

University of Wollongong - Research Online

Thesis Collection

Title: Contributions to credential systems

Author: Lan Zhou

Year: 2007

Repository DOI:

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: This work is copyright. Apart from any use permitted under the Copyright Act 1968, no part of this work may be reproduced by any process, nor may any other exclusive right be exercised, without the permission of the author. 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.

Unless otherwise indicated, the views expressed in this thesis are those of the author and do not necessarily represent the views of the University of Wollongong.

Research Online is the open access repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au

University of Wollongong Thesis Collections

University of Wollongong Thesis Collection

University of Wollongong

Year 2007

Contributions to credential systems

Lan Zhou
University of Wollongong

Zhou, Lan, Contributions to credential systems, M.Comp.Sc.-Res. thesis, School of Information Technology and Computer Science, University of Wollongong, 2007.
<http://ro.uow.edu.au/theses/743>

This paper is posted at Research Online.
<http://ro.uow.edu.au/theses/743>

NOTE

This online version of the thesis may have different page formatting and pagination from the paper copy held in the University of Wollongong Library.

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.



Contributions to Credential Systems

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

Master of Computer Science by Research

from

UNIVERSITY OF WOLLONGONG

by

Lan Zhou

School of Information Technology and Computer Science
May 2007

© Copyright 2007

by

Lan Zhou

All Rights Reserved

Dedicated to
my parents

Declaration

I, Lan Zhou, declare that this thesis, submitted in partial fulfilment of the requirements for the award of Master of Computer Science by research, in the School of Information Technology and Computer Science, University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. The document has not been submitted for qualification at any other academic institution.

Lan Zhou
May 22, 2007

Publication

Conference Papers

L. Zhou, W. Susilo, and Y. Mu. “Three-Round Secret Handshakes Based on El-gamal and DSA.” In *Proceedings of The Second Information Security Practice and Experience Conference (ISPEC 2006)*, volume 3903 of *Lecture Notes In Computer Science*, pages 332–342. Springer-Verlag, 2006.

L. Zhou, W. Susilo, and Y. Mu. “Efficient ID-based Authenticated Group Key Agreement from Bilinear Pairings.” In *Proceedings of The Second International Conference on Mobile Ad Hoc and Sensor Networks (MSN 2006)*, volume 4325 of *Lecture Notes In Computer Science*, pages 288–297. Springer-Verlag, 2006.

Abstract

Three separate credential systems, namely Secret Handshakes (SH), Oblivious Signature-Based Envelopes (OSBE) and Hidden Credentials, have been introduced in recent years. These credential systems are very useful in anonymous communication as they have an interesting common feature which is the ability to combine encryption with access control. This feature allows participants to protect their credentials from being disclosed while running the protocols, which makes these credential systems a natural fit for privacy-preserving and anonymity-oriented applications.

Since these systems have many similarities, interest has arisen in converting them from one to another. Consequently, a series of OSBE schemes based on ElGamal family signatures was proposed, along with a generic construction of SH from OSBE. According to this generic construction, any ElGamal family signature based OSBE scheme can be converted to SH within three communication moves, with the exception of the ElGamal and DSA signatures. To complement the previous result, we propose two three-move SH schemes based on ElGamal and DSA signatures, respectively.

Furthermore, we consider the question of extending the two-party SH to a multi-party setting. We observe that almost all of the SH schemes can be constructed from particular key agreement schemes. Hence we implement an efficient ID-based Authenticated Group Key Agreement (AGKA) scheme, from which we can construct a multi-party SH scheme. Very recently, a new multi-party SH scheme has been proposed based on an unauthenticated group key agreement scheme ahead of our implementation. However, we note that there exists a drawback in this scheme, which may cause the leakage of a valid member's group affiliation in a failed multi-party SH protocol. Therefore, we propose a Group Secret Handshake (GSH) scheme that resists against this attack, and prove that our scheme is secure.

Acknowledgements

I would like to thank my supervisors, Associate Professor Willy Susilo, Associate Professor Yi Mu, for their patient guidance and valuable suggestion during my study. I appreciate them for directing me into the area of cryptography.

I also feel very grateful for all the support I have received from all the staff in the School of Information Technology and Computer Science (SITACS), University of Wollongong.

Finally, I am extremely grateful to my parents for their strong and constantly support. Without their sacrifice, I would never have the opportunity to undertake this research work.

Contents

Publication	v
Abstract	vi
Acknowledgements	vii
1 Introduction	1
1.1 Background	1
1.2 Motivation	3
1.3 Problems and Challenges	5
1.4 Contributions of the Thesis	6
1.5 Structure of the Thesis	7
1.6 Glossary	8
2 Cryptographic Background	10
2.1 Cryptographic Tools	11
2.1.1 Cryptographic Hash Functions	11
2.1.2 Random Oracle Model	13
2.1.3 Elliptic Curves	14
2.1.4 Bilinear Pairings	16
2.2 Complexity Assumptions	17
2.3 Digital Signatures	19
2.3.1 Generic Schemes	19
2.3.2 Attacks to Digital Signatures	21
2.3.3 Example of Digital Signature Schemes	22
2.4 Identity-Based Encryption	24
2.5 Key Agreements	26
2.5.1 Diffie-Hellman Key Exchange	26

2.5.2	Group Key Agreements	27
2.6	Credential Systems	28
2.6.1	Oblivious Signature-Based Envelopes (OSBE)	28
2.6.2	Secret Handshakes (SH)	30
2.7	Reconciling Key Agreements, OSBE and Secret Handshakes	32
2.7.1	Secret Handshakes from Key Agreements	32
2.7.2	Secret Handshakes from OSBE	33
2.8	Summary	33
3	Existing Cryptographic Schemes	34
3.1	Oblivious Signature-Based Envelopes	35
3.1.1	OSBE based on RSA signature	35
3.1.2	OSBE based on Schnorr signature	36
3.1.3	OSBE based on Nyberg/Rueppel signature	36
3.1.4	OSBE based on ElGamal Family Signatures	37
3.1.5	OSBE based on DSA Signature	39
3.2	Two-Party Secret Handshakes	40
3.2.1	SH based on Pairing-Based Key Agreements	40
3.2.2	SH based on CA-Oblivious Encryption	41
3.2.3	SH based on RSA signature	42
3.3	Group Key Agreement Schemes	45
3.4	Group Secret Handshakes	48
3.5	Summary	50
4	Secret Handshake Schemes based on ElGamal and DSA signatures	52
4.1	Security Arguments	53
4.2	Constructions from previously proposed OSBE schemes	54
4.3	Efficient Construction of ElGamal-based SH	55
4.3.1	ElGamal-based Key Agreement Scheme	55
4.3.2	ElGamal based Secret-handshake Scheme	56
4.3.3	Security Proof	58
4.4	Efficient Construction of DSA-based SH	59
4.4.1	Security Proof	61
4.5	Summary	62

5	Group Key Agreement Schemes	63
5.1	Security Arguments	64
5.1.1	Security Model	64
5.1.2	Security Notions	65
5.2	One-Round Group Key Agreement Scheme	66
5.2.1	The Scheme	66
5.2.2	Security Proof	67
5.3	Efficient Group Key Agreement Scheme	70
5.3.1	The Scheme	71
5.3.2	Security Proof	72
5.4	Efficiency Comparison	74
5.5	Summary	75
6	Group Secret Handshakes	76
6.1	Generic Scheme	77
6.2	Security Arguments	78
6.3	Concrete Construction from Bilinear Pairings	80
6.3.1	The Scheme	80
6.3.2	Security Proof	83
6.4	Summary	89
7	Conclusions and Future Work	90
	Bibliography	93

List of Tables

1.1	Glossary	9
5.1	AGKA Scheme Efficiency Comparison	75

List of Figures

2.1	A Typical Hash Function	12
2.2	Elliptic Curve Addition	15
2.3	Creating a Digital Signature	20
2.4	Verifying a Digital Signature	20
2.5	Identity-Based Encryption	25
2.6	Diffie-Hellman Key Agreement	27
2.7	Oblivious Signature-Based Envelopes	29
2.8	Secret Handshake	31
5.1	One-Round ID-based Group Key Agreement	68
5.2	Two-Round ID-based Group Key Agreement	72

Chapter 1

Introduction

1.1 Background

A credential system is a set of authentication protocols assisted by a proof (credential) of qualification, competence, or clearance that is attached to a user. Using these protocols, users can prove their possession of the valid credentials but never required to show their credentials to others. A credential system consists of several proposed protocols, including Secret Handshakes (SH), Oblivious Signature-Based Envelopes (OSBE) and Hidden Credentials.

Secret Handshakes

The concept of Secret Handshakes was introduced by Balfanz *et al.* [5] in 2003. This system allows two group members to authenticate each other secretly. Assume that party A is a member of group G_1 with role r_A , and B is a member of group G_2 with role r_B . After they finish running an SH protocol,

- If G_1 does not equal G_2 , neither A nor B learns anything about the other party.
- Both A and B learn the group affiliations of the other party only if G_1 equals G_2 .
- A can choose to only authenticate the group member with particular roles (e.g., B with role r'_B). Neither A nor B learns anything about the other party if G_1 does not equal G_2 or r_B does not equal r'_B . The same is true for B .
- A third party observing the exchange between A and B does not learn anything about these two parties, including whether they belong to the same group or not.

In addition to introducing the SH concept, Balfanz *et al.* [5] provided the first solution of SH by using the bilinear pairing. Castelluccia *et al.* [29] introduced a new

construction of SH scheme through the use of a CA-Oblivious encryption protocol, in which they use a Schnorr signature [86] as the credential. The term “CA-Oblivious” implies that a credential which is not issued by a Certification Authority (CA) will not enable a user to guess whether another user (s)he interacts with has the credential issued by the CA or not. Xu and Yung [98] also presented a SH scheme with reusable credentials. Their scheme achieves unlinkability, and an invalid user can only infer that a participant is one out of a certain number k users in the worst case. So it is called k -anonymity. Finally, Vergnaud [93] constructed two SH schemes by using RSA signature [84] as credentials.

Tsudik and Xu [91] extended the notion of SH to a multi-party setting by combining three main ingredients: a group signature scheme [8, 13, 62], a centralised group key distribution scheme [97], and a distributed group key agreement scheme [9, 25] to create a framework for multi-party SH. Jarecki, Kim and Tsudik [56] also provided a solution to multi-party SH, which is constructed based on an un-authenticated group key agreement protocol from Burmester and Desmedt [24].

Oblivious Signature-Based Envelopes

Oblivious Signature-Based Envelopes (OSBE) is a system introduced by Li, Du and Boneh [66] in 2003. In this system, a sender S can encrypt a message and send it to a receiver R . They both agree on a common message. When R receives the message from S ,

- R can decrypt the encrypted message from S only if R has a signature on the agreed-upon message issued by a third party.
- S does not learn anything about R , including whether R has the valid signature or not.
- A third party observing the exchange between S and R does not learn anything about them, including what is the content of the message from S , and whether R has the valid signature or not.

Li *et al.* also presented three concrete OSBE schemes. One is based on RSA signature [84], and other two are based on Boneh-Franklin [19] and Cocks [33] identity-based encryption systems respectively. Nasserian and Tsudik [72] proposed a series of OSBE schemes based on the ElGamal family signatures schemes, including Schnorr

signature [86], Nyberg/Rueppel signature [74], ElGamal signature [40] and DSA signature [75]. Nasserian and Tsudik also presented a generic construction of secret handshake from their OSBE schemes.

Hidden Credentials

Holt *et al.* [54] introduced the concept of Hidden Credentials in 2003. Hidden Credentials use ideas from identity-based cryptosystems [88], but this system encrypts messages against policies instead of identities. Suppose that party B encrypts a message against a policy P , and sends it to party A . When A receives the message,

- A can only decrypt the message from B if A fulfills the policy P .
- B does not learn anything about the credential of A . Consequently, B will never know if A fulfills the policy P .
- A does not know the public keys which B used to encrypt the message. A must attempt decryption using each of her credentials. So A learns nothing about the policy controlling access to the message, if (s)he does not possess the right credentials.

Holt *et al.* [54] also constructed the Hidden Credential scheme by making use of identity-based encryption [19]. Some other solutions [21, 44] of Hidden Credential were provided after then. Moreover, Bradshaw *et al.* [21] presented a secret splitting scheme, which can extend indistinguishability of the system to complex policies.

1.2 Motivation

Authentication, a process of verifying the identity of a user, always plays an important role in communications. For example, a privacy system may require a user to authenticate himself first to a server before (s)he can access the system. To authenticate, this user should be equipped with a certificate issued by a trusted authority. However, several threats exist in authentication process. Eavesdropping is a serious problem in authentication. Normally, people communicate over an insecure and open network. Thus an invalid user can easily intercept the certificate when it is sent out for authentication. Even if the certificate is securely delivered to the server of the system, a corrupt server may also leak the certificate to an unqualified user. Once the invalid

user obtains the certificate, (s)he can easily invade the systems which (s)he has no permissions to enter.

The credential system mentioned in Section 1.1 was introduced to overcome these threats. The first proposed system, called Secret Handshakes [5], allows two members of the same group to authenticate each other secretly, which is very useful for maintaining privacy against anonymous peers on a network. In an SH protocol, the group affiliation of an valid member will not be disclosed unless the other party is also a valid member of the same group. Subsequently, a system called Oblivious Signature-Based Envelopes [66] was introduced, in which encrypted messages can only be recovered by a third party's signature on an agreed-upon message, and the valid receiver will never reveal his possession of the signature during the interacting process. Finally, Hidden Credentials [54] was proposed. The system allows messages to be encrypted against complex policies, and protects policies from leaking to unqualified recipients.

The above-mentioned credential systems share many common features. These credential systems do not allow users to generate their own certificates. In other words, credentials need to be issued and potentially logged by a trusted third party, called Certificate Authority. Moreover, credentials used in these systems are all derived from some digital signature schemes, to ensure that these credentials cannot be forged. In addition, all of these credential systems have the ability to integrate encryption with access control, which can protect the credentials of the user from being revealed to adversaries.

With the increasing popularity of networks, two-party protocols become less sufficient in being used in authentication. When a group of users wish to authenticate each other over an open network, they still face threats from malicious users. Therefore, the extension of the credential systems to a multi-party scenario is desired, which is not straightforward. In the two-party scenario of SH, if each party does not know the secret calculated by the other party in a failed protocol, we can assume neither of them will learn the identity of opposite party. However, in a failed multi-party SH protocol, we should not only ensure that the unqualified user has no knowledge about the secret computed by other parties, but also prevent valid parties from generating the same secret. Because in that case, the same secret from them will disclose their group affiliations. Consequently, much work with respect to credential systems is required in the future.

1.3 Problems and Challenges

Since the concept of SH was introduced, many SH schemes have been proposed based on various signatures, and all of them can be completed in three moves. Recently, a series of OSBE schemes was proposed by Nasserian and Tsudik [72] based on ElGamal family signature schemes, where the authors also discussed the generic conversion from OSBE to SH schemes. According to this generic construction, most OSBE schemes based on ElGamal family signatures can be easily converted to SH schemes which require only three moves. However, Nasserian and Tsudik's construction of three-move SH protocol does not work for ElGamal- and DSA-OSBE schemes, and an additional communication move is required. This makes ElGamal-based and DSA-based SH schemes from Nasserian and Tsudik's generic construction [72] less efficient, compared to other previously proposed SH schemes.

Most prior SH schemes in the literature focused on two-party scenarios. Hence a consideration has emerged on extension from two-party SH to a multi-party setting. We observed that SH schemes can be constructed from particular key agreement schemes. Before seeking a multi-party SH scheme, we firstly move to the area of group key agreement protocols, where we note that the ID-based Authenticated Group Key Agreement (AGKA) is suitable for implementing the Group Secret Handshakes (GSH). Group signatures [2, 30] might appear to be an attractive method for implementing SH schemes. However, they only offer anonymity and unlinkability for credentials of group members, not secrecy of membership itself.

AGKA is a group key agreement protocol ensured with an authentication mechanism, which can be classified into two categories: Certificate-based and ID-based. Here we focus on the ID-based AGKA schemes. A number of ID-based AGKA protocols have been proposed in recent years. Nevertheless, most efficient constructions require two rounds to create a group session key. Some single round tripartite AGKA protocols have been proposed but these methods cannot be extended to large groups consisting of more than three parties. Recently, a single round ID-based AGKA protocol was proposed by Shi *et al.* [89]. Although their construction is very efficient and takes only one round, we note that this scheme is not an ID-based AGKA, rather than a public key based scheme. Moreover, in their scheme, a group member can learn the group session key even if (s)he is not involved in the protocol conducted by other members.

A GSH scheme was proposed by Jarecki *et al.* [56] recently. Their construction is based on a previously proposed group key agreement protocol. However, we found a disadvantage in their scheme. Following the definition of SH from Balfanz *et al.* [5], an

honest party will never leak his group affiliation to other parties, if the group handshake protocol fails. In contrast to this requirement, in Jarecki *et al.*'s scheme, an invalid member has the ability to make other honest parties share a common group session key in a failed protocol. Hence (s)he can learn that these parties belong to the same group, which violates the security requirements of SH defined by Balfanz *et al.* [5].

1.4 Contributions of the Thesis

To cope with the challenges we are facing in implementing credential systems, we provide several solutions, which will also be discussed further in the following chapters of this thesis.

We first review Nasserian and Tsudik's [72] OSBE schemes based on ElGamal family signatures, and point out their DSA-based OSBE has not constructed correctly. Thus it cannot be used to implement the DSA-based SH scheme. Subsequently, we construct an ElGamal-based SH scheme following the idea of Nasserian and Tsudik [72], and show that this scheme requires four moves to be completed. Since no three-move SH scheme based on ElGamal or DSA signature exists in the literature, we propose two novel SH schemes based on ElGamal and DSA signatures, respectively, to complement the previous result. More precisely, we first construct an ElGamal signature based key agreement scheme, and then introduce our SH scheme based on this primitive. For the first time, we achieve three-move SH schemes based on these two signatures. We also prove that our proposed schemes are secure.

In order to extend the SH scheme to a multi-party setting, we move to the area of group key agreement, and search for a suitable ID-based AGKA scheme, from which we can construct a multi-party SH scheme. Then we review two efficient ID-based AGKA schemes in prior work, and show an attack to the single round ID-based AGKA scheme introduced by Shi *et al.* [89]. This attack can make a malicious member in the group learn the session key constructed by other group members in an AGKA protocol. Therefore, a secure one-round ID-based AGKA protocol is desired to construct an efficient multi-party SH scheme, just as we propose the three-move ElGamal-based SH scheme from an key agreement protocol based on ElGamal signature.

Afterwards, for the first time in the literature, we present a provably secure one-round ID-based AGKA scheme. The scheme is a contributory key agreement in which each group member takes responsibility for contributing to the generation of group session key. We also present an efficient two-round ID-based AGKA protocol, which is

a variant of our one-round ID-based AGKA. Moreover, we show that the efficiency of this scheme outperforms any other existing schemes in the literature. Then we provide the security proof of these two schemes.

Subsequently, we present a defect in the GSH scheme proposed by Jarecki *et al.* [56], which may cause a leakage of valid members' group affiliation when an adversary is involved in the protocol. To construct a GSH scheme, which is able to fulfill the security requirements of SH system, we first define a security model. Then we propose our GSH scheme, which overcomes this disadvantage, by using the pairing cryptography. Finally, we provide a comprehensive security proof of our GSH scheme in the model we defined.

1.5 Structure of the Thesis

The rest of the thesis is organised as follows:

- In Chapter 2, we introduce the cryptographic primitives related to this thesis, including the definitions and properties of the cryptographic techniques used throughout. Then we review basic security requirements and categories of the attacks to the digital signature schemes. We also provide the example of some well-known digital signatures, which will be used in our proposed schemes. Then we describe the identity-based encryption (IBE) and group key agreement, from which credential schemes can be constructed. After that, we introduce the definitions of two credential systems: OSBE and SH, and present the generic schemes respectively. Finally, we discuss the conversion among these different systems.
- In Chapter 3, we focus on the existing schemes in cryptographic systems described above. Firstly, we review several OSBE schemes based on a series of signature schemes in discrete logarithm cryptosystem. Then we describe some efficient constructions of SH existed in the literature. Subsequently, we discuss three group key agreement schemes. Two of them are ID-based AGKA schemes, which are all constructed by using pairing cryptography. We also point out these two ID-based AGKA schemes are flawed. Finally, we describe a GSH scheme proposed very recently, and show that this scheme has a problem which violates the security requirements of SH.
- We present our first cryptographic construction in Chapter 4. We first review the security requirements of SH scheme. Following the previously proposed generic construction from OSBE to SH, we implement an ElGamal-based SH scheme

from the OSBE based on ElGamal signature, which requires four moves. To bring up our SH scheme based on ElGamal signature, we introduce a concrete ElGamal-based key agreement protocol. Then we present our efficient construction of ElGamal-based SH scheme that can be completed in only three moves, and provide the security proof. We also present our three-move SH scheme based on DSA signature and prove that the scheme is secure.

- In Chapter 5, we describe our group key agreement schemes. We begin with setting up the security model and introducing the security notions, which will be used in proving the security of group key agreement scheme. Subsequently, we present our two ID-based AGKA schemes. One is a single round group key agreement protocol, which is considered as the first secure one-round ID-based AGKA in the literature. Another is our two-round group key agreement scheme, which is more efficient in communication costs than other previously known ID-based AGKA protocols. We provide the security proofs of both two schemes. Finally, we conduct a comparison over previously described AGKA schemes in Section 3.3 and our proposed schemes. We show that our schemes are more efficient than other schemes.
- The description of our GSH scheme is given in Chapter 6. We firstly formalise the generic scheme of GSH. Since the disadvantage of previously described GSH scheme makes it not satisfy the security properties of SH, here we extend the security requirements of SH described in Chapter 4 to a multi-party version, and redefine the model for security proof. Then we present our group secret handshake scheme, along with a comprehensive security proof.
- In Chapter 7 we summarise the contributions of this thesis and make the conclusion. In addition, we state some open problems in future work of this research area.

1.6 Glossary

ACK	Acknowledgement
AGKA	Authenticated Group Key Agreement
BD-GKA	Burmester <i>et al.</i> 's Group Key Agreement Scheme
BDH	Bilinear Diffie-Hellman
CA	Certificate Authority
CDH	Computational Diffie-Hellman
CHL-AGKA	Choi <i>et al.</i> 's Group Key Agreement Scheme
DBDH	Decisional Bilinear Diffie-Hellman
DBH	DSA Based Secret Handshake Scheme
DDH	Decisional Diffie-Hellman
DH	Diffie-Hellman
DSA	Digital Signature Algorithm
EBH	ElGamal Based Secret Handshake Scheme
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
GA	Group Administrator
Gi-SH	Girault's Key Agreement Based Secret Handshkae
GSH	Group Secret Handshake
IBE	Identity-Based Encryption
ID	Identity
MAC	Message Authentication Code
MD	Message Digest Algorithm
NIST	National Institute of Standards and Technology
O-AGKA	One-round Authenticated Group Key Agreement Scheme
OSBE	Oblivious Signature-based Envelope
OT-SH	Okamoto <i>et al.</i> 's Key Agreement Based Secret Handshake
PK	Public Key
PKG	Private Key Generator
PKI	Public Key Infrastructure
PPT	Probabilistic Polynomial Time
ROM	Random Oracle Model
SCL-AGKA	Shi <i>et al.</i> 's Group Key Agreement Scheme
SH	Secret Handshake
SHA	Secure Hash Algorithm
SK	Secret Key
T-AGKA	Two-round Authenticated Group Key Agreement Scheme
PBH	Pairing Based Secret Handshake Scheme

Table 1.1: Glossary

Chapter 2

Cryptographic Background

Recently, several cryptographic systems have been proposed, which allow credential contents to be used directly in access control processes. These new constructions, namely credential systems, include OSBE, SH, and Hidden Credential. In this chapter, we introduce a set of cryptographic backgrounds, which plays a fundamental role in construction and security proof of credential systems. Since the credentials used in these systems are typically built from some digital signature schemes, we then briefly describe the security requirements of digital signatures. Subsequently, we discuss the definitions of OSBE and SH systems, and then the reconciliation of these systems. Firstly, we start by introducing related cryptographic primitives including one-way hash functions, the random oracle model, elliptic curve cryptography, and bilinear pairing.

2.1 Cryptographic Tools

2.1.1 Cryptographic Hash Functions

Cryptographic hash functions play a fundamental role in cryptography, data integrity and digital signatures. The original purpose of a hash function is to generate a “fingerprint” of a message or block of data. There are two main types of hash functions, keyed hash functions and unkeyed hash functions. The keyed hash functions can be used to generate Message Authentication Codes (MAC), which is a symmetric-key method used to protect a message from unauthorized alteration.

It is usually assumed that the hash functions are public and not keyed. Therefore, we mainly focus on unkeyed hash functions in this thesis. An unkeyed hash function is a **one-way function**, which takes a string of bits with arbitrary length as input, and generates a short digest of fixed length. A hash value h is produced by a hash function H of the form

$$h = H(M)$$

where M is a variable-length message and h is the fixed-length hash value. Normally, the hash functions should satisfy the following properties (adapted from a list in [73, 90]):

- (1) **Compression** : H can be applied to a block of data of arbitrary size, and the size of output should be fixed.
- (2) **Computable** : For any given input x , $H(x)$ must be computable in polynomial time, making implementations practical in both hardware and software.
- (3) **Pre-image resistance** : For any given hash value h , it is computationally infeasible to find x such that $H(x) = h$.
- (4) **Weak collision resistance** : For any given message x , it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$.
- (5) **Strong collision resistance** : It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$.

An example of typical hash function is shown in Fig. 2.1.

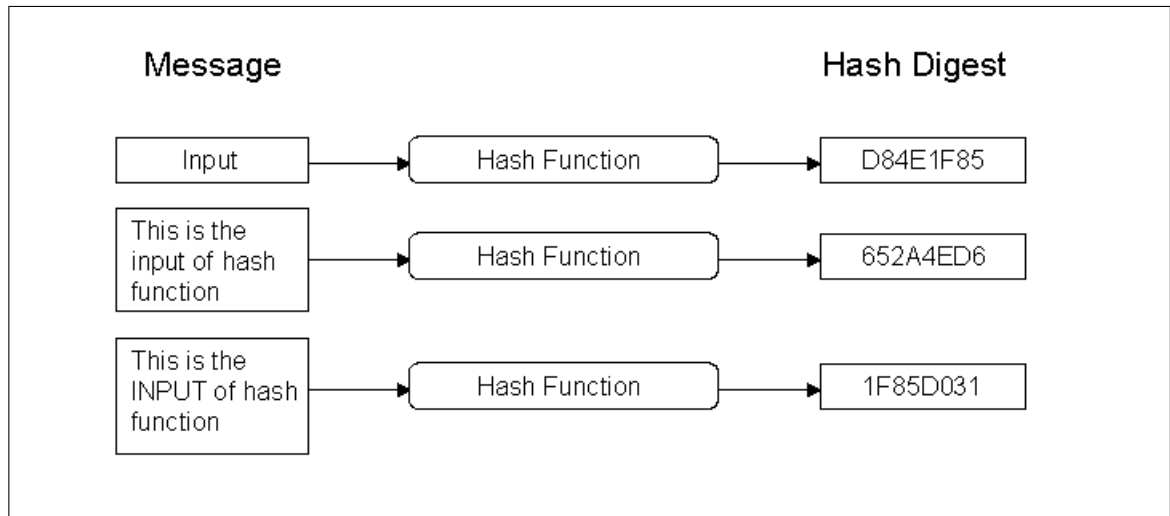


Figure 2.1: A Typical Hash Function

Birthday Attack

Firstly, let us review the birthday paradox: If there are 23 people in a room, the chance that two of them have the same birthday is more than 50%. If we assume that a birthday is a hash function, which takes a human as input and outputs a day in one year, we can have a probability of more than 50% to find a collision of this hash function only after 23 attempts. Therefore, many strategies were proposed to attack the hash functions based on the birthday paradox [78, 99].

The analysis on this paradox shows: Suppose there are m possible hash values. If we evaluate the hash value of k randomly selected messages, the probability of at least one collision is

$$[Pr(m, k)] > 1 - e^{-k(k-1)/2m}, \text{ where } e \approx 2.7$$

The above equation shows that the chance of success in this attack depends only on:

- the size of the message
- the size of the hash digests

So the size of the message and hash space should be large enough to resist the attacks based on birthday paradox.

The drive of hash algorithm started with public key cryptography. MD5 [83] and SHA-1 [3] are two most well-known hash algorithms. MD5 hash algorithm was proposed by Ron Rivest at MIT, and soon became the most widely used secure hash algorithm

at that time. SHA-1 hash algorithm was developed by National Institute of Standards and Technology (NIST). Compared with MD5, SHA-1 is a bit slower to perform and presumably more secure. However, MD5 and SHA-1 were all broken recently [94–96]. A collision in MD5 can be found in several minutes and the time used to produce a collision in SHA-1 has been reduced sharply. Consequently, migration to SHA-2 is recommended by NIST.

2.1.2 Random Oracle Model

Many different models have been proposed to assist with security proofs of cryptographic schemes, for example: random oracle model (ROM) [11], black box model [87], etc. We only focus on random oracle mode in this thesis.

Random oracle model [11] is a mathematical technique used in cryptography to help with simplifying proofs of security. It provides a bridge between cryptographic theory and cryptographic practice. Generally speaking, a random oracle is a theoretical black box that responds to every query with a random response chosen uniformly from its output domain, except that for any specific query, it responds the same way every time it receives that query. Proofs which make use of random oracles are referred to as secure in the “random oracle model” which was formalised by Bellare and Rogaway [11], as opposed to the “standard model”.

