Tsuyoshi Takagi Guilin Wang
Zhiguang Qin Shaoquan Jiang
Yong Yu (Eds.)

LNCS 7496

Provable Security

6th International Conference, ProvSec 2012
Chengdu, China, September 2012
Proceedings

Springer

Lecture Notes in Computer Science 7496

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

Tsuyoshi Takagi Guilin Wang
Zhiguang Qin Shaoquan Jiang Yong Yu (Eds.)

Provable Security

6th International Conference, ProvSec 2012
Chengdu, China, September 26-28, 2012
Proceedings

Springer

Volume Editors

Tsuyoshi Takagi
Kyushu University
Institute of Mathematics for Industry
744, Motooka, Nishi-ku, Fukuoka 819-0395, Japan
E-mail: takagi@imi.kyushu-u.ac.jp

Guilin Wang
University of Wollongong
School of Computer Science and Software Engineering
Northfields Avenue, Wollongong NSW 2522, Australia
E-mail: guilin@uow.edu.au

Zhiguang Qin
Shaoquan Jiang
Yong Yu
University of Electronic Science and Technology of China
School of Computer Science and Engineering
2006 Xiyuan Rd, HighTech West, Chengdu, 611731, China
E-mail: {qinzg, yuyong}@uestc.edu.cn, shaoquan.jiang@gmail.com

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-33271-5 e-ISBN 978-3-642-33272-2
DOI 10.1007/978-3-642-33272-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012946106

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

LNCS Sublibrary: SL 4 – Security and Cryptology

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

Printed on acid-free paper

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

Preface

The 6th International Conference on Provable Security (ProvSec 2012) was held in Chengdu, China, September 26–28, 2012. The workshop was organized by the University of Electronic Science and Technology of China.

ProvSec 2012 received 66 submissions from 15 different countries all over the world. The review process was a challenging task. Almost all submissions were carefully evaluated by four reviewers for a total of 262 reviews, and then discussed among the Program Committee. Moreover, 62 external subreviewers gave review comments on their area of expertise. The Program Committee selected 20 papers for the program out of 66 submissions. Among these 20 papers, 16 were accepted as full papers and four as short papers. Further, the program featured two excellent invited talks given by Masayuki Abe (Secure Platform Laboratories, NTT Corporation, Japan) titled "Tools over Bilinear Groups for Modular Design of Cryptographic Tasks" and Victor Shoup (New York University, USA) titled "GNUC Is not UC."

Many people contributed to the success of ProvSec 2012. First we would like to thank all of the authors for submitting their works. We deeply thank the 46 Program Committee members as well as the external reviewers for their volunteer work of reading and discussing the submissions. We thank the Publicity and Publication Co-chairs, Shaoquan Jiang and Yong Yu, for their support. We also would like to thank the local Organizing Committee, Yongjian Liao, Chunxiang Xu, Sheng Cao, and Xuyun Nie, for their dedication and commitment in organizing the conference. Finally, we want to express our gratitude to our generous sponsor: the University of Electronic Science and Technology of China.

September 2012

Tsuyoshi Takagi
Guilin Wang

ProvSec 2012

The 6th International Conference on Provable Security
Chengdu, China
September 26–28, 2012

Organized and Sponsored by the University of Electronic Science and Technology of China (UESTC)

General Chair

Zhiguang Qin	UESTC, China

Publication and Publicity Co-chairs

Shaoquan Jiang	UESTC, China
Yong Yu	UESTC, China

Program Co-chairs

Tsuyoshi Takagi	Kyushu University, Japan
Guilin Wang	University of Wollongong, Australia

Program Committee

Michel Abdalla	ENS and CNRS, France
Man Ho Au	University of Wollongong, Australia
Feng Bao	Institute for Infocomm Research, Singapore
Carlo Blundo	University of Salerno, Italy
Zhenfu Cao	Shanghai Jiaotong University, China
Liqun Chen	Hewlett-Packard Labs, Bristol, UK
Xiaofeng Chen	Xidian University, China
Raymond Choo	University of South Australia, Australia
Georg Fuchsbauer	University of Bristol, UK
David Galindo	University of Luxembourg, Luxembourg
Wei Gao	Ludong University, China
Goichiro Hanaoka	AIST, Japan
Mingxing He	Xihua University, China
Swee-Huay Heng	Multimedia University, Malaysia
Javier Herranz	Universitat Politècnica de Catalunya, Spain
Qiong Huang	South China Agricultural University, China
Tetsu Iwata	Nagoya University, Japan

Shaoquan Jiang	UESTC, China
Aggelos Kiayias	University of Connecticut, USA
Kaoru Kurosawa	Ibaraki University, Japan
Fagen Li	UESTC, China
Benoit Libert	Université Catholique de Louvain, Belgium
Joseph K. Liu	Institute for Infocomm Research, Singapore
Mark Manulis	TU Darmstadt, Germany
Kanta Matsuura	University of Tokyo, Japan
Atsuko Miyaji	JAIST, Japan
Joern Mueller-Quade	Karlsruhe Institute of Technology, Germany
David Naccache	École Normale Supérieure, France
Juan Manuel González Nieto	QUT, Australia
Claudio Orlandi	Bar Ilan University, Israel
Raphael C.-W. Phan	Loughborough University, UK
Josef Pieprzyk	Macquarie University, Australia
C. Pandu Rangan	IIT Madras, India
M. R. Reyhanitabar	University of Wollongong, Australia
Palash Sarkar	Indian Statistical Institute, India
Willy Susilo	University of Wollongong, Australia
Katsuyuki Takashima	Mitsubishi Electric, Japan
Keisuke Tanaka	Tokyo Institute of Technology, Japan
Mehdi Tibouchi	NTT, Japan
Wen-Guey Tzeng	National Chiao Tung University, Taiwan
Huaxiong Wang	NTU, Singapore
Duncan S. Wong	City University of Hong Kong, China
Rui Xue	Chinese Academy of Sciences, China
Alec Yasinsac	University of South Alabama, USA
Yong Yu	UESTC, China
Fangguo Zhang	Sun Yat-sen University, China
Mingwu Zhang	South China Agricultural University, China
Yuliang Zheng	UNC at Charlotte, USA

External Reviewers

Elena Andreeva	Angelo De Caro	Zhen Liu
Toshinori Araki	Emiliano De Cristofaro	Jacob Loftus
George Argyros	Yi Deng	Xu Ma
Rouzbeh Behnia	Nico Doettling	Cuauhtemoc Mancillas-Lopez
Jie Chen	Keita Emura	Ben Martini
Zhenhua Chen	Vincent Grosso	Takahiro Matsuda
Zhili Chen	Fuchun Guo	Shigeo Mitsunari
Kai Yuen Cheong	Vincenzo Iovino	Khoa Nguyen
Ji-Jian Chin	Jin Li	Ryo Nishimaki
Cas Cremers	Ximing Li	Ryo Nojima
Gareth Davies	Kaitai Liang	

Kazumasa Omote
Serdar Pehlivanoglu
Thomas Peters
Le Trieu Phong
Somindu Ramanna
Yusuke Sakai
Katerina Samari
Jacob Schuldt
Sharmila Deva Selvi
Ying Sun
Syh-Yuan Tan
Xiao Tan
Qiang Tang
Yiannis Tselekounis
Sree Vivek
Jianfeng Wang
Keita Xagawa
Li Xiao
Xiang Xie
Shota Yamada
Naoto Yanai
Xu Shi You
Qingyi Zeng
Shengke Zeng
Bo Zhang
Jinshuang Zhang
Liangfeng Zhang
Rui Zhang
Yinghui Zhang
Yun Zhang

Table of Contents

Encryption Schemes

Short Papers

Information Theoretical Security

Tools over Bilinear Groups for Modular Design of Cryptographic Tasks

Masayuki Abe

Secure Platform Laboratories, NTT Corporation, Japan
abe.masayuki@lab.ntt.co.jp

Modular construction is a design paradigm that combines smaller and more general building blocks for larger and more specific cryptographic tasks. It is often used in theoretical works to show feasibility of the tasks and efficient instantiations are considered separately. Some cryptographic tasks find efficient solutions dedicated for their own purposes, yet efficient modular designs are of value as they can be a reasonable alternative for comparison and offer more comprehensible security proofs.

The building blocks are desired to work over the same primitive or setting where the underlying hardness assumptions are made. Among many primitives ranging from well-studied RSA to recent lattices, bilinear groups are certainly one of the reasonable choices as their rich structure allows to implement several tricks while retaining efficiency to some extent. In fact, vast number of cryptographic schemes have been constructed over bilinear groups. A cryptographic scheme such as a signature scheme, encryption, commitment, and so on is called *structure-preserving* over bilinear groups if its public inputs and outputs consist of group elements and relevant verifications are represented by pairing product equations. Such schemes are interoperable and compatible with efficient non-interactive proofs of knowledge and are quite useful in constructing privacy-protecting cryptographic tasks.

This talk is a survey about structure-preserving schemes. We present the state of art, open issues, and applications. Some of the latest results will be explained in more depth.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, p. 1, 2012.

One-Move Convertible Nominative Signature in the Standard Model

Dennis Y.W. Liu[1,2] and Duncan S. Wong[2]

[1] School of Professional and Continuing Education, University of Hong Kong
[2] Department of Computer Science, City University of Hong Kong
dennis.liu@hkuspace.hku.hk, duncan@cityu.edu.hk

Abstract. A Nominative Signature (NS) is a non-self-authenticating signature which is jointly generated by a signer (or a nominator) and a user (or a nominee), but once generated, its validity can only be determined by the user. No one else including the signer can tell the signature's validity unless the user confirms or disavows so, while the user cannot cheat either. One-move NS is an efficient type of NS that requires the signer to send only one message to the user during the signature generation stage. Currently, there exists only one one-move NS scheme which is proven secure in the standard model, and is convertible, that is, the user can transform a nominative signature to a publicly verifiable one without the help of the signer. However, the number of elements in the keys of both signer and user grows linearly with the value of the security parameter. In this paper, we propose a new one-move NS which is convertible, can be proven secure in the standard model, and also has a constant number of elements in the keys of both signer and user. We apply the Boneh-Boyen short standard signature in a novel way to build this nominative signature scheme. We show that this new scheme achieves the best performance among all the schemes proven secure in the standard model, and its security relies only on the standard q-SDH and DDH assumptions.

Keywords: nominative signature, undeniable signature, non-self-authenticating signature.

1 Introduction

A nominative signature (NS) scheme [14,24,17] allows a *signer* (or *nominator*) A to work jointly with a *user* (or *nominee*) B to generate a signature σ on a message m such that the validity of σ can only be verified by B. In addition, only B can convince a (third-party) *verifier* C the validity of σ by running a confirmation/disavowal protocol which also ensures that B cannot cheat.

In [12], Huang et al. used a practical scenario to illustrate a possible application of NS. A hospital authority certifies and signs on official documents containing the medical records of each patient. For privacy, the patient, however, is the only one who can show and demonstrate the legitimacy of his/her own medical documents to others, such as an insurance company. NS plays an

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 2–20, 2012.

important role here where the hospital authority, the patient and the insurance company are the signer, the user, and the verifier, respectively. Some may notice that the hospital authority may simply release a medical document without participating in the nominative signature generation, but the patient can accuse the hospital authority of making false claims on the patient's medical records. The role of NS in this scenario is to produce a mutual agreement on the validity of the patient's medical documents. Without the hospital authority's involvement, the professional validity of the medical document cannot be ensured, while without the patient's agreement, the hospital authority cannot forge a medical document. Also, only the patient can demonstrate the validity of the medical document to the insurance company (i.e. the verifier) for making an insurance claim. The hospital authority cannot announce to the public or leak anything provable about the patient's medical records.

In general, NS is also useful in user certification systems [23], which concerns about letting a user B convince a (third-party) verifier C the validity of B's birth certificate, driver's licence, academic transcripts or other documents, that are issued by an authority A, but not allowing C to further disseminate the validity information of any of B's certificates without B's consent. It was believed that user certification systems could be constructed from Universal Designated Verifier Signature (UDVS) [23]. However, the signer A has to be fully trusted by the user B. If A is malicious, A may disclose the certificate which is publicly verifiable, and, A can generate a UDVS signature all by himself. An alternative non-self-authenticating signature primitive that may be used to construct such systems is called Designated Confirmer Signature (DCS) [7] in which the ability to prove the validity of a signature is shifted to the user B. In such a DCS scheme, the signer A is still able to confirm the signature to any public verifier C, though the signature is not public verifiable. For a detailed discussion of non-self-authenticating signatures, we refer readers to [17].

Since the introduction of NS by Kim et al. [14] in 1996, there have been a number of refinements on the definitions and security models [17,12,27,21]. The construction of NS has also been improved significantly, particularly on the number of message flows (also known as 'moves') between the signer and the user during the nominative signature generation stage (four-move in [17], two-move in [15] and one-move in [12,27,28,21]). The concept of convertibility is also considered in [13,12,27,28,21], in which a nominative signature can be converted to a publicly verifiable standard signature by the user B. All of the constructions are proven secure in the random oracle model, not until recently, Schuldt et al. [21] proposed an NS scheme which is proven secure in the standard model. Their scheme is based on the standard signature scheme due to Waters [26]. It is not only one-move but is also convertible. However, the number of elements in the keys of both signer A and user B grows linearly with the value of the security parameter as the scheme is based on Waters' standard signature scheme.

Our Results. In this paper, we propose a new NS scheme which is one-move, convertible, and also has security proven in the standard model. Furthermore, the number of elements in the keys of both signer A and user B is constant and

does not grow with the value of the security parameter. In our construction, we apply the Boneh-Boyen short standard signature (BB) [2] in a novel and interesting way, so that it is non-self-authenticating, convertible, and also supports efficient confirmation/disavowal. When compared with the only available NS scheme proven secure in the standard model [21], our new scheme has shorter keys, and makes 45% improvement in efficient during the signature generation. More details will be given in Sec. 5.

Outline. The definition of convertible nominative signature (CNS) is given in Sec. 2 and its security model in Sec. 3. Our proposed CNS scheme is then described in Sec. 4. A comparison on the efficiency with previous CNS schemes will be discussed in Sec. 5. Finally, the paper is concluded in Sec. 6.

2 Convertible Nominative Signature Definitions

In this paper, we focus our attention on *Convertible* NS which requires only one move from the signer to the user during the signature generation stage. Hence in the following, we give a definition for one-move convertible NS:

A one-move Convertible Nominative Signature (CNS) consists of six probabilistic polynomial-time (PPT) algorithms and three protocols. Algorithms are (SystemSetup, SKeyGen, UKeyGen, NSVer, Conv, Ver); protocols are (SigGen, Confirmation and Disavowal).

1. SystemSetup: On input 1^k where $k \in \mathbf{N}$ is a security parameter, it generates a list of system parameters denoted by param.
2. SKeyGen: On input param, it generates a public/private key pair (pk_A, sk_A) for the signer.
3. UKeyGen: On input param, it generates a public/private key pair (pk_B, sk_B) for the user.
4. NSVer: On input a message $m \in \{0,1\}^*$, a nominative signature σ, a public key pk_A and a private key sk_B, it returns valid or invalid.

A CNS scheme proceeds as follows. SystemSetup is first invoked for generating param. SKeyGen and UKeyGen are then executed to initialize the two entities, the signer A and the user B. To generate a nominative signature σ, A chooses a message m, and carries out SigGen protocol below with B. In the one-move setting, A is the initiator and B is the responder. A generates a *partial* nominative signature denoted by σ' and sends it to B. B then generates and outputs a nominative signature denoted by σ. Formally, the SigGen consists of two algorithms, (Sign, Receive), which are carried out by signer A (who is holding (pk_A, sk_A)) and user B (who is holding (pk_B, sk_B)), respectively.

> SigGen Protocol: Besides (pk_A, sk_A), A's input is (param, m, pk_B). Besides (pk_B, sk_B), B's input is (param, m, pk_A). SigGen protocol proceeds as follows:

1. A generates $\sigma' \leftarrow \mathsf{Sign}(\mathsf{param}, pk_B, m, sk_A)$ and sends σ' to B;
2. B generates $\sigma \leftarrow \mathsf{Receive}(\mathsf{param}, pk_A, m, \sigma', sk_B)$.

At the end of the protocol, B either outputs a nominative signature σ or $\perp$ indicating the failure of the protocol run.

Unlike the original definition in [17], the SigGen protocol defined above is specific to the one-move setting, that is, signer A initiates and generates a *partial* nominative signature σ', then B generates the final nominative signature σ upon receiving σ'. Note that the signature space should be specified explicitly in each CNS construction. In the following, let $\mathcal{S}(pk_A, pk_B)$ be the signature space.

For a nominative signature σ in $\mathcal{S}(pk_A, pk_B)$, the validity of σ can be determined by B using NSVer. If σ is valid, B can prove its validity to any third party C using the following Confirmation protocol, otherwise, B can prove its invalidity to C using a Disavowal protocol.

4. NSVer: On input a message $m \in \{0,1\}^*$, a nominative signature σ, a public key pk_A and a private key sk_B, it returns valid or invalid.

   Confirmation/Disavowal Protocol: On input (m, σ, pk_A, pk_B), B sets a bit μ to 1 if $\mathsf{valid} \leftarrow \mathsf{NSVer}(m, \sigma, pk_A, sk_B)$; otherwise, μ is set to 0. B first sends μ to C. If $\mu = 1$, Confirmation protocol is carried out; otherwise, Disavowal protocol is carried out. At the end of the protocol, C outputs either accept or reject while B has no output.

5. Conv: On input an alleged message-signature pair (m, σ), if $\mathsf{valid} \leftarrow \mathsf{NSVer}(m, \sigma, pk_A, sk_B)$, B runs $\mathsf{Conv}(m, \sigma, pk_A, sk_B)$ to extract a standard (publicly verifiable) signature σ^{std}.
6. Ver: On input $(m, \sigma^{Std}, pk_A, pk_B)$, it returns valid or invalid.

Correctness: The scheme is said to satisfy the correctness requirement if, for all $\mathsf{param} \leftarrow \mathsf{SystemSetup}(1^k)$, $(pk_A, sk_A) \leftarrow \mathsf{SKeyGen}(\mathsf{param})$, $(pk_B, sk_B) \leftarrow \mathsf{UKeyGen}(\mathsf{param})$, all message-signature pairs (m, σ) such that $\sigma \leftarrow \mathsf{Receive}(\mathsf{param}, pk_A, m, \mathsf{Sign}(\mathsf{param}, pk_B, m, sk_A), sk_B)$, and all converted standard signatures $\sigma^{std} \leftarrow \mathsf{Conv}(m, \sigma, pk_A, sk_B)$, we have:

- $\mathsf{valid} \leftarrow \mathsf{NSVer}(m, \sigma, pk_A, sk_B)$;
- C outputs accept at the end of the Confirmation protocol when both B and C follow the protocol honestly, and;
- $\mathsf{valid} \leftarrow \mathsf{Ver}(m, \sigma^{std}, pk_A, pk_B)$.

Given a message m and a nominative signature $\sigma \in \mathcal{S}(pk_A, pk_B)$, if $\mathsf{invalid} \leftarrow \mathsf{NSVer}(m, \sigma, pk_A, sk_B)$, then C outputs accept at the end of the Disavowal protocol provided that both B and C follow the protocol honestly.

The *soundness* of Confirmation (resp. Disavowal) protocol requires that no PPT user can convince a third party that an invalid (resp. valid) nominative signature is "valid" (resp. "invalid").

3 Security Model

Before presenting the security games for CNS, we begin with the oracles.

- CreateSigner: On input a query, it generates a key pair (pk_A, sk_A) using SKeyGen, and returns pk_A.
- CreateUser: On input a query, it generates a key pair (pk_B, sk_B) using UKeyGen, and returns pk_B.
- Corrupt: On input a public key pk, if pk is generated by CreateSigner or CreateUser, the corresponding private key is returned; otherwise, $\perp$ is returned. pk is said to be *corrupted*.
- Sign: On input a message m, two distinct public keys, pk_1 (signer) and pk_2 (user), returns σ' where σ' is a partial nominative signature.
- Receive: On input a message m, a partial nominative signature σ', two distinct public keys, pk_1 (signer) and pk_2 (user), returns σ where σ is a valid nominative signature. σ is said to be *valid* on m with respect to pk_1 and pk_2 if $\mathsf{valid} \leftarrow \mathsf{NSVer}(m, \sigma, pk_1, sk_2)$ where sk_2 is the corresponding private key of pk_2.
- Confirmation/Disavowal: On input a message m, a nominative signature σ and two public keys pk_1 (signer) and pk_2 (user), let sk_2 be the corresponding private key of pk_2. The oracle, acting as the user (prover) and runs $\mathsf{NSVer}(m, \sigma, pk_1, sk_2)$. If the output is valid, the oracle returns a bit $\mu = 1$ and carry out the Confirmation protocol. Otherwise, $\mu = 0$ and carry out the Disavowal protocol.
- Convert: On input (m, σ, pk_1, pk_2) such that $\mathsf{valid} \leftarrow \mathsf{NSVer}(m, \sigma, pk_1, sk_2)$, the oracle returns σ^{Std} such that $\mathsf{valid} \leftarrow \mathsf{Ver}(m, \sigma^{Std}, pk_1, pk_2)$.

A secure convertible nominative signature (CNS) scheme should satisfy the following security requirements: (1) **Unforgeability Against Malicious Users**, (2) **Unforgeability Against Malicious Signers**, (3) **Invisibility**, (4) **Non-transferability**, and (5) **User-only Conversion**.

Remark: Similar to [21], we adopt the registered-key model [1], in which the adversary is required to certify that the public keys used are properly generated and it knows the corresponding private keys. We employ the three oracles CreateSigner, CreateUser and Corrupt to capture this model.

3.1 Unforgeability against Malicious Users

This security notion ensures that the user cannot forge a valid nominative signature without the aid of the signer.

Game Unforgeability against Malicious Users: Let S be the simulator and F be a forger.

1. (*Initialization*) Let $k \in \mathbf{N}$ be a security parameter. S runs $\mathsf{param} \leftarrow \mathsf{SystemSetup}(1^k)$ and $(pk_A, sk_A) \leftarrow \mathsf{SKeyGen}(\mathsf{param})$. Then, F is invoked with (param, pk_A).

2. (*Attacking Phase*) F is allowed to make queries to the oracle CreateSigner, CreateUser, Corrupt, Sign mentioned above.
3. (*Output Phase*) F outputs $(m^*, \sigma^*, pk_B, sk_B)$.

F *wins* the game if $\mathsf{valid} \leftarrow \mathsf{NSVer}(m^*, \sigma^*, pk_A, sk_B)$ provided that:

1. F has never queried $\mathsf{Corrupt}(pk_A)$ for corrupting sk_A;
2. (pk_B, sk_B) must be created by querying CreateUser;
3. (m^*, pk_A, pk_B) has never been queried to Sign.

F's advantage in this game is defined to be the probability that F wins.

Definition 1. *A CNS is unforgeable against malicious users if no PPT forger F has a non-negligible advantage in Game Unforgeability Against Malicious Users.*

Note that Confirmation/Disavowal and Convert oracles are not needed to provide in the game above as F can readily carry out these protocols and algorithm as (malicious) users by making use of CreateUser and Corrupt oracles.

A slightly stronger notion of security, called Strong Unforgeability Against Malicious Users requires that the adversary cannot even generate a new signature on a previous signed message. The game is similar to Game Unforgeability Against Malicious Users, with an exception that (m^*, pk_A, pk_B) can be queried to Sign by F. The output (m^*, σ^*) by F must not be a query result from Sign before. Our construction proposed in the next section satisfies this stronger security property.

3.2 Unforgeability against Malicious Signers

This notion captures the security requirement that the signer cannot forge a valid nominative signature without the aid of the user.

Game Unforgeability against Malicious Signers: Let S be the simulator and F be a forger.

1. (*Initialization*) Let $k \in \mathbf{N}$ be a security parameter. S runs $\mathsf{param} \leftarrow \mathsf{SystemSetup}(1^k)$ and $(pk_B,\ sk_B) \leftarrow \mathsf{UKeyGen}(\mathsf{param})$. Then, F is invoked with $(\mathsf{param},\ pk_B)$.
2. (*Attacking Phase*) F is allowed to make queries to the oracles CreateSigner, CreateUser, Corrupt, Receive, Confirmation/Disavowal and Convert mentioned above.
3. (*Output Phase*) F outputs $(m^*, \sigma^*, pk_A, sk_A)$.

F *wins* the game if $\mathsf{valid} \leftarrow \mathsf{NSVer}(m^*, \sigma^*, pk_A, sk_B)$ provided that

1. F has never queried $\mathsf{Corrupt}(pk_B)$ for corrupting sk_B;
2. (pk_A, sk_A) must be created by querying CreateSigner;
3. $(m^*, \sigma'^*, pk_A, pk_B)$ has never been queried to Receive such that $\sigma^* \leftarrow \mathsf{Receive}(m^*, \sigma'^*, pk_A, pk_B)$.

F's advantage in this game is defined to be the probability that F wins.

Definition 2. *A CNS is unforgeable against malicious signers if no PPT forger F has a non-negligible advantage in* Game Unforgeability Against Malicious Signers.

Note that the Sign oracle are not needed to provide in the game above as F can readily carry out this algorithm as (malicious) signers by making use of CreateUser and Corrupt oracles.

Similar to Strong Unforgeability Against Malicious Users, there is a stronger security notion called Strong Unforgeability Against Malicious Signers. The game is similar to Game Unforgeability Against Malicious Signers, with an exception that $(m^*, \sigma'^*, pk_A, pk_B)$ can be queried to Receive by F. The output (m^*, σ^*) by F must not be a query result from Receive before. Our construction proposed in the next section also satisfies this stronger security property.

3.3 Invisibility

We require that no verifier C (including signer A) can tell the validity of a nominative signature, except user B. In the formalization below, we define an auxiliary algorithm called NSSim (which stands for Nominative Signature Simulator). The algorithm takes $(\mathsf{param}, pk_A, pk_B, m, \sigma^{valid})$ as input, where σ^{valid} is a valid nominative signature for message m under signer A's public key pk_A and user B's public key pk_B, outputs $\sigma^{invalid}$ so that $\sigma^{invalid} \in \mathcal{S}(pk_A, pk_B)$ but $\sigma^{invalid}$ is no longer a valid nominative signature for m under (pk_A, pk_B). The purpose of introducing NSSim is to explicitly define the capability of the public who can always convert a valid nominative signature to an invalid one while both σ^{valid} and $\sigma^{invalid}$ would look indistinguishable to verifier C, and only user B can tell which signature is valid and which one is not, and by this, we model the Invisibility requirement. Also note that NSSim has to be explicitly described in the construction of a new NS scheme in order to have the new scheme be proven satisfying the Invisibility requirement.

Game Invisibility: The initialization and attacking phases are the same as that of Game Unforgeability Against Malicious Signers. In the game, the adversary is a distinguisher D. Below are the two additional phases in the game.

1. (*Challenge Signature Generation Phase*) At some point in the game, D sends a message m^* to the simulator while acting as the signer for carrying out a run of SigGen with the simulator which acts as the user. Let σ^{valid} be the nominative signature generated by the simulator at the end of the SigGen protocol run. Note that $\mathsf{valid} \leftarrow \mathsf{NSVer}(m^*, \sigma^{valid}, pk_A, sk_B)$. The challenge signature σ^* is then generated by the simulator based on the outcome of a random coin toss b. If $b = 1$, set $\sigma^* = \sigma^{valid}$. If $b = 0$, set $\sigma^* \leftarrow \mathsf{NSSim}(\mathsf{param}, pk_A, pk_B, m, \sigma^{valid})$.
2. (*Guess Phase*) D continues querying the oracles, until it outputs a guess b'.

D *wins* the game if $b' = b$ provided that

1. D has never queried $\mathsf{Corrupt}(pk_B)$ for corrupting sk_B;
2. (pk_A, sk_A) must be created by querying CreateSigner;

3. $(m^*, \sigma'^*, pk_A, pk_B)$ has never been queried to Receive such that $\sigma^* \leftarrow \mathsf{Receive}(m^*, \sigma'^*, pk_A, pk_B)$.
4. $(m^*, \sigma^*, pk_A, pk_B)$ has never been queried to Confirmation/Disavowal and Convert.

D's advantage in this game is defined as $P[b' = b] - \frac{1}{2}$.

Definition 3. *A CNS has the property of invisibility if no PPT distinguisher D has a non-negligible advantage in Game Invisibility.*

3.4 Non-transferability

The Confirmation/Disavowal protocols should be zero-knowledge so that no PPT verifier (including the signer) can transfer the proof transcript of a nominative signature to others. The zero-knowledge property of the protocols implies non-transferability of proof transcripts. Non-transferability requires that a verifier cannot produce any evidence during the Confirmation/Disavowal protocols and thus ensures that the verifier cannot convince a third party that the validity/invalidity of a message-signature pair. Non-transferability follows directly from the Invisibility property which implies that an invalid signature is indistinguishable from a valid one, and the zero-knowledge property of the Confirmation/Disavowal protocols implies that a verifier can simulate the proof transcripts. Hence, a verifier can sample a signature from the signature space and create the corresponding proof transcripts which are indistinguishable from those that are honestly generated from the Confirmation/Disavowal protocols.

3.5 User-Only Conversion

This security notion requires that it should be infeasible for anyone other than the user to convert a valid nominative signature to a publicly-verifiable one. We consider the following game.

Game User-Only Conversion: Suppose C is a malicious adversary who wants to convert a nominative signature to a standard signature. The initialization and attacking phases are the same as that of Game Unforgeability Against Malicious Signers.

- (*Challenge Signature Generation Phase*) At some point in the game, C is given (m^*, σ^*) from the simulator.
- (*Conversion Phase*) At the end of the game, C outputs (m^*, σ^{Std}) if σ^* is a valid signature on m^* with respect to pk_A and pk_B. Otherwise, it returns $\perp$.

C *wins* if $\mathsf{valid} \leftarrow \mathsf{Ver}(m, \sigma^{Std}, pk_A, pk_B)$ provided that

1. C has never queried $\mathsf{Corrupt}(pk_B)$ for corrupting sk_B;
2. (pk_A, sk_A) must be created by querying CreateSigner;
3. $(m^*, \sigma^*, pk_A, pk_B)$ has never been queried to Confirmation/Disavowal and Convert;

C's advantage is defined as the probability that C wins.

Definition 4. *A CNS satisfies user-only conversion if no PPT adversary C has a non-negligible advantage in* Game User-only Conversion.

Theorem 1. *If a CNS satisfies invisibility with respect to Def. 3, the scheme satisfies user-only conversion.*

Proof. We prove by contradiction. Suppose there is a scheme which does not satisfy User-only Conversion, we show that it cannot satisfy Invisibility. We construct a distinguisher D which breaks the Game Invisibility by using a successful adversary C in Game User-only Conversion. D acts also as the simulator of Game User-only Conversion. First, as the initialization and attacking phases of the Game User-only Conversion are the same as that of the Game Invisibility, D passes param and pk_B to C after receiving them from its simulator in Game Invisibility. D is able to simulate all available oracle queries perfectly from C by routing them to its simulator. At the end of the Game User-only Conversion, if C outputs a valid signature σ^{std}, D knows that σ^* must be a valid nominative signature, and hence wins the Game Invisibility.

4 Our Construction

Let G_1, G_2 and G_T be cyclic groups of prime order p. Let g_1 be the generator of G_1 and g_2 be the generator of G_2. Let $e : G_1 \times G_2 \rightarrow G_T$ be an efficiently computable map with the following properties: (1) Bilinear: for all $a, b \in \mathbb{Z}$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$; and (2) Non-degenerate: $e(g_1, g_2) \neq 1$ where 1 is the identity element of G_T.

Our construction requires an asymmetric (i.e. $G_1 \neq G_2$) pairing algorithm with only efficiently-computable isomorphism $\psi : G_2 \rightarrow G_1$. For more details about asymmetric bilinear pairing and previous cryptographic applications, we refer readers to [22,4,2].

The computational assumptions of our construction are given in Appendix B.

4.1 The Scheme

In [2], a short standard signature (BB) scheme based on bilinear pairing that is strongly existentially unforgeable under an adaptive chosen message attack in the standard security model was proposed. We will adopt this standard signature scheme as the building block of our CNS scheme. Here is a brief review of the BB's scheme:

Let (G_1, G_2) be bilinear groups for some prime p. Let $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$ be a collision resistant hash function.

Key Generation: Select random generators $g_1 \in_R G_1$ and $g_2 \in_R G_2$, and random integers x, $y \in_R \mathbb{Z}_p^*$. Compute $u = {g_2}^x$ and $v = {g_2}^y$. Also compute $z = e(g_1, g_2) \in G_T$. The public key is the tuple (g_1, g_2, u, v, z). The secret key is (x, y).

Signing: Given a secret key (x, y) and a message m, pick a random $r \in_R \mathbb{Z}_p \setminus -\{\frac{x+H(m)}{y}\}$ and compute $\sigma = {g_1}^{1/(x+H(m)+yr)}$. Here, the inverse $1/(x+H(m)+yr)$ is computed modulo p. The signature is the pair (σ, r).

Verification: Given a public key (g_1, g_2, u, v, z), a message m, and a signature (σ, r), verify that $(g_1, g_2, \sigma, g_2^{H(m)}v^r u)$ is a DDH tuple by testing whether $e(\sigma, g_2^{H(m)}v^r u) = z$. If the equality holds the signature is declared valid; otherwise it is declared invalid.

Outline of Our CNS Scheme. The signer A generates a BB signature $\sigma' = (\sigma^{BB}, r_A)$ and sends it to the user B. B 'masks' σ^{BB} in a fashion of ElGamal encryption [10] to produce $(\sigma_1, r_A, \alpha_1)$ so that even A is not able to determine if it contains her valid BB signature. Furthermore, B generates his own BB signature (σ_2, r_B) on the 'message' σ_1 and 'masks' it in the same fashion of ElGamal encryption to produce $(\sigma_2, r_B, \alpha_2)$ so that the CNS can satisfy unforgeability against malicious signer. The final CNS is therefore $(\sigma_1, \sigma_2, r_A, r_B, \alpha_1, \alpha_2)$. We use ElGamal encryption to 'mask' (i.e. blind the BB signature from the public) because of its homomorphic property which allows us to implement the (zero-knowledge) proof of knowledge system for the **Confirmation**/**Disavowal** protocols. Furthermore, it allows B to easily remove the mask during **Conversion**, hence achieving convertibility in our CNS. Fig. 1 illustrates the algorithms and here is a detailed description of the scheme:

SystemSetup: Let $k \in \mathbf{N}$ be a system parameter. The algorithm generates three cyclic groups G_1, G_2 and G_T of prime order $p \geq 2^k$ and a bilinear map $e : G_1 \times G_2 \to G_T$. It also specifies a collision resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p$. Select random generators $g_1 \in G_1$ and $g_2 \in G_2$. Let $\mathsf{param} = (p, G_1, G_2, G_T, g_1, g_2, H)$.

SKeyGen: On input param, generate randomly $x_{A_1}, x_{A_2} \in \mathbb{Z}_p^*$. Calculate $y_{A_1} = g_2^{x_{A_1}}$ and $y_{A_2} = g_2^{x_{A_2}}$. Let (y_{A_1}, y_{A_2}) be the public keys, pk_A, and (x_{A_1}, x_{A_2}) be the private keys, sk_A, of signer A.

UKeyGen: On input param, generate randomly $x_{B_1}, x_{B_2}, x_{B_3} \in \mathbb{Z}_p^*$. Calculate $y_{B_1} = g_2^{x_{B_1}}$, $y_{B_2} = g_2^{x_{B_2}}$ and $y_{B_3} = g_1^{x_{B_3}}$. Let $(y_{B_1}, y_{B_2}, y_{B_3})$ be the public keys, pk_B, and $(x_{B_1}, x_{B_2}, x_{B_3})$ be the private keys, sk_B, of user B.

SigGen Protocol: Let $m \in \{0,1\}^*$ be a message. A and B carry out the following:

1. A picks a random $r_A \in \mathbb{Z}_p \setminus -\{\frac{x_{A_1}+H(m||y_B)}{x_{A_2}}\}$ where $y_B = y_{B_1}||y_{B_2}||y_{B_3}$, computes $\sigma^{BB} = {g_1}^{1/(x_{A_1}+H(m||y_B)+x_{A_2}r_A)}$ and sends $\sigma' \leftarrow (\sigma^{BB}, r_A)$ to B. Here, the inverse $1/(x_{A_1} + H(m||y_B) + x_{A_2}r_A)$ is computed modulo p.
2. B verifies if $e(g_1, g_2) \stackrel{?}{=} e(\sigma^{BB}, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})$. If so, B generates $\sigma_1 = \sigma^{BB} {y_{B_3}}^{r_1}$ and $\alpha_1 = {g_1}^{r_1}$ where $r_1 \in_R \mathbb{Z}_p$, picks a random $r_B \in \mathbb{Z}_p \setminus -\{\frac{x_{B_1}+H(\sigma_1)}{x_{B_2}}\}$ and generates $\sigma_2 = {g_1}^{1/(x_{B_1}+H(\sigma_1)+x_{B_2}r_B)} {y_{B_3}}^{r_2}$ and $\alpha_2 = {g_1}^{r_2}$ where $r_2 \in_R \mathbb{Z}_p$ and forms the signature (m, σ) where $\sigma = (\sigma_1, \sigma_2, r_A, r_B, \alpha_1, \alpha_2)$.

SKeyGen(param):
$x_{A_1}, x_{A_2} \leftarrow \mathbb{Z}_p^*$
$y_{A_1} \leftarrow g_2^{x_{A_1}}$, $y_{A_2} \leftarrow g_2^{x_{A_2}}$
Set $pk_A = (y_{A_1}, y_{A_2})$, $sk_A = (x_{A_1}, x_{A_2})$
return $(pk_A,\ sk_A)$

NSVer(m, σ, pk_A, sk_B):
if $e(\alpha_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})^{x_{B3}}$
$= e(\sigma_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})/e(g_1, g_2) \wedge$
$e(\alpha_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})^{x_{B3}}$
$= e(\sigma_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})/e(g_1, g_2)$
return valid
else return invalid

UKeyGen(param):
$x_{B_1}, x_{B_2}, x_{B_3} \leftarrow \mathbb{Z}_p^*$
$y_{B_1} \leftarrow g_2^{x_{B_1}}$, $y_{B_2} \leftarrow g_2^{x_{B_2}}$, $y_{B_3} \leftarrow g_1^{x_{B_3}}$
Set $pk_B = (y_{B_1}, y_{B_2}, y_{B_3})$
$sk_B = (x_{B_1}, x_{B_2}, x_{B_3})$
return $(pk_B,\ sk_B)$

Conv(m, σ, sk_B):
$r_3 \leftarrow_R \mathbb{Z}_p$
$\delta_1 \leftarrow \sigma_1 / \alpha_1^{x_{B_3}}$
$\delta_2 \leftarrow g_1^{1/(x_{B_1} + H(\delta_1) + x_{B_2} r_3)}$
Set $\delta = (\delta_1, \delta_2, r_A, r_3)$
return δ

Send(param, pk_B, m, sk_A):
$r_A \leftarrow_R \mathbb{Z}_p \setminus -\{\frac{x_{A_1} + H(m||y_B)}{x_{A_2}}\}$
Set $y_B = y_{B_1} || y_{B_2} || y_{B_3}$
$\sigma^{BB} \leftarrow g_1^{1/(x_{A_1} + H(m||y_B) + x_{A_2} r_A)}$
Set $\sigma' = (\sigma^{BB}, r_A)$
return σ'

Ver(m, δ, pk_A, pk_B):
if $e(g_1, g_2) = e(\delta_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A}) \ \wedge$
$e(g_1, g_2) = e(\delta_2, y_{B_1} g_2^{H(\delta_1)} y_{B_2}^{r_3})$
return valid
else return invalid

Receive(param, pk_A, m, σ', sk_B):
if $e(g_1, g_2) = e(\sigma^{BB}, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})$
$\sigma_1 \leftarrow \sigma^{BB} {y_{B_3}}^{r_1}$, $\alpha_1 \leftarrow {g_1}^{r_1}$ where $r_1 \leftarrow_R \mathbb{Z}_p$
$r_B \leftarrow_R \mathbb{Z}_p \setminus -\{\frac{x_{B_1} + H(\sigma_1)}{x_{B_2}}\}$
$\sigma_2 \leftarrow g_1^{1/(x_{B_1} + H(\sigma_1) + x_{B_2} r_B)} {y_{B_3}}^{r_2}$
$\alpha_2 \leftarrow {g_1}^{r_2}$ where $r_2 \leftarrow_R \mathbb{Z}_p$
Set $\sigma = (\sigma_1, \sigma_2, r_A, r_B, \alpha_1, \alpha_2)$
return $(m,\ \sigma)$
else return $\perp$

Fig. 1. Concrete Construction of the CNS Algorithms

Signature Space: σ is said to be in the signature space $\mathcal{S}(pk_A, pk_B)$ if σ_1, $\sigma_2, \alpha_1, \alpha_2 \in G_1$ and $r_A, r_B \in \mathbb{Z}_p$. In order to check the validity of a nominative signature, the following algorithm is executed by user B.

NSVer: On input (m, σ, pk_A, sk_B) where σ is in $\mathcal{S}(pk_A, pk_B)$, the algorithm checks if $e(\alpha_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})^{x_{B3}} \stackrel{?}{=} e(\sigma_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})/e(g_1, g_2)$, and $e(\alpha_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})^{x_{B3}} \stackrel{?}{=} e(\sigma_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})/e(g_1, g_2)$.

If so, output valid; otherwise, output invalid.

Confirmation/Disavowal Protocol: B first runs NSVer(m, σ, pk_A, sk_B). If the output is valid, B sends $\mu = 1$ to a verifier C and carries out the following (zero-knowledge) proof of knowledge with the verifier:

$$PoK\{x_{B3} \ : \ {g_1}^{x_{B3}} = y_{B3}$$
$$\wedge \ e(\alpha_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})^{x_{B3}} = e(\sigma_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})/e(g_1, g_2)$$
$$\wedge \ e(\alpha_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})^{x_{B3}} = e(\sigma_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})/e(g_1, g_2)\}$$

Otherwise, B sends $\mu = 0$ to C and carries out the following (zero-knowledge) proof of knowledge with the verifier:

$$PoK\{x_{B3} \ : \ {g_1}^{x_{B3}} = y_{B3}$$
$$\wedge \ (e(\alpha_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})^{x_{B3}} \neq e(\sigma_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})/e(g_1, g_2)$$
$$\vee \ e(\alpha_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})^{x_{B3}} \neq e(\sigma_2, y_{B_1} g_2^{H(\sigma_1)} y_{B_2}^{r_B})/e(g_1, g_2))\}$$

Conv: On input (m, σ, sk_B) where σ is a valid nominative signature on m respect to A and B, B randomly picks $r_3 \in \mathbb{Z}_p$, and generates $\delta = (\delta_1, \delta_2, r_A, r_3)$ where $\delta_1 = \sigma_1 / \alpha_1^{x_{B_3}}$ and $\delta_2 = {g_1}^{1/(x_{B_1} + H(\delta_1) + x_{B_2} r_3)}$.

Ver: On input (m, δ, pk_A, pk_B), C checks if $e(g_1, g_2) \stackrel{?}{=} e(\delta_1, y_{A_1} g_2^{H(m||y_B)} y_{A_2}^{r_A})$ and $e(g_1, g_2) \stackrel{?}{=} e(\delta_2, y_{B_1} g_2^{H(\delta_1)} y_{B_2}^{r_3})$

If so, output valid; otherwise, output invalid.

For Invisibility (Sec. 3.3), we define $\sigma^{invalid} \leftarrow \mathsf{NSSim}(\mathsf{param}, pk_A, pk_B, m, \sigma^{valid})$ as follows. Given $\sigma^{valid} := (\sigma_1, \sigma_2, r_A, r_B, \alpha_1, \alpha_2)$, NSSim outputs $\sigma^{invalid}$ as $(\sigma_1', \sigma_2', r_A, r_B', \alpha_1', \alpha_2')$ where $\sigma_1', \sigma_2' \leftarrow_R G_1$, $r_B' \leftarrow_R \mathbb{Z}_p$, $\alpha_1', \alpha_2' \leftarrow_R G_1$.

Remark: The confirmation/disavowal protocols given above are Σ-protocols [9] with special soundness and perfect special honest-verifier zero-knowledge. Their corresponding four-move fully fledged perfect zero-knowledge protocols can be obtained by applying the transformation in [8]. We therefore omit the details here and refer readers to [8].

The security analysis of the scheme is given in Appendix C.

5 Efficiency Analysis

In Table 1, comparisons on the signature size, key sizes for signer A and user B, signature generation efficiency in terms of modular exponentiation calculation by A (Sign) and B (Receive) individually, and the security assumptions used for unforgeability and invisibility, are shown. When comparing with previous schemes, ours achieves strong unforgeability with comparable key size (unchanged for A's key and only two additional G and $\mathbb{Z}_p^*$ elements for B's key when compared with the scheme in [28]). The number of elements for the keys of the scheme in [21] is proportional to the security parameters, while in our scheme, the key size remains constant. Our scheme has a 45% increase in efficiency during signature generation, in terms of number of modular exponentiation calculations, when compared with the scheme in [21]. While the schemes in [12,27,28] are secure in random oracle model, our scheme is proven secure using standard computational assumptions (i.e. CDH & q-SDH) which gives more confident in its security among currently available NS schemes.

Table 1. Comparison with Existing Secure One-Move NS Schemes

Scheme	σ	A_{Key}	B_{Key}	$SigGen$[a]
HLW08 [12]	4G	1G+1$\mathbb{Z}_p^*$	1G+1$\mathbb{Z}_p^*$	1 + 4
ZLY08 [27]	2G	1G+1$\mathbb{Z}_p^*$	1G+1$\mathbb{Z}_p^*$	1 + 2
ZY09 [28]	1G	2G+2$\mathbb{Z}_p^*$	2G+2$\mathbb{Z}_p^*$	1 + 2
SH11 [21]	3G+$\mathbb{Z}_p$	2G+(n+2)$\mathbb{Z}_p$[b]	5G+[2(n+1)+3]$\mathbb{Z}_p$	3 + 8
Ours	4G+2$\mathbb{Z}_p$	2G+2$\mathbb{Z}_p$	3G+3$\mathbb{Z}_p$	1 + 5

	Unforgeability	Invisibility	Convertibility	Standard Model
HLW08 [12]	WCDH-I, WCDH-II	WDDH	×	×
ZLY08 [27]	WCDH-I, WCDH-II	WDDH	√	×
ZY09 [28]	CDH	3-DDH	√	×
SH11 [21]	CDH, DLP, DLiP	DLiP	√	√
Ours	q-SDH	DDH	√	√

[a] No. of Modular Exponentiations in signature generation (Sign + Receive).
[b] n: No. of bits of message to be signed.

6 Conclusion

We proposed a more efficient one-move convertible nominative signature (CNS) scheme which is secure under standard computational assumptions, in terms of the reduced key size and the increased efficiency in signature generation when compared with the scheme in [21]. Our scheme is also strongly unforgeable. It is under our further investigation whether our techniques in our CNS construction can be also adopted in other non-self-authenticating signature schemes.

References

1. Barak, B., Canetti, R., Nielsen, J.B., Pass, R.: Universally composable protocols with relaxed set-up assumptions. In: FOCS 2004, pp. 186–195. IEEE Computer Society (2004)
2. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. J. Cryptology 21(2), 149–177 (2008)
3. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
4. Boyen, X., Mei, Q., Waters, B.: Direct chosen ciphertext security from identity-based techniques. In: ACM Conference on Computer and Communications Security, pp. 320–329. ACM (2005)
5. Boyen, X., Waters, B.: Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
6. Camenisch, J., Shoup, V.: Practical Verifiable Encryption and Decryption of Discrete Logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
7. Chaum, D.: Designated Confirmer Signatures. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 86–91. Springer, Heidelberg (1995)

8. Cramer, R., Damgård, I., MacKenzie, P.: Efficient Zero-Knowledge Proofs of Knowledge without Intractability Assumptions. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 354–373. Springer, Heidelberg (2000)
9. Damgård, I.: On Σ-protocols. Course on Cryptologic Protocol Theory. Aarhus University (2010), http://www.daimi.au.dk/~ivan/Sigma.pdf
10. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inform. Theory 31, 469–472 (1985)
11. Guo, L., Wang, G., Wong, D.S., Hu, L.: Further discussions on the security of a nominative signature scheme. In: SAM 2007, pp. 566–572. CSREA Press (June 2007)
12. Huang, Q., Liu, D.Y.W., Wong, D.S.: An efficient one-move nominative signature scheme. IJACT 1(2), 133–143 (2008)
13. Huang, Z., Wang, Y.: Convertible Nominative Signatures. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 348–357. Springer, Heidelberg (2004)
14. Kim, S.J., Park, S.J., Won, D.H.: Zero-knowledge nominative signatures. In: PragoCrypt 1996, pp. 380–392 (1996)
15. Liu, D.Y.W., Chang, S., Wong, D.S.: A more efficient convertible nominative signature. In: SECRYPT 2007, pp. 214–221. INSTICC Press (2007)
16. Liu, D.Y.W., Chang, S., Wong, D.S., Mu, Y.: Nominative Signature from Ring Signature. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC 2007. LNCS, vol. 4752, pp. 396–411. Springer, Heidelberg (2007)
17. Liu, D.Y.W., Wong, D.S., Huang, X., Wang, G., Huang, Q., Mu, Y., Susilo, W.: Formal Definition and Construction of Nominative Signature. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 57–68. Springer, Heidelberg (2007)
18. Rivest, R., Shamir, A., Tauman, Y.: How to Leak a Secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)
19. 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)
20. Schnorr, C.-P.: Efficient signature generation by smart cards. J. Cryptology 4(3), 161–174 (1991)
21. Schuldt, J.C.N., Hanaoka, G.: Non-transferable User Certification Secure against Authority Information Leaks and Impersonation Attacks. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 413–430. Springer, Heidelberg (2011)
22. Smart, N.P., Vercauteren, F.: On computable isomorphisms in efficient asymmetric pairing-based systems. Discrete Applied Mathematics 155(4), 538–547 (2007)
23. Steinfeld, R., Bull, L., Wang, H., Pieprzyk, J.: Universal Designated-Verifier Signatures. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 523–542. Springer, Heidelberg (2003)
24. Susilo, W., Mu, Y.: On the Security of Nominative Signatures. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 329–335. Springer, Heidelberg (2005)
25. Wang, G., Bao, F.: Security Remarks on a Convertible Nominative Signature Scheme. In: Venter, H., Eloff, M., Labuschagne, L., Eloff, J., von Solms, R. (eds.) SEC 2007. IFIP, vol. 232, pp. 265–275. Springer, Boston (2007)
26. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

27. Zhao, W., Lin, C., Ye, D.: Provably Secure Convertible Nominative Signature Scheme. In: Yung, M., Liu, P., Lin, D. (eds.) Inscrypt 2008. LNCS, vol. 5487, pp. 23–40. Springer, Heidelberg (2009)
28. Zhao, W., Ye, D.: Pairing-Based Nominative Signatures with Selective and Universal Convertibility. In: Bao, F., Yung, M., Lin, D., Jing, J. (eds.) Inscrypt 2009. LNCS, vol. 6151, pp. 60–74. Springer, Heidelberg (2010)

A Related Work

The notion and construction of nominative signature were first proposed by Kim et al. [14] Their construction was based on Schnorr's Signature [19,20] and was later found flawed in [13]. In the construction of [14], the signer can always determine the validity of a nominative signature. Huang and Wang [13] proposed the notion of convertible nominative signature, to allow only the user to convert a signature to a publicly-verifiable one. A new scheme was proposed, but later Susilo and Mu [24] described an attack against this new scheme. Later, Guo et al. [11] showed that Susilo-Mu's attacking algorithms would still enable a signer to accept a nominative signature as a valid one even if it is actually invalid, and that the signer cannot prove the validity of a signature to any third party nor convert a nominative signature into a publicly verifiable one using Susilo-Mu's algorithms. In particular, they showed that there exists no efficient algorithm for the signer to check the validity of a published nominative signature if the signature is generated honestly by the signer in the signature generation phase and the decisional Diffie-Hellman problem is hard. In the same paper, Guo et al. described a new attack, which allows the signer of Huang and Wang's scheme to generate valid signatures on his own and show the validity of the signature to anyone without the consent of the user. Wang et al. [25] also showed that Huang-Wang nominative signature scheme is insecure under their original security model, in the sense that the signer can determine the validity of a new message-signature pair with some indirect help from the user and the signer is further able to prove the validity of nominative signatures to a third party.

Liu et al. [17] proposed the first set of formal definitions and security models for NS. The NS construction in [17] requires at least four rounds of communication between the signer and the user during signature generation. More efficient nominative signature schemes based on ring signatures [18] were proposed in [15,16]. The scheme in [15] requires two rounds in generating a nominative signature by utilizing the verifiable decryption technique [6]. The first one-move (non-interactive) NS scheme was proposed in [16]. Later, more one-move NS schemes [12,27,28] based on bilinear pairing were proposed and were proven secure in random oracle model. Schuldt et al. [21] pointed out that the scheme in [27] could not satisfy the basic invisibility requirement. In the same paper, an NS scheme relying on the standard signature scheme by Waters [26] was proposed and it is proven secure based on Decision Linear Problem (DLiP) and Discrete Logarithm Problem (DLP). However, in their scheme, the number of components for both signer and user's public keys are directly proportionally to

the number of bits of the message which is considered inflexible in a variable message length environment.

B Computational Assumptions

B.1 q-Strong Diffie-Hellman (q-SDH) Assumption

To achieve the Strong Unforgeability property in our construction, we adopt the **q-Strong Diffie-Hellman (q-SDH) Assumption** [2,3], defined as below:

Let G_1, G_2 be two cyclic groups of prime order p, respectively generated by g_1 and g_2.

The q-SDH assumption in (G_1, G_2) is defined as follows: given a $(q+3)$-tuple as input $(g_1, g_1^x, g_1^{x^2}, \cdots, g_1^{x^q}, g_2, g_2^x) \in G_1^{q+1} \times G_2^2$, output a pair $(c, g_1^{1/(x+c)})$ where $c \in \mathbb{Z}_p$ for a freely chosen value $c \in \mathbb{Z}_p \setminus \{-x\}$. An algorithm $\mathcal{A}_1$ has advantage ϵ_1 in solving q-SDH in (G_1, G_2) if

$$P[\mathcal{A}_1(g_1, g_1^x, g_1^{x^2}, \cdots, g_1^{x^q}, g_2, g_2^x) = (c, g_1^{1/(x+c)})] \geq \epsilon_1$$

where the probability is taken over the random choices of $g_1 \in G_1$ and $g_2 \in G_2$, the random choice of x in $\mathbb{Z}_p^*$, and random bits consumed by $\mathcal{A}_1$.

Definition 5. *We say that the (q, t_1, ϵ_1)-SDH assumption holds in (G_1, G_2) if no t_1-time algorithm $\mathcal{A}_1$ has advantage at least ϵ_1 in solving the q-SDH problem in (G_1, G_2).*

Remark: The q-SDH assumption applies to all G_1 and G_2 with an efficient bilinear map e, in which $G_1 = G_2$ or $G_1 \neq G_2$, and is proven in generic group model in [2].

B.2 Decisional Diffie-Hellman (DDH) Assumption

Let G_1, G_2 and G_T be cyclic groups of prime order p. Let g_1 be the generator of G_1 and g_2 be the generator of G_2. Let $e : G_1 \times G_2 \to G_T$ be an efficiently computable map. e is an asymmetric (i.e. $G_1 \neq G_2$) pairing with only efficiently-computable isomorphism $\psi : G_2 \to G_1$. Given input $(g_1, g_1^a, g_1^b, g_1^c, g_2, e)$, determine if $c \equiv ab$. An algorithm $\mathcal{A}_2$ has advantage ϵ_2 in solving DDH if

$$P[\mathcal{A}_2(g_1, g_1^a, g_1^b, g_1^c, g_2, e) = 1 \mid c \equiv ab] \geq \frac{1}{2} + \epsilon_2$$

where the probability is taken over the random choices of $g_1 \in G_1$, $g_2 \in G_2$ the random choice of a, b, c in $\mathbb{Z}_p^*$, and random bits consumed by $\mathcal{A}_2$.

Remark: In [5], the authors suggested, "a tempting workaround to this problem (ciphertext anonymity) is to use an 'asymmetric' pairing $e : G \times \hat{G} \to G_T$ in the schemes that allow it, such as Boneh and Boyen's 'BB_1' and 'BB_2', and Waters' by extension. In those schemes, and under the additional assumption

that Decision Diffie-Hellman is hard in G, one may prevent the use of the pairing as a direct test of whether a ciphertext is for a particular identity." We assume the Decisional Diffie-Hellman (DDH) assumption applies in G_1 in our scheme because there is no efficient isomorphism $\psi^{-1} : G_1 \rightarrow G_2$ so that it is not easy to find $g_2^r \leftarrow \psi^{-1}({g_1}^b)$ where $r \in \mathbb{Z}_q$ in order to allow the testing of the DH-tuple $(g_1, g_1^a, g_1^b, g_1^c)$ using bilinear map e.

Definition 6. *We say that the (t_2, ϵ_2)-DDH assumption holds in G_1 if no t_2-time algorithm $\mathcal{A}_2$ has advantage at least ϵ_2 in solving the DDH problem in G_1.*

C Security Analysis

The unforgeability (security against malicious signers/users) of our CNS scheme relies on the unforgeability property of BB's [2] standard signature scheme. Any efficient forger of the CNS signature is able to forge a valid standard signature in BB's scheme. The **Invisibility** property relies on the DDH assumption in G_1 where any efficient distinguisher D, who is able to distinguish the DH-tuple $(g_1, y_{B_3}, \alpha_1, \sigma_1/\sigma^{BB})$, can solve the DDH problem in G_1.

We now first show that our construction described above is secure against malicious users with respect to Def. 1.

Theorem 2. *Let $k \in \mathbf{N}$ be a security parameter. For the CNS proposed above, if a (t, ϵ, Q)-user can forge a valid nominative signature in* Game Unforgeability Against Users *with probability at least ϵ after running at most time t and making at most Q queries, there exists a (t', ϵ')-adversary which can forge a valid standard signature in Boneh-Boyen full signature scheme [2] with probability at least $\epsilon' = \epsilon$ after running at most time $t' = t + Qt_q + c$ where t_q is the maximum time for simulating a query and c denotes some constant time for system setup and key generation.*

Proof. Let F be a (t, ϵ, Q)-forger having user B's private key $x_B = (x_{B_1}, x_{B_2}, x_{B_3})$ (obtained by querying Corrupt). We show that F can be turned into a (t', ϵ')-algorithm S which can forge a valid standard signature in Boneh-Boyen full signature scheme [2]. Assume a public key PK_1 and PK_2 is given by the challenger ch of the Strong Existential Unforgeability Game defined in [2].

Game Simulation: S first generates param according to SystemSetup. For signer A, set, $y_{A_1} = PK_1$ and $y_{A_2} = PK_2$. Let (y_{A_1}, y_{A_2}) be the public keys and (x_{A_1}, x_{A_2}) be the private keys of signer A.

For a Sign query on input (m, y_A, y_2), where $y_A = (y_{A_1}, y_{A_2})$, and $y_2 = (y_{b_1}, y_{b_2}, y_{b_3})$, the simulation can be carried out perfectly by requesting a signature $\sigma'_q = (\sigma_q^{BB}, r_A)$ on m from ch and generates $\sigma_1 = \sigma_q^{BB} y_{b_3}^{r_1}$ where $r_1 \in_R \mathbb{Z}_p$.

Reduction: Eventually, F outputs a forged signature $(\sigma_1^*, \sigma_2^*, r_A^*, r_b^*, \alpha_1^*, \alpha_2^*)$ on m^*. A standard signature can be extracted by calculating $\sigma^{BB} = \sigma_1^*/{\alpha_1^*}^{x_{b_3}} = {g_1}^{1/(x_{A_1} + H(m^*||y_b) + x_{A_2} r_A^*)}$. S successfully forges a standard signature $(m^*||y_b, \sigma^{BB})$ and returns it to the challenger ch. Therefore, the probability $\epsilon' = \epsilon$ and the running time of S is at most $t' = t + Qt_q + c$.

Theorem 3. *Let $k \in \mathbf{N}$ be a security parameter. For the CNS proposed above, if a (t, ϵ, Q)-signer can forge a valid nominative signature in* Game Unforgeability Against Signers *with probability at least ϵ after running at most time t and making at most Q queries, there exists a (t', ϵ')-adversary which can forge a valid standard signature in Boneh-Boyen full signature scheme [2] with probability at least $\epsilon' = \epsilon$ after running at most time $t' \leq t + Qt_q + c$ where t_q is the maximum time for simulating a query and c denotes some constant time for system setup and key generation.*

Proof. We show how to construct a (t', ϵ')-algorithm S to forge a Boneh-Boyen signature from a (t, ϵ, Q)-forger F who has signer A's private keys (x_{A_1}, x_{A_2}) (obtained by querying Corrupt). Set, $y_{B_1} = PK_1$ and $y_{B_2} = PK_2$. Pick also a random $x_{B_3} \in \mathbb{Z}_p^*$ and calculate $y_{B_3} = g_1^{x_{B_3}}$. Let $(y_{B_1}, y_{B_2}, y_{B_3})$ be the public keys and $(x_{B_1}, x_{B_2}, x_{B_3})$ be the private keys of user B.

The simulation of the oracles are shown below:

- Receive: Upon receiving (m, σ^{BB}, r_a) from F for the generation of nominative signature, S requests a standard signature $\sigma_q{}' = (\sigma_q^{BB}, r_B)$ on σ_1 from ch. Then, B generates $\sigma_2 = \sigma_q^{BB} {y_{B_3}}^{r_2}$ and $\alpha_2 = g^{r_2}$ where $r_2 \in_R \mathbb{Z}_p$ and forms the nominative signature (m, σ) where $\sigma = (\sigma_1, \sigma_2, r_a, r_B, \alpha_1, \alpha_2)$.
- Convert: If (m, σ) is valid, S calculates $\delta_1 = \sigma_1 / {\alpha_1}^{x_{B_3}}$ and requests a standard signature (δ_2, r_3) on δ_1 from ch. S then returns $\delta = (\delta_1, \delta_2, r_A, r_3)$ to F. Otherwise, S returns $\perp$.
- Confirmation/Disavowal: Given a (m, σ) pair with respect to A and B, S can always carry out the proof because the witness x_{B_3} is always known to S.

Reduction: Eventually, F outputs a forged signature $(\sigma_1^*, \sigma_2^*, r_a^*, r_B^*, \alpha_1^*, \alpha_2^*)$ on m^*. A standard signature can be extracted by calculating $\sigma^{BB} = \sigma_2^* / {\alpha_2}^{* x_{B_3}}$. S successfully forges a standard signature $(\alpha_1^*, \sigma^{BB})$ and returns it to the challenger ch. The simulation can be performed perfectly. Therefore, the probability $\epsilon' = \epsilon$ and the running time of S is at most $t' = t + Qt_q + c$.

Corollary 1. *The CNS proposed above is strongly unforgeable (Def. 1 & 2) if the q-Strong Diffie-Hellman (q-SDH) problem defined in Appendix B is hard.*

Readers may refer to [2] for reviewing how a forger F of the standard signature scheme can be constructed to efficiently solve the q-Strong Diffie-Hellman (q-SDH) problem.

Theorem 4 (Invisibility). *Assume the NS proposed above is unforgeable, it has the property of invisibility (Def. 3) under Decisional Diffie-Hellman (DDH) assumption.*

Proof. We show that if there exists a distinguisher D, who has signer A's private keys (x_{A_1}, x_{A_2}) (obtained by querying Corrupt), with advantage ϵ in Game Invisibility, we can construct a DDH distinguisher D^{DDH} with advantage ϵ'. D^{DDH} is given a random DDH problem instance $(g_1, X, Y, Z) \in G_1^4$ where

$X = {g_1}^{\mu}, Y = {g_1}^{\nu}$ and determine $Z \stackrel{?}{=} {g_1}^{\mu\nu}$. Generate randomly $x_{B_1}, x_{B_2} \in \mathbb{Z}_p^*$. Calculate $y_{B_1} = {g_2}^{x_{B_1}}$ and $y_{B_2} = {g_2}^{x_{B_2}}$. Set $y_{B_3} = X$. Let $(y_{B_1}, y_{B_2}, y_{B_3})$ be the public keys and (x_{B_1}, x_{B_2}, μ) be the private keys of user B.

The simulation of the oracles are shown below:

- Receive: Upon receiving (m, σ^{BB}, r_a) from D, D^{DDH} generates $\sigma_1 = \sigma^{BB} {y_{B_3}}^{r_1}$ and $\alpha_1 = {g_1}^{r_1}$ where $r_1 \in_R \mathbb{Z}_p$, picks a random $r_B \in \mathbb{Z}_p \setminus -\{\frac{x_{B_1} + H(\sigma_1)}{x_{B_2}}\}$ and forms $\sigma_2 = {g_1}^{1/(x_{B_1} + H(\sigma_1) + x_{B_2} r_B)} {y_{B_3}}^{r_2}$ and $\alpha_2 = {g_1}^{r_2}$. The signature is (m, σ) where $\sigma = (\sigma_1, \sigma_2, r_a, r_B, \alpha_1, \alpha_2)$. D^{DDH} maintains a list:

$$L_{Receive} = \{(m, \sigma, r_1, r_2)_1, (m, \sigma, r_1, r_2)_2, \cdots, (m, \sigma, r_1, r_2)_n\}$$

  of all Receive queries.
- Convert: If (m, σ) is valid, due to the unforgeability property of the scheme, (m, σ) must be generated from a previous query. D^{DDH} looks up $L_{Receive}$, extracts r_1 and calculates $\delta_1 = \sigma_1 / {y_{B_3}}^{r_1}$ and generates a standard signature (δ_2, r_3) on δ_1. D^{DDH} then returns $\delta = (\delta_1, \delta_2, r_a, r_3)$ to D.
- Confirmation/Disavowal: Given a (m, σ) pair with respect to A and B, D^{DDH} returns $\perp$ if σ was not returned in a response to a previous Receive query, which has a negligible probability due to the unforgeability property of the scheme. Otherwise, D^{DDH} exploits the zero knowledge property of the confirmation protocol to simulates the protocol to D by using standard rewind and replay techniques. Note that the zero-knowledge protocol obtained by using the conversion by Cramer et al. [8] provides perfect zero-knowledge proof and thereby allows D^{DDH} to provide a perfect simulation.

At some point in the game, D submits a message m^* to the D^{DDH} and carry out a run of SigGen with D^{DDH}. D^{DDH} then returns a challenge nominative signature $\sigma^* = (\sigma_1^*, \sigma_2^*, r_a^*, r_B^*, Y, \alpha_2^*)$ where $\sigma_1^* = {g_1}^{1/(x_{a_1} + H(m^* || y_B) + x_{a_2} {r_a}^*)} Z$, $\sigma_2^* = {g_1}^{1/(x_{B_1} + H(\sigma_1^*) + x_{B_2} {r_B}^*)} X^{r_2}$, and r_a^* corresponds to the value given by D in the protocol run of SigGen. Note that our simulation is indistinguishable from the simulation defined in Game Invisibility. The reason is that r_2, r_B^* are random in $\mathbb{Z}_p$ and $X, Y, Z \in G_1$ are elements from the random DDH problem instance. Thus, we have $\sigma_1^*, \sigma_2^*, \alpha_1^*, \alpha_2^* \in_R G_1$, $r_B^* \in_R \mathbb{Z}_p$. Therefore, σ^* is indistinguishable from the challenge signature generated by NSSim defined in Game Invisibility. For the event that if D distinguishes the validity of σ^* successfully, so does D^{DDH} on solving the DDH problem instance. Hence D^{DDH} has a success probability of at least $\epsilon' = \epsilon$.

Similar to the evaluation of Theorem 3, the running time of D^{DDH} is at most $t' = t + Qt_q + c$.

Corollary 2 (User-only Conversion). *The CNS proposed has the property of invisibility. By Theorem 1, the CNS has also the property of user-only conversion.*

Efficient and Random Oracle-Free Conditionally Anonymous Ring Signature

Shengke Zeng[1,2,3], Zhiguang Qin[1,2], Qing Lu[1,2], and Qinyi Li[1,2]

[1] School of Computer Science and Engineering,
University of Electronic Science and Technology of China, Chengdu 611731, China
[2] Network and Data Security Key Laboratory of Sichuan Province
[3] Shanghai Key Laboratory of Integrate Administration Technologies
for Information Security
zengshengke@gmail.com

Abstract. Compared to conventional ring signature schemes, conditionally anonymous ring signatures allow to revoke the anonymity of actual signer without the group manager's help if necessary. When the actual signer intends to confirm his role, it can be proved by a confirmation algorithm. In addition, any user, who is in a ring but not signer, can claim this through a disavowal algorithm. There were several proposals which were proved secure in random oracles. In other words, the security of such schemes depends on the randomness of hash functions. Recently, Zeng et al. proposed a generic construction of conditionally anonymous ring signature scheme without random oracles. Their scheme relies on NIZKs and pseudorandom functions, which render it to be inefficient. This paper proposes a practical ring signature scheme with traceability without random oracles in the common reference string model.

Keywords: Conditional Anonymity, Ring Signature, Standard Model, NIWI Proof System.

1 Introduction

In traditional ring signature scheme [9], the identity of the actual signer is unconditionally anonymous. In other words, even if all the private keys of a set are revealed, we still cannot determine who is the signer of a given signature. This property is intended to protect the privacy of the signer. However, when the generated ring signature is disputed, it is impossible to identify who should be responsible for it. The notion of group signature [5] can avoid this problem since there exists a group manager who can revoke the identity of the signer by using a trapdoor. However this kind of construction does not fit for an ad hoc manner.

In conditionally anonymous ring signature, contrast to group signature, every node within the ring is equal and there is no special node as group manger. Different from conventional ring signature, conditionally anonymous ring signature has advanced properties: *confirmation* and *disavowal*. By those means, an actual

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 21–34, 2012.

signer can be traced without group manager's help. Specifically, the signer confirms the fact of signing through a confirmation protocol and non-signer refutes that through a disavowal protocol.

Komano et al. [8] firstly proposed a construction of conditionally anonymous ring signature, though their confirmation protocol and disavowal protocol are interactive. Wu et al. [11] proposed a notion of Ad Hoc Group Signature, which essentially is a conditionally anonymous ring signature. their *self-traceability* algorithm is non-interactive and the signature size is constant. However, the security of [11] is based on a new assumption (called DFDH assumption). Under the security model of [8], Zeng et al. proposed a new efficient conditionally anonymous ring signature scheme [12]. The confirmation/disavowal algorithms in [12] are both non-interactive and have constant costs, and the security is based on standard assumption (DBDH assumption). However, compared to [11], the size of the ring signature in [12] is proportional to the size of ring.

All those conditionally anonymous ring signature schemes are provably secure in the random oracle model, in other words, security of such schemes depends on the randomness of hash functions. As argued in [3], cryptographic protocols constructed under the random oracle models are not secure in the real world. Recently, Zeng et al. proposed a generic construction of conditionally anonymous ring signature scheme without random oracles. Their scheme relies on NIZKs and pseudorandom functions, which makes it inefficient. Constructing an efficient and practical conditionally anonymous ring signature without random oracles is necessary.

Our Contribution
We propose a practical conditionally anonymous ring signature scheme without random oracles under the security model of [8] and [12]. We make use of Non-interactive Witness-indistinguishable (NIWI) proof system [6,7] and sub-linear size ring signature scheme[4] to construct the conditionally anonymous ring signature scheme. The genuine signer can confirm his role through confirmation protocol, while non-signers can refute it through disavowal protocol. Both the confirmation and disavowal algorithms are non-interactive. In addition, the signature size is $\mathcal{O}(k\sqrt{N})$[4], which is more efficient than [8] and [12], where k is a security parameter and N is the number of ring members.

2 Preliminaries

In this section, we introduce the mathematical setting and the complexity assumptions which will be used in the paper.

2.1 Bilinear Groups of Composite Order

We use bilinear groups of composite order $n = pq$, where p and q are primes, introduced by Boneh, Goh and Nissim (BGN)[2]. In this setting, G is a multiplicative cyclic group of order n; G_p and G_q are the subgroups of G with order

p and q respectively; g is the generator of G; h is the generator of G_q; G_T is a multiplicative group of order n; $\hat{e} : G \times G \to G_T$ is an efficiently computable map with the following properties:

- *Bilinearity.* $\forall\ u, v \in G, \forall\ a, b \in Z$, $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
- *Non-degeneracy.* $\hat{e}(g, g)$ is a generator of G_T whenever g is a generator of G.
- *Computability.* $\forall\ u, v \in G$, $\hat{e}(u, v)$ can be computed efficiently.
- $G_{T,p}$ and $G_{T,q}$ are the G_T-subgroups of order p and q, respectively.

2.2 Complexity Assumptions

Definition 1. *[Subgroup Decision (BGN) [2]] Given the description of G without the factorization of n, and r selected randomly either from G or from G_q, decide whether r is in G_q.*

Definition 2. *[Strong Diffie-Hellman Assumption in G_p [1]] Given $\eta, \eta^x, \eta^{x^2}, \cdots, \eta^{x^t} \in G_p$ as input, output a pair $(c, \eta^{1/(x+c)})$, where $c \in Z_p$.*

2.3 Underlying Signature

Our proposal takes the fully secure BB signature scheme [1] as underlying signature, which is secure against the strong existential forgery under an adaptive chosen message attack provided that SDH assumption is hard (Definition 2). This signature scheme is secure in G_p and also adapted for composite order groups [4]. The original BB signature scheme is described in the setting of *asymmetric* paring $\hat{e}$: $G_1 \times G_2 \to G_T$. It is trivial to replace it as the *symmetric* paring setting. Let us review the fully secure BB signature scheme under the symmetric paring version:

KeyGen. Choose $\eta \in G_p$ and $x, y \in Z_p$. The public key is the tuple (η, η^x, η^y). The secret key is (x, y).

Signing. Given the secret key (x, y) and a message $m \in Z_p$, pick $r \leftarrow Z_p \backslash \{-\frac{x+m}{y}\}$, compute $\sigma = \eta^{\frac{1}{x+yr+m}} \in G_p$. The signature is the pair (σ, r).

Verification. Check the equation $\hat{e}(\sigma, \eta^x \cdot (\eta^y)^r \cdot \eta^m) = \hat{e}(\eta, \eta)$ holds or not.

In our ring signature scheme, we adapt this fully secure BB signature under the symmetric paring and composite order groups.

2.4 Non-interactive Witness-Indistinguishable Proof

Proof system allows a prover P to convince a verifier V truth of the statement. For an efficiently computable binary relation R, if $(x, \omega) \in R$, where x is the statement and ω is the witness for this statement, we let $\mathcal{L}$ be the language consisting of statements in R, then a non-interactive proof system (P, V) for $\mathcal{L}$ satisfies the following properties:

- Completeness. For any common reference string crs, $V_{crs}(x, P_{crs}(x, \omega)) = 1$ holds always.
- Adaptive Soundness. For any adversary $\mathcal{A}$, it holds that

$$\Pr[V_{crs}(x, \pi) = 1 : (x, \pi) \leftarrow \mathcal{A}(crs)] = 0$$

If (P, V) is non-interactive witness-indistinguishable proof system, the additional witness-indistinguishability for secure parameter k should be satisfied:

$$\begin{aligned}&\Pr[(x, \omega_0, \omega_1) \leftarrow \mathcal{A}(1^k); \pi \leftarrow P(1^k, x, \omega_0) : \mathcal{A}(\pi) = 1 \wedge (x, \omega_0), (x, \omega_1) \in R] \\ \approx &\Pr[(x, \omega_0, \omega_1) \leftarrow \mathcal{A}(1^k); \pi \leftarrow P(1^k, x, \omega_1) : \mathcal{A}(\pi) = 1 \wedge (x, \omega_0), (x, \omega_1) \in R]\end{aligned}$$

Witness-indistinguishability means that proof does not reveal which witness the prover used.

2.5 NIWI Proofs for Commitment Scheme

Now we intend to use NIWI proofs to prove the committed values satisfying an equation, such as BB signature's verification equation: $\hat{e}(\sigma, g^x \cdot g^m) = \hat{e}(g, g)$ (Here we take the weak BB signature scheme for composite order groups for example). When h has order n, we obtain witness-indistinguishability. When h has order q, the proof has soundness in G_p. The first step is to commit the variables σ and the verification key g^x. The commitments are

$$C = \sigma h^{r_1}, \qquad L = g^x h^{r_2}$$

The prover's goal is to convince the verifier that the commitments contain σ and the verification key g^x which satisfy the BB verification equation. We plug in the commitments in place of the variables in the BB verification equation:

$$\begin{aligned}\hat{e}(C, L \cdot g^m) = \hat{e}(\sigma h^{r_1}, g^x h^{r_2} g^m) &= \hat{e}(\sigma, g^x g^m)\hat{e}(\sigma, h^{r_2})\hat{e}(g^{x+m}.h^{r_1})\hat{e}(h^{r_1}, h^{r_2}) \\ &= \hat{e}(g, g)\hat{e}(\sigma, h^{r_2})\hat{e}(g^{x+m}.h^{r_1})\hat{e}(h^{r_1}, h^{r_2}) \\ &= \hat{e}(g, g)\hat{e}(\sigma^{r_2} \cdot g^{(x+m)r_1} \cdot h^{r_1 r_2}, h)\end{aligned}$$

Then we regard $\sigma^{r_2} \cdot g^{(x+m)r_1} \cdot h^{r_1 r_2}$ as prover's NIWI proof π. It is easy to see that this proof would be convincing in the soundness setting, where h has order q, because in that setting, we have the soundness key λ, $\lambda = 1 \bmod p$ and $\lambda = 0 \bmod q$. Then the verifier would know but not be able to compute

$$\hat{e}(C^\lambda, L^\lambda g^m) = \hat{e}(g, g)\hat{e}(\pi, h)^\lambda = \hat{e}(g, g)$$

This gives us soundness, since $\sigma = C^\lambda$, $g^x = L^\lambda$ satisfy the verification equation of BB signature. This proof also satisfies the witness-indistinguishability. In the witness-indistinguishability setting, where h has order n, the commitments are perfect hiding. Therefore, in the verification equation, nothing except for π has any information about σ and g^x. For $\pi = \sigma^{r_2} \cdot g^{(x+m)r_1} \cdot h^{r_1 r_2}$, since r_1 and r_2 are random values, any witness can satisfy the proof π.

3 Model of Conditionally Anonymous Ring Signature

3.1 Syntax

Definition. A conditionally anonymous ring signature consists of the following algorithms. Let the universe of members $\mathcal{U} = \{1, \cdots, \chi\}$.

1. A probabilistic key generation algorithm $\mathcal{K}$, given a security parameter s, outputs public key and private key $(\mathrm{pk_i}, \mathrm{sk_i})$ for member i.
2. A probabilistic signing algorithm $\mathcal{S}$, given a message m, a private key $\mathrm{sk_k}$ of signer k, and the public keys $\mathrm{pk_1}, \cdots, \mathrm{pk_n}$ of set $\mathrm{R} = \{\mathrm{u_1}, \cdots, \mathrm{u_n}\}$, outputs a tuple $(\mathrm{m}, \sigma, \mathrm{R})$. For simplicity, we do not distinguish i and its public key $\mathrm{pk_i}$.
3. A deterministic verification algorithm $\mathcal{V}$, given $(\mathrm{m}, \sigma, \mathrm{R})$, determines whether σ is a valid ring signature w.r.t. (m, R).
4. A confirmation protocol $\mathcal{C}$, executed between a signer k and a verifier with a common input $(\mathrm{m}, \mathrm{R}, \sigma, \mathrm{k})$. A signer also inputs his secret key $\mathrm{sk_k}$. Finally, a verifier either accepts or rejects signer's confirmation.
5. A disavowal protocol $\mathcal{D}$, executed between a member $\mathrm{i} \in \mathrm{R}$ and verifier with common input$(\mathrm{m}, \sigma, \mathrm{R}, \mathrm{i})$. The member i also has $\mathrm{sk_i}$ as his secret input. Finally, a verifier either accepts or rejects the disavowal of member i.

3.2 Oracles

In this subsection, we introduce some oracles utilized in the security models.

$\mathcal{O}_{sig}(i, m, R)$. This is the *signing oracle* and holds $i \in R$. Upon this, a ring signature σ w.r.t. (m, R) using sk_i is returned.
$\mathcal{O}_{cor}(i)$. This is the *corruption oracle*. Upon this, the secret key sk_i of member i is returned.
$\mathcal{O}_{C/D}(i, m, \sigma, R)$. This is the *confirmation/disavowal oracle*. Upon this, the oracle takes sk_i as the secret input to interact with the verifier to confirm or disavow that σ is generated by sk_i w.r.t. (m, R).

3.3 Security Model

The security of a conditionally anonymous ring signature scheme is formulated in four properties: *anonymity*, *unforgeability*, *traceability* and *non-frameability*. They are now introduced as follows.

Anonymity. Anonymity essentially means that given a signature no one can tell who is the actual signer. Formally, for any distinguisher $\mathcal{D}$, consider the following game (denoted by an *anonymity game*):

- Initially, $\mathcal{D}$ receives pk_i for all $i \in \mathcal{U}$.
- $\mathcal{D}$ can query oracles $\mathcal{O}_{sig}$, $\mathcal{O}_{cor}$, $\mathcal{O}_{C/D}$ adaptively and receive the answer properly.

- $\mathcal{D}$ outputs $(m^*, pk_{i_0}, pk_{i_1}, R^*)$ for $(i_0, i_1) \in R^*$ as his challenge. In turn, the challenger takes $b \leftarrow \{0,1\}$, and uses sk_{i_b} to generate a challenge ring signature σ^* on (m^*, R^*) for $\mathcal{D}$.
- $\mathcal{D}$ can continue to query oracles $\mathcal{O}_{sig}$, $\mathcal{O}_{cor}$, $\mathcal{O}_{C/D}$. Specially, he cannot request $\mathcal{O}_{C/D}(i_b, m^*, R^*, \sigma)$ for any σ and i_b cannot be corrupted.

At the end of game, $\mathcal{D}$ generates a guess bit b' for b. Denote $\mathsf{Succ}^{anon}(\mathcal{D})$ the success event of $\mathcal{D}$ in the game. Define $Adv^{anon}_{\mathcal{D}}(s) = |\Pr[\mathsf{Succ}^{anon}(\mathcal{D})] - \frac{1}{2}|$. A ring signature is conditionally anonymous if for any probabilistic polynomial time distinguisher $\mathcal{D}$, $Adv^{anon}_{\mathcal{D}}(s)$ is negligible in security parameter s.

Unforgeability. A conditionally anonymous ring signature is unforgeable if it is infeasible for any forger to forge a signature on uncorrupted R^*. Formally, for any forger $\mathcal{F}$, consider the following game (*unforgeability game*):

- Initially, $\mathcal{F}$ receives pk_i for all $i \in \mathcal{U}$.
- $\mathcal{F}$ can query oracles $\mathcal{O}_{sig}$, $\mathcal{O}_{cor}$, $\mathcal{O}_{C/D}$ adaptively and receive the answer properly.

At the end of game, $\mathcal{F}$ generates a forgery (m^*, R^*, σ^*). $\mathcal{F}$ succeeds if (m^*, R^*, σ^*) passes the signature verification while (m^*, R^*) was never queried to $\mathcal{O}_{sig}$ oracle and no $i \in R^*$ is corrupted. Denote the success probability of $\mathcal{F}$ by $\Pr[\mathsf{Succ}^{uf}(\mathcal{F})]$. A conditionally anonymous ring signature scheme is unforgeable, if for any probabilistic polynomial time forger $\mathcal{F}$, $\Pr[\mathsf{Succ}^{uf}(\mathcal{F})]$ is negligible.

Traceability. Traceability describes that for any valid ring signature, it is impossible that any member of its ring R can deny generating it. Formally, for an adversary $\mathcal{A}$, consider the following game (called *traceability game*):

- Initially, $\mathcal{A}$ receives pk_i for all $i \in \mathcal{U}$.
- $\mathcal{A}$ can query oracles $\mathcal{O}_{sig}$, $\mathcal{O}_{cor}$, $\mathcal{O}_{C/D}$ adaptively and receive the answer properly.

At the end of game, $\mathcal{A}$ outputs a signature (m, σ, R) and plays the role of each $j \in R$ to execute the disavowal protocol with the challenger. $\mathcal{A}$ succeeds if the challenger is convinced of the disavowal for **all** $j \in R$. $\Pr[\mathsf{Succ}^{tr}(\mathcal{A})]$ denote the success probability of $\mathcal{A}$. A conditionally anonymous ring signature scheme is traceable, if for any probabilistic polynomial time adversary $\mathcal{A}$, $\Pr[\mathsf{Succ}^{tr}(\mathcal{A})]$ is negligible.

Non-frameability. Non-frameability represents that if one did not generate a signature, then he should be able to prove this using a disavowal protocol. Formally, consider the *non-frameability game* below:

- Initially, $\mathcal{A}$ receives pk_i for all $i \in \mathcal{U}$.
- $\mathcal{A}$ can query oracles $\mathcal{O}_{sig}$, $\mathcal{O}_{cor}$, $\mathcal{O}_{C/D}$ adaptively and receive the answer properly.

At the end of game, $\mathcal{A}$ outputs a valid signature (m^*, σ^*, R^*) and *uncorrupted* $j \in R^*$ such that (j, m^*, R^*) was never queried to $\mathcal{O}_{sig}$ oracle. Then the challenger uses sk_j to execute the disavowal protocol with $\mathcal{A}$. $\mathcal{A}$ succeeds if the challenger fails to disavow. Let $\mathsf{Succ}^{nf}(\mathcal{A})$ denote the success event of $\mathcal{A}$. A conditionally anonymous ring signature is non-frameable if for any probabilistic polynomial time attacker $\mathcal{A}$, $\Pr[\mathsf{Succ}^{nf}(\mathcal{A})]$ is negligible.

3.4 High Level

We give the description of this conditionally anonymous ring signature scheme firstly. Consider a ring $R = \{pk_1, \cdots, pk_n\}$, a signer k knows his private key sk_k corresponding to one of the public keys in R and wants to sign a message m.

The actual signer k uses private key $sk_k = (x_k, y_k)$ to generate a fully secure BB signature on m [1]: $\sigma = g^{\frac{1}{x_k + y_k r + m}}$. σ here is regarded as the partial signature. We hope the ring signature can be traced anyway, therefore, we cannot make a BGN encryption [2] to σ. However, we do not hope σ is publicly verifiable, thus the public key pk_i of member i cannot be set as BB signature fashion: $pk_i = (g^{x_i}, g^{y_i})$. Instead, it should be BGN commitment [2]: $pk_i = (g^{x_i}\hat{h}^{e_i}, g^{y_i}\hat{h}^{d_i})$. Therefore, for a given signature σ and public key pk_k, it is impossible to ensure the consistency between σ and pk_k. Then we commit the verification key pk_k of the signer, and make use of the NIWI proofs [6,7] to prove the the partial signature σ and the committed verification key belong to a language $\mathcal{L}_{BB}$. Finally, we adopt the sub-linear size ring signature algorithm [4] to prove the committed pk_k belongs to ring R without revealing which one.

For the traceability algorithm, signer k uses the witness sk_k and the common input σ to generate a confirmation proof π_c, which is used to prove k is the signer of a given ring signature σ. The non-signer i, also can make a disavowal proof π_d to prove he is not the signer of σ using the witness sk_i. Both of the two protocols are non-interactive.

3.5 Construction

Common Reference String. Run BGN generator [2] to get $(G, G_T, n, p, q, \hat{e}, g, h, \hat{h}, \tilde{h})$, where G is a group with order n and g is the random generator of G. $n = pq$, p and q are primes, G_p and G_q are subgroups of G with order p and q respectively: $G = G_p \times G_q$. $h, \hat{h}, \tilde{h}$ are random generators of G_q. The bilinear map $\hat{e} : G \times G \to G_T$. H is the collision free hash function, $H : \{0,1\}^* \to Z_n$. The CRS string is $crs = (n, G, G_T, \hat{e}, g, h, \hat{h}, \tilde{h}, H)$.

Key Generation. Member i chooses $x_i, y_i, e_i, d_i \in Z_n$ as his private key and the corresponding public key $pk_i = (g^{x_i}\hat{h}^{e_i}, g^{y_i}\hat{h}^{d_i})$.

Signing. Suppose a signer k chooses a ring $R = \{pk_1, \cdots, pk_N\}$ and a message m, he generates the ring signature as follows:

1. Compute the hash value: $\tau = H(m, R)$.
2. Generate a partial signature: choose $r \leftarrow Z_n$, compute $\sigma = g^{\frac{1}{x_k + y_k r + \tau}}$. This is a fully secure BB signature [1].
3. Generate a proof $\pi_\sigma = \hat{h}^{\frac{1}{x_k + y_k r + \tau}}$. π_σ is a proof that the signer cannot make another σ', such that $\sigma \neq \sigma' \in G$ but $(\sigma)^q = (\sigma')^q$.
4. Choose $r_1, r_2, r_3 \in Z_n$, compute h^{r_1} and commit pk_k with r_2, r_3: $C_1 = g^{x_k} \hat{h}^{e_k} h^{r_2}$, $C_2 = g^{y_k} \hat{h}^{d_k} h^{r_3}$.
5. Generate NIWI proofs

$$\pi_1 = g^{\frac{e_k + d_k r}{x_k + y_k r + \tau}} h^{r_1(e_k + d_k r)}, \ \pi_2 = g^{\frac{r_2 + r_3 r}{x_k + y_k r + \tau} + (x_k + y_k r + \tau) r_1} h^{r_1(r_2 + r_3 r)}.$$

   These NIWI proofs are used to prove $(\sigma h^{r_1}, \tau, C_1, C_2) \in \mathcal{L}_{BB}$.
6. Generate a NIWI proof π to prove the verification key pk_k in the commitment (C_1, C_2) belongs to the ring R using the sub-linear ring signature algorithm proposed in [4].
7. Publish the ring signature $\Sigma = (R, r, \sigma, h^{r_1}, \pi_\sigma, \{C_1, C_2\}, \pi_1, \pi_2, \pi)$.

Verification. The verifier does as follows:

1. Compute $\tau = H(m, R)$.
2. Reject if $\hat{e}(\sigma, \hat{h}) \neq \hat{e}(g, \pi_\sigma)$.
3. Reject if $\hat{e}(\sigma \cdot h^{r_1}, C_1 \cdot C_2^r \cdot g^\tau) \neq \hat{e}(g, g) \cdot \hat{e}(\pi_1, \hat{h}) \cdot \hat{e}(\pi_2, h)$.
4. Accept if the proof π is valid.

Confirmation. The real signer, i.e., member k, wants to confirm a ring signature Σ is signed by him, he acts as follows:

1. Generate a proof $\pi_c = g^{\frac{e_k + d_k r}{x_k + y_k r + \tau}}$.
2. Publish π_c to the verifier.

The verifier uses member k's public key $pk_k = (g^{x_k} \hat{h}^{e_k}, g^{y_k} \hat{h}^{d_k})$ to check $\hat{e}(\sigma, g^{x_k} \hat{h}^{e_k} \cdot (g^{y_k} \hat{h}^{d_k})^r \cdot g^\tau) = \hat{e}(g, g) \hat{e}(\pi_c, \hat{h})$. If the equation holds, one convinces that member k is the actual signer of Σ.

Disavowal. Non-signer, i.e., member j, can make use of this algorithm to prove he is not the actual signer of Σ w.r.t. the same (m, r, R):

1. Generate $\sigma_j = g^{\frac{1}{x_j + y_j r + \tau}}$.
2. Generate a proof $\pi_{\sigma_j} = \tilde{h}^{\frac{1}{x_j + y_j r + \tau}}$
3. Generate a proof $\pi_d = g^{\frac{e_j + d_j r}{x_j + y_j r + \tau}}$.
4. Publish $(\sigma_j, \pi_{\sigma_j}, \pi_d)$.

The verifier uses member j's public key $pk_j = (g^{x_j}\hat{h}^{e_j}, g^{y_j}\hat{h}^{d_j})$ to check the following steps. If all are accepted, he is convinced that member j is not the signer of Σ.

1. Reject if $\hat{e}(\sigma_j, \tilde{h}) \neq \hat{e}(\pi_{\sigma_j}, g)$.
2. Reject if $\hat{e}(\sigma_j, g^{x_j}\hat{h}^{e_j} \cdot (g^{y_j}\hat{h}^{d_j})^r \cdot g^{\tau}) \neq \hat{e}(g, g) \cdot \hat{e}(\pi_d, \hat{h})$.
3. Accept if $\sigma_j \neq \sigma$.

Remark 1. In both constructions of Shacham et al.[10] and Chandran et al. [4], signer should commit the underlying signature, i.e. BB signature and the verification key. Since the signature is committed (by the randomness), it cannot be used for disavowal. In order to enforce the *confirmation* and *disavowal*, we cannot commit the BB signature. However, it violates the anonymity, since the BB signature is publicly verifiable. Therefore, we modify the original public key setting of BB signature as $pk = (g^x\hat{h}^e, g^y\hat{h}^d)$. It is easy to see that, for a given BB signature $\sigma = g^{\frac{1}{x+yr+m}}$, no one can check the consistency between σ and pk. In this way, we commit the verification key pk only. Therefore, the property of disavowal can be achieved by checking $\sigma_j \stackrel{?}{=} \sigma$.

Remark 2. For a malicious signer, for example, the signer computes $\sigma' = \sigma h^r$, then he also can provide valid NIWI proofs to ensure validity of the ring signature Σ. Therefore, the malicious signer can disavow his signing successfully. In order to prevent this modification by malicious signer, the step 3 in the Signing algorithm is necessary. In this way, the malicious signer cannot make another σ' satisfying that $\sigma' \neq \sigma$ but $(\sigma')^q = (\sigma)^q$.

4 Security

A conditionally anonymous ring signature scheme is *secure* if it satisfies the properties of *anonymity*, *unforgeability*, *traceability* and *non-frameability*.

Anonymity. Informally, anonymity holds if for a given ring signature $\Sigma^* = (R^*, r, \sigma, h^{r_1}, \pi_\sigma, \{C_1, C_2\}, \pi_1, \pi_2, \pi)$, no one can tell who is the actual signer, even though he can access $\mathcal{O}_{C/D}$, $\mathcal{O}_{sig}$ and $\mathcal{O}_{cor}$. Specially, the challenge users $(i_0, i_1) \in R$ are not corrupted and the queries $\mathcal{O}_{C/D}(i_b, m^*, \Sigma, R^*)$ for $b = 0, 1$ are not issued.

Theorem 1. *The proposed scheme is conditionally anonymous if the subgroup decision assumption holds.*

Proof. We consider two experiments E_0 and E_1 in the anonymity game.

$E_1(1^s)$ (Actual Adversary Experiment where h Has Order q)

CRS : Get the CRS string $crs = (n, G, G_T, \hat{e}, g, h, \hat{h}, \tilde{h}, H)$
$KGen$: Choose $x_i, y_i, e_i, d_i \in Z_n$, set $sk_i = (x_i, y_i, e_i, d_i)$, $pk_i = (g^{x_i}\hat{h}^{e_i}, g^{y_i}\hat{h}^{d_i})$
$O_{cor}(i)$: Return sk_i
$O_{sig}(i, m, R)$: Compute $\tau = H(m, R)$
Choose $r, r_1, r_2, r_3 \leftarrow Z_n$, compute $\sigma = g^{\frac{1}{x_i+y_i r+\tau}}$, $\pi_\tau = \hat{h}^{\frac{1}{x_i+y_i r+\tau}}$
h^{r_1}, $C_1 = g^{x_i}\hat{h}^{e_i} \cdot h^{r_2}$, $C_2 = g^{y_i}\hat{h}^{d_i} \cdot h^{r_3}$
Compute $\pi_1 = g^{\frac{e_i+d_i r}{x_i+y_i r+\tau}} h^{r_1(e_i+d_i r)}$,
$\pi_2 = g^{\frac{r_2+r_3 r}{x_i+y_i r+\tau}+(x_i+y_i r+\tau)r_1} h^{r_1(r_2+r_3 r)}$
Generate a NIWI proof π to prove $\{C_1, C_2\}$ is consistent with one public key in ring R
Return $\Sigma = (R, r, \sigma, h^{r_1}, \pi_\sigma, \{C_1, C_2\}, \pi_1, \pi_2, \pi)$
$O_{C/D}(i, m, \sigma, R)$: Return $\sigma_i = g^{\frac{1}{x_i+y_i r+\tau}}$, $\pi_{c/d} = g^{\frac{e_i+d_i r}{x_i+y_i r+\tau}}$, $\pi_{\sigma_i} = \tilde{h}^{\frac{1}{x_i+y_i r+\tau}}$
$Challenge$: Fix i_0, i_1 from R^* and choose $b \leftarrow \{0, 1\}$, generate Σ^* using sk_{i_b}
Let b' be the output of adversary
Return true iff $b' = b$

Let $\Pr[E_0(1^s)]$ be the advantage of adversary $\mathcal{D}$ in E_0, $\Pr[E_0(1^s)] = p_0(s) = |Succ^{anon}(\mathcal{D}) - \frac{1}{2}|$. We have to show $p_0(s)$ is negligible.

We now consider the second experiment, where we change h such that h has order n.

$E_1(1^s)$ (h has order n)

CRS : Get the CRS string $crs = (n, G, G_T, \hat{e}, g, h, \hat{h}, \tilde{h}, H)$
$KGen$: Choose $x_i, y_i, e_i, d_i \in Z_n$, set $sk_i = (x_i, y_i, e_i, d_i)$, $pk_i = (g^{x_i}\hat{h}^{e_i}, g^{y_i}\hat{h}^{d_i})$
$O_{cor}(i)$: Return sk_i
$O_{sig}(i, m, R)$: Compute $\tau = H(m, R)$
Choose $r, r_1, r_2, r_3 \leftarrow Z_n$, compute $\sigma = g^{\frac{1}{x_i+y_i r+\tau}}$, $\pi_\tau = \hat{h}^{\frac{1}{x_i+y_i r+\tau}}$
h^{r_1}, $C_1 = g^{x_i}\hat{h}^{e_i} \cdot h^{r_2}$, $C_2 = g^{y_i}\hat{h}^{d_i} \cdot h^{r_3}$
Compute $\pi_1 = g^{\frac{e_i+d_i r}{x_i+y_i r+\tau}} h^{r_1(e_i+d_i r)}$,
$\pi_2 = g^{\frac{r_2+r_3 r}{x_i+y_i r+\tau}+(x_i+y_i r+\tau)r_1} h^{r_1(r_2+r_3 r)}$
Generate a NIWI proof π to prove $\{C_1, C_2\}$ is consistent with one public key in ring R
Return $\Sigma = (R, r, \sigma, h^{r_1}, \pi_\sigma, \{C_1, C_2\}, \pi_1, \pi_2, \pi)$
$O_{C/D}(i, m, \sigma, R)$: Return $\sigma_i = g^{\frac{1}{x_i+y_i r+\tau}}$, $\pi_{c/d} = g^{\frac{e_i+d_i r}{x_i+y_i r+\tau}}$, $\pi_{\sigma_i} = \tilde{h}^{\frac{1}{x_i+y_i r+\tau}}$
$Challenge$: Fix i_0, i_1 from R^* and choose $b \leftarrow \{0, 1\}$, generate Σ^* using sk_{i_b}
Let b' be the output of adversary
Return true iff $b' = b$

Now, let $p_1(s) = \Pr[E_1(1^s)] = |Succ^{anon}(\mathcal{D}) - \frac{1}{2}|$. We claim $|p_0(s) - p_1(s)| = negl(s)$, otherwise, there exists a distinguisher $\mathcal{D}'$ to tell $h \leftarrow G_q$ from $h \leftarrow G$, which breaks the subgroup decision assumption.

When h has order n, the commitment is perfectly hiding. Therefore, the adversary $\mathcal{D}$ cannot gain any information from the commitments using h in E_1. Since $\sigma^* = g^{\frac{1}{x_{i_b}+y_{i_b}r^*+\tau^*}}$ is generated by the partial secret key (x_i, y_i) of member i_b, no one can check the consistency between σ^* and pk_{i_b}. π is a secure NIWI proof for the committed pk_{i_b}, which was proven in [4]. Therefore, $\mathcal{D}$'s advantage in E_1 is negligible, which means $p_1(s) = negl(s)$. And that $|p_0(s) - p_1(s)| = negl(s)$, thus $p_0(s) = negl(s)$, completes this proof. □

Unforgeability. Unforgeability states that there is no forger $\mathcal{F}$ who can generate a valid ring signature Σ^* w.r.t. the challenge (R^*, m^*) even if $\mathcal{F}$ can access to $\mathcal{O}_{cor}, \mathcal{O}_{sig}, \mathcal{O}_{C/D}$, assuming that (R^*, m^*) is not queried to $\mathcal{O}_{sig}$ and no $i \in R^*$ is corrupted.

Theorem 2. *This proposal is unforgeable if the BB signature is unforgeable.*

Proof. In the unforgeability game, we reduce unforgeability of our ring signature scheme to the BB signature (the stronger version)[1]. We assume $\mathcal{F}$ is the forger of our ring signature scheme. $\mathcal{B}$ is $\mathcal{F}$'s challenger, who tries to forge the full version of BB signature. $\mathcal{C}$ is $\mathcal{B}$'s challenger of BB signature scheme.

$\mathcal{C}$ gives the challenge parameters of BB signature to $\mathcal{B}$: Group G and its order n with n's factorization $n = pq$; the generators η of G_p and h of G_q; BB signature public key (η_1, η_2) in G_p.

Setup. $\mathcal{B}$ picks $r_1, \hat{u}, \tilde{u} \leftarrow Z_q^*$, sets $g = \eta h^{r_1}$, $\hat{h} = h^{\hat{u}}$, $\tilde{h} = h^{\tilde{u}}$. Then, $\mathcal{B}$ generates user keys: Firstly, he picks $i^* \leftarrow \{1, 2, \cdots, \chi\}$. For each $i \neq i^*$, he picks $x_i, y_i \leftarrow Z_n$ and $e_i, d_i \leftarrow Z_q$, sets $pk_i = (g^{x_i}\hat{h}^{e_i}, g^{y_i}\hat{h}^{d_i})$ and $sk_i = (x_i, y_i, e_i, d_i)$. For $i = i^*$, $\mathcal{B}$ picks $r_2, r_3 \leftarrow Z_q$ and sets $pk_{i^*} = (\eta_1\hat{h}^{r_2}, \eta_2\hat{h}^{r_3})$. Note that $\mathcal{B}$ does not know the private key sk_{i^*} of member i^*, and his goal is to forge BB signature of user i^*.

Now $\mathcal{B}$ runs $\mathcal{F}$ with providing the following values: $(n, G, G_T, \hat{e}, g, h, H, \hat{h}, \tilde{h})$, and the public keys $\{pk_i\}_{i=1}^{\chi}$. $\mathcal{F}$ makes the following queries:

Query on $\mathcal{O}_{cor}(i)$: If $i = i^*$, $\mathcal{B}$ terminates with Fail; otherwise, $\mathcal{B}$ returns (x_i, y_i, e_i, d_i) to $\mathcal{F}$.

Query on $\mathcal{O}_{sig}(i, m, R)$: If $i \neq i^*$, $\mathcal{B}$ can use the private key (x_i, y_i, e_i, d_i) of member i to generate the ring signature normally; If $i = i^*$, $\mathcal{B}$ needs to make the signing query of BB signature in G_p from his challenger $\mathcal{C}$. $\mathcal{C}$ first computes $\tau = H(m, R) \in Z_n$, then returns $(\hat{\sigma}, r)$ on (η_1, η_2) to $\mathcal{B}$. $\mathcal{B}$ checks that $\hat{e}(\hat{\sigma}, \eta_1 \cdot \eta_2^r \cdot \eta^\tau) = \hat{e}(\eta, \eta)$ or not. If not, $\mathcal{B}$ terminates with Fail. Otherwise, $\mathcal{B}$ generates a ring signature on (i^*, m, R) for $\mathcal{A}$ as follows:

1. Choose $t_0, t_1 \leftarrow Z_q$, compute $\sigma = \hat{\sigma}h^{t_0}$, h^{t_1};
2. Compute $\pi_\sigma = h^{r_1^{-1}t_0\hat{u}}$;

3. Choose $t_2, t_3 \leftarrow Z_q$, compute $C_1 = (\eta_1 \hat{h}^{r_2}) \cdot h^{t_2}$, $C_2 = (\eta_2 \hat{h}^{r_3}) \cdot h^{t_3}$;
4. Compute $\pi_1 = h^{(t_0+t_1)(r_2+r_3 r)}$, $\pi_2 = h^{(t_0+t_1)(t_2+t_3 r+r_1\tau)-r_1^2}$;
5. Generate a NIWI proof π using the witness $pk_{i^*} = (\eta_1 \hat{h}^{r_2}, \eta_2 \hat{h}^{r_3})$ according to [4];
6. Publish the ring signature $\Sigma = (R, r, \sigma, h^{t_1}, \pi_\sigma, \{C_1, C_2\}, \pi_1, \pi_2, \pi)$.

Query on $\mathcal{O}_{C/D}(i)$: If $i \neq i^*$, $\mathcal{B}$ uses (x_i, y_i, e_i, d_i) to generate $\pi_{c/d}$, σ_i, π_{σ_i}; if $i = i^*$, for a generated ring signature Σ, $\mathcal{B}$ returns $\sigma_i = \hat{\sigma} h^{t_0}$, $\pi_{\sigma_i} = h^{r_1^{-1} t_0 \tilde{u}}$, $\pi_{c/d} = h^{t_0(r_2+r_3 r)+(t_0 r_1 \tau - r_1^2)\tilde{u}^{-1}}$ to $\mathcal{F}$.

Finally, $\mathcal{F}$ outputs a forgery $\Sigma^* = (R^*, r^*, \sigma^*, h^{t_1^*}, \{C_1^*, C_2^*\}, \pi_1^*, \pi_2^*, \pi^*)$ w.r.t. (m^*, R^*). If $i^* \notin R^*$, $\mathcal{B}$ claims Fail, otherwise $\mathcal{B}$ makes use of Σ^* to forge a BB signature w.r.t. (m^*, R^*). In this case, $\mathcal{F}$ cannot make corruption queries and signing queries on all $i \in R^*$. Since Σ^* is valid, so does π^*, and $\{C_1^*, C_2^*\}$ must commit a public key which belongs to R^*, therefore, the following equation holds:

$$\hat{e}(\sigma^* h^{t_1^*}, C_1^* \cdot (C_2^*)^{r^*} \cdot g^{\tau^*}) = \hat{e}(g, g)\hat{e}(\pi_1^*, \hat{h})\hat{e}(\pi_2^*, h)$$

Let λ be chosen so $\lambda \equiv 1 \bmod p$ and $\lambda \equiv 0 \bmod q$, raising both sides to power λ:

$$\begin{aligned} \hat{e}(\sigma^* h^{t_1^*}, C_1^* (C_2^*)^{r^*} \cdot g^{\tau^*})^\lambda &= \hat{e}(g, g)^\lambda \cdot \hat{e}(\pi_1^*, \hat{h})^\lambda \cdot \hat{e}(\pi_2^*, h)^\lambda \\ \hat{e}(\sigma^*, \eta_1 (\eta_2)^{r^*} \cdot \eta^{\tau^*})^\lambda &= \hat{e}(\eta h^{r_1}, \eta h^{r_1})^\lambda \\ \hat{e}((\sigma^*)^\lambda, \eta_1 \cdot \eta_2^{r^*} \cdot \eta^{\tau^*}) &= \hat{e}(\eta, \eta) \end{aligned}$$

Obviously, $(\sigma^*)^\lambda$ is a valid BB signature w.r.t. m^*. Note that, member i^* belongs to R^*, the advantage of $\mathcal{B}$'s forgery on (i^*, m^*) is $\frac{1}{|R^*|}\epsilon_{\mathcal{F}}$, where $\epsilon_{\mathcal{F}}$ is the advantage that $\mathcal{F}$ forges the valid ring signature w.r.t. (m^*, R^*). □

Remark 3. In the unforgeability game, we work with a group of composite order, which can be trivially from the Chinese Remainder Theorem. Given generators $\eta \in G_p$ and $h \in G_q$, $\eta^{r_1} h^{r_2}$ is a generator of G. Given $u = \eta^x h^y \in G$, u's projection into G_p and G_q are η^x and h^y, respectively. For $u \in G_p$ and $v \in G_q$, $\hat{e}(u, v) = 1$ holds, which implies that for all $u_1, u_2 \in G_p$, $v_1, v_2 \in G_q$, $\hat{e}(u_1 v_1, u_2 v_2) = \hat{e}(u_1, u_2) \cdot \hat{e}(v_1, v_2)$ holds.

Traceability. This property states that if a ring signature Σ w.r.t. (m, R) is valid, then there must exist one member $i \in R$ that cannot pass the disavowal protocol. This property can be reduced to the soundness of NIWI proofs in our scheme.

Theorem 3. *This scheme is traceable if the NIWI proofs are sound.*

Proof. For a valid ring signature Σ w.r.t. (m, R), there must exist a member in R who cannot deny his generation of Σ. This proceeds in two steps:

1. Σ must be consistent with pk_k for some $k \in R$. Since Σ is a valid ring signature, π_1, π_2 are used to prove $\{\sigma h^{r_1}, C_1, C_2\} \in \mathcal{L}_{BB}$ and π is a valid NIWI proof. Therefore, σ must be consistent with one public key, say pk_k, which is committed in C_1, C_2, and $pk_k \in R$.

2. If the partial signature σ is consistent with pk_k, then member k cannot disavow. Otherwise, k can output $\sigma' \neq \sigma$ and pass this equation $\hat{e}(\sigma', g^{x_k}\hat{h}^{e_k} \cdot (g^{y_k}\hat{h}^{d_k})^r \cdot g^{\tau}) = \hat{e}(g,g) \cdot \hat{e}(\pi'_{dis}, \hat{h})$. When $\hat{h}$ has order q, let $\lambda \equiv 1 \bmod p$ and $\lambda \equiv 0 \bmod q$, we raise both sides to power λ and get $\hat{e}((\sigma')^{\lambda}, g^{x_k} \cdot g^{y_k r} \cdot g^{\tau}) = \hat{e}(g,g)$. It means, $(\sigma')^{\lambda}$ is the member k's BB signature on (r,τ). σ is also the valid BB signature of member k on (r,τ), so we obtain $(\sigma')^q = \sigma = \sigma^q$ and $\sigma \neq \sigma'$, which contradicts the soundness of π_σ. □

Non-frameability. This property states that no uncorrupted member would be framed if he did not sign. If member $i \in R^*$ is uncorrupted and did not produce a ring signature Σ^* w.r.t. (m^*, R^*), he did not generate NIWI proofs (π_1, π_2, π) such that the partial signature σ^* is consistent with his public key pk_i. If member i fails to disavow the signature Σ^*, it means his disavowal $\sigma = \sigma^*$, which breaks the unforgeability of BB signature: the forger generates a BB signature on behalf of the uncorrupted member i.

5 Conclusion

In this paper, we propose a practical conditionally anonymous ring signature scheme in the standard model. The generated ring signature can be traced when it is necessary. Our traceability algorithm is constructed by confirmation protocol and disavowal protocol. The actual signer can confirm his signing through a confirmation protocol, while non-signer can refute by using the disavowal protocol if he did not generate this ring signature. Both the confirmation and disavowal protocols are non-interactive. In addition, our proposal is secure without the random oracles.

Acknowledgments. This work is supported by NSFC (No. 60973161), Government Basic Research Support for Universities (No. ZYGX2010X015), Fundamental Research Funds for the Central Universities under Grant ZYGX2011J068, Opening Project of Shanghai Key Laboratory of Integrate Administration Technologies for Information Security (No. AGK2010007) and National High Technology Joint Research Program of China (863 Program, Grant No. 2011AA010706).

References

1. 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)
2. 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)
3. Canetti, R., Goldreich, O., Halevi, S.: The Random Oracle Methodology. In: STOC 1998, pp. 209–218. ACM (1998)

4. Chandran, N., Groth, J., Sahai, A.: Ring Signatures of Sub-linear Size Without Random Oracles. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 423–434. Springer, Heidelberg (2007)
5. Chaum, D., van Heyst, E.: Group Signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
6. Groth, J., Ostrovsky, R., Sahai, A.: Perfect Non-interactive Zero Knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006)
7. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
8. Komano, Y., Ohta, K., Shimbo, A., Kawamura, S.-I.: Toward the Fair Anonymous Signatures: Deniable Ring Signatures. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 174–191. Springer, Heidelberg (2006)
9. Rivest, R.L., Shamir, A., Tauman, Y.: How to Leak a Secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)
10. Shacham, H., Waters, B.: Efficient Ring Signatures Without Random Oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 166–180. Springer, Heidelberg (2007)
11. Wu, Q., Susilo, W., Mu, Y., Zhang, F.: Ad Hoc Group Signatures. In: Yoshiura, H., Sakurai, K., Rannenberg, K., Murayama, Y., Kawamura, S.-I. (eds.) IWSEC 2006. LNCS, vol. 4266, pp. 120–135. Springer, Heidelberg (2006)
12. Zeng, S., Jiang, S., Qin, Z.: A New Conditionally Anonymous Ring Signature. In: Fu, B., Du, D.-Z. (eds.) COCOON 2011. LNCS, vol. 6842, pp. 479–491. Springer, Heidelberg (2011)

ID Based Signcryption Scheme in Standard Model

S. Sharmila Deva Selvi[1], S. Sree Vivek[1], Dhinakaran Vinayagamurthy[2,⋆], and C. Pandu Rangan[1]

[1] Theoretical Computer Science Lab, Department of Computer Science and Engineering, Indian Institute of Technology, Madras, India
{sharmila,svivek,rangan}@cse.iitm.ac.in
[2] Department of Computer Science and Engineering, College of Engineering, Guindy, Anna University, Chennai, India
dhinakaran2705@gmail.com

Abstract. Designing an ID based signcryption scheme in the standard model is among the most interesting and important problems in cryptography. However, all the existing systems in the ID based setting, in the standard model, do not have either the unforgeability property or the indistinguishability property or both of them. In this paper, we present the first provably secure ID based signcryption scheme in the standard model with both these properties. The unforgeability property of this scheme is based on the hardness of Computational Diffie-Hellman problem and the indistinguishability property of this scheme is based on the hardness of Decisional Bilinear Diffie-Hellman problem. Our scheme is strongly unforgeable in the strong attack mode called insider security. Moreover, our scheme possess an interesting property called public verifiability of the ciphertext. Our scheme integrates cleverly, a modified version of Waters' IBE and a suitably modified version of the ID based signature scheme in the standard model proposed by Paterson et al. However, our security reductions are more efficient. Specifically, while the security reductions for indistinguishability is similar to the bounds of Waters' scheme, the unforgeability reductions are way better than the bounds for Paterson et al.'s scheme.

Keywords: Provable Security, ID based signcryption, Strong Unforgeability, Standard Model, Public Ciphertext Verifiability, Insider Security.

1 Introduction

Signcryption aims at providing the confidentiality property of encryption and authentication and non-repudiation properties of signature simultaneously with a cost significantly less than the cost of performing encryption and signature separately. This notion was introduced by Zheng [31] in 1997. The reduction in the

⋆ This material is based upon work partially supported by Summer Fellowship offered by the Department of Computer Science and Engineering, Indian Institute of Technology, Madras.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 35–52, 2012.

computational and communication cost makes the scheme more practical and hence it has numerous real time applications. Fast, compact, secure, unforgeable and non-repudiated key transport, multi-cast, electronic commerce, authenticated email are some of the areas where signcryption is highly applicable.

ID based cryptography, introduced by Shamir in 1984 [22] suggests the use of user identity, such as his e-mail address or his telephone number, as his public key rather than using some arbitrary strings which requires certificates from the Certificate Authority (CA). A Private Key Generator (PKG) is a trusted entity which when given a user's identity computes the private key for the corresponding user and returns it to the user through a secure channel. This method eliminates the need for certificates, which were used in the conventional public key setting.

The first ID based signcryption scheme was proposed by Malone-Lee [18] in 2002. Many ID based signcryption schemes have been proposed since then, adopting many different strategies, thereby reducing computational cost and also reducing the ciphertext size ([5], [8], [17], [7], [2]).

But all these above schemes were proven secure in the random oracle model. Canetti et al. in 2004 [6] showed the limitations and challenges of using random oracle model. The instantiation of random oracles with real world hash functions may result in insecure schemes. So, there is a natural urge to design systems that are secure in standard model. It should be noted that the systems that are secure in standard model are in general computationally more expensive than the systems that are secure in random oracle model. We need to pay such an extra cost due to more stringent demands of standard models.

The first ID based signcryption scheme without random oracles was proposed by Yu et al. in 2009 [28] based on Waters' ID based encryption [26]. But their scheme was shown CPA insecure by Wang et al. [24], Zhang et al. [30] and Zhang [29]. Zhang [29] also showed that [28] is SUF-insecure. Meanwhile, Ren and Gu [27] proposed a Signcryption scheme based on Gentry's IBE [9] but it was shown by Wang et al. [25] that it has neither confidentiality nor existential unforgeability. An improved semantically secure scheme was proposed by Jin, Wen and Du [11] again based on Waters IBE but Li et al. [13] showed that the scheme in [11] satisfies neither IND-CCA2 property nor EUF-CMA property. Zhang [29] also proposed a new scheme. But Li et al. in 2011 [15] showed that Zhang's scheme [29] did not have IND-CPA property and they proposed a new scheme claiming it to have both IND-CCA2 and EUF-CMA properties. But the new scheme in [15] satisfies neither IND-CCA2 property nor EUF-CMA property as shown by Selvi et al. in [21]. Li et al. [14] proposed another scheme based on IBE proposed by Kiltz et al. [12] and IBS proposed by Paterson et al. [20]. But Selvi et al. [21] have also shown that there are inconsistencies in the proof of security of [14], thus concluding that all the ID based signcryption schemes proposed till now for the standard model are not provably secure. Selvi et al. [21] have also concluded that achieving a provably secure ID based signcryption scheme in the standard model through direct combination of an ID based signature scheme and an ID based encryption scheme can only be

done by the Sign then Encrypt approach. However, for any Sign then Encrypt scheme, $cost\ of\ signcryption = cost\ of\ signature + cost\ of\ encryption$. But our objective of designing a signcryption scheme is to have a scheme that has $cost\ of\ signcryption < cost\ of\ signature + cost\ of\ encryption$ [31]. Hence we need to take a fresh look at the design of the signcryption protocol and arrive at an efficient customized scheme of signcryption. In the subsequent section we present one such novel scheme and formally prove its security.

Hea An, Dodis and Rabin in 2002 [1] introduced the notion of *strong unforgeability*, to avoid the problems due to malleability. If a scheme is malleable, then an adversary can produce a valid signature on a message when another valid signature on the same message is available. So, they proved the unforgeability property of their signcryption scheme using this strong notion. A signature scheme becomes non-malleable when it satisfies this property. There are several transformations available in literature to convert an EUF-CMA secure scheme to a SUF-CMA secure scheme for signature schemes. Some of the transformations available for the standard model are the transformations proposed by Boneh et al. [4], Bellare et al. [3], Teranishi et al. [23] and Huang et al. [10].

The *public ciphertext verifiability* property of a scheme is very useful in low power devices. This property allows any third party application, like firewalls, to verify the validity of the sender and ciphertext without any interaction with the receiver i.e without knowing the receiver's secret key. This will allow the application to prevent the ciphertexts, modified by an adversary, from reaching the devices. Only valid ciphertexts can reach them, thus preventing unnecessary use of their resources for decrypting the invalid ciphertexts. Here, the important property is that, the third party application while verifying should not obtain any knowledge about the message that is signcrypted. This property is provided by the signcryption scheme proposed by Chow et al. [8]. But that scheme was proven secure only in the random oracle model.

1.1 Our Contribution

In this paper we present the first provably secure ID based signcryption scheme without random oracles. Our scheme is based on the ID based signature scheme in the standard model proposed by Paterson et al. [20], which in turn is based on the PKI based signature scheme proposed by Waters [26]. We base the IND-CCIA2 property of our scheme on the hardness of the Decisional Bilinear Diffie Hellman assumption and the SUF-CMIA property of our scheme on the hardness of the Computational Diffie Hellman assumption. The property of *strong unforgeability* is present in our scheme even without using any of the transformations available to convert an existentially unforgeable scheme to a strongly unforgeable scheme in the standard model. The proposed scheme also offers insider security with respect to both confidentiality and unforgeability which ensures that the signcryption scheme is secure even when one among the sender or the receiver colludes with the adversary against the other. The scheme proposed exhibits the crucial property of public ciphertext verifiability. Recall that all the ID based signcryption schemes in the standard model such as [28], [27],

[11], [29] and [15] are completely broken and the most recent scheme proposed by Li et al. [14] has flaws in the proof. Even if the flaws in the proof of [14] are fixed, our scheme has the following advantages over [14].

- The security of our scheme is based on a harder assumption i.e DBDH, compared to the modified DBDH (mDBDH) used by [14].
- Our scheme has a tighter security reduction.
- Our scheme is more efficient than the one in [14].

1.2 Organisation

The rest of this paper is organized as follows. In section 2, preliminaries like bilinear pairing, computational assumptions, a generic ID based signcryption scheme, formal security model for ID based signcryption scheme are explained. We present our ID based signcryption scheme in section 3. We prove the confidentiality property and the strong unforgeability property of our scheme in section 4. The efficiency of our scheme is explained in section 5 and the paper is concluded in section 6.

2 Preliminaries

2.1 Bilinear Pairing

Let $\mathbb{G}$ and $\mathbb{G}_T$ be multiplicative groups of prime order p and let g be generator of $\mathbb{G}$. The bilinear map $\hat{e}$ is admissible only if it satisfies the following conditions:

- ***Bilinearity.*** For all $g_1, g_2, g_3 \in \mathbb{G}$,
  - $\hat{e}(g_1 g_2, g_3) = \hat{e}(g_1, g_3)\hat{e}(g_2, g_3)$
  - $\hat{e}(g_1, g_2 g_3) = \hat{e}(g_1, g_2)\hat{e}(g_1, g_3)$
  - $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ for all $a, b \in \mathbb{Z}_p$.
- ***Non-Degeneracy.*** For all $g_1, g_2 \in \mathbb{G}$, $\hat{e}(g_1, g_2) \neq I_{\mathbb{G}_T}$, where $I_{\mathbb{G}_T}$ is the identity element of $\mathbb{G}_T$.
- ***Computability.*** There exists an efficient algorithm to compute $\hat{e}(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}$.

2.2 Computational Assumptions

In this section, we review the computational assumptions relevant to the protocol we propose.

Computational Diffie-Hellman Problem (CDH). Given $(g, g^a, g^b) \in \mathbb{G}^3$ for unknown $a, b \in \mathbb{Z}_p$, the CDH problem in $\mathbb{G}$ is to compute g^{ab}.

Definition. The advantage of any probabilistic polynomial time algorithm $\mathcal{A}$ in solving the CDH problem in $\mathbb{G}$ is defined as:

$$Adv_{\mathcal{A}}^{CDH} = Pr\left[\mathcal{A}(g, g^a, g^b) = g^{ab} \mid a, b \in \mathbb{Z}_p\right]$$

The *CDH Assumption* is that, for any probabilistic polynomial time algorithm $\mathcal{A}$, the advantage $Adv_{\mathcal{A}}^{CDH}$ is negligibly small.

Decisional Bilinear Diffie-Hellman Problem (DBDH) Given $(g, g^a, g^b, g^c, \alpha) \in \mathbb{G}^4 \times \mathbb{G}_T$ for unknown $a, b, c \in \mathbb{Z}_p$, the DBDH problem in $\mathbb{G}$ is to decide if $\alpha = \hat{e}(g, g)^{abc}$.

Definition. The advantage of any probabilistic polynomial time algorithm $\mathcal{A}$ in solving the DBDH problem in $\mathbb{G}$ is defined as:

$$Adv_{\mathcal{A}}^{DBDH} = \left| Pr\left[\mathcal{A}(g, g^a, g^b, g^c, \hat{e}(g, g)^{abc}) = 1\right] - Pr\left[\mathcal{A}(g, g^a, g^b, g^c, \alpha) = 1\right] \right|$$

The *DBDH Assumption* is that, for any probabilistic polynomial time algorithm $\mathcal{A}$, the advantage $Adv_{\mathcal{A}}^{DBDH}$ is negligibly small, for $a, b, c \in_R \mathbb{Z}_p$.

2.3 ID Based Signcryption

A generic ID based signcryption scheme consists of the following four algorithms.

- ***Setup:*** This algorithm is run by the Private Key Generator (PKG). When given a security parameter k, this algorithm outputs public parameters *params* and a master secret key MSK. PKG keeps the corresponding MSK as its secret value.
- ***Extract:*** When given an Identity ID, the PKG runs this algorithm using the *params* and MSK and generates the private key d_u for the user. The PKG then transmits the generated private key to the corresponding user through a secure channel.
- ***Signcrypt:*** This algorithm is run by the sender. It takes as input, the public parameters *params*, the private key d_A of the sender, the identity of the receiver ID_B and the message m to be sent to the receiver. The signcryption σ is produced as output which is sent to the receiver.
- ***Unsigncrypt:*** On receiving the signcryption σ from the sender, the receiver runs this algorithm. The public parameters *params*, the identity of the sender ID_A, the private key of the receiver d_B and the signcryption σ are given as input to this algorithm. The message m is obtained as output if the signcryption is valid or $\perp$ is given as output. For the consistency of the signcryption algorithm, if $\sigma = Signcrypt(params, d_A, ID_B, m)$, then $m = Unsigncrypt(params, ID_A, d_B, \sigma)$.

2.4 Security Model for ID Based Signcryption

Indistinguishability. In 2002, Malone-Lee [18] proposed the first ID based signcryption scheme. He extended the semantic security of encryption schemes to signcryption schemes as *Indistinguishability of ID based signcryption under Adaptive Chosen Ciphertext Attack* (IND-IBSC-CCA2). Later, Chow et al. [8] used a stronger notion of security by allowing the adversary to adaptively choose the identities to create a forgery during the challenge phase. This is similar to the one proposed in [16]. This model was termed as *Indistinguishability of ID based signcryption under Adaptive Chosen Ciphertext and Identity Attack* (IND-IBSC-CCIA2). This is the strongest notion available in the literature for proving the

Indistinguishability property of the signcryption schemes. The formal definition is given below.

A signcryption scheme is semantically secure against chosen ciphertext and identity attack (IND-IBSC-CCIA2) if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.

1. The challenger $\mathcal{C}$ runs the **Setup** algorithm and sends the public parameters to the adversary $\mathcal{A}$
2. **Training Phase 1:** The adversary $\mathcal{A}$ can ask a polynomially bound number of queries to the following oracles.
   - **Extract Oracle:** When $\mathcal{A}$ queries for the private key of an identity ID, the challenger $\mathcal{C}$ runs the **Extract** algorithm giving the ID and *params* as input. $\mathcal{C}$ forwards the private key d_u of ID output by the Extract algorithm to $\mathcal{A}$.
   - **Signcrypt Oracle:** $\mathcal{A}$ can ask for the signcryption on any message m from any sender identity ID_A to any receiver identity ID_B. When $\mathcal{A}$ does so, $\mathcal{C}$ runs the **Extract** algorithm for the sender identity ID_A and gets the private key d_A of ID_A. $\mathcal{C}$ then inputs $\langle m, d_A, ID_B \rangle$ into the **Signcrypt** algorithm and forwards its output σ to $\mathcal{A}$.
   - **Unsigncrypt Oracle:** $\mathcal{A}$ queries for the unsigncryption of the ciphertext σ by producing the sender identity ID_A and receiver identity ID_B. The challenger $\mathcal{C}$ runs the **Extract** algorithm to find the private key d_B of the receiver ID_B. $\mathcal{C}$ then runs the **Unsigncrypt** algorithm giving $\langle \sigma, ID_A, d_B \rangle$ as input and forwards the output m or $\perp$ to $\mathcal{A}$.

   During this phase $\mathcal{A}$ can produce its queries adaptively i.e every query can be asked dependent on the output of the previous queries.
3. **Challenge Phase** At the end of Phase 1, $\mathcal{A}$ chooses two plaintext messages m_0^*, $m_1^* \in \{0,1\}^{l_m}$, two identities i.e. sender identity ID_A^* and receiver identity ID_B^* on which it wishes to be challenged and sends them to the challenger $\mathcal{C}$. In this case, $\mathcal{A}$ should not have queried the Extract oracle for ID_B^*. $\mathcal{C}$ takes a bit b randomly from $\{0,1\}$ and runs **Signcrypt**(m_b^*, d_A^*, ID_B^*), where d_A^* is the output of **Extract**(ID_A^*). $\mathcal{C}$ sends the output σ^* to $\mathcal{A}$ as the challenge ciphertext.
4. **Training Phase 2:** The adversary $\mathcal{A}$, after receiving σ^* can ask again for polynomially bound number of queries on the above mentioned oracles adaptively in the same way as in Phase 1 except that $\mathcal{A}$ cannot ask for the **Extract**(ID_B^*) query and **Unsigncrypt** query involving $\langle \sigma^*, ID_A^*, ID_B^* \rangle$.
5. Once this Phase 2 of Training is over, $\mathcal{A}$ outputs b'. $\mathcal{A}$ wins this game if $b' = b$.

The advantage of adversary $\mathcal{A}$ in the above game is defined by $Adv(\mathcal{A}) = (2 \times Pr(b' = b) - 1)$.

The importance of this security model is that the adversary $\mathcal{A}$ can ask for the private key d_A^* of the sender whose identity is ID_A^* during Phase 2. This captures the *insider* security model, which means that $\mathcal{A}$ will not have any added advantage in the above game even when the private key of the sender is leaked.

Also, $\mathcal{A}$ is allowed to query the Signcrypt oracle with the challenge messages m_0^* or m_1^* with the sender identity as ID_A^* and receiver identity as ID_B^*.

