A Multicasting System based on the Elliptic Curve Crypto-system

Ming-Chang Wu[1]
Institute of Electrical Engineering,
National Chung Cheng University
WMC@wireless.ee.ccu.edu.tw

Tzer-Shyong Chen[2]
Department of Information Management,
Tunghai University
arden@mail.thu.edu.tw

Jun-Lin Liu[3]
Computer & Communications Research Lab,
Industrial Technologies Research Institute
jun-lin@itri.org.tw

Jyh-Horng Wen[4]
Institute of Electrical Engineering,
National Chung Cheng University
wen@ee.ccu.edu.tw

Abstract

With the growth of Internet technology and its popularization, numerous users can share a single package in the multicasting system, making networking more efficient and also reduces the bandwidth. The system is, however, constructed by different groups, making secret sharing complex. Therefore, the construction of a secure multicasting system through the use of information security protocol is an important topic of discussion.

All members under the same multicasting system share a group key. Therefore, all broadcasted messages can be enciphered and deciphered using this key. Consequently, only legitimate members have access to the messages. However, if any member were to join or leave the group, the group key must be changed to ensure environment security. Thus, the security of the multicasting system is absolutely dependent on the security of the group key. It is also the biggest problem in the construction of a secure multicasting system.

Kuen-Pin Wu proposed the use of the secure filter method to solve the security problems of the system, but it was still lacking. We shall, however, use the elliptic curve to construct a new framework to solve this problem and then compared it with the secure filter.

Keywords: Multicasting, elliptic curve cryptosystem, access control, secure filter, group key

I. An Introduction to Kuen-Pin Wu's proposal

Kuen-Pin Wu [1] proposed in the year 2000, the usage of secure filter in the multicasting system to solve its security problems and ensure the safety of the group key. This is described in details below.

The secure filter is built on a polynomial x with restrictions $GF(p)$, where p is a public natural prime number. If $S = \{k_1, k_2, \ldots, k_n\}$ is the set of the secret keys of all the members in the group, while gk is the group key, and the tuple of S, and gk, are all contained in the integer Zp, then the CA shall first retrieve the hash function from the secret keys of all the members of the group, and then construct the secure filter as:

$$SF(x) = (x - h(k_1))(x - h(k_2))\cdots (x - h(k_n)) + gk \quad \cdots(1)$$

in which h is a randomly selected one-way hash function. From the secure filter polynomial $SF(x)$, it is known that when $k_i \in S$, $SF(h(k_i)) = gk$. This means that the group members can enter their secret key k_i, and through the processing of the secure filter, retrieve the group key gk; as for the non-multicasting members, because of their inability to make $(x - h(k_i))$ of the secure filter polynomial $SF(x)$ zero, are unable to obtain the group key gk. Besides, for safety reasons, when sending the secure filter polynomial $SF(x)$ through the internet, the polynomial can be send in the expanded form $a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$. Thus avoiding the risk of having non-multicasting members directly using the secure filter polynomial $SF(x)$

to retrieve the group key gk.

When there is a change in the members of the multicasting system, the following operations are performed:

1. When a new member joins the multicasting system

Suppose the secret key of user u_{n+1} is k_{n+1}, then, k_{n+1} is a tuple in the set of the secret keys S' of the new members. When user u_{n+1} joins the multicasting system, the CA has to change to a new group key gk' in order to construct a new secure filter polynomial as:

$$SF'(x) = (x - h(k_1))(x - h(k_2))\cdots(x - h(k_n))(x - h(k_{n+1})) + gk' \quad \cdots(2)$$

The CA shall then send $SF'(x)$ and the one-way hash function h to the group members. The group members on receiving $SF'(x)$, shall enter their secret keys $k_i, 1 \le i \le n+1$, and through the processing of $SF'(x)$, obtain the group key gk'. Non-members are unable to obtain a message through decryption.

