WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2011; **11**:226–239
Published online 15 April 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.971

SPECIAL ISSUE PAPER

Authenticated key exchange protocol with selectable identities

Hua Guo[1], Yi Mu[2], Xiyong Zhang[3] and Zhoujun Li[4]

[1] School of Computer Science and Engineering, Beihang University, Beijing, P.R.C.
[2] School of Computer Science Software Engineering, University of Wollongong, NSW, Australia
[3] Zhengzhou Information Science and Technology Institute, Zhengzhou, P.R.C.
[4] Key Laboratory of Beijing Network Technology, Beihang University, Beijing, P.R.C.

ABSTRACT

In the traditional identity-based cryptography, a user, who holds multiple identities, has to manage multiple private keys, where each private key is associated with an identity. In this paper, we present a key agreement protocol, which allows a single private key to map multiple public keys (identities) that are selectable by the user. That is, the established session key is associated with an arbitrary subset of identities held by the user, while the unselected identities remain secret to other participants. As a bonus, our scheme can be considered as a credential-based key agreement, where the unique private key can be treated as a credential of the user and the user only proves that his credential is associated with some selected identities. We prove that our scheme is secure in the random oracle model. Copyright © 2010 John Wiley & Sons, Ltd.

KEYWORDS

identity-based key exchange; key management; security; pairing

***Correspondence**

Hua Guo, School of Computer Science and Engineering, Beihang University, Beijing, P.R.C.
E-mail: hguo.xyz@163.com

1. INTRODUCTION

The advantage of identity-based key agreements over Public Key Infrastructure (PKI) based key agreements lies in public key handling, i.e., the participants do not need public key certificates. More precisely, in an identity-based system, a user's identity can serve as a public key without the need of a traditional PKI, and the corresponding private key is created by binding the identity string with a master secret of a trusted authority called Key Generation Center (KGC). Inspired by Shamir's first identity-based cryptographic scheme [1], many identity-based key agreement schemes were proposed without using bilinear pairing. The introduction of parings to cryptography [2,3] has opened an entirely new field for identity-based cryptography. Many novel identity-based key agrement protocols from pairings have been introduced (e.g., [4–8]).

We are motivated by the following identity-based scenario where multiple identities of a user must be applied. As a user, Alice holds multiple identities such as her name, birthday, credit card nubmber/pin, bank account number, driver's licence number, etc; each can serve as her public key which has a corresponding private key. Consider the situation where she wants to establish a secure session channel with organizations such as her bank, her insurance company, or her lawyer. When she talks to her bank *via* a computer network, she could have to use her identities such as her name, bank account number, and credit card number, along with the associated private keys to establish a shared session key with the bank, while other information could be regarded as private. When she has to talk to her car insurance company, she might have to use her name, birthday, and driver's licence number, along with the associated private keys to establish a session key. The obvious problem is that she has to handle all those private keys, especially when her private key set is large. Moreover, using multiple private keys and public keys in key agreement could result in a great computational complexity.

The challenge to the above scenario is how to construct an identity-based key agreement that accommodates the need of multiple identities and simplifies private key management. More explicitly, we are looking for a solution that multiple identities are associated with a single private key. We notice that recently, Guo *et al*. [9,10] introduced an encryption scheme that captures the feature of multiple identities. Their scheme provides us with an inspiration to handle multiple identities in key agreement.

In this paper, we present the first authenticated key agreement protocol that captures the feature of multiple identities. We name it "selectable identity key authenticated key agreement." In our scheme, each user holds multiple identities associated with a single private key. By "selectable identity" we meant that the user can select subset of her full identity set. Our scheme is proven secure in the random oracle model.

We present an entire new protocol for authenticated key exchange. Differing from Reference [9], we construct a concrete multi-identity key exchange protocol, which can be considered as a credential-based key agreement protocol. We notice that the security proof in Reference [9] is not completely correct (See Appendix A). In this paper, to give a sound proof of our key exchange scheme, we apply a build-in function presented by Chen *et. al.* [8] to our scheme and introduce a new assumption, k-multiple bilinear collision attack assumption (k-MBCAA1), and show that the hardness of which is equivalent to that of the known k-bilinear collision attack assumption. We prove the security of our scheme under the assumption of k-MBCAA1 in the random oracle model.

The rest of this paper is organized as follows. In Section 2, we describe the preliminaries including bilinear pairing and introduce a new security assumption and prove its soundness. In Section 3, we present a two-identity key agreement protocol. In Section 4, we introduce a multi-identity key agreement protocol. In Section 5, we prove the security of the multi-identity key agreement protocol. In Section 6, we conclude the paper.

2. PRELIMINARIES

In this section, we introduce some basic concepts, including the pairing primitives, assumptions, and the security model.

2.1. Bilinear pairing and security assumptions

We briefly review some basic facts of pairings, which will be used in our protocol.

Definition 1. *Let $\mathbb{G}$ be an additive group of prime order q and $\mathbb{G}_T$ a multiplicative group of the same order. Let P denote a generator of $\mathbb{G}$. An admissible pairing is a bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ which has the following properties:*

1. *Bilinear: given Q, $R \in \mathbb{G}$ and $a, b \in Z_q^*$, we have $\hat{e}(aQ, bR) = \hat{e}(Q, R)^{ab}$.*
2. *Non-degenerate: $\hat{e}(P, P) \neq 1_{\mathbb{G}_T}$.*
3. *Computable: $\hat{e}$ is efficiently computable.*

Collision Attack Assumption (k-CAA1) [11].
For an integer k, and $x \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}$, given $(P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P))$, where $h_i \in_R \mathbb{Z}_q^*$ and are different from each other for $0 \leq i \leq k$, computing $\frac{1}{x+h_0}P$ is hard.

Multiple Collision Attack Assumption (k-MCAA1).
For an integer k, and $x \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}$, given $(P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P))$, where $h_i \in_R \mathbb{Z}_q^*$ and are different from each other for $0 \leq i \leq k$, computing $\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P$ is hard.

Theorem 1. *If there exists a polynomial time algorithm to solve k-CAA1 problem, then there exists a polynomial time algorithm for k-MCAA1 problem. If there exists a polynomial time algorithm to solve k-MCAA1 problem, then there exists a polynomial time algorithm for k-CAA1 problem.*

Proof. Provided in Appendix B. The proof follows from the following Lemma. ■

Lemma 1. *For given integers n, m satisfying $0 \leq n \leq m-1$, and $(\frac{1}{h_1+x}, \cdots, \frac{1}{h_m+x})$ where $x, h_i \in_R \mathbb{Z}_q^*$ and h_i are different from each other for $1 \leq i \leq m$, there exists a unique solution $(c_1, \cdots, c_m) \in \mathbb{Z}_q^m$ for the equation*

$$\frac{x^n}{(h_1+x)\cdots(h_m+x)} = \frac{c_1}{h_1+x} + \cdots + \frac{c_m}{h_m+x}$$

Proof. Provided in Appendix C. ■

Bilinear Collision Attack Assumption (k-BCAA1) [11].
For an integer k, and $x \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}$, $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, given $(P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P))$, where $h_i \in_R \mathbb{Z}_q^*$ and are different from each other for $0 \leq i \leq k$, computing $\hat{e}(P, P)^{\frac{1}{x+h_0}}$ is hard.

Multiple Bilinear Collision Attack Assumption (k-MBCAA1).
For an integer k, and $x \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}$, $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, given $(P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P))$, where $h_i \in_R \mathbb{Z}_q^*$ and are different from each other for $0 \leq i \leq k$, computing $\hat{e}(P, P)^{\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}}$ is hard.

Theorem 2. *If there exists a polynomial time algorithm to solve k-BCAA1 problem, then there exists a polynomial time algorithm for k-MBCAA1 problem. If there exists a polynomial time algorithm to solve k-MBCAA1 problem, then there exists a polynomial time algorithm for k-BCAA1 problem.*

Proof. Provided in Appendix B. ■

2.2. Security model

In this paper, we shall adopt a modified security model proposed by Bellare and Rogaway [12] to analyze the security of our multi-identity key exchange protocol.

The model includes a set of parties and each party involved in a session is modeled by an oracle. An oracle $\Pi_{i,j}^s$ denotes an instance of a party i involved with a partner party j in a session s where the instance of the party j is

$\Pi^t_{j,i}$ for some t. These parties cannot communicate directly; instead they only communicate with each other *via* an adversary. An adversary can access the oracle by issuing some specified queries as follows.

Send($\Pi^s_{i,j}, m$): This query models an active attack. $\Pi^s_{i,j}$ executes the protocol and responds with an outgoing message x or a decision to indicate accepting or rejecting the session. If the oracle $\Pi^s_{i,j}$ does not exist, it will be created. Note that if $m = \lambda$, then the oracle is generated as an initiator; otherwise as a responder.

Reveal($\Pi^s_{i,j}$): $\Pi^s_{i,j}$ returns the session key as its response if the oracle accepts. Otherwise, it returns $\perp$. Such an oracle is called *opened*.

Corrupt(i): The party i responds with its private key.

