情報保護學會論文誌
第 22 卷 第 6 號, 2012. 12

동적 피어 그룹을 위한 삼진 트리방식의 인증된 그룹 키 합의 프로토콜

김 호 희,†‡ 김 순 자
경북대학교 IT대학 전자공학부

A Ternary Tree-based Authenticated Group Key Agreement For Dynamic Peer Group

Ho-hee Kim,†‡ Soon-ja Kim
Kyungpook National University

요 약

그룹응용이 늘어감에 따라 효율적인 인증 그룹 키 합의 프로토콜이 많은 관심을 받고 있다. Lee *et al.*는 삼진트리 구조와 pairing기반의 암호방식을 가진 트리기반의 그룹 키 합의 프로토콜을 제안했다. 그들의 프로토콜은, 모든 멤버들의 세션랜덤 키를 아는 오직 한 명의 그룹 스폰서가 블라인드 키들을 계산한다. 또, 그 스폰서가 그룹을 떠나면, 트리의 모든 정보가 바뀌어져야 한다. 본 논문에서 제안하는 프로토콜은 여러 명의 스폰서를 두었고, 각 멤버의 비밀정보가 그룹스폰서에게 알려지지 않으므로 그룹스폰서가 떠났을 때, 키 갱신이 Lee *et al.*의 프로토콜보다 훨씬 효율적이다. 그러므로, 제안된 프로토콜은 동적 피어 그룹에 적합하다.

ABSTRACT

As a result of the increased popularity of group oriented applications, the design of an efficient authenticated group key agreement protocol has received a lot of attention. Lee *et al.* proposed a tree-based group key agreement protocol, which applies a ternary key tree structure and pairing-based cryptography to the key agreement of Dynamic Peer Group. In their protocol, only the group sponsor knows all member's session random keys computes all blinded keys. In addition, when the group sponsor leaves a group, all nodes of the tree should be changed. In this paper, we present the modified protocol that has several sponsors. Since a secret value for each member isn't given to the group sponsor, the key renewing of our protocol is more secure and efficient than that of Lee *et al.*'s protocol in the previous case. Therefore, our protocol is suitable to Dynamic Peer Groups.

Keywords: Ternary Tree, Bilinear Pairings

I. Introduction

Background. In many modern collaborative and distributed applications such as multicast communication, audio-video conference and collaboration tools, scalable and reliable group communication is a primary concern. An authenticated group key agreement (AGKA) protocol allows a group of authenticated users to share a key, which may later be used to achieve some cryptographic

접수일(2012년 7월 5일), 수정일(2012년 9월 10일), 게재확정일(2012년 10월 30일)
† 주저자, brtcloud@naver.com
‡ 교신저자, brtcloud@naver.com

goals. In order to provide authentication, certificatebased systems or ID-based systems are commonly used. In a typical certificatebased system, a user should obtain a certificate of a long-lived public key from the certifying authority and the entities must verify the certificate of the user before using the public key of a user. Whereas in an ID-based system, the partner just has to know the public identity of the user such as an e-mail address. Many ID-based cryptosystem schemes using bilinear pairing have been proposed[1]. Joux proposed a single round tripartite key agreement using Weil and Tate pairings but unauthenticated[2]. Another direction of research on group key agreement is to handle membership changes in the Dynamic Peer Group(DPG), in which the communicating party can be frequently changed. A group key agreement scheme in a dynamic group must ensure that the session key is updated upon every membership change so that subsequent communication sessions are protected from leaving members and previous communication sessions are protected from joining members. Although this can be achieved by running any authenticated group key agreement protocol whenever group membership changes, alternative approaches to handle this dynamic membership more effectively would be clearly preferable in order to minimize cost of the key renewing operations associated with group updates. An arrangement of participants for key agreement is to consider tree-based setting which requires $\log n$ rounds and has some computational advantages. There have been quite a number of tree based key agreement protocols[4-10].