2. When a member leaves the multicasting system

Suppose that user u_l has secret key $k_l, 1 \le l \le n$, then k_l is a tuple of the multicasting member's secret key set S. When the user u_l leaves the multicasting system, the CA has to change to a new group key gk'' in order to construct a new secure filter polynomial as:

$$SF''(x) = [(x - h(k_1))(x - h(k_2))\cdots(x - h(k_n))]/(x - h(k_l)) + gk'' \quad \cdots(3)$$

The CA shall then send $SF''(x)$ and the one-way hash function h to the group members. The members on receiving $SF''(k_i)$, shall enter their secret keys k_i, $k_i \in S - k_l$, and through the processing of $SF''(h(k_i))$, obtain the group key gk''. Non-members are unable to obtain the message through decryption.

II. An Introduction to the Elliptic Curve Cryptosystem

The general equation for the elliptic curve is $y^2 = x^3 + ax + b \mod p$, p is a natural prime number, and the value of a, b should satisfy the discriminant $D = 4a^3 + 27b^2 \neq 0 \mod p$. Only then could $y^2 = x^3 + ax + b \mod p$, be used as the decrypting elliptic curve [2].

Before we introduce the addition operation of the elliptic curve [3-5], we need to first introduce a special point *O*, known as the point of infinity and it satisfies the following properties:

1. If *P*, Q are two points on the elliptic curve, *O* is the point of infinity, then *P* ＋O＝*O*+*P*=*P*.
2. *O*＝-*O*.
3. If *Q* is not equal to point (-*P*), then *P*＋*Q*＝*O*.
4. If *P*≠*O*, *Q*≠*O*, then *P*＋*Q*＝-*R*.

According to the addition operation of the elliptic curve, if there are two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on the elliptic curve, and if *P*≠-Q, then $P + Q = (x_3, y_3)$, $x_3 = \lambda^2 - x_1 - x_2 \mod m$, $y_3 = \lambda(x_1 - x_3) - y_1 \mod m$, where

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & , if \quad P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1} & , if \quad P = Q \end{cases}$$

If there is a point *G* on the elliptic curve, and this point is the base point, then the operation on *nG* has the following properties, 1*G*=*G*, 2*G*=*G* ＋ *G*, 3*G*=*G* ＋ *G* ＋ *G*=2*G* ＋ *G*, ... , (*n*-1)*G*=*G*＋*G*+ ... ＋*G* with a total of (*n*-1) *G*, *nG*=*O*, (*n*＋1)*G*=*G*. Thus the *nG* is *n*×*G* formal, meaning addition operations of the elliptic curve, and not the general multiplication operation, are continuously performed on *n* number of *G*'s

For example: A, B both have chosen to use the elliptic curve $y^2 = x^3 + x + 6$ for communicating, taking *p* as 11, then $D = 4a^3 + 27b^2 \mod 11 = 8 \mod 11 \neq 0$, hence it is proved that the points on the elliptic curve are (2, 4), (2, 7), (3, 5), (3, 6), (5, 2), (5, 9), (7, 2), (7, 9), (8, 3), (8, 8), (10, 2), (10, 9). If A, B both chose the point (2, 7) as *G* (Generator point), and performs the addition operation of the elliptic curve, then *G*=(2, 7), 2*G*=*G*＋*G*=(5, 2), 3*G*=2*G* ＋*G*=(8, 3), 4*G*=(10, 2), 5*G*=(3, 6), 6*G*=(7, 9), 7*G*=(7, 2), 8*G*=(3, 5), 9*G*=(10, 9), 10*G*=(8, 8), 11*G*=(5, 9), 12*G*=(2, 4), 13*G*=*O*, 14*G*=(2, 7).

III. The Multicasting Secure Filter System

1. The construction of the multicasting secure filter system

Step 1: On the elliptic curve $y^2 = x^3 + ax + b \mod p$, select a point G that satisfies the $D = 4a^3 + 27b^2 \neq 0 \mod p$ condition as the base point.