Test($\Pi^s_{i,j}$): At some point, the adversary can make a Test query to a fresh oracle $\Pi^s_{i,j}$. $\Pi^s_{i,j}$, as a challenger, randomly chooses $b \in \{0, 1\}$ and responds with the real agreed session key, if $b = 0$; otherwise it returns a random sample generated according to the distribution of the session key.

The security of a protocol is defined using the two-phase game $\mathcal{G}$ played between a malicious adversary $\mathcal{A}$ and a collection of oracles. At the first stage, $\mathcal{A}$ is able to send the above first three oracle queries at will. Then, at some point, $\mathcal{A}$ will choose a fresh session $\Pi^s_{i,j}$ on which to be tested and send a Test query to the fresh oracle associated with the test session. After this point, the adversary can continue querying the oracles but cannot reveal the test oracle or its partner, and cannot corrupt the entity j. Eventually, $\mathcal{A}$ terminates the game simulation and outputs a bit b' for b. we say $\mathcal{A}$ wins if the adversary guesses the correct b.

Define the advantage of $\mathcal{A}$ as:

$$\mathrm{Adv}^{\mathcal{A}}(k) = |2\Pr[b' = b] - 1|$$

where k is a security parameter.

The fresh oracle in the game is defined as follows.

Definition 2 (Fresh oracle [13])**.** *An oracle $\Pi^s_{i,j}$ is called fresh if (1) $\Pi^s_{i,j}$ has accepted; (2) $\Pi^s_{i,j}$ is unopened; (3) $j \neq i$ is not corrupted; (4) there is no opened oracle $\Pi^t_{j,i}$, which has had a matching conversation to $\Pi^s_{i,j}$.*

In this work, we use the concatenation of the messages in a session to define the session ID, thus to define the matching conversation, i.e., two oracles $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$ have a matching conversation to each other if both of them have the same session ID.

A secure authenticated key agreement protocol is defined as follows.

Definition 3. *Protocol Π is a secure authenticated key agreement protocol, if:*

> *In the presence of the benign adversary (who faithfully relays messages between parties), on $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$, both oracles always accept holding the same session key and this key is distributed uniformly at random on session key space;*
> *For every probability polynomial time(PPT) adversary $\mathcal{A}$, $\mathrm{Adv}^{\mathcal{A}}(k)$ is negligible.*

As mentioned in Reference [8], if a protocol is proved to be secure with respect to the above definition, then it achieves implicit mutual key authentication and the basic security properties, i.e., known session key security, key-compromise impersonation resilience and unknown key-share resilience.

3. THE TWO-IDENTITY KEY AGREEMENT PROTOCOL

We firstly present a simple two-identity key exchange protocol which is used to make the following multi-identity one to be understood easily. Here we suppose two parties A and B want to establish a shared session key using this protocol. A has two IDs ($ID_{A,1}$, $ID_{A,2}$) and B has two IDs ($ID_{B,1}$, $ID_{B,2}$).

3.1. The scheme

We assume the existence of a trusted KGC that is responsible for the creation and secure distribution of users private keys.

Setup: This algorithm takes a security parameter as its input and conducts the following steps:

1. Generate a prime q, and a bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, where $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order q. Choose two generators $P, Q \in \mathbb{G}^*$ randomly.
2. Choose a random value $s \in \mathbb{Z}^*_q$ and compute $P^k_{\mathrm{pub}} = s^k P$, $k = 1, 2$. Note $P^0_{\mathrm{pub}} = P$.
3. Choose two cryptographic hash functions $H_1 : \{0, 1\}^* \to \mathbb{Z}^*_q$ and $H_2 : \{0, 1\}^* \to \{0, 1\}^n$ for some n.

The KGC publishes

$$params = \langle q, \mathbb{G}, \mathbb{G}_T, \hat{e}, n, P, Q, P^1_{\mathrm{pub}}, P^2_{\mathrm{pub}}, H_1, H_2 \rangle$$

as the system parameters, and keeps s as his own secret master key. The parameters are distributed to the users of the system through a secure authenticated channel.

Extract:

The KGC takes as input *params*, master key s, and the identities $ID_{\mathrm{Ident},i} \in \{0, 1\}^*$, where Ident $\in \{A, B\}$ and $i \in \{1, 2\}$, generates the private key

$$d_{\mathrm{Ident}} = \frac{1}{H_1(ID_{\mathrm{Ident},1}) + s} \cdot \frac{1}{H_1(ID_{\mathrm{Ident},2}) + s} Q$$

where $H_1(ID_{\mathrm{Ident},1})$, $H_1(ID_{\mathrm{Ident},2}) \neq -s \bmod q$, and sends it to the user.

Suppose A and B want to use $ID_{\mathrm{Ident},1}$ as their public keys. Compute

$$Q^k_{ID_{\mathrm{Ident},1}} = H_1(ID_{\mathrm{Ident},1}) \cdot P^k_{\mathrm{pub}}, \ k = 0, 1$$

Key Agreement:

To establish a shared session key, A and B take the follow steps:

1. A chooses x from $\mathbb{Z}_q^*$ as the ephemeral key, and computes the corresponding ephemeral public keys

$$M_{A,1} = xQ^0_{ID_{B,1}} + xP^1_{\text{pub}}$$
$$M_{A,2} = xQ^1_{ID_{B,1}} + xP^2_{\text{pub}}$$

then sends $T_A = (M_{A,1}, M_{A,2})$ to B.
2. Upon receiving the messages from A, B does the following:

Choose y from $\mathbb{Z}_q^*$ as his ephemeral key, and compute the corresponding ephemeral public keys

$$M_{B,1} = yQ^0_{ID_{A,1}} + yP^1_{\text{pub}}$$
$$M_{B,2} = yQ^1_{ID_{A,1}} + yP^2_{\text{pub}}$$

then send $T_B = (M_{B,1}, M_{B,2})$ to A

Compute

$$U_B = M_{A,2} + H_1(ID_{B,2}) \cdot M_{A,1}$$

Compute

$$K_{BA} = \hat{e}(U_B, d_B) \cdot \hat{e}(P, Q)^y$$

3. Upon receiving the messages from B, A firstly computes

$$U_A = M_{B,2} + H_1(ID_{A,2}) \cdot M_{B,1}$$

then computes the shared secret as

$$K_{AB} = \hat{e}(U_A, d_A) \cdot \hat{e}(P, Q)^x$$

3.2. Protocol correctness

Now we verify the correctness of the protocol.

$$\begin{aligned} U_A &= M_{B,2} + H_1(ID_{A,2}) \cdot M_{B,1} \\ &= (yQ^1_{ID_{A,1}} + yP^2_{\text{pub}}) + H_1(ID_{A,2}) \\ &\quad \cdot (yQ^0_{ID_{A,1}} + yP^1_{\text{pub}}) \\ &= (ysH_1(ID_{A,1})P + ys^2P) + yH_1(ID_{A,2}) \\ &\quad \cdot (H_1(ID_{A,1})P + ysP) \\ &= (yH_1(ID_{A,1}) + s)(H_1(ID_{A,2}) + s)P \end{aligned}$$

Thus the session secret is computed as

$$\begin{aligned} K_{AB} &= \hat{e}(U_A, d_A) \cdot \hat{e}(P, Q)^x \\ &= \hat{e}(y(H_1(ID_{A,1}) + s)(H_1(ID_{A,2}) + s)P, \\ &\quad \frac{1}{H_1(ID_{A,1}) + s} \cdot \frac{1}{H_1(ID_{A,2}) + s} Q) \\ &\quad \cdot \hat{e}(P, Q)^x \\ &= \hat{e}(P, Q)^y \cdot \hat{e}(P, Q)^x \\ &= \hat{e}(P, Q)^{x+y} \end{aligned}$$

Similarly, we can obtain

$$U_B = x(H_1(ID_{B,1}) + s)(H_1(ID_{B,2}) + s)P$$

and

$$K_{BA} = \hat{e}(P, Q)^{x+y}$$

Thus, two secret keys computed by A and B are equal, i.e., A and B have successfully established the shared key $K = K_{AB} = K_{BA}$ after running an instance of the protocol. The final shared session key is then $sk = H_2(A \| B \| T_A \| T_B \| K)$.

4. MULTI-IDENTITY KEY AGREEMENT PROTOCOL

In this section, we present a multi-identity key exchange protocol. For simplicity, we suppose that two involved parties hold the same number of identities. Suppose two parties A and B want to establish a shared session key using this protocol. A holds p IDs ($ID_{A,1}, ID_{A,2}, \cdots, ID_{A,p}$) and B holds p IDs ($ID_{B,1}, ID_{B,2}, \cdots, ID_{B,p}$).

4.1. The scheme

Setup:

This algorithm takes a security parameter as its input and conducts the following steps:

1. Generate a prime q, and a bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, where $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order q. Choose two generators $P, Q \in \mathbb{G}^*$ randomly.
2. Choose a random value $s \in \mathbb{Z}_q^*$ and compute $P^k_{\text{pub}} = s^k P$, $k = 1, 2, \cdots, p$. Note $P^0_{\text{pub}} = P$.
3. Choose three cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \mathbb{G} \to \mathbb{G}$ and $H_3 : \{0,1\}^* \to \{0,1\}^n$ for some n.

