A New Efficient Authenticated ID-Based Group Key Agreement Protocol

Morteza Arifi[1], Mahmoud Gardeshi[1] and Mohammad Sabzinejad Farash[2]

[1] Fath research center, Imam Hussein University, Tehran, Iran,
[2] Department of Mathematics and Computer Sciences, Tarbiat Moallem University, Tehran, Iran.
morteza.arifi@gmail.com, mgardeshi@ihu.ac.ir, m.sabzinejad@gmail.com

Abstract

Group key agreement (GKA) protocols Play a main role in constructing secure multicast channels. These protocols are algorithms that describe how a group of parties communicating over a public network can gain a common secret key. ID-based authenticated group key agreement (AGKA) cryptosystems based on bilinear pairings are update researching subject because of the simplicity of their public key management and their efficiency. The key agreement protocol is a good way 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 situation. In this paper we propose a protocol based on Weil pairing, ID-based authentication and complete ternary tree architecture. We show that our protocol satisfies all known security requirements, and therefore it is more secure and efficient than the compared group key exchange protocols that we discuss in this article.

Keywords

Group key agreement, ID-based Authentication, Pairing, Ternary Tree.

1. Introduction

A group key agreement is a protocol which allows a group of users to exchange information over public and insecure network to agree upon a common secret key which a group session key can be derived. This group session key can be used to achieve desirable security goals, such as authentication, confidentiality and data integrity.

There are two methods to generate a session Key: key distribution and key agreement. Key distribution needs a group controller to hold the information of the whole users in the group, if the group controller is stopped or attacked, then the group fail. At the same time as the group members have dynamic changing, the group controller may be effectiveness in this situation. 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 predict the session key.

The first key agreement protocol was proposed by Diffie-Hellman [3]. It can guarantee the security of communication between the two users. But it does not authenticate users; hence it is vulnerable to 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 upon a common session key, only one message must be delivered by each user in the protocol. But, Joux's protocol does not authenticate the users, and is vulnerable to "man-in-the-middle" attack too.

Both group key establishment techniques can be Analyzed in context of either fixed or dynamic groups; obviously we can always create the group key for the modified group by restarting the protocol. Nevertheless, this may be inefficient if groups are large or the protocol has expensive computation cost. Therefore, many dynamic group key establishment protocols designed for efficient operations for addition and leaving out from group members. One-way function trees (OFTs) can be used to compute a tree of keys. The keys are computed from the leaves to the root. Key hierarchies are common in dynamic group key distribution protocols for collaborative schemes, since they improve protocol efficiency upon dynamic group changes. The use of OFTs for group key was first proposed by Sherman in [1]. Any two party key agreement protocols satisfying some particular properties [1] can be extended to a *n*-party key agreement protocol using one-way function trees. Tree-based Group Diffie-Hellman (TGDH) [11] is one of the protocols that extend the Diffie-Hellman protocol to a group key agreement protocol with one-way function trees.

Reddy and Divya Nalla [5] extend the Identity Based two-party authenticated key agreement protocol to an authenticated group key agreement protocol, using the one-way function trees to generate the first ID-based group key agreement protocol. In their protocol the leaves of the tree denote individual users of group. Sheng-Hua Shiau et al.'s protocol [10], also use a key tree structure. But they use complete binary tree structure i.e. each node in the tree represent one user. A ternary tree based protocol was proposed by Barua et al. [8] that extend the basic Joux's [8] protocol to multi-party setting. In their protocol the leaves of the tree denote individual users and each internal node corresponds to a representative that represents set of users in the sub tree rooted at that node. But their protocol was unauthenticated also. Dutta et al. [7] authenticate this unauthenticated protocol using multi-signatures.

In this paper, we propose a group key agreement protocol based on Weil pairing. In our protocol, we use the ID-based authentication and complete ternary Tree architecture such that every node in the tree represents a user of the group. If there are some users want to join or leave the group, not all users in the group need to renew their all computations to get secret key; so it is suit for dynamic changing environment. This paper is organized as followings: Section 2 proposes the notations and assumptions. 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 protocols. Finally, section 6 shows our result.

2. Preliminaries

Assume G_1 be an additive group with a prime Order q and G_2 be a multiplicative group with the order q. P is a generator of G_1. We assume that the discrete logarithm problem (DLP) is intractable in G_1 and G_2. And e is a bilinear mapping between two groups ($e: G1 \times G1 \rightarrow G2$). This bilinear map must satisfy the following properties:

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

For using bilinear mappings for implementation protocol, there are some problems and assumptions [8] as followings:

1. DDH (Decisional Diffie-Hellman) Problem [2] in G_1: Given (P, aP, bP, cP) for some $a, b\ and\ c \in Z_q^*$, decides whether $cP = abP$ or not. The DDH problem can be solved in polynomial time by $e(aP, bP) = e(cP, P)$

   ***DDH* Assumption:** There is not any polynomial time algorithm to solve the DDH problem in G_2.
2. HDH (Hash Decisional Diffie-Hellman) Problem [6] in G_1: Given (P, aP, bP, c) and a hash function $H_1: G_1 \rightarrow Z_q^*$, decides if $c = H_1(abP)\ mod\ q$.

