A Scalable Multicast Security Protocol in Hierarchy Structures

CHIN-CHEN CHANG$^{1,2}$, JUNG-SAN LEE$^2$, YA-FEN CHANG$^3$ and YI-PEI HSIEH$^2$

$^1$ Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan, 40724, R.O.C.
ccc@cs.ccu.edu.tw

$^2$ Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan, 621, R.O.C.
{ljs, hsiehyp}@cs.ccu.edu.tw

$^3$Department of Logistics Engineering and Management, National Taichung Institute of Technology, Taichung 404, Taiwan, R.O.C.
cyf@cs.ccu.edu.tw

Abstract: - The fundamental function of the network security protocols is to allow the authorized participant to communicate with others securely over the insecure network. Point-to-point packet transmission is the most common way of the communication over the network. However, it is not appropriate for many emerging applications such as message services, pay-TV, teleconference, or collaborate tasks. It is due to that these applications are structured on the group communication. Consequently, point-to-group or group-to-group packet transmission has become an important issue of the network in recent periods. In 2004, Aslan proposed a scalable multicast security protocol using a subgroup-key hierarchy. Aslan’s protocol allows the user to communicate with others efficiently. Nevertheless, we find that each communicating user in the system has to maintain many secret keys such that it is not convenient for all users. Besides, while a member joins or leaves the communicating group, lots of involved participants have to change their secret keys to confirm the forward secrecy and the backward secrecy. In this article, we propose an improved multicast security protocol which not only preserves the functionality of Aslan’s protocol but also possesses the performance better than other related works.

Key-Words: - multicast communication, scalability, secret key, hierarchy

1 Introduction

Engineers have proposed lots of security protocols for providing secure communications for network users recently. Among them, those of the multicast communications are regarded as the most critical ones. It is due to that more and more emerging applications, such as teleconference, pay-TV, collaborating tasks and message services, are based on the group communication. Traditional point-to-point communications do not suffice for users’ needs anymore. Point-to-group and group-to-group communications have become the important research issue in computer networks [2, 3, 4, 5, 7, 8, 10, 11].

So far, there are three main solutions, central control, distributed control and subgroup control, proposed for providing secure multicast communications and key distribution.

Central control: A central manager takes the responsibility for the security of the whole group and key distribution. However, this solution is not scalable for large groups. Besides, the failure of the central manager will make the communication of the whole group inactive [6].

Distributed control: All group members take the responsibility for key generation and the security of the whole group. Although this solution can get rid of the disadvantage of the central control solution, in which the failure of the central manager will make the group communication inactive, the scalability of the distributed control solution for large groups is worse than that of the central control one [4, 12, 13].

Subgroup control: The whole group is divided into several subgroups. Each subgroup is controlled by a subgroup manager. The scalability of this approach is better than those of the above two solutions. Furthermore, the failure of the single subgroup manager does not lead to the inactivity of the whole group communication [1, 9].

In 2004, Aslan presented a scalable multicast security protocol with the subgroup control solution using a hierarchy structure. Aslan’s protocol makes all group members be able to communicate with others efficiently. However,
we find that there exist two weaknesses in her multicast protocol. First, each group user has to keep lots of secret key. This is inconvenient for the involved participants. Second, while a member joins or leaves the group, lots of involved participants have to change their secret keys to confirm the forward secrecy and the backward secrecy. We consequently propose an improved version which not only preserves the functionality of Aslan’s protocol but also has the performance better than that of other schemes.

The rest of this paper is organized as follows. A review of Aslan’s protocol is described in Section 2, followed by our proposed multicast security protocol in Section 3. Discussions of our protocol and the comparisons between other related works and our scheme are shown in Section 4. Finally, we make some conclusions in Section 5.

2 Related works

Based on a hierarchy structure, Aslan proposed a scalable multicast security protocol with the subgroup control solution [1]. The whole structure of Aslan’s protocol is illustrated in Fig.1.

![Fig.1: the structure of Aslan’s protocol](image)

The main idea of his protocol is to divide the whole group into several subgroups. As shown in Fig.2, each subgroup i is formed with a hierarchy structure and is controlled by a subgroup manager SGMi, where i = 1, 2, ..., n and n is the number of subgroups. In Aslan’s protocol, each subgroup user has to keep many secret keys. For example, in Subgroup i, the user Ui has to store Ki(h, 1), Ki(h-1, 1), ..., Ki(0, 1), where h is the height of Subgroup i, and d is the maximum degree of each internal nodes. That is, each user must keep all secret keys of the path from the subgroup manager to himself/herself. The group manager GM shares a different secret key Ki(GS) with each SGMi. Besides, GM generates another secret key K(GS) shared between GM and SGMn+1. Next, GM computes EK(GS)[Knew(GS)] and EK(GS)| Knew(GS), where E[k][m] is the encryption algorithm to encrypt the message m with the secret key k. GM then broadcasts the computation results to all SGMi’s including the new one. While receiving the messages, the original SGMi’s retrieve the new secret key Knew(GS) by computing DK(GS) [EK(GS)| Knew(GS)], where D[k][m] is the decryption algorithm to decrypt the message m with the secret key k. On the other hand, the new subgroup manager retrieves the new secret key Knew(GS) by computing DK(GS)| EK(GS)| Knew(GS)]. Hence, the join operation of a new subgroup manager is completed.

