heidelberg (2006)
21. Kleinjung, T., et al.: Discrete logarithms in GF(p) - 160 digits. Posting to the Number Theory List (2007), http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0702&L=nmbrthry&T=0&P=194
22. LaMacchia, B.A., Odlyzko, A.M.: Solving large sparse linear systems over finite fields. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 109–133. Springer, Heidelberg (1991)
23. Matsumoto, R.: Using C_{ab} curves in the function field sieve. IEICE Trans. Fundamentals E82-A, 551–552 (1999)
24. Menezes, A.J., Okamoto, T., Vanstone, S.: Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Trans. Inform. Theory 39(5), 1639–1646 (1993)
25. Page, D., Smart, N.P., Vercauteren, F.: A comparison of MNT curves and supersingular curves. Appl. Algebra Engrg. Comm. Comput. 17(5), 379–392 (2006)
26. Pollard, J.: The lattice sieve. The Development of the Number Field Sieve, 43–49 (1991)
27. Pomerance, C., Smith, J.W.: Reduction of huge, sparse matrices over finite fields via created catastrophes. Experiment. Math. 1(2), 89–94 (1992)
28. Schirokauer, O.: The special function field sieve. SIAM J. Discrete Math. 16(1), 81–98 (2003)
29. Wambach, G., Wettig, H.: Block sieving algorithms. Technical Report 190, Informatik, Universität zu Köln (1995)
30. Wiedemann, D.H.: Solving sparse linear equations over finite fields. IEEE Trans. Inform. Theory 32(1), 54–62 (1986)

Using Equivalence Classes to Accelerate Solving the Discrete Logarithm Problem in a Short Interval

Steven D. Galbraith[1,⋆] and Raminder S. Ruprai[2]

[1] Mathematics Department, Auckland University, Auckland, New Zealand
s.galbraith@math.auckland.ac.nz
[2] Mathematics Department, Royal Holloway University of London, Egham, Surrey TW20 0EX, UK
raminder@email.com

Abstract. The Pollard kangaroo method solves the discrete logarithm problem (DLP) in an interval of size N with heuristic average case expected running time approximately $2\sqrt{N}$ group operations. It is well-known that the Pollard rho method can be sped-up by using equivalence classes (such as orbits of points under an efficiently computed group homomorphism), but such ideas have not been used for the DLP in an interval. Indeed, it seems impossible to implement the standard kangaroo method with equivalence classes.

The main result of the paper is to give an algorithm, building on work of Gaudry and Schost, to solve the DLP in an interval of size N with heuristic average case expected running time of close to $1.36\sqrt{N}$ group operations for groups with fast inversion. In practice the algorithm is not quite this fast, due to the usual problems with pseudorandom walks such as fruitless cycles. In addition, we present experimental results.

Keywords: discrete logarithm problem (DLP), elliptic curves, negation map, efficiently computable group homomorphisms.

1 Introduction

The discrete logarithm problem (DLP) in an interval is the problem: Given g, h in a group G and $N \in \mathbb{Z}_{>0}$ such that $h = g^n$ for some $0 \leq n \leq N$ (where N is less than the order of g), to compute n. This problem arises naturally in a number of contexts, for example the DLP with c-bit exponents (c-DLSE) [15,23,21], decryption in the Boneh-Goh-Nissim homomorphic encryption scheme [1], counting points on curves or abelian varieties over finite fields [14], the analysis of the strong Diffie-Hellman problem [3,17], and side-channel or small subgroup attacks [16,18].

One can solve the DLP in an interval using the baby-step-giant-step algorithm in at worst $2\sqrt{N}$ group operations (or, with minor modifications, with average

⋆ This work supported by EPSRC grant EP/D069904/1.

case running time of $\sqrt{2N}$ group operations). But this method also requires $O(\sqrt{N})$ group elements of storage.

Pollard [24] developed the kangaroo algorithm precisely with this application in mind. Using distinguished points, van Oorschot and Wiener [22] (also see Pollard [26]) achieve a heuristic average case expected complexity of essentially $2\sqrt{N}$ group operations and low storage. We summarise this algorithm in Appendix A. Note that this algorithm has success probability of 1, as do all algorithms in this paper. These algorithms can also be parallelised (or distributed) with a linear speedup. For comparison, the Pollard rho method [24] has heuristic expected running time of $\sqrt{\pi r/2} \approx 1.25\sqrt{r}$ operations if g has order r. All complexity statements in this paper rely on heuristic assumptions; for steps toward a rigorous analysis of the kangaroo method please see Montenegro and Tetali [19].

Gaudry and Schost [14] (building on earlier work of Gaudry and Harley [13]) presented a different approach to solve this problem using a birthday paradox style analysis. Whilst their algorithm is not as fast as that of van Oorschot and Wiener, it is easily parallelisable and importantly there is no requirement to know the number of clients or processors before the algorithm begins. Parallelising the Gaudry-Schost algorithm gives a linear speedup and this also applies to all the algorithms in this paper. For brevity we state all running times for the serial case. The average expected running time of their algorithm is $2.08\sqrt{N}$ group operations on a serial computer (the algorithm of Gaudry and Harley [13] is less efficient). We present their algorithm and recall the analysis of its complexity in Section 2.

Gallant, Lambert and Vanstone [11] and Wiener and Zuccherato [29] showed that the Pollard rho method can be used with equivalence classes (orbits of group elements under an fast computable group homomorphism) to achieve a constant speedup in some groups. In particular, for elliptic curves the rho algorithm can be sped-up by a factor of $\sqrt{2}$ using the equivalence class $\{u, u^{-1}\}$ where u is a group element (which is more commonly written as $\{P, -P\}$ in the case of elliptic curves). In practice, the running times are not so good, since the algorithms use pseudorandom walks which do not behave exactly like true random walks (in particular, walks can fall into short cycles and hence never arrive at a distinguished point; these are called "fruitless cycles" and have been analysed by Duursma, Gaudry and Morain [6] and Bos, Kleinjung and Lenstra [2]).

It seems to be impossible to combine the standard kangaroo method with equivalence classes in general (Section 19.6.3 of [5] claims it can be done but gives no details, and this seems to be an error). Hence, it is necessary to consider other algorithms. A natural observation is that, for a DLP instance (g, h) in an interval of even length N, one can set $h' = hg^{-N/2}$ and then solve $h' = g^n$ where $-N/2 \leq n \leq N/2$. If the discrete logarithm of u lies in the interval $[-N/2, N/2]$ then the equivalence class $\{u, u^{-1}\}$ does correspond to a pair of group elements in the region of interest.

Very recently Pollard [25] developed two new variants of the kangaroo method which require inversion of just one or two group elements. Pollard's three and four kangaroo variants have heuristic running times of roughly $1.82\sqrt{N}$ and

$1.71\sqrt{N}$ group operations respectively. More details about these algorithms will appear in forthcoming joint work.

In Sections 3 and 4 we show how to speed up the Gaudry-Schost method in groups with fast inversion (such as elliptic curves, tori, LUC and XTR). Here fast inversion means that computing u^{-1} for any u in the group is much faster than a general group operation. We also present a further speedup by modifying the search region. Our main result is a method to solve the DLP in an interval in approximately $1.36\sqrt{N}$ group operations. The result uses a new variant of the birthday paradox which is developed in [10]. The theoretical analysis of the algorithm assumes it is run using a truly random walk. In practice one implements the algorithm using a pseudorandom walk which has a number of undesirable consequences, in particular the existence of fruitless cycles. In Section 5 we present experimental results which give a better idea of the actual performance in practice (though it is likely that these figures can be improved).

Our algorithm, as with Gaudry-Schost, requires low storage and can be parallelised with linear speedup very easily.

We indicate in Appendix B how to speed up the Gaudry-Schost algorithm for the multi-dimensional DLP using equivalence classes. A precise analysis of the algorithms in Appendix B is currently an open problem.

2 The Gaudry-Schost Algorithm

To introduce notation and the central ideas, we recall the Gaudry-Schost algorithm [14]. The basic idea is the same as the kangaroo algorithm of Pollard in the van Oorschot and Wiener [22] formulation. Let g and h be the DLP instance we wish to solve, with $h = g^n$ for some integer $-N/2 \leq n \leq N/2$. We run a large number of pseudorandom walks (possibly distributed over a large number of processors). Half the walks are "tame walks", which means that every element in the walk is of the form g^a where the integer a is known. The other half are "wild walks", which means that every element is of the form hg^a where the integer a is known. As is typical in this subject, we visualise the group in terms of the 'exponent space'. More precisely, define the 'tame set'

$$T = [-N/2, N/2]$$

(where by $[N_1, N_2]$ we mean $\{a \in \mathbb{Z} : N_1 \leq a \leq N_2\}$) and the 'wild set'

$$W = n + T = \{n + a : a \in [-N/2, N/2]\}.$$

Although T and W are of the same size, W is a translation of T and of course we do not know the value of n. A tame walk is a sequence of points g^a where $a \in T$ and a wild walk is a sequence of points $g^b = hg^a$ with $b \in W$.

Each walk proceeds until a distinguished point is hit. This distinguished point is then stored on a server, together with the corresponding exponent a and a flag indicating which sort of walk it was. This data is analogous to the 'trap' set in the standard Pollard kangaroo method [24,26]. When the same distinguished point is visited by two different types of walk we have the "tame-wild collision" $g^{a_1} = hg^{a_2}$ and one solves the DLP. We stress that the algorithm continues until the DLP is solved. Hence the probability of success is 1.

The main difference between the Gaudry-Schost algorithm and the kangaroo algorithm is that when a distinguished point is hit, Gaudry and Schost restart the walk from a random starting point in a certain range, whereas the kangaroos keep on running. The theoretical analysis is different too: Gaudry and Schost use a variant of the birthday paradox whereas Pollard and van Oorschot and Wiener use a different probabilistic argument (see Appendix A).

2.1 Theoretical Analysis

We now recall the precise analysis of the idealised version (i.e., using a truly random walk, rather than a pseudorandom walk) of the Gaudry-Schost algorithm [14]. Gaudry and Schost use the following variant of the birthday paradox, which we will call the Tame-Wild birthday paradox.

Theorem 1. *When sampling uniformly at random from a set of size $R \in \mathbb{N}$, with replacement, and alternately recording the element selected in 2 different lists then the expected number of selections that need to be made in total before we have a coincidence between the lists is $\sqrt{\pi R} + O(1)$.*

Proofs of this theorem can be found in Selivanov [28] or in [27] (which derives it from a result of Nishimura and Sibuya [20]). For simplicity we will omit the $O(1)$ term from all subsequent running times.

Since tame points lie in T and wild points lie in W a collision between tame and wild points can only occur in $T \cap W$. We call such a collision 'tame-wild' and this is analogous to a coincidence between the lists in Theorem 1, so we apply Theorem 1 in the case $R = |T \cap W|$.

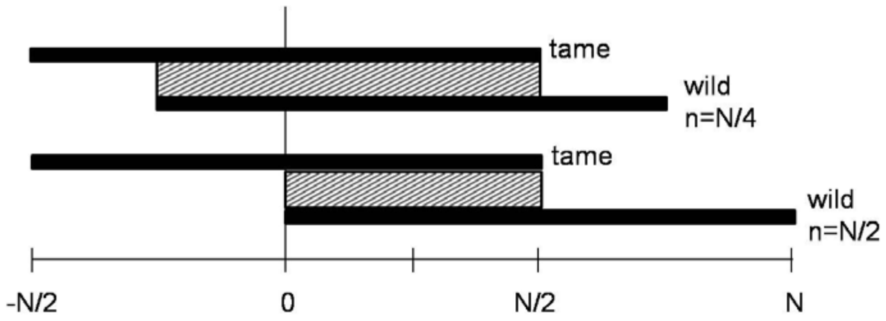


Fig. 1. Overlap between T and W. The sets T and W are represented by black horizontal bars and the shading between them shows the length of the overlap. The first case is when $n = N/4$ and the second case is $n = N/2$.

Figure 1 presents $T \cap W$ in two cases. The first case is $h = g^n$ for $n = N/4$ so $|T \cap W| = 3N/4$ (this is the 'average case'). The second case is $h = g^n$ for $n = N/2$ so $|T \cap W| = N/2$ (which is the 'worst case').

Theorem 2. *(Gaudry-Schost [14]) Let notation be as above. If elements are sampled uniformly at random with replacement alternately from T and W and recorded, the expectation, over all problem instances, of the number of selections before a tame-wild collision is $2.08\sqrt{N}$.*

Proof. The running time of the Gaudry-Schost algorithm is dependent on the problem instance $h = g^n$ but, by symmetry, we can restrict to the case $0 \leq n \leq N/2$. We write this as $h = g^{xN}$ where $x \in [0, 1/2]$.

Let $R = |T \cap W| = N(1 - x)$. By Theorem 1 we expect to need to sample $\sqrt{\pi R}$ elements (half of each type) of $T \cap W$ to find a collision. To select $\frac{1}{2}\sqrt{\pi R}$ elements in $T \cap W$ when sampling uniformly from T requires selecting

$$\frac{|T|}{|R|}\tfrac{1}{2}\sqrt{\pi R} = \tfrac{1}{2}\sqrt{\pi N/(1-x)}.$$

The same argument applies to W. Hence, the expected running time of the algorithm is $\sqrt{\pi N/(1-x)}$ group operations. Note that this is the expected value of the running time, over all choices for the random walk, for a specific problem instance.

We now average this over all problem instances as

$$2\int_0^{1/2}(1-x)^{-1/2}\sqrt{\pi N}dx = 2\sqrt{\pi N}\left[2-\sqrt{2}\right] = 2(2-\sqrt{2})\sqrt{\pi N} \approx 2.08\sqrt{N}.$$

□

This result has been improved to $2.05\sqrt{N}$ by using smaller sets for T and W in [8].

2.2 Pseudorandom Walks and Practical Considerations

Gaudry and Schost present the result in Theorem 2 but they also consider the practical implementation of the algorithm. First, to reduce storage, one does not record every element sampled by the pseudorandom walk but instead uses distinguished points. If we let θ be the probability that an element of the group is a distinguished point then walks are of length $1/\theta$ on average and we require storage of around $\theta\sqrt{N}$ group elements.

Second, it is necessary to use a pseudorandom walk which performs close enough to sampling uniformly at random that the Tame-Wild birthday paradox still applies. Gaudry and Schost, as with the kangaroo method, partition the group into, say, 32 subsets and use a pseudorandom walk where each step is a multiplication of the current group element by g^{a_i}, where a_i is a fixed small positive integer, if the current group element lies in the i-th block of the partition. Our algorithms will necessarily have random walks which step in a "side-to-side" manner, since the equivalence class representative of a group element could be its inverse and steps to the right from the inverse of a group element are the same as steps to the left from the original element. Hence, though we take $a_i \in \mathbb{N}$ and each step is multiplication by g^{a_i}, in practice the walks look like the jumps are of lengths $\pm a_i$. We denote by m the mean of the integers $|a_i|$ and call it the mean absolute step size. For the analysis we recall the following result (note that the mean absolute step size in this walk is $\frac{1}{2}$).

Lemma 1. *(Cofman, Flajolet, Flatto and Hofri [4]) Let* $y_0, y_1, \ldots, y_k$ *be a symmetric random walk that starts at the origin* ($y_0 = 0$) *and takes steps uniformly distributed in* $[-1, +1]$ *then the expected maximum excursion is*

$$E(\max\{|y_i| : 0 \le i \le k\}) = \sqrt{\frac{2k}{3\pi}} + O(1)$$

The average 'distance' covered by a random walk, from its starting point to when it hits a distinguished point, is therefore $m/\sqrt{\theta}$. To have good random walks it is essential that this value is sufficiently large so that each walk covers a reasonable proportion of the tame or wild set. If not, then the walks stay very close to their starting point and the probability of two walks colliding is small. On the other hand, when $m/\sqrt{\theta}$ is large then there is a good chance that the pseudorandom walks will sometimes travel outside T or W. Steps outside the regions of interest cannot be included in our probabilistic analysis and so such steps are "wasted". To reduce these wasted steps it is necessary to start walks inside a subset of T and W. More details about how to do this are given in [8].

We therefore state the following heuristic result. The factor $1 + \epsilon$ takes into account the failure of a pseudorandom walk to behave exactly like a random walk, in particular due to effects at the boundaries of the regions.

Heuristic 1. *The average expected running time for the Gaudry-Schost algorithm to solve the DLP in an interval of size* N *is* $2.08(1+\epsilon)\sqrt{N} + 1/\theta$ *group operations for some small* $\epsilon > 0$.

We admit that the statement of Heuristic 1 (and Heuristic 2 later) is essentially vacuous (for example, is $\epsilon = 1$ "small"?). We would like to be able to replace ϵ by $o(1)$. This may be reasonable for Heuristic 1 but it seems unlikely to be reasonable for Heuristic 2. Certainly we feel it is reasonable to suggest that ϵ can be less than 0.1 in both Heuristics 1 and 2.

The standard Gaudry-Schost algorithm is therefore not as fast as the van Oorschot and Wiener version of the Pollard kangaroo method. Nevertheless, we will improve upon their approach in groups with fast inversion, to obtain a method faster than any known method based on kangaroos.

3 Equivalence Classes

Following the work of Gallant, Lambert and Vanstone [11] and Wiener and Zuccherato [29] it is natural to consider a pseudorandom walk on a set of equivalence classes. For the DLP in an interval this only seems to give an improvement when the equivalence class is a set of group elements all of whose discrete logarithms lie in the interval. Groups with fast inversion are good candidates for this.

It is necessary to be able to compute a unique representative of the equivalence class so that one can define a deterministic pseudorandom walk on the equivalence classes. For example consider the group of points on an elliptic curve $E : y^2 = x^3 + Ax + B$ over a finite field $\mathbb{F}_q$ where q is an odd prime. If we let $P = (x_P, y_P) \in E(\mathbb{F}_q)$ then the inverse of P is simply $-P = (x_P, -y_P)$. Now we need a rule to define a unique representative for each equivalence class $\{P, -P\}$. A simple rule in this case is: treat the y-coordinate of P as an integer $0 \leq y_P < q$ and let the unique representative be $(x_P, \min\{y_P, q - y_P\})$. The pseudorandom walk is then defined using the unique equivalence class representative.

If we denote elements of the group by their discrete logarithms and order those in the interval $[-N/2, N/2]$, then the two elements in an equivalence class are equidistant from the centre of the interval. A step to the right for one representative of the equivalence class corresponds to a step to the left for the other. Hence, when using equivalence classes there is no way to avoid having side-to-side walks. This is essentially the reason why the standard kangaroo method cannot be used with equivalence classes.

An important issue is that there is a danger of small cycles in the walks. This phenomena was noted by Gallant, Lambert and Vanstone [11] and Wiener and Zuccherato [29]. This can cause the pseudorandom walks to never reach a distinguished point. A method to get around this problem is "collapsing the cycle" which can be found in Gallant, Lambert and Vanstone [11, Section 6]. A detailed analysis of these issues is given by Bos, Kleinjung and Lenstra [2].

It is natural to try to apply the Gaudry-Schost algorithm on equivalence classes to solve the DLP in an interval of size N.

3.1 The Gaudry-Schost Algorithm on Equivalence Classes

We only give a short sketch of the method, since our main result is a further improvement on the basic idea. Recall that we wish to solve $h = g^n$ where $-N/2 \leq n \leq N/2$. We assume that computing h^{-1} for any h in the group is much faster than a general group operation.

The natural approach is to perform random walks in sets of equivalence classes corresponding to the tame and wild sets of the standard Gaudry-Schost method. In other words, it is natural to make the following definition.

Definition 1. *Define the tame and wild sets by*

$$T = \{\{a, -a\} : a \in [-N/2, N/2]\},$$
$$W = \{\{n + a, -(n + a)\} : a \in [-N/2, N/2]\}.$$

Note that $|T| = 1 + N/2 \approx N/2$.

As before, our main focus is on $T \cap W$. When $n = 0$ we have $T = W$ and when n is large then $T \cap W$ is only about half the size of T. However, a subtlety which did not arise in the previous case appears: when $n > N/4$ and $a > N/4$ there is only one way an equivalence class $\{n+a, -(n+a)\}$ can arise, but when $|n|$ is small there can be two ways. Specifically, suppose $-N/4 < n < 0$, then the equivalence class $\{n+a, -(n+a)\}$ can arise from a and from $a' = -2n - a$ (for example, if $n = -N/8$ then $a = N/4$ and $a' = 0$ are such that $\{n+a, -(n+a)\} = \{n+a', -(n+a')\}$). This phenomena means that the Gaudry-Schost algorithm samples from the wild set in a *non-uniform* way and this means we cannot apply Theorem 1 to determine the expected running time of the algorithm. We explain these issues more precisely in the next section.

We do not give an analysis of the average case expected number of group operations for this algorithm. In the next section we make a further optimisation which leads to a better algorithm. A full analysis of the improved algorithm is then given.

4 The New Algorithm

We now give an algorithm for the discrete logarithm problem in an interval for groups with efficient inversion. As usual, let N, g and h be given such that $4 \mid N$, $h = g^n$ and $-N/2 \leq n \leq N/2$. The basic idea is to run the Gaudry-Schost algorithm on the set of equivalence classes. A further speedup is given by defining the wild set W to be, in some sense, smaller than the tame set.

Definition 2. *We define the tame and wild sets (as sets of equivalence classes) by*

$$T = \{\{a, -a\} : a \in [-N/2, N/2]\},$$
$$W = \{\{n+a, -(n+a)\} : a \in [-N/4, N/4]\}$$

where, as always, $[N_1, N_2] = \{a \in \mathbb{Z} : N_1 \leq a \leq N_2\}$.

The algorithm is then immediate. One samples from T and W using pseudo-random walks which are well-defined on equivalence classes. When a walk hits a distinguished point then we store the representative of the equivalence class, its discrete logarithm (or the discrete logarithm of the group element divided by h), and the 'type' of the walk. When the same equivalence class is reached by walks of both types then the discrete logarithm problem is solved.

To understand the algorithm it is necessary to consider a 'fundamental domain' for the sets. In other words, we consider sets which are in one-to-one correspondence with the set of equivalence classes. A fundamental domain for T is $\widetilde{T} = [0, N/2]$; it is clear that every pair $\{a, -a\} \in T$ corresponds to exactly one value $a \in [0, N/2]$. One choice of fundamental domain for W is $\{|n| + a \mid a \in [-N/4, N/4]\}$. However, to visualise $T \cap W$ we really want the fundamental domain for W to consist only of positive values, and this is not the case when $|n| < N/4$. Hence, when $|n| < N/4$, we note that the set W in in one-to-one correspondence with the multi-set

$$\begin{aligned}\widetilde{W} &= \{|n| + a : a \in [-|n|, N/4]\} \cup \{-(|n| + a) : a \in [-N/4, -|n|)\} \\ &= [0, |n| + N/4] + [0, N/4 - |n|).\end{aligned} \tag{1}$$

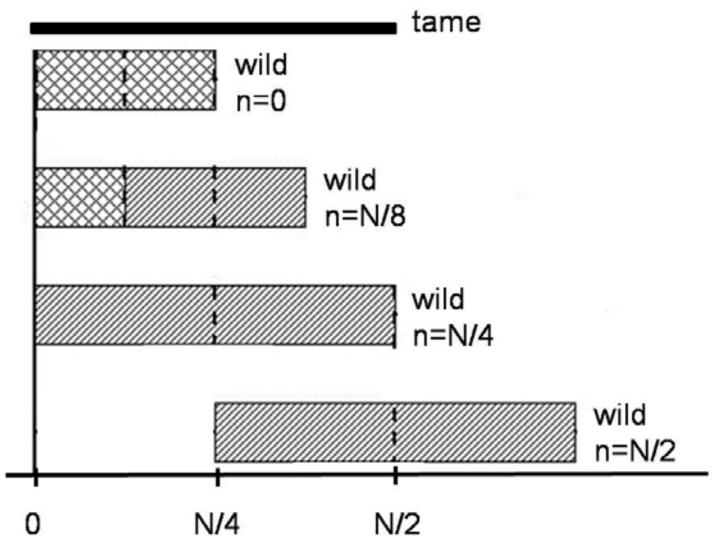


Fig. 2. The set $\widetilde{T}$ is pictured at the top of the diagram as a long black box. The sets $\widetilde{W}$ are given for the values $n = 0, N/8, N/4$ and $N/2$ where the diagonal lines denote single density and the cross hatching denotes double density (i.e., repetitions in the multi-set).

When $|n| < N/4$, sampling uniformly from W corresponds to sampling uniformly from the multi-set $\widetilde{W}$, which in turn corresponds to sampling $a \in [0, |n| + N/4]$ with probability $4/N$ for $0 \leq a < N/4 - |n|$ and probability $2/N$ for $N/4 - |n| \leq a \leq |n| + N/4$. We describe this as saying that there is a 'double density' of walks in the wild set.

To determine the complexity of the algorithm we need a generalisation of Theorem 1. This is a variant of the birthday paradox which applies to coloured balls and non-uniform probabilities. Such a result is proved in [10].

Theorem 3. *Let $R \in \mathbb{N}$ and $0 \leq A \leq R/2$. Suppose we have an unlimited number of balls of two colours, red and blue, and R urns. Suppose we alternately choose balls of each colour and put them in random urns. Red balls are assigned uniformly and independently to the urns. Blue balls are assigned to the urns independently with the following probabilities: urns $1 \leq u < A$ are used with probability $2/R$, urns $A \leq u \leq R-A$ are used with probability $1/R$, and urns $R-A < u \leq R$ are used with probability 0. Then the expected number of assignments that need to be made in total before we have an urn containing two balls of the same colour is $\sqrt{\pi R} + O(R^{1/4})$.*

We refer to [10] for the proof. However, it is relatively easy to see why the result should be true: The probability that a red ball and a blue ball fall in the same urn is

$$A\tfrac{1}{R}\tfrac{2}{R} + (R - 2A)\tfrac{1}{R}\tfrac{1}{R} + A\tfrac{1}{R}0 = \tfrac{1}{R}$$

which is exactly the same as the probability in the case where both red and blue balls are distributed uniformly. One significant difference from the standard Tame-Wild birthday paradox is that there is an increased chance of two or more blue balls being placed in the same urn (and this has the effect of lowering the probability of a collision among balls of different colour). Hence, Theorem 3 does not seem to be an immediate consequence of the results in [20,28].

Theorem 4. *If elements are sampled uniformly at random with replacement alternately from the sets T and W of Definition 2 and recorded, the expectation, over all problem instances, of the number of selections before a tame-wild collision is*

$$(5\sqrt{2}/4 - 1)\sqrt{\pi N} \approx 1.36\sqrt{N}.$$

Proof. Let $h = g^{xN}$ for $-1/2 \leq x \leq 1/2$. Due to symmetry we only need to look at the positive half of the interval of exponents. As we have seen, when $0 \leq x < 1/4$ we have $W \subseteq T$ and we are sampling in $T \cap W$ uniformly with the tame elements and non-uniformly with the wild elements. On the other hand, when $1/4 \leq x \leq 1/2$ then T and W are both sampled uniformly, but $T \cap W$ is now a proper subset of T and W in general. The analysis therefore breaks into two cases.

In the case $0 \leq x < 1/4$, by Theorem 3 (taking R to be the size of the fundemental domain for T, which is $N/2$), the expected number of group operations to get a collision is $\sqrt{\pi N/2}$.

In the case $1/4 \leq x \leq 1/2$ one sees that $|T \cap W| = 3N/4 - xN = N(3/4 - x)$ (here by $|T \cap W|$ we mean the number of equivalence classes in the intersection) and we are in a very similar situation to the proof of Theorem 2. We need to sample $\sqrt{\pi|T \cap W|}$ points in $T \cap W$ (half of them tame and half wild). Since $|T| = |W| = N/2$ we expect to sample

$$\frac{|T|}{|T \cap W|}\sqrt{\pi|T \cap W|} = \frac{N/2}{N(3/4 - x)}\sqrt{\pi N(3/4 - x)} = \tfrac{1}{2}\sqrt{\pi N/(3/4 - x)}$$

group elements in total.

We now average over all problem instances to get an average case running time.

$$\tfrac{1}{2}\sqrt{\pi N/2} + 2\int_{1/4}^{1/2} \tfrac{1}{2}\sqrt{\pi N/(3/4 - x)} = \sqrt{\pi N}\left(\tfrac{5}{4}\sqrt{2} - 1\right). \qquad \square$$

This result suggests the following heuristic statement about the running time of the algorithm using pseudorandom walks. The value ϵ takes into account various undesirable properties of the pseudorandom walk, such as irregular probability distributions at the boundaries of the regions and detecting and escaping from fruitless cycles.

Heuristic 2. *Our algorithm to solve the DLP in an interval of size N in a group with fast inversion has everage case expected running time of approximately $1.36(1 + \epsilon)\sqrt{N} + 1/\theta$ group operations for some small $\epsilon > 0$.*

As mentioned earlier, we believe that ϵ can be taken to be less than 0.1 and our experimental results suggest this is reasonable.

This is a significant improvement on the standard Gaudry-Schost algorithm (Heuristic 2) and the Improved Pollard kangaroo method with heuristic running time $1.71\sqrt{N}$ group operations [9].

5 Experimental Results

We implemented the Improved Gaudry-Schost algorithm using equivalence classes for solving the DLP in an interval using the software package Magma. The group used was the group of points on the following elliptic curve

$$E : y^2 = x^3 + 40x + 1 \text{ over } \mathbb{F}_p$$

where $p = 3645540875029913$. The group of points has cardinality

$$\#E(\mathbb{F}_p) = 3645540854261153 > 2^{51}.$$

We picked various interval sizes and ran a number of experiments on those intervals. Each experiment involved choosing uniformly at random $-N/2 \leq n \leq N/2$ and solving the DLP for $Q = [n]P$. We counted the number of group operations performed and averaged this over the total number of trials. Walks were not permitted to start within a distance $m\sqrt{2/3\pi\theta}$ from the edge of any of the sets (this is roughly half the size of the expected maximum distance travelled by a walk).

The average number of group operations performed for the different experiments are given in Table 1.

To detect small cycles we stored the previous 30 group elements in the walk in the case $N \approx 2^{34}$ (respectively, 30, 45 group elements for $N \approx 2^{40}, 2^{48}$). Each new step in the walk was compared with the previous 30 (respectively, 35, 45) group elements visited. If this group element had already been visited then the walk is in a cycle. We used a deterministic method to jump out of the cycle (using a jump of distinct length from the other jumps used in the pseudorandom walk) so that the pseudorandom walk as a whole remained deterministic. The cost of searching the list of previous group elements is not included in our experimental results, but our count of group operations does include the "wasted" steps from being in a cycle.

We terminated walks which ran for $5/\theta$ steps without arriving at a distinguished point (the usual recommendation is $20/\theta$ steps; see [22]). This will give us a slightly worse running time than optimal.

There is plenty of room for improvement in these experimental results. First, techniques like those in [2] to handle cycles should lead to improved running times (though note that we cannot use doublings/squarings when working in an interval). Second, the relationship between the values of m and θ is probably not optimal. Third, one might get better results by not running the same number of tame walks as wild walks or by slightly changing the sizes of the tame and wild regions.

Table 1. Average number of group operations performed by our algorithm for different values of N

	# of Experiments	Improved GS on equivalence classes
Experiment 1 $N \approx 2^{34}$ $m = 2^8$ $\theta = 2^{-5}$	1000	$1.49\sqrt{N}$
Experiment 2 $N \approx 2^{40}$ $m = 2^{11}$ $\theta = 2^{-5}$	300	$1.47\sqrt{N}$
Experiment 3 $N \approx 2^{48}$ $m = 2^{14.5}$ $\theta = 2^{-6}$	50	$1.46\sqrt{N}$

6 Conclusion

We have presented the first algorithm to exploit equivalence classes for the discrete logarithm problem in an interval. Our algorithm can be applied in groups where we have fast inversion such as in the group of points on an elliptic curve. The average expected running time of our algorithm is close to $1.36\sqrt{N}$ group operations. Our practical experiments confirm that we can achieve a significant improvement over previous methods.

Acknowledgements

We thank the anonymous referees of PKC 2010 and ASIACRYPT 2009 for a number of very helpful comments.

References

1. Boneh, D., Goh, E.J., Nissim, K.: Evaluating 2-DNF formulas on ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)
2. Bos, J.W., Kleinjung, T., Lenstra, A.K.: On the use of the negation map in the Pollard Rho method. (preprint, 2010)
3. 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)

4. Cofman, E.G., Flajolet, P., Flatto, L., Hofri, M.: The Maximum of a Random Walk and its Application to Rectangle Packing. Technical report, INRIA (1997)
5. Cohen, H., Frey, G.: Handbook of Elliptic and Hyperelliptic Curve Cryptography. In: Discrete Mathematics and its Applications. Chapman & Hall/CRC, Boca Raton (2005)
6. Duursma, I.M., Gaudry, P., Morain, F.: Speeding up the discrete log computation on curves with automorphisms. In: Lam, K.-Y., Okamoto, E., Xing, C. (eds.) ASIACRYPT 1999. LNCS, vol. 1716, pp. 103–121. Springer, Heidelberg (1999)
7. Galbraith, S.D., Lin, X., Scott, M.: Endomorphisms for Faster Elliptic Curve Cryptography on a Large Class of Curves. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 518–535. Springer, Heidelberg (2009)
8. Galbraith, S.D., Ruprai, R.S.: An improvement to the Gaudry-Schost algorithm for multidimensional discrete logarithm problems. In: Parker, M.G. (ed.) IMACC 2009. LNCS, vol. 5921, pp. 368–382. Springer, Heidelberg (2009)
9. Galbraith, S.D., Pollard, J.M., Ruprai, R.S.: Improving kangaroo and Gaudry-Schost methods for solving the DLP in an interval (in preparation) (2010)
10. Galbraith, S.D., Holmes, M.: A non-uniform birthday problem with applications to discrete logarithms (in preparation) (2010)
11. Gallant, R., Lambert, R., Vanstone, S.: Improving the Parallelized Pollard Lambda Search on Binary Anomalous Curves. Mathematics of Computation 69, 1699–1705 (2000)
12. Gallant, R.P., Lambert, R.J., Vanstone, S.A.: Faster Point Multiplication on Elliptic Curves with Efficient Endomorphisms. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 190–200. Springer, Heidelberg (2001)
13. Gaudry, P., Harley, R.: Counting Points on Hyperelliptic Curves over Finite Fields. In: Bosma, W. (ed.) ANTS 2000. LNCS, vol. 1838, pp. 313–332. Springer, Heidelberg (2000)
14. Gaudry, P., Schost, E.: A low-memory parallel version of Matsuo, Chao and Tsujii's algorithm. In: Buell, D.A. (ed.) ANTS 2004. LNCS, vol. 3076, pp. 208–222. Springer, Heidelberg (2004)
15. Gennaro, R.: An Improved Pseudo-random Generator Based on Discrete Log. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 469–481. Springer, Heidelberg (2000)
16. Gopalakrishnan, K., Thériault, N., Yao, C.Z.: Solving Discrete Logarithms from Partial Knowledge of the Key. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 224–237. Springer, Heidelberg (2007)
17. Jao, D., Yoshida, K.: Boneh-Boyen signatures and the Strong Diffie-Hellman problem. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 1–16. Springer, Heidelberg (2009)
18. Lim, C.H., Lee, P.J.: A Key Recovery Attack on Discrete Log-based Schemes Using a Prime Order Subgroup. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 249–263. Springer, Heidelberg (1997)
19. Montenegro, R., Tetali, P.: How long does it take to catch a wild kangaroo? In: 41st ACM Symposium on Theory of Computing (2009)
20. Nishimura, K., Sibuya, M.: Probability to meet in the middle. Journal of Cryptology 2, 13–22 (1990)
21. 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)
22. van Oorschot, P.C., Wiener, M.J.: Parallel collision Search with Cryptanalytic Applications. Journal of Cryptology 12, 1–28 (1999)

23. Patel, S., Sundaram, G.: An Efficient Discrete Log Pseudo Random Generator. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 304–317. Springer, Heidelberg (1998)
24. Pollard, J.M.: Monte Carlo Methods for Index Computation mod p. Mathematics of Computation 32(143), 918–924 (1978)
25. Pollard, J.M.: Three kangaroos are better than two! Private Communication (2009)
26. Pollard, J.M.: Kangaroos, Monopoly and Discrete Logarithms. Journal of Cryptology 13, 437–447 (2000)
27. Ruprai, R.S.: An improvement to the Gaudry-Schost algorithm for multidimensional discrete logarithm problems and applications. PhD Thesis, Royal Holloway, University of London (2010)
28. Selivanov, B.I.: On waiting time in the scheme of random allocation of coloured particles. Discrete Math. Appl. 5(1), 73–82 (1995)
29. Wiener, M.J., Zuccerato, R.J.: Faster Attacks on Elliptic Curve Cryptosystems. In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 190–200. Springer, Heidelberg (1999)

A Background on the Pollard Kangaroo Method

We first briefly recall the Pollard kangaroo method using distinguished points as described by van Oorschot and Wiener [22] and Pollard [26]. To fix notation: We are given g, h and N and asked to find $0 \leq n \leq N$ such that $h = g^n$.

As with the rho method, the kangaroo method relies on a pseudorandom walk, however steps in the kangaroo walk correspond to known small increments in the exponent (in other words, kangaroos make small jumps). The tame kangaroo starts in the middle of the interval (i.e., at $g^{N/2}$) and jumps towards the right. The wild kangaroo starts at the group element h and jumps to the right using the same pseudorandom walk. On a serial computer one alternately jumps the tame and wild kangaroos. Every now and then a tame or wild kangaroo lands on a distinguished group element u and stores it in a sorted list, binary tree or hash table together with its discrete logarithm (if the kangaroo is tame) or the discrete logarithm of uh^{-1} (if the kangaroo is wild). Once the same group element is visited twice by different kangaroos the DLP is solved.

The kangaroo method is not analysed using the birthday paradox but using the mean step size m of the pseudorandom walks. Once the rear kangaroo reaches the starting point of the front kangaroo it is jumping over a region where roughly one in m group elements have been visited by the front kangaroo. Hence, there is a roughly $1/m$ probability at each step that the back kangaroo lands on a footprint of the front kangaroo. Therefore, the walks collide after an expected m steps.

One obtains the heuristic average case expected running time of approximately $2\sqrt{N}$ group operations as follows: Choose $m = \sqrt{N}/2$. The rear kangaroo is, on average, distance $N/4$ from the front kangaroo. The rear kangaroo therefore performs $N/(4m)$ jumps to reach the starting point of the front kangaroo, followed by m more steps until the walks collide (and then a small number more steps until a distinguished point is hit). Since there are two kangaroos in action the total running time is roughly $2(N/(4m) + m) = 2\sqrt{N}$ group operations.

B Two-Dimensional Problems

One can consider the multi-dimensional DLP: Given $g_1, \ldots, g_d, h$ and bounds $N_1, \ldots, N_d$, to compute $n_1, \ldots, n_d \in \mathbb{Z}$ such that $h = g_1^{n_1} \cdots g_d^{n_d}$ and $|n_i| \leq N_i$ for $1 \leq i \leq d$. We call the integer d the dimension. The size of the solution region is $N = \prod_{i=1}^{d}(2N_i + 1)$. This problem arises in a number of applications. For example, Gaudry and Schost [14] use algorithms for the 2-dimensional DLP in point counting on hyperelliptic curves of genus 2.

The 2-dimensional DLP also arises if one tries to analyse the security of elliptic curve cryptography using the Gallant-Lambert-Vanstone (GLV) method [12]. In this method one has an efficiently computable group homomorphism ψ and one computes nP for $P \in E(\mathbb{F}_q)$ and $n \in \mathbb{N}$ as $n_1P + n_2\psi(P)$ where $|n_1|, |n_2| \approx \sqrt{n}$. There is an algorithm to compute the pair (n_1, n_2) from n, but a natural trick is to choose (n_1, n_2) directly. It is tempting to choose $|n_1|$ and $|n_2|$ to be a little smaller than $\sqrt{n}$, and the extent to which this can be done without losing security depends on the difficulty of the 2-dimensional DLP. Gaudry and Schost do not give a precise figure for the running time of this algorithm but we have the following heuristic under the usual assumptions (see [8] for further details of this result and an improvement of the constant from 2.43 to 2.36).

Heuristic 3. *The Gaudry and Schost [14] algorithm solves the 2-dimensional DLP in as above in* $2.43(1 + \epsilon)\sqrt{N} + 1/\theta$ *group operations for small* $\epsilon > 0$.

B.1 Solving Using Equivalence Classes

In groups with efficiently computable inverse (such as the groups of interest to Gaudry and Schost and the GLV method) one can consider equivalence classes as we did in the 1-dimensional case. To be precise, let

$$T = \{\{(x, y), (-x, -y)\} : x, y \in \mathbb{Z}\ ,\ -N_1 \leq x \leq N_1, -N_2 \leq y \leq N_2\}$$

be the set of equivalence classes of points in a box of area $N = (2N_1+1)(2N_2+1)$ centered at 0. For (n_1, n_2) such that $-N_i \leq n_i \leq N_i$ $(i \in \{1, 2\})$ we consider the set

$$W = \left\{ \{(n_1 + u_1, n_2 + u_2), (-(n_1 + u_1), -(n_2 + u_2))\} : \begin{array}{r} u_1, u_2 \in \mathbb{Z}, \\ -N_1/2 \leq u_1 \leq N_1/2, \\ -N_2/2 \leq u_2 \leq N_2/2 \end{array} \right\}.$$

To analyse the algorithm again requires visualising the sets via a 'fundamental domain'. Since the map $(x, y) \mapsto (-x, -y)$ is rotation by 180 degrees, a natural fundamental domain is the halfplane $y \geq -x$. One therefore defines the fundamental domain $\widetilde{T}$ for T to be

$$\widetilde{T} = \{(x, y) : -N_1 \leq x \leq N_1, -x \leq y \leq N_2\}.$$

Note that $|\widetilde{T}| \approx 2N_1N_2$. A fundamental domain for $\widetilde{W}$ which is contained in $\widetilde{T}$ is easily defined, but note that, as in Section 4, this can be a multi-set and we can again be in the case of non-uniform sampling of $\widetilde{W}$. Indeed, when $0 \le n_1 < N_1/2$ and $0 \le n_2 < N_2/2$ this is the case and the region which has 'double density' has area $A = \frac{1}{2}(N_1 - 2n_1)(N_2 - 2n_2)$. When $n_1 \ge N_1/2$ or $n_2 \ge N_2/2$ then the distribution on $\widetilde{W}$ is uniform but $\widetilde{W}$ is not usually contained in $\widetilde{T}$ any more. In these cases $|\widetilde{T} \cap \widetilde{W}|$ varies between $N_1N_2 = \frac{1}{2}|\widetilde{T}|$ and $\frac{1}{4}N_1N_2 = \frac{1}{8}|\widetilde{T}|$.

We considered an algorithm which chooses elements from T and W uniformly at random, selecting elements with a ratio of 2 : 1 (i.e., twice as many tame walks as wild walks, since the tame set is at least twice as big as the wild set).

Our rough calculations suggest that the algorithm (when using a truly random walk) should require fewer than $2.01\sqrt{N}$ group operations. This is a significant speedup over the algorithm of Gaudry and Schost for the cases of practical interest. It remains an open problem to find optimal parameters for this algorithm, to analyse its complexity precisely, and to give experimental results which show how closely one can get to the idealised theoretical analysis.

B.2 Larger Equivalence Classes in the GLV Method

We now assume that $N_1 = N_2$ and that the 2-dimensional DLP of interest is $Q = n_1P + n_2\psi(P)$ with $|n_1|, |n_2| \le N_1$. Again write $N = (2N_1 + 1)(2N_2 + 1)$. Since one knows the logarithm of $\psi(P)$ to the base P it is sufficient to compute n_1 and n_2.

Frequently with the GLV method [12] the homomorphism ψ satisfies $\psi^2 = -1$. This happens, for example, with the standard curve $y^2 = x^3 + Ax$ over $\mathbb{F}_p$ with $\psi(x, y) = (-x, iy)$. It also holds for the homomorphisms used by Galbraith, Lin and Scott [7]. In this setting one can consider the equivalence classes

$$\{Q, -Q, \psi(Q), -\psi(Q)\}$$

of size 4. If $Q = n_1P + n_2\psi(P)$ then these 4 points correspond to the pairs of exponents

$$\{(n_1, n_2), (-n_1, -n_2), (-n_2, n_1), (n_2, -n_1)\}$$

and so action by ψ corresponds to rotation by 90 degrees.

It is natural to apply the Gaudry-Schost algorithm on these equivalence classes. We take the sets T and W analogous to those in Section B.1. Finding a suitable fundamental domain for the symmetry under rotation is not hard (for example take $\{(x, y) : 0 \le x, 0 \le y\}$). One now finds that some regions of the wild set can have quadruple density (as well as single and double density).

Again, it remains an open problem to determine the optimal algorithm for this problem and to estimate its complexity. A very rough calculation suggests that there is an algorithm for this problem (using a truly random walk) which requires fewer than $1.11\sqrt{N}$ group operations.

Functional Encryption for Inner Product: Achieving Constant-Size Ciphertexts with Adaptive Security or Support for Negation

Nuttapong Attrapadung[1] and Benoît Libert[2,⋆]

[1] Research Center for Information Security, AIST, Japan
[2] Université catholique de Louvain, Crypto Group, Belgium

Abstract. In functional encryption (FE) schemes, ciphertexts and private keys are associated with attributes and decryption is possible whenever key and ciphertext attributes are suitably related. It is known that expressive realizations can be obtained from a simple FE flavor called inner product encryption (IPE), where decryption is allowed whenever ciphertext and key attributes form orthogonal vectors. In this paper, we construct (non-anonymous) IPE systems with *constant-size* ciphertexts for the zero *and* non-zero evaluations of inner products. These schemes respectively imply an adaptively secure identity-based broadcast encryption scheme and an identity-based revocation mechanism that both feature short ciphertexts and rely on simple assumptions in prime order groups. We also introduce the notion of *negated spatial encryption*, which subsumes non-zero-mode IPE and can be seen as the revocation analogue of the spatial encryption primitive of Boneh and Hamburg.

Keywords: Functional encryption, identity-based broadcast encryption, revocation, efficiency.

1 Introduction

Ordinary encryption schemes usually provide coarse-grained access control since, given a ciphertext, only the holder of the private key can obtain the plaintext. In many applications such as distributed file systems, the need for fine-grained and more complex access control policies frequently arises. To address these concerns, several kinds of *functional public key encryption* schemes have been studied.

Functional encryption can be seen as a generalization of identity-based encryption (IBE) [24,8]. In IBE schemes, the receiver's ability to decrypt is merely contingent on his knowledge of a private key associated with an identity that matches a string chosen by the sender. In contrast, functional encryption (FE) systems make it possible to decrypt using a private key $\mathsf{sk}_{\boldsymbol{x}}$ corresponding to a set $\boldsymbol{x}$ of atomic elements, called *attributes*, that is suitably related – according to some well-defined relation R – to another attribute set $\boldsymbol{y}$ specified by the sender.

⋆ This author acknowledges the Belgian National Fund for Scientific Research (F.R.S.-F.N.R.S.) for their support and the BCRYPT Interuniversity Attraction Pole.

The goal of this paper is to describe new (pairing-based) functional encryption constructions providing short ciphertexts (ideally, their length should not depend on the size of attribute sets) while providing security against adaptive adversaries or supporting negation (e.g. decryption should be disallowed to holders of private keys $\mathsf{sk}_{\boldsymbol{x}}$ for which $R(\boldsymbol{x}, \boldsymbol{y}) = 1$).

Related Work. The first flavor of functional encryption traces back to the work of Sahai and Waters [22] that was subsequently extended in [16,21]. Their concept, called *attribute-based encryption* (ABE), allows a sender to encrypt data under a set of attributes ω while an authority generates private keys for access control policies $\mathcal{T}$. Decryption rights are granted to anyone holding a private key for a policy $\mathcal{T}$ such that $\mathcal{T}(\omega) = 1$. Identity-based broadcast encryption (IBBE) [2,23,13,9] and revocation (IBR) [19] schemes can also be thought of as functional encryption systems where ciphertexts are encrypted for a set of identities $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_n\}$: in IBBE (resp. IBR) systems, decryption requires to hold a private key $\mathsf{sk}_{\mathsf{ID}}$ for which $\mathsf{ID} \in S$ (resp. $\mathsf{ID} \notin S$).

The above kinds of functional encryption systems are only *payload hiding* in that they keep encrypted messages back from unauthorized parties but ciphertexts do not hide their underlying attribute set. *Predicate encryption* schemes [10,18,26,25] additionally provide *anonymity* as ciphertexts also conceal the attribute set they are associated with, which enables [7,1] efficient searches over encrypted data. In [18], Katz, Sahai and Waters devised a predicate encryption scheme for inner products: a ciphertext encrypted for the attribute vector $\vec{Y}$ can be opened by any key $\mathsf{sk}_{\vec{X}}$ such that $\vec{X} \cdot \vec{Y} = 0$. As shown in [18], inner product encryption (IPE) suffices to give functional encryption for a number of relations corresponding to the evaluation of polynomials or CNF/DNF formulae.

Our Contributions. While quite useful, the IPE scheme of [18] strives to anonymize ciphertexts, which makes it difficult to break through the linear complexity barrier (in the vector length n) in terms of ciphertext size. It indeed seems very hard to avoid such a dependency as long as anonymity is required: for instance, anonymous FE constructions [10,17] suffer from the same overhead. A similar problem appears in the context of broadcast encryption, where the only known scheme [3] that conceals the receiver set also has $O(n)$-size ciphertexts.

This paper focuses on applications of IPE schemes, such as identity-based broadcast encryption and revocation systems, where the anonymity property is not fundamental. Assuming public ciphertext attributes rather than anonymity may be useful in other contexts. For instance, suppose that a number of ciphertexts are stored with varying attributes $\boldsymbol{y}$ on a server and we want to decrypt only those for which $R(\boldsymbol{x}, \boldsymbol{y}) = 1$. Anonymous ciphertexts require to decrypt all of them whereas public attributes $\boldsymbol{y}$ make it possible to test whether $R(\boldsymbol{x}, \boldsymbol{y})$ (which is usually faster than decrypting) and only decrypt appropriate ones.

At the expense of sacrificing anonymity, we thus describe IPE schemes where the ciphertext overhead reduces to $O(1)$ as long as the description of the ciphertext attribute vector is not considered as being part of the ciphertext, which is a common assumption in the broadcast encryption/revocation applications.

In addition, the number of pairing evaluations to decrypt is also constant, which significantly improves upon $O(n)$, since pairings calculations still remain costly.

Our first IPE system achieves adaptive security, as opposed to the selective model, used in [18], where the adversary has to choose the target ciphertext vector $\vec{Y}$ upfront. To acquire adaptive security, we basically utilize the method used in the Waters' fully secure IBE [27], albeit we also have to introduce a new trick called "n-equation technique" so as to deal with the richer structure of IPE. Our system directly yields the first adaptively secure identity-based broadcast encryption scheme with constant-size ciphertexts in the standard model. Previous IBBE with $O(1)$-size ciphertexts were either only selective-ID secure [2,13,9,23] or in the random oracle model [15]. Among IBBE systems featuring compact ciphertexts (including selective-ID secure ones), ours is also the first one relying on simple assumptions (*i.e.*, no q-type assumption) in prime order groups.

It is worth mentioning that techniques developed by Lewko and Waters [20] can be applied to the construction of Boneh and Hamburg [9] to give fully secure IBBE with short ciphertexts in composite order groups. However, it was not previously known how to obtain such a scheme in prime order groups (at least without relying on the absence of computable isomorphism in asymmetric pairing configurations). Indeed, despite recent progress [14], there is still no black-box way to translate pairing-based cryptosystems from composite to prime order groups. In particular, Freeman's framework [14] does not apply to [20].

Our second contribution is an IPE system for non-zero inner products: ciphertexts encrypted for vector $\vec{Y}$ can only be decrypted using $\mathsf{sk}_{\vec{X}}$ if $\vec{X} \cdot \vec{Y} \neq 0$, which – without retaining anonymity – solves a question left open by Katz, Sahai and Waters [18][Section 5.4]. The scheme implies the first identity-based revocation (IBR) mechanism [19] with $O(1)$-size ciphertexts. Like the schemes of Lewko, Sahai and Waters [19], its security is analyzed in a non-adaptive model where the adversary has to choose which users to corrupt at the outset of the game[1]. In comparison with [19] where ciphertexts grow linearly with the number of revoked users and public/private keys have constant size, our basic IBR construction performs in the dual way since key sizes depend on the maximal number of revoked users. Depending on the application, one may prefer one scheme over the other one. We actually show how to generalize both implementations and obtain a tradeoff between ciphertext and key sizes (and without assuming a maximal number of revoked users): the second scheme of [19] and ours can be seen as lying at opposite extremities of the spectrum.

On a theoretical side, our non-zero IPE realization turns out to be a particular case of a more general primitive, that we call *negated spatial encryption*, which we define as a negated mode for the spatial encryption primitive of Boneh and Hamburg [9]. Namely, keys correspond to subspaces and can decrypt ciphertexts encrypted under points that lie *outside* the subspace. This generalized primitive turns out to be non-trivial to implement and we had to use a fully generalized

[1] We indeed work in a slightly stronger model, called *co-selective-ID*, where the adversary chooses which parties to corrupt at the beginning – before seeing the public key – but is not required to announce the target revoked set until the challenge phase.

form of our new "n-equation" technique. The proposed scheme is proven secure under a non-standard assumption defined in [19].

Our Techniques. The core technique of our *non-zero* IPE scheme will be used throughout the paper, including in our adaptively secure *zero* IPE scheme. This can be viewed analogously to fact that Waters' fully secure IBE [27] uses the revocation technique of [19]. Our non-zero IPE also builds on [19]. However, the fact that non-zero IPE has much richer structure than revocation scheme and the pursued goal of achieving constant ciphertext size together prevent us from using their techniques directly. To describe the difficulties that arise, we first outline the Lewko-Sahai-Waters revocation scheme in its simplified form where security proof is not provided and where only one user is revoked.

Construction 1. (A simplified revocation scheme)

▶ Setup: lets $(\mathbb{G}, \mathbb{G}_T)$ be bilinear groups of prime order p and picks $g \xleftarrow{\$} \mathbb{G}$, $\alpha, \alpha_1, \alpha_2 \xleftarrow{\$} \mathbb{Z}_p$. The public key is $\big(g, g^{\alpha_1}, g^{\alpha_2}, e(g,g)^{\alpha}\big)$. The master key is g^{α}.

▶ KeyGen: chooses $t \xleftarrow{\$} \mathbb{Z}_p$ and outputs a private key for an identity $\mathsf{ID} \in \mathbb{Z}_p$ as $(K_0 = g^t,\ K_1 = g^{\alpha+\alpha_1 t},\ K_2 = g^{t(\alpha_1 \mathsf{ID}+\alpha_2)})$.

▶ $\mathsf{Encrypt}$: encrypts M and specifies a revoked ID' by choosing $s \xleftarrow{\$} \mathbb{Z}_p$ and computing $(E_0 = \mathsf{M} \cdot e(g,g)^{\alpha s},\ E_1 = g^{s(\alpha_1 \mathsf{ID}'+\alpha_2)},\ E_2 = g^s)$.

▶ $\mathsf{Decrypt}$: decryption computes $e(K_2, E_2)^{\frac{1}{\mathsf{ID}-\mathsf{ID}'}} e(E_1, K_0)^{-\frac{1}{\mathsf{ID}-\mathsf{ID}'}} = e(g,g)^{\alpha_1 ts}$ if $\mathsf{ID} \neq \mathsf{ID}'$. It then computes $e(g,g)^{\alpha s}$ as $e(K_1, E_2)/e(g,g)^{\alpha_1 ts} = e(g,g)^{\alpha s}$.

The scheme can be explained by viewing a key and a ciphertext as forming a linear system of 2 equations in the exponent of $e(g,g)$ with variables $\alpha_1 ts, \alpha_2 ts$.

$$M_{\mathsf{ID},\mathsf{ID}'} \begin{pmatrix} \alpha_1 ts \\ \alpha_2 ts \end{pmatrix} := \begin{pmatrix} \mathsf{ID} & 1 \\ \mathsf{ID}' & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 ts \\ \alpha_2 ts \end{pmatrix} = \begin{pmatrix} \log(e(K_2, E_2)) \\ \log(e(E_1, K_0)) \end{pmatrix}.$$

Computing $e(g,g)^{\alpha_1 ts}$ amounts to solve the system, which is possible when $\det(M_{ID,\mathsf{ID}'}) \neq 0$ (and thus $\mathsf{ID} \neq \mathsf{ID}'$, as required). In particular, decryption computes a linear combination (in the exponent) with coefficients from the first row of $M^{-1}_{\mathsf{ID},\mathsf{ID}'}$ which is $(\frac{1}{\mathsf{ID}-\mathsf{ID}'}, \frac{-1}{\mathsf{ID}-\mathsf{ID}'})$. In [19], this is called "2-equation technique". The scheme is extended to n-dimension, *i.e.,* the revocation of n users $\{\mathsf{ID}'_1, \dots, \mathsf{ID}'_n\}$, by utilizing n local independent systems of two equations

$$M_{\mathsf{ID},\mathsf{ID}'_j}\Big(\alpha_1 ts_j,\ \alpha_2 ts_j\Big)^{\top} = \Big(\log(e(K_2, E_{2,j})),\ \log(e(E_{1,j}, K_0))\Big)^{\top} \text{ for } j \in [1, n],$$

that yield $2n$ ciphertext components $(E_{1,j}, E_{2,j})$, each one of which corresponds to a share s_j of s such that $s = \sum_1^n s_j$. The decryption at j-th system returns $e(g,g)^{\alpha_1 ts_i}$ if $\mathsf{ID} \neq \mathsf{ID}'_j$. Combining these results finally gives $e(g,g)^{\alpha_1 ts}$.

We aim at *constant-size* ciphertexts for non-zero IPE schemes of dimension n. When trying to use the 2-equation technique with n dimensions, the following difficulties arise. First, the "decryptability" condition $\vec{X} \cdot \vec{Y} \neq 0$ cannot be decomposed as simply as that of the revocation scheme, which is decomposable as the conjunction of $\mathsf{ID} \neq \mathsf{ID}'_j$ for $j \in [1, n]$. Second, the ciphertext size was $O(n)$.

Towards solving these, we introduce a technique called "n-equation technique". First, we utilize n secret exponents $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n)^\top$ and let α_1 function as the "master" exponent while $\alpha_2, \ldots, \alpha_n$ serve as the "perturbed" factors. Intuitively, we will set up a system of n linear equations of the form:

$$M_{\vec{X},\vec{Y}}(\alpha_1 ts, \ldots, \alpha_n ts)^\top = \big(\log(e(K_{i_1}, E_{j_1})), \ldots, \log(e(K_{i_n}, E_{j_n}))\big)^\top \tag{1}$$

where $\{K_{i_k}\}$ and $\{E_{j_k}\}$ are elements of $\mathbb{G}$ defined for a key for $\vec{X}$ and a ciphertext for $\vec{Y}$ respectively. At first, this generalized system seems to require linear-size ciphertexts $(E_{j_1}, \ldots, E_{j_n})$. A trick to resolve this is to reuse ciphertext elements throughout the system: we let $E_{j_k} = E_2 = g^s$ for $k \in [1, n-1]$. This effectively yields a constraint $M_{\vec{X},\vec{Y}} = \big(Q_{\vec{X}}^\top \quad R^\top\big)^\top$, where $Q_{\vec{X}}$ is a $(n-1) \times n$ matrix parameterized only by $\vec{X}$ and R is a $1 \times n$ matrix. The remaining problem is then to choose $M_{\vec{X},\vec{Y}}$ in such a way that the system has a solution if $\vec{X} \cdot \vec{Y} \neq 0$ (the decryptability condition). To this end, we define

$$M_{\vec{X},\vec{Y}} := \begin{pmatrix} -\frac{x_2}{x_1} & 1 & & & \\ -\frac{x_3}{x_1} & & 1 & & \\ \vdots & & & \ddots & \\ -\frac{x_n}{x_1} & & & & 1 \\ y_1 & y_2 & y_3 & \cdots & y_n \end{pmatrix}, \tag{2}$$

where it holds that $\det(M_{\vec{X},\vec{Y}}) = (-1)^{n+1} \vec{X} \cdot \vec{Y} / x_1$. By translating this conceptual view back into algorithms, we obtain a basic non-zero IPE scheme. From this, we propose two schemes for non-zero IPE: the first one is a special case of negated spatial encryption scheme in section 5.1, while the second one is proven secure under simple assumptions and given in section 5.2.

Organization. In the forthcoming sections, the syntax and the applications of functional encryption are explained in sections 2 and 3. We describe our zero mode IPE system in section 4. Our negated schemes are detailed in section 5.

2 Definitions

2.1 Syntax and Security Definition for Functional Encryption

Let $R : \Sigma_{\mathsf{k}} \times \Sigma_{\mathsf{e}} \to \{0, 1\}$ be a boolean function where Σ_{k} and Σ_{e} denote "key attribute" and "ciphertext attribute" spaces. A functional encryption (FE) scheme for R consists of the following algorithms.

- $\mathsf{Setup}(1^\lambda, des) \to (\mathsf{pk}, \mathsf{msk})$: takes as input a security parameter 1^λ and a scheme description des (which usually describes the dimension n), and outputs a master public key pk and a master secret key msk.
- $\mathsf{KeyGen}(\boldsymbol{x}, \mathsf{msk}) \to \mathsf{sk}_{\boldsymbol{x}}$: takes as input a key attribute $\boldsymbol{x} \in \Sigma_{\mathsf{k}}$ and the master key msk. It outputs a private decryption key $\mathsf{sk}_{\boldsymbol{x}}$.

- $\mathsf{Encrypt}(\boldsymbol{y}, \mathsf{M}, \mathsf{pk}) \rightarrow C$: takes as input a ciphertext attribute $\boldsymbol{y} \in \Sigma_{\mathsf{e}}$, a message $\mathsf{M} \in \mathcal{M}$, and public key pk. It outputs a ciphertext C.
- $\mathsf{Decrypt}(C, \boldsymbol{y}, \mathsf{sk}_{\boldsymbol{x}}, \boldsymbol{x}) \rightarrow \mathsf{M}$: given a ciphertext C with its attribute $\boldsymbol{y}$ and the decryption key $\mathsf{sk}_{\boldsymbol{x}}$ with its attribute $\boldsymbol{x}$, it outputs a message M or $\perp$.

We require the standard correctness of decryption, that is, for all λ, all $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^{\lambda})$, all $\boldsymbol{x} \in \Sigma_{\mathsf{k}}$, all $\mathsf{sk}_{\boldsymbol{x}} \leftarrow \mathsf{KeyGen}(\boldsymbol{x}, \mathsf{msk})$, and all $\boldsymbol{y} \in \Sigma_{\mathsf{e}}$,

- If $R(\boldsymbol{x}, \boldsymbol{y}) = 1$, then $\mathsf{Decrypt}(\mathsf{Encrypt}(\boldsymbol{y}, \mathsf{M}, \mathsf{pk}), \mathsf{sk}_{\boldsymbol{x}}) = \mathsf{M}$.
- If $R(\boldsymbol{x}, \boldsymbol{y}) = 0$, $\mathsf{Decrypt}(\mathsf{Encrypt}(\boldsymbol{y}, \mathsf{M}, \mathsf{pk}), \mathsf{sk}_{\boldsymbol{x}}) = \perp$ with probability nearly 1.

