An Identity-Based Group Key Agreement Protocol from Pairing*

Hongji Wang[1,3] Gang Yao[2] Qingshan Jiang[1]
[1]Software School, Xiamen University, Xiamen, Fujian province 361005, P.R.China
[2]State Key Laboratory of Information Security,
Graduate University of Chinese Academy of Sciences, Beijing 100049, P.R.China
[3]Laboratory of Computer Science, Institute os Software,
the Chinese Academy of Sciences, Beijing 100080, P.R.China
whj@xmu.edu.cn gyao@is.ac.cn qjiang@xmu.edu.cn

Abstract

In recent years, a large number of identity-based key agreement protocols from pairings have been proposed. Some of them are elegant and practical. However, using most of those protocols, the session key which users arrive is depended on the identity information.

In this paper, we first propose a two-party identity-based key agreement protocol based on pairings whose security relies on the hardness of the bilinear Diffie-Hellman problem in the random oracle model, and the session key is independent of the identity information. After that we extend the protocol to an identity-based group key agreement protocol, which uses no broadcasting. The proposed key agreement protocol achieves the security attributes: known session key security, perfect forward secrecy, no key compromise impersonation, no unknown key share and no key control.

1. Introduction

It is not secure to transfer a message over the channel directly because an adversary might control the channel. How to communicate securely over an insecure channel is a fundament problem in cryptography. There are two common methods for the persons to encrypt and authenticate their messages in order to protect the privacy and authenticity of these messages. One method is using public-key encryption and signatures, but the cost associated with these primitives may be too high for certain applications. The other is by means of a key agreement protocol, in which users establish a session secret.

Key agreement is one of the fundamental cryptographic primitive. Such protocols allow two or more participants, who each has a long-term key, exchange information over an open network with each other. A secure key agreement protocol guarantees that the adversary does not succeed, and serves as basic building block for constructing secure, complex, higher-level protocols.

Identity-based key agreement protocol is a very important type of key agreement protocol with many potential applications. Since Shamir [11] first proposed identity-based encryption and signature scheme, many identity-based cryptosystem protocols have been proposed. Although there were many practical solutions proposed for identity-based cryptosystem protocols, the first practical identity-based encryption scheme was due to Boneh and Franklin [3] in 2001. Since then, many identity-based key agreement protocols such as [4] have been proposed.

In this paper, we first propose a novel identity-based authenticated key agreement protocol, and then extend the protocol to identity-based group key agreement protocol. The proposed protocols are secure, provided that the BDH assumption holds and the hash functions are modelled as random oracles. The proposed key agreement protocols also achieve the security attributes.

2. Related Work

The concept of identity-based cryptography, in which a public key is the identity (an arbitrary string) of a user, and the corresponding private key is created by binding the identity string with a master secret of a trusted authority (called Key Generation Center), was formulated by Shamir in 1984 [11]. In [11], Shamir proposed the first identity-based key construction based on the RSA problem, and presented an identity-based signature scheme. By using varieties of the Shamir key construction, a number of identity-based key

*This work is supported by the National Nature Science Foundation of China under Grant No. 10771176, the National 985 Project of China under Grant No. 0000-X07204, the National Basic Research Program of China (973 Program) under Grant No 2007CB311202.

0-7695-3102-4/08 $25.00 © 2008 IEEE
DOI 10.1109/ARES.2008.65

agreement schemes such as [8, 14] were proposed .

In 2001, Boneh and Franklin proposed the first formally proved identity-based encryption scheme from pairings. After this scheme proposed, many identity-based cryptographic protocols were developed based on pairings.

Sakai, Ohgishi and Kasahara proposed an identity-based key construction from pairings in [9]. Then, many identity-based key agreement protocols from pairings have recently been published, for example [10, 13, 5, 12, 7].

Till now, many two-party identity-based key agreement protocols have been proposed. For most of them the session key that the users constructed at the end of protocol is depended on identity information of the users. Thus, it is difficult to construct group key using these identity-based key agreement protocol. In order to construct identity-based group key agreement protocol, we propose an identity-based authenticated key agreement protocol, which the session key is independent of the identity information.

3. Preliminaries

Here, we briefly recall some basic definitions.

3.1. Pairing

Let G_1 and G_2 be two cyclic groups of order q for some large prime q. We assume that the discrete logarithm problems in both G_1 and G_2 are hard.