Unforgeability. Malone-Lee [18] proposed the *Existential Unforgeability of ID based signcryption under Chosen Message Attack* (EUF-IBSC-CMA). Later, Chow et al. [8] proposed a stronger notion of security called *Existential Unforgeability of ID based signcryption under Chosen Message and Identity Attack* (EUF-IBSC-CMIA), where the adversary can not only choose the message to attack adaptively but also the identities on which it is going to attack. This notion is defined by the game between challenger and adversary as given below.

An ID based signcryption scheme is said to have the property of *Existential Unforgeability under Chosen Message and Identity Attack* if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.

1. The challenger $\mathcal{C}$ runs the **Setup** algorithm and generates the public parameters and the Master Secret Key MSK. $\mathcal{C}$ then gives the public parameters *params* to the adversary $\mathcal{A}$
2. Now the adversary $\mathcal{A}$ can ask a polynomially bound number of queries to any of the following oracles.
   - **Extract Oracle:** When $\mathcal{A}$ queries for the private key of an identity ID, the challenger $\mathcal{C}$ runs the **Extract** algorithm giving the ID, *params* and MSK as input. $\mathcal{C}$ forwards the output d_u given by the algorithm to the adversary $\mathcal{A}$.
   - **Signcrypt Oracle:** $\mathcal{A}$ can ask for the signcryption on any message m by the sender identity ID_A for the receiver identity ID_B. In this case, $\mathcal{C}$ runs the **Extract** algorithm for the sender identity ID_A and gets the private key d_A of ID_A as output. $\mathcal{C}$ then inputs $\langle m, d_A, ID_B \rangle$ into the **Signcrypt** algorithm and forwards its output σ to $\mathcal{A}$.
   - **Unsigncrypt Oracle:** When $\mathcal{A}$ queries for the unsigncryption of the ciphertext σ by producing the sender identity ID_A and receiver identity ID_B. The challenger $\mathcal{C}$ first runs the **Extract** algorithm for finding the private key d_B of the receiver ID_B. $\mathcal{C}$ then runs the **Unsigncrypt** algorithm inputting $\langle \sigma, ID_A, d_B \rangle$ and forwards its output m to $\mathcal{A}$.

   During this phase $\mathcal{A}$ can produce its queries adaptively i.e every query is dependant on the previous queries.
3. At the end this training phase, $\mathcal{A}$ outputs the forgery $\langle \sigma^*, ID_A^*, ID_B^* \rangle$ for some message m^*. This forgery is valid when ID_A^* is not queried to the **Extract** oracle and if $\langle m^*, ID_A^*, ID_B^* \rangle$ is not already queried to the **Signcrypt** oracle.
4. $\mathcal{A}$ wins the game if σ^* is a valid forgery on the message m^* as signcrypted by the identity ID_A^* intended for the identity ID_B^*.

The advantage of adversary $\mathcal{A}$ in the above game is defined by

$$Adv(\mathcal{A}) = Pr\left[Unsigncrypt(\sigma^*, ID_A^*, ID_B^*) \; = \; m^*\right]$$

In this security model, the importance is that the adversary can query the **Extract** oracle for the identity of the receiver ID_B^* in the above game which captures the insider security model for unforgeability. So, even when the private key of the intended receiver is leaked, the adversary $\mathcal{A}$ will not have any added advantage in producing a valid forgery in the above game. But the restriction here is that $\langle m^*, ID_A^*, ID_B^* \rangle$ should not have been queried already to the **Signcrypt** oracle. The work done by Li et al. [14] has used similar security models which provide insider security.

Strong Unforgeability. Hea An et al. [1] proposed that there is no necessity for an adversary to produce forgery on a message that is not already queried. Forgery can also be produced on the message that is queried already to the Signcrypt oracle with the condition that the forged signcryption on m is not the same as the one that is output by the **Signcrypt** oracle for the same message m, with the same sender and the same receiver as the forgery. This notion is called *Strong Unforgeability*. Our new scheme satisfies the notion of *Strong Unforgeability of ID based signcryption under Chosen Message and Identity Attack* (SUF-IBSC-CMIA). This is the strongest security notion available for proving the unforgeability property of signcryption schemes. We state this notion formally as follows.

An ID based signcryption scheme is said to have the property of Strong Unforgeability under Chosen Message and Identity Attack if there is no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the game described as follows.

1. The challenger follows the same procedure as EUF-IBSC-CMIA game during the setup and the training phases.
2. After training is over, the adversary $\mathcal{A}$, produces $\langle \sigma^*, ID_A^*, ID_B^* \rangle$ for the message m^*, where ID_A^* is not queried to the **Extract** oracle and σ^* is not the output of the **Signcrypt** query asked by $\mathcal{A}$ with $\langle m^*, ID_A^*, ID_B^* \rangle$ as input.
3. $\mathcal{A}$ wins the game if σ^* is a valid forgery on the message m^* as signcrypted from the sender identity ID_A^* to the receiver identity ID_B^*.

The advantage of adversary $\mathcal{A}$ in the above game is defined by

$$Adv(\mathcal{A}) = Pr\left[Unsigncrypt(\sigma^*, ID_A^*, ID_B^*) \; = \; m^*\right]$$

In the above security model, $\mathcal{A}$ can produce any valid $\langle \sigma^*, ID_A^*, ID_B^* \rangle$ tuple for the message m^*, where $\langle m^*, \sigma^* \rangle$ is not the output of any **Signcrypt** query with ID_A^* and ID_B^* as the sender and receiver identities during the training phase. So, m^* may have been queried already to the Signcrypt oracle provided that σ^* is not the output of the oracle during that query with the sender and receiver identities being the same during that query and the forgery.

3 Our Scheme

Setup

Consider groups $\mathbb{G}, \mathbb{G}_T$ of prime order p whose size is determined by the security parameter k. Let g be the generator of the group $\mathbb{G}$. There exists a bilinear map defined by $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, which is efficiently computable. Now, choose $\alpha \in \mathbb{Z}_p$ randomly and compute $g_1 = g^\alpha$. Randomly pick g_2, h_2 from $\mathbb{G}$ and compute g_2^α, h_2^α. Also, choose h_1, h_3 randomly from $\mathbb{G}$. Choose u', v', m' randomly from the group $\mathbb{G}$ and also choose vectors $\mathbb{U} = (u_i)$ and $\mathbb{V} = (v_i)$ each of length n_u and $\mathbb{M} = (m_i)$ of length l, whose elements are randomly chosen from group $\mathbb{G}$. Here, n_u is the length of the identity strings that are used. Let n_m be the length of the message sent. There are four one-way, collision resistant cryptographic hash functions $H_1 : \mathbb{G}_T \times \{0,1\}^{l_\tau} \to \{0,1\}^{n_m}$, $H_2 : \{0,1\}^{|p|+n_u+l_\tau} \to \{0,1\}^l$, $H_3 : \mathbb{G} \to \mathbb{Z}_p^*$ and $H_4 : \{0,1\}^{n_m+|p|+n_u} \to \mathbb{Z}_p^*$, where l is large enough that the hash functions are collision resistant and $l_\tau \approx 40$. Note that a typical value of l could be 256 and a random bit string of length l cannot be guessed in polynomial time. The system parameter *params* is given by $\langle \mathbb{G}, \mathbb{G}_T, \hat{e}, H_1, H_2, H_3, H_4, g, g_1, g_2, h_1, h_2, h_3, u', v', m', \mathbb{U}, \mathbb{V}, \mathbb{M} \rangle$. The master secret key of the system is $\langle \alpha, g_2^\alpha, h_2^\alpha \rangle$. The following algorithms define our scheme.

Extract(u, *params*, MSK)

Let an identity of a user u be represented by ID_u which is a bit string of length n_u and let $ID_u[i]$ be the i^{th} bit of ID_u. Define $\Omega_u \subseteq \{1, 2, ..., n_u\}$ to be the set of indices i such that $ID_u[i] = 1$. The private key of a user u is constructed by choosing a random $r_u \in \mathbb{Z}_p^*$ and then computing

$$d_u = (d_S, d_{US}, d_R) = (g_2^\alpha (u' \prod_{i \in \Omega_u} u_i)^{r_u}, h_2^\alpha (v' \prod_{i \in \Omega_u} v_i)^{r_u}, g^{r_u})$$

Signcrypt(*params*, d_A, B, m)

The private key of the sender A with identity ID_A as given by the PKG is

$$d_A = (d_{S_A}, d_{US_A}, d_{R_A}) = (g_2^\alpha (u' \prod_{i \in \Omega_A} u_i)^{r_A}, h_2^\alpha (v' \prod_{i \in \Omega_A} v_i)^{r_A}, g^{r_A})$$

where $\Omega_A \subseteq \{1, 2, ..., n_u\}$ is the set of indices i such that $ID_A[i] = 1$. Now, when given a message $m \in \{0,1\}^{n_m}$ signcryption on the message is done by the sender A as follows.

- Choose $r \in \mathbb{Z}_p$ randomly and compute $\sigma_1 = g^r \in \mathbb{G}$
- Encrypt the message as $\sigma_2 = H_1(\hat{e}\,(g_1, h_2)^r, \tau) \oplus m \in \{0,1\}^{n_m}$, where $\tau \in_R \{0,1\}^{l_\tau}$
- Compute $\sigma_3 = (v' \prod_{i \in \Omega_B} v_i)^r \in \mathbb{G}$, where Ω_B is the set of vertices i such that $ID_B[i] = 1$. Here, B is the receiver of the message.

- Set $\sigma_4 = d_{R_A} \in \mathbb{G}$
- Compute $\lambda = H_3(\sigma_1)$, $\beta = H_2(\sigma_4, ID_A, \tau)$, $\rho = H_4(\sigma_2, \sigma_3, ID_B)$
- Compute $\sigma_5 = d_{S_A}(m' \prod_{j \in \overline{\beta}} m_j)^r \left(h_1^{\lambda} h_3\right)^{r\rho} \in \mathbb{G}$, where $\overline{\beta} \subseteq \{1, 2, ..., l\}$ denotes the set of indices j such that $\beta[j] = 1$

The ciphertext $\sigma = \langle \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \tau \rangle$ is sent to the receiver. The size of the ciphertext formed is $4|p| + n_m + l_\tau$. Note that this scheme achieves the property of strong unforgeability without using any of the transformations available to convert an existentially unforgeable scheme to a strongly unforgeable one.

Unsigncrypt(*params*, A, d_B, σ)

When the receiver B receives the ciphertext $\sigma = \langle \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \tau \rangle$, he proceeds as follows.

- The private key d_B received from the PKG is

$$d_B = (d_{S_B}, d_{US_B}, d_{R_B}) = (g_2^{\alpha}(u' \prod_{i \in \Omega_B} u_i)^{r_B}, h_2^{\alpha}(v' \prod_{i \in \Omega_B} v_i)^{r_B}, g^{r_B})$$

- Compute $\lambda = H_3(\sigma_1)$, $\beta = H_2(\sigma_4, ID_A, \tau)$, $\rho = H_4(\sigma_2, \sigma_3, ID_B)$
- Then, using β, ρ and λ, check the validity of σ as follows

$$\hat{e}(\sigma_5, g) \stackrel{?}{=} \hat{e}(g_1, g_2)\, \hat{e}(u' \prod_{i \in \Omega_A} u_i, \sigma_4)\, \hat{e}((m' \prod_{j \in \overline{\beta}} m_j)(h_1^{\lambda} h_3)^{\rho}, \sigma_1) \qquad (1)$$

  where Ω_A is the set of indices i such that $ID_A[i] = 1$ and $\overline{\beta} \subseteq \{1, 2, ..., l\}$ denotes the set of indices j such that $\beta[j] = 1$
- If σ is invalid, reject σ and halt.
- If σ is valid, compute $\hat{e}\,(g_1, h_2)^r = \dfrac{\hat{e}\,(d_{US_B}, \sigma_1)}{\hat{e}\,(d_{R_B}, \sigma_3)}$
- Obtain the message as $m = \sigma_2 \oplus H_1(\hat{e}\,(g_1, h_2)^r, \tau)$

The above verification process stated in equation (1) can be done by any user who has access to σ, because all the components used in the verification process are either the values in *params* $\langle g, g_1, g_2, u', \mathbb{U}, m', \mathbb{M}, h_1, h_3 \rangle$, components of the ciphertext $\langle \sigma_1, \sigma_4, \sigma_5 \rangle$ or components that are derived from the ciphertext $\langle \lambda, \beta, \rho \rangle$. and thus the integrity and validity of the sender and the ciphertext can be verified by anyone. This gives the property of **Public Ciphertext Verifiability** to our scheme.

4 Security

4.1 Indistinguishability

We first prove the Indistinguishability property, *Indistinguishability of ID based signcryption under Adaptive Chosen Ciphertext and Identity Attack* (IND-IBSC-CCIA2) of our scheme with the following theorem.

Theorem 1. *If there exists an IND-IBSC-CCIA2 adversary for our scheme which can distinguish ciphertexts during the IND-IBSC-CCIA2 game explained above, with a non-negligible probability ϵ when it runs for a polynomial time t, asking at most q_E extract queries, q_S signcrypt queries and q_{US} unsigncrypt queries, then there exists another algorithm, which can solve the Decisional Bilinear Diffie-Hellman (DBDH) problem with probability ϵ' in polynomial time t', where*

$$\epsilon' \geq \frac{\epsilon}{4(q_E)(n_u+1)}$$

$$t' \leq t + \mathcal{O}((n_u q_E + (n_u + l)(q_S + q_{US}))t_m + (q_E + q_S + q_{US})t_e + (q_S + q_{US})t_p)$$

where n_u is the length of the identity string, t_m, t_e, t_p are the time required for each multiplication, each exponentiation and each bilinear pairing respectively and l is a value large enough such that the hash functions outputting $\{0,1\}^l$ in the scheme are collision resistant.

Proof

Let us assume that a $(\epsilon, t, q_E, q_S, q_{US})$-adversary $\mathcal{A}$ for our scheme exists. We will construct another algorithm $\mathcal{B}$ from this adversary $\mathcal{A}$, who can solve the Decisional Bilinear Diffie- Hellman (DBDH) problem with a non-negligible probability ϵ' in polynomial time t'.

The algorithm $\mathcal{B}$ receives a DBDH tuple $\langle g, g^a, g^b, g^c, T\rangle \in \mathbb{G}^4 \times \mathbb{G}_T$, where g is a generator of a prime order group $\mathbb{G}$ of order p. $\mathcal{B}$ simulates a challenger for the adversary $\mathcal{A}$ to decide whether T is $\hat{e}(g,g)^{abc}$ or not. This simulation is described as follows:

Setup

The simulator $\mathcal{B}$ sets $l_u = 2(q_E)$, where q_E is the number of Extract queries. Here, the values q_S and q_{US} are not bounded because the Signcrypt and the Unsigncrypt queries do not abort when an Extract query of the sender or receiver identity, used in any Signcrypt or Unsigncrypt query, aborts. This will be evident from the explanation for the Signcrypt and the Unsigncrypt oracles. $\mathcal{B}$ then chooses an integer k_u randomly such that $0 \leq k_u \leq n_u$. For the given values of q_E, q_S, q_{US} and n_u, we assume that $l_u(n_u + 1) < p$. Then, $\mathcal{B}$ chooses $x' \in \mathbb{Z}_{l_u}$ randomly and also chooses a vector $\mathbb{X} = (x_i)$ of length n_u where the elements of $\mathbb{X}$ are chosen randomly from $\mathbb{Z}_{l_u}$. $\mathcal{B}$ chooses an integer y' randomly from $\mathbb{Z}_p$ and a vector $\mathbb{Y} = (y_i)$ of length n_u, where the elements of $\mathbb{Y}$ are also chosen randomly from $\mathbb{Z}_p$.

Here we define a pair of functions for a user with identity ID_u as follows:

$$F(u) = x' + \textstyle\sum_{i \in \Omega_u} x_i - l_u k_u \qquad\qquad J(u) = y' + \textstyle\sum_{i \in \Omega_u} y_i$$

The simulator now sets the public parameters as follows:

$$g_1 = g^a \qquad g_2 = g^d \qquad h_1 = g_1^{(\lambda^*)^{-1}} \qquad h_3 = g_1^{-1} g^{\theta} \qquad h_2 = g^b$$

where d and θ are chosen randomly from $\mathbb{Z}_p$ and $\lambda^* = H_3(g^c)$. The values g^a, g^b, g^c are from the DBDH tuple given to the challenger $\mathcal{C}$.

$$v' = h_2^{x'-l_u k_u} g^{y'} \qquad v_i = h_2^{x_i} g^{y_i} \qquad v' \prod_{i\in\Omega} v_i = h_2^{F(u)} g^{J(u)}$$

Finally, $\mathcal{B}$ chooses two integers e' and f' randomly from $\mathbb{Z}_p$ and two vectors $\mathbb{E} = (e_i)$ and $\mathbb{F} = (f_i)$ of lengths n_u and n_m respectively, where the elements of $\mathbb{E}$ and $\mathbb{F}$ are chosen randomly from $\mathbb{Z}_p$. For any identity ID_u,

$$u' = g^{e'} \qquad u_i = g^{e_i} \qquad u' \prod_{i\in\Omega_u} u_i = g^{e'} g^{\sum_{i\in\Omega_u} e_i} = g^{e'+\sum_{i\in\Omega_u} e_i}$$

where $\Omega_u \subseteq \{1, 2, .., n_u\}$ is the set of indices i where $ID_u[i] = 1$.

For any β got for a message m as explained in the scheme,

$$m' = g^{f'} \qquad m_i = g^{f_i} \qquad m' \prod_{i\in\overline{\beta}} m_i = g^{f'} g^{\sum_{i\in\overline{\beta}} f_i} = g^{f'+\sum_{i\in\overline{\beta}} f_i}$$

There are four one-way, collision resistant, cryptographic hash functions $H_1 : \mathbb{G}_T \times \{0,1\}^{l_\tau} \to \{0,1\}^{n_m}$, $H_2 : \{0,1\}^{|p|+n_u+l_\tau} \to \{0,1\}^l$, $H_3 : \mathbb{G} \to \mathbb{Z}_p^*$ and $H_4 : \{0,1\}^{n_m+|p|+n_u} \to \mathbb{Z}_p^*$, where l is large enough that the hash functions are collision resistant and $l_\tau \approx 40$. Note that a typical value of l could be 256 and a random bit string of length l cannot be guessed in polynomial time.

Training Phase 1

The simulator during this phase answers to the queries from the adversary $\mathcal{A}$ as follows.

Extract Queries

The simulator $\mathcal{B}$ does not know the master secret key h_2^a. So, when the adversary $\mathcal{A}$ asks for the private key of an identity ID_u, $\mathcal{B}$ responds as follows. $\mathcal{B}$ calculates $F(u)$ for the identity ID_u. If $F(u) = 0 \bmod p$, it aborts. Otherwise, $\mathcal{B}$ randomly chooses $r_u \in \mathbb{Z}_p$ and calculates the private key $d_u = (d_S, d_{US}, d_R)$ similar to [26] as

$$d_S = g_1^d \left(u' \prod_{i\in\Omega_u} u_i \right)^{r_u} g_1^{-(e'+\sum_{i\in\Omega_u} e_i)/F(u)}$$

$$d_{US} = g_1^{-J(u)/F(u)} \left(v' \prod_{i\in\Omega_u} v_i \right)^{r_u}, d_R = g_1^{-1/F(u)} g^{r_u}$$

where $\Omega_u \subseteq \{1, 2, ..., n_u\}$ is the set of indices i such that $ID_u[i] = 1$.

Signcrypt Queries

When $\mathcal{A}$ queries the Signcrypt oracle for signcryption of a message m by the user with identity ID_A as sender and the user with identity ID_B as the intended receiver, $\mathcal{B}$ simulates a valid ciphertext as follows.

$\sigma_1 = g^r$, where $r \in \mathbb{Z}_p$ is randomly chosen by $\mathcal{B}$
$\sigma_2 = H_1(\hat{e}(g_1, h_2)^r, \tau) \oplus m$, where $\tau \in_R \{0,1\}^{l_\tau}$
$\sigma_3 = \left(v' \prod_{i \in \Omega_B} v_i\right)^r$
$\sigma_4 = g^{r_A}$, where r_A is the randomness stored for ID_A in the list l_r. Otherwise choose $r_A \in \mathbb{Z}_p$ randomly and store it in l_r. Note that l_r is the list that stores $\langle ID_u, r_u \rangle$ tuples.
$\beta = H_2(\sigma_4, ID_A, \tau)$, $\rho = H_4(\sigma_2, \sigma_3, ID_B)$ and $\lambda = H_3(\sigma_1)$
$\sigma_5 = g_1^d \left(u' \prod_{i \in \Omega_A} u_i\right)^{r_A} \left(m' \prod_{j \in \overline{\beta}} m_j\right)^r \left(h_1^\lambda h_3\right)^{r\rho}$
where $\overline{\beta} \subseteq \{1, 2, ..., l\}$ denotes the set of indices j such that $\beta[j] = 1$.

The equation above for σ_5 is correct because $g_1^d = (g^a)^d = (g^d)^a = g_2^a$.
The ciphertext $\sigma = \langle \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \tau \rangle$ is sent to the adversary $\mathcal{A}$. Here, the Signcrypt queries never abort and they do not need an Extract query for the sender identity ID_A within them.

Unsigncrypt Queries
When $\mathcal{A}$ queries $\langle \sigma, ID_A, ID_B \rangle$ i.e the unsigncryption of the ciphertext σ which was signcrypted by the sender ID_A for the intended receiver ID_B, to the Unsigncrypt oracle simulated by $\mathcal{B}$, it proceeds as follows.

$\mathcal{B}$ does an Extract query for the receiver identity ID_B. If the Extract query does not abort, $\mathcal{B}$ receives the private key for ID_B as output from the Extract oracle.

$$d_B = (d_{S_B}, d_{US_B}, d_{R_B}) = (g_2^a (u' \prod_{i \in \Omega_B} u_i)^{r_B}, h_2^a (v' \prod_{i \in \Omega_B} v_i)^{r_B}, g^{r_B})$$

The simulator uses the private keys d_{US_B}, d_{R_B} got from the extract oracle to unsigncrypt the ciphertext σ using the Unsigncrypt algorithm as given in the Scheme. If the extract query aborts i.e if $F(B) = 0 \ mod \ p$, the simulator proceeds as follows. The simulator calculates Δ as given below.

$$\Delta = \frac{\sigma_5}{g_1^d \sigma_4^{(e' + \sum_{i \in \Omega_A} e_i)} \sigma_1^{(f' + \sum_{i \in \overline{\beta}} f_i)} \sigma_1^{\theta\rho}} = \frac{\sigma_5}{d_{S_A} (u' \prod_{i \in \Omega_A} u_i)^{r_A} (m' \prod_{j \in \overline{\beta}} m_j)^r \sigma_1^{\theta\rho}}$$
$$= \frac{(h_1^\lambda h_3)^{r\rho}}{\sigma_1^{\theta\rho}} = (g_1^{(\lambda/\lambda^*)-1})^{r\rho}$$

Then, we calculate $\Delta^* = \Delta^{(((\lambda/\lambda^*)-1)\rho)^{-1}} = g_1^r$, where $\lambda = H_3(\sigma_1)$. Now, we can obtain the message as follows. $m = \sigma_2 \oplus H_1(\hat{e}(\Delta^*, h_2), \tau) = \sigma_2 \oplus H_1(\hat{e}(g_1^r, h_2), \tau) = \sigma_2 \oplus H_1(\hat{e}(g_1, h_2)^r, \tau)$

This message can be returned if the verification in Eq.(1) is satisfied. Thus, the Unsigncrypt queries never abort even if the Extract queries for the corresponding receiver identities abort.

Challenge Phase

The adversary $\mathcal{A}$ can adaptively ask polynomially bound number of these Extract, Signcrypt and Unsigncrypt queries to $\mathcal{B}$. When $\mathcal{A}$ decides that training

is enough, it produces two messages m_0^* and m_1^* along with the sender identity ID_A^* and receiver identity ID_B^* adaptively and sends them to the challenger. The challenger randomly chooses $\gamma \in \{0,1\}$ and then simulates the challenge ciphertext as follows.
$\sigma_1^* = g^c$
$\sigma_2^* = H_1(T, \tau^*) \oplus m_\gamma^*$, where g^c and T are taken by $\mathcal{B}$ from the DBDH tuple given and $\tau^* \in_R \{0,1\}^{l_\tau}$.
$\sigma_3^* = \left(v' \prod_{i \in \Omega_B} v_i\right)^c = (g^c)^{J(B^*)}$, where $F(B^*) = 0 \ mod \ p$
$\sigma_4^* = g^{r_A}$, where $r_A \in \mathbb{Z}_p$ is randomly chosen
$\beta^* = H_2(\sigma_4^*, ID_A^*, \tau^*)$, $\rho^* = H_4(\sigma_2^*, \sigma_3^*, ID_B^*)$ and $\lambda^* = H_3(\sigma_1^*)$
$\sigma_5^* = g_1^d \left(u' \prod_{i \in \Omega_A} u_i\right)^{r_A} (g^c)^{f' + \sum_{j \in \overline{\beta^*}} f_i} (g^c)^{\theta \rho^*}$
where $\overline{\beta^*} \subseteq \{1, 2, ..., l\}$ denotes the set of indices j such that $\beta^*[j] = 1$.

Note that, the simulator will be able to successfully simulate the challenge ciphertext without aborting, as explained above, only if $F(B^*) = 0 \ mod \ p$. The simulator aborts if $F(B^*) \neq 0 \ mod \ p$ as it will not be able to simulate the component σ_3 when $F(B^*) \neq 0 \ mod \ p$.

The ciphertext $\sigma^* = \langle \sigma_1^*, \sigma_2^*, \sigma_3^*, \sigma_4^*, \sigma_5^*, \tau^* \rangle$ is sent to the adversary $\mathcal{A}$.

Here, if the simulator $\mathcal{B}$ was given a valid DBDH tuple i.e. if $T = \hat{e}(g,g)^{abc}$, then the challenge ciphertext $\sigma^* = \langle \sigma_1^*, \sigma_2^*, \sigma_3^*, \sigma_4^*, \sigma_5^*, \tau^* \rangle$, which is sent to the adversary $\mathcal{A}$, is a valid signcryption on the message m_γ^* by the sender with identity ID_A^* for the receiver with identity ID_B^*. Otherwise, if T is a random element in $\mathbb{G}_T$, then challenge ciphertext is indistinguishable. So, in this case the simulator will give no information about the choice of γ that it made.

Training Phase II

In this phase, the simulator answers to the queries from the adversary $\mathcal{A}$ in the same way as it did in the Training Phase I. Here, $\mathcal{A}$ cannot ask for the Unsigncrypt query of the challenge ciphertext σ^* with sender identity as ID_A^* and receiver identity as ID_B^* and the Extract query for the receiver identity ID_B^*.

The strength of our scheme is that the adversary can again query the Signcrypt Oracle for the signcryption of either of the challenge ciphertexts m_0^* or m_1^* with the sender identity as ID_A^* and receiver identity as ID_B^*, during this phase. $\mathcal{A}$ can also query the Extract oracle for the sender identity ID_A^*, which makes our scheme insider secure.

Guess Phase

When the adversary $\mathcal{A}$ decides the training is enough, $\mathcal{A}$ outputs its guess γ' of γ. If the guess $\gamma' = \gamma$, then the simulator outputs that T in the given DBDH tuple is valid i.e $T = \hat{e}(g,g)^{abc}$. Otherwise $\mathcal{B}$ outputs that $\langle g, g^a, g^b, g^c, T \rangle$ is not valid DBDH tuple. Thus, $\mathcal{B}$ simulates a challenger for the adversary $\mathcal{A}$ and solves the DBDH problem with a probability ϵ' from the forgery produced by $\mathcal{A}$. This concludes the description of the simulation.

4.2 Unforgeability

We now prove the unforgeability property, *Strong Unforgeability under Chosen Message and Identity Attack*(SUF-CMIA) of our scheme with the following theorem.

Theorem 2. *If there exists an SUF-CMIA adversary for our scheme who can create valid ciphertexts during the SUF-CMIA game explained above, with a non-negligible probability ϵ when it runs for a polynomial time t, asking at most q_E extract queries, q_S signcrypt queries and q_{US} unsigncrypt queries, then there exists another algorithm, who can solve the Computational Diffie-Hellman (CDH) problem with probability ϵ' in polynomial time t', where*

$$\epsilon' \geq \frac{\epsilon}{4\kappa q_E(n_u+1)(n_m+1)}$$

$$t' \leq t + \mathcal{O}((n_u q_E + (n_u + l)(q_S + q_{US}))t_m + (q_E + q_S + q_{US})t_e + (q_S + q_{US})t_p)$$

where n_u is the length of the identity string and n_m is the length of the message, κ is the security parameter, t_m, t_e, t_p are the time required for each multiplication, each exponentiation and each bilinear pairing respectively and l is a value large enough such that the hash functions outputting $\{0,1\}^l$ in the scheme are collision resistant.

Due to page restrictions the proof of this theorem is omitted here.

5 Efficiency

The Signcrypt algorithm of our scheme performs one bilinear pairing operation while calculating $\hat{e}(g_1, h_2)^r$. But note that $\hat{e}(g_1, h_2)$ can be precomputed before the protocol begins since both g_1 and h_2 are public parameters and they are same for all runs of the protocol. The algorithm also performs 5 exponentiations (4 of elements of group $\mathbb{G}$ and one of element of $\mathbb{G}_T$). The unsigncrypt algorithm performs 6 bilinear pairing operations of which $\hat{e}(g_1, g_2)$ can be precomputed and one exponentiation of an element of group $\mathbb{G}$. Note that the calculation of $(h_1^{\lambda} h_3)^{\rho}$ involves only one exponentiation according to the well known "square and multiply" technique explained in [19]. When the number of computations performed by our scheme and the scheme in Li et al. [14] are compared (excluding the precomputed values), our scheme performs one exponentiation less than [14] with same number of bilinear pairings.

Since none of the ID based signcryption schemes without random oracles are provably secure in the literature, we will compare the efficiency of our scheme with the ID based signcryption scheme π that was conceptually formatted in [21] obtained by the 'Sign then Encrypt' approach. Note that π is the most efficient signcryption scheme that can be got by the direct combination of IBE and IBS schemes, since [20] and [12] are the most efficient IBS and IBE schemes with SUF-CMA and IND-CCA2 properties respectively in the standard model.

Table 1. Computational Complexity of π (Direct combination) and Ours

Scheme	**Secret key size**	**Ciphertext size**	**#pairings** S, US	**#exponentiations** S, US
π	$5\|p\|$	$2\|p\| + n_m$	0(+1), 5(+1)	8, 3
Ours	$3\|p\|$	$4\|p\| + n_m + l_\tau$	0(+1), 5(+1)	5, 1

The numbers shown in the brackets indicate the values that can be precomputed before the algorithm begins (and they remain same for all runs of the protocol).

6 Conclusion

We have presented the first secure ID based signcryption scheme and proven its security in the standard model. This scheme satisfies the strongest notions of security available for the signcryption schemes. Moreover, it has additional interesting properties such as public ciphertext verifiability which is very useful in the context of firewalls and spam filters. The security reduction is also tighter compared to many other schemes in the standard model. There is a trade-off in this scheme between the size of public parameters and the tightness to the underlying hard assumption. In our scheme we have included some extra parameters namely a unsigncryption key to increase the probability to a much larger value so that the security of our scheme is more tight to the underlying hard problem much more than the existing signcryption schemes. An interesting and potential future direction will be designing a more efficient protocol with reduced public parameters, key size and reduced ciphertext size.

Acknowledgements. We sincerely thank Prof. Qiong Huang for shepherding and for pointing us a subtle inconsistency in the proof. We also thank the anonymous reviewers of the ProvSec 2012 program committee for their insightful reviews.

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. Barreto, P.S.L.M., Libert, B., McCullagh, N., Quisquater, J.-J.: Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 515–532. Springer, Heidelberg (2005)

3. Bellare, M., 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)
4. Boneh, D., Shen, E., Waters, B.: Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 229–240. Springer, Heidelberg (2006)
5. Boyen, X.: Multipurpose Identity-Based Signcryption A Swiss Army Knife for Identity-based Cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003)
6. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM 51(4), 557–594 (2004)
7. Chen, L., Malone-Lee, J.: Improved Identity-Based Signcryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 362–379. Springer, Heidelberg (2005)
8. Chow, S.S.M., Yiu, S.-M., Hui, L.C.K., Chow, K.P.: Efficient Forward and Provably Secure ID-based Signcryption Scheme with Public Verifiability and Public Ciphertext Authenticity. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 352–369. Springer, Heidelberg (2004)
9. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
10. Huang, Q., Wong, D., Li, J., Zhao, Y.-M.: Generic transformation from weakly to strongly unforgeable signatures. Journal of Computer Science and Technology 23, 240–252 (2008), doi:10.1007/s11390-008-9126-y
11. Jin, Z., Wen, Q., Du, H.: An improved semantically-secure identity-based signcryption scheme in the standard model. Computers & Electrical Engineering 36(3), 545–552 (2010)
12. Kiltz, E., Vahlis, Y.: CCA2 Secure IBE: Standard Model Efficiency through Authenticated Symmetric Encryption. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 221–238. Springer, Heidelberg (2008)
13. Li, F., Liao, Y., Qin, Z.: Analysis of an identity-based signcryption scheme in the standard model. IEICE Transactions 94-A(1), 268–269 (2011)
14. Li, F., Muhaya, F.B., Zhang, M., Takagi, T.: Efficient Identity-Based Signcryption in the Standard Model. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 120–137. Springer, Heidelberg (2011)
15. Li, F., Takagi, T.: Secure identity-based signcryption in the standard model. Mathematical and Computer Modelling (2011), http://www.sciencedirect.com/science/article/pii/S0895717711003840
16. Libert, B., Quisquater, J.-J.: New identity based signcryption schemes from pairings. In: IEEE Information Theory Workshop 2003, pp. 155–158 (January 2003), extended version
17. Libert, B., Quisquater, J.-J.: Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 187–200. Springer, Heidelberg (2004)
18. Malone-Lee, J.: Identity-based signcryption. Cryptology ePrint Archive, Report 2002/098 (2002), http://eprint.iacr.org/
19. Mao, W.: Modern Cryptography: Theory and Practice. Prentice Hall Professional Technical Reference (2003)
20. Paterson, K.G., Schuldt, J.C.N.: Efficient Identity-Based Signatures Secure in the Standard Model. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 207–222. Springer, Heidelberg (2006)

21. Sharmila Deva Selvi, S., Sree Vivek, S., Vinayagamurthy, D., Pandu Rangan, C.: On the security of ID based signcryption schemes. Cryptology ePrint Archive, Report 2011/664 (2011), http://eprint.iacr.org/
22. 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)
23. Teranishi, I., Oyama, T., Ogata, W.: General Conversion for Obtaining Strongly Existentially Unforgeable Signatures. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. LNCS, vol. 4329, pp. 191–205. Springer, Heidelberg (2006)
24. Wang, X., Qian, H.F.: Attacks against two identity-based signcryption schemes. In: Second International Conference on Networks Security Wireless Communications and Trusted Computing (NSWCTC), vol. 1, pp. 24–27 (April 2010)
25. Wang, X.A., Zhong, W., Luo, H.: Cryptanalysis of efficient identity based signature/signcryption schemes in the standard model. In: 2010 International Symposium on Intelligence Information Processing and Trusted Computing (IPTC), pp. 622–625 (October 2010)
26. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
27. Yanli, R., Dawu, G.: Efficient identity based signature/signcryption scheme in the standard model. In: The First International Symposium on Data, Privacy, and E-Commerce, ISDPE 2007, pp. 133–137 (2007)
28. Yu, Y., Yang, B., Sun, Y., Zhu, S.: Identity based signcryption scheme without random oracles. Computer Standards & Interfaces 31(1), 56–62 (2009)
29. Zhang, B.: Cryptanalysis of an identity based signcryption scheme without random oracles. Journal of Computational Information Systems 6(6), 1923–1931 (2010)
30. Zhang, M., Li, P., Yang, B., Wang, H., Takagi, T.: Towards Confidentiality of ID-Based Signcryption Schemes under without Random Oracle Model. In: Chen, H., Chau, M., Li, S.-H., Urs, S., Srinivasa, S., Wang, G.A. (eds.) PAISI 2010. LNCS, vol. 6122, pp. 98–104. Springer, Heidelberg (2010)
31. Zheng, Y.: Digital Signcryption or How to Achieve Cost (Signature & Encryption) << Cost(Signature) + Cost(Encryption). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)

Combined Public-Key Schemes: The Case of ABE and ABS

Cheng Chen[1], Jie Chen[2], Hoon Wei Lim[2], Zhenfeng Zhang[1], and Dengguo Feng[1]

[1] Institute of Software, Chinese Academy of Sciences, Beijing, China
{chencheng,zfzhang,feng}@is.iscas.ac.cn
[2] Division of Mathematical Sciences,
School of Physical & Mathematical Sciences,
Nanyang Technological University, Singapore
s080001@e.ntu.edu.sg, hoonwei@ntu.edu.sg

Abstract. In the context of public key cryptography, combined encryption and signature schemes have attractive properties and are sometimes used in practice. The topic of joint security of signature and encryption schemes has a fairly extensive history. In this paper, we focus on the combined public-key schemes in attribute-based setting. We present a security model for combined CP-ABE and ABS schemes in the joint security setting. An efficient concrete construction of CP-ABE and ABS based on Waters's CP-ABE scheme is proposed. Our scheme is proved to be selectively jointly secure in standard model under reasonable assumptions. Moreover, we consider the problem of how to build attribute-based signcryption (ABSC) and obtain an ABSC scheme and show that it is secure. We also give a general construction of combined ABSC, CP-ABE and ABS schemes from combined CP-ABE and ABS schemes.

1 Introduction

Combining Encryption and Signature. It is common practice that cryptographic key pairs used for encryption and signatures are independently chosen to minimize the risk of key exposure. However, there are also scenarios in which the same key pair is used instead for both encryption and signing operations to reduce key storage requirements and other related costs, such as key certification and verification. This is first formally studied by Haber and Pinkas [9], who introduced the term *combined public-key schemes*—an encryption scheme and a signature scheme are combined such that they share the same key generation algorithm, while the existing encrypt/decrypt and sign/verify algorithms are preserved. They also presented a security model reflecting the *joint security* of the combined public-key schemes. In the model, an adversary attacking the encryption component is allowed access to not only the decryption oracle, but also the signing oracle. Similarly, an adversary attacking the signature component is given access to both the signing and the decryption oracles.

Subsequently, Vasco et al. [23] showed that the Boneh-Franklin identity-based encryption (IBE) scheme [3] and the Hess identity-based signature (IBS) scheme

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 53–69, 2012.

[11] can be combined with provably joint security. Moreover, Paterson et al. [19] recently proposed combined public-key schemes having joint security in the standard model, a property that has not been achieved in the previous work. Their schemes make use of IBE, (pairing-based) short signatures and a data encapsulation mechanism (DEM) as building blocks.

Combining ABE and ABS? In this work, we extend the field to the attribute-based setting. Particularly, we ask the question of whether or not it is possible to *securely combine* attribute-based encryption (ABE) [21] and attribute-based signatures (ABS) [16].

ABE can be categorized into ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE). (ABS has the same analogues.) In the former, a secret key is associated with an attribute set. A user can decrypt a ciphertext if and only if the attribute set satisfies the access structure associated with the ciphertext. In the latter, on the contrary, a secret key is associated with an access structure. A user can decrypt a ciphertext associated with an attribute set if and only if the attribute set satisfies the access structure associated with the user's secret key.

In this paper, we focus on combining CP-ABE and the analogue in ABS. Consider in some scenario (attribute-based messaging system in the campus), both the functionality of ABE and ABS is required for data confidentiality and authentication. At the same time, the attribute representation is the same for one user, i.e. one has the attributes "student" and "computer science" in a campus. Thus, we need two separated ABE and ABS schemes, where two authorities are required to authenticate attributes and each user needs to keep two secret keys with the same attribute subset in the system. It is unrealistic and inefficient. Combined CP-ABE and ABS scheme is a solution for the above problems. Henceforth, we use CCP-ABES as the abbreviation for combined CP-ABE and ABS. With CCP-ABES, our goal is that each user can use the same secret key (associated with some attribute set) to decrypt an attribute-based ciphertext, sign an attribute-based signature, or perform attribute-based key-exchange [8,28], with joint security in the standard model.

We note that, as explained in [19], it is easy to construct a general combined public-key scheme from IBE: the signature component is constructed through the Naor transform and the public-key encryption component through a tag-based version of the CHK transform [4]. Here, signatures from the Naor transform are simply secret keys in IBE, and the secret keys can be used to decrypt ciphertexts from the CHK transform. Hence, one can simply use a bit prefix in the identity space to provide domain separation between the signatures and secret keys, while allowing the use of the same IBE key generation (or extraction) algorithm for both encryption and signing operations.

However, it is not that straightforward to construct a CCP-ABES scheme. The reason is that: for any IBE scheme, the identity space is at least super-polynomial, and thus it is easy to encode a message or a verification key of a one-time signature scheme as a particular "ID". However, this is not always true for in the attribute-based setting if we were to apply the same method to a

delegatable[1] CP-ABE scheme. As a matter of fact, the attribute space of a secret key (by delegation) is usually small (even in large universe cases). Should we use the aforementioned method (treating a message as an attribute analogous to an indentity), we can afford to sign only short messages. As a consequence, the resulting one-time signatures are no longer secure when the corresponding verification keys are short. Furthermore, for the signature component of CCP-ABES to achieve unforgeability under an adaptive chosen message attack, we require that the corresponding CP-ABE be adaptively secure.

Another challenge is that we require that signatures from ABS leak no information about the associated attributes, i.e. *attribute privacy* (or perfect privacy)—an essential property of ABS. When signing a predicate by simply delegating to the subset of attributes that satisfies the predicate, the signature (the delegated decryption key for the predicate) usually reveals information related to the user's attribute set. This is because an access structure may be satisfied by different attribute sets (secret keys) in different ways.

Our Techniques. Using an IND-CPA secure CP-ABE scheme with delegation, users can delegate some of their attributes to a new secret key by running the delegate algorithm. We make use of this feature in our CCP-ABES scheme. For the ABS component of our CCP-ABES scheme, this delegation process becomes a signing operation; while for the IND-CCA secure CP-ABE component, our decrypt algorithm first runs the delegate algorithm, and then uses the delegated key to decrypt the ciphertext.

We encode a predicate and message pair to be signed or a tag to achieve IND-CCA security by some special attributes we call *dummy attributes*. Suppose each user owns a set of dummy attributes. In the ABS component of our CCP-ABES, we use a dummy attribute for each predicate and message pair $(\mathbb{A}, m)$ to be signed. We then use other dummy attributes which intersect with the dummy attributes used in the ABS component for the CP-ABE component. Moreover, one dummy attribute is used for each tag required to achieve IND-CCA security. In our construction, we then apply programmable hash functions to efficiently realize the public parameters and the secret key elements for the dummy attributes. Particularly, we use different programmable hash functions to encode the predicate and message pairs to be signed in the ABS component and the tags to achieve IND-CCA security in the CP-ABE component, respectively. This is also to provide domain separation between the signatures and secret keys.

The adaptive security required in the ABS component for the dummy attributes can be easily obtained from the properties of a programmable hash function, such as that by Waters [24]. Interestingly, our IND-CCA secure CP-ABE component is more efficient than the previous ABE schemes achieving IND-CCA security [27] because ours replies only one dummy attribute in the ciphertext.

Our Contributions. Firstly, we present a security model for combined CP-ABE and ABS (CCP-ABES) schemes in the joint security setting. There are

[1] The delegatability property of ABE mimics the key extraction functionality in IBE.

two essential security definitions with regards to joint security for CCP-ABES, namely, ciphertext indistinguishability under a chosen ciphertext attack (IND-CCA) for the CP-ABE component, and existential unforgeability under an adaptive chosen message attack (EUF-CMA) for the ABS component. As with previous security models for combined public-key schemes, in both security games, the adversary is allowed to query the secret key generation, decryption and signing oracles. Furthermore, we consider perfect privacy for the ABS component, as with standard ABS schemes. This property ensures that the distribution of signatures created from different secret keys for a predicate and message pair is the same.

We then give a concrete construction of CCP-ABES. Our scheme is based on the Waters CP-ABE scheme and is proven to be selectively and jointly secure under well-established assumptions in the standard model. Moreover, the ABS component of our CCP-ABES scheme achieves perfect privacy under unbounded adversaries. Our construction is significantly more efficient than considering CP-ABE and ABS separately. For example, our secret keys have $|S| + 2$ group elements in comparison with $2|S| + 4$ group elements in the latter, where $|S|$ is the number of attributes. Further efficiency comparisons can be found in Section 4.2.

Finally, as a further contribution that is of independent interest, we apply our idea of CCP-ABES to attribute-based signcryption (ABSC). We obtain an ABSC scheme and show that it is jointly secure. We also give a general construction of combined ABSC, CP-ABE and ABS schemes from CCP-ABES schemes. Using our CCP-ABES construction, we show that the resulting ABSC scheme is more efficient and expressive than previous work.

2 Preliminaries

2.1 Bilinear Groups and Complexity Assumptions

We present some basic concepts on groups with efficiently computable bilinear maps. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$ and e be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that $e(g, g) \neq 1$ for g and for any u, $v \in \mathbb{Z}_p$, it holds that $e(g^u, g^v) = e(g, g)^{uv}$. We say that $\mathbb{G}$ is a bilinear group if the group operation in $\mathbb{G}$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ are both efficiently computable. Notice that the map e is symmetric since $e(g^u, g^v) = e(g, g)^{uv} = e(g^v, g^u)$.

We define the decisional bilinear Diffie-Hellman exponent (BDHE) and Diffie-Hellman exponent (DHE) assumptions as follows.

Definition 1. *Let κ be a security parameter. $\mathbb{G}$ is a bilinear group of prime order p, where $p > 2^\kappa$. g is an independent generators of $\mathbb{G}$. Denote $\overrightarrow{y}_{g,\gamma,n} = (g_1, g_2, \ldots, g_n, g_{n+2}, \ldots, g_{2n}) \in \mathbb{G}^{2n-1}$, where $g_i = g^{\gamma^i}$ for some unknown $\gamma \in \mathbb{Z}_p$. We define the advantage function $Adv^{n\text{-}BDHE}_{\mathbb{G},\mathcal{B}}(\kappa)$ of an adversary $\mathcal{B}$ as*

$$\left| \Pr[\mathcal{B}(g, g^s, \overrightarrow{y}_{g,\gamma,n}, e(g_{n+1}, g^s)) = 0] - \Pr[\mathcal{B}(g, g^s, \overrightarrow{y}_{g,\gamma,n}, Z) = 0] \right|$$

where the probability is over the random choices of $g \in \mathbb{G}$, $Z \in \mathbb{G}_T$ *and* $s, \gamma \in \mathbb{Z}_p$. *We say that the decision n-BDHE assumption holds in* $\mathbb{G}$ *if* $Adv^{n\text{-}BDHE}_{\mathbb{G},\mathcal{B}}(\kappa)$ *is negligible for any probabilistic polynomial time (PPT) adversaries in* κ.

Definition 2. *Let* κ *be a security parameter.* $\mathbb{G}$ *is a bilinear group of prime order p, where* $p > 2^\kappa$. g *is an independent generator of* $\mathbb{G}$. *Denote* $\overrightarrow{y}_{g,\gamma,n} = (g_1, g_2, \ldots, g_n, g_{n+2}, \ldots, g_{2n}) \in \mathbb{G}^{2n-1}$, *where* $g_i = g^{\gamma^i}$ *for some unknown* $\gamma \in \mathbb{Z}_p$. *We define the advantage function* $Adv^{n\text{-}DHE}_{\mathbb{G},\mathcal{B}}(\kappa)$ *of an adversary* $\mathcal{B}$ *as*

$$\Pr[\mathcal{B}(g, \overrightarrow{y}_{g,\gamma,n}) = g_{n+1}]$$

where the probability is over the random choices of $g \in \mathbb{G}$ *and* $\gamma \in \mathbb{Z}_p$. *We say that the n-DHE assumption holds in* $\mathbb{G}$ *is if* $Adv^{n\text{-}DHE}_{\mathbb{G},\mathcal{B}}(\kappa)$ *is negligible for any PPT adversaries in* κ.

2.2 Symmetric Encryption

Let κ be a security parameter. A symmetric encryption scheme $\mathsf{SE} := (\mathcal{E}, \mathcal{D})$ is specified by its encryption algorithm $\mathcal{E}$ (encrypting $m \in MsgSp(\kappa)$ with keys $K \in \mathcal{K}(\kappa)$) and decryption algorithm $\mathcal{D}$ (returning $m \in MsgSp(\kappa) or \ \perp$). Here we restrict ourselves to deterministic algorithms $\mathcal{E}$ and $\mathcal{D}$.

The most common notion of security for symmetric encryption is that of ciphertext indistinguishability, which requires that all efficient adversaries fail to distinguish between the encryptions of two messages of their choice. Another common security requirement is ciphertext authenticity. Ciphertext authenticity requires that no efficient adversary can produce a new valid ciphertext under some key when given one encryption of a message of his choice under the same key. A symmetric encryption scheme which satisfies both requirements simultaneously is called secure in the sense of authenticated encryption (AE-OT secure). Note that AE-OT security is a stronger notion than chosen-ciphertext security.

The above requirements are formalized as follows:

Ciphertext Indistinguishability. Let $\mathsf{SE} := (\mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme, and let $\mathcal{A}$ be an adversary. We define the following experiment:

- The challenger chooses a random key $K \in \mathcal{K}(\kappa)$.
- $\mathcal{A}$ submits two message $m_0, m_1 \in MsgSp(\kappa)$ to the challenger. The challenger randomly chooses $b \in \{0, 1\}$ and sends the ciphertext $c^* := \mathcal{E}_K(m_b)$ to $\mathcal{A}$.
- $\mathcal{A}$ eventually outputs a guess b' for b. If $b' = b$, it returns 1 else returns 0.

The advantage of $\mathcal{A}$ in breaking the ciphertext indistinguishability security of SE is:

$$Adv^{IND}_{\mathsf{SE},\mathcal{A}}(\kappa) \stackrel{def}{=} \left| \Pr[Exp^{IND}_{\mathsf{SE},\mathcal{A}}(\kappa) = 1] - 1/2 \right|$$

Definition 3. *The symmetric encryption scheme* SE *has indistinguishable ciphertexts if for any PPT adversary* $\mathcal{A}$ *the advantage* $Adv^{IND}_{\mathsf{SE},\mathcal{A}}(\kappa)$ *is negligible.*

Ciphertext Authenticity. In this work we are only interested in one-time authenticated schemes. That is, schemes for which no efficient adversary can produce a new valid ciphertext after seeing the encryption of a single message.

Let $\mathsf{SE} = (\mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme, and let $\mathcal{A}$ be an adversary. We define the following experiment:

- The challenger chooses a random key $K \in \mathcal{K}(\kappa)$.
- $\mathcal{A}$ submits a message $m \in \mathit{MsgSp}(\kappa)$ to the challenger and receives the ciphertext $c := \mathcal{E}_K(m)$ to $\mathcal{A}$.
- $\mathcal{A}$ eventually outputs a ciphertext c'. If $c' \neq c$ and $\mathcal{D}_K(c') \neq \perp$, it returns 1 else returns 0.

The advantage of $\mathcal{A}$ in breaking the ciphertext indistinguishability security of SE is:

$$Adv^{\text{CT-INT}}_{\mathsf{SE},\mathcal{A}}(\kappa) \stackrel{def}{:=} \Pr[Exp^{\text{CT-INT}}_{\mathsf{SE},\mathcal{A}}(\kappa) = 1]$$

Definition 4. *The symmetric encryption scheme* SE *has ciphertext integrity if for any PPT adversary* $\mathcal{A}$ *the advantage* $Adv^{\text{CT-INT}}_{\mathsf{SE},\mathcal{A}}(\kappa)$ *is negligible in* κ.

2.3 Collision-Resistant Hash

Let κ be a security parameter. Collision-resistant hash functions are a family of keyed hash functions. Let $\mathsf{H} := \{\mathsf{H}^s\}_{s\in\{0,1\}^k}$ is a family of hash functions for each k-bit key s where k is polynomial in security parameter κ. H is said to be collision-resistant if, for any hash function H^s in H, it is infeasible for a PPT adversary to find two distinct $y \neq x$ such that $\mathsf{H}^s(x) = \mathsf{H}^s(y)$. Let $\mathcal{B}$ denote an adversary against collision-resistant hash functions, we define the advantage function of the adversary $\mathcal{B}$ as follows:

$$Adv^{CR}_{\mathsf{H},\mathcal{B}}(\kappa) := \Pr[x \neq y \wedge \mathsf{H}^s(x) = \mathsf{H}^s(y) : s \leftarrow_R \{0,1\}^k]$$

H is collision resistant if for any PPT adversary $\mathcal{B}$, $Adv^{CR}_{\mathsf{H},\mathcal{B}}(\kappa)$ is negligible in κ.

2.4 Access Structure and Secret-Sharing Scheme

Definition 5. *Let* $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ *be a set of parties. An access structure is a set collection* $\mathbb{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}} \backslash \varnothing$. *An access structure is monotone if* $\forall B, C$: *if* $B \in \mathbb{A}$ *and* $B \subseteq C$ *then* $C \in \mathbb{A}$. *The sets in* $\mathbb{A}$ *are called the authorized sets, and the sets not in* $\mathbb{A}$ *are called the unauthorized sets.* $min(\mathbb{A})$ *is called a minterm of* $\mathbb{A}$ *if* $B \in min(\mathbb{A})$, *and for every* $C \subsetneq B$, *the set* C *is unauthorized.*

When used in ABE or ABS schemes, we replace the parties by attributes. Similarly, the access structure is corresponding to an attributes set collection which will contain the authorized sets of attributes.

We will make use of linear secret-sharing schemes in our construction.

Definition 6 (Linear Secret-Sharing Schemes (LSSS)). *A secret-sharing scheme* Π *over a set of parties* $\mathcal{P}$ *is called linear (over* $\mathbb{Z}_p$*) if*

- *The shares for each party form a vector over* $\mathbb{Z}_p$.
- *There exists a matrix* $\mathbf{M}$ *with* ℓ *rows and* n *columns called the share-generating matrix for* Π. *For all* $i = 1, \ldots, \ell$, *the* i*-th row of* $\mathbf{M}$ *we let the function* ρ *defined the party labeling row* i *as* $\rho(i)$. *When we consider the column vector* $\mathbf{v} = (s, y_2, \ldots, y_n)$, *where* $s \in \mathbb{Z}_p$ *is the secret to be shared, and* $y_2, \ldots, y_n \in \mathbb{Z}_p$ *are randomly chosen, then* $\mathbf{M} \cdot \mathbf{v}$ *is the vector of* ℓ *shares of the secret* s *according to* Π. *The share* $\lambda_i = \mathbf{M}_i \cdot \mathbf{v}$ *belongs to party* $\rho(i)$.

For $S \in \mathbb{A}$ *be any authorized set, and let* $I := \{i : \rho(i) \in S\}$. *Then, there exist coefficients* $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ *such that* $\sum_{i \in I} \omega_i \cdot \lambda_i = s$. *Furthermore, these coefficients* $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ *can be found in time polynomial in the size of the share-generating matrix* $\mathbf{M}$.

Remarks. We note that we use the convention that vector $(1, 0, \ldots, 0)$ is the target vector for any linear secret-sharing scheme. For any satisfying set of rows I in $\mathbf{M}$, we will have that the target vector is in the span of I: $\sum_{i \in I} \omega_i \cdot \mathbf{M}_i = (1, 0, \ldots, 0)$. $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ are the coefficients used to reconstruct the secret above.

3 Combined CP-ABE and ABS

A CCP-ABES scheme can been seen as an extension of the combined signature and encryption scheme [19] in attribute-based setting. A CCP-ABES scheme shares a setup algorithm and a key generation algorithm. Hence, the same public parameters and master secret key are used in a CCP-ABES scheme. A single secret key specified by user's attributes is used for both signing and decrypting. It comprises a tuple of algorithms (**Setup**, **KeyGen**, **Sign**, **Verify**, **Encrypt**, **Decrypt**) such that (**Setup**, **KeyGen**, **Sign**, **Verify**) form an ABS scheme and (**Setup**, **KeyGen**, **Encrypt**, **Decrypt**) form a CP-ABE scheme. We show the algorithms in the following:

- **Setup**(κ, U). The algorithm takes a security parameter κ and an attribute universe description U as input, and outputs public parameters PP and a master secret key MSK.
- **KeyGen**$(\mathsf{PP}, \mathsf{MSK}, S)$. The algorithm takes as input public parameters PP, a master key MSK and a set of attributes $S \subseteq \mathsf{U}$, and returns a secret key SK_S associated with S.
- **Encrypt**$(\mathsf{PP}, m, \mathbb{A})$. The algorithm takes as input public parameters PP, a message m and an access structure $\mathbb{A}$ over U. It returns a ciphertext CT such that a secret key generated from the attribute set S can be used to decrypt CT if and only if $S \in \mathbb{A}$. And we assume that $\mathbb{A}$ is implicitly included in the ciphertext.
- **Decrypt**$(\mathsf{PP}, \mathsf{CT}, \mathsf{SK}_S)$. The algorithm takes as input public parameters PP, a ciphertext CT, which contains an access structure $\mathbb{A}$, and a secret key SK_S associated with an attribute set S. It returns the message m if $S \in \mathbb{A}$.

- **Sign**$(\mathsf{PP}, \mathsf{SK}_S, \mathbb{A}, m)$. The algorithm takes as input public parameters PP, a message m, an access structure $\mathbb{A}$ and a signature generation key SK_S associated with an attribute set S such that $S \in \mathbb{A}$. It outputs a signature σ. And we assume that $\mathbb{A}$ is implicitly included in the signature.
- **Verify**(PP, σ, m). This takes as input a message m, an access structure $\mathbb{A}$, a signature σ with an access structure $\mathbb{A}$ and public parameters PP. It outputs 1 if the signature is valid and 0 otherwise.

For correctness of attribute-based signature, if $S \in \mathbb{A}$, we require that

$$\mathbf{Verify}(\mathsf{PP}, \mathbf{Sign}(\mathsf{PP}, \mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S), \mathbb{A}, m), m) = 1$$

where $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathbf{Setup}(\kappa, \mathsf{U})$.

When defining the security of a CCP-ABES scheme, we consider the notions of existential unforgeability of the ABS component under adaptive chosen message attacks (EUF-CMA) and indistinguishability of the CP-ABE component under chosen ciphertext attacks (IND-CCA). The two notions need to be extended to reflect an adversary's ability to request both signatures and decryptions. In the EUF-CMA security of an ABS component, it is necessary to provide the adversary with the access to the decryption oracle of CP-ABE component. In the IND-CCA security of a CP-ABE component, it is necessary to provide the adversary with the access to the signature oracle of ABS component. The security definitions are given formally here.

Definition 7 (EUF-CMA security in the presence of a decryption oracle). *Let κ be a security parameter and $\Pi :=$ (**Setup**, **KeyGen**, **Encrypt**, **Decrypt**, **Sign**, **Verify**) be a CCP-ABES scheme. Existential unforgeability of the ABS component under adaptive chosen message attacks in the presence of an additional decryption oracle of CP-ABE component is defined through the following game between a challenger and an adversary $\mathcal{A}$.*

Setup: *The challenger runs* **Setup**(κ, U)*, and sends the public parameters* PP *to* $\mathcal{A}$.
Queries: $\mathcal{A}$ *can make secret key, signature and decryption queries.*

- **Secret key queries:** $\mathcal{A}$ *adaptively chooses an attribute set* $S \subseteq \mathsf{U}$ *and receives the secret key* $\mathsf{SK}_S :=$ **KeyGen**$(\mathsf{PP}, \mathsf{MSK}, S)$ *from the challenger.*
- **Signature queries:** $\mathcal{A}$ *adaptively chooses a pair* $(\mathbb{A}, m)$ *consisting of an access structure* $\mathbb{A}$ *and a message* m*. The challenger chooses an arbitrary set* $S \in \mathbb{A}$ *and computes a signature* $\sigma :=$ **Sign**$(\mathsf{PP},$ **KeyGen**$(\mathsf{PP}, \mathsf{MSK}, S), \mathbb{A}, m)$ *which is returned to* $\mathcal{A}$.
- **Decryption queries:** $\mathcal{A}$ *adaptively chooses a ciphertext* CT *and an attribute set* S *associated with the secret key used to decrypt. The challenger returns the output of* **Decrypt**$(\mathsf{PP}, \mathsf{CT},$ **KeyGen**$(\mathsf{PP}, \mathsf{MSK}, S))$ *to* $\mathcal{A}$. [2]

[2] We define the decryption queries for CP-ABE component as the definition given in [27].

Forgery: *At the end of the game, $\mathcal{A}$ outputs a tuple (m^*, σ^*), where σ^* consists a challenge signing access structure $\mathbb{A}^*$. $\mathcal{A}$ wins if:*

- *$\mathcal{A}$ has not made any signature query for the pair $(\mathbb{A}^*, m^*)$.*
- *None of the attribute sets in secret key queries phase is authorized for the challenge signing access structure $\mathbb{A}^*$.*
- $\mathbf{Verify}(\mathsf{PP}, \sigma^*, m^*) = 1$.

For the ABS component of a CCP-ABES scheme Π, the probability of an adversary $\mathcal{A}$ succeeding in breaking existential unforgeability under chosen message attacks is defined as

$$Succ_{\mathcal{A},\Pi}^{EUF\text{-}CMA}(\kappa) := \Pr[\mathcal{A}\ wins]$$

We say the ABS component of a CCP-ABES scheme Π is EUF-CMA secure in the presence of a decryption oracle if $Succ_{\mathcal{A},\Pi}^{EUF\text{-}CMA}(\kappa)$ is negligible with respect to the security parameter κ, for any PPT adversary $\mathcal{A}$.

Additionally, we say that the ABS component of a CCP-ABES scheme Π is selectively EUF-CMA secure in the presence of a decryption oracle if we add an Init stage before setup where the adversary selects a challenge signing access structure $\mathbb{A}^*$ to attack.

Definition 8 (IND-CCA security in the presence of a signing oracle). *Let κ be a security parameter and $\Pi :=$ ($\mathbf{Setup}, \mathbf{KeyGen}, \mathbf{Encrypt}, \mathbf{Decrypt}$, $\mathbf{Sign}, \mathbf{Verify}$) be a CCP-ABES scheme. Indistinguishability of the CP-ABE component under chosen ciphertext attacks in the presence of an additional signing oracle of the ABS component is defined through the following game between a challenger and an adversary $\mathcal{A}$.*

Setup: *The challenger runs $\mathbf{Setup}(\kappa, \mathsf{U})$, and gives the public parameters PP to the adversary.*
Phase 1: *$\mathcal{A}$ can make secret key, signature and decryption queries.*

- **Secret key queries:** *$\mathcal{A}$ adaptively chooses an attribute set $S \subseteq \mathsf{U}$ and receives the secret key $\mathsf{SK}_S := \mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S)$ from the challenger.*
- **Signature queries:** *$\mathcal{A}$ adaptively chooses a pair $(\mathbb{A}, m)$ consisting of a message m and an access structure $\mathbb{A}$. The challenger chooses an arbitrary set $S \in \mathbb{A}$ and computes a signature $\sigma :=$ $\mathbf{Sign}(\mathsf{PP},$ $\mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S), \mathbb{A}, m)$ which is returned to $\mathcal{A}$.*
- **Decryption queries:** *$\mathcal{A}$ adaptively chooses a ciphertext CT and an attribute set S associated with the secret key used to decrypt. The challenger returns the output of $\mathbf{Decrypt}(\mathsf{PP}, \mathsf{CT}, \mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S))$ to $\mathcal{A}$.*

Challenge: *The adversary $\mathcal{A}$ submits two messages m_0 and m_1 of equal length and a challenge encrypting access structure $\mathbb{A}^*$. The challenger chooses $\mu \in \{0, 1\}$ at random and encrypts m_μ under $\mathbb{A}^*$. The resulting ciphertext CT^* is given to the adversary.*

Phase 2: *The adversary can continue to make queries as Phase 1 but with the restriction that $\mathcal{A}$ must not request the decryption oracle for the challenge ciphertext CT^*.*

Guess. *Finally, the adversary outputs a guess μ' of μ. We say that $\mathcal{A}$ wins the game if none of the attribute sets in Phase 1 and Phase 2 that satisfies the challenge encrypting access structure $\mathbb{A}^*$ has been queried and $\mu' = \mu$. And the probability is defined as $Succ_{\mathcal{A},\Pi}^{IND\text{-}CCA}(\kappa)$.*

For a CCP-ABES scheme Π, the advantage of an adversary $\mathcal{A}$ in the above game is defined as

$$Adv_{\mathcal{A},\Pi}^{IND\text{-}CCA}(\kappa) := \left| Succ_{\mathcal{A},\Pi}^{IND\text{-}CCA}(\kappa) - \frac{1}{2} \right|$$

We say the CP-ABE component of a CCP-ABES scheme Π is IND-CCA secure in the presence of a signing oracle if $Adv_{\mathcal{A},\Pi}^{IND\text{-}CCA}(\kappa)$ is negligible with respect to the security parameter κ, for any PPT adversary $\mathcal{A}$.

We say that the CP-ABE component of a CCP-ABES scheme Π is selectively IND-CCA secure in the presence of a signing oracle of ABS component if we add an Init stage before setup where the adversary commits to the challenge encrypting access structure $\mathbb{A}^*$.

Informally, we say that a CCP-ABES scheme is (selectively) jointly secure if it is both (selectively) EUF-CMA secure in the presence of a decryption oracle of CP-ABE component and (selectively) IND-CCA secure in the presence of a signing oracle of ABS component.

Perfectly Private. Perfect privacy is a typical requirement for an ABS scheme. This property ensures that for an attribute-based signature the verifier only knows that the actual attributes that have been used to sign satisfy the specified signing predicate. Perfect privacy must hold even against an unbounded adversary with the master secret key. In the ABS component of the CCP-ABES scheme, it is necessarily to consider this security property.

Definition 9. *The ABS component of a CCP-ABES scheme is perfectly private, if for any message m, any attribute sets S_1, S_2 and any access structure $\mathbb{A}$ such that $S_1 \in \mathbb{A}, S_2 \in \mathbb{A}$, the distribution of $\mathbf{Sign}(\mathsf{PP}, \mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S_1), \mathbb{A}, m)$ is identical to that of $\mathbf{Sign}(\mathsf{PP}, \mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S_2), \mathbb{A}, m)$.*

4 Our Construction

In this section, we give a concrete CCP-ABES construction that is selectively and jointly secure in the standard model. Our scheme, built upon the CP-ABE scheme by Waters [26], supports general access structure and is realized by LSSS. We denote an access structure by a pair $(\mathbf{M}, \rho)$, where $\mathbf{M}$ is a matrix used to realize the access structure and ρ is an injective function used to map the rows of matrix $\mathbf{M}$ to the corresponding attributes.

– **Setup**(κ, U). On input a security parameter κ, and an attribute universe U:
  1. Choose groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\kappa$ and two collision-resistant hash functions $\mathsf{H}_1 : \{0,1\}^* \to \{0,1\}^k$, $\mathsf{H}_2 : \{0,1\}^* \to \mathbb{Z}_p$.
  2. Choose a symmetric encryption scheme $\mathsf{SE} := (\mathcal{E}, \mathcal{D})$ with key-space $\mathcal{K} := \mathbb{G}_T$.
  3. Pick random group elements $u, h, v_0, v_1, \ldots, v_k \in \mathbb{G}$. Here $(v_0, v_1, \ldots, v_k)$ are the public descriptions of the programmable hash function $\mathsf{PH}_1(\omega) : \{0,1\}^k \to \mathbb{G}$ evaluated as $\mathsf{PH}_1(\omega) := v_0 \prod_{i=1}^k (v_i)^{\omega_i}$, where ω is parsed as a k-bit string $\omega_1, \ldots, \omega_k$, while u, h are the public descriptions of the programmable hash function $\mathsf{PH}_2(\delta) : \mathbb{Z}_p \to \mathbb{G}$ evaluated as $\mathsf{PH}_2(\delta) := u^\delta h$.
  4. Pick random group elements $g, \{h_x\}_{x \in \mathsf{U}} \in \mathbb{G}$ and exponents $\alpha, a \in \mathbb{Z}_p$.
  5. Output the public parameters and the master secret key:

$$\mathsf{PP} := (g, e(g,g)^\alpha, g^a, u, h, \{h_x\}_{x \in \mathsf{U}}, v_0, v_1, \ldots, v_k), \quad \mathsf{MSK} := (\alpha).$$

– **KeyGen**$(\mathsf{PP}, \mathsf{MSK}, S)$: On input the public parameters PP, the master secret key MSK, and an attribute set $S \subseteq \mathsf{U}$, the algorithm picks a random value $t \in \mathbb{Z}_p$ and creates a secret key as

$$\mathsf{SK}_S := \left(D := g^{\alpha + at},\ L := g^t,\ \{D_x := h_x^t\}_{x \in S}\right).$$

– **Enc**$(\mathsf{PP}, \mathbb{A} = (\mathbf{M}_{(\ell \times n)}, \rho), m)$: On input the public parameters PP, an access structure $\mathbb{A} = (\mathbf{M}_{(\ell \times n)}, \rho)$ and a message $m \in \mathit{MsgSp}(\kappa)$:
  1. Pick a random vector $\mathbf{v} := (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$ and set $\lambda_i := \langle \mathbf{v}, \mathbf{M}_i \rangle$, for $i = 1, \ldots, \ell$, where $\mathbf{M}_i$ is the i-th row of $\mathbf{M}$.
  2. Compute $K := e(g,g)^{\alpha s},\ \delta := \mathsf{H}_2(C_0, C_1, \ldots, C_\ell, \mathbb{A})$.
  3. Output the ciphertext as

$$\mathsf{CT} := \left(C := \mathcal{E}_K(m),\ C_0 := g^s,\ \{C_i := g^{a\lambda_i} h_{\rho(i)}^{-s}\}_{i=1,\ldots,\ell}, C' := (\mathsf{PH}_2(\delta))^s\right).$$

– **Dec**$(\mathsf{PP}, \mathsf{CT}, \mathsf{SK}_S)$: On input the public parameters PP, a ciphertext CT parsed as $(C, C_0, \{C_i\}_{i=1,\ldots,\ell}, C', \mathbb{A})$ and a secret key SK_S parsed as $(D, L, \{D_x\}_{x \in S})$:
  1. Compute an index set $I := \{i : \rho(i) \in S\}$ and $\{\alpha_i\}_{i \in I}$ such that $\sum_{i \in I} \alpha_i \cdot \mathbf{M}_i = (1, 0, \ldots, 0)$. Note that such I always exists if $\mathbb{A}$ accepts S.
  2. Compute $\delta := \mathsf{H}_2(C_0, C_1, \ldots, C_\ell, \mathbb{A})$ and the symmetric key K as

$$K := \frac{e(C_0, D \cdot \mathsf{PH}_2(\delta))}{e(C', g) \cdot \prod_{i \in I} \left(e(C_i, L) \cdot e(C_0, D_{\rho(i)})\right)^{\alpha_i}}.$$

  3. Output the message $m := \mathcal{D}_K(C)$.

– **Sign**$(\mathsf{PP}, \mathbb{A} = (\mathbf{M}_{(\ell \times n)}, \rho), m, \mathsf{SK}_S)$: On input the public parameters PP, an access structure $\mathbb{A} = (\mathbf{M}_{(\ell \times n)}, \rho)$, a message $m \in \{0,1\}^*$ and a secret key SK_S parsed as $(D, L, \{D_x\}_{x \in S})$:

1. Compute an index set $I = \{i : \rho(i) \in S\}$ and $\{\alpha_i\}_{i \in I}$ such that $\sum_{i \in I} \alpha_i \cdot \mathbf{M}_i = (1, 0, \ldots, 0)$. Note that such I always exists if $\mathbb{A}$ accepts S.
2. Pick random $\{\beta_i\}_{i=1,\ldots,\ell}$ satisfying $\sum_{i=1}^{\ell} \beta_i \cdot \mathbf{M}_i = (0, 0, \ldots, 0)$.
3. Pick $t' \in \mathbb{Z}_p$ randomly to re-randomize the secret key in the form of

$$D' := g^{\alpha + a(t+t')},\ L' := g^{(t+t')},\ \forall x \in S\ D'_x := h_x^{(t+t')}.$$

4. Pick $z, r \in \mathbb{Z}_p$ randomly and compute $\omega := \mathsf{H}_1(\mathbb{A}, m)$ and the signature as

$$\sigma := \left(\begin{array}{ll} \sigma_0 := D' \cdot (\mathsf{PH}_1(\omega))^r, & \sigma'_0 := g^r, \\ \left\{ \sigma_{0,i} := (L')^{\alpha_i} (g^z)^{\beta_i}, \right. & \left. \sigma_{1,i} := (D'_{\rho(i)})^{\alpha_i} (h_{\rho(i)}^z)^{\beta_i} \right\}_{i=1,\ldots,\ell} \end{array} \right).$$

   where $\alpha_i = 0$ for $\rho(i) \notin I$.
5. Output the signature $\sigma := (\sigma_0, \sigma'_0, \{(\sigma_{0,i},\ \sigma_{1,i})\}_{i=1,\ldots,\ell}, \mathbb{A})$.

– **Verify**(PP, σ, m): On input the public parameters PP, a signature σ parsed as $(\sigma_0, \sigma'_0, \{(\sigma_{0,i},\ \sigma_{1,i})\}_{i=1,\ldots,\ell}, \mathbb{A})$, and a message m:
1. Pick a random vector $\mathbf{v} := (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$ and compute the shares $\lambda_i := \langle \mathbf{v}, \mathbf{M}_i \rangle$ for $i = 1, \ldots, \ell$.
2. Compute $\omega := \mathsf{H}_1(\mathbb{A}, m)$ and

$$T := \frac{e(g^s, \sigma_0)}{e\big(\mathsf{PH}_1(\omega)^s, \sigma'_0\big) \cdot \prod_{i=1,\ldots,\ell} e\big(g^{a\lambda_i} h_{\rho(i)}^{-s}, \sigma_{0,i}\big) \cdot e(g^s, \sigma_{1,i})}.$$

3. Accept the signature σ as valid, and thus output 1 if $e(g, g)^{\alpha s} = T$. Otherwise, it output 0.

4.1 Security Analysis

In what follows, we prove that our CCP-ABES scheme is selectively and jointly secure. The proof comprises three stages: first, we prove that the ABS component is selectively EUF-CMA secure; then we show that the CP-ABE component is selectively IND-CCA secure; in the last stage, we show that our CCP-ABES scheme is perfectly private.

Theorem 1. *The ABS component in the scheme above is selectively EUF-CMA secure in the presence of a decryption oracle if* H_1 *is a collision-resistant hash function,* $\mathsf{SE} = (\mathcal{E}, \mathcal{D})$ *is an AE-OT secure symmetric scheme and the w-DHE assumption holds in* $\mathbb{G}$*, where* $w = |\mathsf{U}|$.

Theorem 2. *The ABS component in our CCP-ABES scheme is perfectly private.*

Due to space considerations the proof of the theorems is given in the full version of this paper.

4.2 Efficiency Analysis

We now compare the efficiency of our CCP-ABES scheme against two separate CP-ABE and ABS schemes that are used together to give the same functionalities. We compare our scheme with the second CP-ABE scheme in [26] and the ABS scheme in [16], in terms of the sizes of public parameters, secret keys and ciphertexts. Let $|\mathbb{G}|$ and $|\mathbb{G}_T|$ denote the size of the underlying group $\mathbb{G}$ and $\mathbb{G}_T$. Let $|\mathsf{U}|$ be the number of attributes in the system. The access structure is expressed by $\mathbf{M}$ with ℓ rows and n columns. We also let $|S|$ denote the number of attributes associated with user's secret key and τ denote the size of the ciphertext of an AE-OT secure symmetric encryption. We assume that the size of the ciphertext exclude the size of access structure. The comparison is summarized in Table 1.

We can see that our CCP-ABES scheme is significantly more efficient than the two ABE and ABS schemes of [26] and [16], particularly in the public parameters and the secret key. Moreover, our scheme obtains stronger security (CCA versus CPA) in the CP-ABE component.

Table 1. Efficiency Comparison

Schemes	Params Size	Secret Key Size	Ciphertext Size	Signature Size
[26] + [16]	$(6+3\|\mathsf{U}\|+k)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(2\|S\|+4)\|\mathbb{G}\|$	$(1+\ell)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(2+\ell+n)\|\mathbb{G}\|$
Ours	$(5+\|\mathsf{U}\|+k)\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(\|S\|+2)\|\mathbb{G}\|$	$(2+\ell)\|\mathbb{G}\|+\tau$	$(2+2\ell)\|\mathbb{G}\|$

5 Attribute-Based Signcryption

The goal in an attribute-based signcryption (ABSC) scheme is to achieve the combined functionality of attribute-based encryption and attribute-based signature, and allows users to obtain message confidentiality and origin authentication through one operation in an attribute-based manner. We define an ABSC scheme to consist of four algorithms:

- **Setup**(κ, U). The algorithm takes a security parameter κ and an attribute universe description U as input, and outputs public parameters PP and a master secret key MSK.
- **KeyGen**$(\mathsf{PP}, \mathsf{MSK}, S)$. The algorithm takes as input public parameters PP, a master key MSK and a set of attributes $S \subseteq \mathsf{U}$, and returns a secret key SK_S associated with S.
- **Signcrypt**$(\mathsf{PP}, m, \mathbb{A}_S, \mathbb{A}_E, \mathsf{SK}_{S_s})$. The algorithm takes as input public parameters PP, a message m, an access structure $\mathbb{A}_E$ for encrypting, an access structure $\mathbb{A}_S$ for signing and a signing secret key SK_{S_s}, and returns a signcryptext CT of message m. And we assume that $\mathbb{A}_E$ and $\mathbb{A}_S$ are implicitly included in the signcryptext.
- **Unsigncrypt**$(\mathsf{PP}, \mathsf{CT}, \mathsf{SK}_{S_d})$. The algorithm takes as input public parameters PP, a signcryptext CT and a secret key SK_{S_d}, and outputs either the message m or an error symbol $\perp$.

For correctness, if $S_d \in \mathbb{A}_E \wedge S_s \in \mathbb{A}_S$, we require that

$$\mathbf{Unsigncrypt}(\mathsf{PP}, \mathbf{Signcrypt}(\mathsf{PP}, m, \mathbb{A}_S, \mathbb{A}_E, \mathsf{SK}_{S_s}), \mathsf{SK}_{S_d}) = m$$

where $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathbf{Setup}(\kappa, \mathsf{U})$, $\mathsf{SK}_{S_s} \leftarrow \mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S_s)$ and $\mathsf{SK}_{S_d} \leftarrow \mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S_d)$.

5.1 Combined ABSC, CP-ABE and ABS Scheme

While efficient ABSC schemes using short public parameters and secret keys for both sender and receiver roles are interesting in their own right, we will consider the more extended primitive which additionally allows that users can also use the ABSC secret keys associated with their attributes in an ordinary CP-ABE and ABS scheme. We consider a scheme implementing the functionality of ABSC, CP-ABE and ABS scheme using a single secret key and short public parameters. This type of scheme consists of algorithms (**Setup**, **KeyGen**, **Signcrypt**, **Unsigncrypt**, **Encrypt**, **Decrypt**, **Sign**, **Verify**).

As in the case of a combined public key scheme, a combined ABSC, CP-ABE and ABS scheme which is jointly secure (as defined in the following) can trivially be constructed by concatenating public parameters and secret keys of independent ABSC, CP-ABE and ABS schemes. However, as above, we focus on schemes which are more efficient than this type of trivial construction. When capturing the security of the combined scheme, ABSC component. We must give an adversary access to a signcryption and unsigncryption oracle in both the confidentiality and unforgeability definitions. In addition, we must also give the adversary access to signing and decryption oracles since we are considering a scheme which additionally implements the functionality of CP-ABE and ABS. We will formally define the security of the ABSC component of the type of combined scheme we are considering: EUF-ABSC-CMA security in the presence of additional oracles, IND-ABSC-CCA security in the presence of additional oracles, and perfect privacy in the full version of this paper.

5.2 Construction Based on CCP-ABES Scheme

We will now show how a CCP-ABES scheme can be used to construct a combined ABSC, CP-ABE and ABS scheme. Our construction is based on the "sign then encrypt" construction. Given a CCP-ABES scheme $\mathcal{C} := (\mathcal{C}.\mathbf{Setup}, \mathcal{C}.\mathbf{KeyGen}, \mathcal{C}.\mathbf{Encrypt}, \mathcal{C}.\mathbf{Decrypt}, \mathcal{C}.\mathbf{Sign}, \mathcal{C}.\mathbf{Verify})$, the combined ABSC, CP-ABE and ABS scheme $\mathcal{ABSC}(\mathcal{C})$ is given as following.

- $\mathcal{ABSC}(\mathcal{C}).\mathbf{Setup}(\kappa)$: Returns $\mathcal{C}.\mathbf{Setup}(\kappa)$.
- $\mathcal{ABSC}(\mathcal{C}).\mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S)$: Returns $\mathcal{C}.\mathbf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, S)$.
- $\mathcal{ABSC}(\mathcal{C}).\mathbf{Encrypt}(\mathsf{PP}, m, \mathbb{A})$: Returns $\mathcal{C}.\mathbf{Encrypt}(\mathsf{PP}, 0||m, \mathbb{A})$.
- $\mathcal{ABSC}(\mathcal{C}).\mathbf{Decrypt}(\mathsf{PP}, \mathsf{CT}, \mathsf{SK}_S)$: $m' \leftarrow \mathcal{C}.\mathbf{Decrypt}(\mathsf{PP}, \mathsf{CT}, \mathsf{SK}_S)$. Parses $m' = 0||m$ and returns m.
- $\mathcal{ABSC}(\mathcal{C}).\mathbf{Sign}(\mathsf{PP}, \mathsf{SK}_S, \mathbb{A}, m)$: Returns $\mathcal{C}.\mathbf{Sign}(\mathsf{PP}, \mathsf{SK}_S, \mathbb{A}, 0||m)$.

- $\mathcal{ABSC}(\mathcal{C})$.**Verify**$(\mathsf{PP}, \sigma, m)$: Returns $\mathcal{C}$.**Verify**$(\mathsf{PP}, \sigma, 0||m)$.
- $\mathcal{ABSC}(\mathcal{C})$.**Signcrypt**$(\mathsf{PP}, m, \mathbb{A}_S, \mathbb{A}_E, \mathsf{SK}_{S_s})$: $\sigma \leftarrow \mathcal{C}$.**Sign**$(\mathsf{PP}, \mathsf{SK}_{S_s}, \mathbb{A}_S, 1||\mathbb{A}_E||\mathbb{A}_S||m)$ $\mathsf{CT} \leftarrow \mathcal{C}$.**Encrypt**$(\mathsf{PP}, 1||\sigma||m, \mathbb{A}_E)$, and returns CT.
- $\mathcal{ABSC}(\mathcal{C})$.**Unsigncrypt**$(\mathsf{PP}, \mathsf{CT}, \mathsf{SK}_{S_d})$: $m' \leftarrow \mathcal{C}$.**Decrypt**$(\mathsf{PP}, \mathsf{CT}, \mathsf{SK}_{S_d})$. If $m' = \perp$, returns $\perp$. Otherwise, parses $m' = 1||\sigma||m$. If $\mathcal{C}$.**Verify**$(\mathsf{PP}, \sigma, 1||\mathbb{A}_E||\mathbb{A}_S||m) \rightarrow 1$, returns m. Otherwise, returns $\perp$.

Remark. In this general construction, we require that the message space in the CP-ABE component of $\mathcal{C}$ should be large enough that can contain an attribute-based signature. Indeed, our construction in last section satisfies this property.

Theorem 3. *Let $\mathcal{C}$ be a CCP-ABES scheme, the ABS component of which is (selectively) EUF-CMA secure in the presence of a decryption oracle. Then the ABSC component of combined ABSC, CP-ABE and ABS scheme $\mathcal{ABSC}(\mathcal{C})$ is (selectively) EUF-ABSC-CMA secure in the presence of additional oracles.*

Theorem 4. *Let $\mathcal{C}$ be a CCP-ABES scheme, the ABS component of which is (selectively) IND-CCA secure in the presence of a signing oracle. Then the ABSC component of combined ABSC, CP-ABE and ABS scheme $\mathcal{ABSC}(\mathcal{C})$ is (selectively) IND-ABSC-CCA secure in the presence of additional oracles.*

Theorem 5. *Let $\mathcal{C}$ be a CCP-ABES scheme, the ABS component of which is perfectly private. Then the ABSC component of combined ABSC, CP-ABE and ABS scheme $\mathcal{ABSC}(\mathcal{C})$ is perfectly private.*

Due to the space limit, the proof of Theorem 4. 5. 6. will be appear in the complete version of this paper.

Acknowledgment. The work is supported by the National Natural Science Foundation of China under Grant No.61170278,91118006, and the National Basic Research Program (973) of China under Grant No. 2012CB315804.

References

1. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 90–108. Springer, Heidelberg (2011)
2. Brickell, E.F.: Some ideal secret-sharing schemes. Journal of Combinatorial Mathematics and Combinatorial Computing 9, 105–113 (1989)
3. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
4. 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)

5. Escala, A., Herranz, J., Morillo, P.: Revocable Attribute-Based Signatures with Adaptive Security in the Standard Model. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 224–241. Springer, Heidelberg (2011)
6. Gagné, M., Narayan, S., Safavi-Naini, R.: Threshold Attribute-Based Signcryption. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 154–171. Springer, Heidelberg (2010)
7. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: CCS 2006, pp. 89–98. ACM Press (2006)
8. Gorantla, M.C., Boyd, C., González Nieto, J.M.: Attribute-Based Authenticated Key Exchange. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 300–317. Springer, Heidelberg (2010)
9. Haber, S., Pinkas, B.: Securely combining public-key cryptosystems. In: CCS 2001, pp. 215–224 (2001)
10. Herranz, J., Laguillaumie, F., Libert, B., Ràfols, C.: Short Attribute-Based Signatures for Threshold Predicates. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 51–67. Springer, Heidelberg (2012)
11. Hess, F.: Efficient Identity Based Signature Schemes Based on Pairings. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 310–324. Springer, Heidelberg (2003)
12. Hofheinz, D., Kiltz, E.: Programmable Hash Functions and Their Applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 21–38. Springer, Heidelberg (2008)
13. Kiltz, E., Vahlis, Y.: CCA2 Secure IBE: Standard Model Efficiency through Authenticated Symmetric Encryption. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 221–238. Springer, Heidelberg (2008)
14. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
15. Li, J., Au, M.H., Susilo, W., Xie, D., Ren, K.: Attribute-based signature and its applications. In: Asiaccs 2010, pp. 60–69. ACM Press (2010)
16. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-Based Signatures. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 376–392. Springer, Heidelberg (2011)
17. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
18. Okamoto, T., Takashima, K.: Efficient Attribute-Based Signatures for Non-monotone Predicates in the Standard Model. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 35–52. Springer, Heidelberg (2011)
19. Paterson, K.G., Schuldt, J.C.N., Stam, M., Thomson, S.: On the Joint Security of Encryption and Signature, Revisited. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 161–178. Springer, Heidelberg (2011)
20. Pirretti, M., Traynor, P., McDaniel, P., Waters, B.: Secure attribute-based systems. In: CCS 2006, pp. 99–112 (2006)
21. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)

22. Shahandashti, S.F., Safavi-Naini, R.: Threshold Attribute-Based Signatures and Their Application to Anonymous Credential Systems. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 198–216. Springer, Heidelberg (2009)
23. Vasco, M.I.G., Hess, F., Steinwandt, R.: Combined (identity-based) public key schemes. Cryptology ePrint Archive, Report 2008/466 (2008), http://eprint.iacr.org/
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)
25. 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)
26. Waters, B.: Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011)
27. Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.: Generic Constructions for Chosen-Ciphertext Secure Attribute Based Encryption. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 71–89. Springer, Heidelberg (2011)
28. Yoneyama, K.: Strongly Secure Two-Pass Attribute-Based Authenticated Key Exchange. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 147–166. Springer, Heidelberg (2010)

Several Weak Bit-Commitments Using Seal-Once Tamper-Evident Devices*

Ioana Boureanu and Serge Vaudenay

Ecole Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, Switzerland
{ioana.boureanu,serge.vaudenay}@epfl.ch

Abstract. Following both theoretical and practical arguments, we construct UC-secure bit-commitment protocols that place their strength on the sender's side and are built using tamper-evident devices, e.g., a type of distinguishable, sealed envelopes. We show that by using a second formalisation of tamper-evident distinguishable envelopes we can attain better security guarantees, i.e., EUC-security. We show the relations between several flavours of weak bit-commitments, bit-commitments and distinguishable tamper-evident envelopes. We focus, at all points, on the lightweight nature of the underlying mechanisms and on the end-to-end human verifiability.

1 Introduction

Most of the recent approaches to primitive-construction employ the universal composability (UC) framework [6] in order to specify and prove the correctness/security of their cryptographic designs. The UC framework is a formalism that allows for cryptographic protocols to be computationally analysed in a single session, yet the security guarantees thereby obtained are preserved when multiple sessions are composed concurrently, in parallel and/or sequentially. In [6], Canetti shows that any polynomial-time multi-party functionality is feasible in the UC framework if the majority of participants are honest. Otherwise, feasibility is usually attained if the models are augmented with "setup-assumptions", obtaining the so-called "UC hybrid models" (i.e., extra ideal functionalities are made available to the parties).

UC-formalisations of tamper-evident/tamper-resistant hardware devices have been used as setups to UC-realize different cryptographic primitives, from bit-commitment to polling schemes [13,15,12,16,17,19,18]; the tamper-evidence of a device implies that, if tampered with, the device will signal the inflicted abnormalities, whereas tamper-resistance denotes the impossibility of tampering with the device. Tamper-evidence-based UC-secure protocols [16,17,19] also bear lightweight, humanly constructible/verifiable cryptographic mechanisms. To realize UC-secure weak bit-commitment (WBC) protocols, a type of distinguishable tamper-evident envelopes were shown sufficient and necessary (in the sense

* Full version of this paper: [3].

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 70–87, 2012.

that simpler functionalities of distinguishable tamper-evident containers are not sufficient to realize bit-commitments) [18]. The protocols thereby constructed placed their "strength" on the receiver's side, i.e., it is the receiver who creates the tamper-evident devices or prepares them. In [18], Moran and Naor raise the question of finding such lightweight, UC-secure (weak) bit-commitment protocols that in turn place their strength on the sender's side, i.e, *sender-strong* protocols. Along similar lines, Brassard, Chaum and Crépeau in foundation papers have long made the question: "Is it preferable to trust Vic or Peggy? We do not know, but it sure is nice to have the choice. [4]".

Contributions. The contributions of this paper are as follows:

- We create weak bit-commitments that place the (adversarial) strength on the committer side, i.e., sender-strong WBC, and that are UC-secure. To achieve this, we require a new formalisation of distinguishable envelopes and use it as a UC setup functionality (see the motivations below).
- We describe a hierarchy of ideal functionalities for sender-strong weak bit-commitments and UC-realize them. In this, we relate better with the existing literature in the field (see Section 2.2 for details).
- We relate our first functionality of distinguishable envelopes ($\mathcal{F}^{DE}_{\texttt{OneSeal}}$), the standard UC-functionality of bit-commitment ($\mathcal{F}^{BC}$) and those of WBC (already existing and newly introduced herein), showing most implication-relations between them.
- We introduce a second distinguishable envelope functionality ($\mathcal{F}^{\texttt{purpoted}DE}_{\texttt{OneSeal}}$), which allows for the corresponding DE-based WBC protocols herein and the ones in [18] to be enjoyed a stronger security notion: be not only UC-secure, but also EUC-secure.

Motivation for Our Formalisation of Tamper-Evident Envelopes. As Moran et al. state in [18], there are many ways to formalise tamper-evident containers, reflecting the different requirements of the possible physical implementations of such devices. The *sole* motivation given in [18] for allowing creator-forgeability is the desiderata of creating more complex, somewhat stronger protocols. But, when it comes to placing this sort of asymmetric strength on the sender's side, it only makes sense to construct commitment protocols that are, in the standard sense, *computationally hiding* and somewhat binding, i.e., the receiver is powerless and the sender can possibly equivocate his commitments. (By contrast, in [18] are both *partially* hiding and *partially* binding and are then amplified.) In this context, we conjecture that it is not possible to be based only on tamper-evident envelopes à la Moran et al. [18] and construct *hiding* sender-strong bit-commitment protocols, which would further be UC-secure in the same time. To overcome this shortcoming, we have herein slightly modified the original, tamper-evident envelope functionality from [18], preventing the creator from resealing envelopes. Hence, we model *seal-once* distinguishable tamper-evident envelopes (or, envelope allowing *one-seal* only). By contrast, the functionality in [18] formalises a *multi-seal* distinguishable tamper-evident envelope.

Furthermore, the previous protocols designed using tamper-evident envelopes à la Moran et al. [18] were only UC-secure and not EUC-secure. We noted that if we relaxed the forging abilities of the envelope-creator in the aforementioned way and we furthermore allow for purported destinator for envelopes the corresponding DE-based protocols obtained both here and in [18] attain EUC-security and not only UC-security.

Our Weak Bit-Commitments from a theoretical viewpoint. Alongside the UC-framework, sender-strong weak bit-commitments are also interesting by traditional theoretical lines, where they are easier to construct (see Section 3.4). In [4], outside of the UC-framework, Brassard et al. proved that the existence of "chameleon" bit-commitments[1] implies the existence of zero-knowledge (ZK) proofs of knowledge which were MA-protocols (i.e., where the verifier sends independent bits). Moreover, in [2] Beaver proved that in order for the aforementioned ZK proofs of knowledge (PoK) to be provable secure against adaptive adversaries, the chameleon bit-commitments (BC) are not enough, but content-equivocable bit commitments are needed (i.e., the equivocation is possible only if a record of the traffic between the sender and the receiver is available to the sender and not other types of witnesses, like parts of messages). One of our weak BC functionalities, $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$, models this last type of important weak bit-commitments.

Our Weak Sender-Strong *Bit-Commitments from a practical viewpoint.* A real-life situation were the committer/sender should be given the chance to "change his mind" is the case in negotiation-based protocols where the receiver is known or thought to be corrupt (e.g, hostage-release cases, reputations [1], anonymous special auctioning [21], etc.).

To sum up, we are motivated to present certain means of attaining different UC and EUC-secure, bit-commitment protocols placing their strength on the sender's side, i.e., *sender-strong (SS).*

Related Work. A series of works on designing UC-secure protocols using tamper-resistant building blocks have recently emerged [13,8,15,12,19]. For example, the formalism by Katz, in [13], opens for the creation and exchange of tamper-proof hardware tokens used in a commitment protocol, which is UC-secure if the tokens are stateful and the DDH assumption holds. In [8], the two-party computation can equally be UC-realized, but the model is relaxed: the tokens are stateless and the assumption is switched to the existence of oblivious transfer protocols in the UC plain model. Similar results are obtained using tamper-resistant devices as building blocks in a model called the trusted agent model [15]. Like in [8] and unlike in [13], Mateus and Vaudenay [15] permit a freer flow of devices from their creator to their users and backwards. Similar protocols are constructed by Moran et al., in [19], using tamper-resistant hardware tokens that can be passed

[1] These are commitments where the sender could cheat at the decommitment phase if given extra information.

in one direction only. We note that the distinction of having UC-commitments which place the strength on the sender or, on the contrary, place their strength on the receiver has also been underlined [19] within this context of using tamper-resistant hardware as UC-setup.

Simpler cryptographic protocols UC-constructed using not tamper-resistant devices, but tamper-evident devices in form of sealed envelopes and sealed locks have been studied in [18,16,17]. All the protocols thereby presented place their strength on the receiver's side.

2 Setup and Target UC Functionalities

2.1 UC-Setup Functionalities Modelling Tamper-Evident Envelopes

The $\mathcal{F}_{\mathtt{OneSeal}}^{DE}$ Functionality. In general, a functionality for tamper-evidence stores a table of envelopes, indexed by their unique *id*. More precisely, an entry in this table is of the form (*id*, *value*, *holder*, *state*). The values in one entry indexed by *id* are respectively denoted $value_{id}$, $holder_{id}$ and $state_{id}$.

In particular, the functionality $\mathcal{F}_{\mathtt{OneSeal}}^{DE}$ models a tamper-evident "envelope", distinguishable by some obvious mark (e.g., barcode, serial number, colour, etc.). Protocol parties can simply open such containers, but any such opening will be obvious to other parties who receive the "torn" envelope. The $\mathcal{F}_{\mathtt{OneSeal}}^{DE}$ ideal functionality, running in the presence of parties $P_1, \ldots, P_n$ and an ideal adversary $\mathcal{I}$ is described in the following.

Seal(*id*, *value*). Let this command be received from party P_i. It creates and seals an envelope. If this is the first **Seal** message with id *id*, the functionality stores the tuple (*id*, *value*, P_i, **sealed**) in the table. If this is not the first command of type **Seal** for envelope *id*, then the functionality halts.

Send(*id*, P_j). Let this command be received from party P_i. This command encodes the sending of an envelope held by P_i to a party P_j. Upon receiving this command from party P_i, the functionality verifies that there is an entry in its table which is indexed by *id* and has $holder_{id} = P_i$. If so, it outputs (**Receipt**, *id*, P_i, P_j) to P_j and $\mathcal{I}$ and replaces the entry in the table with (*id*, $value_{id}$, P_j, $state_{id}$).

Open *id*. Let this command be received from party P_i. This command encodes an envelope being opened by the party that currently holds it. Upon receiving this command, the functionality verifies that an entry for container *id* appears in the table and that $holder_{id} = P_i$. If so, it sends (**Opened**, *id*, $value_{id}$) to P_i and $\mathcal{I}$. It also replaces the entry in the table with (*id*, $value_{id}$, $holder_{id}$, **broken**).

Verify *id*. Let this command be received from party P_i. This command denotes P_i's verification of whether or not the seal on an envelope has been broken. The functionality verifies that an entry indexed by *id* appears in the table and that $holder_{id} = P_i$. It sends (**Verified**, *id*, $state_{id}$) to P_i and to $\mathcal{I}$.

One of the differences from the corresponding functionality presented in [18] is that the one introduced above does not output tuples containing the creator's

identity. This would have been of no interest for the protocols constructed in the following and would hinder EUC-security proofs given herein. However, a more important difference is that the creator of an envelope cannot re-seal it, i.e., he cannot forge the value stored initially inside the envelope. Hence, we use the syntagm "`OneSeal`" to refer to the functionality herein and, sometimes, we use the expression "`MultiSeal`" to designate the tamper-evident envelopes in [18]. This modification is driven by the fact that we could not yet prove or disprove the existence of a sender-strong, somewhat binding and *not partially hiding, but computationally hiding* bit commitment *that is also simulatable within UC*, using only creator-forgeable/multi-seal tamper-evident envelopes.

It is relatively easy to see that *regular* bit-commitments can be immediately constructed using one distinguishable tamper-evident envelope, see Section 4. The relation with the regular commitment functionality is however not symmetric, as will detail (i.e., if $\mathcal{F}^{DE}_{\texttt{OneSeal}}$ implies BC, it is not necessarily the case that $\mathcal{F}^{BC}_{\texttt{OneSeal}}$ implies DE).

The tamper-evident envelope functionality in [18] is denoted as $\mathcal{F}^{DE}_{\texttt{MultiSeal}}$.

2.2 Target UC Functionalities of Bit-Commitment

We now describe our target functionalities $\mathcal{F}^{q-\mathrm{WBC}}_{\star}$ that model different weak bit-commitment (WBC) protocols, where

$$\star \in \{\texttt{EscapeThenMayCheat}, \texttt{LearnAtCommitment}, \texttt{LearnAtOpening}\}.$$

In this fashion, we can relate the WBCs UC-realized herein both with traditional weak bit-commitments [2] of theoretical importance (e.g., see our $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$), and with weak bit-commitments UC-created in [18] with distinguishable envelopes (see our $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$). The differences between these functionalities lie mainly in learning that equivocation is possible (yet not obligatory) at the commitment phase ($\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtCommitment}}$) or the opening phase ($\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$) vs. cheating only when the committer has not yet been caught abusing the protocol ($\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$).

The $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$ *functionality idealising q-weak bit-commitment.* Let $q \in (0,1)$. The functionality maintains a variable *bit*, where *bit* ranges over $\{0,1,\square\}$.

Commit b. When the **Commit** b command ($b \in \{0,1\}$) is sent to the functionality by a sender S, the value b is recorded in the variable *bit*. The functionality of $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$ outputs **Committed** to the receiver R and to the ideal adversary $\mathcal{I}$[2]. Further commands of this type or of type **EquivocatoryCommit** below are ignored by the functionality.

EquivocatoryCommit. When the **EquivocatoryCommit** command is sent to the functionality, the $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$ functionality replies to the sender and

[2] Throughout, the fact that the output is sent to the ideal adversary as well is inherent to the UC framework, i.e., see the UC-notion of "delayed output".

the ideal adversary with a $\perp$ message, with probability $1 - q$. With probability q, the functionality sets the variable *bit* to the value $\square$, outputs **Committed** to the sender, the receiver and to the ideal adversary. Further commands of this type or of type **Commit** above are ignored by the functionality.

AbortCommit. When the **AbortCommit** command is sent to the functionality, the $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$ functionality replies to the sender, to the receiver, and to the ideal adversary with a $\perp$ message (denoting an abnormal end of the execution). Further commands are ignored.

Open. Upon receiving the command **Open** from the sender, the functionality verifies that the sender has already sent the **Commit** b command. Then, the $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$ functionality outputs (**Opened**, *bit*) to the receiver and to the ideal adversary. Further commands are ignored by the functionality.

EquivocatoryOpen c. Upon receiving this command from the sender, with $c \in \{0, 1\}$, the functionality verifies that $bit = \square$. Then, the functionality $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{EscapeThenMayCheat}}$ outputs (**Opened**, c) to the receiver and to the ideal adversary. Further commands are ignored by the functionality.

In this functionality, the binding property of commitments can be defied. It corresponds to the weak bit-commitment functionality used by Moran and Naor [18], but it applies to the sender-strong case. In that sense, a dishonest player decides to try and open his commitment to any value even from the very beginning of the protocol and he can be successful in doing so with a probability of $q \in (0, 1)$, once he has not been caught red-handed.

Note that the WBC functionality presented above and the ones to be presented further model single bit commitments. Yet, they can easily be extended to respective functionalities for multiple commitments: i.e., each **Commit** b command sent by a sender S aimed at a receiver R would become **Commit**(id, b, R) and each corresponding functionality would, for each commitment, store a tuple $(id, sender, receiver, value)$ doing the respective checks.

The $\mathcal{F}^{q-\mathrm{WBC}}_{\textit{LearnAtCommitment}}$ *functionality idealising q-weak bit-commitment.* Let $q \in (0, 1)$. The functionality maintains a tuple (*bit*, *equiv*), where *bit* ranges over $\{0, 1\}$ and *equiv* ranges over {"Yes", "No"}.

Commit b. When the **Commit** b command ($b \in \{0, 1\}$) is sent to the functionality, the value b is recorded in the variable *bit*. With probability q the value "Yes" is stored in *equiv* or, with probability $1 - q$ the value "No" is stored in *equiv*. The $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtCommitment}}$ functionality outputs **Committed** to the receiver and to the ideal adversary. The $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtCommitment}}$ functionality outputs the updated value of *equiv* to the sender and to the ideal adversary. Further commands of this type are ignored by the functionality.

Open. Upon receiving this command, the functionality verifies that the sender has already sent the **Commit** b command. Then, the $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtCommitment}}$ functionality outputs (**Opened**, *bit*) to the receiver and to the ideal adversary. Further commands are ignored by the functionality.

EquivocatoryOpen. Upon receiving this command, the functionality verifies that the sender has already sent the **Commit** b command. Then, the functionality checks the value of *equiv*. If the value is "Yes", then $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtCommitment}}$ outputs (**Opened**, $\overline{bit}$) to the receiver and to the ideal adversary. If the value is "No", then $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtCommitment}}$ halts. Further commands are ignored by the functionality.

The $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtCommitment}}$ functionality mirrors a protocol which allows the sender to cheat by breaking the binding property of the protocol. Note that this cheating possibility is "decided" at the commitment phase, i.e., it is at some point during the commitment phase that the potential cheater *learns* about his opportunity. Also, note that while the cheating is allowed, it does not necessarily need to happen (i.e., there are two distinct opening commands).

Next, we give a similar functionality where in turn the possibility of equivocation becomes clear only at the opening phase.

The $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$ *functionality idealising q-weak bit-commitment.* Let $q \in (0,1)$. The functionality maintains a variable *bit*, ranging over $\{0,1\}$.

Commit. When the **Commit** b command ($b \in \{0,1\}$) is sent to the functionality, the value b is recorded in the variable *bit*. The $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$ functionality outputs **Committed** to the receiver and to the ideal adversary. Further commands of this type are ignored by the functionality.

Open. Upon receiving this command, the functionality verifies that the sender has already sent the **Commit** b command. Then, the $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$ functionality outputs (**Opened**, *bit*) to the receiver and to the ideal adversary. Further commands are ignored by the functionality.

EquivocatoryOpen. Upon receiving this command, the functionality verifies that the sender has already sent the **Commit** b command. With probability q, the $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$ outputs (**Opened**, $\overline{bit}$) to the receiver and to the ideal adversary. With probability $1-q$, the $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$ sends $\perp$ to the sender S and the ideal adversary $\mathcal{I}$. Further commands of this type are ignored by the functionality (but commands of type **Open** are still allowed).

The $\mathcal{F}^{q-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$ functionality mirrors a protocol which allows the sender to cheat by breaking the binding property of the protocol, knowingly at some point during the opening phase, i.e., i.e., it is at some point during the opening phase that the potential cheater *learns* about his opportunity, similarly to traditional lines in [2]. As aforementioned, note that while the cheating is allowed, it does not necessarily need to happen.

Note that sender-strong weak bit-commitments protocols with distinguishable, tamper-evident envelopes that allow only partial hiding are of course easier to UC-construct than those that require perfect hiding. Amplifications techniques could then be applied. However, we take the view that once the protocols that we seek are sender-strong, UC-realizing a functionality which is only partially hiding is would contradict the aim for the senders' strength (hence, the "*computationally hiding*" UC functionalities that we have given above).

3 UC (Sender-Strong) Bit-Commitments

Driven by the theoretical and practical motivations presented in the introduction, we are now going to present protocols that UC-implement the weak bit-commitment (WBC) functionalities above, use the herein introduced functionalities of distinguishable envelopes as the UC-setup and place their strength on the senders' sides.

We start with the protocol `Pass&MayCheat` which UC-realizes the functionality $\mathcal{F}^{\frac{1}{2}-\text{WBC}}_{\texttt{EscapeThenMayCheat}}$ using the $\mathcal{F}^{DE}_{\texttt{OneSeal}}$. We continue the protocols `CommitEnablesCheat` and `OpenEnablesCheat` which respectively UC-realize the functionalities of $\mathcal{F}^{\frac{2}{3}-\text{WBC}}_{\texttt{LearnAtCommitment}}$ and $\mathcal{F}^{\frac{2}{3}-\text{WBC}}_{\texttt{LearnAtOpening}}$, using the $\mathcal{F}^{DE}_{\texttt{OneSeal}}$. We then present amplification techniques of such weak BC protocols. The techniques maintain the lightweight character of the constructions. We conclude the section by a strengthening of the $\mathcal{F}^{DE}_{\texttt{OneSeal}}$ functionality such that we attain EUC-security [7], i.e., not only UC-security.

3.1 The `Pass&MayCheat` Protocol

The Commitment Phase.

1. A sender S seals four envelopes and creates two pairs out of them such that each pair contains the set $\{x, x\}$ of values, for a random $x \in \{0, 1\}$. Each pair "contains" its own value x. He sends two envelopes, one from each pair, to the receiver R. (E.g., the pairs are $pair_1 = (E_1, E_2)$, $pair_2 = (E_3, E_4)$ and S sends, e.g., E_1, E_3 to R).
2. The receiver R stores the identifiers of the envelopes in a register W. (I.e., it stores $(1, 3)$, given the illustrated execution by S above.). Then, R sends them back without opening them.
3. The sender S verifies that the recently returned envelopes have the seals unbroken. If this is not so, he halts. Otherwise, he sends the two envelopes not sent before. (I.e., If seals are unbroken, then S sends the remaining E_2, E_4.)
4. The receiver verifies that the envelopes received do not have the ids stored already. If they do, he halts. Otherwise, he opens one of these envelopes, sends back the other one without opening it, together with the value of an *id* stored already in W to request back one envelope. The receiver also stores the ids of the envelopes seen this time round. (I.e., R opens, e.g., E_2, sends back E_4 and, e.g., 1, thus requesting back envelope E_1.) *Given the steps of the protocol so far, note that the opened envelope together with the requested one form an initial pair. Also, once the sender has sent this requested envelope, the sender will be left with the other of the initial pairs at his end. These comments equally apply to the equivocatorial commitment phase to follow.*
5. The sender S verifies that the recently returned envelope has the seal unbroken. If this is not so, he halts. Otherwise, he sends the one requested envelope to R. He also sends the value $d = b \oplus x$, where b is the bit he is

committing to and x is the bit hidden inside each envelope in the pair to be found at his side. (I.e., If the seal is unbroken, then S sends the requested E_1, $d = b \oplus x$, where x is in E_3 and/or in E_4).
6. The receiver R opens the last envelope received and checks that the value at its side are equal. If not, he aborts. (I.e., If E_1 and E_2 do not contain the same value, then R aborts).

The EquivocatoryCommitment Phase.

1. A sender S seals four envelopes and creates two pairs out of them such that one pair contains the set $\{x, x\}$ of values, for a random $x \in \{0, 1\}$ and the other pair contains the values $\{0, 1\}$. He sends two envelopes, one from each pair, to the receiver R. (E.g., The pairs are $pair_1 = (E_1, E_2)$, $pair_2 = (E_3, E_4)$ and S sends E_1, E_3 to R).
2. Same as in the commitment phase.
3. Same as in the commitment phase.
4. Same as in the commitment phase.
5. Same as in the commitment phase.
6. Same as in the commitment phase.

Let A denote the pair of envelopes to be found at this stage on the sender side.

The Opening Phase.

1. The sender S sends one envelope E_k in the remaining pair, i.e., $E_k \in A$ or $k \in \{i, j\}$.
2. The receiver R checks that E_k is in the set A (by checking the ids). If so, he opens the envelope E_k to find the value b_k hidden inside and then he sets the commitment-bit b' to $d \oplus b_k$. Otherwise, the receiver halts.

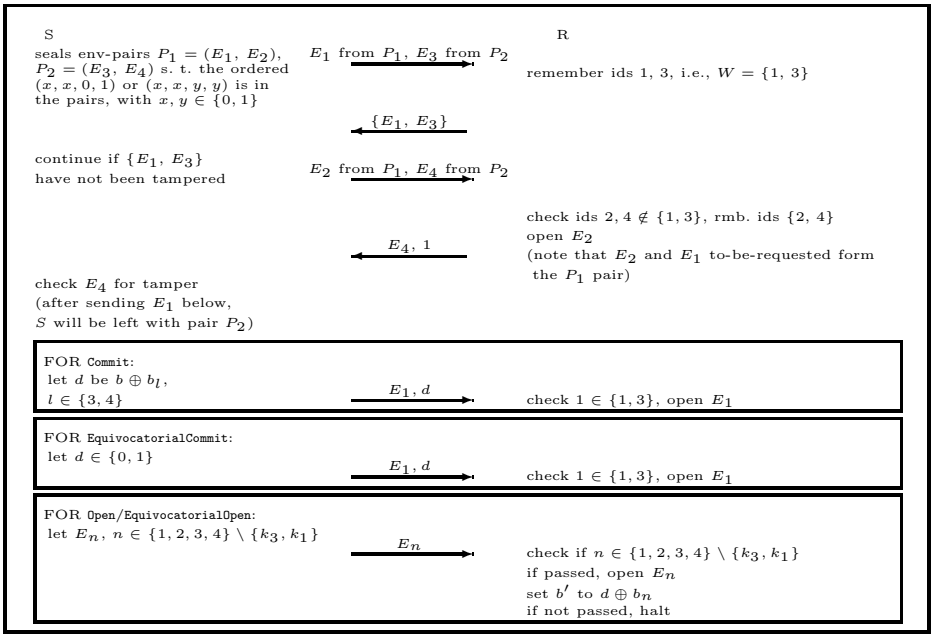
The EquivocatoryOpening Phase.

1. The sender S sends from the remaining pair A the envelope E_k that contains the bit $d \oplus c$, where c is the bit that the sender wants to open to.
2. Same as in the opening phase.

In the following figure, we give an illustration of the protocol above, in a symmetric way (i.e., E_1 and E_2 could be interchanged in their appearances, etc.).

Explanations on the `Pass&MayCheat` *protocol.* Assume first that the sender S creates pairs of envelopes such that each contains the set $\{x, x\}$ of values and that the sender respects the calculation of d for the non-equivocable case. It is clear that at the end of the protocol, the sender has no choice but to open to the correct bit. If, in turn, S does not form d as specified and R does follow the protocol, then S may not be able to open.

Assume now that the sender S creates pairs of envelopes such that one contains the set $\{x, x\}$ of values and the other contains the set $\{0, 1\}$ of values. Depending



on the choice of R to open envelopes, the sender may continue the protocol; clearly this is possible in half of the cases (the possibility that a randomly chosen bit x is equal either to 0 or to 1, depending which was the opening of R). In such cases, S can clearly open the value d to any bit-value, since he is left with x and its negation $\overline{x}$ in the pair A at his end.

Note that the receiver R cannot cheat without being caught: i.e., torn envelopes are obvious to the sender and opening by R of more than two envelopes –to try and break the hiding property– is not possible due to the stage-by-stage unsealing enforced by the protocol.

Also, note that the envelopes used within are seal-once envelopes. Thus, the sender S is not able to change the values x stored inside the envelopes, after step 4 of the commitment phase (say, in order to avoid being caught by R).

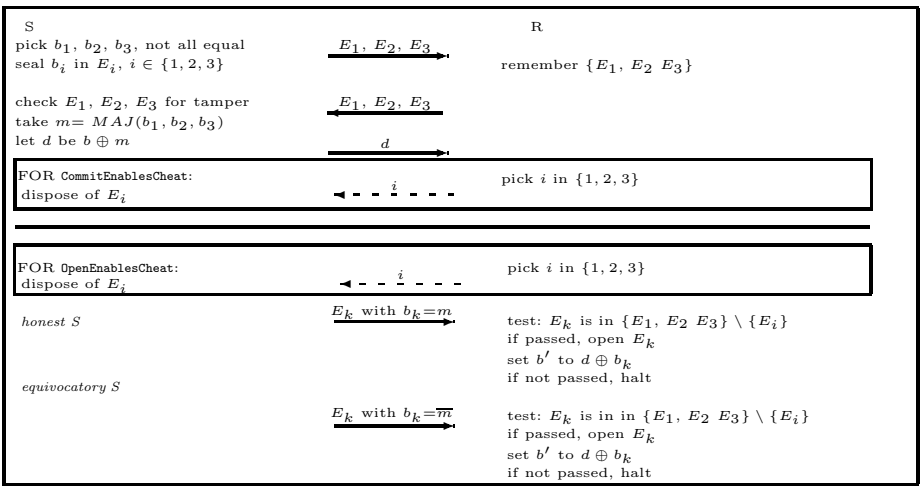
Theorem 1. *In a hybrid UC-model, where the setup is the* $\mathcal{F}_{\mathit{OneSeal}}^{DE}$ *functionality, the* `Pass&MayCheat` *protocol UC-realizes the* $\mathcal{F}_{\mathit{EscapeThenMayCheat}}^{\frac{1}{2}-\mathrm{WBC}}$ *functionality.*

Due to space constraints, the proof of Theorem 1 is given in the the full version of this paper [3].

3.2 The `CommitEnablesCheat` and `OpenEnablesCheat` Protocols

The `CommitEnablesCheat` Protocol

The Commitment Phase. The sender wants to commit to a bit b and proceeds as it follows.



1. The sender S creates 3 sealed envelopes denoted E_1, E_2, E_3 respectively containing the bits denoted b_1, b_2, b_3, such that not all bits are equal. The sender sends the envelopes over to the receiver R.
2. The receiver memorises the set $\{E_1, E_2, E_3\}$ of envelopes and sends them back to the sender
3. The sender verifies that the envelopes sent back are untampered with. Then, he computes m as the majority of the bits sealed inside, i.e., he computes $m = MAJ(b_1, b_2, b_3)$. The sender wants to commit to a bit b. He calculates $d = b \oplus m$. Then, the sender sends d to the receiver.
4. The receiver sends the identifier i of an envelope that the sender should dispose of, i.e., $i \in \{1, 2, 3\}$. Let the set $S = \{E_1, E_2, E_3\} \setminus \{E_i\}$ denote the set of remaining envelopes.
5. The sender disposes of envelope i. (Note that after this the sender can equivocate if the remaining envelopes contain different bits.)

The *equiv* value is $\frac{2}{3}$.

The Opening Phase.

1. The non-equivoquing sender sends an envelope E_k such that $b_k = m$.
   The equivoquing sender sends an envelope E_k such that $b_k = \overline{m}$.
2. The receiver tests that $E_k \in S$ and if so, he sets b', the commitment bit, as follows: $b' = d \oplus b_k$. If the test fails, the receiver halts.

Note that by being asked to discard[3] an envelope at the opening phase instead of in step 4 of the commitment phase, the idea behind protocol `CommitEnablesCheat` can be shaped to obtain a protocol where the equivocation becomes clear only at

[3] A possible way of implementing discarding is sending the emptied envelope back to the receiver.

the opening time. The protocol obtained in this way is hereby denoted `OpenEnablesCheat`. The protocols `CommitEnablesCheat` and `OpenEnablesCheat` are graphically represented in the previous figure.

Note once more that, unlike in the `Pass&MayCheat` protocol, in `CommitEnablesCheat` and `OpenEnablesCheat` protocols, the committer can cheat with some probability (i.e., $\frac{2}{3}$), yet this is not influenced by him being caught cheating, but rather by a mere choice of the receiver.

These requirements sound similar to looking for a means in which Alice would commit to a bit b using a BSC (binary symmetric channel) with noise level q [5]. Nevertheless, the existing solutions [5,10,20] to problems of the latter kind are receiver-strong, not sender-strong. Also, they are not constructed to be UC-secure, but secure by classical lines, which may be weaker. Moreover, those original constructions involve error-correction codes and/or pseudo-random generators being manipulated by the participants. Thus, those primitives are also beyond our cryptographically lightweight scope. Therefore, to obtain senders' strength, UC-security, simplicity and human operability we have proposed protocols `CommitEnablesCheat` and `OpenEnablesCheat` above.

Explanations on the `CommitEnablesCheat` and `OpenEnablesCheat` protocols. We detail on the `CommitEnablesCheat` protocol above, the explanations on `OpenEnablesCheat` being very similar and immediately following. Let us consider the case where the parties follow the protocol. We can see that if S prepares the envelopes correctly (i.e., they contain a permutation of $\{x, x, \overline{x}\}$, $x \in_U \{0, 1\}$) and he adheres to step 2, then at step 3, the value $m = x$. No matter what value b_i has (i.e., x or $\overline{x}$), in the set S of remaining envelopes there is always an envelope E_k with the value x inside that opens the commitment correctly. With probability $\frac{2}{3}$, the set S still contains an envelope with value $\overline{x}$. In this last case, S could open his commitment to the flipped bit (i.e., point 2 in the opening phase). By the above, the protocol is complete. One can see that the case where S does not follows the protocol in terms of envelope sealing does not bring him any benefit. In Theorem 2, we formalise the above explanations, in the context of the UC framework.

Theorem 2. *In a hybrid UC-model, where the setup is the $\mathcal{F}_{OneSeal}^{DE}$ functionality, the `CommitEnablesCheat` and `OpenEnablesCheat` protocols UC-realize the $\mathcal{F}_{LearnAtCommitment}^{\frac{2}{3}-\mathrm{WBC}}$ and the $\mathcal{F}_{LearnAtOpening}^{\frac{2}{3}-\mathrm{WBC}}$ functionalities, respectively.*

Due to space constraints, this proof is given in the full version of this paper [3].

3.3 Amplifying q-WBC Sender-Strong Protocols

Let ✠ ∈ {`Pass&MayCheat`, `CommitEnablesCheat`, `OpenEnablesCheat`}.
Let ⋆ ∈ {`EscapeThenMayCheat`, `LearnAtCommitment`, `LearnAtOpening`}.

By using k instances of a q-weak sender-strong protocol of the ✠-kind of protocols, we can obtain a protocol `Amplified_`✠ protocol that UC-realizes $\mathcal{F}_{\star}^{q^k-WBC}$. Hence, we can attain regular bit-commitments for a conveniently large k. See the formalisations below.

The `Amplified_Pass&MayCheat` Protocol:

(Equivocatory) Commitment Phase. The sender commits, all equivocally or all normally, to a bit b_j in k sequential rounds, each time using the $\mathcal{F}^{q-WBC}_{\texttt{EscapeThenMayCheat}}$ functionality, $j \in \{1, \ldots, k\}$. The j-th such functionality is denoted $\mathcal{F}^{q-WBC}_{\texttt{EscapeThenMayCheat};\, j}$.

Each functionality $\mathcal{F}^{q-WBC}_{\texttt{EscapeThenMayCheat};\, j}$ to which **EquivocatoryCommit** was sent, outputs to its sender `Committed`, with probability q and $\perp$ otherwise. If $\perp$ is sent, then the receiver aborts.

(Equivocatory) Opening Phase. The sender opens (equivocally or not) all commitments using the functionalities of $\mathcal{F}^{q-WBC}_{\texttt{EscapeThenMayCheat};\, j}$. The receiver halts if the openings are not all the same.

Theorem 3. *Let $q \in (0,1)$ and λ be a security parameter. By using $k = \Omega(\lambda)$ instances of an $\mathcal{F}^{q-WBC}_{\star}$ functionality, we can construct a protocol `Amplified_`$\maltese$ that UC-realizes the $\mathcal{F}^{BC}$ functionality, where*
$\star \in \{$`EscapeThenMayCheat`, `LearnAtCommitment`, `LearnAtOpening`$\}$ and
$\maltese \in \{$`Pass&MayCheat`, `CommitEnablesCheat`, `OpenEnablesCheat`$\}$.

In particular, the protocol `Amplified_Pass&MayCheat` UC-realizes the functionality of $\mathcal{F}^{q^k-WBC}_{\texttt{EscapeThenMayCheat}}$.

For the regular BC functionality, $\mathcal{F}^{BC}$, see the full version of this paper [3]. The `Amplified_Pass&MayCheat` BC protocol is trivially following out of `Amplified_Pass&MayCheat`, i.e., where equivocation is not possible. By letting $k = \frac{\log \varepsilon}{\log q}$ in Theorem 3, we make `Amplified_Pass&MayCheat` a ε-WBC, with ε arbitrarily close to 0. However, for `Amplified_Pass&MayCheat` to UC-realize $\mathcal{F}^{BC}$, we need a k to be of linear-size in the security parameter λ. Proofs that weak bit-commitment protocols in the above sense can be amplified to regular bit-commitments exist already, e.g., [20]. The proofs therein follow long-established lines, i.e., not the UC framework [4]. Also, they often refer to receiver-strong protocols and generally use more convoluted primitives, e.g., pseudo-random generators, error-correcting codes, outside our lightweight interests. Our proof is done in the UC framework and, as we can see, the protocol respects the sender-strong aspects sought-after herein. Due to space constraints, the actual proof of Theorem 3 is given in the full version of this paper [3].

3.4 (Stronger) Universally Composable Security

A UC-oriented note is that something as little as the order of the messages in the commitment-phase of the weak protocol `CommitEnablesCheat` above and/or the amount of randomness given to the sender does impact the UC-simulatability. A protocol only different from `CommitEnablesCheat` in that it inverts the order of events 3 and 4 in the commitment phase does not UC-realize the $\mathcal{F}^{\frac{2}{3}-\text{WBC}}_{\texttt{LearnAtCommitment}}$

[4] Similar proofs of amplifications may exist in the UC framework, however they would not be with respect to the $\mathcal{F}^{q-WBC}_{i}$ functionalities as introduced in Section 2.

functionality, while it is perfectly hiding and binding with probability $\frac{2}{3}$ in the classical sense. Thus, it seems to be easier to construct q-weak bit-commitments using just a formalisation of distinguishable envelopes, but when sender-strength and UC-security are both sought after subtle difficulties arise ($q \in (0, 1)$).

Another important UC note is that the protocols above (and in fact all weak bit-commitment protocols constructed previously for the receiver-strong case in Moran and Noar's work [18]) are not secure in stronger versions of the UC framework, e.g., GUC (Generalised UC) or EUC (externalised UC) [7]. For a wrap-up on GUC (Generalised UC) or EUC (externalised UC), see the full version of this paper [3]. To support this claim, it is enough to show that the protocols are not secure in the EUC framework. So, in an EUC model with the $\mathcal{F}_{\texttt{OneSeal}}^{DE}$-setup consider an environment that prepares the envelopes and feeds them to the adversary. It is clear that the ideal adversary cannot "extract" the bit b to commit to and thus he cannot indistinguishably simulate the commitment phase.

Lemma 4. *In a hybrid EUC-model, where the setup is the $\mathcal{F}_{\texttt{OneSeal}}^{DE}$ functionality, the `CommitEnablesCheat` protocol does not EUC-realizes the $\mathcal{F}_{\texttt{LearnAtCommitment}}^{\frac{2}{3}-\text{WBC}}$ functionality.*

Due to space constraints, the proof of this lemma is given in the full version of this paper [3].

As aforementioned, along very similar lines the receiver-strong protocols in previous works [18] are not EUC-secure either. We modify the $\mathcal{F}_{\texttt{OneSeal}}^{DE}$ functionality slightly such that when used as a setup, we attain EUC-security of the protocols herein and those WBC protocols in Moran and Noar's work [18].

$\mathcal{F}_{\texttt{OneSeal}}^{purpotedDE}$: A Stronger Functionality for Tamper-Evident Distinguishable Sealed Envelopes. This functionality stores tuples of the form (*id*, *value*, *holder*, *state*). The values in one entry indexed with *id*, like before.

SealSend(*id*, *value*, P_j). Let this command be received from an envelope-creator party P_i. It seals an envelope and sends its id to the future holder P_j. If this is the first **Seal** message with id *id*, the functionality stores the tuple (*id*, *value*, P_j, **sealed**) in the table. The functionality sends (*id*, P_i) to P_j and to $\mathcal{I}$. (Optionally, it can send (*id*, *sealed*) to P_i and to $\mathcal{I}$). If this is not the first command of type **Seal** for envelope *id*, then the functionality halts.

Send(*id*, P_j). Let this command be received from a holder-party P_i. This command encodes the sending of an envelope held by P_i to a party P_j. Upon receiving this command from party P_i, the functionality verifies that there is an entry in its table which is indexed by *id* and has $holder_{id} = P_i$. If so, it outputs (**Receipt**, *id*, P_i, P_j) to P_j and $\mathcal{I}$ and replaces the entry in the table with (*id*, $value_{id}$, P_j, $state_{id}$).

Open *id*. Let this command be received from a holder-party P_i. This command encodes an envelop being opened by the party that currently holds it. Upon receiving this command, the functionality verifies that an entry for container *id*

appears in the table and that $holder_{id} = P_i$. If so, it sends (**Opened**, id, $value_{id}$) to P_i and $\mathcal{I}$ and replaces the entry with (id, $value_{id}$, $holder_{id}$, **broken**).

Verify id. Let this command be received from a holder-party P_i. This command denotes P_i's verification of whether or not the seal on an envelope has been broken. The functionality verifies that an entry indexed by id appears in the table and that $holder_{id} = P_i$. It sends (**Verified**, id, $state_{id}$) to P_i and to $\mathcal{I}$.

The main difference between $\mathcal{F}_{\texttt{OneSeal}}^{\texttt{purpoted}DE}$ and the original $\mathcal{F}_{\texttt{OneSeal}}^{DE}$ functionality is that an envelope is created for a specifically intended holder and this holder is consequently notified with a message of the form (id, $creator$). Note that this enhancement is realistic (i.e., if to be used in a protocol, the delivery address of the receiver is to be specified by a manufacturing body). However, note that the functionality does not store or reveal publicly the creators of the envelopes (i.e., that would be a stronger enhancement, akin to signing the tamper-evident devices). With this modification, the holder-to-be knows that a specific envelope has been freshly produced by a specific creator. Intuitively, this prevents the weakness in the proof of Lemma 4 from happening, i.e., R cannot accept envelopes that are not (newly) meant for him. Also, the creator is authenticated by the functionality, in the sense that he cannot use envelopes made by others. In a larger sense, this can prevent relay attacks. More formally, the following holds.

Theorem 5. *In a hybrid EUC-model, where the setup is the* $\mathcal{F}_{\texttt{OneSeal}}^{\texttt{purpoted}DE}$ *functionality, the* `CommitEnablesCheat` *protocol EUC-realizes the* $\mathcal{F}_{\texttt{LearnAtCommitment}}^{\frac{2}{3}-\text{WBC}}$ *functionality.*

The proof of the above theorem follows from the proofs of Theorem 2 and that of Lemma 4, combined with the fact that $\mathcal{F}_{\texttt{OneSeal}}^{\texttt{purpoted}DE}$-envelopes have a specified entity as their destination and this entity knows this fact upon the creation of the envelopes. We conjecture that Theorem 5 holds even in the case of adaptive adversaries.