Related Work. Many ID-based group key agreement protocols which use the one-way function trees and pairing are proposed [7-10]. Kim *et al.* proposed a secure, simple and efficient key management method, called TGDH (Tree-based Group Diffie-Hellman) protocol[4], which uses a binary key tree with Diffie-Hellman key exchange to efficiently compute and update group keys[3]. The computation cost of tree-based key management is proportional to the height of configured key tree. Lee *et al.* extended TGDH using a ternary key tree and Joux's one round authenticated tripartite protocol[7]. Lee *et al.*'s protocol could reduce the computation cost $O(\log_2 n)$ of TGDH to $O(\log_3 n)$. But, Lee *et al.*'s protocol has several operating flaws. First, it is dangerous the group sponsor to know session random keys of all members. When the group sponsor leaves a group, all nodes of the tree should be changed completely. Second, in a distributed network, it is not suitable for a group sponsor to be solely responsible of computing all key values and broadcasting them. Finally, Lee *et al.*'s protocol does not have authentication process of any kind. Wu *et al.* added this process to TGDH protocol using timestamping[10]. However, the group sponsor of Wu *et al.*'s protocol has the ability to learn the session random key of each member as well. Phan *et al.*'s pointed out that Lee *et al.*'s protocol is weak against insider or outsider attack[14]. Because there is no verification process for member authentication and message integrity.

Contribution. In this paper, we modify Lee *et al.*'s protocol so that the group sponsor can not learn the session random keys of members. Thus, the group sponsor can not impersonate any member to other members. Even if the group sponsor leaves a group, the next group sponsor does not need to obtain new session random keys and can reuse the session random keys of remaining members. Therefore, the communication and computation costs of the key renewing can be reduced significantly. In addition, our protocol distributes the loads

among its members and performs an implicit authentication process to protect the man-in-the-middle attack.

II. Bilinear Pairings

Let G_1 be an additive group generated by P, whose order is a prime q, and G_2 be a multiplicative group of the same order q. We assume that the discrete logarithm problem (DLP) in both G_1 and G_2 is hard. Let $e: G_1 \times G_1 \rightarrow G_2$ be a pairing which satisfies the following conditions:

1. Bilinear: $e(P_1+P_2, Q) = e(P_1, Q)e(P_2, Q)$, $e(P, Q_1+Q_2) = e(P, Q_1)e(P, Q_2)$ and $e(aP, bQ) = e(P, Q)^{ab}$

2. Non-degenerate: The map does not send all pairs in $G_1 \times G_1$ to the identity in G_2. Observe that since G_1, G_2 are groups of prime order this implies that if P is a generator of G_1 then $e(P,P)$ is a generator of G_2.

3. Computability: There is an efficient algorithm to compute $e(P,Q)$ for all $P, Q \in G_1$.

BDH Problem : The Bilinear Diffie-Hellman(BDH) Problem for a bilinear map $e: G_1 \times G_1 \rightarrow G_2$ is defined as follows: given P, aP, bP, $cP \in G_1$, compute $e(P,P)^{abc}$,

〔Table 1〕 Notations

n	Number of group members
M_i	i-th group member
h	The height of the key tree
$\langle l, v \rangle$	v-th node at the level l-th in a tree
r_i	M_i's session random number
$K_{<l,v>}$	Key of node $\langle l, v \rangle$
k_i^*	Set of keys of nodes on his key-path of M_i
$BK_{<l,v>}$	Blinded key of node $\langle l, v \rangle$
BK_i^*	Set of M_i's blinded keys
$H()$	Hash function, $H : \{0,1\}^* \rightarrow G_1$
$H_1()$	Hash function, $H_1 : G_1 \rightarrow Z_q^*$

where a, b, c are randomly chosen from Z_q^*. An algorithm A is said to solve the BDH problem with an advantage of ϵ if

$$\Pr[A(P, aP, bP, cP) = e(P,P)^{abc}] \geq \epsilon$$

BDH Assumption : We assume that the BDH problem is hard, which means there is no polynomial algorithm to solve BDH problem with non-negligible probability.

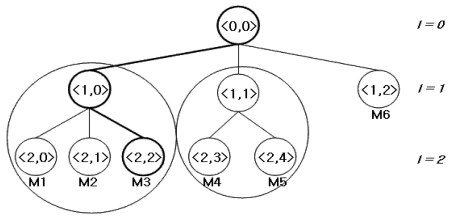
III. Lee et al.'s Tree-based Group Diffie-Hellman protocol

Lee *et al.* proposed a tree-based group key agreement protocol which employs the concept of pairing to achieve ID-based key establishment and a ternary tree structure to apply Joux's one round authenticated tripartite protocol〔2〕〔7〕. Each member contributes a secret value to establish the group session key.

〔Table 1〕 shows the notations used.