A *pairing* is a bilinear map $e : G_1 \times G_1 \rightarrow G_2$ which satisfies the following conditions:

1. Bilinear: For $\forall(P_1, P_2) \in G_1 \times G_1$ and $\forall(a, b) \in Z_q \times Z_q$, we have $e(aP_1, bP_2) = e(P_1, P_2)^{ab}$.
2. Non-degenerate: There exist non-trivial points $P_1, P_2 \in G_1$ such that $e(P_1, P_2) \neq 1$.
3. Computable: For $\forall(P_1, P_2) \in G_1 \times G_1$, $e(P_1, P_2)$ is efficiently computable.

The *Bilinear Diffie-Hellman* (BDH) Problem for a bilinear pairing $e : G_1 \times G_1 \rightarrow G_2$ is defined as follows: Let a, b, c be the random number chosen from Z_q^*, given (P, aP, bP, cP), compute $e(P, P)^{abc}$. The *Diffie-Hellman* (DH) Problem for group G_1 is defined as follows: Let a, b be the random number chosen from Z_q^*, given (P, aP, bP), compute abP.

We assume that both the BDH problem and the DH problem are hard.

3.2. Security model

We shall use a modified Bellare-Rogaway key exchange model [2] to analyze the protocol security. An oracle $\Pi_{i,j}^s$ denotes the s-th instance of party i involved with a partner party j in a session.

The security of a protocol is defined by a game with two phases. In the first phase, an adversary E is allowed to issue the following queries in any order.

$Send(\Pi_{i,j}^s, x)$. Upon receiving the message x, oracle $\Pi_{i,j}^s$ executes the protocol and responds with an outgoing message m or a decision to indicate accepting or rejecting the session. If the oracle $\Pi_{i,j}^s$ does not exist, it will be created as initiator if $x = \lambda$, or as a responder otherwise.

$Reveal(\Pi_{i,j}^s)$. If the oracle has not accepted, it returns $\perp$; otherwise, it reveals the session key.

Once the adversary decides that the first phase is over, it starts the second phase by choosing a fresh oracle $\Pi_{i,j}^s$ and issuing a $Test(\Pi_{i,j}^s)$ query.

$Test(\Pi_{i,j}^s)$. Oracle $\Pi_{i,j}^s$, which is fresh as a challenger, randomly chooses $b \in \{0, 1\}$, and responds with the session key if $b = 0$, or a random sample from the distribution of the session key otherwise.

After this point the adversary can continue querying the oracles except that it cannot reveal the test oracle $\Pi_{i,j}^s$ or its partner $\Pi_{j,i}^t$ (if it exists). Finally the adversary outputs a guess b' for b. If $b' = b$, we say that the adversary wins. The adversary's advantage is defined as

$$Adv_E(k) = \max\{0, \Pr[E \text{ wins}] - 1/2\}.$$

Protocol Π is a *secure key agreement protocol*, if (1) In the presence of a benign adversary, which faithfully conveys messages, on $\Pi_{i,j}^s$ and $\Pi_{j,i}^t$, both oracles always accept holding the same session key, and this key is distributed uniformly on $\{0, 1\}^k$; (2) For any polynomial time adversary E, $Adv_E(k)$ is negligible.

3.3. Hess's identity-based signature

Let PKG be the private key generator. Hess's identity-based signature goes as follows [6]:

Setup: The PKG chooses a random number $s \in Z_q^*$, and computes the global public key $R = sP$. The PKG also selects a Map-to-point hash function $H_0 : \{0, 1\}^* \rightarrow G_1$ and another cryptographic hash function $H : \{0, 1\}^* \rightarrow Z_q^*$. Then the PKG publishes $\{q, G_1, G_2, e, P, R, H_0, H\}$ and keeps s as the master key.

Extract: Given the public identity information ID, compute $Q = H_1(ID)$ as the role of the public key, and then compute the corresponding secret key as $S = sQ$.

Sign: To sign a message $m \in \{0, 1\}^*$ using the secret key S, the signer chooses an arbitrary $P_1 \in G_1$, picks a random integer $k \in Z_q^*$, and computes $r = e(P_1, P)^k$, $v = H(m, r)$, and $U = vS + kP_1$. The signature σ is $< U, v >$.