   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), compute $e(P, P)^{abc}$.

   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 $d = H_2(e(P, P)^{abc})\ mod\ q$.

   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. In Order to perform ID-based authentication, each user needs to register to the *KGC* (Key Generation Center) in initial phase. We separate our protocol into three phases: the initial phase, the key agreement phase and the member changing phase.

3.1. The initial phase

In this subsection we show that how each user can registers to the *KGC*. After registering to the *KGC* every member can perform the key agreement phase to compute the group session key. For this purpose at first KGC selects a random number $s \in_R Z_q^*$ then compute and publish $P_{pub} = sP$ as his or her public key. KGC keeps s as his or her master key secretly.

The identity of each user U_i and his or her long-term public key are respectively $ID_i \in \{0, 1\}^*$ and $Q_i = H(ID_i)$. Each user will use Q_i to register to the *KGC* from secure channel by the following steps:

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

Step 2: *KGC* computes user U_i*'s* long-term private key $S_i = sQ_i$ and send it to U_i.

The public parameters of the protocol are: $(G_1, G_2, e, q, P, P_{pub}, H, H_1, H_2, H_3)$. Where $H: \{0,1\}^* \rightarrow G_1$, $H_1: G_1 \rightarrow Z_q^*$, $H_2: G_2 \rightarrow Z_q^*$ and $H_3: G_1 \times G_1 \rightarrow Z_q^*$ are cryptographic hash functions.

3.2. The key agreement phase

In this subsection, we show that how permissible users collaborate to compute a common session key. In our protocol, the key agreement process is based on complete ternary tree structure. Each node in that tree is representing one user; Figure 1 is an example of 16 users.

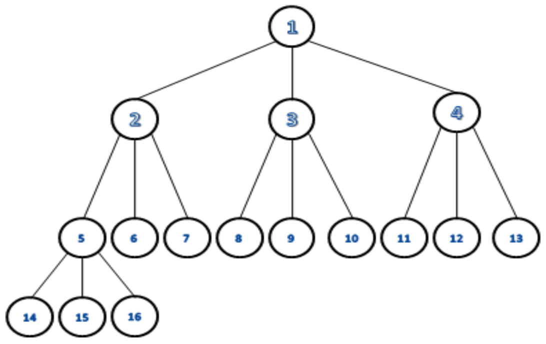

Fig. 1. A complete ternary tree 0f a group with 16 users

Assume there are n users in this group, every user U_i ($i \in \{1,..,n\}$) has his/her long-term public/private key (Q_i/S_i). users will choose a random number a_i as short-term private key in each new run of the protocol. There are four kinds of nodes in a complete ternary tree: the leaf node, the internal node with one left child only, the internal node with two children (Boy & Girl) and the forth kind is internal node with three children.

Case1. The node is a leaf ($3i > n$).

1.1. Sets $t_i = a_i$.

1.2. User U_i sends $t_i.P$ to his (her) parent and his (her) sibling node (brother or sister).

Case2. The node only has one Boy child ($3i$-*1* = n)

2.1. User U_i selects another random number $\acute{a}_i$ additionally.

2.2. User U_i sends messages $(P_i, \acute{P}_i, T_i)$ to the user U_{3i-1}, where $P_i = a_i P$, $\acute{P}_i = \acute{a}_i P$ and $T_i = H_3(P_i, \acute{P}_i)\, S_i + a_i P_{pub}$.
User U_{3i-1} sends messages (P_{3i-1}, T_{3i-1}) to the user U_i, where $P_{3i-1} = a_{3i-1}P$ and $T_{3i-1} = H_1\,(P_{3i-1})\, S_{3i-1} + a_{3i-1}P_{pub}$.

2.3. User U_i verifies the following equation $e(T_{3i-1}, P) = e(H_1(P_{3i-1})Q_{3i-1} + P_{3i-1}, P_{pub})$ and user U_{3i-1} verifies the following equation $e(T_i, P) = e(H_3(P_i, \acute{P}_i)Q_i + P_i,\ P_{pub})$.