The KGC publishes

$$params = \langle q, p, \mathbb{G}, \mathbb{G}_T, \hat{e}, n, P, Q, P^1_{\text{pub}}, \cdots, P^p_{\text{pub}}, H_1, H_2, H_3 \rangle$$

as the system parameters, and keeps s as his own secret master key. The parameters are distributed to the users of the system through a secure authenticated channel.

Extract: The KGC takes as input *params*, master key s, and the identities $ID_{\text{Ident},i} \in \{0,1\}^*$, where Ident $\in \{A, B\}$ and $i \in \{1, 2, \cdots, p\}$, generates the private key

$$d_{\text{Ident}} = \frac{1}{H_1(ID_{\text{Ident},1}) + s} \cdot \frac{1}{H_1(ID_{\text{Ident},2}) + s} \cdot \cdots \cdot \frac{1}{H_1(ID_{\text{Ident},p}) + s} Q$$

where $H_1(ID_{\text{Ident},k}) \neq -s \bmod q$, $k = 1, 2, \cdots, p$. KGC sends the private key to the user.

Suppose A and B want to use $ID_{\text{Ident},i}$ as their public key. Compute

$$Q^k_{ID_{\text{Ident},i}} = H_1(ID_{\text{Ident},i}) \cdot P^k_{\text{pub}}$$

where $k = 0, 1, \cdots, p-1$, and

$$S^1_{\text{Ident},i} = \sum_{k=1,k\neq i}^{p} H_1(ID_{\text{Ident},k}),$$

$$S^2_{\text{Ident},i} = \sum_{k,l=1,k,l\neq i}^{p} H_1(ID_{\text{Ident},k}) H_1(ID_{\text{Ident},l}),$$

$$\cdots$$

$$S^{p-1}_{\text{Ident},i} = \prod_{k=1,k\neq i}^{p} H_1(ID_{\text{Ident},k})$$

Key Agreement: To establish a shared session key, A and B conduct the follow steps:

1. A chooses x from $\mathbb{Z}^*_q$ as the ephemeral key, and computes the corresponding ephemeral public keys

$$M_{A,1} = xQ^0_{ID_{B,i}} + xP^1_{\text{pub}},$$
$$M_{A,2} = xQ^1_{ID_{B,i}} + xP^2_{\text{pub}},$$
$$\cdots$$
$$M_{A,p} = xQ^{p-1}_{ID_{B,i}} + xP^p_{\text{pub}}$$
$$N_A = xH_2(M_{A,1})$$

   Then sends $T_A = (M_{A,1}, M_{A,2}, \cdots, M_{A,p}, N_A)$ to B.
2. Upon receiving the message from A, B does the following:
   Check if the equation

$$\hat{e}(M_{A,1}, H_2(M_{A,1})) = \hat{e}(Q^0_{ID_{B,i}} + P^1_{\text{pub}}, N_A)$$

   holds or not. If not, abort the session.
   Choose y from $\mathbb{Z}^*_q$ as the ephemeral key, and compute the corresponding ephemeral public keys

$$M_{B,1} = yQ^0_{ID_{A,i}} + yP^1_{\text{pub}},$$
$$M_{B,2} = yQ^1_{ID_{A,i}} + yP^2_{\text{pub}},$$
$$\cdots$$
$$M_{B,p} = yQ^{p-1}_{ID_{A,i}} + yP^p_{\text{pub}}$$
$$N_B = yH_2(M_{B,1}).$$

   Then sends $T_B = (M_{B,1}, M_{B,2}, \cdots, M_{B,p}, N_B)$ to A.
   Compute

$$U_B = M_{A,p} + S^1_{B,i} M_{A,p-1} + \cdots + S^{p-1}_{B,i} M_{A,1}$$

   Compute

$$K_{BA} = \hat{e}(U_B, d_B) \cdot \hat{e}(P, Q)^y$$

3. Upon receiving the message from B, A does the following:
   Check if the equation

$$\hat{e}(M_{B,1}, H_2(M_{B,1})) = \hat{e}(Q^0_{ID_{A,i}} + P^1_{\text{pub}}, N_B)$$

   holds or not. If not, abort the session.
   Compute

$$U_A = M_{B,p} + S^1_{A,i} M_{B,p-1} + \cdots + S^{p-1}_{A,i} M_{B,1}$$

   Compute

$$K_{AB} = \hat{e}(U_A, d_A) \cdot \hat{e}(P, Q)^x$$

4.2. Protocol correctness

Now we verify the correctness of the protocol.

$$\begin{aligned} U_A &= M_{B,p} + S^1_{A,i} M_{B,p-1} + \cdots + S^{p-1}_{A,i} M_{B,1} \\ &= (s + H_1(ID_{A,2})) \cdots (s + H_1(ID_{A,p})) M_{B,1} \\ &= y(s + H_1(ID_{A,1}))(s + H_1(ID_{A,2})) \cdot \\ &\quad \cdots (s + H_1(ID_{A,p}))P. \end{aligned}$$

Therefore, we compute the session secret as

$$\begin{aligned} K_{AB} &= \hat{e}(U_A, d_A) \cdot \hat{e}(P, Q)^x \\ &= \hat{e}(y(H_1(ID_{A,1}) + s)(s + H_1(ID_{A,2})) \\ &\quad \cdots (H_1(ID_{A,p}) + s)P, \frac{1}{H_1(ID_{A,1}) + s} \\ &\quad \frac{1}{H_1(ID_{A,2}) + s} \cdots \frac{1}{H_1(ID_{A,p}) + s} Q) \\ &\quad \cdot \hat{e}(P, Q)^x \\ &= \hat{e}(P, Q)^y \cdot \hat{e}(P, Q)^x \\ &= \hat{e}(P, Q)^{x+y} \end{aligned}$$

Similarly, we can obtain

$$U_B = x(H_1(ID_{B,1}) + s)(H_1(ID_{B,2}) + s) \cdots (H_1(ID_{B,p}) + s)P$$

Therefore,

$$K_{BA} = \hat{e}(U_B, d_B) \cdot \hat{e}(P, Q)^y = \hat{e}(P, Q)^{x+y}$$

Thus, the two secret keys computed by A and B are equal, i.e., A and B have successfully established the shared key $K = K_{AB} = K_{BA}$ after running an instance of the protocol. The final shared session key is then $sk = H_3(A\|B\|T_A\|T_B\|K)$.

Remark 1. *In the presentation, we have assumed that users select one ID from their ID set. It is not hard to generalize the protocol by allowing users to select multiple IDs. Without loosing generality, suppose B chooses two identities $ID_{A,i}$ and $ID_{A,j}$ $(i, j \in \{1, 2, \cdots, p\})$ as user A's public keys and generates the messages as follows:*

$$\begin{aligned}
M_{B,1} &= y(P_{\text{pub}}^2 + (H_1(ID_{A,i}) + H_1(ID_{A,j})) \\
&\quad \cdot P_{\text{pub}}^1 + H_1(ID_{A,i})H_1(ID_{A,j}) \cdot P_{\text{pub}}^0), \\
M_{B,2} &= y(P_{\text{pub}}^3 + (H_1(ID_{A,i}) + H_1(ID_{A,j})) \\
&\quad \cdot P_{\text{pub}}^2 + H_1(ID_{A,i})H_1(ID_{A,j}) \cdot P_{\text{pub}}^1), \\
&\quad \cdots \\
M_{B,p-1} &= y(P_{\text{pub}}^p + (H_1(ID_{A,i}) + H_1(ID_{A,j})) \\
&\quad \cdot P_{\text{pub}}^{p-1} + H_1(ID_{A,i})H_1(ID_{A,j}) \cdot P_{\text{pub}}^{p-2}) \\
N_B &= yH_2(M_{B,1}).
\end{aligned}$$

Remark 2. *For simplicity, we suppose that both parties hold the same number of identities. It is easy to generalize it to the case in which they hold different numbers of identities. All we need to do is to modify $S_{A,i}^k$, $S_{B,i}^k$, U_A, and U_B slightly.*

5. SECURITY ANALYSIS

Theorem 3. *If H_1, H_2 and H_3 are random oracles and the $(pq_1\text{-}1)$-MBCAA1 assumption holds, then our multi-identity scheme is a secure key agreement protocol. In particular, suppose $\mathcal{A}$ is an adversary that attacks the multi-identity scheme in which each party has p identities in the random oracle model with non-negligible probability ϵ and makes at most q_1, q_3 queries to H_1 and H_3, respectively, and creates at most q_o oracles. Then there exists an algorithm $\mathcal{B}$ to solve the $(pq_1\text{-}1)$-MBCAA1 problem with advantage*

$$Adv_{\mathcal{B}}^{(pq_1-1)-\text{MBCAA1}} \geq \frac{1}{p \cdot q_1 \cdot q_0 \cdot q_3} \cdot \epsilon$$

Proof. Firstly, we define Session ID as a concatenation of $T_A \parallel T_B$. We focus on how to construct an algorithm $\mathcal{B}$ using the adversary $\mathcal{A}$ to solve a $(pq_1\text{-}1)$-MBCAA1 problem with non-negligible probability.