4 Relations between (Weak) Bit-Commitments and Distinguishable Envelopes in UC

Given the results above and those in [18], we have that $\mathcal{F}_{\texttt{OneSeal}}^{DE}$ (or $\mathcal{F}_{\texttt{OneSeal}}^{\texttt{purpoted}DE}$) can create receiver-strong and sender-strong weak UC bit-commitments, which in turn can be amplified to obtain regular UC bit-commitments. The $\mathcal{F}^{BC}$ functionality or a flavour of it (see $\mathcal{F}_{COM}$ in [6]) constitutes a sufficient setup to UC-realize a ZK protocol [6]. Under these circumstances, it is definitely interesting to investigate the existent implications between different sort of weak bit-commitment, regular bit-commitment and tamper-evident envelopes, in the UC framework.

Firstly, note that a multiple commitment $\mathcal{F}_{MCOM}$ [6] setup suffices to UC-realize an $\mathcal{F}_{\texttt{EscapeThenMayCheat}}^{q-\text{WBC}}$, $\mathcal{F}_{\texttt{LearnAtCommitment}}^{q-\text{WBC}}$ or an $\mathcal{F}_{\texttt{LearnAtOpening}}^{q-\text{WBC}}$ functionality, for some $q \in (0, 1)$. (We recall the $\mathcal{F}_{MCOM}$ [6] functionality in the full version of this paper [3]). In other words, several instances of the regular bit-commitment

functionality $\mathcal{F}^{BC}$ suffice to UC-construct a sender-strong weak bit-commitment. In particular, three regular bit-commitment $\mathcal{F}^{BC}$ functionalities (see the full version of this paper [3]) UC-construct a $\frac{2}{3}$-WBC which is sender-strong. Let a protocol $\mathcal{P}$ be obtained from protocol `CommitEnablesCheat` where the creation and transmission of the three envelopes is respectively replaced by creation and transmission of three commitments using $\mathcal{F}_{MCOM}$ or using three respective instances of $\mathcal{F}^{BC}$. An analogous fact holds also for $\mathcal{F}^{\frac{2}{3}-\mathrm{WBC}}_{\texttt{LearnAtOpening}}$, where to construct the protocol $\mathcal{P}$ we use `OpenEnablesCheat` instead of `CommitEnablesCheat`.

Secondly, as a consequence of Theorem 3, note that all sender-strong weak BCs UC-imply regular BCs.

Thirdly, it should be answered whether bit-commitment setup suffice to UC-realize the $\mathcal{F}^{DE}_{\texttt{OneSeal}}$ distinguishable tamper-evident envelope functionality. We conjecture that question 1 has a negative answer. This is intuitively due to the fact that in bit-commitments it is the sender who opens the commitment and, in an envelope-emulating protocol, it should be the envelope's creator who needs to perform the corresponding opening. But, in order for this to be generally possible, an envelope creator ought to know the current envelope holder, which is not the case in our formalisations (i.e., envelopes can be passed on from holder to holder, without the notification of the creator).

To sum up, amplification proofs considered, we have completed the picture of UC-realisability of different flavours of sender-strong weak BC with tamper-evident envelopes and of their relation to (almost) regular BC and receiver-strong weak BCs by Moran and Naor [18]. To some level, we can say that all weak BCs are equivalent to regular BCs. We leave the EUC or the GUC correspondents of the implications enumerated above as open questions.

5 Conclusions

Answering a variant of the open question in Moran and Naor's work [18] and several practical needs [9,14,11], we conclude that simple, sealed envelopes can also create sender-strong (weak) bit-commitments protocols. In the process, we have also discussed the fact that the protocols in [18] are not EUC-secure but only UC-secure. We mainly focused on creating sender-strong bit-commitments with the same level of security. Nevertheless, we showed how to modify the $\mathcal{F}^{DE}_{\texttt{OneSeal}}$ functionality given in Moran and Naor's work [18] such that we also create (weak) bit-commitment protocols that are EUC-secure. We showed lightweight amplification proofs of our WBC protocols. We lastly discussed some of the implications between UC weak BCs, UC regular BCs and distinguishable tamper-evident UC envelopes. The interest in *weak* BC protocol per se was motivated by both theoretical and practical reasons. The GUC-security of our schemes remains to be discussed.

Acknowledgements. The authors acknowledge the support of the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a centre supported by the Swiss National Science Foundation.

References

1. Awerbuch, B., Patt-Shamir, B., Peleg, D., Tuttle, M.: Collaboration of Untrusting Peers with Changing Interests. In: Proceedings of the 5th ACM Conference on Electronic Commerce, EC 2004, pp. 112–119. ACM, New York (2004)
2. Beaver, D.: Adaptive Zero Knowledge and Computational Equivocation (Extended Abstract). In: The 28th Annual ACM Symposium on Theory of Computing (STOC), pp. 629–638 (1996)
3. Boureanu, I., Vaudenay, S.: Several weak bit-commitments using seal-once tamper-evident devices. Cryptology ePrint Archive, Report 2012/380 (2012), http://eprint.iacr.org/2012/380
4. Brassard, G., Chaum, D., Crépeau, C.: Minimum Disclosure Proofs of Knowledge. Journal of Computer Systems Science 37, 156–189 (1988)
5. Crépeau, C.: Efficient Cryptographic Protocols Based on Noisy Channels. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 306–317. Springer, Heidelberg (1997)
6. Canetti, R.: A Unified Framework for Analyzing Security of Protocols. Electronic Colloquium on Computational Complexity (ECCC) 8(16) (2001)
7. Canetti, R., Dodis, Y., Pass, R., Walfish, S.: Universally Composable Security with Global Setup. Cryptology ePrint Archive, Report 2006/432 (2006), http://eprint.iacr.org/
8. Chandran, N., Goyal, V., Sahai, A.: New Constructions for UC Secure Computation Using Tamper-Proof Hardware. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 545–562. Springer, Heidelberg (2008)
9. Chin-Chen, C., Ya-Fen, C.: Efficient Anonymous Auction Protocols with Freewheeling Bids. Computers & Security 22(8), 728–734 (2003)
10. Damgård, I.: On the Existence of Bit Commitment Schemes and Zero-Knowledge Proofs. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 17–27. Springer, Heidelberg (1990)
11. Dane, G.: The Implementation of an Auction Protocol over Anonymous Networks (2000), http://research.microsoft.com/en-us/um/people/gdane/papers/partiiproj-anonauctions.pdf
12. Goyal, V., Ishai, Y., Sahai, A., Venkatesan, R., Wadia, A.: Founding Cryptography on Tamper-Proof Hardware Tokens. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 308–326. Springer, Heidelberg (2010)
13. Katz, J.: Universally Composable Multi-party Computation Using Tamper-Proof Hardware. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 115–128. Springer, Heidelberg (2007)
14. Kikuchi, H., Harkavy, M., Tygar, J.D.: Multi-round Anonymous Auction Protocols. In: Proceedings of the 1st IEEE Workshop on Dependable and Real-Time E-Commerce Systems, pp. 62–69. Springer (1998)
15. Mateus, P., Vaudenay, S.: On Tamper-Resistance from a Theoretical Viewpoint. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 411–428. Springer, Heidelberg (2009)
16. Moran, T., Naor, M.: Basing Cryptographic Protocols on Tamper-Evident Seals. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 285–297. Springer, Heidelberg (2005)

17. Moran, T., Naor, M.: Polling with Physical Envelopes: A Rigorous Analysis of a Human-Centric Protocol. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 88–108. Springer, Heidelberg (2006)
18. Moran, T., Naor, M.: Basing Cryptographic Protocols on Tamper-Evident Seals. Theoretical Computer Science 411, 1283–1310 (2010)
19. Moran, T., Segev, G.: David and Goliath Commitments: UC Computation for Asymmetric Parties Using Tamper-Proof Hardware. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 527–544. Springer, Heidelberg (2008)
20. Naor, M.: Bit Commitment Using Pseudo-Randomness. Journal of Cryptology 4, 151–158 (1991)
21. Stajano, F., Anderson, R.: The Cocaine Auction Protocol: On the Power of Anonymous Broadcast. In: Pfitzmann, A. (ed.) IH 1999. LNCS, vol. 1768, pp. 434–447. Springer, Heidelberg (2000)

Deterministic Random Oracles

Margus Niitsoo

University of Tartu, Liivi 2, 50409 Tartu, Estonia
Margus.Niitsoo@ut.ee

Abstract. The Random Oracle model popularized by Bellare and Rogaway in 1993 has proven to be hugely successful, allowing cryptographers to give security proofs for very efficient and practical schemes. In this paper, we discuss the possibility of using an incompressible but fixed, "algorithmically random" oracle instead of the standard random oracle and show that this approach allows for rather similar results to be proven but in a completely different way. We also show that anything provably secure in the standard random oracle model is also secure with respect to any algorithmically random oracle and then discuss the implications.

1 Introduction

Cryptology is the science of secure communication. Security, however, is often quite hard to achieve unconditionally, and one usually needs to base the security of a primitive on some set of computational assumptions. Nevertheless, there are cases where computational assumptions seem insufficient for a protocol that is fast enough for practical purposes. In these cases, the standard model is often augmented by giving access to an oracle that is chosen uniformly and at random from the set of all possible oracles. Proofs in such a Random Oracle Model (ROM) are usually considerably simpler and still offer rather convincing evidence that the result does indeed hold in practice. The approach of using random oracles has its roots in complexity theory [1]. In cryptology, its use was popularized by Bellare and Rogaway [2] and it has seen widespread use since then.

The near-ideal functionality supplied by such an oracle simplifies the proofs and often allows for very simple constructions to be proved secure with respect to very strong security properties. The main problem is that in reality, no such ideally random oracles are available and as such, one would need to use something else – usually either a hash function such as SHA-2 or a symmetric encryption primitive such as AES – in their place. Since the instantiation is not ideally random, however, the proof in the random oracle model is only of heuristic value and there are indeed constructions that are secure in the random oracle model but which have no secure poly-time implementations [3,4]. Nevertheless, it has proven to be a valuable heuristic, as most of the schemes that have been developed in such a way have not been broken yet even with concrete instantiations.

Random oracles are also used extensively in proving separation results and lower bounds for black box reductions. In this case, the random oracle is used as

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 88–103, 2012.

a provably secure instance of a specified primitive (such as a one-way function or collision resistant hash function) in order to create a computational world in which the base primitive does indeed exist whereas the thing that one should be able to construct from it does not.

When working in the ROM, one is not dealing with a single fixed oracle, but rather a random family of oracles where security is shown only on average. This can make it hard to say anything about any concrete oracle in particular. It is, of course clear that if a property holds on measure 1 of all the oracles, there have to exist fixed oracles for which it holds. This may not be enough for some applications, however. For instance, it seems to provide a major obstacle for generalizing separation results to the non-uniform model [5].

In this paper we show that practically everything that is secure in the ROM is also true with respect to *every* "algorithmically random" deterministic oracle. This gives a better characterization of the oracles by essentially giving a constructive specification of the measure 1 set for which the claim holds. Among other things, it allows one to freely replace the "truly" random oracle with a single fixed oracle that is only "algorithmically" random.

To prove our results, we will use Algorithmic Information Theory along with Chaitin-Kolmogorov-Solomonoff randomness (also called *algorithmic* or *deterministic* randomness), which is built around the notion of incompressible bit sequences. We show how to give cryptographic proofs in that model by demonstrating how to prove one-wayness of a function based directly on the algorithmically random oracle. We will then proceed to the main result, using a somewhat different side of AIT to give a simple and concise proof of the fact that using an algorithmically random oracle actually yields a stronger model than the classical ROM. We will then discuss the implications of the result to the previous work in the field.

To date, Chaitin-Kolmogorov-Solomonoff complexity has seen only very limited use in cryptology, mainly in the study of pseudo-randomness. For instance, Beth and Dai [6] showed that the linear complexity of a string is nearly equal to the Kolmogorov complexity for practically all bit strings. Its use in computational complexity theory has been somewhat more widespread, even being applied to the study of oracle separations in the random oracle model (see [7,8]). However, our approach differs markedly from the one used there by being considerably less technical and assuming far less background knowledge about algorithmic randomness. This should make our result more approachable to the wider audience and may perhaps serve as a simple and practical introduction into Algorithmic Information Theory for cryptologists.

2 Basic Terminology

In this section we will introduce the terminology used throughout the following paper.

We say that a function $\epsilon(n)$ is *negligible* when it is asymptotically smaller than any inverse polynomial.

The basic model in the following discussion will be a Turing machine with a binary alphabet $\{0,1\}$. It has three tapes: one input tape, one output tape and one working tape. Movement on the working tape is unrestricted, but only forward movement is allowed on both input and output tapes. In the following we assume the machines to have programs that always terminate on the inputs they are given (although the set of inputs can be restricted in which case the behavior on the other inputs does not concern us). This means that in general we are talking about computable partial functions $\phi : \{0,1\}^* \to \{0,1\}^*$ where $\{0,1\}^*$ is defined to be the set of all finite bit strings. We assume the existence of a standard coding $\langle \cdot, \cdot \rangle$ for pairs of bit strings into a single bit string and henceforth assume that a Turing machine can take any number of finite inputs on its input tape. We will also assume a canonical correspondence between natural numbers $\mathbb{N}$ and bit strings $\{0,1\}^*$ so that one set can be substituted for the other in the argumentation whenever convenient. We call a real number ξ (bitwise) computable when there exists a Turing machine $\mathsf{K} : \mathbb{N} \to \{0,1\}^*$ so that the output of $\mathsf{K}(n)$ is the first n bits of ξ. We will later also need machines that take one infinite input, which is then accounted for by appending that input on the tape after the finite inputs. We also fix a Universal Turing Machine $\mathsf{U} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that $\mathsf{U}(p,i) = x$ precisely when the p-th Turing machine (in some ordering of Turing machines) gives the output x on input i. In this case the bit string p is called the program for the Turing machine it stands for. We will use xy to mean the concatenation of x with y. We will also be talking about infinite bit sequences. We denote the set of all such sequences by $\{0,1\}^\omega$ and note that this is disjoint with $\{0,1\}^*$ that contains all finite strings. For an infinite bit sequence $\mathbf{x} \in \{0,1\}^\omega$ we will write $\mathbf{x}(n)$ for the $(n+1)$-th bit of the sequence (so the first bit is $\mathbf{x}(0)$) and $\mathbf{x}_{a:b}$ for the substring of the sequence formed by concatenation of $\mathbf{x}(a)\mathbf{x}(a+1)\ldots\mathbf{x}(b-1)$. For convenience, we use special notation $\mathbf{x}_n = \mathbf{x}_{0:n}$.

We will also make use of the standard (uniform) Lebesgue probability measure defined over $\{0,1\}^\omega$, which is implicitly assumed whenever probability or measure over infinite bit sequences is discussed, i.e. when noting that a property holds with measure 1 over $\{0,1\}^\omega$.

By a *polynomial-time* Turing machine we mean a machine that always halts within a number of steps polynomial in its input length. When talking about probabilistic Turing machines, we assume that the machines additionally have one extra input tape that is filled with independently and uniformly chosen random bits. When talking about non-uniform Turing machines, we assume that there is one extra input tape that has some *advice* information that is the same for all inputs of the same length. We note that for polynomial-time machines both models (and the combination of the two) can be simulated in polynomial time on the original three tape machine by packing the contents of the randomness tape and the advice tape both on the one input tape along with the input, as only a polynomially bounded number of bits could be read from either of them.

We also introduce the notion of an *Oracle Turing machine*, that has an additional functionality for dealing with an outside subroutine (oracle) function

$f : \{0,1\}^* \to \{0,1\}$. The oracle function f can be any well-defined deterministic function, which does not have to be computable. The oracle query is assumed to take just one step in terms of execution time. We will use $()^{\cdot}$ to differentiate between normal and oracle Turing machines. There is a trivial isomorphism between the set of all oracles and the set $\{0,1\}^\omega$ (where $f(n)$ is the n-th bit of the bit sequence) and we will make implicit use of it whenever convenient.

We will use the standard definition of a primitive used in the separation result literature, best explained in Reingold et al. [9]. In brief, a *primitive* $\mathcal{P}$ is a class of (not necessarily computable) functions intended to perform a security related task (e.g. data confidentiality, integrity etc.). Each primitive $\mathcal{P}$ is characterized by the success function $\epsilon\mathcal{P}k\cdot$, which for every instance f of $\mathcal{P}$, an adversary A, and the security parameter k returns the breakage success $\epsilon\mathcal{P}kf, \mathsf{A} \in [0,1]$ (the unit interval of real numbers). In this paper, both instances and the success functions are generally assumed to be computable, however.

3 The Random Oracle Model

The random oracle is often modeled informally as an agent that, when queried with input x, flips random coins to get a response y, except when the query x has occurred before, in which case it answers with the exact same response as it did before. The agent can be queried by both honest parties and adversaries and has to answer to both consistently. More formally, the random oracle model entails giving all parties access to an oracle that is chosen uniformly and at random from the set of all oracles. Success and failure probabilities are now defined over not only the randomness tapes of the parties but also over the choice of oracles.

In practice, the oracle, which is formally a function $\mathbf{o} : \{0,1\}^* \to \{0,1\}$ (or equivalently, a bit sequence $\mathbf{o} \in \{0,1\}^\omega$) is often used in the form of a family of functions $f_n : \{0,1\}^{m(n)} \to \{0,1\}^n$ (where m is a polynomial). Such a function can be constructed from a given bit sequence $\mathbf{o}$ (provided by the oracle) in a completely straightforward way by defining $f_n^{\mathbf{o}}(i) = \mathbf{o}_{S_n+ni,\ldots,S_n+n(i+1)-1}$ where $S_n = \sum_{j=1}^{n} j \cdot 2^{m(j)}$ is the length of the descriptions of $f_j^{\mathbf{o}}$ for $j < n$. This construction is bijective, meaning that if $\mathbf{o}$ is chosen uniformly among all candidates, so is the family $\{f_n : n \in \mathbb{N}\}$. Therefore, it makes no difference which of the two is taken as a basis for the definition of a random oracle. In the following, we will denote the standard random oracle as $\mathcal{O}$.

We will use the following definition

Definition 1. *Let g be an instance of a primitive $\mathcal{P}$ and let $\epsilon_{\mathsf{A}}^{\mathbf{o}}(n)$ be the probability that adversary A succeeds in breaking g_n relative to oracle $\mathbf{o} \in \mathcal{O}$. We say that g is secure against A relative to $\mathbf{o}$ if $\epsilon_{\mathsf{A}}^{\mathbf{o}}(n)$ is negligible in n. We say that g is secure in the ROM if for all adversaries A, g is secure with probability 1 (taken over $\mathbf{o} \leftarrow \mathcal{O}$).*

There is another, stronger definition of security in the ROM, which is favored in the more practical branches of cryptology and which basically states that a primitive is secure if for all adversaries A, the probability (taken over both $\mathbf{o}$

and the random coins of A) of A succeeding is negligible. Anything secure w.r.t. that stronger model is also secure with respect to our definition. The distinction between the two models is mainly technical, but there are also some deeper issues. The critique of Canetti et al. [3] illustrates the difference pretty well, as it is easy to see that it only makes sense with respect to the stronger model. Essentially, our definition corresponds to the notion of non-programmable random oracle of Nielsen [4].

In this work, we will basically show that the measure 1 set of oracles for which the security is proven in this case is sure to contain the set of all algorithmically random sequences, thus allowing us to replace the "random" random oracle with a fixed, algorithmically random bit sequence. We will now give a brief introduction into the field of algorithmic randomness and make the idea of algorithmically random oracles somewhat more precise.

4 Algorithmic Information Theory and the Algorithmically Random Oracle Model

Algorithmic Information Theory (AIT) stems from independent work by Kolmogorov [10], Chaitin [11] and Solomonoff [12] who all tried to develop a purely algorithmic and deterministic concept of randomness. The intuition behind their work is that a bit string can be considered random when it cannot be compressed or (equivalently) contains no significant testable patterns.

In the context of modern algorithmic information theory, the notion of incompressibility is usually formulated in terms of self-delimiting programs[1]. In this case the Universal Turing machine U is such that it is assumed to terminate only when the machine halts with its input tape head at the last bit of the input. This *self-delimiting* requirement essentially means that the length of the program has to be encoded inside the program. For our purposes, it is basically just a minor technical constraint, but the assumption is of central importance when using the results from AIT. The self-delimiting requirement is not restrictive as any non-self delimiting program can easily be converted into a self-delimiting format by prefixing it with its length and a short (self-delimiting) program that interprets the length in the correct way.

The self-delimiting description-length complexity (or *Chaitin complexity* for short) $H(x)$ of a bit string x is then defined as the length $|x'|$ of the shortest (self-delimiting) program and input pair $x' = \langle p, i\rangle$ such that $\mathsf{U}(x') = x$. Due to the self-delimiting requirement, $H(x)$ can actually be somewhat larger than $|x|$, but the difference is upper bounded by $O(\lg |x|)$. Finite strings of length n whose complexity is equal to $\Sigma(n) = \max\{H(x)|x \in \{0,1\}^n\}$ are defined as being *Chaitin random*.

[1] This has not always been the case. The self-delimiting model was first introduced by Levin [13] and popularized by Chaitin [14] and is seen by many as the biggest breakthrough in AIT [15].

In this work, however, we are mainly concerned with randomness of infinite bit sequences. There are many equivalent definitions, the most common of which is probably the following.

Definition 2 (Algorithmic Randomness). *A bit sequence* $\mathbf{x} \in \{0,1\}^\omega$ *is said to be* algorithmically random *when there exists a constant* $c_{\mathbf{x}}$ *(the* randomness threshold*) such that all of its prefixes* $\mathbf{x}_n$ *have complexity* $H(\mathbf{x}_n) \geq n - c_{\mathbf{x}}$.

It turns out that such sequences do exist and are, in fact, quite common. The following result is due to Martin-Löf [16].

Theorem 1. *A measure* 1 *of bit sequences* $\mathbf{x} \in \{0,1\}^\omega$ *are algorithmically random.*

There are also concrete examples of such sequences, the best known of which is probably the Chaitin's constant, Ω_{U} – the halting probability of a given Universal Turing Machine U over all its valid inputs. This number is uncomputable in the strongest possible sense in that any program computing its n initial bits correctly has to have a length of at most $n - c$ for some fixed constant[2] c.

In some sense, an algorithmically random sequence serves as a good replacement for "real" randomness in the probability theory sense. The Algorithmically Random Oracle Model (AROM) is thus very simple – we just augment the normal computational world with an oracle access to an algorithmically random bit sequence. Depending on the circumstances, one may actually wish to choose a particular random bit sequence that has other good properties besides being algorithmically random. In this work, however, we will get by with properties that every such sequence possesses.

Definition 3. *We say that* g *is secure in the AROM if for all adversaries* A, g *is secure relative to all algorithmically random oracles* $\mathbf{o} \leftarrow \mathcal{O}$.

We will demonstrate that replacing "real" randomness with its algorithmic counterpart allows us to use description-length based proof methods in cryptology. For that, we begin by proving that a secure one-way function exists in AROM. This is an analogue of a similar result proven by Impagliazzo [18] for the standard ROM.

5 A One-Way Function in an Algorithmically Random Oracle Model

One-way functions are one of the most widely used primitives in cryptology.

Definition 4 (One-Way Function). *A family of functions* $\{f_n : \{0,1\}^{m(n)} \to \{0,1\}^n | n \in \mathbb{N}\}$ *is* one-way *if for every randomized polynomial-time adversary* A, *the probability* $\Pr_{x \leftarrow \{0,1\}^{m(n)}}[x' := \mathsf{A}^f(n, f(x)) : f(x') = f(x)]$ *is negligible.*

[2] The same also holds for proving the correctness of the computed bits i.e. you can only prove the correctness of a limited number of bits in any finite theory. Remarkably, there even exists a choice of U for which not a single bit of Ω_{U} can be proven correct in ZFC [17].

For every algorithmically random string $\mathbf{r}$, the standard construction (described in Section 3) results in a function family that is one-way in a very strong sense. To prove that, however, we will first need a Lemma about the output distribution of such a construction. The proof of the Lemma will also serve as an introduction to the incompressibility arguments used in the following two theorems.

Lemma 1. *Let $\mathbf{r}$ be an algorithmically random sequence and let $f_n^{\mathbf{r}}$ be defined as above. Let $m = m(n)$ be such that $m \geq n$. Then for a sufficiently large n, no more than $m^3 2^{m-n}$ different inputs can map to the same output.*

Proof. To prove the claim, we try to construct a program that would print out the $n2^m$ bits of $\mathbf{r}$ corresponding to $f_n^{\mathbf{r}}$, but which, along with its inputs would itself be describable in less than $n2^m - c_{\mathbf{r}}$ bits. This leads to a contradiction, as it allows us to give a description of $\mathbf{r}_{S_{n+1}}$ that is shorter than should be possible for an algorithmically random sequence[3].

The description of $f_n^{\mathbf{r}}$ consists of two parts: the description of the algorithm followed by the bit strings describing its inputs. The algorithm itself is universal and the length of its description is independent of the lengths of its inputs. The algorithm is also assumed to be encoded in a self-delimiting fashion p so that it can determine where its own description ends and the inputs begin.

Let y_f be the most frequent output of $f_n^{\mathbf{r}}$ and let X be the set of inputs that leads to this output. Denote $k = |X|$.

We construct an algorithm that will take the following inputs: the numbers m and n, followed by the value y_f and its number of occurrences k. After them there is a list of k indices $i_0, \ldots, i_{k-1}$ that describe the set X . Indices are incremental, so that the elements of X in their lexicographic order would have indices $i_0, i_0 + i_1, \ldots, \sum_{j=0}^{k-1} i_j$. The last input contains all the contents of the yet unspecified blocks in their normal order.

The program works in the following way:

- Read m,n. Initialize a table of 2^m segments each of length n bits.
- Iterate over all the values $x \in X$ using $i_0, \ldots, i_{k-1}$:
  - Fill the segment corresponding to x with y_f
- Fill the table sequentially with the remaining bits of input.
- Output the contents of the table.

It should be clear that such a program will indeed output the first $n2^m$ bits of $\mathbf{r}$. We now turn to the question of how many bits are needed to encode the inputs. We firstly note that we need to be able to tell, when one of the inputs ends and another one begins. As such, we would like all the inputs to be encoded in a self-delimiting way.

There are many standard ways to achieve that [15]. The simplest approach would be to encode 0 as 00, 1 as 11 and to reserve 01 and 01 as the delimiting

[3] It is trivial to convert a short description of $f_n^{\mathbf{r}}$ to that of $\mathbf{r}_{S_{n+1}}$ once n is known by simply building a program that outputs first S_n bits of it's input verbatim and then runs the program describing $f_n^{\mathbf{r}}$ on the remaining input. Such a description is necessarily shorter as $n2^m$ bits suffice for the description of f_n if n is known.

symbols signaling that this is where the current input ends and the next one begins. To code an integer n in this manner would take $2\lg n + 2$ bits. There is a more efficient way to encode a bit string, however. Namely, instead of coding the whole string in this costly manner, we can start by first coding the length of the bit string and then just writing the bit string after it. When parsing, we can first read the length and then read length bits after it to get the string itself. Coding an integer in that way requires $\lg n + 2\lg\lg n + 2$ bits.

Coding m and n therefore takes $\lg n + \lg m + 2\lg\lg m + 2\lg\lg n + 4 = O(\lg m)$ bits. As n is known, coding y_f takes just n bits as it can just be written verbatim. Coding k is again standard, taking $\lg k + 2\lg\lg k + 2 < m + O(\lg m)$ bits. Coding the following list of indices will also be done in the standard way. However, we note that since $\sum_{j=0}^{k-1} i_j < 2^m$, the total length of their encoding cannot exceed $k\left(\lg\left(\frac{2^m}{k}\right) + \lg\lg\left(\frac{2^m}{k}\right) + 2\right)$, which follows directly from the fact that the uniform distribution always has maximal Shannon entropy. The remaining bits of $\mathbf{r}_{n2^m}$ can also be encoded verbatim as their number is completely determined by n, m and k. The inputs can thus be encoded with $n2^m + k\left(m - n + 2\lg m + 2 - \lg k\right) + m + O(\lg m)$ bits.

If $\mathbf{r}$ is an algorithmically random sequence, $k \geq m^3 2^{m-n}$ leads to a contradiction as it produces a coding that is asymptotically shorter than $n2^m - c_{\mathbf{r}}$. □

We will now proceed to the main theorem of this section, which states that an algorithmically random sequence will indeed yield a very strong one-way function. Our result is similar to that of Impagliazzo [18], where it was shown that measure 1 of all oracles will result in a one-way function secure in the non-uniform model. However, our result is stronger in that we explicitly state that the measure 1 set in question is that of algorithmically random sequences:

Theorem 2. *Let* $\mathbf{r}$ *be an algorithmically random sequence and let* $f_n^{\mathbf{r}}$ *be defined as before. If* $m = m(n)$ *is such that* $m \geq n$ *then* $\{f_n^{\mathbf{r}} | n \in \mathbb{N}\}$ *is an* ϵ*-secure one-way function (secure even in the non-uniform model).*

Proof. Let $\mathbf{r}$ and the function $m(n)$ be fixed and denote $f_n := f_n^{\mathbf{r}}$. Assume that there exists a (possibly non-uniform) polynomial-time oracle machine $\mathsf{A}^{\cdot}$ that is able to invert $\{f_n | n \in \mathbb{N}\}$ (i.e. $\mathsf{A}^{f_n}(n, y)$ can find a x such that $y = f_n(x)$ with probability greater than ϵ where the probability is taken over all the possible inputs $x \in \{0,1\}^m$). Since $\mathsf{A}^{\cdot}$ is polynomial-time, there exists a polynomial upper bound $q(n)$ on the number of oracle, randomness and advice queries that it can make. We show that existence of such an adversary will lead to a contradiction in a similar manner as in the previous proof, i.e we show how to use it to get an impossibly short description of the bits of $\mathbf{r}$ that correspond to f_n.

When constructing the shorter description, we will encode the adversary along with its randomness and advice tapes. Since the adversary is successful on average, there has to exist a randomness string for which the adversary is at least as successful as on average over all the randomness strings. Encoding the adversary algorithm (which is of fixed size) along with this randomness string and advice string can clearly be done in $O(q(n))$ bits.

Since $\mathsf{A}^{\cdot}$ is at least ϵ successful, Lemma 1 implies that there are at least $\frac{\epsilon 2^m}{m^3 2^{m-n}} = m^{-3}\epsilon 2^n$ different output values that it can invert successfully, as in the worst case, each output value that can be inverted can have at most $m^3 2^{m-n}$ input values correspond to it. Let Y be the set of all such values. Therefore $|Y| > m^{-3}\epsilon 2^n$.

We can assume that whenever $A^{f_n}(y) = x$ such that $f_n(x) = y$ (so $y \in Y$), the query $f_n(x)$ occurs somewhere within the execution of $\mathsf{A}^{\cdot}(y)$ – if not, we can augment $\mathsf{A}^{\cdot}$ by forcing it to query its output from f_n before actually returning it (which does not increase the program length or running time by more than $O(1)$). This means that we can define for all $y \in Y$ a number k_y as the index of the first query for x such that $f_n(x) = y$ that is made within the execution of $A^{f_n}(y)$. Note that $k_y \leq q(n)$ and as such we can write it with just $\lg q(n)$ bits, if $q(n)$ is known beforehand.

However, for theoretical considerations it is somewhat easier to assume the adversary successful only on a subset of Y with a certain structure. Let y_0 be the (lexicographically) first element in Y and define y_i, $i > 0$ as the first such element of Y that is not given as an answer to the oracle queries made by the adversary $\mathsf{A}^{\cdot}$ on any of the previous values $y_0, \ldots, y_{i-1}$. Now define $Y' = \{y_0, y_1, \ldots, y_k\}$. It should be clear that $|Y'| > q(n)^{-1}|Y| > q^{-1}(n)m^{-3}(n)\epsilon 2^n$ since each new element added to Y' can stop at most $q(n)$ elements of Y from also being added. This selection ensures us that if $\mathsf{A}^{\cdot}$ is run consecutively on the elements of Y' then the first time that a query to f resulting in $y \in Y'$ is seen is precisely within the computation of $\mathsf{A}^{\cdot}(y)$. Let $\epsilon' = |Y'|2^{-n}$.

The description we construct for $\mathbf{r}_{n2^m}$ is composed of the integers m, n, l, the program code for $\mathsf{A}^{\cdot}$ with its advice and randomness tapes, the list of indices i (encoded as in the previous proof) for Y' and the corresponding values of k_y and finally the remaining bits of $\mathbf{r}$ that are in a specific order dependent on $\mathsf{A}^{\cdot}$.

The description algorithm is the following:

- Read m,n. Initialize a table of 2^m segments each of length n bits.
- Iterate through the index list for $y \in Y'$:
  - Start computing $\mathsf{A}^{\cdot}(y)$.
  - When you need to query $f(x)$ see if the result is already in the table for f. If not, assume that the next n bits in the store are just that value and add it to the table before using it within $\mathsf{A}^{\cdot}$.
  - Whenever you get to k_y-th query of $\mathsf{A}^{\cdot}$, just write y into the slot corresponding to the query x that was made and move on to the next value of y in Y'.
- Fill the table sequentially with the remaining bits of input.
- Output the contents of the table.

It is clear we can order the bits in the last part of the store of bits so that the final answer is indeed $\mathbf{r}_{n2^m}$.

Let us now consider the size of this description. The program we described is clearly of a fixed size p. The numbers m and n can be written in $O(\lg n)$ bits. Encoding $\mathsf{A}^{\cdot}$ along with its advice and randomness takes $l = O(q(n))$ bits. The index list and the values of k_y can be written in $|Y'|(\lg(\epsilon'^{-1}) + 2\lg\lg(\epsilon'^{-1}) +$

$\lg q(n)) \leq |Y'|(n - \lg|Y'| + 2\lg n + \lg q(n))$ bits. The last part of the bit store takes $n2^m - n|Y'|$ room because for each $y \in Y'$ we get one pair $f(x) = y$ for free from $\mathsf{A}^{\cdot}$. Adding everything up we get that the complexity of the description is $n2^m + l + |Y'|\,(2\lg n + \lg(q(n)) - \lg|Y'|)$. Thus, if $|Y'| > n^3 q(n)$ (i.e $\epsilon > n^3 q^2(n) m^3(n) 2^{-n}$), this contradicts the fact that $H(\mathbf{r}_{n2^m}) > n2^m - c_{\mathbf{r}}$ for large enough n.

This means that even a relatively low inversion rate of an adversary will lead to a contradiction. □

6 One-Way Permutation from an Algorithmically Random Permutation Oracle

In most cases, the construction of the required "randomness" functionality (such as a hash function or a pseudorandom generator) is pretty straightforward. However, there are cases where the randomness required is assumed to have more structure. We will now show how to prove similar results in these cases as well.

In these cases, we cannot use the random string as the oracle directly. We can, however, build a computable injective correspondence O between the bit sequences $\{0,1\}^\omega$ and the oracles $\mathfrak{F}$ that we want. It turns out that this is sufficient.

Formally, the construction itself needs to behave as an oracle, i.e. have the shape $\mathsf{O} : \{0,1\}^* \to \{0,1\}$. For ease of understanding, we will instead construct a function $\mathsf{O} : \mathbb{N} \times \{0,1\}^* \to \{0,1\}^*$ so that $\mathsf{O}(n,\cdot)$ is a permutation of n bits. This is just for notational convenience and can easily be accounted for with the standard definition by using the standard pairing function $\langle\cdot,\cdot\rangle$ and defining $\mathsf{O}'(\langle n,k,j\rangle)$ to equal the j-th bit of $\mathsf{O}(n,k)$ whenever $j < n$ and $k < 2^n$ and to be 0 otherwise.

The construction we use is completely straightforward, but very slow, working in exponential-time. There are $(2^n)!$ different permutations of n bits so encoding one of them uniformly at random requires $\lg(2^n)!$ bits on average. Since $(2^n)!$ is not a power of two, however, this figure only holds on average and the number of bits required for uniform sampling is unbounded in the worst case. However, we claim that this poses no real problems and that we can work as if $\lceil \lg N \rceil$ bits allowed uniform selection from N elements even when N is not a power of two (see Appendix A for justification for this claim).

We fix a computable canonical enumeration function P for permutations (so that $\mathsf{P}(N,i,\cdot)$ is the i-th N-element permutation). We will construct the oracle $\mathsf{O}^{\mathbf{x}}(n,\cdot) = P(2^n, \mathsf{S}(\mathbf{x},n),\cdot)$, where S is a sampling routine that for each n independently samples an index from $(2^n)!$ choices based on the bit sequence $\mathbf{x}$. This concludes the construction.

We will now show that whenever $\mathbf{r}$ is an algorithmically random string, $\{f_n^{\mathbf{r}} := \mathsf{O}^{\mathbf{r}}(n,\cdot)\}$ is a one-way permutation in a very strong sense. This somewhat strengthens an analogous result of Gennaro and Trevisan [19] who proved the same, but in the standard ROM.

Theorem 3. *If* $\mathbf{r}$ *is an algorithmically random sequence, then* $f_n^{\mathbf{r}}$ *(as defined above) is a one-way permutation (again secure even in the non-uniform model).*

The proof is completely analogous to that of Theorem 2, except for the fact that since inputs and outputs are in a one-to-one correspondence, we no longer need to encode the list of indices i_n, meaning that encoding takes $2^n(n - \lg e) - |Y'|(\lg |Y'| - \lg q(n) - \lg e) + O(q(n))$, where $(n - \lg e)$ is due to the upper bound provided by the Stirling approximation for factorials.

It is pretty easy to convince oneself that similar proof techniques can be used for other types of oracles as well, provided they satisfy the following simple computability restriction.

Definition 5 (Computable oracle family). *We say that a family of oracles* $\mathfrak{F}$ *is a* computable oracle family *if there exists a computable function* $\mathsf{O} : \{0,1\}^* \times \{0,1\}^\omega \to \{0,1\}$ *so that* $\mathsf{O}(\cdot, \mathbf{x})$ *terminates for all algorithmically random choices*[4] *of* $\mathbf{x}$ *and so that* $\mathfrak{F} = \mathsf{O}(\cdot, \mathcal{O})$ *(with the proper distribution assuming* $\mathcal{O}$ *is the standard uniformly distributed random oracle).*

In principle, this definition also allows the oracle to be augmented with extra functionality (instead of just structure), which in principle means one can easily make the proofs work when they are relative to multiple oracles simultaneously. This is useful in the context of oracle separations. For instance, it allows one to combine a random oracle with a **PSPACE** oracle (which is the approach used in [20]) or even embed a special "universal collision finder" as is done in [21].

7 Relation to the Random Oracle Model

In this section we prove a somewhat surprising result – that AROM is nearly equivalent to classical ROM. Our proof makes use of an alternative characterization of algorithmic randomness due to Martin-Löf [16] that is based on computable tests for randomness.

Definition 6. *Let* $n \in \mathbb{N}$ *and* $x \in \{0,1\}^\omega$*. A computable partial function* $T(n, x)$ *is called a* computable test for randomness *when there exists a computable function* $\alpha : \mathbb{N} \to [0, 1]$ *such that*

- $\lim_{n\to\infty} \alpha(n) = 0$
- *For all* $n \in \mathbb{N}$*,* $T(n, x)$ *halts for less than a measure* $\alpha(n)$ *of inputs* $x \in \{0,1\}^\omega$*.*
- *For all* $n \in \mathbb{N}$*,* $T(n + 1, x)$ *halts implies that* $T(n, x)$ *also halts.* [5]

[4] Again, see Appendix A for the motivation for this relaxation.

[5] The third condition essentially guarantees the test is for the same pattern for different values of n. It just states that if a string is labeled non-random for a larger confidence parameter (where there are less non-random strings), it should also have been labeled as non-random for all the previous confidence parameters.

In essence, a computable test for randomness is a program that runs on a randomness string, looking for statistically significant patterns where $\alpha(n)$ is the significance level. A randomness string is then labeled non-random when such a pattern be found for arbitrarily small significance levels.

We note that from the computability point of view, this definition is highly asymmetric – it is expected to finish computation only on the non-random instances. This makes sense, considering we are looking for uncommon patterns, since if no such pattern is to be found, the program may keep on searching for it indefinitely, only stopping if it finds something.

Such tests can be used to provide for an alternative characterization of algorithmically random sequences.

Theorem 4 (Martin-Löf). *An infinite sequence $x \in \{0,1\}^\omega$ is algorithmically random precisely when no computable test for randomness halts on it for infinitely many values of n.*

Before stating our main theorem, we will need to introduce an additional technical assumption.

Definition 7. *We say that the security criterion of a primitive $\mathcal{P}$ is computable if for any given poly-time adversary construction A, it's success probability $\epsilon_{\mathsf{A}}(k)$ is a computable function of the security parameter k.*

This computation is usually completely straightforward – given k, it is normally possible to run A sequentially on all the valid inputs with all the possible outcomes of random coins and this is also usually sufficient to determine the success probability (even if it is defined as a similarity of output distributions, for instance). For non-uniform adversaries, one can add an outer loop that runs the computation over all the possible advice strings and outputs the maximal success probability achieved by any fixed advice. As such, this seems to be a purely technical restriction.

Theorem 5. *Assume that a construction $\mathsf{C}^{\mathcal{O}}$ is an instance of some primitive $\mathcal{P}$ (that has a computable security criterion) w.r.t. a random oracle $\mathcal{O}$. Then $\mathsf{C}^{\mathcal{O}}$ is secure in the ROM precisely when $\mathsf{C}^{\mathbf{r}}$ is secure for all algorithmically random bit sequences $\mathbf{r}$.*

Proof. Right-to-left implication is trivial. If $\mathsf{C}^{\mathbf{r}}$ is secure for all algorithmically random bit sequences $\mathbf{r}$ then it is also secure in the ROM due to Theorem 1.

Left-to-right implication is just slightly more complicated. Assume the contrapositive, i.e. that there exists an algorithmically random string $\mathbf{r}$ for which $\mathsf{C}^{\mathbf{r}}$ is not secure. This means that there exists a poly-time adversary construction $\mathsf{A}^{\mathbf{o}}$ which succeeds with non-negligible probability on infinitely many different values of the security parameter n.

Since the security criterion is computable, the success probability of $\mathsf{A}^{\mathbf{x}}$ for the security parameter k is a computable function $\epsilon_{\mathsf{A}}(\mathbf{x}, k)$. Since $\epsilon_{\mathsf{A}}(\mathbf{r}, k)$ is non-negligible, there exists a (computable) c such that $\epsilon_{\mathsf{A}}(\mathbf{r}, k) > k^{-c}$ infinitely often. Let $\mathsf{T}(\mathbf{x}, k)$ be a Turing machine that sequentially checks $k, k+1, k+2, \ldots$

and halts when it finds a k' so that $\epsilon_{\mathsf{A}}(\mathbf{x}, k') > k'^{-c}$. It is easy to verify that T satisfies the criterions for a computable test for randomness and that it also halts for all values of k when given $\mathbf{r}$ as input. This contradicts the fact that $\mathbf{r}$ is an algorithmically random sequence.

It should be fairly easy to see that the proof will still work when the standard random oracle is replaced by a computable oracle family $\mathsf{O}(\cdot, \mathcal{O})$. For completeness, we will state that result formally as well.

Corollary 1. *Assume that a construction* $\mathsf{C}^{\mathfrak{F}}$ *is an instance of some primitive* $\mathcal{P}$ *(that has a computable security criterion) w.r.t. a given computable oracle family* $\mathfrak{F} = \mathsf{O}(\cdot, \mathcal{O})$*. Then* $\mathsf{C}^{\mathfrak{F}}$ *is secure in the ROM precisely when* $\mathsf{C}^{\mathsf{O}(\cdot,\mathbf{r})}$ *is secure for all algorithmically random bit sequences* $\mathbf{r}$*.*

8 Discussion

Theorem 5 shows that AROM and ROM are nearly equivalent. One slight difference may arise when one chooses a specific algorithmically random sequence that has extra properties. However, this may very well be desirable, especially considering that one could do part of the proof in the standard ROM, then translate it to AROM by following the same steps as in the proof of Theorem 5 and then carry out the final part of the proof by making an additional assumption about the used bit sequence.

There is one more factor that needs to be taken into account. In the comments about definition 7, we noted that non-uniform adversaries are not a problem. However, there is a small nuance there that needs to be elaborated on. In most proofs for ROM, security is proven in a completely black-box way i.e. without assuming that the advice string depends on the oracle. In these cases, all that can be guaranteed by our method in AROM is security against uniform adversaries. However, we believe it to be only a somewhat technical point as "generic" non-uniform advice is likely to be of only very limited help.

What may actually prove useful, however, is oracle-dependent advice which was studied in the context of the standard ROM by Unruh [22]. He used clever information-theoretic tricks to show that in many cases, a polynomial-length oracle-dependent advice string can just be dealt with by fixing a small part of the oracle and then choosing the rest uniformly and at random like in the classical ROM. His methodology is fairly general and allows already known results to be re-proven in the stronger model in a fairly standard way. The results in this model achieve true non-uniform security and are the ones that can be carried over with full strength to the AROM. It is also worth noting that Theorems 2 and 3 achieve non-uniform security in that stronger sense.

9 Conclusions and Further Work

The model of algorithmically random oracles shows a lot of promise. In this work, we have shown that AROM is equivalent to the standard ROM in all but some

minor technical details. We have also shown how security proofs can be directly based on description length argumentation, which allows for more algorithmic and possibly cleaner proofs.

We have shown how to use the methods of AIT to prove security of a cryptographic primitive in AROM. As, by Theorem 5, a security proof for all algorithmically random oracles implies a proof relative to a standard random oracle, this essentially provides a completely new proof technique for the ROM. It would be nice to see this proof method actually being used for new and practical security proofs.

As described in Section 8, AROM can also be slightly stronger than the standard ROM when one assumes additional properties beside the algorithmic randomness. It would, therefore, be very interesting to see a proof that works in AROM but not in the ROM. It would be even more interesting if the result proven in AROM in such a way was provably impossible in the ROM.

The model of AROM was first conceived as an alternative for the oracle extraction step used in oracle separations, which has the potential of generalizing to the non-uniform model of computation. As such, it would be very interesting for the author to see it being used for proving separation results or even as a basis for a framework for generalizing already existing separation results into the non-uniform model.

References

1. Bennett, C.H., Gill, J.: Relative to a random oracle A, $\mathbf{P}^A! = \mathbf{NP}^A! = \mathbf{co}-\mathbf{NP}^A$ with probability 1. SIAM J. Comput. 10(1), 96–113 (1981)
2. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security, pp. 62–73 (1993)
3. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM 51(4), 557–594 (2004)
4. Nielsen, J.B.: Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 111–126. Springer, Heidelberg (2002)
5. Buldas, A., Laur, S., Niitsoo, M.: Oracle Separation in the Non-uniform Model. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 230–244. Springer, Heidelberg (2009)
6. Beth, T., Dai, Z.-D.: On the Complexity of Pseudo-random Sequences - or: If You Can Describe a Sequence It Can't Be Random. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 533–543. Springer, Heidelberg (1990)
7. Lutz, J.H.: Almost everywhere high nonuniform complexity. In: Structure in Complexity Theory, pp. 37–53 (1989)
8. Kautz, S.M., Miltersen, P.B.: Relative to a random oracle, **NP** is not small. Journal of Computer and System Sciences 53(2), 235–250 (1996)
9. Reingold, O., Trevisan, L., Vadhan, S.: Notions of Reducibility between Cryptographic Primitives. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 1–20. Springer, Heidelberg (2004)

10. Kolmogorov, A.N.: Three approaches to the quantitative definition of 'information'. Problems of Information Transmission 1, 1–7 (1965)
11. Chaitin, G.J.: On the length of programs for computing finite binary sequences. Journal of the ACM 13(4), 547–569 (1966)
12. Solomonoff, R.J.: A formal theory of inductive inference. Information and Control 7(2,3), 1–22, 224–254 (1964)
13. Levin, L.A.: Laws of information conservation (nongrowth) and aspects of the foundation of probability theory. Probl. Peredachi Inf. 10(3), 30–35 (1974)
14. Chaitin, G.J.: A theory of program size formally identical to information theory. Journal of the ACM 22(3), 329–340 (1975)
15. Claude, C.: Information and Randomness: An Algorithmic Perspective. Springer-Verlag New York, Inc. (1994)
16. Martin-Löf, P.: The definition of random sequences. Information and Control 9(6), 602–619 (1966)
17. Solovay, R.M.: A version of omega for which ZFC can not predict a single bit. Technical report, CDMTCS (1999)
18. Impagliazzo, R.: Very strong one-way functions and pseudo-random generators exist relative to a random oracle (1996) (manuscript)
19. Gennaro, R., Trevisan, L.: Lower bounds on the efficiency of generic cryptographic constructions. Electronic Colloquium on Computational Complexity (ECCC) 7(22) (2000)
20. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: Proceedings of 21st Annual ACM Symposium on the Theory of Computing, pp. 44–61 (1989)
21. Simon, D.R.: Findings Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions? In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 334–345. Springer, Heidelberg (1998)
22. Unruh, D.: Random Oracles and Auxiliary Input. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 205–223. Springer, Heidelberg (2007)

A Notes on Uniform Selection

Oracles are quite often sampled uniformly and at random from a finite set of N different choices or are constructed based on a countable number of such samplings (for example, when first choosing a random permutation and then choosing a collision query reply for each possible circuit-input pair, as in [21]). This can be somewhat tricky when the choice has to be made based on a uniformly distributed bit sequence $\mathbf{x} \in \{0,1\}^\omega$. When $N = 2^k$ is a power of two, this is a simple matter of just interpreting the first k bits of $\mathbf{x}$ as the index and choosing the choice corresponding to them.

When N is not a power of two, there still exists a rather straightforward method of uniform sampling. Let $n := \lceil \lg N \rceil$. The procedure works as follows: Take the first n bits of $\mathbf{x}$. If the value encoded in these bits is less than N, use that as the result. If not, recursively take the next n bits and repeat the process.

This sampling procedure is not guaranteed to terminate for every choice of $\mathbf{x} \in \{0,1\}^\omega$. However, when this happens, $\mathbf{x}$ is compressible, as one can code $\mathbf{x}_{kn}$ with just $kn - k + 2\lg k + c$ bits. This is easy to see since the set of n bit blocks for which a process is continued is of size $2^n - N < 2^{n-1}$ and one can thus reconstruct the string based on the number of blocks k and $n-1$ bits per each block. This, among other things, implies that the measure of bit strings for which the procedure does not terminate has measure zero. It is easy to verify that this is still the case even when the bit sequence is used not for a single sampling but for a countable number of samplings i.e. that all such sampling attempts will succeed in finite time for every algorithmically random $\mathbf{x}$. We also note that if N is known, the string that encodes the choice is self-delimiting as the selection algorithm is deterministic and it is easy to determine when it halts.

Suppose now that $\mathbf{r}$ is an algorithmically random sequence and suppose one part of the oracle was chosen by this sampling procedure from the set of N elements. Suppose further that the existence of some adversary $\mathsf{A}^{\cdot}$ would imply that this part of the oracle (and hence of $\mathbf{r}$) could be encoded in less than $\lg N - f(N)$ bits for some $f(N) = \omega(1)$. Then it is also clear from the previous paragraph that if the bit string that resulted in this choice was of length kn (so choice was made based on the k-th block), that choice string could also be encoded with just $\lg N - f(N) + (k-1)n + c$ bits. This means that the existence of $\mathsf{A}^{\cdot}$ would still lead to a contradiction even when this sampling method is used.

On the (Non-)Equivalence of UC Security Notions

Oana Ciobotaru

Saarland University
Saarbrücken, Germany
ociobota@mpi-inf.mpg.de

Abstract. Over the years, various security notions have been proposed in order to cope with a wide range of security scenarios. Recently, the study of security notions has been extended towards comparing cryptographic definitions of secure implementation with game-theoretic definitions of universal implementation of a trusted mediator. In this work we go a step further: We define the notion of game universal implementation and we show it is equivalent to weak stand-alone security. Thus, we are able to answer positively the open question from [17,18] regarding the existence of game-theoretic definitions that are equivalent to cryptographic security notions for which the ideal world simulator does not depend on both the distinguisher and the input distribution.

Additionally, we investigate the propagation of the weak stand-alone security notion through the existing security hierarchy, from stand-alone security to universal composability. Our main achievement in this direction is a separation result between two variants of the UC security definition: 1-bit specialized simulator UC security and specialized simulator UC security. The separation result between the UC variants was stated as an open question [23] and it comes in contrast with the well known equivalence result between 1-bit UC security and UC security.

Keywords: security models, UC security, time-lock puzzles, game theory.

1 Introduction

Nowadays we rely more and more often for everyday tasks on security protocols. Moreover, the number of contexts where the use of security protocols is required by law or expected by users has also grown rapidly in recent years. A wide range of security properties have been defined and implemented into real-world systems, but so far there is no unique notion that fulfills all requirements: For example, a given notion may ensure strong security guarantees, but comes at the price of inefficiency or it offers good scalability in practice, but there are scenarios where it is too permissive. In order to ensure the most appropriate security notion is chosen when designing a system, one should know very well how various security notions relate to each other.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 104–124, 2012.

Recently the view on security definitions has been extended [18] with the incipient study of the equivalence relation between weak precise secure computation and a weak variant of the game theoretic notion of universal implementation for a trusted mediator. However, it is still left as an open problem [17,18] how to obtain such a comparisons for other, possibly stonger security notions.

1.1 Contribution

We have a three fold contribution.

First, we relate the notion of weak stand-alone security[1] to the emerging game-theoretic concept of universal implementation [17,18]. In contrast to previous work, for our result we use a variant of universal implementation that discards the cost of computation. We are able to answer positively the open question from [17,18] regarding the existence of game-theoretic concepts that are equivalent to cryptographic security notions where the simulator does not depend on both the input distribution and the distinguisher.

Second, we present a separation result between two variants of UC security: 1-bit specialized simulator UC security and specialized simulator UC security. The separation result between the UC variants was stated as an open question [23] and it comes in contrast with the well known equivalence result between 1-bit UC security and UC security. Both variants of the UC security notion are obtained from the UC security definition by changing the order of quantifiers[2]. Thus, we continue the line of study started by [8,23]. In order to obtain the separation, we first show that the 1-bit specialized simulator UC security is equivalent to a seemingly weaker version of security, namely weak specialized simulator UC security[3].

The main proof technique used in our separation result is to employ a cryptographic tool called time-lock puzzles. Intuitively, this cryptographic tool can be used for comparing the computational power of two different polynomially bounded Turing machines. In order to achieve the separation result, we use time-lock puzzles from which we derive a result interesting also on its own, mainly a construction of a one-way function and a hard-core predicate.

Third, we study the propagation of weak security notion through the hierarchy security definitions. More precisely, we show that the notion weak security composed under concurrent general composition is equivalent to 1-bit specialized

[1] The difference between stand-alone security and weak stand-alone security is in the order of quantifiers. For stand-alone security, the simulator is universally quantified over all distinguishers and input distributions. As detailed in Sect. 2, for our notion of weak security the simulator depends only on the distinguisher and not on the input distribution. This comes in contrast with [18], where the simulator for weak precise secure computation depends on both distinguisher and input distribution.

[2] This means that in contrast to the UC security definition, the simulator may depend on the environment.

[3] This notion, additionally to having the simulator depend on the environment, also has the simulator depend on the distinguisher that compares the views of the environment from the real and the ideal world.

simulator UC security, which is a variant of UC security. Together with our second result, this implies that weak stand-alone security and stand-alone security are not equivalent.

1.2 Background and Related Work

The initial work [31] on general security definitions highlighted the need for a framework expressing security requirements in a formal way. The first formal definition of *secure computation* was introduced by Goldreich et al.[11]. The first approaches for formally defining security notions [13,14] have taken into account only the stand-alone model. In this model, the security of the protocol is considered with respect to its adversary, in isolation from any other copy of itself or from a different protocol.

Micali and Rogaway [25] introduce the first study of protocol composition, which the authors call *reducibility*. The first security definition expressed as a comparison with an ideal process, as well as the corresponding sequential composition theorem for the stand-alone model are provided in [3]. The framework of *universally composable security*, for short UC security [4] allows for specifying the requirements for any cryptographic task and within this framework protocols are guaranteed to maintain their security even in the presence of an unbounded number of arbitrary protocol instances that run concurrently in an adversarially controlled manner.

The notion of *specialized simulator UC* security has been introduced in [23] and it was shown that this is equivalent to *general concurrent composability* when the protocol under consideration is composed with only one instance of any possible protocol. Changing the order of quantifiers in the context of security definitions has been previously used in [8,17,18] for strengthening or weakening given security notions.

In parallel with the UC framework, the notion of *reactive simulatability* has been developed [2,19,26,27,28,29]. A detailed review about the differences between reactive simulatability and universal composability notions can be read in the related work section from [4].

Our study of the relation between security and game theoretic notions has been triggered by the recently emerging field of rational cryptography, where users are assumed to only deviate from a protocol if doing so offers them an advantage. Rational cryptography is centered around (adapted) notions of game theory such as computational equilibria [7]. A comprehensive line of work already exists developing novel protocols for cryptographic primitives such as rational secret sharing and rational secure multiparty computation [1,9,10,15,16,22].

In a related line of work [17,18] it is considered that computation is costly for protocol participants who are defined as rational. The focus is on studying how costly computation affects participants' utilities and the design of appropriate rational protocols. A participant's strategy is defined as a Turing machine and both the input and the complexity of the machine (e.g., the running time or the space used by the machine for a given input) influence the utilities. The work [17,18] develops a game-theoretic notion of protocol implementation and

the authors show that a special case of their definition is equivalent to a weak variant of precise secure computation.

1.3 Organization

This work is structured as follows: In Sect. 2 we review security notions and in Sect. 3 we revise the game theoretic notion of universal implementation. In Sect. 4 we prove our separation result between specialized simulator UC security and 1-bit specialized simulator UC security. In Sect. 5 we show our equivalence relation between weak security under 1-bounded concurrent general composition and 1-bit specialized simulator UC security. In Sect. 5.1 we present the equivalence between our weak security notion and the game-theoretic notion of strong universal implementation. In Sect. 6 we present out conclusions and future directions.

2 Review of Security Notions

In this work we consider all parties and adversaries run in polynomial time in the security parameter k and not in the length of input. In this section we review two models of security under composition: concurrent general composition and universal composability. Both frameworks require the notion of (computational) indistinguishability given below.

Definition 1 (Computational Indistinguishability). *We call distribution ensembles* $\{X(k,z)\}_{k\in\mathbb{N},z\in\{0,1\}^*}$ *and* $\{Y(k,z)\}_{k\in\mathbb{N},z\in\{0,1\}^*}$ *computationally indistinguishable and we write* $X \equiv Y$*, if for every probabilistic distinguisher* $\mathcal{D}$*, polynomial in* k *there exists a function* ϵ*, negligible in* k*, such that for every* $z \in \{0,1\}^*$

$$|(Pr(\mathcal{D}(X(k,z)) = 1) - (Pr(\mathcal{D}(Y(k,z)) = 1)| < \epsilon(k)$$

A variant of this definition, which we call *indistinguishability with respect to a given adversary* $\mathcal{D}$ and we denote by $\overset{\mathcal{D}}{\equiv}$, is similar to the definition above, with the difference that "for every probabilistic distinguisher $\mathcal{D}$" is replaced with "for distinguisher $\mathcal{D}$". Such a definition we use in relation with our notion of weak security.

2.1 Universal Composability

The standard method for defining security notions is by comparing a real world protocol execution to an ideal world process execution. In the real world execution, a protocol interacts with its adversary and possibly with other parties. In the ideal world execution, an idealized version of the protocol (called ideal functionality) interacts with an ideal world adversary (usually called simulator) and possibly with other parties. The ideal functionality is defined by the security requirements that we want our protocol to fulfill.

On an intuitive level, given an adversary, the purpose of the simulator is to mount an attack on the ideal functionality; any probabilistic polynomial time (or PPT) distinguisher may try to tell apart the output of the interaction between the ideal functionality and the simulator and the output of the interaction between the protocol and its adversary. If for every adversary, a simulator exists such that the two outputs cannot be told apart by any PPT distinguisher, then our initial protocol is as secure as the ideal functionality, with respect to what is called *the stand-alone model*.

Definition 2 (Stand-alone Security). *Let ρ be a protocol and $\mathcal{F}$ an ideal functionality. We say* ρ securely implements *$\mathcal{F}$ if for every probabilistic polynomial-time real-model adversary $\mathcal{A}$ there exists a probabilistic polynomial-time ideal-model adversary $\mathcal{S}$ such that for every protocol input x and every auxiliary input z (given to the adversary) with $x, z \in \{0,1\}^{poly(n)}$, where k is the security parameter:*

$$\{IDEAL_{\mathcal{S}}^{\mathcal{F}}(k, x, z)\}_{k\in\mathbb{N}} \equiv \{REAL_{\rho,\mathcal{A}}(k, x, z)\}_{k\in\mathbb{N}}.$$

By $IDEAL_{\mathcal{S}}^{\mathcal{F}}(k, x, z)$ we denote the output of $\mathcal{F}$ and $\mathcal{S}$ after their interaction and $REAL_{\rho,\mathcal{A}}(k, x, z)$ denotes the output of the parties of ρ and adversary $\mathcal{A}$ after their interaction. If we allow the simulator to depend on the distinguisher, we obtain the weak stand-alone security notion.

The definition of universal composability follows the paradigm described above, however it introduces an additional adversarial entity which is called environment. The environment, usually denoted by $\mathcal{Z}$, is present in both the UC real world and UC ideal world. The environment represents everything that is external to the current execution of the real-world protocol or to the ideal functionality. Throughout the execution, both in the real and in the ideal world, the environment can provide inputs to parties running ρ or the ideal functionality $\mathcal{F}$ respectively, and to the adversary. These inputs can be a part of the auxiliary input of $\mathcal{Z}$ or can be adaptively chosen. Also $\mathcal{Z}$ receives all the output messages of the parties it interacts with and of the adversary. Moreover, the only interaction between the environment $\mathcal{Z}$ and the parties of ρ or $\mathcal{F}$ is when the environment sends the inputs and receives the outputs. Finally, at the end of the execution, the environment outputs all the messages it received. The environment is modeled as a PPT machine with auxiliary input. This auxiliary input captures the intuition that $\mathcal{Z}$ may learn some information from previous executions and it may also use it at any point later. Parties involved in the ideal execution give their inputs to the ideal functionality which computes some outputs and sends back these values.

When the protocol execution ends, $\mathcal{Z}$ outputs its view of that execution. In the real world, his view contains messages that $\mathcal{Z}$ has received from the adversary $\mathcal{A}$ and outputs of all parties of ρ. This is denoted by $EXEC_{\rho,\mathcal{A},\mathcal{Z}}(k, z)$, where k is the security parameter and z is the auxiliary input to $\mathcal{Z}$. Similarly, in the ideal world execution, the environment $\mathcal{Z}$ outputs its view which contains all the messages received from $\mathcal{S}$ as well as all messages that the dummy parties of $\mathcal{F}$

output to $\mathcal{Z}$. This is denoted by $EXEC_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)$. We are now ready to define UC security:

Definition 3 (UC Security). *Let ρ be a PPT protocol and let $\mathcal{F}$ be an ideal functionality. We say that ρ UC emulates $\mathcal{F}$ (or ρ is as secure as $\mathcal{F}$ with respect to UC security) if for every PPT adversary $\mathcal{A}$ there is a PPT simulator $\mathcal{S}$ such that for every PPT distinguisher $\mathcal{Z}$ and for every distribution of auxiliary input $z \in \{0,1\}^*$, the two families of random variables $\{EXEC_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)\}_{k\in\mathbb{N}}$ and $\{EXEC_{\rho,\mathcal{A},\mathcal{Z}}(k,z)\}_{k\in\mathbb{N}}$ are computationally indistinguishable.*

In the following we also use a relaxed version of this definition, where the order of quantifiers between the environment and the ideal-world simulator is reversed [23].

Definition 4 (Specialized Simulator UC Security). *Let ρ be a protocol and $\mathcal{F}$ an ideal functionality. We say that ρ emulates $\mathcal{F}$ under specialized simulator UC security if for every probabilistic polynomial time adversary $\mathcal{A}$ and for every environment $\mathcal{Z}$, there exists a simulator $\mathcal{S}$ such that for every input $z \in \{0,1\}^*$, we have:*

$$\{EXEC_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)\}_{k\in\mathbb{N}} \equiv \{EXEC_{\rho,\mathcal{A},\mathcal{Z}}(k,z)\}_{k\in\mathbb{N}}.$$

It had been shown [20] that the two notions defined above are not equivalent. In the above definition, the output of the environment is considered to be a string of arbitrary length. If the only change we make to the above definition is to consider environments that have a 1-bit output, we obtain the notion of *1-bit specialized simulator UC security*. It has been an open problem [23] whether considering only environments with one bit output would produce an equivalent definition. In this work we show this is not the case. If in the specialized simulator UC definition we let the simulator also depend on the distinguisher (i.e., the only machine to establish whether the output of the executions in the real UC world and in the ideal UC world cannot be told apart), then we obtain the notion of *weak specialized simulator UC security*. Both specialized simulator UC variants are defined in full detail in the full version [6].

2.2 Weak Security under 1- Bounded Concurrent General Composition

Similarly to the above security concepts, the notion of security under concurrent general composition [23] is defined using the real-ideal world paradigm. In this model, an external and arbitrary protocol π gives inputs to and collects outputs from an "internal protocol" that can be a real-world protocol or an ideal functionality. We denote by ρ the real-world protocol interacting with π and by $\mathcal{F}$ the ideal functionality. Protocol π may call multiple instances of the protocol it interacts with as long as all of them run independently and all its messages may be sent in a concurrent manner.

The computation in the ideal world is performed among the parties of π and an ideal functionality $\mathcal{F}$. Protocol π is providing $\mathcal{F}$ with inputs and after

performing necessary computations, $\mathcal{F}$ sends the results to parties of π. The messages between π and $\mathcal{F}$ are ideally secure, so the ideal adversary (or simulator) can neither read nor change them.[4]

The ideal-world honest parties follow the instructions of π and output the value prescribed by π. The corrupted parties output a special corrupted symbol and additionally the adversary may output an arbitrary image of its view. Let z be the auxiliary input for the ideal-world adversary $\mathcal{S}$ and let $\bar{x} = (x_1, ..., x_m)$ be the inputs vector for parties of π. The outcome of the computation of π with $\mathcal{F}$ in the ideal world is defined by the output of all parties of π and $\mathcal{S}$ and is denoted by $\{HYBRID^{\mathcal{F}}_{\pi,\mathcal{S}}(k, \bar{x}, z)\}_{k\in\mathbb{N}}$.

The computation in the real world follows the same rules as the computation in the ideal world, only that this time there is no trusted party. Instead, each party of π has an ITM that works as the specification of ρ for that party. Thus, all messages that a party of π sends to the ideal functionality in the ideal world are now written on the input tape of its designated ITM. These ITMs communicate with each other in the same manner as specified for the parties of ρ. After the computation is performed, the results are output by these ITMs to their corresponding parties of π. The outcome of the computation of π with ρ in the real world is defined by the output of all parties and $\mathcal{A}$ and is denoted by $\{REAL_{\pi^\rho,\mathcal{A}}(k, \bar{x}, z)\}_{k\in\mathbb{N}}$.

We are now ready to state the definition of security under concurrent general composition [23], with the additional flavor of weak security.

Definition 5 (Weak Security under Concurrent General Composition). *Let ρ be a protocol and $\mathcal{F}$ a functionality. Then, ρ* computes *$\mathcal{F}$* under concurrent general composition with weak security *if for every probabilistic polynomial-time protocol π in the $\mathcal{F}$-hybrid model that utilizes ideals calls to $\mathcal{F}$, for every probabilistic polynomial-time real-model adversary $\mathcal{A}$ for π^ρ and for every probabilistic polynomial-time distinguisher $\mathcal{D}$, there exists a probabilistic polynomial-time ideal-model adversary $\mathcal{S}$ such that for every $\bar{x}, z \in \{0,1\}^*$:*

$$\{HYBRID^{\mathcal{F}}_{\pi,\mathcal{S}}(k, \bar{x}, z)\}_{k\in\mathbb{N}} \overset{\mathcal{D}}{\equiv} \{REAL_{\pi^\rho,\mathcal{A}}(k, \bar{x}, z)\}_{k\in\mathbb{N}}.$$

If we restrict the protocols π to those that utilize at most ℓ ideal calls to $\mathcal{F}$, then ρ is said to compute $\mathcal{F}$ under ℓ-bounded concurrent general composition with weak security.

3 Review of Game-Theoretic Definitions

In this section we review basic game-theoretic definitions that we further need for establishing the equivalence between the our notion of weak security and the strong univeral implementation notion given in [17] and redefined below.

[4] This comes in contrast with the standard definition of UC ideal protocol execution, where it is not enforced that the channels between the trusted parties and the rest of the participants are ideally secure.

A *computational Bayesian game* [21] denoted as $\Gamma = (\{T_i\}_{i=1}^n, \{A_i\}_{i=1}^n, Pr, \{u_i\}_{i=1}^n)$ (also called a game with incomplete information) consists of *players* $1, \ldots, n$ where each of them makes a single move. The actions of each of the players is computed by an interactive Turing machine M_i (short PPITM). The incomplete information is captured by the fact that the *type* for each player i (i.e., its private information) is chosen externally, from a set T_i, prior to the beginning of the game. Pr is a publicly known distribution over the types. Each player has a set A_i of possible *actions* to play and individual *utility functions* u_i. All actions are played simultaneously; afterwards, every player i receives a *payoff* that is determined by applying its utility function u_i to the vector of types received in the game (i.e., *profile types*) and the actions played (i.e., *action profile*). The machine M_i is called the *strategy* for player i. The output of M_i in the joint execution of these interactive Turing machines denotes the action of player i. Because of the probabilistic strategies, the utility functions u_i actually now correspond to the expected payoffs.

Rationally behaving players aim to maximize these payoffs. In particular, if a player knew which strategies the remaining players intend to choose, he would hence pick the strategy that induces the most benefit for him. As this simultaneously holds for every player, we are looking for a so-called *Nash equilibrium*, i.e., a strategy vector where each player has no incentive to deviate from, provided that the remaining strategies do not change.

Definition 6 (Computational Nash Equilibrium). *Let Γ be a computational game, where $\Gamma = (\{T_i\}_{i=1}^n, \{A_i\}_{i=1}^n, Pr, \{u_i\}_{i=1}^n)$ and let k be the security parameter. A* strategy vector *(or* machine profile*)* consisting of PPITMs $\overrightarrow{M} = (M_1, \ldots, M_n)$ is a *computational Nash equilibrium* if for all i and any PPITM M_i' there exists a negligible function ϵ such that $u_i(k, M_i', \overrightarrow{M_{-i}}) - u_i(k, \overrightarrow{M}) \leq \epsilon(k)$ holds.

Here $u_i(k, M_i', \overrightarrow{M_{-i}})$ denotes the function u_i applied to the setting where every player $j \neq i$ sticks to its designated strategy M_j and only player i deviates by choosing the strategy M_i'. In the definition above, we call M_i a *computational best response* to $\overrightarrow{M_{-i}}$.

The definition of a game can be extended to take into account which are the utilities of a group of players participating in the prescribed protocol, or deviating from it. In the rest of the paper we denote by Z the set of players participating in such a coalition and we denote by u_Z and U_Z respectively, the utility and the expected utility for such a coalition. We also denote for example by M_Z the vector of strategies (or the PPT ITMs) that the parties in Z run (or are controlled by).

The definition of computational Nash equilibrium can be extended to the notion of *computational Nash equilibrium with immunity with respect to coalitions*. This requires that the property in the definition of computational Nash equilibrium is fulfilled for all subsets Z of players, i.e., for all Z and all PPITM M_Z' controlling the parties in Z there exists a negligible function ϵ_Z such that $U_Z(k, M_Z', \overrightarrow{M_{-Z}}) - U_i(k, \overrightarrow{M}) \leq \epsilon_Z(k)$ holds.

So far we have assumed that players communicate only among each other. We extend a computational game to a *computational game with mediator*. The mediator is modeled by an ITM denoted $\mathcal{F}$. Without loss of generality, we assume all communication passes between players and the trusted mediator (that can also forward messages among players).

Next we follow the approach from [18] to formalize the intuition that the machine profile $\overrightarrow{M} = (M_1, \ldots, M_n)$ implements a mediator $\mathcal{F}$ whenever a set of players want to truthfully provide a value (e.g., their input or type) to the mediator $\mathcal{F}$, they also want to run $\overrightarrow{M}$ using the same values. For each player i, let its type be $t_i = (x_i, z_i)$, where x_i is player's input and z_i is some auxiliary information (i.e., about the state of the world).

Let $\Lambda^{\mathcal{F}}$ denote the machine that, given the type $t = (x, z)$ sends x to the mediator $\mathcal{F}$, outputs as action the string it receives from $\mathcal{F}$ and halts. So $\Lambda^{\mathcal{F}}$ uses only input[5] x and ignores auxiliary information z. By $\overrightarrow{\Lambda^{\mathcal{F}}}$ we denote the machine profile where each player uses only $\Lambda^{\mathcal{F}}$. We ensure that whenever the players want to use mediator $\mathcal{F}$, they also want to run $\overrightarrow{M}$ if every time $\overrightarrow{\Lambda^{\mathcal{F}}}$ is a computational Nash equilibrium for the game $(G, \mathcal{F})$, then running $\overrightarrow{M}$ using the intended input is a computational Nash equilibrium as well.

Finally, we provide our definition for game theoretic protocols implementing trusted mediators. We call our notion game universal implementation. A closely related notion, called strong universal implementation, has been previously defined [17]. On an intuitive level, the main difference between the existing notion and the new notion is that for strong universal implementation, parties consider computation to be costly (i.e., time or memory used for computation may incur additional costs in the utility of the users), while our notion basically regards computation as "for free". The naive intuition suggests that game universal implementation is a weaker notion than strong universal implementation. However, as we will see in Sect. 5.1, this intuition does not hold.

Definition 7 (Game Universal Implementation). *Let $\perp_i$ be the PPT ITM ran by party i that sends no message (to the other parties or to the mediator) and outputs nothing. Let Games be a set of m-player games, $\mathcal{F}$ and $\mathcal{F}'$ be mediators and let $M_1, \ldots, M_m$ be PPT ITMs. We call $((M_1, \ldots, M_m), \mathcal{F}')$ a game universal implementation of $\mathcal{F}$ with respect to Games if for all $n \in \mathbb{N}$ and all games $G \in$ Games with input length n if $\overrightarrow{\Lambda}^{\mathcal{F}}$ is a computational Nash equilibrium in the mediated game $(G, \mathcal{F})$ with immunity with respect to coalitions, then the following two properties hold:*

- *(Preserving Equilibrium) $(M_1, \ldots, M_m)$ is a computational Nash equilibrium in the mediated machine game $(G, \mathcal{F}')$ with immunity with respect to coalitions;*

[5] As in [17], the games considered are canonical games of fixed input n. Any game where there are only finitely many possible types can be represented (by corresponding padding of the input) as a canonical game for some length n.

- *(Preserving Action Distributions) For each type profile $(t_1, \ldots, t_m)$, the output distribution induced by $\overrightarrow{\Lambda}^{\mathcal{F}}$ in $(G, \mathcal{F})$ is statistically close to the output distribution induced by $(M_1, \ldots, M_m)$ in $(G, \mathcal{F}')$;*
- *(Preservation of Best Response $\perp_i$) Additionally, for all $n \in \mathbb{N}$, all games $G \in Games$ with input length n and all $i \in \{1, \ldots, m\}$, if $\perp_i$ is a computational best response to $\overrightarrow{\Lambda}^{\mathcal{F}}_{-i}$ in $(G, \mathcal{F})$, then $\perp_i$ is a computational best response to $\overrightarrow{M}_{-i}$ in $(G, \mathcal{F}')$.*

4 Specialized Simulator UC Variants

Our main result in this section shows the separation between the notions of specialized simulator UC and 1-bit specialized simulator UC. This answers an existing open problem from [23] and furthermore clarifies the relations among different (weak) security notions. We start with a lemma on the equivalence between 1-bit specialized simulator UC (1-bit SSUC) and weak specialized simulator UC (weak SSUC).

Lemma 1 (Equivalence between 1-bit SSUC and weak SSUC). *A protocol fulfills the 1-bit specialized simulator UC security if and only if it fulfills the weak specialized simulator UC security.*

Next we separate the notions of weak specialized simulator UC and specialized simulator UC. For this we use a cryptographic tool called time-lock puzzles, originally introduced in [30].

Definition 8 (Time-lock puzzles). *A PPT algorithm $\mathcal{G}$ (problem generator) together with a PPT algorithm $\mathcal{V}$ (solution verifier) represent a* time-lock puzzle *if the following holds:*
-sufficiently hard puzzles: for every PPT algorithm B and for every $e \in \mathbb{N}$, there is some $f \in \mathbb{N}$ such that

$$\sup_{t \geq k^f, |h| \leq k^e} Pr[(q, a) \leftarrow \mathcal{G}(1^k, t) : \mathcal{V}(1^k, a, B(1^k, q, h)) = 1] \tag{1}$$

is negligible in k.
-sufficiently good solvers: there is some $m \in \mathbb{N}$ such that for every $d \in \mathbb{N}$ there is a PPT algorithm C such that

$$\min_{t \leq k^d} Pr[(q, a) \leftarrow \mathcal{G}(1^k, t); v \leftarrow C(1^k, q) : \mathcal{V}(1^k, a, v) = 1 \wedge |v| \leq k^m] \tag{2}$$

is overwhelming in k.

Intuitively, a time-lock puzzle is a cryptographic tool used for proving the computational power of a PPT machine. $\mathcal{G}(1^k, t)$ generates puzzles of hardness t and $\mathcal{V}(1^k, a, v)$ verifies that v is a valid solution as specified by a. The first requirement is that B cannot solve any puzzle of hardness t, with $t \geq k^f$, for

some f depending on B, with more than negligible probability. The algorithm B may have an auxiliary input. This ensures that even puzzles generated using hardness t chosen by B together with a trap-door like auxiliary information (of polynomial length), do not provide B with more help in solving the puzzle.

The second requirement is that for any polynomial hardness value there exist an algorithm that can solve any puzzle of that hardness. It is important that the solution for any puzzle can be expressed as a string of length bounded above by a fixed polynomial.

As promoted by [30] and later by [20], a candidate family for time-lock puzzles which is secure if the RSA assumption holds, is presented next. A puzzle of hardness t consists of the task to compute $2^{2^{t'}} \mod n$ where $t' := \min(t, 2^k)$ and $n = p_1 \cdot p_2$ is a randomly chosen Blum integer. Thus, $\mathcal{G}(1^k, t) = ((n, \min\{t, 2^k\}), (p_1, p_2, \min\{t, 2^k\}))$, where n is a k-bit Blum integer with factorization $n = p_1 \cdot p_2$, and $\mathcal{V}(1^k, (p_1, p_2, t'), v) = 1$ if and only if $(v = v_1, v_2)$ [6] and $v_1 \equiv 2^{2^{t'}} \mod n$ and $v_2 = n$. Both solving the puzzle and verifying the solution can be efficiently done if p_1 and p_2 are known. From this point further we call these puzzles the Blum integer puzzles. An important property that we use in the following is that any Blum integer puzzle has a unique solution.

Before we state and prove our main separation result in Theorem 1, we give as reminder the definition of hard-core predicates and then we state two properties related to them. These properties we use to conclude our main separation result. Due to space constraints, we add their corresponding proofs in the full version.

Definition 9 (Hard-Core Predicate). *A hard-core predicate of a collection of functions $g_{k,t} : \{0,1\}^* \to \{0,1\}^*$ is a boolean predicate $HC : \{0,1\}^* \to \{0,1\}$ such that:*

- *there exists a PPT algorithm E with $HC(x) = E(x)$, for every x;*
- *for every PPT algorithm A and for every polynomial p, there exists k_p and t_p such that for every $k > k_p$ and $t > t_p$, we have $Pr[A(1^k, t, g_{k,t}(x)) = HC(x)] < \frac{1}{2} + \frac{1}{p(k)}$.*

Now we are ready to state the two lemmas related hard-core predicates. The first result shows that from a Blum integer time-lock puzzle we can construct a one-way function and a hard-core predicate.

Lemma 2 (One-Way Function and Hard-Core Predicate from Blum Integer Time-Lock Puzzles). *Let $(\mathcal{G}, \mathcal{V})$ be a Blum integer time-lock puzzle and let t be an integer. Let $S_{k,t}$ be the set of all correctly generated solutions $v = (2^{2^t} \mod n, n)$ for puzzles q, where $q = (t, n)$ is the output of*

[6] Without loosing any security of the initial definition of time-lock puzzles [30,20], in addition to the value $2^{2^t} \mod n$, our solution for the puzzle $q = (t, n)$ contains also the value n. The full use of defining solutions in such a way, will become more clear when we define the one-way function based on time-lock puzzles: There is a one-to-one correspondence between the pair of values $(v = (2^{2^{t'}} \mod n, n), t)$ and $q = (t, n)$.

algorithm $\mathcal{G}$ when invoked with parameters 1^k and t. Then the collection of functions $\{f_{k,t} : S_{k,t} \to \{0,1\}^*\}_{(k\in\{0,1\}^*, t\in\{0,1\}^k)}$ and $\{g_{k,t} : S_{k,t} \times \{0,1\}^* \to \{0,1\}^*\}_{(k\in\{0,1\}^*, t\in\{0,1\}^k)}$ defined below are collections of one-way functions and the predicate $HC : \{0,1\}^* \to \{0,1\}^*$ defined below is a hard-core predicate for $\{g_{k,t}\}_{(k\in\{0,1\}^*, t\in\{0,1\}^k)}$.[7] *We define $f_{k,t}(2^{2^t} \bmod n, n) = (t, n)$ and for $v, r \in \{0,1\}^*$ such that $|v| = |r|$, let $g_{k,t}(v, r) = (f_{k,t}(v), r)$ and $HC(v, r) = \sum_{i=1}^{|v|} v_i \cdot r_i \bmod 2$.*

Lemma 3 (Distribution of Hard-Core Predicates). *Let k be a security parameter. Then, for any given integer t, let $g_{k,t} : D_{k,t} \to \{0,1\}^*$ be a function such that $HC : \{0,1\}^* \to \{0,1\}$ is a hard-core predicate for the collection of functions $\{g_{k,t}\}_{k\in\{0,1\}^*, t\in\{0,1\}^k}$. Let $X(k,t)$ be the distribution of $(g_{k,t}(x), HC(x))$ and let $Y(k,t)$ be the distribution of $(g_{k,t}(x), U(x))$ with x taken from the domain $D_{k,t}$ and $U(x)$ being the uniform distribution on $\{0,1\}$. Then the ensembles $\{X(k,t)\}_{(k\in\{0,1\}^*, t\in\{0,1\}^k)}$ and $\{Y(k,t)\}_{(k\in\{0,1\}^*, t\in\{0,1\}^k)}$ are computationally indistinguishable.*

Lemma 4 (Weak SSUC Does Not Imply SSUC)
Assume Blum integer time-lock puzzles exist. Then there are protocols that fulfill weak specialized simulator UC security but do not fulfill specialized simulator UC security.

Proof. Let $(\pi, \mathcal{F})$ be a pair of protocol and ideal functionality as defined below. The only input the ideal functionality $\mathcal{F}$ requires is the security parameter 1^k. Then $\mathcal{F}$ sends a message to the adversary (i.e. ideal simulator $\mathcal{S}$) asking for its computational hardness. Using the reply value t' from $\mathcal{S}$ (which is truncated by $\mathcal{F}$ to maximum k bits), the ideal functionality invokes $Gen(1^k, t') \to (q', a')$ to generate a time-lock puzzle q' of hardness t', whose solution should verify the property a'. The puzzle q' is sent to $\mathcal{S}$ which replies with v'. Finally, $\mathcal{F}$ checks whether v' verifies the property a'. In case a' does not hold, $\mathcal{F}$ stops without outputting any message to the environment. Otherwise, for every value $i \in \{1, \ldots, k\}$, $\mathcal{F}$ generates a puzzle q_i of hardness $t_i = 2^i$. Let j be such that $2^j \leq t' < 2^{j+1}$, so $j \in \{1, \ldots, k\}$.

For the puzzle q_j, $\mathcal{F}$ computes the solution v_j. $\mathcal{F}$ can efficiently compute this solution as it knows the additional information a_j. Additionally, $\mathcal{F}$ chooses r uniformly at random from $\{0,1\}^{2k}$.[8] The output of $\mathcal{F}$ to the environment is the tuple $(q_1, \ldots, q_k, r, HC(v_j, r))$, where HC is the hard-core predicate of $(\mathcal{G}, \mathcal{V})$ as given by Lemma 2.

For each hardness t', we call $P(t')$ the distribution of the view of $\mathcal{Z}$ when interacting in the ideal world.

The real world protocol π, is defined similarly to $\mathcal{F}$, the only difference is the final output: π outputs to $\mathcal{Z}$ a tuple $(q_1, \ldots, q_k, r, b)$, with r randomly chosen

[7] We would alternatively call HC the hard-core predicate for $(\mathcal{G}, \mathcal{V})$.

[8] Without loss of generality, we can assume the solution v of each puzzle q generated using the parameters 1^k and t has length $2 \cdot k$. Indeed, we can prepend with 0's to the string v such that its length reaches $2 \cdot k$. It is easy to see that after this operation, the properties stated in Lemma 2 still hold.

from $\{0,1\}^{2k}$ and b randomly chosen from $\{0,1\}$. For each hardness t used by the adversary $\mathcal{A}$ when interacting with $\mathcal{Z}$, we call $R(t)$ the distribution of the view of $\mathcal{Z}$ when interacting in the real world.

The proof has two steps. First, we show that π is as secure as $\mathcal{F}$ with respect to weak specialized simulator UC security. Let $\mathcal{D}$ be a distinguisher of hardness $t_{\mathcal{D}}$ (i.e., it can solve puzzles of hardness less or equal to $t_{\mathcal{D}}$ with overwhelming probability but it cannot solve puzzles of hardness greater than $t_{\mathcal{D}}$ with more than negligible probability) and an adversary $\mathcal{A}$ of hardness $t_{\mathcal{A}}$. Let l be the minimum value such that $2^l > \max(t_{\mathcal{D}}, t_{\mathcal{A}})$. We now require that the simulator $\mathcal{S}$ has hardness t' such that $t' \geq 2^l$. As we will see next, this is one of the constraints necessary for making the two distributions $R(t')$ and $P(t)$ indistinguishable to $\mathcal{D}$.

The intuition is that in the ideal world $\mathcal{D}$ would have to solve a puzzle with hardness larger than $t_{\mathcal{D}}$ and learn the hard-core bit for such a puzzle. According to Lemma 3, this hard-core bit is indistinguishable from a random bit, which is actually what the protocol π outputs to the environment.

More formally, let $(\mathcal{A}, \mathcal{Z}, \mathcal{D})$ be a triple of real world adversary, environment and distinguisher and let 1^k be the security parameter. Then, let e be such that the length of the messages sent by $\mathcal{Z}$ to $\mathcal{D}$ is bounded above by k^e. From (1), there exists $f_e^{\mathcal{D}}$ such that for every polynomial p there exists k_p^0 such that:

$$\sup_{t \geq k^{f_e^{\mathcal{D}}}, |h| \leq k^e} Pr[(q', a') \leftarrow \mathcal{G}(1^k, t') : \mathcal{V}(1^k, a', \mathcal{D}(1^k, q', h)) = 1] < \frac{1}{p(k)}$$

for every $k > k_p^0$.[9] Given $\mathcal{A}$, in an analogue way we define $k^{f_e^{\mathcal{A}}}$ and k_p^1. With the notation used in the description of π and $\mathcal{F}$, it now becomes clear that we can take $t_{\mathcal{D}} = k^{f_e^{\mathcal{D}}}$ and $t_{\mathcal{A}} = k^{f_e^{\mathcal{A}}}$.

We construct $\mathcal{S}$ such that there exists a negligible function ϵ and k_2 such that for every $k \geq k_2$ and for every distribution of auxiliary input z we have:

$$|(Pr(\mathcal{D}(EXEC_{\mathcal{A},\pi,\mathcal{Z}}(k,z)) = 1) - (Pr(\mathcal{D}(EXEC_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k,z)) = 1)| < \epsilon(k). \quad (3)$$

We take k_2 such that for every $k \geq k_2$, it holds that $\max(t_{\mathcal{A}}, t_{\mathcal{D}}) < 2^k$.

For a given $t_{\mathcal{A}}$ and $t_{\mathcal{D}}$ and for l defined as above, let f' be such that for sufficiently large k, $2^l \leq k^{f'} \leq 2^k$. Let $\mathcal{S}$ be the simulator of hardness $k^{f'}$ that as first reply to $\mathcal{F}$ sends $t' := k^{f'}$. According to (2), there exists m such that for $d := f'$ there exists $C_{f'}$ such that

$$Pr[(q', a') \leftarrow \mathcal{G}(1^k, k^{f'}); v' \leftarrow C_{f'}(1^k, q') : \mathcal{V}(1^k, a', v') = 1 \wedge |v'| \leq k^m]$$

is overwhelming in k. When $\mathcal{F}$ sends a puzzle q' to $\mathcal{S}$, the simulator invokes $C_{f'}$ for $(1^k, q')$ and sends to $\mathcal{F}$ the output v' of $C_{f'}$. Internally, $\mathcal{S}$ simulates the adversary $\mathcal{A}$ and emulates the messages that the adversary would receive from $\mathcal{Z}$ and π as follows: When $\mathcal{F}$ requires the value of the computational hardness from

[9] This intuitively means that $\mathcal{D}$ can solve puzzles of hardness larger than $k^{f_e^{\mathcal{D}}}$ only with negligible probability.

$\mathcal{S}$, then $\mathcal{S}$ acts as π and requires the computational hardness from simulated $\mathcal{A}$. When $\mathcal{S}$ receives t from $\mathcal{A}$, then it invokes $Gen(1^k, t)$, obtaining output (q, a) and forwards to simulated $\mathcal{A}$ the puzzle q. Moreover, any message that internal $\mathcal{A}$ wants to send to the environment, $\mathcal{S}$ forwards it to $\mathcal{Z}$. Any message for $\mathcal{A}$ coming from $\mathcal{Z}$ is immediately forwarded by $\mathcal{S}$ to the internally simulated adversary. This completes the construction of $\mathcal{S}$.

By construction, $\mathcal{S}$ solves the puzzle sent by $\mathcal{F}$ with overwhelming probability and hence the output of $\mathcal{F}$ to $\mathcal{Z}$ is $(q_1, \ldots, q_k, r, HC(v_j, r))$ with the same probability. The view of $\mathcal{Z}$ in the real world is $(1^k, t, q, v, (q_1, \ldots, q_k, r, b))$ and the view of $\mathcal{Z}$ in the ideal world[10] is $(1^k, t, q, v, (q_1, \ldots, q_k, r, HC(v_j, r)))$. By applying Lemma 3 for the distinguisher $\mathcal{D}$ and polynomial p, there exists k_p and t_p, such that for every $k > k_p$ and $t > t_p$, the advantage of $\mathcal{D}$ for distinguishing between the distributions of $((q, r), b)$ and $((q, r), HC(v, r))$ (with $\mathcal{G}(1^k, t) \leftarrow (q, a)$, v the solution to q, b the random bit and r the uniformly distributed string of k bits) is less than $\frac{1}{p(k)}$. Hence, additionally to the previous constraints on k and t', we take k such that $k > k_p$ and $\max\{t_{\mathcal{A}}, t_{\mathcal{D}}, t_p\} < 2^k$ and t' such that $t' > \max\{t_{\mathcal{A}}, t_{\mathcal{D}}, t_p\}$. With this we can conclude that the real and the ideal world views are indistinguishable to $\mathcal{D}$.

Second, we prove that π is not as secure as $\mathcal{F}$ with respect to specialized simulator UC security. Intuitively, for every hardness $t_{\mathcal{S}}$ (polynomial in the security parameter k) of a simulator machine $\mathcal{S}$, there exists a distinguisher $\mathcal{D}_{\mathcal{S}}$ such that for every $t \leq t_{\mathcal{S}}$, $\mathcal{D}_{\mathcal{S}}$ can solve puzzles of hardness t. As we will see next, $\mathcal{D}_{\mathcal{S}}$ uses this property to distinguish with non-negligible probability between the environment's output distribution in the real and in the ideal world.

Formally, let $\mathcal{A}$ be the real world adversary that can solve puzzles of hardness $t_{\mathcal{A}}$ such that when receiving its input from the environment, it replies to π with $t_{\mathcal{A}}$ and the corresponding correct solution for the puzzle received. Let $\mathcal{Z}$ be the environment that just sends the security parameter to all parties (i.e., including the adversarial parties), receives their outputs and then outputs as view the messages received from the honest parties (i.e., protocol π in the real world or $\mathcal{F}$ in the ideal world). For every simulator $\mathcal{S}$, we show that there exists a distinguisher $\mathcal{D}_{\mathcal{S}}$ and a distribution for the auxiliary input z such that:

$$\{EXEC_{\mathcal{F},\mathcal{S},\mathcal{Z}}(k, z)\}_{k\in\mathbb{N}} \overset{\mathcal{D}_{\mathcal{S}}}{\not\equiv} \{EXEC_{\pi,\mathcal{A},\mathcal{Z}}(k, z)\}_{k\in\mathbb{N}}.$$

Given $\mathcal{S}$ of hardness $t_{\mathcal{S}}$, we choose $\mathcal{D}_{\mathcal{S}}$ such that it can solve puzzles of hardness at least $t_{\mathcal{D}} = max(t_{\mathcal{S}}, t_{\mathcal{A}})$ with overwhelming probability in k. Such a $\mathcal{D}_{\mathcal{S}}$ exists according to (2). Additionally, after receiving the view of $\mathcal{Z}$, $\mathcal{D}_{\mathcal{S}}$ solves one by one each puzzle q_i included in that view that has associated hardness $t_i \leq t_{\mathcal{D}}$ and it obtains each time the corresponding correct and unique solution v_i with

[10] One may argue of course that the view of $\mathcal{Z}$ may or may not contain the values t, q, v, depending on the adversary $\mathcal{A}$. Also, additionally to the view(s) stated above, the environment could output the interaction that it has with $\mathcal{A}$ besides messages t, q, v. However, for the analysis of this proof, the views considered above are the worst case scenario that would allow a distinguisher to tell apart the two worlds.

overwhelming probability. Then $\mathcal{D}_{\mathcal{S}}$ evaluates $HC(v_i, r)$. Lets call m the last bit in the output of the honest party (i.e., $\mathcal{F}$ or π) to $\mathcal{Z}$[11]. Next, $\mathcal{D}_{\mathcal{S}}$ checks if $m \neq HC(v_i, r)$ for all i as defined above. If this holds, then $\mathcal{D}$ outputs 1, otherwise it outputs 0.

If m is part of the view of the real world, then according to the definition of π, m is a random bit in $\{0, 1\}$ so it is different than a given bit $HC(v_i, r)$ with probability $\frac{1}{2}$. This is equivalent to $\mathcal{D}_{\mathcal{S}}$ outputting 1 with probability $\frac{1}{2^{\log 2t_{\mathcal{D}}}} = \frac{1}{t_{\mathcal{D}}}$ when the view of $\mathcal{Z}$ is from the real world. Similarly, if m is part of the view of $\mathcal{Z}$ in the ideal world, then there exists at least an index i such that $HC(v_i, r)$ can be computed by $\mathcal{D}_{\mathcal{S}}$ and $m = HC(v_i, r)$; so $\mathcal{D}_{\mathcal{S}}$ outputs 1 with probability 0. This implies $\mathcal{D}_{\mathcal{S}}$ can distinguish at least with the non-negligible probability $\frac{1}{t_{\mathcal{D}}}$ [12] between the output distributions from the two worlds and this concludes the proof.

By putting together the results from Lemma 1 and from Lemma 4 we obtain:

Theorem 1 (1-bit SSUC and SSUC Not Equivalent). *Assume Blum integer time-lock puzzles exist. Then there are protocols secure with respect to 1-bit specialized simulator UC security which are not secure with respect to specialized simulator UC security.*

5 Equivalence of Security Notions

Implication relations among various security notions with respect to computational security are depicted in Fig. 1. Previously existing notions are written in a regular font, while the notions defined in this work are written in a boldface font. The continuos line arrows depict relations we prove in this paper[13]; all other relations have been previously known (the number in the square brackets denotes the reference) or can be trivially derived (i.e., denoted by letter t). Continuous line frames highlight security notions, while dotted frames highlight game-theoretic concepts. Finally, open questions are marked by question marks.

It is a well-known result that UC security and 1-bit UC security are equivalent [4]. It has been also shown [20] that specialized simulator UC security does not imply UC security. Moreover, specialized simulator UC security is equivalent to security under 1-bounded concurrent general composition [23]. It has been shown [5] that stand-alone security does not imply specialized simulator UC security.[14]

[11] Due to the definition of $\mathcal{Z}$, the string m is also a part of the output of the environment.

[12] Since $\mathcal{D}$ is a polynomial time machine, its hardness $t_{\mathcal{D}}$ is also a polynomial in the security parameter k, so the function $\frac{1}{t_{\mathcal{D}}}$ is non-negligible.

[13] A letter and number next to an arrow represent the theorem T or the lemma L or the corollary C where the respective result is shown in this paper.

[14] In order to preserve the symmetry and clarity of our picture, we have indicated that the result in [5] is that stand-alone security does not imply 1-bounded concurrent general composition. This is indeed a immediate consequence of combining the results from [5] and [23].

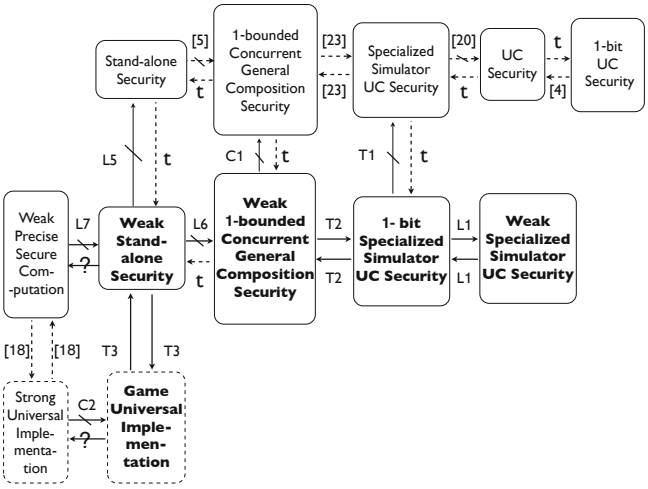


Fig. 1. Implication Relations among Computational Security Concepts

The implication in the opposite direction holds trivially. Similarly, it is trivial to see that universal composability implies specialized simulator UC security.

Our first result in this section proves that weak security under 1-bounded concurrent general composition is equivalent to 1-bit specialized simulator UC security. A similar proof technique has been used in [23], however, as detailed in the full version, our proof requires more technicalities.

Theorem 2 (Equivalence between Weak 1-bounded CGC Security and 1-bit SS UC Security). *Let ρ be a protocol and $\mathcal{F}$ an ideal functionality. We have that ρ implements $\mathcal{F}$ under weak 1-bounded concurrent general composition security, if and only if ρ securely computes $\mathcal{F}$ under 1-bit specialized simulator UC security.*

As a consequence of the above theorem, we are now also able to compare the notion of 1-bounded concurrent general composition security [23] with our variant, i.e., weak 1-bounded concurrent general composition security.

Corollary 1 (Weak 1-bounded CGC and 1-bounded CGC Not Equivalent). *Assume Blum integer time-lock puzzles exist. Then there are protocols secure with respect to weak 1-bounded concurrent general composition which are not secure with respect to 1-bounded concurrent general composition.*

We show that the approach taken in Theorem 2 is not an overkill. Indeed, there are protocols that are secure with respect to weak stand-alone security but they are not secure anymore in the standard stand-alone model (proof placed in full version). The results of Lemma 6 and Lemma 7 complete Fig. 1.

Lemma 5 (Weak Security Does Not Imply Stand-alone Security). *If Blum integer time-lock puzzles exist, then there are protocols that fulfill the weak security notion, but do not fulfill the stand-alone security notion.*

Lemma 6 (Weak Stand-alone Security Does Not Imply Weak 1-bounded CGC Security). *There exists a protocol π which is secure with respect to weak stand-alone model, but is not secure with respect to weak 1-bounded concurrent general composition security.*

As shown in Sect. 5.1, the next security result is essential for establishing the relation between the existing game-theoretic notion of strong universal implementation [18] and our notion of game universal implementation. As a preamble, we first give the intuition for weak precise secure computation. While the traditional notion of secure computation [12] requires only the worst case running time complexity of the ideal world simulator to match the running time of the real world adversary, weak precise secure computation [24] requires the complexity of the simulator to match the complexity of the real world adversary, for each arbitrary distinguisher and input.

Definition 10 (Weak Precise Secure Computation). *Let π be a protocol, $\mathcal{F}$ an ideal functionality and let $\mathcal{C}$ be the function that given a security parameter k, a polynomially bounded party Q and the view v of Q in the protocol π, it computes the complexity of Q running with k and v. We say that π is a weak precise secure computation of $\mathcal{F}$ if there exists a polynomial p such that for every real world adversary $\mathcal{A}$, for every distinguisher $\mathcal{D}$ and for every input z, there exists an ideal simulator $\mathcal{S}$, with $\mathcal{C}(k, \mathcal{S}, v) \leq p(k, \mathcal{C}(k, \mathcal{A}, \mathcal{S}(v)))$ such that :*

$$\{IDEAL(k, z, \mathcal{S}, \mathcal{F})\}_{k\in\mathbb{N}} \stackrel{D}{\equiv} \{REAL(k, z, A, \overrightarrow{M}\}_{k\in\mathbb{N}}.$$

Lemma 7 (Weak Precise Secure Computation Does Not Imply Weak Stand-alone Security). *If Blum integer time-lock puzzles exist, then there exists a protocol π which is secure with respect to weak precise secure computation, but is not secure with respect to weak stand-alone security.*

5.1 Relation between 1-Bit Specialized Simulator UC and Game Universal Implementation

In the following we prove an equivalence result between game universal implementation and our definition of weak security. A similar result exists in connection with strong universal implementation [18], but that notion considers a refined version for computational games, where the utility of the players may have strong correlations with the complexity of the computation they perform (e.g., time complexity, memory complexity, communication complexity or complexity of operations like reading inputs or copying messages). Our proof technique (see full version) is in general similar to the one used in [18].

Theorem 3 (Equivalence Between Game Universal Implementation and Weak Stand-alone Security). *Let comm be the communication mediator represented by the cryptographic notion of ideally secure channels. Let f be an m-ary function with the property that outputs the empty string to a party if and only if it had received the empty string from that party. Let $\mathcal{F}$ be a mediator that computes f* [15] *and let $\overrightarrow{M}$ be an abort-preserving computation of f*[16]*. Then $\overrightarrow{M}$ is a weak secure computation of f*[17] *with respect to statistical security if and only if $(\overrightarrow{M}, comm)$ is a game universal implementation of $\mathcal{F}$ with respect to Games, where Games is the class of games for which the utility functions of the players depend only on players types and on the output values.*

So we have shown that by restricting the class of games to those for which the computation cost for parties during protocol execution is free, our variant of universal implementation becomes equivalent to more standard notions of security (i.e., where the simulator depends only on the distinguisher and not anymore on both the distinguisher and input). Finding such equivalences was stated as an open question in [18] and to the best of our knowledge, our work makes the first step towards answering it.

One may ask if our new notion of game universal implementation is a subcase of the existing notion of strong universal implementation [18]. Using Lemma 7, Theorem 3 and the equivalence result between strong universal implementation and weak precise secure computation [18], we obtain:

Corollary 2 (Non-Equivalence of Universal Implementation Variants). *The notion of strong universal implementation does not imply the notion of game universal implementation.*

---

[15] The ideal machine profile $\overrightarrow{\Lambda^{\mathcal{F}}}$ *computes* f if for all $n \in \mathbb{N}$, all inputs $\overrightarrow{x} \in (\{0,1\}^n)^m$, the output vector of the players after an execution of $\overrightarrow{\Lambda^{\mathcal{F}}}$ on input $\overrightarrow{x}$ is identically distributed to $f(\overrightarrow{x})$.

[16] $\overrightarrow{M}$ is an abort-preserving computation of f if for all $n \in \mathbb{N}$ and for all inputs $\bar{x} \in (\{0,1\}^n)^m$, the output vector of the players after an execution of $(\perp, \overrightarrow{M_{-Z}})$ on input $\bar{x}$ is identically distributed to $f(\lambda, x_{-Z})$, where Z is a subset of all parties and λ is the empty string.

[17] We call $\overrightarrow{M}$ a weak secure computation of f if the following two properties are fulfilled:

- For all $n \in \mathbb{N}$, all inputs $\overrightarrow{x} \in (\{0,1\}^n)^m$, the output vector of the players after an execution of $\overrightarrow{M}$ on input $\overrightarrow{x}$ is distributed statistically close to $f(\overrightarrow{x})$;
- For every adversary A and for every distinguisher D, there exists a simulator $\mathcal{S}$ such that for every input z, the following relation is fulfilled :

$$\{IDEAL(k, z, \mathcal{S}, \mathcal{F})\}_{k\in\mathbb{N}} \stackrel{D}{\equiv} \{REAL(k, z, A, \overrightarrow{M}\}_{k\in\mathbb{N}}.$$

In the second property, the indistinguishability relation can be further detailed with respect to perfect, statistical or computational security.

6 Conclusions

In this work we have shown that two variants of the UC security definition where the order of quantifiers is reversed, namely 1-bit specialized simulator UC security and specialized simulator UC security are not equivalent. This comes in contrast to the well known result that UC security and 1-bit UC security are equivalent. We also show that weak security under concurrent general composition is equivalent to 1-bit specialized simulator UC. Additionally, these results combined imply weak security and stand-alone security are not equivalent.

We have also established an equivalence result between a security notion (i.e., weak stand-alone security) and a game-theoretic notion (i.e., game universal implementation). Based on the results mentioned above, as future work it is worth investigating whether one can add "composability properties" to game universal implementation in order to derive a game theoretic notion equivalent to 1-bit specialized simulator UC.

Acknowledgements. I would like to thank Dominique Unruh for pointing out to me the open problem from [23] and for insightful discussions, and to Esfandiar Mohammadi for fruitful comments on drafts of this document. I would also like to thank the anonymous reviewers for their useful feedback and comments. Their suggestions helped clarify the proof of the separation result.

References

1. Abraham, I., Dolev, D., Gonen, R., Halpern, J.: Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In: 25th Annual ACM Symposium on Principles of Distributed Computing (PODC 2006), pp. 53–62 (2006)
2. Backes, M., Pfitzmann, B., Waidner, M.: A General Composition Theorem for Secure Reactive Systems. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 336–354. Springer, Heidelberg (2004)
3. Beaver, D.: Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority. J. Cryptology 4(2) (1991)
4. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: FOCS 2001: Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science, pp. 136–145 (2001), full version on Cryptology ePrint Archive, Report 2000/067
5. Canetti, R., Fischlin, M.: Universally Composable Commitments. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 19–40. Springer, Heidelberg (2001)
6. Ciobotaru, O.: On the (non-)equivalence of uc security notions. Cryptology ePrint Archive, Report 2011/355 (2011), http://eprint.iacr.org/
7. Dodis, Y., Halevi, S., Rabin, T.: A Cryptographic Solution to a Game Theoretic Problem. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 112–130. Springer, Heidelberg (2000)
8. Dwork, C., Naor, M., Reingold, O., Stockmeyer, L.J.: Magic functions. J. ACM 50(6), 852–921 (2003)

9. Feigenbaum, J., Shenker, S.: Distributed algorithmic mechanism design: recent results and future directions. In: 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M 2002), pp. 1–13 (2002)
10. Fuchsbauer, G., Katz, J., Naccache, D.: Efficient Rational Secret Sharing in Standard Communication Networks. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 419–436. Springer, Heidelberg (2010)
11. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract). In: FOCS, pp. 174–187 (1986)
12. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: STOC, pp. 218–229 (1987)
13. Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. Journal of Cryptology 7(1), 1–32 (1994)
14. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing 18(1), 186–208 (1989)
15. Gordon, S.D., Katz, J.: Rational Secret Sharing, Revisited. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 229–241. Springer, Heidelberg (2006)
16. Halpern, J., Teague, V.: Rational secret sharing and multiparty computation: extended abstract. In: STOC 2004, pp. 623–632 (2004)
17. Halpern, J.Y., Pass, R.: A computational game-theoretic framework for cryptography (2010) (unpublished manuscript)
18. Halpern, J.Y., Pass, R.: Game theory with costly computation: Formulation and application to protocol security. In: Innovations in Computer Science (ICS 2010), pp. 120–142 (2010)
19. Hirt, M., Maurer, U.: Complete characterization of adversaries tolerable in secure multi-party computation (extended abstract). In: Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing, PODC 1997, pp. 25–34 (1997)
20. Hofheinz, D., Unruh, D.: Comparing Two Notions of Simulatability. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 86–103. Springer, Heidelberg (2005)
21. Katz, J.: Bridging Game Theory and Cryptography: Recent Results and Future Directions. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 251–272. Springer, Heidelberg (2008)
22. Kol, G., Naor, M.: Cryptography and Game Theory: Designing Protocols for Exchanging Information. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 320–339. Springer, Heidelberg (2008)
23. Lindell, Y.: General composition and universal composability in secure multi-party computation. In: FOCS, pp. 394–403 (2003)
24. Micali, S., Pass, R.: Local zero knowledge. In: STOC, pp. 306–315 (2006)
25. Micali, S., Rogaway, P.: Secure Computation (abstract). In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 392–404. Springer, Heidelberg (1992)
26. Pfitzmann, B., Schunter, M., Waidner, M.: Cryptographic security of reactive systems. Electr. Notes Theor. Comput. Sci. 32 (2000)
27. Pfitzmann, B., Waidner, M.: A general framework for formal notions of secure systems (1994), http://www.semper.org/sirene/lit

28. Pfitzmann, B., Waidner, M.: Composition and integrity preservation of secure reactive systems. In: CCS 2000: Proceedings of the 7th ACM Conference on Computer and Communications Security, pp. 245–254. ACM (2000)
29. Pfitzmann, B., Waidner, M.: A model for asynchronous reactive systems and its application to secure message transmission. In: IEEE Symposium on Security and Privacy, pp. 184–200 (2001)
30. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Tech. rep., Massachusetts Institute of Technology (1996)
31. Yao, A.C.: Theory and application of trapdoor functions. In: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS 1982), pp. 80–91 (1982)

LR-UESDE: A Continual-Leakage Resilient Encryption with Unbounded Extensible Set Delegation

Bo Yang[1] and Mingwu Zhang[1,2,3]

[1] School of Computer Science, Shaanxi Normal University
[2] College of Informatics, South China Agricultural University
[3] Institute of Mathematics for Industry, Kyushu University

Abstract. We propose a leakage-resilient unbounded extensible set delegation encryption scheme, which tolerates amount of bounded key leakage and supports unbounded delegation depth. Our scheme provides the tolerance of *key continual leakage* that can capture both memory leakage and continual leakage, which has appealing applications since there are multiple secret keys per identity set and also can periodically update and refresh the secret key. We construct the scheme in composite order bilinear groups and prove the security with dual system encryption methodology.

Keywords: Leakage-resilient encryption, statistical indistinguishability, semantic security, unbounded delegation.

1 Introduction

In an encryption system, traditionally, security are usually proven under some mathematical assumptions, such as the hardness of factoring, quadratic residue, discrete logarithm and learn with errors etc, that the secret key must be securely stored and the internal states are fully hidden for the possible attackers. Especially, the secret key corresponding to the challenged public keys or identities must not leak any bit information to the attacker. However, this situation is not necessarily true in real systems. For example, in side-channel attacks, an attacker obtains information about the internal state of a device by measuring the power consumption, computation time, and emitted sound and radiation, and then uses this information to break the security of a cryptographic primitive. In response to this, there has been a surge of interest in creating leakage-resilient cryptographic schemes [1–3,7,8,13].

In order to tolerant the key leakage, leakage-resilient cryptography models a class of leakage output by allowing the attacker is able to specify an efficiently computable leakage function and learning the partial keys or other possibly internal state from the output of function throughout the lifetime of the system. In 2009, Akavia et al. [1] first presented a notion of *memory attacks*, in which the attacker can learn arbitrary information about the challenge keys, only subject to the constraint that the total amount of output is bounded by a leakage

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 125–142, 2012.

parameter ℓ. However, if a device runs a key generation algorithm leaks one bit of information for every message that it encrypts, then the secret key will be thoroughly leaked. In order to avoid this case, we can refresh the keys before the leakage is dangerous for the decryption. Thus, in the implementations of leakage-resilient cryptographic systems, refreshing secret keys is considered a good solution in practice. If a key can be refreshed and it can also has the same decryption ability, we call *continual leakage-resilient*.

Most cryptographic schemes become insecure in the continuous leakage setting if the secret key remains static. Intuitively, if the attacker is allowed even a single bit of leakage from the state, he can eventually recover all the parts of the secret key and break security by repeatedly query the key. This leads us to the realization that in the continuous leakage setting, the key can not remain static, and must be refreshed. Brakershi et al. [7] proposed a leakage-resilient IBE in the continual leakage model that relies on selective security (weaker model than adaptive security) to allow the simulator to produce the keys incapable of decryption challenge ciphertext.

Alwen et al. [3] presented three leakage-resilient identity-based encryption schemes, which only allow bounded leakage on one secret key per decryption identity. Chow et al. [9] also constructed three leakage-resilient identity-based encryption schemes, which are based on the technique of hash proof systems [2] to obtain leakage resilience. In a hash proof system, it leads to impose on restriction that the attacker can only leak from one key for the challenge role, and no leakage on the master key is allowed. Also, the schemes in [3] and [9] do not allow a user to update/refresh his secret key during the lifetime of the system, then they can not tolerant the continual leakage.

In delegatable encryption systems [6,10,11], a use r can delegate the access right (decryption ability) from ancestors to their descendants. For example, in a hierarchical identity-based encryption scheme, both secret keys and ciphertexts are associated with ordered lists of identities. The identities if a user must fit within the hierarchy depth specified by a public parameter [6,10,11,14,16], that is, the size of public parameter will grow linearly with the maximum of the hierarchy. Similarly, the keys and possible ciphertexts are growing linearly the depth of the hierarchy. This will be inflexible in practical applications: it is impossible to add new level to the hierarchy once the maximum depth is previously fixed. In order to support possible hierarchy, we can set an enough large hierarchy, however, it will aggravate the storage burden and increase the communication overhead since the user hierarchy is less than the maximum depth.

It is a challenging issue to remove the restriction of maximum hierarchy depth. Lewko and Waters [15] first constructed unbounded HIBE and ABE, in which the system public parameters are a constant number of group elements that eliminate the need to pre-decide maximum hierarchy depth. The main idea is to employ a secret-sharing mechanism to split the master key into shares.

Traditionally, the security in hierarchical encryption systems rely on the partitioning technique for proof of security in the adaptive security model. More

precisely, in the partition model, the user identities will be partitioned into two subsets: one contains identities for which challenge ciphertext can be generated and the other of the secret keys can be extracted by the attacker. In delegatable encryption systems, however, this restricts the attacker from querying keys for any identity. To overcome this constraint, Waters [18] proposed a notion of dual system encryption to obtain adaptive security. In dual system encryption, it guarantees that the challenge identity is different from the identities for which the keys are extracted. If there is a collision then the attacker gains no advantage in breaking the security of the system. Later, Lewko and Waters [14] presented more efficient schemes whose security is based on static assumptions in the composite order bilinear groups. Lewko et al. [13] constructed the leakage-resilient IBE, HIBE and ABE in composite order groups, which are based on the subgroup decision assumptions. Due to the property of the subgroup orthogonality, it provides a possible mechanism to implement the statistical indistinguishability of a function's output in multiple spaces. Although in [9], Alwen et al. presented a leakage-resilient scheme in composite order bilinear groups, however, their scheme still relies on the technique of hash proof system to obtain leakage resilience and the dual system encryption is only for full security proof.

Our Result. We design a leakage-resilient encryption that supports unbounded extensible set delegation depth, in which we employ a secret-sharing approach to split the master key into multiple shares in key components corresponding to the elements in the set. User identities are specified as sets whose elements are in $\mathbb{Z}_N$. The decryption key d_χ can decrypt a ciphertext $\sigma_{\chi'}$ when the encryption policy χ' and decryption role χ has $\chi \sqsubseteq \chi'$. Moreover, role χ_1 can perform a delegation to another role of extensible set χ_2, i.e., $\chi_1 \sqsubseteq \chi_2$. In order to implement unbounded delegation, we do not set a maximum depth (set size) in the setup algorithm. Our scheme is a general extension of fuzzy identity encryption [16] supporting unbounded delegation ability and leakage resilience. Compared with related literatures that impose a pre-determined maximum depth [15,18–20], the public parameters of our scheme has the smallest size.

In the security of our scheme, we allow the attacker to learn arbitrary functions of secret keys, as long as the total amount leakage bits per set are bounded by parameter ℓ. More precisely, we allow the attacker to handle key extraction query and leakage query in the dual system encryption mechanism and provide leakage information from multiple keys of the same ciphertext, which eliminates the need for a separate technique to achieve leakage resilience in [9]. As we provide an implicit refresh procedure for a key, the scheme supports continual resilient to secret keys. Compared with previous schemes that only allow either bounded leakage on one secret key per decryption role [9], our scheme has stronger leakage resilience with no sacrifice efficiency.

Proposed scheme is constructed with dual system encryption methodology in composite order bilinear groups, which tolerates the secret key leakage. The scheme achieves the adaptively semantic security in the presence of bounded key leakage model. The security proofs rely on the static mathematical assumptions that are based on the static (bilinear) subgroup decisional problems [13–15].

Roadmap. The remainder paper is organized as follows: Some preliminaries are provided in Section 2; The model of leakage-resilient encryption with unbounded extensible set delegation and its security model are given in Section 3; The construction is presented in Section 4; The consistency is analyzed in Section 5; The security analysis is provided in Section 6, and the performances of leakage bound and leakage fraction are discussed in Section 7; Finally, the conclusion is drawn in Section 8.

2 Preliminaries

Let N be a positive integer and $\mathbb{Z}_N = \{1, \cdots, N-1\}$. We let x be chosen uniformly from $\mathbb{Z}_N$ denote by $x \xleftarrow{\$} \mathbb{Z}_N$. For a vector $\boldsymbol{x} = (x_1, \cdots, \boldsymbol{x}_d) \in \mathbb{Z}_N^d$, we use $g^{\boldsymbol{x}}$ to denote the vector of group elements $g^{\boldsymbol{x}} = (g^{\boldsymbol{x}_1}, \cdots, g^{\boldsymbol{x}_d}) \in \mathbb{G}_N^d$. We denote $\langle \boldsymbol{\rho}, \boldsymbol{\sigma} \rangle$ as the inner product of vectors $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}$, i.e., $\langle \boldsymbol{\rho}, \boldsymbol{\sigma} \rangle = \sum_i \rho_i \sigma_i$. Similarly, $\langle g^{\boldsymbol{\rho}}, g^{\boldsymbol{\sigma}} \rangle = g^{\sum_i \rho_i \sigma_i}$. Let χ be a set, the size of χ is denoted by $|\chi|$. We use $negl(\cdot)$ to denote a negligible function.

2.1 Arbitrary Function Leakage Subspaces

In order to guarantee that, in case of obtaining partial leakage information of the secret key (no more than ℓ-bit), the attacker has the same decryption ability with non-leakage information, we use a statistical indistinguishable theorem to describe the bounded leakage. Let $Dist(X_1, X_2)$ be the statistical distance of two random variables X_1 and X_2. In our system, the leakage resilience relies on the following Lemma from [7], which is also proved in [4].

Lemma 1. *[7] Let p_2 be a large prime, $m, l, d \in \mathbb{N}$, $2d \leq l \leq m$. Let $X_1 \xleftarrow{\$} \mathbb{Z}_{p_2}^{m \times l}$ and $X_2 \xleftarrow{\$} \mathbb{Z}_{p_2}^{m \times d}$, and let $T \xleftarrow{\$} Rank_d(\mathbb{Z}_{p_2}^{l \times d})$, where $Rank_d(\mathbb{Z}_{p_2}^{l \times d})$ denotes the set of $l \times d$ matrices of tank d with entries in $\mathbb{Z}_{p_2}$. For any leakage function $f : \mathbb{Z}_{p_2}^{m \times d} \to W$, there exists*

$$Dist((X_1, f(X_1 T)), (X_1, f(X_2))) \leq \epsilon(\cdot), \quad |W| \leq 4(1 - \frac{1}{p_2}) \cdot p_2^{l-2d+1} \cdot \epsilon(\cdot)^2$$

In particular, if the leakage $f(X_1 T)$ reveals bounded information X_1, then $(X_1, f(X_1 T))$ and $(X_1, f(X_2))$ are statistically close. In the latter pair, X_2 is a random vector and the leakage function reveals nothing about the subspace X_1.

Corollary 1. *Let p_2 be a prime and $m \geq 3$ be an integer. Let $\Delta, \boldsymbol{\mu} \xleftarrow{\$} \mathbb{Z}_{p_2}^m$ and $\boldsymbol{\mu}'$ be selected uniformly randomly from the set of vector in $\mathbb{Z}_{p_2}^m$ which are orthogonal to Δ under the dot product modulo p_2. For any function $f : \mathbb{Z}_{p_2}^m \to W$, there exists*

$$Dist((\Delta, f(\boldsymbol{\mu})), (\Delta, f(\boldsymbol{\mu}'))) \leq \epsilon(\cdot), \ |W| \leq 4 p_2^{m-3}(p_2 - 1) \cdot \epsilon(\cdot)^2$$

The corollary 1 allows us to set leakage bound $\ell = 2 + (n - 1 - 2c) \log_2 p_2$ for our construction: set $n + 1 = m$ and c ($c \geq 2$) is any fixed positive constant, $\epsilon(\cdot) = p_2^{-c}$ is negligible.

2.2 Bilinear Subgroups of Composite Order

Definition 1. ***Composite order groups*** *Let $\mathcal{G}_{cp}$ be a group generator, that takes a security parameter τ as input where $\tau \in \mathbb{N}$, outputs a description of bilinear group $G = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_t, \hat{e})$, where p_1, p_2, p_3 are distinct prime τ-bit primes, $\mathbb{G}$ and $\mathbb{G}_t$ are cyclic groups of order N, and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_t$ is a bilinear map such that $\forall u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_N$, $\hat{e}(u^a, v^b) = \hat{e}(u^b, v^a) = \hat{e}(u, v^b)^a = \hat{e}(u, v)^{ab}$. A bilinear map is admissible if $\hat{e}$ can be computed efficiently.*

We set the order of the product of three primes. That is, $N = p_1p_2p_3$. Let $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}$ denote the subgroups of $\mathbb{G}$ having order p_1, p_2, p_3. Obviously, for $a, b, c \in \{1, p_1, p_2, p_3\}$, assume that $\mathbb{G}_1 = 1_{\mathbb{G}}$, we indicate that $\mathbb{G}_{abc} = \mathbb{G}_a \times \mathbb{G}_b \times \mathbb{G}_c$, where the subgroups of $\mathbb{G}$ having order π is denoted by $\mathbb{G}_\pi$. $\mathbb{G}_\pi$ can be written uniquely as the product $g_1^{d_1} g_2^{d_2} g_3^{d_3}$ where g_1, g_2, g_3 are the generators of subgroups $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}$, respectively. Suppose that $h_1 \in \mathbb{G}_{\pi_1}, h_2 \in \mathbb{G}_{\pi_2}$, it has $\hat{e}(h_1, h_2) = 1$ if $gcd(\pi_1\pi_2 | N^2, N) = N$. This is called the orthogonality of subgroups. In our system, the orthogonality of subgroups provides an ability to construct the dual system encryption, i.e., normal keys/ciphertexts and semi-functional keys/ciphertexts.

2.3 Complexity Assumptions

To prove the security of our scheme, we use the following well-established assumptions in composite order bilinear groups. We first define the variable $G = (N, \mathbb{G}, \mathbb{G}_t, \hat{e}(,\cdot,)) \leftarrow \mathcal{G}_{\mathsf{cp}}(\tau)$ that is generated by a composite order bilinear group generating algorithm. In the security assumptions, the attacker outputs a guess $\psi \in \{0, 1\}$ and wins the assumption if the advantage in distinguishing between T_ψ and $T_{1-\psi}$ is non-negligible.

Assumption 1. *Given a random instance derived from composite order group generator $\mathcal{G}_{cp}$, we define the distribution of Assumption 1 as follows:*

$$\left[g \xleftarrow{\$} \mathbb{G}_{p_1},\ D_1 = (G, g), T_1 \xleftarrow{\$} G_{p_1p_2},\ T_2 \xleftarrow{\$} \mathbb{G}_{p_1} \right]$$

We say that Assumption 1 holds if for all polynomial-time algorithm $\mathcal{A}$ the advantage is negligible, where the advantage of $\mathcal{A}$ is defined as

$$Adv^1_{\mathcal{G}_{\mathsf{cp}},\mathcal{A}}(\tau) := |Pr[\mathcal{A}(D_1, T_1) = 1] - Pr[\mathcal{A}(D_1, T_2)] = 1|$$

Assumption 2. *Given a random instance derived from composite order group generator $\mathcal{G}_{cp}$, we define the distribution of Assumption 2 as follows:*

$$\left[\begin{array}{l} g, X_1 \xleftarrow{\$} \mathbb{G}_{p_1}, X_2, Y_2 \xleftarrow{\$} \mathbb{G}_{p_2}, g_3, Y_3 \xleftarrow{\$} \mathbb{G}_{p_3} \\ D_2 = (G, g, g_3, X_1X_2, Y_2Y_3) \\ T_1 \xleftarrow{\$} \mathbb{G}_{p_1p_3},\ T_2 \xleftarrow{\$} \mathbb{G}_{p_1p_2p_3} \end{array} \right]$$

We say that Assumption 2 holds if for all polynomial-time algorithm $\mathcal{A}$ the advantage is negligible, where the advantage of $\mathcal{A}$ is defined as

$$Adv^{2}_{\mathcal{G}_{\mathsf{cp}},\mathcal{A}}(\tau) := |Pr[\mathcal{A}(D_2, T_1) = 1] - Pr[\mathcal{A}(D_2, T_2)] = 1|$$

Assumption 3. *Given a random instance derived from composite order group generator $\mathcal{G}_{\mathsf{cp}}$, we define the distribution of Assumption 3 as follows:*

$$\left[\begin{array}{l} g \xleftarrow{\$} \mathbb{G}_{p_1}, g_2, X_2, Y_2 \xleftarrow{\$} \mathbb{G}_{p_2}, g_3 \xleftarrow{\$} \mathbb{G}_{p_3}, \alpha, s \xleftarrow{\$} \mathbb{Z}_N \\ D_3 = (G, g, g_2, g_3, g^{\alpha} X_2, g^{s} Y_2) \\ T_1 = \hat{e}(g,g)^{\alpha s},\ T_2 \xleftarrow{\$} \mathbb{G}_t \end{array}\right]$$

We say that Assumption 3 holds if for all polynomial-time algorithm $\mathcal{A}$ the advantage is negligible, where the advantage of $\mathcal{A}$ is defined as

$$Adv^{3}_{\mathcal{G}_{\mathsf{cp}},\mathcal{A}}(\tau) := |Pr[\mathcal{A}(D_3, T_1) = 1] - Pr[\mathcal{A}(D_3, T_2)] = 1|$$

3 Leakage-Resilient Unbounded Extensible Set Delegation Encryption

3.1 The Model

A leakage-resilient unbounded extensible set delegation encryption scheme (LR-UESDE) $\prod := (\mathsf{Init}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del})$ is informally defined as follows:

$(\mathtt{PP}, \mathtt{SK}) \leftarrow \mathbf{Init}(\tau, \ell)$ The system setup algorithm takes the security parameter τ and the secret key leakage bound ℓ as inputs, and outputs public key $\mathtt{PP}$ and master secret key $\mathtt{SK}$.

$d_{\chi} \leftarrow \mathbf{Ext}(\mathtt{PP}, \mathtt{SK}, \chi)$ The key generation takes system public key $\mathtt{PP}$, master key $\mathtt{SK}$, and a set $\chi = \{S_1, S_2, \cdots, S_j\}$ as inputs, and outputs a secret key d_{χ}.

$\sigma_{\chi} \leftarrow \mathbf{Enc}(\mathtt{PP}, \chi, M)$ The encryption algorithm takes a message M, a receiver set χ, and the system public key $\mathtt{PP}$ as inputs, and outputs a ciphertext σ_{χ}.

$M \leftarrow \mathbf{Dec}(\mathtt{PP}, d_{\chi}, \sigma_{\chi'})$ The decryption algorithm takes a ciphertext $\sigma_{\chi'}$, a secret key d_{χ}, and the system public key $\mathtt{PP}$ as inputs, and outputs the message M if $\chi \sqsubseteq \chi'$.

$d_{\chi \cup \chi'} \leftarrow \mathbf{Del}(\mathtt{PP}, \chi, d_{\chi}, \chi')$ The delegation algorithm takes a secret key d_{χ} of χ, a set χ', and the public key $\mathtt{PP}$ as inputs, and outputs a secret key $d_{\chi \cup \chi'}$ for the union set $\chi \cup \chi'$.

