

2011

Optimistic fair exchange of ring signatures

Lie Qu

University of Wollongong, lq594@uowmail.edu.au

Recommended Citation

Qu, Lie, Optimistic fair exchange of ring signatures, Master of Computer Science - Research thesis, School of Computer Science and Software Engineering, University of Wollongong, 2011. <http://ro.uow.edu.au/theses/3436>

Research Online is the open access institutional repository for the University of Wollongong. For further information contact Manager Repository Services: morgan@uow.edu.au.

UNIVERSITY OF WOLLONGONG

COPYRIGHT WARNING

You may print or download ONE copy of this document for the purpose of your own research or study. The University does not authorise you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site. You are reminded of the following:

Copyright owners are entitled to take legal action against persons who infringe their copyright. A reproduction of material that is protected by copyright may be a copyright infringement. A court may impose penalties and award damages in relation to offences and infringements relating to copyright material. Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.



Optimistic Fair Exchange of Ring Signatures

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

Master of Computer Science

from

UNIVERSITY OF WOLLONGONG

by

Lie Qu

School of Computer Science and Software Engineering
December 2011

© Copyright 2011

by

Lie Qu

All Rights Reserved

Dedicated to

My Family

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.

Lie Qu
December 2, 2011

Abstract

Fair exchange of digital items via computer networks is an important research topic in modern cryptography. Generally speaking, a fair exchange protocol can help two mistrusted parties in networks exchange their digital items in a fair way, that is, both two parties get each other's item, or neither of them gets anything valuable after the protocol runs successfully. In practical applications, the technology of fair exchange is widely used in different but relevant fields, such as contract signing protocols, non-repudiation protocols, e-payment system and certified e-mails.

In a fair exchange protocol, a trusted third party (TTP) is usually needed as an authentic mediator between two mistrusted parties. In order to reduce the load of such a TTP, the notion of *optimistic fair exchange* (OFE) is proposed, in which there is an off-line TTP, called *arbitrator*, who acts as a judge to settle the dispute between parties and should only be involved if necessary. In previous studies, fair exchange is usually carried out between individual parties. When a fair exchange protocol runs between two members from two distinct groups, anonymity of the signer in a group could be necessary for achieving better privacy in some cases (e.g. protecting customers' trading habits in e-payment systems).

In this thesis, we study optimistic fair exchange of ring signatures (OFERS), i.e., two members from two different groups can exchange their ring signatures in an equitable way. Each user in these groups has his/her own public-private key pair and is able to sign a message on behalf of his/her group anonymously due to the property *signer ambiguity* inherited from ring signatures. We first define the security model of OFERS in the multi-user setting under adaptive chosen message, chosen-key and chosen public-key attacks. Then, based on verifiably encrypted ring signatures (VERS), we construct a concrete scheme by combining the technologies of ring signatures, public-key encryption and zero-knowledge proof, and then show that our OFERS solution is provably secure in our security model and preserving *signer ambiguity* of ring signatures. Moreover, we improve the proposed OFERS scheme in

order to meet the property *abuse-freeness*, which means, before the OFE protocol is successfully executed, any interim result in the protocol cannot be accepted as valid evidence showing that some participant has committed to complete the protocol. We propose the formal model of *abuse-freeness* in OFERS, and construct the first concrete scheme of abuse-free optimistic fair exchange of ring signatures (AOFERS), which is formally proven *perfectly abuse-free* in our security model. To the best of our knowledge, this is the first formal work on the topic of optimistic fair exchange of ring signatures.

Acknowledgement

The thesis would not have been possible without the help of many people who contributed their valuable guidance and assistance to my study in one way or another.

First of all, I am very grateful to my supervisors Dr. Guilin Wang and A/Prof. Yi Mu whose supervision, suggestions and encouragement enabled me to not only learn so much on research methodology but also develop a deep understanding of my research field. Without their patient guidance, the achievement in my research would never be possible.

Then I would like to thank my schoolmates Mr. Nan Li and Mr. Jinguang Han who provided helpful suggestions in my research. I would also appreciate all the staff of Center for Computer and Information Security Research and School of Computer Science and Software Engineering for their assistance.

Lastly, I am highly indebted to my family for their unselfish support. Without their encouragement, I would not achieve what I have today.

Lie Qu
Wollongong, August 2011

Publications

Lie Qu, Guilin Wang and Yi Mu. Optimistic Fair Exchange of Ring signatures. In Proc. of *7th International ICST Conference on Security and Privacy in Communication Networks (SecureComm 2011 - ERA Rank A conference)*.

Lie Qu, Guilin Wang and Yi Mu. Abuse-free Optimistic Fair Exchange of Ring signatures. (Draft)

Contents

Abstract	v
Acknowledgement	vii
Publications	viii
1 Introduction	1
1.1 Fair Exchange	1
1.2 Related Work	3
1.3 Challenging Issues	7
1.4 Contributions	8
1.5 Organization of the Thesis	9
2 Background	11
2.1 Cryptographic Hash Functions	11
2.2 Standard Model and Random Oracle Model	12
2.3 Computational Complexity Assumptions	13
2.3.1 Discrete Logarithm Assumption	13
2.3.2 Strong RSA Assumption	13
2.3.3 Decisional Composite Residuosity Assumption	14
2.4 Zero-knowledge Proof	14
2.4.1 Non-interactive Zero-knowledge Proof	15
2.5 Public-key Encryption	17
2.5.1 Security Notions	17
2.6 Digital Signatures	19
2.6.1 Ring Signatures	20
2.6.2 Verifiably Encrypted Signature	21

2.7	Commitment Schemes	22
2.7.1	Trapdoor Commitment Schemes	22
3	Optimistic Fair Exchange of Ring Signatures	24
3.1	Introduction	24
3.2	Building Blocks	25
3.2.1	Abe <i>et al.</i> 's Ring Signature Schemes	25
3.2.2	Zero-knowledge Proof of Equality of Discrete Logarithms from Different Groups	28
3.2.3	Camenisch-Shoup Encryption Scheme	29
3.3	Security Definitions	31
3.4	The Proposed Scheme	36
3.4.1	Verifiably Encrypted Ring Signature	36
3.4.2	Optimistic Fair Exchange of Ring Signatures	38
3.5	Security Proofs	39
3.6	Summary	43
4	Abuse-free Optimistic Fair Exchange of Ring Signatures	45
4.1	Introduction	45
4.2	Security Definitions	46
4.3	The Proposed Scheme	49
4.3.1	All Discrete-log Case	49
4.3.2	Generic Case	53
4.4	Security Analysis	57
4.5	Summary	60
5	Conclusion	61
	Bibliography	63

List of Figures

2.1	[BDPR98] The relations of security notions for public-key encryption schemes	18
4.1	The modified interactive zero-knowledge proof for abuse-freeness . . .	51

Chapter 1

Introduction

In this chapter, the concept of *fair exchange* is briefly introduced. After reviewing some related work on fair exchange protocols, we point out the challenging issues and our aims on this topic, then summarise our contributions and the organization of the thesis.

1.1 Fair Exchange

With the pervasive applications of the Internet in the modern world, the traditional ways of information communication have been changed fundamentally. More and more people prefer electronic information to paper-based documents. Digital information has numerous advantages in transmission, preservation and searching. However, when data flow in this vast, free but uncontrolled cyberworld, security, privacy and authentication of digital information have gradually become very important issues. Hence we need to study how to address these issues by designing secure and efficient cryptographic tools.

As e-commerce is getting more and more popular and important in business, it is desirable that two parties can fairly exchange their digital items via the Internet. In the real world, such an exchange could be very simple, that is, both parties make an appointment at some time and place, and get together to do the exchange face-to-face. Every party can be considered to obtain what he/she wants *simultaneously*. However, in computer networks, such *simultaneity* does not exist. That is because the rate of data transmission in networks depends on many complex and uncertain factors, such as network bandwidth, hardware condition, the physical access location of computers and even bad weather. It is impractical for any two parties in the Internet to simultaneously obtain each other's item in a normal way. That means, there always exists a party who can first obtain the other's digital item. A dishonest

party may be therefore benefited from this advantage, which may badly damage the other's interests. In order to provide '*simultaneity*' in networks, fair exchange protocols are designed based on many different technologies. In fact, these protocols cannot truly make data transmission simultaneously. The solution of the problem essentially focuses on the concept of *fairness*, i.e., at the end of the protocol either both parties have each other's item or none of them does. Such *fairness* should always hold even though some parties cheating.

A fair exchange protocol basically has at least the properties: *fairness* and *non-repudiation*. *Fairness* ensures that, if an honest party does not obtain a valid signature of the other party at the end of a fair exchange protocol, the other party cannot get that either, that is, either both two parties get each other's valid signature, or neither of them gets anything valuable. *Non-repudiation* guarantees that any party in a fair exchange protocol cannot repudiate or refute a valid signature after the protocol executed successfully. Due to these properties, in practical applications, fair exchange protocols are widely used in different but relevant fields such as contract signing protocols [BOGMR90, GJM99, BWZZ04], non-repudiation protocols [KMZ02, GRV03], e-payment system [BF98, PCS03] and certified e-mails [KM01, ANR02].

In early studies, the most direct method to achieve fair exchange in networks is gradual exchange protocols [Blu83, EGL85]. In such a protocol, both parties gradually exchange their digital items bit by bit over many rounds of exchange until every party obtains the whole expected items. Because only a little amount of information is transmitted in each round. The expected digital items for each party can be considered to be revealed approximately at the same time, which guarantees *fairness*. However, gradual exchange protocol can hardly be realised in practice. That is because, the security of these protocols depends on the equal computing resources for both parties, but such absolute equivalence in networks can not exist in most applications. Therefore, it is difficult to achieve provable *fairness* in such a protocol. Moreover, gradual exchange protocols need to consume a vast amount of network resources in gradual exchanges, which makes the protocols highly inefficient and expensive.

To overcome these shortcomings, a fair exchange protocol based on an on-line trusted third party (TTP) was always designed in the following studies. In such a protocol [DGLW96, ZG96], all the parties' digital signatures are sent to the TTP, which checks the validity of these signatures. If all the signatures are correctly

prepared, the TTP sends each signature to its corresponding receiver. Because the TTP is involved in every step of exchanges, a fair exchange protocol based on an on-line TTP is easily implemented. However, in practice, the on-line TTP may be inefficient and insecure due to its heavy involvement in the protocol. In addition, such a TTP is required to be perfectly trusted, which may lead some unexpected security risks. Comparing with the fair exchange protocols based on an on-line TTP, the protocols based on an off-line TTP are more practical, efficient and secure due to the property *optimism*. To reduce the load of the TTP, Asokan, Schunter and Waidner first proposed *optimistic* fair exchange (OFE) in [ASW97]. In an OFE protocol [ASW98, DLY07, HYWS08b], there is an off-line TTP, called *arbitrator*, who acts as a judge to settle the possible dispute between two parties and should only be involved when the protocol does not run correctly (e.g. some parties cheating or communication channel interrupted).

1.2 Related Work

In cryptography, the notion of *fair exchange* is generally referred to a fair exchange protocol for some kinds of digital signatures. In the recent studies, an optimistic fair exchange protocol typically comprises three steps. In the first step, Alice, who first starts the OFE protocol, sends a partial signature generated via an arbitrator's public key, which shows she has committed to send her full signature to the verifier Bob later if the protocol is executed correctly, together with an interactive or non-interactive zero-knowledge proof showing that the partial signature indeed corresponds to her full signature and can be recovered by the arbitrator. If Alice's partial signature is correctly verified, Bob sends his full signature to Alice in the second step. After checking the validity of Bob's full signature, Alice sends her full signature back to Bob in the third step. If Alice refuses to reveal her full signature to Bob or sends an invalid full signature, Bob can ask the arbitrator to convert Alice's partial signature into her full signature. And in case Bob's full signature is invalid, Alice does not need to worry about her giving away partial signature since, without the arbitrator's help, Bob cannot extract the full signature from the corresponding partial signature.

A conventional way to produce such a partial signature is *verifiably encrypted signature* (VES) which was first formalised by Boneh *et al.* in [BGLS03]. A VES usually consists of at least two parts: one is a conventional signature encrypted

under a TTP's public key; the other is a zero-knowledge proof for verifiers to check whether the encrypted signature corresponds to the conventional signature. In the OFE protocols based on VES [ASW98, CD00], the TTP in VES can serve as the arbitrator in OFE. If no misbehavior happens, the TTP does not involve in the protocol. However, if Bob does not receive the valid full signature from Alice at the end of the protocol, he can ask the arbitrator to decrypt Alice's partial signature in the dispute resolution protocol. The main drawback of this approach is that generating a VES from an ordinary signature is difficult for many signature schemes since the verifiable zero-knowledge proof, which shows the validity of the encryption, is not always achieved efficiently. In most cases, the construction of a VES has to employ the signature schemes which have some specific mathematical structure in order to give an efficient proof. In [Ate04], Ateniese proposed a method to construct verifiably encrypted signatures from some popular signature schemes, such as RSA [RSA78], Cramer-Shoup [CS98], Schnorr [Sch89] and other discrete-log-type signatures. The basic cryptographic tool in this method is to prove the equality of discrete logarithms in one or different groups, that is, given $y_1 = g_1^x$ and $y_2 = g_2^x$, to prove $\log_{g_1}^{y_1} = \log_{g_2}^{y_2}$ without revealing any valuable information about x . However, such a proof can only efficiently works in the signature schemes with simple structure. For the complex signature schemes of high level security, the computation and communication overheads of the proof are still overwhelming.

Another way to construct OFE protocols is based on *sequential two-party multisignatures*. In a multisignature scheme [Boy89, MOR01], multiple signers are able to sign a single message using their own private keys, called *partial private keys*, together and generate many different *partial signatures* of their own. All these partial private keys are used to compute a *joint private key* with some algebraic relation. A multisignature, which is composed of all these partial signatures, should only be verified by the corresponding *joint public key* of the joint private key. In [PCS03], based on a RSA-based multisignature scheme, Park, Chong and Siegel proposed an OFE protocol. In Park *et al.*'s protocol, the signer Alice arbitrarily splits her RSA-based private key into two partial private keys, and uses these two keys to compute a multisignature as the full signature of OFE. Before executing the exchange protocol, Alice needs to send one of these two partial private keys to the arbitrator in order to get a voucher which convinces the verifier Bob that the arbitrator has the capability to compute the same multisignature with this partial private key. Then Alice signs the message with the other partial private key and sends the verifier Bob

this signature as the partial signature of OFE. After receiving Bob’s full signature, Alice reveals her full multisignature to Bob. If Alice refuses to do that, Bob can ask the arbitrator to recover the multisignature via the partial private key received for Alice. Park *et al.*’s protocol is fair, optimistic, but insecure for a malicious arbitrator since there exists an efficient algorithm for the arbitrator to compute Alice’s full private key from the partial private key with some related information. Therefore, an adversarial arbitrator can easily break *security against the arbitrator* in this OFE protocol, i.e., the arbitrator can forge a valid full signature of the signer via the signer’s private key. To solve this problem, in [DR03], Dodis and Reyzin followed Park *et al.*’s idea to propose an efficient OFE scheme based on Boldyreva’s non-interactive two-signature [Bol03], and the scheme can be formally proven secure in the random oracle model.

In [ASW98], Asokan, Shoup and Waidner first formalised the notion of *optimistic fair exchange* and proposed a concrete protocol which not only satisfies *optimism* but also *timely-termination* and *accountability*. *Timely-termination* means, during the execution of the protocol, if a party is in a unresponsive situation for a long time, the other party can ask the arbitrator to stop the protocol without the permission of the unresponsive party, i.e., any party in this protocol can always achieve a timely and fair termination. *Accountability* guarantees that, if the arbitrator cheats, its misbehavior can be detected and proved. The aim of this property is to supervise and enforce the arbitrator to behave honestly.

In [DR03], Dodis and Reyzin proposed the notion of *verifiably committed signatures* which can be used to formalise non-interactive optimistic fair exchange. According to the three different roles in OFE, the security model of verifiably committed signatures is defined in three aspects, i.e. *security against the signer, the verifier and the arbitrator*. *Security against the signer* requires that any signer cannot generate a partial signature, which can be accepted by the honest verifier, but cannot be converted to a valid full signature by an honest arbitrator. *Security against the verifier* guarantees that any verifier cannot recover a valid full signature from the corresponding partial signature without assistance of the signer or the arbitrator. *Security against the arbitrator* means that any arbitrator cannot forge a valid full signature without asking the signer to issue one. Dodis and Reyzin’s security model generalises the fundamental security notions in OFE, which makes the security of OFE protocols can be rigorously proven in a formal model. In addition, this security model can be easily updated for other security notions, such as

abuse-freeness, security in multi-user setting and chosen-key model.

Note that the security model of Dodis and Reyzin’s OFE is only defined in the single-user setting, i.e., an OFE protocol only has one single signer, one single verifier and one arbitrator. In [DLY07], Dodis, Lee and Yum updated the previous model and first formally defined OFE in the multi-user setting. Different from OFE in the single-user setting, OFE in the multi-user setting has many distinct signers and verifiers so that a malicious party can attack the OFE protocol by colluding with other parties. Although the security of public key encryption and digital signature schemes in the single-user setting can imply the security in the multi-user setting, Dodis *et al.* proved such implication does not always exist in OFE and presented a counterexample. Then they defined a formal security model of OFE in the multi-user setting and proposed a generic scheme which is setup-free [ZB06] (i.e., there is no initial-key-setup protocol between the signer and the arbitrator except necessary verification of their public keys’ certificates). Furthermore, they proved the abstract schemes of OFE based on *verifiably encrypted signatures* or *sequential two-party multisignatures* still secure in the multi-user setting if the underlying primitives satisfy some security properties, such as *existential unforgeability under adaptive chosen message attacks* and *indistinguishability against adaptive chosen ciphertext attacks*.

In [HYWS08b], Huang *et al.* updated Dodis, Lee and Yum’s security model and considered OFE not only in the multi-user setting but also in the chosen-key model. Before this study, all the OFE protocols are only proven secure in the certified-key model, in which the adversary must prove that he/she knows the corresponding secret key before using a public key. Therefore, the adversary can only query the oracles using certified keys. However, in the chosen-key model, without this restriction the adversary can arbitrarily select public keys without knowing the corresponding secret keys and query the oracles using any keys he/she chooses. Huang *et al.* presented an attack against **WVES**-based OFE [LOS⁺06] under the chosen-key model to break *security against verifiers*. This attack proves that security in the certified-key model might not imply that in the chosen-key model. After formally defining OFE in the multi-user setting and chosen-key model, Huang *et al.* constructed a generic OFE scheme which can be proven secure without random oracles. A novel feature of this construction is that the generation of a partial signature does not require the arbitrator’s public key. A conventional signature is used as a partial signature, and a full signature consists of the conventional signature and a ring

signature generated under the signer and the arbitrator's keys. In the resolution protocol, the arbitrator can produce such a ring signature using its own private key on behalf of the signer.

1.3 Challenging Issues

In previous studies, most of fair exchange protocols do not satisfy the property *abuse-freeness* which is an important security requirement for some applications (e.g. contract signing). Intuitively, a fair exchange protocol is said to be *abuse-free*, if any intermediate outputs in such a protocol cannot be accepted as the valid evidence showing that some party has committed to sign the message before the protocol successfully runs. In contrast, an OFE protocol without *abuse-freeness* might benefit a dishonest party via revealing the public verifiable interim outputs to an outsider of the protocol. For example, a customer Alice wants to buy some products from a merchant Carl, but the price of Carl's products is a little higher than Alice's expectation. Another merchant Bob has the same type of products with a lower price but the quality of the products does not meet Alice's requirement. To get more advantages, Alice first pretends to sign a contract with Bob. After getting the valid verifiable interim results, she terminates the contract signing protocol with Bob, and shows the results to Carl for reducing his price. Due to the risk of losing customers, Carl may make concessions to Alice about the price. Obviously the contract signing protocol between Alice and Bob is not favorable to Bob as he is just exploited by Alice for getting a better contract.