Verify: To verify the signature $\sigma = < U, v >$ of an identity ID on a message m, the verifier computes $r =$

$e(U,P) \cdot e(Q,-R)^v$. He accept the signature if and only if $v = H(m,r)$.

3.4. 2^d-cube group key agreement protocol

Becker and Willie have formally described the 2^d-cube group key agreement protocol with d rounds in [1]. The protocol uses no broadcasting. In the protocol, the 2^d participants are identified with the vectors in the d-dimensional vector space $GF(2)^d$ and a basis $\{\vec{b}_1, \ldots, \vec{b}_d\}$ of $GF(2)^d$ is chosen. The protocol may be performed as follows:

In the first round, every participant $\vec{z} \in GF(2)^d$ generates a random number $r_{\vec{z}}$ and performs a Diffie-Hellman key exchange with participant $\vec{z}+\vec{b}_1$ using the values $r_{\vec{z}}$ and $r_{\vec{v}+\vec{b}_1}$.

In the i-th round, every participant $\vec{z} \in GF(2)^d$ performs a Diffie-Hellman key exchange with participant $\vec{z}+\vec{b}_i$, where both parties use the value generated in round $i-1$ as the secret value for the key exchange.

4. Two-party ID-based key agreement protocol

4.1. Details of the the protocol

Let U_1 and U_2 be two participants, and PKG be the private key generator. Let ID_i be the identity of U_i.

Let G_1 and G_2 be two cyclic groups of order q for some large prime q. G_1 is a cyclic additive group and G_2 is a cyclic multiplicative group. Let P be an arbitrary generator of G_1, and $e : G_1 \times G_1 \rightarrow G_2$ be a cryptographic pairing.

Set up:

The PKG chooses a random number $s \in Z_q^*$ and set $R = sP$. The PKG also chooses $H_0 : \{0,1\}^* \rightarrow G_1^*$ be a Map-to-Point hash function, and H be a cryptographic hash function. Then the PKG publishes system parameters $\{q, G_1, G_2, e, P, R, H_0, H\}$, and keeps s as its master key.

Extract:

Given a public identity $ID \in \{0,1\}^*$, the PKG computes the public key $Q = H_0(ID) \in G_1$ and generates the associated private key $S = sQ$. The PKG passes S as the private key to the user via some secure channel.

Key agreement:

Let two users U_1 and U_2 with respective public keys $Q_1 = H_0(ID_1)$ and $Q_2 = H_0(ID_2)$ decide to agree upon a common secret key. $S_1 = sQ_1$ is the long term secret key of U_1 sent by the PKG on submitting U_1's public identity. $S_2 = sQ_2$ is the long term secret key of U_2 sent by the PKG on submitting U_2's public identity. They perform the following operations.

1. User U_1 randomly chooses a number $r_1 \in Z_q^*$, computes $E_1 = r_1P$, $F_1 = H(U_1, U_2, E_1, e(E_1, R))S_1 + r_1R$, and sends E_1 and F_1 to U_2.
2. User U_2 randomly chooses a number $r_2 \in Z_q^*$, computes $E_2 = r_2P$, $F_2 = H(U_2, U_1, E_2, e(E_2, R))S_2 + r_2R$, and sends E_2 and F_2 to U_1.
3. User U_1 verifies whether $e(F_2, P) = e(H(U_2, U_1, E_2, e(E_2, R))Q_2 + E_2, R)$ holds or not. If the verification succeed, U_1 computes the session key $K = H(U_1, U_2, r_1E_2, e(E_2, R)^{r_1})$.
4. User U_2 verifies whether $e(F_1, P) = e(H(U_1, U_2, E_1, e(E_1, R))Q_1 + E_1, R)$ holds or not. If the verification succeed, U_2 computes the session key $K = H(U_1, U_2, r_2E_1, e(E_1, R)^{r_2})$.

When the protocol is finished, the user U_1 and U_2 share the same session key K.

4.2. Security attributes

Theorem 1 *In our protocol, if an adversary do not have the private key of a user, he cannot impersonate the user to carry out the key agreement protocol with the others successfully, provided that the Hess's identity-based signature is secure.*

Sketch: Suppose that an adversary B do not have the private key of a user U_1, but he can impersonate the user U_1 to carry out the key agreement protocol with the others successfully. We can get that the adversary B has generated a message E_1 and F_1, such that E_1 and F_1 can pass the verification by the other user. That is to say, E_1 and F_1 satisfies the equation $e(F_1, P) = e(H(U_1, U_2, E_1, e(E_1, R))Q_1 + E_1, R)$.