The root is located at the *0-th* level and the lowest leaves are at the *h-th* level. The node are denoted $\langle l, v \rangle$, where $0 \leq v \leq 3^{l-1}$. Each node $\langle l, v \rangle$ is associated with *the key* $K_{<l,v>}$ and *the blinded key(bkey)* $BK_{<l,v>} = K_{<l,v>}P$.

The multiplication kP is obtained by repeating k times addition over an elliptic curve. For the leaf node, the $K_{<l,v>} = r_i$ and $BK_{<l,v>} = r_iP$, where r_i is the session random key selected by the member $M_i (1 \leq i \leq n)$. The group sponsor, the member of rightmost leaf node of the tree, is responsible for computing all the keys of the tree and broadcasting all the *bkeys* of the tree to the members. In order to establish the key agreement, at first, the group sponsor obtains the session random key r_i from M_i. After computing the keys and the *bkeys*, the group sponsor broadcasts the *bkeys*.



〔Fig. 1〕 An Example of a Key Tree (h=2, n=6)

Using the *bkey*s, M_i computes every key along the path from $\langle l, v\rangle$ to $\langle 0, 0\rangle$, referred to as the *key-path*.

〔Fig. 1〕 shows an example of a key tree.

In 〔Fig. 1〕, for example, the bold lines mean the *key-path* of M_3, and denote $k_3^* = \{K_{<2,2>}, K_{<1,0>}, K_{<0,0>}\}$ and $BK_3^* = \{BK_{<2,2>}, BK_{<1,0>}, BK_{<0,0>}\}$. M_6, the member of right-most leaf node of the tree, becomes a group sponsor. M_6 obtains $r_i (1 \le i \le 6)$ from each member and then computes all values of the tree as follows:

$$
\begin{aligned}
&K_{<2,0>} = r_1,\ BK_{<2,0>} = r_1 P,\\
&K_{<2,1>} = r_2,\ BK_{<2,1>} = r_2 P,\\
&K_{<2,2>} = r_3,\ BK_{<2,2>} = r_3 P,\\
&K_{<2,3>} = r_4,\ BK_{<2,3>} = r_4 P,\\
&K_{<2,4>} = r_5,\ BK_{<2,4>} = r_5 P,\\
&K_{<1,2>} = r_6,\ BK_{<1,2>} = r_6 P,\\
&K_{<1,0>} = H_1(e(r_1 P, r_2 P)^{r_3}),\\
&BK_{<1,0>} = H_1(e(r_1 P, r_2 P)^{r_3}) P,\\
&K_{<1,1>} = H_1(r_4 r_5 P),\ BK_{<1,1>} = H_1(r_4 r_5 P) P.
\end{aligned}
$$

And then broadcasts all the *bkey*s of the tree to members. Using the *bkey*s of the tree, finally, $M_i (1 \le i \le 6)$ computes every key along his *key-path*. For example, the group session key computed respectively by M_3, M_4 and M_6 are

$$
\begin{aligned}
&K_{<0,0>} = H_1(e(H_1(r_4 r_5 P) P, r_6 P)^{H_1(e(r_1 P, r_2 P)^{r_3})}),\\
&K_{<0,0>} = H_1(e(H_1(e(r_1 P, r_2 P)^{r_3}) P, r_6 P)^{H_1(r_4 r_5 P)}),\\
&K_{<0,0>} = H_1(e(H_1(e(r_1 P, r_2 P)^{r_3}) P, H_1(r_4 r_5 P) P)^{r_6}).
\end{aligned}
$$

Lee *et al.*'s protocol has several operating flaws.

First, it is dangerous the group sponsor to know session random keys $\{r_i\}$ of all members. In the worst case, when the group sponsor leaves a group, all nodes of the tree should be changed completely. Thus, the next group sponsor should obtain new session random keys $\{r_i\}$ from all members to generate the new key tree and all members should also compute all the keys. Second, in a distributed network, it is not suitable for a group sponsor to be solely responsible of computing all key values and broadcasting them. It requires the group sponsor to have heavy loads and high trust value. Finally, Lee *et al.*'s protocol does not have authentication process of any kind.

IV. Our Protocol

Any member of a group can be a group sponsor in DPG and a group sponsor can leave a group freely. So, it is not desirable that a group sponsor learns the session random keys $\{r_i\}$ of all members. We modify Lee *et al.*'s protocol so that the group sponsor can not learn $\{r_i\}$. In addition, our protocol performs an implicit authentication process and distributes the loads among the members.

