Efficient Key Agreement for Large and Dynamic Multicast Groups

Liming Wang[1,2], and Chuan-Kun Wu[1]
(Corresponding author: Liming Wang)

State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences[1]
Beijing 100080, P.R.China (Email: limingwang@is.iscas.ac.cn)
Graduate School of Chinese Academy of Sciences[2]
Beijing 100039, P.R. China (Email: ckwu@is.iscas.ac.cn)

(Received July 20, 2005; revised and accepted Aug. 17 & Sep. 12, 2005)

Abstract

Secure multicast represents the core component of many web and multimedia applications such as pay-TV, teleconferencing, real-time distribution of stock market price and etc. The main challenges for secure multicast is scalability, efficiency and authenticity. In this paper, we propose a scalable, efficient, authenticated group key agreement scheme for large and dynamic multicast systems. The proposed key agreement scheme is identity-based which uses the bilinear map over the elliptic curves. Compared with the previously published schemes, our scheme provides group member authenticity without imposing extra mechanism. Furthermore, we give a scalability solution based on the subgroups, which has advantages over the existing schemes. Security analysis shows that our scheme satisfies both forward secrecy and backward secrecy.

Keywords: multicast, bilinear pairing, key agreement

1 Introduction

Many types of group applications, such as pay per view distribution of digital media, teleconferencing, software updates and real-time delivery of stock market information can benefit from IP multicast [13, 14, 15], which greatly reduced the server overhead and bandwidth usage by enabling source to send a single copy of message to multiple recipients.

One of the main challenges for secure multicast is access control for making sure that only legitimate members of multicast group have access to the group communication. In the passed two or three decades, cryptography has become the well-established means to solve the security problems in networking. However, there are still a lot of difficulties for directly deploying cryptography algorithms into multicasting environment as what has been done for unicasting environment. The commonly used technique to secure multicast communication is to maintain a group key that is known to all users in the multicast group, but is unknown to any one outside the group [8, 16, 20, 21, 23, 29, 31, 32, 33, 34]. Efficiently managing the group key is a difficult problem for large dynamic groups. Each time a member is added to or evicted from the communication group, the group key must be refreshed. The members in the group must be able to compute the new group key efficiently, at the same time forward and backward secrecy must be guaranteed. Because the group rekeying is very consumptive and frequently performed due to the nature of multicast communication, the way to update it in a scalable and secure fashion is required.

1.1 Related Work

There are several schemes proposed for secure multicast. In this section, we will briefly review some of these schemes.

Iolus [28] approach proposed the notion of hierarchy subgroup for scalable and secure mulitcast. In this method, a large communication group is divided into smaller subgroups. Each subgroup is treated almost like a separate multicast group and is managed by a trusted group security intermediary (GSI). GSI connect between the subgroups and share the subgroup key with each of their subgroup members. GSIs act as message relays and key translators between the subgroups by receiving the multicast messages from one subgroup, decrypting them and then remulticasing them to the next subgroup after encrypting them by the subgroup key of the next subgroup. The GSIs are also grouped in a top-level group that is managed by a group security controller (GSC), see Figure 1.

Although Iolus has improved the scalability of the system, because the member join or leave only affect their

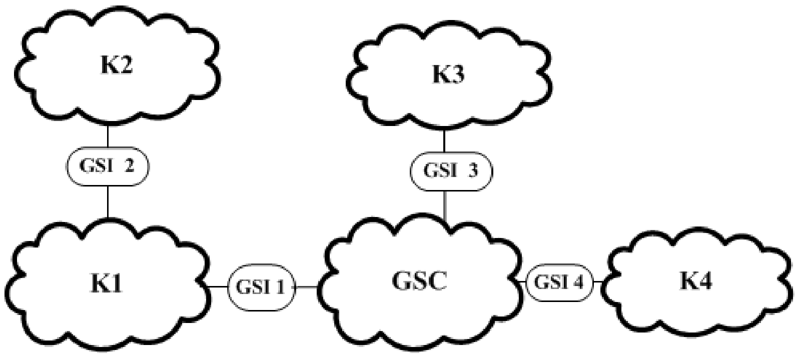


Figure 1: Framework of Iulos

subgroup only while the other subgroup will not be affected. It has the drawback of affecting data path. This occurs in the sense that there is a need for translating the data that goes from one subgroup, and thereby one key, to another. This becomes even more problematic when it takes into account that the GSI has to manage the subgroup and perform the translation needed. The GSI may thus becomes the bottleneck.

The logical key hierarchy(LKH) is an efficient approach that supports dynamic group membership. This method was proposed by Wallner et al. [33] and Wong et al. [34] individually. Waller et al. discussed binary trees and Wong et al. discussed the generalized case - key graphs, but the implicated ideas in their method is identical - to convert the cost of communication from linearly to logarithm with the group size of n. In this approach, the group controller (GC) maintains a logical key tree where each node represents a key encryption key (KEK). The root of the key tree is the group key used for encrypting data in group communications and it is shared by all users. The leave node of the key tree is associated with a user in the communication group. Each user secretly maintains the keys related to the nodes in the path from its leaf node to the root. We call the set of keys that a member knows the key path. Figure 2 shows a sample of key tree. When a member leaves the group, all the keys that the member knows, including the group key and its key path, need to be refreshed. When a member joins the group, GC authenticates the member and assigns it to a leaf node of the key tree. The GC will send the new member all the keys from his/her corresponding leaf node to the root. The main reason for using such a key tree is to efficiently update the group key if a member joins or leaves the group.

An optimization of the logical key hierarchy approach is one-way function tree (OFT) proposed by McGrew and Sherman [25, 31]. Their scheme reduces the size of rekeying message from $2\log_2 n$ to $\log_2 n$. Canetti et al. [8] proposed a slightly different method that achieves the same communication overhead using a pseudo-random generator tree. This algorithm is known as the one-way function chain tree (OFCT) and it is applied only on users

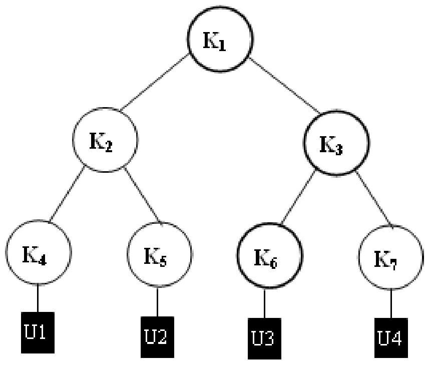


Figure 2: Sample of hierarchical key tree

removal. One of the main drawback of LKH and its variants is that they are centralized. In this kind of systems, there is only one entity (GC) controlling the whole group. With the growth of the group member, GC should pay out heavy cost to manage and maintain a huge key tree. This problem is exacerbated when the group has a highly dynamic membership change. Moreover, if the GC aborts, the whole communication group will be affected.

The rekeying method used in [11] has considered a different distribution of keys in the key tree. In this approach, Chang et al. use a dynamic key hierarchy instead of a fixed hierarchy of keys. Also, they use the Boolean function minimization technique to minimize the cost of communication. Although the size of rekeying messages and the storage of GC are reduced, their scheme suffers from collusion attack.

1.2 Our Work

In this paper, we propose a scalable, efficient, authenticated group key agreement scheme for multicast. Our scheme makes use of the bilinear pairings over the elliptic curves [4, 5, 6, 7, 9, 10, 18, 19, 22, 24] and is inspired on the work of McCullagh and Barreto [26]. Our scheme has advantages over the exiting schemes proposed for secure multicast. First, compared with the previously published tree-based schemes [20, 21, 29, 31, 33, 34], our scheme achieves group member authentication without imposing extra mechanism. Since we use an identity tree instead of key tree in our scheme. Each node in the identity tree is associated with an identity. The leaf node's identity is corresponding to the user's identity and the intermediate node's identity is generated by its children's identity. Hence, in an identity tree, an intermediate node represents a set users in the subtree rooted at this node.

Next, our scheme solves the scalability problem in multicast systems. Since we divide the large communication group into several smaller subgroups. Each subgroup is independently maintained by the subgroup controller (SGC). In our scheme, even though a subgroup controller fails, it does not affect its subgroup. Because every user in the subgroup can act as the subgroup group controller. This is an amazing feature especially for the groups that has a highly dynamic membership change in mobile and ad hoc networks.

Third, in our scheme, the keys used in each subgroup can be generated by a group of key generation centers (KGCs) in parallel. All the members in the same subgroup can compute the same subgroup key though the keys for them are generated by different KGCs. This is a desirable feature especially for the large-scale network systems, because it minimizes the the problem of concentrating the workload on a single entity.

The rest of paper is organized as follows. Section 2 gives the background definitions associated with our scheme. Section 3 presents the authenticated identity based multicasting scheme. Section 4 gives security analysis of our scheme. Section 5 compares the proposed scheme with the previously published schemes. Section 6 concludes the paper.

2 Security Definitions

2.1 Bilinear Groups

Let G_1 be a cyclic additive group, whose order is a prime q and G_2 be a multiplicative group of the same order. A bilinear map $\hat{e} : G_1 \times G_1 \rightarrow G_2$ must satisfy the following properties:

1) Bilinear: for all $P, Q \in G_1$ and $a, b \in Z_q^*$, we have $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$

2) Non-degenerate: $\hat{e}(P, P) \neq 1_{G_2}$, if P is a generator of G_1.

3) The map $\hat{e}$ is efficiently computable.

We note that the weil and tate pairings associated with supersingular elliptic curves or Abelian varieties can be modified to create such bilinear pairings. Please refer to [1, 2, 3, 12, 17, 27, 30] for more details.

2.2 Complexity Assumptions

Definition 1 *Elliptic Curve Discrete Logarithm Problem (ECDLP): Given G_1 as above, choose P a generator from G_1, given xP, the ECDLP is to find x, where x is an random element of Z_q^*.*

Definition 2 *Computational Diffie-Hellman Problem (CDHP): the computational DH problem is to compute abP when given a generator P of G and aP, bP for some $a, b \in Z_q^*$.*

Definition 3 *Bilinear Inverse Diffie-Hellman Problem (BIDHP): Let G_1, G_2, P and $\hat{e}$ be as above. The BIDHP in $\langle G_1, G_2, \hat{e}\rangle$ is as follows: Given (P, aP, bP) with uniformly random choices of $a, b \in Z_q^*$, compute $\hat{e}(P, P)^{a^{-1}b}$.*

Assumptions 1 *We assume that ECDLP, CDHP, BIDHP are hard, which means there is no polynomial time algorithm to solve any of them with non-negligible probability.*

2.3 Security Requirements for Multicast

We consider dynamic groups where users can join or leave the multicast group at any time. The main security properties of multicast are:

1) Group Key Secrecy guarantees that it is computationally infeasible for a passive adversary to discover any group key.

2) Backward Secrecy is used to prevent a new member from decoding messages exchanged before it joined the group. This property guarantees that a passive adversary who knows a subset of group keys cannot discover the previous group keys.

3) Forward Secrecy is used to prevent a leaving user or expelled group member to continue accessing the group communication. This property guarantees that a passive adversary who knows a subset of old group keys cannot discover the subsequent group keys.

3 Our Scheme

We use the following notation throughout of the remainder this paper shown as Table 1.

3.1 Framework of Our Protocol

From our earlier discussion, it can be seen that, in a centralized multicast system, there is only one entity controlling the whole communication group. The group does not rely on any auxiliary entity to perform key generation, key distribution and group rekeying. If there is any problem with the group controller, all the group members in the communication group will be affected. So the group controller is the single point of failure. Additionally, a

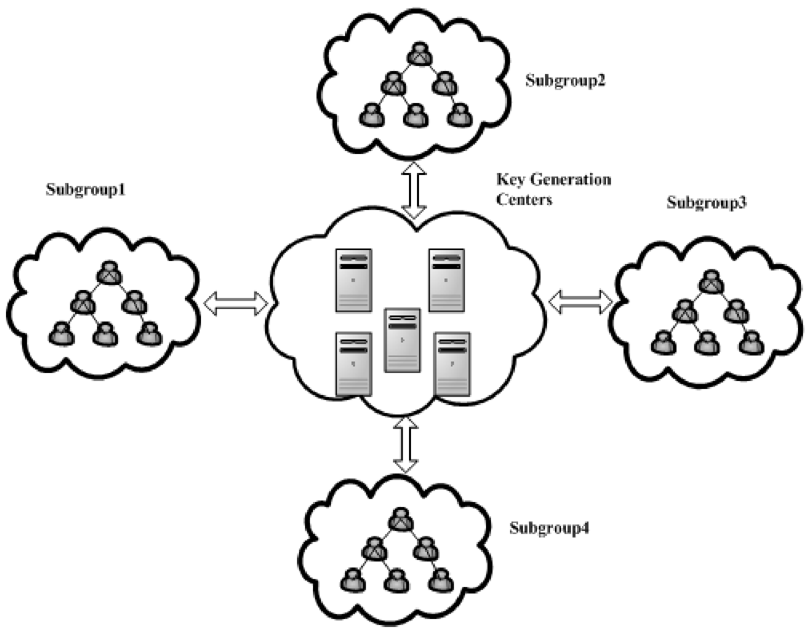


Figure 3: Architecture of our multicast system

Table 1: Notations

n	number of subgroup members
U_i	i-th group member, $i \in \{1, 2, \cdots, n\}$
ID_i	U_i's identity
N_v^l	l-th level v-th node in an identity tree
Q_j^i	hash value of N_j^i's identity
h	height of an identity tree
P_j^i	private key of the node N_j^i
K_j^i	key generation key of the node N_j^i
BK_j^i	the blinded key of the node N_j^i
KGC_i	i-th key generation center
s_v	the local master key of v-th key generation center

multicast communication group may has a large number of users, controlled by only one single entity may raise the problem of scalability.

Our protocol directly addresses the problem of reducing the overload of the group controller. We divides the multicast communication group into regional subgroups. Each subgroup is independently managed by a subgroup controller (SGC) like a separate multicast group with its own subgroup key. Thus, when a member joins or leaves the communication group, it joins or leaves only its local subgroup. As a result, only the local subgroup communication key needs to be refreshed and the scalability problem is greatly mitigated. We use a 'group' of key generation centers (KGCs) to share the overall key generation and distribution workload. In our scheme the task of SGC is just to update the identity tree when there is a membership change in the subgroup and send it to KGCs. Note that this task can be done by any user in the subgroup. All the keys including the users' private keys, blinded keys in our multicast system are generated by the KGCs. Moreover the key distribution is also fulfilled by the KGCs. Using the subgroup key, KGCs can encrypt the message for the subgroup. Although the group members' private keys/blinded keys are generated by distinct KGCs, all members in the same subgroup can generate the same subgroup communication key. This is a significant feature especially for the large and dynamic communication groups. Figure 3 shows the architecture of our mulitcast system.

3.2 System Setup

Given security parameter 1^k, the KGCs generates two groups G_1, G_2, and an admissible bilinear map $\hat{e} : G_1 \times G_1 \longrightarrow G_2$, where G_1 denotes a cyclic additive group of prime order q and G_2 is a multiplicative group of the same order. The KGCs chooses a generator P of G_1 and publishes the system parameters **params** $= \{G_1, G_2, \hat{e}, P, H_1, H_2, H_3\}$, here $H_1 : \{0,1\}^* \rightarrow Z_q^*, H_2 : G_2 \rightarrow Z_q^*, H_3 : Z_q^* \times Z_q^* \rightarrow Z_q^*$ are cryptographic hash functions.

The basic idea of our scheme is the usage of an identity tree, where each node in the tree has an identity. The leaf node's identity is corresponding to a user's identity and the interior node's identity is generated from it's children's identity. Figure 4 shows an example of identity tree. A node in the identity tree is also associate with a

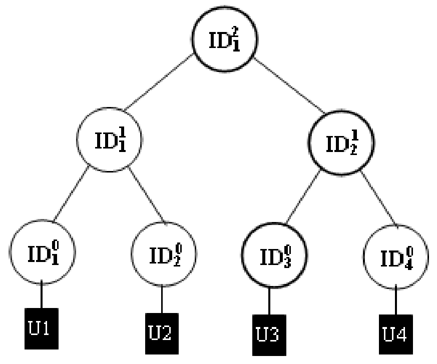


Figure 4: An example of identity tree

key generation key (KGK) which is used for generating a parent key. The root node's KGK is used as the group key.

Given the public parameters, each SGC constructs an identity tree and gives it to the KGCs. Given a node N_j^i, KGC$_v$ generates the keys for the nodes in an identity tree as follows. First it chooses a random element $s_v \in Z_q^*$ and keeps s_v as its *local master key.* KGC$_v$ computes $P_j^i = (Q_j^i + s_v)^{-1}P$ and $TK_j^i = (Q_j^i + s_v)P$. Here, if the node is a leaf node then $Q_j^i = H_1(ID_j)$. Otherwise if the node is an intermediate node, $Q_j^i = H_3(Q_{2j-1}^{i-1}, Q_{2j}^{i-1})$. We call P_j^i, TK_j^i the private key and the temporary key for the node N_j^i respectively. Note that TK_j^i is used only by KGCs and is unknown to the users. Then KGCs generate the key generation key for the node N_j^i as follows. If N_j^i is a leaf node in the identity tree, KGC$_v$ randomly chooses a random number K_j^i from Z_q^* for U_j. The KGK on the leaf node is also called individual key and can be updated periodically. KGC$_v$ securely unicast the private keys from the leaf node to the root as well as the individual key to U_j. We define $K_{2j-1}^{i-1}TK_{2j}^{i-1}$ and $K_{2j}^{i-1}TK_{2j-1}^{i-1}$ as a pair of blinded factor of N_j^i. $K_{2j-1}^{i-1}TK_{2j}^{i-1}(K_{2j}^{i-1}TK_{2j-1}^{i-1})$ is the blinded key of $N_{2j-1}^{i-1}(N_{2j}^{i-1})$. Note that the blinded keys and the identity tree are the public information. Using the private keys and the blinded keys in the identity tree, we can compute the KGK on an intermediate node key as follows.

$$\begin{aligned}
& K_j^i \\
= \;& H_2(\hat{e}(P_{2j-1}^{i-1}, BK_{2j}^{i-1}))^{K_{2j-1}^{i-1}} \\
= \;& H_2(\hat{e}((Q_{2j-1}^{i-1} + s_v)^{-1}P, K_{2j}^{i-1}(Q_{2j-1}^{i-1} + s_v)P)^{K_{2j-1}^{i-1}} \\
= \;& H_2(\hat{e}(P_{2j}^{i-1}, BK_{2j-1}^{i-1}))^{K_{2j}^{i-1}} \\
= \;& H_2(\hat{e}((Q_{2j}^{i-1} + s_l)^{-1}P, K_{2j-1}^{i-1}(Q_{2j}^{i-1} + s_l)P)^{K_{2j}^{i-1}} \\
= \;& H_2(\hat{e}(P,P)^{K_{2j-1}^{i-1}K_{2j}^{i-1}}).
\end{aligned}$$

For example, as shown in Figure 4, U_3 received the individual key K_3^0 from KGCs. In addition, U_3 knows the private keys from its corresponding leaf node to the root node. So the node N_3^0's private key $P_3^0 = (Q_3^0 + s_v)^{-1}P$ and the node N_2^1's private key $P_2^1 = (Q_2^1 + s_\omega)^{-1}P$ are stored by U_3 locally. Since the identity tree and all the blinded keys are public information, U_3 can easily obtain the blinded key $BK_4^0 = K_4^0(Q_3^0 + s_v)P$ and $BK_1^1 = K_1^1(Q_2^1 + s_\omega)P$. Then U_3 can compute K_2^1 and K_1^2 as follows:

$$\begin{aligned}
K_2^1 &= H_2(\hat{e}((Q_3^0 + s_v)^{-1}P, K_4^0(Q_3^0 + s_v)P)^{K_3^0}) \\
&= H_2(\hat{e}(P,P)^{K_3^0K_4^0}).
\end{aligned}$$

$$\begin{aligned}
K_1^2 &= H_2(\hat{e}((Q_2^1 + s_\omega)^{-1}P, K_1^1(Q_2^1 + s_\omega)P)^{K_2^1}) \\
&= H_2(\hat{e}(P,P)^{K_1^1K_2^1}).
\end{aligned}$$

From the above computation, we can see that all users can compute the same group key though the private keys and the blinded keys of the node in the identity tree are generated by distinct KGCs. But for a passive eavesdropper, who has access to all the public information cannot compute the KGKs. For example, the eavesdropper who knows $K_{2j-1}^{i-1}(Q_{2j}^{i-1} + s_v)P$, $K_{2j}^{i-1}(Q_{2j-1}^{i-1} + s_\omega)P$, Q_{2j-1}^{i-1}, Q_{2j}^{i-1} and Q_j^i, and wishes to determine $K_j^i = H_2(\hat{e}(P,P)^{K_{2j-1}^{i-1}K_{2j}^{i-1}})$. Even if he can obtain $\hat{e}(P,P)^{K_{2j-1}^{i-1}}$ and $\hat{e}(P,P)^{K_{2j}^{i-1}}$ from the tuple $(BK_{2j-1}^{i-1}, BK_{2j}^{i-1}, Q_{2j-1}^{i-1}, Q_{2j}^{i-1}, Q_j^i)$, he would still have to compute $\hat{e}(P,P)^{K_{2j-1}^{i-1}K_{2j}^{i-1}}$ from $\hat{e}(P,P)^{K_{2j-1}^{i-1}}$ and $\hat{e}(P,P)^{K_{2j}^{i-1}}$, thus effectively solving the CDHP in G_2. This implies that the group key in our scheme is secure against eavesdroppers.

3.3 Member Join Event

We assume that the subgroup has n users, $\{U_1, U_2, \cdots, U_n\}$, when a subgroup receives a joining request from a new user U_j, the SGC searches the nearest leaf node N_x from the root to keep the height of the identity tree as low as possible. Now SGC generates a new node, the member associated with N_x and the new member become the new node's left child and right child respectively. SGC rearranges the levels of the affected nodes and keys in the updated tree $T^{'}$ and gives $T^{'}$ to KGCs. Then KGCs recompute the private keys and blinded keys of the affected nodes in $T^{'}$ for the subgroup. Figure 5 shows an example of new user U_4 joining the group. A new node N_2^1 is generated by SGC and becomes the parent of leaves N_3^0 and N_4^0.

To guarantee the backward secrecy, SGC has to update the identities of the affected nodes in the identity tree. The SGC recomputes the identities $Q_2^{1^{'}} = H_3(Q_3^0, Q_4^0)$,$Q_1^{2^{'}} = H_3(Q_1^1, Q_2^{1^{'}})$. Then, SGC sends the new tree $T^{'}$ to KGCs. KGCs generate the private keys and the blinded keys for the affected nodes in $T^{'}$ and send the private keys $P_4^0 = (Q_4^0 + s_v)^{-1}P$,$P_2^{1^{'}} = (Q_2^{1^{'}} + s_\phi)^{-1}P$, $P_1^{2^{'}} = (Q_1^2 + s_\omega)^{-1}P$ as well as a random number K_4^0 to U_4 securely. Note that K_4^0 is the individual key for U_4. Then KGCs publishes the new tree with all blinded

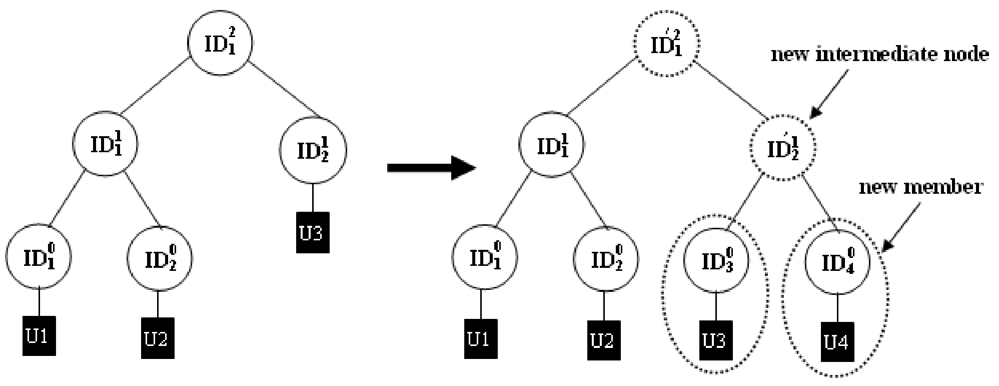


Figure 5: U_4 is added to the group

keys on a public board. Now all the group members can compute the new group key, for example, U_4 can compute the new group key as follows:

$$\begin{aligned} K_2^{1'} &= H_2(\hat{e}(P_4^0, BK_3^0)^{K_4^0}) = H_2(\hat{e}(P,P)^{K_4^0 K_3^0}) \\ K_1^{2'} &= H_2(\hat{e}(P_2^{1'}, BK_1^1)^{K_2^{1'}}) = H_2(\hat{e}(P,P)^{K_1^1 K_2^{1'}}) \end{aligned}$$

Note that after U_4 joined the group, all the private keys from U_3's parent to the root are changed. Since U_4 does not know the private keys in the previous identity tree, So he/she cannot compute the group key in the previous session. Thus the backward secrecy is satisfied.

3.4 Member Leave Event

When a user U_j leaves the group, all the private keys and KGKs held by nodes in the path from its parent node to the root are compromised and should be updated. This process is handled similarly to the member join event. The only difference is that KGCs compute fewer keys. SGC updates the identity tree by deleting the leaf node corresponding to U_j and rearranges the levels of affected nodes in the updated tree $T^{'}$. Then SGC sends the updated identity tree $T^{'}$ to KGCs. KGCs perform the key generation for $T^{'}$ as described above. For example, see Figure 6, U_3 and U_4 are deleted from the group.

Since U_3 and U_4 are deleted from the tree, SGC removes the node N_3^1 and N_1^0 from the identity tree. the node N_2^0 is prompted to the place of the node N_2^1. The SGC sets $Q_2^{1'} = Q_5$ and computes $Q_1^{2'} = H_3(Q_1^1, Q_2^{1'})$. All the group member can compute the new group key $K_1^{2'}$ as follows:

$$\begin{aligned} K_1^{2'} &= H_2(\hat{e}(P_1^1, BK_2^{1'})^{K_1^1}) \\ &= H_2(\hat{e}(P_2^{1'}, BK_1^1)^{K_2^{1'}}) \\ &= H_2(\hat{e}(P,P)^{K_2^{1'} K_1^1}). \end{aligned}$$

Note that after U_3 and U_4 leave the group, all the private keys known by U_3 and U_4 are changed. So U_3 and U_4 cannot compute the group key in the future. This means that our scheme satisfies forward secrecy. Furthermore, in the rekeying process, SGC just needs to update the identity tree and send it to KGCs. This task can be done by every user in the subgroup. So in our multicast system, even if the SGC aborts or leaves, the subgroup will not be affected. This is a significant feature especially for the mobile and ad hoc networks where have a highly dynamic membership change.

4 Security Analysis

In this section, we present security proofs of our scheme.

KGCs run the setup procedure for a given of security parameter 1^k, and provide two groups G_1, G_2 of prime order q, and an admissible bilinear map $\hat{e} : G_1 \times G_1 \longrightarrow G_2$. For $(q, G_1, G_2, \hat{e}) \leftarrow g(1^k)$, $X = (x_1, x_2, \cdots, x_n)$ and $Q = (I_1, I_2, \cdots, I_n)$, for $x_i, I_i \in Z_q^*$ and a binary identity tree T, we define the following random variables:

- $vw(h, X, Q, T) := \{BK_j^i$ where i and j are defined according to the identity tree $T\}$
- $K(h, X, Q, T) := H_2(\hat{e}(P,P)^{K_{2j-1}^{h-1} K_{2j}^{h-1}})$

Note that $vw(h, X, Q, T)$ is exactly the view of the adversary in our scheme, where the final key is $K(h, X, Q, T)$. Our goal is to show that $K(h, X, Q, T)$ in our scheme can not be distinguished by a polynomial time algorithm from a random number, if all the public values such as blinded keys, identity tree T and public parameters are known. We define the following two random variables:

- $A_k := (vw(h, X, Q, T), y), y \in_R Z_q^*$
- $F_k := (vw(h, X, Q, T), K(h, X, Q, T))$

Theorem 1 *If two-party key in our scheme is hard, then there is no polynomial time algorithm which can distinguish A_h from F_h.*

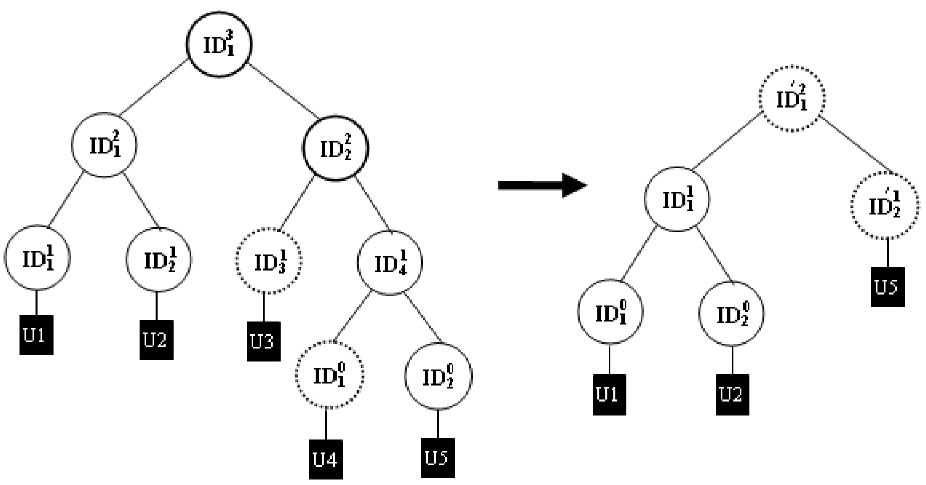


Figure 6: U_3 and U_4 are deleted from the group

Proof:
We proof the theorem by induction and contradiction. We assume that A_{h-1} and F_{h-1} can not be distinguished in polynomial time as the induction hypothesis. We will show that the ability to distinguish A_h and F_h implies either can be used to distinguish two-party key from a random value or can be used to distinguish A_{h-1} and F_{h-1}.

Let T_L, T_R respectively be the left and right subtree of height at most $h-1$ of the identity tree T. Let $X_L = (x_1, x_2, \cdots, x_l)$ and $X_R = (x_{l+1}, x_{l+2}, \cdots, x_n)$, where x_1 through x_l are associated with T_L, x_{l+1} through x_n are associated with T_R. Then A_h and F_h can be rewritten as:

$$\begin{aligned}
A_h &:= (vw(h, X, Q, T), y) \\
&= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad BK_1^{h-1}, BK_2^{h-1}, y) \\
&= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad K_L^{h-1}(s_\omega + Q_R)P, K_R^{h-1}(s_\phi + Q_L)P, y)
\end{aligned}$$

$$\begin{aligned}
F_h &:= (vw(h, X, Q, T), K(h, X, Q, T)) \\
&= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad BK_1^{h-1}, BK_2^{h-1}, KEY) \\
&= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad K_L^{h-1}(s_\omega + Q_R)P, K_R^{h-1}(s_\phi + Q_L)P, KEY)
\end{aligned}$$

We consider the following random variables:

$$\begin{aligned}
A_h &:= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad K_L^{h-1}(s_\omega + Q_R)P, K_R^{h-1}(s_\phi + Q_L)P, y) \\
B_h &:= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad r_1(s_\omega + Q_R)P, K_R^{h-1}(s_\phi + Q_L)P, y) \\
C_h &:= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad r_1(s_\omega + Q_R)P, r_2(s_\phi + Q_L)P, y) \\
D_h &:= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad r_1(s_\omega + Q_R)P, r_2(s_\phi + Q_L)P, K_1) \\
E_h &:= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad r_1(s_\omega + Q_R)P, K_R^{h-1}(s_\phi + Q_L)P, K_2) \\
F_h &:= (vw(h-1, X_L, Q_L, T_L), vw(h-1, X_R, Q_R, T_R), \\
&\quad K_L^{h-1}(s_\omega + Q_R)P, K_R^{h-1}(s_\phi + Q_L)P, K)
\end{aligned}$$

where r_1, r_2, y are random numbers in Z_q^*, $K_1 = H_2(\hat{e}(P,P)^{r_1 r_2})$, $K_2 = H_2(\hat{e}(P,P)^{r_1 K_R^{h-1}})$, $K = H_2(\hat{e}(P,P)^{K_L^{h-1} K_R^{h-1}})$.

If A_h and F_h can be distinguished in a polynomial time, then at least one of the following can be distinguished: $(A_h, B_h), (B_h, C_h), (C_h, D_h), (D_h, E_h), (E_h, F_h)$.

- (A_h and B_h): Suppose A_h and B_h can be distinguished in polynomial time by a distinguisher $\mathcal{D}_{AB_h}$. We will show that $\mathcal{D}_{AB_h}$ can be used to distinguish A_{h-1} and F_{h-1}. Let $V_{h-1}^* = (vw(h-1, X^*, Q^*, T^*), r^*)$. We want to decide V_{h-1}^* is an instance of our scheme or r^* is a random value. To solve this problem, we generate another identity tree T' of height $h-1$ with leaf level secrete key distribution X'. Using T' and T^*, we generate the distribution:

$$\begin{aligned}
V_h^* &= (vw(h-1, X^*, Q^*, T^*), vw(k-1, X', Q', T'), \\
&\quad r^*(Q' + s_\omega)P, K'(Q^* + s_\phi)P, y)
\end{aligned}$$

where $y \in_R G_1$ and $K' = K(h-1, X', Q', T')$. Now, we put V_h^* as input $\mathcal{D}_{AB_h}$. Let us first consider the case that V_h^* is an instance of A_h. In this case, it implies that V_{h-1}^* is an instance of F_{h-1}. Hence, we have

$$\begin{aligned}
& Prob[\mathcal{D}_{AB_h}(A_h = V_h^*) = 1] \\
= \;& Prob[\mathcal{D}_{AF_{h-1}}(F_{h-1} = V_{h-1}^*) = 1]
\end{aligned}$$

If V_h^* is an instance of B_h, it implies that V_{h-1}^* is an instance of A_{h-1}. We have

$$\begin{aligned} & Prob[\mathcal{D}_{AB_h}(B_h = V_h^*) = 1] \\ = \quad & Prob[\mathcal{D}_{AF_{h-1}}(A_{h-1} = V_{h-1}^*) = 1] \end{aligned}$$

Consequently,

$$\begin{aligned} & |Prob[\mathcal{D}_{AB_h}(A_h = V_h^*) = 1] \\ & \quad -Prob[\mathcal{D}_{AB_h}(B_h = V_h^*) = 1]| \\ = \quad & |Prob[\mathcal{D}_{AF_{h-1}}(F_{h-1} = V_{h-1}^*) = 1] \\ & \quad -Prob[\mathcal{D}_{AF_{h-1}}(A_{h-1} = V_{h-1}^*) = 1]| \end{aligned}$$

Hence, if $\mathcal{D}_{AB_h}$ can distinguish A_h and B_h, then $\mathcal{D}_{AF_{h-1}}$ can distinguish A_{h-1} and F_{h-1}.

In the case of (B_h and C_h), (D_h and E_h) and (E_h and F_h), we can use the similar proof method as in (A_h and B_h). If one of the pair is distinguishable in a polynomial time, we can construct a distinguisher $\mathcal{D}_{AF_{h-1}}$ to distinguish A_{h-1} and F_{h-1}.

- (C_h and D_h): If C_h and D_h can be distinguished by $\mathcal{D}_{CD_h}$ in polynomial time. Then $\mathcal{D}_{CD_h}$ can be used to distinguish two-party key from a random value in our scheme. Let $u = r_1(Q_R + s_\omega)P$, $v = r_2(Q_L + s_\phi)P$, $w = H_2(\hat{e}(P,P)^{r_1 r_2})$(or a randomly chosen from Z_q^*). If (u, v, w) is an instance of $C_h(D_h)$, then it is an instance of $A_2(F_2)$. McCullagh and Barreto show that the two-party key is indistinguishable from a random value. Please refer to [26] for details. **Q.E.D.**

Theorem 2 *Our scheme satisfies backward secrecy and forward secrecy.*

Proof (sketch):
In order to show that our scheme satisfies backward secrecy and forward secrecy, we only need to show that the view of the former(prospective) member to the current identity tree is exactly the same as the view of the passive adversary respectively.

We first consider forward secrecy. When a user $\mathcal{J}$ leaves the subgroup, the SGC erases the leaf node corresponding to the $U_{\mathcal{J}}$ from the identity tree and refreshes the identity tree as described in Section 3.4. Consequently, all the keys known to $U_{\mathcal{J}}$ are refreshed accordingly. Therefore, the view of $\mathcal{J}$ is exactly same as the view of the passive adversary.

Now we consider the backward secrecy. When a new user $\mathcal{J}$ is added to the subgroup, SGC updates the identity tree as described in Section 3.3, and consequently, the previous group key is changed. Therefore, the new user $\mathcal{J}$'s view is exactly same as the view of an passive adversary. This shows that the new user has exactly the same advantage of the old group key as an outsider. **Q.E.D.**

5 Comparison

In this section, we compared our scheme with Iolus and LKH approach and its variants as introduced in Section 1.

In Iolus, Scalability is achieved by splitting the large group into small groups. If a subgroup controller is failed, only its subgroup is affected. In our scheme, however, even the subgroup controller is failed, its subgroup will not be affected. Because any user in the subgroup can perform the functionality of the subgroup controller. This is a very desirable feature especially for the mobile and ad hoc networks.

Furthermore, each subgroup key in Iolus is generated by each group security intermediary (GSI). When the GSI is aborted, how to update the whole system is a difficult problem. Many other schemes also have the same problem [29, 31, 33, 34]. In our scheme all the keys used in the subgroup are generated by a group of KGCs in parallel. Even some of them do not work, it does not have any effect at all. Because the key generation/distribution task can be fulfilled by the remainder KGCs.

Compared with the LKH method and its variants, our protocol provides explicitly group member authentication. Since we use the identity tree to achieve this property.

In LKH method and its variants, the group controller has a heavy burden to carry out access control policy and maintain a huge key tree. Further, the group controller also has responsibilities to generate, distribute keys used in the group communication. As a result, with the growth of the communication group, the group controller becomes the single bottle neck of the system. When the group controller is not working, the whole communication group becomes vulnerable because the keys, which are the base of the group privacy, are not being generated and distributed. In our scheme, all of these problems are avoided by using a group of KGCs to fulfill these tasks.

In our scheme, however, the node in the identity tree is associated with three keys: private key, blinded key and key generation key. This makes the user storage in our scheme is larger than that in LKH approach and its variants. However the key generation key on the interior node can be computed by the user, so the user does not need to store it locally. Moreover, the blinded keys as well as the identity tree are the public information, KGCs may store them in a shared storage medium where the users can access to. Using this approach, we achieve the same user storage as in LKH method and its variants.

6 Conclusion

We have proposed an efficient, authenticated, scalable key agreement for large and dynamic multicast systems, which is based on the bilinear map. Compared with the previously published schemes in literature, we use an identity tree to achieve the authentication of the group member. Further, our scheme solve the scalability problem in multicast communications. Since a large group is divided into many small groups. Each subgroup is treated almost like a separate multicast group with its own subgroup key. All the keys used in each subgroup can be generated by a group of KGCs in parallel. The intuitively surprising aspect of this scheme is that, even the subgroup controller aborts, it does not affect the users in this subgroup. Because every user in the subgroup can act as a subgroup controller. This is a significant feature especially for the mobile and ad hoc networks. From the security analysis we can see that our scheme satisfies both forward and backward secrecy.

Acknowledgements

This work was supported by the National Natural Science Foundation of China No.90304007 and China National 973 Project No.2004CB318004.

References

[1] P. S. L. M. Barreto, H. Y. Kim, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," in *CRYPTO 2002*, LNCS 2442, pp. 354–368, 2002.

[2] P. S. L. M. Barreto, B. Lynn, and M. Scott, "On the selection of pairing-friendly groups," in *SAC'2003*, LNCS 3006, pp. 17–25, 2004.

[3] I. Blake, G. Seroussi, and N. Smart, *Elliptic Curves in Cryptography*, Cambridge Unversity Press, 2001.

[4] D. Boneh and X. Boyen, "Efficient selective-ID secure identity based encryption without random oracles," in *Eurocrypt 2004*, LNCS 3027, pp. 223–238, 2004.

[5] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *CRYPTO 2001*, LNCS 2139, pp. 213–229, 2001.

[6] D. Boneh and J. Katz, "Improved efficiency for CCA-secure cryptosystems built using identity-based encryption," in *CT-RSA 2005*, LNCS 3376, pp. 87–103, 2005.

[7] X. Boyen, "Multipurpose identity-based signcryption: a swiss army knife for identity-based cryptography," in *CRYPTO 2003*, LNCS 2729, pp. 382–398, 2003.

[8] R. Canetti, J. Garay, G. Itkis, K. Micciancio, M. Naor, and B. Pinkas, "Multicast security: a taxonomy and some efficient constructions," in *INFOCOM 99*, pp. 708–716, 1999.

[9] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption scheme," in *Eurocrypt 2003*, LNCS 2656, pp. 255–271, 2003.

[10] R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," in *Eurocrypt 2004*, LNCS 3027, pp. 207–222, 2004.

[11] I. Chang, R. Engel, D. Pendarakis, and D. Saha, "Key management for secure Internet multicast using boolean function minimization techniques," in *INFOCOM '99*, pp. 689–698, 1999.

[12] Y. Choie and E. Lee, "Implementation of tate pairing on hyperelliptic curves of genus 2," in *ICISC 2003*, LNCS 2971, pp. 97–111, 2004.

[13] S. E. Deering, "Multicast routing in internetworks and extended LANs," in *Proceedings of the ACM SIGCOMM '88*, pp. 55–64, Stanford, California, Aug. 1988.

[14] S. E. Deering, *Host Extensions for IP Multicasting*, RFC 1112, Aug. 1989.

[15] S. E. Deering, D. Estrin, D. Farinacci, V. Jacosen, L. Ching Gung, and L. Wei, "An architecture for wide-area multicasting," in *Proceedings of the ACM SIGCOMM '94*, pp. 126–135, London, Sep. 1994.

[16] R. Dutta, R. Barua and P. Sarkar, "Provably secure authenticated tree based key agreement," in *ICICS 2004*, LNCS 3269, pp. 92–104, 2004.

[17] S. D. Galbraith, K. Harrison, and D. Soldera, "Implementing the tate pairing," in *ANTS 2002*, LNCS 2369, pp. 324–337, 2002.

[18] C. Gentry and A. Silverberg, "Hierarchical ID-based cryptography," in *ASIACRYPT 2002*, LNCS 2501, pp. 548–566, 2002.

[19] F. Hess, "Efficient identity based signature schemes based on pairings," in *SAC 2002*, LNCS 2595, pp. 310–324, 2003.

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

[21] Y. Kim, A. Perrig, and G. Tsudik, "Tree-based group key agreement," *ACM Transactions on Information and System Security*, vol. 7, no. 1, pp. 60–96, Feb. 2004.

[22] B. Libert, J. J. Quisquater, *New Identity Based Signcryption Schemes from Pairing*, Cryptology ePrint Archive, Report 2003/023, available at http://eprint.iacr.org/2003/023.

[23] H. Lu, "A novel high-order tree for secure multicast key management," *IEEE Transactions on Computers*, vol. 54, no. 2, pp. 214–224, Feb. 2005.

[24] B. Lynn, *Authenticated Identity-Based Encryption*, Cryptology ePrint Archive, Report 2002/072, 2002

[25] D. A. McGrew and A. T. Sherman, *Key Establishment in large Dynamic Groups Using One-Way Function Trees*, Technical Report No. 0755, TIS Labs at Network Associates, Inc., Glenwood, MD, May 1998.

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

[27] V. S. Miller, "The weil pairing and its efficient calculation," *Journal of Cryptology*, vol. 17, no. 4, pp. 235–261, 2004.

[28] S. Mittra, "Iolus: a framework for scaclable secure multicasting," in *Proceedings of ACM SIGCOMM'97*, pp. 277–288, New York, 1997.

[29] A. Perrig, D. Song and J. D. Tygar, "ELK, a new protocol for efficient large group key distribution," *IEEE Symposium on Security and Privacy*, pp. 247–262, 2001.

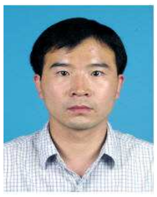
[30] M. Scott and P. S. L. M. Barreto, "Compressed pairings," in *CRYPTO 2004*, LNCS 3152, pp. 140–156, 2004.

[31] A. T. Sherman and D. A. McGrew, "Key establishment in large dynamic groups using one-way function trees," *IEEE Transations on Software Engineering,* vol. 29, no. 5, pp. 444–458, 2003.

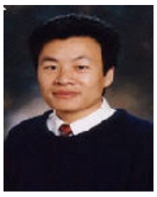
[32] M. Steiner, G. Tsudik, and M. Waidner, "Cliques: a new approach to group key agreement," *IEEE Conference on Distributed Computing Systems (ICDCS'98)*, pp. 380–387, 1998.

[33] D. Wallner, E. Harder, and R. Agee, *Key Management for Multicast: Issues and Architectures*, RFC 2627, Internet Engineering Task Force, June 1999.

[34] C. K. Wong and S. Lam, "Secure group communications using key graphs," in *SIGCOMM '98*, pp. 68–79, 1998.



Liming Wang is currently a Ph.D. candidate in the State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences. His research interests include secure electronic commerce, cryptography and network security.



Chuan-Kun Wu is currently a professor in Institute of Software, Chinese Academy of Sciences. Chuan-Kun Wu has served as a Program Chair for 2001, 2002 and 2003 International Workshop on Cryptology and Network Security. He is now a senior member of IEEE, a member of International Association for Cryptologic Research (IACR). He also serves as an Associate Editor for IEEE COMMUNICATIONS LETTERS. His research interests include spec- trum analysis of cryptographic properties of Boolean functions, design and analysis of network security protocols, mobile computing, secure electronic commerce, information hiding, cryptography and network security.