Terminology and Variants. We refer to any encryption primitive A that can be casted as a functional encryption by specifying its corresponding function $R^{\mathsf{A}} : \Sigma_{\mathsf{k}}^{\mathsf{A}} \times \Sigma_{\mathsf{e}}^{\mathsf{A}} \rightarrow \{0, 1\}$. For a FE primitive A, we can define two variants:

- **Dual Variant**, denoted by $\mathsf{Dual}(\mathsf{A})$, is defined by setting $\Sigma_{\mathsf{k}}^{\mathsf{Dual}(\mathsf{A})} := \Sigma_{\mathsf{e}}^{\mathsf{A}}$ and $\Sigma_{\mathsf{e}}^{\mathsf{Dual}(\mathsf{A})} := \Sigma_{\mathsf{k}}^{\mathsf{A}}$ and $R^{\mathsf{A}}(\boldsymbol{x}, \boldsymbol{y}) = R^{\mathsf{Dual}(\mathsf{A})}(\boldsymbol{y}, \boldsymbol{x})$. In a dual variant, the roles of key and ciphertext attributes are swapped from those of its original primitive.
- **Negated Variant**, denoted by $\mathsf{Neg}(\mathsf{A})$, is defined by using the same domains as A and setting $R^{\mathsf{Neg}(\mathsf{A})}(\boldsymbol{x}, \boldsymbol{y}) = 1 \Leftrightarrow R^{\mathsf{A}}(\boldsymbol{x}, \boldsymbol{y}) = 0$. The condition is thus the opposite of the original primitive.

Security Definition. A FE scheme for a function $R : \Sigma_{\mathsf{k}} \times \Sigma_{\mathsf{e}} \rightarrow \{0, 1\}$ is fully secure if no PPT adversary $\mathcal{A}$ has non-negligible advantage in this game.

Setup. The challenger runs $\mathsf{Setup}(\mathrm{n})$ and hands the public key pk to $\mathcal{A}$.

Query Phase 1. The challenger answers private key queries for $\boldsymbol{x} \in \Sigma_{\mathsf{k}}$ by returning $\mathsf{sk}_{\boldsymbol{x}} \leftarrow \mathsf{KeyGen}(\boldsymbol{x}, \mathsf{msk})$.

Challenge. $\mathcal{A}$ submits messages $\mathsf{M}_0, \mathsf{M}_1$ and a target ciphertext attribute vector $\boldsymbol{y}^{\star} \in \Sigma_{\mathsf{e}}$ such that $R(\boldsymbol{x}, \boldsymbol{y}^{\star}) = 0$ for all key attributes $\boldsymbol{x}$ that have been queried so far. The challenger then flips a bit $\beta \overset{\$}{\leftarrow} \{0, 1\}$ and computes the challenge ciphertext $C^{\star} \leftarrow \mathsf{Encrypt}(\boldsymbol{y}, \mathsf{M}_{\beta}, \mathsf{pk})$ which is given to $\mathcal{A}$.

Query Phase 2. The adversary is allowed to make further private key queries $\boldsymbol{x} \in \Sigma_{\mathsf{k}}$ under the same restriction as above, *i.e.,* $R(\boldsymbol{x}, \boldsymbol{y}^{\star}) = 0$.

Guess. The adversary $\mathcal{A}$ outputs a guess $\beta' \in \{0, 1\}$ and wins if $\beta' = \beta$. In the game, $\mathcal{A}$'s advantage is typically defined as $\mathbf{Adv}_{\mathcal{A}}(\lambda) = |\Pr[\beta = \beta'] - \frac{1}{2}|$.

(Co-)Selective Security. We also consider the notion of selective security [11,4], where $\mathcal{A}$ has to choose the challenge attribute $\boldsymbol{y}^{\star}$ before the setup phase, but can adaptively choose the key queries for $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q$. One can consider its "dual" notion where $\mathcal{A}$ must output the q key queries for attribute vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q$ before the setup phase, but can adaptively choose the target challenge attribute $\boldsymbol{y}^{\star}$. We refer to this scenario as the *co-selective* security model, which is useful in some applications such as revocation. By definition, both notions are incomparable in general and we do not know about their relation yet.

We shall show how one FE primitive can be obtained from another. The following useful lemma from [9] describes a sufficient criterion for implication.

Proposition 1 (Embedding Lemma [9]). *Consider encryption primitives* A, B *that can be casted as functional encryption for functions* $R^{\mathsf{A}}, R^{\mathsf{B}}$*, respectively. Suppose there exists efficient injective mappings* $f_{\mathsf{k}} : \Sigma_{\mathsf{k}}^{\mathsf{A}} \to \Sigma_{\mathsf{k}}^{\mathsf{B}}$ *and* $f_{\mathsf{e}} : \Sigma_{\mathsf{e}}^{\mathsf{A}} \to \Sigma_{\mathsf{e}}^{\mathsf{B}}$ *such that* $R^{\mathsf{B}}(f_{\mathsf{k}}(\boldsymbol{x}), f_{\mathsf{e}}(\boldsymbol{y})) = 1 \Leftrightarrow R^{\mathsf{A}}(\boldsymbol{x}, \boldsymbol{y}) = 1$*. Let* Π_{B} *be a construction for primitive* B*. We then construct* Π_{A} *for primitive* A *from* Π_{B} *by applying mappings* $f_{\mathsf{k}}, f_{\mathsf{e}}$ *to all key attributes and ciphertext attributes, respectively. More precisely, we use exactly the same setup algorithm and define key generation and encryption procedures as* $\Pi_{\mathsf{A}}.\mathsf{KeyGen}(x, \mathsf{msk}) := \Pi_{\mathsf{B}}.\mathsf{KeyGen}(f_{\mathsf{k}}(x), \mathsf{msk})$ *and* $\Pi_{\mathsf{A}}.\mathsf{Encrypt}(y, M, \mathsf{pk}) := \Pi_{\mathsf{B}}.\mathsf{Encrypt}(f_{\mathsf{e}}(y), M, \mathsf{pk})$*, respectively. Then, if* Π_{B} *is secure, so is* Π_{A}*. This holds for adaptive, selective, co-selective security models. We denote this primitive implication by* $\mathsf{B} \xrightarrow{f_{\mathsf{k}}, f_{\mathsf{e}}} \mathsf{A}$*.*

We immediately obtain the next corollary stating that the implication applies to the negated (resp. dual) variant with the same (resp. swapped) mappings.

Corollary 1. $\mathsf{B} \xrightarrow{f_{\mathsf{k}}, f_{\mathsf{e}}} \mathsf{A}$ *implies* $\mathsf{Dual}(\mathsf{B}) \xrightarrow{f_{\mathsf{e}}, f_{\mathsf{k}}} \mathsf{Dual}(\mathsf{A})$ *and* $\mathsf{Neg}(\mathsf{B}) \xrightarrow{f_{\mathsf{k}}, f_{\mathsf{e}}} \mathsf{Neg}(\mathsf{A})$*.*

2.2 Complexity Assumptions in Bilinear Groups

We consider groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order p with an efficiently computable map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G} \times \mathbb{G}$ and $a, b \in \mathbb{Z}$ and $e(g, h) \neq 1_{\mathbb{G}_T}$ whenever $g, h \neq 1_{\mathbb{G}}$. In these groups, we assume the hardness of the Decision Bilinear Diffie-Hellman and Decision Linear [5] problems.

Definition 1. *The* **Decision Bilinear Diffie-Hellman Problem** *(DBDH) in* $(\mathbb{G}, \mathbb{G}_T)$ *is, given elements* $(g, g^{\theta_1}, g^{\theta_2}, g^{\theta_3}, \eta) \in \mathbb{G}^4 \times \mathbb{G}_T$ *with* $\theta_1, \theta_2, \theta_3 \xleftarrow{\$} \mathbb{Z}_p$*, to decide whether* $\eta = e(g, g)^{\theta_1 \theta_2 \theta_3}$ *or* $\eta \in_R \mathbb{G}_T$*.*

Definition 2. *The* **Decision Linear Problem** *(DLIN) in* $\mathbb{G}$ *consists in, given a tuple* $(g, f, \nu, g^{\theta_1}, f^{\theta_2}, \eta) \in \mathbb{G}^6$ *with* $\theta_1, \theta_2 \xleftarrow{\$} \mathbb{Z}_p$ *and* $f, g, \nu \xleftarrow{\$} \mathbb{G}$*, deciding whether* $\eta = \nu^{\theta_1 + \theta_2}$ *or* $\nu \in_R \mathbb{G}$*.*

3 Functional Encryption Instances and Their Implications

3.1 Inner Product Encryption and Its Consequences

We underline the power of IPE by showing its implications in this section. Each primitive is defined by describing the corresponding boolean function R. We then show how to construct one primitive from another by explicitly describing attribute mappings. In this way, correctness and security are consequences of the embedding lemma. Basically, the approach follows exactly the same way as [18]. A new contribution is that we also consider the negated variant of primitives, which will be useful for non-zero polynomial evaluation and revocation schemes. The implication for negated variants follows from Corollary 1.

Inner Product. An inner product encryption (IPE) scheme over $\mathbb{Z}_p^n$, for some prime p, is defined as follows. Both attribute domains are $\Sigma_{\mathsf{k}}^{\mathsf{IPE}_n} = \Sigma_{\mathsf{e}}^{\mathsf{IPE}_n} = \mathbb{Z}_p^n$.

We consider two distinct IPE modes here. The first one is zero-mode IPE where $R^{\mathsf{ZIPE}_n}(\vec{X},\vec{Y}) = 1$ iff $\vec{X}\cdot\vec{Y} = 0$. The other one is its negated primitive, which we call the non-zero-mode IPE, where $R^{\mathsf{NIPE}_n}(\vec{X},\vec{Y}) = 1$ iff $\vec{X}\cdot\vec{Y} \neq 0$.

Polynomial Evaluation. Functional encryption for the zero evaluation of polynomials of degree $\leq n$ is defined as follows. The ciphertext and key attribute domains are defined as $\Sigma_{\mathsf{e}}^{\mathsf{ZPoly}_{\leq n}} = \mathbb{Z}_p$ and $\Sigma_{\mathsf{k}}^{\mathsf{ZPoly}_{\leq n}} = \{P \in \mathbb{Z}_p[x] \mid \deg(P) \leq n\}$, respectively. The relation is defined by $R^{\mathsf{ZPoly}_{\leq n}}(P,x) = 1$ iff $P(x) = 0$. The non-zero evaluation mode can be defined as its negated primitive $\mathsf{Neg}(\mathsf{ZPoly}_{\leq n})$.

Given an IPE scheme over $\mathbb{Z}_p^{n+1}$, one obtain a functional encryption system for polynomial evaluation via the following embedding. For the key attribute, we map the polynomial $P[X] = \rho_0 + \rho_1 X + \cdots + \rho_n X^n$ to $\vec{X}_p = (\rho_0,\ldots,\rho_n)$. Regarding ciphertext attributes, each element $w \in \mathbb{Z}_p$ is mapped onto a vector $\vec{Y}_w = (1, w, w^2, \ldots, w^n)$. Correctness and security hold since $P(w) = 0$ whenever $\vec{X}_p \cdot \vec{Y}_w = 0$. The non-zero evaluation case can be analogously derived from the non-zero-mode IPE using the same mappings, due to Corollary 1.

We can also consider other variants such as a scheme that supports multivariate polynomials and a dual variant, where the key attribute corresponds to a fixed point and the ciphertext attribute corresponds to a polynomial, as in [18].

OR, AND, DNF, CNF Formulae. We now consider a FE scheme for some boolean formulae that evaluate disjunctions, conjunctions, and their extensions to disjunctive or conjunctive normal forms. As an example, a functional encryption scheme for boolean formula $R^{\mathsf{OR}_{\leq n}} : \mathbb{Z}_N^{\leq n} \times \mathbb{Z}_N \to \{0,1\}$ can be defined by $R^{\mathsf{OR}_{\leq n}}((I_1,\ldots,I_k),z) \mapsto 1$ (for $k \leq n$) iff $(z = I_1)$ or $\cdots$ or $(z = I_k)$. This can be obtained from a functional encryption for the zero evaluation of a univariate polynomial of degree smaller than n by generating a private key for $f_{\mathsf{OR},I_1,\ldots,I_k}(z) = (z - I_1)\cdots(z - I_k)$, and letting senders encrypting to z.

Other fundamental cases can be considered similarly as in [18] and are shown below. In [18] only non-negated policies (the first three cases below and their extensions) were considered. Schemes supporting negated policies (the latter three cases below and their extensions) are introduced in this paper. The negated case can be implemented by IPE for non-zero evaluation. One can combine these cases to obtain DNF, CNF formulae. Below, $r \xleftarrow{\$} \mathbb{Z}_p$ is chosen by KeyGen.[2]

Policy	Implementation
$(z = I_1)$ or $(z = I_2)$	$f_{\mathsf{OR},I_1,I_2}(z) = (z - I_1)(z - I_2) = 0$
$(z_1 = I_1)$ or $(z_2 = I_2)$	$f_{\overline{\mathsf{OR}},I_1,I_2}(z_1,z_2) = (z_1 - I_1)(z_2 - I_2) = 0$
$(z_1 = I_1)$ and $(z_2 = I_2)$	$f_{\mathsf{AND},I_1,I_2}(z_1,z_2) = (z_1 - I_1)r + (z_2 - I_2) = 0$
$(z_1 \neq I_1)$ or $(z_2 \neq I_2)$	$f_{\mathsf{NOR},I_1,I_2}(z_1,z_2) = (z_1 - I_1)r + (z_2 - I_2) \neq 0$
$(z \neq I_1)$ and $(z \neq I_2)$	$f_{\mathsf{NAND},I_1,I_2}(z) = (z - I_1)(z - I_2) \neq 0$
$(z_1 \neq I_1)$ and $(z_2 \neq I_2)$	$f_{\overline{\mathsf{NAND}},I_1,I_2}(z_1,z_2) = (z_1 - I_1)(z_2 - I_2) \neq 0$

ID-based Broadcast Encryption and Revocation. Let $\mathcal{I}$ be an identity space. An ID-based broadcast encryption scheme (IBBE) for maximum n receivers

[2] As noted in [18], the AND (and NOR) case will not work in the adaptive security model since the information on r leaks.

per ciphertext is a functional encryption for $R^{\mathsf{IBBE}_{\leq n}} : \mathcal{I} \times 2^{\mathcal{I}} \to \{0,1\}$ defined by $R^{\mathsf{IBBE}_{\leq n}} : (\mathsf{ID}, S) \mapsto 1$ iff $\mathsf{ID} \in S$. An IBBE system can be constructed by a functional encryption for $R^{\mathsf{Dual}(\mathsf{OR}_{\leq n})}$. To encrypt a message for the receiver set $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_k\}$, one encrypts using the policy $(z = \mathsf{ID}_1)$ or $\cdots$ or $(z = \mathsf{ID}_k)$.

Likewise, identity-based revocation (IBR) [19] for at most n revocations per ciphertext can be casted as a negated IBBE, *i.e.*, $R^{\mathsf{IBR}_{\leq n}} : (\mathsf{ID}, R) \mapsto 1$ iff $\mathsf{ID} \notin R$.

3.2 Spatial Encryption

We now recall the concept of spatial encryption [9]. For a $n \times d$ matrix M of which elements are in $\mathbb{Z}_p$ and a vector $\vec{c} \in \mathbb{Z}_p^n$, we define its corresponding affine space as $\mathrm{Aff}(M, \vec{c}) = \{M\vec{w} + \vec{c} \mid \vec{w} \in \mathbb{Z}_p^d\}$. Let $\mathcal{V}_n \subseteq 2^{(\mathbb{Z}_p^n)}$ be the collection of all affine spaces inside $\mathbb{Z}_p^n$. That is, $\mathcal{V}_n = \{\mathrm{Aff}(M, \vec{c}) \mid M \in \mathbb{M}_{n\times d}, c \in \mathbb{Z}_p^n, d \leq n\}$, where $\mathbb{M}_{n\times d}$ is the set of all $n \times d$ matrices in $\mathbb{Z}_p$.

A spatial encryption in $\mathbb{Z}_p^n$ is a functional encryption for a relation $R^{\mathsf{Spatial}} : \mathcal{V}_n \times \mathbb{Z}_p^n \to \{0,1\}$ defined by $R^{\mathsf{Spatial}} : (V, \vec{y}) \mapsto 1$ iff $\vec{y} \in V$.

The notion of spatial encryption was motivated by Boneh and Hamburg [9]. It has many applications as it notably implies broadcast HIBE and multi-authority schemes. Nevertheless, its connection to inner-product encryption has not been investigated so far. In section 4.1, we prove that spatial encryption implies inner product encryption by providing a simple attribute mapping.

As a result of independent interest, we also consider the negated spatial encryption primitive (namely, FE that is defined by $R^{\mathsf{Neg}(\mathsf{Spatial})} : (V, \vec{y}) \mapsto 1$ iff $\vec{y} \notin V$) and provide a construction in section 5.1. From this scheme and Corollary 1 together with our implication result of zero-mode IPE from spatial encryption, we then obtain a non-zero-mode IPE construction.

4 Functional Encryption for Zero Inner-Product

4.1 Warm-Up: Selectively Secure Zero IPE from Spatial Encryption

We first show that spatial encryption implies zero IPE. For the key attribute, we map $\vec{X} = (x_1, \ldots, x_n)^\top \in \mathbb{Z}_p^n$ to an $(n-1)$-dimension affine space $V_{\vec{X}} = \mathrm{Aff}(M_{\vec{X}}, \vec{0}_n) = \{M_{\vec{X}}\vec{w} + \vec{0}_n \mid \vec{w} \in \mathbb{Z}_p^{n-1}\}$ with the matrix $M_{\vec{X}} \in \mathbb{Z}_p^{n\times(n-1)}$

$$M_{\vec{X}} = \begin{pmatrix} -\frac{x_2}{x_1}, -\frac{x_3}{x_1}, \cdots, -\frac{x_n}{x_1} \\ I_{n-1} \end{pmatrix}. \tag{3}$$

For any $\vec{Y} = (y_1, \ldots, y_n)^\top \in \mathbb{Z}_p^n$, we then have $\vec{X} \cdot \vec{Y} = 0 \;\Leftrightarrow\; \vec{Y} \in V_{\vec{X}}$ since $\vec{X} \cdot \vec{Y} = 0 \;\Leftrightarrow\; y_1 = y_2 \cdot (-\frac{x_2}{x_1}) + \cdots + y_n \cdot (-\frac{x_n}{x_1}) \Leftrightarrow \vec{Y} = M_{\vec{X}} \cdot (y_2, \ldots, y_n)^\top \;\Leftrightarrow\; \vec{Y} \in V_{\vec{X}}$. By the embedding lemma, we can therefore conclude its implication.

In [9], Boneh and Hamburg described a selectively secure construction of spatial encryption that achieves constant-size ciphertexts (by generalizing the Boneh-Boyen-Goh HIBE [6]). We thus immediately obtain a selectively secure zero IPE scheme with constant-size ciphertext as shown below.

We give some notations here. For a vector $\vec{a} = (a_1, \dots, a_n)^\top \in \mathbb{Z}_p^n$, we write $g^{\vec{a}}$ to denote $(g^{a_1}, \dots, g^{a_n})^\top$. Given $g^{\vec{a}}, \vec{z}$, one can easily compute $(g^{\vec{a}})^{\vec{z}} := g^{\langle \vec{a}, \vec{z} \rangle}$, where $\langle \vec{a}, \vec{z} \rangle$ denotes the inner product $\vec{a} \cdot \vec{z} = \vec{a}^\top \vec{z}$.

Construction 2. (Selectively secure zero IPE)

▶ $\mathsf{Setup}(1^\lambda, n)$: chooses bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$ with a generator $g \xleftarrow{\$} \mathbb{G}$. It chooses $\alpha, \alpha_0, \dots, \alpha_n \xleftarrow{\$} \mathbb{Z}_p$. Let $\vec{\alpha} = (\alpha_1, \dots, \alpha_n)$. The public key is $\mathsf{pk} = \big(g, g^{\alpha_0}, \vec{H} = g^{\vec{\alpha}}, Z = e(g,g)^\alpha\big)$. The master key is $\mathsf{msk} = g^\alpha$.

▶ $\mathsf{KeyGen}(\vec{X}, \mathsf{msk}, \mathsf{pk})$: chooses $t \xleftarrow{\$} \mathbb{Z}_p$ and parses $\vec{X}$ as $(x_1, \dots, x_n)$ and returns $\perp$ if $x_1 = 0$. It outputs the private key as $\mathsf{sk}_{\vec{X}} = (D_0, D_1, K_2, \dots, K_n)$ where

$$D_0 = g^t, \qquad D_1 = g^{\alpha + \alpha_0 t}, \qquad \{K_i = (g^{-\alpha_1 \frac{x_i}{x_1}} g^{\alpha_i})^t\}_{i=2,\dots,n}.$$

▶ $\mathsf{Encrypt}(\vec{Y}, \mathsf{pk})$: the encryption algorithm first picks $s \xleftarrow{\$} \mathbb{Z}_p$. It parses $\vec{Y}$ as $(y_1, \dots, y_n)$ and computes the ciphertext as

$$E_0 = \mathsf{M} \cdot e(g,g)^{\alpha s}, \qquad E_1 = (g^{\alpha_0} g^{\langle \vec{\alpha}, \vec{Y} \rangle})^s, \qquad E_2 = g^s.$$

▶ $\mathsf{Decrypt}(C, \vec{Y}, \mathsf{sk}_{\vec{X}}, \vec{X}, \mathsf{pk})$: to decrypt, the algorithm computes the message blinding factor as $\frac{e(D_1 K_2^{y_2} \cdots K_n^{y_n}, E_2)}{e(E_1, D_0)} = e(g,g)^{\alpha s}$.

The selective security of this scheme is a consequence of a result given in [9].

Theorem 1. *Construction 2 is selectively secure under the n-Decisional Bilinear Diffie-Hellman Exponent assumption* (see [9] for a description of the latter).

4.2 Adaptively Secure Zero IPE under Simple Assumptions

We extend the above selectively secure zero IPE to acquire adaptive security by applying the Waters' dual system method [27]. However, we have to use our "n-equation technique" as opposed to 2-equation technique used for IBE in [27]. The reason is that we have to deal with the difficulties arising from the richer structure of IPE and the aggregation of ciphertexts into a constant number of elements, analogously to what we described in section 1.

The scheme basically goes as follows. A ciphertext contains a random tag tagc in the element E_1 while each key contains $n-1$ tags (tagk_i for each K_i element) that are aggregated into $\mathsf{tagk} = \sum_{i=2}^n \mathsf{tagk}_i y_i$ upon decryption of a ciphertext intended for $\vec{Y}$. The receiver can decrypt if $\mathsf{tagk} \neq \mathsf{tagc}$ (and $\vec{X} \cdot \vec{Y} = 0$).

Construction 3. (Adaptively secure zero IPE)

▶ $\mathsf{Setup}(1^\lambda, n)$: chooses bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$. It then picks generators $g, v, v_1, v_2 \xleftarrow{\$} \mathbb{G}$ and chooses $\alpha, \alpha_0, \alpha_1, \dots, \alpha_n, a_1, a_2, b \xleftarrow{\$} \mathbb{Z}_p$. Let $\vec{\alpha} = (\alpha_1, \dots, \alpha_n)$ and $\vec{H} = (h_1, \dots, h_n) = g^{\vec{\alpha}}$. The public key consists of

$$\mathsf{pk} = \begin{pmatrix} g, \ w = g^{\alpha_0}, \ Z = e(g,g)^{\alpha \cdot a_1 \cdot b}, \ \vec{H} = g^{\vec{\alpha}}, \ A_1 = g^{a_1}, \ A_2 = g^{a_2}, B = g^b, \\ B_1 = g^{b \cdot a_1}, \ B_2 = g^{b \cdot a_2}, \ \tau_1 = v \cdot v_1^{a_1}, \ \tau_2 = v \cdot v_2^{a_2}, \ T_1 = \tau_1^b, \ T_2 = \tau_2^b \end{pmatrix}$$

The master key is defined to be $\mathsf{msk} = (g^\alpha, g^{\alpha a_1}, v, v_1, v_2)$.

▶ $\mathsf{Keygen}(\vec{X}, \mathsf{msk}, \mathsf{pk})$: parses $\vec{X}$ as $(x_1, \ldots, x_n)$ and returns $\perp$ if $x_1 = 0$. Otherwise, it picks $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$, $z_1, z_2 \xleftarrow{\$} \mathbb{Z}_p$, $\mathsf{tagk}_2, \ldots, \mathsf{tagk}_n \xleftarrow{\$} \mathbb{Z}_p$, sets $r = r_1 + r_2$ and generates $\mathsf{sk}_{\vec{X}} = (D_1, \ldots, D_7, K_2, \ldots, K_n, \mathsf{tagk}_2, \ldots, \mathsf{tagk}_n)$ by computing

$$\mathsf{sk}_{\mathsf{core}} = \left\{K_i = \left(g^{-\alpha_1 \frac{x_i}{x_1}} \cdot g^{\alpha_i} \cdot g^{\alpha_0 \cdot \mathsf{tagk}_i}\right)^{r_1}\right\}_{i=2,\ldots,n},$$

$$\mathsf{sk}_{\mathsf{adapt}} = \begin{pmatrix} D_1 = g^{\alpha a_1} \cdot v^r, & D_2 = g^{-\alpha} \cdot v_1^r \cdot g^{z_1}, & D_3 = B^{-z_1}, & D_4 = v_2^r \cdot g^{z_2}, \\ D_5 = B^{-z_2}, & D_6 = B^{r_2}, & D_7 = g^{r_1} & \end{pmatrix}.$$

▶ $\mathsf{Encrypt}(\vec{Y}, \mathsf{M}, \mathsf{pk})$: to encrypt $\mathsf{M} \in \mathbb{G}_T$ under $\vec{Y} = (y_1, \ldots, y_n) \in (\mathbb{Z}_p)^n$, pick $s_1, s_2, t, \mathsf{tagc} \xleftarrow{\$} \mathbb{Z}_p$ and compute $C = (C_1, \ldots, C_7, E_0, E_1, E_2, \mathsf{tagc})$ where

$$C_{\mathsf{core}} = \left(E_0 = \mathsf{M} \cdot Z^{s_2}, \quad E_1 = (g^{\alpha_0 \cdot \mathsf{tagc}} \cdot g^{\langle \vec{\alpha}, \vec{Y} \rangle})^t, \quad E_2 = g^t\right),$$

$$C_{\mathsf{adapt}} = \begin{pmatrix} C_1 = B^{s_1 + s_2}, & C_2 = B_1^{s_1}, & C_3 = A_1^{s_1}, & C_4 = B_2^{s_2}, \\ C_5 = A_2^{s_2}, & C_6 = \tau_1^{s_1} \cdot \tau_2^{s_2}, & C_7 = T_1^{s_1} \cdot T_2^{s_2} \cdot w^{-t} & \end{pmatrix}.$$

▶ $\mathsf{Decrypt}(C, \vec{Y}, \mathsf{sk}_{\vec{X}}, \vec{X}, \mathsf{pk})$: computes $\mathsf{tagk} = \mathsf{tagk}_2 y_2 + \cdots + \mathsf{tagk}_n y_n$ and then $W_1 = \prod_{j=1}^{5} e(C_j, D_j) \cdot (\prod_{j=6}^{7} e(C_j, D_j))^{-1} = e(g,g)^{\alpha \cdot a_1 \cdot b \cdot s_2} \cdot e(g,w)^{r_1 t}$, as well as $W_2 = \left(\frac{e(K_2^{y_2} \cdots K_n^{y_n}, E_2)}{e(E_1, D_7)}\right)^{\frac{1}{\mathsf{tagk} - \mathsf{tagc}}} = e(g,w)^{r_1 t}$. It finally recovers the plaintext as $\mathsf{M} = E_0 / Z^{s_2} = E_0 / e(g,g)^{\alpha \cdot a_1 \cdot b \cdot s_2} \leftarrow E_0 \cdot W_2 \cdot W_1^{-1}$.

The correctness of W_2 is shown in appendix A.1, while the rest follows from [27]. As we can see, ciphertexts have the same size as in the IBE scheme of [27], no matter how large the vector $\vec{Y}$ is. Also, decryption entails a constant number of pairing evaluations (whereas ciphertexts cost $O(n)$ pairings to decrypt in [18]).

Theorem 2. *Construction 3 is adaptively secure under the DLIN and DBDH assumptions.*

Proof. The proof uses the dual system methodology similar to [27], which involves ciphertexts and private keys that can be normal or semi-functional.

- Semi-functional ciphertexts are generated by first computing a normal ciphertext $(C_0', C_1', \ldots, C_7', E_1', E_7', \mathsf{tagc}')$ and then choosing $\chi \xleftarrow{\$} \mathbb{Z}_p$ before replacing (C_4', C_5', C_6', C_7'), respectively, by

$$C_4 = C_4' \cdot g^{b a_2 \chi}, \quad C_5 = C_5' \cdot g^{a_2 \chi}, \quad C_6 = C_6' \cdot v_2^{a_2 \chi}, \quad C_7 = C_7' \cdot v_2^{a_2 b \chi}. \quad (4)$$

- From a normal key $(D_1', \ldots, D_7', K_2', \ldots, K_n', \mathsf{tagk}_2', \ldots, \mathsf{tagk}_n')$, semi-functional keys are obtained by choosing $\gamma \xleftarrow{\$} \mathbb{Z}_p$ and replacing (D_1', D_2', D_4') by

$$D_1 = D_1' \cdot g^{-a_1 a_2 \gamma}, \qquad D_2 = D_2' \cdot g^{a_2 \gamma}, \qquad D_4 = D_4' \cdot g^{a_1 \gamma}. \quad (5)$$

The proof proceeds with a game sequence starting from Game_{Real}, which is the actual attack game. The following games are defined below.

Game_0 is the real attack game but the challenge ciphertext is semi-functional.

Game_k (for $1 \le k \le q$) is identical to Game_0 except that the first i private key generation queries are answered by returning a semi-functional key.

Game_{q+1} is as Game q but the challenge ciphertext is a semi-functional encryption of a random element of $\mathbb{G}_T$ instead of the actual plaintext.

We prove the indistinguishability between two consecutive games under some assumptions. The sequence ends in Game_{q+1}, where the challenge ciphertext is independent of the challenger's bit β, hence any adversary has no advantage. $\square$

The indistinguishability of Game_{Real} and Game_0 as well as that of Game_q and Game_{q+1} can be proved exactly in the same way as in [27] and the details are given in the full version of the paper.

Lemma 1. *If DLIN is hard,* Game_0 *is indistinguishable from* Game_{Real}.

Lemma 2. *For any $1 \le k \le q$, if an adversary $\mathcal{A}$ can distinguish* Game_k *from* Game_{k-1}*, we can build a distinguisher for the DLIN problem.*

This lemma is the most non-trivial part in the theorem. The main issue is that, in order to enable adaptive security, the reduction must be done in such a way that the simulator $\mathcal{B}$ can create semi-functional keys for any vector $\vec{X}$, including those for which $\vec{X} \cdot \vec{Y}^\star = 0$. However, the crucial point is that we must prevent $\mathcal{B}$ from directly deciding whether the k^{th} queried private key is normal or semi-functional by generating a semi-functional ciphertext for itself. Indeed, if this were possible, the reduction from $\mathcal{A}$ would not be established.

To resolve this, we use a secret exponent vector $\vec{\zeta} \in \mathbb{Z}_p^n$ and embed the DLIN instance so that $\mathcal{B}$ can simulate only the key at k^{th} query for $\vec{X}$ with tags $(\mathsf{tagk}_2, \ldots, \mathsf{tagk}_n)$ and the challenge ciphertext for $\vec{Y}^\star$ with $\mathsf{tagc}^\star$ that obey the relation: $(\mathsf{tagk}_2, \ldots, \mathsf{tagk}_n, \mathsf{tagc}^\star)^\top = -M_{\vec{X},\vec{Y}^\star}\vec{\zeta}$, where $M_{\vec{X},\vec{Y}}$ is the $n \times n$ matrix defined in Eq.(2). We thereby conceptually use the n-equation technique here. A particular consequence is that if we have $\vec{X} \cdot \vec{Y}^\star = 0$ then it holds that

$$\mathsf{tagk} = \sum_{i=2}^{n} \mathsf{tagk}_i y_i^\star = \zeta_1 \sum_{i=2}^{n} \frac{x_i}{x_1} y_i^\star - \sum_{i=2}^{n} \zeta_i y_i^\star = \zeta_1 \cdot (-y_1^\star) - \sum_{i=2}^{n} \zeta_i y_i^\star = \mathsf{tagc}^\star,$$

which is the exact condition that hampers the decryption, thus $\mathcal{B}$ cannot test by itself, as desired. We are now ready to describe the proof of Lemma 2.

Proof. The distinguisher $\mathcal{B}$ receives $(g, f, \nu, g^{\theta_1}, f^{\theta_2}, \eta)$ and decides if $\eta = \nu^{\theta_1+\theta_2}$.

Setup. Algorithm $\mathcal{B}$ picks $\alpha, a_1, a_2, \delta_{v_1}, \delta_{v_2} \xleftarrow{\$} \mathbb{Z}_p$ and sets $g = g$, $Z = e(f, g)^{\alpha a_1}$,

$$\begin{array}{llll} A_1 = g^{a_1}, & A_2 = g^{a_2}, & B = g^b = f, & v_1 = \nu^{a_2} \cdot g^{\delta_{v_1}} \\ B_1 = g^{ba_1} = f^{a_1}, & B_2 = g^{ba_2} = f^{a_2}, & v = \nu^{-a_1 a_2}, & v_2 = \nu^{a_1} \cdot g^{\delta_{v_2}}, \\ \tau_1 = v v_1^{a_1} = g^{\delta_{v_1} a_1}, & \tau_2 = v v_2^{a_2} = g^{\delta_{v_2} a_2}, & \tau_1^b = f^{\delta_{v_1} a_1}, & \tau_2^b = f^{\delta_{v_2} a_2}. \end{array}$$

Next, $\mathcal{B}$ chooses $\delta_w \xleftarrow{\$} \mathbb{Z}_p, \vec{\zeta} = (\zeta_1, \ldots, \zeta_n) \xleftarrow{\$} \mathbb{Z}_p^n$, $\vec{\delta} = (\delta_1, \ldots, \delta_n) \xleftarrow{\$} \mathbb{Z}_p^n$, then defines $w = g^{\alpha_0} = f \cdot g^{\delta_w}$, and $h_i = g^{\alpha_i} = f^{\zeta_i} \cdot g^{\delta_i}$ for $i = 1, \ldots, n$. Note that, as in the proof of lemma 2 in [27] , $\mathcal{B}$ knows $\mathsf{msk} = (g^\alpha, g^{\alpha a_1}, v, v_1, v_2)$.

Key Queries. When $\mathcal{A}$ makes the j^{th} private key query, $\mathcal{B}$ does as follows.
[**Case** $j > k$] It generates a normal key, using the master secret key msk.
[**Case** $j < k$] It creates a semi-functional key, which it can do using $g^{a_1 a_2}$.
[**Case** $j = k$] It defines $\mathsf{tagk}_2, \ldots, \mathsf{tagk}_n$ as $\mathsf{tagk}_i = \zeta_1 \cdot \frac{x_i}{x_1} - \zeta_i$ for $i = 2, \ldots, n$, which implies that $(h_1^{-x_i/x_1} \cdot h_i \cdot w^{\mathsf{tagk}_i}) = g^{-\delta_1(x_i/x_1)+\delta_i+\delta_w \mathsf{tagk}_i}$, for $i = 2, \ldots, n$. Using these tags, it generates a normal private key $(D_1', \ldots, D_7', K_2', \ldots, K_n')$ using random exponents $r_1', r_2', z_1', z_2' \stackrel{\$}{\leftarrow} \mathbb{Z}_p$. Then, it sets

$$D_1 = D_1' \cdot \eta^{-a_1 a_2}, \qquad D_2 = D_2' \cdot \eta^{a_2} \cdot (g^{\theta_1})^{\delta_{v_1}}, \quad D_3 = D_3' \cdot (f^{\theta_2})^{\delta_{v_1}},$$
$$D_4 = D_4' \cdot \eta^{a_1} \cdot (g^{\theta_1})^{\delta_{v_2}}, \quad D_5 = D_5' \cdot (f^{\theta_2})^{\delta_{v_2}}, \qquad D_6 = D_6' \cdot f^{\theta_2},$$

as well as $D_7 = D_7' \cdot (g^{\theta_1})$ and $K_i = K_i' \cdot (g^{\theta_1})^{-\delta_1(x_i/x_1)+\delta_i+\delta_w \mathsf{tagk}_i}$ for $i = 2, \ldots, n$.

If $\eta = \nu^{\theta_1+\theta_2}$, $\mathsf{sk}_{\vec{X}} = (D_1, \ldots, D_7, K_2, \ldots, K_n, \mathsf{tagk}_2, \ldots, \mathsf{tagk}_n)$ is easily seen to form a normal key where $r_1 = r_1' + \theta_1$, $r_2 = r_2' + \theta_2$, $z_1 = z_1' - \delta_{v_1}\theta_2$, $z_2 = z_2' - \delta_{v_2}\theta_2$ are the underlying random exponents. If $\eta \in_R \mathbb{G}$, it can be written $\eta = \nu^{\theta_1+\theta_2} \cdot g^{\gamma}$ for some $\gamma \in_R \mathbb{Z}_p$, so that $\mathsf{sk}_{\vec{X}}$ is distributed as a semi-functional key. We note that $\mathsf{tagk}_2, \ldots, \mathsf{tagk}_n$ are independent and uniformly distributed since $\zeta_1, \ldots, \zeta_n$ (which are the solutions of a system of $n-1$ equations with n unknowns) are uniformly random and perfectly hidden from $\mathcal{A}$'s view.

Challenge. $\mathcal{A}$ outputs $M_0, M_1 \in \mathbb{G}_T$ along with a vector $\vec{Y}^\star = (y_1^\star, \ldots, y_n^\star)$. $\mathcal{B}$ flips a coin $\beta \stackrel{\$}{\leftarrow} \{0,1\}$ and computes the tag $\mathsf{tagc}^\star = -\langle \vec{Y}^\star, \vec{\zeta} \rangle$ for which $\mathcal{B}$ will be able to prepare the semi-functional ciphertext. To this end, $\mathcal{B}$ first computes a normal encryption $(C_0', C_1', \ldots, C_7', E_1', E_2', \mathsf{tagc}^\star)$ of M_β using exponents s_1', s_2', t'. It then chooses $\chi \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and computes

$$C_4 = C_4' \cdot f^{a_2 \cdot \chi}, \quad C_5 = C_5' \cdot g^{a_2 \cdot \chi}, \qquad C_7 = C_7' \cdot \nu^{-\delta_w \cdot a_1 \cdot a_2 \cdot \chi} \cdot f^{\delta_{v_2} \cdot a_2 \cdot \chi},$$
$$C_6 = C_6' \cdot v_2^{a_2 \cdot \chi}, \quad E_2 = E_2' \cdot \nu^{a_1 \cdot a_2 \cdot \chi}, \quad E_1 = E_1' \cdot (\nu^{\delta_w \cdot \mathsf{tagc}^\star + \langle \vec{Y}^\star, \vec{\delta} \rangle})^{a_1 \cdot a_2 \cdot \chi}.$$

We claim that $(C_0', C_1', C_2', C_3', C_4, C_5, C_6, C_7, E_1, E_2, \mathsf{tagc}^\star)$ is a semi-functional ciphertext with underlying exponents $\chi, s_1 = s_1', s_2 = s_2'$ and $t = t' + \log_g(\nu) a_1 a_2 \chi$. To prove this, we observe that

$$\begin{aligned} C_7 &= T_1^{s_1} \cdot T_2^{s_2} \cdot w^{-t} \cdot v_2^{a_2 b \chi} = T_1^{s_1} \cdot T_2^{s_2} \cdot w^{-t' - \log_g(\nu) a_1 a_2 \chi} \cdot (\nu^{a_1} \cdot g^{\delta_{v_2}})^{a_2 b \chi} \\ &= T_1^{s_1} \cdot T_2^{s_2} \cdot w^{-t'} \cdot (f \cdot g^{\delta_w})^{-\log_g(\nu) a_1 a_2 \chi} \cdot (\nu^{a_1} \cdot g^{\delta_{v_2}})^{a_2 b \chi} \\ &= C_7' \cdot \nu^{-\delta_w a_1 a_2 \chi} \cdot f^{\delta_{v_2} a_2 \chi}, \end{aligned}$$

where the unknown term in $v_2^{a_2 b \chi}$ is canceled out by w^{-t}. Also,

$$\begin{aligned} E_1 &= E_1' \cdot \left(h_1^{y_1^\star} \cdots h_n^{y_n^\star} \cdot w^{\mathsf{tagc}^\star}\right)^{\log_g(\nu) a_1 a_2 \chi} \\ &= E_1' \cdot \left((f^{\zeta_1} g^{\delta_1})^{y_1^\star} \cdots (f^{\zeta_n} g^{\delta_n})^{y_n^\star} \cdot (f g^{\delta_w})^{-\langle \vec{Y}^\star, \vec{\zeta} \rangle}\right)^{\log_g(\nu) a_1 a_2 \chi} \\ &= E_1' \cdot (\nu^{\langle \vec{Y}^\star, \vec{\delta} \rangle + \delta_w \cdot \mathsf{tagc}^\star})^{a_1 a_2 \chi}, \end{aligned}$$

where the unknown $f^{\log_g(\nu)}$ vanishes due to our definition of $\mathsf{tagc}^\star$. It then remains to show that $\mathsf{tagc}^\star, \mathsf{tagk}_2, \ldots, \mathsf{tagk}_n$ are still n-wise independent. But this holds since their relations form a system

$$M \cdot \vec{\zeta} := \begin{pmatrix} -\frac{x_2}{x_1} & 1 & & & \\ -\frac{x_3}{x_1} & & 1 & & \\ \vdots & & & \ddots & \\ -\frac{x_n}{x_1} & & & & 1 \\ y_1^\star & y_2^\star & y_3^\star & \ldots & y_n^\star \end{pmatrix} \begin{pmatrix} \zeta_1 \\ \zeta_2 \\ \vdots \\ \\ \zeta_n \end{pmatrix} = - \begin{pmatrix} \mathsf{tagk}_2 \\ \mathsf{tagk}_3 \\ \vdots \\ \mathsf{tagk}_n \\ \mathsf{tagc}^\star \end{pmatrix},$$

which has a solution in $\vec{\zeta}$ whenever $\det(M) = (-1)^{n+1}\vec{X} \cdot \vec{Y}^\star / x_1 \neq 0$.

Eventually, $\mathcal{A}$ outputs a bit β' and $\mathcal{B}$ outputs 0 if $\beta = \beta'$. As in [27], we see that $\mathcal{A}$ is playing Game_{k-1} if $\eta = \nu^{\theta_1+\theta_2}$ and Game_k otherwise. □

Lemma 3. *If DBDH is hard,* Game_q *and* Game_{q+1} *are indistinguishable.*

5 Functional Encryption for Non-zero Inner-Product

5.1 Negated Spatial Encryption

We begin this section by providing a co-selectively-secure construction of negated spatial encryption, which is motivated by its implication of non-zero IPE. At a high-level, our scheme can be viewed as a "negative" analogue of the Boneh-Hamburg spatial encryption [9], in very much the same way as the Lewko-Sahai-Waters revocation scheme [19] is a negative analogue of the Boneh-Boyen IBE [4]. The intuition follows exactly from section 1, where we have to use "n-equation technique". In spatial encryption, we have to deal with, in general, how we can set up a system of n equations similarly to Eq.(1). To this end, we confine the vector subspaces that we can use as follows. Our construction is a FE for $R^{\mathsf{Neg(Spatial)}} : \mathcal{W}_n \times \mathbb{Z}_p^n \to \{0,1\}$, where we define a collection $\mathcal{W}_n \subseteq \mathcal{V}_n$ of vector subspaces in $\mathbb{Z}_p^n$ as $\mathcal{W}_n = \{\mathrm{Aff}(M, \vec{0}) \in \mathcal{V}_n \mid \mathrm{rank}(M_{(-1)}) = n-1\}$, where we denote $M_{(-1)}$ as the matrix obtained by deleting the first row $M_1 \in \mathbb{Z}_p^{1\times d}$ of M.

Construction 4. (Co-selectively secure negated spatial encryption)

▶ $\mathsf{Setup}(1^\lambda, n)$: chooses a bilinear group $\mathbb{G}$ of prime order $p > 2^\lambda$ with a random generator $g \xleftarrow{\$} \mathbb{G}$. It randomly chooses $\alpha, \alpha_1, \ldots, \alpha_n \xleftarrow{\$} \mathbb{Z}_p$. Let $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n)$. The public key is $\mathsf{pk} = \big(g, g^{\vec{\alpha}}, g^{\alpha_1\vec{\alpha}}, e(g,g)^\alpha\big)$. The master key is $\mathsf{msk} = (\alpha, \vec{\alpha})$.

▶ $\mathsf{KeyGen}(V, \mathsf{msk}, \mathsf{pk})$: suppose that $V = \mathrm{Aff}(M, \vec{0})$, from a matrix $M \in (\mathbb{Z}_p)^{n\times d}$. The algorithm picks $t \xleftarrow{\$} \mathbb{Z}_p$ and outputs $\mathsf{sk}_V = (D_0, D_1, \vec{K}) \in \mathbb{G}^{d+2}$ where

$$D_0 = g^t, \qquad D_1 = g^{\alpha + t\alpha_1^2}, \qquad \vec{K} = g^{tM^\top\vec{\alpha}}.$$

▶ $\mathsf{Encrypt}(\vec{y}, \mathsf{M}, \mathsf{pk})$: picks $s \xleftarrow{\$} \mathbb{Z}_p$ and computes (C_0, C_1, C_2, C_3) as

$$C_0 = \mathsf{M} \cdot e(g,g)^{\alpha s}, \qquad C_1 = g^{s\alpha_1\langle\vec{y},\vec{\alpha}\rangle}, \qquad C_2 = g^s, \qquad C_3 = g^{\alpha_1 s}.$$

▶ $\mathsf{Decrypt}(C, \vec{y}, \mathsf{sk}_V, V, \mathsf{pk})$: the algorithm first obtains M from V. We also recall the the notation of M_1, which is the vector of the first row of M. It first solves the system of equations in $\vec{w}$ from $M_{(-1)}\vec{w} = (y_2, \ldots, y_n)^\top$, which it can do since $V \in \mathcal{W}_n$. It computes the message blinding factor $e(g,g)^{\alpha s}$ as

$$e(D_1, C_2) \cdot \left(\frac{e(C_1, D_0)}{e(\vec{K}^{\vec{w}}, C_3)}\right)^{\frac{1}{M_1\vec{w}-y_1}} = e(g^{\alpha+t\alpha_1^2}, g^s) \cdot \left(\frac{e(g^{s\alpha_1\langle\vec{y},\vec{\alpha}\rangle}, g^t)}{g^{t\vec{w}^\top M^\top \vec{\alpha}}, g^{\alpha_1 s})}\right)^{\frac{1}{M_1\vec{w}-y_1}}.$$

Computability. We claim that the decryption can be computed if $y \notin V$. Indeed, we prove that if $y \notin V$ then $M_1\vec{w} - y_1 \neq 0$ (and the above equation is well-defined). To prove the contrapositive, suppose that $M_1\vec{w} - y_1 = 0$. Then, we must have $\vec{y} \in V$ since $M\vec{w} = \begin{bmatrix} M_1 \\ M_{(-1)} \end{bmatrix} \vec{w} = \begin{bmatrix} M_1\vec{w} \\ M_{(-1)}\vec{w} \end{bmatrix} = \vec{y}$.

Correctness. We verify that decryption is correct as follows. First, we note that due to our definition of $\vec{w}$, we have $\langle M\vec{w} - \vec{y}, \vec{\alpha}\rangle = (M_1\vec{w} - y_1)\alpha_1$. Therefore, the correctness follows from the fact that

$$\left(\frac{e(g^{s\alpha_1\langle\vec{y},\vec{\alpha}\rangle}, g^t)}{e(g^{t\vec{w}^\top M^\top \vec{\alpha}}, g^{\alpha_1 s})}\right)^{\frac{1}{M_1\vec{w}-y_1}} = \left(\frac{1}{e(g,g)^{ts\alpha_1\langle M\vec{w}-\vec{y},\vec{\alpha}\rangle}}\right)^{\frac{1}{M_1\vec{w}-y_1}} = e(g,g)^{-st\alpha_1^2}.$$

Theorem 3. *Construction 4 is co-selectively secure under the q-Decisional Multi-Exponent Bilinear Diffie-Hellman assumption (q is the number of key queries).* (The proof is given in the full paper where the assumption [19] is also recalled).

Implications. For a vector $\vec{X} \in \mathbb{Z}_p^n$, the embedding $V_{\vec{X}} = \mathrm{Aff}(M_{\vec{X}}, \vec{0_n})$ defined in Eq.(3) is easily seen to be in the limited domain $\mathcal{W}_n$ since $(M_{\vec{X}})_{(-1)}$ is an identity matrix of size $n-1$ and hence $\mathrm{rank}((M_{\vec{X}})_{(-1)}) = n-1$. Therefore, from Corollary 1, the above scheme implies non-zero IPE.

5.2 Non-zero IPE under Simple Assumptions

We prove the co-selective security of our negated spatial encryption scheme under a non-standard q-type assumption introduced in [19]. Here, we show that the dual system technique [27] makes it possible to rest on simple assumptions such as DBDH and DLIN. The scheme is very similar to the zero IPE scheme of section 4.2 and we only state the differences. The intuition again follows exactly from section 1 and the security proof uses similar techniques as in [19].

Construction 5. (Co-selectively secure non-zero IPE)

▶ $\mathsf{Setup}(1^\lambda, n)$: outputs pk exactly as in the construction 3 except that we define $w = g^{\alpha_1} (= h_1)$ in this scheme, instead of g^{α_0}.

▶ $\mathsf{Keygen}(\vec{X}, \mathsf{msk}, \mathsf{pk})$: outputs $\mathsf{sk}_{\vec{X}} = (\mathsf{sk}_{\mathsf{adapt}}, \mathsf{sk}_{\mathsf{core}})$ where $\mathsf{sk}_{\mathsf{adapt}}$ is the same as in the construction 3 (with $w = g^{\alpha_1}$) and $\mathsf{sk}_{\mathsf{core}} = \{K_i = \left(g^{-\alpha_1\frac{x_i}{x_1}} \cdot g^{\alpha_i}\right)^{r_1}\}_{i=2,\ldots,n}$.

▶ $\mathsf{Encrypt}(\vec{Y}, \mathsf{M}, \mathsf{pk})$: outputs $C = (C_{\mathsf{adapt}}, C_{\mathsf{core}})$ where C_{adapt} is as in the construction 3 (with $w = g^{\alpha_1}$) and $C_{\mathsf{core}} = \left(E_0 = \mathsf{M} \cdot Z^{s_2},\ E_1 = (g^{\langle\vec{\alpha},\vec{Y}\rangle})^t,\ E_2 = g^t\right)$.

▶ $\mathsf{Decrypt}(C, \vec{Y}, \mathsf{sk}_{\vec{X}}, \vec{X}, \mathsf{pk})$: computes W_1 as in the construction 3 and W_2 as $W_2 = \left(\frac{e(K_2^{y_2} \cdots K_n^{y_n}, E_2)}{e(E_1, D_7)}\right)^{-\frac{x_1}{\vec{X}\cdot\vec{Y}}} = e(g,w)^{r_1 t}$. (See appendix A.2).

Theorem 4. *Construction 5 is co-selectively secure under the DLIN and DBDH assumptions.* (The proof is deferred to the full version of the paper.)

5.3 A Generalization of the Scheme and Its Application

Extended Ciphertext Attribute Domain. The above scheme for the relation $R^{\mathsf{NIPE}_n} : \mathbb{Z}_p^n \times \mathbb{Z}_p^n \to \{0,1\}$ can be extended so as to support relations of the form $R^{\mathsf{NIPE}_n^*} : \mathbb{Z}_p^n \times (\mathbb{Z}_p^n)^d \to \{0,1\}$, for some $d \in \mathsf{poly}(\lambda)$, and defined as $R^{\mathsf{NIPE}_n^*}(\vec{X}, (\vec{Y}_1, \ldots, \vec{Y}_d)) = 1$ iff for all $i = 1, \ldots, d$: $\vec{X} \cdot \vec{Y}_i \neq 0$.

We construct this extended system by setting up exactly the same public and private keys (for $\vec{X}$) as in the original scheme. To encrypt to $(\vec{Y}_1, \ldots, \vec{Y}_d)$, the scheme generates $C_0, \ldots, C_7$ as usual with the underlying exponents s_1, s_2, t. Then, it chooses $t_1, \ldots, t_d \in \mathbb{Z}_p$ so that $t = t_1 + \cdots + t_d$ and for $i = 1, \ldots, d$, parses $\vec{Y}_i = (y_{i,1}, \ldots, y_{i,n})$ and computes $E_{1,i} = (g^{\langle \vec{\alpha}, \vec{Y}_i \rangle})^{t_i} = (h_1^{y_{i,1}} \cdots h_n^{y_{i,n}})^{t_i}$ and $E_{2,i} = g^{t_i}$, in such a way that the ciphertext is $(C_0, \ldots, C_7, \{E_{1,i}, E_{2,i}\}_{i=1,\ldots,d})$. Decryption requires to first compute

$$W_{2,i} = \left(\frac{e(K_2^{y_{i,2}} \cdots K_n^{y_{i,n}}, E_{2,i})}{e(E_{1,i}, D_7)}\right)^{-\frac{x_1}{\vec{X}\cdot\vec{Y}_i}} = e(g,w)^{r_1 t_i},$$

for $i = 1, \ldots, d$, from which the receiver obtains $W_2 = W_{2,1} \cdots W_{2,d} = e(g,w)^{r_1 t}$. The rest is then done as usual and we explain in the full version of the paper how the security proof must be adapted.

Applications. We can obtain an identity-based revocation scheme with parameter tradeoff from the aforementioned extension. The instantiation of ID-based revocation scheme ($\mathsf{IBR}_{\leq n}$) from our non-zero inner-product system NIPE_{n+1} yields a construction with $O(1)$-size ciphertexts and $O(n)$-size private keys, where n denotes the maximal number of revoked users.

From our extended scheme NIPE_{n+1}^*, we can obtain an ID-based revocation scheme $\mathsf{IBR}_{\mathsf{poly}(\lambda)}$, without a fixed maximal number of revoked users. To revoke the set R where $|R| = r$, we divide it into a disjointed union $R = R_1 \cup \cdots \cup R_{r/n}$, where $|R_i| = n$ for all i (we assume that n divides r). We then simply construct the vector $\vec{Y}_i$ from the revocation subset R_i for each $i \in [1, r/n]$, in the same way as we use NIPE_{n+1} to instantiate $\mathsf{IBR}_{\leq n}$. We then finally encrypt using the set of vectors $(\vec{Y}_1, \ldots, \vec{Y}_{r/n})$. The correctness and security properties hold since $R^{\mathsf{IBR}_{\leq n}}(\mathsf{ID}, R) = 1 \Leftrightarrow R^{\mathsf{IBR}_{\mathsf{poly}(\lambda)}}(\mathsf{ID}, (R_1, \ldots, R_{r/n})) = 1$. The construction has $O(r/n)$-size ciphertexts and $O(n)$-size private keys. Interestingly, we note that the second scheme described by Lewko, Sahai and Waters [19] (which indeed inspires ours) can be viewed as a special case of our scheme where $n = 1$.

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. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 205–222. Springer, Heidelberg (2005)
2. Abdalla, M., Kiltz, E., Neven, G.: Generalized Key Delegation for Hierarchical Identity-Based Encryption. In: Biskup, J., López, J. (eds.) ESORICS 2007. LNCS, vol. 4734, pp. 139–154. Springer, Heidelberg (2007)
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. 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)
5. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
6. Boneh, D., 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., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
8. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 586–615. Springer, Heidelberg (2001)
9. 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)
10. 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)
11. Canetti, R., Halevi, S., Katz, J.: A Forward-Secure Public-Key Encryption Scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 254–271. Springer, Heidelberg (2003)
12. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
13. Delerablée, C.: Identity-Based Broadcast Encryption with Constant Size Ciphertexts and Private Keys. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 200–215. Springer, Heidelberg (2007)
14. Freeman, D.: Converting Pairing-Based Cryptosystems from Composite-Order Groups to Prime-Order Groups. In: Eurocrypt 2010. LNCS. Springer, Heiidelberg (to appear, 2010)
15. Gentry, C., Waters, B.: Adaptive Security in Broadcast Encryption Systems (with Short Ciphertexts). In: Joux, A. (ed.) Eurocrypt 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (2009)
16. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM CCS 2006, pp. 89–98 (2006)

17. Iovino, V., Persiano, G.: Hidden-Vector Encryption with Groups of Prime Order. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 75–88. Springer, Heidelberg (2008)
18. 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)
19. Lewko, A., Sahai, A., Waters, B.: Revocation Systems with Very Small Private Keys. In: IEEE Symposium on Security and Privacy, S&P (to appear, 2010)
20. 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)
21. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: ACM CCS 2007, pp. 195–203 (2007)
22. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
23. Sakai, R., Furukawa, J.: Identity-Based Broadcast Encryption. In: Cryptology ePrint Archive: Report 2007/217 (2007), http://eprint.iacr.org/2007/217
24. 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)
25. Shen, E., Shi, E., Waters, B.: Predicate Privacy in Encryption Systems. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 457–473. Springer, Heidelberg (2009)
26. Shi, E., Waters, B.: Delegating Capabilities in Predicate Encryption Systems. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 560–578. Springer, Heidelberg (2008)
27. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

A Verifying Correctness in Decryption

A.1 For the Zero IPE Scheme of Section 4.2

$$W_2 = \left(\frac{e(\prod_{i=2}^{n} K_i^{y_i}, E_2)}{e(E_1, D_7)}\right)^{\frac{1}{\mathsf{tagk}-\mathsf{tagc}}} = \left(\frac{e\left(\prod_{i=2}^{n}(g^{-\alpha_1\frac{x_i}{x_1}} g^{\alpha_i} w^{\mathsf{tagk}_i})^{r_1 y_i}, g^t\right)}{e\left(\left(g^{\langle\vec{\alpha},\vec{Y}\rangle}\cdot w^{\mathsf{tagc}}\right)^t, g^{r_1}\right)}\right)^{\frac{1}{\mathsf{tagk}-\mathsf{tagc}}}$$

$$= \left(\frac{e\left((g^{-\alpha_1\frac{x_2y_2+\cdots+x_ny_n}{x_1}} g^{\alpha_2y_2+\cdots+\alpha_ny_n} w^{\mathsf{tagk}_2y_2+\cdots+\mathsf{tagk}_ny_n})^{r_1}, g^t\right)}{e\left(\left(g^{\alpha_1y_1+\alpha_2y_2+\cdots+\alpha_ny_n}\cdot w^{\mathsf{tagc}}\right)^t, g^{r_1}\right)}\right)^{\frac{1}{\mathsf{tagk}-\mathsf{tagc}}}$$

$$= e\left(g^{-\alpha_1(\frac{x_2y_2+\cdots x_ny_n}{x_1}+y_1)} w^{(\mathsf{tagk}-\mathsf{tagc})}, g\right)^{\frac{r_1t}{\mathsf{tagk}-\mathsf{tagc}}}$$

$$= e\left(g^{-\alpha_1\frac{\vec{X}\cdot\vec{Y}}{x_1}} w^{(\mathsf{tagk}-\mathsf{tagc})}, g\right)^{\frac{r_1t}{\mathsf{tagk}-\mathsf{tagc}}} = e(g,w)^{r_1t}.$$

A.2 For the Non-zero IPE Scheme of Section 5.2

$$
\begin{aligned}
W_2 &= \left(\frac{e(\prod_{i=2}^{n} K_i^{y_i}, E_2)}{e(E_1, D_7)}\right)^{-\frac{x_1}{\vec{X}\cdot\vec{Y}}} = \left(\frac{e\Big(\prod_{i=2}^{n}(g^{-\alpha_1\frac{x_i}{x_1}}g^{\alpha_i})^{r_1 y_i}, g^t\Big)}{e\Big(\big(g^{\alpha_1 y_1+\cdots+\alpha_n y_n}\big)^t, g^{r_1}\Big)}\right)^{-\frac{x_1}{\vec{X}\cdot\vec{Y}}} \\
&= \left(\frac{e\Big((w^{-\frac{x_2y_2+\cdots+x_ny_n}{x_1}}g^{\alpha_2 y_2+\cdots+\alpha_n y_n})^{r_1}, g^t\Big)}{e\Big(\big(w^{y_1}\cdot g^{\alpha_2 y_2+\cdots+\alpha_n y_n}\big)^t, g^{r_1}\Big)}\right)^{-\frac{x_1}{\vec{X}\cdot\vec{Y}}} \\
&= e\Big(w^{\frac{\vec{X}\cdot\vec{Y}}{x_1}}, g\Big)^{r_1 t\cdot\frac{x_1}{\vec{X}\cdot\vec{Y}}} = e(g,w)^{r_1 t}.
\end{aligned}
$$

Security of Encryption Schemes in Weakened Random Oracle Models (Extended Abstract)

Akinori Kawachi, Akira Numayama, Keisuke Tanaka, and Keita Xagawa

Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, W8-55, 2-12-1 Ookayama Meguro-ku, Tokyo 152-8552, Japan
{kawachi,numayam4,keisuke,xagawa5}@is.titech.ac.jp

Abstract. Liskov proposed several weakened versions of the random oracle model, called *weakened random oracle models* (WROMs), to capture the vulnerability of ideal compression functions, which are expected to have the standard security of hash functions, i.e., collision resistance, second-preimage resistance, and one-wayness properties. The WROMs offer additional oracles to break such properties of the random oracle. In this paper, we investigate whether public-key encryption schemes in the random oracle model essentially require the standard security of hash functions by the WROMs. In particular, we deal with four WROMs associated with the standard security of hash functions; the standard, collision tractable, second-preimage tractable, first-preimage tractable ones (ROM, CT-ROM, SPT-ROM, and FPT-ROM, respectively), done by Numayama et al. for digital signature schemes in the WROMs. We obtain the following results: (1) The OAEP is secure in all the four models. (2) The encryption schemes obtained by the Fujisaki-Okamoto conversion (FO) are secure in the SPT-ROM. However, some encryption schemes with FO are insecure in the FPT-ROM. (3) We consider two artificial variants wFO and dFO of FO for separation of the WROMs in the context of encryption schemes. The encryption schemes with wFO (dFO, respectively) are secure in the CT-ROM (ROM, respectively). However, some encryption schemes obtained by wFO (dFO, respectively) are insecure in the SPT-ROM (CT-ROM, respectively). These results imply that standard encryption schemes such as the OAEP and FO-based one do not always require the standard security of hash functions. Moreover, in order to make our security proofs complete, we construct an efficient sampling algorithm for the binomial distribution with exponentially large parameters, which was left open in Numayama et al.'s paper.

