2015

Contributions to identity-based cryptography

Kefeng Wang
University of Wollongong

Contributions to Identity-Based Cryptography

A thesis submitted in fulfillment of the
requirements for the award of the degree

Doctor of Philosophy

from

UNIVERSITY OF WOLLONGONG

by

Kefeng Wang

School of Computer Science and Software Engineering
January 2015

Dedicated to
My family and friends

Declaration

This is to certify that the work reported in this thesis was done by the author, unless specified otherwise, and that no part of it has been submitted in a thesis to any other university or similar institution.

Kefeng Wang
January 28, 2015

Abstract

In traditional public key cryptography, public keys of users are essentially random strings generated from random secret keys. Hence, public key certificates are required to attest to the relations between users' identities and their public keys. In the identity-based cryptography, public keys can be identities such as names, email addresses or IP addresses. This avoids the use of certificates which is a burden in traditional public key cryptography. Attribute-based cryptography originated from the identity-based cryptography goes one step further to support fine-grain access control. In the attribute-based cryptography, a user is defined by a set of attributes rather than atomically by a single string. In this thesis, we investigate several cryptographic primitives in the identity-based setting and its successor, attribute-based setting.

There are two classes of digital signature schemes: signature schemes that require the original message as input to the verification algorithm and signature schemes with message recovery which do not require the original message as input to the verification algorithm. One of effective methods for saving bandwidth in transmission is to eliminate the requirement of transmitting the original message for the signature verification. In a signature with message recovery, all or part of the original message is embedded within the signature and can be recovered. Therefore, it minimizes the total length of the original message and the appended signature. In this thesis, we consider digital signatures with message recovery in both the identity-based multisignature setting and the attribute-based setting. In the identity-based multisignature with message recovery, multiple signers generate a single constant size multisignature on the same message regardless of the number of signers. The size of the multisignature is the same as that of a signature generated by one signer. Furthermore, it does not require the transmission of the original message in order to verify the multisignature. In the attribute-based signature with message recovery, the signature size is the same as that of a traditional attribute-based signature,

but it does not require the transmission of the original message for the signature verification.

When messages are transmitted in such a way that both privacy and authenticity are needed, authenticated encryption or signcryption could be used. Usually, there may be some additional information, such as a header, transmitted along with the ciphertext. The header might be public, but have to be authenticated. Authenticated encryption with associated data can be achieved only in the symmetric key setting. In this thesis, we propose a generic construction of identity-based authenticated encryption with authenticated header. Using this cryptographic primitive, everyone is able to check the validity of the authenticated ciphertext and access to the authenticated header; only the designated receiver can recover the payload. Our scheme has potential applicability to big data.

We consider two types of computation on authenticated data. One requires secret information of the original signer, such as the sanitizable signature and the incremental signature. The other one does not require any secret information of the original signer, which means that anyone is able to conduct the computation, but only for a class of specified predicates. In this thesis, we propose two novel schemes, one for each type. The first one is the identity-based quotable ring signature scheme. We extend the ring signature scheme to be quotable. Anyone can derive new ring signatures on substrings of an original message from an original ring signature on the original message. No matter whether a ring signature is originally generated or is quoted from another ring signature, it will convince the verifier that it is generated by one of the ring members. The verifier could not distinguish whether a ring signature is originally generated or is quoted from another ring signature. The second one is the attribute-based proxy re-signature scheme. Only the designated proxy who possesses some secret information of the delegator can re-sign original signatures. The semi-trusted proxy acts as a translator to convert a signature satisfying one predicate into a signature satisfying another different predicate on the same message. The proxy cannot learn any signing key and cannot sign arbitrary messages on behalf of either the delegator or the delegatee. It solves the open problem of finding a multi-use unidirectional proxy re-signature scheme where the size of the signatures and the verification cost do not grow linearly with the number of translations.

Acknowledgement

I would like to express my sincere gratitude to my supervisors Professor Yi Mu, and Professor Willy Susilo, for their careful guidance and patience. They have been providing invaluable encouragement and suggestions to guide me through this research process from the beginning of my research career. I also would like to express my sincere gratitude to Dr. Fuchun Guo, for his invaluable discussions whenever I encounter difficult problems in my research. This thesis would have been impossible without their support.

I wish to take this opportunity to give my thanks to Professor Zhihong Zhang, my supervisor when I was doing a master degree at the Zhengzhou University in China, for his advice and support since 2008. I wish to thank all my colleagues, a non-exhaustive list of whom includes: Minjie Zhang, Yong Yu, Guomin Yang, Man Ho Au, Xixiang Lv, Baocang Wang, Jinguang Han, Yang Wang, Hui Cui, Nan Li, Shams Qazi, Ibrahim Elashry, Jiangshan Yu, Yafu Ji, Xinyu Fan, Weiwei Liu, Yangguang Tian, Rongmao Chen, Shiwei Zhang, for their help and support. I am grateful to my friends, Lei Lv, and Yan Kong, who happened to be my schoolmates from master degree in Zhengzhou to doctor degree here, and roommates, Chao Yu, and Liang Luo, for being with me during my stay in Australia.

It is an honour for me to be a research student of Centre for Computer and Information Security Research, School of Compute Science and Software Engineering, University of Wollongong. I also would like to thank the China Scholarship Council and the University of Wollongong for providing the financial support during my study.

Last but not least, I am sincerely grateful to my family, for everything.

Publications

This thesis is related to the following publications/manuscripts.

1. Kefeng Wang, Yi Mu, and Willy Susilo. Identity-Based Multisignature with Message Recovery. In Robert H. Deng and Tao Feng, editors, *Information Security Practice and Experience*, volume 7863 of *Lecture Notes in Computer Science*, pages 91-104. Springer Berlin Heidelberg, 2013.

2. Kefeng Wang, Yi Mu, Willy Susilo, and Fuchun Guo. Attribute-Based Signature with Message Recovery. In Xinyi Huang and Jianying Zhou, editors, *Information Security Practice and Experience*, volume 8434 of *Lecture Notes in Computer Science*, pages 433-447. Springer Berlin Heidelberg, 2014.

3. Kefeng Wang, Yi Mu, and Willy Susilo. Identity-Based Authenticated Encryption with Authenticated Header. (Submitted)

4. Kefeng Wang, Yi Mu, and Willy Susilo. Identity-Based Quotable Ring Signature. (Submitted)

5. Kefeng Wang, Yi Mu, Willy Susilo, and Fuchun Guo. Attribute-Based Proxy Re-Signature. (Submitted)

List of Notations

The following notations are used throughout this thesis.

λ	A security parameter;
1^λ	The string of λ ones;
$\epsilon(\lambda)$	A negligible function on λ;
$\forall$	For all;
$\exists$	There exists;
$\mathbb{G}$	A group;
$\mathbb{Z}$	The set of integers;
$\mathbb{Z}_p$	The set consists of the integers modulo p;
$\mathbb{Z}_n^*$	The multiple group of integers modulo n;
$\mathbb{A}$	The universal attribute set;
$a\|\|b$	The concatenation of the string a and the string b;
$\Pr[\mathbf{A}]$	The probability of the event $\mathbf{A}$ occurring;
$\mathbf{A} \bigcup \mathbf{B}$	The union of sets $\mathbf{A}$ and $\mathbf{B}$;
$\mathbf{A} \bigcap \mathbf{B}$	The intersection of sets $\mathbf{A}$ and $\mathbf{B}$;
$a \xleftarrow{R} \mathbf{A}$	a is selected from $\mathbf{A}$ uniformly at random if $\mathbf{A}$ is a finite set;
$a \in \mathbf{A}$	a is in the set $\mathbf{A}$.
$a \notin \mathbf{A}$	a is not in the set $\mathbf{A}$.

List of Abbreviations

The following abbreviations are used throughout this thesis.

ABC	Attribute-Based Cryptography;
ABS	Attribute-Based Signature;
AEAH	Authenticated Encryption with Authenticated Header;
BDH	Computational Bilinear Diffie-Hellman;
CDH	Computational Diffie-Hellman;
DBDH	Decisional Bilinear Diffie-Hellman;
DDH	Decisional Diffie-Hellman;
IBC	Identity-Based Cryptography;
IBE	Identity-Based Encryption;
IBS	Identity-Based Signature;
IBSMR	Identity-Based Signature with Message Recovery;
NIZK	Non-Interactive Zero Knowledge;
PKC	Public Key Cryptography;
PKE	Public Key Encryption;
PKG	Private Key Generator;
PPT	Probabilistic Polynomial Time;
PRS	Proxy Re-Signature;
TTP	Trusted Third Party;

Contents

Chapter 1

Introduction

In traditional public key cryptography (PKC), public keys of users are essentially random strings generated from random secret keys. Hence, certificates are required to attest to the relations between users' identities and their public keys. Furthermore, the sender in the encrypting scenario, and the verifier in the signing scenario, have to retrieve the public key from certificate before he/she communicates with others. Identity-based cryptography (IBC) was proposed to alleviate the burden of certificates. In IBC, public keys can be identities such as names, email addresses or IP addresses. However, the receiver in the encrypting scenario, and the signer in the signing scenario, have to obtain the private key from a trusted third party (TTP). Attribute-based cryptography (ABC) originated from identity-based cryptography goes one step further to support fine-grain access control. Compared with IBC in which a single string represents a user's identity, in ABC a user is defined by a set of attributes. In this thesis, we investigate several cryptographic primitives in the identity-based setting and its successor, attribute-based setting.

1.1 Background

1.1.1 Digital Signature with Message Recovery

There are two general classes of digital signature schemes: signature schemes that require the original message as input to the verification algorithm and signature schemes with message recovery which do not require the original message as input to the verification algorithm.

Signature scheme with message recovery means that the original message to be verified is embedded within the signature. The original message will be recovered from the signature during the verification. A digital signature with message recovery

scheme is a special kind of digital signature scheme. Compared with other digital signature schemes, it has one advantage, namely it often adds less overhead to the length of a signed message. In order to resist existential forgery, a suitable redundancy function is usually required. Digital signature with message recovery scheme is usually deployed when the data to be signed is small. It is not suitable for signing long messages such as documents.

In networks with limited bandwidth, long digital signatures will obviously be a drawback. One of effective methods for saving bandwidth in transmission is to eliminate the requirement of transmitting the original message for the signature verification. In a signature with message recovery, all or part of the original message is embedded within the signature and can be recovered. Therefore, it minimizes the total length of the original message and the appended signature. In this thesis, we consider digital signature with message recovery schemes both in the identity-based multisignature setting and in the attribute-based setting.

To convince a verifier that each member of a stated subgroup signed a message and the size of the subgroup can be arbitrary. A trivial approach is to sign and send their individual signatures separately. But it requires more bandwidth. In addition, the transmission and the verification of these signatures are very intricate.

In the identity-based multisignature with message recovery scheme, multiple signers generate a single constant size multisignature on the same message regardless of the number of signers. The multisignature convinces a verifier that each member of the stated group of signers endorsed the message. The size of the multisignature is the same as that of a signature generated by one signer. Thus, a single multisignature can greatly save communication costs instead of transmitting several individual signatures. In addition, the message recovery property eliminates the requirement of transmitting the original message. It shortens the total length of the original message and the appended multisignature even more.

In the attribute-based signature (ABS) with message recovery scheme, signatures reveal no more than the fact that a single signer with some set of attributes satisfying the predicate has attested to the message. Signatures hide the attributes used to satisfy the predicate and users cannot collude to pool their attributes together. The signature size is the same as that of a traditional ABS, but the new scheme does not require the transmission of the original message for the signature verification. We also extend this scheme to deal with large messages.

1.1.2 Authenticated Encryption with Authenticated Header

Consider an interesting scenario, where a party has to handle many big files. Usually, these files should be signed and encrypted. In the scenario of a gateway, it is extremely infeasible for the gateway to check the validity and content of all the received files in particular, since these files are large. Suppose a file is divided into a header and the payload. Both parts are authenticatable. The authenticated header is accessible by everyone while the payload itself is encrypted for confidentiality. Therefore, the gateway only needs to check the validity of the authenticated ciphertext and content of the authenticated header in order to decide how to handle the file.

In this case, both privacy and authenticity are desirable during the transmission of messages. Authenticated encryption is a cryptographic primitive that simultaneously provides confidentiality, integrity, and authenticity on messages. It is difficult to derive a secure authenticated encryption scheme by just gluing a traditional encryption scheme and a digital signature scheme. This inspires the distinct design of authenticated encryption scheme.

There may be additional information that travels alongside the authenticated ciphertext. This information may not be considered about the confidentiality, but must be authenticated. The authenticated encryption with associated data problem has been solved and formalized by Rogaway [Rog02], in the symmetric key setting. In this thesis, we propose a generic construction of identity-based authenticated encryption with authenticated header (AEAH) scheme, in the asymmetric key setting, by using a normal identity-based signature with message recovery (IBSMR) scheme and a symmetric encryption scheme. In a concrete instance of the generic construction by using an identity-based partial message recovery scheme [ZSM05], we also address a problem which was not considered in the work of Rogaway [Rog02], namely how the associated data is made known to the receiver. Although the technique of message recovery is not suitable for signing long messages, often a few bytes are all that one needed, such as an IP address. Using this cryptographic primitive, everyone can check the validity of the authenticated ciphertext and access to the authenticated header; only the designated receiver can recover the payload. Our scheme has potential applicability to big data.

1.1.3 Computing on Authenticated Data

Normally, it is desirable for digital signature schemes not allowing existential forgery, which means anyone other than the actual signer cannot generate any new signature on behalf of the actual signer. However, in some special cases, such as when the actual signer is not available or when the signing key has been expired, it is desirable to allow some designated users or anyone to derive new signatures on behalf of the actual signer from existing signatures, only when some conditions are satisfied. The design of signature schemes that allow "forgeries" of pre-determined types was introduced by Rivest [Riv00]. Some of them require secret information of the actual signer, others do not require secret information of the actual signer, which means anyone is able to conduct these computations. We investigate both of them.

- **Quotable Signature.** Quotable signature schemes belong to the type of computing on authenticated data which does not require secret information of the actual signer. Quoting is usually applied to derive signatures on substrings when text messages are signed. It can also be applied to derive signatures on subregions of an image when images are signed, such as a face. The verifier could not distinguish whether a signature is originally generated or is quoted from another signature. It is desirable to allow repeated computation on the signatures, which means it is possible to quote from a quoted signature. It is also desirable that the signature size will not grow with every derivation. Ahn et al. [ABC+12] proposed an efficient quotable signature scheme. In this thesis, we extend the ring signature scheme to be quotable by using the technique of Shacham and Waters [SW07]. In our identity-based quotable ring signature scheme, no matter whether a ring signature is originally generated or is quoted from another ring signature, it will convince the verifier that it is generated by one of the ring members, without revealing any information about which ring member is the actual signer. Furthermore, the verifier could not distinguish whether a ring signature is originally generated or is quoted from another ring signature.

- **Proxy Re-Signature.** In 2005, Ateniese and Hohenberger [AH05] proposed an interesting application, in which proxy re-signature schemes can be used as a space-efficient proof that a specific path was taken in a graph without taking any shortcuts. It is suitable for either E-passport to show that a visitor followed a prescribed path, or network to show that a packet followed a prescribed path.

The basic idea is that only the first node is given a signing key, all other nodes in the path are given a re-signature key which only allows it to translate signatures from adjacent nodes, but not to generate signatures.

Proxy Re-Signature (PRS) scheme belongs to the type of computing on authenticated data which requires secret information of the actual signer. It aims at secure delegation of signing without fully trusting the proxy. In a PRS scheme, a semi-trusted proxy who possesses some secret information from a delegator acts as a translator between a delegatee and the delegator. The proxy is able to convert a signature from the delegatee into a signature from the delegator on the same message, while the proxy cannot learn any signing key and cannot sign arbitrary messages on behalf of either the delegatee or the delegator. Obviously, generating proxy re-signing key requires the delegator's secret. Otherwise, the underlying system is not secure.

The notion of PRS was introduced by Blaze, Bleumer, and Strauss [BBS98]. The construction of [BBS98] is bidirectional, which means the proxy is able to translate signatures in either direction, and multi-use, which means the translation of signatures can be performed in sequence and multiple times by distinct proxies. Ateniese and Hohenberger [AH05] revisited the notion of PRS by providing appropriate security definitions and efficient constructions. One of their schemes is a unidirectional but single-use PRS scheme. Libert and Vergnaud [LV08] presented the first constructions of multi-use unidirectional PRS scheme. The size of signatures in [LV08] grows linearly with the number of past translations. In this thesis, we introduce the notion of attribute-based PRS, which is the first multi-use unidirectional PRS scheme where the size of signatures and the verification cost do not grow linearly with the number of translations. In the attribute-based setting, the proxy converts a signature satisfying one predicate into a signature satisfying another different predicate on the same message.

1.2 Contributions of This Thesis

In this thesis, we mainly focus on IBC and its successor ABC. The main contributions of this thesis are as follows.

1. **Digital signature with message recovery.** In the literature, constructions

of identity-based multisignature [GGDS06, HR08] and ABS [YCD08, SY08, Kha08, MPR11, LK08, SSN09, LAS+10] have been proposed. However, most of them require the original message as input to the verification algorithm to verify the signatures, and some of them are not formally proved. In this thesis, we propose an identity-based multisignature with message recovery scheme, and two ABS with message recovery schemes. All of them are formally defined and proved. The identity-based multisignature with message recovery scheme is proven to be existentially unforgeable against adaptively chosen message attacks under the Computational Diffie-Hellman (CDH) assumption in the random oracle model. Since there is no need to transmit the original message to the verifier, this scheme minimizes the total length of the original message and the appended multisignature. Two ABS with message recovery schemes are proposed. They allow fine-grain access control and support flexible threshold predicate. The first scheme embeds short original message in the signature and the short message will be recovered in the process of verification, while keeping the signature size the same as that of a traditional scheme [LAS+10] which requires transmission of the original message to verify the signature. The second scheme is extended from the first one to deal with large messages. Both of them are proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the CDH assumption. These two ABS with message recovery schemes are also equipped with attribute-privacy property.

2. **Identity-based authenticated encryption with authenticated header.** In the literature, authenticated encryption with associated data can be achieved in the symmetric key setting [Rog02]. In this thesis, we propose a generic construction of identity-based authenticated encryption with authenticated header (AEAH) scheme in the asymmetric key setting. The construction is proven to be existentially unforgeable against adaptively chosen message attacks and indistinguishable under adaptive chosen ciphertext attacks, if the underlying IBSMR scheme and the symmetric encryption scheme are secure in the same manner, respectively. We also give a concrete instance of the generic construction by using an identity-based partial message recovery signature scheme [ZSM05]. The instance is proven to be existentially unforgeable against adaptively chosen message attacks and indistinguishable under adaptive chosen

ciphertext attacks under the CDH assumption and BDH assumption in the random oracle model, respectively. The instance makes use of a technique of authenticated encryption with message recovery. It also addresses a problem which was not considered in the symmetric key setting resolution [Rog02], namely how the associated data is made known to the receiver.

3. **Identity-based quotable ring signature.** In the literature, efficient ring signature schemes [RST01, RST06, AOS02, SW07, ZK02] have been proposed. In this thesis, we introduce a new notion of identity-based quotable ring signature based on bilinear pairing in composite order groups. The new construction extends the ring signature scheme to be quotable. The proposed scheme is proven to be anonymous under the assumption that the subgroup decision problem is hard, and selectively unforgeable against adaptively chosen message attacks under the assumption that the CDH problem is hard, and strongly context hiding.

4. **Attribute-based proxy re-signature.** In the literature, several proxy re-signature (PRS) schemes [BBS98, AH05, LV08] have been proposed. The first construction of PRS scheme [BBS98] is bidirectional, which means the proxy is able to translate signatures in either direction, and multi-use, which means the translation of signatures can be performed in sequence and multiple times. Subsequently, unidirectional and single-use PRS scheme [AH05], and unidirectional and multi-use PRS scheme [LV08] have been proposed. The size of signatures in the proposed unidirectional multi-use PRS scheme [LV08] grows linearly with the number of translations. The problem of designing multi-use unidirectional PRS scheme where the size of signatures and the verification cost do not grow linearly with the number of translations remains open. In this thesis, we introduce the notion of attribute-based PRS and propose the first provably secure attribute-based PRS scheme based on bilinear pairing in the generic group model. This construction solves the open problem of finding out a multi-use unidirectional PRS scheme where the signature size and the verification cost do not grow linearly with the number of translations.

1.3 Organization of This Thesis

The rest of this thesis is organized as follows.

In Chapter 2, we review some preliminaries. We introduce cyclic group, bilinear pairing, Lagrange interpolation, monotone span programs, generic group model, and present some assumptions used in the process of security proof. We also describe some basic cryptographic primitives, such as digital signature, IBC, ABC, NIZK proof, AEAH, multisignature, ring signature, quotable signature, and PRS.

In Chapter 3, we propose an identity-based multisignature with message recovery scheme. Firstly, we introduce the background about identity-based multisignature and digital signature with message recovery. Then, we propose the formal definition of identity-based multisignature with message recovery scheme and present the security model. Finally, we present a concrete scheme based on bilinear pairing and security proofs of the proposed scheme.

In Chapter 4, we design a generic construction of identity-based authenticated encryption with authenticated header scheme. We propose the formal definition of identity-based authenticated encryption with authenticated header scheme and present the security model. We present a generic construction from a normal IBSMR scheme and a symmetric encryption scheme. We also give a concrete instance of the generic construction.

In Chapter 5, we propose an identity-based quotable ring signature scheme. We propose the formal definition of identity-based quotable ring signature scheme and present the security model. We present a concrete scheme based on bilinear pairing, and security proofs based on subgroup decision problem and CDH problem for the proposed scheme.

In Chapter 6, we give two attribute-based signature with message recovery schemes. We propose the formal definition of attribute-based signature with message recovery scheme and present the security model. We construct two concrete schemes based on bilinear pairing and present security proofs of the proposed schemes.

In Chapter 7, we construct an attribute-based proxy re-signature scheme. We propose the formal definition of attribute-based proxy re-signature scheme and present the security model. We present a concrete scheme based on bilinear pairing and security proofs about correctness, attribute privacy, and existentially unforgeability of the proposed scheme.

Chapter 8 concludes this thesis and provides a future research direction.

Chapter 2

Preliminaries

We introduce some preliminaries used throughout this thesis, including some mathematical foundations and cryptographic primitives.

2.1 Cyclic Group

A group consists of a set of elements together with an operation. The operation could be conducted between any two of its elements to work out another element. The operation has to satisfy four conditions: closure, associativity, identity, and invertibility [Mao04].

Definition 2.1 Group. *A group $(\mathbb{G}, \circ)$ is a set $\mathbb{G}$ together with an operation $\circ$, which satisfies the following properties.*

- *Closure.* $\forall \alpha, \beta \in \mathbb{G}, \alpha \circ \beta \in \mathbb{G}$.
- *Associativity.* $\forall \alpha, \beta, \gamma \in \mathbb{G}, (\alpha \circ \beta) \circ \gamma = \alpha \circ (\beta \circ \gamma)$.
- *Identity.* $\exists$ *unique element* $I \in \mathbb{G}$, *such that* $\forall \alpha \in \mathbb{G}, \alpha \circ I = I \circ \alpha = \alpha$. *This unique element* I *is called the identity element of this group.*
- *Invertibility.* $\forall \alpha \in \mathbb{G}, \exists \alpha^{-1} \in \mathbb{G}$, *such that* $\alpha \circ \alpha^{-1} = \alpha^{-1} \circ \alpha = I$. α^{-1} *is called the inverse of* α.

Hereafter, when we mention a group $(\mathbb{G}, \circ)$, the operation $\circ$ is usually omitted for simplicity.

Definition 2.2 Order of Group. *The number of elements in a group $\mathbb{G}$ is called the order of the group $\mathbb{G}$.*

Definition 2.3 Finite and Infinite Group. *When the order of a group is finite, the group is called a finite group. When the order of a group is infinite, it is called an infinite group.*

If every element of a group $\mathbb{G}$ is a power of a fixed element of that group. We say the group is a cyclic group.

Definition 2.4 Cyclic Group. *A group $\mathbb{G}$ is said to be cyclic if there exists an element $\alpha \in \mathbb{G}$, such that for every element $\beta \in \mathbb{G}, \beta = \alpha^n$ for some integer n. Element α is called a generator of the group $\mathbb{G}$. We say that α generates $\mathbb{G}$, and write as $\mathbb{G} =< \alpha >$.*

Every cyclic group is an abelian group, which means its group operation is commutative. Every subgroup of a cyclic group is still a cyclic group. Every infinite cyclic group is isomorphic to the additive group of $\mathbb{Z}$, the integers. Every finite cyclic group of order n is isomorphic to the additive group of $\mathbb{Z}/n\mathbb{Z}$, the integers modulo n.

Definition 2.5 Prime Order Group. *For finite groups, when the order of a group is a prime number, it is called a prime order group.*

All groups of prime order are cyclic groups. Hence, they are also abelian groups. Any non-identity element in a prime order group generates the whole group.

Definition 2.6 Composite Order Group. *For finite groups, when the order of a group is a composite number, it is called a composite order group.*

When the composite order group is used in cryptographic protocols, the composite order is usually public, while the factorization of the composite order is kept secret. The security of cryptosystems using composite order groups is usually based on variants of the subgroup decision assumption.

2.2 Bilinear Pairing

Bilinear pairing is a useful tool in the pairing-based cryptography. Firstly, it was used by Menezes, Okamato and Vanstone [MOV93] to attack elliptic curve cryptographic systems. It became a useful tool since the provably secure identity-based

encryption (IBE) scheme proposed by Boneh and Franklin [BF01] in 2001. It associates a pair of elements in two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with an element in the third group $\mathbb{G}_T$. $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are all isomorphic to each other because they are all cyclic groups and have the same order. The order of these cyclic groups may be either prime or composite number.

For simplicity, $\mathbb{G}_1$ and $\mathbb{G}_2$ are usually the same group. They are both denoted by $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$. In this case, the pairing is called symmetric pairing. For convenience, operations in group $\mathbb{G}$ could be written either additively or multiplicatively. Essentially, both of these representations are the same operation of group $\mathbb{G}$.

Definition 2.7 Bilinear mapping. *Let $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups of the same order. $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$ is a bilinear mapping from $\mathbb{G}\times\mathbb{G}$ to $\mathbb{G}_T$ with the following properties.*

- *Bilinearity: $e(u^a, v^b) = e(u, v)^{ab}$ for all $\{u, v\} \in \mathbb{G}, \{a, b\} \in \mathbb{Z}_n$.*
- *Non-degeneracy: $< e(g, g) >= \mathbb{G}_T$ whenever $< g >= \mathbb{G}$.*
- *Computability: There exists an efficient algorithm to compute $e(u, v)$ for all $\{u, v\} \in \mathbb{G}$.*

The Decisional Diffie-Hellman (DDH) problem in $\mathbb{G}$ is not hard when $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$ in bilinear pairing. The DDH problem in $\mathbb{G}_T$ is still hard. When the groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are distinct and there is no efficient computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$, the DDH problem might be still hard in $\mathbb{G}_1$ and $\mathbb{G}_2$. The Computational Diffie-Hellman (CDH) problem in $\mathbb{G}$ is still hard when $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$. If the DDH problem is easy but the CDH problem is hard in a group $\mathbb{G}$, the group is called a gap Diffie-Hellman group. Since the DDH problem is easy in bilinear pairing, some new hard problems are defined, such as the computational Bilinear Diffie-Hellman (BDH) problem and the Decisional Bilinear Diffie-Hellman (DBDH) problem.

Bilinear pairing usually uses prime order groups. Recently, bilinear groups of composite order began to be used to construct cryptographic systems by Boneh, Goh, and Nissim [BGN05]. Let n be a composite number with factorization $n = pq$, where p and q are sufficient large prime numbers. We have: $\mathbb{G}$ is a cyclic group of composite order n. $\mathbb{G}_p$ is its cyclic order-p subgroup, and $\mathbb{G}_q$ is its cyclic order-q subgroup. g is a generator of $\mathbb{G}$. $\mathbb{G}_T$ is a cyclic group of composite order n. $\mathbb{G}_{T,p}$ and $\mathbb{G}_{T,q}$ are its order-p and order-q subgroups, respectively. Let $g_p = g^q$ be a generator of $\mathbb{G}_p$, and $g_q = g^p$ be a generator of $\mathbb{G}_q$, then, $e(g_p, g_q) = 1$. They cancel each other out.

2.3 Lagrange Interpolation

Polynomial interpolation is used to find a polynomial which goes exactly through some given set of discrete points. Lagrange interpolation is used in secret sharing scheme [Sha79] in cryptography.

Given d points $(x_1, q(x_1)), \cdots, (x_d, q(x_d))$ on a $(d-1)$ degree polynomial, where no two of them are the same, the $(d-1)$ degree polynomial $q(x)$, which goes exactly though these d points, is uniquely determined by these d points. Let S be the d-element set $(x_1, \cdots, x_d)$. The Lagrange coefficient $\Delta_{x_j,S}(x)$ of $q(x_j)$ in the computation of $q(x)$ is:

$$\Delta_{x_j,S}(x) = \prod_{\eta \in S, \eta \neq x_j} \frac{x-\eta}{x_j-\eta}.$$

The polynomial $q(x)$ is a linear combination of the form

$$q(x) = \sum_{j=1}^{d} q(x_j)\Delta_{x_j,S}(x).$$

Lagrange interpolation is used to compute $q(i)$ for any $i \in \mathbb{Z}_p$, while some discrete points are given.

2.4 Monotone Span Programs

Let $\Upsilon : \{0,1\}^n \to \{0,1\}$ be a monotone boolean function. A monotone span program for Υ over a field $\mathbb{F}$ is an $l \times t$ matrix $\mathcal{M}$ with entries in $\mathbb{F}$, along with a labeling function $u : [l] \to [n]$ that associates each row of $\mathcal{M}$ with an input variable of Υ, that satisfies the following [MPR08]:

$$\Upsilon(x_1, \cdots, x_n) = 1 \Leftrightarrow \exists \overrightarrow{v} \in \mathbb{F}^{1\times l} : \overrightarrow{v}\mathcal{M} = [1, 0, 0, \cdots, 0] \\ \text{and } (\forall i : x_{u(i)} = 0 \Rightarrow v_i = 0)$$

In other words, $\Upsilon(x_1, \cdots, x_n) = 1$ if and only if the rows of $\mathcal{M}$ indexed by $\{i | x_{u(i)} = 1\}$ span the vector $[1, 0, 0, \cdots, 0]$.

We call l the length and t the width of the span program, and $l + t$ the size of the span program. Every monotone boolean function can be represented by some monotone span program [Bei96], and a large class do have compact monotone span programs.

2.5 Generic Group Model

Generic group model was first introduced by Shoup [Sho97] where one assumes that access to group elements is via a randomly selected representation. Maurer [Mau05] gave more interpretations and generalizations.

The generic group model is used to discuss the security of cryptographic schemes. It is an idealized cryptographic model. In this model, a group is considered as a black-box, and group elements are randomly encoded. All details of group elements representation are hidden. Only basic group operations on the elements of the group such as applying the group law, inversion of group elements and equality testing are allowed to be executed. A generic group algorithm executes as an oracle, which can execute only basic group operations. It cannot exploit any special properties of a concrete group representation. It takes encodings of two group elements as input and outputs an encoding of a third element. The generic group model aims to capture the idea that a scheme is secure on some unspecified group.

The generic group model is also used for providing evidence about newly introduced hardness assumptions.

2.6 Complexity Problem

2.6.1 Computational Diffie-Hellman Problem

The computational Diffie-Hellman problem was proposed by Diffie and Hellman [DH76].

Definition 2.8 Computational Diffie-Hellman Problem. *Let $\mathbb{G}$ be a cyclic group of order q. Let g be a generator of $\mathbb{G}$. Let $\mathcal{A}$ be an attacker. $\mathcal{A}$ tries to solve the following problem: Given (g, g^a, g^b) for some unknown $a, b \in \mathbb{Z}_q^*$, compute g^{ab}.*

The advantage of an probabilistic algorithm $\mathcal{A}$, which is polynomially bounded with a security parameter λ, is defined as

$$Adv_{\mathbb{G},\mathcal{A}}^{CDH}(\lambda) = \Pr[\mathcal{A}(g, g^a, g^b, g^{ab}) = 1],$$

where a, b are drawn from the uniform distribution on $\mathbb{Z}_q^*$.

The CDH problem is said to be intractable, if for every probabilistic polynomial time (PPT) algorithm $\mathcal{A}$, $Adv_{\mathbb{G},\mathcal{A}}^{CDH}(\lambda)$ is negligible.

2.6.2 Decisional Diffie-Hellman Problem

Boneh [Bon98] surveyed the decisional Diffie-Hellman problem and demonstrated its security.

Definition 2.9 Decisional Diffie-Hellman Problem. *Let $\mathbb{G}$ be a cyclic group of order q. Let g be a generator of $\mathbb{G}$. Let $\mathcal{A}$ be an attacker. $\mathcal{A}$ tries to distinguish (g, g^a, g^b, g^{ab}) from (g, g^a, g^b, g^z) for some unknown $a, b, z \in \mathbb{Z}_q^*$.*

The advantage of an probabilistic algorithm $\mathcal{A}$, which is polynomially bounded with a security parameter λ, is defined as

$$Adv_{\mathbb{G},\mathcal{A}}^{DDH} = \left| Pr[\mathcal{A}(g, g^a, g^b, g^{ab}) = 1] - Pr[\mathcal{A}(g, g^a, g^b, g^z) = 1] \right|,$$

where a, b, z are drawn from the uniform distribution on $\mathbb{Z}_q^*$.

The DDH problem is said to be intractable, if for every probabilistic polynomial time algorithm $\mathcal{A}$, $Adv_{\mathbb{G},\mathcal{A}}^{DDH}(\lambda)$ is negligible.

2.6.3 Computational Bilinear Diffie-Hellman Problem

The computational bilinear Diffie-Hellman problem was introduced by Boneh and Franklin [BF01].

Definition 2.10 Computational Bilinear Diffie-Hellman Problem. *Let $\mathbb{G}$ and $\mathbb{G}_T$ be two groups of the same order q. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear mapping on $(\mathbb{G}, \mathbb{G}_T)$. Let g be a generator of $\mathbb{G}$. The computational bilinear Diffie-Hellman (BDH) problem is the following: Given (g, g^a, g^b, g^c) for some unknown $a, b, c \in \mathbb{Z}_q^*$, compute $e(g, g)^{abc}$.*

The advantage of an probabilistic algorithm $\mathcal{A}$, which is polynomially bounded with a security parameter λ, is defined as

$$Adv_{\mathbb{G},\mathcal{A}}^{BDH}(\lambda) = \Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 1],$$

where a, b, c are drawn from the uniform distribution on $\mathbb{Z}_q^*$.

The BDH problem is said to be intractable, if for every probabilistic polynomial time algorithm $\mathcal{A}$, $Adv_{\mathbb{G},\mathcal{A}}^{BDH}(\lambda)$ is negligible.

2.6.4 Decisional Bilinear Diffie-Hellman Problem

The decisional bilinear Diffie-Hellman problem was also introduced by Boneh and Franklin [BF01] to construct provably secure IBE.

Definition 2.11 Decisional Bilinear Diffie-Hellman Problem. *Let $\mathbb{G}$ and $\mathbb{G}_T$ be two groups of the same order q. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear mapping on $(\mathbb{G}, \mathbb{G}_T)$. Let g be a generator of $\mathbb{G}$. The decisional bilinear Diffie-Hellman (DBDH) problem is to distinguish $(g, g^a, g^b, g^c, e(g,g)^{abc})$ from $(g, g^a, g^b, g^c, e(g,g)^z)$ for some unknown $a, b, c, z \in \mathbb{Z}_q^*$.*

The advantage of an probabilistic algorithm $\mathcal{A}$, which is polynomially bounded with a security parameter λ, is defined as

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

where a, b, c, z are drawn from the uniform distribution on $\mathbb{Z}_q^*$.

The DBDH problem is said to be intractable, if for every probabilistic polynomial time algorithm $\mathcal{A}$, $Adv_{\mathbb{G},\mathcal{A}}^{DBDH}(\lambda)$ is negligible.

2.6.5 Subgroup Decision Problem

The subgroup decision problem says that given an element $g \in \mathbb{G}$, where $\mathbb{G}$ is a composite order $n = pq$ cyclic group, there is no efficient algorithm to determine whether g has order p. In particular, this problem implies that it is infeasible to factor the group order n. It was introduced by Boneh, Goh, and Nissim [BGN05]

Definition 2.12 Subgroup Decision Problem. *Given a composite order $n = pq$ cyclic group $\mathbb{G}$ and its subgroup $\mathbb{G}_q$ of order q, w is selected at random either from $\mathbb{G}$ (with probability 1/2) or from $\mathbb{G}_q$ (with probability 1/2). Decide whether w is in $\mathbb{G}_q$.*

The advantage of an algorithm $\mathcal{A}$ solving the subgroup decision problem is defined as $\mathcal{A}$'s excess probability beyond 1/2 of outputting the correct solution. The subgroup decision problem is said to be intractable, if for every PPT algorithm $\mathcal{A}$, the success probability of guessing advantage for the problem is negligible. The assumption that the subgroup decision problem is hard is called the Subgroup Hiding (SGH) assumption [BGN05].

2.7 Cryptographic Primitives

2.7.1 Digital Signature

In the digital world, digital signature implements functions of traditional handwritten signature by using mathematical approaches. It is usually used to convince the verifier that a claimed sender has endorsed a message and the message has not been altered during its transmission. On one hand, the sender cannot deny he has sent the message. On the other hand, digital signature confirms the integrity of digital messages. The generation of digital signatures depends on some secret known only to the signer and the messages to be signed. The verification only requires public information of the signer and the signed message.

In 1976, Diffie and Hellman [DH76] introduced the notion of digital signature in the public key cryptography. After that, Rivest, Shamir, and Adleman [RSA78] proposed the famous RSA algorithm. In 1988, Goldwasser, Micali, and Rivest [GMR88] formalized the security model of digital signature schemes. They also presented the first digital signature scheme which can be proven to prevent existential forgery against chosen message attacks.

A typical digital signature scheme consists of the following three algorithms:

- **KeyGen:** On input of the security parameter λ, this algorithm outputs user's private key SK and the corresponding public key PK.
- **Sign:** On input of the message m to be signed and a signer's private key SK, this algorithm produces a signature σ of m on behalf of the signer.
- **Verify:** On input of a claimed message m, a claimed signer's public key PK and a claimed signature σ, this algorithm either accepts or rejects the claimed signature on the claimed message depending on its validity.

Two main properties should be ensured about digital signature schemes. One is *correctness*, which means for every validly generated signature, it will pass the verification. The other is *unforgeability*, which means without knowing the actual signer's private key, anyone should be computationally infeasible to forge a valid signature on behalf of the actual signer.

2.7.2 Identity-Based Cryptography

In the identity-based cryptography, a user's public key is no longer attested by digital certificate. It could be publicly computed from the user's identity which is an arbitrary single string associated with the user, such as an email address or an IP address. User's private key should be computed by a trusted third party, called the Private Key Generator (PKG), after the user's identity is authenticated. Along with the emergence of the concept of identity-based cryptography [Sha85], the first identity-based signature scheme was also proposed by using the RSA [RSA78] function. The provably secure identity-based encryption was still an open problem until 2001, when Boneh and Franklin [BF01] and Cocks [Coc01] solved it independently.

A typical identity-based signature scheme consists of the following four algorithms.

- **Setup:** On input of a security parameter λ, the trusted third party PKG creates its master secret key MK and public parameters $params$. The master secret key MK is kept secret, which is only known to the PKG. The public parameters $params$ are published to as many as possible interested parties.

- **Extract:** After authenticating the signer, the PKG takes the public parameters $params$, the signer's identity ID, and the master secret key MK as input, and gives the private key S_{ID} to the signer.

- **Sign:** The signer takes the public parameters $params$, his private key S_{ID} and the message m, which is about to be signed, as input, and outputs a signature σ on the message m.

- **Verify:** The verifier takes the claimed signature σ, the signed message m, and the signer's identity ID as input, and either accepts or rejects the signature σ depending on its validity.

A typical identity-based encryption scheme consists of the following four algorithms.

- **Setup:** On input of a security parameter λ, the trusted third party PKG creates its master secret key MK and public parameters $params$. The master secret key MK is kept secret, which is only known to the PKG. The public parameters $params$ are published to as many as possible interested parties.

- **Extract:** After authenticating the receiver, the PKG takes the public parameters $params$, the receiver's identity ID and the master secret key MK as input, and gives the private key S_{ID} to the receiver.
- **Encrypt:** The sender takes the public parameters $params$, the receiver's identity ID, the message m, which is about to be encrypted, as input, and outputs a ciphertext C on the message m.
- **Decrypt:** After receiving the ciphertext C, the receiver uses the private key S_{ID} and the public parameters $params$ to restore the plaintext m.

2.7.3 Attribute-Based Signature

In traditional digital signature schemes, every user possesses a randomly chosen private key and a corresponding public key. Every signature is associated with a particular public key. Only the user who possesses the corresponding private key can sign materials associated with the specified public key. This situation remains in the identity-based signature schemes. In the attribute-based signature schemes, signatures are no longer associated with a particular public key, but associated with a set of attributes. Attribute-based signature was first introduced as fuzzy identity-based signature [YCD08]. It is also proposed in [SY08, MPR08]. Attribute-based signatures endorse messages by users who claim to have attributes satisfying some policies. Every user whose attributes satisfies the policy can sign the message. Thus, attribute-based signature supports fine-grain access control. In addition to this, attribute-based signature also provides attribute privacy, which means in the attribute-based signature scheme the signature reveals nothing about the attributes of the signer beyond what is explicitly revealed by the claim being made. The verification does not reveal how the policy was satisfied. Furthermore, users cannot collude to pool their attributes together.

A typical attribute-based signature scheme consists of the following four algorithms.

- **Setup:** On input of a security parameter λ and the universe of attributes $\mathcal{U}$, the central authority creates its master secret key MK and public parameters $params$. The master secret key MK is kept secret, which is only known to the central authority. The public parameters $params$ are published to as many as possible interested parties.

- **Extract:** After authenticating the signer who possesses some attribute set ω, the central authority takes the signer's attribute set ω and the master secret key MK as input, and gives the private key $\{D_i\}_{i\in\omega}$ corresponding to the attribute set ω to the signer.

- **Sign:** In order to sign a message m for a predicate Υ, the signer takes the public parameters $params$, private key $\{D_i\}_{i\in\omega}$ corresponding to the attribute set ω, and the message m, which is about to be signed, as input, and outputs a signature σ on the message m when the attributes ω satisfy the predicate Υ.

- **Verify:** After receiving a claimed signature σ, the signed message m, and a predicate Υ, this algorithm takes the public parameters $params$ as input to check the validity of the signature. Then, the claimed signature σ is either accepted or rejected depending on its validity.

2.7.4 Non-Interactive Zero Knowledge Proof

Zero knowledge proof was introduced by Goldreich, Micali, and Wigderson [GMW86]. In a zero knowledge proof, the prover proves to the verifier that a statement is true, without leaking any information apart from the fact that the statement is really true.

A zero-knowledge proof must satisfy three properties:

1. **Completeness:** If a statement is true, an honest verifier will be convinced of this fact by an honest prover.

2. **Soundness:** If a statement is false, a cheating prover cannot convince an honest verifier that the statement is true, except with very small probability.

3. **Zero-knowledge:** If a statement is true, no verifier can get any extra information beyond the fact that the statement is indeed true.

After sharing a common random string between the prover and the verifier, Blum, Feldman, and Micali [BFM88] showed that zero knowledge proof could be achieved without interaction between the prover and the verifier. This is the so-called non-interactive zero knowledge proof. Pass [Pas03] showed that some properties, such as deniability of the interactive zero knowledge proof, is not preserved in the common reference string model non-interactive zero knowledge proof. However, due to its

good privacy, authentication, and non-interactive property, it is widely used for non-interactive tasks.