2.4. If the equations in 2.3 hold, U_i computes $K_i = e(P_{3i-1}, \acute{P}_i)^{a_i}$ and U_{3i-1} computes $K_{3i-1} = e(P_i, \acute{P}_i)^{a_{3i-1}}$.
Note that $K_{3i-1} = e(P,P)^{a_{3i-1}\acute{a}_i a_i} = K_i$.

2.5. If $i = 1$, then the session key is K_i, else user U_i sets $t_i = H_2(K_i)$ and sends $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group.

Case3. The node has two children ($3i = n$). In this case three users U_i and U_{3i-1} and U_{3i} simply do the tripartite one round key agreement.

3.1. User U_i sends messages $(P_i,\ T_i)$ to the users U_{3i-1} and U_{3i} User U_{3i-1} sends messages (P_{3i-1}, T_{3i-1}) to the user U_i, and U_{3i} and finally User U_{3i} sends messages (P_{3i}, T_{3i}) to the user U_i, and U_{3i-1} where in general $P_k = a_k P$ and $T_k = H_1(P_k)S_k + a_k P_{pub}$, $(k = i, 3i-1, 3i)$.

3.2. In this step in general each User U_k verifies the received messages $(P_A,\ T_A)$, $(P_B,\ T_B)$ from the two other users with the following equation: $e(T_A + T_B, P) = e(H_1(P_A)Q_A + H_1(P_B)Q_B + P_A + P_B\ ,\ P_{pub})$.

3.3. If the equation in step 3.2 holds,

- U_i Computes:
$K_i = e(P_{3i}, P_{3i-1})^{a_i} = e(P,P)^{a_i a_{3i} a_{3i-1}}$
- U_{3i-1} computes:
$K_{3i-1} = e(P_{3i}, P_i)^{a_{3i-1}} = e(P,P)^{a_i a_{3i} a_{3i-1}}$
- U_{3i} computes:
$K_{3i} = e(P_{3i-1}, P_i)^{a_{3i}} = e(P,P)^{a_i a_{3i} a_{3i-1}}$.

It is clear that $K_i = K_{3i} = K_{3i-1}$.

3.4. If $i = 1$, the session key is K_i, else user U_i sets $t_i = H_2(K_i)$ and sends $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group.

Case4. The node has three children.

Previous cases (case1, case2 and case3) that we explained before are somehow like complete binary cases that Sheng et al. proposed in their protocol [10] (in this paper we used different equation for authentication that needs two pairing whereas their authentication needs three pairing).

This case which is the main contribution of the paper and the most important part of the key agreement phase of the complete ternary tree is as follows:

4.1. Each user chooses $a_k \in_R Z_q^*$ then computes $P_k = a_k P$ and $T = H_1(P_k)\, S + aP_{pub}$.

- U_i Sends the message (P_i, T_i) to the users U_{3i-1}, U_{3i} and U_{3i+1}.
- U_{3i-1} Also sends the message (P_{3i-1}, T_{3i-1}) to the users U_i, U_{3i} and U_{3i+1}.
- U_{3i} Also sends messages (P_{3i}, T_{3i}) to the users U_i, U_{3i-1} and U_{3i+1}.

- U_{3i+1} Also sends messages (P_{3i+1}, T_{3i+1}) to the user U_i , U_{3i} and U_{3i-1}.

4.2. Each user verifies the messages received in the previous step.

- Generally, the user U_k verifies the received messages (P_A, T_A), (P_B, T_B), (P_C, T_C) by the following equation:

$$e(T_A + T_B + T_C, P) = e(H_1(P_A)Q_A + H_1(P_B)Q_B + H_1(P_C)Q_C + P_A + P_B + P_C, P_{pub}) \quad (1)$$

- Notice that each user verifies the three other users simultaneously.

4.3. If the verification relation (1) holds for users U_i and U_{3i} then:

- U_i Computes and sends the messages $(P_{i,3i+1}, \acute{T}_{3i-1})$, $(P_{i,3i-1}, \acute{T}_{3i})$ and $(P_{i,3i-1}, \acute{T}_{3i+1})$ to users U_{3i-1}, U_{3i} and U_{3i+1} respectively.
- U_{3i} also computes and sends the message $(P_{3i,3i-1}, \acute{T}_i)$ to the user U_i.
- Generally, $P_{i,j} = a_i P_j$ and $\acute{T}_k = H_1(P_{i,j})S_i + a_i P_{pub}$.