Step 2: p is a public natural prime number, while $S = \{k_1, k_2, \ldots, k_n\}$ is a set of the secret keys of all the group member, gk is the group key, and the tuple of S, and gk, are all contained in the integer Zp, then the multicasting elliptic curve secure filter system constructed by the CA shall be:

$$ECF(x) = (x - f(k_1G))(x - f(k_2G))\cdots (x - f(k_nG)) + gk \mod p \quad \cdots(4)$$

where $k_iG = (x_i, y_i)$ and $f(k_iG)$ is $x_i \oplus y_i$, for safety reasons, when sending the secure filter polynomial $ECF(x)$ through the internet, the polynomial can be send in the expanded form, $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$.

Step 3: The CA shall then send $ECF(x)$ and point G, to the group members.

2. The retrieval of the group key gk by the group members

On receiving $ECF(x)$, the group members shall each enter their secret keys k_i, and through the processing of $ECF(f(k_iG))$, $k_i \in S$, obtain the group key gk, and retrieve the message through decryption. Non-members are unable to perform the above operations.

IV. The entrance and the departure of members in the system

1. The entrance of new members into the system

Suppose the secret key of user u_{n+1} is k_{n+1}, then, k_{n+1} is a tuple in the set of the secret keys S' of the new members. When user u_{n+1} joins the multicasting system, the CA has to change to a new group key gk' in order to construct a new secure filter polynomial:

$$ECF'(x) = (x - f(k_1G'))(x - f(k_2G'))\cdots (x - f(k_nG'))(x - f(k_{n+1}G')) + gk' \mod p \cdots(5)$$

The CA shall then send $ECF'(x)$ and point G' to the group members. The group members on receiving $ECF'(x)$, shall enter their secret keys k_i, $1 \le i \le n+1$, and through the processing of $ECF'(f(k_iG'))$, obtain the group key gk'. Non-members are unable to obtain a message through decryption.

2. The departure of members from the system

Suppose that user u_l has secret key k_l, $1 \le l \le n$, then k_l is a tuple of the multicasting member's secret key set S. When the user u_l leaves the multicasting system, the CA has to change to a new group key gk'' in order to construct a new secure filter polynomial as:

$$ECF''(x) = (x - f(k_1G''))(x - f(k_2G''))\cdots (x - f(k_nG''))/(x - f(k_lG'')) + gk'' \mod p \quad \cdots(6)$$

The CA shall then send $ECF''(x)$ and G'' to the group members. The members on receiving $ECF''(x)$, shall enter their secret keys k_i, $k_i \in S - k_l$, and through the processing of $ECF''(f(k_iG''))$, obtain the group key gk''. Non-members are unable to obtain the message through decryption.

V. Security Analysis

(i) There are two security drawbacks in Wu's proposal

1. Drawback 1

If the multicasting system has only two members, then the secure filter polynomial shall be:

$$SF(x) = (x - h(k_1))(x - h(k_2)) + gk \mod p \cdots(7)$$

Member u_1 can use his secret key k_1 to enter the secure filter polynomial $SF(x)$ and obtain the group key gk, and then calculate $h(k_2) = [SF(0) - gk]/h(k_1)$. When member u_1 leaves the system, as long as u_2 is still a member, the departing member u_1 can obtain the group key gk' through the new secure filter polynomial

$$SF'(x) = (x - h(k_2))\cdots(x - h(k_n)) + gk' \quad \cdots(8)$$

Hence, the secure filter polynomial system becomes unsafe.

2. Drawback 2

If among the n system members, $n-1$ members conspire, by taking $x = 0$, they shall be able to obtain from the secure filter polynomial,

$$SF(0) = (h(k_1))(h(k_2))\cdots(h(k_n)) + gk \quad \cdots(9)$$

The $n-1$ members can use their secret key k_i to enter the secure filter polynomial (1) and obtain the group key gk, and then calculate

$$h(k_i) = [SF(0) - gk]/[(h(k_1)(h(k_2)\cdots (h(k_n)/h(k_i)] \quad \cdots(10)$$

Thus, they shall be able to know the $h(k_i)$, of the remaining member. When the conspiring members leaves the system, as long as u_i is still a member of the new system, any of the conspiring members or any non-members who knows the message, can obtain the group key gk'' through the new secure filter polynomial,

$$SF''(x) = (x - h(k_1))(x - h(k_2))\cdots(x - h(k_i))\cdots (x - h(k_n)) + gk'' \mod p \quad \cdots(11)$$

Hence, the secure filter polynomial system becomes unsafe.