Keywords: public-key encryption schemes, weakened random oracle models, OAEP, Fujisaki-Okamoto conversion.

1 Introduction

Background. In order to design new cryptographic schemes, we often follow the random oracle methodology [1]. First, we analyze the security of cryptographic schemes, by idealizing hash functions as truly random functions called the *random oracle*. When it comes to implementations of these schemes, we replace the random oracles by cryptographic hash functions such as MD5 [2] and SHA-1 [3]. This replacement is called an instantiation of the random oracle.

The random oracle methodology causes a trade-off between efficiency and provable security. The schemes proven secure in the random oracle model (ROM) are in general more efficient than those proven secure in the standard model. However, the security proofs in the ROM do not directly guarantee the security in the standard model, i.e., an instantiation of the random oracle might make the cryptographic schemes insecure. Even worse, several recent works [4,5,6] showed that some schemes secure in the ROM have no secure instantiation.

There are several properties of the ROM to prove the security of cryptographic properties. In particular, the ROM is expected to satisfy the one-wayness, second-preimage resistance, and collision resistance properties. We call these properties as the *standard security of hash functions*. These properties are indeed critical in many schemes for their security proofs. For example, the security of the Full-Domain-Hash (FDH) signature schemes (e.g., [7]), which are secure in the ROM, relies on the collision-resistance property of the ROM. That is, if we can obtain two distinct messages m, m' such that $H(m) = H(m')$ and the signature $\sigma = \mathsf{Sig}(H(m))$, then we can obtain a valid forgery (m', σ), where H is a hash function and Sig is a signing algorithm. Leurent and Nguyen also presented the attacks extracting the secret keys on several *hash-then-sign* type signature schemes and identity-based encryption schemes if the underlying hash functions are not collision resistant [8].

Recent progress on the attacks against cryptographic hash functions such as MD5 and SHA-1 raises the question on the assumption that hash functions are collision resistant and one-way (e.g.,[9,10,11]). Therefore, it is significant to investigate whether the collision resistance property (as well as the one-wayness and second-preimage resistance properties, which are weaker notions than the collision resistance one) of the ROM is essential to prove the security of the schemes or not. More generally, it is worth classifying the schemes by the first-preimage, second-preimage, and collision resistance properties of the ROM that their security essentially requires.

Weak versions of random oracle models. Several works recently highlighted some specific properties of the ROM for secure cryptographic constructions in the ROM.

Nielsen proposed the *non-programmable* random oracle model where the random oracle is not *programmable* [12]. In this model, one cannot set the values that the random oracle answers to some convenient values. It was showed in [12] that a non-interactive non-committing encryption scheme exists in the ROM (assuming that trapdoor permutations exists), but not in the *non-programmable* random oracle model.

Unruh proposed a ROM with *oracle-dependent* auxiliary inputs [13]. In this setting, adversaries obtain an auxiliary input that contains information with respect to the random oracle (e.g. collisions). He showed that the RSA-OAEP encryption scheme [14] is secure in the ROM even under the presence of *oracle-dependent* auxiliary inputs.

Liskov proposed several weakened versions of the random oracle model, called *weakened random oracle models* (WROMs), which offer additional oracles to break some properties of the random oracle [15]. These model captures the situation that adversaries are given an attack algorithm for breaking some specific property of the functions. For example, the first-preimage tractable random oracle model offers the random oracle and the first-preimage oracle associated with the random oracle, which returns a first-preimage of the random oracle to adversaries. This first-preimage oracle then corresponds to the attack to the first preimage property of a hash function. We can replace the additional oracle to others such as the second-preimage and collision ones that correspond to the attack to the properties. Thus, the WROMs can capture vulnerability of hash functions even if the parties are allowed to utilize ideal ones as in the ROM. By using WROMs, Liskov constructed hash functions based on weak ideal compression functions and proved it is indifferentiable from the random oracle.

Several results already analyzed the security in the WROMs. Hoch and Shamir applied Liskov's idea to prove the indifferentiability of another hash construction [16]. Pasini and Vaudenay also applied Liskov's idea to the security analysis of digital signature schemes [17]. They considered the security of *hash-then-sign* type signature schemes in the first-preimage tractable random oracle model. Numayama, Isshiki, and Tanaka formalized the WROMs, which allows us to formally analyze the security of the schemes [18]. By using these models, they classified several digital signature schemes by the properties of the ROM. Fischlin and Lehmann also proposed a weakened random oracle model in a similar way to Liskov's one in the context of secure combiners [19].

Our contributions. In this paper, we investigate whether public-key encryption schemes constructed in the ROM essentially require the standard security of hash functions by further extending the direction originated from Liskov. In particular, we consider their security in the standard, collision tractable, second-preimage tractable, and first-preimage tractable random oracle models (ROM, CT-ROM, SPT-ROM, and FPT-ROM, respectively for short). Note that they are ordered according to their strengths, i.e., the security of encryption schemes in the FPT-ROM implies that in the SPT-ROM and such implications hold between each adjacent two models.

We demonstrate that the security notions in the four WROMs can be strictly separated in the context of encryption schemes. For the separation, we focus on the security of the encryption schemes obtained by the Fujisaki-Okamoto conversion (FO) [20], its two artificial variants (dFO and wFO), and the OAEP [14]. Precisely, we prove the following four statements:

1. OAEP is IND-CCA2 secure in the FPT-ROM.
2. FO is IND-CCA2 secure in the SPT-ROM, but *not* IND-CPA secure in the FPT-ROM.
3. wFO is IND-CCA2 secure in the CT-ROM, but *not* IND-CCA2 secure in the SPT-ROM.
4. dFO is IND-CCA2 secure in the ROM, but *not* IND-CCA2 secure in the CT-ROM.

We summarize the security of four schemes in Table 1.

Table 1. Security of four schemes

scheme/model	ROM	CT-ROM	SPT-ROM	FPT-ROM
OAEP	secure			
FO	secure			insecure
wFO	secure		insecure	
dFO	secure	insecure		

This separation suggests that some public-key encryption schemes essentially require the standard security of hash functions. These notions were also separated in the context of digital signature schemes in [18]. We stress that the role of the collision and second-preimage oracles in encryption schemes is not as clear as that in digital signature schemes. For example, it is easy to see that the collision oracle, breaking the collision resistance property of the random oracle, directly makes a simple scheme vulnerable, but not so easy for the case of encryption schemes. Actually, we need to develop new proof techniques for the (in)security of encryption schemes under additional oracles.

It also suggests that standard encryption schemes such as the OAEP and FO-based ones do not always require the standard security of hash functions for the random oracle. We believe that our results do not only give an example of the first application of the WROMs to encryption schemes, but they are also of independent interest. As far as we know, our results give the first evidence that the OAEP encryption scheme can be used in a practical application even without the first-preimage resistance property, i.e., the one-wayness property. In other words, the OAEP remains secure even if we remove the first-preimage resistance property. This can also be said on FO-based encryption schemes on the second-preimage resistance property.

On the security of the OAEP, Kiltz and Pietrzak recently showed that there is no construction for padding-based encryption schemes including the OAEP that has a black-box reduction from ideal trapdoor permutations to its IND-CCA2 security in [21]. However, they wrote in the paper that the security proof in the ROM can be still a valid argument in practice. We believe so is our security proof in the WROMs.

For the security proof, we explicitly show how to sample approximately in polynomial time from binomial distributions with exponentially large parameters, that is, a polynomial-time sampling algorithm whose output distribution is statistically close to the binomial distribution. For this algorithm, we arrange and combine sampling algorithms that run over real numbers proposed in the field of statistics [22,23,24,25], and give a precise analysis for discretization.

It should be noted that on the security proofs of the digital signature schemes in the WROMs [18], Numayama et al. assumed such an efficient sampling algorithm and thus gave no explicit construction. They left the construction of the sampling algorithm as an open problem. By the sampling algorithm we explicitly show, it is no longer necessary to assume the sampling algorithm in their security proofs of the digital signature schemes [18] as well as those of the public-key encryption scheme in this paper.

The sampling algorithm shown in this paper is adapted for cryptographic use since the statistical closeness to the original distribution is measured by the total variation distance, which is standard in cryptography but not usually required in statistics. The sampling algorithm is useful for other cryptographic tasks as in Numayama et al.'s and this paper.

Comparisons with other models. As mentioned above, a few models that weaken the power of the random oracle were already proposed such as the non-programmable model [12] and the oracle-dependent auxiliary input model [13].

The non-programmable model is not simply comparable with WROMs since the programmability does not imply the collision resistance and vice versa. The target of the oracle-dependent auxiliary input model partially overlaps that of the WROMs.

For a simple comparison, we now focus on the security of the OAEP in both models. Unruh showed a similar result as ours for the OAEP encryption scheme [13]. He proposed a random oracle model where oracle-dependent auxiliary inputs are allowed. In his setting, the adversary of some cryptographic protocol obtains an auxiliary input that contains the information (e.g., collisions) on the random oracle. He showed that the OAEP encryption scheme [14] is still secure in the random oracle model even in his model. This result indicates an important fact that the security of the OAEP encryption scheme does not depend on the collision resistance property since the oracle-dependent auxiliary input can contain a sufficiently long list of collisions.

Our results also present the security of the OAEP in a weak version of the random oracle. However, there are at least two differences between Unruh's result and ours. First, the random oracle model with the oracle-dependent auxiliary input does not completely capture the *adaptive* security of hash functions, and this model still has the second-preimage resistance and the first-preimage resistance properties. Hence, only by his result, we cannot say whether these two properties are necessary or not in order to prove the security of the OAEP encryption scheme. In contrast to Unruh's result, our result clearly shows that the two adaptive securities of hash functions such as the first-preimage resistance and the second-preimage resistance are not necessary to prove the security of the OAEP encryption scheme.

Second, Unruh constructed the reduction algorithm which breaks the partial-domain one-wayness of the underlying trapdoor permutation using the adversary which breaks the IND-CCA2 security of the OAEP encryption scheme. The running time of the reduction algorithm is not bounded by any polynomial. Therefore, he use the security amplification technique for the partial-domain one-wayness. By using this technique, he can avoid employing a stronger assumption that even quasi-polynomial time adversary cannot break the partial-domain one-wayness, and can prove the security under the standard partial-domain one-wayness against polynomial-time adversary.

In contrast to Unruh's result, we construct the polynomial-time reduction algorithm using the adversary, and hence we do not require the security amplification technique for the partial-domain one-wayness, which can be considered as a simplification of Unruh's proof.

Organization. In Section 2, we describe the details of the WROMs and their properties. We also discuss the simulation methods that are applicable to these models. In Section 3, after reviewing the encryption schemes we consider, we show their (in)security in the WROMs. Many technical details will be omitted from this extended abstract. We will describe them in the full version [26].

Notation. Before starting technical parts of this paper, we introduce our notation used in the rest of the paper. For a table $\mathbb{T} = \{(x, y)\}$, we define $\mathbb{T}(y) = \{(x', y') \in \mathbb{T} \mid y' = y\}$. For a distribution D, $x \leftarrow D$ denotes that x is sampled according to D. The function $D(x)$ stands for the probability function of the distribution D.

Let $s \leftarrow S$ denote that s is sampled from the uniform distribution over a finite set S. $\#S$ denotes the number of elements in S. For a probabilistic Turing machine $\mathcal{A}$ and its input x, let $\mathcal{A}(x)$ denote the output distribution of $\mathcal{A}$ on input x.

We usually denote by k a security parameter of a cryptographic scheme in this paper. We also denote by k' length of plaintexts unless it is specified. k' is implicitly assumed to be polynomially related to the security parameter k, that is, $k' = k^{\Theta(1)}$. We say a function $f(k)$ is negligible in k if $f(k) \leq 2^{-\omega(\log k)}$. For two distributions D_1 and D_2 over a finte set S, we denote the statistical distance (the total variation distance) between them by $\Delta(D_1, D_2)$, defined by $\frac{1}{2}\sum_{s \in S} |D_1(s) - D_2(s)|$. We say two distributions D_1 and D_2 are statistically close if $\Delta(D_1, D_2) \leq 2^{-\omega(\log k)}$.

2 The Weakened Random Oracle Models

In this section, we first review the definitions of the WROMs. Next, we present an important property called *weak uniformity* of the WROMs, which is useful for security proofs of encryption schemes. We also discuss the simulation methods of [18] used for the security proofs in the WROMs.

2.1 Definitions of the Weakened Random Oracle Models

To give formal definitions of the WROMs, we define some notation. Let X and Y be finite sets. Let H be a hash function chosen randomly from all of the functions from X to Y. We denote by $\mathbb{T}_H$ the table $\{(x, H(x)) \mid x \in X\}$. We identify the hash function H with the table $\mathbb{T}_H$.

We next define the random oracle and the additional oracles associated with $H : X \to Y$ as follows. (For more details, see [18].)

Random oracle $\mathcal{RO}^H$: Given x, return y such that $(x, y) \in \mathbb{T}_H$.

Collision oracle $\mathcal{CO}^H$: On the query, first pick one entry $(x, y) \in \mathbb{T}_H$ uniformly at random. If there is no other entry $(x', y) \in \mathbb{T}_H$, then answer $\perp$. Otherwise, pick one entry $(x', y) \in \mathbb{T}_H$ satisfying $x \neq x'$ uniformly at random and answer (x, x').

Second-preimage oracle $\mathcal{SPO}^H$: Given (x, y), if $(x, y) \notin \mathbb{T}_H$ answer $\perp$. If there is no other entry $(x', y) \in \mathbb{T}_H$, then answer $\perp$. Otherwise, pick one entry $(x', y) \in \mathbb{T}_H$ satisfying $x \neq x'$ uniformly at random and answer x'.

First-preimage oracle $\mathcal{FPO}^H$: Given y, if there is any entry $(x, y) \in \mathbb{T}_H$ then return such an x uniformly at random. Otherwise return $\perp$.

Remark 1. We usually identify the random oracle and the underlying hash function. However, in this paper as in [18], we explicitly distinguish them by regarding the random oracle as an interface to the underlying hash function. This setting helps us to make the WROMs with an additional oracle well-defined.

The formal definitions of the WROMs are given as follows. The WROMs consist of three components, a hash function h chosen randomly from all of the functions from X to Y, the random oracle, and the additional oracle associated with h. The models are called the CT-ROM, SPT-ROM, and FPT-ROM, if the additional oracle is the collision, second-preimage, and first-preimage oracle, respectively.

Remark 2. The collision oracle may output $\perp$ even if there exists a collision (x, x') in the table. This stems from the simulation method of Numayama et al. [18], and causes no serious problems. Note that the collision oracle outputs $\perp$ with probability

$(1 - 1/\#Y)^{\#X-1}$. In the case where $\#X \geq \#Y$, we can find a collision with polynomially many queries since since $(1 - 1/\#Y)^{\#X-1} \leq \exp(-(\#X - 1)/\#Y)$. In the case where $\#Y = k^{O(1)} \cdot \#X$, we can again find a collision with polynomially many queries $(1 - 1/\#Y)^{\#X-1} \leq 1 - 1/k^{O(1)}$. Finally, in the case where $\#Y = k^{\omega(1)} \cdot \#X$, the following lemma shows that there are no collisions with overwhelming probability.

Lemma 1. *Let $H : X \to Y$ be the hash function, and n_y the number of preimages of y under the function H, that is, $n_y = \#\mathbb{T}_H(y)$. Let BAD denote the event that there is some y such that $n_y > L$. Then for all sufficiently large Y, we have $\Pr_H[\mathsf{BAD}] < \frac{1}{(\#Y)^2}$, where $L = \frac{5 \ln \#Y}{\ln \ln \#Y} \frac{\#X}{\#Y}$ if $\#X \geq \#Y$, or $L = \frac{5 \ln \#Y}{\ln \ln \#Y}$ otherwise.*

The proof is obtained by the standard argument on the balls and bins game by regarding X and Y as sets of balls and bins, respectively. For the details on the game, see a standard textbook (e.g., [27]).

2.2 Difference from the Random Oracle Model

We observe an important difference between the ROM and WROMs by considering the ROM and FPT-ROM. In the both models, the function H, i.e., the table $\mathbb{T}_H$ is uniformly distributed.

In the ROM, if one queries some x that has never been queried to the random oracle, the value of $H(x)$ is uniformly distributed regardless of the past queries. That is, the knowledge of the past queries does not affect the entries not queried in the table. This property of the ROM is called *uniformity*. In contrast to the situation in the ROM, when it comes to the FPT-ROM, this property is not attained. Recall that the first-preimage oracle *uniformly* returns one of the preimages, say x, of queried value y. If the first-preimage oracle leaks a number of preimages of y, the value of $H(x)$ is *not* uniformly distributed for an x not queried yet.

In order to observe this situation, let us consider the following extreme case. Let $y^* = H(x^*)$ for some $x^* \in X$ and suppose that y^* has the unique preimage x^*. Then the first-preimage oracle always returns the same x^* on the input y^*, which convinces us that the number of the preimages of y^* is exactly 1. This implies that the other $x \neq x^*$ does not take a value y^* under H. Therefore, the random oracle no longer has the uniformity in the FPT-ROM. This is a critical difference between the ROM and FPT-ROM since we often make use of the uniformity in the security proofs of the public-key encryption schemes.

We prove the following lemma to overcome this barrier in the WROMs, which states that the WROMs still has weak uniformity instead of the uniformity. The weak uniformity is still useful for the security proofs of the public-key encryption schemes in the WROMs.

Lemma 2 (Weak Uniformity). *In the WROMs, the output distribution of the random oracle is statistically close to the uniform distribution. More formally, it is stated as follows. Let $H : X \to Y$ be the hash function in the WROMs. Let $\mathcal{A}$ be a probabilistic oracle Turing machine that makes at most q queries to the random oracle $\mathcal{RO}^H$ and the additional oracle $\mathcal{O}^H$, where $\mathcal{O}^H$ represents one of the additional oracles $\mathcal{CO}^H$, $\mathcal{SPO}^H$, and $\mathcal{FPO}^H$. $V_{\mathcal{A},H}(x)$ denotes the random variable that represents the hash value*

$\mathcal{RO}^H(x)$, where $x \leftarrow \mathcal{A}^{\mathcal{RO}^H,\mathcal{O}^H}$ and the correspondence $(x, H(x)) \in \mathbb{T}_H$ is not answered by the two oracles.

Then, for any $\mathcal{A}$, the following holds:

$$\Delta(V_{\mathcal{A},H}(x), U_Y) \leq \begin{cases} \frac{1}{\#Y}\left(5q + 1 + \frac{4q^2}{\#Y} + 20q\frac{\ln \#Y}{\ln\ln \#Y}\right) & \text{if } \#X \geq \#Y, \\ \frac{1}{\#X}\left(5q + 1 + \frac{4q^2}{\#X} + 20q\frac{\ln \#Y}{\ln\ln \#Y}\right) & \text{if } \#X < \#Y. \end{cases}$$

Here, the probability is taken over random choices of the hash function H and the random coin of $\mathcal{A}$.

2.3 Simulation Methods

In almost all the security proofs in the ROM, the reduction algorithms simulate the random oracles. When it comes to the security proofs in the WROMs, the reduction algorithms have to simulate both the random and the additional oracle, which makes differences of the simulation methods in the WROMs from those in the ROM.

Numayama et al.'s methods. Numayama et al. proposed the simulation methods for WROMs, but they required an unproven assumption. Let $B_{N,p}$ denote the binomial distribution with parameters N and p whose probability function is $B_{N,p}(x) = \binom{N}{x} p^x(1 - p)^{N-x}$ for $x = 0, \ldots, N$, where the parameters N and p take values approximately $\#X$ and $1/\#Y$ for a hash function $H : X \rightarrow Y$, say, $(N, p) = (2^{128}, 2^{-128})$. Their simulation methods required the efficient sampler for $B_{N,p}$ with exponentially large N and small p, and they assumed its existence.

Assumption 1. *There is a probabilistic Turing machine $\mathsf{B_N}$ such that the output distribution $\mathsf{B_N}(N, p)$ on inputs N and p is equal to the binomial distribution $B_{N,p}$ and it runs in polynomial time in $\log N$ and $\log p^{-1}$, where N is a positive integer and $0 \leq p \leq 1$ is a rational number.*

Under this assumption, they constructed the simulation algorithms, RO, CO, SPO, and FPO, for the security proofs in the WROMs as given in the following proposition. See [18] for the details of the algorithms.

Proposition 1 (Simulation Method [18]). *We can perfectly simulate the random oracle, the collision oracle, second-preimage oracle, and first-preimage oracle in the* WROM*s under Assumption 1. That is, the output distributions of the random oracle, collision oracle, second-preimage oracle, and first-preimage oracle in the* WROM*s are identical to the output distributions of the algorithms* RO*,* CO*,* SPO*, and* FPO*, under Assumption 1.*

Removing the assumption. For the security proof in the WROMs of digital signature schemes in [18] and encryption schemes in this paper, it is sufficient to utilize a weaker sampling algorithm that generates a distribution *not equal but statistically close* to the binomial distribution $B_{N,p}$. Then, their security proofs can work by just adding negligibly small errors induced by the statistical distance in their analyses.

There are quite many papers (e.g., [25]) on the efficient sampling methods from the binomial distribution in the field of statistics. However, their basic computation model is totally different from the model in the cryptography. As far as the authors' knowledge, all these results are based on the computation model that directly manipulates *real* numbers without errors. If we translate them to those in the bit computation model used in the cryptography, we have to bound the statistical distance between the real distribution and the output distribution generated by the sampling algorithms in the bit computation model rather than the real-number one. Numayama et al. mentioned that they could neither find precise analyses of the statistical distance, nor construct the sampling algorithms by themselves in [18]. Therefore, they had to put the above assumption.

In fact, there is an efficient sampling algorithm appropriate for our purpose in the real-number computation model [25]. We modify the algorithm and rigorously analyze the error bound in the bit computation model. We can finally obtain the following theorem on the sampling algorithm.

Theorem 1. *There is a probabilistic Turing machine* $\mathsf{B_N}$ *such that, for the output distribution* $\mathsf{B_N}(N, p, \epsilon)$ *on inputs* N, p *and* ϵ*, the statistical distance between* $\mathsf{B_N}(N, p, \epsilon)$ *and* $B_{N,p}$ *is at most* ϵ *and it runs in polynomial time in* $\log N, \log p^{-1}$ *and* $\log \epsilon^{-1}$*, where* N *is a positive integer and* $0 \le p \le 1, 0 < \epsilon \le 1$ *are rational numbers.*

Note that the algorithm can control the error parameter ϵ. This property is useful in cryptographic applications for the security proofs even if the other parameters N and p are not sufficiently large. We will put the details of the algorithm and its analysis in the full version.

As a result, we can remove the above assumption and obtain the following theorem.

Theorem 2 (Simulation Method without Assumption 1). *We can statistically simulate the random oracle, collision oracle, second-preimage oracle, and first-preimage oracle in the* WROM*s. That is, the output distributions of the oracles in the* WROM*s are statistically close to the output distributions of the algorithms* RO*,* CO*,* SPO*, and* FPO*, respectively.*

3 The Encryption Schemes and Their Security in the Weakened Random Oracle Models

In this section, we examine the security in the WROMs of the public-key encryption schemes. We particularly discuss separations for notions of ROM, CT-ROM, SPT-ROM, and FPT-ROM by showing (in)security of public-key encryption schemes obtained by the Fujisaki-Okamoto conversion (FO) and its two variants (dFO and wFO), and OAEP.

Public-key encryption schemes. We first give notation and notions for public-key encryption schemes briefly. For details, see standard textbooks, e.g., [28].

A public-key encryption scheme $\mathcal{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ over a plaintext space $\mathcal{M}$ and a random coin space $\mathcal{R}$ is defined by the following three algorithms. Let k denote the security parameter.

Key Generation: On input 1^k, the key generation algorithm $\mathsf{Gen}(1^k)$ produces a public/secret key pair $(\mathsf{pk}, \mathsf{sk})$.

Encryption: Given a public key pk, a plaintext $m \in \mathcal{M}$, and a random string $r \in \mathcal{R}$, the encryption algorithm $\mathsf{Enc}_{\mathsf{pk}}(m; r)$ outputs a ciphertext c corresponding to the plaintext m.

Decryption: Given a secret key sk and ciphertext c, the decryption algorithm $\mathsf{Dec}_{\mathsf{sk}}(c)$ outputs the plaintext $m \in \mathcal{M}$ or the special symbol $\perp \notin \mathcal{M}$ corresponding to the ciphertext c.

We require the perfect completeness, that is, for every $(\mathsf{pk}, \mathsf{sk})$ generated by $\mathsf{Gen}(1^k)$, every plaintext $m \in \mathcal{M}$, and every random string $r \in \mathcal{R}$, it should be satisfied that $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m; r)) = m$.

We only consider three standard security notions for public-key encryption schemes, the one-wayness against chosen-plaintext attack (OW-CPA), the indistinguishability against chosen-plaintext attack (IND-CPA), and the indistinguishability against adaptive chosen-ciphertext attack (IND-CCA2).

For $\gamma = \gamma(k)$, we say $\mathcal{PKE}$ is γ-uniform if for any key pair $(\mathsf{pk}, \mathsf{sk})$ generated by $\mathsf{Gen}(1^k)$, any $m \in \mathcal{M}$, and $c \in \{0,1\}^*$, we have $\Pr_{r \leftarrow \mathcal{R}}[c = \mathsf{Enc}_{\mathsf{pk}}(m; r)] \leq \gamma$. There exists a OW-CPA public-key encryption scheme with γ-uniformity (e.g., the ElGamal encryption scheme).

Brief review for FO. Fujisaki and Okamoto proposed a conversion, called the Fujisaki-Okamoto (FO) conversion, to obtain highly secure public-key encryption schemes in the ROM [20]. Since the standard one-time pad satisfies the requirement of the FO conversion, we fix the one-time pad as the symmetric-key encryption scheme used in the FO conversion for simplicity.

Let $\mathcal{PKE}$ be a OW-CPA secure and γ-uniform public-key encryption scheme over a plaintext space $\mathcal{M}$ and a randomness space $\mathcal{R}$. Then the FO conversion converts $\mathcal{PKE}$ to an IND-CCA2 secure one $\mathcal{PKE}' = \mathrm{FO}(\mathcal{PKE})$ over a plaintext space $\mathcal{M}' = \{0,1\}^{k'}$ and a randomness space $\mathcal{R}' = \mathcal{M}$, where k' denotes the length of plaintexts, which is polynomially related to the security parameter k. The encryption procedure of $\mathcal{PKE}'$ is given as follows: For a plaintext $m \in \mathcal{M}' = \{0,1\}^{k'}$ and a random string $r \in \mathcal{R}' = \mathcal{M}$, the ciphertext is

$$(c_1, c_2) = (\mathsf{Enc}_{\mathsf{pk}}(r; H(m, r)), G(r) \oplus m),$$

where $H : \{0,1\}^{k'} \times \mathcal{M} \to \mathcal{R}$ and $G : \mathcal{M} \to \{0,1\}^{k'}$ are hash functions modeled as the random oracles. The decryption procedure is given as follows: For a given ciphertext (c_1, c_2), decrypt c_1 by sk and obtain r. Then, extract m by $c_2 \oplus G(r)$ and verify $c_1 = \mathsf{Enc}_{\mathsf{pk}}(r; H(m, r))$. If not output $\perp$. Roughly speaking, $H(m, r)$ ensures that if a ciphertext (c_1, c_2) is valid then the encryptor producing (c_1, c_2) knows corresponding m and r.

3.1 The First Variant dFO

We introduce the first artificial variant dFO and show that dFO is secure in the ROM, but not secure in general in the CT-ROM.

The variant dFO converts a public-key encryption scheme $\mathcal{PKE}$ (with the one-time pad) to another public-key encryption scheme $\mathcal{PKE}' = \mathrm{dFO}(\mathcal{PKE})$ similarly to FO. The encryption procedure of $\mathcal{PKE}'$ is defined as follows. For a plaintext $m \in \mathcal{M}' = \{0,1\}^{k'}$ and a random string $r \in \mathcal{R}' = \mathcal{M}$, the ciphertext of $\mathcal{PKE}'$ is

$$(c_1, c_2) = (\mathsf{Enc}_{\mathsf{pk}}(r; H(F(m), r)), G(r) \oplus m),$$

where $F : \{0,1\}^{k'} \to \mathcal{P}$, $G : \mathcal{M} \to \{0,1\}^{k'}$, and $H : \mathcal{P} \times \mathcal{M} \to \mathcal{R}$, for an appropriate set $\mathcal{P}$, are hash functions modeled as the random oracle.

The idea to weaken the conversion is summarized as follows: Recall that $H(m, r)$ in the FO conversion can be considered as encryptor's signature (or a proof of knowledge) on m and r. To make it vulnerable by a collision, we introduce a new random oracle F and replace $H(m, r)$ with $H(F(m), r)$. The replacement does not harm the security in the random oracle model, while it can be exploited by the presence of the collision oracle $\mathcal{CO}^F$.

Formally, we have following theorems on the (in)security. We omit the proof of Theorem 3, which is similar to the original one.

Theorem 3. *Assume that $\mathcal{PKE}$ is a* $\mathsf{OW\text{-}CPA}$ *secure and γ-uniform public-key encryption scheme for some negligible γ. Then, $\mathcal{PKE}' = \mathrm{dFO}(\mathcal{PKE})$ is* $\mathsf{IND\text{-}CCA2}$ *secure in the* ROM *if $\#\mathcal{P} = 2^{\omega(\log k)}$.*

Theorem 4. *Let $\mathcal{PKE}$ be a public-key encryption scheme. If $\#\mathcal{P} \leq 2^{k'}$ then $\mathcal{PKE}' = \mathrm{dFO}(\mathcal{PKE})$ is not* $\mathsf{IND\text{-}CCA2}$ *secure in the* $\mathsf{CT\text{-}ROM}$.

Proof. We construct the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that breaks the $\mathsf{IND\text{-}CCA2}$ security of $\mathcal{PKE}'$, which exploits the collision oracle $\mathcal{CO}^F$ of F.

The adversary $\mathcal{A}_1$, on input pk, first queries to $\mathcal{CO}^F$. If the answer is $\perp$, then the adversary flips a random fair coin b', outputs b', and halts. Otherwise, it obtains a collision (m_1, m_2) of F and outputs it as a challenge. The adversary $\mathcal{A}_2$ receives the target ciphertext $(c_1^*, c_2^*) = (\mathsf{Enc}_{\mathsf{pk}}(r; H(F(m_b), r)), G(r) \oplus m_b)$ for some $r \in \mathcal{R}'$. It queries $(c_1', c_2') = (c_1^*, c_2^* \oplus m_0 \oplus m_1)$ to the decryption oracle and obtains m_{1-b}, since

$$\begin{aligned} c_1' &= \mathsf{Enc}_{\mathsf{pk}}(r; H(F(m_0), r)) = \mathsf{Enc}_{\mathsf{pk}}(r; H(F(m_1), r)),\\ c_2' &= G(r) \oplus m_b \oplus m_0 \oplus m_1 = G(r) \oplus m_{1-b}. \end{aligned}$$

Hence, the adversary can answer $b' = b$ correctly.

Finally, we upper-bound the probability that the collision oracle outputs $\perp$, which stems from the definition of the collision oracle. The probability is bounded by $(1 - 1/\#\mathcal{P})^{2^{k'}-1} \leq \exp(-(2^{k'} - 1)/\#\mathcal{P}) \leq 1/\sqrt{e}$. This completes the proof. □

3.2 The Second Variant wFO

We next introduce the second artificial variant wFO and show that the obtained scheme by wFO is secure in the $\mathsf{CT\text{-}ROM}$, however not generally secure in the $\mathsf{SPT\text{-}ROM}$.

The encryption procedure of $\mathcal{PKE}' = \mathrm{wFO}(\mathcal{PKE})$ is given as follows. For a plaintext $m \in \mathcal{M}' = \{0,1\}^{k'}$ and random strings $(r, s) \in \mathcal{R}' = \mathcal{M} \times \mathcal{S}$, the ciphertext of $\mathcal{PKE}'$ is

$$(c_1, c_2, c_3) = (\mathsf{Enc}_{\mathsf{pk}}(r; H(F(m, s), r)), G(r) \oplus m, s),$$

where $F : \{0,1\}^{k'} \times \mathcal{S} \to \mathcal{P}$, $G : \mathcal{M} \to \{0,1\}^{k'}$, and $H : \mathcal{P} \times \mathcal{M} \to \mathcal{R}$ are hash functions modeled as the random oracles.

Notice that $(H(F(m, s), r), s)$ is a proof of knowledge on (m, r, s) which resists a collision on F however is vulnerable by a second-preimage attack against F as in Numayama et al. [18].

We can show that the obtained scheme is IND-CCA2 secure in the CT-ROM by using Lemma 2.

Theorem 5. *Suppose that $\mathcal{PKE}$ is a* OW-CPA *secure and γ-uniform public-key encryption scheme for some negligible γ. Then, $\mathcal{PKE}' = \mathrm{wFO}(\mathcal{PKE})$ is* IND-CCA2 *secure in the* CT-ROM *if $\#\mathcal{P}^{-1}$ and $\#\mathcal{S}^{-1}$ are negligible in k.*

However, its security is broken under the presence of the second-preimage oracle for F.

Theorem 6. *Let $\mathcal{PKE}$ be a public-key encryption. If $\#\mathcal{P} \leq 2^{k'} \cdot \#\mathcal{S}$, then the scheme $\mathcal{PKE}' = \mathrm{wFO}(\mathcal{PKE})$ is not* IND-CCA2 *secure in the* SPT-ROM.

Proof. We construct the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that exploits the second-preimage oracle $\mathcal{SPO}^F$ associated to F. The adversary $\mathcal{A}_1$ chooses random distinct plaintexts m_0 and m_1 and queries them to the challenger. The challenger responses

$$(c_1^*, c_2^*, c_3^*) = (\mathsf{Enc}_{\mathsf{pk}}(r; H(F(m_b, s), r)), G(r) \oplus m_b, s).$$

Receiving (c_1^*, c_2^*, c_3^*), the adversary $\mathcal{A}_2$ queries (m_0, s) to the second-preimage oracle $\mathcal{SPO}^F$. If it receives $\perp$ from the second-preimage oracle, then it flips a random fair coin b', outputs b', and halts. Otherwise, it obtains $(m', s') \neq (m_0, s)$ such that $F(m_0, s) = F(m', s')$. So, the adversary queries

$$(c_1', c_2', c_3') = (c_1^*, c_2^* \oplus m_0 \oplus m', s')$$

to the decryption oracle. Notice that, if (c_1^*, c_2^*, c_3^*) is the valid ciphertext of m_0, then we have

$$\begin{aligned} c_1' &= \mathsf{Enc}_{\mathsf{pk}}(r; H(F(m_0, s), r)) = \mathsf{Enc}_{\mathsf{pk}}(r; H(F(m', s'), r)), \\ c_2' &= G(r) \oplus m_0 \oplus m_0 \oplus m' = G(r) \oplus m', \\ c_3' &= s', \end{aligned}$$

and (c_1', c_2', c_3') is a valid ciphertext for m'. On the other hand, if the ciphertext is the encryption of m_1, we have

$$(c_1', c_2', c_3') = (\mathsf{Enc}_{\mathsf{pk}}(r; H(F(m_1, s), r)), G(r) \oplus m_1 \oplus m_0 \oplus m', s').$$

Thus, if $f = F(m_1, s)$ is equal to $F(m_1 \oplus m_0 \oplus m', s')$ the decryption oracle returns $m_1 \oplus m_0 \oplus m' (\neq m')$. Otherwise, the decryption oracle returns $\perp$.

Thus, if the answer is m', then the adversary concludes that (c_1^*, c_2^*, c_3^*) is the ciphertext of m_0, that is, it outputs $b' = 0$. Otherwise, the adversary concludes that it is the ciphertext of m_1, that is, it outputs $b' = 1$. Therefore, $\mathcal{A}$ can output the correct answer unless $\mathcal{A}$ receives $\perp$ from the second-preimage oracle.

We finally bound the probability that the oracle outputs $\perp$. It is bounded by $(1 - 1/\#\mathcal{P})^{2^{k'} \cdot \#\mathcal{S} - 1} \leq \exp(-(2^{k'} \cdot \#\mathcal{S} - 1)/\#\mathcal{P}) \leq 1/\sqrt{e}$ as required. This completes the proof. □

3.3 The Original Fujisaki-Okamoto Conversion

We next show that the obtained scheme by the conversion FO with the one-time pad is secure in the SPT-ROM, but not secure in the FPT-ROM in some parameter setting.

Let $G : \mathcal{M} \to \{0,1\}^{k'}$ and $H : \{0,1\}^{k'} \times \mathcal{M} \to \mathcal{R}$ be hash functions modeled as the random oracles. Recall the encryption procedure of $\mathcal{PKE}' = \mathrm{FO}(\mathcal{PKE})$. For a plaintext $m \in \mathcal{M}' = \{0,1\}^{k'}$ and a random string $r \in \mathcal{R}' = \mathcal{M}$, the ciphertext is $(\mathsf{Enc}_{\mathsf{pk}}(r; H(m,r)), G(r) \oplus m)$.

Modifying the existing proofs, we can show the scheme is secure in the SPT-ROM using Lemma 2.

Theorem 7. *Suppose that $\mathcal{PKE}$ is* OW-CPA *secure and γ-uniform for some negligible γ. Then, $\mathcal{PKE}' = \mathrm{FO}(\mathcal{PKE})$ is* IND-CCA2 *secure in the* SPT-ROM.

However, the presence of the first-preimage oracle for G violates the IND-CPA security of $\mathcal{PKE}'$ in some parameter settings. Note that if m is $0^{k'}$, the second component of the ciphertext is $G(r)$, which is vulnerable the first-preimage oracle of G.

Theorem 8. *Let $C = \#\mathcal{M}/2^{k'}$. Assume that $C = k^{O(1)}$. Then, $\mathcal{PKE}' = \mathrm{FO}(\mathcal{PKE})$ is not* IND-CPA *secure in the* FPT-ROM.

Proof. We prove the theorem by constructing the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ which exploits the first-preimage oracle of G, $\mathcal{FPO}^G$. The adversary $\mathcal{A}_1$, on input pk, queries $m_0 = 0^{k'}$ and $m_1 = 1^{k'}$ to the challenger. The adversary $\mathcal{A}_2$, on input the target ciphertext (c_1^*, c_2^*), queries c_2^* to the first-preimage oracle of G. If it obtains $\tilde{r}$, it checks that $c_1 = \mathsf{Enc}_{\mathsf{pk}}(\tilde{r}; H(0^{k'}, \tilde{r}))$. If the check passes, the adversary outputs $b' = 0$. Otherwise, it flips a random fair coin b', outputs b', and halts.

It is obvious that if $b = 0$ and $\tilde{r} = r$, the adversary answers correctly, that is, it outputs $b' = b$. If $b = 1$, the preimage of the query $G(r) \oplus 1^{k'}$ never equals to r since $G(r) \neq G(r) \oplus 1^{k'}$. Hence, the adversary's check fails if $b = 1$.

We estimate the probability that the adversary wins. By Lemma 1, with probability at least $1 - 2^{-2k'}$, there is no preimage of size larger than L, where if $C \geq 1$ then $L = 5Ck' \ln 2/(\ln k' + \ln\ln 2) \leq 4Ck'/\ln k'$ and otherwise $L = 5k' \ln 2/(\ln k' + \ln\ln 2) \leq 4k'/\ln k'$ for all sufficiently large k'.

Let Good denote the event that $r \leftarrow \mathcal{FPO}_G(G(r))$. We then have $\Pr[\mathsf{Good}] \geq (1 - 2^{-2k'})/L$. Hence, we obtain that

$$\begin{aligned}
\Pr[b' = b] &= \Pr[b' = 0 \mid b = 0 \wedge \mathsf{Good}] \Pr[b = 0 \wedge \mathsf{Good}] \\
&\quad + \Pr[b' = 0 \mid b = 0 \wedge \neg\mathsf{Good}] \Pr[b = 0 \wedge \neg\mathsf{Good}] \\
&\quad + \Pr[b' = 1 \mid b = 1] \Pr[b = 1] \\
&= 1 \cdot \frac{1}{2} \cdot \Pr[\mathsf{Good}] + \frac{1}{2} \cdot \frac{1}{2} \cdot (1 - \Pr[\mathsf{Good}]) + \frac{1}{2} \cdot \frac{1}{2} \\
&= \frac{1}{2} + \frac{1}{4} \Pr[\mathsf{Good}] \geq \frac{1}{2} + \frac{1 - 2^{-2k'}}{4L}.
\end{aligned}$$

and $4L$ is a polynomial in the security parameter k. This completes the proof. □

As shown above, the FO conversion is not secure in the FPT-ROM, but there is a way to modify it so as to maintain the security in the FPT-ROM. Naito, Wang, and Ohta

```
Key Generation
  Input: $1^k$
    1: $(f_{\mathsf{pk}}, g_{\mathsf{sk}}) \leftarrow F$
  Output: $(f_{\mathsf{pk}}, g_{\mathsf{sk}})$
```

```
Encryption
  Input: $m \in \{0,1\}^{k-k_0-k_1}$, $f_{\mathsf{pk}}$
    1: $r \leftarrow \{0,1\}^{k_0}$
    2: $s \leftarrow (m \,\|\, 0^{k_1}) \oplus G(r)$
    3: $t \leftarrow H(s) \oplus r$
    4: $c \leftarrow f_{\mathsf{pk}}(s \,\|\, t)$
  Output: $c$
```

```
Decryption
  Input: $c$, $g_{\mathsf{sk}}$
    1: $s \,\|\, t \leftarrow g_{\mathsf{sk}}(c)$
    2: $r \leftarrow t \oplus H(s)$
    3: $M \leftarrow s \oplus G(r)$
    5: If $M = m \,\|\, 0^{k_1}$ set $o \leftarrow m$
    6: Otherwise set $o \leftarrow \perp$
  Output: $o$
```

Fig. 1. OAEP

proposed the conversion method that converts a cryptosystem secure in the ROM to that secure even in the FPT-ROM [29]. In the case of the FO conversion, the public key is (pk, c), where $c \leftarrow \{0,1\}^k$, and the ciphertext is

$$(c_1, c_2) = (\mathsf{Enc}_{\mathsf{pk}}(r; H(c, m, r)), G(c, r) \oplus m),$$

where the domains of H and G are modified. Intuitively, this change makes the first-preimage oracles, $\mathcal{FPO}^H$ and $\mathcal{FPO}^G$, useless.

3.4 OAEP

We finally focus on the OAEP and present its IND-CCA2 security in the FPT-ROM. For the security parameter k, let k_0 and k_1 be functions in k, where $k_0 < k - k_0$. Let F be a family of partial-domain one-way trapdoor permutations of a domain $\{0,1\}^{k-k_0} \times \{0,1\}^{k_0}$. (See [30] for the definition of the partial-domain one-wayness.) Furthermore, let G and H be hash functions such that $G : \{0,1\}^{k_0} \to \{0,1\}^{k-k_0}$ and $H : \{0,1\}^{k-k_0} \to \{0,1\}^{k_0}$. Then, the OAEP encryption scheme based on F is described in Fig. 1.

We obtain the following theorem that states the security of the OAEP encryption scheme in the FPT-ROM.

Theorem 9. *Let F be a family of partial-domain one-way trapdoor permutations. Then, the OAEP encryption scheme based on F is* IND-CCA2 *secure in the* FPT-ROM.

We here only give the sketch of the security proof.

Proof (Sketch). As in the proof of Fujisaki et al. [30], we prove the security by defining a sequence of games and bounding the advantages of the adversary among the games. The games are the almost same as the original ones in [30]. However, we need to pay attention to the following two points. First, as mentioned, we no longer have the uniformity of the ROM because of the first-preimage oracle. Second, the adversary can make use of the first-preimage oracle. These points make the security proofs difficult.

In order to observe the difference between the security proofs in the FPT-ROM and ROM, let us consider the following two games. We will describe the sequence of the games in the full version.

– Game_1: The challenger generates a pair of keys $(f_{\mathsf{pk}}, g_{\mathsf{sk}})$ by using the key-generation algorithm. It next produces $r^+ \leftarrow \{0,1\}^{k_0}$ and obtains $g^+ \leftarrow \mathsf{RO}_G(r^+)$. In generation of the target ciphertext, the challenger generates the random string r^+. The target ciphertext y^* is generated as follows:

$$r^* \leftarrow r^+, \qquad s^* \leftarrow (m_b \,\|\, 0^{k_1}) \oplus g^+, \qquad t^* \leftarrow r^* \oplus \mathsf{RO}_H(s^*),$$
$$x^* \leftarrow (s^*, t^*), \qquad y^* \leftarrow f_{\mathsf{pk}}(x^*).$$

The ciphertext y^* is given to $\mathcal{A}$. Finally, the adversary $\mathcal{A}$ outputs a bit b'.

– Game_2: We modify the above game, by changing the rule for generation of g^+. That is, g^+ is not obtained by the query of the random oracle, but obtained by choosing from $\{0,1\}^{k-k_0}$ uniformly at random. Notice that (r^+, g^+) is not contained in the table $\mathbb{T}_G$.

Let AskG be the event that r^+ is queried to RO_G. The original proof in the ROM showed that, if the value r^+ is not queried to RO_G, the Game_1 and Game_2 are identical.

On the other hand, in our case in the FPT-ROM, even if the event AskG does not occur, that is, the value r^+ is not queried, we cannot say that Game_1 and Game_2 are identical. Notice that the adversary would distinguish the games by querying g^+ to FPO_G, which leads to a contradiction to the partial-domain one-wayness in the final game. The value g^+ must have the preimage r^+ in Game_1 since (r^+, g^+) is contained in the table $\mathbb{T}_G$. In contrast, the value g^+ has no preimages in Game_2 with high probability if $k - k_0$ is much larger than k_0 since (r^+, g^+) is not inserted in the table $\mathbb{T}_G$ and $\perp \leftarrow \mathsf{FPO}_G(g^+)$ with high probability. We must take care of this event AskG^-. Additionally, it would distinguish between Game_1 and Game_2 by querying $(m_{1-b}\|0^{k_1})\oplus s^*$ to FPO_G, which also leads to contradiction to the partial-domain one-wayness in the final game. This event is denoted by $\mathsf{AskG}^\diamond$. Notice that, conditioned on the above events, AskG, AskG^-, and $\mathsf{AskG}^\diamond$, do not occur, g^+ is almost perfectly uniform in Game_1 by Lemma 2. Hence, we can show two games Game_1 and Game_2 are statistically close if the events do not occur.

By carefully applying similar arguments, we can show the IND-CCA2 security for the OAEP encryption scheme in FPT-ROM. □

4 Future Work

It should be noted that our WROMs are based on a simplified variant, which Numayama et al. [18] and Pasini and Vaudenay [17] also adopted, of the original WROMs of Liskov [15].

The original WROMs consists of the ideal compression function $h : \{0,1\}^{k+k'} \rightarrow \{0,1\}^k$ of *fixed input length* and the first-preimage oracle. Then, he discussed the security of the *flexible input-length* hash functions $H^h : \{0,1\}^* \rightarrow \{0,1\}^k$ employing h as the component in the context of indifferentiability [31]. A random oracle H is often instantiated by employing a compression h. (See, e.g., the survey in [8, Section 2].) Therefore, his work reflects the attacks against the compression function of MD5 and SHA-1 rather than the construction H.

On the contrary, we (and similarly [18,17]) discussed the *monolithic* random oracle H and the additional oracles associated with H. Hence, our model has a gap from such a realistic instantiation of the random oracle in some sense. We leave filling this gap as future work.

Except for the FO conversion, there are several conversion methods in the ROM, such as REACT [32] and GEM [33]. It would also be interesting as future work to examine the security of these conversion methods in the WROMs.

Acknowledgements

We thank anonymous reviewers for their helpful comments. This research was supported in part by NTT Information Sharing Platform Laboratories, JSPS Global COE program “Computationalism as Foundation for the Sciences,” KAKENHI 18300002, KAKENHI 19-55201, and the Japan Science and Technology Agency, Strategic Japanese-French Cooperative Program “Quantum Computer: Theory and Feasibility.”

References

1. Bellare, M., Rogaway, P.: Random oracle are practical: A paradigm for designing efficient protocols. In: CCS 1993, pp. 62–73. ACM, New York (1993)
2. Rivest, R.L.: The MD5 message-digest algorithm. Internet Request for Comments, RFC 1321 (April 1992)
3. National Institute of Standards and Technology: Secure hash standard. FIPS 180-2 (August 2002)
4. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. Journal of the ACM 51(4), 557–594 (2004); Preliminary version in STOC 1998 (1998)
5. Goldwasser, S., Kalai, Y.T.: On the (in)security of the Fiat-Shamir paradigm. In: FOCS 2003, pp. 102–113. IEEE Computer Society, Los Alamitos (2003)
6. Bellare, M., Boldyreva, A., Palacio, A.: An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 171–188. Springer, Heidelberg (2004)
7. Bellare, M., Rogaway, P.: The exact security of digital signatures – how to sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
8. Leurent, G., Nguyen, P.Q.: How risky is the random-oracle model? In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 445–464. Springer, Heidelberg (2009), http://eprint.iacr.org/2008/441
9. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)
10. Wang, X., Yin, Y.L., Yu, H.: Finding collisions in the full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
11. Aoki, K., Sasaki, Y.: Preimage attacks on one-block MD4, 63-step MD5 and more. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2008. LNCS, vol. 5381, pp. 103–119. Springer, Heidelberg (2008)
12. Nielsen, J.B.N.: 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)
13. Unruh, D.: Random oracles and auxiliary input. In: [34], pp. 205–223
14. Bellare, M., Rogaway, P.: Optimal asymmetric encryption. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995)

15. Liskov, M.: Constructing an ideal hash function from weak ideal compression functions. In: Biham, E., Youssef, A.M. (eds.) SAC 2006. LNCS, vol. 4356, pp. 358–375. Springer, Heidelberg (2007)
16. Hoch, J.J., Shamir, A.: On the strength of the concatenated hash combiner when all the hash functions are weak. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 616–630. Springer, Heidelberg (2008)
17. Pasini, S., Vaudenay, S.: Hash-and-sign with weak hashing made secure. In: Pieprzyk, J., Ghodosi, H., Dawson, E. (eds.) ACISP 2007. LNCS, vol. 4586, pp. 338–354. Springer, Heidelberg (2007)
18. Numayama, A., Isshiki, T., Tanaka, K.: Security of digital signature schemes in weakened random oracle models. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 268–287. Springer, Heidelberg (2008)
19. Fischlin, M., Lehmann, A.: Security-amplifying combiners for collision-resistant hash functions. In: [34], pp. 224–243
20. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (1999)
21. Kiltz, E., Pietrzak, K.: On the security of padding-based encryption schemes (or: Why we cannot prove OAEP secure in the standard model). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 389–406. Springer, Heidelberg (2009)
22. Devroye, L.D.: Non-Uniform Random Variate Generation. Springer, Heidelberg (1986)
23. Ahrens, J.H., Dieter, U.: Computer methods for sampling from Gamma, Beta, Poisson and Binomial distributions. Computing 12(3), 223–246 (1974)
24. Ahrens, J.H., Dieter, U.: Sampling from Binomial and Poisson distributions: A method with bounded computation times. Computing 25(3), 193–208 (1980)
25. Relles, D.A.: A simple algorithm for generating Binomial random variables when *N* is large. American Statistical Association 67(339), 612–613 (1972)
26. Kawachi, A., Numayama, A., Tanaka, K., Xagawa, K.: Security of encryption schemes in weakened random oracle models. Cryptology ePrint Archive, Report 2010/122 (2010)
27. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, Cambridge (1995)
28. Katz, J., Lindell, Y.: Introduction to Modern Cryptography. Chapman & Hall/CRC, Boca Raton (2007)
29. Naito, Y., Wang, L., Ohta, K.: How to construct cryptosystems and hash functions in weakenend random oracle models. Cryptology ePrint Archive, Report 2009/550 (2009)
30. Fujisaki, E., Okamoto, T., Pointcheval, D., Stern, J.: RSA-OAEP is secure under the RSA assumption. Journal of Cryptology 17(2), 81–104 (2004)
31. Maurer, U.M., Renner, R., Holenstein, C.: Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (2004)
32. Okamoto, T., Pointcheval, D.: REACT: Rapid enhanced-security asymmetric cryptosystem transform. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 159–175. Springer, Heidelberg (2001)
33. Coron, J.S., Handschuh, H., Joye, M., Paillier, P., Pointcheval, D., Tymen, C.: Gem: A Generic chosen-ciphertext secure Encryption Method. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 175–184. Springer, Heidelberg (2002)
34. Menezes, A. (ed.): CRYPTO 2007. LNCS, vol. 4622. Springer, Heidelberg (2007)

Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes

Nigel P. Smart[1] and Frederik Vercauteren[2]

[1] Dept. Computer Science,
University of Bristol,
Merchant Venturers Building,
Woodland Road,
Bristol, BS8 1UB,
United Kingdom
nigel@cs.bris.ac.uk

[2] COSIC - Electrical Engineering,
Katholieke Universiteit Leuven,
Kasteelpark Arenberg 10,
B-3001 Heverlee,
Belgium
fvercaut@esat.kuleuven.ac.be

Abstract. We present a fully homomorphic encryption scheme which has both relatively small key and ciphertext size. Our construction follows that of Gentry by producing a fully homomorphic scheme from a "somewhat" homomorphic scheme. For the somewhat homomorphic scheme the public and private keys consist of two large integers (one of which is shared by both the public and private key) and the ciphertext consists of one large integer. As such, our scheme has smaller message expansion and key size than Gentry's original scheme. In addition, our proposal allows efficient fully homomorphic encryption over any field of characteristic two.

1 Introduction

A fully homomorphic public key encryption scheme has been a "holy grail" of cryptography for a very long time. In the last year this problem has been solved by Gentry [7,8], by using properties of ideal lattices. Various cryptographic schemes make use of lattices, sometimes just to argue about their security (such as NTRU [11]), in other cases lattices are vital to understand the workings of the scheme algorithms (such as [9]). Gentry's fully homomorphic scheme falls into the latter category. In this paper we present a fully homomorphic scheme which can be described using the elementary theory of algebraic number fields, and hence we do not require lattices to understand its encryption and decryption operations. However, our scheme does fall into the category of schemes whose best known attack is based on lattices.

At a high level our scheme is very simple, and is mainly parametrized by an integer N (there are other parameters which are less important). The public key consists of a prime p and an integer α modulo p. The private key consists of either an integer z (if we are encrypting bits), or an integer polynomial $Z(x)$ of degree $N-1$ (if we are encrypting general binary polynomials of degree $N-1$). To encrypt a message one encodes the message as a binary polynomial, then one randomizes the message by adding on two times a small random polynomial. To obtain the ciphertext, the resulting polynomial is evaluated at α modulo p. As such, the ciphertext is simply an integer modulo p (irrespective of whether we are encrypting bits or binary polynomials of degree $N-1$).

To decrypt in the case where we know the message is a single bit, we multiply the ciphertext by z and divide by p. We then round this rational number to the nearest integer value, and subtract the result from the ciphertext. The plaintext is then recovered by reducing this intermediate result modulo 2. When we are decrypting a binary polynomial we follow the same procedure, but this time we multiply by the polynomial $Z(x)$ and divide by p, to obtain a rational polynomial. Rounding the coefficients of this polynomial to the nearest integer, subtracting from the original ciphertext, and reducing modulo two will result again in recovering the plaintext.

2 Preliminaries

2.1 Notation

Given a polynomial $g(x) = \sum_{i=0}^{t} g_i x^i \in \mathbb{Q}[x]$, we define the 2-norm and ∞-norm as

$$\|g(x)\|_2 = \sqrt{\sum_{i=0}^{t} g_i^2} \quad \text{and} \quad \|g(x)\|_\infty = \max_{i=0,\ldots,t} |g_i| \,.$$

For a positive value r, we define two corresponding types of "ball" centered at the origin:

$$\mathcal{B}_{2,N}(r) = \left\{ \sum_{i=0}^{N-1} a_i x^i : \sum_{i=0}^{N-1} a_i^2 \le r^2 \right\},$$

$$\mathcal{B}_{\infty,N}(r) = \left\{ \sum_{i=0}^{N-1} a_i x^i : -r \le a_i \le r \right\}.$$

We have the usual inclusions $\mathcal{B}_{2,N}(r) \subset \mathcal{B}_{\infty,N}(r)$ and $\mathcal{B}_{\infty,N}(r) \subset \mathcal{B}_{2,N}(\sqrt{N} \cdot r)$. We also define the following half-ball

$$\mathcal{B}_{\infty,N}^{+}(r) = \left\{ \sum_{i=0}^{N-1} a_i x^i : 0 \le a_i \le r \right\}.$$

All reductions in this paper modulo an odd integer m are defined to result in a value in the range $[-(m-1)/2, \ldots, (m-1)/2]$. The notation $a \leftarrow b$, means assign the value on the left to the value on the right. Whereas $a \leftarrow_R A$ where A is a set, means select a from the set A using a uniform distribution.

2.2 Ideals in Number Fields

Since the underlying workings of our scheme are based on prime ideals in a number field, we first recap on some basic properties. See [4] for an introduction to the elementary computational number theory needed.

Let K be a number field $\mathbb{Q}(\theta)$ where θ is a root of a monic irreducible polynomial $F(x) \in \mathbb{Z}[x]$ of degree N. Consider the equation order $\mathbb{Z}[\theta]$ inside the ring of integers $\mathcal{O}_K$. For our parameter choices we typically have $\mathcal{O}_K = \mathbb{Z}[\theta]$, but this need not be the case in general. Our scheme works with ideals in $\mathbb{Z}[\theta]$ that are assumed coprime with the index $[\mathcal{O}_K : \mathbb{Z}[\theta]]$, so there is little difference with working in $\mathcal{O}_K$. These ideals can be represented in one of two ways, either by an N-dimensional $\mathbb{Z}$-basis or as a two element $\mathbb{Z}[\theta]$-basis. When presenting an ideal $\mathfrak{a}$ as an N-dimensional $\mathbb{Z}$ basis we give N elements $\gamma_1, \ldots, \gamma_N \in \mathbb{Z}[\theta]$, and every element in $\mathfrak{a}$ is represented by the $\mathbb{Z}$-module generated by $\gamma_1, \ldots, \gamma_N$. It is common practice to present this basis as an $n \times n$-matrix. The matrix is then set to be $(\gamma_{i,j})$, where we set $\gamma_i = \sum_{j=0}^{N-1} \gamma_{i,j}\theta^j$, i.e. we take a row oriented formulation. Taking the Hermite Normal Form (HNF) of this basis will produce a lower triangular basis in which the leading diagonal $(d_1, \ldots, d_N)$ satisfies $d_{i+1}|d_i$. Note that this last property of the HNF of a basis only follows for matrices corresponding to ideals [5] (who use a different orientation).

However, every such ideal can also be represented by a $\mathbb{Z}[\theta]$-basis given by two elements, $\langle \delta_1, \delta_2 \rangle$. In particular one can always select δ_1 to be an integer. For ideals lying above a rational prime p, it is very easy to write down a two element representation of an ideal. If we factor $F(x)$ modulo p into irreducible polynomials

$$F(x) = \prod_{i=1}^{t} F_i(x)^{e_i} \pmod{p}$$

then, for p not dividing $[\mathcal{O}_K : \mathbb{Z}[\theta]]$, the prime ideals dividing $p\mathbb{Z}[\theta]$ are given by the two element representation

$$\mathfrak{p}_i = \langle p, F_i(\theta) \rangle .$$

We define the residue degree of $\mathfrak{p}_i$ to be equal to the degree d_i of the polynomial $F_i(x)$. Reduction modulo $\mathfrak{p}_i$ produces a homomorphism

$$\iota_{\mathfrak{p}_i} : \mathbb{Z}[\theta] \longrightarrow \mathbb{F}_{p^{d_i}} .$$

We will be particularly interested in prime ideals of residue degree one. These can be represented as a two element representation by $\langle p, \theta - \alpha \rangle$ where p is the norm of the ideal and α is a root of $F(x)$ modulo p. If $\chi \in \mathbb{Z}[\theta]$ is given by $\chi = \sum_{i=0}^{N-1} c_i\theta^i$ then the homomorphism $\iota_{\mathfrak{p}}$ simply corresponds to evaluation of the polynomial $\chi(\theta)$ in α modulo p.

Given a prime ideal of the form $\langle p, \theta - \alpha \rangle$, the corresponding HNF representation is very simple to construct, and is closely related to the two element representation, as we shall now show. We need to row reduce the $2N \times N$ matrix

$$\begin{pmatrix} p & & & & & \\ & p & & 0 & & \\ & & & \ddots & & \\ & 0 & & & & \\ & & & & & p \\ -\alpha & 1 & & & 0 & \\ & -\alpha & 1 & & & \\ 0 & & & \ddots & \ddots & \\ & & & & -\alpha & 1 \\ -F_0 & -F_1 & -F_2 & \dots & -F_{N-2} & -F_{N-1}-\alpha \end{pmatrix}$$

where $F(x) = \sum_{i=0}^{N} F_i x^i$. It is not hard to see that the HNF of the above matrix is then given by

$$H = \begin{pmatrix} p & & & 0 \\ -\alpha & 1 & & \\ -\alpha^2 & & 1 & \\ \vdots & & & \ddots \\ -\alpha^{N-1} & & 0 & & 1 \end{pmatrix},$$

where all the integers in the first column, in rows two and onward, are taken modulo p.

Recall that an ideal is called principal if it is generated by one element, i.e. we can write $\mathfrak{p} = \langle \gamma \rangle = \gamma \cdot \mathbb{Z}[\theta]$. Note that given an HNF or two-element representation of an ideal, determining whether it is principal, and finding a generator is considered to be a hard problem for growing N. Indeed the best known algorithms (which are essentially equivalent to finding the class and unit group of a number field) run in exponential time in the degree of the field. For fixed degree they run in sub-exponential time in the discriminant [2]. In addition the generator of a principal ideal output by these algorithms will be very large. Indeed, this generator will typically be so large that writing it down as a polynomial in θ may itself take exponential time [14]. Thus finding a small generator of a principal ideal is possibly an even harder problem. Quantumly finding a generator of a principal ideal is relatively easy [10], however writing down a small generator is not known to be easy.

3 Our Somewhat Homomorphic Scheme

In this section we present our somewhat homomorphic scheme and analyze for which parameter sets decryption works. To simplify the presentation we present the scheme at this point as one which just encrypts elements in $\mathcal{P} = \{0, 1\}$.

3.1 The Scheme

A somewhat homomorphic encryption scheme consists of five algorithms: {KeyGen, Encrypt, Decrypt, Add, Mult}. We shall describe each in turn; notice that the most complex phase is that of KeyGen. The scheme is parametrized by three values (N, η, μ). A typical set of parameters would be $(N, 2^{\sqrt{N}}, \sqrt{N})$. Later we shall return to discussing the effects of the sizes of these values on the security level λ and performance of the scheme.