The idea of random oracle builds on the work of Goldreich, Goldwasser and Micali [49, 50] and Fiat-Shamir [42]. Suppose that all parties participating in a security protocol have access to a public random function, which maps from $\{0, 1\}^*$ to $\{0, 1\}^\infty$, and all oracle queries in the protocol are answered by this single function. Security proofs of the cryptographic protocols are made assuming this function is chosen randomly. Then this random function will be replaced by an object like a hash function.

In practice, random oracles are typically used to model cryptographic hash functions in schemes where strong random assumptions are required for the hash function’s output. However, not all usage of cryptographic hash functions require random oracles: schemes which require only the property of collision resistance can be proven secure in the standard model. Moreover, quite a few people dislike the ROM as a tool for proving security, and the schemes without using random oracle are preferred.

The problem with the ROM is that it does not really model real life. In real life, there are no random functions. Canetti, Goldreich and Halevi [27] constructed an artificial counterexample: one that is provably secure in the random oracle model but insecure when the random oracle is instantiated in real-life with any polynomial-time

computable function. Nevertheless, the proof in the random oracle model can still be taken as evidence of security when the random oracle is replaced by a particular hash function, and no practical protocol proven secure in the random oracle model has been broken when used with a “good” hash function.

2.1.3 Elliptic Curves

Elliptic curves system have been studied in mathematics for a long time, but their use in cryptographic applications as an alternative mechanism for implementing public-key cryptography was first suggested by Miller [70] and Koblitz [65] in the mid 1980s. Comparing with the discrete logarithm cryptosystems, the main benefit of using elliptic curve is that they offer a significantly greater level of security within the same computational complexity.

An elliptic curve is an algebraic curve defined by an non-singular equation of the form

$$y^2 = x^3 + ax + b \quad (2.1)$$

A set of points (x, y) which satisfy the above equation, together with a special point \mathcal{O} called the *point at infinity*, make up of the elliptic curve group, denoted by E . For applications to cryptography, we consider finite fields of q elements. Two families of elliptic curves are used in cryptographic applications: *prime curves* defined over \mathbb{Z}_p and *binary curves* constructed over $GF(2^n)$. Prime curves are pointed out to be the best for software applications [41].

The elliptic curve group is a commutative group, we describe the group by using additive notation in this thesis. Assume $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ are the two points on the curve, and they are not negative of each other. Then we can define a new point $R = P + Q = (x_R, y_R)$ as follows:

$$\begin{aligned} x_R &= s^2 - x_P - x_Q \\ y_R &= -y_P + s(x_P - x_R) \\ \text{where } s &= (y_P - y_Q)/(x_P - x_Q) \end{aligned}$$

When we double the point P , we compute a new point $R = 2P = (x_R, y_R)$ as follows:

$$\begin{aligned} x_R &= s^2 - 2x_P \\ y_R &= -y_P + s(x_P - x_R) \\ \text{where } s &= (3x_P^2 + a)/(2y_P) \end{aligned}$$

Recall that a is one of the coefficients in equation (2.1). A geometric description of the addition over elliptic curves is shown in Fig. 2.2.

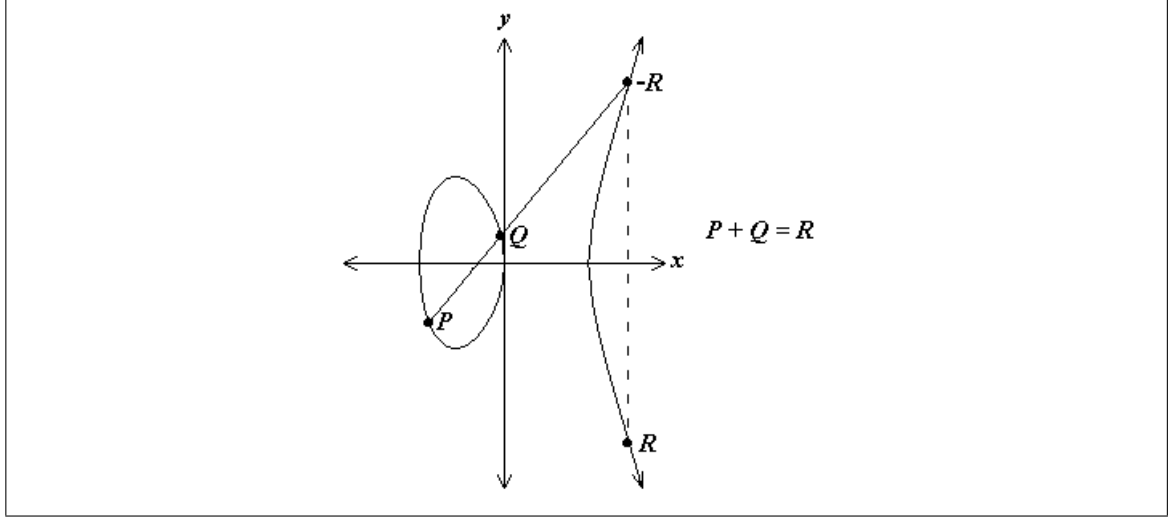


Figure 2.2: Elliptic Curve Addition

From the above definitions and the accepted conventions as to point at infinity \mathcal{O} , we can have the following properties of points on elliptic curve:

- (1) $P + Q = Q + P$ for all points $P, Q \in E$;
- (2) $P + (Q + S) = (P + Q) + S$ for all points $P, Q, S \in E$;
- (3) there exists a point at infinity \mathcal{O} such that $P + \mathcal{O} = \mathcal{O} + P = P$ for all $P \in E$;
- (4) for every point $P \in E$ there exists an opposite point $-P \in E$, such that $P + (-P) = \mathcal{O}$.

The discrete logarithm problem is the basis for the security of many cryptosystems including the Elliptic Curve Cryptosystem (ECC). More specifically, the ECC relies upon the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP).

To utilize ECDLP in constructions of ECC, the cyclic subgroup of E is generated by firstly choosing a point $P \in E$ and an integer k where

$$kP = \underbrace{P + P + \dots + P}_k$$

We say n is the order of P if $nP = \mathcal{O}$. Then the ECDLP is defined as follows:

Definition 1 Given points P and $Q \in E$, of which the order are n , find an integer k , $0 \leq k \leq n - 1$, such that $Q = kP$.

The basic operation of ECDLP in ECC is elliptic curve scalar multiplication, which is the main cryptographic operation on elliptic curve. Any discrete logarithms based cryptosystem can be extended to elliptic curve easily. The basic principle is using $P = kQ$ to replace the operation $y = g^k \pmod{p}$. Therefore, the scalar multiplication can be performed by using the same methods as for exponentiation in discrete logarithm cryptosystem. Recently, some efficient methods for scalar multiplication have been proposed [14,68], but the performance improvement is not very great and these methods go beyond the scope of this thesis.

2.1.4 Bilinear Pairings

Bilinear pairing is a map which maps a point on an elliptic curve to a finite field, which is defined as follows:

Let \mathbb{G}_1 be a cyclic additive group of a prime order q , and \mathbb{G}_2 be a cyclic multiplicative group with a prime order p .

Definition 2 (*Bilinear map*). We say that a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is bilinear if it satisfies all the following properties:

- (1) **Bilinear:** for all $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$, we have $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$;
- (2) **Non-degenerate:** there exists $P, Q \in \mathbb{G}_1$ such that $\hat{e}(P, Q) \neq 1$;
- (3) **Computable:** for all $P, Q \in \mathbb{G}_1$, the pairing $\hat{e}(P, Q)$ is computable in polynomial time.

The Non-degeneracy property list above implies that when P is the generator of \mathbb{G}_1 , $\hat{e}(P, P)$ is also the generator of \mathbb{G}_2 . We call such bilinear map as an admissible bilinear pairing.

Although earlier work suggesting the use of pairings in cryptography was done by [71] and [85], the pairing did not impress on most people until the publication of “Identity-based encryption scheme” by Boneh and Franklin [19] in 2001, which solves the long-standing open problem. A new implementation of pairing called Weil pairing was also introduced in this paper. In fact, another implementation called Tate pairing [43] is superior to the Weil pairing.

Now matter how these pairings are implemented, they all have following additional properties:

- $\hat{e}(P_1 + P_2, Q) = \hat{e}(P_1, Q) \cdot \hat{e}(P_2, Q)$, for all $P_1, P_2, Q \in \mathbb{G}_1$;

- $\hat{e}(P, Q) = \hat{e}(Q, P)$, for all $P, Q \in \mathbb{G}_1$;
- if for $P \in \mathbb{G}_1$ we have $\hat{e}(P, Q) = 1$ for all $Q \in \mathbb{G}_1$, then $Q = \mathcal{O}$.

Because of the excellent properties, pairing is widely used in identity-based cryptography, which makes it currently the most active research area in ECC [46, 58].

2.2 Complexity Assumptions

In this section, we review some cryptographic assumptions that will be used throughout the thesis. Firstly, we describe two well-known assumptions in discrete logarithm cryptosystem.

Definition 3 (*Computational Diffie-Hellman (CDH) Problem* [34]). *Given a cyclic group \mathbb{G} of the order q , g is a generator of group \mathbb{G} , the Computational Diffie-Hellman problem is (t, ε) -hard if for all t -time adversaries \mathcal{A} we have*

$$\text{Adv}_{\mathcal{A}}^{\text{CDH}} = \Pr[\mathcal{A}(g, g^a, g^b) = g^{ab} \pmod{p} | a, b \in \mathbb{Z}_p^*] < \varepsilon$$

CDH Assumption : Solving the Computational Diffie-Hellman problem in polynomial time is infeasible. In other words, for $a, b \in \mathbb{Z}_p^*$, given g^a and g^b , compute g^{ab} is infeasible.

Definition 4 (*Decisional Diffie-Hellman (DDH) Problem* [40]). *Given a cyclic group \mathbb{G} of the order q , g is a generator of group \mathbb{G} , for $a, b, c \in \mathbb{Z}_p^*$, the Decisional Diffie-Hellman problem is (t, ε) -hard if for all t -time adversaries \mathcal{A} we have*

$$\text{Adv}_{\mathcal{A}}^{\text{DDH}} = |\Pr[\mathcal{A}(g, g^a, g^b, g^{ab}) = 1] - \Pr[\mathcal{A}(g, g^a, g^b, g^c) = 1]| < \varepsilon$$

DDH Assumption : No efficient algorithm can solve DDH problem with non-negligible advantage. In other words, for $a, b, c \in \mathbb{Z}_p^*$, given g^a , g^b , and g^c , decide whether $c \equiv ab \pmod{q}$ is infeasible.

As we mentioned above, assumptions in discrete logarithms based cryptosystem can be easily extended to elliptic curve. In bilinear pairings, Computational Diffie-Hellman problem is still hard. However, Joux and Nguyen [59] point out that Decision Diffie-Hellman problem [16] is easy. To see this, observe that given $P, aP, bP, cP \in \mathbb{G}_1$, we have

$$c = ab \bmod q \iff \hat{e}(P, cP) = \hat{e}(aP, bP)$$

Then the assumptions in bilinear pairings are modified to preserve the security which makes them a bit different from previously mentioned assumptions. The assumptions are described as follows:

Definition 5 (*Bilinear Diffie-Hellman (BDH) Problem* [19,57]). Given a cyclic group $\mathbb{G}_1, \mathbb{G}_2$ of the order q together with a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, and P is a generator of group \mathbb{G}_1 . The Bilinear Diffie-Hellman problem is (t, ε) -hard if for all t -time adversaries \mathcal{A} we have

$$\text{Adv}_{\mathcal{A}}^{\text{BDH}} = |\Pr[\mathcal{A}(P, aP, bP, cP) = \hat{e}(P, P)^{abc}]| < \varepsilon$$

BDH Assumption : We say that if there exists a polynomial time algorithm which can solve BDH problem, the probability is negligible. In other words, given (P, aP, bP, cP) for random $P \in \mathbb{G}_1$, and $a, b, c \in \mathbb{Z}_q^*$, compute $\hat{e}(P, P)^{abc}$ is infeasible.

Definition 6 (*Decisional Bilinear Diffie-Hellman (DBDH) Problem* [17,18]). Given a cyclic group $\mathbb{G}_1, \mathbb{G}_2$ of the order q together with a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, and P is a generator of group \mathbb{G}_1 . The Decisional Bilinear Diffie-Hellman problem is (t, ε) -hard if for all t -time adversaries \mathcal{A} we have

$$\text{Adv}_{\mathcal{A}}^{\text{DBDH}} = |\Pr[\mathcal{A}(P, aP, bP, cP, \hat{e}(P, P)^{abc}) = 1] - \Pr[\mathcal{A}(P, aP, bP, cP, \hat{e}(P, P)^d) = 1]| < \varepsilon$$

DBDH Assumption : We assume that the probability of a polynomial time algorithm to solve DBDH problem is negligible. In other words, for random $P \in \mathbb{G}_1$, and $a, b, c, d \in \mathbb{Z}_q^*$, distinguish between tuples of the form $(P, aP, bP, cP, \hat{e}(P, P)^{abc})$ and $(P, aP, bP, cP, \hat{e}(P, P)^d)$ is infeasible.

Above system parameters can be obtained through running a probabilistic polynomial time (PPT) algorithm, called the **BDH Parameter Generator** \mathcal{IG}_{BDH} , which takes a security parameter k as input, runs in polynomial time in k , and outputs a prime number q , the description of two groups $\mathbb{G}_1, \mathbb{G}_2$ of the same order q , and the description of an admissible bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.

2.3 Digital Signatures

Digital signature, as the term suggests, is a sort of signature in electronic form. Like traditional signatures, digital signatures can protect information from undesirable modification and prove the authorship of the information. Different from the traditional handwritten signatures, digital signatures' verification requires the assistance from some authorised public information, which we will discuss in the following. Generally speaking, digital signatures can provide following security services :

- **Authentication** : authorisation of the identity. When a user in a group has been issued a digital signature on his identity, (s)he can use this signature as a proof of his identity's validity, so that other parties can use the group public key to verify. If the verification succeeds, one can assume that the signature was generated by using the group private key, and the user is proved to be authorised.
- **Data Integrity** : assure that the data has not been modified since the digital signature was generated. When a sender sends a message together with the digital signature on it, receiver can use the public key of the sender to verify the signature with the received message. Even a slight modification of message will result in signature verification failure. Adversaries can alter the message in communication easily, but they cannot forge a signature on it without knowing the private key of the sender.
- **Non-repudiation** : providing evidence to a third-party that a party participated in a transaction. When a sender sends out a message, (s)he can never deny that the message was written by him once (s)he produces a digital signature on the message, because nobody else can generate the signature for they have no knowledge of the send's private key.

2.3.1 Generic Schemes

Digital signature scheme is a cryptographic algorithm based on asymmetric key cryptosystem. The main difference between digital signature and public key encryption is that digital signature encrypts messages with private key, and decrypts with public key. A digital signature scheme consists of three algorithms:

- **Key generation algorithm** : takes as input a security parameter and outputs a pair (PK, SK) of matching public and private key.

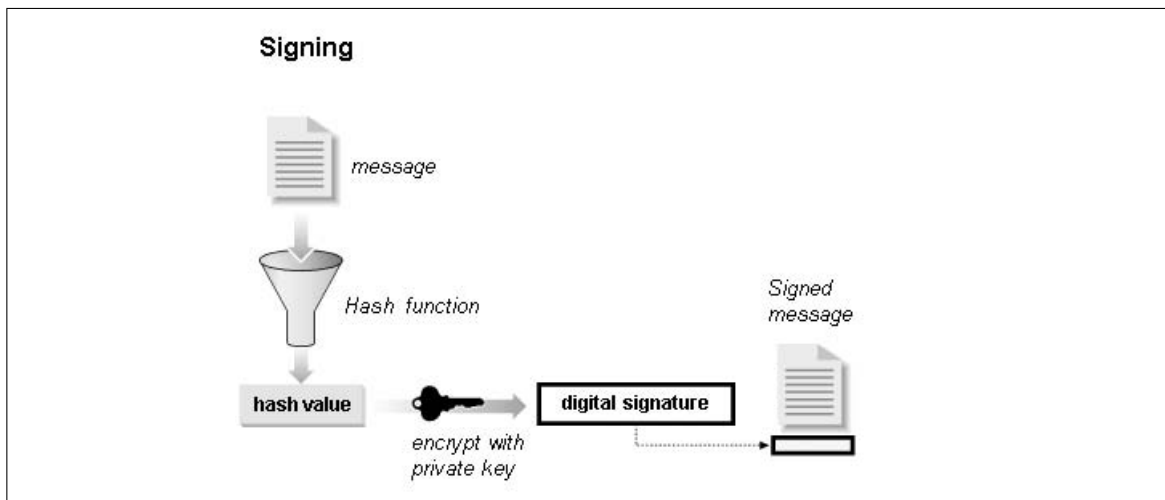


Figure 2.3: Creating a Digital Signature

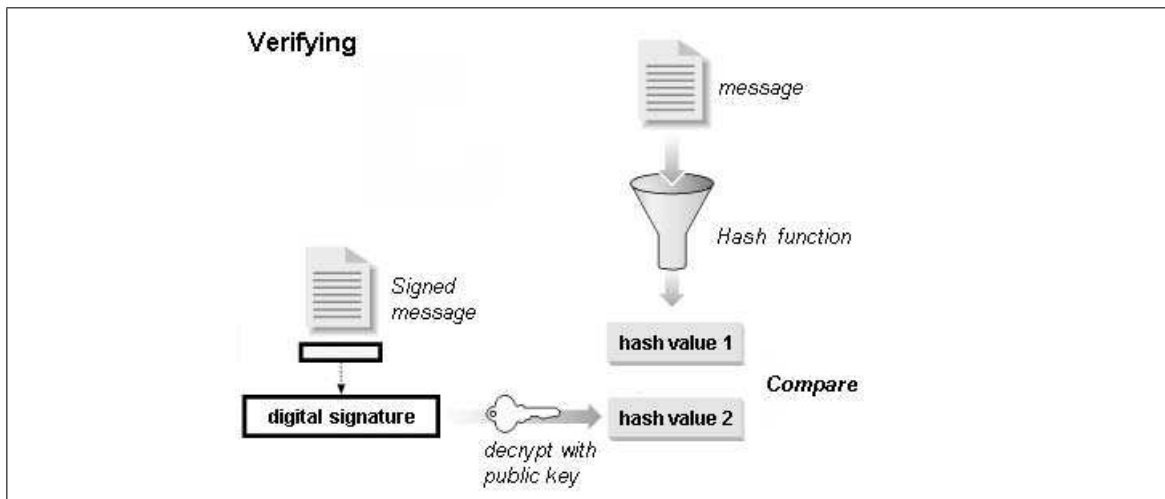


Figure 2.4: Verifying a Digital Signature

- **Signature creation algorithm** : takes as input a message M , and produces a signature $Sig(M)$. Usually we don't sign the whole message, but just a hash value of the message. The detail steps of the algorithm are as shown in Fig. 2.3. Firstly, a sender calculates the hash value of the message, and generates the signature by encrypting the hash value with his private key. The signature is attached to the message to create a signed message.
- **Signature verification algorithm** : takes as input a message M and the corresponding signature $Sig(M)$, outputs true or false. We show the detail steps of the algorithm in Fig. 2.4. When a signed message is received, the receiver calculates

the hash value of the message, and decrypts the signature with the public key of the sender. The receiver then compares these two value, and outputs **true** only if they are identical.

2.3.2 Attacks to Digital Signatures

We describe the classification of different attacks to digital signature schemes by adopting the definitions in [80].

Attacks

We distinguish two basic kinds of attacks:

No Message Attack : The adversary knows only the public key of the signer.

Known Message Attack : The adversary has access to some signature corresponding to known or chosen messages before his attempt to break the scheme.

According to the way how the messages are chosen, whose corresponding signatures are accessed by adversary, the **Known Message attack** can be classified into four different types:

Plain Known Message Attack : The adversary is given access to signatures for a set of messages, which are not chosen by him.

Generic Chosen Message Attack : The adversary is allowed to obtain the signatures corresponding to the messages, which are chosen by him. But the attacks should be independent of the signer's public key, which means the choice of the messages must be made before (s)he knows the public key of the signer.

Oriented Chosen Message Attack : This is similar to the generic chosen message attack. But the choice of the messages can be made after the adversary knows the public key of the signer, but before (s)he obtains any signature.

Adaptive Chosen Message Attack : The adversary can ask the signer to sign any message that (s)he wants, while (s)he knows the public key of the signer. (S)he can also adapt his queries according to previous obtained signature.

Forgeries

The expected results of an attack can be considered into four different types:

Total Break : The private key of the signer is disclosed, which is the most serious attack.

Universal Forgery : An efficient signing algorithm is found to be functionally equivalent to the original signing algorithm.

Selective Forgery : The signature of a particular message chosen by the adversary can be forged.

Existential Forgery : The signature for at least one message can be forged, but the adversary has no control over the message whose signature (s)he obtains. So in many cases this attack is not dangerous, because the output message is likely to be meaningless.

We adopt (s)he standard notion of security for a signature scheme which is called *existential unforgeability under a adaptive chosen message attack* [48, 79].

2.3.3 Example of Digital Signature Schemes

Here we present three well-known digital signature schemes, which are related to our proposed credential schemes. The message described below is regarded as the hash value of the message to be signed.

RSA Digital Signature Scheme

The RSA signature scheme [84], introduced in 1978, is the first method discovered for creating signature, which is described as follows:

Key Generation : Select two large prime p and q . Compute $n = p \cdot q$, and $m = (p - 1) \cdot (q - 1)$. Choose a number $e, 1 \leq e \leq m - 1$ such that $\gcd(e, m) = 1$ to ensure that there always exists an inverse with respect to modulo m . Find $d = e^{-1} \pmod{m}$. Then the public key is (e, n) , and private key is (d, p, q) .

Sign : to sign a message M , simply compute the signature δ as:

$$\delta = H(M)^d \pmod{n}$$

Verify : to verify the signature, output **true** if the following equation holds:

$$M \stackrel{?}{=} \delta^e \pmod{n}$$

ElGamal Family Digital Signature Schemes

Since the publication of ElGamal signature scheme [40] in 1985, a number of ElGamal variants are proposed in the literature. The following six type are taken from [69].

Key Generation : Choose a large prime p and a generator g of group \mathbb{Z}_p . Select a random number $s, 1 < s < p - 1$ as the secret. Compute $y = g^s \pmod{p}$. Then the public key is (p, g, y) , and private key is s .

Sign : to sign a message M , the signer first chooses a random number $r < p - 1$ such that $\gcd(r, p - 1) = 1$ to ensure that there always exists an inverse with respect to modulo $(p - 1)$. Compute the signature pair $\delta = \langle \alpha, \beta \rangle$ as:

$$\alpha = g^r \pmod{p}$$

$$\beta = (M - s \cdot \alpha) \cdot r^{-1} \pmod{p - 1} \quad (2.2)$$

$$\beta = (M - r \cdot \alpha) \cdot s^{-1} \pmod{p - 1} \quad (2.3)$$

$$\beta = s \cdot \alpha + r \cdot M \pmod{p - 1} \quad (2.4)$$

$$\beta = s \cdot M + r \cdot \alpha \pmod{p - 1} \quad (2.5)$$

$$\beta = (\alpha - r \cdot M) \cdot s^{-1} \pmod{p - 1} \quad (2.6)$$

$$\beta = (\alpha - s \cdot M) \cdot r^{-1} \pmod{p - 1} \quad (2.7)$$

Verify : to verify the signature, receiver checks following equations, and outputs true if the following equations hold:

$$g^M \stackrel{?}{=} y^\alpha \cdot \alpha^\beta \pmod{p} \quad \text{for variant (2.2)}$$

$$g^M \stackrel{?}{=} y^\beta \cdot \alpha^\alpha \pmod{p} \quad \text{for variant (2.3)}$$

$$g^\beta \stackrel{?}{=} y^\alpha \cdot \alpha^M \pmod{p} \quad \text{for variant (2.4)}$$

$$g^\beta \stackrel{?}{=} y^M \cdot \alpha^\alpha \pmod{p} \quad \text{for variant (2.5)}$$

$$g^\alpha \stackrel{?}{=} y^\beta \cdot \alpha^M \pmod{p} \quad \text{for variant (2.6)}$$

$$g^\alpha \stackrel{?}{=} y^M \cdot \alpha^\beta \pmod{p} \quad \text{for variant (2.7)}$$

DSA Digital Signature Scheme

The Digital Signature Algorithm (DSA) [75] was developed by NIST as a more efficient alternative to ElGamal signature.

Key Generation : Choose a large prime q and find a prime p of the form $kq + 1$ for a random k . Find a number g such that $g^q = 1 \pmod{p}$. Select a random number $s, 1 < s < p - 1$ as the secret. Compute $y = g^s \pmod{p}$. Then the public key is (p, q, g, y) , and private key is s .

Sign : The signer first chooses a random number $r < q$ such that $\gcd(r, q) = 1$ to ensure that there always exists an inverse with respect to modulo q . Compute the signature pair $\delta = \langle \alpha, \beta \rangle$ as:

$$\begin{aligned}\alpha &= (g^r \pmod{p}) \pmod{q} \\ \beta &= (M + s \cdot \alpha) \cdot r^{-1} \pmod{q}\end{aligned}$$

Verify : to verify the signature, the receiver first computes

$$\begin{aligned}w &= \beta^{-1} \pmod{q} \\ t_1 &= (M \cdot w) \pmod{q} \\ t_2 &= \alpha \cdot w \pmod{q}\end{aligned}$$

then outputs true if the following equation holds:

$$\alpha \stackrel{?}{=} ((g^{t_1} \cdot y^{t_2}) \pmod{p}) \pmod{q}$$

2.4 Identity-Based Encryption

The concept of identity-based cryptography was first proposed by Shamir [88] in 1984. In his paper, Shamir asked for a new model of asymmetric cryptography in which the public key can be an arbitrary string. In this system, the sender can use any identity information (e.g., e-mail address) of the receiver instead of some meaningless public key to encrypt the message, which is very useful in simplifying the certificate management.

As shown in Fig. 2.5, when Alice wants to send message to Bob, she can choose any identity information of Bob such as e-mail or IP address to encrypt the message. When Bob receives the encrypted message, he authenticates himself to a trusted third party and obtains the private key corresponding to his identity information to decrypt

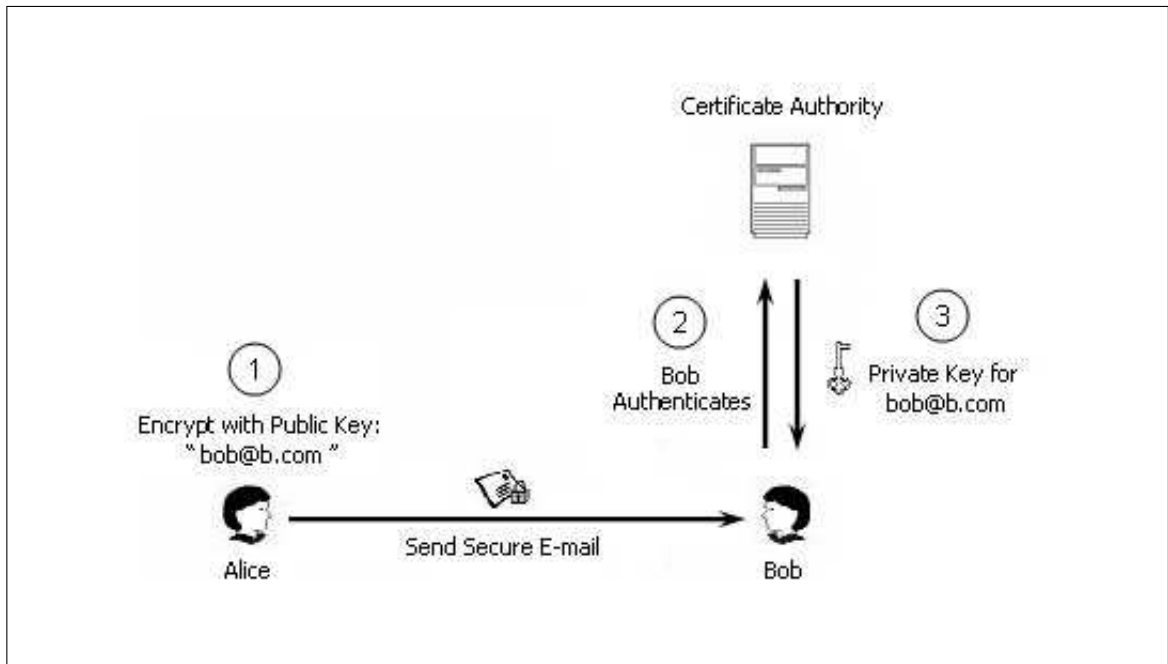


Figure 2.5: Identity-Based Encryption

the message. In this way, Alice can send encrypted information to Bob even if Bob has not setup his public key certificate yet.

Since the introduction of the concept in 1984, IBE had long been viewed as an open problem until it was solved by Boneh and Franklin [19] in 2001. Boneh and Franklin, for the first time, proposed a concrete construction of the IBE scheme based on bilinear maps, which used a technique due to Fujisaki-Okamoto [45] to achieve the chosen ciphertext security [7, 35, 82].

The IBE scheme proposed by them consists of four algorithms: **Setup**, **Extract**, **Encrypt**, **Decrypt**, which are described as follows:

Setup. The algorithm takes as input a security parameter and outputs a master key s and a list of system parameters. The system parameters are publicly known, while the master key s will be known only to the trusted third party, called Private Key Generator (PKG).

Extract. The algorithm takes as input the system parameters, the master key s , an arbitrary string $ID \in \{0, 1\}^*$, and outputs a private key d . ID can be any identity information of the user, or the public key derived from the identity, which will be used as a public key. After running this algorithm, a corresponding private

key d will be extracted from the given public key.

