2013

Contributions to pairing-based digital signatures

Yunmei Zhang
University of Wollongong

Contributions to Pairing-Based Digital Signatures

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

Master of Computer Science by Research

from

UNIVERSITY OF WOLLONGONG

by

Yunmei Zhang

School of Computer Science and Software Engineering
September 2013

Dedicated to
My mother and my father

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.

Yunmei Zhang
September 20, 2013

Abstract

Nowadays, electronic communication plays a key role in the way people communicate in business or financial transactions. As e-commerce becomes more and more popular, the demand for digital signature is increasing rapidly. In 1976 Whitfield Diffie and Martin Hellman introduced the concept of digital signature [31] which is used to demonstrate the authenticity of a message or document. In 1977, Ronald Rivest, Adi Shamir and Len Adleman [66] proposed the notion of the RSA algorithm based on the factoring problem. In addition to the RSA signature, other signatures such as ElGamal signature [34], Rabin signature [65], Pairing-based signature [14], Undeniable signature [21] and others have been proposed by a number of different researchers.

Due to the fact that users can enjoy properties such as authentication of the message, integrity of the message and non-repudiation of the message, digital signature has partially replaced the original ink on paper signatures. However, there exist a number of problems and potential attacks on digital signatures. It is observed that the majority of IBS schemes have the weakness of private key escrow. Additionally, existing solutions for security model could be made simpler and much more

practical. In this thesis, two different pairing-based signatures: efficient escrow free identity based signature [88] and (strong) multi-designated verifiers signatures secure against rogue key attack - are proposed to enhance the security of the pairing-based signature against a number of attacks.

This thesis addresses two problems in the two different pairing-based signatures mentioned earlier and comes up with solutions that is neat, correct, secure and efficient.

Acknowledgements

I would like to express my sincere gratitude to the people who have assisted me in completing this thesis.

I would like to sincerely acknowledge my supervisor Dr. Man Ho Au for his guidance and patience. It has been an honor to be his first Master Degree student. I highly appreciate his teaching efforts and encouragement to guide me through this research process.

I would like to thank my co-supervisor Professor Willy Susilo for his invaluable advice and suggestions. Without his encouragement and abundant help this work would not have been completed. I owe sincere and earnest thanks to my research collaborators: Dr. Joseph Liu, Dr. Xinyi Huang and Dr. Guomin Yang for their help and support throughout my research.

Last but not least, I would like to thank my parents for their generous financial support and care, as well as all my best friends, particularly Jie Liu, Ron, Zoe, Aziz and Elmin for their support and encouragement.

Publications

The following papers have been published, and they are related to the work written in this thesis.

1. Yunmei Zhang, Joseph Liu, Xinyi Huang, Man Ho Au and Willy Susilo, Efficient Escrow-Free Identity-Based Signature, The Sixth International Conference on Provable Security (ProvSec 2012), Lecture Notes in Computer Science 7496, pp. 161 - 174, 2012.

2. Yunmei Zhang, Man Ho Au, Guomin Yang and Willy Susilo, (Strong) Multi-Designated Verifiers Signatures Secure Against Rogue Key Attack, The 6th International Conference on Network and System Security (NSS 2012), Lecture Notes in Computer Science 7645, 2012.

Contents

List of Tables

Chapter 1

Introduction

To accommodate a new business model, people have been conducting electronic-commerce (E-commerce) on the internet. The concept of e-commerce is associated with selling or buying products or services between businesses and consumers through the internet. As e-commerce has both adversary and cooperative activities [48], there is a growing demand for applications to enhance telecommunication privacy, so that an adversary cannot alter the message or eavesdrop on the communication.

1.1 Public-key Cryptography

One of the approaches to enhance security is to apply public-key cryptography which is proposed in Whitfield Diffie and Martin Hellman's seminal paper "New directions in Cryptography" [31]. Public-key cryptography is an asymmetric cryptosystem which uses a pair of private and public keys. The public key can be used for encryption while the corresponding private key is used for decryption. Although the

private key and public key have mathematical relationship in public-key cryptography, it is computationally unfeasible to derive a private key from a public key which is known by the public. In comparison to a symmetric cryptosystem, it does not require a secure channel to transmit a shared key in advance between the sender and receiver. In public-key cryptography, an entity called certification authority (CA) is responsible for issuing digital certificates which contain a public key together with the identity of the user in the system. The digital certificate certifies the authenticity of the public key which is associated with the corresponding private key, thus the end users can trust the certificates if they are published by the CA.

1.2 Digital Signature

The public key cryptography is used in digital signature where can enhance the security requirements such as authentication, integrity and non-requdidation. The concept of digital signature was proposed by Whitfield Diffie and Martin Hellman in 1976 [31]. In this system, a user possesses a public key and a secret key. Given the secret key, the user can create a signature which can be verified by anyone based on the user's public key, message and public parameters of the system.

The digital signature has equivalent function as the handwritten signature. The advantages of digital signature include authentication and integrity of the message in which any changes of messages would be detectable and finally, non-repudiation in which the signer cannot deny the fact that he/she has signed the signature. A

typical digital signature scheme consists of three algorithms including key generation algorithm, signing algorithm and verifying algorithm. With regard to key generation algorithm, given the public parameters, it outputs the private key and a corresponding public key. For signing algorithm, the user puts a signature on a message with the user's private key. Finally, for verifying algorithm, it proves the validity of the signature based on the signer's public key with the associated message. A pair of public key and private key can be generated by using the key generation algorithm. The user should obtain a digital certificate which contains a public key together with the trustworthy personal information of the users from CA. Receiving the certificate, users can sign the message or document and send the signature with the certificate. As CAs have a good reputation and reliability, the receiver can validate the signature according to sender's public key included in the digital certificate. In addition, the receiver also can check whether the certificate is signed by a CA. However, in the digital signature, in particular the use of public-key cryptosystem, the corresponding certificate verification is required.

There are a number of security considerations when implementing digital signature schemes in electronic transactions since there exist different attackers with different goals. The goal of attackers on digital signatures can be total break, selective forgery and existential forgery [38]. In terms of total break, the adversary derives secret keys from public keys, thus forging any signatures on any messages. For selective forgery, the adversary can create a valid signature on a message chosen by others with a significant probability. In relation to existential forgery, the adversary can create a pair of message and signature in a way that the signature of message

Goal	Describtion
Total Break	Given public keys, attackers can recover secret keys.
Selective Forgery	Attackers can forge a signature on a message chosen by someone else.
Existential Forgery	Adversaries can forge a signature for at least one new message.

Table 1.1: The Goal of Attackers

is valid. In order to enhance security, several major attacks should be addressed in digital signatures. In a key-only attack, the attacker only knows the public key of the signer whereas in a known message attack, the attacker knows signatures for a variety of messages, which are not chosen by the attacker. Additionally, in generic chosen-message attack, the attackers can choose any messages they want and obtain corresponding signatures before they attempt to break the signature. For adaptive chosen-message attack, the adversary can request signatures on any messages depending on previously obtained message signature pairs. Overall, we should construct a signature scheme with strong security requirements by considering various attacks and adversary goals which are described in Table 1.1 and Table 1.2 in details.

Attack	Describtion
Key-Only Attack	Attackers can only access to public keys.
Known-Message Attack	Attackers have signatures together with corresponding messages.
Generic Chosen-Message Attack	Attackers can choose any messages they want and obtain corresponding signatures.
Adaptive Chosen-Message Attack	Attackers can request signatures on any messages depending on previously obtained message signature pairs.

Table 1.2: Different Types of Attacks

1.3 Digital Signature with Different Settings

Many researchers have proposed several digital signatures with different settings [31, 71, 35, 1]. These settings can be summarized as follows.

1.3.1 Public Key Infrastructure (PKI)

A public key infrastructure consists of a certificate authority,a registration authority (RA),directories for keeping certificates and a certificate management system. The role of the RA is to verify user's request and ask CA to issue a digital certificate to the requestor. In public key cryptography [31], Alice possesses a pair of private key and public key. The private key can only be known by Alice whereas the public key can be accessed by all parties. Alice can put a signature using her secret key and send it to a receiver Bob. In order to provide authentication of the signer, Alice

needs to obtain a certificate which includes the public key and identity of herself. Before Bob verifies the signature received by Alice, he needs to verify the validity of her certificate. Overall, two verification steps are needed when verifiers want to verify independent signatures.

1.3.2 Identity-Based Cryptography (IBC)

The concept of identity-based cryptography was proposed by Shamir [71]. In IBC, an exchange of public keys or private keys and keeping key directories are un-necessary when verifying each other's signature. This is a special public-key cryptosystem in which the public key can be any arbitrary string. An entity namely the private key generator (PKG) is responsible to compute the private key with respect to the identities for all users in the system. Comparing to the public-key cryptosystem, it avoids the complicated and costly certificate verification which ensures the authenticity of public key. In this system, the signer creates a signature under the signer's secret key and sends it to the receiver. Upon receiving the signature, the receiver can verify the signature based on the signer's public key and associated message. With regard to advantages of IBS, the need for a public key distribution infrastructure is un-necessary and the need for checking certificate verification can be eliminated as public keys can be derived from the identity of users. Though the identity-based signature has many advantages over public-key cryptosystems, it has an inherent drawback which is key escrow problem. PKG can always impersonate users because the PKG is responsible for creating secret keys for all users.

1.3.3 Certificate-Based Cryptography (CBC)

The notion of certificate-based cryptography was introduced by Gentry [35]. It preserves most of the advantages of PKI and IBC. In CBC, a user generates a pair of public key and secret key for himself/herself and request a certificate from the CA. The CA generates the certificate using IBE scheme. A user needs both private key and certificate from CA to decrypt or sign a message. The concept of certificate-based signature (CBS) was proposed by B. G. Kang et al [49] with the same public parameters and certificate revocation strategy of CBE.

1.3.4 Certificateless Public Key Cryptography (CL-PKC)

The concept of Certificateless Public Key Cryptography (CL-PKC) was proposed by Al-Riyami and Paterson [1]. The use of certificates is not requried in CL-PKC. In CL-PKC system, there is an entity named key generating center (KGC) helping users to generate partial secret key. Firstly, the KGC generates a partial private key by public parameters, master key of KGC and an identity of a user. Receiving the partial private key, the user can choose a secret value which can be used in computing a secret key. Taking the secret value, public parameters and the user's partial secret key, the user can generate a secret key. After that the user can compute his/her public key which can be known by the public according to the secret key. The user's secret key is not available to the KGC or other parties. Finally, the user can encrypt or decrypt the message based on the private key and secret key. In terms of signature, Al-Riyami and Paterson described a construction of certificateless public

key signature (CL-PKS) [1]. The algorithms used before verifying and signing are essentially the same as the algorithms in CL-PKC. With regard to signing, a user can put a signature on a message with private key and public parameters. As to verifying, receivers can verify the signatures on the associated message with public parameters, identifier and public key of the signer. However, this system is no longer identity-based because the public key cannot be computed from an identity itself.

1.4 Digital Signatures with Different Properties

There are several types of digital signatures with different properties such as verifiable encrypted signature [12, 2], directed signature [57, 54], blind signature [19], undeniable signature [22], designated verifier signature [46, 20], aggregate signature [12], group signature [23], ring signature [68], threshold signature [29, 74, 84] and multi-designated verifier signature [46, 50, 61].

1.4.1 Verifiable Encrypted Signature

In verifiable encrypted signature, a user Alice can put a signature on a message and encrypt it under the public key of a trusted third party. Receiver Bob obtains the signature created by Alice along with a proof that she has put a valid encryption of her signature [12]. Regardless Bob can verify that the signature is created by the Alice, but he does not know any information about her signature. In some cases when Alice is unable or unwilling to disclose her signature, Bob can ask the trusted

third party to reveal Alice's signature [2] With this property, verifiable encrypted signatures are used in fair exchange processes.

1.4.2 Blind Signature

A blind signature scheme [19] is a type of digital signature in which a signer can create a valid signature on a message without knowing the content of the message. In other words, the content of the message will be enclosed before the message is signed by a user. This can be used in some digital schemes where the privacy of senders is highly concerned. For instance, the blind signature scheme are commonly used in E-cash systems where users can make the transaction of e-cash anonymously. In addition, it is widely used in E-voting systems where any voter can choose secret ballots electronically.

1.4.3 Directed Signature

The concept of directed signature was introduced by Lim and Lee [57]. In directed signature, signature receivers can control the verification of the signature. This indicates that without the cooperation of the signature receiver, the validity of signatures cannot be processed. Directed signatures are used in some situations such as medical records, tax information or personal/business transactions in which the signed message is sensitive to the receivers [54].

1.4.4 Undeniable Signature

To verify a signature in undeniable signature, the cooperation of signer is needed. However, if the signer is malicious, this dishonest signer would refuse to interact with the verifier. To address this problem, undeniable signature has a distinct property which can check the validity of the signature using disavowal protocol [22].

1.4.5 Designated verifier Signature

Designated verifier signature/proofs (DVS/DVP) was proposed by Jakobsson, Sako and Impagliazzo [46], and independently Chaum [20]. In DVS/DVP, the signer can choose the designated verifier who can only verify the signature created by the signer. It has additional property that the designated verifier cannot convince anyone that the signature is created by the signer.

1.4.6 Aggregate Signature

Aggregate signature [12] is a digital signature scheme in which different signatures on different messages created by different users can be aggregated into a single signature. The single signature can convince any verifier that these users indeed put the signatures on associated messages. As to size of the signature, the size of the aggregated signature will be identical to the size of each signature generated by each user.

1.4.7 Group Signature

In group signature [23], only the members of group can sign a signature and the receiver can verify the signature without knowing who actually signs the signature in that group. However, there is an additional entity named group manager who can reveal the identity of the actual signer when needed.

1.4.8 Ring Signature

Ring signature [68] is a type of digital signature which has similar properties to group signature. In ring signature, the signature is created by one of the possible signers in a particular group. It differs from group signature in which it is impossible to revoke the anonymity of the message signer.

1.4.9 Threshold Signature

In a (t, n) threshold signature scheme, only subsets with t or more group members can represent a group with n members to create or confirm a signature. A secret key is shared among a set of n group members in a way that any subset with at least t members can create a valid signature. As regards the verification, a receiver can check the validity of the signature by a verification protocol which is the same as in a regular signature scheme [29].

1.4.10 Multi-designated Verifiers Signature

As an extension of the DVS, Desmedt proposed the notion of multi-designated verifiers signatures (MDVS) in 2003. In MDVS, the number of designated verifiers can be pluralized instead of a single verifier. Afterwards, many researchers proposed a number of MDVS constructions [46, 50, 61] with different properties in various settings. However, rogue-key attack which was proposed by Jakobsson, Sako and Impagliazzo [46] is possible and threatening in real life, and thus security requirement against rogue-key attack should be improved in MDVS. In such cases, an adversary convinces the external party that the signature is created by the signer instead of designated verifiers. Another type of rogue key attack is that the adversary computes the public key according to the designated verifiers' public keys. Therefore, a signature which can be verified by other honest verifiers can be created. As a counter-measure to the rogue key attack, the adversary is required to produce a proof-of-knowledge of the secret key. However, applying the proof-of-knowledge of the secret key requires changing the setting which is costly and impractical. Therefore, we proposed an efficient MDVS scheme without disclosing the knowledge of the secret key. All these different types of digital signatures on different settings are summarized in Table 1.3.

1.5 Thesis Structure

The thesis is organized as follows: Chapter 1 is the introduction while Chapter 2 describes the background of two pairing-based digital signatures namely Efficient

		Settings			
		Public Key Infrastructure	Identity-Based Cryptography	Certificate-Based Cryptography	Certificateless Public Key Cryptography
Single User	Verifiable Encrypted Signature	[2]	[41]	[72]	[83]
	Blind Signature	[63]	[86]	[45]	[25]
	Directed Signature	[53]	[75]	?	[80]
	Undeniable Signature	[33]	[56]	?	[32]
	Designated Verifier Signature	[47]	[76]	?	[44]
Multi-User	Aggregate Signature	[12]	[36]	[58]	[87]
	Group Signature	[24]	[64]	[60]	?
	Ring Signature	[67]	[4]	[3]	[82]
	Threshold Signature	[74]	[5]	[84]	[17]
	Multi-designated Verifiers Signature	[51]	[26]	?	?

Table 1.3: Different Types of Signatures with Different Settings

Escrow Free Identity Based Signature (EF-IBS) and Strong Multi-Designated Verifiers Signatures (MDVS) Against Rogue Key Attack. Chapter 3 discusses the details regarding the syntax, construction and security analysis of EF-IBS. In Chapter 4, the details of syntax and construction of MDVS against rogue key attack are presented. Finally, Chapter 5 concludes this thesis. We will also present our future work in this line of research direction.

Chapter 2

Background

This chapter firstly introduces the preliminaries used including groups, abelian groups, rings, fields, finite fields and random oracle and standard model. Following that, the later part further describes a variety of complexity problems such as Discrete Logarithm, Computational Diffie-Hellman, Decisional Diffie-Hellman, Bilinear Diffie-Hellman, Decision Bilinear Diffie-Hellman and Strong RSA.

2.1 Preliminaries

2.1.1 Groups

A group $\mathcal{G}$ is a set of elements together with a binary operation o satisfying four properties such as closure, associativity, identity, and inverses. For instance, we denote Z_n as a group under addition in which $n \geq 1$ and the elements of Z_n is a set of $0, 1, 2, 3, 4..., n-1$. Some of the important definitions related to groups are as under.

Subgroup: A subgroup of a group $\mathcal{G}$ is a non-empty subset of group $\mathcal{G}$ under the same operation with group $\mathcal{G}$. Assume a subgroup of $\mathcal{G}$ is $\mathcal{H}$, the subgroup $\mathcal{H}$ is

denoted as $\mathcal{H} \subseteq \mathcal{G}$ and the proper subgroup can be denoted by $\mathcal{H} \subset \mathcal{G}$.

Order of a Group: The number of elements in a finite group $\mathcal{G}$ is named the order of a group.

Order of Group Element: Assume element $a \in \mathcal{G}$, the order of element a is the least positive integer $i \in N$ that satisfies a^i is equal to the identity e. The order of group element a is denoted by $ord(a)$. If such an integer i cannot be found, then the element a is called an element of infinite order.