KeyGen():

- Set the plaintext space to be $\mathcal{P} = \{0, 1\}$.
- Choose a monic irreducible polynomial $F(x) \in \mathbb{Z}[x]$ of degree N.
- Repeat:
  - $S(x) \leftarrow_R \mathcal{B}_{\infty,N}(\eta/2)$.
  - $G(x) \leftarrow 1 + 2 \cdot S(x)$.
  - $p \leftarrow \mathsf{resultant}(G(x), F(x))$.
- Until p is prime.
- $D(x) \leftarrow \gcd(G(x), F(x))$ over $\mathbb{F}_p[x]$.
- Let $\alpha \in \mathbb{F}_p$ denote the unique root of $D(x)$.
- Apply the XGCD-algorithm over $\mathbb{Q}[x]$ to obtain $Z(x) = \sum_{i=0}^{N-1} z_i x^i \in \mathbb{Z}[x]$ such that
$$Z(x) \cdot G(x) = p \mod F(x).$$
- $B \leftarrow z_0 \pmod{2p}$.
- The public key is $\mathsf{PK} = (p, \alpha)$, whilst the private key is $\mathsf{SK} = (p, B)$.

Encrypt(M, PK):

- Parse PK as (p, α).
- If $M \notin \{0, 1\}$ then abort.
- $R(x) \leftarrow_R \mathcal{B}_{\infty,N}(\mu/2)$.
- $C(x) \leftarrow M + 2 \cdot R(x)$.
- $c \leftarrow C(\alpha) \pmod{p}$.
- Output c.

Decrypt(c, SK):

- Parse SK as (p, B).
- $M \leftarrow (c - \lfloor c \cdot B/p \rceil) \pmod{2}$.
- Output M.

Add(c_1, c_2, PK):

- Parse PK as (p, α).
- $c_3 \leftarrow (c_1 + c_2) \pmod{p}$.
- Output c_3.

Mult(c_1, c_2, PK):

- Parse PK as (p, α).
- $c_3 \leftarrow (c_1 \cdot c_2) \pmod{p}$.
- Output c_3.

3.2 Analysis

In this section we analyze for which parameter sets our scheme is correct and also determine how many homomorphic operations can be performed before decryption will fail.

KeyGen algorithm. We can see that KeyGen generates an element $\gamma = G(\theta)$ of prime norm p in the number field K defined by $F(x)$. As such we have constructed a small generator of the degree one prime ideal $\mathfrak{p} = \gamma \cdot \mathbb{Z}[\theta]$. To find the two element representation of $\mathfrak{p}$, we need to select the correct root α of $F(x)$ modulo p. Since $\gamma = G(\theta) \in \mathfrak{p}$, we have that $G(\alpha) \equiv 0 \bmod p$, so $G(x)$ and $F(x)$ have at least one common root modulo p. Furthermore, there will be precisely one root in common, since otherwise γ would generate two different prime ideals, which clearly is impossible. This explains the fact that $D(x)$ has degree one; we are using $D(x)$ to select the precise root of $F(x)$ which corresponds to the ideal $\mathfrak{p}$ generated by γ. The two element representation of the ideal $\mathfrak{p}$ then simply is $\mathfrak{p} = p \cdot \mathbb{Z}[\theta] + (\theta - \alpha)\mathbb{Z}[\theta]$.

Encrypt algorithm. The message M is added to twice a small random polynomial $R(x)$ resulting in a polynomial $C(x)$. The ∞-norm of the polynomial $R(x)$ is controlled by the parameter μ. Encryption then simply equals reduction of $C(\theta)$ modulo $\mathfrak{p}$ using the public two element representation $\langle p, \theta - \alpha \rangle$. As explained before, this simply corresponds to evaluating $C(x)$ in α modulo p. Furthermore, note that this precisely implies that $C(\theta) - c \in \mathfrak{p}$.

Decrypt algorithm. By definition of encryption, we have that $C(\theta) - c \in \mathfrak{p}$ and $\mathfrak{p}$ is principal and generated by $\gamma = G(\theta)$. Hence, we can write

$$C(\theta) - c = q(\theta) \cdot \gamma \,,$$

with $q(\theta) \in \mathbb{Z}[\theta]$. It is clear that if we recover the element $C(\theta)$, then decryption will work since $C(\theta) = M + 2 \cdot R(\theta)$. Note that γ^{-1} is precisely given by $Z(\theta)/p$, where Z was computed in KeyGen. Dividing by γ therefore leads to the following equality

$$-c \cdot Z(\theta)/p = q(\theta) - (C(\theta) \cdot Z(\theta))\,/p \,.$$

The above equation shows that if $\|C(\theta) \cdot Z(\theta)/p\|_\infty < 1/2$, then simply rounding the coefficients of $-c \cdot Z(\theta)/p$ will result in the correct quotient $q(\theta)$. This will allow for correct decryption by computing $C(\theta) = c + q(\theta) \cdot \gamma$. The crucial part therefore is to obtain a bound on $\|Z(x)\|_\infty$.

Lemma 1. *Let $F(x), G(x) \in \mathbb{Z}[x]$ with $F(x)$ monic,* $\deg(F) = N$ *and* $\deg(G) = M < N$ *and* $\mathsf{resultant}(F, G) = p$*, then there exists a polynomial $Z(x) \in \mathbb{Z}[x]$ with $Z(x) \cdot G(x) = p \bmod F(X)$ and*

$$\|Z(x)\|_\infty \leq \|G(x)\|_2^{N-1} \cdot \|F(x)\|_2^{M} \,.$$

Proof: Over $\mathbb{Q}[x]$, we have that $\gcd(G(x), F(x)) = 1$, so there exists polynomials $S(x), T(x) \in \mathbb{Q}[x]$ with $\deg(S) < N$ and $\deg(T) < M$ such that $S(x) \cdot G(x) + T(x) \cdot F(x) = 1$. It is well known (see for instance Corollary 6.15 of [6]) that the polynomials S and T are given by $S = \sum_{i=0}^{N-1} s_i x^i$ and $T = \sum_{i=0}^{M-1} t_i x^i$, where the s_i and t_i are the solutions of

$$\mathsf{Syl}(G,F)^{\mathrm{T}} \cdot \begin{pmatrix} s_{N-1} \\ \vdots \\ s_0 \\ t_{M-1} \\ \vdots \\ t_0 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ \\ \vdots \\ 0 \\ 1 \end{pmatrix},$$

where $\mathsf{Syl}(G,F)$ is the Sylvester matrix of G and F. The resultant is precisely $\det(\mathsf{Syl}(G,F)) = p$, so by Cramer's rule we find an explicit expression for the coefficients s_i, namely, the determinant of a submatrix of $\mathsf{Syl}(G,F)^{\mathrm{T}}$ (remove one of the columns containing the coefficients of G and the last row) divided by p. Using Hadamard's inequality to bound the determinant of such submatrices, we finally conclude that $|z_i| \leq \|G\|_2^{N-1} \cdot \|F\|_2^M$. □

In the remainder of the paper we will assume that $M = N-1$ which will happen with very high probability.

Define

$$\delta_\infty := \sup\left\{ \frac{\|g(x) \cdot h(x) \bmod F(x)\|_\infty}{\|g(x)\|_\infty \cdot \|h(x)\|_\infty} \mid \deg(g), \deg(h) < N \right\}.$$

We then have that

$$\|g(\theta) \cdot h(\theta)\|_\infty \leq \delta_\infty \cdot \|g\|_\infty \cdot \|h\|_\infty,$$

where $\deg(g), \deg(h) < N$. Gentry [8, Section 7.4] derives several bounds on the above quantity but for the 2-norm and it is easy to obtain the equivalent bounds for the ∞-norm. To illustrate the two extreme cases, i.e. that δ_∞ can range from fully exponential in N to linear in N, we give the following lemma, which also motivates why we propose to use $F(x) = x^{2^n} + 1$ in practice.

Lemma 2. *Let $F_1(x) = x^N - a$ and $F_2(x) = x^N - ax^{N-1}$ then*

$$\delta_\infty(F_1) \leq |a|N \qquad \textit{and} \qquad \delta_\infty(F_2) \leq |a|^{N-1}N.$$

Proof: Let $g = \sum_{i=0}^{N-1} g_i x^i$ and $h = \sum_{i=0}^{N-1} h_i x^i$, then

$$g \cdot h \bmod F_1 = \sum_{k=0}^{N-1} \left(\sum_{0 \leq i \leq k} g_i h_{k-i} + a \sum_{k<i<N} g_i h_{N+k-i} \right) x^k,$$

from which the bound on $\delta_\infty(F_1)$ immediately follows. Similarly, write $g \cdot h = \sum_{k=0}^{2N-2} c_k x^k$, then $g \cdot h \bmod F_2 = \sum_{k=0}^{N-1} d_k x^k$ with $d_k = c_k$ for $k = 0, \ldots, N-2$ and

$$d_{N-1} = \sum_{i=0}^{N-1} c_{N-1+i} a^i$$

Since all c_i clearly are smaller than $N\|g\|_\infty \|h\|_\infty$ the bound on $\delta_\infty(F_2)$ follows. □

From this we can conclude that

$$\left\| \frac{C(\theta) \cdot Z(\theta)}{p} \right\|_\infty \leq \frac{\delta_\infty \cdot \|C\|_\infty \cdot \|G\|_2^{N-1} \cdot \|F\|_2^{N-1}}{p},$$

so decryption will work as long as

$$\|C\|_\infty < \frac{p}{2 \cdot \delta_\infty \cdot \|G\|_2^{N-1} \cdot \|F\|_2^{N-1}} = \mathsf{r}_{\mathsf{Dec}}.$$

Note that the expected value of $\mathsf{r}_{\mathsf{Dec}}$ will be roughly $\|G\|_2 / 2\delta_\infty$, since the resultant p will be about $\|G\|_2^N \cdot \|F\|_2^{N-1}$. So for $\|C\|_\infty < \mathsf{r}_{\mathsf{Dec}}$, we have

$$C(x) = c + q(\theta) \cdot \gamma = c - \lfloor c \cdot Z(x)/p \rceil \gamma,$$

and since $M \equiv C(x) \bmod 2$ and $\gamma \equiv 1 \bmod 2$ we finally obtain the simplified decryption function

$$M \equiv c - \lfloor c \cdot B/p \rceil \bmod 2,$$

where B is z_0. Note, we can take B as z_0 modulo $2p$ as we are only interested in rounding $c \cdot B/p$ to the nearest integer and then taking the result modulo 2. Furthermore, Lemma 1 implies that all coefficients of $Z(x)$ typically will be smaller than p, since $p = \mathsf{resultant}(F, G)$ and thus $p \simeq \|G(x)\|_2^N \cdot \|F(x)\|_2^M$. This means that the reduction modulo $2p$ in the key generation will have no effect in most cases. However, it will turn out to be a necessary assumption in assuring a uniform distribution when we switch to the full homomorphic scheme.

For our KeyGen algorithm we have that each coefficient of G has size approximately η, which implies that we have the estimate

$$\mathsf{r}_{\mathsf{Dec}} \approx \frac{\sqrt{N} \cdot \eta}{2 \cdot \delta_\infty}.$$

For $F(x) = x^N + 1$ we thus obtain the estimate $\mathsf{r}_{\mathsf{Dec}} \approx \eta/(2 \cdot \sqrt{N})$. In the remainder of the paper we will also sometimes use $\mathsf{r}_{\mathsf{Enc}}$ instead of μ. Note that if one wants to compare with Gentry's scheme, one should take into account that our bounds are formulated for the ∞-norm, whereas Gentry works with the 2-norm.

Add and Mult algorithms. It is clear that both algorithms are correct. However, we need to consider how the error values propagate as we apply Add and Mult. In particular, decryption of $c = C(\alpha)$ will work for a polynomial $C(x)$ if $C(x) \in \mathcal{B}_{\infty,N}(\mathsf{r}_{\mathsf{Dec}})$. However, as we apply Add and Mult to a ciphertext the value of $C(x)$ starts to lie in balls of larger and larger radius. As soon as

$C(x) \notin \mathcal{B}_{\infty,N}(\mathsf{r}_{\mathsf{Dec}})$, we are no longer guaranteed to be able to decrypt correctly. This is why our basic scheme is only somewhat homomorphic, since we are only able to apply Add and Mult a limited number of times.

Let c_1 and c_2 denote two ciphertexts, corresponding to two randomizations $C_1(x) = M_1 + N_1(x)$ and $C_2(x) = M_2 + N_2(x)$; where $M_i \in \{0,1\}$ are the messages and $N_i(x) \in \mathcal{B}_{\infty,N}(r_i - 1)$ is the randomness, i.e. $C_i(x) \in \mathcal{B}_{\infty,N}(r_i)$. We let

$$C_3(x) = M_3 + N_3(x) = (M_1 + N_1(x)) + (M_2 + N_2(x)),$$
$$C_4(x) = M_4 + N_4(x) = (M_1 + N_1(x)) \cdot (M_2 + N_2(x)),$$

where $M_3, M_4 \in \{0,1\}$. Then

$$C_3(x) \in \mathcal{B}_{\infty,N}(r_1 + r_2)$$

and

$$C_4(x) \in \mathcal{B}_{\infty,N}(\delta_\infty \cdot r_1 \cdot r_2).$$

Initially we start with a ciphertext with $C(x)$ lying in $\mathcal{B}_{\infty,N}(\mu+1)$. After executing a circuit with multiplicative depth d, we expect the ciphertext to correspond to a polynomial $C'(x)$ lying in a ball $\mathcal{B}_{\infty,N}(r)$ with

$$r \approx (\delta_\infty \cdot \mu)^{2^d}.$$

Thus we can only decrypt the output of such a circuit if $r \le \mathsf{r}_{\mathsf{Dec}}$, i.e.

$$\begin{aligned} d \log 2 &\le \log\log \mathsf{r}_{\mathsf{Dec}} - \log\log(\delta_\infty \cdot \mu) \\ &\approx \log\log\left(\frac{\sqrt{N} \cdot \eta}{2 \cdot \delta_\infty}\right) - \log\log(\delta_\infty \cdot \mu). \end{aligned}$$

4 Security Analysis

We consider three aspects of security; key recovery, onewayness of the encryption and semantic security. Whilst semantic security is based on what might at first appear a non-traditional problem, the other two notions of security are related to well studied problems in number theory. This is similar to other notions in cryptography; for example key recovery in ElGamal is related to the DLP problem, and semantic security to the relatively obscure (for mathematicians) DDH problem. However, we first show that our scheme is in some sense a specialisation and optimization of Gentry's scheme.

Link With Gentry's Scheme. To discuss the security in more detail, we first show that our scheme is a specialisation and simplification of the lattice based scheme of Gentry [7]. The generator γ in our scheme is equivalent to the private basis of the ideal J in Gentry's scheme, the public basis is then the two element representation $\langle p, \theta - \alpha \rangle$. The ideal I of Gentry's scheme is simply set to the principal ideal $\langle 2\rangle$. Therefore, we see that KeyGen is a specialised form of KeyGen for Gentry's scheme: in particular we use the compact two element representation $\langle p, \alpha\rangle$ of the public basis, instead of the larger HNF representation as Gentry does.

We now turn to the encryption algorithm. The element $C(\theta) = M(\theta) + 2 \cdot R(\theta)$ is precisely the value of ψ' computed in Gentry's encryption algorithm, with a value of $\mathsf{r}_{\mathsf{Enc}}$ (in the 2-norm) equal to $\sqrt{N} \cdot \mu$. Gentry then produces his ciphertext ψ by reducing ψ' modulo the ideal J using the HNF basis. It is at this point that we seem to depart from Gentry's presentation: we actually compute the reduction of ψ' modulo $\mathfrak{p}$ using the public two element representation. Given ψ' as a polynomial in θ, this involves replacing θ by α and reducing the result modulo p. So given $C(x)$, we produce c by simply computing $c = \iota_{\mathfrak{p}}(C(\theta)) \in \mathbb{F}_p$. However, given our earlier discussion on the HNF of the ideal given by $\langle p, \theta - \alpha\rangle$ we see that the two reduction algorithms are equivalent when we are working in the equation order $\mathbb{Z}[\theta]$.

Hence, we conclude that our scheme is a specialisation of Gentry's scheme. For the given specialisation our key sizes are much smaller than Gentry's, whilst our ciphertexts are the same size. When compared to the full generality of Gentry's scheme our ciphertexts are also much smaller than Gentry's. The link between the two schemes, and the relative simplicity of our scheme, may help shed light on parameter choices in Gentry's original scheme.

Key Recovery. Recall the public key in our scheme consists of a principal degree one prime ideal in two element representation, whilst the private key consists of the inverse of a small generator of this principal prime ideal. To see that the generator γ is small, notice that the polynomial $G(x)$ has an ∞-norm given roughly by η, whereas the size of p is roughly $\sqrt{N}^N \eta^N \cdot \|F\|_2^{N-1}$. Recovering the private key given the public key is therefore an instance of the small principal ideal problem:

Definition 1 (Small Principal Ideal Problem (SPIP)). *Given a principal ideal $\mathfrak{a}$ in either two element or HNF representation compute a "small" generator of the ideal.*

This is one of the core problems in computational number theory and has formed the basis of previous cryptographic proposals, see for example [3]. There are currently two approaches to the above problem. The first approach is a deterministic method based on the Baby-Step/Giant-Step method attributed to [1]. This takes time

$$N^{O(N)} \cdot \sqrt{\min(A, R)} \cdot |\Delta|^{o(1)},$$

where Δ is the discriminant of $\mathbb{Z}[\theta]$, R is the regulator and $A = \min_{i=1}^{N} \log|\gamma^{(i)}|$ is the mimimal logarithmic embedding of γ. Clearly A can itself be bounded by η, a minor detail which we leave to the reader.

The second approach to this problem is via Buchmann's sub-exponential algorithm for units and class groups which is described in [2] and [4][Chapter 6]. This method has complexity

$$\exp\left(O(N\log N)\cdot\sqrt{\log(\Delta)\cdot\log\log(\Delta)}\right)$$

where again Δ is the discriminant of the order $\mathbb{Z}[\theta]$. However, this method is likely to produce a generator of large height, i.e. with large coefficients. Indeed so large, that writing the obtained generator down as a polynomial in θ may take exponential time.

In conclusion determining the private key given only the public key is an instance of a classical and well studied problem in algorithmic number theory. In particular there are no efficient solutions for this problem, and the only sub-exponential method does not find a solution which is equivalent to our private key.

Onewayness of Encryption. In this section we consider the problem of recovering a message given a ciphertext element. It is readily seen that this is equivalent to solving the following problem: Given p and $\alpha, c \in \mathbb{F}_p$ find x_i for $i = 0, \dots, N-1$, such that

$$\sum_{i=0}^{N-1} x_i \cdot \alpha^i = c - k \cdot p,$$

where $|x_i| \leq \mathsf{r}_{\mathsf{Enc}}$, for some integer value of k.

To recast this as a lattice problem, consider the lattice generated by the rows of the matrix H given earlier. Consider the lattice vector

$$(k, -x_1, \dots, -x_n) \cdot H = (c - x_0, -x_1, \dots, -x_n).$$

This is a lattice vector which is very close (within $\mathsf{r}_{\mathsf{Enc}}$ in the ∞-norm, or $\sqrt{N}\cdot\mathsf{r}_{\mathsf{Enc}}$ in the 2-norm) to the non-lattice vector $(c, 0, \dots, 0)$. Hence, determining the underlying plaintext given the ciphertext is an instance of the closest vector problem.

However, the underlying lattice is a well-studied lattice in algorithmic number theory, see for example the applications of LLL described in [12,13,15]. A lattice generated by a matrix such as H, namely a matrix in Hermite Normal Form in which all but one diagonal entry is equal to one, is probably the most studied lattice problem from the computational perspective in number theory. Thus whilst we are unable to make use of modern worst-case/average-case reductions for our scheme, the underlying lattice problem is well studied.

However, for later use, we will recap on the analysis Gentry has given for this problem. Although one should bear in mind that Gentry's analysis is for a general lattice arising from the HNF of an ideal and not for the specific one in our scheme. The best known attack on Gentry's scheme is one of lattice reduction, related to the bounded distance decoding problem (BDDP). In particular it is related to finding short/closest vectors within a multiplicative factor of $\mathsf{r}_{\mathsf{Dec}}/\mathsf{r}_{\mathsf{Enc}}$ in a lattice of dimension N. If we set

$$2^{\epsilon} = \frac{\mathsf{r}_{\mathsf{Dec}}}{\mathsf{r}_{\mathsf{Enc}}} = \frac{\sqrt{N}\cdot\eta}{2\cdot\delta_{\infty}\cdot\mu},$$

then it is believed that solving BDDP has difficulty $2^{N/\epsilon}$ (see [8][Section 7.7]). We shall refer to the value $2^{N/\epsilon}$ as the security level of our somewhat homomorphic scheme.

Semantic Security. Finally we discuss the semantic security of our somewhat homomorphic encryption scheme. Consider the following distinguishing problem:

Definition 2 (Polynomial Coset Problem (PCP)). *The challenger first selects* $b \leftarrow_R \{0,1\}$ *and runs* KeyGen *as above to obtain a value of* α *and* p. *If* $b = 0$ *then the challenger performs*

– $R(x) \leftarrow_R \mathcal{B}_{\infty,N}(\mathsf{r}_{\mathsf{Enc}})$.
– $r \leftarrow R(\alpha) \pmod{p}$.

Whilst if $b = 1$ *the challenger performs*

– $r \leftarrow_R \mathbb{F}_p$.

Given (r, PK) *the problem is to guess whether* $b = 0$ *or* $b = 1$.

We call the problem the Polynomial Coset Problem as it is akin to Gentry's Ideal Coset Problem from [7]. The problem basically says one cannot determine whether r is the evaluation of some small polynomial at α or is a random value modulo p. Note that the size of the space $\mathcal{B}_{\infty,N}(\mathsf{r}_{\mathsf{Enc}})$ is roughly ${\mathsf{r}_{\mathsf{Enc}}}^N$, whereas $\mathbb{F}_p$ has size η^N. So if $\mathsf{r}_{\mathsf{Enc}}$ is much smaller than η, we are trying to distinguish a relatively small space within a larger one. Note, in the case where $b = 0$ we generate the value $R(x)$ from $\mathcal{B}_{\infty,N}(\mathsf{r}_{\mathsf{Enc}})$ as opposed to $\mathcal{B}_{\infty,N}(\mathsf{r}_{\mathsf{Dec}})$, since we are interested in arguing about semantic security for what are the simplest ciphertexts to break.

The proof of the following theorem closely follows the proof of Theorem 7 of [7], but we include it here for completeness.

Theorem 1. *Suppose there is an algorithm* $\mathcal{A}$ *which breaks the semantic security of our somewhat homomorphic scheme with advantage* ϵ. *Then there is an algorithm* $\mathcal{B}$, *running in about the same time as* $\mathcal{A}$, *which solves the PCP with advantage* $\epsilon/2$.

Proof: The algorithm $\mathcal{B}$ creates a challenge ciphertext for algorithm $\mathcal{A}$ from its own challenge (r, PK) by setting

$$c \leftarrow (M_\beta(\alpha) + 2 \cdot r) \pmod{p},$$

where M_0 and M_1 are $\mathcal{A}$'s two challenge messages and $\beta \leftarrow_R \{0,1\}$, is $\mathcal{B}$'s choice of a challenge bit. $\mathcal{A}$ sends back a guess β' for β and $\mathcal{B}$ returns $\beta \oplus \beta'$.

When $b = 0$ in the PCP problem, it is clear that the challenge ciphertext c has the correct distribution, so $\mathcal{B}$ obtains the same advantage as $\mathcal{A}$, namely ϵ. When $b = 1$, r is uniformly random modulo p and since p is odd, $2r$ is uniformly random modulo p and therefore so is c. Hence, the advantage of $\mathcal{A}$ is 0, which implies that $\mathcal{B}$'s overall advantage is $\epsilon/2$. $\square$

5 A Fully Homomorphic Scheme

We now proceed to turning the somewhat homomorphic scheme into a fully homomorphic scheme. Since we have shown that our scheme is a specialisation of Gentry's scheme, we only need to recast Gentry's method for our parameters. Indeed we can simplify the method somewhat, since our ciphertext is an integer rather than a vector. We assume that our scheme is secure under key dependent encryptions, purely to keep the notation simpler; to deal with the more general case is immediate from our discussion.

At a high level we need to define a new algorithm called Recrypt, which takes a ciphertext c and re-encrypts it to c_{new}, whilst at the same time removing some of the errors in c. Intuitively this takes a "dirty ciphertext" c and "cleans it" to obtain the ciphertext c_{new}.

To do this we augment the encryption key with some additional information, by extending the algorithm KeyGen with the following additional operations, based on two integer parameters s_1 and s_2. We make use of the fact that we are only interested in the coefficients of $Z(x)$ modulo $2p$.

– Generate s_1 uniformly random integers B_i in $[-p, \ldots, p]$ such that there exists a subset S of s_2 elements with
$$\sum_{j \in S} B_j = B$$
over the integers.
– Define $\mathsf{sk}_i = 1$ if $i \in S$ and 0 otherwise. Notice that only s_2 of the bits $\{\mathsf{sk}_i\}$ are set to one.
– Encrypt the bits sk_i under the somewhat homomorphic scheme to obtain $\mathfrak{c}_i \leftarrow \mathsf{Encrypt}(\mathsf{sk}_i, \mathsf{PK})$.
– The public key now consists of
$$\mathsf{PK} = (p, \alpha, s_1, s_2, \{\mathfrak{c}_i, B_i\}_{i=1}^{s_1}) .$$

We can now describe the re-encryption operation.

$\mathsf{Recrypt}(c, \mathsf{PK})$: This algorithm takes as input a "dirty" ciphertext c, and then produces a "cleaner" ciphertext c_{new} of the same message, but with less "errors" in its randomization vector. The re-encryption works by performing a homomorphic decryption on an encryption of the ciphertexts bits. In the Appendix we explain the Recrypt algorithm in detail and analyse precisely how complicated it is for possible real life values.

Note that we have
$$B = \sum_{i=1}^{s_1} \mathsf{sk}_i \cdot B_i,$$
hence we will now require that this additional information in the public key does not compromise the security of the scheme. Gentry reduces this security issue to the decisional version of the sparse subset-sum problem (SSSP), and hence

the same assumption needs to be made in our situation. The SSSP problem is believed to take at least $\sqrt{\binom{s_1}{s_2}} > (s_1/s_2)^{s_2/2}$ steps to solve, assuming we are not in a low density subset sum, i.e. $s_1/\log p > 1$. If we take s_1 to be slightly greater than $\log p$, then we need to select s_2 such that

$$\sqrt{\binom{s_1}{s_2}} > 2^{N/\epsilon},$$

so as to ensure that the SSSP difficulty is at least as difficult as the difficulty of the BDDP underlying the somewhat homomorphic scheme.

6 Extension to Large Message Space

We now show that our scheme provides for a more powerful fully homomorphic scheme than that of Gentry. In [7] the fully homomorphic property can only be applied to single bit messages, since the Recrypt algorithm for full size messages is relatively complicated. We shall show we can obtain fully homomorphic encryption on N-bit messages and then discuss what this actually means.

First return to our basic scheme. We alter the KeyGen algorithm to output the whole polynomial $Z(x) = \sum_{i=0}^{N-1} z_i x^i$ modulo $2p$ as the secret key as opposed to the single term B. Let the resulting polynomial be denoted $B(x) = \sum_{i=0}^{N-1} b_i x^i$. Encryption is now modified to take any message from the space $\mathcal{B}^{+}_{\infty,N}(2)$, i.e. any binary polynomial of degree less than N. Decryption is then performed coefficient wise, namely each coefficient m_i of M is recovered by computing

$$m_i \leftarrow (c - \lfloor c \cdot b_i/p \rceil) \pmod 2.$$

It is easily seen that this modification results in a somewhat homomorphic scheme with the same multiplicative depth as the original scheme.

We now extend this somewhat homomorphic scheme to a fully homomorphic scheme. We write each coefficient of $B(x)$ as a different sum, over a different set of indices S_i,

$$\sum_{j \in S_i} B_{i,j} = b_i.$$

The secret key is now defined to be $\mathsf{sk}_{i,j} = 1$ if $j \in S_i$ and 0 otherwise. The Recrypt algorithm is then immediate. We first apply the Recrypt algorithm as above, coefficient wise, to obtain new "cleaner" encryptions of each bit of the message, i.e. we obtain

$$c^{(i)}_{\mathsf{new}} = \mathsf{Encrypt}(m_i, \mathsf{PK}).$$

To obtain the encryption of the entire message we simply compute

$$c_{\mathsf{new}} = \mathsf{Encrypt}(m, \mathsf{PK}) = \sum_{i=0}^{N-1} c^{(i)}_{\mathsf{new}} \cdot \alpha^i \pmod p.$$

Note that recombining the different encryptions causes an extra increase in the error term with a factor of δ_∞. This increase in the error term is due to the multiplication, by α^i, of the error term underlying $c_{\mathsf{new}}^{(i)}$.

Hence, we can obtain fully homomorphic encryption with respect to the algebra $\mathbb{F}_2[x]/(F)$. To see the power of this we need to examine the algebra $\mathbb{F}_2[x]/(F)$. If $F(x)$ splits as $\prod_{i=1}^{t} f_i \pmod 2$ with f_i coprime and $\deg f_i = d_i$ then by the Chinese Remainder Theorem we have

$$F_2[x]/(F) \equiv \mathbb{F}_{2^{d_1}} \times \cdots \times \mathbb{F}_{2^{d_t}}.$$

By concentrating on a single component of the product on the right we therefore, by careful choice of F, obtain fully homomorphic encryption in any finite field of characteristic two of degree less than N. Furthermore, we could also obtain SIMD style homomorphic encryption in multiple finite fields of characteristic two at the same time.

7 Implementation Results

We now examine a practical instantiation of our scheme. We take the polynomial $F(x) = x^{2^n} + 1$, which is always irreducible over the integers. In particular our main parameter N is equal to 2^n, and we have $\delta_\infty = N$. We take $\eta = 2^{\sqrt{N}}$ and either $\mu = \sqrt{N}$ or $\mu = 2$. The case of $\eta = 2^{\sqrt{N}}$ and $\mu = \sqrt{N}$ are (for comparison) also the suggested parameter choices made in [7] (albeit in the 2-norm). The case of $\mu = 2$ is chosen to try to obtain as large a depth for the somewhat homomorphic scheme as possible.

Recall that if we write $\eta/(2 \cdot \sqrt{N} \cdot \mu) = 2^\epsilon$, then the security of our somewhat homomorphic scheme is assumed to be $2^{N/\epsilon}$. We then select $s_1 = \log p$ and s_2 to be such that

$$\sqrt{\binom{s_1}{s_2}} > 2^{N/\epsilon},$$

which ensures the difficulty of the SSSP is at least $2^{N/\epsilon}$. In addition, for our choice of $F(x)$, the expected multiplicative depth d for our somewhat homomorphic scheme, is estimated by

$$d \log 2 \le \log\log\left(\frac{\eta}{2 \cdot \sqrt{N}}\right) - \log\log(N \cdot \mu)\,.$$

We present the implications in the following table, for increasing values of n.

		$\mu = 2$			$\mu = \sqrt{N}$		
n	$\log_2 p$	$2^{N/\epsilon}$	s_2	d	$2^{N/\epsilon}$	s_2	d
8	4096	2^{25}	5	0.3	2^{36}	8	0.0
9	11585	2^{31}	6	0.8	2^{40}	7	0.3
10	32768	2^{41}	7	1.2	2^{48}	8	0.8
11	92681	2^{54}	8	1.7	2^{61}	9	1.2
12	262144	2^{73}	10	2.1	2^{80}	11	1.6
13	741455	2^{100}	12	2.5	2^{107}	13	2.1

In the Appendix we make a precise estimate for each value of s_2 what the corresponding Recrypt algorithm will produce in terms of the "dirtyness" of the ciphertext. This allows us to be able to estimate, for each value of s_2, the multiplicative depth $\hat{d}$ which would be required to obtain a fully homomorphic scheme. In the following table we present the value of $\hat{d}$ required. The given value includes the final and-gate to recombine two Recypted ciphertexts. We note that we only obtain a fully homomorphic scheme if $\hat{d} \leq d$, so we see that for practical values of n our scheme cannot be made fully homomorphic, although asymptotically it can be. In fact, in the Appendix we show that for $n \geq 27$ it is possible to obtain a fully homomorphic scheme. For a given fixed security level (and not the maximum possible for a given N and our choice of parameters), it should be possible to obtain a slightly lower n.

s_2	6	7	8	9	10	11	12	13
$\hat{d}$	7	7	7	8	8	8	8	8

The above estimates are very crude and we refer to the Appendix for a more detailed analysis.

Despite this problem with obtaining a fully homomorphic scheme, we timed the various algorithms for the somewhat homomorphic scheme on a desk-top machine using the NTL library: This was an x86-64 platform, and housed 2.4 GHz Intel Core2 (6600) processor cores and used the GCC 4.3.2 C compiler. We were unable to generate keys for the parameter size of $N = 2^{12}$, and smaller values of N key generation could take many hours. The problem with KeyGen being the need to compute many resultants and test the resulting number for primality. This is because the output of the resultant calculation will have $\log_2 p$ bits, so not only are we working with huge numbers; we also have little chance that this number is prime on any one iteration. A more general version of KeyGen would allow for non-prime, but squarefree resultants. But even in this case obtaining keys for say $n = 15$ seems daunting. We thus do not present times for the KeyGen algorithm. The times (in milli-seconds), and the actual value of d computed for the specific key, are presented in the following table:

				d	
n	Encrypt	Decrypt	Mult	$\mu = 2$	$\mu = \sqrt{N}$
8	4.2	0.2	0.2	1.0	0.0
9	38.8	0.3	0.2	1.5	1.0
10	386.4	0.6	0.4	2.0	1.0
11	3717.2	3.0	1.6	2.5	1.5

We see that in practice our scheme appears to obtain a better depth of decryption circuit than theory predicts, although still not deep enough to enable fully homomorphic encryption; at least at practical key sizes.

Acknowledgements. The authors would like to thank the eCrypt NoE funded by the EU for partially supporting the work in this paper. The first author was supported by a Royal Society Wolfson Merit Award, whilst the second was supported by a post-doctoral fellowship of the Research Foundation - Flanders. The authors would also like to thank Florian Hess, Steven Galbraith and Joe Silverman for valuable comments on earlier versions of this paper and Craig Gentry for detailed explanations of his paper [7].

References

1. Buchmann, J.: Zur Komplexität der Berechungung von Einheiten und Klassenzahlen algebraischer Zahlkörper, Habilitationsschrift (1987)
2. Buchmann, J.: A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. Séminaire de Théorie des Nombres – Paris 1988-89, 27–41 (1990)
3. Buchmann, J., Maurer, M., Möller, B.: Cryptography based on number fields with large regulator. Journal de Théorie des Nombres de Bordeaux 12, 293–307 (2000)
4. Cohen, H.: A Course in Computational Algebraic Number Theory. Springer GTM 138 (1993)
5. Ding, J., Lindner, R.: Identifying ideal lattices. IACR eprint 2009/322
6. Von Zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Cambridge University Press, Cambridge (1999)
7. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Symposium on Theory of Computing – STOC 2009, pp. 169–178. ACM, New York (2009)
8. Gentry, C.: A fully homomorphic encryption scheme, (manuscript) (2009)
9. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 112–131. Springer, Heidelberg (1997)
10. Hallgren, S.: Fast quantum algorithms for computing the unit group and class group of a number field. In: Symposium on Theory of Computing – STOC 2005, pp. 468–474. ACM, New York (2005)
11. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: a ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998)
12. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Ann. 261, 513–534 (1982)
13. Nguyen, P.Q., Stern, J.: The two faces of lattices in cryptology. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 146–180. Springer, Heidelberg (2001)
14. Thiel, C.: On the complexity of some problems in algorithmic algebraic number theory. PhD thesis, Universität des Saarlandes, Saarbrücken, Germany (1995)
15. de Weger, B.M.M.: Algorithms for Diophantine Equations. PhD thesis, University of Leiden (1987)

A Analysis of the Recrypt Procedure

In this appendix we explain exactly how Gentry's re-encryption circuit is implemented in the context of our scheme. We first decrease the size of $\mathsf{r}_{\mathsf{Dec}}$ by a factor of two to ensure that the floating point number obtained in the decryption procedure is within 1/4 of an integer, i.e. we know that

$$c \cdot B/p \in]x - 1/4, x + 1/4[\,,$$

for some integer x. Since we are only interested in the result modulo 2, we can actually compute

$$\left\lfloor \sum_{i=1}^{s_1} \mathsf{sk}_i (c \cdot B_i \bmod 2p)/p \right\rceil \pmod 2 .$$

As such we are adding up a subset of s_2 values out of s_1 floating point values, all of which lie in the range $[0, \ldots, 2)$.

To keep the Recryption method manageable we need to minimize the precision of the floating point numbers $(c \cdot B_i \bmod 2p)/p$ that we work with. First note that if we truncate these values to a fixed precision and make a maximum error of $\leq 1/2$, then we can still recover the correct result since the approximated sum will be in the interval $]x - 3/4, x + 1/4[$, and any number in this interval determines x uniquely. More precisely, if we obtain bits e_0, e_1 and e_2 such that the sum computed with fixed precision is given by

$$e_0 + 2^{-1} \cdot e_1 + 2^{-2} \cdot e_2 + \cdots ,$$

then the final output is given by

$$(e_0 + e_1 + e_2 + e_1 \cdot e_2) \pmod 2 .$$

The above equation is derived from examining the four possible cases corresponding to the values of e_1 and e_2;

e_1	e_2	Output
0	0	e_0
1	0	$e_0 + 1 \pmod 2$
0	1	$e_0 + 1 \pmod 2$
1	1	$e_0 + 1 \pmod 2$

Assume we work with t bits of precision, i.e. each floating point number in $[0, \ldots, 2)$ is represented as $\sum_{i=0}^{t-1} e_i 2^{-i}$. Then since only s_2 numbers are non-zero, the maximum error in the total sum is given by

$$s_2 \sum_{i=t}^{\infty} 2^{-i} = s_2 2^{-t+1} .$$

As such we need to choose t such that $s_2 \cdot 2^{-t+1} \leq 1/2$ which implies that $t = \lceil \log_2 s_2 \rceil + 2$. We also define s to be the number of bits to represent all integers up to s_2, i.e. $s = \lfloor \log_2 s_2 \rfloor + 1$. To get some idea of the practical implications of these two values in what follows we give the following table:

s_2	s	t		s_2	s	t
5	3	5		10	4	6
6	3	5		11	4	6
7	3	5		12	4	6
8	4	5		13	4	6
9	4	6		14	4	6

So we see that the value of s is essentially either 3 or 4, and t is either 5 or 6.

The algorithm re-encryption takes as input a ciphertext c and a public key $\mathsf{PK} = (p, \alpha, s_1, s_2, \{\mathfrak{c}_i, B_i\}_{i=1}^{s_1})$ and consists of the following distinct phases:

1. Write down the first t bits of the s_1 floating point numbers $(c \cdot B_i \bmod 2p)/p$ as an $s_1 \times t$ matrix $(b_{i,j})$ for $i = 1, \ldots, s_1$ and $j = 1, \ldots, t$.
2. Encrypt each of the bits $b_{i,j}$ under the public key PK to obtain an $s_1 \times t$ matrix of clean ciphertexts $(c_{i,j})$.
3. Multiply each row of the matrix by the corresponding encryption $\mathfrak{c}_i$ of sk_i to obtain $(\mathfrak{c}_i \cdot c_{i,j}) \bmod p$. As such we obtain the encryption of a matrix with only s_2 non-zero rows.
4. Compute the sum of each column as the Hamming weight using symmetric polynomials and hence reduce the sum of s_1 floating point values to the sum of t floating point values of t bits of precision. More precisely, denote by $h_{i,j}$ the j-th bit of the Hamming weight of the i-th column for $i = 1, \ldots, t$ and $j = 1, \ldots, s$ and form the $t \times t$ matrix $(H_{i,j})$ with $H_{i,j} = h_{i,i-j+s}$ whenever the right hand side is defined and zero otherwise.
5. Merge rows of the matrix H, so as to obtain an $s \times t$ matrix H' such that the sum of the rows of H' equals the sum of the rows of H.
6. Apply carry-save-adders to progressively reduce the matrix to one with two rows. Each set of three rows is reduced to two, and then this procedure is repeated.
7. Perform the final addition, and output the encryption of a single bit.

It is perhaps worth recalling that for our scheme we have $s_1 \approx \log_2 p$ and s_2 is chosen so that $\sqrt{\binom{s_1}{s_2}} \geq 2^\tau$ for our required security level τ, which itself defines the parameter N. The value p is approximately equal to $2^{N \cdot \sqrt{N}}$, thus s_1 is very large indeed. Ciphertexts are integers modulo p and each valid decryptable ciphertext can be considered to lie in ball of a given radius. A "clean" ciphertext lies in a ball of radius $\mu + 1$ and as we add/multiply ciphertexts this radius increases, with the ciphertext becoming increasingly "dirtier". Recall that we have the following behaviour: Let c_1 and c_2 denote two ciphertexts, corresponding to two randomizations $C_1(x) = M_1 + N_1(x)$ and $C_2(x) = M_2 + N_2(x)$; where $M_i \in \{0,1\}$ are the messages and $N_i(x) \in \mathcal{B}_{\infty,N}(r_i - 1)$ is the randomness, i.e. $C_i(x) \in \mathcal{B}_{\infty,N}(r_i)$. For a ciphertext c, denote with $\mathsf{rad}(c)$ the radius of the ball containing the corresponding polynomial $C(x)$, i.e. we have $C(x) \in \mathcal{B}_{\infty,N}(\mathsf{rad}(c))$. Then

$$\begin{aligned} \mathsf{rad}(c_1 + c_2) &= \mathsf{rad}(c_1) + \mathsf{rad}(c_2), \\ \mathsf{rad}(c_1 \cdot c_2) &= \delta_\infty \cdot \mathsf{rad}(c_1) \cdot \mathsf{rad}(c_2). \end{aligned}$$

We will now analyze the growth of the error terms during each of the phases of the re-encryption. Recall that for our choice of parameters we have $\mu = \sqrt{N}$, $\delta_\infty = N$ and $s_1 = N\sqrt{N}$. Therefore define $\rho = \sqrt{N}$, then we will compute explicit expressions for the radii of the ciphertexts as a function of ρ. Recall that the notation $f \sim g$ means that $\lim_{\rho \to \infty} f/g = 1$. If A is a matrix of ciphertexts we let $\mathsf{rad}(A)$ denote the matrix obtained by applying rad to each entry in A.

Stages 1 and 2

The result of the first two stages is that we obtain an $s_1 \times t$ matrix A_2 containing clean ciphertexts $c_{i,j}$ with $\mathsf{rad}(c_{i,j}) = \mu + 1 := r_2 \sim \rho$.

Stage 3

Here we take the clean encryptions of the values of sk_i and multiply through each corresponding row to obtain a matrix A_3, with $(\mathsf{rad}(A_3))_{i,j} = \delta_\infty \cdot r_2^2 := r_3 \sim \rho^4$ for all i, j.

Stage 4

In this stage, we need to compute the encryption of the sum of the (plaintext) bits $\mathsf{sk}_i \cdot b_{i,j}$ in each of the columns seperately. Note that the sum is simply the Hamming weight of the column, so it suffices to compute the bits of the Hamming weight. Furthermore, since only s_2 entries in each column are one, the number of bits in the Hamming weight is bounded by s. We let $\mathrm{SymPol}_i(x_1, \dots, x_k)$ denote the i-th symmetric polynomial on the variables $x_1, \dots, x_k$. Then the bits of the Hamming weight of the bit vector $(b_1, \dots, b_k)$ is given by

$$\left(\mathrm{SymPol}_{2^{s-1}}(b_1, \dots, b_k) \pmod 2, \; \dots, \; \mathrm{SymPol}_{2^0}(b_1, \dots, b_k) \pmod 2\right).$$

So for each column of our matrix A_3 we need to compute all the symmetric polynomials up to $S = 2^{s-1}$. To compute the S symmetric polynomials on (the encryptions of) s_1 bits we proceed in a recursive manner, essentially using Horner's Rule to compute the last $S+1$ coefficients of the product $\prod_{i=1}^{s_1}(b_i \cdot x + 1)$.

For each column of A_3 we execute the following function to compute the (encryptions) of the bits of the Hamming weight of the j-th column for $j = 1, \dots, t$:

```
– Set $(\mathfrak{s}_1, \dots, \mathfrak{s}_S) \leftarrow (0, \dots, 0)$.
– For $i = 1, \dots, s_1$ do
    • For $k = \min(i, S), \dots, 3, 2$ do
        ∗ $\mathfrak{s}_k \leftarrow \mathfrak{s}_k + \mathfrak{s}_{k-1} \cdot A_3(i, j) \pmod p$
    • $\mathfrak{s}_1 \leftarrow \mathfrak{s}_1 + A_3(i, j) \pmod p$.
– Return $(\mathfrak{s}_1, \dots, \mathfrak{s}_S)$.
```

We can also see by analysing the above algorithm how dirty the ciphertexts will become. To produce $\mathfrak{s}_i$ we need to sum $\binom{s_1}{i}$ terms which consist of the multiplication of i of the ciphertexts in A_3 together. If $c_1, \dots, c_8$ are eight ciphertexts given by entries in A_3 then we have

$$\mathsf{rad}(c_1) \sim \rho^4, \quad \mathsf{rad}(c_1 \cdot c_2) \sim \rho^{10}, \quad \mathsf{rad}(c_1 \cdots c_4) \sim \rho^{22}, \quad \mathsf{rad}(c_1 \cdots c_8) \sim \rho^{46}.$$

Thus we have

$$\mathsf{rad}(\mathfrak{s}_1) \sim s_1 \cdot \rho^4 \sim \rho^7, \qquad \mathsf{rad}(\mathfrak{s}_2) \sim \binom{s_1}{2} \cdot \rho^{10} \sim \rho^{16}/2,$$

$$\mathsf{rad}(\mathfrak{s}_4) \sim \binom{s_1}{4} \cdot \rho^{22} \sim \rho^{34}/4!, \qquad \mathsf{rad}(\mathfrak{s}_8) \sim \binom{s_1}{8} \cdot \rho^{46} \sim \rho^{70}/8!.$$

Given the bits $h_{i,j}$ for $i = 1, \ldots, t$ and $j = 1, \ldots, s$ of the t Hamming weights of the t columns, the sum of the resulting floating point numbers represented by the rows of A_3 is given by

$$\sum_{i=1}^{t} \sum_{j=1}^{s} 2^{j-i} h_{i,j} \, .$$

Since we are only interested in the sum modulo 2, we can see that the above sum modulo 2 corresponds to the sum of the rows of the $t \times t$ matrix H with $H_{i,j} = h_{i,i-j+s}$ whenever the right hand side is defined and zero otherwise. We therefore obtain the following matrices H depending on the combination of s and t.

Case $(s, t) = (3, 5)$:
We find

$$\mathsf{rad}(H) \sim \begin{pmatrix} \rho^7 & 0 & 0 & 0 & 0 \\ \rho^{16}/2 & \rho^7 & 0 & 0 & 0 \\ \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 & 0 \\ 0 & \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 \\ 0 & 0 & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \end{pmatrix}$$

Case $(s, t) = (4, 5)$:
We find

$$\mathsf{rad}(H) \sim \begin{pmatrix} \rho^7 & 0 & 0 & 0 & 0 \\ \rho^{16}/2 & \rho^7 & 0 & 0 & 0 \\ \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 & 0 \\ \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 \\ 0 & \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \end{pmatrix}$$

Case $(s, t) = (4, 6)$:
We find

$$\mathsf{rad}(H) \sim \begin{pmatrix} \rho^7 & 0 & 0 & 0 & 0 & 0 \\ \rho^{16}/2 & \rho^7 & 0 & 0 & 0 & 0 \\ \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 & 0 & 0 \\ \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 & 0 \\ 0 & \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 \\ 0 & 0 & \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \end{pmatrix}$$

Stage 5

We now notice that the entries in each column can be permuted around independently. It turns out that it makes more sense, due to the way we will add up the columns in Stage 6, to order the column entries so that the dirtyness increases as you descend a column. This also allows us to delete rows which consist entirely of zeros. We notice that the resulting matrix H' will be of size $s \times t$. In the three examples we do not give the precise permutation of the columns, as this can be deduced from the implied permutation of the d values.

Case $(s,t) = (3,5)$:
We find

$$\mathsf{rad}(H') \sim \begin{pmatrix} \rho^7 & \rho^7 & \rho^7 & 0 & 0 \\ \rho^{16}/2 & \rho^{16}/2 & \rho^{16}/2 & \rho^7 & 0 \\ \rho^{34}/4! & \rho^{34}/4! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \end{pmatrix}$$

Case $(s,t) = (4,5)$:
We find

$$\mathsf{rad}(H') \sim \begin{pmatrix} \rho^7 & \rho^7 & 0 & 0 & 0 \\ \rho^{16}/2 & \rho^{16}/2 & \rho^7 & 0 & 0 \\ \rho^{34}/4! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 \\ \rho^{70}/8! & \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \end{pmatrix}$$

Case $(s,t) = (4,6)$:
We find

$$\mathsf{rad}(H') \sim \begin{pmatrix} \rho^7 & \rho^7 & \rho^7 & 0 & 0 & 0 \\ \rho^{16}/2 & \rho^{16}/2 & \rho^{16}/2 & \rho^7 & 0 & 0 \\ \rho^{34}/4! & \rho^{34}/4! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 & 0 \\ \rho^{70}/8! & \rho^{70}/8! & \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \end{pmatrix}$$

Stage 6

We now apply a sequence of carry-save-adders to reduce the number of rows down to two. We first apply a single chain of carry-save-adders to add the bits in the first three rows of matrix H' which produces two rows as output, where the first row simply contains the exor of the three rows and the second row contains the sum of all products of two out of three rows. Note, we ignore any overflow into the bit position corresponding to the binary weight 2^1 and above. If H' has four rows we then append the fourth row to the result and apply another chain of carry-save-adders. At the end of this stage we have a matrix A_5 of dimension $2 \times t$.

From the above estimates for $\mathsf{rad}(H')$ we can then derive the following estimates for $\mathsf{rad}(A_5)$:

Case $(s,t) = (3,5)$:
We find

$$\mathsf{rad}(A_5) \sim \begin{pmatrix} \rho^{34}/4! & \rho^{34}/4! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \\ \rho^{52}/48 & \rho^{52}/48 & \rho^{25}/2 & 0 & 0 \end{pmatrix}.$$

Case $(s,t) = (4,5)$:
We find

$$\mathsf{rad}(A_5) \sim \begin{pmatrix} \rho^{70}/8! & \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^7 \\ \rho^{106}/4! \cdot 8! & \rho^{52}/2 \cdot 4! & \rho^{25}/2 & 0 & 0 \end{pmatrix}.$$

Case $(s,t) = (4,6)$:
We find

$$\mathsf{rad}(A_5) \sim \begin{pmatrix} \rho^{70}/8! & \rho^{70}/8! & \rho^{70}/8! & \rho^{34}/4! & \rho^{16}/2 & \rho^{7} \\ \rho^{124}/2\cdot 4!\cdot 8! & \rho^{106}/4!\cdot 8! & \rho^{52}/2\cdot 4! & \rho^{25}/2 & 0 & 0 \end{pmatrix}.$$

A.1 Stage 7

The final stage is to add the two remaining rows together, and then use the analysis from earlier. More precisely, as before we write the final output as

$$e_0 + 2^{-1}\cdot e_1 + 2^{-2}\cdot e_2 + \cdots,$$

where, for $i = t-1, \ldots, 0$,

$$\begin{aligned} e_i &= (A_5)_{1,i} + (A_5)_{2,i} + c_{i+1}, \\ c_i &= ((A_5)_{1,i} + (A_5)_{2,i})\cdot c_{i+1} + (A_5)_{1,i}\cdot (A_5)_{2,i}. \end{aligned}$$

where $c_t = 0$. Note that the last two elements on the final row of the above matrices are equal to zero, so there are no carries to worry about from these columns. This simplifies the above expressions a little. We then obtain in each of our cases:

– **Case** $(s,t) = (3,5)$:

$$\mathsf{rad}(e_0, e_1, e_2, e_3, e_4) = \left(\frac{\rho^{115}}{(2\cdot 4!)^2}, \frac{\rho^{61}}{2\cdot 4!}, \frac{\rho^{34}}{4!}, \frac{\rho^{16}}{2}, \rho^7\right)$$

– **Case** $(s,t) = (4,5)$:

$$\mathsf{rad}(e_0, e_1, e_2, e_3, e_4) = \left(\frac{\rho^{133}}{(2\cdot 4!\cdot 8!)}, \frac{\rho^{70}}{8!}, \frac{\rho^{34}}{4!}, \frac{\rho^{16}}{2}, \rho^7\right)$$

– **Case** $(s,t) = (4,6)$:

$$\mathsf{rad}(e_0, e_1, e_2, e_3, e_5) = \left(\frac{\rho^{241}}{2(4!\cdot 8!)^2}, \frac{\rho^{133}}{(2\cdot 4!\cdot 8!)}, \frac{\rho^{70}}{8!}, \frac{\rho^{34}}{4!}, \frac{\rho^{16}}{2}, \rho^7\right)$$

As discussed earlier we can obtain the result Res of the Recryption procedure from the values of e_0, e_1 and e_2, by computing the expression $(e_0+e_1+e_2+e_1\cdot e_2) \pmod 2$. This enables us to determine the value of $\mathsf{rad}(\text{Res})$ in our three cases as follows:

(s,t)	$(3,5)$	$(4,5)$	$(4,6)$
$\mathsf{rad}(\text{Res})$	$\frac{\rho^{115}}{(2\cdot 4!)^2}$	$\frac{\rho^{133}}{(2\cdot 4!\cdot 8!)}$	$\frac{\rho^{241}}{2(4!\cdot 8!)^2}$

So using the above method we can Recrypt a ciphertext to obtain a new ciphertext whose dirtyness measure is bounded by the radius in the above table. We then operate on this ciphertext by applying an addition or a multiplication with another similar ciphertext so as to produce a new ciphertext which we then apply the Recrypt procedure to again. For this to work we require that the ciphertext before Recryption can itself be validly decrypted. This means that we need to be able to decrypt a ciphertext with dirtyness measure given by $\delta_\infty \cdot \mathsf{rad}(\mathrm{Res})^2$.

In the following table we present the final outcome. For a specific value of s_2 we give the values of (s, t) in the algorithm, then we give the value of $\mathsf{rad}(c)$ which needs to be able to be decrypted to obtain fully homomorphic encryption, and then the corresponding minimum value of the "depth" of the circuit. We note that this measure of depth is a very crude estimate since it measures the number of multiplications in a perfectly balanced circuit consisting solely of multiplications, whereas our measure $\mathsf{rad}(c)$ is much more precise.

s_2	(s,t)	$\mathsf{rad}(c)$	depth
$5, 6, 7$	$(3, 5)$	$\dfrac{\rho^{232}}{(2 \cdot 4!)^4}$	7
8	$(4, 5)$	$\dfrac{\rho^{268}}{(2 \cdot 4! \cdot 8!)^2}$	7
$9, 10, 11, 12, 13, 14$	$(4, 6)$	$\dfrac{\rho^{484}}{2^2(4! \cdot 8!)^4}$	8

Recall that the original $\mathsf{r}_{\mathsf{Dec}}$ is given by $2^{\sqrt{N}}/2\sqrt{N}$ and thus equal to $2^\rho/2\rho$. To obtain a fully homomorphic encryption scheme we therefore require that

$$\mathsf{rad}(c) < \frac{2^\rho}{4\rho},$$

where the extra factor of 2 comes from the fact that we made $\mathsf{r}_{\mathsf{Dec}}$ smaller by a factor of 2. It is easy to see that this bound is not attained for the practical parameter sizes given in Section 7. A complete similar analysis for the case $(s, t) = (5, 7)$ gives a radius $\mathsf{rad}(c)$ of

$$\mathsf{rad}(c) = \frac{\rho^{880}}{(8!)^2 \cdot (4! \cdot 16!)^4}$$

which shows that for $\rho \geq 11680$ it is possible to obtain a fully homomorphic encryption scheme. This corresponds to $N \geq 136422400$ or thus $n \simeq 27$.

Unlinkability of Sanitizable Signatures

Christina Brzuska, Marc Fischlin, Anja Lehmann, and Dominique Schröder

Darmstadt University of Technology, Germany
www.minicrypt.de

Abstract. Sanitizable signatures allow a designated party, called the sanitizer, to modify parts of signed data such that the immutable parts can still be verified with respect to the original signer. Ateniese et al. (ESORICS 2005) discuss five security properties for such signature schemes: unforgeability, immutability, privacy, transparency and accountability. These notions have been formalized in a recent work by Brzuska et al. (PKC 2009), discussing also the relationships among the security notions. In addition, they prove a modification of the scheme of Ateniese et al. to be secure according to these notions.

Here we discuss that a sixth property of sanitizable signature schemes may be desirable: unlinkability. Basically, this property prevents that one can link sanitized message-signature pairs of the same document, thus allowing to deduce combined information about the original document. We show that this notion implies privacy, the inability to recover the original data of sanitized parts, but is not implied by any of the other five notions. We also discuss a scheme based on group signatures meeting all six security properties.

1 Introduction

For a regular signature scheme any modification of the message makes the signature for the modified message invalid. In some applications, though, it may be preferable to support message modifications such that one can still verify the authenticity of the immutable message part, and that only authorized parties can make such changes. Signature schemes having this property are called *sanitizable*, as introduced by Ateniese et al. [1]. Related concepts have been discussed concurrently in [24,23,20].

Ateniese et al. [1] discuss the applicability of sanitizable signatures to anonymization of medical data, replacing commercials in authenticated media streams or updates of reliable routing information. They identified five desirable security properties for sanitizable signature schemes. Informally, these are:

UNFORGEABILITY. Says that no one except for the honest signer and sanitizer can create valid signatures.
IMMUTABILITY. Demands that even a malicious sanitizer cannot change message parts which have not been marked as modifiable by the signer.
PRIVACY. Prevents an outsider to recover the original data of sanitized message parts.

Transparency. Covers the indistinguishability of signatures created by the signer or the sanitizer.

Accountability. Refers to the inability of a malicious signer or sanitizer to deny authorship.

Brzuska et al. [3] define these five properties with game-based approaches formally and relate them, showing that accountability implies unforgeability and transparency implies privacy; all other properties are independent. They also prove a modification of the scheme by Ateniese et al. [1] to be secure according to these five properties.

Unlinkability. Here we discuss that an additional property may be useful in some settings. We call this property *unlinkability* and motivate it by the following example (see also Figure 1): Assume that we have signed medical records and at some point we anonymize the data by redacting the personal information of the patients like names, addresses etc. At some other time, say for revenues reasons, we remove the actual medical treatments and leave only the personal information. Then one should not be able to link these data through the (sanitized) signatures and therefore reconstruct the full records. However, previous schemes like the one by Brzuska et al. [3] and, for example, the ones in [21,11,10] in fact allow such attacks. They are usually based on chameleon hashes which remain unchanged for the sanitization step and thus allow to identify two sanitized signatures derived from the same signature through the hash value. Other constructions like the one in [23] even come with an explicit document identifier, allowing to link sanitized messages easily.

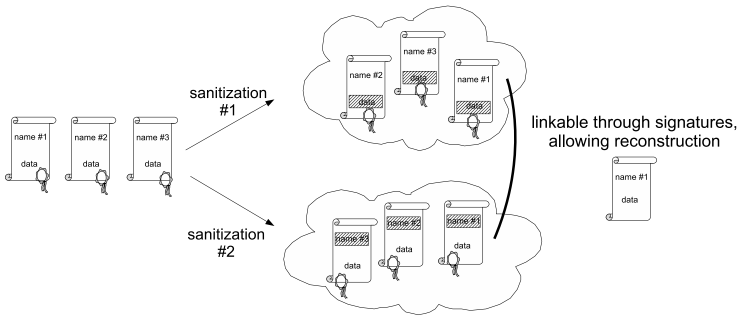


Fig. 1. Linkability problem

We hence introduce a formal definition of unlinkability and relate it to the previously given notions. It turns out that unlinkability is not implied by any of the other properties, but vice versa implies privacy. The reason is that privacy prevents an adversary of recovering the original data for sanitized parts, and violation of this property also enables the adversary to reconstruct and to link messages easily.

Construction. We then present a construction of a sanitizable signature scheme obeying all six properties, including unlinkability. The idea is fundamentally different from previous approaches which usually rely on chameleon hashes. In our case the signer first signs the fixed parts with a regular signature scheme. For the modifiable parts the signer and the sanitizer use a group signature scheme [13], i.e., a signature scheme which allows to sign anonymously on behalf of the group but such that a group manager can revoke the identity of the user that has signed [5]. In our case the group only consists of the signer and sanitizer, and the signer also incarnates the group manager. If the sanitizer later changes (some of) the modifiable message parts it can create a new group signature and replace the signer's group signature.

The anonymity of the group signature scheme in our context guarantees transparency (the indistinguishability of signatures originating from the signer and the sanitizer). The possibility to identify a group member by the group manager (i.e., the signer in our case) supports sanitizer-accountability, i.e., the ability to provide a proof that the sanitizer has created the signature. Signer-accountability is provided by the non-frameability of the group signature scheme which prevents a malicious group manager (i.e., the signer) from falsely accusing the sanitizer to be the source of a signature. Immutability follows from the unforgeability of the regular signature scheme for the fixed parts, and unlinkability from the fact that the sanitizer signs the entire message from scratch (the signature for fixed message parts remains unchanged).

We remark that the actual construction needs a careful implementation of the idea above to make the derived sanitizable signature scheme satisfy all desired security properties. This is in particular true since proposed group signature schemes in the literature like [5,9,22,14,18,19] come with varying security features and set-up assumptions. In this version we thus present a simple but not necessarily the most practical approach to turn our idea into a secure sanitizable scheme, e.g., following the definitions in [3] we do not rely on the fact that public keys of the signer or sanitizer are registered, although this is most likely in practice. In the full version we discuss further variations, e.g., multiple sanitizers, or using a ring signature scheme instead of a group signature scheme, thus dropping the accountability requirement for the derived sanitizable scheme.

Our solution shows that, in general, sanitizable signatures can be built from group signatures, thereby providing a new application for the latter primitive. This relation also immediately gives a feasibility result for sanitizable signatures: Since the work by Bellare et al. [5] about group signatures proves that one can derive them from IND-CCA secure encryption, non-interactive zero-knowledge proofs and digital signatures, all known to exist given trapdoor permutations, it follows that one can also build secure sanitizable signatures from trapdoor permutations.

Organization. In Section 2 we introduce the notion of sanitizable signatures and the security properties given in [1,3]. In Section 3 we discuss the notion of unlinkability and its relationship to the other security properties. In Section 4 we present our construction of a secure sanitizable scheme based on group signatures.

2 Preliminaries

In this section we revisit the notion of sanitizable signatures and the previously given security properties.

2.1 Sanitizable Signatures

In a sanitizable signature scheme both the signer and the sanitizer hold a key pair $(sk_{\mathrm{sig}}, pk_{\mathrm{sig}})$, $(sk_{\mathrm{san}}, pk_{\mathrm{san}})$ such that the signer can sign messages with its secret key sk_{sig} and "attach" a description of the admissible modifications $\textsc{adm}$ which are allowed to the sanitizer pk_{san}. The sanitizer can then later change such a message according to some modification $\textsc{mod}$ and update the signature using his secret key sk_{san}. In order to settle disputes about the origin of a message-signature pair the algorithm Proof enables the signer to produce a proof π from previously signed messages that a signature has been created by the sanitizer. This proof can then be verified with the help of the Judge algorithm (but which only needs to decide about the origin in case of a valid message-signature pair in question; for invalid pairs such decisions are in general impossible).