Given an instance of the $(pq_1\text{-}1)$-MBCAA1 problem

$$\left\langle q, p, n, \mathbb{G}, \mathbb{G}_T, \hat{e}, P, Q, sQ, \left(h_{1,1}, \left(h_{1,2}, \frac{1}{h_{1,2}+s}Q\right), h_{1,3}, \frac{1}{h_{1,3}+s}Q\right) \cdots, \left(h_{q_1,p}, \frac{1}{h_{q_1,p}+s}Q\right)\right)\right\rangle$$

where $h_{i,j} \in_R \mathbb{Z}_q^*$ for $1 \leq i \leq q_1$ and $1 \leq j \leq p$, $\hat{e}$ is a bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, $\mathcal{B}$'s task is computing $\hat{e}(Q, Q)^{\frac{1}{(h_{1,1}+s)(h_{1,2}+s)\cdots(h_{q_1,p}+s)}}$. ■

Setup: $\mathcal{B}$ simulates the Setup algorithm as follows:

With Lemma 1, compute

$$\begin{aligned}
P &= \frac{1}{(s+h_{1,2})(s+h_{1,3})\cdots(s+h_{q_1,p})}Q \\
&= \sum_{j=2}^{p} \frac{c_{1,j}}{s+h_{1,j}}Q + \sum_{i=2,j=1}^{q_1,p} \frac{c_{i,j}}{s+h_{i,j}}Q
\end{aligned}$$

where $c_{i,j}$ are computable from $h_{i,j}$.

For $k = 1, 2, \cdots, p$, by Lemma 1 compute

$$\begin{aligned}
s^k P &= \frac{s^k}{(s+h_{1,2})(s+h_{1,3})\cdots(s+h_{q_1,p})}Q \\
&= \sum_{j=2}^{p} \frac{c_{1,j,k}}{s+h_{1,j}}Q + \sum_{i=2,j=1}^{q_1,p} \frac{c_{i,j,k}}{s+h_{i,j}}Q
\end{aligned}$$

where $c_{i,j,k}$ are computable from $h_{i,j}$.

After computing $P_{\text{pub}}^k = s^k P$ where s is the master key which is unknown to the simulator, $\mathcal{B}$ sends the system parameters $\langle q, p, n, \mathbb{G}, \mathbb{G}_T, \hat{e}, P, Q, P_{\text{pub}}^1, \cdots, P_{\text{pub}}^p, H_1, H_2, H_3\rangle$ to $\mathcal{A}$. The hash functions H_1, H_2, and H_3 are random oracles controlled by $\mathcal{B}$.

Algorithm $\mathcal{B}$ randomly chooses $I \in_R \{1, \cdots q_1\}$ and $J \in_R \{1, \cdots, q_0\}$ and begins its simulation. Here we should note that the notation $ID_{i,u}$ means the uth identity of the user i, $M_{i,u}^k$ denotes this is the kth message generated by user i using uth identity from user $j$as the public key where j is the user with whom i wants to communicate, and $\Pi_{i,j}^s$ is the sth oracle among all the created oracles. Another thing we should note is that all of the superscripts and subscripts in a message during the proof are all known to the simulator. Algorithm $\mathcal{B}$ answers the queries which are asked by adversary $\mathcal{A}$ in arbitrary order as follows.

$H_1(ID_{i,u})$ queries: Algorithm $\mathcal{B}$ maintains an initially empty list H_1^{list} with entries of the form $(i, ID_{i,u}, (h_{i,1}, h_{i,2}, \cdots, h_{i,q}), d_i)$. When $\mathcal{A}$ queries the oracle H_1 at a point $ID_{i,u}$, $\mathcal{B}$ responds to the query in the followig way:

If $ID_{i,u}$ already appears on the H_1^{list} in a tuple $(i, ID_{i,u}, (h_{i,1}, h_{i,2}, \cdots, h_{i,q}), d_i)$, then $\mathcal{B}$ responds with $H_1(ID_{i,u}) = h_{i,u}$.

Otherwise, if $ID_{i,u}$ is the Ith unique query to H_1, then $\mathcal{B}$ stores $(i, ID_{i,u}, (h_{1,1}, h_{1,2}, \cdots, h_{1,q}), \perp)$ into the tuple list and responds with $H_1(ID_{i,u}) = h_{1,u}$.

Otherwise, $\mathcal{B}$ randomly selects $h_{i,k}(i > 1, k = 1, 2, \cdots, p)$ from the $(pq_1\text{-}1)$-MBCAA1 instance according to $h_{i,u}$ which has not been chosen by $\mathcal{B}$, then computes

$$d_i = \frac{1}{(s + h_{i,1})(s + h_{i,2}) \cdots (s + h_{i,p})} Q = \sum_{j=1}^{p} \frac{c_{i,j}}{s + h_{i,j}} Q$$

where $c_{i,j}$ are computable from $h_{i,j}$.

After that, $\mathcal{B}$ inserts $(i, ID_{i,u}, (h_{i,1}, h_{i,2}, \cdots, h_{i,q}), \frac{1}{(h_{i,1}+s)(h_{i,2}+s)\cdots(h_{i,q}+s)} Q)$ into the tuple list and responds with $H_1(ID_{i,u}) = h_{i,u}$.

$H_2(M^1_{i,u})$ queries: Algorithm $\mathcal{B}$ maintains an initially empty list H_2^{list} with entries of the form $(M^1_{i,u}, m_{i,u}, R_{i,u}, w_{i,u})$. $\mathcal{B}$ responds to the query in the follows way:

If a tuple $(M^1_{i,u}, m_{i,u}, R_{i,u}, w_{i,u})$ has already appeared on the H_2^{list}, then $\mathcal{B}$ responds with $R_{i,u}$.

Otherwise, $\mathcal{B}$ randomly selects $m_{i,u}$ $(i > 0) \in_R \mathbb{Z}_q^*$ and inserts $(M^1_{i,u}, m_{i,u}, m_{i,u}Q, \perp)$ into the tuple list. $\mathcal{B}$ responds with $m_{i,u}Q$.

$H_3(ID_{i,v}, ID_{j,u}, T^t_{i,v}, T^t_{j,u}, K^t_{i,j})$ queries: $\mathcal{B}$ maintains an initially empty list H_3^{list} with entries of the form $(ID_{i,v}, ID_{j,u}, T^t_{i,v}, T^t_{j,u}, K^t_{i,j}, \zeta^t)$ which is indexed by $(ID_{i,v}, ID_{j,u}, T^t_{i,v}, T^t_{j,u}, K^t_{i,j})$. $\mathcal{B}$ responds to the query in the following way.

If a tuple indexed by $(ID_{i,v}, ID_{j,u}, T^t_{i,v}, T^t_{j,u}, K^t_{i,j})$ is on the list, then $\mathcal{B}$ responds with ζ^t.

Otherwise, $\mathcal{B}$ chooses a random string $\zeta^t \in \{0, 1\}^n$ and inserts a new tuple $(ID_{i,v}, ID_{j,u}, T^t_{i,v}, T^t_{j,u}, K^t_{i,j}, \zeta^t)$ into the list H_3^{list} and returns ζ^t.

Corrupt(i) queries: When receiving this query, $\mathcal{B}$ goes through list H_1^{list}. If i is not on the list, $\mathcal{B}$ randomly chooses $u \in \{1, 2, \cdots, p\}$, and queries $H_1(ID_{i,u})$. If $H_1(ID_{i,u}) = h_{1,u}$, then $\mathcal{B}$ aborts the game (Event 1). Otherwise, $\mathcal{B}$ sends d_i to $\mathcal{A}$.

Send($\Pi^t_{i,j}$, $((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u})$) queries: $\mathcal{B}$ maintains a list for each oracle of the form $(\Pi^t_{i,j}, tran^t_{i,j}, r^t_{i,j}, K^t_{i,j}, SK^t_{i,j})$ where $M^k_{j,u}$ means this is the kth message generated by user j using uth identity from user i's p identities as the public key; $tran^t_{i,j}$ is the transcript of the oracle so far; $r^t_{i,j}$ is the random integer used by the oracle to generate the messages; $K^t_{i,j}$ and $SK^t_{i,j}$ are set $\perp$ initially. This list is updated in other queries as well. $\mathcal{B}$ proceeds in the following way:

$\mathcal{B}$ looks through the list H_1^{list}. If i is not on the list, $\mathcal{B}$ randomly chooses $u' \in \{1, 2, \cdots, p\}$, and queries $H_1(ID_{i,u'})$.

$\mathcal{B}$ computes

$$Q^k_{i,v} = H_1(ID_{j,v}) P^k_{\text{pub}} + P^{k+1}_{\text{pub}} \quad (k \in \{0, 1, \cdots, p-1\})$$

and

$$R_{i,v} = H_2(M^1_{i,v}).$$

After that, $\mathcal{B}$ checks t.