2.7.5 Authenticated Encryption with Authenticated Header

Authenticated encryption provides simultaneously confidentiality, integrity, and authenticity. It has been studied by Bellare, and Namprempre [BN00]. Authenticated encryption became a distinct cryptographic primitive mainly because that trivially gluing an encryption scheme and a signature scheme usually results in insecure system [BRW04, KVW04].

Authenticated encryption with authenticated header in the symmetric key setting has been studied by Rogaway [Rog02]. The header provides authenticity and integrity for messages which confidentiality is not required, but authenticity is desired.

A typical authenticated encryption with authenticated header scheme consists of the following three algorithms.

- **KeyGen:** On input of the security parameter λ, this algorithm outputs user's private key SK and the corresponding public key PK.
- **Authenticated Encryption:** On input of the plaintext m, the sender's private key SK_{sender}, the receiver's public key $PK_{receiver}$, and a header h in plaintext, which will not be encrypted but will be authenticated, the authenticated encryption algorithm produces an authenticated ciphertext C.
- **Authenticated Decryption:** On input of the authenticated ciphertext C, the sender's public key PK_{sender}, the receiver's private key $SK_{receiver}$, and the header h, the authenticated decryption algorithm outputs the plaintext m when the ciphertext C and the header h are verified as valid, or an error otherwise.

2.7.6 Multisignature

A multisignature convinces the verifier that a certain number of signers have endorsed the given message. The number of signers of a multisignature is not fixed. A multisignature is much shorter than the simple collection of the corresponding individual signatures.

Multisignature was introduced by Itakura and Nakamura [IN83], and has been formalized by Ohta and Okamoto [OO99] and Micali, Ohta, and Reyzin [MOR01].

A normal multisignature scheme consists of the following three algorithms.

- **MKeyGen:** On input of the security parameter λ, this algorithm outputs user's private key SK and the corresponding public key PK.
- **MSign:** This is usually an interactive algorithm run by arbitrary number of signers. On input of each signer's private key SK_i and the message m to be signed, it outputs a multisignature σ.
- **MVerify:** The verification algorithm takes as input a claimed multisignature σ on a message m and all signers' public keys PK_i. It either accepts or rejects the claimed multisignature on the claimed message depending on its validity..

2.7.7 Ring Signature

In the ring signature scheme, an actual signer among a set of users signs messages using his private key, his public key, and other users' public keys. A ring signature convinces the verifier that a message is endorsed by one of the ring members, but do not reveal which member is the actual signer. Ring signature was introduced by Rivest, Shamir, and Tauman [RST01].

Usually, A ring signature scheme consists of the following three algorithms.

- **KeyGen:** On input of the security parameter λ, this algorithm outputs user's private key SK and the corresponding public key PK.
- **Sign:** On input of the signers' public keys $\{PK_i\}_{i \in R}$ included in the ring R, the actual signer's private key SK_{i^*}, and a message m, this algorithm outputs a ring signature σ on the message m on behalf of the whole ring R.
- **Verify:** The verifier takes the claimed ring signature σ, the message m, and the public keys $\{PK_i\}_{i \in R}$ of all members of the ring R as input, and either accepts or rejects the claimed ring signature on the claimed message depending on its validity.

2.7.8 Quotable Signature

In the quotable signature schemes, for every substring m' of a message m, it is possible for a third party to derive a signature on m' from a signature on m on behalf of the same signer. Moreover, the derived signature on m' reveals no extra information about m, which means the derived signature cannot be distinguished from a fresh one even when the original signature on m is given. In the quotable signature, only signatures on arbitrary substrings of the original message could be derived. Signatures on subsequences of the original message are not allowed to be derived. Quotable signature was introduced by Ahn et al. [ABC+12]. Then, it is improved by Attrapadung, Libert, and Peters [ALP13].

Normally, a quotable signature scheme consists of the following four algorithms.

- **KeyGen:** On input of the security parameter λ, this algorithm outputs user's private key SK and the corresponding public key PK.
- **Sign:** On input of the message m to be signed and a signer's private key SK, this algorithm produces a signature σ of m on behalf of the signer.
- **Quote:** This algorithm takes as input a signature σ on a message m, and a substring m' of m. It first checks the validity of σ. If σ is valid, it produces a new signature σ' on m' on behalf of the same signer. Otherwise, it outputs a special symbol $\perp$ to represent failure.
- **Verify:** The verification algorithm takes as input a purported signature σ on a message m and the corresponding public key PK. It either accepts or rejects the claimed signature on the claimed message depending on its validity.

2.7.9 Proxy Re-Signature

In the proxy re-signature schemes, a semi-trusted proxy, who is given some secret information of the delegator, say Bob, acts as a translator between the delegatee Alice and the delegator Bob. The proxy is able to convert a signature from Alice into a signature from Bob on the same message, while the proxy cannot learn any signing key and cannot sign arbitrary messages on behalf of either Alice or Bob.

The notion of proxy re-signature was introduced by Blaze, Bleumer, and Strauss [BBS98], then formalized by Ateniese and Hohenberger [AH05].

Normally, a proxy re-signature scheme consists of the following five algorithms.

- **KeyGen:** On input of the security parameter λ, this algorithm outputs user's private key SK and the corresponding public key PK.

- **ReKey:** On input of the private keys and the public keys of the delegator PK_{Bob}, SK_{Bob} and the delegatee PK_{Alice}, SK_{Alice}, in which the delegatee's private key SK_{Alice} is usually not required, this algorithm outputs a re-signing key $RK_{Alice \to Bob}$ for the proxy. The re-signing key $RK_{Alice \to Bob}$ allows the proxy to transform Alice's signatures into Bob's signatures. Therefore, Bob is the delegator, and Alice is the delegatee.

- **Sign:** On input of the message m to be signed and a signer's private key SK, this algorithm produces a signature σ of m on behalf of the signer.

- **ReSign:** On input of a re-signing key $RK_{Alice \to Bob}$, the public key of the delegatee PK_{Alice}, a purported signature σ_{Alice} of the delegatee, a message m, this algorithm outputs a signature σ_{Bob} on the message m on behalf of the delegator, if the purported signature σ_{Alice} of the delegatee on the same message if valid.

- **Verify:** On input of the claimed message m, the claimed sender's public key PK and a claimed signature σ, this algorithm either accepts or rejects the claimed signature on the claimed message depending on its validity.

Chapter 3

Identity-Based Multisignature with Message Recovery

In this chapter, we present a new notion of *identity-based multisignature with message recovery*. We propose a concrete identity-based multisignature with message recovery scheme based on bilinear pairing in which multiple signers can generate a single constant size multisignature on the same message regardless of the number of signers. There is no requirement to transmit the original message to the verifier, since the original message can be recovered. Therefore, this scheme minimizes the total length of the original message and the appended multisignature. The proposed scheme is proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the CDH problem is hard. The original scheme was presented at *ISPEC 2013*.

3.1 Introduction

In networks with limited bandwidth, long digital signatures will obviously be a drawback. Apart from shortening the signature itself, the other effective method for saving bandwidth in transmission is to eliminate the requirement of transmitting the original message for the signature verification. In this work, we consider on the latter approach.

Consider n different signers. In order to allow any subgroup of them to produce a joint signature on a message m and convince a verifier that each member of the stated subgroup signed the message, two or more signers cooperate to generate a single compact digital signature in a multisignature scheme. A single multisignature can greatly save communication costs instead of transmitting several individual signatures. To verify the validity of a multisignature, one still needs public keys of all signers. In most applications these public keys will have to be transmitted along

with the multisignature. In this case, it partially defeats the primary purpose of using a multisignature scheme, namely to save bandwidth. But the inclusion of some information that uniquely identifies the signers seems inevitable for the verification. Fortunately, in the identity-based setting, this information can be represented in a more succinct way.

Compared to the public key of the signer is essentially random string generated from random secret key in traditional public key signature schemes, in the identity-based scenario, the public key of a signer is simply his identity such as name, email address or IP address. The associated private key can only be computed by a trusted Private Key Generator (PKG). It can avoid using certificates. These features make the identity-based concept particularly appealing for use in conjunction with multisignatures.

When bandwidth is at a premium, another potential problem is that the combined length of the original message and the signature is large. Signature schemes with total or partial message recovery provide a solution to this problem by embedding all or part of the message within the signature. That is, the message does not need to be hashed or sent along with the signature, which saves storage space and communication bandwidth.

In the work of this chapter, the signers are revealed to the verifiers by using their identities. In Chapter 6, we introduce another notion related to digital signature with message recovery. In that work, the signer is anonymous. The real signer is hidden in the signers whose attributes satisfy a certain predicate.

Related Work. In 1984, Shamir introduced the notion of identity-based cryptography to simplify the key management of certificate-based public key infrastructure and proposed an identity-based signature scheme [Sha85]. Since then several practical identity-based signature schemes have been devised [FS87, GQ90, CHC02, Hes03]. Cha and Cheon [CHC02] proposed an identity-based signature scheme using the gap Diffie-Hellman (GDH) groups, and proved their scheme is secure against existential forgery on adaptively chosen message and ID attack under the random oracle model. Hess [Hes03] also proposed an efficient identity-based signature scheme based on pairings. The security of their scheme relies on the hardness of the Diffie-Hellman problem in the random oracle model.

The notion of multisignature was introduced by Itakura and Nakamura [IN83]. Several works on this topic have been done [Bol02, MOR01, Oka99]. In [MOR01], the first formalized strong notion of security for multisignature was proposed. They

modified the Schnorr-signature-based multisignature scheme originally proposed by Ohta and Okamoto [Oka99] and proved its security. Gangishetti et al. [GGDS06] presented identity-based serial and parallel multisignature schemes using bilinear pairings. Harn and Ren [HR08] proposed an efficient RSA multisignature scheme based on Shamir's identity-based signature.

In order to minimize the total length of the original message and the appended signature, the message recovery schemes were introduced (e.g. [NR93]). Zhang et al. [ZSM05] proposed an identity-based message recovery signatures scheme. Their scheme can be regarded as the identity-based version of Abe-Okamoto's scheme [AO99]. Their scheme was also extended to achieve an identity-based partial message recovery signature scheme. Based on the scheme due to Zhang et al. [ZSM05], we achieve the goal of minimizing the total length of the original message and the appended multisignature in the identity-based setting.

Our Contributions. We present a provably secure (existentially unforgeable against adaptively chosen message attacks) identity-based multisignature with message recovery scheme based on bilinear pairing under the Computational Diffie-Hellman assumption in the random oracle model. Since the original message can be recovered, there is no need to transmit the original message to the verifier. This scheme minimizes the total length of the original message and the multisignature. It also could be viewed as a short identity-based multisignature scheme, since it eliminates the requirement of transmitting the original message for the signature verification. We also present a concrete analysis of the reduction to prove the security of the proposed multisignature scheme. More precisely, we can show that if there is an attacker who can forge a valid multisignature to pass the verification, then the Computational Diffie-Hellman problem is solved.

Organization of This Chapter. The rest of this chapter is organized as follows: In Section 3.2, we provide a formal definition of the identity-based multisignature with message recovery scheme. In Section 3.3, we present a security model for the new scheme. In Section 3.4, we present a concrete identity-based multisignature with message recovery scheme based on bilinear pairing. Section 3.5 provides the security proof for the proposed scheme. Section 3.6 concludes this chapter.

3.2 Formal Definitions

In an identity-based multisignature with message recovery scheme, there is a trusted party Private Key Generator (PKG). PKG is required to generate all the users' private keys.

There are three parties in the system, the PKG, the signer and the verifier. The scheme is ideal for closed groups of users such as the executives of a multinational company or the branches of a large bank, since the headquarters of the corporation can serve as a key generation centre that everyone trusts. This scheme consists of the following four algorithms.

Setup: PKG sets up its secret key with respect to a security parameter λ as the master key of this scheme and publishes the corresponding public key. PKG should generate related groups and point out the generator of these groups. PKG also should describe which bilinear mapping and hash functions will be used in this scheme and publish these public information to all interested principals.

Extract: When a principal requires its private key S_{ID} corresponding its identity ID, this algorithm generates the private key using the master key and the principal's identity, and returns the private key to the principal.

Sign: This is an interactive algorithm. Several principals who got their private keys from the Extract algorithm can firstly generate their individual signatures on a message m respectively, and one of them or other specified trusted principal can generate a single compact multisignature σ on the message m corresponding to these principals who participate in this algorithm.

Verify: On receiving a multisignature σ and several principals' identities ID_1, $ID_2, \cdots, ID_n$, this algorithm checks whether the multisignature is valid corresponding to these principals' public keys. If the multisignature is valid, the original message m can be recovered from this multisignature.

3.3 Security Model

Boldyreva [Bol02] defined the notion of security for multisignature as no valid multisignature should keep an honest player that part of the alleged subgroup accountable if it did not participate in signing. That is to say, no adversary can forge an alleged multisignature of some message corresponding to an alleged subgroup of

signers so that a verifier can check the multisignature as valid when not all signers of the alleged subgroup did sign the message. In order to achieve its goal, an adversary is allowed to corrupt players and send arbitrary messages during the multisignature generation process.

We use a similar definition of existential unforgeability against chosen message attacks of [Bol02]. Our definition is strong enough to capture an adversary who can simulate and observe the scheme. It is defined using the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

Assume in a subgroup of n signers who want to participate in generating a multisignature, there is only one honest signer. All other $n-1$ members of the subgroup have been corrupted by the adversary. This means the adversary can get secret keys of the corrupted signers. But the adversary only knows the public key of the single honest signer. The adversary can participate in the multisignature generation process on behalf of these $n-1$ corrupted signers. Its goal is to frame the honest signer.

Firstly, challenger $\mathcal{C}$ runs the Setup algorithm to get the system's master secret key with respect to a security parameter λ and sends the system's public parameters *params* to adversary $\mathcal{A}$. $\mathcal{A}$ can access to some random oracles. In these random oracles, for every unique query, they respond a random response. If a query has been submitted before, they respond the same value as they responded the first time. $\mathcal{A}$ can also access to the following oracles to conduct an attack.

Extract Oracle: For each Extract query with respect to a user ID_i except for the honest user ID^*, $\mathcal{C}$ returns S_{ID_i} as the user's private key.

Sign Oracle: For each Sign query on arbitrary message m with respect to a subgroup of n signers' identities $ID_1, ID_2, \cdots, ID_n$, this oracle returns a valid multisignature σ on the message m with respect to these n signers.

Output: $\mathcal{A}$ outputs an alleged multisignature σ^* on a message m with respect to a subgroup of n signers $ID_1, \cdots, ID^*, \cdots, ID_n$ in which includes an honest signer ID^* who did not participate in the multisignature generation process. If there was no Sign queries with respect to the message m and a subgroup of signers in which includes the honest signer ID^* have been queried to the Sign Oracle, and there was no Extract query with respect to the honest signer ID^* has been queried to the Extract Oracle, $\mathcal{A}$ wins the game if the multisignature σ^* can be verified as a valid multisignature.

If there is no such polynomial-time adversary that can forge a valid multisignature with respect to a subgroup of signers which includes an honest signer, while the honest signer did not participate in the multisignature generation process in the game described above, we say that the multisignature scheme is secure against existential forgery under the chosen message attack.

The success probability of an adversary to win the game is defined by

$$Succ_{\mathcal{A}}^{UF-IDMMR-CMA}(\lambda).$$

We say that an identity-based multisignature with message recovery scheme is existentially unforgeable under a chosen message attack if the success probability of any polynomially bounded adversary in the above game is negligible. In other words,

$$Succ_{\mathcal{A}}^{UF-IDMMR-CMA}(\lambda) \leq \epsilon(\lambda).$$

3.4 Proposed Scheme

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of the same prime order q. Let P be a generator of $\mathbb{G}_1$. Suppose there exists a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.

Setup: PKG chooses a random number $s \in \mathbb{Z}_q^*$ and keeps it as the master-key of this system. This master-key is known only to PKG itself. PKG sets $P_{pub} = P^s$ as the system's public key and publishes this public key and other system parameters $params = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, H_1, H_2, F_1, F_2, k_1, k_2\}$.

Here $|q| = k_1 + k_2$. $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \{0,1\}^* \to \mathbb{G}_1^*$, $F_1 : \{0,1\}^{k_2} \to \{0,1\}^{k_1}$ and $F_2 : \{0,1\}^{k_1} \to \{0,1\}^{k_2}$ are four cryptographic hash functions.

Extract: A user submits his/her identity information ID_i to PKG. PKG computes the user's public key as $Q_{ID_i} = H_2(ID_i)$, and returns $S_{ID_i} = Q_{ID_i}^s$ to the user as his/her private key.

Sign: Let the message be $m \in \{0,1\}^{k_2}$.

Each signer randomly selects an element K_i in $\mathbb{G}_1$ and computes $v_i = \hat{e}(K_i, P)$. v_i is broadcast to other signers.

Once each signer's v_i is available through the broadcast channel. They compute

their individual signatures as follows:

$$\begin{aligned} v &= \prod_{i=1}^{n} v_i = \hat{e}(K_1, P)\hat{e}(K_2, P)\cdots\hat{e}(K_n, P) = \hat{e}(\prod_{i=1}^{n} K_i, P); \\ f &= F_1(m)||(F_2(F_1(m)) \oplus m); \\ r &= H_1(v) + f; \\ U_i &= K_i/S_{ID_i}^r. \end{aligned}$$

In the above computation, the symbol $||$ denotes concatenation of two operands.

Each signer transmits its individual signature (v_i, r, U_i) to the clerk who may be one of these signers or other specified trusted principal.

Once the clerk receives an individual signature (v_i, r, U_i), he needs to verify the validity of this individual signature. The verification procedure of the clerk checks that

$$v_i = \hat{e}(U_i, P)\hat{e}(Q_{ID_i}, P_{pub})^r.$$

Once all individual signatures are received and verified by the clerk as valid, the multisignature of the message m with respect to these signers who generate these individual signatures can be generated as (r, U), where

$$U = \prod_{i=1}^{n} U_i = \prod_{i=1}^{n} K_i/S_{ID_i}^r.$$

Verify: Given a multisignature (r, U) and n signers' identities $ID_1, ID_2, \cdots, ID_n$ who stated have signed a message, a verifier computes

$$r - H_1(\hat{e}(U, P)\hat{e}(\prod_{i=1}^{n} Q_{ID_i}, P_{pub})^r) = f,$$

and

$$m = [f]_{k_2} \oplus F_2([f]^{k_1}).$$

In the above computation, the subscript k_2 of f denotes the least significant k_2 bits of f, and the superscript k_1 of f denotes the most significant k_1 bits of f.

The verifier checks whether $[f]^{k_1} = F_1(m)$ holds. If this equation holds, the verifier accepts this multisignature and recovers the original message m from this multisignature. Otherwise, the verifier rejects the multisignature.

3.5 Security Analysis

Theorem 3.1 *This identity-based multisignature with message recovery scheme is correct.*

Proof. The correctness of this identity-based multisignature with message recovery scheme can be shown as follows.

When the individual signature (v_i, r, U_i) is verified,

$$\begin{aligned}
& \hat{e}(U_i, P)\hat{e}(Q_{ID_i}, P_{pub})^r \\
= \ & \hat{e}(K_i/S_{ID_i}^r, P)\hat{e}(Q_{ID_i}, P^s)^r \\
= \ & \hat{e}(K_i/S_{ID_i}^r, P)\hat{e}(Q_{ID_i}^s, P)^r \\
= \ & \hat{e}(K_i/S_{ID_i}^r, P)\hat{e}(S_{ID_i}, P)^r \\
= \ & \hat{e}(K_i/S_{ID_i}^r, P)\hat{e}(S_{ID_i}^r, P) \\
= \ & \hat{e}(K_i, P) \\
= \ & v_i.
\end{aligned}$$

This means if the individual signature (v_i, r, U_i) is indeed generated by signer ID_i, the equation $v_i = \hat{e}(U_i, P)\hat{e}(Q_{ID_i}, P_{pub})^r$ will always hold.

When the multisignature (r, U) is verified, we can recover v which is used by each signer in the multisignature generation from the following computation.

$$\begin{aligned}
& \hat{e}(U, P)\hat{e}(\prod_{i=1}^{n} Q_{ID_i}, P_{pub})^r \\
= \ & \hat{e}(\prod_{i=1}^{n} K_i/\prod_{i=1}^{n} S_{ID_i}^r, P)\hat{e}(\prod_{i=1}^{n} Q_{ID_i}, P^s)^r \\
= \ & \hat{e}(\prod_{i=1}^{n} K_i/\prod_{i=1}^{n} S_{ID_i}^r, P)\hat{e}(\prod_{i=1}^{n} Q_{ID_i}^s, P)^r \\
= \ & \hat{e}(\prod_{i=1}^{n} K_i/\prod_{i=1}^{n} S_{ID_i}^r, P)\hat{e}(\prod_{i=1}^{n} S_{ID_i}, P)^r \\
= \ & \hat{e}(\prod_{i=1}^{n} K_i/\prod_{i=1}^{n} S_{ID_i}^r, P)\hat{e}(\prod_{i=1}^{n} S_{ID_i}^r, P) \\
= \ & \hat{e}(\prod_{i=1}^{n} K_i, P) \\
= \ & v.
\end{aligned}$$

Then, using this v and part of the multisignature r, we can recover f from the following computation.

$$\begin{aligned} & r - H_1(\hat{e}(U,P)\hat{e}(\prod_{i=1}^{n} Q_{ID_i}, P_{pub})^r) \\ = \; & r - H_1(v) \\ = \; & H_1(v) + f - H_1(v) \\ = \; & f. \end{aligned}$$

Since f is computed from $f = F_1(m)||(F_2(F_1(m)) \oplus m)$, we will try to recover the original message m from f like this:

$$\begin{aligned} & [f]_{k_2} \oplus F_2([f]^{k_1}) \\ = \; & [F_1(m)||(F_2(F_1(m)) \oplus m)]_{k_2} \oplus F_2([F_1(m)||(F_2(F_1(m)) \oplus m)]^{k_1}) \\ = \; & F_2(F_1(m)) \oplus m \oplus F_2(F_1(m)) \\ = \; & m. \end{aligned}$$

As previously declared, the subscript k_2 and the superscript k_1 of f denote the least significant k_2 and the most significant k_1 bits of f respectively.

After recovering the alleged original message m, we need to check whether $[f]^{k_1} = F_1(m)$ to verify the validity of the multisignature. If this equation holds, the multisignature (r, U) is valid and the original message m is recovered. Otherwise, the multisignature (r, U) is a forged one. □

Theorem 3.2 *This identity-based multisignature with message recovery scheme is existentially unforgeable under chosen message attacks in the random oracle model, under the assumption that the Computational Diffie-Hellman problem is hard.*

Proof. Assume there is an algorithm $\mathcal{A}$ that can forge a multisignature under chosen message attacks. There will be another algorithm $\mathcal{B}$ that can run the algorithm $\mathcal{A}$ to solve the CDH problem.

In the process of $\mathcal{B}$ using $\mathcal{A}$ to solve the CDH problem, $\mathcal{B}$ needs to simulate all the oracles that $\mathcal{A}$ can query as follows.

Setup: $\mathcal{B}$ sets up $P_{pub} = P^a$ as the system's public key and sends P_{pub} and other system parameters $params = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, H_1, H_2, F_1, F_2, k_1, k_2\}$ to adversary $\mathcal{A}$. In this case, $\mathcal{B}$ only knows the system's public key is P^a, but he does not know

the corresponding master-key s which is actually a in this concrete situation. Two hash functions F_1, F_2 of the four hash functions used in this scheme are published as normal hash functions. The other two hash functions H_1, H_2 are both treated as random oracles.

H_1 Queries: $\mathcal{B}$ creates and keeps two lists of tuples to simulate H_1 Oracle. At the beginning of the simulation, both of these lists are empty.

One list is called Hv_n-List which is used to store tuples like

$$(v_1, v_2, \cdots, v_n, h).$$

In this type of tuples, the first n elements come from group $\mathbb{G}_2$ and the last element comes from $\mathbb{Z}_q^*$.

After receiving an H_1 hash query with respect to several elements $v_1, v_2, \cdots, v_n$ in $\mathbb{G}_2$ and a message m, if the first n elements $v_1, v_2, \cdots, v_n$ are not as a record in the v^*-List which is constructed in the Sign Oracle and not in a record in this Hv_n-List, $\mathcal{B}$ randomly selects $h \in \mathbb{Z}_q^*$ and returns h as the H_1 hash value of $v = \prod_{i=1}^{n} v_i$. Then, $\mathcal{B}$ records the tuple $(v_1, v_2, \cdots, v_n, h)$ in this Hv_n-List. If the first n elements $v_1, v_2, \cdots, v_n$ are already in a record in this Hv_n-List, $\mathcal{B}$ only returns the corresponding h in the record as the H_1 hash value. All in all, this list matches the situation that the honest signer is not required to participate in the multisignature generation.

The other list is called Hv^*-List which is used to store tuples like

$$(m, v_1, v_2, \cdots, v_{n-1}, v^*, y - f).$$

In this type of tuples, the first element m is an arbitrary message to be signed by a subgroup which includes the honest signer. The next n elements come from group $\mathbb{G}_2$ and the last element comes from $\mathbb{Z}_q^*$.

After receiving an H_1 hash query with respect to several elements $v_1, v_2, \cdots, v^*$ in $\mathbb{G}_2$ and a message m, if the first n elements $v_1, v_2, \cdots, v_{n-1}, v^*$ are as a record in the v^*-List which is constructed in the Sign Oracle but not as a record in this Hv^*-List, $\mathcal{B}$ returns $y - f$ as the H_1 hash value of $v = \prod_{i=1}^{n-1} v_i \cdot v^*$ in which y is got from the corresponding record in the v^*-List and f is computed by the equation $f = F_1(m)||(F_2(F_1(m)) \oplus m)$ with respect to the message m. Then, $\mathcal{B}$ records the tuple $(m, v_1, v_2, \cdots, v_{n-1}, v^*, y - f)$ in this Hv^*-List. Note that for the same n elements $v_1, v_2, \cdots, v_{n-1}, v^*$ but different message m, the value y is same because it

comes from the same record in the v^*-List, but the value f is different because it is computed by the equation $f = F_1(m)||(F_2(F_1(m)) \oplus m)$ for different message. So, the returned hash value $y - f$ is different. In this case, we need to add a new record in this Hv^*-List. If these elements $m, v_1, v_2, \cdots, v_{n-1}, v^*$ are already in a record in this Hv^*-List, $\mathcal{B}$ only returns the corresponding $y - f$ in the record as the H_1 hash value. In a word, this list matches the situation that the honest signer is required to participate in the multisignature generation.

H_2 Queries: $\mathcal{B}$ creates and keeps one list H_2-List to simulate H_2 Oracle. At the beginning of the simulation, this list is empty.

For each H_2 hash query with respect to a signer ID_i except for the honest signer ID^*, if ID_i is not in a record in this H_2-List, $\mathcal{B}$ randomly selects $k_i \in \mathbb{Z}_q^*$ and returns $Q_{ID_i} = P^{k_i}$ as the H_2 hash value of ID_i. Then, $\mathcal{B}$ records the tuple (ID_i, k_i, Q_{ID_i}) in this H_2-List. If ID_i is already in a record in this H_2-List, $\mathcal{B}$ only returns the corresponding Q_{ID_i} in the record as the H_2 hash value.

For the H_2 hash query with respect to the honest signer ID^*, $\mathcal{B}$ returns $Q_{ID^*} = P^b$ as the H_2 hash value of ID^*.

Extract Queries: $\mathcal{B}$ creates and keeps one list Ex-List to simulate Extract Oracle. At the beginning of the simulation, this list is empty.

For each Extract query with respect to a signer ID_i except for the honest signer ID^*, if ID_i is not in a record in this Ex-List, $\mathcal{B}$ looks up the H_2-List which is created by H_2 Oracle to find the record about ID_i. Because a signer needs to query H_2 Oracle prior to its any other operation, the Extract Oracle can always find out the record with respect to ID_i in the H_2-List. Using the k_i value in the record in the H_2-List with respect to ID_i, $\mathcal{B}$ returns

$$S_{ID_i} = P_{pub}^{k_i} = P^{ak_i} = P^{k_i a} = Q_{ID_i}^a$$

as the signer ID_i's private key. Then, $\mathcal{B}$ records the tuple (ID_i, S_{ID_i}) in this Ex-List. If ID_i is already in a record in this Ex-List, $\mathcal{B}$ only returns the corresponding S_{ID_i} in the record as the signer ID_i's private key.

Sign Queries: $\mathcal{B}$ creates and keeps two lists of tuples to simulate Sign Oracle. At the beginning of the simulation, both of these lists v_n-List and v^*-List are empty. v_n-List matches the situation that the honest signer is not required to participate in the multisignature generation. v^*-List matches the situation that the honest signer is

required to participate in the multisignature generation. Without loss of generality, we assume that the target signer is always the last signer ID_n.

For each Sign query with respect to an arbitrary message m and a subgroup of n signers $ID_1, ID_2, \cdots, ID_n$, this oracle are divided into two phases.

In the first phase, $n-1$ signers $ID_1, ID_2, \cdots, ID_{n-1}$ generate their individual $v_i = \hat{e}(K_i, P)$ in which K_i is randomly selected from group $\mathbb{G}_1$ and send their v_i and the target signer's identity ID_n to $\mathcal{B}$.

If ID_n is not the honest signer ID^*, $\mathcal{B}$ can randomly select an element K_n from group $\mathbb{G}_1$ and compute $v_n = \hat{e}(K_n, P)$. $\mathcal{B}$ returns v_n to $\mathcal{A}$ and records the tuple

$$(v_1, v_2, \cdots, v_{n-1}, v_n, K_n)$$

in the v_n-List.

If ID_n is the honest signer ID^*, $\mathcal{B}$ can randomly select two integers $x, y \in_R \mathbb{Z}_q^*$. Then $\mathcal{B}$ computes

$$v^* = \hat{e}(P^a, P^b)^y \cdot \hat{e}(P, P)^x = \hat{e}(P^{(yab+x)}, P),$$

and returns this v^* to $\mathcal{A}$. In this case, the corresponding random element from group $\mathbb{G}_1$ is

$$K^* = P^{(yab+x)}.$$

$\mathcal{B}$ records the tuple

$$(v_1, v_2, \cdots, v_{n-1}, v^*, y, x)$$

in the v^*-List.

In the second phase, $\mathcal{A}$ computes $f = F_1(m)||(F_2(F_1(m)) \oplus m)$ with respect to the message m. $\mathcal{A}$ queries H_1 Oracle the H_1 hash value with respect to $(v_1, v_2, \cdots, v_{n-1}, v_n)$ or $(v_1, v_2, \cdots, v_{n-1}, v^*)$ and the message m and uses this H_1 hash value to compute the second part of $n-1$ signers' individual signatures (v_i, r, U_i) as $r = H_1(v) + f$. $\mathcal{A}$ computes the third part

$$U_i = K_i / S_{ID_i}^r = K_i / Q_{ID_i}^{ra}$$

of $n-1$ signers' individual signatures by the real Sign algorithm using the previous r and the corresponding private key $S_{ID_i} = Q_{ID_i}^a$ got from the Extract Oracle and sends these $n-1$ individual signatures and the message m to $\mathcal{B}$.

$\mathcal{B}$ needs to compute $f = F_1(m)||(F_2(F_1(m)) \oplus m)$ at first. If ID_n is not the honest signer ID^*, $\mathcal{B}$ computes the individual signature (v_n, r, U_n) by the real Sign

algorithm using the corresponding r which is computed the same as previous process and S_{ID_n} which is got from the Extract Oracle. Then, $\mathcal{B}$ computes $U = \prod_{i=1}^{n} U_i$ and returns (r, U) as the multisignature on the message m with respect to n signers $ID_1, ID_2, \cdots, ID_n$. In this case, both of the individual signature (v_n, r, U_n) of ID_n and the multisignature (r, U) can pass their own verification process. These verifications can be checked by using the method in Theorem 3.1.

If ID_n is the honest signer ID^*, $\mathcal{B}$ computes r by using H_1 Oracle as

$$r = H_1(v) + f = y - f + f = y,$$

and simulates the third part of the honest signer ID^*'s individual signature as

$$U^* = K^*/S_{ID^*}^r = P^{(yab+x)}/P^{yab} = P^x,$$

in which the corresponding x can be found out in the v^*-List.

Then, $\mathcal{B}$ computes

$$U = \prod_{i=1}^{n-1} U_i \cdot U^*,$$

and returns (r, U) as the multisignature on the message m with respect to n signers $ID_1, ID_2, \cdots, ID_{n-1}, ID^*$.

Verify: Both of the individual signature and the multisignature can pass the verifications. The individual signature (v^*, r, U^*) can pass the verification as follows.

$$\begin{aligned} & \hat{e}(U^*, P)\hat{e}(Q_{ID^*}, P_{pub})^r \\ = \ & \hat{e}(P^x, P)\hat{e}(P^b, P^a)^y \\ = \ & \hat{e}(P^x, P)\hat{e}(P^{yab}, P) \\ = \ & \hat{e}(P^{(yab+x)}, P) \\ = \ & v^*. \end{aligned}$$

The multisignature (r, U) can also pass the verification as follows.

$$\begin{aligned} & \hat{e}(U, P)\hat{e}(\prod_{i=1}^{n} Q_{ID_i}, P_{pub})^r \\ = \ & \hat{e}(\prod_{i=1}^{n-1} U_i \cdot U^*, P)\hat{e}(\prod_{i=1}^{n-1} Q_{ID_i} \cdot Q_{ID^*}, P^a)^y \\ = \ & \hat{e}(\prod_{i=1}^{n-1} U_i, P)\hat{e}(U^*, P)\hat{e}(\prod_{i=1}^{n-1} Q_{ID_i}, P^a)^y\hat{e}(Q_{ID^*}, P^a)^y \end{aligned}$$

$$
\begin{aligned}
&= \hat{e}(\prod_{i=1}^{n-1} U_i, P)\hat{e}(\prod_{i=1}^{n-1} Q_{ID_i}, P^a)^y \hat{e}(U^*, P)\hat{e}(Q_{ID^*}, P^a)^y \\
&= \hat{e}(\prod_{i=1}^{n-1} K_i / \prod_{i=1}^{n-1} Q_{ID_i}^{ya}, P)\hat{e}(\prod_{i=1}^{n-1} Q_{ID_i}^{ya}, P)\hat{e}(P^x, P)\hat{e}(P^{yab}, P) \\
&= \hat{e}(\prod_{i=1}^{n-1} K_i, P)\hat{e}(P^{(yab+x)}, P) \\
&= \prod_{i=1}^{n-1} v_i \cdot v^*.
\end{aligned}
$$

Since we have assumed that adversary $\mathcal{A}$ can forge a multisignature under a chosen message attack, after the simulation process above, $\mathcal{A}$ can output a valid multisignature (r_1, U_1) on the message m with respect to a subgroup of n signers which includes the honest signer ID^* who did not participate in the multisignature generation. There are two restrictions about this multisignature generation. The first one is there is no query to the Extract Oracle with respect to the honest signer ID^*. The second one is there is no query to the Sign Oracle with respect to the message m and a subgroup of signers which includes the honest signer ID^*. $\mathcal{B}$ can compute the third part U_1^* of the honest signer ID^*'s individual signature (v_i^*, r_1^*, U_1^*) from the valid multisignature (r_1, U_1) as follows.

$$U_1^* = U / \prod_{i=1}^{n-1} U_i.$$

All these U_i come from $\mathcal{A}$ in the second phase of the Sign Query.

$\mathcal{B}$ can reset all the oracles and runs $\mathcal{A}$ for the second time. At the end of the simulation, with a non-negligible probability $\mathcal{B}$ can get another different individual signature (v_2^*, r_2^*, U_2^*) on the same message m and with respect to the same honest signer ID^* when v_1^* equals to v_2^*. That means for two different random integer pairs (x_1, y_1) and (x_2, y_2),

$$K_1^* = P^{(y_1ab+x_1)} \text{ is equal to } K_2^* = P^{(y_2ab+x_2)}.$$

Both (r_1^*, U_1^*) and (r_2^*, U_2^*) can pass the verification process, and K_1^* equals K_2^*. So,

$$
\begin{aligned}
\hat{e}(U_1^*, P)\hat{e}(Q_{ID^*}, P_{pub})^{r_1^*} &= \hat{e}(U_2^*, P)\hat{e}(Q_{ID^*}, P_{pub})^{r_2^*}, \\
\hat{e}(U_1^* \cdot S_{ID^*}^{r_1^*}, P) &= \hat{e}(U_2^* \cdot S_{ID^*}^{r_2^*}, P),
\end{aligned}
$$

$$\begin{aligned} U_1^* \cdot S_{ID^*}^{r_1^*} &= U_2^* \cdot S_{ID^*}^{r_2^*}, \\ S_{ID^*}^{r_1^* - r_2^*} &= U_2^* / U_1^*, \\ S_{ID^*} &= (U_2^* / U_1^*)^{(r_1^* - r_2^*)^{-1}}. \end{aligned}$$

In this case, $\mathcal{B}$ can compute the honest signer ID^*'s private key S_{ID^*} when he only knows the honest signer ID^*'s public key Q_{ID^*} and the system's public key P_{pub}. Since S_{ID^*} is expressed as P^{ab}, Q_{ID^*} is expressed as P^b, P_{pub} is expressed as P^a, $\mathcal{B}$ can solve an CDH problem if $\mathcal{A}$ is able to forge valid multisignatures.

If there is no such polynomial-time adversary that can forge a valid multisignature corresponding to a subgroup of signers that include an honest signer, we say that this identity-based multisignature with message recovery scheme is secure against existential forgery under chosen message attack. □

3.6 Summary

We proposed a new notion of identity-based multisignature with message recovery. The notion of identity-based multisignature with message recovery can be viewed as short identity-based multisignature. In order to sign short messages using a scheme that minimizes the total length of the original message and the appended signature, we proposed a concrete identity-based multisignature with message recovery scheme based on bilinear pairing in which multiple signers can generate a single constant size multisignature on the same message regardless the number of signers. There is no need to transmit the original message to the verifier, because the original message can be recovered from the multisignature. We also proved that our scheme is existentially unforgeable against adaptively chosen message attack in the random oracle model, under the hardness assumption of CDH problem.

Compared with the existing schemes, the goal of aggregating the individual signatures to minimize the size of the multisignature is achieved at the expense of extra interactive communication between each signer in the process of generating the individual signatures.

Chapter 4

Identity-Based Authenticated Encryption with Authenticated Header

In this chapter, we present a new notion of *identity-based authenticated encryption with authenticated header*, which has potential applicability to big data. We assume that a big data file naturally comprises two parts: a header and the file (or payload) itself, both are authenticatable while the file is confidential. The header provides a short text describing the content of the file such as title, sender/receiver, content, etc. and can be public. The notion of header is useful in several scenarios, where the information embedded in a file header can be used to determine further actions. That is, the receiver does not have to decrypt the entire file to understand the basic information of the file. In this work, we propose a generic construction of identity-based authenticated encryption with authenticated header by using an identity-based signature with message recovery scheme and a symmetric encryption scheme. The construction is proven to be existentially unforgeable against adaptively chosen message attacks and indistinguishable under adaptive chosen ciphertext attacks, given the underlying identity-based signature with message recovery scheme and the symmetric encryption scheme are secure in the same manner, respectively. We also present a concrete instance of identity-based authenticated encryption with authenticated header scheme based on bilinear pairing.

4.1 Introduction

We consider an interesting scenario, where a party has to handle/receive many (big) files. Usually, these files should be signed and encrypted. In the scenario of a gateway, it is extremely infeasible for the gateway to check the validity and the content of all the received files in particular, since these files are large. Our idea is

to divide a file into a header and the file itself, and hence the payload. Both parts are authenticatable. The authenticated header is accessible by everyone while the file (or payload) itself is encrypted for confidentiality. Therefore, the gateway only needs to check the validity of the authenticated ciphertext and the content of the authenticated header in order to decide how to handle the file.

At first glance, it seems that this scenario could be trivially sorted out by signing a separate header accompanied by an encryption scheme. However, the obvious issue is that one cannot ensure the header is associated with the file. Furthermore, as discussed by Katz and Lindell [KL07], simply combining separated encryption and authentication may lead to an insecure scheme. Notice that signcryption does not capture our goal, i.e. the header can be accessible by everyone. Signcryption has its drawback because in signcryption scheme while the ciphertext is verified, the plaintext has to be also revealed to the verifier, which is contradictory to the confidential aim of the file. There are also publicly verifiable signcryption schemes in which everyone can check the validity of the ciphertext. But they also do not match our goal very well, i.e. the header is accessible by any third party in an authenticated way. Although we can add a label, which is some public data binded to the ciphertext, to the publicly verifiable signcryption in an authenticated way, it is not desirable for some already employed systems to provide additional functions. That is because the publicly verifiable signcryption is specially designed. Namely, the signcryption key and the decryption key in the publicly verifiable signcryption scheme are different. If we want to achieve the authenticated encryption with authenticated header function in an employed system, we would better use the existing public keys and private keys of the users rather than upgrade their keys. Because the upgrading of keys is expensive, which usually means the replacement of existing smart cards or usb devices. Therefore, finding a provably secure and efficient solution that is practical and makes the least changes to the existing systems is hence very desirable.

In this work, we provide a sound solution to the problem by proposing a generic construction using a normal identity-based signature with message recovery scheme and a normal symmetric encryption scheme. In our scheme, the validity and originality of both the header and the payload can be checked by everyone, but the confidentiality of the payload is kept such that only the designated receiver can decrypt the payload. This property is especially desirable in the scenario of retransmission, because the authenticated header can be used to decide how to deal with

the encrypted payload. The computation of encryption and decryption of the file is very efficient due to its symmetric setting. By verifying the validity of the authenticated header and the ciphertext, the integrity and originality of the file encrypted in the ciphertext are also ensured, which is implicitly guaranteed by the sender.

Different from the work of digital signature with message recovery in Chapter 3, which only provides nonrepudiation and integrity/origin protection, the notion of authenticated encryption with authenticated header in this chapter provides extra confidentiality property.

Related Work. The concept of authenticated encryption has been mainly studied in the symmetric setting [BN00, BR00, Kra01, BN08]. It is also has been studied in the public key setting [ADR02, PSST11]. The concept of authenticated encryption with authenticated header in the symmetric setting has been studied by Rogaway [Rog02]. But the counterpart in the asymmetric setting has not been studied in the literature. This work will concentrate on this problem in the identity-based setting.

In 2001, two independent lines of research (Boneh and Franklin [BF01], as well as Cocks [Coc01]) arrived at solutions to the identity-based encryption (IBE). More specifically focusing on joint authentication and encryption, we note an authenticated IBE [Lyn02], and a couple of identity-based signcryption schemes that efficiently combine signature and encryption [ML02, LQ03] have been proposed. In 2002, Lynn [Lyn02] proposed an authenticated IBE scheme. He augmented the system of Boneh and Franklin [BF01] to allow communication with integrity without non-repudiation. In 2003, Boyen [Boy03] proposed a joint identity-based signature/encryption (IBSE) scheme with a common set of parameters and keys. The scheme is very efficient, more secure than the original signcryption model of [Zhe97], and more compact than generic compositions of IBE and IBS. Libert and Quisquater [LQ03] proposed an identity-based signcryption scheme that achieves both public verifiability and resistance to chosen-ciphertext attacks. Chow et al. [CYHC04] improved [LQ03] to achieve public ciphertext authenticity which means any third party should be able to verify the origin of the ciphertext without knowing the content of the message and getting any help from the intended recipient. It is worth noting that by adding a label, which is some public data binded to the ciphertext, to the scheme of [CYHC04], it achieves the goal described above. But this approach is specially designed. Namely, the signcryption key and the decryption key in the publicly verifiable signcryption scheme are different.

Our Contributions. For the first time, this work presents a provably secure (existentially unforgeable against adaptively chosen message attacks and indistinguishable under adaptive chosen ciphertext attacks) generic construction of identity-based authenticated encryption with authenticated header scheme. Different from the previous authenticated encryption schemes in the symmetric setting, in which both the encryption function and the authentication function are symmetric, and those in the asymmetric setting, in which both the encryption function and the authentication function are asymmetric, our scheme is only asymmetric in the authentication function part and symmetric in the encryption function part. We show that only the asymmetric authentication function part is enough to result in an monolithic identity-based authenticated encryption with authenticated header scheme. Thus, compared with those using asymmetric setting in both encryption function and authentication function, our scheme is more efficient. We also present a concrete security analysis to prove the security of the proposed scheme and give a concrete instance of the generic scheme.