Cyclic Groups: If a group has a cyclic view, the group is called cyclic group. In general, a group $\mathcal{G}$ is a cyclic group if all of the elements of group $\mathcal{G}$ can be generated using the power of an element a. The element a called a **generator** of $\mathcal{G}$ and denoted by $\mathcal{G} = \langle a \rangle$.

The properties of groups namely closure under addition, associativity of addition, existence of identity, and existence of inverse are defined as follows.

Closure: For all elements $a,\ b$ in group $\mathcal{G}$, the result of of the operation, $a \circ b$, is a also an element of group $\mathcal{G}$.

Associativity: For all elements $a,\ b,\ c$ in group $\mathcal{G}$, then the result of $a \circ (b \circ c)$ is identical to $(a \circ b) \circ c$.

Existence of Identity: There exists an element e in group $\mathcal{G}$, such that $a \circ e = e \circ a = a$ for any element a in group $\mathcal{G}$.

Existence of Inverse: For any element a in $\mathcal{G}$, there exists an element a^{-1}, such that $a \circ a^{-1} = e$ and $a^{-1} \circ a = e$.

Conditions of each property are shown in Table 2.1.

Property	Condition
Closure	$\forall a, b \in \mathcal{G} : a \circ b \in \mathcal{G}$
Associativity	$\forall a, b, c \in \mathcal{G} : a \circ (b \circ c) = (a \circ b) \circ c$
Existence of Identity	$\exists e \in \mathcal{G}, \forall a \in \mathcal{G} : a \circ e = e \circ a = a$
Existence of Inverse	$\forall a \in \mathcal{G}, \exists a^{-1} \in \mathcal{G} : a \circ a^{-1} = a^{-1} \circ a = e$

Table 2.1: Properties of Groups

2.1.2 Abelian Groups

Abelian groups is a set of elements together with a binary operation $\circ$ satisfying four properties such as closure, associativity, identity, inverses and commutativity. In comparison to group, the properties in abelian groups such as closure, associativity, identity and inverses are essentially the same as in group with the exception of

commutativity property. The Table 2.2 is showing the properties of abelian groups with different conditions.

Closure: For all elements $a,\ b$ in abelian group $\mathcal{G}$, the result of of the operation, $a \circ b$, is also an element of group $\mathcal{G}$.

Associativity: For all elements $a,\ b,\ c$ in abelian group $\mathcal{G}$, then the result of $a \circ (b \circ c)$ is identical to $(a \circ b) \circ c$.

Existence of Identity: There exists an element e in abelian group $\mathcal{G}$, such that $a \ \circ e = e \circ a = a$ for any element a in abelian group $\mathcal{G}$.

Existence of Inverse: For any element a in abelian group $\mathcal{G}$, there exists an element a^{-1}, such that $a \circ a^{-1} = e$ and $a^{-1} \circ a = e$.

Commutativity: For all elements a, b in abelian group $\mathcal{G}$, such that the result of $a \circ b$ is same as the result of $b \circ a$.

2.1.3 Rings

Ring is a type of algebraic structures in which a set of elements together with two binary operations that satisfies following properties described in Table 2.3.

Property	Condition
Closure	$\forall a, b \in \mathcal{G} : a \circ b \in \mathcal{G}$
Associativity	$\forall a, b, c \in \mathcal{G} : a \circ (b \circ c) = (a \circ b) \circ c$
Existence of Identity	$\exists e \in \mathcal{G}, \forall a \in \mathcal{G} : a \ \circ e = e \circ a = a$
Existence of Inverse	$\forall a \in \mathcal{G}, \exists a^{-1} \in \mathcal{G} : a \circ a^{-1} = a^{-1} \circ a = e$
Commutativity	$\forall a, b \in \mathcal{G}, a \circ b = b \circ a$

Table 2.2: Properties of Abelian Groups

Additive Associativity: For all elements $a,\ b,\ c$ in ring $\mathcal{R}$, the result of of the operation, $a + (b + c)$, is equal to the result of $(a + b) + c$.

Additive Commutativity: For all elements $a,\ b$ in ring $\mathcal{R}$, the outcome of $a + b$ is identical to the outcome of $b + a$.

Additive Identity: There exists an element 0 in ring $\mathcal{R}$, such that $0+a = a+0 = a$ for any element a in $\mathcal{R}$.

Additive Inverse: For any element a in ring $\mathcal{R}$, there exists an element a^{-1}, such that $a + a^{-1} = 0$ and $a^{-1} + a = 0$.

Distributivity: For all elements $a,\ b,\ c$ in ring $\mathcal{R}$, the outcome of $a * (b + c)$ is same as $(a * b) + (a * c)$.

Multiplicative Associativity: For all elements $a,\ b,\ c$ in ring $\mathcal{R}$, the result of of the operation, $a * (b * c)$, is equal to the result of $(a * b) * c$.

Multiplicative Commutativity: For all elements $a,\ b$ in ring $\mathcal{R}$, the outcome of $a * b$ is identical to the outcome of $b * a$.

Multiplicative Identity: There exists an element 1 in ring $\mathcal{R}$, such that $1 * a = a * 1 = a$ for any element a in $\mathcal{R}$.

Multiplicative Inverse: For any element a in ring $\mathcal{R}$, there exists an element a^{-1}, such that $a * a^{-1} = 1$ and $a^{-1} * a = 1$.

2.1.4 Fields

Fields: If the non-zero elements of a ring forms a group under multiplicational, then the ring is named a field. In field, there are additional properties compare with ring and group which are defined in Table 2.4. Fields maintain several properties such as closure under addition, associativity of addition, additive identity, additive inverse, commutativity of addition, closure under multiplication, associativity of multiplication, distributive laws, commutativity of multiplication, multiplicative identity,

Property	Condition
Additive Associativity	$\forall a, b, c \in \mathcal{R} : a + (b + c) = (a + b) + c$
Additive Commutativity	$\forall a, b \in \mathcal{R} : a + b = b + a$
Additive Identity	$\exists 0 \in \mathcal{R} : \forall a \in R : 0 + a = a + 0 = a$
Additive Inverse	$\forall a \in \mathcal{R},\ \exists - a \in \mathcal{R} : a + (-a) = (-a) + a = 0$
Distributivity	$\forall a, b, c \in \mathcal{R}, a * (b + c) = (a * b) + (a * c)$
Multiplicative Associativity	$\forall a, b, c \in \mathcal{R} : a * (b * c) = (a * b) * c$
Multiplicative Commutativity	$\forall a, b \in \mathcal{R} : a * b = b * a$
Multiplicative Identity	$\exists 1 \in \mathcal{R} : \forall a \in \mathcal{R}, a \neq 0 : 1 * a = a * 1 = a$
Multiplicative Inverse	$\forall a \in \mathcal{R}, a \neq 0,\ \exists a^{-1} \in \mathcal{R} : a * a^{-1} = a^{-1} * a = 1$

Table 2.3: Properties of Rings

multiplicative inverse and no zero divisors.

Closure under Addition: For all elements $a,\ b$ in field $\mathcal{F}$, the result of of the operation, $a + b$, is a also an element of field $\mathcal{F}$.

Associativity of Addition: For all elements $a,\ b,\ c$ in field $\mathcal{F}$, the result of of the operation, $a + (b + c)$, is equal to the result of $(a + b) + c$.

Additive Identity: There exists an element 0 in field $\mathcal{F}$, such that $0+a = a+0 = a$ for any element a in $\mathcal{F}$.

Additive Inverse: For any element a in field $\mathcal{F}$, there exists an element a^{-1}, such that $a + a^{-1} = 0$ and $a^{-1} + a = 0$.

Commutativity of Addition: For all elements $a,\ b$ in field $\mathcal{F}$, the outcome of $a + b$ is identical to the outcome of $b + a$.

Closure under Multiplication: For all elements $a,\ b$ in field $\mathcal{F}$, the result of of the operation, $a * b$, is a also an element of field $\mathcal{F}$.

Associativity of Multiplication: For all elements $a,\ b,\ c$ in field $\mathcal{F}$, the result of of the operation, $a * (b * c)$, is equal to the result of $(a * b) * c$.

Distributive Laws: For all elements $a,\ b,\ c$ in field $\mathcal{F}$, the outcome of $a * (b + c)$ is same as $(a * b) + (a * c)$.

Commutativity of Multiplication: For all elements $a,\ b$ in field $\mathcal{F}$, the outcome of $a * b$ is identical to the outcome of $b * a$.

Multiplicative Identity: There exists an element 1 in field $\mathcal{F}$, such that $1 * a = a * 1 = a$ for any element a in $\mathcal{F}$.

Multiplicative Inverse: For any element a in field $\mathcal{F}$, there exists an element

a^{-1}, such that $a * a^{-1} = 1$ and $a^{-1} * a = 1$.

No Zero Divisors: If any elements $a,\ b$ are in field $\mathcal{F}$ and $a * b = 0$, then either $a = 0$ or $b = 0$.

2.1.5 Finite Fields

Finite fields is a special type of fields in which the number of elements is finite. In finite fields, the number of elements is the order of the finite field which is always a prime or a power of prime number. For instance, A finite field consists of a set of elements from 0 to $p - 1$ in which p is a prime number. It is denoted by $GF(p^n)$ where n can be any positive integer. In finite field $GF(p)$, there are two arithmetic operations named addition and multiplication. As to addition, if a and b are two elements of finite field $GF(p)$, the additional result of $a + b$ is the remainder of the division of $a+b$ by p. In terms of multiplication, if a and b are two elements of finite field $GF(p)$, the Multiplicative result of $a * b$ is the remainder of the division of $a * b$ by p. This process is called modulo operation.

2.1.6 Random Oracle and Standard Model

In 1993, Mihir Bellare and Phillip Rogaway has proposed that the random oracle model can provide a bridge between cryptographic theory and cryptographic practice [9]. The random oracle model represents a perfect hash function or idealized hash function. In the random oracle, it can be assumed that hash functions are placed

by a publickly accessible random function. In this way, adversary must query the random oracle since he/she cannot compute the result of the hash function. The security proofs are more difficult to achieve in standard model in comparison with random oracle model. In standard model, the adversary is limited by the amount of time and computational ability.

2.2 Number Theoretic Problems

Confidentiality, integrity, authentication and non-repudiation play an important role in the security of network communication. There are two types of attacks in communication where one is passive attacks and the other is active attacks. In active attacks, a malicious attacker attempts to modify the information among the communicators. As to passive attacks, a malicious attacker attempts to read or listen to the source information without modifying the information.In order to prove their security, the security of many cryptosystems relies on the intractability of solving some hard problems being listed in this section. The concept of hard problem means that there is no algorithm can solve the problem within polynomial time. With computational hardness assumptions, cryptographic systems can be secure assuming that any attackers have limited computational capabilities. Summary of different hardness assumptions are listed as follows.

2.2.1 Discrete Logarithm

Discrete Logarithm Problem (DL Problem)

INPUT: In finite field $\mathcal{F}_q$, there exists a generator element g, and an element h in $\mathcal{F}_q$.

OUTPUT: Find a unique integer a which is smaller than q, such that the following equation holds. $h = g^a$

Discrete Logarithm Assumption (DL Assumption)

Given two elements $g,\ h$ in finite field, there is no DL problem solver with PPT algorithm to compute the value a defined in DL problem. The notion of PPT algorithm is a type of algorithms in which the output is a random choice during the execution time and the termination of processing is within polynomial time.

2.2.2 Computational Diffie-Hellman

Computational Diffie-Hellman Problem (CDH Problem)

INPUT: In finite field $\mathcal{F}_q$, there exists a generator element g, and an element $g^a,\ g^b$ for some integers $0 < a, b < q$ in $\mathcal{F}_q$.

OUTPUT: Find a value g^{ab} in $\mathcal{F}_q$.

Computational Diffie-Hellman Assumption (CDH Assumption)

Given three elements $g,\ g^a,\ g^b$ in finite field, there is no CDH problem solver with PPT algorithm to compute the value g^{ab} defined in CDH problem.

2.2.3 Decisional Diffie-Hellman

Decisional Diffie-Hellman Problem (DDH Problem)

INPUT: In abelian group $\mathcal{G}$ of order q, there exists a generator element g, and three element $g^a,\ g^b,\ g^c$ for some integers $0 < a, b, c < q$.

OUTPUT: To decide if the value ab is same as the value c.

Computational Diffie-Hellman Assumption (DDH Assumption)

Given four elements $g,\ g^a,\ g^b,\ g^c$ in abelian group, there is no DDH problem solver with PPT algorithm to decide whether ab is identical to the value c defined in DDH problem.

2.2.4 Bilinear Diffie-Hellman

Bilinear Pairings

Let $(\mathcal{G}_1, +)$ and $(\mathcal{G}_2, +)$ be two cyclic groups with prime order q. The bilinear pairing is denoted as $\hat{e} : \mathcal{G}_1 \times \mathcal{G}_1 \rightarrow \mathcal{G}_2$ which maintains following properties.

Bilinearity: For all elements $P,\ Q,\ R$ in $\mathcal{G}_1$, the equation of $\hat{e}(P + Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$ and $\hat{e}(P, Q + R) = \hat{e}(P, Q)\hat{e}(P, R)$ always hold.

Non-degeneracy: There exists elements $P,\ Q$ in $\mathcal{G}_1$, such that the result of $\hat{e}(P, Q)$ is not equal to 1.

Computability: The operation $\hat{e}(P, Q)$ can be computed efficiently.

Bilinear Diffie-Hellman Problem (BDH Problem)

INPUT: In cyclic group $\mathcal{G}_1,\ \mathcal{G}_2$, there exists a generator element P, bilinear pairing e and three elements $rP,\ sP,\ tP$ for some integer $r,\ s,\ t$ which are smaller than q.

OUTPUT: Find the result of $\hat{e}(P, P)^{rst}$

Bilinear Diffie-Hellman Assumption (BDH Assumption)

Given $P,\ rP,\ sP,\ tP$ in cyclic group $\mathcal{G}_1$ with prime order q, there is no BDH problem solver with PPT algorithm to compute the result of $\hat{e}(P,P)^{rst}$ defined in BDH problem.

2.2.5 Decision Bilinear Diffie-Hellman

Decision Bilinear Diffie-Hellman Problem (DBDH Problem)

INPUT: In cyclic group $\mathcal{G}_1,\ \mathcal{G}_2$, there exists a generator element P in $\mathcal{G}_1$, an element v from $\mathcal{G}_2$, bilinear pairing $\hat{e}$ and three elements $rP,\ sP,\ tP$ for some integer $r,\ s,\ t$ which are smaller than q.

OUTPUT: Decide if v is equal to the value of $\hat{e}(P,P)^{rst}$.

Bilinear Diffie-Hellman Assumption (BDH Assumption)

Given $P,\ rP,\ sP,\ tP$ in cyclic group $\mathcal{G}_1$ with prime order q, and an element v from $\mathcal{G}_2$, there is no BDH problem solver with PPT algorithm to decide if the result of $\hat{e}(P,P)^{rst}$ is equal to the value v defined in DBDH problem.

2.2.6 Strong RSA

Strong RSA Problem

INPUT: There is $N = pq$ with p and q are prime numbers and an integer C which is smaller than N.

OUTPUT: Find a unique integer m and e which is satisfying $m^e \equiv C$. As to e, it should satisfy $gcd(e, (p-1)(q-1)) = 1$.

Strong RSA Assumption

Given a public exponent e which is bigger than 3, ciphertext C, there is no Strong RSA solver with PPT algorithm to find any pair of $(M,\ e)$ such that $C \equiv M^e$.

Property	Condition
Additive Closure	$\forall a, b \in \mathcal{F} : a + b \in \mathcal{F}$
Additive Associativity	$\forall a, b, c \in \mathcal{F} : a + (b + c) = (a + b) + c$
Existence of Identity	$\exists 0 \in \mathcal{F}, \forall a \in \mathcal{F} : a + 0 = 0 + a = a$
Existence of Inverse	$\forall a \in \mathcal{F}, \exists a^{-1} \in \mathcal{F} : a + a^{-1} = a^{-1} + a = 0$
Additive Commutativity	$\forall a, b \in \mathcal{F}, a + b = b + a$
Multiplicative Closure	$\forall a, b \in \mathcal{F} : a * b \in \mathcal{F}$
Multiplicative Associativity	$\forall a, b, c \in \mathcal{F} : a * (b * c) = (a * b) * c$
Distributivity	$\forall a, b, c \in \mathcal{F}, a * (b + c) = (a * b) + (a * c)$
Multiplicative Commutativity	$\forall a, b \in \mathcal{F} : a * b = b * a$
Multiplicative Identity	$\exists 1 \in \mathcal{F} : \forall a \in \mathcal{F}, a \neq 0 : 1 * a = a * 1 = a$
Multiplicative Inverse	$\forall a \in \mathcal{F}, a \neq 0,\ \exists a^{-1} \in \mathcal{F} : a * a^{-1} = a^{-1} * a = 1$
No Zero Divisors	$If\ a, b \in \mathcal{F}, a * b = 0 : either\ a = 0\ or\ b = 0$

Table 2.4: Properties of Fields

Chapter 3

Efficient Escrow-Free Identity-Based Signature (EF-IBS)

3.1 Chapter Overview

As discussed in the introduction, digital signatures with different properties in different setting have been proposed by many researchers. In terms of different settings in digital signatures, there are public key infrastructure, identity-based cryptography, certificate-based cryptography and certificateless public key cryptography. As to different properties of digital signatures, there are verifiable encrypted signature, directed signature, blind signature, undeniable signature, designated verifier signature, aggregate signature, group signature, ring signature, threshold signature, multi-designated verifier signature, etc. In this chapter, our focus is on the digital signature with identity-based cryptography in standard signature.

3.2 Related Work

In order to practically deploy the traditional public key infrastructure (PKI), Shamir [71] proposed the notion of identity-based (ID-based) cryptosystem in which the necessity of verifying the validity of the certificates can be eliminated. In an ID-based cryptosystem, the public key of a user presenting the user's unique identity can be any arbitrary string, such as an email address.