If $t = J$, $\mathcal{B}$ checks the value of d_j and gives the different response depending on it as below.

* If $d_j \neq \perp$ or $H_1(ID_{j,v}) \neq h_{1,1}$, $\mathcal{B}$ aborts the game (Event 2).
* Otherwise,

  If $T_{j,u} = ((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u})$ is not the last message, randomly sample $x \in \mathbb{Z}_q^*$ such that xP is not shown on the list of H_2^{list} as some $M^1_{i,v}$, and then randomly sample $w_{i,v} \in \mathbb{Z}_q^*$ and insert the tuple $(xP, \perp, w_{i,v} \cdot (h_{1,1} + s)Q, w_{i,v})$ into H_2^{list}.

  If $T_{j,u} = \lambda$, compute

$$M^{k+1}_{i,v} = xs^k P = xP^k_{\text{pub}}, \quad k = 0, 1, \cdots, p-1$$

and

$$N_{i,v} = rR_{i,v} = \frac{x}{h_{1,1} + s} \cdot (w_{i,v} \cdot (h_{1,1} + s)Q) = xw_{i,v}Q$$

where $r = \frac{x}{h_{1,1}+s}$ which is unknown to the simulator. Obviously the equation

$$\hat{e}(M^1_{i,v}, H_2(M^1_{i,v})) = \hat{e}(Q^0_{ID_{j,v}} + P^1_{\text{pub}}, N_{i,v})$$

holds. Then respond with $T_{i,v} = ((M^1_{i,v}, M^2_{i,v}, \cdots, M^p_{i,v}), N_{i,v})$.

If $T_{j,u} = ((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u})$ is the first message of the session, then check if the equation

$$\hat{e}(M^1_{j,u}, H_2(M^1_{j,u})) = \hat{e}(Q^0_{ID_{i,u}} + P^1_{\text{pub}}, N_{j,u})$$

holds or not. If so, compute

$$M^{k+1}_{i,v} = xs^k P = xP^k_{\text{pub}}, \quad k = 0, 1, \cdots, p-1$$

and

$$N_{i,v} = rR_{i,v} = \frac{x}{h_{1,1} + s} \cdot (w_{i,v} \cdot (h_{1,1} + s)Q) = xw_{i,v}Q$$

where $r = \frac{x}{h_{1,1}+s}$ which is unknown to the simulator. Then respond with $T_{i,v} = ((M^1_{i,v}, M^2_{i,v}, \cdots, M^p_{i,v}), N_{i,v})$ and accept the session. Otherwise, reject the session.

If $T_{j,u} = ((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u})$ is the last message of the session, then check if the equation

$$\hat{e}(M^1_{j,u}, H_2(M^1_{j,u})) = \hat{e}(Q^0_{ID_{i,u}} + P^1_{\text{pub}}, N_{j,u})$$

holds or not. If so, do nothing but accept the session. Otherwise, reject the session.

If $t \neq J$, $\mathcal{B}$ proceeds the protocol as follows.

* If $T_{j,u} = ((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u})$ is not the second message on the transcript:
  If $d_i \neq \perp$, randomly sample $r^t_{i,j} \in \mathbb{Z}^*_q$.
  Otherwise, randomly sample $r^t_{i,j} \in \mathbb{Z}^*_q$, compute

$$M^{k+1}_{i,v} = r^t_{i,j} Q^k_{i,v} + r^t_{i,j} P^{k+1}_{\text{pub}},$$
$$N_{i,v} = r^t_{i,j} R_{i,v}$$

  where $k = 0, 1, \cdots, q-1$, so that they have not been shown on H^{list}_3 as a part of some $T_{i,v}$ if $\Pi^t_{i,j}$ is the initiator or $T_{j,u}$ otherwise.
* If $((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u}) = \lambda$, compute

$$M^{k+1}_{i,v} = r^t_{i,j} Q^k_{i,v} + r^t_{i,j} P^{k+1}_{\text{pub}},$$
$$N_{i,v} = r^t_{i,j} R_{i,v}$$

  where $k = 0, 1, \cdots, p-1$. Then respond with $T_{i,v} = ((M^1_{i,v}, M^2_{i,v}, \cdots, M^p_{i,v}), N_{i,v})$.
* Otherwise, check if the equation

$$\hat{e}(M^1_{j,u}, H_2(M^1_{j,u})) = \hat{e}(Q^0_{ID_{i,u}} + P^1_{\text{pub}}, N_{j,u})$$

  holds or not. If the equation does not hold, reject the session. Otherwise,
  - If $T_{j,u} = ((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u})$ is the first message of the session, compute

$$M^{k+1}_{i,v} = r^t_{i,j} Q^k_{i,v} + r^t_{i,j} P^{k+1}_{\text{pub}},$$
$$N_{i,v} = r^t_{i,j} R_{i,v}$$

    where $k = 0, 1, \cdots, p-1$.
    Then respond with $T_{i,v} = ((M^1_{i,v}, M^2_{i,v}, \cdots, M^p_{i,v}), N_{i,v})$.
  - If $T_{j,u} = ((M^1_{j,u}, M^2_{j,u}, \cdots, M^p_{j,u}), N_{j,u})$ is the last message of the session, then accept the session.

  For both cases, compute $SK^t_{i,j}$ as below. If $d_i \neq \perp$, compute

$$U_{i,u} = M^p_{j,u} + S^1_{i,u} M^{p-1}_{j,u} + \cdots + S^{p-1}_{i,u} M^1_{j,u}$$

and

$$K^t_{i,j} = \hat{e}(U_{i,u}, d_i) \cdot \hat{e}(P, Q)^{r^t_{i,j}}$$

where $M^{k+1}_{j,u}$ $(k = 0, 1, \cdots, p-1)$ are a part of the incoming messages and $r^t_{i,j}$ is selected randomly by oracle $\Pi^t_{i,j}$. If $d_i = \perp$, similarly we have

$$U_{i,u} = M^p_{j,u} + S^1_{i,u} M^{p-1}_{j,u} + \cdots + S^{p-1}_{i,u} M^1_{j,u},$$

and

$$K^t_{i,j} = \hat{e}(U_{i,u}, d_i) \cdot \hat{e}(P, Q)^{r^t_{i,j}}.$$

However, the simulator can not compute $K^t_{i,j}$ directly since the secret key d_i is unknown to the simulator.

If $M^1_{j,u} \neq xP$ and $h_{1,u} \neq h_{1,1}$, since $(h_{1,u}, \frac{1}{h_{1,u}+s} Q)$ is known to $\mathcal{B}$, thus compute

$$K^t_{i,j} = \hat{e}(U_{i,u}, d_i) \cdot \hat{e}(P, Q)^{r^t_{i,j}}$$
$$= \hat{e}(M^1_{j,u}, \frac{1}{h_{1,u}+s} Q) \cdot \hat{e}(P, Q)^{r^t_{i,j}}.$$

If $M^1_{j,u} \neq xP$ but $h_{1,u} = h_{1,1}$, since $\frac{1}{h_{1,1}+s} Q$ is unknown to $\mathcal{B}$, following the equation

$$\hat{e}(M^1_{j,u}, H_2(M^1_{j,u})) = \hat{e}(Q^0_{ID_{i,u}} + P^1_{\text{pub}}, N_{j,u}),$$

where $H_2(N_{j,u}) = m_{j,u} Q$ (find $m_{j,u}$ in the H^{list}_2), compute

$$\hat{e}(U_{i,u}, d_i) = \hat{e}(M^1_{j,u}, \frac{1}{s+h_{1,1}} Q)$$
$$= \hat{e}(P, \frac{1}{m_{j,u}} N_{j,u}).$$

Thus

$$K^t_{i,j} = \hat{e}(U_{i,u}, d_i) \cdot \hat{e}(P, Q)^{r^t_{i,j}}$$
$$= \hat{e}(P, \frac{1}{m_{j,u}} N_{j,u}) \cdot \hat{e}(P, Q)^{r^t_{i,j}}$$

where $N_{j,u}$ is a part of the incoming messages and $r^t_{i,j}$ is selected randomly by $\Pi^t_{i,j}$.

If $M^1_{j,u} = xP$, do nothing.

If $K^t_{i,j}$ is computed, then set $SK^t_{i,j} = H_3(ID_{i,v}, ID_{j,u}, T^t_{i,v}, T^t_{j,u}, K^t_{i,j})$ if i is the initiator, or $SK^t_{i,j} = H_3(ID_{j,u}, ID_{i,v}, T^t_{j,u}, T^t_{i,v}, K^t_{i,j})$ otherwise. If $K^t_{i,j}$ is not computed, then randomly sample $SK^t_{i,j}$.

Reveal($\Pi^t_{i,j}$) queries: $\mathcal{B}$ answers the queries as follows:

If oracle $\Pi^t_{i,j}$ has not accepted, then respond with $\perp$.