To model admissible modifications we assume that $\textsc{adm}$ and $\textsc{mod}$ are (descriptions of) efficient deterministic algorithms such that $\textsc{mod}$ maps any message m to the modified message $m' = \textsc{mod}(m)$, and $\textsc{adm}(\textsc{mod}) \in \{0, 1\}$ indicates if the modification is admissible and matches $\textsc{adm}$, in which case $\textsc{adm}(\textsc{mod}) = 1$. For example, for messages $m = m[1] \dots m[k]$ divided into blocks $m[i]$ of equal bit length t we can let $\textsc{adm}$ contain t and the indices of the modifiable blocks, and $\textsc{mod}$ then essentially consists of pairs $(j, m'[j])$ defining the new value for the j-th block.

For ease of notation we let $\textsc{fix}_{\textsc{adm}}$ be an efficient deterministic algorithm which is uniquely determined by $\textsc{adm}$ and which maps m to the immutable message part $\textsc{fix}_{\textsc{adm}}(m)$, e.g., for block-divided messages $\textsc{fix}_{\textsc{adm}}(m)$ is the concatenation of all blocks not appearing in $\textsc{adm}$. To exclude trivial examples we demand that admissible modifications leave the fixed part of a message unchanged, i.e., $\textsc{fix}_{\textsc{adm}}(m) = \textsc{fix}_{\textsc{adm}}(\textsc{mod}(m))$ for all $m \in \{0, 1\}^*$ and all $\textsc{mod}$ with $\textsc{adm}(\textsc{mod}) = 1$. Analogously, to avoid choices like $\textsc{fix}_{\textsc{adm}}$ having empty output, we also require that the fixed part must be "maximal" given $\textsc{adm}$, i.e., $\textsc{fix}_{\textsc{adm}}(m') \neq \textsc{fix}_{\textsc{adm}}(m)$ for $m' \notin \{\textsc{mod}(m) \mid \textsc{mod} \text{ with } \textsc{adm}(\textsc{mod}) = 1\}$.

Jumping ahead, we note that for our construction based on group signatures we make another assumption on $\textsc{adm}$. This property, denoted modification-decidability, allows to decide efficiently for given messages m, m^* and $\textsc{adm}$ whether m^* is an admissible modification of m with respect to $\textsc{adm}$ or not. This property is for example satisfied for the block-based approach. However, for our definitions of the security properties and their relationships we do not impose any restriction at this point.

The following definition is taken from [3]:

Definition 1 (Sanitizable Signature Scheme). *A sanitizable signature scheme* SanSig *consists of seven efficient algorithms* (KGen_{sig}, KGen_{san}, Sign, Sanit, Verify, Proof, Judge) *such that:*

Key Generation. *There are two key generation algorithms, one for the signer and one for the sanitizer. Both create a pair of keys, a private key and the corresponding public key:*

$$(pk_{sig}, sk_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^n), \qquad (pk_{san}, sk_{san}) \leftarrow \mathsf{KGen}_{san}(1^n)$$

Signing. *The* Sign *algorithm takes as input a message $m \in \{0,1\}^*$, the secret key sk_{sig} of the signer, the public key pk_{san} of the sanitizer, as well as a description* ADM *of the admissibly modifiable message parts. It outputs a signature (or $\perp$, indicating an error):*

$$\sigma \leftarrow \mathsf{Sign}(m, sk_{sig}, pk_{san}, \text{ADM}).$$

We assume that ADM *is recoverable from any signature $\sigma \neq \perp$.*

Sanitizing. *Algorithm* Sanit *takes a message $m \in \{0,1\}^*$, a signature σ, the public key pk_{sig} of the signer and the secret key sk_{san} of the sanitizer. It modifies the message m according to the modification instruction* MOD *and determines a new signature σ' for the modified message $m' = \text{MOD}(m)$. Then* Sanit *outputs m' and σ' (or possibly $\perp$ in case of an error).*

$$(m', \sigma') \leftarrow \mathsf{Sanit}(m, \text{MOD}, \sigma, pk_{sig}, sk_{san})$$

Verification. *The* Verify *algorithm outputs a bit $d \in \{\mathtt{true}, \mathtt{false}\}$ verifying the correctness of a signature σ for a message m with respect to the public keys pk_{sig} and pk_{san}.*

$$d \leftarrow \mathsf{Verify}(m, \sigma, pk_{sig}, pk_{san})$$

Proof. *The* Proof *algorithm takes as input the secret signing key sk_{sig}, a message m and a signature σ as well a set of (polynomially many) additional message-signature pairs $(m_i, \sigma_i)_{i=1,2,\ldots,q}$ and the public key pk_{san}. It outputs a string $\pi \in \{0,1\}^*$:*

$$\pi \leftarrow \mathsf{Proof}(sk_{sig}, m, \sigma, (m_1, \sigma_1), \ldots, (m_q, \sigma_q), pk_{san})$$

Judge. *Algorithm* Judge *takes as input a message m and a valid signature σ, the public keys of the parties and a proof π. It outputs a decision $d \in \{\mathtt{Sig}, \mathtt{San}\}$ indicating whether the message-signature pair has been created by the signer or the sanitizer:*

$$d \leftarrow \mathsf{Judge}(m, \sigma, pk_{sig}, pk_{san}, \pi)$$

For a sanitizable signature scheme the usual correctness properties should hold, saying that genuinely signed or sanitized messages are accepted and that a genuinely created proof by the signer leads the judge to decide in favor of the signer. For a formal approach to correctness see [3].

2.2 Security of Sanitizable Signatures

Here we recall the security notions for sanitizable signatures given by Brzuska et al. [3]. We note that, there, they show that signer and sanitizer accountability together imply unforgeability, and that transparency implies privacy. Hence, in principle it suffices to show immutability, accountability and transparency. We therefore omit the formal definitions of unforgeability and privacy here and refer the reader to the full version of the paper.

Immutability. This property demands informally that a malicious sanitizer cannot change inadmissible blocks. In the attack model below the malicious sanitizer $\mathcal{A}$ interacts with the signer to receive signatures σ_i for messages m_i, descriptions ADM_i and keys $pk_{\text{san},i}$ of its choice, before eventually outputting a valid pair $(pk^*_{\text{san}}, m^*, \sigma^*)$ such that message m^* is not a "legitimate" transformation of one of the m_i's under the same key $pk^*_{\text{san}} = pk_{\text{san},i}$. The latter is formalized by requiring that for each query $pk^*_{\text{san}} \neq pk_{\text{san},i}$ or $m^* \notin \{\text{MOD}(m_i) \mid \text{MOD} \text{ with } \text{ADM}_i(\text{MOD}) = 1\}$ for the value ADM_i in σ_i, e.g., that for block-divided messages m^* and m_i differ in at least one inadmissible block. As the adversary can query the signer for several sanitizer keys pk_{san}, the security definition also covers the case where the signer interacts with several sanitizers simultaneously.

Definition 2 (Immutability). *A sanitizable signature scheme* SanSig *is* immutable *if for any efficient algorithm $\mathcal{A}$ the probability that the following experiment* $\mathsf{Immutability}^{\mathsf{SanSig}}_{\mathcal{A}}(n)$ *returns* 1 *is negligible (as a function of n).*

```
Experiment $\mathsf{Immutability}^{\mathsf{SanSig}}_{\mathcal{A}}(n)$
    $(pk_{sig}, sk_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^n)$
    $(pk^*_{san}, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\cdot, sk_{sig}, \cdot, \cdot), \mathsf{Proof}(sk_{sig}, \ldots, \cdot)}(pk_{sig})$
        letting $(m_i, \text{ADM}_i, pk_{san,i})$ and $\sigma_i$ for $i = 1, 2, \ldots, q$
        denote the queries and answers to and from oracle Sign.
    return 1 if
        $\mathsf{Verify}(m^*, \sigma^*, pk_{sig}, pk^*_{san}) = \mathtt{true}$ and
        for all $i = 1, 2, \ldots, q$ we have
            $pk^*_{san} \neq pk_{san,i}$ or
            $m^* \notin \{\text{MOD}(m_i) \mid \text{MOD} \text{ with } \text{ADM}_i(\text{MOD}) = 1\}$
```

Accountability. Accountability says the origin of a (sanitized) signature should be undeniable. There are the following two types of accountability: *sanitizer-accountability* says that, if a message has not been signed by the signer, then even a malicious sanitizer should not be able to make the judge accuse the signer. *Signer-accountability* says that, if a message and its signature have not been sanitized, then even a malicious signer should not be able to make the judge accuse the sanitizer.

In the sanitizer-accountability game let $\mathcal{A}_{\mathsf{Sanit}}$ be an efficient adversary playing the role of the malicious sanitizer. Adversary $\mathcal{A}_{\mathsf{Sanit}}$ has access to a Sign and Proof oracle. Her task is to output a valid message-signature pair m^*, σ^* together with a key pk^*_{san} (with (pk^*_{san}, m^*) being different from pairs $(m_i, pksani)$ previously queried to the Sign oracle) such that the proof produced by the signer via Proof still leads the judge to decide "`Sig`", i.e., that the signature has been created by the signer.

Definition 3 (Sanitizer-Accountability). *One calls a sanitizable signature scheme* SanSig sanitizer-accountable *if for any efficient* $\mathcal{A}_{\mathsf{Sanit}}$ *the probability that the following experiment* $\mathsf{San\text{-}Accountability}^{\mathsf{SanSig}}_{\mathcal{A}_{\mathsf{Sanit}}}(n)$ *returns* 1 *is negligible (as a function of n).*

```
Experiment $\mathit{San\text{-}Accountability}^{\mathsf{SanSig}}_{\mathcal{A}_{Sanit}}(n)$
    $(pk_{sig}, sk_{sig}) \leftarrow \mathit{KGen}_{sig}(1^n)$
    $(pk^*_{san}, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathit{Sign}(\cdot, sk_{sig}, \cdot, \cdot), \mathit{Proof}(sk_{sig}, \ldots, \cdot)}_{\mathit{Sanit}}(pk_{sig})$
        letting $(m_i, \text{ADM}_i, pk_{san,i})$ and $\sigma_i$ for $i = 1, 2, \ldots, q$
        denote the queries and answers to and from oracle Sign
    $\pi \leftarrow \mathit{Proof}(sk_{sig}, m^*, \sigma^*, (m_1, \sigma_1), \ldots, (m_q, \sigma_q), pk^*_{san})$
    return 1 iff
        $(pk^*_{san}, m^*) \neq (pk_{san,i}, m_i)$ for all $i = 1, 2, \ldots, q$, and
        $\mathit{Verify}(m^*, \sigma^*, pk_{sig}, pk^*_{san}) = \mathtt{true}$, and
        $\mathit{Judge}(m^*, \sigma^*, pk_{sig}, pk^*_{san}, \pi) = \mathtt{Sig}$
```

In the signer-accountability game a malicious signer $\mathcal{A}_{\mathsf{Sign}}$ gets a public sanitizing key pk_{san} as input. She is allowed to query a sanitizing oracle about tuples $(m_i, \text{MOD}_i, \sigma_i, pk_{\text{sig},i})$ receiving answers (m'_i, σ'_i). Adversary $\mathcal{A}_{\mathsf{Sign}}$ finally outputs a tuple $(pk^*_{\text{sig}}, m^*, \sigma^*)$ and is considered to succeed if Judge accuses the sanitizer for the new key-message pair pk^*_{sig}, m^* with a valid signature σ^*.

Definition 4 (Signer-Accountability). *A sanitizable signature scheme* SanSig *is called* signer-accountable *if for any efficient* $\mathcal{A}_{\mathsf{Sign}}$ *the probability that the following experiment* $\mathsf{Sig\text{-}Accountability}^{\mathsf{SanSig}}_{\mathcal{A}_{\mathsf{Sign}}}(n)$ *returns* 1 *is negligible (as a function of n):*

```
Experiment $\mathit{Sig\text{-}Accountability}^{\mathsf{SanSig}}_{\mathcal{A}_{Sign}}(n)$
    $(pk_{san}, sk_{san}) \leftarrow \mathit{KGen}_{san}(1^n)$
    $(pk^*_{sig}, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathit{Sanit}(\cdot, \cdot, \cdot, \cdot, sk_{san})}_{\mathit{Sign}}(pk_{san})$
        letting $(m'_i, \sigma'_i)$ for $i = 1, 2, \ldots, q$
        denote the answers from oracle Sanit.
    return 1 iff
        $(pk^*_{sig}, m^*) \neq (pk_{sig,i}, m'_i)$ for all $i = 1, 2, \ldots, q$, and
        $\mathit{Verify}(m^*, \sigma^*, pk^*_{sig}, pk_{san}) = \mathtt{true}$ and
        $\mathit{Judge}(m^*, \sigma^*, pk^*_{sig}, pk_{san}, \pi^*) = \mathtt{San}$
```

Transparency. We define transparency by the following adversarial game. We consider an adversary $\mathcal{A}$ with access to Sign, Sanit and Proof oracles with which the adversary can create signatures for (sanitized) messages and learn proofs. In addition, $\mathcal{A}$ gets access to a Sanit/Sign box which contains a secret random bit $b \in \{0, 1\}$ and which, on input a message m, a modification information MOD and a description ADM

- for $b = 0$ runs the signer algorithm to create $\sigma \leftarrow \mathsf{Sign}(m, sk_{\text{sig}}, pk_{\text{sig}}, \text{ADM})$, then runs the sanitizer algorithm and returns the sanitized message m' with the new signature σ', and
- for $b = 1$ acts as in the case $b = 0$ but also signs m' from scratch with the signing algorithm to create a signature σ' and returns the pair (m', σ').

Adversary $\mathcal{A}$ eventually produces an output a, the guess for b. A sanitizable signature is now said to be *transparent* if for all efficient algorithms $\mathcal{A}$ the probability for a right guess $a = b$ in the above game is negligibly close to $\frac{1}{2}$. Below we also define a relaxed version called *proof-restricted transparency* and discuss the idea after the definition.

Definition 5 ((Proof-Restricted) Transparency). *A sanitizable signature scheme* SanSig *is* (proof-restricted) transparent *if for any efficient algorithm $\mathcal{A}$ the probability that the following experiment* $\mathsf{Transparency}_{\mathcal{A}}^{\mathsf{SanSig}}(n)$ *returns* 1 *is negligibly close to $\frac{1}{2}$.*

```
Experiment $\mathsf{Transparency}_{\mathcal{A}}^{\mathsf{SanSig}}(n)$
    $(pk_{sig}, sk_{sig}) \leftarrow KGen_{sig}(1^n)$
    $(pk_{san}, sk_{san}) \leftarrow KGen_{san}(1^n)$
    $b \leftarrow \{0,1\}$
    $a \leftarrow \mathcal{A}^{Sign(\cdot, sk_{sig}, \cdot, \cdot), Sanit(\cdot,\cdot,\cdot,\cdot, sk_{san}), Proof(sk_{sig}, \ldots, \cdot), Sanit/Sign(\cdot,\cdot,\cdot, sk_{sig}, sk_{san}, pk_{sig}, pk_{san}, b)}$
        with input $(pk_{sig}, pk_{san})$
        where oracle Sanit/Sign for input $m_k, \text{MOD}_k, \text{ADM}_k$
        first computes $\sigma_k \leftarrow Sign(m_k, sk_{sig}, pk_{san}, \text{ADM}_k)$,
        then computes $(m'_k, \sigma'_k) \leftarrow Sanit(m_k, \text{MOD}_k, \sigma_k, pk_{sig}, sk_{san})$,
        then, if $b = 1$, replaces $\sigma'_k$ by $\sigma'_k \leftarrow Sign(m'_k, sk_{sig}, pk_{san}, \text{ADM}_k)$,
        and finally returns $(m'_k, \sigma'_k)$.
    return 1 iff
        $a = b$
        (and, in the proof-restricted case, $\mathcal{A}$ has not queried
        any $m'_k$ output by Sanit/Sign to Proof)
```

The original definition of Brzuska et al. [3] does not consider the proof-restricted case. Without this restriction, though, achieving transparency at first seems to be impossible because the adversary can then always submit the replies of the Sanit/Sign oracle to the Proof oracle and thereby recover the secret bit b. However, in their construction the Proof algorithm searches in the list of previously signed messages and only gives a useful answer if it finds a match, enabling transparency without this restriction. Yet, any solution (like ours here) where the Proof algorithm is "history-free" can only achieve the proof-restricted version. Note that Proof algorithms forgoing the set of previously signed messages are preferable from an efficiency point of view, of course.

As for the implications among the security notions [3] we note that proof-restricted transparency only implies a proof-restricted form of privacy, where the answer messages of the LoRSanit oracle cannot be submitted to the Proof oracle either. However, since we show in the next section that unlinkability implies full privacy and our construction achieves unlinkability, our scheme is also private in the non-restricted sense. We note that all the separation results in [3] remain valid for proof-restricted transparency.

3 Unlinkability

In this section we define unlinkability formally and discuss its relationship to the other security notions.

3.1 Definition

As explained in the introduction, unlinkability refers to the impossibility to use the signatures to identify sanitized message-signature pairs originating from the same source. Technically, we use an indistinguishability-based approach to define this property, saying that, given a signature for a sanitized message of two possible sources, the adversary cannot predict the actual original message better than by guessing. This should even hold if the adversary herself provides the two source message-signature pairs and modifications of which one is sanitized. The stipulation here is that the two modifications yield the same sanitized message. Else, if for example the sanitized messages still contain some unique but distinct entry, then predicting the source is easy, of course. This, however, is beyond the scope of signature schemes: the scheme should only prevent that *signatures* can be used to link data.

Formally, we use a game-based approach to define unlinkability, similar to the other security notions in [3]. The adversary is given access to a signing oracle and a sanitizer oracle (and a proof oracle since this step depends on the signer's secret key and may leak valuable information). The adversary is also allowed to query a left-or-right oracle LoRSanit which is initialized with a secret random bit b. In each of the multiple queries to LoRSanit the adversary provides a pair of tuples, each consisting of a message, a modification and a *valid* signature, such that the recoverable description of admissible modifications is identical in both cases (since we assume that ADM is recoverable from a signature providing distinct descriptions ADM would allow a trivial attack; so would the case that only one signature is valid). Depending on the bit b, the adversary gets the sanitized message-signature pair of either the left or right input pair. The adversary should eventually predict the bit b significantly better than with the guessing probability of $\frac{1}{2}$.

Definition 6 (Unlinkability). *A sanitizable signature scheme* SanSig *is* unlinkable *if for any efficient algorithm $\mathcal{A}$ the probability that the following experiment* $\mathsf{Unlinkability}_{\mathcal{A}}^{\mathsf{SanSig}}(n)$ *returns* 1 *is negligibly close to* $\frac{1}{2}$.

```
***Experiment*** $\mathsf{Unlinkability}_{\mathcal{A}}^{\mathsf{SanSig}}(n)$
    $(pk_{sig}, sk_{sig}) \leftarrow \mathit{KGen}_{sig}(1^n)$
    $(pk_{san}, sk_{san}) \leftarrow \mathit{KGen}_{san}(1^n)$
    $b \leftarrow \{0,1\}$
```

$a \leftarrow \mathcal{A}^{\mathsf{Sign}(sk_{sig},\cdots),\mathsf{Sanit}(sk_{san},\cdots),\mathsf{Proof}(sk_{sig},\cdots),\mathsf{LoRSanit}(sk_{sig},sk_{san},b,\cdots)}(pk_{sig}, pk_{san})$
where oracle $\mathsf{LoRSanit}(\cdot,\cdot,\cdot,sk_{sig},sk_{san},b)$*, on input*
$(m_{j,0}, \text{MOD}_{j,0}, \sigma_{j,0}, m_{j,1}, \text{MOD}_{j,1}, \sigma_{j,1})$ *with recoverable* $\text{ADM}_{j,0} = \text{ADM}_{j,1}$
$\mathsf{Verify}(m_{j_0}, \sigma_{j,0}, pk_{sig}, pk_{san}) = \mathtt{true}$, $\mathsf{Verify}(m_{j_1}, \sigma_{j,1}, pk_{sig}, pk_{san}) = \mathtt{true}$,
returns $(m'_j, \sigma'_j) \leftarrow \mathsf{Sanit}(m_{j,b}, \text{MOD}_{j,b}, \sigma_{j,b}, pk_{sig}, sk_{san})$,
and where $(m_{j,0}, \text{MOD}_{j,0}, \text{ADM}_{j,0}) \equiv (m_{j,1}, \text{MOD}_{j,1}, \text{ADM}_{j,1})$,
i.e., are mapped to the same modified message.
return 1 *if* $a = b$.

A pictorial description is given in Figure 2. We note that the definition above is for example robust concerning several sanitization steps in the LoRSanit oracle. That is, we could allow the adversary in the experiment above to submit arbitrarily long "modification chains" $\text{MOD}^1_{j,0}, \ldots, \text{MOD}^m_{j,0}$ and $\text{MOD}^1_{j,1}, \ldots, \text{MOD}^m_{j,1}$ such that the two source documents are gradually sanitized with a match in the resulting documents. Still, predicting b remains hard, as such chains can potentially be simulated by calling the sanitizer oracle for the first $m-1$ modifications manually, before entering the final sanitization step into the LoRSanit oracle.

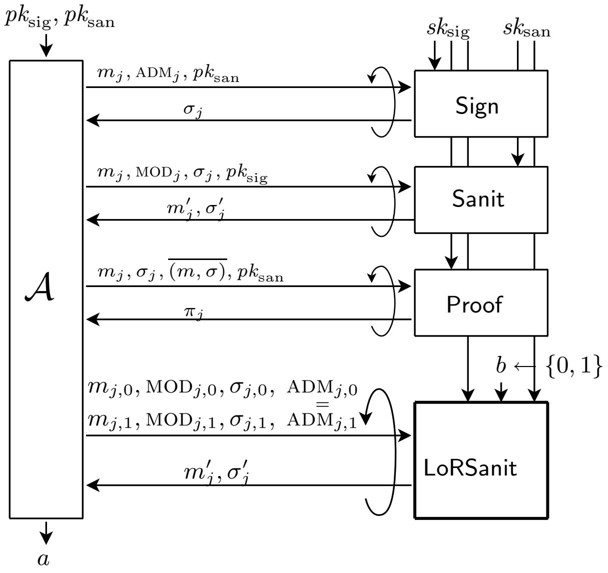

$pk_{\text{sig}}, pk_{\text{san}}$
sk_{sig}
sk_{san}
m_j, ADM$_j$, pk_{san}
σ_j
Sign
m_j, MOD$_j$, σ_j, pk_{sig}
m'_j, σ'_j
Sanit
$m_j, \sigma_j, \overline{(m,\sigma)}, pk_{\text{san}}$
π_j
Proof
$\mathcal{A}$
$b \leftarrow \{0,1\}$
$m_{j,0}$, MOD$_{j,0}$, $\sigma_{j,0}$, ADM$_{j,0}$
=
$m_{j,1}$, MOD$_{j,1}$, $\sigma_{j,1}$, ADM$_{j,1}$
m'_j, σ'_j
LoRSanit
a

Fig. 2. Unlinkability. $\mathcal{A}$ wins if it outputs $a = b$.

Recall the example of medical records which are sanitized twice, one time basically removing the personal information and the other time removing the medical data. Our notion of unlinkability can then be used to show that such sanitized message-signature pairs do not allow to reconstruct the full data better than by guessing. Assume for simplicity that we only have two records with entries (`name#0`, `data#0`) and (`name#1`, `data#1`). Then we create all four possible combinations ($\texttt{name\#}a, \texttt{data\#}b$) for $a, b \in \{0, 1\}$ and ask for signatures for them (with both parts being admissibly changeable). For each $a \in \{0, 1\}$ we then insert the pairs ($\texttt{name\#}a, \texttt{data\#0}$) and ($\texttt{name\#}a, \texttt{data\#1}$) twice into the $\mathsf{LoRSanit}$ oracle, one time cutting off the name-part, the other time removing the data-part. Altogether we make thus four calls to the $\mathsf{LoRSanit}$ oracle, and we hand those four replies to the adversary. Our unlinkability definition says that one cannot distinguish the two cases (left or right sanitization) better than by guessing, thus also disallowing to tell which data belong to whose name.

Our definition above is for unlinkability with respect to message-signature pairs sanitized by the *same* sanitizer. One can easily extend the above definition by demanding that the adversary can also determine different sanitizers for the left and for the right input data. But then both sanitizers must have been declared to have the permission to sanitize, otherwise one could easily determine the secret bit of the $\mathsf{LoRSanit}$ by picking an invalid sanitizer for one of the input tuples.

3.2 Relationships of the Security Notions

We first show that unlinkability does not follow from any of the other security requirements. Then we prove that unlinkability implies privacy, and finally discuss that unlinkability does not imply any of the other properties.

Proposition 1 (Independence of Unlinkability). *Assume that there exists a sanitizable signature scheme (obeying one or more of the properties unforgeability, immutability, privacy, (proof-restricted) transparency, signer-accountability and sanitizer-accountability). Then there exists a sanitizable signature scheme which is not unlinkable but preserves the other security properties.*

The proof follows by simply appending a unique identifier id to each signature. This does not destroy any of the other security properties but clearly violates unlinkability. The proof of the following is straightforward as the privacy experiment is essentially the unlinkability experiment with less control for the adversary:

Proposition 2 (Unlinkability Implies Privacy). *Any unlinkable sanitizable signature scheme is also private.*

With the next proposition we show that unlinkability does not imply any of the other security properties (assuming that we start with a secure sanitizable signature scheme like the one we construct in the next section):

Proposition 3 (Independence of Other Properties). *Assume that there exists a sanitizable signature scheme which is unforgeable, immutable, private, (proof-restricted) transparent, signer-accountable, sanitizer-accountable and unlinkable. Then for any of the properties immutability, transparency, unlinkability, signer-accountability and sanitizer-accountability, there exists a sanitizable signature scheme obeying all properties except for the one in question.*

Proof. The fact that unlinkability does not follow from the other properties has already been shown in Proposition 1. For the other properties we remark that the counterexamples in [3] which seperate immutability, transparency, signer-accountability and sanitizer-accountability from the other properties also preserve unlinkability in each case (and also hold for proof-restricted transparency). □

4 Constructions Based on Group Signatures

In this section we present our unlinkable sanitizable signature scheme (which also satisfies the other security properties). As explained in the introduction, the idea is to use a group signature scheme for the group consisting of the signer and the sanitizer, such that the signer signs the immutable message part with a regular signature scheme and the full message with the group signature scheme. The sanitizer can then update the full message and only sign this second component. The signer also takes over the role of the group manager in order to provide accountability.

4.1 Group Signatures

Group signatures, introduced by Chaum and van Heyst [13], allow a set of users to sign on behalf of the group such that outsiders cannot distinguish between different signers (anonymity) but such that a group manager can trace the signer's identity (traceability). We follow the formal framework of Bellare et al. [5] but add the non-frameability requirement [9] that even the group manager cannot sign on behalf of the users. Recall that this is necessary for the accountability in our sanitizable signature scheme, where the signer acts as the group manager and should not be able to make the judge falsely accuse the sanitizer.

We briefly recall group signature schemes and their security properties. For comprehensive definitions see the full version of the paper and [5]. A group signature scheme GS consists of six efficient algorithms $\mathsf{GS} = (\mathsf{GKGen}, \mathsf{UKGen}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{GJudge})$ where

- $(sk_{\text{user}}, pk_{\text{user}}) \leftarrow \mathsf{UKGen}(1^n)$ generates individual user key pairs,
- $(gmsk, gpk, \mathbf{cert}) \leftarrow \mathsf{GKGen}(1^n, \mathbf{gpk}_{user})$ takes the tuple $\mathbf{gpk}_{user}$ of the users' public keys and generates a group manager secret key $gmsk$, a group public key gpk and an individual certificate $cert_i$ for each user, where $\mathbf{cert}$ designates the tuple of all $cert_i$,
- $\sigma \leftarrow \mathsf{GSig}(sk_{\text{user},i}, cert_i, gpk, m)$ signs a message m given the user's secret data $sk_{\text{user},i}$, $cert_i$ and the group's public key gpk,
- $(i, \pi) \leftarrow \mathsf{Open}(gmsk, m, \sigma, \mathbf{gpk}_{user}, gpk)$ on input a message m and signature σ returns the index i of the alleged signer and a proof π such that
- $\mathsf{GJudge}(m, \sigma, i, \pi, gpk, \mathbf{gpk}_{user})$ either confirms the accusation or denies it.

There are three security properties for group signatures [5,9]:

Anonymity. Means that one cannot tell from a group signature who signed a message, even if one knows the secret data of the user and can ask the group manager to reveal the identities for other signatures.

Traceability. Refers to the fact that a malicious user cannot falsely accuse an honest user to be the signer of a message, even if the malicious user is allowed to see other signatures generated by this honest user and can call the group manager.

Non-Frameability. Strengthens traceability in the sense that even if the malicious user colludes with the group manager they cannot frame an honest user.

Definition 7 (Secure Group Signature). *We call a group signature scheme* secure *if it is anonymous and non-frameable.*

Note that we tailor the group signature definitions to our needs thereby adding non-frameability, making the scheme syntax setup session free and relaxing the security model concerning some technical issues which are discussed in the full version of this paper. As for instantiations we remark that the (generic) construction by Bellare et al. [5] satisfies our adapted definitions. As for more efficient group signature schemes, we can implement our sanitizable signature scheme with other group signature schemes like [22,14,18,19]. Yet, these group signature schemes need additional set-up assumptions like a trusted party generating common parameters or interactive registration of users. Our sanitizable signature scheme then inherits these characteristics (recall that, in practice, registration of signer and sanitizer keys is for example necessary to provide meaningful accountability).

4.2 Construction

In this section we show that the new security requirement of unlinkability can be achieved in combination with the five established security properties formalized in [3]. Recall that we sign the entire message, including the modifiable parts, with the group signature scheme, and —in order to prevent inadmissible changes— the signer also signs the fixed part with a regular scheme. This requires some care because if we take an arbitrary signature scheme then the signature itself may act as a unique identifier, even for messages with identical fixed parts. Thereby, unlinkability would be violated.

The solution is to use a secure deterministic signature scheme for the fixed part (such that the signature is identical for messages with the same fixed part). Alternatively, one can deploy a rerandomizable signature scheme such that the sanitizer can rerandomize the signature, excising the link to the input signature. Below we use the "deterministic solution" for simplicity, and since every secure signature scheme can be easily turned into a deterministic one via pseudorandom functions [15].

For a formal definition of *strongly unforgeable* signature schemes see [17]. We need this unforgeability notion (saying that one cannot even find a new signature for a previously signed message) to provide unlinkability. Examples of signature schemes achieving this strong notion are [6,12,4,2,8]. Moreover, it is possible to obtain a strongly unforgeable signature scheme out of any unforgeable signature scheme applying the transformation of Bellare and Shoup [7]. Applying the transformation of [15] one can then make such schemes also deterministic.

Recall that the idea behind our scheme is that for each signature the signer uses a group manager key, creates a certified user key to sign the modifiable parts, and certifies the sanitizer's public key as a group member to support modifications. But since our definition of sanitizable signatures demands state-free solutions, the signer formally cannot store the group manager key for this sanitizer and would need to create a new one for each call. We bypass this as follows: we let the signer for each signing request, including a public key of the sanitizer pk_{san}, create the group manager's keys etc. via the corresponding group signature algorithms, but provide the randomness for these algorithms by applying a pseudorandom function to pk_{san} (see [16] for a definition of pseudorandom functions). By this, we end up with (almost) independent keys for different sanitizers, but use consistent parameters for each sanitizer. For the same reason we also include the group membership certificate of the sanitizer in the signature, although it would be given directly to the sanitizer instead. As a side effect, since the group manager's public key is tied to the sanitizer in question, we also rely on group signatures with static joins only.

Construction 1 (Sanitizable Signature Scheme). *Let $\mathcal{S} = (\mathsf{SKGen}, \mathsf{SSign}, \mathsf{SVf})$ be a (regular) signature scheme, let $\mathsf{GS} = (\mathsf{GKGen}, \mathsf{UKGen}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{GJudge})$ be a group signature scheme. Let $\mathcal{PRF} = (\mathsf{KGen}_{prf}, \mathsf{PRF})$ be pseudorandom function. Define the sanitizable signature scheme* $\mathsf{SanSig} = (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$ *as follows:*

Key Generation. *First, algorithm KGen_{sig} gets the input 1^n and runs $(ssk, spk) \leftarrow \mathsf{SKGen}(1^n)$ to create a key pair for the signature scheme, and then also invokes $k \leftarrow \mathsf{KGen}_{prf}(1^n)$ to derive a key for the pseudorandom function. It outputs $(sk_{sig}, pk_{sig}) = ((ssk, k), spk)$. Algorithm $\mathsf{KGen}_{san}(1^n)$ generates a key pair $(sk_{san}, pk_{san}) = (gsk_{san}, gpk_{san}) \leftarrow \mathsf{UKGen}(1^n)$ of the group signature scheme.*

Signing. *Algorithm Sign on input $m \in \{0,1\}^*$, $sk_{sig} = (ssk, k)$, pk_{san}, ADM sets $m_{\mathrm{FIX}} = \mathrm{FIX}_{\mathrm{ADM}}(m)$ for the algorithm $\mathrm{FIX}_{\mathrm{ADM}}$ determined by ADM. It runs the user key generation algorithm $(gsk_{sig}, gpk_{sig}) \leftarrow \mathsf{UKGen}(1^n; \mathsf{PRF}(k, 0\|pk_{san}))$ for randomness $\mathsf{PRF}(k, 0\|pk_{san})$ and afterwards the group manager algorithm to compute*

$$(gmsk, gpk, cert_{sig}, cert_{san}) \leftarrow \mathsf{GKGen}(1^n, (gpk_{sig}, pk_{san}); \mathsf{PRF}(k, 1\|pk_{san}))$$

for randomness $\mathsf{PRF}(k, 1\|pk_{san})$. It computes

$$\sigma_{\mathrm{FIX}} = \mathsf{SSign}(ssk, (m_{\mathrm{FIX}}, \mathrm{ADM}, pk_{san}, gpk)) \text{ and}$$

$$\sigma_{\mathrm{FULL}} = \mathsf{GSig}(gsk_{sig}, cert_{sig}, (m, pk_{sig}), gpk)$$

using the signing algorithms of the regular and of the group signature scheme. The algorithm finally returns $\sigma = (\sigma_{\mathrm{FIX}}, \sigma_{\mathrm{FULL}}, \mathrm{ADM}, pk_{san}, cert_{san}, gpk)$.

Sanitizing. *Algorithm* Sanit *on input a message* m*, information* MOD*, a signature* $\sigma = (\sigma_{\text{FIX}}, \sigma_{\text{FULL}}, \text{ADM}, pk_{san}, cert_{san}, gpk)$*, keys* pk_{sig} *and* sk_{san} *first recovers* $m_{\text{FIX}} = \text{FIX}_{\text{ADM}}(m)$*. It then checks that* MOD *is admissible according to* ADM *and that* σ_{FIX} *is a valid signature for message* $(m_{\text{FIX}}, \text{ADM}, pk_{san}, gpk)$ *under key spk. If not, it stops outputting* $\perp$*. Else, it derives the modified message* $m' = \text{MOD}(m)$ *and computes*

$$\sigma'_{\text{FULL}} = \mathsf{GSig}(gsk_{san}, cert_{san}, (m', pk_{sig}), gpk)$$

and outputs m' *together with* $\sigma' = (\sigma_{\text{FIX}}, \sigma'_{\text{FULL}}, \text{ADM}, pk_{san}, cert_{san}, gpk)$*.*

Verification. *Algorithm* Verify *gets as input a message* $m \in \{0,1\}^*$*, a signature* $\sigma = (\sigma_{\text{FIX}}, \sigma_{\text{FULL}}, \text{ADM}, pk_{san}, cert_{san}, gpk)$ *and public keys* $pk_{sig} = spk$ *and* pk_{san}*. It first recovers* $m_{\text{FIX}} = \text{FIX}_{\text{ADM}}(m)$*. It then checks whether* $\mathsf{SVf}(spk, (m_{\text{FIX}}, \text{ADM}, pk_{san}, gpk), \sigma_{\text{FIX}}) = 1$ *and* $\mathsf{GVf}(gpk, (m, pk_{sig}), \sigma_{\text{FULL}})$ *verifies under the group public key as* $\mathtt{true}$*, too. If so, it outputs* 1*, declaring the entire signature as valid. Otherwise it returns* 0*, indicating an invalid signature.*

Proof. *Algorithm* Proof *gets as input* sk_{sig}*,* m *and* $\sigma = (\sigma_{\text{FIX}}, \sigma_{\text{FULL}}, \text{ADM}, pk_{san}, cert_{san}, gpk)$*. It parses the key as* $sk_{sig} = (ssk, k)$ *and recomputes*

$$(gmsk, gpk', cert'_{sig}, cert'_{san}) = \mathsf{GKGen}(1^n, (gpk_{sig}, pk_{san}); \mathsf{PRF}(k, 1||pk_{san}))$$

and checks that $gpk' = gpk$ *and* $cert'_{san} = cert_{san}$ *(and immediately returns* $\perp$ *if not). It next verifies that* $\mathsf{SVf}(spk, (m_{\text{FIX}}, \text{ADM}, pk_{san}, gpk), \sigma_{\text{FIX}}) = 1$ *and, if so, computes and outputs* $(i, \pi) \leftarrow \mathsf{Open}(gmsk, (m, pk_{sig}), \sigma_{\text{FULL}}, gpk)$*, where* $i \in \{\mathtt{Sig}, \mathtt{San}\}$ *is the identity returned by the* Open *algorithm (or,* Proof *returns* $\perp$ *if any of the verification steps above fail).*

Judge. *The judge on input* $m, \sigma, pk_{sig}, pk_{san}$ *and a proof* (i, π) *with* $i \in \{\mathtt{Sig}, \mathtt{San}\}$ *parses* σ *as* $(\sigma_{\text{FIX}}, \sigma_{\text{FULL}}, \text{ADM}, pk_{san}, cert_{san}, gpk)$*. It derives* $b \leftarrow \mathsf{GJudge}((m, pk_{sig}), \sigma_{\text{FULL}}, i, \pi, gpk)$ *using the judge algorithm of the group signature scheme. If* $b = \mathtt{true}$ *it outputs* i*, else it outputs* $i = \mathtt{Sig}$*.*

Completeness of signatures generated by the signer and sanitizer follows easily from the completeness of the underlying signature schemes and the fact that FIX_{ADM} leaves the fixed message parts unchanged for modified messages. There is a negligible probability that a signature of the signer or the sanitizer also verifies under the other party's other key, yielding possibly a wrong answer from the judge. We ignore this issue here for simplicity.

4.3 Security Proof

We need an additional property of the admissible modifications ADM: given arbitrary messages $m, m^* \in \{0,1\}^*$ (and a security parameter 1^n) it should be efficiently decidable whether $m^* \in \{\text{MOD}(m) \mid \text{MOD} \text{ with } \text{ADM}(\text{MOD}) = 1\}$ or not. We call such ADM *modification-decidable* and a sanitizable signature scheme *modification-restricted* if it only allows modification-decidable ADM. As an example consider again block-divided messages where ADM describes the block-length and the indices of changeable blocks. Then it is easy to check whether m^* has been changed in admissible blocks only or not.

Theorem 2. *Let $\mathcal{S}$ be a strongly unforgeable deterministic signature scheme and let GS be a secure group signature scheme. Assume further that $\mathcal{PRF}$ is a pseudorandom function. Then the modification-restricted sanitizable signature scheme in Construction 1 is unforgeable, immutable, private, proof-restricted transparent, accountable and unlinkable.*

As unlinkability implies privacy, and as moreover, sanitizer-accountability and signer-accountability imply unforgeability, it suffices to prove these two types of accountability as well as with unlinkability, immutability and (proof-restricted) transparency.

For the proof idea note that we can reduce transparency of our sanitizable signatures to anonymity of the underlying group signature scheme. Traceability of the group signature scheme enables the group manager (i.e., the signer) to provide a proof that a message has indeed been signed by a certain group member. Thus, if the sanitizer signs a message, the signer can produce evidence that this signature originates from the sanitizer. This shows sanitizer-accountability. Vice versa, the unframeability property of group signature scheme assures that the group manager (i.e., the signer) cannot falsely accuse a group member of having signed a message. Therefore, signer-accountability follows from unframeability.

The unforgeability of the underlying regular signature scheme assures immutability: If the sanitizer changed admissible parts of a message, she would be obliged to forge a signature for the fixed part. Unlinkability holds as the sanitizer creates a new group signature from scratch when sanitizing. Furthermore, the signature of the regular signature scheme remains unchanged, and is identical for different documents with the same fixed part because we use a deterministic scheme. The formal proof follows these ideas and appears in the full paper.

Acknowledgments

We thank the anonymous reviewers for valuable comments. Marc Fischlin, Anja Lehmann and Dominique Schröder are supported by the Emmy Noether Program Fi 940/2-1 of the German Research Foundation (DFG). This work was also supported by CASED (`www.cased.de`).

References

1. Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable Signatures. In: di Vimercati, S.D.C., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 159–177. Springer, Heidelberg (2005)
2. Boneh, D., Boyen, X.: Short Signatures Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
3. Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schroeder, 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)

4. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. Journal of Cryptology 17(4), 297–319 (2004)
5. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer, Heidelberg (2003)
6. Bellare, M., Rogaway, P.: The Exact Security of Digital Signatures - How to Sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
7. 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)
8. Boneh, D., Shen, E., Waters, B.: Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 229–240. Springer, Heidelberg (2006)
9. Bellare, M., Shi, H., Zhang, C.: Foundations of Group Signatures: The Case of Dynamic Groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Heidelberg (2005)
10. Canard, S., Jambert, A.: On Extended Sanitizable Signature Schemes. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 179–194. Springer, Heidelberg (2010)
11. Canard, S., Laguillaumie, F., Milhau, M.: Trapdoor Sanitizable Signatures and Their Application to Content Protection. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 258–276. Springer, Heidelberg (2008)
12. Coron, J.-S.: On the Exact Security of Full Domain Hash. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 229–235. Springer, Heidelberg (2000)
13. Chaum, D., van Heyst, E.: Group Signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 241–246. Springer, Heidelberg (1991)
14. Delerablée, C., Pointcheval, D.: Dynamic Fully Anonymous Short Group Signatures. In: Nguyên, P.Q. (ed.) VIETCRYPT 2006. LNCS, vol. 4341, pp. 193–210. Springer, Heidelberg (2006)
15. Goldreich, O.: Two Remarks Concerning the Goldwasser-Micali-Rivest Signature Scheme. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 104–110. Springer, Heidelberg (1987)
16. Goldreich, O.: The Foundations of Cryptography, vol. 1. Cambridge University Press, Cambridge (2001)
17. Goldreich, O.: The Foundations of Cryptography, vol. 2. Cambridge University Press, Cambridge (2004)
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.: Fully Anonymous Group Signatures Without Random Oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007)
20. Johnson, R., Molnar, D., Song, D.X., Wagner, D.: Homomorphic Signature Schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 244–262. Springer, Heidelberg (2002)
21. Klonowski, M., Lauks, A.: Extended Sanitizable Signatures. In: Rhee, M.S., Lee, B. (eds.) ICISC 2006. LNCS, vol. 4296, pp. 343–355. Springer, Heidelberg (2006)

22. Kiayias, A., Yung, M.: Group Signatures with Efficient Concurrent Join. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 198–214. Springer, Heidelberg (2005)
23. Miyazaki, K., Susaki, S., Iwamura, M., Matsumoto, T., Sasaki, R., Yoshiura, H.: Digital documents sanitizing problem. Technical Report ISEC2003-20. IEICE (2003)
24. Steinfeld, R., Bull, L., Zheng, Y.: Content Extraction Signatures. In: Kim, K.-c. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 285–304. Springer, Heidelberg (2002)

Confidential Signatures and Deterministic Signcryption

Alexander W. Dent[1], Marc Fischlin[2], Mark Manulis[2], Martijn Stam[3], and Dominique Schröder[2]

[1] Royal Holloway, University of London, U.K.
[2] Darmstadt University of Technology, Germany
[3] LACAL, EPFL, Switzerland

Abstract. Encrypt-and-sign, where one encrypts and signs a message in parallel, is usually not recommended for confidential message transmission as the signature may leak information about the message. This motivates our investigation of confidential signature schemes, which hide all information about (high-entropy) input messages. In this work we provide a formal treatment of confidentiality for such schemes. We give constructions meeting our notions, both in the random oracle model and the standard model. As part of this we show that full domain hash signatures achieve a weaker level of confidentiality than Fiat-Shamir signatures. We then examine the connection of confidential signatures to signcryption schemes. We give formal security models for deterministic signcryption schemes for high-entropy and low-entropy messages, and prove encrypt-and-sign to be secure for confidential signature schemes and high-entropy messages. Finally, we show that one can derandomize any signcryption scheme in our model and obtain a secure deterministic scheme.

1 Introduction

A common mistake amongst novice cryptographers is to assume that digital signature schemes provide some kind of confidentiality service to the message being signed. The (faulty) argument in support of this statement is (a) that all signature schemes are of the "hash-and-sign" variety, which apply a hash function to a message before applying any kind of keyed operation, and (b) that a one-way hash function will hide all partial information about a message. Both facets of this argument are incorrect. However, it does suggest that notions of confidentiality for signature schemes are an interesting avenue of research.

The question of confidentiality of hash functions in signature schemes was previously considered by Canetti [7] as "content-concealing signatures"; however the original treatment only serves to motivate the concept of perfect one-way hash functions [7,8]. We provide a more formal treatment here. The question of entropic security has been considered by several other authors. Dodis and Smith studied entropic secure primitives requiring that no function leaks their input [12]. Russell and Wang [22] consider the security of symmetric encryption schemes based on high-entropy messages, and several authors have considered the security of asymmetric encryption schemes based on high-entropy messages [3,4,6]. However, we are the first authors to consider the confidentiality of signatures and signcryption schemes in this scenario.

We believe that the concept of confidential signatures is intrinsically interesting and may prove to be useful in the construction of protocols in which two entities need to check that they are both aware of a particular message which (a) contains some confidential information, such as a password, and (b) contains a high entropy component, such as a confidential nonce.

Defining Confidential Signatures. Our first contribution is to define confidential signatures. Our starting point are high-entropy messages (signatures for messages with low entropy inevitably leak through the verification algorithm of the signature scheme). Our definitions are based on previous efforts for deterministic public-key encryption [3], and yield three models for confidential signature schemes:

– Weak confidentiality means that no information is leaked to a passive adversary, except possibly for information related to the technical details of the signature scheme.
– Mezzo confidentiality means that no information is leaked to a passive adversary (in possession of the verification key). Note that this is in contrast to deterministic public-key encryption where information cannot be hidden in such circumstances [3].
– Strong confidentiality means that no information is leaked to an active adversary (in possession of the verification key).

Our definitions are general enough to cover probabilistic and deterministic signature schemes, although we need an additional stipulation in the latter case, preventing the case where the leaked information is the unique signature itself.

Relation to Anonymous Signatures. There are similarities between confidential signatures and anonymous signatures [16,23]. Anonymous signatures hide the identity of the signer of a high-entropy message, whereas confidential signatures hide all the information about the message itself. The relationship between these two primitives is similar to the relationship between anonymous encryption and traditional public key encryption.

Constructing Confidential Signatures. We then show how to obtain confidential signatures. We first introduce the related concept of confidential hash functions, akin to hiding hash functions [3]. We prove that random oracles are confidential hash functions, as are perfectly one-way hash functions [7,8] in a weaker form.

We then show that the use of weakly confidential hash functions in full domain hash (FDH) signature schemes yields weakly confidential signatures. We show that FDH signature schemes and Fiat-Shamir signatures are confidential in the random oracle model. We also show that strongly secure confidential signatures can be obtained in the standard model via the use of a randomness extractor [19,20] (provided the message entropy lies above some fixed bound).

Applications to Signcryption. Secure message transmission is usually performed via the encrypt-then-sign paradigm, where the sender encrypts the message under the receiver's public encryption key and then signs the ciphertext with his own signing key. Signcryption schemes, introduced by [24], aim to gain efficiency by combining the two operations. One consequence of previous security definitions [1,2] is that the encrypt-and-sign approach, where one encrypts the message and signs the message in parallel, does not provide a secure signcryption in general as the signature may reveal information about the message.

We introduce security notions for (possibly deterministic) signcryption schemes with high-entropy messages, along the lines of deterministic public-key encryption and confidential signatures. In case of signcryption schemes, we can also give a low-entropy-message version and show that this definition is strictly stronger than the definitions for high-entropy messages. We show that the parallelizable encrypt-and-sign scheme is high-entropy confidential if the underlying encryption scheme is IND-CCA2 and the signature scheme is confidential (and deterministic). We finally prove that we can derandomize any signcryption scheme to derive a secure deterministic scheme.

Besides the fact that some of our results require the signcryption scheme to be deterministic, we also believe that deterministic signcryption schemes may be intrinsically more secure than many current schemes. The reason is that most of the current signcryption schemes are based on discrete-logarithm-based digital signature schemes which are highly sensitive to imperfect randomness [18].

In situations where we have been forced due to size constraints to omit a theorem's proof, the proof can be found in the full version of the paper [10].

2 Confidential Signature Schemes

We formalise the notion of a confidential signature in three ways and give constructions. These confidentiality notions can be applied to either probabilistic or deterministic signature schemes.

2.1 Definition of Confidential Signature Schemes

A digital signature scheme is a tuple of efficient algorithms $\mathtt{SS} = (\mathtt{SS.Setup}, \mathtt{SS.Kg}, \mathtt{SS.Sign}, \mathtt{SS.Ver})$. All algorithms (in this article) are probabilistic polynomial-time (PPT) in the security parameter k (which we assume clear from the context). The parameter generation algorithm produces a set of parameters common to all users $\lambda_{ss} \overset{R}{\leftarrow} \mathtt{SS.Setup}(1^k)$; subsequently the key generation algorithm produces a public/private key pair $(pk, sk) \overset{R}{\leftarrow} \mathtt{SS.Kg}(\lambda_{ss})$. (Until Section 4.2 we will silently assume that λ_{ss} allows retrieval of k and both pk and sk allow retrieval of λ_{ss}, simplifying notation.) The signing algorithm takes a message $m \in \{0,1\}^*$ and the private key, and outputs a signature $\sigma \overset{R}{\leftarrow} \mathtt{SS.Sign}(sk, m)$. The verification algorithm takes as input a message, signature and public key, and outputs either a valid symbol $\top$ or an invalid symbol $\perp$. This is written $\mathtt{SS.Ver}(pk, m, \sigma)$. The standard notion for signature security

```
$Expt_{\mathcal{A}}^{wSig-b}(k)$:
    $\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
    $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
    $(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}_1(\lambda_{ss})$
    $(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}_1(\lambda_{ss})$
    $\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m}_b)$
    $t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(pk, \boldsymbol{\sigma}^*)$
    If $t' = t_0$ then output 1
    Else return 0
```

```
$Expt_{\mathcal{A}}^{mSig-b}(k)$:
    $\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
    $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
    $(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}_1(pk)$
    $(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}_1(pk)$
    $\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m}_b)$
    $t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(pk, \boldsymbol{\sigma}^*)$
    If $t' = t_0$ then output 1
    Else return 0
```

```
$Expt_{\mathcal{A}}^{sSig-b}(k)$:
    $\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
    $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
    $(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}_1^{\texttt{SS.Sign}(sk,\cdot)}(pk)$
    $(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}_1^{\texttt{SS.Sign}(sk,\cdot)}(pk)$
    $\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m}_b)$
    $t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(pk, \boldsymbol{\sigma}^*)$
    If $t' = t_0$ then output 1
    Else return 0
```

Fig. 1. Notions of confidentiality for (a) weakly confidential signature schemes; (b) mezzo confidential signature schemes; (c) strongly confidential signature schemes. The signing algorithm is applied to the message vector $\boldsymbol{m}$ component-wise.

is that of unforgeability under chosen message attacks (see Appendix A.1 for formal definitions).

We present three confidentiality notions for a digital signature scheme — see Figure 1. These notions are split depending on the adversary's capabilities, which corresponds in a natural way to real-life scenarios where it may be possible to derive some information about a message from a signature which might be deemed practically useless, e.g., the value of the hash of the message, but leakage of which cannot be avoided.

In the weak confidentiality model, the attacker should not be able to determine any information about the messages apart from that which can be obtained directly from the signature itself. Mezzo confidentiality models the scenario where the attacker is able to retrieve public keys of the users, but cannot interact directly with their communication network and obtain signatures of messages. In the strong model, an active attacker should not be able to determine any information about the messages apart from the signature itself.

For $x \in \{w, m, s\}$, the attacker $\mathcal{A}$'s advantage in the $xSig$ game is defined to be:

$$Adv_{\mathcal{A}}^{xSig}(k) = |\Pr[Expt_{\mathcal{A}}^{xSig-0}(k) = 1] - \Pr[Expt_{\mathcal{A}}^{xSig-1}(k) = 1]|\,.$$

A signature scheme is weakly confidential (resp. mezzo confidential/strongly confidential) if all PPT attackers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ have negligible advantage $Adv_{\mathcal{A}}^{xSig}(k)$ in the *wSig* (resp. *mSig*/*sSig*) security game, subject to the following restraints:

- Pattern preserving: there exist a length function $\ell(k)$ and equality functions $\diamond_{ij} \in \{=, \neq\}$ $(1 \le i, j \le \ell(k))$ such that for any admissible input a in the corresponding game and all possible $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(a)$ we have that $|\boldsymbol{m}| = \ell(k)$ and $m_i \diamond_{ij} m_j$.
- High entropy: the function $\pi(k) = \max_{m \in \{0,1\}^*} \Pr[m_i = m : (\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(a)]$ is negligible, where the probability is over $\mathcal{A}_1$'s random tape only (and $i \in \mathbb{N}$ and all choices of the other algorithms are fixed). The value $\mu(k) = -\log_2 \pi(k)$ is termed the adversary's *min entropy.*

```
$\texttt{SS.Kg}'(\lambda_{ss})$:
  $r \xleftarrow{R} \{0,1\}^k$
  $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
  Return $(pk\|r, sk\|r)$

$\texttt{SS.Sign}'(sk\|r, m)$:
  If $m = m'\|r$
    Return $\texttt{SS.Sign}(sk, m)\|m$
  Else
    Return $\texttt{SS.Sign}(sk, m)$

$\texttt{SS.Ver}'(pk\|r, m, \sigma)$:
  If $m = m'\|r$
    Parse $\sigma$ as $\sigma'\|m$
    $\sigma \leftarrow \sigma'$
  Return $\texttt{SS.Ver}(pk, m, \sigma)$
```

Fig. 2. A signature scheme which is weakly confidential but not mezzo confidential

For deterministic schemes we need the following additional constraint, ruling out trivial attacks:

– Signature free: $\mathcal{A}_1$ does not output a message $m_i \in \boldsymbol{m}$ where it has queried the signature oracle on m_i. (This security requirement only affects strongly confidential signature schemes.)

The latter condition prevents an attacker against a deterministic scheme from "winning" by setting $t \leftarrow \texttt{SS.Sign}(sk, m)$ — i.e., it prevents the attacker from "winning" the game simply by determining that the message m has the property that its unique signature is $\texttt{SS.Sign}(sk, m)$.

The notions of confidentiality are strictly increasing in strength. If SS is a weakly confidential signature schemes, then Figure 2 depicts a scheme which is weakly confidential but not mezzo confidential. Similarly, if SS is a mezzo confidential signature scheme, then Figure 3 shows a scheme which is mezzo confidential but not strongly confidential.

```
$\texttt{SS.Kg}'(\lambda_{ss})$:
  $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
  $r \xleftarrow{R} \{0,1\}^k$
  $\sigma_r \leftarrow \texttt{SS.Sign}(sk, 0\|r)$
  Return $(pk, sk\|r\|\sigma_r)$

$\texttt{SS.Sign}'(sk\|r\|\sigma_r, m)$:
  If $m = m'\|r\|\sigma_r$
    Set $\sigma' \leftarrow \texttt{SS.Sign}(sk, 1\|m)$
    Return $\sigma = (\sigma', m)$
  Else
    Set $\sigma \leftarrow \texttt{SS.Sign}(sk, 2\|m)$
    Return $\sigma = (\sigma', r, \sigma_r)$

$\texttt{SS.Ver}'(pk, m, \sigma)$:
  If $\sigma = (\sigma', m')$
    Parse $m'$ as $m' = m''\|r'\|\sigma'_r$
    Return $\top$ iff
      $\texttt{SS.Ver}(pk, 1\|m', \sigma') = \top$, and
      $m = m'$, and
      $\texttt{SS.Ver}(pk, 0\|r', \sigma'_r) = \top$
  If $\sigma = (\sigma', r', \sigma'_r)$
    Return $\top$ iff
      $\texttt{SS.Ver}(pk, 2\|m, \sigma') = \top$, and
      $m \neq m''\|r'\|\sigma'_r$ for any $m'' \in \{0,1\}^*$,
      and $\texttt{SS.Ver}(pk, 0\|r', \sigma'_r) = \top$
  Else return $\perp$
```

Fig. 3. A signature scheme which is mezzo confidential but not strongly confidential

3 Confidential Hash Functions and Signature Schemes

3.1 Confidential Hash Functions

We recap the notion of a *hiding* hash function by Bellare *et al.* [3], but call such functions confidential here. For our purposes, a hash function $\mathsf{H} = (\texttt{H.Kg}, \texttt{H})$ is

```
$Expt_{\mathcal{A}}^{\text{wHash-b}}(k)$:
  $\mathtt{H} \xleftarrow{R} \mathtt{H.Kg}(1^k)$
  $(\boldsymbol{x}_0, t_0) \xleftarrow{R} \mathcal{A}_1(1^k)$
  $(\boldsymbol{x}_1, t_1) \xleftarrow{R} \mathcal{A}_1(1^k)$
  $\boldsymbol{h} \leftarrow \mathtt{H}(\boldsymbol{x}_b)$
  $t' \xleftarrow{R} \mathcal{A}_2(\mathtt{H}, \boldsymbol{h})$
  If $t' = t_0$ then output 1
  Else return 0
```

```
$Expt_{\mathcal{A}}^{\text{sHash-b}}(k)$:
  $\mathtt{H} \xleftarrow{R} \mathtt{H.Kg}(1^k)$
  $(\boldsymbol{x}_0, t_0) \xleftarrow{R} \mathcal{A}_1(\mathtt{H})$
  $(\boldsymbol{x}_1, t_1) \xleftarrow{R} \mathcal{A}_1(\mathtt{H})$
  $\boldsymbol{h} \leftarrow \mathtt{H}(\boldsymbol{x}_b)$
  $t' \xleftarrow{R} \mathcal{A}_2(\mathtt{H}, \boldsymbol{h})$
  If $t' = t_0$ then output 1
  Else return 0
```

Fig. 4. Notions of confidentiality for (a) weakly confidential hash functions; (b) strongly confidential hash functions. The hash function is applied to the data vector $\boldsymbol{x}$ component-wise.

a PPT pair of algorithms for key generation and hashing, respectively. We will identify the description output by the key generation algorithm $\mathtt{H.Kg}$ with the hash function $\mathtt{H}$ itself. The collision-finding advantage $Adv_{\mathcal{A}}^{col}$ of an attacker $\mathcal{A}$ against a hash function H is defined as

$$Adv_{\mathsf{H},\mathcal{A}}^{col}(k) = \Pr\left[\begin{array}{l}\mathtt{H}(x;r) = \mathtt{H}(x';r') \\ \text{and } (x,r) \neq (x',r')\end{array} : (x,x',r,r') \xleftarrow{R} \mathcal{A}(\mathtt{H});\ \mathtt{H} \xleftarrow{R} \mathtt{H.Kg}(1^k)\right] .$$

The hash function H is called *collision-resistant* if all PPT attackers $\mathcal{A}$ have negligible advantage $Adv_{\mathsf{H},\mathcal{A}}^{col}(k)$ (as a function of k). We require that the hash function is hiding/confidential against an attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ playing one of the games in Figure 4. For $x \in \{w, s\}$ the attacker's advantage is defined to be

$$Adv_{\mathsf{H},\mathcal{A}}^{\text{xHash}}(k) = |\Pr[Expt_{\mathcal{A}}^{\text{xHash-0}}(k) = 1] - \Pr[Expt_{\mathcal{A}}^{\text{xHash-1}}(k) = 1]| .$$

A hash function is *weakly (resp. strongly) confidential* if every PPT attacker $\mathcal{A}$ has negligible advantage in the corresponding game subject to the following restraints:

– Pattern preserving: there exist a length function $\ell(k)$ and equality functions $\diamond_{ij} \in \{=, \neq\}$ $(1 \leq i, j \leq \ell(k))$ such that for all possible $(\boldsymbol{x}, t) \xleftarrow{R} \mathcal{A}_1(1^k)$ we have that $|\boldsymbol{x}| = \ell(k)$ and $x_i \diamond_{ij} x_j$.
– High entropy: the function $\pi(k) = \max_{x \in \{0,1\}^*} \Pr[x_i = x : (\boldsymbol{x}, t) \xleftarrow{R} \mathcal{A}_1(a)]$ is negligible where the probability is only over $\mathcal{A}_1$'s random tape. We define $\mu(k) = -\log_2 \pi(k)$ to be the adversary's *minimum entropy.*

Note that collision-resistant deterministic hash functions cannot achieve strong confidentiality because an adversary $\mathcal{A}_1$ can set $t = \mathtt{H}(x)$ for some message x and $\mathcal{A}_2$ can easily obtain this value from the hash vector $\boldsymbol{h}$. We also note that for "unkeyed" hash functions both notions are equivalent and so no unkeyed, deterministic hash function can be considered confidential (unless the hash function is almost constant).

In the random oracle model, where the adversary is granted oracle access to the hash function $\mathtt{H}$ instead of receiving the description as input, we give

$\mathcal{A}_1$ access to the random oracle in the strong case, but deny $\mathcal{A}_1$ access to $\mathtt{H}$ in the weak case. It is easy to see that a random oracle thus achieves weak confidentiality, whereas the above attack on deterministic functions still applies in the strong case. However, under the additional constraint that $\mathcal{A}_1$ does not query $\mathtt{H}$ about any x in its output $\boldsymbol{x}$ (*hash-free adversaries*) a random oracle is also strongly confidential:

Proposition 1 (Confidentiality of Random Oracles). *For any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ where $\mathcal{A}_1$ outputs vectors of length $\ell(k)$ and with min-entropy $\mu(k) = -\log \pi(k)$, and where $\mathcal{A}_2$ makes at most $q_h(k)$ queries to the random oracle, we have*

$$Adv^{xHash}_{\mathtt{H},\mathcal{A}}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot \pi(k)$$

for $x \in \{w, s\}$ where $\mathcal{A}$ is assumed to be hash-free (in the strong case).