4.1 System Setup

The Private Key Generator (PKG) chooses the following system parameters:

$$\{p,\ q,\ P,\ e,\ G_1,\ G_2,\ H(),\ H_1()\}.$$

1. Select an elliptic curve E defined over $GF(p)$ with order q and a base point P, then make them public.

2. Choose a master key $s \in Z_q^*$ and compute P_{pub} by $P_{pub} = sP$.

3. Publish system parameters.

4. Each member of group submits his identity id to PKG. His public key is $Q_i = H(id_i)$.

PKG computes his private key $S_i = sQ_i$.

5. Finally, PKG sends S_i to each member of group via a secure channel.

By the above step, PKG generates the private keys of group members.

4.2 Member Authentication

Any of group members can become a group sponsor. To be a group sponsor is not related with the location of the node. But, for the sake of further distributed network, it is desirable that the member has the lowest computational cost in obtaining the group session key becomes a group sponsor.

If a group sponsor is decided, he performs an implicit authentication process. The member M_i coming into the group chooses a session random number $r_i \in Z_q^*$ and computes $X_i = r_iP$ and $Y_i = H_1(X_i)S_i + r_iP_{pub}$.

M_i broadcasts $\{id_i, X_i, Y_i\}$ to the group. The group sponsor verifies

$$e(Y_i, P) = e(H_1(X_i)sQ_i + r_isP, P)$$
$$= e(H_1(X_i)Q_i + X_i, P_{pub}).$$

If the equation holds〔11-13〕, then the group sponsor believes the message was sent from member M_i.

The group sponsor collects $\{r_iP\}$ from qualified members and then broadcasts them with the information concerning the location of members in key tree. Every group member can keep watch on the action of the group sponsor by checking on $\{r_iP\}$.

Note that unlike Lee *et al.*′ protocol, the group sponsor doesn′t know $\{r_i\}$ of all members.

4.3 Key Generation

There are several subgroups in key tree(if $n > 3$). Anyone of a subgroup can become a subgroup sponsor. The subgroup sponsor computes the *bkey* of each subgroup using published $\{X_i\}$. And then broadcasts the value to different subgroup members. Thus, the sibling member of the subgroup sponsor can check on the value.

In 〔Fig.1〕, assume M_6 as the group sponsor and M_1, M_5 and M_6 as the subgroup sponsors. After the authentication process of $M_i (1 \le i \le 6)$, M_6 broadcasts $\{X_i\}$.

(i.e.{ $BK_{<2,0>}, BK_{<2,1>}, BK_{<2,2>}, BK_{<2,3>}$, $BK_{<2,4>}, BK_{<1,2>}$ })

M_1 computes the *bkey* of the subgroup $BK_{<1,0>} = K_{<1,0>}P = H(e(r_2P, r_3P)^{r_1})P$ and then broadcasts it. M_2 and M_3 can check on it.

M_5 computes the *bkey* of the subgroup $BK_{<1,1>} = K_{<1,1>}P = H(r_4r_5P)P$ and then broadcasts it. M_4 can check on it.

M_6 does not need to broadcast the *bkey* of the subgroup, because r_6P has been already published.

Using the *bkey*s of the tree, finally, M_i computes every key along his *key-path* and obtains $K_{<0,0>}$.

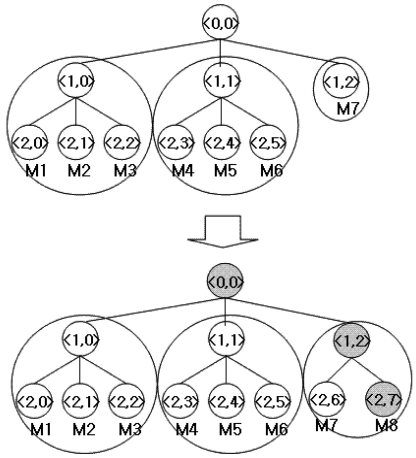
As a result, each group member has the group session key of the same type with Lee *et al.*′s protocol.