Using E_1 and F_1, we can compute the Hess's identity-based signature on message $m = (U_1, U_2, E_1)$ as follows: Compute $v = H(U_1, U_2, E_1, e(E_1, R))$, and set $U = F_1$. Then, the Hess's identity-based signature on message m is $\sigma = < U, v >$.

To verify the signature $\sigma = < U, v >$ on a message m, we compute $r = e(U, P) \cdot e(Q_1, -R)^v = e(r_1P, R)$. Then $H(m, r) = H(U_1, U_2, E_1, e(r_1P, R)) = v$. That is, $< U, v >$ is a signature on message m.

Since Hess's identity-based signature a secure scheme (Theorem 1 in [6]), we can draw the conclusion that the adversary B cannot impersonate the user U_1 to carry out the key agreement protocol with the others successfully. □

Theorem 2 *The proposed protocol is a secure key agreement, provided that the BDH assumption holds and the hash functions are modelled as random oracles. Specifically, suppose that in the attack, an adversary B which makes p queries to H and creates p' oracles, wins the game with advantage $\epsilon(k)$. Then there exists an algorithm A to solve the BDH problem with advantage*

$$Adv_A^{BDH}(k) \geq \frac{1}{2 \cdot p \cdot p'} \epsilon(k).$$

Sketch: We define the session ID as the concatenation of $xP||yP$. The first condition in definition of secure key agreement protocol is trivial. Now we prove that the protocol meets the second condition.

Given a BDH problem instance (P, aP, bP, cP), we construct an algorithm A using the adversary B against the protocol to solve the BDH problem.

A simulates the system setup to adversary B as follows. The system public parameters are defined to be the pairing parameters of the input problem. The master public key is set to be $R = aP$. Hence A does not know the master secret key. The functions H are instantiated as random oracles under the control of A.

Algorithm A starts simulating the real world where the adversary B launches an attack. Algorithm A maintains three lists: H^{list}, Ω, and Γ. Algorithm A answers the following queries, which are asked by the adversary B in an arbitrary order.

H query, $H(ID_u^a, ID_u^b, X_u, Y_u, Z_u, k_u)$**:** Algorithm A maintains an initially empty list H^{list} with entries of the form $(ID_u^a, ID_u^b, X_u, Y_u, Z_u, k_u, h_u)$. The algorithm A responds to the query in the following way:

If a tuple indexed by $(ID_u^a, ID_u^b, X_u, Y_u, Z_u, k_u)$ is on the list, then A responds with h_u.

Otherwise, A goes through the list Γ (maintained in the Reveal query) to find a tuple with values $(ID_u^a, ID_u^b, X_u, Y_u, \Pi_{i,j}^t)$ and proceeds as follows (without losing generality, we assume that Y_u is the message generated by oracle $\Pi_{i,j}^t$, so X_u is the incoming message to $\Pi_{i,j}^t$.):

- Test if $e(Y_u, X_u) = e(P, Z_u)$ holds. If the equation holds, A then

  Computes the shared secret via the following equation $k_{i,j}^t = e(Z_u, R) = e(Z_u, aP)$. Note that $\Pi_{i,j}^t$ is put on the list Γ in the Reveal query only when $\Pi_{i,j}^t$, but $H(ID_u^a, ID_u^b, Z_u, k_{i,j}^t)$ had not been queried before the reveal query. So $K_{i,j}^t$ has been randomly sampled.

  Set $h_u = K_{i,j}^t$.

  Remove $(ID_u^a, ID_u^b, X_u, Y_u, \Pi_{i,j}^t)$ from the list Γ. Put $(ID_u^a, ID_u^b, X_u, Y_u, Z_u, k_{i,j}^t, h_u)$ in the list H^{list}.

  Check if $k_{i,j}^t = k_u$. If it is not true, A then randomly chooses new $h_u \in \{0,1\}^n$ and inserts $(ID_u^a, ID_u^b, X_u, Y_u, Z_u, k_u, h_u)$ into the list H^{list}.