To solve this dilemma, Garay, Jakobsson and MacKenzie [GJM99] constructed the first *abuse-free optimistic contract signing protocol* from designated verifier proofs [SKM03, HSMW06], in which only a designated verifier can be convinced. Then Wang [Wan10] proposed an abuse-free contract-signing protocol based on the RSA signature. In Wang's scheme, an interactive protocol based on trapdoor commitment schemes is employed to prove the validity of a RSA-based undeniable signature [GRK00]. The main shortcoming of these two studies of *abuse-freeness* is no precise and formal security model is given on this topic, therefore, both the protocols are not formally proven secure. In [HYWS08a], Huang *et al.* proposed the notion of *ambiguous optimistic fair exchange*, in which an outsider cannot tell whether a signer or a verifier has issued a given partial signature, i.e., the partial signature is indistinguishable with respect to its issuer so that the verifier cannot convince

an outsider about the authorship of the partial signature. However, Huang *et al.*'s scheme is not *perfectly abuse-free* since an outsider can be certain of that the real producer of the partial signature must be either the signer or the verifier, which makes the outsider convinced by a dishonest verifier with probability at least $1/2$.

In some cases, the anonymity of participants might be important in order to protect participants' privacy. For example, the personal preferences of negotiators in business contract signing usually influence the terms of the final agreement. If a trading company A has the contract signing records of an employee as a negotiator in another company B which is a potential trade cooperator of A , A can use these records to deduce the negotiator's trading habits, by which Company A might get advantages in the future contract negotiation with Company B . Hence it is desirable that the employees who have the right to independently sign a contract on behalf of their own company can sign contracts anonymously, which will prevent other companies from knowing the signer's trading habits. To this end, ring signatures are the good primitive to provide the property *signer ambiguity* [AOS02]. Informally, in a ring signature scheme, the public keys of a group of users are collected spontaneously to form a public-key list. When a signer signs a message on behalf of such a ring, he uses the public-key list and adds his own private key as a glue value to issue a ring signature. A verifier cannot tell who the real signer is, because the ring signature is validated using all the public keys of the ring without revealing any information about who produced it. On this issue, none of the previous OFE solutions formally studies optimistic fair exchange of ring signatures (OFERS) and especially analyses its security in a high level security model. In addition, no efficient OFE protocol is known to achieve abuse-free OFERS with provable security.

1.4 Contributions

For the sake of privacy in fair exchange, we aim to construct an efficient optimistic fair exchange protocol of ring signatures, which not only achieves high level security but also can be easily converted to an efficient abuse-free OFE protocol. In this thesis, we make the following contributions.

- We propose the notion of *optimistic fair exchange of ring signatures* (OFERS) which allows two members from two different groups exchange their ring signatures in an equitable way with ambiguous signers for the other group.

Then we formalize optimistic fair exchange of ring signatures and propose a strong security model in the multi-user setting under adaptive chosen message, chosen-key, and chosen public-key attacks.

- By combining the technologies of ring signatures, zero-knowledge proof and public-key encryption, we construct the first efficient and concrete OFERS scheme from verifiably encrypted ring signatures (VERS) based on Abe *et al.*'s all discrete-log ring signature scheme, and formally prove that the proposed OFERS scheme is *secure against signers, verifiers and the arbitrator* and perfectly inherits the property *signer ambiguity* of ring signatures.
- We propose the notion of *abuse-free optimistic fair exchange of ring signatures* (AOFERS), in which any interim output in an OFERS protocol cannot be accepted as the valid evidence showing that certain participant has committed to sign the message before the protocol is executed successfully. We propose the first formal security definition and model of AOFERS and construct an efficient concrete AOFERS scheme based on Abe *et al.*'s ring signature scheme in all discrete-log case via updating our construction of OFERS.
- We find a solution to improve the proposed AOFERS scheme such that not only the all discrete-log ring signature scheme but also the generic scheme of Abe *et al.*'s ring signatures can be efficiently employed in our AOFERS scheme, and prove that the proposed AOFERS schemes are perfectly abuse-free in our security model.

1.5 Organization of the Thesis

The rest of the thesis is organized as follows:

In Chapter 2, some background knowledge of cryptography is introduced to help readers understand the thesis. After presenting some basic preliminaries including cryptographic hash functions, standard model, random oracle model and some complexity assumptions, we introduce the underlying knowledge of the technologies which are applied in optimistic fair exchange of ring signatures, such as zero-knowledge proof, public-key encryption, ring signatures, verifiably encrypted signatures and commitment schemes.

In Chapter 3, we present the first solution of OFERS by first formally defining its security model in the multi-user setting under adaptive chosen message, chosen-key, and chosen public-key attacks. Then we present a concrete OFERS scheme which is constructed from verifiably encrypted ring signatures (VERS), and formally prove the proposed scheme secure in our security model. At last, we point out some further improvements of our OFERS scheme.

In Chapter 4, we improve the proposed OFERS scheme in Chapter 3 so as to meet the property *abuse-freeness*. We first propose the formal definition and security model of abuse-free optimistic fair exchange of ring signatures (AOFERS), and then construct a concrete AOFERS scheme by updating our OFERS scheme based on Abe *et al.*'s all discrete-log ring signature scheme in Chapter 3. Moreover, we extend the proposed AOFERS scheme such that the generic scheme of Abe *et al.*'s ring signatures can also be efficiently employed in our construction. Lastly, we prove that the proposed schemes is perfectly abuse-free.

Finally, the thesis is concluded in Chapter 5.

Chapter 2

Background

In this chapter, some background knowledge of cryptography is briefly introduced to help readers get a better understanding of this thesis.

2.1 Cryptographic Hash Functions

In cryptography, a hash function is a mathematical function which takes an arbitrary length block of data as input and outputs a fixed-size bit string called a hash value. A hash function is typically described by a mapping $H : \{0, 1\}^* \rightarrow \{0, 1\}^n$, where n denotes the size of a hash value. There are many practical applications of cryptographic hash functions in digital signatures, message authentication codes and so on, where the most fundamental application of secure hash functions is verification of message integrity. In most digital signature schemes, for the sake of security and efficiency, only the hash value of a given message is signed instead of the original message.

In [RS04], Rogaway and Shrimpton proposed the security properties for cryptographic hash functions, i.e., *preimage resistance*, *second-preimage resistance* and *collision resistance*, and discussed the relations among these properties. Intuitively, the security properties of a hash function can be described in the following way:

- **Preimage resistance:** For any hash value h , it should be computationally difficult to find an input message m such that $H(m) = h$, that is, there does not exist an efficient inverse function of H to compute the pre-image of h . This property is usually related to the concept of a one-way function.
- **Second-preimage resistance:** For any input message m_1 , it should be computationally difficult to find a distinct input message m_2 such that $H(m_1) = H(m_2)$. This property is sometimes called *weak collision resistance* [NY89].

- **Collision resistance:** It should be computationally difficult to find two distinct input messages m_1 and m_2 such that $H(m_1) = H(m_2)$. This property is sometimes called *strong collision resistance* [Dam87].

There are many practical hash functions [Riv92, KR00, AdM04] applied in different fields of cryptography. One of the most widely used hash functions is MD5 [Riv92], however, which has been considered cryptographically broken in recent study. A more secure hash function MD6 [RAB⁺08] has been therefore proposed to replace MD5 in 2008.

2.2 Standard Model and Random Oracle Model

In cryptography, the security of cryptographic primitives (e.g. digital signature or encryption scheme) usually depends on some complexity assumptions, such as discrete logarithm assumption and strong RSA assumption. A complexity assumption usually describes a problem which is difficult to be solved in polynomial time. If a cryptographic scheme is proven secure only based on such complexity assumptions, the scheme is said to be secure in the standard model.

Standard model is a strong model in security proofs of cryptographic schemes. However, some proofs are very hard to be found in the standard model and can only be achieved in some idealized models [CPS08]. To this end, Bellare and Rogaway introduced random oracle model for security proofs in [BR93]. In random oracle model, a cryptographic hash function is considered as a random oracle which answers each query with a random response from a particular domain, and returns the same response for the same query. In another way, if a cryptographic scheme is proven secure in random oracle model, the adversary is not only restricted by his computational power but also cannot require impossible abilities from the oracle except common queries. Random oracle model is weaker than standard model in security proofs since there does not exist a real hash function which can implement a true random oracle in practice. Hence some cryptographic schemes, which are proven secure in random oracle model, might be insecure in the standard model.

2.3 Computational Complexity Assumptions

In cryptography, a computational complexity assumption assumes a *hard* problem which cannot be solved in polynomial time [Mao03]. In practice, most cryptographic primitives with provable security depends on some computational complexity assumptions since the absolute security in information theory cannot be achieved in many cases. Hence, in these cryptographic schemes, the adversary's computational ability must be restricted in polynomial time and cannot solve the *hard* problem out of this limitation as the real attackers do in practice. In this section, we present some widely used computational complexity assumptions which are referred in this thesis.

2.3.1 Discrete Logarithm Assumption

The discrete logarithm assumption is the most fundamental computational complexity assumption in cryptography. The concept of discrete logarithms is proposed in abstract algebra as analogues of ordinary logarithms in real or complex numbers. Roughly, the discrete logarithm assumption is that, given a finite cyclic group \mathbb{G} and its elements g and h , it is difficult to find a solution x in \mathbb{G} such that $h = g^x$, where x is called a discrete logarithm to the base g of h in \mathbb{G} . Formally, the discrete logarithm assumption is defined by Mao [Mao03] as follow:

Definition 2.1 Let \mathbb{F}_q^* be a finite cycle group with the order q , g is a generator element of \mathbb{F}_q^* and $h \in \mathbb{F}_q^*$. a is the unique integer $a < q$ such that $h = g^a$. A discrete logarithm solver is a probabilistic polynomial-time (PPT) algorithm \mathcal{A} with an advantage $\epsilon = \Pr[a \leftarrow \mathcal{A}(g, h)] > 0$. Let \mathcal{IG} be an instance generator that on input 1^k , runs in time polynomial in k , and outputs \mathbb{F}_q^*, g and h . We say that \mathcal{IG} satisfies the discrete logarithm assumption if there exists no discrete logarithm solver for $\mathcal{IG}(1^k)$ with an advantage $\epsilon > 0$ non-negligible in k for all sufficiently large k .

2.3.2 Strong RSA Assumption

The strong RSA assumption is usually used in security proofs of cryptographic primitives such as [BP97, FO97]. Based on the strong RSA assumption, some digital signature schemes, such as [GHR99, CS00], can be proven secure without random oracles. Informally, the strong RSA assumption assumes a hard problem that, given

a RSA modulus n and a random integer $y \in \mathbb{Z}_n^*$, it is hard to find integers x and e such that $y = x^e \bmod n$ and $1 < e \leq n$. Formally, the strong RSA assumption is defined as follow [BP97]:

Definition 2.2 Let $RSA_{Mod}()$ denote the set of RSA moduli of length k . For any probabilistic polynomial-time algorithm \mathcal{A} and any sufficiently large integer k ,

$$\Pr = [(x, e) \leftarrow \mathcal{A}(n, y) \wedge y = x^e \bmod n \wedge n \in_R RSA_{Mod}(k) \\ \wedge y \in_R \mathbb{Z}_n^* \wedge 1 < e \leq n] = \text{negl}(k)$$

In this thesis, Camenisch and Michels' zero-knowledge proof [CM99] is employed for verifying the validity of partial ring signatures in our fair exchange schemes. The security of the zero-knowledge proof is based on the strong RSA assumption.

2.3.3 Decisional Composite Residuosity Assumption

The decisional composite residuosity assumption (DCRA) is first proposed by Pascal Paillier [Pai99] for the proof of Paillier cryptosystem. The definition of DCRA is given in [Pai99] as follow:

Definition 2.3 A number z is said to be a n -th residue modulo n^2 if there exists a number $y \in \mathbb{Z}_{n^2}^*$ such that $z = y^n \bmod n^2$. $\mathbf{CR}[n]$ denotes distinguishing n -th residues from non n -th residues. There exists no polynomial time distinguisher for n -th residues modulo n^2 , i.e., $\mathbf{CR}[n]$ is intractable.

In this thesis, in order to provide the high level security which protects against adaptive chosen-ciphertext attacks (CCA2), we choose Camenisch and Shoup's public key encryption scheme in our fair exchange schemes, which is an adaptation of Paillier cryptosystem. Like Paillier cryptosystem, the security of Camenisch and Shoup's encryption scheme also relies on the decisional composite residuosity assumption.

2.4 Zero-knowledge Proof

The concept of zero-knowledge proof is first proposed by Goldwasser *et al.* [GMR89]. A zero-knowledge proof is a proof of the truth of a statement, which allows a prover

to convince a verifier without revealing anything else. By this proof, the verifier is successfully convinced that the prover does know something, but no more information is provided. Typically, it is used for one party to convince another party that the first party knows some secret but does not want the second party to learn anything about this secret. For this reason, zero-knowledge proofs are widely used in many fields of cryptography, such as fair exchange protocols in e-commerce [PCS03], authentication systems [BM93] and secure multiparty computation [CDD⁺99, IKOS09].

Generally, an interactive zero-knowledge proof system must satisfy three properties: *correctness*, *soundness* and *zero-knowledge*. Following Bellare and Goldreich's definitions in [BG06], we describe a zero-knowledge proof system in the following way:

Definition 2.4 An interactive zero-knowledge proof system (\mathbf{P}, \mathbf{V}) for a set S is a two party game between a probabilistic polynomial-time (PPT) prover P and a PPT verifier V , and satisfies the following properties:

- **Correctness:** For any $x \in S$, the verifier V always accepts after the proof with the prover P on common input x , i.e., if a statement is true, the prover always succeeds to convince the verifier via such a proof.
- **Soundness:** For any $x \notin S$, the verifier V rejects with probability at least $1/2$ after the proof with the prover P on common input x , i.e., if a statement is not true, the verifier V can not be convinced by the prover P except with some small probability.
- **Zero-knowledge:** There exists an expected probabilistic polynomial-time algorithm A^* . For any $x \in S$, $(\mathbf{P}, \mathbf{V})(x)$ and $A^*(x)$ are computationally indistinguishable, that is, the distribution of the outputs of $(\mathbf{P}, \mathbf{V})(x)$ and $A^*(x)$ are identical. If a statement is true, the verifier cannot learn anything other than the fact of the statement.

2.4.1 Non-interactive Zero-knowledge Proof

In [BFM88], Blum, Feldman and Micali proposed a method to achieve non-interactive zero-knowledge (NIZK) proofs without interaction between provers and verifiers. In this method, a random challenge bits shared between a prover and a verifier is used

as a common reference string to provide computational zero-knowledge. Such a common reference string can be considered to be generated by a third party which is mutually trusted by provers and verifiers. In practice, the common reference string is usually produced by a trusted setup algorithm which is agreed by both the prover and the verifier. A non-interactive zero-knowledge (NIZK) proof system can be formalized as follow [GS08]:

Definition 2.5: A non-interactive zero-knowledge proof system for a language L with a relation R comprises a prover P , a verifier V and a common reference string (CRS) generation algorithm K outputting a CRS σ . \approx denotes being equal except negligible probability. To prove $x \in L$ with witness w , i.e. $(x, w) \in R$, the prover P generates a proof $\pi \leftarrow P(\sigma, x, w)$ and sends it to the verifier. Then on input (σ, x, π) , the verifier V checks whether the proof is valid, if yes, outputs 1, otherwise outputs 0. This proof system satisfies the following conditions:

- Completeness: For any adversary \mathcal{A} ,

$$\Pr[\sigma \leftarrow K(1^k); (x, w) \leftarrow \mathcal{A}(\sigma); \pi \leftarrow P(\sigma, x, w) : V(\sigma, x, \pi) = 1 \text{ if } (x, w) \in R] \approx 1.$$

- Soundness: For any adversary \mathcal{A} ,

$$\Pr[\sigma \leftarrow K(1^k); (x, \pi) \leftarrow \mathcal{A}(\sigma) : V(\sigma, x, \pi) = 0 \text{ if } x \notin L] \approx 1.$$

- Zero-knowledge: There exists a simulator $Sim = (Sim_1, Sim_2)$. On input 1^k , Sim_1 outputs a simulated CRS and its corresponding trapdoor τ , where the simulated CRS is indistinguishable from a real one. Sim_2 takes as input (σ, τ, x) and outputs a simulated proof which is indistinguishable from that generated by a real prover. For any adversary \mathcal{A} ,

$$\Pr[\sigma \leftarrow K(1^k) : \mathcal{A}^{P(\sigma, \dots)}(\sigma) = 1] \approx \Pr[(\sigma, \tau) \leftarrow Sim_1(1^k) : \mathcal{A}^{Sim(\sigma, \tau, \dots)} = 1],$$

where $Sim(\sigma, \tau, x, w) = Sim_2(\sigma, \tau, x)$.

In our optimistic fair exchange of ring signatures scheme (Chapter 3), the verification of a partial ring signature is based on a non-interactive zero-knowledge proof which is updated from Camenisch and Michels' zero-knowledge proof [CM99]. In our abuse-free OFERS scheme (Chapter 4), the non-interactive proof is modified again to a new interactive zero-knowledge proof in order to achieve the property *abuse-freeness*.

2.5 Public-key Encryption

In symmetric cryptosystems, a secret key is involved in both encryption and decryption algorithms, which leads a difficulty in the distribution of keys, i.e., how to guarantee the security of such a secret key during the exchange of encrypted messages. To solve this dilemma, the concept of public-key cryptosystems was first proposed by Diffie and Hellman [DH76a] in 1976. In public-key cryptosystems, the encryption key and the decryption key are distinct. When a user generates a public-private key-pair, he/she only keeps the private key as a secret and sends the public key to an authentication server which is in charge of the key distribution. Typically, the construction of a public-key encryption scheme is based on a *one-way trapdoor function* which can be defined as follow [Mao03]:

Definition 2.6: A one-way trapdoor function denoted by $f_t(x) : \mathcal{D} \rightarrow \mathcal{R}$ is a one-way function, i.e., it is easy to evaluate for all $x \in \mathcal{D}$ and difficult to invert for almost all values in \mathcal{R} . However, if the trapdoor information t is used, for all values $y \in \mathcal{R}$, it is easy to compute $x \in \mathcal{D}$ such that $y = f_t(x)$.

The security of public-key cryptosystems is usually based on some computational complexity assumptions, i.e., the computational capabilities of the attacker for such a cryptosystem is limited in polynomial time. However, with the development of high-performance computers, especially quantum computers, some of the most popular public-key cryptosystems, such as RSA [RSA78], ElGamal [ElG85] and Paillier [Pai99] cryptosystems, may not be secure anymore and replaced by some new cryptosystems with high level security.

2.5.1 Security Notions

In order to analyze the security of an encryption scheme, the possible aims and attack models should be first defined. The security notions of public-key encryption schemes are therefore categorized according to this way.

The attack models against an encryption scheme usually fall into three patterns in order of increasing capability of attackers: chosen-plaintext attack (CPA), non-adaptive chosen-ciphertext attack (CCA1) [NY90] and adaptive chosen-ciphertext attack (CCA2) [RS91]. Chosen-plaintext attacks are the most fundamental attack

model, in which the adversary can arbitrarily choose plaintexts and get its corresponding ciphertexts. And in CCA1, the adversary not only has the public key but also is allowed to access a decryption oracle under the corresponding private key to decrypt any ciphertext of his choice. However, once the challenge ciphertext is given to the adversary, the decryption oracle cannot be accessed anymore. Moreover, in CCA2, the adversary's power is further strengthened. He/She can access the decryption oracle all the time even after obtaining the challenge ciphertext. The only limitation for the adversary is that he/she cannot ask the decryption oracle to decrypt the challenge ciphertext.

In addition, there are another two important security properties desired in encryption schemes: *indistinguishability* [GM84] and *non-malleability* [DDN91]. Indistinguishability means that, given two messages x_0, x_1 of equal length and a ciphertext c_b of either x_0 or x_1 , the adversary cannot indicate the plaintext of c_b from x_0 and x_1 , that is, the adversary cannot learn anything about the plaintext from the corresponding ciphertext. Non-malleability means, given a ciphertext c , the adversary cannot output a distinct ciphertext c' such that the plaintexts x_0 and x_1 of these two ciphertexts have some relation in mathematical structure.