Encrypt. The algorithm encrypts a message M by using the public key ID, and returns a ciphertext C .

Decrypt. The algorithm takes as input a ciphertext C and the private key d , and outputs the original message M .

Because the identity-based cryptography can reduce the system complexity and the costs for generating, managing and storing the public key certificate, it is now flourishing within the research community.

2.5 Key Agreements

In a secret key cryptosystem, two or more parties may need to construct a shared key for communication. The key agreement protocols, also called key exchange, enable users to share keys securely over any insecure medium, without the need for a previously-established shared secret. For example, when Alice and Bob want to communicate using an symmetric encryption algorithm, they will need to run the key agreement protocol to establish a common secret key that can be useful for subsequent encryption and decryption of messages.

2.5.1 Diffie-Hellman Key Exchange

Diffie and Hellman [34] proposed a simple key exchange scheme in 1976, so called Diffie-Hellman (DH) key exchange. Assume that Alice and Bob want to construct a shared secret key, as shown in Fig. 2.6, the original DH key exchange scheme proceeds as follows:

- Alice and Bob agree on a common prime p and a generator g of the multiplicative group \mathbb{Z}_p^* .
- Alice picks $k_A \xleftarrow{R} \mathbb{Z}_p$, computes $g^{k_A} \pmod{p}$, and sends to Bob.
Bob picks $k_B \xleftarrow{R} \mathbb{Z}_p$, computes $g^{k_B} \pmod{p}$, and sends to Alice.
- Now Alice can compute the shared secret $s = (g^{k_B})^{k_A} \pmod{p}$ and
Bob can compute the shared secret $s = (g^{k_A})^{k_B} \pmod{p}$.

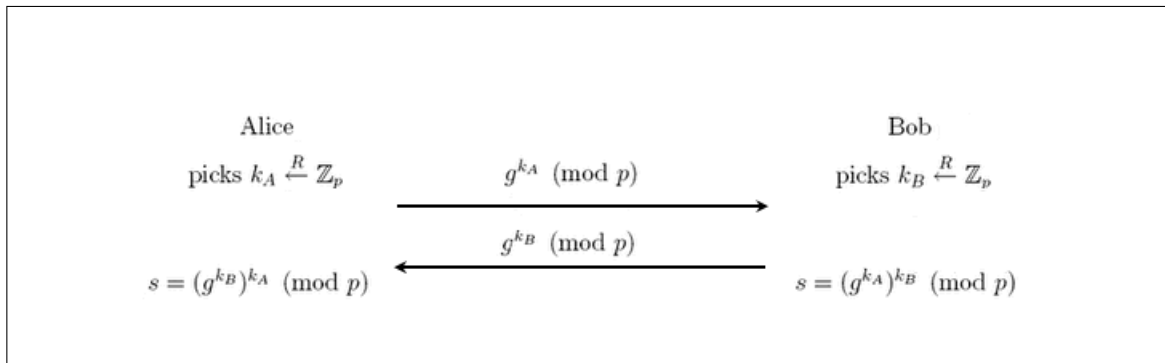


Figure 2.6: Diffie-Hellman Key Agreement

From the discussion above we can see that Alice and Bob share the same secret after running the key exchange protocol. Security of the system depends on the difficulty of computing discrete logarithm.

However, the original DH key exchange suffers from the man-in-the-middle attack. In this attack, an adversary Eve intercepts the message which Alice sends to Bob, substitutes it with Eve's own message and sends it to Bob. When Bob sends his message to Alice, Eve replaces the message and sends it to Alice. Eve and Alice thus agree on one shared key, while Eve and Bob agreeing on another shared key. In this way, Eve now can eavesdrop all the messages communicated between Alice and Bob without being discovered. Therefore, in recent years, DH key exchange is only treated as an traditional example of two-party key agreement protocols in most of time.

2.5.2 Group Key Agreements

Since the publication of the DH key exchange [34], many solutions have been proposed to extend DH key exchange to the multi-party setting. Notable solutions, which can be viewed as the first group key agreement schemes, have been proposed by Ingemarsson *et al.* [55] in 1982.

Group key agreement is a protocol that allows a group of users communicating over an insecure, open network to create a common session key. This session key, which is only known to the users who are the valid members of the group, may later be used to facilitate the communication among these users. By using group key agreement protocols, the presence of a central authority is no longer required. Moreover, when the group composition changes, one can employ supplementary key agreement protocols to obtain a new group key. Thus, a transient secure channel can be constructed during the lifetime of one session of a group.

Authenticated group key agreement is a group key agreement protocol ensured with an authentication mechanism, which is used to guarantee that no other users aside from the valid members of the group can learn any information about the session key. Authenticated group key agreement can be classified into two categories: Certificate-based and ID-based [39].

The certificate-based protocols work by assuming that each user has a (long-term) public/private key pair, and each user knows the public key of each other. Thus, the problem of authenticating the session key is replaced by the problem of authenticating the long-term public keys. Hence, in a certificate-based system, the participants *must* firstly verify the certificate of the user *before* using the user's public key. Consequently, the system requires a large amount of computing time and storage.

The ID-based protocols allow each user using identities (IDs) of other users as their public keys. Compared with the certificate-based group key agreement, users do not need to search for the public keys of other users before running the protocol, which makes the protocol simpler and more efficient.

2.6 Credential Systems

Recently, several separate credential systems were proposed which have very similar features. The most typical properties of these systems are that they allow credential contents to be used directly in access control processes, making the credentials in the systems can be used without ever being disclosed. However, most of these systems do not allow users to generate their own credentials, which are needed to be issued by some trusted third parties.

2.6.1 Oblivious Signature-Based Envelopes (OSBE)

Oblivious Signature-Based Envelopes (OSBE) system was first proposed by Li, Du and Boneh [66] in 2003, which was very similar to the IBE system. In this system, when a sender sends an encrypted message to a receiver, the receiver can decrypt the message only if (s)he has a third party's signature (e.g., a signature from a certificate authority) on a previous agreed-upon message. But the sender will never know whether the receiver can decrypt the message or not. Meanwhile, no other party can have the knowledge of the sender's message and the receiver's possession or lack of the credential. Consequently, the receiver can prove to have the credential but do not need to disclose it to others.

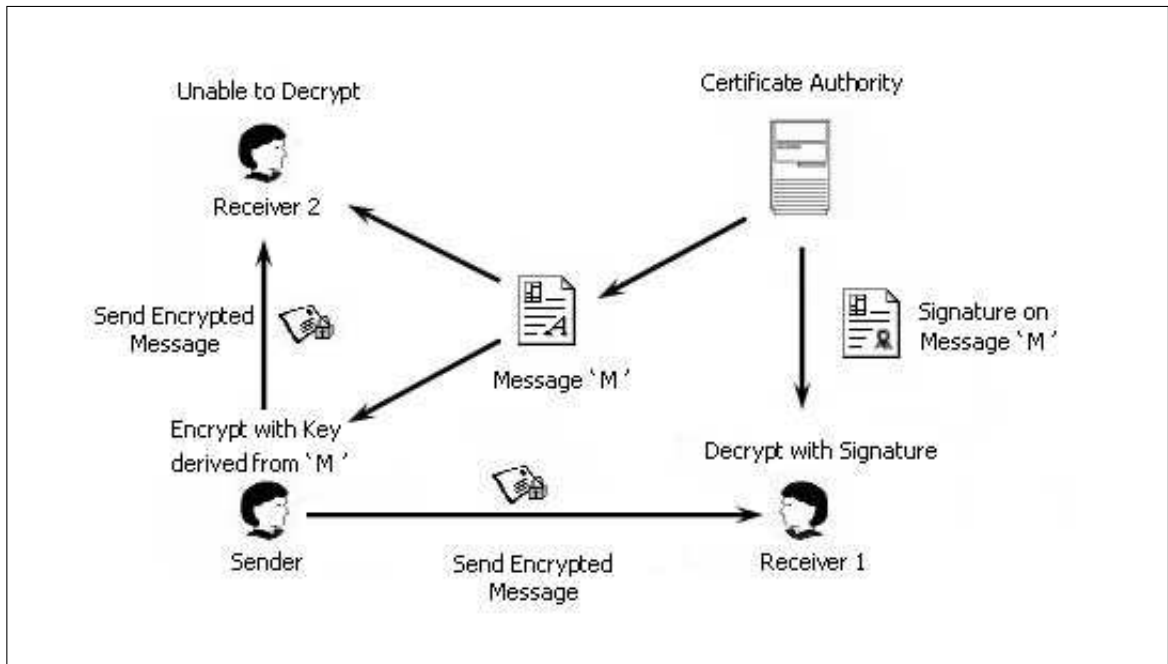


Figure 2.7: Oblivious Signature-Based Envelopes

As shown in Fig. 2.7, there are four parties involved in an OSBE scheme: certificate authority CA, sender S , receiver R_1 , and receiver R_2 . Receiver R_1 has the CA's signature on message M . Receiver R_2 does not have the signature. OSBE consists of three communication phases among these four parties, **Setup**, **Interaction** and **Open**, which are described as follows:

Setup. CA generates the system parameters required by taking the security parameter and two messages M and P as input. Then CA runs key generation algorithm to create a signing key and the corresponding public key denoted by PK .

The CA keeps the signing key as secret, gives the message P to S , and gives the system parameters, the public key PK and the message M to S , R_1 , and R_2 . Then CA signs the message M and gives the signature δ to R_1 .

Interaction. In this phase, the sender S communicates with R_1 and R_2 . S encrypts the message P and sends to R_1 and R_2 separately.

Open. After receiving the encrypted message from S , R_1 can output the message P , but R_2 cannot.

OSBE scheme is very useful in anonymous communication since it allows the receiver to obtain the information from the sender but never need to worry about disclosure of his own credential.

2.6.2 Secret Handshakes (SH)

The secret handshake scheme introduced by Balfanz *et al.* [5] allows two members of the same group to identify each other secretly, but if one party does not belong to the group, (s)he will learn nothing about group affiliation of the other party. One scenario for SH is shown in Fig. 2.8: a CIA agent Alice wants to authenticate herself to Bob, but Alice does not know whether Bob is a CIA agent or not. If Alice shows Bob her credential directly, her CIA identity will be revealed to Bob, who could be an adversary. The situation will be different if Alice authenticates with Bob via an SH protocol; namely, Bob will learn nothing about Alice's identity if (s)he is not a CIA agent. Alice will never worry about the leakage of her CIA affiliation no matter whom she authenticates to, even with other CIA servers. Even if there is an adversary Eve eavesdropping on the communication channel between Alice and Bob, he will never know the group affiliation of either Alice or Bob.

This secrecy property can be extended to ensure that group members' affiliations are revealed only to members who hold specific roles in the group. For example, Alice wants to authenticate to Bob if and only if Bob is a CIA agent with security level one, while herself as a CIA agent with security level two. Another important property of the handshake is that even if a third party Eve observes the exchange between Alice and Bob, (s)he can learn nothing about the process including whether Alice and Bob belong to the same group, the specific identities of the group, and the roles of either Alice or Bob.

The secret handshake scheme defined in [5] consists of five probabilistic algorithms `CreateGroup`, `AddUser`, `Handshake`, `TraceUser`, and `RemoveUser`.

- **CreateGroup**, a key generation algorithm executed by the group administrator CA, on input of **params**, outputs the group public key G , and the CA's private key s_G .
- **AddUser** is an algorithm executed between a group member and CA on CA's private key s_G and shared inputs: **params**, G , and the identity of the group member which is bit string ID of size regulated by **params**. After performing the algorithm, the group member will be issued a secret credential produced by CA for

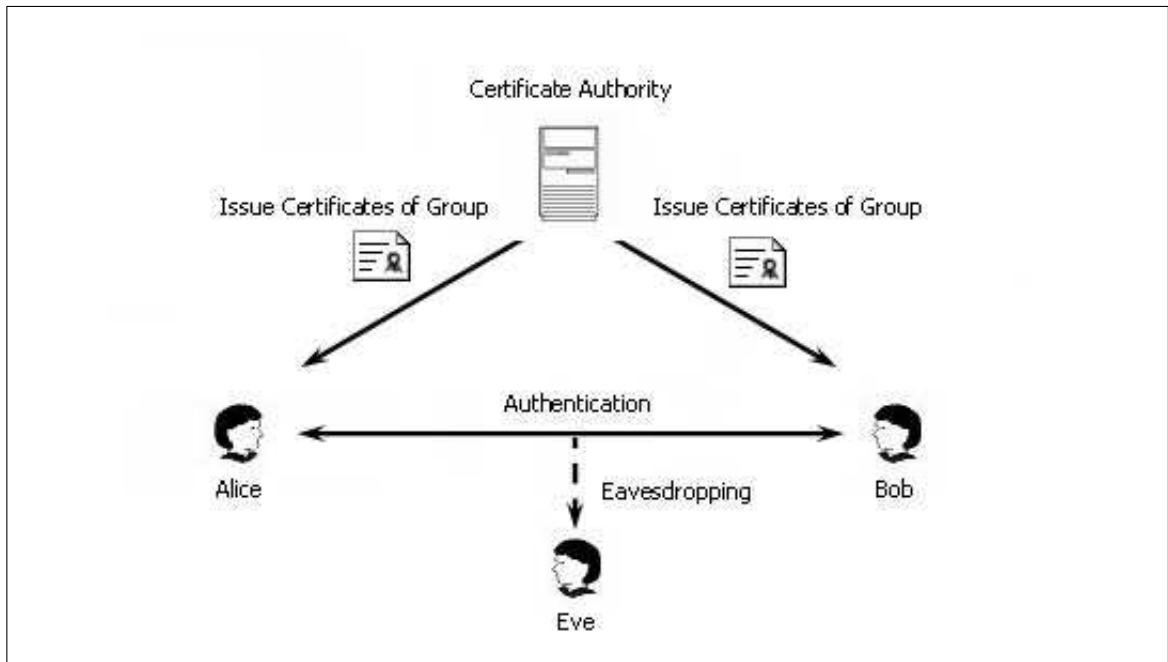


Figure 2.8: Secret Handshake

the member's identity ID .

- **HandShake** is the authentication protocol, executed between two parties A and B who want to authenticate each other on the public input ID_A , ID_B , and **params**. The private input of each party is their secret credential, and the output of the protocol for either party is either *reject* or *accept*.
- **TraceUser**, an algorithm executed by the CA, given a transcript of the handshake between a user U with one or more users, outputs the identity of the user U .
- **RemoveUser**: on input an identity of the user U and the revoked user list, inserts U into the revoked user list.

Secret handshakes can be used to securely discover services which are restricted to authorised users. It can also be used for privacy-preserving authentication. The use of secret handshakes does not required users to blind, or withhold, part of their credentials in order to achieve privacy. Instead, users can present all their credentials,

and rest assured that the receiving party will not learn anything about credentials that were issued by a different group.

2.7 Reconciling Key Agreements, OSBE and Secret Handshakes

From what we discussed above, these systems can be converted from one to another, so if a new scheme is introduced, it can be easily used to construct the scheme of other systems, which makes finding a more efficient way to implement above system easier. Holt discussed the conversion among several credential systems in [53]. Here we focus on the transformation among three systems: Key Agreements, OSBE, and Secret Handshakes.

2.7.1 Secret Handshakes from Key Agreements

In an SH protocol, completing the protocol is essentially equivalent to computing a key that is particular to the two interacting group members. Thus the SH scheme can be viewed as variant of key agreement scheme. However, not every key agreement schemes are suitable to be converted into an SH scheme directly. If we select a key agreement scheme to construct an SH scheme, it should satisfy the following properties:

ID-based Authentication : The secret handshake should be performed by the members in the same group, so the protocol requires each participant authenticates his identity before the exchange of the key. Thus a non ID-based key agreement scheme cannot be used directly to construct the secret handshake scheme.

Collusion Resistance : In a secret handshake protocol, a coalition of corrupted group members should not be able to perform the handshakes with group members outside the coalition. For example, a scheme based on shared group key is not collusion resistance. Furthermore, this scheme even has the additional drawback of untraceable key leaks.

In addition, when we convert a key agreement scheme into a secret handshake scheme, the new scheme should be able to protect the identities of the participants from both eavesdroppers and active attackers, which usually is not the option for key agreement scheme. For example, the standardised key exchange scheme IKE [52]

does not provide identity protection from active attackers, and [1] provides protection against eavesdroppers but does not preserve the privacy of the communicating parties.

2.7.2 Secret Handshakes from OSBE

The concept of OSBE are very closed to SH. From the definitions described above, we can observe that OSBE can be viewed as a sort of one-side or asymmetric secret handshakes. The simplest approach is to combine two OSBE interactions, which are in opposite direction, to obtain a secret handshake. This kind of transformation was shown in [72], and we will discuss the details in the following chapters.

2.8 Summary

Credential systems also include Hidden Credential [54], which is very similar to the Identity-based encryption. However, Hidden Credential is more like a policy based protocol. The sender in Hidden Credential encrypts message based on particular policy or even complex policies [21, 44] instead of simply using the identity. We only focus on identity-based cryptography in this thesis, so we do not refer to any Hidden Credential schemes and related policy-based protocol.

In this chapter, we described two kinds of credential systems: OSBE and SH. Since the credentials used in these two systems are based on digital signatures, we also discussed the classification of the attacks to the signature schemes. The security of credential schemes relies on how strong the signature schemes are. Then we included the key agreement protocols which can be used to construct the SH schemes. Finally, we discussed the relationship between these three systems, and the possibility of converting the schemes from one to another.

Chapter 3

Existing Cryptographic Schemes

In 2003, Secret Handshake was introduced by Balfanz *et al.* [5]. They also proposed a secret handshake scheme based on the key agreement protocol of Sakai *et al.* [85]. Li, Du, and Boneh introduced the concept of OSBE, and implemented the schemes in three ways. One is based on RSA signature [84] in which the two parties need to communicate with each other before the receiver can decrypt the message. The other two use the idea of the IBE systems: Boneh-Franklin [19] and Cocks [33] IBE systems, which are based on BLS signature [20] and Rabin signature [81] respectively. Since the introduction of these two credential systems, a flurry of papers have been written in this new area of research. Recently, SH has been extended to a multi-party setting by Tsudik and Xu [92]. In this chapter, we aim to briefly review the existing schemes of these credential systems, together with several ID-based group key agreement schemes, which can be used to construct the multi-party SH scheme.

3.1 Oblivious Signature-Based Envelopes

In this section, we do not include the OSBE schemes based on IBE systems, because they are not as efficient as schemes described following. Notations which will be used throughout this section is defined as follows: \mathcal{E} denotes a semantically secure symmetric encryption scheme [51], and H' is a cryptographic hash function for key derivation, which maps a number in an abelian group of integers to the key space of \mathcal{E} . H is a hash function which maps an arbitrary string $\{0, 1\}^*$ to a suitable commutative group of integers.

3.1.1 OSBE based on RSA signature

The first proposed scheme is the RSA based OSBE scheme [66], which is as follows:

Setup. CA runs the RSA key generation algorithm to create an RSA key (n, e, d) , then sends S, R_1, R_2 the RSA public key (n, e) and the message M . In addition, the CA gives the signature $\delta = (h^d \bmod n)$ to R_1 , where $h = H(M)$, and gives to S the message P .

Interaction.

- $R_1 \rightarrow S$: $\eta = \delta h^x \pmod{n}$, in which $x \xleftarrow{R} [1, n]$.
 $R_2 \rightarrow S$: $\eta = h^{x'} \pmod{n}$, in which $x' \xleftarrow{R} [1, n]$.
- S receives η , checks that $\eta \notin \{0, 1, n-1\}$, picks $z \xleftarrow{R} [1, n]$, computes $\gamma = \eta^{ez} h^{-z} \pmod{n}$ and then
- $S \rightarrow R_1$ or R_2 : $\zeta = h^{ze} \pmod{n}$, $C = \mathcal{E}_{H'(\gamma)}[P]$.
- R_1 and R_2 receive $\langle \zeta, C \rangle$ from S .

Open. R_1 computes $\gamma' = \zeta^x \pmod{n}$, and decrypts C using $H'(\gamma')$.

To see that this scheme is sound, observe that when η is sent by R_1 , $\eta = h^{d+x} \pmod{n}$; therefore:

$$\gamma = \eta^{ze} h^{-z} = h^{(d+x)ez} h^{-z} = h^{dez} h^{xez} h^{-z} = h^{xze} = \zeta^x = \gamma' \pmod{n}$$

Thus S and R_1 share the same symmetric key.

3.1.2 OSBE based on Schnorr signature

Recently, a series of OSBE schemes was proposed by Nasserian and Tsudik [72]. First we describe the OSBE scheme based on Schnorr signature [86] presented by them.

Setup. CA runs the key generation algorithm to create a Schnorr key (p, q, g, y, s) , sends S, R_1, R_2 the Schnorr public key (p, q, g, y) and the message M . Then it computes a Schnorr signature $\delta = \langle \alpha, \beta \rangle$ on message M as follows:

$$\alpha = H(M, g^k \bmod p), \text{ where } k \in_R \mathbb{Z}_q^*,$$

$$\beta = s\alpha + k \pmod{q}$$

Finally, it gives δ to R_1 , and gives the message P to S .

Interaction.

- $R_1 \longrightarrow S: \eta = g^\beta \cdot y^{-\alpha} \pmod{p} = g^k \pmod{p}$.
 $R_2 \longrightarrow S: \eta = g^{k'} \pmod{p}$, in which $k' \xleftarrow{R} \mathbb{Z}_q^*$.
- S receives η , checks that $\eta^{(p-1)/q} \pmod{p} \notin \{0, 1\}$, picks $z \xleftarrow{R} \mathbb{Z}_q$ with $z \bmod q \neq 0$, computes $\gamma = (y^{H(M, \eta)} \eta)^z$ and then
- $S \longrightarrow R_1 \text{ or } R_2: \zeta = g^z \pmod{p}, C = \mathcal{E}_{H'(\gamma)}[P]$.
- R_1 and R_2 receive $\langle \zeta, C \rangle$ from S .

Open. R_1 computes $\gamma' = (\zeta^\beta \bmod p)$, and decrypts C using $H'(\gamma')$.

To see that this scheme is sound, observe that when η is sent by R_1 :

$$\gamma = (y^{H(M, \eta)} \eta)^z = (y^\alpha g^k)^z = (g^{s\alpha + k})^z = g^{\beta z} = \zeta^\beta = \gamma' \pmod{p}$$

Thus S and R_1 share the same symmetric key.

3.1.3 OSBE based on Nyberg/Rueppel signature

Nasserian and Tsudik also introduced an OSBE scheme based on Nyberg/Rueppel signature [74] as follows:

Setup. CA runs the key generation algorithm to create a Nyberg-Rueppel key (p, q, g, y, s) , sends S, R_1, R_2 the Schnorr public key (p, q, g, y) and the message M . Then it sets $h = H(M)$ and computes a Schnorr signature $\delta = \langle \alpha, \beta \rangle$ on message M as follows:

$$\alpha = hg^{-k} \pmod{p}, \text{ where } k \in_R \mathbb{Z}_q^*, \alpha \pmod{q} \neq 0$$

$$\beta = s\alpha + k \pmod{q}$$

Finally, it gives δ to R_1 , and gives the message P to S .

Interaction.

- $R_1 \longrightarrow S$: $\eta = hg^{-k} \pmod{p} = \alpha$.
- $R_2 \longrightarrow S$: $\eta = hg^{-k'} \pmod{p}$, in which $k' \xleftarrow{R} \mathbb{Z}_q^*$.
- S receives η , checks that $(\eta/h)^{(p-1)/q} \pmod{q} \notin \{0, 1\}$, picks $z \xleftarrow{R} \mathbb{Z}_q$, computes $\gamma = (y^\alpha h / \alpha)^z$ and then
- $S \longrightarrow R_1$ or R_2 : $\zeta = g^z \pmod{p}$, $C = \mathcal{E}_{H'(\gamma)}[P]$.
- R_1 and R_2 receive $\langle \zeta, C \rangle$ from S .

Open. R_1 computes $\gamma' = (\zeta^\beta \pmod{p})$, and decrypts C using $H'(\gamma')$.

To see that this scheme is sound, observe that when η is sent by R_1 :

$$\gamma = (y^\alpha h / \alpha)^z = (y^\alpha g^k)^z = (g^{s\alpha+k})^z = g^{\beta z} = \zeta^\beta = \gamma' \pmod{p}$$

Thus S and R_1 share the same symmetric key.

3.1.4 OSBE based on ElGamal Family Signatures

Then we describe OSBE schemes based on ElGamal Family Signatures. Firstly, we focus on the variant (2.2) introduced in Section 2.3.3, since it represents the original ElGamal signature scheme [40] and also naturally leads to an OSBE scheme for DSA. The OSBE scheme is as follows:

Setup. CA runs the key generation algorithm to create an ElGamal key: (p, q, g, s, y) , sends S, R_1, R_2 the ElGamal public key (p, q, g, y) and the message M . It also chooses two messages M and P , set $h = H(M)$, and computes the ElGamal signature $\delta = \langle \alpha, \beta \rangle$ as follows:

$$\alpha = g^k \pmod{p}, \text{ where } k \in_R \mathbb{Z}_q^*,$$

$$\beta = (h - sg^k)k^{-1} \pmod{q}$$

Finally, it gives δ to R_1 , and gives the message P to S .

Interaction.

- $R_1 \longrightarrow S$: $\alpha = g^k \bmod p$
 $R_2 \longrightarrow S$: $\alpha = g^{k'} \bmod p$, in which $k' \xleftarrow{R} \mathbb{Z}_q^*$.
- S receives α , checks that $\alpha \bmod (p-1) \neq 0$, generates $z \in_R \mathbb{Z}_q^*$, computes $\gamma = y^{\alpha z} \cdot g^{-hz} \bmod p$.
- $S \longrightarrow R_1$ or R_2 : $\zeta = \alpha^z \bmod p$, $C = \mathcal{E}_{H'(\gamma)}[P]$.
- R_1 and R_2 receive $\langle \zeta, C \rangle$ from S .

Open. Upon receiving (Z, C) , R_1 computes $\gamma' = \zeta^{-\beta}$, and decrypts C with $H'(\gamma')$. where the correctness is easy to see that:

$$\gamma = y^{\alpha z} g^{-hz} = g^{(\alpha s - h)z} = g^{k(\alpha s - h)k^{-1}z} = \alpha^{-\beta z} = \zeta^{-\beta} = \gamma' \bmod p$$

The constructions of OBSE based on variants (2.4) and (2.5) were taken by Nasserian and Tsudik [72] as the example of OSBE scheme based on the variants of ElGamal signature. We describe these two schemes as followings:

Setup. The step is identical to that for ElGamal-based OSBE. CA computes the signature as follows:

$$\alpha = g^k \bmod p, \text{ where } k \in_R \mathbb{Z}_q^*,$$

$$\beta = sg^k + kh \bmod q \text{ for variant (2.4)}$$

$$\beta = sh + kg^k \bmod q \text{ for variant (2.5)}$$

Interaction.

- $R_1 \longrightarrow S$: $\alpha = g^k \bmod p$
 $R_2 \longrightarrow S$: $\alpha = g^{k'} \bmod p$, in which $k' \xleftarrow{R} \mathbb{Z}_q^*$.
- S receives α , picks $z \xleftarrow{R} \mathbb{Z}_q^*$, computes:

$$\gamma = (y^\alpha \cdot \alpha^h)^z = g^{z(sg^k + kh)} \bmod p \text{ for variant(2.4)}$$

$$\gamma = (y^h \cdot \alpha^\alpha)^z = g^{z(sh + kg^k)} \bmod p \text{ for variant(2.5)}$$

- $S \longrightarrow R_1$ or R_2 : $\zeta = g^z \bmod p$, $C = \mathcal{E}_{H'(\gamma)}[P]$.
- R_1 and R_2 receive $\langle \zeta, C \rangle$ from S .

Open. R_1 computes $\gamma' = (\zeta^\beta \bmod p)$, and decrypts C using $H'(\gamma')$.

3.1.5 OSBE based on DSA Signature

In order to avoid repetition, the full description of OSBE based on DSA signature was not presented in [72], and Nasserian and Tsudik only gave the arithmetic of computing the secret. However, their construction is flawed due to the confusion of the modular in DSA signature scheme. Firstly, we present the OSBE scheme based on DSA signature following the idea of Nasserian and Tsudik [72].

Setup. CA runs the key generation algorithm to create an DSA key: (p, q, g, s, y) , sends S, R_1, R_2 the DSA public key (p, q, g, y) and the message M . It also chooses two messages M and P , set $h = H(M)$, and computes the DSA signature $\delta = \langle \alpha, \beta \rangle$ as follows:

$$\alpha = (g^k \bmod p) \bmod q, \text{ where } k \in_R \mathbb{Z}_q^*,$$

$$\beta = (h + s \cdot \alpha) \cdot k^{-1} \bmod q$$

Finally, it gives δ to R_1 , and gives the message P to S .

Interaction.

- $R_1 \longrightarrow S$: $\alpha = (g^k \bmod p) \bmod q$
 $R_2 \longrightarrow S$: $\alpha = (g^{k'} \bmod p) \bmod q$, in which $k' \xleftarrow{R} \mathbb{Z}_q^*$.
- S receives α , checks that $\alpha \bmod q \neq 0$, generates $z \in_R \mathbb{Z}_q^*$, computes $\gamma = (y^\alpha g^h)^z \bmod p$.
- $S \longrightarrow R_1$ or R_2 : $\zeta = \alpha^z \bmod p$, $C = \mathcal{E}_{H'(\gamma)}[P]$.
- R_1 and R_2 receive $\langle \zeta, C \rangle$ from S .