In ID-based cryptosystem an entity named the private key generator (PKG) is responsible of the computation of the user's private keys from a master secret. This property avoids using certificates and associates an implicit public key (user identity) to every user within the system. A sender only needs to know the recipient's identity when he/she sends an encrypted message to the receiver. Hence, in ID-based cryptosystem the complicated and costly certificate verification for the authentication can be avoided. In terms of signature, verifiers only need to take the identity together with the associated message and signature pair as input to check the validity of the signature. (On the contrary, the traditional PKI requires an extra certification verification algorithm which equals to the time required in processing two signatures verification.) Identity-based signature is an effective method to minimize the time for the overall signature verification and the size of the signature.

However, the PKG must be trustworthy because the PKG generates and holds the secret key for all users. Nonetheless, this may not be realistic in practice. The reason is that a malicious PKG can sell users' keys or pretend any user to sign messages or decrypt ciphertexts without being confronted in a court of law. This

problem is regarded as the *key escrow problem* which is a fundamental problem in the ID-based cryptosystem. To address this problem, there exists a main stumbling block applying the ID-based cryptosystem, especially where building a complete trust on the PKG is impractical.

Several researchers proposed other crytposystems to solve the key escrow problem. They proposed a solution combinationing identity-based cryptosystem and public-key infrastructure. Certificateless cryptosystem [1], certificate-based cryptosystem [35], self-certificated cryptosystem [37] and self-generated-certificate public key cryptosystem [59] are some instances of such solutions. In these systems, a user owns both the user public key and secret key, along his identity-based secret key computed by the PKG in which the user secret key helps prevent the key escrow problem. Nonetheless, these systems are no longer purely identity-based because the encryptor or the verifier has to know the user public key together with the user identity. It means the original advantage of identity-based cryptography has been lost.

In terms of *pure* identity-based cryptosystems, Boneh and Franklin [11] proposed a new approach to address the issue of key escrow through using multiple PKGs. Their idea is that a number of PKGs jointly compute the master secret key, such that no individual PKG has the knowledge of it. However, this solution only partially solves the problem because it requires an extra infrastructure and communication overhead. A user needs to run the key extraction protocol with different PKGs which is an inefficient and inconvenient process. Thus, it is unrealistic to maintain multiple PKGs in business infrastructure.

Goyal [39] proposed the concept of accountable authority identity-based encryption (A-IBE) which can decrease the trust in the PKG. The PKG assists users to compute his identity-based secret key without knowing it. If the PKG computes another set of secret key by himself and exposes it to other parties, this key created by the PKG will not be the same as the user's original secret key. Consequently the PKG can be caught when he reveals the secret key because the user can provide his/her original secret key as evidence. However, it is still possible for a malicious PKG to sell a signed message or decrypted ciphertext without being detected. Afterwards, Goyal *et al.* [40] improved the model by suggesting the concept of a black-box A-IBE. In the improved model, a PKG will be caught in a trace algorithm if he sells a decoder box which is able to decrypt ciphertexts.

The previously mentioned solutions only cope with identity-based encryption. With regards to signature, Yuen *et al.* [85] proposed an escrow-free identity-based signature scheme to solve the *key escrow problem*. Their scheme demands an additional entity called judge blaming the malicious PKG. However, it also calls for the PKG to offer some secret data to the judge. If the PKG does not cooperate, the judge cannot produce any evidence to blame the PKG. Therefore, it is unlikely practice in most cases because if the PKG is malicious, he would not work together with judge.

3.3 Syntax

An EF-IBS is possession of five algorithms/protocols, namely, Gen, Ext, Sign, Verify, Blame. The first four algorithms are Identical to a regular IBS with the exception of Ext which is currently an interactive protocol between the PKG and the user. We introduce a public algorithm named Blame which permits the public to determine whether or not the signature is created by the PKG instead of an honest user.

$(\mathsf{param}, \mathsf{msk}) \leftarrow \mathsf{Gen}(1^\lambda)$: On input security parameter λ, this algorithm outputs the public parameter param for the system as well as the master secret key msk for the PKG. We assume param is an implicit input to all the algorithms/protocols below.

$(d_{\mathsf{ID}}) \leftarrow \mathsf{Ext}_{\mathsf{User}}(\mathsf{ID}) \Longleftrightarrow \mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$: This is a pair of interactive algorithms $\mathsf{Ext}_{\mathsf{User}}$ and $\mathsf{Ext}_{\mathsf{PKG}}$ between the user and the PKG. The identity ID of the user is a common input to both parties. The PKG has an additional input msk. Upon successful completion of the protocol, the user obtains a secret key d_{ID} with respect to the identity ID.

$(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$: This algorithm outputs a signature σ on message m with respect to identity ID.

$(1/0) \leftarrow \mathsf{Verify}(\mathsf{ID}, \sigma, m)$: This algorithm verifies a signature σ on message m with respect to identity ID.

$(1/0) \leftarrow \mathsf{Blame}(\mathsf{ID}, \sigma, m, \sigma', m')$: Given a message-signature pair (m, σ) from an honest signer with identity ID, this algorithm outputs 1 if and only if (m', σ')

is created by the PKG and $m \neq m'$.[1]

Correctness. As in IBS, EF-IBS should satisfy correctness. That is, signatures signed by honest signers are verified to be valid.

3.4 Security Requirements

A scheme should be in possession of the standard requirement for signatures as unforgeability along with additional requirements which is defined underneath.

The requirement of *non-slanderability* is to describe the property that the malicious PKG should not be able of creating a signature on behalf of an honest user without being discovered by the algorithm Blame. In addition, the requirement of *non-framability* is to describe the property that a malicious user should not be able to create two signatures in a way that the Blame algorithm's output is 1. The former property defends the user and discourages the PKG from signing on the user's behalf. The latter property defends the PKG so that malicious users cannot blame the honest PKG for being malicious. The security requirements with the following games taken from [85] are formalized as below.

Game Unforgeability

The requirement of existential unforgeability against adaptive chosen ID and message attacks is captured in the following game which is played by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

[1]The condition $m \neq m'$ was added because in case the signature scheme is not strongly unforgeable, it might be feasible to construct another message-signature pair (m, σ^*) given a pair (m, σ).

Setup $\mathcal{C}$ invokes $\mathsf{Gen}(1^k)$ and receives $(\mathsf{param}, \mathsf{msk})$. param is provided to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to make the following queries:

- Extraction Query. $\mathcal{A}$ submits an identity ID and engages $\mathcal{C}$ as a user. $\mathcal{C}$ runs $\mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$.
- Signature Query. $\mathcal{A}$ submits a message m and an identity ID. If $\mathcal{C}$ does not have a secret key d_{ID} with respect to ID, it creates one by invoking $(d_{\mathsf{ID}}) \leftarrow \mathsf{Ext}_{\mathsf{User}}(\mathsf{ID}) \Longleftrightarrow \mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$. Next, it computes $(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$. σ is returned to $\mathcal{A}$.

Output $\mathcal{A}$ submits $(\sigma^*, \mathsf{ID}^*, m^*)$ and wins if and only if

1. $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma^*, m^*)$.
2. A Signature Query with input (m^*, ID^*) has not been submitted by $\mathcal{A}$.
3. An Extraction Query with input ID^* has not been submitted by $\mathcal{A}$.

Definition 3.1 (Unforgeability) *A scheme is unforgeable if there does not exist an adversary wins the above game with non-negligible probability within PPT .*

Game Non-slanderability

The requirement of non-slanderability is formally described in the following game played by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ invokes $\mathsf{Gen}(1^k)$ and receives $(\mathsf{param}, \mathsf{msk})$. Both of these param and msk are sent to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to made the following queries:

- PKG-Extraction Query. $\mathcal{A}$ submits an identity ID and engages $\mathcal{C}$. $\mathcal{C}$ plays the role of a user and runs $\mathsf{Ext}_{\mathsf{User}}(\mathsf{ID})$ to obtain d_{ID}.
- Signature Query. $\mathcal{A}$ submits a message m and an identity ID that has been submitted to PKG-Extraction Query. $\mathcal{C}$ computes $(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$. σ is returned to $\mathcal{A}$.

Output $\mathcal{A}$ submits $(\sigma^*, \mathsf{ID}^*, m^*)$ and wins if and only if

1. $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma^*, m^*)$.
2. $\mathcal{A}$ has submitted a Signature Query with input (m, ID^*) and obtains σ.
3. $0 \leftarrow \mathsf{Blame}(\mathsf{ID}^*, \sigma, m, \sigma^*, m^*)$ and $m \neq m^*$.

Definition 3.2 (Non-slanderability) *A scheme is non-slanderable if there does not exist an adversary wins the above game with non-negligible probability within PPT.*

Game Non-frameability

The requirement of non-frameability is formally presented in the following game played by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ invokes $\mathsf{Gen}(1^k)$ and receives $(\mathsf{param}, \mathsf{msk})$. param is sent to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to made the following queries:

- Extraction Query. $\mathcal{A}$ submits an identity ID and engages $\mathcal{C}$ as a user. $\mathcal{C}$ runs $\mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$.

- Signature Query. $\mathcal{A}$ submits a message m and an identity ID. If $\mathcal{C}$ does not has a secret key d_{ID} with respect to ID, it creates one by invoking $(d_{\mathsf{ID}}) \leftarrow \mathsf{Ext}_{\mathsf{User}}(\mathsf{ID}) \Longleftrightarrow \mathsf{Ext}_{\mathsf{PKG}}(\mathsf{msk}, \mathsf{ID})$. Next, it computes $(\sigma) \leftarrow \mathsf{Sign}(\mathsf{ID}, d_{\mathsf{ID}}, m)$. σ is returned to $\mathcal{A}$.

Output $\mathcal{A}$ submits $(\mathsf{ID}^*, \sigma^*, m^*, \sigma', m')$ and wins if and only if

1. $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma^*, m^*)$, $1 \leftarrow \mathsf{Verify}(\mathsf{ID}^*, \sigma', m')$.
2. $m^* \neq m'$ and a signature query with input $(\cdot, \mathsf{ID}^*)$ has never been submitted by $\mathcal{A}$.
3. $1 \leftarrow \mathsf{Blame}(\mathsf{ID}^*, \sigma', m', \sigma^*, m^*)$.

Definition 3.3 (Non-frameability) *A scheme is non-frameable if there does not exist an adversary wins the above game with non-negligible probability within PPT.*

3.5 The Scheme

Construction of EF-IBS

The construction of EF-IBS is presented in this section. Firstly, the intuition behind the construction and the technique to achieve escrow-freeness are described. Secondly, a high-level description of our scheme is discussed. Finally, the construction followed by the security analysis is presented.

Intuition of EF-IBS

The reason is that there exist many possible d_{ID} with regard to each identity ID, and that the PKG does not have the knowledge of the actual d_{ID} received by an honest user. In addition, the d_{ID} is not re-randomizable. Thus, given one d_{ID}, it is impossible to generate another d_{ID} without knowing the master secret key. Each signature consists of a component, which is denoted by d_{ID}, computed by a one-way function. Therefore, signature created by different d_{ID}'s are different. Concurrently, the signature created by the PKG will be different to a signature created by the honest user considering that the PKG does not know the actual d_{ID}.

However, an honest PKG would only generate one d_{ID} to the user with regard to each identity. Each user will only obtain one d_{ID} and it is computationally impossible to acquire another. Therefore, if there exists two signatures using different d_{ID}, it can be concluded that the PKG is being malicious. In practice, when an honest user witnesses a signature that is not created by himself, he can create another signature to prove that the PKG is dishonest. Anyone can be convinced that it is the PKG that is dishonest if the two signatures are created under different d_{ID}.

The construction starts with the well-known generic construction of IBS from normal digital signatures [7]. Let $\mathcal{S} = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Verify}')$ be a normal digital signature scheme. It can be used to construct an IBS as follows. The PKG generates param and msk, which is merely the public and private key of an instance of $\mathcal{S}$ by invoking

Gen'. In order to create a secret key of identity ID, the PKG invokes Gen' again and generate another pair of public and private keys, say, $y_{\mathsf{ID}}, x_{\mathsf{ID}}$. Subsequently the PKG creates a signature σ_{ID} under his master secret key on "message" $y_{\mathsf{ID}}||\mathsf{ID}$. The value $(\sigma_{\mathsf{ID}}, y_{\mathsf{ID}}, x_{\mathsf{ID}})$ is sent to the user and the user stores it as his secret key.

To produce a signature on message m, the user produces a signature σ_m on m under his x_{ID}. The overall identity-based signature possessing three components is defined as $(\sigma_{\mathsf{ID}}, y_{\mathsf{ID}}, \sigma_m)$. The validity of the IBS can be stated by verifying σ_m and σ_{ID} as valid signatures under "public key" y_{ID} and param respectively.

We come up with the observation that if y_{ID} is generated by the user rather than the PKG, the PKG cannot possess the secret value x_{ID}. Each identity-based signature is computed under an exclusive value y_{ID} and thus the signatures generated by the PKG will be distinct to the signatures generated by the user. This instantiates the idea discussed in the previous section and would yield an escrow-free IBS.

Nevertheless, there is still one unsolved problem mainly because these schemes need to possess two digital signatures and one public key. In the following, we consider specific case based on a particular aggregate signature.

This problem is addressed by using the technology of aggregate signatures. Due to the fact that an IBS consists of three parts, namely, $\sigma_m, \sigma_{\mathsf{ID}}, y_{\mathsf{ID}}$, we made the observation that the first two parts are both digital signatures of the same type.

Therefore, these two components can be aggregated into a single signature if the aggregate signature can be employed in the construction. In this way the final construction can be an efficient scheme as any existing IBS consisting of two elements. In fact, aggregate signatures that supports sequential aggregation are required considering that the PKG creates σ_{ID} followed by the creation of σ_m from the user. The construction of EF-IBS being realized by applying the aggregate signature from Boneh *et al.* [12] is described in the following.

Construction of EF-IBS

We describe our EF-IBS by following the idea described as below. For convenience, we select symmetric pairing setting to present our scheme in spite of expecting it to be adapted to asymmetric pairing setting without any difficulties.

Gen: On input security parameter λ, the PKG generates two groups $\mathcal{G}$, $\mathcal{G}_T$ of prime order p so that p is of λ-bit and that there exists a bilinear pairing $\hat{e} : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}_T$. It selects a generator $g \in_R \mathcal{G}$, a random value $x \in_R \mathbb{Z}_p$ and computes $Y = g^x$. It also chooses a hash function $H : \{0,1\}^* \rightarrow \mathcal{G}$. It publishes $\mathsf{param} := (\hat{e}, \mathcal{G}, \mathcal{G}_T, p, g, Y, H)$. The master secret key msk is (x).

Ext: User whose identity is $\mathsf{ID} \in \{0,1\}^\lambda$ interacts with the PKG in the following protocol. We assume there is an external mechanism assisting the PKG to verify whether the user is the legitimate owner of this identity ID. Moreover, the PKG would stop proceeding if he has seen the value of U received in the

first step.

1. User chooses $u \in_R \mathbb{Z}_p$ at random, computes $U = g^u$ and sends U to the PKG.
2. The PKG computes $V = H(U||\mathsf{ID})^x$ and returns V to the user.
3. The user saves (u, V) as his secret key d_{ID}.
4. The PKG saves the value of U and would reject any further Ext request with value U.

Sign: User with identity ID and secret key $d_{\mathsf{ID}} = (u, V)$ creates a signature on message $m \in \{0,1\}^\lambda$ as follows. User computes $W = H(m)^u$ and $U = g^u$.[2] Next, the user computes $S = VW$. The signature σ_m on message m is parsed as (S, U).

Verify: To verify a signature (S, U) on message m with regard to identity ID, the verifier outputs 1 if and only if the following equation holds (and 0 otherwise):

$$\hat{e}\big(g, S\big) \stackrel{?}{=} \hat{e}\big(U, H(m)\big)\hat{e}\big(Y, H(U||\mathsf{ID})\big).$$

Blame: Given a valid signature (S, U) from an honest user with identity ID on message m and another message-signature pair $\big(m', (S', U')\big)$, this algorithm outputs 1 if and only if $U \neq U'$ and 0 otherwise.

[2] U could be stored as part of the secret key to speed up the signature generation process.

3.6 Security Analysis

With regard to the security of our construction, we have the following theorem.

Theorem 3.1 *Our construction of EF-IBS possesses Unforgeability, Non-slan- derability and Non-frameability under the CDH assumption in the random oracle model.*

We prove the security of our scheme through three lemmas, one for each security requirement.

Lemma 3.2 (Unforgeability) *If there exists a PPT $\mathcal{A}$ that wins Game Unforgeability, we demonstrate that a PPT $\mathcal{S}$ can break the CDH assumption with the help of $\mathcal{A}$.*

Proof:

Let $\hat{e} : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}_T$ be a bilinear map such that $\mathcal{G} = \langle g \rangle$ of prime order p where p is of λ-bit. We show how the CDH assumption can be broken by a PPT $\mathcal{S}$. $\mathcal{S}$ is given g^a, g^b and its goal is to outcome the value g^{ab}. Let q_1, q_2 be the number of hash queries made by $\mathcal{A}$ to the hash oracle H with input length λ and 2λ respectively. Let q_3 be the number of distinct identities which appear in the signature query.

Setup $\mathcal{S}$ chooses $\alpha \in_R \mathbb{Z}_p$ at random and computes $Y = (g^a)^\alpha$. The public key is set as $\mathsf{param} = (\hat{e}, \mathcal{G}, \mathcal{G}_T, p, g, y, H)$, where H is treated as a random oracle. Note that $\mathcal{S}$ is not ware of the master secret key, which has a value of $a\alpha$. $\mathcal{S}$ randomly selects three indexes $\hat{i} \in \{1, \ldots, q_1\}$, $\hat{j} \in \{1, \ldots, q_2\}$ and $\hat{\ell} \in \{1, \ldots q_3\}$.

Query The way $\mathcal{S}$ responses the various queries made by $\mathcal{A}$ is shown in the following.