(ii) Reform Strategy

1. The first method

When there is entrance of new members or departure of members, at the time of constructing a new secure filter polynomial, the one-way hash function h should be changed to h', and h' becomes the new one-way hash function. Thus, the new secure filter polynomial becomes:

$$SF'(x) = (x - h'(k_1))(x - h'(k_2))\cdots (x - h'(k_n)) + gk' \quad \cdots(12)$$

Since the new one-way hash function h of the new secure filter polynomial isn't the h of the former secure filter polynomial, non-members of the new system are unable to decode the new polynomial and obtain the new group key gk'.

2. The second method

When there is entrance of new members or departure of members, at the time of constructing a new secure filter polynomial, a virtual member can be added to the group, with secret key as k_s, and secure filter polynomial as

$$SF''(x) = (x - h(k_1))(x - h(k_2))\cdots(x - h(k_n))(x - h(k_s)) + gk'' \quad \cdots(13)$$

where k_s is a random number contained in the integer Zp. If in the new polynomial, the group has n members, and there are $n-1$ members conspiring, although they can, from the secure filter polynomial (13), make $x = 0$, and obtain:

$$SF''(0) = (h(k_1))(h(k_2))\cdots(h(k_n))(h(k_s)) + gk'' \quad \cdots(14)$$

From the $n-1$ members, a member can enter his secret key k_i into the secure filter polynomial (13) and obtain the group key gk'' and then calculate:

$$h(k_i)h(k_s) = [SF(0) - gk]/[(h(k_1)(h(k_2)\cdots (h(k_n)/h(k_i)h(k_s)] \quad \cdots(15)$$

From $h(k_i)h(k_s)$, because the virtual member's $h(k_s)$ is unknown, the $h(k_i)$, of the remaining member cannot be obtained. Therefore non-new members are unable to retrieve the group key gk'' from the new secure filter polynomial $SF''(x)$.

3. The third method

Use the multicasting elliptic curve cryptosystem to construct a new multicasting elliptic curve secure filter polynomial:

$$ECF'(x) = (x - f(k_1G'))(x - f(k_2G'))\cdots (x - f(k_nG')) + gk' \mod p \quad \cdots(16)$$

When there are entrance or exit of members from the group, although the attackers, can obtain $ECF(0)$ and the group key, gk, but for the obtained $f(k_iG)$, because $ECF(x)$ has already been changed to $ECF'(x)$, and G to G', the attacker is unable to log in through $ECF'(x)$ and obtain the group key gk'.

We summarize Wu′s method and comparison of the security of the proposal in the table 1.

VI. Improvement on Operation speed

When there is entrance or departure of members from the system, the CA will need to construct a new secure filter polynomial, causing a huge operation overload and slow operation speed, which in turn, consumes a lot of time. Hence, a group-division method has been proposed in order to save computation time and reduce computation load, by dividing all the group members into sub-groups, forming unequal sub-groups; each of these sub-groups shall have their own secure filter polynomial as well as different group keys.

Suppose that the system has n members, these members can be divided into m subgroups, and each subgroup shall have different group keys $sgk_1, sgk_2, \ldots, sgk_m$, if

$S=\{k_1,k_2,\ \ldots,k_n\}$ is the set of secret keys of all the members, and gk is the group key, then the CA can build the secure filter polynomial in accordance to the different subgroups as:

$$\begin{aligned}
ECF_1(x)&=(x-f(k_1G_1))(x-f(k_2G_1))\cdots(x-f(k_{\lfloor n/m\rfloor}G_1))\\
&\quad+sgk_1 \mod p_1\\
ECF_2(x)&=(x-f(k_{\lfloor n/m\rfloor+1}G_2))\cdots(x-f(k_{2\lfloor n/m\rfloor}G_2))\\
&\quad+sgk_2 \mod p_2 \qquad\cdots(17)\\
&\vdots \qquad\qquad\qquad \vdots\\
ECF_m(x)&=(x-f(k_{(m-1)\lfloor n/m\rfloor+1}G_m))\cdots(x-f(k_nG_m))\\
&\quad+sgk_m \mod p_m\\
ECF_{m+1}(x)&=(x-f(sgk_1G_{m+1}))(x-f(sgk_2G_{m+1}))\cdots\\
&\quad(x-f(sgk_mG_{m+1}))+gk \mod p_{m+1}
\end{aligned}$$

In the equation, p_i is a heterogeneous prime number, G_i satisfies a base point on the elliptic curve $E_{p_i}(a_i,b_i)$, where $1\le i\le m+1$.