Open. Upon receiving (Z, C) , R_1 computes $\gamma' = \zeta^\beta$, and decrypts C with $H'(\gamma')$.

Similar to the handshake scheme from ElGamal-OSBE, only if

$$\gamma \stackrel{?}{=} \gamma'$$

holds then the scheme is sound. Unfortunately, this verification is incorrect. We check first equation:

$$\gamma = (y^\alpha \cdot g^h)^z = g^{(\alpha s + h)z}$$

which is correct. The problem is because of ζ^β :

$$\gamma' \equiv \zeta^\beta = \alpha^{z\beta} = g^{k(\alpha s + h)k^{-1}z} = g^{(\alpha s + h)z}$$

The authors of the original paper [72] believe $\gamma = \gamma'$. However, the problem is as follows:

$$((g^k \bmod p) \bmod q)^{(\alpha s + h)k^{-1}z} \neq ((g^k)^{(\alpha s + h)k^{-1}z} \bmod p) \bmod q.$$

Obviously, the equality does not hold. Hence the DSA-OSBE scheme in [72] is flawed.

3.2 Two-Party Secret Handshakes

3.2.1 SH based on Pairing-Based Key Agreements

When the concept of Secret Handshake was proposed by Balfanz *et al* [5], a concrete implementation of SH was provided based on pairing cryptography, which is described as follows:

PBH.CreateGroup. The administrator CA runs parameter generation algorithm to generate a prime q , two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q , and a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Then CA picks two hash functions: $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1$ maps arbitrary strings to points in \mathbb{G}_1 , and H_2 is a collision resistant hash function taking arbitrary strings as input. In addition, CA also chooses a random generator $P \in \mathbb{G}_1$. Then CA picks a random $s \in \mathbb{Z}_q^*$, sets $P_{pub} = sP$. CA keeps s secret as the *master secret key* and publishes system parameters

$$\text{params} = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P_{pub}, P\}$$

PBH.AddUser. To add a user U to the group, the administrator CA first allocates a list of random “pseudonyms” $ID_{U_1}, \dots, ID_{U_t} \in \{0, 1\}^*$ for U , where t is chosen to be larger than the number of handshakes U will execute before receiving new user secret. The CA then computes a corresponding list of secret S_{U_1}, \dots, S_{U_t} as

$$S_{U_i} = sH_1(ID_{U_i})$$

PBH.Handshake. Let A and B be two users who would like to conduct the secret handshake. A chooses an unused pseudonym $ID_A \in \{ID_{A_1}, \dots, ID_{A_t}\}$, together with the corresponding secret S_{ID_A} . B chooses ID_B and S_{ID_B} likewise. Then they run the handshake protocol as follows:

- $A \longrightarrow B$: ID_A, k_A , where $k_A \xleftarrow{R} \{0, 1\}^*$
- $B \longrightarrow A$: ID_B, k_B, V_0
 $V_0 = H_2(\hat{e}(H_1(ID_A), S_{ID_B}) || ID_A || ID_B || k_A || k_B || 0)$

- $A \longrightarrow B: V_1$

$$V_1 = H_2(\hat{e}(S_{ID_A}, H_1(ID_B)) \| ID_A \| ID_B \| k_A \| k_B \| 1)$$

A verifies the V_0 and accepts only if the following equation holds

$$V_0 \stackrel{?}{=} H_2(\hat{e}(S_{ID_A}, H_1(ID_B)) \| ID_A \| ID_B \| k_A \| k_B \| 0)$$

B verifies the V_0 and accepts only if the following equation holds

$$V_1 \stackrel{?}{=} H_2(\hat{e}(H_1(ID_A), S_{ID_B}) \| ID_A \| ID_B \| k_A \| k_B \| 1)$$

If both verification succeed, then A and B finish all the steps of the SH, and the handshake has been successful.

PBH.TraceUser. Given a transcript of a handshake between user A and B , the administrator can easily recover the pseudonyms ID_A and ID_B and look up which users these pseudonyms had been issued to.

PBH.RemoveUser. To remove a user U from the group G , the administrator looks up the user secret $(ID_{U_1}, \dots, ID_{U_t}, S_{U_1}, \dots, S_{U_t})$ it has issued to U and publishes them on the revoked user list. Whenever other users perform the protocol, they can abort the handshake with the user whose secret is on the list.

3.2.2 SH based on CA-Oblivious Encryption

Castelluccia *et al.* introduced a new system, called CA-Oblivious Encryption [28, 29]. They constructed a concrete scheme by using the Schnorr signature as the credential. After running the protocol, each party can recover a ElGamal public key correspond to the secret of the other party, which can make the protocol be viewed as an asymmetric key exchange scheme. Based on the CA-Oblivious Encryption scheme, Castelluccia *et al.* proposed an SH scheme as follows:

CreateGroup. The administrator CA runs parameter generation algorithm to generate a standard discrete logarithm parameters (p, q, g) . g is a generator of a subgroup in \mathbb{Z}_p^* of order q . CA also defines hash functions: $H : \{0, 1\}^* \rightarrow \mathbb{Z}_q$ and $H' : \{0, 1\}^* \rightarrow \{0, 1\}^k$. Then CA picks random private key $s \in \mathbb{Z}_q$ and sets public key $y = g^s \bmod p$. \mathcal{E} denotes the ElGamal encryption scheme.

AddUser. To add a user U to the group, the administrator computes the Schnorr signature $\delta = \langle \alpha, \beta \rangle$ on his identity ID_U as follows:

$$\alpha = g^k \pmod{p}, \text{ where } k \in_R \mathbb{Z}_q,$$

$$\beta = sH(\alpha, \text{ID}_U) + k \pmod{q}$$

Handshake. Let A and B be two users who would like to conduct the secret handshake. Then they run the handshake protocol as follows:

- $B \longrightarrow A: \text{ID}_B, \alpha_B$
- $A \longrightarrow B: \text{ID}_A, \alpha_A, C_A, ch_A$
 A recovers $PK_B = \alpha_B y^{H(\alpha_B, \text{ID}_B)}$
 A picks $r_A, ch_A \xleftarrow{R} \{0, 1\}^*$
 A computes $C_A = \mathcal{E}_{PK_B}[r_A]$
- $B \longrightarrow A: C_B, resp_B, ch_B$
 B recovers $PK_A = \alpha_A y^{H(\alpha_A, \text{ID}_A)}$
 B decrypts r_A from C_A and picks $r_B, ch_B \xleftarrow{R} \{0, 1\}^*$
 B computes $C_B = \mathcal{E}_{PK_A}[r_B]$
 B computes $resp_B = H'(r_A, r_B, ch_A)$
- $A \longrightarrow B: resp_A$
 A decrypts r_B from C_B
 A computes $resp_A = H'(r_A, r_B, ch_B)$

A verifies the $resp_B$ and accepts only if the following equation holds

$$resp_B \stackrel{?}{=} H'(r_A, r_B, ch_A)$$

B verifies the $resp_A$ and accepts only if the following equation holds

$$resp_A \stackrel{?}{=} H'(r_A, r_B, ch_B)$$

If both verification succeed, then A and B finish all the steps of the SH, and the handshake has been successful.

Castelluccia *et al.* [29] also eliminating one communication round in above protocol by using the zero-knowledge signature [26] and presented an efficient scheme. Thus SH based on CA-Oblivious Encryption can still be completed in three rounds.

3.2.3 SH based on RSA signature

Vergnaud presented two SH schemes based on RSA signature [93] in 2005.

OT-SH

The OT-SH was constructed from Okamoto and Tanaka's Identity-based key agreement [77] scheme, whose security relies on the RSA problem [63, 64, 67].

OT-SH.CreateGroup. The administrator CA runs the RSA key generation algorithm to create an RSA key (n, e, d) . Then CA picks a random $g \in \mathbb{Z}_n$, and chooses two hash functions: $H : \{0, 1\}^* \rightarrow \mathbb{Z}_n$, and H' is a collision resistant hash function taking arbitrary strings as input. CA keeps d secret as the *master secret key* and publishes system parameters $\text{params} = \{n, e, g\}$.

OT-SH.AddUser. To add a user U to the group, the administrator CA first allocates a unique identity $\text{ID}_U \in \{0, 1\}^*$ for U . The CA then computes a corresponding secret $S_{\text{ID}_U} = h_U^{-d} \pmod{n}$, where $h_U = H(\text{ID}_U)$, and sends it to U .

OT-SH.Handshake. Let A and B be two users who would like to conduct the secret handshake. They run the handshake protocol as follows:

- $A \longrightarrow B: \text{ID}_A, C_A$
 $C_A = S_{\text{ID}_A} \cdot g^{k_A} \pmod{n}$, where $k_A \xleftarrow{R} \{0, 1\}^*$
- $B \longrightarrow A: \text{ID}_B, C_B, V_0$
 $C_B = S_{\text{ID}_B} \cdot g^{k_B} \pmod{n}$, where $k_B \xleftarrow{R} \{0, 1\}^*$
 $V_0 = H'((h_A \cdot C_A^e)^{k_B} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel C_A \parallel C_B \parallel 0)$
- $A \longrightarrow B: V_1$
 $V_1 = H'((h_B \cdot C_B^e)^{k_A} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel C_A \parallel C_B \parallel 1)$

A verifies the V_0 and accepts only if the following equation holds

$$V_0 \stackrel{?}{=} H'((h_B \cdot C_B^e)^{k_A} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel C_A \parallel C_B \parallel 0)$$

B verifies the V_1 and accepts only if the following equation holds

$$V_1 \stackrel{?}{=} H'((h_A \cdot C_A^e)^{k_B} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel C_A \parallel C_B \parallel 1)$$

If both verification succeed, then A and B finish all the steps of the SH, and the handshake has been successful.

To check the correctness of the scheme, observe that

$$(h_A \cdot C_A^e)^{k_B} = (h_A \cdot h_A^{-de} \cdot g^{k_A e})^{k_B} = g^{k_A k_B e} = (h_B \cdot h_B^{-de} \cdot g^{k_B e})^{k_A} = (h_B \cdot C_B^e)^{k_A}$$

Gi-SH

The second scheme based on RSA signature is implemented from Girault's self-certified key agreement scheme [47], whose security is also based on the difficulty of solving RSA [76].

Gi-SH.CreateGroup. The administrator CA runs the RSA key generation algorithm to create an RSA key (n, e, d) . Then CA picks a random $g \in \mathbb{Z}_n$, and chooses two hash functions: $H : \{0, 1\}^* \rightarrow \mathbb{Z}_n$, and H' is a collision resistant hash function taking arbitrary strings as input. CA keeps d secret as the *master secret key* and publishes system parameters $\text{params} = \{n, e, g\}$.

Gi-SH.AddUser. To add a user U to the group, the administrator CA first allocates a unique identity $\text{ID}_U \in \{0, 1\}^*$ for U . The CA picks a random $r_U \xleftarrow{R} \mathbb{Z}_n$, and sends U the r_U as a secret. Then CA computes $P_U = (g^{r_U} \oplus h_U)^d \pmod{n}$, where $h_U = H(\text{ID}_U)$, and sets it to public.

Gi-SH.Handshake. Let A and B be two users who would like to conduct the secret handshake. They run the handshake protocol as follows:

- $A \longrightarrow B$: ID_A, k_A
 A picks a random $k_A \xleftarrow{R} \{0, 1\}^*$
- $B \longrightarrow A$: ID_B, k_B, V_0
 B picks a random $k_B \xleftarrow{R} \{0, 1\}^*$
 $V_0 = H'((P_A^e \oplus h_A)^{r_B} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel k_A \parallel k_B \parallel 0)$
- $A \longrightarrow B$: V_1
 $V_1 = H'((P_B^e \oplus h_B)^{r_A} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel k_A \parallel k_B \parallel 1)$

A verifies the V_0 and accepts only if the following equation holds

$$V_0 \stackrel{?}{=} H'((P_B^e \oplus h_B)^{r_A} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel k_A \parallel k_B \parallel 0)$$

B verifies the V_1 and accepts only if the following equation holds

$$V_1 \stackrel{?}{=} H'((P_A^e \oplus h_A)^{r_B} \pmod{n} \parallel \text{ID}_A \parallel \text{ID}_B \parallel k_A \parallel k_B \parallel 1)$$

If both verification succeed, then A and B finish all the steps of the SH, and the handshake has been successful.

To check the correctness of the scheme, observe that

$$(P_A^e \oplus h_A)^{r_B} = ((g^{r_A} \oplus h_A)^{de} \oplus h_A)^{r_B} = g^{r_A r_B} = ((g^{r_B} \oplus h_B)^{de} \oplus h_B)^{r_A} = (P_B^e \oplus h_B)^{r_A}$$

3.3 Group Key Agreement Schemes

In this section, we describe three group key agreement schemes related closely to the following chapters: one is a non-authenticated protocol in discrete logarithm cryptosystem, and two are ID-based authenticated protocols by using pairing-based cryptography. In the following description $H : \{0, 1\}^* \rightarrow \mathbb{G}_1$ and $H_1 : \mathbb{G}_1 \rightarrow \mathbb{Z}_q$ are cryptographic hash functions.

Burmeister-Desmedt Group Key Agreement

This group key agreement scheme is proposed by Burmeister-Desmedt [24] in 1994. The protocol runs as follows:

$GKA(\Delta)$: Let Δ denote the group of users involved in the group key agreement scheme. $\Delta = \{U_1, \dots, U_n\}$, where U_i are members of a group G that want to create a group key. g is a generator in \mathbb{Z}_p^* .

[Round 1] : Each user U_i picks a random $r_i \in_R \mathbb{Z}_q$ and then computes and broadcasts $z_i = g^{r_i} \bmod p$.

[Round 2] : On receiving z_i , each user U_i computes and broadcasts $X_i = (z_{i+1}/z_{i-1})^{r_i}$.

Then U_i can compute the same group key,

$$K_i = (z_{i-1})^{nr_i} \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdots X_{i-2} \pmod{p}$$

Choi-Hwang-Lee ID-based Group Key Agreement

Choi *et al.* [32] proposed a new ID-based AGKA scheme from bilinear maps, which is based on Burmeister-Desmedt group key agreement scheme [24].

Setup. GA runs BDH parameter generator to generate a prime q , two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q , and an bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Choose a random generator $P \in \mathbb{G}_1$. Then GA picks a random $s \in \mathbb{Z}_q^*$ and sets $P_{pub} = sP$. GA keeps s secret as the *master secret key* and publishes system parameters **params** = $\{\hat{e}, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, H, H_1\}$.

Extract. When a user U_i with identity ID_i wishes to obtain a key pair, GA computes $Q_i = H(ID_i)$ and the long-term private key $S_i = sQ_i$, and returns S_i to the user

U_i .

Let U_1, \dots, U_n be the n users who want to establish a session key. The protocol is as follows:

Round 1 Each user U_i picks a random number $r_i \in_R \mathbb{Z}_q^*$ and computes $P_i = r_i P$, and $h_i = H_1(P_i)$ and $T_i = r_i P_{pub} + h_i S_i$. Each U_i broadcasts $\langle P_i, T_i \rangle$ to all others and keeps r_i secret.

Round 2 Upon receiving $\langle P_{i-1}, T_{i-1} \rangle$, $\langle P_{i+1}, T_{i+1} \rangle$ and $\langle P_{i+2}, T_{i+2} \rangle$, each user U_i checks if the following equation holds:

$$\hat{e}(\sum_{k \in \{-1, 1, 2\}} T_{i+k}, P) \stackrel{?}{=} \hat{e}(\sum_{k \in \{-1, 1, 2\}} (P_{i+k} + h_{i+k} Q_{i+k}), P_{pub})$$

If the above equation is satisfied, then U_i computes and broadcasts

$$X_i = \hat{e}(r_i(P_{i+2} - P_{i-1}), P_{i+1})$$

to all others.

Key Computation. Each U_i now can compute the common session key as follows:

$$K_i = \hat{e}(r_i P_{i-1}, P_{i+1})^n \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdots X_{i-2}$$

However, the above scheme was shown to be flawed soon after the construction of the scheme. Zhang and Chen [100] proposed an impersonation attack on above scheme, in which any two malicious users can impersonate an entity to agree some session keys in a new group if these two malicious users have the previous authentication transcripts of this entity. Hence an fixed solution was proposed by Du *et al.* [37] later. In the improved ID-based AGKA scheme, each user in a group holds a synchronous counter, which is increased by one after a successful group key agreement, and the users' static key pairs are updated along with the counters, which make the scheme resist the collusive impersonation attack in [100].

Shi-Chen-Li ID-based Group Key Agreement

A single round ID-based AGKA protocol was proposed by Shi *et al.* [89] in 2005. However, we note that the scheme requires the user to keep a public key issued by the

group administrator (GA) and verify each other's public key before using it, which is the idea of a certificate-based group key agreement protocol. This means, that their scheme is *not* an ID-based scheme, rather than a public key based scheme. Moreover, the scheme in the paper [89] is flawed in fact.

Firstly, we review the one round ID-based AGKA in [89].

Setup. GA generates a prime q , two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q , and an bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Choose a random generator $P \in \mathbb{G}_1$. Then GA randomly picks $s_1, s_2 \in \mathbb{Z}_q^*$ and sets $P_{pub} = s_1 P, P'_{pub} = s_2 P$. GA then publishes system parameters $\text{params} = \{\hat{e}, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, P'_{pub}, H\}$.

Extract. When a user U_i with identity ID_{U_i} wishes to obtain a key pair, GA computes $I_i = H(ID_i)$, $Q_i = (I_i s_1 + s_2)P$ and the secret $S_i = (I_i s_1 + s_2)^{-1}P$, and returns S_i to the user U_i . Q_i is the user's public key.

Let U_1, \dots, U_n be the n users who want to establish a session key. The protocol is as follows:

Interacting Each user U_i picks $a_i \xleftarrow{R} \mathbb{Z}_p^*$. Then U_i computes $T_i^j = a_i Q_j$, where $1 \leq j \leq n$ and $j \neq i$. Now U_i can check public key of each user:

$$Q_j \stackrel{?}{=} I_j P_{pub} + P'_{pub}$$

If the above equation holds, U_i can assume that U_j is a valid member of the group. Then U_i sends T_i^j to U_j .

Key Computation. Upon receipt of T_j^i , each U_i now can compute the common session key as follows:

$$K = K_i = \hat{e}(T_1^i + \dots + T_{i-1}^i + a_i Q_i + T_{i+1}^i + \dots + T_n^i, S_i) = \hat{e}(P, P)^{(a_1 + \dots + a_n)}$$

We observe that if U_i wants to authenticate a user U_j , he has to obtain the public key of U_j in advance, which makes the scheme not an ID-based one. This scheme is indeed a public key based scheme. The public key needs to be certified, and hence, the advantage of having an ID-based protocol has vanished.

Next we show how to attack the protocol above. An adversary \mathcal{A} asks the GA for a key pair, and thus (s)he obtains $I_{\mathcal{A}} = H(ID_{\mathcal{A}})$, $Q_{\mathcal{A}} = (I_{\mathcal{A}} s_1 + s_2)P$ and the secret $S_{\mathcal{A}} = (I_{\mathcal{A}} s_1 + s_2)^{-1}P$, where $I_{\mathcal{A}} \neq I_i, 1 \leq i \leq n$. Then \mathcal{A} chooses T_i^1 and T_j^1 which are two messages sent to U_i and U_j by U_1 , and computes:

$$\begin{aligned}
a_1 s_1 P &= (I_i - I_j)^{-1} (T_i^1 - T_j^1) = (I_i - I_j)^{-1} (a_1 Q_i - a_1 Q_j) \\
&= (I_i - I_j)^{-1} (a_1 (I_i s_1 + s_2) P - a_1 (I_j s_1 + s_2) P) \\
&= (I_i - I_j)^{-1} (a_1 (I_i - I_j) s_1 P)
\end{aligned}$$

$$\begin{aligned}
a_1 s_2 P &= (I_i^{-1} - I_j^{-1})^{-1} (I_i^{-1} T_i^1 - I_j^{-1} T_j^1) \\
&= (I_i^{-1} - I_j^{-1})^{-1} (I_i^{-1} a_1 Q_i - I_j^{-1} a_1 Q_j) \\
&= (I_i^{-1} - I_j^{-1})^{-1} (I_i^{-1} a_1 (I_i s_1 + s_2) P - I_j^{-1} a_1 (I_j s_1 + s_2) P) \\
&= (I_i^{-1} - I_j^{-1})^{-1} (a_1 (I_i^{-1} - I_j^{-1}) s_2 P)
\end{aligned}$$

In the same way, \mathcal{A} can obtain $a_2 s_1 P, \dots, a_n s_1 P$ and $a_2 s_2 P, \dots, a_n s_2 P$. Then \mathcal{A} continues to compute:

$$T_i^{\mathcal{A}} = a_i s_1 P \cdot I_{\mathcal{A}} + a_i s_2 P = a_i (I_{\mathcal{A}} s_1 + s_2) P$$

\mathcal{A} now can compute the group session key as follows:

$$K = K_i = \hat{e}(T_1^{\mathcal{A}} + \dots + T_n^{\mathcal{A}}, S_{\mathcal{A}}) = \hat{e}(P, P)^{(a_1 + \dots + a_n)}$$

From the description above, we can observe that if there is a user who is belong to this group and has the valid group key pair, (s)he can know the group session key of any execution of the group key agreement protocol even if (s)he is not involved in it.

3.4 Group Secret Handshakes

A construction of GSH scheme was proposed by Jarecki *et al.* [56] in 2006, which extends the SH protocol to a multi-party setting based on Burmester-Desmedt group key agreement scheme [24]. The definition of SH requires that if a handshake among all participants fails, the group affiliation of each party will not be disclosed. However, we show an attack to Jarecki *et al.*'s scheme [56], which makes the honest parties involved in the protocol share a same session key, even if there is an adversary in the protocol.

Firstly, we review Jarecki *et al.*'s GSH scheme [56].

CreateGroup. The administrator \mathbf{GA} runs key generation algorithm, which takes as input a security parameter k , to generate the discrete logarithm parameters (p, q, g) . g is a generator of a subgroup in \mathbb{Z}_p of order q . Then \mathbf{GA} picks a random $s \xleftarrow{R} \mathbb{Z}_q^*$, sets it the group secret, and computes the public key $y = g^s \bmod p$.

AddUser. To add a user U to the group, the administrator **GA** first allocates a list of random “pseudonyms” $\text{ID}_{U_1}, \dots, \text{ID}_{U_t} \in \{0, 1\}^*$ for U , where t is chosen to be larger than the number of handshakes U will execute before receiving new user secret. The **GA** then computes a corresponding list of Schnorr signature $(\alpha_1, \beta_1), \dots, (\alpha_t, \beta_t)$, where $\alpha_k = g^{r_k} \pmod{p}$, and $\beta_k = r_k + sH(\alpha_k, \text{ID}_{U_k}) \pmod{p}$, for random $r_k \xleftarrow{R} \mathbb{Z}_q$.

Handshake. Let $\Delta = \{U_1, \dots, U_n\}$ be n users who would like to conduct the secret handshake. Each user U_i chooses an unused pseudonym $\text{ID}_i \in \{\text{ID}_1, \dots, \text{ID}_t\}$, together with the corresponding secret $\langle \alpha_i, \beta_i \rangle$. Then the group of users run the handshake protocol as follows:

Round 1: Each user U_i broadcasts (ID_i, α_i)

- When a user U_i finds a collision in the group of IDs , or finds a ID in the revoked user list, the protocol terminates.
- If there are no collision or revoked ID , U_i determines the order of each user based on their identities. Assume that the order of users in the group is (U_1, U_2, \dots, U_n) , and $U_{n+1} = U_1$.

Round 2: U_i computes

$$\begin{aligned} z_{i+1} &= \alpha_{i+1} y^{H(\alpha_{i+1}, \text{ID}_{i+1})} = g^{\beta_{i+1}} \pmod{p} \\ z_{i-1} &= \alpha_{i-1} y^{H(\alpha_{i-1}, \text{ID}_{i-1})} = g^{\beta_{i-1}} \pmod{p} \\ X_i &= H'(z_{i+1}^{\beta_i}) / H'(z_{i-1}^{\beta_i}) \pmod{p} \end{aligned}$$

Each U_i broadcasts X_i .

Round 3: U_i computes $K_i = H'(z_{i-1}^{\beta_i})^n \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdots X_{i-2} \pmod{p}$.

Then each user U_i in the group outputs “accept” if they hold a common shared key. All the steps of the SH are finished, and the handshake has been successful.

To see that this scheme is sound, let:

$$\begin{aligned} C_{i-1} &= H'(z_{i-1}^{\beta_i}) = H'(g^{\beta_{i-1}\beta_i}) \pmod{p} \\ C_i &= H'(z_{i-1}^{\beta_i}) \cdot X_i = H'(g^{\beta_i\beta_{i+1}}) \pmod{p} \\ C_{i+1} &= H'(z_{i-1}^{\beta_i}) \cdot X_i \cdot X_{i+1} = H'(g^{\beta_{i+1}\beta_{i+2}}) \pmod{p} \\ &\dots \\ C_{i-2} &= H'(z_{i-1}^{\beta_i}) \cdot X_i \cdot X_{i+1} \cdots X_{i-2} = H'(g^{\beta_{i-2}\beta_{i-1}}) \pmod{p} \end{aligned}$$

It is obvious that

$$\begin{aligned} K_i &= C_{i-1}C_iC_{i+1}\cdots C_{i-2} = H'(z_{i-1}^{\beta_i})^n \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdots X_{i-2} \\ &= g^{\beta_1\beta_2+\beta_2\beta_3+\cdots+\beta_n\beta_1} \pmod{p} \end{aligned}$$

Then if all the users involved in the protocol are valid, they will share the same symmetric key.

RemoveUser. To remove a user U from the group G , the administrator **GA** looks up the user pseudonyms $(\text{ID}_{U_1}, \dots, \text{ID}_{U_t})$ it has issued to U and publishes them on the revoked user list.

Note that if U_i is not a valid member of the group, the group secret handshake will not succeed, because U_i cannot produce a valid X_i without having the knowledge of β_i , which corresponds to the α_i broadcasted at the beginning of the protocol.

However, we observe that U_i can wait till (s)he receives all the X_j , where $j \neq i$, before (s)he broadcasts X_i in Round 2. Then U_i computes

$$\begin{aligned} X'_i &= \left(\prod_{j \in [1, n]}^{j \neq i} X_j \right)^{-1} = \left(\prod_{j \in [1, n]}^{j \neq i} (H'(z_{j+1}^{\beta_j})/H'(z_{j-1}^{\beta_j})) \right)^{-1} \\ &= \left(\prod_{j \in [1, n]}^{j \neq i} (H'(g^{\beta_{j+1}\beta_j})/H'(g^{\beta_{j-1}\beta_j})) \right)^{-1} = (H'(g^{\beta_{i-1}\beta_i})/H'(g^{\beta_{i+1}\beta_i}))^{-1} \\ &= H'(g^{\beta_{i+1}\beta_i})/H'(g^{\beta_{i-1}\beta_i}) = H'(z_{i+1}^{\beta_i})/H'(z_{i-1}^{\beta_i}) \end{aligned}$$

From the discussion above, we can see that U_i computes the valid value X'_i and broadcasts it to other parties. Without knowing β_i , U_i cannot calculate the common group key K_i , but we observe that other valid members still can share the same symmetric key. Then U_i can know that these parties belong to a same group, which makes the honest parties leak their group affiliation even if the handshake fails.

3.5 Summary

In this chapter, we first reviewed several OSBE schemes based on a series of signature schemes. Then we described some different constructions of SH schemes. We note that each signature scheme used in OSBE can also be used to implement an SH scheme. Even if the SH schemes based on some signatures have not been proposed formally yet, we can still convert an OSBE scheme based on these signatures into an SH scheme.

For example, Nasserian and Tsudik [72] proposed a generic construction of SH from OSBE based on ElGamal family signatures. We also reviewed some efficient group key agreement schemes. Though some of them were proved to be flawed, we can still consider the efficiency of these schemes and compare with other schemes proposed later. Finally, we introduced a GSH scheme proposed recently.

Chapter 4

Secret Handshake Schemes based on ElGamal and DSA signatures

The SH system was introduced by Balfanz *et al.* [5] in 2003. In addition to introducing the concept of the SH, Balfanz *et al.* also presented an SH scheme based on pairing cryptography, where the computation is not as efficient as the schemes in discrete logarithm cryptosystem. Subsequently, an SH scheme based on CA-Oblivious Encryption was introduced by Castelluccia *et al.* [29]. In their scheme, ElGamal encryption and Schnorr signature are combined to construct a CA-oblivious PKI-enabled encryption secure under the CDH assumption. Based on this primitive, they proposed a new SH scheme. Xu and Yung [98] also proposed an SH schemes that achieve unlinkability with reusable credentials. However, their schemes require users to know the information of other groups. We do not discuss their schemes in this thesis. Another SH scheme based on RSA was proposed by Vergnaud [93]. Recently, a series of OSBE schemes based on the ElGamal family signatures schemes was proposed by Nasserian and Tsudik [72], where they also discussed the generic conversion from OSBE to SH schemes. According to their generic construction, any ElGamal family signatures based OSBE scheme can be converted to secret handshake within three communication moves, except the ElGamal and DSA signatures based ones. In this chapter, we propose two novel SH schemes based on ElGamal and DSA signature schemes [103], respectively. For the first time, we achieve three-move SH based on these two signature schemes, which was believed infeasible following the generic construction of Nasserian and Tsudik [72]. We also prove that our proposed schemes are secure. Part of this chapter has appeared in [103].