As for constructions in the standard model, we note that perfectly one-way functions (POWs) [7,8] provide a partial solution. POWs have been designed to hide all information about preimages, akin to our confidentiality notion. However, all known constructions of POWs are only good for fixed (sets of) input distributions where the distributions can depend only on the security parameter but not the hash function description. Furthermore, known POWs usually require the conditional entropy of any x_i to be high, given the other x_j's. In light of this, any $\ell(k)$-valued perfectly one-way function [8] is a weakly confidential hash function. Hence, we can build such hash functions based, for example, on claw-free permutations [8] or one-way permutations [8,15].

3.2 Full-Domain Hash Signatures

A *full-domain hash (FDH) signature scheme* FDH *for deterministic hash function* H is a signature scheme in which the signing algorithm computes a signature as $\sigma = f(\mathtt{H}(m))$ for some secret function f, and the verification algorithm checks that $g(\sigma) = \mathtt{H}(m)$ for some public function g. More formally (assuming that $\mathtt{FDH.Setup}(1^k)$ outputs $\lambda_{ss} = 1^k$ and that there exists a PPT algorithm which generates the functions $(f, g) \leftarrow \mathtt{FDH.Kg'}(\lambda_{ss})$):

```
FDH.Kg($\lambda_{ss}$):
  $(f, g) \leftarrow$ FDH.Kg'($\lambda_{ss}$)
  H $\leftarrow$ H.Kg($1^k$)
  $(pk, sk) = ((g, \mathtt{H}), (f, \mathtt{H}))$
  Return $(pk, sk)$

FDH.Sign($sk, m$):
  Parse $sk$ as $(f, \mathtt{H})$
  Return $\sigma = f(\mathtt{H}(m))$

FDH.Ver($pk, m, \sigma$):
  Parse $pk$ as $(g, \mathtt{H})$
  Return $\top$ if $\mathtt{H}(m) = g(\sigma)$
  Otherwise return $\bot$
```

Unforgeability of FDH signatures in the ROM has been shown in [5,9].

Proposition 2 (Weak Confidentiality of FDH). *The FDH-signature scheme* FDH *for hash function* H *is weakly confidential if* H *is weakly confidential. More precisely, for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ against the weak confidentiality of* FDH, *where $\mathcal{A}_1$ outputs $\ell(k)$ messages and $\mathcal{A}_2$ makes at most $q_s(k)$ signature* *queries, there exists an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the weak confidentiality of the hash function such that*

$$Adv^{wSig}_{\mathsf{FDH},\mathcal{A}}(k) \leq Adv^{wHash}_{\mathtt{H},\mathcal{B}}(k),$$

where $\mathcal{B}_1$'s running time is identical to the one of $\mathcal{A}_1$, and $\mathcal{B}_2$'s running time is the one of $\mathcal{A}_2$ plus $Time_{\mathtt{FDH.Kg}}(k) + (q_s + \ell(k)) \cdot Time_{\mathtt{FDH.Sign}}(k) + O(k)$.

The proof actually shows that the signature scheme remains confidential for an adversarially chosen key pair (f, g), i.e., confidentiality only relies on the confidentiality of the hash function. Moreover, by Proposition 1, we have that FDH-signature schemes are weakly confidential in the random oracle model.

Proof. Assume that FDH is not weakly confidential and that there exists an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ successfully breaking this property. Then we construct an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the weak confidentiality of the hash function as follows. Adversary $\mathcal{B}_1$ on input 1^k runs $\mathcal{A}_1$ on input 1^k and outputs this algorithm's answer $(\boldsymbol{m}, t)$.

Algorithm $\mathcal{B}_2$ receives as input a description $\mathtt{H}$ of the confidential hash function and a vector $\boldsymbol{h}$ of hash values. $\mathcal{B}_2$ runs $(f, g) \leftarrow \mathtt{FDH.Kg}'(1^k)$, sets $pk \leftarrow (g, \mathtt{H})$ and $sk \leftarrow (f, \mathtt{H})$, and computes signatures $\boldsymbol{\sigma}^* = f(\boldsymbol{h})$. It invokes $\mathcal{A}_2$ on $(1^k, pk, \boldsymbol{\sigma}^*)$ and answers each subsequent signature request for message m by computing $\sigma = \mathtt{FDH.Sign}(sk, m)$. When $\mathcal{A}_2$ outputs t' algorithm $\mathcal{B}_2$ copies this output and stops.

It is easy to see that $\mathcal{B}$'s advantage attacking the confidentiality of the hash function is identical to $\mathcal{A}$'s advantage attacking the confidentiality of the FDH signature scheme (the fact that $\mathcal{A}_1$ preserves pattern and produces high-entropy messages carries over to $\mathcal{B}_1$). □

No (unforgeable) FDH-signature scheme is mezzo confidential, because a signature on the message m leaks the value $\mathtt{H}(m)$. More formally, an attacker $\mathcal{A}_1$ can pick a message $m \overset{R}{\leftarrow} \{0,1\}^k$ and set $t \leftarrow \mathtt{H}(m)$. Adversary $\mathcal{A}_2$ then receives $\sigma \leftarrow f(\mathtt{H}(m))$ and can recover $t = \mathtt{H}(m)$ by computing $g(\sigma)$.

3.3 Strongly Confidential Signatures in the ROM

Recall from the previous section that FDH signatures leak the hash value of a message. To prevent this, we make the hashing process probabilistic and compute $(r, \mathtt{H}(r, m))$ for randomness r. Then $\mathcal{A}_1$ cannot predict the hash values of the challenge messages due to r (which becomes public only afterwards) and $\mathcal{A}_2$ cannot guess the hash values due to the entropy in the message m (even though r is then known). Our instantiation is shown in Figure 5.

Proposition 3 (Random Oracle Instantiation). *If $\mathtt{H}$ is a hash function modeled as a random oracle, then the signature scheme SS' is strongly confidential. That is, for any attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ against the strong confidentiality of the signature scheme SS', where $\mathcal{A}_1$ outputs a vector of length $\ell(k)$ and with*

Suppose $\mathsf{SS} = (\mathtt{SS.Setup}, \mathtt{SS.Kg}, \mathtt{SS.Sign}, \mathtt{SS.Ver})$ is a signature scheme. We define a new signature scheme SS' as follows (where $\mathtt{SS.Setup}' \equiv \mathtt{SS.Setup}$):

```
$\mathtt{SS.Kg}'(\lambda_{ss})$:
  $(pk, sk) \leftarrow \mathtt{SS.Kg}(\lambda_{ss})$
  $\mathtt{H} \xleftarrow{R} \mathtt{H.Kg}(1^k)$
  $pk' \leftarrow (pk, \mathtt{H})$; $sk' \leftarrow (sk, \mathtt{H})$
  Return $(pk', sk')$

$\mathtt{SS.Sign}'(sk', m)$:
  Parse $sk'$ as $(sk, \mathtt{H})$
  $r \xleftarrow{R} \{0,1\}^k$
  $h \leftarrow \mathtt{H}(r, m)$
  $\sigma' \leftarrow \mathtt{SS.Sign}(sk, h)$
  $\sigma \leftarrow (\sigma', r)$
  Return $\sigma$

$\mathtt{SS.Ver}'(pk', m, \sigma)$:
  Parse $pk'$ as $(pk, \mathtt{H})$
  Parse $\sigma$ as $(\sigma', r)$
  Return $\mathtt{SS.Ver}(pk, \mathtt{H}(r, m), \sigma')$
```

Fig. 5. Construction of a strongly confidential signature scheme in the ROM

min-entropy $\mu(k) = -\log \pi(k)$, and where $\mathcal{A}_2$ asks at most q_h oracle queries (signing queries and direct hash oracle queries), we have

$$Adv^{sSig}_{\mathsf{SS}',\mathcal{A}}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot (2^{-k} + \pi(k))\,.$$

Clearly, the scheme is also (strongly) unforgeable if the underlying signature scheme is (strongly) unforgeable.

3.4 Fiat-Shamir Signature Schemes

Our second instantiation is based upon the Fiat-Shamir paradigm [14] that turns every (three-round) identification scheme into a signature scheme. An identification scheme (ID scheme) is defined by a triplet (G, S, R), where G is a key generation algorithm and the sender S wishes to prove his identity to the receiver R. More formally: $G(1^k)$ is an efficient algorithm that outputs a key pair (ipk, isk). $(S(isk), R(ipk))$ are interactive algorithms and it is required that $\Pr[\,(S(isk), R(ipk)) = 1] = 1$ (where the probability is taken over the coin tosses of S, R and G). A canonical ID scheme is a 3-round ID scheme $(\alpha; \beta; \gamma)$ in which α is sent by the sender S, β by the receiver R and consists of R's random coins, and γ is sent by the sender. For a sender S with randomness r, we denote $\alpha = S(isk; r)$ and $\gamma = S(isk, \alpha, \beta; r)$. The Fiat-Shamir signature scheme is given in Figure 6.

In order to prove the confidentiality of this scheme, we need to assume that the commitment α of the Fiat-Shamir scheme has non-trivial entropy. This can always be achieved by appending public randomness.

Proposition 4 (Fiat-Shamir Instantiation). *If $\mathtt{H}$ is a hash function modeled as a random oracle, then the Fiat-Shamir instantiation SS'' for non-trivial commitments is strongly confidential. More precisely, for any attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ against the strong confidentiality of the signature scheme SS'' where $\mathcal{A}_1$ outputs a message vector of length $\ell(k)$ with min-entropy $\mu(k) = -\log \pi(k)$, α has min-entropy $\mu'(k) = -\log \pi'(k)$, and $\mathcal{A}_2$ asks at most q_h oracle queries (signing queries and direct hash oracle queries), we have*

$$Adv^{sSig}_{\mathsf{SS}'',\mathcal{A}}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot (\pi(k) + \pi'(k)).$$

Suppose (G, S, R) is a canonical identification scheme and H is a hash function family. We define the signature scheme $\mathsf{SS}'' = (\texttt{SS.Setup}'', \texttt{SS.Kg}'', \texttt{SS.Sign}'', \texttt{SS.Ver}'')$ as follows (where $\texttt{SS.Setup}(1^\lambda)$ returns $\lambda_{ss} = 1^\lambda$):

```
SS.Kg''(λ_ss):
    $(ipk, isk) \leftarrow G(\lambda_{ss})$
    $\mathtt{H} \xleftarrow{R} \mathtt{H.Kg}(1^k)$
    $pk' \leftarrow (ipk, \mathtt{H})$; $sk' \leftarrow (isk, \mathtt{H})$
    Return $(pk', sk')$

SS.Sign''(sk', m):
    Parse $sk'$ as $(isk, \mathtt{H})$
    $r \xleftarrow{R} \{0,1\}^k$
    $\alpha \leftarrow S(isk; r)$
    $\beta \leftarrow \mathtt{H}(\alpha, m)$
    $\gamma \leftarrow S(isk, \alpha, \beta; r)$
    $\sigma \leftarrow (\alpha, \beta, \gamma)$
    Return $\sigma$

SS.Ver''(pk', m, σ):
    Parse $pk'$ as $(ipk, \mathtt{H})$
    Parse $\sigma$ as $(\alpha, \beta, \gamma)$
    $\beta' \leftarrow \mathtt{H}(\alpha, m)$
    Return 1 iff $\beta = \beta'$
        and $R(ipk, \alpha, \beta, \gamma) = 1$
```

Fig. 6. The Fiat-Shamir paradigm that turns every ID scheme into a signature scheme

3.5 Strongly Confidential Signatures from Randomness Extraction

Our instantiation in the standard model relies on randomness extractors [19,20] and is depicted in Figure 7. The main idea is to smooth the distribution of the message via an extractor, and to sign the almost uniform value h.

Recall that a strong (a, b, n, t, ϵ)-extractor is an efficient algorithm $\mathsf{Ext} : \{0,1\}^a \times \{0,1\}^b \to \{0,1\}^n$ which takes some random input $m \in \{0,1\}^a$ (sampled according to some distribution with min-entropy at least t) and some randomness $r \in \{0,1\}^b$. It outputs $h \leftarrow \mathsf{Ext}(m, r)$ such that the statistical distance between (r, h) and (r, u) is at most ϵ for uniform random values $r \in \{0,1\}^b$ and $u \in \{0,1\}^n$.

To ensure unforgeability we need to augment the extractor's extraction property by collision-resistance, imposing the requirement that the extractors be keyed and introducing dependency of the extractor's parameters a, b, n, t, ϵ on the security parameter k. For a survey about very efficient constructions of such collision-resistant extractors see [11].

In order to use extractors, we need a stronger assumption on the message distribution: we assume that the adversary $\mathcal{A}_1$ now outputs vectors of messages such that each message in the vector has min-entropy greater than some fixed bound $\mu(k)$ *given the other messages*. Observe that the collision-resistance requirement on the extractor implies that μ must be super-logarithmic. We say that the output has *conditional min-entropy* $\mu(k)$.

Proposition 5 (Extractor Instantiation). *If* Ext *is an* (a, b, n, t, ϵ)*-extractor then the extractor instantiation of* SS''' *is strongly confidential. More specifically, for any attacker* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against the strong confidentiality of the signature scheme* SS'''*, where* $\mathcal{A}_1$ *outputs a vector of length* $\ell(k)$ *with conditional min-entropy* $\mu(k) \geq t(k)$*, we have*

$$Adv^{sSig}_{\mathsf{SS}''',\mathcal{A}}(k) \leq 2 \cdot \ell(k) \cdot \epsilon(k).$$

Note that our construction of the randomness extractor operates on messages of a fixed length of $a(k)$ input bits, and the signature length depends on this

Suppose $\mathsf{SS} = (\texttt{SS.Setup}, \texttt{SS.Kg}, \texttt{SS.Sign}, \texttt{SS.Ver})$ is a signature scheme. We define a new signature scheme SS''' as follows (where $\texttt{SS.Setup}''' \equiv \texttt{SS.Setup}$):

```
$\texttt{SS.Kg}'''(\lambda_{ss})$:
  $(pk, sk) \leftarrow \texttt{SS.Kg}(\lambda_{ss})$
  Choose an extractor $\mathsf{Ext}$
  $pk' \leftarrow (pk, \mathsf{Ext})$
  $sk' \leftarrow (sk, \mathsf{Ext})$
  Return $(pk', sk')$

$\texttt{SS.Sign}'''(sk', m)$:
  Parse $sk'$ as $(sk, \mathsf{Ext})$
  $r \overset{R}{\leftarrow} \{0,1\}^b$
  $h \leftarrow \mathsf{Ext}(m, r)$
  $\sigma' \leftarrow \texttt{SS.Sign}(sk, h)$
  $\sigma \leftarrow (\sigma', r)$
  Return $\sigma$

$\texttt{SS.Ver}'''(pk', m, \sigma)$:
  Parse $pk'$ as $(pk, \mathsf{Ext})$
  Parse $\sigma$ as $(r, \sigma')$
  Set $h \leftarrow \mathsf{Ext}(m, r)$
  Return $\texttt{SS.Ver}(pk, h, \sigma')$
```

Fig. 7. Construction of strongly confidential signature scheme based on randomness extractors

value $a(k)$. To process larger messages we can first hash input messages with a collision-resistant hash function, before passing it to the extractor. In this case, some care must be taken to determine a correct bound for the entropy lost through the hash function computation.

4 Deterministic Signcryption

Signcryption is a public-key primitive which aims to simultaneously provide message confidentiality and message integrity. Signcryption was introduced by Zheng [24] and security models were independently introduced by An, Dodis and Rabin [1] and by Baek, Steinfeld and Zheng [2]. Similar to public-key encryption, achieving confidentiality in the formal security models requires that signcryption is a randomised process; however, we may also consider the confidentiality of deterministic signcryption schemes on high-entropy message spaces. We will also see that a practical version of confidentiality may even be achieved by a deterministic signcryption scheme for low entropy message distributions.

4.1 Notions of Confidentiality for Signcryption Schemes

A signcryption scheme is a tuple of PPT algorithms $\mathsf{SC} = (\texttt{SC.Setup}, \texttt{SC.Kg}_\texttt{s}, \texttt{SC.Kg}_\texttt{r}, \texttt{SC.SignCrypt}, \texttt{SC.UnSignCrypt})$. The setup algorithm generates public parameters $\lambda_{sc} \overset{R}{\leftarrow} \texttt{SC.Setup}(1^k)$ common to all algorithms. We will generally assume that all algorithms take λ_{sc} as an implicit input, even if it is not explicitly stated. The sender key-generation algorithm generates a key pair for the sender $(pk_S, sk_S) \overset{R}{\leftarrow} \texttt{SC.Kg}_\texttt{s}(\lambda_{sc})$ and the receiver key-generation algorithm generates a key pair for a receiver $(pk_R, sk_R) \overset{R}{\leftarrow} \texttt{SC.Kg}_\texttt{r}(\lambda_{sc})$. The signcryption algorithm takes as input a message $m \in \mathcal{M}$, the sender's private key sk_S, and the receiver's public key pk_R, and outputs a signcryption ciphertext $C \overset{R}{\leftarrow} \texttt{SC.SignCrypt}(sk_S, pk_R, m)$. The unsigncryption algorithm takes as input a ciphertext $C \in \mathcal{C}$, the sender's public key pk_S, and the receiver's private key sk_R, and outputs either a message $m \overset{R}{\leftarrow} \texttt{SC.UnSignCrypt}(pk_S, sk_R, C)$ or an error symbol $\perp$.

It is interesting to consider the basic attack on a deterministic signcryption scheme. In such an attack, the attacker picks two messages (m_0, m_1) and receives a signcryption C^* of the message m_b. The attacker checks whether C^* is the signcryption of m_0 by requesting the signcryption of m_0 from the signcryption oracle. As in the case of public-key encryption, we may prevent this basic attack by using a high-entropy message space and so prevent the attacker being able to determine which message to query to the signcryption oracle. However, unlike the case of public-key encryption, we may also prevent this attacker by forbidding the attacker to query the signcryption oracle on m_0 and m_1. We can therefore differentiate between the high-entropy case (in which the message distribution chosen by the attacker has high entropy) and the low-entropy case (in which the attacker is forbidden from querying the signcryption oracle on a challenge message).

We give definitions for the high-entropy and low-entropy confidentiality in Figure 8. In both cases, i.e. for $x \in \{h, l\}$, the attacker's advantage is defined as

$$Adv^{xSCR}_{\mathsf{SS},\mathcal{A}}(k) = |\Pr[Expt^{xSCR-1}_{\mathcal{A}} = 1] - \Pr[Expt^{xSCR-0}_{\mathcal{A}} = 1]| \,.$$

A signcryption scheme is high-entropy confidential if every PPT attacker $\mathcal{A}$ has negligible advantage in the hSCR game subject to the following restrictions:

– Strongly pattern preserving: there exists a length function $\ell(k)$, message length functions $q_i(k)$, and equality functions $\diamond_{ij} \in \{=, \neq\}$ $(1 \leq i, j \leq \ell(k))$ such that for all possible $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(\lambda_{sc}, pk^*_S, pk^*_R)$ we have that $|\boldsymbol{m}| = \ell(k)$, $|m_i| = q_i(k)$ and $m_i \diamond_{ij} m_j$.
– High entropy: the function $\pi(k) = \max_{m \in \{0,1\}^*} \Pr[m_i = m : (\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(a)]$ is negligible where the probability is only over $\mathcal{A}_1$'s random tape. The value $\mu(k) = -\log \pi(k)$ is known as the adversary's minimum entropy.
– Signature free: $\mathcal{A}_1$ does not output a message $m_i \in \boldsymbol{m}$ where it has queried the signcryption oracle on the pair (pk^*_R, m_i).
– Non-trivial: $\mathcal{A}_2$ does not query the unsigncryption oracle on any pair (pk^*_S, C) where $C \in \boldsymbol{C}^*$.

A signcryption scheme is low-entropy confidential if every PPT attacker $\mathcal{A}$ has negligible advantage in the lSCR game subject to the restrictions that $\mathcal{A}$ never queries the encryption oracle on either (pk^*_R, m_0) or (pk^*_R, m_1), and $\mathcal{A}_2$ never queries the decryption oracle on (pk^*_S, C^*).

Proposition 6. *Any deterministic signcryption scheme* SC *which is low-entropy confidential is also high-entropy confidential. In particular, for any adversary* $\mathcal{A}$ *against high-entropy confidentiality, making at most* $q_s(k)$ *signcryption queries and where* $\mathcal{A}_1$ *outputs* $\ell(k)$ *messages with min-entropy* $\mu(k) = -\log \pi(k)$*, there exists an adversary* $\bar{\mathcal{A}}$ *such that*

$$Adv^{hSCR}_{\mathsf{SC},\mathcal{A}}(k) \leq \ell(k) \cdot Adv^{lSCR}_{\mathsf{SC},\bar{\mathcal{A}}}(k) + 4 \cdot q_s(k) \cdot \ell(k) \cdot \pi(k),$$

where the running time of $\bar{\mathcal{A}}$ *equals the time of* $\mathcal{A}$ *plus* $O(k)$*.*

```
$Expt^{hSCR-b}_{\mathcal{A}}(k)$:
  $\lambda_{sc} \xleftarrow{R} \texttt{SC.Setup}(1^k)$
  $(pk^*_S, sk^*_S) \xleftarrow{R} \texttt{SC.Kg}_\texttt{s}(\lambda_{sc})$
  $(pk^*_R, sk^*_R) \xleftarrow{R} \texttt{SC.Kg}_\texttt{r}(\lambda_{sc})$
  $(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}^{\mathcal{O}}_1(\lambda_{sc}, pk^*_S, pk^*_R)$
  $(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}^{\mathcal{O}}_1(\lambda_{sc}, pk^*_S, pk^*_R)$
  $\boldsymbol{C}^* \leftarrow \texttt{SC.SignCrypt}(\lambda_{sc}, sk^*_S, pk^*_R, \boldsymbol{m}_b)$
  $t' \xleftarrow{R} \mathcal{A}^{\mathcal{O}}_2(\lambda_{sc}, pk^*_S, pk^*_R, \boldsymbol{C}^*)$
  If $t' = t_0$ then output 1
  Else return 0
```

```
$Expt^{lSCR-b}_{\mathcal{A}}(k)$:
  $\lambda_{sc} \xleftarrow{R} \texttt{SC.Setup}(1^k)$
  $(pk^*_S, sk^*_S) \xleftarrow{R} \texttt{SC.Kg}_\texttt{s}(\lambda_{sc})$
  $(pk^*_R, sk^*_R) \xleftarrow{R} \texttt{SC.Kg}_\texttt{r}(\lambda_{sc})$
  $(m_0, m_1, \omega) \xleftarrow{R} \mathcal{A}^{\mathcal{O}}_1(\lambda_{sc}, pk^*_S, pk^*_R)$
  $C^* \leftarrow \texttt{SC.SignCrypt}(\lambda_{sc}, sk^*_S, pk^*_R, m_b)$
  $b' \xleftarrow{R} \mathcal{A}^{\mathcal{O}}_2(C^*, \omega)$
  Output $b'$
```

Fig. 8. Notions of confidentiality for (a) high-entropy signcryption schemes and (b) low-entropy signcryption schemes. Note that $\mathcal{A}_1$ may pass the state information ω to $\mathcal{A}_2$ in the lSCR game. The attacker's have access to a signcryption oracle $\texttt{SC.SignCrypt}(sk^*_S, \cdot, \cdot)$ and an unsigncryption oracle $\texttt{SC.UnSignCrypt}(\cdot, sk^*_R, \cdot)$.

The proof essentially shows that, since the challenge messages produced by a high-entropy attacker $\mathcal{A}_1$ have min-entropy $\mu(k)$, the probability that $\mathcal{A}_2$ queries the signcryption oracle on one of those messages is bounded by $4 \cdot q_s(k) \cdot \ell(k) \cdot \pi(k)$. If this does not occur, then a low-entropy attacker can easily run a high-entropy attacker as a black-box subroutine. The proof holds for deterministic schemes only. We are not aware if the same is true for probabilistic schemes.

We also have that the low-entropy confidentiality definition is strictly stronger than the high-entropy confidentiality definition. If SC is a high-entropy confidential signcryption scheme, then the signcryption scheme SC' given in Figure 9 is high-entropy confidential signcryption scheme but not a low-entropy confidential signcryption scheme.

```
$\texttt{SC.SignCrypt}'(sk_S, pk_R, m)$:
  $C \leftarrow \texttt{SC.SignCrypt}(sk_S, pk_R, m)$
  If $m = 0^k$
      Return $C \| 0$
  Else
      Return $C \| 1$
```

```
$\texttt{SC.UnSignCrypt}'(pk_S, sk_R, C)$:
  Parse $C$ as $C' \| c$ for $c \in \{0, 1\}$
  $m \leftarrow \texttt{SC.UnSignCrypt}(pk_S, sk_R, C')$
  If $c = 0$ and $m \neq 0^k$
      Return $\perp$
  If $c = 1$ and $m = 0^k$
      Return $\perp$
  Else
      Return $m$
```

Fig. 9. A signcryption scheme which is high-entropy secure but not low-entropy secure

4.2 The Encrypt-and-Sign Signcryption Scheme

Initially, it may be thought that high-entropy confidentiality may be easily achieved through the combination of deterministic encryption and confidential signatures. However, many of the classic composition theorems, such as encrypt-then-sign, fail to achieve high-entropy security even when instantiated with secure components.

```
SC.Setup($1^k$)
    $\lambda_{ss} \leftarrow$ SS.Setup($1^k$)
    $\lambda_{pke} \leftarrow$ PKE.Setup($1^k$)
    $\lambda_{sc} \leftarrow (\lambda_{ss}, \lambda_{pke})$
    Return $(\lambda_{sc})$

SC.Kg_r($\lambda_{sc}$)
    Parse $\lambda_{sc}$ as $(\lambda_{ss}, \lambda_{pke})$
    $(pk_R, sk_R) \leftarrow$ PKE.Kg($\lambda_{pke}$)
    Return $(pk_R, sk_R)$

SC.Kg_s($\lambda_{sc}$)
    Parse $\lambda_{sc}$ as $(\lambda_{ss}, \lambda_{pke})$
    $(pk_S, sk_S) \leftarrow$ SS.Kg($\lambda_{ss}$)
    Return $(pk_S, sk_S)$

SC.SignCrypt($\lambda_{sc}, pk_R, sk_S, m$)
    Parse $\lambda_{sc}$ as $(\lambda_{ss}, \lambda_{pke})$
    $c \leftarrow$ PKE.Enc($\lambda_{pke}, pk_R, (pk_S||m)$)
    $\sigma \leftarrow$ SS.Sign($\lambda_{ss}, sk_S, (pk_R||m)$)
    Return $C = (c, \sigma)$

SC.UnSignCrypt($\lambda_{sc}, sk_R, pk_S, C$)
    Parse $\lambda_{sc}$ as $(\lambda_{ss}, \lambda_{pke})$
    Parse $C$ as $(c, \sigma)$
    $(pk'_S||m') \leftarrow$ PKE.Dec($\lambda_{pke}, sk_R, c$)
    If $pk'_S \neq pk_S$, reject
    Extract $pk_R$ from $sk_R$
    If SS.Ver($\lambda_{ss}, pk_S, (pk_R||m'), \sigma$) $= \perp$, reject
    Return $m'$
```

Fig. 10. The Encrypt-and-Sign signcryption scheme

However, we can show that the encrypt-and-sign (which is typically insecure as a signcryption scheme) is secure when instantiated with an IND-CCA2 public-key encryption scheme and a strongly confidential signature scheme[1]. The construction is given in Figure 10. The scheme can easily be shown to be unforgeable (in the sense that an attacker cannot obtain a signcryption of any message which was not previously sent by that sender to that receiver).

Theorem 1. *If the signature scheme is deterministic, strongly unforgeable, and strongly confidential, and the encryption scheme is* IND-CCA2 *secure, then the signcryption scheme is confidential in the high-entropy model. In particular, if there exists an attacker $\mathcal{A}$ against the high-entropy security of the signcryption scheme (asking $\ell(k)$ challenge messages and making at most $q_{sc}(k)$ signcryption queries), then there exist attackers $\mathcal{A}_{pke}$, $\mathcal{A}_{ss}$, and $\mathcal{A}_{sunf}$ against the* IND-CCA2 *security of the encryption scheme, against the strong confidentiality of the signature scheme, and against the strong unforgeability of the signature scheme, such that*

$$\mathsf{Adv}^{\mathrm{hSCR}}_{\mathsf{E+S},\mathcal{A}}(k) \leq \ell(k) \cdot \mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}}(k) + \mathsf{Adv}^{\mathrm{sSig}}_{\mathsf{SS},\mathcal{A}_{ss}}(k) + \mathsf{Adv}^{\mathrm{seuf-cma}}_{\mathsf{SS},\mathcal{A}_{sunf}}(k) \ .$$

where the running times of $\mathcal{A}_{pke}$, $\mathcal{A}_{ss}$, and $\mathcal{A}_{sunf}$ equal the one of $\mathcal{A}$ plus $(q_{sc}(k)+\ell(k)) \cdot (Time_{\texttt{SC.SignCrypt}}(k) + Time_{\texttt{SC.UnSignCrypt}}(k)) + O(k)$.

The security of this scheme can be proven in a manner similar to the encryption/signature composition theorems proven by An *et al.* [1].

[1] Strongly confidential, probabilistic signature schemes are given in Sections 3.3 and 3.4. These can be transformed in a strongly confidential, deterministic signature schemes using the derandomization techniques discussed in the next section.

4.3 Derandomization

Goldreich [17] presents a technique to turn any probabilistic signature scheme into a deterministic one. The idea is to include the secret key κ of a pseudorandom function $(\mathtt{PRF.Kg}, \mathtt{PRF})$ in the secret signing key and, when signing a message m, use the random coins $r = \mathtt{PRF}(\kappa; m)$ in this process. Note that the resulting scheme now yields the same signature if run twice on the same message. A formal definition of a PRF can be found in Appendix A.

We show that Goldreich's idea applies to signcryption schemes as well, taking advantage of the fact that a signcryption scheme involves a secret signing key in which we can put the key κ of the pseudorandom function. Nonetheless, whereas a probabilistic signcryption scheme usually hides the fact that the same message has been encrypted twice, a derandomized version clearly leaks this information.

For a signcryption scheme SC the derandomized version $\mathsf{SC}^{\mathsf{PRF}}$ based on a pseudorandom function PRF works according to Goldreich's strategy:

```
$\mathtt{SC.Setup}^{\mathsf{PRF}}(1^k)$:
  Return $\lambda_{sc} \leftarrow \mathtt{SC.Setup}(1^k)$

$\mathtt{SC.Kg_s}^{\mathsf{PRF}}(\lambda_{sc})$:
  $(sk_S, pk_S) \leftarrow \mathtt{SC.Kg_s}(\lambda_{sc})$
  $\kappa \leftarrow \mathtt{PRF.Kg}(1^k)$
  $sk_S^{\mathsf{PRF}} \leftarrow (sk_S, \kappa)$; $pk_S^{\mathsf{PRF}} \leftarrow pk_S$
  Return $(sk_S^{\mathsf{PRF}}, pk_S^{\mathsf{PRF}})$

$\mathtt{SC.Kg_r}^{\mathsf{PRF}}(\lambda_{sc})$:
  Return $(sk_R, pk_R) \leftarrow \mathtt{SC.Kg_r}$

$\mathtt{SC.SignCrypt}^{\mathsf{PRF}}(sk_S^{\mathsf{PRF}}, pk_R, m)$:
  Parse $sk_S^{\mathsf{PRF}}$ as $(sk_S, \kappa)$
  $r \leftarrow \mathtt{PRF}(\kappa, (pk_R, m))$
  $C \leftarrow \mathtt{SC.SignCrypt}(sk_S, pk_R, m; r)$
      (i.e. using randomness $r$)
  Return $C$

$\mathtt{SC.UnSignCrypt}^{\mathsf{PRF}}(sk_R, pk_S^{\mathsf{PRF}}, C)$:
  Return $\mathtt{SC.UnSignCrypt}(sk_R, pk_S, C)$
```

Proposition 7 (Derandomized Signcryption). *Let* SC *be an unforgeable and high-entropy (resp. low-entropy) confidential signcryption scheme. Then the scheme* $\mathsf{SC}^{\mathsf{PRF}}$ *is a deterministic, unforgeable signcryption scheme which is high-entropy (resp. low-entropy) confidential. That is, for* $x \in \{l, h\}$ *and any adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against xSCR confidentiality, there exist adversaries* $\mathcal{D}$ *and* $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ *such that*

$$\mathsf{Adv}^{\mathrm{xSCR}}_{\mathsf{SC}^{\mathsf{PRF}},\mathcal{A}}(k) \leq 2 \cdot Adv^{\mathsf{PRF}}_{\mathcal{D}}(k) + \mathsf{Adv}^{\mathrm{xSCR}}_{\mathsf{SC},\mathcal{B}}(k) + 2q_{sc}(k) \cdot \ell(k) \cdot \pi(k)$$

where $\mathcal{D}$*'s running time is identical to the time of* $\mathcal{A}$*, plus* $Time_{\mathtt{SC.Setup}}(k) + Time_{\mathtt{SC.Kg_s}}(k) + Time_{\mathtt{SC.Kg_r}}(k) + (q_{sc} + \ell(k)) \cdot Time_{\mathtt{SC.SignCrypt}}(k) + O(k)$*; the running time of* $\mathcal{B}$ *equals the time of* $\mathcal{A}$ *plus* $O(q_{sc} \cdot \log q_{sc})$*.*

Acknowledgements. The authors wish to thank the ECRYPT II MAYA working group on the design and analysis of primitives and protocols for interesting preliminary discussions on this topic. The work described in this report has in part been supported by the Commission of the European Communities through the ICT program under contract ICT-2007-216676 ECRYPT II. The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. Dominique and Marc were supported by the Emmy Noether grant Fi 940/2-1 of the German Research Foundation (DFG), and by CASED (`www.cased.de`).

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. Baek, J., Steinfeld, R., Zheng, Y.: Formal proofs for the security of signcryption. Journal of Cryptology 20(2), 203–235 (2007)
3. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and efficiently searchable encryption. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 535–552. Springer, Heidelberg (2007)
4. Bellare, M., Fischlin, M., O'Neill, A., Ristenpart, T.: Deterministic encryption: Definitional equivalences and constructions without random oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 360–378. Springer, Heidelberg (2008)
5. Bellare, M., Rogaway, P.: The exact security of digital signatures — how to sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
6. Boldyreva, A., Fehr, S., O'Neill, A.: On notions of security for deterministic encryption, and efficient constructions without random oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 335–359. Springer, Heidelberg (2008)
7. Canetti, R.: Towards realizing random oracles: Hash functions that hide all partial information. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 455–469. Springer, Heidelberg (1997)
8. Canetti, R., Micciancio, D., Reingold, O.: Perfectly one-way probabilistic hash functions. In: Proc. 30th Symposium on the Theory of Computing – STOC 1998, pp. 131–140. ACM, New York (1998)
9. Coron, J.-S.: On the exact security of full domain hash. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 229–235. Springer, Heidelberg (2000)
10. Dent, A.W., Fischlin, M., Manulis, M., Stam, M., Schröder, D.: Confidential signatures and deterministic signcryption (2009), `http://eprint.iacr.org/2009/588`
11. Dodis, Y., Gennaro, R., Håstad, J., Krawczyk, H., Rabin, T.: Randomness extraction and key derivation using the CBC, Cascade and HMAC modes. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 494–510. Springer, Heidelberg (2004)
12. Dodis, Y., Smith, A.: Entropic security and the encryption of high entropy messages. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 556–577. Springer, Heidelberg (2005)
13. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography. SIAM Journal on Computing 30(2), 391–437 (2000)
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. Fischlin, M.: Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 429–444. Springer, Heidelberg (1999)

16. Fischlin, M.: Anonymous signatures made easy. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 31–42. Springer, Heidelberg (2007)
17. Goldreich, O.: Two remarks concerning the Goldwasser-Micali-Rivest signature scheme. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 104–110. Springer, Heidelberg (1987)
18. Howgrave-Graham, N.A., Smart, N.P.: Lattice attacks on digital signature schemes. Designs, Codes and Cryptography 23(3), 283–290 (2001)
19. Nisan, N., Ta-Shma, A.: Extracting randomness: A survey and new constructions. Journal of Computer and System Science 58(1), 148–173 (1999)
20. Nisan, N., Zuckerman, D.: Randomness is linear in space. Journal of Computer and System Science 52(1), 43–52 (1996)
21. Rackoff, C., Simon, D.: 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)
22. Russell, A., Wang, H.: How to fool an unbounded adversary with a short key. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 133–148. Springer, Heidelberg (2002)
23. Yang, G., Wong, D.S., Deng, X., Wang, H.: Anonymous signature schemes. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 347–363. Springer, Heidelberg (2006)
24. Zheng, Y.: Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)

A Standard Security Notions

A.1 Signature Schemes

The standard notion for signature security is that of (strong) existential unforgeability under chosen message attacks ($\mathsf{sEUF\text{-}CMA}$). The strong version is defined below. Freshness of (m, σ) indicates that σ was never received by $\mathcal{A}$ as response to a signing request on m.

$$\mathsf{Adv}^{\mathrm{seuf-cma}}_{\mathsf{SS},\mathcal{A}}(k) = \Pr\left[\begin{array}{l} \mathtt{SS.Ver}(\lambda_{ss}, pk, m, \sigma) = \top \\ (m,\sigma) \text{ is fresh} \end{array} : \begin{array}{l} \lambda_{ss} \xleftarrow{R} \mathtt{SS.Setup}(1^k) \\ (pk, sk) \xleftarrow{R} \mathtt{SS.Kg}(\lambda_{ss}) \\ (m,\sigma) \xleftarrow{R} \mathcal{A}^{\mathtt{SS.Sign}(\lambda_{ss}, sk, \cdot)}(\lambda_{pke}, pk) \end{array}\right].$$

The advantage $\mathsf{Adv}^{\mathrm{euf-cma}}_{\mathsf{SS},\mathcal{A}}(k)$ of the slightly weaker notion ($\mathsf{EUF\text{-}CMA}$) is defined analogously, but this time m only needs to be fresh.

A.2 Public-Key Encryption

A *public key encryption* scheme is a tuple of algorithms $\mathsf{PKE} = (\mathtt{PKE.Setup}, \mathtt{PKE.Kg}, \mathtt{PKE.Enc}, \mathtt{PKE.Dec})$. First the common parameters for the given security level $k \in \mathbb{N}$ are generated by $\lambda_{pke} \xleftarrow{R} \mathtt{PKE.Setup}(1^k)$ after which a user's public/private keys are generated using $(pk, sk) \xleftarrow{R} \mathtt{PKE.Kg}(\lambda_{pke})$. Given such a key pair, a message $m \in \{0,1\}^*$ is encrypted by $c \xleftarrow{R} \mathtt{PKE.Enc}(\lambda_{pke}, pk, m)$; a ciphertext is decrypted by $m \xleftarrow{R} \mathtt{PKE.Dec}(\lambda_{pke}, sk, c)$. For consistency, we require that for all messages $m \in \{0,1\}^*$, we have that $\mathtt{PKE.Dec}(sk, \mathtt{PKE.Enc}(pk, m)) = m$.

We require a PKE is secure against $\mathsf{IND\text{-}CCA2}$ attacks [21,13], for which the advantage of an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is defined as

$$\mathsf{Adv}^{\text{cca2}}_{\mathsf{PKE},\mathcal{A}}(k) = \left|\Pr\left[\mathit{Expt}^{cca2-0}_{\mathcal{A}} = 1\right] - \Pr\left[\mathit{Expt}^{cca-1}_{\mathcal{A}} = 1\right]\right| \ ,$$

where (for $b \in \{0,1\}$):

```
$Expt^{cca2-b}_{\mathcal{A}}$
  $\lambda_{pke} \xleftarrow{R} \texttt{PKE.Setup}(1^k)$
  $(pk, sk) \xleftarrow{R} \texttt{PKE.Kg}(\lambda_{pke})$
  $(m_0, m_1, \omega) \xleftarrow{R} \mathcal{A}_1^{\texttt{PKE.Dec}(\lambda_{pke}, sk, \cdot)}(\lambda_{pke}, pk)$
  $c^* \xleftarrow{R} \texttt{PKE.Enc}(\lambda_{pke}, pk, m_b)$
  $b' \xleftarrow{R} \mathcal{A}_2^{\texttt{PKE.Dec}(\lambda_{pke}, sk, \cdot)}(c^*, \omega)$
  Output 1 if $b' = b$
```

The adversary $\mathcal{A}_2$ is may not query $\texttt{PKE.Dec}(sk, \cdot)$ with c^*. A PKE scheme PKE is $\mathsf{IND\text{-}CCA2}$ secure if the advantage function $\mathsf{Adv}^{\text{cca2}}_{\mathsf{PKE},\mathcal{A}}(k)$ is a negligible function for all probabilistic polynomial-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

A.3 Pseudo-Random Functions

A pseudo-random function is a pair of algorithms $\mathsf{PRF} = (\texttt{PRF.Kg}, \texttt{PRF})$. The key generation algorithm outputs a key $\kappa \xleftarrow{R} \texttt{PRF.Kg}(1^k)$. For our purposes, a pseudo-random function $\texttt{PRF}(\kappa, \cdot)$ takes arbitrary bitstrings as inputs and outputs a bitstring in a given space $\mathcal{R}$. Let $\mathcal{F}$ be the set of all functions from $f : \{0,1\}^* \to \mathcal{R}$. The security of a PRF against a PPT attacker $\mathcal{A}$ is defined by the following two games:

```
$Expt^{PRF-0}_{\mathcal{A}}(k)$:                    $Expt^{PRF-1}_{\mathcal{A}}(k)$:
  $\kappa \xleftarrow{R} \texttt{PRF.Kg}(1^k)$                      $f \xleftarrow{R} \mathcal{F}$
  Return $\mathcal{A}^{\texttt{PRF}(\kappa,\cdot)}(1^k)$                  Return $\mathcal{A}^{f(\cdot)}(1^k)$
```

The attacker's advantage is defined to be:

$$Adv^{\texttt{PRF}}_{\mathsf{PRF},\mathcal{A}}(k) = |\Pr[\mathit{Expt}^{PRF-0}_{\mathcal{A}}(k) = 1] - \Pr[\mathit{Expt}^{PRF-1}_{\mathcal{A}}(k) = 1]|.$$

Identity-Based Aggregate and Multi-Signature Schemes Based on RSA

Ali Bagherzandi and Stanisław Jarecki

Department of Computer Science, University of California, Irvine
{zandi,stasio}@ics.uci.edu

Abstract. We propose new identity-based multi-signature (IBMS) and aggregate signature (IBAS) schemes, secure under RSA assumption. Our schemes reduce round complexity of previous RSA-based IBMS scheme of Bellare and Neven [BN07] from three to two rounds. Surprisingly, this improvement comes at virtually no cost, as the computational efficiency and exact security of the new scheme are almost identical to those of [BN07]. The new scheme is enabled by a technical tool of independent interest, a class of zero-knowledge proofs of knowledge of preimages of one-way functions which is straight-line simulatable, enabling concurrency and good exact security, and *aggregatable*, enabling aggregation of parallel instances of such proofs into short multi/aggregate signatures.

1 Introduction

A multisignature protocol allows a group of players to sign the same message by generating a short string, called a multisignature, which can be verified against the set of the public keys of these players. Aggregate signature is a generalization of this notion to the case where each player signs a potentially different message. Such schemes reduce the bandwidth needed to transmit signatures, the space needed to store them, and the time needed to verify them, from linear in the number of the cosigners to a constant. Reducing bandwidth is especially important for low-energy devices, such as RFID chips and sensors, which communicate over energy-consuming wireless channels where data transmision consumes several orders of magnitude more energy than arithmetic operations (see *e.g.* [BA03]). Standard multi-/aggregate signatures reduce the space taken by n signatures from $O(n)$ to $O(1)$, but the verifiers still need the public keys of n signers. Therefore in applications where bandwidth is a bottleneck it can be useful to consider *identity-based* multi-/aggregate signatures where verifiers only need unique identifiers of signers, e.g. 32-bit IP addresses, instead of public keys.

Identity-Based (Multi-/Aggregate) Signatures. Identity-based cryptography [Sha84] simplifies public key management by replacing users' public keys with their identity *e.g.* their names, e-mails or IP addresses. In identity-based scheme a trusted party, a Private Key Generator (PKG), generates a private key corresponding to each user's identity, and messages signed using such keys can be then verified using the signer's identity and the PKG's master public key.

In the case of identity-based multi-/aggregate signatures, if all signers have their private keys issued by the same PKG then the verifier needs only the PKG's master public key and the identities of all signers. Note that in many applications the identities of signers are often present in the protocol messages, *e.g.* the usernames or IP addresses in packet headers, in which case an identity-based multi-/aggregate signature adds only a constant bandwidth overhead over unauthenticated messages.

Current State of the Art. Standard signatures imply identity-based signatures following the "certification paradigm", *e.g.* [GHK06], *i.e.* by simply attaching signer's public key and certificate to each signature. However, it is not clear how to apply this idea to convert standard multi-/aggregate signatures, *e.g.* [BN06, BCJ08], into identity-based ones, because it is not clear how to aggregate n separate public keys and certificates, even if all certificates are signed by the same CA. (Standard aggregate signatures can be used to eliminate the overhead of CA's signatures on the certificates, but this would not eliminate the overhead due to the public keys.)

The first efficient IBAS/IBMS schemes designed from scratch are due to Gentry and Ramzan [GR06]. Their schemes employ a group with a bilinear map, their security relies, in the Random Oracle Model (ROM), on the hardness of GapDH problem, the schemes are non-interactive, and both the signing and verification times take $O(1)$ exponentiations and bilinear map operations. However, the IBAS scheme of [GR06] requires all cosigners to share a common token for every set of signatures they want to aggregate, and each cosigner must ensure that this token has not been previously used in signing a different message, hence in some applications this scheme will need an extra communication round for the participants to agree on a fresh common token. In subsequent work, Boldyreva *et. al* [BGOY10] (correcting a previous version of this paper) proposed an IBAS scheme which does not need these unique tokens but it requires sequential communication pattern, and it is based on a more complex bilinear map assumption. Note that while sequential communication is perfectly suited to some applications, *e.g.* secure route discovery [KT05], it introduces unnecessary overhead for players connected *e.g.* by a broadcast channel or a tree topology.

Without bilinear maps, Bellare and Neven [BN07] gave an IBMS scheme which relies on the RSA assumption in ROM. Their scheme also has fast multi-signature generation and verification, requiring $O(1)$ exponentiations, but it takes three rounds of interaction. Note that any 3-round IBMS implies a 4-round IBAS if all cosigners' messages are broadcast and the IBMS scheme is run on their concatenation. (Moreover, in the IBMS scheme of [BN07] this broadcast can be piggybacked on the first protocol round, giving a 3-round IBAS scheme.) However, such broadcast of all messages to all co-signers imposes bandwidth usage which might not be otherwise required, and so apart from this generic transformation it is interesting to consider IBAS schemes which do not require such broadcast. (As a side remark, we believe that the 3-round IBMS scheme of [BN07] can be modified to a 3-round IBAS scheme without such broadcast, *e.g.* using ideas similar to our IBAS scheme [BJ10].)

IBAS/IBMS Schemes	Underlying Problem[(1)]	Restr-ictions[(2)]	Number of Rounds	Verification Time[(3)]	Signing Time[(3)]	Signature Length[(4)]
[GR06]-IBAS	GapDH	Stateful	1	3P+nM	5M	$2\|G_1\| + \kappa$
[BGOY10]	GapDH	Sequential	1	6P+nM	7M	$3\|G\|$
OUR IBAS	RSA	-	2	nE	2E	$\|\mathbb{Z}_n^*\| + 2\kappa + \log l$
[GR06]-IBMS	GapDH	-	1	3P	3M	$2\|G_1\|$
[BN07]	RSA	-	3	1E	2E	$\|\mathbb{Z}_n^*\| + \kappa$
OUR IBMS	RSA	-	2	1E	2E	$\|\mathbb{Z}_n^*\| + 2\kappa + \log l$

Fig. 1. (1) All schemes have been given security proofs only in the *ROM* model; (2) The IBAS scheme of [GR06] assumes that the players share a unique and common token for every instance of the IBAS scheme. This requirement can be avoided at the cost of an additional round of interaction, while the scheme of [BGOY10] requires sequential aggregation; (3) Signing time is measured per player. In both signing and verification costs, P is the cost of one pairing operation, M is the cost of scalar multiplication on an elliptic curve, and E is the cost of (multi-)exponentiation in $\mathbb{Z}_n^*$ (with about 80-bit exponents); (4) Signature length is measured in bits where κ is the security parameter, n is an RSA modulus, l is an upper bound on the number of players, G_1 and G_2 are two groups of elliptic curve points with an asymmetric bilinear map, G is a group of elliptic curve points with a symmetric bilinear map, and $|A|$ stands for the bitsize of representation of elements in group A. Typical values for these parameters are $\kappa = 160$, $|G_1| = 160$, $|G| = 512$, $\log l = 20$, and $|\mathbb{Z}_n^*| = 1024$ or 2048.

Our Contributions. We propose IBMS and IBAS schemes secure under RSA assumption in ROM which require only two rounds of communication. This provides alternatives to IBMS/IBAS schemes based on bilinear maps especially in applications which intrinsically take two communication rounds, such as authenticated route discovery or aggregation of broadcast acknowledgements. Since bilinear map operations are still more expensive than RSA exponentiation, our computational costs are slightly lower in signing and significantly lower in verification, compared to *e.g.* [GR06], although our signatures are longer. A summary of these comparisons is in Figure 1.

Further Related Work. Gregory Neven introduced two primitives, *sequential aggregate signed data* and *multi-signed data*, corresponding to aggregate signatures and multisignatures respectively, whose goal is to minimize the total bandwidth consumed by signatures *and messages* incurred in transmission of authenticated data originated by multiple sources [Nev08]. His constructions use message recovery techniques to squeeze message bits into a (multi/aggregate) signature. Comparing his work to ours, we note that (1) his schemes support only sequential aggregation when signing different messages; (2) bandwidth savings depend on message sizes (for small messages the bandwidth can be worse than with standard signatures); (3) these schemes do not address the overhead due to public keys, which raises an interesting question whether total bandwidth due to signatures and messages can be further reduced, perhaps using message-recovery techniques, with *identity-based* multi/aggregate signatures. In other related work Herranz and Galindo *et. al* [Her06, GHK06] show identity-based signatures which can be aggregated if they originate from *the same signer*.

Organization/Roadmap. Section 2 contains a technical overview of our constructions. In Section 3 we define IBMS schemes. (We relegate a formal description of IBAS schemes to [BJ10].) In Section 4 we develop our tools, namely we introduce structured-instance zero-knowledge (ZK) proofs and Σ-equivocable commitments and we show that Σ-equivocable commitments suffice to compile a class of Σ-protocols which includes an RSA-based identification protocol, a proof of knowledge of e-th root, to straight-line simulatable structured-instance ZK. In Section 5 we show *homomorphic* Σ-equivocable commitments secure under RSA. By the results from Section 4 this implies an *aggragatable* structured-instance ZK proof of knowledge of e-th root, which leads to an IBMS scheme construction, described in Section 6, and an IBAS scheme sketched in Section 7.

2 Technical Overview

Our IBAS/IBMS scheme is a multi-prover version of Guillou-Quisquater signature [GQ88]. The ID-based version of GQ signature is a non-interactive zero-knowledge (NIZK) proof of knowledge (PK) of e-th root modulo n (in ROM). Let $y = H(ID)$ be an element in $\mathbb{Z}_n^*$ and let x be the e-th root of y, a private key corresponding to identity ID. (Such private key can be computed the PKG who knows the factorization of n.) To sign message m, the signer with identity ID follows the ROM-based NIZK PK of e-th root of y: It computes the first proof message $a = k^e$ for random k in $\mathbb{Z}_n^*$, gets challenge c by querying (m, a) to a hash function (modeled as random oracle), and computes response z to this challenge as $z = kx^c$. The signature is (a, z) verified by checking if $z^e = ay^c$ for $c = H(m, a)$. Due to homomorphic property of exponentiation one might hope to obtain an IBAS/IBMS scheme by aggregating such ROM-based NIZK PK's of e-th root made by several cosigners. For instance, consider the two-round protocol built along the lines of the DL-based multisignature scheme of [MOR01]: In the first round each player broadcasts its first message a_i. All players obtain a common challenge c by querying the hash function on input including $a = \prod a_i$ and the message being signed. Finally each player broadcasts its response z_i to this challenge. The multi-signature is (a, z) where $z = \prod z_i$. Note that if $z_i^e = a_i y_i^c$ for each i then (a, z) satisfies the verification equation $z^e = a(\prod y_i)^c$ where $y_i = H(ID_i)$. We believe that an adaptation of the security proof of [MOR01] would show security of this scheme, but the resulting security argument would have several limitations: (1) The reduction would be only from expected-time hardness of RSA problem; (2) It would encounter substantial security degradation due to extensive use of rewindings; (3) It would therefore not extend to concurrent executions of multiple instances of this scheme.

To explain how we overcome these limitations we need to first explain why they appear in the above draft scheme. The simulator for the NIZK PK of e-th root picks a random challenge c and a random z in $\mathbb{Z}_n^*$, computes prover's first message as $a = z^e y^{-c}$ and defines the hash of (a, m) as c because it controls the hash function H. Note that since the adversary has no information about a, there is only a negligible chance that it queries H on the same (a, m) before the simulator attempts to define its value as c, and hence the simulator passes with overwhelming probability. The fundamental difference between this simulation and the simulation for aggregated proof in the draft scheme above is that in the aggregated proof corrupt cosigners can choose their contributions a_i on the basis of a_i's broadcasted by the honest cosigners. Consequently, the simulator can only guess the resulting a value with probability $1/q_h$ where q_h is the number of hash queries the adversary makes. This gives rise to a simulation procedure which rewinds the adversary expected q_h times in each signature instance, which causes all the limitations listed above: reduction to expected-time hardness, loose security reduction, and no argument for security of concurrent protocol instances.

Bellare and Neven [BN07] showed how to overcome all these issues in the ROM model by adding an extra communication round in which each player first commits to its a_i contribution by broadcasting a hash $H(a_i)$. By controlling the hash function H the simulator can learn the a_i's committed by the adversary and then decide on the a_i's published on behalf of the honest players. This way the simulator passes without rewinding with overwhelming probability, similarly to the NIZK simulation sketched above. The main technical challenge we handle in this work is how to achieve such straight-line simulation without introducing such extra communication round, i.e. with only two rounds of interaction. Our technique is a variant of Damgard's HVZK-to-ZK compilation [Dam00] which constructs a straight-line simulatable zero-knowledge proof from any Σ-protocol using an equivocable commitment scheme, but we introduce an interesting twist: In Damgard's scheme a signer commits to its a_i value using an equivocable commitment scheme, and the simulator, on any challenge c can open this commitments to the value $a_i = z_i^e y_i^{-c}$ needed for the proof to verify (where response z_i is chosen at random, to match the response distribution in the real proof). However, to create an IBMS/IBAS scheme by aggregating such proofs we need this commitment scheme to be multiplicatively homomorphic, and to the best of our knowledge no efficient commitment scheme is both equivocable and multiplicatively homomorphic. Instead, we show a commitment scheme which is multiplicatively homomorphic over $\mathbb{Z}_n^*$ and satisfies a restricted form of equivocability which we call *Σ-equivocability*, and which suffices for straight-line simulation of Σ-protocol compiled as above. For example, Σ-equivocable commitment for relation $R = \{(x, y) \,|\, y = x^e\}$ allows for equivocation of commitments to messages of the form $z^e y^{-c}$ for any c and z, and this is exactly the form of message a which the simulator needs in the above proof.

The idea to use commitments with similarly restricted equivocability appeared before in [BCJ08], where it was used to construct a straight-line simulatable and aggregatable proof of DL knowledge, and a DL-based multi-signature scheme. However, the equivocability notion (and the construction) of [BCJ08] gives rise to only single-instance zero-knowledge proofs. Intuitively, this suffices for security of multi-signatures (as opposed to identity-based multi-signatures) because in multi-signatures the adversary w.l.o.g. corrupts all players except of one, so the simulator needs to embed its challenge problem in just one public key, and needs to simulate multi-signature protocol on behalf of only that one player. Using this form of equivocation in security argument for identity-based schemes would introduce security degradation by factor of q_H, the number of hash function queries, because the simulator would have to guess the single identity into which to embed its challenge. Here we define a more general notion of Σ-equivocability which allows for straight-line simulatable "structured instance" zero-knowledge proofs in the CRS model: In structured-instance zero-knowledge proofs, formalized in this paper, the simulator can simulate on any statement in a class of *related* instances, in contrast to a single statement in single-instance ZK and any instance in (standard) multi-instance ZK. The class of instances which is particularly useful in showing a security reduction for an IBMS/IBAS scheme based on Σ-protocol for proving knowledge of preimage of function $f(x) = x^e$ are instances of the form $y = \mathring{y} f(\delta)$ where $\mathring{y}$ is the simulator's challenge. In this way the simulator can embed its challenge into any number of identities, picking random δ for each identity, and yet straight-line simulate the proofs performed on behalf of all these entities in parallel. Thus our main technical contribution is two-fold: First, we formalize the notion of Σ-equivocability and apply it to a compilation from Σ-protocols to straight-line simulatable structured-instance ZKPK (Section 4). Secondly, we construct a multiplicatively homomorphic and Σ-equivocable commitment scheme based on the RSA problem (Section 5). Together, these two parts immediately imply the IBMS and IBAS schemes we present in this paper (Section 6 and Section 7).

3 Identity-Based Multi-/Aggregate Signature Schemes

We define the notion of identity-based multisignature scheme (IBMS) building on the definitions given by [MOR01, BN06, GR06, BN07]. (Due to lack of space, we relegate the extension of our definitions to IBAS schemes to the full version of the paper [BJ10]). Our notion is more flexible than that of [BN07, MOR01, BN06] because we do not require the set of participants' identities as input to the multi-/aggregate signature protocol. The participating players must be aware of each other in the protocol execution, but this is needed only to ensure proper communication, and the participant identities are not required as inputs to the cryptographic protocol. The schemes secure in this setting provide flexibility to applications of multi-/aggregate signatures because sometimes signers might care only about the message they are signing and not about the identities of the cosigners. Otherwise the list of cosigners can always be attached to the message being signed.

Syntax of an IBMS Scheme. We define an identity-based multisignature as $\mathsf{IBMS} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{MSign}, \mathsf{Vrfy})$ where Setup, KeyDer and Vrfy are probabilistic poly-time algorithms, and MSign is a distributed protocol executed by a set of parties *s.t.*

– $(mpk, msk) \leftarrow \mathsf{Setup}(1^\kappa)$, run by a trusted party, on input the security parameter κ, generates master public key mpk and corresponding master secret key msk.

- $sk_{Id} \leftarrow \mathsf{KeyDer}(msk, Id)$, run by a trusted party, on input master secret key msk and an identity $Id \in \{0,1\}^*$ provides a secret key sk_{Id} to the user with identity Id.
- MSign is a multisignature protocol run by a group of players who intend to sign the same message m. Player with identity Id executes this protocol on public inputs mpk and message m and private input sk_{Id} which is his own secret key. The local output of the protocol for every participant is a multisignature denoted σ.
- $\{0,1\} \leftarrow \mathsf{Vrfy}(mpk, m, IdSet, \sigma)$ verifies whether σ is a valid multisignature on message m on behalf of the set of the identities $IdSet$.

In the random oracle model (ROM), KeyDer, MSign and Vrfy procedures additionally have access to a random oracle $\mathsf{H}(\cdot) : \{0,1\} \rightarrow D$, where D depends on the scheme. This set of procedures must satisfy the following *completeness* properties: For any integer n, any message m, and any (mpk, msk) output by $\mathsf{Setup}(1^\kappa)$, if for $i = 1..n$, one obtains $sk_{Id_i} \leftarrow \mathsf{KeyDer}(msk, Id_i)$ and correctly follows MSign on input m using secret keys sk_{Id_i}, then assuming all messages are delivered between players, each player outputs the same string σ which satisfies $\mathsf{Vrfy}(mpk, m, \{Id_1, ..., Id_n\}, \sigma) = 1$.

Security Notion of an IBMS Scheme. We model the security as existential unforgeability under an adaptive chosen message *and* adaptive chosen identity attack: The adversary participates in a game in which it issues a number of key derivation and signature queries. In a key derivation query, the adversary corrupts a player by submitting its identity Id to the key derivation oracle and receiving its secret key sk_{Id}. In a signature query the adversary specifies the message m and the identity Id that it wants to interact with; and the signing oracle performs MSign protocol on message m on behalf of Id. The adversary wins the game if it eventually outputs a message m, a multisignature σ and a set of identities $IdSet$ *s.t.* $\mathsf{Vrfy}(mpk, m, IdSet, \sigma) = 1$ and there exists an identity Id *s.t.*, the adversary never queried the key derivation oracle on Id and never queried the signing oracle on (m, Id). More formally we define the *adversarial advantage* of $\mathcal{A}$ against $\mathsf{IBMS} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{MSign}, \mathsf{Vrfy})$ as a probability that experiment $\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A})$ described in Figure 2 outputs 1 *i.e.*

$$\mathbf{Adv}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A}) = Pr[\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A}) = 1]$$

where the probability goes over the random coins of the adversary and all the randomness used in the experiment. We call an IBMS scheme $(t, \epsilon, n, q_K, q_S)$-secure if $\mathbf{Adv}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A}) \leq \epsilon$ for every adversary $\mathcal{A}$ that runs in time at most t, makes at most q_K key derivation queries and at most q_S signature queries, and produces a forgery on behalf of at most n parties. In the random oracle model we extend this notion to $(t, \epsilon, n, q_K, q_S, q_H)$-security, where $\mathcal{A}$ is additionally restricted to at most q_H hash queries and the probability in the experiment $\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A})$ goes also over random choice of a hash function.

Experiment $\mathbf{Exp}^{\mathsf{uu-cma}}_{\mathsf{IBMS}}(\mathcal{A})$

- $(mpk, msk) \leftarrow \mathsf{Setup}(1^{\kappa})$; $\mathsf{MIdLst} \leftarrow \emptyset$; $\mathsf{CIdLst} \leftarrow \emptyset$;
- Run $\mathcal{A}(mpk)$, and handle $\mathcal{A}$'s key derivation and signature queries as follows:
- On a key derivation query on identity Id, add Id to CIdLst, run KeyDer on input (msk, Id) and return sk_{Id} to $\mathcal{A}$.
- On a signing query on pair (m, Id), add (m, Id) to MIdLst, run MSign protocol on behalf of identity Id on message m forwarding messages to and from $\mathcal{A}$.
- When $\mathcal{A}$ halts, parse its output as $(m, IdSet, \sigma)$.
- If $(\mathsf{Vrfy}(mpk, m, IdSet, \sigma){=}1) \wedge (\exists\ Id{\in}IdSet\ s.t.\ (Id{\notin}\mathsf{CIdLst}) \wedge ((m, Id){\notin}\mathsf{MIdLst}))$ then return 1, otherwise return 0.