V. Group Membership Operations

5.1 Join Protocol

The joining process is described as follows:

1. We assume the group has n members. The new member M_{n+1} broadcasts the join request including id_n, X_{n+1} and Y_{n+1}. Upon receiving the join request, the group



〔Fig. 2〕 Tree-updating in Join Operation

sponsor performs the authentication process. If M_{n+1} is qualified, the group sponsor generates the leaf node nearest the root node for M_{n+1}. The height of the tree should not be increased as possible.

2. The group sponsor broadcasts $\{X_i\}(1 \leq i \leq n+1)$ and the location information of the new member.

3. The subgroup sponsors compute the subgroup *bkey*s and broadcast them.

4. Upon receiving $\{X_i\}$ and the subgroup *bkey*s, each group member computes every key along his *key-path* and then obtains the new group session key.

〔Fig. 2〕 illustrates an example of member M_8 joining a group where M_1 is the group sponsor and M_3, M_5 and M_7 are the subgroup sponsors. M_8 broadcasts the join request including $\{id_8, X_8, Y_8\}$ to the group. M_1 performs the authentication process. If M_8 is qualified, M_1 renames $BK_{<1,2>}$ to $BK_{<2,6>}$ and generates $BK_{<2,7>} = X_8 = r_8 P$. And then, broadcasts the *bkey*s $\{r_i P\}$ $(1 \leq i \leq 8)$ of all leaf nodes. The subgroup sponsors M_3, M_5 and M_7 broadcast $BK_{<1,0>}$, $BK_{<1,1>}$ and $BK_{<1,2>}$, respectively. Each group member computes $K_{<0,0>}$.

5.2 Leave Protocol

When a member wants to leave a group, the group session key has to be renewed to prevent the leaving member from acquiring any information about future communication.

The property of forward security has to be assured. The leaving process is described as follows:

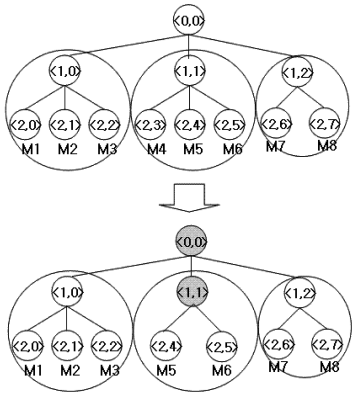
1. We assume that the group has n members and the member M_d wants to leave the group. The leaving member M_d broadcasts the leave request. After receiving the leave request, the group members delete the leaf node of M_d. Then, the relevant parent node with only one child node should be also deleted.

2. The subgroup sponsor related to M_d computes the updated subgroup *bkey*s and broadcasts them.

Note that unlike Lee *et al.*′ protocol, the group sponsor doesn′t need to broadcast all *bkey*s again.

3. Upon receiving the updated subgroup *bkey*s, each group member computes every key along his *key-path* and then obtains the new group session key.

〔Fig. 3〕 illustrates an example of member M_4 leaving the group where M_1 is the group sponsor and M_3, M_5 and M_7 are the subgroup



〔Fig. 3〕 Tree-updating in Leave Operation

sponsors. M_4 broadcasts the leave request to the group. Every member deletes the leaf node 〈2, 3〉. M_5 updates $BK_{<1,1>}$ to $H_1(r_5 r_6 P)P$ using $BK_{<2,5>} = X_6 = r_6 P$. And then broadcasts it. Each group member computes $K_{<0,0>}$.

In Lee *et al.*'s protocol, when the group sponsor knowing r_i of all members leaves a group, the new group sponsor receives new $\{r_i P\}$ from remaining members and computes all *bkey*s of updated key tree and broadcasts them. But, in our protocol, $\{r_i P\}$ from remaining members can be reused and the only subgroup sponsor related to the previous group sponsor computes the new subgroup *bkey*s and broadcasts them.

5.3 Merge Protocol

We now describe the merge protocol for two groups.

1. We assume that it is natural to merge the smaller group onto the larger one. The smaller group sponsor broadcasts the merge request including $\{id_i, X_i, Y_i\}$ of own group members and tree structure to the larger group. Upon receiving the request, the larger group sponsor checks if the following equation holds[11-13]:

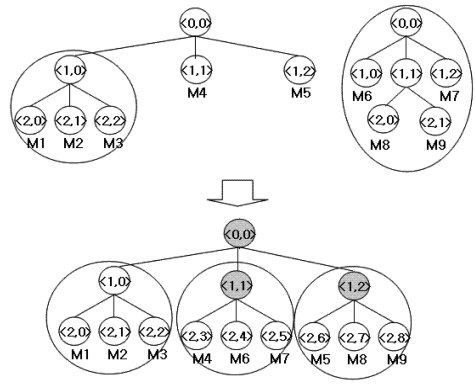
$$e(\sum_i Y_i, P) = e(\sum_i (H_1(X_i)Q_i + X_i), P_{pub}).$$

If all members are qualified, the larger group sponsor generates the leaf nodes.