4.1 Security Arguments

Following the definition in [5], an SH scheme must satisfy the properties of completeness, impersonator resistant, and detector resistant. The adversary is allowed to run the protocols several times and be able to make additional queries after each attempt, before (s)he announces that (s)he is ready for the true challenges. The adversary is also allowed to ask for signatures on additional $ID_i \neq ID_A$ strings during the handshake protocol with honest member. (S)he can see all exchanged messages, can delete, modify, inject and redirect messages, can communicate with other party, and can reuse messages from past communications.

Completeness. If honest members A, B of the same group run Handshake with valid certificates from the group administrator, which are the signatures generated for their ID strings ID_A, ID_B and for the same group $G_A = G_B$, then both parties output “accept”.

Impersonator Resistance. The impersonator resistance property is violated if an honest party V who is a member of group G authenticates an adversary \mathcal{A} as a group member, even though \mathcal{A} is not a member of G . We denote the probability that the property is not violated as follows:

$$Pr[\mathcal{A} \text{ succeeds in making } V \text{ output “accept”} \mid V \in G \cap \mathcal{A} \notin G] \leq \varepsilon,$$

where ε is negligible.

Detector Resistance. An adversary \mathcal{A} violates the detector resistance property if it can decide whether some honest party V is a member of some group G by determining the relationship between the public message of the member and the public key of the group, even though \mathcal{A} is not a member of G . The probability that the property is not violated is as follows:

$$Pr[\mathcal{A} \text{ knows whether } V \text{ is the valid member} \mid \text{public messages of } V \cap \mathcal{A} \notin G] \leq \varepsilon$$

4.2 Constructions from previously proposed OSBE schemes

As we discussed above, an OSBE scheme can be easily converted into an SH scheme. Nasserian and Tsudik [72] have introduced a generic construction, and mentioned that the SH schemes transformed from ElGamal-OSBE and DSA-OSBE cannot be completed in three moves following their generic construction. Since we have already shown that the DSA-OSBE introduced by Nasserian and Tsudik [72] is not correct, even we transform it into a DSA-based SH, the scheme will not be correct as well. Hence we only review the SH scheme constructed from the ElGamal-based OSBE according to the generic construction of Nasserian and Tsudik [72].

CreateGroup. The administrator CA runs the ElGamal key generation algorithm to create the set of all keys $\{(p, g, y, s) \mid y \equiv g^s \pmod{p}\}$, in which s is the group secret.

AddUser. Select two collision-resistant cryptographic hash functions $H : \{0, 1\}^* \rightarrow \mathbb{Z}_{p-1}$, and $H' : \{0, 1\}^* \rightarrow \{0, 1\}^n$ for some n . To add a user U to the group, the administrator CA first allocates a unique identity ID_U to user, and generates a random nonce $r_U \in \mathbb{Z}_{p-1}$. The CA then computes the hash value $h_U = H(ID_U)$, and gives the user U the corresponding signature $\langle \alpha_U, \beta_U \rangle$, where $\alpha_U = g^{r_U} \pmod{p}$, $\beta_U = ((h_U - \alpha_U \cdot s) \cdot r_U^{-1}) \pmod{p-1}$.

Let A and B be two users who would like to perform the protocol.

Handshake.

- $A \rightarrow B$: ID_A, α_A
 $\alpha_A = g^{k_A} \pmod{p}$, where $k_A \in_R \mathbb{Z}_p^*$.
- $B \rightarrow A$: ID_B, α_B, ζ_B
 $\alpha_B = g^{k_B} \pmod{p}$, where $k_B \in_R \mathbb{Z}_p^*$
 B computes $\zeta_B = \alpha_A^{z_B}$, where $z_B \in_R \mathbb{Z}_p^*$
 B computes $K_B = y^{\alpha_A z_B} \cdot g^{-h_A z_B} \pmod{p}$
- $A \rightarrow B$: ζ_A, V_0
 A computes $\zeta_A = \alpha_B^{z_A}$, where $z_A \in_R \mathbb{Z}_p^*$

A computes $K_A = y^{\alpha_B z_A} \cdot g^{-h_B z_A} \pmod p$

A computes $V_0 = H'(K_A \| \zeta_B^{-\beta_A} \| 0)$

- $B \rightarrow A: V_1$
 $V_1 = H'(\zeta_A^{-\beta_B} \| K_B \| 1)$

As shown in the OSBE scheme in Section 3.1.4, $K_A = \zeta_A^{-\beta_B}$ and $K_B = \zeta_B^{-\beta_A}$, so that A verifies the V_1 and accepts only if $V_1 = H'(K_A \| \zeta_B^{-\beta_A} \| 1)$. B verifies the V_0 and accepts only if $V_0 = H'(\zeta_A^{-\beta_B} \| K_B \| 0)$.

If both verification are successful, then A and B finish all the steps of the SH, and the handshake succeeds. From this SH scheme, we can observe that four moves are required to complete the new SH scheme.

4.3 Efficient Construction of ElGamal-based SH

In this section, we present our three-move SH scheme based on ElGamal signature in two stage. Firstly, we present an ElGamal-based key agreement scheme, and then, we construct a three-move SH scheme based on it.

4.3.1 ElGamal-based Key Agreement Scheme

The first step is key generation. Pick a large prime p such that $p - 1$ has a large prime divisor q , and g which is an element of order q in \mathbb{Z}_p^* . Then, choose a random number $s \in \mathbb{Z}_q^*$, and compute $y \equiv g^s \pmod p$. The public key is $PK = \langle p, q, g, y \rangle$ and the private key is s .

Assume that there are two parties A and B whose unique identifications are ID_A and ID_B . Their identifications are signed with the third party's ElGamal signature. Consequently, A obtains the signature $\langle \alpha_A, \beta_A \rangle$, where

$$\begin{aligned} \alpha_A &= g^{r_A} \pmod p, \text{ where } r_A \in_R \mathbb{Z}_q^* \\ \beta_A &= (h_A - \alpha_A \cdot s) \cdot r_A^{-1} \pmod q, \text{ where } h_A = H(ID_A) \end{aligned}$$

B obtains the signature $\langle \alpha_B, \beta_B \rangle$. The key agreement is carried out as follows:

- B chooses $k_B \in \mathbb{Z}_q$ at random, and computes $\zeta_B = \alpha_B^{(k_B+1)} \pmod{p \cdot q}$, $\eta_B = \beta_B \cdot (k_B + 1)^{-1} \cdot \alpha_B^{k_B} \pmod q$. Then, B sends $\langle \zeta_B, \eta_B \rangle$ to A .

- Upon receiving $\langle \zeta_B, \eta_B \rangle$ from B , A chooses $k_A \in \mathbb{Z}_q$ at random and computes the shared key $K = ((y^{(\zeta_B \bmod q)} \cdot (\zeta_B \bmod p)^{\eta_B})^{h_B^{-1}})^{\alpha_A^{k_A}} \pmod{p}$, where $h_B = H(\text{ID}_B)$. A computes $\zeta_A = \alpha_A^{(k_A+1)} \pmod{p \cdot q}$, $\eta_A = \beta_A \cdot (k_A + 1)^{-1} \cdot \alpha_A^{k_A} \pmod{q}$. Then A sends the pair $\langle \zeta_A, \eta_A \rangle$ to B .
- Upon receiving the pair $\langle \zeta_A, \eta_A \rangle$ from A , B computes the shared key $K = ((y^{(\zeta_A \bmod q)} \cdot (\zeta_A \bmod p)^{\eta_A})^{h_A^{-1}})^{\alpha_B^{k_B}} \pmod{p}$, where $h_A = H(\text{ID}_A)$.

Note that the value $(\langle \alpha_A, \beta_A \rangle, k_A)$ is unknown to B and $(\langle \alpha_B, \beta_B \rangle, k_B)$ is unknown to A . To check the correctness of the scheme, we show that the shared key that A and B will compute are equal.

$$\begin{aligned}
A: \quad K &= ((y^{(\zeta_B \bmod q)} \cdot (\zeta_B \bmod p)^{\eta_B})^{h_B^{-1}})^{\alpha_A^{k_A}} \\
&= ((y^{(\alpha_B^{(k_B+1)} \bmod q)} \cdot (\alpha_B^{(k_B+1)} \bmod p)^{\beta_B \cdot (k_B+1)^{-1} \cdot \alpha_B^{k_B}})^{h_B^{-1}})^{\alpha_A^{k_A}} \\
&= ((g^{(\alpha_B^{(k_B+1)} \cdot s)} \cdot (g^{(h_B - \alpha_B \cdot s) \cdot \alpha_B^{k_B}}))^{h_B^{-1}})^{\alpha_A^{k_A}} \\
&= g^{\alpha_A^{k_A} \cdot \alpha_B^{k_B}} \pmod{p} \\
B: \quad K &= ((y^{(\zeta_A \bmod q)} \cdot (\zeta_A \bmod p)^{\eta_A})^{h_A^{-1}})^{\alpha_B^{k_B}} \\
&= ((y^{(\alpha_A^{(k_A+1)} \bmod q)} \cdot (\alpha_A^{(k_A+1)} \bmod p)^{\beta_A \cdot (k_A+1)^{-1} \cdot \alpha_A^{k_A}})^{h_A^{-1}})^{\alpha_B^{k_B}} \\
&= ((g^{(\alpha_A^{(k_A+1)} \cdot s)} \cdot (g^{(h_A - \alpha_A \cdot s) \cdot \alpha_A^{k_A}}))^{h_A^{-1}})^{\alpha_B^{k_B}} \\
&= g^{\alpha_A^{k_A} \cdot \alpha_B^{k_B}} \pmod{p}
\end{aligned}$$

4.3.2 ElGamal based Secret-handshake Scheme

As shown in the above scheme, when A and B finish running the key agreement protocol, they will share a common key K . Now we add another step, in which each party constructs a new message V by combining the common key and the identities of both parties and sends it to each other. V is computed as follows:

$$V = H(K \| \text{ID}_A \| \text{ID}_B)$$

Then A and B output “accept” only if the message V from the other party is identical to V of its own. By simply adding this step for authentication after the key agreement protocol described above, we can obtain an SH scheme. We call this scheme ElGamal-Based Handshake (EBH). We adapt the definition of an SH scheme of Balfanz *et al.* [5] to our SH scheme, which might potentially restrict the notion of a secret handshake scheme, but both the SH scheme of Balfanz *et al.* [5] and our SH scheme fall into this category. Our SH scheme is constructed as a triple of probabilistic algorithms

CreateGroup, AddUser, Handshake describe as follows:

EBH.CreateGroup. The administrator CA runs the ElGamal key generation algorithm to create the set of all keys $\{(p, q, g, y, s) \mid y \equiv g^s \pmod{p}\}$, in which s is the group secret.

EBH.AddUser. Select two collision-resistant cryptographic hash functions $H_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_q^*$, and $H_2 : \{0, 1\}^* \rightarrow \{0, 1\}^n$ for some n . To add a user U to the group, the administrator CA first allocates a unique identity ID_U to user, and generates a random nonce $r_U \in \mathbb{Z}_q$. The CA then computes the hash value $h_U = H_1(ID_U)$, and gives the user U the corresponding signature $\langle \alpha_U, \beta_U \rangle$, where $\alpha_U = g^{r_U} \pmod{p}$, $\beta_U = ((h_U - \alpha_U \cdot s) \cdot r_U^{-1}) \pmod{q}$.

EBH.Handshake. Let A and B be two users who would like to conduct the secret handshake. The three-move handshake protocol is given as follows.

- $B \rightarrow A$: ID_B, ζ_B, η_B
 $\zeta_B = \alpha_B^{(k_B+1)} \pmod{p \cdot q}$
 $\eta_B = \beta_B \cdot (k_B + 1)^{-1} \cdot \alpha_B^{k_B} \pmod{q}$
- $A \rightarrow B$: $ID_A, V_0, \zeta_A, \eta_A$
 $V_0 = H_2(((y^{(\zeta_B \bmod q)} \cdot (\zeta_B \bmod p)^{\eta_B})^{h_B^{-1}})^{\alpha_A^{k_A}} \bmod p \| ID_A \| ID_B \| 0)$
 $\zeta_A = \alpha_A^{(k_A+1)} \pmod{p \cdot q}$
 $\eta_A = \beta_A \cdot (k_A + 1)^{-1} \cdot \alpha_A^{k_A} \pmod{q}$
- $B \rightarrow A$: V_1
 $V_1 = H_2(((y^{(\zeta_A \bmod q)} \cdot (\zeta_A \bmod p)^{\eta_A})^{h_A^{-1}})^{\alpha_B^{k_B}} \bmod p \| ID_A \| ID_B \| 1)$

A verifies the V_1 and accepts only if the following equation holds

$$V_1 \stackrel{?}{=} H_2(((y^{(\zeta_B \bmod q)} \cdot (\zeta_B \bmod p)^{\eta_B})^{h_B^{-1}})^{\alpha_A^{k_A}} \bmod p \| ID_A \| ID_B \| 1)$$

B verifies the V_0 and accepts only if the following equation holds

$$V_0 \stackrel{?}{=} H_2(((y^{(\zeta_A \bmod q)} \cdot (\zeta_A \bmod p)^{\eta_A})^{h_A^{-1}})^{\alpha_B^{k_B}} \bmod p \| ID_A \| ID_B \| 0)$$

If both verification succeed, then A and B finish all the steps of the SH, and the handshake has been successful.

4.3.3 Security Proof

An adversary \mathcal{A} who can forge a valid signature can surely attack the SH protocol just as an honest member. Obviously, the probability to attack the SH scheme cannot be smaller than the probability to forge a valid signature.

Theorem 1 *The above ElGamal-based SH scheme is Impersonator Resistant under the assumption that ElGamal signature is existentially unforgeable in the random oracle model.*

Proof. ElGamal-based SH is impersonator resistant if no polynomially bounded adversary wins the following game against the Challenger with non-negligible probability: The Challenger randomly picks a public key (g, p, q, y) , and gives it to the adversary. The adversary responds with an ID_A . The Challenger then picks a random pair $\langle \zeta_A, \eta_A \rangle$, where $\zeta_A \in \mathbb{Z}_{p-q}$ and $\eta_A \in \mathbb{Z}_q$. The adversary then outputs $k'_A \in \mathbb{Z}_q$, and the adversary wins the game if $(g^{h_A})^{k'_A} = y^{\zeta_A} \cdot \zeta_A^{\eta_A} \pmod{p}$.

Given an attacker \mathcal{A} that wins the above game with probability ε . We construct another attacker \mathcal{B} that can successfully forge the ElGamal signature with probability ε . \mathcal{B} does the following:

1. \mathcal{B} , when given (g, p, q, y) , passes (g, p, q, y) to \mathcal{A} and gets ID_A back.
2. \mathcal{B} then computes $h_A = H(\text{ID}_A)$, picks a random pair $\langle \zeta_A, \eta_A \rangle$, and sends to \mathcal{A} . Then \mathcal{B} gets k'_A from \mathcal{A} .
3. Note that $y^{\zeta_A} \cdot \zeta_A^{\eta_A} = (g^{h_A})^{k'_A} \pmod{p}$. If \mathcal{B} uses g^{h_A} as the generator, $\langle \zeta_A, \eta_A \rangle$ can be viewed as the ElGamal signatures of k'_A in (g^{h_A}, p, q, y) .

Then \mathcal{B} succeeds in forging the signature if and only if \mathcal{A} wins the above game.

Hence, we can see that if the adversary \mathcal{A} can impersonate a user with valid credential, a polynomial time algorithm can be constructed to forge the ElGamal signature. There is a assumption that ElGamal signature is existentially unforgeable. So we can see that if this assumption holds, the probability ε that \mathcal{A} can impersonate a valid user in the protocol should be a negligible value.

Theorem 2 *The above ElGamal-based SH scheme is Detector Resistant under the Computational Diffie-Hellman (CDH) assumption in the Random Oracle Model.*

Proof. Firstly, let us review the CDH assumption: given a cyclic group G , a generator $g \in G$, and group elements g^a, g^b , the probability to compute g^{ab} is negligible. Then we

consider the proof as follows. ElGamal-based SH is detector resistant if no polynomially bounded adversary wins the following game against the Challenger with non-negligible probability: The group administrator holds a key set for ElGamal (g, p, q, y, s) , and the Challenger gets the (g, p, q) , and gives it to the adversary. The Challenger first asks the member for a triple $\langle \text{ID}_A, \zeta_A, \eta_A \rangle$, where $\zeta_A = \alpha_A^{(k_A+1)} \pmod{p \cdot q}$ and $\eta_A = \beta_A \cdot (k_A + 1)^{-1} \cdot \alpha_A^{k_A} \pmod{q}$. $\langle \alpha_A, \beta_A \rangle$ is the ElGamal signature on ID_A . The adversary then outputs $y' \in \mathbb{Z}_p$, and the adversary wins the game if $y' = y$.

Given an attacker \mathcal{A} that wins the above game with probability ε . We construct another algorithm \mathcal{B} that can successfully break the CDH assumption with probability ε . Algorithm \mathcal{B} is as follows:

1. Given (g, p, q) , \mathcal{B} passes (g, p, q) to \mathcal{A} .
2. Given $\langle \zeta_A, \eta_A \rangle$, \mathcal{B} can compute $g^{\alpha_A^{-(k_A+1)}} = g^{\zeta_A^{-1}}$ and $g^{\alpha_A^{k_A}} = (y^{\zeta_A} \cdot \zeta_A^{\eta_A})^{h_A^{-1}}$.
Let a be $\alpha_A^{-(k_A+1)} \pmod{q}$ and b be $\alpha_A^{k_A} \pmod{q}$ as defined in the CDH problem.
3. \mathcal{B} sends the pair $\langle \zeta_A, \eta_A \rangle$ to \mathcal{A} . Subsequently, \mathcal{B} obtains y from \mathcal{A} .
4. \mathcal{B} can compute $g^{\alpha_A^{-1}} = (\zeta_A^{\eta_A \cdot \zeta_A^{-1}} \cdot y)^{h_A^{-1}}$.

Then, \mathcal{B} has successfully broken the CDH assumption with probability ε by computing $g^{\alpha_A^{-1}} = g^{ab} = g^{\alpha_A^{-(k_A+1)} \cdot \alpha_A^{k_A}} \pmod{p}$. Thus we can see that if this CDH assumption holds, the probability ε that \mathcal{A} can violate the detector resistant property should be a negligible value.

4.4 Efficient Construction of DSA-based SH

In this section, we construct an SH Scheme based on DSA signature using a similar idea as above. DSA-based [75] scheme is a bit complex since there are two modulus used in the scheme. The Digital Signature Algorithm (DSA) was developed by NIST as a more efficient alternative to ElGamal.

From the description in Section 2.3.3, we notice that the DSA signature is very similar to the ElGamal, but they have different modulus. Therefore, we cannot construct a DSA-based handshake (DBH) scheme in a straightforward manner. The idea is to convert the DSA signature into the form of one modulus first, and then apply the ElGamal based SH scheme to the DSA signature based handshake.

Firstly, we compute γ as follows: $\gamma = (g^h \cdot y^\alpha)^{\beta^{-1}} \pmod{p}$. Now we can use the $\langle \gamma, \beta \rangle$ as the certificates of the member to conduct the handshake. We describe the scheme as follows:

DBH.CreateGroup. The administrator CA runs the DSA key generation algorithm to create the set of all keys $\{(p, q, g, y, s) \mid y \equiv g^s \pmod{p}\}$, in which s is the group secret.

DBH.AddUser. Select two collision-resistant cryptographic hash functions $H_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_q^*$, and $H_2 : \{0, 1\}^* \rightarrow \{0, 1\}^n$ for some n . To add a user U to the group, the administrator CA first allocates a unique identity ID_U to user, and generates a random nonce $r_U \in \mathbb{Z}_q$. The CA then computes the hash value $h_U = H_1(\text{ID}_U)$, and gives the user U the corresponding signature $\langle \alpha_U, \beta_U \rangle$, where $\alpha_U = (g^{r_U} \bmod p) \bmod q$, and $\beta_U = ((h_U + \alpha_U \cdot s) \cdot r_U^{-1}) \pmod{q}$.

DBH.Handshake. Let A and B be two users who would like to conduct the secret handshake. The three-move DSA-based handshake protocol runs as follows:

- $B \rightarrow A$: $\text{ID}_B, \zeta_B, \eta_B$
 $\gamma_B = (g^{h_B} \cdot y^{\alpha_B})^{\beta_B^{-1}} \bmod p$
 $\zeta_B = \gamma_B^{(k_B+1)} \pmod{p \cdot q}$
 $\eta_B = \beta_B \cdot (k_B + 1)^{-1} \cdot \gamma_B^{k_B} \pmod{q}$
- $A \rightarrow B$: $\text{ID}_A, V_0, \zeta_A, \eta_A$
 $V_0 = H_2(((y^{(-\zeta_B \bmod q)} \cdot (\zeta_B \bmod p)^{\eta_B})^{h_B^{-1}})^{\gamma_A^{k_A}} \bmod p \parallel \text{ID}_A \parallel \text{ID}_B \parallel 0)$
 $\gamma_A = (g^{h_A} \cdot y^{\alpha_A})^{\beta_A^{-1}} \bmod p$
 $\zeta_A = \gamma_A^{(k_A+1)} \pmod{p \cdot q}$
 $\eta_A = \beta_A \cdot (k_A + 1)^{-1} \cdot \gamma_A^{k_A} \pmod{q}$
- $B \rightarrow A$: V_1
 $V_1 = H_2(((y^{(-\zeta_A \bmod q)} \cdot (\zeta_A \bmod p)^{\eta_A})^{h_A^{-1}})^{\gamma_B^{k_B}} \bmod p \parallel \text{ID}_A \parallel \text{ID}_B \parallel 1)$

A verifies the V_1 and accepts only if the following equation holds

$$V_1 \stackrel{?}{=} H_2(((y^{(-\zeta_B \bmod q)} \cdot (\zeta_B \bmod p)^{\eta_B})^{h_B^{-1}})^{\gamma_A^{k_A}} \bmod p \parallel \text{ID}_A \parallel \text{ID}_B \parallel 1)$$

B verifies the V_0 and accepts only if the following equation holds

$$V_0 \stackrel{?}{=} H_2(((y^{(-\zeta_A \bmod q)} \cdot (\zeta_A \bmod p)^{\eta_A})^{h_A^{-1}})^{\gamma_B^{k_B}} \bmod p \parallel \text{ID}_A \parallel \text{ID}_B \parallel 0)$$

If both verification succeed, then A and B finish all the steps of the SH, and the handshake has been successful.

4.4.1 Security Proof

Theorem 3 *The above DSA-based SH scheme is impersonator resistant under the assumption that DSA signature is existentially unforgeable in the Random Oracle Model.*

Proof. Similar to the proof in ElGamal-based SH, DSA-based SH is impersonator resistant if no polynomially bounded adversary wins the following game against the Challenger with non-negligible probability: The Challenger randomly picks a public key (g, p, q, y) , and gives it to the adversary. The adversary responds with an ID_A . The Challenger then picks a random pair $\langle \zeta_A, \eta_A \rangle$, where $\zeta_A \in \mathbb{Z}_{p \cdot q}$ and $\eta_A \in \mathbb{Z}_q$. The adversary then outputs $k'_A \in \mathbb{Z}_q$, and the adversary wins the game if $(g^{h_A})^{k'_A} = y^{-\zeta_A} \cdot \zeta_A^{\eta_A} \pmod{p}$.

Given an attacker \mathcal{A} that wins the above game with probability ε . We construct another attacker \mathcal{B} that can successfully forge the DSA signature with probability ε . \mathcal{B} does the following:

1. \mathcal{B} , when given (g, p, q, y) , passes (g, p, q, y) to \mathcal{A} and gets ID_A back.
2. \mathcal{B} then computes $h_A = H(ID_A)$, picks a random pair $\langle \zeta_A, \eta_A \rangle$, and sends to \mathcal{A} . Then \mathcal{B} gets k'_A from \mathcal{A} .
3. Note that $y^{-\zeta_A} \cdot \zeta_A^{\eta_A} = (g^{h_A})^{k'_A} \pmod{p}$. If \mathcal{B} uses g^{h_A} as the generator, $\langle \zeta_A \pmod{q}, \eta_A \rangle$ can be used as the DSA signatures on k'_A in (g^{h_A}, p, q, y) .

Then \mathcal{B} succeeds in forging the signature if and only if \mathcal{A} wins the above game.

Hence, we can see that if the adversary \mathcal{A} can impersonate a user with valid credential, a polynomial time algorithm can be constructed to forge the DSA signature. Consequently, The above DSA-based SH scheme is impersonator resistant under the assumption that DSA signature is existentially unforgeable.

Theorem 4 *The above DSA-based SH scheme is detector resistant under the Computational Diffie-Hellman (CDH) assumption in the Random Oracle Model.*

Proof. DSA-based SH is detector resistant if no polynomially bounded adversary wins the following game against the Challenger with non-negligible probability: The group administrator holds a key set for DSA (g, p, q, y, s) , and the Challenger gets the (g, p, q) , and gives it to the adversary. The Challenger first asks the member for a triple $\langle ID_A, \zeta_A, \eta_A \rangle$, where $\zeta_A = \alpha_A^{(k_A+1)} \pmod{p \cdot q}$ and $\eta_A = \beta_A \cdot (k_A + 1)^{-1} \cdot \alpha_A^{k_A} \pmod{q}$. $\langle \alpha_A, \beta_A \rangle$ is the DSA signature on ID_A . The adversary then outputs $y' \in \mathbb{Z}_p$,

and the adversary wins the game if $y' = y$.

Given an attacker \mathcal{A} that wins the above game with non-negligible probability ε . We construct another algorithm \mathcal{B} that can successfully break the CDH assumption with probability ε . Algorithm \mathcal{B} is as follows:

1. Given (g, p, q) , \mathcal{B} passes (g, p, q) to \mathcal{A} .
2. Given $\langle \zeta_A, \eta_A \rangle$, \mathcal{B} can compute $g^{\alpha_A^{-(k_A+1)}} = g^{\zeta_A^{-1}}$ and $g^{\alpha_A^{k_A}} = (y^{-\zeta_A} \cdot \zeta_A^{\eta_A})^{h_A^{-1}}$.
Let a be $\alpha_A^{-(k_A+1)} \bmod q$ and b be $\alpha_A^{k_A} \bmod q$ as defined in the CDH problem.
3. \mathcal{B} sends the pair $\langle \zeta_A, \eta_A \rangle$ to \mathcal{A} . Subsequently, \mathcal{B} obtains y from \mathcal{A} .
4. \mathcal{B} can compute $g^{\alpha_A^{-1}} = (\zeta_A^{\eta_A \cdot \zeta_A^{-1}} \cdot y^{-1})^{h_A^{-1}}$.

Hence, \mathcal{B} has successfully broken the CDH assumption by computing $g^{\alpha_A^{-1}} = g^{ab} = g^{\alpha_A^{-(k_A+1)} \cdot \alpha_A^{k_A}} \bmod p$.

4.5 Summary

In this section, we proposed two three-move secret handshake schemes based on the ElGamal signature and DSA signature. Our work answered the open problem of constructing three-move SH schemes using ElGamal signature affirmatively. We also showed that our ElGamal (DSA) based scheme is secure against impersonator and detector attacks under the assumption the existentially unforgeable of the ElGamal (DSA) signature.

Chapter 5

Group Key Agreement Schemes

Many ID-based AGKA protocols have been proposed in recent years. Nevertheless, some efficient results in [15, 32, 36, 38, 61] require two rounds to construct a session key and some of these protocols are found to be flawed [32, 36]. In addition, some single round tripartite authenticated key agreement protocols [31, 57, 101] have been proposed but these methods cannot be extended to large groups consisting of more than three parties since these methods rely on the bilinearity property of bilinear pairing. Barua *et al.* [6] attempted to extend Joux's tripartite protocol [57] to an ID-based AGKA, but the scheme requires $\lceil \log_3 n \rceil$ rounds. Very recently, a single round ID-based AGKA protocol was proposed by Shi *et al.* [89]. However, their scheme is flawed in fact, which was shown in Section 3.3. In this chapter, we describe our single round ID-based AGKA protocol, and an efficient two-round ID-based AGKA scheme [102], which is a variant of our one-round ID-based AGKA. We then prove that our ID-based AGKA protocols are secure against active adversary under the assumption of *DBDH* in the ROM. Part of this chapter has appeared in [102].

5.1 Security Arguments

In this section, we first describe the security model in which we prove the security of our group key agreement protocol. Then, we define some security notions related to this model.

5.1.1 Security Model

The model described below follows Bresson *et al.*'s [22] formal security model, which builds on prior work from the two-party setting [9, 10, 12]. We restrict to recalling some details of their cryptographic proof model used for the security proof below. A more detailed discussion of this model can be found in [22, 23].

Participants. A finite set \mathcal{U} of PPT Turing machines U_i models the users that constitute the (potential) protocol participants. In this model we allow each user $U_i \in \mathcal{U}$ to execute a protocol many times with different users. A user may execute a polynomial number of protocol instances in parallel. We denote the instance $t \in \mathbb{N}$ of principal $U_i \in \mathcal{U}$ by Π_i^t .

Initialization. During this phase, which is conducted before the first execution of the key establishment protocol, the master secret key s and global parameters **Params** are generated by algorithm **Setup**. Each user $U_i \in \mathcal{U}$ gets public and private keys from a group administrator **GA** by using algorithm **Setup**, while the long-term private key S_{U_i} is only revealed to U_i , the corresponding public key is given to all users.

Adversarial model. Normally, the security of a protocol is related to the adversary's ability. The abilities are formally modeled by queries issued by adversaries. We assume that a probabilistic polynomial time adversary \mathcal{A} controls the communications completely and can make queries to any instance. The list of queries that \mathcal{A} can make is summarised below:

- **Execute**($\{U_1, U_2, \dots, U_r\}$): This query executes a protocol run between the users $\{U_1, U_2, \dots, U_r\}$, and the adversary \mathcal{A} gets the complete transcripts of all the messages sent during the protocol execution.