  Return h_u.
- Otherwise (no tuple on Γ meets the test), algorithm A randomly chooses $h_u \in \{0,1\}^n$, inserts $(ID_u^a, ID_u^b, X_u, Y_u, Z_u, k_u, h_u)$ into the list, and returns h_u.

Otherwise (no relative tuple on Γ is found), A randomly chooses $h_u \in \{0,1\}^n$, inserts $(ID_u^a, ID_u^b, X_u, Y_u, Z_u, k_u, h_u)$ into the list, and returns h_u.

Send query, $Send(\Pi_{i,j}^t, M)$**:** A maintains a list Ω for each oracle of the form $(\Pi_{i,j}^t, tran_{i,j}^t, f_{i,j}^t, k_{i,j}^t, K_{i,j}^t, x_{i,j}^t)$, where $tran_{i,j}^t$ is the transcript of the oracle so far, $f_{i,j}^t$ and $x_{i,j}^t$ are used for special purpose explained below, and $k_{i,j}^t$ and $K_{i,j}^t$ are set $\perp$ initially. Note that this list can be updated in other queries as well. A proceeds in the following way:

If M is the second message on the transcript, do nothing but simply accept the session.

Otherwise, M is not the second message,

- Randomly sample $f_{i,j}^t \in Z_q^*$.
- Randomly flip $x_{i,j}^t \in \{0,1\}$. If $x_{i,j}^t = 0$, set $V = f_{i,j}^t bP$, else $V = f_{i,j}^t cP$.
- Return V.

Reveal query, $Reveal(\Pi_{i,j}^t)$**:** A maintains a list Γ with tuples of the form $(ID_i, ID_j, X_i, Y_j, \Pi_{i,j}^t)$. A proceeds in the following way to respond:

Get the tuple of oracle $\Pi_{i,j}^t$ from Ω.

If oracle $\Pi_{i,j}^t$ has not accepted, then responds with $\perp$.

If the $Test(\Pi_{a,b}^w)$ query has been issued and if $\Pi_{i,j}^t = \Pi_{a,b}^w$, or $ID_a = ID_j$ and $ID_b = ID_j$ and two oracles have the same transcripts, then disallow the query (this should not happen).

If $K_{i,j}^t \neq \perp$, return $K_{i,j}^t$.

Otherwise, let M be the received message on $tran_{i,j}^t$.

- Go through the list H^{list} to find a tuple $(ID_i, ID_j, M_u, M, Z_u, k_u, h_u)$ if ID_i is the initiator, or a tuple $(ID_j, ID_i, M, M_u, Z_u, k_u, h_u)$ otherwise, meeting the equation $e(M_u, M) = e(P, Z_u)$.
- If such Z_u is found, then compute $k_{i,j}^t = e(Z_u, R)$ and set $K_{i,j}^t = H(ID_i, ID_j, Z_u, k_{i,j}^t)$ if oracle $\Pi_{i,j}^t$ is the initiator, or set $K_{i,j}^t = H(ID_j, ID_i, Z_u, k_{i,j}^t)$ otherwise.
- Otherwise, randomly sample $K_{i,j}^t \in \{0,1\}^n$, put $(ID_i, ID_j, M_u, M, \Pi_{i,j}^t)$ if ID_i is the initiator or $(ID_j, ID_i, M, M_u, \Pi_{i,j}^t)$ into list Γ.
- A responds with $K_{i,j}^t$ and updates Ω by putting $K_{i,j}^t$.

Test query, $Test(\Pi_{i,j}^t)$**:** By the rule of the game, there is a partner oracle $\Pi_{j,i}^w$ with the same transcript as $\Pi_{i,j}^t$, and both should not be revealed. A proceeds as follows:

Check if $x_{i,j}^t = x_{j,i}^w$. If it is true, then abort the game (event Δ_1).

Otherwise, without losing generality, we assume $x_{i,j}^t = 0$ and $x_{j,i}^w$ = 1, i.e., $M_i = f_{i,j}^t bP$ and $M_j = f_{j,i}^w cP$. A then randomly chooses $\zeta \in \{0,1\}^n$ and responds to B with ζ.