The three attack models and the two security properties can be formed to six security notions, i.e., IND-CPA, IND-CCA1, IND-CCA2, NM-CPA, NM-CCA1 and NM-CCA2. In [BDPR98], Bellare *et al.* studied the relations among these security notions for public-key encryption schemes. Their achievement is summarized in the following figure:

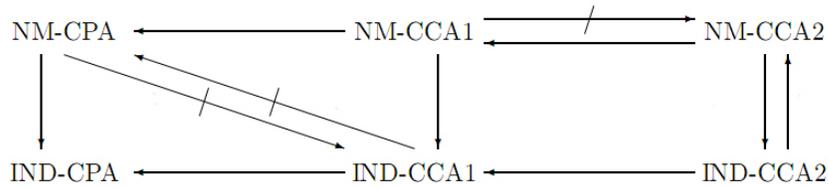


Figure 2.1: [BDPR98] The relations of security notions for public-key encryption schemes

In Figure 2.1, for any security notion $A, B \in \{\text{IND-CPA}, \text{IND-CCA1}, \text{IND-CCA2}, \text{NM-CPA}, \text{NM-CCA1}, \text{NM-CCA2}\}$, $A \rightarrow B$ means, if an encryption scheme satisfies the notion A , then it also provably satisfies the notion B , and $A \not\rightarrow B$ means, if an encryption scheme satisfies the notion A but it is proved not to satisfy

the notion B . From this figure, we can see that the notion of *non-malleability* always implies the notion of *indistinguishability*, but the reverse does not exist except the CCA2 case, in which *non-malleability* is equivalent to *indistinguishability*. And CCA2 is the strongest notion since the power of adversaries is the least limited in CCA2.

In this thesis, in order to achieve the high level security, we choose Camenisch and Shoup’s encryption scheme [CS03], which is proven secure against adaptive chosen-ciphertext attacks, as our encryption scheme for the generation of verifiably encrypted ring signatures.

2.6 Digital Signatures

The concept of digital signatures was first described by Diffie and Hellman in [DH76b]. A digital signature scheme [Sch89, CA89, BLS04] is a mathematical method for proving the authorship of a digital message by a one-way trapdoor function. Intuitively, a digital signature scheme is a variant of public-key encryption schemes. When a user of a public-private key-pair issues a digital signature on a message, he/she ‘encrypts’ the message using his/her private key, and the ‘encrypted’ message can be ‘decrypted’ by anyone using the user’s public key. Because only the user has the right to access his/her private key, such an ‘encrypted message’ can only be created by the user as his/her digital signature.

Definition 2.7: A conventional digital signature scheme consists of three probabilistic polynomial-time algorithms.

- **KeyGen:** On input a security parameter, the key generation algorithm outputs a verifying-signing key-pair (vk, sk) for a user.
- **Sign:** On input a digital message m and a signing key sk , the signing algorithm outputs a signature σ on m .
- **Veri:** On input a signature σ , a message m and a verifying key vk , the verification algorithm outputs 1 or 0, which means whether σ is a valid signature on m under vk , or not respectively.

In this thesis, we consider the digital signature schemes referred in our fair exchange protocols in a strong security notion, i.e. existential unforgeability against

adaptive chosen-message attacks [GMR88], which means an attacker can obtain the signature of any message of his/her choice, but the probability of forging a signature for a challenge message without the signing oracle is negligible.

2.6.1 Ring Signatures

A ring signature [AOS02, LSW06, CWLY06], sometimes referred as *1-out-of- n* signature, is a special digital signature with an additional property *signer ambiguity*, which was first introduced by Rivest, Shamir and Tauman in [RST01]. Informally, in a ring signature scheme, the public keys of a group of users are collected spontaneously to form a public-key list. When a signer signs a message on behalf of such a ring, he uses the public-key list and adds his own private key as a glue value to issue a ring signature. A verifier cannot tell who the real signer is, because the ring signature is validated using all the public keys of the ring without revealing any information about who produced it. The definition of *1-out-of- n* signature can be described as follow [AOS02]:

Definition 2.8: A *1-out-of- n* signature scheme $\mathcal{S}^{1,n}$ consists of three probabilistic polynomial-time algorithms.

- $\mathcal{G}^{1,n}(1^k)$: On input a security parameter k , the key generation algorithm outputs a public-private key-pair (vk, sk) for each user.
- $\mathcal{S}_{sk}^{1,n}$: On input a message m and a public-key list L including the public key which corresponds to sk , the signing algorithm outputs a signature σ on m under L .
- $\mathcal{V}_L^{1,n}$: On input a signature σ , a message m and a public-key list L , the verification algorithm outputs 1 or 0, which respectively means to accept or reject σ as a valid signature on m under L .

Definition 2.9 (Signer Ambiguity): $L = \{vk_1, \dots, vk_n\}$ is a public-key list, in which each key is generated by $(vk_i, sk_i) \leftarrow \mathcal{G}^{1,n}(1^{k_i})$. A *1-out-of- n* signature scheme $\mathcal{S}^{1,n}$ is said to be *perfectly signer-ambiguous* if, for any L , any message m and any signature σ generated by $\sigma \leftarrow \mathcal{S}_{sk}^{1,n}(m, L)$ where $sk \in \{sk_1, \dots, sk_n\}$, given (L, m, σ) , any unbound adversary \mathcal{A} outputs i such that $sk = sk_i$ with exactly probability $1/|L|$.

In our optimistic fair exchange of ring signature schemes, we choose Abe *et al.*'s ring signature schemes which are proven *perfectly signer-ambiguous* and *existential unforgeability against adaptive chosen-message and chosen public-key attacks* [AOS02].

2.6.2 Verifiably Encrypted Signature

In practice, an effective way to construct optimistic fair exchange protocols is *verifiably encrypted signature* (VES) [ZSNS03, GS05, ZM07, ZWQ09, RSS10], which was formally defined by Boneh *et al.* [BGLS03]. A VES is an ordinary signature encrypted using the public key of a trusted third party, together with a verifiable proof showing the validity of the encryption. In OFE protocols, a VES is usually used as a partial signature that is the commitment for the corresponding full signature.

Definition 2.10: A verifiably encrypted signature scheme consists of seven probabilistic polynomial-time algorithms.

- **KeyGen, Signing, Verification:** These three algorithms are the same as that in conventional signature schemes.
- **AdjKeyGen:** On input a security parameter, the algorithm outputs a public-private key-pair (APK, ASK) for an adjudicator.
- **VESsig:** On input a message m , a private key sk and an adjudicator's public key APK , the algorithm outputs a VES θ on m under pk .
- **VESver:** On input a VES θ , a message m , a public key pk and an adjudicator's public key APK , the algorithm outputs 1 or 0, which respectively means to accept or reject θ as a valid VES on m under pk .
- **Adjudication:** On input an adjudicator's key-pair (APK, ASK) and a VES θ on a message m under a public key pk , the algorithm outputs a conventional signature σ on m under pk .

In an OFE protocol base on VES, Alice can first send a VES generated under a TTP's public key to Bob. Then, Bob is able to verify the validity of the VES together with a proof showing that Alice's signature encrypted in the VES can be

recovered by the TTP as the arbitrator, but cannot obtain the original signature from Alice unless Bob sends his own signature to Alice. After that, if Alice refuses to reveal Bob her signature, Bob can ask the TTP to decrypt Alice's VES and obtain her original signature.

2.7 Commitment Schemes

In cryptography, a commitment scheme is an important tool for designing cryptographic protocols. The notion of commitments schemes was first formally defined by Brassard, Chaum and Crépeau in [BCC88]. Informally speaking, when a two-party protocol runs between a sender and a receiver, a commitment scheme [BCC88, Ped91, ZCDM09] allows the sender to commit a value hidden in a commitment which will be opened by the sender revealing the hidden value and some related information later, so that the receiver can check whether the hidden value is committed correctly in advance. A commitment scheme satisfies two properties *hiding* and *binding*: *hiding* means no one can know what value is committed in a given commitment except its producer, and *binding* guarantees that, once a commitment has been made, the committed value in the given commitment cannot be changed by anyone.

2.7.1 Trapdoor Commitment Schemes

A trapdoor commitment scheme is a variant of common commitment schemes, which is mostly used for designing zero-knowledge proofs. Beside the properties *hiding* and *binding*, in a trapdoor commitment scheme [Gen04, MY04, SST05], there exists a secret trapdoor, by which a commitment can be opened ambiguously with different answers, but without the trapdoor, the property *binding* still holds. A trapdoor commitment scheme can be defined as follow:

Definition 2.11: A trapdoor commitment scheme consists of four probabilistic polynomial-time algorithms.

- **TCsetup:** On input a security parameter, the algorithm outputs a trapdoor td and its corresponding public key pk .

- **TCcomit**: On input a value m and a public key pk , the committing algorithm outputs a commitment σ for m with a random value r .
- **TCveri**: On input a value m , a public key pk , a trapdoor commitment σ and its related value r , the verification algorithm outputs 1 or 0, which means whether (m, r) is a valid answer to σ under pk , or not respectively.
- **TCsim**: On input a trapdoor td and a trapdoor commitment σ with a valid answer (m, r) , the simulation algorithm outputs a distinct answer (m', r') for σ , where $(m', r') \neq (m, r)$.

In our abuse-free optimistic fair exchange of ring signature scheme, to verify the validity of an encrypted ring signature, we use an interactive zero-knowledge proof based on trapdoor commitment schemes since such a proof is secure against the on-line attack for *abuse-freeness* introduced by Wang [Wan10] (see Chapter 4 for details).

Chapter 3

Optimistic Fair Exchange of Ring Signatures

In this chapter, the notion of *optimistic fair exchange of ring signatures* (OFERS) is first described in formal security model. After introducing the building blocks, a concrete OFERS scheme is presented based on the technology of verifiably encrypted ring signatures (VERS). And the formal proofs are given to show that the concrete scheme is secure in our security model.

3.1 Introduction

After the concept of optimistic fair exchange (OFE) was first proposed by Asokan *et al.* [ASW97] in 1997, fair exchange protocols are usually executed between individual parties. When two members on behalf of two different groups intend to fairly exchange their digital items in networks, the privacy of the participants should be carefully considered in some cases. Hence it is desirable that the parties in fair exchange of digital signatures, who have the right to sign a message on behalf of their own group, can sign messages anonymously. For this reason, we study optimistic fair exchange of ring signatures (OFERS), in which users in each ring can fairly exchange their ring signatures with ambiguous signers for the other ring. By inheriting the property *signer ambiguity* of ring signatures, the real signers in OFERS can be perfectly hidden in their own ring without revealing who signs the message. To the best of our knowledge, this is the first formal work on this topic to present a formal security model of OFERS and a concrete solution with provable security.

After introducing the building blocks in our construction of OFERS in Section 3.2, we first rigorously define the security model of OFERS in the multi-user setting under adaptive chosen message, chosen-key and chosen public-key attacks (Section 3.3). This is done by updating the formal models of OFE [DLY07, HYWS08b]

in the scenario of ring signatures. Then we present a concrete OFERS scheme (Section 3.4), which is constructed from verifiably encrypted ring signatures (VERS) based on Abe *et al.*'s all discrete-log ring signatures [AOS02] under an arbitrator's public key, together with a proof of knowledge showing the validity of the original ring signature's encryption. Theoretically, any CCA2-secure public key encryption scheme can be used as such a proof of knowledge always exists (but may be not efficient). To provide practicality and high efficiency, Camenisch and Shoup's CCA2-secure encryption scheme [CS03] is particularly selected in the proposed scheme. Then, we formally show that the proposed OFERS solution is provably secure in our security model (Section 3.5).

As the VES technique is employed, a notable feature of our scheme is that any holder (not necessarily the signer) of a valid ring signature can verifiably encrypt the ring signature to get a VERS without using any secret information from the signer. Due to this feature, our scheme not only preserves the property *signer ambiguity* [AOS02] of ring signatures, but also allows a signer to delegate a proxy (e.g. his/her secretary) to run OFERS after he/she produced a ring signature in advance. Moreover, the proposed OFERS scheme can be easily updated to an *abuse-free* OFERS scheme by applying an interactive zero-knowledge proof showing the validity of verifiably encrypted ring signatures. And not only the all discrete-log ring signature scheme but also Abe *et al.*'s generic ring signature scheme can be applied in the abuse-free OFERS scheme. The details of this part will be shown in Chapter 4.

3.2 Building Blocks

In this section, we introduce the concrete building blocks in our OFERS scheme.

3.2.1 Abe *et al.*'s Ring Signature Schemes

For the sake of simplicity and efficiency in mathematical structure, we choose Abe *et al.*'s ring signature schemes in our OFERS scheme. In [AOS02], Abe *et al.* proposed a generic scheme of ring signatures and some concrete examples which are proved unconditionally signer-ambiguous and existential unforgeability against adaptive chosen message and chosen public-key attacks.

Generic Scheme

Abe *et al.* first described two types of signature schemes, called *Hash-then-One-Way type* (type-H) and *Three-move type* (type-T), which can be used in their generic scheme of ring signatures. Type-H signature schemes, such as RSA signature, typically consist of a trapdoor one-way function F , its inverse function I and a hash function H . For any c from an appropriate message domain, computing $c = F_{vk}(s)$ is easy but computing any preimage of s is hard in polynomial-time, where vk is the verification key. But by using the trapdoor key sk , anyone can compute $s = I_{sk}(c)$ efficiently. A type-H signature is $\sigma \leftarrow S_{sk}^{sig}(m) = (s, aux)$, where $s = I_{sk}(H(m, aux))$ and aux is the optional auxiliary information. In the verification algorithm, the signature σ is verified by checking whether $H(m, aux) \stackrel{?}{=} F_{vk}(s)$.

Type-T signature schemes, such as Schnorr signature, are based on three-move honest verifier zero-knowledge proofs. The signing algorithm involves three functions denoted by A, Z and H , where A generates the commitment a using the signer's secret key sk and a randomness r , H is a hash function to generate a challenge c from the message m and the commitment a , and Z generates an answer s to the challenge c . V is a verifying algorithm of a zero-knowledge proof showing the relation between a and s, c . On input s, c and the signer's verification key vk , V outputs z which is supposed to equal a . In the type-T signature schemes, the signer computes the commitment $a \leftarrow A(sk; r)$, the challenge $c = H(m, a)$ and the response $s = Z(sk, r, c)$ in sequence. The resulting signature is $\sigma = (s, c)$. In the verification algorithm, σ is verified by checking whether $c \stackrel{?}{=} H(m, z)$, where $z = V(s, c, vk)$.

In Abe *et al.*'s generic scheme of ring signatures, users in a ring can generate their own secret-verification key pairs (sk, vk) from any type-H or type-T signature schemes. Let $L = \{vk_0, \dots, vk_{n-1}\}$. A signer with the secret key sk_k generates a ring signature on a message m under L as follow:

1. (Initialization): Compute

$$e_k = \begin{cases} A_k(sk_k; \alpha) & , (\text{type-T}), \text{ or} \\ \beta & , (\text{type-H}), \end{cases}$$

where α is randomly selected from the space of the randomness r defined by the algorithm $A_k(sk_k; r)$ in the type-T signature scheme, and β is randomly selected from the space of c defined by the algorithm $I_{sk_k}(c)$ in the type-H signature scheme. Then compute $c_{k+1} = H_{k+1}(L, m, e_k)$.

2. (Forward sequence): For $i = k + 1, \dots, n - 1, 0, \dots, k - 1$, compute

$$e_i = \begin{cases} V_i(s_i, c_i, vk_i) & , (\text{type-T}), \text{ or} \\ c_i + F_i(s_i, vk_i) & , (\text{type-H}), \end{cases}$$

where s_i is randomly selected. Then compute $c_{i+1} = H_{i+1}(L, m, e_i)$.

3. (Forming the ring): Compute

$$s_k = \begin{cases} Z_k(sk_k, \alpha, c_k) & , (\text{type-T}), \text{ or} \\ I_k(\beta - c_k, sk_k) & , (\text{type-H}). \end{cases}$$

The resulting ring signature is $(c_0, s_0, \dots, s_{n-1})$.

In the verification algorithm, for $i = 0, \dots, n - 1$, compute

$$e_i = \begin{cases} V_i(s_i, c_i, vk_i) & , (\text{type-T}), \text{ or} \\ c_i + F_i(s_i, vk_i) & , (\text{type-H}). \end{cases}$$

Then compute $c_{i+1} = H_{i+1}(L, m, e_i)$ if $i \neq n - 1$. If $c_0 = H_0(L, m, e_{n-1})$, the verifier accepts σ as a valid ring signature, reject otherwise.

Concrete Scheme of All Discrete-log Case

Let p_i, q_i be large primes, $\langle g_i \rangle$ denote a prime subgroup of $\mathbb{Z}_{p_i}^*$ generated by g_i whose order is q_i . Compute $y_i = g_i^{x_i} \bmod p_i$, where x_i is the secret key and (y_i, p_i, q_i, g_i) is the public key. $H_i : \{0, 1\}^* \rightarrow \mathbb{Z}_{q_i}$ denotes a collision-resistant hash function. L is a list of (y_i, p_i, q_i, g_i) , where $i = 0, \dots, n - 1$ and $n = |L|$. A signer with the secret key x_k generates a ring signature on a message m under L as follows:

1. Randomly select $\alpha \in \mathbb{Z}_{q_k}$ and compute $c_{k+1} = H_{k+1}(L, m, g_k^\alpha \bmod p_k)$.
2. For $i = k + 1, \dots, n - 1, 0, \dots, k - 1$, randomly select $s_i \in \mathbb{Z}_{q_i}$ and compute $c_{i+1} = H_{i+1}(L, m, g_i^{s_i} y_i^{c_i} \bmod p_i)$, and then $s_k = \alpha - x_k c_k \bmod q_k$.
3. Send the verifier $(c_0, s_0, s_1, \dots, s_{n-1})$ as the resulting ring signature on the message m under the public-key list L .

For $i = 0, \dots, n - 1$, the verifier computes $e_i = g_i^{s_i} y_i^{c_i} \bmod p_i$, and then $c_{i+1} = H_{i+1}(L, m, e_i)$ if $i \neq n - 1$. The verifier accepts the ring signature if $c_0 = H_0(L, m, e_{n-1})$, otherwise rejects.

Concrete Scheme of All RSA Case

For $i = 0, \dots, n-1$, let (e_i, N_i) be the RSA public key and d_i be the private key for the user U_i . $H_i : \{0, 1\}^* \rightarrow \mathbb{Z}_{N_i}$ is a collision-resistant hash function. L is a list of these public keys. A signer U_k signs a message m under L as follow:

1. Randomly select $\beta \in \mathbb{Z}_{N_k}$, and compute $c_{k+1} = H_{k+1}(L, m, \beta)$.
2. For $i = k+1, \dots, n-1, 0, 1, \dots, k-1$, randomly select $s_i \in \mathbb{Z}_{N_i}$, and compute $c_{i+1} = H_{i+1}(L, m, c_i + s_i^{e_i} \bmod N_i)$.
3. Compute $s_k = (\beta - c_k)^{d_k} \bmod N_k$. The resulting ring signature is $\sigma = (c_0, s_0, \dots, s_{n-1})$.

For $i = 0, \dots, n-1$, the verifier computes $r_i = c_i + s_i^{e_i} \bmod N_i$, then compute $c_{i+1} = H_{i+1}(L, m, r_i)$ if $i \neq n-1$. If $c_0 = H_0(L, m, r_{n-1})$, the verifier accepts σ as a valid ring signature, reject otherwise.