If $t = J$ or if the Jth oracle has been generated as $\Pi^J_{a,b}$ and $ID_{a,u} = ID_{j,u}$, $ID_{b,v} = ID_{i,v}$ and two oracles have the same session ID, then abort the game (Event 3).

Return $SK^t_{i,j}$.

Test($\Pi^t_{i,j}$) query: If $t \neq J$ or ($t = J$ but) and there is an oracle $\Pi^s_{j,i}$ which has the same session ID as $\Pi^t_{i,j}$ that has been revealed, $\mathcal{B}$ aborts the game (Event 4). Otherwise, $\mathcal{B}$ responds to $\mathcal{A}$ with a random number $\zeta \in \{0, 1\}^n$.

After $\mathcal{A}$ finishes the queries, it returns its guess. Then $\mathcal{B}$ proceeds with the following steps:

1. Firstly compute

$$U_{i,u} = M_{j,u}^{p} + S_{i,u}^{1} M_{j,u}^{p-1} + \cdots + S_{i,u}^{p-1} M_{j,u}^{1}$$

then compute $D = \hat{e}(U_{i,u},\ d_i)$ where $M_{j,u}^{k+1}$ $(k = 0, 1, \cdots, p-1)$ are messages generated by user j who uses the uth identity of user i as the public key, d_i is found from H_1^{list} corresponding to $ID_{i,u}$ of fresh oracle $\Pi_{i,j}^{J}$. Note that $H(ID_{j,v}) = h_{1,1}$. Thus

$$\begin{aligned} K_{i,j}^{J} &= \hat{e}(U_{i,u}, d_i) \cdot \hat{e}(P, Q)^{\frac{x}{h_{1,1}+s}} \\ &= D \cdot \left(\hat{e}(P, Q)^{\frac{1}{h_{1,1}+s}}\right)^{x} \\ &= D \cdot \left(\hat{e}(Q, Q)^{\frac{1}{(h_{1,1}+s)(h_{1,2}+s)\cdots(h_{q_1,p}+s)}}\right)^{x} \end{aligned}$$

2. $\mathcal{B}$ randomly samples K_l from the H_3^{list}, and returns $(K_l/D)^{\frac{1}{x}}$ as the response to the $(pq_1$-1)-MBCAA1 challenge.

Claim 1. *If B did not abort the game, A could not find inconsistence between the simulation and the real world. More precisely, if $\mathcal{A}$ noticed the inconsistence between the simulation and the real world when $\mathcal{B}$ did not abort the simulation, then the probability $\mathcal{B}$ solves the (pq_1-1)-MBCAA1 problem is non-negligible.*

Proof. $\mathcal{B}$ gives the satisfying response to most of the oracles by following the protocol specification honestly, except for the one $\Pi_{i,j}^{t}$ whose private key is $\perp$ and $H(ID_{i,u}) = h_{1,1}$ and the incoming messages $((M_{j,u}^{1}, M_{j,u}^{2}, \cdots, M_{j,u}^{p}),\ N_{j,u})$ is from the tested oracle where $M_{j,u}^{1} = xP$. Note that the transcripts are one part of the input to H_3 which is modelled as the random oracle to compute the session key. If there is some difference between the reveal query on $\Pi_{i,j}^{t}$ and a query on H_3, it must have queried H_3 with $\Pi_{i,j}^{t}$ such that

$$U_{i,u} = M_{j,u}^{p} + S_{i,u}^{1} M_{j,u}^{p-1} + \cdots + S_{i,u}^{p-1} M_{j,u}^{1}$$

and

$$\begin{aligned} K_{i,j}^{t} &= \hat{e}(U_{i,u}, d_i) \cdot \hat{e}(P, Q)^{r_{i,j}^{t}} \\ &= \hat{e}(P, Q)^{r_{i,j}^{t}} \cdot \left(\hat{e}(P, Q)^{\frac{1}{h_{1,1}+s}}\right)^{x}. \end{aligned}$$

If $\mathcal{A}$ can distinguish the session key $K_{i,j}^{t}$ in the simulation from the real world, then $\mathcal{B}$ can return $(K_l/\hat{e}(P, Q)^{r_{i,j}^{t}})^{\frac{1}{x}}$ as the response to the $(pq_1$-1)-MBCAA1 challenge with probability $\frac{1}{p \cdot q_1 \cdot q_0 \cdot q_3}$, where K_l is a random value choosing from H_3 by $\mathcal{B}$. This completes the proof. ■

Claim 2. *During the simulation, the probability that $\mathcal{B}$ did not abort the game is non-negligible.*

Proof. We now evaluate the probability that $\mathcal{B}$ did not abort during the game, i.e., Events 1 - 4 did not happen. $\mathcal{B}$ aborts the game only when at least one of following events happens:

Event 1, denoted as $\mathcal{F}_1$: $\mathcal{A}$ corrupted party i whose private key is represented by $\perp$, i.e., $\mathcal{A}$ made a query to party i to get its private key if it chose $\Pi_{j,i}^{s}$ as the fresh oracle, which is disallowed according to the definition of the fresh oracle.

Event 2, denoted as $\mathcal{F}_2$: $\mathcal{A}$ impersonated party i whose private key is represented by $\perp$ in the sth session.

Event 3, denoted as $\mathcal{F}_3$: $\mathcal{A}$ revealed the Jth oracle or its partner oracle, which is against the definition of the fresh oracle.

Event 4, denoted as $\mathcal{F}_4$: $\mathcal{A}$ did not choose the Jth oracle as the challenge fresh oracle or the parter of the fresh oracle has been revealed, which made that the test query cannot work.

According to the rules of the game, we have

$$\neg\mathcal{F}_4 \wedge \neg\mathcal{F}_2 \rightarrow \neg\mathcal{F}_1$$

and

$$\neg\mathcal{F}_4 \rightarrow \neg\mathcal{F}_3.$$

Let $\mathcal{F}_{21}$ be the event that $d_j \neq \perp$ during the simulation of the send query, and $\mathcal{F}_{22}$ be the event that $H_1(ID_{j,v}) \neq h_{1,1}$. Then we have

$$\begin{aligned} \Pr[\neg\mathcal{F}_2] &= \Pr[\neg(\mathcal{F}_{21} \vee \neg\mathcal{F}_{22})] \\ &= \Pr[\neg\mathcal{F}_{21} \wedge \neg\mathcal{F}_{22}] \\ &= \Pr[\neg\mathcal{F}_{22}] \\ &= \frac{1}{pq_1} \end{aligned}$$

Now let $\mathcal{F}$ be the event that $\mathcal{B}$ did not abort during the game. Then, we get

$$\begin{aligned} \Pr[\mathcal{F}] &= \Pr[\neg\mathcal{F}_1 \wedge \neg\mathcal{F}_2 \wedge \neg\mathcal{F}_3 \wedge \neg\mathcal{F}_4] \\ &= \Pr[\neg\mathcal{F}_2 \wedge \neg\mathcal{F}_4] \\ &= \Pr[\neg\mathcal{F}_2] \cdot Pr[\neg\mathcal{F}_4] \\ &\geq \frac{1}{pq_1} \cdot \frac{1}{q_o} \\ &= \frac{1}{p \cdot q_1 \cdot q_o}. \end{aligned}$$

■

Claim 3. *Let $\mathcal{H}$ be the event that*

$$K = \hat{e}(U_{i,u}, d_i) \cdot \hat{e}(P, Q)^{\frac{x}{h_{1,1}+s}}$$

was not queried on H_3. Then

$$\Pr[\neg\mathcal{H}] \geq \epsilon.$$

Proof. Similar to the analysis of Reference [14], we have

$$\Pr[\mathcal{A} \text{ wins}| \mathcal{H}] \leq \frac{1}{2}.$$

Thus

$$\begin{aligned}\Pr[\mathcal{A} \text{ wins}] &= \Pr[\mathcal{A} \text{ wins}|\neg\mathcal{H}] \cdot \Pr[\neg\mathcal{H}] \\ &\quad + \Pr[\mathcal{A} \text{ wins}|\mathcal{H}] \\ &\quad \cdot \Pr[\mathcal{H}] \\ &\leq \Pr[\mathcal{A} \text{ wins}|\neg\mathcal{H}] \cdot \Pr[\neg\mathcal{H}] \\ &\quad + \Pr[\mathcal{A} \text{ wins}|\mathcal{H}] \\ &\leq \frac{1}{2}\Pr[\neg\mathcal{H}] + \frac{1}{2}.\end{aligned}$$

So we have

$$\Pr[\neg\mathcal{H}] \geq 2\left(\Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}\right) = \epsilon.$$

Thus, the claim is correct.

Let $\mathcal{I}$ be the event that $\mathcal{B}$ found the correct K_I. Then combining all of the above results, we have

$$\begin{aligned}\Pr[\mathcal{B} \text{ wins}] &= \Pr[\mathcal{F} \wedge \neg H \wedge I] \\ &\geq \frac{1}{p \cdot q_1 \cdot q_o \cdot q_3} \cdot \Pr[\neg\mathcal{H}] \\ &\geq \frac{1}{p \cdot q_1 \cdot q_o \cdot q_3} \cdot \epsilon,\end{aligned}$$

which contradicts to the hardness of the (pq_1-1)-MBCAA1 problem.