Fig. 2. Chosen Message Attack against an Identity-Based Multisignature Scheme

4 Σ-Equivocable Commitments and Structured-Instance Zero-Knowledge

Homomorphic Σ-Protocols. Σ-protocol, a notion introduced by Cramer, Damgard and Schoenmakers [CDS94], is a three-move proof system with *special* honest-verifier zero-knowledge (HVZK) and strong soundness properties. Let $R = \{(x, y)\}$ be a relation whose membership can be verified in polynomial time. We consider a special case where X and Y are algebraic groups (for notational simplicity we use multiplicative notation for both), and $R = \{(x, f(x)) \,|\, x \in X\}$ where $f : X \rightarrow Y$ is a homomorphic one-way function. We consider a proof of knowledge system for relation R which we call *homomorphic Σ-protocol* (for R): The prover, on input $x \in X$, sends $a = f(k)$ where $k \overset{r}{\leftarrow} X$. The verifier, on input $y \in Y$, creates a challenge c as a random κ-bit string, and the prover responds with $z = kx^c$. The verifier accepts iff $f(z) = ay^c$. This is a form of several Σ-protocols for known homomorphic one-way functions, *e.g.* Guillou-Quisquater identification scheme [GQ88] for a power function $f_{e,n}(x) = x^e \bmod n$ and Schnorr's scheme [Sch89] for exponentiation $f_{g,p}(x) = g^x \bmod p$. The special HVZK property of a Σ-protocol says that there exists an efficient simulator which on input y computes pair (a, z) for any c with the distribution matching that of the prover. The special strong soundness says that there exists an efficient extractor which computes witness x *s.t.* $(x, y) \in R$ for any y from any pair of accepting conversations (a, c, z) and (a, c', z') *s.t.* $c \neq c'$.

Structured-Instance Zero-Knowledge. Multi-instance zero-knowledge (ZK) (*a.k.a.* multi-theorem ZK) in common reference string (CRS) model requires a two-phase probabilistic poly-time simulator *s.t.* (1) in the first phase, given public parameters, the simulator outputs the CRS string together with some trapdoor information; (2) In the second phase, given a statement and the trapdoor, simulator outputs the simulated proof for that statement. In the single-instance ZK, the simulator knows the statement beforehand and can set the CRS string as a function of this particular statement. Structured instance zero-knowledge proof for relation R introduced above is an intermediary notion: The simulator is given a "core statement" $\mathring{y} \in Y$ before it sets the CRS string, and then it can simulate the proof for statement $y = \mathring{y} \cdot f(\delta)$ for any $\delta \in X$. Here is the formal definition:

Definition 1. *Let X and Y be algebraic groups and $f : X \rightarrow Y$ be a surjective homomorphic one-way function, all indexed by a public parameter* par. *Let $\Pi = (\mathcal{G}, \mathcal{P}, \mathcal{V})$ be a proof system in CRS model for relation $R = \{(x, y) \in X \times Y | \ y = f(x)\}$ where $\mathcal{G}$ is an algorithm that outputs the common reference string. We say that Π is* straight-line ϵ-structured-instance zero-knowledge *if there exist efficient algorithms $\mathcal{S}_1, \mathcal{S}_2$* s.t. *$\mathcal{S}_1$ on input* par *and a core instance $\mathring{y} \in Y$, outputs the CRS string σ and trapdoor td, while $\mathcal{S}_2$ on input td and a "witness-shift" $\delta \in X$ outputs a simulated proof $\tilde{\pi}$ for instance $y = \mathring{y}f(\delta)$, and for all $(\mathring{x}, \mathring{y}) \in X \times Y$* s.t. *$f(\mathring{x}) = \mathring{y}$ the following two properties hold:*

1. *Statistical difference between the following two distributions is at most ϵ:*

$$\{\sigma \mid (\sigma, td) \leftarrow \mathcal{S}_1(\mathsf{par}, \mathring{y})\}$$
$$\{\sigma \mid \sigma \leftarrow \mathcal{G}(\mathsf{par})\}$$

2. *$\forall$ verifier $\mathcal{V}^*$ and $\forall \delta \in X$, the following two distributions are identical:*

$$\{\tilde{\pi} \mid \tilde{\pi} \leftarrow \mathcal{V}^*(y, \sigma)^{\mathcal{S}_2(td, \delta, \sigma)}; (td, \sigma) \leftarrow \mathcal{S}_1(\mathsf{par}, \mathring{y}); y \leftarrow \mathring{y}f(\delta)\}$$
$$\{\pi \mid \pi \leftarrow \mathcal{V}^*(y, \sigma)^{\mathcal{P}(x, y, \sigma)}; \sigma \leftarrow \mathcal{G}(\mathsf{par}); y \leftarrow \mathring{y}f(\delta); x \leftarrow \mathring{x}\delta\}$$

Commitment Schemes. A commitment scheme $\mathcal{C}$ in the CRS model consists of probabilistic poly-time algorithms CSetup, CKG, Com and Open. CSetup on input the security parameter κ, generates public parameters cpar, which also determine the commitment message space $\mathcal{M}$. $\mathsf{CKG}(\mathsf{cpar})$ generates the commitment key K, $\mathsf{Com}_K(m)$ generates the commitment $\mathcal{C}$ and the decommitment D on message $m \in \mathcal{M}$, and finally $\mathsf{Open}_K(\mathcal{C}, D, m)$ determines if D is a valid decommitment of commitment $\mathcal{C}$ to message m. A commitment scheme must satisfy that if $\mathsf{cpar} \leftarrow \mathsf{CSetup}(1^\kappa)$, $K \leftarrow \mathsf{CKG}(\mathsf{cpar})$, and $(\mathcal{C}, D) \leftarrow \mathsf{Com}_K(m)$, then $\mathsf{Open}_K(\mathcal{C}, D, m) = 1$. Below we define statistical hiding and computational binding properties of commitments because these will be variants of these notions which our scheme satisfies.

ϵ-Hiding: For all $\mathsf{cpar} \leftarrow \mathsf{CSetup}(1^\kappa)$, $m_0, m_1 \in \mathcal{M}$, and $K \leftarrow \mathsf{CKG}(\mathsf{cpar})$, there is less than ϵ statistical difference between the distribution of $\mathcal{C}$'s output by $\mathsf{Com}_K(m_0)$ and the distribution of $\mathcal{C}$'s output by $\mathsf{Com}_K(m_1)$. A commitment scheme is *perfectly* hiding if $\epsilon = 0$.

(t, ϵ)-Binding: For any algorithm $\mathcal{A}$ running in time t and any cpar output by $\mathsf{CSetup}(1^\kappa)$, the probability of $\mathsf{Open}_K(\mathcal{C}, D_0, m_0) = \mathsf{Open}_K(\mathcal{C}, D_1, m_1) = 1$ and $m_0 \neq m_1$ is less than ϵ where $(\mathcal{C}, D_0, D_1, m_0, m_1)$ is outputted by $\mathcal{A}$ on input K and $K \leftarrow \mathsf{CKG}(\mathsf{cpar})$ and probability is over the coins of CKG and $\mathcal{A}$.

Notation: In this paper we only deal with the commitment schemes in which the commitment is a deterministic function of the message and the decommitment.

Therefore we assume there exist a decommitment space denoted as $\mathcal{R}$ and the Com procedure picks decommitment $D \xleftarrow{r} \mathcal{R}$ and computes the commitment $\mathcal{C}$ as the deterministic function of m and D.

Σ-Equivocable Commitments. A commitment scheme is *equivocable* if there exists an efficient simulator that generates commitment key K, indistinguishable from real key, together with a *trapdoor td*. The trapdoor allows simulator to create fake commitments indistinguishable from real ones, and later decommit them to *any* message. Using equivocable commitments, one can compile a Σ-protocol to a multi-instance ZK proof system with straight-line simulation [Dam00]. Here we define a rather restrictive form of equivocability called Σ-equivocability and we show that it is sufficient for compiling Σ-protocols into structured-instance ZK proofs with straight-line simulation. It turns out that structured-instance ZK is sufficient for our application of ZK proofs to multi-/aggregate signatures and multi-instance ZK is not required. Moreover the straight-line simulatability of this system allows us to have multi-/aggregate schemes with concurrency, better exact security and with improved round complexity.

Definition 2. *Let X and Y be algebraic groups and let $f : X \rightarrow Y$ be a homomorphic one-way function, all indexed by a commitment parameter* cpar. *We call a commitment scheme* ϵ-Σ-equivocable for f *if there exist probabilistic poly-time algorithms* tdCKG, tdCom, *and* $\mathsf{RstEquiv}$, *where* $(K, td) \leftarrow \mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$, $(\tilde{\mathcal{C}}, st) \leftarrow \mathsf{tdCom}_K(td)$, *and* $(\tilde{D}, z) \leftarrow \mathsf{RstEquiv}_K(td, st, c, \delta)$, s.t. *for any* cpar *output by* CSetup *and any* $\mathring{y} \in Y$ *the following properties hold:*

1. *There is at most ϵ statistical difference between the distribution of K's output by* $\mathsf{CKG}(\mathsf{cpar})$ *and K's output by* $\mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$.
2. *For all* $(K, td) \leftarrow \mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$, $\delta \in X$, *and* $c \in \{0,1\}^\kappa$, *if* $(\tilde{\mathcal{C}}, st)$ *is output by* $\mathsf{tdCom}_K(td)$ *and* $(\tilde{D}, z)$ *is output by* $\mathsf{RstEquiv}_K(td, st, c, \delta)$ *then $\tilde{D}$ is distributed as random decommitment in $\mathcal{R}$ and* $\mathsf{Open}_K(\tilde{\mathcal{C}}, \tilde{D}, f(z)(\mathring{y}f(\delta))^{-c}) = 1$.

Intuitively definition 2 says that the equivocation procedure, given $(\mathring{y}, c, \delta)$, can open a fake commitment to a message of the form $a = f(z)(\mathring{y}f(\delta))^{-c}$ for some z. This is useful in straight-line simulation of a proof of knowledge for relation $R = \{(x, y) \in X \times Y | \ y = f(x)\}$. For example, let $f : \mathrm{QR}_n \rightarrow \mathrm{QR}_n$ where $f(z) = z^e \pmod n$. Consider the HVZK simulator of the Σ-protocol for proving knowledge of e-th root: This simulator picks random c and z and computes prover's first message $a = z^e y^{-c}$. Below we show that Damgard's compilation [Dam00] (see Figure 3 below) transforms such Σ-protocol to structured-instance zero-knowledge using only such Σ-equivocable commitments, because the simulator can output a fake commitment and then open it to what the Σ-protocol simulator would output as the prover's first message *i.e.* $a = z^e y^{-c}$. Definition 2 implies that a fake commitment can be opened to $a = z^e(\mathring{y}\delta^e)^{-c}$ for any δ and c. Hence the structured-instance zero-knowledge simulator can use this property to simulate a proof for any instance $y = \mathring{y}\delta^e$ where $\mathring{y}$ is set before the simulator creates the CRS string (see theorem 1).

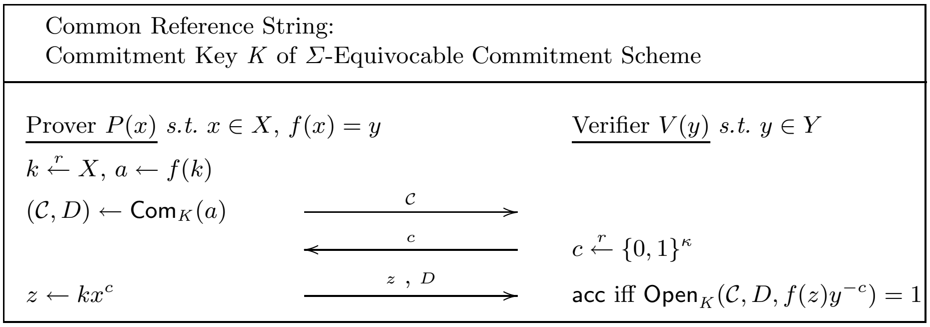

Common Reference String:
Commitment Key K of Σ-Equivocable Commitment Scheme
Prover $P(x)$ s.t. $x \in X$, $f(x) = y$
Verifier $V(y)$ s.t. $y \in Y$
$k \overset{r}{\leftarrow} X$, $a \leftarrow f(k)$
$(\mathcal{C}, D) \leftarrow \mathsf{Com}_K(a)$
$\mathcal{C}$
c
$c \overset{r}{\leftarrow} \{0,1\}^\kappa$
$z \leftarrow kx^c$
z , D
acc iff $\mathsf{Open}_K(\mathcal{C}, D, f(z)y^{-c}) = 1$

Fig. 3. Straight-line simulatable structured-instance ZKPK of pre-image of f

Homomorphic Commitments. We call a commitment scheme multiplicatively homomorphic if there are efficiently computable operations $\otimes$ and $\oplus$ *s.t.* if $\mathsf{Open}_K(\mathcal{C}_1, D_1, m_1) = 1$ and $\mathsf{Open}_K(\mathcal{C}_2, D_2, m_2) = 1$, then $\mathsf{Open}_K(\mathcal{C}, D, m) = 1$ for $\mathcal{C} = \mathcal{C}_1 \otimes \mathcal{C}_2$, $D = D_1 \oplus D_2$, and $m = m_1 m_2$. Accordingly, a commitment scheme is l-restricted multiplicatively homomorphic if the homomorphic operation can be applied on only l commitment-decommitment pairs generated by Com procedure. Our construction is l-restricted multiplicatively homomorphic.

Structured-Instance Zero-Knowledge from Homomorphic Σ-Protocol. Figure 3 shows a construction of a straight-line simulatable structured-instance zero-knowledge proof of knowledge system, in the CRS model, from homomorphic Σ-protocol and Σ-equivocable commitment. This is an identical construction to Damgard's compiler from Σ-protocol to ZKPK proof [Dam00]. Below we show that using only Σ-equivocable commitments the same compilation produces *structured-instance* zero-knowledge proof given *homomorphic* Σ-protocol. As in [Dam00] the resulting protocol is an argument of knowledge, subject to the binding property of the commitment scheme.

Theorem 1. *Let X and Y be algebraic groups, $f : X \leftarrow Y$ a homomorphic one-way function, $\mathcal{C}$ a Σ-equivocable commitment over message space $\mathcal{M} \subseteq Y$. Then the protocol in figure 3 is a straight-line simulatable structured-instance zero-knowledge proof of knowledge of pre-image of f in the CRS model.*

Proof. The straight-line simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, for structured-instance zero-knowledge proof acts as follows: In the first phase, given cpar and $\mathring{y} \in Y$, $\mathcal{S}_1$ runs $\mathsf{tdCKG}(\mathsf{cpar}, \mathring{y})$ to obtain (td, K) and sets the common reference string σ as K. In the second phase, given td and witness shift $\delta \in X$, $\mathcal{S}_2$ runs $\mathsf{tdCom}_K(td)$ to obtain the fake commitment $\tilde{\mathcal{C}}$ and state st and sends $\tilde{\mathcal{C}}$ to the verifier. Upon receiving the challenge c from the verifier, $\mathcal{S}_2$ runs $\mathsf{RstEquiv}_K(td, st, \mathring{y}, \delta)$ to get the response z and fake commitment $\tilde{D}$. According to Σ-equivocability property (definition 2) it immediately follows that $\mathcal{S}$ satisfies conditions in definition 4.

5 Aggregatable Zero-Knowledge Proof of Knowledge of e-th Root

Safe RSA Assumption. Since our construction relies on two *related* instances of RSA cryptosystems which share same RSA modulus n but use two different public exponents e and e', it is convenient for us to use the following notation for RSA instance generation: We call an algorithm $\mathsf{KG_{sRSA}}$ a safe RSA generator if on input security parameter κ and a prime e *s.t.* $2^\kappa \le e \le 2^{2\kappa}$, $\mathsf{KG_{sRSA}}$ generates a pair (n, d) where (1) $n = pq$ *s.t.* $p = 2p' + 1$, $q = 2q' + 1$ and p, q, p' and q' are all prime numbers *s.t.* $|p'| = |q'|$ and $p', q' > 2^{2\kappa}$ and (2) $d = e^{-1} \bmod \phi(n)$. For later use we define $n' = p'q'$. The advantage of an algorithm $\mathcal{A}$ in breaking the RSA(e) problem is defined as

$$\mathbf{Adv}^{\mathsf{ow\_RSA}}_{\mathsf{KG_{sRSA}},\mathcal{A},\mathsf{e}}(\kappa) = \Pr[x^e \overset{n}{\equiv} y \mid (n, d) \overset{r}{\leftarrow} \mathsf{KG_{sRSA}}(\kappa, e); y \overset{r}{\leftarrow} \mathbb{Z}_n^*; x \overset{r}{\leftarrow} \mathcal{A}(n, e, y)] \quad (1)$$

We say algorithm $\mathcal{A}$, (t, ϵ)-breaks the RSA(e) problem on security parameter κ if $\mathcal{A}$ runs in time at most t and $\mathbf{Adv}^{\mathsf{ow\_RSA}}_{\mathsf{KG_{sRSA}},\mathcal{A},\mathsf{e}}(\kappa) \ge \epsilon$. We say that the RSA($e$) problem is (t, ϵ)-hard (for security parameter κ) if no algorithm $\mathcal{A}$, (t, ϵ)-breaks it. We note that the requirement that $p', q' > 2^{2\kappa}$ is just a lower-bound we introduce to enable any party to choose "secondary" public exponent e' *s.t.* $\gcd(e', \phi(n)) = 1$ and $e' > le$ where l is a maximum number of participants in any single instance of the multi-signature scheme.

5.1 RSA-Based Multiplicatively Homomorphic Σ-Equivocable Commitment

Let e and e' be two prime numbers *s.t.* $2^\kappa \le e, e' \le 2^{2\kappa}$ and $e \le e'/l$ for some integer l and let (n, d) be output by $\mathsf{KG_{sRSA}}(\kappa, e)$. This assures that both (n, e) and (n, e') are safe RSA instances. We describe an efficient commitment scheme, which is computationally binding under the RSA(e') assumption, has l-restricted multiplicatively homomorphic property on message space $\mathcal{M} = \mathrm{QR}_n$, and is Σ-equivocable for $f(x) = x^e \pmod n$. Curiously, this commitment is statistically hiding only for the messages picked from a specific subset of the message space, but in our application of this commitment scheme to straight line simulatable ZKPK of e-th root, standard hiding property is not necessary, and Σ-equivocability property for the above function is sufficient.

- $\mathsf{CSetup}(\kappa)$: Pick prime numbers e and e' *s.t.* $2^\kappa \le e, e' \le 2^{2\kappa}$ and $e \le e'/l$. Run $\mathsf{KG_{sRSA}}$ on input (κ, e) to obtain (n, d). Set $\mathsf{cpar} \leftarrow (n, e, e')$.
- $\mathsf{CKG}(n, e, e')$: Pick $h \overset{r}{\leftarrow} \mathrm{QR}_n$ and set $K \leftarrow (n, e, e', h)$. Note that it is easy to sample random elements in QR_n by squaring a random element in $\mathbb{Z}_n^*$.
- $\mathsf{Com}_K(m)$: Pick $r \overset{r}{\leftarrow} \mathbb{Z}_e$ and set $\mathcal{C} \leftarrow h^r m^{e'}$ and $D \leftarrow r$. (Hence the decommitment space is $\mathbb{Z}_e$.)
- $\mathsf{Open}_K(\mathcal{C}, r, m)$: Accept iff $\mathcal{C} = h^r m^{e'}$ and $0 \le r < e'$.
- $\mathsf{tdCKG}((n, e, e'), \mathring{y})$: Pick $\gamma \overset{r}{\leftarrow} [n]$, and set $h \leftarrow (\mathring{y})^{\gamma e'}$, $K \leftarrow (n, e, e', h)$, and $td \leftarrow (\gamma, \mathring{y})$.

- $\mathsf{tdCom}_K(td)$: Pick $s \xleftarrow{r} \mathbb{Z}_e$ and return $(\tilde{\mathcal{C}}, st)$ where $\tilde{\mathcal{C}} = (\mathring{y})^{e's}$ and $st = s$.
- $\mathsf{RstEquiv}_K(td, st, c, \delta)$: Compute $r = (s + c)\gamma^{-1} \pmod{e}$ and $i = (s + c - \gamma r)/e$ (over integers) and return (r, z) where $z = (\mathring{y})^i(\delta)^c$.

Statistical Hiding. This commitment scheme is ϵ-hiding for the messages picked from $\tilde{\mathcal{M}} \subset \mathrm{QR}_n$ where $\tilde{\mathcal{M}} = \{h^{i(e')^{-1}} | i \in [\epsilon e/2]\}$ and h is determined by the commitment key. To argue this note that the maximum statistical difference between the distributions of the commitments to $m_0, m_1 \in \tilde{\mathcal{M}}$ happens when they correspond to $i = 0$ and $i = \epsilon e/2$ respectively. This way the distributions of the commitments would be $\{h^r\}_{r \xleftarrow{r} [e]}$ and $\{h^{r+\epsilon e/2}\}_{r \xleftarrow{r} [e]}$ respectively which has a statistical difference equal to ϵ.

Computational Binding. This commitment scheme is (t, ϵ)-binding if RSA(e') problem is (t, ϵ)-hard. Indeed given the challenge (n, e', h), one can use the attacker on binding to find the e'-th root of h. The reduction runs the binding attacker to obtain $(\mathcal{C}, r, m, r', m')$ *s.t.* $\mathsf{Open}_K(\mathcal{C}, r, m) = \mathsf{Open}_K(\mathcal{C}, r', m') = 1$ and $m \neq m'$. Since $\mathcal{C} = h^r m^{e'} = h^{r'} m'^{e'}$ it follows that $h^{r-r'} = (m'/m)^{e'}$. Now since $r, r' < e'$, then $\gcd(e', r - r') = 1$ and using extended Euclidian algorithm one can compute α, β *s.t.* $\alpha(r - r') + \beta e' = 1$. Thus $h = h^{\alpha(r-r')+\beta e'} = ((m'/m)^\alpha h^\beta)^{e'}$ and e'-th root of h can be computed as $(m'/m)^\alpha h^\beta$.

l-Restricted Multiplicative Homomorphism. This commitment scheme is multiplicatively homomorphic on QR_n in the sense that up to $l \leq \lfloor e'/e \rfloor$ messages can be combined: If $\{(\mathcal{C}_i, r_i)\}_{i=1..l}$ are commitment-decommitment pairs for messages $m_1, ..., m_l \in \mathrm{QR}_n$ each computed by the commitment procedure, then $r = \sum_{i=1}^{l} r_i$(over integers) is a valid decommitment for commitment $\mathcal{C} = \prod_{i=1}^{l} \mathcal{C}_i$ for message $m = \prod_{i=1}^{l} m_i$. Note that by setting $e' \geq e2^\kappa$, homomorphism can be used on any feasible set of messages.

Σ-Equivocability. This commitment scheme is $2^{-2\kappa}$-Σ-equivocable for function (family) $f_{(n,e)}(x) = x^e \pmod{n}$. First note that for every (n, e, e') output by CSetup and every $\mathring{y} \in \mathrm{QR}_n$ *s.t.* $\mathring{y}$ is a generator of QR_n, the distributions of keys generated by $\mathsf{CKG}(n, e, e')$ and $\mathsf{tdCKG}((n, e, e'), \mathring{y})$ are at most $2^{-2\kappa}$ apart, because CKG chooses the key h as a random element in QR_n while tdCKG picks $h = (\mathring{y})^{e'\gamma}$ for e' *s.t.* $\gcd(e', \phi(n)) = 1$ and γ chosen at random in $[n]$. Moreover the statistical difference between $[n]$ and $[4n']$ is equal to $1 - 4n'/n < 2^{2\kappa}$. Secondly, if $\mathring{y}$ is a generator of QR_n then for every $\gamma \in [n]$, every $\delta \in \mathrm{QR}_n$ and every $c \in \{0, 1\}^\kappa$, according to the code of tdCom and $\mathsf{RstEquiv}$, r, z satisfy $s + c = \gamma r + ie$ and $z = (\mathring{y})^i(\delta)^c$, therefore for $m = z^e(\mathring{y}(\delta)^e)^{-c}$ we have $\tilde{\mathcal{C}} = h^r m^{e'}$, and hence $\mathsf{Open}(\tilde{\mathcal{C}}, r, m) = 1$. Moreover the distribution of the decommitments in the equivocation process *i.e.* $\{\tilde{r} | s \xleftarrow{r} \mathbb{Z}_e; \tilde{r} \leftarrow (s + c)\gamma^{-1} \pmod{e}\}$ is identical to uniform distribution over $\mathbb{Z}_e$.

Corollary 1. *Consider prime number* $2^\kappa \leq e \leq 2^{2\kappa}$ *and let* n *be a safe* RSA *modulus output by* $\mathsf{KG}_{\mathsf{sRSA}}$ *on input* e *and security parameter* κ*. Consider compilation shown in figure* 3 *and let the function (family)* f *be* $f_{(n,e)} : \mathrm{QR}_n \to \mathrm{QR}_n$

s.t. $f_{(n,e)}(x) = x^e \pmod{n}$ *and let the compilation be instantiated with the commitment scheme described in this section. Then from theorem 1, it immediately follows that the resulting scheme is a straight-line structured-instance zero-knowledge proof of knowledge of e-th root.*

6 Identity-Based Multisignature Scheme Based on RSA

We describe our IBMS scheme based on the RSA assumption. The scheme takes two communication rounds, requires two double-exponentiations per party for signing and one triple-exponentiation for verification. The scheme is based on the GQ ID-based identification protocol [GQ88], which is the Σ-protocol for proving knowledge of e-th root. Each party simply executes the aggregatable zero-knowledge proof of e-th root of its (hashed) identity string, using the straight-line simulatable aggregatable ZKPK of e-th root described in Section 5. Figure 4 contains the Setup, KeyDer, MSign and Vrfy algorithms for this IBMS scheme.

Note on multi-signature length. In Figure 4 the final multi-signature is a tuple $(z, \mathcal{C}, D)$ where $z \in \mathbb{Z}_n^*$ and $(\mathcal{C}, D) \in \mathbb{Z}_n^* \times \mathbb{Z}_e$ is a commitment-decommitment pair on message $a = z^e(\mathring{y})^{-c}$. However this commitment can be computed as a deterministic function of the committed message a and the decommitment D (see Section 4). Therefore $\mathcal{C}$ can be computed given (z, c, D), and hence one can use (z, c, D) as the final multi-signature, which reduces the multi-signature size to $|\mathbb{Z}_n^*| + |\mathbb{Z}_e| + \kappa < |n| + 2\kappa + \log l$.

Theorem 2. *If* $RSA(e)$ *and* $RSA(e')$ *problems are* (t', ϵ')*-hard, and the IBMS scheme in figure 4 is instantiated with commitment scheme in section 5, which is* (t_B, ϵ_B)*-binding and* ϵ_E*-*Σ*-equivocable for function* $f_{(n,e)}(x) = x^e \pmod{n}$*, then the resulting IBMS scheme is* $(t, \epsilon, n, q_k, q_s, q_h)$*-secure in random oracle model where*

$$t \geq \frac{1}{2}\min(t', t_B) - (3q_s + q_h)t_{exp}$$

$$\epsilon \leq 4q_k\sqrt{(\epsilon' + \epsilon_B + \epsilon_E)q_h + \left(\frac{q_h}{2^{\kappa+1}}\right)^2} + \frac{q_k q_h}{2^{\kappa-1}} + \epsilon_E$$

and t_{exp} *is the time of one exponentiation in* $\mathbb{Z}_n^*$.

Proof. Let $\mathcal{C} = (\mathsf{CKG}, \mathsf{Com}, \mathsf{Open}, \mathsf{tdCKG}, \mathsf{tdCom}, \mathsf{RstEquiv})$ be a commitment scheme for public parameters $\mathsf{cpar} = (n, e, e')$ and the message space $\mathcal{M}$ equal to QR_n. Assume $\mathcal{C}$ is l-restricted multiplicatively homomorphic, (t_B, ϵ_B)-binding and ϵ_E-Σ-equivocable for $f_{(e,n)}(x) = x^e \pmod{n}$. Given a $(t, \epsilon, n, q_k, q_s, q_h)$-forger $\mathcal{F}$, consider two simulators $\mathcal{B}_0$ and $\mathcal{B}_1$ that simulate the role of the honest player as in the experiment $\mathbf{Exp}_{\mathsf{IBMS}}^{\mathsf{uu-cma}}$ interacting with the forger $\mathcal{F}$. $\mathcal{B}_0$ takes as an input a set $\{c_1, c_2, ..., c_{q_h}\}$ where c_i's are in $\{0,1\}^\kappa$ and runs Setup procedure to obtain (mpk, msk) and follows the real protocol *i.e.* answers to forger's key derivation queries and signing queries using procedures KeyDer and MSign

1. **Setup**(1^κ):

   Let l be the maximum number of players in the IBMS scheme. Pick prime numbers e and e' *s.t.* $2^\kappa \le e, e' \le 2^{2\kappa}$ and $2^{\kappa+1}l < e < e'/l$. Run $\mathsf{KG}_{\mathsf{sRSA}}$ on input (κ, e) to obtain (n, d). Note that $\gcd(e', \phi(n)) = 1$ because $\phi(n) = 4p'q'$ where $p', q' > 2^{2\kappa}$. Run $\mathsf{CKG}(n, e, e')$ to obtain the commitment key K. Set $mpk = (n, e, e', K)$ and $msk = d$. Assume $\mathcal{H}_1 : \{0,1\}^* \to \mathrm{QR}_n$ and $\mathcal{H}_2 : \mathrm{QR}_n \times \{0,1\}^* \times \mathrm{QR}_n \to \{0,1\}^\kappa$ are random oracles that every other algorithm in the protocol has access to them.

2. **KeyDer**(msk, Id):

   The PKG computes $x_{Id} \leftarrow (\mathcal{H}_1(Id))^{2d} (\mathrm{mod}\ n)$, sets the private key of the user with identity Id as $sk_{Id} \leftarrow x_{Id}$ and sends it back to him via a secure and authenticated channel.

3. **MSign**: Let $\mathcal{P}$ be the set of players participating in the protocol. Each player determines $\mathcal{P}$ after the first step of MSign. Player with identity Id_i on input (mpk, m, sk_{Id_i}), performs the following steps:

   3.1 Pick $k_i \xleftarrow{r} \mathrm{QR}_n$, $a_i \leftarrow k_i^e$; Set $(\mathcal{C}_i, D_i) \xleftarrow{r} \mathsf{Com}_K(a_i)$ and broadcast $(Id_i, \mathcal{C}_i)$;
   3.2 Upon receiving $(Id_j, \mathcal{C}_j)\ \forall P_j \in \mathcal{P}$, Set $IdSet \leftarrow \{Id_j\}_{P_j \in \mathcal{P}}$ and $\mathcal{C} \leftarrow \bigotimes_{P_j \in \mathcal{P}} \mathcal{C}_j$; Set $c \leftarrow \mathcal{H}_2(\mathcal{C}, IdSet, m)$; Compute $z_i \leftarrow k_i (x_{Id_i})^c$ and broadcast (z_i, D_i);
   3.3 Output multisignature $\sigma = (z, \mathcal{C}, D)$, where $z = \prod_{P_j \in \mathcal{P}} z_j$ and $D = \bigoplus_{P_j \in \mathcal{P}} D_j$.

4. **Vrfy**$(mpk, m, IdSet, \sigma)$:

   Parse σ as $(z, \mathcal{C}, D)$ and mpk as (n, e, e', K); Set $c \leftarrow \mathcal{H}_2(\mathcal{C}, IdSet, m)$; $y \leftarrow \prod_{Id_i \in IdSet} \mathcal{H}_1(Id_i)^2$; If $\mathsf{Open}_K(\mathcal{C}, D, z^e y^{-c}) = 1$ then accept otherwise reject.

Fig. 4. Identity-based multisignature scheme based on RSA

respectively. Additionally, $\mathcal{B}_0$ answers the forger's hash queries and performs an extra finalization process by following the procedures $\mathsf{SimHash}$ and $\mathsf{Finalize}$ in Figure 5. The simulator $\mathcal{B}_1$, on the other hand, takes as an input an RSA challenge $(n, e, \mathring{y})$ and a set $\{c_1, c_2, ..., c_{q_h}\}$ where c_i's are in $\{0,1\}^\kappa$ and follows the Init, $\mathsf{SimKeyDer}$, $\mathsf{SimMSign}$, $\mathsf{SimHash}$ and $\mathsf{Finalize}$ procedures detailed in Figure 5 to perform the initialization, answering to key derivation, signing and hash queries and finalization processes, respectively. Intuitively, the simulator $\mathcal{B}_1$ uses Coron's technique [Cor00] to embed the RSA challenge in the hashes of the ID's of the players with some biased probability $1 - \rho$ hoping that the forgery be based upon the ID of the player for which the RSA challenge is indeed embedded. This way $\mathcal{B}_1$ passes the signing queries on behalf of identity Id just like real protocol using the procedure MSign if the RSA challenge is not embedded in the hash of Id and otherwise $\mathcal{B}_1$ uses the straight-line structured-instance zero-knowledge simulator for proof of knowledge of e-th root (see corollary 1) to simulate the signature protocol on behalf of the identity Id. Both $\mathcal{B}_0$ and $\mathcal{B}_1$, after receiving a valid forgery from $\mathcal{F}$, perform a finalization phase in which the forged multisignature is returned together with the index of the hash responses upon which they are based. Namely both $\mathcal{B}_0$ and $\mathcal{B}_1$ return $(j, (m, IdSet, \sigma))$ *s.t.* $\mathsf{Vrfy}(mpk, m, IdSet, \sigma) = 1$ and there exists at least one uncorrupted Id *s.t.*

(m, Id) is never queried for signing. The simulators $\mathcal{B}_0$ and $\mathcal{B}_1$ set up empty tables H_1 and H_2 to simulate the hash functions $\mathcal{H}_1$ and $\mathcal{H}_2$ respectively and use the set $\{c_1, c_2, ..., c_{q_h}\}$ to answer to the hash queries to $\mathcal{H}_2$ which enables the utilization of forking lemma (as formulated *e.g.* in [BN06, BCJ08]).

Now for $I \in \{0,1\}$ let's lower-bound $acc_{\mathcal{B}_I}$ the probability that $\mathcal{B}_I$ generates a "useful" output *i.e.* an output other than $(0, \lambda)$. This happens when $\mathcal{B}_I$ does not abort in any of the key derivation queries or finalization procedure. Therefore $acc_{\mathcal{B}_I} \geq \rho^{q_K}(1-\rho)$. This function reaches its maximum when $\rho = q_K/(q_K+1)$. Substituting this value of ρ yields:

$$acc_{\mathcal{B}_I} \geq \left(\frac{q_K}{q_K+1}\right)^{q_K}\left(1 - \frac{q_K}{q_K+1}\right) \geq \frac{1}{q_K}\left(\frac{q_K}{q_K+1}\right)^{q_K+1} \geq \frac{1}{4q_K}$$

For $I \in \{0,1\}$, consider $\mathcal{F}_{\mathcal{B}_I}$-the forking algorithm associated with $\mathcal{B}_I$. The success event of $\mathcal{F}_{\mathcal{B}_I}$ denoted by $E^{\mathcal{B}_I}$ is that the algorithm $\mathcal{B}_I$ outputs two tuples $(c_j, (x, n_1, m, IdSet, \sigma))$ and $(\tilde{c_j}, (\tilde{x}, \tilde{n_1}, \tilde{m}, Id\tilde{S}et, \tilde{\sigma}))$ *s.t.* $c_j \neq \tilde{c}_j$ where j is the index of the hash responses upon which the forged multisignature is based. Since the random coins of the algorithm $\mathcal{B}_I$ and the hash responses of the algorithm $\mathcal{B}_I$ previous to j^{th} query are the same in the first and second executions, all the computations and communications and in particular the queries submitted to the hash function $\mathcal{H}_2$ before j^{th} query must be the same, too. Thus the occurrence of $E^{\mathcal{B}_I}$ implies $IdSet = Id\tilde{S}et$, $\mathcal{C} = \tilde{\mathcal{C}}$ and $m = \tilde{m}$. Note that $IdSet = Id\tilde{S}et$ also implies $y = \tilde{y}$. This is because $y = \prod_{Id_i \in IdSet} (\mathcal{H}_1(Id_i))^2$, $\tilde{y} = \prod_{Id_i \in Id\tilde{S}et} (\mathcal{H}_1(Id_i))^2$ and the values for $\mathcal{H}_1(Id_i)$ for all $Id_i \in IdSet$ is fixed before the fork. The success event $E^{\mathcal{B}_I}$ can be partitioned into two cases (1) event $E_1^{\mathcal{B}_I}$ in which $E^{\mathcal{B}_I}$ happens and $z^e y^{-c} = \tilde{z}^e \tilde{y}^{-\tilde{c}}$ (2) event $E_2^{\mathcal{B}_I}$ in which $E^{\mathcal{B}_I}$ happens and $z^e y^{-c} \neq \tilde{z}^e \tilde{y}^{-\tilde{c}}$. Obviously $E^{\mathcal{B}_I} = E_1^{\mathcal{B}_I} \cup E_2^{\mathcal{B}_I}$ and hence $\Pr[E^{\mathcal{B}_I}] \leq \Pr[E_1^{\mathcal{B}_I}] + \Pr[E_2^{\mathcal{B}_I}]$. On the other hand, according to the forking lemma, $E^{\mathcal{B}_I}$ can be lower bounded by $\epsilon_{\mathcal{B}_I}$, the success probability of the simulator $\mathcal{B}_I$:

$$acc_{\mathcal{B}_I} \cdot \left(\frac{acc_{\mathcal{B}_I}}{q_h} - \frac{1}{2^\kappa}\right) \leq \Pr[E^{\mathcal{B}_I}] \leq \Pr[E_1^{\mathcal{B}_I}] + \Pr[E_2^{\mathcal{B}_I}] \tag{2}$$

If c_i's are uniformly distributed in $\{0,1\}^\kappa$ then $\mathcal{F}$'s view in interaction with $\mathcal{B}_0$ is identical to the real execution of the protocol. As for $\mathcal{B}_1$, since $\mathcal{C}$ is ϵ_E-Σ-equivocable, by straight-line structured-instance simulatability of ZKPK of e-th root, firstly the distributions of the commitment keys in the simulation and in the real protocol are at most ϵ_E apart and secondly the distribution of the tuples $(\mathcal{C}_1, D_1, z_1)$ generated in each signature instance in the interaction between $\mathcal{F}$ and $\mathcal{B}_1$ is identical the distributions of the same variables in the real execution. Thus, since our simulation is straight line, total distance between $\mathcal{F}$'s view in interaction with $\mathcal{B}_1$ and in real execution is at most ϵ_E. This implies in particular that $\epsilon_{\mathcal{B}_0} = \epsilon$, $|\epsilon_{\mathcal{B}_1} - \epsilon| \leq \epsilon_E$ and $|\Pr[E_2^{\mathcal{B}_0}] - \Pr[E_2^{\mathcal{B}_1}]| \leq \epsilon_E$. So $\epsilon/4q_k \leq acc_{\mathcal{B}_0}$ and $(\epsilon - \epsilon_E)/4q_k \leq acc_{\mathcal{B}_0}$. Thus equation (2) becomes:

$$\frac{\epsilon - \epsilon_E}{4q_k}\left(\frac{\epsilon - \epsilon_E}{4q_k q_h} - \frac{1}{2^\kappa}\right) \leq \Pr[E_1^{\mathcal{B}_1}] + \Pr[E_2^{\mathcal{B}_0}] + \epsilon_E \tag{3}$$

Init$(n, e, \mathring{y})$:

Pick prime e' where $el \leq e' \leq 2^{2\kappa}$ and run $\mathsf{tdCKG}((n, e, e'), \mathring{y})$ to get (td, K), set mpk as (n, e, e', K) and run $\mathcal{F}$ on input mpk;

SimKeyDer(Id):

Query $\mathcal{H}_1$ on Id and look up $\mathsf{H}_1[Id]$ to get (b, δ, y). If $b = 0$, return δ otherwise abort the simulation with failure outputting $(0, \lambda)$.

SimMSign(m, Id):

Query $\mathcal{H}_1$ on Id and look up $\mathsf{H}_1[Id]$ to get (b, δ, y). If $(b = 0)$ then run $\mathsf{MSign}(m, Id)$; otherwise:

```
−$(\tilde{\mathcal{C}}, st) \leftarrow \mathsf{tdCom}_K(td)$;
    Send $(Id, \tilde{\mathcal{C}})$ to $\mathcal{F}$;
−Upon receiving $(Id_j, \mathcal{C}_j)$ for $P_j \in \mathcal{P}$,
    $IdSet \leftarrow \{Id_j\}_{P_j \in \mathcal{P}}$; $\mathcal{C} \leftarrow \bigotimes_{P_j \in \mathcal{P}} \mathcal{C}_j$;
    $c \leftarrow \mathcal{H}_2(\mathcal{C}, IdSet, m)$;
    $(\tilde{D}, \tilde{z}) \leftarrow \mathsf{RstEquiv}_K(td, st, c, \delta)$;
    Send $(\tilde{z}, \tilde{D})$ to $\mathcal{F}$;
−$z \leftarrow \prod_{P_j \in \mathcal{P}} z_j$; $D \leftarrow \bigoplus_{P_j \in \mathcal{P}} D_j$;
    Output $\sigma = (z, \mathcal{C}, D)$;
```

SimHash:

$\mathcal{H}_1(Id)$: If Id is not previously queried, pick δ uniformly at random from QR_n, toss a biased coin b so that $b = 0$ with probability ρ and $b = 1$ with probability $1 - \rho$. If $b = 0$, set $y \leftarrow \delta^e$ otherwise set $y \leftarrow \mathring{y}\delta^e$. Store (b, δ, y) to $\mathsf{H}_1[Id]$. Return $\mathsf{H}_1[Id]$.

$\mathcal{H}_2(\mathcal{C}, IdSet, m)$: If $(\mathcal{C}, IdSet, m)$ is an i^{th} distinct query of $\mathcal{F}$ to $\mathcal{H}_2$, then query $\mathcal{H}_1(Id_i)$ for every $Id_i \in IdSet$ and set $\mathsf{H}_2[(\mathcal{C}, IdSet, m)] \leftarrow c_i$; Return $\mathsf{H}_2[(\mathcal{C}, IdSet, m)]$;

Finalize:

Upon receiving a valid forgery $(m, IdSet, \sigma)$ from $\mathcal{F}$, parse σ as $(z, \mathcal{C}, D)$ and query $\mathcal{H}_2$ on $(\mathcal{C}, IdSet, m)$. Let $IdSet_0 = \{Id_i | b_i = 0\}$ and $IdSet_1 = \{Id_i | b_i = 1\}$. If $IdSet_1 = \emptyset$ then abort the simulation with failure outputting $(0, \lambda)$. Otherwise set $x \leftarrow \prod_{Id_i \in IdSet}(x_i)$, $n_1 = |IdSet_1|$ and return $(j, (x, n_1, m, IdSet, \sigma))$ where j is the index of c in the hash table H_2.

Fig. 5. The procedures SimHash and Finalize that $\mathcal{B}_0$ and $\mathcal{B}_1$ use and the procedures Init, SimKeyDer, and SimMSign that $\mathcal{B}_1$ uses

The actual reduction algorithm $\mathcal{R}$, runs both $\mathcal{F}_{\mathcal{B}_0}$ and $\mathcal{F}_{\mathcal{B}_1}$. If ${E_1}^{\mathcal{B}_1}$ happens, then $z^e y^{-c_j} = \tilde{z}^e \tilde{y}^{-\tilde{c}_j}$. Substituting $y = \tilde{y} = (\mathring{y})^{2n_1} x^{2e}$ where n_1 is the number of players for whom the reduction has embedded the challenge (see figure 5) yields

$$\left((z/\tilde{z})x^{2(c_j - \tilde{c}_j)}\right)^e = (\mathring{y})^{2n_1(c_j - \tilde{c}_j)} \tag{4}$$

Now since $l2^{\kappa+1} < e$, therefore $\gcd(e, n_1(c_j - \tilde{c_j})) = 1$ and one can easily compute the e-th root of $\mathring{y}$ using the extended Euclidean algorithm.

If ${E_2}^{\mathcal{B}_0}$ happens, then $\mathcal{R}$ immediately translates it into an attack against binding property of commitment scheme $\mathcal{C}$ by returning $(c, D, \tilde{D}, z^e y^{-c_j}, \tilde{z}^e \tilde{y}^{-\tilde{c}_j})$. To see this note that as argued before, $y = \tilde{y}$, $\mathcal{C} = \tilde{\mathcal{C}}$ and since ${E_2}^{\mathcal{B}_0}$ is occurred, thus $z^e y^{-c_j} \neq \tilde{z}^e \tilde{y}^{-\tilde{c}_j}$ and due to validity of the forgeries we have $\mathsf{Open}_K(\mathcal{C}, D, z^e y^{-c_j}) = \mathsf{Open}_K(\mathcal{C}, \tilde{D}, \tilde{z}^e \tilde{y}^{-\tilde{c}_j}) = 1$. Moreover the commitment key K is outputted by CKG in the execution of $\mathcal{B}_0$. Thus $\Pr[E_1^{\mathcal{B}_1}] \leq \epsilon'$ and $\Pr[E_2^{\mathcal{B}_0}] \leq \epsilon_B$ and hence equation (3) becomes

$$\frac{\epsilon - \epsilon_E}{4q_k}\left(\frac{\epsilon - \epsilon_E}{4q_k q_h} - \frac{1}{2^\kappa}\right) \leq \epsilon' + \epsilon_B + \epsilon_E \tag{5}$$

The running time t_R of the reduction algorithm $\mathcal{R}$ is twice the maximum of running time of the algorithms $\mathcal{B}_0$ and $\mathcal{B}_1$. But the running time of $\mathcal{B}_0$ and $\mathcal{B}_1$ is dominated by the running time of the forger $\mathcal{F}$ plus the time spent by the simulators to answer the hash, signing and key derivation queries. Thus $t_R \leq 2(t+(3q_s+q_h)t_{exp})$ where t_{exp} is the time required for exponentiation in $\mathbb{Z}_n^*$. On the other hand since $\mathcal{R}$ either answers the RSA challenge or returns an attack against the binding property of the commitment $\mathcal{C}$, it must be true that $\min(t', t_B) \leq t_R$. Thus:

$$t \geq \frac{1}{2}\min(t', t_B) - (3q_s + q_h)t_{exp}$$

7 Identity-Based Aggregate Signature Scheme

The construction in the previous section can be easily modified to obtain a 2-round identity based aggregate signature (IBAS) scheme provably secure under RSA assumption. For this purpose, one needs to modify the verification algorithm to support the case where different challenges are acquired in step 4.2 of the protocol due to querying $\mathcal{H}_2$ on different messages. More precisely, the resulting IBAS scheme is exactly the same as the scheme described in figure 4 except that its verification algorithm would be as follows: Parse σ as $(z, \mathcal{C}, D)$ and mpk as (n, e, e', K); Compute $R \leftarrow z^e \prod_{Id_i \in IdSet} (\mathcal{H}_1(Id_i))^{2c_i}$ where c_i is output of $\mathcal{H}_2$ on input $(\mathcal{C}, IdSet, m_i)$ and check whether $\mathsf{Open}_K(\mathcal{C}, D, R) = 1$.

The security proof for this IBAS scheme is similar to the proof given in the previous section. Namely the reduction runs two simulators; in one simulator the challenge is embedded in the commitment key and in the other it is embedded in hashes of IDs. Therefore with high probability, if the forgery happens the reduction translates it either to either attack the binding property of the commitment scheme (event E_2 in the previous proof) or to find e-th root of the challenge (event E_1 in the previous proof). The security proof of the IBAS scheme is similar to the security proof of IBMS scheme described in the previous section. The most important difference is that in order to find the e-th root of the challenge we have the following equation instead of equation 4 in the previous proof:

$$(\mathring{y})^{2\sum_{Id_i \in IdSet_1}(\tilde{c_i} - c_i)} = \left((z/\tilde{z}) \prod_{Id_i \in IdSet} x_i^{2(\tilde{c_i} - c_i)}\right)^e$$

Therefore to be able to compute e-th root of $\mathring{y}$, we need $\gcd(e, 2\sum_{Id_i \in IdSet_1}(\tilde{c_i} - c_i)) = 1$. In particular, the reduction succeeds as long as $\sum_{Id_i \in IdSet_1}(\tilde{c_i} - c_i) \neq 0 \bmod e$, *i.e.* unless the challenges in the two branches of the forking algorithm sum up to the same value mod e, which happens with only negligible probability.

References

[BA03] Barr, K., Asanovic, K.: Energy aware lossless data compression. In: MobiSys (2003)

[BCJ08] Bagherzandi, A., Cheon, J.H., Jarecki, S.: Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In: ACM Conference on Computer and Communications Security, pp. 449–458 (2008)

[BGOY10] 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, Revised 21/02/2010 (2010)

[BJ10] Bagherzandi, A., Jarecki, S.: Identity-based aggregate and multi-signature schemes based on RSA (full version) (2010)

[BN06] Bellare, M., Neven, G.: Mult-signatures in the plain public-key model and a general forking lemma. In: Conference on Computer and Communications Security, CCS 2006, pp. 390–399 (2006)

[BN07] 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)

[CDS94] Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)

[Cor00] Coron, J.-S.: On the exact security of full domain hash. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 229–235. Springer, Heidelberg (2000)

[Dam00] 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)

[GHK06] Galindo, D., Herranz, J., Kiltz, E.: On the generic construction of identity-based signatures with additional properties. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 178–193. Springer, Heidelberg (2006)

[GQ88] Guillou, L.C., Quisquater, J.-J.: A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 216–231. Springer, Heidelberg (1998)

[GR06] Gentry, C., Ramzan, Z.: Identity-based aggregate signatures. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 257–273. Springer, Heidelberg (2006)

[Her06] Herranz, J.: Deterministic identity-based signatures for partial aggregation. Comput. J. 49(3), 322–330 (2006)

[KT05] Kim, J., Tsudik, G.: Srdp: Securing route discovery in dsr. In: MobiQuitous, pp. 247–260 (2005)

[MOR01] Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multisignatures. In: ACM Conference on Computer and Communications Security, CCS 2001 (October 2001)

[Nev08] Neven, G.: Efficient sequential aggregate signed data. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 52–69. Springer, Heidelberg (2008)

[Sch89] Schnorr, C.-P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)

[Sha84] 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)

Lattice Mixing and Vanishing Trapdoors: A Framework for Fully Secure Short Signatures and More

Xavier Boyen

Universitas Leodiensis
Institut Montefiore
Liège, Belgium
xb@boyen.org

Abstract. We propose a framework for adaptive security from hard random lattices in the standard model. Our approach borrows from the recent Agrawal-Boneh-Boyen families of lattices, which can admit reliable and punctured trapdoors, respectively used in reality and in simulation. We extend this idea to make the simulation trapdoors cancel not for a specific forgery but on a non-negligible subset of the possible challenges. Conceptually, we build a compactly representable, large family of input-dependent "mixture" lattices, set up with trapdoors that "vanish" for a secret subset which we hope the forger will target. Technically, we tweak the lattice structure to achieve "naturally nice" distributions for arbitrary choices of subset size. The framework is very general. Here we obtain fully secure signatures, and also IBE, that are compact, simple, and elegant.

1 Introduction

Lattices are currently enjoying renewed interest in cryptography, owing to a combination of mathematical elegance, implementation simplicity, provable security reductions, and, more recently, rather dramatic gains in efficiency that bring them closer to the familiar discrete-log and factoring-based approaches. Lattice-based crypto also offers the hope of withstanding quantum computers, against which both discrete-log and factoring-based approaches are known to be utterly defenseless. As a few examples of influential lattice-based cryptosystems and foundations, we mention [5,6,17,18,4,23,22,19], among many others.

Still, by far the biggest barrier to the practical deployment of lattice-based cryptographic systems remains their space inefficiency, which may exceed by several orders of magnitude that of the mainstream. This is especially true for systems based on so-called "hard" random integer lattices, which have essentially no structure other than being periodic modulo the same modulus q along every coordinate axis. Hard lattices have the drawback of requiring voluminous representations, especially when compared to lattices with additional structure such as cyclic or ideal lattices. Being devoid of structure, however, hard lattices may harbor potentially tougher "hard problems" for a safer foundation for crypto.

A primary motivation for lattice cryptography being a hedge against the doomsday of mainstream assumptions, it seems worthwhile to endeavor to build cryptosystems as efficient and provably secure as we can from hard lattices.

1.1 Related Work

A number of progresses toward efficient lattice-based signatures have recently been made. We mention the most closely comparable ones to this work.

Lyubashevsky and Micciancio [16] gave an elegant one-time signature on cyclic lattices, that was then lifted into a many-time signature using a standard tree construction. The signature was stateful and not truly hash-and-sign.

Gentry et al. [13] were the first to realize identity-based encryption from (hard) lattices, with a fully secure construction that implied a very efficient signature as a by-product. Their security proof crucially relied on random oracles.

Cash et al. [11] and Peikert [21] then managed to remove the random oracle and add a hierarchy, using an elegant but bandwidth-intensive bit-by-bit scheme (also concurrently proposed by Agrawal and Boyen [3] *sans* hierarchy), reminiscent of Canetti et al. [10]. Additionally, Peikert [21] showed how to make a simpler signature from the bit-by-bit framework, albeit with an IBE-precluding "salt", using the recent prefix signature technique of Hohenberger and Waters [14].

Boneh et al. [9,1] soon thereafter showed how to avoid the bit-by-bit IBE construction in favor of a compact and efficient all-at-once encoding, creating a selectively secure scheme reminiscent of Boneh and Boyen [8]. Though it does not natively give a secure signature (a costly generic conversion would be needed), we mention it because our framework turns it into a fully secure IBE and more.

Bandwidth Requirements. The following table compares the space efficiency of those recent signature schemes (in hard lattices, unless indicated otherwise).

	SIS strength β	Std. model?	|VerKey| # in $\mathbb{Z}_q$	|SigKey| # in $\mathbb{Z}$	|**Signature**| # in $\mathbb{Z}$
LM'08 [16]	✗ CYCLIC LATTCS	✓	# in $\mathbb{Z}$: $\tilde{O}(\lambda)+\tilde{O}(\ell)$		$\tilde{O}(\ell)$
GPV'08 [13]	$\tilde{O}(n^{1.5})$	ROM	$n\,m$	m^2	m
CHK'09 [11]	$\tilde{O}(\ell\, n^2)$	✓	$2\,\ell^{2+\epsilon}\,n\,m$	m^2	$\ell^{2+\epsilon}\,m$
P'09 [21]	$\tilde{O}(\sqrt{\ell}\,n^{1.5})$	✓	$2\,\ell\,n\,m$	m^2	$\ell\,m$
This work	$\tilde{O}(\ell\,n^{2.5})$	✓	$\ell\,n\,m$	m^2	m

Parameter λ is the security level; ℓ the message bit-size; q the modulus; and m and n the lattice and constraint dimensions where $\lambda \approx n < m = \Theta(n \log q)$. Tabulated SIS strength is the approximation factor β incurred by the security reduction; and $|entity|$ the # of entries in $\mathbb{Z}$ and/or $\mathbb{Z}_q$ per *entity*, with up to $\lceil \log q \rceil \approx \lceil \log \beta \rceil$ bits per entry.

We remark that moderate differences of approximation parameter (β) have limited practical impact compared to variations in the number (#) of entries. Indeed, β is linked to the modulus of $\mathbb{Z}_q$ and the norm of entries in $\mathbb{Z}$; and varying their *magnitudes* by a factor $z = \ell^{k_1} n^{k_2} = \mathrm{poly}(q)$ only affects the information-theoretic bit sizes by a factor $1 + \log z / \log q = \Theta(1)$. By contrast, if we vary the *number* of entries by a factor z, the total bit sizes vary by a factor $\Theta(z)$. Note that $\forall \beta = \mathrm{poly}(n)$, there is an average-case β-SIS reduction [5] from worst-case SIVP with approximation factors $\gamma = \tilde{O}(\beta \sqrt{n})$, widely believed hard $\forall \gamma = \mathrm{poly}(n)$. The concrete parametric hardness of these assumptions is estimated in [12,20].

1.2 Contribution

In this work, we propose a lattice-based encoding framework that generalizes the all-at-once encoding of Agrawal et al. [1]. The relationship of this work to the other one is akin to that linking Waters [24] to Boneh and Boyen [8] in pairing groups. Our goal is to build compact, practical, and "fully secure" signatures and identity-based encryption, from hard integer lattices in the standard model.

Here we focus on signatures. Our main construction is a stateless "hash-and-sign" fully secure signature, i.e., existentially unforgeable under chosen-message attacks, that is about as short as [13]. Our main result is a standard-model security reduction for it and related schemes (from the classic average-case SIS problem, itself reducible from worst-case SIVP and other hard problems [5,23]).

As a bonus, our framework yields a clean "unsalted" construction that extends effortlessly from signature to identity-based private-key extraction. The two are indeed closely related, except that certain tricks used to make signatures secure are incompatible with IBE, such as black-box randomized hashes whose "nonces" would be inaccessible to a non-interactive encrypting party. Our framework does not have this problem, and has already been used to make the IBE scheme of [1] fully secure with little loss of efficiency (see the full version of [1] for details).

1.3 Highlights

Technically, we obtain our compact signature by "mixing" together, in a message-dependent manner, a number of public-key matrices in order to induce in a deterministic way a large family of hard lattices. A signature is a short non-zero vector in the appropriate lattice. For proving adaptive security, we arrange the lattice melange in such a way that a signing trapdoor, i.e., a short lattice basis, is always available for every possible input in the real scheme. In the simulation, faulty trapdoors will be made to vanish through spurious cancellations for a certain, suitably sized set of "challengeable inputs", unknown to the adversary.

A crucial and novel feature of our framework is to ensure that the challengeable inputs are well spread out over the entire input space, regardless of the selected size of the challengeable set. This ensures that, regardless of the actions of the adversary, the simulation will unfold with a significant and more or less invariant probability of success. This simulation robustness property is unusual and key to achieving an efficient security reduction.

Earlier schemes, also based on this principle of small but non-negligible challengeable input sets, generally did not have the luxury of uniform distributions over custom domains; they had to provision complex mechanisms to compensate for the non-uniformity of certain events in function of the adversary's actions. The Waters [24] scheme, for example, contains such a mechanism, prompted by the non-existence of distributions of non-negligible equal weights over exponentially sized groups as used in pairing-based cryptography.

With lattices, by contrast, the possibility to work with smaller moduli gives us an extra handle on the construction of "nice" distributions for a very wide range of challengeable input set sizes. As a result, we obtain security reductions that are simpler, tighter, and more efficient.

2 Lattice Notions

Here we gather a number of useful notions and results from the literature.

We denote by $\|\mathsf{A}\|$ or $\|\mathbf{a}\|$ the ℓ_2-norm of a matrix A or vector $\mathbf{a}$. We denote by $\tilde{\mathsf{A}}$ the Gram-Schmidt ordered orthogonalization of A, and its ℓ_2-norm by $\|\tilde{\mathsf{A}}\|$.

2.1 Random Integer Lattices

Definition 1. Let a basis $\mathsf{B} = \left[\ \mathbf{b}_1 \ \middle|\ \dots \ \middle|\ \mathbf{b}_m\ \right] \in \mathbb{R}^{m\times m}$ be an $m \times m$ matrix with linearly independent columns $\mathbf{b}_1, \dots, \mathbf{b}_m \in \mathbb{R}^m$. The lattice Λ generated by the basis B and its dual Λ^* are defined as (both are m-dimensional),

$$\Lambda = \mathcal{L}(\mathsf{B}) = \left\{\ \mathbf{y} \in \mathbb{R}^m \quad \text{s.t.} \quad \exists \mathbf{s} \in \mathbb{Z}^m\ ,\quad \mathbf{y} = \mathsf{B}\,\mathbf{s} = \sum_{i=1}^{m} \mathsf{s}_i\,\mathbf{b}_i\ \right\}$$

$$\Lambda^* = \left\{\ \mathbf{z} \in \mathbb{R}^m \quad \text{s.t.} \quad \forall \mathbf{y} \in \Lambda\ ,\quad \mathbf{z}^T\,\mathbf{y} = \langle \mathbf{z}, \mathbf{y}\rangle\ \in \mathbb{Z}\ \right\}$$

Definition 2. For a positive integer q (later a prime) and a matrix $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$, define two m-dimensional full-rank integer lattices:

$$\Lambda^{\perp}(\mathsf{A}) = \left\{\ \mathbf{e} \in \mathbb{Z}^m \quad \text{s.t.} \quad \mathsf{A}\,\mathbf{e} = 0 \pmod q\ \right\}$$

$$\Lambda(A) = \left\{\ \mathbf{y} \in \mathbb{Z}^m \quad \text{s.t.} \quad \exists \mathbf{s} \in \mathbb{Z}^n\ ,\quad \mathsf{A}^T\,\mathbf{s} = \mathbf{y} \pmod q\ \right\}$$

These are dual when properly scaled, as $\Lambda^{\perp}(\mathsf{A}) = q\,\Lambda(\mathsf{A})^*$ and $\Lambda(\mathsf{A}) = q\,\Lambda^{\perp}(\mathsf{A})^*$.

2.2 Bases and Trapdoors

A fundamental result in the geometry of numbers is that every lattice Λ has a basis; e.g., see [15]. Implicit in its proof, is the well known fact that any full-rank set $\mathsf{S}_A \subset \Lambda$ can be converted into a basis T_A for Λ with no greater orthogonalized norm $\|\tilde{\mathsf{T}}_A\| \le \|\tilde{\mathsf{S}}_A\|$.

Fact 3. *For a set $\mathsf{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_m\}$ of lattice vectors, let $\tilde{\mathsf{X}} = \{\tilde{\boldsymbol{x}}_1, \dots, \tilde{\boldsymbol{x}}_m\}$ be its Gram-Schmidt ordered orthogonalization. There is a deterministic polynomial-time algorithm that, on input an arbitrary basis of an m-dimensional lattice Λ and a full-rank set $\mathsf{S} = \{\boldsymbol{s}_1, \dots, \boldsymbol{s}_m\} \subset \Lambda$ of lattice vectors, returns a basis $\mathsf{T} = \{\boldsymbol{t}_1, \dots, \boldsymbol{t}_m\}$ of Λ such that $\|\tilde{\boldsymbol{t}}_i\| \le \|\tilde{\boldsymbol{s}}_i\|$ for all $i = 1, \dots, m$.*

Ajtai [6] shows how to sample a uniform matrix $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ with an associated full-rank set $\mathsf{S}_A \subset \Lambda^\perp(\mathsf{A})$ of low-norm vectors orthogonal to A modulo q. Tightness was later improved by Alwen and Peikert [7].

Proposition 4 ([7])**.** For any $\delta_0 > 0$, there is a probabilistic polynomial-time algorithm that, on input a security parameter 1^λ, an odd prime $q = \mathrm{poly}(\lambda)$, and two integers $n = \Theta(\lambda)$ and $m \geq (5 + 3\,\delta_0)\, n \log q$, outputs a statistically $(m\, q^{-\delta_0\, n/2})$-close to uniform matrix $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ and a basis $\mathsf{T}_A \subset \Lambda^\perp(A)$ such that with overwhelming probability $\|\mathsf{T}_A\| \leq O(n \log q)$ and $\|\tilde{\mathsf{T}}_A\| \leq O(\sqrt{n\,\log q})$.

For the purpose of this paper, we take $\delta_0 = 1/3$, assume $L = \tilde{\Omega}(\sqrt{m})$, and summarize the foregoing as follows.