2. The group sponsor broadcasts $\{X_i\}$ of all members of two group including the information of the new tree structure.

3. The subgroup sponsors of updated subtree compute the new subgroup *bkey*s and broadcast them.

4. Upon receiving $\{X_i\}$ of all members and the subgroup *bkey*s, each group member computes every key along his *key-path* and



〔Fig. 4〕 Tree-updating in Merge Operation

then obtains the new group session key.

As a result, merge protocol is similar to join protocol. To reduce the height of the tree, the tree structure of the smaller group may not be kept.

〔Fig. 4〕 shows an example of merging two groups, where the group sponsors are M_5 and M_6. M_6 broadcasts the merge request including $\{id_i, X_i, Y_i\}(6 \le i \le 9)$ and tree structure to the larger group. Then, M_5 performs the authentication process as follows:

$$e(Y_6 + Y_7 + Y_8 + Y_9, P) = e(H_1(X_6)Q_6 + H_1(X_7)Q_7 + H_1(X_8)Q_8 + H_1(X_9)Q_9 + X_6 + X_7 + X_8 + X_9, P_{pub}).$$

If the equation holds, M_5 generates the leaf nodes. To reduce the height of the tree, the smaller group members are divided into two different subgroups. M_5 renames $BK_{<1,1>}$ and $BK_{<1,2>}$ to $BK_{<2,3>}$ and $BK_{<2,6>}$. And then, generates the leaf nodes for M_6, M_7, M_8 and M_9 and broadcasts the *bkey*s $\{r_i P\}$ $(1 \le i \le 9)$ of all leaf nodes. The subgroup sponsors M_4 and M_5 compute new

$$BK_{<1,1>} = H_1(e(r_6 P, r_7 P)^{r_4})P \quad \text{and}$$
$$BK_{<1,2>} = H_1(e(r_8 P, r_9 P)^{r_5})P$$

and broadcast them, respectively. Each group member computes $K_{<0,0>}$.

5.4 Partition Protocol

We now describe the partition protocol.

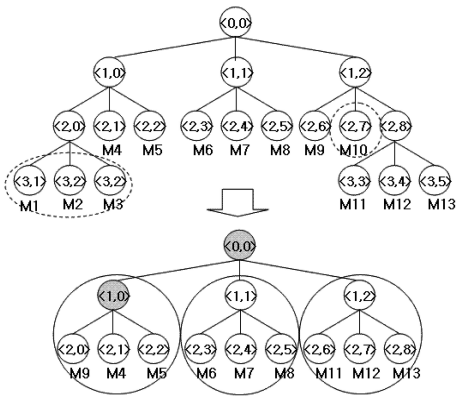
1. The leaving members broadcast the leave requests. After receiving the leave requests, the group members delete the leaf nodes of leaving members.

2. The subgroup sponsors related to leaving members compute the new subgroup *bkey*s and broadcast them. Note that the group sponsor doesn't need to broadcast all *bkey*s again.

3. Upon receiving the updated subgroup *bkey*s, each group member computes every key along his *key-path* and then obtains the new group session key.

This event appears as a simultaneous leaving of multiple members. As a result, partition protocol is similar to leave protocol. Like merge protocol, the height of the tree should be reduced as possible.

〔Fig. 5〕 shows an example of partitioning the group where M_8 is the group sponsor. After M_1, M_2, M_3 and M_{10} broadcast the leave requests and leave the group, all remaining members delete all nodes of leaving members and the node 〈2, 0〉. To reduce the height of the tree, M_9 moves to other subgroup and the node 〈1, 2〉 is deleted. Remaining members rename $BK_{<2,6>}$, $BK_{<2,8>}$, $BK_{<3,3>}$, $BK_{<3,4>}$ and $BK_{<3,5>}$ to $BK_{<2,0>}$, $BK_{<1,2>}$, $BK_{<2,6>}$, $BK_{<2,7>}$ and $BK_{<2,8>}$. The subgroup sponsor M_5 computes $BK_{<1,0>}$ and broadcasts it. Each group member computes $K_{<0,0>}$.