The consistency of LR-UESDE holds: For all correctly generated $\mathtt{PP}$ and d_{χ}; Let f_i and f'_i be any polynomial-time computable functions. $\sigma_{\chi} \leftarrow \mathsf{Enc}(\mathtt{PP}, \chi, M)$ and $M' \leftarrow \mathsf{Dec}(\mathtt{PP}, d_{\chi}, \sigma_{\chi'})$. If $\chi \sqsubseteq \chi'$, there holds that $M = M'$. Otherwise, $M \neq M'$ except for negligible probability, i.e.,

$$Pr\left[\begin{array}{l}(\mathsf{PP},\mathsf{SK}) \leftarrow \mathsf{Init}(\tau,\ell)\\ \chi \sqsubseteq \chi', f_i, f_i' : d \rightarrow \{0,1\}^*\\ \sigma_{\chi'} \leftarrow \mathsf{Enc}(\mathsf{PP},\chi',M)\\ d_\chi \leftarrow \mathsf{Ext}(\mathsf{PP},\mathsf{SK},\chi)\\ d_{\chi'} \leftarrow \mathsf{Del}(\mathsf{PP},\chi,d_\chi,\chi'-\chi)\\ \sum_i f_i(d_\chi) \leq \ell, \sum_i f_i'(d_{\chi'}) \leq \ell\\ M' \leftarrow \mathsf{Dec}(\mathsf{PP},d_\chi,\sigma_{\chi'})\\ M'' \leftarrow \mathsf{Dec}(\mathsf{PP},d_{\chi'},\sigma_{\chi'})\\ M' = M,\ M'' = M\end{array}\right] \geq 1-\epsilon(\tau)$$

3.2 Leakage-Resilient Semantic Security

We model our leakage-resilient encryption scheme with unbounded extensible set delegation to be semantically secure under the key leakage situation. We note that the security still holds in the sense that the attacker can specify any efficiently computable functions and learns the function outputs. We also allow the attacker to query the leakage oracle about the user's secret key.

In our security experiment, the attacker has access to the queries of key extraction, delegation, leak and reveal. In order to model the leakage oracle, we allow the attacker to get access to the leakage oracle on the secret key with only the constraint that the amount outputs of the leakage can not get more than ℓ bits per secret key.

Definition 2. *A leakage oracle $\mathcal{O}_{\mathsf{Leak}}$ is parameterized by secret key d_χ and leakage parameter ℓ. A query to the oracle $\mathcal{O}_{\mathsf{Leak}}$ is launched by a leakage function $f : d \rightarrow \{0,1\}^*$. The oracle computes and outputs $f(d_\chi)$. The leakage oracle responds at most amount of ℓ bits of a key d_χ, and ignores all queries afterwards.*

In order to simulate and elides the query for the attacker's behavior, we define four oracles: $\mathcal{O}_{\mathsf{Ext}}, \mathcal{O}_{\mathsf{Del}}, \mathcal{O}_{\mathsf{Leak}}, \mathcal{O}_{\mathsf{Rvl}}$, to answer the attacker's query. These oracles are associated with a handle h. The attacker issues a $\mathcal{O}_{\mathsf{Ext}}$ oracle or a $\mathcal{O}_{\mathsf{Del}}$ oracle to generate an entire secret key and stores it into the key queue. The attacker issues a $\mathcal{O}_{\mathsf{Leak}}$ oracle query to obtain the leakage information on a secret key, and issues a $\mathcal{O}_{\mathsf{Rvl}}$ oracle to get access to an entire secret key.

We define the security experiment $Exp^{LR\text{-}CPA}_{\prod,\mathcal{A}}$ for the LR-UESDE scheme under the key leakage situation. In the security experiment, any attacker must not distinguish two messages in a ciphertext after performing a bounded of allowable queries as previous defined. The security experiment $Exp^{LR\text{-}CPA}_{\prod,\mathcal{A}}$ between a challenger $\mathcal{C}$ and an attacker $\mathcal{A}$ is formally described as follows:

Setup. The challenger $\mathcal{C}$ runs Init algorithm to create the system public key PP and the master key SK, and sends PP to attacker $\mathcal{A}$. Also, $\mathcal{C}$ creates two initial empty sets: $\mathcal{Z} = \{Revealed\text{-}Set\}$ and $\mathcal{H} = \{h, \chi, d_\chi, LeakBits\}$ (handle-set-key-leaked bits), where $\mathcal{Z}$ records all identities that the secret keys have been revealed from $\mathcal{O}_{\mathsf{Rvl}}$, and $\mathcal{H}$ records the leakage bits of secret keys leaked from $\mathcal{O}_{\mathsf{Leak}}$. We use a handle h to get access to $\mathcal{H}$.

Query 1. $\mathcal{A}$ can make the following queries with an adaptive manner.

- Key extraction oracle ($\mathcal{O}_{\mathsf{Ext}}(h, \chi)$): The challenger $\mathcal{C}$ searches $\mathcal{H}$ to get the record. Otherwise, if not found (the secret key has not been created), $\mathcal{C}$ makes call to $\mathcal{O}_{\mathsf{Ext}}$ to generate d_χ and adds a new record $(h, \chi, d_\chi, 0)$ into queue $\mathcal{H}$.
- Key delegation oracle ($\mathcal{O}_{\mathsf{Del}}(h, \chi, \chi')$): At first $\mathcal{C}$ searches the secret key d_χ in $\mathcal{H}$. $\mathcal{C}$ responds the query as follows:
  1. Case the record found in $\mathcal{H}$ and the request is key update ($\chi' = \phi$): $\mathcal{C}$ performs the refresh procedure and sets $LeakBits = 0$.
  2. Case the record found and $\chi' \neq \phi$: Make call to $\mathcal{O}_{\mathsf{Del}}$ to produce a delegation key $d_{\chi \cup \chi'}$, and add a record $(h, \chi \cup \chi', d_{\chi \cup \chi'}, 0)$ into $\mathcal{H}$.
  3. Case the record not found: Make call to $\mathcal{O}_{\mathsf{Ext}}$ to generate d_χ and add a new record $(h, \chi, d_\chi, 0)$ into $\mathcal{H}$. Then, make call to $\mathcal{O}_{\mathsf{Del}}$ to produce a delegation key $d_{\chi \cup \chi'}$, and add a record $(h, \chi \cup \chi', d_{\chi \cup \chi'}, 0)$ into $\mathcal{H}$.
- Leakage oracle ($\mathcal{O}_{\mathsf{Leak}}(h, f)$): $\mathcal{A}$ request a key leakage query with a polynomial-time function $f : d \to \{0,1\}^*$. $\mathcal{C}$ searches $\mathcal{H}$ to find the record, and answers with $f(d_\chi)$ if $LeakBits + f(d_\chi) \leq \ell$, and updates $LeakBits$ with $LeakBits + f(d_\chi)$; Outputs 0 otherwise.
- Reveal oracle ($\mathcal{O}_{\mathsf{Rvl}}(h)$): $\mathcal{A}$ provides a handle h and uses $\mathcal{O}_{\mathsf{Rvl}}(h)$ oracle to obtain an entire secret key. $\mathcal{C}$ searches $\mathcal{H}$ to find the record of index h, adds the corresponding χ into list $\mathcal{Z}$ and returns d_χ to $\mathcal{A}$.

Challenge. $\mathcal{A}$ chooses two equal plaintexts M_0, M_1 and a challenge set χ^* with the restriction that $\chi^* \notin \mathcal{Z}$. $\mathcal{C}$ tosses a random coin $\psi \in \{0,1\}$, creates the challenge ciphertext $\sigma_{\chi^*} \leftarrow \mathsf{Enc}(\mathsf{PP}, \chi^*, M_\psi)$ and sends σ_{χ^*} to $\mathcal{A}$.

Query 2. This stage is the same as Stage-I, with the added constraint that $\mathcal{O}_{\mathsf{Ext}}$, $\mathcal{O}_{\mathsf{Del}}$ and $\mathcal{O}_{\mathsf{Rvl}}$ queries that involved secret keys of sets are not the subset of χ^*.

Output. Attacker returns ψ' as the guess of ψ, and wins if $\psi' = \psi$.

$\mathcal{A}$'s advantage in experiment $Exp^{LR\text{-}CPA}_{\prod,\mathcal{A}}$ is defined as

$$Adv^{LR\text{-}CPA}_{\prod,\mathcal{A}}(\tau) := |Pr[(\psi = \psi') - \frac{1}{2}]|$$

Definition 3. ***Leakage-resilient Semantic Security*** *Suppose that the attacker has at most q queries for the keys, the LR-UESDE scheme is said to be (q, ℓ) semantically secure with leakage bound ℓ if any probabilistic polynomial-time attacker $\mathcal{A}$ achieves at most a negligible advantage $Adv^{LR\text{-}CPA}_{\prod,\mathcal{A}}$ in experiment $Exp^{LR\text{-}CPA}_{\prod,\mathcal{A}}$.*

Definition 4. ***Continual-Leakage Resilience*** *If a leakage-resilient encryption scheme is implicitly equipped with a refresh algorithm that takes in a secret key and outputs a new and re-randomized key from the same distribution generated by a fresh call to Ext or Del algorithms, then the scheme is called continual leakage resilient.*

4 Our Construction

In this section, we present the concrete construction of LR-UESDE scheme. In our scheme, we use a composite order bilinear group of order $N = p_1p_2p_3$, where p_1, p_2 and p_3 are distinct primes. We assume that the set elements are defined in $\mathbb{Z}_N \backslash \{0\}$. In order to simplify the design, we will assume that the extensible set has not same element to the previous set, i.e., $\chi' = \chi_1 \cup \chi_2$ is an extensible set from χ_1 such that $\chi_1 \cap \chi_2 = \phi$. We also assume that the elements in the set χ are ordered so that we can guarantee that two equal sets have the same elements sequence.

4.1 Intuition

The main idea of our construction is to employ a secret-sharing approach across the size of the set of secret key. A secret key involves a sharing of the master key α as a sum of exponents, where each piece of the sum is additionally blinded by a random term which is unique to the piece. To successfully decrypt a ciphertext, a user must effectively unblind each share, which can only be accomplished by a user with all j elements in the set which matches the ciphertext vector in all of the components through j. Also, our construction is derived from the dual system encryption technique, which allows us to tolerant multiple keys per user set. We implement the leakage resilience by expanding the semi-functional space to multiple dimensional.

In our scheme, a secret key of set $\chi = \{S_1, \ldots, S_j\}$ has the form

$$
\begin{aligned}
d_{\chi=\{S_1,\ldots,S_j\}} &= (\boldsymbol{k}_\rho, \boldsymbol{k}_w, \boldsymbol{k}_y, \boldsymbol{k}_v, \boldsymbol{k}_r) \\
&= \left\langle \begin{pmatrix} g^{\rho_1} \\ g^{\rho_2} \\ \vdots \\ g^{\rho_n} \end{pmatrix}^\top, \begin{pmatrix} g^{\lambda_1+\langle \rho'_1, \boldsymbol{\sigma}\rangle} w^{y_1} \\ g^{\lambda_2+\langle \rho'_2, \boldsymbol{\sigma}\rangle} w^{y_2} \\ \vdots \\ g^{\lambda_j+\langle \rho'_j, \boldsymbol{\sigma}\rangle} w^{y_j} \end{pmatrix}^\top, \begin{pmatrix} g^{y_1} \\ g^{y_2} \\ \vdots \\ g^{y_j} \end{pmatrix}^\top, \begin{pmatrix} v^{y_1}(u^{S_i}h)^{r_1} \\ v^{y_2}(u^{S_i}h)^{r_2} \\ \vdots \\ v^{y_j}(u^{S_i}h)^{r_j} \end{pmatrix}^\top, \begin{pmatrix} g^{r_1} \\ g^{r_2} \\ \vdots \\ g^{r_j} \end{pmatrix}^\top \right\rangle \times g_3^{\boldsymbol{z}}
\end{aligned} \tag{1}
$$

where $g_3^{\boldsymbol{z}} \in \mathbb{G}_{p_3}^{n+4j}$, which means that all elements in d_χ are in $\mathbb{G}_{p_1p_3}$. We note that the secret key d_χ implicitly holds $\sum_{i\in\chi} \lambda_i = \alpha$ and $\sum_{i\in\chi} \rho'_i = \sum_{i\in[1,n]} \rho$. n ($n \geq 2$) is a positive integer that determines the leakage-resilient intensity. The larger n causes a better leakage fraction being tolerated, and the smaller n yields a shorter key and ciphertext.

A ciphertext σ_χ has the following form:

$$
\begin{aligned}
\sigma_\chi &= (c_m, \boldsymbol{c}_\sigma, c_s, \boldsymbol{c}_w, \boldsymbol{c}_t, \boldsymbol{c}_u) \\
&= \left(M \cdot \hat{e}(g,g)^{\alpha s}, \begin{pmatrix} g^{\sigma_1} \\ g^{\sigma_2} \\ \vdots \\ g^{\sigma_n} \end{pmatrix}^\top, g^s, \begin{pmatrix} w^s v^{t_1} \\ w^s v^{t_2} \\ \vdots \\ w^s v^{t_j} \end{pmatrix}^\top, \begin{pmatrix} g^{t_1} \\ g^{t_2} \\ \vdots \\ g^{t_j} \end{pmatrix}^\top, \begin{pmatrix} (u^{S_i}h)^{t_1} \\ (u^{S_i}h)^{t_2} \\ \vdots \\ (u^{S_i}h)^{t_j} \end{pmatrix}^\top \right)
\end{aligned} \tag{2}
$$

4.2 Concrete Construction

Init(τ, ℓ) On input the system security parameter τ and the key leakage bound ℓ, this algorithm generate system public key $\mathtt{PP}$ and master key $\mathtt{SK}$ as follows:

1. Take a system security parameter τ as input to generate a description of bilinear composite-order group $G = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_t, \hat{e})$, where p_1, p_2, p_3 are distinct primes of size τ_1, τ_2, τ_3-bit[1], and for $i = 1, 2, 3$, $2^{\tau_i - 1} \leq p_i \leq 2^{\tau_i}$ and $\tau_i = \mathsf{poly}(\tau)$;
2. At random pick $\alpha \xleftarrow{\$} \mathbb{Z}_N$, $g, u, h, v, w, g^{\boldsymbol{\sigma}} \xleftarrow{\$} \mathbb{G}_{p_1}^{5+n}$ where $|\boldsymbol{\sigma}| = n$ $(n \geq 2)$, and $g_3 \xleftarrow{\$} \mathbb{G}_{p_3}$;
3. Set the system public key as $\mathtt{PP} := (G, \ell, g, g_3, u, h, v, w, g^{\boldsymbol{\sigma}}, \hat{e}(g, g)^{\alpha})$, where $\ell = \ell(\tau)$ is the maximum leakage bound of secret key. We define the message space in $\mathbb{G}_t$, the ciphertext space in $\mathbb{G}_t \times \mathbb{G}^{4j+n+1}$, and the set space in $\mathbb{Z}_N^j$, where j is the value of the size of set χ.
4. Keep the master key $\mathtt{SK} := g^{\alpha}$.

Ext$(\mathtt{PP}, \mathtt{SK}, \chi)$ The key extraction algorithm creates a secret key d_{χ} of set $\chi = \{S_1, \cdots, S_j\}$ as follows:

1. Randomly pick $g^{\lambda_1}, \cdots, g^{\lambda_j} \in \mathbb{G}_{p_1}$ subject to the constraint that $\prod_{i \in [1,j]} g^{\lambda_i} = g^{\alpha}$. Note that g^{λ_i} $(1 \leq i \leq j)$ can be chosen as follows: at first uniformly choose $\lambda_1, \cdots, \lambda_{j-1} \in \mathbb{Z}_N$, compute $g^{\lambda_1}, \cdots, g^{\lambda_{j-1}}$, and $g^{\lambda_j} = g^{\alpha} / \prod_{i \in [1,j-1]} g^{\lambda_i}$;
2. At random choose a vector $\boldsymbol{\rho} \xleftarrow{\$} \mathbb{Z}_N^n$ and $\boldsymbol{\rho}'_1, \boldsymbol{\rho}'_2, \cdots, \boldsymbol{\rho}'_j \xleftarrow{\$} \mathbb{Z}_N^n$ with the constraint that $\sum_{i \in [1,j]} \boldsymbol{\rho}'_i = \boldsymbol{\rho}$;
3. Choose $\boldsymbol{r}, \boldsymbol{y} \xleftarrow{\$} \mathbb{Z}_N^j$, $\boldsymbol{z} \xleftarrow{\$} \mathbb{Z}_N^{n+4j}$ randomly, and output the secret key d_{χ}

$$\begin{aligned} d_{\chi} &:= (\boldsymbol{k}_{\rho}, \boldsymbol{k}_w, \boldsymbol{k}_y, \boldsymbol{k}_v, \boldsymbol{k}_r) \\ &= \left\langle g^{\boldsymbol{\rho}}, g^{\boldsymbol{\lambda} + \langle \boldsymbol{\rho}', \boldsymbol{\sigma} \rangle} w^{\boldsymbol{y}}, g^{\boldsymbol{y}}, v^{\boldsymbol{y}} (u^{\boldsymbol{\chi}} h)^{\boldsymbol{r}}, g^{\boldsymbol{r}} \right\rangle \times g_3^{\boldsymbol{z}} \end{aligned} \tag{3}$$

   Notice that all the components of d_{χ} are the elements of subgroup $\mathbb{G}_{p_1p_3}$.

Del$(\mathtt{PP}, \chi, d_{\chi}, \chi')$ If $\chi' = \phi$, this algorithm only performs the secret key re-randomization procedure (step 2) to update the secret key d_{χ}. Otherwise, the algorithm does the following two steps (derivation procedure and refreshness procedure) to produce a derivation key $d_{\chi \cup \chi'}$, which means that a user χ of secret key d_{χ} generates a secret key for $\chi \cup \chi' = \{S_1, \cdots, S_{|\chi \cup \chi'|}\}$. Let $d_{\chi} = (\boldsymbol{k}_{\rho}, \boldsymbol{k}_w, \boldsymbol{k}_y, \boldsymbol{k}_v, \boldsymbol{k}_r)$.

[1] τ_1, τ_2, τ_3 are depended on the security parameter τ. We can choose different length and get a different leakage fraction and leakage-resilient probability. In Section 7, we will discuss the relationship among them.

1. **Derivation procedure**. Let $j = |\chi|$ and $k = |\chi \cup \chi'|$. At random choose $\triangle \boldsymbol{\lambda}' \xleftarrow{\$} \mathbb{Z}_N^k$ with the constraint that $\sum_{i\in[1,k]} \triangle\lambda'_i = 0$, and extend the secret key vector and compute the derivated secret key $d_{\chi\cup\chi'}$ as follows:

$$
\begin{aligned}
d'_{\chi'} &= (\boldsymbol{k}'_\rho, \boldsymbol{k}'_w, \boldsymbol{k}'_y, \boldsymbol{k}'_v, \boldsymbol{k}'_r) \\
&= (\boldsymbol{k}_\rho, (\boldsymbol{k}_w || \{\underbrace{1, \cdots, 1}_{k-j}\} \times g^{\boldsymbol{\lambda}'}, \boldsymbol{k}_y || \{\underbrace{1, \cdots, 1}_{k-j}\}, \boldsymbol{k}_v || \{\underbrace{1, \cdots, 1}_{k-j}\}, \boldsymbol{k}_r || \{\underbrace{1, \cdots, 1}_{k-j}\}) \\
&= \left(g^{\boldsymbol{\rho}}, \begin{pmatrix} g^{\lambda_1+\triangle\lambda_1+\langle\rho'_1,\boldsymbol{\sigma}\rangle} w^{y_1} \\ g^{\lambda_2+\triangle\lambda_2+\langle\rho'_2,\boldsymbol{\sigma}\rangle} w^{y_2} \\ \vdots \\ g^{\lambda_j+\triangle\lambda_j+\langle\rho'_j,\boldsymbol{\sigma}\rangle} w^{y_j} \\ \boxed{g^{\triangle\lambda_{j+1}}} \\ \vdots \\ \boxed{g^{\triangle\lambda_k}} \end{pmatrix}^\top, \begin{pmatrix} g^{y_1} \\ g^{y_2} \\ \vdots \\ g^{y_j} \\ \boxed{1} \\ \vdots \\ \boxed{1} \end{pmatrix}^\top, \begin{pmatrix} v^{y_1}(u^{\chi_1}h)^{r_1} \\ v^{y_2}(u^{\chi_2}h)^{r_2} \\ \vdots \\ v^{y_j}(u^{\chi_j}h)^{r_j} \\ \boxed{1} \\ \vdots \\ \boxed{1} \end{pmatrix}^\top, \begin{pmatrix} g^{r_1} \\ g^{r_2} \\ \vdots \\ g^{r_j} \\ \boxed{1} \\ \vdots \\ \boxed{1} \end{pmatrix}^\top \right)
\end{aligned} \tag{4}
$$

2. **Refresh procedure**. At random pick $\triangle\boldsymbol{\rho} \xleftarrow{\$} \mathbb{Z}_N^n$ and $\triangle\boldsymbol{\rho}'_1, \triangle\boldsymbol{\rho}'_2, \cdots, \triangle\boldsymbol{\rho}'_k \xleftarrow{\$} \mathbb{Z}_N^n$ with the constraint that $\triangle\boldsymbol{\rho} = \sum_{i\in[1,k]} \triangle\boldsymbol{\rho}'_i$. Pick $\triangle\boldsymbol{r}, \triangle\boldsymbol{y} \xleftarrow{\$} \mathbb{Z}_N^k$ and $\triangle\boldsymbol{z} \xleftarrow{\$} \mathbb{Z}_N^{n+4k+4}$ randomly, and re-randomize the secret key $d'_{\chi\cup\chi'}$ as follows:

$$
\begin{aligned}
d_{\chi'} &= (\boldsymbol{k}_\rho, \boldsymbol{k}_w, \boldsymbol{k}_y, \boldsymbol{k}_v, \boldsymbol{k}_r) \\
&= (\boldsymbol{k}'_\rho, \boldsymbol{k}'_w, \boldsymbol{k}'_y, \boldsymbol{k}'_v, \boldsymbol{k}'_r) \times (g^{\triangle\boldsymbol{\rho}}, g^{\langle\triangle\boldsymbol{\rho}',\boldsymbol{\sigma}\rangle} w^{\triangle\boldsymbol{y}}, g^{\triangle\boldsymbol{y}}, v^{\triangle\boldsymbol{y}}(u^{\boldsymbol{\chi}'}h)^{\triangle\boldsymbol{r}}, g^{\triangle\boldsymbol{r}}) \times g_3^{\triangle\boldsymbol{z}} \\
&= (g^{\boldsymbol{\rho}+\triangle\boldsymbol{\rho}}, g^{\boldsymbol{\lambda}+\triangle\boldsymbol{\lambda}+\langle\boldsymbol{\rho}'+\triangle\boldsymbol{\rho}',\boldsymbol{\sigma}\rangle} w^{\boldsymbol{y}+\triangle\boldsymbol{y}}, g^{\boldsymbol{y}+\triangle\boldsymbol{y}}, v^{\boldsymbol{y}+\triangle\boldsymbol{y}}(u^{\boldsymbol{\chi}'}h)^{\boldsymbol{r}+\triangle\boldsymbol{r}}, g^{\boldsymbol{r}+\triangle\boldsymbol{r}}) \\
&\quad \times g_3^{\boldsymbol{z}+\triangle\boldsymbol{z}} \\
&= (g^{\boldsymbol{\rho}_\flat}, g^{\boldsymbol{\lambda}_\flat+\langle\boldsymbol{\rho}_\flat,\boldsymbol{\sigma}\rangle} w^{\boldsymbol{y}_\flat}, g^{\boldsymbol{y}_\flat}, v^{\boldsymbol{y}_\flat}(u^{\boldsymbol{\chi}'}h)^{\boldsymbol{r}_\flat}, g^{\boldsymbol{r}_\flat}) \times g_3^{\boldsymbol{z}_\flat}
\end{aligned} \tag{5}
$$

where $\boldsymbol{\rho}_\flat = \boldsymbol{\rho}+\triangle\boldsymbol{\rho}$, $\boldsymbol{\rho}'_\flat = \boldsymbol{\rho}'+\triangle\boldsymbol{\rho}'$, $\boldsymbol{\lambda}_\flat = \boldsymbol{\lambda}+\triangle\boldsymbol{\lambda}$, $\boldsymbol{y}_\flat = \boldsymbol{y}+\triangle\boldsymbol{y}$. $\boldsymbol{z}_\flat = \boldsymbol{z}+\triangle\boldsymbol{z}$.

Remark 1. Obviously, if $\chi' = \phi$, the derivation algorithm will perform the secret key refresh for d_χ. That is, the new secret key renews the randomness $\boldsymbol{\rho}, \boldsymbol{\rho}', \boldsymbol{r}, \boldsymbol{y}$ and parties of $\mathbb{G}_{p_3}$. More specifically, our scheme is continually leakage-resilient such that there are many secret keys per user set, and the attacker is allowed to obtain new leakage with maximum leakage bound on the new secret key after a secret key is refreshed.

Remark 2. It is easy to see that the updated secret key has the the same distribution with the previous secret key, since the new random exponents $\boldsymbol{\rho}, \boldsymbol{\rho}', \boldsymbol{r}'$ and $\boldsymbol{y}'$ are added by uniform randomness $\triangle\boldsymbol{\rho}, \triangle\boldsymbol{\rho}', \triangle\boldsymbol{r}$ and $\triangle\boldsymbol{y}$, independently.

Enc$(\mathtt{PP}, \chi, M)$ In order to transmits a secret message $M \in \mathbb{G}_t$ to a user of set $\chi = \{S_1, \cdots, S_j\}$, the sender picks $s, \boldsymbol{t} \xleftarrow{\$} \mathbb{Z}_N \times \mathbb{Z}_N^j$, and outputs the ciphertext

$$\sigma_\chi := (c_m, \boldsymbol{c}_\sigma, c_s, \boldsymbol{c}_w, \boldsymbol{c}_t, \boldsymbol{c}_u) = (M\hat{e}(g,g)^{s\alpha}, g^{s\boldsymbol{\sigma}}, g^s, w^s v^{\boldsymbol{t}}, g^{\boldsymbol{t}}, (u^{\boldsymbol{\chi}} h)^{\boldsymbol{t}})$$

Dec$(\mathtt{PP}, \chi, d_\chi, \chi', \sigma_{\chi'})$ The decrypter χ can decrypt the message from $\sigma_{\chi'}$. At first, the decrypter computes the indice of $V = \chi \cap \chi'$. If $V \neq \chi$, outputs $\perp$. Let $\mathcal{V} \subseteq \{(c_{\sigma_i}, c_{w_i}, c_{t_i}, c_{u_i}) | i = 1, \ldots, k\}$ be the set of elements of ciphertext $\sigma_{\chi'}$ for which $S_i' = S_i$.

$$M \leftarrow c_m \frac{\hat{e}_n(c_{\boldsymbol{\sigma}}, k_{\boldsymbol{\rho}})}{\hat{e}(c_s, \prod_{i\in\mathcal{V}} k_{w_i})} \prod_{i\in\mathcal{V}} \frac{\hat{e}(c_{w_i}, k_{y_i})\hat{e}(c_{u_i}, k_{r_i})}{\hat{e}(c_{t_i}, k_{v_i})}$$

where $\hat{e}_n(g^{\boldsymbol{\rho}}, g^{\boldsymbol{\sigma}})$ denotes as n pairing operations, i.e., $\hat{e}(g^{\boldsymbol{\rho}}, g^{\boldsymbol{\sigma}}) = \hat{e}(g,g)^{\langle \boldsymbol{\rho}, \boldsymbol{\sigma} \rangle}$.

5 Consistency

5.1 Decryption Correctness

Assume that $\sigma_{\chi'}=(c_m, \boldsymbol{c}_\sigma, c_s, \boldsymbol{c}_w, \boldsymbol{c}_t, \boldsymbol{c}_u)$, and $d_\chi = (\boldsymbol{k}_\rho, \boldsymbol{k}_w, \boldsymbol{k}_y, \boldsymbol{k}_v, \boldsymbol{k}_r)$, we observe that all components of $\sigma_{\chi'}$ (except c_m) are the elements in $\mathbb{G}_{p_1}$ and all components of d_χ are the elements in $\mathbb{G}_{p_1 p_3}$. According to the subgroups orthogonality, we can eliminate all terms of $\mathbb{G}_{p_3}$ in secret key d_χ when performing the bilinear calculations. Also, $\chi \sqsubseteq \chi'$ if d_χ can correctly decrypt the ciphertext $\sigma_{\chi'}$, which means that the components in $\mathcal{V}$. We have

$$\begin{aligned}
& c_m \frac{\hat{e}_n(c_{\boldsymbol{\sigma}}, k_{\boldsymbol{\rho}})}{\hat{e}(c_s, \prod_{i\in\mathcal{V}} k_{w_i})} \prod_{i\in\mathcal{V}} \frac{\hat{e}(c_{w_i}, k_{y_i})\hat{e}(c_{u_i}, k_{r_i})}{\hat{e}(c_{t_i}, k_{v_i})} \\
&= c_m \frac{\hat{e}_n(g^{\boldsymbol{\rho}}, g^{s\boldsymbol{\sigma}})}{\hat{e}(c_s, \prod_{i=1}^{j} k_{w_i})} \prod_{i\in\mathcal{V}} \frac{\hat{e}(w^s v^{t_i}, g^{y_i})\hat{e}((u^{\chi_i} h)^{t_i}, g^{r_i})}{\hat{e}(g^{t_i}, v^{y_i}(u^{\chi_i} h)^{r_i})} \\
&= c_m \frac{\hat{e}_n(g^{\boldsymbol{\rho}}, g^{s\boldsymbol{\sigma}})}{\hat{e}(c_s, \prod_{i=1}^{j} g^{\lambda_i + \langle \rho_i', \boldsymbol{\sigma} \rangle} w^{y_i})} \prod_{i\in\mathcal{V}} \frac{\hat{e}(w^s v^{t_i}, g^{y_i})\hat{e}((u^{\chi_i} h)^{t_i}, g^{r_i})}{\hat{e}(g^{t_i}, v^{y_i})\hat{e}(g^{t_i}, (u^{\chi_i} h)^{r_i})} \\
&= c_m \frac{\hat{e}(g,g)^{s\langle \boldsymbol{\rho}, \boldsymbol{\sigma} \rangle} \prod_{i\in\mathcal{V}} \hat{e}(w^s, g^{y_i})}{\hat{e}(g^s, g^{\sum_{i=1}^{j} \lambda_i})\hat{e}(g^s, g^{\langle \sum_{i=1}^{j} \rho_i', \boldsymbol{\sigma} \rangle}) \prod_{i\in\mathcal{V}} \hat{e}(g^s, w^{y_i})} \\
&= \frac{M\hat{e}(g,g)^{s\alpha}\hat{e}(g,g)^{s\langle \boldsymbol{\rho}, \boldsymbol{\sigma} \rangle}}{\hat{e}(g^s, g^\alpha)\hat{e}(g^s, g^{\langle \sum_{i=1}^{j} \rho_i', \boldsymbol{\sigma} \rangle})} \qquad (\sum_{i=1}^{j} g^{\lambda_i} = g^\alpha) \\
&= \frac{M\hat{e}(g,g)^{s\langle \boldsymbol{\rho}, \boldsymbol{\sigma} \rangle}}{\hat{e}(g^s, g^{\langle \boldsymbol{\rho}, \boldsymbol{\sigma} \rangle})} = M \qquad (\sum_{i=1}^{j} \boldsymbol{\rho}_i' = \boldsymbol{\rho}) \qquad (6)
\end{aligned}$$

5.2 Derivation Invariance

Since the derivation algorithm additively re-randomizes each exponents $r_1, \cdots, r_k, y_1, \cdots, y_k, \rho_1, \cdots, \rho_n$ in a secret key $d_{\chi=\{S_1,\cdots,S_k\}}$, and the product of $g^{\lambda_1} g^{\lambda_2} \cdots g^{\lambda_k}$ is equal to g^{α} with randomized $\lambda_i \in \mathbb{Z}_N$, it has the same distribution of a parent secret key d_χ. Also, it is easy to see that $d_{\chi \cup \chi'}$ created through any sequence of derivations Del is the same as the distribution of a secret key of same set created by Ext algorithm.

6 Security

6.1 Straightforward Proof Idea

In order to simplify the proof, we assume that the element is appended to the tail of the previous set. Because multiple elements can be emerged by multiple calling to Del algorithm that each time emerges an element, we consider the case of one element emerge. The security proof works as follows:

At first we convert the challenge ciphertext into semi-functional form, and then convert the secret keys into a semi-functional form one by one. A semi-functional key/ciphertext has $\mathbb{G}_{p_2}$ part, while a normal key/ciphertext has not. If the order N can not be factored, we can show that these conversions are indistinguishable. At the same time, we indicate that the attacker's decryption ability is indistinguishable before/after he obtains partial decryption secret key.

Let g_2 be a generator of $\mathbb{G}_{p_2}$, a semi-functional key and a semi-functional ciphertext are created as follows.

EncSF Let $\sigma_\chi = (c_m, \boldsymbol{c}_\sigma, c_s, \boldsymbol{c}_w, \boldsymbol{c}_t, \boldsymbol{c}_u)$ be a normal ciphertext created by Enc algorithm, a semi-functional ciphertext is constructed as follows: at random select $\boldsymbol{z}_1, z_2, \boldsymbol{z}_3, \boldsymbol{z}_4, \boldsymbol{z}_5 \xleftarrow{\$} \mathbb{Z}_N^n \times \mathbb{Z}_N \times \mathbb{Z}_N^j \times \mathbb{Z}_N^j \times \mathbb{Z}_N^j$ and set the corresponding semi-functional ciphertext $\widehat{\sigma}_\chi$ as

$$\begin{aligned} \widehat{\sigma}_\chi &:= (c_m, \hat{c}_\sigma, \hat{c}_s, \hat{c}_w, \hat{c}_t, \hat{c}_u) \\ &= (c_m, \boldsymbol{c}_\sigma \times g_2^{\boldsymbol{z}_1}, c_s \times g_2^{z_2}, \boldsymbol{c}_w \times g_2^{\boldsymbol{z}_3}, \boldsymbol{c}_t \times g_2^{\boldsymbol{z}_4}, \boldsymbol{c}_u \times g_2^{\boldsymbol{z}_5}) \end{aligned} \tag{7}$$

ExtSF Let $d_\chi = (\boldsymbol{k}_\rho, \boldsymbol{k}_w, \boldsymbol{k}_y, \boldsymbol{k}_v, \boldsymbol{k}_r)$ be a normal secret key produced by Ext or Del algorithm, a semi-functional key $\widetilde{d}_\chi$ is produced as follows: pick $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_5 \xleftarrow{\$} \mathbb{Z}_N^n \times \mathbb{Z}_N^j \times \mathbb{Z}_N^j \times \mathbb{Z}_N^j \times \mathbb{Z}_N^j$ randomly, and calculate

$$\begin{aligned} \widehat{d}_\chi &:= (\hat{k}_\rho, \hat{k}_w, \hat{k}_y, \hat{k}_v, \hat{k}_r) \\ &= (\boldsymbol{k}_\rho \times g_2^{\boldsymbol{x}_1}, \boldsymbol{k}_w \times g_2^{\boldsymbol{x}_2}, \boldsymbol{k}_y \times g_2^{\boldsymbol{x}_3}, \boldsymbol{k}_v \times g_2^{\boldsymbol{x}_4}, \boldsymbol{k}_r \times g_2^{\boldsymbol{x}_5}) \end{aligned} \tag{8}$$

Here $\{\boldsymbol{x}_i, \boldsymbol{z}_i\}_{i \in [1,5]}$ are called the semi-functional factors of secret key and ciphertext. If χ_1 uses a semi-functional key $\widehat{d}_{\chi_1}$ to create a delegation key of χ_2, the new semi-functional factors are extend to be $|\chi_2|$ dimensional except $\boldsymbol{x}_1, \boldsymbol{z}_1$.

Decryption will succeed if a valid semi-functional key is used to decrypt a normal ciphertext, or a normal key is used to decrypt a semi-functional ciphertext However, decryption will cause an ambiguous output when using a semi-functional key to decrypt a semi-functional ciphertext, since it will lead to an extra term of g_2.

$$\hat{e}(g_2, g_2)^{\sum_{i=1}^{n} x_{1i} z_{1i} + \sum_{i \in \mathcal{V}} (x_{3i} z_{3i} + x_{5i} z_{5i} - x_{2i} z_2 - x_{4i} z_{4i}) \bmod p_2}$$

Obviously, if the exponent in $\hat{e}(g_2, g_2)$ is zero, decryption still works and the key is *nominally* semi-functional. Otherwise, we say that the key is *truly* semi-functional.

6.2 Indistinguishable Games

To prove the security for our scheme, we devise a sequence of games that are proven to be indistinguishable. The first game is the real scheme and security model of our scheme that the ciphertexts and secret keys are normal. In the second game, we convert the challenge ciphertext into a semi-functional form, and the secret keys are still unchanged. For an attacker $\mathcal{A}$ who at most queries q secret keys, where $q = \mathsf{poly}(\tau)$. We convert the secret keys to be semi-functional one by one. During the change of secret keys into semi-functional form, we guarantee that in $\text{Game}_{3,k}$ the first $k (1 \leq k \leq q)$ keys are semi-functional and the rest of keys are normal. In $\text{Game}_{3,q}$, all the secret keys and the challenge ciphertext are semi-functional. Therefore, according to the thought of dual system encryption, all queried secret keys are unable to decrypt the challenge ciphertext. We also prove that the attacker $\mathcal{A}$ is not aware of these changes that converts from normal forms into semi-functional forms. We define the indistinguishable games as follows:

Game_0. This game is the real scheme of proposed construction in Section 4.2 and the security is defined in Section 3.2. In this game, all secret keys and ciphertexts are normal, and a valid normal key can decrypt a normal ciphertext.

Game_1. This game is the same as Game_0 except that all $\mathcal{O}_{\mathsf{Del}}$ queries are replaced by $\mathcal{O}_{\mathsf{Ext}}$ oracle.

Game_2. This game is similar to Game_1 except that the challenge ciphertext is converted into semi-functional form. Under Assumption 1, we show that the advantage of an attacker in distinguishing this conversion is negligible.

$\text{Game}_{3,0}$. This game is the same as Game_2 except that the attacker can not ask for the secret keys that the set is the challenge set modulo p_2; We will hold this constraint in the next games.

$\text{Game}_{3,k}$. Let $q (q = \mathsf{poly}(\tau))$ be the amount of secret key queries that the attacker makes. For $k = 1, \cdots, q$, we refine $\text{Game}_{3,k}$ like in Game_2, except that the first k keys are semi-functional and the rest secret keys are normal. Obviously, in these games, (1) for all queried set $\chi = \{S_1, \cdots, S_j\}$, $S_i \neq 0 \bmod p_2$; (2) the ciphertexts are semi-functional; (3) the first k keys are semi-functional and the remaining keys are normal.

Game$_4$. This game is the same as Game$_{3,q}$ with the added restriction that the component c_m is replaced by a random element in $\mathbb{G}_t$. This game means that the message M is hidden in the ciphertext so that the attacker has negligible advantage in guessing the message from the ciphertext. We show that the advantage of an attacker in distinguishing this change is negligible under Assumption 3.

Claim 1. *Any polynomial-time attacker $\mathcal{A}$ has a negligible advantage in distinguishing Game$_1$ from Game$_0$.*

Claim 2. *If there exists an attacker $\mathcal{A}$ in distinguishing Game$_2$ from Game$_1$ with advantage ϵ, there exists a PPT algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 1.*

Claim 3. *If there exists a polynomial attacker $\mathcal{A}$ with advantage ϵ in distinguishing Game$_{3,0}$ from Game$_2$, then we can construct a PPT algorithm $\mathcal{B}$ with the same advantage to break Assumption 2.*

Claim 4. *Suppose that there exists an attacker $\mathcal{A}$ in distinguishing Game$_{3,k}$ from Game$_{3,k+1}$ with non-negligible advantage ϵ, then we can construct a PPT algorithm $\mathcal{B}$ in breaking Assumption 2.*

Claim 5. *Suppose that the leakage is at most $n_d = 2 + (n - 1 - 2c) \log p_2$, where c (> 0) is a positive constant. For any PPT attacker $\mathcal{A}$ in Game$_{k+1}$, whenever $\mathcal{A}$ declares the k-th key to be associated to a space that contains the challenge ciphertext vector, $\mathcal{A}$'s advantage in changing when the truly semi-functional k key is replaced by a nominal semi-functional key is p_2^{-c}.*

Claim 6. *If there exists a polynomial attacker $\mathcal{A}$ in distinguishing Game$_4$ from Game$_{3,q}$ with non-negligible advantage, we can construct a PPT algorithm $\mathcal{B}$ in breaking Assumption 3.*

The proofs are based on the subgroup decisional problems defined in Section 2.3. The proof of lemma 2, 3, 4, 5 and 6 are given in the full version.

Theorem 1. ***Leakage resilience security*** *Suppose that a composite order group generator $\mathcal{G}_{\mathsf{cp}}$ satisfies the security assumptions 1, 3 and 3, an attacker has at most q-times key extraction/delegation queries, at most ℓ-bit leakage queries for every secret key, then the LR-UESDE scheme is (q, ℓ) semantically secure against secret keys leakage with leakage bound ℓ.*

Proof. In the composite order bilinear group G, if the assumptions 1,3 and 3 hold, then we prove that the real Game$_0$ is computationally indistinguishable from Game$_4$ by Claims (2) - (6), in which the value ψ is information-theoretically hidden. The attacker has negligible advantage in winning Game$_1$, which is the actual security game for the proposed LR-UESDE scheme. □

7 Leakage Bound and Performance Analysis

We evaluate the performance of our scheme and provide the efficiency tradeoffs. In Lemma 1 and Corollary 1, we set $n = m - 1$ and $d = 1, c \geq 2$, then ${p_2}^{-c}$ is negligible when p_2 is a large prime. Since $|W| \leq 4{p_2}^{m-3}(p_2-1)\epsilon(\cdot)^2 \leq 4{p_2}^{n-1-2c}$, we have $\ell = \log_2 |W| = 2 + (n-1-2c)\log_2 p_2$ bits. Here $n \geq 2$ is an integer and c is a positive constant. Obviously, the leakage bound is determined by the order of subgroup $\mathbb{G}_{p_2}$. We can obtain variable key sizes and allow different fractions of leakage by varying the relative size of subgroups of $\mathbb{G}$.

We use the term *leakage fraction* to denote the number of bits allowed to be leaked from a key divided by the number of bits to the total key size. We assume that p_1, p_2, p_3 are primes of τ_1, τ_2, τ_3 bits, where $2^{\tau_i - 1} \leq p_i \leq 2^{\tau_i}$ for $i = 1, 2, 3$, respectively. We assume that the size of subgroup elements are represented by approximately $\sum_{i=1}^{3} \tau_i = \Theta(\log_2 N)$ bits. By fixing $\tau_1 = \beta_1\tau$, $\tau_2 = \tau$, and $\tau_3 = \beta_2\tau$, where τ is the system security parameter and β_1, β_2 are positive constants, we can calculate the leakage fraction LF_{d_χ} of secret key d_χ is

$$LF_{d_\chi} = \frac{n - 1 - 2c + \frac{2}{\log_2 p_2}}{(1 + \beta_1 + \beta_2)(n + 4|\chi|)} \approx \frac{n - 1 - 2c}{(1 + \beta_1 + \beta_2)(n + 4|\chi|)}$$

Intuitively, the higher value of n, the better leakage fraction. However, it will yield the larger system public key, secret keys and ciphertexts. We are worth mentioning that the smaller values of β_1 and β_3 lead to a better leakage fraction, but it will imply shorter security parameters in subgroup $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_3}$. More detail, the size of system public key $|\mathsf{PP}| = |\mathbb{G}_t| + (n+5)|\mathbb{G}_{p_1}| + |\mathbb{G}_{p_3}|$, the size of secret key $|d_\chi| = (n+4|\chi|)|\mathbb{G}|$, and the size of ciphertext $|\sigma_\chi| = |\mathbb{G}_t| + (4|\chi| + n + 1)|\mathbb{G}|$. Obviously, the system public key is constant that is independent to the delegation set χ. That is, our system can perform unbounded delegation depth.

8 Conclusion

We presented a leakage-resilient extensible set encryption with unbounded delegation ability. The proposed scheme can tolerate the secret key leakage of both memory leakage and continual leakage. We provided the security proof under the dual system encryption mechanism in the standard model. We also assessed the allowable leakage bound. As our scheme does not need to pre-establish a possible global delegation depth in the system, it is flexible and scalable in supporting arbitrary level of delegation depth in the practical applications. We also leave an open problem to construct a leakage-resilient encryption to obtain both unbounded delegation depth and constant keys/ciphertexts simultaneously.

Acknowledgment. The authors grateful thank the anonymous reviewers for their helpful comments and suggestion. This work is supported by the NSFC (№60973134,№61173164, №61170135),Guangdong Natural Science Foundation (№10151064201000028), and the support by Grant-in-Aid for JSPS Fellows of Japan(№22·00045).

References

1. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous Hardcore Bits and Cryptography against Memory Attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
2. Alwen, J., Dodis, Y., Naor, M., Segev, G., Walfish, S., Wichs, D.: Public-Key Encryption in the Bounded-Retrieval Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 113–134. Springer, Heidelberg (2010)
3. Alwen, J., Dodis, Y., Wichs, D.: Leakage-Resilient Public-Key Cryptography in the Bounded-Retrieval Model. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 36–54. Springer, Heidelberg (2009)
4. 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)
5. 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)
6. Boyen, X., Waters, B.: Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
7. Brakershi, Z., Kalai, Y.T., Katz, J., Vaikuntanathan, V.: Overcoming the hole in the bucket: public-key cryptogaphy resilient to continual memory leakage. In: FOCS 2010, pp. 501–510. IEEE (2010)
8. Brakerski, Z., Goldwasser, S.: Circular and Leakage Resilient Public-Key Encryption under Subgroup Indistinguishability. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 1–20. Springer, Heidelberg (2010)
9. Chow, S., Dodis, D., Rouselakis, Y., Waters, B.: Practical leakage-resilient identity-based encryption from simple assumptions. In: ACM-CCS 2010, pp. 152–161 (2010)
10. Ducas, L.: Anonymity from Asymmetry: New Constructions for Anonymous HIBE. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 148–164. Springer, Heidelberg (2010)
11. Gentry, C., Halevi, S.: Hierarchical Identity Based Encryption with Polynomially Many Levels. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 437–456. Springer, Heidelberg (2009)
12. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
13. Lewko, A., Rouselakis, Y., Waters, B.: Achieving Leakage Resilience through Dual System Encryption. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 70–88. Springer, Heidelberg (2011)
14. 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)
15. Lewko, A., Waters, B.: Unbounded HIBE and Attribute-Based Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 547–567. Springer, Heidelberg (2011)
16. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)

17. 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)
18. 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)
19. Yuen, T.H., Chow, S.S.M., Zhang, Y., Yiu, S.M.: Identity-Based Encryption Resilient to Continual Auxiliary Leakage. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 117–134. Springer, Heidelberg (2012)
20. Zhang, M., Nishide, T., Yang, B., Takagi, T.: Anonymous Encryption with Partial-Order Subset Delegation Functionality. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 154–169. Springer, Heidelberg (2011)

Anonymous Identity-Based Hash Proof System and Its Applications

Yu Chen[1], Zongyang Zhang[2,⋆], Dongdai Lin[1], and Zhenfu Cao[2]

[1] State Key Laboratory of Information Security (SKLOIS),
Institute of Information Engineering, CAS, Beijing 100093, China
{chenyu,ddlin}@iie.ac.cn
[2] Department of Computer Science and Engineering,
Shanghai Jiao Tong University, China
{zongyangzhang,zfcao}@sjtu.edu.cn

Abstract. We introduce the concept of anonymous identity-based hash proof system (IB-HPS), and show how to use it to construct identity-based encryption schemes providing anonymity in the presence of key leakage. We give four different constructions of anonymous IB-HPS based on: (1) the decision bilinear Diffie-Hellman assumption, (2) the decision truncated augmented bilinear Diffie-Hellman exponent assumption, (3) the quadratic residuosity assumption, and (4) the decision learning with errors assumption.

1 Introduction

The classical definitions of security for encryption schemes are mainly concerned with the data privacy of the encrypted data. For example, the widely accepted notions such as one-wayness (OW) and indistinguishability (IND) are both directed at capturing different levels of data privacy in encryption schemes. However, in some applications, key privacy of the encrypted data is also required.

The concept of key privacy (or anonymity) was first formalized in the context of symmetric-key encryption [1, 20, 25] and was later extended to the case of public-key encryption (PKE) [9] and identity-based encryption (IBE) [2]. Briefly speaking, in public-key setting and identity-based setting, anonymity requires that a ciphertext does not reveal information of its intended recipient. Several PKE and IBE schemes in the literature are shown to be anonymous, such as the work [18, 12, 26, 5] in the standard model, and the work [4, 10, 11, 27] in the random oracle model.

It is easy to see that the goals of data privacy and key privacy are orthogonal [9, 35]. However, unlike data privacy, key privacy is comparatively less studied. Recently, a large body of work [32, 29, 8, 28] emerged on constructing encryption schemes with data privacy against various *key leakage attacks*. Several IBE schemes [7, 15, 30, 14] secure against key leakage attacks have been proposed. However, all these work only focus on the security (data privacy) but

⋆ Corresponding author.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 143–160, 2012.

not the anonymity (key privacy). Thus, it is compelling to study the anonymity of IBE schemes in the context of key leakage attacks.

1.1 Related Work

Leakage Resilient Cryptography. Recently, much progress has been made on *leakage-resilient cryptography* with the goal to design provably secure cryptographic primitives against a myriad of *side-channel attacks* (e.g., power, timing, radiation, cold-boot etc.), where the adversary may obtain limited additional information about secret keys and other internal secret state, not captured by the traditional definitions. So far, there are mainly two leakage models for leakage-resilient cryptographic schemes. One is called the "bounded-leakage model" [6, 32, 8, 29, 7, 15], which does not restrict the type of leakage that the adversary can obtain, but has to bound the overall amount of leakage information. The other is called the "continuous-leakage model" [13, 31, 23, 24], which only bounds the amount of leakage *per period* (as opposed to overall) by continually refreshing the secret keys, but has to place additional non-trivial restrictions on the types of leakage.

Hash Proof System and Identity-Based Hash Proof System. Cramer and Shoup [18] generalized their initial PKE scheme [17] to the paradigm of hash proof system (HPS), thereby serving as a useful primitive to construct CCA-secure PKE schemes. Boneh *et al.* [11] presented a variant of HPS, named HPS with trapdoor, and use it to construct anonymous and CCA-secure IBE schemes. As an instantiation, they proposed a variant of Cocks IBE [16] which achieves anonymity and shorter ciphertext size. Naor and Segev [32] presented a generic construction of PKE schemes that are resilient to key leakage from any HPS. Alwen *et al.* [7] extended HPS to the identity-based setting, namely the identity-based hash proof system (IB-HPS). They used it as a basic tool to construct IBE schemes that provide security against various forms of key leakage attacks in the "bounded-retrieval model" (BRM).

1.2 Our Contribution

As our main contribution, we investigate the anonymity of IBE schemes in the presence of key leakage attacks and construct the first leakage-resilient anonymous IBE schemes in the BRM. Along the way, we develop new notions and get results of independent interest. In particular, we:

- Define the notion of leakage-resilient anonymity of IBE schemes.
- Reformulate the paradigm of IB-HPS in a fine-grained way and highlight its relation to the subset membership problem and the projective hash family; define an additional property named (leakage-resilient) anonymity of IB-HPS; show how to construct (leakage-resilient) anonymous IBE schemes from anonymous IB-HPS.
- Give four constructions of anonymous IB-HPS based on the ideas behind prior IBE schemes: [26, 11, 27, 14].

– Define the notion of leakage-resilient anonymity of PEKS schemes, and construct leakage-resilient anonymous PEKS schemes through leakage-resilient anonymous IBE schemes, which is in turn derived from anonymous IB-HPS.

2 Definitions and Preliminaries

2.1 Notation

For a finite set S, we use $s \xleftarrow{R} S$ to denote that s is sampled from the set S uniformly at random, and use U_S to denote the uniform distribution over S. For an integer $n \in \mathbb{N}$, we use U_n to denote the uniform distribution over $\{0,1\}^n$. The main security parameter through this paper is κ, and all algorithms (including the adversary) are implicitly given the security parameter κ. A probabilistic polynomial-time (PPT) algorithm is a randomized algorithm that runs in time polynomially in κ. If $\mathcal{A}$ is a randomized algorithm, we write $z \leftarrow \mathcal{A}(x_1, \ldots, x_n; r)$ to indicate that $\mathcal{A}$ outputs z on inputs $(x_1, \ldots, x_n)$ and random coins r. Sometimes for brevity, we omit r and write $z \leftarrow \mathcal{A}(x_1, \ldots, x_n)$ when it is not necessary to make explicit the random coins $\mathcal{A}$ uses.

2.2 Min-entropy and Randomness Extractor

Here we review some concepts related probability distributions and randomness extractors.

The *statistical distance* between two random variables X, Y over a finite domain Ω is defined as $\mathbf{SD}(X,Y) = \frac{1}{2}\sum_{\omega\in\Omega} |\Pr[X=\omega] - \Pr[Y=\omega]|$. We say that two variables are ϵ-*close* if their statistical distance is at most ϵ.

The *min-entropy* of a random variable X over a domain Ω is the negative (base-2) logarithm of the *predictability* of X: $\mathbf{H}_\infty(X) = -\log_2(\max_{\omega\in\Omega}\Pr[X=\omega])$. In many natural settings, the variable X is correlated with another variable Y whose value is known to an adversary. In such scenarios, it is more convenient to use the notion of *average min-entropy* [21], which captures the *average predictability* of X given knowledge of Y. This is formally defined as

$$\widetilde{\mathbf{H}}_\infty(X|Y) = -\log_2\left(E_{y\leftarrow Y}\left[\max_{\omega\in\Omega}\Pr[X=\omega|Y=y]\right]\right)$$

where $E_{y\leftarrow Y}$ denotes the expected value over all values of Y. The following lemma bound on average min-entropy was proved in [21]:

Lemma 1. *If Y takes at most 2^r possible values and Z is any random variable, then*

$$\widetilde{\mathbf{H}}_\infty(X|(Y,Z)) \geq \mathbf{H}_\infty(X|Z) - r$$

A main tool in our constructions is a strong randomness extractor [33]. The following definition naturally generalized the standard definition of a strong extractor to the setting of average min-entropy:

Definition 1. *A polynomial-time function* $\mathsf{Ext} : \Omega \to \{0,1\}^v$ *is an average case* (m, ϵ)*-strong extractor if for all pairs of random variables* (X, Y) *such that* X *is distributed over* Ω *and* $\widetilde{\mathbf{H}}_\infty(X|Y) \geq m$ *and a random variable* S *over the seeds set* $\{0,1\}^\mu$ *for some integer* μ*, we have that*

$$\mathbf{SD}((\mathsf{Ext}(X;S), S, Y)), (U_m, S, Y)) \leq \epsilon$$

Dodis *et al.* [21] proved that any strong extractor is in fact an average-case strong extractor, for a proper setting of the parameters:

Lemma 2 ([21]). *For any* $\delta > 0$*, if* Ext *is a (worst case)* $(m - \log(1/\delta), \epsilon)$*-strong extractor, then* Ext *is also an average-case* $(m, \epsilon + \delta)$*-strong extractor.*

As a specific example, they proved the following lemma which essentially gives an explicit construction of an average-case strong extractor:

Lemma 3 ([21]). *Let* X *and* Y *be two random variables such that* $X \in \{0,1\}^n$ *and* $\widetilde{\mathbf{H}}_\infty(X|Y) \geq k$*. Let* $\mathcal{H}$ *be a family of universal hash functions from* $\{0,1\}^n$ *to* $\{0,1\}^m$*. Then for* $h \xleftarrow{R} \mathcal{H}$*, it holds that* $\mathbf{SD}((h(X), U_s, Y)), (U_m, U_s, Y)) \leq \epsilon$ *as long as* $m \leq k - 2\log(1/\epsilon)$*.*

2.3 Identity-Based Encryption

An IBE scheme [34, 10] consists of four PPT algorithms:

- $\mathsf{Setup}(\kappa)$: take as input a security parameter κ, and output a master public/secret key pair (mpk, msk). Here mpk is the system parameters which is publicly known, while msk is the master secret key and is known only to Private Key Generator (PKG). We assume that mpk also includes the description of identity set I and message set M. mpk will be used as an implicit input of algorithms KeyGen, $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$.
- $\mathsf{KeyGen}(msk, id)$: take as input msk and an identity id, and output a private key sk.
- $\mathsf{Encrypt}(id, m)$: take as input an identity id and a message m, and output a ciphertext c.
- $\mathsf{Decrypt}(sk, c)$: take as input a private key sk and a ciphertext c, and output the message m or a reject symbol $\perp$ indicates that c is not well-formed.

3 Leakage-Resilient Anonymous IBE

Intuitively, we say that an IBE scheme is anonymous if no PPT adversary can distinguish the identity under which a ciphertext was generated. Formal definitions for anonymity of IBE can be founded in [2, 3]. To capture the anonymity of IBE schemes in the presence of a variety of key leakage attack, we define leakage-resilient anonymity of IBE schemes by modifying the usual anonymity game (against chosen-plaintext attacks) appropriately in the bounded-retrieval

model [19, 22, 8, 7]. This model imposes an additional requirement on LR schemes by insisting them to provide a way to "grow" the secret key so as to proportionally increase the amount of tolerated leakage, but without remarkably increasing the size of the public key and lowing the efficiency of the schemes. The game is parametrized by a security parameter κ and a leakage parameter ℓ played between an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$.

Setup: $\mathcal{CH}$ runs $\mathsf{Setup}(\kappa)$ to generate (mpk, msk) and gives mpk to $\mathcal{A}$.
Phase 1: $\mathcal{A}$ can adaptively make one of the following two types queries to $\mathcal{CH}$. In the following, let $f_i : SK \to \{0,1\}^{\ell_i}$ be an arbitrary function from SK to $\{0,1\}^{\ell_i}$, where SK is the set of private keys.

- Private key reveal query $\langle id \rangle$: $\mathcal{CH}$ responds by running KeyGen algorithm to generate a private key sk for identity id.
- Private key leakage query $\langle id, f_i \rangle$[1]: $\mathcal{CH}$ checks if the overall amount leakage with respect to id exceeds ℓ. If not, it responds with $f_i(sk)$. Otherwise it responds with a reject symbol $\perp$.

Challenge: Once $\mathcal{A}$ decides that Phase 1 is over, it outputs a message m and two equal-length identities id_0 and id_1 on which it wishes to be challenged. The restriction is that $\mathcal{A}$ did not issue the private key reveal query for id_0 or id_1 in Phase 1. $\mathcal{CH}$ picks a random bit $b \in \{0,1\}$ and gives $\mathcal{A}$ the challenge ciphertext $c^* = \mathsf{Encrypt}(id_b, m)$.
Phase 2: The same as Phase 1 except that the private key leakage queries or reveal queries related to id_0 and id_1 are forbidden.
Guess: $\mathcal{A}$ outputs $b' \in \{0,1\}$ and wins the game if $b' = b$.

We call $\mathcal{A}$ in the above-described game a ℓ-leakage anonymity adversary and define its advantage as $\mathrm{Adv}_{\mathcal{A}}(\kappa, \ell) = |\Pr[b' = b] - 1/2|$. We note that for the main purpose of this paper, leakage-resilient security and anonymity are only considered under chosen plaintext attack.

Definition 2 (Leakage-Resilient Anonymous IBE). *An IBE scheme is ℓ-leakage-resilient anonymous if the advantage of any PPT adversary $\mathcal{A}$ in the above game is negligible in κ. Let the relative leakage of the scheme be $\alpha = \ell/\hat{m}$, where $\hat{m}$ is the number of bits needed to efficiently store a private key sk.*

4 Identity-Based Hash Proof Systems

The paradigm of IB-HPS has appeared in different forms in previous literature [11, 7]. In [11], an IB-HPS is viewed as a HPS with trapdoor. However, their definition is not fine-grained enough to encompass all the IBE schemes relying on hash proof techniques, e.g. Gentry IBE [26]. In the work [7], an IB-HPS is viewed as an IB-KEM. Their treatment makes the transform from IB-HPS to

[1] As prior work [15, 7] did, we allow the generation of multiple private keys but we require that the leakage comes from only one.

IBE more directly but obscure the connection among IB-HPS and the underlying subset membership problem and projective hash family. They also introduced a property named "anonymous encapsulation" of IB-HPS, which is useful to bring anonymity, improve leakage-amplification, and shorten the ciphertexts for the resulting IBE schemes. Their definition of anonymous encapsulation is: for the original encapsulation algorithm, there exists an efficient and equivalent encapsulation algorithm that can generate the ciphertext and the DEM key without knowing the intended identity (independently of id). However, this definition seems a bit narrow. At least, it is unclear how to explain the anonymity of Gentry IBE [26] using their framework.

In what follows, we redefine the paradigm of IB-HPS in a fine-grained way and highlight its relation to the subset membership problem and the projective hash family. We also define a property named anonymity of IB-HPS. The resulting anonymous IB-HPS is general enough to encompass all the currently known anonymous IBE schemes based on hash proof techniques.

4.1 Subset Membership Problem (SMP)

A subset membership problem $\mathbf{M}$ specifies a collection $(D_\kappa)_{\kappa \geq 0}$ of distributions. For each security parameter $\kappa \geq 0$, D_κ is a probability distribution of instances. Each instance Γ specifies the following:

- Finite non-empty sets X, W, PK, and a collection of sets $V = (V_{pk})_{pk \in PK}$ indexed by PK. For each $pk \in PK$, V_{pk} is a proper subset of X.
- A collection of binary relations $\mathsf{R} = (\mathsf{R}_{pk})_{pk \in PK}$ indexed by PK. For each $pk \in PK$, R_{pk} is a binary relation over $X \times W$. In particular, for $x \in X$ and $w \in W$ with $(x, w) \in \mathsf{R}_{pk}$, we say that w is a *witness* for x. We require that for all $x \in X$, $(x, w) \in \mathsf{R}_{pk}$ for some $w \in W$ if and only if $x \in V_{pk}$. The relation R_{pk} is actually defined over $V_{pk} \times W$.

We write $\Gamma[X, V, W, PK, \mathsf{R}]$ to indicate that the instance Γ specifies X, V, W, PK, and R as above. $\mathbf{M}$ also provides several basic algorithms:

- $\mathsf{Gen}(\kappa)$: take as input a security parameter κ and sample a random instance Γ according to the distribution D_κ. We denote by SP the random coins used by Gen.
- $\mathsf{SampR}(pk)$: take as input a value $pk \in PK$, and output a random $x \in V_{pk}$ with witness $w \in W$.
- $\mathsf{SampR}^*(pk)$: take as input a value $pk \in PK$, and output a random $x \in X \backslash V_{pk}$.

We say that $\mathbf{M}$ is a hard subset membership problem if for any $pk \in PK$ it is computationally hard to distinguish random elements of V_{pk} from random elements of $X \backslash V_{pk}$. The subset membership assumption for $\mathbf{M}$ means that $\mathbf{M}$ is a hard subset membership problem.

4.2 Projective Hash Family (PHF)

Let X, Y, SK, PK be finite non-empty sets, and V be a collections of sets indexed by PK as defined in SMP. Let $\mathsf{H} = \{\mathsf{H}_{sk} : X \to Y\}_{sk \in SK}$ be a collection of functions indexed by SK. Let $\alpha : SK \to PK$ be a function from SK to PK, which can be seen as a *projection*. Here the term "projection" indicates that α is a many-to-one mapping. We refer to the tuple $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$ as a projective hash family (PHF) if for any $sk \in SK$ and $pk = \alpha(sk)$, the action of H_{sk} on V_{pk} is determined by $\alpha(sk)$ – i.e., given $\alpha(sk)$ and $x \in V_{pk}$, $\mathsf{H}_{sk}(x)$ is uniquely determined.

4.3 Definition of IB-HPS

Compared to the definitions of SMP and PHF in [18, 11], our definitions introduces V as a collection of sets indexed by PK but not a single set, and R as a collection of relations indexed also by PK but not a single relation. Correspondingly, in [18, 11] the algorithms SampR and SampR^* of $\mathbf{M}$ are independent of the intended public key pk, while in our definition the two algorithms take $pk \in PK$ as their input. We will see that these modifications allow us to build an IB-HPS that can encompass all known IBE schemes using hash proof technique.

Let $\mathbf{P}$ be an identity-based hash proof system (IB-HPS) for a subset membership problem $\mathbf{M}$ associates with each instance $\Gamma[X, V, W, PK, \mathsf{R}]$ of $\mathbf{M}$ and a corresponding projective hash family $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$. $\mathbf{P}$ specifies an identity set I, a function $\mathsf{IHF} : I \to PK$, and consists of four algorithms (Setup, KeyGen, Pub, Priv) as follows.

- $\mathsf{Setup}(\kappa)$: run $\mathsf{Gen}(\kappa)$ to generate an instance $\Gamma[X, V, W, PK, \mathsf{R}]$ of $\mathbf{M}$, pick a suitable projective hash family $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$, and create a master public/secret key pair (mpk, msk), in which mpk includes the descriptions of Γ, $\mathbf{H}$, I and IHF. We require that α can be efficiently invertible with msk, and write its inverse function as $\sigma(msk, \cdot)$, which satisfies $\alpha(\sigma(msk, pk)) = pk$ for any $pk \in PK$.
- $\mathsf{KeyGen}(msk, id)$: take as input msk and $id \in I$, and output $\sigma(msk, \mathsf{IHF}(id))$.
- $\mathsf{Pub}(id, x, w)$: take as input $id \in I$, an element $x \in V_{pk}$ (where $pk = \mathsf{IHF}(id)$) and a witness $w \in W$ for x's membership in V_{pk}, and output $y = \mathsf{H}_{sk}(x)$ (where $\alpha(sk) = pk$). This is the *public evaluation algorithm.*
- $\mathsf{Priv}(sk, x)$: take as input a private key sk and an element $x \in X$, and output $y \in Y$. This is the *private evaluation algorithm.*

We require that an IB-HPS satisfies the following basic properties.

Correctness of Evaluation. For any master key pair (mpk, msk) produced by $\mathsf{Setup}(\kappa)$ and any $id \in I$ we have

$$\Pr\left[y \neq y' \middle| \begin{array}{l} x \leftarrow \mathsf{SampR}(\mathsf{IHF}(id)), y \leftarrow \mathsf{Pub}(id, x, w) \\ sk \leftarrow \sigma(msk, \mathsf{IHF}(id)), y' \leftarrow \mathsf{Priv}(sk, x), \end{array}\right] \leq \mathsf{negl}(\kappa)$$

Valid/Invalid Sample Indistinguishability. For any $id \in I$, a valid sample generated by SampR and an invalid sample generated by SampR^* should be computationally indistinguishable even given $sk \leftarrow \mathsf{KeyGen}(msk, id)$. The valid/invalid sample indistinguishability can be formally addressed by the Interactive Subset Membership (ISM) assumption. We say that the interactive subset membership assumption holds for $\mathbf{P}$ if for all PPT algorithms $\mathcal{A}$ its advantage is negligible in κ in the following game.

Setup: $\mathcal{CH}$ computes $(mpk, msk) \leftarrow \mathsf{Setup}(\kappa)$ and gives mpk to $\mathcal{A}$.
Phase 1: $\mathcal{A}$ adaptively queries $\mathcal{CH}$ with $id \in I$ and $\mathcal{CH}$ responds with $\sigma(msk, \mathsf{IHF}(id))$.
Challenge: $\mathcal{A}$ chooses an arbitrary $id^* \in I$ as the target identity. $\mathcal{CH}$ picks a random bit $b \in \{0, 1\}$, if $b = 0$, computes $x^* \leftarrow \mathsf{SampR}(\mathsf{IHF}(id^*))$, else computes $x^* \leftarrow \mathsf{SampR}^*(\mathsf{IHF}(id^*))$. $\mathcal{CH}$ gives x^* to $\mathcal{A}$.
Phase 2: $\mathcal{A}$ can continue to issue more private key queries adaptively, and $\mathcal{CH}$ responds the same way as it did in Phase 1.
Guess: $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and wins the game if $b' = b$.

We define $\mathcal{A}$'s advantage to be $\mathrm{Adv}_{\mathcal{A}}(\kappa) = |\Pr[b' = b] - 1/2|$.

Smoothness. To attain the data privacy for the resulting IBE schemes, an information theoretic property named smoothness is needed, which ensures that for any $id \in I$ and an invalid sample x with respect to $pk = \mathsf{IHF}(id)$, the distribution of $\mathsf{Priv}(sk, x)$ is almost uniform over Y in the view of any PPT adversary. This property has already been introduced in [7], and we re-define it using our fine-grained syntax in and without the presence of leakage as follows:

Definition 3 (Smoothness). *We say that an IB-HPS is smooth if, for any (mpk, msk) produced by $\mathsf{Setup}(\kappa)$, and any $id \in I$, we have:*

$$\mathbf{SD}((x, y, R), (x, y', R)) \leq \mathsf{negl}(\kappa)$$

where $x \leftarrow \mathsf{SampR}^*(\mathsf{IHF}(id))$, $y' \xleftarrow{R} U_Y$, y is obtained by $sk \leftarrow \mathsf{KeyGen}(msk, id)$ and computing $\mathsf{Priv}(sk, x)$, and R is the ensemble of all terms (master public parameters, private keys) given to the adversary except the challenge ciphertext and the leakage on sk. Particularly, we say that an IB-HPS is ℓ-leakage-resilient smooth if for any (possibly randomized and need not be efficient) function $f(\cdot)$ with ℓ-bit output, we have

$$\mathbf{SD}((x, y, f(sk), R), (x, y', f(sk), R)) \leq \mathsf{negl}(\kappa)$$

where x, y, y' and sk are sampled as above.

Anonymity. To attain the key privacy for the resulting IBE schemes, we need to set an additional information theoretic property named anonymity of IB-HPS. Essentially, this property ensures that given two distinct identities id_0 and

id_1, the distributions of their invalid samples x_0 and x_1, and the corresponding outputs y_0 and y_1 of Priv are indistinguishable. We define this property in and without the presence of leakage as follows:

Definition 4 (Anonymity). *We say that an IB-HPS is anonymous if, for any* (mpk, msk) *produced by* $\mathsf{Setup}(\kappa)$*, and any two distinct* $id_0, id_1 \in I$*, we have:*

$$\mathbf{SD}((x_0, y_0, R), (x_1, y_1, R)) \leq \mathsf{negl}(\kappa)$$

where $x_i \leftarrow \mathsf{SampR}^*(\mathsf{IHF}(id_i))$*,* y_i *is obtained by choosing* $sk_i \leftarrow \mathsf{KeyGen}(msk, id_i)$ *and computing* $y_i = \mathsf{Priv}(x_i, sk_i)$ *for* $i = \{0, 1\}$*, and* R *is the ensemble of all terms (master public parameters, private keys) given to the adversary except the the challenge ciphertext and the leakage on* sk_0 *and* sk_1*. Particularly, we say that an IB-HPS is* ℓ*-leakage-resilient anonymous if, for any (possibly randomized and need not be efficient) function* $f_0(\cdot)$ *and* $f_1(\cdot)$ *with* ℓ*-bit output, we have:*

$$\mathbf{SD}((x_0, y_0, f_0(sk_0), f_1(sk_1), R), (x_1, y_1, f_0(sk_0), f_1(sk_1), R)) \leq \mathsf{negl}(\kappa)$$

where (x_0, y_0, sk_0) *and* (x_1, y_1, sk_1) *are sampled as above.*

We now show how to convert a smooth and anonymous IB-HPS (Setup, KeyGen, Pub, Priv) into a leakage-resilient smooth and anonymous IB-HPS using an average-case randomness extractor $\mathsf{Ext} : Y \rightarrow \{0,1\}^v$ with seeds set $\{0,1\}^\mu$. We modify the algorithms SampR and SampR^* of $\mathbf{M}$ as follows:

- $\overline{\mathsf{SampR}}(pk)$: sample $\bar{x} \leftarrow \mathsf{SampR}(pk)$, pick a seed $s \xleftarrow{R} \{0,1\}^\mu$, and output $x = (\bar{x}, s)$.
- $\overline{\mathsf{SampR}^*}(pk)$: sample $\bar{x} \leftarrow \mathsf{SampR}^*(pk)$, pick a seed $s \xleftarrow{R} \{0,1\}^\mu$, and output $x = (\bar{x}, s)$.

We keep algorithms Setup and KeyGen unchanged, define:

- $\overline{\mathsf{Pub}}(id, x, w)$: parse $x = (\bar{x}, s)$, compute $\bar{y} = \mathsf{Pub}(id, \bar{x}, w)$, and output $y = \mathsf{Ext}(\bar{y}; s)$.
- $\overline{\mathsf{Priv}}(sk, x)$: parse $x = (\bar{x}, s)$, compute $\bar{y} = \mathsf{Priv}(id, \bar{x})$, and output $y = \mathsf{Ext}(\bar{y}; s)$.

The theorem below shows that the transformed IB-HPS (Setup, KeyGen, $\overline{\mathsf{Pub}}$, $\overline{\mathsf{Priv}}$) is leakage-resilient smooth and anonymous for appropriate parameters.

Theorem 1. *Assume that an IB-HPS is smooth and anonymous and* $|Y| = 2^m$*. Let* $\mathsf{Ext} : Y \rightarrow \{0,1\}^v$ *be an* $(m - \ell, \epsilon_{\mathsf{ext}})$ *average-case extractor for some* $\epsilon_{\mathsf{ext}} = \mathsf{negl}(\kappa)$*. Then the above transform produces an* ℓ*-leakage-resilient smooth and* ℓ*-leakage-resilient anonymous IB-HPS.*

Proof. Based on the smooth property of the underlying IB-HPS, we have that $\mathbf{SD}((\bar{x}, \bar{y}, R), (\bar{x}, \bar{y}', R)) \leq \mathsf{negl}(\kappa)$, and thus $\mathbf{SD}((x, \bar{y}, R), (x, \bar{y}', R)) \leq \mathsf{negl}(\kappa)$, which implies $\widetilde{\mathbf{H}}_\infty(\bar{y}|(x, R)) \approx \log_2 |Y| = m$. In the presence of leakage, an adversary has access to at most ℓ bits of leakage from the private key sk,

i.e., to a random variable $f(sk)$ with 2^ℓ values. By Lemma 1 we know that $\widetilde{\mathbf{H}}_\infty(\bar{y}|(x, f(sk), R) \geq \widetilde{\mathbf{H}}_\infty(\bar{y}|(x, R)) - \ell = m - \ell$, therefore according to the definition of a $(m-\ell, \epsilon)$ extractor, we have $\mathbf{SD}((x, y, f(sk), R), (x, y', f(sk), R)) \leq \epsilon_{\mathsf{ext}}$ where $y' \xleftarrow{R} U_Y$. Since ϵ_{ext} is also negligible in κ, the leakage-resilient smoothness of the IB-HPS immediately follows. This part of proof has been presented in [7].

Based on the anonymous property of the underlying IB-HPS, we have that $\mathbf{SD}((\bar{x}_0, \bar{y}_0, R), (\bar{x}_1, \bar{y}_1, R)) \leq \mathsf{negl}(\kappa)$, and thus $\mathbf{SD}((x_0, \bar{y}_0, R), (x_1, \bar{y}_1, R)) \leq \mathsf{negl}(\kappa)$. Applying the smooth property twice, we have

$$\mathbf{SD}((x_0, y_0, f_0(sk_0), f_1(sk_1), R), (x_0, y', f_0(sk_0), f_1(sk_1), R)) \leq \epsilon_{\mathsf{ext}} \quad (1)$$

$$\mathbf{SD}((x_1, y_1, f_0(sk_0), f_1(sk_1), R), (x_1, y', f_0(sk_0), f_1(sk_1), R)) \leq \epsilon_{\mathsf{ext}} \quad (2)$$

Since $\mathbf{SD}((x_0, \bar{y}_0, R), (x_1, \bar{y}_1, R)) \leq \mathsf{negl}(\kappa)$, then certainly we have

$$\mathbf{SD}((x_0, y', f_0(sk_0), f_1(sk_1), R), (x_1, y', f_0(sk_0), f_1(sk_1), R)) \leq \mathsf{negl}(\kappa) \quad (3)$$

$\mathbf{SD}((x_0, y_0, f_0(sk_0), f_1(sk_1), R), (x_1, y_1, f_0(sk_0), f_1(sk_1), R)) \leq \mathsf{negl}(\kappa) + 2\epsilon_{\mathsf{ext}}$ immediately follows by combining the inequalities 1, 2, 3. Since ϵ_{ext} is also negligible in κ, the desired leakage-resilient anonymity of the IB-HPS immediately follows. This proves the theorem. □

5 Leakage-Resilient Anonymous IBE from Anonymous IB-HPS

We now show how to construct leakage-resilient anonymous IBE schemes from anonymous IB-HPS. Our notion of leakage-resilience only allows leakage on single private key for each identity, but not on master secret key.

5.1 Construction of Leakage-Resilient Anonymous IBE

The construction of a leakage-resilient anonymous IBE from a leakage-resilient anonymous IB-HPS is almost immediate, by simply using the hashing value as a one-time-pad to encrypt a message. In particular, given an IB-HPS where the hashing value set Y has some group structure $(Y, +)$ (e.g. bit-strings with $\oplus$), we construct an IBE scheme with the same identity set I and message set $M = Y$. The algorithms Setup and KeyGen are identical to that of IB-HPS. The algorithms $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$ are constructed as follows:

- $\mathsf{Encrypt}(id, m)$: compute $pk = \mathsf{IHF}(id)$, generate $x = \mathsf{SampR}(pk; w)$, compute $y = \mathsf{Pub}(id, x, w)$, set $z = y + m$, and output $c = (x, z)$.
- $\mathsf{Decrypt}(c, sk)$: parse c as (x, z), compute $y = \mathsf{Priv}(x, sk)$, and output $m = z - y$.

Note that the algorithm SampR^* of the IB-HPS is not used in the construction, but will be used to argue security.

Theorem 2. *The above construction yields an ℓ-leakage-resilient CPA-secure IBE if the underlying IB-HPS is ℓ-leakage-resilient smooth.*

Proof. The proof of this theorem has been presented in [7]. □

Theorem 3. *The above construction yields an ℓ-leakage-resilient anonymous IBE if the underlying IB-HPS is ℓ-leakage-resilient anonymous.*

Proof. We proceed via a sequence of games.

Game 0: Define Game 0 as the standard anonymous game. In the challenge stage of Game 0, the challenger computes $c_b \leftarrow \mathsf{Encrypt}(id_b, m)$ which we expand as $c_b = (x_b, z_b)$ where

$$x_b = \mathsf{SampR}(id_b; w), y_b = \mathsf{Pub}(id_b, x_b, w), z_b = y_b + m$$

Game 1: Compared to Game 0, we modify the challenge stage by having the challenger generate the ciphertext $c_b = (x_b, z_b)$ using the private key sk of id_b:

$$x_b = \mathsf{SampR}(id_b), y'_b = \mathsf{Priv}(x_b, sk), z_b = y'_b + m$$

The difference between Game 0 an Game 1 is only the use of y'_b versus y_b. By the correctness of evaluation, $y_b \neq y'_b$ happens with negligible probability so Game 0 and Game 1 are (statistically) indistinguishable.
Game 2: Based on Game 1, we further modify the challenge stage by having the challenger generate the ciphertext $c_b = (x_b, z_b)$ as follows:

$$x_b = \mathsf{SampR}^*(id_b), y'_b = \mathsf{Priv}(x_b, sk), z_b = y'_b + m$$

We claim that Game 1 and Game 2 are computationally indistinguishable by the valid/invalid sample indistinguishability of the underlying IB-HPS. Note that, although the valid/invalid sample indistinguishability game does not explicitly embody leakage queries, it allows the adversary to learn all private keys. Therefore indistinguishability between Game 1 and Game 2 holds even if the adversary obtains the full information of private keys for the two target identities, and hence certainly holds when just given limited amount of leakage.

According to the ℓ-leakage-resilient anonymous property of IB-HPS, the advantage of any PPT adversary in Game 2 is negligible. Therefore the advantage of any PPT adversary in Game 0 is also negligible in κ, which concludes the Theorem 3. □

Remark 1. As implicitly noted in [7, Section 7], for an IB-HPS smoothness instantly implies anonymity when the algorithm SampR is independent of pk. This approach is used as a natural methodology to achieve anonymity for many PKE/IBE schemes. We will see that the instantiations presented in Section 6.2, Section 6.4 and Section 6.5 exactly follow this approach. However, there do exist anonymous IBE schemes falling outside this methodology, for example, Gentry IBE [26], as we will analyze in Section 6.3. Our paradigm of anonymous IB-HPS

serves as a good framework to explain all the currently known anonymous IBE schemes [14, 26, 11, 27] relying on hash proof techniques, either in the random oracle model or in the standard model. We also stress that smoothness does not always guarantee anonymity, for example, the IBE scheme [15] is derived from an IB-HPS based on the DBDH assumption which is smooth but not anonymous, as we will show in Section 6.1.

6 Instantiations

IB-HPS is known to exist based on a variety of assumptions [7]. We first describe a construction of IB-HPS which is smooth but not anonymous, then describe four constructions of IB-HPS which are smooth and anonymous. We note that the last three constructions have been presented in [7]. For completeness, we interpret them using our fine-grained paradigm.

6.1 Non-anonymous IB-HPS Based on the DBDH Assumption

We now describe an IB-HPS based on the DBDH assumption, which can be viewed as the backbone of the leakage-resilient IBE scheme presented in [15].

Let $\mathbf{M}$ be a subset membership problem based on the DBDH assumption. $\mathsf{Gen}(\kappa)$ runs bilinear group generator $\mathsf{GroupGen}(\kappa)$ to generate global public parameters $(e, \mathbb{G}, \mathbb{G}_T, p)$, picks five random generators g, $g_1 = g^a$, $g_2 = g^b$, $g_3 = g^c$, $g_4 = g^d$ from $\mathbb{G}$, outputs an instance description $\Gamma = (X, V, W, PK, \mathsf{R})$ of $\mathbf{M}$, where $X = \mathbb{G} \times \mathbb{G} \times \mathbb{G}_T$, $W = \mathbb{Z}_p$, $PK = \mathbb{G}$, $V_{pk} = \{(pk^w, g^w, e(g,g)^{abw}) \in X : w \in W\}$, $\mathsf{R}_{pk} = \{(x, w) \in X \times W : ((pk^w, g^w, e(g,g)^{abw}), w)\}$. SampR and SampR^* are defined as follows:

- $\mathsf{SampR}(pk)$: pick $w \xleftarrow{R} \mathbb{Z}_p$, output $x = (pk^w, g^w, e(g,g)^{abw}) \in V_{pk}$.
- $\mathsf{SampR}^*(pk)$: pick $w, w' \xleftarrow{R} \mathbb{Z}_p$ subjected to the condition $w \neq w'$, output $x = (pk^w, g^w, e(g,g)^{abw'}) \in X \backslash V_{pk}$.

Let $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$ be a corresponding projective hash family, where $SK = \mathbb{Z}_p \times \mathbb{G} \times \mathbb{G}$, $Y = \mathbb{G}_T$. For $sk = (sk_1, sk_2, sk_3)$ and $x = (x_1, x_2, x_3)$, we define H as $\mathsf{H}_{sk}(x) = e(x_1, sk_3)e(x_2, sk_2)x_3^{sk_1}$.

Let $\mathbf{P}$ be an IB-HPS for $\mathbf{M}$ associating $\mathbf{H}$, which consists of four algorithms as below:

- $\mathsf{Setup}(\kappa)$: run $\mathsf{Gen}(\kappa)$ to obtain $(g, g_1 = g^a, g_2 = g^b, g_3 = g^c, g_4 = g^d)$, pick $u_0, u_1, \ldots, u_n \xleftarrow{R} \mathbb{G}^*$, set $mpk = (g, u_0, u_1, \ldots, u_n, e(g,g)^{ab}, e(g,g)^{cd})$, $msk = (g^{ab}, g^{cd})$. The identity set I is $\mathbb{Z}_p$ and $\mathsf{IHF} : \mathbb{Z}_p \to \mathbb{G}$ is defined as $\mathsf{IHF}(id) = u_0 \prod_{i=1}^{n} u_i^{id_i}$ where id_i denotes the i-th bit of identity id (known as Waters hash). $\sigma(msk, pk)$ is constructed as: parse msk as (msk_1, msk_2), pick $t, r \xleftarrow{R} \mathbb{Z}_p$, output $(t, msk_1 msk_2^{-t} pk^r, g^{-r})$.
- $\mathsf{KeyGen}(msk, id)$: compute $pk = \mathsf{IHF}(id)$, output $sk = \sigma(msk, pk)$.

- $\mathsf{Pub}(id, x, w)$: compute $pk = \mathsf{IHF}(id)$, for $x = (pk^w, g^w, e(g,g)^{abw})$ output $y = e(g,g)^{cdw}$.
- $\mathsf{Priv}(sk, x)$: parse sk as (sk_1, sk_2, sk_3) and x as (x_1, x_2, x_3), output $y = e(x_1, sk_3)e(x_2, sk_2)x_3^{sk_1}$.

It is obvious that one can use the bilinear map as a tool to test if an invalid sample x is generated by SampR^* with respect to pk. Therefore the above IB-HPS is smooth but not anonymous. This provides us an evidence that for IB-HPS smoothness does not guarantee anonymity.

6.2 Anonymous IB-HPS Based on the DBDH Assumption

We now describe an IB-HPS based on the DBDH assumption, which can be viewed as the backbone of the leakage-resilient IBE scheme presented in [14].

Let $\mathbf{M}$ be a subset membership problem based on the DBDH assumption. $\mathsf{Gen}(\kappa)$ runs bilinear group generator $\mathsf{GroupGen}(\kappa)$ to generate global public parameters $(e, \mathbb{G}, \mathbb{G}_T, p)$, picks three random generators g, $g_1 = g^a$, g_2 from $\mathbb{G}$, outputs an instance description $\Gamma = (X, V, W, PK, \mathsf{R})$ of $\mathbf{M}$, where $X = \mathbb{G} \times \mathbb{G}_T$, $W = \mathbb{Z}_p$, $PK = \mathbb{G}$, $V_{pk} = \{(g^w, e(g_1, g_2)^w) \in X : w \in W\}$, $\mathsf{R}_{pk} = \{(x, w) \in X \times W : ((g^w, e(g_1, g_2)^w), w)\}$. Note that V_{pk} and R_{pk} for all $pk \in PK$ are same in this case. For simplicity we write V and R for short. SampR and SampR^* are defined as follows:

- $\mathsf{SampR}(pk)$: pick $w \xleftarrow{R} \mathbb{Z}_p$, output $x = (g^w, e(g_1, g_2)^w) \in V$.
- $\mathsf{SampR}^*(pk)$: pick $w, w' \xleftarrow{R} \mathbb{Z}_p$ subjected to the condition $w \neq w'$, output $x = (g^w, e(g_1, g_2)^{w'}) \in X \backslash V$.

Let $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$ be a corresponding projective hash family, where $SK = \mathbb{Z}_p \times \mathbb{G}$, $Y = \mathbb{G}_T$. For $sk = (sk_1, sk_2)$ and $x = (x_1, x_2)$, H is defined as $\mathsf{H}_{sk}(x) = e(x_1, sk_2)x_2^{sk_1}$.

Let $\mathbf{P}$ be an IB-HPS for $\mathbf{M}$ associating $\mathbf{H}$, which consists of four algorithms as below:

- $\mathsf{Setup}(\kappa)$: run $\mathsf{Gen}(\kappa)$ to generate g, $g_1 = g^a$, g_2, set $mpk = (g, g_1 = g^a, g_2)$, $msk = a$. The identity set I is $\{0,1\}^*$ and IHF is a function from $\{0,1\}^*$ to $\mathbb{G}$. $\sigma(msk, pk)$ is constructed as: pick $t \xleftarrow{R} \mathbb{Z}_p$, output $(t, (pk \cdot g_2^{-t})^{msk})$.
- $\mathsf{KeyGen}(msk, id)$: compute $pk = \mathsf{IHF}(id)$, output $sk = \sigma(msk, pk)$.
- $\mathsf{Pub}(id, x, w)$: compute $pk = \mathsf{IHF}(id)$, for $x = (g^w, e(g_1, g_2)^w)$ output $y = e(pk, g_1)^w$.
- $\mathsf{Priv}(sk, x)$: parse sk as (sk_1, sk_2) and x as (x_1, x_2), output $y = e(x_1, sk_2)x_2^{sk_1}$.

As shown in [14], the above IB-HPS is smooth and anonymous.

6.3 Anonymous IB-HPS Based on the DTABDHE Assumption

We now describe an IB-HPS based on the DTABDHE assumption [26], which can be viewed as the backbone of Gentry's IBE [26].

Let $\mathbf{M}$ be a subset membership problem based on the DTABDHE assumption. The algorithm $\mathsf{Gen}(\kappa)$ runs bilinear group generator $\mathsf{GroupGen}(\kappa)$ to generate global public parameters $(e, \mathbb{G}, \mathbb{G}_T, p)$, picks three random generators g, $g_1 = g^a$, h from $\mathbb{G}$, outputs an instance description $\Gamma = (X, V, W, PK, \mathsf{R})$ of $\mathbf{M}$, where $X = \mathbb{G} \times \mathbb{G}_T$, $W = \mathbb{Z}_p$, $PK = \mathbb{Z}_p$, $V_{pk} = \{(g_1^w g^{-w \cdot pk}, e(g,g)^w) \in X : w \in W\}$, $\mathsf{R}_{pk} = \{(x, w) \in X \times W : ((g_1^w g^{-w \cdot pk}, e(g,g)^w), w)\}$. SampR and SampR^* are defined as follows:

- $\mathsf{SampR}(pk)$: pick $w \xleftarrow{R} \mathbb{Z}_p$, output $x = (g_1^w g^{-w \cdot pk}, e(g,g)^w) \in V_{pk}$.
- $\mathsf{SampR}^*(pk)$: pick $w, w' \xleftarrow{R} \mathbb{Z}_p$ subjected to the condition $w \neq w'$, output $x = (g_1^w g^{-w \cdot pk}, e(g,g)^{w'}) \in X \backslash V_{pk}$.

Let $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$ be a corresponding projective hash family, where $SK = \mathbb{Z}_p \times \mathbb{G}$, $Y = \mathbb{G}_T$.

Let $\mathbf{P}$ be an IB-HPS for $\mathbf{M}$ associating $\mathbf{H}$, which consists of four algorithms as below:

- $\mathsf{Setup}(\kappa)$: run $\mathsf{Gen}(\kappa)$ to generate $(g, g_1 = g^a, h)$, set $mpk = (g, g_1, h)$, $msk = a$. The identity set I is $\mathbb{Z}_p$ and IHF is an identity function. $\sigma(msk, pk)$ is constructed as: pick $t \in \mathbb{Z}_p$, output $(t, (hg^{-t})^{1/(msk - pk)})$.
- $\mathsf{KeyGen}(msk, id)$: set $pk = id$, output $\sigma(msk, pk)$.
- $\mathsf{Pub}(id, x, w)$: set $pk = id$, for $x = (g_1^w g^{-w \cdot pk}, e(g,g)^w)$ output $y = e(g,h)^{-w}$.
- $\mathsf{Priv}(sk, x)$: parse $sk = (sk_1, sk_2)$ and $x = (x_1, x_2)$, output $y = e(x_1, sk_2) x_2^{sk_1}$.

As shown in [26], the above IB-HPS is smoothness and anonymous.

6.4 Anonymous IB-HPS Based on the Quadratic Residues Assumption

We now describe an IB-HPS based on the QR assumption, which can be viewed as the backbone of the IBE scheme presented in [11].

For a positive integer N, let $J(N)$ denote the set $J(N) = \{x \in \mathbb{Z}_N : \left(\frac{x}{N}\right) = 1\}$ where $\left(\frac{x}{N}\right)$ denotes the Jacobi symbol of x in $\mathbb{Z}_n$. Let $QR(N) \subseteq J(N)$ be the set of quadratic residues modulo N.

Let $\mathbf{M}$ be a subset membership problem based on the QR assumption. The algorithm $\mathsf{Gen}(\kappa)$ runs prime generator $\mathsf{PrimeGen}(\kappa)$ to generate two primes (p, q) and set the global public parameter to be $(N, u, \mathcal{Q})$ (where $N = pq$ and $u \xleftarrow{R} J(N) \backslash QR(N)$ and $\mathcal{Q}$ is the algorithm defined in [7, Appendix C]), outputs an instance description $\Gamma = (X, V, W, PK, \mathsf{R})$ of $\mathbf{M}$, where $X = J(N) \times \{\pm 1\}$, $W = \mathbb{Z}_N$, $PK = J(N)$, $V_{pk} = \left\{(w^2, b) \in X : w \in W, b = \left(\frac{\tau(w)}{N}\right)\right\}$, $\mathsf{R}_{pk} = \{(x, w) \in X \times W : ((w^2, \left(\frac{\tau(w)}{N}\right)), w)\}$. Note that V_{pk} and R_{pk} for all $pk \in PK$ are same in this case. For simplicity we write V and R for short. SampR and SampR^* are defined as follows:

- $\mathsf{SampR}(pk)$: pick $w \xleftarrow{R} \mathbb{Z}_p$, set $x_1 = w^2$, run $\mathcal{Q}(N, u, 1, x_1)$ to obtain τ and compute $x_2 = \left(\frac{\tau(w)}{N}\right)$, output $x = (x_1, x_2) \in X$.

– $\mathsf{SampR}^*(pk)$: pick $x_1 \xleftarrow{R} J(N)\backslash QR(N)$, $x_2 \xleftarrow{R} \{\pm 1\}$, output $x = (x_1, x_2) \in X\backslash V$.

Let $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$ be a corresponding projective hash family, where $SK = \mathbb{Z}_N$, $Y = \{\pm 1\}$. For $sk = r$ and $x = (x_1, x_2)$, $\mathsf{H}_{sk}(x) = y$ is defined as when $r^2 = pk$ output $\left(\frac{f(r)}{N}\right)$ else output $x_2\left(\frac{\bar{f}(r)}{N}\right)$, where f, $\bar{f}$ are the polynomials output by $\mathcal{Q}(N, u, R, x_1)$.

Let $\mathbf{P}$ be an IB-HPS for $\mathbf{M}$ associating $\mathbf{H}$, which consists of four algorithms as below:

– $\mathsf{Setup}(\kappa)$: run $\mathsf{Gen}(\kappa)$ to obtain two primes (p, q), compute $N = pq$, pick $u \xleftarrow{R} J(N)\backslash QR(N)$, set $mpk = (N, u)$, $msk = (p, q)$. The identity set I is $\{0,1\}^*$ and IHF is a function from $\{0,1\}^*$ to $J(N)$. $\sigma(msk, pk)$ is constructed as: let $a \in \{0,1\}$ be the unique choice for which $u^a pk \in QR(N)$, let $\{r_1, r_2, r_3, r_4\}$ be the labeling of the four square-roots of $u^a pk$ so that $r_1 < r_2 < r_3 < r_4$ (in $\mathbb{Z}$) and $r_1 = -r_4$, $r_2 = -r_3$, choose $r \xleftarrow{R} \{r_1, r_2\}$.
– $\mathsf{KeyGen}(msk, id)$: compute $pk = \mathsf{IHF}(id)$, output $sk = \alpha(msk, pk)$.
– $\mathsf{Pub}(id, x, w)$: compute $pk = \mathsf{IHF}(id)$, for $x = (x_1, x_2)$, run $\mathcal{Q}(N, u, pk, x_1)$ to obtain a polynomial g, output $y = \left(\frac{g(w)}{N}\right)$.
– $\mathsf{Priv}(sk, x)$: suppose $sk = r$ for id and $x = (x_1, x_2)$, compute $pk = \mathsf{IHF}(id)$, run $\mathcal{Q}(N, u, pk, x_1)$ to obtain polynomials f, $\bar{f}$. If $r^2 = pk$ output $y = \left(\frac{f(r)}{N}\right)$, else output $y = x_2\left(\frac{\bar{f}(r)}{N}\right)$.

As shown in [11], the above IB-HPS is smooth and anonymous.

6.5 Anonymous IB-HPS Based on the DLWE Assumption

We now describe an anonymous IB-HPS based on the DLWE assumption, which can be viewed as the backbone of the IBE scheme presented in [27].

Let $\mathbf{M}$ be a subset membership problem based on the LWE assumption. $\mathsf{Gen}(\kappa)$ generates global public parameters $\mathbf{A}$ with trapdoor $\mathbf{S}$ and a function f indexed by $\mathbf{A}$, outputs an instance description $\Gamma = (X, V, W, PK, \mathsf{R})$ of $\mathbf{M}$, where $X = \mathbb{Z}_q^n \times \mathbb{Z}_q$, $W = \mathbb{Z}_q^n$, $PK = \mathbb{Z}_q^n$, $V = \{(\mathbf{A}^T\mathbf{s} + \mathbf{x}, v) \in X : \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{x} \leftarrow \chi^m, v \leftarrow \mathbb{Z}_p\}$, $\mathsf{R} = \{((\mathbf{p}, \mathbf{s}), v) \in X \times W : ((\mathbf{A}^T\mathbf{s} + \mathbf{x}, v), \mathbf{s})\}$. Note that V_{pk} and R_{pk} for all $pk \in PK$ are same in this case. For simplicity we write V and R for short. SampR and SampR^* are defined as follows:

– $\mathsf{SampR}(pk)$: pick $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{x} \xleftarrow{R} \chi^m$, $v \xleftarrow{R} \mathbb{Z}_q$, compute $\mathbf{p} = \mathbf{A}^T\mathbf{s} + \mathbf{x} \in \mathbb{Z}_q^m$, output $x = (\mathbf{p}, v)$.
– $\mathsf{SampR}^*(pk)$: pick $\mathbf{p} \xleftarrow{R} \mathbb{Z}_q^m$ and $v \xleftarrow{R} \mathbb{Z}_q$, output $x = (\mathbf{p}, v)$.

Let $\mathbf{H} = (\mathsf{H}, SK, PK, X, V, Y, \alpha)$ be a corresponding projective hash family, where $SK = \mathbb{Z}_m$, $Y = \mathbb{Z}_2 \times \mathbb{Z}_q$. For $sk = \mathbf{e}$ and $x = (\mathbf{p}, v)$, we define $\mathsf{H}_{sk}(x) = y$ as $y = 1$ if $|v - pk^T\mathbf{s}| \leq \frac{q-1}{4}$ and $y = 0$ otherwise.

Let $\mathbf{P}$ be an IB-HPS for $\mathbf{M}$ associating $\mathbf{H}$, which consists of four algorithms as below:

$\mathsf{Setup}(\kappa)$: run $\mathsf{Gen}(\kappa)$ to generate $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ a with trapdoor $\mathbf{S} \subset \Lambda^{\perp}(\mathbf{A}, q)$ according to the trapdoor generation algorithm of [27]. The mpk is $\mathbf{A}$ and the msk is $\mathbf{S}$. The identity set I is $\{0,1\}^*$ and the identity mapping function is $\mathsf{IHF} : \{0,1\}^* \to \mathbb{Z}_q^n$. The inversion of α is constructed as $\sigma(\mathbf{S}, pk) = f_{\mathbf{A}}^{-1}(pk)$ using the preimage sampler with $\mathbf{S}$.
$\mathsf{KeyGen}(msk, id)$: compute $pk = \mathsf{IHF}(id)$, generate a private key $sk = \sigma(\mathbf{S}, pk)$.
$\mathsf{Pub}(id, x, w)$: compute $pk = \mathsf{IHF}(id)$, parse $x = (\mathbf{p}, v)$, if $|v - pk^T\mathbf{s}| \leq \frac{q-1}{4}$ then set $y = 1$ else set $y = 0$.
$\mathsf{Priv}(sk, x)$: parse $sk = \mathbf{e}$, $x = (\mathbf{p}, v)$, if $|v - \mathbf{e}^T\mathbf{p}| \leq \frac{q-1}{4}$ then output $y = 1$ else output $y = 0$.

As shown in [27], the above IB-HPS is smooth and anonymous.

7 Conclusion

In this paper, we reformulated the paradigm of IB-HPS in a fine-grained manner, and formally introduced a new property named anonymity for it. We gave four different constructions of anonymous IB-HPS based on a variety of assumptions. As one important application, we defined the leakage-resilient anonymity of IBE schemes, and showed how to construct leakage-resilient anonymous IBE schemes through anonymous IB-HPS in a generic way. As another promising application, we defined leakage-resilient anonymity of PEKS schemes, and showed how to construct leakage-resilient anonymous PEKS schemes via anonymous IB-HPS (using the BDOP IBE-to-PEKS transform [2] as the stepping stone). Due to space limit, we include this part in the full version of this paper.

Acknowledgment. The authors are grateful to the anonymous ProvSec 2012 reviewers for many helpful comments, and to Mingwu Zhang for his invaluable help in improving this paper. The first and third authors are supported by IIE's Cryptography Research Project, the Strategic Priority Research Program of CAS under Grant No. XDA06010701, the National 973 Program of China under Grant No. 2011CB302400, and the National Natural Science Foundation of China under Grant No. 60970152. The second and the fourth authors are supported by the National Natural Science Foundation of China under Grant Nos. 61033014, 60970110, 60972034 and Asia 3 Foresight Program under Grant No. 61161140320.

References

[1] Abadi, M., Rogaway, P.: Reconciling two views of cryptography (the computational soundness of formal encryption). J. Cryptology 15(2), 103–127 (2002)
[2] 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)

[3] Abdalla, M., Bellare, M., Neven, G.: Robust Encryption. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 480–497. Springer, Heidelberg (2010)
[4] 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)
[5] Agrawal, S., Boneh, D., Boyen, X.: Efficient Lattice (H)IBE in the Standard Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)
[6] Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous Hardcore Bits and Cryptography against Memory Attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
[7] Alwen, J., Dodis, Y., Naor, M., Segev, G., Walfish, S., Wichs, D.: Public-Key Encryption in the Bounded-Retrieval Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 113–134. Springer, Heidelberg (2010)
[8] Alwen, J., Dodis, Y., Wichs, D.: Leakage-Resilient Public-Key Cryptography in the Bounded-Retrieval Model. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 36–54. Springer, Heidelberg (2009)
[9] Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-Privacy in Public-Key Encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (2001)
[10] Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. SIAM Journal on Computation 32, 586–615 (2003)
[11] Boneh, D., Gentry, C., Hamburg, M.: Space-efficient identity based encryption without pairings. In: 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), pp. 647–657. IEEE Computer Society (2007)
[12] Boyen, X., Waters, B.: Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
[13] Canetti, R., Dodis, Y., Halevi, S., Kushilevitz, E., Sahai, A.: Exposure-Resilient Functions and All-or-Nothing Transforms. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 453–469. Springer, Heidelberg (2000)
[14] Chen, Y., Luo, S., Chen, Z.: A New Leakage-Resilient IBE Scheme in the Relative Leakage Model. In: Li, Y. (ed.) DBSec 2011. LNCS, vol. 6818, pp. 263–270. Springer, Heidelberg (2011)
[15] Chow, S.S.M., Dodis, Y., Rouselakis, Y., Waters, B.: Practical leakage-resilient identity-based encryption from simple assumptions. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, pp. 152–161. ACM (2010)
[16] Cocks, C.: An Identity Based Encryption Scheme Based on Quadratic Residues. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (2001)
[17] Cramer, R., Shoup, V.: A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
[18] 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)
[19] Di Crescenzo, G., Lipton, R.J., Walfish, S.: Perfectly Secure Password Protocols in the Bounded Retrieval Model. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 225–244. Springer, Heidelberg (2006)

[20] Desai, A.: The Security of All-or-Nothing Encryption: Protecting against Exhaustive Key Search. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 359–375. Springer, Heidelberg (2000)
[21] 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)
[22] Dziembowski, S.: Intrusion-Resilience Via the Bounded-Storage Model. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 207–224. Springer, Heidelberg (2006)
[23] Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: 49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, pp. 293–302. IEEE Computer Society (2008)
[24] Faust, S., Kiltz, E., Pietrzak, K., Rothblum, G.N.: Leakage-Resilient Signatures. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 343–360. Springer, Heidelberg (2010)
[25] Fischlin, M.: Pseudorandom Function Tribe Ensembles Based on One-Way Permutations: Improvements and Applications. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 432–445. Springer, Heidelberg (1999)
[26] Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
[27] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC, pp. 197–206. ACM (2008)
[28] Halevi, S., Lin, H.: After-the-Fact Leakage in Public-Key Encryption. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 107–124. Springer, Heidelberg (2011)
[29] Katz, J., Vaikuntanathan, V.: Signature Schemes with Bounded Leakage Resilience. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 703–720. Springer, Heidelberg (2009)
[30] Lewko, A., Rouselakis, Y., Waters, B.: Achieving Leakage Resilience through Dual System Encryption. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 70–88. Springer, Heidelberg (2011)
[31] Micali, S., Reyzin, L.: Physically Observable Cryptography (Extended Abstract). In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 278–296. Springer, Heidelberg (2004)
[32] 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)
[33] Shaltiel, R.: Recent developments in explicit constructions of extractors. Bulletin of the EATCS 77, 67–95 (2002)
[34] 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)
[35] Zhang, R., Hanaoka, G., Imai, H.: Orthogonality between Key Privacy and Data Privacy, Revisited. In: Pei, D., Yung, M., Lin, D., Wu, C. (eds.) Inscrypt 2007. LNCS, vol. 4990, pp. 313–327. Springer, Heidelberg (2008)

Efficient Escrow-Free Identity-Based Signature

Yunmei Zhang[1], Joseph K. Liu[2], Xinyi Huang[3], Man Ho Au[1], and Willy Susilo[1]

[1] Centre for Computer and Information Security Research(CCISR)
School of Computer Science and Software Engineering
University of Wollongong, Australia
yz841@uowmail.edu.au, {aau,wsusilo}@uow.edu.au
[2] Cryptography and Security Department
Institute for Infocomm Research, Singapore
ksliu@i2r.a-star.edu.sg
[3] School of Mathematics and Computer Science
Fujian Normal University, Fuzhou
Fujian, China
xyhuang81@gmail.com

Abstract. The notion of identity-based signature scheme (IBS) has been proven useful in some scenarios where relying on the validity of the certificates is impractical. Nevertheless, one remaining inherent problem that hinders the adoption of this cryptographic primitive in practice is due to the key escrow problem, where the private key generator (PKG) can always impersonate the user in the system. In 2010, Yuen et al. proposed the notion of IBS that does not suffer from the key escrow problem. Nevertheless, their approach relies on the judge who will later blame the malicious PKG when such a dispute occurs, assuming that the PKG is willing to collaborate. Although the approach is attractive, but unfortunately it is impractical since the malicious PKG may just refuse to collaborate when such an incident happens. In this paper, we propose a new escrow-free IBS, which enjoys three main advantages, namely key escrow free, practical and very efficient. We present a generic intuition as well as an efficient instantiation. In our approach, there is no judge involvement required, as the public can determine the malicious behaviour of PKG when such an incident happens. Further, the signature size of our instantiation is only two group elements, which outperforms the existing constructions in the literature.

Keywords: identity-based signature, key escrow, efficiency, practicality.

1 Introduction

Due to the practical deployment of traditional public key infrastructure (PKI), Shamir [14] introduced the concept of identity-based (ID-based) cryptosystem with the main aim to eliminate the necessity to verify the validity of the certificates. In an ID-based cryptosystem, the public key of a user can be any arbitrary string, such as an email address. An entity called the private key generator

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 161–174, 2012.

(PKG) then computes the private keys from a master secret for the users. This property avoids the use of certificates and associates an implicit public key (user identity) to each user within the system. One only needs to know the recipient's identity in order to send an encrypted message to him. It avoids the complicated and costly certificate (chain) verification for the authentication purpose. In the case of signature, verification takes only the identity together with the message and signature pair as input and executes the algorithm directly. (In contrast to traditional PKI, whereas an additional certification verification algorithm is needed. That is equivalent to the process of two signatures verification.) Identity-based signature is a good solution to shorten the time for the overall signature verification and the size of the signature.

However, as the PKG generates and holds the secret key for all users, a complete trust must be placed on the PKG. Nonetheless, this may not be desirable in a real world scenario, where a malicious PKG can sell users' keys or pretend any user to sign messages or decrypt ciphertexts without being confronted in a court of law. This problem is referred to as the *key escrow problem*, which is an inherent problem in the ID-based cryptosystem. This problem has been seen as the main stumbling block of the adoption of ID-based cyptosystem in practice, especially in the scenario where having a complete trust on the PKG is unrealistic.

Some researchers proposed other crytposystems to solve the key escrow problem. They use a combination of identity-based cryptosystem and public-key infrastructure. Certificateless cryptosystem [1], certificate-based cryptosystem [7], self-certificated cryptosystem [8] and self-generated-certificate public key cryptosystem [12] are some examples of such solutions. In these systems, a user possesses a user public key and a user secret key, together with his identity-based secret key computed by the PKG. The user secret key protects the user from the key escrow problem. However, these systems are no longer identity-based. That is, the encryptor or the verifier has to know the user public key in addition to the user identity. In other words, the original advantage of identity-based cryptography has been lost.

For those *pure* identity-based cryptosystems, Boneh and Franklin [4] proposed using multiple PKGs to solve the problem of key escrow. Their idea is to let the master secret key jointly computed by a number of PKGs, such that no single PKG has the knowledge of it. However, this approach only partially solves the problem. It also requires an extra infrastructure and communication overhead. A user needs to run the key extraction protocol with different PKGs which is inefficient and inconvenient. Maintaining multiple PKGs is also impractical for a commercially used infrastructure.

Goyal [9] proposed the concept of accountable authority identity-based encryption (A-IBE) to reduce the trust in the PKG. The PKG helps the user to compute his identity-based secret key without knowing it. If the PKG computes another set of secret key by himself and reveals it to other parties, this key will be different from the user's original secret key. Therefore the PKG can be caught when revealing the secret key as the user's original secret key is the evidence.

However, malicious PKG is still able to sell a signed message or decrypted ciphertext without being caught. Goyal *et al.* [10] later enhanced the model by proposing the concept of a black-box A-IBE. In the enhanced model, if a PKG sells a decoder box which can decrypt ciphertexts, he will be caught in a trace algorithm.

The above solutions only deal with identity-based encryption. On the orthogonal direction, Yuen *et al.* [16] proposed an escrow-free identity-based signature scheme, which makes the notion of ID-signature very interesting. Their scheme requires a judge to blame the malicious PKG. It also requires the PKG to provide some secret information to the judge. If the PKG does not cooperate, the judge cannot output any evidence to blame the PKG. It seems the logic is quite impractical, since if the PKG is malicious, most likely it will not cooperate with the judge.

1.1 Contribution

In this paper, we propose a new escrow-free identity-based signature scheme. It is presented in twofold. We first provide a generic intuition, followed by an efficient instantiation. Our scheme enjoys the following advantages:

1. It is *key escrow free*. That is, any malicious PKG who pretends to be any user[1] to sign a message can be detected.
2. Different from [16], our scheme does not require any judge. Any verifier can detect the malicious behaviour of the PKG given a signature generated by a user and another signature generated by the PKG, who pretends to be that user. Hence, it is indeed *very practical*[2].
3. In terms of *efficiency*, our signature size contains only two group elements, while the scheme in [16] requires at least three group elements. Hence, our scheme outperforms the other existing schemes in the literature.

Organization. The rest of this paper is organized as follows. In Sec. 2, we review the syntax of an identity-based signature and present a formal definition for escrow-free identity-based signatures (or EF-IBS, for short). We present our construction in Sec. 3, followed by security analysis. In Sec. 4, we demonstrate

[1] We should stress that the "framed user" in this notion is any user who has extracted his/her private key via the extraction protocol. The definition will not capture any bogus user created by the PKG, while this user may not even exist in the system. Therefore, the protection against the key escrow is provided to users who have extrated their private keys. In practice, the user who has conducted the extraction protocol may receive a physical evidence, which can be used to prove to public in the case of PKG misbehaviour. Nevertheless, this is outside the system and we do not deal with this in the description of our scheme.

[2] Here we use the term practical to describe systems whose functionalities depends on realistic assumptions while the term efficient is reserved to describe systems with high efficiency.

the practicality of our scheme by showing various timing of the implementation. Comparison with existing identity-based signature schemes are shown in Section 5. We conclude our paper in Sec. 6.

2 Syntax

An EF-IBS is a tuple of five algorithms/protocols, namely, Gen, Ext, Sign, Verify, Blame. The first four algorithms are the same as a regular IBS except Ext is now an interactive protocol between the PKG and the user. We introduce a public algorithm called Blame which allows the public to determine if the PKG has created a signature on behalf of an honest user.

- $(\mathsf{param}, \mathsf{msk}) \leftarrow \mathsf{Gen}(1^\lambda)$: On input security parameter λ, this algorithm outputs the public parameter param for the system as well as the master secret key msk for the PKG. We assume param is an implicit input to all the algorithms/protocols below.
- $(d_{\mathsf{ID}}) \leftarrow \mathsf{Ext}_{\mathsf{User}}(\mathsf{ID}) \Longleftrightarrow \mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$: This is a pair of interactive algorithms $\mathsf{Ext}_{\mathsf{User}}$ and $\mathsf{Ext}_{\mathsf{PKG}}$ between the user and the PKG. The identity ID of the user is a common input to both parties. The PKG has an additional input msk. Upon successful completion of the protocol, the user obtains a secret key d_{ID} with respect to the identity ID.
- $(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$: This algorithm outputs a signature σ on message m with respect to identity ID.
- $(1/0) \leftarrow \mathsf{Verify}(\mathsf{ID}, \sigma, m)$: This algorithm verifies a signature σ on message m with respect to identity ID.
- $(1/0) \leftarrow \mathsf{Blame}(\mathsf{ID}, \sigma, m, \sigma', m')$: Given a message-signature pair (m, σ) from an honest signer with identity ID, this algorithm outputs 1 if and only if (m', σ') is created by the PKG and $m \neq m'$.[3]

Correctness. As in IBS, EF-IBS should satisfy correctness. That is, signatures signed by honest signers are verified to be valid.

2.1 Security Requirements

A EF-IBS should possess the standard requirement for signatures as *unforgeability* together with additional requirements described below.

We use the term *non-slanderability* to describe the requirement that the malicious PKG should not be able to create a signature on an honest user's behalf without being detected by the algorithm Blame. On the other hand, we use the term *non-framability* to describe the requirement that a malicious user should

[3] The condition $m \neq m'$ was added because in case the signature scheme is not strongly unforgeable, it might be feasible to construct another message-signature pair (m, σ^*) given a pair (m, σ).

not be able to create two signatures such that the Blame algorithm would output 1. The former protects the user and discourages the PKG from signing on the user's behalf. The latter protects the PKG so that it will not be falsely held accountable for the user-created signatures. We formalise the security requirements with the following games adapted from [16].

Game Unforgeability. The following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ formally captures the requirement of existential unforgeability against adaptive chosen ID and message attacks.

Setup $\mathcal{C}$ invokes $\mathsf{Gen}(1^k)$ and obtains $(\mathsf{param}, \mathsf{msk})$. param is given to $\mathcal{A}$.
Query $\mathcal{A}$ is allowed to make the following queries:
- Extraction Query. $\mathcal{A}$ submits an identity ID and engages $\mathcal{C}$ as a user. $\mathcal{C}$ runs $\mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$.
- Signature Query. $\mathcal{A}$ submits a message m and an identity ID. If $\mathcal{C}$ does not has a secret key d_{ID} with respect to ID, it creates one by invoking $(d_{\mathsf{ID}}) \leftarrow \mathsf{Ext}_{\mathsf{User}}(\mathsf{ID}) \Longleftrightarrow \mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$. Next, it computes $(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$. σ is returned to $\mathcal{A}$.

Output $\mathcal{A}$ submits $(\sigma^*, \mathsf{ID}^*, m^*)$ and wins if and only if
1. $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma^*, m^*)$.
2. $\mathcal{A}$ has not submitted a Signature Query with input (m^*, ID^*).
3. $\mathcal{A}$ has not submitted an Extraction Query with input ID^*.

Definition 1 (Unforgeability). *A scheme is unforgeable if no PPT adversary wins the above game with non-negligible probability.*

Game Non-slanderability. The following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ formally captures the requirement of non-slanderability.

Setup $\mathcal{C}$ invokes $\mathsf{Gen}(1^k)$ and obtains $(\mathsf{param}, \mathsf{msk})$. Both param and msk are given to $\mathcal{A}$.
Query $\mathcal{A}$ is allowed to made the following queries:
- PKG-Extraction Query. $\mathcal{A}$ submits an identity ID and engages $\mathcal{C}$. $\mathcal{C}$ plays the role of a user and runs $\mathsf{Ext}_{\mathsf{User}}(\mathsf{ID})$ to obtain d_{ID}.
- Signature Query. $\mathcal{A}$ submits a message m and an identity ID that has been submitted to PKG-Extraction Query. $\mathcal{C}$ computes $(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$. σ is returned to $\mathcal{A}$.

Output $\mathcal{A}$ submits $(\sigma^*, \mathsf{ID}^*, m^*)$ and wins if and only if
1. $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma^*, m^*)$.
2. $\mathcal{A}$ has submitted a Signature Query with input (m, ID^*) and obtains σ.
3. $0 \leftarrow \mathsf{Blame}(\mathsf{ID}^*, \sigma, m, \sigma^*, m^*)$ and $m \neq m^*$.

Definition 2 (Non-slanderability). *A scheme is non-slanderable if no PPT adversary wins the above game with non-negligible probability.*

Game Non-frameability. The following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ formally captures the requirement of non-frameability.

Setup $\mathcal{C}$ invokes $\mathsf{Gen}(1^k)$ and obtains $(\mathsf{param}, \mathsf{msk})$. param is given to $\mathcal{A}$.
Query $\mathcal{A}$ is allowed to made the following queries:

- Extraction Query. $\mathcal{A}$ submits an identity ID and engages $\mathcal{C}$ as a user. $\mathcal{C}$ runs $\mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$.
- Signature Query. $\mathcal{A}$ submits a message m and an identity ID. If $\mathcal{C}$ does not has a secret key d_{ID} with respect to ID, it creates one by invoking $(d_{\mathsf{ID}}) \leftarrow \mathsf{Ext}_{\mathsf{User}}(\mathsf{ID}) \Longleftrightarrow \mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$. Next, it computes $(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$. σ is returned to $\mathcal{A}$.

Output $\mathcal{A}$ submits $(\mathsf{ID}^*, \sigma^*, m^*, \sigma', m')$ and wins if and only if

1. $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma^*, m^*)$, $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma', m')$.
2. $m^* \neq m'$ and $\mathcal{A}$ has never submitted signature query with input $(\cdot, \mathsf{ID}^*)$.
3. $1 \leftarrow \mathsf{Blame}(\mathsf{ID}^*, \sigma', m', \sigma^*, m^*)$.

Definition 3 (Non-frameability). *A scheme is non-frameable if no PPT adversary wins the above game with non-negligible probability.*

3 Our Construction of EF-IBS

In this section, we present our construction. We first describe the intuition behind our construction and in particular, the technique to achieve escrow-freeness. Then, we discuss a high-level description of our scheme. Finally, we present our construction followed by the security analysis.

3.1 Intuition of EF-IBS

The rationale is that there are exponentially many possible d_{ID} for each identity ID, and that the PKG does not know what is the actual d_{ID} obtained by an honest user. At the same time, the d_{ID} is not re-randomizable. That is, given one d_{ID}, it is impossible to create another d_{ID} without the master secret key. Each signature contains a component which is a one-way function of the underlying d_{ID}. Thus, signature created by different d_{ID}'s are different. At the same time, since the PKG does not know the actual d_{ID}, the signature created by the PKG will be different to a signature created by the honest user.

On the other hand, an honest PKG would only create one d_{ID} to the user for each identity. Any user will only receive one d_{ID} and it is computationally infeasible to derive another. Thus, if there exists two signatures that uses different d_{ID}, it must be the case that the PKG is dishonest. In practice, when an honest user observes a signature that is not created by himself, he can show that the PKG is dishonest by creating another signature. Anyone observing the two signatures are created using different d_{ID} would be convinced that it is the PKG that is dishonest.

3.2 Overview of Our Construction of EF-IBS

Our construction starts with the well-known generic construction of IBS from normal digital signatures [2]. Let $\mathcal{S} = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Verify}')$ be a normal digital signature scheme. It can be used to construct an IBS as follows. The PKG invokes Gen' to create param and msk, which is merely the public and private key of an instance of $\mathcal{S}$. To create a secret key of identity ID, the PKG invokes Gen' again and create another instance of public and private key, say, $y_{\mathsf{ID}}, x_{\mathsf{ID}}$. Then the PKG creates a signature σ_{ID} using his master secret key which is a signature on "message" $y_{\mathsf{ID}}||\mathsf{ID}$. The value $(\sigma_{\mathsf{ID}}, y_{\mathsf{ID}}, x_{\mathsf{ID}})$ is returned to the user as his secret key.

To create a signature on message m, the user creates a signature σ_m on m using his x_{ID}. The overall identity-based signature is defined as $(\sigma_{\mathsf{ID}}, y_{\mathsf{ID}}, \sigma_m)$. By verifying σ_m as a valid signature under "public key" y_{ID} and that σ_{ID} is a valid signature on $y_{\mathsf{ID}}||\mathsf{ID}$ under "public key" param, the validity of the IBS can be asserted.

We make the observation that if y_{ID} is created by the user instead of the PKG, the PKG cannot obtain the secret value x_{ID}. Each identity-based signature will be associated with a unique value y_{ID} and that signatures created by the PKG will be different to the signatures created by the user. This instantiates the idea discussed in the previous section and would yield an escrow-free IBS.

However, one problem remains. Specifically, any scheme created following the above idea would have size of two digital signatures plus one public key. Thus, it will be inferior to the existing IBS by default. We tackle this issue based on the technique in aggregate signatures. Recall that an IBS consists of three parts, namely, $\sigma_m, \sigma_{\mathsf{ID}}, y_{\mathsf{ID}}$. We made the observation that the first two parts are both digital signatures of the same type. Thus, if we can employ an aggregate signature in the construction so that these two components can be aggregated into a single signature, then the final construction can be as efficient as any existing IBS (which consists of two elements). Indeed, we only require aggregate signatures that supports sequential aggregation since it is always the case that the PKG creates σ_{ID} followed by the creation of σ_m from the user. In the following, we use the aggregate signature from Boneh *et al.* [5] to realize our construction of EF-IBS.

3.3 Our Construction of EF-IBS

Following the idea described in Section 3.2, we describe our EF-IBS. For simplicity, we choose to present our scheme in the symmetric pairing setting, though we expect it could be adapted to asymmetric pairing readily.

Gen: On input security parameter λ, the PKG generates two groups $\mathbb{G}$, $\mathbb{G}_T$ of prime order p such that p is of λ-bit and that there exists a bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. It picks a generator $g \in_R \mathbb{G}$, a random value $x \in_R \mathbb{Z}_p$ and computes $Y = g^x$. It also chooses a hash function $H : \{0,1\}^* \to \mathbb{G}$. It publishes $\mathsf{param} := (\hat{e}, \mathbb{G}, \mathbb{G}_T, p, g, Y, H)$. The master secret key msk is (x).

Ext: User with identity $\mathsf{ID} \in \{0,1\}^\lambda$ engages the PKG in the following protocol. We assume there exists an external mechanism for the PKG to verify that the user is the legitimate owner of this identity ID. Furthermore, the PKG would not proceed if the value of U received in the first step has been seen before.

1. User randomly chooses $u \in_R \mathbb{Z}_p$, computes $U = g^u$ and sends U to the PKG.
2. The PKG computes $V = H(U||\mathsf{ID})^x$ and returns V to the user.
3. The user stores (u, V) as his secret key d_{ID}.
4. The PKG stores the value of U and would reject any further Ext request with value U.

Sign: User with identity ID and secret key $d_{\mathsf{ID}} = (u, V)$ creates a signature on message $m \in \{0,1\}^\lambda$ as follows. User computes $W = H(m)^u$ and $U = g^u$.[4] Next, the user computes $S = VW$. The signature σ_m on message m is parsed as (S, U).

Verify: To verify a signature (S, U) on message m with respect to identity ID, the verifier outputs 1 if and only if the following equation holds (and 0 otherwise):

$$\hat{e}\big(g, S\big) \stackrel{?}{=} \hat{e}\big(U, H(m)\big)\hat{e}\big(Y, H(U||\mathsf{ID})\big).$$

Blame: Given a valid signature (S, U) from an honest user with identity ID on message m and another message-signature pair $\big(m', (S', U')\big)$, this algorithm outputs 1 if and only if $U \neq U'$ and 0 otherwise.

3.4 Security Analysis

We first state the well-known computational Diffie-Hellman Assumption.

Definition 4 (CDH Assumption). *Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. The Computational Diffie-Hellman Assumption states that given a tuple g, g^a, g^b for some $a, b \in_R \mathbb{Z}_p$, it is computationally infeasible to compute the value g^{ab}.*

Regarding the security of our construction, we have the following theorem.

Theorem 1. *Our construction of EF-IBS possesses Unforgeability, Non-slanderability and Non-frameability under the CDH assumption in the random oracle model.*

We prove the security of our scheme through three lemmas, one for each security requirement.

Lemma 1 (Unforgeability). *If there exists a PPT $\mathcal{A}$ that wins Game Unforgeability, we show how to construct a PPT $\mathcal{S}$ that breaks the CDH assumption.*

[4] U could be stored as part of the secret key to speed up the signature generation process.

Proof. Let $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map such that $\mathbb{G} = \langle g \rangle$ of prime order p where p is of λ-bit. We show how to construct a PPT $\mathcal{S}$ that violates the CDH assumption. $\mathcal{S}$ is given g^a, g^b and its goal is to output the value g^{ab}. Let q_1, q_2 be the number of hash queries made by $\mathcal{A}$ to the hash oracle H with input length λ and 2λ respectively. Let q_3 be the number of distinct identities that appear in the signature query.

Setup $\mathcal{S}$ randomly picks $\alpha \in_R \mathbb{Z}_p$ and computes $Y = (g^a)^\alpha$. The public key is set as $\mathsf{param} = (\hat{e}, \mathbb{G}, \mathbb{G}_T, p, g, y, H)$, where H is treated as a random oracle. Note that $\mathcal{S}$ does not know the master secret key, which has a value of $a\alpha$. $\mathcal{S}$ randomly picks three indexes $\hat{i} \in \{1, \ldots, q_1\}$, $\hat{j} \in \{1, \ldots, q_2\}$ and $\hat{\ell} \in \{1, \ldots q_3\}$.

Query We show how $\mathcal{S}$ answers the various queries made by $\mathcal{A}$.

- Hash Query. For the i-th query with input $m_i \in \{0,1\}^\lambda$, $\mathcal{S}$ randomly picks $r_i \in_R \mathbb{Z}_p$. If $i \neq \hat{i}$, return g^{r_i}. Otherwise, return $(g^b)^{r_i}$. Likewise, for the j-th query with input $I_j \in \{0,1\}^{2\lambda}$, $\mathcal{S}$ randomly picks $s_j \in_R \mathbb{Z}_p$. If $j \neq \hat{j}$, return g^{s_j}. Otherwise, return $(g^b)^{s_j}$.
- Extraction Query. $\mathcal{A}$ submits an identity ID and a value U. If $U||\mathsf{ID} = I_{\hat{j}}$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ replies $(g^a)^{\alpha s_j}$ where $I_j = U||\mathsf{ID}$.
- Signature Query. For the ℓ-th query, if $\ell = \hat{\ell}$, denote $\mathcal{A}$'s input by m and ID. $\mathcal{S}$ sets $U = g^{a\beta}$ for a random value $\beta \in_R \mathbb{Z}_p$ and invokes a hash query such that $H(U||\mathsf{ID}) = g^{s_j}$. Locate the index i such that $m = m_i$ (one of the input to the hash query). If $m = m_{\hat{i}}$, $\mathcal{S}$ aborts. Otherwise, it returns $(g^a)^{\beta r_i}(g^a)^{s_j \alpha}$. On the other hand, if $\ell \neq \hat{\ell}$, (again, let m and ID be the input of $\mathcal{A}$), $\mathcal{S}$ randomly picks a value u, computes $U = g^u$ and made one hash query such that $H(U||\mathsf{ID}) = g^{s_j}$. It returns the signature as (S, U) such that $S = H(m)^u (g^a)^{\alpha s_j}$.

Output $\mathcal{A}$ submits $(S^*, U^*, \mathsf{ID}^*, m^*)$ such that

1. $\hat{e}(g, S^*) = \hat{e}(U^*, H(m^*))\hat{e}(Y, H(U^*||\mathsf{ID}^*))$.
2. $\mathcal{A}$ has not submitted a Signature Query with input (m^*, ID^*).
3. $\mathcal{A}$ has not submitted an Extraction Query with input ID^*.

With probability $\frac{(q_1-1)(q_2-q)}{q_1 q_2}$, $\mathcal{S}$ does not abort and with non-negligible probability, one of the following is true:

1. $H(U^*||\mathsf{ID}^*) = (g^b)^{s_{\hat{j}}}$ and $H(m^*) = g^{r_{\hat{i}}}$; or
2. $U^* = U' = g^{a\beta}$ and $H(m^*) = (g^b)^{r_{\hat{i}}}$ and $H(U^*||\mathsf{ID}^*) = (g^b)^{s_{\hat{j}}}$; or
3. $U^* = U' = g^{a\beta}$ and $H(m^*) = (g^b)^{r_{\hat{i}}}$ and $H(U^*||\mathsf{ID}^*) = g^{s_j}$.

In case (1), $\mathcal{S}$ computes g^{ab} as $(S^*(U^*)^{-r_{\hat{i}}})^{\frac{1}{\alpha s_{\hat{j}}}}$. In case (2), $\mathcal{S}$ computes g^{ab} as $(S^*)^{\frac{1}{\beta r_{\hat{i}} + \alpha s_j}}$. In case (3), $\mathcal{S}$ computes g^{ab} as $(S^*(g^a)^{-\alpha s_j})^{\frac{1}{\beta r_{\hat{i}}}}$.

Case (1) happens with probability $\frac{q_1-1}{q_1 q_2}$, case (2) happens with probability $\frac{1}{q_1 q_2}$ and case (3) happens with probability $\frac{q_2-1}{q_1 q_2}$. □

Lemma 2 (Non-slanderability). *If there exists a PPT $\mathcal{A}$ that wins Game Non-slanderability, we show how to construct a PPT $\mathcal{S}$ that breaks the CDH assumption.*

Proof. Let $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map such that $\mathbb{G} = \langle g \rangle$ of prime order p where p is of λ-bit. We show how to construct a PPT $\mathcal{S}$ that violates the CDH assumption. $\mathcal{S}$ is given g^a, g^b and its goal is to output the value g^{ab}. Let q_1, q_2 be the number of hash queries made by $\mathcal{A}$ to the hash oracle H with input length λ and 2λ respectively. Let q_3 be the number of PKG-Extraction Queries.