This completes the security analysis of the protocol. ■

6. CONCLUSION

We proposed the first selectable identity authenticated key agreement protocol and proved it in the random oracle model and the k-multiple bilinear collision attack assumption (k-MBCAA1). To justify our security proof, we proved that k-MBCAA1 is equivalent to the hardness of the known k-bilinear collision attack assumption (k-BCAA1), where both were recognized as cryptographic hard problems.

Our scheme can be considered as a credential-based key authenticated key agreement, since the private key can be regarded as a credential authorized by the KGC. We believe that our protocol has great applicability as highlighted in the introduction.

ACKNOWLEDGEMENT

This work was supported by National Natural Science Foundation of China under grant no. 60973105, no. 60803154, no. 90718017; Research Fund for the Doctoral Program of Higher Education of China under grant no. 20070006055; Doctoral Innovation Foundation of Beihang University under grant no. 211619. Part of this work was done while the first author was with University of Wollongong, Australia.

Appendix A

We provide some comments on the security proof in Guo *et al.*'s encryption scheme [9]. The main problem in the security proof of Guo *et al.*'s encryption scheme is due to the hash function H_0, i.e., the hash function H_0 is treated improperly. Consider the following two cases:

H_0 is not treated as a random oracle. In this case, when the adversary queried H_0, the simulator should respond the right value as $H_0(ID) = b_i - s$ when coin =0, or $H_0(ID) = \frac{b_i s - s}{1 - b_i}$ otherwise. In this way the simulator can compute the correspondent values of Q_i^k when he is queried H_1-queries by the adversary. Although b_i is chosen by the simulator, the master key s is unknown to him.

H_0 is treated as a random oracle. In this case, when queried H_0-queries, the simulator would select a random value $\sigma \in \mathbb{Z}_p^*$. Then, the adversary $\mathcal{A}$ can use this value to compute Q_i^k. Also, the adversary $\mathcal{A}$ can obtain another value of Q_i^k by H_1-queries. When $\mathcal{A}$ compared the two values, he would realize the difference since the probability the two values equal is no more than $\frac{1}{p}$. Thus, from $\mathcal{A}$'s point of view, the simulation offered by the simulator is distinguishable from the real world.

The reason lies in the incompatibility between the scheme they presented and the proof they offered. In their scheme, they used the exponent inversion family [15] to generate the private keys; while in their security proof, they adopted the method from BF-IBE scheme which is based on the full-domain hash family [15] to generate the private keys.

In our paper, to relieve the tension between the scheme and the security proof, we apply the build-in function presented in Reference [8] to our key exchange protocol and the security proof.

Appendix B

Proof of Theorem 1.

If there is a polynomial time algorithm $\mathcal{A}$ to solve the k-CAA1 problem, we construct a polynomial time algorithm $\mathcal{B}$ to solve the k-MCAA1 problem.

Given an instance of k-MCAA1 problem

$$\left\langle P, xP, h_0, \left(h_1, \frac{1}{h_1 + x}P\right), \cdots, \left(h_k, \frac{1}{h_k + x}P\right)\right\rangle,$$

$\mathcal{B}$ computes $\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P$ as follows: Pass $\mathcal{A}$ the k-CAA1 challenge

$$\left\langle P, xP, h_0, \left(h_1, \frac{1}{h_1+x}P\right), \cdots, \left(h_k, \frac{1}{h_k+x}P\right)\right\rangle$$

and get

$$A = \frac{1}{x+h_0}P$$

By Lemma 1, we can compute

$$\begin{aligned} C &= \frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P \\ &= \frac{c_0}{x+h_0}P + \sum_{i=1}^{k}\frac{c_i}{x+h_i}P \\ &= c_0 \cdot A + \sum_{i=1}^{k}\frac{c_i}{x+h_i}P \end{aligned}$$

where $c_i \in \mathbb{Z}_q$.

If there is a polynomial time algorithm $\mathcal{A}$ to solve the k-MCAA1 problem, we construct a polynomial time algorithm $\mathcal{B}$ to solve the k-CAA1 problem.

Given an instance of k-CAA1 problem

$$\left\langle P, xP, h_0, \left(h_1, \frac{1}{h_1+x}P\right), \cdots, \left(h_k, \frac{1}{h_k+x}P\right)\right\rangle$$

$\mathcal{B}$ works as follows to compute $\frac{1}{x+h_0}P$: Pass $\mathcal{A}$ the k-MCAA1 challenge

$$\langle P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P)\rangle$$

and get

$$C = \frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P$$

By Lemma 1, there exists unique $(c_0, c_1, \cdots, c_k) \in \mathbb{Z}_q^{k+1}$ satisfying

$$\begin{aligned} &\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P \\ &= \frac{c_0}{x+h_0}P + \sum_{i=1}^{k}\frac{c_i}{x+h_i}P \end{aligned}$$

Thus, we have the solution

$$\frac{1}{x+h_0}P = \frac{1}{c_0}(C - \sum_{i=1}^{k}\frac{c_i}{x+h_i}P)$$

■

Proof of Theorem 2.

If there is a polynomial time algorithm $\mathcal{A}$ to solve the k-BCAA1 problem, we construct a polynomial time algorithm $\mathcal{B}$ to solve the k-MBCAA1 problem.

Given an instance of k-MBCAA1 problem

$$\langle P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P)\rangle$$

$\mathcal{B}$ computes $\hat{e}(P, P)^{\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}}$ as follows:

By Lemma 1, we can compute

$$\begin{aligned} C &= \frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P \\ &= \frac{c_0}{x+h_0}P + \sum_{i=1}^{k}\frac{c_i}{x+h_i}P \\ &= c_0 \cdot A + R \end{aligned}$$

where $c_i \in \mathbb{Z}_q$ and

$$R = \sum_{i=1}^{k}\frac{c_i}{x+h_i}P$$

Compute

$$\begin{aligned} B = \hat{e}(P, R) &= \hat{e}(P, \sum_{i=1}^{k}\frac{c_i}{x+h_i}P) \\ &= \prod_{i=1}^{k}\hat{e}(P, \frac{c_i}{x+h_i}P) = \prod_{i=1}^{k}\hat{e}(P, \frac{1}{x+h_i}P)^{c_i} \end{aligned}$$

Pass $\mathcal{A}$ the k-BCAA1 challenge

$$\langle P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P)\rangle$$

and get

$$A = \hat{e}(P, P)^{\frac{1}{x+h_0}}$$

Since

$$\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P = \frac{c_0}{x+h_0}P + R$$

compute

$$\begin{aligned} &\hat{e}(P, P)^{\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}} \\ &= \hat{e}(P, P)^{\frac{c_0}{x+h_0}+\sum_{i=1}^{k}\frac{c_i}{x+h_i}} \\ &= \hat{e}(P, P)^{\frac{c_0}{x+h_0}} \cdot \hat{e}(P, P)^{\sum_{i=1}^{k}\frac{c_i}{x+h_i}} \\ &= \hat{e}(P, \frac{1}{x+h_0}P)^{c_0} \cdot \hat{e}(P, \sum_{i=1}^{k}\frac{c_i}{x+h_i}P) \\ &= A^{c_0} \cdot B \end{aligned}$$

DOI: 10.1002/wcm

If there is a polynomial time algorithm $\mathcal{A}$ to solve the k-MBCAA1 problem, we construct a polynomial time algorithm $\mathcal{B}$ to solve the k-BCAA1 problem.

Given an instance of k-BCAA1 problem

$$\langle P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P)\rangle$$

$\mathcal{B}$ computes $\hat{e}(P, P)^{\frac{1}{x+h_0}}$ as follows:

By Lemma 1, we can compute

$$\begin{aligned}&\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P\\ &= \frac{c_0}{x+h_0}P + \sum_{i=1}^{k}\frac{c_i}{x+h_i}P\\ &= c_0\cdot\frac{1}{x+h_0}P + R\end{aligned}$$

where $c_i \in \mathbb{Z}_q$ and

$$R = \sum_{i=1}^{k}\frac{c_i}{x+h_i}P$$

Hence,

$$\frac{c_0}{x+h_0}P = \frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}P - R$$

Compute

$$B = \hat{e}(P, R) = \prod_{i=1}^{k}\hat{e}(P, \frac{1}{x+h_i}P)^{c_i}$$

Pass $\mathcal{A}$ the k-MBCAA1 challenge

$$\langle P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_k, \frac{1}{h_k+x}P)\rangle$$

and get

$$C = \hat{e}(P, P)^{\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}}$$

Compute

$$\begin{aligned}&\hat{e}(P, P)^{\frac{c_0}{x+h_0}}\\ &= \hat{e}(P, P)^{\frac{1}{(x+h_0)(x+h_1)\cdots(x+h_k)}}/\hat{e}(P, P)^{\sum_{i=1}^{k}\frac{c_i}{x+h_i}}\\ &= C/B\end{aligned}$$