- Hash Query. For the i-th query with input $m_i \in \{0,1\}^\lambda$, $\mathcal{S}$ randomly chooses $r_i \in_R \mathbb{Z}_p$. If $i \neq \hat{i}$, return g^{r_i}. Otherwise, return $(g^b)^{r_i}$. In the same way, for the j-th query with input $I_j \in \{0,1\}^{2\lambda}$, $\mathcal{S}$ randomly chooses $s_j \in_R \mathbb{Z}_p$. If $j \neq \hat{j}$, return g^{s_j}. Otherwise, return $(g^b)^{s_j}$.
- Extraction Query. $\mathcal{A}$ sends an identity ID and a value U. If $U||\mathsf{ID} = I_{\hat{j}}$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ responses $(g^a)^{\alpha s_j}$ where $I_j = U||\mathsf{ID}$.
- Signature Query. As to the ℓ-th query, if $\ell = \hat{\ell}$, denote $\mathcal{A}$'s input by m and ID. $\mathcal{S}$ sets $U = g^{a\beta}$ with a random value $\beta \in_R \mathbb{Z}_p$ and makes a hash query such that $H(U||\mathsf{ID}) = g^{s_j}$. Locate the index i such that $m = m_i$ (one of the input to the hash query). If $m = m_{\hat{i}}$, $\mathcal{S}$ aborts. Otherwise, it returns $(g^a)^{\beta r_i}(g^a)^{s_j \alpha}$. In contrast, if $\ell \neq \hat{\ell}$, (again, let m and ID be the input of $\mathcal{A}$), $\mathcal{S}$ randomly chooses a value u, computes $U = g^u$ and makes one hash query such that $H(U||\mathsf{ID}) = g^{s_j}$. It returns the signature as (S, U) such that $S = H(m)^u (g^a)^{\alpha s_j}$.

Output $\mathcal{A}$ sends $(S^*, U^*, \mathsf{ID}^*, m^*)$ in a way that

1. $\hat{e}(g, S^*) = \hat{e}(U^*, H(m^*))\hat{e}(Y, H(U^*||\mathsf{ID}^*))$.
2. A Signature Query with input (m^*, ID^*) has not been submitted by $\mathcal{A}$.
3. An Extraction Query with input ID^* has not been submitted by $\mathcal{A}$.

With probability $\frac{(q_1-1)(q_2-q)}{q_1 q_2}$, $\mathcal{S}$ does not abort and with non-negligible probability, one of the following is true:

1. $H(U^*||\mathsf{ID}^*) = (g^b)^{s_j}$ and $H(m^*) = g^{r_i}$; or

2. $U^* = U' = g^{a\beta}$ and $H(m^*) = (g^b)^{r_{\hat{i}}}$ and $H(U^*||\mathsf{ID}^*) = (g^b)^{s_{\hat{j}}}$; or

3. $U^* = U' = g^{a\beta}$ and $H(m^*) = (g^b)^{r_{\hat{i}}}$ and $H(U^*||\mathsf{ID}^*) = g^{s_j}$.

In case (1), $\mathcal{S}$ computes g^{ab} as $(S^*(U^*)^{-r_i})^{\frac{1}{\alpha s_{\hat{j}}}}$. In case (2), $\mathcal{S}$ computes g^{ab} as $(S^*)^{\frac{1}{\beta r_i + \alpha s_j}}$. In case (3), $\mathcal{S}$ computes g^{ab} as $(S^*(g^a)^{-\alpha s_j})^{\frac{1}{\beta r_i}}$.

Case (1) occurs with probability $\frac{q_1 - 1}{q_1 q_2}$, case (2) occurs with probability $\frac{1}{q_1 q_2}$ and case (3) occurs with probability $\frac{q_2 - 1}{q_1 q_2}$. □

Lemma 3.3 (Non-slanderability) *If there exists a PPT $\mathcal{A}$ that wins Game Non-slanderability, we demonstrate that a PPT $\mathcal{S}$ can break the CDH assumption with the help of $\mathcal{A}$.*

Proof:

Let $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{G}_T$ be a bilinear map such that $\mathcal{G} = \langle g \rangle$ of prime order p where p is of λ-bit. We show how to construct a PPT $\mathcal{S}$ that breaks the CDH assumption. $\mathcal{S}$ is given g^a, g^b and its goal is to outcome the value g^{ab}. Let q_1, q_2 be the number of hash queries made by $\mathcal{A}$ to the hash oracle H with input length λ and 2λ respectively. Let q_3 be the number of PKG-Extraction Queries.

Setup $\mathcal{S}$ chooses $x \in_R \mathbb{Z}_p$ at random and computes $Y = g^x$. The public key is set as $\mathsf{param} = (\hat{e}, \mathcal{G}, \mathcal{G}_T, p, g, y, H)$, where H is treated as a random oracle. The master secret key msk is x. Both of the value of param and x are given to $\mathcal{A}$. $\mathcal{S}$ randomly chooses two indexes $\hat{i} \in \{1, \ldots, q_1\}$ and $\hat{j} \in \{1, \ldots, q_3\}$.

Query The way $\mathcal{S}$ responses the various queries made by $\mathcal{A}$ is shown in the following.

- Hash Query. As to the i-th query with input $m_i \in \{0,1\}^\lambda$, $\mathcal{S}$ chooses $r_i \in_R \mathbb{Z}_p$ at random. If $i \neq \hat{i}$, return g^{r_i}. Otherwise, return $(g^b)^{r_i}$. Regarding to any H query with input of length 2λ-bit, $\mathcal{S}$ returns with a random element of $\mathcal{G}$.

- PKG-Extraction Query. $\mathcal{A}$ sends an identity ID. With regard to the j-th query such that $j \neq \hat{j}$, $\mathcal{S}$ randomly chooses a value $u_j \in_R \mathbb{Z}_p$, computes $U_j = g^{u_j}$, submits it to $\mathcal{A}$ and obtains V. For $j = \hat{j}$, $\mathcal{S}$ submits $U_j = g^a$ to $\mathcal{A}$.

- Signature Query. As to signature query which involves ID having been submitted to j-th PKG-Extraction Query such that $j \neq \hat{j}$, $\mathcal{S}$ is aware of the secret key associated with ID and thus responses the query correctly. If ID is involved with the $\hat{j}$-th PKG-Extraction query, $\mathcal{S}$ aborts if the message m to be signed is the input to the $\hat{i}$-th query. Otherwise, it can compute S as VW where $W = (g^b)^{j_i}$.

Output With probability $\frac{q_1-1}{q_1}$, $\mathcal{S}$ does not abort and $\mathcal{A}$ submits (S^*, U^*, ID^*, m^*) such that there exists a Signature Query of which the return values is (S, U, ID, m) and that $U^* = U$. Moreover, with probability at least $1/q_1q_3$, $U = U_{\hat{j}}$ and $m^* = m_{\hat{i}}$ and $\mathcal{S}$ computes g^{ab} as $(S^*H(U^*||\mathsf{ID})^{-x})^{\frac{1}{r_{\hat{i}}}}$. □

Lemma 3.4 (Non-frameability) *If there exists a PPT $\mathcal{A}$ that wins Game Non-frameability, we demonstrate that a PPT $\mathcal{S}$ can break the CDH assumption with the help of $\mathcal{A}$.*

Proof:

Let $\hat{e} : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}_T$ be a bilinear map such that $\mathcal{G} = \langle g \rangle$ of prime order p where p is of λ-bit. We show how to construct a PPT $\mathcal{S}$ which can break the CDH assumption. $\mathcal{S}$ is given g^a, g^b and its goal is to outcome the value g^{ab}. Let q_1, q_2 be the number of hash queries made by $\mathcal{A}$ to the hash oracle H with input length λ and 2λ respectively.

Setup $\mathcal{S}$ chooses $\alpha \in_R \mathbb{Z}_p$ at random and computes $Y = (g^a)^\alpha$. The public key is set as $\mathsf{param} = (\hat{e}, \mathcal{G}, \mathcal{G}_T, p, g, y, H)$, where H is treated as a random oracle. Note that $\mathcal{S}$ is not aware of the master secret key, which has a value of $a\alpha$. $\mathcal{S}$ chooses an index $\hat{j} \in \{1, \ldots, q_2\}$ at random.

Query We show how $\mathcal{S}$ responses the various queries made by $\mathcal{A}$.

- Hash Query. With regard to the i-th query with input length λ, $\mathcal{S}$ chooses a value r_i at random and returns g^{r_i}. As regards the j-th query with input $I_j \in \{0,1\}^{2\lambda}$, $\mathcal{S}$ chooses $s_j \in_R \mathbb{Z}_p$ at random. If $j \neq \hat{j}$, return g^{s_j}. Otherwise, return $(g^b)^{s_j}$.
- Extraction Query. $\mathcal{A}$ sends an identity ID and a value U. If $U||\mathsf{ID} = I_{\hat{j}}$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ responses $(g^a)^{\alpha s_j}$ where $I_j = U||\mathsf{ID}$.
- Signature Query. Regarding each query, $\mathcal{A}$ sends a message m and an identity ID. $\mathcal{S}$ aborts if $U||\mathsf{ID} = I_{\hat{j}}$ for some U. Otherwise, $\mathcal{S}$ randomly computes $U = g^u$ (if it has not done so before) and obtains s_j such that $H(U||\mathsf{ID}) = g^{s_j}$. It then computes S based on $(g^a)^{\alpha s_j} H(m)^u$. Note that $\mathcal{A}$ is not allowed to query signatures on the identity it is going to attack.

Output $\mathcal{S}$ does not abort with probability $\frac{q_2-1}{q_2}$, and $\mathcal{A}$ sends (S^*, U^*, ID^*, m^*) and (S', U', ID^*, m') such that $U^* \neq U'$ and $m \neq m'$. The equation of $U^*||\mathsf{ID}^* = I_{\hat{j}}$ or $U'||\mathsf{ID}^* = I_{\hat{j}}$ holds with the probability at least $2/q_2$. Assuming $U^*||\mathsf{ID}^* = I_{\hat{j}}$ without losing generality, $\mathcal{S}$ locates i such that $H(m^*) = g^{r_i}$ and computes $g^{ab} = (S^* U^{*-r_i})^{\frac{1}{\alpha s_j}}$. □

3.7 Implementation

The efficiency of the EF-IBS algorithm in terms of time complexities is presented in this section. Empirical Analysis as well as experimental results are described as follows.

3.7.1 Empirical Analysis

The complexity of the five algorithms/protocols does not depend on the number of users in the system. Our system is highly efficient in Gen, Ext, Sign and Blame. There is only one modular exponentiation of a cyclic group in the former three algorithms/protocols. As to Verify, it is the slowest operation which involves 3 pairing operations. In terms of space complexity, the secret key d_{ID}, which consists of one element in $\mathcal{G}$ and one element in $\mathbb{Z}_p$ is required to store by users. In considering efficiency, an extra group element (the element U) is also required to store by users . In addition, the size of signature becomes short and consists of 2 elements per signature.

The number of pairing operations and exponentiations in various groups needed

Operation	Gen	Ext(User)	Ext(PKG)	Sign	Verify	Blame
EXP in $\mathcal{G}$	1	1	1	1	0	0
Pairing	0	0	0	0	3	0

Table 3.1: Empirical Complexities

in each protocols is shown in Table 3.1.

3.7.2 Experimental Results

The performance of EF-IBS is estimated by measuring the time cost of various basic operations according to the pairing-based library (version 0.5.12)[3] and the performance of the system is benchmarked. The experimental outcome was produced through 10 test runs. The test machine was an Acer 3820TzG with 2.8GHz Intel I 5-520 CPU and 4GB of Ram running Ubuntu 10.01 with kernel 2.6.338/8. Two settings are considered, namely, symmetric pairing and asymmetric pairing. Recorded that if symmetric pairing (Type A pairing) is to be employed, group elements are of size 512 bits. Moreover, the time required in hashing to point is slow and the cost it takes is comparable to that of a pairing operation. However, if asymmetric pairing (type D pairing, using parameter d62003-159-158.param) $\mathcal{G}_1 \times \mathcal{G}_2 \rightarrow \mathcal{G}_T$ is to be employed, short representation of group elements is only possible in $\mathcal{G}_1$. Group order p in this setting is of 158 bit and elements in $\mathcal{G}_1$ are of size 159 bit only. Nonetheless, the time required in operations of group $\mathcal{G}_2$ are slow. The implementation result is shown in Table 3.2.

The benchmark should give a precise estimation on the actual performance of

[3]http://crypto.stanford.edu/pbc/

	Symmetric Pairing (Type A)	Asymmetric Pairing(Type D)
Exponentiations	5.4 ms	2.16 ($\mathcal{G}_1$) / 10.1 ($\mathcal{G}_2$) ms
Pairing	6.25 ms	9.4 ms
Hash to point	11 ms	0.2 ($\mathcal{G}_1$) / 46.02 ($\mathcal{G}_2$) ms
Size of group elements	64 bytes	20 ($\mathcal{G}_1$) / 60 ($\mathcal{G}_2$) bytes

Table 3.2: Experimental Results

the construction. The purpose of our experiment is for feasibility of our proposal, therefore one parameter type is used to analyze the time required in different operations. In addition,the specific parameter chosen here is one of the most popular choices.

3.7.3 Comparison

Various IBS schemes are compared with our proposed scheme, in terms of efficiency and security features in Table. Compared with the construction proposed by Yuen *et al.* [85], our proposed scheme is very practical, since it allows the public to determine whether the PKG has put the signature on behalf of the honest user. As to the construction proposed by Yuen using BLS, the signature size is three and it is escrow free in random oracle model. The time required in sign algorithm takes one exponentiation operation time while the time required in verify algorithm takes four pairing operation time. Regarding to the scheme using BB [85], the signature requires six group elements and it is escrow free under standard oracle model. The time required in sign algorithm is identical to the time required in an exponentiation. In addition, the time required in verify algorithm is four pairing

and two exponentiations. In terms of Hess [42], the signature contains two group elements and under random oracle model, key escrow problem still exists. Whereas the time required in sign is two exponentiations, the time required in verify is two pairings and two exponentiations. With regards to Waters [81, 62], the size of signature is three and key escrow problem exists in the standard model. The time required in sign algorithm is the time required in three exponentiations and the time required in verify algorithm is three pairings. All these comparison result is summarized in Table 3.3.

Scheme	Signature Size[a]	Escrow Free	Security Model	Sign Computation[b]	Verify Computation[b]
Yuen *et al.* [85] (using BLS [13])[c]	3	✓	ROM	1 E	4 P
Yuen *et al.* [85] (using BB [10])[c]	6	✓	standard	1 E	4 P + 2 E
Hess [42]	2	×	ROM	2 E	2 P + 2 E
Waters [81, 62]	3	×	standard	3 E	3 P
Our proposed scheme	2	✓	ROM	1 E	3 P

Table 3.3: Comparison of different IBS

[a] The unit for signature size is group element.

[b] When we come across the computation of sign and verify, we use E to represent an exponentiation and P to represent a pairing. Other operations such as hashing and modulus addition are negligible.

[c] Yuen *et al.* [85] provides a generic construction for escrow-free IBS scheme. We divide the comparison into using BLS [13] scheme for random oracle model and using BB [10] scheme for standard model, which are the shortest signature schemes in the literature for the corresponding models.

Chapter 4

(Strong) MDVS Secure Against Rouge Key Attack

4.1 Chapter Overview

It is known that digital signatures with different properties in various settings have been introduced by many researchers. Four settings of digital signatures and a number of different properties of digital signatures are discussed in the introduction. In this chapter, the focus is on the digital signature with multi-designated verifiers signature in public key infrastructure.

4.2 Related Work

The notion of Designated verifier signatures/proof (DVS/DVP) was proposed by Jakobsson, Sako and Impagliazzo [46], and independently by Chaum [20] in 1996. In DVS scheme a signer can convince a designated verifier that the message is endorsed by the signer while the designated verifier cannot transfer this conviction to anyone else. The fundamental principle of DVS is that a signature which can declares

the validity of the statement " The message has been endorsed by the signer" or " The designated verifier's secret key is known by the signer " is non-interactive proof. Whereas being convinced that the signer has endorsed the message, the designated verifier cannot convince other parties because the proof could have been produced by designated verifiers. As discussed in the same paper, Jakobsson *et al.* presented the concept of strong DVS (SDVS) in which the private key of the designated verifier is essential to verify the signature. As mentioned that DVS itself reveals the information that the signature is created by the signer or the designated verifier. However, an external party is aware of the signature produced by the signer if she/he captures the signature before it reaches the designated verifier. This requirement is formalized as privacy of signer's identity in [69]. It is necessary that external party cannot confirm who created the signature without the designated verifier's private key.

The concept of multiple verifiers has been described in [46], and in the rump session of Crypto'03. In addition, Desmedt [30] discussed the notion of multi-designated verifiers signatures (MDVS) as a generalization of DVS which was later formalized in [50]. Subsequently, many researchers has introduced a number of MDVS constructions [46, 50, 61, 70, 26, 52, 55, 27, 79, 18, 77, 78] with different features in various settings. Interested readers may refer to [78] for a survey.

Jakobsson *et al.* has first discussed the problem of rogue key attack in DVS in [46]. In the discussion, the goal of a malicious verifier is to convince an external party that the message has been endorsed by the signer. For instance, malicious verifier can produce his/her public key based on the output of a hash function

using a random number as input. Afterwards, everyone will be convinced that the signatures must have been produced by the signer when the malicious verifier discloses the value of the random number. One of the counter-measures suggested is to require Bob to prove the knowledge of his secret key. On the other hand, one of the counter-measures recommended is to demand the designated verifier to prove the knowledge of the secret key. Another type of rogue key attack specifically on MDVS was proposed by Shim in [73]. In this attack, a malicious verifier creates their own public key as a function of other honest verifiers' public keys in which a signature that passes the verification of other honest verifiers can be created. The recommended counter-measure is to demand the verifier to prove the knowledge of the secret key. More importantly, there is no formal model proposed to capture the attack. It is acknowledged that the two types of rogue key attacks have different properties. The former is against non-transferability whereas the latter is against unforgeability which is the primary focus of the current chapter.

As mentioned, a counter-measure against rogue key attack in MDVS is to demand the adversary to produce a proof-of-knowledge of the secret key. This particularly indicates that all users are required to provide a proof-of-knowledge of the secret key to the certification authority (CA) prior to certifying the corresponding public key. However, this solution requires modification of the current PKI which is costly [8]. Therefore, designing a MDVS scheme which is securely against rogue key attack in the plain model is desired. In respond to this, a partial solution proposed as the first MDVS scheme that is formally proven unforgeable under rogue key attack is presented in this chapter.