Setup $\mathcal{S}$ randomly picks $x \in_R \mathbb{Z}_p$ and computes $Y = g^x$. The public key is set as $\mathsf{param} = (\hat{e}, \mathbb{G}, \mathbb{G}_T, p, g, y, H)$, where H is treated as a random oracle. The master secret key msk is x. The value of param and x are both given to $\mathcal{A}$. $\mathcal{S}$ randomly picks two indexes $\hat{i} \in \{1, \ldots, q_1\}$ and $\hat{j} \in \{1, \ldots, q_3\}$.

Query We show how $\mathcal{S}$ answers the various queries made by $\mathcal{A}$.

- Hash Query. For the i-th query with input $m_i \in \{0,1\}^\lambda$, $\mathcal{S}$ randomly picks $r_i \in_R \mathbb{Z}_p$. If $i \neq \hat{i}$, return g^{r_i}. Otherwise, return $(g^b)^{r_i}$. For any H query with input of length 2λ-bit, $\mathcal{S}$ returns with a random element of $\mathbb{G}$.
- PKG-Extraction Query. $\mathcal{A}$ submits an identity ID. For the j-th query such that $j \neq \hat{j}$, $\mathcal{S}$ randomly picks a value $u_j \in_R \mathbb{Z}_p$, computes $U_j = g^{u_j}$, sends it to $\mathcal{A}$ and obtains V. For $j = \hat{j}$, $\mathcal{S}$ sends $U_j = g^a$ to $\mathcal{A}$.
- Signature Query. For signature query involving ID that has been submitted to j-th PKG-Extraction Query such that $j \neq \hat{j}$, $\mathcal{S}$ knows the secret key related to ID and thus replies the query properly. If ID is involved with the $\hat{j}$-th PKG-Extraction query, $\mathcal{S}$ aborts if the message m to be signed is the input to the $\hat{i}$-th query. Otherwise, it can compute S as VW where $W = (g^b)^{j_i}$.

Output With probability $\frac{q_1 - 1}{q_1}$, $\mathcal{S}$ does not abort and $\mathcal{A}$ submits (S^*, U^*, ID^*, m^*) such that there exists a Signature Query of which the return values is (S, U, ID, m) and that $U^* = U$. Further, with probability at least $1/q_1 q_3$, $U = U_{\hat{j}}$ and $m^* = m_{\hat{i}}$ and $\mathcal{S}$ computes g^{ab} as $(S^* H(U^* || \mathsf{ID})^{-x})^{\frac{1}{r_{\hat{i}}}}$. □

Lemma 3 (Non-frameability). *If there exists a PPT $\mathcal{A}$ that wins Game Non-frameability, we show how to construct a PPT $\mathcal{S}$ that breaks the CDH assumption.*

Proof. Let $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map such that $\mathbb{G} = \langle g \rangle$ of prime order p where p is of λ-bit. We show how to construct a PPT $\mathcal{S}$ that violates the CDH assumption. $\mathcal{S}$ is given g^a, g^b and its goal is to output the value g^{ab}. Let q_1, q_2 be the number of hash queries made by $\mathcal{A}$ to the hash oracle H with input length λ and 2λ respectively.

Setup $\mathcal{S}$ randomly picks $\alpha \in_R \mathbb{Z}_p$ and computes $Y = (g^a)^\alpha$. The public key is set as $\mathsf{param} = (\hat{e}, \mathbb{G}, \mathbb{G}_T, p, g, y, H)$, where H is treated as a random oracle. Note that $\mathcal{S}$ does not know the master secret key, which has a value of $a\alpha$. $\mathcal{S}$ randomly picks an index $\hat{j} \in \{1, \ldots, q_2\}$.

Query We show how $\mathcal{S}$ answers the various queries made by $\mathcal{A}$.

- Hash Query. For the i-th query with input length λ, $\mathcal{S}$ randomly picks a value r_i and return g^{r_i}. For the j-th query with input $I_j \in \{0,1\}^{2\lambda}$, $\mathcal{S}$ randomly picks $s_j \in_R \mathbb{Z}_p$. If $j \neq \hat{j}$, return g^{s_j}. Otherwise, return $(g^b)^{s_j}$.

– Extraction Query. $\mathcal{A}$ submits an identity ID and a value U. If $U||\mathsf{ID} = I_{\hat{j}}$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ replies $(g^a)^{\alpha s_j}$ where $I_j = U||\mathsf{ID}$.
– Signature Query. For each query, $\mathcal{A}$ submits a message m and an identity ID. $\mathcal{S}$ aborts if $U||\mathsf{ID} = I_{\hat{j}}$ for some U. Otherwise, $\mathcal{S}$ randomly generates $U = g^u$ (if it has not done so before) and retrieves s_j such that $H(U||\mathsf{ID}) = g^{s_j}$. It then computes S as $(g^a)^{\alpha s_j} H(m)^u$. Note that $\mathcal{A}$ is not allowed to query signatures on the identity it is going to attack.

Output With probability $\frac{q_2-1}{q_2}$, $\mathcal{S}$ does not abort and $\mathcal{A}$ submits $(S^*, U^*, \mathsf{ID}^*, m^*)$ and (S', U', ID^*, m') such that $U^* \neq U'$ and $m \neq m'$. With probability at least $2/q_2$, $U^*||\mathsf{ID}^* = I_{\hat{j}}$ or $U'||\mathsf{ID}^* = I_{\hat{j}}$. Without loss of generality, assume $U^*||\mathsf{ID}^* = I_{\hat{j}}$. $\mathcal{S}$ locate i such that $H(m^*) = g^{r_i}$ and computes $g^{ab} = (S^* U^{*-r_i})^{\frac{1}{\alpha s_j}}$. □

4 Complexity Analysis

This section aims to analyze the efficiency of the EF-IBS algorithm in terms of time complexities.

4.1 Empirical Analysis

The complexities of the five algorithms/protocols are independent of the number of users in the system. Our system is extremely efficient in Gen, Ext, Sign and Blame. The former three algorithms/protocols involve only one modular exponentiation of a cyclic group for the participants. Verify is the slowest operation which involves 3 pairing operations. As for space complexity, users are required to store the secret key d_{ID}, which consists of one element in $\mathbb{G}$ and one element in $\mathbb{Z}_p$. For efficiency consideration, the user will also store an extra group element (the element U). Signature size is short and consists of 2 elements per signature.

A breakdown of time complexity of the five protocols into the number of pairing operations and exponentiations in various groups is shown in Table 1.

Table 1. Empirical Complexities

Operation	Gen	Ext(User)	Ext(PKG)	Sign	Verify	Blame
EXP in $\mathbb{G}$	1	1	1	1	0	0
Pairing	0	0	0	0	3	0

4.2 Experimental Results

We estimate the performance of EF-IBS by measuring the time cost of various basic operations based on the pairing-based library (version 0.5.12)[5] and benchmark the performance of our system. The experimental result was produced

[5] http://crypto.stanford.edu/pbc/

through 10 test runs. The test machine was a Acer 3820TzG with 2.8GHz Intel I 5-520 CPU and 4GB of Ram running Ubuntu 10.01 with kernel 2.6.338/8. Two settings are investigated, namely, symmetric pairing and asymmetric pairing. Note that if symmetric pairing (Type A pairing) is to be employed, group elements are of size 512 bits. Furthermore, hashing to point is slow and the cost it takes is similar to that of a pairing operation. On the other hand, if asymmetric pairing (type D pairing, using parameter d62003-159-158.param) $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is to be employed, short representation of group elements is only possible in $\mathbb{G}_1$. Group order p in this setting is of 158 bit and elements in $\mathbb{G}_1$ are of size 159 bit only. However, operations in group $\mathbb{G}_2$ are slow. Our results is shown in Table 2.

Our benchmark should give an accurate estimation on the actual performance of our construction.

Table 2. Experimental Results

	Symmetric Pairing (Type A)	Asymmetric Pairing (Type D)
Exponentiations	5.4 ms	2.16 ($\mathbb{G}_1$) / 10.1 ($\mathbb{G}_2$) ms
Pairing	6.25 ms	9.4 ms
Hash to point	11 ms	0.2 ($\mathbb{G}_1$) / 46.02 ($\mathbb{G}_2$) ms
Size of group elements	64 bytes	20 ($\mathbb{G}_1$) / 60 ($\mathbb{G}_2$) bytes

5 Comparison

We compare various IBS schemes with our proposed scheme, in terms of efficiency and security features. We choose the escrow-free IBS from [16], and other efficient non escrow-free IBS. The result is summarized in Table 3.

Table 3. Comparison of different IBS

Scheme	Signature Size[a]	Escrow Free	Security Model	Sign Computation[b]	Verify Computation[b]
Yuen *et al.* [16] (using BLS [6])[c]	3	✓	ROM	1 E	4 P
Yuen *et al.* [16] (using BB [3])[c]	6	✓	standard	1 E	4 P + 2 E
Hess [11]	2	×	ROM	2 E	2 P + 2 E
Waters [15,13]	3	×	standard	3 E	3 P
Our proposed scheme	2	✓	ROM	1 E	3 P

[a] The unit for signature size is group element.

[b] When we come across the computation of sign and verify, we use E to represent an exponentiation and P to represent a pairing. Other operations such as hashing and modulus addition are negligible.

[c] Yuen *et al.* [16] provides a generic construction for escrow-free IBS scheme. We divide the comparison into using BLS [6] scheme for random oracle model and using BB [3] scheme for standard model, which are the shortest signature schemes in the literature for the corresponding models.

6 Conclusion

In this work, we presented a generic construction of escrow-free identity-based signature as well as an efficient instantiation. In contrast to the prior work due to Yuen *et al.* [16], our scheme is very practical, since it allows the public to determine whether the PKG has behaved maliciously given the evidence produced by the honest user. Further, our instantiation is very efficient since the signature size only comprises two group elements, which outperforms the existing schemes in the literature. We further provided a complexity analysis based on empirical results as well as experimental results.

Acknowledgement. This work is supported by National Natural Science Foundation of China (Grant No: 61003232).

References

1. Al-Riyami, S.S., Paterson, K.G.: Certificateless Public Key Cryptography. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 452–473. Springer, Heidelberg (2003)
2. Bellare, M., Namprempre, C., Neven, G.: Security Proofs for Identity-Based Identification and Signature Schemes. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 268–286. Springer, Heidelberg (2004)
3. 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)
4. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boneh, D., 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. 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)
7. Gentry, C.: Certificate-Based Encryption and the Certificate Revocation Problem. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 272–293. Springer, Heidelberg (2003)
8. Girault, M.: Self-certified Public Keys. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 490–497. Springer, Heidelberg (1991)
9. Goyal, V.: Reducing Trust in the PKG in Identity Based Cryptosystems. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 430–447. Springer, Heidelberg (2007)
10. Goyal, V., Lu, S., Sahai, A., Waters, B.: Black-box Accountable Authority Identity-Based Encryption. In: ACM Conference on Computer and Communications Security, pp. 427–436. ACM (2008)
11. Hess, F.: Efficient Identity Based Signature Schemes Based on Pairings. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 310–324. Springer, Heidelberg (2003)

12. Liu, J.K., Au, M.H., Susilo, W.: Self-Generated-Certificate Public Key Cryptography and Certificateless Signature/Encryption Scheme in the Standard Model. In: ASIACCS 2007, pp. 273–283. ACM Press (2007)
13. Paterson, K.G., Schuldt, J.C.N.: Efficient Identity-Based Signatures Secure in the Standard Model. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 207–222. Springer, Heidelberg (2006)
14. 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)
15. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
16. Yuen, T.H., Susilo, W., Mu, Y.: How to Construct Identity-Based Signatures without the Key Escrow Problem. Int. J. Inf. Sec. 9(4), 297–311 (2010)

Perfect Keyword Privacy in PEKS Systems

Mototsugu Nishioka

HITACHI, Ltd., Yokohama Research Laboratory, Japan
mototsugu.nishioka.rc@hitachi.com

Abstract. This paper presents a new security notion, called *perfect keyword privacy (PKP)*, for non-interactive public-key encryption with keyword search (PEKS) [5]. Although the conventional security notion for PEKS guarantees that a searchable ciphertext leaks no information about keywords, it gives no guarantee concerning leakage of a keyword from the trapdoor. PKP is a notion for overcoming this fatal deficiency. Since the trapdoor has verification functionality, the popular concept of "indistinguishability" is inadequate for capturing the notion of keyword privacy from the trapdoor. Hence, our formalization of PKP depends on the idea of formalizing a perfectly one-way hash function [10,11]. We also present *IND-PKP security* as a useful notion for showing that a given PEKS scheme has PKP. Furthermore, we present PKP+ and IND-PKP+ as enhanced notions of PKP and IND-PKP, respectively. Finally, we present several instances of an IND-PKP or IND-PKP+ secure PEKS scheme, in either the random oracle model or the standard model.

1 Introduction

Much attention has been paid to encryption systems that go beyond traditional public-key encryption (PKE) systems, such as identity-based encryption (IBE) [6,13,17], public-key searchable encryption [5,19], attribute-based encryption (ABE) [16], and functional encryption (FE) [7]. This paper deals with non-interactive public-key encryption with keyword search (PEKS), which is first presented in [5]. The PEKS provides a simple but useful mechanism to cryptographically protect data while keeping it available for search. For example, Alice can generate a searchable ciphertext corresponding to her selected keyword using Bob's public key. She then stores the ciphertext to a server. Bob can generate another key, called a *trapdoor*, corresponding to his selected keyword by using own secret key. Bob then sends the trapdoor to the server. The server can test whether or not the keywords corresponding to the ciphertext and the trapdoor are identical, and Bob can receive the ciphertext from the server only when the test is passed. In an email system, the server could be a gateway that forwards emails from Alice to Bob's portable terminal, depending on his selected keywords, such as "urgent" or "the next business meeting".

The conventional security for PEKS, called *IND-PEKS-CKA security* (cf. Definition 2), requires that the searchable ciphertext does not leak any information about the keyword. This security, however, gives no guarantee about leakage of the keyword from the trapdoor. Indeed, there exist PEKS schemes, such as the

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 175–192, 2012.

statistical consistent scheme presented in [1], that are IND-PEKS-CKA secure but the trapdoor includes the keyword itself. This could bring serious problems in many systems. For instance, in the above example, the malicious server (or gateway) could collect the keywords selected by Bob from the given trapdoors and use them to analyze his activities. The privacy of keywords from the trapdoor has been discussed in the symmetric-key setting [14] and the interactive public-key setting [9]. On the other hand, to solve a similar problem in symmetric-key predicate encryption, Shen, Shi, and Waters [18] presented a security notion, *predicate privacy*, to ensure that tokens reveal no information about the encoded query predicate. Subsequently, Blundo, Iovino, and Persiano [4] presented a *predicate encryption scheme with partial public key*, and defined a *token security* to ensure the privacy of a pattern vector from a token. To the best of our knowledge, however, there has been no discussion of the leakage of keywords from trapdoors within the framework of PEKS, which is a non-interactive and "total" public key setting.

1.1 Contributions

This paper presents a new security notion for PEKS, called *perfect keyword privacy (PKP)*, to protect the privacy of a keyword from an adversary having both the trapdoor and the ciphertext of the underlying keyword. For formalizing PKP, the well-known concept of "indistinguishability" is inadequate. This is because a trapdoor has verification functionality; that is, when a keyword and trapdoor are given, one can easily verify whether the trapdoor corresponds to the keyword (see Section 3.1 for details). Therefore, we have applied the idea of formalizing a perfectly one-way hash function (POWHF) [10,11].

Next, we present *IND-PKP security* as a useful notion for showing that a given PEKS scheme has PKP. The IND-PKP security can be defined in a game-based manner, whereas PKP is defined in a simulation-based manner. As compared with IND-PEKS-CKA security, IND-PKP security is a more extensive notion in the sense that it can ensure the privacy of a keyword from not only the ciphertext but also the trapdoor. Concerning the privacy of the keyword from only the ciphertext, however, IND-PKP security is a strictly weaker notion than IND-PEKS-CKA security. We demonstrate this by giving an instance of a PEKS scheme that is IND-PKP secure but not IND-PEKS-CKA secure (cf. Remark 7). Thus, PKP and IND-PKP security are independent notions from IND-PEKS-CKA security. Therefore, for higher security in PEKS, both IND-PEKS-CKA and IND-PKP securities are required. We also present PKP+ and IND-PKP+ security notions to enhance the PKP and IND-PKP security notions, respectively, from the viewpoint of *search pattern privacy*; that is, when two trapdoors are given, it is hard to guess whether they correspond to the same keyword.

Lastly, we give several instances of PEKS schemes that are IND-PKP secure or IND-PKP+ secure, in addition to being IND-PEKS-CKA secure. In Section 4.1, we describe the general methodology for constructing IND-PKP secure PEKS schemes. By using this methodology, in Section 4.2 we present a PEKS scheme that is IND-PEKS-CKA and IND-PKP secure in the standard

model. In Section 4.3, we present a PEKS scheme that is IND-PEKS-CKA and IND-PKP secure in the random oracle (RO) model, by direct construction. This scheme is based on the PEKS scheme in [5] and requires no computational assumptions for achieving IND-PKP security. In Section 4.4, we present a PEKS scheme that is IND-PEKS-CKA and IND-PKP+ secure in the RO model.

1.2 Related Works

Numerous works on searchable encryption have been presented so far. In this section, we briefly describe only prior works that are specifically related to this paper. In particular, we concentrate on the public-key setting.

Boneh, Di Crescenzo, Ostrovsky, and Persiano [5] first presented the framework of PEKS. They formally defined its security and presented concrete schemes with this security. They also showed a general transformation from anonymous IBE to PEKS. Abdalla et al. [1] defined consistency in PEKS and gave an improved transformation from anonymous IBE to PEKS that guarantees consistency. They also introduced three extensions of the established notions: anonymous HIBE, PKE with temporary keyword search, and IBE with keyword search. Bellare, Boldyreva, and O'Neill [3] presented an efficiently searchable encryption (ESE) system to enable fast data search (i.e., logarithmic time in the database size) in outsourced databases. The ESE system utilizes a "tag", which can be generated in a deterministic manner both from the plaintext and from the corresponding ciphertext, as an index for search. Specifically, the server computes the tag of a ciphertext to be stored in the database and uses the tag to store the ciphertext appropriately in a data structure. The client computes and sends its tag to the server and receives any matches and associated data. They also presented an ESE scheme, called "Hash-and-Encrypt" encryption scheme, and showed that it is PRIV secure in the RO model when the underlying encryption scheme is IND-CPA secure. Unlike the trapdoor in the PEKS system, an ESE tag can be computed without a secret key. Therefore, ESE always allows data searches by anyone who can access the server. Moreover, its security depends on only the ciphertext, because the tag can be computed from it. Thus, the ESE system essentially has a different structure from that of PEKS, and it is outside the scope of this paper. Camenisch, Kohlweiss, Rial, and Sheedy [9] presented an extended notion of PEKS, called public-key encryption with oblivious keyword search (PEOKS), in which a user can obtain the trapdoor from the secret key holder without revealing the keyword. They constructed a PEOKS scheme by using a committed blind anonymous IBE scheme based on the anonymous IBE scheme in [8]. In PEOKS, however, the trapdoor is generated in an interactive manner. In contrast, our goal in this paper is to define and achieve security for guaranteeing the privacy of the keyword from the trapdoor, within the framework of PEKS (i.e., trapdoors are generated in a non-interactive manner). Boneh, Sahai, and Waters [7] presented a general framework for FE, and showed that existing encryption concepts, such as ABE and PE, can be expressed as particular functionalities of FE. They also discussed the formal security definition for FE. They showed that the natural indistinguishability game-based definition is

inadequate for certain functionalities since trivially insecure constructions may satisfy it. They hence presented a simulation-based security in which one getting the secret key reveals no information other than the result of decryption when the ciphertext is given. However, although simulation-based security can be achieved in the random oracle model, for a quite simple functionality (the functionality corresponding to IBE), it cannot be achieved even in the non-programmable random oracle model. Since PEKS can also be considered as a special case of FE, the security in [7] is applicable to PEKS. Both game-based security and simulation-based security, however, have the goal of achieving privacy of a keyword from a ciphertext, and they give no guarantee concerning keyword leakage from a trapdoor.

2 Preliminaries

We say that a function $f : \mathbb{N} \to [0,1]$ is *negligible* if, for every constant $c > 0$, there exists an integer k_c such that $f(k) \leq k^{-c}$ for all $k \geq k_c$. For a group G, G^* denotes a set $G \backslash \{1_G\}$, where 1_G is an identity element of G. For a finite set S, $x \leftarrow S$ denotes the operation of picking an element uniformly from S. We use $x, x' \leftarrow S$ as shorthand for $x \leftarrow S\ ;\ x' \leftarrow S$. If A is a probabilistic algorithm, then $y \leftarrow A(x_1, x_2, \ldots; r)$ is the result of running A on inputs $x_1, x_2, \ldots$ and coins r. We let $y \leftarrow A(x_1, x_2, \ldots)$ denote the experiment of picking r at random and letting y to be $A(x_1, x_2, \ldots; r)$. The notation $\Pr[x_1 \leftarrow S_1;\ x_2 \leftarrow S_2;\ \ldots : p(x_1, x_2, \ldots)]$ denotes the probability that the predicate $p(x_1, x_2, \ldots)$ is true after the ordered execution of $x_1 \leftarrow S_1,\ x_2 \leftarrow S_2$, and so on. If α is neither an algorithm nor a set then $x \leftarrow \alpha$ is a simple assignment statement. For a random variable $\mathbf{X}$, $[\mathbf{X}]$ denotes a set $\{x \mid \Pr[\mathbf{X} = x] > 0\}$, and $||\mathbf{X}||$ denotes a value $\max_{x \in [\mathbf{X}]}\{\Pr[\mathbf{X} = x]\}$. $E(\mathbf{X})$ denotes the *expectation* of $\mathbf{X}$, and $x \leftarrow \mathbf{X}$ denotes selection of a random sample from $\mathbf{X}$; thus, $\Pr[x \leftarrow \mathbf{X}] = \Pr[\mathbf{X} = x]$. We use $x, x' \leftarrow \mathbf{X}$ as shorthand for $x \leftarrow \mathbf{X}\ ;\ x' \leftarrow \mathbf{X}$. The random variables $\mathbf{X}$ and $\mathbf{Y}$ are *independent* if $\Pr[\mathbf{X} = a \wedge \mathbf{Y} = b] = \Pr[\mathbf{X} = a] \cdot \Pr[\mathbf{Y} = b]$ for any $a, b \in \{0,1\}^*$. A *probability ensemble* is a sequence $\mathcal{X} = \{\mathbf{X}_k\}_{k \in \mathbb{N}}$ of random variables $\mathbf{X}_k$. We say that $\mathcal{X}$ is *well-spread* if $||\mathbf{X}_k||$ is negligible in k. The *d-composite bilinear group generator* $\mathcal{G}$ is a PPT algorithm that takes a security parameter k as input and outputs $(p_1, \ldots, p_d, \mathbb{G}, \mathbb{G}_T, \mathbf{e})$, where p_i are prime numbers with $p_i > 2^k$, $\mathbb{G}$ and $\mathbb{G}_T$ are multiplicative cyclic groups with order $N = \prod_{i=1}^{d} p_i$, and $\mathbf{e}$ is a map from $\mathbb{G} \times \mathbb{G}$ to $\mathbb{G}_T$, called a *bilinear map*, with the following properties:

1. Computable: There is an efficient algorithm to compute $\mathbf{e}(g, h)$ for any $g, h \in \mathbb{G}$.
2. Bilinear: $\mathbf{e}(g^x, g^y) = \mathbf{e}(g, g)^{xy}$ for any $g \in \mathbb{G}$ and any $x, y \in \mathbb{Z}_N$.
3. Non-degenerate: If g is a generator of $\mathbb{G}$ then $\mathbf{e}(g, g)$ is a generator of $\mathbb{G}_T$.

In particular, the 1-composite bilinear group generator is simply called a *bilinear group generator*. For an integer m dividing N, $\mathbb{G}_m$ denotes the subgroup of $\mathbb{G}$ with order m. Then, $\mathbf{e}(x, y) = 1_{\mathbb{G}}$ for any $x \in \mathbb{G}_m$ and any $y \in \mathbb{G}_n$ when m and n are coprime. This is called the "orthogonality property".

Definition 1. *A* non-interactive public-key encryption with keyword search (PEKS) *scheme consists of the following polynomial-time randomized algorithms:*

- $\mathsf{KG}(1^k)$*: Takes a security parameter* k*, and generates a public/secret key pair* (PK, SK)*. Here, the keys include the information about the keyword space* KSP_k*.*
- $\mathsf{Td}(SK, w)$*: For* SK *and a keyword* $w \in \mathsf{KSP}_k$*, produces a* trapdoor T_w*.*
- $\mathsf{PEKS}(PK, w)$*: For* PK *and* $w \in \mathsf{KSP}_k$*, produces a* searchable ciphertext C_w *of* w*.*
- $\mathsf{Test}(PK, C_w, T_{w'})$*: For* PK*,* $C_w = \mathsf{PEKS}(PK, w)$*, and* $T_{w'} = \mathsf{Td}(SK, w')$*, where* $w, w' \in \mathsf{KSP}_k$*, outputs* 1 *if* $w = w'$*. Otherwise, outputs* 0 *with an overwhelming probability*[1]*.*

The security of PEKS is defined against an active attacker who is able to obtain a trapdoor T_w for any keyword w of his choice, to ensure that a $\mathsf{PEKS}(PK, w)$ does not reveal any information about w *unless* T_w *is available* [5].

IND-PEKS-CKA Security. Let $\Pi = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ be a PEKS scheme, and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a probabilistic polynomial-time (PPT) adversary. We then consider the following experiment.

Experiment $\mathsf{Exp}^{\text{ind:peks}}_{\mathcal{A},\Pi}(k)$
$(PK, SK) \leftarrow \mathsf{KG}(1^k)$; $(w_0, w_1, \sigma) \leftarrow \mathcal{A}_1^{\mathsf{Td}(SK,\cdot)}(1^k, PK)$
$b \leftarrow \{0, 1\}$; $C_{w_b} \leftarrow \mathsf{PEKS}(PK, w_b)$; $b' \leftarrow \mathcal{A}_2^{\mathsf{Td}(SK,\cdot)}(1^k, PK, \sigma, w_0, w_1, C_{w_b})$
If $b = b'$ then return 1 else return 0.

Here, $w_0, w_1 \in \mathsf{KSP}_k$ and $w_0 \neq w_1$, σ is a string representing the configuration of $\mathcal{A}_1$ at its quitting point, and $\mathcal{A}$ is prohibited from asking for the trapdoors w_0 or w_1. The *advantage* of $\mathcal{A}$ in the above experiment is defined as

$$\mathsf{Adv}^{\text{ind:peks}}_{\mathcal{A},\Pi}(k) = \left| \Pr\left[\mathsf{Exp}^{\text{ind:peks}}_{\mathcal{A},\Pi}(k) = 1\right] - \frac{1}{2} \right|.$$

Definition 2. *We say that a PEKS scheme* Π *is* indistinguishable against a chosen-keyword attack (CKA)*, briefly,* IND-PEKS-CKA secure*, if* $\mathsf{Adv}^{\text{ind:peks}}_{\mathcal{A},\Pi}(k)$ *is negligible for any* $\mathcal{A}$*.*

3 Perfect Keyword Privacy

3.1 Definition

The IND-PEKS-CKA security (in Definition 2) guarantees the privacy of the keyword from a searchable ciphertext. It does not, however, guarantee any security concerning leakage of the keyword from the trapdoor. For example, in [1], a PEKS scheme with statistical consistency is presented and shown to be

[1] This property is called *computational consistency* in [1]. In this paper, we call it "consistency" for brevity.

IND-PEKS-CKA secure under the BDH assumption. That scheme is designed, however, so that the trapdoor includes the keyword itself. To overcome this deficiency, we present a new security notion, *perfect keyword privacy* (briefly, PKP), for a PEKS to ensure the privacy of the keyword from both the trapdoor and the searchable ciphertext. In this section, we present a formal definition of PKP. In formulating security against information leakage, the natural, popular concept that comes to mind is "indistinguishability". We first explain why indistinguishability is inadequate for defining PKP. We now consider the following game based on indistinguishability.

1. For $(PK, SK) \leftarrow \mathsf{KG}(1^k)$, the adversary receives the public key PK and is allowed to access to the trapdoor oracle $\mathsf{Td}(SK, \cdot)$.
2. In the challenge phase, the adversary submits two keywords, w_0, w_1, and receives a target trapdoor $T_{w_b} = \mathsf{Td}(SK, w_b)$ for a randomly chosen $b \in \{0, 1\}$. The adversary can continuously make queries to the trapdoor oracle $\mathsf{Td}(SK, \cdot)$, except for querying w_0 or w_1.
3. In the guess phase, the adversary finally outputs $b' \in \{0, 1\}$ as its guess for b.

It is then required that no PPT adversary can guess the challenge bit b with a non-negligible advantage. There exists an adversary, however, that can guess b with an overwhelming probability in the above game. After receiving the trapdoor T_{w_b} in Step 2, the adversary computes $C_{w_i} = \mathsf{PEKS}(PK, w_i)$ for each $i = 0, 1$ and outputs $b' \in \{0, 1\}$ such that $\mathsf{Test}(T_{w_b}, C_{w_{b'}}) = 1$. Then, from the consistency of PEKS, the probability $\Pr[b = b']$ is overwhelming.

Our formalization of PKP depends on an idea of formalizing a POWHF [10,11]. Informally, we say that a PEKS scheme has PKP if there is no efficient way to guess the keyword w from the given trapdoor T_w and ciphertext C_w other than the "select and test" method; in other words, the adversary selects a keyword w' in an arbitrary manner and tests whether $\mathsf{Test}(T_w, \mathsf{PEKS}(PK, w')) = 1$ holds. If the test is passed, the adversary decides that $w = w'$. In our definition, the "select and test" method is formalized by an oracle $\mathcal{O}_w$, called a *test oracle*, in the *ideal system*: for a query (keyword) w', $\mathcal{O}_w$ responds with 1 if $w = w'$; otherwise, it responds with 0. Note that one may think that the oracle $\mathcal{O}_w$ should be defined so that it outputs 0 with an overwhelming probability when $w \neq w'$ because Definition 1 adopts computational consistency. It can easily be shown, however, that this difference does not affect Definition 3.

Perfect Keyword Privacy. Let $\Pi = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ be a PEKS scheme. Let $\mathcal{X} = \{\mathbf{X}_k\}_{k \in \mathbb{N}}$ be a probability ensemble such that $[\mathbf{X}_k] = \mathsf{KSP}_k$. From now on, unless otherwise indicated, we assume that $\mathcal{X}$ is well-spread and independent from key generation (cf. Remarks 1 and 2). $\mathcal{X}$ determines the distribution of keywords; that is, when the security parameter k is given, the keyword w is given as a random sample from $\mathbf{X}_k$. Let $\mathcal{P} = \{P_k\}_{k \in \mathbb{N}}$ be a predicate family, where P_k is an efficiently computable predicate over $[\mathbf{X}_k]$. Let $\mathcal{A}$ and $\mathcal{B}$ be PPT algorithms. We then define the following experiments. See Section 2 for other notations and conventions.

Experiment $\mathsf{Exp}^{\mathrm{pkp:real}}_{\mathcal{A},\Pi,\mathcal{X},\mathcal{P}}(k)$
$w \leftarrow \mathbf{X}_k$; $(PK, SK) \leftarrow \mathsf{KG}(1^k)$
$T_w \leftarrow \mathsf{Td}(SK, w)$; $C_w \leftarrow \mathsf{PEKS}(PK, w)$
$z \leftarrow \mathcal{A}^{\mathsf{Td}(SK,\cdot)}(1^k, PK, T_w, C_w)$
If $z = P_k(w)$ then return 1 else return 0.

Experiment $\mathsf{Exp}^{\mathrm{pkp:ideal}}_{\mathcal{B},\Pi,\mathcal{X},\mathcal{P}}(k)$
$w \leftarrow \mathbf{X}_k$; $(PK, SK) \leftarrow \mathsf{KG}(1^k)$
$z \leftarrow \mathcal{B}^{\mathcal{O}_w,\mathsf{Td}(SK,\cdot)}(1^k, PK)$
If $z = P_k(w)$ then return 1
else return 0.

Definition 3. *We say that a PEKS scheme Π has* perfect keyword privacy (PKP) with respect to $\mathcal{X}$ *if for any $\mathcal{P}$ and $\mathcal{A}$, there exists a negligible function* negl *and $\mathcal{B}$ such that*

$$\Pr\left[\mathsf{Exp}^{\mathrm{pkp:real}}_{\mathcal{A},\Pi,\mathcal{X},\mathcal{P}}(k) = 1\right] \leq \Pr\left[\mathsf{Exp}^{\mathrm{pkp:ideal}}_{\mathcal{B},\Pi,\mathcal{X},\mathcal{P}}(k) = 1\right] + \mathsf{negl}(k) \tag{1}$$

for all $k \in \mathbb{N}$. We also say that Π has PKP *if it has PKP with respect to any $\mathcal{X}$.*

Remark 1. In Definition 3, the probability ensemble $\mathcal{X}$ is given independently from the key generation of the PEKS scheme. This setting is very significant for obtaining a useful notion, *IND-PKP security*, to achieve PKP (see the proof of Theorem 1). From a practical viewpoint, we think that this is a natural setting in the real world. Generally, public keys are not used as keywords because they are large, meaningless phrases, whereas other identifiers, such as a user's name and email address, are usually used to designate a person.

Remark 2. Definition 3 is meaningful even if $\mathcal{X}$ is not well-spread. However, without loss of generality, we can assume that the probability ensemble $\mathcal{X}$ is well-spread when defining the privacy of the keyword from the trapdoor. As described at the beginning of this section, if the trapdoor is given, the adversary can always verify whether it corresponds to his own chosen keyword. From this fact, in (1) we can exclude the case of choosing $w \in [\mathbf{X}_k]$ such that $\Pr[\mathbf{X}_k = w]$ is non-negligible. Notice that the number of keywords appearing with a non-negligible probability is polynomially bounded in k.

Remark 3. In Definition 3, only a single tuple of the trapdoor and ciphertext is given to the adversary $\mathcal{A}$. In Section 3.2, we present a notion, IND-PKP security, and use it to show that a given PEKS scheme has PKP. From a hybrid argument [2], we can show that (single-target) IND-PKP security implies multi-target IND-PKP security. Thus, IND-PKP security implies multi-target PKP.

Remark 4. Concerning the privacy of a keyword from only the searchable ciphertext, IND-PEKS-CKA security gives strictly stronger security than that of PKP. In Remark 7, we demonstrate this by presenting a PEKS scheme that is IND-PKP secure (cf. Section 3.2) but not IND-PEKS-CKA secure. On the other hand, there exist PEKS schemes, such as the scheme in [1] described above, that are IND-PEKS-CKA secure but do not have PKP. Thus, PKP is a separate security notion from IND-PEKS-CKA security; that is, PKP and IND-PEKS-CKA security are independent of each other. Hence, for higher security in a PEKS system, both IND-PEKS-CKA security and PKP are required. Note that strictly speaking, the above results on separation and comparison are true under some computational complexity assumptions because they are required for achieving the securities of the instances.

We expect that the idea of PKP will be applied in FE systems to ensure the privacy of a key from a secret key (see [7] for the detail of FE); since FE is a generalized concept of many other primitives, such as IBE, PE, and ABE, this idea is also applicable to those primitives. Informally, we say that an FE scheme for a functionality F over (K, X) has *perfect key privacy* if a secret key sk_k corresponding to the key $k \in K$ leaks no information about k, beyond the information obtained from the oracle $\mathcal{O}_{F(k,\cdot)}$, where for the query x, $\mathcal{O}_{F(k,\cdot)}$ returns $F(k, x)$. If $\mathcal{O}_{F(k,\cdot)}$ gives only trivial information[2], like $\mathcal{O}_x$ in PEKS, then this notion will give meaningful security in an FE system. We leave a detailed, formal discussion to subsequent works.

3.2 How to Achieve PKP

In this section, we present a useful notion, called *IND-PKP security*, to show that a given PEKS scheme has PKP. The IND-PKP security can be defined in a game-based manner, whereas we defined PKP above in a simulation-based manner. The IND-PKP security can be regarded as a strictly stronger notion than PKP from the viewpoint of the strength relation between the cryptographic assumptions for achieving these securities (cf. Remark 6).

IND-PKP Security. Let $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$ be a probability ensemble, and let $\Pi = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ be a PEKS scheme. Let $\mathcal{A}$ be a PPT algorithm, called *IND-PKP adversary*. We then define the following experiment (cf. Remark 3).

Experiment $\mathsf{Exp}^{\text{ind-pkp}}_{\mathcal{A},\Pi,\mathcal{X}}(k)$
$w_0, w_1 \leftarrow \mathbf{X}_k$; $b \leftarrow \{0,1\}$; $(PK, SK), (PK', SK') \leftarrow \mathsf{KG}(1^k)$
$T_{w_0} \leftarrow \mathsf{Td}(SK, w_0)$; $C_{w_0} \leftarrow \mathsf{PEKS}(PK, w_0)$
$T'_{w_b} \leftarrow \mathsf{Td}(SK', w_b)$; $C'_{w_b} \leftarrow \mathsf{PEKS}(PK', w_b)$
$b' \leftarrow \mathcal{A}^{\mathsf{Td}(SK,\cdot),\mathsf{Td}(SK',\cdot)}(1^k, PK, T_{w_0}, C_{w_0}, PK', T'_{w_b}, C'_{w_b})$
If $b = b'$ then return 1 else return 0.

The *advantage* of $\mathcal{A}$ in the above experiment is defined as

$$\mathsf{Adv}^{\text{ind-pkp}}_{\mathcal{A},\Pi,\mathcal{X}}(k) = \left| \Pr\left[\mathsf{Exp}^{\text{ind-pkp}}_{\mathcal{A},\Pi,\mathcal{X}}(k) = 1\right] - \frac{1}{2} \right|,$$

and $b \in \{0,1\}$ is called a *challenge bit*.

Definition 4. *We say that a PEKS scheme Π is* IND-PKP secure with respect to $\mathcal{X}$ *if* $\mathsf{Adv}^{\text{ind-pkp}}_{\mathcal{A},\Pi,\mathcal{X}}(k)$ *is negligible for any $\mathcal{A}$. We also say that Π is* IND-PKP secure *if it is IND-PKP secure with respect to any $\mathcal{X}$.*

Theorem 1. *If the PEKS scheme $\Pi = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ is IND-PKP secure, then it has PKP.*

[2] For example, $F(k, x)$ represents a result of execution of certain program P_k for input x. P_k outputs a meaningful string only for particular x, whereas it outputs $\perp$ for other input. It is easy to find such particular x from k but difficult to find it from sk_k.

Proof. Let $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$ be a well-spread probability ensemble. We show that if the PEKS scheme Π is IND-PKP secure with respect to $\mathcal{X}$, then it has PKP with respect to $\mathcal{X}$. For an IND-PKP adversary $\mathcal{A}$, we define

$$\rho^{(1)}_{\mathcal{A},\Pi,\mathcal{X}}(k) = \Pr\Big[(PK,SK),(PK',SK') \leftarrow \mathsf{KG}(1^k)\ ;\ w \leftarrow \mathbf{X}_k\ ;\ T_w \leftarrow \mathsf{Td}(SK,w)\ ;\ C_w \leftarrow \mathsf{PEKS}(PK,w)\ ;\ T'_w \leftarrow \mathsf{Td}(SK',w)\ ;\ C'_w \leftarrow \mathsf{PEKS}(PK',w)\ :\ \mathcal{A}^{\mathsf{Td}(SK,\cdot),\mathsf{Td}(SK',\cdot)}(1^k, PK, T_w, C_w, PK', T'_w, C'_w) = 1\Big],$$

$$\rho^{(2)}_{\mathcal{A},\Pi,\mathcal{X}}(k) = \Pr\Big[(PK,SK),(PK',SK') \leftarrow \mathsf{KG}(1^k)\ ;\ w,w' \leftarrow \mathbf{X}_k\ ;\ T_w \leftarrow \mathsf{Td}(SK,w)\ ;\ C_w \leftarrow \mathsf{PEKS}(PK,w)\ ;\ T'_{w'} \leftarrow \mathsf{Td}(SK',w')\ ;\ C'_{w'} \leftarrow \mathsf{PEKS}(PK',w') : \mathcal{A}^{\mathsf{Td}(SK,\cdot),\mathsf{Td}(SK',\cdot)}(1^k, PK, T_w, C_w, PK', T'_{w'}, C'_{w'}) = 1\Big].$$

Then we have

$$2 \cdot \mathsf{Adv}^{\text{ind-pkp}}_{\mathcal{A},\Pi,\mathcal{X}}(k) = \left|\rho^{(1)}_{\mathcal{A},\Pi,\mathcal{X}}(k) - \rho^{(2)}_{\mathcal{A},\Pi,\mathcal{X}}(k)\right|. \tag{2}$$

We now suppose that Π does not have PKP with respect to $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$. Then from Definition 3, there exists a predicate family $\mathcal{P} = \{P_k\}_{k\in\mathbb{N}}$ and a PPT algorithm $\mathcal{B}$ such that for any PPT algorithm $\mathcal{C}$,

$$\rho(k) = \Pr\left[\mathsf{Exp}^{\text{pkp:real}}_{\mathcal{B},\Pi,\mathcal{X},\mathcal{P},t}(k) = 1\right] - \Pr\left[\mathsf{Exp}^{\text{pkp:ideal}}_{\mathcal{C},\Pi,\mathcal{X},\mathcal{P}}(k) = 1\right] \tag{3}$$

is non-negligible. We now consider a PPT algorithm $\mathcal{C}$ (in the ideal system) that works as follows:

1. Select a random sample $w' \leftarrow \mathbf{X}_k$ and make a trapdoor query w' to obtain the trapdoor $T_{w'}$. Generate a searchable ciphertext of w' by $C_{w'} \leftarrow \mathsf{PEKS}(PK, w')$.
2. Run $\mathcal{B}$ on input $(1^k, PK, T_{w'}, C_{w'})$, and output the corresponding response of $\mathcal{B}$. If $\mathcal{B}$ makes trapdoor queries then respond to them by using $\mathcal{C}$'s trapdoor oracle.

This completes the description of $\mathcal{C}$. Moreover, for each $w \in [\mathbf{X}_k]$, we define

$$\zeta_{w;\mathcal{B}}(k) = \Pr\Big[(PK,SK) \leftarrow \mathsf{KG}(1^k)\ ;\ T_w \leftarrow \mathsf{Td}(SK,w)\ ;\ C_w \leftarrow \mathsf{PEKS}(PK,w)\ :\ \mathcal{B}^{\mathsf{Td}(SK,\cdot)}(1^k, PK, T_w, C_w) = 1\Big].$$

Let $S^i_{P_k}$ denote a set $\{w \in [\mathbf{X}_k] \mid P_k(w) = i\}$, for each $i \in \{0,1\}$. Then, since $\mathbf{X}_k$ is independent from the key generation, we have

$$\Pr\left[\mathsf{Exp}^{\text{pc:real}}_{\mathcal{B},\Pi,\mathcal{X},\mathcal{P}}(k) = 1\right] = \sum_{w\in S^1_{P_k}} \Pr[\mathbf{X}_k = w] \cdot \zeta_{w;\mathcal{B}}(k) + \sum_{w\in S^0_{P_k}} \Pr[\mathbf{X}_k = w] \cdot (1 - \zeta_{w;\mathcal{B}}(k)), \tag{4}$$

$$\Pr\left[\mathsf{Exp}^{\text{pc:ideal}}_{\mathcal{B},\Pi,\mathcal{X},\mathcal{P}}(k)=1\right] = \sum_{w\in S^1_{P_k}} \Pr[\mathbf{X}_k=w]\cdot \sum_{w'\in[\mathbf{X}_k]} \Pr[\mathbf{X}_k=w']\cdot \zeta_{w';\mathcal{B}}(k) + \sum_{w\in S^0_{P_k}} \Pr[\mathbf{X}_k=w]\cdot \sum_{w'\in[\mathbf{X}_k]} \Pr[\mathbf{X}_k=w']\cdot (1-\zeta_{w';\mathcal{B}}(k)). \tag{5}$$

Let $\mathbf{Z}_k$ be a random variable over $[0,1]=\{a\in\mathbb{R}\mid 0\le a\le 1\}$ such that $\Pr[\mathbf{Z}_k=\zeta_{w;\mathcal{B}}(k)]=\Pr[\mathbf{X}_k=w]$. Then, from (3), (4), and (5), we have

$$\begin{aligned}\rho(k) &= \sum_{w\in S^1_{P_k}} \Pr[\mathbf{X}_k=w]\cdot\left(\zeta_{w;\mathcal{B}}(k)-\sum_{w'\in[\mathbf{X}_k]}\Pr[\mathbf{X}_k=w']\cdot\zeta_{w';\mathcal{B}}(k)\right)\\ &\quad-\sum_{w\in S^0_{P_k}} \Pr[\mathbf{X}_k=w]\cdot\left(\zeta_{w;\mathcal{B}}(k)-\sum_{w'\in[\mathbf{X}_k]}\Pr[\mathbf{X}_k=w']\cdot\zeta_{w',\mathcal{B}}(k)\right)\\ &\le \sum_{w\in[\mathbf{X}_k]}\Pr[\mathbf{X}_k=w]\cdot\left|\zeta_{w;\mathcal{B}}(k)-\sum_{w'\in[\mathbf{X}_k]}\Pr[\mathbf{X}_k=w']\cdot\zeta_{w',\mathcal{B}}(k)\right|\\ &= E(|\mathbf{Z}_k-E(\mathbf{Z}_k)|)\le\sqrt{E(\mathbf{Z}_k^2)-E(\mathbf{Z}_k)^2}. \end{aligned}\tag{6}$$

Let $\mathcal{A}^*$ be an IND-PKP adversary such that for a given input $(1^k, PK, T_w, C_w, PK', \bar{T}, \bar{C})$, where $(\bar{T},\bar{C})$ is (T'_w, C'_w) or $(T'_{w'}, C'_{w'})$, it runs $\mathcal{B}$ twice and outputs 1 only when $\mathcal{B}(1^k, PK, T_w, C_w)=\mathcal{B}(1^k, PK', \bar{T}, \bar{C})=1$. Then we have

$$E(\mathbf{Z}_k^2)=\rho^{(1)}_{\mathcal{A}^*,\Pi,\mathcal{X}}(k) \qquad \text{and} \qquad E(\mathbf{Z}_k)^2=\rho^{(2)}_{\mathcal{A}^*,\Pi,\mathcal{X}}(k). \tag{7}$$

From (2), (6), and (7), we finally have

$$\rho(k)^2\le\left|\rho^{(1)}_{\mathcal{A}^*,\Pi,\mathcal{X}}(k)-\rho^{(2)}_{\mathcal{A}^*,\Pi,\mathcal{X}}(k)\right|=\frac{1}{2}\mathsf{Adv}^{\text{ind-pkp}}_{\mathcal{A}^*,\Pi,\mathcal{X}}(k).$$

This contradicts the assumption that Π is IND-PKP secure.

3.3 Additional Notions

In Section 4.3, we present an IND-PKP secure PEKS system in which the trapdoor is generated in a deterministic manner. In this system, when two trapdoors are given under the same secret key, one can easily guess whether they correspond to the same keyword. Thus, IND-PKP security cannot assure "search pattern privacy", in general. In this section, we address this issue.

Search Pattern Privacy. Let $\mathcal{X}=\{\mathbf{X}_k\}_{k\in\mathbb{N}}$ be a probability ensemble, and let $\Pi=(\mathsf{KG},\mathsf{Td},\mathsf{PEKS},\mathsf{Test})$ be a PEKS scheme. Let $\mathcal{A}$ be a PPT algorithm, called a *SPP adversary*. We then define the following experiment.

Experiment $\mathsf{Exp}^{\mathrm{spp}}_{\mathcal{A},\Pi,\mathcal{X}}(k)$
$w_0, w_1 \leftarrow \mathbf{X}_k$; $b \leftarrow \{0,1\}$; $(PK, SK) \leftarrow \mathsf{KG}(1^k)$
$T_{w_0} \leftarrow \mathsf{Td}(SK, w_0)$; $T_{w_b} \leftarrow \mathsf{Td}(SK, w_b)$; $b' \leftarrow \mathcal{A}^{\mathsf{Td}(SK,\cdot)}(1^k, PK, T_{w_0}, T_{w_b})$
If $b = b'$ then return 1 else return 0.

The *advantage* of $\mathcal{A}$ in the above experiment is defined as

$$\mathsf{Adv}^{\mathrm{spp}}_{\mathcal{A},\Pi,\mathcal{X}}(k) = \left| \Pr\left[\mathsf{Exp}^{\mathrm{spp}}_{\mathcal{A},\Pi,\mathcal{X}}(k) = 1\right] - \frac{1}{2} \right|,$$

and $b \in \{0,1\}$ is called a *challenge bit*.

Definition 5. *We say that a PEKS scheme Π has* search pattern privacy *(briefly, SPP) if* $\mathsf{Adv}^{\mathrm{spp}}_{\mathcal{A},\Pi,\mathcal{X}}(k)$ *is negligible for any $\mathcal{A}$ and $\mathcal{X}$.*

Definition 6 (PKP+ and IND-PKP+). *We say that a PEKS scheme has* PKP+ *if it has both PKP and SPP. We also say that a PEKS scheme is* IND-PKP+ secure *if it is IND-PKP secure and has SPP.*

Remark 5. In Definition 5, it is essential that the adversary cannot see the ciphertexts C_{w_0} and C_{w_1}. If either of these is given, the adversary can easily guess b by running the test algorithm. Thus, in a real system, SPP is meaningful in a situation in which there is no ciphertext corresponding to the search keyword (although the searcher has multiple trapdoors corresponding to the underlying keyword). In our definition of SPP, the adversary is not allowed to choose the keywords w_0, w_1. This is because we regard SPP as an additional notion for PKP to strengthen the privacy of keywords.

4 PEKS Schemes with Perfect Keyword Privacy

As described in Remark 4, concerning the privacy of a keyword from only a searchable ciphertext, IND-PKP security ensures strictly weaker security than that of IND-PEKS-CKA security. Therefore, for higher security in PEKS, we present several instances of a PEKS scheme that is IND-PKP secure or IND-PKP+ secure, in addition to being IND-PEKS-CKA secure. As much as we possible, we looked for appropriate instances in existing schemes and modified them if necessary.

4.1 General Methodology

Before giving concrete instances, we describe a general methodology for achieving IND-PKP security in PEKS schemes. We first introduce the notion of a *secure injective-function generator*.

Definition 7. *The* injective-function generator *is a pair of PPT algorithms $\mathcal{I}$ and $\mathfrak{G}$ such that (1) $\mathcal{I}$ takes a security parameter k as input and outputs $\lambda_k \in \{0,1\}^*$, and (2) $\mathfrak{G}$ takes λ_k as input and outputs an injective function $\pi : Y_{\lambda_k} \rightarrow$*

Z_{λ_k}, where Y_{λ_k} and Z_{λ_k} are sets uniquely determined from λ_k. We say that the injective-function generator $(\mathcal{I}, \mathfrak{G})$ is secure *if for any well-spread probability ensemble $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$ with $[\mathbf{X}_k] \subseteq Y_{\lambda_k}$, and any PPT algorithm $\mathcal{B}$,*

$$\mathsf{Adv}^{\mathrm{sif}}_{\mathcal{B},\mathcal{I},\mathfrak{G},\mathcal{X}}(k) = \left|\Pr\left[\lambda_k \leftarrow \mathcal{I}(1^k)\ ;\ \pi, \pi' \leftarrow \mathfrak{G}(\lambda_k)\ ;\ x_0, x_1 \leftarrow \mathbf{X}_k\ ;\right.\right.$$
$$\left.\left. b \leftarrow \{0,1\}\ ;\ b' \leftarrow \mathcal{B}(1^k, \pi, \pi', \pi(x_0), \pi'(x_b))\ :\ b = b'\right] - \frac{1}{2}\right|$$

is negligible.

An example of a secure injective-function generator is given in Section 4.2. Next, we describe how to convert a PEKS scheme into an IND-PKP secure PEKS scheme by using a secure injective-function generator. The essential point of the conversion is that the secure function generator yields a fresh injective function for each user, and the trapdoor and ciphertext are created from the keyword's function value. Let $\Pi = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ be a PEKS scheme, and let $(\mathcal{I}, \mathfrak{G})$ be an injective-function generator such that for $\lambda_k \leftarrow \mathcal{I}(1^k)$, $\mathfrak{G}(\lambda_k)$ outputs an injective function from KSP_k to KSP_k. We then define a PEKS scheme $\Pi^* = (\mathsf{KG}^*, \mathsf{Td}^*, \mathsf{PEKS}^*, \mathsf{Test}^*)$ as follows.

- $\mathsf{KG}^*(1^k)$ outputs $(PK^*, SK^*) = ((PK, \lambda_k, \pi), (SK, \lambda_k, \pi))$ for $(PK, SK) \leftarrow \mathsf{KG}(1^k)$, $\lambda_k \leftarrow \mathcal{I}(1^k)$, and $\pi \leftarrow \mathfrak{G}(\lambda_k)$, where λ_k is a common parameter for all users in this system.
- $\mathsf{Td}^*(SK^*, w)$ outputs $T_{\pi(w)} \leftarrow \mathsf{Td}(SK, \pi(w))$.
- $\mathsf{PEKS}^*(PK^*, w)$ outputs $C_{\pi(w)} \leftarrow \mathsf{PEKS}(PK, \pi(w))$.
- Test^* is identical with Test.

Theorem 2. *In the PEKS scheme Π^*, we have the following results.*

(a) *If $(\mathcal{I}, \mathfrak{G})$ is secure, then Π^* is IND-PKP secure.*
(b) *If Π is IND-PEKS-CKA secure, then Π^* is IND-PEKS-CKA secure.*

The proof of Theorem 2 is given in the full version of this paper. The above methodology is simple and useful although some additional assumption may be required for secure function generator. This methodology however cannot guarantee SPP in Π^*. The brute force approach (under a constraint) for obtaining an IND-PKP+ secure PEKS scheme Π^* by using a secure injective-function generator $(\mathcal{I}, \mathfrak{G})$ is as follows. KG^* creates $(PK^*, SK^*) = ((PK, \lambda_k, \pi_1, \ldots, \pi_n), (SK, \lambda_k, \pi_1, \ldots, \pi_n))$ for $(PK, SK) \leftarrow \mathsf{KG}(1^k)$, $\lambda_k \leftarrow \mathcal{I}(1^k)$, and $\pi_1, \ldots, \pi_n \leftarrow \mathfrak{G}(\lambda_k)$. $\mathsf{Td}^*(SK, w)$ has a counter, and it outputs $T^*_w = T_{\pi_i(w)}$ if this is the i-th execution for the same keyword w. $\mathsf{PEKS}^*(PK, w)$ outputs $C^*_w = (C_{\pi_i(w)})_{1\le i\le n}$. It can readily be shown that if $(\mathcal{I}, \mathfrak{G})$ is a secure injective-function generator and the adversary is restricted to making at most n trapdoor queries to the same keyword, then Π^* is IND-PKP+ secure. We do not know of a general methodology for obtaining an IND-PKP+ secure PEKS scheme without restriction. This problem remains open. Note that obviously, we can obtain a similar result to Theorem 2 when applying an RO generator (i.e., π is an RO in the above Π^*) instead of a secure injective-function generator (cf. Proposition 5).

4.2 Instance 1

In this section, we present a concrete instance of a PEKS scheme that can be obtained by the methodology described in Section 4.1. This instance is based on the Gentry IBE scheme [15] and the conversion [1] from the IBE scheme to the PEKS scheme. The resulting scheme is both IND-PKP and IND-PEKS-CKA secure in the standard model. Let $\mathcal{G}$ be a bilinear group generator. We define an injective function generator $(\mathcal{G}, \mathfrak{G})$ as follows: For $I = (p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}) \leftarrow \mathcal{G}(1^k)$, $\mathfrak{G}(I)$ picks a primitive element $\xi \in \mathbb{Z}_p^*$ at random and outputs a function π such that $\pi(x) = \xi^x$ for $x \in \mathbb{Z}_p$. Since ξ is a primitive element, π is injective. The following assumption can be seen as a variant of the Decisional Diffie-Hellman (DDH) assumption.

Assumption I. We say that $\mathcal{G}$ satisfies Assumption I if for any well-spread probability ensemble $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$ with $[\mathbf{X}_k] \subseteq \mathbb{Z}_p$, and any PPT algorithm $\mathcal{B}$,

$$\left|\Pr[I = (p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}) \leftarrow \mathcal{G}(1^k)\ ;\ x_0, x_1 \leftarrow \mathbf{X}_k\ ;\ \xi_1, \xi_2 \leftarrow \mathsf{PRIM}(p)\ ;\right.$$
$$\left. b \leftarrow \{0,1\}\ ;\ b' \leftarrow \mathcal{B}(1^k, I, \xi_1, \xi_2, \xi_1^{x_0}, \xi_2^{x_b})\ :\ b = b'] - \frac{1}{2}\right|$$

is negligible, where $\mathsf{PRIM}(p)$ is a set of all primitive elements in $\mathbb{Z}_p$.

From the definitions, the following proposition is clear.

Proposition 1. *If $\mathcal{G}$ satisfies Assumption I, then $(\mathcal{G}, \mathfrak{G})$ is secure.*

Let $(\mathcal{G}, \mathfrak{G})$ be the injective-function generator mentioned above. We then define the PEKS scheme $\Pi_1 = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ as follows.

- $\mathsf{KG}(1^k)$: For a security parameter k, run $\mathcal{G}$ and $\mathfrak{G}$ to obtain $I = (p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}) \leftarrow \mathcal{G}(1^k)$ and $\pi(x) = \xi^x \leftarrow \mathfrak{G}(I)$. Pick $g, h \in \mathbb{G}^*$ and $\alpha \in \mathbb{Z}_p$ at random, set $PK = (I, \pi, g, g_1 = g^\alpha, h)$ and $SK = \alpha$, and output (PK, SK), where (I, g) are common parameters for all users in this system.
- $\mathsf{Td}(SK, w)$: To generate a trapdoor for a keyword $w \in \mathbb{Z}_p$ under the secret key SK, pick a random $r_w \in \mathbb{Z}_p$ and output $T_w = (r_w, h_w = (hg^{-r_w})^{\frac{1}{\alpha - \pi(w)}})$. Note that the same r_w is used for the same keyword w.
- $\mathsf{PEKS}(PK, w)$: To encrypt a keyword w under the public key PK, pick random $s \in \mathbb{Z}_p$ and $R \in \mathbb{G}_T$, and output $C_w = (R,\ C_1 = g_1^s g^{-s\pi(w)},\ C_2 = \mathbf{e}(g,g)^s,\ C_3 = R \cdot \mathbf{e}(g,h)^{-s})$.
- $\mathsf{Test}(PK, T_w, C_w)$: Using the notation in the description of Td and PEKS, if $R = C_3 \cdot \mathbf{e}(C_1, h_w) C_2^{r_w}$ then output 1; otherwise, output 0.

The Gentry IBE scheme is shown to be anonymous and IND-ID-CPA secure under the truncated decision ABDHE assumption (see [15] for details). Therefore, from Theorem 4.2 in [1] and Theorem 2 (b), Π_1 is IND-PEKS-CKA secure under the same assumption, and it is computationally consistent. In addition, from Proposition 1 and Theorem 2 (a), Π_1 is IND-PKP secure under Assumption I. However, Π_1 is not IND-PKP+ secure because the trapdoor in Π_1 is uniquely determined per the keyword. An instance of a PEKS scheme that is both IND-PEKS-CKA and IND-PKP+ secure is given in Section 4.4.

4.3 Instance 2

In this section, we present an efficient PEKS scheme that is IND-PKP and IND-PEKS-CKA secure in the RO model, without depending on a secure injective-function generator in its construction. To achieve IND-PKP security, this instance requires no cryptographic assumption beyond those for achieving IND-PEKS-CKA security. This scheme is based on the PEKS scheme proposed in [5], with slight modification. Let $\mathcal{G}$ be a bilinear group generator. We begin by describing the PEKS scheme $\Pi_2 = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ associated with $\mathcal{G}$.

- $\mathsf{KG}(1^k)$: For a security parameter k, run $\mathcal{G}$ to obtain $(p, \mathbb{G}, \mathbb{G}_1, \mathbf{e}) \leftarrow \mathcal{G}(1^k)$, and select $a \in \mathbb{Z}_p$ and $g \in \mathbb{G}^*$ (i.e., g is a generator of $\mathbb{G}$) at random. Set $PK = (p, g, \mathbb{G}, \mathbb{G}_1, \mathbf{e}, g, h = g^a, H_1, H_2)$ and $SK = (PK, a)$, where H_1 and H_2 are hash functions, and $(p, \mathbb{G}, \mathbb{G}_1, \mathbf{e}, g, H_1, H_2)$ are common parameters for all users in this system, and output (PK, SK).
- $\mathsf{Td}(SK, w)$: As a trapdoor for a keyword $w \in \{0,1\}^*$ under the secret key $SK = (PK, a)$, output $T_w = H_1(PK||w)^a \in \mathbb{G}$.
- $\mathsf{PEKS}(PK, w)$: To encrypt a keyword w under the public key PK, pick a random $r \in \mathbb{Z}_p$ and output $C_w = (C_1 = g^r,\ C_2 = H_2(\mathbf{e}(H_1(PK||w), h^r)))$.
- $\mathsf{Test}(PK, C_w, T_w)$: Using the notation in the description of Td and PEKS, if $H_2(\mathbf{e}(T_w, C_1)) = C_2$, then output 1; otherwise, output 0.

The consistency of the above PEKS scheme is shown in [1]. As compared to the original scheme, the input to H_1 includes a public key in Π_2. This modification does not collapse the IND-PEKS-CKA security of the scheme because it can be seen as the original PEKS scheme with a special keyword form. Therefore, like the original scheme, this scheme can be shown IND-PEKS-CKA secure in the RO model under the BDH assumption [5]. Interestingly, IND-PKP security of Π_2 can be shown only under the RO assumption (i.e., without a computational assumption).

Proposition 2. *Suppose that H_1 and H_2 are ROs. For any probability ensemble $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$ and any IND-PKP adversary $\mathcal{A}$ against Π_2 that makes at most $q_{H_1}(k)$ queries to H_1 and at most $q_t(k)$ trapdoor queries when the security parameter k is given,*

$$\mathsf{Adv}^{\text{ind-pkp}}_{\mathcal{A},\Pi_2,\mathcal{X}}(k) \leq 2(q_t(k) + q_{H_1}(k)) \cdot ||\mathbf{X}_k|| + 2^{-k} \quad (k \in \mathbb{N}).$$

The proof of Proposition 2 is given in the full version of this paper.

Remark 6. From Definition 3, the original PEKS scheme in [5] is shown directly to has a PKP under the RO assumption. This is because in the trapdoor, the keyword is hidden by the RO H_1, and the ciphertexts can be created from the trapdoor. However, it will be impossible to show the IND-PKP security of this scheme only under the RO assumption. If the DDH assumption (on $\mathbb{G}$) is added, then the scheme is shown to be IND-PKP secure. In this sense, IND-PKP security can be regarded as a strictly stronger notion than PKP. On the other hand, if H_1 and H_2 are freshly chosen in each key generation (not used as common parameters), then the original scheme is shown to be IND-PKP secure only under the RO assumption.

4.4 Instance 3

As described in Section 3.1, achievement of both IND-PEKS-CKA and IND-PKP+ securities can be considered as the highest security in a PEKS system. Unfortunately, we could not find an appropriate instance within any existing schemes (even allowing for slight modification). We then present a new PEKS scheme that is IND-PKP+ and IND-PEKS-CKA secure in the RO model. Let $\mathcal{G}_3$ be a 3-composite bilinear group generator. We begin by describing the PEKS scheme $\Pi_3 = (\mathsf{KG}, \mathsf{Td}, \mathsf{PEKS}, \mathsf{Test})$ associated with $\mathcal{G}_3$.

- $\mathsf{KG}(1^k)$: For a security parameter k, run $\mathcal{G}_3$ to obtain $I = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, \mathbf{e}) \leftarrow \mathcal{G}_3(1^k)$ and set $N = p_1p_2p_3$. Pick $g_i \in \mathbb{G}_{p_i}^*$ $(1 \le i \le 3)$ and $R_2 \in \mathbb{G}_{p_2}$ at random. Set $PK = (N, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g_2, g_3, g = g_1R_2, H)$ and $SK = (PK, g_1)$, where H is a hash function from $\{0,1\}^*$ to $\mathbb{G}_{p_1}$, and output (PK, SK).
- $\mathsf{Td}(SK, w)$: To generate a trapdoor of a keyword $w \in \{0,1\}^*$ under the secret key SK, pick $s \in \mathbb{Z}_{p_1}$ and $R_3, S_3 \in \mathbb{G}_{p_3}$ at random, and output $T_w = (T_1 = g_1^s R_3,\ T_2 = H(w)^s S_3)$.
- $\mathsf{PEKS}(PK, w)$: To encrypt a keyword w under the public key PK, pick $r \in \mathbb{Z}_N$ and $Y_2, Z_2 \in \mathbb{G}_{p_2}$ at random, and output $C_w = (C_1 = g^r Y_2,\ C_2 = H(w)^r Z_2)$.
- $\mathsf{Test}(PK, C_w, T_w)$: Using the notation in the description of Td and PEKS, if $\mathbf{e}(T_1, C_2) = \mathbf{e}(T_2, C_1)$, then output 1; otherwise, output 0.

From the orthogonality property, the completeness and consistency of Π_3 can readily be verified. To show the security of Π_3, we introduce the following assumptions.

Assumption II. We say that $\mathcal{G}_3$ *satisfies Assumption II* if for any PPT algorithm $\mathcal{B}$,

$$\mathsf{Adv}^{\text{A-2}}_{\mathcal{A},\mathcal{G}_3}(k) = \Big| \Pr\Big[\ I \leftarrow \mathcal{G}_3(1^k)\ ;\ N \leftarrow p_1p_2p_3\ ;\ g_i \leftarrow \mathbb{G}_{p_i}^* \quad (1 \le i \le 3)\ ;$$
$$X_0, X_1 \leftarrow \mathbb{G}_{p_1}\ ;\ s, s' \leftarrow \mathbb{Z}_N\ ;\ R_3, S_3, R_3', S_3' \leftarrow \mathbb{G}_{p_3}\ ;\ b \leftarrow \{0,1\}\ ;$$
$$b' \leftarrow \mathcal{B}\left(1^k, N, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g_1, g_2, g_3, (g_1^s R_3, X_0^s S_3), (g_1^{s'} R_3', X_b^{s'} S_3')\right)\ :\ b = b'\ \Big] - \frac{1}{2}\Big|$$

is negligible.

Assumption III. We say that $\mathcal{G}_3$ *satisfies Assumption III* if for any PPT algorithm $\mathcal{B}$,

$$\mathsf{Adv}^{\text{A-3}}_{\mathcal{A},\mathcal{G}_3}(k) = \Big| \Pr\Big[I \leftarrow \mathcal{G}_3(1^k)\ ;\ N \leftarrow p_1p_2p_3\ ;\ g_i \leftarrow \mathbb{G}_{p_i}^*\ (1 \le i \le 3)\ ;\ \alpha, \beta \leftarrow \mathbb{Z}_N^*\ ;$$
$$X_0, X_1 \leftarrow \mathbb{G}_{p_1}\ ;\ r \leftarrow \mathbb{Z}_N\ ;\ R_2, Y_2, Z_2 \leftarrow \mathbb{G}_{p_2}\ ;\ g \leftarrow g_1R_2\ ;\ b \leftarrow \{0,1\}\ ;$$
$$b' \leftarrow \mathcal{B}\left(1^k, N, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g_2, g_3, g_1^\alpha, g_1^\beta, g_1^{\alpha\beta}, g, X_0, X_1, (g^r Y_2, X_b^r Z_2)\right) : b = b'\Big] - \frac{1}{2}\Big|$$

is negligible.

Assumption IV. We say that $\mathcal{G}_3$ *satisfies Assumption IV* if for any PPT algorithm $\mathcal{B}$,

$$\mathsf{Adv}^{\text{A-4}}_{\mathcal{A},\mathcal{G}_3}(k) = \left| \Pr\left[I \leftarrow \mathcal{G}_3(1^k)\ ;\ N \leftarrow p_1p_2p_3\ ;\ h_0, h_1 \leftarrow (\mathbb{G}_T)_{p_1}\ ;\ \alpha, \beta \leftarrow \mathbb{Z}_N^*\ ; \right.\right.$$
$$\left.\left. b \leftarrow \{0,1\}\ ;\ b' \leftarrow \mathcal{B}\left(1^k, N, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, h_0, h_1, h_b^{\alpha}, h_b^{\beta}, h_b^{\alpha\beta}\right)\ :\ b = b' \right] - \frac{1}{2} \right|$$

is negligible.

It can readily be shown that if $\mathcal{G}_3$ satisfies Assumption II then it also satisfies the DDH assumption over $(\mathbb{G}_T)_{p_1}$. Thus, Assumption II is a stronger assumption than the DDH assumption over $(\mathbb{G}_T)_{p_1}$. Assumption IV is presented to explain the position of Assumption III but with a simpler representation. Proposition 3 says that Assumption III is a stronger assumption than Assumption IV its proof is straightforward and left to the reader.

Proposition 3. *If $\mathcal{G}_3$ satisfies Assumption III, then it also satisfies Assumption IV.*

Proposition 4. *Suppose that H is an RO. For any probability ensemble $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$ and any SPP adversary $\mathcal{A}$ that makes at most $q_H(k)$ queries to H and at most $q_t(k)$ trapdoor queries when the security parameter k is given, there exists a PPT algorithm $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\text{spp}}_{\mathcal{A},\Pi_3,\mathcal{X}}(k) \leq \mathsf{Adv}^{\text{A-2}}_{\mathcal{B},\mathcal{G}_2}(k) + 2(q_H(k) + q_t(k)) \cdot ||\mathbf{X}_k|| \quad (k \in \mathbb{N}).$$

Proposition 5. *Suppose that H is an RO. For any probability ensemble $\mathcal{X} = \{\mathbf{X}_k\}_{k\in\mathbb{N}}$ and any IND-PKP adversary $\mathcal{A}$ against Π_3 that makes at most $q_H(k)$ queries[3] to H and at most $q_t(k)$ trapdoor queries when the security parameter k is given,*

$$\mathsf{Adv}^{\text{ind-pkp}}_{\mathcal{A},\Pi_3,\mathcal{X}}(k) \leq 2(q_H(k) + q_t(k)) \cdot ||\mathbf{X}_k|| \quad (k \in \mathbb{N}).$$

Proposition 6. *Suppose that H is an RO. For any IND-PEKS-CKA adversary $\mathcal{A}$ against Π_3 that makes at most $q_H(k)$ queries to H when the security parameter k is given, there exists a PPT algorithm $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\text{ind:peks}}_{\mathcal{A},\Pi_3}(k) \leq (q_H(k) + 1)(q_H(k) + 2) \cdot \mathsf{Adv}^{\text{A-3}}_{\mathcal{B},\mathcal{G}_3}(k) \quad (k \in \mathbb{N}).$$

The proofs of Propositions 4, 5, and 6 are given in the full version of this paper. The open problem is to construct a PEKS scheme that is IND-PKP+ secure and IND-PEKS-CKA secure, either in the standard model or the RO model, under reasonable assumptions.

[3] In the PEKS scheme Π_3, H cannot be used as a common parameter for all users because Π_3 depends on a composite bilinear map. Hence, an IND-PKP adversary can make queries to both H and H'. For simplicity, we assume that the total numbers of queries to both H and H' is written by $q_H(k)$.

Remark 7. We now consider a PEKS scheme that is identical with Π_3 except that the searchable ciphertext of w is given by $C_w = (C_1 = g^r,\ C_2 = H(w)^r)$. From a similar discussion to that for Proposition 5, it can be shown that this PEKS scheme is IND-PKP secure; however, it is not IND-PEKS-CKA secure. This is because for a given target ciphertext $C_{w_b} = (C_1, C_2)$, an adversary can easily guess the challenge bit b by outputting $b' \in \{0, 1\}$ such that $\mathbf{e}(g, C_2) = \mathbf{e}(C_1, H(w_{b'}))$. This instance demonstrates the separation between the IND-PKP and IND-PEKS-CKA securities.

5 Postscript

We have introduced new security notions for PEKS systems, namely PKP, IND-PKP, PKP+, and IND-PKP+, which take account of the privacy of a keyword from a trapdoor. We have also showed that these notions ensure strictly weaker security with respect to keyword leakage from only the ciphertext, as compared to IND-PEKS-CKA security. Accordingly, for achieving higher security in PEKS, we have presented several instances of a PEKS scheme that is IND-PKP or IND-PKP+ secure, in addition to being IND-PEKS-CKA secure. From a practical viewpoint, however, we have no corroboration that either IND-PKP or IND-PKP+ security is insufficient to ensure the privacy of a keyword from a ciphertext. We expect that the underlying notion and PRIV security [3] give equal security levels, because they are defined for the situation in which the target keywords are chosen from a well-spread distribution, and the (guessing) adversary cannot see them. We are sure that it is easier to design efficient IND-PKP (or IND-PKP+) secure PEKS schemes than it is to design efficient IND-PEKS-CKA secure PEKS schemes. Indeed, we can use secure injective-function generators to achieve IND-PKP security, and it is easy to design practical injective-function generators that are secure under reasonable assumptions.

Acknowledgements. We would like to thank anonymous referees of ProvSec 2012 for their valuable comments.

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. Journal of Cryptology 21(3), 350–391 (2008)
2. Bellare, M., Boldyreva, A., Micali, S.: Public-Key Encryption in a Multi-user Setting: Security Proofs and Improvements. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 259–274. Springer, Heidelberg (2000)
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. Blundo, C., Iovino, V., Persiano, G.: Predicate Encryption with Partial Public Keys. In: Heng, S.-H., Wright, R.N., Goi, B.-M. (eds.) CANS 2010. LNCS, vol. 6467, pp. 298–313. Springer, Heidelberg (2010)

5. 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)
6. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
7. Boneh, D., Sahai, A., Waters, B.: Functional Encryption: Definitions and Challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011)
8. Boyen, X., Waters, B.: Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
9. Camenisch, J., Kohlweiss, M., Rial, A., Sheedy, C.: Blind and Anonymous Identity-Based Encryption and Authorised Private Searches on Public Key Encrypted Data. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 196–214. Springer, Heidelberg (2009)
10. 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)
11. Canetti, R., Micciancio, D., Reingold, O.: Perfectly one-way probabilistic hash functions. In: Proceedings of the 30th ACM STOC 1998, pp. 131–140 (1998)
12. De Caro, A., Iovino, V., Persiano, G.: Hidden vector encryption fully secure against unrestricted queries. IACR Cryptology ePrint Archive, Report 2011/546, http://eprint.iacr.org/2011/546
13. Cocks, C.: An Identity Based Encryption Scheme Based on Quadratic Residues. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (2001)
14. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: Improved definitions and efficient constructions. In: Proceedings of the 13th ACM Conference on Computer and Communication Security, pp. 79–88 (2006)
15. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
16. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
17. 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)
18. 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)
19. Waters, B., Balfanz, D., Durfee, G., Smetters, D.K.: Building an encrypted and searchable audit log. In: NDSS. The Internet Society (2004)

Efficient Fully Secure Attribute-Based Encryption Schemes for General Access Structures

Tapas Pandit and Rana Barua

Indian Statistical Institute
Kolkata -700108
{pandit_r,rana}@isical.ac.in

Abstract. In this paper, we present an efficient ciphertext-policy attribute based encryption (CP-ABE) scheme with "short" ciphertext and a key-policy attribute based encryption (KP-ABE) scheme with "short" key for monotone access structures (MAS) which are fully secure in the standard model over composite order bilinear groups. We obtain our schemes by using a simple "encoding technique", representing the monotone access structure by their minimal sets only, thereby obtaining schemes whose ciphertext size or key size depends on number of minimal sets. Most of the recent CP-ABE/KP-ABE schemes have ciphertext size or key size roughly of the order of the size of the monotone span program (MSP) or the number of attributes. Consequently, our schemes will, in general, have shorter ciphertext or shorter key. To illustrate, we give examples of MAS where the number of minimal sets is constant whereas the size of the corresponding MSP is linear in the number of attributes. Using similar ideas, we show how to obtain a CP-ABE scheme with *constant size key* and a Hierarchical (H) KP-ABE scheme with *constant size ciphertext* for arbitrary access structures (not necessarily monotone) which are also fully secure in the standard model under three static assumptions over composite order bilinear groups. To date, for all general policies, the decryption cost is polynomial in the number of qualified rows in the span programs. But in all of our proposed schemes, the decryption cost is *contant* for general access structures.

Keywords: Attribute Based Encryption, Ciphertext-Policy, Key-Policy.

1 Introduction

Identity-Based Encryption (IBE) [1, 5] allows a sender to encrypt a message for an identity without access to a public key certificate. One drawback of IBE is that a sender encrypts a message for a particular recipient and not for a group of recipients. Thus IBE is still coarse-grained. In many distributed systems, a secret data may be accessible to different users depending upon their positions in these settings. Recently functional encryption has been introduced as a more sophisticated solution to handle the generic access policy in many distributed systems,

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 193–214, 2012.

where an encryptor allows the secret key holder to "conditionally decrypt" the ciphertext. More precisely, the ciphertext or the message M (resp. key) is associated with a predicate or a function, say f and the key (resp. ciphertext) is labeled with a set of attributes or a solution vector A of f and the decryption algorithm outputs the message M if and only if $f(A) = 1$. Functional encryption is categorized into two classes, one is ABE (sometimes called payload hiding or message hiding) and the other is Predicate Encryption (PE) (called attribute hiding). Sahai and Waters [8] first proposed the concept of attribute-based encryption, a new type of IBE scheme called Fuzzy Identity-Based Encryption in which, an identity was viewed as a set of descriptive attributes. Consequently, Goyal et al. [9] introduced the idea of a more general Key-policy ABE system where a policy (represented by an access tree) is associated with the key and a set of attributes with the ciphertext. However, the construction of a CP-ABE, where the policy is associated with the ciphertext and a set of attributes with the key, was left open. Finally, Bethencourt et al. [11] came up with a ciphertext-policy ABE construction. In both the schemes the policy was taken to be a monotone access structure, more precisely an access tree and the encrypting data are shared at a fine-grained level. R.Ostrovsky et al. [13] first introduced a KP-ABE scheme that can handle non-monotone access structures over attributes and they showed how the user's private key is associated with any access formula over attributes. The key idea of their scheme was converting a non-monotone access structure over attributes to a monotone access structure over attributes and their negation. Subsequently, a variety of ABE [10, 12, 15, 20–22] has been proposed all of which (except [22]) were shown to be selectively secure.

Lewko et al. [22] are the first who proposed a CP-ABE scheme and a KP-ABE scheme using monotone span programs (MSP), both of which were fully secure in the standard model over composite order bilinear groups. In their schemes, either ciphertext size (in case of CP-ABE) or key size (in case of KP-ABE) was polynomial in size of monotone span program and the number of pairing computations in the decryption algorithm in both the schemes was polynomial in the number of qualified rows in monotone span program to compute the target vector.

Okamoto and Takashima [24] proposed KP-ABE and CP-ABE schemes for non-monotone access structures which are fully secure under a standard assumption, the decisional linear (DLIN) assumption, in the standard model over prime order bilinear groups. In [24], either the ciphertext size or the key size is polynomial in the number of rows in the non-monotone span program and the number of pairing computations in the decryption algorithm in both the schemes is polynomial in the number of qualified rows in the non-monotone span program for computing the target vector.

Brent Waters [26] proposed three efficient selectively secure CP-ABE constructions using MSP in the standard model under three different hardness assumptions in prime order bilinear groups. In his schemes, the ciphertext size was either polynomial in the size of the monotone span program or some factor times the size of the monotone span program and the number of pairing computations

in the decryption algorithm is polynomial in the number of qualified rows in the monotone span program.

Recently, Attrapadung et al. [28] proposed a KP-ABE scheme for non-monotone access structures with constant size ciphertext which is selectively secure in the standard model over prime order bilinear group. To get this scheme, they constructed a certain class of identity based broadcast encryption (IBBE) schemes and identity-based revocation (IBR) mechanisms. They also employed the technique of R.Ostrovsky et al. [13] to convert the non-monotone access structure to monotone access structure with negative attributes. The key size in their scheme is polynomial in size of the monotone span program. The number of exponents in the decryption algorithm is polynomial in number of qualified rows in the monotone span program to compute the target vector.

In most of the access control systems, the secret data are stored in a distributed fashion across many servers. The same (even different) encrypted data are accessed by many legitimate users on the same server at a time. To get faster accessibility of the secret data, the constant time decryption algorithms are always welcome. To the best of our knowledge, our ABE schemes are the only schemes which take constant time on decryption for general policies.

There is a close relation between linear secret sharing scheme (LSSS) and monotone span program. The existence of an efficient LSSS for a specific monotone access structure is equivalent to the existence of a smallest monotone span program for the characteristic function of that monotone access structure (see [4]). V.Nikova et al. [6] gave a theoretical lower bound for any monotone span program using some linear algebraic machineries. Informally, the size of a monotone span program is at least the size of the critical set of minimal sets for the corresponding monotone access structure plus the size of the critical set for the minimal sets of the dual of the access structure minus one (see [6] and section 2). Using this lower bound, we show that there exist some classes of monotone access structures for which the size of monotone span program is at least polynomial in the number of attributes in the access structure, but the number of minimal sets in the access structures is constant (see section 2.1). It is not known if there is any explicit relation between the size of an access structure and the size of the smallest span program computing it. Since the size of a smallest monotone span program for the characteristic function of a monotone access structure is equivalent to the size an efficient LSSS realizing that access structure, for getting the exact size of a small span program, we have to look at existing efficient LSSS. Most of [18, 23, 26, 27] used the techniques of generalized secret sharing and monotone functions due to J.Benaloh et al. [2]. For threshold policy, the size of the MSP is linear in the number of attributes but the number of minimal sets may be exponential (say, in case of (n, n/2)-threshold policy) in the number of attributes. Informally, for other monotone policies, where the threshold functionality can not be applied, the size of the monotone span programs are the sum of the frequencies of all attributes present in the corresponding minimal sets (see monotone circuit construction in [3] which is an explication of [2]). Hence for such policies, the size of the monotone span programs is at least the number

of minimal sets. In such cases, our CP-ABE or PK-ABE schemes would have shorter ciphertext or shorter key size.

Our Contribution

- We propose a CP-ABE scheme for monotone access structures which has ciphertext of polynomial size in the number of minimal sets and with constant number of bilinear pairing computations during decryption. If $|\mathcal{B}|$ is the number of minimal sets, then in our CP-ABE, the size of cipher-text is $2|\mathcal{B}| + 2$ and key size is polynomial in the number of associated attributes. The number of bilinear pairing computations is 3 during decryption.
- We also present a CP-ABE scheme for non-monotone access structures with constant size key which has constant number of bilinear pairing computations during decryption.
- Further, we propose a KP-ABE scheme for monotone access structures which has key of polynomial size in the number of minimal sets and constant number of bilinear pairing computations and constant number of exponent computations during decryption and which is fully secure in the standard model.
- Finally, we present a hierarchical (H)KP-ABE scheme for non-monotone access structures with constant size key, has a constant time bilinear computations and constant number of exponent computations during decryption and which is fully secure in the standard model.

All our schemes have been shown to be fully secure in the standard model over composite order bilinear groups. In table[1][1-4], we give the comparisons between our schemes and the existence schemes.

Our Technique Since any monotone access structure is uniquely represented by its minimal sets of, by adopting the idea of Emura et al. [20], we build a simple technique to give an efficient ABE scheme for monotone access structures. It is as follows.

- The secret key components of a user are labeled with associated attributes.
- The ciphertext components are labeled with the minimal sets in the monotone access structure Γ.
- For an authorized member of Γ, her set of attributes must be a superset of a minimal set, say S, of Γ.
- To get the message, she pools her key components labeled with the attributes from S and ciphertext components labeled by the minimal set S in the decryption algorithm.

By a similar technique, we give an ABE scheme either with constant size key or constant size ciphertext for non-monotone access structures. The idea is as follows.

[1] *The description of all the symbols in table [2,3,4] are found at the bottom of the table 1. Since a HKP-ABE scheme is nothing but KP-ABE scheme with a delegate algorithm, when we compare our HKP-ABE scheme with the existence KP-ABE schemes, we just ignore the delegate algorithm like in table 4.*

- All the secret key components labeled with the attributes from an authorized set in a non-monotone access structure are multiplied to get a single component. A user cannot extract the the individual components from this single component.
- The ciphertext components are labeled with the authorized sets of the non-MAS Γ.
- For an authorized member, her set of attributes must be a member, say S, of Γ.
- To get the message, she pools a constant number of key components and ciphertext components labeled with S in the decryption algorithm

Note. The above techniques are described in terms of CP-ABE. To understand the techniques for KP-ABE, just interchange the role of the access policy and the set of attributes.

2 Preliminaries

Definition 1 (Access Structure). *Let $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ be a set of attributes. A collection $\Gamma \subset 2^{\mathcal{P}}$ is said to be monotone if Γ is closed under superset, i.e, if $\forall$ B, C: if $B \in \Gamma$ and $B \subset C$, then $C \in \Gamma$. An access structure (respectively, MAS) is a collection (respectively, monotone collection) Γ of non-empty subsets of $\mathcal{P}$, i.e., $\Gamma \subset 2^{\mathcal{P}} \setminus \{\emptyset\}$. The members of Γ are called authorized sets, and the sets not in Γ are called unauthorized sets.*

Definition 2 (Minimal set of a Monotone Access Structure). *Let Γ be a monotone access structure over the set of attributes $\mathcal{P}$. Then $B \in \Gamma$ is a minimal authorized set if $\forall A \in \Gamma \setminus \{B\}$, we have $A \not\subset B$. The set of all minimal sets in Γ is called the basis of Γ.*

Definition 3 (Dual of an Access Structure). *The dual access structure $\Gamma^{\perp}$ of an access structure Γ, defined on $\mathcal{P}$, is the collection of sets $A \subset \mathcal{P}$ such that $\mathcal{P} \setminus A = A^c \notin \Gamma$.*

Definition 4 (Critical Set of Minimal Sets). [d] *Let $\mathcal{B} = \{X_1, X_2, .., X_r\}$ be the set of minimal sets of an access structure Γ. Let $\mathcal{H} \subset \mathcal{B}$ be a subset of the set of minimal sets. We say that a subset $\mathcal{H} \subset \mathcal{B}$ is a critical set of minimal sets for $\mathcal{B}$, if every $X_i \in \mathcal{H}$ contains a set $B_i \subset X_i, |B_i| \geq 2$, such that the following two conditions are satisfied.*

- *The set B_i uniquely determines X_i in the set $\mathcal{H}$. That is, no other set in $\mathcal{H}$ contains B_i.*
- *For any subset $Y \subset B_i$, the set $S_Y = \cup_{X_j \in \mathcal{H}, X_j \cap Y \neq \emptyset}(X_j \setminus Y)$ does not contain any member of $\mathcal{B}$.*

Theorem 1. [d] *Let Γ be an access structure and $\Gamma^{\perp}$ be its dual, let $\mathcal{H}$ be a critical set of minimal sets for Γ and let $\mathcal{H}^{\perp}$ be a critical set of minimal sets for $\Gamma^{\perp}$. Then the size of any monotone span program $\mathcal{M}$ computing Γ is bounded from below by $|\mathcal{H}| + |\mathcal{H}^{\perp}| - 1$.*

2.1 Example

In this section, we explore some examples of monotone access structures such that the number of minimal sets is constant but the size of the monotone span program computing the access structure is at least polynomial in the number of attributes.

Example 1. Let $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ be a set of attributes. Let $\mathcal{B} = \{X_1 = \{P_1, P_2\}, X_2 = \{P_3, ..., P_n\}\}$ be the set of minimal sets for a monotone access structure Γ. Then it is easy to check that $\mathcal{H} = \mathcal{B}$. Now, the set of minimal sets for the dual of Γ is $\mathcal{B}^{\perp} = \{\{P_1, P_3\}, ..\{P_1, P_n\}, \{P_2, P_3\}, .., \{P_2, P_n\}\}$. Then we can find a critical set $\mathcal{H}^{\perp}$ for $\Gamma^{\perp}$ as $\{\{P_1, P_3\}, ..., \{P_1, P_n\}\}$. So by the above theorem, the size of the monotone span program computing Γ is at least $2 + (n-2) - 1 = n - 1 = \mathcal{O}(n)$.

Example 2. Let $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ be a set of attributes. Let $\mathcal{B} = \{X_1 = \{P_1, ...P_{n/2}\}$, $X_2 = \{P_{n/2+1}, ..., P_n\}\}$ be the set of minimal sets for a monotone access structure Γ. Then we can find a critical set $\mathcal{H}^{\perp}$ for $\Gamma^{\perp}$ as $\{\{P_1, P_{n/2}\}, ..., \{P_1, P_n\}\}$. So by the above theorem, the size of the monotone span program computing Γ is at least $2 + (n/2) - 1 = n/2 + 1 = \mathcal{O}(n)$.

Example 3. Let $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ be a set of attributes. Let $\mathcal{B} = \{X_1 = \{P_1, ...P_{n/3}\}$, $X_2 = \{P_{n/3+1}, ..., P_{2n/3}\}, X_3 = \{P_{2n/3+1}, ..., P_n\}\}$ be the set of minimal sets for a monotone access structure Γ. Then we can find a critical set $\mathcal{H}^{\perp}$ for $\Gamma^{\perp}$ as $\{\{P_1, P_{n/3+1}, P_{2n/3+1}\}, ..., \{P_1, P_{n/3+1}, P_n\}\}$. So the size of the monotone span program computing Γ is at least $3 + (n/3) - 1 = n/3 + 2 = \mathcal{O}(n)$.

Similarly, one can choose constant number of different size minimal sets for a monotone access structure, where the minimal sets may intersect but the size of the monotone span program will be at least $\mathcal{O}(n)$.

Note. Thus if one uses such MAS, our schemes will have either short ciphertext or short key.

2.2 CP-ABE

A CP-ABE consists of four probabilistic polynomial time (in short PPT) algorithms which are following:

- **Setup**(1^{λ},$\mathcal{U}$) It takes a security parameter 1^{λ} and a universe of attributes $\mathcal{U}$ as input and it outputs public parameters `Pp` and a master secret key `Msk`.
- **KeyGen**(`Msk`,A): It takes as input the master secret key `Msk` and a set of attributes A and outputs a secret key Sk_A.
- **Encrypt**(`Pp`,Γ,M) The algorithm takes public parameter `Pp`, an access structure Γ over the universe of attributes $\mathcal{U}$ and a message M as input and outputs a ciphertext Ct_{Γ} such that a user whose set of attributes satisfies the access structure Γ, can extract the message M. The ciphertext Ct_{Γ} implicitly contains the access structure Γ.
- **Decrypt**(Ct_{Γ},Sk_A) It takes as input a ciphertext Ct_{Γ} corresponding to an access structure Γ and a secret key Sk_A associated with a set of attributes

A. If the set of attributes A satisfies the access policy Γ, then the algorithm outputs the message M of the corresponding ciphertext Ct_{Γ}

2.3 Security Definition for CP-ABE

We now formalize the full (adaptive) security model against chosen plaintext attacks (CPA) for CP-ABE. This is a game denoted by $Game_{Real}$ between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. We define the game $Game_{Real}$ in the following way.

- **Setup** $\mathcal{C}$ runs the Setup algorithm on input a security parameter 1^{λ} and a universe of attributes $\mathcal{U}$ to generate public parameters `Pp` and a master secret key `Msk`. The challenger $\mathcal{C}$ starts the interaction with the adversary $\mathcal{A}$ by giving the public parameter `Pp`.
- **Phase 1** The adversary $\mathcal{A}$ queries the challenger $\mathcal{C}$ for the secret keys labeled with the sets of attributes $A_1, A_2, .., A_l$.
- **Challenge Phase** The adversary gives two equal length messages M_0 and M_1 and an access structure Γ^* such that it is not satisfied by the queried sets of attributes $A_1, A_2, ..., A_l$. The challenger randomly chooses a bit b from $\{0, 1\}$ and encrypts the message M_b using the access structure Γ^* and gives the challenge ciphertext Ct_{Γ^*} to the adversary $\mathcal{A}$
- **Phase 2** The adversary $\mathcal{A}$ again queries the challenger $\mathcal{C}$ for the secret keys labeled with the sets of attributes $A_{l+1}, A_{l+2}, .., A_q$ with the restriction that no A_i satisfies the access structure Γ^*. The challenger returns the secret key by running the KeyGen algorithm.
- **Guess** The adversary outputs a guess $\hat{b}$ for b.

The advantage $Adv^{\mathcal{A}}_{Game_{Real}}(\lambda)$ of the adversary $\mathcal{A}$ in $Game_{Real}$ is defined by $|Pr(\hat{b} = b) - 1/2|$

Definition 5. *A CP-ABE scheme is said to be fully secure if for all PPT adversary $\mathcal{A}$, the advantage $Adv^{\mathcal{A}}_{Game_{Real}}(\lambda)$ is a negligible function of λ in $Game_{Real}$.*

Remark. If we interchange the role of the access policy and the set of attributes, we get the definition of KP-ABE scheme and KP-ABE security very similar to the above. (see in Appendix A in [22])

2.4 Composite Order Bilinear Groups and Complexity Assumptions

D.Boneh et al. first introduced composite order bilinear groups in [7]. We define a composite order bilinear groups generator $\mathcal{G}$ as an algorithm which takes input λ as a security parameter and outputs a description $\mathcal{I}$ of a composite order bilinear groups $\mathbb{G}$. In this case, $\mathcal{I}$ consists of $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e)$, where p_1, p_2, p_3 are three distinct primes and $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a map such that

1. (Bilinear) $\forall g, h \in \mathbb{G}, a, b \in \mathbb{Z}_N, e(g^a, h^b) = e(g, h)^{ab}$
2. (Non-degenerate) $\exists g \in \mathbb{G}$ such that $e(g, g)$ has order N in $\mathbb{G}_T$

Let $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$ respectively denote the subgroups of $\mathbb{G}$ of order p_1, p_2 and p_3. Let $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$ be arbitrary elements with $i \neq j$, then $e(h_i, h_j) = 1$. This property is called orthogonal property of $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}$ (for details see [23]).

We use the complexity assumptions due to Lewko et al. [23] to prove the full security of our CP-ABE and KP-ABE constructions in the standard model over composite order bilinear groups and the assumptions are described below:

Assumption 1 (Subgroup decision problem for 3 primes). *Choose a composite order bilinear group generator $\mathcal{G}$. Define the following distribution : $\mathcal{I} = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and then choose random $g \in \mathbb{G}_{p_1}, X_3 \in \mathbb{G}_{p_3}$ and set $D = (\mathcal{I}, g, X_3)$. Choose random $T_1 \in \mathbb{G}_{p_1p_2}, T_2 \in \mathbb{G}_{p_1}$. Now the advantage of an algorithm $\mathcal{A}$ in breaking Assumption 1 is defined by*

$$Adv_1^{\mathcal{A}}(\lambda) = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$$

Definition 6. *We say that Assumption 1 holds for $\mathcal{G}$ if for all PPT Adversary $\mathcal{A}$, $Adv_1^{\mathcal{A}}(\lambda)$ is a negligible function of λ.*

Assumption 2. *Pick a composite order bilinear group generator $\mathcal{G}$. Define the following distribution : $\mathcal{I} = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and then choose random $g \in \mathbb{G}_{p_1}, X_1X_2 \in \mathbb{G}_{p_1p_2}, X_3 \in \mathbb{G}_{p_3}, Y_2Y_3 \in \mathbb{G}_{p_2p_3}$ and set $D = (\mathcal{I}, g, X_1X_2, X_3, Y_2Y_3)$. Choose random $T_1 \in \mathbb{G}, T_2 \in \mathbb{G}_{p_1p_3}$. Now the advantage of an algorithm $\mathcal{A}$ in breaking Assumption 2 is defined by*

$$Adv_2^{\mathcal{A}}(\lambda) = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$$

Definition 7. *We say that Assumption 2 holds for $\mathcal{G}$ if for all PPT Adversary $\mathcal{A}$, $Adv_2^{\mathcal{A}}(\lambda)$ is a negligible function of λ.*

Assumption 3. *Pick a composite order bilinear group generator $\mathcal{G}$. Define the following distribution : $\mathcal{I} = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and then choose random $\alpha, s \in \mathbb{Z}_N, g \in \mathbb{G}_{p_1}, X_2, Y_2, Z_2 \in \mathbb{G}_{p_2}, X_3 \in \mathbb{G}_{p_3}$ and set $D = (\mathcal{I}, g, g^\alpha X_2, X_3, g^s Y_2, Z_2)$. Let $T_1 = e(g, g)^{\alpha s}$. Choose random $T_2 \in \mathbb{G}_T$. Now the advantage of an algorithm $\mathcal{A}$ in breaking Assumption 3 is defined by*

$$Adv_3^{\mathcal{A}}(\lambda) = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$$

Definition 8. *We say that Assumption 3 holds for $\mathcal{G}$ if for all PPT Adversary $\mathcal{A}$, $Adv_3^{\mathcal{A}}(\lambda)$ is a negligible function of λ.*

2.5 CP-ABE Scheme with Short Ciphertext for MAS

In this section, we present our efficient CP-ABE scheme using very a simple encoding technique and is fully secure in the standard model over composite order bilinear groups. In this construction, the size of the group is $N = p_1p_2p_3$, product of three distinct primes and we frequently use the elements of the subgroup $\mathbb{G}_{p_1}$ for encoding the policy and the set of attributes and we use the random

elements of $\mathbb{G}_{p_3}$ to randomize the key and ciphertext but we do not incorporate the elements of $\mathbb{G}_{p_2}$. We use the dual system proof technique of Brent Waters [19] which requires the concept of semi-functional key and ciphertext and we use the elements of $\mathbb{G}_{p_2}$ to construct such key and ciphertext. In this construction, a monotone access structure over a universe of attributes is represented by set of minimal sets. We use the notation $[m]$ for the set $\{i \in \mathbb{N} : 1 \leq i \leq m\}$

Setup$(1^\lambda, \mathcal{U})$. The setup algorithm runs the composite order bilinear groups generator $\mathcal{G}(1^\lambda)$ to get a random composite order bilinear groups descriptor, $\mathcal{I} = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e)$ with known factorization of N (p_1, p_2, p_3 are distinct primes). It chooses random elements $g \in \mathbb{G}_{p_1}, X_3 \in \mathbb{G}_{p_3}$ and random exponents $a, \alpha \in \mathbb{Z}_N$ and chooses random $t_i \in \mathbb{Z}_N$ for each attribute $i \in \mathcal{U}$ and sets $T_i = g^{t_i}$. The public parameters are $\texttt{Pp} = [N, g, g^a, Y = e(g,g)^\alpha, (T_i)_{i\in\mathcal{U}}]$ and the master secret key is $\texttt{Msk} = [\alpha, X_3]$.

KeyGen(Pp,Msk,A). It chooses random $r \in \mathbb{Z}_N; R_0, R' \in \mathbb{G}_{p_3}$. For each attribute $i \in A$, the algorithm chooses random $R_i \in \mathbb{G}_{p_3}$ and outputs the secret key

$$Sk_A = [K_0 = g^{\alpha+ar}R_0, K' = g^rR', K_i = {T_i}^rR_i, \forall i \in A]$$

Encrypt(Pp,$\mathcal{B}, M$). Here $\mathcal{B}$ is the set of minimal sets for a monotone access structure Γ. Let $\mathcal{B} = \{A_1, A_2, ..., A_m\}$, where $A_i \subset \mathcal{U}\ \forall i \in [m]$ and m is the size of $\mathcal{B}$. The encryption algorithm chooses random $s \in \mathbb{Z}_N$ and $s_i \in \mathbb{Z}_N$ for each $i \in [m]$. The algorithm returns the ciphertext

$$Ct_{\mathcal{B}} = \big[\mathcal{B}, C_0 = MY^s, C_1 = g^s, (C_{1,i} = g^{s_i})_{i\in[m]}, \big(C_{2,i} = g^{as}\big(\prod_{j\in A_i} T_j\big)^{s_i}\big)_{i\in[m]}\big]$$

Decrypt$(Ct_{\mathcal{B}}, Sk_A)$. Suppose A satisfies the access structure Γ, generated by $\mathcal{B}$, then A must be a superset of a minimal set in $\mathcal{B}$. Let $A_j \subset A$ for some $j \in [m]$. Then it computes

$$e(C_{2,j}, K')/\big(e(C_1, K_0).e(C_{1,j}, \prod_{i\in A_j} K_i)\big) = e(g,g)^{-\alpha s}$$

The algorithm outputs $C_0.e(g,g)^{-\alpha s} = M$

2.6 The Security Proof of CP-ABE Scheme for MAS

Suppose a PPT adversary $\mathcal{A}$ makes q number of key queries, then our proof of security considers a sequence of $2q + 3$ games between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. Our proof employs the dual system technique of Brent Water [19] and for that we need to construct the semi-functional ciphertext and key. We use the orthogonal property of $\mathbb{G}_{p_2}$ to generate the semi-functional ciphertext and key. Any legitimate normal key and legitimate semi-functional key can decrypt the semi-functional ciphertext and normal ciphertext respectively. But when a legitimate semi-functional key decrypts a semi-functional ciphertext, it gives a random element in $\mathbb{G}_T$. The semi-functional key and ciphertext are given below:

Table 1. Comparison with our CP-ABE for MAS

Schemes	Security	Assumption	Ciphertext Size	Key Size	Decryption Cost	
					Pairing	Exponent
LOST [22]	Adaptive	*Assmp* 1,2,3	$(2l+1)\|\mathbb{G}\|$	$\mathcal{O}(\|A\|\phi)$	$\mathcal{O}(I)$	$\mathcal{O}(I)$
OT [24]	Adaptive	DLIN	$\mathcal{O}(l\phi)\|\mathbb{G}\|$	$\mathcal{O}(\|A\|\phi)$	$\mathcal{O}(I\phi)$	$\mathcal{O}(I)$
BW [26] Schm1	Selective	d-p-BDHE	$(2l+1)\|\mathbb{G}\|$	$\mathcal{O}(\|A\|)$	$\mathcal{O}(I)$	$\mathcal{O}(I)$
BW [26] Schm2	Selective	d-BDHE	$(l+1)\|\mathbb{G}\|$	$\mathcal{O}(\|A\|\phi)$	$\mathcal{O}(I)$	$\mathcal{O}(I)$
Our CP-ABE	Adaptive	*Assmp* 1,2,3	$(2\|\mathcal{B}\|+1)\|\mathbb{G}\|$	$\mathcal{O}(\|A\|)$	3	None

A= set of attributes in a user's key; l = #rows in a span program $(\mathcal{M}, \rho)$; $|\mathcal{B}|$ = #minimal sets in MAS; ϕ =maximum # of times an attribute can be associated with the rows of a MSP; I= minimum #rows labeled by user's attributes to compute the target vector; d = #sub universe id; Pairing and Exponent respectively denote the #pairing and #exponent computations in $\mathbb{G}$ and $\mathbb{G}_T$; n =maximum #attributes used in encryption. Since in all the schemes in the tables have one $\mathbb{G}_T$ element, so we only mention the number of elements of $\mathbb{G}$ in the ciphertext. We use d-p-BDHE, Assmp and schm for d-parallel BDHE, Assumption and scheme respectively.

Semi-functional Key. We will consider two forms of semi-functional key. The first one is semi-functional key of type 1 and has the following distribution. Let A be a set of attributes for which a semi-functional key will be constructed. Choose random $t, d, r, z_i \in \mathbb{Z}_N, g_2 \in \mathbb{G}_{p_2}, R_0, R', R_i \in \mathbb{G}_{p_3}$ for each $i \in A$. Finally the type 1 key is

$$Sk_A = [K_0 = g^{\alpha+ar} g_2^d R_0, K' = g^r g_2^t R', K_i = {T_i}^r g_2^{z_i} R_i]$$

The other one is semi-functional key of type 2 whose distribution is same as type 1 except the key components K' and K_i do not contain any $\mathbb{G}_{p_2}$ part.

Semi-functional Ciphertext. Let $\mathcal{B}$ be the basis of a monotone access structure Γ. Let $\mathcal{B} = \{A_1, A_2, ..., A_m\}$, where $A_i \subset U \ \forall i \in [m]$ and m is the size of the basis $\mathcal{B}$. Then choose random $s, c, c' \in \mathbb{Z}_N, g_2 \in \mathbb{G}_{p_2}$ and $s_i \in \mathbb{Z}_N$ for each $i \in [m]$. The semi-functional ciphertext is

$$Ct_{\mathcal{B}} = \Big[\mathcal{B}, C_0 = MY^s, C_1 = g^s g_2^c, (C_{1,i} = g^{s_i})_{i\in[m]}, \big(C_{2,i} = g^{as}\big(\prod_{j\in A_i} T_j\big)^{s_i} g_2^{c'}\big)_{i\in[m]}\Big]$$

Note that if a legitimate semi-functional key decrypts a semi-functional ciphertext, we will get an extra term $e(g_2, g_2)^{tc'-cd}$

In the sequence of $2q+3$ games, the first game is $\mathbf{Game}_{Real}$ which is the real CP-ABE security game described in section 2.3 and the second game is $\mathbf{Game}_0$, where all the keys are normal but the ciphertext is semi-functional. For for $k = 1$ to q, we define the next consecutive games are as follow.

$\mathbf{Game}_{k,1}$. The challenge ciphertext is semi-functional. The first $k-1$ keys are semi-functional of type 2, k^{th} key is semi-functional of type 1 and the remaining keys are normal.

$\mathbf{Game}_{k,2}$. The challenge ciphertext is semi-functional. The first k keys are semi functional of type 2 and the remaining keys are normal.

Finally in the last game $\mathbf{Game}_{Final}$, all the keys are semi-functional of type 2 and the ciphertexts are semi-functional but the challenge message is masked with a random element of $\mathbb{G}_T$.

Lemma 1. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{Real}} - Adv^{\mathcal{A}}_{Game_0} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 1.*

Proof. We establish a PPT algorithm $\mathcal{S}$, called Simulator, which takes the parameters, $(\mathcal{I}, g, X_3, T)$ of the Assumption 1 from the challenger $\mathcal{C}$ and depending on the distribution of T, $\mathcal{S}$ simulates either $Game_{Real}$ or $Game_0$.

Setup. $\mathcal{S}$ constructs the public parameters $\mathtt{Pp}$ in the following way. $\mathcal{S}$ chooses random $a, \alpha \in \mathbb{Z}_N$ and $\forall i \in U, t_i \in \mathbb{Z}_N$ and sets $T_i = g^{t_i}$ and starts interaction with the adversary $\mathcal{A}$ by giving

$$\mathtt{Pp} = [N, g, g^a, Y = e(g,g)^{\alpha}, (T_i)_{i \in U}]$$

and $\mathcal{S}$ keeps the master key $\mathtt{Msk} = [\alpha, X_3]$

Key Query Answering. $\mathcal{S}$ handles the $\mathcal{A}$'s key queries by running KeyGen algorithm, since he has the master $\mathtt{Msk}$.

Challenge Phase. $\mathcal{S}$ receives two challenge messages M_0, M_1 and a challenge basis $\mathcal{B}^*$ from the adversary $\mathcal{A}$. $\mathcal{S}$ chooses random $M_b \in \{M_0, M_1\}$. Let $\mathcal{B}^* = \{A_0, A_1, ..., A_m\}$, where each $A_i \subset U$. $\mathcal{S}$ chooses random $s_i \in \mathbb{Z}_N$ and gives the challenge ciphertext $Ct_{\mathcal{B}^*}$ to $\mathcal{A}$

$$Ct_{\mathcal{B}^*} = \big[C_0 = M_b.e(g^{\alpha}, T), C_1 = T, \big(C_{1,i} = g^{s_i}\big)_{i \in [m]}, \big(C_{2,i} = T^a \big(\prod_{j \in A_i} T_j \big)^{s_i}\big)_{i \in [m]}\big]$$

This completes the simulation of algorithm $\mathcal{S}$

Now suppose that $T \in \mathbb{G}_{p_1 p_2}$. Then T can be written as $T = g^s g_2{}^c$ for some $s, c \in \mathbb{Z}_N$. Then $C_0 = M_b.Y^s, C_1 = g^s g_2{}^c, C_{1,i} = g^{s_i}$ and $C_{2,i} = g^{as}\big(\prod_{j \in A_i} T_j\big)^{s_i}$ g_2^{ac} for each $i \in [m]$. Here we implicitly set $c' = ac$. Since *a modulo* p_1 is uncorrelated from *a modulo* p_2 by Chinese Remainder Theorem, so $Ct_{\mathcal{B}^*}$ is a properly distributed semi-functional ciphertext and therefore if $T \in \mathbb{G}_{p_1 p_3}$, $\mathcal{S}$ simulates $Game_0$ for $\mathcal{A}$.

Similarly if $T \in \mathbb{G}_{p_1}$, $\mathcal{S}$ simulates $Game_{Real}$ for $\mathcal{A}$. This concludes the proof of the lemma. □

Lemma 2. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{k-1,2}} - Adv^{\mathcal{A}}_{Game_{k,1}} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 2.*

Proof. We establish a PPT algorithm $\mathcal{S}$ which takes the parameters, $(\mathcal{I}, g, X_1X_2, X_3, Y_2Y_3, T)$ of the *Assumption 2* from the challenger $\mathcal{C}$ and depending on the distribution of T, $\mathcal{S}$ simulates either $Game_{k-1,2}$ or $Game_{k,1}$.

Setup. $\mathcal{S}$ constructs the public parameters $\mathtt{Pp}$ in the following way. $\mathcal{S}$ chooses random $a, \alpha \in \mathbb{Z}_N$ and $\forall i \in U, t_i \in \mathbb{Z}_N$ and sets $T_i = g^{t_i}$ and starts interaction with the adversary $\mathcal{A}$ by giving

$$\texttt{Pp} = [N, g, g^a, Y = e(g,g)^\alpha, (T_i)_{i \in U}]$$

and $\mathcal{S}$ keeps the master key $\texttt{Msk} = [\alpha, X_3]$

For both the games, the challenge ciphertext is semi-functional, the first $(k-1)$ keys are semi-functional of type 2 and the last $(q-k)$ keys are normal. For the game $Game_{k,1}$, the k^{th} key is semi-functional of type 1 and for $Game_{k-1,2}$, it is normal.

Key Query Answering. $\mathcal{S}$ handles the first $k-1$ key queries by the following way: Let A be a query set of attributes. $\mathcal{S}$ chooses random $h, r \in \mathbb{Z}_N, R_0, R', R_i \in \mathbb{G}_{p_3}$ for each $i \in A$ and $\mathcal{S}$ answers the key of type 2

$$Sk_A = [K_0 = g^{\alpha + ar}(Y_2 Y_3)^h R_0, K' = g^r R', K_i = T_i^r R_i, \forall i \in A]$$

It is easy to check that Sk_A is properly distributed semi-functional key of type 2.

To answer the k^{th} query, the simulator $\mathcal{S}$ uses T and answers the k^{th} key as

$$Sk_A = [K_0 = g^\alpha T^a R_0, K' = TR', K_i = T^{t_i} R_i, \forall i \in A]$$

Now suppose that $T \in \mathbb{G}$. Let the $\mathbb{G}_{p_1 p_2}$ part of T be $g^r {g_2}^t$ for some $r, t \in \mathbb{Z}_N$. Here we implicitly set $z_i = t.t_i$ and $d = ta$. So the key Sk_A is a semi-functional key of type 1. Note that $a \; modulo \; p_1$ and $t_i \; modulo \; p_1$ are uncorrelated respectively from $a \; modulo \; p_2$ and $t_i \; modulo \; p_2$ by Chinese Remainder Theorem. Similarly if $T \in \mathbb{G}_{p_1 p_3}$, the key is normal.

The simulator $\mathcal{S}$ handles the last $q-k$ key queries by running KeyGen algorithm, since he has the master key $\texttt{Msk}$.

Challenge Phase. $\mathcal{S}$ receives two challenge messages M_0, M_1 and a challenge basis $\mathcal{B}^*$ from the adversary $\mathcal{A}$. $\mathcal{S}$ chooses random $M_b \in \{M_0, M_1\}$. Let $\mathcal{B}^* = \{A_0, A_1, ..., A_m\}$, where each $A_i \subset U$. $\mathcal{S}$ chooses random $s_i \in \mathbb{Z}_N$ and gives the challenge ciphertext $Ct_{\mathcal{B}^*}$ to $\mathcal{A}$

$$Ct_{\mathcal{B}^*} = \big[C_0 = M_b.e(g^\alpha, X_1 X_2), C_1 = X_1 X_2,$$

$$\big(C_{1,i} = g^{s_i}\big)_{i \in [m]}, \big(C_{2,i} = (X_1 X_2)^a \big(\prod_{j \in A_i} T_j \big)^{s_i} \big)_{i \in [m]}\big]$$

The semi-functional key in k^{th} query and the semi-functional ciphertext are properly distributed except that the exponent $c' = ac \; modulo \; p_2$ of g_2 in $C_{2,i}$ part of the ciphertext is correlated with $a \; modulo \; p_2$ in the K_0 part of the key. So if a legitimate semi-functional key of type 1 decrypts the semi-functional ciphertext, it will provide a valid message M because $tc' - cd = tac - cat = 0 \; modulo \; p_2$. But the adversary is not provided any keys whose labeled sets of attributes satisfy the target policy.

Therefore if $T \in \mathbb{G}$, $\mathcal{S}$ simulates $Game_{k,1}$ for $\mathcal{A}$ and if $T \in \mathbb{G}_{p_1 p_3}$, $\mathcal{S}$ simulates $Game_{k-1,2}$ for $\mathcal{A}$. This concludes the proof of the lemma. □

Lemma 3. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{k,1}} - Adv^{\mathcal{A}}_{Game_{k,2}} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 2.*

Proof. We establish a PPT algorithm $\mathcal{S}$ which takes the parameters, $(\mathcal{I}, g, X_1X_2, X_3, Y_2Y_3, T)$ of the *Assumption 2* from the challenger $\mathcal{C}$ and depending on the distribution of T, $\mathcal{S}$ simulates either $Game_{k,1}$ or $Game_{k,2}$.

Setup. $\mathcal{S}$ constructs the public parameters $\mathtt{Pp}$ in the following way. $\mathcal{S}$ chooses random $a, \alpha \in \mathbb{Z}_N$ and $\forall i \in U, t_i \in \mathbb{Z}_N$ and sets $T_i = g^{t_i}$ and starts interaction with the adversary $\mathcal{A}$ by giving

$$\mathtt{Pp} = [N, g, g^a, Y = e(g,g)^{\alpha}, (T_i)_{i \in U}]$$

and $\mathcal{S}$ keeps the master key $\mathtt{Msk} = [\alpha, X_3]$
For both the games, the challenge ciphertext is semi-functional, the first $(k-1)$ keys are semi-functional of type 2 and the last $(q-k)$ keys are normal. For the game $Game_{k,1}$, the k^{th} key is semi-functional of type 1 and for $Game_{k,2}$, it is semi-functional of type 2.

Key Query Answering. $\mathcal{S}$ handles the first $k-1$ key queries by the same way as in the lemma 2. Now to answer the k^{th} query, the simulator $\mathcal{S}$ uses T and answers the k^{th} key as

$$Sk_A = [K_0 = g^{\alpha} T^a R_0 (Y_2Y_3)^h, K' = TR', K_i = T^{t_i} R_i, \forall i \in A]$$

Note that the $\mathbb{G}_{p_2}$ part of the K_0 part of the key is randomized by $(Y_2Y_3)^h$. It is easy to check that if $T \in \mathbb{G}$, the k^{th} is of semi-functional of type 1 and similarly if $T \in \mathbb{G}_{p_1p_3}$, the key is of type 2. The simulator $\mathcal{S}$ handles the last $q-k$ key queries by running KeyGen algorithm, since he has the master key $\mathtt{Msk}$.

The challenge semi-functional ciphertext is generated exactly the same way as in the lemma 2. Since the $\mathbb{G}_{p_2}$ part of the key is perfectly random, no legitimate semi-functional key decrypts the semi-functional ciphertext. So if $T \in \mathbb{G}$, the k^{th} key is properly distributed semi-functional key of type 1 and therefore it simulates $Game_{K,1}$ for $\mathcal{A}$ and if $T \in \mathbb{G}_{p_1p_3}$, the key is properly distributed of type 2 and therefore it simulates $Game_{K,2}$ for $\mathcal{A}$. This concludes the proof of the lemma. □

Lemma 4. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{q,2}} - Adv^{\mathcal{A}}_{Game_{Final}} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 3.*

Proof. We establish a PPT algorithm $\mathcal{S}$ which takes the parameters, $(\mathcal{I}, g, X_3, g^{\alpha} X_2, g^s Y_2, Z_2, T)$ of the *Assumption 3* from the challenger $\mathcal{C}$ and depending on the distribution of T, $\mathcal{S}$ simulates either $Game_{q,2}$ or $Game_{Final}$.

Setup. $\mathcal{S}$ constructs the public parameters $\mathtt{Pp}$ in the following way. $\mathcal{S}$ chooses random $a \in \mathbb{Z}_N$ and $\forall i \in U, t_i \in \mathbb{Z}_N, R_i \in \mathbb{G}_{p_3}$ and sets $T_i = g^{t_i}$ and starts interaction with the adversary $\mathcal{A}$ by giving

$$\mathtt{Pp} = [N, g, g^a, Y = e(g, g^{\alpha} X_2) = e(g,g)^{\alpha}, (T_i)_{i \in U}]$$

For both the games, the keys are semi-functional and in $Game_{q,2}$, the ciphertext is semi-functional and in the game $Game_{Final}$, it is almost the semi-functional except C_0, where the challenge message is masked with a random element from $\mathbb{G}_T$

Key Query Answering. $\mathcal{S}$ handles key queries by the following way: Let A be a query set of attributes. $\mathcal{S}$ chooses random $g_2 \in \mathbb{G}_{p_2}, r \in \mathbb{Z}_N$ and $R_0, R', R_i \in \mathbb{G}_{p_3}$ for each $i \in A$ and answers the key

$$Sk_A = [K_0 = g^{\alpha} X_2 (g^a)^r Z_2^r R_0, K' = g^r R', K_i = T_i^r R_i, \forall i \in A]$$

This shows that the key Sk_A is properly distributed.

Challenge Phase. $\mathcal{S}$ receives two challenge messages M_0, M_1 and a challenge basis $\mathcal{B}^*$ from $\mathcal{A}$. $\mathcal{S}$ chooses random $M_b \in \{M_0, M_1\}$. Let $\mathcal{B}^* = \{A_0, A_1, ..., A_m\}$, where each $A_i \subset U$. $\mathcal{S}$ chooses random $s_i \in \mathbb{Z}_N$ and gives the challenge ciphertext $Ct_{\mathcal{B}^*}$ to $\mathcal{A}$

$$Ct_{\mathcal{B}^*} = \big[C_0 = M_b.T, C_1 = g^s Y_2,$$

$$\big(C_{1,i} = g^{s_i}\big)_{i\in[m]}, \big(C_{2,i} = (g^s Y_2)^a \big(\prod_{j \in A_i} T_j \big)^{s_i}\big)_{i\in[m]}\big]$$

If $T = e(g,g)^{\alpha s}$, $Ct_{\mathcal{B}^*}$ is exactly semi-functional ciphertext and therefore $\mathcal{S}$ simulates the game $Game_{q,2}$ for the adversary $\mathcal{A}$. Similarly if T is a random element from $\mathbb{G}_T$, then $\mathcal{S}$ simulates the game $Game_{Final}$. This concludes the proof of the lemma. □

Theorem 2. *If Assumptions 1,2 and 3 hold, then our CP-ABE scheme for MAS is fully secure.*

Proof. If Assumption 1,2 and 3 hold, then by the lemma 1, lemma 2, lemma 3, lemma 4 and hybrid arguments over the sequence of games, we have that the real security game $Game_{Real}$ is indistinguishable from the game $Game_{Final}$. Since in game $Game_{Final}$, the challenge message M_b is masked with a random element from $\mathbb{G}_T$, the value of b is theoretically hidden from the view of adversary $\mathcal{A}$. Hence by the security definition 5, the adversary $\mathcal{A}$ has at most negligible advantage in breaking our CP-ABE scheme. □

2.7 KP-ABE Scheme with Short Key for MAS

Now we present our efficient KP-ABE scheme using the same technique as in the CP-ABE scheme and which is also fully secure in the standard model over composite order bilinear groups. In our KP-ABE construction, all the algorithms are almost the same as that of CP-ABE construction but we interchange the role of the access policy and the set of attributes i.e, the secret keys are labeled with monotone access structures and ciphertext is associated with a set of attributes. Like the CP-ABE scheme, we represent the monotone access structure by the set of minimal sets to reduce the key size. In this construction, we refer to the setup algorithm in section 2.5.

KeyGen(Pp,Msk,$\mathcal{B}$). Here $\mathcal{B}$ is the set of minimal sets for a monotone access structure Γ. Let $\mathcal{B} = \{A_1, A_2, ..., A_m\}$, where $A_i \subset \mathcal{U}\ \forall i \in [m]$ and m is the size of $\mathcal{B}$. The KeyGen algorithm chooses random $r, \hat{r}_1, \hat{r}_2 \in \mathbb{Z}_N$ such that $r = \hat{r}_1 + \hat{r}_2$, $R_0, R_1, R_{1,i}, R_{2,i} \in \mathbb{G}_{p_3}$ and $r_i \in \mathbb{Z}_N$ for each $i \in [m]$. The algorithm returns the secret key

$$Sk_{\mathcal{B}} = [K_0 = g^{\alpha + ar} R_0, K_1 = g^{\hat{r}_1} R_1, (K_{1,i} = g^{r_i} R_{1,i})_{i \in [m]},$$

$$(K_{2,i} = g^{a\hat{r}_2} \big(\prod_{j \in A_i} T_j \big)^{r_i} R_{2,i})_{i \in [m]}]$$

Encrypt(Pp,A, M). It chooses a random $s \in \mathbb{Z}_N$. It outputs the ciphertext corresponding to the set of attributes A

$$Ct_A = \big[A, C_0 = MY^s, C_1 = g^s, C_2 = g^{as}, C_{3,i} = T_i^s, \forall i \in A\big]$$

Decrypt($Ct_A, Sk_{\mathcal{B}}$). Suppose A satisfies the monotone access structure Γ, generated by $\mathcal{B}$, then it must be a superset of a minimal set in $\mathcal{B}$. Let $A_j \subset A$ for some $j \in [m]$. Then it computes

$$e(K_1, C_2) e(\frac{K_{2,j}}{K_0}, C_1) / e(K_{1,j}, \prod_{i \in A_j} C_{3,i}) = e(g, g)^{-\alpha s}$$

The algorithm outputs $C_0 . e(g, g)^{-\alpha s} = M$

Theorem 3. *If Assumptions 1, 2 and 3 hold, then our KP-ABE scheme for MAS is fully secure. (The proof is given in Appendix A)*

3 Extending to Non-Monotone Access Structures

In section 2.5 and 2.7, we constructed our efficient CP-ABE scheme and KP-ABE scheme for monotone access structures using their minimal sets representation. Using similar technique for the entire non-MAS, we construct a HKP-ABE scheme with constant size ciphertext and a CP-ABE scheme with constant size key for non-MAS.

3.1 CP-ABE Scheme with Constant Size Key for Non-MAS

In this construction, a non-monotone access stricture over the universe of attributes is represented by the set of authorized sets in the non-monotone access structure. We refer to the setup algorithm in section 2.5.

KeyGen(Pp,Msk,A). It chooses random $r \in \mathbb{Z}_N, R_0, R', R \in \mathbb{G}_{p_3}$. It outputs the secret key

$$Sk_A = [K_0 = g^{\alpha + ar} R_0, K' = g^r R', K = \big(\prod_{i \in A} T_i \big)^r R]$$

Table 2. Comparison with our KP-ABE for MAS

Schemes	Security	Assumption	Ciphertext Size	Key Size	Decryption Cost	
					Pairing	Exponent
LOST [22]	Adaptive	*Assmp* 1,2,3	$\mathcal{O}(\lvert A\rvert\phi)\lvert\mathbb{G}\rvert$	$\mathcal{O}(l)$	$\mathcal{O}(I)$	$\mathcal{O}(I)$
OT [24]	Adaptive	DLIN	$\mathcal{O}(\lvert A\rvert\phi)\lvert\mathbb{G}\rvert$	$\mathcal{O}(l\phi)$	$\mathcal{O}(I\phi)$	$\mathcal{O}(I)$
ALD [28]	Selective	n-DBDHE	$3\lvert\mathbb{G}\rvert$	$\mathcal{O}(ln)$	3	$\mathcal{O}(In)$
Our KP-ABE	Adaptive	*Assmp* 1,2,3	$(2\lvert A\rvert+1)\lvert\mathbb{G}\rvert$	$2\lvert\mathcal{B}\rvert+2$	3	None

Encrypt(Pp,Γ, M). Here Γ is a non-monotone access structure. Let $\Gamma = \{A_1, A_2, ..., A_m\}$, where $A_i \subset \mathcal{U}\ \forall i \in [m]$ and m is the size of the non-monotone access structure Γ. The encryption algorithm chooses random $s \in \mathbb{Z}_N$ and $s_i \in \mathbb{Z}_N$ for each $i \in [m]$. The algorithm returns the ciphertext

$$Ct_\Gamma = \big[\Gamma, C_0 = MY^s, C_1 = g^s, (C_{1,i} = g^{s_i})_{i\in[m]}, \big(C_{2,i} = g^{as}\big(\prod_{j\in A_i} T_j\big)^{s_i}\big)_{i\in[m]}\big]$$

Decrypt(Ct_Γ, Sk_A). Suppose A satisfies the non-monotone access structure Γ, then $A \in \Gamma$ i,e. $A = A_j$ for some $A_j \in \Gamma$. Then it computes

$$e(C_{2,j}, K')/\big(e(C_1, K_0).e(C_{1,j}, K\big) = e(g,g)^{-\alpha s}$$

The algorithm outputs $C_0.e(g,g)^{-\alpha s} = M$

Theorem 4. *If Assumptions 1,2 and 3 hold, then our CP-ABE scheme for non-MAS is fully secure.*

Proof. The security proof of CP-ABE scheme for non-MAS can be derived in a straightforward way from the security proof of CP-ABE scheme for MAS with minor modification. The minor modification is that in the simulation of CP-ABE for MAS, the key components K_i for each attribute $i \in A$ are just multiplied to get a single key component $K = \prod_{i\in A} K_i$. □

3.2 Hierarchical KP-ABE Scheme with Constant Size Ciphertext for Non-MAS

Hierarchical KP-ABE schemes for general monotone access structures have been proposed in [9, 29], where the monotone access structures were presented by access trees or monotone span programs. In this paper, we first give the delegation for any access structures (not necessarily monotone). Let Γ and $\hat{\Gamma}$ be two access structures over the universe of attributes $\mathcal{U}$. We say that $\hat{\Gamma}$ is a delegation of Γ, in notation $\hat{\Gamma} \prec \Gamma$ if $\hat{\Gamma} \subset \Gamma$. A HKP-ABE scheme consists of five efficient algorithms (Setup, Encrypt, KeyGen, Decrypt, Delegate). The semantics of Setup, Encrypt, KeyGen and Decrypt are identical to those given for KP-ABE and the delegation algorithm has the following semantics.

Table 3. Comparison with our CP-ABE for Non-MAS

Schemes	Security	Assumption	Ciphertext Size	Key Size	Decryption Cost	
					Pairing	Exponent
OT[24]	Adaptive	DLIN	$\mathcal{O}(l\phi)\|\mathbb{G}\|$	$\mathcal{O}(\|A\|\phi)$	$\mathcal{O}(I\phi)$	$\mathcal{O}(I)$
Our CP-ABE	Adaptive	*Assmp* 1,2,3	$(2\|\Gamma\|+1)\|\mathbb{G}\|$	3	3	None

Delegate($\mathtt{Pp}$,Sk_Γ,Γ,$\hat{\Gamma}$). It takes as input public parameters $\mathtt{Pp}$, access structures Γ and $\hat{\Gamma}$ over the universe of attributes $\mathcal{U}$ with $\hat{\Gamma} \prec \Gamma$, a secret key Sk_Γ for Γ and outputs a secret key $Sk_{\hat{\Gamma}}$ for $\hat{\Gamma}$.

3.3 Full Security Definition of HKP-ABE

Our security definition follows the security definition of [17] and our security experiment is denoted by $Game_{Delegate}$. The advantage of a PPT adversary $\mathcal{A}$ in the security game $Game_{Delegate}$ is the absolute difference of the winning probability against the challenger $\mathcal{C}$ and $1/2$. The game $Game_{Delegate}$ consists of Setup Phase, Key Query Phase and Challenge Phase. Precisely, all these phases are given below.