Organization of This Chapter. The rest of this chapter is organized as follows: In Section 4.2, we provide a formal definition of the identity-based authenticated encryption with authenticated header scheme. In Section 4.3, we present a security model for the new scheme. In Section 4.4, we present a generic construction of identity-based authenticated encryption with authenticated header scheme by using a normal identity-based signature with message recovery scheme and a symmetric encryption scheme. Section 4.5 provides a concrete instance of the generic identity-based authenticated encryption with authenticated header construction based on bilinear pairing. Security proof for the instance is also provided in this section. Section 4.6 concludes this chapter.

4.2 Formal Definitions

The trusted party Private Key Generator (PKG) generates all users' private keys. Our scheme consists of the following four algorithms.

Setup: On input of a security parameter λ, PKG selects the master secret key of this scheme and publishes public parameters *params*.

Extract: When a party requires its private key corresponding to its identity, this algorithm produces the private key using the master secret key and the identity, then returns the private key to the party.

Authenticated Encryption: On input of a large file and the corresponding header, the sender's private key and the receiver's identity, this algorithm produces an authenticated ciphertext. Both the file and the header are authenticated. The header is public. Only the file is confidential.

Authenticated Decryption: On input of an authenticated ciphertext and the sender's identity, everyone can use this algorithm to check the validity of both the ciphertext and the header with respect to the sender. In order to reveal the corresponding plaintext, the designated receiver's private key is required. So, only the designated receiver can reveal the corresponding plaintext.

4.3 Security Model

Henceforth, we will show security model of our scheme with respect to existential unforgeability against chosen message attacks and indistinguishability against adaptive chosen ciphertext attacks.

Existential unforgeability against chosen message attacks.

It can be defined using a game between an adversary and a challenger.

Since this authenticated encryption with authenticated header scheme only offers unforgeability with respect to the sender, we allow the adversary having access to the receiver's private key, which means even the receiver cannot forge a valid authenticated ciphertext on behalf of the sender. The adversary $\mathcal{A}$ knows the private key of the receiver. Its goal is to forge a valid authenticated ciphertext of a large file, which also implies a successful forgery of the corresponding header on behalf of the specified sender.

Firstly, challenger $\mathcal{C}$ runs the Setup algorithm to get system's master secret key and system's public parameters $params$. Then, $\mathcal{C}$ sends $params$ to $\mathcal{A}$. $\mathcal{A}$ can access to some random oracles. In these random oracles, for every unique query, they respond a random response. If a query has been submitted before, they respond the same value as they responded the first time. $\mathcal{A}$ can also access to the following oracles to conduct an attack.

Extract Oracle: For each Extract query with respect to a user's identity ID, $\mathcal{C}$ returns S_{ID} as the user's private key.

Authenticated Encryption Oracle: For each Authenticated Encryption query on arbitrary large file M with a header M_h, and the designated sender and receiver, $\mathcal{C}$ returns a valid authenticated ciphertext corresponding to the query.

Authenticated Decryption Oracle: For each Authenticated Decryption query on arbitrary alleged authenticated ciphertext, $\mathcal{C}$ checks the validity of this authenticated ciphertext. If the ciphertext is valid, $\mathcal{C}$ returns the corresponding plaintext file M. Otherwise, $\mathcal{C}$ returns $\perp$.

Output: $\mathcal{A}$ outputs an alleged authenticated ciphertext on a large file M^* with a header M_h^* with the specified sender as signer. If no Authenticated Encryption query with respect to the large file M^* and the corresponding header M_h^* with the specified sender as signer has been queried and no Extract query with respect to the specified sender has been queried, $\mathcal{A}$ wins the game if the authenticated ciphertext can be verified as valid.

If there is no such polynomial time adversary $\mathcal{A}$ that can forge a valid authenticated ciphertext, which also implies a successful forgery with respect to the corresponding header as described above, we say that the authenticated encryption with authenticated header scheme is secure against existential forgery under chosen message attacks.

Indistinguishability against adaptive chosen ciphertext attacks.

It can be defined using a game between an adversary and a challenger. The adversary $\mathcal{A}$'s goal is to break the confidentiality of files between a specified sender (e.g. Alice) and a specified receiver (e.g. Bob).

This game composed of a find stage and a guess stage. In the find stage, after querying oracles which will be given below, $\mathcal{A}$ outputs two chosen files M^0, M^1 with equal length. As the authenticated header in our scheme is public, we are only about to prove indistinguishability with respect to the payload. So we should prevent the adversary from differentiating which file is encrypted directly by checking the authenticated header. We should restrict the adversary is only allowed to use identical header although the contents of the chosen files is different. $\mathcal{A}$ sends these two chosen files M^0, M^1 and the corresponding commonly used header M_h to $\mathcal{C}$. $\mathcal{C}$ randomly selects b from 0 and 1, authentically encrypts the file M^b and the corresponding header M_h to generate the challenge authenticated ciphertext. Then, $\mathcal{C}$ sends the challenge authenticated ciphertext to $\mathcal{A}$. In the guess stage, $\mathcal{A}$ can also query these oracles and make a guess b'. $\mathcal{A}$ wins the game if $b' = b$ and the challenge authenticated ciphertext has never been queried to the Authenticated Decryption oracle.

Firstly, challenger $\mathcal{C}$ runs the Setup algorithm to get system's master secret key and system's public parameters *params*. Then, $\mathcal{C}$ sends *params* to $\mathcal{A}$. $\mathcal{A}$ can access

to some random oracles. In these random oracles, for every unique query, they respond a random response. If a query has been submitted before, they respond the same value as they responded the first time. $\mathcal{A}$ can also access to the following oracles to conduct an attack.

Extract Oracle: For each Extract query with respect to a user's identity ID, $\mathcal{C}$ returns S_{ID} as the user's private key.

Authenticated Encryption Oracle: For each Authenticated Encryption query on arbitrary large file M with a header M_h, and the designated sender and receiver, $\mathcal{C}$ returns a valid authenticated ciphertext corresponding to the query.

Authenticated Decryption Oracle: For each Authenticated Decryption query on arbitrary alleged authenticated ciphertext except for the challenge authenticated ciphertext, $\mathcal{C}$ checks the validity of this authenticated ciphertext. If the ciphertext is valid, $\mathcal{C}$ returns the corresponding plaintext file M. Otherwise, $\mathcal{C}$ returns $\perp$.

We say an identity-based authenticated encryption with authenticated header scheme is secure in the sense of indistinguishability, if there is no polynomial time adversary that can win this game.

4.4 Generic Construction

A generic identity-based authenticated encryption with authenticated header scheme comprises a normal identity-based signature with message recovery scheme denoted by $\prod^{IBSMR} = (Setup^{IBSMR}, Extract^{IBSMR}, Sign^{IBSMR}, Verify^{IBSMR})$ and a normal symmetric encryption scheme denoted by $\prod^{SE} = (E^{SE}, D^{SE})$. The generic construction is as follows.

Setup: On input of a security parameter λ, run $Setup^{IBSMR}(\lambda)$ to get the master secret key MK^{IBSMR} and the master public key PK^{IBSMR} of the identity-based signature with message recovery scheme. Set the master secret key MK and the master public key PK of the identity-based authenticated encryption with authenticated header scheme the same as that of the identity-based signature with message recovery scheme, which means $MK = MK^{IBSMR}, PK = PK^{IBSMR}$.

Extract: On input of the master secret key MK and a user's identity ID, run $Extract^{IBSMR}(ID)$ to get the user's private key S_{ID}^{IBSMR} in the identity-based signature with message recovery scheme. Set the users' private keys of the identity-based authenticated encryption with authenticated header scheme the same as that in the identity-based signature with message recovery scheme, which means $S_{ID} =$

S_{ID}^{IBSMR}.

Authenticated Encryption: On input of a message M and the associated header M_h, the sender's private key $S_{ID_{sender}}$, and the receiver's public key $Q_{ID_{receiver}}$ where $Q_{ID_{receiver}}$ is the hash value of the receiver's identity $ID_{receiver}$, select a random item v. Then, generate a random key $K = H(v, e(S_{ID_{sender}}, Q_{ID_{receiver}}))$ for the symmetric encryption scheme. Encrypt the message M as $C = E_K^{SE}(M)$. Use the algorithm $Sign^{IBSMR}$ to authenticate the ciphertext C and the header M_h as $\sigma = Sign^{IBSMR}(S_{ID_{sender}}, v, C, M_h)$. The authenticated ciphertext is σ.

Authenticated Decryption: On input of an authenticated ciphertext σ, the sender's identity, run $Verify^{IBSMR}(\sigma, C, M_h, Q_{ID_{sender}})$, where $Q_{ID_{sender}}$ is the hash value of the sender's identity ID_{sender}, to verify the validity of the authenticate ciphertext σ and recover the random item v. Then, the designated receiver generates the decryption key of the symmetric encryption scheme as $K = H(v, e(Q_{ID_{sender}}, S_{ID_{receiver}}))$. Finally, the receiver recovers the message as $M = D_K^{SE}(C)$.

Theorem 4.1 *If the underlying identity-based signature with message recovery scheme is existentially unforgeable under chosen message attacks, and the underlying symmetric encryption scheme is indistinguishable under adaptive chosen ciphertext attacks, this generic identity-based authenticated encryption with authenticated header scheme is existentially unforgeable under chosen message attacks, and indistinguishable under adaptive chosen ciphertext attacks.*

Proof. We sketch the proofs as follows.

In the security model about existential unforgeability, we allow the adversary to access to the receiver's private key in order to prevent the receiver from forging a valid authenticated ciphertext on behalf of the sender. Thus, the adversary can get access to the symmetric key of the underlying symmetric encryption scheme, since $H(v, e(S_{ID_{sender}}, Q_{ID_{receiver}}))$ is equal to $H(v, e(Q_{ID_{sender}}, S_{ID_{receiver}}))$. In this case, the identity-based authenticated encryption with authenticated header scheme is reduced to the underlying identity-based signature with message recovery scheme. If there is a forger $\mathcal{A}'$ for the identity-based authenticated encryption with authenticated header scheme, we can easily construct a forger $\mathcal{A}$ for the underlying identity-based signature with message recovery scheme.

In terms of the indistinguishability property, the encryption function of the identity-based authenticated encryption with authenticated header scheme is essentially a one-time padding, since in the generation of the symmetric key there is

a random item v. Although this randomness v could be derived by everyone, the other two parameters used in the generation of the symmetric key is only known to the sender and receiver. This will result in an unconditionally secure encryption scheme. □

4.5 An Instance

Hereafter, we propose an instance of the generic construction by using an identity-based partial message recovery signature scheme [ZSM05].

Setup: PKG chooses a random number $s \in \mathbb{Z}_q^*$ as the master secret key of this system where q is a sufficient large prime. PKG sets $P_{pub} = P^s$ as public key. The public parameters are $params = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, P_{pub}, H_1, H_2, H_3, H_4, F_1, F_2, k_1, k_2\}$. Here $|q| = k_1 + k_2$. $H_1 : \mathbb{G}_2 \times \mathbb{G}_2 \times \{0,1\}^{k_2} \to \{0,1\}^l$, $H_2 : \mathbb{G}_2 \times \mathbb{G}_2 \times \{0,1\}^l \to \{0,1\}^l$, $H_3 : \mathbb{G}_2 \times \{0,1\}^l \cdots \times \{0,1\}^l \to \mathbb{Z}_q^*$, $H_4 : \{0,1\}^* \to \mathbb{G}_1^*$, $F_1 : \{0,1\}^{k_2} \to \{0,1\}^{k_1}$ and $F_2 : \{0,1\}^{k_1} \to \{0,1\}^{k_2}$ are six cryptographic hash functions, in which l is the length of a segment of the large file.

Extract: A user submits his/her identity information ID_i to PKG. PKG computes user's public key as $Q_{ID_i} = H_4(ID_i)$, and returns $S_{ID_i} = Q_{ID_i}^s$ to the user as his/her private key.

Authenticated Encryption: Suppose that a user Alice wants to send a large confidential file M to a specified receiver Bob. The large file M is made up of the sequence $\{M_1, M_2, \cdots, M_n\}$, where $M_i \in \{0,1\}^l$ for $i = 1, \cdots, n$. There is also a header M_h which is used to describe M. The header M_h is not confidential. Both the confidential file and the non-confidential header should be authenticated. Alice can carry out the following procedure to generate the authenticated ciphertext blocks of M and the corresponding header M_h. Let the header be $M_h \in \{0,1\}^{k_2}$. Alice does the follows:

- Choose a random number k, and compute $v = \hat{e}(P, P)^k$.
- Compute $w = \hat{e}(S_{ID_{Alice}}, Q_{ID_{Bob}})$.
- Compute $f = F_1(M_h)||(F_2(F_1(M_h)) \oplus M_h)$.
- Compute $c_1 = H_1(v, w, M_h) \oplus M_1$.
- Compute $c_i = H_2(v, w, c_{i-1}) \oplus M_i \quad$ for $i = 2, 3, \cdots, n$.

- Compute $r = H_3(v, c_1, \cdots, c_n) + f$.
- Compute $U = P^k / S^r_{ID_{Alice}}$.

$\{c_1, \cdots, c_n, r, U\}$ is the authenticated ciphertext of M and the corresponding header M_h. In the above computation, the symbol $||$ denotes concatenation of two operands. It is also worth noting that the length of f is equal to $|q|$, so it is able to treated as a member in $\mathbb{Z}^*_q$.

Authenticated Decryption: Upon receiving the authenticated ciphertext $\{c_1, \cdots, c_n, r, U\}$, everyone can perform the following verification procedure to verify the validity of the whole authenticated ciphertext and recover the non-confidential header M_h. One can compute

$v = \hat{e}(U, P) \cdot \hat{e}(Q_{ID_{Alice}}, P_{pub})^r$,

$r - H_3(v, c_1, \cdots, c_n) = f$,

and recover $M_h = [f]_{k_2} \oplus F_2([f]^{k_1})$ from f, and check whether the equation $[f]^{k_1} = F_1(M_h)$ holds. The subscript k_2 of f denotes the least significant k_2 bits of f, and the superscript k_1 of f denotes the most significant k_1 bits of f.

If the previous equation holds, the verifier accepts this authenticated ciphertext as a valid ciphertext and recovers the authenticated header M_h. Otherwise, the verifier rejects the ciphertext.

It is worth noting that the previous verification procedure can be conducted by everyone. Afterwards, only Bob can decrypt and get the corresponding large confidential file M as follows if he wants to:

- $w = \hat{e}(Q_{ID_{Alice}}, S_{ID_{Bob}})$,
- $M_1 = c_1 \oplus H_1(v, w, M_h)$,
- $M_i = c_i \oplus H_2(v, w, c_{i-1})$ for $i = 2, 3, \cdots, n$.

Theorem 4.2 *This identity-based authenticated encryption with authenticated header scheme is correct.*

Proof. The correctness of this scheme can be shown as follows.

Upon receiving an authenticated ciphertext $\{c_1, \cdots, c_n, r, U\}$ with Alice as the sender and Bob as the receiver, we can recover v which is used during the generation of this authenticated ciphertext from the following computation.

$$\hat{e}(U, P) \cdot \hat{e}(Q_{ID_{Alice}}, P_{pub})^r = v.$$

Then, using this v and part of the authenticated ciphertext r, we can recover f from the following computation.

$$r - H_3(v, c_1, \cdots, c_n) = f.$$

Since f is computed from $f = F_1(M_h)||(F_2(F_1(M_h)) \oplus M_h)$, the original header M_h can be recovered from f like this:

$$[f]_{k_2} \oplus F_2([f]^{k_1}) = M_h.$$

For a valid ciphertext, $[f]^{k_1} = F_1(M_h)$ will always hold. After recovering the original authenticated header M_h, Bob can decrypt and get the corresponding large file M by using his private key $S_{ID_{Bob}}$ as follows:

$$\begin{aligned} w &= \hat{e}(Q_{ID_{Alice}}, S_{ID_{Bob}}), \\ M_1 &= c_1 \oplus H_1(v, w, M_h), \\ M_i &= c_i \oplus H_2(v, w, c_{i-1}) \quad \text{for } i = 2, 3, \cdots, n. \end{aligned}$$

□

Theorem 4.3 *This identity-based authenticated encryption with authenticated header scheme is existentially unforgeable under chosen message attacks in the random oracle model, under the assumption that the CDH problem is hard.*

Proof. Assume there is an algorithm $\mathcal{A}$ that can forge a valid authenticated ciphertext under chosen message attacks on behalf of a specified sender (e.g. ID^*). There will be another algorithm $\mathcal{B}$ that can run the algorithm $\mathcal{A}$ as a subroutine to solve the CDH problem.

We assume the instance of CDH problem consists of group elements $(P, P^a, P^b) \in \mathbb{G}_1^3$ for some unknown $a, b \in \mathbb{Z}_q^*$. Our goal is to compute an element $P^{ab} \in \mathbb{G}_1$.

Setup: $\mathcal{B}$ sets $P_{pub} = P^a$ as system's public key and sends system's public parameters *params* to adversary $\mathcal{A}$.

H_1 Queries: $\mathcal{B}$ creates and keeps a list H_1-List to simulate H_1 Oracle. At the beginning of the simulation, this list is empty.

For each H_1 hash query with respect to a tuple (v, w, M_h), if it has not been queried before, $\mathcal{B}$ randomly selects a binary string S which has the same length as that of file segment of the large file M and records the tuple (v, w, M_h, S) in the H_1-List. $\mathcal{B}$ returns S as the H_1 hash value. If the tuple already appears in a record

in the H_1-List, $\mathcal{B}$ only returns the corresponding S in the record as the H_1 hash value.

H_2 Queries: $\mathcal{B}$ creates and keeps a list H_2-List to simulate H_2 Oracle. At the beginning of the simulation, this list is empty.

For each H_2 hash query with respect to a tuple (v, w, c_i), if it has not been queried before, $\mathcal{B}$ randomly selects a binary string R which has the same length as that of file segment of the large file M and records the tuple (v, w, c_i, R) in the H_2-List. $\mathcal{B}$ returns R as the H_2 hash value. If the tuple already appears in a record in the H_2-List, $\mathcal{B}$ only returns the corresponding R in the record as the H_2 hash value.

H_3 Queries: $\mathcal{B}$ creates and keeps two lists of tuples to simulate H_3 Oracle. At the beginning of the simulation, both of these lists are empty.

One list is called Hv-List which is used to store tuples like $(v, c_1, \cdots, c_n, h)$. In this type of tuples, the first element v comes from group $\mathbb{G}_2$, the next n elements $c_1, \cdots, c_n$ belong to $\{0,1\}^l$ and the last element comes from $\mathbb{Z}_q^*$.

Upon receiving an H_3 hash query with respect to a tuple $(v, c_1, \cdots, c_n)$ and a header M_h, if the first element v is not in a record in the v^*-List which is constructed in the Authenticated Encryption Oracle and the tuple $(v, c_1, \cdots, c_n)$ is not in a record in this Hv-List, $\mathcal{B}$ randomly selects $h \in \mathbb{Z}_q^*$ and returns h as the H_3 hash value. Then, $\mathcal{B}$ records the tuple $(v, c_1, \cdots, c_n, h)$ in this Hv-List. If the first $n+1$ elements $(v, c_1, \cdots, c_n)$ are already in a record in this Hv-List, $\mathcal{B}$ only returns the corresponding h in the record as the H_3 hash value. All in all, this list matches the situation that the sender is the specified sender.

The other list called Hv^*-List is used to store tuples like $(M_h, v^*, c_1, \cdots, c_n, y-f)$. In this type of tuples, the first element M_h is an arbitrary header to be authenticated by the sender, the second element v^* comes from group $\mathbb{G}_2$, the next n elements $c_1, \cdots, c_n$ belong to $\{0,1\}^l$ and the last element comes from $\mathbb{Z}_q^*$.

Upon receiving an H_3 hash query with respect to a tuple $(v^*, c_1, \cdots, c_n)$ and a header M_h, if the first element v^* is in a record in the v^*-List which is constructed in the Authenticated Encryption Oracle but the tuple $(v^*, c_1, \cdots, c_n)$ is not in a record in this Hv^*-List, $\mathcal{B}$ returns $(y - f)$ as the H_3 hash value of this tuple, in which y is got from the corresponding record in the v^*-List and f is computed by the equation $f = F_1(M_h)||(F_2(F_1(M_h)) \oplus M_h)$. Then, $\mathcal{B}$ records the tuple $(M_h, v^*, c_1, \cdots, c_n, y-f)$ in this Hv^*-List. Note that for the same element v^* but different header M_h, the value y is same because it comes from the same record in the v^*-List, but the value f is different because it is computed by the equation $f = F_1(M_h)||(F_2(F_1(M_h)) \oplus M_h)$

for different header. So, the returned hash value $(y - f)$ is different. In this case, we need to add a new record in this Hv^*-List. If the tuple $(M_h, v^*, c_1, \cdots, c_n)$ is already in a record in this Hv^*-List, $\mathcal{B}$ only returns the corresponding $(y - f)$ in the record as the H_3 hash value. In a word, this list matches the situation that the sender is the specified sender.

H_4 Queries: $\mathcal{B}$ creates and keeps one list H_4-List to simulate H_4 Oracle. At the beginning of the simulation, this list is empty.

For each H_4 hash query with respect to a user ID_i except for the specified sender, if ID_i has not been queried before, $\mathcal{B}$ randomly selects $d_i \in \mathbb{Z}_q^*$ and returns $Q_{ID_i} = P^{d_i}$ as the hash value of ID_i. Then, $\mathcal{B}$ records the tuple (ID_i, d_i, Q_{ID_i}) in this H_4-List. If ID_i already appears in a record in this H_4-List, $\mathcal{B}$ only returns the corresponding Q_{ID_i} in the record as the H_4 hash value.

For the hash query with respect to the specified sender, $\mathcal{B}$ returns P^b as the hash value.

Extract Queries: $\mathcal{B}$ creates and keeps one list Ex-List to simulate Extract Oracle. At the beginning of the simulation, this list is empty.

For each Extract query with respect to a user ID_i except for the specified sender, if ID_i has not been queried before, $\mathcal{B}$ looks up the H_4-List which is created by H_4 Oracle to find the record about ID_i. Because a user is required to query H_4 Oracle prior to its any other operation, the Extract Oracle can always find out the record with respect to ID_i in the H_4-List. Using the d_i value in the record corresponding to ID_i, $\mathcal{B}$ returns

$$S_{ID_i} = P_{pub}^{d_i} = P^{ad_i} = P^{d_i a} = Q_{ID_i}^a,$$

as the user ID_i's private key. Then, $\mathcal{B}$ records the tuple (ID_i, S_{ID_i}) in this Ex-List. If ID_i already appears in a record in this Ex-List, $\mathcal{B}$ only returns the corresponding S_{ID_i} in the record.

Authenticated Encryption Queries: $\mathcal{B}$ creates and keeps one list v^*-List to simulate Authenticated Encryption Oracle. At the beginning of the simulation, this list is empty.

For each Authenticated Encryption query with respect to an arbitrary file M and the corresponding header M_h and two users ID_1, ID_2 in which ID_1 is the sender and ID_2 is the receiver, if ID_1 is not the specified sender, $\mathcal{B}$ can get the private key S_{ID_1} through the Extract Oracle and generate authenticated ciphertext using the normal authenticated encryption algorithm.

If ID_1 is the specified sender, $\mathcal{B}$ can randomly select two numbers $x, y \in_R \mathbb{Z}_q^*$. Then $\mathcal{B}$ computes

$$v^* = \hat{e}(P^a, P^b)^y \cdot \hat{e}(P, P)^x = \hat{e}(P^{(yab+x)}, P),$$

and $w = \hat{e}(Q_{ID_1}, S_{ID_2})$ in which the Q_{ID_1} is got from the H_4 Oracle and the S_{ID_2} is got from the Extract Oracle. In this case, the corresponding random number is $k^* = (yab + x)$. $\mathcal{B}$ records the tuple (v^*, x, y) in the v^*-List.

Then, $\mathcal{B}$ computes $f^*, c_1^*, \cdots, c_n^*$ as normal, but computes r^* by using H_3 Oracle as

$$r^* = H_3(v^*, c_1^*, \cdots, c_n^*) + f^* = y - f^* + f^* = y,$$

and simulates the last part of the authenticated ciphertext as

$$U^* = P^{k^*}/S_{ID_1}^r = P^{(yab+x)}/P^{yab} = P^x,$$

in which the corresponding x can be found out in the v^*-List.

Then, $\mathcal{B}$ returns $\{c_1^*, \cdots, c_n^*, r^*, U^*\}$ as the authenticated ciphertext on file M and the corresponding header M_h with the specified sender as the sender and user ID_2 as the receiver.

Authenticated Decryption Queries: For an Authenticated Decryption query with respect to an alleged ciphertext $\{c_1, \cdots, c_n, r, U\}$, at least one of the alleged sender and receiver is not the specified sender. $\mathcal{B}$ can get the private key of this user through the Extract Oracle and decrypt the ciphertext using the normal authenticated decryption algorithm.

Verify: While the sender is not the specified sender, the simulated authenticated ciphertext $\{c_1, \cdots, c_n, r, U\}$ will certainly pass the normal verification process, because in this type of simulations we can access to the sender's private key through the Extract Oracle.

While the sender is the specified sender, we will show the simulated authenticated ciphertext $\{c_1^*, \cdots, c_n^*, r^*, U^*\}$ on large file M and the corresponding header M_h can also pass the normal verification process as follows:

Firstly, we can try to recover the header M_h:

$$\hat{e}(U^*, P) \cdot \hat{e}(Q_{ID^*}, P_{pub})^{r^*} = v^*,$$

$$r^* - H_3(v^*, c_1^*, \cdots, c_n^*) = y - (y - f^*) = f^*,$$

$$M_h = [f^*]_{k_2} \oplus F_2([f^*]^{k_1}).$$

We can check that the equation $[f^*]^{k_1} = F_1(M_h)$ holds, then compute $w = \hat{e}(Q_{ID^*}, S_{ID_2})$ and recover the large file M:

$$\begin{aligned} M_1 &= c_1^* \oplus H_1(v^*, w, M_h), \\ M_i &= c_i^* \oplus H_2(v^*, w, c_{i-1}^*) \quad \text{for } i = 2, 3, \cdots, n. \end{aligned}$$

Since we have assumed that adversary $\mathcal{A}$ can forge a valid authenticated ciphertext under chosen message attacks, after the simulation process above, $\mathcal{A}$ can output a valid authenticated ciphertext $\{c_1^*, \cdots, c_n^*, r^*, U^*\}$ on a large file M^* and the corresponding header ${M_h}^*$ with the specified sender as sender and user ID_2 as receiver. There is one restriction that there should be no query to Authenticated Encryption Oracle with respect to the large file M^* and the corresponding header ${M_h}^*$ with the specified sender as sender.

$\mathcal{B}$ can reset all the oracles and runs $\mathcal{A}$ for the second time. At the end of another simulation, with a non-negligible probability $\mathcal{B}$ can forge another different valid authenticated ciphertext $\{c_1^{*\prime}, \cdots, c_n^{*\prime}, r^{*\prime}, U^{*\prime}\}$ with the same specified sender as sender and user ID_2 as receiver when $v^* = v^{*\prime}$. That means for two different random integer pairs (x^*, y^*) and $(x^{*\prime}, y^{*\prime})$,

$$k^* = y^* ab + x^* \text{ is equal to } k^{*\prime} = y^{*\prime} ab + x^{*\prime}.$$

Both $\{c_1^*, \cdots, c_n^*, r^*, U^*\}$ and $\{c_1^{*\prime}, \cdots, c_n^{*\prime}, r^{*\prime}, U^{*\prime}\}$ can pass the verification process, and k^* equals $k^{*\prime}$. So,

$$\begin{aligned} \hat{e}(U^*, P) \cdot \hat{e}(Q_{ID^*}, P_{pub})^{r^*} &= \hat{e}(U^{*\prime}, P) \cdot \hat{e}(Q_{ID^*}, P_{pub})^{r^{*\prime}}, \\ \hat{e}(U^* \cdot S_{ID^*}^{r^*}, P) &= \hat{e}(U^{*\prime} \cdot S_{ID^*}^{r^{*\prime}}, P), \\ U^* \cdot S_{ID^*}^{r^*} &= U^{*\prime} \cdot S_{ID^*}^{r^{*\prime}}, \\ S_{ID^*}^{r^* - r^{*\prime}} &= U^{*\prime} / U^*, \\ S_{ID^*} &= (U^{*\prime} / U^*)^{(r^* - r^{*\prime})^{-1}}. \end{aligned}$$

In this case, $\mathcal{B}$ can compute the specified sender ID^*'s private key S_{ID^*} when he only knows ID^*'s public key Q_{ID^*} and system's public key P_{pub}. Since S_{ID^*} is expressed as P^{ab}, Q_{ID^*} is expressed as P^b, P_{pub} is expressed as P^a, $\mathcal{B}$ can solve the CDH problem if $\mathcal{A}$ is able to forge valid ciphertexts.

If there is no such polynomial-time adversary that can forge a valid authenticated ciphertext on behalf of the specified sender, we say that this identity-based authenticated encryption with authenticated header scheme is secure against existential forgery under chosen message attacks. □

Theorem 4.4 *This identity-based authenticated encryption with authenticated header scheme is indistinguishable under adaptive chosen ciphertext attacks in the random oracle model, under the assumption that the BDH problem is hard.*

Proof. Assume there is an algorithm $\mathcal{A}$ that can win the second game mentioned in the security model. There will be another algorithm $\mathcal{B}$ that can run the algorithm $\mathcal{A}$ as a subroutine to solve the BDH problem.

We assume that the instance of BDH problem consists of group elements $(P, P^s, P^a, P^b) \in \mathbb{G}_1^4$ for some unknown $s, a, b \in \mathbb{Z}_q^*$, and our goal is to compute an element $\hat{e}(P,P)^{sab} \in \mathbb{G}_2$.

Setup: $\mathcal{B}$ sets $P_{pub} = P^s$ as system's public key and sends system's public parameters *params* to adversary $\mathcal{A}$. In this case, $\mathcal{B}$ only knows the system's public key is P^s, but he does not know the corresponding master secret key which is actually s in this situation.

H_1 Queries: $\mathcal{B}$ simulates H_1 Oracle the same as that in the proof of Theorem 4.3.

H_2 Queries: $\mathcal{B}$ simulates H_2 Oracle the same as that in the proof of Theorem 4.3.

H_3 Queries: $\mathcal{B}$ creates and keeps two lists of tuples to simulate H_3 Oracle. At the beginning of the simulation, both of these lists are empty.

One list is called Hv-List which is used to store tuples like $(v, c_1, \cdots, c_n, h)$. In this type of tuples, the first element v comes from group $\mathbb{G}_2$, the next n elements $c_1, \cdots, c_n$ belong to $\{0,1\}^l$ and the last element comes from $\mathbb{Z}_q^*$.

Upon receiving an H_3 hash query with respect to a tuple $(v, c_1, \cdots, c_n)$ and a header M_h, if the first element v is not in a record in the v^*-List which is constructed in the Authenticated Encryption Oracle and the tuple $(v, c_1, \cdots, c_n)$ is not in a record in this Hv-List, $\mathcal{B}$ randomly selects $h \in \mathbb{Z}_q^*$ and returns h as the H_3 hash value. Then, $\mathcal{B}$ records the tuple $(v, c_1, \cdots, c_n, h)$ in this Hv-List. If the first $n+1$ elements $(v, c_1, \cdots, c_n)$ are already in a record in this Hv-List, $\mathcal{B}$ only returns the corresponding h in the record as the H_3 hash value. All in all, this list matches the situation that the sender is neither the specified sender nor the specified receiver.

The other list called Hv^*-List is used to store tuples like $(M_h, v^*, c_1, \cdots, c_n, y-f)$. In this type of tuples, the first element M_h is an arbitrary header to be authenticated by the sender, the second element v^* comes from group $\mathbb{G}_2$, the next n elements $c_1, \cdots, c_n$ belong to $\{0,1\}^l$ and the last element comes from $\mathbb{Z}_q^*$.

Upon receiving an H_3 hash query with respect to a tuple $(v^*, c_1, \cdots, c_n)$ and a header M_h, if the first element v^* is in a record in the v^*-List which is constructed in the Authenticated Encryption Oracle but the tuple $(v^*, c_1, \cdots, c_n)$ is not in a record

in this Hv^*-List, $\mathcal{B}$ returns $(y - f)$ as the H_3 hash value of this tuple, in which y is got from the corresponding record in the v^*-List and f is computed by the equation $f = F_1(M_h)||(F_2(F_1(M_h))\oplus M_h)$. Then, $\mathcal{B}$ records the tuple $(M_h, v^*, c_1, \cdots, c_n, y-f)$ in this Hv^*-List. Note that for the same element v^* but different header M_h, the value y is same because it comes from the same record in the v^*-List, but the value f is different because it is computed by the equation $f = F_1(M_h)||(F_2(F_1(M_h)) \oplus M_h)$ for different header. So, the returned hash value $(y - f)$ is different. In this case, we need to add a new record in this Hv^*-List. If the tuple $(M_h, v^*, c_1, \cdots, c_n)$ is already in a record in this Hv^*-List, $\mathcal{B}$ only returns the corresponding $(y - f)$ in the record as the H_3 hash value. In a word, this list matches the situation that the sender is either the specified sender or the specified receiver.

H_4 Queries: $\mathcal{B}$ creates and keeps one list H_4-List to simulate H_4 Oracle. At the beginning of the simulation, this list is empty.

For each H_4 hash query with respect to a user ID_i except for the specified sender and the specified receiver, if ID_i has not been queried before, $\mathcal{B}$ randomly selects $d_i \in \mathbb{Z}_q^*$ and returns $Q_{ID_i} = P^{d_i}$ as the H_4 hash value of ID_i. Then, $\mathcal{B}$ records the tuple (ID_i, d_i, Q_{ID_i}) in this H_4-List. If ID_i already appears in a record in this H_4-List, $\mathcal{B}$ only returns the corresponding Q_{ID_i} in the record as the H_4 hash value.

For the hash query with respect to the specified sender Alice, $\mathcal{B}$ returns $Q_{ID_{Alice}} = P^a$ as the hash value. For the hash query with respect to the specified receiver Bob, $\mathcal{B}$ returns $Q_{ID_{Bob}} = P^b$ as the hash value.

Extract Queries: $\mathcal{B}$ creates and keeps one list Ex-List to simulate Extract Oracle. At the beginning of the simulation, this list is empty.

For each Extract query with respect to a user ID_i except for the specified sender and the specified receiver, if ID_i has not been queried before, $\mathcal{B}$ looks up the H_4-List which is created by H_4 Oracle to find the record about ID_i. Because a user is required to query H_4 Oracle prior to its any other operation, the Extract Oracle can always find out the record with respect to ID_i in the H_4-List. Using the d_i value in the record corresponding to ID_i, $\mathcal{B}$ returns

$$S_{ID_i} = P_{pub}^{d_i} = P^{sd_i} = P^{d_i s} = Q_{ID_i}^s,$$

as the user ID_i's private key. Then, $\mathcal{B}$ records the tuple (ID_i, S_{ID_i}) in this Ex-List. If ID_i already appears in a record in this Ex-List, $\mathcal{B}$ only returns the corresponding S_{ID_i} in the record.

Authenticated Encryption Queries: $\mathcal{B}$ creates and keeps one list v-List to simulate Authenticated Encryption Oracle. At the beginning of the simulation, this list is empty.

For each Authenticated Encryption query with respect to an arbitrary file M and the corresponding header M_h and two users ID_1, ID_2 in which ID_1 is the sender and ID_2 is the receiver, if ID_1 is neither the specified sender nor the specified receiver, $\mathcal{B}$ can get the private key S_{ID_1} through the Extract Oracle and generate authenticated ciphertext using the normal authenticated encryption algorithm.

If ID_1 is the specified sender and ID_2 is not the specified receiver, $\mathcal{B}$ can randomly select two numbers $x, y \in_R \mathbb{Z}_q^*$. Then $\mathcal{B}$ computes

$$v = \hat{e}(P^a, P^s)^y \cdot \hat{e}(P, P)^x = \hat{e}(P^{(yas+x)}, P),$$

and $w = \hat{e}(Q_{ID_1}, S_{ID_2})$ in which the Q_{ID_1} is got from the H_4 Oracle and the S_{ID_2} is got from the Extract Oracle. In this case, the corresponding random number is $k = (yas + x)$. $\mathcal{B}$ records the tuple (v, x, y) in the v-List.

Then, $\mathcal{B}$ computes $f, c_1, \cdots, c_n$ as normal, but computes r by using H_3 Oracle as

$$r = H_3(v, c_1, \cdots, c_n) + f = y - f + f = y,$$

and simulates the last part of the authenticated ciphertext as

$$U = P^k / {S_{ID_1}}^r = P^{(yas+x)} / P^{yas} = P^x,$$

in which the corresponding x can be found out in the v-List.

Then, $\mathcal{B}$ returns $\{c_1, \cdots, c_n, r, U\}$ as the authenticated ciphertext on file M and the corresponding header M_h with the specified sender as sender and user ID_2 as receiver.

If ID_1 is the specified receiver and ID_2 is not the specified sender, this situation is very similar with that of ID_1 is the specified sender and ID_2 is not the specified receiver. $\mathcal{B}$ can simulate this situation as above just by replacing the role of ID_1 from the specified sender to the specified receiver.

If ID_1 is the specified sender and ID_2 is the specified receiver, $\mathcal{B}$ randomly selects two numbers $x, y \in_R \mathbb{Z}_q^*$. Then $\mathcal{B}$ computes

$$v = \hat{e}(P^a, P^s)^y \cdot \hat{e}(P, P)^x = \hat{e}(P^{(yas+x)}, P),$$

and randomly selects $w \in \mathbb{G}_2$. In this case, the corresponding random number is $k = (yas + x)$. $\mathcal{B}$ records the tuple (v, x, y) in the v-List.

Then, $\mathcal{B}$ computes $f = F_1(M_h)||(F_2(F_1(M_h)) \oplus M_h)$ at first, and randomly selects $c_1, \cdots, c_n \in \{0,1\}^l$.

$\mathcal{B}$ computes r by using H_3 Oracle as

$$r = H_3(v, c_1, \cdots, c_n) + f = y - f + f = y,$$

and simulates the last part of the challenge authenticated ciphertext as

$$U = P^k / {S_{ID_{Alice}}}^r = P^{(yas+x)} / P^{ysa} = P^x.$$

Then, $\mathcal{B}$ returns $\{c_1, \cdots, c_n, r, U\}$ as the authenticated ciphertext on file M and the corresponding header M_h with the specified sender as sender and the specified receiver as receiver.

If ID_1 is the specified receiver and ID_2 is the specified sender, this situation is very similar with that of ID_1 is the specified sender and ID_2 is the specified receiver. $\mathcal{B}$ can simulate this situation as above just by exchanging the role of ID_1 and ID_2.

Challenge Authenticated Ciphertext: Upon receipt of two chosen plaintext files M^0, M^1 with equal length and the corresponding commonly used header M_h, $\mathcal{B}$ chooses $b \in \{0,1\}$ at random, sets the specified sender as the sender and the specified receiver as the receiver. Then $\mathcal{B}$ runs the Authenticated Encryption oracle on file M^b and the corresponding header M_h to generate the challenge authenticated ciphertext $\{c_1^*, \cdots, c_n^*, r^*, U^*\}$ and returns it to the adversary.

Authenticated Decryption Queries: For an Authenticated Decryption query with respect to an alleged ciphertext $\{c_1, \cdots, c_n, r, U\}$, if not both of the alleged sender and receiver are the specified sender and the specified receiver respectively, $\mathcal{B}$ can get the user's private key who is not the specified one through the Extract Oracle and decrypt the ciphertext using normal decryption method.

If both of the sender and the receiver are the specified one, i.e. with Alice as the sender and Bob as the receiver, $\mathcal{B}$ computes

$$v = \hat{e}(U, P) \cdot \hat{e}(Q_{ID_{Alice}}, P_{pub})^r,$$

and

$$r - H_3(v, c_1, \cdots, c_n) = f,$$

and recovers $M_h = [f]_{k_2} \oplus F_2([f]^{k_1})$ from f.

Then, $\mathcal{B}$ checks whether the authenticated ciphertext is valid by checking whether the equation $[f]^{k_1} = F_1(M_h)$ holds.

If the authenticated ciphertext is valid, $\mathcal{B}$ checks whether there exists a tuple (v_i, w_i, M_{hi}, S_i) in the H_1-List such that

$$v = v_i \text{ and } M_h = M_{hi}.$$

If this tuple exists, $\mathcal{B}$ can reveal the first part M_1 of the large file M as

$$M_1 = c_1 \oplus S_i,$$

in which S_i is the H_1 hash value corresponding to the tuple (v_i, w_i, M_{hi}). $\mathcal{B}$ can continue to check whether there exists a tuple (v_j, w_j, c_j, R_j) in the H_2-List such that $c_j = c_1$ and $v_j = v_i$. If this tuple exists, $\mathcal{B}$ can continue to recover the second part M_2 of the large file M as $M_2 = c_2 \oplus R_j$ in which R_j is the H_2 hash value corresponding to the tuple (v_j, w_j, c_j). $\mathcal{B}$ can use the same method to recover the rest parts of the whole file M. At the end, $\mathcal{B}$ returns M as the plaintext of the queried authenticated ciphertext to the adversary $\mathcal{A}$.

At the end, $\mathcal{A}$ makes a guess b'. If $\mathcal{A}$ has some non-negligible advantages to win the game mentioned above, $\mathcal{B}$ can check whether there exists a record containing v^* in H_1-List or H_2-List, where v^* is computed from the challenge ciphertext. If the record exists, $\mathcal{B}$ can have the same advantage to output the corresponding w in the record as the value of $\hat{e}(P, P)^{sab}$. □

4.6 Summary

We introduced a new notion of identity-based authenticated encryption with authenticated header. This notion has potential applications in the environment with big data. Using this cryptographic primitive, everyone can check the validity of the authenticated ciphertext and access to the authenticated header, but only the designated receiver can recover the file. Additionally, the designated receiver is given the liberty whether to decrypt the file or not by only checking the corresponding authenticated header. We gave a generic construction and proved that our scheme is secure against existential forgery under adaptively chosen message attacks and indistinguishable under adaptive chosen ciphertext attacks. We also gave a concrete instance based on bilinear pairing, and gave concrete security analysis of the instance. The instance addressed a problem which was not considered in the symmetric key setting counterpart.

Chapter 5

Identity-Based Quotable Ring Signature

In this chapter, we present a new notion of *identity-based quotable ring signature*. This new cryptographic primitive can be used to derive new ring signatures on substrings of an original message from an original ring signature on the original message, which is generated by an actual signer included in the ring. No matter whether a ring signature is originally generated or is quoted from another valid ring signature, it will convince the verifier that it is generated by one of the ring members, without revealing any information about which ring member is the actual signer. The set of signers could be arbitrarily selected by the actual signer without need of other signers' approval. The actual signer is anonymous among this set of signers. At the same time, the verifier could not distinguish whether a ring signature is originally generated or is quoted from another ring signature. In this work, we propose a concrete identity-based quotable ring signature scheme based on bilinear pairing. We make use of bilinear groups of composite order. The construction is identity-based to alleviate the problem of certificate verification, especially for applications involving a large number of public keys in each execution such as ring signature schemes. The proposed scheme is proven to be anonymous under the assumption that the subgroup decision problem is hard, selectively unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the Computational Diffie-Hellman problem is hard, and strongly context hiding.

5.1 Introduction

In the ring signature schemes, the actual signer can choose arbitrary other signers to form a ring that includes himself. The actual signer anonymously signs messages by using his private key and other users' public keys on behalf of the whole ring. There is no requirement to get other users' approval. On one hand, similar with the

group signature schemes, the verifier must be convinced that a signature has been generated by a member of this ring, but could not have a better way to identify the actual signer than at random to guess which member is the actual signer. The actual signer remains completely anonymous. On the other hand, unlike the group signature schemes, there is no group manager, no setup procedure, no revocation procedure, and no coordination in traditional ring signature. There is no way to revoke the anonymity of the actual signer. Ring signature schemes can be considered as simplified group signature schemes which consist of only users without managers. Recently, in order to realize an efficient ring signature scheme provably secure in the standard model, Shacham and Waters [SW07] introduced an efficient ring signature scheme by allowing for a trusted global setup step by an authority.