It is acknowledged that if the DVS is encrypted under the designated verifier's public key, the resulting scheme would be a strong DVS. However, a subtle issue described in [50] can avoid such generic transformation to be applicable in MDVS. Particularly, the challenge is to guarantee correctness of the resulting scheme considering that it is entirely feasible for a signer to encrypt different values under different designated verifier's public key. In that case, a signature could be considered as valid by some of the designated verifiers only. This issue is addressed by a hybrid encryption using a simple one-way secure encryption and a symmetric encryption. The unforgeability under rogue key attack is then preserved in our generic transformation. Particularly, the following contributions are made.

4.3 Preliminary

4.3.1 Syntax

The definitions and security models of MDVS are adapted from various literatures [50, 52]. A MDVS scheme is composed of four algorithms, namely, Setup, Gen, Sign, Verify with different properties listed as follows.

$\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$: On input a security parameter λ, this algorithm produces the public parameter param for the system. Noteworthy that this algorithm is *optional* if all users could create their key pairs with no coordination. However, all schemes known to us demand the key pairs of users to be generated through some common system parameters. We assume param is an implicit input to

all algorithms mentioned as follows.

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}()$: For a user (who can take the role of a signer or a designated verifier) this algorithm produces a key pair $(\mathsf{pk}, \mathsf{sk})$. Therefore, pk is the matching public key of sk (and vice versa).

$(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$: As input a message m, a secret key of a signer sk_S (with the public key pk_S) and a set of designated verifiers' public keys $\mathcal{V}$, this algorithm produces a signature σ, which is a designated verifier on message m with regard to the public key pk_S.

$\texttt{valid/invalid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_V)$: Given as input a public key pk_S, a message m, a signature σ with a set of designated verifiers' public keys $\mathcal{V}$ and a private key sk_V such that the corresponding public key is $\mathsf{pk}_V \in \mathcal{V}$, this algorithm checks the validity of the signature and produces `valid/invalid`.

A MDVS scheme must possess three properties namely *Correctness*, *Unforgeability* and *Source-Hiding* presented as follows.

Correctness. With regard to any security parameter λ and $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{pk}_S, \mathsf{sk}_S) \leftarrow \mathsf{Gen}()$ and $\mathcal{V} = \{\mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}\}$ so that $(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}) \leftarrow \mathsf{Gen}()$ for $i \in [n]$. For any message m, if $(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$, subsequently $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_{V_i})$ for all $i \in [n]$. Moreover, regarding to any values σ, $\mathcal{V}$, m, pk_S, if there is a private key sk_V such that its corresponding public key $\mathsf{pk}_V \in \mathcal{V}$ and that $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_V)$, then for any private key $\mathsf{sk}_{V'}$, it holds that $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_{V'})$ if the corresponding public key $\mathsf{pk}_{V'} \in \mathcal{V}$.

Unforgeability. The requirement of *Unforgeability* is formally captured by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and subsequently $\mathsf{Gen}()$ to receive $(\mathsf{param}, (\mathsf{pk}_S, \mathsf{sk}_S), \{(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i})\}_{i\in[n]})$. The set $\{\mathsf{pk}_{V_i}\}_{i\in[n]}$ is denoted by $\mathcal{V}$. $\mathcal{C}$ sends $(\mathsf{param}, \mathsf{pk}_S, \mathcal{V})$ to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to ask the following queries:

- Corruption Query. $\mathcal{A}$ sends a public key $\mathsf{pk}_V \in \mathcal{V}$ to $\mathcal{C}$ and obtains sk_V.
- Signature Query. $\mathcal{A}$ sends a message m to $\mathcal{C}$ and obtains $(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$ from $\mathcal{C}$.

Output $\mathcal{A}$ sends (σ^*, m^*) to $\mathcal{C}$ and wins if and only if

1. There is a public key $\mathsf{pk}_{V^*} \in \mathcal{V}$ such that $\mathtt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma^*, \mathcal{V}, m^*, \mathsf{sk}_{V^*})$.
2. A Signature Query with input m^* has not been submitted by $\mathcal{A}$.
3. There is a public key $\mathsf{pk}_V \in \mathcal{V}$ which has not been submitted a Corruption Query as input by $\mathcal{A}$.

Definition 4.1 (Unforgeability) *A MDVS scheme is unforgeable if there does not exist any PPT adversary winning the above game with non-negligible probability.*

As presented in [50], the adversary is not given an oracle for signature verification since any signatures can be verified by him with corrupting some of the verifiers.

Source hiding. That's to say, given a message m and a signature $(\sigma, \mathcal{V})$, it is impossible to identify who from the original signer or the designated verifiers all together created the signature, even if all the secret keys are known. The formal definition is adapted from Definition 3 of [43] for normal DVS into that for MDVS.

Definition 4.2 (Source Hiding) *A MDVS scheme is* source hiding *if there is a PPT simulation algorithm* Sim *that given as input a public key* pk_S*, a set of key pairs* $(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i})_{i\in[n]}$ *and a message* m*, produces a tuple* $(\sigma, \mathcal{V})$ *(such that* $\mathcal{V} = \{\mathsf{pk}_{V_i}\}_{i\in[n]}$*) that is indistinguishale to* $(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$ *(where* sk_S *is the corresponding private key of* pk_S*). In other words, for all PPT algorithmn* $\mathcal{D}$*, for any security parameter* λ*,* $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$*,* $(\mathsf{pk}_S, \mathsf{sk}_S) \leftarrow \mathsf{Gen}()$*,* $\{(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}) \leftarrow \mathsf{Gen}()\}_{i\in[n]}$ *and any message* m*, it holds that:*

$$\left| Pr\left[\begin{array}{r} (\sigma_0, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m) \\ (\sigma_1, \mathcal{V}) \leftarrow \mathsf{Sim}(\mathsf{pk}_S, \{\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}\}_{i\in[n]}, m) \\ b \in_R \{0,1\} \\ b' \leftarrow \mathcal{D}(\sigma_b, \mathsf{pk}_S, \mathsf{sk}_S, \{\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}\}_{i\in[n]}, m) \end{array} : b = b' \right] - 1/2 \right| = \mathit{negl}(\lambda)$$

where $\mathit{negl}(\lambda)$ *is a negligible function in* λ*. A function* $\mathit{negl}(\lambda)$ *is said to be negligible in* λ *if for all polynomial* $q(\cdot)$*, there is a value* k_0 *such that for every* $\lambda > k_0$*,* $\mathit{negl}(\lambda) < 1/q(\lambda)$.

4.3.2 Strong Multi-Designated Verifiers Signatures

It is desirable in many scenarios that, in addition to the signer and the verifier, a third party cannot determine if a signature for the verifier is created by that particular signer or by someone else. This concept appeared in [46] and is formally defined as privacy of signer's identity (PSI) in [50]. This applies to the case of multiple designated verifiers and the property PSI for MDVS is defined in [50].

Privacy of signer's identity. The requirement of *PSI* is formally captured by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and then $\mathsf{Gen}()$ to receive (param, (pk_{S_0}, sk_{S_0}), (pk_{S_1}, sk_{S_1}), $\{(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i})\}_{i\in[n]}$). The set $\{\mathsf{pk}_{V_i}\}_{i\in[n]}$ is denoted by $\mathcal{V}$. (param, pk_{S_0}, pk_{S_1}, $\mathcal{V}$) is given to $\mathcal{A}$.

Query $\mathcal{A}$ is permitted to ask the following queries:

- Verification Query. $\mathcal{A}$ sends $(m, \sigma, \mathcal{V}, \mathsf{pk}_{S_c} : c \in \{0,1\}, V \in \mathcal{V})$ to $\mathcal{C}$ and obtains $\texttt{valid/invalid} \leftarrow \mathsf{Verify}(\mathsf{pk}_{S_c}, \sigma, \mathcal{V} \cup \mathsf{pk}_V, m, \mathsf{sk}_V)$.
- Signature Query. $\mathcal{A}$ sends a message m, a bit b to $\mathcal{C}$ and obtains (σ, $\mathcal{V}$) $\leftarrow \mathsf{Sign}(\mathsf{sk}_{S_b}, \mathcal{V}, m)$.

Challenge At some point $\mathcal{A}$ sends a message m^*. $\mathcal{C}$ tosses a fair coin b and obtains (σ^*, $\mathcal{V}$) $\leftarrow \mathsf{Sign}(\mathsf{sk}_{S_b}, \mathcal{V}, m)$ from $\mathcal{C}$.

Query $\mathcal{A}$ keeps to make verification and signature queries.

Output $\mathcal{A}$ sends a bit b' and wins if and only if $b' = b$.

$\mathcal{A}$'s advantage in the game PSI is determined as the probability of $\mathcal{A}$ winning the game minus 1/2.

Definition 4.3 (Privacy of signer's identity) *A MDVS scheme is said to own privacy of signer's identity if there does not exist any PPT adversary with non-negligible advantage in game PSI.*

A strong MDVS scheme is a MDVS scheme having the privacy of signer's identity.

4.4 The Scheme

4.4.1 Rouge Key Attack in MDVS and its Solution

Firstly, the generic construction of MDVS from discrete logarithm-based ring signatures [50] is reviewed. In the next subsection, we will see the way how a malicious designated verifier could start a rogue key attack to force an honest verifier into accepting a forged signature. Noted that this attack is outside the original security model and does not indicate the scheme is insecure. On the contrary, it is possible to say that a signature that passes the verification of a particular honest designated verifier could have been generated by a real signer or some other malicious verifiers. We will see the proposed fix soon.

4.4.2 Generic Construction of MDVS [50]

The generic construction makes use of ring signatures as building blocks and requires that all the keys are discrete logarithm-based. Readers are referred to [67] for the

formal definition of a ring signature scheme. Roughly speaking, a ring signature is a signature generated by one of the possible signers in a set of signers (often called a ring of signers). The ring of signers are produced in an ad-hoc manner by the actual signer. The formation is spontaneous in which the members can be completely unaware of being conscripted into the ring. In the generic construction of MDVS, ring signatures are required to support a ring size of 2.

- Setup. This is identical to the parameter generation of the ring signature scheme (if any).

- Gen. This is identical to the key generation of the ring signature scheme. The generic construction demands the key of the ring signature to be of the form (g^x, x) where g is included in the parameter, x is the signing key and g^x is the corresponding public key.

- Sign. Let the signer's key pair be (g^{x_S}, x_S) and the set of designated verifiers' key pairs be $\{(g^{x_{V_i}}, x_{V_i})\}$ for $i = 1$ to n. The signer computes $g^{X_V} = \prod_{i\in[n]} g^{x_{V_i}}$. Then, the signer generates a ring signature on message m on the ring $\{g^{x_S}, g^{x_V}\}$ under the secret key x_S. The output is denoted by σ. This value, along with the set $\{g^{x_{V_i}}\}_{i\in[n]}$, is outputted as the multi-designated verifier signature.

- Verify. To verify the signature $(\sigma, \{g^{x_{V_i}}\}_{i\in[n]})$ on message m, a verifier computes $g^{X_V} = \prod_{i\in[n]} g^{x_{V_i}}$. Then it uses the verification algorithm of the ring signature scheme on the ring $\{g^{X_S}, g^{X_V}\}$.

The property of *unforgeability* comes from the fact that to generate a ring signature on the ring $\{g^{x_S}, g^{x_V}\}$, one of x_S or x_V should be known. Since the adversary

is not aware of x_S or x_V[1], forging a signature implies breaking the unforgeability of the underlying ring signature scheme. However, the *source hiding* property comes from the fact that if one knows all secret keys of the verifiers, he/she can construct a PPT Sim by computing $x_V = \sum_{i\in[n]} x_{V_i}$ and using it to generate a ring signature on behalf of the ring $\{g^{x_S}, g^{x_V}\}$. As a result of the anonymity of ring signature, there does not exist any PPT algorithm which can distinguish a signature created by the real signer using x_S or by Sim using x_V.

4.4.3 Rouge Key Attack and Its Defence

Existential Forgery under Rouge Key Attack. Shim has discussed rouge key attack against a concrete scheme in [50] in [73]. The attack extended to the generic construction of [50] is presented here. Assume the goal of an adversary is to persuade an honest designated verifier to accept a forged signature created by himself/herself. Let g^{x_S}, $g^{x_{V'}}$ be the public keys of the targeted signer and designated verifier respectively. To defraud the verifier, the adversary randomly selects a value x_A and crafts a mal-formed public key $K = g^{x_A}/g^{x_{V'}}$. Next, the adversary computes $g^{x_V} = Kg^{x_{V'}} = g^{x_A}$. Due to the adversary possessing the value x_A, he can generate a ring signature on the ring $\{g^{x_S}, g^{x_V}\}$. He outputs the signature, together with the set of designated verifiers as $\{g^{x_{V'}}, K\}$. As a result, the designated verifier would accept a forged signature generated by the adversary instead of the signer. This kind of attack can be denoted as forgery against rogue key attack (RKA).

[1]Since the adversary cannot corrupt all the verifiers, it does not know the value x_V, which is equal to $\sum_{i\in[n]} x_{V_i}$.

A Proposed Fix. The problem comes from the adversary being given extra power to create malformed public key. The fix suggest in [73] is to demand the certification authority to check the validity of the public key before sending a digital certificate to users. As to modelling, this implies the stronger certified key model in which the users are needed to conduct a proof-of-knowledge of his secret key to the CA. As discussed in [8], this requires alteration of the client and CA functioning software. Another solution is introduced that could resist this attack in the plain model according to a technique used in multisignautres [8] and batch verification of digital signatures [6]. Simply speaking, g^{x_V} is defined to be $\prod_{i\in[n]}(g^{x_{V_i}})^{h_i}$, where $h_i = H(g^{x_S}, g^{x_{V_1}}, \ldots, g^{x_{V_n}}, m, i)$ for a hash function H which shall be acted as a random oracle. Note that with this modification, the value x_V can still be computed if all the values x_{V_i} are known. However, if one of the secret keys, say x_{V_i}, is unknown, the value x_V cannot be computed since the probability of "canceling" $g^{x_{V_i}}$ in the computation of g^{x_V} is negligible supposing the values h_i are randomly distributed and are only known after the value of the public keys are chosen.

4.4.4 Formal Security Definition for Unforgeability Under Rogue Key Attack

To formally assert the security of our proposed solution, a security model which intends to capture attack of this kind.[2] is defined. We believe a verification query with the target verifier can be used by the adversary because the adversary might

[2]While rogue key attack on MDVS is discussed in [73], no formal security model has been proposed to capture such an attack.

try to send mal-formed signatures in order to learn information about the target verifier's verification procedure.

Unforgeability Against Rogue Key Attack. The requirement of *UF-RKA* is formally captured by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and then $\mathsf{Gen}()$ to receive $(\mathsf{param}, (\mathsf{pk}_S, \mathsf{sk}_S), (\mathsf{pk}_V, \mathsf{sk}_V)$ and send $(\mathsf{param}, \mathsf{pk}_S, \mathsf{pk}_V)$ to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to make the following queries:

- Verification Query. $\mathcal{A}$ sends a set of public keys $\mathcal{V}$, a signature $(\sigma, \mathcal{V} \cup \mathsf{pk}_V)$, a message m and obtains $\texttt{valid/invalid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V} \cup \mathsf{pk}_V, m, \mathsf{sk}_V)$.
- Signature Query. $\mathcal{A}$ sends a message m, a set of public keys $\mathcal{V}$ and obtains $(\sigma, \mathcal{V}) \leftarrow (\mathsf{sk}_S, \mathcal{V}, m)$. Recorded that $\mathcal{A}$ can send an arbitrary set of verifiers of his choice (even a set without pk_V).

Output $\mathcal{A}$ sends (σ^*, m^*) and a set of public keys $\mathcal{V}^*$ and wins if and only if

1. $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma^*, \mathcal{V}^* \cup \mathsf{pk}_V, m^*, \mathsf{sk}_V)$.
2. A Signature Query with input $(m^*, \mathcal{V}^* \cup \mathsf{pk}_V)$ has not been submitted by $\mathcal{A}$.

Definition 4.4 (UF-RKA) *A MDVS scheme is unforgeable under rogue key attack if there does not exist any adversary wins the above game with non-negligible*

probability in PPT.

We believe UF-RKA for MDVS is a stronger notion compared with the notion *Unforgeability.*

4.4.5 A Concrete Construction

Here, a concrete MDVS scheme is presented from a commonly used two-party ring signature following the generic construction along with proposed fix.

- Setup. Let $\mathcal{G} = \langle g \rangle$ be a cyclic group of prime order p. The outcome of this algorithm is param as $(\mathcal{G}, p, g)$.

- Gen. Select a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$ which will be modelled as a random oracle[3]. Randomly select value $x \in_R \mathbb{Z}_p$, compute g^x. The outcome is pk as (g^x, H) and sk as x.

- Sign. On input the signer's key pair $(\mathsf{pk}_S, \mathsf{sk}_S)$, a set of designated verifier's public keys $\mathcal{V} = \{\mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}\}$ and a message m, parse pk_S as (Y_S, H_S), sk_S as x_S, pk_{V_i} as (Y_{V_i}, H_{V_i}). Compute $Y = \prod_{i \in [n]} Y_{V_i}^{h_i}$ where $h_i = H_{V_i}(\mathsf{pk}_S, \mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}, m)$.

  1. Randomly choose $r, c_2, z_2 \in \mathbb{Z}_p$, compute $T_1 = g^r$, $T_2 = Y^{c_2} g^{z_2}$.

  2. Compute $c = H_S(T_1, T_2, \mathsf{pk}_S, \mathsf{pk}_{V_1}, h_1, \ldots, \mathsf{pk}_{V_n}, h_n, Y, m)$ and $c_1 = c - c_2$.

  3. Compute $z_1 = r - c_1 x_S$.

[3]We abuse the notation and assume a full domain hash. In the following when we write $c = H(X, Y)$ where X and Y may be elements from different domains, we assume a suitable encoding scheme is employed to convert X, Y into a bit-string.

The outcome is the signature as $(c_1, c_2, z_1, z_2, \mathcal{V})$. Recorded that (c_1, c_2, z_1, z_2) is a ring signature on message m with respective to ring $\{Y_S, Y\}$.

