Key Agreement Protocol Based on Weil Pairing

Sheng-Hua Shiau Ren-Junn Hwang Ming-Fuu Lin
Department of Computer Science and Information Engineering, Tamkang University,
Tamsui, Taipei, Taiwan 251, R.O.C.
E-mail: aaron@mail.mine.tku.edu.tw, victor@mail.tku.edu.tw, u6342156@tknet.tku.edu.tw

Abstract

We propose a group key agreement protocol in this paper. The key agreement protocol is a good solution to establish a common session key for communication. But in a group of member's communication, we not only need to establish a common session key, but also need to concern the member changing environment. The proposed protocol is based on weil pairing, ID-based authentication and a complete binary tree architecture. The users in the group will establish a common session key. If there are users want to join or leave the group, our protocol can reconstruct a new common session key for security considerations. Furthermore, our proposed protocol is efficiency when the group member is small and dynamic changing.

1. Introduction

It is important to assure security in the group communication environment. A secure group communication should provide communicate confidentially among users in the group, that is, the messages during communication should not be known by users outside the group and the users in the group can join or leave dynamically during the communication. It needs a session key to encrypt the transmitted messages. There are two technologies to generate a session key for confidence: the key distribution and the key agreement. In key distribution, it needs a group controller to hold the information of entire users in the group, if the group controller is crashed or attacked, then the group break down. While the group member is dynamic changing, the group controller may be inefficiency in this environment.

In contrast, key agreement does not need the group controller; all users in the group generate the session key by key agreement. The session key includes information of all users so that no user can control or forecast it.

Diffie-Hellman key agreement [3] is the first key agreement protocol. It can assure the security of communication between the two users. But it does not authenticate users, hence suffers the "man-in-the-middle" attack.

Joux [4] gave another direction of key agreement. He implements a tripartite key agreement protocol using weil pairing. When three users want to agree a common session key, only one message must be delivered by each user in the protocol. However, Joux's protocol still does not authenticate the users, and is vulnerable to "man-in-the-middle" attack.

With authentication, Shamir [6] proposed an identity-based encryption and signature scheme. It provides authentication without CA. In the scheme, it uses identity information as user's public key, and so that it is not need to verify user's public key. It needs a *KGC* (Key Generation Center) to be responsible to generate user's private key from user's identity. Since then, there are many ID-based encryptsystem have been proposed [1,2,7,8].

In this paper, we propose a group key agreement protocol based on weil pairing. In our protocol, we use the ID-based architecture to authenticate the received messages and the users in the group. If there are some users want to join or leave the group, not all users in the group need to renew their secret key, it is suit for dynamic changing environment.

This paper is organized as followings: Section 2 proposes the notation and assumption in this paper. Section 3 is the proposed protocol. We show the analysis of some security properties that we concerned in section 4. Section 5 describes the comparison of computation overhead with other protocol. Finally,

Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05)
1550-445X/05 $20.00 © 2005 **IEEE**

section 6 concludes the paper.

2. Notation and assumption

Let G_1 be an additive group with a prime order q, and G_2 to be a multiplicative group with the same order. P is an arbitrary generator of G_1.

We assume that the discrete logarithm problem (DLP) is hard in G_1 and G_2. e is a bilinear mapping between two groups (e: $G_1 \times G_1 \rightarrow G_2$). It must satisfy the following properties:

1. Bilinear: for all P, $Q \in G_1$ and a, $b \in Z_q^*$, we have $e(aP, bQ) = e(P, Q)^{ab}$.
2. Non-degenerate: if P is a generator of G_1, then $e(P, P)$ is a generator of G_2.
3. Computable: There is an efficient algorithm to compute $e(P, Q)$ for all P, $Q \in G_1$.

For using bilinear mappings to implement the protocol, there are some problems and assumptions [5] as followings:

1. DDH (Decisional Diffie-Hellman) Problem in G_1:
   Given (P, aP, bP, cP) for some a, b and $c \in Z_q^*$, decides if $c = ab \bmod q$. The DDH problem can be solved in polynomial time by $e(aP, bP) = e(cP, P)$.
   DDH assumption:
   There is no polynomial time algorithm to solve the DDH problem in G_2.
2. HDH (Hash Dicisional Diffie-Hellman) Problem in G_1:
   Given (P, aP, bP, c) and a hash function H_1: $G_1 \rightarrow Z_q^*$, decides if $c = H_1(abP) \bmod q$.
   HDH assumption:
   There is no polynomial time algorithm to solve the HDH problem in G_1.
3. BDH (Bilinear Diffie-Hellman) Problem:
   Given (P, aP, bP, cP), computes $e(P, P)^{abc}$.
   BDH assumption:
   There is no polynomial time algorithm to solve the BDH problem.
4. DHBDH (Decisional Hash Bilinear Diffie-Hellman) Problem:
   Given (P, aP, bP, cP, d) and a hash function H_2: $G_2 \rightarrow Z_q^*$, decides if $d = H_2(e(P, P)^{abc}) \bmod q$.
   DHBDH assumption:
   There is no polynomial time algorithm to solve the DHBDH problem.