Furthermore, several assumptions are made in the multicast system. First, the number of subgroups increases progressively. That is, SGMi’s are not permitted to leave the group. Second, while a new member wants to join the group, GM must take the responsibility for finding an empty place for him/her. If all subgroups are full, GM has to create a new subgroup. Third, all SGMi’s and group members are assumed to be trustworthy.

There are four main operations in Aslan’s protocol: subgroup manager join operation, message broadcast operation, member join operation and member leave operation. The details of these operations are described as follows.

Subgroup manager join operation:

If a new subgroup manager SGMn+1 joins the group, GM has to change the secret key K(GS) to Knew(GS). Besides, GM generates a new secret key K(GS)| Knew(GS) shared between GM and SGMn+1. Next, GM computes EK(GS)| Knew(GS)] and EK(GS)| Knew(GS)], where E[k][m] is the encryption algorithm to encrypt the message m with the secret key k. GM then broadcasts the computation results to all SGMi’s including the new one. While receiving the messages, the original SGMi’s retrieve the new secret key Knew(GS) by computing DK(GS)| EK(GS)| Knew(GS)], where D[k][m] is the decryption algorithm to decrypt the message m with the secret key k. On the other hand, the new subgroup manager retrieves the new secret key Knew(GS) by computing DK(GS)| EK(GS)| Knew(GS)]. Hence, the join operation of a new subgroup manager is completed.
Message broadcast operation:
While a message M needs to be broadcasted, GM generates a new secret key K(msg) to encrypt M. GM then computes the followings,
\[ E_{K_G}(M) \] and
\[ E_{K_G}(K(msg)) \], and broadcasts the computation results to all SGMi's. After receiving the messages, each SGMi computes
\[ K(msg) = D_{K_G}(E_{K_G}(K(msg))) \] and
\[ E_{K_i}(0, 1)[K(msg)] \],
where \( K(0, 1) \) is the common secret key shared between all members in Subgroup i. SGMi then broadcasts the two following messages to all subgroup users:
\[ E_{K_i}(0, 1)[K(msg)] \] and
\[ E_{K_i}(1, 1)[K(msg)] \].
Therefore, each subgroup user can obtain the message as follows:
\[ K(msg) = D_{K_i}(E_{K_i}(0, 1)[K_i](0, 1)) \] and
\[ M = D_{K_i}(E_{K_i}(1, 1)[K_i](1, 1)) \].

Member join operation:
While a new user Ud wants to join the communication group, GM has to find an appropriate place and generate a secret key \( K_i(0, 1) \) for Ud. As illustrated in Fig.3, all secret keys of the path from SGMi to Ud must be modified to preserve the backward secrecy. The secret key \( K_i(j, 1) \) will be changed to \( K_i(new)(j, 1) \), where \( j = 0, 1, \ldots, h-1 \). These new secret keys have to be sent to the involved participants securely. Hence, SGMi computes and broadcasts the following messages to all subgroup members:
\[ E_{K_i}(0, 1)[K_i(new)(0, 1)] \],
\[ E_{K_i}(1, 1)[K_i(new)(1, 1)] \],
\[ \cdots \]
\[ E_{K_i}(h-1, 1)[K_i(new)(h-1, 1)] \].
\( K_i(new)(j, 1) \) is required by all subgroup members, \( K_i(new)(1, 1) \) is required by the first \( d^{th} \) members, and so forth until \( K_i(new)(h-1, 1) \) is only required by the first d subgroup members. Furthermore, Ud will obtain all needed secret keys by computing
\[ D_{K_i}(E_{K_i}(0, 1)[K_i(new)(0, 1), K_i(new)(1, 1), \ldots, K_i(new)(h-1, 1)]) \].
Here, member join operation is completed.

Member leave operation:
While a user Ud leaves Subgroup i, as illustrated in Fig.4, all secret keys of the path from SGMi to Ud must be modified to preserve the forward secrecy. SGMi has to compute and broadcast the following messages to all subgroup members:
\[ E_{K_i}(1, f)[K_i(new)(0, 1)] \],
\[ E_{K_i}(2, f)[K_i(new)(1, 2)] \],
\[ \cdots \]
\[ E_{K_i}(h, f)[K_i(new)(h, 1)] \],
where \( f = 2, 3, \ldots, d \). Hence, all involved participants can obtain the required secret keys securely. That is, member leave operation is completed.