In the ring signature schemes, all ring members' information serves as a part of the ring signature. In traditional Public Key Infrastructure, prior to the generation of a ring signature, the actual signer has to check the validity of public keys of other users which are included in the ring. Similarly, prior to the verification of a ring signature, the verifier has to check the validity of public keys of all the users in the ring. This increases both the generation and the verification cost of ring signatures. In the identity-based setting, introduced by Shamir, it avoids these checks about public keys. In this way, the public keys of users can be easily and publicly computed from their identities by anyone. This is especially desirable for applications which involve a large number of public key checks such as ring signature schemes.

Quoting is usually applied to derive a signature on a substring when text messages are signed. It can also be applied to derive a signature on a subregion of an image when images are signed, such as a face. In the quotable signature schemes, for every substring m' of a message m, it is possible for a third party to derive a signature on m' from a signature on m on behalf of the same signer. Moreover, the derived signature on m' reveals no extra information about m, which means the derived signature cannot be distinguished from a fresh one even when the original signature on m is given. The inability to link derived signatures to their original sources prevents some practical privacy and linking attacks. It is desirable to allow repeated computation on the signatures, which means it is possible to quote from a quoted signature. It is also desirable that the size of the signatures depends only on the size of the object being signed, no matter whether the signature is fresh or derived, even if being quoted several times. This means the signature size will not

grow with every derivation.

Related Work. In the identity-based setting, user's public key could be easily and publicly computed from his identity. Digital certificates are not needed.

The concept of ring signature scheme was formalized by Rivest, Shamir, and Tauman [RST01, RST06]. They proposed a scheme based on certificate-based public key setting, which is proved existentially unforgeable under adaptive chosen-message attacks assuming the hardness of the RSA problem. Before the concept of ring signature scheme is formalized, it is used as a tool to construct group signature schemes in [CvH91, Cam97]. There are two main differences between the concepts of ring signature schemes and group signature schemes. First, the ring is determined by the actual signer and is dynamic, while the group members are controlled by the manager and are fixed at any given time. Second, no one can identify the actual signer in ring signature schemes, while the group manager can identify the actual signer in group signature schemes. Bresson, Stern, and Szydlo [BSS02] gave a simpler proof of the security of the scheme in [RST01], under the strictly weaker assumption of the random oracle model. Abe, Ohkubo, and Suzuki [AOS02] proposed some general constructions of ring signature schemes, where the public keys of the users can be totally independent. Their scheme is also based on the certificate-based public key setting. Herranz and Sáez [HS03a] gave some security results for generic ring signature schemes, and they designed a new specific scheme based on Schnorr's signature scheme.

Shacham and Waters [SW07] described the first efficient ring signature scheme secure without random oracles, based on standard assumptions. Their scheme is related to a group signature scheme secure without random oracles due to Boyen and Waters [BW06]. The main difference is that in [BW06] the master public key is public and the first level message is encrypted, while in [SW07] the signer's public key is encrypted and the message is public.

The first identity-based ring signature scheme was proposed by Zhang and Kim [ZK02] based on pairings. But they did not provide a formal proof of the existential unforgeability of their scheme. Herranz [Her04] proposed such a proof of [ZK02]. Later, Lin and Wu [LW04] proposed a more efficient identity-based ring signature scheme. Tang, Liu, and Wang [TLW03] pointed out some mistakes in [LW04] and proposed an improved scheme. Herranz and Sáez [HS03b] extended their work [HS03a] on ring forking lemmas to the identity-based scenario.

Ahn et al. [ABC$^+$12] proposed an efficient quotable signature scheme, which

equipped with strongly context hiding and selectively unforgeability property. Early work regarding anyone deriving quoted signatures such as redactable signature schemes [JMSW02, CLX09, BBD+10, BF11a, BF11b] supports quoting from a single document, but does not achieve the privacy or unforgeability properties required in [ABC+12]. The work whose definition is closest to [ABC+12] is that on redacted signatures of Chang, Lim, and Xu [CLX09], and Brzuska et al. [BBD+10], and Boneh, and Freeman [BF11a, BF11b]. However, in [ABC+12], a quoted signature should be indistinguishable from a fresh signature, even when the distinguisher is given the original signature. In contrast, the definitions of [CLX09, BBD+10, BF11a, BF11b] do not provide the distinguisher with the original signature. Thus, it may be possible to link a quoted document to its original source, which can have negative privacy implications.

Another type of studies computing on authenticated data require secret information of the original signer, such as sanitizable signatures [ACdMT05, BFF+09, BFLS10], and incremental signatures [BGG94], where the signer can efficiently make small edits to his signed data. We also consider this type of computation on authenticated data in Chapter 7. In contrast, our work in this chapter followed [ABC+12] concentrate more about anyone can compute on the authenticated data.

Our Contributions. For the first time, this work presents a provably secure (correct, anonymous, selectively unforgeable, and strongly context hiding) identity-based quotable ring signature scheme based on bilinear pairing, under the Subgroup Decision Problem assumption and Computational Diffie-Hellman assumption in the composite order groups. We also present a security model and concrete security analysis by the reduction to prove the security of the proposed scheme. More precisely, we can show that if there exists an attacker who can identify the actual signer among a ring of signers, then the Subgroup Decision Problem is solved, and if there exists an attacker who can selectively forge a valid quotable ring signature, then the Computational Diffie-Hellman problem is solved. We also prove the scheme is strongly context hiding in a statistical definition.

Organization of This Chapter. The rest of this chapter is organized as follows: In Section 5.2, we recall some known results about homomorphic encryption and NIZK, which are used as building blocks in the proposed scheme. In Section 5.3, we provide a formal definition of the identity-based quotable ring signature scheme. In Section 5.4, we present a security model for the new scheme. In Section 5.5, we present a concrete identity-based quotable ring signature scheme based on bilinear

pairing in the composite order groups. Section 5.6 provides the security proof about correctness, anonymity, selectively unforgeability against adaptively chosen message attacks and strongly context hiding property. Section 5.7 concludes this chapter.

5.2 NIZK Proof That Ciphertext C Encrypts 0 or 1

We use some cryptographic primitives as building blocks in this scheme. The first one is a homomorphic public key encryption scheme, which is proposed by Boneh, Goh and Nissim [BGN05]. The homomorphic encryption scheme consists of three algorithms, which are as follows.

KeyGen: Given a security parameter λ, run $\mathcal{G}(1^\lambda)$ to obtain a tuple $(p, q, \mathbb{G}, \mathbb{G}_T, e)$. Let $n = pq$, where p, q are sufficient large prime numbers. Both $\mathbb{G}, \mathbb{G}_T$ are cyclic groups of composite order n. Select g as a random generator of $\mathbb{G}$ and h as a random generator of $\mathbb{G}_q$, which is a cyclic order q subgroup of $\mathbb{G}$. The public key is $(n, \mathbb{G}, \mathbb{G}_T, e, g, h)$. The private key is q.

Encrypt: To encrypt a message m, pick a random $s \leftarrow \{0, 1, \cdots, n-1\}$ and compute $C = g^m h^s \in \mathbb{G}$. Output C as the ciphertext.

Decrypt: To decrypt a ciphertext C, compute $C^q = g^{mq} h^{sq} = (g^q)^m$. Let $\widehat{g} = g^q$ and exhaustively search for m.

Note that decryption in this system takes polynomial time in the size of the message space. Therefore, the system can only be used to encrypt short messages.

The second building block is a non-interactive zero-knowledge proof which is proposed by Groth, Ostrovsky and Sahai [GOS06]. It proves that a BGN-ciphertext has either 0 or 1 as plaintext. The non-interactive zero-knowledge proof is described as follows.

Common reference string: Given a security parameter λ, run $\mathcal{G}(1^\lambda)$ to obtain a tuple $(p, q, \mathbb{G}, \mathbb{G}_T, e)$. Let $n = pq$, where p, q are sufficient large prime numbers. Both $\mathbb{G}, \mathbb{G}_T$ are cyclic groups of composite order n. Select g as a random generator of $\mathbb{G}$ and h as a random generator of $\mathbb{G}_q$, which is a cyclic order q subgroup of $\mathbb{G}$. The common reference string is $\sigma = (n, \mathbb{G}, \mathbb{G}_T, e, g, h)$.

Statement: The statement is an element $C \in \mathbb{G}$. The claim is that there exists a pair $(m, s) \in \mathbb{Z}^2$ such that $m \in \{0, 1\}$ and $C = g^m h^s$.

Proof: Input $(\sigma, C, (m, s))$.

- Check $m \in \{0,1\}$ and $C = g^m h^s$. Return failure if check fails.
- $r \leftarrow \mathbb{Z}_n^*$.
- $\pi_1 = h^r, \quad \pi_2 = (g^{2m-1}h^s)^{sr^{-1}}, \quad \pi_3 = g^r$.
- Return $\pi = (\pi_1, \pi_2, \pi_3)$.

Verification: Input $(\sigma, C, \pi = (\pi_1, \pi_2, \pi_3))$.

- Check $C \in \mathbb{G}$ and $\pi \in \mathbb{G}^3$.
- Check $e(C, Cg^{-1}) \stackrel{?}{=} e(\pi_1, \pi_2)$ and $e(\pi_1, g) \stackrel{?}{=} e(h, \pi_3)$.
- Return 1 if both checks pass, else return 0.

In order to make these building blocks suitable for the quotable signature scheme, we should slightly change the random number s used in the Encrypt algorithm of Boneh, Goh, and Nissim encryption scheme [BGN05] to $s \leftarrow \mathbb{Z}_n^*$.

The Proof algorithm of Groth, Ostrovsky, and Sahai NIZK proof scheme [GOS06] should also be slightly changed as follows.

Proof: Input $(\sigma, C, (m, s))$.

- Check $m \in \{0,1\}$ and $C = g^m h^s$. Return failure if check fails.
- $\widehat{r} \leftarrow \mathbb{Z}_n^*, \quad r = \widehat{r}s$.
- $\pi_1 = h^r, \quad \pi_2 = (g^{2m-1}h^s)^{sr^{-1}}, \quad \pi_3 = g^r$.
- Return $\pi = (\pi_1, \pi_2, \pi_3)$.

In fact, as $\widehat{r}$ is a random number, after multiplied to s, r is still a random number. The modified proof algorithm is essentially the same as the original one. But the elements π_1, π_2, and π_3 have been changed to $\pi_1 = h^{sr}, \pi_2 = (g^{2m-1}h^s)^{r^{-1}}$, and $\pi_3 = g^{sr}$, respectively.

5.3 Formal Definitions

There will be two types of different signatures named Type I signature and Type II signature, where Type I signature can be quoted down to another Type I or Type II signature, Type II signature cannot be quoted again but with a shorter

signature size. The trusted party Private Key Generator (PKG) generates all users' private keys. An identity-based quotable ring signature scheme for message space $\mathcal{M}$ consists of the following five algorithms.

Setup: On input of a security parameter λ, PKG selects the master secret key of this scheme and publishes public parameters *params*.

Extract: When a party requires its private key corresponding to its identity, this algorithm generates the private key by using the master secret key and the identity, then returns the private key to the party.

Sign: This algorithm takes as input the actual signer's private key S_{ID} and a set of public keys R that constitutes the ring, along with a message M in the message space to be signed. It is required that $Q_{ID} \in R$ holds. This algorithm returns a ring signature σ on M on behalf of the ring R.

Quote: This algorithm takes as input a Type I ring signature σ with the corresponding ring R and message M, and a substring M' of M. It first checks the validity of σ with respect to R and M. If σ is valid, it produces a new ring signature σ', which is either Type I or Type II, of M' on behalf of the ring R. Otherwise, it outputs a special symbol $\perp$ to represent failure.

Verify: The verification algorithm takes as input a set of public keys R that constitutes the ring, and a purported ring signature σ of a message M on behalf of the ring R. It returns either valid or invalid.

5.4 Security Model

Informally, a ring signature scheme should satisfy two security properties. First, it should be anonymous, which means an adversary should not be able to determine which member of a ring generated a signature. Second, it should be unforgeable, which means an adversary should be able to construct a valid signature on behalf of a ring only if he knows the secret key corresponding to one of them. Rivest, Shamir, and Tauman [RST01] gave a formalization which has been used in much subsequent work. Bender, Katz, and Morselli [BKM06] described several possible stronger formulations of each notion. In addition to this, as a quotable signature, it also should be context hiding, which means a derived signature on M', from an honestly generated original signature on M, is statistically indistinguishable from a fresh signature on M', even if the original signature on M is known. Ahn et al. [ABC+12] proposed a strong definition of context hiding for quotable signature.

For identity-based quotable ring signature scheme, the security model should be slightly modified. For example, in terms of the unforgeability property, new ring signatures quoted from an valid original ring signature should not be considered as a forgery even if the adversary did not know the secret key corresponding to one of the members of the ring.

Correctness. We require that for all private key S_{ID} generated by Extract algorithm and for all $M \in \mathcal{M}$, all substring $M' \subseteq M$, and all ring of public keys R where $Q_{ID} \in R$ we have:

- For both Type I and Type II signatures, Sign$(S_{ID}, R, M) \neq \perp$ and Verify$(R, M,$ Sign$(S_{ID}, R, M)) = 1$.
- For all Type I signature σ_I generated by $\sigma_I \leftarrow Sign(S_{ID}, R, M')$ or $\sigma_I \leftarrow$ Quote(σ'_I, R, M, M'), Quote$(\sigma_I, R, M', M'') \neq \perp$, and Verify $(R, M'', Quote(\sigma_I,$ $R, M', M'')) = 1$.

In particular, correctness implies that a signature generated by Quote algorithm can be used as an input to Quote algorithm so that signatures can be further quoted from quoted signatures.

Anonymity. We require that any verifier should not have probability greater than $1/d$ to guess the identity of the actual signer who has computed a ring signature on behalf of a ring of d members. If the verifier is a member of the ring distinct from the actual signer, then his probability to guess the identity of the actual signer should not be greater than $1/(d-1)$.

Anonymity against full key exposure for an identity-based quotable ring signature scheme is defined using the following game between a challenger and an adversary $\mathcal{A}$.

- **Setup.** The challenger selects $\mathcal{ID} = \{ID_1, \cdots, ID_\xi\}$ where ξ is a game parameter. The adversary $\mathcal{A}$ is given the public key set $\mathcal{ID}$.
- **Queries.** Algorithm $\mathcal{A}$ is allowed to make ring signing queries and extract queries. A ring signing query is of the form (i', R, M). Here M is the message to be signed. R is a set of public keys, and i' is an index such that $Q_{ID_{i'}} \in R$ holds. (The other keys in R need not be keys in the set $\mathcal{ID}$). The challenger responds with $\sigma = Sign(S_{ID_{i'}}, R, M)$. An extract query is of the form ID_i. The challenger provides S_{ID_i} to $\mathcal{A}$.

- **Challenge.** Algorithm $\mathcal{A}$ requests a challenge by sending to the challenger the values (i_0, i_1, R, M). Here M is to be signed with respect to the ring R, and i_0 and i_1 are indices such that $\{Q_{ID_{i_0}}, Q_{ID_{i_1}}\} \in R$. (The other keys in R need not be keys in the set $\mathcal{ID}$.) The challenger chooses a bit $b \leftarrow \{0, 1\}$, computes the challenge signature $\sigma \leftarrow \text{Sign}(S_{ID_{i_b}}, R, M)$, and provides $\mathcal{A}$ with σ.

- **Output.** Algorithm $\mathcal{A}$ finally outputs its guess b' for b, and wins if $b = b'$.

We define $Adv_{\mathcal{A}}^{anon-ke}$ to be the advantage over 1/2 of $\mathcal{A}$ in the above game.

Unforgeability. We prove our construction selectively secure. Selective security for signature schemes requires the attacker to give the forgery message before seeing the verification key.

To define unforgeability, we extend the basic notion of existential unforgeability with respect to adaptive chosen-message attacks [GMR88]. The definition captures the idea that if the attacker is given a set of signed messages (either primary or quoted), then the only messages he can sign are derivations of the signed messages he was given.

Unforgeability for an identity-based quotable ring signature scheme is defined using the following game between a challenger and an adversary $\mathcal{A}$.

- **Setup.** The challenger selects $\mathcal{ID} = \{ID_1, \cdots, ID_\xi\}$ where ξ is a game parameter. The adversary $\mathcal{A}$ is given the public key set $\mathcal{ID}$.

- **Queries.** Algorithm $\mathcal{A}$ is allowed to make ring signing queries and extract queries. A ring signing query is of the form (i', R, M). Here M is the message to be signed. R is a set of public keys, and i' is an index such that $Q_{ID_{i'}} \in R$ holds. (The other keys in R need not be keys in the set $\mathcal{ID}$). The challenger responds with $\sigma = Sign(S_{ID_{i'}}, R, M)$. An extract query is of the form ID_i. The challenger provides S_{ID_i} to $\mathcal{A}$.

- **Output.** Eventually, $\mathcal{A}$ outputs a tuple (R^*, M^*, σ^*) and wins the game if (1) it never made a ring signing query (i, R, M) such that $Q_{ID_i} \in R^*$ and M equals to or is a superstring of M^*; (2) it never made an extract query ID_i for any $ID_i \in \mathcal{ID}$, and $R^* \subseteq \mathcal{ID}$; and (3) Verify $(R^*, M^*, \sigma^*) =$ valid.

An identity-based quotable ring signature schemes is selectively unforgeable with

respect to adaptive chosen-message attacks if for all PPT adversaries $\mathcal{A}$, the probability that he wins the game is negligible in λ.

We define $Adv_{\mathcal{A}}^{sel-uf}$ to be the probability that $\mathcal{A}$ wins in the above game.

Context Hiding. The notion of anonymity has considered the problem of hiding the identity of a signer among a set of users. Context hiding ensures privacy for the data rather than the signer. Our goal is to hide how a ring signature was created. Context hiding captures an important privacy property, which means a ring signature should reveal nothing more than the message being signed. In particular, if a ring signature on M' was quoted from a ring signature on M, an attacker should not learn anything about M other than what can be inferred from M'. This should be true even if the original ring signature on M is revealed. For example, a signed quote should not reveal anything about the message from which it was quoted, including its length, the position of the quote, whether its parent document is the same as another quote, whether it was derived from a given signed message or generated freshly, etc.

We can view a message M as a pair $(t, m) \in \{0,1\}, \{0,1\}^*$. The bit t will identify the message as being Type I or Type II (assume $t = 1$ signifies Type I signatures) and m will be the message to be signed. We note that this description allows an attacker to distinguish between any Type I signature from any Type II signature since the "type bit" of the messages will be different and thus they will technically be two different messages even if the message components are equal. For this reason we will only need to prove context hiding between messages of Type I or Type II, but not across types. In general, flipping the bit t will not result in a valid signature of a different type on the same core message, because the format will be wrong. However, moving from a Type I to a Type II on the same core message is not considered a forgery since Type II signatures can be legally derived from Type I.

We put forth the following powerful statistical definition of context hiding.

Let $M \in \mathcal{M}$ and $M' \subseteq M$ be a substring of M. Let $S_{ID_i} \leftarrow \text{Extract}(ID_i)$, R be a ring such that $Q_{ID_i} \in R$. An identity-based quotable ring signature scheme is strongly context hiding if for all such triples (S_{ID_i}, M, M'), the following two distribution are statistically close.

$$\{(S_{ID_i}, \sigma_M \leftarrow \text{Sign}(S_{ID_i}, R, M), \text{Sign}(S_{ID_i}, R, M'))\}_{S_{ID_i}, M, M'},$$

$$\{(S_{ID_i}, \sigma_M \leftarrow \text{Sign}(S_{ID_i}, R, M), \text{Quote}(\sigma_M, R, M, M'))\}_{S_{ID_i}, M, M'}.$$

The definition states that a derived signature on M', from an honestly generated original ring signature, is statistically indistinguishable from a fresh ring signature on M'. This implies that a derived ring signature on M' is indistinguishable from a ring signature generated independently of M. Therefore, the derived ring signature cannot (provably) reveal any information about M beyond what is revealed by M'.

Using statistical indistinguishability meaning that even an unbounded adversary cannot distinguish derived ring signatures from newly created ones. The same holds even if the signing key is leaked.

5.5 Proposed Scheme

The design of this scheme follows the idea of the Ahn et al. quotable signature scheme [ABC+12], which is not identity-based and not ring signature.

Setup (1^λ): On input of the security parameter 1^λ, construct a group $\mathbb{G}$ of composite order $n = pq$ as described in Sect.3.1. Let L be the maximum message length supported and denote $n' = \lfloor \lg(L) \rfloor$. Let $H_a : \{0,1\}^* \to \mathbb{G}^*, H_b : \{0,1\}^* \to \mathbb{G}^*$, and $H : \{0,1\}^* \to \mathbb{G}^*$ be three hash functions. Choose random $w, z_0, \cdots, z_{n'-1}, \alpha \leftarrow \mathbb{Z}_n^*$. Set $W = g^w, U = h^\alpha, P_{pub} = g^\alpha$. The master secret key is MK $= (\alpha)$. The public parameter is $params = (H_a, H_b, H, g, h, n, g^{z_0}, \cdots, g^{z_{n'-1}}, W, U, P_{pub})$.

Extract (MK, ID): On input of the master secret key MK and a user's identity ID, compute $Q_{ID} = H(ID)$, the user's secret key is $S_{ID} = Q_{ID}^\alpha$.

Sign ($S_{ID}, R, M = (t, m) \in \{0,1\} \times \sum^{l \leq L}$): We sketch how this algorithm works for a message of length l. Firstly, visualize a matrix with $(l+1)$ columns and $(\lfloor \lg l \rfloor + 2)$ rows. The columns correspond to the characters of the message, with a character in between each column. The rows correspond to the numbers $\lg l$ down to 0, plus an extra row at the bottom. Each location (i_c, i_r) in the matrix (except along the bottom-most row) contains one or more out-going arrows. A "start" arrow goes down one row and over 2^{i_r} columns ending in $(i_c + 2^{i_r}, i_r - 1)$, if this end point is in the matrix. This type of arrow indicates that a quote starts here. A "one" arrow operates similarly to start arrows and is used to include characters after a start arrow includes the quote prefix. A "zero" arrow goes straight down one row ending in $(i_c, i_r - 1)$. This does not add any characters to the quoted substring. We refer the reader to [ABC+12] for more details about this.

This algorithm takes as input a signer's private key S_{ID}, a ring R of public keys, and a message M. No public key may appear twice in R, and R must include Q_{ID}. If $t = 1$, ring signatures produced by this algorithm are Type I as described below. If $t = 0$, the Type II signature can be obtained by running this algorithm and then running the Quote-Type II algorithm below to obtain a quote on the entire message. The message space is treated as $l \leq L$ symbols from alphabet $\sum$. We use notation $m_{i,j}$ to denote the substring of m of length j starting at position i.

Let $d = |R|$, parse the elements of R as $Q_{ID_k} \in \mathbb{G}, 1 \leq k \leq d$. Let k^* be the index such that $Q_{ID_{k^*}} = Q_{ID}$. Define $\{f_k\}_{k=1}^{d}$ as

$$f_k = \begin{cases} 1 & \text{if} \quad k = k^*, \\ 0 & \text{otherwise.} \end{cases}$$

For $i = 3$ to $l + 1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$, choose random values $x_{i,j} \in \mathbb{Z}_n$. Set $x_{i,-1} = 0$ for all $i = 1$ to $l + 1$.

- Choose random number u from $\mathbb{Z}_n^*$, and set $\bar{U} = h^u, \widehat{U} = h^{1/u}, \widetilde{U} = g^u$.
- For each $k, 1 \leq k \leq d$, choose random exponents s_k from $\mathbb{Z}_n^*$ and set

$$V_k = (Q_{ID_k}/W)^{f_k} h^{s_k},$$

$$\pi_{k1} = h^{s_k u}, \quad \pi_{k2} = \left((Q_{ID_k}/W)^{2f_k - 1} h^{s_k}\right)^{1/u}, \quad \pi_{k3} = g^{s_k u}, \quad \pi_{k4} = g^{s_k}.$$

- Let $s = \sum_{k=1}^{d} s_k$, and set $G = P_{pub}^s = g^{\alpha s}$.
- For $i = 1$ to l and $j = 0$ to $\lfloor \lg(l - i + 1) \rfloor$, for randomly chosen values $r_{i,j} \in \mathbb{Z}_n$:

$$B_{i,j} = H_b(m_{i,2^j})^{r_{i,j}} g^{-x_{i+2^j,j-1}} S_{ID} \cdot U^s, \quad \widetilde{B}_{i,j} = g^{r_{i,j}}.$$

- For $i = 3$ to l and $j = 0$ to $\min(\lfloor \lg(i-1) - 1 \rfloor, \lfloor \lg(l - i + 1) \rfloor)$, for randomly chosen values $r'_{i,j} \in \mathbb{Z}_n$:

$$A_{i,j} = H_a(m_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}}, \quad \widetilde{A}_{i,j} = g^{r'_{i,j}}.$$

- For $i = 3$ to $l + 1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$, for randomly chosen values $r''_{i,j} \in \mathbb{Z}_n$:

$$D_{i,j} = g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j \cdot r''_{i,j}}, \quad \widetilde{D}_{i,j} = g^{r''_{i,j}}.$$

The ring signature is $\sigma = (\bar{U}, \widehat{U}, \widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^{d}, G, B_{i,j}, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}, D_{i,j}, \widetilde{D}_{i,j})$.

The values $\pi_{k1}, \pi_{k2}, \pi_{k3}$ and π_{k4} act as a proof that V_k is well-formed, which means that $f_k \in \{0, 1\}$.

Observe that, when there is exactly one non-zero value amongst $\{f_k\}$, say f_{k^*}, we have $W \cdot \prod_{k=1}^{d} V_k = Q_{ID^*} h^s$, so $\prod_{k=1}^{d} V_k$ serves as an encryption of the user's public key.

Quote $(\sigma, R, M = (t, m), M' = (t', m'))$: To derive a new ring signature on a substring M' of M, one removes the group elements not associated with the new substring and then re-randomizes the remaining part of the ring signature. In addition, there is a second option in our quote algorithm that allows for the derivation of a short ring signature. However the quote procedure cannot be applied again to this short ring signature. Thus, we support quoting from quotes, and also provide a compression option which produces a very short quote, but the price for this is that it cannot be quoted from further.

First, check the validity of σ with respect to R and M. If it is not valid, output $\perp$. If M' is not a substring of M, output $\perp$. Otherwise, if $t' = 1$, output Quote-Type I (σ, R, m, m'); if $t' = 0$, output Quote-Type II (σ, R, m, m').

Quote-Type I (σ, R, m, m'): This quote algorithm takes a Type I signature and produces another Type I signature that maintains the ability to be quoted again.

If m' is not a substring of m, then output $\perp$. Otherwise, let $l' = |m'|$. Determine the first index δ at which substring m' occurs in m. Parse σ as a collection of $\bar{U}, \widehat{U}, \widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^{d}, G, B_{i,j}, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}, D_{i,j}, \widetilde{D}_{i,j}$ values.

Choose re-randomization values to re-randomize the $x_{i,j}$ terms of σ. For $i = 2$ to $l' + 1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$, choose random values $y_{i,j} \in \mathbb{Z}_n$. Set $y_{i,-1} = 0$ for all $i = 1$ to $l' + 1$.

- Choose random number u' from $\mathbb{Z}_n^*$, and set $\bar{U}' = \bar{U}^{u'} = h^{uu'}, \widehat{U}' = \widehat{U}^{(1/u')} = h^{(1/uu')}, \widetilde{U}' = \widetilde{U}^{u'} = g^{uu'}$.

- For each $k, 1 \le k \le d$, choose random exponents s'_k from $\mathbb{Z}_n^*$ and set

$$\begin{aligned} V'_k &= V_k \cdot h^{s'_k} = (Q_{ID_k}/W)^{f_k} h^{(s_k + s'_k)}, \\ \pi'_{k1} &= (\pi_{k1} \cdot \bar{U}^{s'_k})^{u'} = (h^{(s_k + s'_k)})^{uu'}, \\ \pi'_{k2} &= (\pi_{k2} \cdot \widehat{U}^{s'_k})^{(1/u')} = \left((Q_{ID_k}/W)^{2f_k - 1} h^{(s_k + s'_k)}\right)^{(1/uu')}, \end{aligned}$$

$$\begin{aligned}\pi'_{k3} &= (\pi_{k3} \cdot \widetilde{U}^{s'_k})^{u'} = (g^{(s_k+s'_k)})^{uu'},\\ \pi'_{k4} &= \pi_{k4} \cdot g^{s'_k} = g^{(s_k+s'_k)}.\end{aligned}$$

- Let $s' = \sum_{k=1}^{d} s'_k$, and set $G' = G \cdot P_{pub}^{s'} = g^{\alpha(s+s')}$.
- For $i = 1$ to l' and $j = 0$ to $\lfloor \lg(l' - i + 1) \rfloor$, for randomly chosen $t_{i,j} \in \mathbb{Z}_n$:

$$B'_{i,j} = B_{i+\delta-1,j} \cdot H_b(m_{i+\delta-1,2^j})^{t_{i,j}} \cdot g^{-y_{i+2^j,j-1}} \cdot U^{\sum_{k=1}^{d} s'_k}, \quad \widetilde{B}'_{i,j} = \widetilde{B}_{i+\delta-1,j} \cdot g^{t_{i,j}}.$$

- For $i = 3$ to l' and $j = 0$ to $\min(\lfloor \lg(i-1) - 1 \rfloor, \lfloor \lg(l' - i + 1) \rfloor)$, for randomly chosen values $t'_{i,j} \in \mathbb{Z}_n$:

$$A'_{i,j} = A_{i+\delta-1,j} \cdot H_a(m_{i+\delta-1,2^j})^{t'_{i,j}} \cdot g^{y_{i,j}} \cdot g^{-y_{i+2^j,j-1}}, \quad \widetilde{A}'_{i,j} = \widetilde{A}_{i+\delta-1,j} \cdot g^{t'_{i,j}}.$$

- For $i = 3$ to $l' + 1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$, for randomly chosen values $t''_{i,j} \in \mathbb{Z}_n$:

$$D'_{i,j} = D_{i+\delta-1,j} \cdot g^{y_{i,j}} \cdot g^{-y_{i,j-1}} \cdot g^{z_j \cdot t''_{i,j}}, \quad \widetilde{D}'_{i,j} = \widetilde{D}_{i+\delta-1,j} \cdot g^{t''_{i,j}}.$$

The ring signature is $\sigma' = (\bar{U}', \widehat{U}', \widetilde{U}', \{V'_k, \pi'_{k1}, \pi'_{k2}, \pi'_{k3}, \pi'_{k4}\}_{k=1}^{d}, G', B'_{i,j}, \widetilde{B}'_{i,j}, A'_{i,j}, \widetilde{A}'_{i,j}, D'_{i,j}, \widetilde{D}'_{i,j})$.

Quote-Type II (σ, R, m, m'): This quote algorithm takes a Type I signature and produces a Type II signature. A Type II quote will trace a $(\lg(l') + 1)$-length path on those arrows through the matrix of the original Type I signature, where l' is the length of the quote. It always starts with a start arrow and then contains one and zero arrows according to the binary representation of the length of the quote. Intuitively, taking an arrow over a character includes it in the quote.

Consider the length l' written as a binary string. Let β' be the largest index of $l' = |m'|$ that is set to 1, where we start counting with zero as the least significant bit. That is, set $\beta' = \lfloor \lg(l') \rfloor$. Select random values $v, v_{\beta'-1}, \cdots, v_0 \in \mathbb{Z}_n$.

- Choose random number u' from $\mathbb{Z}_n^*$, and set $\widetilde{U}' = \widetilde{U}^{u'} = g^{uu'}$.
- For each $k, 1 \le k \le d$, choose random exponents s'_k from $\mathbb{Z}_n^*$ and set

$$\begin{aligned}V'_k &= V_k \cdot h^{s'_k} = (Q_{ID_k}/W)^{f_k} h^{(s_k+s'_k)},\\ \pi'_{k1} &= (\pi_{k1} \cdot \bar{U}^{s'_k})^{u'} = (h^{(s_k+s'_k)})^{uu'},\\ \pi'_{k2} &= (\pi_{k2} \cdot \widehat{U}^{s'_k})^{(1/u')} = \left((Q_{ID_k}/W)^{2f_k-1} h^{(s_k+s'_k)}\right)^{(1/uu')},\\ \pi'_{k3} &= (\pi_{k3} \cdot \widetilde{U}^{s'_k})^{u'} = (g^{(s_k+s'_k)})^{uu'},\\ \pi'_{k4} &= \pi_{k4} \cdot g^{s'_k} = g^{(s_k+s'_k)}.\end{aligned}$$

- Let $s' = \sum_{k=1}^{d} s'_k$, and set $G' = G \cdot P_{pub}^{s'} = g^{\alpha(s+s')}$.
- Set the start position as $K' = B_{\delta,\beta'} \cdot U^{\sum_{k=1}^{d} s'_k}$, and $\delta' = \delta + 2^{\beta'}$. Then, from $j = \beta' - 1$ down to 0, proceed as follows.
  - If the jth bit of l' is 1, set $K' = K' \cdot A_{\delta',j} \cdot H_a(m_{\delta',2^j})^{v_j}$, and $Z'_j = \widetilde{A}_{\delta',j} \cdot g^{v_j}$. Then, set $\delta' = \delta' + 2^j$;
  - If the jth bit of l' is 0, set $K' = K' \cdot D_{\delta',j} \cdot g^{z_j \cdot v_j}$ and $Z'_j = \widetilde{D}_{\delta',j} \cdot g^{v_j}$.
- To end, re-randomize as $K' = K' \cdot H_b(m_{\delta,2^{\beta'}})^v$ and $\widetilde{B}' = \widetilde{B}_{\delta,\beta'} \cdot g^v$.

Output the quote as $\sigma' = \left(\widetilde{U}', \{V'_k, \pi'_{k1}, \pi'_{k2}, \pi'_{k3}, \pi'_{k4}\}_{k=1}^{d}, G', K', \widetilde{B}', Z'_{\beta'-1}, \cdots, Z'_0\right)$.

Verify $(R, M = (t, m), \sigma)$: Let $d = |R|$, parse the elements of R as $Q_{ID_k} \in \mathbb{G}, 1 \leq k \leq d$. Verify that no element is repeated in R and reject otherwise.

Parse Type I signature as $\sigma = (\bar{U}, \widehat{U}, \widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^{d}, G, B_{i,j}, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}, D_{i,j}, \widetilde{D}_{i,j},)$. Parse Type II signature as $\sigma = (\widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^{d}, G, K, \widetilde{B}, Z_{\beta-1}, \cdots, Z_0)$.

Check that the proofs $\{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^{d}$ are valid. First check $\{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^{d} \in \mathbb{G}^{5d}$. Then, for each $k, 1 \leq k \leq d$, check whether

$$\begin{aligned} e(V_k, V_k/(Q_{ID_k}/W)) &\stackrel{?}{=} e(\pi_{k1}, \pi_{k2}), \\ e(\pi_{k1}, g) &\stackrel{?}{=} e(h, \pi_{k3}), \\ e(\pi_{k4}, \widetilde{U}) &\stackrel{?}{=} e(g, \pi_{k3}), \end{aligned}$$

hold. If any of the proofs is invalid, reject. Otherwise, set $V = \prod_{k=1}^{d} V_k$. Then, check whether $e(\prod_{k=1}^{d} \pi_{k4}, P_{pub}) \stackrel{?}{=} e(G, g)$ holds. If it is invalid, reject.

If $t = 1$, output **Verify-Type I(R, m, σ)**. Otherwise, output **Verify-Type II(R, m, σ)**.

Verify-Type I (R, m, σ): Let $l = |m|$. Let $X_{i,j}$ denote $e(g, g)^{x_{i,j}}$. The value $X_{i,-1} = 1$, since for all $i = 1$ to $l + 1$, $x_{i,-1} = 0$.

For $i = 3$ to $l + 1$ and $j = 0$ to $\lfloor \lg(i - 1) - 1 \rfloor$, let $I = i - 2^{j+1}$ and $J = j + 1$. Compute

$$X_{i,j} = \left(e\left(H_b(m_{I,2^J}), \widetilde{B}_{I,J}\right) \cdot e\left(WV, P_{pub}\right)\right) / e\left(B_{I,J}, g\right).$$

The verification accepts if and only if all of the following hold.

- For $i = 3$ to l and $j = 0$ to $\min(\lfloor \lg(i - 1) - 1 \rfloor, \lfloor \lg(l - i + 1) \rfloor)$,

$$e\left(A_{i,j}, g\right) = \left(X_{i,j} / X_{i+2^j, j-1}\right) \cdot e\left(H_a(m_{i,2^j}), \widetilde{A}_{i,j}\right). \tag{5.1}$$

- For $i = 3$ to $l + 1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$,

$$e(D_{i,j}, g) = (X_{i,j}/X_{i,j-1}) \cdot e\left(g^{z_j}, \widetilde{D}_{i,j}\right). \tag{5.2}$$

Verify-Type II (R, m, σ): Let $l = |m|$ and β be the index of the highest bit of l that is set to 1. Set $N = 1$ and $\delta = 1 + 2^\beta$. From $j = \beta - 1$ down to 0, proceed as follows.

- If the jth bit of l is 1, set $N = N \cdot e\left(H_a\left(m_{\delta,2^j}\right), Z_j\right)$ and $\delta = \delta + 2^j$.
- If the jth bit of l is 0, set $N = N \cdot e(g^{z_j}, Z_j)$.

Accept if and only if

$$e(K, g) = e\left(H_b(m_{1,2^\beta}), \widetilde{B}\right) \cdot e\left(WV, P_{pub}\right) \cdot N. \tag{5.3}$$

5.6 Security Analysis

Theorem 5.1 *This identity-based quotable ring signature scheme is correct.*

Proof. For the **Type I signature.** Let $l = |m|$. Parse σ as the set of $\bar{U}, \widehat{U}, \widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^d, G, B_{i,j}, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}, D_{i,j}, \widetilde{D}_{i,j}$.

$$\begin{aligned}
\text{When } f_k = 1, \quad e\left(V_k, V_k/(Q_{ID_k}/W)\right) &= e\left((Q_{ID_k}/W)h^{s_k}, h^{s_k}\right) \\
&= e\left(h^{s_k u}, \left((Q_{ID_k}/W)h^{s_k}\right)^{1/u}\right) \\
&= e(\pi_{k1}, \pi_{k2}).
\end{aligned}$$

$$\begin{aligned}
\text{When } f_k = 0, \quad e\left(V_k, V_k/(Q_{ID_k}/W)\right) &= e\left(h^{s_k}, h^{s_k}/(Q_{ID_k}/W)\right) \\
&= e\left(h^{s_k u}, \left(h^{s_k}/(Q_{ID_k}/W)\right)^{1/u}\right) \\
&= e(\pi_{k1}, \pi_{k2}).
\end{aligned}$$

Thus, $e\left(V_k, V_k/(Q_{ID_k}/W)\right) = e(\pi_{k1}, \pi_{k2})$.

Through substitution, we can also check that the following equations hold.

$$\begin{aligned}
e(\pi_{k1}, g) = e(h^{s_k u}, g) = e(h, g^{s_k u}) = e(h, \pi_{k3}). \\
e(\pi_{k4}, \widetilde{U}) = e(g^{s_k}, g^u) = e(g, g^{s_k u}) = e(g, \pi_{k3}). \\
e(\textstyle\prod_{k=1}^d \pi_{k4}, P_{pub}) = e(g^s, g^\alpha) = e(g^{\alpha s}, g) = e(G, g).
\end{aligned}$$

For $i = 3$ to $l+1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$,

$$\begin{aligned}
X_{i,j} &= \left(e\left(H_b(m_{I,2^J}), \widetilde{B}_{I,J}\right) \cdot e\left(WV, P_{pub}\right)\right) / e\left(B_{I,J}, g\right) \\
&= \left(e\left(H_b(m_{i-2^{j+1},2^{j+1}}), \widetilde{B}_{i-2^{j+1},j+1}\right) \cdot e\left(Q_{ID} \cdot h^s, g^\alpha\right)\right) / e\left(B_{i-2^{j+1},j+1}, g\right) \\
&= \frac{\left(e\left(H_b(m_{i-2^{j+1},2^{j+1}}), g^{r_{i-2^{j+1},j+1}}\right) \cdot e\left(Q_{ID} \cdot h^s, g^\alpha\right)\right)}{e\left(H_b\left(m_{i-2^{j+1},2^{j+1}}\right)^{r_{i-2^{j+1},j+1}} g^{-x_{i-2^{j+1}+2^{j+1},j+1-1}} S_{ID} \cdot U^s, g\right)} \\
&= \frac{\left(e\left(H_b(m_{i-2^{j+1},2^{j+1}}), g^{r_{i-2^{j+1},j+1}}\right) \cdot e\left(Q_{ID} \cdot h^s, g^\alpha\right)\right)}{e\left(H_b\left(m_{i-2^{j+1},2^{j+1}}\right)^{r_{i-2^{j+1},j+1}} g^{-x_{i,j}} Q_{ID}^\alpha \cdot h^{\alpha s}, g\right)} \\
&= 1 / e\left(g^{-x_{i,j}}, g\right) \\
&= e\left(g, g\right)^{x_{i,j}}.
\end{aligned}$$

For $i = 3$ to l and $j = 0$ to $\min(\lfloor \lg(i-1) - 1 \rfloor, \lfloor \lg(l-i+1) \rfloor)$,

$$\begin{aligned}
e\left(A_{i,j}, g\right) &= e\left(H_a(m_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}}, g\right) \\
&= \left(e\left(g, g\right)^{x_{i,j}} / e\left(g, g\right)^{x_{i+2^j,j-1}}\right) \cdot e\left(H_a(m_{i,2^j}), g^{r'_{i,j}}\right) \\
&= \left(X_{i,j} / X_{i+2^j,j-1}\right) \cdot e\left(H_a(m_{i,2^j}), \widetilde{A}_{i,j}\right).
\end{aligned}$$

For $i = 3$ to $l+1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$,

$$\begin{aligned}
e(D_{i,j}, g) &= e\left(g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j \cdot r''_{i,j}}, g\right) \\
&= \left(e\left(g, g\right)^{x_{i,j}} / e\left(g, g\right)^{x_{i,j-1}}\right) \cdot e\left(g^{z_j}, g^{r''_{i,j}}\right) \\
&= \left(X_{i,j} / X_{i,j-1}\right) \cdot e\left(g^{z_j}, \widetilde{D}_{i,j}\right).
\end{aligned}$$

Both equation (5.1) and (5.2) hold.

For the **Type II signature.** Parse σ' as the set of $\widetilde{U}', \{V'_k, \pi'_{k1}, \pi'_{k2}, \pi'_{k3}, \pi'_{k4}\}_{k=1}^d$, $G', K', \widetilde{B}', Z'_{\beta-1}, \cdots, Z'_0$. Recall that $l' = |m'|$ and $\beta' = \lfloor \lg(l') \rfloor$. Here l'_j denotes the j-th bit of l' when we start counting with zero as the least significant bit. Parse m' as $m'_{\beta'} m'_{\beta'-1} \cdots m'_0$ where m'_j is a string of length 2^j (when $l'_j = 1$) or a null string (when $l'_j = 0$). ψ_j denotes the position where m'_j starts.