- **Setup** $\mathcal{C}$ runs the Setup algorithm on input a security parameter 1^λ and a universe of attributes $\mathcal{U}$ to generate public parameters $\mathtt{Pp}$ and master secret key $\mathtt{Msk}$. The challenger $\mathcal{C}$ starts the interaction with the adversary $\mathcal{A}$ by giving the public parameter $\mathtt{Pp}$.
- **Phase 1.** Key query consists of three different phases (algorithms), Create, Delegate and Reveal. $\mathcal{C}$ answers these queries in the following way. $\mathcal{C}$ chooses a set $\mathcal{R}$ of private keys and initializes it to $\emptyset$.
  - **Create.** $\mathcal{A}$ makes a create query by specifying the access structure Γ over the universe of attributes $\mathcal{U}$. In response, $\mathcal{C}$ creates a key for Γ by running the KeyGen algorithm on input $\mathtt{Msk}$ and Γ. The challenger $\mathcal{C}$ adds the key Sk_Γ to the set $\mathcal{R}$ and gives to $\mathcal{A}$ a reference to it, not the actual key Sk_Γ.
  - **Delegate.** $\mathcal{A}$ specifies a reference to a key Sk_Γ in the set $\mathcal{R}$ and an access structure $\hat{\Gamma}$ such that $\hat{\Gamma} \prec \Gamma$. In response, $\mathcal{C}$ makes a key for $\hat{\Gamma}$ by executing the Delegate algorithm on input $\mathtt{Pp}$, Sk_Γ, Γ and $\hat{\Gamma}$. $\mathcal{C}$ adds this key $sk_{\hat{\Gamma}}$ to the set $\mathcal{R}$ and gives to $\mathcal{A}$ a reference to it, not the actual key Sk_Γ.
  - **Reveal.** To make a reveal query, $\mathcal{A}$ specifies a reference to a key Sk_Γ in the set $\mathcal{S}$ and the challenger $\mathcal{C}$ gives to the adversary $\mathcal{A}$ the corresponding secret key Sk_Γ and removes it from $\mathcal{R}$.
- **Challenge Phase.** The adversary gives two equal length messages M_0 and M_1 and a set of attributes A^* such that it does not satisfy the revealed access structures. The challenger randomly chooses a bit b from $\{0,1\}$ and encrypts the message M_b using the set of attributes A^* and gives the challenge ciphertext Ct_{A^*} to the adversary $\mathcal{A}$

- **Phase 2.** Repeat phase 1 with the restriction that no revealed access structure Γ is satisfied by the set of attributes A^*.
- **Guess.** The adversary $\mathcal{A}$ outputs the guess $\hat{b}$ for b.

The advantage $Adv^{\mathcal{A}}_{Game_{Delegate}}(\lambda)$ of the adversary $\mathcal{A}$ in $Game_{Delegate}$ is defined by $|Pr(\hat{b} = b) - 1/2|$

Definition 9. *A HKP-ABE scheme is said to be fully secure if for all PPT adversary $\mathcal{A}$, the advantage $Adv^{\mathcal{A}}_{Game_{Delegate}}(\lambda)$ is a negligible function of λ in $Game_{Delegate}$.*

3.4 HKP-ABE Construction

Like CP-ABE scheme for non-monotone access structures, we represent the non-MAS by the set of authorized sets. We refer to the setup algorithm in section 2.5.

KeyGen(Pp,Msk,Γ). Here Γ is a non-monotone access structure. Let $\Gamma = \{A_1, A_2, ..., A_m\}$, where $A_i \subset \mathcal{U}$ $\forall i \in [m]$ and m is the number of authorized sets in Γ. The KeyGen algorithm chooses random $r, \hat{r}_1, \hat{r}_2 \in \mathbb{Z}_N$ such that $r = \hat{r}_1 + \hat{r}_2$, $R_0, R_1, R_{1,i}, R_{2,i} \in \mathbb{G}_{p_3}$ and $r_i \in \mathbb{Z}_N$ for each $i \in [m]$. The algorithm returns the secret key

$$Sk_\Gamma = [K_0 = g^{\alpha + ar} R_0, K_1 = g^{\hat{r}_1} R_1, (K_{1,i} = g^{r_i} R_{1,i})_{i \in [m]},$$

$$(K_{2,i} = g^{a\hat{r}_2} \big(\prod_{j \in A_i} T_j \big)^{r_i} R_{2,i})_{i \in [m]}]$$

Encrypt(Pp,A, M). It chooses a random $s \in \mathbb{Z}_N$. It outputs the ciphertext corresponding to the set of attributes A

$$Ct_A = \big[A, C_0 = MY^s, C_1 = g^s, C_2 = g^{as}, C = \big(\prod_{i \in A} T_i \big)^s \big]$$

Decrypt(Ct_A, Sk_Γ). Suppose A satisfies the non-monotone access structure Γ, then $A \in \Gamma$ i,e. $A = A_j$ for some $A_j \in \Gamma$. Then it computes

$$e(K_1, C_2) e(\frac{K_{2,j}}{K_0}, C_1) / e(K_{1,j}, C) = e(g, g)^{-\alpha s}$$

The algorithm outputs $C_0 . e(g, g)^{-\alpha s} = M$

Delegate(Pp,Sk_Γ,Γ,$\hat{\Gamma}$). Here $\hat{\Gamma} \prec \Gamma$. Let $|\Gamma| = l$ and $|\hat{\Gamma}| = \hat{l}$ and $Sk_\Gamma = [K_0, K_1, (K_{1,i})_{i \in [l]}, (K_{2,i})_{i \in [l]}]$. The delegate algorithm chooses random $\bar{r}, \bar{r}_1, \bar{r}_2 \in \mathbb{Z}_N$ such that $\bar{r}_1 + \bar{r}_2 = \bar{r}$ and $r'_i \in \mathbb{Z}_N$ for each $i \in [\hat{l}]$. The algorithm returns the delegate key

$$Sk_{\hat{\Gamma}} = [\hat{K}_0 = K_0 . g^{a\bar{r}}, \hat{K}_1 = K_1 . g^{\bar{r}_1}, (\hat{K}_{1,i} = K_{1,i} . g^{r'_i})_{i \in [\hat{l}]},$$

$$(\hat{K}_{2,i} = K_{2,i} . g^{a\bar{r}_2} \big(\prod_{j \in A_i} T_j \big)^{r'_i})_{i \in [\hat{l}]}]$$

Table 4. Comparison with our HKP-ABE for Non-MAS

Schemes	Security	Assumption	Ciphertext Size	Key Size	Decryption Cost	
					Pairing	Exponent
OT [24]	Adaptive	DLIN	$\mathcal{O}(\lvert A\rvert\phi)\lvert\mathbb{G}\rvert$	$\mathcal{O}(l\phi)$	$\mathcal{O}(I\phi)$	$\mathcal{O}(I)$
ALD [28]	Selective	n-DBDHE	$3\lvert\mathbb{G}\rvert$	$\mathcal{O}(ln)$	3	$\mathcal{O}(In)$
Our HKP-ABE	Adaptive	*Assmp* 1,2,3	$3\lvert\mathbb{G}\rvert$	$2\lvert\Gamma\rvert+2$	3	None

Theorem 5. *If Assumptions 1, 2 and 3 hold, then our HKP-ABE scheme for non-MAS is fully secure. (The proof is given in Appendix B)*

4 Conclusions

In this paper, we have presented a fully secure CP-ABE (resp. KP-ABE) scheme with short ciphertext (resp. key) for monotone access structures using simple "encoding technique". By using a similar idea, we proposed a CP-ABE scheme with constant size key and a HKP-ABE scheme with constant size ciphertext for non-monotone access structures which are fully secure in the standard model over composite order bilinear groups. In all the schemes the number of pairing and exponent computations in the decryption algorithm is constant.

References

1. 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)
2. Benaloh, J., Leichter, J.: Generalized Secret Sharing and Monotone Functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, Heidelberg (1990)
3. Stinson, D.R.: An explication of secret sharing schemes. Designs, Codes and Cryptography 2, 357–390 (1992)
4. Beimel, A.: Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel (1996)
5. Boneh, D., Franklin, M.K.: Identity based encryption from the Weil pairing. SIAM Journal on Computing 32(3), 586–615 (2003)
6. Nikov, V., Nikova, S., Preneel, B.: On the Size of Monotone Span Programs. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 249–262. Springer, Heidelberg (2005)
7. 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)
8. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
9. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute Based Encryption for Fine-Grained Access Conrol of Encrypted Data. In: ACM Conference on Computer and Communications Security, ACM CCS (2006)
10. Chase, M.: Multi-authority Attribute Based Encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)

11. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334 (2007)
12. Cheung, L., Newport, C.: Provably Secure Ciphertext Policy ABE. In: ACM Conference on Computer and Communications Security, ACM CCS (2007)
13. Ostrovsky, R., Sahai, A., Waters, B.: Attribute Based Encryption with Non-Monotonic Access Structures. In: ACM Conference on Computer and Communications Security, ACM CCS (2007)
14. 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)
15. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. Cryptology ePrint Archive, Report 2008/290 (2008)
16. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
17. 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)
18. Maji, H., Prabhakaran, M., Rosulek, M.: Attribute-based signatures: Achieving attribute privacy and collusion-resistance. ePrint, IACR, http://eprint.iacr.org/2008/328
19. 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)
20. 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)
21. Herranz, J., Laguillaumie, F., Ràfols, C.: Constant Size Ciphertexts in Threshold Attribute-Based Encryption. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 19–34. Springer, Heidelberg (2010)
22. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
23. 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)
24. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
25. Lewko, A., Waters, B.: Decentralizing attribute-based encryption. Cryptology ePrint Archive, Report 2010/351 (2010), http://eprint.iacr.org/
26. Waters, B.: Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011)
27. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-Based Signatures. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 376–392. Springer, Heidelberg (2011) Full version is available at, http://eprint.iacr.org/2010/595

28. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 90–108. Springer, Heidelberg (2011)
29. Lewko, A., Waters, B.: Unbounded HIBE and Attribute-Based Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 547–567. Springer, Heidelberg (2011)

A The Security Proof of KP-ABE Scheme for MAS

The proof technique for KP-ABE scheme is almost the same as that of CP-ABE and for which we need the description of semi-functional keys and ciphertext and are given below:

Semi-functional Key. Like CP-ABE, we also consider two forms of semi-functional key. Let $\mathcal{B}$ be the basis of a MAS with $|\mathcal{B}| = m$. Choose random $d, d_1, d_2, r, \hat{r}_1, \hat{r}_2 \in \mathbb{Z}_N$ such that $r = \hat{r}_1 + \hat{r}_2, g_2 \in \mathbb{G}_{p_2}, R_0, R_1, R_{1,i}, R_{2,i} \in \mathbb{G}_{p_3}$ and $r_i \in \mathbb{Z}_N$ for each $i \in [m]$. The distribution of type 1 key is the following.

$$Sk_{\mathcal{B}} = [K_0 = g^{\alpha + ar} R_0 {g_2}^d, K_1 = g^{\hat{r}_1} R_1 {g_2}^{d_1}, (K_{1,i} = g^{r_i} R_{1,i})_{i \in [m]},$$

$$(K_{2,i} = g^{a\hat{r}_2} \big(\prod_{j \in A_i} T_j \big)^{r_i} R_{2,i} {g_2}^{d_2})_{i \in [m]}]$$

The distribution of type 2 key is same as type 1 except the key components $K_1, K_{1,i}$ and $K_{2,i}$ do not contain any $\mathbb{G}_{p_2}$ part.

Semi-functional Ciphertext. Let A be a set of attributes. Then choose random $s, c, c' \in \mathbb{Z}_N, g_2 \in \mathbb{G}_{p_2}$ and $c_i \in \mathbb{Z}_N$ for each $i \in A$. The semi-functional ciphertext is

$$Ct_A = \big[A, C_0 = MY^s, C_1 = g^s g_2^c, C_2 = g^{as} g_2^{c'}, C_{3,i} = {T_i}^s {g_2}^{c_i} \forall i \in A\big]$$

Note that if a legitimate semi-functional key decrypts a semi-functional ciphertext, we will get an extra term $e(g_2, g_2)^{cd_2 + c'd_1 - cd}$

The number of games between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$ in the security proof of KP-ABE is $2q + 3$ as in case of CP-ABE. The description of games and the proof techniques are almost the same as that of CP-ABE and that is why we only state the lemmas and theorem and skip the proofs.

Lemma 5. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{Real}} - Adv^{\mathcal{A}}_{Game_0} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 1.*

Lemma 6. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{k-1,2}} - Adv^{\mathcal{A}}_{Game_{k,1}} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 2.*

Lemma 7. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{k,1}} - Adv^{\mathcal{A}}_{Game_{k,2}} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 2.*

Lemma 8. *Suppose there exists a PPT algorithm $\mathcal{A}$ with $Adv^{\mathcal{A}}_{Game_{q,2}} - Adv^{\mathcal{A}}_{Game_{Final}} = \epsilon$. Then there exist a PPT algorithm $\mathcal{S}$ with advantage ϵ in breaking Assumption 3.*

Theorem 6. *If Assumptions 1, 2 and 3 hold, then our KP-ABE scheme is fully secure.*

Proof. The proof follows from the lemma 5, lemma 6, lemma 7 and lemma 8. □

B The Security Proof of Our HKP-ABE for Non-MAS

The security proof of HKP-ABE for non-MAS will be followed very similar way from that of KP-ABE in Appendix A. The proof of HKP-ABE involves all the games in KP-ABE, plus one extra game $Game_{Delegate}$ which is the experiment in the security definition of HKP-ABE defined in section 3.3. In $Game_{Real}$, all the queries of the adversary are answered by running KeyGen algorithm but in $Game_{Delegate}$, the key queries and the delegate queries are answered by executing the KeyGen and the Delegate algorithm respectively. Since in Appendix A, we show that under Assumption 1, 2 and 3, the game $Game_{Real}$ and $Game_{Final}$ are indistinguishable and all the adversaries $\mathcal{A}$ has at most negligible advantage in $Game_{Final}$, so to complete the security proof of our HKP-ABE, we need only to prove the following lemma.

Lemma 9. *$Game_{Real}$ and $Game_{Delegate}$ are indistinguishable.*

Proof. The only difference between $Game_{Real}$ and $Game_{Delegate}$ is the delegate key query answering. In case of delegate key query, the challenger $\mathcal{C}$ answers to the adversary $\mathcal{A}$ in $Game_{Delegate}$ by running the Delegate algorithm which takes $\mathtt{Pp}, Sk_\Gamma, \Gamma, \hat{\Gamma}$ as input, where $\hat{\Gamma} \prec \Gamma$ and in $Game_{Real}$, $\mathcal{C}$ answers to $\mathcal{A}$ by using the keyGen algorithm which takes $\mathtt{Msk}, \Gamma$ as input. Since the Delegate algorithm re-randomizes the key components, so in both the case the secret key $Sk_{\hat{\Gamma}}$ corresponding to $\hat{\Gamma}$ have the same distribution. This concludes the proof of the lemma. □

C Discussion

Our ABE schemes can be treated as small universe constructions, where the size of public parameters is linear in the size of the universe of attributes. It is also possible to give a large universe construction using our techniques, where all the elements of $\mathbb{Z}^*_{p_1}$ can be used as attributes but the size of public parameters is linear in n, a parameter which denotes the maximum number of attributes used in encryption. By applying a collision-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}^*_{p_1}$, we can allow to use arbitrary string as attributes. Instead of using T_i associated with attribute i, we use a function $T : \mathbb{Z}_{p_1} \to \mathbb{G}_{p_1}$ based on a n degree polynomial as in [9] and [8].

Symmetric Inner-Product Predicate Encryption Based on Three Groups

Masayuki Yoshino[1,2], Noboru Kunihiro[2], Ken Naganuma[1], and Hisayoshi Sato[1]

[1] Hitachi, Ltd.
{masayuki.yoshino.aa,ken.naganuma.dn,hisayoshi.sato.th}@hitachi.com
[2] The University of Tokyo
masayuki.yoshino@it.k.u-tokyo.ac.jp, kunihiro@k.u-tokyo.ac.jp

Abstract. This paper presents the first symmetric-key inner-product predicate encryption scheme based on *three* groups. The performance of predicate encryption schemes based on hidden subgroup problems depends on the number of hidden subgroups and this number should be optimized. The scheme presented here satisfies the selective security model under a non-interactive assumption where the number of terms does not depend on the number of adversarial queries. It is therefore as secure as the symmetric predicate scheme proposed by Shen et al., which is based on *four* groups, under a simpler assumption. Using three hidden groups instead of four, it has a message space more than 33% wider and is more resistant to integer factoring attacks with moderate security parameters. The available techniques for converting encryption schemes using composite-order bilinear groups into schemes using prime-order groups are applicable to our scheme. Compared with the previous scheme using the conversion techniques, our prime-order group instantiation is asymptotically more than 33% faster and has ciphertexts and tokens that are asymptotically 25% smaller.

Keywords: symmetric predicate encryption, subgroup decision problem, public cloud infrastructure.

1 Introduction

Progress in networking technology and an increase in the demand for computing resources have prompted many organizations to outsource their computer environments. This has resulted in a new computing model, often called cloud computing, that can be roughly categorized as private or public. In a private cloud the infrastructure is owned and managed by the user and is located on-premise: access to the user's data is under the user's control. In a public cloud the infrastructure is owned and managed by a service provider and is located off-premise: access to a user's data is not under the user's control and can potentially be granted to untrusted parties.

Moving user data to a public cloud provider such as Google, Microsoft, or Amazon enables the user to avoid the costs of building and maintaining cloud infrastructure and instead simply pay for using the services offered by the provider.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 215–234, 2012.

The users of public clouds can access their data anytime from anywhere and do not need to worry about data backups, but this convenience is associated with significant security and privacy risks.

Traditional encryption schemes provide access control in all-or-nothing manner: data encrypted using a key can be decrypted only by the key owner. This rigid security does not meet the needs of public cloud users. In a line of research beginning with the work of Sahai and Waters [34], a number of researches have asked how to implement more complex access policies in ciphertexts or token queries. The result is the notion of functional encryption [9], which is a new way of giving functionality to encrypted data while keeping the data confidential. One of the most important functional encryption schemes is *predicate encryption* [23, 36]. Informally, a predicate encryption scheme lets the owner of a master secret key issue tokens that allows users to learn whether ciphertexts are associated with the token but keeps the token content (called the predicate) hidden. in most predicate encryptions, the technical barriers to data access are broken by using pairing operations on bilinear groups [2, 23, 25–28, 36, 38].

Predicate encryption schemes may encourage to the implementation of secure cloud computing in public cloud infrastructures. A public-key setting is appropriate for multiple-user applications such as e-mail services [6] and broadcast services [10], but its security is weaker than that of a symmetric-key setting. As pointed out in [33, 36], an adversary can use public keys to encrypt any plaintext and evaluate a token query to learn whether the ciphertext is associated with that token. Thus he can acquire information about the content of a token. A symmetric-key setting, on the other hand, is appropriate for single-user applications and is more secure. Some secure symmetric-key schemes have been proposed for simple applications such as remote storage services [16] and private database services [20, 32] as shown in Figure 1.

To make more expressive token queries such as equality checks of conjunction, disjunction and their combinations, Shen, Shi, and Waters proposed a symmetric-key encryption scheme for inner-product predicates [36]. Their scheme, here called the SSW scheme, uses bilinear groups, which are significant obstacles to practical implementation. Its security is based on a variant of the *subgroup decision assumption*. This assumption implies that it is infeasible to factor a composite order of the bilinear group. Such large composite-order groups, however, result in a heavy load of group operations, markedly reducing the efficiency of the SSW scheme. Freeman [18] and Lewko [24] therefore proposed techniques converting composite-order groups with hidden subgroups into prime-order groups, which do not require the assumption of integer factorization infeasibility.

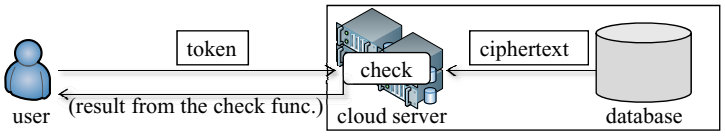


Fig. 1. Single-user application of symmetric predicate encryption schemes

Despite these conversion techniques, symmetric predicate encryption schemes still face practical problems, one of which is that the converted SSW scheme using d-bit prime groups generates $8d(n+2)$-bit ciphertext, where n is the dimensionality of the vector space representing plaintexts and predicates. Unfortunately, this ciphertext is much larger than the ciphertext generated by simple functional encryption schemes such as [17, 37]. A second problem, long running time, may be more serious. After receiving a token, a computer calls d-bit pairing operations $(4n+2)$ times to verify a relation of a ciphertext and the token. The heavy computational load penalizes the performance of most applications of symmetric predicate encryptions. The running time for the check procedure is predominantly due to the heavy load of pairing operations, so fewer calls to paring operation would be preferred.

1.1 Our Motivation and Contribution

Cryptography may contribute to the security of cloud computing in the public environment, but, there is a well-known trade off between security and efficiency. Researchers have proposed several predicate encryption schemes based on security models with a strict restrictions on queries: adversaries cannot freely query ciphertexts or tokens if there is an obvious relation with a challenge [8, 16, 11, 25]. In applications actually using cryptosystems, however, there is no such restriction [32]. Server administrators (or intruders) thus can break the indistinguishability by learning a relation between ciphertext and token queries. Secure token query techniques such as PIR [14, 29] and oblivious RAM [20, 31] can provide perfect security (even administrator cannot learn anything in theory), but their computational cost is impractically high. We therefore think that practical efficiency is one of the most interesting aspects of predicate encryption.

This paper presents the first encryption schemes for inner-product predicates in symmetric-key settings that is based on a bilinear group consisting of *three* hidden subgroups. The performance of predicate encryption schemes based on hidden subgroup problems depends on the number of hidden subgroups, and an efficient scheme can be obtained by optimizing the number of subgroups. This paper presents a symmetric predicate encryption scheme satisfying the selective security model under a non-interactive assumption where the number of terms does not depend on the number of adversarial queries. The means our proposed scheme, which is based on *three* groups, is as secure as the SSW scheme based on four. Having one less group, the proposed scheme has a message space more than 33% wider. Furthermore, the proposed scheme uses a bilinear group whose composite order is more resistant to integer factoring algorithms than that of the SSW scheme. Taking advantage of the techniques introduced by [18, 24] to convert encryption schemes using composite-order bilinear groups into schemes using prime-order groups, our prime-order group instantiation has ciphertexts and tokens that are asymptotically 25% smaller than those of the converted SSW scheme and it performs all processes more than 33% faster than that scheme does.

1.2 Related Work

Predicate Encryption: An important research direction is to construct predicate encryption schemes that are as expressive as possible, with the ultimate goal being to handle all polynomial-time predicates. Predicate encryption for inner products was presented by Katz, Sahai, and Waters [23] as a fine-grained notion of encryption that covers identity-based encryption [3, 4, 7, 15, 19, 22, 35], hidden-vector encryption [2, 11] and attribute-based encryption [1, 21, 25, 30, 34, 38]. The relation of inner-product queries covers a wide class of predicates such as those used in conjunction, subset, and range queries on encrypted data [11] as well as in disjunction and CNF/DNF formulas [23]. The security (attribute-hiding and payload-hiding) model of the predicate encryptions when an adversary requests only token queries and chooses a ciphertext as a challenge has been discussed informally. The public-key predicate encryption scheme was proven to be selectively attribute-hiding [23]. Adaptively attribute-hiding predicate schemes were later studied in [25, 27, 28].

Prime-Order Group Instantiation: Most predicate encryption schemes make use of bilinear groups of composite order. In order to create more efficient versions of the cryptosystems originally requiring composite-order bilinear groups, both Freeman [18] and Lewko [24] gave ways for converting systems requiring composite-order groups to systems using prime-order groups. The performance of the converted versions, such as their running time and their ciphertext and token sizes, depends on the original number of hidden subgroups of the composite-order bilinear groups. Converted cryptosystems requiring fewer subgroups may perform better in terms of ciphertext size and running time.

1.3 Organization

The rest of this paper is organized as follows: the notations and the security model are defined in Section 2. The mathematical background is introduced in Section 3. The proposed scheme is given in Section 4 and its security proof is given in Section 5. A variant of the proposed scheme is considered in Section 6.

2 Definitions

We briefly review definitions of a general framework for tokens on encrypted data [36]. Let Σ be a finite set of binary strings. A predicate f over Σ is a function $f : \Sigma \to \{0, 1\}$. Let $\mathcal{F}$ be a set of predicates over a finite set of binary strings Σ. In this paper we say that $x \in \Sigma$ satisfies the predicate f iff $f(x) = 1$.

2.1 Predicate Encryption in Symmetric-Key Setting

Definition 1. A *s*ymmetric-key predicate encryption (SK-PE) scheme for the class of predicates $\mathcal{F}$ over the set of attributes Σ consists of the following probabilistic polynomial-time algorithms.

- **Setup**(1^λ): Takes a security parameter 1^λ as input and outputs a public parameter and a secret key SK.
- **Encrypt**(SK, x): Takes a secret key SK and a plaintext $x \in \Sigma$ as input and outputs a ciphertext CT.
- **GenToken**(SK, f): Takes a secret key SK and a description of predicate $f \in \mathcal{F}$ as input and outputs a token TK.
- **Check**(CT, TK): Takes a token TK and a ciphertext CT as input and outputs either 0 or 1. The boolean value indicates the value of the predicate f evaluated on the underlying plaintext x.
- **Correctness**. For correctness, this paper requires the following conditions.
  - If $f(x) = 1$, then **Check**$(CT, TK) = 1$.
  - If $f(x) \neq 1$, then **Check**$(CT, TK) = 0$ with provability $1 - \epsilon(\lambda)$ and ϵ is a negligible function.

2.2 Security Model

Selective security in a public-key setting was first described by [13], and selective security in a symmetric-key setting was first described by [36]. These concepts have been used in [3],[5],[10],[12],[37].

This section introduces a selective security model for an SK-PE, which was defined by Shen, Shi, and Waters [36]. They described a query game in which an adversary $\mathcal{A}$ given a set of tokens and ciphertexts tries to get information about any of the predicates or the plaintexts. The game proceeds as follows.

- **Setup:** The challenger $\mathcal{C}$ runs $Setup(1^\lambda)$ and gives the adversary $\mathcal{A}$ a public parameter. $\mathcal{A}$ outputs a bit $d = 0$ indicating a ciphertext challenge and two plaintexts x_0, $x_1 \in \Sigma$, or a bit $d = 1$ indicating a token challenge and two descriptions of predicates f_0, $f_1 \in \mathcal{F}$.
- **Phase 1:** $\mathcal{A}$ adaptively outputs one of the following two queries.
  - On the ith ciphertext query, $\mathcal{A}$ requests a ciphertext and outputs a plaintext $x_i \in \Sigma$ subject to the restriction that, for a token challenge, $f_0(x_i) = f_1(x_i)$. (In a ciphertext challenge, such a restriction for ciphertext queries is not required.) $\mathcal{C}$ responds with the corresponding ciphertext $CT_i \leftarrow Encrypt(SK, x_i)$.
  - On the ith token query, $\mathcal{A}$ requests a token and outputs a predicate $f_i \in \mathcal{F}$ subject to the restriction that, for a ciphertext challenge, $f_i(x_0) = f_i(x_1)$. (In a token challenge, such a restriction for token queries is not required.) $\mathcal{C}$ responds with the corresponding token $TK_i \leftarrow GenToken(SK, f_i)$.
- **Challenge:** $\mathcal{A}$ outputs the bit $d \in \{0, 1\}$ indicating a ciphertext challenge or a token challenge. $\mathcal{C}$ flips a random coin $\mathtt{b} \in \{0, 1\}$ and outputs one of the following two challenges.
  - $\mathcal{C}$ gives $CT_* \leftarrow Encrypt(SK, x_\mathtt{b})$ to $\mathcal{A}$.
  - $\mathcal{C}$ gives $TK_* \leftarrow GenToken(SK, f_\mathtt{b})$ to $\mathcal{A}$.

- **Phase 2:** $\mathcal{A}$ continues to adaptively query ciphertexts CT_i and tokens TK_i, subject to the same restriction as in Phase 1. $\mathcal{C}$ responds with the corresponding ciphertexts $CT_i \leftarrow Encrypt(SK, x_i)$ and the corresponding tokens $TK_i \leftarrow GenToken(SK, f_i)$.
- **Guess:** $\mathcal{A}$ returns a guess $\mathtt{b}' \in \{0, 1\}$ of the random coin $\mathtt{b}$.

One might consider this security model similar to attribute-hiding against chosen plaintext attacks (AH-CPA) [25, 28], but, there are at least two distinctive differences: in this security game $\mathcal{A}$ can issue two kinds of queries (in Phases 1 and 2), while in the AH-CPA game the adversary can issue only token queries. Furthermore, in this security game $\mathcal{A}$ can choose a ciphertext or a token as a challenge, while in the AH-CPA game the adversary can use only a ciphertext as a challenge.

For our proof of security it will be useful to introduce the concepts of *token indistinguishability* (token IND) and *ciphertext indistinguishability* (ciphertext IND), which respectively correspond to the *predicate privacy* and *plaintext privacy* that Shen, Shi, and Waters defined using token challenges and ciphertext challenges in their selective single challenge security game [36]. We renamed their definitions to highlight indistinguishability of token challenges and ciphertext challenges in the selective security game.

Definition 2. *(Token IND)* A symmetric-key predicate encryption scheme has token indistinguishability if for a token challenge the advantage of any polynomial-time adversary $\mathcal{A}$ in winning the selective challenge game is negligible in the security parameter λ.

Definition 3. *(Ciphertext IND)* A symmetric-key predicate encryption scheme has ciphertext indistinguishability if for a ciphertext challenge the advantage of any polynomial-time adversary $\mathcal{A}$ in winning the selective challenge game is negligible in the security parameter λ.

3 Background and Assumption

Bilinear groups of composite order were first described by Boneh, Goh, and Nissim [8]. We briefly review some facts about the bilinear groups whose composite order has only three factors and then state an assumption to prove the security of our scheme. For simplicity, we assume in Sections 3–5 that the pairing $\hat{e}$ is symmetric ($\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$).

3.1 3-Factor-Based Composite-Order Bilinear Groups

Let $\mathcal{G}$ denote a composite-order bilinear group generator algorithm that takes as input a security parameter λ and outputs a tuple $(p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$ where p_1, p_2, p_3 are distinct primes, $\mathbb{G}$ and $\mathbb{G}_T$ are three cyclic groups of order $N = p_1 p_2 p_3$, and the pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ satisfies the following properties.

- Bilinearity: $^{\forall}g, ^{\forall} h \in \mathbb{G}, a, b \in \mathbb{Z}_N, \hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$
- Nondegeneracy: For any $g \in \mathbb{G}$, if $\hat{e}(g, h) = 1$ for all $h \in \mathbb{G}$, then $g = 1$.

We assume that group operations in $\mathbb{G}$ and $\mathbb{G}_T$ as well as the bilinear map $\hat{e}$ can be computed in polynomial time. The notations $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_3$ denote the subgroups of $\mathbb{G}$ having orders p_1, p_2, and p_3. The following facts about composite-order bilinear groups will be used in this paper.

- $\hat{e}(a_1, a_2) = \hat{e}(a_2, a_3) = \hat{e}(a_3, a_1) = 1$ with $a_1 \in \mathbb{G}_1, a_2 \in \mathbb{G}_2, a_3 \in \mathbb{G}_3$.
- $a, b \in \mathbb{G}_N$ are uniquely represented as $a = a_1 a_2 a_3$, $b = b_1 b_2 b_3$ where $a_1, b_1 \in \mathbb{G}_1$, $a_2, b_2 \in \mathbb{G}_2$ and $a_3, b_3 \in \mathbb{G}_3$. Then $\hat{e}(a, b) = \hat{e}(a_1, b_1) \cdot \hat{e}(a_2, b_2) \cdot \hat{e}(a_3, b_3)$.

3.2 Our Assumption

The security of our scheme relies on only a *non-interactive* assumption; that is, the number of parameters given to adversaries is static (constant size) and does not depend on the number of adversarial queries. The assumption is new but we justify it in Appendix A by proving that it holds in the generic model of bilinear groups.

Assumption 1. *Given a bilinear group generator $\mathcal{G}$ such that for $i = 1, 2, 3$ the output groups $\mathbb{G}_i$ are of prime order p_i using the following experiment*

1. $(p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}(1^\lambda)$
2. $N \leftarrow p_1 p_2 p_3$, $g_1 \xleftarrow{R} \mathbb{G}_1$, $g_2 \xleftarrow{R} \mathbb{G}_2$, $g_3 \xleftarrow{R} \mathbb{G}_3$
3. $P \leftarrow (N, \mathbb{G}, \mathbb{G}_T, \hat{e})$
4. $D \leftarrow (g_1, g_1^{a_1} g_2, g_2^{b_1} g_3^{c_1}, g_3^{c_2 d}, g_3^{d}, g_3^{d^2}, g_1^{a_2} g_3^{c_1 d})$
   with $a_1, a_2 \xleftarrow{R} \mathbb{Z}_{p_1}, b_1 \xleftarrow{R} \mathbb{Z}_{p_2}, c_1, c_2, d \xleftarrow{R} \mathbb{Z}_{p_3}$
5. $T_0 \leftarrow g_1^{a_3} g_3^{c_2}$, $T_1 \leftarrow g_1^{a_3} g_2^{b_2} g_3^{c_2}$ with $a_3 \xleftarrow{R} \mathbb{Z}_{p_1}$, $b_2 \xleftarrow{R} \mathbb{Z}_{p_2}$

and given that the advantage of an adversary $\mathcal{A}$ in distinguishing T_0 from T_1 with the parameters (P, D) is defined as

$$Adv_{\mathcal{A}} := |Pr[\mathcal{A}(P, D, T_0)] = 1 - Pr[\mathcal{A}(P, D, T_1)] = 1|$$

for any polynomial-time adversary $\mathcal{A}$ the advantage $Adv_{\mathcal{A}}$ is negligible in the security parameter λ.

Note that in our assumption the naming of subgroups is not significant: the assumption is the same if the subgroups are renamed such as $\mathbb{G}_1 \leftrightarrow \mathbb{G}_3$.

4 Construction

The goal of this section is to construct an SK-PE scheme that supports inner product queries and that uses bilinear groups whose composite order has only *three* factors. In Section 6 we will introduce the corresponding construction using prime-order bilinear groups.

4.1 Intuition for the Construction

In our scheme, each ciphertext is associated with a secret vector $\vec{x}$ representing a plaintext, and each token corresponds to a secret vector $\vec{y}$ describing a predicate. The scheme must check whether $\vec{x} \cdot \vec{y} = 0 \pmod N$ and reveal nothing else about $\vec{x}$ and $\vec{y}$.

We use the observation that, for inner product queries, ciphertexts and tokens play symmetric roles: a token and a ciphertext each encode a vector, and the inner product $\langle x, y \rangle$ is commutative. One way to interpret this observation is to view a ciphertext and a token as symmetric. Our approach is therefore to design the ciphertext and the token in such a way that they are complementarily symmetric so that we can leverage the token IND proven for our main construction to achieve ciphertext IND as well.

4.2 Our Construction of an SK-PE Scheme

At a high level the subgroups of $\mathbb{G}$ will be used as follows: $\mathbb{G}_2$ will be used to encode the vectors $\vec{x}$ and $\vec{y}$ in the ciphertexts and tokens. The inner product $\langle \vec{x}, \vec{y} \rangle$ in $\mathbb{G}_2$ will be computed using the bilinear map. Elements of $\mathbb{G}_3$ will be used in the ciphertexts for masking terms in $\mathbb{G}_2$, and elements of $\mathbb{G}_1$ will be used in the tokens for masking terms in $\mathbb{G}_2$.

Our main construction is an SK-PE where the set of attributes is $\Sigma = (\mathbb{Z}_N)^n$ and the class of predicates is $\mathcal{F} = \{f_{\vec{y}} | \ \vec{y} \in (\mathbb{Z}_N)^n\}$ with $f_{\vec{y}}(\vec{x}) = 1$ iff $\langle \vec{x}, \vec{y} \rangle = 0 \pmod N$. We now describe our scheme in detail calling it **SK-PE-1**.

- **Setup**(1^λ): The setup algorithm first produces $(p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$ by running $\mathcal{G}(1^\lambda)$ with $\mathbb{G} = \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_3$. It then picks generators g_1, g_2, and g_3 of $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_3$. It chooses $q_{1,i}, q_{2,i} \in \mathbb{Z}_{p_1}$ and $r_{1,i}, r_{2,i} \in \mathbb{Z}_{p_3}$ uniformly at random for $i = 1$ to n. The public parameter is $N\ (= p_1 p_2 p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$. The secret key SK is the generators g_1, g_2, g_3, the prime-orders p_1, p_2, p_3, and the random numbers $\{q_{1,i}, q_{2,i}, r_{1,i}, r_{2,i}\}_{i=1}^n$.

- **Encrypt**$(SK, \vec{x})$: Let $\vec{x} = (x_1, \cdots, x_n)$ with $x_i \in \mathbb{Z}_N$. The encryption algorithm chooses $S \in \mathbb{Z}_{p_1}, \alpha_1, \alpha_2 \in \mathbb{Z}_{p_2}, U_{1,i}, U_{2,i} \in \mathbb{Z}_{p_3}$ for $i = 1$ to n, $U_1 \in \mathbb{Z}_{p_1}$, and $U_2 \in \mathbb{Z}_{p_2}$ uniformly at random. It outputs the ciphertext

$$CT := \left(\begin{array}{l} \{C_{1,i} = g_1^{Sq_{1,i}} g_2^{\alpha_1 x_i} g_3^{U_{1,i}}\}_{i=1}^n,\ \{C_{2,i} = g_1^{Sq_{2,i}} g_2^{\alpha_2 x_i} g_3^{U_{2,i}}\}_{i=1}^n, \\ C_1 = g_1^S,\ C_2 = g_1^{U_1} g_2^{U_2} \prod_{i=1}^n g_3^{-U_{1,i} r_{1,i} - U_{2,i} r_{2,i}} \end{array} \right)$$

- **GenToken**$(SK, \vec{y})$: Let $\vec{y} = (y_1, \cdots, y_n)$ with $y_i \in \mathbb{Z}_N$. This algorithm chooses $T \in \mathbb{Z}_{p_3}$, $\beta_1, \beta_2 \in \mathbb{Z}_{p_2}$, $V_{1,i}, V_{2,i} \in \mathbb{Z}_{p_1}$ for $i = 1$ to n, $V_1 \in \mathbb{Z}_{p_2}$, and $V_2 \in \mathbb{Z}_{p_3}$ uniformly at random. It then outputs the token

$$TK := \left(\begin{array}{l} \{K_{1,i} = g_1^{V_{1,i}} g_2^{\beta_1 y_i} g_3^{Tr_{1,i}}\}_{i=1}^n,\ \{K_{2,i} = g_1^{V_{2,i}} g_2^{\beta_2 y_i} g_3^{Tr_{2,i}}\}_{i=1}^n, \\ K_1 = \prod_{i=1}^n g_1^{-V_{1,i} q_{1,i} - V_{2,i} q_{1,i}} \cdot g_2^{V_1} g_3^{V_2}, K_2 = g_3^T \end{array} \right)$$

– **Check**(CT, TK): This algorithm outputs 1 iff $\prod_{i=1}^{n}\{\hat{e}(C_{1,i}, K_{1,i})\ \hat{e}(C_{2,i}, K_{2,i})\}\ \hat{e}(C_1, K_1)\hat{e}(C_2, K_2) = 1$.

Correctness

$$
\begin{aligned}
&\prod_{i=1}^{n} \hat{e}(C_{1,i}, K_{1,i})\hat{e}(C_{2,i}, K_{2,i}) \cdot \hat{e}(C_1, K_1)\hat{e}(C_2, K_2) \\
&= \prod_{i=1}^{n} \hat{e}(g_1^{Sq_{1,i}} g_2^{\alpha_1 x_i} g_3^{U_{1,i}}, g_1^{V_{1,i}} g_2^{\beta_1 y_i} g_3^{Tr_{1,i}}) \prod_{i=1}^{n} \hat{e}(g_1^{Sq_{1,i}} g_2^{\alpha_2 x_i} g_3^{U_{2,i}}, g_1^{V_{2,i}} g_2^{\beta_2 y_i} g_3^{Tr_{2,i}}) \\
&\quad \cdot \hat{e}(g_1^{S}, g_2^{V_1} g_3^{V_2} \prod_{i=1}^{n} g_1^{-V_{1,i}q_{1,i} - V_{2,i}q_{1,i}}) \cdot \hat{e}(g_1^{U_1} g_2^{U_2} \prod_{i=1}^{n} g_3^{-U_{1,i}r_{1,i} - U_{2,i}r_{2,i}}, g_3^{T}) \\
&= \prod_{i=1}^{n} \hat{e}(g_1^{Sq_{1,i}}, g_1^{V_{1,i}}) \cdot \prod_{i=1}^{n} \hat{e}(g_2^{\alpha_1 x_i}, g_2^{\beta_1 y_i}) \cdot \prod_{i=1}^{n} \hat{e}(g_3^{U_{1,i}}, g_3^{Tr_{2,i}}) \\
&\quad \cdot \prod_{i=1}^{n} \hat{e}(g_1^{Sq_{1,i}}, g_1^{V_{2,i}}) \cdot \prod_{i=1}^{n} \hat{e}(g_2^{\alpha_2 x_i}, g_2^{\beta_2 y_i}) \cdot \prod_{i=1}^{n} \hat{e}(g_3^{U_{2,i}}, g_3^{Tr_{2,i}}) \\
&\quad \cdot \prod_{i=1}^{n} \hat{e}(g_1^{S}, g_1^{-V_{1,i}q_{1,i} - V_{2,i}q_{1,i}}) \cdot \prod_{i=1}^{n} \hat{e}(g_3^{T}, g_3^{-U_{1,i}r_{1,i} - U_{2,i}r_{2,i}}) \\
&= \hat{e}(g_2, g_2)^{(\alpha_1\beta_1 + \alpha_2\beta_2)\langle \vec{x}, \vec{y} \rangle}
\end{aligned}
$$

The above expression evaluates to 1 if $\langle \vec{x}, \vec{y} \rangle = 0 \pmod{N}$, and the probability that it does not evaluate to 1 is extremely high if $\langle \vec{x}, \vec{y} \rangle \neq 0 \pmod{p_2}$. The above expression would reveal a non-trivial factor of N if $\langle \vec{x}, \vec{y} \rangle \neq 0 \pmod{p_2}$, so the probability of $\langle \vec{x}, \vec{y} \rangle \neq 0 \pmod{p_2}$ muse be negligible. If $\langle \vec{x}, \vec{y} \rangle \neq 0 \pmod{p_2}$, the probability of $\alpha_1\beta_1 + \alpha_2\beta_2 = 0 \pmod{p_2}$ is negligible because α_1, α_2, β_1, $\beta_2 \xleftarrow{R} \mathbb{Z}_{p_2}$.

Our scheme SK-PE-1 uses duplicate encoding of the ciphertext $(C_{1,i}, C_{2,i})$ and the token $(K_{1,i}, K_{2,i})$, and these two parallel subsystems are similar to parallel subsystems in the KSW scheme and the SSW scheme [23, 36].

Analysis. A predicate encryption scheme based on bilinear groups needs to use a large number of groups if it is to resist integer factorization attacks, and one might want to implement an SK-PE scheme based on a smaller number of groups without using the conversion techniques introduced by [18, 24]. SK-PE-1, which is based on only three groups, provides a message space more than 33% wider than the SSW scheme does because it uses one fewer groups. It is also resistant to integer factoring attacks with moderate security parameters because the performance of elliptic curve factorization depends on the number of nontrivial factors.

5 Security Proof for SK-PE-1

This section describes a sequence of hybrid security games demonstrating that when SK-PE-1 is used the ciphertexts and tokens reveal no unintended information about what they describe (plaintexts and predicates).

5.1 Proof Intuition

SK-PE-1 is designed so that ciphertexts and tokens are symmetric, so the most challenging aspects of providing a proof of SK-PE-1 security are showing token IND and ciphertext IND. We do this in a way similar to that done by Katz, Sahai, and Waters [23]. We use a sequence of hybrid games where a challenge token will be encrypted with one vector in the first subsystem and with another vector in the second subsystem. Let $(\vec{w}, \vec{z})$ denote a token encrypted using vector $\vec{w}$ in the first subsystem ($\{K_{1,i}\}_{i=1}^n$) and vector $\vec{z}$ in the second subsystem ($\{K_{2,i}\}_{i=1}^n$). To prove that the challenge token associated with $\vec{w}$ corresponding to $(\vec{w}, \vec{w})$ is indistinguishable from the challenge token associated with $\vec{z}$ corresponding to $(\vec{z}, \vec{z})$, we use a series of games demonstrating that $(\vec{w}, \vec{w}) \simeq (\vec{w}, \vec{0})$, $(\vec{w}, \vec{0}) \simeq (\vec{w}, \vec{z})$, $(\vec{w}, \vec{z}) \simeq (\vec{0}, \vec{z})$, and $(\vec{0}, \vec{z}) \simeq (\vec{z}, \vec{z})$.

5.2 Sequence of Hybrid Games

A security game for selective security offers the adversary a choice of predicates or plaintexts as challenges and gives the adversary a number of ciphertexts and tokens. To prove token IND, we use the following restriction: the adversary can query only tokens.

Theorem 1. *If $\mathcal{G}$ satisfies Assumption 1, SK-PE-1 has token IND.*

Proof. (Proof sketch.) To prove that SK-PE-1 has token IND, we use a sequence of hybrid games in which all ciphertexts and tokens are generated properly at query phases and only challenge tokens are differently defined (for clarity here, the differences between two consecutive games are surrounded with frames).

Game 1:
$$TK = \left(\begin{array}{l} \{K_{1,i} = g_1^{V_{1,i}} g_2^{\beta_1 w_i} g_3^{Tr_{1,i}}\}_{i=1}^n,\ \{K_{2,i} = g_1^{V_{2,i}} g_2^{\beta_2 w_i} g_3^{Tr_{2,i}}\}_{i=1}^n, \\ K_1 = \prod_{i=1}^n g_1^{-V_{1,i}q_{1,i} - V_{2,i}q_{1,i}} g_2^{V_1} g_3^{V_2},\ K_2 = g_3^T \end{array} \right)$$

Game 2:
$$TK = \left(\begin{array}{l} \{K_{1,i} = g_1^{V_{1,i}} g_2^{\beta_1 w_i} g_3^{Tr_{1,i}}\}_{i=1}^n, \boxed{\{K_{2,i} = g_1^{V_{2,i}} g_3^{Tr_{2,i}}\}_{i=1}^n}, \\ K_1 = \prod_{i=1}^n g_1^{-V_{1,i}q_{1,i} - V_{2,i}q_{1,i}} g_2^{V_1} g_3^{V_2},\ K_2 = g_3^T \end{array} \right)$$

Game 3:
$$TK = \left(\begin{array}{l} \{K_{1,i} = g_1^{V_{1,i}} g_2^{\beta_1 w_i} g_3^{Tr_{1,i}}\}_{i=1}^n, \boxed{\{K_{2,i} = g_1^{V_{2,i}} g_2^{\beta_2 z_i} g_3^{Tr_{2,i}}\}_{i=1}^n}, \\ K_1 = \prod_{i=1}^n g_1^{-V_{1,i}q_{1,i} - V_{2,i}q_{1,i}} g_2^{V_1} g_3^{V_2},\ K_2 = g_3^T \end{array} \right)$$

Game 4:
$$TK = \left(\begin{array}{l} \boxed{\{K_{1,i} = g_1^{V_i} g_3^{Tr_i}\}_{i=1}^n},\ \{K_{2,i} = g_1^{V_{2,i}} g_2^{\beta_2 z_i} g_3^{Tr_{2,i}}\}_{i=1}^n, \\ K_1 = \prod_{i=1}^n g_1^{-V_{1,i}q_{1,i} - V_{2,i}q_{1,i}} g_2^{V_1} g_3^{V_2},\ K_2 = g_3^T \end{array} \right)$$

Game 5:
$$TK = \left(\begin{array}{l} \boxed{\{K_{1,i} = g_1^{V_i} g_2^{\beta_1 z_i} g_3^{Tr_i}\}_{i=1}^n},\ \{K_{2,i} = g_1^{V_{2,i}} g_2^{\beta_2 z_i} g_3^{Tr_{2,i}}\}_{i=1}^n, \\ K_1 = \prod_{i=1}^n g_1^{-V_{1,i}q_{1,i} - V_{2,i}q_{1,i}} g_2^{V_1} g_3^{V_2},\ K_2 = g_3^T \end{array} \right)$$

If $\mathcal{G}$ satisfies Assumption 1, one can respectively prove token IND between Games 1 and 2, Games 2 and 3, Games 3 and 4, and Games 4 and 5. We describe all details in Lemmas 1, 2, and 3 in the Appendix B. Thanks to these lemmas, the adversary $\mathcal{A}$ cannot distinguish Game 1 from Game 5, each of which is generated from the challenge description of predicates $\vec{y}$ and $\vec{z}$. □

In SK-PE-1, the tokens and ciphertexts in the subgroups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ are clearly symmetric: they are formed symmetrically. The $\mathbb{G}_2$ subgroup has the same form in both the ciphertext and the token, and the $\mathbb{G}_1$ and $\mathbb{G}_3$ subgroups masking the elements in $\mathbb{G}_2$ mirror each other. Thus the proof of the ciphertext IND of SK-PE-1 is almost the same as the proof of the token IND. The only difference between them is that the challenge query is a ciphertext instead of a token.

Theorem 2. *If $\mathcal{G}$ satisfies Assumption 1, SK-PE-1 has ciphertext IND.*

Proof. The tokens and the ciphertexts of SK-PE-1 are formed symmetrically. So Assumption 1 lets us convert the sequence of security game proving Theorem 1 to a sequence proving ciphertext IND by simply renaming the subgroups $\mathbb{G}_1 \leftrightarrow \mathbb{G}_3$. The ciphertext IND thus is proven in the same way that the token IND was proved.

□

Thanks to Theorem 1 and Theorem 2, the following proposition holds.

Proposition 1. *If $\mathcal{G}$ satisfies Assumption 1, SK-PE-1 is selectively secure.*

Proof. We have already proved Theorem 1 (that SK-PE-1 has token IND) and Theorem 2 (that SK-PE-1 has ciphertext IND). If the adversary $\mathcal{A}$ has advantage ϵ in breaking token IND or ciphertext IND, then the simulator $\mathcal{B}$ also has the same advantage ϵ in breaking Assumption 1. This completes the proof of Proposition 1.

□

6 Prime-Order Group Instantiation

The construction introduced in Section 4 uses the fact that if each element g, h belongs to distinct prime-order subgroups, then $\hat{e}(g, h) = 1$. To create more efficient versions, this section introduces a construction taking the pairing on the composite-order groups to be any nontrivial linear combination of the pairings on the prime-order groups, for which both Freeman and Lewko have given ways [18, 24]. We call this construction as **SK-PE-2**.

6.1 Our Assumption for a 3-Cancelling Bilinear Group

This subsection prepares the way for instantiating our second scheme by using prime-order groups with an asymmetric paring that makes it more efficient than SK-PE-1, which was instantiated with a symmetric paring requiring supersingular curves (details are described in [18]).

We introduce a 3-cancelling bilinear group generator algorithm $\mathcal{G}_{3c}$ that takes as input a security parameter λ and outputs a description of abelian group G, H, G_T. We assume that this description permits efficient group operations and random sampling in each group.

Definition 4. ([18]) Let $\mathcal{P}$ be a prime-order bilinear group generator. Define $\mathcal{G}_{3c}$ to be a bilinear group generator that on input λ does the following:

1. $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{P}(1^\lambda)$,
2. $G = \mathbb{G}_1^3$, $H = \mathbb{G}_2^3$, $G_T = \mathbb{G}_T$,
3. $g_1, g_2, g_3 \xleftarrow{R} \mathbb{G}_1$, $h_1, h_2, h_3 \xleftarrow{R} \mathbb{G}_2$,
4. choose x, y, z, u, v, $w \xleftarrow{R} \mathbb{Z}_p$ s.t. $\begin{cases} -xv - xw - yu + yw + zu + zv \neq 0, \\ xv - xw - yu + yw + zu - zv \neq 0. \end{cases}$
5. Define the subgroups $G_1 = \langle (g_1, g_1^x, g_1^u) \rangle$, $G_2 = \langle (g_2, g_2^y, g_2^v) \rangle$, $G_3 = \langle (g_3, g_3^z, g_3^w) \rangle$, $H_1 = \langle (h_1^{zv-yw}, h_1^{w-v}, h_1^{y-z}) \rangle$, $H_2 = \langle (h_2^{zu-xw}, h_2^{w-u}, h_2^{x-z}) \rangle$, $H_3 = \langle (h_3^{yu-xv}, h_3^{v-u}, h_3^{x-y}) \rangle$.
6. Define $e : G \times H \to G_t$ by $e((g, g', g'')(h, h', h'')) := \hat{e}(g, h)\hat{e}(g', h')\hat{e}(g'', h'')$.
7. Output the tuple $(G, G_1, G_2, G_3, H, H_1, H_2, H_3, G_t, e)$.

The inequalities in Step 4 guarantee that G_i and H_i are linearly independent, which implies the pairing e is non-degenerate. That is,

- $G \simeq G_1 \times G_2 \times G_3$ and $H \simeq H_1 \times H_2 \times H_3$
- $e(g_i, h_j) = 1$ with $g_i \in G_i$, $h_j \in H_j$ and $i \neq j$.

6.2 Our Construction of SK-PE Using Prime-Order Groups

We introduce the procedure using a bilinear group generator $\mathcal{G}_{3c}$ with an asymmetric pairing. We streamline the setup function by computing ciphertexts in G and tokens in H. Since secret keys are used to form ciphertexts and tokens, we generate two keys: one will be in G and the other in H.

- **Setup**(1^λ): The setup algorithm first runs $\mathcal{G}_{3c}(1^\lambda)$ to produce N $(:= lcm(p_1, p_2, p_3), G, G_1, G_2, G_3, H, H_1, H_2, H_3, G_T, e)$ and then picks generators $g_1, g_2, g_3, h_1, h_2, h_3$ of $G_1, G_2, G_3, H_1, H_2, H_3$. It chooses $q_{1,i}, q_{2,i} \in \mathbb{Z}_{p_1}$, $r_{1,i}, r_{2,i} \in \mathbb{Z}_{p_3}$ uniformly at random for $i = 1$ to n. The public parameter is (N, G, H, G_T, e). The secret key SK is the generators $g_1, g_2, g_3, h_1, h_2, h_3$ and the random numbers $\{q_{1,i}, q_{2,i}, r_{1,i}, r_{2,i}\}_{i=1}^n$.
- **Encrypt**$(SK, \vec{x})$: Let $\vec{x} = (x_1, \cdots, x_n)$ with $x_i \in \mathbb{Z}_{p_2}$. The encryption algorithm chooses $S \in \mathbb{Z}_{p_1}$, $\alpha_1, \alpha_2 \in \mathbb{Z}_{p_2}$, $U_{1,i}, U_{2,i} \in \mathbb{Z}_{p_3}$ for $i = 1$ to n, $U_1 \in \mathbb{Z}_{p_1}$ $U_2 \in \mathbb{Z}_{p_2}$ uniformly at random. It outputs the ciphertext

$$CT := \left(\begin{array}{l} \{C_{1,i} = g_1^{Sq_{1,i}} g_2^{\alpha_1 x_i} g_3^{U_{1,i}}\}_{i=1}^n, \ \{C_{2,i} = g_1^{Sq_{1,i}} g_2^{\alpha_2 x_i} g_3^{U_{2,i}}\}_{i=1}^n, \\ C_1 = g_1^S, \ C_2 = g_1^{U_1} g_2^{U_2} \prod_{i=1}^n g_3^{-U_{1,i} r_{1,i} - U_{2,i} r_{2,i}} \end{array} \right)$$

- **GenToken**$(SK, \vec{y})$: Let $\vec{y} = (y_1, \cdots, y_n)$ with $y_i \in \mathbb{Z}_{p_2}$. This algorithm chooses $T \in \mathbb{Z}_{p_3}$, $\beta_1, \beta_2 \in \mathbb{Z}_{p_2}$ and $V_{1,i}, V_{2,i} \in \mathbb{Z}_{p_1}$ for $i = 1$ to n, $V_1 \in \mathbb{Z}_{p_2}$, $V_2 \in \mathbb{Z}_{p_3}$ uniformly at random. It then outputs the token

$$TK := \left(\begin{array}{l} \{K_{1,i} = h_1^{V_{1,i}} h_2^{\beta_1 y_i} h_3^{Tr_{1,i}}\}_{i=1}^n, \ \{K_{2,i} = h_1^{V_{2,i}} h_2^{\beta_2 y_i} h_3^{Tr_{2,i}}\}_{i=1}^n, \\ K_1 = h_2^{V_1} h_3^{V_2} \prod_{i=1}^n \{h_1^{-V_{1,i} q_{1,i} - V_{2,i} q_{1,i}}\}, K_2 = h_3^T \end{array} \right)$$

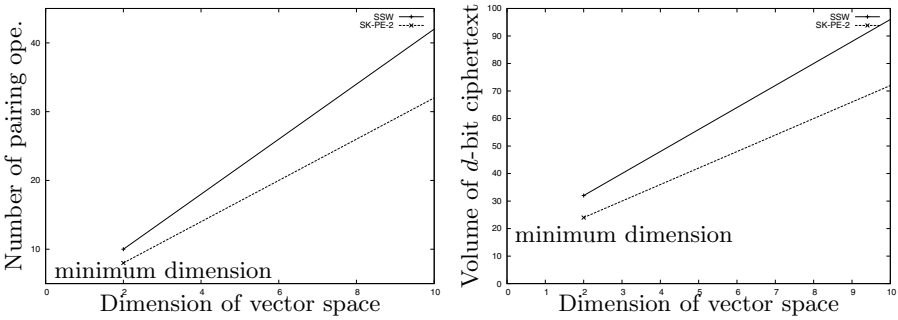


Fig. 2. SK-PE-2 and the SSW schemes compared in terms of pairing operation calls in the Check function (left) and ciphertext size (right)

– **Check**(CT, TK): This algorithm outputs 1 iff $\prod_{i=1}^{n}\{e(C_{1,i}, K_{1,i})\ e(C_{2,i}, K_{2,i})\}\ e(C_1, K_1)e(C_2, K_2) = 1$.

The cancelling property of the componentwise pairing e implies that the Check function outputs $e(g_2, h_2)^{(\alpha_1\beta_1+\alpha_2\beta_2)\langle \vec{x}\,\vec{y}\rangle}$. If $\langle x, y\rangle = 0$, then we obtain 1; otherwise, there is an extremely high probability that we obtain random elements of $\mathbb{G}_2$.

Analysis. The number of elements required for the prime-order group instantiations is exactly proportional to the original number of hidden subgroups required for the composite-order bilinear group instantiations. SK-PE-2 using d-bit prime groups generates $6d(n + 2)$-bit ciphertext and requires d-bit pairing operations $(3n + 2)$ times in the Check function. Figure 2 shows number of pairing operations in left side and size of ciphertexts and tokens of SK-PE-2 and the SSW scheme in right side. As a consequence, SK-PE-2 is asymptotically more than 33% faster than the SSW scheme requiring four groups, and its ciphertexts and tokens are asymptotically 25% smaller than those of the SSW scheme requiring four groups.

Acknowledgments. This work has been supported by Ministry of Internal Affairs and Communications of the Japanese Government.

References

1. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-Policy Attribute-Based Encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334. IEEE Computer Society (2007)
2. Blundo, C., Iovino, V., Persiano, G.: Predicate Encryption with Partial Public Keys. IACR Cryptology ePrint Archive, 2010:476 (2010)
3. 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)

4. Boneh, D., Boyen, X.: Secure Identity Based Encryption Without Random Oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004)
5. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. SIAM J. Comput. 36(5), 1301–1328 (2007)
6. 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)
7. Boneh, D., Franklin, M.K.: Identity based encryption from the weil pairing. IACR Cryptology ePrint Archive, 2001:90 (2001)
8. 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)
9. Boneh, D., Sahai, A., Waters, B.: Functional Encryption: Definitions and Challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011)
10. Boneh, D., Waters, B.: A fully collusion resistant broadcast, trace, and revoke system. In: ACM Conference on Computer and Communications Security, pp. 211–220 (2006)
11. 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)
12. Boyen, X., Waters, B.: Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). IACR Cryptology ePrint Archive, 85 (2006)
13. Canetti, R., Halevi, S., Katz, J.: A Forward-Secure Public-Key Encryption Scheme. J. Cryptology 20(3), 265–294 (2007)
14. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private Information Retrieval. J. ACM 45(6), 965–981 (1998)
15. Cocks, C.: An Identity Based Encryption Scheme Based on Quadratic Residues. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (2001)
16. Curtmola, R., Garay, J.A., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. In: ACM Conference on Computer and Communications Security, pp. 79–88. ACM (2006)
17. Dawn, A.P., Song, X., Wagner, D.: Practical Techniques for Searches on Encrypted Data. In: The 2000 IEEE Symposium on Security and Privacy, pp. 44–55 (2000)
18. Freeman, D.M.: Converting Pairing-Based Cryptosystems from Composite-Order Groups to Prime-Order Groups. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 44–61. Springer, Heidelberg (2010)
19. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
20. Goldreich, O., Ostrovsky, R.: Software Protection and Simulation on Oblivious RAMs. J. ACM 43(3), 431–473 (1996)
21. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM Conference on Computer and Communications Security, pp. 89–98. ACM (2006)
22. Horwitz, J., Lynn, B.: Toward Hierarchical Identity-Based Encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 466–481. Springer, Heidelberg (2002)

23. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
24. Lewko, A.: Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 318–335. Springer, Heidelberg (2012) Full version is available, http://eprint.iacr.org/2011/490.pdf
25. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
26. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
27. Okamoto, T., Takashima, K.: Achieving Short Ciphertexts or Short Secret-Keys for Adaptively Secure General Inner-Product Encryption. In: Lin, D., Tsudik, G., Wang, X. (eds.) CANS 2011. LNCS, vol. 7092, pp. 138–159. Springer, Heidelberg (2011)
28. Okamoto, T., Takashima, K.: Adaptively attribute-hiding (hierarchical) inner product encryption. IACR Cryptology ePrint Archive, 2011:543 (2011)
29. Ostrovsky, R., Skeith III, W.E.: A Survey of Single-Database Private Information Retrieval: Techniques and Applications. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 393–411. Springer, Heidelberg (2007)
30. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: ACM Conference on Computer and Communications Security, pp. 195–203. ACM (2007)
31. Pinkas, B., Reinman, T.: Oblivious RAM Revisited. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 502–519. Springer, Heidelberg (2010)
32. Popa, R.A., Redfield, C.M.S., Zeldovich, N., Balakrishnan, H.: CryptDB: protecting confidentiality with encrypted query processing. In: Wobber, T., Druschel, P. (eds.) SOSP, pp. 85–100. ACM (2011)
33. Rhee, H.S., Jeong, I.R., Byun, J.W., Lee, D.-H.: Difference Set Attacks on Conjunctive Keyword Search Schemes. In: Jonker, W., Petković, M. (eds.) SDM 2006. LNCS, vol. 4165, pp. 64–74. Springer, Heidelberg (2006)
34. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
35. 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)
36. 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)
37. Shi, E., Bethencourt, J., Chan, H.T.-H., Song, D.X., Perrig, A.: Multi-Dimensional Range Query over Encrypted Data. In: IEEE Symposium on Security and Privacy, pp. 350–364. IEEE Computer Society (2007)
38. 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)

A Validation of Assumption 1

This section proves the soundness of Assumption 1 by showing that it holds in generic bilinear groups of composite order.

This section considers cyclic bilinear groups of composite order N, where $N = \prod_{i=1}^{m} p_i$ is the product of m distinct primes. Each element $g \in \mathbb{G}$ can be written as $g = g_1^{a_1} g_2^{a_2} \dots g_m^{a_m}$ where $a_i \in \mathbb{Z}_{p_i}$ and g_i denotes some fixed generator of the subgroup of order p_i. We can therefore represent each element $g \in \mathbb{G}$ as an m-tuple $(a_1, \dots, a_m)$. We use the same representation with elements in the target group $\mathbb{G}_T$ for the paring $\hat{e}$: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$.

In an experiment involving the generic group, we will describe an algorithm with a set of elements generated at random according to some distribution. These random variables use capital letters that are each chosen independently and uniformly at random from the approximation. That is, a random element of $\mathbb{G}$ would be described as $(X_1, \dots, X_m)$, where each X_i is chosen uniformly from $\mathbb{Z}_{p_i}$. We say a random variable expressed in this way has degree t if the maximum degree of any variable is t. Random variables taking values in $\mathbb{G}_T$ are expressed in the same way but using bracket notation.

Given random variables X, A_1, …, A_ℓ over the same group, X is said to *be dependent on* $\{A_i\}$ if there exist $\gamma_i \in \mathbb{Z}^*$ such that $X = \sum_i \gamma_i A_i$, where equality refers to equality in terms of the underlying formal variables. If no such γ_i exists, then X is said to be independent of $\{A_i\}$.

Given a random variable $A = (X_1, \dots, X_m)$, when we say that an algorithm is given A we mean that random $x_1, \dots, x_m$ are chosen appropriately and the adversary is given (the handle for) the element $(x_1, \dots, x_m)$.

Theorem 3. *[23] Let $N = \prod_{i=1}^{m} p_i$ be a product of distinct primes, each greater than 2^n. Let $\{A_i\}, T_0, T_1$ be random variables over $\mathbb{G}$, and let $\{B_i\}$ be random variables over $\mathbb{G}_T$, where all random variable have degree at most t.*

Let $S \leftarrow \{i | \hat{e}(T_0, A_i) \neq \hat{e}(T_1, A_i)\}$. Say each of T_0 and T_1 is independent of $\{A_i\}$, and furthermore that for all $k \in S$ it holds that $\hat{e}(T_0, A_k)$ is independent of $\{B_i\} \cup \{\hat{e}(A_i, A_j)\} \cup \{\hat{e}(T_0, A_i)\}_{i \neq k}$. Then given any algorithm A issuing at most q instructions and having advantage δ, the algorithm can be used to find a non-trivial factor of N with at least $\delta - O(q^2 t / 2^n)$.

We now show how to apply Theorem 3 of [23] to prove that our assumption hold in the generic group model.

Assumption 1. Using the notations of Theorem 3, Assumption 1 may be written as: $A_1 = (1, 0, 0), A_2 = (a_1, 1, 0), A_3 = (0, b_1, c_1), A_4 = (0, 0, c_2 d), A_5 = (0, 0, d), A_6 = (0, 0, d^2), A_7 = (a_2, 0, c_1 d), T_0 = (a_3, 0, c_2), T_1 = (a_3, b_2, c_2)$. Note that under Assumption 1 no element in $\mathbb{G}_T$ is given to the adversary, $\{B_i\}$ is thus $\emptyset$. It is not difficult to see that both T_0 and T_1 are independent of $\{A_i\}$. We thus give a description of independence on $\mathbb{G}_T$. Using the notation of Theorem 3, we have $S = \{2, 3\}$ since only A_2 and A_3 have non-zero second elements in $\mathbb{G}$.

Considering T_0 first, we obtain the following tuples: $\hat{e}(T_0, A_2) = [a_1 a_3, 0, 0]$, $\hat{e}(T_0, A_3) = [0, 0, c_1 c_2]$. It is clear that $\hat{e}(T_0, A_2)$ is independent of anything else,

since an element in $\mathbb{G}_T$ whose first component contains $a_1 a_3$ cannot be generated any other way. Similarly, $\hat{e}(T_0, A_3)$ is independent of anything else, since an element in $\mathbb{G}_T$ whose third component contains $c_1 c_2$ cannot be generated. Thus, the above two elements satisfy the independence requirement of Theorem 3. Exactly analogous arguments apply for the case of T_1.

B A Sequence of Hybrid Games among Games 1 – 5

This section proves that SK-PE-1 satisfies Definition 2 (token IND). Using a sequence of hybrid games, Game 1–5 are proven to be indistinguishable.

Lemma 1. *If $\mathcal{G}$ satisfies Assumption 1, Game 1 and Game 2 are computationally indistinguishable.*

Proof. We build a simulator $\mathcal{B}$ that tries to break Assumption 1. The simulator $\mathcal{B}$ uses an adversary $\mathcal{A}$ that tries to distinguish Game 1 from Game 2. If $\mathcal{A}$ has advantage ϵ in distinguishing Game 1 from Game 2, then $\mathcal{B}$ has the same advantage ϵ in breaking Assumption 1.

Fix an adversary $\mathcal{A}$. The simulator $\mathcal{B}$ is given an instance of Assumption 1: the public parameter $(N, \mathbb{G}, \mathbb{G}_T, \hat{e})$ along with the elements $(g_1, g_1^{a_1} g_2, g_2^{b_1} g_3^{c_1}, g_3^{c_2 d}, g_3^{d}, g_3^{d^2}, g_1^{a_2} g_3^{c_1 d})$, and an element $T_{\mathsf{b}} = g_1^{a_3} g_2^{\mathsf{b} b_2} g_3^{c_2}$ where a random bit $\mathsf{b} \in \{0, 1\}$ and $a_1, a_2, a_3 \xleftarrow{R} \mathbb{Z}_{p_1}$, $b_1, b_2 \xleftarrow{R} \mathbb{Z}_{p_2}$, $c_1, c_2, d \xleftarrow{R} \mathbb{Z}_{p_3}$, $g_1 \xleftarrow{R} \mathbb{G}_1$, $g_2 \xleftarrow{R} \mathbb{G}_2$, $g_3 \xleftarrow{R} \mathbb{G}_3$.

Setup: $\mathcal{A}$ is given the public parameter and outputs description of two challenge predicates $\vec{w}$, $\vec{z} \in (\mathbb{Z}_N)^n$ to $\mathcal{B}$. $\mathcal{B}$ gives the predicates $\vec{w}$, $\vec{z}$ to the challenger $\mathcal{C}$, and generates random numbers $\{q_{1,i}, q_{2,i}\}_{i=1}^n$, $\{r_{1,i}\}_{i=1}^n$, $\{r'_{2,i}\}_{i=1}^n \xleftarrow{R} \mathbb{Z}_N$.

Phase 1 (token): $\mathcal{B}$ receives description of predicate $\vec{y} = \langle y_1, \ldots, y_n \rangle$ from $\mathcal{A}$. $\mathcal{B}$ generates elements T', β_1, β_2, $\{V'_{1,i}\}_{i=1}^n$, $\{V'_{2,i}\}_{i=1}^n$, V'_1, $V'_2 \xleftarrow{R} \mathbb{Z}_N$ respectively and outputs the following token for $\mathcal{A}$:

$$
\begin{aligned}
K_{1,i} &= g_1^{V'_{1,i}} (g_1^{a_1} g_2)^{\beta_1 y_i} (g_3^{d^2})^{T' r_{1,i}} = g_1^{V_{1,i}} g_2^{\beta_1 y_i} g_3^{T r_{1,i}} \\
K_{2,i} &= g_1^{V'_{2,i}} (g_1^{a_1} g_2)^{\beta_2 y_i} (g_3^{d})^{T' w_i} (g_3^{d^2})^{T' r'_{2,i}} = g_1^{V_{2,i}} g_2^{\beta_2 y_i} g_3^{T r_{2,i}} \\
K_1 &= \prod_{i=1}^{n} \left(K_{1,i}^{-q_{1,i}} K_{2,i}^{-q_{2,i}}\right) (g_2^{b_1} g_3^{c_1 d})^{V'_1} (g_3^{d})^{V'_2} = g_1^{-\sum_{i=1}^n (V_{1,i} q_{1,i} + V_{2,i} q_{1,i})} g_2^{V_1} g_3^{V_2} \\
K_2 &= (g_3^{d^2})^{T'} = g_3^{T}
\end{aligned}
$$

with $T = d^2 T'$, $V_{1,i} = \beta_1 a_1 y_i + V'_{1,i}$, $V_{2,i} = \beta_2 a_1 y_i + V'_{2,i}$, $r_{2,i} = d^{-1} w_i + r'_{2,i}$, $V_1 = b_1 V'_1 - \beta_1 \sum_{i=1}^n q_{1,i} y_i - \beta_2 \sum_{i=1}^n q_{2,i} y_i$, $V_2 = c_1 d V'_1 + d V'_2 - T(\sum_{i=1}^n q_{1,i} r_{1,i} + \sum_{i=1}^n q_{2,i} r_{2,i})$.

Note that $V_{1,i}, V_{2,i}$ for $i = 1$ to n and V_1, V_2 are uniformly distributed in $\mathbb{Z}_N$. Thus, distribution of token in Phase 1 is completely the same as that of the real token defined in Section 4.

Phase 1 (ciphertext): $\mathcal{B}$ receives plaintext $\vec{x}= \langle x_1, \ldots, x_n \rangle$ from $\mathcal{A}$ subject to the restriction of token challenge: $\langle \vec{x}, \vec{w} \rangle = \langle \vec{x}, \vec{z} \rangle$. $\mathcal{B}$ generates elements $S, \alpha'_1, \alpha'_2, \{U'_{1,i}\}_{i=1}^n$, $\{U'_{2,i}\}_{i=1}^n$, $U'_1, U'_2 \xleftarrow{R} \mathbb{Z}_N$ respectively and outputs the following ciphertext for $\mathcal{A}$:

$$
\begin{aligned}
C_{1,i} &= (g_1)^{Sq_{1,i}} (g_2^{b_1} g_3^{c_1})^{\alpha'_1 x_i} (g_3^d)^{U'_{1,i}} = g_1^{Sq_{1,i}} g_2^{\alpha_1 x_i} g_3^{U_{1,i}}. \\
C_{2,i} &= (g_1)^{Sq_{2,i}} (g_2^{b_1} g_3^{c_1})^{\alpha'_2 x_i} (g_3^d)^{U'_{2,i}} = g_1^{Sq_{2,i}} g_2^{\alpha_2 x_i} g_3^{U_{2,i}}. \\
C_1 &= g_1^S. \\
C_2 &= \prod_{i=1}^n \left(g_3^{d^2}\right)^{-U'_{1,i} r_{1,i}} \prod_{i=1}^n \left(g_3^{d^2}\right)^{-U'_{2,i} r'_{2,i}} \prod_{i=1}^n \left(g_3^d\right)^{-U'_{2,i} w_i} \cdot (g_1^{a_1} g_2)^{U'_1} g_1^{U'_2} \\
&= g_1^{a_1 U'_1 + U'_2} g_2^{U'_1} g_3^{-\sum_{i=1}^n U_{1,i} r_{1,i} - \sum_{i=1}^n U_{2,i} (r'_{2,i} + d^{-1} w_i)} \\
&= g_1^{U_1} g_2^{U_2} g_3^{-\sum_{i=1}^n (U_{1,i} r_{1,i} + U_{2,i} r_{2,i})}.
\end{aligned}
$$

with $\alpha_1 = b_1 \alpha'_1$, $\alpha_2 = b_1 \alpha'_2$, $\{U_{1,i} = d^2 U'_{1,i} + c_1 \alpha'_1 x_i\}_{i=1}^n$, $\{U_{2,i} = d^2 U'_{2,i} + c_1 \alpha'_2 x_i\}_{i=1}^n$, $U_1 = a_1 U'_1 + U'_2$, $U_2 = U'_1$.

$U_{1,i}, U_{2,i}$ for $i = 1$ to n and U_1, U_2 are uniformly distributed in $\mathbb{Z}_N$. Thus, distribution of ciphertext in Phase 1 is completely the same as that of the real ciphertext.

Challenge: $\mathcal{B}$ receives query of challenge token from $\mathcal{A}$. Given the challenge query for Assumption 1: $T_{\mathtt{b}} = g_1^{a_3} g_2^{\mathtt{b} b_2} g_3^{c_2}$ with a random bit $\mathtt{b} \in \{0, 1\}$ and random numbers $a_3 \xleftarrow{R} \mathbb{Z}_{p_1}, b_2 \xleftarrow{R} \mathbb{Z}_{p_2}, c_2, d \xleftarrow{R} \mathbb{Z}_{p_3}$. $\mathcal{B}$ generates random numbers $\beta_1, \beta_2, \{V'_{1,i}\}_{i=1}^n, \{V'_{2,i}\}_{i=1}^n, V'_1, V'_2 \xleftarrow{R} \mathbb{Z}_N$ respectively and output challenge token for $\mathcal{A}$.

$$
\begin{aligned}
K_{1,i} &= g_1^{V'_{1,i}} (g_1^{a_1} g_2)^{\beta_1 w_i} (g_3^{c_2 d})^{r_{1,i}} = g_1^{V_{1,i}} g_2^{\beta_1 w_i} g_3^{T r_{1,i}} \\
K_{2,i} &= (T_{\mathtt{b}})^{w_i} g_1^{V'_{2,i}} (g_3^{c_2 d})^{r'_{2,i}} = g_1^{V'_{2,i} + a_3 w_i} g_2^{\mathtt{b} b_2 w_i} g_3^{c_2 d (d^{-1} w_i + r'_{2,i})} = g_1^{V_{2,i}} g_2^{\beta_2 w_i} g_3^{T r_{2,i}} \\
K_1 &= g_3^{c_2 d} = g_3^T \\
K_2 &= \prod_{i=1}^n \left(K_{1,i}^{-q_{1,i}} K_{2,i}^{-q_{2,i}}\right) (g_3^{c_2 d} g_2^{b_1})^{V'_1} g_3^{V'_2} = g_1^{-\sum_{i=1}^n (V_{1,i} q_{1,i} + V_{2,i} q_{1,i})} g_2^{V_1} g_3^{V_2}
\end{aligned}
$$

with $c_2 d = T$, $\beta_2 = \mathtt{b} b_2$, $\{V_{1,i} = V'_{1,i} + a_1 \beta_1 w_i\}_{i=1}^n$, $\{V_{2,i} = V'_{2,i} + a_3 w_i\}_{i=1}^n$, $V_1 = b_1 V'_1 - \sum_{i=1}^n (\beta_1 q_{1,i} + \beta_2 q_{1,i}) w_i$ and $V_2 = T V'_1 + V'_2 - T \sum_{i=1}^n (q_{1,i} r_{1,i} + q_{1,i} r_{2,i})$.

By examining the projections of the components of the challenge ciphertext in the groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$, it can be verified that when $T_1 = g_1^{a_3} g_2^{b_2} g_3^{c_2}$ is given, the challenge token is distributed exactly as in Game 1, whereas if $T_0 = g_1^{a_3} g_3^{c_2}$ is given, the challenge token is distributed exactly as in Game 2.

Phase 2 (token/ciphertext): $\mathcal{B}$ behaves the same as Phase 1.

Guess: $\mathcal{B}$ receives the guess $\mathtt{b}'$ from $\mathcal{A}$ and outputs the same guess $\mathtt{b}'$ to $\mathcal{C}$.

Clearly, if the adversary $\mathcal{A}$ has advantage ϵ in distinguishing Game 1 from Game 2, then the simulator $\mathcal{B}$ also has the same advantage ϵ in breaking Assumption 1. This completes the proof of Lemma 1.

□

Lemma 2. *If $\mathcal{G}$ satisfies Assumption 1, Game 2 and Game 3 are computationally indistinguishable.*

Proof. We build a simulator $\mathcal{B}$ using an adversary $\mathcal{A}$, which tries to distinguish Game 2 from Game 3.

The simulator $\mathcal{B}$ is given an instance of Assumption 1: the public parameter $(N, \mathbb{G}, \mathbb{G}_T, \hat{e})$ along with the elements $(g_1, g_1^{a_1}g_2, g_2^{b_1}g_3^{c_1}, g_3^{c_2 d}, g_3^{d}, g_3^{d^2}, g_1^{a_2}g_3^{c_1 d})$, and an element $T_{\mathtt{b}} = g_1^{a_3}g_2^{\mathtt{b}b_2}g_3^{c_2}$ where a random bit $\mathtt{b} \in \{0,1\}$ and random numbers $a_1, a_2, a_3 \xleftarrow{R} \mathbb{Z}_{p_1}$, $b_1, b_2 \xleftarrow{R} \mathbb{Z}_{p_2}$, $c_1, c_2, d \xleftarrow{R} \mathbb{Z}_{p_3}$, $g_1 \xleftarrow{R} \mathbb{G}_1$, $g_2 \xleftarrow{R} \mathbb{G}_2$, $g_3 \xleftarrow{R} \mathbb{G}_3$.

Setup: $\mathcal{A}$ is given the public parameter and outputs description of two challenge predicates $\vec{w}$, $\vec{z} \in (\mathbb{Z}_N)^n$ to $\mathcal{B}$. $\mathcal{B}$ gives the predicates $\vec{w}$, $\vec{z}$ to the challenger $\mathcal{C}$, and generates random numbers $\{q_{1,i}, q_{2,i}\}_{i=1}^n$, $\{r_{1,i}\}_{i=1}^n$, $\{r'_{2,i}\}_{i=1}^n \xleftarrow{R} \mathbb{Z}_N$.

Phase 1 (token): $\mathcal{B}$ receives description of predicate $\vec{y} = \langle y_1, \ldots, y_n \rangle$ from $\mathcal{A}$. $\mathcal{B}$ generates elements T', β_1, β_2, $\{V'_{1,i}\}_{i=1}^n$, $\{V'_{2,i}\}_{i=1}^n$, V'_1, $V'_2 \xleftarrow{R} \mathbb{Z}_N$ respectively and outputs the token similar with the case of Lemma 1 for $\mathcal{A}$. The difference in Phase 1 (token) between Lemma 1 and Lemma 2 is that $\mathcal{B}$ implicitly sets $r_{2,i}$ such that $r_{2,i} = d^{-1}z_i + r'_{2,i}$ for $i = 1$ to n.

Phase 1 (ciphertext): $\mathcal{B}$ receives plaintext $\vec{x} = \langle x_1, \ldots, x_n \rangle$ from $\mathcal{A}$ subject to the restriction of token challenge: $\langle \vec{x}, \vec{w} \rangle = \langle \vec{x}, \vec{z} \rangle$. $\mathcal{B}$ generates random numbers S, α_1, α_2, $\{U'_{1,i}\}_{i=1}^n$, $\{U'_{2,i}\}_{i=1}^n$, U'_1, $U'_2 \xleftarrow{R} \mathbb{Z}_N$ respectively and outputs the ciphertext similar with the case of Lemma 1 for $\mathcal{A}$. The difference in Phase 1 (ciphertext) between Lemma 1 and Lemma 2 is that $\mathcal{B}$ implicitly sets $r_{2,i}$ such that $r_{2,i} = d^{-1}z_i + r'_{2,i}$ for $i = 1$ to n.

Challenge: $\mathcal{B}$ receives query of challenge token from $\mathcal{A}$. Given the challenge query for Assumption 1: $T_{\mathtt{b}} = g_1^{a_3}g_2^{\mathtt{b}b_2}g_3^{c_2}$ with a random bit $\mathtt{b} \in \{0,1\}$ and random numbers $a_3 \xleftarrow{R} \mathbb{Z}_{p_1}, b_2 \xleftarrow{R} \mathbb{Z}_{p_2}, c_2, d \xleftarrow{R} \mathbb{Z}_{p_3}$. $\mathcal{B}$ generates random numbers $\beta_1, \beta_2, \{V'_{1,i}\}_{i=1}^n, \{V'_{2,i}\}_{i=1}^n, V'_1, V'_2 \xleftarrow{R} \mathbb{Z}_N$ respectively and output challenge token for $\mathcal{A}$.

$$K_{1,i} = g_1^{V'_{1,i}}(g_1^{a_1}g_2)^{\beta_1 w_i}(g_3^{c_2 d})^{r_{1,i}} = g_1^{V_{1,i}} g_2^{\beta_1 w_i} g_3^{T r_{1,i}}$$

$$K_{2,i} = (T_{\mathfrak{b}})^{z_i} g_1^{V'_{2,i}}(g_3^{c_2 d})^{r'_{2,i}} = g_1^{V'_{2,i}+a_3 z_i} g_2^{\mathfrak{b} b_2 w_i} g_3^{c_2 d(d^{-1} z_i + r'_{2,i})} = g_1^{V_{2,i}} g_2^{\beta_2 z_i} g_3^{T r_{2,i}}$$

$$K_1 = g_3^{c_2 d} = g_3^T$$

$$K_2 = \prod_{i=1}^{n} \left(K_{1,i}^{-q_{1,i}} K_{2,i}^{-q_{2,i}}\right)(g_3^{c_2 d} g_2^{b_1})^{V'_1} g_3^{V'_2} = g_1^{-\sum_{i=1}^{n}(V_{1,i} q_{1,i} + V_{2,i} q_{1,i})} g_2^{V_1} g_3^{V_2}$$

with $c_2 d = T$, $\beta_1 = \mathfrak{b} b_2$, $\{V_{1,i} = V'_{1,i} + a_1 \beta_1 w_i\}_{i=1}^n$, $\{V_{2,i} = V'_{2,i} + a_3 \beta_2 z_i\}_{i=1}^n$, $V_1 = b_1 V'_1 - \sum_{i=1}^n (\beta_1 q_{1,i} w_i + \beta_2 q_{1,i} z_i)$ and $V_2 = T V'_1 + V'_2 - T \sum_{i=1}^n (q_{1,i} r_{1,i} + q_{1,i} r_{2,i})$.

By examining the projections of the components of the challenge ciphertext in the groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$, it can be verified that when $T_0 = g_1^{a_3} g_3^{c_2}$ is given, the challenge token is distributed exactly as in Game 2, whereas if $T_1 = g_1^{a_3} g_2^{b_2} g_3^{c_2}$ is given, the challenge token is distributed exactly as in Game 3.

Phase 2 (token/ciphertext): $\mathcal{B}$ behaves the same as Phase 1.

Guess: $\mathcal{B}$ receives the guess $\mathfrak{b}'$ from $\mathcal{A}$ and outputs the same guess $\mathfrak{b}'$ to $\mathcal{C}$.

Clearly, if the adversary $\mathcal{A}$ has advantage ϵ in distinguishing Game 3 from Game 2, then the simulator $\mathcal{B}$ also has the same advantage ϵ in breaking Assumption 1. This completes the proof of Lemma 2.

Lemma 3. *If $\mathcal{G}$ satisfies Assumption 1, Game 3 and Game 5 are computationally indistinguishable.*

Proof. SK-PE-1 is symmetric with respect to the roles of elements in $\mathbb{G}_1$ and $\mathbb{G}_3$. Thus, the proof that Game 3 and Game 4 are indistinguishable exactly parallels the proof of Lemma 2, while the proof that Game 4 and Game 5 are indistinguishable exactly parallels the proof of Lemma 1.

□

Secure Keyword Search Using Bloom Filter with Specified Character Positions

Takanori Suga[1], Takashi Nishide[2], and Kouichi Sakurai[2]

[1] Department of Informatics, Graduate School of Information Science and Electrical Engineering, Kyushu University
[2] Department of Informatics, Faculty of Information Science and Electrical Engineering, Kyushu University

Abstract. There are encryption schemes called searchable encryption which enable keyword searches. Traditional symmetric ones support only full keyword matches. Therefore, both a data owner and data searcher have to enumerate all possible keywords to realize a variety of searches. It causes increases of data size and run time.We propose searchable symmetric encryption which can check characters in the specified position as we perform search on plaintexts. Our scheme realizes a variety of searches such as fuzzy keyword search, wildcard search, and so on.

Keywords: Symmetric encryption, searchable encryption, Bloom filter.

1 Introduction

1.1 Background

In recent years, cloud computing is spreading rapidly and widely due to advance in computer and telecommunication technology. In cloud computing, we outsource not only data but also processing to cloud servers.

The users cannot carry out investigations into the cause of security incidents and the measures for preventing the recurrence of such incidents because the users cannot know how cloud providers manage their servers. Therefore, we need the measures for preventing the leakages before sending data to the cloud servers. However, traditional encryption schemes prevent not only the leakages but also the conveniences like searches. There are encryption schemes which enable us to search without decryption. These schemes are called searchable encryption and attract a great deal of attention.

There exist symmetric searchable encryption schemes and asymmetric ones. The asymmetric ones can be used when it is difficult for us to share a secret key securely as we send an e-mail. On the other hand, we can use symmetric ones if and only if we can share a secret key securely, for example, when we share files in the same organization. However, the symmetric ones can be executed faster than asymmetric encryption in general. Therefore, we focus on symmetric searchable encryption in this work.

An information flow of searchable encryption schemes is shown in Figure 1.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 235–252, 2012.

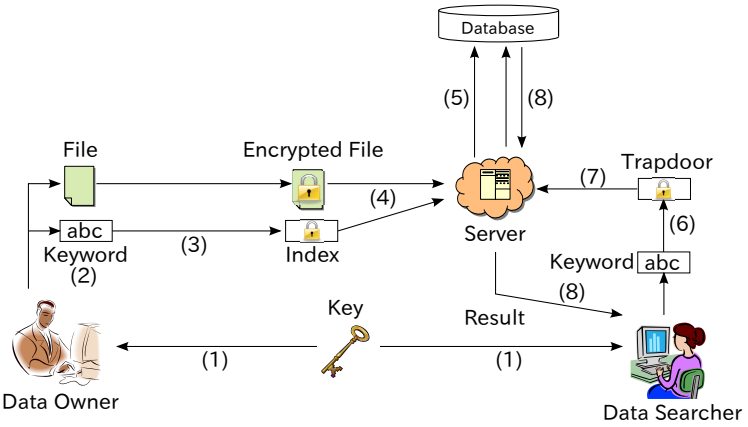


Fig. 1. Flow of Searchable Encryption

1. A data owner shares a key with data searchers before encryption.
2. The data owner specifies the keywords $\mathrm{Kw_A}$ which represent the contents of the file.
3. The data owner computes indexes $\mathrm{Idx_A}$ of $\mathrm{Kw_A}$. "Index" means a data structure which enables the server to perform an encrypted keyword search if and only if the server obtains a corresponding trapdoor that is explained later.
4. The data owner sends encrypted data C_A and $\mathrm{Idx_A}$ to a server.
5. The server registers C_A and $\mathrm{Idx_A}$ to the database.
6. The data searcher computes trapdoors $\mathrm{Td_B}$ of keywords $\mathrm{Kw_B}$. "Trapdoor" means a data structure which enables the data searcher to query the keyword the data searcher wants to search in the server without revealing it.
7. The data searcher sends $\mathrm{Td_B}$ to the server.
8. The server searches indexes in the database by using $\mathrm{Td_B}$ and returns the result to the data searcher.

$\mathrm{Idx_A}, \mathrm{Td_B}$ do not reveal $\mathrm{Kw_A}, \mathrm{Kw_B}$ to the server. Therefore, we can retrieve data without letting the server know the keywords.

The types of searches are as follows.

Equality search. The keywords which match the query completely hit.
Prefix search. The keywords which contain the query in the head hit.
Suffix search. The keywords which contain the query in the tail hit.
Partial matching search. The keywords which contain the query anywhere hit.
Wildcard search. The keywords which match any character other than wildcard characters hit. Wildcard characters represent any characters.
Fuzzy keyword search. The keywords within a certain edit distance from the query hit.

Full-text search. We search the full content of a document for specified several keywords.

Relational database search. We search a relational database for some values (keywords, numbers, and so on). We usually use a query language like SQL in this search.

1.2 Motivation

The searches for plaintexts such as search engines, searches in the computer, searches for e-mails in a mailbox are realized in various ways and very convenient. However, the available types of searches in searchable encryption schemes have been restricted because the server searches hidden plaintexts for hidden keywords. Therefore, our motivation is to enable to compare keywords in the trapdoor per character with the hidden keyword in the index to realize more advanced searches.

1.3 Related Works

Song et al. proposed the first practical searchable symmetric encryption scheme in 2000 [15]D This scheme supports only equality search and thus is not so convenient. After that, many searchable symmetric encryption schemes were proposed [2,6,7,17]. These schemes aim to enhance the situations where they are usable or to improve security and do not support any method other than equality search.

In recent years, Li et al. proposed the first searchable encryption scheme which supports a fuzzy keyword search [11]. This scheme is based on wildcard realized by enumerating keywords like "?apple", "?pple", ..., and "apple?" for "apple". However, we cannot use any other wildcard patterns than prepared patterns for fuzzy keyword search because this wildcard is realized by whole keyword match as equality search. Furthermore, we cannot search any keyword other than those which are enumerated by the data owner.

Boneh et al. proposed the first searchable asymmetric encryption scheme [4]. Abdalla et al. proposed anonymous IBE [1] and mentioned the relationship to searchable asymmetric encryption scheme. After that, many searchable asymmetric encryption schemes were proposed [5]. Sedghi et al. proposed a searchable asymmetric encryption scheme which supports wildcard search [14]. This scheme support a fixed wildcard search as our scheme and is realized with bilinear pairing. In symmetric key settings, we can execute symmetric key encryption schemes or hash functions faster than asymmetric key encryption schemes in general. However, there is no searchable symmetric encryption scheme which supports general wildcard searches.

Goh proposed a searchable symmetric encryption scheme for full-text search with Bloom filter [8] and Watanabe et al. proposed a searchable symmetric encryption scheme for relational database with Bloom filter [16]. These schemes use Bloom filter for efficiency.

1.4 Challenging Issues

After the first searchable encryption scheme was proposed, many searchable encryption schemes were proposed. As an example, consider the wildcard search like "2011/??/??" to find the dates from "2011/01/01" to "2011/12/31". These schemes make data searchers enumerate all possible keywords like "2011/01/01", "2011/01/02", ..., and "2011/12/31" or the data owner enumerates all possible patterns the data searcher may query. In this case, the data searcher has to enumerate 365 keywords because we know wildcard characters represent the dates. However, the data searcher has to enumerate more keywords when the data searcher cannot know what wildcard character can be because we have to enumerate all possible characters.

We can perform search by testing whether any character other than wildcard characters matches if we can compare not keywords but characters. Therefore, our challenging task is designing a searchable encryption which supports searches based on character comparison in a secure manner.

1.5 Our Contribution

In this work, we propose a searchable symmetric encryption scheme which supports searches based on the comparison per character. In this paper, we call this search position-specific keyword search.

This search brings us the following advantages:

- We can realize wildcard search efficiently. For example, in [11], we can perform a search for "2011/??/??" by comparing any character other than wildcard characters. In this example, we need 365 trapdoors with the traditional searchable symmetric encryption schemes but only one trapdoor with our scheme.
- We can also realize the wildcard-based fuzzy keyword search proposed by Li et al. [11] efficiently. Given keyword length ℓ and edit distance d, we can decrease index data size from $\mathcal{O}(\ell^d)$ to $\mathcal{O}(1)$.
- We can realize partial matching search with a few indexes because the data searcher can ignore characters other than specified characters. We can realize it with a single index, but we can realize it efficiently if we put the same number of indexes as the length of the specified keyword. For example, we can perform a partial matching search for "dog" by sending several trapdoors "dog", "?dog", ..., "?...?dog" when a single index "housedog" is put in the cloud servers. In this search, given the upper bound of the keyword length u and the keyword length ℓ, the number of trapdoors that must be sent to the cloud server is $u-\ell$. On the other hand, we can perform a partial matching search for "dog" by sending a single trapdoor if eight indexes "housedog", "ousedog", "usedog", ..., "g" are put in the cloud servers because the indexes contain "dog". Another example is the indexes for "doggy". In this example, we put five indexes "doggy", ..., "y", in the cloud server and we can also perform a partial matching search for "dog" by sending a single trapdoor because "doggy" matches "dog" by ignoring characters other than specified characters.

Table 1. Comparison of Search Functions

Name	Key	Search type
our scheme	symmetric	searches with specified characters
Song et al's scheme [15]	symmetric	equality search
Li et al.'s scheme [11]	symmetric	fuzzy keyword search
Z-IDX [8]	symmetric	full-text search
Watanabe et al.'s scheme [16]	symmetric	relational database search
PEKS [4]	asymmetric	equality search
Sedghi et al.'s scheme [14]	asymmetric	wildcard search

We can execute our scheme faster than asymmetric ones because we do not need complex computations like pairing used by asymmetric ones. Furthermore, we can perform searches efficiently without testing all indexes on the server side as explained in Section 5.1.

In our construction, we use pseudo-random functions for security improvement. The pseudo-random functions can conceal the information because no efficient algorithm can distinguish an output of pseudo-random functions from an output of random functions. We also use the Bloom filter to decrease the data size of the output of pseudo-random functions because pseudo-random functions needs some bits (e.g., 256 bits with HMAC-SHA256) for a single input, but one Bloom filter with the same length can contain multiple outputs of pseudo-random functions.

Note that Goh's scheme [8] and Watanabe et al.'s scheme [16] already use the Bloom filter only for an index, whereas we use the Bloom filter not only for an index but also for a trapdoor. We cannot apply this our technique to the Goh's original scheme and Watanabe et al.'s scheme, so the data size of the trapdoor cannot be decreased. Although we can apply this our technique to Goh's second scheme described in the appendix of [8], its applicability was not mentioned in [8] and the security proof of the second scheme was not given explicitly in [8].

1.6 Comparison with Existing Works

Let the keyword length be ℓ, and edit distance d for fuzzy keyword search.

We show the comparison of search functions in Table 1, and the comparison of data size and run time to generate an index and a trapdoor with Goh's scheme, Song et al.'s scheme and Li et al.'s scheme in Table 2. In Table 2, "single index" and "single trapdoor" are two methods described in Section 1.5. Note that although we show the orders of the index data size in Table 2 together, these orders have the different meanings because an index of Goh's scheme consists of multiple keywords, and an index of our scheme consists of a single keyword.

We can see the decrease in data size and run time compared with existing schemes in Table 2. In particular, we can decrease the index size in the fuzzy keyword search. Since our scheme is not designed for full-text search, Goh's scheme is more space- and computation-efficient than our scheme when we

Table 2. Efficiency Comparison

Type of search	Name	Data size		Run time	
		Index	Trapdoor	Index	Trapdoor
equality search	our scheme	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(\ell)$	$\mathcal{O}(\ell)$
	Song et al's scheme [15]	$\mathcal{O}(\ell)$	$\mathcal{O}(\ell)$	$\mathcal{O}(\ell)$	$\mathcal{O}(\ell)$
	Goh's scheme [8]	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(1)$
fuzzy keyword search	our scheme	$\mathcal{O}(1)$	$\mathcal{O}(\ell^d)$	$\mathcal{O}(\ell)$	$\mathcal{O}(\ell^{d+1})$
	Song et al's scheme [15]	—	—	—	—
	Li et al's scheme [11]	$\mathcal{O}(\ell^d)$	$\mathcal{O}(\ell^d)$	$\mathcal{O}(\ell^d)$	$\mathcal{O}(\ell^d)$
partial matching search	our scheme (single index)	$\mathcal{O}(1)$	$\mathcal{O}(u)$	$\mathcal{O}(\ell)$	$\mathcal{O}(u\ell)$
	our scheme (single trapdoor)	$\mathcal{O}(\ell)$	$\mathcal{O}(1)$	$\mathcal{O}(\ell^2)$	$\mathcal{O}(\ell)$
	Song et al's scheme [15]	—	—	—	—
	Li et al's scheme [11]	—	—	—	—

(ℓ: keyword length, d: edit distance, u: upper bound of keyword length, n: number of keywords in one file).

perform a full-text search. However, our scheme is more space-efficient than the existing schemes when we perform more complex search like a fuzzy keyword search. c

2 Preliminaries

Notations

We use the following notations in this paper.

Symmetric set difference. We define a symmetric set difference $A \triangle B$ as $A \triangle B = (A - B) \cup (B - A)$.

Random number. $x \xleftarrow{R} S$ denotes random variable x chosen at random from the set S.

The number of elements. We use $|A|$ to denote the number of elements in A

String concatenation. $a \parallel b$ denotes concatenation of two strings a, b.

Character. $w[n]$ denotes n-th character in string w.

Logical operations. Given two logical values a, b, $a \wedge b$ denotes the logical AND between a and b, and $a \vee b$ denotes the logical OR between a and b.

Pseudo-random Functions

A pseudo-random function is computationally indistinguishable from a random function. To be more specific, we call a function $f: \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^m$ which has the following features as (t, ϵ, q)-pseudo-random function.

- Given an input $x \in \{0,1\}^n$ and a key $sk \in \{0,1\}^s$, $f(x, sk)$ can be computed efficiently.
- For any t time oracle algorithm $\mathcal{A}$ with at most q adaptive queries,

$$|\Pr[\mathcal{A}^{f(\cdot,sk)} = 0 | sk \xleftarrow{R} \{0,1\}^s] - \Pr[\mathcal{A}^g = 0 | g \xleftarrow{R} \{F : \{0,1\}^n \to \{0,1\}^m\}]| < \epsilon.$$

In this paper, we use a keyed hash function like HMAC-SHA256 where the key sk is shared among a data owner and data searchers.

Symmetric Key Encryption

We use $\Pi = (Setup(1^\lambda), Enc(sk, \cdot), Dec(sk, \cdot))$ to denote a symmetric key encryption scheme. Given a security parameter λ, $Setup(1^\lambda)$ outputs a secret key. Given a secret key sk, $Enc(sk, \cdot)$ and $Dec(sk, \cdot)$ are an encryption and decryption scheme with sk.

Bloom Filter

Bloom filter [3] is space-efficient probabilistic data structure. Bloom filter has a false positive. It means that Bloom filter may output true even if it does not have the element to be checked. On the other hand, Bloom filter does not have a false negative. It means the element is guaranteed not to be in Bloom filter if Bloom filter outputs false.

Bloom filter is denoted as m-bit array with address from 1 to m. First, this bit array is initialized with 0. To add an element to Bloom filter, we compute k addresses $a_1, a_2, ..., a_k$ with k hash functions $h_1, h_2, ..., h_k$ first. Then, we make the bits corresponding to the addresses be 1. To determine if Bloom filter has the element, we also compute k addresses $a'_1, a'_2, ..., a'_k$. Then, we check if all bits corresponding to the addresses are 1. We can know with certain error rate that the element is in the Bloom filter if and only if all bits are 1. We can also without error know the element is not in Bloom filter if some bits are 0.

Search Expression

In this paper, we express a position-specific keyword search as a DNF logical formula $p = (p_{(1,1)} \wedge ... \wedge p_{(1,m_1)}) \vee ... \vee (p_{(n,1)} \wedge ... \wedge p_{(n,m_n)})$. In this formula, $p_{(i,j)} (i \in [1,n], j \in [1,m_i])$ denotes the logical formula which compares the character in the specified position, and let $w[x] = c$ denote a comparison which checks if x-th character of w is c.

We call this formula p as search expression in this paper.

For example, the search expression p to perform an equality search for either keyword "dog" or "cat" is as follows:

$$\begin{aligned} f = &((w[1] = \text{“d”}) \wedge (w[2] = \text{“o”}) \wedge (w[3] = \text{“g”}) \wedge (w[4] = null)) \\ &\vee ((w[1] = \text{“c”}) \wedge (w[2] = \text{“a”}) \wedge (w[3] = \text{“t”}) \wedge (w[4] = null)) \end{aligned}$$

File Identifier

File identifier means the unique name of a file. We can use an absolute path for the file, a universal unique identifier (UUID) [10], and so on. We can use any type of file identifier.

3 Proposed Scheme

We use one Bloom filter to store all characters of one keyword. We also use pseudo-random functions when we add a character to Bloom filter and symmetric key encryption scheme to encrypt data for the search results. We can use any secure symmetric key encryption scheme.

We have to specify an upper bound u of the keyword length before use.

Note that each keyword is terminated with a *null* which denotes a null symbol. *null* denotes the end of the keyword and is used for the query including the end of the keyword.

This scheme has the following four algorithms.

KeyGen(1^λ) Given a security parameter λ, output a secret key $sk \xleftarrow{R} \{0,1\}^\lambda$.

Trapdoor(sk, p) Given a secret key sk and a search expression p, output a trapdoor for p. Let $p = p_1 \vee ... \vee p_n$, $p_i = (p_{(i,1)} \wedge ... \wedge p_{(i,m_i)})$, where $p_{(i,j)}$ denotes a comparison which checks if $w[x_{(i,j)}] = c_{(i,j)}$. We use $T = \{T_1, ..., T_n\}$ to denote a trapdoor for p. We can compute T_i for $i \in [1, n]$ as follows.

1. Initialize a Bloom filter T_i.
2. For each term $p_{(i,j)}$ for $j \in [1, m_i]$, given $p_{(i,j)}$ denotes a comparison which checks if $w[k] = c$, add a concatenated string $k \parallel c$ to the Bloom filter T_i.

BuildIndex(sk, FID_w, w) Given a secret key sk, a file identifier FID_wCand a keyword w, compute an index $\mathcal{I} = \{\mathcal{I}_I, \mathcal{I}_{II}\}$ for w as follows.

1. Initialize a Bloom filter $\mathcal{I}_\mathrm{I}$.
2. For each character $w[i]$ for $i \in [1, |w|]$, add a concatenated string $i \parallel w[i]$ to $\mathcal{I}_\mathrm{I}$.
3. Let the number of pseudo-random functions used for the Bloom filter $\mathcal{I}_\mathrm{I}$ be k. Pick $(u - |w|) \cdot k$ random values and set the respective bits of the Bloom filter to 1. This operation is equivalent to an insertion of $u - |w|$ random characters, where $u - |w|$ is the difference between the actual length and the upper bound, into the Bloom filter. We have this operation to prevent the number of 1's in $\mathcal{I}_\mathrm{I}$ from revealing the length of the keyword w.
4. Choose a random value $rd \xleftarrow{R} \{0,1\}^\lambda$. This value is used to randomize $\mathcal{I}_\mathrm{II}$.
5. Encrypt a collection of a file identifier FID_w, a keyword w, and rd with a symmetric key encryption scheme as $\mathcal{I}_\mathrm{II} = \mathrm{Enc}(sk, \mathrm{FID}_w \parallel w \parallel rd)$. This value is used to query the file and to check if the search result is correct. rd randomizes $\mathcal{I}_\mathrm{II}$ to hide the equality of the keyword.
6. Output the index $\mathcal{I} = \{\mathcal{I}_\mathrm{I}, \mathcal{I}_\mathrm{II}\}$.

This algorithm is executed as many times as the number of keywords.

SearchIndex($\mathrm{T}, \mathcal{I}$) Given a trapdoor T for a certain keyword and an index $\mathcal{I}$, search the indexes in which all bits set to 1 in trapdoor are 1 and output the search result $\mathcal{I}_\mathrm{II}$ in the index $\mathcal{I}$.

We describe the information flow of this scheme as follows.

1. Those who share files share a key generated by KeyGen before using this scheme.
2. A data owner encrypts a file with Enc. The data owner specifies keywords and computes indexes of them with BuildIndex.
3. The data owner sends the encrypted file and index.
4. A server stores them in the database.
5. A data searcher computes a trapdoor for a specified keyword with Trapdoor.
6. The data searcher sends it to the server.
7. The server searches with SearchIndex and returns the result.

We describe the concrete examples of the outputs of BuildIndex and Trapdoor in Appendix B that will help understand our scheme intuitively.

Optimization

We can use a linear search to execute SearchIndex. However, we can achieve more efficient search with a binary tree search because we do not have to test all indexes in a binary tree search. See Appendix C for the details.

Determining Suitable Parameters for Bloom Filter

We have to determine the length of Bloom filter m and the number of pseudo-random functions k before use. Given the maximum number of characters n and the acceptable possibility of false-positive f_p, we can determine these parameters as follows [8]:

$$k = -\log_2 f_p, m = \frac{kn}{\ln 2} \tag{1}$$

4 Security Analysis

4.1 Security Model

The security model we use is based on IND-CKA [8]. We define the indistinguishability of keywords. Although Z-IDX [8] creates indexes from the set of keywords, our scheme creates indexes from one keyword. We call this security model IND-CPSKA[1] to make the difference clear.

This security model is defined by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows. We say that an adversary $\mathcal{A}$ (t, ϵ, q)-breaks our scheme if $\mathrm{Adv}_{\mathcal{A}}$ is at least ϵ after $\mathcal{A}$ takes at most t time and query trapdoors to $\mathcal{C}$ q times. We say that the symmetric searchable encryption $\mathcal{I}$ is (t, ϵ, q)-IND-CPSKA secure if there is no adversary who can (t, ϵ, q)-break $\mathcal{I}$.

[1] IND-CPSKA denotes Indistinguishability under Chosen Position-Specific Keyword Attack.

Setup. $\mathcal{C}$ creates a set S of q pairs of position and character, and gives this to $\mathcal{A}$ D $\mathcal{A}$ chooses some subsets S^*, that is, keywords from S and gives this to $\mathcal{C}$. After receiving S^*, $\mathcal{C}$ runs KeyGen to generate a secret key K_{priv}, and $\mathcal{C}$ computes the indexes for all subsets of S^* with BuildIndex. Finally, $\mathcal{C}$ gives all indexes and related subsets S^* to $\mathcal{A}$ after computing all indexes. We note that the correspondence relation between the indexes and S^* is unknown to $\mathcal{A}$.

Query. $\mathcal{A}$ can query trapdoor T_x for word x to $\mathcal{C}$. For each index $\mathcal{I}$, $\mathcal{A}$ can execute SearchIndex for $T_x, \mathcal{I}$ to tell whether $\mathcal{I}$ matches x.

Challenge. $\mathcal{A}$ picks nonempty two subsets $V_0, V_1 \in S^*$ such that $|V_0 - V_1| \neq 0$, $|V_1 - V_0| \neq 0$ and $|V_0| = |V_1|$. Here, $\mathcal{A}$ must not have queried $\mathcal{C}$ for the trapdoor of any character in $V_0 \triangle V_1$. $\mathcal{A}$ cannot query any trapdoor for a character in $V_0 \triangle V_1$. $\mathcal{A}$ gives V_0 and V_1 to $\mathcal{C}$. $\mathcal{C}$ chooses b from $\{0,1\}$ at random. $\mathcal{C}$ computes BuildIndex(V_b, K_{priv}) to get an index corresponding to V_b and gives it back to $\mathcal{A}$. After $\mathcal{C}$ gives the challenge (i.e., BuildIndex(V_b, K_{priv})) to $\mathcal{A}$, $\mathcal{A}$ cannot query any trapdoor for any character $x \in V_0 \triangle V_1$ to $\mathcal{C}$.

Response. $\mathcal{A}$ outputs b' to guess b. The advantage $\mathcal{A}$ obtains is defined as $\mathrm{Adv}_{\mathcal{A}} = |Pr[b = b'] - 1/2|$.

4.2 Security Proof

Theorem 1 *Given the number of pseudo-random functions k, our scheme is $(t, \epsilon, q/k)$-IND-CPSKA secure if f is a (t, ϵ, q)-pseudo-random function.*

Proof. We can prove this theorem as the proof in [8].

We prove this theorem using its contrapositive. Suppose our scheme is not $(t, \epsilon, q/k)$-IND-CPSKA secure, that is, there is an algorithm $\mathcal{A}$ which $(t, \epsilon, q/k)$-breaks our scheme. Then we show we can construct the algorithm $\mathcal{B}$ which distinguishes whether f is a pseudo-random function or a random function. Given $x \in \{0,1\}^n$, $\mathcal{B}$ can use an oracle $\mathcal{O}_f$ which outputs $f(x) \in \{0,1\}^s$ for unknown function f. $\mathcal{B}$ evaluates f with a query to $\mathcal{O}_f$ whenever computing any four index algorithms.

The algorithm $\mathcal{B}$ makes the simulation for $\mathcal{A}$ as follows.

Setup. $\mathcal{B}$ chooses a set S of q/k pairs of position and character from $\{0,1\}^n$ at random and sends it to $\mathcal{A}$. $\mathcal{A}$ returns a collection S^* of polynomial numbers of subsets. For each subset D of S^*, $\mathcal{B}$ assigns a file identifier FID_D and gets $\mathcal{I}_{\mathrm{FID}_D}$ by computing BuildIndex for FID_D. $\mathcal{B}$ gives all indexes and related subsets S^* to $\mathcal{A}$ after computing all indexes. We note that correspondence relation between the indexes and S^* is unknown to $\mathcal{A}$.

Query. $\mathcal{B}$ computes Trapdoor for x and returns a trapdoor T_x for x.

Challenge. $\mathcal{A}$ picks nonempty subset $V_0, V_1 \in S^*$ such that $|V_0 - V_1| \neq 0$, $|V_1 - V_0| \neq 0$ and $|V_0| = |V_1|$. $\mathcal{A}$ cannot query any trapdoor for a character in $V_0 \triangle V_1$ to $\mathcal{B}$. $\mathcal{A}$ gives V_0 and V_1 to $\mathcal{B}$. $\mathcal{B}$ chooses b from $\{0,1\}$ and a file identifier V_{id} at random. $\mathcal{B}$ computes BuildIndex to get $\mathcal{I}_{V_b}$ and gives $\mathcal{I}_{V_b}$ to $\mathcal{A}$. The challenge to $\mathcal{A}$ is to guess b. $\mathcal{A}$ cannot query any trapdoor which contains any character $x \in V_0 \triangle V_1$.

Response. $\mathcal{A}$ outputs b' finally. $\mathcal{B}$ outputs 0 if $b = b'$, that is, f is a pseudo-random function. Otherwise, $\mathcal{B}$ outputs 1.

$\mathcal{B}$ takes at most t time because $\mathcal{A}$ takes at most t time. $\mathcal{B}$ sends at most q queries to $\mathcal{O}_f$ because there are only q/k characters, $\mathcal{A}$ creates at most q/k queries, and $\mathcal{B}$ creates k queries for $\mathcal{A}$'s single query.

Finally, according to the following lemmas, $\mathcal{B}$ has advantage greater than ϵ to determines if the unknown function f is a pseudo-random function or f is a random-function because we have the following equation:

$$|\Pr[\mathcal{B}^{f(\cdot,k)} = 0 | k \xleftarrow{R} \{0,1\}^s] - \Pr[\mathcal{B}^g = 0 | g \xleftarrow{R} \{F : \{0,1\}^n \to \{0,1\}^s\}]| \geq \epsilon$$

This contradicts the assumption of pseudo-random functions.

Therefore, our scheme is $(t, \epsilon, q/k)$-IND-CPSKA secure if f is a (t, ϵ, q)-pseudo random function.

Lemma 1. $|\Pr[\mathcal{B}^{f(\cdot,k)} = 0 | k \xleftarrow{R} \{0,1\}^s] - \frac{1}{2}]| \geq \epsilon$ *if* f *is a pseudo-random function.*

Lemma 2. $\Pr[\mathcal{B}^g = 0 | g \xleftarrow{R} \{F : \{0,1\}^n \to \{0,1\}^s\}] = \frac{1}{2}$ *if* g *is a random function.*

Proof. Lemma 1 is obvious because $\mathcal{B}$ simulates $\mathcal{C}$ completely in an IND-CPSKA game if f is a pseudo-random function.

We prove Lemma 2. We have to consider only Challenge subsets V_0, V_1 because other subsets in S do not reveal any information about the Challenge subsets.

Without loss of generality, assume that $V_0 \triangle V_1$ has two characters x, y such that $x \in V_0, y \in V_1$ and $\mathcal{A}$ guesses b with advantage δ. Given $f(z)$, it means that $\mathcal{A}$ can determine if $z = x$ or $z = y$ with advantage δ, that is, $\mathcal{A}$ can distinguish the output of a random function f with advantage δ. However, if f is a random function, $\mathcal{A}$ cannot distinguish the output, so we have $\delta = 0$. Therefore, $\mathcal{A}$ can guess b with the probability of at best 1/2. Finally, we proved Lemma 2. □

4.3 Limitation

In this scheme, the trapdoor is divided into the clauses. Therefore, the server can know the search result of each clause. For example, in an equality search for "dog" or "cat", the server cannot know the plaintext "dog" and "cat", but the server can know the search result for "dog" and the search result for "cat" respectively.

This limitation exists in many of the existing works as well as this work. For example, Goh's scheme [8] generates a trapdoor per keyword and the server can know the search result for each keyword. Similarly, Li et al.'s scheme [11] generates a trapdoor set per keyword and the server can know the search result for not only each keyword but also each clause.

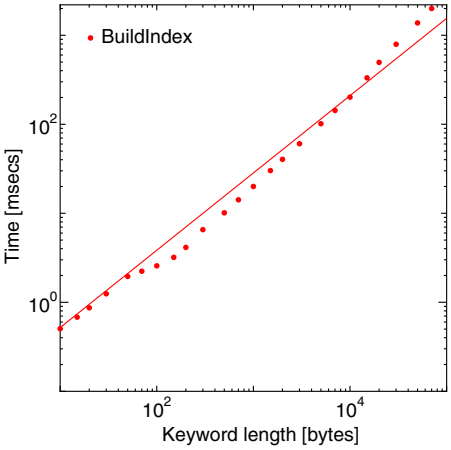


Fig. 2. Run Time for BuildIndex

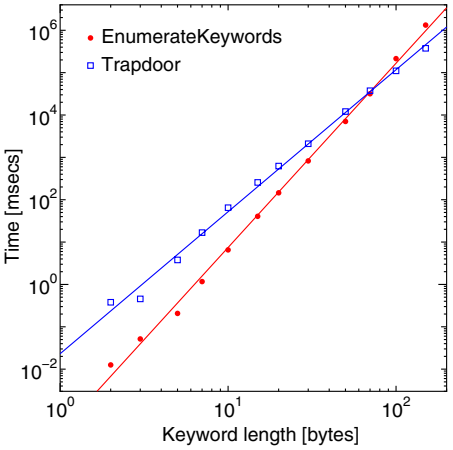


Fig. 3. Run Time for Search

5 Evaluation

Given a bit length of Bloom filter m, an index consists of m-bit Bloom filter and $\mathcal{I}_{\mathrm{II}}$ in $\mathcal{I}$. Given the number of terms divided by disjunctions in the search expression n_ℓ, the size of trapdoor is $n_\ell m$ bits because it has n_ℓ Bloom filters. Given the total number of characters in all keywords ℓ_m, the execution of BuildIndex needs $\mathcal{O}(\ell_m)$ time because we have to compute $\mathcal{O}(\ell_m)$ pseudo-random functions. Similarly, given the number of terms in search expression n_t, the execution of Trapdoor needs $\mathcal{O}(n_t)$ time.

5.1 Implementation

We implemented our scheme with fuzzy keyword search [11] on a 2.8 GHz Intel Core 2 Duo CPU. We used 256-bit Bloom filter, symmetric key encryption scheme AES [12], and keyed hash function HMAC-SHA256 [9,13]. HMAC-SHA256 can be used as distinct pseudo-random functions as $f(sk, x)$, $f_i(sk, x) = f(sk, i \parallel x)$.

The run time for BuildIndex is shown in Figure 2 and the run time for Trapdoor is shown in Figure 3. *EnumerateKeywords* is an algorithm to enumerate keywords proposed by Li et al. We can see in the figure that the larger the keyword length becomes, the more time we need, and our scheme is unsuitable for very long keyword. However, it is not a practical problem because a keyword does not often have a large number of characters.

We implemented *SearchIndex* in the following two ways.

Method 1. This method is based on complete search with a relational database SQLite. The server stores indexes divided into 64 bits. Trapdoor is also divided into 64 bits. Given divided indexes $\mathrm{Idx}_1, ..., \mathrm{Idx}_n$ and trapdoors

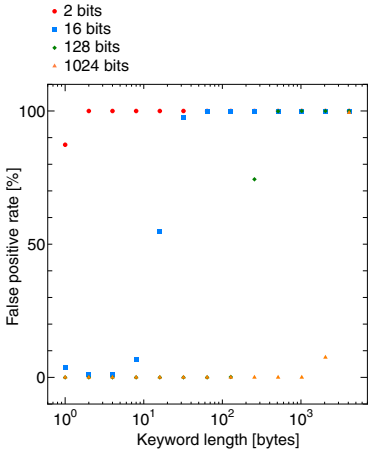


Fig. 4. False-Positive Rate

$\mathrm{Td}_1, ..., \mathrm{Td}_n$, a query for the trapdoor can be constructed as $((\mathrm{Idx}_1 \& \mathrm{Td}_1) = \mathrm{Td}_1) \wedge ... \wedge ((\mathrm{Idx}_1 \& \mathrm{Td}_n) = \mathrm{Td}_n)$. We divide indexes and trapdoors because the maximum length SQLite can compute AND operation is 64 bits. This is an implementation problem. We do not have to divide indexes and trapdoors if the database supports AND operation of larger bits.

Method 2. This method is based on binary tree search. Each bit of indexes corresponds to a link of the tree. The leaf node has the search result. In this method, we construct a binary tree from the indexes generated with the same key. This search is realized by following the link recursively. We can ignore the link of 0 corresponding to the bit of 0. See Appendix C for the details. Therefore, we do not have to go through the whole tree, and we can achieve more efficient search than the sequential exhaustive search.

A fuzzy keyword search with 6-byte keyword on the database which has 1,000,000 keywords takes 44 seconds on average when we use Method 1. However, it takes only 280 milliseconds on average when we use Method 2.

5.2 Bloom Filter Parameters v.s. False-Positive Rate

We performed an experiment to clarify how the Bloom filter parameters affect the false-positive rate. In this experiment, we used HMAC-SHA256 [9,13], and the number of pseudo-random functions is 3 (fixed).

We performed this experiment as follows. First, we generate a random keyword kw_{idx} and generate an index $\mathrm{Idx}_{kw_{idx}}$ for kw_{idx}. Next, we generate another random keyword kw_{td} ($kw_{idx} \neq kw_{td}$) and generate a trapdoor $\mathrm{Td}_{kw_{td}}$. Finally, we check if $\mathrm{Idx}_{kw_{idx}}$ matches $\mathrm{Td}_{kw_{td}}$. If $\mathrm{Idx}_{kw_{idx}}$ matches $\mathrm{Td}_{kw_{td}}$, this result is a false-positive because $kw_{idx} \neq kw_{td}$.

We show the result of this experiment in Figure 4. In this figure, the x-axis is a keyword length of kw_{idx} and kw_{td}, the y-axis is a false-positive rate, and the graph legends are the bit lengths of the Bloom filter.

This figure shows that an m-bit Bloom filter is effective with small false-positive rates up to m-byte keyword.

6 Conclusion

In this work, we proposed a searchable symmetric encryption scheme which supports a variety of searches by enabling comparison per character as searches for plaintexts. We implemented our scheme and we confirmed that our scheme can be performed on both of client and server in practical time.

Our future work is to find a scheme which creates a single index for the multiple keywords or single trapdoor for multiple terms divided by the disjunctions to decrease data size, run time or information obtained by the server.

Acknowledgments. This work is partially supported by Grant-in-Aid for Young Scientists (B) (23700021), Japan Society for the Promotion of Science (JSPS). This work is also partially supported by the Telecommunications Advancement Foundation (TAF).

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. Journal of Cryptology 21, 350–391 (2008)
2. Bao, F., Deng, R.H., Ding, X., Yang, Y.: Private Query on Encrypted Data in Multi-user Settings. In: Chen, L., Mu, Y., Susilo, W. (eds.) ISPEC 2008. LNCS, vol. 4991, pp. 71–85. Springer, Heidelberg (2008)
3. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Commun. ACM 13, 422–426 (1970)
4. 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)
5. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Vadhan, S. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
6. Chang, Y.-C., Mitzenmacher, M.: Privacy Preserving Keyword Searches on Remote Encrypted Data. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 442–455. Springer, Heidelberg (2005)
7. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, pp. 79–88. ACM, New York (2006)
8. Goh, E.J.: Secure indexes. Cryptology ePrint Archive, Report 2003/216 (2003), http://eprint.iacr.org/2003/216/

9. Krawczyk, H., Bellare, M., Canetti, R.: HMAC: Keyed-Hashing for Message Authentication. RFC 2104 (Informational) (February 1997)
10. Leach, P., Mealling, M., Salz, R.: RFC 4122: A universally unique identifier (UUID) URN namespace (2005)
11. Li, J., Wang, Q., Wang, C., Cao, N., Ren, K., Lou, W.: Fuzzy keyword search over encrypted data in cloud computing. In: 2010 Proceedings of the IEEE INFOCOM, pp. 1–5 (March 2010)
12. NIST: Announcing the advanced encryption standards (AES). Federal Information Processing Standards Publication 197 (2001)
13. NIST: Announcing the secure hash standard. Federal Information Processing Standards Publication 180-2 (2002)
14. Sedghi, S., van Liesdonk, P., Nikova, S., Hartel, P., Jonker, W.: Searching Keywords with Wildcards on Encrypted Data. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 138–153. Springer, Heidelberg (2010)
15. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: Proceedings of the IEEE Symposium on Security and Privacy, S&P 2000, pp. 44–55 (2000)
16. Watanabe, C., Arai, Y.: Privacy-Preserving Queries for a DAS Model Using Encrypted Bloom Filter. In: Zhou, X., Yokota, H., Deng, K., Liu, Q. (eds.) DASFAA 2009. LNCS, vol. 5463, pp. 491–495. Springer, Heidelberg (2009)
17. Waters, B.R., Balfanz, D., Durfee, G., Smetters, D.K.: Building an encrypted and searchable audit log. In: The 11th Annual Network and Distributed System Security Symposium (2004)

A Pseudocode

We show the algorithms of our scheme with pseudocodes.

In these algorithms, we use m to denote the bit length of a Bloom filter and k to denote the number of pseudo-random functions.

In Algorithm 2, the search expression se is denoted as $((x_{(1,1)}, c_{(1,1)}), ..., (x_{(1,m_1)}, c_{(1,m_1)})), ..., ((x_{(n,1)}, c_{(n,1)}), ..., (x_{(n,m_n)}, c_{(n,m_n)}))$ in order to denote it as an array. For example, $((1, d), (2, o), (3, g), (4, null))$ denotes an equality search for "dog" only in this section. $|se|$ denotes the number of elements of the array se.

B Execution Example of Proposed Scheme

We describe the concrete examples of the outputs of BuildIndex and Trapdoor in this section. Suppose that Alice outsources data and Bob searches data. Alice specifies "dog" to denote the content of the file. Bob searches "dog" by an equality search. The bit length of the Bloom filters is 16 bits, the number of pseudo-random functions is 2, and the upper bound of the keyword length is 5. Suppose that they share these parameters.

1. First, Alice executes KeyGen and sends sk to Bob in a secure manner.
2. Alice executes BuildIndex as follows:

Algorithm 1. BuildIndex

Require: w is a keyword to build an index.
Ensure: (BF, c) is an index for w.
  Initialize a bit array BF with zeros.
  for $i = 1$ **to** $|w|$ **do**
    for $j = 1$ **to** k **do**
      $p \leftarrow f_j(i \parallel w[i])$.
      $BF[p] \leftarrow 1$.
    end for
    for $j = 1$ **to** $(u - |w|) \cdot k$ **do**
      Pick $rd \in [1, |BF|]$ at random.
      $BF[rd] \leftarrow 1$.
    end for
  end for
  Encrypt $FID_w \parallel w \parallel rd$ and assign it to c.
  Return (BF, c).

Algorithm 2. Trapdoor

Require: se is a search expression to generate a trapdoor.
Ensure: t is a trapdoor for se.
  $t \leftarrow \{\}$.
  for $i = 1$ **to** $|se|$ **do**
    Initialize a bit array BF_i with zeros.
    for $j = 1$ **to** $|se[i]|$ **do**
      for $h = 1$ **to** k **do**
        $p \leftarrow f_h(se[i][j][1] \parallel se[i][j][2])$.
        $BF_i[p] \leftarrow 1$.
      end for
    end for
    Append BF_i to t.
  end for
  Return t.

Algorithm 3. SearchIndex

Require: td is a trapdoor and idx is an index.
Ensure: Output 'true' if idx matches $trapdoor$, otherwise return 'false.'
  for $i = 1$ **to** m **do**
    if $td[i] = 1$ **and** $idx[i] = 0$ **then**
      Return 'false.'
    end if
  end for
  Return 'true.'

(a) Alice creates a 16-bit Bloom filter. The Bloom filter is 0000000000000000 at this point.

(b) Alice computes two pseudo-random functions for the respective characters of "1 ‖ d", "2 ‖ o", "3 ‖ g"C"4 ‖ *null*". Suppose that Alice obtains 6, 2 for "1 ‖ d", 12, 4 for "2 ‖ o", 5, 13 for "‖ g" and 2, 9 for "4 ‖ *null*". Alice sets the respective bits to 1. The resultant Bloom filter is 0101110010011000 at this point. Alice picks two random values because $(u - |w|) \cdot k = (5 - 4) \cdot 2 = 2$. Suppose that Alice obtains 15 and 4. Alice sets the respective bits to 1. The final Bloom filter is 0101110010011010 at this point. This value is $\mathcal{I}_{\mathrm{I}}$.

(c) Alice picks random value rd and computes $\mathcal{I}_{\mathrm{II}} = \mathrm{Enc}(sk, \mathrm{FID}_w \parallel w \parallel rd)$.

3. Alice sends $\mathcal{I} = (\mathcal{I}_{\mathrm{I}}, \mathcal{I}_{\mathrm{II}})$ to the server.
4. Bob executes Trapdoor to perform an equality search. The search expression is $(w[1] =$ "d"$) \wedge (w[2] =$ "o"$) \wedge (w[3] =$ "g"$) \wedge (w[4] = null)$. This search expression means that the searched keyword equals to "dog" including the terminal symbol. Bob computes two pseudo-random functions for the respective characters of "1 ‖ d", "2 ‖ o", "3 ‖ g"C"4 ‖ *null*". These values are the same as those Alice computed. Bob sets the respective bits to 1. The Bloom filter is 0101110010011000 at this point.
5. Bob sends this value to the server.
6. The server searches the indexes in which all bits set to 1 in trapdoor are 1 and returns the search result $\mathcal{I}_{\mathrm{II}}$ in the index $\mathcal{I}$ to Bob. This was generated by Alice in Step 2.
7. Bob decrypts $\mathcal{I}_{\mathrm{II}}$ and confirms whether the original keyword is "dog".
8. Bob queries the file whose identifier is FID_w.
9. The server returns the queried file.

The trapdoor in the example contains the termination character. Therefore, even if Alice registers "doggy", "doggy" does not hit and only "dog" hits because $w[4] \neq null$. Therefore, Bob can perform an exact equality search.

In this example, although we achieve an equality search, we can achieve a wildcard search for "d?g" by computing pseudo-random functions of "1 ‖ d", "3 ‖ g", "4 ‖ *null*" in Step 4. On the other hand, we can achieve a fuzzy keyword search by an enumeration proposed by Li et al. [11]. When we perform a fuzzy keyword search for "dog" with an edit distance 1, the data searcher Bob enumerates "?dog", "?og", "d?g", "do?", "dog?", and Bob executes Trapdoor for each search expression. When we perform a fuzzy keyword search with longer edit distance, we increase the number of wildcard characters like "??dog". In Li et al.'s scheme, the user who computes an index also has to enumerate possible keywords and prepare corresponding indexes. However, our scheme does not require the data owner to enumerate the keywords for a fuzzy keyword search when the data owner computes an index.