4.4. In general, each user U_k verifies the received message $(P_{i,j}, \acute{T}_k)$ by the following equation:

$$e(\acute{T}_k, P) = e(H_1(P_{i,j})Q_i + P_i , P_{pub}) \quad (2)$$

4.5. Finally, if the equation (2) holds, each user computes the secret key as follows:

$$K_i = e(P_{3i,3i-1}, P_{3i+1})^{a_i} = e(P,P)^{a_i a_{3i} a_{3i-1} a_{3i+1}}$$
$$K_{3i-1} = e(P_{i,3i+1}, P_{3i})^{a_{3i-1}} = e(P,P)^{a_i a_{3i} a_{3i-1} a_{3i+1}}$$
$$K_{3i} = e(P_{i,3i-1}, P_{3i+1})^{a_{3i}} = e(P,P)^{a_i a_{3i} a_{3i-1} a_{3i+1}}$$
$$K_{3i+1} = e(P_{i,3i-1}, P_{3i})^{a_{3i+1}} = e(P,P)^{a_i a_{3i} a_{3i-1} a_{3i+1}}$$

It is clear that all the computed keys are equal, i.e. $K_i = K_{3i} = K_{3i-1} = K_{3i+1}$.

4.6. If $i = 1$, it means that the users reached to the root of the tree and the session key is K_i, else U_i sets $t_i = H_2(K_i)$ and sends $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group.

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

3.3. The member changing phase

It is possible that users may want to join or leave the group during a communication. For the security considerations, the users before joining and after leaving the group must be unable to get the messages delivered in the group. Therefore we must perform some actions for the users that want to join or leave the group.

3.3.1. The join protocol

Assume that, there are *n* users in the group before Any member joins the group. The position of the newcomer user will be at $(n + 1)$th node of the complete ternary tree. (S)he will perform the following steps:

1. User U_1 sends the information of the group which contains the number of the users in the group and the public key of all users, to the user U_{n+1} (the newcomer).
2. User U_{n+1} choose random number $a_{n+1} \in_R Z_q^*$ as his/her short-term private key, then computes and broadcasts $P_{n+1} = a_{n+1}P$ and the signature $T_{n+1} = H_1(P_{n+1})\, a_{n+1} + a_{n+1}P_{pub}$.
3. According to the following moods the new session key will be generated. Each key K_i kept by the node i on the path from $(n + 1)$th node to root of the tree will be changed.

When the user U_{n+1} joins into the group with *n* user, there are three possible moods in the original group:

Mood1. $n = 0 \ mod\ 3$ or $n = 3K$.

The last parent has three children after the user U_{n+1} joins into the group. See the figure 2.

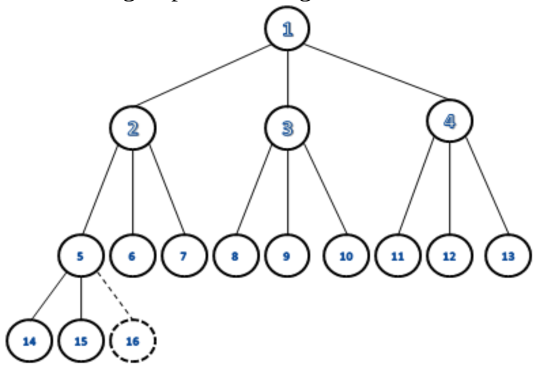

Fig.2 There are 15 (n=3k) users in the group originally, the 16th node is the newcomer.

Let $= n/3$, U_i is the parent of the newcomer user U_{n+1}. In this case U_i has three children and the process of computing the session key is like Case4 in the key agreement phase, and in this situation U_{n+1} acts as U_{3i+1} in Case4. At the end of Case4:

- U_i Computes
$$K_i = e(P_{3i,3i-1}, P_{n+1})^{a_i}$$
- U_{3i-1} computes
$$K_{3i-1} = e(P_{i,n+1}, P_{3i})^{a_{3i-1}}$$
- U_{3i} computes
$$K_{3i} = e(P_{i,3i-1}, P_{n+1})^{a_{3i}}$$

- U_{n+1} computes

$$K_{n+1} = e(P_{i,3i-1}, P_{3i})^{a_{n+1}}$$

It is clear that:

$$K_i = K_{3i} = K_{3i-1} = K_{n+1} = e(P,P)^{a_i a_{3i} a_{3i-1} a_{n+1}}$$

If $i = 1$, then the session key is K_i, else U_i sets $t_i = H_2(k_i)$ and broadcasts $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group. Then he continues the key agreement phase to reach the root.

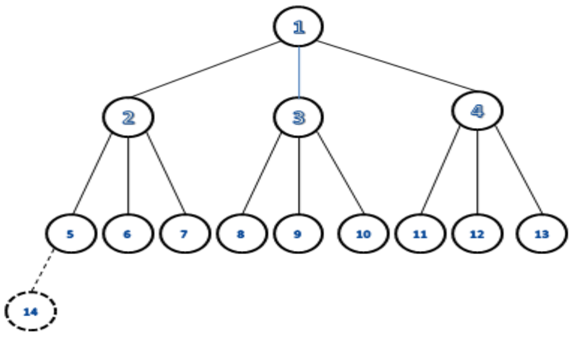

Fig.3 There are 13 (n=3k+1) users in the group originally, the 14th node is the newcomer.