3. The proposed protocol

In this section, we propose our new protocol. We divide our protocol into three phases: the initial phase, the key agreement phase and the member changing phase. In order to perform ID-based authentication, each user need to register to the *KGC* (Key Generation Center) in initial phase. Key agreement phase describes how members in the group to agree a common session key. Membership changing phase shows what should be done if members join or leave the group. We need some system parameters in our protocol, we show the definitions in Table 1.

Table 1. The system parameters

G_1	An additive group with prime order q.
G_2	A multiplicative group with the same order q.
P	A generator of G_1.
s_i	The short term private key of users, $1 \le i \le n$.
i	Each user is in the name of integer i, $1 \le i \le n$.
H	A cryptographic hash function, H: $\{0, 1\}^* \rightarrow G_1$.
H_1	A cryptographic hash function, H_1: $G_1 \rightarrow Z_q^*$.
H_2	A cryptographic hash function, H_2: $G_2 \rightarrow Z_q^*$.
H_3	A cryptographic hash function, H_3: $G_1 \times G_1 \rightarrow Z_q^*$.
k_i	The common value of users i, $2i$ (if user i has left child only) or i, $2i$, $2i + 1$ (if user i has two children).
ID_i	The identity of the user i, $ID_i \in \{0, 1\}^*$, $1 \le i \le n$.
KGC	The key generation center, it is responsible for ID-based authentication.
Q_i	The long-term public key of user i, $Qi = H(ID_i)$.
S_i	The long-term private key of user i, $S_i = sQ_i$.
s	It is chosen from Z_q^* by *KGC*. The *KGC* must keeps s as secret and treats it as the master key.
P_{pub}	The public key of *KGC*, $P_{pub} = sP$.

3.1 The initial phase

We show that how each user registers to the *KGC*, and get their private key. They only need to process this phase one time. After that, every member can process the key agreement phase to compute the common session key.

The *KGC* selects a random number s form Z_q^* and computes $P_{pub} = sP$. The *KGC* publishes P_{pub} as a system parameter and keeps s secretly, where s is the master key.

Each user U_i's identity is $ID_i \in \{0, 1\}^*$ and their long-term public key is $Q_i = H(ID_i)$. They use Q_i to register to the *KGC* in secure channel by the following steps:

Step 1: User U_i sends Q_i to *KGC*.

Step 2: *KGC* compute user U_i's long-term private key $S_i = sQ_i$ and sends back to U_i.

The public system parameters are (G_1, G_2, e, q, P, P_{pub}, H, H_1, H_2, H_3).

3.2 The key agreement phase

In this subsection, we show that how legal users cooperate to compute a common session key. In our protocol, the key agreement process is based on complete binary tree architecture. Each nodes in that tree is representing one user, Figure 1 is an example of 15 users.

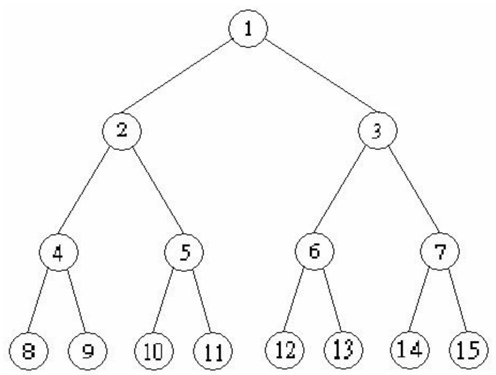

Figure 1. A complete binary tree of a group with 15 users

Assume there are n user in this group, every user U_i ($i \in \{1,..,n\}$) holds their long-term public key Q_i , long-term private key S_i , and they will choose one random number s_i as short-term private key.

There are three kinds of nodes in a complete binary tree: the leaf node, the internal node with one left child only and the internal node with two children.

Case 1: If the node is a leaf ($2i > n$)

Step 1: Set $t_i = s_i$.