3 The Improved Multicast Security Protocol
Although Aslan’s multicast protocol is efficient and scalable, members in the multicast system have to keep lots of secret keys. Furthermore, the operations of member join and member leave make many involved participants have to change their required secret keys to confirm the forward secrecy and the backward secrecy. Hence, we
propose an improved version with Lagrange Interpolating Polynomial. Instead of maintaining lots of secret keys, each member in the multicast system needs to keep only one secret key. Besides, a bulletin board is adopted in our protocol. Note that only GM and the legal SGMi’s can modify and update the bulletin board.

The same as Aslan’s scheme, the whole group is divided into several subgroups. As shown in Fig.5, each subgroup formed with a hierarchy structure of height h is controlled by a subgroup manager SGMi, where i = 1, 2, …, n and n is the number of the subgroups. Every node in the hierarchy structure of Subgroup i is assigned a unique identity IDi(b, j), where b = 0, 1, …, h, j = 1, 2, …, d, and d is the maximum degree of the internal node in the hierarchy tree.

Fig.5: the hierarchy structure of Subgroup i

There are four operations in our protocol: subgroup manager join operation, member join operation, member leave operation, and message broadcast operation. The first operation is inherited from Aslan’s protocol for its practicability. The notations used in our protocol are the same as those in Aslan’s protocol. The details of other three operations are described as follows.

Definition of Lagrange Interpolating Polynomial:

Let (x1, y1), (x2, y2), …, (xt, yt) be t points on two-dimensional plane [14]. N is a prime. a0, a1, …, and at are integers within [1, N-1]. We can obtain the polynomial by computing

\[ f(x) = \sum_{j=1}^{t} y_j \prod_{i=1, i \neq j}^{t} \frac{x-x_i}{x_j-x_i} \mod N \]

\[ = a_0 + a_1x + a_2x^2 + … + a_tx^t \mod N, \]

where \( y = f(x) \).

The bulletin board setup:

At first, GM takes the responsibility for constructing the bulletin board as shown in Table 1. For each internal node IDi(b, j) in Subgroup i, SGMi bottom-up computes a corresponding polynomial IDi(b, j)_P(x) as follows, where b = h-1, h-2, …, 0 and j = 1, 2, …, d.

Step 1: Computes d hash values

\[ h_ib1 = h(K_i(b+1, 1), ID_i(b, j), ID_i(b+1, 2), …, ID_i(b+1, d)), \]

\[ h_ib2 = h(K_i(b+1, 2), ID_i(b+1, 1), ID_i(b, j), ID_i(b+1, 3), ID_i(b+1, 4), …, ID_i(b+1, d)), \]

\[ N \]

\[ h_ibd = h(K_i(b+1, d), ID_i(b+1, 1), ID_i(b+1, 2), …, ID_i(b+1, d-1), ID_i(b, j)). \]

Step 2: Performs Lagrange Interpolating Polynomial on these coordinates (h_ib1, K_i(b, j)), (h_ib2, K_i(b, j)), …, and (h_ibd, K_i(b, j)), to obtain the polynomial

\[ ID_i(b, j)_P(x) = a_0 + a_1x + … + a_dx^d, \]

where a0, a1, …, ad are integers.

Step 3: Publishes all identities of nodes and their corresponding polynomials on the bulletin board.

Table 1: the example of the bulletin board

<table>
<thead>
<tr>
<th>SGMi</th>
<th>Node</th>
<th>Polynomial</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDi(b, 1)</td>
<td>IDi(b, 1)_P(x)</td>
<td></td>
</tr>
<tr>
<td>IDi(b, 2)</td>
<td>IDi(b, 2)_P(x)</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>IDi(b, d)</td>
<td>IDi(b, d)_P(x)</td>
<td></td>
</tr>
</tbody>
</table>

Fig.6: the example of member join operation of our protocol

Member join operation:

While a new member Ud wants to join the communication group, GM has to find a suitable place and generate a secret key for him/her. As illustrated in Fig.6, all secret keys of the path from SGMi to Ud must be modified to confirm the backward secrecy. The secret key K_i(b, 1) must be changed, where b = 0, 1, …, h-1. All involved internal nodes’ polynomials published on the bulletin board will be updated by SGMi. That is, SGMi has to bottom-up perform...
Step 1: Computes $d$ hash values
$h_{ib1} = h(K_{ib}(b+1, 1), ID_i(b, 1), ID_i(b+1, 2), \ldots, ID_i(b+1, d))$
$h_{ib2} = h(K_{ib}(b+1, 2), ID_i(b+1,1),ID_i(b,1), ID_i(b+1,3), ID_i(b+1,4), \ldots, ID_i(b+1, d))$

Step 2: Performs Lagrange Interpolating Polynomial (h-1) times to reconstruct (h-1) involved polynomials. For each involved internal node ID_i(b, 1), where $b = h-1$, h-2, $\ldots$, 0 (i.e. the internal nodes on the path from U_d to SGM_i), SGM_i executes the followings.