- **Send**(Π_i^t, M): This query allows the adversary \mathcal{A} to send a message M to instance Π_i^t , and \mathcal{A} gets back the reply generated by this instance.
- **Reveal**(Π_i^t): This query returns the session key. \mathcal{A} is allowed to use this query only if the oracle Π_i^t has accepted, then \mathcal{A} gets the session key.
- **Corrupt**(U_i): This query allows the adversary \mathcal{A} to obtain the long-term private key corresponding to U_i . But the adversary \mathcal{A} does not get the internal data of any instance of U_i executing the protocol.
- **Test**(U_i, t): The adversary \mathcal{A} can use this query only once. This query models the semantic security of a session key. \mathcal{A} can ask any of the above queries, and once, asks a Test query. Then, a random bit b is drawn and the session key is returned if $b = 1$, otherwise a random value is returned.

In the model, we consider two types of adversaries according to their attack types. The attack types are simulated by the queries issued by adversaries. A *passive adversary* is not allowed to use **Send** and **Corrupt** queries, while an *active adversary* can issue all the above queries.

5.1.2 Security Notions

We define session IDS (SIDS) for oracle Π_i^t in a execution protocol as $SIDS(\Pi_i^t) = \{SID_{ij}, j \in \mathcal{U}\}$ where SID_{ij} is the concatenation of all messages exchanged by oracle Π_i^t with Π_j^w . The partner ID for an oracle Π_i^t , denoted by $PIDS(\Pi_i^t)$, is a set of the users with whom Π_i^t intends to establish a session key.

Definition 7 Partnering Now we define instances Π_i^t and Π_j^w are partnered if and only if $PIDS(\Pi_i^t) = PIDS(\Pi_j^w)$ and $SIDS(\Pi_i^t) = SIDS(\Pi_j^w)$.

Definition 8 Freshness Following [22, 60, 61], we define a user instance Π_i^t that has accepted fresh if:

- For a $U_j \in \mathcal{U}$, a **Corrupt**(U_j) query was never executed before a query of the form **Send**($\Pi_i^t, *$) or **Send**($\Pi_j^w, *$), where Π_i^t and Π_j^w are partnered, has taken place.
- Π_i^t has accepted a session key $K \neq \text{NULL}$ and neither Π_i^t nor one of its partners has been asked for a **Reveal** query.

Before we look into the security of the protocol, we first define the following game between the adversary \mathcal{A} and a set of oracles Π_i^t for $U_i \in \mathcal{U}$.

1. Each user is given a long-term private key during the initialization phase.
2. Adversary \mathcal{A} interacts with some queries and gets back the reply generated by the corresponding oracles.
3. \mathcal{A} executes a **Test**(Π_i^t) query for a fresh oracle Π_i^t , and finally outputs a guess bit b' .

In above game, we denote by **Succ** the probability that the bit b' outputs by \mathcal{A} satisfies $b = b'$.

Definition 9 Protocol Security We denote the advantage of the adversary \mathcal{A} attacking the protocol as $\text{Adv}_{\mathcal{A}}(k) = |2 \cdot \text{Succ} - 1|$. We say the group key establishment protocol secure if for all PPT adversary \mathcal{A} $\text{Adv}_{\mathcal{A}}(k)$ is negligible.

Definition 10 Authentication A key agreement protocol is said to provide authentication if for a user U_i , no other users except partners can learn the value of a session key.

Definition 11 Forward Secrecy Forward secrecy means that an adversary gets negligible advantage in knowing information about previously established session keys when making a **Corrupt** query.

5.2 One-Round Group Key Agreement Scheme

In this section, we present our one-round ID-based AGKA scheme, which we called O-AGKA. The protocol involves a group administrator **GA**. In the following description $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1$, $H_2 : \mathbb{G}_2 \rightarrow \{0, 1\}^n$ and $H_3 : \{0, 1\}^n \rightarrow \{0, 1\}^n$ are cryptographic hash functions. H_1 , H_2 and H_3 are considered as random oracles in the security analysis.

5.2.1 The Scheme

Our O-AGKA scheme consists of four probabilistic algorithms **Setup**, **Extract**, **Interacting**, and **Key Computation**.

Setup. GA runs BDH parameter generator to generate a prime q , two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q , and an bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Choose a random generator $P \in \mathbb{G}_1$. Then GA picks a random $s \in \mathbb{Z}_q^*$ and sets $P_{pub} = sP$. GA keeps s secret as the *master secret key* and publishes system parameters $\text{params} = \{\hat{e}, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, H_1, H_2, H_3\}$.

Extract. When a user U_i with identity ID_i wishes to obtain a key pair, GA computes $Q_i = H_1(ID_i)$ and the long-term private key $S_i = sQ_i$, and returns S_i to the user U_i .

Let U_1, \dots, U_n be the n users who want to establish a session key. The protocol is as follows:

Interacting. Each user U_i picks $\delta_i \xleftarrow{R} \mathbb{G}_2$ and $r_i, k_i \xleftarrow{R} \{0, 1\}^n$. Then U_i computes $P_i^j = H_2(\hat{e}(S_i, Q_j) \cdot \delta_i) \oplus r_i$, where $1 \leq j \leq n$ and $j \neq i$. U_i then computes

$$X_i = \langle \delta_i, P_i^1, \dots, P_i^{i-1}, P_i^{i+1}, \dots, P_i^n, H_3(r_i) \oplus k_i, \mathcal{L} \rangle,$$

where \mathcal{L} is a label that contains information about how “ P_i^j ” is associated with each receiver. Then U_i broadcasts X_i to all others.

Key Computation. Let $X_j = \langle R_j, P_j^1, \dots, P_j^n, V_j, \mathcal{L} \rangle$. Upon receiving X_j , each responder U_i , using \mathcal{L} , finds appropriate P_j^i and computes

$$k'_j = H_3(H_2(\hat{e}(Q_j, S_i) \cdot R_j) \oplus P_j^i) \oplus V_j$$

Each U_i can now compute the common session key as follows:

$$K = K_i = k'_1 \oplus \dots \oplus k'_{i-1} \oplus k_i \oplus k'_{i+1} \oplus \dots \oplus k'_n$$

From the scheme we described above, we can observe that each participant involved broadcasts message only once, as shown in Fig. 5.1, so that the protocol can be completed in one round.

5.2.2 Security Proof

In this section, we show that the protocol O-AGKA is secure against an active adversary under the *DBDH* assumption. In other words, if there exists an active adversary who has non-negligible probability of breaking the *DBDH* assumption, then (s)he also has non-negligible probability of breaking protocol O-AGKA.

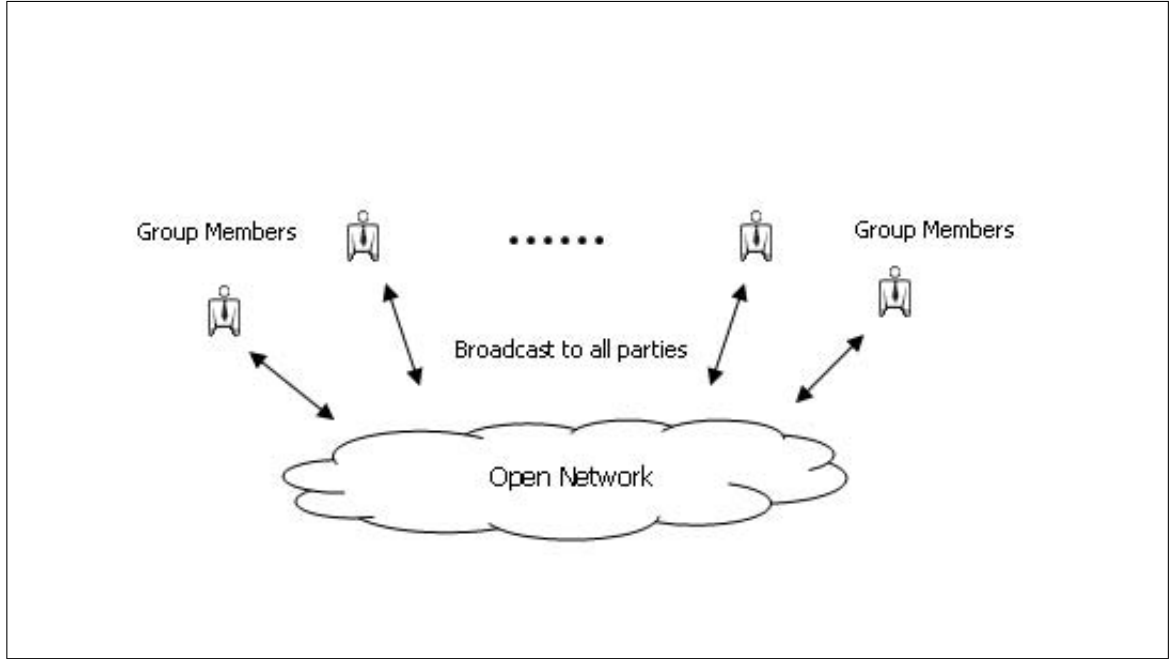


Figure 5.1: One-Round ID-based Group Key Agreement

Theorem 5 *The above O-AGKA protocol is secure against an active adversary under the DBDH assumption in the Random Oracle Model. Concretely,*

$$\text{Adv}_{\mathcal{A}} \leq 2n \cdot q_{\text{ex}} \cdot \text{Adv}_{\mathcal{A}}^{\text{DBDH}}$$

Proof. Let \mathcal{A} be an active adversary that can get an advantage in breaking O-AGKA. We first consider the case that an adversary \mathcal{A} makes only one **Execute** query and then extend this to the case that \mathcal{A} makes multiple **Execute** queries. Let n be the number of users chosen by the adversary \mathcal{A} . The distribution of the transcript \mathcal{T} and the resulting group session key K is given by:

$$\text{params} = \left[\begin{array}{l} (\mathbb{G}_1, \mathbb{G}_2, \hat{e}) \leftarrow \mathcal{IG}_{BDH}; P \leftarrow \mathbb{G}_1; s \leftarrow \mathbb{Z}_q^*; P_{\text{pub}} = sP \\ Q_1, \dots, Q_n \leftarrow \mathbb{G}_1; S_1 = sQ_1, \dots, S_n = sQ_n : \\ (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{\text{pub}}) \end{array} \right]$$

$$Real = \left[\begin{array}{l} \delta_1, \dots, \delta_n \leftarrow \mathbb{G}_2; r_1, \dots, r_n, k_1, \dots, k_n \leftarrow \{0, 1\}^n; \\ R_1 = \delta_1, \dots, R_n = \delta_n \\ P_1^i = H_2(\hat{e}(S_1, Q_i) \cdot \delta_1) \oplus r_1, \dots, P_n^i = H_2(\hat{e}(S_n, Q_i) \cdot \delta_n) \oplus r_n \\ V_1 = H_3(r_1) \oplus k_1, \dots, V_n = H_3(r_n) \oplus k_n; \\ k'_j = H_3(H_2(\hat{e}(Q_j, S_i) \cdot R_j) \oplus P_j^i) \oplus V_j \\ \mathcal{T} = \langle R_1, \dots, R_n, P_1^i, \dots, P_n^i, V_1, \dots, V_n \rangle; \\ K = k'_1 \oplus \dots \oplus k'_{i-1} \oplus k_i \oplus k'_{i+1} \oplus \dots \oplus k'_n \end{array} \right]$$

Consider the distributions $Fake_i$ defined as follows:

$$Fake_1 = \left[\begin{array}{l} \delta_1, \dots, \delta_n \leftarrow \mathbb{G}_2; r_1, \dots, r_n, k_1, \dots, k_n \leftarrow \{0, 1\}^n; b_1, \dots, b_n \leftarrow \mathbb{Z}_q^* \\ R_1 = \delta_1, \dots, R_n = \delta_n \\ P_1^i = H_2(\hat{e}(b_1 P_{pub}, b_i P) \cdot \delta_1) \oplus r_1, \dots, P_n^i = H_2(\hat{e}(S_n, Q_i) \cdot \delta_n) \oplus r_n \\ V_1 = H_3(r_1) \oplus k_1, \dots, V_n = H_3(r_n) \oplus k_n; \\ k'_1 = H_3(H_2(\hat{e}(b_1 P, b_i P_{pub}) \cdot R_1) \oplus P_1^i) \oplus V_1 \\ k'_j = H_3(H_2(\hat{e}(Q_j, S_i) \cdot R_j) \oplus P_j^i) \oplus V_j | 2 \leq j \leq n, j \neq i \\ \mathcal{T} = \langle R_1, \dots, R_n, P_1^i, \dots, P_n^i, V_1, \dots, V_n \rangle; \\ K = k'_1 \oplus \dots \oplus k'_{i-1} \oplus k_i \oplus k'_{i+1} \oplus \dots \oplus k'_n \end{array} \right] \dots$$

Continuing in this way, we obtain the distribution:

$$Fake_n = \left[\begin{array}{l} \delta_1, \dots, \delta_n \leftarrow \mathbb{G}_2; r_1, \dots, r_n, k_1, \dots, k_n \leftarrow \{0, 1\}^n; b_1, \dots, b_n \leftarrow \mathbb{Z}_q^* \\ R_1 = \delta_1, \dots, R_n = \delta_n \\ P_1^i = H_2(\hat{e}(b_1 P_{pub}, b_i P) \cdot \delta_1) \oplus r_1, \dots, P_n^i = H_2(\hat{e}(b_n P_{pub}, b_i P) \cdot \delta_n) \oplus r_n \\ V_1 = H_3(r_1) \oplus k_1, \dots, V_n = H_3(r_n) \oplus k_n; \\ k'_j = H_3(H_2(\hat{e}(b_j P, b_i P_{pub}) \cdot R_j) \oplus P_j^i) \oplus V_j \\ \mathcal{T} = \langle R_1, \dots, R_n, P_1^i, \dots, P_n^i, V_1, \dots, V_n \rangle; \\ K = k'_1 \oplus \dots \oplus k'_{i-1} \oplus k_i \oplus k'_{i+1} \oplus \dots \oplus k'_n \end{array} \right]$$

Let $\epsilon = \text{Adv}_{\mathcal{A}}^{DBDH}$. Assume that \mathcal{A} made q_{se} times **Send** queries and q_{ex} times **Execute** queries. Then \mathcal{A} randomly chooses (\mathcal{T}, K) pairs to make a **Test** query and outputs b' . Since \mathcal{A} can obtain $b_1 P, \dots, b_n P$ by using multiple H_1 queries, and $P_{pub} = sP$ is public, it is obviously that \mathcal{A} can distinguish $\hat{e}(S_{U_1}, Q_{U_i})$ from $\hat{e}(b_1 sP, b_i P)$ with probability ϵ' , where $\epsilon' \leq \epsilon$. Hence \mathcal{A} can correctly guess $b = b'$ with probability ϵ' . The remaining steps continue in the same way and we obtain:

$$\begin{aligned}
& |Pr[\mathcal{T} \leftarrow Real; K \leftarrow Real; \mathcal{A}(\mathcal{T}, K) = 1] - \\
& \quad Pr[\mathcal{T} \leftarrow Fake_1; K \leftarrow Fake_1; \mathcal{A}(\mathcal{T}, K) = 1]| \leq \epsilon \\
& |Pr[\mathcal{T} \leftarrow Fake_1; K \leftarrow Fake_1; \mathcal{A}(\mathcal{T}, K) = 1] - \\
& \quad Pr[\mathcal{T} \leftarrow Fake_2; K \leftarrow Fake_2; \mathcal{A}(\mathcal{T}, K) = 1]| \leq \epsilon \\
& \quad \vdots \\
& |Pr[\mathcal{T} \leftarrow Fake_{n-1}; K \leftarrow Fake_{n-1}; \mathcal{A}(\mathcal{T}, K) = 1] - \\
& \quad Pr[\mathcal{T} \leftarrow Fake_n; K \leftarrow Fake_n; \mathcal{A}(\mathcal{T}, K) = 1]| \leq \epsilon
\end{aligned}$$

Combining the above equations, we obtain the following.

$$\begin{aligned}
\epsilon'' &= |Pr[\mathcal{T} \leftarrow Real; K \leftarrow Real; \mathcal{A}(\mathcal{T}, K) = 1] - \\
& \quad Pr[\mathcal{T} \leftarrow Fake_n; K \leftarrow Fake_n; \mathcal{A}(\mathcal{T}, K) = 1]| \leq n \cdot \epsilon
\end{aligned}$$

Hence ϵ'' is the probability that the session key can be correctly guessed when \mathcal{A} make the **Test** query. Assume that \mathcal{A} has made q_h times H_1 queries during the breaking process, then there will be a \mathcal{H} -list which contains all the messages that \mathcal{A} has queried before. Let **Ask** be the event that what \mathcal{A} make to the *Hash* query is on the \mathcal{H} -list when \mathcal{A} make the **Test** query. The advantage \mathcal{A} in breaking the protocol conditioned by the fact that the session key is correctly guessed, is:

$$\begin{aligned}
Adv_{\mathcal{A}} &= 2 \cdot Succ - 1 = 2Pr[b = b'] - 1 \\
&= 2Pr[b = b' | \neg Ask]Pr[\neg Ask] + 2Pr[b = b' | Ask]Pr[Ask] - 1 \\
&= 2Pr[b = b' | \neg Ask] + 2Pr[b = b' | Ask] - 1 \\
&= 2Pr[b = b' | Ask] = 2\epsilon''
\end{aligned}$$

In the random oracle model, $2Pr[b = b' | \neg Ask] - 1 = 0$, since \mathcal{A} cannot gain any advantage on a random oracle without asking for it. Then we can have the probability that \mathcal{A} breaks the O-AGKA, which is less than $2n \cdot Adv_{\mathcal{A}}^{DBDH}$. By adapting a standard hybrid argument, we obtain the probability that an active adversary \mathcal{A} breaks the protocol O-AGKA as follows:

$$Adv_{\mathcal{A}} \leq 2n \cdot q_{ex} \cdot Adv_{\mathcal{A}}^{DBDH}$$

5.3 Efficient Group Key Agreement Scheme

Then we present our two-round ID-based AGKA protocol called T-AGKA, which is more efficient in communication costs than other previously known AGKA protocols.

The T-AGKA protocol is a variant of O-AGKA protocol presented above. As shown in Fig. 5.2, an initiator will broadcast message to other users firstly before all the parties start communicating with each other. We describe the scheme as follows:

5.3.1 The Scheme

Setup. As in the O-AGKA scheme. In addition, we pick three new hash functions

$$H_4 : \mathbb{G}_2 \rightarrow \{0, 1\}^n, H_5 : \{0, 1\}^n \rightarrow Z_q^* \text{ and } H_6 : \mathbb{G}_1 \rightarrow \{0, 1\}^n.$$

Extract. As in the O-AGKA scheme.

Interacting.

- **Round 1** The initiator U_1 picks $\delta \xleftarrow{R} \mathbb{G}_2$, $r \xleftarrow{R} \{0, 1\}^n$ and $k_1 \xleftarrow{R} \mathbb{Z}_p^*$. Then U_1 computes $D_1 = \langle R, P_2, \dots, P_n, V, W, \mathcal{L} \rangle$ such that

$$D_1 = \langle \delta, r \oplus H_4(\hat{e}(S_1, Q_2) \cdot \delta), \dots, r \oplus H_4(\hat{e}(S_1, Q_n) \cdot \delta), H_5(r) \cdot k_1 P, k_1 P_{pub}, \mathcal{L} \rangle,$$

where \mathcal{L} is a label that contains information about how “ P_i ” is associated with each receiver, and broadcasts D_1 to all others.

- **Round 2** Upon receiving D_1 , each responder U_i , $2 \leq i \leq n$, using \mathcal{L} , finds appropriate P_i , and computes $r' = H_4(\hat{e}(Q_1, S_i) \cdot R) \oplus P_i$. If D_1 is the valid message, it is obvious that $r' = r$. Then U_i picks $k_i \xleftarrow{R} \mathbb{Z}_p^*$, computes

$$D_i = \langle H_5(r) \cdot k_i P, k_i P_{pub} \rangle$$

and broadcasts D_i to all others.

Key Computation. Let $D_j = \langle X_j, Y_j \rangle$. When received D_j , each U_i , including the initiator U_1 , computes $z_1 = H_5(r)^{-1} \cdot V$ and $z_j = H_5(r)^{-1} \cdot X_j$, $2 \leq j \leq n$. Then each U_i can hold a list of z_j , and U_i can verify all the z_j :

$$\hat{e}(P, \sum_{j=1}^n Y_j) \stackrel{?}{=} \hat{e}(P_{pub}, \sum_{j=1}^n z_j)$$

If the above equation holds, U_i can assume that all the parties involved are valid members of the group with the corresponding long-term private keys.

Each U_i now can compute the common session key as follows:

$$K = K_i = H_6(z_1) \oplus \dots \oplus H_6(z_n)$$

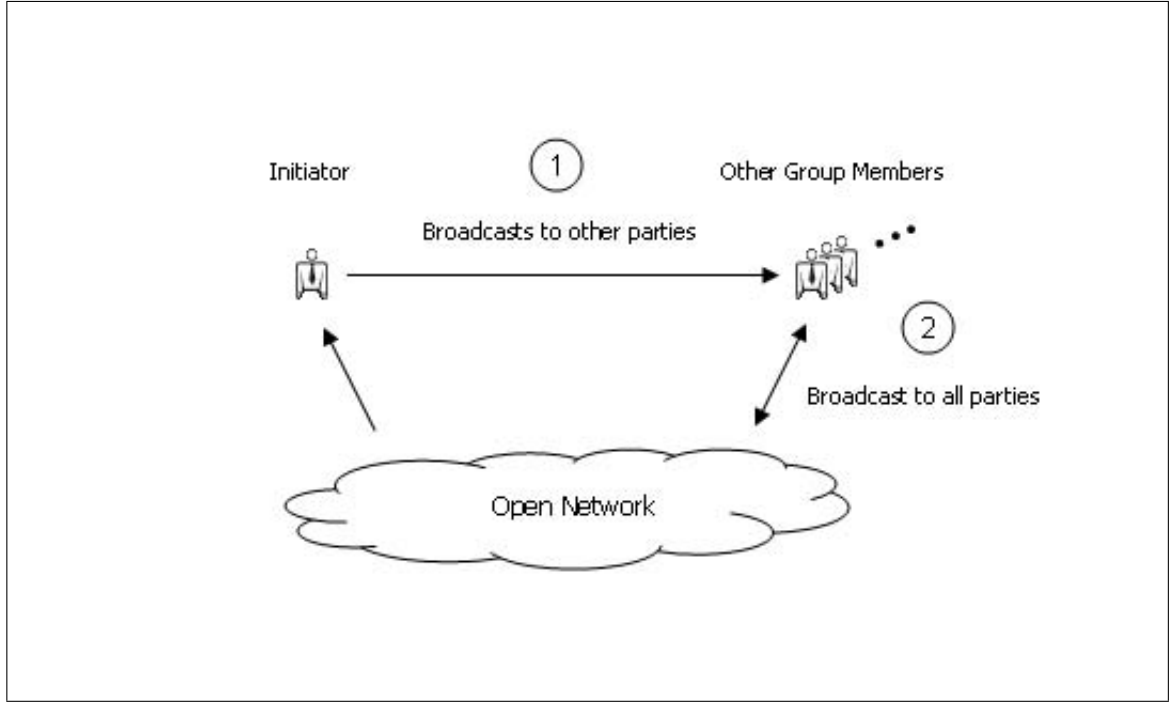


Figure 5.2: Two-Round ID-based Group Key Agreement

5.3.2 Security Proof

Theorem 6 *The two-round group key agreement protocol T-AGKA is secure against an active adversary under the DBDH assumption in the Random Oracle Model. Concretely,*

$$\text{Adv}_{\mathcal{A}} \leq 2q_{ex} \cdot \text{Adv}_{\mathcal{A}}^{\text{DBDH}}$$

Proof. Let \mathcal{A} be an active adversary that can get an advantage in breaking O-AGKA. The distribution of the transcript \mathcal{T} and the resulting group session key K is given by:

$$\text{params} = \left[\begin{array}{l} (\mathbb{G}_1, \mathbb{G}_2, \hat{e}) \leftarrow \mathcal{IG}_{BDH}; P \leftarrow \mathbb{G}_1; s \leftarrow \mathbb{Z}_q^*; P_{pub} = sP \\ Q_1, \dots, Q_n \leftarrow \mathbb{G}_1; S_1 = sQ_1, \dots, S_n = sQ_n : \\ (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{pub}) \end{array} \right]$$

$$Real = \left[\begin{array}{l} \delta \leftarrow \mathbb{G}_2; r \leftarrow \{0, 1\}^n; k_1, \dots, k_n \leftarrow \mathbb{Z}_q^*; \\ R = \delta; P_i = r \oplus H_4(\hat{e}(S_1, Q_i) \cdot \delta); V = H_5(r) \cdot k_1 P; W = k_1 P_{pub} \\ X_2 = H_5(r) \cdot k_2 P, \dots, X_n = H_5(r) \cdot k_n P \\ r' = H_4(\hat{e}(Q_1, S_i) \cdot R) \oplus P_i \\ z_1 = H_5(r')^{-1} \cdot V; z_j = H_5(r')^{-1} \cdot X_j \\ \mathcal{T} = \langle R, P_i, X_1, \dots, X_n, Y_i, V, W \rangle; \\ K = H_6(z_1) \oplus \dots \oplus H_6(z_n) \end{array} \right]$$

Consider the distributions *Fake* defined as follows:

$$Fake = \left[\begin{array}{l} \delta \leftarrow \mathbb{G}_2; r \leftarrow \{0, 1\}^n; a, b_2, \dots, b_n, k_1, \dots, k_n \leftarrow \mathbb{Z}_q^*; \\ R = \delta; P_i = r \oplus H_4(\hat{e}(aP_{pub}, b_i P) \cdot \delta); V = H_5(r) \cdot k_1 P; W = k_1 P_{pub} \\ X_2 = H_5(r) \cdot k_2 P, \dots, X_n = H_5(r) \cdot k_n P \\ r' = H_4(\hat{e}(aP, b_i P_{pub}) \cdot R) \oplus P_i \\ z_1 = H_5(r')^{-1} \cdot V; z_j = H_5(r')^{-1} \cdot X_j \\ \mathcal{T} = \langle R, P_i, X_1, \dots, X_n, Y_i, V, W \rangle; \\ K = H_6(z_1) \oplus \dots \oplus H_6(z_n) \end{array} \right]$$

Let $\epsilon = \text{Adv}_{\mathcal{A}}^{DBDH}$. Assume that \mathcal{A} made q_{se} times **Send** queries and q_{ex} times **Execute** queries. Then \mathcal{A} randomly chooses (\mathcal{T}, K) pairs to make a **Test** query and outputs b' . We observe that once \mathcal{A} can distinguish r from a random number, then he can distinguish the session key from a random number. Since \mathcal{A} can obtain aP, b_2P, \dots, b_nP by using multiple H_1 queries, and $P_{pub} = sP$ is public, it is obvious that \mathcal{A} can distinguish $\hat{e}(S_{U_1}, Q_{U_i})$ from $\hat{e}(asP, b_iP)$ with probability ϵ' , where $\epsilon' \leq \epsilon$. Hence \mathcal{A} can correctly guesses $b = b'$ with probability ϵ' . Then we can have:

$$\epsilon' = |Pr[\mathcal{T} \leftarrow Real; K \leftarrow Real; \mathcal{A}(\mathcal{T}, K) = 1] - Pr[\mathcal{T} \leftarrow Fake; K \leftarrow Fake; \mathcal{A}(\mathcal{T}, K) = 1]| \leq \epsilon$$

ϵ' is the probability that the session key can be correctly guessed when \mathcal{A} make the **Test** query. Assume that \mathcal{A} has made q_h times *Hash* queries during the breaking process, then there will be a \mathcal{H} -list which contains all the messages that \mathcal{A} has queried before. Let **Ask** be the event that what \mathcal{A} make to the H_1 query is on the \mathcal{H} -list when \mathcal{A} make the **Test** query. The advantage \mathcal{A} in breaking the protocol conditioned by the fact that the session key is correctly guessed, is

$$\begin{aligned}
\text{Adv}_{\mathcal{A}} &= 2 \cdot \text{Succ} - 1 = 2\Pr[b = b'] - 1 \\
&= 2\Pr[b = b' | \neg \text{Ask}] \Pr[\neg \text{Ask}] + 2\Pr[b = b' | \text{Ask}] \Pr[\text{Ask}] - 1 \\
&= 2\Pr[b = b' | \neg \text{Ask}] + 2\Pr[b = b' | \text{Ask}] - 1 \\
&= 2\Pr[b = b' | \text{Ask}] = 2\epsilon'
\end{aligned}$$

In the random oracle model, $2\Pr[b = b' | \neg \text{Ask}] - 1 = 0$, since \mathcal{A} cannot gain any advantage on a random oracle without asking for it. Then we can have the probability that \mathcal{A} breaks the T-AGKA, which is less than $2 \cdot \text{Adv}_{\mathcal{A}}^{DBDH}$. By adapting a standard hybrid argument, we obtain the probability that an active adversary breaks the protocol T-AGKA as follows:

$$\text{Adv}_{\mathcal{A}} \leq 2q_{ex} \cdot \text{Adv}_{\mathcal{A}}^{DBDH}$$

5.4 Efficiency Comparison

In this section, we compare our protocols O-AGKA, T-AGKA with two previously described AGKA protocols, the CHL-AGKA by Choi *et al.* [32], and the SCL-AGKA by Shi *et al.* [89]. The CHL-AGKA scheme [32] is proved to be flawed, but an improved scheme [37] fixes the problem of CHL-AGKA, which is very similar to the CHL-AGKA. So we still treat CHL-AGKA as secure. We use the following notations:

n	total number of the users in the group
Round	total number of rounds
Pairing	total number of pairing computations for all users
Ucasts	total number of unicast of all members
Bcasts	total number of broadcast of all members
Msize	total number of the messages of all members

Then, we obtain the following comparison.