- We remark that the above steps constitute a standard signature proof-of-knowledge of 1-out-of-2 discrete logarithms (which can be viewed as a two-party ring signature). This can be described as follows using the Camenisch and Stadler notation [16].

$$\mathrm{SPK}\left\{(\alpha) : Y_S = g^{\alpha} \vee Y = g^{\alpha}\right\}(m)$$

- Verify. To verify the signature $(c_1, c_2, z_1, z_2, \{\mathsf{pk}_{V_i}\}_{i\in[n]})$ on message m, pk_{V_i} can be parsed as (Y_{V_i}, H_{V_i}) and the value Y can be computed by $Y = \prod_{i\in[n]} Y_{V_i}^{H_{V_i}(\mathsf{pk}_S, \mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}, m)}$. The outcome is `valid` if and only if

$$c_1 + c_2 = H_S\big(Y_S^{c_1} g^{z_1}, Y^{c_2} g^{z_2}, \mathsf{pk}_S, \mathsf{pk}_{V_1}, h_1, \ldots, \mathsf{pk}_{V_n}, h_n, Y, m\big)$$

and `invalid` otherwise.

Relating to the security of our concrete construction, the theorem, whose proof can be found in security analysis section is presented as follows.

Theorem 4.1 *Our concrete construction is secure under the discrete logarithm assumption under the random oracle model. Particularly, it satisfies*

- *definition 4.1 under the discrete logarithm assumption in the random oracle model;*

- *definition 4.2 unconditionally;*
- *definition 4.4 under the discrete logarithm assumption in the random oracle model.*

4.5 Generic Strong MDVS

Strong DVS can be constructed from DVS by encrypting the signature under the designated verifier's public key. Nonetheless, the intuitive solution that encrypting the signature under each designated verifier's public key is not satisfactory in the case of multiple designated verifiers. As discussed in [50], this intuitive solution produces a subtle issue in correctness. Specifically, the signer could generate an "invalid" signature that would be regarded as valid by some verifiers in some scenarios where the signer does not execute some of the encryption properly.

4.5.1 Generic Construction

To overcome this challenge, we observe that it is straightforward to apply a verifiable encryption [15] allowing the signer to create a proof that all ciphertext decrypts to the same value. To verify the proof, all verifiers need to assure that the values obtained by the signer are identical. However, this solution is costly. As to the view of an abstract level, the goal of this encryption is to ensure all verifiers receive the same value by decryption. This can be achieved by using a very weak one-way encryption with an explicit "IND-CPA" attack. This encryption scheme can be denoted as $\mathcal{WE}$. That is, given a message k, anyone can check if the ciphertext C

decrypts to it. It is easy for a verifier to check whether or not all the encryptions of the designated verifier signature are properly executed.

However, this creates another problem. Due to the fact that $\mathcal{WE}$ is only one-way secure, the ciphertext might leak information about the signature and thus privacy of signer's identity is not guaranteed. Therefore, we employ a hybrid approach to solve this problem. $\mathcal{WE}$ is used to encrypt a symmetric key k under all the designated verifiers public keys and outputs corresponding ciphertexts $C_1, \ldots, C_n$. The ordinary MDVS is encrypted with a symmetric key encryption $\mathcal{SE}$ with key k. As long as the key k cannot be known from the ciphertext C_i's, there is no information about the MDVS being learnt as long as the symmetric encryption $\mathcal{SE}$ is secure. As discussed, $\mathcal{SE}$ is assumed to be an idealized cipher for the ease of security analysis. In other words, our generic construction is secure in the ideal cipher model, which is identical to the random oracle model based on the result of [28].

4.5.2 Building Block of Our Generic Construction

Regarding to $\mathcal{WE}$, two properties are needed. The first one is an efficient and explicit "IND-CPA" attack. The second one is an efficient and explicit malleability attack allowing anyone to transform a ciphertext C under public key Y into another ciphertext C' under public key Y' so that the same message is encrypted by the different verifiers. The malleability attack on $\mathcal{WE}$ is essential in the proof of security for multiple designated verifiers.

Below the requirement of the weakly secure encryption $\mathcal{WE}$ is presented as follows.

- $\mathsf{param}_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Setup}(1^\lambda)$: On input a security parameter λ, the outcome of this algorithm is the public parameter $\mathsf{param}_{\mathcal{WE}}$ for the system. $\mathsf{param}_{\mathcal{WE}}$ is assumed an implicit input to all algorithms listed below.

- $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk}) \leftarrow \mathcal{WE}.\mathsf{Gen}()$: The outcome of this algorithm is a key pair $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk})$.

- $C_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}, m)$: On input a message m and a public key of the receiver $\mathcal{WE}.\mathsf{pk}$, the outcome of this algorithm is the ciphertext $C_{\mathcal{WE}}$.

- $m \leftarrow \mathcal{WE}.\mathsf{Dec}(\mathcal{WE}.\mathsf{sk}, C_{\mathcal{WE}})$: On input a secret key $\mathcal{WE}.\mathsf{sk}$, a ciphertext $C_{\mathcal{WE}}$, the outcome of this algorithm is the plaintext m.

- $0/1 \leftarrow \mathcal{WE}.\mathsf{iAtk}(\mathcal{WE}.\mathsf{pk}, C_{\mathcal{WE}}, m)$: This is an attack on indistinguishability of ciphertext. On input a public key $\mathcal{WE}.\mathsf{pk}$, a ciphertext $C_{\mathcal{WE}}$ and a plaintext m, the outcome is 1 if and only if $m = \mathcal{WE}.\mathsf{Dec}(\mathcal{WE}.\mathsf{sk}, C_{\mathcal{WE}})$, where $\mathcal{WE}.\mathsf{sk}$ is the corresponding private key of $\mathcal{WE}.\mathsf{pk}$ and 0 otherwise. Note that $\mathcal{WE}.\mathsf{sk}$ is not an input to this algorithm.

- $(C'_{\mathcal{WE}}, \mathcal{WE}.\mathsf{pk}') \leftarrow \mathcal{WE}.\mathsf{mAtk}(\mathcal{WE}.\mathsf{pk}, C_{\mathcal{WE}})$: This is an attack on malleability of ciphertext. On input a public key $\mathcal{WE}.\mathsf{pk}$, a ciphertext $C_{\mathcal{WE}}$, the outcome is $C'_{\mathcal{WE}}$, $\mathcal{WE}.\mathsf{pk}'$ such that the distribution of $C'_{\mathcal{WE}}$ is indistinguishable to that of $\mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}', \mathcal{WE}.\mathsf{Dec}(\mathcal{WE}.\mathsf{sk}, C_{\mathcal{WE}}))$. Note that the algorithm does not respond the corresponding secret key for $\mathcal{WE}.\mathsf{pk}'$.

$\mathcal{WE}$ is required to be one-way secure, which is formally defined as the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and then $\mathsf{Gen}()$ to receive ($\mathsf{param}_{\mathcal{WE}}$, $\mathcal{WE}.\mathsf{pk}$, $\mathcal{WE}.\mathsf{sk}$).

Challenge $\mathcal{C}$ selects a random message m, compute $C_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}, m)$. ($\mathsf{param}_{\mathcal{WE}}$, $\mathcal{WE}.\mathsf{pk}$, and $\mathcal{C}$ sends $C_{\mathcal{WE}}$) to $\mathcal{A}$.

Output $\mathcal{A}$ submits m' and win if and only if $m = m'$.

$\mathcal{WE}$ is one-way secure if there does not exist any adversary $\mathcal{A}$ wins the above game with non-negligible probability within PPT.

A construction of $\mathcal{WE}$ is proposed according to the Elgamal encryption in a cyclic group equipped with a bilinear map.

- $\mathcal{WE}.\mathsf{Setup}(1^\lambda)$: Generate a pair of groups $\mathcal{G}, \mathcal{G}_T$ of the same prime order p of λ-bit and a bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}_T$. Let g be a generator of $\mathcal{G}$. Set $\mathsf{param}_{\mathcal{WE}} = (\mathcal{G}, \mathcal{G}_T, p, g, \hat{e})$.

- $\mathcal{WE}.\mathsf{Gen}()$: Randomly select $u \in_R \mathbb{Z}_p$, compute $U = g^u$. Set $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk}) = (U, u)$.

- $\mathcal{WE}.\mathsf{Enc}(U, m)$: On input a message $m \in \mathcal{G}$, randomly select $r \in_R \mathbb{Z}_p$, output $C_{\mathcal{WE}} = (C, D)$ as (mU^r, g^r).

- $\mathcal{WE}.\mathsf{Dec}(u, (C, D))$: Output C/D^u.

- $\mathcal{WE}.\mathsf{iAtk}(U, (C, D), m)$: The outcome is 1 if and only if

$$\hat{e}(C/m, g) = (D, U)$$

and 0 otherwise.

- $\mathcal{WE}.\mathsf{mAtk}(U,(C,D))$: Randomly select $e, f \in_R \mathbb{Z}_p$, compute $U' = Ug^e$. Compute $\tilde{C} = CD^e$, $C' = \tilde{C}U'^f$, $D' = Dg^f$. The outcome is $\big((C', D'), U'\big)$.

Recorded that if $U = g^u$, $C = mU^r$, $D = g^r$, it is easy to see that $C' = m(Ug^e)^{r+f}$, $D = g^{r+f}$ and $U' = Ug^e$. Therefore, (C', D') is encrypting the message m under the public key U' with the proper distribution.

Subsequently, the construction of $\mathcal{WE}$ which is one-way secure under the computational Diffie-Hellman assumption is described.

Proof: Suppose there exists an adversary $\mathcal{A}$ that can win the game one-way security, we show how to construction an algorithm $\mathcal{S}$ that solves the CDH problem in a group equipped with a bilinear map. $\mathcal{S}$ is given $(\mathcal{G}, \mathcal{G}_T, \hat{e}, p, g, g^a, g^b)$ and its goal is to output g^{ab}.

$\mathcal{S}$ randomly picks a value C, gives $\mathsf{param}_{\mathcal{WE}} = (\mathcal{G}, \mathcal{G}_T, p, g, \hat{e})$, $U = g^a$, $(C, D) = (C, g^b)$ to $\mathcal{A}$. Note that this implicit set the message being encrypted as $m = C/g^{ab}$. $\mathcal{A}$ returns with a value m'. $\mathcal{S}$ computes C/m' and outputs it as the solution to the CDH problem. □

4.5.3 Our Generic Construction of Strong MDVS

Here, the generic construction of Strong MDVS is presented. Let $\mathcal{MS} = (\mathcal{MS}.\mathsf{Setup}, \mathcal{MS}.\mathsf{Gen}, \mathcal{MS}.\mathsf{Sign}, \mathcal{MS}.\mathsf{Verify})$ be a secure MDVS scheme. Let $\mathcal{WE} = (\mathcal{WE}.\mathsf{Setup}, \mathcal{WE}.\mathsf{Gen}, \mathcal{WE}.\mathsf{Enc}, \mathcal{WE}.\mathsf{Dec}, \mathcal{WE}.\mathsf{iAtk}, \mathcal{WE}.\mathsf{mAtk})$ be a one-way secure encryption. Let H be a hash function and $\mathcal{SE}$ be a symmetric key encryption. $\mathcal{SE}.\mathsf{Enc}_k$ and $\mathcal{SE}.\mathsf{Dec}_k$ are denoted as encryption and decryption operation of $\mathcal{SE}$ using key k. H, $\mathcal{SE}$ will be modelled as a random oracle and an ideal cipher respectively. The way

how to construct a strong MDVS scheme ($\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}$) is presented as follows.

- Setup. On input security parameter 1^λ, invoke $\mathsf{param}_{\mathcal{MS}} \leftarrow \mathcal{MS}.\mathsf{Setup}(1^\lambda)$ and $\mathsf{param}_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Setup}(1^\lambda)$, specify a weak encryption $\mathcal{WE}$, a hash function H and a symmetric cipher $\mathcal{SE}$. Set $\mathsf{param} = (\mathsf{param}_{\mathcal{MS}}, \mathsf{param}_{\mathcal{WE}}, H, \mathcal{SE})$.

- Gen. Invoke $(\mathcal{MS}.\mathsf{pk}, \mathcal{MS}.\mathsf{sk}) \leftarrow \mathcal{MS}.\mathsf{Gen}()$, $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk}) \leftarrow \mathcal{WE}.\mathsf{Gen}()$. The outcomes are $\mathsf{pk} = (\mathcal{MS}.\mathsf{pk}, \mathcal{WE}.\mathsf{pk})$ and $\mathsf{sk} = (\mathcal{MS}.\mathsf{sk}, \mathcal{WE}.\mathsf{sk})$.

- Sign. Let $\mathsf{pk}_S = (\mathcal{MS}.\mathsf{pk}_S, \mathcal{WE}.\mathsf{pk}_S)$ and $\mathsf{sk}_S = (\mathcal{MS}.\mathsf{sk}_S, \mathcal{WE}.\mathsf{sk}_S)$ be the key pair of the signer. Let m be the message which needs to be signed. Parse the set of verifiers to be $\mathcal{V} = \{\mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}\}$ such that $\mathsf{pk}_{V_i} = (\mathcal{MS}.\mathsf{pk}_i, \mathcal{WE}.\mathsf{pk}_i)$. The set $\{\mathcal{MS}.\mathsf{pk}_1, \ldots, \mathcal{MS}.\mathsf{pk}_n\}$ is denoted by $\mathcal{V}_{\mathcal{MS}}$.

  The signer randomly selects $k \in_R \{0,1\}^\lambda$. For $i = 1$ to n, compute

  $$C_i = \mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}_i, k)$$

  Next, compute $\tau = H(\mathsf{pk}_S, \mathsf{pk}_{V_1}, C_1, \ldots, \mathsf{pk}_{V_n}, C_n, m)$. Invoke $(\sigma_{\mathcal{MS}}, \mathcal{V}_{\mathcal{MS}}) \leftarrow \mathsf{Sign}(\mathcal{MS}.\mathsf{sk}_S, \mathcal{V}_{\mathcal{MS}}, m||\tau)$. Invoke $E = \mathcal{SE}.\mathsf{Enc}_k(\sigma_{\mathcal{MS}}||\tau||\mathsf{pk}_S)$.

  The outcome of the signature is $(E, \mathcal{V}, \{C_i\}_{i\in[n]})$.

- Verify. To verify a signature $(E, \mathcal{V}, \{C_i\}_{i\in[n]})$ on message m, a verifier V parses pk_{V_i} as $(\mathcal{MS}.\mathsf{pk}_i, \mathcal{WE}.\mathsf{pk}_i)$ for all $\mathsf{pk}_{V_i} \in \mathcal{V}$ and makes the use of his secret key $(\mathcal{MS}.\mathsf{sk}_V, \mathcal{WE}.\mathsf{sk}_V)$ as follows.

- Locate the index i in a way that $\mathsf{pk}_V = \mathsf{pk}_{V_i}$. Compute $k = \mathcal{VE}.\mathsf{Dec}(C_i, \mathcal{WE}.\mathsf{sk}_V)$ under his secret key.
- For all $j \in [n] \backslash \{i\}$, check whether the equation of $1 = \mathcal{WE}.\mathsf{iAtk}(\mathcal{WE}.\mathsf{pk}_i, C_i, k)$ holds. The outcome is `invalid` if any of the check outputs 0.
- Compute $\sigma_{\mathcal{MS}}$, τ, pk_S by $\mathcal{SE}.\mathsf{Dec}_k(E)$.
- The outcome is `invalid` if $\tau \neq H(\mathsf{pk}_S, \mathsf{pk}_{V_1}, C_1, \ldots, \mathsf{pk}_{V_n}, C_n, m)$.
- Parse pk_S as $(\mathcal{MS}.\mathsf{pk}_S, \mathcal{WE}.\mathsf{pk}_S)$.
- Invoke `valid`/ `invalid` $\leftarrow \mathcal{MS}.\mathsf{Verify}(\ \mathcal{MS}.\mathsf{pk}_S,\ \sigma_{\mathcal{MS}},\ \{\mathcal{MS}.\mathsf{pk}_1,\ \ldots,\ \mathcal{MS}.\mathsf{pk}_n\},\ m||\tau,\ \mathcal{MS}.\mathsf{sk}_V)$.

With regard to the security of our generic construction, the following theorem whose proof can be found in the following section is presented as follows.

Theorem 4.2 *Our generic construction satisfies definition* 4.1, 4.2, 4.4 *if the underlying MDVS scheme* $\mathcal{MS}$ *satisfies definition* 4.1, 4.2, 4.4. *Moreover, our generic construction satisfies definition 4.3 in the random oracle if* $\mathcal{WE}$ *is one-way secure.*

4.6 Security Analysis

Theorem 4.1 is proven by the following three lemmas.

Lemma 4.3 (Unforgeability) *Under the discrete logarithm assumption, the construction of MDVS satisfies definition 4.1 in the random oracle model.*

Proof: We prove by reduction. Assume there exists a PPT adversary $\mathcal{A}$ who wins in the game Unforgeability with probability ϵ, we present how to construct a simulator

$\mathcal{S}$ solving the discrete logarithm problem with probability at least $\frac{(n-1)\epsilon}{n}(\frac{(n-1)\epsilon}{nq_H} - \frac{1}{p})$, where q_{H_S} is the number of hash query made by $\mathcal{A}$ to the random oracle H_S and n is the number of designated verifiers.

$\mathcal{S}$ obtaines a problem instance $(\mathcal{G}, g, p, Z)$ and its goal is to compute $z \in \mathbb{Z}_p$ such that $Z = g^z$. $\mathcal{S}$ plays the role of the challenger $\mathcal{C}$ to the adversary $\mathcal{A}$.