The CA shall then send the different subgroup's secure filter polynomial $ECF_i(x)$ and G_i, $1\le i\le m+1$, to the members of the subgroups. Each subgroup members on receiving $ECF_i(x)$ and G_i, $1\le i\le m+1$, can enter their subgroup keys sgk_i, $1\le i\le m$ and obtain the parent-group's group key gk.

For example, if there are a total of nine group members, $n_1,n_2,\ \ldots,n_9$, and $S=\{k_1,k_2,\ \ldots,k_9\}$ is the set of secret keys of the group members, if the group is divided into subgroups sg_1,sg_2,sg_3, where n_1,n_2,n_3 belong to sg_1, n_4,n_5,n_6 belong to sg_2, n_7,n_8,n_9 belong to sg_3, then the CA can build the secure filter polynomial in accordance to the different subgroups as:

$$\begin{aligned}
ECF_1(x)&=(x-f(k_1G_1))(x-f(k_2G_1))\\
&\quad(x-f(k_3G_1))+sgk_1 \mod p_1\\
ECF_2(x)&=(x-f(k_4G_2))(x-f(k_5G_2))\\
&\quad(x-f(k_6G_2))+sgk_2 \mod p_2 \qquad\cdots(18)\\
ECF_3(x)&=(x-f(k_7G_3))(x-f(k_8G_3))\\
&\quad(x-f(k_9G_3)+sgk_3 \mod p_3\\
ECF_4(x)&=(x-f(sgk_1G_4))(x-f(sgk_2G_4))\\
&\quad(x-f(sgk_3G_4))+k \mod p_4
\end{aligned}$$

The CA shall send the different subgroup's secure filter polynomial $ECF_i(x)$ and G_i, $i=\{1,2,3,4\}$, to each of the subgroup members. The subgroup members can then enter their secret keys sgk_i, $i=\{1,2,3\}$ and obtain the parent-group's group key gk.

When a member n_i leaves the system, the CA shall first find out which subgroup sg_i, $1\le i\le m$, the departing member belonged to, and then change the G_i of that subgroup to G_i' and the new group key to sgk_i', and then construct a new secure filter polynomial $ECF_i'(x)$, $1\le i\le m+1$. Like in the above example, member n_2 of subgroup sg_1 has left the system. The CA shall then build the secure filter polynomial in accordance to the different subgroups as:

$$\begin{aligned}
ECF_1'(x)&=(x-f(k_1G_1'))(x-f(k_3G_1'))\\
&\quad+sgk_1' \mod p_1\\
ECF_2(x)&=(x-f(k_4G_2))(x-f(k_5G_2))\\
&\quad(x-f(k_6G_2))+sgk_2 \mod p_2 \qquad\cdots(19)\\
ECF_3(x)&=(x-f(k_7G_3))(x-f(k_8G_3))\\
&\quad(x-f(k_9G_3))+sgk_3 \mod p_3\\
ECF_4'(x)&=(x-f(sgk_1'G_4'))(x-f(sgk_2G_4'))\\
&\quad(x-f(sgk_3G_4'))+gk' \mod p_4
\end{aligned}$$

The CA shall send the different new secure filter polynomials $ECF_i'(x)$ and G_i', $i=\{1,4\}$, to the their subgroups. Each member of the subgroup can enter their secret key and obtain the parent-group's group key gk'.