The values of some elements in the verification of Type II signature are as follows.

$$K' = H_b(m_{\delta,2^\beta})^{\left(r_{\delta,\beta}+v\right)} U^{(s+s')} S_{ID} \cdot \prod_{j<\beta, l'_j=1} H_a(m'_j)^{\left(r'_{\delta+\psi_j-1,j}+v_j\right)} \cdot \prod_{j<\beta, l'_j=0} g^{z_j\left(r''_{\delta+\psi_j-1,j}+v_j\right)},$$

$$\{Z'_j\}_{j<\beta, l'_j=1} = g^{\left(r'_{\delta+\psi_j-1,j}+v_j\right)}, \quad \{Z'_j\}_{j<\beta, l'_j=0} = g^{\left(r''_{\delta+\psi_j-1,j}+v_j\right)},$$

$$N' = \prod_{j<\beta, l'_j=1} e\left(H_a(m'_j), Z'_j\right) \cdot \prod_{j<\beta, l'_j=0} e\left(g^{z_j}, Z'_j\right).$$

$$\text{When } f_k = 1, \quad e\left(V'_k, V'_k/(Q_{ID_k}/W)\right) = e\left((Q_{ID_k}/W)h^{(s_k+s'_k)}, h^{(s_k+s'_k)}\right)$$
$$= e\left((h^{(s_k+s'_k)})^{uu'}, \left((Q_{ID_k}/W)h^{(s_k+s'_k)}\right)^{(1/uu')}\right)$$
$$= e(\pi'_{k1}, \pi'_{k2}).$$

$$\text{When } f_k = 0, \quad e\left(V'_k, V'_k/(Q_{ID_k}/W)\right) = e\left(h^{(s_k+s'_k)}, h^{(s_k+s'_k)}/(Q_{ID_k}/W)\right)$$
$$= e\left((h^{(s_k+s'_k)})^{uu'}, \left((Q_{ID_k}/W)^{-1}h^{(s_k+s'_k)}\right)^{(1/uu')}\right)$$
$$= e(\pi'_{k1}, \pi'_{k2}).$$

Thus, $e\left(V'_k, V'_k/(Q_{ID_k}/W)\right) = e(\pi'_{k1}, \pi'_{k2})$.
Through substitution, we can also check that the following equations hold.

$$e(\pi'_{k1}, g) = e\left((h^{(s_k+s'_k)})^{uu'}, g\right) = e\left(h, (g^{(s_k+s'_k)})^{uu'}\right) = e(h, \pi'_{k3}).$$
$$e(\pi'_{k4}, \widetilde{U'}) = e\left(g^{(s_k+s'_k)}, g^{uu'}\right) = e\left(g, (g^{(s_k+s'_k)})^{uu'}\right) = e(g, \pi'_{k3}).$$
$$e(\textstyle\prod_{k=1}^{d} \pi'_{k4}, P_{pub}) = e(g^{(s+s')}, g^{\alpha}) = e(g^{\alpha(s+s')}, g) = e(G', g).$$

$$\begin{aligned}
&e(K', g)\\
&= e(H_b(m_{\delta,2^\beta})^{(r_{\delta,\beta}+v)}U^{(s+s')}S_{ID}\prod_{j<\beta, l'_j=1} H_a(m'_j)^{(r'_{\delta+\psi_j-1,j}+v_j)}\prod_{j<\beta, l'_j=0} g^{z_j(r''_{\delta+\psi_j-1,j}+v_j)}, g)\\
&= e\left(H_b(m_{\delta,2^\beta}), g^{\left(r_{\delta,\beta}+v\right)}\right)e(Q_{ID}\cdot h^{(s+s')}, g^{\alpha})\prod_{j<\beta, l'_j=1} e\left(H_a(m'_j), Z'_j\right)\prod_{j<\beta, l'_j=0} e(g^{z_j}, Z'_j)\\
&= e\left(H_b(m'_{1,2^\beta}), \widetilde{B}\right)\cdot e\left(WV', P_{pub}\right)\cdot N'.
\end{aligned}$$

Equation (5.3) holds. □

Theorem 5.2 *This identity-based quotable ring signature scheme is anonymous against full key exposure under the assumption that the subgroup decision problem is hard.*

Proof. The anonymity proof closely follows that given by Shacham and Waters for their ring signature scheme [SW07].

The proof proceeds in games. We define Game 0 as follows. Algorithm $\mathcal{B}$ is given the group order n (but not its factorization), the description of the group $\mathbb{G}$, together with generators g of $\mathbb{G}$ and h which is uniformly chosen from $\mathbb{G}_q$. $\mathcal{B}$ follows the Setup algorithm to obtain system parameters $(w, z_0, \cdots, z_{n'-1}, \alpha, W, U, P_{pub})$. $\mathcal{B}$ also chooses three hash functions H_a, H_b, H and selects $\mathcal{ID} = \{ID_1, \cdots, ID_\xi\}$.

$\mathcal{B}$ runs $\mathcal{A}$, providing to it the description of the group $\mathbb{G}$, including its order n and the generators g and h. $\mathcal{B}$ also provides to $\mathcal{A}$ system parameters $(w, z_0, \cdots, z_{n'-1}, \alpha,$

W, U, P_{pub}), along with the description of the hash functions H_a, H_b, H, and the challenge public keys $\{Q_{ID_i}\}_{i=1}^{\xi}$. When $\mathcal{A}$ makes a ring signing query of the form (j, R, M), $\mathcal{B}$ responds with $\sigma = Sign(S_{ID_j}, R, M)$. Finally, $\mathcal{A}$ requests a challenge with the values (j_0, j_1, R, M). Algorithm $\mathcal{B}$ chooses a bit $b \leftarrow \{0, 1\}$, computes the challenge ring signature $\sigma = Sign(S_{ID_{j_b}}, R, M)$, and provides $\mathcal{A}$ with σ. In addition, the challenger provides $\mathcal{A}$ with the Extract oracle, which can be queried for any identity. Actually, as the secret key is also given to the adversary $\mathcal{A}$, ring signing queries and Extract queries could be answered by himself. Algorithm $\mathcal{A}$ finally outputs its guess b' for b. $\mathcal{B}$ outputs 1 if $b = b'$, 0 otherwise.

Game 1 is identical to Game 0, except for the h is uniformly chosen from $\mathbb{G}$.

Denote by $Adv_{\mathcal{B}}^{game-0}$ the advantage $\mathcal{B}$ has over 1/2 in Game 0, and by $Adv_{\mathcal{B}}^{game-1}$ the advantage over 1/2 it has in Game 1. We have $Adv_{\mathcal{B}}^{game-0} = Adv_{\mathcal{A}}^{anon-ke}$, since in Game 0 $\mathcal{A}$'s environment is exactly as specified in the anonymity game. Moreover, suppose that $\mathcal{B}$'s output were different in the two games. Then we could use $\mathcal{B}$, with $\mathcal{A}$ as a subroutine, to solve the subgroup decision problem. Given generators (g, h) to test, we provide them to $\mathcal{B}$ and output 1 if $\mathcal{B}$ does. This gives a new algorithm $\mathcal{C}$ for which we have

$$\begin{aligned} Adv_{\mathcal{C}}^{sdp} &= \left| Pr[\mathcal{B} = 1 | h \xleftarrow{R} \mathbb{G}_q] - Pr[\mathcal{B} = 1 | h \xleftarrow{R} \mathbb{G}] \right| \\ & \frac{1}{2} \left| (2Pr[\mathcal{B} = 1, h \xleftarrow{R} \mathbb{G}_q] - 1) - (2Pr[\mathcal{B} = 1 | h \xleftarrow{R} \mathbb{G}] - 1) \right| \\ & \frac{1}{2} \left| (Adv_{\mathcal{B}}^{game-0} - Adv_{\mathcal{B}}^{game-1} \right|. \end{aligned}$$

We argue that $Adv_{\mathcal{B}}^{game-1} = 0$, even if $\mathcal{A}$ is computationally unbounded. Consider the distinguishing challenge $(\bar{U}, \widehat{U}, \widetilde{U}, \{(V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4})\}_{k=1}^{d}, G, B_{i,j}, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}$, $D_{i,j}, \widetilde{D}_{i,j})$. For each k, we have $V_k = (Q_{ID_k}/W)^{f_k} h^{s_k}$ with $f_k \in \{0, 1\}$ and $s_k \in \mathbb{Z}_n^*$. But when h is a generator of $\mathbb{G}$ there exist $\tau_{k0}, \tau_{k1} \in \mathbb{Z}_n$ and $\gamma \in \mathbb{Z}_n^*$ such that $V_k = (Q_{ID_k}/W) h^{\tau_{k1}} = h^{\tau_{k0}}$ and $g = h^{\gamma}$. Moreover, denoting by $(\pi_k | f_k = b)$ the values which $\pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}$ are assigned if f_k is set to $b \in \{0, 1\}$, we have

$$\begin{aligned} & (\pi_k | f_k = 1) \\ = \; & \left((h^{\tau_{k1}})^u,\ ((Q_{ID_k}/W) h^{\tau_{k1}})^{1/u},\ (g^{\tau_{k1}})^u = (h^{\tau_{k1}})^{u\gamma},\ g^{\tau_{k1}} = (h^{\tau_{k1}})^{\gamma} \right) \\ = \; & \left((h^{\tau_{k0}}/(Q_{ID_k}/W))^u,\ (h^{\tau_{k0}})^{1/u},\ (h^{\tau_{k0}}/(Q_{ID_k}/W))^{u\gamma},\ (h^{\tau_{k0}}/(Q_{ID_k}/W))^{\gamma} \right) \\ = \; & (\pi_k | f_k = 0). \end{aligned}$$

so for each k the tuple $(V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4})$ is consistent with either $f_k = 0$ or $f_k = 1$, and $\mathcal{A}$ can gain no information from this part of the signature. The values

$\bar{U}, \widehat{U}, \widetilde{U}, G, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}, D_{i,j}, \widetilde{D}_{i,j}$ are unrelated to the choice of signer. Thus if $\mathcal{A}$ can gain information, it is only from $B_{i,j}$. But, having fixed $(\bar{U}, \widehat{U}, \widetilde{U}, \{(V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4})\}_{k=1}^{d}, G, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}, D_{i,j}, \widetilde{D}_{i,j})$, $B_{i,j}$ are fixed values to satisfy the verification equations. Specifically, letting $P_{pub} = g^{\alpha}$, $\widetilde{B}_{i,j} = g^{r_{i,j}}$, and $WV = g^{c}$ which serves as an encryption of the user's public key, we have $B_{i,j} = H_b(m_{i,2^j})^{r_{i,j}} g^{-x_{i+2^j,j-1}} \cdot g^{\alpha c}$. Thus these values give no information about whether $S_{ID_{j0}}$ or $S_{ID_{j1}}$ was used to generate the challenge signature, and $\mathcal{A}$ can do no better than guess b. This establishes $Adv_{\mathcal{B}}^{game-1} = 0$. We see that $Adv_{\mathcal{A}}^{anon-ke} \leq 2Adv_{\mathcal{C}}^{sdp}$. If $Adv_{\mathcal{A}}^{anon-ke}$ is non-negligible, then so is $Adv_{\mathcal{C}}^{sdp}$.

Because Type II signatures are quotes on the entire message of corresponding Type I signatures, which are obtained by properly re-randomizing and compressing the corresponding Type I signatures, they will leak nothing about the actual signer when the Type I signatures do so. □

Theorem 5.3 *If the CDH assumption holds in $\mathbb{G}$, then this identity-based quotable ring signature scheme is selectively unforgeable in the random oracle model.*

Proof. The algorithm that makes the reduction is given the factorization of n. This allows it to undo BGN blinding with h^s terms, and to recover from a signature the values f_k used in generating it.

Suppose an adversary $\mathcal{A}$ can produce a forgery, then we can construct an adversary $\mathcal{B}$ that breaks the CDH assumption.

On input the CDH challenge (g, g^a, g^b), $\mathcal{B}$ begins to run $\mathcal{A}$ and proceeds as follows.

$\mathcal{A}$ first announces the message M^* on which he will forge.

Let L be the maximum size of any message and let $n' = \lfloor \lg(L) \rfloor$. Let $M^* = (t^*, m^*)$ and $l^* = |m^*|$. Let β^* be the highest bit of l^* set to 1 (numbering the least significant bit as zero). Let g be the generator of $\mathbb{G}$. Algorithm $\mathcal{B}$ is given $g_1, g_2 \in \mathbb{G}$. Here, $g_1 = g^a, g_2 = g^b$, and (g, g_1, g_2) is a random instance of the CDH problem. Its goal is to compute g^{ab}.

Algorithm $\mathcal{B}$ starts by setting $P_{pub} = g_1 = g^a, h = g^p, U = (g^a)^p = h^a$. $\mathcal{B}$ randomly selects $w \in \mathbb{Z}_n^*$ and set $W = g^w$. For $i = 0$ to $n' - 1$, choose a random $v_i \in \mathbb{Z}_n^*$ and set

$$g^{z_i} = \begin{cases} g^{bv_i} & \text{if the } i\text{th bit of } l^* \text{ is 1;} \\ g^{v_i} & \text{otherwise.} \end{cases}$$

Algorithm $\mathcal{B}$ selects $\mathcal{ID} = \{ID_1, \cdots, ID_\xi\}$. Algorithm $\mathcal{B}$ sends $(g, h, n, g^{z_0}, \cdots, g^{z_{n'-1}}, W, U, P_{pub}, \mathcal{ID})$ to the algorithm $\mathcal{A}$. Algorithm $\mathcal{B}$ will simulate the oracles and interact with the forger $\mathcal{A}$ as described below.

H **queries:** In the simulation, $\mathcal{B}$ maintains a list H-List of tuples $(ID_i, H(ID_i), \mu_i)$ as explained below. This list is initially empty.

If the query ID_i already appears on the H-List in a tuple $(ID_i, H(ID_i), \mu_i)$, then algorithm $\mathcal{B}$ responds with $H(ID_i)$. Otherwise, for the ith new query ID_i, the following conditions are satisfied.

- if $ID_i \notin \mathcal{ID}, \mathcal{B}$ chooses $\mu_i \in_R \mathbb{Z}_n^*$ and sets $H(ID_i) = g^{\mu_i}$. Then $\mathcal{B}$ adds $(ID_i, H(ID_i), \mu_i)$ to the H-List;
- otherwise $ID_i \in \mathcal{ID}, \mathcal{B}$ chooses $\mu_i \in_R \mathbb{Z}_n^*$ and sets $H(ID_i) = g_2^{\mu_i} = g^{b\mu_i}$ where g_2 is in the instance of the CDH problem. Then $\mathcal{B}$ adds $(ID_i, H(ID_i), \mu_i)$ to the H-List.

In either case, $H(ID_i)$ is returned to $\mathcal{A}$ as the answer.

H_b **queries:** If the query has been made before, return the same response as before.

Imagine dividing up m^* into a sequence of segments whose lengths are decreasing powers of two, that is, the first segment would be of length 2^{β^*} where β^* is the largest power of two less than l^*, the second segment would contain the next largest power of two, etc. Let m_j^* denote the segment of m^* corresponding to power j. If no such segment exists, let $m_j^* = \perp$.

Select a random $\eta \in \mathbb{Z}_n^*$ and return the response as:

$$H_b(x) = \begin{cases} g^{\eta} & \text{if } |x| = 2^{\beta^*} \text{ and } m_{\beta^*}^* = x; \\ g^{b\eta} & \text{otherwise.} \end{cases}$$

Note that $H_b(m_j^*)$ is set according to the first method only for the first segment of m^*.

H_a **queries:** If the query has been made before, return the same response as before. Otherwise, select a random $\theta \in \mathbb{Z}_n^*$ and return the response as:

$$H_a(x) = \begin{cases} g^{\theta} & \text{if } |x| = 2^j \text{ and } j < \beta^* \text{ and } m_j^* = x \quad (x \text{ is on the selective path}); \\ g^{b\theta} & \text{otherwise} \qquad\qquad\qquad\qquad\quad (x \text{ is not on the selective path}). \end{cases}$$

Note that $H_a(m_j^*)$ is set according to the first method for all segments of m^* except the first segment $m_{\beta^*}^*$.

Extract queries: If the query has been made before, return the same response as before. Otherwise, algorithm $\mathcal{B}$ checks H-List. If there is no tuple on the H-List containing ID_i, $\mathcal{B}$ will issue this query by itself to ensure that there is a tuple $(ID_i, H(ID_i), \mu_i)$ on the H-List. $\mathcal{B}$ creates and keeps one list Ex-List to simulate Extract oracle. At the beginning of the simulation, this list is empty.

For each Extract query with respect to a user ID_i except for those in the set $\mathcal{ID}$, using the μ_i value in the record on H-List corresponding to ID_i, $\mathcal{B}$ computes and returns $S_{ID_i} = P_{pub}^{\mu_i} = g^{a\mu_i} = (g^{\mu_i})^a = Q_{ID_i}^a$ as the user ID_i's private key. Then, $\mathcal{B}$ records the tuple (ID_i, S_{ID_i}) in the Ex-List.

Sign queries: For every query (i', R, M), check whether $Q_{ID_{i'}} \in R$. If $Q_{ID_{i'}}$ is not in the ring R, output $\perp$ to indicate the query is invalid. Otherwise, choose random number u from $\mathbb{Z}_n^*$ and set $\bar{U} = h^u, \widehat{U} = h^{1/u}, \widetilde{U} = g^u$.

Let $d = |R|$, parse the elements of R as $Q_{ID_k} \in \mathbb{G}, 1 \leq k \leq d$. Define $\{f_k\}_{k=1}^d$ as

$$f_k = \begin{cases} 1 & \text{if } k = i', \\ 0 & \text{otherwise.} \end{cases}$$

For each $k, 1 \leq k \leq d$, choose random exponents s_k from $\mathbb{Z}_n^*$ and set

$$V_k = (Q_{ID_k}/W)^{f_k} h^{s_k},$$

$$\pi_{k1} = h^{s_k u}, \quad \pi_{k2} = \left((Q_{ID_k}/W)^{2f_k - 1} h^{s_k}\right)^{1/u}, \quad \pi_{k3} = g^{s_k u}, \quad \pi_{k4} = g^{s_k}.$$

Let $s = \sum_{k=1}^d s_k$. Set $G = P_{pub}^s = g^{\alpha s}$.

Let $M = (t, m)$ and $l = |m|$. Recall that β^* is the highest bit of l^* set to 1 and that we are counting up from zero as the least significant bit.

We describe how to create signatures. If $ID_{i'} \notin \mathcal{ID}$, $S_{ID_{i'}}$ could be derived by querying the Extract oracle. Then the challenger could use $S_{ID_{i'}}$ to generate signatures via the normal Sign algorithm. Otherwise, the challenger could simulate signatures as follows.

1. When $t = 1$ and m^* is not a substring of m (Type I Signature Generation):

   Here $m_{i,j}$ denotes the substring m of length j starting at position i. It will help us to first establish the variables $X_{i,j}$, which will be set to 1 if on the selective forgery path and 0 otherwise. For convenience, we use a set of "rules" defining terms and a few observations given in [ABC+12]. Then we describe how the reduction algorithm creates the signatures.

Rules.

For $i = 1$ up to $l + 1$,

For $j = \lfloor \lg(l - i + 1) \rfloor$ down to -1,

(a) If $j + 1 = \beta^*$ and $m_{i-2^{j+1},2^{j+1}} = m^*_{j+1}$, then set $X_{i,j} = 1$.

(b) Else, if $j + 1 < \beta^*$ and the $(j + 1)$th bit of l^* is 1 and $m_{i-2^{j+1},2^{j+1}} = m^*_{j+1}$ and $X_{i-2^{j+1},j+1} = 1$, then set $X_{i,j} = 1$.

(c) Else if $j + 1 < \beta^*$ and the $(j + 1)$th bit of l^* is 0 and $X_{i,j+1} = 1$, then set $X_{i,j} = 1$.

(d) Else set $X_{i,j} = 0$.

Observations. Before we show how $\mathcal{B}$ will simulate the signatures, we make a set of useful observations.

(a) For all i and $j \geq \beta^*$, $X_{i,j} = 0$.

(b) For all $i, X_{i,-1} = 0$. Otherwise, $m_{i-l^*,l^*} = m^*$.

(c) For all i, j, if $X_{i,j} = 1$ and $X_{i,j-1} = 0$, then the jth bit of l^* is 1. If the jth bit were 0, then $X_{i,j-1}$ would have been set to 1 by Rule c.

(d) For all i, j, if $X_{i,j} = 0$ and $X_{i,j-1} = 1$, then the jth bit of l^* is 1. If the jth bit were 0, then the only way to set $X_{i,j-1}$ to 1 would be by Rule c, however, $X_{i,j} = 0$ so Rule c does not apply.

(e) For all i, j, if $X_{i,j} = 1$ and $X_{i+2^j,j-1} = 0$, then $H_a(m_{i,2^j}) = g^{b\theta}$ for some known $\theta \in \mathbb{Z}^*_n$. Otherwise, $X_{i+2^j,j-1}$ would have been set by Rule b to be 1.

(f) For all i, j, if $X_{i,j} = 0$ and $X_{i+2^j,j-1} = 1$, then $H_a(m_{i,2^j}) = g^{b\theta}$ for some known $\theta \in \mathbb{Z}^*_n$. If $X_{i+2^j,j-1} = 1$ and $X_{i,j} = 0$, then $X_{i-2^j,j-1}$ was set to be 1 either by Rule a or Rule c. If it were Rule a, then $j = \beta^*$ and it follows from the programming of the random oracle that $H_a(m_{i,2^j}) = g^{b\theta}$. If it were Rule c, then the jth bit of l^* is 0, meaning m_j cannot be on the selective path and therefore again $H_a(m_{i,2^j}) = g^{b\theta}$.

(g) For all i, j, if $X_{i+2^j,j-1} = 0$, then $H_b(m_{i,2^j}) = g^{b\eta}$ for some known $\eta \in \mathbb{Z}^*_n$. If $j \neq \beta^*$, this follows immediately from the programming of the random oracle. Otherwise, if $j = \beta^*$, then the only way for $X_{i+2^j,j-1} = 0$ would be if $m_\beta \neq m^*_\beta$ by Rule a. Thus, it also follows that $H_b(m_{i,2^j}) = g^{b\eta}$.

Signature Components. Next, for $i = 1$ to $l+1$ and $j = 0$ to $\lfloor \lg(l-i+1) \rfloor$, choose a random $x'_{i,j} \in \mathbb{Z}_n$ and logically set $x_{i,j} = x'_{i,j} + X_{i,j} \cdot (ab\mu_{i'})$, where $\mu_{i'}$ has been used in the H-list corresponding to $ID_{i'}$. For $i = 1$ to $l+1$, set $x_{i,-1} = 0$ (as consistent with Observation b).

A signature is comprised of the following values.

- *Start.* For $i = 1$ to l and $j = 0$ to $\lfloor \lg(l-i+1) \rfloor$.
  (a) If $X_{i+2^j,j-1} = 0$, then it follows by Observation g that $H_b(m_{i,2^j}) = g^{b\eta}$ for some known $\eta \in \mathbb{Z}_n^*$, so choose random $\epsilon_{i,j} \in \mathbb{Z}_n$, implicitly set $r_{i,j} = -a\mu_{i'}/\eta + \epsilon_{i,j}$ and set

$$\begin{aligned} B_{i,j} &= g^{-x'_{i+2^j,j-1}} g^{b\eta\epsilon_{i,j}} U^s \\ &= g^{-x_{i+2^j,j-1}} g^{b\eta(r_{i,j}+a\mu_{i'}/\eta)} h^{as} \\ &= (g^{b\eta})^{r_{i,j}} g^{-x_{i+2^j,j-1}} (g^{b\mu_{i'}})^a h^{as} \\ &= H_b(m_{i,2^j})^{r_{i,j}} g^{-x_{i+2^j,j-1}} S_{ID_{i'}} h^{as}, \\ \widetilde{B}_{i,j} &= g^{-a\mu_{i'}/\eta+\epsilon_{i,j}} = g^{r_{i,j}}. \end{aligned}$$

  (b) Else $X_{i+2^j,j-1} = 1$, so choose random $r_{i,j} \in \mathbb{Z}_n$ and with $x_{i+2^j,j-1} = x'_{i+2^j,j-1} + ab\mu_{i'}$ set

$$\begin{aligned} B_{i,j} &= H_b(m_{i,2^j})^{r_{i,j}} g^{-x'_{i+2^j,j-1}} U^s \\ &= H_b(m_{i,2^j})^{r_{i,j}} g^{-x_{i+2^j,j-1}} (g^{b\mu_{i'}})^a h^{as} \\ &= H_b(m_{i,2^j})^{r_{i,j}} g^{-x_{i+2^j,j-1}} S_{ID_{i'}} h^{as}, \\ \widetilde{B}_{i,j} &= g^{r_{i,j}}. \end{aligned}$$

- *Across.* Together with the following values for $i = 3$ to l and $j = 0$ to $\min(\lfloor \lg(i-1)-1 \rfloor, \lfloor \lg(l-i+1) \rfloor)$.
  (a) If $X_{i,j} = 1$ and $X_{i+2^j,j-1} = 1$, choose random $r'_{i,j} \in \mathbb{Z}_n$ with implicitly set $x_{i,j} = x'_{i,j} + ab\mu_{i'}$ and $x_{i+2^j,j-1} = x'_{i+2^j,j-1} + ab\mu_{i'}$ and set

$$\begin{aligned} A_{i,j} &= H_a(m_{i,2^j})^{r'_{i,j}} g^{x'_{i,j}} g^{-x'_{i+2^j,j-1}} \\ &= H_a(m_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}}, \\ \widetilde{A}_{i,j} &= g^{r'_{i,j}}. \end{aligned}$$

  (b) Else, if $X_{i,j} = 1$ and $X_{i+2^j,j-1} = 0$, then $H_a(m_{i,2^j}) = g^{b\theta}$ for some known $\theta \in \mathbb{Z}_n^*$ by Observation e. Choose random $\epsilon'_{i,j} \in \mathbb{Z}_n$ with

implicitly set $x_{i,j} = x'_{i,j} + ab\mu_{i'}$, $x_{i+2^j,j-1} = x'_{i+2^j,j-1}$ and $r'_{i,j} = -a\mu_{i'}/\theta + \epsilon'_{i,j}$ and set

$$\begin{aligned} A_{i,j} &= g^{x'_{i,j}} g^{-x'_{i+2^j,j-1}} g^{b\theta\epsilon'_{i,j}} \\ &= g^{(x_{i,j}-ab\mu_{i'})} g^{-x_{i+2^j,j-1}} g^{b\theta(r'_{i,j}+a\mu_{i'}/\theta)} \\ &= (g^{b\theta})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}} \\ &= H_a(m_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}}, \\ \widetilde{A}_{i,j} &= g^{-a\mu_{i'}/\theta+\epsilon'_{i,j}} = g^{r'_{i,j}}. \end{aligned}$$

(c) Else, if $X_{i,j} = 0$ and $X_{i+2^j,j-1} = 1$, then $H_a(m_{i,2^j}) = g^{b\theta}$ for some known $\theta \in \mathbb{Z}_n^*$ by Observation f. Choose random $\epsilon'_{i,j} \in \mathbb{Z}_n$ with implicitly set $x_{i,j} = x'_{i,j}$, $x_{i+2^j,j-1} = x'_{i+2^j,j-1} + ab\mu_{i'}$ and $r'_{i,j} = a\mu_{i'}/\theta + \epsilon'_{i,j}$ and set

$$\begin{aligned} A_{i,j} &= g^{x'_{i,j}} g^{-x'_{i+2^j,j-1}} g^{b\theta\epsilon'_{i,j}} \\ &= g^{x_{i,j}} g^{(-x_{i+2^j,j-1}+ab\mu_{i'})} g^{b\theta(r'_{i,j}-a\mu_{i'}/\theta)} \\ &= (g^{b\theta})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}} \\ &= H_a(m_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}}, \\ \widetilde{A}_{i,j} &= g^{a\mu_{i'}/\theta+\epsilon'_{i,j}} = g^{r'_{i,j}}. \end{aligned}$$

(d) Else, $X_{i,j} = 0$ and $X_{i+2^j,j-1} = 0$, so choose random $r'_{i,j} \in \mathbb{Z}_n$ with implicitly set $x_{i,j} = x'_{i,j}$ and $x_{i+2^j,j-1} = x'_{i+2^j,j-1}$ and set

$$\begin{aligned} A_{i,j} &= H_a(m_{i,2^j})^{r'_{i,j}} g^{x'_{i,j}} g^{-x'_{i+2^j,j-1}} \\ &= H_a(m_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}}, \\ \widetilde{A}_{i,j} &= g^{r'_{i,j}}. \end{aligned}$$

- *Down.* Together with the following values for $i = 3$ to $l+1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$.

  (a) If $X_{i,j} = 1$ and $X_{i,j-1} = 1$, choose random $r''_{i,j} \in \mathbb{Z}_n$ with implicitly set $x_{i,j} = x'_{i,j} + ab\mu_{i'}$ and $x_{i,j-1} = x'_{i,j-1} + ab\mu_{i'}$ and set

$$\begin{aligned} D_{i,j} &= g^{x'_{i,j}} g^{-x'_{i,j-1}} g^{z_j r''_{i,j}} = g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j r''_{i,j}}, \\ \widetilde{D}_{i,j} &= g^{r''_{i,j}}. \end{aligned}$$

  (b) Else, if $X_{i,j} = 1$ and $X_{i,j-1} = 0$, then the jth bit of l^* is 1 by Observation c. Thus $z_j = bv_j$, so choose random $\epsilon''_{i,j} \in \mathbb{Z}_n$ with

implicitly set $x_{i,j} = x'_{i,j} + ab\mu_{i'}$, $x_{i,j-1} = x'_{i,j-1}$, and $r''_{i,j} = -a\mu_{i'}/v_j + \epsilon''_{i,j}$ and set

$$\begin{aligned} D_{i,j} &= g^{x'_{i,j}} g^{-x'_{i,j-1}} g^{bv_j\epsilon''_{i,j}} = g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j r''_{i,j}}, \\ \widetilde{D}_{i,j} &= g^{-a\mu_{i'}/v_j + \epsilon''_{i,j}} = g^{r''_{i,j}}. \end{aligned}$$

(c) Else, if $X_{i,j} = 0$ and $X_{i,j-1} = 1$, then the jth bit of l^* is 1 by Observation d. Thus $z_j = bv_j$, so choose random $\epsilon''_{i,j} \in \mathbb{Z}_n$ with implicitly set $x_{i,j} = x'_{i,j}$, $x_{i,j-1} = x'_{i,j-1} + ab\mu_{i'}$, and $r''_{i,j} = a\mu_{i'}/v_j + \epsilon''_{i,j}$ and set

$$\begin{aligned} D_{i,j} &= g^{x'_{i,j}} g^{-x'_{i,j-1}} g^{bv_j\epsilon''_{i,j}} = g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j r''_{i,j}}, \\ \widetilde{D}_{i,j} &= g^{a\mu_{i'}/v_j + \epsilon''_{i,j}} = g^{r''_{i,j}}. \end{aligned}$$

(d) Else, $X_{i,j} = 0$ and $X_{i,j-1} = 0$, so choose random $r''_{i,j} \in \mathbb{Z}_n$ with implicitly set $x_{i,j} = x'_{i,j}$ and $x_{i,j-1} = x'_{i,j-1}$ and set

$$\begin{aligned} D_{i,j} &= g^{x'_{i,j}} g^{-x'_{i,j-1}} g^{z_j r''_{i,j}} = g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j r''_{i,j}}, \\ \widetilde{D}_{i,j} &= g^{r''_{i,j}}. \end{aligned}$$

$\mathcal{B}$ returns $\left(\bar{U}, \widehat{U}, \widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^{d}, G, B_{i,j}, \widetilde{B}_{i,j}, A_{i,j}, \widetilde{A}_{i,j}, D_{i,j}, \widetilde{D}_{i,j}\right)$.

2. When $t = 0$ and $m \neq m^*$ (Type II Signature Generation):

Let $l = |m|$, and $\beta = \lfloor \lg(l) \rfloor$. l^*_i denotes the i-th bit of l^* when we start counting with zero as the least significant bit, and l_i denotes the i-th bit of l.

Parse m^* as $m^*_{\beta^*} m^*_{\beta^*-1} \cdots m^*_0$ where m^*_i is a string of length 2^i or a null string. m^*_i is of length 2^i if $l^*_i = 1$, and is null otherwise. Similarly, Parse m as $m_\beta m_{\beta-1} \cdots m_0$.

$\mathcal{B}$ constructs $(K, \widetilde{B}, Z_{\beta-1}, \cdots, Z_0)$ in the following way.

- If $m_\beta \neq m^*_{\beta^*}$, then $H_b(m_\beta) = g^{b\eta}$ for a η which is known to $\mathcal{B}$.
  (a) $\mathcal{B}$ sets $\widetilde{B} = g^{-a\mu_{i'}/\eta + r}$ for a randomly chosen r and $K = g^{b\eta r} U^s$.
  (b) For $j = \beta - 1$ down to 0, $Z_j = g^{r_j}$ for a randomly chosen r_j, and
    - If $l_j = 1$, then $K = K \cdot H_a(m_j)^{r_j}$.
    - If $l_j = 0$, then $K = K \cdot g^{z_j r_j}$.
- Otherwise, if $\beta = \beta^*$ and $m_\beta = m^*_{\beta^*}$, there exists $j_\phi < \beta$ such that

- $l_{j_\phi} \neq l^*_{j_\phi}$, or
- $l_{j_\phi} = l^*_{j_\phi} = 1$ and $H_a(m_{j_\phi}) \neq H_a(m^*_{j_\phi})$.

so $\mathcal{B}$ can construct a signature $(K, \widetilde{B}, Z_{\beta-1}, \cdots, Z_0)$ in the following way.

(a) $\mathcal{B}$ sets $\widetilde{B} = g^{r_c}$ for a randomly chosen r_c and $K = g^{\eta r_c} U^s$.

(b) For $j = \beta - 1$ down to $j_\phi + 1$ and $j = j_\phi - 1$ to 0, $Z_j = g^{r_j}$ for randomly chosen r_j, and
- If $l_j = 1$, then $K = K \cdot H_a(m_j)^{r_j}$.
- If $l_j = 0$, then $K = K \cdot g^{z_j r_j}$.

(c) For $j = j_\phi$,
- If $l_j = 1$, whether $l^*_j = 0$ or not, $\mathcal{B}$ knows θ such that $H_a(m_j) = g^{b\theta}$. $\mathcal{B}$ sets $Z_j = g^{-a\mu_{i'}/\theta + r_j}$ for a randomly chosen r_j, and $K = K \cdot g^{b\theta r_j}$.
- If $l_j = 0$ and $l^*_j = 1$, then $\mathcal{B}$ knows v such that $g^{z_j} = g^{bv}$. $\mathcal{B}$ sets $Z_j = g^{-a\mu_{i'}/v + r_j}$ for a randomly chosen r_j, and $K = K \cdot g^{bvr_j}$.

$\mathcal{B}$ returns $\left(\widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^d, G, K, \widetilde{B}, Z_{\beta-1}, \cdots, Z_0\right)$.

Response Eventually, $\mathcal{A}$ outputs a valid signature σ^* on $M^* = (t^*, m^*)$ on behalf of R^*. Recall that $l^* = |m^*|$ and $\beta^* = \lfloor \lg(l^*) \rfloor$. Here l^*_i denotes the i-th bit of l^* when we start counting with zero as the least significant bit. Parse m^* as $m^*_{\beta^*} m^*_{\beta^*-1} \cdots m^*_0$ where m^*_i is a string of length 2^i (when $l^*_i = 1$) or a null string (when $l^*_i = 0$). ψ_i denotes the position where m^*_i starts.

Because of the selective disclosure and setup, $\mathcal{B}$ knows the following exponents:

1. η such that $H_b(m^*_{\beta^*}) = g^\eta$.
2. θ_j such that $H_a(m^*_{\psi_j, 2^j}) = g^{\theta_j}$ when $l^*_j = 1$ and $j \neq \beta^*$.
3. z_j when $l^*_j = 0$.

t^* is either 1 or 0. As the challenger is given the factorization of n, this allows the challenger to recover from a signature the values f_k used in generating it. Therefore, the challenger knows the actual signer of σ^* and the corresponding $\mu_{i'}$ in the H-List.

- If $t^* = 1$, $\mathcal{B}$ can compute the information of some $x_{i,j}$ with the following components of σ^*.

– $B_{1,\beta^*} = H_b(m^*_{\beta^*})^{r_{1,\beta^*}} g^{-x_{1+2^{\beta^*},\beta^*-1}} g^{ab\mu_{i'}U^s}$, $\widetilde{B}_{1,\beta^*} = g^{r_{1,\beta^*}}$.
$\mathcal{B}$ knows η such that $H_b(m^*_{\beta^*}) = g^\eta$, so $\mathcal{B}$ can compute $g^{-x_{1+2^{\beta^*},\beta^*-1}} g^{ab\mu_{i'}U^s}$ $= B_{1,\beta^*}/\widetilde{B}^\eta_{1,\beta^*}$.

– For $j = \beta^* - 1$ down to 0,

∗ when $l^*_j = 1, A_{\psi_j,j} = H_a(m^*_j)^{r'_{\psi_j,j}} g^{x_{\psi_j,j}} g^{-x_{\psi_{j-1},j-1}}$, $\widetilde{A}_{\psi_j,j} = g^{r'_{\psi_j,j}}$.
$\mathcal{B}$ knows θ such that $H_a(m^*_j) = g^\theta$, so $\mathcal{B}$ can compute $g^{x_{\psi_j,j}} g^{-x_{\psi_{j-1},j-1}}$ $= A_{\psi_j,j}/\widetilde{A}^\theta_{\psi_j,j}$.

∗ when $l^*_j = 0, D_{\psi_j,j} = g^{x_{\psi_j,j}} g^{-x_{\psi_{j-1},j-1}} g^{z_j r''_{\psi_j,j}}$, $\widetilde{D}_{\psi_j,j} = g^{r''_{\psi_j,j}}$.
$\mathcal{B}$ knows z_j, so $\mathcal{B}$ can compute $g^{x_{\psi_j,j}} g^{-x_{\psi_{j-1},j-1}} = D_{\psi_j,j}/\widetilde{D}^{z_j}_{\psi_j,j}$.

so $\mathcal{B}$ can compute $g^{x_{\psi_j,j}} g^{-x_{\psi_{j-1},j-1}}$.

$\mathcal{B}$ has the values of $g^{-x_{1+2^{\beta^*},\beta^*-1}} g^{ab\mu_{i'}U^s}$ and $g^{x_{\psi_j,j}} g^{-x_{\psi_{j-1},j-1}}$ for $j = \beta^* - 1$ down to 0, so $\mathcal{B}$ can compute $g^{-x_{1+2^{\beta^*},\beta^*-1}} g^{ab\mu_{i'}U^s} \cdot \prod_{j=0}^{\beta^*-1} g^{x_{\psi_j,j}} g^{-x_{\psi_{j-1},j-1}} = g^{ab\mu_{i'}U^s} g^{-x_{s_{-1},-1}} = g^{ab\mu_{i'}U^s}$.

Then, $\mathcal{B}$ can compute $g^{ab} = \left((g^{ab\mu_{i'}U^s})/G^p\right)^{1/\mu_{i'}}$.

- If $t^* = 0$, $\mathcal{B}$ parses σ^* as $(\widetilde{U}, \{V_k, \pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}\}_{k=1}^d, G, K, \widetilde{B}, Z_{\beta^*-1}, \cdots, Z_0)$, with $\widetilde{B} = g^c, Z_{\beta^*-1} = g^{c_{\beta^*-1}}, \cdots, Z_0 = g^{c_0}$ for some $c, c_{\beta^*-1}, \cdots, c_0 \in \mathbb{Z}_n$.

$$K = g^{ab\mu_{i'}U^s} \cdot H_b(m^*_{\beta^*})^c \cdot \prod_{j<\beta, l^*_j=1} H_a(m^*_j)^{c_j} \cdot \prod_{j<\beta, l^*_j=0} (g^{z_j})^{c_j},$$

because the signature σ^* is valid.

$\mathcal{B}$ knows η such that $H_b(m^*_{\beta^*}) = g^\eta$. $\mathcal{B}$ sets $N = \widetilde{B}^\eta$. From $j = \beta^* - 1$ down to 0, $\mathcal{B}$ proceeds as:

– If $l^*_j = 1$, $\mathcal{B}$ knows θ_j such that $H_a(m^*_j) = g^{\theta_j}$. $\mathcal{B}$ sets $N = N \cdot Z_j^{\theta_j}$.

– If $l^*_j = 0$, $\mathcal{B}$ knows z_j. $\mathcal{B}$ sets $N = N \cdot Z_j^{z_j}$.

Then

$$N = H_b(m^*_{\beta^*})^c \prod_{j<\beta, l^*_j=1} H_a(m^*_j)^{c_j} \prod_{j<\beta, l^*_j=0} (g^{z_j})^{c_j},$$

so $\mathcal{B}$ can compute $K/N = g^{ab\mu_{i'}U^s}$.

Then, $\mathcal{B}$ can compute $g^{ab} = \left((g^{ab\mu_{i'}U^s})/G^p\right)^{1/\mu_{i'}}$.

Thus, whether t^* is 1 or 0, $\mathcal{B}$ can solve for g^{ab} and correctly answer to the CDH challenge. □

Theorem 5.4 *This identity-based quotable ring signature scheme is strongly context hiding.*

Proof. Given any two challenge messages $M = (t, m), M' = (t', m')$ such that m' is a substring of m, we claim that whether $t' = 1$ or 0, $\sigma' \leftarrow \text{Quote}(\sigma, R, M, M')$ has an identical distribution to that of $\sigma \leftarrow \text{Sign}(S_{ID_i}, R, M')$, which implies that the following two distributions are statistically close.

$$\{(S_{ID_i}, \sigma \leftarrow \text{Sign}(S_{ID_i}, R, M), \text{Sign}(S_{ID_i}, R, M'))\}_{S_{ID_i}, M, M'}.$$

$$\{(S_{ID_i}, \sigma \leftarrow \text{Sign}(S_{ID_i}, R, M), \text{Quote}(\sigma, R, M, M'))\}_{S_{ID_i}, M, M'}.$$

Let l, l' denote $|m|$ and $|m'|$ respectively. Let $\Gamma = min(\lfloor \lg(i-1) - 1 \rfloor, \lfloor \lg(l - i + 1) \rfloor)$. Signature $\sigma \leftarrow \text{Sign}(S_{ID_i}, R, M)$ on M is composed of the following values.

- $\bar{U} = h^u, \quad \widehat{U} = h^{1/u}, \quad \widetilde{U} = g^u.$
- $V_k = (Q_{ID_k}/W)^{f_k} h^{s_k}, \quad \pi_{k1} = h^{s_k u}, \quad \pi_{k2} = \left((Q_{ID_k}/W)^{2f_k - 1} h^{s_k}\right)^{1/u},$ $\pi_{k3} = g^{s_k u}, \quad \pi_{k4} = g^{s_k}$, for each $1 \le k \le d$.
- $G = P_{pub}^s = g^{\alpha s}$, where $s = \sum_{k=1}^{d} s_k$.
- $B_{i,j} = H_b(m_{i,2^j})^{r_{i,j}} g^{-x_{i+2^j, j-1}} S_{ID} \cdot U^s, \quad \widetilde{B}_{i,j} = g^{r_{i,j}}$, for $i = 1$ to l and $j = 0$ to $\lfloor \lg(l - i + 1) \rfloor$.
- $A_{i,j} = H_a(m_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j, j-1}}, \quad \widetilde{A}_{i,j} = g^{r'_{i,j}}$, for $i = 3$ to l and $j = 0$ to Γ.
- $D_{i,j} = g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j \cdot r''_{i,j}}, \quad \widetilde{D}_{i,j} = g^{r''_{i,j}}$, for $i = 3$ to $l + 1$ and $j = 0$ to $\lfloor \lg(i-1) - 1 \rfloor$.

for randomly chosen $u \in \mathbb{Z}_n^*$ and $r_{i,j}, r'_{i,j}, r''_{i,j}, x_{i,j}, s_k \in \mathbb{Z}_n$.

Type I Signatures. Let $\Gamma' = min(\lfloor \lg(i-1) - 1 \rfloor, \lfloor \lg(l' - i + 1) \rfloor)$. When $t' = 1$, Original Type I signature $\sigma \leftarrow \text{Sign}(S_{ID_i}, R, M')$ on M' is composed of the following values.