〔Fig. 5〕 Tree-updating in Partition Operation

VI. Discussion

6.1 Security

The proposed key agreement protocol with authentication satisfies the following security properties:

Group key secrecy: means that even an adversary who knows all *bkey*s can not derive the group key. If an adversary doesn't know the session random key r_i of current member, he can not compute the group key with only all *bkey*s in our protocol. To extract r_i from the given r_iP is computationally impossible. It is equivalent to solve the Elliptic Curve Discrete Logarithm Problem.

Known-group key secrecy: means that knowledge of previous group keys will not enable an adversary to know other group keys. If the session is changed after membership operation, *bkey*s and group key are also changed. If an adversary doesn't know r_i of current member, he can't compute current group key. The knowledge of previous group keys cannot deduce the future group keys in our protocol.

No key-compromise impersonation: If the long term private key of a member is compromised, the adversary can impersonate the member in this protocol. But he cannot impersonate other members.

No unknown key-share: If an adversary convinces a group of members that they share a key with another member, while in fact they share the key with the adversary, we call the protocol suffering from unknown key share attack. The group sponsor can

accept $\{r_xP\}$ from only members to have the long term private key S_i by member authentication process. Though an adversary who doesn't have S_i generates the session random key r_x, the group sponsor can not accept r_xP and r_xP can not be published. Our protocol can withstand the man-in-the-middle attacks.

No Key control: It is not possible for any of the entities or the adversary to force the group key to be a pre-selected value or predict the value of the group key. All members in the group negotiate collaboratively the group key and neither member can control the outcome of the negotiation. No one can force the group key to a preselected value.

Backward and forward secrecy: Backward secrecy means that a new member who knows the current group key cannot derive any previous group key. A new member picks session random key r_i and broadcasts r_iP. Once he receives all *bkey*s on his *key-path*, he can compute current group key. Cleary, all these keys will contain a new joiner's contribution r_i. Hence, they are independent of previous group keys. Therefore, he can't derive any previous group keys.

The group key that the leaving member knows must be renewed to provide forward secrecy. When a member leaves a group, all *bkey*s that contain his session random key r_i are deleted from key-tree and current group key is also changed. Thus, he knowing previous group key cannot derive current group key. There are the properties of backward and forward secrecy in our protocol.

6.2 Complexity

While TGDH uses a binary tree, Lee *et al.*'s protocol uses a ternary tree. Using a ternary tree can reduce the computation cost $O(\log_2 n)$ to $O(\log_3 n)$. Since we modify Lee *et al.*'s protocol, our protocol has the group session key of the same type with Lee *et al.*'s.

〔Table 2〕 summarizes the communication and computation costs of Lee *et al.*'s and our protocol except the system setup and the authentication process. The number of remaining group members after membership

〔Table 2〕 Communication and Computation Costs

a. When the member not a group sponsor joins and leaves a group

		Communication	Computation	
		Rounds	Pairings	Multiplications
Lee et al.'s protocol	Join	2	$\lceil \log_3 n \rceil - 1$	$\lceil \log_3 n \rceil + 1$
	Leave	1	$\lceil \log_3 n \rceil - 1$	$\lceil \log_3 n \rceil + 1$
Our protocol	Join	3	$\lceil \log_3 n \rceil - 1$	$\lceil \log_3 n \rceil + 1$
	Leave	1	$\lceil \log_3 n \rceil - 1$	$\lceil \log_3 n \rceil + 1$

b. When the group sponsor leaves a group

	Communication		Computation	
	Rounds	Massages	Pairings	Multiplications
Lee et al.'s protocol	2	n+1	$n(\lceil \log_3 n \rceil) - 2$	$\sum_{k=1}^{h} 3^k$
Our protocol	1	1	$\lceil \log_3 n \rceil - 1$	$\lceil \log_3 n \rceil + 1$

operation are denoted by n. The load of protocol depends on the height of tree, the balance of the key-tree and the location of the joining and the leaving nodes. In our analysis, we assume the configuration that the levels are full of nodes in the key tree with the height h and list the total cost of the group for Lee *et al.*'s protocol and our protocol.