Mood2. $n = 1 \ mod \ 3$ or $n = 3k + 1$.

The last parent has one child after the user U_{n+1} joins into the group. See the figure 3.

Let $= (n + 2)/3$, U_i is the parent of the newcomer user U_{n+1} and U_i now has just U_{n+1} as his child. Like the Case2 in the key agreement phase,

- User U_i selects another random number $\acute{a}_i$ additionally and computes $P_i = a_i P$, $\acute{P}_i = \acute{a}_i P$ and $T_i = H_3(P_i, \acute{P}_i)\, S_i + a_i P_{pub}$ then sends the message $(P_i, \acute{P}_i, T_i)$ to the user U_{n+1}.
- User U_{n+1} also computes $P_{n+1} = a_{n+1} P$ and $T_{n+1} = H_1\,(P_{n+1})\, S_{n+1} + a_{n+1} P_{pub}$ then sends the message (P_{n+1}, T_{n+1}) to the user U_i.
- U_i and U_{n+1} verify their received messages same as step 2.3 of Case2. If verification is valid then U_i computes $K_i = e(P_{n+1}, \acute{P}_i)^{a_i}$, and U_{n+1} computes $K_{n+1} = e(P_i, \acute{P}_i)^{a_{n+1}}$. Note that $K_{n+1} = e(P,P)^{a_{n+1} \acute{a}_i a_i} = K_i$.

If $i = 1$, then session key is K_i, else user U_i set $t_i = H_2(K_i)$, and sends $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group and then continues the key agreement phase until reaching the root.

Mood3. $n = 2 \ mod \ 3$ or $n = 3K - 1$.

It means that the last parent in the ternary tree has two children after that user U_{n+1} joins into the group. See the figure 4.

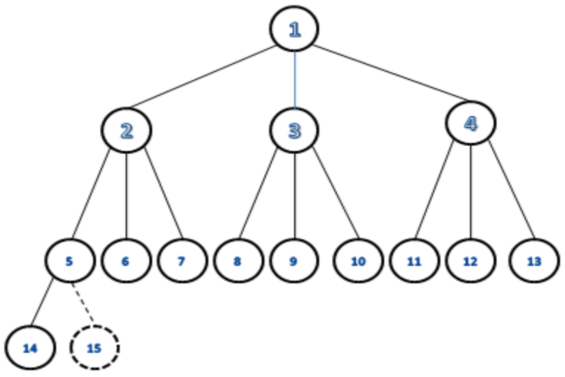

Fig.4 There are 14 (n=3k-1) users in the group originally, the 15th node is the newcomer

Let $i = (n + 1)/3$, U_i is the parent of the newcomer user U_{n+1}, in this case U_i has two children and the process of computing the session key is like Case3 in the key agreement phase, after performing steps 3.1 and 3.2,

- U_i Computes
  $K_i = e(P_{n+1}, P_{3i-1})^{a_i} = e(P,P)^{a_i a_{n+1} a_{3i-1}}$
- U_{3i-1} computes
  $K_{3i-1} = e(P_{n+1}, P_i)^{a_{3i-1}} = e(P,P)^{a_i a_{3i} a_{3i-1}}$
- U_{n+1} computes
  $K_{n+1} = e(P_{3i-1}, P_i)^{a_{n+1}} = e(P,P)^{a_i a_{n+1} a_{3i-1}}$

It is clear that $K_i = K_{n+1} = K_{3i-1}$. If $i = 1$, then session key is K_i, else U_i sets $t_i = H_2(K_i)$ and sends $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group. And then continues the key agreement phase until reaching the root.

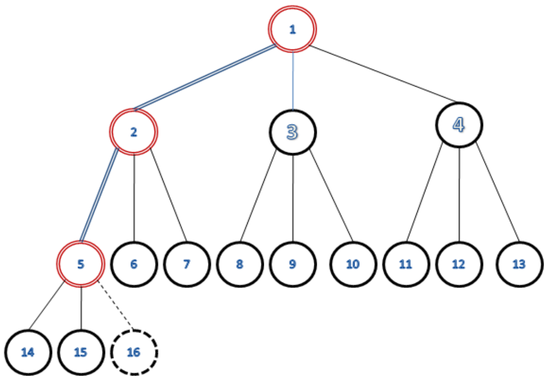

Fig.2 when user U_{16} join the group values K_5, K_2 and K_1 will change.