Finally, compute

$$\hat{e}(P, P)^{\frac{1}{x+h_0}} = (C/B)^{c_0^{-1}}$$

■

Appendix C

Proof of Lemma 1.
Since

$$\begin{aligned}\sum_{i=1}^{m}\frac{c_i}{x+h_i} &= (c_1\prod_{i=2}^{m}(h_i+x) + c_2\prod_{i=1,i\neq 2}^{m}(h_i+x)\\ &+\cdots+c_m\prod_{i=1}^{m-1}(h_i+x))/((x+h_1)\cdot(x+h_2)\\ &\cdots(x+h_m))\end{aligned}$$

let

$$\begin{aligned}&f(c_1,\cdots,c_m)\\ &= c_1\prod_{i=2}^{m}(h_i+x) + c_2\prod_{i=1,i\neq 2}^{m}(h_i+x)+\cdots+\\ &c_m\prod_{i=1}^{m-1}(h_i+x)\\ &= x^{m-1}\sum_{i=1}^{m}c_i + x^{m-2}(c_1\sum_{i=2}^{m}h_i + c_2\sum_{i=1,i\neq 2}^{m}h_i +\\ &\cdots+c_m\sum_{i=1}^{m-1}h_i)+\cdots+(c_1\prod_{i=2}^{m}h_i + c_2\prod_{i=1,i\neq 2}^{m}h_i +\\ &\cdots+c_m\prod_{i=1}^{m-1}h_i)\\ &= x^{m-1}\sum_{i=1}^{m}c_i + x^{m-2}\sum_{i=1}^{m}c_i\sum_{j=1,j\neq i}^{m}h_j+\cdots\\ &+\sum_{i=1}^{m}c_i\prod_{j=1,j\neq i}^{m}h_j.\end{aligned}$$

Now let $f(c_1,\cdots,c_m) = x^{m-1-n}$ $(0\le n\le m-1)$. Then, we have

$$\begin{cases}c_1+c_2+\cdots+c_m = 0,\\ c_1\sum_{i=2}^{m}h_i + c_2\sum_{i=1,i\neq 2}^{m}h_i+\cdots+\\ c_m\sum_{i=1}^{m-1}h_i = 0,\\ \vdots\\ c_1\sum_{i_1\neq 1,\cdots,i_n\neq 1}\prod_{j=1}^{n}h_{i_j}+\cdots\\ +c_m\sum_{i_1\neq m,\cdots,i_n\neq m}\prod_{j=1}^{n}h_{i_j} = 1,\\ \vdots\\ c_1\prod_{i=2}^{m}h_i + c_2\prod_{i=1,i\neq 2}^{m}h_i+\cdots+\\ c_m\prod_{i=1}^{m-1}h_i = 0.\end{cases}$$

To show that the above linear equation system has a solution, it suffices to prove that the coefficient matrix is non-degenerative. In fact, the determinant of the matrix is

$$\begin{vmatrix} 1 & \cdots & 1 \\ \sum_{i=2}^{m} h_i & \cdots & \sum_{i=1}^{m-1} h_i \\ & & \vdots \\ \sum\limits_{i_1\neq 1,\cdots,i_n\neq 1} \prod_{j=1}^{n} h_{i_j} & \cdots & \sum\limits_{i_1\neq m,\cdots,i_n\neq m} \prod_{j=1}^{n} h_{i_j} \\ & & \vdots \\ \prod_{i=2}^{m} h_i & \cdots & \prod_{i=1}^{m-1} h_i \end{vmatrix}$$

By using row operations, it can be reduced to

$$\begin{vmatrix} 1 \cdots & 1 \\ 0 \cdots & h_1 - h_m \\ 0 \cdots & (h_1 - h_m)(h_2 - h_m) \\ \vdots & \\ 0 \cdots & (h_1 - h_m)(h_2 - h_m)\cdots(h_{m-1} - h_m) \end{vmatrix}$$

it equals

$$\prod_{1\le i<j\le m} (h_i - h_j) \neq 0$$

Thus, by the theory of linear algebra, there exists a unique solution $(c_1, \cdots, c_m) \in \mathbb{Z}_q^m$ for every equation $f(c_1, \cdots, c_m) = x^{m-1-n} (0 \le n \le m-1)$. Hence there exists a unique solution $(c_1, \cdots, c_m) \in \mathbb{Z}_q^m$ for the equation

$$\frac{x^n}{(h_1 + x)\cdots(h_m + x)} = \frac{c_1}{h_1 + x} + \cdots + \frac{c_m}{h_m + x}$$

For example, the solution for the case of $n = 0$ is

$$\begin{aligned} c_1 &= \tfrac{1}{(h_2-h_1)(h_3-h_1)\cdots(h_m-h_1)}, \\ c_2 &= \tfrac{1}{(h_1-h_2)(h_3-h_2)\cdots(h_m-h_2)}, \\ &\cdots \\ c_m &= \tfrac{1}{(h_1-h_m)(h_2-h_m)\cdots(h_{m-1}-h_m)}. \end{aligned}$$

■

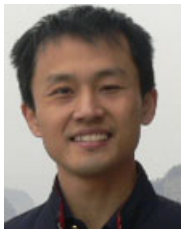
REFERENCES

1. Shamir A. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology-Crypto'84, LNCS 196*. 1984; 47–53.
2. Sakai R, Ohgishi K, Kasahara M. Cryptosystems based on pairing. *Symposium on Cryptography and Information Security, 2000*, Okinawa, Japan.
3. Boneh D, Franklin M. Identity based encryption from the Weil pairing. In *Proceedings of Advances in Cryptology-Crypto'01, LNCS 2139*. 2001; 213–229.
4. Smart NP. An identity based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters* 2002; **38**: 630–632,
5. Shim K. Efficient ID-based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters* 2003; **39**: 653–654.
6. Chen L, Kudla C. Identity based authenticated key agreement from pairings. In *IEEE Computer Security Foundations Workshop*. 2003; 219–233.
7. McCullagh N, Barreto PSLM. A new two-party identitybased authenticated key agreement. In *Topics in Cryptology-CT-RSA, LNCS 3376*. 2005; 262–274.
8. Chen L, Cheng Z, Smart NP. Identity-based key agreement protocols from pairings. *International Journal Information Security* 2007; **6**: 213–241.
9. Guo F, Mu Y, Chen Z. Identity-based encryption: how to decrypt multiple ciphertexts using a single decryption key. In *Pairing, LNCS 4575*. 2007; 392–406.
10. Guo F, Mu Y, Chen Z, Xu L. Mluti-identity single-key decryption without random oracles. In *Inscrypt, LNCS 4990*. 2007; 384–398.
11. Chen L, Cheng Z. Security proof of Sakai-Kasahara's identity-based encryption scheme. *10th IMA International Conference on Cryptography and Coding, LNCS 3796*. 2005; 442–459.
12. Bellare M, Rogaway P. Entity authentication and key distribution. In *Advances in Cryptology-Crypto'93, LNCS 773*. 1993; 232–249.
13. Cheng Z, Nistazakis M, Comley R, Vasiu L. On the indistinguishability-based security model of key agreement protocols-simple cases. In *Proceedings of ACNS 2004*, June 2004.
14. Cheng Z, Chen L. On security proof of McCullagh–Barreto's key agreemnet protocol and its variants. *International Journal of Security and Networks* 2007; **2**(3/4): 251–259.
15. Boyen X. General *ad hoc* encryption from exponent inversion IBE. In *Advances in Cryptology-Eurocrypt, LNCS 4515*. 2007; 394–411.

AUTHORS' BIOGRAPHIES

Hua Guo received her B.S. degree in Computer Science from Information Engineer University, China, in 2003. She received the M.S. degrees in Computer Science from Zhengzhou University, China, in 2006. She is currently a PhD candidate of Beihang University in China. Her research interest includes cryptography and information security.

Yi Mu received his Ph.D. from the Australian National University in 1994. He currently is an associate professor in School of Computer Science and Software Engineering and the director of Centre for Computer and Information Security Research, University of Wollongong. Prior to joining University of Wollongong, he was a lecturer in the School of Computing and IT, University of Western Sydney, and a senior lecturer in the Department of Computing, Macquarie University. His current research interest includes network security, computer security, and cryptography. He is the editor-in-chief of International Journal of Applied Cryptography and serves as editor for nine other international journals. He is a senior member of the IEEE and a member of the IACR.



Xiyong Zhang received his B.S., M.S., and Ph.D. degree in Mathematics from Zhengzhou Information Science and Technology Institute, China, in 1997, 2000, and 2003, respectively. He is currently an associate professor of Zhengzhou Information Science and Technology Institute. His research area includes cryptography.

Zhoujun Li received his B.S. degree in Computer Science from Wuhan University, China, in 1984. He received the M.S. and Ph.D. degrees in Computer Science from National University of Defense Technology, China, in 1986 and 1999, respectively. He is currently a professor of Beihang University. His research interest includes information security.