<i>Protocol</i>	Round	Pairing	Ucasts	Bcasts	Msize	Remark
CHL-AGKA	2	$4n$	0	$2n$	$3n$	ID-based
SCL-AGKA	1	n	$(n-1)^2$	0	n^2	flawed
Our O-AGKA	1	n^2	0	n	$n(n+2)$	ID-based
Our T-AGKA	2	$4n$	0	n	$3n$	ID-based

Table 5.1: AGKA Scheme Efficiency Comparison

As shown in Table 5.1, our T-AGKA protocol is the most efficient one as compared to other protocols. Our O-AGKA protocol requires to involve more messages compared to other protocols, but it *only* requires one round. We noted that SCL-AGKA actually is flawed and is *not* an ID-based protocol, which was presented in Section 3.3, and hence, our O-AGKA protocol is *the only* provably secure protocol that requires one round that is known to date.

5.5 Summary

In this section, we presented a provably secure one-round ID-based AGKA protocol for the first time in the literature. The protocol is a contributory key agreement in which each group member takes responsibility for contributing to the generation of group session key. We also provided an alternative way of achieving an efficient two-round ID-based AGKA protocol. The scheme itself requires two rounds. However, as we showed, this scheme is very efficient in communication costs, compared with other previously known ID-based AGKA protocols, and hence, this scheme outperforms any other existing schemes in the literature. We provided security proofs for our schemes, and showed that they are secure against active adversaries under the DBDH assumption in the random oracle model. Our protocols provide forward secrecy in the sense that any exposure of any user's long-term private keys does not compromise the security of previous session keys.

Chapter 6

Group Secret Handshakes

Most two-party secret handshakes in literature are based on key agreement protocols. Once two parties can construct the same shared key, they complete the handshake successfully. Now we look into the group secret handshake. Recently, a framework of multi-party secret handshakes was introduced by Tsudik and Xu [91, 92]. They proposed a framework called GCD, which is essentially a compiler that transforms three main ingredients, including a group signature scheme [8, 13, 62], a centralised group key distribution scheme [97], and a distributed group key agreement scheme [9, 25], into a secure secret handshake scheme. They also constructed two instantiations following this framework. However, Tsudik and Xu [91, 92] mentioned that they only aimed to construct a framework of multi-party secret handshake, and never optimize the efficiency of the framework. Another efficient GSH scheme was proposed by Jarecki, Kim and Tsudik [56], but their scheme has a defect, which was shown in Section 3.4. Adversary involved in the protocol can know the group affiliation of other honest parties, which violates the security requirement of SH. In this chapter, we present our GSH scheme, which is motivated by the multi-receiver identity-based encryption scheme by Baek *et al.* [4]. Then we prove that our scheme does not have this defect and is secure under the BDH and DBDH assumptions.

6.1 Generic Scheme

In this section we extend the generic scheme of SH to a multi-party setting. The model consists of a set \mathcal{U} of possible users, and a set G of groups, where each group is a set of members managed by a group administrator **GA**. We define a *group secret handshake* scheme **GSH** by the following algorithms:

- **GSH.CreateGroup**: a key generation algorithm executed by the group administrator **GA** to establish a group G . It takes as input security parameters, and outputs the group public key p_G , the **GA**'s private key s_G , and a revoked user list \mathcal{RUL} , which is originally set to empty. The \mathcal{RUL} is made known only to current group members.
- **GSH.AddUser**: an algorithm executed between **GA** and a group member on **GA**'s private key s_G and shared inputs: **params**, p_G , and the identity of the group member which is bit string ID of size regulated by **params**. After performing the algorithm, the group member will be issued a secret credential produced by **GA** for the member's identity ID .
- **GSH.HandShake**: an authentication protocol, executed by a set Δ of n users purporting to be members of a group G , where $\Delta = \{U_1, \dots, U_n\}$ and $n \geq 2$. The protocol takes as public input the identities $ID_{U_1}, \dots, ID_{U_n}$ of all the users in Δ , and **params**, and the private input is their secret credentials. The output of the protocol for each party is either *reject* or *accept*.
- **GSH.TraceUser**: an algorithm executed by the **GA**, on input of a transcript of a successful handshake between the set Δ of users, outputs the identities of all the users involved in the handshake.
- **GSH.RemoveUser**: an algorithm executed by **GA** on input an identity of the user U and the \mathcal{RUL} , inserts U into the \mathcal{RUL} and sends the updated \mathcal{RUL} to the existing group members through the authenticated anonymous channel.

6.2 Security Arguments

A GSH scheme must satisfy the properties of completeness, impersonation resistant, and detection resistant. The adversary is allowed to run the protocols several times and be able to make additional queries after each attempt, before the adversary announces that (s)he is ready for the true challenges.

We denote the set of users involved in the GSH as Δ . $\{U_1, \dots, U_n\}$ each denotes a user $U \in \Delta$. Consider that \mathcal{A} may join the set Δ , and perform a GSH protocol with the valid users in G .

Completeness. If all the participants $\{U_1, \dots, U_n\}$ involved in the group secret handshakes are honest members of the same group with valid certificates from the group administrator. Then both parties output “accept”, otherwise output “reject”.

Impersonation Resistance. If an adversary $\mathcal{A} \notin G$ does not corrupt any member of its target group G , it has only a negligible probability in impersonating as an honest member of G .

Let \mathcal{A} denote an attacker, and \mathcal{B} denote a challenger. Consider the following game in which \mathcal{A} interacts with \mathcal{B} :

Phase 1: \mathcal{A} outputs target multiple identities $\Delta = (\text{ID}_1^*, \dots, \text{ID}_n^*)$, where $\text{ID}_{\mathcal{A}} = \text{ID}_i^*$ and $i \in [1, n]$.

Phase 2: \mathcal{B} runs a key generation algorithm to generate the group public key p_G , the GA’s private key s_G , and sends p_G to \mathcal{A} while keeping s_G secret from \mathcal{A} .

Phase 3: \mathcal{A} makes a number of credential extraction queries. To answer each query made by \mathcal{A} , \mathcal{B} runs the `GSH.AddUser` algorithm, and outputs S_{ID} which is the group credential for identity ID , where $\text{ID} \notin \Delta$.

Phase 4: \mathcal{A} triggers a handshake protocol. \mathcal{B} acts as honest parties in the protocol, who have valid credentials for their identities.

Phase 5: \mathcal{B} answers \mathcal{A} ’s credential extraction / random oracle queries as in Phase 2 and 3.

Phase 6: \mathcal{A} returns the messages $\langle X_{\mathcal{A}}, Y_{\mathcal{A}} \rangle$, which can make other parties output “accept” after running the protocol.

We define the probability that attacker \mathcal{A} impersonates successfully in the handshake protocol as $\text{Adv}_{\mathcal{A}}^{\text{Impersonate}}$, which is identical to the probability that \mathcal{A} outputs the valid pair $\langle X_{\mathcal{A}}, Y_{\mathcal{A}} \rangle$.

Detection Resistance. If an adversary $\mathcal{A} \notin G$, who does not corrupt any member of its target group G , is involved in the group secret handshakes, each participant in handshakes has only a negligible probability in distinguishing an interaction with any honest user $U_i \in G$ from the one with a simulator.

Let \mathcal{A} denote an attacker, and \mathcal{B} denote a challenger. Phases 2, 3, 5 of the attack game are identical to those for impersonation resistant. We only describe Phase 1, 4, 6 in the following:

Phase 1: \mathcal{A} outputs two target multiple identities $\Delta = (\text{ID}_1^*, \dots, \text{ID}_{i-1}^*, \text{ID}_{i+1}^*, \text{ID}_n^*)$ and $\text{ID}_i^0, \text{ID}_i^1$, where $\text{ID}_{\mathcal{A}} \neq \text{ID}_i^*, i \in [1, n]$. ID_i^0 is a member, who holds an valid credential. ID_i^1 is a unqualified member, who does not have the valid group credential.

Phase 4: \mathcal{A} triggers a handshake protocol. \mathcal{B} randomly chooses a $\beta \in \{0, 1\}$, and set $\text{ID}_i^* = \text{ID}_i^\beta$. Then \mathcal{B} simulates the handshake protocol with \mathcal{A} .

Phase 6: \mathcal{A} outputs its guess β' .

We denote the probability that the attacker \mathcal{A} can distinguish an interaction with any honest user $U_i \in G$ from the one with a simulator in the handshake protocol as $\text{Adv}_{\mathcal{A}}^{\text{Detect}} = |\Pr[\beta' = \beta] - \frac{1}{2}|$. In other words, an attacker involved in the handshake protocol has a probability of $\text{Adv}_{\mathcal{A}}^{\text{Detect}}$ to know the group affiliation of other participants.

Indistinguishability to eavesdropper. An adversary \mathcal{A} , who does not participate in a handshake protocol, has only a negligible probability in learning any knowledge about whether the handshake is successful or not, even if $\mathcal{A} \in G$.

Let \mathcal{A} denote an attacker, and \mathcal{B} denote a challenger. Phases 2, 3, 5 of the attack game are identical to those for impersonation resistant. We only describe Phase 1, 4, 6 in the following:

Phase 1: \mathcal{A} outputs two target multiple identities $\Delta_0 = (\text{ID}_1^*, \dots, \text{ID}_n^*)$ and $\Delta_1 = (\text{ID}_1, \dots, \text{ID}_n)$, where $\text{ID}_{\mathcal{A}} \neq \text{ID}_i^*$ and $\text{ID}_{\mathcal{A}} \neq \text{ID}_i, i \in [1, n]$. Δ_0 is a group, in which each user holds an valid credential. Δ_1 is a group, in which one or some users do not have the valid group credential.

Phase 4: \mathcal{B} randomly chooses a $\beta \in \{0, 1\}$, and simulates a handshake protocol among the group Δ_β . An additional copy will be sent to \mathcal{A} every time \mathcal{B} simulates an interaction in GSH protocol.

Phase 6: \mathcal{A} outputs its guess $\beta' \in \{0, 1\}$.

We denote the probability that the attacker \mathcal{A} can distinguish an successful protocol from an unsuccessful one as $\text{Adv}_{\mathcal{A}}^{\text{Distinguish}} = |\Pr[\beta' = \beta] - \frac{1}{2}|$. If an attacker does not take part in a handshake protocol, (s)he has a probability of $\text{Adv}_{\mathcal{A}}^{\text{Distinguish}}$ to know whether the handshake protocol is successful or not.

Unlinkability. No adversary \mathcal{A} is able to associate two handshakes involving a same honest user or a same set of honest users, even if $\mathcal{A} \in G$ and \mathcal{A} participated in both executions.

Traceability. Given a transcript of a successful handshake execution, group administrator GA can trace all users involved.

6.3 Concrete Construction from Bilinear Pairings

In this section, we present our group secret handshake scheme called GSH. The protocol involves a group administrator GA. In the following description $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1$ and $H_2 : \mathbb{G}_2 \rightarrow \{0, 1\}^n$ are cryptographic hash functions. H_1 and H_2 are considered as random oracles in the security analysis.

6.3.1 The Scheme

GSH.CreateGroup. GA runs BDH parameter generator to generate a prime q , two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q , and an bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Choose three random generators $P, Q, N \in \mathbb{G}_1$. Then GA picks a random $s \in \mathbb{Z}_q^*$, sets $P_{pub} = sP$ and $Q_{pub} = sQ$. GA keeps s secret as the *master secret key* and publishes system parameters

$$\text{params} = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P_{pub}, P, Q_{pub}, Q, N, H_1, H_2\}$$

GSH.AddUser. When a user U_i with identity ID_i wishes to obtain a secret credential for his identity, GA computes the corresponding credential $S_{\text{ID}_i} = sH_1(\text{ID}_i)$, and returns S_{ID_i} to the user U_i .

GSH.HandShake Let U_1, \dots, U_n be the n users who want to conduct a group secret handshake. The protocol runs as follows:

- Each user U_i picks $k_i \xleftarrow{R} \mathbb{G}_2$ and $r_i \xleftarrow{R} \mathbb{Z}_p^*$. Then U_i computes $T_i^j = r_i H_1(\text{ID}_j) + r_i N$, where $1 \leq j \leq n$ and $j \neq i$. U_i then computes

$$C_i = \langle r_i H_1(\text{ID}_i), T_i^1, \dots, T_i^n, r_i P, r_i Q, \mathcal{L} \rangle,$$

where \mathcal{L} is a label that contains information about how “ T_i^j ” is associated with each receiver. Then U_i broadcasts C_i to all others.

- Let $C_j = \langle R_j, T_j^1, \dots, T_j^n, V_j, W_j, \mathcal{L} \rangle$. Upon receiving C_j , each responder U_i computes $K_i = k_i \cdot \hat{e}(S_{\text{ID}_i}, -H_1(\text{ID}_1) - \dots - H_1(\text{ID}_{i-1}) + H_1(\text{ID}_{i+1}) + \dots + H_1(\text{ID}_n))$ and $K'_i = k_i \cdot \hat{e}(r_i S_{\text{ID}_i}, -R_1 - \dots - R_{i-1} + R_{i+1} + \dots + R_n)$. U_i then computes

$$D_i = \langle \hat{e}(N, P_{\text{pub}})^{r_i} \cdot K_i, \hat{e}(N, Q_{\text{pub}})^{r_i} \cdot K'_i \rangle,$$

and broadcasts D_i to all others.

Let $D_j = \langle X_j, Y_j \rangle$. Now each user U_i , using \mathcal{L} , finds appropriate T_j^i and computes

$$K_1 K_2 \cdots K_n = \frac{\hat{e}(S_{\text{ID}_i}, V_1 + \dots + V_n)}{\hat{e}(T_1^i + \dots + T_n^i, P_{\text{pub}})} \cdot (X_1 X_2 \cdots X_n)$$

$$K'_1 K'_2 \cdots K'_n = \frac{\hat{e}(S_{\text{ID}_i}, W_1 + \dots + W_n)}{\hat{e}(T_1^i + \dots + T_n^i, Q_{\text{pub}})} \cdot (Y_1 Y_2 \cdots Y_n)$$

It is easy to see that the above equations are consistent. If C_i, D_i are valid messages,

$$\begin{aligned} & \frac{\hat{e}(S_{\text{ID}_i}, \sum_{j=1}^n V_j) \cdot \prod_{j=1}^n X_j}{\hat{e}(\sum_{j=1}^n T_j^i, P_{\text{pub}})} = \frac{\hat{e}(s H_1(\text{ID}_i), \sum_{j=1}^n r_j P) \cdot \prod_{j=1}^n X_j}{\hat{e}(\sum_{j=1}^n (r_j H_1(\text{ID}_i) + r_j N), sP)} \\ &= \frac{\hat{e}(\sum_{j=1}^n r_j H_1(\text{ID}_i), sP) \cdot \prod_{j=1}^n X_j}{\hat{e}(\sum_{j=1}^n (r_j H_1(\text{ID}_i) + r_j N), sP)} = \frac{\prod_{i=1}^n \hat{e}(N, P_{\text{pub}})^{r_i} \cdot K_i}{\hat{e}(\sum_{i=1}^n r_i N, sP)} = \prod_{i=1}^n K_i \end{aligned}$$

The same as above, we can also obtain

$$\frac{\hat{e}(S_{\text{ID}_i}, \sum_{j=1}^n W_j) \cdot \prod_{j=1}^n Y_j}{\hat{e}(\sum_{j=1}^n T_j^i, Q_{\text{pub}})} = \prod_{i=1}^n K'_i$$

Then each user U_i verifies and accepts only if the following equation holds

$$K_1 K_2 \cdots K_n \stackrel{?}{=} K'_1 K'_2 \cdots K'_n \quad (6.1)$$

If the above verification succeed, then U_1, \dots, U_n finish all the steps of the GSH, and the handshake has been successful.

Each U_i can also create a shared secret key for future communication as follows:

$$K = H_2(K_1 K_2 \cdots K_n)$$

GSH.TraceUser. Given a transcript of a handshake among U_1, \dots, U_n , the administrator **GA** can easily recover the S_{ID_i} of each user U_i by iterating over all the credentials **GA** has issued, and adding each attempt at simulating the verification of the handshake protocol. If **GA** outputs “accept” after the verification, then **GA** can look up which user this credential had been issued to.

GSH.RemoveUser. To remove a user U from the group G , the administrator can simply add the identity of the user to the \mathcal{RUL} , and encrypts the update information by using an Identity-Based Encryption scheme. Then **GA** distributes the information to the members of the group, alerting them to abort any handshake should they find themselves performing the handshake with a user using any identity on the \mathcal{RUL} .

Correctness

To check the correctness of the scheme, now we show that both sides of equation 6.1 are equal. Assume that each user has the correct credential from the **GA**.

First, we look into the left side of the equation:

$$\begin{aligned}
\prod_{i=1}^n K_i &= \prod_{i=1}^n \left(k_i \cdot \hat{e}(S_{\text{ID}_i}, -\sum_{j=1}^{i-1} H_1(\text{ID}_j) + \sum_{j=i+1}^n H_1(\text{ID}_j)) \right) \\
&= \prod_{i=1}^n \left(k_i \cdot \prod_{j=1}^{i-1} \hat{e}(H_1(\text{ID}_j), H_1(\text{ID}_i))^{-s} \cdot \prod_{j=i+1}^n \hat{e}(H_1(\text{ID}_i), H_1(\text{ID}_j))^s \right) \\
&= \prod_{i=1}^n k_i \cdot \prod_{i=1}^n \left(\prod_{j=1}^{i-1} \hat{e}(H_1(\text{ID}_j), H_1(\text{ID}_i))^{-s} \cdot \prod_{j=i+1}^n \hat{e}(H_1(\text{ID}_i), H_1(\text{ID}_j))^s \right) \\
&= \prod_{i=1}^n k_i
\end{aligned}$$

Then we look into the right side:

$$\begin{aligned}
\prod_{i=1}^n K'_i &= \prod_{i=1}^n \left(k_i \cdot \hat{e}(r_i S_{\text{ID}_i}, -\sum_{j=1}^{i-1} R_j + \sum_{j=i+1}^n R_j) \right) \\
&= \prod_{i=1}^n \left(k_i \cdot \prod_{j=1}^{i-1} \hat{e}(r_j H_1(\text{ID}_j), r_i H_1(\text{ID}_i))^{-s} \cdot \prod_{j=i+1}^n \hat{e}(r_i H_1(\text{ID}_i), r_j H_1(\text{ID}_j))^s \right) \\
&= \prod_{i=1}^n k_i \cdot \prod_{i=1}^n \left(\prod_{j=1}^{i-1} \hat{e}(r_j H_1(\text{ID}_j), r_i H_1(\text{ID}_i))^{-s} \cdot \prod_{j=i+1}^n \hat{e}(r_i H_1(\text{ID}_i), r_j H_1(\text{ID}_j))^s \right) \\
&= \prod_{i=1}^n k_i
\end{aligned}$$

6.3.2 Security Proof

Theorem 7 *The above GSH scheme is impersonation resistant under the Bilinear Diffie-Hellman assumption in the Random Oracle Model.*

Proof. Assume that an adversary \mathcal{A} violates the impersonation resistant property. Now we show how to construct an attacker \mathcal{B} for solving the BDH problem. Suppose that \mathcal{B} is given $(q, \mathbb{G}_1, \mathbb{G}_2, P, aP, bP, cP)$ as an instance of the BDH problem. Attacker \mathcal{B} interacts with \mathcal{A} as follows:

Phase 1: Suppose that \mathcal{A} outputs target multiple identities $\Delta = (\text{ID}_1^*, \dots, \text{ID}_n^*)$, where $\text{ID}_{\mathcal{A}} = \text{ID}_i^*$ and $i \in [1, n]$.

Phase 2: \mathcal{B} sets $P_{\text{pub}} = cP$, and gives $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P_{\text{pub}}, P, Q_{\text{pub}}, Q, N, H_1\}$ to \mathcal{A} as the system parameters, where H_1 is a random oracle controlled by \mathcal{B} as follows:

Upon receiving a random oracle query ID_j to H_1 :

- If there exists (ID_j, l_j, L_j) in H_1List , return L_j . Otherwise, do the following:
 - * If $\text{ID}_j = \text{ID}_{\mathcal{A}}$, where $\text{ID}_{\mathcal{A}}$ is the identity of the adversary \mathcal{A} , randomly choose $l_j \in \mathbb{Z}_q^*$ and set $L_j = aP$.
 - * Else if $\text{ID}_j \neq \text{ID}_n^*$, choose $l_j \in \mathbb{Z}_q^*$ uniformly at random and compute $L_j = l_jP$.
 - * If $\text{ID}_j = \text{ID}_n^*$, search H_1List to get l_j that corresponds to ID_i^* for $i \in [1, n)$, and compute $L_j = (\sum_{j=1}^{i-1} l_j - \sum_{j=i+1}^{n-1} l_j)P + bP$.
 - * Put (ID_j, l_j, L_j) in H_1List and return L_j as answer.

Phase 3: \mathcal{B} answers \mathcal{A} 's private key extraction queries as follows:

Upon receiving a private key extraction query on ID_j :

- If there exists (ID_j, l_j, L_j) in H_1List , compute $S_{\text{ID}_j} = l_jP_{\text{pub}}$. Otherwise, do the following:
 - * Choose $l_j \in \mathbb{Z}_q^*$ uniformly at random and compute $S_{\text{ID}_j} = l_jP_{\text{pub}}$.
 - * Put (ID_j, l_j, L_j) in H_1List and return S_{ID_j} as answer.

Phase 4: \mathcal{B} acts as honest parties and broadcasts messages as follows:

- Choose $r_j \in \mathbb{Z}_q^*$, where $j \in [1, n)$.
- Compute $R_j = r_jP$, and $R_n = (\sum_{j=1}^{i-1} r_j - \sum_{j=i+1}^{n-1} r_j)P$.
- Randomly choose $T_j^1, \dots, T_j^n, V_j, W_j$ for $j \in [1, n]$ and $j \neq i$.
- Return $C_j = (R_j, T_j^1, \dots, T_j^n, V_j, W_j)$ to \mathcal{A} .

Phase 5: \mathcal{B} answers \mathcal{A} 's random oracle/private key extraction queries as in Phase 2 and 3.

Phase 6: \mathcal{A} returns the messages $\langle X_{\mathcal{A}}, Y_{\mathcal{A}} \rangle$.

Analysis: Note that if $\langle X_{\mathcal{A}}, Y_{\mathcal{A}} \rangle$ is the valid response, ID_j^* , where $j \in [1, n)$, can extract the $(K_{\mathcal{A}}, K'_{\mathcal{A}})$. Then $K_{\mathcal{A}} \cdot K'_{\mathcal{A}}{}^{-1}$ should be identical to $(k_{\mathcal{A}} \cdot \hat{e}(S_{\text{ID}_{\mathcal{A}}}, - \sum_{j=1}^{i-1} H_1(\text{ID}_j^*) +$

$\sum_{j=i+1}^{n-1} H_1(\text{ID}_j^*) + H_1(\text{ID}_n^*) \cdot (k_{\mathcal{A}} \cdot \hat{e}(r_{\mathcal{A}} S_{\text{ID}_{\mathcal{A}}}, -\sum_{j=1}^{i-1} R_j + \sum_{j=i+1}^{n-1} R_j + R_n))^{-1} = \hat{e}(S_{\text{ID}_{\mathcal{A}}}, bP)$. Since $S_{\text{ID}_{\mathcal{A}}} = caP$, \mathcal{B} now gets $\hat{e}(P, P)^{abc} = \hat{e}(caP, bP)$. Consequently, we obtain

$$\text{Adv}_{\mathcal{A}}^{\text{Impersonate}} < |\Pr[\mathcal{B}(P, aP, bP, cP) = \hat{e}(P, P)^{abc}]| = \text{Adv}_{\mathcal{A}}^{\text{BDH}}$$

Theorem 8 *The above GSH scheme is Detection Resistant under the Decisional Bilinear Diffie-Hellman assumption in the Random Oracle Model.*

Proof. Assume that one party U_v identified by ID_v is involved in a failure handshake protocol. An attacker \mathcal{A} , who also participates the handshake, can know whether U_v is an honest party or not.

Now we show how to construct an attacker \mathcal{B} for solving the DBDH problem. Suppose that \mathcal{B} is given $(q, \mathbb{G}_1, \mathbb{G}_2, P, aP, bP, cP, \gamma)$ as an instance of the DBDH problem. Attacker \mathcal{B} interacts with \mathcal{A} as follows:

Phase 1: Suppose that \mathcal{A} outputs two target multiple identities $\Delta = (\text{ID}_1^*, \dots, \text{ID}_{i-1}^*, \text{ID}_{i+1}^*, \text{ID}_n^*)$ and $\text{ID}_i^0, \text{ID}_i^1$, where $\text{ID}_{\mathcal{A}} \neq \text{ID}_i^*, i \in [1, n]$. ID_i^0 is a member, who holds an valid credential. ID_i^1 is a unqualified member, who does not have the valid group credential.

Phase 2: \mathcal{B} sets $P_{\text{pub}} = cP$, and gives $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P_{\text{pub}}, P, Q_{\text{pub}}, Q, N, H_1\}$ to \mathcal{A} as the system parameters, where H_1 is a random oracle controlled by \mathcal{B} as follows:

Upon receiving a random oracle query ID_j to H_1 :

- If there exists (ID_j, l_j, L_j) in H_1List , return L_j . Otherwise, do the following:
 - * If $\text{ID}_j = \text{ID}_v$, randomly choose $l_j \in \mathbb{Z}_q^*$, and set $L_j = aP$.
 - * Else if $\text{ID}_j \neq \text{ID}_n^*$, choose $l_j \in \mathbb{Z}_q^*$ uniformly at random and compute $L_j = l_j P$.
 - * If $\text{ID}_j = \text{ID}_n^*$, search H_1List to get l_j that corresponds to ID_j^* for $j \in [1, n]$, and compute $L_j = (\sum_{j=1}^{i-1} l_j - \sum_{j=i+1}^{n-1} l_j)P + bP$.
 - * Put (ID_j, l_j, L_j) in H_1List and return L_j as answer.

Phase 3: \mathcal{B} answers \mathcal{A} 's private key extraction queries as follows:

Upon receiving a private key extraction query on ID_j :

- If there exists (ID_j, l_j, L_j) in H_1List , compute $S_{ID_j} = l_j P_{pub}$. Otherwise, do the following:
 - * Choose $l_j \in \mathbb{Z}_q^*$ uniformly at random and compute $S_{ID_j} = l_j P_{pub}$.
 - * Put (ID_j, l_j, L_j) in H_1List and return S_{ID_j} as answer.

Phase 4: \mathcal{B} now simulates the handshake protocol as follows:

\mathcal{B} choose (k_v^0, k_v^1, k_v) , where $k_v^0 = k_v$ and $k_v^1 \neq k_v$.

- Choose $\beta \in \{0, 1\}$ at random.
- Choose $t_j, r_j \in \mathbb{Z}_q^*$, where $j \in [1, n), j \neq i$.
- Compute $R_j = t_j P$, and $R_n = (\sum_{j=1}^{i-1} t_j - \sum_{j=i+1}^{n-1} t_j) P$.
- Compute $T_j^k = r_j l_k P + r_j N, V_j = r_j P, W_j = r_j Q$, for $j, k \in [1, n]$.
- Return $C_j = (R_j, T_j^1, \dots, T_j^n, V_j, W_j)$ to \mathcal{A} .
- Return D_j following the valid scheme for $j \neq i$.
- Return $D_i = (\hat{e}(N, P_{pub})^{r_v} \cdot \gamma k_v^\beta, \hat{e}(N, Q_{pub})^{r_v} \cdot k_v)$.

Phase 5: \mathcal{B} answers \mathcal{A} 's random oracle/private key extraction queries as in Phase 2 and 3.

Phase 6: \mathcal{A} outputs its guess β' . If $\beta' = \beta$, \mathcal{B} outputs 1. Otherwise, it outputs 0.

Analysis: Let ϵ denote the probability $\text{Adv}_{\mathcal{A}}^{\text{Detect}}$. We note that if $\gamma = \hat{e}(P, P)^{abc}$,

$$\begin{aligned} \gamma k_v^\beta &= \hat{e}(acP, bP) k_v^\beta = \hat{e}(S_{ID_v}, bP) k_v^\beta \\ &= k_v^\beta \cdot \hat{e}(S_{ID_v}, - \sum_{j=1}^{i-1} H_1(ID_j^*) + \sum_{j=i+1}^n H_1(ID_j^*)) \cdot \hat{e}(r_v S_{ID_v}, - \sum_{j=1}^{i-1} R_j + \sum_{j=i+1}^n R_j)^{-1} \end{aligned}$$

It is clear that from the construction above, \mathcal{B} simulates the random oracle H_1 and the private key extraction in Phase 3 and 5. Hence, we obtain $\Pr[\mathcal{B}(P, aP, bP, cP, \hat{e}(P, P)^{abc}) = 1] = \Pr[\beta = \beta']$, where $|\Pr[\beta' = \beta] - \frac{1}{2}| > \epsilon$, and then $\Pr[\mathcal{B}(P, aP, bP, cP, \gamma) = 1] = \Pr[\beta = \beta'] = \frac{1}{2}$, where γ is uniform. Consequently, we obtain

$$\begin{aligned} \text{Adv}_{\mathcal{A}}^{\text{DBDH}} &= |\Pr[\mathcal{B}(P, aP, bP, cP, \hat{e}(P, P)^{abc}) = 1] - \Pr[\mathcal{B}(P, aP, bP, cP, \gamma) = 1]| \\ &> |(\frac{1}{2} \pm \epsilon) - \frac{1}{2}| = \text{Adv}_{\mathcal{A}}^{\text{Detect}} \end{aligned}$$

Theorem 9 *The above GSH scheme is Indistinguishable to eavesdropper under the Decisional Bilinear Diffie-Hellman assumption in the Random Oracle Model.*

Proof. Assume that if an adversary \mathcal{A} is able to eavesdrop all the transcripts in a secret handshake protocol, \mathcal{A} can distinguish between a successful handshake and an unsuccessful one.