Once B finishes queries and returns its guess, A proceeds with the following steps:

- Compute $v = (f^t_{i,j} \cdot f^w_{j,i})^{-1}$. In $\Pi^t_{i,j}$, it has $k^t_{i,j} = e(f^w_{j,i}c \cdot f^t_{i,j}bP, aP) = (e(P,P)^{abc})^{1/v}$.
- Algorithm A randomly chooses $\tilde{k}$ from H^{list} and returns $\tilde{k}^v$ as the response to the BDH challenge.

Since the simulations of all the random oracles are valid and the messages of the oracles are uniformly distributed in the message space, the adversary should not notice any difference from the real attack environment.

Let Δ_2 be that $k = e(cbP, aP)^{1/v}$ was not queried on H. Because H is a random oracle and both oracle $\Pi^w_{j,i}$ and oracle $\Pi^t_{i,j}$ are not revealed, if Δ_2 happens, B could win the game only in the ways that B random guesses whether ζ is $K^t_{i,j}$ or not. Then we have $\Pr[B \text{ wins}|\Delta_2] \leq 1/2$. Then $\epsilon(k) + 1/2 = \Pr[B \text{ wins}] = \Pr[B \text{ wins}|\Delta_2]\Pr[\Delta_2] + \Pr[B \text{ wins}|\overline{\Delta_2}]\Pr[\overline{\Delta_2}] \leq 1/2 + \Pr[\overline{\Delta_2}]$. Hence, $\Pr[\overline{\Delta_2}] \geq \epsilon(k)$.

Let Δ_3 be that, in the attack, adversary B indeed chose $\Pi^t_{i,j}$ as the challenge oracle and the game will not abort. So $\Pr[\Delta_3] \geq 1/(2 \cdot p')$.

Let Δ_4 be that A found the correct $\tilde{k}$. Overall, we have $\Pr[A \text{ wins}] = \Pr[\Delta_3 \wedge \overline{\Delta_2} \wedge \Delta_4] \geq \Pr[\overline{\Delta_2}]/(2 \cdot p \cdot p') \geq \epsilon(k)/(2 \cdot p \cdot p')$. This concludes the proof. □

Below is security attributes of our protocol.

Known session key security

As with our protocol, suppose that the adversary learned the session keys of a previous key agreement protocol. To extract the ephemeral key from a session key, for example, to determine a key from $K = H(U_1, U_2, r_2E_1, e(E_1, R)^{r_2})$, is equivalent to solving the BDH problem. Since the process of computing new session key do not use the information used before, and the parameters are independent, knowing the previous session key is useless for computing the current session key $H(U_1, U_2, r_2E_1, e(E_1, R)^{r_2})$.

Perfect forward secrecy

As with our protocol, to learn the previous session keys, the adversary has to get the corresponding ephemeral keys. Suppose that the adversary has got U_1's private key S_1. From U_1's messages (E_1, F_1), he can computes the random number r_1, which is equivalent to solve the discrete logarithm problem in G_1. Therefore, our protocol is Perfect forward secrecy.

Even the adversary has to get the master keys, he cannot learn the previous session keys. Similar to the proof of Theorem 2, we can get the conclusion:

The proposed protocol has master key forward secrecy, provided that the DH assumption is sound and H is modelled as random oracle. Specifically, suppose that an adversary B wins the game with advantage $\epsilon(k)$.

No key-compromise impersonation

The compromise of one entity's private key does not imply that the private keys of other entities will also be compromised in our protocol. The adversary may impersonate the compromised entity in subsequent protocols, but he cannot impersonate other entities.

No unknown key-share

To implement such an attack on our protocol, the adversary is required to learn the private key of some entity. Otherwise, the attack hardly works. Hence, we claim that our protocol has the attribute of no unknown key-share.

No key control

The session keys in our protocol are determined by the two entities, and no one can influence the outcome of the session keys, or enforce them to fall into a pre-determined interval. In other words, there is no key control in our protocol.

4.3. Efficiency attributes

The proposed protocol is a one-round protocol, and it is clear that each user can only send two elements in group G_1 when he performs the protocol.

When a person performs the proposed protocol to construct a session key with another user, the total computation of each user is five scalar multiplications in group G_1, five pairing computations, one map-to-point hash operation (H_0), and three hash function (H) evaluations.

5. ID-based group key agreement protocol

We extend the protocol in preceding section to the ID-based group key agreement protocol.