When a new member u_{n+1} joins the system, the CA shall put the new member in one of the subgroups sg_i, $1\le i\le m$, then change the G_i of that subgroup to G_i'' and the subgroup key to sgk_i'', in order to build a new subgroup secure filter polynomial $ECF_i''(x)$. Like in the above example, user u_{10} is a new member, if the subgroup he'd been put into is sg_2, then the CA shall build the secure filter polynomial in accordance to the different subgroups as:

$$\begin{aligned}
ECF_1(x)&=(x-f(k_1G_1))(x-f(k_2G_1))(x-f(k_3G_1))\\
&\quad+sgk_1 \mod p_1 \qquad\cdots(20)\\
ECF_2''(x)&=(x-f(k_4G_2''))(x-f(k_5G_2''))\\
&\quad(x-f(k_6G_2''))(x-f(k_{10}G_2''))+sgk_2'' \bmod p_2\\
ECF_3(x)&=(x-f(k_7G_3))(x-f(k_8G_3))(x-f(k_9G_3))\\
&\quad+sgk_3 \mod p_3\\
ECF_4''(x)&=(x-f(sgk_1G_4''))(x-f(sgk_2''G_4''))\\
&\quad(x-f(sgk_3G_4''))+gk'' \mod p_4
\end{aligned}$$

The CA shall send the different group's new secure filter polynomial $ECF_i''(x)$ and G_i'', $i=\{2,4\}$, to the different members. Each member can enter their secret key and obtain the

parent group's group key gk''.

VII. A probe into the Operating time Comparison of time complexity:

1. Parameter definitions

(1). T_{MUL} : the time required to perform a 1024 bit multiplication operation.

(2). T_{ADD} : the time required to perform a 1024 bit addition operation.

(3). T_{SUB} : the time required to perform a 1024 bit subtraction operation.

(4). T_H : the time required to perform a 160 bit hash function operation.

(5). $T_{EC\text{-}MUL}$: the time required to perform a 160 bit elliptic curve multiplication operation.

(6). $x - f(k_i G_i)$ sets all as 160 bits using the elliptic curve related parameters

(7). T_{ADD}、T_{SUB} and *XOR*: No time complexity calculation is done on this.

From references [6], it is known that

(8). $T_{EC\text{-}MUL} \approx 29\ T_{MUL}$.

2. Analysis of the polynomial operating time of Wu's polynomial $SF(x)$ and the polynomial $ECF(x)$ used herein

(1). The production of the polynomials $SF(x)$ and $ECF(x)$:

$$SF(x) = (x - h(k_1))(x - h(k_2))\cdots(x - h(k_n)) + gk \mod p$$

The required computation time is $n\ T_H + (n\text{-}1)\ T_{MUL}$;

$$ECF(x) = (x - f(k_1 G))(x - f(k_2 G))\cdots(x - f(k_n G)) + gk \mod p$$

The required computation time is $(30n\text{-}1)\ T_{MUL}$.

(2). The retrieval of the group key:

$$gk = SF(h(k_i)),\ 1 \le i \le n$$

The required computation time of each member is 2 T_H, then the required computation time for all members will be $2n\ T_H$,

$$gk = ECF(f(k_i G)),\ 1 \le i \le n\ ;$$

The required computation time of each member is $58T_{MUL}$, then, the required computation time for all members will be $58n\ T_{MUL}$.

(3). The production of polynomials $SF(x)$ and $ECF(x)$ on the entrance of new members:

$$SF'(x) = (x - h(k_1))(x - h(k_2))\cdots(x - h(k_n))(x - h(k_{n+1})) + gk'$$

The required computation time is $(n+1)\ T_H + n\ T_{MUL}$;

$$ECF'(x) = (x - f(k_1 G'))(x - f(k_2 G'))\cdots(x - f(k_n G'))(x - f(k_{n+1} G')) + gk' \mod p$$

The required computation time is $(30n+29)T_{MUL}$.

(4). The retrieval of the new group key:

$$gk' = SF'(h(k_i)),\ 1 \le i \le n+1\ ;$$

The required computation time of each member is $2T_H$, then the required computation time for all members will be $2(n+1)T_H$,

$$gk' = ECF'(f(k_i G'))\ ,\ 1 \le i \le n+1\ ;$$

The required computation time of each member is 58 T_{MUL}, then, the required computation time for all members will be 58 $(n+1)\ T_{MUL}$.