Setup $\mathcal{S}$ sets param as $(\mathcal{G}, g, p)$, randomly selects an integer n, a value $x_i \in_R \mathbb{Z}_p$ and a hash function H_i for $i = 1$ to n. Select a random index $i^* \in [n]$. For $i \in [n] \setminus \{i^*\}$, set pk_{V_i} as $(Y_{V_i} := g^{x_i}, H_i)$. Set pk_{i^*} as $(Y_{V_i^*} := Z^{x_{i^*}}, H_{i^*})$. Select a hash function H_S and set pk_S as $(Y_S := Z, H_S)$. The set $\{\mathsf{pk}_{V_i}\}_{i\in[n]}$ is denoted by $\mathcal{V}$. $\mathcal{S}$ sends $(\mathsf{param}, \mathsf{pk}_S, \mathcal{V})$ to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to ask the following queries:

- Hash Queries. As to all hash queries to H_S, $\mathcal{S}$ responses with a random value while maintaining consistency. It is necessary that H_S is assumed as a random oracle in this proof.
- Corruption Query. On input a public key $\mathsf{pk}_V \in \mathcal{V}$, locate the index i such that $\mathsf{pk}_V = \mathsf{pk}_{V_i}$. If $i = i^*$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ sends x_i to $\mathcal{A}$.
- Signature Query. $\mathcal{A}$ sends a message m. $\mathcal{S}$ models the signature as follows.
  1. $\mathcal{S}$ computes $h_i = H_{V_i}(\mathsf{pk}_S, \mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}, m)$ and $Y = \prod_{i\in[n]} Y_{V_i}^{h_i}$.
  2. $\mathcal{S}$ selects $c_1, c_2, z_1, z_2 \in_R \mathbb{Z}_p$ at random, assigns the value $c_1 + c_2$ to the hash query of $H_S\big(Y_S^{c_1} g^{z_1}, Y^{c_2} g^{z_2}, \mathsf{pk}_S, \mathsf{pk}_{V_1}, h_1, \ldots, \mathsf{pk}_{V_n}, h_n, Y,$

$m\big)$. If the tuple $\big(Y_S^{c_1} g^{z_1}, Y^{c_2} g^{z_2}, \mathsf{pk}_S, \mathsf{pk}_{V_1}, h_1, \ldots, \mathsf{pk}_{V_n}, h_n, Y, m\big)$ can be found in the set of input query for H_S, $\mathcal{S}$ selects another set of (c_1, c_2, z_1, z_2).

3. $\mathcal{S}$ submits $(c_1, c_2, z_1, z_2, \mathcal{V})$ to $\mathcal{A}$.

Output The probability of not aborting is $\frac{n-1}{n}$ and thus the probability that $\mathcal{A}$ sends a valid signature is $\frac{(n-1)\epsilon}{n}$. Due to the general forking lemma[8], $\mathcal{S}$ can invoke a forking algorithm associated with $\mathcal{A}$ and receives two valid signatures $(c_1, c_2, z_1, z_2, \mathcal{V}, m)$ and $(c_1', c_2', z_1', z_2', \mathcal{V}, m)$ such that $c_1 + c_2 \neq c_1' + c_2'$ with probability at least $\frac{(n-1)\epsilon}{n}(\frac{(n-1)\epsilon}{nq_H} - \frac{1}{p})$.

Solving the Hard Problem Due to $c_1 + c_2 \neq c_1' + c_2'$, either $c_1 \neq c_1'$ or $c_2 \neq c_2'$ is true. $\mathcal{S}$ solves the discrete logarithm problem in each case as follows.

- $c_1 \neq c_1'$: By standard argument, $Y_S^{c_1} g^{z_1} = Y_S^{c_1'} g^{z_1'}$. This implies $Z = g^{\frac{z_1' - z_1}{c_1 - c_1'}}$. $\mathcal{S}$ returns $z = \frac{z_1' - z_1}{c_1 - c_1'}$ as the answer to the discrete logarithm problem.
- $c_2 \neq c_2'$: Similarly, $Y^{c_2} g^{z_2} = Y^{c_2'} g^{z_2'}$. $\mathcal{S}$ first compute $X = \frac{z_1' - z_1}{c_1 - c_1'}$ such that $Y = g^X$. Since $Y = \prod_{i \in [n]} Y_{V_i}^{h_i}$, $X = (\sum_{i \in [n] \setminus \{i^*\}} h_i x_i) + z h_{i^*} x_{i^*}$, where z is the discrete logarithm of Z to the base g. $\mathcal{S}$ replies $\frac{X - (\sum_{i \in [n] \setminus \{i^*\}} h_i x_i)}{(h_i^* x_{i^*})}$ as the answer to the discrete logarithm problem. □

Lemma 4.4 (Source Hiding) *Our construction of MDVS satisfies definition 4.2 unconditionally.*

Proof: An algorithm Sim is described as below. We follow the view in [27] in which this algorithm should be distributed in a way that verifiers do not disclose their

secret keys to other verifiers.

Sim. On input the signer's public key $\mathsf{pk}_S = (Y_S, H_S)$, a set of designated verifier's public key $\mathcal{V} = \{\mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}\}$ and a message m, each individual verifier executes the following:

1. For $i = 1$ to n, parse the public key pk_{V_i} as (Y_{V_i}, H_{V_i}), compute $h_i = H_{V_i}(\mathsf{pk}_S, \mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}, m)$ and $Y = \prod_{i\in[n]} Y_{V_i}^{h_i}$.

2. Verifier V_i selects r_i, $c_{1,i}$, $z_{1,i} \in \mathbb{Z}_p$ ar random, computes $T_{2,i} = g^{r_i}$, $T_{1,i} = {Y_S}^{c_{1,i}} g^{z_{1,i}}$ and shows $T_{1,i}$, $T_{2,i}$, $c_{1,i}$, $z_{1,i}$ to all other verifiers.

3. Upon receiving all the $T_{1,i}$, $T_{2,i}$ for $i \in [n]$, all verifiers compute locally $T_1 = \prod_{i\in[n]} T_{1,i}$, $T_2 = \prod_{i\in[n]} T_{2,i}$, $c_1 = \sum_{i\in[n]} c_{1,i}$ and $z_1 = \sum_{i\in[n]} z_{1,i}$.

4. All verifiers compute locally $c = H_S(T_1, T_2, \mathsf{pk}_S, \mathsf{pk}_{V_1}, h_1, \ldots, \mathsf{pk}_{V_n}, h_n, Y, m)$ and $c_2 = c - c_1$.

5. Verifier V_i computes $z_{2,i} = r_i - c_2 h_i x_{V_i}$, where x_{V_i} is the secret key of the verifier V_i, and shows the value $z_{2,i}$.

6. Upon receiving all the $z_{2,i}$ for $i \in [n]$ from other verifiers, all verifiers compute locally $z_2 = \sum_{i\in[n]} z_{2,i}$

7. All verifiers can output locally the signature as $(c_1, c_2, z_1, z_2, \mathcal{V})$.

The distribution of the signature which is the outcome of Sim is equivalent to that outputted by Sign and thus *source hiding* holds unconditionally. □

Lemma 4.5 (Unforgeability Against Rogue Key Attack) *The construction of MDVS satisfies definition 4.4 under the DL assumption in the random oracle model.*

Proof: We prove by reduction. Suppose there exists a PPT adversary $\mathcal{A}$ winning the game Unforgeability Against Rogue Key Attack with probability ϵ, we show how to construct a simulator $\mathcal{S}$ solving the discrete logarithm problem with probability at least $\frac{\epsilon^4}{q_{H_S}^2 q_{H_V}} - \frac{1}{p}(\frac{2+q_{H_V}}{q_{H_S} q_{H_V}})$, where q_{H_S}, q_{H_V} are the number of hash queries made by $\mathcal{A}$ to the random oracles H_S and H_V respectively.

$\mathcal{S}$ obtains a problem instance $(\mathcal{G}, g, p, Z)$ and its goal is to compute $z \in \mathbb{Z}_p$ such that $Z = g^z$. $\mathcal{S}$ plays the role of the challenger $\mathcal{C}$ to the adversary $\mathcal{A}$.

Setup $\mathcal{S}$ sets param as $(\mathcal{G}, g, p)$, randomly selects $w \in_R \mathbb{Z}_p$ and two hash functions H_S, H_V. Parse pk_S and pk_V as $(Y_S := Z, H_S)$ and $(Y_V := Z^w, H_V)$ respectively. $\mathcal{S}$ sends $(\mathsf{param}, \mathsf{pk}_S, \mathcal{V})$ to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to ask the following queries:

- Hash Queries. For each hash query to H_S or H_V, $\mathcal{S}$ replies with a new random value while maintaining consistency.
- Verification Query. Verification in our construction does not demand the secret key and can thus be simulated perfectly.
- Signature Query. $\mathcal{A}$ sends a message m and a set of public keys $\mathcal{V}$. $\mathcal{S}$ exectues the signature as follows.
  1. Let $\ell = |\mathcal{V}|$. $\mathcal{S}$ parses $\mathcal{V}$ as $\{\mathsf{pk}_1 := (Y_1, H_1), \ldots, \mathsf{pk}_\ell := (Y_\ell, H_\ell)\}$.
  2. $\mathcal{S}$ computes $h_j = H_j(\mathsf{pk}_S, \mathsf{pk}_1, \ldots, \mathsf{pk}_\ell, m)$ and $Y = \prod_{j \in [\ell]} Y_j^{h_j}$.

3. $\mathcal{S}$ randomly selects $c_1, c_2, z_1, z_2 \in_R \mathbb{Z}_p$, assigns the value $c_1 + c_2$ to the hash query of $H_S\big(Y_S^{c_1} g^{z_1},\ Y^{c_2} g^{z_2},\ \mathsf{pk}_S,\ \mathsf{pk}_1,\ h_1,\ \ldots,\ \mathsf{pk}_\ell,\ h_\ell,\ Y,\ m\big)$. If the tuple $\big(Y_S^{c_1} g^{z_1},\ Y^{c_2} g^{z_2},\ \mathsf{pk}_S,\ \mathsf{pk}_1,\ h_1,\ \ldots,\ \mathsf{pk}_\ell,\ h_\ell,\ Y,\ m\big)$ can not be found in the input query for H_S, $\mathcal{S}$ selects another set of (c_1, c_2, z_1, z_2).

4. $\mathcal{S}$ replies $(c_1, c_2, z_1, z_2, \mathcal{V})$ to $\mathcal{A}$.

Output $\mathcal{A}$ sends a valid forgery (σ^*, m^*) and a set of verifiers $\mathcal{V}^* = \{\mathsf{pk}_{V_1},\ \ldots,\ \mathsf{pk}_{V_{n^*}}\}$. Treating H_S as a random oracle, $\mathcal{S}$ can invoke a forking algorithm associated with $\mathcal{A}$ with respect to random oracle H_S and receives two valid signatures $\sigma_1 := (c_1,\ c_2,\ z_1,\ z_2,\ \mathcal{V}^*,\ m^*)$ and $\sigma_2 := (c_1',\ c_2',\ z_1',\ z_2',\ \mathcal{V}^*,\ m^*)$ such that $c_1 + c_2 \neq c_1' + c_2'$. The value is denoted as $Y = \prod_{i \in [n^*]} Y_{V_i}^{H_{V_i}(\mathsf{pk}_S, \mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}, m)}$. Using the same argument as in the proof of Lemma 4.3, $\mathcal{S}$ receives a value z such that $Y_S = g^z$ or $Y = g^z$. In the previous case, $\mathcal{S}$ successfully submits the solution of the discrete logarithm problem. Next, show how $\mathcal{S}$ solves the discrete logarithm problem in the latter situation.

The Second Rewind To solve the discrete logarithm in the second case, $\mathcal{S}$ makes another rewind simulation to the case when $\mathcal{A}$ issues a H_V query for $H_V(\mathsf{pk}_S,\ \mathsf{pk}_1,\ \ldots,\ \mathsf{pk}_\ell, m^*)$ associated with the forged signature. $\mathcal{S}$ invokes the forking algorithm associated with $\mathcal{A}$ with respect to random oracle H_V to receive another forged signature $\sigma_3 := (\tilde{c}_1, \tilde{c}_2, \tilde{z}_1, \tilde{z}_2, \mathcal{V}^*, m^*)$. Afterwards, $\mathcal{S}$ makes another rewind simulation regarding to random oracle H_S again to receive $\sigma_4 := (\hat{c}_1, \hat{c}_2, \hat{z}_1, \hat{z}_2, \mathcal{V}^*, m^*)$. Due to the output of the H_V query is

an input to the random oracle H_S in the forged signature, $\mathcal{A}$ must have made the H_V query before marking the query to H_S. Therefore, σ_3 and σ_4 will involve the same set of public keys compared with σ_1 and σ_2. Moreover, the value $H_V(\mathsf{pk}_S, \mathsf{pk}_1, \ldots, \mathsf{pk}_\ell, m^*)$ in (σ_1, σ_2) (say, h_V) is different to the value in the signatures (σ_3, σ_4) (say, h'_V) . Therefore, the value $Y' = \prod_{i\in[n^*]} Y_{V_i}^{H_{V_i}(\mathsf{pk}_S,\mathsf{pk}_{V_1},\ldots,\mathsf{pk}_{V_n},m)}$ is different to Y. With signatures (σ_3, σ_4), $\mathcal{S}$ can compute the value z' such that $Y_S = g^{z'}$ or $Y' = g^{z'}$. If the former is true, $\mathcal{S}$ can submit the solution of the discrete logarithm problem directly. Otherwise, the value $(Y/Y') = g^{z-z'}$. Recorded that $\frac{Y}{Y'} = Z^{wh_V - wh'_V}$. Thus, $\mathcal{S}$ can output $(z - z')(w(h_V - h'_V))^{-1}$ as the solution to the discrete logarithm problem.

Probability Analysis The probability of $\mathcal{A}$ submitting a valid forgery is ϵ. The success probability of invoking the forking algorithm associated with $\mathcal{A}$, denoted as $\mathcal{F}_\mathcal{A}$, with respect to the random oracle H_S is at least $\epsilon' := \epsilon(\frac{\epsilon}{q_{H_S}} - \frac{1}{p})$ due to the general forking lemma [8]. The success probability of invoking the forking algorithm associated with $\mathcal{F}_\mathcal{A}$ with respect to random oracle H_V is at least $\epsilon'(\frac{\epsilon'}{q_{H_V}} - \frac{1}{p})$, again due to the general forking lemma [6]. The overall success probability of $\mathcal{S}$ is thus bounded below by

$$\begin{aligned}
\epsilon'(\tfrac{\epsilon'}{q_{H_V}} - \tfrac{1}{p}) &= \tfrac{\epsilon'^2}{q_{H_V}} - \tfrac{\epsilon'}{p} \\
&= \tfrac{\epsilon^2(\epsilon/q_{H_S} - 1/p)^2}{q_{H_V}} - \tfrac{\epsilon}{p}(\tfrac{\epsilon}{q_{H_S}} - \tfrac{1}{p}) \\
&\geq \tfrac{\epsilon^4}{q_{H_S}^2 q_{H_V}} - \tfrac{2\epsilon^3}{q_{H_S} q_{H_V} p} - \tfrac{\epsilon^2}{q_{H_S} p} \\
&\geq \tfrac{\epsilon^4}{q_{H_S}^2 q_{H_V}} - \tfrac{2+q_{H_V}}{p q_{H_S} q_{H_V}}
\end{aligned}$$

□

4.6.1 Proof of Theorem 4.2

We sketch the proof of theorem 4.2 as follows.

Proof: The proof that our generic construction satisfies definitions 4.1,4.2,4.4 if the underlying MDVS $\mathcal{MS}$ satisfies the corresponding definitions is straightforward. We can easily construct a simulator $\mathcal{S}$ that acts as an attacker against $\mathcal{MS}$ if there exists an adversary $\mathcal{A}$ against the generic construction. $\mathcal{S}$ receives the system parameters of $\mathcal{MS}$, creates the public and secret keys of a one-way encryption $\mathcal{WE}$ for all users and gives them to $\mathcal{A}$. For any verification query submitted by $\mathcal{A}$, $\mathcal{S}$ can use the secret key of the $\mathcal{WE}$ to translate it to an appropriate query to the $\mathcal{MS}$. Finally, when $\mathcal{A}$ submits a forgery, $\mathcal{S}$ can decrypt it to produce the forgery of the underlying $\mathcal{MS}$.

We focus on the proof against privacy of signer's identity. Let $\mathcal{A}$ be an adversary with non-negligible advantage in game PSI. We show how to construct a simulator $\mathcal{S}$ that breaks the one-way security of the encryption $\mathcal{WE}$ in the ideal cipher model. The ideal cipher model is equivalent to the random oracle model [28] and is thus subsumed into the random oracle model in the theorem statement. $\mathcal{S}$ is given an instance of $\mathcal{WE} = (\mathsf{param}_{\mathcal{WE}}, \mathcal{WE}.\mathsf{pk})$ and a challenge ciphertext $C_{\mathcal{WE}}$ and its goal is to output the corresponding plaintext. $\mathcal{S}$ creates the parameter of $\mathcal{MS}$ honestly and is in possession of the keys for the two signers and the set of verifiers, say $(\mathcal{MS}.\mathsf{pk}_i, \mathcal{MS}.\mathsf{sk}_i)$ for verifier V_i. $\mathcal{S}$ invokes $\mathcal{WE}.\mathsf{mAtk}(\mathcal{WE}.\mathsf{pk}, C_{\mathcal{WE}})$ repeatedly to obtain a set of public keys and ciphertexts $(C^*_{\mathcal{WE},i}, \mathcal{WE}.\mathsf{pk}_i)$. Set the public keys of

the verifiers as $(\mathcal{MS}.\mathsf{pk}_i, \mathcal{WE}.\mathsf{pk}_i)$ for verifier V_i. $\mathcal{S}$ chooses a symmetric cipher $\mathcal{SE}$ and a hash function H. $\mathcal{SE}$ is modelled as an idealized cipher.

Setup $\mathcal{S}$ gives $(\mathcal{MS}.\mathsf{param}, \mathsf{param}_{\mathcal{WE}}, \mathcal{SE}, H)$ to $\mathcal{A}$ as param, together with the public keys of the signers and verifiers.

Query $\mathcal{S}$ uses the secret key of the signers to answer signature queries. Other types of queries are discussed below.

- For verification query of signature $(E, \mathcal{V}, \{C_i\}_{i\in[n]})$ on message m, $\mathcal{S}$ looks through the encryption query on $\mathcal{SE}$ with key k that produces E. If the query is not found, return `invalid`. Otherwise, $\mathcal{S}$ checks if $1 = \mathcal{WE}.\mathsf{iAtk}(\mathcal{WE}.\mathsf{pk}_i, C_i, k)$ for all $i = 1$ to n. If not, $\mathcal{S}$ returns `invalid`. $\mathcal{S}$ obtains $(\sigma_{\mathcal{MS}}, \tau, \mathsf{pk}_S)$ from the encryption query of $\mathcal{SE}$ with output E. $\mathcal{S}$ returns `invalid` if $\tau \neq H(\mathsf{pk}_S, \mathsf{pk}_{V_1}, C_1, \ldots, \mathsf{pk}_{V_n}, C_n, m)$. Otherwise, $\mathcal{S}$ returns the verification results of $\sigma_{\mathcal{MS}}$ with secret key $\mathcal{MS}.\mathsf{sk}_i$.
- For each encryption or decryption query submitted by $\mathcal{A}$ with key k, $\mathcal{S}$ invokes $0/1 \leftarrow \mathcal{WE}.\mathsf{iAtk}(\mathcal{WE}.\mathsf{pk}, C_{\mathcal{WE}}, k)$. If the result is 1, $\mathcal{S}$ returns k and break the one-way security of $\mathcal{WE}$.