Fact 5. *There is a probabilistic polynomial-time algorithm that, on input a security parameter* 1^λ*, an odd prime* $q = \mathrm{poly}(\lambda)$*, and two integers* $n = \Theta(\lambda)$ *and* $m \geq 6\,n\,\log q$*, outputs a matrix* $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ *statistically close to uniform, and a basis* T_A *for* $\Lambda^\perp(\mathsf{A})$ *with overwhelming probability such that* $\|\tilde{\mathsf{T}}_A\| \leq \tilde{\Theta}(\sqrt{m}) \leq L$*.*

2.3 Discrete Gaussians

Given a basis for an integer random lattice, we recall how to sample random lattice points from a discrete Gaussian distribution whose minimum "width" is function of the norm of the lattice basis. We follow the works of [23,4,19,13].

Definition 6. Let $m \in \mathbb{Z}_{>0}$ be a positive integer and $\Lambda \subset \mathbb{R}^m$ an m-dimensional lattice. For any vector $\mathbf{c} \in \mathbb{R}^m$ and any positive parameter $\sigma \in \mathbb{R}_{>0}$, we define:

$\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp\left(-\pi\frac{\|\mathbf{x}-\mathbf{c}\|^2}{\sigma^2}\right)$: a Gaussian-shaped function on $\mathbb{R}^m$ with center $\mathbf{c}$ and parameter σ, (For $x \in \mathbb{R}$, $\rho_{\sigma,c}(x) \propto \mathcal{N}_{\frac{\sigma}{\sqrt{2\pi}},0}(x)$, the normal probability density of variance $\frac{\sigma^2}{2\,\pi}$ and mean 0.)

$\rho_{\sigma,\mathbf{c}}(\Lambda) = \sum_{\mathbf{x}\in\Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$: the (always converging) discrete integral of $\rho_{\sigma,\mathbf{c}}$ over the lattice Λ,

$\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$: the discrete Gaussian distribution over Λ with center $\mathbf{c}$ and parameter σ,

$$\forall y \in \Lambda \quad , \quad \mathcal{D}_{\Lambda,\sigma,\mathbf{c}}(y) = \rho_{\sigma,\mathbf{c}}(y)/\rho_{\sigma,\mathbf{c}}(\Lambda)$$

For notational convenience, origin-centered $\rho_{\sigma,\mathbf{0}}$ and $\mathcal{D}_{\Lambda,\sigma,\mathbf{0}}$ are abbreviated as ρ_σ and $\mathcal{D}_{\Lambda,\sigma}$.

Gentry et al. [13] show that, given a basis B for a lattice Λ, one can efficiently sample points in Λ with discrete Gaussian distribution for sufficiently large values of σ.

Proposition 7 ([13])**.** There exists a probabilistic polynomial-time algorithm that, on input an arbitrary basis B of an m-dimensional full-rank lattice $\Lambda = \mathcal{L}(\mathsf{B})$, a parameter $\sigma \geq \|\tilde{\mathsf{B}}\|\,\omega(\sqrt{\log m})$, and a center $\mathbf{c} \in \mathbb{R}^m$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$.

For concreteness, we will refer to the algorithm of Proposition 7 as follows:

SampleGaussian$(\mathsf{B}, \sigma, \mathbf{c})$**:** On input a basis B for a lattice $\Lambda \subset \mathbb{R}^m$, a positive real parameter $\sigma \geq \|\tilde{\mathsf{B}}\| \, \omega(\sqrt{\log m})$, and a center vector $\mathbf{c} \in \mathbb{R}^m$, it outputs a fresh random lattice vector $\mathbf{x} \in \Lambda$ drawn from a distribution statistically close to $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$.

2.4 Smoothing Parameter

We recall the notion of smoothing parameter of a lattice which lower-bounds the "density" of points on a lattice across all directions, and how this relates to discrete Gaussian sampling on the lattice.

Micciancio and Regev [19] define the smoothing parameter of a lattice as follows.

Definition 8 ([19])**.** For any m-dimensional lattice Λ and any positive real $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(\Lambda)$ is the smallest real $\eta > 0$ such that $\rho_{1/\eta}(\Lambda^* \setminus \{0\}) \leq \epsilon$.

Micciancio and Regev [19] show that large deviations from lattice points vanish exponentially.

Proposition 9 ([19])**.** For any lattice Λ of integer dimension m, any point $\mathbf{c}$, and any two reals $\epsilon \in (0,1)$ and $\eta \geq \eta_\epsilon(\Lambda)$,

$$\Pr\left\{ \ \mathbf{x} \sim \mathcal{D}_{\Lambda,\eta,\mathbf{c}} \ : \ \|\mathbf{x} - \mathbf{c}\| > \sqrt{m}\,\eta \ \right\} \ \leq \ \frac{1+\epsilon}{1-\epsilon}\, 2^{-m}$$

Peikert and Rosen [22] show that the Gaussian function itself vanishes away from any point.

Proposition 10 ([22])**.** For any lattice Λ of integer dimension m, any center $\mathbf{c} \in \mathbb{R}^m$, any two reals $\epsilon \in (0,1)$ and $\eta \geq 2\,\eta_\epsilon(\Lambda)$, and any lattice point $\mathbf{x} \in \operatorname{span}(\Lambda)$,

$$\mathcal{D}_{\Lambda,\eta,\mathbf{c}}(x) \ \leq \ \frac{1+\epsilon}{1-\epsilon}\, 2^{-m}$$

2.5 Statistical Mixing

We recall some useful statistical mixing properties relating to the reduction of an integer vector modulo a lattice to yield a syndrome.

Ajtai [5] then Regev [23] show that binary combinations of enough vectors alsmost always span the space.

Proposition 11 ([23])**.** Let $m \geq 2\,n \log q$. Then for all except at most some q^{-n} fraction of matrices $\mathsf{A} \in \mathbb{Z}_q^{n \times m}$, the subset sums of the columns of A generate $\mathbb{Z}_q^n$. In other words, for every syndrome $\mathbf{u} \in \mathbb{Z}^n$ there exists a binary vector $\mathbf{e} \in \{0,1\}^m$ such that $\mathsf{A}\,\mathbf{e} = \mathbf{u} \pmod q$.

Gentry et al. [13] show that short Gaussian combinations of any spanning vector set yields uniformity.

Proposition 12 ([13])**.** Assume the columns of $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ generate $\mathbb{Z}_q^n$, and let $\epsilon \in (0,1)$ and $\eta \geq \eta_\epsilon(\Lambda^\perp(\mathsf{A}))$. Then for $\mathbf{e} \sim \mathcal{D}_{\mathbb{Z}^m,\eta}$ the distribution of the syndrome $\mathbf{u} = \mathsf{A}\,\mathbf{e} \bmod q$ is within statistical distance $2\,\epsilon$ of uniform over $\mathbb{Z}_q^n$.

Furthermore, fix $\mathbf{u} \in \mathbb{Z}_q^n$ and let $\mathbf{c} \in \mathbb{Z}^m$ be an arbitrary solution to $\mathsf{A}\,\mathbf{c} = \mathbf{u} \pmod q$. Then the conditional distribution of $\mathbf{e} \sim \mathcal{D}_{\mathbb{Z}^m,\eta}$ given $\mathsf{A}\,\mathbf{e} = \mathbf{u} \pmod q$ is exactly $\mathbf{c} + \mathcal{D}_{\Lambda^\perp(\mathsf{A}),\eta,-\mathbf{c}}$.

Gentry et al. [13] then show that for random A the lattice $\Lambda(\mathsf{A})$ has large minimal distance in ℓ_∞ and thus that $\Lambda^\perp(\mathsf{A})$ has small smoothing parameter.

Proposition 13 ([13])**.** Let q be a prime and n and m be two integers satisfying $m \geq 2\,n\log q$. Then, for all but at most some q^{-n} fraction of matrices $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$, it holds that $\lambda_1^\infty(\Lambda(\mathsf{A})) \geq q/4$. Also, for any such A and any $\omega(\sqrt{\log m})$ function, there is a negligible function $\epsilon(m)$ such that the smoothing parameter $\eta_\epsilon(\Lambda^\perp(\mathsf{A})) \leq \omega(\sqrt{\log m})$.

Combining the previous propositions, Gentry et al. [13] summarize the results as follows.

Fact 14. *Fix a prime q and two integers n and m satisfying $m \geq 2\,n\log q$. For all but at most $2\,q^{-n}$ of matrices $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ and for any Gaussian parameter $\eta \geq \omega(\sqrt{\log m})$, on input $\boldsymbol{e} \sim \mathcal{D}_{\mathbb{Z}^m,\eta}$ the distribution of the syndrome $\boldsymbol{u} = \mathsf{A}\,\boldsymbol{e} \bmod q$ is statistically close to uniform over $\mathbb{Z}^n$.*

2.6 Preimage Sampling

We recall the notion of preimage-samplable functions (PSF) defined in [13], which is based on the combination of a trapdoor construction for integer lattices and an efficient discrete Gaussian sampling algorithm.

Let a uniform matrix $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ and a low-norm basis T_A for the lattice $\Lambda^\perp(\mathsf{A})$. Used in the discrete Gaussian sampling algorithm, the short basis T_A can act as a trapdoor for finding small non-zero solutions $\mathbf{e} \in \mathbb{Z}^m$ of the equation $\mathsf{A}^T\,\mathbf{e} = 0 \pmod q$ or more generally $\mathsf{A}^T\,\mathbf{e} = \mathbf{u} \pmod q$ for any $\mathbf{u} \in \mathbb{Z}_q^n$. This leads to the notion of preimage-samplable functions [13].

We give the following definition of preimage-samplable function, following [13]:

Definition 15. Let λ, q, n, m, and L be as in Fact 5. Let $\sigma \geq L\,\omega(\sqrt{\log m})$ be some Gaussian parameter. A preimage-samplable function family is a collection of maps $f_A : \mathbb{D}_{\mathbb{Z}^m,\sigma} \to \mathbb{Z}_q^n$ from $\mathbb{D}_{\mathbb{Z}^m,\sigma} = \{\mathbf{e} \in \mathbb{Z}^m : \|\mathbf{e}\| \leq \sqrt{m}\,\sigma\} \subseteq \mathbb{Z}^m$ into $\mathbb{Z}_q^n$, and specified by the following four algorithms:

TrapGen(1^λ)**:** On input 1^λ, it uses the algorithm of Fact 5 to obtain a pair $(\mathsf{A}, \mathsf{T}_A)$, where $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ is statistically close to uniform and $\mathsf{T}_A \subset \lambda^\perp(\mathsf{A})$ is a short basis with $\|\tilde{\mathsf{T}}\| \leq L$. The public function parameters are (A, q). The preimage-sampling trapdoor is the basis T_A.

EvalFun$(\mathsf{A}, q, \mathbf{e})$**:** On input function parameters (A, q) and an input point $\mathbf{e} \in \mathbb{D}_{\mathbb{Z}^m,\sigma}$, it outputs the image $f_A(\mathbf{e}) = \mathsf{A}\,\mathbf{e} \bmod q$ in $\mathbb{Z}_q^n$. (The output is undefined on large input $\mathbf{e} \in \mathbb{Z}^m \setminus \mathbb{D}_{\mathbb{Z}^m,\sigma}$.)

SampleDom$(1^{(m)}, \sigma)$**:** On input the $m \times m$ identity matrix $1^{(m)}$ and a Gaussian parameter σ, it outputs $\mathbf{e} \leftarrow \mathsf{SampleGaussian}(1^{(m)}, \sigma, \mathbf{0})$, i.e., outputs an element $\mathbf{e} \in \mathbb{Z}^m$ such that $\mathbf{e} \sim \mathcal{D}_{\mathbb{Z}^m,\sigma}$. The input matrix $1^{(m)}$ conveys the dimension m and its columns give a basis for Gaussian sampling in the lattice $\mathbb{Z}^m$. By Proposition 10, with overwhelming probability $\mathbf{e} \in \mathbb{D}_{\mathbb{Z}^m,\sigma}$.

SamplePre$(\mathsf{A}, q, T_A, \sigma, \mathbf{u})$**:** On input function parameters A and q and a trapdoor T_A, a Gaussian parameter σ as above, and a target image $\mathbf{u} \in \mathbb{Z}_q^n$, it samples a preimage $\mathbf{e} \in \mathbb{D}_{\mathbb{Z}^m,\sigma}$ from the distribution $\mathcal{D}_{\mathbb{Z}^m,\sigma}$ conditioned on the event that $\mathsf{A}\,\mathbf{e} = \mathbf{u} \pmod q$. To do this, it solves for an arbitrary solution $\mathbf{c} \in \mathbb{Z}^m$ in the linear system $\mathsf{A}\,\mathbf{c} = \mathbf{u} \pmod q$; it then samples $\mathbf{d} \leftarrow \mathsf{SampleGaussian}(\mathsf{T}_A, \sigma, -\mathbf{c}) \sim \mathcal{D}_{\Lambda^\perp(\mathsf{A}),\sigma,-\mathbf{c}}$ and outputs $\mathbf{e} = \mathbf{c} + \mathbf{d}$ in $\mathbb{Z}^m$. By Proposition 10, with overwhelming probability $\mathbf{e} \in \mathbb{D}_{\mathbb{Z}^m,\sigma}$.

The construction is correct and efficient by Proposition 12; see [13] for details.

2.7 Elementary Delegation

There are several ways to delegate a short basis for $\Lambda^\perp(\mathsf{A})$ into one for $\Lambda^\perp([\mathsf{A}|\mathsf{B}])$. If there is no one-wayness requirement on the delegation process, then Peikert [21] describes a very effective elementary deterministic way to do this.

Proposition 16 ([21])**.** Take any matrix $\mathsf{A} \in \mathbb{Z}_q^{n\times m_1}$ such that the columns of A span the group $\mathbb{Z}_q^n$. Let an arbitrary $\mathsf{B} \in \mathbb{Z}_q^{n\times m_2}$, and define $\mathsf{F} = [\mathsf{A}|\mathsf{B}]$. There exists a polynomial-time deterministic algorithm that, given A, B, and an arbitrary basis T_A for $\Lambda^\perp(\mathsf{A})$, outputs a basis T_F for $\Lambda^\perp(\mathsf{F})$ while preserving the Gram-Schmidt norm of the basis (i.e., such that $\|\tilde{\mathsf{T}}_F\| = \|\tilde{\mathsf{T}}_A\|$).

2.8 Hardness Assumption

The following lattice problem was first suggested to be hard on average by Ajtai [5] and formally defined by Micciancio and Regev [19].

Definition 17. The Small Integer Solution (SIS) problem in L_2-norm is: given an integer q, a matrix $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$, and a real β, find a non-zero integer vector $\mathbf{e} \in \mathbb{Z}^m$ such that $\mathsf{A}\,\mathbf{e} = \mathbf{0} \pmod q$ and $\|\mathbf{e}\|_2 \le \beta$. The average-case (q, n, m, β)-SIS problem is defined similarly, where A is uniformly random.

This problem was shown to be as hard as certain worst-case lattice problems, first by Ajtai [5], then by Micciancio and Regev [19], and Gentry et al. [13].

Proposition 18 ([13])**.** For any poly-bounded m, any $\beta = \mathrm{poly}(n)$ and for any prime $q \ge \beta \cdot \omega(\sqrt{n \log n})$, the average-case (q, n, m, β)-SIS problems is as hard as approximating the Shortest Independent Vector Problem (SIVP), among others, in the worst case to within certain $\gamma = \beta \cdot \tilde{O}(\sqrt{n})$ factors.

2.9 More Useful Facts

Lemma 19. *Let* $\mathsf{B}_0 \in \mathbb{Z}_q^{n\times m}$*. Let* H *be a scalar* $\mathsf{h} \in \mathbb{Z}_q$ *or a matrix* $\mathsf{H} \in \mathbb{Z}_q^{n\times n}$*. Suppose that* H *is invertible modulo* q *(i.e.,* $|H| \ne 0 \pmod q$ *when* q *is prime). Then, the two preimage-samplable functions* $(\mathsf{B}_0)(\cdot) \bmod q$ *and* $(H\,\mathsf{B}_0)(\cdot) \bmod q$ *from* $\mathbb{Z}^m$ *into* $\mathbb{Z}_q^n$ *admit exactly the same trapdoors* $\mathsf{T}_{B_0} \subset \mathbb{Z}^m$*.*

Proof. For all $\mathbf{e} \in \mathbb{Z}^m$ we have $\mathsf{B}_0\,\mathbf{e} = \mathbf{0} \pmod q$ if and only if $H\,\mathsf{B}_0\,\mathbf{e} = \mathbf{0} \pmod q$, hence the two lattices $\Lambda^\perp(\mathsf{B}_0)$ and $\Lambda^\perp(H\,\mathsf{B}_0)$ are the same. Thus, $\mathsf{T}_{B_0} \subset \Lambda^\perp(\mathsf{B}_0) \Leftrightarrow \mathsf{T}_{B_0} \subset \Lambda^\perp(H\,\mathsf{B}_0)$. $\square$

3 General Simulation Framework

We now describe the core scheme. At a high level, we achieve short signatures with full adaptive security by providing a relatively large number of public-key matrices, which are then "mixed through" together in a message-dependent manner — as opposed to merely juxtaposed as in the constructions of [3,11,21]. In the simulation, the public-key matrices will hide a trapdoor component that has a non-negligible probability of vanishing in the mix for certain unpredictable choices of messages: on those messages the simulator will be unable to answer signature queries, but will be able instead to exploit an existential forgery.

Our key-mixing technique is at some level reminiscent of Waters' scheme [24] in bilinear groups, but with a number of crucial differences. The farther-reaching difference is that in the lattice setting we can exploit the smaller groups and their richer structure to create a (much) more efficient "mixing" effect than in the large cyclic groups of the discrete-log setting. Another difference concerns randomization, which in a lattice setting tends to be rather more involved than in discrete-log settings; our approach is based on the method of randomization by a low-norm matrix from [1], with the small added contribution to show that it can be done in a way that supports the mixing effect that we need.

3.1 Two-Sided Trapdoors

To facilitate the description of the scheme and its proof, we first construct a preimage-samplable function of a special form that will be able to sample short preimages from the same distribution, using either one of two types of trapdoors: "firm" trapdoors will be used in the real scheme, and will never fail to work; "fickle" trapdoors will be used in the simulation, and will be fragile by design.

Lattices with dual trapdoors were first introduced in [9,1]. Here, we seek to let the matrix R, below, be generated as a mixture of certain low-norm matrices. All the algorithms in this subsection are adapted from § 4 of [1].

Definition 20. Consider an algorithm $\mathsf{TwoSideGen}(1^\lambda)$ that outputs two random matrices $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathsf{R} \in Z^{m\times m}$, where A is uniform and R has some distribution $\mathcal{R}$. Let $\mathsf{B} \in \mathbb{Z}_q^{n\times m}$ be an independent third matrix. Write $\mathsf{A}\,\mathsf{R}$ as shorthand for $(\mathsf{A}\,\mathsf{R} \bmod q) \in \mathbb{Z}_q^{n\times m}$, and define,

$$\mathsf{F} \;=\; \big[\mathsf{A} \;\big|\; \mathsf{A}\,\mathsf{R} + \mathsf{B}\big] \quad \in \; \mathbb{Z}_q^{n\times 2m}$$

We say that the pair (F, q) defines the public parameters of a *two-sided function.*

The following lemmas show that a two-sided function (F, q) is a preimage-samplable function given a trapdoor for either A or B, provided that A and R are drawn from suitable distributions.

Lemma 21. *For any parameter $\eta \geq \omega(\sqrt{\log m})$, there exists an efficiently samplable distribution $\mathcal{R}_\eta$ over $\mathbb{Z}^{m\times m}$, such that with overwhelming probability $\mathsf{R} = \sum_{i=1}^{\ell}$ for independent $\mathsf{R}_i \sim \mathcal{R}_\eta$ has norm $\|\mathsf{R}\| \leq \sqrt{m}\,\eta$, and such that for $(\mathsf{A},\mathsf{R}) \sim \mathcal{U}_{\mathbb{Z}_q^{n\times m}} \times \mathcal{R}$ and fixed $\mathsf{B} \in \mathbb{Z}_q^{n\times m}$ the matrix $\mathsf{F} = [\mathsf{A}|\mathsf{A}\,\mathsf{R}+\mathsf{B}] \in \mathbb{Z}_q^{n\times 2m}$ is statistically close to uniform.*

Proof. According to Fact 14, it suffices to pick the columns of R independently wiht $\sim \mathcal{D}_{\mathbb{Z}^m,\eta}$. $\square$

Lemma 22 ("Firm" trapdoor)**.** *Let L and σ be as in Definition 15 and $\mathcal{R}_\eta$ as in Lemma 21. If $[\mathsf{A}|\mathsf{B}] \sim \mathcal{U}_{\mathbb{Z}_q^{n\times 2m}}$ and $\mathsf{T}_A \subset \Lambda^\perp(\mathsf{A})$ of norm $\|\tilde{\mathsf{T}}_A\| \leq L$, then the pair $\big(\mathsf{F} = [\mathsf{A}|\mathsf{B}],\ q\big)$ is a preimage-samplable function in the sense of Definition 15.*

Proof. Per Lemma 21, F is statistically close to uniform in $\mathbb{Z}_q^{n\times 2m}$, thus F has the right distribution. It remains to show how to perform public and trapdoor sampling.

$\mathsf{SampleDom}$. To sample short vectors $\mathbf{e} \sim \mathcal{D}_{\mathbb{Z}^{2m},\sigma}$ in the domain of F, one proceeds exactly as in the GPV scheme, i.e., by executing $\mathsf{SampleDom}(1^{(2m)}, \sigma)$ which does not require any trapdoor.

$\mathsf{SamplePre}$. For preimage sampling, we show how to sample a short preimage $\mathbf{e} \in \mathbb{Z}^{2m}$ of any $\mathbf{u} \in \mathbb{Z}_q^n$ with conditional distribution $\mathcal{D}_{\mathbb{Z}^{2m},\sigma} \mid \mathsf{F}\,\mathbf{e} = \mathbf{u} \pmod q$. Since a random $\mathsf{A} \in \mathbb{Z}_q^{n\times m}$ will almost always span all of $\mathbb{Z}_q^n$, we can use the deterministic delegation mechanism of Proposition 16 to obtain a basis T_F for F with short Gram-Schmidt norm $\|\tilde{\mathsf{T}}_F\| \leq L$. Having such a trapdoor T_F for F, we invoke $\mathsf{SamplePre}(\mathsf{F}, q, \mathsf{T}_F, \sigma, \mathbf{u})$ to obtain a short random preimage $\mathbf{e}$. $\square$

Lemma 23 ("Fickle" trapdoor)**.** *Let L be as in Definition 15, η as in Lemma 21, and $\sigma = L'\,\omega(\sqrt{\log m})$ where $L' = 2\,\eta\,\sigma'\,\sqrt{m}$ and where $\sigma' \geq L\,\sqrt{\ell\,m}\,\omega(\sqrt{\log m})$. Fix a matrix $\mathsf{B} \in \mathbb{Z}_q^{n\times m}$ with a short basis T_B of orthogonalized norm $\|\tilde{\mathsf{T}}_B\| \leq L$. For (A,R) such that $[\mathsf{A}|\mathsf{A}\,\mathsf{R}] \sim \mathcal{U}_{\mathbb{Z}_q^{n\times 2m}}$ and $\|\mathsf{R}\| \leq \eta\,\sqrt{\ell\,m}$, the pair $\big(\mathsf{F} = [\mathsf{A}|\mathsf{A}\,\mathsf{R} + \mathsf{B}],\ q\big)$ is a preimage-samplable function in the sense of Definition 15.*

In this lemma, we allow $\|\mathsf{R}\| \leq \eta\,\sqrt{\ell\,m}$, where the factor $\sqrt{\ell}$ will account for the fact that in the simulation the matrix $\mathsf{R} = \mathsf{R}_{\mathsf{msg}} = \sum_{i=1}^{\ell} \pm \mathsf{R}_i$ for independent R_i of norm $\|\mathsf{R}_i\| \leq \eta\,\sqrt{m}$ and coefficients ± 1 function of the message msg.

Proof. Per Lemma 21, F is statistically close to uniform in $\mathbb{Z}_q^{n\times 2m}$, thus F has the right distribution. We need to show how to perform public and trapdoor sampling.

$\mathsf{SampleDom}$. Sampling short vectors $\mathbf{e} \sim \mathcal{D}_{\mathbb{Z}^{2m},\sigma}$ is done without any trapdoor by invoking $\mathsf{SampleDom}(1^{(2m)}, \sigma)$, as in the previous lemma.

$\mathsf{SamplePre}$. For preimage sampling, we need to show, given any input $\mathbf{u} \in \mathbb{Z}_q^n$, how to sample a short preimage $\mathbf{e} \in \mathbb{Z}^{2m}$ of $\mathbf{u}$ with conditional distribution $\mathcal{D}_{\mathbb{Z}^{2m},\sigma} \mid \mathsf{F}\,\mathbf{e} = \mathbf{u} \pmod q$. We do this in three steps:

1. We build a full-rank set $\mathsf{S}_F \subset \Lambda^\perp(\mathsf{F})$ such that $\|\tilde{\mathsf{S}}_F\| \leq 2\,\eta\,L\,\sqrt{\ell}\,m\,\omega(\sqrt{\log m})$. This is done by independently sampling short vectors $\mathbf{e}_i \in \Lambda^\perp(\mathsf{F})$ until a linearly

independent set of $2\,m$ such vectors is found. To sample one short vector $\mathbf{e} \in \Lambda^{\perp}(\mathsf{F})$ given the trapdoor T_B, we compute $\mathbf{d}_1 \leftarrow \mathsf{SampleDom}(1^{(m)}, (\eta\,\sqrt{\ell}-1)\,\sigma')$ and $\mathbf{d}_2 \leftarrow \mathsf{SamplePre}(\mathsf{B}, q, \mathsf{T}_B, \sigma', -\mathsf{A}\,\mathbf{d}_1)$, and define,

$$\mathbf{d} \;=\; \begin{bmatrix}\mathbf{d}_1\\ \mathbf{d}_2\end{bmatrix} \;\in\; \mathbb{Z}^{2m} \qquad\qquad \mathbf{e} \;=\; \begin{bmatrix}\mathbf{d}_1 - \mathsf{R}\,\mathbf{d}_2\\ \mathbf{d}_2\end{bmatrix} \;\in\; \mathbb{Z}^{2m}$$

Observe that $\mathbf{e}$ is a fixed invertible linear function of $\mathbf{d}$, and that $\mathbf{d}$ is discrete Gaussian by construction. A result of Regev [23] shows that, with overwhelming probability, at most $4\,m^2$ samples will be needed to get $2\,m$ linearly independent vectors "$\mathbf{d}$", and therefore also $2\,m$ linearly independent vectors "$\mathbf{e}$". For each $\mathbf{e}$, we have $\mathsf{F}\,\mathbf{e} = \mathsf{A}\,(\mathbf{d}_1 - \mathsf{R}\,\mathbf{d}_2) + (\mathsf{A}\,\mathsf{R} + \mathsf{B})\,\mathbf{d}_2 = \mathsf{A}\,\mathbf{d}_1 + \mathsf{B}\,\mathbf{d}_2 = \mathsf{A}\,\mathbf{d}_1 - \mathsf{A}\,\mathbf{d}_1 = \mathbf{0} \in \mathbb{Z}_q^n$, hence $\mathbf{e} \in \Lambda^{\perp}(\mathsf{F})$. We have also $\|\mathbf{e}\| \le (\eta\,\sqrt{\ell} - 1)\,\sigma'\,\sqrt{m} + \eta\,\sigma'\,m\,\sqrt{\ell} + \sigma'\,\sqrt{m} \le 2\,\eta\,\sigma'\,m\,\sqrt{\ell}$. Thus by assembling $2\,m$ linearly independent such vectors "$\mathbf{e}$", we obtain a full-rank set $\mathsf{S}_F \subset \Lambda^{\perp}(\mathsf{F})$ of orthogonalized norm $\|\tilde{\mathsf{S}}_F\| \le 2\,\eta\,\sigma'\,m\,\sqrt{\ell}$.

2. We convert the short set S_F into an equally short basis T_F, i.e., such that $\|\tilde{\mathsf{T}}_F\| \le \|\tilde{\mathsf{S}}_F\|$. We can do this efficiently using the algorithm of Fact 3, starting from an arbitrary basis for $\Lambda^{\perp}(\mathsf{F})$, itself easy to construct by linear algebra.

3. We use the newly constructed basis T_F to sample a short preimage $\mathbf{e}$ of the given target $\mathbf{u} \in \mathbb{Z}_q^n$, using $\mathbf{e} \leftarrow \mathsf{SamplePre}(\mathsf{F}, q, \mathsf{T}_F, \sigma, \mathbf{u})$. Notice that the Gaussian parameter $\sigma \ge \|\tilde{\mathsf{T}}_F\|\,\omega(\sqrt{\log m})$, so the algorithm $\mathsf{SamplePre}$ can be applied with the stated parameters, and hence $\mathbf{e}$ sampled in this manner will have conditional distribution $\mathbf{e} \sim \mathcal{D}_{\mathbb{Z}^{2m},\sigma} \mid \mathsf{F}\,\mathbf{e} = \mathbf{u} \pmod q$. $\square$

Remark 24. Agrawal et al. [2] show the sampling overhead is only a factor ≤ 2, hence in Step 1 we need to sample at most $4\,m$ vectors "$\mathbf{e}$" on expectation.

We also mention that a lower-norm fickle trapdoor may be obtained by using the Alwen-Peikert delegation method as in Lemma 22 instead of the repeated sampling as above. We shall present it in the full version.

The point of the two-sided preimage-samplable function is that in the actual scheme we use the "firm" preimage mechanism with an always-available trapdoor T_A, whereas in the simulation we use the "fickle" preimage mechanism T_B for a matrix $\mathsf{B} = \mathsf{h}_{\mathsf{msg}}\,\mathsf{B}_0$ that sometimes vanishes.

3.2 Main Signature Scheme

The following is our core construction of a fully secure short signature. It is very simple and already achieves most of the compactness benefits while illustrating the framework. In the full version, we show how to squeeze out some additional factor from the signature size, albeit at the cost of a more complex system.

From now on, a message msg is an ℓ-bit string $\big(\mathsf{msg}[1], \ldots, \mathsf{msg}[\ell]\big) \in \{0,1\}^{\ell}$ indexed from 1 to ℓ, augmented with a 0-th dummy extra bit set to $\mathsf{msg}[0] = 0$. This will let us easily include a constant term of index 0 in various summations.

KeyGen(1^{λ})**:** On input a security parameter λ in unary, do these steps:

1. Draw an n-by-m matrix $\mathsf{A}_0 \in \mathbb{Z}_q^{n\times m}$ with a short basis $\mathsf{T}_{A_0} \subset \Lambda^{\perp}(\mathsf{A}_0)$.

– Do so by invoking $\mathsf{TrapGen}(1^\lambda)$, resulting in T_{A_0} such that $\|\tilde{\mathsf{T}}_{A_0}\| \leq L$.
2. Draw $\ell+1$ independent n-by-m-matrices $\mathsf{C}_0, \ldots, \mathsf{C}_\ell \in \mathbb{Z}_q^{n\times m}$.
3. Output the signing and verification keys,

$$\mathsf{SK} \;=\; \big(\mathsf{T}_{A_0}\big) \;\in\; \mathbb{Z}^{m\times m} \qquad\qquad \mathsf{VK} \;=\; \big(\mathsf{A}_0,\; \mathsf{C}_0, \ldots, \mathsf{C}_\ell\big) \;\in\; (\mathbb{Z}_q^{n\times m})^{\ell+2}$$

Sign$(\mathsf{SK}, \mathsf{msg})$**:** On input a signing key SK and a message $\mathsf{msg} \in \{0\} \times \{0,1\}^\ell$:

1. Define the n-by-m-matrix $\mathsf{C}_{\mathsf{msg}} = \sum_{i=0}^{\ell} (-1)^{\mathsf{msg}[i]}\, \mathsf{C}_i$.
2. Define the message-dependent matrix $\mathsf{F}_{\mathsf{msg}} = \big[\mathsf{A}_0 \;\big|\; \mathsf{C}_{\mathsf{msg}}\big] \in \mathbb{Z}_q^{n\times 2m}$.
3. Sample a short non-zero random point $\mathbf{d} \in \Lambda^\perp(\mathsf{F}_{\mathsf{msg}})$, using $\mathsf{SK} = \mathsf{T}_{A_0}$.
– Do so by sampling $\mathbf{d} \sim \mathcal{D}_{\mathbb{Z}^{2m},\sigma} \mid \mathsf{F}_{\mathsf{msg}}\, \mathbf{d} = 0$, using Lemma 22.
4. Output the digital signature,

$$\mathsf{sig}_{\mathsf{msg}} \;=\; \big(\mathbf{d}\big) \quad\in\; \mathbb{Z}^{2m}$$

Verify$(\mathsf{VK}, \mathsf{msg}, \mathsf{sig}_{\mathsf{msg}})$**:** On input a verification key VK, a message msg, and a signature $\mathsf{sig}_{\mathsf{msg}}$:

1. Check that the message msg is well formed in $\{0\} \times \{0,1\}^\ell$.
2. Check that the signature $\mathsf{sig}_{\mathsf{msg}}$ is a small but non-zero vector.
– Do so by verifying that $\mathsf{sig}_{\mathsf{msg}} = \mathbf{d} \in \mathbb{Z}^{2m}$ and $0 < \|\mathbf{d}\| \leq \sqrt{2\,m} \cdot \sigma$.
3. Check that $\mathsf{sig}_{\mathsf{msg}}$ is a point on the “mixed” lattice specified by msg.
– Do so by verifying that

$$\Big[\mathsf{A}_0 \;\Big|\; \sum_{i=0}^{\ell} (-1)^{\mathsf{msg}[i]}\, \mathsf{C}_i\Big]\, \mathbf{d} \;=\; \mathbf{0} \pmod q$$

4. If all the verifications pass, accept the signature; otherwise, reject.

3.3 Security Reduction

It is easy to see by inspection that the signature scheme is consistent with overwhelming probability.

The next theorem reduces the SIS problem to the existential forgery of our signature. The proof involves a moderate polynomial SIS parameter β. The expression of β arises in Lemma 26, but otherwise “passes through” the reduction. In § 3.4, we revisit the question of the lattice parameters in greater detail.

Theorem 25. *For a prime modulus $q = q(\lambda)$, if there is a probabilistic algorithm $\mathcal{A}$ that outputs an existential signature forgery, with probability ϵ, in time τ, and making $Q \leq q/2$ adaptive chosen-message queries, then there is a probabilistic algorithm $\mathcal{B}$ that solves the (q, n, m, β)-SIS problem in time $\tau' \approx \tau$ and with probability $\epsilon' \geq \epsilon/(3\,q)$, for some polynomial function $\beta = \mathrm{poly}(\lambda)$.*

Proof. Suppose that there exists such a forger $\mathcal{A}$. We construct a solver $\mathcal{B}$ that simulates an attack environment and uses the forgery to create its solution. The various operations performed by $\mathcal{B}$ are the following.

Invocation. $\mathcal{B}$ is invoked on a random instance of the (q, n, m, β)-SIS problem, and is asked to return an admissible solution.

- Supplied: an n-by-m-matrix $\mathsf{A}_0 \in \mathbb{Z}_q^{n\times m}$ from the uniform distribution.
- Requested: any $\mathbf{e}_0 \in \mathbb{Z}^m$ such that $\mathsf{A}_0\,\mathbf{e}_0 = 0 \pmod q$ and $0 \neq \|\mathbf{e}_0\| \leq \beta$.

Setup. $\mathcal{B}$ gives to the adversary $\mathcal{A}$ a simulated verification key constructed as follows:

1. Pick a random matrix $\mathsf{B}_0 \in \mathbb{Z}_q^{n\times m}$ with a short basis $\mathsf{T}_{B_0} \subset \Lambda^\perp(\mathsf{B}_0)$.
   - Do so by invoking $\mathsf{TrapGen}(1^\lambda)$, resulting in T_{B_0} such that $\|\tilde{\mathsf{T}}_{B_0}\| \leq L$.
2. Pick $\ell + 1$ short random square m-by-m-matrices $\mathsf{R}_0, \ldots, \mathsf{R}_\ell \in \mathbb{Z}^{m\times m}$.
   - Do so by independently sampling the columns of the $\mathsf{R}_i \sim \mathcal{D}_{\mathbb{Z}^m,\eta}$.
3. Pick ℓ uniformly random scalars $\mathsf{h}_1, \ldots, \mathsf{h}_\ell \in \mathbb{Z}_q$ and fix $\mathsf{h}_0 = 1 \in \mathbb{Z}_q$.
4. Output the verification key $\mathsf{VK} = \big(\ \mathsf{A}_0,\ \ \mathsf{C}_0 = (\mathsf{A}_0\,\mathsf{R}_0 + \mathsf{h}_0\,\mathsf{B}_0) \bmod q,$ $\mathsf{C}_1 = (\mathsf{A}_0\,\mathsf{R}_1 + \mathsf{h}_1\,\mathsf{B}_0) \bmod q,\ \ldots,\ \mathsf{C}_\ell = (\mathsf{A}_0\,\mathsf{R}_\ell + \mathsf{h}_\ell\,\mathsf{B}_0) \bmod q\ \big)$.

Queries. $\mathcal{B}$ answers adaptive signature queries from $\mathcal{A}$ on any message msg as follows:

1. Compute the matrix $\mathsf{R}_{\mathsf{msg}} = \sum_{i=0}^{\ell} (-1)^{\mathsf{msg}[i]}\,\mathsf{R}_i$.
2. Compute the scalar $\mathsf{h}_{\mathsf{msg}} = \sum_{i=0}^{\ell} (-1)^{\mathsf{msg}[i]}\,\mathsf{h}_i$.
3. If $\mathsf{h}_{\mathsf{msg}} = 0 \pmod q$, abort the simulation.
4. Compute the matrix $\mathsf{F}_{\mathsf{msg}} = \big[\mathsf{A}_0 \ \big|\ \mathsf{A}_0\,\mathsf{R}_{\mathsf{msg}} + \mathsf{h}_{\mathsf{msg}}\,\mathsf{B}_0\big] \in \mathbb{Z}_q^{n\times 2m}$.
5. Find a short random $\mathbf{d} \in \Lambda^\perp(\mathsf{F}_{\mathsf{msg}}) \subset \mathbb{Z}^{2m}$, using the trapdoor T_{B_0}.
   - Do so by sampling $\mathbf{d} \sim \mathcal{D}_{\mathbb{Z}^{2m},\sigma}$ given $\mathsf{F}_{\mathsf{msg}}\,\mathbf{d} = 0$, using the procedure of Lemma 23, using T_{B_0} as short basis for $\Lambda^\perp(\mathsf{h}_{\mathsf{msg}}\,\mathsf{B}_0)$ per Lemma 19.
6. Output the digital signature $\mathsf{sig}_{\mathsf{msg}} = \mathbf{d} \in \mathbb{Z}^{2m}$.

Forgery. $\mathcal{B}$ receives from $\mathcal{A}$ a forged signature $\mathbf{d}^*$ on a new (unqueried) message msg^*, and does:

1. Compute the matrix $\mathsf{R}^* = \sum_{i=0}^{\ell} (-1)^{\mathsf{msg}^*[i]}\,\mathsf{R}_i$.
2. Compute the scalar $\mathsf{h}^* = \sum_{i=0}^{\ell} (-1)^{\mathsf{msg}^*[i]}\,\mathsf{h}_i$.
3. If $\mathsf{h}^* \neq 0 \pmod q$, abort the simulation.
4. Separate $\mathbf{d}^{*T}$ into $\big[\mathbf{d}_1^{*T} \ \big|\ \mathbf{d}_2^{*T}\big]$.
5. Return $\mathbf{e}_0 = \mathbf{d}_1^* + \mathsf{R}^*\,\mathbf{d}_2^* \in \mathbb{Z}^m$ as solution to $\mathsf{A}_0\,\mathbf{e}_0 = \mathbf{0} \pmod q$.

Lemma 26 shows that the answer $\mathbf{e}_0$ will be with small and non-zero with good probability, and thus a valid (q, n, m, β)-SIS solution for the stated approximation β. (An instantiation of β is given in § 3.4.)

Outcome. The reduction is valid provided that $\mathcal{B}$ can complete the simulation (without aborting) with a substantial probability that is independent of the view of $\mathcal{A}$ and the choices it makes. The completion probability for $\mathcal{B}$ against an arbitrary strategy for $\mathcal{A}$ is quantified in Lemma 27.

It follows from the bounds of Lemmas 26 and 27, under the assumption that $Q \leq q/2$, that if $\mathcal{A}$ existentially forges a signature with probability ϵ, then $\mathcal{B}$ solves the SIS instance with probability,

$$\epsilon' \ \geq\ \pi_0\,\big(1 - q^{-1}\,Q\big)\,q^{-1}\,\epsilon \ \geq\ \pi_0\,\epsilon/2\,q \ \geq\ \epsilon/3\,q \qquad \text{for}\ \ \pi_0 \geq 2/3$$

With the stated lemmas, this concludes the security reduction. □

Lemma 26. *Given a valid forgery* $\left[\boldsymbol{d}_1^{*T} \mid \boldsymbol{d}_2^{*T}\right]$ *from* $\mathcal{A}$ *on some* msg^* *such that* $\mathsf{h}_{\mathsf{msg}^*} = 0 \pmod q$*, the vector* $\boldsymbol{e}_0 = \boldsymbol{d}_1^* + \mathsf{R}_{\mathsf{msg}^*}\, \boldsymbol{d}_2^* \in \mathbb{Z}^m$ *is with high probability* $\pi_0 = \Theta(1) \geq 2/3$ *a short non-zero preimage of* 0 *under* A_0*, namely,* $\boldsymbol{e}_0 \in \Lambda^\perp(\mathsf{A}_0)$ *and* $0 \neq \|\boldsymbol{e}_0\| \leq \beta$ *for some polynomial function* $\beta = \mathrm{poly}(\ell, n, m) = \mathrm{poly}(\lambda)$.

Sketch. Let $\mathsf{h}^* = \mathsf{h}_{\mathsf{msg}^*}$ and $\mathsf{R}^* = \mathsf{R}_{\mathsf{msg}^*}$. Let $\mathsf{C}^* = \mathsf{C}_{\mathsf{msg}^*} = \sum_{i=0}^{\ell} (-1)^{\mathsf{msg}^*[i]}\, \mathsf{C}_i$. First, when $\mathsf{h}^* = 0$, we have $\mathsf{C}^* = \mathsf{A}_0\, \mathsf{R}^* + \mathsf{h}^*\, \mathsf{B}_0 = \mathsf{A}_0\, \mathsf{R}^*$, and thus for a valid signature forgery $\mathbf{d}^*$,

$$\mathsf{A}_0\, \mathbf{e}_0 \;=\; \mathsf{A}_0 \left(\mathbf{d}_1^* + \mathsf{R}^*\, \mathbf{d}_2^*\right) \;=\; \left[\mathsf{A}_0 \mid \mathsf{A}_0\, \mathsf{R}^*\right] \begin{bmatrix}\mathbf{d}_1^* \\ \mathbf{d}_2^*\end{bmatrix} \;=\; \left[\mathsf{A}_0 \mid \mathsf{C}^*\right] \mathbf{d}^* \;=\; 0 \pmod q$$

Next, we show that $\mathbf{e}_0$ is suitably short, which is true since R^* is a sum of $\ell + 1$ low-norm matrices R_i with coefficients ± 1, where the summands are all short discrete Gaussian by construction of $\mathsf{R}_0, \ldots, \mathsf{R}_\ell$. Since the matrices $\pm\mathsf{R}_i$ are nearly independent with the same variance $\mathrm{V}\{\pm\mathsf{R}_i\} = \mathrm{V}\{\mathsf{R}_1\}$, we have,

$$\mathrm{V}\{\mathsf{R}^*\} \;=\; \mathrm{V}\{\sum_{i=0}^{\ell} \pm\mathsf{R}_i\} \;\approx\; \sum_{i=0}^{\ell} \mathrm{V}\{\pm\mathsf{R}_i\} \;=\; \sum_{i=0}^{\ell} \mathrm{V}\{\mathsf{R}_i\} \;=\; (\ell + 1) \cdot \mathrm{V}\{\mathsf{R}_1\}$$

Since the $\pm\mathsf{R}_i$ closely approximate real normal Gaussian variables, so does R^* and therefore the Gaussian "vanishing tail" inequalities apply. Especially, as they are almost independent discrete Gaussian with center 0 and parameter η, and thus $\mathrm{E}\{\mathsf{R}^*\} \approx \mathrm{E}\{\mathsf{R}_i\} = 0$, we have $\Pr\{\| \pm \mathsf{R}_i\| > \sqrt{m}\,\eta\} = \mathrm{negl}(m)$; and thus,[1]

$$\Pr\{\|\mathsf{R}^*\| > \sqrt{\ell + 1} \cdot \sqrt{m}\,\eta\} \;\leq\; \Pr\{\|\mathsf{R}_1\| > \sqrt{m}\,\eta\} \;=\; \mathrm{negl}(m)$$

Hence with overwhelming probability $\|\mathbf{e}_0\| \leq \beta$ for $\beta = \mathrm{poly}(\ell, n, m) = \mathrm{poly}(\lambda)$, provided we set,[1]

$$\beta \;=\; \left(1 + \sqrt{\ell + 1}\, \sqrt{m}\,\eta\right) \sqrt{2\, m}\, \sigma$$

Finally, it remains to show that $\mathbf{e}_0 = \mathbf{d}_1^* + \mathsf{R}_{\mathsf{msg}^*}\, \mathbf{d}_2^* \neq \mathbf{0}$. Suppose for an easy case that $\mathbf{d}_2^* = \mathbf{0}$; then for a valid forgery we must have $\mathbf{d}_1^* \neq \mathbf{0}$ and thus $\mathbf{e}_0 \neq \mathbf{0}$. Suppose on the contrary that $\mathbf{d}_2^* \neq \mathbf{0}$. In that case, $0 \neq \|\mathbf{d}_2^*\| < \sqrt{2\, m}\, \sigma \ll q$; and thus there must be at least one coordinate of $\mathbf{d}_2^*$ that is non-zero modulo q. W.l.o.g., let this coordinate be the last one in $\mathbf{d}_2^*$, and call it y. Let $\mathbf{r}^*$ be the last column of R^*, and let $\mathbf{r}_i$ the last column of R_i for each i. As $\mathsf{R}^* = \sum (-1)^{\mathsf{msg}[i]} \mathsf{R}_i$, we have $\mathbf{r}^* = \sum (-1)^{\mathsf{msg}[i]} \mathbf{r}_i$, where the coefficients ± 1 depend on the message bits. We focus on $\mathbf{r}_1$: the last column of the matrix R_1 associated with the first message bit $\mathsf{msg}[1]$. Let $\mathbf{v} = (-1)^{\mathsf{msg}[1]}\, \mathsf{y}\, \mathbf{r}_1$. The expression of $\mathbf{e}_0$ can be rewritten $\mathbf{e}_0 = \mathsf{y}\, \mathbf{r}^* + \mathbf{e}_0' = \mathbf{v} + \mathbf{e}_0''$, where $\mathbf{v}$ depends on $\mathbf{r}_1$ and $\mathbf{e}_0''$ does not.

The last step is to observe that the only information about $\mathbf{r}_1$ available to $\mathcal{A}$ is contained in the last column of C_1 (with "pollution" $\mathsf{h}_1\, \mathsf{B}_0$, known in the worst case). By leftover hash or a simple pigeonhole principle, there are a very large

[1] Without using any independence, we can show $\Pr\{\|\mathsf{R}^*\| > (\ell + 1) \cdot \sqrt{m}\,\eta\} = \mathrm{negl}(m)$, and accordingly set $\beta = \left(1 + (\ell + 1)\, \sqrt{m}\,\eta\right) \sqrt{2\, m}\, \sigma$, which is a factor $\approx \sqrt{\ell}$ worse.

(exponential in $m - n \log q$) number of admissible and equally likely vectors $\mathbf{r}_1$ that are compatible with the view of $\mathcal{A}$, and in particular more than six of them. Since $\mathcal{A}$ can set the bit $\mathsf{msg}[1]$ in one of two ways, it follows that $\mathcal{A}$ cannot know the value of $\mathbf{v}$ with probability exceeding one third. At most one such value can result in a cancellation of $\mathbf{e}_0$, for if some $\mathbf{v}$ caused all coordinates of $\mathbf{e}_0$ to cancel, then every other $\mathbf{v}$ would fail to do so. We deduce that $\pi_0 = \Pr\{\mathbf{e}_0 \neq 0\} \geq 2/3$. (In fact, we have $\pi_0 > 1 - \exp(-\Omega(m - n \log q)) \to 1$ as $\lambda \to \infty$.) $\square$

Lemma 27. *For a prime modulus $q = q(\lambda)$ and a number of queries $Q \geq 0$, the simulation completes both the Queries and Forgery phases without aborting, with probability,*

$$\frac{1}{q}\left(1 - \frac{Q}{q}\right) \;\leq\; \Pr\{\mathit{completion}\} \;\leq\; \frac{1}{q}$$

In particular, for $Q \leq q/2$, this probability is $\Pr\{\mathit{completion}\} \in \left[q^{-1}/2,\ q^{-1}\right]$ *regardless of the adversary's strategy.*

Intuitively, we first observe that *provided* $\mathcal{B}$ does not abort, then the simulation is (almost) perfect in the sense that the view of $\mathcal{A}$ has the same distribution as in an attack against the real scheme (modulo a negligible sampling error owing to the imperfection of $\mathsf{TrapGen}$). In particular, $\mathcal{A}$'s view remains independent of $\mathcal{B}$'s choice of $\mathsf{h}_1, \ldots, \mathsf{h}_\ell$, simply because those values have no counterpart in an actual attack environment.

Now, the adversary can always *assume* that it is facing a simulator instead of a real challenger, and accordingly attempt to derail the simulation. Since the necessity to abort, for a given adversarial strategy, hinges entirely on $\mathcal{B}$'s secret choice of random $\mathsf{h}_1, \ldots, \mathsf{h}_\ell$, it suffices to show that these values remain mostly unlearnable no matter $\mathcal{A}$'s attack strategy.

To show this, we consider a hypothetical unbounded perfect adversary $\mathcal{A}$ and show that, even with perfect Bayesian updating upon each new adaptive query it makes, such adversary is unable to infer enough information about $\mathsf{h}_1, \ldots, \mathsf{h}_\ell$ to affect significantly the success probability of the simulation.

Proof. Consider the ℓ-dimensional space $\mathbb{Z}_q^\ell$, which is the domain of the unknown $(\mathsf{h}_1, \ldots, \mathsf{h}_\ell)$, and recall that $\mathsf{h}_0 = 1$. Denote by $\mathcal{H}_j$ the distribution of $(\mathsf{h}_1, \ldots, \mathsf{h}_\ell)$ over $\mathbb{Z}_q^\ell$ as perceived by the adversary after the first j signature queries have been answered without aborting.

At the start of the attack, since the simulator's selection of $(\mathsf{h}_1, \ldots, \mathsf{h}_\ell)$ is a uniformly random point in $\mathbb{Z}_q^\ell$, the adversary's prior distribution $\mathcal{H}_0$ is necessarily the uniform distribution $\mathcal{U}(\mathbb{Z}_q^\ell)$ over $\mathbb{Z}_q^\ell$. For every query message msg_j that is answered without aborting, $\mathcal{A}$ can prune from the support of $\mathcal{H}$ every point $(\mathsf{h}_1, \ldots, \mathsf{h}_\ell)$ that lies on the "incompatible" hyperplane $\mathsf{h}_{\mathsf{msg}_j} = 0 \pmod q$.

Denote by $\mathbb{V}_j$ the hyperplane thus eliminated after a successful j-th query. Suppose by induction that $\mathcal{H}_{j-1} = \mathcal{U}(\mathbb{W})$, a uniform distribution over some support set $\mathbb{W} \subseteq \mathbb{Z}_q^\ell$. By conditioning $\mathcal{H}_{j-1}$ on the new evidence gained at the j-th query, namely that $(\mathsf{h}_1, \ldots, \mathsf{h}_\ell) \notin \mathbb{V}_j$, one obtains an updated or posterior distribution $\mathcal{H}_j = \mathcal{U}(\mathbb{W}\setminus\mathbb{V}_j)$, which is uniform over the smaller support set given by $\mathbb{W} \setminus \mathbb{V}_j$. By induction on the number of queries, starting from $\mathcal{H}_0 = \mathcal{U}(\mathbb{Z}_q^\ell)$, we deduce that, after the j-th query, $\mathcal{H}_j = \mathcal{U}(\mathbb{Z}_q^\ell \setminus \cup_{i=1}^j \mathbb{V}_i)$.

In particular, after all Q allowed queries have been made, the fully updated posterior distribution $\mathcal{H}_Q$ in the view of the adversary is then,

$$\mathcal{H}_Q \;=\; \mathcal{U}\big(\mathbb{Z}_q^\ell \setminus \cup_{i=1}^Q \mathbb{V}_i\big)$$

In other words, this shows that, in the event that $\mathcal{B}$ was able to answer all the queries, the unknown vector $(\mathsf{h}_1,\ldots,\mathsf{h}_\ell)$ remains equally likely to lie anywhere in all of $\mathbb{Z}_q^\ell$ outside of the Q query-dependent hyperplanes $\mathbb{V}_1,\ldots,\mathbb{V}_Q$. Being the result of perfect Bayesian updating from all available observations, this distribution captures all the information about $(\mathsf{h}_1,\ldots,\mathsf{h}_\ell)$ leaked by $\mathcal{B}$ to $\mathcal{A}$ during the Queries phase.

To complete the argument, consider the hyperplane $\mathbb{V}^* \subset \mathbb{Z}_q^\ell$ defined by the scalar equation $\mathsf{h}_{\mathsf{msg}^*} = 0 \pmod q$ corresponding to the forgery message msg^* chosen by the adversary. By the requirements of what constitutes a valid existential forgery, we know that $\mathsf{msg}^* \neq \mathsf{msg}_j$ and thus $\mathbb{V}^* \neq \mathbb{V}_j$ for all j. (Indeed, the purpose of adding a fixed dummy message bit $\mathsf{msg}[0]$ and setting $\mathsf{h}_0 = 1 \neq 0$ is to ensure that any two distinct messages $\mathsf{msg}_j \neq \mathsf{msg}^* \in \{0,1\}^\ell$ always induce distinct hyperplaces $\mathbb{V}_j \neq \mathbb{V}^* \subset \mathbb{Z}_q^\ell$.)

Since $\mathbb{V}^*$ and $\mathbb{V}_j$ are distinct affine subspaces of dimension $\ell-1$ in $\mathbb{Z}_q^\ell$, we have $\big|\mathbb{V}^* \cap \mathbb{V}_j\big| \leq q^{\ell-2}$ whereas $\big|\mathbb{V}^*\big| = \big|\mathbb{V}_j\big| = q^{\ell-1}$ and of course $\big|\mathbb{Z}_q^\ell\big| = q^\ell$. Consequently, $\mathbb{V}^*$ and $\mathbb{V}_j$ have at most a fraction $1/q$ of their points in common, and more specifically $\big|\mathbb{V}^* \setminus \mathbb{V}_j\big| \geq (1-q^{-1})\,\big|\mathbb{V}^*\big| = (1-q^{-1})\,q^{-1}\,\big|\mathbb{Z}_q^\ell\big|$ for all j.

Considering the event *completion* $= \wedge_{i=1}^Q \big\{(\mathsf{h}_1,\ldots,\mathsf{h}_\ell) \in (\mathbb{V}^* \setminus \mathbb{V}_i)\big\}$ and invoking the union bound on this conjunction, we thus establish a lower bound,

$$\begin{aligned}\Pr\big\{\mathit{completion}\big\} \;&=\; \Pr\big\{(\mathsf{h}_1,\ldots,\mathsf{h}_\ell) \in (\mathbb{V}^* \setminus \cup_{i=1}^Q \mathbb{V}_i)\big\}\\ &\geq\; \big(1-q^{-1}\,Q\big)\;\Pr\big\{(\mathsf{h}_1,\ldots,\mathsf{h}_\ell) \in \mathbb{V}^*\big\} \;=\; \big(1-q^{-1}\,Q\big)\,q^{-1}\end{aligned}$$

Conversely, we can trivially establish an upper bound,

$$\begin{aligned}\Pr\big\{\mathit{completion}\big\} \;&=\; \Pr\big\{(\mathsf{h}_1,\ldots,\mathsf{h}_\ell) \in (\mathbb{V}^* \setminus \cup_{i=1}^Q \mathbb{V}_i)\big\}\\ &\leq\; \Pr\big\{(\mathsf{h}_1,\ldots,\mathsf{h}_\ell) \in \mathbb{V}^*\big\} \;=\; \big|\mathbb{V}^*\big| \,/\, \big|\mathbb{Z}_q^\ell\big| = q^{-1}\end{aligned}$$

In both cases the probability is over the simulator's initial choice of $\mathsf{h}_1,\ldots,\mathsf{h}_\ell$.

We have shown that the probability of *completion* without aborting is bounded in the narrow range $\big[(1-q^{-1}\,Q)\,q^{-1},\; q^{-1}\big]$, regardless of the adversary's actions. The lemma follows. □

3.4 Lattice Parameters

It is not so obvious to see that the various parameters can be instantiated in a way that satisfies the flurry of constraints and inequalities evoked in § 2 and § 3.3. This is necessary for us, later, to prove the security of the signature from a polynomial average-case SIS that reduces to a worst-case lattice hardness assumption.

Example 28. To ensure that hard lattices with good short bases can be generated (i.e., $m \geq 6\,n \log q$), that our flavor of SIS has a worst-case lattice reduction (i.e., $q \geq \beta \cdot \omega(\sqrt{n \log n})$), that the two-sided trapdoors can operate smooothly (i.e., σ sufficiently large), that vectors samples using a trapdoor are difficult SIS solutions (i.e., $\beta \geq \sqrt{2\,\ell\, m\, \eta\, \sigma}$), etc., in function of a security parameter λ, we may choose a function $\omega(\sqrt{\log m})$, a constant $\delta_1 > 0$, and a threshold $\lambda_0 \gg 0$; and $\forall \lambda > \lambda_0$ we set:

$$n = \lambda$$
$$m = n^{1+\delta_1}$$
$$\eta = \omega(\sqrt{\log m})$$
$$L = \sqrt{m}\, \omega(\sqrt{\log m})$$
$$\sigma = \sqrt{\ell}\, m^{3/2}\, \omega(\sqrt{\log m})^4$$
$$\beta = \sqrt{2}\,\ell\, m^{5/2}\, \omega(\sqrt{\log m})^5$$
$$q = \sqrt{2}\,\ell\, m^{3}\, \omega(\sqrt{\log m})^6$$

One must however keep in mind that the security reduction given in Theorem 25 holds only if $q \geq 2\,Q$, so it may be necessary to increase q and the other parameters beyond the baseline values listed above. We avoid this in § 3.5.

3.5 Refined Simulation Framework

In the full version, we give a refined analysis of the scheme that lets us keep the baseline q even for very large Q. The idea is to replace the random scalars $\mathsf{h}_i \in \mathbb{Z}_q$ by block-diagonal matrices $\mathsf{H}_i \in \mathbb{Z}_q^{n\times n}$ consisting of a repeated random submatrix drawn from a *full-rank difference* group $\mathcal{G} \subset \mathbb{Z}_q^{k\times k}$ for a special $k|n$, where any difference $G_1 - G_2 \in \mathcal{G}$ is either zero or an invertible matrix in $\mathbb{Z}_q^{k\times k}$. Visually, a random input k-vector $\in \mathbb{Z}_q^k$ is mapped to a random matrix H_i using an encoding map μ built from an FRD encoding φ, according to this picture,

$$\mu \;:\; \mathbb{Z}_q^k \to \mathbb{Z}_q^{n\times n} \;:\; \mathbf{v} \mapsto \begin{pmatrix} \boxed{\varphi(\mathbf{v})} & & & 0 \\ & \boxed{\varphi(\mathbf{v})} & & \\ & & \ddots & \\ 0 & & & \boxed{\varphi(\mathbf{v})} \end{pmatrix}$$

Full-rank difference (FRD) families in $\mathbb{Z}_q^{n\times n}$ were used as a plentiful IBE encoding [1] able to represent as many as possible, up to q^n, distinct identities.

Here, FRD families will serve in a very different way, internal to the simulator, to turn the mixing coefficients h_i into *uniformly* drawn matrices H_i from a domain whose size q^k is *just right* in function of the number Q of queries, without worrying about the modulus q. The benefit is that smaller moduli makes signatures smaller and faster, and security tighter. Remarkably, except for relaxing q, the actual scheme is unchanged. The theorem below is proven in the full paper.

Theorem 29. *If there exists a probabilistic algorithm $\mathcal{A}$ that creates an existential signature forgery, in time τ, with probability ϵ, making Q adaptive chosen-message queries, then there exists a probabilistic algorithm $\mathcal{B}$ that solves the SIS problem of Theorem 25 in time $\tau' \approx \tau$ with probability $\epsilon' \geq \epsilon/(6\,q\,Q)$.*

Since both theorems apply to the same scheme in our framework, we can pick q obliviously of Q, and invoke Theorem 25 if $Q \leq q/2$ or Theorem 29 if $Q \gg q$.

Acknowledgments. The author thanks Shweta Agrawal, Dan Boneh, Ronald Cramer, David Freeman, and anonymous referees for valuable insights.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: EUROCRYPT (2010)
2. Agrawal, S., Boneh, D., Boyen, X.: Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE (2010) (manuscript)
3. Agrawal, S., Boyen, X.: Identity-based encryption from lattices in the standard model (2009) (manuscript), http://www.cs.stanford.edu/~xb/ab09/
4. Aharonov, D., Regev, O.: Lattice problems in NP ∩ coNP. Journal of the ACM 52(5), 749–765 (2005)
5. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: STOC (1996)
6. 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, p. 1. Springer, Heidelberg (1999)
7. Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: STACS (2009)
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., Boyen, X.: Efficient lattice (H)IBE in the standard model from the BB-1 framework (2009) (manuscript), http://rump2009.cr.yp.to/
10. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656. Springer, Heidelberg (2003)
11. Cash, D., Hofheinz, D., Kiltz, E.: How to delegate a lattice basis. Cryptology ePrint Archive, Report 2009/351 (2009), http://eprint.iacr.org/
12. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
13. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC (2008)
14. 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)
15. Lovász, L.: An Algorthmic Theory of Numbers, Graphs and Convexity. SIAM, Philadelphia (1986)
16. Lyubashevsky, V., Micciancio, D.: Asymptotically efficient lattice-based digital signatures. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 37–54. Springer, Heidelberg (2008)

17. Micciancio, D., Goldwasser, S.: Complexity of lattice problems: a cryptographic perspective. Kluwer Series on Engineering and Computer Science (2002)
18. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: FOCS (2004)
19. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. SIAM Journal of Computing 37(1), 267–302 (2007)
20. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Bernstein, D.J., Buchmann, J. (eds.) Post-quantum Cryptography. Springer, Heidelberg (2008)
21. Peikert, C.: Bonsai trees (or, arboriculture in lattice-based cryptography). Cryptology ePrint Archive, Report 2009/359 (2009), http://eprint.iacr.org/
22. 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)
23. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC (2005)
24. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

Author Index