The analysis of the production of the group key and time complexity of $SF(x)$ and $ECF(x)$ are listed in table 2.

3. Comparison of the operating speeds before and after the improvements

(1). The production of the polynomial $ECF(x)$:

$$ECF(x) = (x - f(k_1 G))(x - f(k_2 G))\cdots(x - f(k_n G)) + gk \mod p$$

The time required before the improvement of the operating speed is $(30n\text{-}1)T_{MUL}$

$$ECF_1(x) = (x - f(k_1 G_1))(x - f(k_2 G_1))\cdots(x - f(k_{\lfloor n/m \rfloor} G_1)) + sgk_1 \mod p_1$$

$$ECF_2(x) = (x - f(k_{\lfloor n/m \rfloor + 1} G_2))\cdots(x - f(k_{2\lfloor n/m \rfloor} G_2)) + sgk_2 \mod p_2$$

$$\vdots \qquad\qquad \vdots$$

$$ECF_m(x) = (x - f(k_{(m-1)\lfloor n/m \rfloor + 1} G_m))\cdots(x - f(k_n G_m)) + sgk_m \mod p_m$$

$$ECF_{m+1}(x) = (x - f(sgk_1 G_{m+1}))(x - f(sgk_2 G_{m+1}))\cdots(x - f(sgk_m G_{m+1})) + gk \mod p_{m+1}$$

After improvements were made, the time required became $(30n+29m\text{-}1)T_{MUL}$, the time difference is $-(29m)T_{MUL}$.

(2). Retrieval of the group key:

$$gk = ECF(f(k_i G))$$

Before improvement, the time requirement

for each member was $58T_{MUL}$, and that for all members was $58n\ T_{MUL}$:

$$gk = ECF_{m+1}(f(sgk_i G_{m+1}))$$

After improvements were made, the time requirement for a single member became $58T_{MUL}$, and that for all members is now $58m\ T_{MUL}$, the time difference is $58(n\text{-}m)\ T_{MUL}$.

(3). The production of polynomial $ECF(x)$ on the entrance of new members:

$$ECF'(x) = (x - f(k_1 G'))(x - f(k_2 G'))\cdots (x - f(k_n G'))(x - f(k_{n+1} G')) + gk' \bmod p$$

Before improvement, the time requirement was $(30n+29)T_{MUL}$.

$$ECF_m'(x) = (x - f(k_{(m-1)\lfloor n/m \rfloor + 1} G_m'))\cdots (x - f(k_n G_m'))(x - f(k_{n+1} G_m')) + sgk_m' \bmod p_m$$

$$ECF_{m+1}'(x) = (x - f(sgk_1 G_{m+1}'))(x - f(sgk_2 G_{m+1}'))\cdots (x - f(sgk_m' G_{m+1}')) + gk' \bmod p_{m+1}$$

After improvement, the time required became:

Taking $y=(n+1)-[(m-1)\ \lfloor n/m \rfloor\ +1]+1=(n+1)-(m-1)\lfloor n/m \rfloor$,

$Y= y+m=(n+m+1)-(m-1)\ \lfloor n/m \rfloor\ [30(Y)-2]\ T_{MUL}$;

After improvement, the time required became $\{-30[(m+1)+(m-1)\ \lfloor n/m \rfloor]+31\}T_{MUL}$.

(4). The retrieval of the new group key:

$$gk' = ECF'(f(k_i G'))$$

Before improvement, the time requirement for a single member was $58T_{MUL}$, and that for all members was $58(n+1)T_{MUL}$:

$$gk' = ECF'_{m+1}(f(sgk_i G_{m+1}'))$$

After improvement, the time requirement for a single member became$58T_{MUL}$, and that for all members is now $58m\ T_{MUL}$, so the time difference is $58(n\text{-}m+1)T_{MUL}$.