Challenge At some point $\mathcal{A}$ submits a message m^*. $\mathcal{S}$ flips a fair coin $b \in \{0, 1\}$ and computes $\tau = H(\mathsf{pk}_{S_b}, \mathsf{pk}_{V_1}, C^*_{\mathcal{WE},1}, \ldots, \mathsf{pk}_{V_n}, C^*_{\mathcal{WE},n}, m)$. Invoke $(\sigma_{\mathcal{MS}}, \mathcal{V}_{\mathcal{MS}}) \leftarrow \mathsf{Sign}(\mathcal{MS}.\mathsf{sk}_{S_b}, \mathcal{V}_{\mathcal{MS}}, m^*||\tau)$. Pick a random value E^*. Return $(E^*, \mathcal{V}, \{C^*_{\mathcal{WE},i}\}_{i\in[n]})$ as the challenge signature. This implicitly sets $E^* = \mathcal{SE}.\mathsf{Enc}_k(\sigma_{\mathcal{MS}}||\tau||\mathsf{pk}_{S_b})$ where k is the value encrypted in $C^*_{\mathcal{WE},i}$ for $i = 1$ to n.

Query Same as the previous query phase.

Output $\mathcal{A}$ outputs a guess bit b'.

It remains to argue $\mathcal{A}$ must submit a decryption or encryption query with value k such that $1 = \mathcal{WE}.\mathsf{iAtk}(\mathcal{WE}.\mathsf{pk}, C_{\mathcal{WE}}, k)$. Since anything related to the value b is encrypted with a value k which is encrypted in $C^*_{\mathcal{WE},i}$ for $i = 1$ to n, the only way for $\mathcal{A}$ to win is to learn anything about b is to issue an encryption or decryption query with k. Thus, the probability that $\mathcal{S}$ can break the one-way encryption $\mathcal{WE}$ is equivalent to the advantage of $\mathcal{A}$ in game PSI. □

Chapter 5

Conclusions and Future Work

5.1 Overview

Among a variety of digital signatures with different properties in various settings, our focus is on Multi-Designated Verifiers Signatures in public key infrastructure setting and Identity-Based Signature. As to Identity-Based Signature, we proposed a new escrow-free identity-based signature scheme. This scheme is key escrow free. In other words, any malicious PKG who pretends to be any users to put a signature can be detected. In addition, the concrete construction is very efficient in terms of space complexity, and consists of only two group elements. Regarding to Multi-Designated Verifiers Signature, our goal is to propose a construction that is provably secure against rogue key attack and present a generic construction of strong MDVS secure against rogue key attack.

We conducted a thorough study about digital signatures. As discussed, digital signature has different settings such as public-key infrastructure, identity-based cryptography, certificate-based cryptography and certificateless public key cryptography. With regard to different properties, we studied verifiable encrypted signature,

blind signature, directed signature, undeniable signature, designated verifier signature, aggregate signature, group signature, ring signature, threshold signature and multi-designated verifiers signature.

We reviewed the literature in detail and identified key escrow problem in Identity-Based Signature and rogue key attack problem in Multi-Designated Verifier Signature. By comparing different approaches to key escrow problem in identity-based signature, a new escrow-free IBS scheme is proposed. We also studied different approaches to rough key attack in multi-designated verifiers signature and propose a new approach which does not need to make the knowledge of secret key assumption.

As to IBS, we compare various IBS schemes with our proposed scheme based on efficiency and security features. In addition, we presented a generic construction of escrow-free identity-based signature as well as an efficient instantiation. In contrast to the prior work due to Yuen, our scheme is very practical, since it allows the public to determine whether the PKG has behaved maliciously given the evidence produced by the honest user. Further, the instantiation is very efficient since the signature size only comprises two group elements, which outperforms the existing schemes in the literature. Moreover, a complexity analysis based on empirical results as well as experimental results are provided.

Regarding multi-designated verifiers signature, we formalized the security notion unforgeability under rogue key attack. Additonally, an efficient construction that is provably secure in the new model was proposed. Furthermore, we present a generic transformation that convert any secure MDVS scheme into a strong MDVS scheme.

5.2 Future Work

For future research area, we envisage to provide a solution of key escrow problem without using random oracles. By extending the research, we can provide efficient key-escrow identity-based signature in standard model. Our proposed scheme possess Unforgeability, Non-slan-derability and Non frameability under the CDH assumption in the random oracle model. The security of the construction is analyzed in Chapter 3. Firstly, we can consider the signature scheme which is secure under key-escrow problem in Identity-based signature where signatures are as short as our proposed scheme, but the random oracles are not required in the security requirements. However, the signature can be as short as current proposed signature, but can be provably secure in the standard model.

With regard to multi-designated verifiers signature, we can extend the research to threshold undeniable signature where the group signature can be created when the number of participating group members who can represent the whole group is larger than the threshold number. We can construct the scheme without the assistance of trusted party. In addition, this approach can be extended to the signature scheme which is provably secure without using random oracles. Specifically, the generic strong MDVS scheme which is proposed IND-CCA2 secure in the random oracle model can be constructed IND-CCA2 secure in the standard model.

In terms of the size of signature, we leave the construction of constant size strong MDVS secure as an open problem. Recall that the size of the signature is linear in the number of designated verifiers. Therefore, the proposed scheme can be more

efficient if the length of the signature is independent of the number of verifiers. Furthermore, we can extend it to a signature scheme where there is no necessity to modify the public key when to add a new member to the group. We have planned to implement these schemes and to check and compare the efficiency of our schemes with the existing protocols.

Bibliography

[1] S. S. Al-Riyami and K. G. Paterson. Certificateless public key cryptography. *IACR Cryptology ePrint Archive*, 2003:126, 2003.

[2] G. Ateniese. Verifiable encryption of digital signatures and applications. *ACM Transactions on Information and System Security (TISSEC)*, 7(1):1–20, 2004.

[3] M. Au, J. Liu, W. Susilo, and T. Yuen. Certificate based (linkable) ring signature. *Information Security Practice and Experience*, pages 79–92, 2007.

[4] M. Au, J. Liu, T. Yuen, and D. Wong. Id-based ring signature scheme secure in the standard model. *Advances in Information and Computer Security*, pages 1–16, 2006.

[5] J. Baek and Y. Zheng. Identity-based threshold signature scheme from the bilinear pairings. In *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, volume 1, pages 124–128. IEEE, 2004.

[6] M. Bellare, J. A. Garay, and T. Rabin. Fast Batch Verification for Modular Exponentiation and Digital Signatures. In *EUROCRYPT*, pages 236–250, 1998.

[7] M. Bellare, C. Namprempre, and G. Neven. Security Proofs for Identity-Based Identification and Signature Schemes. In *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 268–286. Springer, 2004.

[8] M. Bellare and G. Neven. Multi-Signatures in the Plain Public-Key Model and a General Forking Lemma. In *ACM Conference on Computer and Communications Security*, pages 390–399, 2006.

[9] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[10] D. Boneh and X. Boyen. Short Signatures without Random Oracles. In *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 2004.

[11] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.

[12] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. *Advances in Cryptology-EUROCRYPT 2003*, pages 641–641, 2003.

[13] D. Boneh, B. Lynn, and H. Shacham. Short Signatures from the Weil Pairing. In *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2001.

[14] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004.

[15] J. Camenisch and V. Shoup. Practical Verifiable Encryption and Decryption of Discrete Logarithms. In *CRYPTO*, pages 126–144, 2003.

[16] J. Camenisch and M. Stadler. Efficient Group Signature Schemes for Large Groups (Extended Abstract). In *CRYPTO*, pages 410–424, 1997.

[17] S. Chang, D. Wong, Y. Mu, and Z. Zhang. Certificateless threshold ring signature. *Information Sciences*, 179(20):3685–3696, 2009.

[18] T. Y. Chang. An ID-Based Multi-Signer Universal Designated Multi-Verifier Signature Scheme. *Inf. Comput.*, 209(7):1007–1015, 2011.

[19] D. Chaum. Blind signature system. In *CRYPTO*, page 153, 1983.

[20] D. Chaum. Private Signature and Proof Systems, 1996. US Patent 5,493,614.

[21] D. Chaum and H. V. Antwerpen. Undeniable signatures. In *CRYPTO*, pages 212–216, 1989.

[22] D. Chaum and H. Van Antwerpen. Undeniable signatures. In *Advances in Cryptology-CRYPTO89 Proceedings*, pages 212–216. Springer, 1990.

[23] D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.

[24] D. Chaum and E. Van Heyst. Group signatures. In *Advances in CryptologyEUROCRYPT91*, pages 257–265. Springer, 1991.

[25] K. Choi, J. Park, J. Hwang, and D. Lee. Efficient certificateless signature schemes. In *Applied Cryptography and Network Security*, pages 443–458. Springer, 2007.

[26] S. Chow. Identity-based strong multi-designated verifiers signatures. *Public Key Infrastructure*, pages 257–259, 2006.

[27] S. S. M. Chow. Multi-Designated Verifiers Signatures Revisited. *I. J. Network Security*, 7(3):348–357, 2008.

[28] J.-S. Coron, J. Patarin, and Y. Seurin. The random oracle model and the ideal cipher model are equivalent. In *CRYPTO*, pages 1–20, 2008.

[29] V. Daza, J. Herranz, and G. Sáez. Some applications of threshold signature schemes to distributed protocols. *IACR Cryptology ePrint Archive*, 2002:81, 2002.

[30] Y. Desmedt. Verifier-Designated Signatures. *CRYPTO Rump Session*, 2003.

[31] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[32] S. Duan. Certificateless undeniable signature scheme. *Information Sciences*, 178(3):742–755, 2008.

[33] S. Galbraith, W. Mao, and K. Paterson. Rsa-based undeniable signatures for general moduli. *Topics in CryptologyCT-RSA 2002*, pages 200–217, 2002.

[34] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.

[35] C. Gentry. Certificate-based encryption and the certificate revocation problem. *IACR Cryptology ePrint Archive*, 2003:183, 2003.

[36] C. Gentry and Z. Ramzan. Identity-based aggregate signatures. *Public Key Cryptography-PKC 2006*, pages 257–273, 2006.

[37] M. Girault. Self-Certified Public Keys. In *EUROCRYPT*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer, 1991.

[38] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[39] V. Goyal. Reducing Trust in the PKG in Identity Based Cryptosystems. In *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 430–447. Springer, 2007.

[40] V. Goyal, S. Lu, A. Sahai, and B. Waters. Black-box Accountable Authority Identity-Based Encryption. In *ACM Conference on Computer and Communications Security*, pages 427–436. ACM, 2008.

[41] C. Gu and Y. Zhu. An id-based verifiable encrypted signature scheme based on hesss scheme. In *Information Security and Cryptology*, pages 42–52. Springer, 2005.

[42] F. Hess. Efficient Identity Based Signature Schemes Based on Pairings. In *Selected Area in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 310–324. Springer, 2002.

[43] Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Efficient Strong Designated Verifier Signature Schemes without Random Oracle or with Non-Delegatability. *Int. J. Inf. Sec.*, 10(6):373–385, 2011.

[44] X. Huang, W. Susilo, Y. Mu, and F. Zhang. Certificateless designated verifier signature schemes. In *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on*, volume 2, pages 15–19. IEEE, 2006.

[45] Z. Huang, Q. Chen, and Y. Hao. Certificate-based blind signatures from bilinear pairings. In *Information Science and Engineering (ICISE), 2010 2nd International Conference on*, pages 1903 –1906, dec. 2010.

[46] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In *EUROCRYPT*, pages 143–154, 1996.

[47] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In *Advances in CryptologyEUROCRYPT96*, pages 143–154. Springer, 1996.

[48] W. Jamroga. Multilevel modeling of dialogue environment for e-commerce agents. *Electronic Commerce-gospodarka XXI wieku*, pages 49–53, 2001.

[49] B. G. Kang, J. H. Park, and S. G. Hahn. A certificate-based signature scheme. In *CT-RSA*, pages 99–111, 2004.

[50] F. Laguillaumie and D. Vergnaud. Multi-designated verifiers signatures. In *ICICS*, pages 495–507, 2004.

[51] F. Laguillaumie and D. Vergnaud. Multi-designated verifiers signatures. *Information and Communications Security*, pages 495–507, 2004.

[52] F. Laguillaumie and D. Vergnaud. Multi-Designated Verifiers Signatures: Anonymity without Encryption. *Inf. Process. Lett.*, 102(2-3):127–132, 2007.

[53] S. Lal and M. Kumar. A directed signature scheme and its applications. *arXiv preprint cs/0409036*, 2004.

[54] S. Lal and M. Kumar. Some applications of directed signature scheme. *arXiv preprint cs/0409050*, 2004.

[55] Y. Li, W. Susilo, Y. Mu, and D. Pei. Designated Verifier Signature: Definition, Framework and New Constructions. In *UIC*, pages 1191–1200, 2007.

[56] B. Libert and J. Quisquater. Identity based undeniable signatures. *Topics in Cryptology–CT-RSA 2004*, pages 1997–1997, 2004.

[57] C. Lim and P. Lee. Modified maurer-yacobi's scheme and its applications. In *Advances in CryptologyAUSCRYPT'92*, pages 308–323. Springer, 1993.

[58] J. Liu, J. Baek, and J. Zhou. Certificate-based sequential aggregate signature. In *Proceedings of the second ACM conference on Wireless network security*, pages 21–28. ACM, 2009.

[59] J. K. Liu, M. H. Au, and W. Susilo. Self-Generated-Certificate Public Key Cryptography and Certificateless Signature / Encryption Scheme in the Standard Model. In *ASIACCS 07*, pages 273–283. ACM Press, 2007.

[60] Y. Lu, J. Li, and J. Xiao. Threshold certificate-based encryption. *Journal of Software*, 4(3):210–217, 2009.

[61] C. Y. Ng, W. Susilo, and Y. Mu. Universal designated multi verifier signature schemes. In *ICPADS (2)*, pages 305–309, 2005.

[62] K. G. Paterson and J. C. N. Schuldt. Efficient Identity-Based Signatures Secure in the Standard Model. In *ACISP*, volume 4058 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2006.

[63] D. Pointcheval and J. Stern. Provably secure blind signature schemes. In *ASIACRYPT*, pages 252–265, 1996.

[64] C. Popescu. An efficient id-based group signature scheme. *Studia Univ. Babes-Bolyai, Informatica*, 47(2):29–38, 2002.

[65] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Cambridge, MA, USA, 1979.

[66] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[67] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT*, pages 552–565, 2001.

[68] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret: Theory and applications of ring signatures. In *Essays in Memory of Shimon Even*, pages 164–186, 2006.

[69] S. Saeednia, S. Kremer, and O. Markowitch. An Efficient Strong Designated Verifier Signature Scheme. In *ICISC*, pages 40–54, 2003.

[70] G. Shailaja, K. P. Kumar, and A. Saxena. Universal Designated Multi Verifier Signature without Random Oracles. In *ICIT*, pages 168–171, 2006.

[71] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.

[72] Z. Shao. Certificate-based verifiably encrypted signatures from pairings. *Information Sciences*, 178(10):2360–2373, 2008.

[73] K.-A. Shim. Rogue-key Attacks on the Multi-designated Verifiers Signature Scheme. *Inf. Process. Lett.*, 107(2):83–86, 2008.

[74] V. Shoup. Practical threshold signatures. In *Advances in Cryptology-EUROCRYPT 2000*, pages 207–220. Springer, 2000.

[75] X. Sun, J. Li, G. Chen, and S. Yung. Identity-based directed signature scheme from bilinear pairings. Technical report, Cryptology eprint Archive, Report 2008/305, 2008. http://eprint. iacr. org, 2008.

[76] W. Susilo, F. Zhang, and Y. Mu. Identity-based strong designated verifier signature schemes. In *Information Security and Privacy*, pages 313–324. Springer, 2004.

[77] H. Tian. A New Strong Multiple Designated Verifiers Signature for Broadcast Propagation. In *INCoS*, pages 268–274, 2011.

[78] H. Tian. A New Strong Multiple Designated Verifiers Signature. *IJGUC*, 3(1):1–11, 2012.

[79] D. Vergnaud. New Extensions of Pairing-based Signatures into Universal (Multi) Designated Verifier Signatures. *CoRR*, abs/0802.1076, 2008.

[80] Z. Wan. Certificateless directed signature scheme. In *Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on*, pages 1–4. IEEE, 2011.

[81] B. Waters. Efficient Identity-Based Encryption without Random Oracles. In *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer, 2005.

[82] X. Xia, F. Hong, and G. Cui. A forward certificateless ring signature scheme. In *Multimedia Technology (ICMT), 2011 International Conference on*, pages 3315–3318. IEEE, 2011.

[83] B. Yang, Z. Xiao, and S. Li. Certificateless verifiably encrypted signature scheme. In *Network Infrastructure and Digital Content, 2010 2nd IEEE International Conference on*, pages 783–788. IEEE, 2010.

[84] H. Yuan, F. Zhang, X. Huang, Y. Mu, W. Susilo, and L. Zhang. Certificateless threshold signature scheme from bilinear maps. *Information Sciences*, 180(23):4714–4728, 2010.

[85] T. H. Yuen, W. Susilo, and Y. Mu. How to Construct Identity-Based Signatures without the Key Escrow Problem. *Int. J. Inf. Sec.*, 9(4):297–311, 2010.

[86] F. Zhang and K. Kim. Id-based blind signature and ring signature from pairings. In *ASIACRYPT*, pages 533–547, 2002.

[87] L. Zhang and F. Zhang. A new certificateless aggregate signature scheme. *Computer Communications*, 32(6):1079–1085, 2009.

[88] Y. Zhang, J. K. Liu, X. Huang, M. H. Au, and W. Susilo. Efficient escrow-free identity-based signature. In *ProvSec*, pages 161–174, 2012.