Member leave operation:
While a user U_d leaves Subgroup i, as illustrated in Fig.7, all secret keys of the path from SGM_i to U_d must be modified to confirm the forward secrecy. The secret key $K_{ib}(b, 1)$ must be changed, where $b = 0, 1, \ldots, h-1$. All involved internal nodes’ polynomials published on the bulletin board will be modified by SGM_i. That is, SGM_i has to bottom-up perform Lagrange Interpolating Polynomial (h-1) times to reconstruct (h-1) involved polynomials.

For each involved internal node $ID_i(b, 1)$, where $b = h-1, h-2, \ldots, 0$ (i.e. the internal nodes on the path from U_d to SGM_i), SGM_i executes the followings.

Step 1: Computes $d$ hash values
$h_{ib1} = h(K_{ib}(b+1, 1), ID_i(b, 1), ID_i(b+1, 2), \ldots, ID_i(b+1, d-1))$
$h_{ib2} = h(K_{ib}(b+1, 2), ID_i(b+1,1),ID_i(b,1), ID_i(b+1,3), ID_i(b+1,4), \ldots, ID_i(b+1, d-1))$

Step 2: Performs Lagrange Interpolating Polynomial on these coordinates (h_{ib1}, K_{ib}(b, 1)), (h_{ib2}, K_{ibnew}(b, 1)), \ldots, and (h_{ib(d-1)}, K_{ibnew}(b, 1)), to obtain the polynomial
$ID_i(b, 1) = a_0 + a_1x + \ldots + a_{d-1}x^{d-1}$
where $a_0, a_1, \ldots, a_{d-1}$ are integers.

Step 3: Updates the modified information on the bulletin board as shown in Table 1.
4 Discussions
In the following, several comparisons between other related works and our proposed scheme are presented. At first, we define the notations used in Table 2.

- **n**: the total number of subgroups
- **m**: the total number of subgroup members
- **d**: the maximum degree of each internal node in the hierarchy structure
- **h**: the height of each subgroup, i.e. \( m = d^h \)
- **h1**: \( h_1 = \log_d (n \times m) \)
- **SGM**: the subgroup manager
- **in**: the involved member
- **non**: the non-involved member
- **K**: the symmetric encryption operation
- **P**: the operation of constructing the polynomial by Lagrange Interpolating Polynomial
- **Y**: the operation for obtaining \( y \) with input \( x \), where \( y = f(x) \)

Obliviously, the computation load of constructing the polynomial by Lagrange Interpolating Polynomial is quite lighter than that of performing the symmetric encryption operation. It is due to the fact that the construction of LIP polynomial is based on simple multiplication while that of encryption function such as DES and FEAL, depends on lots of round operations including permutation, key transformation, expansion, substitution, and modulus [15-19]. As shown in Table 2, in the case for a new member to join the communication group, the computation loads of each subgroup manager in our protocol are lighter than those in Aslan’s and Wong et al.’s protocol but similar to those in Iolus protocol. As for member join operation, the computation loads of each subgroup member in our protocol are quite lighter than those in other related works.

Table 2: the comparisons between other related works and our protocol

Furthermore, in case that a member leaves the communication group, the computation loads of both subgroup managers and subgroup members in our protocol are far lighter than those in other related works. Nevertheless, involved users of our scheme have to perform \( h \times Y + 1 \times K \) operations for retrieving a broadcast message while those of other works only need to execute \( 1 \times K \) operation. We therefore conclude that our scheme is more suitable for the multicast system with high mobility than other related works.

5 Conclusion
Aslan proposed a scalable multicast security protocol for providing secure communications with a hierarchy structure in 2004. However, we find that each subgroup member in the multicast system has to keep many secret keys in his/her databases. This is so inconvenient for all users. Furthermore, while a member joins or leaves the communicating group, lots of involved participants have to modify their secret keys to confirm the forward and backward secrecy. We thus present an improved version based on the structure of Aslan’s scheme. Each group member in our multicast system needs to keep only one secret key in his/her database. As shown in Table 2, our proposed protocol not only preserves the functionality of Aslan’s protocol but also possesses the performance better than that of other related works. Specifically, our scheme is more suitable for the multicast system with high mobility than other works.

References:


