Journal of Theoretical and Applied Information Technology
20th March 2013. Vol. 49 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A KEY AGREEMENT FOR LARGE GROUP USING BILINEAR MAPS

[1]TZU-CHUN LIN, [2]TE-YU CHEN, [3]CHIUN-SHIANG GAU, [4*]MIN-SHIANG HWANG

[1]Assoc Prof., Department of Applied Mathematics, Feng Chia University
[2]Asstt. Prof., Department of Elder Care, National Tainan Institute of Nursing
[3]Department of Applied Mathematics, Feng Chia University
[4]Chair Prof., Department of Computer Science and Information Engineering, Asia University
E-mail: [1]lintc@fcu.edu.tw, [2]chendytw@gmail.com ,[3]therock0331@hotmail.com, [4]mshwang@asia.edu.tw

*Corresponding author

ABSTRACT

A key agreement scheme for large dynamic multicast group systems has been designed to cope with such applications including pay-tv, teleconferencing, collaborative work, online games, and so forth. To avoid heavy loads and unauthorized accesses to the system, it is necessary to construct an efficient and secure scheme. Some recently released schemes have tree-based underlying structures. The efficiency of these tree-based schemes is closely associated with the height of the underlying tree. In this paper, we propose a secure and efficient key agreement scheme for large groups that adopts a quad tree as the underlying key tree.

Keywords: *Multicast, Tree-Based Group, Key Agreement, Elliptic Curve, Bilinear Mapping*

1. INTRODUCTION

Due to the rapid development of computer technologies, group communication has become a become a main focus. Generally speaking, group communication applications include video conferencing, online games or videos, military command transmission, etc. A common feature among these applications is that the members of the group may join or leave at any time as their wish. In order to relieve the burden on the network bandwidth, multicasting techniques, have been adopted in a wide variety of applications. However, In order to prevent unauthorized accesses to the messages during transmission, it is necessary to encrypt the messages. Therefore, the mechanism in which the authorized members efficiently and securely agree on a group key has been the major consideration in these applications [1, 2, 9-12].

A secure key agreement protocol for the large dynamic groups should meet the following requirements [20]:
1. Group key secrecy: It guarantees that the group keys used to encrypt the broadcast messages must not be compromised by a nonmember.
2. Backward secrecy: It guarantees that the previously used group keys must not be compromised by new group members.
3. Forward secrecy: It guarantees that the new group keys must not be compromised by former group members.

The rest of this paper is organized as follows. Section 2 reviews some related works. Then, in Section 3, the details of the proposed scheme are presented. In Section 4, the security of our scheme is analyzed, and the comparisons among our new work and some other schemes are also illustrated. Finally, the conclusions are drawn in Section 5.

2. RELATED WORK

To date, many researchers have offered their key management schemes for secure multicast [3, 5, 7, 13, 15-25]. These schemes can be classified into three categories [14]: ***the centralized architecture***, ***the decentralized architecture***, and ***the distributed architecture***. Recently, some researches have focused on the tree based hierarchy where a logical tree of keys is maintained for the reason of efficiency [7, 10, 17, 18, 20, 21].

Kim et al. proposed a distributed group key agreement protocol and named it the Tree-based Group Diffie-Hellman (TGDH) [8]. They combined a binary key tree with the Diffie-Hellman key agreement [4]. That is, the logical key tree adopted in this protocol is binary. Members in the group can

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

deduce the upper level keys by using the Diffie-Hellman key agreement protocol.

In 2003, Lee et al. proposed an efficient tree-based group key agreement using bilinear map [10]. This distributed scheme extends from the one round protocol for tripartite Diffie-Hellman [6]. The logical key tree adopted in their system is ternary. The stretch of the underlying key tree from the binary form to ternion results in the reduction of the computation complexity from $O(\log_2 n)$ to $O(\log_3 n)$.

In 2005, Liming Wang and Chuan-KunWu [20] proposed a decentralized key agreement scheme. They adopted an identity tree instead of the key tree and thereby turned the scheme identity-based. In their scheme, the whole group is divided into some smaller subgroups maintained by a subgroup controller. Nam et al. developed a contributory group key agreement protocol in the same year [13]. In their scheme, all the users of the network are divided into two groups according to the computational capabilities of users.

3. THE PROPOSED SCHEME

In this section, we shall elaborate how our new scheme handles the membership operations as well as how dynamic group communication can be smoothly carried out.

3.1 Group Membership Operations

A good group key agreement system must be able to handle adjustments in a simple, efficient, and secure manner. Adjustments are taken place when members attempt to join (Member Join) or leave (Member Leave) the group. In the proposed scheme, a quad tree is adopted as the underlying key tree. Each individual member in the system is assigned to a leaf node of the key tree, and each node of the key tree is associated with a key which is shared by all the members of the subtree rooted at this node. A member can use his /her own private secret and some public information to derive the keys on the path from its leaf node to the root, which is also called ***the key path***. When a Member Join event happens, a new node will be inserted into the key tree, and the new member is assigned to the new node. The keys on the key path of the new node should be refreshed for assuring backward secrecy. When a Member Leave event happens, the node assigned to the leaving member should be deleted form the tree, and the keys on the key path of the deleted node should be refreshed for assuring forward secrecy. All communication channels are considered public but authentic in the proposed protocol.

3.2 System Setup

The system parameters (G_1, G_2, e, P, H_1, H_2) are generated and published, where G_1 is an additive cyclic elliptic curve group generated by P with a prime order q, G_2 is a multiplicative group with the same order q, e: $G_1 \times G_1 \rightarrow G_2$ is an bilinear map, and H_1: $G_1 \rightarrow Z_q^*$, H_2: $G_2 \rightarrow Z_q^*$ are the hash functions.

The nodes of the quad key tree adopted in our scheme can be classified into three roles. ***The root node*** is associated with the shared group key which can be computed by all the members in the group. ***The key node*** located at the internal node is associated with two or three keys depending on the number of its sibling nodes and its position. When the number of sibling nodes is smaller than four, there will be two keys: one is *the blinded key* for the group that goes public, and the other is *the key generation key* which can be computed by all the members in the subtree rooted at this key node. For a node with four child nodes, *a union blinded key* is assigned to its rightmost child node additionally. ***The member node*** represents each group member as a leaf node. The keys that come with the member nodes are similar to those of the key node except that the key generation key is randomly chosen from Z_q^*. Figure 1 shows an example of the key tree. Table 1 summarizes the notations used in the proposed scheme.

Table 1: The Notations Used In The Proposed Scheme

U_C	The *c-th* group member, $c \in \{1, 2, \ldots, n\}$
N^i_j	The *i-th* node at the *j-th* level of a key tree
K^i_j	The key generation key of the node N^i_j
BK^i_j	The blinded key of the node
UBK^k_j	The union blinded key of the *k-th* team group with four nodes at the *j-th* level

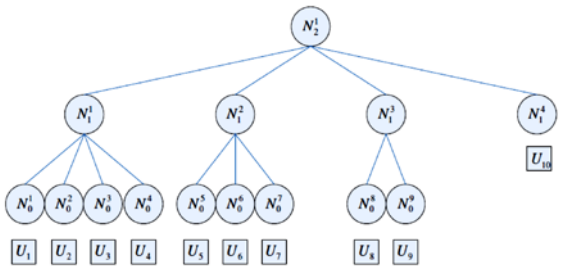

Figure 1: An Example Of The Quad Key Tree

For the key node N^i_j, the key generation key K^i_j is defined according to the following three cases:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$K^i_j = \begin{cases} H_1(H_2(e(P,P)^{K^{4i-3}_{j-1}K^{4i-2}_{j-1}K^{4i-1}_{j-1}})K^{4i}_{j-1}P), \\ \quad \text{if } N^i_j \text{has four child node;} \\ H_2(e(P,P)^{K^{4i-3}_{j-1}K^{4i-2}_{j-1}K^{4i-1}_{j-1}}), \\ \quad \text{if } N^i_j \text{has three child node;} \\ H_3\left(K^{4i-3}_{j-1}K^{4i-2}_{j-1}P\right), \\ \quad \text{if } N^i_j \text{ has two child node;} \end{cases}$$

The blinded key BK^i_j for a node N^i_j is K^i_jP. In the case where a node has four sibling nodes, a union blinded key is assigned to the rightmost node of the four sibling nodes additionally, and the following computation is done:

$$UBK^i_j = H_2(e(P,P)^{K^{i-3}_j K^{i-2}_j K^{i-1}_j})P$$

If the system is constructed by the previous rules and all the blinded keys and union blinded keys are published, then each member in the group can compute the key generation keys on the key path by using his/her own private key generation key and the public blinded keys or union blinded keys.

For example, as shown in Figure 1, U_2 can derive the key generation key K^1_1 from his private key generation key K^2_0 and the public keys BK^1_0 , BK^3_0 and BK^4_0 as follows.

$$\begin{aligned} &H_1(H_2(e(BK^1_0, BK^3_0)^{K^2_0})BK^4_0) \\ &= H_1(H_2(e(K^1_0P, K^3_0P)^{K^2_0})K^4_0P) \\ &= H_1(H_2(e(P,P)^{K^1_0K^3_0K^2_0})K^4_0P) = K^1_1 \end{aligned}$$

Subsequently, U_2 can go on to compute the group key K^1_2 by using the derived K^1_1 and the public keys BK^2_1, BK^3_1 and BK^4_1.

$$\begin{aligned} &H_1(H_2(e(BK^2_1, BK^3_1)^{K^1_1})BK^4_1) \\ &= H_1(H_2(e(K^2_1P, K^3_1P)^{K^1_1})K^4_1P) \\ &= H_1(H_2(e(P,P)^{K^1_1K^2_1K^3_1})K^4_1P) = K^1_2 \end{aligned}$$

Furthermore, U_7 can compute the key generation key K^2_1 by using his own private key K^7_0 and the public keys BK^5_0 and BK^6_0 as follows.

$$\begin{aligned} &H_2(e(BK^5_0, BK^6_0)^{K^7_0}) \\ &= H_2(e(K^5_0P, K^6_0P)^{K^7_0}) \\ &= H_2(e(P,P)^{K^5_0K^6_0K^7_0}) = K^2_1 \end{aligned}$$

Afterward, U_7 can compute the group key K^1_2 by using the derived K^2_1 and the public keys BK^1_1, BK^2_1 and BK^4_1 the way U_2 does so.

$$\begin{aligned} &H_1(H_2(e(BK^1_1, BK^3_1)^{K^2_1})BK^4_1) \\ &= H_1(H_2(e(K^1_1P, K^3_1P)^{K^2_1})K^4_1P) \\ &= H_1(H_2(e(P,P)^{K^1_1K^2_1K^3_1})K^4_1P) = K^1_2 \end{aligned}$$

Similarly, U_9 can compute the key generation keys K^3_1 and K^1_2 as follows:

$$H_1(K^9_0BK^8_0) = H_1(K^9_0K^8_0P) = K^3_1 ;$$

And

$$\begin{aligned} &H_1(H_2(e(BK^1_1, BK^2_1)^{K^3_1})BK^4_1) \\ &= H_1(H_2(e(K^1_1P, K^2_1P)^{K^3_1})K^4_1P) \\ &= H_1(H_2(e(P,P)^{K^1_1K^2_1K^3_1})K^4_1P) = K^1_2 . \end{aligned}$$

U_{10} can compute the group key K^1_2 from his own private key K^4_1 and the public key UBK^4_1 as follows.

$$\begin{aligned} &H_1(UBK^4_1 \cdot K^4_1) \\ &= H_1(H_2(e(P,P)^{K^1_1K^2_1K^3_1})P \cdot K^4_1) = K^1_2 \end{aligned}$$

3.3 Member Join Protocol

We assume that there are n users, U_1, U_2, … U_n, in the group. If a new member U_{n+1} wants to join in, he/she first randomly chooses a key generation key from Z_q^* and computes the corresponding blinded key. Then, the new member will broadcast a join request message, which includes his/her own blinded key, to all the members in the group. After receiving the request, all the members in the group will agree on the insertion point of the new member as well as the sponsor. The insertion point is the position where the new member will be allocated in the key tree; and all the members in the group will update the key tree accordingly. The sponsor who is one of the old members knowing all the necessary keys will take charge of the key refreshing.

The insertion point of the new member node is decided on the principle of keeping the height of the key tree as low as possible. Hence, the insertion point of the new member node is the first node with a degree that is lower than 4 on the traversal of the key tree from the top level to the bottom level and from left to right. There are two cases should be considered for modifying the key tree and determining the sponsor. First, if the node N^i_j is a key node or the root node, the new member node will be inserted as the rightmost child node of the node N^i_j, and the sponsor is the rightmost child node of the node N^i_j before the new member node is inserted. Second, if the node N^i_j is a member node, then the node N^i_j will become a key node and branch into two member nodes, where the left hand side node is assigned to the old member who was associated with the node N^i_j originally and will be the sponsor, and the right hand side node is assigned to the new member.

According to the preceding rules, all the members in the group will modified their key trees properly. The agreed sponsor who knows all the necessary keys will stand out and be responsible for the key refreshing. After completing the modification on key tree and the key refreshing, the sponsor will broadcast all the refreshed blinded

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

keys and union blinded keys to all the group members including the new member. After receiving the broadcast message, each member in the group can compute all the key generation keys on its key path, and all the legitimate members can achieve the same new group key.

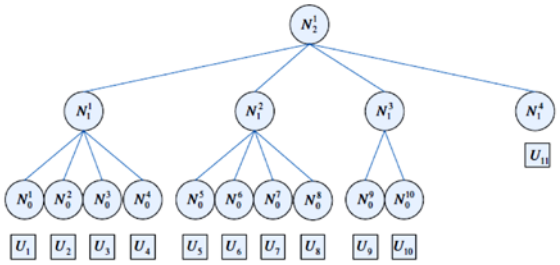

Figure 2: A Quad Key Tree With Eleven Members

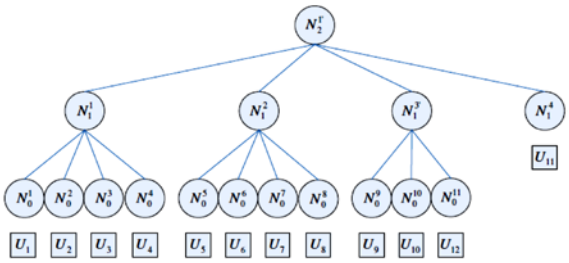

Figure 3: The New Key Tree After Member Join Event

For example, as shown in Figure 2, if a new member U_{12} wants to join the group, he/she has to first randomly choose the key generation key from Z_q^* and then compute the corresponding blinded key, and then he/she broadcasts the join request to the group. After the group receives the Member Join request from the new member U_{12}, a new member node $N^{11}{}_0$ for U_{12} is inserted and the key tree is updated (see Figure 3). The member U_{10} associated with the node $N^{10}{}_0$ is then the sponsor responsible for refreshing the related keys. The member U_{10} now re-computes the new blinded key $BK^3{}_1{}'$ and union blinded key $UBK^4{}_1{}'$ as follows.

$$K^3{}_1{}' = H_2(e(BK^9{}_0, BK^{11}{}_0)^{K^{10}{}_0}) = H_2(e(P, P)^{K^9{}_0 K^{10}{}_0 K^{11}{}_0})$$

$$BK^3{}_1{}' = K^3{}_1{}' P = H_2(e(P, P)^{K^9{}_0 K^{10}{}_0 K^{11}{}_0})P$$

$$UBK^4{}_1{}' = H_2(e(BK^1{}_1, BK^2{}_1)^{K^3{}_1{}'})P = H_2(e(P, P)^{K^1{}_1 K^2{}_1 K^3{}_1{}'})P$$

It is clear that in addition to $UBK^4{}_1$, all the keys on the key path, namely $K^3{}_1$, $K^1{}_2$, and $BK^3{}_1$, are refreshed after the update so that backward secrecy can be accomplished. After finishing the computation, the sponsor U_{10} broadcasts the modified key tree and the updated blinded key as well as the union blinded key to the members of the group. All the legitimate members, including the new member U_{12}, can now compute the group key $K^1{}_2{}' = H_1(H_2(e(P, P)^{K^1{}_1 K^2{}_1 K^3{}_1{}'})K^4{}_1 P)$.

3.4 Member Leave Protocol

When a Member Leave event happens, the key tree should be modified and some keys should be replaced for forward secrecy. Upon receiving the leaving message, all the members in the group will update the key tree accordingly and decide on a sponsor.

Suppose a member U_c is leaving the group. Upon receiving the member leaving request, all the members in the group will agree on a sponsor and update the key tree as follows. If the parent node of the member node originally assigned to the leaving member has more than two child nodes, than the member node assigned to the leaving member is removed directly, and the sponsor is the member associated with the rightmost remaining child node under the deleted member node's parent node. Otherwise, if the parent node of the member node assigned to the leaving member has only two child nodes, after the removal of the member node assigned to the leaving member, there is only one child node left. The member associated with this solitary node is promoted and re-associated with its parent node and will be the sponsor; this solitary node is no more necessary and is removed too.

After updating the key tree, the sponsor, who knows all the necessary keys, will stand out and be responsible for the key refreshing. The sponsor computes and updates all the blinded keys and union blinded keys of the nodes on its key path. Then, the sponsor will broadcast these refreshed keys to all of the group members. After receiving the broadcast message, each legitimate member in the group can derive all the key generation keys on his/her key path, and all the legitimate members can obtain the same new group key.

For example, as shown in Figure 4, there are thirteen members in this group and the member U_7 will leave the group. The node $N^7{}_0$ assigned to the leaving member U_7 will be deleted from the key tree. The key generation key $K^7{}_0$ and the union blinded key $UBK^8{}_0$ which are no longer necessary will also be deleted. The new key tree is shown in Figure 5. U_8 will be the sponsor and responsible for the keys refreshing in this case. After computing the key generation keys $K^2{}_1{}'$,

$$K^2{}_1{}' = H_2(e(BK^5{}_0, BK^6{}_0)^{K^8{}_0}) = H_2(e(P, P)^{K^5{}_0 K^6{}_0 K^8{}_0}),$$

U_8 can proceed to compute the blinded key $BK^2{}_1{}'$ and the union blinded key $UBK^4{}_1{}'$ as follows.

Journal of Theoretical and Applied Information Technology
20th March 2013. Vol. 49 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$BK^2{}_1{}' = K^2{}_1{}' P = H_2(e(P, P)^{K^5{}_0 K^6{}_0 K^8{}_0})P$

$UBK^4{}_1{}' = H_2(e(BK^1{}_1, BK^3{}_1)^{K^2{}_1{}'})P = H_2(e(P, P)^{K^1{}_1 K^2{}_1{}' K^3{}_1})P$

After finishing the computation, the sponsor U_8 broadcasts the updated blinded key $BK^2{}_1{}'$ and union blinded key $UBK^4{}_1{}'$ to the members of the group. All the legitimate members now can compute the group key $K^1{}_2{}' = H_1(H_2(e(P, P)^{K^1{}_1 K^2{}_1{}' K^3{}_1})K^4{}_1 P)$. It is clear that the key generation key $K^2{}_1{}'$ and the group key $K^1{}_2{}'$ are refreshed after the update, and the forward secrecy is guaranteed.

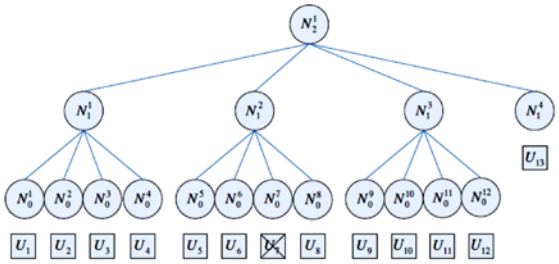

Figure 4: The Key Tree Before Member Leave Event

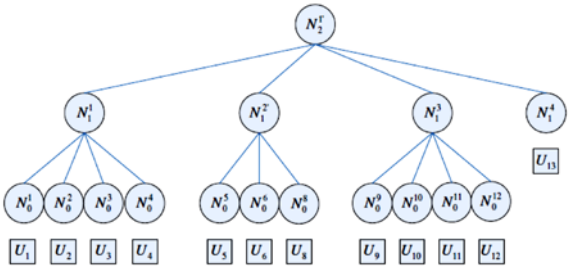

Figure 5: The New Key Tree After Member Leave Event

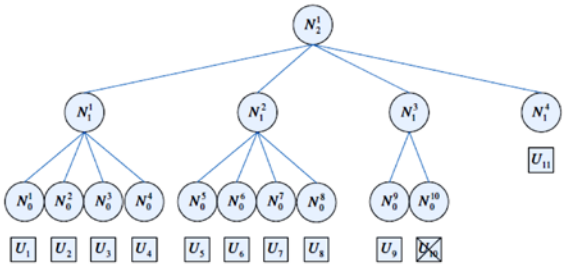

Figure 6: The Key Tree Before Member Leave Event

Another case in which more than one node will be deleted form the key tree after a member leaving is illustrated in Figure 6. If the member U_{10} will leave the group, in addition to $N^{10}{}_0$, the node $N^9{}_0$ should be deleted too and the member U_9 will be promoted and assigned to $N^3{}_1$. The new key tree is shown in Figure 7. In this case, the member U_9 will be the sponsor, and refresh the keys as follows.

$K^3{}_1{}' = K^9{}_0$

$BK^3{}_1{}' = K^9{}_0 P$

$UBK^4{}_1{}' = H_2(e(P, P)^{K^1{}_1 K^2{}_1 K^3{}_1{}'})P$

After receiving the blinded key $BK^3{}_1{}'$ and union blinded key $UBK^4{}_1{}'$ broadcasted by the sponsor U_9, all the legitimate members in this group can derive the group key $K^1{}_2{}' = H_1(H_2(e(P, P)^{K^1{}_1 K^2{}_1 K^3{}_1{}'})K^4{}_1 P)$.

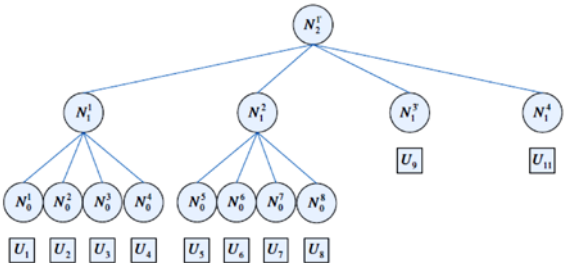

Figure 7: The Key Tree After Member Leave Event

4. ANALYSES AND COMPARISONS

A secure key agreement protocol for large dynamic groups should provide group key secrecy, backward secrecy and forward secrecy. This section will show how the proposed scheme does achieve these security requirements. Once a Member Join event or Member Leave event happens, the key tree will be updated accordingly, and all the keys concerning the joining or leaving member will be refreshed. In the Member Join case, the new member cannot learn any knowledge of the previous key generation keys. And the leaving member cannot derive any key generation keys from the old information he/she has in the Member Leave scenario.

Now let's discuss the group key secrecy of our scheme. An outside attacker might know the public system parameters (G_1, G_2, e, P, H_1, H_2), and all the blinded keys and union blinded keys. If an attacker attempted to compromise a key generation key $K^i{}_j$ directly from its corresponding blinded key $BK^i{}_j$ which is equal to $K^i{}_j P$, the problem the attacker would have to solve is the DLP in G_1, which is an impossible mission to accomplish nowadays.

Attempts that an attacker might possibly make to compromise the key generation key $K^i{}_j$ of a key node from some public information can be categorized into 3 different cases.

Case I. Suppose the attacker tries to compromise a key generation key $K^i{}_j$ of $N^i{}_j$ which has two child nodes $N^{4i-3}{}_{j-1}$ and $N^{4i-2}{}_{j-1}$. Since the key $K^i{}_j$ is defined as $H_1(K^{4i-3}{}_{j-1} K^{4i-2}{}_{j-1} P)$, and the correlative public information the attacker can gather is $BK^{4i-3}{}_{j-}$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$_1$ (= $K^{4i-3}_{j-1}P$) and BK^{4i-2}_{j-1} (= $K^{4i-2}_{j-1}P$), the problem the attacker is facing now, namely to derive K^i_j from BK^{4i-3}_{j-1} and BK^{4i-2}_{j-1}, is equivalent to solving the CDHP in G_1

Case II. Suppose the attacker tries to compromise the key generation key K^i_j of some N^i_j which has three child nodes N^{4i-3}_{j-1}, N^{4i-2}_{j-1}, and N^{4i-1}_{j-1}. Since the key K^i_j is defined as $H_2(e(P,P)^{K^{4i-3}_{j-1}K^{4i-2}_{j-1}K^{4i-1}_{j-1}})$, and the correlative public information the attacker can gather is BK^{4i-3}_{j-1} (= $K^{4i-3}_{j-1}P$), BK^{4i-2}_{j-1} (= $K^{4i-2}_{j-1}P$), and BK^{4i-1}_{j-1} (= $K^{4i-1}_{j-1}P$), the problem the attacker is facing now, namely to derive K^i_j from BK^{4i-3}_{j-1}, BK^{4i-2}_{j-1}, and BK^{4i-1}_{j-1}, is equivalent to solving the BDHP in (G_1, G_2, e).

Case III. Suppose the attacker tries to compromise the key generation key K^i_j of some N^i_j which has four child nodes N^{4i-3}_{j-1}, N^{4i-2}_{j-1}, N^{4i-1}_{j-1}, and N^{4i}_{j-1}. Since the key K^i_j is defined as $H_1(H_2(e(P,P)^{K^{4i-3}_{j-1}K^{4i-2}_{j-1}K^{4i-1}_{j-1}})K^{4i}_{j-1}$, and the correlative public information the attacker can gather is BK^{4i-3}_{j-1} (=$K^{4i-3}_{j-1}P$), BK^{4i-2}_{j-1} (= $K^{4i-2}_{j-1}P$), BK^{4i-1}_{j-1} (= $K^{4i-1}_{j-1}P$), BK^{4i}_{j-1}(= $K^{4i}_{j-1}P$), and $UBK^{4i}_{j-1} = H_2(e(P,P)^{K^{4i-3}_{j-1}K^{4i-2}_{j-1}K^{4i-1}_{j-1}})K^{4i}_{j-1})P$, the problem the attacker is facing now, namely to derive K^i_j from BK^{4i-3}_{j-1}, BK^{4i-2}_{j-1}, BK^{4i-1}_{j-1}, and BK^{4i}_{j-1}, is equivalent to solving the BDHP in (G_1, G_2, e). With the help of UBK^{4i}_{j-1}, in order to get K^i_j , the attacker would have to solve K^{4i}_{j-1} first, and that is the difficulty of solving a DLP in G_1.

Through the above analyses, we prove that the proposed scheme does provide group key secrecy, backward secrecy, and forward secrecy.

The computational efficiency and the economy of the number of keys are also very important criteria used to check out whether a group key agreement protocol can stand out among others. In this section, we shall show how our new scheme compares with [20] and [10] in these two aspects. The computational cost arises from two kinds of circumstances: one is the key deducing process, and the other is the key refreshing process. Computational costs of the two circumstances both depend on the height of the underlying key tree, the balance of the key tree, and the position where the involved member is located. In order to simplify the analyses, we only consider the worst case while the underlying key tree is full and highly balanced. Table 4 summarizes the computational costs of [20], [10] and the proposed scheme, where n is the number of members in the current group and x is the number of nodes with 2 sibling nodes.

Table 4 reveals that the computational cost is highly correlated with the height of the underlying key tree. The proposed scheme gains the advantage over the others. The computations of pairings and exponentiations are rather costly and almost dominate the computational cost. The proposed scheme is superior to the others in terms of such computations.

Table 4: Computational Cost Comparisons

The Key Deduction			
	[20]	[10]	Ours
Round	$[\log_2 n]$	$[\log_3 n]$	$[\log_4 n]$
Hash	$[\log_2 n]$	$[\log_3 n]$	$2[\log_4 n]$
$\times_{G1}$	0	x	$[\log_4 n]$
Exp.	$[\log_2 n]$	$[\log_3 n]$- x	$[\log_4 n]$
Pairing	$[\log_2 n]$	$[\log_3 n]$- x	$[\log_4 n]$
Inv.	0	0	0
The Member Join event			
Round	$[\log_2 n]$	$[\log_3 n]$	$[\log_4 n]$
Hash	$2[\log_2 n]+1$	$[\log_3 n]$	$3[\log_4 n]$
$\times_{G1}$	$3([\log_2 n]+1)$	$[\log_3 n]+x$	$3[\log_4 n]$
Exp.	$[\log_2 n]$	$[\log_3 n]$- x	$[\log_4 n]$
Pairing	$[\log_2 n]$	$[\log_3 n]$- x	$[\log_4 n]$
Inv.	$[\log_2 n]+1$	0	0
The Member Leave Event			
Round	$[\log_2 n]$	$[\log_3 n]$	$[\log_4 n]$
Hash	$2[\log_2 n]-2$	$[\log_3 n]$	$3[\log_4 n]$
$\times_{G1}$	$3([\log_2 n]-1)$	$[\log_3 n]+x$	$3[\log_4 n]$
Exp.	$[\log_2 n]-1$	$[\log_3 n]$- x	$[\log_4 n]$
Pairing	$[\log_2 n]-1$	$[\log_3 n]$- x	$[\log_4 n]$
Inv.	$[\log_2 n]-1$	0	0

The number of keys is another important factor we need to consider when we evaluate group key agreement protocols. We consider the worst case when the underlying key tree is full and highly balanced. Table 5 summarizes the numbers of keys and their types adopted in [20], [10], and the proposed scheme, where n is the number of members in the communication group.

The number of keys is correlated with the total number of nodes in the underlying key tree. Since the underlying key tree considered in this analysis is full and balanced, in order to handle a system of n members, there are $2n$-1, ($3n$-1)/2, and ($4n$-1)/3 nodes in the binary tree, ternary tree, and quad tree, respectively. Public keys known to all the members in the group, including blinded keys and union blinded keys, should be published in a public domain or kept by each member. Therefore, the number of public keys is a good indicator and is also what we have compared among the schemes (see Table 5). As the table suggests, in terms of the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

number of public keys, the move from binary tree to ternary tree does mean more efficienct. However, due to the additional use of union blinded keys, our quad tree protocol needs a bit more public keys than the ternary tree. With all the keys counted, our new scheme has approximately the same number as [10] with the binary tree left far behind.

Table 5: Total Number Of Keys And Blinded Keys

	Binary	Ternary	Ours
Nodes	2*n*-1	(3*n*-1)/2	(4*n*-1)/3
KGKs	2*n*-1	(3*n*-1)/2	(4*n*-1)/3
Private keys	2*n*-1	0	0
BKs	2*n*-1	(3*n*-1)/2	(4*n*-1)/3
UBKs	0	0	(4*n*-1)/12
Public keys	2*n*-1	(3*n*-1)/2	5(4*n*-1)/12

5. CONCLUSIONS

In this paper, we propose a secure, efficient, and scalable key agreement scheme for large and dynamic multicast groups. The underlying key tree adopted in the proposed scheme is expanded to a quad tree. In this way, the height of the underlying key tree can be reduced, which acquired the efficiency. In fact, the proposed scheme not only performs with high efficiency but meets all the security requirements. Among these related schemes, our new scheme has made remarkable advances in comparisons on computational costs for the key deducing, Member Join event, and Member Leave event. The less improvement on the number of keys is caused by introducing the union blinded keys in our scheme. Owing to the properties of the bilinear mapping, the union blinded keys cannot be removed from our scheme. Therefore, how to effectively reduce the amount of keys is a significant and interesting problem which deserves further research.

ACKNOWLEDGEMENT

Part of this research was supported by National Science Council (NSC), Taiwan, under contract of NSC 101-2221-E-439 -001.

REFERENCES:

[1] S. Al-Riyami and K. Paterson. Authenticated three party key agreement protocols from pairings. *Cryptology Print Archive*, 2002.