We can deduce from the above analysis and examples that computing only a portion takes a lot less time than computing the whole thing. At the same time, the greater the number of subgroups, the more the computation time saved. Hence, effective improvement has indeed been made herein with regard to the production of group keys and retrieval of group keys. The results are listed in table 3.

VIII. Conclusion

Three solutions were presented with regard to the security problem in Wu's proposal. The first solution was to change the hash function from h to h', h' is the new one-way hash function, to construct a new secure filter polynomial. The second solution was to add a virtual new member to the group at the time of the construction of a new secure filter polynomial. The third was to use the elliptic curve encryption method to build the new polynomial $ECF(x)$. The difference in the above methods being, the changing to h' which will ensure the security, and that attackers will not be able to crack it. Besides this, there is the subgroup method, which saves on time and computation load to improve operating speed and increase efficiency.

References

[1] Kuen-Pin Wu, Shanq-Jang Ruan, Feipei Lai and Chih-Kuang Tseng, "On Key Distribution in Secure Multicasting," *Proceedings 25th annual IEEE Conference on Local Computer Networks*, pp. 208-212, 2000.

[2] Alferd J. Menezes, "Elliptic Curve Public Key Cryptosystem," *Anburn*, pp.19-21, 1997.

[3] Anthong-w.knapp, "Elliptic Curve," *Princeton University Prece*, pp. 67-74, 1992.

[4] Joseph H. Silverman, John Tate, "Rational Points on Elliptic Curves," *Springer-Verlag New York*, pp. 28-32, 1992.

[5] Joseph H. Silverman, "The Arithmetic of Elliptic Curves," *Springer-Verlag new work* , pp. 55-63,1986.

[6] N. Koblitz, A. Menezes and S. Vanstone, "The State of Elliptic Curve Cryptography," *Design, Codes and Cryptography*, Vol.19, pp. 173-193, 2000.

Table 1. Wu′s method and comparison of the security of the proposal

Scheme Key Generation	Wu′s scheme	Our scheme
Key Generation Phase	$SF(x) = (s - h(k_1))(x - h(k_2)) \cdots (x - h(k_n)) + gk$	$ECF(x) = (x - f(k_1 G))(x - f(k_2 G)) \cdots (x\text{-}f(k_n G)) + gk$
Group key Derivation Phase	$gk = SF(k_i), \quad 1 \le i \le n$	$gk = ECF(k_i), \quad 1 \le i \le n$
Changing Key Generation	$SF'(x) = (s - h(k_1))(x - h(k_2)) \cdots (x - h(k_n)) + gk'$	$ECF'(x) = (x - f(k_1 G'))(x - f(k_2 G')) \cdots (x\text{-}f(k_n G')) + gk'$
Derivation the New Group Key Phase	$gk' = SF'(k_i), \quad 1 \le i \le n$	$gk' = ECF'(k_i), \quad 1 \le i \le n$

Table 2. The analysis of the production of the group key and time complexity of $SF(x)$ and $ECF(x)$

Scheme / Key Generation	Wu′s scheme	Our scheme
The production of polynomials $SF(x)$ and $ECF(x)$	$n\ T_H$+(n-1) T_{MUL}	(30 n -1)T_{MUL}
Retrieving of the group key	2 $n\ T_H$	58 $n\ T_{MUL}$
Production of polynomials $SF(x)$ and $ECF(x)$ at the time of addition of new members	(n +1) T_H + $n\ T_{MUL}$	(30 n +29) T_{MUL}
The retrieving of the new group key	2(n +1) T_H	58(n +1) T_{MUL}

Table 3. Comparison of time complexity in relation to the production and retrieval of group keys before and after the improvement were made herein

Scheme / Key Generation	The time difference after the improvement of operation speed
The production of the polynomial $ECF(x)$	$-(29m)\ T_{MUL}$
The retrieving of the group key	$58(n\text{-}m)\ T_{MUL}$
The production of the polynomial $ECF(x)$ **at the time of addition of new members**	$\{-30[(m+1)+(m-1)\lfloor n/m \rfloor]+31\}\ T_{MUL}$
The retrieving of the new group key	$58(n\text{-}m+1)\ T_{MUL}$