Step 2: User U_i broadcasts $t_i{\cdot}P$ to all users in the group.

Case 2: If the node only has one left child ($2i = n$)

Step 1: User U_i selects another random number s_i' additionally.

Step 2: User U_i sends messages (P_i, P_i', T_i) to the user U_{2i}, where $P_i{=}s_i{\cdot}P$, $P_i'{=}s_i'{\cdot}P$ and $T_i{=}H_3(P_i, P_i')\ S_i + s_i{\cdot}P_i'$.

User U_{2i} sends messages (P_{2i}, T_{2i}) to the user U_i, where $P_{2i} = s_{2i}{\cdot}P$ and $T_{2i} = H_1(P_{2i})\ S_{2i} + s_{2i}{\cdot}P_{2i}$.

Step 3: User U_i verifies the following equation:

$$e(T_{2i}, P) = e(P_{2i}, P_{2i})\ e(H_1(P_{2i})Q_{2i}, P_{pub}).$$

User U_{2i} verifies the following equation:

$$e(T_i, P) = e(P_i, P_i)\ e(H_3(P_i, P_i')Q_i, P_{pub}).$$

Step 4: If the equation in step 3 holds, the user U_i computes $k_i = e(s_i'{\cdot}P, P_{2i})^{s_i}$, and the user U_{2i} computes $k_i = e(P_i, P_i')^{s_{2i}}$, where

$$k_i = e(s_i'{\cdot}P, P_{2i})^{s_i} = e(P_i, P_i')^{s_{2i}}$$
$$= e(P, P)^{s_i{\cdot}s_i'{\cdot}s_{2i}}.$$

Step 5: If $i = 1$, then session key is k_i, else set $t_i = H_2(k_i)$ and User U_i broadcasts $t_i{\cdot}P$ to all users in the group.

Case 3: If the node has two children

Step 1: User U_i sends messages (P_i, T_i) to user U_{2i} and U_{2i+1}, where $P_i = s_i{\cdot}P$ and $T_i = H_1(P_i)S_i + s_i{\cdot}P_i$.

User U_{2i} sends messages (P_{2i}, T_{2i}) to user U_i and U_{2i+1}, where $P_{2i}{=}s_{2i}{\cdot}P$ and $T_{2i}{=}\ H_1(P_{2i})S_{2i} + s_{2i}{\cdot}P_{2i}$.

User U_{2i+1} sends messages (P_{2i+1}, T_{2i+1}) to user U_i and U_{2i}, where $P_{2i+1} = s_{2i+1}{\cdot}P$ and $T_{2i+1} = H_1(P_{2i+1})\ S_{2i+1} + s_{2i+1}{\cdot}P_{2i+1}$.

Step 2: User U_i verifies

$$e(T_{2i} + T_{2i+1}, P) = e(P_{2i}, P_{2i})\ e(P_{2i+1}, P_{2i+1})\times e(H_1(P_{2i})\ Q_{2i} + H_1(P_{2i+1})\ Q_{2i+1}, P_{pub}).$$

User U_{2i} verifies

$$e(T_i + T_{2i+1}, P) = e(P_i, P_i)\ e(P_{2i+1}, P_{2i+1})\times e(H_1(P_i)\ Q_i + H_1(P_{2i+1})\ Q_{2i+1}, P_{pub}).$$

User U_{2i+1} verifies

$$e(T_i + T_{2i}, P) = e(P_i, P_i)\ e(P_{2i}, P_{2i}) \times e(H_1(P_i)\ Q_i + H_1(P_{2i})\ Q_{2i}, P_{pub}).$$

Step 3: If the equation in step 2 holds, then the user U_i computes $k_i = e(P_{2i}, P_{2i+1})^{s_i}$, the user U_{2i} computes $k_i = e(P_i, P_{2i+1})^{s_{2i}}$ and the user U_{2i+1} computes $k_i = e(P_i, P_{2i})^{s_{2i+1}}$, where

$$k_i = e(P_{2i}, P_{2i+1})^{s_i} = e(P_i, P_{2i+1})^{s_{2i}}$$
$$= e(P_i, P_{2i})^{s_{2i+1}} = e(P, P)^{s_i{\cdot}s_{2i}{\cdot}s_{2i+1}}.$$

Step 4: If $i = 1$, then the session key is k_i, else set $t_i = H_2(k_i)$ and User U_i broadcasts $t_i{\cdot}P$ to all users in the group.