Now we show how to construct an attacker \mathcal{B} for solving the DBDH problem. Suppose that \mathcal{B} is given $(q, \mathbb{G}_1, \mathbb{G}_2, P, aP, bP, cP, \gamma)$ as an instance of the DBDH problem. Attacker \mathcal{B} interacts with \mathcal{A} as follows:

Phase 1: Suppose that \mathcal{A} outputs two target multiple identities $\Delta_0 = (\text{ID}_1^*, \dots, \text{ID}_n^*)$ and $\Delta_1 = (\text{ID}_1, \dots, \text{ID}_n)$, where $\text{ID}_{\mathcal{A}} \neq \text{ID}_i^*$ and $\text{ID}_{\mathcal{A}} \neq \text{ID}_i, i \in [1, n]$. Δ_0 is a group, in which each user holds an valid credential. Δ_1 is a group, in which one or some users does not have the valid group credential.

Phase 2: \mathcal{B} sets $N = bP, P_{pub} = cP + Q_{pub}$, and gives $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P_{pub}, P, Q_{pub}, Q, N, H_1\}$ to \mathcal{A} as the system parameters, where H_1 is a random oracle controlled by \mathcal{B} as follows:

Upon receiving a random oracle query ID_j to H_1 :

- If there exists $(\text{ID}_j, l_j, r_j, L_j)$ in $H_1\text{List}$, return L_j . Otherwise, do the following:
 - * If $\text{ID}_j = \text{ID}_i^*$ for some $i \in [1, n]$, compute $L_j = l_jP - N$.
 - * Else choose $l_j, r_j \in \mathbb{Z}_q^*$ uniformly at random and compute $L_j = l_jP$.
 - * Put $(\text{ID}_j, l_j, r_j, L_j)$ in $H_1\text{List}$ and return L_j as answer.

Phase 3: \mathcal{B} answers \mathcal{A} 's private key extraction queries as follows:

Upon receiving a private key extraction query on ID_j :

- If there exists $(\text{ID}_j, l_j, r_j, L_j)$ in $H_1\text{List}$, compute $S_{\text{ID}_j} = l_jP_{pub}$. Otherwise, do the following:
 - * Choose $l_j, r_j \in \mathbb{Z}_q^*$ uniformly at random and compute $S_{\text{ID}_j} = l_jP_{pub}$.
 - * Put $(\text{ID}_j, l_j, r_j, L_j)$ in $H_1\text{List}$ and return S_{ID_j} as answer.

Phase 4: \mathcal{B} now simulates the handshake protocol as follows:

\mathcal{B} constructs three sequences $(K_1^0, K_2^0, \dots, K_n^0)$, $(K_1^1, K_2^1, \dots, K_n^1)$ and (K_1, K_2, \dots, K_n) , where $K_1^0 K_2^0 \dots K_n^0 = K_1 K_2 \dots K_n$, $K_1^1 K_2^1 \dots K_n^1 \neq K_1 K_2 \dots K_n$ and $K_i^0 \neq K_i^1 \neq K_i$.

- Choose $\beta \in \{0, 1\}$ at random.
- Search $H_1\text{List}$ to get l_j, r_j that corresponds to ID_j^* for $j \in [1, n]$.
- Compute $l_j r_i P$ for $i \in [1, n]$, $j \in [1, n]$.
- Compute $l_j(aP - \sum_{i=1}^{n-1} r_i P), \gamma K_i^\beta$ for $i = n$, $j \in [1, n]$, and choose $e \in \mathbb{Z}_q^*$ at random.
- Return $C_i = (l_i r_i P, l_1 r_i P, \dots, l_n r_i P, aP + eP, eP)$ and $D_i = (K_i^\beta, K_i)$ for $i \in [1, n]$ as transcripts in handshake protocol.
- Return $C_n = (l_n r_n P, l_1(aP - \sum_{i=1}^{n-1} r_i P), \dots, l_n(aP - \sum_{i=1}^{n-1} r_i P), aP + eP, eP)$ and $D_n = (\gamma K_n^\beta, K_n)$ as transcripts in handshake protocol.

Phase 5: \mathcal{B} answers \mathcal{A} 's random oracle/private key extraction queries as in Phase 2/3.

Phase 6: \mathcal{A} outputs its guess β' . If $\beta' = \beta$, \mathcal{B} outputs 1. Otherwise, it outputs 0.

Analysis: Let ϵ denote the probability $\text{Adv}_{\mathcal{A}}^{\text{Distinguish}}$. We note that if $\gamma = \hat{e}(P, P)^{abc}$,

$$\gamma \prod_{i=1}^n K_i^\beta = \hat{e}(bP, cP)^a \prod_{i=1}^n K_i^\beta = \hat{e}(\sum_{i=1}^n r_i N, P_{\text{pub}} - Q_{\text{pub}}) \prod_{i=1}^n K_i^\beta$$

Note also that

$$l_i r_i P = l_i r_i P - r_i N + r_i N = r_i(l_i P - N) + r_i N = r_i H_1(ID_i^*) + r_i N$$

for $i \in [1, n]$ and

$$\begin{aligned} l_n(aP - \sum_{i=1}^{n-1} r_i P) &= l_n(aP - \sum_{i=1}^{n-1} r_i P) - (aN - \sum_{i=1}^{n-1} r_i N) + (aN - \sum_{i=1}^{n-1} r_i N) \\ &= (a - \sum_{i=1}^{n-1} r_i)(l_n P - N) + (a - \sum_{i=1}^{n-1} r_i)N = (a - \sum_{i=1}^{n-1} r_i)H_1(ID_n^*) + (a - \sum_{i=1}^{n-1} r_i)N \end{aligned}$$

Hence C_i, D_i are valid messages. It is clear that from the construction above, \mathcal{B} simulates the random oracle H_1 and the private key extraction in Phase 3 and

5. Hence, we obtain $\Pr[\mathcal{B}(P, aP, bP, cP, \hat{e}(P, P)^{abc}) = 1] = \Pr[\beta = \beta']$, where $|\Pr[\beta' = \beta] - \frac{1}{2}| > \epsilon$, and $\Pr[\mathcal{B}(P, aP, bP, cP, \gamma) = 1] = \Pr[\beta = \beta'] = \frac{1}{2}$, where γ is uniform. Consequently, we obtain

$$\begin{aligned} \text{Adv}_{\mathcal{A}}^{\text{DBDH}} &= |\Pr[\mathcal{B}(P, aP, bP, cP, \hat{e}(P, P)^{abc}) = 1] - \Pr[\mathcal{B}(P, aP, bP, cP, \gamma) = 1]| \\ &> |(\frac{1}{2} \pm \epsilon) - \frac{1}{2}| = \text{Adv}_{\mathcal{A}}^{\text{Distinguish}} \end{aligned}$$

6.4 Summary

A GSH is an extension of the SH model which allows members of the same group to authenticate each other secretly, and the group affiliation of each member will never be disclosed if the handshake protocol fails. In this chapter, we firstly defined the security requirements of GSH scheme. Then we presented a group secret handshake scheme by using pairing on elliptic curves. We also proved that our scheme is secure under the BDH and DBDH assumption in the ROM.

Chapter 7

Conclusions and Future Work

In this thesis, we focused on the implementation of efficient credential schemes. The credential systems were proposed to protect the certificates of users while using in authentication. We briefly described the recent work on credential systems, and identified the challenges that we faced. To assist in understanding the descriptions in this thesis, we introduced a set of related cryptographic primitives. For the reason that credential systems are constructed based on digital signatures, so the security of the credential systems relies on how strong the signature schemes are. Hence we reviewed the basic security requirements of the digital signature. In order to illuminate the standard notion of security for a signature scheme that we adopted in this thesis, we also introduced the classification of different attacks to the signature schemes.

After introducing the definitions and generic schemes of different credential systems, we reviewed a list of credential schemes in literature. We analysed the OSBE schemes based on ElGamal family signatures, and found that their DSA-based OSBE scheme is flawed. Following the generic construction from OSBE to SH introduced along with the ElGamal family signature based OSBE scheme, we presented the ElGamal-based SH scheme from their ElGamal-based OSBE, where we noted that the scheme required four moves to be completed. Consequently, compared with other SH schemes, ElGamal-based SH scheme is less efficient in communication costs.

In order to complement the previous result, we constructed an ElGamal signature based key agreement protocol first. Based on this primitive, we proposed a new construction of ElGamal-based SH scheme, which requires only three moves. Since DSA signature is very similar to the ElGamal signature, but with different modulus, we firstly converted the DSA signature into the format of one modulus, then applied the ElGamal-based SH scheme to the DSA signature with a minor modification, and obtain a DSA-based SH scheme. We then proved these two SH schemes are secure.

To extend the SH scheme from the two-party scenario to the multi-party setting,

we considered implementing the multi-party SH scheme based on key agreement protocol, like the construction of our three-move SH scheme from the ElGamal-based key agreement scheme. When we reviewed the existing group key agreement scheme, we found that ID-based AGKA scheme is most suitable in implementing GSH scheme, and then we listed two efficient constructions found in literature. After analysing these two ID-based AGKA schemes, we observed that both schemes are flawed, and no existing secure ID-based AGKA protocol can be completed in a single round.

Therefore, we proposed our single round ID-based AGKA scheme. However, the one-round scheme involves more messages than prior work. Then we provide an alternative way of achieving an efficient two-round ID-based AGKA protocol. Comparing these two schemes, the first scheme costs more computing time, and involves more messages, but requires only one round, whereas the second scheme is more efficient in communication costs, even compared with the prior work in the literature. Thus, when these two schemes are implemented in some applications, the appropriate one can be chosen according to the function of the applications.

Subsequently, Jarecki *et al.* proposed a new GSH scheme by adding affiliation hiding authentication to a previously proposed group key agreement scheme. They also gave a definition of GSH scheme and described the security properties it should satisfy. However, they neglected the difference between two-party version and the group version. When a group of users wish to create a common session key, they may choose to conduct a group key agreement protocol. If an invalid user is involved in the protocol, there will be two results after they run the protocol: valid users share a common key, while the unqualified user learn nothing about this key; the keys calculated by all users, good and bad, are different from the one of each other. In the first case, though the invalid user cannot know the session key agreed on among other valid users, (s)he may learn that these users belong to the same group if they conduct further interactions. Hence (s)he is able to know that they are valid users.

The GSH scheme proposed by Jarecki *et al.* goes into the first case that we discussed above, in which an adversary can have knowledge of the group affiliation of other participants involved after they run the protocol. We emphasized this property when we redefined the security requirements of GSH scheme. Afterwards, we presented a GSH scheme without having this weakness, and provided a comprehensive security proof of our GSH scheme.

Future Works

We proposed two ID-based AGKA schemes by using bilinear pairing. The single round scheme is less efficient in computation, and another efficient scheme takes two rounds. We suggest that there may exist an efficient construction of ID-based AGKA scheme which only takes one round.

Furthermore, the size of the messages transferred in our ID-based AGKA schemes will grow with the increase in the numbers of participants involved in the protocol. The implementation of a constant size ID-based AGKA scheme is therefore desirable. In a constant size scheme, the length of the messages communicated in the protocol will not be affected by the increased scale of the group of participants. Hence how to construct a constant size ID-based AGKA, which is suitable for large-scale network, is still an open problem.

Additionally, our ID-based AGKA schemes are constructed by using pairing cryptography. Compared to classical public key cryptosystems, like RSA and Diffie-Hellman, the bilinear pairing operations are known to be costly operations. Hence, an interesting and challenging construction of one-round ID-based AGKA scheme based on classical public key cryptosystem remains an open problem.

Our proposed GSH scheme faces the similar problems. Thus the following work is expected to be done in the area of GSH:

- Implement a constant size GSH scheme, in which the size of the messages will not increase with the growing scale of the group.
- Propose an efficient GSH scheme based on traditional public key cryptosystem, which can make implementation easier and more efficient in both hardware and software.

Implementing an efficient credential system is a useful and interesting research area. We hope that our work has provided some valuable ideas, which can benefit future researchers.

Bibliography

- [1] M. Abadi. Private authentication. In *Proceedings of the Workshop on Privacy Enhancing Technologies: PET 2002*, volume 2482 of *Lecture Notes in Computer Science*, pages 27–40. Springer-Verlag, 2002.
- [2] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO '00: Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology*, volume 1880 of *Lecture Notes in Computer Science*, pages 255–270, London, UK, 2000. Springer-Verlag.
- [3] Available from <http://csrc.nist.gov/fips/>. *FIPS publication 180-1: Secure hash standard*, 1995.
- [4] J. Baek, R. Safavi-Naini, and W. Susilo. Efficient multi-receiver identity-based encryption and its application to broadcast encryption. In *PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 380–397. Springer-Verlag, 2005.
- [5] D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon, and H. Wong. Secret handshakes from pairing-based key agreements. In *In Proceedings of 2003 IEEE Symposium on Security and Privacy*, pages 180–196, 2003.
- [6] R. Barua, R. Dutta, and P. Sarkar. Extending joux’s protocol to multi party key agreement. In *INDOCRYPT 2003: 4th International Conference on Cryptology in India*, volume 2904 of *Lecture Notes in Computer Science*, pages 205–217. Springer-Verlag, 2003.
- [7] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In *CRYPTO '98: Proceedings of the 18th Annual International Cryptology Conference on Advances in Cryptology*,

- volume 1462 of *Lecture Notes In Computer Science*, pages 26–45, London, UK, 1998. Springer-Verlag.
- [8] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: formal definition, simplified requirements and a construction based on trapdoor permutations. In *EUROCRYPT 2003: Proceedings of the international conference on the theory and application of cryptographic techniques*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629. Springer-Verlag, 2003.
- [9] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *EUROCRYPT 2000: International Conference on the Theory and Application of Cryptographic Techniques*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155. Springer-Verlag, 2000.
- [10] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *CRYPTO '93: Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249, London, UK, 1993. Springer-Verlag.
- [11] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *CCS '93: Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73, New York, NY, USA, 1993. ACM Press.
- [12] M. Bellare and P. Rogaway. Provably secure session key distribution: The three party case. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 57–66, New York, NY, USA, 1995. ACM Press.
- [13] M. Bellare¹, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *The Cryptographers Track at the RSA Conference 2005 (CT-RSA 2005)*, volume 3376 of *Lecture Notes in Computer Science*, pages 136–153. Springer-Verlag, 2005.
- [14] I. Blake, G. Seroussi, and N. Smart. *Elliptic Curves in Cryptography*. Cambridge University Press, London Mathematical Society 265, 1999.
- [15] J. Bohli, B. Glas, and R. Steinwandt. Towards provably secure group key agreement building on group theory. Cryptology ePrint Archive, Report 2006/079, 2006. <http://eprint.iacr.org/>.

- [16] D. Boneh. The decision diffie-hellman problem. In *ANTS-III: Proceedings of the Third International Symposium on Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63, London, UK, 1998. Springer-Verlag.
- [17] D. Boneh and X. Boyen. Efficient selective-id secure identity based encryption without random oracles. In *Advances in Cryptology: EUROCRYPT 2005*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, 2004.
- [18] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology: CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer-Verlag, 2004.
- [19] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.
- [20] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. *Journal of Cryptology*, 17(4):297–319, 2004.
- [21] R. Bradshaw, J. Holt, and K. Seamons. Concealing complex policies with hidden credentials. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 146–157, New York, NY, USA, 2004. ACM Press.
- [22] E. Bresson, O. Chevassut, and D. Pointcheval. Provably authenticated group diffie-hellman key exchange - the dynamic case. In *ASIACRYPT 2001: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security*, volume 2248 of *Lecture Notes In Computer Science*, pages 290–309, London, UK, 2001. Springer-Verlag.
- [23] E. Bresson, O. Chevassut, D. Pointcheval, and J. Quisquater. Provably authenticated group diffie-hellman key exchange. In *CCS 2001: Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 255–264, New York, NY, USA, 2001. ACM Press.
- [24] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In *Advances in Cryptology: EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 275–286. Springer-Verlag, 1994.

- [25] J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76, London, UK, 2002. Springer-Verlag.
- [26] J. Camenisch and M. Stadler. Proof systems for general statements about discrete logarithms. Technical Report TR 260, ETH Zurich, 1997.
- [27] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing*, pages 209–218, New York, NY, USA, 1998. ACM Press.
- [28] C. Castelluccia, S. Jarecki, and G. Tsudik. Brief announcement: secret handshakes from ca-oblivious encryption. In *PODC '04: Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 394–394, New York, NY, USA, 2004. ACM Press.
- [29] C. Castelluccia, S. Jarecki, and G. Tsudik. Secret handshakes from ca-oblivious encryption. In *ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security*, volume 3329 of *Lecture Notes in Computer Science*, pages 293–307. Springer-Verlag, 2004.
- [30] D. Chaum and E. Van Heyst. Group signatures. In *Advances in Cryptology: EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer-Verlag, 1991.
- [31] Z. Cheng, L. Vasiu, and R. Comley. Pairing-based one-round tripartite key agreement protocols. Cryptology ePrint Archive, Report 2004/079, 2004. <http://eprint.iacr.org/>.
- [32] K. Y. Choi, J. Y. Hwang, and D. H. Lee. Efficient id-based group key agreement with bilinear maps. In *PKC 2004, 7th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2947 of *Lecture Notes in Computer Science*, pages 130–144. Springer-Verlag, 2004.
- [33] C. Cocks. An identity based encryption scheme based on quadratic residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363, London, UK, 2001. Springer-Verlag.

- [34] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
- [35] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. In *STOC '91: Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 542–552, New York, NY, USA, 1991. ACM Press.
- [36] X. Du, Y. Wang, J. Ge, and Y. Wang. Id-based authenticated two round multi-party key agreement. Cryptology ePrint Archive, Report 2003/247, 2003. <http://eprint.iacr.org/>.
- [37] X. Du, Y. Wang, J. Ge, and Y. Wang. An improved id-based authenticated group key agreement scheme. Cryptology ePrint Archive, Report 2003/260, 2003. <http://eprint.iacr.org/>.
- [38] R. Dutta and R. Barua. Constant round dynamic group key agreement. In *ISC 2005: Information Security, 8th International Conference*, volume 3650 of *Lecture Notes in Computer Science*, pages 74–88. Springer-Verlag, 2005.
- [39] R. Dutta and R. Barua. Overview of key agreement protocols. Cryptology ePrint Archive, Report 2005/289, 2005. <http://eprint.iacr.org/>.
- [40] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc.
- [41] A. Fernandes. Elliptic curve cryptography. *Dr. Dobb's Journal*, December 1999.
- [42] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Proceedings on Advances in cryptology: Crypto'86*, pages 186–194, London, UK, 1986. Springer-Verlag.
- [43] G. Frey, M. Muller, and H. Ruck. The tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, 45(5):1717–1719, 1999.
- [44] K. Frikken, M. Atallah, and J. Li. Hidden access control policies with hidden credentials. In *WPES '04: Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 27–27, New York, NY, USA, 2004. ACM Press.

- [45] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO '99: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, volume 1666 of *Lecture Notes In Computer Science*, pages 537–554, London, UK, 1999. Springer-Verlag.
- [46] M. Gagne. Applications of bilinear maps in cryptography. Master's thesis, University of Waterloo, 2002.
- [47] M. Girault. Self-certified public keys. In *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of Cryptographic Techniques*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer-Verlag, 1991.
- [48] S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, 17(2):281–308, 1988.
- [49] O. Goldreich, S. Goldwasser, and S. Micali. On the cryptographic applications of random function. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 276–288, New York, NY, USA, 1985. Springer-Verlag New York, Inc.
- [50] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.
- [51] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
- [52] D. Harkin and D. Carrel. *IETF RFC 2409: The Internet Key Exchange (IKE)*. Network Working Group, The Internet Society, November 1998.
- [53] J. Holt. Reconciling ca-oblivious encryption, hidden credentials, osbe and secret handshakes. Cryptology ePrint Archive, Report 2005/215, 2005. <http://eprint.iacr.org/>.
- [54] J. Holt, R. Bradshaw, K. Seamons, and H. Orman. Hidden credentials. In *WPES '03: Proceedings of the 2003 ACM workshop on Privacy in the electronic society*, pages 1–8, New York, NY, USA, 2003. ACM Press.
- [55] I. Ingemarsson, D. T. Tang, and C. K. Wong. A conference key distribution system. *IEEE Transactions on Information Theory*, IT-28(5):714–720, 1982.

- [56] S. Jarecki, J. Kim, and G. Tsudik. Authentication for paranoids: Multi-party secret handshakes. In *ACNS 2006: Applied Cryptography and Network Security, 4th International Conference*, volume 3989 of *Lecture Notes in Computer Science*, pages 325–339. Springer-Verlag, 2006.
- [57] A. Joux. A one round protocol for tripartite diffie-hellman. In *ANTS-IV: Proceedings of the 4th International Symposium on Algorithmic Number Theory*, volume 1838 of *Lecture Notes In Computer Science*, pages 385–394, London, UK, 2000. Springer-Verlag.
- [58] A. Joux. The weil and tate pairings as building blocks for public key cryptosystems. In *Algorithmic Number Theory: 5th International Symposium*, volume 2369 of *Lecture Notes in Computer Science*, pages 20–32. Springer-Verlag, 2002.
- [59] A. Joux and K. Nguyen. Separating decision diffie-hellman from diffie-hellman in cryptographic groups. Cryptology ePrint Archive, Report 2001/003, 2001. <http://eprint.iacr.org/>.
- [60] J. Katz, R. Ostrovsky, and M. Yung. Forward secrecy in password-only key exchange protocols. In *Security in Communication Networks: Third International Conference (SCN 2002)*, volume 2576 of *Lecture Notes in Computer Science*, pages 29–44. Springer-Verlag, 2003.
- [61] J. Katz and M. Yung. Scalable protocols for authenticated group key exchange. In *Advances in Cryptology: CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 110–125. Springer-Verlag, 2003.
- [62] A. Kiayias and M. Yung. Group signatures: Provable security, efficient constructions and anonymity from trapdoor-holders. Cryptology ePrint Archive, Report 2004/076, 2004. <http://eprint.iacr.org/>.
- [63] S. Kim, M. Mambo, T. Okamoto, H. Shizuya, M. Tada, and D. Won. On the security of the okamoto-tanaka id-based key exchange scheme against active attacks. *IEICE Trans. Fundamentals*, E84-A(1):231–238, 2001.
- [64] S. Kim, M. Mambo, H. Shizuya, and D. Won. On the security of the okamoto-tanaka id-based key exchange scheme against active attacks. In *JW-ISC 2000: Proceedings of Japan-Korea Joint Workshop on Information Security and Cryptology*, pages 191–198, 2000.

-
- [65] M. Koblitz. Elliptic curve cryptosystems. In *Mathematics of Computation*, volume 48, pages 203–209, 1987.
- [66] N. Li, W. Du, and D. Boneh. Oblivious signature-based envelope. In *Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing*, pages 182–189. ACM Press, 2003.
- [67] M. Mambo and H. Shizuya. A note on the complexity of breaking okamoto-tanaka id-based key exchange scheme. In *PKC '98: Proceedings of the First International Workshop on Practice and Theory in Public Key Cryptography*, volume 1431 of *Lecture Notes In Computer Science*, pages 258–262, London, UK, 1998. Springer-Verlag.
- [68] A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC press, 1996.
- [69] A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*, chapter 11. CRC press, 2nd edition, 2001.
- [70] V. Miller. Uses of elliptic curves in cryptography. In *Advances in Cryptology: Crypto'85*, pages 417–426. Springer-Verlag, 1986.
- [71] S. Mitsunari, R. Sakai, and M. Kasahara. A new traitor tracing. *IEICE Transactions*, E82(1):1–4, 1999.
- [72] S. Nasserian and G. Tsudik. Revisiting oblivious signature-based envelopes. In *Financial Cryptography and Data Security, 10th International Conference(FC'2006)*, volume 4107 of *Lecture Notes in Computer Science*, pages 221–235. Springer-Verlag, 2006.
- [73] J. Nechvatal. *Public-Key Cryptography*. National Institute of Standards and Technology, 1992.
- [74] K. Nyberg and R. Rueppel. A new signature scheme based on the dsa giving message recovery. In *CCS '93: Proceedings of the 1st ACM conference on Computer and communications security*, pages 58–61, New York, NY, USA, 1993. ACM Press.
- [75] National Institute of Standards and Technology. *Digital Signature Standard*, page 186. NIST FIPS PUB, U.S. Department of Commerce, 1994.

- [76] S. Oh, M. Mambo, H. Shizuya, and D. Won. On the security of girault key agreement protocols against active attacks. *IEICE Trans. Fundamentals*, E86-A(5):1181–1189, 2003.
- [77] E. Okamoto and K. Tanaka. Key distribution systems based on identification information. *IEEE Journal on Selected Areas in Communications*, 7(4):481–485, 1989.
- [78] P. Van Oorschot and M. Wiener. Parallel collision search with application to hash functions and discrete logarithms. In *CCS '94: Proceedings of the 2nd ACM Conference on Computer and communications security*, pages 210–218, New York, NY, USA, 1994. ACM Press.
- [79] D. Pointcheval and J. Stern. Security proofs for signature schemes. In *EUROCRYPT '96: International Conference on the Theory and Application of Cryptographic Techniques*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398. Springer-Verlag, 1996.
- [80] D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology: the journal of the International Association for Cryptologic Research*, 13(3):361–396, 2000.
- [81] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Cambridge, MA, USA, 1979.
- [82] C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO '91: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, volume 576 of *Lecture Notes In Computer Science*, pages 433–444, London, UK, 1992. Springer-Verlag.
- [83] R. L. Rivest. *IETF RFC 1321: The MD5 Message-Digest Algorithm*. Internet Activities Board, 1992.
- [84] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.
- [85] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *SCIS 2000: 2000 Symposium on Cryptography and Information Security*, pages 26–28, 2000.

- [86] C. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991.
- [87] C. Schnorr and S. Vaudenay. The black-box model for cryptographic primitives. *Journal of Cryptology*, 11(2):125–140, 1998.
- [88] A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology: Crypto'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1984.
- [89] Y. Shi, G. Chen, and J. Li. Id-based one round authenticated group key agreement protocol with bilinear pairings. In *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing*, volume 1, pages 757–761, Washington, DC, USA, 2005. IEEE Computer Society.
- [90] G. Simmons. Contemporary cryptologythe science of information integrity. *Designs, Codes and Cryptography*, 4(2):193–195, 1992.
- [91] G. Tsudik and S. Xu. Brief announcement: a flexible framework for secret handshakes. In *PODC '05: Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, pages 39–39, New York, NY, USA, 2005. ACM Press.
- [92] G. Tsudik and S. Xu. Flexible framework for secret handshakes (multi-party anonymous and un-observable authentication). Cryptology ePrint Archive, Report 2005/034, 2005. <http://eprint.iacr.org/>.
- [93] D. Vergnaud. Rsa-based secret handshakes. In *WCC 2005: International Workshop on Coding and Cryptography*, volume 3969 of *Lecture Notes in Computer Science*, pages 252–274. Springer-Verlag, 2005.
- [94] X. Wang, D. Feng, X. Lai, and H. Yu. Collisions for hash functions md4, md5, haval-128 and ripemd. Cryptology ePrint Archive, Report 2004/199, 2004. <http://eprint.iacr.org/>.
- [95] X. Wang, Y. Yin, and H. Yu. Finding collisions in the full sha-1. In *Advances in Cryptology: CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer-Verlag, 2005.

- [96] X. Wang and H. Yu. How to break md5 and other hash functions. In *Advances in Cryptology: EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer-Verlag, 2005.
- [97] S. Xu. On the security of group communication schemes based on symmetric key cryptosystems. In *SASN '05: Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*, pages 22–31, New York, NY, USA, 2005. ACM Press.
- [98] S. Xu and M. Yung. k-anonymous secret handshakes with reusable credential. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 158–167, New York, NY, USA, 2004. ACM Press.
- [99] G. Yuval. How to swindle rabin. *Cryptologia*, 3(3):187–190, 1979.
- [100] F. Zhang and X. Chen. Attack on two id-based authenticated group key agreement schemes from pkc 2004. *Information Processing Letters*, 91(4):191–193, 2004.
- [101] F. Zhang, S. Liu, and K. Kim. Id-based one round authenticated tripartite key agreement protocol with pairings. Cryptology ePrint Archive, Report 2005/034, 2005. <http://eprint.iacr.org/>.
- [102] L. Zhou, W. Susilo, and Y. Mu. Efficient id-based authenticated group key agreement from bilinear pairings. In *Proceedings of The Second International Conference on Mobile Ad Hoc and Sensor Networks (MSN 2006)*, volume 4325 of *Lecture Notes In Computer Science*, pages 288–297. Springer-Verlag, 2006.
- [103] L. Zhou, W. Susilo, and Y. Mu. Three-round secret handshakes based on elgamal and dsa. In *Proceedings of The Second Information Security Practice and Experience Conference (ISPEC 2006)*, volume 3903 of *Lecture Notes In Computer Science*, pages 332–342. Springer-Verlag, 2006.