5.1. Details of the the protocol

Let $U_1, \ldots, U_n$ be n participants, and PKG be the private key generator. Let ID_i be the identity of U_i. Let d be the least integer greater than $\log_2 n$, that is $d-1 < \log_2 n \leq d$. Let $U_{n+1}, \ldots, U_{2^d}$ denote the virtual users in the protocol. The protocol may randomly choose $2^d - n$ users in the set $\{U_1, \ldots, U_n\}$ to substitute for $U_{n+1}, \ldots, U_{2^d}$.

Let $\vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{2^d}$ be all vectors in the d-dimensional vector space $GF(2)^d$ and a basis $\vec{b}_1, \ldots, \vec{b}_d$ of $GF(2)^d$ is chosen. Suppose that G_1 and G_2 are two cyclic groups of order q for some large prime q. G_1 is a cyclic additive group and G_2 is a cyclic multiplicative group. Let P be an arbitrary generator of G_1, and $e : G_1 \times G_1 \rightarrow G_2$ be a cryptographic pairing.

Set up:

The PKG chooses a random number $s \in Z^*_q$ and set $R = sP$. The PKG also chooses $H_0 : \{0,1\}^* \rightarrow G^*_1$ be a Map-to-Point hash function, and H be a cryptographic hash function. Then the PKG publishes system parameters $\{q, G_1, G_2, e, P, R, H_0, H\}$, and keeps s as its master key.

Extract:

Given a public identity $ID \in \{0,1\}^*$, the PKG computes the public key $Q = H_0(ID) \in G_1$ and generates the corresponding private key $S = sQ$. The PKG passes S as the private key to the user via some secure channel.

Key agreement:

Let 2^d users $U_1, \ldots, U_{2^d}$ with respective public keys $Q_i = H_0(ID_i)$ $(1 \leq i \leq 2^d)$ decide to agree upon a common secret key. $S_i = sQ_i$ is the long term secret key of U_i sent by the PKG on submitting U_i's public identity $(1 \leq i \leq 2^d)$. Let U denote $U_1||\ldots||U_{2^d}$. The protocol may be performed in d rounds as follows:

In the first round: Every participant $U_{\vec{z}}$ (where $\vec{z} \in GF(2)^d$) generates a random number $r_{\vec{z}} \in Z_q^*$, and computes $E_{\vec{z}}^{(1)} = r_{\vec{z}}P$, $F_{\vec{z}}^{(1)} = H(U, E_{\vec{z}}^{(1)}, e(E_{\vec{z}}^{(1)}, R))S_{\vec{z}} + r_{\vec{z}}R$, and sends $E_{\vec{z}}^{(1)}$ and $F_{\vec{z}}^{(1)}$ to $U_{\vec{z}+\vec{b}_1}$. User $U_{\vec{z}+\vec{b}_1}$ receives the message from $U_{\vec{z}}$, verifies whether $e(F_{\vec{z}}^{(1)}, P) = e(H(U, E_{\vec{z}}^{(1)}, e(E_{\vec{z}}^{(1)}, R))Q_{\vec{z}} + E_{\vec{z}}^{(1)}, R)$ holds or not. If the verification succeed, $U_{\vec{z}+\vec{b}_1}$ computes the session key $K_{\vec{z}+\vec{b}_1}^{(1)} = H(U, r_{\vec{z}+\vec{b}_1}E_{\vec{z}}^{(1)}, e(r_{\vec{z}+\vec{b}_1}E_{\vec{z}}^{(1)}, R))$.

In the i-th round: Every participant $U_{\vec{z}}$ computes $E_{\vec{z}}^{(i)} = K_{\vec{z}}^{(i-1)}P$, $F_{\vec{z}}^{(i)} = H(U, E_{\vec{z}}^{(i)}, e(E_{\vec{z}}^{(i)}, R))S_{\vec{z}} + K_{\vec{z}}^{(i-1)}R$, and sends $E_{\vec{z}}^{(i)}$ and $F_{\vec{z}}^{(i)}$ to $U_{\vec{z}+\vec{b}_i}$. User $U_{\vec{z}+\vec{b}_i}$ receives the message from $U_{\vec{z}}$, verifies whether $e(F_{\vec{z}}^{(i)}, P) = e(H(U, E_{\vec{z}}^{(i)}, e(E_{\vec{z}}^{(i)}, R))Q_{\vec{z}} + E_{\vec{z}}^{(i)}, R)$ holds or not. If the verification succeed, $U_{\vec{z}+\vec{b}_i}$ computes the session key $K_{\vec{z}+\vec{b}_1}^{(i)} = H(U, K_{\vec{z}+\vec{b}_1}^{(i-1)}E_{\vec{z}}^{(i)}, e(K_{\vec{z}+\vec{b}_1}^{(i-1)}E_{\vec{z}}^{(i)}, R))$.