Each user performs the procedure above until reaching the root, thus all users in the group can get a common session key k_1 .

3.3 The member changing phase

It is possible that users may join or leave the group during the communication. For the security considerations, the users before joining and after leaving the group cannot get the messages delivered in the group. Therefore it must perform some procedures if there are users want to join or leave the group.

3.2.1 Join protocol. Assume there are n users in the group originally. The newcomer will be inserting in the position of $n + 1$ of the complete binary tree. He will process the following steps:

Step 1: User U_{n+1} (the newcomer) gets the information of the group from User U_1, the information contains the number of the users in the group and the public key of all users.

Step 2: User U_{n+1} selects $s_{n+1} \in Z_q^*$ as his short-term private key, and broadcasts $P_{n+1} = s_{n+1}{\cdot}P$ and $T_{n+1} = H_1(P_{n+1})\ S_{n+1} + s_{n+1}{\cdot}P_{n+1}$ (for authenticate

P_{n+1}).

Step 3: Upon receiving P_{n+1} and T_{n+1}, each user authenticate P_{n+1} with T_{n+1}.

Step 4: New session key generation. Each value k_i on the node i on the path from n+1 to 1(root) will change.

When the user U_{n+1} join into the group, there are two cases in the original group: n (the number of users in the original group) is even or odd. If n is even, it means that the last parent in the binary tree has two children after the user U_{n+1} join in. If n is odd, then the last parent has only one left children. In this case, the last parent must pick another random number to complete key refreshing.

Case 1: If n is even

Let $i = n/2$, then the user U_i computes $k_i = e(P_{2i}, P_{n+1})^{s_i}$, the user U_{2i} computes $k_i = e(P_i, P_{n+1})^{s_{2i}}$ and the user U_{n+1} computes $k_i = e(P_i, P_{2i})^{s_{n+1}}$, where

$$k_i = e(P_{2i}, P_{n+1})^{s_i} = e(P_i, P_{n+1})^{s_{2i}} = e(P_i, P_{2i})^{s_{n+1}} = e(P, P)^{s_i \cdot s_{2i} \cdot s_{n+1}}.$$

If i = 1, then the new session key is k_1, else U_i sets $t_i = H(k_i)$, broadcasts $P_i = t_i \cdot P$, and performs the key agreement phase in subsection 3.2 until reach the root. Figure 2 is an example when U_{15} join the group, the values k_7 , k_3 and k_1 will change.

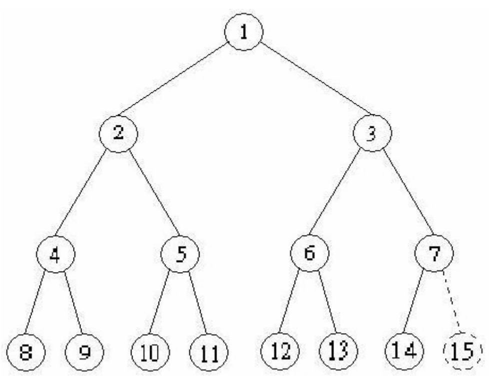

Figure 2. There are 14 (even) users in the group originally, the 15-th node is the newcomer.

Case 2: If n is odd

Let $i = (n+1)/2$, the user U_i selects $s_i' \in Z_q^*$, and broadcasts $P_i' = s_i' \cdot P$ and $T_i' = H_1(P_i)\, S_i + s_i \cdot P_i$ (for authenticate P_i'). Then the user U_i computes $k_i = e(P_i', P_{n+1})^{s_i}$ and the user U_{n+1} computes $k_i = e(P_i, P_i')^{s_{n+1}}$, where

$$k_i = e(P_i', P_{n+1})^{s_i} = e(P_i, P_i')^{s_{n+1}} = e(P, P)^{s_i \cdot s_i' \cdot s_{n+1}}.$$

If i = 1, then the new session key is k_1, else U_i sets $t_i = H(k_i)$, broadcasts $P_i = t_i \cdot P$, performs the key agreement phase in subsection 3.2 until reach the root. Figure 3 is an example when U_{14} join the group, the values k_7 , k_3 and k_1 will change.

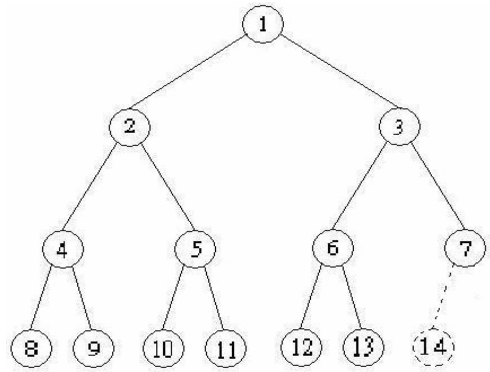

Figure 3. There are 13 (odd) users in the group originally, the 14-th node is the newcomer.