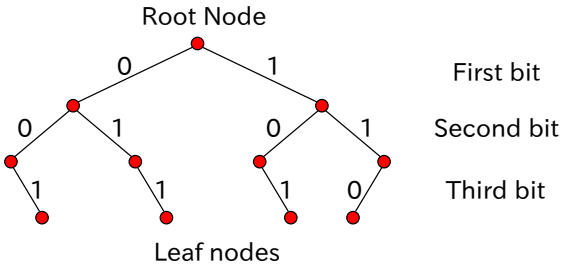


Fig. 5. Binary Search Tree

C Example of Binary Tree Search

We show an example of the binary search tree in Fig. 5. In this example, the tree has four indexes 001, 011, 101 and 110 (i.e., the length of a Bloom filter is three here). The leaf node has the search result.

Suppose that the server receives a trapdoor 101. The server can perform a search as follows:

1. From the root node, follow the link of 1.
2. Follow the link of 0.
3. Follow the link of 1.
4. Obtain the search result from the leaf node corresponding to 101.
5. Go back to the node of Step 2, and follow the link of 1 because the second bit of the trapdoor 101 is 0.
6. Finish the search since there is no more link to follow.

Fully Secure Doubly-Spatial Encryption under Simple Assumptions

Cheng Chen, Zhenfeng Zhang, and Dengguo Feng

Institute of Software, Chinese Academy of Sciences, Beijing, China
{chencheng,zfzhang,feng}@is.iscas.ac.cn

Abstract. Recently, Hamburg improved the notion of spatial encryption by presenting a variant called doubly-spatial encryption. As a generalization of the spatial encryption, the doubly-spatial encryption is more powerful and expressive. More useful cryptography systems can be built from it, such as attribute-based encryption, etc. However, the only presented doubly-spatial encryption scheme can only be proved to be selectively secure.

In this paper, we primarily focus on the full security of doubly-spatial encryption. A doubly-spatial encryption scheme has been proposed. We apply the dual system methodology proposed by Waters in the security proof. Our scheme can be proved adaptively secure under standard assumptions, the decisional linear assumption and the decisional bilinear Diffe-Hellman assumption, over prime order groups in the standard model. Our scheme is the first fully secure construction of doubly-spatial encryption. As an independent interest, we also propose a fully secure spatial encryption with weak anonymity and constant ciphertext size in the composite order group settings.

1 Introduction

In 2008, Boneh and Hamburg [1] proposed a general framework for constructing identity-based and broadcast crypto systems, which they called Generalized Identity-Based Encryption (GIBE). As an important instance of the GIBE framework, the spatial encryption scheme can be used to construct a host of efficient IBE-like schemes: multicast IBE, broadcast hierarchical IBE, predicate encryption, multiple authorities IBE and so on. In a spatial encryption scheme, encryption policies are vectors in an affine space $\mathbb{Z}_p^n$ and roles of the secret key are affine subspaces. The decryption succeeds when the affine subspace of the secret key contains the vector of a encryption policy. The delegation relation $\succeq$ on roles is defined by subspace inclusion.

Recently, Hamburg [7] first proposed the notion of doubly-spatial encryption. Doubly-spatial encryption is a generalization of spatial encryption, but it is more expressive than spatial encryption. In the doubly-spatial encryption scheme, the encryption policies are affine subspaces in $\mathbb{Z}_p^n$ instead of vectors. And the secret key can decrypt the ciphertext if its affine subspace intersects with the affine subspace of the ciphertext. As mentioned in [7], many useful crypto systems

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 253–263, 2012.

can be implemented by doubly-spatial encryption scheme, such as attribute-based encryption [13,6], threshold-based encryption [5], all-but-one signatures [7], etc. Spatial encryption can be embedded in doubly-spatial encryption. But the construction of doubly-spatial encryption is not so efficient as spatial encryption, since the length of the ciphertext is not constant. The only construction of doubly-spatial encryption [7] is proven to be selectively secure under some unnatural assumptions.

Related Work. Boneh and Hamburg [1] proposed the first selectively secure spatial encryption. The first fully secure spatial encryption scheme was proposed by [11]. The construction [11] was based on the three composite order bilinear groups and proven fully secure under three non-standard assumptions over composite order bilinear groups. Recently, Hamburg [7] proposed an adaptively secure scheme based on some static assumptions over prime order groups, but the assumptions are still non-standard. [7] also proposed the first doubly-spatial encryption with selective security. Chen et al. [3] obtained a fully secure spatial encryption scheme under *DBDH* and *DLIN* assumptions in prime order groups. But how to construct fully secure doubly-spatial encryption schemes using natural assumptions is still a problem that needs to solve.

Waters [14] introduced the dual system encryption to overcome the limitations of partitioning. And later, dual system encryption used in [8,12,10,9] to obtain adaptive security for IBE, HIBE, and ABE systems.

Our Contributions. In this paper, we construct first fully secure doubly-spatial encryption scheme. And we use dual system encryption technique introduced by Waters [14] in the proof. The security of our scheme depends on neither some non-standard assumptions [7] nor the assumptions over composite order pairing groups [11], but two standard assumptions: DLIN and DBDH, in the standard model. Our scheme is the first fully secure construction. This paper solves the problem brought forward by Hamburg in [7]. Finally, we also propose a fully secure spatial encryption with weak anonymity in the composite order group which is a product of four primes such as $\mathbb{G} = \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3} \times \mathbb{G}_{p_4}$. The weak anonymity is proposed by [7] where the decryption algorithm is required to take the policy of the ciphertext (vectors in spatial encryption) as input. We will give the definition of the weak anonymity in **Sec.** 3.2 The scheme can also achieve constant-size ciphertext.

Our Techniques. Our doubly-spatial encryption scheme is based on Waters' tag-based IBE [14]. In the construction, we extend the "two-equation revocation" technique of [9] to "n-equation revocation". We create each ciphertext with a uniformly distributed tag and each secret key with a uniformly distributed tag, too. The decryption algorithm will not work if the tag of the secret key and the tag of ciphertext has some relations. While in the actual simulation from normal secret key to semi-functional secret key, the tags created by simulator are linear dependent. The simulator can create the semi-functional secret keys of all affine spaces in the vector space. All the semi-functional secret keys can not decrypt the challenged ciphertext, even if the affine spaces of the semi-functional

secret keys contain the challenged vector due to the setting of tags. With the linear relationship of the tags, the simulation can process successfully. But this relationship is information theoretically hidden to the adversary.

Our second scheme is based on fully secure anonymous HIBE [2] in the composite order group setting. In the construction, we require a composite order group whose order is a product of four primes and three assumptions based on the group which are proposed in [2]. Intuitively, the reason the scheme can achieve weak anonymity is that the vector of the ciphertext is blinded by some random elements in $\mathbb{G}_{p_4}$.

2 Preliminaries

2.1 Notations and Affine Space

For any vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{Z}_p^n$, and any element g of a group $\mathbb{G}$, $g^{\boldsymbol{x}}$ stands for the vector of group elements $(g^{x_1}, \ldots, g^{x_n}) \in \mathbb{G}^n$. For $\boldsymbol{x}, \boldsymbol{v} \in \mathbb{Z}_p$, we denote their inner product as $\langle \boldsymbol{x}, \boldsymbol{v} \rangle := \sum_{i=1}^{n} x_i v_i$. Given $g^{\boldsymbol{x}}$ and $\boldsymbol{v}$, $(g^{\boldsymbol{x}})^{\boldsymbol{v}} := g^{\langle \boldsymbol{x}, \boldsymbol{v} \rangle}$ is computable without knowing $\boldsymbol{x}$. $\boldsymbol{x} \cdot \boldsymbol{v}$ stands for their component-wise product, and we denote $g^{\boldsymbol{x}} \cdot g^{\boldsymbol{v}} := g^{\boldsymbol{x} \cdot \boldsymbol{v}}$.

For any vector $\boldsymbol{x} \in \mathbb{Z}_p^n$ and any matrix $\mathbf{M} \in \mathbb{Z}_p^{n \times n}$ (w.l.o.g., we consider $\mathbf{M}$ as a phalanx in this paper), we define the affine subspace $S(\mathbf{M}, \boldsymbol{x}) \subseteq \mathbb{Z}_p^n$ by $S(\mathbf{M}, \boldsymbol{x}) := \{\boldsymbol{x} + \mathbf{M}^\top \cdot \boldsymbol{y} \mid \boldsymbol{y} \in \mathbb{Z}_p^n\}$. If these elements $\mathbf{M}^\top \cdot \boldsymbol{y}$ are all unique, we have $S(\mathbf{M}, \boldsymbol{x}) = \mathbb{Z}_p^n$ and we say that $S(\mathbf{M}, \boldsymbol{x})$ is an n-dimensional affine subspace. It is a basic theorem from linear algebra that the dimension of an affine subspace $S(M, \boldsymbol{x})$ is the rank of M.

If $S(\mathbf{M}', \boldsymbol{x}') \subseteq S(\mathbf{M}, \boldsymbol{x})$, we must have $\mathbf{M}' = \mathbf{M} \cdot \mathbf{T}$ and $\boldsymbol{x}' = \boldsymbol{x} + \mathbf{M}^\top \cdot \boldsymbol{y}$ for some (efficiently computable) matrix $\mathbf{T} \in \mathbb{Z}_p^{n \times n}$ and $\boldsymbol{y} \in \mathbb{Z}_p^n$.

2.2 Bilinear Groups and Complexity Assumptions

We present a few facts related to groups with efficiently prime and computable order bilinear maps. We define them by using a group generator $\mathcal{G}$, an algorithm which takes a security parameter κ as input and outputs a description of a bilinear group $\mathbb{G}$. $\mathcal{G}$ outputs $(N = p_1 \ldots p_n, \mathbb{G}, \mathbb{G}_T, e)$ where $p_1, \ldots, p_n$ are distinct primes, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N, and and e be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that $e(g, g) \neq 1$ for g and for any $u, v \in \mathbb{Z}_N$, it holds that $e(g^u, g^v) = e(g, g)^{uv}$. We say that $\mathcal{G}$ is a bilinear group if the group operation in $\mathbb{G}$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ are both efficiently computable. Notice that the map e is symmetric since $e(g^u, g^v) = e(g, g)^{uv} = e(g^v, g^u)$. When N only has one prime factor p, we call it prime order bilinear groups. Otherwise, it is composite order bilinear groups.

We let $\mathbb{G}_{p_1}, \ldots, \mathbb{G}_{p_n}$ denote the subgroups of order $p_1, \ldots, p_n$ in $\mathbb{G}$ respectively. Furthermore, for $a, b, c \in \{1, p_1, \ldots, p_n\}$ we denote by $\mathbb{G}_{abc}$ the subgroup of order abc.

We define the **Decisional Bilinear Diffie-Hellman** (*DBDH*) and **Decisional Linear** (*DLIN*) assumptions in the prime order groups as follows.

Definition 1. *The DBDH problem in* $(\mathbb{G}, \mathbb{G}_T, e)$ *is, given elements* $(g, g^{c_1}, g^{c_2}, g^{c_3}, T) \in \mathbb{G}^4 \times \mathbb{G}_T$ *with random exponents* $c_1, c_2, c_3 \in \mathbb{Z}_p$ *to decide whether* $T = e(g,g)^{c_1 c_2 c_3}$ *or a random choice of* $\mathbb{G}_T$.

Definition 2. *The DBDH problem in* $(\mathbb{G}, \mathbb{G}_T, e)$ *is, given elements* $(g, f, \nu, g^{c_1}, f^{c_2}, T) \in \mathbb{G}^6$ *with random generators* $g, f, \nu \in \mathbb{G}$ *and exponents* $c_1, c_2 \in \mathbb{Z}_p$ *to decide whether* $T = \nu^{c_1+c_2}$ *or a random choice of* $\mathbb{G}$.

Next, we give three complexity assumptions in the composite order group settings where the group order is a product of 4 primes. All the three assumptions are static (not dependent on the depth of the hierarchy or the number of queries made by an attacker). They were proposed by [2] and can be proved in the generic group model if finding a nontrivial factor of the group order is hard.

Definition 3. *The assumption 1 in* $(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, e)$ *is, given elements* $(N, g_1, g_3, g_4, A_1A_2, B_2B_3)$, *where* $g_1, A_1 \in \mathbb{G}_{p_1}, A_2, B_2 \in \mathbb{G}_{p_2}, A_3, g_3 \in \mathbb{G}_{p_3}, g_4 \in \mathbb{G}_{p_4}$ *are random choices, to decide whether* T *in* $\mathbb{G}_{p_1p_2p_3}$ *or in* $\mathbb{G}_{p_1p_3}$.

Definition 4. *The assumption 2 in* $(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, e)$ *is, given elements* $(N, g_1, g_2, g_3, g_4, g_1^{\alpha}A_2, g_1^{s}B_2, g_2^{r}, A_2^{r})$, *where* $\alpha, s, r \in \mathbb{Z}_N, g_1 \in \mathbb{G}_{p_1}, g_2, A_2, B_2 \in \mathbb{G}_{p_2}, g_3 \in \mathbb{G}_{p_3}, g_4 \in \mathbb{G}_{p_4}$ *are random choices, to decide whether* $T = e(g_1,g_1)^{\alpha s}$ *or a random choice of* $\mathbb{G}_T$.

Definition 5. *The assumption 3 in* $(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, e)$ *is, given elements* $(N, g_1, g_2, g_3, g_4, U, U^sA_{24}, U^r, A_1A_4, A_1^rA_2, g_1^rB_2, g_1^sB_{24})$, $s, r \in \mathbb{Z}_N, g_1, U, A_1 \in \mathbb{G}_{p_1}, g_2, A_2, B_2, D_2, F_2 \in \mathbb{G}_{p_2}, g_3 \in \mathbb{G}_{p_3}, g_4, A_4, B_4, D_4 \in \mathbb{G}_{p_4}, A_{24}, B_{24}, D_{24} \in \mathbb{G}_{p_2p_4}, T_2 \in \mathbb{G}_{p_1p_2p_4}$ *are random choices, to decide whether* $T = A_1^sD_{24}$ *or a random choice of* $\mathbb{G}_{p_1p_2p_4}$.

3 Doubly-Spatial Encryption

Below, we give the definition of doubly-spatial encryption and its security model.

3.1 Algorithms of Doubly-Spatial Encryption

A doubly-spatial encryption system is a quite expressive GIBE contribution that it can be embed many other GIBEs inside. Both the encryption policies and roles are the affine subspaces in the vector space $\mathbb{Z}_p^n$. A role can satisfy a police if and only if corresponding affine subspaces intersect. We note that the doubly-spatial encryption degenerates to the spatial encryption when the encryption policies are restricted to vectors. A doubly-spatial encryption scheme consists of four polynomial time algorithms described as follows:

- **Setup**(λ, n): The algorithm takes as input a security parameter λ and a space dimension n. It returns public parameters PP and a master secret key $\mathsf{Msk} = \mathsf{SK}_{\Gamma}$.

- **Delegate**$(\mathsf{PP}, \mathbb{V}_1, \mathsf{SK}_{\mathbb{V}_1}, \mathbb{V}_2)$: The delegation algorithm takes as input the secret key $\mathsf{SK}_{\mathbb{V}_1}$ for an affine vector subspace $\mathbb{V}_1$ and outputs the secret key $\mathsf{SK}_{\mathbb{V}_2}$ for another affine vector subspace $\mathbb{V}_2$, where $\mathbb{V}_2 \subseteq \mathbb{V}_1$.
- **Enc**$(\mathsf{PP}, \mathbb{W}, m)$: The encryption algorithm encrypts a message m under an affine vector subspace $\mathbb{W} \subseteq \Gamma$ and outputs a ciphertext $\mathsf{CT}_{\mathbb{W}}$.
- **Dec**$(\mathsf{PP}, \mathsf{CT}_{\mathbb{W}}, \mathsf{SK}_{\mathbb{V}}, \mathbb{W})$: The decryption algorithm takes as input the secret key $\mathsf{SK}_{\mathbb{V}}$ to decrypt the ciphertext $\mathsf{CT}_{\mathbb{W}}$. Decryption succeeds if $\mathbb{W} \cap \mathbb{V} \neq \varnothing$, and it outputs the plaintext m.

Remarks. We omit the key generation algorithm since we can create it as a special delegation process. That is,

$$\mathbf{KeyGen}(\mathsf{PP}, \mathbb{V}, \mathsf{Msk}) = \mathbf{Delegate}(\mathsf{PP}, \Gamma, \mathsf{SK}_{\Gamma}, \mathbb{V})$$

3.2 Full Security of Doubly-Spatial Encryption

Our security definition captures semantic security and weak anonymity for the doubly-spatial encryption system by means of the following game between an adversary and a challenger.

- **Setup**(λ, n): The challenger runs the algorithm **Setup**(λ, n) and sends public parameters PP to the adversary.
- **Phase I**: An adaptively $\mathcal{A}$ makes repeated key queries of one of three types:
  1. **Create**: The adversary $\mathcal{A}$ submits delegation queries of $\mathbb{V} \subseteq \Gamma$ to the challenger, who runs the delegation algorithm **Delegate**$(\mathsf{PP}, \Gamma, \mathsf{SK}_{\Gamma}, \mathbb{V})$, but does not give it to $\mathcal{A}$. It instead adds the key to the set S and gives the adversary a reference to it.
  2. **Delegate**: The adversary $\mathcal{A}$ specifies a key $\mathsf{SK}_{\mathbb{V}}$ in the set S and an affine vector space $\mathbb{V}' \subseteq \mathbb{V}$. In response, the challenger runs the delegation algorithm **Delegate**$(\mathsf{PP}, \mathbb{V}, \mathsf{SK}_{\mathbb{V}}, \mathbb{V}')$. It adds this key to the set S and again gives the attacker only a reference to $\mathcal{A}$, not the actual key.
  3. **Reveal**: The adversary $\mathcal{A}$ specifies a key $\mathsf{SK}_{\mathbb{V}}$ in the set S. The challenger gives this key to the attacker and removes it from the set S. We note that the attacker need no longer make any delegation queries for this key because it can run the delegation algorithm on the revealed key for itself.

  Challenge: The adversary $\mathcal{A}$ submits two messages m_0, m_1 and two affine vector subspaces $\mathbb{W}_0, \mathbb{W}_1$ for challenge. We require that the adversary has not been given a decryption key whose affine vector subspace intersects with the challenged affine subspaces for the doubly-spatial encryption case, that is, $\mathbb{W}_0 \cap \mathbb{V} = \varnothing$ or $\mathbb{W}_1 \cap \mathbb{V} = \varnothing$ for all reveal queries of $\mathbb{V}$ in Phase I. The challenger chooses a random $\mu \in \{0, 1\}$, runs the algorithm **Enc**$(\mathsf{PP}, \mathbb{W}_{\mu}, m_{\mu})$, and returns the resulting challenge ciphertext CT^* to the adversary.
- **Phase II**: The second query phase is exactly like the first one, except that the adversary may not issue delegation queries for affine subspaces that intersect with $\mathbb{W}_1$ or $\mathbb{W}_2$.
- **Guess**: The adversary outputs a guess $\mu' \in \{0, 1\}$ and wins if $\mu' = \mu$.

We define the advantage of the adversary $\mathcal{A}$ in attacking a doubly-spatial encryption scheme Π as $Adv^{DSE}_{\mathcal{A},\Pi}(\kappa) := |Pr[\mu' = \mu] - 1/2|$.

Definition 6. *A doubly-spatial encryption scheme Π is adaptively secure if no PPT adversaries have at most non-negligible advantage in winning the above game.*

Non-anonymity. The above game enforces weak anonymity: not only is the adversary unable to determine any properties of the message m_μ based on the ciphertext, he is also unable to determine any properties of the policy $\mathbb{W}_\mu$. To model the non-anonymous security, we can add the constraint that $\mathbb{W}_0 = \mathbb{W}_1$.

We also note that the anonymity we define above is weak anonymity [7], which means that the decryption algorithm should take as input the encryption policy of the ciphertext, and otherwise it is strong anonymity.

4 Fully Secure Doubly-Spatial Encryption

In this section, we propose a fully secure non-anonymous doubly-spatial encryption based on Waters tag-based IBE [14]. Our construction is not so efficient as the analogous spatial encryption constructions [3,7,11], because the length of the ciphertext depends on the dimension of the affine space.

- **Setup**(λ, n): The setup algorithm takes input a security parameter λ. It first chooses bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$ and an n-dimensional affine space $\Gamma = \mathbb{Z}_p^n$. Next, it randomly chooses generators $g, v, v_1, v_2 \in \mathbb{G}$, $\alpha, \alpha_0, \alpha_1, \ldots, \alpha_n, a_1, a_2, b \in \mathbb{Z}_p$. Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$. It publishes the public parameters

$$\mathsf{PP} = \begin{pmatrix} \Gamma,\ g,\ g^\beta,\ w = g^{\alpha_0},\ Z = e(g,g)^{\alpha a_1 b},\ g^{\boldsymbol{\alpha}},\ g^{a_1},\ g^{a_2},\ g^b,\ g^{ba_1}, \\ g^{ba_2},\ \tau_1 = v \cdot v_1^{a_1},\ \tau_2 = v \cdot v_2^{a_2},\ T_1 = \tau_1^b,\ T_2 = \tau_2^b \end{pmatrix}$$

  Then the master secret key is $(g^\alpha, g^{a_1\alpha})$.
- **KeyGen**$(\mathsf{PP}, \mathbb{V}, \mathsf{Msk})$: The algorithm takes input an affine subspace $\mathbb{V} = S(\mathbf{M}, \boldsymbol{x})$ and a master secret key. It randomly chooses r_1, r_2, z_1, z_2, $tag_\mathbb{V} \in \mathbb{Z}_p, \boldsymbol{tag}_\mathbb{V} \in \mathbb{Z}_p^n$ and computes

$$\begin{aligned} &D_1 = g^{\alpha a_1} \cdot v^r, \quad D_2 = g^{-\alpha + z_1} \cdot v_1^r, \quad D_3 = g^{-bz_1}, \\ &D_4 = v_2^r \cdot g^{z_2}, \quad D_5 = g^{-bz_2}, \quad D_6 = g^{br_2}, \quad D_7 = g^{r_1}, \\ &K_0 = g^{r_1(\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle + \alpha_0 tag_\mathbb{V} + \beta)}, \quad \boldsymbol{K} = g^{r_1(\mathbf{M}^\top \boldsymbol{\alpha} + \alpha_0 \boldsymbol{tag}_\mathbb{V})} \end{aligned}$$

  Output the secret key as $\mathsf{SK}_\mathbb{V} = (D_1, \ldots, D_7, K_0, \boldsymbol{K}, tag_\mathbb{V}, \boldsymbol{tag}_\mathbb{V})$.
- **Delegate**$(\mathsf{PP}, \mathbb{V}_1, \mathsf{SK}_{\mathbb{V}_1}, \mathbb{V}_2)$: The algorithm takes input two affine subspaces $\mathbb{V}_1 = S(\mathbf{M}_1, \boldsymbol{x_1})$, $\mathbb{V}_2 = S(\mathbf{M}_2, \boldsymbol{x_2})$ and a secret key of the affine subspace $\mathbb{V}_1$ parsed as $(D_1, \ldots, D_7, K_0, \boldsymbol{K}, tag_{\mathbb{V}_1}, \boldsymbol{tag}_{\mathbb{V}_1})$. Since $\mathbb{V}_2 \subseteq \mathbb{V}_1$, we must have $\mathbf{M}_2 = \mathbf{M}_1 \cdot \mathbf{T}$ and $\boldsymbol{x_2} = \boldsymbol{x_1} + \mathbf{M}_1^\top \cdot \boldsymbol{y}$ for some efficiently

computable matrix $\mathbf{T}$ and vector $\boldsymbol{y} \in \mathbb{Z}_p^n$. We can then compute a key $K'_{\mathbb{V}_2} = (D'_1, \ldots, D'_7, K'_0, \boldsymbol{K'}, tag_{\mathbb{V}_2}, \boldsymbol{tag}_{\mathbb{V}_2})$ as follows:

$$D'_1 = D_1,\ D'_2 = D_2,\ D'_3 = D_3,\ D'_4 = D_4,\ D'_5 = D_5,\ D'_6 = D_6,\ D'_7 = D_7,$$
$$K'_0 = K_0 \cdot \boldsymbol{K}^{\boldsymbol{y}} = g^{r_1(\langle \boldsymbol{x_2}, \boldsymbol{\alpha} \rangle + \alpha_0 tag_{\mathbb{V}_2} + \beta)}, \boldsymbol{K'} = \boldsymbol{K}^{\mathbf{T}^\top} = g^{r_1(\mathbf{M}_2^\top \boldsymbol{\alpha} + \alpha_0 \boldsymbol{tag}_{\mathbb{V}_2})}$$

where the tag values are $tag_{\mathbb{V}_2} = tag_{\mathbb{V}_1} + \langle \mathbf{M}_1^\top \cdot \boldsymbol{y}, \boldsymbol{tag}_{\mathbb{V}_1} \rangle$, $\boldsymbol{tag}_{\mathbb{V}_2} = \mathbf{T}^\top \cdot \boldsymbol{tag}_{\mathbb{V}_1}$. In addition, we need to re-randomize the new secret key. To do this, it randomly picks $r'_1, r'_2, z'_1, z'_2 \in \mathbb{Z}_p$ and computes

$$D''_1 = D'_1 \cdot v^{r'_1 + r'_2} = g^{\alpha a_1} \cdot v^{r''}, \quad D_2 = D'_2 \cdot g^{z'_1} \cdot v_1^{r'_1 + r'_2} = g^{-\alpha + z''_1} \cdot v_1^{r''},$$
$$D''_3 = D'_3 \cdot g^{-bz'_1} = g^{-bz''_1}, \quad D''_4 = D'_4 \cdot v_2^{r'_1 + r'_2} \cdot g^{z'_2} = v_2^{r''} \cdot g^{z''_2},$$
$$D''_5 = D'_5 \cdot g^{-bz'_2} = g^{-bz''_2}, \quad D''_6 = D'_6 \cdot g^{br'_2} = g^{br''_2}, \quad D''_7 = D'_7 \cdot g^{r'_1} = g^{r''_1},$$
$$K''_0 = K'_0 \cdot g^{r'_1(\langle \boldsymbol{x_2}, \boldsymbol{\alpha} \rangle + \alpha_0 tag_{\mathbb{V}_2} + \beta)} = g^{r''_1(\langle \boldsymbol{x_1}, \boldsymbol{\alpha} \rangle + \alpha_0 tag_{V_2} + \beta)},$$
$$\boldsymbol{K''} = \boldsymbol{K'} \cdot g^{r'_1(\mathbf{M}_2^\top \cdot \boldsymbol{\alpha} + \alpha_0 \boldsymbol{tag}_{\mathbb{V}_2})} = g^{r''_1(\mathbf{M}_2^\top \cdot \boldsymbol{\alpha} + \alpha_0 \boldsymbol{tag}_{\mathbb{V}_2})}$$

where $z''_1 = z_1 + z'_1, z''_2 = z_2 + z'_2, r''_1 = r_1 + r'_1, r''_2 = r_2 + r'_2, r'' = r + r'_1 + r'_2$. And it outputs $\mathsf{SK}_{\mathbb{V}_2} = (D''_1, \ldots, D''_7, K''_0, \boldsymbol{K''}, tag_{\mathbb{V}_2}, \boldsymbol{tag}_{\mathbb{V}_2})$.

– **Enc**$(\mathsf{PP}, \mathbb{W}, m)$: Given a message $m \in \mathbb{G}_T$ and an affine subspace $\mathbb{W} = S(\mathbf{M}, \boldsymbol{x})$, the encryption algorithm randomly chooses $s_1, s_2, t, tag_c \in \mathbb{Z}_p$, $\boldsymbol{tag}_c \in \mathbb{Z}_p^n$ and computes

$$C_0 = m \cdot Z^{s_2}, \quad C_1 = g^{b(s_1 + s_2)}, \quad C_2 = g^{ba_1 s_1}, \quad C_3 = g^{a_1 s_1},$$
$$C_4 = g^{ba_2 s_2}, \quad C_5 = g^{a_2 s_2}, \quad C_6 = \tau_1^{s_1} \cdot \tau_2^{s_2}, \quad C_7 = T_1^{s_1} \cdot T_2^{s_2} \cdot w^{-t},$$
$$E_1 = (g^{\alpha_0 \cdot tag_c + \langle \boldsymbol{x}, \boldsymbol{a} \rangle + \beta})^t, \quad E_2 = g^t, \quad \boldsymbol{E}_3 = (g^{\alpha_0 \boldsymbol{tag}_c + \mathbf{M}^\top \cdot \boldsymbol{a}})^t$$

And outputs $\mathsf{CT}_{\mathbb{W}} = (C_0, C_1, \ldots, C_7, E_1, E_2, \boldsymbol{E}_3, tag_c, \boldsymbol{tag}_c)$.

– **Dec**$(\mathsf{PP}, \mathsf{CT}_{\mathbb{W}}, \mathsf{SK}_{\mathbb{V}'}, \mathbb{W})$: Given a ciphertext parsed as $\mathsf{CT}_{\mathbb{W}} = (C_0, C_1, \ldots, C_7, E_1, E_2, \boldsymbol{E}_3, tag_c, \boldsymbol{tag}_c)$ and a secret key of the affine subspace $\mathbb{V}'$ parsed as $(D_1, \ldots, D_7, K_0, \boldsymbol{K}, tag_{\mathbb{V}'}, \boldsymbol{tag}_{\mathbb{V}'})$, if $\mathbb{V}' \cap \mathbb{W} \neq \emptyset$, there exits a vector $\boldsymbol{x}^* \in \mathbb{V}' \cap \mathbb{W}$. Then we can efficiently find $\boldsymbol{y}, \boldsymbol{y}'$ such that $\boldsymbol{x}^* = \boldsymbol{x} + \mathbf{M}^\top \cdot \boldsymbol{y} = \boldsymbol{x}' + (\mathbf{M}')^\top \cdot \boldsymbol{y}'$, where $\mathbb{W} = S(\mathbf{M}, \boldsymbol{x}), \mathbb{V}' = S(\mathbf{M}', \boldsymbol{x}')$. If $\langle \boldsymbol{tag}_{\mathbb{V}'}, \boldsymbol{y}' \rangle + tag_{\mathbb{V}'} - (tag_c + \langle \boldsymbol{tag}_c, \boldsymbol{y} \rangle) \neq 0$, it then recovers

$$\phi_1 = \Big(\prod_{j=1}^{5} e(C_j, D_j) \Big) \cdot \Big(\prod_{j=6}^{7} e(C_j, D_j) \Big)^{-1} = e(g,g)^{\alpha a_1 b s_2} \cdot e(g, w)^{r_1 t}$$
$$\phi_2 = \left(\frac{e(\boldsymbol{K}^{\boldsymbol{y}'} K_0, E_2)}{e(\boldsymbol{E_3}^{\boldsymbol{y}} E_1, D_7)} \right)^{\frac{1}{\langle \boldsymbol{tag}_{\mathbb{V}'}, \boldsymbol{y}' \rangle + tag_{\mathbb{V}'} - (tag_c + \langle \boldsymbol{tag}_c, \boldsymbol{y} \rangle)}} = e(g, w)^{r_1 t}$$

It finally recovers the plaintext as $m = C_0 \cdot \phi_2 \cdot \phi_1^{-1}$; Otherwise, the algorithm aborts and returns $\perp$.

Theorem 1. *The doubly-spatial encryption construction is non-anonymous fully secure under the DLIN and DBDH assumptions.*

Proof. The proof uses the dual system methodology introduced in [14], which involves ciphertexts and private keys that can be normal or semi-functional.

- Semi-functional ciphertexts are generated by first computing a normal ciphertext $\mathsf{CT}_{\boldsymbol{x}} = (C_0, C_1, \ldots, C_7, E_1, E_2, \boldsymbol{E}_3, tag_c, \boldsymbol{tag}_c)$. Then it chooses a random $x \in \mathbb{Z}_p$. It sets $C'_0 = C_0, C'_1 = C_1, C'_2 = C_2, C'_3 = C_3, E'_1 = E_1, E'_2 = E_2, \boldsymbol{E}'_3 = \boldsymbol{E}_3, tag'_c = tag_c, \boldsymbol{tag}'_c = \boldsymbol{tag}_c$, leaving these elements and the tag unchanged. It then sets

$$C'_4 = C_4 \cdot g^{ba_2x}, \quad C'_5 = C_5 \cdot g^{a_2x}, \quad C'_6 = C_6 \cdot v_2^{a_2x}, \quad C'_7 = C_7 \cdot v_2^{ba_2x}$$

  The semi-functional ciphertext is $\mathsf{CT}'_{\boldsymbol{x}} = (C'_0, C'_1, \ldots, C'_7, E'_1, E'_2, \boldsymbol{E}'_3, tag'_c, \boldsymbol{tag}'_c)$.
- Semi-functional secret keys are generated by first computing a normal secret key $\mathsf{SK}_{\mathbb{V}} = (D_1, \ldots, D_7, K_0, \boldsymbol{K}, tag_{\mathbb{V}}, \boldsymbol{tag}_{\mathbb{V}})$. Then it chooses a random $\gamma \in \mathbb{Z}_p$. It sets $D'_3 = D_3, D'_5 = D_5, D'_6 = D_6, D'_7 = D_7, K'_0 = K_0, \boldsymbol{K'} = \boldsymbol{K}, tag'_{\mathbb{V}} = tag_{\mathbb{V}}, \boldsymbol{tag}'_{\mathbb{V}} = \boldsymbol{tag}_{\mathbb{V}}$, leaving these elements and the tags unchanged. It then sets

$$D'_1 = D_1 \cdot g^{-a_1a_2\gamma}, \quad D'_2 = D_2 \cdot g^{a_2\gamma}, \quad D'_4 = D_4 \cdot g^{a_1\gamma}$$

  The semi-functional secret key is $\mathsf{SK}'_{\mathbb{V}} = (D'_1, \ldots, D'_7, K'_0, \boldsymbol{K'}, tag'_{\mathbb{V}}, \boldsymbol{tag}'_{\mathbb{V}})$.

The proof proceeds with a game sequence starting from Game_{Real}, which is the actual attack game. We suppose $\mathcal{A}$ make q reveal key queries in total. 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 k secret key reveal queries are answered by semi-functional secret keys.

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 below. The sequence ends in $q+1$, where the challenge ciphertext is independent of the challenger's bit μ, hence any adversary has no advantage. □

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.*

Lemma 3. *Suppose that there exists an adversary* $\mathcal{A}$ *that makes at most* q *queries and* $|\mathsf{Game}_q - \mathsf{Game}_{q+1}| = \epsilon$*. Then we can build a distinguisher for the DBDH problem.*

Due to space considerations the proof of these lemmas above is given in the full version of this paper.

5 Spatial Encryption with Weak Anonymity

[3,4] use a fully secure and anonymous IPE scheme as a building block to construct a fully secure with anonymity but without constant-size ciphertexts. The ciphertext size incurs linearly with the dimension of the subspace for the encryption policy.

In this section, we propose a fully secure spatial encryption achieving weak anonymity and constant-size ciphertext at the same time in composite order bilinear groups setting. In addition, the distribution of the secret keys produced by delegation are independent of the path taken. That is, all keys for a given affine space come from the same distribution, no matter how they were delegated. The scheme can be seen as a transform from anonymous-HIBE [2] to spatial encryption. We describe our construction of the weakly anonymous spatial encryption scheme.

– **Setup**(λ, n): The setup algorithm chooses a bilinear group $\mathbb{G}$ with order $N = p_1p_2p_3p_4$ and an n-dimensional affine space $\Gamma = \mathbb{Z}_N^n$. It randomly chooses $\alpha, a_0 \in \mathbb{Z}_N$,$\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_N^n$, $g, h \in \mathbb{G}_{p_1}$, $X_3 \in \mathbb{G}_{p_3}$ and $X_4, Y_4 \in \mathbb{G}_{p_4}$. It then outputs public parameters and master key as

$$\mathsf{PP} := (\Gamma, N, X_3, X_4, g^{a_0}Y_4, g, g^{a_1}, \ldots, g^{a_n}, e(g,g)^{\alpha}), \quad \mathsf{Msk} := (g^{a_0}, \alpha).$$

– **KeyGen**$(\mathsf{PP}, \mathbb{V}, \mathsf{Msk})$: On input an affine subspace $\mathbb{V} = S(\mathbf{M}, \boldsymbol{x})$ and a master secret key, it randomly chooses $r, r' \in \mathbb{Z}_N$, $R_0, R_1, R_0', R_1' \in \mathbb{G}_{p_3}$ and $\boldsymbol{Q} = (Q_1, \ldots, Q_n), \boldsymbol{Q'} = (Q_1', \ldots, Q_n') \in \mathbb{G}_{p_3}^n$. It then outputs the secret key $\mathsf{SK}_{\mathbb{V}} = (K_0, K_1, \boldsymbol{E} = (E_1, \ldots, E_n), K_0', K_1', \boldsymbol{E'} = (E_1', \ldots, E_n'))$ by computing

$$K_0 = g^r \cdot R_0, \quad K_1 = g^{\alpha + r(a_0 + \langle \boldsymbol{a}, \boldsymbol{x} \rangle)} \cdot R_1, \quad \boldsymbol{E} = \left(g^{\mathbf{M}^\top \cdot \boldsymbol{a}}\right)^r \cdot \boldsymbol{Q}$$
$$K_0' = g^{r'} \cdot R_0', \quad K_1' = g^{r'(a_0 + \langle \boldsymbol{a}, \boldsymbol{x} \rangle)} \cdot R_1', \quad \boldsymbol{E'} = \left(g^{\mathbf{M}^\top \cdot \boldsymbol{a}}\right)^{r'} \cdot \boldsymbol{Q'}$$

Note that, $\mathsf{SK}_{\mathbb{V}}$ is composed by two sub-keys. The first sub-key, $(K_0, K_1, \boldsymbol{E})$, is used to decrypt, the second, $(K_0', K_1', \boldsymbol{E'})$, is used to delegate.

– **Delegate**$(\mathsf{PP}, \mathbb{V}_1, \mathsf{SK}_{\mathbb{V}_1}, \mathbb{V}_2)$: The algorithm takes input two affine subspaces $\mathbb{V}_1 = S(\mathbf{M}_1, \boldsymbol{x_1})$, $\mathbb{V}_2 = S(\mathbf{M}_2, \boldsymbol{x_2})$ and a secret key parsed as $\mathsf{SK}_{\mathbb{V}_1} = (K_0, K_1, \boldsymbol{E}, K_0', K_1', \boldsymbol{E'})$. Since $\mathbb{V}_2 \subseteq \mathbb{V}_1$, we have $\mathbf{M}_2 = \mathbf{M}_1 \cdot \mathbf{T}$ and $\boldsymbol{x_2} = \boldsymbol{x_1} + \mathbf{M}_1^\top \cdot \boldsymbol{y}$ for some matrix $\mathbf{T}$ and vector $\boldsymbol{y} \in \mathbb{Z}_N^n$. It randomly chooses $r_1, r_2 \in \mathbb{Z}_N$, $R_0, R_1, R_0', R_1' \in \mathbb{G}_{p_3}$ and $\boldsymbol{Q} = (Q_1, \ldots, Q_n), \boldsymbol{Q'} = (Q_1', \ldots, Q_n') \in \mathbb{G}_{p_3}^n$. Then compute a key $\mathsf{SK}_{\mathbb{V}_2} = (K_{2,0}, K_{2,1}, \boldsymbol{E}_2, K_{2,0}', K_{2,1}', \boldsymbol{E}_2')$ as follows:

$$K_{2,0} = K_0 (K_0')^{r_1} \cdot R_0, \quad K_{2,1} = K_1 \cdot \boldsymbol{E}^{\boldsymbol{y}} \cdot (K_1' \cdot (\boldsymbol{E'})^{\boldsymbol{y}})^{r_1} \cdot R_1,$$
$$K_{2,0}' = (K_0')^{r_2} \cdot R_0', \quad K_{2,1}' = (K_1' \cdot (\boldsymbol{E'})^{\boldsymbol{y}})^{r_2} \cdot R_1',$$
$$\boldsymbol{E}_2 = (\boldsymbol{E} \cdot (\boldsymbol{E'})^{r_1})^{\mathbf{T}^\top} \cdot \boldsymbol{Q}, \quad \boldsymbol{E}_2' = ((\boldsymbol{E'})^{r_2})^{\mathbf{T}^\top} \cdot \boldsymbol{Q'}$$

– **Enc**$(\mathsf{PP}, \boldsymbol{x}, m)$: Given a message $m \in \mathbb{G}_T$ and a vector $\boldsymbol{x} \in \mathbb{Z}_N^n$, the encryption algorithm randomly chooses $s \in \mathbb{Z}_N$ and $Z, Z' \in \mathbb{G}_{p_4}$. Then computes the ciphertext $\mathsf{CT}_{\boldsymbol{x}} = (C, C_0, C_1)$ as

$$C = m \cdot e(g,g)^{\alpha s}, \quad C_0 = g^s \cdot Z', \quad C_1 = (g^{a_0} Y_4 \cdot g^{\langle \boldsymbol{a}, \boldsymbol{x} \rangle})^s \cdot Z$$

– **Dec**$(\mathsf{PP}, \mathsf{SK}_{\mathbb{V}}, \mathsf{CT}_{\boldsymbol{x}})$: Given a ciphertext parsed as $\mathsf{CT}_{\boldsymbol{x}} = (C, C_0, C_1)$ and a secret key parse as $\mathsf{SK}_{\mathbb{V}} = (K_0, K_1, \boldsymbol{E}, K_0', K_1', \boldsymbol{E}')$, if $\boldsymbol{x} \in \mathbb{V} = S(\mathbf{M}, \boldsymbol{x}')$, we can efficiently find $\boldsymbol{y}$ such that $\boldsymbol{x} = \boldsymbol{x}' + \mathbf{M}^{\top} \cdot \boldsymbol{y}$. Then it computes

$$e(g,g)^{\alpha s} = \frac{e(C_0, K_1 \boldsymbol{E}^{\boldsymbol{y}})}{e(C_1, K_0)}$$

and recovers the message as $m = C/e(g,g)^{\alpha s}$.

Theorem 2. *The construction above is fully secure under the Assumption 1, 2, 3.*

Due to space considerations the proof is given in the full version of this paper.

Acknowledgment. The work is supported by the National Natural Science Foundation of China under Grant No.61170278,91118006, and the National Basic Research Program (973) of China under Grant No. 2012CB315804.

References

1. 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)
2. De Caro, A., Iovino, V., Persiano, G.: Fully Secure Anonymous HIBE and Secret-Key Anonymous IBE with Short Ciphertexts. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 347–366. Springer, Heidelberg (2010)
3. Chen, J., Lim, H. W., Ling, S., Wang, H.: Fully Secure Spatial Encryption under Simple Assumptions with Constant-Size Ciphertexts. IACR Cryptology ePrint Archive 2011:650 (2011)
4. Chen, J., Lim, H.W., Ling, S., Wang, H.: The Relation and Transformation between Hierarchical Inner Product Encryption and Spatial Encryption. IACR Cryptology ePrint Archive 2011:455 (2011)
5. Delerablée, C., Pointcheval, D.: Dynamic Threshold Public-Key Encryption. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 317–334. Springer, Heidelberg (2008)
6. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access conrol of encrypted data. In: ACM Conference on Computer and Communications Security, ACM CCS (2006)
7. Hamburg, M.: Spatial encryption. IACR Cryptology ePrint Archive 2011:389
8. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
9. Lewko, A., Sahai, A., Waters, B.: Revocation systems with very small private keys. In: IEEE Symposium on Security and Privacy 2010, pp. 273–285 (2010)
10. 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)

11. Moriyama, D., Doi, H.: A fully secure spatial encryption scheme. IEICE Transactions 94-A(1), 28–35 (2011)
12. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
13. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
14. 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)

Strongly Authenticated Key Exchange Protocol from Bilinear Groups without Random Oracles

Zheng Yang and Jörg Schwenk

Horst Görtz Institute for IT Security
Ruhr-University Bochum, Germany
zheng.yang@rub.de

Abstract. Malicious insider security of authenticated key exchange (AKE) protocol addresses the situation that an AKE protocol is secure even with existing dishonest parties established by adversary in corresponding security experiment. In the eCK model, the EstablishParty query is used to model the malicious insider setting. However such strong query is not clearly formalized so far. We show that the proof of possession assumptions for registering public keys are of prime importance to malicious insider security. In contrast to previous schemes, we present an eCK secure protocol in the standard model, without assuming impractical, strong, concurrent zero-knowledge proofs of knowledge of secret keys done to the CA at key registration. The security proof of our scheme is based on standard pairing assumption, collision resistant hash functions, bilinear decision Diffie-Hellman (BDDH) and decision linear Diffie-Hellman (DLIN) assumptions, and pseudo-random functions with pairwise independent random source πPRF [12].

Keywords: one-round authenticated key exchange, pairing, insider security.

1 Introduction

Many critical applications rely on the existence of a confidential channel established by authenticated Key Exchange (AKE) protocols over open networks. In contrast to the most prominent key exchange protocol is the Diffie-Hellman protocol [7] which is vulnerable to the existence of an active adversary (i.e. man-in-the-middle attacks), a secure AKE should be secure against an active adversaries. Over the last decade, the security of AKE against active attacks has been developed increasingly in stronger models. In this paper, we consider PKI-based two party AKE protocol in presence of adversary with strong capabilities. LaMacchia, Lauter and Mityagin [8] recently presented strong security definitions for two-pass key exchange protocol, which is referred as eCK security model. Since the introducing of eCK model, many protocols (e.g. [12,18,10,11]) have been proposed to provide eCK security. But most of those protocols are proven under random oracle model.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 264–275, 2012.

Public Key Registration and EstablishParty Query. In the original eCK model [8], the public key registration was considered from three situations: (i) honest key registration, (ii) proof of knowledge (POK) key registration, and (iii) arbitrary key registration. In the security experiment, the above cases are simulated differently by the challenger. As for the honest key registration, all public keys are generated honestly by challenger, and for the other two cases the public keys might be chosen by adversary. In the latter literatures, the EstablishParty query was introduced to model such chosen public key attacks, that might relate to attacks like unknown key share (UKS) attacks [4], etc. In the security experiment, each registered corrupted party by EstablishParty query is controlled by the adversary, which can be used to interact with honest parties in sessions.

We notice that the EstablishParty query has not been clearly formalized so far, where no POK assumption for key registration is addressed by this query. In particular, different POK assumptions would result in different type of adversaries in the security experiment, that would impact the proof simulation, in particular for the proof without random oracles. General speaking, there are two major POK assumptions: knowledge of secret key (KOSK) assumption and plain public key (PPK) assumption. The KOSK assumption (e.g. used in [9]), that requires each party provides the certification authority (CA) with a proof of knowledge of its secret key before the CA certifies the corresponding public key. While implementing the (KOSK) assumption, it is assumed that there exists either efficient knowledge extractor (satisfying requirement in [1]), or the adversary simply hands the challenger corresponding secret keys. The another assumption is the plain public key (PPK) assumption (following the real-world standards PKCS#10 [13]) that nothing more is required than in any usage of public-key cryptography, where the proof of possession might be implemented by having the user send the CA a signature (under the public key it is attempting to get certified) of some message that includes the public key and the user identity. On the contrary, the private keys, of dishonest parties registered under PPK assumption, might be only known by adversary, nor by the challenger. As pointed out by Mihir Bellare and Gregory Neven in [2], the KOSK assumption can't be implemented by the proof based on plain public key (PPK) assumption, and the PPK assumption is much cheaper and more realistic than KOSK assumption.

While designing and analysing eCK protocol against chosen public key attacks, corresponding POK assumption should be explicitly modeled by EstablishParty query. Recently Moriyama and Okamoto (MO) presented an eCK-secure key exchange protocol [11] in the standard model. However, as a negative example, an appropriate POK assumption is never clearly made in the proof of MO protocol. In particular, the MO protocol can't be proven secure without KOSK assumption. Since under PPK assumption, if the long-term keys of test oracle (e.g. owned by party $\hat{A}$) are not corrupted and set in terms of a DDH challenge instance, then the challenger is unable to simulate the session key of other oracles of $\hat{A}$ which have dishonest peer (e.g. party $\hat{C}$) established by

adversary. Because computing the long-term shared key involving parties $\hat{A}$ and $\hat{C}$ is a CDH hard problem for the challenger. Also, it still left out the task of formally justifying a claim on how to implement the abstract KOSK assumption for MO protocol. Therefore we are motivated to clearly formalize the EstablishParty query and strive to seek eCK secure protocol against chosen public key attacks without KOSK assumption and NAXOS tricks in the standard model.

Potential Threat on leakage of Secret Exponent. Besides the leakage of long-term and ephemeral private keys modeled by eCK model, a 'well' designed protocol should resist with the compromise of other session key related secret information, even though such compromise is not normally expected. A noteworthy instance is the leakage of ephemeral intermediate exponent (e.g., the $a_1 + a_3\alpha$ in MO protocol), due to the up-to-date side-channel attacks. Such kind of leakage has been studied by Sarr et al. [15,14] based on HMQV protocol. In particular, as pointed by Yoneyama et al. [20], the leakage of intermediate exponent of Okamoto protocol [12] and MO protocol (in two different sessions) would result in exposure of long-term keys. Therefore one should take care of those intermediate exponents while designing protocols, even though it is hard to prove the security on resilience of such leakage (as claimed in [20]). Moreover, the ephemeral secrets that can be revealed in the eCK model, should be clearly specified by each protocol based on appropriate implementation scenario. Note that if the protocol is executed in a computer infected with malware, then all secret session states (including those intermediate exponents mentioned above) might be exposed.

1.1 Contribution

In this paper, we clarify the EstablishParty query in terms of different type of POK assumptions. We present an eCK secure AKE protocol in the standard model, that is able to resist with chosen public key attacks based on only plain public key registration assumption and without NAXOS trick. The security of proposed protocol is based on standard pairing assumption, collision resistant hash functions, bilinear decision Diffie-Hellman (BDDH) and decision linear Diffie-Hellman (DLIN) assumptions, and pseudo-random functions with pairwise independent random source πPRF [12]. Not surprisingly, one must pay a small price for added security with one paring operation. However our protocol can be implemented in a group where DDH problem is easy.

We show that the internal computation algorithm really matters for the security of a protocol. From our construction approach, we illustrate an example on how to mitigate the threat due to leakage of intermediate exponents, for which exponents involve only long-term secrets. In order to relieve the consequences of such leakage, we adapt a generic strategy: first blind those intermediate exponents using uniform random value (e.g. the ephemeral private keys) and next remove the random value after completing corresponding exponential operation.[1]

[1] This would mitigate the attacks described in [14,20], when the secret intermediate exponent is exposed somehow.

Our approach can also be applied to improve the MO protocol [11] or Okamoto protocol [12] in a similar way.

1.2 Related Work

In the first eCK security model introduced by LaMacchia, Lauter and Mityagin [8], they model the insider security by allowing adversary to register arbitrary public keys without proving knowledge of the corresponding secret key, which was formalized by EstablishParty query in later literatures.

Since then many eCK secure protocols, e.g. [8,12,10,15,14], have been correctly proven under the malicious insider setting. But most of them are only provable secure with the help of random oracles. Although the protocol [12] by Okamoto is eCK secure in the standard model without KOSK assumption, this protocol heavily relies on the NAXOS trick. Even though the NAXOS trick hides the exponent of the ephemeral public key, it might be leaked because of the up-to-date side-channel attacks. Therefore, a lot of works [11,18] are motivated to propose eCK-secure key exchange protocols without the NAXOS tricks.

Sarr et al. [15], recently described some potential threats on HMQV due to the leakage of secret intermediate exponent (i.e. the $x+aD$, where $D = H(\hat{A}, \hat{B}, X)$). Namely, if such intermediate exponents in different sessions are identical, the adversary can obtain the secret signature in the target session. In the later, Sarr et al. [14] strengthened the eCK model by allowing the adversary to learn certain intermediate results while computing the session key, under specific implementation environment wherein a tamper-proof device is involved to store long-term keys while session keys are used on an untrusted host machine. The seCK model was further studied by Yoneyama et al., in recent work [20]. They pointed out errors in the security proofs of SMQV and FHMQV [14] on leakage of intermediate computations. Unfortunately, their results also showed that there is no scheme has been provably secure in the seCK model.

2 Preliminaries

Notations. We let κ denote the security parameter and 1^κ the string that consists of κ ones. Each party has a long-term authentication key which is used to prove the identity of the party in an AKE protocol. We let a 'hat' on top of a capital letter denotes an identifier of a participant, without the hat the letter denotes the public key of that party, and the same letter in lower case denotes a private key. For example, a party $\hat{A}$ is supposed to register its public key $A = g^a$ at certificate authority (CA) and keeps corresponding long-term secret key $sk_A = a$ privately. Let $[n] = \{1, \ldots, n\} \subset \mathbb{N}$ be the set of integers between 1 and n. If S is a set, then $a \in_R S$ denotes the action of sampling a uniformly random element from S.

To construct our scheme, we need standard security notions of pseudo-random functions (PRF), pseudo-random functions with pairwise independent random sources (πPRF), collision resistant hash functions, the Bilinear Decision Diffie-Hellman (BDDH) and Decision Linear Diffie-Hellman (DLIN) assumptions. These are detailed in the full version [19].

3 AKE Security

In this section we present the formal security model for two party PKI-based authenticated key exchange (AKE) protocol. While modeling the active adversaries, we provide with an 'execution environment' following an important line of research [5,8,17] dates back to Bellare and Rogaway [3]. We will use the framework as in [17] with slight modification.

Execution Environment. Assume there exist a fixed number of parties $\{P_1, \ldots, P_\ell\}$ for $\ell \in \mathbb{N}$, where each party $P_i \in \{P_1, \ldots, P_\ell\}$ is a potential protocol participant and each party has a long-term key pair $(pk_i, sk_i) \in (\mathcal{PK}, \mathcal{SK})$ corresponds to its identity i, where $\{\mathcal{PK}, \mathcal{SK}\}$ are keyspaces of long-term keys. To model several sequential and parallel executions of the protocol, each party P_i is modeled by a collection of oracles $\pi_i^1, \ldots, \pi_i^d$ for $d \in \mathbb{N}$. Each oracle π_i^s represents one single process that executes an instance of the protocol. All oracles $\pi_i^1, \ldots, \pi_i^d$ representing party P_i have access to the same long-term key pair (pk_i, sk_i) of P_i and to all public keys $pk_1, \ldots, pk_\ell$. Moreover, each oracle π_i^s maintains a separate internal state

- a variable Φ storing the identity j of an intended communication partner P_j,
- a variable $\Psi \in \{\texttt{accept}, \texttt{reject}\}$,
- a variable $K \in \mathcal{K}$ storing the session key used for symmetric encryption between π_i^s and party P_Φ, where $\mathcal{K}$ is the keyspace of the protocol.
- and some additional temporary state variable st (which may, for instance, be used to store ephemeral Diffie-Hellman exponents or other intermediate values).

The internal state of each oracle is initialized to $(\Phi, \Psi, K, st) = (\emptyset, \emptyset, \emptyset, \emptyset)$. At some point during the protocol execution each party would generate the session key according to the key exchange protocol specification when turning to state $(\Psi, K) = (\texttt{accept}, K)$ for some K, and at some point with internal state $(\Psi, K) = (\texttt{reject}, \emptyset)$ where $\emptyset$ denotes the empty string. We will always assume (for simplicity) that $K \neq \emptyset$ if an oracle has reached `accept` state.

An adversary may interact with these oracles by issuing the following queries.

- $\mathsf{Send}(\pi_i^s, m)$: The adversary can use this query to send any message m of his own choice to oracle π_i^s. The oracle will respond according to the protocol specification, depending on its internal state. If the first message $m = (\top, \tilde{j})$ consists of a special symbol $\top$ and a value $\tilde{j}$ which is either $\emptyset$ or identity j, then π_i^s will set $\Phi = \tilde{j}$ and respond with the first protocol message. If $\tilde{j} = \emptyset$ then Φ will be set as identity j at some point according to protocol specification. [2]
- $\mathsf{RevealKey}(\pi_i^s)$: Oracle π_i^s responds to a $\mathsf{RevealKey}$-query with the contents of variable K.
- $\mathsf{StateReveal}(\pi_i^s)$: Oracle π_i^s responds the contents secret state stored in variable st.

[2] A protocol might be run in either pre- or post-specified peer model here [6].

- $\mathsf{EstablishParty}(pk_m, sk_m, P_m)$ This query registers an identity $m(\ell < m < \mathbb{N})$ and a static public/private key pair (pk_m, sk_m) on behalf of a party P_m, if one of the following conditions is held: (i) $sk_m = \emptyset$ and $pk_m \in \mathcal{PK}$, (ii) $sk_m \in \mathcal{SK}$ and sk_m is the correct private key for public key pk_m; otherwise a failure symbol $\perp$ is returned. Parties established by this query are called corrupted or adversary controlled.
- $\mathsf{Corrupt}(P_i)$: Oracle π_i^1 responds with the long-term secret key sk_i of party P_i. After this query, oracles π_i^s can still be asked queries using the compromised key sk_i.
- $\mathsf{Test}(\pi_i^s)$: This query may only be asked once throughout the game. Oracle π_i^s handles this query as follows: If the oracle has state $\Psi = \texttt{reject}$ or $K = \emptyset$, then it returns some failure symbol $\perp$. Otherwise it flips a fair coin b, samples a random element $K_0 \xleftarrow{\$} \mathcal{K}$, sets $K_1 = K$ to the 'real' session key, and returns K_b.

We note that the exact meaning of the StateReveal must be defined for each protocol separately, namely the content stored in the variable st during protocol execution. In EstablishParty query, the private key sk_m corresponds to the proof of knowledge assumptions for public key registration, which should be specified in the security proof of each protocol. If $sk_m = \emptyset$ then the plain public key or arbitrary key registration assumption is modeled, otherwise the knowledge of secret key assumption is modeled.

Secure AKE Protocols. We first define the partnering of two oracles via *matching conversations* that was first introduced by Bellare and Rogaway [3] in order to define correctness and security of an AKE protocol precisely, and refined latter in [17]. In the following let T_i^s denote the transcript of messages sent and received by oracle π_i^s. We assume that messages in a transcript T_i^s are represented as binary strings. Let $|T_i^s|$ denote the number of its messages. Assume there are two transcripts T_i^s and T_t^j, where $m := |T_i^s|$ and $n := |T_j^t|$. We say that T_i^s is a prefix of T_j^t if $0 < m \leq n$ and the first m messages in transcripts T_i^s and T_j^t are pairwise equivalent as binary strings.

Definition 1. *We say that a processes π_i^s has a* matching conversation *to oracle π_j^t, if*

- *π_i^s has sent the last message(s) and T_j^t is a prefix of T_i^s, or*
- *π_j^t has sent the last message(s) and T_i^s is a prefix of T_j^t.*

We say that two oracles π_i^s and π_j^t have matching conversations *if π_i^s has a* matching conversation *to process π_j^t, and vice versa.*

Definition 2 (Freshness). *Let π_i^s be a completed oracle held by an honest party P_i with honest peer P_j, and both parties P_i and P_j are not registered by* EstablishParty *query. Let π_j^t be a completed oracle, if it exists, such that π_i^s and π_j^t have matching conversations. Then the oracle π_i^s is said to be fresh (unexposed) if none of the following conditions holds:*

1. *The adversary* $\mathcal{A}$ *either issued* $\mathsf{RevealKey}(\pi_i^s)$, *or* $\mathsf{RevealKey}(\pi_j^t)$ *(if such* π_j^t *exists).*
2. *If* π_j^t *exists,* $\mathcal{A}$ *either issued:*
   (a) Both $\mathsf{Corrupt}(P_i)$ *and* $\mathsf{StateReveal}(\pi_i^s)$, *or.*
   (b) Both $\mathsf{Corrupt}(P_j)$ *and* $\mathsf{StateReveal}(\pi_j^t)$.
3. *If* π_j^t *does not exist,* $\mathcal{A}$ *either issued:*
   (a) Both $\mathsf{Corrupt}(P_i)$ *and* $\mathsf{StateReveal}(\pi_i^s)$, *or*
   (b) $\mathsf{Corrupt}(P_j)$.

Definition 3 (Security Experiment). *In the experiment, the following steps are performed:*

1. *The challenger implements the collection of oracles* $\{\pi_i^s : i \in [\ell], s \in [d]\}$. *At the beginning of the experiment, the challenger generates* ℓ *long-term key pairs* (pk_i, sk_i) *for all* $i \in [\ell]$, *and gives the adversary* $\mathcal{A}$ *all public keys* $pk_1, \ldots, pk_\ell$ *as input.*
2. $\mathcal{A}$ *may issue polynomial number (in the security parameter* κ*) of queries as described above in the execution environment, namely* $\mathcal{A}$ *makes queries:* Send, StateReveal, EstablishParty, Corrupt *and* RevealKey.
3. *At some point,* $\mathcal{A}$ *issues a* $\mathsf{Test}(\pi_i^s)$ *query on a fresh oracle* π_i^s *during the experiment with only once.*
4. *At the end of the experiment, the* $\mathcal{A}$ *terminates with outputting a bit* b' *as its guess for bit* b *of* Test *query.*

Security of AKE protocols is now defined by requiring that the protocol is a secure AKE protocol, thus an adversary cannot distinguish the session key K of a fresh oracle from a random key.

Definition 4 (Secure Authenticated Key Exchange Protocol). *We say an AKE protocol is secure in the security experiment as Definition 3, if for all probabilistic polynomial-time (PPT) adversaries* $\mathcal{A}$ *and for some negligible probability* $\epsilon = \epsilon(\kappa)$ *in the security parameter hold that:*

- *If two fresh oracles* π_i^s *and* π_j^t *accept with matching conversations, then both oracles hold the same session key* K.
- *When* $\mathcal{A}$ *returns* b' *such that*
  - $\mathcal{A}$ *has issued a* Test *query on an oracle* π_i^s *without failure, and*
  - π_i^s *has internal state* $\Phi = j$, *and*
  - π_i^s *is fresh throughout the security experiment.*

  Then the probability that b' *equals the bit* b *sampled by the* ***Test****-query is bounded by*

$$|\Pr[b = b'] - 1/2| \leq \epsilon.$$

4 A Strong AKE Protocol without Random Oracles

In this section we present a pairing-based strong AKE protocol without random oracles, which is informally depicted in Figure 1.

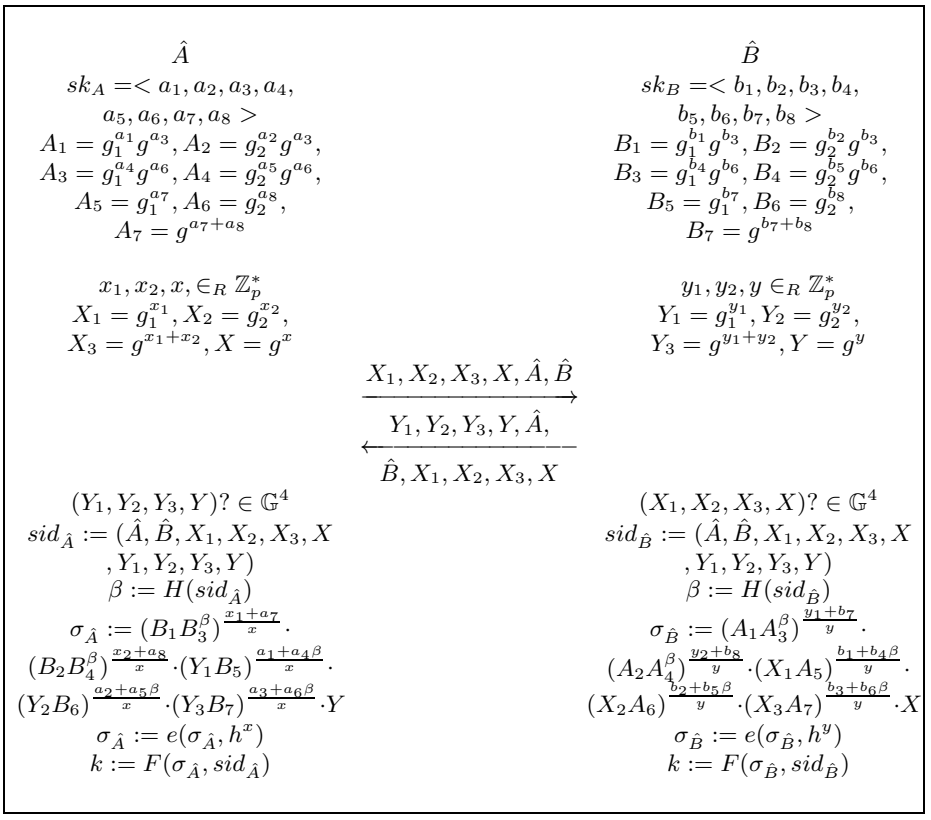


Fig. 1. The AKE Protocol without Random Oracles

Protocol Description. The AKE protocol takes as input the following building blocks:

- Symmetric bilinear groups $(\mathbb{G}, g, \mathbb{G}_T, p, e)$, where the generator of group $\mathbb{G}_T$ is $e(g, g)$ and along with another random generators g_1, g_2 and h of $\mathbb{G}$.
- A collision resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p^*$,
- A pairwise independent pseudo-random function (πPRF) F, with index $\{I_{\mathbb{G}_T}, f_{\mathbb{G}_T}\}$ where $I_{\mathbb{G}_T} := \{(U, V, \alpha) | (U, V, \alpha) \in \mathbb{G}_T^2 \times \mathbb{Z}_p\}$ and $f_{\mathbb{G}_T} := (U, V, \alpha) \to U^{r_1+\alpha r_2} V$ with $(r_1, r_2) \in_R \mathbb{Z}_p^2$.

Long-term Key Generation: on input the security parameter κ, the long-term keys of each party $\hat{A}$ is generated as following: $\hat{A}$ selects long-term private keys : $(a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8) \in_R \mathbb{Z}_p^8$, and compute the long-term public keys: $(A_1, A_2, A_3, A_4, A_5, A_6, A_7) := (g_1^{a_1} g^{a_3}, g_2^{a_2} g^{a_3}, g_1^{a_4} g^{a_6}, g_2^{a_5} g^{a_6}, g_1^{a_7}, g_2^{a_8}, g^{a_7+a_8})$.

Protocol Execution

1. Upon activation a session $(\hat{A}, \hat{B})$, the initiator $\hat{A}$ performs the steps: (a) Choose three ephemeral private keys $x_1, x_2, x, \in_R \mathbb{Z}_p^3$; (b) Compute $X_1 :=$

$g^{x_1}, X_2 := g^{x_2}, X_3 := g^{x_1+x_2}$ and $X := g^x$; (c) Set session identifier $sid_{\hat{A}} := (\hat{A}, \hat{B}, X_1, X_2, X_3, X)$; (d) Send $(X_1, X_2, X_3, X, \hat{A}, \hat{B})$ to $\hat{B}$.

2. Upon receiving $(X_1, X_2, X_3, X, \hat{A}, \hat{B})$, the responder $\hat{B}$ does the following: (a) Verify that $(X_1, X_2, X_3, X) \in \mathbb{G}^4$; (b) Choose three ephemeral private keys $y_1, y_2, y \in_R Z_p^3$; (c) Compute $Y_1 := g^{y_1}, Y_2 := g^{y_2}, Y_3 := g^{y_1+y_2}$ and $Y := g^y$; (d) Set session identifier $sid_{\hat{B}} := (\hat{A}, \hat{B}, X_1, X_2, X_3, X, Y_1, Y_2, Y_3, Y)$ and compute $\beta := H(sid_{\hat{B}})$; (e) Compute $\sigma_{\hat{B}} := (A_1 A_3^{\beta})^{\frac{y_1+b_7}{y}} \cdot (A_2 A_4^{\beta})^{\frac{y_2+b_8}{y}} \cdot (X_1 A_5)^{\frac{b_1+b_4\beta}{y}} \cdot (X_2 A_6)^{\frac{b_2+b_5\beta}{y}} \cdot (X_3 A_7)^{\frac{b_3+b_6\beta}{y}} \cdot X$ and $\sigma_{\hat{B}} := e(\sigma_{\hat{B}}, h^y)$; (f) Compute session key $k := F(\sigma_{\hat{B}}, sid_{\hat{B}})$ and erase all intermediate values; (g) Send $(Y_1, Y_2, Y_3, Y, \hat{A}, \hat{B}, X_1, X_2, X_3, X)$ to $\hat{A}$.
3. Upon receiving $(Y_1, Y_2, Y_3, Y, \hat{A}, \hat{B}, X_1, X_2, X_3, X)$ does the following: (a) Verify that exist a session identified by $(\hat{A}, \hat{B}, X_1, X_2, X_3, X)$ and $(Y_1, Y_2, Y_3, Y) \in \mathbb{G}^4$; (b) Update session identifier $sid_{\hat{A}} := (\hat{A}, \hat{B}, X_1, X_2, X_3, X, Y_1, Y_2, Y_3, Y)$, and compute $\beta := H(sid_{\hat{A}})$; (c) Compute $\sigma_{\hat{A}} := (B_1 B_3^{\beta})^{\frac{x_1+a_7}{x}} \cdot (B_2 B_4^{\beta})^{\frac{x_2+a_8}{x}} \cdot (Y_1 B_5)^{\frac{a_1+a_4\beta}{x}} \cdot (Y_2 B_6)^{\frac{a_2+a_5\beta}{x}} \cdot (Y_3 B_7)^{\frac{a_3+a_6\beta}{x}} \cdot Y$ and $\sigma_{\hat{A}} := e(\sigma_{\hat{A}}, h^x)$; (d) Compute session key as $k := F(\sigma_{\hat{A}}, sid_{\hat{A}})$ and erase all intermediate values.

   We assume, only the ephemeral private keys, i.e. (x_1, x_2, x) and (y_1, y_2, y) would be stored as secret in the state variable st.[3]

Security Analysis. In the following, we show that the proposed protocol is an eCK secure protocol in the sense of Definition 4.

Theorem 1. *Suppose that the* $(t, q, \epsilon_{\mathsf{BDDH}})$*-Bilinear DDH assumption and* $(t, q, \epsilon_{\mathsf{DLIN}})$*-Decision linear assumption hold in bilinear groups* $(\mathbb{G}, g, \mathbb{G}_T, p, e)$*, the hash function* H *is* $(t, \epsilon_{\mathsf{CR}})$*-secure, and a* $(t, \epsilon_{\pi\mathsf{PRF}})$*-secure* $\pi\mathsf{PRF}$ *family with index* $\{I_{\mathbb{G}_T}, f_{\mathbb{G}_T}\}$ *where* $I_{\mathbb{G}_T} := \{(U, V, \alpha) | (U, V, \alpha) \in \mathbb{G}^2 \times \mathbb{Z}_p\}$ *and* $f_{\mathbb{G}_T} := (U, V, \alpha) \rightarrow U^{r_1+\alpha r_2} V$ *with* $(r_1, r_2) \in_R \mathbb{Z}_p^2$*, with respect to the definitions in Section 2. Then the proposed protocol is a* (t', ϵ')*-eCK secure AKE as Definition 4.*

Proof of Theorem 1. Due to space limitation, we here only provide the sketch of the proof. We will present the details of the proof in the full paper [19].

In the security experiment, the adversary is allowed to query $\mathsf{EstablishParty}(pk_m, sk_m, P_m)$ with $sk_m = \emptyset$ while registering a public key pk_m for dishonest party P_m. Namely, we allow arbitrary public key registration.

In order to complete the proof of Theorem 1, we must provide the security proofs for all freshness related cases as Definition 2. However, applying the Propositions 1 and 2 from [11], the security can be reduced to the following two cases:

[3] This can be achieved by performing the computation in steps 2(e) and 2(f) (resp. steps 3(c) and 3(d)) on a smart card, where the long-term keys are stored. In this case, the intermediate values would not be exposed due to e.g. malware attacks on the PC, which we model with $\mathsf{StateReveal}$ query.

- Case 1 ($C1$): There is an oracle $\pi_{\hat{B}}^{t^*}$ held by $\hat{B}$, such that $\pi_{\hat{A}}^{s^*}$ and $\pi_{\hat{B}}^{t^*}$ have matching conversations, and the adversary doesn't issue Corrupt($\hat{A}$) and StateReveal($\pi_{\hat{B}}^{t^*}$).
- Case 2 ($C2$): There is no oracle $\pi_{\hat{B}}^{t^*}$ held by $\hat{B}$, such that $\pi_{\hat{A}}^{s^*}$ and $\pi_{\hat{B}}^{t^*}$ have matching conversations, and the adversary doesn't issue Corrupt($\hat{A}$) and Corrupt($\hat{B}$).

Proof of Case $C1$ **(sketch):** The proof proceeds in a sequence of games, following [16].

Game G_0^1**.** This is the original eCK game with adversary $\mathcal{A}_1$ in Case $C1$.

Game G_1^1**.** This game proceeds exactly as Game G_0^1, but the simulator aborts the game if it does not correctly guess the test oracle and its partner. Since the challenger activates d oracles for each ℓ parties. Then the probability that the challenger guesses correctly the test oracle and its partner is at least $1/(\ell^2 d^2)$.

Game G_2^1**.** This game proceeds exactly like the previous game, except that we replace the secret value σ^* of test oracle and its partner oracle with a random one. If there exists adversary $\mathcal{A}_1$ can distinguish game G_2^1 from game G_1^1, then we can use it to construct an efficient algorithm $\mathcal{B}$ to solve the BDDH problem.

Game G_3^1**.** We modify Game G_2^1 to G_3^1 by changing pseudo-random function F to a truly random function RF for test oracle. Note that σ^* is an independent random value. Thus we can use the security of the PRF to argue that this game is indistinguishable from Game G_2^1.

Collecting the advantages from Game G_0^1 to Game G_3^1, we have that

$$\epsilon' \leq \ell^2 d^2 \cdot (\epsilon_{\mathsf{BDDH}} + \epsilon_{\mathsf{PRF}}). \tag{1}$$

Proof of Case $C2$ **(sketch):** Similarly, we proceed in Games G_i^2 with adversary $\mathcal{A}_2$ for Case $C2$ as follows. Let S_i^2 be the event that the adversary wins the security experiment in Game G_i^2 respectively.

Game G_0^2**.** This is the original eCK game with adversary in Case $C2$.

Game G_1^2**.** This game proceeds as the previous game, except that the simulator aborts if the adversary completes an oracle $\pi_{\hat{B}}^t$ such that $H(sid_{\hat{A}}^{(s^*)}) = H(sid_{\hat{B}}^{(t)})$ and $\pi_{\hat{B}}^t$ has no matching conversation to test oracle. Hence we have for any $sid_{\hat{A}}^{(s^*)} \neq sid_{\hat{B}}^{(t)} (t \in [d])$. When the event does occur, we can easily construct algorithm that breaks the collision-resistant hash function H.

Game G_2^2**.** The challenger proceeds as Game G_1^2 but aborts the game if it does not correctly guess the test oracle and its peer . Then the probability that the challenger guesses correctly is at least $1/d\ell^2$.

Game G_3^2**.** We modify game G_2^2 to game G_3^2 by changing the value of $e((B_1{B_3}^{\beta})^{x_1+a_7} \cdot (B_2{B_4}^{\beta})^{x_2+a_8}, h)$ in computation of secret material $\sigma_{\hat{A}}$ for oracles of $\hat{A}$ to $(e(X_1A_5, B_1{B_4}^{\beta}) \cdot e(X_2A_6, B_2{B_5}^{\beta}) \cdot e(X_3A_7, B_3{B_6}^{\beta}))^r$, where r is an

uniform random exponent of group generator $h = g^r$ which is chosen by simulator. This change is possible, even if the party $\hat{B}$ in the game is established by adversary. Since the challenger knows the trapdoor r of h.

Game G_4^2. This game proceeds as the previous game, except that we change the DH tuple $(g, g_1, g_2, A_5, A_6, A_7)$ to a random tuple. If there exists adversary $\mathcal{A}_2$ which can distinguish game G_4^2 from Game G_3^2, then we can use it to construct an efficient algorithm $\mathcal{B}$ to solve the DLIN problem.

Game G_5^2. We modify Game G_5^2 to G_6^2 by changing πPRF function F to a truly random function RF for test oracle. Due to the modifications of Game G_3^2 and G_4^2, we first note that key secret $\sigma_{\hat{A}}^{(s^*)}$ and each the key secret $\sigma_{\hat{B}}^{(t)}$ of oracle $\pi_{\hat{B}}^t$ are pairwise independent. Therefore, we can use the security of the πPRF to argue that this game is indistinguishable from Game G_4^2.

Collecting the advantages from Game G_0^2 to Game G_5^2, we have that

$$\epsilon' \leq \epsilon_{\mathsf{CR}} + \ell^2 d \cdot (\epsilon_{\mathsf{DLIN}} + \epsilon_{\pi\mathsf{PRF}} + 6/p) \tag{2}$$

5 Conclusions

We have presented an efficient eCK-secure key exchange protocols without random oracles (and without NAXOS trick), that the security against chosen public key attacks based on the plain public key assumption (i.e. without KOSK assumption). An open question here is how to construct an eCK secure protocol without πPRF, we leave out this for future work.

Acknowledgments. We would like to thank the anonymous reviewers for their helpful comments.

References

1. 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)
2. Bellare, M., Neven, G.: Multi-signatures in the Plain Public-key Model and a General Forking Lemma. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM Conference on Computer and Communications Security, pp. 390–399. ACM (October 2006)
3. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
4. Blake-Wilson, S., Menezes, A.: Unknown Key-Share Attacks on the Station-to-Station (STS) Protocol. In: Imai, H., Zheng, Y. (eds.) PKC 1999. LNCS, vol. 1560, pp. 154–170. Springer, Heidelberg (1999)
5. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)

6. Canetti, R., Krawczyk, H.: Security Analysis of IKE's Signature-Based Key-Exchange Protocol. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 143–161. Springer, Heidelberg (2002)
7. Diffie, W., Hellman, M.E.: New Directions in Cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
8. LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007)
9. 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)
10. Menezes, A., Ustaoglu, B.: Comparing the Pre- and Post-specified Peer Models for Key Agreement. IJACT 1(3), 236–250 (2009)
11. Moriyama, D., Okamoto, T.: An eCK-secure Authenticated Key Exchange Protocol Without Random Oracles. TIIS 5(3), 607–625 (2011)
12. Okamoto, T.: Authenticated Key Exchange and Key Encapsulation in the Standard Model. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007)
13. Inc. RSA Data Security. Certification Request Syntax Standard. RSA Data Security, Inc. (2000)
14. Sarr, A.P., Elbaz-Vincent, P., Bajard, J.-C.: A New Security Model for Authenticated Key Agreement. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 219–234. Springer, Heidelberg (2010)
15. Sarr, A.P., Elbaz-Vincent, P., Bajard, J.-C.: A Secure and Efficient Authenticated Diffie–Hellman Protocol. In: Martinelli, F., Preneel, B. (eds.) EuroPKI 2009. LNCS, vol. 6391, pp. 83–98. Springer, Heidelberg (2010)
16. Shoup, V.: Sequences of Games: A Tool for Taming Complexity in Security Proofs. Cryptology ePrint Archive, Report 2004/332 (2004), http://eprint.iacr.org/
17. Schäge, S., Jager, T., Kohlar, F., Schwenk, J.: A Standard-Model Security Analysis of TLS-DHE. Cryptology ePrint Archive, Report 2011/219 (2011), http://eprint.iacr.org/
18. Ustaoglu, B.: Comparing *SessionStateReveal* and *EphemeralKeyReveal* for Diffie-Hellman Protocols. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 183–197. Springer, Heidelberg (2009)
19. Yang, Z., Schwenk, J.: Strongly Authenticated Key Exchange Protocol from Bilinear Groups without Random Oracles. Cryptology ePrint Archive, Report 2012/381 (2012), http://eprint.iacr.org/
20. Yoneyama, K., Zhao, Y.: Taxonomical Security Consideration of Authenticated Key Exchange Resilient to Intermediate Computation Leakage. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 348–365. Springer, Heidelberg (2011)

Authenticated Key Exchange with Entities from Different Settings and Varied Groups

Yanfei Guo and Zhenfeng Zhang

SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences
Institute of Software, Chinese Academy of Sciences
{guoyanfei,zfzhang}@is.iscas.ac.cn

Abstract. Authenticated key exchange (AKE) protocol is one of the most widely used cryptographic primitives, and lots of protocols have been proposed either in the certificate-based (cert-based) setting or in the identity-based (id-based) setting. In practical applications, entities from different settings may have the requirement to communicate with each other. Though there are works concentrated on supporting either multiple certification authorities (CAs) or multiple key generation centers (KGCs), very few papers have focused on the interoperability between the two settings. Furthermore, existing approaches are still inadequate in supporting parameters from different algebraic groups introduced by multiple CAs and multiple KGCs.

In this paper, we focus on AKE protocols integrating cert-based settings and id-based settings with varied groups, and propose an AKE protocol where one entity is cert-based and the other is id-based, and the parameters of both entities may come from different groups. An extended AKE security model of [6,22] is proposed to support multiple KGCs and CAs. The proposed protocol is proved to be secure in the extended security model. Finally, we extend the protocol to achieve forward secrecy and resistance to leakage of both ephemeral keys.

1 Introduction

AKE protocols enable two parties to generate a shared, cryptographically strong key while communicating over an insecure network under the complete control of an adversary. This kind of protocol is one of the most widely used cryptographic primitives; indeed, agreement on a shared key is necessary for the realization of "higher-level" tasks such as encryption and message authentication.

Diffie and Hellman [10] proposed the first key agreement protocol, which became known as the Diffie-Hellman (DH) protocol. After that, many cert-based variants [17,15,16,19] providing additional properties have been suggested. In these protocols, both participants use public/private key pairs binding to their certificates in the protocol for authentication and key establishment.

In 1984, Shamir [20] introduced identity-based technology to construct AKE protocols. That is, both participants use identity-based asymmetric key pairs in

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 276–287, 2012.

the protocol for authentication and key establishment. Since then, researchers proposed lots of id-based AKE protocols (e.g., [13,21,23,9,7,14]).

Apparently, both parties need to be the same type in almost all the above key agreement protocols. However, in some cases, parties of different types may also want to share a session key to communicate with each other. Consider the situation that Alice has a certificate and wants to securely communicate with an entity Bob who belongs to a company. After Alice sending a request, Bob may either answer with a public key along with a certificate or inform that "Bob" is part of an ID-based system. Their static keys may be generated by different KGCs with different algebraic groups, or be bound to certificates issued by different CAs. Chain and cross certifications allow users to trust different CAs to interact. Similarly, there are methods to extend identity-based solutions across multiple KGCs. However, very few works have been done on the interoperability between the two settings.

A straightforward method to solve this problem is that parties register certificates from CAs as well as ask for id-based keys from KGCs to accommodate all peers. However, this approach is not practical because there are multiple CAs and multiple KGCs and maintaining one secret key for each CA and each KGC needs a large amount of memory and maintenance costs. As a result, the integrating protocols (i.e., one party of the protocol is cert-based and the other one is id-based) are needed in this occasion.

One can design an integrating protocol using a Diffie-Hellman protocol together with the authenticators proposed by Canetti and Krawczyk [5], that is, one authenticator with an id-based signature and the other with a cert-based signature. The protocols using this approach are not resistant to leakage of ephemeral keys. It is also possible to obtain an integrating protocol by using the generic key encapsulation mechanisms (KEM) techniques proposed by Boyd et al. [4]. In their construction, one of the KEMs can be id-based and the other can be cert-based. However, to provide forward secrecy, the parties have to run an extra ephemeral Diffie-Hellman protocol along with the basic protocol and it is unclear whether ephemeral leakage resilience can be achieved.

In 2003, Chen and Kudla [8] introduced the concept of communication through "different domains", that is, the communicating parties' keys are generated by different KGCs, and proposed an id-based protocol that allows two parties to communicate through "different domains". They solved the problem when the communicating parties are both id-based and their keys are generated by different KGCs. In 2005, McCullagh and Barreto [18] proposed a more efficient construction to communicate through "different domains" like [8].

In 2011, Chatterjee, Menezes, et al. [6] proposed a generic three-pass key agreement protocol based on a certain kind of trapdoor one-way function family, and presented three cert-based instantiations. An interesting feature of their discrete-log instantiation is that parties can use different groups (e.g., different elliptic curves). However, they only considered the cert-based instantiations, and did not discuss the extensions to id-based settings.

Ustaoğlu [22] proposed a collection of integrating protocols which requires that the participants use parameters from the same algebraic group. Thus, the protocols can not be used to multiple-CAs and multiple-KGCs when they use parameters from different groups.

In 2012, Fujioka et al. [12] proposed a generic construction of AKE from a KEM which can allow the initiator and the responder to use different KEMs under different groups. However, if the randomnesses used in the KEM are compromised, the shared session key will be revealed. Although they use the output of the twisted PRF trick as the randomness of KEMs to avoid the attacks caused by leakage of ephemeral keys, but sometimes the PRF output may be compromised as easy as ephemeral keys.

For the security models of AKE, traditional security models [2,3,1,5,16] don't consider the occasion when there are many KGCs and many CAs and participants of the protocol are from different settings. Ustaoğlu [22] defined security model for integrating protocols but actually there is only one KGC and one CA in the model. Meanwhile, the traditional way [2,3,1] of capturing explicit authentication is to use the event "no matching", and is independent of the definition of session key's indistinguishability property. Usually, one have to prove the two properties separately.

1.1 Our Contributions

In this paper, we first consider the security model of AKE with entities from different settings, and extend the security model of [22] to capture multiple CAs and multiple KGCs. Furthermore, we find that the traditional way of capturing explicit authentication in key establishment protocols can be facilitated and the approach to define explicit authentication in [6] can be more accurate. We modify this model slightly to capture explicit authentication in an easier way than the traditional way. The explicit authentication and the indistinguishability of session key are defined together in our security model.

Next, we propose an explicitly authenticated key agreement protocol with one party being id-based while the other being cert-based. The protocol is proved to be secure in the extended model under reasonable assumptions, and satisfies the demands of supporting multiple CAs and multiple KGCs. Thus, parties participating the protocol can use parameters from different groups. Furthermore, the efficiency if the proposed protocol is comparable with other related works.

After that, we extend our protocol to achieve eCK variant security with higher efficiency. The proposed integrating protocols can be used when parties are from different settings using parameters of different groups. However, as protocols proposed in [6], the basic protocol doesn't have forward secrecy nor resistance to leakage of both ephemeral keys. So we show how to extend the basic protocols to achieve forward secrecy and resistance to leakage of both ephemeral keys. We compare our protocols with related protocols in section 6.

2 Preliminaries

Throughout, the following notations are used in this paper:

- $\mathcal{G}$ denotes a multiplicatively written cyclic group of prime order q_1 generated by G, $|q_1|$ is the bit length of q_1; $\mathcal{G}^*$ is the set of non-identity elements in $\mathcal{G}$.
- The identity of a party A is denoted by $\hat{A}$ ($\hat{A}$ is supposed to contain A). If $\hat{A} \neq \hat{B}$, we suppose that no substring of $\hat{A}$ equals $\hat{B}$.
- Let $\hat{e} : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_T$ be a bilinear pairing defined as in [7], where $\mathcal{G}_1$ is an additive group of points over an elliptic curve of prime order q_2, and $\mathcal{G}_T$ is a multiplicative subgroup of the multiplicative group of a finite field.
- $H : \{0,1\}^* \to \{0,1\}^\lambda$, $H_1 : \{0,1\}^* \to \mathcal{G}_1$ are cryptographic hash functions.
- The symbol $\in_R$ stands for "chosen uniformly at random in".

Assumption 1 (GDHD). *The DHD problem in a cyclic group $\mathcal{G}$ of prime order q_1 is the problem of determining $g^{u/v}$, given $g, g^u, g^v \in_R \mathcal{G}$. The gap DHD (GDHD) assumption asserts that the DHD problem is intractable even when the solver is given a Decision DHD (DDHD) oracle which, on input a quadruple (h, h^a, h^b, h^c), determines whether $c \equiv a/b \mod q_1$.*

Assumption 2 (GBDH). *Let $\mathcal{P}$ be a generator of $\mathcal{G}_1$, $\hat{e}(\mathcal{P}, \mathcal{P})$ be a generator of $\mathcal{G}_T$. The BDH problem in group $\mathcal{G}_1$ of prime order q_2 is the problem of computing $\hat{e}(\mathcal{P}, \mathcal{P})^{abc}$, given instance : $(\mathcal{P}, a\mathcal{P}, b\mathcal{P}, c\mathcal{P})$ for some $a, b, c \in Z_{q_2}^*$. The gap BDH (GBDH) assumption asserts that the BDH problem is intractable even when the solver is given a Decision BDH (DBDH) oracle which, on input $(\mathcal{P}, a\mathcal{P}, b\mathcal{P}, c\mathcal{P}, \hat{e}(\mathcal{P}, \mathcal{P})^d)$, determines whether $d \equiv abc \mod q_2$.*

3 Model

3.1 Our Security Model

Session and its Execution. The AKE security experiment involves multiple honest parties and an adversary $\mathcal{A}$ connected via an unauthenticated network. In our model, we identify a party $\hat{P}$ with a probabilistic Turing machine. The set of all parties is divided into two: CP(cert-based parties) and IP(id-based parties). We suppose there are $n_{CA} \leq \mathcal{L}_{CA}(|q|)$ CAs trusted by cert-based parties. These CAs are identified with numbers $1, 2, \ldots, n_{CA}$ and every CA can establish $n_{Cert} \leq \mathcal{L}_{Cert}(|q|)$ cert-based parties. In sum, there are $n_1 = n_{CA} \cdot n_{Cert}$ cert-based parties. Meanwhile, there are $n_{KGC} \leq \mathcal{L}_{KGC}(|q|)$ KGCs trusted by id-based parties. These KGCs are identified with numbers $1, 2, \ldots, n_{KGC}$ and every KGC can establish $n_{ID} \leq \mathcal{L}_{ID}(|q|)$ id-based parties. Altogether, there are $n_2 = n_{KGC} \cdot n_{ID}$ id-based parties. Here, $\mathcal{L}_{CA}, \mathcal{L}_{Cert}, \mathcal{L}_{KGC}, \mathcal{L}_{ID}$ are polynomials. Certificates issued by different CAs and secret keys issued by different KGCs may correspond to public keys in different groups. A session is an instance of the considered protocol, run at a party. The adversary which is also a probabilistic

polynomial time machine selects parties to execute sessions and selects an order in which the sessions will be executed.

Session Creation. A session of $\hat{P}_i$ (with peer $\hat{P}_j$) can be created with parameters $(\hat{P}_i, \hat{P}_j)$ or $(\hat{P}_i, \hat{P}_j, \mathcal{R}, In)$, where In is an incoming message, supposed from $\hat{P}_j$; $\hat{P}_i$ is the initiator if the parameter is $(\hat{P}_i, \hat{P}_j)$, otherwise a responder.

Session Initiator. If $\hat{P}_i$ is the session initiator then $\hat{P}_i$ creates a separate session state where session-specific short-lived data is stored, and prepares a reply $Out = (ePK_{\hat{P}_i}, OtherInfo)$, where $ePK_{\hat{P}_i}$ is $\hat{P}_i$'s ephemeral public key, and $OtherInfo$ is additional data that the protocol may specify. The session is labeled active and identified via a (temporary and incomplete) session identifier $sid = (\hat{P}_i, \hat{P}_j, \mathcal{I}, ePK_{\hat{P}_i})$, $\hat{P}_i \in \{0,1\}^*$ is the identity of the party executing the session, $\hat{P}_j$ is the identity of the other party participating, $\mathcal{I}$ means that $\hat{P}_i$ is the initiator of the protocol. The outgoing message is $(\hat{P}_j, \hat{P}_i, \mathcal{R}, Out)$.

Session Responder. If $\hat{P}_i$ is the session responder, $\hat{P}_i$ creates a separate session state and prepares a reply Out that includes $ePK_{\hat{P}_i}$ which is the ephemeral public key of $\hat{P}_i$. The session is labeled active and identified via a session identifier $sid = (\hat{P}_i, \hat{P}_j, \mathcal{R}, ePK_{\hat{P}_i}, ePK_{\hat{P}_j})$, where $ePK_{\hat{P}_j}$ is the ephemeral public key in the incoming message In. The outgoing message is $(\hat{P}_j, \hat{P}_i, \mathcal{I}, Out)$.

Session Update. A party $\hat{P}_i$ can be activated to update a session via an incoming message of the form $(\hat{P}_i, \hat{P}_j, role, ePK_{\hat{P}_i}, ePK_{\hat{P}_j}, In)$, where $role \in \{\mathcal{I}, \mathcal{R}\}$. Upon receipt of this message, $\hat{P}_i$ checks that he owns an active session with identifier $sid = (\hat{P}_i, \hat{P}_j, role, ePK_{\hat{P}_i}, ePK_{\hat{P}_j})$. If ephemeral keys are chosen uniformly at random from the appropriate domain, except with negligible probability $\hat{P}_i$ can own at most one such session. If no such session exists then the message is rejected; otherwise $\hat{P}_i$ follows the protocol specifications. Initiator $\hat{P}_i$ can also be activated to update a session with incomplete session identifier $(\hat{P}_i, \hat{P}_j, \mathcal{I}, ePK_{\hat{P}_i})$ with an incoming message of the form $(\hat{P}_i, \hat{P}_j, \mathcal{I}, ePK_{\hat{P}_i}, ePK_{\hat{P}_j}, In)$ where In is any message specified by the protocol. In this case $\hat{P}_i$ performs the required validations before updating the session identifier to $(\hat{P}_i, \hat{P}_j, \mathcal{I}, ePK_{\hat{P}_i}, ePK_{\hat{P}_j})$.

Completed Sessions. When the protocol specifies that no further messages will be received, the session accepts a session key and marks itself as completed.

Aborted Sessions. A protocol may require parties to perform some checks on incoming messages. For example, a party may be required to perform some form of public key validation or verify a message authentication tag. If a party is activated to create a session with an incoming message that does not meet the protocol specifications, then that message is rejected and no session is created. If a party is activated to update an active session with an incoming message that does not meet the protocol specifications, then the party deletes all information specific to that session (including the session state and the session key if it has been computed) and aborts the session. Abortion occurs before the session identifier is updated.

Matching Sessions. If ephemeral keys are selected at random on a per-session basis, session identifiers are unique except with negligible probability. Party $\hat{P}_i$ is said to be the owner of a session $(\hat{P}_i, \hat{P}_j, role, *, *)$, where $role \in \{\mathcal{I}, \mathcal{R}\}$. For a session $(\hat{P}_i, \hat{P}_j, role, *, *)$ we call $\hat{P}_j$ the session peer; $\hat{P}_i$ and $\hat{P}_j$ are referred to as the communicating parties. Let sid be a session with complete session identifier $(\hat{P}_i, \hat{P}_j, role, ePK_{\hat{P}_i}, ePK_{\hat{P}_j})$. A session $\overline{sid}$ with session identifier $(\overline{\hat{P}_i}, \overline{\hat{P}_j}, \overline{role}, \overline{ePK_{\overline{\hat{P}_i}}}, \overline{ePK_{\overline{\hat{P}_j}}})$ is said to be matching to sid if $\hat{P}_i = \overline{\hat{P}_j}, \hat{P}_j = \overline{\hat{P}_i}$, $role \neq \overline{role}$, $ePK_{\hat{P}_i} = \overline{ePK_{\overline{\hat{P}_j}}}$ and $ePK_{\hat{P}_j} = \overline{ePK_{\overline{\hat{P}_i}}}$. A session sid with incomplete session identifier $(\hat{P}_i, \hat{P}_j, \mathcal{I}, ePK_{\hat{P}_i})$ is matching to any session $\overline{sid} = (\overline{\hat{P}_i}, \overline{\hat{P}_j}, \overline{\mathcal{R}}, \overline{ePK_{\overline{\hat{P}_i}}}, \overline{ePK_{\overline{\hat{P}_j}}})$ with $\hat{P}_i = \overline{\hat{P}_j}$, $\hat{P}_j = \overline{\hat{P}_i}$ and $ePK_{\hat{P}_i} = \overline{ePK_{\overline{\hat{P}_j}}}$. Since ephemeral keys are selected at random on a per-session basis, only sessions with incomplete session identifiers can have more than one matching session.

Adversarial Capabilities. The adversary is allowed to make the following oracle queries.

- **Establish**$(party, type, i)$**.** $type \in \{0, 1\}$ identifies the kind of the party, $type = 0$ means the party to be established is cert-based, $type = 1$ means the party is id-based. i denotes the identity of the corresponding CA or KGC. For cert-based type, the adversary $\mathcal{A}$ registers a static key on behalf of a party to the i-th CA; for id-based parties $\mathcal{A}$ register an identifier and obtain the corresponding private key from the i-th KGC. As the adversary controls communications, from then the party is supposed totally controlled by $\mathcal{A}$. A party against which this query is not issued is said to be honest.
- **Send**(sid, m)**.** $\mathcal{A}$ sends any message m to session sid established by one party. The oracle will respond according to the protocol description.
- **SessionkeyReveal**(sid)**.** The adversary gets the session key of the session sid after the session has completed via this query. If the session has not completed, output $\perp$.
- **StaticKeyReveal**$(party)$**.** Adversary gets the long-term private key of the party $party$ without full control of the party.
- **EphemeralKeyReveal**(sid)**.** This query returns the ephemeral exponent chosen by the corresponding party of the specific session sid.
- **MasterKeyReveal**(KGC)**.** The adversary obtains the master secret key used by the KGC to generate private keys. Note that there may be lots of KGCs, and we only obtain the secret key of the KGC being asked.
- **Test**(sid)**.** This query can be asked by the adversary only once during the game. After making this query, the challenger randomly chooses a bit $b \in \{0, 1\}$, if $b = 0$, he returns the real key, otherwise, he returns a random key chosen uniformly from the session key space. The adversary wins the game if he guesses correctly whether the key is random or not.

Security of AKE Protocols. We first define the session freshness notion. Test queries can only be performed on fresh sessions.

Remark: The adversary can obtain a static private key corresponding to an identity either by explicitly querying for it or by obtaining the corresponding KGC master private key. Taking the same method of [22], in the following definition, we assume that RevealMasterKey(KGC) implies that RevealStaticKey has been issued against the corresponding identities.

Definition 1 (Session Freshness). *Let sid be a completed session held by a party $\hat{A}$ with other party $\hat{B}$, and both parties $\hat{A}$ and $\hat{B}$ are honest. Meanwhile, $\hat{B}$ supposedly engaged in a session sid^*, such that sid and sid^* (if it exists) are matching. Then the session sid is said to be fresh if none of the following conditions hold:*

1. $\mathcal{A}$ issues a SessionKeyReveal query on sid or sid^* (if sid^* exists).

2. sid^* exists and $\mathcal{A}$ issued one of the following:

- *(a) Both StaticKeyReveal($\hat{A}$) and EphemeralKeyReveal(sid).*
- *(b) Both StaticKeyReveal($\hat{B}$) and EphemeralKeyReveal(sid^*).*
- *(c) Both StaticKeyReveal($\hat{A}$) and StaticKeyReveal($\hat{B}$).*
- *(d) Both EphemeralKeyReveal(sid) and EphemeralKeyReveal(sid^*).*

3. sid^* doesn't exist and $\mathcal{A}$ issued one of the following:

- *(a) EphemeralKeyReveal(sid).*
- *(b) StaticKeyReveal($\hat{B}$) before session sid completes.*

Through the game, adversary makes all the queries defined above. At one point he issues a Test query to a fresh session, after that he can also issue queries but must insure that the test session remains fresh. The goal of the adversary $\mathcal{A}$ is to guess a bit b' such that $b' = b$ where b is the private bit involved in Test query. The advantage of the adversary $\mathcal{A}$ is defined by $Adv^k_{\mathcal{P}}(\mathcal{A}) = |2Pr[b' = b] - 1|$.

Definition 2. *An AKE protocol $\mathcal{P}$ is secure if both the conditions hold,*

1. If two honest parties complete matching sessions, except with negligible probability, they both compute the same session key.

2. $Adv^k_{\mathcal{P}}(\mathcal{A})$ is negligible for any polynomially bounded adversary $\mathcal{A}$.

3.2 Relations to Other Security Models

In many security models for AKE protocols, indistinguishability and mutual authentication are usually defined separately. The traditional explicit authentication [2,3,1] can be expressed as "*There should be no such fresh session that the session completes, but has no partner session*".

In our definition, we use a different freshness definition to capture the property of explicit authentication. That is, in 3(b), "if sid^* doesn't exist, StaticKeyReveal ($\hat{B}$) should not be issued before session *sid* completes". This security model captures explicit authentication and indistinguishability simultaneously.

To facilitate comparison, in the following proposition, we assume that the adversary's abilities and the definitions of matching sessions in our security

model are the same as in the traditional model. And the main difference between the two models is the freshness definition in 3(b). That is, "$\mathcal{A}$ should not issues StaticKeyReveal($\hat{B}$) before session *sid* completes" and "$\mathcal{A}$ should not issues StaticKeyReveal($\hat{B}$)" relatively.

Claim. Our security definition implies the traditional "mutual authentication" property.

In fact, if a key establishment protocol doesn't have mutual authentication property, a fresh and completed session without matching session exists. Without loss of generality, we assume such a session is identified by *sid* with owner $\hat{A}$ and intended peer $\hat{B}$ (It is easy for the adversary to find this session because it is the adversary who makes all the queries). By the freshness definition in traditional model, adversary $\mathcal{A}$ doesn't issue StaticKeyReveal($\hat{B}$) query, and he chooses the ephemeral key of $\hat{B}$. However, *sid* is still fresh in our security model even though $\mathcal{A}$ makes a StaticKeyReveal($\hat{B}$) query. Hence, if $\mathcal{A}$ tests the session *sid*, he can output a guess correctly. Therefore, a protocol without mutual authentication property in the traditional model can't be secure in our security model.

4 Our Protocol

Without loss of generality, we use the Discrete Log setting for cert-based parties. As summarized in [7], there are four types of pairings and two kinds of extract algorithms in id-based setting. We choose the *type*1 parings to describe the id-based parties, and take extract 1 algorithms as an example to present the integrating protocol. The protocol can be easily extended to other types of pairings and protocol of extract 2 algorithm case will be given in the full version.

Without loss of generality, we describe situations when the initiator ($\hat{A}$) is id-based and the responder ($\hat{B}$) is cert-based. The symmetric setting can be extended trivially. The following notations are used in our Extract 1 case protocol.

Algorithm 1 (Extract 1 [7]). *Given the pairing parameters, an identity string ID_A for a user A, a hash-function $H_1 : \{0,1\}^* \to \mathcal{G}_1$, the master private key $s \in \mathbb{Z}_{q_2}^*$, and the master public key $P_0 = s\mathcal{P} \in \mathcal{G}_1$, the algorithm computes $Q_A = H_1(ID_A) \in \mathcal{G}_1$ and $d_A = sQ_A \in \mathbb{G}_1$. The values Q_A and d_A will be used as the public and private key pair corresponding to A's identity ID_A.*

Let $\hat{A}$ be an id-based party with public key Q_A and private key d_A defined in the above Extract 1 algorithm. Let $\hat{B}$ be a cert-based party with private key $b \in \mathbb{Z}_{q_1}^*$ and public key B satisfying $B = g^b$. The protocol with Extract 1 case is described in table 1. The detailed specification of protocol Π_1 is as follows.

1. $\hat{A}$ chooses $x \in_R \mathbb{Z}_{q_1}^*$, computes $X = G^x, X_A = B^x$, destroy x. $\hat{A}$ sends $\hat{B}, \hat{A}, \mathcal{R}, X_A$ to $\hat{B}$ and initialize the session identifier to $(\hat{A}, \hat{B}, \mathcal{I}, X_A)$.
2. After receiving the message from $\hat{A}$, $\hat{B}$ first verifies that $X_A \in \mathcal{G}^*$, then chooses $y \in_R \mathbb{Z}_{q_2}^*$, computes $Y = \hat{e}(P_0, yQ_A)$, $Y_B = y\mathcal{P}$. After that, $\hat{B}$ computes $X = (X_A)^{1/b}$, $(\kappa_m, \kappa) = H(X, Y, \hat{A}, \hat{B}, X_A, Y_B)$. Compute $tag_B =$

Table 1. Protocol Π_1: The Extract 1 case

$\hat{A}(Q_A, d_A)$		$\hat{B}(B, b)$
$x \in_R \mathbb{Z}_{q_1}^*, X = G^x, X_A = B^x$ Destroy x.		
Initialize sid to $(\hat{A}, \hat{B}, \mathcal{I}, X_A)$	$\xrightarrow{\hat{B}, \hat{A}, \mathcal{R}, X_A}$	Verify that $X_A \in \mathcal{G}^*$ $y \in_R \mathbb{Z}_{q_2}^*, Y = \hat{e}(P_0, yQ_A), Y_B = y\mathcal{P}$ Compute $X = (X_A)^{1/b}$, $(\kappa_m, \kappa) = H(X, Y, \hat{A}, \hat{B}, X_A, Y_B)$ $tag_B = MAC_{\kappa_m}(\mathcal{R}, \hat{B}, \hat{A}, Y_B, X_A)$ $tag_A = MAC_{\kappa_m}(\mathcal{I}, \hat{A}, \hat{B}, X_A, Y_B)$, Destroy y, X, Y and κ_m
Verify that $Y_B \in \mathcal{G}_1^*$ Compute $Y = \hat{e}(Y_B, d_A)$ $(\kappa_m, \kappa) = H(X, Y, \hat{A}, \hat{B}, X_A, Y_B)$ $tag_B \stackrel{?}{=} MAC_{\kappa_m}(\mathcal{R}, \hat{B}, \hat{A}, Y_B, X_A)$ $tag_A = MAC_{\kappa_m}(\mathcal{I}, \hat{A}, \hat{B}, X_A, Y_B)$	$\xleftarrow{\hat{A}, \hat{B}, \mathcal{I}, X_A, Y_B, tag_B}$	Set sid to $(\hat{B}, \hat{A}, \mathcal{R}, Y_B, X_A)$
Update sid to $(\hat{A}, \hat{B}, \mathcal{I}, X_A, Y_B)$ accept κ, complete	$\xrightarrow{\hat{B}, \hat{A}, \mathcal{R}, Y_B, X_A, tag_A}$	$tag_A \stackrel{?}{=} MAC_{\kappa_m}(\mathcal{I}, \hat{A}, \hat{B}, X_A, Y_B)$ accept κ, complete

$MAC_{\kappa_m}(\mathcal{R}, \hat{B}, \hat{A}, Y_B, X_A)$, $tag_A = MAC_{\kappa_m}(\mathcal{I}, \hat{A}, \hat{B}, X_A, Y_B)$, store tag_A and destroy y, X, Y, κ_m. Send $\hat{A}, \hat{B}, \mathcal{I}, X_A, Y_B, tag_B$ to $\hat{A}$ and set the session identifier to $(\hat{B}, \hat{A}, \mathcal{R}, Y_B, X_A)$.

3. Receiving the message from $\hat{B}$, $\hat{A}$ computes $Y = \hat{e}(Y_B, d_A)$, $(\kappa_m, \kappa) = H(X, Y, \hat{A}, \hat{B}, X_A, Y_B)$, verifies $tag_B = MAC_{\kappa_m}(\mathcal{R}, \hat{B}, \hat{A}, Y_B, X_A)$, if verification is passed, computes $tag_A = MAC_{\kappa_m}(\mathcal{I}, \hat{A}, \hat{B}, X_A, Y_B)$, and sends $\hat{B}, \hat{A}, \mathcal{R}, Y_B, X_A, tag_A$ to $\hat{B}$. After that $\hat{A}$ updates the session's session identifier to $(\hat{A}, \hat{B}, \mathcal{I}, X_A, Y_B)$, accepts κ, and completes the session.
4. $\hat{B}$ verifies $tag_A = MAC_{\kappa_m}(\mathcal{I}, \hat{A}, \hat{B}, X_A, Y_B)$, if the verification is passed, $\hat{B}$ accepts κ and completes the session.

Theorem 1. *Suppose that the GDHD assumption for $\mathcal{G}$ holds; the GBDH assumption for $\mathcal{G}_1, \mathcal{G}_T$ holds ; the MAC scheme is secure; H and H_1 are random oracles. Then the above key agreement protocol is secure in our security model.*

The proof of this theorem will be given in the full version of the paper.

Remark: In 2010, Fiore and Gennaro [11] presented an id-based key agreement protocol without pairings, which uses a new type of extract algorithm. Our protocol can be easily extended to such kind of extract algorithm.

5 Forward Security and Resistance to Leakage of Ephemeral Keys

Both protocols above have neither the properties of forward security nor resistance to leakage of ephemeral keys of both sessions. In this subsection, we show

Table 2. Protocol Π_2: The Extract 1 case with enhanced security

$\hat{A}(Q_A, d_A)$		$\hat{B}(B, b)$
$x \in_R \mathbb{Z}_q^*, X = G^x, X_A = B^x$ $\pi = H_3(d_A, r), \Pi = G^\pi, S_2 = B^\pi$ Destroy π, r.		
Initialize sid to $(\hat{A}, \hat{B}, \mathcal{I}, X_A, \Pi)$	$\xrightarrow{\hat{B}, \hat{A}, \mathcal{R}, X_A, \Pi}$	Verify that $X_A, \Pi \in \mathcal{G}^*$ $y \in_R \mathbb{Z}_q^*, Y = \hat{e}(S, yQ_A), Y_B = y\mathcal{P}$ Compute $Y_1 = G^y, S_2 = \Pi^b$ Compute $X = (X_A)^{1/b}, S_1 = (X)^y$ $(\kappa_m, \kappa) = H(X, Y, S_1, S_2,$ $\hat{A}, \hat{B}, X_A, \Pi, Y_B, Y_1)$ $tag_B = MAC_{\kappa_m}(\mathcal{R}, \hat{B}, \hat{A}, Y_B, Y_1,$ $X_A, \Pi), tag_A = MAC_{\kappa_m}(\mathcal{I},$ $\hat{A}, \hat{B}, X_A, \Pi, Y_B, Y_1)$ Destroy y, y_1, X, Y, S_1, S_2 and κ_m
Verify that $Y_B, \in \mathcal{G}_1^*, Y_1 \in \mathcal{G}^*$ Compute $Y = \hat{e}(Y_B, d_A)$, $S_1 = Y_1^x$ $(\kappa_m, \kappa) = H(X, Y, S_1, S_2,$ $\hat{A}, \hat{B}, X_A, \Pi, Y_B, Y_1)$ Verify tag_B, Compute tag_A Update sid to $(\hat{A}, \hat{B}, \mathcal{I}, X_A, \Pi, Y_B, Y_1)$ accept κ, complete	$\xleftarrow{\hat{A}, \hat{B}, \mathcal{I}, X_A, \Pi, Y_B, Y_1, tag_B}$ $\xrightarrow{\hat{B}, \hat{A}, \mathcal{R}, Y_B, Y_1, X_A, \Pi, tag_A}$	set sid to $(\hat{B}, \hat{A}, \mathcal{R}, Y_B, Y_1, X_A, \Pi)$ Verify tag_A accept κ, complete

how to extend our protocol to be forward secure and resistant to leakage of both ephemeral keys. The protocol Π_2 is described in Table 2. Protocols in other extract algorithm case can be easily extended to achieve the security properties using the same method.

Here, we let $|q_1| = |q_2| = q$. In this precondition, $\mathcal{G}$ and $\mathcal{G}_1$ have the same order, but they are still different groups: one is a multiplication group while the other is a addition group. Situation of the case when the order of these two groups are different can be easily extended.

Similar to [22], we can define the eCK variant of our security model. The protocol Π_2 can prove secure in the eCK variant model under the GDHD, GDH and GBDH assumptions, which will be given in the full paper.

6 Comparisons with Related Protocols

In table 3, we compare our protocols with other related protocols in terms of computation cost of id-based party ($id-based$) and cert-based party ($cert-based$) respectively, and the scopes protocols can be used to support varied groups, multi-CAs and multi-KGCs. We consider exponent (e), bilinear map operations (p), multiplication operation of elliptic curve groups (m) and exponentiation in $\mathcal{G}_T$ (E) for computation cost. We use y or n to denote whether the protocol supports varied-groups, multi-CAs, multi-KGCs or not.

Table 3. The comparisons with integrating protocols

protocols	party	computation cost	varied-groups	multi-CAs	multi-KGCs
SCK-3 [8]	$id-based(\hat{A}/\hat{B})$	$2P+3m$	n	n	y
MB-3 [18]	$id-based(\hat{A}/\hat{B})$	$1P+3m$	n	n	y
B11 [22]	$id-based(\hat{A})$	$2P+5m$	n	n	n
	$cert-based(\hat{B})$	$2P+4m$			
protocol Π_1	$id-based(\hat{A})$	$1P+2e$	y	y	y
	$cert-based(\hat{B})$	$1P+1e+2m$			
protocol Π_2	$id-based(\hat{A})$	$1P+5e$	y	y	y
	$cert-based(\hat{B})$	$1P+4e+2m$			

Table 3 shows that protocols Π_1 have obvious advantage in computation cost. The protocol Π_2 is more efficient than the protocol in [22]. Besides, the participants of our protocols can use parameters from different groups and get static keys from different KGCs.

7 Conclusion and Future Work

In this paper we propose an explicitly authenticated key agreement protocol with one party being identity-based and the other being certificate-based. Participants of the protocol can use parameters from different groups. There can also be multiple CAs issuing certification to certificate-based parties and multiple KGCs generating static keys for identity-based parties. We extend the security models of [6,22] to capture multiple CAs, multiple KGCs and explicit authentication. The protocol's security is proved in the extended model. We also extend the protocol to be forward secure and resistant to leakage of both session's ephemeral keys. Comparison shows that our protocols satisfy higher demands for usability and have comparable efficiency.

Acknowledgement. The work is supported by the National Natural Science Foundation of China (No.61170278), the National Basic Research Program (973) of China (No.2012CB315804), and the NSFC of China (No.91118006).

References

1. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure against Dictionary Attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
2. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
3. Bellare, M., Rogaway, P.: Provably secure session key distribution: the three party case. In: STOC 1995, pp. 57–66. ACM, New York (1995)
4. Boyd, C., Cliff, Y., Gonzalez Nieto, J.M., Paterson, K.G.: Efficient One-Round Key Exchange in the Standard Model. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 69–83. Springer, Heidelberg (2008)
5. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)

6. Chatterjee, S., Menezes, A., Ustaoglu, B.: A Generic Variant of NIST's KAS2 Key Agreement Protocol. In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 353–370. Springer, Heidelberg (2011)
7. Chen, L., Cheng, Z., Smart, N.: Identity-based key agreement protocols from pairings. Int. J. Inf. Secur. 6, 213–241 (2007)
8. Chen, L., Kudla, C.: Identity based authenticated key agreement from pairings. In: Proc. 16th IEEE Computer Security Foundations Workshop, pp. 219–233 (2003)
9. Chow, S.S.M., Choo, K.-K.R.: Strongly-Secure Identity-Based Key Agreement and Anonymous Extension. In: Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 203–220. Springer, Heidelberg (2007)
10. Diffie, W., Hellman, M.: New directions in cryptography. IEEE T. Inform. Theory 22(6), 644–654 (1976)
11. Fiore, D., Gennaro, R.: Making the Diffie-Hellman Protocol Identity-Based. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 165–178. Springer, Heidelberg (2010)
12. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly Secure Authenticated Key Exchange from Factoring, Codes, and Lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 467–484. Springer, Heidelberg (2012)
13. Günther, C.G.: An Identity-Based Key-Exchange Protocol. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 29–37. Springer, Heidelberg (1990)
14. Huang, H., Cao, Z.: An id-based authenticated key exchange protocol based on bilinear diffie-hellman problem. In: ASIACCS 2009, pp. 333–342. ACM (2009)
15. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
16. LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007)
17. Law, L., Menezes, A., Qu, M., Solinas, J., Vanstone, S.: Some new key agreement protocols providing mutual implicit authentication. In: SAC 1995, pp. 22–32 (1995)
18. McCullagh, N., Barreto, P.S.L.M.: A New Two-Party Identity-Based Authenticated Key Agreement. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 262–274. Springer, Heidelberg (2005)
19. Sarr, A.P., Elbaz-Vincent, P., Bajard, J.-C.: A New Security Model for Authenticated Key Agreement. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 219–234. Springer, Heidelberg (2010)
20. 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)
21. Smart, N.: Identity-based authenticated key agreement protocol based on weil pairing. Electron. Lett. 38(13), 630–632 (2002)
22. Ustaoğlu, B.: Integrating identity-based and certificate-based authenticated key exchange protocols. Int. J. Inf. Secur. 10, 201–212 (2011)
23. Wang, Y.: Efficient identity-based and authenticated key agreement protocol. Cryptology ePrint Archive, Report 2005/108 (2005)

On Capabilities of Hash Domain Extenders to Preserve Enhanced Security Properties

Mohammad Reza Reyhanitabar and Willy Susilo

Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
Faculty of Informatics
University of Wollongong, Australia
{rezar,wsusilo}@uow.edu.au

Abstract. In this paper, we study property preservation capabilities of several domain extension transforms for hash functions with respect to multiple enhanced security notions. The transforms investigated include MD with strengthening padding (sMD), HAIFA, Enveloped Shoup (ESh) and Nested Linear Hash (nLH). While the first two transforms and their straightforward variants are among the most popular ones in practical hash designs including several SHA-3 candidates, the last two transforms (i.e. ESh and nLH) are mainly of a theoretical interest in the analysis of multi-property-preservation (MPP) capabilities of hash domain extenders. The security notions considered are the enhanced (or strengthened) variants of the traditional properties (collision resistance, second-preimage resistance, and preimage resistance) for the setting of dedicated-key hash functions. The results show that most of these enhanced security notions *are not* preserved by the investigated domain extenders. This might seem a bit disappointing from a provable security viewpoint, that advocates MPP paradigm (i.e. the more properties preserved simultaneously by a transform the more popular is the transform from a theoretical viewpoint); however, it is worth stressing that the mere fact that a domain extender fails to preserve a property P does *not* imply that a hash function built upon it is insecure. Rather, it just implies that security of the hash function in the sense of the property P cannot be deduced based on the assumption that the underlying compression function possesses P.

Keywords: hash functions, security properties, domain extension, multi-property-preservation.

1 Introduction

A cryptographic hash function converts a variable-length input message into a short (typically) fixed-length digest, while providing some security properties. Hash functions are used in a vast variety of cryptographic applications, both in symmetric key and public key cryptosystems; for example, in message authentication codes and digital signatures. Designing multi-purpose cryptographic hash

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 288–299, 2012.

functions has turned out to be a challenging task; it has been considered for decades [8, 7], yet remains a highly active and interesting area of research, taking a lot of effort from the cryptographic community to come up with a new hash function standard through the current SHA-3 competition [13].

A major difficulty in designing a "secure" cryptographic hash function lies in the fact that hash functions are often expected to satisfy several different security properties depending on the security requirements of applications employing them [14, 4, 1, 18]. In regard to security notions for cryptographic hash functions, two theoretical aspects are of great interest; namely, formalization of security notions and analysis of implications and separations between different properties (there are a few works in this line of research, e.g. [9, 19, 15, 14, 18]), and study of a property-preserving domain extension transforms or "modes of operation". The latter problem is further investigated in this paper.

The most well-known and widely-used domain extender is the *strengthened* Merkle-Damgård (s-MD) construction which is known to be a collision resistance (Coll) preserving transform [7, 8]. Bellare and Rogaway [3] showed that s-MD, despite preserving collision resistance, is unable to preserve the UOWHF property [12] which is a weaker notion of security compared to the Coll property. Bellare and Ristenpart in Asiacrypt 2006 [4] advocated designing domain extension transforms in a way that they can simultaneously preserve multiple security properties of the underlying compression function. Multi-property-preserving (MPP) domain extension has been studies in several follow-up works, e.g. [5, 1, 17].

Due to the "orthogonality of property preservation" [4], *in general* one cannot use the mere fact that a domain extender preserves a given security property P to deduce whether the extender can also preserve another weaker security property P' or another stronger security property P''. To clarify this, note that Bellare and Rogaway [3] showed that the s-MD transform, despite being able to preserve the Coll property, is unable to preserve the notion of target collision resistance (TCR) which is a weaker property than the Coll property. As another example, Reyhanitabar, Susilo and Mu [16] showed that several TCR-preseving domain extenders fail to preserve eTCR, which is a stronger security notion than the TCR property. We also refer to [4] where it has been shown that the prefix-free MD which is a random-oracle preserving transform [6] does not preserve the Coll property.

Our Contribution. We investigate MPP capabilities of four domain extension transforms with respect to the enhanced security properties of [18]; namely, we analyze the Strengthened MD (sMD) [8, 7], HAIFA [2], Enveloped Shoup [5] and Nested Linear Hash (nLH) [3, 16] domain extension transforms. Table 1 provides an overview of the MPP capabilities of these transforms, where the unreferenced entries related to the enhanced (or "strengthened" [18]) security properties are the results shown in this paper; namely, the (unreferenced) results on strengthened collision-resistance (s-Coll), strengthened second-preimage resistance (s-Sec), strengthened always Sec (s-aSec), strengthened everywhere Sec

Table 1. Overview of the constructions and the properties they preserve. "Y" means that the property is preserved by the construction; "N" means that it is not preserved.

	Coll	Sec	aSec	eSec	Pre	aPre	ePre	s-Coll	s-Sec	s-aSec	s-eSec	s-Pre	s-aPre
sMD	Y [7, 8]	N [1]	N [1]	N [3]	N [1]	N [1]	Y [1]	Y	N	N	N [16]	N	N
HAIFA	Y [2]	N [1]	N [1]	N [1]	N [1]	N [1]	Y [1]	Y	N	N	N	N	N
ESh	Y [5]	N [17]	N [17]	Y [5]	N [17]	N [17]	Y [17]	N	N	N	N [16]	N	N
nLH	Y [3]	N [1]	N [1]	Y [3]	N [1]	N [1]	Y [1]	Y	N	N	Y [16]	N	N

(s-eSec a.k.a. eTCR), strengthened preimage resistance (s-Pre) and strengthened always Pre (s-aPre). We follow the nomenclature used in [18] for referring to these security properties.

2 Preliminaries

Notations and Conventions. If X is a finite set, by $x \xleftarrow{\$} X$ it is meant that x is chosen from X uniformly at random. For a binary string $M = M_1||M_2||\cdots||M_m$, let $M_{1\ldots n}$ denote the first n bits of M (i.e. $M_1||\cdots||M_n$) and $|M|$ denote its length in bits (where $n \leq m = |M|$). Let $x||y$ denote the string obtained from concatenating string y to string x. Let 1^m and 0^m, respectively, denote a string of m consecutive 1 and 0 bits, and 1^m0^n denote the concatenation of 0^n to 1^m. The set of all binary strings of length n bits (for some positive integer n) is denoted by $\{0,1\}^n$, the set of all binary strings whose lengths are variable but upper-bounded by N is denoted by $\{0,1\}^{\leq N}$ and the set of all finite binary strings is denoted by $\{0,1\}^*$. If S is a finite set we denote size of S by $|S|$. The symbol $\wedge$ denotes logical 'AND' operation, and the symbol $\vee$ denotes logical 'OR' operation. For a positive integer m, let $\langle m \rangle_b$ denotes binary representation of m by a string of length exactly b bits. The operation $M_1 \cdots M_L \xleftarrow{b} M$ is defined as follows. Let $L = \lceil |M|/b \rceil$; if $|M| \bmod b = 0$ then parse M into its b-bit blocks $M_1 \cdots M_L$ where $|M_j| = b$ for $1 \leq j \leq L$; otherwise parse M into $M_1, M_2, \cdots, M_{L-1}, M_L$ such that $|M_j| = b$ for $1 \leq j \leq L-1$ and $|M_L| = |M| \bmod b$ (i.e. the last block M_L may be shorter than a normal block, though it is still counted as a block).

We denote a hash function by $H : \mathcal{K} \times \mathcal{M} \rightarrow \{0,1\}^n$ where $\mathcal{K}$ and $\mathcal{M}$ are called the key space and message space, respectively, and n is a positive integer s.t. $log_2|\mathcal{M}| > n$ (this is to stress that H can compress). The keyspace $\mathcal{K}$ is a non-empty set of strings which may be called keys, indexes or salts (depending on the setting).

Definitions of Enhanced Security Notions. As usual in concrete-security definitions, the resource parameterized function $\mathbf{Adv}^{\mathsf{xxx}}_H(\mathbf{r})$ denotes the maximal value of the adversarial advantage ($\mathbf{Adv}^{\mathsf{xxx}}_H(\mathbf{r}) = max_A \{\mathbf{Adv}^{\mathsf{xxx}}_H(A)\}$) over all adversaries A, against the xxx property of H, that use resources bounded by $\mathbf{r}$. For the six strengthened security notions, the advantage functions are defined in Fig. 1.

$$\mathrm{Adv}_H^{\text{s-Coll}}(A) = \Pr\left[K \xleftarrow{\$} \mathcal{K}; (M, M', K') \xleftarrow{\$} A(K) : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M')\right]$$

$$\mathrm{Adv}_H^{\text{s-Sec}[\delta]}(A) = \Pr\left[\begin{array}{l} K \xleftarrow{\$} \mathcal{K}; M \xleftarrow{\$} \{0,1\}^{\delta}; \\ K', M' \xleftarrow{\$} A(K, M) \quad : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M') \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-aSec}[\delta]}(A) = \Pr\left[\begin{array}{l} (K, State) \xleftarrow{\$} A_1(); \\ M \xleftarrow{\$} \{0,1\}^{\delta}; \\ K', M' \xleftarrow{\$} A_2(M, State) \quad : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M') \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-eSec}[\delta]}(A) = \Pr\left[\begin{array}{l} (M, State) \xleftarrow{\$} A_1(); \\ K \xleftarrow{\$} \mathcal{K}; \\ K', M' \xleftarrow{\$} A_2(K, State) \quad : (K, M) \neq (K', M') \wedge H_K(M) = H_{K'}(M') \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-Pre}[\delta]}(A) = \Pr\left[\begin{array}{l} K \xleftarrow{\$} \mathcal{K}; M \xleftarrow{\$} \{0,1\}^{\delta}; Y \leftarrow H_K(M); \\ K', M' \xleftarrow{\$} A(K, Y) \quad : \quad H_{K'}(M') = Y \end{array}\right]$$

$$\mathrm{Adv}_H^{\text{s-aPre}[\delta]}(A) = \Pr\left[\begin{array}{l} (K, State) \xleftarrow{\$} A_1(); \\ M \xleftarrow{\$} \{0,1\}^{\delta}; Y \leftarrow H_K(M); \\ K', M' \xleftarrow{\$} A_2(Y, State) \quad : \quad H_{K'}(M') = Y \end{array}\right]$$

Fig. 1. Definitions of the enhanced security notions for hash functions [18]

3 Our Target Domain Extenders

A domain extender transforms a compression function $h : \{0,1\}^k \times \{0,1\}^{n+b} \to \{0,1\}^n$ to a hash function $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$ where the message space $\mathcal{M}$ can be either $\{0,1\}^*$ (in which case H is an arbitrary-input-length function) or $\{0,1\}^{<2^\lambda}$ (in which case H is a VIL function), for some positive integer λ (e.g. $\lambda = 64$). A domain extension transform comprises two functions: an (injective) padding function f_p, and an iteration function f_i. First, the padding function $f_p : \mathcal{M} \to D_I$ is applied to an input message $M \in \mathcal{M}$ to convert it to the padded message $f_p(M)$ in a domain D_I. Then, the iteration function $f_i : \mathcal{K} \times D_I \to \{0,1\}^n$ uses the compression function h as many times as required, and outputs the final hash value. The full-fledged hash function $H : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$ is obtained by combining the two functions; that is, $H(K, M) = f_i(K, f_p(M))$.

Padding Schemes. The padding functions, which are used by our target domain extension transforms (namely, sMD, HAIFA, nLH, and ESh) are defined in the following, where 2^λ is the maximum message length in bits (typically $\lambda = 64$).

Strengthening (a.k.a. suffix-free) Padding. This padding function is defined as $\mathrm{pad}_s : \{0,1\}^{<2^\lambda} \to \bigcup_{i \geq 1} \{0,1\}^{ib}$ where $\mathrm{pad}_s(M) = M||10^p||\langle |M| \rangle_\lambda$ and p is the minimum number of 0's required to make the padded message $\mathrm{pad}_s(M)$ be of length ib bits, for a positive integer i. A variant of this strengthening padding, which may be named as 'Nested Strengthening' or 'Full-Final-Block Strengthening' and we denote it by pad_s^*, is obtained when $\lambda = b$, i.e. when the complete final block is merely used for length indication.

HAIFA Padding. We note that our description of the HAIFA padding here is different from the original definition in [2], as here, we aim to show that HAIFA can be described by MD iteration combined with this specific padding. More precisely, here we consider inclusion of the "number of bits hashed so for" as part of the padding scheme before the iteration, while in [2] inclusion of the number of bits hashed so for is considered as part of the iteration process. Clearly, these are just two interpretations of how HAIFA works, and hence, as far as the property-preservation analysis of the whole HAIFA construction as a domain extender matters, these two representations are in fact equivalent. We also note that, here we assume only a fixed-length digest and hence the padding does not include the hash size.

The padding function for HAIFA can be defined as $\text{padHAIFA} : \{0,1\}^{<2^\lambda} \rightarrow \bigcup_{i\geq 1} \{0,1\}^{ib}$ where $\text{padHAIFA}(M)$ is computed using the following algorithm:

1. $M' \leftarrow M||10^p||\,\langle|M|\rangle_\lambda$ where p is the minimum number of 0's required to make the new message M' be of length $i(b-\lambda)$ bits, for some positive integer i.
2. $M'_1 \cdots M'_L \xleftarrow{b-\lambda} M'$; i.e. parse M' into L blocks each of length $b-\lambda$ bits.
3. Set $\#bits = 0$ (where $\#bits$ denotes the number of bits hashed so far). Now, for $1 \leq j \leq L-1$ do: $\#bits \leftarrow \#bits + (b-\lambda)$; $M''_j \leftarrow M'_j||\,\langle \#bits\rangle_\lambda$.
4. If M'_L is a full padding block (i.e., the full original message was already included in $M'_1 \cdots M'_{L-1}$) then $M''_L \leftarrow M'_L||\,\langle 0\rangle_\lambda$ else $M''_L \leftarrow M'_L||\,\langle|M|\rangle_\lambda$.
5. Return the padded message $M'' = M''_1 \cdots M''_L$.