In every round i, $1 \leq i \leq d$, the participants communicate on a maximum number of parallel edges of the d-dimensional cube in the direction $\vec{b}_i$, Thus every party is involved in exactly one two-party ID-based key agreement protocol per round. Furthermore, all the users $U_1, \ldots, U_{2^d}$ share a common key at the end of this protocol.

5.2. Security attributes and efficiency attributes

It is easy to draw the following conclusion:

In our group key agreement protocol, if an adversary do not have the private key of a user, he cannot impersonate the user to carry out the key agreement protocol with the others successfully, provided that the Hess's identity-based signature is secure.

The proposed group key agreement protocol is a secure key agreement, provided that the BDH assumption holds and the hash functions are modelled as random oracles.

Furthermore, the proposed group key agreement protocol also achieves the security attributes: known session key security, perfect forward secrecy, no key compromise impersonation, no unknown key share and no key control.

The proposed protocol is a d-round protocol. The total communication of each user is d messages, and each message consists of two elements in group G_1.

Suppose that a person performs the proposed group key agreement protocol to construct a session key. The total computation of each user is $5d$ scalar multiplications in group G_1, $5d$ pairing computations, d map-to-point hash operation (H_0), and $3d$ hash function (H) evaluations.

Acknowledgements The authors would like to thanks the anonymous referees for their helpful comments.

References

[1] K. Becker and U. Wille. "Communication complexity of group key distribution". *Proceedings of CCS'98*, ACM Press, pp. 1-6, 1998.

[2] M. Bellare and P. Rogaway. "Entity authentication and key distribution". *Crypto'93*, LNCS 773, Springer-Verlag, pp. 232-249, 1993.

[3] D. Boneh and M. Franklin. "Identity-based Encryption from the Weil pairing". *Crypto 2001*. LNCS 2139, Springer-Verlag, pp. 213-229, 2001.

[4] L. Chen, Z. Cheng and N. P. Smart. "Identity-based Key Agreement Protocols From Pairings". *Cryptology ePrint Archive*, Report 2006/199.

[5] L. Chen and C. Kudla. "Identity based authenticated key agreement from pairings". *IEEE Computer Security Foundations Workshop*, IEEE Press, pp. 219-233, 2003.

[6] F. Hess. "Efficient Identity Based Signature Schemes Based on Pairings". *SAC 2002*, LNCS 2595, Springer-Verlag, pp. 310-324, 2003.

[7] N. McCullagh and P. S. L. M. Barreto. "A new two-party identity-based authenticated key agreement". *CT-RSA 2005*, LNCS 3376, Springer-Verlag, pp. 262-274, 2005.

[8] E. Okamoto. "Proposal for identity-based key distribution system". *Electronics Letters*, vol. 22, pp. 1283-1284, 1986.

[9] R. Sakai, K. Ohgishi and M. Kasahara. "Cryptosystems based on pairing". *Symposium on Cryptography and Information Security*, Okinawa, Japan, 2000.

[10] M. Scott. "Authenticated ID-based key exchange and remote login with insecure token and PIN number". *Cryptology ePrint Archive*, Report 2002/164.

[11] A. Shamir. "Identity-based cryptosystems and signature schemes". *Crypto'84*. LNCS 196, Springer-Verlag, pp. 47-53, 1984.

[12] K. Shim. "Efficient ID-based authenticated key agreement protocol based on the weil Pairing". *Electronic Letters*, vol. 39, pp. 653-654, 2003.

[13] N. P. Smart. "An identity-based authenticated key agreement protocol based on the weil pairing". *Cryptology ePrint Archive*, Report 2001/111.

[14] K. Tanaka and E. Okamoto. "Key distribution system for mail systems using ID-related information directory". *Computers & Security*, vol. 10, pp. 25-33, 1991.