3.2.2 Zero-knowledge Proof of Equality of Discrete Logarithms from Different Groups

In [Ate04], Ateniese introduced an underlying proof of the equality of discrete logarithms, which is used for constructing verifiably encrypted signatures. In [CM99], Camenisch and Michels proposed a concrete scheme to prove the equality of discrete logarithms from different groups under the strong RSA assumption. In our OFER-S scheme, we modify Camenisch and Michels' proof as our zero-knowledge proof so as to build a verifiably encrypted signature scheme based on the all discrete-log ring signature introduced above. Camenisch and Michels' proof is denoted by $PK\{(\alpha, \beta) : y_1 \stackrel{G_1}{=} g_1^\alpha \wedge y_2 \stackrel{G_2}{=} g_2^\alpha \wedge \tilde{y} \stackrel{\mathbb{Z}_n^*}{=} h_1^\beta h_2^\alpha \wedge (-2^l < \alpha < 2^l)\}$. The details of the proof are shown below:

n is the product of two sufficiently large safe primes and must be large enough to avoid being factored. h_1 and h_2 are two random elements with large order from \mathbb{Z}_n . Let G_1 and G_2 be two distinct groups of orders q_1 and q_2 such that $2^{l+1} < \min(q_1, q_2)$, where l is an integer, and g_1 and g_2 are the generators of G_1 and G_2 respectively. Let $y_1 \stackrel{G_1}{=} g_1^x$ and $y_2 \stackrel{G_2}{=} g_2^x$, $\epsilon > 1$ is a security parameter which controls the tightness of the statistical zero-knowledgeness. If $-2^{(l-2)/\epsilon} < x < 2^{(l-2)/\epsilon}$, the prover can convince the verifier that $\log_{g_1}^{y_1} = \log_{g_2}^{y_2}$ in \mathbb{Z} by the following steps:

1. The prover randomly chooses $r \in \mathbb{Z}_n$ and computes $\tilde{y} = h_1^r h_2^x \bmod n$, then randomly selects $r_1 \in \{-2^{l-2}, \dots, 2^{l-2}\}$ and $r_2 \in \{-(n2^k)^\epsilon, \dots, (n2^k)^\epsilon\}$, where k

is the length of bits of the verifier's challenge, and computes the commitments: $t_1 = g_1^{r_1}$, $t_2 = g_2^{r_1}$, and $t_3 = h_1^{r_2} h_2^{r_1}$. After that, the prover sends (t_1, t_2, t_3) to the verifier.

2. The verifier returns a random challenge $c \in \{0, 1\}^k$.
3. The prover computes the responses $s_1 = r_1 - cx$ and $s_2 = r_2 - cr$ in \mathbb{Z} , then sends (s_1, s_2) to the verifier.
4. The verifier accepts the proof if and only if $-2^{l-1} < s_1 < 2^{l-1}$, $t_1 = g_1^{s_1} y_1^c$, $t_2 = g_2^{s_1} y_2^c$ and $t_3 = h_1^{s_2} h_2^{s_1} \tilde{y}^c$ hold.

Note that the proof above is based on the strong RSA assumption. The prover should not know the factoring of n . Hence n, h_1, h_2 might be generated by the verifier or a trusted third party. Before executing the proof, the prover should check whether n is the product of two safe primes (see [CM99] for details) and whether h_1 and h_2 have large order (see [GKR97] for details). To convert this interactive proof into a signature form on a message m , the prover can use a suitable hash function $h(\cdot)$, which is agreed by the verifier, to compute the hash value of all the public information instead of the verifier's challenge c (e.g., $c = h(m || \tilde{y} || y_1 || y_2 || g_1 || g_2 || t_1 || t_2 || t_3)$).

3.2.3 Camenisch-Shoup Encryption Scheme

In [Ate04], Ateniese proposed a method to construct verifiably encrypted signatures by encrypting an ordinary signature using some specific public-key cryptosystems and giving a proof showing the validity of the signature's encryption. In such cryptosystems (e.g. Naccache-Stern [NS98], Okamoto-Uchiyama [OU98] and Paillier [Pai99] public-key cryptosystems), computing a discrete logarithm using the secret key is an easy task, but without the secret key, it is still hard. However, all these public-key cryptosystems above do not satisfy the high level security which protects against adaptive chosen-ciphertext attacks (CCA2). In [CS03], Camenisch and Shoup proposed an adaptation of Paillier cryptosystem, which is proven secure against adaptive chosen ciphertext attacks under the decisional composite residuosity assumption. To achieve the high level security, we use Camenisch and Shoup's scheme as our encryption scheme, which is briefly described as follows:

1. Randomly select two Sophie Germain primes p' and q' , where $p' \neq q'$, and compute safe primes $p = 2p' + 1$, $q = 2q' + 1$ and $n = pq$. Then randomly select $x_1, x_2, x_3 \in_R [n^2/4]^1$ and $g' \in \mathbb{Z}_{n^2}^*$, and compute $g = (g')^{2n}$, $y_1 = g^{x_1}$, $y_2 = g^{x_2}$, $y_3 = g^{x_3}$. Let $h = (1 + n \bmod n^2) \in \mathbb{Z}_{n^2}^*$, **abs**: $\mathbb{Z}_{n^2}^* \rightarrow \mathbb{Z}_{n^2}^*$ map $(a \bmod n^2)$, where $0 < a < n^2$, to $(n^2 - a \bmod n^2)$ if $a > n^2/2$, and to $(a \bmod n^2)$ otherwise. Obviously for any $v \in \mathbb{Z}_{n^2}^*$, $v^2 = (\mathbf{abs}(v))^2$ holds. H is a collision-resistant hash function. A label L is some public information added to the ciphertext (e.g., user's identity or expiration time). The public key is (n, g, y_1, y_2, y_3) , and the private key is (x_1, x_2, x_3) .
2. To encrypt a message $m \in [n]$ with a label $L \in \{0, 1\}^*$, randomly select $r \in_R [n/4]$ and compute $u = g^r$, $e = y_1^r h^m$ and $v = \mathbf{abs}((y_2 y_3^{H(u,e,L)})^r)$. The triple (u, e, v) is the resulting ciphertext.
3. To decrypt a ciphertext (u, e, v) , first check whether $\mathbf{abs}(v) = v$ and $u^{2(x_2 + H(u,e,L)x_3)} = v^2$. If fail, output reject, otherwise compute $\hat{m} = (e/u^{x_1})^{2t}$, where $t = 2^{-1} \bmod n$. If \hat{m} is of the form h^m for some $m \in [n]$, then output m , otherwise output reject.

Recall the ring signature scheme of all discrete-log case presented in Section 3.2.1. Suppose the signer generates a ring signature $(c_0, s_0, s_1, \dots, s_{n-1})$. In the verification of this signature, the verifier needs to compute $e_i = g_i^{s_i} y_i^{c_i}$, where $i = 0, 1, \dots, n-1$. In order to convert the ring signature into a verifiably encrypted ring signature (VERS), the signer sends the verifier $w_i = g_i^{s_i}$ instead of s_i and encrypts s_i using a TTP's public key. The verifier can do the verification by computing $e_i = w_i y_i^{c_i}$ instead, but s_i is 'hidden' in w_i since in this ring signature scheme computing a discrete logarithm is hard, which means the verifier has not got the full ring signature yet. Beside that, the signer needs to give a zero-knowledge proof for convincing the verifier that the encrypted s_i is just the s_i hidden in w_i . Note that encrypting only one value in $(s_0, s_1, \dots, s_{n-1})$ can also ensure the initial ring signature hidden partially, which means the verifier still cannot draw the full ring signature from the partially encrypted ring signature even though he gets the most parts of the initial ring signature. Encrypting one value makes the cost of generating a VERS does not depend on the size of the public-key list, which improves the efficiency of the generation of a VERS.

¹For a positive integer a , $[a]$ denotes the set $\{0, 1, \dots, a-1\}$.

To produce a verifiably encrypted ring signature, suppose the signer randomly chooses s_u , where $0 \leq u \leq n - 1$, from $(s_0, s_1, \dots, s_{n-1})$ as the hidden value, and encrypts s_u using Camenisch and Shoup's encryption scheme above. Let $(n, \mathbf{g}, y_1, y_2, y_3, \mathbf{h})$ be the public key of a TTP. H is a collision-resistant hash function, and L is the public label. The signer computes s_u 's ciphertext $\mathbf{u} = \mathbf{g}^t, \mathbf{e} = y_1^t \mathbf{h}^{s_u}, \mathbf{v} = \mathbf{abs}((y_2 y_3^{H(u, \mathbf{e}, L)})^t)$, where $t \in_R [\mathbf{n}/4]$. After that, by modifying the zero-knowledge proof introduced in Section 3.2.2, the signer gives a non-interactive proof: $PK\{(s_u, \mathbf{t}, r) : w = g_u^{s_u} \wedge \mathbf{u}^2 = \mathbf{g}^{2t} \wedge \mathbf{e}^2 = y_1^{2t} \mathbf{h}^{2s_u} \wedge \mathbf{v}^2 = (y_2 y_3^{H(u, \mathbf{e}, L)})^{2t} \wedge \hat{w} = h_1^r h_2^{s_u} \wedge -2^l < s_u < 2^l\}$ to convince the verifier that the TTP can extract s_u using its secret key and recover the original ring signature completely. Note that anyone beside the signer has the capability to convert a valid ring signature into a VERS without knowing any secret information from the signer. The property *signer ambiguity* of ring signatures [AOS02] is well preserved since the hidden value can be arbitrarily chosen in (s_0, \dots, s_{n-1}) and no secret of the signer is needed for producing a VERS based on a given ring signature. In our verifiably encrypted ring signature scheme, for the sake of simplicity, we specify s_{n-1} as the hidden value encrypted using a TTP's public key no matter who the signer is. The details are shown in Section 3.4.

3.3 Security Definitions

In [DLY07], Dodis *et al.* presented a formal security model of optimistic fair exchange under *adaptive chosen message attacks* in a *multi-user setting*, in which the optimistic fair exchange protocol can be executed between different signers and different verifiers. That is, multiple pairs of users can run the two-party fair exchange protocol without compromising security. In the adaptive chosen message attacks [GMR88], an adversary can access the signing oracle by asking for signatures on arbitrary messages. In ring signatures, there are multiple users belonging to each public-key list. So the multi-user setting is necessary for fair exchange of ring signatures. Furthermore, Huang *et al.* [HYWS08b] extended Dodis *et al.*'s model by considering *chosen-key model*, i.e., an adversary may win a computational game if it is allowed to employ some public keys without knowing the corresponding private keys. By providing this extra flexibility, the chosen-key model is stronger than the certified-key model [HYWS08b]. In addition, we also consider *chosen public-key attacks* in the setting of ring signatures, which is proposed by Abe *et al.* [AOS02]. In

chosen public-key attacks, any adversary who wants to forge a ring signature is only allowed to use arbitrary subsets of the initially considered public-key list to access the signing oracle, but cannot append new public keys to the initial public-key list. Therefore, in our security definitions specified below, all the four factors above are addressed in the setting of OFERS as a whole.

Definition 3.1. (Syntax) *Optimistic fair exchange of ring signatures (OFERS) consists of seven probabilistic polynomial-time algorithms.*

- **Setup^{TTP}**: *On input a security parameter κ , the arbitrator executes the algorithm to generate a public-private key pair (APK, ASK) and some auxiliary information if necessary.*
- **Setup^{User}**: *On input κ and (optionally) the arbitrator's public key with the auxiliary information, the algorithm outputs public-private key pairs (PK_i, SK_i) for every user in the ring. The public keys form a public-key list L .*
- **RSig (m, L, SK_s)** : *A signer U_s in the ring executes the algorithm by inputting a message m , a public-keys list L including PK_s and its corresponding private key SK_s , then outputs a ring signature σ .*
- **RVer (m, L, σ)** : *On input a message m , a ring signature σ on m under a public-key list L , a verifier executes the algorithm to output either 1 or 0, which means accept or reject respectively.*
- **PRSig (m, L, σ, APK)** : *On input a message m , a signer's public-key list L , a ring signature σ on m under L , and the arbitrator's public key APK , the algorithm outputs a verifiably partial ring signature θ .*
- **PRVer (m, L, θ, APK)** : *On input a message m , a signer's public-key list L , a verifiably partial ring signature θ on m under L , and the arbitrator's public key APK , the verifier executes the algorithm to output either 1 or 0, which means accept or reject respectively.*
- **Res (m, L, θ, ASK)** : *The resolution algorithm is executed by the arbitrator if the verifier does not receive the full ring signature σ from the signer ring, but has got the corresponding verifiably partial ring signature θ . On input a message m , a signer's public-key list L and a verifiably partial ring signature*

θ on m under L , if θ is valid and the verifier has fulfilled its obligation to the signer, the arbitrator extracts the full ring signature σ from θ using its private key ASK and reveals it to the verifier, otherwise rejects.

Since there are three roles (*signer, verifier, arbitrator*) in OFERS, we should consider how each role may violate different aspects of security, i.e., different security properties. Here we require the arbitrator should not be able to cheat some participant by colluding with the other participant in the protocol since such a collusive adversarial arbitrator can break the fair exchange trivially. Moreover, the property *signer ambiguity* should also be addressed as it is the heritage of ring signatures.

Security Against Signers: For the fairness to verifiers, it is required that except negligible probability, any probabilistic polynomial-time (PPT) adversarial signer \mathcal{A} should be not able to generate a verifiably partial ring signature, which can be accepted by verifiers, but cannot be recovered to a valid full ring signature by an honest arbitrator. The property is formally defined by the following game:

$$\begin{aligned} \mathbf{Setup}^{\mathbf{TTP}}(\kappa) &\longrightarrow (ASK, APK) \\ (m, L^*, \theta) &\longleftarrow \mathcal{A}^{O_{Res}}(APK) \\ \sigma &\longleftarrow \mathbf{Res}(m, L^*, \theta, ASK) \\ \text{Success of } \mathcal{A} &= [\mathbf{PRVer}(m, L^*, \theta, APK)=1 \wedge \mathbf{RVer}(m, L^*, \sigma)=0] \end{aligned}$$

where O_{Res} denotes a resolution oracle, which takes as input a verifiably partial ring signature on a message m under a public-key list L , and outputs a full ring signature σ on m under L . In this game, the adversary \mathcal{A} is allowed to *arbitrarily* (i.e., not necessarily following the key generation algorithm) generate public keys to form a list L^* . For each public key in L^* , \mathcal{A} may not know the corresponding private key. The chosen-key model is therefore accommodated here.

Definition 3.2 (Security Against Signers) *Optimistic fair exchange of ring signatures is said to be secure against signers if there is no PPT adversarial signer \mathcal{A} who wins the game above with non-negligible probability.*

Security Against Verifiers: The property of *security against verifiers* requires that, without help from the signer or the arbitrator, any PPT adversarial verifier \mathcal{B} should not be able to extract a full ring signature from the corresponding verifiably partial ring signature with non-negligible probability. The property is formally defined by the following game:

$$\begin{aligned}
\text{Setup}^{\text{TTP}}(\kappa) &\longrightarrow (ASK, APK) \\
\text{Setup}^{\text{User}}(\kappa) &\longrightarrow (SK_i, PK_i) \\
(m, L', \sigma) &\longleftarrow \mathcal{B}^{O_{PRSig}, O_{Res}}(APK, L) \\
\text{Success of } \mathcal{B} &= [\mathbf{RVer}(m, L', \sigma)=1 \wedge (m, L', \cdot) \notin \text{Query}(\mathcal{B}, O_{Res})]
\end{aligned}$$

where L' is an arbitrary subset of the initial public-key list L consisting of all the PK_i , the oracle O_{Res} has been defined in the previous game, and the partial ring signature signing oracle O_{PRSig} , given as input a message m and a public key list L'' , outputs a verifiably partial ring signature on m under L'' using the arbitrator's public key APK . The $\text{Query}(\mathcal{B}, O_{Res})$ is the set of valid queries which \mathcal{B} asks to O_{Res} . In this game, \mathcal{B} can ask the arbitrator for resolving any verifiably partial ring signature with respect to any sublist of L . Note that here chosen-public key attacks are considered, as the adversary \mathcal{B} is only required to output a valid ring signature under L' which is a subset of L but not necessarily L . Moreover, L' does not contain any public key generated by \mathcal{B} . Otherwise, \mathcal{B} can win the game above trivially.

Definition 3.3 (Security Against Verifiers) *Optimistic fair exchange of ring signatures is said to be secure against verifiers if there is no PPT adversarial verifier \mathcal{B} who wins the game above with non-negligible probability.*

Security Against the Arbitrator: For the fairness to signers, the property of *security against the arbitrator* requires that except negligible probability, any PPT adversarial arbitrator \mathcal{C} should not be able to produce a full ring signature without demanding the signer to generate a verifiably partial ring signatures. The property is formally defined by the following game:

$$\begin{aligned}
\text{Setup}^{\text{User}}(\kappa) &\longrightarrow (PK_i, SK_i) \\
(ASK^*, APK) &\longleftarrow \mathcal{C}(L) \\
(m, L', \sigma) &\longleftarrow \mathcal{C}^{O_{PRSig}}(ASK^*, APK, L) \\
\text{Success of } \mathcal{C} &= [\mathbf{RVer}(m, L', \sigma)=1 \wedge (m, L') \notin \text{Query}(\mathcal{C}, O_{PRSig})]
\end{aligned}$$

where the oracles O_{Res} , O_{PRSig} , the public-key lists L' and L have been described in the previous games, and ASK^* is the state information of \mathcal{C} , which may not correspond to the arbitrator's public key APK . $\text{Query}(\mathcal{C}, O_{PRSig})$ is the set of valid queries which \mathcal{C} asks to O_{PRSig} . We remark that this game considers both chosen-key and chosen public-key attacks in the multi-user setting, as the adversary \mathcal{C} (a malicious arbitrator) does not need to know the corresponding private key of the

public key APK and can choose any sublist L' of the initial public-key list to forge a ring signature.

Definition 3.4 (Security Against the Arbitrator) *Optimistic fair exchange of ring signatures is said to be secure against the arbitrator if there is no PPT adversarial arbitrator \mathcal{C} who wins the game above with non-negligible probability.*

In [AOS02], Abe *et al.* specified the security definition of *signer ambiguity*. In our OFERS scheme, the signer should be still ambiguous in its own ring. By updating Abe *et al.*'s definition in the setting of OFERS, we formally define signer ambiguity as follows:

Definition 3.5 (Signer Ambiguity) *Let $L = \{PK_i\}$ be an initial public-key list, where each PK_i is generated by running $\mathbf{Setup}^{User} \rightarrow (PK_i, SK_i)$, and APK be the arbitrator's public key generated by running $\mathbf{Setup}^{TTP} \rightarrow (APK, ASK)$. An OFERS protocol is called **perfectly signer-ambiguous**, if for any message m , any public-key list L , any public key APK of the arbitrator, any valid full ring signature $\sigma \leftarrow \mathbf{RSign}(m, L, SK_s)$, and an associated verifiably partial ring signature $\theta \leftarrow \mathbf{PRSig}(m, L, \sigma, APK)$, where SK_s is the signer's private key, given $(m, L, \theta, \sigma, APK)$, any unbound adversary \mathcal{D} outputs index i such that $SK_s = SK_i$ with probability exactly $\frac{1}{|L|}$, where $|L|$ denotes the size of L .*

Remark 1. Comparing with Abe *et al.*'s signer ambiguity [AOS02] for ring signatures, we also provide the verifiably partial ring signature θ of a full ring signature σ to the adversary \mathcal{D} , which allows \mathcal{D} acquiring more information to break signer ambiguity. In fact, this is necessary because the signer ambiguity in ring signatures does not always guarantee the same property for OFERS (refer to the counterexample discussion in Section 3.5). As the unbound adversary \mathcal{D} can derive all private keys from L , the above definition essentially means that for fixed (m, L, APK) , the distributions of θ and σ generated by using any private key SK_i are identical. In addition, Definition 3.5 specifies *perfect signer ambiguity*, and it can be easily extended to define *statistical* and *computational signer ambiguity*, two weaker versions of ambiguity.