[2] Shu-Fen Chiou, Min-Shiang Hwang, Song-Kong Chong, A Simple and Secure Key Agreement Protocol to Integrate a Key Distribution Procedure into the DSS, *International Journal of Advancements in Computing Technology*, Vol.4, No.19, 2012, pp.529-535.

[3] C. L. Liu, G. Horng, and T. Y. Chen, Further refinement of pairing computation based on Miller's algorithm. *Applied Mathematics and Computation*, Vol.189, No.1, 2007, pp.395-409.

[4] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transaction on Information Theory*, Vol.22, No.6, 1976, pp.644-654.

[5] D. J. Huang and D. Medhi. A key-chain-based keying scheme for many-to-many secure group communication. *ACM Trans. on Information and System Security*, Vol.7, No.4, 2004, pp.523-552.

[6] A. Joux. A one round protocol for tripartite Diffe-Hellman. *Journal of Cryptology*, Vol.17, No.4, 2004, pp.263-276.

[7] Ting-Yi Chang and Min-Shiang Hwang, User-Anonymous and Short-Term Conference Key Distribution System via Link-Layer Routing in Mobile Communications, *International Journal of Mobile Communications*, Vol.9, No.2, 2011, pp.144-158.

[8] Y. Kim, A. Perrig, and G. Tsudik. Tree-based group key agreement. *ACM Trans. on Information and System Security*, Vol.7, No.1, 2004, pp.60-96.

[9] Jung-Wen Lo, Min-Shiang Hwang, Chia-Hsin Liu, An Efficient Key Assignment Scheme for Access Control in a Class Hierarchy, Information Sciences, Vol.181, No.4, 2011, pp.917-925.

[10] 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*, vol.2846, 2003, pp.357-371.

[11] Ting-Yi Chang, Min-Shiang Hwang, Wei-Pang Yang, A Communication-Efficient Three-Party Password Authenticated Key Exchange Protocol, *Information Sciences*, Vol.181, 2011, pp. 217-226.

[12] Tsuei-Hung Sun and Min-Shiang Hwang, A Hierarchical Data Access and Key Management in Cloud Computing, *ICIC Express Letters*, Vol.6, No.2, 2012, pp.569-574.

[13] J. Nam, S. Kim, and D. Won. Secure group communications over combined wired and wireless networks. *International Conference on Trust, Privacy and Security in Digital Business*, Vol.3592, Demark, 2005, pp.90-99.

[14] S. Rafaeli and D. Hutchison. A survey of key management for secure group communication. *ACM Computing Surveys*, Vol.35, No.3, 2003, pp.309-329.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[15] M. M. N. Rasslan, Y. H. Dakroury, and H. K. Aslan. A new secure multicast key distribution protocol using combinatorial boolean approach. *International Journal of Network Security*, Vol.8, No.1, 2009, pp.75-89.

[16] A. El Sayed, V. Roca, and L. Mathy. A survey of proposals for an alternative group communication service. *IEEE Network*, Vol.17, No.1, 2003, pp.46-51.

[17] A. T. Sherman and D. A. McGrew. Key establishment in large dynamic groups using one-way function trees. *IEEE Transactions on Software Engineering*, Vol.29, No.5, 2003, pp.444-458.

[18] Y. M. Tseng. A scalable key management scheme with minimizing key storage for secure group communications. *International Journal of Network Management*, Vol.13, No.6, 2003, pp.419-425.

[19] D. Wallner, E. Harder, and R. Agee. Key management for multicast: issues and architectures. RFC 2627, 1999.

[20] L. M Wang and C. K. Wu. Efficient key agreement for large and dynamic multicast groups. *International Journal of Network Security*, Vol.3, No.1, 2006, pp.8-17.

[21] C. K. Wong, M. Gouda, and S. S. Lam. Secure group communications using key graphs. *IEEE/ACM Transactions on Networking*, Vol.8, No.1, 2000, pp.16-30.

[22] S. Xu, C. T. Huang, and M. M. Matthews. Secure multicast in WiMAX. *Journal of Networks*, Vol.3, No.2, 2008, pp.48-57.

[23] Q. Zhang and K. L. Calvert. A peer-based recovery scheme for group rekeying in secure multicast. *International Journal of Network Security*, Vol.6, No.1, 2008, pp.15-25.

[24] C. C. Lee, C. T. Li, T. Y. Chen, P.H. Wu, C. T. Chen, A New Key Exchange Protocol with Anonymity between STB and Smart Card in IPTV Broadcasting, *7th International Conference on Wireless Communications, Networking and Mobile Computing*, China, 2011.

[25] Song-Kong Chong, Hsien-Chu Wu, and Chia-Chun Wu, A Scheme for Key Management on Alternate Temporal Key Hash, *International Journal of Network Security*, Vol.1, No.1, 2005, pp.8-13.