- $\bar{U} = h^u, \quad \widehat{U} = h^{1/u}, \quad \widetilde{U} = g^u.$
- $V_k = (Q_{ID_k}/W)^{f_k} h^{s_k}, \quad \pi_{k1} = h^{s_k u}, \quad \pi_{k2} = \left((Q_{ID_k}/W)^{2f_k - 1} h^{s_k}\right)^{1/u},$ $\pi_{k3} = g^{s_k u}, \quad \pi_{k4} = g^{s_k}$, for each $1 \le k \le d$.
- $G = P_{pub}^s = g^{\alpha s}$, where $s = \sum_{k=1}^{d} s_k$.

- $B_{i,j} = H_b(m'_{i,2^j})^{r_{i,j}} g^{-x_{i+2^j,j-1}} S_{ID} \cdot U^s$, $\widetilde{B}_{i,j} = g^{r_{i,j}}$, for $i = 1$ to l' and $j = 0$ to $\lfloor \lg(l' - i + 1) \rfloor$.
- $A_{i,j} = H_a(m'_{i,2^j})^{r'_{i,j}} g^{x_{i,j}} g^{-x_{i+2^j,j-1}}$, $\widetilde{A}_{i,j} = g^{r'_{i,j}}$, for $i = 3$ to l' and $j = 0$ to Γ'.
- $D_{i,j} = g^{x_{i,j}} g^{-x_{i,j-1}} g^{z_j \cdot r''_{i,j}}$, $\widetilde{D}_{i,j} = g^{r''_{i,j}}$, for $i = 3$ to $l' + 1$ and $j = 0$ to $\lfloor \lg(i - 1) - 1 \rfloor$.

for randomly chosen $u \in \mathbb{Z}_n^*$ and $r_{i,j}, r'_{i,j}, r''_{i,j}, x_{i,j}, s_k \in \mathbb{Z}_n$.

Type I quoted signature $\sigma' \leftarrow \text{Quote}(\sigma, R, M, M')$ on M' from original Type I signature σ on M, which is Quote-Type I(σ, R, m, m') is comprised of the following.

- $\bar{U}' = h^{uu'}$, $\widehat{U}' = h^{1/uu'}$, $\widetilde{U}' = g^{uu'}$.
- $V'_k = (Q_{ID_k}/W)^{f_k} h^{(s_k + s'_k)}$,
  $\pi'_{k1} = h^{(s_k+s'_k)uu'}$, $\pi'_{k2} = ((Q_{ID_k}/W)^{2f_k - 1} h^{(s_k+s'_k)})^{1/uu'}$,
  $\pi'_{k3} = g^{(s_k+s'_k)uu'}$, $\pi'_{k4} = g^{(s_k+s'_k)}$, for each $1 \le k \le d$.
- $G' = P_{pub}^{(s+s')} = g^{\alpha(s+s')}$, where $s = \sum_{k=1}^{d} s_k$, $s' = \sum_{k=1}^{d} s'_k$.
- $B'_{i,j} = H_b(m'_{i,2^j})^{(r_{I,j} + t_{i,j})} g^{(-x_{I+2^j,j-1} - y_{i+2^j,j-1})} S_{ID} \cdot U^{(s+s')}$, $\widetilde{B}'_{i,j} = g^{(r_{I,j}+t_{i,j})}$, for $i = 1$ to l' and $j = 0$ to $\lfloor \lg(l' - i + 1) \rfloor$.
- $A'_{i,j} = H_a(m'_{i,2^j})^{(r'_{I,j}+t'_{i,j})} g^{(x_{I,j}+y_{i,j})} g^{(-x_{I+2^j,j-1} - y_{i+2^j,j-1})}$, $\widetilde{A}'_{i,j} = g^{(r'_{I,j}+t'_{i,j})}$, for $i = 3$ to l' and $j = 0$ to $\min(\lfloor \lg(i - 1) - 1 \rfloor, \lfloor \lg(l' - i + 1) \rfloor)$.
- $D'_{i,j} = g^{(x_{I,j}+y_{i,j})} g^{(-x_{I,j-1} - y_{i,j-1})} g^{z_j \cdot (r''_{I,j} + t''_{i,j})}$, $\widetilde{D}'_{i,j} = g^{(r''_{I,j}+t''_{i,j})}$, for $i = 3$ to $l'+1$ and $j = 0$ to $\lfloor \lg(i - 1) - 1 \rfloor$.

for randomly chosen $u' \in \mathbb{Z}_n^*$ and $t_{i,j}, t'_{i,j}, t''_{i,j}, y_{i,j}, s'_k \in \mathbb{Z}_n$, where m' occurs at position δ as a substring of m, $I = i + \delta - 1$.

Since all exponents have been independently re-randomized, one can see by inspection that Quote (σ, R, M, M') has identical distribution as that of Sign(S_{ID_i}, R, M').

Type II Signatures. Parse $m' = m'_{\beta'} m'_{\beta'-1} \cdots m'_0$ where m'_j is of length 2^j or a null string where $\beta' = \lfloor \lg(l') \rfloor$. l'_i denotes the i-th bit of l' when we start counting with zero as the least significant bit. m' occurs at position δ of m. Original Type II signature $\sigma \leftarrow \text{Sign}(S_{ID_i}, R, M')$ on M' composed of the following values, for random $u \in \mathbb{Z}_n^*$, and $s_k, \widehat{u}, \widehat{u}_j \in \mathbb{Z}_n$.

- $\widetilde{U} = g^u$.
- $V_k = (Q_{ID_k}/W)^{f_k} h^{s_k}, \quad \pi_{k1} = h^{s_k u}, \quad \pi_{k2} = \left((Q_{ID_k}/W)^{2f_k - 1} h^{s_k}\right)^{1/u}$, $\pi_{k3} = g^{s_k u}, \quad \pi_{k4} = g^{s_k}$, for each $1 \leq k \leq d$.
- $G = P_{pub}^s = g^{\alpha s}$, where $s = \sum_{k=1}^{d} s_k$.
- $K = g^{ab\mu_i} \cdot H_b(m'_\beta)^{\widehat{u}} \cdot U^s \cdot \prod_{j<\beta, l'_j=1} H_a(m'_j)^{\widehat{u}_j} \cdot \prod_{j<\beta, l'_j=0} g^{z_j \cdot \widehat{u}_j}$.
- $\widetilde{B} = g^{\widehat{u}}$.
- $Z_j = g^{\widehat{u}_j}$.

Let each m'_j start at position ψ_j in m'. Type II quoted signature $\sigma' \leftarrow \text{Quote}(\sigma, R, M, M')$ on M' from original Type I signature σ on M, which is Quote-Type II(σ, R, m, m') is comprised of the following.

- $\widetilde{U}' = g^{uu'}$.
- $V'_k = (Q_{ID_k}/W)^{f_k} h^{(s_k + s'_k)}$, $\pi'_{k1} = h^{(s_k+s'_k)uu'}, \quad \pi'_{k2} = \left((Q_{ID_k}/W)^{2f_k-1} h^{(s_k+s'_k)}\right)^{1/uu'}$, $\pi'_{k3} = g^{(s_k+s'_k)uu'}, \quad \pi'_{k4} = g^{(s_k+s'_k)}$, for each $1 \leq k \leq d$.
- $G' = P_{pub}^{(s+s')} = g^{\alpha(s+s')}$, where $s = \sum_{k=1}^{d} s_k, \quad s' = \sum_{k=1}^{d} s'_k$.
- $K' = g^{ab\mu_i} \cdot H_b(m'_\beta)^{(r_{\delta,\beta}+v)} \cdot U^{(s+s')} \cdot \prod_{j<\beta, l'_j=1} H_a(m'_j)^{\left(r'_{\delta+\psi_j-1,j}+v_j\right)}$ $\cdot \prod_{j<\beta, l'_j=0} g^{z_j (r''_{\delta+\psi_j-1,j}+v_j)}$.
- $\widetilde{B}' = g^{(r_{\delta,\beta}+v)}$.
- $\{Z'_j\}_{j<\beta, l'_j=1} = g^{\left(r'_{\delta+\psi_j-1,j}+v_j\right)}, \quad \{Z'_j\}_{j<\beta, l'_j=0} = g^{\left(r''_{\delta+\psi_j-1,j}+v_j\right)}$.

for randomly chosen $u' \in \mathbb{Z}_n^*$, and $s'_k, v, v_j \in \mathbb{Z}_n$. Since all exponents have been independently re-randomized, one can see by inspection that $\text{Quote}(\sigma, R, M, M')$ has identical distribution as that of $\text{Sign}(S_{ID_i}, R, M')$.

Thus, this identity-based quotable ring signature scheme is strongly context hiding. □

5.7 Summary

We introduced a new notion of identity-based quotable ring signature based on bilinear paring in composite order groups. We extended the ring signature scheme to be quotable. Using this cryptographic primitive, anyone could derive new ring signatures on a substring of an original message from a ring signature on the original message. There are two different types of ring signatures. The first one could be quoted further down to these two types of ring signatures. The other one could not be quoted any further, but will be a shorter signature. We also proved that our scheme is anonymous under the assumption that the Subgroup Decision Problem is hard, selectively unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the Computational Diffie-Hellman problem is hard, and strongly context hiding.

The goal of letting the ring signature to be quotable is achieved at the expense of adding extra non-interactive zero knowledge proof system to encrypt the actual signer's identity.

Chapter 6

Attribute-Based Signature with Message Recovery

In this chapter, we present a new notion called the *attribute-based signature with message recovery*. Compared with the existing attribute-based signature schemes, an attribute-based signature with message recovery scheme does not require the transmission of the original message for the signature verification, since the original message can be recovered from the signature. The contributions of this work are threefold. First, we introduce the notion of attribute-based signature with message recovery. Second, we present a concrete construction of an attribute-based signature with message recovery scheme based on bilinear pairing. Finally, we extend our scheme to deal with large messages. The proposed schemes support flexible threshold predicates and are proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the Computational Diffie-Hellman problem is hard. We demonstrate that the proposed schemes are also equipped with the attribute privacy property. The original scheme was presented at *ISPEC 2014*.

6.1 Introduction

In the signature schemes with message recovery, all or part of the original message is embedded within the signature and can be recovered. Attribute-based signature is an elaborated cryptographic notion that supports fine-grain access control in anonymous authentication systems. A related approach, but much simpler, to attribute-based signature is identity-based signature. Compared with identity-based signature scheme in which a single string representing the signer's identity, in attribute-based signature scheme, a signer who obtains a certificate for a set of attributes from the

attribute authority is defined by the set of attributes. An attribute-based signature attests not to the identity of the individual who signed a message, but assures the verifier that a signer whose set of attributes satisfies a predicate has endorsed the message. In an attribute-based signature, the signature reveals no more than the fact that a single user with some set of attributes satisfying the predicate has attested to the message. In particular, the signature hides the attributes used to satisfy the predicate and any identifying information about the signer. Furthermore, users cannot collude to pool their attributes together.

Compared with the scheme in Chapter 3, these two chapters are both considering digital signature with message recovery. However, there are also differences between the work in these two chapters. In Chapter 3, the signers are revealed to the verifiers by using their identities. In this chapter, the signer is anonymous. The real signer is hidden in the signers whose attributes satisfy a certain predicate.

Related Work. Attribute-based signatures extend the identity-based signature of Shamir [Sha85] by allowing the identity of a signer to be a set of descriptive attributes rather than a single string. As a related notion to attribute-based signature, fuzzy identity-based signature was proposed and formalized in [YCD08], which enables users to generate signatures with part of their attributes. An attribute-based signature was also proposed in [SY08], to achieve almost the same goal. However, these kinds of signatures do not consider the anonymity for signers. Khader [Kha07b, Kha07a] proposed a notion called attribute-based group signatures. This primitive hides the identity of the signer, but reveals which attributes the signer used to satisfy the predicate. It also allows a group manager to identify the signer of any signature. In Khader [Kha08] and Maji et al. [MPR08, MPR11], they treated attribute-privacy as a fundamental requirement of attribute-based signatures.

Maji et al. [MPR08] constructed an attribute-based signature scheme that supports a powerful set of predicates, namely, any predicate consists of AND, OR and Threshold gates. However, their construction is only proved in the generic group model. Li and Kim [LK08] first proposed an attribute-based signature scheme that is secure under the standard CDH assumption. Their scheme only considered (n, n)-threshold, where n is the number of attributes purported in the signature. Shahandashti and Safavi-Naini [SSN09] extended Li and Kim's scheme [LK08] and presented an attribute-based signature scheme supporting (k, n)-threshold. Li et al. [LAS+10] explored a new signing technique integrating all the secret attributes components into one. Their constructions provide better efficiency in terms of both

the computational cost and signature size.

Zhang et al. [ZSM05] presented the seminal construction of an identity-based message recovery signature scheme. Inspired by the schemes due to Zhang et al. [ZSM05] and Li et al. [LAS+10], we propose our attribute-based signature with message recovery scheme.

Comparison. As we have mentioned above, the scheme of Li et al. [LAS+10] have improved schemes of [LK08, SSN09] to provide better efficiency in terms of both the computational cost and signature size. Compared with the scheme of Li et al. [LAS+10] which requires transmission of the original message, our scheme embeds the original message in the signature while keeping the signature size same as that of [LAS+10]. We also note that Gagné et al. [GNSN13] proposed a new threshold attribute-based signature scheme which they claimed the signature size is independent of the number of attributes. However, this result is restricted only to a very special (n, n) threshold scenario. For general attribute policies such as (k, n) threshold scenario, the signature size still grows linearly with the number of attributes used to generate the signature. Furthermore, the scheme of Gagné et al. [GNSN13] only deals with fixed threshold. While our scheme can deal with flexible threshold from 1 to d which is predefined in the setup step.

Our Contributions. In this work, we introduce the notion of attribute-based signature with message recovery. This notion allows fine-grain access control as well as enjoys the shortness of message-signature length. We propose two efficient schemes supporting flexible threshold predicate. The first one embeds short original message in the signature, while keeping the signature size the same as an existing scheme which requires transmission of the original message to verify the signature. For large messages, the second scheme separates the original message to two parts. The signature is appended to a truncated message and the discarded bytes can be recovered by the verification algorithm. The security of our schemes is proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the CDH problem is hard. These schemes are also equipped with attribute-privacy property.

Organization of This Chapter. The rest of this chapter is organized as follows: In Section 6.2, we provide a formal definition of the attribute-based signature with message recovery scheme. In Section 6.3, we present a security model for the new

scheme. In Section 6.4, we present a concrete attribute-based signature with message recovery scheme based on bilinear pairing. Section 6.5 provides the security proofs about existential unforgeability against adaptively chosen message attacks and attribute-privacy property. Section 6.6 extends the first scheme in order to deal with large messages. Section 6.7 concludes this chapter.

6.2 Formal Definitions

We assume there is a universal set of attributes $\mathbb{A}$. Each signer is associated with a subset $\omega \subset \mathbb{A}$ of attributes that is verified by an attribute authority. Our scheme consists of the following four algorithms.

Setup: On input of a security parameter, this algorithm selects the master secret key and generates the corresponding public key.

Extract: When a party requires its attribute private key $\{D_i\}_{i\in\omega}$ corresponding to an attribute set ω, this algorithm generates the attribute private key using the master key and the attributes in ω if he is eligible to be issued with these attributes.

Sign: This scheme supports all predicates $\Upsilon_{t,\bar{\omega}}(\cdot) \rightarrow 0/1$ consisting of t out of n threshold gates, in which $\bar{\omega}$ is an n-element attribute set with threshold value t flexible from 1 to d where $\Upsilon_{t,\bar{\omega}}(\omega) = 1$ when $|\omega \cap \bar{\omega}| \geq t$. On input a message m, a predicate $\Upsilon_{t,\bar{\omega}}(\cdot) \rightarrow 0/1$, and a sender's private key $\{D_i\}_{i\in\omega}$, this algorithm generates a signature σ when $|\omega \cap \bar{\omega}| \geq t$.

Verify: When receiving a signature σ and a predicate $\Upsilon_{t,\bar{\omega}}(\cdot) \rightarrow 0/1$, this algorithm checks whether the signature is valid corresponding to the predicate $\Upsilon_{t,\bar{\omega}}(\cdot) \rightarrow 0/1$. If the signature σ is valid, this algorithm recovers the original message m.

6.3 Security Model

Existential unforgeability against chosen message attacks.
It can be defined using a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

The adversary $\mathcal{A}$ knows the public key of the signer. Its goal is to forge a valid signature of a message m^* with a predicate $\Upsilon_{t,\bar{\omega}}(\cdot) \rightarrow 0/1$ that his attributes do not satisfy.

Firstly, challenger $\mathcal{C}$ runs the Setup algorithm to get the master secret key with respect to a security parameter and the system's public parameters *params*. Then,

$\mathcal{C}$ sends *params* to adversary $\mathcal{A}$. $\mathcal{A}$ can access to some random oracles. In these random oracles, for every unique query, they respond a random response. If a query has been submitted before, they respond the same value as they responded the first time. $\mathcal{A}$ can also access to the following oracles to conduct an attack.

Extract Oracle: For each Extract query with respect to an attribute set ω such that $|\bar{\omega} \cap \omega| < t$, $\mathcal{C}$ returns D_i for each $i \in \omega$ as the private key of attribute set ω.

Sign Oracle: For each Sign query on arbitrary designated attribute set ω and arbitrary message m, $\mathcal{C}$ returns a valid signature σ with respect to m on behalf of the designated signer who possesses the attribute set ω.

Output: $\mathcal{A}$ outputs an alleged signature σ^* on the message m^* on behalf of a user who possesses an attribute set ω^* such that $|\bar{\omega} \cap \omega^*| \geq t$. If no Sign queries of the message m^* with an attribute set ω such that $|\bar{\omega} \cap \omega| \geq t$ and no Extract queries with respect to an attribute set ω such that $|\bar{\omega} \cap \omega| \geq t$ have been queried, $\mathcal{A}$ wins the game if the signature σ^* is valid.

If there is no such polynomial-time adversary $\mathcal{A}$ that can forge a valid signature in the game described above, we say this scheme is secure against existential forgery under chosen message attacks.

It is worth noting that this model also guarantees *collusion resistance*. This is because if a group of signers can cooperate to construct a signature that none of them could individually produce, then they can build another adversary which can forge a valid signature to win the above game.

Attribute privacy.

In an attribute-based signature scheme, a legitimate signer is indistinguishable among all the users whose attributes satisfying the predicate specified in the signature. The signature reveals nothing about the identity or attributes of the signer beyond what is explicitly revealed by the claim being made.

It can be defined using a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

The adversary $\mathcal{A}$ even knows the master secret key. So he could generate all signers' private keys. Its goal is to distinguish between two signers which one generates the valid signature of a message with a predicate such that both of their attributes satisfy the predicate.

Firstly, challenger $\mathcal{C}$ runs the Setup algorithm to get the master secret key and the public parameters *params*. Then, $\mathcal{C}$ sends *params* as well as the master secret key to adversary $\mathcal{A}$. $\mathcal{A}$ can access polynomially bounded number of random oracles

which are the same as that described in the previous game. $\mathcal{A}$ can generate private keys and signatures itself, because he has got the master secret key.

Challenge: $\mathcal{A}$ outputs a message m^*, two challenged attribute sets ω_0^*, ω_1^* for signature query, where both ω_0^* and ω_1^* satisfy a predicate Υ^*. $\mathcal{C}$ chooses $b \in \{0,1\}$, computes the challenge signature σ^* satisfying Υ^* on behalf of the signer who possesses attribute set ω_b^* and provides σ^* to $\mathcal{A}$.

Guess: $\mathcal{A}$ tries to guess which attribute set between ω_0^* and ω_1^* is used to generate the challenge signature σ^*. Finally, $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ and wins the game if $b' = b$.

If there is no such polynomial-time adversary $\mathcal{A}$ that can win the game described above, we say this scheme holds attribute privacy property.

It is worth noting that this property holds even for the attribute authority, because the master secret key is also given to the adversary.

6.4 Proposed Scheme

Setup: First, define the attributes in universe $\mathbb{A}$ as elements in $\mathbb{Z}_p^*$ where p is a sufficient large prime. A $(d-1)$ default attribute set from $\mathbb{Z}_p^*$ is given as $\Omega = \{\Omega_1, \Omega_2, \cdots, \Omega_{d-1}\}$. Select a random generator $g \in \mathbb{G}_1$, a random $x \in \mathbb{Z}_p^*$, and set $g_1 = g^x$. Next, pick a random element $g_2 \in \mathbb{G}_1$. Five hash functions are also chosen such that $H_1 : \mathbb{Z}_p^* \to \mathbb{G}_1$, $H_2 : \{0,1\}^* \to \mathbb{Z}_p^*$, $H_3 : \{0,1\}^* \to \mathbb{G}_1$, $F_1 : \{0,1\}^{k_2} \to \{0,1\}^{k_1}$, $F_2 : \{0,1\}^{k_1} \to \{0,1\}^{k_2}$. The public parameters are $params = (g, g_1, g_2, d, H_1, H_2, H_3, F_1, F_2)$, the master secret key is x.

Extract: To generate private key for an attribute set ω,

- First, randomly choose a $(d-1)$ degree polynomial $q(z)$ such that $q(0) = x$;
- Generate a new attribute set $\hat{\omega} = \omega \cup \Omega$. For each $i \in \hat{\omega}$, choose $r_i \in_R \mathbb{Z}_p$ and compute $d_{i0} = g_2^{q(i)} \cdot H_1(i)^{r_i}$ and $d_{i1} = g^{r_i}$;
- Finally, output $D_i = (d_{i0}, d_{i1})$ as the private key for each $i \in \hat{\omega}$.

Sign: Suppose one has private key for the attribute set ω. To sign a message m which length is equal to k_2 with predicate $\Upsilon_{t,\bar{\omega}}(\cdot)$, namely, to prove owning at least t attributes among an n-element attribute set $\bar{\omega}$, he selects a t-element subset $\omega' \subseteq \omega \cap \bar{\omega}$ and selects randomly an element j from subset $\bar{\omega}/\omega'$, and proceeds as follows:

- First, select a default attribute subset $\Omega' \subseteq \Omega$ with $|\Omega'| = d - t$ and choose $(n + d - t - 1)$ random values $r'_i \in \mathbb{Z}_p^*$ for $i \in (\bar{\omega}/j) \cup \Omega'$, choose a random value $s \in \mathbb{Z}_p^*$;
- Compute $v = e(g_1, g_2)$;
- Compute $f = F_1(m)||(F_2(F_1(m)) \oplus m)$;
- Compute $r = H_2(v) + f$;
- Compute $\sigma_i = d_{i1}^{\Delta_{i,S}(0)} \cdot g^{r'_i}$ for $i \in \omega' \cup \Omega'$;
- Compute $\sigma_i = g^{r'_i}$ for $i \in \bar{\omega}/(\omega' \cup j)$;
- Compute $\sigma_j = g^s$;
- Compute $\sigma_0 = \left[\prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right] \cdot \left[\prod_{i \in (\bar{\omega} \cup \Omega')/j} H_1(i)^{r'_i}\right] \cdot \left(H_1(j) \cdot H_3(r, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)\right)^s$;
- Finally, output the signature $\sigma = (r, \sigma_0, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)$.

To sign a message m which length is shorter than k_2, one can just pad spaces after the message until k_2.

Verify: To verify the validity of a signature $\sigma = (r, \sigma_0, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)$ with threshold t for attributes $\bar{\omega}$, the verifier performs the following verification procedure to recover the message m:

$$\frac{e\left(g, \sigma_0\right)}{\prod_{i \in \bar{\omega} \cup \Omega'} e\left(H_1(i), \sigma_i\right) \cdot e\left(\sigma_j, H_3\left(r, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j\right)\right)} = v,$$

$$r - H_2(v) = f.$$

Then, $m = [f]_{k_2} \oplus F_2([f]^{k_1})$ is recovered from f. The verifier checks whether the equation $[f]^{k_1} = F_1(m)$ holds. If it holds, output accept and the message m is recovered. Otherwise, output reject to denote the signature is not valid.

In the above computation, the subscript k_2 of f denotes the least significant k_2 bits of f, and the superscript k_1 of f denotes the most significant k_1 bits of f.

6.5 Security Analysis

Theorem 6.1 *This attribute-based signature with message recovery scheme is correct.*

Proof. The correctness of this scheme can be justified as follows:

$$
\begin{aligned}
& \frac{e\left(g, \sigma_0\right)}{\prod_{i \in \bar{\omega} \cup \Omega'} e\left(H_1(i), \sigma_i\right) \cdot e\left(\sigma_j, H_3\left(r, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j\right)\right)} \\
= & \frac{e\left(g, \left[\prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right] \cdot \left[\prod_{i \in (\bar{\omega} \cup \Omega')/j} H_1(i)^{r_i'}\right] \cdot H_1(j)^s\right)}{\prod_{i \in \bar{\omega} \cup \Omega'} e\left(H_1(i), \sigma_i\right)} \\
= & \frac{e\left(g, \left[\prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right] \cdot \left[\prod_{i \in (\bar{\omega} \cup \Omega')/j} H_1(i)^{r_i'}\right] \cdot H_1(j)^s\right)}{\prod_{i \in \omega' \cup \Omega'} e\left(H_1(i), d_{i1}^{\Delta_{i,S}(0)} \cdot g^{r_i'}\right) \cdot \prod_{i \in \bar{\omega}/(\omega' \cup j)} e\left(H_1(i), g^{r_i'}\right) \cdot e\left(H_1(j), g^s\right)} \\
= & \frac{e\left(g, \prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right) \cdot e\left(g, \prod_{i \in (\bar{\omega} \cup \Omega')/j} H_1(i)^{r_i'}\right)}{\prod_{i \in \omega' \cup \Omega'} e\left(H_1(i), d_{i1}^{\Delta_{i,S}(0)} \cdot g^{r_i'}\right) \cdot \prod_{i \in \bar{\omega}/(\omega' \cup j)} e\left(H_1(i), g^{r_i'}\right)} \\
= & \frac{e\left(g, \prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right) \cdot e\left(g, \prod_{i \in \omega' \cup \Omega'} H_1(i)^{r_i'}\right) \cdot e\left(g, \prod_{i \in \bar{\omega}/(\omega' \cup j)} H_1(i)^{r_i'}\right)}{\prod_{i \in \omega' \cup \Omega'} e\left(H_1(i), d_{i1}^{\Delta_{i,S}(0)} \cdot g^{r_i'}\right) \cdot \prod_{i \in \bar{\omega}/(\omega' \cup j)} e\left(H_1(i), g^{r_i'}\right)} \\
= & \frac{e\left(g, \prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right) \cdot e\left(g, \prod_{i \in \omega' \cup \Omega'} H_1(i)^{r_i'}\right)}{\prod_{i \in \omega' \cup \Omega'} e\left(H_1(i), d_{i1}^{\Delta_{i,S}(0)} \cdot g^{r_i'}\right)} \\
= & \frac{e\left(g, \prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right)}{\prod_{i \in \omega' \cup \Omega'} e\left(H_1(i), d_{i1}^{\Delta_{i,S}(0)}\right)} \\
= & \frac{e\left(g, \prod_{i \in \omega' \cup \Omega'} \left(g_2^{q(i)} \cdot H_1(i)^{r_i}\right)^{\Delta_{i,S}(0)}\right)}{\prod_{i \in \omega' \cup \Omega'} e\left(H_1(i), (g^{r_i})^{\Delta_{i,S}(0)}\right)} \\
= & e\left(g, g_2^{\sum_{i \in \omega' \cup \Omega'} q(i) \Delta_{i,S}(0)}\right) \\
= & v,
\end{aligned}
$$

Then, using this v and r, we can recover f from the following computation.

$$r - H(v) = f.$$

Since f is computed from $f = F_1(m)||(F_2(F_1(m)) \oplus m)$, the original message m will be recovered from f like this:

$$
\begin{aligned}
& [f]_{k_2} \oplus F_2([f]^{k_1}) \\
= & [F_1(m)||(F_2(F_1(m)) \oplus m)]_{k_2} \oplus F_2([F_1(m)||(F_2(F_1(m)) \oplus m)]^{k_1})
\end{aligned}
$$

$$= F_2(F_1(m)) \oplus m \oplus F_2(F_1(m))$$
$$= m.$$

Theorem 6.2 *This attribute-based signature with message recovery scheme is existentially unforgeable under chosen message attacks in the random oracle model, under the assumption that the CDH problem is hard.*

Proof. Assume there is an algorithm $\mathcal{A}$ that can forge a valid signature under chosen message attacks. There will be another algorithm $\mathcal{B}$ that can run the algorithm $\mathcal{A}$ as a subroutine to solve the CDH problem.

We assume that the instance of the CDH problem consists of group elements $(g, g^x, g^y) \in \mathbb{G}_1^3$, and our goal is to compute an element $g^{xy} \in \mathbb{G}_1$.

Setup: Let the default attribute set be $\Omega = \{\Omega_1, \Omega_2, \cdots, \Omega_{d-1}\}$. Since the threshold in our scheme is flexible from 1 to d, without loss of generality, we fix the threshold to $t \leq d$ in this proof. Firstly, $\mathcal{B}$ selects randomly a subset $\Omega' \subseteq \Omega$ with $|\Omega'| = d - t$. $\mathcal{B}$ selects g as the generator of $\mathbb{G}_1$, and sets $g_1 = g^x$ and $g_2 = g^y$. $\mathcal{B}$ sends public system parameters *params* to adversary $\mathcal{A}$.

H_1 Queries: $\mathcal{B}$ creates and keeps one list H_1-List to simulate H_1 Oracle. This list is used to store tuples like $(i, \alpha_i, H_1(i))$. In this type of tuples, the first element $i \in \mathbb{Z}_p^*$ indicates an attribute. The second element α_i is a random number in $\mathbb{Z}_p^*$. The last element $H_1(i)$ is a random element selected from $\mathbb{G}_1$.

Upon receiving an H_1 hash query with respect to an attribute i, if this i is not included in this H_1-List and $i \in \bar{\omega} \cup \Omega'$, $\mathcal{B}$ randomly selects a number $\alpha_i \in \mathbb{Z}_q^*$ and returns $H_1(i) = g^{\alpha_i}$ as the H_1 hash value of this i. Then, $\mathcal{B}$ records the tuple $(i, \alpha_i, g^{\alpha_i})$ in this H_1-List. If this i is not included in this H_1-List and $i \notin (\bar{\omega} \cup \Omega')$, $\mathcal{B}$ randomly selects a number $\alpha_i \in \mathbb{Z}_q^*$ and returns $H_1(i) = g_1^{-\alpha_i}$ as the H_1 hash value of this i. Then, $\mathcal{B}$ records the tuple $(i, \alpha_i, g_1^{-\alpha_i})$ in this H_1-List. If the i is already in a record in this H_1-List, $\mathcal{B}$ only returns the corresponding $H_1(i)$ in the record as the H_1 hash value.

H_3 Queries: $\mathcal{B}$ creates and keeps one list H_3-List to simulate H_3 Oracle. This list is used to store tuples like

$$\left((r, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j), \quad \beta, \quad H_3(r, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)\right).$$

In this type of tuples, the first tuple $(r, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)$ includes an element in $\mathbb{Z}_p^*$ and some other elements in $\mathbb{G}_1$. The second element β is a random number in $\mathbb{Z}_p^*$. The last element $H_3(r, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)$ is a random element selected from $\mathbb{G}_1$.

Part of the records in this H_3-List corresponding to the queries which are queried by the adversary $\mathcal{A}$. We will discuss this situation soon. The other part of the records in this H_3-List corresponding to the queries which are conducted by the simulator $\mathcal{B}$ when $\mathcal{B}$ responds to the Sign queries. We will postpone to discuss this situation in the Sign Queries.

Upon receiving the k-th H_3 hash query which is conducted by the adversary $\mathcal{A}$ with respect to a tuple $(r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j)_k$, if this tuple is not included in this H_3-List, $\mathcal{B}$ randomly selects a number $\beta_k \in \mathbb{Z}_q^*$ and returns $H_3((r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j)_k) = g^{\beta_k}$ as the H_3 hash value of this tuple. $\mathcal{B}$ records $((r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j)_k, \beta_k, g^{\beta_k})$ in this H_3-List. If the tuple is already in a record in this H_3-List, $\mathcal{B}$ only returns the corresponding $H_3((r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j)_k)$ in the record as the H_3 hash value.

Extract Queries: $\mathcal{A}$ can make requests for private keys of attribute set ω such that $|\bar{\omega}\cap\omega| < t$. First define three sets Γ, Γ', S in the following manner: $\Gamma = (\omega\cap\bar{\omega})\cup\Omega'$, Γ' such that $\Gamma \subseteq \Gamma' \subseteq S$ and $|\Gamma'| = d-1$, and $S = \Gamma' \cup \{0\}$.

Similar to the case in the normal scheme, $\mathcal{B}$ should randomly choose a $(d-1)$ degree polynomial $q(z)$ such that $q(0) = x$. We will show how $\mathcal{B}$ simulate private keys for attribute sets although $\mathcal{B}$ does not know exactly the value of x.

For $i \in \Gamma'$, $\mathcal{B}$ randomly selects two numbers $\tau_i, r_i \in \mathbb{Z}_p^*$. In this case, $\mathcal{B}$ assumes the value $q(i)$ corresponding to this i of the randomly chosen $(d-1)$ degree polynomial $q(z)$ is $q(i) = \tau_i$. Then, $\mathcal{B}$ can compute D_i for $i \in \Gamma'$ as follows:

$$D_i = (g_2^{q(i)} \cdot H_1(i)^{r_i}, g^{r_i}) = (g_2^{\tau_i} \cdot H_1(i)^{r_i}, g^{r_i}).$$

For $i \notin \Gamma'$, $\mathcal{B}$ looks up the H_1-List which is created by H_1 Oracle to find the record about attribute i and get the corresponding α_i. $\mathcal{B}$ randomly selects a number $r_i' \in \mathbb{Z}_p^*$, and let

$$r_i = \frac{\Delta_{0,S}(i)}{\alpha_i} y + r_i'.$$

We will show how $\mathcal{B}$ simulate private keys for attribute $i \notin \Gamma'$ although $\mathcal{B}$ does not know exactly the value of y. In case of the values $q(i)$ for $i \in \Gamma'$ are determined in the previous stage, $\mathcal{B}$ can compute the value $q(i)$ corresponding to $i \notin \Gamma'$ of the randomly chosen $(d-1)$ degree polynomial $q(z)$ by using Lagrange interpolation as

$$q(i) = \sum_{j\in\Gamma'} \Delta_{j,S}(i) \cdot q(j) + \Delta_{0,S}(i) \cdot q(0),$$

in which $q(0) = x$. Then, $\mathcal{B}$ can compute D_i for $i \notin \Gamma'$ as follows:

$$\begin{aligned} D_i &= (g_2^{q(i)} \cdot H_1(i)^{r_i}, g^{r_i}) \\ &= (g_2^{\sum_{j\in\Gamma'} \Delta_{j,S}(i)\cdot q(j)} \cdot (g_1^{-\alpha_i})^{r'_i}, g_2^{\frac{\Delta_{0,S}(i)}{\alpha_i}} \cdot g^{r'_i}), \end{aligned}$$

although $\mathcal{B}$ does not know exactly the value of x and y.

$\mathcal{B}$ returns D_i for each $i \in (\omega \cup \Omega)$ as the private key of ω.

Sign Queries: For a Sign query on message m with respect to an attribute set ω. If $|\bar{\omega} \cap \omega| < t$, $\mathcal{B}$ can get a simulated private key with respect to ω by querying the Extract Oracle, and compute a signature on message m with respect to ω normally.

If $|\bar{\omega} \cap \omega| \geq t$, $\mathcal{B}$ selects a t-element subset $\omega' \subseteq \bar{\omega} \cap \omega$ and selects randomly an element j from subset $\bar{\omega}/\omega'$, and simulates the signature as follows:

Firstly, $\mathcal{B}$ selects a random $(d-t)$-element subset Ω' from Ω. Then, $\mathcal{B}$ chooses two random numbers r_i and r''_i for each $i \in \omega' \cup \Omega'$, and let $r'_i = r_i \cdot \Delta_{i,S}(0) + r''_i$. It is obviously that r'_i is still a random number for each $i \in \omega' \cup \Omega'$. $\mathcal{B}$ also chooses random number r'_i for each $i \in \bar{\omega}/(\omega' \cup j)$. $\mathcal{B}$ also chooses two random values $\beta_h, s' \in \mathbb{Z}_p^*$ and let $s = \frac{1}{\beta_h} y + s'$ which is also a random number because β_h and s' are randomly chosen. We will show how $\mathcal{B}$ simulate a correct signature although $\mathcal{B}$ does not know exactly the value of y.

Firstly, $\mathcal{B}$ computes the following parts by using previous parameters as in the normal scheme:

- Compute $v = e(g_1, g_2)$;
- Compute $f = F_1(m)||(F_2(F_1(m)) \oplus m)$;
- Compute $r = H_2(v) + f$;
- Compute $\sigma_i = (g^{r_i})^{\Delta_{i,S}(0)} \cdot g^{r''_i} = g^{r_i \cdot \Delta_{i,S}(0) + r''_i} = g^{r'_i}$ for $i \in \omega' \cup \Omega'$;
- Compute $\sigma_i = g^{r'_i}$ for $i \in \bar{\omega}/(\omega' \cup j)$;
- Compute $\sigma_j = g^s = g^{\frac{1}{\beta_h} y + s'} = g_2^{\frac{1}{\beta_h}} \cdot g^{s'}$;

After this computation, $\mathcal{B}$ inserts a record $((r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j), \beta_h, g_1^{-\beta_h})$ in the H_3-List. Then, $\mathcal{B}$ computes σ_0 as follows:

$$\sigma_0 = g_2^x \cdot \left[\prod_{i\in(\bar{\omega}\cup\Omega')/j} H_1(i)^{r'_i} \right] \cdot H_1(j)^s \cdot H_3(r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j)^s$$

$$
\begin{aligned}
&= g_2^x \cdot H_3(r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j)^s \cdot \left[\prod_{i\in(\bar{\omega}\cup\Omega')/j} H_1(i)^{r'_i}\right] \cdot H_1(j)^s \\
&= g_2^x \cdot (g_1^{-\beta_h})^{\frac{1}{\beta_h}y+s'} \cdot \left[\prod_{i\in(\bar{\omega}\cup\Omega')/j} H_1(i)^{r'_i}\right] \cdot (g^{\alpha_j})^{\frac{1}{\beta_h}y+s'} \\
&= g_2^x \cdot g_1^{-y} \cdot g_1^{-\beta_h s'} \cdot \left[\prod_{i\in(\bar{\omega}\cup\Omega')/j} H_1(i)^{r'_i}\right] \cdot (g_2^{\frac{\alpha_j}{\beta_h}} \cdot g^{\alpha_j s'}) \\
&= g_1^{-\beta_h s'} \cdot (g_2^{\frac{\alpha_j}{\beta_h}} \cdot g^{\alpha_j s'}) \cdot \left[\prod_{i\in(\bar{\omega}\cup\Omega')/j} H_1(i)^{r'_i}\right].
\end{aligned}
$$

We will show this simulated σ_0 have the same form as in the normal scheme as follows:

$$
\begin{aligned}
\sigma_0 &= g_2^x \cdot \left[\prod_{i\in(\bar{\omega}\cup\Omega')/j} H_1(i)^{r'_i}\right] \cdot (H_1(j) \cdot H_3(r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j))^s \\
&= g_2^{\sum_{i\in\omega'\cup\Omega'} q(i)\cdot\Delta_{i,S}(0)} \cdot \prod_{i\in\omega'\cup\Omega'} H_1(i)^{r'_i} \cdot \prod_{i\in\bar{\omega}/(\omega'\cup j)} H_1(i)^{r'_i} \\
&\quad \cdot (H_1(j) \cdot H_3(r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j))^s \\
&= \prod_{i\in\omega'\cup\Omega'} g_2^{q(i)\cdot\Delta_{i,S}(0)} \cdot \prod_{i\in\omega'\cup\Omega'} H_1(i)^{r_i\cdot\Delta_{i,S}(0)} \cdot \prod_{i\in\omega'\cup\Omega'} H_1(i)^{r''_i} \\
&\quad \cdot \prod_{i\in\bar{\omega}/(\omega'\cup j)} H_1(i)^{r'_i} \cdot (H_1(j) \cdot H_3(r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j))^s \\
&= \prod_{i\in\omega'\cup\Omega'} (g_2^{q(i)} \cdot H_1(i)^{r_i})^{\Delta_{i,S}(0)} \cdot \prod_{i\in\omega'\cup\Omega'} H_1(i)^{r''_i} \cdot \prod_{i\in\bar{\omega}/(\omega'\cup j)} H_1(i)^{r'_i} \\
&\quad \cdot (H_1(j) \cdot H_3(r, \{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j}, \sigma_j))^s.
\end{aligned}
$$

Compared with a signature generated from the normal scheme, we will find out that this simulated signature can be regarded as a normal signature which is generated by a signer who possesses private keys $D_i = (g_2^{q(i)} \cdot H_1(i)^{r_i}, g^{r_i})$ for attribute $i \in \omega' \cup \Omega'$ in which $q(z)$ is a random $(d-1)$ degree polynomial such that $q(0) = x$. It is worth noting that although r''_i and r'_i are not the same form at the first glance, they are indeed the same form because both of r''_i and r'_i are random numbers. So the two parts $\prod_{i\in\omega'\cup\Omega'} H_1(i)^{r''_i}$ and $\prod_{i\in\bar{\omega}/(\omega'\cup j)} H_1(i)^{r'_i}$ can be merged into one part as $\prod_{i\in(\bar{\omega}\cup\Omega')/j} H_1(i)^{r'_i}$ in the normal scheme.

Verify: While $|\bar{\omega} \cap \omega| < t$, the simulated signature on message m with respect to ω is computed by querying the Extract Oracle to get a simulated private key with

respect to ω normally. It will certainly pass the normal verification process. While $|\bar{\omega} \cap \omega| \geq t$, we can check that the simulated signature can also pass the normal verification process by straight-forward substitutions.

Finally, The adversary outputs a forged signature σ^* on message m^* for attribute set ω^* such that $|\bar{\omega} \cap \omega^*| \geq t$. It satisfies the verification equation, which means that

$$\sigma^* = \{r^*, g_2^x \cdot \left[\prod_{i \in (\bar{\omega} \cup \Omega')/j} H_1(i)^{r'_i}\right] \cdot H_1(j)^s \cdot H_3(r^*, \{\sigma_i^*\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j^*)^s,$$

$$\{g^{r'_i}\}_{i \in (\bar{\omega} \cup \Omega')/j}, g^s\}.$$

Then, $\mathcal{B}$ can compute

$$\frac{\sigma_0^*}{\prod_{i \in (\bar{\omega} \cup \Omega')/j} (\sigma_i^*)^{\alpha_i} \cdot (\sigma_j^*)^{\alpha_j} \cdot (\sigma_j^*)^{\beta_k}} = g^{xy}.$$

So, $\mathcal{B}$ can solve an CDH problem if $\mathcal{A}$ is able to forge valid signatures.

If there is no such polynomial-time adversary that can forge a valid attribute-based signature with a predicate that his attributes do not satisfy, we say that this attribute-based signature with message recovery scheme is secure against existential forgery under chosen message attacks. □

Theorem 6.3 *This attribute-based signature with message recovery scheme is equipped with the attribute privacy property in the random oracle model.*

Proof. **Setup**: First, a $(d-1)$ default attribute set from $\mathbb{Z}_p^*$ is given as $\Omega = \{\Omega_1, \Omega_2, \cdots, \Omega_{d-1}\}$ for some predefined integer d. $\mathcal{C}$ selects a random generator $g \in \mathbb{G}_1$, a random $x \in \mathbb{Z}_p^*$ as the master secret key, and set $g_1 = g^x$. Next, $\mathcal{C}$ picks a random element $g_2 \in \mathbb{G}_1$. $\mathcal{C}$ sends these public parameters *params* as well as the master secret key x to adversary $\mathcal{A}$.