When the member not a group sponsor joins and leaves a group, the computation cost of our protocol is equal to that of Lee *et al.*'s, while the communication cost of our protocol is a little more expensive than that of Lee *et al.*'s. The reason is that our protocol has a group sponsor broadcasting *bkey*s of leaf nodes and several subgroup sponsors broadcasting *bkey*s of internal nodes, while Lee *et al.*'s protocol has only one group sponsor broadcasting all *bkey*s.

On the one hand, when the group sponsor leaves a group, the next group sponsor of Lee *et al.*'s should obtain the new session random keys from remaining members to update all nodes of the tree. However, in this case, the next group sponsor of our protocol doesn't need to. The reason is that unlike Lee *et al.*'s, the previous group sponsor of our protocol does not know the session random keys $\{r_i\}$ of remaining members except his session random key. He knows only $\{r_iP\}$ of members. Thus, The only subtree related to the group sponsor is changed and $\{r_iP\}$ of remaining members can be reused. The only subgroup sponsor related to the previous group sponsor broadcasts *bkey*s of the updated subtree. Therefore, communication and computation costs of the key renewing can be reduced significantly in this event.

VII. Conclusion

This paper modified Lee *et al.*'s protocol for distributive, secure and efficient networks. Our protocol distributes the loads among the members and performs an implicit authentication function and has the group sponsor unknowing the session random keys of members. When the group sponsor leaves a group, the communication and computation costs of the key renewing can be reduced significantly while preserving the security properties. Therefore, our protocol is suitable to Dynamic Peer Groups.

References

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

[2] A. Joux, "A one round protocol for tripartite Diffie-Hellman," Proc. Algorithmic Number Theory Symposium-ANTS IV LNCS 1838 pp. 385-394, 2000

[3] W. Diffie and M. E. Hellman, "New directions in cryptography," IEEE Trans. Inf. Theory 22 pp. 644-654, 1976

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

[5] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communication using key graphs," IEEE/ACM Trans. Net. 8 pp. 16-29, 2000

[6] Y. Kim, A. Perrig, and G. Tsudik, "Communication-efficient group key agreement," Proc. IFIP SEC pp. 229- 244, 2001

[7] S. Lee, Y. Kim, K. Kim, and D.-H. Ryu, "An efficient tree-based group key agreement using bilinear map," Applied

Cryptography and Network Security 2846 pp.357-371, 2003
〔8〕 R. Dutta, R. Barua, and P. Sarkar, "Provably secure authenticated tree based group key agreement," ICICS LNCS 3269 pp. 92-104, 2004
〔9〕 R. Dutta and R. Barua, "Dynamic group key agreement in tree-based setting," Proc. ACISP LNCS 3574 pp. 101-112, 2005
〔10〕 S. Wu, J. Chiu, and B. Chieu, "Identity-based key agreement for peer group communication from pairings," IEICE Trans. fundamentals E88-A, 2005
〔11〕 K. Choi, J. Hwang, and D. Lee, "Efficient ID-based group key agreement with bilinear maps," International Workshop on Practice and Theory in Public Key Crpytography LNCS 2947 pp. 130-144, 2004
〔12〕 X. Du, Yi. Wang, J. Ge, and Yu. Wang, "ID-based authenticated two round multi-party key agreement," IACR eprint 2003-247, 2003
〔13〕 X. Du, Yi. Wang, J. Ge, and Yu. Wang, "An improved ID-based authenticated group key agreement scheme", IACR eprint 2003-260, 2003
〔14〕 R. C-W. Phan and B-M. Goi, "(In)Security of efficient Tree-based group key agreement using Bilinear Map", IEEE/IFIP International Conference on Embeded and Ubiquitous Computing, EUC'08 pp. 443-446, 2008

〈著者紹介〉

김 호 희 (Ho-hee Kim) 학생회원
1993년 2월: 경북대학교 전자공학과 졸업
1996년 2월: 경북대학교 전자공학과 석사
2003년 3월~현재: 경북대학교 전자공학과 박사과정
〈관심분야〉 정보보호, 전자공학

사 진

김 순 자 (Soon-ja Kim) 종신회원
1975년 2월: 경북대학교 수학교육과 졸업
1977년 2월: 경북대학교 수학교육과 석사
1988년 2월: 계명대학교 이학박사
1980년~현재: 경북대학교 전자공학부 교수
〈관심분야〉 정보보호 응용기술, 전자상거래 및 보안