3.2.2 Leave protocol. Assume there are n users in the group originally. Let the leaving user is U_l, exchange the position of U_l and U_n, then delete U_l, and compute the new session key. According to the position of U_l, there are three cases be concerned. While $l = n$ (Case 1), it means that the leaving user is the last node in the binary tree. The protocol can delete the last node (U_n) directly, and generates a new common session key. If l = 1 (Case 2), it means that the position of the leaving user is the root of the binary tree. In the case, the protocol deletes the root node (U_1), then replaces the root with the last node (U_n) and generates a new common session key. While l not equate to 1 or n (Case 3), the protocol replaces U_l with U_n (the last node in the binary tree), then generates a new common session key of the group. We show the processes of each case in the following:

Case 1: If $l = n$

(i) If n is odd

Let $i = (n-1)/2$, the user U_i selects $s_i' \in Z_q^*$, and broadcasts $P_i' = s_i' \cdot P$ and $T_i' = H_1(P_i)\, S_i + s_i \cdot P_i$ (for authenticate P_i'). Then the user U_i computes $k_i = e(P_i', P_{n-1})^{s_i}$ and the user U_{n-1} computes $k_i = e(P_i, P_i')^{s_{n-1}}$, where

$$k_i = e(P_i', P_{n-1})^{s_i} = e(P_i, P_i')^{s_{n-1}} = e(P, P)^{s_i \cdot s_i' \cdot s_{n-1}}.$$

If i = 1, then the new session key is k_1, else U_i sets $t_i = H(k_i)$, broadcasts $P_i = t_i \cdot P$, performs the key agreement phase in subsection 3.2 until reach the root.

(ii) If n is even

Let $i = n/2$, the user U_i selects $s_i' \in Z_q^*$, replaces t_i with s_i', then broadcasts $P_i' = s_i' \cdot P$ and

$T_i' = H_1(P_i)\, S_i + s_i \cdot P_i$ (for authenticating P_i').

User U_i refreshes k_i , performs the key agreement phase in subsection 3.2 until reach the root.

Case 2: If $l = 1$

The protocol replaces U_1 with U_n, and performs the case 1. Figure 4 is the example when there are 15 users in the group originally and U_1 is leaving.

Case 3: If $l \in 2, \ldots, n\text{-}1$

The protocol replaces U_l with U_n, and performs the case 1.

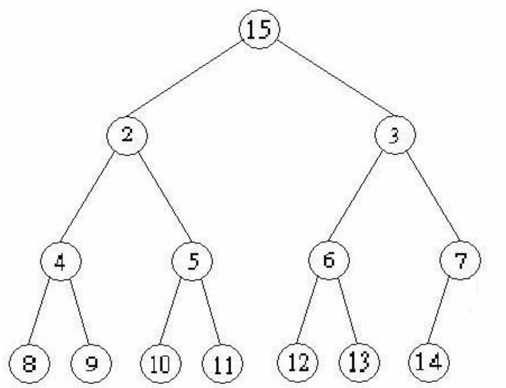

Figure 4. The leaving node is 1st node, replaced root by the last node 15.

4. Security analysis

We show the analysis of some security properties that we concerned in our proposed protocol. Those security properties are as following: key authentication, forward secrecy, key compromise, known session key security and key control.

(1) Key authentication:

The property of implicit key authentication to a user *A* is that no other users except *A* wants to agree upon can compute a particular key.

In our protocol, if user *A* wants to agree upon user *B*, then *B* must get the information from *A* to compute the particular key. By the ID-based authentication architecture, user *B* can verify the message that user *A* send. Without *A*'s information, no one can compute a particular key. It is clear that our protocol provide key authentication.

(2) Forward secrecy:

The property of forward secrecy is that if the compromise of any long-term private key of users does not affect the security of previous session keys.

In our protocol, the compromise of certain long-term private key gives no information about the session key, since the session key does not compute from long-term private key. The long-term private key is for authentication in the protocol. It shows that our protocol provide the forward secrecy.

(3) Key compromise:

The property of key compromise is that compromise of one user's long-term private key does not imply the other users' long-term private key.