3.4 The Proposed Scheme

In our OFERS scheme, we use verifiably encrypted ring signatures (VERS) as verifiably partial ring signatures. In this section, we first present how to produce a VERS, and then give an optimistic fair exchange protocol of ring signatures. In order to provide an efficient zero-knowledge proof, Abe *et al.*'s ring signature scheme in all discrete-log case is employed in our OFERS scheme. The generation and verification of ring signatures are similar to the all discrete-log ring signature scheme in Section 3.2.1 except some limitation of selecting α and s_i . For the sake of simplicity, in our VERS scheme, we always encrypt the last s_i , i.e. s_{n-1} , as the hidden value. Obviously this does not affect the scheme's security since the sequence of the public-key list can be arbitrarily changed by the signer and any s_i in (s_0, \dots, s_{n-1}) can be the hidden value no matter who the signer is. Then we use Camenisch and Shoup's CCA2-secure encryption scheme and give a proof:

$$PK\{(s_{n-1}, \mathbf{t}, r) : w = g_{n-1}^{s_{n-1}} \wedge \mathbf{u}^2 = \mathbf{g}^{2\mathbf{t}} \wedge \mathbf{e}^2 = y_1^{2\mathbf{t}} h^{2s_{n-1}} \wedge \mathbf{v}^2 = (y_2 y_3^{\mathbf{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^{2\mathbf{t}} \wedge \hat{w} = h_1^r h_2^{s_{n-1}} \wedge -2^l < s_{n-1} < 2^l\}$$

for convincing the verifier the validity of the encryption.

3.4.1 Verifiably Encrypted Ring Signature

The generation of a VERS consists of two steps. One is producing a conventional ring signature consisting of three algorithms denoted by $\mathbf{RS} = (\mathbf{RKG}, \mathbf{Sig}, \mathbf{Ver})$, the other is encrypting the ring signature consisting of three algorithms denoted by $\mathbf{EN} = (\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ with a zero-knowledge showing the validity of the ring signature's encryption. Suppose there are two rings called R_I and R_J . U_i and U_j denote the users in these two rings respectively. A signer U_k in the ring R_I sends a VERS on a message m to a verifier in the ring R_J . L_I and L_J denote the public-key list of the ring R_I and R_J , and $n_I = |L_I|$ and $n_J = |L_J|$ denote the size of L_I and L_J respectively.

Setup^{TTP}: On input the security parameter κ , the arbitrator executes the key generation algorithm to output the public key $(\mathbf{n}, \mathbf{g}, y_1, y_2, y_3, \mathbf{h})$ and the private key (x_1, x_2, x_3) under Camenisch and Shoup's encryption scheme [CS03]. q_A denotes the order of \mathbf{g} , and l is an integer such that $2^{l+1} < q_A$. Meanwhile, the arbitrator generates h_1 , h_2 and n , which are used in the zero-knowledge proof introduced

in Section 3.2.2 (In order to avoid being factored, the modulus n must be large enough but does not need to depend on κ) and publishes $(n, \mathbf{g}, y_1, y_2, y_3, \mathbf{h}, h_1, h_2, n, l)$.

Setup^{User}: The setup of users is similar to the ring signature scheme in Section 3.2.1. For the user U_i , let $y_i = g_i^{x_i} \bmod p_i$, where the order of g_i is $q_i > 2^{l+1}$. x_i is the secret key and (y_i, p_i, q_i, g_i) is the public key. $H_i : \{0, 1\}^* \rightarrow \mathbb{Z}_{q_i}$ is public collision-resistant hash functions.

RSign: The signer U_k in the ring R_I signs a message m by executing the algorithm below:

1. Randomly select $\alpha \in \mathbb{Z}_{q_k}$, and compute $c_{k+1} = H_{k+1}(L_I, m, g_k^\alpha \bmod p_k)$.
2. For $i = k + 1, \dots, n_I - 1, 0, 1, \dots, k - 1$, randomly select $s_i \in (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$, and compute $c_{i+1} = H_{i+1}(L_I, m, g_i^{s_i} y_i^{c_i} \bmod p_i)$.
3. Compute $s_k = \alpha - x_k c_k \bmod q_k$, where $s_k \in (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$. If $s_k \notin (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$, properly reselect α and run the Step 1 to 3 again until s_k lies in the right interval. The resulting ring signature is $\sigma_I = (c_0, s_0, \dots, s_{n_I-1})$.

RVer: For $i = 0, \dots, n_I - 1$, the verifier computes $e_i = g_i^{s_i} y_i^{c_i} \bmod p_i$, then compute $c_{i+1} = H_{i+1}(L_I, m, e_i)$ if $i \neq n_I - 1$. If $c_0 = H_0(L_I, m, e_{n_I-1})$, the verifier accepts σ_I as a valid ring signature, reject otherwise.

PRSig: The algorithm is used for converting a full ring signature σ_I to a verifiably encrypted ring signature θ_I . Let $\hat{h} : \{0, 1\}^* \rightarrow \{0, 1\}^\eta$ be a collision-resistant hash function and the public label $\mathbf{L} = m || L_I$.

1. Compute $w = g_{n_I-1}^{s_{n_I-1}}$ and encrypt s_{n_I-1} by computing

$$\mathbf{u} = \mathbf{g}^t, \quad \mathbf{e} = y_1^t h^{s_{n_I-1}}, \quad \mathbf{v} = \text{abs}(y_2 y_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^t$$

under Camenisch and Shoup's encryption scheme.

2. Randomly select $r \in \mathbb{Z}_n$, $r_1 \in (-2^{l-2}, 2^{l-2})$, $r_2 \in (-(n2^\eta)^\epsilon, (n2^\eta)^\epsilon)$ and $r_3 \in (-(n2^\eta)^\epsilon, (n2^\eta)^\epsilon)$, compute $\hat{w} = h_1^r h_2^{s_{n_I-1}} \bmod n$ and $t_1 = g_{n_I-1}^{r_1}$, $t_2 = h_1^{r_2} h_2^{r_1}$, $\mathbf{u}' = \mathbf{g}^{r_3}$, $\mathbf{e}' = y_1^{r_3} h^{r_1}$ and $\mathbf{v}' = (y_2 y_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^{r_3}$ in their own groups.

3. Compute $\hat{c} = \hat{h}(L_I, m, w, \hat{w}, \mathbf{u}, \mathbf{e}, \mathbf{v}, g_{n_I-1}, \mathbf{g}, h_1, h_2, t_1, t_2, \mathbf{u}'^2, \mathbf{e}'^2, \mathbf{v}'^2)$ and $v_1 = r_1 - \hat{c}s_{n_I-1}$, $v_2 = r_2 - \hat{c}r$, $v_3 = r_3 - \hat{c}t$ in \mathbb{Z} . The resulting VERS is $\theta_I = (c_0, s_0, \dots, s_{n_I-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v}, \hat{w}, \hat{c}, t_1, t_2, \mathbf{u}', \mathbf{e}', \mathbf{v}', v_1, v_2, v_3)$.

PRVer: The verifier first computes $\hat{c}' = \hat{h}(L_I, m, w, \hat{w}, \mathbf{u}, \mathbf{e}, \mathbf{v}, g_{n_I-1}, \mathbf{g}, h_1, h_2, g_{n_I-1}^{v_1} w^{\hat{c}}, h_1^{v_2} h_2^{v_1} \hat{w}^{\hat{c}}, \mathbf{g}^{2v_3} \mathbf{u}^{2\hat{c}}, \mathbf{y}_1^{2v_3} \mathbf{h}^{2v_1} \mathbf{e}^{2\hat{c}}, (\mathbf{y}_2 \mathbf{y}_3^{\mathbf{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^{2v_3} \mathbf{v}^{2\hat{c}})$, and checks whether $\hat{c}' = \hat{c}$ and $-2^{l-1} < v_1 < 2^{l-1}$. If any condition does not hold, outputs the VERS θ_I is invalid, otherwise computes $e_i = g_i^{s_i} y_i^{c_i}$ for $i = 0, \dots, n_I - 2$ and $e_{n_I-1} = w y_{n_I-1}^{c_{n_I-1}}$, and then computes $c_{i+1} = H_{i+1}(L_I, m, e_i)$ if $i \neq n_I - 1$. If $c_0 = H_0(L_I, m, e_{n_I-1})$, the verifier accepts θ_I , reject otherwise.

Res: After the verifier shows a proof that he has fulfilled his obligation to the signer, the arbitrator decrypts the ciphertext $(\mathbf{u}, \mathbf{e}, \mathbf{v})$ using its secret key $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ to extract s_{n_I-1} , and reveals the full ring signature σ_I to the verifier.

3.4.2 Optimistic Fair Exchange of Ring Signatures

By applying the verifiably encrypted ring signature scheme above, an optimistic fair exchange protocol of ring signatures can easily be set up. Suppose two users U_i and U_j in the rings R_I and R_J respectively exchange their ring signatures on a message m . Either U_i or U_j sends his/her own signatures to the other ring without knowing who is the real receiver in the other ring. The optimistic fair exchange protocol proceeds as follows:

1. U_i computes his ring signature $\sigma_I = \mathbf{RSign}(m, L_I, SK_i)$, and converts this ring signature into a VERS $\theta_I = \mathbf{PRSig}(m, L_I, \sigma_I, APK)$ using the arbitrator's public key APK , then sends θ_I to R_J .
2. U_j checks whether $\mathbf{PRVer}(m, L_I, \theta_I, APK) = 1$. If no, U_j quits, otherwise U_j computes his ring signature σ_J and sends it to R_I .
3. U_i checks whether $\mathbf{RVer}(m, L_J, \sigma_J) = 1$, if no, U_i stops the protocol, otherwise U_i sends σ_I to R_J .
4. U_j checks whether $\mathbf{RVer}(m, L_I, \sigma_I) = 1$, if yes, U_j accepts this ring signature. If σ_I is invalid or U_j receives nothing from R_I , U_j sends the arbitrator θ_I and σ_J to apply for resolution. The arbitrator first checks whether σ_J is valid, if yes, the arbitrator runs the algorithm $\mathbf{Res}(m, L_I, \theta_I, ASK)$ to recover σ_I , then

sends σ_I to R_J and σ_J to R_I . If σ_J is invalid, the arbitrator will send a signal to both R_I and R_J to inform U_i and U_j that the protocol has been terminated.

Note that after Step 1, U_j can decide to carry on the protocol at any time he wants, which might give U_j some advantages. To solve this problem, before the protocol runs, U_i and U_j can set up a time point at which the protocol must be completed.

3.5 Security Proofs

In this session, we prove that our OFE protocol for ring signatures is secure in the multi-user setting under adaptive chosen message, chosen-key and chosen public-key attacks. Let $\mathbf{RS} = (\mathbf{RKG}, \mathbf{RSig}, \mathbf{RVer})$ denote Abe *et al.*'s ring signature scheme, $\mathbf{EN} = (\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ denote Camenisch-Shoup public-key encryption scheme, and π be a non-interactive zero-knowledge proof showing the proper encryption of a full ring signature. We have the following theorem:

Theorem 1: *The proposed optimistic fair exchange of ring signatures is secure, i.e., satisfies Definitions 3.2-3.5, if the underlying \mathbf{RS} is secure with signer ambiguity and existential unforgeability against adaptive chosen message and chosen public-key attacks, \mathbf{EN} is secure against adaptive chosen ciphertext attacks (CCA2), and π is a simulation-sound non-interactive zero-knowledge proof.*

Proof. SECURITY AGAINST SIGNERS: In our OFERS protocol, a valid verifiably encrypted ring signature $\theta = (c_0, s_0, s_1, \dots, s_{n-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v}, \hat{w}, \hat{c}, t_1, t_2, \mathbf{u}', \mathbf{e}', \mathbf{v}', v_1, v_2, v_3)$ consists of three parts. The first part $(c_0, s_0, s_1, \dots, s_{n-2}, w)$ is a ‘ring signature’, where s_{n-1} is hidden in $w = g_{n-1}^{s_{n-1}}$. The second part $(\mathbf{u}, \mathbf{e}, \mathbf{v})$ is the ciphertext of encrypting s_{n-1} under the arbitrator’s public key, where $\mathbf{u} = \mathbf{g}^t$, $\mathbf{e} = \mathbf{y}_1^t \mathbf{h}^{s_{n-1}}$ and $\mathbf{v} = \text{abs}(\mathbf{y}_2 \mathbf{y}_3^{\mathbf{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^t$ for some t . The third part $(\hat{w}, \hat{c}, t_1, t_2, \mathbf{u}', \mathbf{e}', \mathbf{v}', v_1, v_2, v_3)$ provides a non-interactive zero-knowledge proof:

$$\begin{aligned} \pi = PK\{(s_{n-1}, t, r) : w = g_{n-1}^{s_{n-1}} \wedge \mathbf{u}^2 = \mathbf{g}^{2t} \wedge \mathbf{e}^2 = \mathbf{y}_1^{2t} \mathbf{h}^{2s_{n-1}} \wedge \mathbf{v}^2 = (\mathbf{y}_2 \mathbf{y}_3^{\mathbf{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^{2t} \\ \wedge \hat{w} = h_1^r h_2^{s_{n-1}} \wedge -2^l < s_{n-1} < 2^l\}, \end{aligned}$$

which shows that the encrypted s_{n-1} is the same value hidden in w . Suppose an adversary \mathcal{A} breaks the security against signers in our OFERS protocol by forging a

VERS $\theta = (c_0, s_0, s_1, \dots, s_{n-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v}, \hat{w}, \hat{c}, t_1, t_2, \mathbf{u}', \mathbf{e}', \mathbf{v}', v_1, v_2, v_3)$ w.r.t a public-key list L^* generated by himself, where $w = g_{n-1}^{s_{n-1}}$ but $\mathbf{e} = \mathbf{y}_1^\dagger \mathbf{h}^{s'_{n-1}}$ for $s'_{n-1} \neq s_{n-1}$. For each public key in L^* , \mathcal{A} may not know the corresponding private key. According to Definition 3.2, \mathcal{A} wins the game of *security against signers* if and only if the corresponding full ring signature of θ is $\sigma = (c_0, s_0, s_1, \dots, s_{n-2}, s_{n-1})$ and $(\mathbf{u}, \mathbf{e}, \mathbf{v})$ is decrypted to get s'_{n-1} , where $s'_{n-1} \neq s_{n-1}$. However, this is infeasible due to the *soundness* of the zero-knowledge proof π . Hence our OFERS protocol is secure against signers if π is a non-interactive zero-knowledge proof (NIZK).

SECURITY AGAINST VERIFIERS: Suppose an adversarial verifier \mathcal{B} breaks the security against verifiers in the proposed OFERS protocol. We now construct a distinguisher $\bar{\mathcal{B}}$, who can successfully distinguish the encryption of two messages with the same length of its choice from a challenger in the CCA2 game for Camenisch-Shoup encryption scheme with non-negligible probability. Note that $\bar{\mathcal{B}}$ is allowed to access the decryption oracle O_{Dec} of the encryption scheme. According to Definition 3.3, \mathcal{B} wins the game of *security against verifiers* if \mathcal{B} produces a valid ring signature σ on a message m under a public-key list L' without asking the resolution oracle O_{Res} any query (m, L', θ) . As (m, L', σ) is a successful forgery of \mathcal{B} , the situation that \mathcal{B} did not ask any corresponding VERS θ of σ via the partial ring signature signing oracle O_{PRSig} is negligible due to *security against the arbitrator* proved below. Hence we require that \mathcal{B} gets θ from O_{PRSig} here. Now we show how to construct $\bar{\mathcal{B}}$ in detail.

For the given target Camenisch-Shoup encryption scheme $\mathbf{EN}=(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ with the public key APK , the distinguisher $\bar{\mathcal{B}}$ repeatedly executes Abe *et al.*'s key generation algorithm, $\mathbf{RKG} \rightarrow \{PK_i, SK_i\}$, to form a public-key list L . Then $\bar{\mathcal{B}}$ sends (APK, L) to \mathcal{B} as the input of the OFERS protocol. Let k be the total number of the queries that \mathcal{B} issues to O_{PRSig} . After arbitrarily selecting j from $\{1, 2, \dots, k\}$, $\bar{\mathcal{B}}$ simulates O_{PRSig} 's response to each query (m_i, L_i) issued by \mathcal{B} , where $i = 1, 2, \dots, k$, $L_i \subseteq L$ and $n_i = |L_i|$, as follows:

1. If $i \neq j$, $\bar{\mathcal{B}}$ signs the message m_i w.r.t L_i using the private key SK_0 to generate a ring signature $\sigma_i = \mathbf{RSig}(m_i, L_i, SK_0) = (c_{i_0}, s_{i_0}, \dots, s_{i_{n_i-1}})$ and returns a VERS $\theta_i = (c_{i_0}, s_{i_0}, \dots, s_{i_{n_i-2}}, w_i, \varepsilon_i, \pi_i)$, where $w_i = g_{i_{n_i-1}}^{s_{i_{n_i-1}}}$ and $\varepsilon_i = \mathbf{Enc}_{APK}(s_{i_{n_i-1}})$ under Camenisch-Shoup encryption scheme, and π_i is a NIZK proof showing that ε_i encrypts the same value hidden in w_i , i.e. $s_{i_{n_i-1}}$.

2. If $i = j$, $\bar{\mathcal{B}}$ computes $\sigma_i = \mathbf{RSig}(m_i, L_i, SK_0) = (c_{i_0}, s_{i_0}, \dots, s_{i_{n_i-1}})$ and chooses a proper $\hat{s}_{i_{n_i-1}}$ in the same interval of $s_{i_{n_i-1}}$ but $s_{i_{n_i-1}} \neq \hat{s}_{i_{n_i-1}}$. Then $\bar{\mathcal{B}}$ sets $\dot{s}_1 = s_{i_{n_i-1}}$ and $\dot{s}_0 = \hat{s}_{i_{n_i-1}}$ and sends \dot{s}_1 and \dot{s}_0 to its CCA2 challenger. The challenger returns a ciphertext ε_b , which equals either $\mathbf{Enc}_{APK}(s_{i_{n_i-1}})$ or $\mathbf{Enc}_{APK}(\hat{s}_{i_{n_i-1}})$. After that, $\bar{\mathcal{B}}$ returns $\theta_i = (c_{i_0}, s_{i_0}, \dots, s_{i_{n_i-2}}, w_i, \varepsilon_i, \pi_i)$, where $w_i = g_{i_{n_i-1}}^{s_{i_{n_i-1}}}$, $\varepsilon_i = \varepsilon_b$, and π_i is a *simulated* NIZK proof showing that ε_i encrypts the same value hidden in w_i .

The distinguisher $\bar{\mathcal{B}}$ simulates the resolution oracle O_{Res} 's response to \mathcal{B} 's queries (m_i, L_i, θ_i) using the decryption oracle O_{Dec} as follows:

1. If π_i is valid and $L_i \neq L_j$, $\bar{\mathcal{B}}$ asks O_{Dec} to extract the plaintext $s_{i_{n_i-1}}$ from ε_i and returns the ring signature $\sigma_i = (c_{i_0}, s_{i_0}, \dots, s_{i_{n_i-1}})$ on m_i under L_i .
2. If π_i is valid and $L_i = L_j$, $\bar{\mathcal{B}}$ checks whether $m_i = m_j$. If yes, $\bar{\mathcal{B}}$ aborts the simulation and sends a random bit to its CCA2 challenger. Otherwise, $\bar{\mathcal{B}}$ asks O_{Dec} to extract the plaintext $s_{i_{n_i-1}}$ from ε_i and returns the ring signature $\sigma_i = (c_{i_0}, s_{i_0}, \dots, s_{i_{n_i-1}})$ on m_i under L_i .
3. If π_i is invalid, $\bar{\mathcal{B}}$ returns \mathcal{B} a random value.

Note that \mathcal{B} cannot get the valid response from O_{Res} on any partial ring signature which contains ε_b . This is guaranteed by the properties *soundness* and *simulation-soundness* [Sah99] of the non-interactive zero-knowledge proof π . *Soundness* guarantees that a PPT prover should not be able to convince the verifier of a false statement except negligible probability. And *simulation-soundness* remains this case even through the prover has been given a simulated proof of his/her choosing. Therefore, in the case of $i = j$, even after \mathcal{B} has seen a simulated NIZK proof π_i , he/she cannot forge a partial signature θ^* containing ε_b together with a valid π^* under m^* and L^* , where $(m^*, L^*) \neq (m_j, L_j)$.