Both of the $\mathbf{H}_1$ oracle and $\mathbf{H}_3$ oracle are the same as described in Theorem 6.2. **Challenge**: The adversary outputs two attribute sets ω_0^* and ω_1^*. Both the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ can generate secret keys corresponding to these two attribute sets as D_i^0 for $i \in \omega_0^* \cup \Omega$ and D_i^1 for $i \in \omega_1^* \cup \Omega$ respectively. Then, the adversary outputs a message m^* and a predicate Υ^* where $\Upsilon^*(\omega_0^*) = \Upsilon^*(\omega_1^*) = 1$. The adversary $\mathcal{A}$ asks the challenger to generate a signature on the message m^* satisfying Υ^* from either ω_0^* or ω_1^*. The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and outputs a signature $\sigma^* = \{r^*, \sigma_0^*, \{\sigma_i^*\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j^*\}$ by running algorithm which is described as the Sign oracle in Theorem 6.2 using the secret key D_i^b for $i \in \omega_b^* \cup \Omega$.

As we have mentioned in Theorem 6.2, part of the signature σ_0^* can be written as $\prod_{i\in\omega^*\cup\Omega'}(g_2^{q(i)}\cdot H_1(i)^{r_i})^{\Delta_{i,S}(0)}\cdot\prod_{i\in\omega^*\cup\Omega'}H_1(i)^{r_i''}\cdot\prod_{i\in\bar{\omega}/(\omega^*\cup j)}H_1(i)^{r_i'}\cdot(H_1(j)\cdot H_3(r,\{\sigma_i\}_{i\in(\bar{\omega}\cup\Omega')/j},\sigma_j))^s$, σ_i for $i\in\omega^*\cup\Omega'$ can be written as $\sigma_i=(g^{r_i})^{\Delta_{i,S}(0)}\cdot g^{r_i''}=g^{r_i\cdot\Delta_{i,S}(0)+r_i''}=g^{r_i'}$.

So, the challenge signature can be regarded as generated by a signer who possesses private keys $D_i=(g_2^{q(i)}\cdot H_1(i)^{r_i},g^{r_i})$ in which $q(z)$ is a random $(d-1)$ degree polynomial such that $q(0)=x$. Thus, if this challenge signature is generated by using the secret key D_i^0 for $i\in\omega_0^*\cup\Omega$, it can also be generated by using the secret key D_i^1 for $i\in\omega_1^*\cup\Omega$ since the secret key D_i^1 for $i\in\omega_1^*\cup\Omega$ also satisfy the situation mentioned above. Similarly, if this challenge signature is generated by using the secret key D_i^1 for $i\in\omega_1^*\cup\Omega$, it can also be generated by using the secret key D_i^0 for $i\in\omega_0^*\cup\Omega$.

Therefore, even the adversary has access to the master secret key and has unbounded computation ability, he cannot distinguish between two signers which one generates a valid signature of a message with a predicate such that both of their attributes satisfy the predicate. □

6.6 Extended Scheme

In order to deal with messages which are larger than k_2, we can extend the previous scheme as follows.

Setup: The Setup algorithm is the same as that in the previous scheme.

Extract: The Extract algorithm is also the same as that in the previous scheme.

Sign: Suppose one has private key for the attribute set ω. To sign a message m which length is larger than k_2 with predicate $\Upsilon_{t,\bar{\omega}}(\cdot)$, namely, to prove owning at least t attributes among an n-element attribute set $\bar{\omega}$, he selects a t-element subset $\omega'\subseteq\omega\cap\bar{\omega}$ and selects randomly an element j from subset $\bar{\omega}/\omega'$, and proceeds as follows:

- First, separate the message m into two parts $m=m_1||m_2$, and let the length of m_1 be k_2.
- Select a default attribute subset $\Omega'\subseteq\Omega$ with $|\Omega'|=d-t$ and choose $(n+d-t-1)$ random values $r_i'\in\mathbb{Z}_p^*$ for $i\in(\bar{\omega}/j)\cup\Omega'$, choose a random value $s\in\mathbb{Z}_p^*$;

- Compute $v = e(g_1, g_2)$;
- Compute $f = F_1(m_1)||(F_2(F_1(m_1)) \oplus m_1)$;
- Compute $r = H_2(v) + f$;
- Compute $c = H_2(r, m_2)$;
- Compute $\sigma_i = d_{i1}^{\Delta_{i,S}(0)} \cdot g^{r'_i}$ for $i \in \omega' \cup \Omega'$;
- Compute $\sigma_i = g^{r'_i}$ for $i \in \bar{\omega}/(\omega' \cup j)$;
- Compute $\sigma_j = g^s$;
- Compute $\sigma_0 = \left[\prod_{i \in \omega' \cup \Omega'} d_{i0}^{\Delta_{i,S}(0)}\right] \cdot \left[\prod_{i \in (\bar{\omega} \cup \Omega')/j} H_1(i)^{r'_i}\right]$ $\cdot \left(H_1(j) \cdot H_3(c, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)\right)^s$;
- Finally, output the signature $\sigma = (m_2, r, \sigma_0, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)$.

Verify: To verify the validity of a signature $\sigma = (m_2, r, \sigma_0, \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j)$ with threshold t for attributes $\bar{\omega}$, the verifier performs the following verification procedure to recover the message m_1:

$$\frac{e(g, \sigma_0)}{\prod_{i \in \bar{\omega} \cup \Omega'} e(H_1(i), \sigma_i) \cdot e\left(\sigma_j, H_3\left(H_2(r, m_2), \{\sigma_i\}_{i \in (\bar{\omega} \cup \Omega')/j}, \sigma_j\right)\right)} = v,$$

$$r - H_2(v) = f.$$

Then, $m_1 = [f]_{k_2} \oplus F_2([f]^{k_1})$ is recovered from f. The verifier checks whether the equation $[f]^{k_1} = F_1(m_1)$ holds. If it holds, output accept. Then the verifier combines $m = m_1||m_2$ and the message m is recovered. Otherwise, output reject to denote the signature is not valid.

In the above computation, the subscript k_2 of f denotes the least significant k_2 bits of f, and the superscript k_1 of f denotes the most significant k_1 bits of f.

Theorem 6.4 *This extended attribute-based signature with message recovery scheme is correct.*

Proof. Correctness can be verified similar with the above attribute-based signature with message recovery scheme in Theorem 6.1. □

Theorem 6.5 *This extended attribute-based signature with message recovery scheme is existentially unforgeable under chosen message attacks in the random oracle model, under the assumption that the CDH problem is hard.*

Proof. This proof is similar to that of Theorem 6.2 and therefore it is omitted. □

Theorem 6.6 *This extended attribute-based signature with message recovery scheme is equipped with the attribute privacy property in the random oracle model.*

Proof. This proof is similar to that of Theorem 6.3 and therefore it is omitted. □

6.7 Summary

We proposed a new notion of attribute-based signature with message recovery, and presented two concrete attribute-based signature with message recovery schemes based on bilinear pairing that support flexible threshold predicates. The first scheme allows the signer to embed short original message in the signature, while keeping the same signature size. The original message can be recovered from the signature. The second scheme is extended from the first one in order to deal with large messages. These schemes have been proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the CDH problem is hard. These schemes have also been proven to have the attribute privacy property.

Chapter 7

Attribute-Based Proxy Re-Signature

In 2008, Libert and Vergnaud [LV08] proposed the first multi-use unidirectional proxy re-signature schemes in order to solve the open problem left by Ateniese and Hohenberger [AH05], about constructing such a scheme where the proxy is only able to translate in one direction and signatures can be re-translated several times. However, in Libert and Vergnaud's schemes, signatures have a linear size with the number of translations. Another open problem is left about how to find such a scheme where the size of signatures and the verification cost do not grow linearly with the number of translations. In this work, we provide the first attribute-based proxy re-signature scheme, which solves the new open problem. In proxy re-signature scheme a semi-trusted proxy acts as a translator between Alice and Bob. The proxy is able to convert a signature from Alice into a signature from Bob on the same message, while the proxy cannot learn any signing key and cannot sign arbitrary messages on behalf of either Alice or Bob. In attribute-based setting, the proxy is able to convert a signature satisfying a predicate into a signature satisfying another different predicate on the same message. Our scheme satisfies the requirements of the Atenises-Hohenberger security model and the size of signatures and verification cost do not grow linearly with the number of translations.

7.1 Introduction

In 1998, Blaze, Bleumer, and Strauss proposed the notion of proxy re-signature [BBS98]. Proxy re-signature aims at secure delegation of signatures without fully trusting the proxy. In a proxy re-signature scheme, a semi-trusted proxy which is given the proxy re-signing key $RSK_{Alice \to Bob}$ acts as a translator to transform a perfectly valid and publicly verifiable signature from Alice into one from Bob on the same message. However, the proxy does not learn any signing key and cannot

generate signatures on behalf of either Alice or Bob by itself. Generating proxy re-signing key $RSK_{Alice \to Bob}$ requires Bob's secret, otherwise the underlying system is not secure. Notice that, in a proxy re-signature scheme, both the original signer Alice and the proxy cannot generate signatures on behalf of the delegator Bob individually. Only when they cooperate, can they generate signatures on behalf of Bob. The two signatures, one from the original signer Alice and the other from the proxy on behalf of Bob, can coexist. Both signatures can be publicly verified as being two signatures from two distinct people on the same message. It is a very useful tool for sharing web certificates, forming weak group signatures, and authenticating a network path. The notion of proxy re-signature can be very useful in cases when one user needs to perform signature generation without holding the necessary secret keys.

In 2005, Ateniese and Hohenberger [AH05] proposed an interesting application, in which proxy re-signature schemes can be used as a space-efficient proof that a specific path was taken in a graph without taking any shortcuts. It is suitable for either E-passport to show that a visitor followed a prescribed path, or network to show that a packet followed a prescribed path. The basic idea is that only the first node is given a signing key, all other nodes in the path are given a re-signature key which only allows it to translate signatures from adjacent nodes, but not to generate signatures. This provides several benefits compared with the approach of signing individual signature at every node separately. Firstly, because only the first node has the ability to generate signature, even if an internal node is compromised, no new messages can be injected in the graph. Secondly, only a single signature traverses the path, the verifier is not required to collect signatures and verification keys of every nodes. Finally, no information of the path itself is collected, the actual path traversed could remain private, which means the last node of the path knows that the visitor or the message followed a legitimate path but does not know which one. Of course, one could employ multisignatures or aggregate signatures to solve this problem, but both of these solutions require the verifier to collect the verification keys of all these nodes. The authors of [AH05] emphasized that a proxy re-signature scheme that is both multi-use and unidirectional would be ideal for this application, but no such scheme existed at that time. Later, Libert and Vergnaud [LV08] presented the first such constructions of multi-use unidirectional proxy re-signature, which is suitable for the previous application. But it encounters signature size blowing up with the number of translations. To solve the signature size blowing

up problem, we propose a new scheme in this work.

The concept of proxy re-signature is different from the notion of proxy signature. The proxy signature [MUO96b] allows a designated person, called proxy signer, to sign on behalf of the original signer. The proxy signer is given a secret value which is different from the original signer's secret, but a signature created from the proxy signer shows an agreement of the original signer. Different from the proxy re-signature where the proxy is semi-trusted, in proxy signature the proxy is fully trusted, because the proxy signer itself can generate signatures on arbitrary messages on behalf of the original signer. A proxy signature might be regarded as an electronic alternative only to the seal instead of the hand-written signature. A seal can be created by any person that is given a copy of the seal from its original possessor, but the hand-written signature can be created only by the true signer.

Attribute-based signatures have natural applications in many systems where users' capabilities depend on possibly complex combinations of attributes. It attests not to the identity of the individual who endorsed a message, but instead to a claim regarding the attributes she possesses. The roles of the users depend on the combination of attributes they possess. Users obtain multiple attributes from one or more attribute authorities. Attribute-based signatures describe a message and a predicate which is satisfied by the signer's attributes. The signature reveals no more than the fact that a single signer with some set of attributes satisfying the predicate has attested to the message. The signature hides the attributes used to satisfy the predicate. Users cannot collude to pool their attributes together.

Since anonymity is an important property of attribute-based signature, it is meaningless to translate a signature attesting to an attribute set to one on the same message attesting to another attribute set, when both of these attribute sets intend to satisfy the same predicate. In this work, we consider the situation that the signature transformation being proceeded between two different attribute sets which intend to satisfy different predicates.

In case of the attribute authority to revoke the signing capability of the original signer or the delegator to revoke the re-signing capability of the proxy signer, we can specify their valid period when dispatching attribute signing keys and attribute re-signing keys. Another approach to prevent a dishonest proxy signer from creating proxy re-signatures is replacing the original signer's signing key. Different from the traditional identity-based cryptography, for designated public setting, an identity only has one designated secret key. In this attribute-based proxy re-signature

scheme, for designated public setting, even users with the same attribute sets can easily get different secret keys. So, it is feasible to replace signer's secret key when it is necessary.

Compared with the scheme in Chapter 5, these two chapters are both considering computation on authenticated data. However, there are also differences between the work in these two chapters. In Chapter 5, it concentrates more about computing on authenticated data which does not require secret information of the original signer. In other word, any third party can compute on the authenticated data publicly. In this chapter, it concentrates more about computing on authenticated data which requires secret information of the original signer, which means only some designated users who have got some secret information of the original signer can compute on the authenticated data.

Related Work. The notion of proxy re-signature was proposed by Blaze, Bleumer, and Strauss [BBS98]. Its definition is informal and has created some confusion with a distinct but similar sounding definition of proxy signature introduced by Mambo, Usuda, and Okamoto [MUO96a]. Proxy signature [MUO96a] allows Bob to delegate his signing rights to Alice. In this case, Alice acts as a proxy of Bob, because Alice can sign messages on behalf of Bob. The proxy Alice by herself can generate signatures on arbitrary messages on behalf of the delegator Bob. While in proxy re-signature scheme, the proxy transforms a perfectly valid and publicly verifiable signature from Alice into one from Bob on the same message. In particular, definitions in Ivan and Dodis [ID03] are for proxy signatures, although they claimed their work generalizes, simplifies and clarifies the model of "atomic proxy" suggested by Blaze and Strauss [BS98] which is actually a proxy re-signature scheme. Ivan and Dodis [ID03] allows Bob to delegate his signing rights to Alice but only if the proxy cooperates. In their general construction, Bob's signature can be seen as a double signature which includes a signature from Alice and the other one from the proxy. There is clearly no translation between Alice's valid signatures into Bob's ones. So, [ID03] can be seen as a threshold version of proxy signature scheme, rather than a proxy re-signature scheme.

The construction of [BBS98] is bidirectional, which means the proxy information allows translating signatures in either direction, and multi-use, which means the translation of signatures can be performed in sequence and multiple times by distinct proxies without requiring the intervention of signing entities. The design of unidirectional proxy re-signature scheme, which means the proxy is only able to

translate signatures in one direction is an open problem left by [BBS98].

At CCS 2005, Ateniese and Hohenberger [AH05] revisited the notion of proxy re-signature by providing appropriate security definitions and efficient constructions in the random oracle model. They proposed two constructions based on bilinear maps. Their second scheme is unidirectional but single-use. It involves two different signing algorithms. The first-level signatures can be translated by the proxy while the second-level signatures cannot be translated. Signers have a "strong" secret and a "weak" secret that are used to produce the first level and the second level signatures respectively. They left open the problem of designing a multi-use unidirectional scheme where the proxy is able to translate in only one direction and signatures can be re-translated several times.

Libert and Vergnaud [LV08] presented the first multi-use unidirectional proxy re-signature schemes where the proxy can only translate signatures in one direction and messages can be re-signed a polynomial number of times. Like in the previous single-use unidirectional such scheme of [AH05], proxies are not completely transparent since the signatures have different shapes and lengths across successive levels. The size of signatures actually grows linearly with the number of past translations. The design of multi-use unidirectional proxy re-signature scheme where the size of signatures and the verification cost do not grow linearly with the number of translations remains an open problem in [LV08].

Attribute-based signature extends the identity-based signature of Shamir [Sha85] by allowing identity of a signer to be a set of descriptive attributes rather than a single string. Identity-based signature can be seen as a specific case of attribute-based signature with both identity size and threshold equal to one. As a related notion to attribute-based signature, fuzzy identity-based signature was proposed and formalized in [YCD08], which enables users to generate signatures with part of their attributes. Maji, prabhakaran, and Rosulek [MPR11] give a general framework for constructing attribute-based signature schemes. They show several practical instantiations based on groups with bilinear pairing operations. One of their schemes has a constant number of group elements in the signature and public-key, which means the signature and public-key size is independent of the security parameter, and depending only on the size of the predicate. It expresses the predicate as a monotone span program, which is the state of the art for attribute-based cryptography [GPSW06, BSW07, Wat11]. Given a collision resistant hash function, attributes can be arbitrary strings in this scheme.

The authors mentioned delegation in the appendix in [MPR11]. But it is entirely different from our scheme. Compared with the delegation mentioned in [MPR11] which only supports delegation of attributes regarding attribute subsets of the delegator, our scheme supports delegation of arbitrary attribute sets. Due to its support of widest class of predicates among the existing attribute-based signature schemes, its almost optimally efficiency, and its constant number of group elements in the signature which depend only on the size of the predicate, we select the scheme in [MPR11] as a building block to design attribute based proxy re-signature scheme.

Generic group model was first introduced by Shoup [Sho97] where one assumes that access to group elements is via a randomly selected representation. Maurer [Mau05] gives more interpretations and generalizations.

Our Contributions. In this work, we introduce the notion of attribute-based proxy re-signature. For the first time, this work presents a provably secure attribute-based proxy re-signature scheme based on bilinear pairing under the generic group model. This scheme solves an open problem left by Libert and Vergnaud in [LV08], which is finding out a multi-use unidirectional proxy re-signature scheme where the size of signatures and the verification cost do not grow linearly with the number of translations.

Organization of This Chapter. The rest of this chapter is organized as follows: In Section 7.2, we provide a formal definition of the attribute-based proxy re-signature scheme. In Section 7.3, we present a security model for the new scheme. In Section 7.4, we present a concrete attribute-based proxy re-signature scheme based on bilinear pairing. Section 7.5 provides the security proofs about correctness, attribute privacy, and existentially unforgeability of our scheme. Section 7.6 concludes this chapter.

7.2 Formal Definitions

Let $\mathbb{A}$ be the universe of possible attributes. A predicate over $\mathbb{A}$ is a monotone boolean function, whose inputs are associated with attributes in $\mathbb{A}$. We say that an attribute set $\mathcal{A} \subseteq \mathbb{A}$ satisfies a predicate Υ if $\Upsilon(\mathcal{A}) = 1$.

An attribute-based proxy re-signature scheme is parameterized by a universe of possible attributes $\mathbb{A}$ and message space $\mathbb{M}$, and consists of the following six algorithms.

Setup: This is a randomized algorithm that takes as input a security parameter λ and outputs the global public parameter PK and master key MK.

KeyGen: This is a probabilistic algorithm that on input of $(MK, \mathcal{A} \subseteq \mathbb{A})$, outputs a signing key $SK_{\mathcal{A}}$ representing a signer who possesses attribute set $\mathcal{A}$.

RekeyGen: This is a probabilistic algorithm that on input of $(MK, SK_{\mathcal{A}_1}, \mathcal{A}_2 \subseteq \mathbb{A})$, outputs a re-signing key $RSK_{\mathcal{A}_1 \to \mathcal{A}_2}$ for the proxy. At the same time, this algorithm gives some secure information to the original signer, which is used for generating signatures that can be re-signed. $RSK_{\mathcal{A}_1 \to \mathcal{A}_2}$ allows the proxy to transform signatures on behalf of a signer who possesses attribute set $\mathcal{A}_1$ into signatures on behalf of a signer who possesses attribute set $\mathcal{A}_2$.

If the message will be allowed to be re-signed by the proxy, the original signer uses the secure information. Otherwise, when the message is very important such that the original signer does not want it to be re-signed by any proxy, the original signer does not use the secure information, which means the original signer does not have to use this secure information in order to sign a message. The original signer still works fine without the secure information. In this case, the originally signed signatures cannot be re-signed by any proxy, even the proxy has got the correct re-signing key. In both cases, any third party cannot distinguish whether a signature is originally signed or has been re-signed by a proxy.

Sign: This is a probabilistic algorithm that on input of $(PK, SK_{\mathcal{A}}, m \in \mathbb{M}, \Upsilon)$, where $\Upsilon(\mathcal{A}) = 1$, outputs a signature σ on message m satisfying predicate Υ.

ReSign: This is a probabilistic algorithm that on input of $(PK, m, \Upsilon_1, \sigma_1, RSK_{\mathcal{A}_1 \to \mathcal{A}_2}, \Upsilon_2)$, where $\Upsilon_1(\mathcal{A}_1) = 1$ and $\Upsilon_2(\mathcal{A}_2) = 1$, if σ_1 can be verified as valid on message m satisfying predicate Υ_1, which means $Verify(PK, m, \Upsilon_1, \sigma_1) = 1$, outputs a re-signature σ_2 on the same message m but satisfying predicate Υ_2.

Verify: This is a deterministic algorithm that on input of $(PK, m, \Upsilon, \sigma)$, outputs a boolean value to indicate whether σ is a valid signature satisfying the predicate Υ on message m.

Algorithms (Setup, KeyGen, Sign, Verify) form the standard setup, key generation, signing, and verification algorithms of an attribute-based signature scheme.

7.3 Security Model

Correctness.

We call an attribute-based proxy re-signature scheme is correct, if both of the original signatures and the proxy re-signatures pass verification.

In terms of the original signatures, for all $(PK, MK) \leftarrow Setup$, all messages $m \in \mathbb{M}$, all attribute sets $\mathcal{A} \subseteq \mathbb{A}$, all signing keys $SK_{\mathcal{A}} \leftarrow KeyGen(MK, \mathcal{A})$, all predicates Υ such that $\Upsilon(\mathcal{A}) = 1$, and all signature $\sigma \leftarrow Sign(PK, SK_{\mathcal{A}}, m, \Upsilon)$, we have

$$Verify(PK, m, \Upsilon, \sigma) = 1.$$

In terms of the proxy re-signatures, for all $(PK, MK) \leftarrow Setup$, all messages $m \in \mathbb{M}$, all attribute sets $(\mathcal{A}_1, \mathcal{A}_2) \subseteq \mathbb{A}$, all signing keys $SK_{\mathcal{A}_1} \leftarrow KeyGen(MK, \mathcal{A}_1)$, all re-signing keys $RSK_{\mathcal{A}_1 \to \mathcal{A}_2} \leftarrow RekeyGen(MK, SK_{\mathcal{A}_1}, \mathcal{A}_2 \subseteq \mathbb{A})$, all predicates Υ_1 and Υ_2 such that $\Upsilon_1(\mathcal{A}_1) = 1$ and $\Upsilon_2(\mathcal{A}_2) = 1$, and all re-signatures $\sigma_2 \leftarrow ReSign(PK, m, \Upsilon_1, \sigma_1, RSK_{\mathcal{A}_1 \to \mathcal{A}_2}, \Upsilon_2)$, where $Verify(PK, m, \Upsilon_1, \sigma_1) = 1$, we have

$$Verify(PK, m, \Upsilon_2, \sigma_2) = 1.$$

Attribute Privacy.

In attribute-based signature scheme, a legitimate signer is indistinguishable among all the users whose attributes satisfying the predicate specified in the signature. The signature reveals nothing about the attributes of the signer beyond what is explicitly revealed by the claim being made.

An attribute-based proxy re-signature scheme is attribute private, if both of the original signatures and the proxy re-signatures are attribute private.

In terms of the original signatures, if for all $(PK, MK) \leftarrow Setup$, all attribute sets $(\mathcal{A}_1, \mathcal{A}_2) \subseteq \mathbb{A}$, all signing keys $SK_{\mathcal{A}_1} \leftarrow KeyGen(MK, \mathcal{A}_1), SK_{\mathcal{A}_2} \leftarrow KeyGen$ $(MK, \mathcal{A}_2)$, all messages $m \in \mathbb{M}$, and all predicates Υ such that $\Upsilon(\mathcal{A}_1) = \Upsilon(\mathcal{A}_2) = 1$, the distributions

$$Sign(PK, SK_{\mathcal{A}_1}, m, \Upsilon),$$

and

$$Sign(PK, SK_{\mathcal{A}_2}, m, \Upsilon),$$

are equal.

In terms of the proxy re-signatures, if for all $(PK, MK) \leftarrow Setup$, all attribute sets $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4) \subseteq \mathbb{A}$, all signing keys $SK_{\mathcal{A}_1} \leftarrow KeyGen(MK, \mathcal{A}_1), SK_{\mathcal{A}_2} \leftarrow$

$KeyGen(MK, \mathcal{A}_2)$, all re-signing keys $RSK_{\mathcal{A}_1 \to \mathcal{A}_3} \leftarrow RekeyGen(MK, SK_{\mathcal{A}_1}, \mathcal{A}_3 \subseteq \mathbb{A})$, $RSK_{\mathcal{A}_2 \to \mathcal{A}_4} \leftarrow RekeyGen(MK, SK_{\mathcal{A}_2}, \mathcal{A}_4 \subseteq \mathbb{A})$, all messages $m \in \mathbb{M}$, and all predicates $\Upsilon_1, \Upsilon_2, \Upsilon_3$ such that $\Upsilon_1(\mathcal{A}_1) = 1, \Upsilon_2(\mathcal{A}_2) = 1$ and $\Upsilon_3(\mathcal{A}_3) = \Upsilon_3(\mathcal{A}_4) = 1$, the distributions

$$ReSign(PK, m, \Upsilon_1, \sigma_1, RSK_{\mathcal{A}_1 \to \mathcal{A}_3}, \Upsilon_3) \text{ where } Verify(PK, m, \Upsilon_1, \sigma_1) = 1,$$

and

$$ReSign(PK, m, \Upsilon_2, \sigma_2, RSK_{\mathcal{A}_2 \to \mathcal{A}_4}, \Upsilon_3) \text{ where } Verify(PK, m, \Upsilon_2, \sigma_2) = 1,$$

are equal.

In terms of both the original signatures and the proxy re-signatures, if for all $(PK,\ MK) \leftarrow Setup$, all attribute sets $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3) \subseteq \mathbb{A}$, all signing keys $SK_{\mathcal{A}_1} \leftarrow KeyGen(MK, \mathcal{A}_1)$, $SK_{\mathcal{A}_2} \leftarrow KeyGen(MK, \mathcal{A}_2)$, all re-signing keys $RSK_{\mathcal{A}_1 \to \mathcal{A}_3} \leftarrow RekeyGen(MK, SK_{\mathcal{A}_1}, \mathcal{A}_3 \subseteq \mathbb{A})$, all messages $m \in \mathbb{M}$, and all predicates Υ_1, Υ_2 such that $\Upsilon_1(\mathcal{A}_1) = 1, \Upsilon_2(\mathcal{A}_2) = \Upsilon_2(\mathcal{A}_3) = 1$, the distributions

$$Sign(PK, SK_{\mathcal{A}_2}, m, \Upsilon_2),$$

and

$$ReSign(PK, m, \Upsilon_1, \sigma_1, RSK_{\mathcal{A}_1 \to \mathcal{A}_3}, \Upsilon_2) \text{ where } Verify(PK, m, \Upsilon_1, \sigma_1) = 1,$$

are equal.

It can be defined using a game between an adversary $\mathcal{D}$ and a challenger $\mathcal{C}$.

The adversary $\mathcal{D}$ even knows the master key of the system. So he could generate all signer's private keys as well as public keys. Its goal is to distinguish between two signers which one generates the valid signature of a message with a predicate such that both of their attributes satisfy the predicate.

Firstly, challenger $\mathcal{C}$ runs the Setup algorithm to get the system's master key MK with respect to the security parameter and the system's public parameter PK. Then, $\mathcal{C}$ sends PK as well as the master key MK to adversary $\mathcal{D}$. $\mathcal{D}$ can generate private keys and signatures itself, because he has got the master key of the system.

Challenge: $\mathcal{D}$ outputs a message m^*, two challenged attribute sets $\mathcal{A}_0^*, \mathcal{A}_1^*$ for signature query, where both $\mathcal{A}_0^*$ and $\mathcal{A}_1^*$ satisfy a predicate Υ^*. $\mathcal{C}$ chooses a bit $b \in \{0, 1\}$, computes the challenge signature σ^* satisfying Υ^* on behalf of the signer who possesses attribute set $\mathcal{A}_b^*$ and provides σ^* to $\mathcal{D}$.

Guess: $\mathcal{D}$ tries to guess which attribute set between $\mathcal{A}_0^*$ and $\mathcal{A}_1^*$ was used to generate the challenge signature σ^*. Finally, $\mathcal{D}$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b' = b$.

If there is no such polynomial-time adversary $\mathcal{D}$ that can win the game described above, which means $\mathcal{D}$ can only output the attribute set used in the challenge signature generation with probability no better than $1/2$, we say that the attribute-based proxy re-signature scheme holds attribute privacy property.

It is worth noting that this property holds even for the central attribute authority, because the master key is also released to the adversary. If the attribute privacy requirement holds, then a signature does not leak which set of attributes or signing key was used to generate it. This holds even if the adversary is unbounded and has access to the signer's private keys.

Unforgeability.

The security model of [AH05] protects users from two types of attacks: those launched from parties outside the system (External Security), and those launched from parties inside the system, such as the proxy, another delegation partner, or some collusion between them (Internal Security).

- **External security:** This security notion protects a user from adversaries outside the system (i.e., excluding the proxy and any delegation partners).

  For $i = 1$ to N where $N \in poly(\lambda)$, assume there are N pairs of attribute sets $\mathcal{A}_i$ and corresponding predicates Υ_i. For these N pairs of Υ_i and $\mathcal{A}_i$, only when the subscripts i of Υ_i and $\mathcal{A}_i$ are equal then $\Upsilon_i(\mathcal{A}_i) = 1$, otherwise $\Upsilon_i(\mathcal{A}_j) \neq 1$ when $i \neq j$, which implies that for different $\mathcal{A}_i$ and $\mathcal{A}_j$, neither $\mathcal{A}_i \subset \mathcal{A}_j$ nor $\mathcal{A}_j \subset \mathcal{A}_i$. This notion demands that for all PPT algorithms $\mathcal{B}$ the next probability be a negligible function of the security parameter λ.

$$\Pr[\{SK_{\mathcal{A}_i} \leftarrow KeyGen(MK, \mathcal{A}_i)\}_{i\in[1,N]},$$
$$(m, \Upsilon_k, \sigma) \leftarrow \mathcal{B}^{\mathcal{O}_{Sign}(\cdot,\cdot,\cdot),\mathcal{O}_{ReSign}(\cdot,\cdot,\cdot,\cdot)}(PK) :$$
$$Verify(PK, m, \Upsilon_k, \sigma) = 1$$
$$\wedge(1 \leq k \leq N) \wedge (m, \Upsilon_k, \sigma) \notin Q].$$

  where the oracle $\mathcal{O}_{Sign}$ takes as input an attribute set $\mathcal{A}_i$, a message $m \in \mathbb{M}$ and a predicate Υ_i, and produces the output of $Sign(PK, SK_{\mathcal{A}_i}, m, \Upsilon_i)$ where $\Upsilon_i(\mathcal{A}_i) = 1$; the oracle $\mathcal{O}_{ReSign}$ takes as input two distinct predicate Υ_i, Υ_j,

a message $m \in \mathbb{M}$, and a signature σ_i where $Verify(PK, m, \Upsilon_i, \sigma_i) = 1$, and produces the output of $ReSign(PK, m, \Upsilon_i, \sigma_i, RSK_{\mathcal{A}_i \to \mathcal{A}_j}, \Upsilon_j)$ where $\Upsilon_i(\mathcal{A}_i) = 1$ and $\Upsilon_j(\mathcal{A}_j) = 1$; and Q denotes the set of tuples (m, Υ, σ) where $\mathcal{B}$ obtained a signature σ on m satisfying predicate Υ by querying $\mathcal{O}_{Sign}$ on (m, Υ) or $\mathcal{O}_{ReSign}(m, \Upsilon_i, \sigma_i, \Upsilon)$.

- **Internal security:** This security notion protects users against dishonest proxies and colluding delegation partners. Three security guarantees should be ensured.

  1. **Limited Proxy Security:** This notion captures the proxy's inability to sign messages on behalf of the delegatee or to create signatures on behalf of the delegator without the messages were firstly signed by one of the latter's delegatees. If the delegator and the delegatee are both honest, then: (1) the proxy cannot produce signatures for the delegator unless the message was first signed by one of her delegatees, and (2) the proxy cannot create any signatures for the delegatee. This is identical to the external security game, except that instead of a re-signing oracle $\mathcal{O}_{ReSign}$, $\mathcal{B}$ may directly obtain the re-signing keys via $\mathcal{O}_{RekeyGen}$. Formally, we consider a game where adversaries have all re-signing keys but are denied access to signers' private keys.

     For $i = 1$ to N where $N \in poly(\lambda)$, assume there are N pairs of attribute sets $\mathcal{A}_i$ and corresponding predicates Υ_i. For these N pairs of Υ_i and $\mathcal{A}_i$, only when the subscripts i of Υ_i and $\mathcal{A}_i$ are equal then $\Upsilon_i(\mathcal{A}_i) = 1$, otherwise $\Upsilon_i(\mathcal{A}_j) \neq 1$ when $i \neq j$, which implies that for different $\mathcal{A}_i$ and $\mathcal{A}_j$, neither $\mathcal{A}_i \subset \mathcal{A}_j$ nor $\mathcal{A}_j \subset \mathcal{A}_i$. For all PPT algorithms $\mathcal{B}$, The following probability should be negligible:

$$\Pr[\{SK_{\mathcal{A}_i} \leftarrow KeyGen(MK, \mathcal{A}_i)\}_{i \in [1,N]},$$
$$(m, \Upsilon_k, \sigma) \leftarrow \mathcal{B}^{\mathcal{O}_{Sign}(\cdot,\cdot,\cdot), \mathcal{O}_{RekeyGen}(\cdot,\cdot,\cdot,\cdot)}(PK) :$$
$$Verify(PK, m, \Upsilon_k, \sigma) = 1$$
$$\wedge (1 \leq k \leq N) \wedge (m, \Upsilon_k, \sigma) \notin Q].$$

     where the oracle $\mathcal{O}_{RekeyGen}$ takes as input two distinct attribute sets $\mathcal{A}_i, \mathcal{A}_j$ and returns the output of $RSK_{\mathcal{A}_i \to \mathcal{A}_j} = RekeyGen(MK, SK_{\mathcal{A}_i}, \mathcal{A}_j)$; and Q denotes the set of tuples (m, Υ, σ) where $\mathcal{B}$ obtained a signature

on m satisfying predicate Υ or one of its delegatee predicates by querying $\mathcal{O}_{Sign}$.

2. **Delegatee Security:** This notion protects the delegatee from a colluding delegator and proxy. If the delegatee is honest, then she is "safe" from a colluding delegator and proxy. That is, they cannot produce any signatures on her behalf.

   For $i = 1$ to N where $N \in poly(\lambda)$, assume there are N pairs of attribute sets $\mathcal{A}_i$ and corresponding predicates Υ_i. For these N pairs of Υ_i and $\mathcal{A}_i$, only when the subscripts i of Υ_i and $\mathcal{A}_i$ are equal then $\Upsilon_i(\mathcal{A}_i) = 1$, otherwise $\Upsilon_i(\mathcal{A}_j) \neq 1$ when $i \neq j$, which implies that for different $\mathcal{A}_i$ and $\mathcal{A}_j$, neither $\mathcal{A}_i \subset \mathcal{A}_j$ nor $\mathcal{A}_j \subset \mathcal{A}_i$. We associate the attribute set $\mathcal{A}^*$ to the delegatee, and there is a predicate Υ^* such that only $\mathcal{A}^*$ itself or the superset of $\mathcal{A}^*$ can satisfy. $\mathcal{A}^*$ is not a subset of any $\{\mathcal{A}_i\}_{i\in[1,N]}$. For all PPT algorithms $\mathcal{B}$, the following probability should be negligible:

$$\begin{aligned}
&\Pr[\{SK_{\mathcal{A}_i} \leftarrow KeyGen(MK, \mathcal{A}_i)\}_{i\in[1,N]},\\
&\quad (m^*, \Upsilon^*, \sigma^*) \leftarrow \mathcal{B}^{\mathcal{O}_{Sign}(\mathcal{A}^*,\cdot,\cdot),\mathcal{O}_{RekeyGen}(\cdot,\cdot,\star)}(PK, \{SK_{\mathcal{A}_i}\}_{i\in[1,N]}) :\\
&\qquad Verify(PK, m^*, \Upsilon^*, \sigma^*) = 1\\
&\qquad\quad \wedge(\Upsilon^*(\{\mathcal{A}_i\}_{i\in[1,N]}) \neq 1) \wedge (m^*, \Upsilon^*, \sigma^*) \notin Q].
\end{aligned}$$

   where $\star \neq \mathcal{A}^*$; and Q denotes the set of tuples (m, Υ, σ) such that $\mathcal{B}$ queried $\mathcal{O}_{Sign}(\mathcal{A}^*, m, \Upsilon)$ to obtain a signature σ on m satisfying predicate Υ.

3. **Delegator Security:** Different from [AH05, LV08] where the proxy re-signature has distinct shape with the original signature, in our scheme the proxy re-signature has the same shape with the original signature. It is indeed for this reason that our scheme achieves the goal of multi-use unidirectional proxy re-signature scheme where the size of signatures and the verification cost do not grow linearly with the number of translations. Coupled with the anonymity property of attribute-based signature, in our scheme collusion of the delegatee and the proxy, whose objective attribute set satisfies the delegator's predicate, can obviously generate valid signature on behalf of the delegator. We should slightly change the security model of delegator security of [AH05, LV08]. After modification, the Delegator Security and the Delegatee Security are almost the same,

except that we associate the attribute set $\mathcal{A}^*$ to the delegator or the delegatee respectively. This notion protects the delegator from a colluding delegatee and proxy. If the delegator is honest, then he is "safe" from the colluding delegatee and proxy. That is, they cannot produce any signatures on behalf of the delegator.

For $i = 1$ to N where $N \in poly(\lambda)$, assume there are N pairs of attribute sets $\mathcal{A}_i$ and corresponding predicates Υ_i. For these N pairs of Υ_i and $\mathcal{A}_i$, only when the subscripts i of Υ_i and $\mathcal{A}_i$ are equal then $\Upsilon_i(\mathcal{A}_i) = 1$, otherwise $\Upsilon_i(\mathcal{A}_j) \neq 1$ when $i \neq j$, which implies that for different $\mathcal{A}_i$ and $\mathcal{A}_j$, neither $\mathcal{A}_i \subset \mathcal{A}_j$ nor $\mathcal{A}_j \subset \mathcal{A}_i$. We associate the attribute set $\mathcal{A}^*$ to the delegator, and there is a predicate Υ^* such that only $\mathcal{A}^*$ itself or the superset of $\mathcal{A}^*$ can satisfy. $\mathcal{A}^*$ is not a subset of any $\{\mathcal{A}_i\}_{i\in[1,N]}$. For all PPT algorithms $\mathcal{B}$, the following probability should be negligible:

$$\begin{aligned}\Pr[&\{SK_{\mathcal{A}_i} \leftarrow KeyGen(MK, \mathcal{A}_i)\}_{i\in[1,N]},\\ &(m^*, \Upsilon^*, \sigma^*) \leftarrow \mathcal{B}^{\mathcal{O}_{Sign}(\mathcal{A}^*,\cdot,\cdot),\mathcal{O}_{RekeyGen}(\cdot,\cdot,\star)}(PK, \{SK_{\mathcal{A}_i}\}_{i\in[1,N]}) :\\ &\quad Verify(PK, m^*, \Upsilon^*, \sigma^*) = 1\\ &\quad\quad \wedge(\Upsilon^*(\{\mathcal{A}_i\}_{i\in[1,N]}) \neq 1) \wedge (m^*, \Upsilon^*, \sigma^*) \notin Q].\end{aligned}$$

where $\star \neq \mathcal{A}^*$; and Q denotes the set of tuples (m, Υ, σ) such that $\mathcal{B}$ queried $\mathcal{O}_{Sign}(\mathcal{A}^*, m, \Upsilon)$ to obtain a signature σ on m satisfying predicate Υ.

7.4 Proposed Scheme

This construction supports all predicates whose monotone span programs have width at most t_{max}, where t_{max} is an arbitrary parameter. We treat $\mathbb{A} = \mathbb{Z}_p^*$ as the universe of attributes, where p is the size of the sufficient large cyclic prime order group used in the scheme. Signatures in this scheme consist of exactly $(s + 4)$ group elements, where s is the size of the predicate's monotone span program.

Setup: Choose suitable cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p, equipped with a bilinear pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Choose two collision-resistant hash functions $H : \{0,1\}^* \rightarrow \mathbb{G}^*, \mathcal{H} : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. Choose random generators:

$$(g, C, h_0, \cdots, h_{t_{max}}) \leftarrow \mathbb{G}^*.$$

Choose random $(a_0, a, b) \leftarrow \mathbb{Z}_p^*$ and set:

$$A'_0 = h_0^{a_0}; \quad A_0 = h_0^a, B_0 = h_0^b;$$

$$A_j = h_j^a \text{ and } B_j = h_j^b \quad (\forall j \in [t_{max}]).$$

The master key is $MK = (a_0, a, b)$. The public key PK is a description of the groups $\mathbb{G}, \mathbb{G}_T$ and their pairing function, as well as:

$$(H, \mathcal{H}, g, C, h_0, \cdots, h_{t_{max}}, A'_0, A_0, \cdots, A_{t_{max}}, B_0, \cdots, B_{t_{max}}).$$

KeyGen: On input of MK as above and attribute set $\mathcal{A}_1 \subseteq \mathbb{A}$, choose random generator $K_{\mathcal{A}_1} \leftarrow \mathbb{G}^*$. Set:

$$K_0 = K_{\mathcal{A}_1}^{1/a_0}; \quad K_{u_{i1}} = K_{\mathcal{A}_1}^{1/(a+bu_{i1})} \quad (\forall u_{i1} \in \mathcal{A}_1).$$

The signing key is then:

$$SK_{\mathcal{A}_1} = (K_{\mathcal{A}_1}, \quad K_0, \quad \{K_{u_{i1}} | u_{i1} \in \mathcal{A}_1\}).$$

in which u_{i1} is an attribute in the attribute set $\mathcal{A}_1$.

RekeyGen: On input of MK as above, a signing key $SK_{\mathcal{A}_1}$ corresponding to attribute set $\mathcal{A}_1 \subseteq \mathbb{A}$, and attribute set $\mathcal{A}_2 \subseteq \mathbb{A}$, the attribute authority chooses random $r \leftarrow \mathbb{Z}_p^*$. Set:

$$K_{u_{i2}} = K_{\mathcal{A}_1}^{r/(a+bu_{i2})} \quad (\forall u_{i2} \in \mathcal{A}_2).$$

The re-signing key that is able to transform signatures on behalf of a signer who possesses attribute set $\mathcal{A}_1$ to signatures on behalf of another signer who possesses attribute set $\mathcal{A}_2$ is:

$$RSK_{\mathcal{A}_1 \to \mathcal{A}_2} = \{K_{u_{i2}} | u_{i2} \in \mathcal{A}_2\},$$

in which u_{i2} is an attribute in the attribute set $\mathcal{A}_2$.

The attribute authority also gives the randomness r to the original signer via a secure channel.

In order to conduct consecutive proxy re-signing, while the multiple proxy signers are specified, the attribute authority should use a unique r to compute re-signing key for every proxy signer with respect to each original signer.

Sign: On input of $(MK, SK_{\mathcal{A}}, m, \Upsilon)$ such that $\Upsilon(\mathcal{A}) = 1$, first convert Υ to its corresponding monotone span program $\mathcal{M} \in (\mathbb{Z}_p)^{l \times t}$, with row labeling function

$u : [l] \to \mathbb{A}$. Note that the range of this labeling function are attributes, so we use the symbol u representing this function, which is the same as that used in the signing key and re-signing key representing attributes. Also compute the vector $\overrightarrow{v}$ that corresponds to the satisfying assignment $\mathcal{A}$. Compute $\mu = \mathcal{H}(m||\Upsilon)$.