In our protocol, each user's long term private key is chosen individually, so even the adversary have got the long term private key of a certain user, he still cannot imply the long term private key of other users.

(4) Known session key security:

The property of known key security is that the compromise of one session key should not affect the security of the current run of the protocol.

Suppose that there are three users U_1, U_2 and U_3 in the group, and the previous session key is $k_{prev} = e(P_2, P_3)^{s_1} = e(P_1, P_3)^{s_2} = e(P_1, P_2)^{s_3} = e(P, P)^{s_1 \cdot s_2 \cdot s_3}$, if the adversary wants to extract certain short term private key (e.g. s_3), then the adversary must face the BDHP in G_2, which is supposed to be hard.

(5) Key control:

The property of key control is that there is no user in the group can influence or control the value of the session key.

In our protocol, the common session key is determined by all users in the group, and no one can control or pre-determinate the session key.

5. Performance

We compare the computation of our protocol with authentication version of Barua et al.'s protocol [5] as Table 2. In their protocol, they also use a key tree structure. But each user is represented in the leaf node, every user need to hold the secret value from leaf node to the root. In our proposed protocol, we use a complete binary tree structure. Each node in the tree represent one user, we try to reduce the amount of secret value.

Table 2. The comparison of computational overhead

	Authentication version of Barua et al.'s protocol	Our proposed protocol
$R(n)$	$\lceil log_3\, n \rceil$	$\lceil log_2\, [(n+1)/2] \rceil$
$B(n)$	$3\times[(3^{\lfloor log_3 n \rfloor}\text{-}1)/2 + MIN(3^{\lfloor log_3 n \rfloor}, n\text{-}3^{\lfloor log_3 n \rfloor})]$	$3\times\lceil (n\text{-}1)/2 \rceil$
$P(n)$	$(3^{\lfloor log_3 n \rfloor}\times\lfloor log_3 n \rfloor) + (n - 3^{\lfloor log_3 n \rfloor})$	$\sum_{i=1}^{n} (i\times2^{i\text{-}1})+ [n\text{-}(2^{\lfloor log\, n \rfloor}\text{-}1)]\times\lfloor log\, n \rfloor$

$R(n)$: the rounds can be performed concurrently.
$B(n)$: the numbers of messages delivering.
$P(n)$: total numbers of pairings.

6. Conclusion

We proposed a key agreement protocol based on weil pairing. We use a complete binary tree to maintain a group key agreement process. In this protocol, each user can authenticate the received messages and identity of user by ID-based authentication architecture. It doesn't need to perform the certificate of users' public key and provides better efficiency. We also propose two methods for member joining and leaving, it shows that our protocol is suit for dynamic member changing. And our protocol fits in with some major security properties, which includes key authentication, forward secrecy, key compromise, known session key security and key control.

ACKNOWLEDGEMENTS

Research supported in part by the National Science Council grant NSC-93-2213-E-032-019 Taiwan, Republic of China.

Reference

[1] Boneh D, Franklin M. "Identity-based encryption from the Weil pairing," *Advances in Cryptology-Crypto'2001, LNCS 2139*, Springer-Verlag, 2001, pp. 213-229.

[2] Cocks C. "An identity based encryption scheme based on quadratic residues," *Cryptography and Coding, LNCS 2260*, Springer-Verlag, 2001, pp. 360-363.

[3] Diffie W, Hellman M. "New directions in cryptography," *IEEE Transations on Information Theory*, Vol. 22, 1976, pp. 644-654.

[4] Joux A., "A one-round protocol for tripartite Diffie-Hellman," *Proc. Fourth algorithmic Number Theory Symposium*, Lecture Notes in Computer Science, Springer-Verlag, Vol. 1838, 2000, pp. 385-394.

[5] R. Barua, R. Dutta, P. Sarkar, "Extending Joux's Protocol to Multi Party Key Agreement," *3rd International Cryptology Conference in India -- Indocrypt'2003*, *LNCS* 2904, Springer-Verlag, 2003, pp. 205--217.

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

[7] Sheng-Li Liu, Fang-Guo Zhang and Ke-Fei Chen, "Authenticating Tripartite Key Agreement Protocol with Pairings," *Journal of computer science and Technology*, Vol. 19, No. 2, 2004, pp. 169-176.

[8] Tsuji S, Itoh T. "An ID-based cryptosystem based on the discrete logarithm problem," *IEEE Journal of Selected Areas in Communications*, Vol. 7, No. 4, 1989, pp. 467-473.