As mentioned before for refreshing the session key, each session key K_i kept by the node i on the path from $(n + 1)$th node to first (root) node will be changed. So for all three cases explained in the join protocol, the session keys K_5, K_2 and consequently K_1 will be changed. Figure 5 shows the path of these changes when user U_{16} joins into the group.

3.3.2. The leave protocol

Suppose that, there are n users in the group Originally. Let the leaving user be U_l, hence we exchange the position of U_n and U_l, then delete U_l

and compute a new session key. According to the position of U_l, there are three moods as follow.

Mood1. $l = n$

In this mood, the leaving user is the last node in the ternary tree. The protocol can delete the last node U_n directly, and generate a new common session key.

1. If $n = 3k + 1$, let $i = (n + 2)/3$. In this case after U_n left the group, U_i has two children. U_i selects a new random number $\acute{a}_i$ as his short-term private key and performs same as Case3 in the key agreement phase.
   - ➢ U_i computes $\acute{P}_i = \acute{a}_i P$ and $\acute{T}_i = H_3(\acute{P}_i)\, S_i + \acute{a}_i P_{pub}$, then sends the message $(\acute{P}_i,\ \acute{T}_i)$ to users U_{3i-1} and U_{3i} and finally computes $K_i = e(P_{3i}, P_{3i-1})^{\acute{a}_i} = e(P,P)^{\acute{a}_i a_{3i} a_{3i-1}}$
   - ➢ User U_{3i-1} after verifying the message $(\acute{P}_i,\ \acute{T}_i)$ computes $K_{3i-1} = e(P_{3i}, \acute{P}_i)^{a_{3i-1}} = e(P,P)^{\acute{a}_i a_{3i} a_{3i-1}}$
   - ➢ User U_{3i} also after verifying the message $(\acute{P}_i,\ \acute{T}_i)$ computes $K_{3i} = e(P_{3i-1}, \acute{P}_i)^{a_{3i}} = e(P,P)^{\acute{a}_i a_{3i} a_{3i-1}}$
   - ➢ If $i = 1$, then the session key is K_i, otherwise U_i sets $t_i = H_2(K_i)$ and sends $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group and then continues the key agreement phase until reaching the root.
2. If $n = 3K$, let $i = n/3$. In this case after that U_n left the group, U_i has one child and same as the Case2 in the key agreement phase selects another random number $\acute{a}_i$ additionally.
   - ➢ U_i computes $P_i = a_i P$, $\acute{P}_i = \acute{a}_i P$ and $T_i = H_3(P_i, \acute{P}_i)\, S_i + a_i P_{pub}$ and sends the message $(P_i, \acute{P}_i, T_i)$ to the user U_{3n-1}.
   - ➢ User U_{3n-1} also sends the message (P_{3n-1}, T_{3n-1}) to the user U_i, where $P_{3n-1} = a_{3n-1}P$ and $T_{3n-1} = H_1(P_{3n-1})\, S_{3n-1} + a_{3n-1}P_{pub}$.
   - ➢ U_i and U_{3n-1} verify their received messages like step 2.3 of Case2. If verification relations hold, U_i computes $K_i = e(\acute{a}_i P_{3n-1}, \acute{P}_{3n-1})^{a_i}$, and U_{3n-1} computes $K_{3n-1} = e(\acute{a}_{3n-1} P_i, \acute{P}_i)^{a_{3n-1}}$, where $K_{3n-1} = e(P,P)^{\acute{a}_{3n-1} a_{3n-1} \acute{a}_i a_i} = K_i$.
   - ➢ If $i = 1$, then the session key is K_i, else U_i sets $t_i = H_2(K_i)$ and sends $P_i = t_i P$ to his parent, sibling nodes and sibling's children in the group and then continues the key agreement phase until reaching the root.
3. If $n = 3k - 1$, let $i = (n + 1)/3$. In this case after that U_n left the group, U_i does not have any children so he chooses a new random number $\acute{a}_i$ as his short-term private key and replaces t_i with $\acute{a}_i$ then sends $\acute{P}_i = \acute{a}_i P$ and $\acute{T}_i = H_3(\acute{P}_i)\, S_i + \acute{a}_i \acute{P}_i$. Finally U_i refreshes K_i and then continues the key agreement phase until reaching the root.

Mood2. $l = 1$

In this mood, the position of the leaving user is the root of the ternary tree. So the protocol deletes the root node U_1 and replaces the root with the last node U_n then performs as mood 1 in the leave protocol which we explained to generate a new common session key.

Figure 6 shows an example for a group with 16 users originally and U_1 left the group.