Strengthened Chain Shift Padding. This padding function is defined as $\text{padCS}_s : \{0,1\}^{<2^\lambda} \rightarrow \bigcup_{i\geq 1} \{0,1\}^{(i+1)b-n}$ where $\text{padCS}_s(M) = M||10^r||\,\langle|M|\rangle_\lambda\,||0^p$, and parameters p and r are defined in the following two ways depending on the block length b. If $b \geq n+\lambda$ then $p = 0$, otherwise $p = b-n$. Then r is the minimum number of 0's required to make the padded message $\text{padCS}_s(M)$ be of length $ib + (b-n)$ bits, for a positive integer i.

Modes of Iteration. The iteration functions, used by our target domain extension transforms, namely, MD variants (including sMD, pre-MD, HAIFA), nLH and ESh, are shown in Figure 2 where IV, IV_1, and IV_2 are some known initial values $\in \{0,1\}^n$, and $IV_1 \neq IV_2$; L_{max} denotes the maximum allowed message length in blocks after padding; L denotes the length (in blocks) of a specific input message after padding applied.

Full-fledged Constructions. Given a compression function $h : \{0,1\}^k \times \{0,1\}^{n+b} \rightarrow \{0,1\}^n$, and some fixed initial values IV, IV_1, and $IV_2 \in \{0,1\}^n$ s.t. $IV_1 \neq IV_2$, the full-fledged hash constructions obtained by using the target domain extension transforms are defined as follows.

MD Variants. We say that a domain extension transform is an MD variant if it uses MD^h_{IV} for its mode of iteration together with its specific padding scheme.

– **sMD** [8, 7]: This transform defines a VIL hash function $H : \{0,1\}^k \times \{0,1\}^{<2^\lambda} \rightarrow \{0,1\}^n$ by $H(K,M) = MD^h_{IV}(K, \text{pad}_s(M))$.

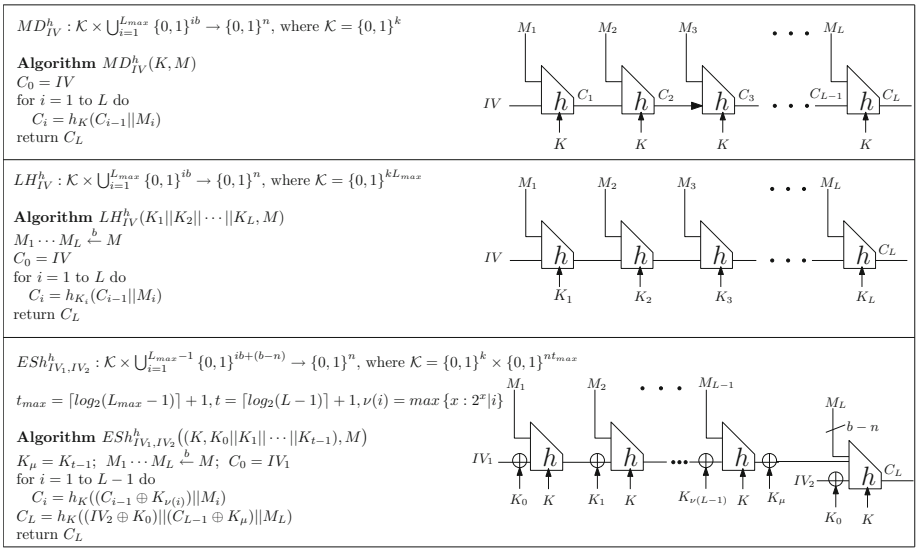


Fig. 2. From top to bottom: MD iteration [8, 7], LH iteration [3] and ESh [5]

– **HAIFA** [2]: This transform defines a VIL hash function $H : \{0,1\}^k \times \{0,1\}^{<2^\lambda} \to \{0,1\}^n$ by $H(K, M) = MD^h_{IV}(K, \text{padHAIFA}(M))$. We note that HAIFA can also handle variable length digests as shown in [2], but here we assume that the hash size is fixed to n bits. The key K here is the same as the "salt" S in [2].

nLH [3]. Here, we consider a variant of LH, denoted by nLH *in this paper*, which combines the LH iteration with the full-final-block (nesting) strengthening which we denoted by pad^*_s. The nLH domain extension transform defines a VIL hash function $H : \mathcal{K} \times \{0,1\}^{<2^\lambda} \to \{0,1\}^n$ by $H\big(K_1||\cdots||K_{L_{max}}, M\big) = LH^h_{IV}\big(K_1||\cdots||K_L, \text{pad}^*_s(M)\big)$ where $\mathcal{K} = \{0,1\}^{kL_{max}}$, L_{max} denotes the maximum allowed message length in blocks (after applying pad^*_s) and L denotes the length of the input message M in blocks after padding.

ESh [5]. The VIL hash function $H : \mathcal{K} \times \{0,1\}^{<2^\lambda} \to \{0,1\}^n$ is defined by $H\big((K, K_0||\cdots||K_{t_{max}-1}), M\big) = ESh^h_{IV_1,IV_2}\big((K, K_0||\cdots||K_{t-1}), \text{padCS}(M)\big)$ in which $\mathcal{K} = \{0,1\}^k \times \{0,1\}^{nt_{max}}$, $t_{max} = \lceil log_2(L_{max} - 1) \rceil + 1$, L_{max} denotes the maximum allowed message length in blocks (after applying padCS), $t = \lceil log_2(L) \rceil$ and L denotes the length of the input message M in blocks after padding. Note that, although the final block (after applying padCS) has only $b - n$ bits, it is still counted as a block.

4 Analysis of the Transforms

4.1 Analysis of s-Coll Preservation

We show both negative and positive results with regard to preservation of s-Coll by the four target transforms. Namely, Theorem 1 shows that ESh does not preserve the s-Coll property and Theorem 2 shows that s-Coll is preserved by sMD, HAIFA and nLH transforms.

Theorem 1. *Let $h : \{0,1\}^k \times \{0,1\}^{n+b} \rightarrow \{0,1\}^n$ be a $(t,\epsilon)-$s-Coll compression function, and IV_1 and IV_2 be any arbitrary fixed initial values $\in \{0,1\}^n$ s.t. $IV_1 \neq IV_2$. The full-fledged VIL hash function H obtained by using ESh domain extension transform is insecure in the sense of s-Coll property; i.e., there is an efficient adversary that can break its s-Coll property with probability 1.*

Proof. We construct an adversary A that has a (small) constant time complexity and breaks the s-Coll property of H with probability 1. A works as follows. After receiving the first key K from the challenger in the s-Coll game, A outputs (M, M', K') as follows:

- A chooses $M \neq M'$ s.t. $M = M_1||\cdots||M_{L-1}$ and $M' = M'_1||\cdots||M'_{L-1}$ (where $M_i \in \{0,1\}^b$ for $1 \leq i \leq L-1$). After applying the padding (where $b \geq n+\lambda$) we have $\text{padCS}_s(M) = M_1||\cdots||M_{L-1}||10^{b-n-1-\lambda}||\langle(L-1)b\rangle_\lambda$ and $\text{padCS}_s(M') = M'_1||\cdots||M'_{L-1}||10^{b-n-1-\lambda}||\langle(L-1)b\rangle_\lambda$. That is, the inputs to ESh iteration function (see Fig. 2) will have the same last block as $M_L = M'_L = 10^{b-n-1-\lambda}\langle(L-1)b\rangle_\lambda$.
- A computes K' by putting all blocks of it the same as those of K *except* the last key block (i.e. $K'_\mu \neq K_\mu$). The value of K'_μ is computed as $K'_\mu = K_\mu \oplus C_{L-1} \oplus C'_{L-1}$ in order to *cancel out the introduced difference in the chaining variables* C_{L-1} and C'_{L-1} (related to the computation for M and M', respectively). Clearly, this makes $C_L = C'_L$, i.e. the pairs (K, M) and (K', M') collide under H.

The above description was for the case that $b \geq n+\lambda$. The attack description for the case that $b < n+\lambda$ is quite similar. In this case $\text{padCS}_s(M) = M_1||\cdots||M_{L-1}||10^{b-\lambda-1}||\langle(L-1)b\rangle_\lambda||0^{b-n}$ and $\text{padCS}_s(M') = M'_1||\cdots||M'_{L-1}||10^{b-\lambda-1}||\langle(L-1)b\rangle_\lambda||0^{b-n}$, hence, again the last blocks input to iteration function will be the same for two messages as $M_{L+1} = M'_{L+1} = 0^{b-n}$. Therefore, the difference between C_L and C'_L can be canceled out by adjusting $K'_\mu = K_\mu \oplus C_L \oplus C'_L$ to make collision at the hash values (i.e. $C_{L+1} = C'_{L+1}$). □

Theorem 2. *Let $h : \{0,1\}^k \times \{0,1\}^{n+b} \rightarrow \{0,1\}^n$ be a $(t,\epsilon)-$s-Coll compression function, and $IV \in \{0,1\}^n$ be an arbitrary fixed initial value. The full-fledged VIL hash construction obtained by using either of the sMD, HAIFA or nLH domain extension transforms will be $(t',\ell,\epsilon)-$s-Coll where $t' = t - cT_hL$. c is a small constant, L is the number of blocks after padding a message of length ℓ bits, and T_h denote the time for computing h.*

Proof (Sketch). The proof is a slight modification of the Coll-preservation proofs for these domain extenders [8, 7, 2], using a simple reduction to transform any adversary A against the Coll property of the VIL hash function H to an adversary B against the Coll property of the underlying compression function h. The only difference here for the case of s-Coll is that now an adversary attacking s-Coll can also select a second key K' which may be different from the first key K given as a challenge. That is, A after receiving a random key K may output $(K, M) \neq (K', M')$ s.t. $H(K, M) = H(K', M')$ (where ℓ=maximum $\{|M|, |M'|\}$) . Clearly, if $K = K'$ (enforcing $M \neq M'$) then the reduction is the same as in the case of Coll preservation. If $K' \neq K$ the reduction is quite similar; namely, adversary B runs A by passing its own challenge key K to A, and after receiving the colliding pair (K, M) and (K', M') from A, computes all the intermediate results in computations of $H(K, M)$ and $H(K', M')$. Then starting from the final hash value where the two pairs collide (and going backward) B searches for the first position i where the corresponding $(n + b)$-bit messages or k-bit keys, used in the computations of (K, M) and (K', M'), are different from each other and output these as a colliding pair (in the sense of s-Coll) for h. Figure 2 may be helpful for verification of this, if needed. □

4.2 Analysis of s-eSec Preservation

s-eSec (or eTCR) preservation capabilities of several domain extension transforms (including ESh and nLH but not HAIFA) were first investigated in [16], where it was shown that, except nLH, none of the other studied transforms can preserve s-eSec. An issue with the nLH is that it has a linear key expansion in the message length (i.e. $|\mathcal{K}| = Lk$ where L is the message length in blocks and k is the key size for the underlying compression function). Mironov at FSE 2010 [10] provided an eTCR (s-eSec) preserving construction whose key expansion is only logarithmic in the message length and hence improves upon the nLH. Here, we study the e-eSec preservation capability of HAIFA. Theorem 3 shows that HAIFA, similar to other variants of MD, *does not preserve* the s-eSec property. To show this, we borrow a counterexample from [3] that was originally used for the TCR (eSec) preservation analysis in [3] and later for eTCR (e-eSec) preservation analysis in [16].

Theorem 3 (HAIFA does not preserve s-eSec). *Let* $h : \{0,1\}^k \times \{0,1\}^{n+b} \rightarrow \{0,1\}^n$ *be a* $(t, \epsilon)-$*s-eSec compression function, s.t.* $b > k$. *Set the new block length* $b' = b - k$ *and define a new compression function* $h' : \{0,1\}^k \times \{0,1\}^{(n+k)+b'} \rightarrow \{0,1\}^{n+k}$ *by*

$$h'_K(C_1||C_2||M) = \begin{cases} h_K(C_1||C_2||M)||K & \text{if } K \neq C_2 \\ 0^{n+k} & \text{if } K = C_2 \end{cases}$$

where $K \in \{0,1\}^k, C_1 \in \{0,1\}^n, C_2 \in \{0,1\}^k, M \in \{0,1\}^{b'}$ *(i.e.,* $n + k$ *is the chaining variable length and* b' *is the block length for* h'*). The compression function* h' *defined as above is* $(t', \epsilon')-$*s-eSec, where* $\epsilon' = \epsilon + 2^{-k+1}$ *and* $t' = t - c$,

for a small constant c, but the VIL hash function obtained from it using the HAIFA domain extension transform is completely insecure in the s-eSec sense.

Proof. It is known from [16] that h' is $(t', \epsilon')-$s-eSec (i.e. h' inherits the s-eSec property from h). It remains to show that the VIL hash function $H : \{0,1\}^k \times \{0,1\}^{<2^\lambda} \rightarrow \{0,1\}^n$ obtained by applying HAIFA on h', i.e. $H(K, M) = MD^{h'}_{IV}(K, \text{padHAIFA}(M))$ (where the initial value $IV = IV_1||IV_2 \in \{0,1\}^{n+k}$ is fixed and known) is completely insecure.

Adversary $A = (A_1, A_2)$ attacking the e-eSec property of H works as follow. A_1 outputs $M = 0^{b'-\lambda}$ and A_2, on receiving the first key K, outputs a different message $M' = 1^{b'-\lambda}$ together with any *arbitrary* key K' s.t. $K' \neq IV_2$, as the second key. Considering that the initial value $IV = IV_1||IV_2 \in \{0,1\}^{n+k}$ is fixed before the adversary starts the attack and the key K is chosen at random by the challenger, we have $\Pr[K = IV_2] = 2^{-k}$. If $K \neq IV_2$ which is the case with probability $1 - 2^{-k}$ then adversary succeeds in the s-eSec game as we have:

$$\begin{aligned} H(K, 0^{b'-\lambda}) &= MD^{h'}_{IV}(K, 0^{b'-\lambda}||\langle b'-\lambda\rangle_\lambda||10^{b'-2\lambda-1}||\langle b'-\lambda\rangle_\lambda||\langle 0\rangle_\lambda) \\ &= h'_K\Big(h'_K(IV||0^{b'-\lambda}||\langle b'-\lambda\rangle_\lambda)||10^{b'-2\lambda-1}||\langle b'-\lambda\rangle_\lambda||\langle 0\rangle_\lambda\Big) \\ &= h'_K\Big(h_K(IV||0^{b'-\lambda}||\langle b'-\lambda\rangle_\lambda)||K||10^{b'-2\lambda-1}||\langle b'-\lambda\rangle_\lambda||\langle 0\rangle_\lambda\Big) \\ &= 0^{n+k} \end{aligned}$$

$$\begin{aligned} H(K', 1^{b'-\lambda}) &= MD^{h'}_{IV}(K', 1^{b'-\lambda}||\langle b'-\lambda\rangle_\lambda||10^{b'-2\lambda-1}||\langle b'-\lambda\rangle_\lambda||\langle 0\rangle_\lambda) \\ &= h'_{K'}\Big(h'_{K'}(IV||1^{b'-\lambda}||\langle b'-\lambda\rangle_\lambda)||10^{b'-2\lambda-1}||\langle b'-\lambda\rangle_\lambda||\langle 0\rangle_\lambda\Big) \\ &= h'_{K'}\Big(h_{K'}(IV||1^{b'-\lambda}||\langle b'-\lambda\rangle_\lambda)||K||10^{b'-2\lambda-1}||\langle b'-\lambda\rangle_\lambda||\langle 0\rangle_\lambda\Big) \\ &= 0^{n+k} \end{aligned}$$

□

4.3 Analysis of s-Sec, s-aSec, s-Pre and s-aPre Preservation

In regard to preservation of these four strengthened variants of the second-preimage and preimage resistance properties for a dedicated-key hash function, we show a negative result stating that none of these properties can be provably preserved by any of the sMD, HAIFA, nLH, or ESh transform. This is in line with the previously known negative results about inability of these transforms and many others [1, 17] to preserve the associated (weaker) notions of Sec, aSec, Pre, and aPre. However, we note that orthogonality of property-preservation implies that this should be verified directly and may not be deduced from the previous results on the weaker notions.

Theorem 4. *None of the s-Sec, s-aSec, s-Pre and s-aPre properties can be preserved by any of the sMD, HAIFA, nLH, or ESh domain extension transforms.*

We prove this theorem using Lemma 1 and Lemma 2. These lemmas show that we can construct counterexample compression functions which are secure in the sense of xxx ∈ {s-Sec, s-aSec, s-Pre, s-aPre} assuming that there exist any xxx-secure compression function, but for which the full-fledged hash functions obtained using the target domain extension transforms are completely insecure in the sense of xxx ∈ {s-Sec[δ], s-aSec[δ], s-Pre[δ], s-aPre[δ]}, respectively, and for *any value of the parameter* $\delta < 2^{\lambda}$ (2^{λ} is the maximum input message length in bits). Construction of these counterexamples are inspired from similar ones in [17] where they were used in study of the Sec, aSec, Pre and aPre preservation.

Lemma 1. *Let* $h : \{0,1\}^k \times \{0,1\}^{n+b} \rightarrow \{0,1\}^n$ *be a* (t,ϵ)−xxx *compression function where* xxx ∈ {*s-Sec, s-aSec, s-Pre, s-aPre*}. *Define new compression functions* $h' : \{0,1\}^k \times \{0,1\}^{n+b} \rightarrow \{0,1\}^{n+1}$ *and* $h'' : \{0,1\}^k \times \{0,1\}^{n+b} \rightarrow \{0,1\}^{n+1}$ *by*

$$h'_K(M) = \begin{cases} 1^{n+1} & \text{if } M_{(n+b-\lambda+1)\cdots(n+b)} = \langle\delta\rangle_\lambda \\ h_K(M)||0 & \text{otherwise} \end{cases}$$

$$h''_K(M) = \begin{cases} 1^{n+1} & \text{if } M_{(n+b-\lambda+1)\cdots(n+b)} = \langle\delta\rangle_\lambda \vee M_{(n+b-\lambda+1)\cdots(n+b)} = \langle 0\rangle_\lambda \\ h_K(M)||0 & \text{otherwise} \end{cases}.$$

The compression function h' *is* $(t-c, \epsilon+2^{-\lambda})$−xxx *and* h'' *is* $(t-c, \epsilon+2^{-\lambda+1})$−xxx *for any* xxx ∈ {*s-Sec,s-aSec, s-Pre, s-aPre*} *for which* h *is* (t,ϵ)−xxx, *respectively.* c *is a small absolute constant.*

Proof. The idea behind the construction of h' and h'' is that in all of the four games defining the s-Sec, s-aSec, s-Pre, and s-aPre properties (see Fig. 1) the first message M is chosen at random by the challenger; hence, the probability of the event, denoted by *Bad*, where the last λ bits of M equal to a fixed λ-bit string in the case of h' or either of two fixed stings in the definition of h'' is , respectively, $2^{-\lambda}$ and $2^{-\lambda+1}$ (becoming negligible for a typical value of λ e.g. $\lambda = 64$). The domain separation in the constructions of h' and h'' is achieved by simply appending a constant bit with value 0 to the hash value $h_K(M)$ and aims to force (second)preimages to belong to the same domain. It remains to show that unless *Bad* happens, any adversary against an xxx property of h' or h'' can be easily transformed to an adversary against the xxx property of h. The reductions showing this for any xxx ∈ {s-Sec, s-aSec, s-Pre, s-aPre} are simple, and hence in the following we only describe the reduction for one of the cases; namely, we show that h' inherits s-Sec from h. All the remaining reductions for the other cases are quite similar and omitted here.

The case of h' and xxx=s-Sec: Let A be an adversary attacking the s-Sec property of h', that has time complexity t' and advantage ϵ'. We construct and adversary B against the s-Sec property of h that has time complexity $t = t' + c$ (for a small constant c) and advantage $\epsilon = \epsilon' - 2^{-\lambda}$. Adversary B on receiving the key K and the first message M (where $K \xleftarrow{\$} \{0,1\}^k$ and $M \xleftarrow{\$} \{0,1\}^{n+b}$) checks whether the event *Bad* has happened, i.e. whether $M_{(n+b-\lambda+1)\cdots(n+b)} = \langle\delta\rangle_\lambda$

or not. If *Bad* happened then B would abort; otherwise, it forwards K and M to A, gets the second preimage M' and the second key K' from A for h', and outputs M' and K' as its own second preimage and key pair for h. We note that if *Bad* does not happen, then according to the construction of h' adversary B wins the s-Sec game against h (i.e. we have $(K, M) \neq (K', M')$ and $h_K(M) = h_{K'}(M')$) whenever A wins the s-Sec game against h' (i.e. if $(K, M) \neq (K', M')$ and $h'_K(M) = h'_{K'}(M')$). Hence, we have Pr[B wins]=Pr[A wins and $\overline{Bad}$] $\geq$ Pr[A wins]-Pr[*Bad*] $= \epsilon' - 2^{-\lambda}$. The time complexity of B is that of A plus a small constant time used for reading/writing and forwarding the messages. □

Lemma 2. *For any* xxx $\in$ {*s-Sec[δ], s-aSec[δ], s-Pre[δ], s-aPre[δ]*} *and for any value of the parameter* $\delta < 2^{\lambda}$*, there is an adversary with constant time complexity that with probability 1 breaks the hash functions obtained by applying sMD, nLH and ESh on the counterexample compression function* h' *and the HAIFA hash function constructed using the counterexample compression function* h''*.*

Proof. For the cases of sMD, nLH and ESh using h', from the definitions of the padding sachems for these transforms (see Section 3) and the construction of h', we have $H(K, M) = 1^{n+1}$ for any arbitrary $K \in \mathcal{K}$ and any $M \in \{0,1\}^{\delta}$ (for an arbitrary value of the parameter δ s.t. $\delta < 2^{\lambda}$). This is because, the last λ bits of the *padded* message will encode the message length, i.e. $|M| = \delta$, and this forces h' (applied to the final padded block in the iteration) to output the constant value 1^{n+1}. Hence, in s-Pre[δ] and s-aPre[δ] attacks adversary A just outputs any arbitrary $M' \in \{0,1\}^{\delta}$ and $K' \in \mathcal{K}$ and always wins. Similarly, in s-Sec[δ] and s-aSec[δ] attacks A simply outputs an arbitrary $K' \in \mathcal{K}$ and any $M' \in \{0,1\}^{\delta}$ such that $M' \neq M$ message and always wins.

The case for HAIFA using h'' is quite similar and we have $H(K, M) = 1^{n+1}$ for any arbitrary $K \in \mathcal{K}$ and any $M \in \{0,1\}^{\delta}$. This is because by the definition of padHAIFA, the last λ bits of padHAIFA(M) is either $\langle 0 \rangle_{\lambda}$ (if it is a full padding block) or $\langle |M| \rangle_{\lambda}$ (if it contains some message bits). Now, it is easy to see that both of these cases will force the final application of the compression function h'' to output the constant value 1^{n+1} □

Acknowledgments. This work is supported by the NSFC Research Fund for International Young Scientists under Grant 61150110483. The authors would like to thank the anonymous reviewers of ProvSec 2012 for their constructive comments and suggestions.

References

[1] Andreeva, E., Neven, G., Preneel, B., Shrimpton, T.: Seven-Property-Preserving Iterated Hashing: ROX. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 130–146. Springer, Heidelberg (2007)

[2] Biham, E., Dunkelman, O.: A Framework for Iterative Hash Functions HAIFA. In: NIST Second Cryptographic Hash Workshop, Santa Barbara (August 2006), http://eprint.iacr.org/2007/278

[3] Bellare, M., Rogaway, P.: Collision-Resistant Hashing: Towards Making UOWHFs Practical. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 470–484. Springer, Heidelberg (1997)
[4] Bellare, M., Ristenpart, T.: Multi-Property-Preserving Hash Domain Extension and the EMD Transform. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 299–314. Springer, Heidelberg (2006)
[5] Bellare, M., Ristenpart, T.: Hash Functions in the Dedicated-Key Setting: Design Choices and MPP Transforms. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 399–410. Springer, Heidelberg (2007)
[6] Coron, J.-S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård Revisited: How to Construct a Hash Function. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer, Heidelberg (2005)
[7] Damgård, I.: A Design Principle for Hash Functions. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 416–427. Springer, Heidelberg (1990)
[8] Merkle, R.C.: One Way Hash Functions and DES. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 428–446. Springer, Heidelberg (1990)
[9] Mironov, I.: Hash Functions: From Merkle-Damgård to Shoup. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 166–181. Springer, Heidelberg (2001)
[10] Mironov, I.: Domain Extension for Enhanced Target Collision-Resistant Hash Functions. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 153–167. Springer, Heidelberg (2010)
[11] Maurer, U., Sjödin, J.: Single-Key AIL-MACs from Any FIL-MAC. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 472–484. Springer, Heidelberg (2005)
[12] Naor, M., Yung, M.: Universal One-Way Hash Functions and Their Cryptographic Applications. In: Proceedings of the 21st ACM Symposium on the Theory of Computing–STOC 1989, pp. 33–43. ACM (1989)
[13] National Institute of Standards and Technology. Cryptographic Hash Algorithm Competition, http://csrc.nist.gov/groups/ST/hash/sha-3/index.html
[14] Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 371–388. Springer, Heidelberg (2004)
[15] Rogaway, R., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. Cryptology ePrint Archive, Report 2004/035 (latest revised version: August 9, 2009), http://eprint.iacr.org/2004/035.pdf
[16] Reyhanitabar, M.R., Susilo, W., Mu, Y.: Enhanced Target Collision Resistant Hash Functions Revisited. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 327–344. Springer, Heidelberg (2009)
[17] Reyhanitabar, M.R., Susilo, W., Mu, Y.: Analysis of Property-Preservation Capabilities of the ROX and ESh Hash Domain Extenders. In: Boyd, C., González Nieto, J. (eds.) ACISP 2009. LNCS, vol. 5594, pp. 153–170. Springer, Heidelberg (2009)
[18] Reyhanitabar, M.R., Susilo, W., Mu, Y.: Enhanced Security Notions for Dedicated-Key Hash Functions: Definitions and Relationships. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 192–211. Springer, Heidelberg (2010)
[19] Stinson, D.R.: Some Observation on the Theory of Cryptographic Hash Functions. Design, Codes and Cryptography 38(2), 259–277 (2006)

Revisiting a Secret Sharing Approach to Network Codes

Zhaohui Tang, Hoon Wei Lim, and Huaxiong Wang

Division of Mathematical Sciences
School of Physical and Mathematical Sciences
Nanyang Technological University, Singapore
TANG0209@e.ntu.edu.sg, {hoonwei,HXWang}@ntu.edu.sg

Abstract. Secret sharing has been used in multicast network coding to transmit information securely in the presence of a wiretapper who is capable of eavesdropping a bounded number of network edges. Typically information-theoretic security against the wiretapper can be achieved when multicasting a message by concatenating it with some random symbols and representing them with a carefully chosen network code. In this paper, we revisit a secret sharing approach to network codes. Particularly, we are concerned with whether or not a given network code is secure. We derive a necessary and sufficient condition for a given network code to be secure for a network coding instance. In comparison with previous work, our condition is more explicit in the sense that it is easier to verify. Furthermore, we devise a precise algorithm to transform an insecure network code into a secure one. Our algorithm achieves smaller secure code alphabet size over previous work.

Keywords: Network coding, secret sharing, information-theoretic security.

1 Introduction

As opposed to the standard routing technology that allows intermediate nodes to simply copy and forward their incoming packets, *network coding* [1] enables an intermediate node to encode received packets before forwarding them. One consequence of this is that network coding can provide a higher throughput rate than the standard copy-and-forward routing does. Particularly, in a multicast network where multiple receivers demand all messages from the same source, Li et al. [2] showed that one can achieve a maximal transmission rate at every receiver through network coding. On the contrary, it seems hard to attain maximal capacity using the standard routing [1]. Moreover, network coding increases the robustness of packet routing and has lower energy consumption [5]. This makes network coding attractive to various applications, for example content distribution, wireless network coding and so forth [6, 7, 9–12].

Security can be a major concern in a networking system and securing network coding against both passive and active attacks is non-trivial at all. There has

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 300–317, 2012.

been a great deal of work on network coding security against an active adversary, see for example [13–19]. In this paper, however, we focus on *information-theoretic security against a passive adversary*, modeled as a wiretapper who eavesdrops along a limited number of network links within a *multicast network*. Particularly, we require that no information about a protected message is revealed to the wiretapper. Note that this is a stronger requirement than a security model that allows disclosure of partial information to the passive adversary [20, 21]. Putting this into the context of securing a network code, our goal is to find an appropriate network code such that the wiretapper has no way to deduce any information about the original message. We achieve this using techniques from *secret sharing* [28, 34, 35].

Succinctly, secret sharing is a concept that allows a dealer to share a secret with multiple participants in an information-theoretically secure way against the passive adversary. The dealer can achieve this using a secret sharing scheme that performs the following: (i) computing a suitable function over the secret concatenated with some random elements; and (ii) distributing as shares the function output to multiple participants. The function is chosen by the dealer in such a way that with the resultant shares, the participants share the secret while ensuring information-theoretic security. Interestingly, there exists a natural connection between secure network coding and secret sharing. This is uncovered by Cai and Yeung [22] and further investigated by Feldman et al. [23]. Using a secret sharing approach, transmission of a network code can be regarded as a secret sharing scheme where all network edges (participants) share a message (secret) originated from the source (dealer). A network code, in the form of a function, is chosen by the source in order to compute shares as in a secret sharing scheme. Informally, we say a network code is secure, if the function representing the associated network code is information-theoretically secure against the adversary.

There exists another interesting approach in the literature that also addresses information-theoretic security against the wiretapper in network coding. It is called *coset coding* since it treats a secure network code as a combination of a coset code [27] and a network code. This approach was introduced by Rouayheb and Soljanin [24] and further developed by Silva and Kschischang [25]. However, as shown in [25], the construction proposed by Rouayheb and Soljanin is equivalent to the above secret sharing approach. The only advantage of a coset code over those presented in [22, 23] is that one can obtain a smaller secure code alphabet size using the former approach. (We will revisit this point in Section 5.2). Furthermore, the code designed by Silva and Kschischang has been showed to be not achieving perfect security [26].

Motivating Problems. In network coding security, we are concerned with the security condition for a given network code. This is analogous to secret sharing where it is essential to provide a security condition for a chosen function. On this aspect, previous work [22, 23] simply adopts Shannon's general condition for perfect secrecy [8]. However, it is generally known that an entropy-based condition is often too abstract, and thus, difficult to verify in practice.

A natural and interesting question in securing network coding is how to make a given network code secure. Cai and Yeung [22] and Feldman et al. [23] have studied the feasibility of transforming an insecure network code into a secure one. However, they have never provided an executable realization of their transformation. Furthermore, Cai and Yeung derived an exponential lower bound (in the size of edges wiretapped) on the required bandwidth for a secure network code to achieve a maximally secure capacity. This is not very useful in practice.

In this work, we focus on addressing the above problems, which we will formally define in Section 3.

Our Contributions. We first propose a new concept which we call *k-threshold secure matrix* (see Definition 1). With this concept, we can then provide a sufficient and necessary condition for a given network code to be secure for a network coding instance (see Theorem 1). Our security condition is more explicit than Shannon's entropy-based condition. Consequently, it is easier to verify than that adopted in [22, 23].

Further, we present a concrete algorithm that derives a secure network code from a given network code. We also show that our required bandwidth is smaller than Cai and Yeung's bound. More importantly, we prove that our bandwidth is as optimal as Rouayheb and Soljanin's bound. Details on these results are presented in Section 5. Note that we do not compare our results with those by Feldman et al. [23], since the latter worked on smaller capacity and their probability of realizing perfect security is less than 1.

We obtain our results by exploiting techniques from information theory and secret sharing. Particularly, we realize linear transformation through linear algebra to make a network code secure.

Organization. The remainder of the paper is organized as follows. Section 2 defines some concepts in network coding and its security, as well as secret sharing. In Section 3, we formalize our motivating problems. In section 4, we introduce the concept of k-threshold secure matrix and propose a condition to determine whether or not a given network code is secure. Subsequently, in Section 5, we present a concrete algorithm that is used to derive a secure network code. Lastly, we conclude in Section 6.

2 Preliminaries

We first give some basic concepts and definitions related to network coding, secure network coding, and secret sharing.

Security against k-Threshold Adversary. Suppose we are given a finite field $\mathbb{F}_q$, an original input $x \in \mathbb{F}_q^t$, and a function $W(x, r) : \mathbb{F}_q^t \times \mathbb{F}_q^l \to \mathbb{F}_q^N$. Here $r \in \mathbb{F}_q^l$ is used as a random input for security consideration; and in our settings where W is linear, we can easily write $W(x, r)$ as $(x, r)S$ with S being an $(t + \ell) \times N$ matrix and (x, r) being a concatenation of row vectors x and r . Let also $I \subseteq [N]$ and S_I represent the columns in S with indices restricted to I.

We then say that S is secure against I if for all $x, x' \in \mathbb{F}_q^t$, we have that the random variables $(x, r)S_I$ and $(x', r)S_I$ are identically distributed. Naturally, we say that S is *secure against a k-threshold adversary* if S is secure against I for all $I \subseteq [N]$ with $|I| \leq k$.

NETWORK CODING. We consider a multicast network which is modeled as a unit capacity directed graph $G(\mathcal{V}, \mathcal{E})$ with $\mathcal{V}$ its node set, $\mathcal{E}$ edge set, s_G representing the source node, T_G denoting the set of sinks, n referring to the message length, $\mathbb{F}_q$ being the finite field where symbols transmitted along network edges are chosen from.

A *network coding solution* is a scheme, by which an arbitrarily generated message $\mathbf{m} \in \mathbb{F}_q^n$ can be multicast from the source to all the sinks simultaneously.[1] More formally, a *network coding solution* is an assignment of $|\mathcal{E}|$ many functions $f_{(u,v)}$, one for each edge $(u, v) \in \mathcal{E}$, satisfying two properties below:

- For each edge (u, v), the symbol transmitted over (u, v) is the value of function $f_{(u,v)}$ applied to symbols flowing into node u (or the entire message vector $\mathbf{m}$ in the case of u being the source node);
- For each sink $v \in T_G$, if $f_{(w,v)}$ is applied to symbols flowing into (w, v) for all w, it yields the original message vector $\mathbf{m}$.

In *linear network coding*, each of the functions above are linear. If we regard the original message $\mathbf{m}$ as a sequence of n symbols over $\mathbb{F}_q$, by recursion, in linear network coding, the function for each edge (u, v) is a linear function on the original n symbols. We represent $V_{(u,v)}$ as the *encoding vector* for edge (u, v) such that $\mathbf{m} \cdot V_{(u,v)}$ is the symbol carried over the edge (u, v).

A *linear network code* V for a network coding instance $\mathcal{I}$= $(G(\mathcal{V}, \mathcal{E}),n, N, q)$ then consists of an encoding vector for each of the N edges, formally, $V = \{V_e : e \in \mathcal{E}\}$. It has been proved in [1, 2] that a linear network coding solution always exists for $\mathcal{I}$, as long as $q \geq |T_G|$ (recall that T_G is the sink set in G) and minimal cut of G does not exceed n.

Unless otherwise indicated, from now on we refer to network coding as linear network coding, and a network code as a linear network code.

SECURE NETWORK CODING. We now consider the security aspect of linear network coding. We assume that an adversary can listen to transmissions over up to k links of his choice.

We say a network coding solution is *secure* if an arbitrarily generated message $x \in \mathbb{F}_q^t$ ($t \leq n$ necessarily [22, 24]) can be multicast from the source to all the sinks, in a way that is information-theoretically secure against a k-threshold adversary. A *k-threshold secure network code* for a network instance $\mathcal{I}$= $(G(\mathcal{V}, \mathcal{E}),n, N, q)$ is then a network code for $\mathcal{I}$ that is secure against a k-threshold adversary.

[1] We don't consider the timing issue within the network and thus assume that the transmission across the entire network takes place in one unit time.

(k, n, N)-Ramp Secret Sharing. A classic (k, n, N)-ramp secret sharing scheme [36] is a secret sharing scheme where a dealer wants to share a secret $x \in \mathbb{F}_q^{n-k}$ to N parties with two goals:

- *Recoverability*: This requires that any collaboration of at least n parties can reconstruct x from their shares;
- *Secrecy*: This ensures that no collusion of at most k parties can gain any information about x.

3 Problem Statement

We now illustrate a connection between secure network coding and classic ramp secret sharing.

Firstly, it is obvious that they are connected with each other. To see why, imagine there is a network coding scenario where a network code V is employed to transmit a secret $x \in \mathbb{F}_q^{n-k}$ from the source through a network with N edges and in the presence of a wiretapper who is capable of eavesdropping up to k edges. A secure network coding under this situation is expected to achieve the recoverability and secrecy goals mentioned before. From basic network coding knowledge, any destination who receives n linearly independent vectors is able to recover x. In other words, we can say that V is a *valid* network code for a network coding instance $\mathcal{I}$ if V realizes recoverability in $\mathcal{I}$. To achieve secrecy with regards to x, it is sufficient that the network code V be a (k, n, N)-ramp secret sharing scheme; and it is not hard to give a necessary condition for a given network code to be secure for a given network coding instance. However, it is hard to construct, or even to give a sufficient condition for a (k, n, N)-ramp secret sharing scheme that is simultaneously also a valid network code for a general network coding instance. The reason for this is that, for a general network topology it is difficult to take care of the linear dependency of encoding vectors between different edges, which is required, however, for all valid linear network codes.

For the rest of the paper, therefore, as with [22] and [23], we assume that V has been given as a valid network code for the underlying network coding instance $\mathcal{I}$. This way, we can restrict our concern to the security of a network code constructed for an underlying network coding instance.

To our knowledge, the most studied (k, n, N)-ramp secret sharing schemes are linear schemes where the share for each participant is computed by a linear function on the secret x appended with some random elements. To achieve the required security, the function is chosen to be secure against a k-threshold adversary. Following the connection between secure network coding and secret sharing described above, one can make use of a suitable secret sharing scheme to select a secure network code from a set of valid network codes. The following scheme was adopted in [22, 23] to generate a secure network code V for the network instance $\mathcal{I}$:

Step 1: Choose a random vector $r \in \mathbb{F}_q^k$ and append r to the original message x. We denote the resulting vector as $X = (x, r)$.

Step 2: From all valid network code candidates, choose a network code V such that V is secure against a $k-$threshold adversary.
Step 3: Encode the vector X by transmitting along each link e the value XV_e.

We call the method for constructing a secure network code using the above scheme a *secret sharing approach*.

We remark that one crucial part in the secret sharing approach is how to choose a proper V in Step 2 of the above scheme. Prior to choosing V, it is also essential to determine whether or not a candidate V is secure. Cai and Yeung [22] studied the maximally secure capacity case where $t = n-k$ (the maximality was proved in [22]). Their security condition was following closely Shannon's general condition for perfect secrecy [8], that is, a given network code V is secure if:

$$H(\mathbf{m}|Y_A) = H(\mathbf{m}) \tag{1}$$

holds for all $A \subseteq \mathcal{E}$ with $|A| \leq k$, where $H(X)$ denotes the entropy of random variable X and Y_A represents the information intercepted by the adversary A when V is employed. Similarly, Feldman et al. [23] also adopted the same security condition as [22] in their work (although they considered the case with a lower capacity).

In general, a candidate network code V may not be a k-threshold secure network code for $\mathcal{I}$. However, it has been proved in [22] that if $q > \binom{N}{k}$, there exists a full rank $n \times n$ matrix T such that $S = TV$ is a k-threshold secure network code for $\mathcal{I}$. This way, V becomes a secure network code for $\mathcal{I}$, when the source computes $(x, r)T$ and uses it to represent a message (as compared to (x, r) previously). Cai and Yeung also exhibited a sufficient constraint for T to make V secure. Feldman et al. also showed that it is necessary to meet the sufficient condition for T. However, none of them has given any concrete algorithm to find a such a converter T.

We are now ready to state the problems that we intend to address in this paper. Here we consider the maximally secure capacity case where $t = n - k$. Given a network coding instance $\mathcal{I}$= $(G(\mathcal{V}, \mathcal{E}),n, N, q)$ and a network code V for $\mathcal{I}$, we are interested in solving the following:

Problem 1. Given a network code V, can one find an easier way than using (1) above to determine whether or not V is a k-threshold secure network code for $\mathcal{I}$?

Problem 2. Can one lower the Cai and Yeung's bound of $q > \binom{N}{k}$ with regards to T, as described above? More importantly, how can one construct a concrete converter T such that $S = TV$ is a k-threshold secure network code?

4 Security Condition for Network Codes

In this section, we give a solution to Problem 1.

Given a valid network code V for a network coding instance $(G(\mathcal{V}, \mathcal{E}),q, k)$, we first present the code as a full rank matrix (which we still call V) that comprises column vectors $(V^T[e])_{e\in\mathcal{E}}$. Henceforth, we will not distinguish between

the vector and matrix representations of a network code. In what follows, we define a concept called *k-threshold secure matrix*, which we subsequently use to determine the security of V.

4.1 *k*-Threshold Secure Matrices

We denote the rank of a matrix S over $\mathbb{F}_q$ by $\mathsf{rank}_q(S)$. Moreover, for $\forall I \subseteq [N]$, we write S_I as $\begin{bmatrix} S_I^U \\ S_I^L \end{bmatrix}$ where S_I^U is the upper sub-matrix with size $(n-k) \times |I|$ and S_I^L the lower sub-matrix with size $k \times |I|$.

Definition 1 ($k-$threshold secure matrix) *Given $n, N, k, \mathbb{F}_q$ where $k < n$ and a full rank $n \times N$ matrix S, S is a k-threshold secure matrix over $\mathbb{F}_q$ if*

$$\mathsf{rank}_q(S_I) = \mathsf{rank}_q(S_I^L)$$

holds for all $I \subset [N]$ with $|I| \leq k$.

We provide some examples of k-threshold secure matrices in Appendix A.

4.2 Determining Secure Network Codes

Theorem 1 below shows how one can verify if a network code is secure in a straightforward manner using our concept of k-threshold secure matrices. Since the latter is more explicit than the entropy-based condition shown in (1), our theorem may be of independent interest.

Theorem 1. *Given a network code V for a network coding instance $\mathcal{I}$= $(G(\mathcal{V}, \mathcal{E})$,$n, N, q)$, with n denoting the minimal cut of G and $N = |\mathcal{E}|$, V is a k-threshold secure network code for $\mathcal{I}$ if and only if V is a k-threshold secure matrix over $\mathbb{F}_q$.*

Proof. Let $y_0 \in \mathrm{Im}(V) = \{(x, r)V : x \in \mathbb{F}_q^{n-k}, r \in \mathbb{F}_q^k\}$, $x_0 \in x$ and $r_0 \in r$ such that $y_0 = \begin{bmatrix} x_0 \; r_0 \end{bmatrix} V$. For $x \in \mathbb{F}_q^{n-k}$ and $I \subset [N]$, we let

$$R_I(x, y_0) = \{r \in \mathbb{F}_q^k : \begin{bmatrix} x \; r \end{bmatrix} V_I = y_{0I}\}.$$

By definition, V_I is secure against I if and only if $|R_I(x', y_0)| = |R_I(x'', y_0)|$ for all $x', x'' \in \mathbb{F}_q^t$ where $|R_I(x', y_0) > 0|$ and $|R_I(x'', y_0)| > 0$ for $y_0 \in \mathrm{Im}(V)$.

Since V is linear, it is easy to see that

$$R_I(x, y_0) = r_0 + R_I(x - x_0, 0).$$

This implies that we have

$$|R_I(x', y_0)| = |R_I(x'', y_0)| \iff |R_I(x', 0)| = |R_I(x'', 0)|$$

for $x', x'' \in \mathbb{F}_q^{n-k}$, $y_0 \in \mathrm{Im}(V)$, and where $|R_I(x', y_0)|$,$|R_I(x'', y_0)|$,$|R_I(x', 0)|$, $|R_I(x'', 0)| > 0$.

Without loss of generality, we may assume V_I has $\mathsf{rank}\ |I|$; otherwise, we can drop some rows that are linear combinations of others without affecting our proof. Also, we know that the kernel of V_I, denoted by $\mathsf{ker}(V_I)$, has dimension $n - |I|$.

Hence, writing $V_I = \begin{bmatrix} V_I^U \\ V_I^L \end{bmatrix}$ with V_I^U an $(n-k) \times |I|$ matrix and V_I^L a $k \times |I|$ matrix, we have

$$R_I(x', 0) = \{r \in \mathbb{F}_q^k : x' V_I^U + r V_I^L = 0\} \quad \text{and} \quad R_I(0,0) = \{r : r V_I^L = 0\}.$$

It is clear that $R_I(0,0)$ is a linear subspace in $\mathbb{F}_q^k$ of dimension $\mathsf{rank}_q(V_I^L)$. Thus

$$|R_I(0,0)| = q^{k - \mathsf{rank}_q(V_I^L)}.$$

If $r' \in R_I(x', 0)$ and $r \in R_I(0,0)$, then $r' + r \in R_I(x', 0)$. Therefore,

$$|R_I(x', r)| \geq |R_I(0,0)| \quad \text{if } |R_I(x', 0)| > 0. \tag{2}$$

Moreover,

$$\mathsf{ker}(V_I) = \cup_{x \in \mathbb{F}_q^{n-k}} \ (\{x\} \oplus R_I(x, 0)).$$

Since the union is disjoint, one has

$$q^{n-|I|} = |\mathsf{ker}(V_I)| = \sum_{x \in \mathbb{F}_q^{n-k}} |(\{x\} \oplus R_I(x,0))| = \sum_{x \in \mathbb{F}_q^{n-k}} |R_I(x,0)|. \tag{3}$$

Since V_I is equally distributed, we have $|R_I(x,0)| = |R_I(0,0)|$ for all $x \in \mathbb{F}_q^{n-k}$. Therefore, we have

$$q^{n-|I|} = q^{n-k} q^{k - \mathsf{rank}_q(V_I^L)}$$

which implies that $\mathsf{rank}_q(V_I^L) = |I|$.

On the other hand, if $\mathsf{rank}_q(V_I^L) = |I|$ then for any $x \in \mathbb{F}_q^{n-k}$, we have that

$$V_I \begin{bmatrix} x \ r \end{bmatrix} = 0 \iff x V_I^U + r V_I^L = 0 \iff r V_I^L = -x V_I^U$$

always has a solution $r \in \mathbb{F}_q^k$. Therefore, $|R_I(x,0)| > 0$. By (2) and (3), one has

$$q^{n-|I|} \geq \sum_{x \in \mathbb{F}^{n-k}} |R_I(x,0)| \geq q^{n-k} |R_I(0,0)| = q^{n-k} q^{k-|I|}$$

This implies that

$$|R_I(x,0)| = |R_I(0,0)|, \quad x \in \mathbb{F}_q^{n-k}.$$

Therefore, we have that V is secure against I if and only if $\mathsf{rank}_q(V_I) = |I| = \mathsf{rank}_q(V_I^L)$. Restricting I such that $|I| \leq k$ and by the definition of a k-threshold secure matrix, Theorem 1 follows. □

5 Securing Network Codes

We now address Problem 2. It is worth mentioning that this problem was studied in [23], where it is proved that, with some probability, the alphabet size can be lowered by making $t < n - k$. However, since we consider a maximally secure capacity $t = n - k$, we believe that it is more fundamental and meaningful to try to improve the bound given in [22] instead. We defer the investigation of the case when $t < n - k$ to future work.

In what follows, we give a concrete algorithm that generates T, which in turn, yields a sufficient condition to make V secure, and more interestingly, with a smaller alphabet size than the bound in [22].

Before stating our theorem, we need to introduce some notation. For any sub-indices $J \subseteq \{j_1, \cdots, j_{k+1}\}$, let V^J be a sub-matrix of V with rows matching J, and let V_I^J be a sub-matrix of V^J with columns matching indices I. We choose I such that every $k+1$ columns from V_I^J are linearly independent. We set $p_{J,I}$ to be $|I|$, and let p_J be largest number of $|I|$ so that each of the $k+1$ columns of V_I^J are linearly independent. Let $\ell_{n-k} = \max\{p_J : J \subset [n], |J| = k+1\}$. It is then obvious that $\ell_{n-k} \leq N - 1$ and our theorem is as follows.

Theorem 2. *For any given positive integers $1 \leq k < n \leq N$ and an $n \times N$ full rank matrix V, there exists an $n \times n$ nonsingular matrix T, such that the resulting $S = TV$ is a k-threshold secure matrix if one of the following conditions holds:*

(i) $k = 1$ and $q \geq N = N_1$;

(ii) $k \geq 2$ and $q \geq max\left\{\binom{\ell_{n-i+1}}{i}, N : i = 2, \cdots, k\right\} = N_k$. (Here, ℓ_{n-i+1} will be defined later in the proof and will be shown that $\ell_{n-i+1} < N$ for all i.)

Proof. We use $Q[i, c, j]$ to denote the elementary operation of multiplying $c \in \mathbb{F}_q$ by the j-th row and then adding the result to the i-th row. We now prove Theorem 2 by using the Mathematics Induction Principle as follows:

Case 1: when $k = 1$. Since V is a valid network code, we may, without loss of generality, assume that V does not contain any column with all zero entries. By the definition of $1-$threshold secure matrix, it suffices to construct an $n \times n$ full rank matrix T such that the n-th row of $S = TV$ has no zero entries.

From elementary linear algebraic knowledge, if $q \geq N$ then there is an $N \times N$ invertible matrix P (right multiplying by P may reorder columns) and an $n \times n$ invertible matrix Q (left multiplying by Q may reorder rows) such that

$$QVP = A = \begin{bmatrix} A_1 \\ \cdots \\ A_n \end{bmatrix} = [a_{ij}],$$

where A^j represents the j-th row of A satisfying following properties;[2] we then have $\ell_n \geq \ell_{n-1} \geq \cdots \geq \ell_s$ for some $1 \leq s \leq n$ such that

[2] Note that A^j is a row vector containing N elements.

- $\sum_{i=s}^{n} \ell_i = N$ (where ℓ_i represents the number of non-zero entries in the i-th row);
- $a_{n1}, ..., a_{n\ell_n}$ are non-zeros (here the n-th row is the row with the largest number, i.e. ℓ_n, of consecutive nonzero entries and that are initialized at the 1-st position);
- $a_{ij} \neq 0$ for $s \leq i < n$ and when $1 + \sum_{m=i+1}^{n} \ell_m \leq j \leq \sum_{m=i}^{n} \ell_m$ (for the i-th row, non-zero entries are consecutive, starting from a column indexed by $1 + \sum_{m=i+1}^{n} \ell_m$);
- $a_{ij} = 0$ for $i \geq s$ and when $j \geq \sum_{m=i}^{n} \ell_m$ (for the i-th row, all entries are zeros with indices starting from $\sum_{m=i}^{n} \ell_m$).

For the convenience of our proof, here we require that all the nonzero entries be consecutive, although this is not necessarily so in actual realization (see our transformation algorithm in Section 5.1). If $\ell_n = N$ and we let $T = Q$, then we are done (where $q \geq N$).

Otherwise (when $\ell_n < N$), since $q \geq N$, one can choose $c_{n-1} \neq 0$ so that the first $\ell_n + \ell_{n-1}$ entries of $c_{n-1}A_{n-1} + A_n$ are non-zeros, while the other entries are all zeros. Inductively, one can choose non-zero numbers $c_s, \cdots, c_{n-1}$ so that all entries of $c_s A_s + \cdots + c_{n-1}A_{n-1} + A_n$ are non-zeros. Therefore, if we let

$$T = Q[n, c_s, s] \cdots Q[n, c_{n-1}, n-1]Q,$$

then T is a nonsingular $n \times n$ matrix such that TV is a 1-threshold secure code.

Case 2: from $k = k'$ to $k = k' + 1$. Given an $n \times n$ nonsingular matrix $\mathcal{Q}_{k'}$ such that for $\mathcal{Q}_{k'}V = \begin{bmatrix} C \\ D \end{bmatrix}$ where D is the lower $k' \times N$ sub-matrix and for any $I \subset [N]$ with $|I| \leq k'$, one has

$$\mathsf{rank}((\mathcal{Q}_{k'}V)_I) = \mathsf{rank}(D_I). \tag{4}$$

Our task is to construct a $\mathcal{Q}_{k'+1}$ such that $\mathcal{Q}_{k'+1}V = \begin{bmatrix} A \\ B \end{bmatrix}$ satisfies

$$\mathsf{rank}((\mathcal{Q}_{k'+1}V)_I) = \mathsf{rank}(B_I) \tag{5}$$

for $\forall I \subset [N]$, where $|I| \leq k' + 1$ and B is the lower $(k' + 1) \times N$ sub-matrix.

If we can obtain $\mathcal{Q}_{k'+1}$, then $T = \mathcal{Q}_{k'+1}$. The construction of T follows and the proof of Theorem 2 is completed.

Now the next task is to find $\mathcal{Q}_{k'+1}$. With the induction assumption from (4) and without loss of generality, one may assume further that

$$\mathsf{rank}(D_I) = k' \text{ if } |I| = k', \text{ and } \mathsf{rank}(\mathcal{Q}'_k V)_I = k' + 1 \text{ if } |I| = k' + 1. \tag{6}$$

Otherwise, one may drop those columns to obtain an $n \times n$ full rank matrix Q' so that $\mathcal{Q}_{k'+1} = Q'\mathcal{Q}'_k$ and (5) holds. This way, it is easy to see that, (5) remains true after adding those removed columns back to $\mathcal{Q}_{k'+1}V$ (by induction).

So far the task has been on how to construct Q' such that $\mathcal{Q}_{k'+1} = Q'\mathcal{Q}'_k$ and (5) holds, under the assumption from (6). For (5) to hold, it suffices that we require any $k'+1$ columns from $\mathcal{Q}_{k'+1}V$ to be linearly independent. We now work towards this goal.

From elementary linear algebra knowledge, we can find an $N \times N$ nonsingular matrix P (right multiplying by P may reorder columns) such that

$$\mathcal{Q}'_k V P = \left[\tilde{A}\ \tilde{B}\right] = \begin{bmatrix} \tilde{A}_1 \\ \cdots \\ \tilde{A}_{n-k'-1} \\ \tilde{B}_{n-k'} \\ \cdots \\ \tilde{B}_n \end{bmatrix} \tag{7}$$

where $\tilde{B}$ is the lower $(k'+1) \times N$ sub-matrix satisfying the following properties. There exist positive integers $\ell_s, \cdots, \ell_{n-k'}$ such that $\ell_s \leq \ell_{s+1} \leq \cdots \leq \ell_{n-k'}$ and $\sum_{j=s}^{n-k'} \ell_j = N$. Moreover, we have

$$\mathsf{rank}(\tilde{B}_I) = \begin{cases} = k'+1 \text{ if } I \subset [\ell_{n-k'}] \ \text{ where } \ |I| = k'+1; \\ < k'+1 \text{ if } I \not\subset [\ell_{n-k'}] \ \text{ where } \ |I| = k'+1; \end{cases} \tag{8}$$

and for $\forall\ \ 1 \leq j \leq n-k'-1$, it holds that

$$\mathsf{rank} \begin{bmatrix} \tilde{A}_j \\ \cdots \\ \tilde{A}_{n-k'-1} \\ \tilde{B} \end{bmatrix}_I = \begin{cases} k'+1, \text{ if } I \subset [\sum_{m=j}^{n-k'} \ell_m] \text{ and } |I| = k'+1 \\ < k'+1, \text{ if } I \not\subset [\sum_{m=j}^{n-k'} \ell_m] \text{ and } |I| = k'+1 \end{cases}. \tag{9}$$

Remark 1. From (8), we can infer that right multiplying by P realizes the swapping of the $(n-k')$-th row with a row (say y) from C, such that the new sub-matrix consisting of y and the sub-matrix D has the largest number (denoted as $\ell_{n-k'}$) of columns where any $k'+1$ columns are linearly independent. On the other hand, (9) tells us that right multiplying by P realizes the swapping of s-th row with a row (say z) so that the new sub-matrix, formed as a union of z and the last $n-s$ rows from the current matrix, has the largest number (denoted as ℓ_s) of columns where any $(k'+1)-$ columns are linearly independent.

If $\ell_{n-k'} = N$, we are done (where $q \geq N_{k'}$ and $T = \mathcal{Q}_{k'}$). Otherwise when $\ell_{n-k'} \leq N-1$, we claim that there is a number $c = c_{n-k'-1} \in \mathbb{F}_q$ such that if

$$E := \begin{bmatrix} c\tilde{A}_{n-k'-1} + \tilde{B}_{n-k'} \\ \tilde{B}_{n-k'+1} \\ \cdots \\ \tilde{B}_n \end{bmatrix} \tag{10}$$

then E_I has full rank when $q \geq \binom{\ell_{n-k'}}{k'+1}$, where $I \subset \{1, \cdots, \ell_{n-k'} + \ell_{n-k'-1}\}$ and $|I| \leq k'+1$. In fact, for $I \subset \{1, \cdots, \ell_{n-k'} + \ell_{n-k'-1}\}$ where $|I| = k'+1$, we need to prove there is indeed a $c \in \mathbb{F}_q$ such that

$$\mathsf{det}(E_I) \neq 0. \tag{11}$$

Notice that

$$\mathsf{det}(E_I) = \mathsf{det}(\tilde{B}_I) + c\mathsf{det}\left(\begin{bmatrix} \tilde{A}_{n-k'-1} \\ \tilde{B}_{n-k'+1} \\ \cdots \\ \tilde{B}_n \end{bmatrix}_I \right) = \mathsf{det}(\tilde{B}_I) + ch_I. \tag{12}$$

By assumptions from (6), (8) and (9), there are at most $\binom{\ell_{n-k'}}{k'+1} - 1$ integers $h_I \neq 0$. Since $q \geq \binom{\ell_{n-k'}}{k'+1}$, we can always choose an integer $c = c_{n-k'-1} \neq 0$ so that (11) holds when $I \subset \{1, 2, \cdots, \ell_{n-k'}\}$. On the other hand, when $I \subset \{1, \cdots, \ell_{n-k'} + \ell_{n-k'-1}\}$ and $I \cap \{\ell_{n-k'} + 1, \cdots, \ell_{n-k'} + \ell_{n-k'-1}\} \neq \emptyset$, we have $h_I \neq 0$ and $\mathsf{det}(D_I) = 0$. Therefore, when $q \geq \binom{\ell_{n-k'}}{k'+1}$, it is easy to see that there always exists an integer $c = c_{n-k'-1}$ so that $\mathsf{det}(E_I) \neq 0$ with $|I| = k' + 1$ and $I \subset \{1, \cdots, \ell_{n-k'} + \ell_{n-k'-1}\}$.

Repeating the above process in a similar way, when $q \geq \binom{\ell_{n-k'}}{k'+1}$, there are integers $c_s, \cdots, c_{n-k'-1}$ such that

$$Q'\mathcal{Q}'_k VP =: Q[n-k', c_s, s] \cdots Q[n-k', c_{n-k'-1}, n-k'-1]\mathcal{Q}'_k VP = \begin{bmatrix} \tilde{A} \\ \tilde{B} \end{bmatrix}$$

satisfies $\mathsf{det}(\tilde{B}_I) \neq 0$ for any $I \subset \{1, 2, \cdots, N\}$ with $|I| = k' + 1$. We then set

$$T = Q'\mathcal{Q}'_k \quad \text{and } TV = \begin{bmatrix} A \\ B \end{bmatrix}$$

where $\mathsf{det}(\tilde{B}_I) \neq 0$ for any $I \subset \{1, 2, \cdots, N\}$ with $|I| = k' + 1$. We then add the columns which we removed back in order to satisfy (6). The resulting new matrix $\begin{bmatrix} A \\ B \end{bmatrix}$ still satisfies 5. Therefore, $S = TV$ is a $(k'+1)$-threshold secure matrix and thus is a $(k'+1)$-threshold secure network code. This completes our proof for Theorem 2. □

5.1 Our Transformation Algorithm

The proof of Theorem 2 above not only ensures the existence of T when $q \geq N_k$, but also provides an algorithm to construct T. We can find T by using the Mathematics Induction Principle as below.

Case 1: when $k = 1$. We find T in four steps as follow:

Step 1 – Exchange: We exchange the n-th row with a row containing the largest number (denoted as ℓ_n) of nonzero entries. We denote this operation as Q_1^0 and the resulting matrix consists of $\{\xi_i\}_{i=1}^n$ (ξ_i is the i-th row). If $\ell_n = N$ and let $T = Q_1^0$, then we are done; otherwise, we proceed to Step 2 below.

Step 2 – Sort: We sort the remaining rows in a following way:

(i) Initialize s to be $n-1$.

(ii) Swap the s-th row with a row containing the largest number (denoted as ℓ_s) of nonzero entries in terms of the column set where all the $(s+1)$-th $,\ldots,n$-th entries are zeros. If $\sum_{i=s}^{n}\ell_i = N$ we exist this step and go directly to Step 3.

(iii) Subtract s by 1.

Repeat (ii) and (iii) above until we exist from Step 2.

Step 3 – Find coefficients: We then find the coefficients $c_s,\cdots c_{n-1}$ such that $\zeta_n = \xi_n + \sum_{j=s}^{n-1} c_j\xi_j$ has no zero entries.

Step 4 – Output T: We output $T = Q[n,c_s,s]\cdots Q[n,c_{n-1},n-1]Q_1^0$.

Case 2: from $k=k'$ to $k=k'+1$ $(k'\leq(k-1))$. Given an $n\times n$ nonsingular matrix $\mathcal{Q}'_k$ such that $\mathcal{Q}_{k'}V$ is a k'-threshold secure matrix, our goal is to find a $\mathcal{Q}_{k'+1}$ such that $T=\mathcal{Q}_{k'+1}$ is sufficient for $k=k'+1$ in terms of security.

Step 1 – Exchange: We exchange the $(n-k')$-th row with a row (say y) satisfying such the following property: the new sub-matrix, formed as a union of y and the last k' rows of $\mathcal{Q}_{k'}V$, has the largest number (denoted as $\ell_{n-k'}$) of columns where any $k'+1$ columns are linearly independent. We denote this exchange operation by Q'. If $\ell_{n-k'}=N$ and let $T=\mathcal{Q}_{k'+1}=Q'\mathcal{Q}_{k'}$, then we are done; otherwise, we go to Step 2 below.

Step 2 – Sort: We sort the remaining rows as follows:

(i) Initialize s to be $n-k'-1$.

(ii) Swap the s-th row with a row (say z) satisfying the following property: the new sub-matrix, formed as a union of z and the last $n-s$ rows from the current matrix, has the largest number (denoted as ℓ_s) of columns where any $k'+1$ columns are linearly independent. If $\sum_{i=s}^{n-k'}\ell_i=N$, we exit and go to Step 3.

(iii) Subtract s by 1.

Repeat (ii) and (iii) above until we exist from Step 2.

Step 3 – Find coefficients: We find the coefficients $c_s,\cdots,c_{n-k'-1}$ for rows indexed by $s,\cdots,n-k'-1$ respectively, such that for

$$Q[n-k',c_s,s]\cdots Q[n-k',c_{n-k'-1},n-k'-1]Q'\mathcal{Q}_{k'}V=\begin{bmatrix}A\\B\end{bmatrix},$$

where B represents the last $k'+1$ rows, we have $\mathsf{det}(B_I)\neq 0$ for any $I\subset\{1,2,\cdots,N\}$ with $|I|=k'+1$.

Step 4 – Output T: We output $T=\mathcal{Q}_{k'+1}=Q[n-k',c_s,s]\cdots Q[n-k',c_{n-k'-1},n-k'-1]Q'\mathcal{Q}_{k'}$.

Combining Cases 1 and 2 above, we can find a T that gives us a sufficient condition for V to be a k-threshold secure matrix.

5.2 Efficiency Analysis

BOUNDS ON q. We give a lemma that compares our bound with that of Cai and Yeung [22], and Rouayheb and Soljanin [24].

Lemma 1. *Our bound, defined as* $max\left\{\binom{\ell_{n-i+1}}{i}, N : i = 2, \cdots, k\right\}$ *is independent of the network edge size and dependent only on the information length and the size of sink set. Thus, our bound is smaller than Cai and Yeung' bound when* $k \ll N$. *Moreover, our bound is as optimal as Rouayheb and Soljanin's bound.*

The proof for Lemma 1 and further analysis are provided in Appendix B.

TIME COMPLEXITY. If we implement Step 3 of our transformation algorithm shown in Section 5.1 (for both two cases) by randomly choosing elements from $\mathbb{F}_q$ until the relevant requirements are satisfied, then the time complexity of the algorithm is $O\binom{N}{k}$. This is equivalent to that of [22].

6 Conclusions

Building on the previous work by Cai and Yeung, and Feldman et al. we further studied the application and connection of secret sharing to network coding security. Particularly, we defined a new concept called k-threshold secure matrix that allows us to determine the security condition of a given network code for an underlying network over a certain field. Moreover, we proposed a concrete algorithm that transforms a network code into a secure one. Our algorithm achieves the required optimal bandwidth as with the currently best available result in the literature.

Acknowledgements. Research of this work is supported in part by the National Research Foundation of Singapore under Research Grant NRF-CRP2-2007-03.

References

1. Ahlswede, R., Cai, N., Li, S.Y.R., Yeung, R.W.: Network information flow. IEEE Trans. on Information Theory 46(4), 1204–1216 (2000)
2. Li, S.Y.R., Yeung, R.W., Cai, N.: Linear network coding. IEEE Trans. on Information Theory 49(2), 371–381 (2003)
3. Carr, B., Vempala, S.: Randomized meta-rounding. In: Proc. 32nd ACM Symp. Theory of Computing, pp. 58–62 (May 2000)
4. Jain, K., Mahdian, M., Salavatipour, M.R.: Packing Steiner trees. In: Proc. 14th ACM-SIAM Symposium on Discrete Algorithms (SODA) (January 2003)
5. Fragouli, C., Soljanin, E.: Network Coding Fundamentals. Now Publishers Inc. (2007)
6. Chou, P., Wu, Y., Jain, K.: Practical Network Coding. In: Allerton Conference on Communication, Control and Computing (2003)

7. Avalanche: File swarming with network coding, http://research.microsoft.com/pablo/avalanche.aspx
8. Shannon, C.E.: Communication theory of secrecy system. Bell Syst. Techn. J., 656–715 (1949)
9. Gkantsidis, C., Rodriguez, P.: Network coding for large scale content distribution. In: IEEE INFOCOM (2005)
10. Gkantsidis, C., Miller, J., Rodriguez, P.: Comprehensive view of a live network coding P2P system. In: ACM SIGCOM Conference on Internet Measurements (2006)
11. Chen, H.: Distributed File Sharing: Network Coding Meets Compressed Sensing. In: IEEE CHINACOM (2008)
12. Dimakis, A.G., Godfrey, P.B., Wainwright, M.J., Ramchandran, K.: Network Coding for Distributed Storage Systems. In: IEEE INFOCOM (2007)
13. Dong, J., Curtmola, R., Sethi, R., Nita-Rotaru, C.: Toward secure network coding in wireless networks: Threats and challenges. In: NPSec (2008)
14. 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)
15. 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, pp. 292–305. Springer, Heidelberg (2009)
16. Oggier, F., Fathi, H.: An Authentication Code against Pollution Attacks in Network Coding. CoRR abs/0909.3146 (2009)
17. Jaggi, S., Langberg, M., Katti, S., Ho, T., Katabi, D., Medard, M.: Resilient network coding in the presence of Byzantine adversaries. In: Proc. 26th Annual IEEE Conf. on Computer Commun., INFOCOM, pp. 616–624 (2007)
18. Kotter, R., Kschischang, F.R.: Coding for errors and erasures in random network coding. IEEE Trans. on Information Theory 54(8), 3579–3591 (2008)
19. Cai, N., Yeung, R.W.: Network coding and error correction. In: Proc. 2002 IEEE Inform. Theory Workshop, pp. 119–122 (October 2002)
20. Bhattad, K., Narayanan, K.R.: Weakly secure network coding. In: Proc. NetCod (April 2005)
21. Wei, Y., Yu, Z., Guan, Y.: Efficient Weakly-Secure Network Coding Schemes against Wiretapping Attacks. In: Proc. NetCod (June 2010)
22. Cai, N., Yeung, R.W.: Secure Network Coding on a Wiretap Network. IEEE Trans. on Information Theory 57(1), 424–435 (2011)
23. Feldman, J., Malkin, T., Stein, C., Servedio, R.A.: Secure network coding via filtered secret sharing. In: Proc. 42nd Annual Allerton Conf. Commun., Control and Comput. (September 2004)
24. Rouayheb, S.Y.E., Soljanin, E.: Secure Network Coding for Wiretap Networks of Type II. IEEE Trans. on Information Theory 58(3), 1361–1371 (2012)
25. Silva, D., Kschischang, F.R.: Security for wiretap networks via rank-metric codes. In: IEEE International Symposium on Information Theory (July 2008)
26. Shioji, E., Matsumoto, R., Uyematsu, T.: Vulnerability of MRD-code-based universal secure network coding against stronger eavesdroppers. In: Proc. 2010 IEEE ISIT, pp. 2433–2437 (June 2010)
27. Ozarow, L.H., Wyner, A.D.: Wire-Tap Channel II. In: Beth, T., Cot, N., Ingemarsson, I. (eds.) EUROCRYPT 1984. LNCS, vol. 209, pp. 33–50. Springer, Heidelberg (1985)
28. Blakley, G.R.: Safeguarding cryptography keys. In: Proc. of the 1979 AFIPS national Computer Conference, vol. 48, pp. 313–317 (1979)

29. Ho, T., Koetter, R., Médard, M., Karger, D.R., Effros, M.: The benefits of coding over routing in a randomized setting. In: IEEE International Symposium on Information Theory (June 2003)
30. Jaggi, S., Sanders, P., Chou, P.A., Effros, M., Egner, S., Jain, K., Tolhuizen, L.: Polynomial time algorithms for multicast network code construction. IEEE Wireless Communications, 68–71 (February 2004)
31. Fragouli, C., Soljanin, E.: Information flow decomposition for network coding. IEEE Trans. on Information Theory, 829–848 (March 2006)
32. Langberg, M., Sprintson, A., Bruck, J.: The encoding complexity of network coding. IEEE Trans. on Information Theory 52(6), 2386–2397 (2006)
33. Langberg, M., Sprintson, A., Bruck, J.: Network coding: A computational perspective. IEEE Trans. on Information Theory 55(1), 147–157 (2009)
34. Shamir, A.: How to share a secret. Communications of the ACM, 612–613 (1979)
35. Stinson, D.R.: An explication of secret sharing schemes. Designs, Codes and Cryptography 2, 235–390 (1992)
36. Blakley, G.R., Meadows, C.: The Security of RAMP Schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 242–268. Springer, Heidelberg (1985)
37. Lun, D.S., Ratnakar, N., Medard, M., Koetter, R., Karger, D.R., Ho, T., Ahmed, E., Zhao, F.: Minimum-cost multicast over coded packet networks. IEEE Trans. on Information Theory 52(6), 2608–2623 (2006)

A Examples of k-Threshold Secure Matrices

Let matrix S be a generator matrix for some $\mathbb{F}_q$-linear code $\mathcal{C}(N, n)$.

Example 1. S satisfies that S_I^L is a generator matrix for some maximum distance separable (MDS) code [24, 27] $\mathcal{C}_0(N, k, N-k+1)$.

We claim that S shown in Example 1 is a k-threshold secure matrix. Indeed, since $\mathcal{C}_0$ is an MDS code, then any k columns from S_I^L are linearly independent and thus full rank. This implies that S is a $k-$threshold secure matrix.

Example 2. More generally, there exist other examples of k-threshold secure matrices if $\mathcal{C}_0$ is a $\mathbb{F}_q$-linear code but not restricted to an MDS code. In fact, we can construct a more general $k-$threshold secure matrix S in the following way. Firstly, we observe that, as a generator matrix for an $\mathbb{F}_q$-linear code $\mathcal{C}_0(N, k)$, S_I^L can be described as follows:

$$S_I^L = \begin{bmatrix} I_k & A & B \end{bmatrix}$$

where I_k denotes the identity matrix with $\mathsf{rank}\ k$ (over $\mathbb{F}_q$) and A is a $k \times (n-k)$ sub matrix. Secondly, we require that any k columns from $[I_k\ A]$ or $[A\ B]$ are linearly independent. Note that, here, we do not rely on the linear independency of k-tuple from $[I_k\ B]$ for $\mathcal{C}_0$ to be an MDS code. Finally, we construct S as follows:

$$S = \begin{bmatrix} 0 & I_{n-k} & D \\ I_k & A & B \end{bmatrix}$$

such that for any $1 \leq k' < k$ and any $I' \subset [1, ..., N-n]$ with $|I'| = k'$:

- if there exist not all zero constants $c_{i_1}, ..., c_{i_{k'}}$ such that $\sum_{j=1}^{k'} c_{i_j} B_{I'}$ is a linear combination of some columns from I_k, then D is chosen in such a way that with same constants $c_{i_1}, ..., c_{i_{k'}}$ and the same I' we have $\sum_{j=1}^{k'}(c_{i_j} D_{I'}) = \mathbf{O}$.
- Otherwise, D is arbitrarily chosen.

S constructed as in Example 2 is always a $k-$threshold secure matrix. The proof is omitted due to space constraints.

B Proof of Lemma 1

We give a comparison between our work and that of [22] and [23] in terms of the required secure code alphabet size.

Since the general bound in [23] was derived in a network with a sacrificed capacity and its security is attained with a probability that is less than 1 (although a high probability), here we look into only Cai and Yeung's bound [22].

The bound derived in our approach is tighter than Cai and Yeung's bound if $k \ll N$. The reason is that for $k \geq 2$, our lower bound on q is $max\left\{\binom{\ell_{n-i+1}}{i}, N : i = 2, \cdots, k\right\}$ and we have

$$max\left\{\binom{\ell_{n-i+1}}{i}, N : i = 2, \cdots, k\right\} < max_{i=2}^{k}\binom{N}{i} = \binom{N}{k} \quad (\text{if} \quad k \ll N)$$

where $\binom{N}{k}$ is Cai and Yeung's lower bound. It is worth mentioning that in cases where a large portion of column vectors from the code are linearly independent from each other and thus $\ell_{n-i+1} \ll N$, our bound is much tighter than Cai and Yeung's bound.

We would also want to point out that the condition above, $k \ll N$, is easily satisfied in reality, particularly in large-scaled networks. Since in most network instances it holds that $k < n \ll N$, especially in huge networks. Based on this, we claim that we have a tighter bound than [22].

Further, we compare our bound against that of the coset coding approach. Since [25] is not fully information-theoretically secure and the bound of [24] gives a smaller code than Cai and Yeung's bound, we compare ours with that of [24]. The smaller bound in [24] was obtained from a network code constructed in a more efficient way [33, 31, 32]. In the more efficient network code construction, the authors observed that all the network edges are classified into two parts: *encoding edges* that employ coding operations on incoming links and *forwarding edges* that can only forward incoming packets. A further observation made in [24] was that the set of edges which the wiretapper might have access to consist of encoding edges and edges leaving the source node. Combining with our definition of ℓ_{n-i+1} for all i, it is not hard to check that for each i, ℓ_{n-i+1} is upper bounded by the size of the union of encoding edges and edges leaving the source node. Therefore, similarly to Corollary 1 in [24] where the alphabet size was shown as:

$q \geq (\binom{2t^3|T_G|^2}{k-1} + |T_G|)$, we have that if V is provided in a same way as [24], we reach a bound $q \geq max_{i=2}^{k} \binom{2t^3|T_G|^2}{i}$. In this way, the bound we derive is also independent of the network edge size and only dependent on the information length and the size of sink set and thus it is as optimal Rouayheb and Soljanin's bound. □

Codes Based Tracing and Revoking Scheme with Constant Ciphertext

Xingwen Zhao and Hui Li

State Key Laboratory of ISN, Xidian University
Xi'an 710071, China
School of Telecommunications Engineering, Xidian University
Xi'an 710071, China
sevenzhao@hotmail.com, lihui@mail.xidian.edu.cn

Abstract. Traitor tracing is needed because some users in broadcast encryption system may give out their decryption keys to construct pirate decoders. Many codes based traitor tracing schemes were proposed. However, as stated by Billet and Phan in ICITS 2008, they were lacking in revocation ability. In this paper, we bring forward a codes based tracing and revoking scheme. Revocation ability helps to disable identified traitors and users who fail to fulfill the payments in each broadcast, so that the broadcast encryption system can be more practical. Based on Park et al.'s public key broadcast encryption scheme, we embed collusion secure code into each user's decryption keys so as we can send messages to a set of designated receivers while at the same time we can recover information of codeword from the feedback of the pirate decoder by employing Boneh and Naor's traitor tracing method. Our scheme achieves constant-size ciphertext which makes it suitable for situations where bandwidth is precious. Our scheme is based on collusion secure codes, and it can be extended to adopt other codes such as identifiable parent property (IPP) codes. Our method presents an answer to the problem left open by Billet and Phan.

Keywords: Broadcast encryption, collusion secure codes, copyright protection, traitor tracing, user revocation.

1 Introduction

Broadcast encryption provides a convenient method to distribute digital content to subscribers over an insecure broadcast channel so that only the qualified users can recover the data. Broadcast encryption is quite useful and enjoys many applications including pay-TV systems, distribution of copyrighted materials such as DVD. Because some users (called traitors) may give out their decryption keys to construct pirate decoders, the ability of traitor tracing is needed for broadcast encryption system.

The first traitor tracing scheme against pirate decoders was presented by Chor, Fiat and Naor in [12]. Since then, many traitor tracing schemes against pirate decoders were proposed and they can be roughly classified into three categories.

T. Takagi et al. (Eds.): ProvSec 2012, LNCS 7496, pp. 318–335, 2012.

The first category is called combinatorial, as in [12,28,16,22]. These schemes carefully choose some subsets of keys to be put in each decoder box. By analyzing the keys used in a pirate decoder, it is possible to trace back to one of the traitors. Another category is called algebraic, as in [20,6,23,21,14,8,9,17,25,26]. These schemes use algebraic method to assign private keys to users, and the broadcasting can be done in public since public-key techniques are used. Collusion secure codes based schemes can be regarded as the third category, which combines ideas from the two previous classes. For instance, [19,10,2,7,3,11,29] belong to this category. These schemes assign keys to each user according to each bit of his/her codeword. By analyzing the keys used in each bit position, the tracer can recover the codeword embedded in the decoder and trace back to at least one of the traitors. However, it is pointed out in [3] that it is still an open problem whether codes based traitor tracing scheme can achieve revocation. As we observed, these codes based traitor tracing scheme mentioned above lacked revocation ability.

Motivated by this problem, our paper mainly focuses on constructing codes based traitor tracing scheme with revocation ability. Revocation ability is a useful property in many circumstances. For instance, when traitors are found, it is desirable to make them useless so as to disable the pirate decoders generated by them. In many broadcast encryption based commercial applications, such as Pay-TV system, it is necessary to find ways to make sure that only those consumers who fulfill the payment are capable to recover the messages. So revocation ability is needed to temporarily rule out the users in arrear with payment.

Our idea is inspired by Park et al.'s public key broadcast encryption scheme [24] and Boneh and Naor's traitor tracing scheme [7]. We carefully embed collusion secure code into each user's decryption key of Park et al.'s scheme, so as we can take advantages of Park et al.'s scheme that the sender can send messages to a set of designated receivers. At the same time, we can make use of the traitor tracing method by Boneh and Naor to recover information of embedded codeword from the pirate decoder. The advantages of our scheme over previous codes based schemes are that the ciphertext length is constant, and it achieves revocation ability while previous codes based schemes did not. We also show that our scheme is resistant to public collaboration attacks [4].

Public collaboration attacks were introduced in [4] as that traitors collaborate in a public way. In other words, traitors do not secretly collude and they release part of their private keys in a public place so that pirate maker can collect these partial keys to build useful pirate decoders. Billet and Phan [4] show that such type of attacks presents a real threat for subset cover based schemes (such as [22]) and code based schemes (such as [19]). Each traitor remains anonymous because a large number of users contain the same keys as those released in public. The partial key is referred as decryption key for a specified subset in subset cover based schemes and as decryption key for a single codeword bit in code based schemes. There are several schemes [2,29,13,27] that are resistant to such attacks. In schemes of [2,29,27], wildcarded identity based encryption [1] are used to connect all separated parts of each user's identity so the decryption

key should be released as a whole in order for the key to be useful. Releasing decryption key as a whole will immediately identify the traitor. In scheme of [13], secret sharing skill is used to bind each user's distinct identity to decryption keys of subsets that it belongs to. Thus, releasing decryption key of any subset will immediately identify the traitor. In our scheme, decryption key for each codeword bit is embedded with user's identity so each key is self-enforced.

The proposed scheme is proved secure under decisional v-mBDH assumption and random oracle model. We show that it is easy to remove the random oracle in the succeeding discussions. Finally, we describe some extensions to our scheme, such as scheme based on identifiable parent property (IPP) codes [18] and scheme against imperfect decoders.

2 Preliminaries

2.1 Collusion Secure Codes

We first review the definition of collusion secure codes required for constructing our traitor tracing scheme. The definition is similar to that in [7].

- For a word $\bar{w} \in \{0,1\}^L$ we write $\bar{w} = \bar{w}_1 \dots \bar{w}_L$, where $\bar{w}_i \in \{0,1\}$ is the ith bit of $\bar{w}$ for $i = 1, \dots, L$.
- Let $W = \{\bar{w}^{(1)}, \dots, \bar{w}^{(t)}\}$ be a set of words in $\{0,1\}^L$. We say that a word $\bar{w} \in \{0,1\}^L$ is feasible for W if for all $i = 1, \dots, L$ there is a $j \in \{1, \dots, t\}$ such that $\bar{w}_i = \bar{w}_i^{(j)}$. For example, if W consists of the two words $\binom{0\ 1\ 1\ 0\ 0}{0\ 0\ 1\ 1\ 0}$, then all words of the form $[0\ \binom{1}{0}\ 1\ \binom{0}{1}\ 0]$ are feasible for W.
- For a set of words $W \subseteq \{0,1\}^L$ we say that the feasible set of W, denoted $F(W)$, is the set of all words that are feasible for W.

The collusion secure code can be denoted with a pair of polynomial time algorithms (G, T) defined as follows:

- Algorithm G, called a code generator, is a probabilistic algorithm that takes a pair (N, ϵ) as input, where N is the number of words to output and $\epsilon \in (0,1)$ is a security parameter. The algorithm outputs a pair (Γ, TK). Here Γ (called a code) contains N words in $\{0,1\}^L$ for some $L > 0$ (called the code length). TK is called the tracing key.
- Algorithm T, called a tracing algorithm, is a deterministic algorithm that takes as input a pair $(\bar{w}^*, \mathrm{TK})$ where $\bar{w}^* \in \{0,1\}^L$. The algorithm outputs a subset S of $\{1, \dots, N\}$. Informally, elements in S are accused of creating the word $\bar{w}^*$.

The collusion resistant property of collusion secure code (G, T) is defined using the following game between a challenger and an adversary. Let N be an integer and $\epsilon \in (0,1)$. Let C be a subset of $\{1, \dots, N\}$. Both the challenger and adversary are given (N, ϵ, C) as input. Then the game proceeds as follows:

1. The challenger runs $G(N, \epsilon)$ to obtain (Γ, TK) where $\Gamma = \{\bar{w}^{(1)}, \dots, \bar{w}^{(N)}\}$. It sends the set $W := \{\bar{w}^{(i)}\}_{i \in C}$ to the adversary.
2. The adversary outputs a word $\bar{w}^* \in F(W)$.

We say that the adversary $\mathcal{A}$ wins the game if $T(\bar{w}^*, \mathrm{TK})$ is empty or not a subset of C. We denote $Adv_{CR}^{\mathcal{A},G(N,\epsilon),T,C}$ as the advantage that $\mathcal{A}$ wins the collusion resistant game.

A collusion secure code (G, T) is said to be fully collusion resistant if for all polynomial time adversaries $\mathcal{A}$, all $N > 0$, all $\epsilon \in (0, 1)$, and all $C \subseteq \{1, \ldots, N\}$, we have $Adv_{CR}^{\mathcal{A},G(N,\epsilon),T,C}$ is negligible (less than ϵ).

A collusion secure code (G, T) is said to be t-collusion resistant if for all polynomial time adversaries $\mathcal{A}$, all $N > t$, all $\epsilon \in (0, 1)$, and all $C \subseteq \{1, \ldots, N\}$ of size at most t, we have $Adv_{CR}^{\mathcal{A},G(N,\epsilon),T,C}$ is negligible (less than ϵ).

Our readers can refer to [7] for known results on collusion secure codes. Additionally, Boneh and Naor [7] also constructed δ-robust Boneh-Shaw codes in order to trace high error-rate pirate decoders.

2.2 Bilinear Pairings and Complexity Assumption

We review the definition of bilinear pairings [5] and the decisional v-modified Bilinear Diffie-Hellman assumption [24] in brief.

Bilinear Pairings: Let $\mathbb{G}$ be a (multiplicative) cyclic group of prime order p and g is a generator of $\mathbb{G}$. A one-way map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear pairing if the following conditions hold.

- Bilinear: For all $u, v \in \mathbb{G}$, and $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$, i.e., if g generates $\mathbb{G}$, then $e(g, g)$ generates $\mathbb{G}_T$.
- Computability: There exists an efficient algorithm for computing $e(u, v)$, $\forall u, v \in \mathbb{G}$.

The decisional v-Modified Bilinear Diffie-Hellman Assumption: The decisional v-Modified Bilinear Diffie-Hellman (v-mBDH) problem [24] is: given a tuple $TP = (g, z, g^{1/(x+1)}, \ldots, g^{1/(x+v)}, wg^{1/(x+1)^2}, \ldots, wg^{1/(x+v)^2}) \in \mathbb{G}^{2v+2}$ and a $T \in \mathbb{G}_T$ as input, to decide whether $T = e(w, z)$.

We say an algorithm $\mathcal{B}$ that outputs $b \in \{0, 1\}$ has advantage ϵ in solving the decisional v-mBDH problem in $\mathbb{G}$ if

$$|Pr[\mathcal{B}(TP, e(w, z))] = 0| - |Pr[\mathcal{B}(TP, T)] = 0| \geq \epsilon,$$

where the probability is taken over the random choices of $w, z \in \mathbb{G}$, the random choice of x in $\mathbb{Z}_p$, the random choice of $T \in \mathbb{G}_T$, and the random decision b of $\mathcal{B}$.

We say that the decisional v-mBDH assumption holds if no t-time algorithm has advantage of at least ϵ in solving the decisional v-mBDH problem.

2.3 Protocol Model

The protocol model for our traitor tracing scheme consists of four algorithms (Setup, Encrypt, Decrypt, Trace) described as follows.

- **Setup**(1^λ, N). It is a probabilistic algorithm that given 1^λ and the number of users in the system N, outputs the public parameter PK, a secret trace-key TK, and the private user-key SK_u for each user $u \in \{1, \ldots, N\}$.
- **Encrypt**(PK, S, TSK). It is a probabilistic algorithm that given the public parameter PK, a set of designated receivers S and a temporary session key TSK, a broadcast ciphertext header Hdr is generated. The messages are symmetrically encrypted using the session key TSK to generate ciphertext body C. We refer to the ciphertext length as the length of Hdr in which a session key is protected and distributed by the traitor tracing scheme.
- **Decrypt**(SK_u, S, Hdr, PK). It is an algorithm that given a broadcast ciphertext header Hdr, the set of designated receivers S and the set of private keys SK_u of user u, returns the recovered temporary session key TSK or $\perp$. TSK is used to decrypt the messages from ciphertext body C.
- **Trace**$^{\mathcal{D}}$(TK). It is an algorithm that given a pirate decoder $\mathcal{D}$ and private trace-key TK, it queries decoder $\mathcal{D}$ as a black-box oracle and then outputs a set of traitors $T \subseteq \{1, \ldots, N\}$.

2.4 Security Requirements

- **Correctness**. Each honest user is able to recover the messages in normal broadcasting, if he/she is included in the set of designated receivers.
- **Semantic Security**. The users cannot obtain any information of messages encrypted in the broadcast ciphertext, if their identities are not included in the specified receiver set.

  The semantic security of the proposed traitor tracing scheme is defined using the following game between a challenger and an adversary. The game proceeds as follows:

  1. The challenger runs $G(N, \epsilon)$ to obtain (Γ, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ and $\bar{w}^{(i)}$ is the codeword for user i. The challenger also generates public parameters PK. It sends Γ and PK to the adversary.
  2. The adversary selects a subset of $\{1, \ldots, N\}$, denoted as C. The adversary can query the challenger for decryption keys of the users in C. The challenger generates the keys and gives them to the adversary.
  3. The adversary submits two messages (m_1, m_0) and a set of users S for challenging, with $u^* \notin C$ for each $u^* \in S$. In selective adversary model, S should be submitted to the challenger at the beginning of the game. The challenger chooses a fair coin $b \in \{0, 1\}$ and encrypts m_b using S as the set of designated receivers. The ciphertext is sent to the adversary, and the adversary is required to output a guessed bit b'.

  If $b' = b$, the adversary wins the game.
- **Collusion Resistant**. Collusion of users cannot produce a decoder that cannot be traced to any of these users.

  The collusion resistant property of the proposed traitor tracing scheme is defined using the following game between a challenger and an adversary. Let (G, T) be a collusion secure code. Let N be an integer and $\epsilon \in (0, 1)$. Then the game proceeds as follows:

1. The challenger runs $G(N, \epsilon)$ to obtain (Γ, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ and $\bar{w}^{(i)}$ is the codeword for user i. The challenger also generates public parameters PK. It sends Γ and PK to the adversary.
2. The adversary selects a subset of $\{1, \ldots, N\}$, denoted as C. The adversary can query the challenger for decryption keys of the users in C. The challenger generates the keys and gives them to the adversary.
3. The challenger selects receivers set $S \subseteq \{1, \ldots, N\}$ for which the decoder can recover the messages, and asks the adversary to decrypt ciphertexts for S a number of times. Finally the challenger recovers a codeword $\bar{w}^*$.

We say that the adversary $\mathcal{A}$ wins the game if $T(\bar{w}^*, \text{TK})$ is empty or not a subset of $C \cap S$.

– **Public Collaboration Resistant**. The property of public collaboration resistance requires that the release of any partial key in public by any user should lead to the detection of the identity of the corresponding user.

3 Tracing and Revoking Scheme Based on Collusion Secure Codes

In this section, we present our collusion secure codes based traitor tracing and revoking scheme with constant ciphertext. Revocation ability is achieved by means of encrypting messages only to the set of designated receivers, so all other users not in the set are revoked at current broadcasting. Constant ciphertext is achieved by using the method of Boneh and Naor [7], where only one bit of the codeword is used in each time of broadcasting.

3.1 The Scheme

Let $\mathbb{G}$ and $\mathbb{G}_T$ be bilinear groups of some large prime order p and let $e : \mathbb{G} \times \mathbb{G} \mapsto \mathbb{G}_T$ be a bilinear map. We also assume that the messages are encrypted symmetrically using the session key from $\mathbb{G}_T$. The traitor tracing scheme works as follows:

– **Setup**$(1^\lambda, N)$. A trusted party, given 1^λ and the number of users in the system N, selects $\epsilon \in (0, 1)$ and runs collusion secure code generation algorithm $\mathbf{G}(N, \epsilon)$ to generate a pair (Γ, TK). The set $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ contains N codewords in $\{0, 1\}^L$, where L is the codeword length. TK is the tracing key for Γ. $\bar{w}^{(u)}$ is assigned to user u, with $1 \leq u \leq N$. A random generator $g \in_R \mathbb{G}$ and a random element $w \in_R \mathbb{G}$ are selected. $H(\cdot) : \{1 \ldots L\} \times \{0, 1\} \to \mathbb{G}$ is a collision resistant hash function (the selection is out of scope of this paper). A random $x \in \mathbb{Z}_p$ is selected. $g_i = g^{1/(x+i)}$ is computed for each $i = 1, \ldots, N$. The master public parameter PK is $(\mathbb{G}, \mathbb{G}_T, p, g, g_1, \ldots, g_N, e(w, g), H(\cdot))$. (x, w) are kept private. For each user $u \in \{1, \ldots, N\}$, a set of decryption keys is generated as:

$$d_{u,i} = w \cdot (H(i, \bar{w}_i^{(u)}))^{\frac{1}{x+u}} \cdot g^{\frac{1}{(x+u)^2}}, \forall i = 1, \ldots, L.$$

Each set of decryption keys $\mathrm{SK}_u = (d_{u,1}, \ldots, d_{u,L})$ is transferred to the corresponding user u over a secure and authenticated channel which is not considered in this paper.

– **Encrypt**(PK, S, TSK). This algorithm is used for normal broadcasting. Given the public parameter PK and a set of designated receivers $S \subseteq \{1, \ldots, N\}$, the sender selects two random numbers $s_1, s_0 \in_R \mathbb{Z}_p$ and sets $K_1 = e(w, g)^{s_1}$, $K_0 = e(w, g)^{s_0}$. Then the sender selects a position $i \in \{1, \ldots, L\}$ and a random temporary session key $TSK \in \mathbb{G}_T$ to generate the ciphetext header Hdr as follows:

$$\begin{aligned} Hdr &= (i, c_{1,1}, c_{1,2}, c_{1,3}, c_{0,1}, c_{0,2}, c_{0,3}) \\ &= (i, (H(i,1)\prod_{j\in S} g_j)^{s_1}, g^{s_1}, K_1 \oplus TSK, \\ &\quad (H(i,0)\prod_{j\in S} g_j)^{s_0}, g^{s_0}, K_0 \oplus TSK). \end{aligned}$$

(Hdr, S) are sent to all users. The messages are symmetrically encrypted using the session key TSK to generate ciphertext body C, which is not discussed in details for brevity. We assume the employed symmetric encryption scheme is secure enough so that we do not need to care about it.

– **Decrypt**(SK_u, S, Hdr, PK). We assume that user u belongs to S. Parsing the SK_u as ($d_{u,1}$, …, $d_{u,L}$) and the ciphertext header Hdr as (i, $c_{1,1}$, $c_{1,2}$, $c_{1,3}$, $c_{0,1}$, $c_{0,2}$, $c_{0,3}$), user u recovers K_b as follows (we use b to denote $\bar{w}_i^{(u)}$ and h to denote $H(i, b)$):

$$\begin{aligned} K_b &= \frac{e(d_{u,i}\prod_{j\in S, j\neq u} g^{\frac{1}{(x+u)(x+j)}}, c_{b,2})}{e(c_{b,1}, g_u)} \\ &= \frac{e(wh^{\frac{1}{x+u}} g^{\frac{1}{(x+u)^2}} \prod_{j\in S, j\neq u} g^{\frac{1}{(x+u)(x+j)}}, g^{s_b})}{e((h\prod_{j\in S} g_j)^{s_b}, g^{\frac{1}{x+u}})} \\ &= \frac{e(w,g)^{s_b} \cdot e(hg^{\frac{1}{(x+u)}} \prod_{j\in S, j\neq u} g^{\frac{1}{(x+j)}}, g^{\frac{s_b}{x+u}})}{e((h\prod_{j\in S} g_j)^{s_b}, g^{\frac{1}{x+u}})} \\ &= \frac{e(w,g)^{s_b} \cdot e(h\prod_{j\in S} g^{\frac{1}{(x+j)}}, g^{\frac{s_b}{x+u}})}{e((h\prod_{j\in S} g_j)^{s_b}, g^{\frac{1}{x+u}})} \\ &= e(w,g)^{s_b}, \end{aligned}$$

and then the temporary session key is recovered as $TSK = K_b \oplus c_{b,3}$. In above equations, each value $g^{1/(x+u)(x+j)}$ can be computed from the public parameter PK for all $j(\neq u) \in S$ as

$$g^{1/(x+u)(x+j)} = g_u^{1/(j-u)} \cdot g_j^{1/(u-j)}.$$

Then the temporary session key TSK is used to decrypt messages from the ciphertext body.

– **Trace**$^{\mathcal{D}}$(TK). Given a perfect pirate decoder $\mathcal{D}$ with complete decryption keys, the trusted party queries decoder $\mathcal{D}$ as a black-box oracle. We say the decoder is perfect if it will always decrypt well-formed ciphertext correctly [7]. We assume that the captured decoder can recover the messages encrypted for receivers set $S \subseteq \{1, \ldots, N\}$. For $i = 1, \ldots, L$, the sender selects a random temporary session key $TSK \in_R \mathbb{G}_T$, another key $K_R \in_R \mathbb{G}_T$ ($K_R \neq TSK$), two random numbers $s_1, s_0 \in_R \mathbb{Z}_p$. The sender sets $K_1 = e(w, g)^{s_1}$ and $K_0 = e(w, g)^{s_0}$, and then generates the ciphetext header Hdr as follows:

$$\begin{aligned} Hdr &= (i, c_{1,1}, c_{1,2}, c_{1,3}, c_{0,1}, c_{0,2}, c_{0,3}) \\ &= (i, (H(i,1)\prod_{j\in S} g_j)^{s_1}, g^{s_1}, K_1 \oplus TSK, \\ &\quad (H(i,0)\prod_{j\in S} g_j)^{s_0}, g^{s_0}, K_0 \oplus K_R). \end{aligned}$$

A message m is symmetrically encrypted using the session key TSK to generate ciphertext body C. (Hdr, S) and C are fed to the decoder as in normal broadcasting. If the pirate decoder outputs m, the trusted party decides that the decoder contains a codeword $\bar{w}^*$ with $\bar{w}_i^* = 1$. Otherwise, $\bar{w}_i^* = 0$. When the tracing on all positions $(1, \ldots, L)$ is completed, the recovered codeword $\bar{w}^* = (\bar{w}_1^* \ \ldots \ \bar{w}_L^*)$ is put to the tracing algorithm of collusion secure code $\mathbf{T}(\bar{w}^*, \text{TK})$ to obtain a set of traitors $T \cap S \subseteq \{1, \ldots, N\}$, with $T = \mathbf{T}(\bar{w}^*, \text{TK})$.

3.2 Security Analysis

Correctness. The correctness is straightforward from the above detailed descriptions.

Semantic Security. Theorem 1 is used to prove the semantic security of the proposed traitor tracing scheme in which users outside the receiver set cannot recover any information of messages encrypted in the ciphertext.

Theorem 1. *Let $\mathbb{G}$ be a bilinear group of prime order p. For any positive integer n, the proposed traitor tracing scheme is (t, ϵ, n)-secure against chosen plaintext attacks under random oracle model assuming the decisional (t, ϵ, n)-mBDH assumption holds in $\mathbb{G}$.*

Proof. Suppose that there exists a t-time adversary $\mathcal{A}$ with advantage ϵ (with integer n) for the proposed scheme. We will construct a simulator $\mathcal{B}$ to solve the decisional n-mBDH problem with the help of $\mathcal{A}$. Suppose $\mathcal{B}$ is given decisional n-mBDH problem as a tuple (g, z, $g^{1/(x+1)}$, $\ldots$, $g^{1/(x+n)}$, $wg^{1/(x+1)^2}$, $\ldots$, $wg^{1/(x+n)^2}$)$\in \mathbb{G}^{2n+2}$ and a $T \in \mathbb{G}_T$, and $\mathcal{B}$ should output 1 if $T = e(w, z)$ and 0 otherwise. $\mathcal{B}$ interacts with $\mathcal{A}$ as follows.

Init. Given 1^λ and the number of users in the system $N = n$, $\mathcal{B}$ selects $\epsilon \in (0, 1)$ and runs collusion secure code generation algorithm $\mathbf{G}(N, \epsilon)$ to generate a pair

(Γ, TK). The set $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ contains N codewords in $\{0,1\}^L$, where L is the codeword length. TK is the tracing key for Γ. $\bar{w}^{(u)}$ is assigned to user u, with $1 \leq u \leq N$. $\mathcal{B}$ forwards Γ to $\mathcal{A}$, and $\mathcal{A}$ outputs the set S^* of users that it intends to attack, where $S^* \subset \{1, \ldots, N\}$. $\mathcal{A}$ also submits two messages (m_1, m_0) to $\mathcal{B}$.

Setup. From the given decisional n-mBDH problem, $\mathcal{B}$ sets $g_1 = g^{1/(x+1)}$, $\ldots$, $g_N = g^{1/(x+N)}$, $e(w,g) = e(wg^{1/(x+1)^2}, g)/e(g^{1/(x+1)}, g^{1/(x+1)})$. Public parameter PK = ($\mathbb{G}$, $\mathbb{G}_T$, p, g, g_1, $\ldots$, g_N, $e(w,g)$, $H(\cdot)$) is forwarded to $\mathcal{A}$, where $H(\cdot)$ is controlled by $\mathcal{B}$ as random oracle.

KeyGen. For $i = 1, \ldots, L$, $\mathcal{B}$ sets $H(i,1) = g^{t_{i,1}}(\prod_{j \in S^*} g_j)^{-1}$ and $H(i,0) = g^{t_{i,0}}(\prod_{j \in S^*} g_j)^{-1}$, with random $t_{i,1}, t_{i,0} \in_R \mathbb{Z}_p$. $\mathcal{B}$ records these values ($t_{i,1}$, $t_{i,0}$, $H(i,1)$, $H(i,0)$, $i = 1, \ldots, L$) for later usage. When $\mathcal{A}$ queries the decryption keys for user $u \notin S^*$, for $i = 1, \ldots, L$ $\mathcal{B}$ sets $b = \bar{w}_i^{(u)}$ and computes

$$d_{u,i} = wg^{\frac{1}{(x+u)^2}} \cdot g^{\frac{t_{i,b}}{x+u}} \cdot (\prod_{j \in S^*} g^{\frac{1}{(x+u)(x+j)}})^{-1}.$$

We notice that $wg^{\frac{1}{(x+u)^2}}$, $g^{\frac{1}{x+u}}$ and $g^{\frac{1}{(x+u)(x+j)}}$ can be obtained from the given decisional n-mBDH problem. We can check that $g^{\frac{t_{i,b}}{x+u}} \cdot (\prod_{j \in S^*} g^{\frac{1}{(x+u)(x+j)}})^{-1} = g^{\frac{t_{i,b}}{x+u}} \cdot (\prod_{j \in S^*} g_j^{\frac{1}{x+u}})^{-1} = H(i,b)^{\frac{1}{x+u}}$. Thus, the decryption keys $\mathrm{SK}_u = (d_{u,1}, \ldots, d_{u,L})$ are of the valid form.

Challenge. $\mathcal{B}$ selects a position $i \in \{1, \ldots, L\}$ and a random temporary session key $TSK \in \mathbb{G}_T$ to generate the ciphetext header Hdr as follows:

1. selects random $s_1, s_0 \in_R \mathbb{Z}_p$;
2. sets $K_1 = T^{s_1}$, $K_0 = T^{s_0}$. Suppose $z = g^r$ for some unkown $r \in \mathbb{Z}_p$. If $T = e(w,z)$, we will have $K_1 = e(w,g)^{r \cdot s_1}$ and $K_0 = e(w,g)^{r \cdot s_0}$;
3. sets $c_{1,1} = z^{s_1 \cdot t_{i,1}}$, $c_{0,1} = z^{s_0 \cdot t_{i,0}}$. As we notice that,

$$\begin{aligned}(H(i,b) \prod_{j \in S^*} g_j)^{r \cdot s_b} &= (g^{t_{i,b}}(\prod_{j \in S^*} g_j)^{-1} \prod_{j \in S^*} g_j)^{r \cdot s_b} \\ &= (g^{t_{i,b}})^{r \cdot s_b} \\ &= z^{s_b \cdot t_{i,b}},\end{aligned}$$

   for any $b \in \{0,1\}$;
4. sets $c_{1,2} = z^{s_1} = g^{r \cdot s_1}$, $c_{0,2} = z^{s_0} = g^{r \cdot s_0}$.

As we can see, if $T = e(w,z)$, the ciphertext header

$$\begin{aligned}Hdr &= (i, c_{1,1}, c_{1,2}, c_{1,3}, c_{0,1}, c_{0,2}, c_{0,3}) \\ &= (i, z^{s_1 \cdot t_{i,1}}, z^{s_1}, K_1 \oplus TSK, \\ &\quad z^{s_0 \cdot t_{i,0}}, z^{s_0}, K_0 \oplus TSK)\end{aligned}$$

is a ciphertext header of the valid form as for the public parameter PK. TSK is used to symmetrically encrypt a random message m_x ($x \in_R \{0,1\}$) to generate the ciphertext body C. (Hdr, S, C) are sent to $\mathcal{A}$.

Guess. $\mathcal{A}$ outputs a guessed $x' \in \{0,1\}$ as the response to the challenge ciphertext. If $x' = x$, $\mathcal{B}$ decides that $T = e(w,z)$ and outputs 1. Else, $\mathcal{B}$ outputs 0.

As we notice, if $T = e(w,z)$, $\mathcal{B}$ generates a valid ciphertext header with temporary session key TSK. Then, $\mathcal{A}$ will answer the right x' with advantage ϵ. If T is uniformly random from $\mathbb{G}_T$, the ciphertext header is also of valid form but the temporary session key is unknown and an unknown message is encrypted, which does not help $\mathcal{A}$ to guess the bit choice. So $\mathcal{B}$ can solve the decisional n-mBDH problem with the same advantage ϵ. This concludes the proof of Theorem 1. □

Collusion Resistance. Theorem 3 is used to prove that the proposed traitor tracing scheme is t-collusion resistant, i.e., t colluding users cannot generate a perfect pirate decoder that cannot be traced back to any of its creators. Before proving Theorem 3, we firstly prove that t colluding users cannot decrypt the ciphertext which is encrypted to a codeword bit outside the feasible set of their codewords, basing on the security of general construction in [24] that has the form of decryption keys similar to our scheme.

Theorem 2. *The general construction described by Park et al. in [24] is semantically secure.*

We need this theorem to prove Lemma 1, which will be used to prove the collusion resistant security of the proposed scheme. Our readers can refer to Theorem 1 of [24] for detailed proof.

Lemma 1. *Assuming Theorem 2, t users (with a same value for a same tracing position) colluding in our proposed scheme, cannot recover the message encrypted for the other value of the same tracing position.*

Proof. (Sketch) For simplicity, we suppose these t colluding users are $\{1,\ldots,t\}$, and they are controlled by the adversary $\mathcal{A}$. Without loss of generality, we suppose each of these users holds a codeword with value 0 in tracing position i, and they attempt to recover the message encrypted for value 1 in tracing position i. If they have any advantage in doing so, we can construct $\mathcal{B}$ to break the semantic security of general construction by Park et al. in [24]. What we need to do is to invoke the semantic security game $Game_{SS}$ of general construction in [24] and set $H(i,0)$ and $H(i,1)$ to distinct elements that are setup for different subsets in $Game_{SS}$. The sketch process is as follows:

Init. $\mathcal{B}$ submits V ($V > k > 1$) users to the challenger $\mathcal{CH}$ in $Game_{SS}$.

Setup. Since the general construction in [24] can support subset of any size, we suppose the V users are split by $\mathcal{CH}$ and put into k subsets. In these k subsets, each subset contains s ($s > t$) users with some users not belonging to

those V users. Without loss of generality, we suppose the first t users in subset j $(1 < j < k)$ are not included in the V users submitted in Init phase. $\mathcal{CH}$ generates public key as

$$PK = (g, h_1, \ldots, h_k, g_1, \ldots, g_s) \in \mathbb{G}^{k+s+1}.$$

$\mathcal{B}$ queries $\mathcal{CH}$ for decryption key of the first user in subset j, and $\mathcal{B}$ can compute the value $e(w, g)$ with the returned decryption key $d_{j,1}$ as $e(w, g) = e(d_{j,1}, g)/e(h_j, g_1)e(g_1, g_1)$. As for our game, the public parameter is ($\mathbb{G}$, $\mathbb{G}_T$, p, g, g_1, ..., g_s, $e(w, g)$, $H(\cdot)$), where $H(\cdot)$ is controlled by $\mathcal{B}$ as random oracle. $\mathcal{B}$ sets $H(i, 0) = h_j$, and $H(i, 1) = h_1$. $\mathcal{B}$ queries $\mathcal{CH}$ for decryption keys of the rest of the first t users in subset j. $\mathcal{CH}$ should return valid keys for them, since they do not belong to the V users submitted in **Init** phase. As we notice that, these keys (for first t users in subset j) are of valid form in our scheme, so they can serve as the decryption keys (for value 0 in tracing position i) of the t colluding users in our game.

Challenge. $\mathcal{CH}$ sends (Hdr^*, K^*) to $\mathcal{B}$ with $Hdr^* = (z^{t_1}, \ldots, z^{t_k}, z) \in \mathbb{G}^{k+1}$ and $K^* = T \in \mathbb{G}_T$. $\mathcal{B}$ selects randomly $r \in_R \mathbb{Z}_p$, $TSK \in_R \mathbb{G}_T$ and forwards $(i, z^{t_1 \cdot r}, z^r, (K^*)^r \oplus TSK)$ to $\mathcal{A}$ as the challenge ciphertext header for value 1 in tracing position i. As we notice that, the ciphertext header is for h_1, which is set as $H(i, 1)$ in our game. If $T = e(w, z)$ in $Game_{SS}$, the ciphertext header is of valid form in our game for value 1 in tracing position i to a set of users including the t users controlled by $\mathcal{A}$.

Guess. $\mathcal{B}$ forwards the guessed bit of $\mathcal{A}$ to $\mathcal{CH}$.

If $\mathcal{A}$ has any advantage in making the right guess, $\mathcal{B}$ has the same advantage in breaking the semantic security game $Game_{SS}$ of general construction in [24]. Assuming Theorem 2, such advantage is negligible. □

Theorem 3. *The proposed traitor tracing scheme is t-collusion secure assuming the scheme is semantically secure and the collusion secure code is t-collusion secure.*

Proof. The t-collusion resistant game of our proposed scheme is played between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$ as described in Section 2.

Let (G, T) be a collusion secure code. Let N be an positive integer and $\epsilon \in (0, 1)$. The challenger runs $G(N, \epsilon)$ to obtain (Γ, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ and $\bar{w}^{(i)}$ is the codeword for user i. The challenger also generates a proposed traitor tracing scheme with public parameter PK. It sends Γ and PK to the adversary. Then, the adversary selects a subset of $\{1, \ldots, N\}$, denoted as C with $|C| \leq t$. The adversary can query the challenger for decryption keys of users in C. The challenger generates the keys as in the proposed scheme and gives them to the adversary.

When it is time for the challenger to query the adversary on decryptions, for the tracing position $i = 1, \ldots, L$, the challenger queries the adversary with message m encrypted as in the tracing algorithm to a set S of receivers. There are four cases for the decoder:

- Case 1: The adversary does not hold any valid decryption key of users in S. As we proved in Theorem 1, the adversary will always output a random message other than m. The probability that the adversary outputs the right message is at most $1/|\mathcal{M}|$, where $|\mathcal{M}|$ is the number of messages in the message space. In this case, the tracer changes to another set to continue. The tracer will be deceived with probability at most $1/|\mathcal{M}|$;
- Case 2: The adversary holds decryption keys for at least one user in S (these users with decryption keys held by the adversary are denoted as S_A, with $S_A = S \cap C$), and all codewords of users in S_A contain "1" in tracing position i. That is to say, all $\bar{w}^{(j)}(\forall j \in S_A)$ satisfy $\bar{w}_i^{(j)} = 1$. Thus, the adversary will always output $m' = m$. The recovered bit $\bar{w}_i^*$ will always be 1. Since all codewords of users in S_A do not contain "0" in position i, the probability that the adversary outputs $m' \neq m$ is less than $Adv_{SS}^{\mathcal{A}}$, the probability that the adversary breaks Lemma 1 (so as to break the semantic security game of the general construction in [24]). As implied in Lemma 1, $Adv_{SS}^{\mathcal{A}}$ is negligible;
- Case 3: The adversary holds decryption keys for at least one user in S (these users are denoted as S_A), and all codewords of users in S_A contain "0" in tracing position i. That is to say, all $\bar{w}^{(j)}(\forall j \in S_A)$ satisfy $\bar{w}_i^{(j)} = 0$. Thus, the adversary will always output $m' \neq m$. The recovered bit $\bar{w}_i^*$ will always be 0. Since all codewords in S_A do not contain "1" in position i, the probability that the adversary outputs m is less than $Adv_{SS}^{\mathcal{A}}$, the probability that the adversary breaks Lemma 1;
- Case 4: The adversary holds decryption keys for at least one user in S (these users are denoted as S_A), and codewords of users in S_A contain both "0" and "1" in tracing position i. No matter what message the adversary outputs (m or others), $\bar{w}_i^*$ must be in the feasible set of all codewords corresponding to S_A.

We use W_{S_A} to denote the set of codewords corresponding to S_A. Therefore, the final recovered codeword $\bar{w}^* \in F(W_{S_A})$. From the assumption that collusion secure code (G, T) is t-collusion resistant, the probability that $T(\bar{w}^*, \mathrm{TK})$ is empty or not a subset of S_A is less than ϵ. Thus, the probability that the adversary breaks the property of t-collusion resistance of our traitor tracing scheme is less than $(1/|\mathcal{M}|)^L + 2L \cdot Adv_{SS}^{\mathcal{A}} + \epsilon$. □

As we notice that, when $t = N$, our proposed scheme is fully collusion resistant.

Public Collaboration Resistance As discussed in Section 1 that most codes based traitor tracing scheme suffered from a kind of attack called public collaboration (Pirate 2.0) [4]. Now we take a look at the proposed scheme.

Theorem 4. *The proposed scheme is public collaboration resistant.*

Proof. If some user in the proposed scheme releases certain partial key in public, it may release together its index u in the system, the key index of the released partial key in its set of decryption keys and the corresponding codeword bit so as that the partial key can be used immediately. If so, the traitor's identity (its

Table 1. Comparison with Previous Codes Based Works

	Public Key	Private Key	Ciphertext Length	Encryption Computation	Decryption Computation	Revocation	Pirate 2.0 Resistant
[19]	$2L\mathrm{G}$	$L\mathrm{Z}_p$	$2L\mathrm{G}$	L(2E+ 1M)	L(1E+ 2M)	No	No
[10]	1G+ $(L+1)\mathrm{G}_T$	$L\mathrm{Z}_p$+ $L\mathrm{G}$	2G+ $L\mathrm{G}_T$	$L\mathrm{E}_T$+ 2M	L(M+ 2P)	No	No
[7]	$2L\mathrm{G}$	$L\mathrm{Z}_p$	$1\mathrm{Z}_p$+ 2G	(2E+ 1M)	(1E+ 2M)	No	No
[3]	$2L\mathrm{G}$	$L\mathrm{Z}_p$	u(1Z_p+ 2G)	u(2E+ 1M)	u(1E+ 2M)	No	No
[11]	$(2L+1)\mathrm{G}$	$(L+2)\mathrm{Z}_p$	$2\mathrm{Z}_p$+ $(L+2)\mathrm{G}$	(3E+1M)	(2E+2M)	No	No
[29]	$(L+2)\mathrm{G}_1$ +2G_2	1G_1 +1G_2	$2L\mathrm{G}_1$ +2G_2+3Z_p	2(L+1)E_1 +2M +2E_2 +2E_T	(L-1)E_1 +(L-1)M +2P+1M_T	No	Yes
Ours	$(N+1)\mathrm{G}$ +1Z_p	$L\mathrm{G}$	1Z_p +4G +2G_T	2($\|S\|$+1)M +4E+2E_T	2($\|S\|$-1)E+$\|S\|$M +2P+1M_T	Yes	Yes

L: the length of codeword; $|S|$: the number of users in the set S of receivers;
N: the total number of users in the system;
u: the number of codeword positions used in encryption [3];
G: element in $\mathbb{G}$; G_1: element in $\mathbb{G}_1$; G_2: element in $\mathbb{G}_2$; G_T: element in $\mathbb{G}_T$;
Z_p: element in $\mathbb{Z}_p$; P: pairing in $\mathbb{G} \times \mathbb{G}$ or $\mathbb{G}_1 \times \mathbb{G}_2$;
E: exponentiation in $\mathbb{G}$; E_1: exponentiation in $\mathbb{G}_1$; E_2: exponentiation in $\mathbb{G}_2$;
M: multiplication (or division) in $\mathbb{G}$; M_1: multiplication (or division) in $\mathbb{G}_1$;
M_T: multiplication (or division) in $\mathbb{G}_T$.

Table 2. Comparison with Previous Public Key Trace and Revoke Schemes

	Public Key	Private Key	Ciphertext Length	Number of Revoked Users	Encryption Computation	Decryption Computation	Collusion Threshold
[14]	$O(R)$	$O(1)$	$O(R)$	$\leq R$	$O(R)$	$O(R)$	R
[9]	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$	Unlimited	$O(\|S\|+\sqrt{N})$	$O(\|S\|)$	No Threshold
[27]	$O(\log N)$	$O(1)$	$O(r \log(N/r) \cdot \log N)$	Unlimited	$O(r \log(N/r) \cdot \log N)$	$O(\log N)$	No Threshold
Ours	$O(N)$	$O(L)$	$O(1)$	Unlimited	$O(\|S\|)$	$O(\|S\|)$	Depends on codes

L: the length of codeword; $|S|$: the number of users in the set S of receivers;
N: the total number of users in the system; r: the number of revoked users;
R: the revocation threshold in each broadcasting.

index u in the system) is exposed already. If not, the tracer can still find out the traitor's identity by $2LN$ rounds of computations. For all $j = 1, \ldots, L$, $b = 1, 0$, and $u = 1, \ldots, N$, the tracer tests whether the following equation holds:

$$e(d^*, g) = e(w, g)e(H(j, b), g_u)e(g_u, g_u),$$

where d^* denotes the released partial key. If the equation holds for certain (j, b, u), user with system index u will be accused for releasing this partial key.

Thus, the partial decryption key in our scheme is self-enforced so that our scheme is resistant to public collaboration. □

3.3 Efficiency Analysis

In Table 1, we compare our scheme with previous codes based traitor tracing schemes [19,10,7,3,11,29] on public key size, private key size, ciphertext length, encryption computation, decryption computation, revocation ability and the property of public collaboration resistance. We assume that all schemes use collusion secure codes of a same length L. Since schemes in [7] and [3] did not mention the public key encryption scheme used for each position, we suppose both schemes use secure ElGamal encryption scheme over a cyclic group of order p, where p is a large strong prime.

We show that our scheme achieves constant ciphertext length as schemes in [7,3] did. Our scheme is resistant to public collaboration according to the discussion in previous subsection.

Our scheme is also compared with previous public key trace and revoke schemes, including threshold secret sharing based scheme [14], augmented broadcast encryption based scheme [9] and subset cover based scheme [27]. As we can see in Table 2, our scheme achieves constant ciphertext while others do not.

4 Conclusion and Discussions

We present a codes based tracing and revoking scheme by combining Park et al.'s public key broadcast encryption scheme [24] and Boneh and Naor's traitor tracing scheme [7], which presents an answer to the problem left open by Billet and Phan [3].

Our scheme is proved secure under random oracle model, and the random oracle can be removed by adding more elements in public parameters (discussed in Appendix A.1).

Our scheme can also be extended to scheme based on IPP codes and scheme against imperfect pirate decoders, and we discuss them in Appendix A.2 and Appendix A.3.

We notice that our scheme is based on v-Modified Bilinear Diffie-Hellman assumption [24] that is not well studied, which urges us to present codes based tracing and revoking scheme based on standard complexity assumptions in future work.

Acknowledgments. This work is supported by the Fundamental Research Funds for the Central Universities (No. K50511010001).

References

1. Abdalla, M., Catalano, D., Dent, A.W., Malone-Lee, J., Neven, G., Smart, N.P.: Identity-Based Encryption Gone Wild. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006, Part II. LNCS, vol. 4052, pp. 300–311. Springer, Heidelberg (2006)

2. Abdalla, M., Dent, A.W., Malone-Lee, J., Neven, G., Phan, D.H., Smart, N.P.: Identity-Based Traitor Tracing. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 361–376. Springer, Heidelberg (2007)
3. Billet, O., Phan, D.H.: Efficient Traitor Tracing from Collusion Secure Codes. In: Safavi-Naini, R. (ed.) ICITS 2008. LNCS, vol. 5155, pp. 171–182. Springer, Heidelberg (2008)
4. Billet, O., Phan, D.H.: Traitors Collaborating in Public: Pirates 2.0. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 189–205. Springer, Heidelberg (2009)
5. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
6. Boneh, D., Franklin, M.K.: An Efficient Public Key Traitor Scheme (Extended Abstract). In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 338–353. Springer, Heidelberg (1999)
7. Boneh, D., Naor, M.: Traitor tracing with constant size ciphertext. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM Conference on Computer and Communications Security, pp. 501–510. ACM (2008)
8. Boneh, D., Sahai, A., Waters, B.: Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 573–592. Springer, Heidelberg (2006)
9. Boneh, D., Waters, B.: A fully collusion resistant broadcast, trace, and revoke system. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S.: (eds.) ACM Conference on Computer and Communications Security, pp. 211–220. ACM (2006)
10. Chabanne, H., Phan, D.H., Pointcheval, D.: Public Traceability in Traitor Tracing Schemes. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 542–558. Springer, Heidelberg (2005)
11. Chen, Y.-R., Tzeng, W.-G.: A Public-Key Traitor Tracing Scheme with an Optimal Transmission Rate. In: Qing, S., Mitchell, C.J., Wang, G. (eds.) ICICS 2009. LNCS, vol. 5927, pp. 121–134. Springer, Heidelberg (2009)
12. Chor, B., Fiat, A., Naor, M.: Tracing Traitors. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 257–270. Springer, Heidelberg (1994)
13. D'Arco, P., Perez del Pozo, A.L.: Fighting Pirates 2.0. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 359–376. Springer, Heidelberg (2011)
14. Dodis, Y., Fazio, N.: Public Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 100–115. Springer, Heidelberg (2002)
15. Fernandez, M., Soriano, M.: Decoding codes with the identifiable parent property. In: ISCC, pp. 1028–1033. IEEE Computer Society (2002)
16. Fiat, A., Tassa, T.: Dynamic Traitor Tracing. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 354–371. Springer, Heidelberg (1999)
17. Garg, S., Kumarasubramanian, A., Sahai, A., Waters, B.: Building efficient fully collusion-resilient traitor tracing and revocation schemes. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM Conference on Computer and Communications Security, pp. 121–130. ACM (2010)
18. Hollmann, H.D.L., van Lint, J.H., Linnartz, J.P.M.G., Tolhuizen, L.M.G.M.: On codes with the identifiable parent property. J. Comb. Theory, Ser. A 82(2), 121–133 (1998)

19. Kiayias, A., Yung, M.: Traitor Tracing with Constant Transmission Rate. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 450–465. Springer, Heidelberg (2002)
20. Kurosawa, K., Desmedt, Y.: Optimum Traitor Tracing and Asymmetric Schemes. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 145–157. Springer, Heidelberg (1998)
21. Mitsunari, S., Sakai, R., Kasahara, M.: A new traitor tracing. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E85-A(2), 481–484 (2002)
22. Naor, D., Naor, M., Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)
23. Naor, M., Pinkas, B.: Efficient Trace and Revoke Schemes. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 1–20. Springer, Heidelberg (2001)
24. Park, J.H., Kim, H.J., Sung, M., Lee, D.H.: Public key broadcast encryption schemes with shorter transmissions. IEEE Transactions on Broadcasting 54(3), 401–411 (2008)
25. Park, J.H., Lee, D.H.: Fully collusion-resistant traitor tracing scheme with shorter ciphertexts. Des. Codes Cryptography 60(3), 255–276 (2011)
26. Park, J.H., Rhee, H.S., Lee, D.H.: Fully collusion-resistant trace-and-revoke scheme in prime-order groups. Journal of Communications and Networks 13(5), 428–441 (2011)
27. Phan, D.H., Trinh, V.C.: Identity-Based Trace and Revoke Schemes. In: Boyen, X., Chen, X. (eds.) ProvSec 2011. LNCS, vol. 6980, pp. 204–221. Springer, Heidelberg (2011)
28. Stinson, D.R., Wei, R.: Combinatorial properties and constructions of traceability schemes and frameproof codes. SIAM J. Discrete Math. 11(1), 41–53 (1998)
29. Zhao, X., Zhang, F.: Traitor Tracing against Public Collaboration. In: Bao, F., Weng, J. (eds.) ISPEC 2011. LNCS, vol. 6672, pp. 302–316. Springer, Heidelberg (2011)

A Extension Discussions

A.1 Removing Random Oracle

The proposed scheme is proved secure against chosen plaintext attacks in random oracle model. Here, we show a method to remove the random oracle. What we need to do is to remove the hash function $H(\cdot)$ and add $2L$ random elements in $\mathbb{G}$ into public parameter as the values of hash function. The resulting public parameter is of the length $O(L + N)$.

A.2 Scheme Based on IPP Codes

As for a q-ary IPP code [15], let $\Theta = \{Sym_1, \ldots, Sym_q\}$ be the set containing these q symbols. Then each user will receive a codeword from Θ^L and the corresponding set of decryption keys generated as described in the proposed scheme. Each set of decryption keys is of the length $O(qL)$.

When in broadcasting, a tracing position i and a set S of receivers are selected. The ciphertext header (encrypting a same temporary session key) will enable all q symbols in position i instead of two symbols when using collusion secure codes. The ciphertext is of the length $O(q)$.

When in tracing, however, the ciphertext header will encrypt different temporary session keys for different symbols in position i, i.e., q distinct session keys $(TSK_1, \ldots, TSK_q)$ are encrypted and TSK_j is for Sym_j, $j = 1, \ldots, q$. As for the ciphertext body, a simple method is that q distinct messages $(m_1, \ldots, m_q)$ are encrypted and m_j is for Sym_j, $j = 1, \ldots, q$. If the pirate decoder outputs m_j, the tracer decides that the decoder contains a codeword $\bar{w}^*$ with $\bar{w}^*_j = Sym_j$. There are other methods to recover the codeword, and we will not discuss them further for brevity. When the tracing on all positions is completed, the recovered codeword $\bar{w}^* = (\bar{w}^*_1 \ \ldots \ \bar{w}^*_L)$ will be input to the decoding (tracing) algorithm of IPP code, and a list of parent codewords are obtained.

A.3 Scheme against Imperfect Pirate Decoders

As for imperfect pirate decoders [7] that will fail to decrypt well-formed ciphertext with a probability δ $(0 \leq \delta < 1)$, Boneh and Naor described a sophisticated tracing method with the help of δ-robust fingerprinting codes [7]. Their method can also be modified and applied in our scheme to enable such feature.

We assume that $\mathcal{M}$ is the message space. i is the tracing position. S is the set of receivers for which the captured decoder can decrypt the ciphertext. $TSK1$ is the session key for value 1 in position i and $TSK0$ is for value 0. m is the message to be encrypted. Hdr is the output header and C is the generated ciphertext body using $TSK1$. We use $(Hdr, C) \leftarrow TraceEnc(i, S, TSK1, TSK0, m)$ to denote the computations in tracing algorithm, and use $m' \leftarrow \mathcal{D}(S, Hdr, C)$ to denote that decoder $\mathcal{D}$ outputs m' on input (S, Hdr, C). For $i = 1, \ldots, L$, the tracing algorithm is defined as follows:

The tracer repeats the following steps $\lambda \ln L$ times:

$$\begin{aligned} m &\xleftarrow{R} \mathcal{M}; \\ TSK1 &\xleftarrow{R} \mathbb{G}_T; \\ TSK0 &= TSK1; \\ (Hdr, C) &\leftarrow TraceEnc(i, S, TSK1, TSK0, m); \\ m' &\leftarrow \mathcal{D}(S, Hdr, C). \end{aligned}$$

Let p_i be the fraction of times that $m' = m$;

The tracer repeats the following steps $\lambda \ln L$ times:

$$\begin{aligned} m &\xleftarrow{R} \mathcal{M}; \\ TSK1 &\xleftarrow{R} \mathbb{G}_T; \\ TSK0 &\xleftarrow{R} \mathbb{G}_T, \ with \ (TSK0 \neq TSK1); \\ (Hdr, C) &\leftarrow TraceEnc(i, S, TSK1, TSK0, m); \\ m' &\leftarrow \mathcal{D}(S, Hdr, C). \end{aligned}$$

Let q_i be the fraction of times that $m' = m$. Define $\bar{w}_i^* \in \{0, 1\}$ as:

$$\bar{w}_i^* = \begin{cases} 1 \ \text{if } q_i > 0 \\ 0 \ \text{if } q_i = 0 \text{ and } p_i > 0 \\ \text{'?' otherwise} \end{cases}$$

and $\bar{w}^* = \bar{w}_1^* \ldots \bar{w}_L^*$. By using the δ-robust collusion secure code presented in [7] we obtain a tracing scheme that can trace imperfect pirate decoders as long as

$$\delta < (1/L) - (1/\lambda).$$

For details about tracing imperfect pirate decoders and constructing δ-robust collusion secure codes, please refer to [7].

A.4 Subset Cover Based Traitor Tracing Method

Beside the codes based tracing method by Boneh and Naor, we have another choice for our scheme, i.e., to use the subset cover based traitor tracing method [22]. At first, the tracer specifies a set of receivers for which the pirate decoder can decrypt the ciphertext. Then, the tracer randomly splits the set into two subsets (roughly half the receivers) and chooses the subset for which the pirate decoder can decrypt the ciphertext to continue. The process of tracing and splitting is performed repeatedly until only one user remained and the decoder still can decrypt. The remaining user is the traitor. As stated in [22], the subset cover based traitor tracing method requires that the pirate decoder is available and resettable.

Author Index