If $\varepsilon_b = \mathbf{Enc}_{APK}(\hat{s}_{i_{n_i-1}})$ (i.e. $b = 0$), θ_i looks valid but, in fact, $\sigma_i = (c_{i_0}, s_{i_0}, \dots, \hat{s}_{i_{n_i-1}})$ is not a valid ring signature because of $\hat{s}_{i_{n_i-1}} \neq s_{i_{n_i-1}}$. The probability of \mathcal{B} forging a valid ring signature on m_j is therefore negligible. If $\varepsilon_b = \mathbf{Enc}_{APK}(s_{i_{n_i-1}})$ (i.e. $b = 1$), ε_j is a valid encryption of $s_{i_{n_i-1}}$ which is a part of a valid ring signature on m_j . The attack environment required by \mathcal{B} is perfectly simulated. Suppose (m, L', σ) is the forgery of \mathcal{B} , if $m = m_j$ and $L' = L_j$, $\bar{\mathcal{B}}$ outputs 1 and wins the CCA2 game by indicating that $\dot{s}_1 = s_{i_{n_i-1}}$ is the plaintext of ε_b , otherwise $\bar{\mathcal{B}}$ sends a random bit to

the CCA2 challenger. Consequently, if \mathcal{B} wins the game of *security against verifiers* with a non-negligible probability, $\bar{\mathcal{B}}$'s advantage against its CCA2 challenger is also non-negligible. Hence our OFERS protocol is secure against verifiers if the underlying encryption scheme \mathbf{EN} is CCA2-secure.

SECURITY AGAINST THE ARBITRATOR: Suppose an adversarial arbitrator \mathcal{C} breaks the security against the arbitrator in the proposed OFERS protocol. We construct a forger $\bar{\mathcal{C}}$ for Abe *et al.*'s ring signature scheme $\mathbf{RS} = (\mathbf{RKG}, \mathbf{RSig}, \mathbf{RVer})$ with access to a signing oracle O_{RSig} .

For the initial public-key list L given to the forger $\bar{\mathcal{C}}$, the adversarial arbitrator \mathcal{C} takes L as input and then outputs (ASK^*, APK) , where APK is set as the arbitrator's public key for Camenisch-Shoup encryption scheme, and ASK^* is the state information which may not correspond to APK . (ASK^*, APK, L) is the input of the OFERS protocol. After that, \mathcal{C} begins to ask queries to the partial ring signature signing oracle O_{PRSig} , for which the responses can be perfectly simulated by $\bar{\mathcal{C}}$ using O_{RSig} : For any message m_i and any sublist $L'' \subseteq L$, $\bar{\mathcal{C}}$ asks its signing oracle O_{RSig} to get a ring signature σ_i , then encrypts σ_i under APK to get a VERS θ_i and generates the NIZK proof π_i . Finally, \mathcal{C} outputs the forgery (m', σ') such that $\mathbf{RVer}(m', L', \sigma') = 1$ and $(m', L') \notin \text{Query}(\mathcal{C}, O_{PRSig})$, which means $\bar{\mathcal{C}}$ never asks O_{RSig} to response a valid ring signature on m' w.r.t L' . In our OFERS protocol, σ' is just the conventional ring signature on m' w.r.t L' , so $\bar{\mathcal{C}}$ has succeeded for obtaining σ' as the forgery of the message m' without asking the signing oracle O_{RSig} . It is contradictory to the existential unforgeability of Abe *et al.*'s ring signature scheme against adaptive chosen message and chosen public-key attacks. Hence our OFERS protocol must be secure against the arbitrator.

SIGNER AMBIGUITY: Suppose that our OFERS protocol does not meet signer ambiguity, which means that there is an unbound adversary \mathcal{D} can tell which private key SK_s was used to produce a given tuple $(m, L, \theta, \sigma, APK)$ with the probability not equal to $1/|L|$. Then, from \mathcal{D} we now construct an adversary $\bar{\mathcal{D}}$ that breaks signer ambiguity of Abe *et al.*'s ring signature scheme, which thus leads to a contradiction. For a given initial public-key list L in Abe *et al.*'s scheme we run the key generation algorithm of Camenisch-Shoup encryption scheme to get the arbitrator's key pair (ASK, APK) . For a target (m, L, σ, APK) , $\bar{\mathcal{D}}$ runs \mathbf{PRSig} algorithm to get θ , i.e. $\theta \leftarrow \mathbf{PRSig}(m, L, \sigma, APK)$. By forwarding $(m, L, \theta, \sigma, APK)$ to \mathcal{D} ,

$\bar{\mathcal{D}}$ just outputs the index returned by \mathcal{D} as its guess which private key was used to issue (m, L, σ, APK) . It is easy to see that $\bar{\mathcal{D}}$ breaks the signer ambiguity of Abe *et al.*'s ring signature scheme with the exact same probability as \mathcal{D} breaks the signer ambiguity of our OFERS protocol. \square

Remark 2. In the proofs above, we do not give the specific details about the underlying (Abe *et al.*'s) ring signature scheme and (Camenisch and Shoup's) encryption scheme, as our construction (specified in Section 3.4) can be extended to a generic scheme, i.e., based on any secure ring signature scheme and encryption scheme, the associated proofs can be obtained by simply adapting the proofs above. In addition, from our proofs we can see that a secure ring signature scheme with signer ambiguity does not necessarily guarantee an OFERS protocol preserving the same property. The counterexample is very simple: just modify our OFERS protocol such that the VERS θ includes a public key PK_i which indicates that the private key SK_i was used to issue the corresponding ring signature σ . For this scheme, it is not difficult to see that the proofs for the first three properties still hold, but not for signer ambiguity since, with the reminder PK_i , the adversary can tell with the probability 1 that SK_i was used to issue a tuple $(m, L, \theta, \sigma, APK)$.

3.6 Summary

In this chapter, for achieving better privacy in optimistic fair exchange, we present the first solution of optimistic fair exchange of ring signatures (OFERS) by first formally defining its security model in the multi-user setting under adaptive chosen message, chosen-key and chosen public-key attacks. We have also proposed a concrete scheme of verifiably encrypted ring signature (VERS) and used it to build an optimistic fair exchange protocol. The proposed scheme is proven to be *secure against signers, verifiers and the arbitrator* and satisfy the property *signer ambiguity* under our security definitions.

It should be noted that the proposed OFERS scheme does not satisfy the important property *abuse-freeness*. That is because, once a VERS θ is correctly issued by the signer, θ can be verified by any outsider of the protocol since the non-interactive zero-knowledge proof showing the validity of VERS can be universally verified. This flaw might benefit some malicious parties in fair exchange. Furthermore, in order to give an efficient zero-knowledge proof, only Abe *et al.*'s ring signature scheme in

all discrete-log case is applied in the proposed OFERS scheme. Actually, by some suitable modifications, the generic scheme of Abe *et al.*'s ring signatures can also be efficiently employed in our construction of OFERS. The further solutions on these two issues will be given in Chapter 4.

Abuse-free Optimistic Fair Exchange of Ring Signatures

In this chapter, the notion of *abuse-free optimistic fair exchange of ring signatures* (AOFERS) is first proposed, together with the formal security model of *abuse-freeness* in OFERS. By updating the proposed OFERS scheme introduced in Chapter 3, we propose two efficient and concrete schemes of AOFERS based on all discrete-log ring signature scheme and the generic scheme of Abe *et al.*'s ring signatures respectively. Finally, we formally prove that the proposed AOFERS schemes are perfectly *abuse-free* in our security model.

4.1 Introduction

In recent studies, the property *abuse-freeness* is highly desired in optimistic fair exchange in order to prevent a honest party from being exploited to benefit some dishonest party. Intuitively, in an abuse-free fair exchange protocol, any interim output of the protocol cannot be accepted as the valid evidence showing that some participant has committed to sign the message before the protocol is executed successfully. In contrast, an OFE protocol without *abuse-freeness* may give a dishonest party some advantages by showing the universally verifiable interim results to an outsider of the protocol. There are only few works on this topic, In [GJM99], Garay *et al.* proposed an abuse-free optimistic contract signing protocol based on the technology of *private contract signatures*. Wang [Wan10] proposed an abuse-free fair contract-signing protocol based on the RSA signature, in which an interactive protocol is employed to prove the validity of the RSA-based undeniable signature [GRK00]. In [HYWS08a], Huang *et al.* introduced *ambiguous optimistic fair exchange*, in which an outsider cannot tell which of a signer and a verifier issues a given partial signature, that is, the partial signature is indistinguishable between

the signer and the verifier so that the verifier cannot convince anybody about the authorship of the partial signature. Huang *et al.*'s scheme is not perfectly *abuse-free* since an outsider can be certain of that the real producer of the partial signature must be either the signer or the verifier. Except these two, no one can issue such a partial signature.

In this chapter, in order to meet the property *abuse-freeness*, we follow Wang's idea in [Wan10] to construct *abuse-free optimistic fair exchange of ring signatures* based on an interactive zero-knowledge proof. We first give the formal definition of AOFERS and define the security model of *abuse-freeness* in the multi-user setting under adaptive chosen message, chosen-key and chosen public-key attacks. Then we propose a concrete scheme of AOFERS based on Abe *et al.*'s all discrete-log ring signature scheme. An interactive zero-knowledge proof, which is secure against the on-line attack introduced in [Wan10], is applied in the construction of our scheme. Moreover, we present a solution to employ the generic scheme of Abe *et al.*'s ring signatures [AOS02] in our AOFERS scheme. Finally, we give the formal proof to show that the proposed AOFERS scheme is perfectly *abuse-free* in our security model.

4.2 Security Definitions

By updating the security definitions of OFERS in Section 3.3, we define the notion of *abuse-free optimistic fair exchange of ring signatures* (AOFERS) in the following way:

Definition 4.1. (Syntax) *abuse-free optimistic fair exchange of ring signatures* (AOFERS) consists of nine probabilistic polynomial-time algorithms and a zero-knowledge proof¹.

- **Setup^{TTP}**: On input a security parameter κ , the algorithm outputs a public-private arbitration key pair (APK, ASK) and some auxiliary information if necessary.
- **Setup^{User}**: On input κ and (optionally) the arbitrator's public key APK , the algorithm outputs public-private key pairs (PK_i, SK_i) for each user in the

¹In concrete schemes, the zero-knowledge proof for showing the validity of partial ring signatures is executed in the form of an interactive protocol.

ring. These public keys form a public-key list L .

- **RSig**($\mathbf{m}, L, \mathbf{SK}_s$): A signer U_s in the ring executes the algorithm by inputting a message m , a public-keys list L including PK_s and its corresponding private key SK_s , then outputs a ring signature σ .
- **RVer**(\mathbf{m}, L, σ): On input a message m , a ring signature σ on m under a public-key list L , a verifier executes the algorithm to output either 1 or 0, which means accept or reject respectively.
- **PRGen**($\mathbf{m}, L, \mathbf{SK}_s, \mathbf{APK}$): On input a message m , a public-key list L including PK_s , the signer U_s 's private key SK_s and the arbitrator's public key APK , the algorithm outputs a partial ring signature θ with the state information ς ².
- **PRP** ($\mathbf{m}, L, \theta, \varsigma, \mathbf{APK}$): On input a message m , a signer's public-key list L , the arbitrator's public key APK , a partial ring signature θ on m under L with the state information ς , the zero-knowledge proof between signers and verifiers outputs the evidence π to show that the partial ring signature θ can be converted to a full ring signature σ by the arbitrator.
- **PRVer**($\mathbf{m}, L, \theta, \pi, \mathbf{APK}$): On input a message m , a signer's public-key list L , a partial ring signature θ on m under L , the evidence π generated by a zero-knowledge proof between signers and verifiers and the arbitrator's public key APK , the verifier executes the algorithm to output either 1 or 0, which means accept or reject respectively.
- **PRSim**($\mathbf{m}, L, \mathbf{APK}$): The algorithm perfectly simulates the outputs in **PRGen** without the signer's secret key SK_s . On input a message m , a signer's public-key list L and the arbitrator's public key APK , the algorithm outputs a simulated partial ring signature θ on m under L with the simulated state information ς .
- **PRPSim**($\mathbf{m}, L, \theta, \mathbf{APK}$): The algorithm perfectly simulates the outputs in **PRP**. On input a message m , a signer's public-key list L , a partial ring signature θ and the arbitrator's public key APK , the algorithm outputs the simulated evidence π of the zero-knowledge proof.

² ς contains some necessary state information used in the following zero-knowledge proof.

- **Res**($m, L, \theta, \mathbf{ASK}$): The resolution algorithm is executed by the arbitrator if the verifier does not receive the original ring signature σ from the signer ring, but has obtained the corresponding partial ring signature θ on the message m under the public-key list L . After the verifier has fulfilled its obligation to the signer, the arbitrator converts the partial ring signature θ to its corresponding full ring signature σ using its private key \mathbf{ASK} . If σ is valid, the arbitrator sends it to the verifier ring, otherwise sends an error message.

Abuse-freeness: The property *abuse-freeness* requires that, in a fair exchange protocol, with non-negligible probability any probabilistic polynomial-time (PPT) adversary \mathcal{D} should not be able to convince an outside party that the signer has committed to sign the message by showing any intermediate result at any point of the fair exchange protocol. The property is formally defined by the following game:

$$\begin{aligned}
\mathbf{Setup}^{\mathbf{TTP}}(\kappa) &\longrightarrow (ASK, APK) \\
(m, L, \Delta) &\longleftarrow \mathcal{D}^{O_{Res}}(APK) \\
\mathbf{PRGen}(m, L', SK_s, APK) &\longrightarrow (\theta_1, \varsigma_1) \\
\mathbf{PRP}(m, L', \theta_1, \varsigma_1, APK) &\longrightarrow \pi_1 \\
\mathbf{PRSim}(m, L', APK) &\longrightarrow (\theta_0, \varsigma_0) \\
\mathbf{PRPSim}(m, L', \theta_0, APK) &\longrightarrow \pi_0 \\
(0, 1) &\longrightarrow b \\
(\theta_b, \pi_b) &\longrightarrow (\theta^*, \pi^*) \\
b' &\longleftarrow \mathcal{D}^{O_{Res}}(\Delta, \theta^*, \pi^*, APK) \\
\text{Success of } \mathcal{D} &= [b = b' \wedge (m, L', \theta^*) \notin \text{Query}(\mathcal{D}, O_{Res})]
\end{aligned}$$

where Δ is the state information of \mathcal{D} , L is a public-key list generated by \mathcal{D} , L' is a sublist of L , O_{Res} is the resolution oracle, and $\text{Query}(\mathcal{D}, O_{Res})$ is the set of valid queries which \mathcal{D} asks to O_{Res} . In the oracle queries, the adversary \mathcal{D} is allowed to ask O_{Res} for resolving any partial ring signature except θ^* with respect to any message and any public list. And \mathcal{D} may not know the corresponding private keys of such a public list. However, for the challenge public-key list L' , it is required that \mathcal{D} must prove that he/she knows the corresponding private keys, and all the public-private key pairs are generated by the algorithm $\mathbf{Setup}^{\mathbf{User}}$.

Definition 4.2 (Abuse-freeness) *Abuse-free optimistic fair exchange of ring signatures is said to be abuse-free if there is no PPT adversary \mathcal{D} who wins the game*

above with non-negligible probability.

Remark 3. Essentially, *abuse-freeness* depends on whether there exists a simulator which can perfectly simulate all the intermediate outputs in an OFE protocol. In [HYWS08a], Huang *et al.* proposed the notion of *ambiguous optimistic fair exchange*, in which the verifier cannot convince any outsider about the authorship of a partial signature issued by the signer since the partial signature can be indistinguishably generated by the signer or the verifier. In some cases, ambiguous fair exchange and abuse-free fair exchange can solve the same problem. However, ambiguous fair exchange is not equivalent to abuse-free fair exchange. That is because, in ambiguous fair exchange, the signer’s intermediate outputs, i.e. the partial signature, can only be simulated by the verifier using its secret key. But in abuse-free fair exchange, the signer’s intermediate results can be perfectly simulated by anyone. In our construction of AOFERS, an interactive zero-knowledge proof is used for showing the validity of the signer’s partial signature. That means only the verifier who is involved in the interactive proof can be convinced by the signer. And after executing the protocol, any intermediate result will become useless since there exists an efficient simulation algorithm which can simulate all these results perfectly. Hence, in the partial ring signature verification algorithm **PRVer**, only the evidence π is not enough to convince the verifier the validity of the partial ring signature θ . It is necessary for the verifier to ensure that π is the correct outputs of the interactive zero-knowledge proof **PRP** with the signer.

4.3 The Proposed Scheme

In this section, we first present how to build a concrete AOFERS scheme based on Abe *et al.*’s ring signatures in all discrete-log case. Then we propose a solution, by which the generic scheme of Abe *et al.*’s ring signatures can be applied in our AOFERS scheme.

4.3.1 All Discrete-log Case

In our AOFERS scheme, an interactive zero-knowledge proof is employed instead of the non-interactive proof in the normal OFERS scheme in Chapter 3. In order to meet the property *abuse-freeness*, it is required that the verifier cannot convince

an outsider that the proof is just given by the prover via showing any intermediate output of the proof. Due to the property *zero-knowledge* of the proof, the distribution of the outputs in a simulated proof is (computationally) indistinguishable from that in a real proof. Concretely, in order to simulate the outputs in Camenisch and Michels' zero-knowledge proof introduced in Section 3.2.2, the verifier can randomly select s_1, s_2 and c in the corresponding intervals, and then compute $t_1 = g_1^{s_1} y_1^c$, $t_2 = g_2^{s_2} y_2^c$ and $t_3 = h_1^{s_1} h_2^{s_2} \tilde{y}^c$ instead. Obviously, any intermediate output of the proof can be perfectly simulated by such an efficient algorithm. However, in [Wan10], Wang described an on-line attack, in which, by colluding with an outsider during the execution of the interactive zero-knowledge proof, an adversarial verifier can successfully convince the outsider that the proof is given by the prover. In such an on-line attack, after receiving the prover's commitments t_1, t_2 and t_3 in Step 1, the verifier can show all these commitments to the outsider and ask him/her to generate a challenge c instead in Step 2. Finally the outsider can be convinced by the right responses s_1 and s_2 from the prover in Step 3 since the commitments t_1, t_2 and t_3 he/she has seen cannot be changed by the verifier.

In order to protect against such an on-line attack, Wang proposed an idea based on trapdoor commitment schemes in [Wan10]. Generally, a common commitment scheme [BCC88, Ped91] only has two properties *hiding* and *binding*: *hiding* means no one can know which value is committed in a given commitment except its producer, and *binding* guarantees that, once a commitment has been made, the committed value in a given commitment cannot be changed. Moreover, in a trapdoor commitment scheme [Gen04, MY04], there is a trapdoor by which a trapdoor commitment can be opened in different ways, but without the trapdoor, the property *binding* is still guaranteed.

Hence, by a trapdoor commitment scheme, Camenisch and Michels' interactive zero-knowledge proof can be modified for *abuse-freeness* in Figure 4.1. Before the interactive zero-knowledge proof is executed, the verifier must publish his own trapdoor commitment scheme denoted by $TCom()$. During the execution of the proof, instead of directly sending t_1, t_2 and t_3 to the verifier in Step 1, the prover first uses the verifier's trapdoor commitment scheme to compute $\bar{t}_1 = TCom(t_1, t'_1)$, $\bar{t}_2 = TCom(t_2, t'_2)$ and $\bar{t}_3 = TCom(t_3, t'_3)$ by randomly selecting t'_1, t'_2 and t'_3 , then sends \bar{t}_1, \bar{t}_2 and \bar{t}_3 to the verifier. After receiving the challenge c in Step 2, the prover sends the responses s_1 and s_2 together with $t_1, t'_1, t_2, t'_2, t_3, t'_3$ to the verifier in Step 3. The verifier first checks whether the trapdoor commitments $\bar{t}_1 = TCom(t_1, t'_1)$, $\bar{t}_2 =$

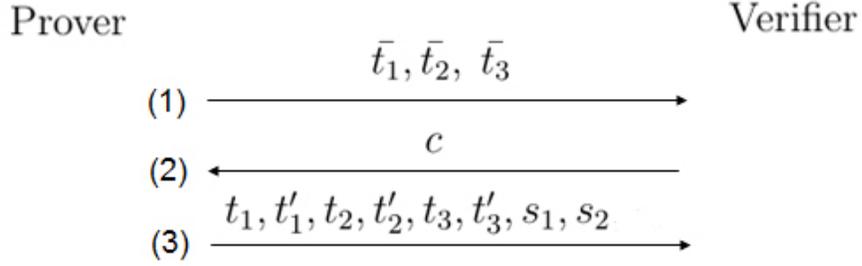


Figure 4.1: The modified interactive zero-knowledge proof for abuse-freeness

$TCom(t_2, t'_2)$ and $\bar{t}_3 = TCom(t_3, t'_3)$ are prepared correctly, and then checks whether s_1 and s_2 are the right responses to t_1, t_2 and t_3 .