For generating an original signature, a signer chooses random numbers $\alpha \leftarrow \mathbb{Z}_p^*$ and $(r_1, \cdots, r_l) \leftarrow \mathbb{Z}_p^*$ and computes:

$$
\begin{aligned}
&H_0 = h_0^{r\cdot\alpha};\\
&H_1 = h_1^{r\cdot\alpha};\\
&Y = (K_{\mathcal{A}} \cdot H(m))^{r\cdot\alpha};\\
&W = K_0^{r\cdot\alpha};\\
&S_i = \left(K_{u(i)}^{v_i}\right)^{r\cdot\alpha} \cdot (Cg^{\mu})^{r_i} \qquad \text{for} \qquad (i \in [l]);\\
&P_j = \prod_{i=1}^{l} \left(A_j B_j^{u(i)}\right)^{\mathcal{M}_{ij}\cdot r_i} \qquad \text{for} \qquad (j \in [t]);
\end{aligned}
$$

The signer may not have $K_{u(i)}$ for every attribute $u(i)$ mentioned in the predicate. But when this is the case, $v_i = 0$. So the value is not needed. The signature is:

$$\sigma = (H_0, H_1, Y, W, S_1, \cdots, S_l, P_1, \cdots, P_t).$$

In order to let a specified proxy to be able to re-sign the original signature, the randomness $R = \alpha$ also should be sent to the specified proxy signer via secure channel. Otherwise, if the original signer does not allow his signature to be re-signed, this step is not required. In this case, the proxy cannot re-sign the signature even the proxy has got the correct re-signing key.

ReSign: On input of $(PK, m, \Upsilon_1, \sigma_1, RSK_{\mathcal{A}_1\to\mathcal{A}_2}, \Upsilon_2)$ along with the randomness R such that $\Upsilon_1(\mathcal{A}_1) = 1$ and $\Upsilon_2(\mathcal{A}_2) = 1$, the proxy firstly checks the validity of the signature σ_1 and R. If $Verify(PK, m, \Upsilon_1, \sigma_1) = 1$, the proxy takes any attribute u_i included in the re-signing key and continues to check whether

$$e\left(K_{u_i}^R, A_0 \cdot B_0^{u_i}\right) \stackrel{?}{=} e\left(W, A_0'\right).$$

If this equation also holds, the signature σ_1 can be re-signed using the re-signing key $RSK_{\mathcal{A}_1\to\mathcal{A}_2}$. Let the original signature $\sigma_1 = (H_0, H_1, Y, W, S_1, \cdots, S_l, P_1, \cdots, P_t)$, the re-signing key $RSK_{\mathcal{A}_1\to\mathcal{A}_2} = \{K_{u_{i2}} | u_{i2} \in \mathcal{A}_2\}$. The proxy converts Υ_2 to its corresponding monotone span program $\mathcal{M}' \in (\mathbb{Z}_p)^{l'\times t'}$, with row labeling $u' : [l'] \to \mathbb{A}$. Also compute the vector $\overrightarrow{v'}$ which corresponds to the satisfying assignment $\mathcal{A}_2$. Compute $\mu' = \mathcal{H}(m||\Upsilon_2)$.

The proxy chooses randomness $(\beta, r_1', \cdots, r_{l'}') \leftarrow \mathbb{Z}_p^*$ and computes:

$$
\begin{aligned}
&H_0' = H_0^{\beta} = h_0^{r\cdot R\cdot\beta};\\
&H_1' = H_1^{\beta} = h_1^{r\cdot R\cdot\beta};\\
&Y' = Y^{\beta} = \left(K_{\mathcal{A}_1}\cdot H(m)\right)^{r\cdot R\cdot\beta};\\
&W' = W^{\beta} = K_0^{r\cdot R\cdot\beta};\\
&S_i' = \left(K_{u'(i)}^{v_i'}\right)^{R\cdot\beta}\cdot\left(Cg^{\mu'}\right)^{r_i'} \qquad \text{for} \qquad (i\in[l']);\\
&P_j' = \prod_{i=1}^{l}\left(A_jB_j^{u'(i)}\right)^{\mathcal{M}_{ij}'\cdot r_i'} \qquad \text{for} \qquad (j\in[t']);
\end{aligned}
$$

The re-signature is:

$$\sigma_2 = (H_0', H_1', Y', W', S_1', \cdots, S_{l'}', P_1', \cdots, P_{t'}').$$

It has the same shape as that of signatures generated by Sign algorithm. In this case, the randomness $R' = R\cdot\beta$ also should be sent to the next specified proxy via secure channel, if this re-signature is also for re-signing.

Both of the signatures generated by the Sign algorithm and the ReSign algorithm can be as input of this ReSign algorithm, so this scheme is multi-use.

Verify: On input of $(PK, m, \Upsilon, \sigma = (H_0, H_1, Y, W, S_1, \cdots, S_l, P_1, \cdots, P_t))$, first convert Υ to its corresponding monotone span program $\mathcal{M}\in(\mathbb{Z}_p)^{l\times t}$, with row labeling $u:[l]\to\mathbb{A}$. Compute $\mu = \mathcal{H}(m||\Upsilon)$. Check the following constraints:

$$e(W, A_0')\cdot e(H(m), H_0)\stackrel{?}{=} e(Y, h_0),$$

$$e\left(H(m), H_1\right)\cdot\prod_{i=1}^{l} e\left(S_i, \left(A_1B_1^{u(i)}\right)^{\mathcal{M}_{i1}}\right)\stackrel{?}{=} e\left(Y, h_1\right)\cdot e\left(Cg^{\mu}, P_1\right),$$

$$\prod_{i=1}^{l} e\left(S_i, \left(A_jB_j^{u(i)}\right)^{\mathcal{M}_{ij}}\right)\stackrel{?}{=} e\left(Cg^{\mu}, P_j\right) \quad \text{for} \quad (j>1 \ \ \& \ \ j\in[t]).$$

Return 1 if all the above checks succeed, and 0 otherwise.

7.5 Security Analysis

Theorem 7.1 *This attribute-based proxy re-signature scheme is correct in terms of signatures generated by both the Sign algorithm and the ReSign algorithm.*

Proof. For signatures generated by the Sign algorithm.

For the formula $e(W, A_0')e(H(m), H_0) \stackrel{?}{=} e(Y, h_0)$.

$$\begin{aligned} & e(W, A_0') \cdot e(H(m), H_0) \\ = \; & e\left((K_{\mathcal{A}})^{\alpha/a_0}, h_0^{a_0}\right) \cdot e\left(H(m), h_0^{\alpha}\right) \\ = \; & e\left((K_{\mathcal{A}} \cdot H(m))^{\alpha}, h_0\right) \\ = \; & e\left(Y, h_0\right). \end{aligned}$$

The left hand side equals to the right hand side.

For the formula

$$e\left(H(m), H_1\right) \cdot \prod_{i=1}^{l} e\left(S_i, \left(A_1 B_1^{u(i)}\right)^{\mathcal{M}_{i1}}\right) \stackrel{?}{=} e\left(Y, h_1\right) \cdot e\left(Cg^{\mu}, P_1\right).$$

$$\begin{aligned} & e\left(H(m), H_1\right) \cdot \prod_{i=1}^{l} e\left(S_i, \left(A_1 B_1^{u(i)}\right)^{\mathcal{M}_{i1}}\right) \\ = \; & e\left(H(m), h_1^{\alpha}\right) \cdot \prod_{i=1}^{l} e\left(\left(K_{u(i)}^{v_i}\right)^{\alpha}, \left(A_1 B_1^{u(i)}\right)^{\mathcal{M}_{i1}}\right) \cdot \prod_{i=1}^{l} e\left(\left(Cg^{\mu}\right)^{r_i}, \left(A_1 B_1^{u(i)}\right)^{\mathcal{M}_{i1}}\right) \\ = \; & e\left(H(m)^{\alpha}, h_1\right) \cdot \prod_{i=1}^{l} e\left(K_{\mathcal{A}}^{\frac{1}{(a+bu_i)} \cdot v_i \cdot \alpha}, \left(h_1^{a+bu(i)}\right)^{\mathcal{M}_{i1}}\right) \cdot \prod_{i=1}^{l} e\left(Cg^{\mu}, \left(A_1 B_1^{u(i)}\right)^{\mathcal{M}_{i1} \cdot r_i}\right) \\ = \; & e\left(H(m)^{\alpha}, h_1\right) \cdot e\left(K_{\mathcal{A}}^{\alpha}, h_1\right)^{\sum_{i=1}^{l} v_i \cdot \mathcal{M}_{i1}} \cdot \prod_{i=1}^{l} e\left(Cg^{\mu}, \left(A_1 B_1^{u(i)}\right)^{\mathcal{M}_{i1} \cdot r_i}\right) \\ = \; & e\left(\left(H(m) \cdot K_{\mathcal{A}}\right)^{\alpha}, h_1\right) \cdot \prod_{i=1}^{l} e\left(Cg^{\mu}, \left(A_1 B_1^{u(i)}\right)^{\mathcal{M}_{i1} \cdot r_i}\right) \\ = \; & e\left(Y, h_1\right) \cdot e\left(Cg^{\mu}, P_1\right). \end{aligned}$$

The left hand side equals to the right hand side.

When $j > 1 \;\; \& \;\; j \in [t]$, for the formula

$$\prod_{i=1}^{l} e\left(S_i, \left(A_j B_j^{u(i)}\right)^{\mathcal{M}_{ij}}\right) \stackrel{?}{=} e\left(Cg^{\mu}, P_j\right).$$

$$\begin{aligned} & \prod_{i=1}^{l} e\left(S_i, \left(A_j B_j^{u(i)}\right)^{\mathcal{M}_{ij}}\right) \\ = \; & \prod_{i=1}^{l} e\left(\left(K_{u(i)}^{v_i}\right)^{\alpha}, \left(A_j B_j^{u(i)}\right)^{\mathcal{M}_{ij}}\right) \cdot \prod_{i=1}^{l} e\left(\left(Cg^{\mu}\right)^{r_i}, \left(A_j B_j^{u(i)}\right)^{\mathcal{M}_{ij}}\right) \end{aligned}$$

$$
\begin{aligned}
&= \prod_{i=1}^{l} e\left(K_{\mathcal{A}}^{\frac{1}{(a+bu_i)}\cdot v_i\cdot\alpha}, \left(h_j^{a+bu(i)}\right)^{\mathcal{M}_{ij}}\right)\cdot\prod_{i=1}^{l} e\left(Cg^{\mu}, \left(A_jB_j^{u(i)}\right)^{\mathcal{M}_{ij}\cdot r_i}\right)\\
&= e\left(K_{\mathcal{A}}^{\alpha}, h_j\right)^{\sum_{i=1}^{l} v_i\cdot\mathcal{M}_{ij}}\cdot\prod_{i=1}^{l} e\left(Cg^{\mu}, \left(A_jB_j^{u(i)}\right)^{\mathcal{M}_{ij}\cdot r_i}\right)\\
&= \prod_{i=1}^{l} e\left(Cg^{\mu}, \left(A_jB_j^{u(i)}\right)^{\mathcal{M}_{ij}\cdot r_i}\right)\\
&= e\left(Cg^{\mu}, P_j\right).
\end{aligned}
$$

The left hand side equals to the right hand side.

For signatures generated by the ReSign algorithm, the correctness can also be proved in a similar way as above. □

Theorem 7.2 *This attribute-based proxy re-signature scheme is attribute private.*

Proof. **Setup**: $\mathcal{C}$ runs the normal Setup algorithm and sends the public parameters PK as well as the master key MK to adversary $\mathcal{D}$.

Challenge: The adversary outputs two attribute sets $\mathcal{A}_0^*$ and $\mathcal{A}_1^*$. Both the adversary $\mathcal{D}$ and the challenger $\mathcal{C}$ can generate secret keys corresponding to these two attribute sets as $SK_{\mathcal{A}_0^*}$ and $SK_{\mathcal{A}_1^*}$ respectively. Then, the adversary outputs a message m^* and a predicate Υ^* where $\Upsilon^*(\mathcal{A}_0^*) = \Upsilon^*(\mathcal{A}_1^*) = 1$. The adversary $\mathcal{D}$ asks the challenger to generate a signature on the message m^* satisfying Υ^* from either $\mathcal{A}_0^*$ or $\mathcal{A}_1^*$. The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and outputs a signature $\sigma^* = (H_0^*, H_1^*, Y^*, W^*, S_1^*, \cdots, S_l^*, P_1^*, \cdots, P_t^*)$ by running the Sign algorithm using the secret key $SK_{\mathcal{A}_b^*}$.

As described in the Sign and the Verify algorithms, the signing key of any attribute set which satisfies the predicate Υ^* can generate a valid signature σ^*. During the generation of the signature, different attribute sets $\mathcal{A}_0^*, \mathcal{A}_1^*$ corresponds to different vector $\overrightarrow{v}_0$, $\overrightarrow{v}_1$, but the monotone span program $\mathcal{M}^*$ corresponding to the predicate Υ^* is identical. Since both $\Upsilon^*(\mathcal{A}_0^*) = 1$ and $\Upsilon^*(\mathcal{A}_1^*) = 1$, $\overrightarrow{v}_0\mathcal{M}^* = \overrightarrow{v}_1\mathcal{M}^* = 1$. Thus, both signatures generated by $SK_{\mathcal{A}_0^*}$ and $SK_{\mathcal{A}_1^*}$ on the same message m^*, which have the same shape, are uniformly distributed, subject to the constraint that $Verify(PK, m^*, \Upsilon^*, \sigma^*) = 1$.

So, the challenge signature can be regarded as generated by a signer who possesses signing keys corresponding to attribute sets satisfying the predicate Υ^*. Thus, if this challenge signature is generated by using the secret key $SK_{\mathcal{A}_0^*}$, it can also be

generated by using the secret key $SK_{\mathcal{A}_1^*}$ since the secret key $SK_{\mathcal{A}_1^*}$ also satisfy the situation mentioned above. Similarly, if this challenge signature is generated by using the secret key $SK_{\mathcal{A}_1^*}$, it can also be generated by using the secret key $SK_{\mathcal{A}_0^*}$.

In our scheme, the original signatures and the proxy re-signatures have the same shape. So, the above proof can be extended to the proxy re-signature case. Similarly, it can also be extended to one is the original signature and the other is proxy re-signature case.

Therefore, even the adversary has access to the master key and has unbounded computation ability, he cannot distinguish between two signers which one generates a valid signature with respect to a predicate such that both of their attributes satisfying the predicate. □

Theorem 7.3 *This attribute-based proxy re-signature scheme is unforgeable in the generic group model.*

Proof. Following the proof of unforgeability in [MPR11], we will show our scheme is unforgeable in the generic group model.

The main difference of our scheme is there exists a hash value $H(m)$ of the message m in element Y. In order to let the simulator keep track of its discrete logarithm by means of a multivariate rational functions in the generic group model, we should provide a random oracle H-oracle to the adversary. For every query to the H-oracle, if the message m has never been queried, the simulator randomly selects a number $\Delta_m \leftarrow \mathbb{Z}_p^*$, and returns g^{Δ_m} as the hash value. Then the simulator records this record. If the message has been queried, the simulator only returns the corresponding hash value to the adversary.

Since our proof is very similar with that of [MPR11], we will only point out the differences. More details can be found in [MPR11].

The simulation associates each group element with some rational function. The functions are associated with the group elements in the simulation as follows:

1. Public key components generated by Setup:
   (a) 1, representing the generator g.
   (b) c, representing $C = g^c$.
   (c) Δ_0, representing $h_0 = g^{\Delta_0}$.
   (d) $\{\Delta_j | j \in [t_{max}]\}$, representing $h_j = g^{\Delta_j}$.

(e) $\Delta_0 a_0$, representing $A'_0 = h_0^{a_0}$.

(f) $\Delta_0 a$, representing $A_0 = h_0^a$.

(g) $\Delta_0 b$, representing $B_0 = h_0^b$.

(h) $\{\Delta_j a | j \in [t_{max}]\}$, representing $A_j = h_j^a$.

(i) $\{\Delta_j b | j \in [t_{max}]\}$, representing $B_j = h_j^b$.

2. Signing key components given by KeyGen.

   Let $\mathcal{A}_{k_1}$ be the k_1th set of attributes queried to KeyGen:

   (a) x_{k_1}, representing $K_{\mathcal{A}_{k_1}}^{(k_1)} = g^{x_{k_1}}$.

   (b) x_{k_1}/a_0, representing $K_0^{(k_1)} = g^{x_{k_1}/a_0}$.

   (c) $\{x_{k_1}/(a+bu) | u \in \mathcal{A}_{k_1}\}$, representing $K_u^{(k_1)} = g^{x_{k_1}/(a+bu)}$.

3. Re-Signing key components given by RekeyGen.

   Let $\mathcal{A}_{k_2}$ be the k_2th set of attributes queried to RekeyGen, which is based on the k_1th signing key generated by KeyGen.

   (a) $\{x_{k_1} \cdot r/(a+bu) | u \in \mathcal{A}_{k_2}\}$, representing $K_u^{(k_2)} = g^{x_{k_1} \cdot r/(a+bu)}$.

4. Signature queries.

   Suppose the q-th signature query is on message $m^{(q)}$ under the predicate $\Upsilon^{(q)}$. Let $\mathcal{M}^{(q)} \in (\mathbb{Z}_p)^{l^{(q)} \times t^{(q)}}$ be the corresponding monotone span program, with row labeling $u^{(q)} : [l^{(q)}] \to \mathbb{A}$. Let $\mu^{(q)} = \mathcal{H}(m^{(q)} || \Upsilon^{(q)})$.

   (a) Δ_0, representing $H_0 = h_0^\alpha$.

   (b) Δ_1, representing $H_1 = h_1^\alpha$.

   (c) $y^{(q)} + \Delta_m^{(q)}$, representing $Y^{(q)} = (K_{\mathcal{A}} \cdot H(m))^\alpha = g^{y^{(q)} + \Delta_m^{(q)}}$.

   (d) $y^{(q)}/a_0$, representing $W = K_0^\alpha = g^{y^{(q)}/a_0}$.

   (e) $\{s_i^{(q)} | i \in [l^{(q)}]\}$, representing $S_i = g^{s_i^{(q)}}$.

   (f) $\{p_j^{(q)} | j \in [t^{(q)}]\}$, where

$$p_j^{(q)} = \frac{\Delta_j}{c + \mu^{(q)}} \left[\sum_{i=1}^{l^{(q)}} s_i^{(q)} (a + u^{(q)}(i) b) \mathcal{M}_{i,j}^{(q)} - y^{(q)} z_j \right],$$

   where $\overrightarrow{z} = [1, 0, \cdots, 0]$, representing $P_j^{(q)} = g^{p_j^{(q)}}$.

5. Re-Signature Queries.

   Suppose the q-th re-signature query is on message $m^{(q)}$ under the predicate $\Upsilon_2^{(q)}$, the signature intended to be transformed is σ_1 under the predicate Υ_1. If $Verify(PK, m, \Upsilon_1, \sigma_1) = 1$, output $\mathcal{O}_{Sign}(\cdot, m, \Upsilon_2)$ via calling oracle $\mathcal{O}_{Sign}$.

The simulator returns a distinct handle for each group element that is associated with a distinct function over the formal variables.

After the adversary outputs a purported forgery signature σ^* on a policy Υ^* and message m^* such that $(m^*, \Upsilon^*) \neq (m^{(q)}, \Upsilon^{(q)})$ for all q. Let $\mathcal{M}^* \in \mathbb{Z}_p^{l^* \times t^*}$ be the corresponding monotone span program with row labeling $u^*(\cdot)$. Let $\mu^* = \mathcal{H}(m^* || \Upsilon^*)$, and suppose the signature has the form $\sigma^* = (g^{h_0^*}, g^{h_1^*}, g^{y^*}, g^{w^*}, g^{s_1^*}, \cdots, g^{s_l^*}, g^{p_1^*}, \cdots, g^{p_t^*})$.

Then each of these group elements must be associated with a function which must be a multilinear combination of the functions given as input to the adversary (public key, signing keys, re-signing keys, signature query responses, and re-signature query responses).

To be a forgery, we need $y^* \neq 0$, and $w^* \cdot \Delta_0 a_0 + \Delta_m^* \cdot \Delta_0 = (y^* + \Delta_m^*) \cdot \Delta_0$, and

$$\Delta_m^* \Delta_1 + \sum_{i=1}^{l^*} s_i^* \mathcal{M}_{i,1}^* (a + u^*(i) b) \Delta_1 = (y^* + \Delta_m^*) \Delta_1 + (c + \mu^*) p_1^*,$$

$$\sum_{i=1}^{l^*} s_i^* \mathcal{M}_{i,j}^* (a + u^*(i) b) \Delta_j = (c + \mu^*) p_j^* \quad \text{for} \quad (j > 1 \ \ \& \ \ j \in [t]),$$

for a random assignment of the formal variables.

This is identical to the requirement in [MPR11] which is $w^* = y^*/a_0$, and

$$\sum_{i=1}^{l^*} s_i^* \mathcal{M}_{i,j}^* (a + u^*(i) b) \Delta_j = y^* z_j \Delta_j + (c + \mu^*) p_j^* \quad (\forall j \in [t^*]).$$

The difference between our scheme and [MPR11] does not introduce any new items into the multilinear homogeneous polynomial sets to which y^* and p_j^* belongs.

Identical to the proof in [MPR11]. We assume these constraints are functionally equivalent, and eventually obtaining a contradiction: that there exists a $k_0 \in [n]$ such that $\Upsilon^*(\mathcal{A}_{k_0}) = 1$. In other words, the adversary could have generated a signature legitimately with the signing key for $\mathcal{A}_{k_0}$, and thus the output is not a forgery. More details can be found in [MPR11].

Compared with the security model about unforgeability in [MPR11], our security model adds the ReKeyGen oracle and the ReSign oracle.It is worth noting that

even when the RekeyGen oracle is added, there is no change about the multilinear homogeneous polynomial sets to which y^* and p_j^* belongs. Furthermore, the ReSign oracle is implemented by calling the Sign oracle. It also does not introduce any change about the multilinear homogeneous polynomial sets to which y^* and p_j^* belongs. So the security about unforgeability of this scheme, including both external security and internal security, have been proved. □

7.6 Summary

We proposed the notion of attribute-based proxy re-signature. Using this cryptographic primitive, a semi-trusted proxy can convert a signature satisfying a predicate into a signature satisfying another different predicate on the same message. At the same time, the proxy cannot learn any signing key and cannot sign arbitrary messages on behalf of either the delegator or the delegatee by himself. Our scheme is multi-use unidirectional, which means the proxy is only able to translate in one direction and signatures can be re-translated several times. More important, the size of signatures and the verification cost do not grow linearly with the number of translations in our scheme. This solves the open problem left by Libert and Vergnaud at CCS 2008. We also proved that our scheme is existentially unforgeable under the generic group model and equipped with the attribute-privacy property, which is a fundamental requirement of attribute-based signatures.

Compared with existing schemes, the goal of letting the size of proxy re-signature does not blow up with the number of translations is achieved at the expense of the unforgeability property is only proved in the generic group model.

Chapter 8

Conclusion

8.1 Summary of Contributions

In this thesis, we investigate several cryptographic primitives in the identity-based setting and the attribute-based setting. The contributions of this thesis can be summarized in the following four aspects: digital signatures with message recovery both in the identity-based multisignature setting and the attribute-based setting, identity-based authenticated encryption with authenticated header, identity-based quotable ring signature, attribute-based proxy re-signature.

In Chapter 3, an identity-based multisignature with message recovery scheme was proposed. In this scheme, multiple signers generate a single constant size multisignature on the same message regardless of the number of signers. The size of the multisignature is the same as that of a signature generated by one signer. Furthermore, it does not require transmission of the original message to verify the multisignature, since the original message can be recovered from the multisignature. Therefore, this scheme minimizes the total length of the original message and the appended multisignature. The proposed scheme is proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the CDH problem is hard.

In Chapter 4, a generic construction of identity-based authenticated encryption with authenticated header was proposed. The generic construction composed of an identity-based signature with message recovery scheme and a symmetric encryption scheme. This notion has potential applications in the environment with big data. Using this cryptographic primitive, everyone can check the validity of the authenticated ciphertext and access to the authenticated header; only the designated receiver can recover the payload. Additionally, the designated receiver is given the liberty

whether to decrypt the payload or not by only checking the corresponding authenticated header. The construction is proven to be existentially unforgeable against adaptively chosen message attacks and indistinguishable under adaptive chosen ciphertext attacks, given the underlying identity-based signature with message recovery scheme and the symmetric encryption scheme are secure in the same manner, respectively. We also gave a concrete instance of the identity-based authenticated encryption with authenticated header scheme based on bilinear pairing, and gave concrete security analysis of the instance. The instance addressed a problem which was not considered in the symmetric key setting counterpart.

In Chapter 5, an identity-based quotable ring signature scheme was proposed. In this scheme, we extended the ring signature scheme to be quotable. Using this cryptographic primitive, anyone could derive new ring signatures on a substring of an original message from a ring signature on the original message. There are two different types of ring signatures. The first one could be quoted further down to these two types of ring signatures. The other one could not be quoted any further, but will be a shorter signature. We also proved that our scheme is anonymous under the assumption that the subgroup decision problem is hard, selectively unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the CDH problem is hard, and strongly context hiding, which means the verifier could not distinguish whether a signature is originally generated or is quoted from another ring signature.

In Chapter 6, two attribute-based signature with message recovery schemes were proposed. The first scheme allows the signer to embed short original message in the signature without the need of sending the original message to the verifier, while keeping the same signature size. The second scheme is extended from the first one in order to deal with large messages. The signature size of the new schemes is the same as that of existing attribute-based signature scheme, but it does not require the transmission of the original message for the signature verification. The proposed schemes support flexible threshold predicates and are proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the Computational Diffie-Hellman problem is hard. We demonstrate that the proposed schemes are also equipped with the attribute privacy property.

In Chapter 7, an attribute-based proxy re-signature scheme was proposed. In

this scheme, only the designated proxy who gets some secret information of the delegator can re-sign the original signature. The semi-trusted proxy acts as a translator to convert a signature satisfying one predicate into a signature satisfying another different predicate on the same message, while the proxy cannot learn any signing key and cannot sign arbitrary messages on behalf of either the delegator or the delegatee. Our scheme is multi-use unidirectional, which means the proxy is only able to translate in one direction and signatures can be re-translated several times. More important, the size of signatures and the verification cost do not grow linearly with the number of translations in our scheme. It solves an open problem left by Libert and Vergnaud at CCS 2008 [LV08]. We also proved that our scheme is existentially unforgeable under the generic group model and equipped with the attribute-privacy property, which is a fundamental requirement of attribute-based signatures.

8.2 Future Work

In our identity-based quotable ring signature scheme, there are two different types of ring signatures. The first one could be quoted further down to these two types of ring signatures. The other one could not be quoted any further, but will be a shorter signature. Finding such a scheme that both could be quoted further and enjoys a shorter signature size might be an interesting future work.

Although our attribute-based proxy re-signature scheme achieves the goal of finding out multi-use unidirectional proxy re-signature scheme where the size of signatures and the verification cost do not grow linearly with the number of translations, the scheme is proved in the generic group model. Finding such a scheme that can be proved in more standard hard problem assumptions might be another interesting future work.

Bibliography

[ABC+12] JaeHyun Ahn, Dan Boneh, Jan Camenisch, Susan Hohenberger, abhi shelat, and Brent Waters. Computing on authenticated data. In Ronald Cramer, editor, *Theory of Cryptography*, volume 7194 of *Lecture Notes in Computer Science*, pages 1–20. Springer Berlin Heidelberg, 2012.

[ACdMT05] Giuseppe Ateniese, DanielH. Chou, Breno de Medeiros, and Gene Tsudik. Sanitizable signatures. In SabrinadeCapitani di Vimercati, Paul Syverson, and Dieter Gollmann, editors, *Computer Security ESORICS 2005*, volume 3679 of *Lecture Notes in Computer Science*, pages 159–177. Springer Berlin Heidelberg, 2005.

[ADR02] JeeHea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In LarsR. Knudsen, editor, *Advances in Cryptology EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer Berlin Heidelberg, 2002.

[AH05] Giuseppe Ateniese and Susan Hohenberger. Proxy re-signatures: New definitions, algorithms, and applications. In *Proceedings of the 12th ACM Conference on Computer and Communications Security*, CCS '05, pages 310–319, New York, NY, USA, 2005. ACM.

[ALP13] Nuttapong Attrapadung, Benot Libert, and Thomas Peters. Efficient completely context-hiding quotable and linearly homomorphic signatures. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 386–404. Springer Berlin Heidelberg, 2013.

[AO99] Masayuki Abe and Tatsuaki Okamoto. A signature scheme with message recovery as secure as discrete logarithm. 1716:378–389, 1999.

[AOS02] Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n signatures from a variety of keys. In Yuliang Zheng, editor, *Advances in Cryptology ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 415–432. Springer Berlin Heidelberg, 2002.

[BBD+10] Christina Brzuska, Heike Busch, Oezguer Dagdelen, Marc Fischlin, Martin Franz, Stefan Katzenbeisser, Mark Manulis, Cristina Onete, Andreas Peter, Bertram Poettering, and Dominique Schrder. Redactable signatures for tree-structured data: Definitions and constructions. In Jianying Zhou and Moti Yung, editors, *Applied Cryptography and Network Security*, volume 6123 of *Lecture Notes in Computer Science*, pages 87–104. Springer Berlin Heidelberg, 2010.

[BBS98] Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In Kaisa Nyberg, editor, *Advances in Cryptology EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 127–144. Springer Berlin Heidelberg, 1998.

[Bei96] Amos Beimel. *Secure schemes for secret sharing and key distribution*. PhD thesis, PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[BF01] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *Advances in Cryptology CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer Berlin Heidelberg, 2001.

[BF11a] Dan Boneh and DavidMandell Freeman. Homomorphic signatures for polynomial functions. In KennethG. Paterson, editor, *Advances in Cryptology EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 149–168. Springer Berlin Heidelberg, 2011.

[BF11b] Dan Boneh and DavidMandell Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *Public Key Cryptography PKC 2011*, volume 6571 of *Lecture Notes in Computer Science*, pages 1–16. Springer Berlin Heidelberg, 2011.

[BFF+09] Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus Page, Jakob Schelbert, Dominique Schrder, and Florian Volk. Security of sanitizable signatures revisited. In Stanisaw Jarecki and Gene Tsudik, editors, *Public Key Cryptography PKC 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 317–336. Springer Berlin Heidelberg, 2009.

[BFLS10] Christina Brzuska, Marc Fischlin, Anja Lehmann, and Dominique Schrder. Unlinkability of sanitizable signatures. In PhongQ. Nguyen and David Pointcheval, editors, *Public Key Cryptography PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 444–461. Springer Berlin Heidelberg, 2010.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, pages 103–112, New York, NY, USA, 1988. ACM.

[BGG94] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Incremental cryptography: The case of hashing and signing. In YvoG. Desmedt, editor, *Advances in Cryptology CRYPTO 94*, volume 839 of *Lecture Notes in Computer Science*, pages 216–233. Springer Berlin Heidelberg, 1994.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In Joe Kilian, editor, *Theory of Cryptography*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer Berlin Heidelberg, 2005.

[BKM06] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, volume 3876 of *Lecture Notes in Computer Science*, pages 60–79. Springer Berlin Heidelberg, 2006.

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition

paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545. Springer Berlin Heidelberg, 2000.

[BN08] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *Journal of Cryptology*, 21(4):469–491, 2008.

[Bol02] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. 2567:31–46, 2002.

[Bon98] Dan Boneh. The decision diffie-hellman problem. In JoeP. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer Berlin Heidelberg, 1998.

[Boy03] Xavier Boyen. Multipurpose identity-based signcryption. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 383–399. Springer Berlin Heidelberg, 2003.

[BR00] Mihir Bellare and Phillip Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In Tatsuaki Okamoto, editor, *Advances in Cryptology ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 317–330. Springer Berlin Heidelberg, 2000.

[BRW04] Mihir Bellare, Phillip Rogaway, and David Wagner. The eax mode of operation. In Bimal Roy and Willi Meier, editors, *Fast Software Encryption*, volume 3017 of *Lecture Notes in Computer Science*, pages 389–407. Springer Berlin Heidelberg, 2004.

[BS98] Matt Blaze and Martin Strauss. Atomic proxy cryptography. In *Proc. EuroCrypt'97*. Citeseer, 1998.

[BSS02] Emmanuel Bresson, Jacques Stern, and Michael Szydlo. Threshold ring signatures and applications to ad-hoc groups. In Moti Yung, editor, *Advances in Cryptology CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 465–480. Springer Berlin Heidelberg, 2002.

[BSW07] J. Bethencourt, A Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Security and Privacy, 2007. SP '07. IEEE Symposium on*, pages 321–334, May 2007.

[BW06] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 427–444. Springer Berlin Heidelberg, 2006.

[Cam97] Jan Camenisch. Efficient and generalized group signatures. In Walter Fumy, editor, *Advances in Cryptology EUROCRYPT 97*, volume 1233 of *Lecture Notes in Computer Science*, pages 465–479. Springer Berlin Heidelberg, 1997.

[CHC02] JaeCha Choon and Jung Hee Cheon. An identity-based signature from gap diffie-hellman groups. In YvoG. Desmedt, editor, *Public Key Cryptography PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 18–30. Springer Berlin Heidelberg, 2002.

[CLX09] Ee-Chien Chang, CheeLiang Lim, and Jia Xu. Short redactable signatures using random trees. In Marc Fischlin, editor, *Topics in Cryptology CT-RSA 2009*, volume 5473 of *Lecture Notes in Computer Science*, pages 133–147. Springer Berlin Heidelberg, 2009.

[Coc01] Clifford Cocks. An identity-based encryption scheme based on quadratic residues. In Bahram Honary, editor, *Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer Berlin Heidelberg, 2001.

[CvH91] David Chaum and Eugne van Heyst. Group signatures. In DonaldW. Davies, editor, *Advances in Cryptology EUROCRYPT 91*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer Berlin Heidelberg, 1991.

[CYHC04] ShermanS.M. Chow, S.M. Yiu, LucasC.K. Hui, and K.P. Chow. Efficient forward and provably secure id-based signcryption scheme with public verifiability and public ciphertext authenticity. In Jong-In Lim and Dong-Hoon Lee, editors, *Information Security and Cryptology -*

ICISC 2003, volume 2971 of *Lecture Notes in Computer Science*, pages 352–369. Springer Berlin Heidelberg, 2004.

[DH76] W. Diffie and M.E. Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, Nov 1976.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In AndrewM. Odlyzko, editor, *Advances in Cryptology CRYPTO 86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer Berlin Heidelberg, 1987.

[GGDS06] R. Gangishetti, M.C. Gorantla, M.L. Das, and A. Saxena. Identity-based multisignatures. *Informatica*, 17(2):177–186, 2006.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[GMW86] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, 0:174–187, 1986.

[GNSN13] Martin Gagn, Shivaramakrishnan Narayan, and Reihaneh Safavi-Naini. Short pairing-efficient threshold-attribute-based signature. In Michel Abdalla and Tanja Lange, editors, *Pairing-Based Cryptography Pairing 2012*, volume 7708 of *Lecture Notes in Computer Science*, pages 295–313. Springer Berlin Heidelberg, 2013.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for np. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358. Springer Berlin Heidelberg, 2006.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, pages 89–98, New York, NY, USA, 2006. ACM.

[GQ90] LouisClaude Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In Shafi Goldwasser, editor, *Advances in Cryptology CRYPTO 88*, volume 403 of *Lecture Notes in Computer Science*, pages 216–231. Springer New York, 1990.

[Her04] Javier Herranz. A formal proof of security of zhang and kim's id-based ring signature scheme. *WOSIS*, 4:63–72, 2004.

[Hes03] Florian Hess. Efficient identity-based signature schemes based on pairings. In Kaisa Nyberg and Howard Heys, editors, *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 310–324. Springer Berlin Heidelberg, 2003.

[HR08] L. Harn and J. Ren. Efficient identity-based rsa multisignatures. *computers & security*, 27(1):12–15, 2008.

[HS03a] Javier Herranz and Germán Sáez. Forking lemmas for ring signature schemes. In Thomas Johansson and Subhamoy Maitra, editors, *Progress in Cryptology - INDOCRYPT 2003*, volume 2904 of *Lecture Notes in Computer Science*, pages 266–279. Springer Berlin Heidelberg, 2003.

[HS03b] Javier Herranz and Germán Sáez. A provably secure id-based ring signature scheme. In *eprint*. Citeseer, 2003.

[ID03] Anca-Andreea Ivan and Yevgeniy Dodis. Proxy cryptography revisited. In *NDSS*, 2003.

[IN83] Kazuharu Itakura and Katsuhiro Nakamura. A public-key cryptosystem suitable for digital multisignatures. *NEC Research & Development*, (71):1–8, 1983.

[JMSW02] Robert Johnson, David Molnar, Dawn Song, and David Wagner. Homomorphic signature schemes. In Bart Preneel, editor, *Topics in Cryptology CT-RSA 2002*, volume 2271 of *Lecture Notes in Computer Science*, pages 244–262. Springer Berlin Heidelberg, 2002.

[Kha07a] Dalia Khader. Attribute based group signature with revocation. *IACR Cryptology ePrint Archive*, 2007:241, 2007.

[Kha07b] Dalia Khader. Attribute based group signatures. *IACR Cryptology ePrint Archive*, 2007:159, 2007.

[Kha08] Dalia Khader. Authenticating with attributes. *IACR Cryptology ePrint Archive*, 2008:31, 2008.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, 2007.

[Kra01] Hugo Krawczyk. The order of encryption and authentication for protecting communications (or: How secure is ssl?). In Joe Kilian, editor, *Advances in Cryptology CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 310–331. Springer Berlin Heidelberg, 2001.

[KVW04] Tadayoshi Kohno, John Viega, and Doug Whiting. Cwc: A high-performance conventional authenticated encryption mode. In Bimal Roy and Willi Meier, editors, *Fast Software Encryption*, volume 3017 of *Lecture Notes in Computer Science*, pages 408–426. Springer Berlin Heidelberg, 2004.

[LAS+10] Jin Li, Man Ho Au, Willy Susilo, Dongqing Xie, and Kui Ren. Attribute-based signature and its applications. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '10, pages 60–69, New York, NY, USA, 2010. ACM.

[LK08] Jin Li and Kwangjo Kim. Attribute-based ring signatures. *IACR Cryptology ePrint Archive*, 2008:394, 2008.

[LQ03] B. Libert and J-J Quisquater. A new identity-based signcryption scheme from pairings. In *Information Theory Workshop, 2003. Proceedings. 2003 IEEE*, pages 155–158, March 2003.

[LV08] Benoît Libert and Damien Vergnaud. Multi-use unidirectional proxy re-signatures. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, pages 511–520, New York, NY, USA, 2008. ACM.

[LW04] Chih-Yin Lin and Tzong-Chen Wu. An identity-based ring signature scheme from bilinear pairings. In *18th International Conference on Advanced Information Networking and Applications (AINA'04)*, volume 2, page 182, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[Lyn02] Ben Lynn. Authenticated identity-based encryption. *IACR Cryptology ePrint Archive*, 2002:72, 2002.

[Mao04] Wenbo Mao. Modern cryptography: theory and practice. *Publisher: Prentice Hall PTR, Copyright: Hewlett Packard*, 2004.

[Mau05] Ueli Maurer. Abstract models of computation in cryptography. In NigelP. Smart, editor, *Cryptography and Coding*, volume 3796 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg, 2005.

[ML02] John Malone-Lee. Identity-based signcryption. *IACR Cryptology ePrint Archive*, 2002:98, 2002.

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures: Extended abstract. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, CCS '01, pages 245–254, New York, NY, USA, 2001. ACM.

[MOV93] Alfred J Menezes, Tatsuaki Okamoto, and Scott A Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *Information Theory, IEEE Transactions on*, 39(5):1639–1646, Sep 1993.

[MPR08] Hemanta K Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. *IACR Cryptology ePrint Archive*, 2008:328, 2008.

[MPR11] HemantaK. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In Aggelos Kiayias, editor, *Topics in Cryptology CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 376–392. Springer Berlin Heidelberg, 2011.

[MUO96a] Masahiro Mambo, Keisuke Usuda, and Eiji Okamoto. Proxy signatures: Delegation of the power to sign messages. *IEICE transactions on*

fundamentals of electronics, communications and computer sciences, 79(9):1338–1354, 1996.

[MUO96b] Masahiro Mambo, Keisuke Usuda, and Eiji Okamoto. Proxy signatures for delegating signing operation. In *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, CCS '96, pages 48–57, New York, NY, USA, 1996. ACM.

[NR93] Kaisa Nyberg and Rainer A. Rueppel. A new signature scheme based on the dsa giving message recovery. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 58–61, New York, NY, USA, 1993. ACM.

[Oka99] Tatsuaki Okamoto. Multisignature schemes secure against active insider attacks. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 82(1):21–31, 1999.

[OO99] Kazuo OHTA and Tatsuaki OKAMOTO. Multisignature schemes secure against active insider attacks. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 82(1):21–31, 1999.

[Pas03] Rafael Pass. On deniability in the common reference string and random oracle model. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 316–337. Springer Berlin Heidelberg, 2003.

[PSST11] KennethG. Paterson, JacobC.N. Schuldt, Martijn Stam, and Susan Thomson. On the joint security of encryption and signature, revisited. In DongHoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 161–178. Springer Berlin Heidelberg, 2011.

[Riv00] Ronald Rivest. Two signature schemes. *Slides from talk given at Cambridge University, October 17*, 2000. http://people.csail.mit.edu/rivest/Rivest-CambridgeTalk.pdf.

[Rog02] Phillip Rogaway. Authenticated-encryption with associated-data. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 98–107, New York, NY, USA, 2002. ACM.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, February 1978.

[RST01] RonaldL. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *Advances in Cryptology ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer Berlin Heidelberg, 2001.

[RST06] RonaldL. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret: Theory and applications of ring signatures. In Oded Goldreich, ArnoldL. Rosenberg, and AlanL. Selman, editors, *Theoretical Computer Science*, volume 3895 of *Lecture Notes in Computer Science*, pages 164–186. Springer Berlin Heidelberg, 2006.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In GeorgeRobert Blakley and David Chaum, editors, *Advances in Cryptology*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer Berlin Heidelberg, 1985.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology EUROCRYPT 97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer Berlin Heidelberg, 1997.

[SSN09] SiamakF. Shahandashti and Reihaneh Safavi-Naini. Threshold attribute-based signatures and their application to anonymous credential systems. In Bart Preneel, editor, *Progress in Cryptology AFRICACRYPT 2009*, volume 5580 of *Lecture Notes in Computer Science*, pages 198–216. Springer Berlin Heidelberg, 2009.

[SW07] Hovav Shacham and Brent Waters. Efficient ring signatures without random oracles. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography PKC 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 166–180. Springer Berlin Heidelberg, 2007.

[SY08] Guo Shaniqng and Zeng Yingpei. Attribute-based signature scheme. In *Information Security and Assurance, 2008. ISA 2008. International Conference on*, pages 509–511, April 2008.

[TLW03] Chunming Tang, Zhupjun Liu, and Mingsheng Wang. An improved identity-based ring signature scheme from bilinear pairings. *NM Research Preprints*, pages 231–234, 2003.

[Wat11] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *Public Key Cryptography PKC 2011*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70. Springer Berlin Heidelberg, 2011.

[YCD08] Piyi Yang, Zhenfu Cao, and Xiaolei Dong. Fuzzy identity-based signature. *IACR Cryptology ePrint Archive*, 2008:2, 2008.

[Zhe97] Yuliang Zheng. Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In Jr. Kaliski, BurtonS., editor, *Advances in Cryptology CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 165–179. Springer Berlin Heidelberg, 1997.

[ZK02] Fangguo Zhang and Kwangjo Kim. Id-based blind signature and ring signature from pairings. In Yuliang Zheng, editor, *Advances in Cryptology ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 533–547. Springer Berlin Heidelberg, 2002.

[ZSM05] Fangguo Zhang, Willy Susilo, and Yi Mu. Identity-based partial message recovery signatures (or how to shorten id-based signatures). In AndrewS. Patrick and Moti Yung, editors, *Financial Cryptography and Data Security*, volume 3570 of *Lecture Notes in Computer Science*, pages 45–56. Springer Berlin Heidelberg, 2005.