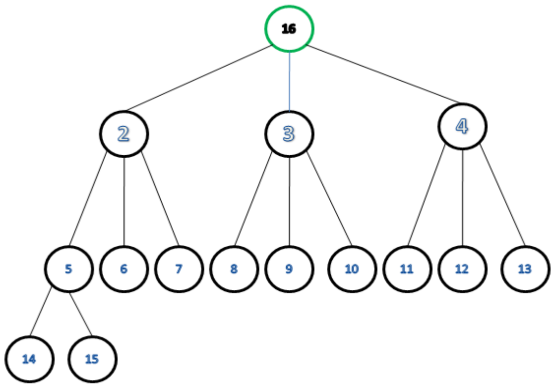

Fig. 3. The leaving node is 1st node, replaced root with the last node 16.

Mood3. $l \in \{2, \ldots, n - 1\}$

In this mood, the protocol replaces U_l with U_n (the last node in the ternary tree), and continues as mood 1 in the leave protocol to generate a new common session key.

4. Security Analysis

In this section we show the analysis of some security properties of our proposed protocol. These security properties are as following:

(1) Known session key security:

This property states that if one session key has been compromised, the security of the current run of the protocol should not be affected. Assume that there are four users U_1, U_2, U_3 and U_4 in the group, and the previous session key is

$K_{prev} = e(P_2, a_4P_3)^{a_1} = e(a_4P_1, P_3)^{a_2} = e(a_4P_1, P_2)^{a_3} = e(a_2P_1, P_3)^{a_4} = e(P,P)^{a_1a_2a_3a_4}$,

if the adversary wants to extract certain short-term private key (e.g.a_1), then (s)he must solve the BDHP in G_2, which is supposed to be hard. Also, the session key depends on random numbers selected by the users in each run of the key agreement phase, so the session key will be different each time.

(2) Key authentication: (implicit) key authentication requires that each legitimate protocol participant is assured that no other party except other legitimate participants can establish the group session key.

In our protocol, each participant signs his/her generated messages by his/her own long term private key, consequently all users upon receiving a message from each other, first verifies it then follows the protocol's procedure. So the participant can be assured that only legitimate users can perform the protocol and establish the group session key.

(3) Forward secrecy:

If any long-term private key of the users has been revealed the security of the previous session keys should not be affected. In the proposed protocol, the long-term private key is used only for the authentication, and the protocol does not use the long term private key of the users to compute the common session key. So it is clear that our protocol satisfies the forward secrecy.

(4) key-compromise impersonation resilience: This security property prevents the adversary who obtains a long-term key of a user from being able to impersonate other users. We note that long-term keys are usually private keys which used either for signature generation or decryption; so long-term keys are used primarily for the purpose of authentication rather than the actual computation of the group key. So we do not need to concern for this attack.

(5) Key control: The property of key control says that there is no any legitimate user in the group whom pre-determines or influences the value of the session key. In our protocol, the common session key is determined by the collaboration of all users in the group, so no one can control or pre-determinate the session key.

5. Performance

We compare the computations and communications of our protocol with Sheng et al.'s protocol [10] and Barua et al.'s protocol [7] as Table 1. Both of these protocols use a key tree structure. But in the later each user is represented in the leaf node, also (s)he needs to hold the secret value from leaf node to the root. Barua et al used ternary tree structure but the former uses complete binary tree structure and each node in the tree represent one user. In our proposed protocol we use complete ternary tree structure also and each node in the tree represent one user. In contrast with Barua et al.'s protocol [7] our protocol is based on the identity of the users so we omit the expenses of PKI.

To compute the total number of pairings we sum the total number of pairings which the leaf nodes compute and the total number of pairings that internal nodes compute. To compute the session key, leaf nodes should continue the computation procedure which explained in section 3.2 (according to his case) until reaches the root of the tree. So the user who is in the Leaf node of tree should repeat the computations for $R(n)$ times where $R(n)$ is the number of protocol's round, and there are $\left(n-\left(\frac{3^{R(n)}-1}{2}\right)\right)$ leaf nodes in the leaves, but we should note that there may be leaf nodes which are not in the last level ($i.e. R(n)$) and they may be in the $[R(n)-1]$-th level so they repeat the computations one round less than the leaf nodes which lie in the $R(n)$-th level. So we should minus the number of them from the $\left(n-\left(\frac{3^{R(n)}-1}{2}\right)\right)R(n)$, and we can check that there are $3^{R(n)-1}-\left\lceil\frac{n-3^{R(n)}-1/2}{3}\right\rceil$ leaf nodes that are not in the last level. For the internal nodes, in each level l we have 3^{l} users. Each of them repeat the procedure which explained in section 3.2 for $l+1$ times until get the session key, so the total number from level 0 to level $R(n)-1$ is $\sum_{i=1}^{R(n)} i\,3^{i-1}$. Finally the total number of repetitions of procedure which we explain in section 3.2 is