In this improved interactive zero-knowledge proof, the outsider cannot be convinced anymore by the on-line attack. That is because, due to the verifier's trapdoor commitment scheme used in Step 1, the verifier cannot know the prover's answers t_1, t_2 and t_3 , which are committed in the trapdoor commitments \bar{t}_1, \bar{t}_2 and \bar{t}_3 , but can open the trapdoor commitments in a different way. That means, even if the verifier shows all these trapdoor commitments to the outsider for a challenge c in Step 2, the outsider cannot believe that t_1, t_2 and t_3 received in Step 3 are just generated by the prover since the outsider cannot distinguish whether t_1, t_2 and t_3 are prepared by the prover or forged by the verifier using the trapdoor. Hence the on-line attack does not work in this improved interactive zero-knowledge proof.

The construction of the concrete AOFERS scheme based on Abe *et al.*'s ring signatures in all discrete-log case is similar to the normal OFERS scheme in Chapter 3 except that the non-interactive zero-knowledge proof is replaced by the interactive proof introduced above, and both rings must publish their own trapdoor commitment scheme denoted by $TCom_I()$ and $TCom_J()$ in advance. The algorithms **Setup^{TTP}**, **Setup^{User}**, **RSign** and **RVer** in the proposed AOFERS scheme are the same as that in the normal OFERS scheme. The other algorithms and the interactive zero-knowledge proof are shown as follows:

PRGen: The algorithm is used for converting a full ring signature σ_I to an encrypted ring signature θ_I . Let $\mathbf{L} = m || L_I$ be the public label under Camenisch and

Shoup's encryption scheme. Compute $w = g_{n_I-1}^{s_{n_I-1}}$ and encrypt s_{n_I-1} by computing

$$\mathbf{u} = \mathbf{g}^t, \quad \mathbf{e} = \mathbf{y}_1^t \mathbf{h}^{s_{n_I-1}}, \quad \mathbf{v} = \text{abs}(\mathbf{y}_2 \mathbf{y}_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^t$$

Then the signer sends the encrypted ring signature $\theta_I = (c_0, s_0, \dots, s_{n_I-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v})$ to the verifier ring.

PRP: For convincing the verifier that the signer's full ring signature has been correctly encrypted under the arbitration key, an interactive zero-knowledge proof is executed between the signer ring and the verifier ring.

1. The signer randomly selects $r \in \mathbb{Z}_n$, $r_1 \in (-2^{l-2}, 2^{l-2})$, $r_2 \in (-(n2^\eta)^\epsilon, (n2^\eta)^\epsilon)$ and $r_3 \in (-(n2^\eta)^\epsilon, (n2^\eta)^\epsilon)$, and computes $\hat{w} = h_1^r h_2^{s_{n_I-1}} \bmod n$, $t_1 = g_{n_I-1}^{r_1}$, $t_2 = h_1^{r_2} h_2^{r_1}$, $\mathbf{u}' = \mathbf{g}^{r_3}$, $\mathbf{e}' = \mathbf{y}_1^{r_3} \mathbf{h}^{r_1}$ and $\mathbf{v}' = (\mathbf{y}_2 \mathbf{y}_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^{r_3}$ in their own groups respectively. Then the signer uses the verifier ring's trapdoor commitment scheme to compute $\bar{w} = TCom_J(\hat{w}, \hat{w}')$, $\bar{t}_1 = TCom_J(t_1, t_1')$, $\bar{t}_2 = TCom_J(t_2, t_2')$, $\bar{\mathbf{u}}' = TCom_J(\mathbf{u}', \mathbf{u}'')$, $\bar{\mathbf{e}}' = TCom_J(\mathbf{e}', \mathbf{e}'')$ and $\bar{\mathbf{v}}' = TCom_J(\mathbf{v}', \mathbf{v}'')$ by randomly selecting $\hat{w}', t_1', t_2', \mathbf{u}'', \mathbf{e}'', \mathbf{v}''$, and sends all the trapdoor commitments to the verifier ring.
2. The verifier sends a challenge $\hat{c} \in_R \{0, 1\}^\eta$ back to the signer ring.
3. The signers computes the responses $v_1 = r_1 - \hat{c}s_{n_I-1}$, $v_2 = r_2 - \hat{c}r$, $v_3 = r_3 - \hat{c}t$ in \mathbb{Z} and sends $\hat{w}, t_1, t_2, \mathbf{u}', \mathbf{e}', \mathbf{v}', \hat{w}', t_1', t_2', \mathbf{u}'', \mathbf{e}'', \mathbf{v}''$ with the responses v_1, v_2, v_3 to the verifier ring.

PRVer: The verifier first checks whether all the trapdoor commitments are correctly prepared, and then $t_1 = g_{n_I-1}^{v_1} w^{\hat{c}}$, $t_2 = h_1^{v_2} h_2^{v_1} \hat{w}^{\hat{c}}$, $\mathbf{u}'^2 = \mathbf{g}^{2v_3} \mathbf{u}^{2\hat{c}}$, $\mathbf{e}'^2 = \mathbf{y}_1^{2v_3} \mathbf{h}^{2v_1} \mathbf{e}^{2\hat{c}}$, $\mathbf{v}'^2 = (\mathbf{y}_2 \mathbf{y}_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^{2v_3} \mathbf{v}^{2\hat{c}}$ and $-2^{l-1} < v_1 < 2^{l-1}$. If any condition does not hold, outputs the encrypted ring signature θ_I is invalid, otherwise computes $e_i = g_i^{s_i} y_i^{c_i}$ for $i = 0, \dots, n_I-2$ and $e_{n_I-1} = w y_{n_I-1}^{c_{n_I-1}}$, then computes $c_{i+1} = H_{i+1}(L_I, m, e_i)$ if $i \neq n_I-1$. If $c_0 = H_0(L_I, m, e_{n_I-1})$, the verifier accepts θ_I , reject otherwise.

PRSim: The algorithm outputs a simulated encrypted ring signature by the following steps:

1. Randomly select $e_{n_I-1} \in \mathbb{Z}_{p_{n_I-1}}$, and compute $c_0 = H_0(L_I, m, e_{n_I-1})$.

2. Randomly select $s_i \in (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$. For $i = 0, \dots, n_I - 2$, compute $e_i = g_i^{s_i} y_i^{c_i} \bmod p_i$ and $c_{i+1} = H_{i+1}(L_I, m, e_i)$. At last compute $w = e_{n_I-1} y_{n_I-1}^{-c_{n_I-1}} \bmod p_{n_I-1}$.
3. Randomly select $s' \in (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$, and compute the ciphertext

$$\mathbf{u} = \mathbf{g}^t, \quad \mathbf{e} = y_1^t \mathbf{h}^{s'}, \quad \mathbf{v} = \text{abs}(y_2 y_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^t$$

Then the simulated encrypted ring signature is $\theta_I = (c_0, s_0, \dots, s_{n_I-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v})$.

PRPSim: The algorithm simulates the outputs in the interactive zero-knowledge proof shown in **PRP**. Randomly select $\hat{w} \in \mathbb{Z}_n, v_1 \in (-2^{l-2}, 2^{l-2}), v_2 \in (-(n2^\eta)^\epsilon, (n2^\eta)^\epsilon), v_3 \in (-(n2^\eta)^\epsilon, (n2^\eta)^\epsilon)$ and $\hat{c} \in_R \{0, 1\}^\eta$, then compute $t_1 = g_{n_I-1}^{v_1} w^{\hat{c}}, t_2 = h_1^{v_2} h_2^{v_1} \hat{w}^{\hat{c}}, \mathbf{u}' = \mathbf{g}^{v_3} \mathbf{u}^{\hat{c}}, \mathbf{e}' = y_1^{v_3} \mathbf{h}^{v_1} \mathbf{e}^{\hat{c}}, \mathbf{v}' = (y_2 y_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^{v_3} \mathbf{v}^{\hat{c}}$ in their own groups respectively, then computes their trapdoor commitments $\bar{w} = TCom_J(\hat{w}, \hat{w}'), \bar{t}_1 = TCom_J(t_1, t_1'), \bar{t}_2 = TCom_J(t_2, t_2'), \bar{\mathbf{u}}' = TCom_J(\mathbf{u}', \mathbf{u}''), \bar{\mathbf{e}}' = TCom_J(\mathbf{e}', \mathbf{e}'')$ and $\bar{\mathbf{v}}' = TCom_J(\mathbf{v}', \mathbf{v}'')$ by randomly selecting $\hat{w}', t_1', t_2', \mathbf{u}'', \mathbf{e}'', \mathbf{v}''$. The simulated outputs of the proof are $(\hat{w}, t_1, t_2, \mathbf{u}', \mathbf{e}', \mathbf{v}', \bar{w}, \bar{t}_1, \bar{t}_2, \bar{\mathbf{u}}', \bar{\mathbf{e}}', \bar{\mathbf{v}}', \hat{w}', t_1', t_2', \mathbf{u}'', \mathbf{e}'', \mathbf{v}'', \hat{c}, v_1, v_2, v_3)$.

Res: After the verifier shows a proof that he has fulfilled his obligation to the signer, the arbitrator decrypts the ciphertext $(\mathbf{u}, \mathbf{e}, \mathbf{v})$ using its secret key (x_1, x_2, x_3) to extract s_{n_I-1} and obtains the full ring signature σ_I . If σ_I is valid, the arbitrator sends it to the verifier ring, otherwise sends an error message.

4.3.2 Generic Case

In Abe *et al.*'s generic scheme of ring signatures shown in Section 3.2.1, a user in a ring is allowed to arbitrarily select any signature scheme, which is required to belong to either type-T or type-H, as its own signature scheme for ring signatures. And the signature schemes which are neither type-T nor type-H can be considered *compatible* with some type-T or type-H schemes in some cases. Hence Abe *et al.*'s ring signatures are constructed in not only all discrete-log case or all RSA case but also in some mixture cases. In the generation and verification of a ring signature in mixture case, a signer must follow the generic scheme to compute each link of the ring signature using the corresponding signature scheme of each user. And such a mixture ring signature is still proven existentially unforgeable against adaptive chosen message and chosen public-key attacks [AOS02].

In our AOEFERS scheme shown in Section 4.3.1, we only consider the ring signature scheme of all discrete-log case. In fact, for any mixture type of Abe *et al.*'s ring signatures, if the last user U_{n-1} formed in the public-key list uses a public-private key pair in the discrete-log case, the proposed AOEFERS scheme can still work in the same way. That is because, in our AOFERS scheme, only the last link of a full ring signature, i.e. s_{n-1} , is involved in the generation and verification of a partial ring signature no matter what signature schemes are applied in the rest parts of the full ring signature. However, in all RSA case or some mixture cases without discrete-log keys, the proposed AOFERS scheme will not work anymore. To solve this dilemma, we propose an idea that by adding an extra discrete-log public key to the end of the initial public-key list, we can easily form a new ring with a discrete-log key at the end. And in order to prevent anyone from forging a ring signature using this extra public key, we require that any party including the arbitrator cannot know its corresponding private key. One solution to this end is that the arbitrator publishes a secure hash function $h : \{0, 1\}^* \rightarrow \langle g' \rangle$, where $\langle g' \rangle$ is the same type of group in all discrete-log ring signature scheme in Section 3.2.1. h takes some public information, such as the initial public-key list, the message or the expiry time, as input, and outputs an integer y' in the public key domain of $\langle g' \rangle$, where (g', y', p', q') can be used as the extra public key. Then the signer adds the new public key with an unknown private key to the end of the initial public-key list and uses the new public-key list in the following fair exchange. Based on this idea, the generic scheme of Abe *et al.*'s ring signatures can be easily employed in our AOFERS scheme.

Setup^{TTP}: The setup of the arbitrator is similar with **Setup^{TTP}** in Section 4.3.1 except that the arbitrator additionally publishes a secure hash function $h : \{0, 1\}^* \rightarrow \langle g' \rangle$, where the order of $\langle g' \rangle$ is $q' > 2^{l+1}$.

Setup^{User}: In the setup of users, each user can select his/her own type-H or type-T signature scheme and generates the public-private key pair. All these public keys form a public-key list L .

RSign: The signer U_k in a ring signs a message m by executing the algorithm below:

1. Compute $h(L, m, aux) = y'$, where aux is auxiliary information, then add (g', y', p', q') to the end of L to form a new public-key list L' , where n denotes

the size of L' .

2. (Initialization): Similar with the generic scheme of ring signatures in Section 3.2.1, compute

$$e_k = \begin{cases} A_k(sk_k; \alpha) & , (\text{type-T}), \text{ or} \\ \beta & , (\text{type-H}), \end{cases}$$

then compute $c_{k+1} = H_{k+1}(L', m, e_k)$.

3. (Forward sequence): For $i = k + 1, \dots, n - 1, 0, \dots, k - 1$, compute

$$e_i = \begin{cases} V_i(s_i, c_i, vk_i) & , (\text{type-T}), \text{ or} \\ c_i + F_i(s_i, vk_i) & , (\text{type-H}), \end{cases}$$

then compute $c_{i+1} = H_{i+1}(L', m, e_i)$. If $i = n - 1$, compute $e_{n-1} = g^{s_{n-1}}y^{c_{n-1}} \bmod p'$ instead, where $s_{n-1} \in (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$.

4. (Forming the ring): Compute

$$s_k = \begin{cases} Z_k(sk_k, \alpha, c_k) & , (\text{type-T}), \text{ or} \\ I_k(\beta - c_k, sk_k) & , (\text{type-H}). \end{cases}$$

The resulting ring signature σ is $(c_0, s_0, \dots, s_{n-1})$.

RVer: The verifier first checks whether L' is correctly generated from L . For $i = 0, \dots, n - 2$, compute

$$e_i = \begin{cases} V_i(s_i, c_i, vk_i) & , (\text{type-T}), \text{ or} \\ c_i + F_i(s_i, vk_i) & , (\text{type-H}), \end{cases}$$

and $e_{n-1} = g^{s_{n-1}}y^{c_{n-1}} \bmod p'$, then compute $c_{i+1} = H_{i+1}(L', m, e_i)$ if $i \neq n - 1$. If $c_0 = H_0(L', m, e_{n-1})$, the verifier accepts σ as a valid ring signature, reject otherwise.

The algorithm **PRGen** and the interactive zero-knowledge proof **PRP** are the same as that in Section 4.3.1, where $(g_{n-1}, y_{n-1}, p_{n-1}, q_{n-1})$ is equivalent to (g', y', p', q') here.

PRVer: The verifier first checks whether L' is valid and whether the interactive zero-knowledge proof is correctly executed. After that, for $i = 0, \dots, n - 2$, compute

$$e_i = \begin{cases} V_i(s_i, c_i, vk_i) & , (\text{type-T}), \text{ or} \\ c_i + F_i(s_i, vk_i) & , (\text{type-H}). \end{cases}$$

and $e_{n-1} = wy'^{c_{n-1}} \bmod p'$, then compute $c_{i+1} = H_{i+1}(L', m, e_i)$ if $i \neq n - 1$. If $c_0 = H_0(L', m, e_{n-1})$, the verifier accepts the partial ring signature θ , reject otherwise.

PRSim: The algorithm outputs a simulated encrypted ring signature by the following steps:

1. Randomly select $e_{n-1} \in \mathbb{Z}_{p'}$, and compute $c_0 = H_0(L', m, e_{n-1})$.
2. For $i = 0, \dots, n - 2$, compute

$$e_i = \begin{cases} V_i(s_i, c_i, vk_i) & , (\text{type-T}), \text{ or} \\ c_i + F_i(s_i, vk_i) & , (\text{type-H}), \end{cases}$$

then compute $c_{i+1} = H_{i+1}(L', m, e_i)$. At last, compute $w = e_{n-1}y'^{-c_{n-1}} \bmod p'$.

3. Randomly select $s' \in (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$, and compute the ciphertext

$$\mathbf{u} = \mathbf{g}^t, \quad \mathbf{e} = y_1^t h^{s'}, \quad \mathbf{v} = \text{abs}(y_2 y_3^{\text{H}(\mathbf{u}, \mathbf{e}, \mathbf{L})})^t$$

Then the simulated encrypted ring signature is $\theta = (c_0, s_0, \dots, s_{n-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v})$.

The algorithms **PRPSim** and **Res** are the same as that in Section 4.3.1.

Remark 4. In the partial ring signature simulation algorithm **PRSim**, no matter what types of public keys used in the public-key list, there always exists such an efficient simulation algorithm. That is because a partial ring signature $\theta = (c_0, s_0, \dots, s_{n-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v})$ consists of two parts. The first part $(c_0, s_0, \dots, s_{n-2}, w)$ is a ‘ring signature’ with a hidden value s_{n-1} in $w = g'^{s_{n-1}}$, and the second part $(\mathbf{u}, \mathbf{e}, \mathbf{v})$ is the ciphertext of s_{n-1} encrypted under the arbitrator’s public key. In **PRSim**, after first randomly selecting the last e_i , i.e. e_{n-1} , the simulation of the ring signature part $(c_0, s_0, s_1, \dots, s_{n-2}, w)$ strictly follows the ring signature verification algorithm to compute each e_i and c_i , then outputs $w = e_{n-1}y'^{-c_{n-1}} \bmod p'$ at last. Obviously, for such a w , there must exist a s_{n-1} such that $w = g'^{s_{n-1}} = e_{n-1}y'^{-c_{n-1}} \bmod p'$, but

the s_{n-1} is unknown since computing a discrete logarithm is hard in the group $\langle g' \rangle$. Hence the only difference between a normal and a simulated partial ring signature is that, in a simulated partial ring signature, the hidden value s_{n-1} in w is unknown and equal to s' encrypted in the ciphertext with negligible probability.

4.4 Security Analysis

In this session, we prove that the proposed AOFERS schemes are perfectly *abuse-free* in our security model. Obviously if the AOFERS scheme based on the generic scheme of ring signatures is proven abuse-free, the scheme based on all discrete-log ring signatures is also abuse-free. Let $\mathbf{RS} = (\mathbf{RKG}, \mathbf{RSig}, \mathbf{RVer})$ denote the generic scheme of Abe *et al.*'s ring signatures, $\mathbf{EN} = (\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ denote Camenisch and Shoup's public-key encryption scheme, and (\mathbf{P}, \mathbf{V}) denote the improved interactive zero-knowledge proof system introduced in Section 4.3.1, where π is the outputs of the proof. We have the following theorem:

Theorem 2 *The proposed abuse-free optimistic fair exchange of ring signatures is perfectly abuse-free, i.e., satisfies Definition 4.2, if the underlying \mathbf{EN} is secure against adaptive chosen ciphertext (CCA2) attacks, and (\mathbf{P}, \mathbf{V}) is a simulation-sound interactive zero-knowledge proof system.*

Proof. ABUSE-FREENESS: Suppose a PPT adversary \mathcal{D} breaks **abuse-freeness** in the proposed AOFERS scheme based on Abe *et al.*'s generic scheme of ring signatures. We construct a distinguisher $\bar{\mathcal{D}}$ who can successfully distinguish the encryption of two messages with the same length of its choice from a challenger in a CCA2 game for Camenisch and Shoup's encryption scheme with non-negligible probability. For the given target encryption scheme \mathbf{EN} with the public key APK , the distinguisher $\bar{\mathcal{D}}$ first sends APK to \mathcal{D} as the arbitrator's public key. On input APK , the adversary \mathcal{D} can arbitrarily outputs public-key lists and may not know the corresponding private keys of these public-key lists. Then $\bar{\mathcal{D}}$ simulates the resolution oracle O_{Res} using its own decryption oracle O_{Des} . Given a query (m, L, θ) , where $\theta = (c_0, s_0, \dots, s_{n-2}, w, \mathbf{u}, \mathbf{e}, \mathbf{v})$, $\bar{\mathcal{D}}$ first decrypts the ciphertext $(\mathbf{u}, \mathbf{e}, \mathbf{v})$ to get s_{n-1} , then checks whether $\sigma = (c_0, s_0, \dots, s_{n-1})$ is a valid ring signature. If yes, $\bar{\mathcal{D}}$ returns σ to \mathcal{D} , otherwise returns an error message.

At some time, \mathcal{D} sends the challenge (m^*, L^*, SL^*) to $\bar{\mathcal{D}}$, where SL^* is L^* 's secret-key list. $\bar{\mathcal{D}}$ first check whether each public key PK_i in L^* matches each private key SK_i in SL^* . If not, $\bar{\mathcal{D}}$ aborts and returns a random bit, otherwise generates the targeted partial ring signature in the following steps:

1. $\bar{\mathcal{D}}$ arbitrarily selects a secret key SK_s in SL^* and produces a full ring signature $\sigma^* \leftarrow \mathbf{RSig}(m^*, L^*, SK_s)$, where $\sigma^* = (c_0^*, s_0^*, \dots, s_{n-1}^*)$.
2. Randomly select $s' \in (-2^{(l-2)/\epsilon}, 2^{(l-2)/\epsilon})$, and set $M_0 = s'$, $M_1 = s_{n-1}^*$. Then send M_0 and M_1 to its CCA2 encryption challenger. The challenger returns a ciphertext ε_b , which equals either $\mathbf{Enc}_{APK}(M_0)$ or $\mathbf{Enc}_{APK}(M_1)$.
3. $\bar{\mathcal{D}}$ executes the proof simulation algorithm \mathbf{PRPSim} to get the simulated outputs π^* , then returns $\theta^* = (c_0^*, s_0^*, \dots, s_{n-2}^*, w^*, \varepsilon_b, \pi^*)$ to \mathcal{D} .

After that, $\bar{\mathcal{D}}$ carries on simulating the resolution oracle O_{Res} as follow:

1. If the query $(m, L, \theta) \neq (m^*, L^*, \theta^*)$, $\bar{\mathcal{D}}$ simulates O_{Res} in the same way above.
2. If the query $(m, L, \theta) = (m^*, L^*, \theta^*)$, $\bar{\mathcal{D}}$ aborts the simulation and returns an error message.

So far the attack environment required by \mathcal{D} has been perfectly simulated. Note that in θ^* , the ring signature part $(c_0^*, s_0^*, \dots, s_{n-2}^*, w^*)$ can be perfectly simulated by the algorithm \mathbf{PRSim} . Concretely, we first compute e_{n-1}^* via σ^* and take this e_{n-1}^* as the input of \mathbf{PRSim} , then select the same s_i^* to compute w^* . Moreover, due to the property *zero-knowledge* of (\mathbf{P}, \mathbf{V}) , the simulated outputs generated by \mathbf{PRPSim} and the real outputs of the interactive proof \mathbf{PRP} are indistinguishable. The adversary \mathcal{D} therefore can only get a negligible advantage from the simulated outputs π^* . Finally, \mathcal{D} outputs a bit $b' = b$, if $b' = 1$, that means the message M_1 is encrypted, otherwise M_0 , then $\bar{\mathcal{D}}$ wins its CCA2 encryption game. Hence our AOFERS scheme is abuse-free if the underlying encryption scheme \mathbf{EN} is CCA2-secure. \square

Remark 5. Here, the proofs of *security against signers*, *verifiers*, *the arbitrator* and *signer ambiguity* for abuse-free OFERS can be easily achieved by adapting the proofs of non-abuse-free OFERS in Chapter 3. In our construction of fair exchange of ring signatures, the only difference between non-abuse-free and abuse-free OFERS

is whether a non-interactive or an interactive zero-knowledge proof is employed to show the validity of encrypted ring signatures. Because these two zero-knowledge proofs are almost same (both based on Camenisch and Michels' zero-knowledge proof [CM99]) but in different versions (non-interactive or interactive). These security properties satisfied in non-abuse-free OFERS can also be satisfied in abuse-free OFERS. However, some changes in the security proofs of abuse-free OFERS should claim attention.

In abuse-free OFERS, the model of *security against signers* should be modified as follow:

$$\begin{aligned}
\mathbf{Setup}^{\mathbf{TTP}}(\kappa) &\longrightarrow (ASK, APK) \\
(m, L^*, \theta, \varsigma) &\longleftarrow \mathcal{A}^{O_{Res}}(APK) \\
\pi &\longleftarrow \mathbf{PRP}(m, L^*, \theta, \varsigma, APK) \\
\sigma &\longleftarrow \mathbf{Res}(m, L^*, \theta, ASK) \\
\text{Success of } \mathcal{A} &= [\mathbf{PRVer}(m, L^*, \theta, \pi, APK)=1 \wedge \mathbf{RVer}(m, L^*, \sigma)=0]
\end{aligned}$$

Here, due to the existence of the simulation algorithms \mathbf{PRSim} and \mathbf{PRPSim} in abuse-free OFERS, we require that the partial ring signature θ issued by \mathcal{A} must be verified via the interactive zero-knowledge proof \mathbf{PRP} , and π should be the valid outputs of the proof. That is because the evidence π alone is not enough to convince the verifier the validity of the partial ring signature θ . It is necessary for the verifier to ensure that π is correctly generated via \mathbf{PRP} with the signer. The security against signers in abuse-free OFERS still depends on the property *soundness* of the interactive zero-knowledge proof system.

The models of *security against verifiers* and *security against the arbitrator* in abuse-free OFERS are the same as those of non-abuse-free OFERS in Chapter 3 except the partial ring signature signing oracle O_{PRSig} . In abuse-free OFERS, O_{PRSig} not only outputs partial ring signatures but also runs the interactive zero-knowledge proofs \mathbf{PRP} showing the validity of these partial signatures as the signer does. The adversary can ask O_{PRSig} for any partial ring signatures of his/her choosing, and execute \mathbf{PRP} with O_{PRSig} to ensure these partial signatures are valid. Likewise, in the model of *signer ambiguity* of abuse-free OFERS, the adversary can also check the validity of the partial ring signatures via \mathbf{PRP} , which allows the adversary acquiring the complete information of these partial signatures.

4.5 Summary

In this chapter, by updating the concept of *optimistic fair exchange of ring signature* (OFERS) in Chapter 3, we propose the notion of *abuse-free optimistic fair exchange of ring signatures* (AOFERS) by formally defining its security model in the multi-user setting under adaptive chosen message, chosen-key, and chosen public-key attacks. We follow Wang’s construction of abuse-free contract signing protocol [Wan10] to build an efficient and concrete AOFERS protocol based on Abe *et al.*’s ring signature scheme in all discrete-log case. Then we extend the proposed AOFERS protocol such that the generic scheme of Abe *et al.*’s ring signatures can also be employed in our AOFERS construction efficiently. Finally, we prove that the proposed AOFERS schemes are perfectly *abuse-free* in our security model.

Chapter 5

Conclusion

In this thesis, we study optimistic fair exchange of ring signatures, which aims to provide better privacy for participants in fair exchange protocols, that is, two users from two different groups can fairly exchange their ring signatures, and each user is able to sign a message on behalf of its own group anonymously by inheriting the property *signer ambiguity* of ring signatures. Such a fair exchange protocol can effectively protect customers' privacy in electronic commerce (e.g. preventing customer's trading habits from leaking).

In Chapter 1, we briefly introduce the history of fair exchange in computer networks and discuss its applications. After reviewing some related achievements in this fields, we explain why anonymity of participants is desirable when a fair exchange protocol is executed between two members from distinct groups. Then we propose an underlying solution on this issue, that is, optimistic fair exchange of ring signatures.

After introducing some background knowledge and underlying technologies referred in this thesis in Chapter 2, we propose the notion of optimistic fair exchange of ring signatures (OFERS) which allows two members from two different groups exchange their ring signatures in an equitable way with ambiguous signers for the other group. We formalize OFERS and propose a strong security model in the multi-user setting under adaptive chosen message, chosen-key, and chosen public-key attacks. By combining the technologies of ring signatures, zero-knowledge proof and public key encryption, we construct the first efficient and concrete OFERS scheme from verifiably encrypted ring signatures (VERS) based on Abe *et al.*'s all discrete-log ring signature scheme. And for adversaries in different roles, we formally prove that the proposed OFERS scheme is *secure against signers, verifiers and the arbitrator* and perfectly inherits the property *signer ambiguity* of ring signatures (Chapter 3).

In Chapter 4, we improve our OFERS scheme in order to provide the property *abuse-freeness* in our fair exchange protocol. We formally define abuse-free optimistic fair exchange of ring signatures (AOFERS) and its security model via updating the security definitions of OFERS in Chapter 3, and propose a concrete AOFERS scheme based on Abe *et al.*'s all discrete-log ring signature scheme. Furthermore, we extend our AOFERS scheme such that the generic scheme of Abe *et al.*'s ring signatures can also be employed in our AOFERS construction efficiently. By modifying Camenisch and Michels' interactive zero-knowledge proof [CM99], our AOFERS schemes are perfectly abuse-free against not only the off-line attack but also the on-line attack introduced in [Wan10].

To the best of our knowledge, this is the first work on the topic of fair exchange to present the formal security model of (abuse-free) optimistic fair exchange of ring signatures and concrete solutions with provable security. All the security proofs are achieved in a strong security model, that is, in the multi-user setting under adaptive chosen message, chosen-key and chosen public-key attacks.

Bibliography

- [AdM04] Giuseppe Ateniese and Breno de Medeiros. Identity-based chameleon hash and applications. In Ari Juels, editor, *Financial Cryptography*, volume 3110, pages 164–180, 2004.
- [ANR02] Giuseppe Ateniese and Cristina Nita-Rotaru. Stateless-recipient certified e-mail system based on verifiable encryption. In *CT-RSA*, pages 182–199, 2002.
- [AOS02] Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n signatures from a variety of keys. In *ASIACRYPT*, pages 415–432, 2002.
- [ASW97] N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. In *ACM Conference on Computer and Communications Security*, pages 7–17, 1997.
- [ASW98] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures (extended abstract). In *EUROCRYPT*, pages 591–606, 1998.
- [Ate04] Giuseppe Ateniese. Verifiable encryption of digital signatures and applications. *ACM Transactions on Information and System Security*, 7(1):1–20, 2004.
- [BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, 1988.
- [BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In *CRYPTO*, pages 26–45, 1998.

- [BF98] Colin Boyd and Ernest Foo. Off-line fair payment protocols using convertible signatures. In *ASIACRYPT*, pages 271–285, 1998.
- [BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103–112, 1988.
- [BG06] Mihir Bellare and Oded Goldreich. On probabilistic versus deterministic provers in the definition of proofs of knowledge. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(136), 2006.
- [BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT*, pages 416–432, 2003.
- [BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004.
- [Blu83] Manuel Blum. How to exchange (secret) keys (extended abstract). In *STOC*, pages 440–447, 1983.
- [BM93] Steven M. Bellovin and Michael Merritt. Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In *ACM Conference on Computer and Communications Security*, pages 244–250, 1993.
- [BOGMR90] Michael Ben-Or, Oded Goldreich, Silvio Micali, and Ronald L. Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1), 1990.
- [Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In *Public Key Cryptography*, pages 31–46, 2003.
- [Boy89] Colin Boyd. Digital multisignatures. In *Cryptography and Coding*, pages 241–246, 1989.
- [BP97] Niko Bari and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *EUROCRYPT*, pages 480–494, 1997.

- [BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
- [BWZZ04] Feng Bao, Guilin Wang, Jianying Zhou, and Huafei Zhu. Analysis and improvement of Micali’s fair contract signing protocol. In *ACISP*, pages 176–187, 2004.
- [CA89] David Chaum and Hans Van Antwerpen. Undeniable signatures. In *CRYPTO*, pages 212–216, 1989.
- [CD00] Jan Camenisch and Ivan Damgård. Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In *ASIACRYPT*, pages 331–345, 2000.
- [CDD⁺99] Ronald Cramer, Ivan Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin. Efficient multiparty computations secure against an adaptive adversary. In *EUROCRYPT*, pages 311–326, 1999.
- [CM99] Jan Camenisch and Markus Michels. Separability and efficiency for generic group signature schemes. In *CRYPTO*, pages 413–430, 1999.
- [CPS08] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. The random oracle model and the ideal cipher model are equivalent. In *CRYPTO*, pages 1–20, 2008.
- [CS98] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO*, pages 13–25, 1998.
- [CS00] Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. *ACM Trans. Inf. Syst. Secur.*, 3(3):161–185, 2000.
- [CS03] Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *CRYPTO*, pages 126–144, 2003.
- [CWLY06] Sherman S. M. Chow, Victor K.-W. Wei, Joseph K. Liu, and Tsz Hon Yuen. Ring signatures without random oracles. In *ASIACCS*, pages 297–302, 2006.

- [Dam87] Ivan Damgård. Collision free hash functions and public key signature schemes. In *EUROCRYPT*, pages 203–216, 1987.
- [DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC*, pages 542–552, 1991.
- [DGLW96] Robert H. Deng, Li Gong, Aurel A. Lazar, and Weiguang Wang. Practical protocols for certified electronic mail. *J. Network Syst. Manage.*, 4(3):279–297, 1996.
- [DH76a] Whitfield Diffie and Martin E. Hellman. Multiuser cryptographic techniques. In *AFIPS National Computer Conference*, pages 109–112, 1976.
- [DH76b] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
- [DLY07] Yevgeniy Dodis, Pil Joong Lee, and Dae Hyun Yum. Optimistic fair exchange in a multi-user setting. In *Public Key Cryptography*, pages 118–133, 2007.
- [DR03] Yevgeniy Dodis and Leonid Reyzin. Breaking and repairing optimistic fair exchange from PODC 2003. In *Digital Rights Management Workshop*, pages 47–54, 2003.
- [EGL85] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.
- [ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
- [FO97] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO*, pages 16–30, 1997.
- [Gen04] Rosario Gennaro. Multi-trapdoor commitments and their applications to proofs of knowledge secure under concurrent man-in-the-middle attacks. In *CRYPTO*, pages 220–236, 2004.

- [GHR99] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In *EUROCRYPT*, pages 123–139, 1999.
- [GJM99] Juan A. Garay, Markus Jakobsson, and Philip D. MacKenzie. Abuse-free optimistic contract signing. In *CRYPTO*, pages 449–466, 1999.
- [GKR97] Rosario Gennaro, Hugo Krawczyk, and Tal Rabin. RSA-based undeniable signatures. In *CRYPTO*, pages 132–149, 1997.
- [GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
- [GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2), 1988.
- [GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.
- [GRK00] Rosario Gennaro, Tal Rabin, and Hugo Krawczyk. RSA-based undeniable signatures. *J. Cryptology*, 13(4):397–416, 2000.
- [GRV03] Sigrid Gürgens, Carsten Rudolph, and Holger Vogt. On the security of fair non-repudiation protocols. In *ISC*, pages 193–207, 2003.
- [GS05] M. Choudary Gorantla and Ashutosh Saxena. Verifiably encrypted signature scheme without random oracles. In *ICDCIT*, pages 357–363, 2005.
- [GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, pages 415–432, 2008.
- [HSMW06] Xinyi Huang, Willy Susilo, Yi Mu, and Wei Wu. Universal designated verifier signature without delegatability. In *ICICS*, pages 479–498, 2006.
- [HYWS08a] Qiong Huang, Guomin Yang, Duncan S. Wong, and Willy Susilo. Ambiguous optimistic fair exchange. In *ASIACRYPT*, pages 74–89, 2008.

- [HYWS08b] Qiong Huang, Guomin Yang, Duncan S. Wong, and Willy Susilo. Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles. In *CT-RSA*, pages 106–120, 2008.
- [IKOS09] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM J. Comput.*, 39(3):1121–1152, 2009.
- [KM01] Steve Kremer and Olivier Markowitch. Selective receipt in certified e-mail. In *INDOCRYPT*, pages 136–148, 2001.
- [KMZ02] Steve Kremer, Olivier Markowitch, and Jianying Zhou. An intensive survey of fair non-repudiation protocols. *Computer Communications*, 25(17):1606–1621, 2002.
- [KR00] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS*. The Internet Society, 2000.
- [LOS⁺06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In *EUROCRYPT*, pages 465–485, 2006.
- [LSW06] Joseph K. Liu, Willy Susilo, and Duncan S. Wong. Ring signature with designated linkability. In *IWSEC*, pages 104–119, 2006.
- [Mao03] Wenbo Mao. *Modern Cryptography: Theory and Practice*. Prentice Hall PTR, 2003.
- [MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures: extended abstract. In *ACM Conference on Computer and Communications Security*, pages 245–254, 2001.
- [MY04] Philip D. MacKenzie and Ke Yang. On simulation-sound trapdoor commitments. In *EUROCRYPT*, pages 382–400, 2004.
- [NS98] David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *ACM Conference on Computer and Communications Security*, pages 59–66, 1998.

- [NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43, 1989.
- [NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*, pages 427–437, 1990.
- [OU98] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In *EUROCRYPT*, pages 308–318, 1998.
- [Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.
- [PCS03] Jung-Min Park, Edwin K. P. Chong, and Howard Jay Siegel. Constructing fair-exchange protocols for e-commerce via distributed computation of RSA signatures. In *PODC*, pages 172–181, 2003.
- [Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, pages 129–140, 1991.
- [RAB⁺08] Ronald L. Rivest, Benjamin Agre, Daniel V. Bailey, Christopher Crutchfield, Yevgeniy Dodis, Kermin Elliott, Fleming Asif Khan, Jayant Krishnamurthy, Yuncheng Lin, Leo Reyzin, Emily Shen, Jim Sukha, Drew Sutherland, Eran Tromer, and Yiqun Lisa Yin. The MD6 hash function a proposal to NIST for SHA-3, 2008.
- [Riv92] R. Rivest. The MD5 message-digest algorithm. RFC 1321, 1992.
- [RS91] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO*, pages 433–444, 1991.
- [RS04] Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In *FSE*, pages 371–388, 2004.
- [RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

- [RSS10] Markus Rückert, Michael Schneider, and Dominique Schröder. Generic constructions for verifiably encrypted signatures without random oracles or nizks. In *ACNS*, pages 69–86, 2010.
- [RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *ASIACRYPT*, pages 552–565, 2001.
- [Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS*, pages 543–553, 1999.
- [Sch89] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO*, pages 239–252, 1989.
- [SKM03] Shahrokh Saeednia, Steve Kremer, and Olivier Markowitch. An efficient strong designated verifier signature scheme. In *ICISC*, pages 40–54, 2003.
- [SST05] Katja Schmidt-Samoa and Tsuyoshi Takagi. Paillier’s cryptosystem modulo p^2q and its applications to trapdoor commitment schemes. In *Mycrypt*, pages 296–313, 2005.
- [Wan10] Guilin Wang. An abuse-free fair contract-signing protocol based on the rsa signature. *IEEE Transactions on Information Forensics and Security*, 5(1):158–168, 2010.
- [ZB06] Huafei Zhu and Feng Bao. Stand-alone and setup-free verifiably committed signatures. In *CT-RSA*, pages 159–173, 2006.
- [ZCDM09] Zongyang Zhang, Zhenfu Cao, Ning Ding, and Rong Ma. Non-malleable statistically hiding commitment from any one-way function. In *ASIACRYPT*, pages 303–318, 2009.
- [ZG96] Jianying Zhou and Dieter Gollmann. A fair non-repudiation protocol. In *IEEE Symposium on Security and Privacy*, pages 55–61, 1996.
- [ZM07] Jianhong Zhang and Jian Mao. A novel verifiably encrypted signature scheme without random oracle. In *ISPEC*, pages 65–78, 2007.
- [ZSNS03] Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. Efficient verifiably encrypted signature and partially blind signature from bilinear pairings. In *INDOCRYPT*, pages 191–204, 2003.

- [ZWQ09] Lei Zhang, Qianhong Wu, and Bo Qin. Identity-based verifiably encrypted signatures without random oracles. In *ProvSec*, pages 76–89, 2009.