$$\sum_{i=1}^{R(n)} i\,3^{i-1}+\left(n-\left(\frac{3^{R(n)}-1}{2}\right)\right)R(n)$$
$$-3^{R(n)-1}+\left\lceil\frac{n-3^{R(n)}-1/2}{3}\right\rceil \qquad (3)$$

and for getting the common secret K_i in procedure 3.2 we need 4 pairing for authenticating the messages and one for computing the K_i. So we should multiply the equation (3) by the number 5. We can check that our protocol is more efficient in computation cost comparing with the two other protocols when the number of users is high, but when then number of users is not high their computation cost may be close to our protocol.

For computing the total numbers of messages that users will deliver, each internal node send 9 messages and each leaf node send 4 messages, by multiplying the total number of internal nodes and also the total number of the leaf nodes, by 9 and 4 respectively and adding them together we can find that the $B(n)$ is almost $\leq 5(n-1)$. Our protocol is better than Barua et al's [3] protocol in the communication cost. In the table1:

$R(n)$: total number of rounds that can be performed concurrently.

$B(n)$:total numbers of messages delivering.

$P(n)$: total numbers of pairings.

6. Conclusion

We proposed an authenticated ID-based group key agreement protocol based on pairing. We use a complete ternary tree to maintain a group key agreement process and each node in the tree represents one user. In this protocol, each user can authenticate the received messages by ID-based authentication structure. It doesn't need to verify the certificate of users' public key. It provides better efficiency. We also proposed how users can join to or leave from the group. It shows that our protocol is suit for dynamic member changing. And our protocol fits with some most important security properties, which includes known session key security, key authentication, forward secrecy, key compromise impersonation and key control.

Table 1. The comparison of computational and communication overhead

	$R(n)$	$B(n)$	$P(n)$
Barua et al's [3]	$\lceil \log_3 n \rceil$	$\leq \frac{5}{2}(n-1) + n[R(n)-1] - \frac{3}{2}\left(3^{R(n)-1} - 1\right)$	$\lceil \log_3 n \rceil \times n \times 5$
Sheng et al's [10]	$\lfloor \log_2 n \rfloor$	$\leq 3(n-1)+1$	$\left\{ \begin{matrix} \sum_{i=1}^{R(n)} i\,2^{i-1} + \left(n - \left(2^{R(n)}\right)\right) R(n) \\ -2^{R(n)} + \left\lceil \frac{n+1}{2} \right\rceil \end{matrix} \right\} \times 4$
Our protocol	$\lceil \log_3(2n + 1/3) \rceil$	$\leq 5(n-1)$	$\left\{ \begin{matrix} \sum_{i=1}^{R(n)} i\,3^{i-1} + \left(n - \left(\frac{3^{R(n)}-1}{2}\right)\right) R(n) \\ -3^{R(n)-1} + \left\lceil \frac{n - 3^{R(n)} - 1/2}{3} \right\rceil \end{matrix} \right\} \times 5$

Acknowledgment

I would like to thank Iran Telecommunication Research Center (ITRC) for supporting this research.

References

[1] D.A. McGrew and A.T. Sherman. "Key establishment in large dynamic groups using one-way function trees". Manuscript, 1998.

[2] D. Boneh and M. Franklin. "Identity-Based Encryption from the Weil Pairing." In Advances in Cryptology - CRYPTO '01, LNCS 2139, pages 213-229, Springer-Verlag, 2001.

[3] Diffie W, Hellman M. "New directions in cryptography," IEEE Transactions 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] K. C. Reddy and D. Nalla, "Identity Based Authenticated Group Key Agreement Protocol," in Proceedings of INDOCRYPT'02, vol. LNCS 2551, 2002, pp. 215–233

[6] M.Abdalla, M.Bllare and P.Rogaway. DHIES "An encryption scheme based on the Diffie-Hellman problem," CT-RSA 2001 : 143-158

[7] R. Dutta, R. Barua and P. Sarkar. "Provably Secure Authenticated Tree Based Group Key Agreement," Proc. of ICICS'04, LNCS 3269, Springer 2004, pp. 92-104.

[8] 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.

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

[10] Sheng-H, R Hwang, M Lin, "Key Agreement Protocol Based on Weil Pairing," aina, vol. 1, pp.597-602

[11] Y. Kim, A. Perrig, and G. Tsudik. "Simple and fault tolerant key agreement for dynamic collaborative groups," in Proceedings of 7th ACM Conference on Computer and Communications Security, pp. 235-244, ACM Press, November 2000.