# Performance Evaluation of Group Key Management Scheme Blocking Collusion Attack

Jong-In Chung†

## ABSTRACT

Multicast services are provided on the Internet in fast increasing. Therefore it is important to keep security for multicast communication. If a member of the group is removed, new group key has to be generated and distributed to all remaining members of group. Minimizing number of messages and operation cost for generation of the composite keys to be used to encrypting group key are important evaluating criteria of multicast key management scheme since generation and distribution of new keys for rekeying require expensive operation. Periodic batch rekeying can reduce these important parameters rather than rekeying sequentially in fashion one after another. In this paper, Hamming distance is calculated between every members to be removed. In batch rekeying the members with Hamming distance less than threshold are selected for rekeying procedure. With running the round assignment algorithm in the case of removing several members simultaneously, our scheme has advantages of reducing messages and operation cost for generation of the composite keys and eliminating possibility of collusion attack for rekeying. We evaluate performance of round assignment algorithm through simulation and show that our scheme is excellent after performance comparison of existent schemes and our scheme.

**Keywords** : Group Key Management, Collusion Attack, Multicast

# 공모공격의 차단기능을 갖는 그룹 키 관리기법의 성능평가

정종인 †

## 요 약

오늘날 인터넷상에서 멀티캐스트 서비스가 급속도로 증가하고 있으며 이에 따라 멀티캐스트 통신의 보안을 유지하는 것이 중요하다. 그룹의 멤버가 제거되면 새로운 그룹키를 생성하여 그룹의 나머지 모든 멤버들에게 전달되어야 한다. 새로운 키를 생성하여 분배하는 것은 많은 연산을 요구하므로 rekey하기 위하여 보내는 메시지의 수와 복합키를 생성하기 위한 연산비용을 최소화하는 것은 키 관리기법을 평가하는 중요한 기준이다. 주기적인 일괄제거는 멤버를 순차적으로 1개씩 제거하는 것보다 rekey에 대한 메시지의 수와 복합키의 연산 비용을 줄일 수 있다. 본 논문에서는 그룹에서 제거될 멤버들간의 해밍거리가 임계치보다 작은 멤버들만 동시에 제거된다. 여러 개의 멤버를 제거할 때 라운드 조정 알고리즘을 수행하면 rekey를 위하여 메시지의 수와 복합키를 생성하기 위한 연산의 비용을 줄이며 공모공격의 가능성이 제거되는 이점이 있다. 시뮬레이션을 통하여 라운드 조정알고리즘을 적용한 결과의 성능을 평가하며 기존의 기법과 비교를 하여 본 논문에서 제안하는 기법이 우수함을 보인다.

**키워드**: 그룹키 관리, 공모공격, 멀티캐스트

## 1. Introductions

Internet users are using a lot of parts of communication for multimedia applications including video, audio, and data. These applications are unicast service and multicast service, multicast service is increasing rapidly today. Multicast service users are managed by multicast group. To keep multicast

† 종신회원: 공주대학교 컴퓨터교육과 교수(교신저자)
논문접수: 2006년 1월 3일, 심사완료: 2006년 3월 14일
* 이 논문은 2005년 교육인적자원부의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임.
(과제번호: R05-2004-000-11027-0).

communication secure, it needs group key that group members share. Group key offers group's security and authentication function of source. Secure multicast groups are managed by key server. This key server is known as group controller. To join multicast group, clients must request group controller access to group. If controller is required, controller confirms identification by clients' login and password or certificate. If clients are permitted join group, controller offers them group address and essential keys that messages deliver.

Through member must change key that have whenever group membership changes, FS (Forward Secrecy) and BS (Backward Secrecy) are guaranteed. Thus, this change of key is called rekeying. FS means that can not get group information that is attained after if some member leaves group. Also BS means that can not get group information that is attained before when a new member joins group [1].

Member's join and leaving are closely related to scalability problem of multicast key management scheme. Let's assume multicast group that is consisted of N members and share 1 group key. It should change an existent group key to guarantee BS if members join group, and deliver to group members of all remaining N-1 a new group key to guarantee FS if members leave. But, it is not simple problem to send a new group key securely to all remaining members.

To offer scalability, many studies have used logical binary tree of KEK[2-4]. Method that use binary key tree of KEK is very efficient because number of messages needed to change a new group key are proportional in depth of tree ($\log_2 N$). In rekeying of LKH(Logical Key Hierarchy) scheme, complexity of messages to be delivered can reduce in $O(\log_2 N)$ but there is possibility of group members' collusion attack. Wong proposed for rekeying a typical LKH multicast scheme that is encrypted by his children keys in some node of key management tree and controller manages 1 group key, N-2 auxiliary keys, and N individual keys in case group size is N. Chang proposed a LKH key management scheme that each node of key tree is allocated by binary ID and management is easy. If group size is N, controller must manage 1 group key and 2logN auxiliary keys. Chang suggested a scheme that generates new keys when a number of leaving members reach in any level or in periodic time but there is collusion attack's possibility that destroy key's confidential.

Whenever group members leave from group dynamically, it requires a lot of operations to change and delivers a new group key[5-7]. When there are member leaving requests in most applications, controller gathers the requests and removes leaving member at the same time periodically without rekeying immediately [8-10]. For example, in web TV, leavings and joins group are achieved at specified short time of program. In this case, controller must change group key together in periodic time because it is inefficient if group key is changed whenever joins and leavings are requested.

When we need rekeying scheme that group controller has less keys that it should store, controller does not send changed keys but sends necessary information to change keys[11-14]. It can reduce keys that controller keeps by making keys of node from pseudo random function that only controller knows. Also, it is impossible for nongroup member to produce node keys because a multiplication result of random numbers created from pseudo random function is sent as message. This is used to avoid collusion attack of group members. But this scheme is difficult to create random numbers in server actually.

The most important factor that should consider when run rekeying is number of keys that controller and member should store, number of messages that deliver whenever to rekey, blocking possibility of collusion attack, the operation cost of composite key to encrypt group key and so on.

In this paper, we use Chang's key management scheme, but propose a scheme that has less operation cost of composite keys that use to encrypt group key, fewer messages than Chang scheme by selecting members being removed and reassigning round them, and solves collusion attack problem. It is shown that our scheme is more excellent than existent schemes by performance evaluation through simulation

## 2. Key Management Scheme

User ID (UID) is given n bit string to each member of group. UID is expressed by $x_{n-1}x_{n-2}\cdots x_0$ and each $x_i$ is 0 or 1. The length of UID depends on group size. For example, 9 bits UID is used in case that group size is 300.

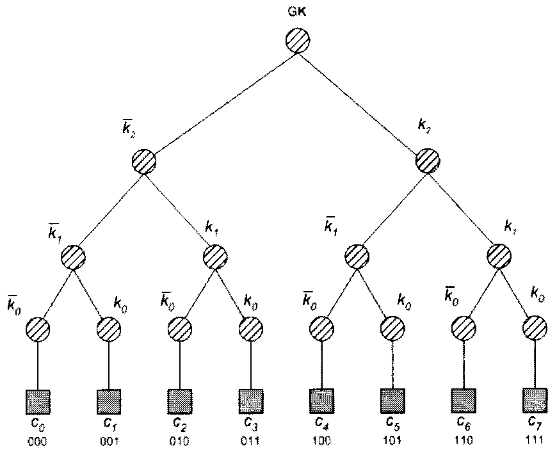
If a member, UID $x_{n-1}x_{n-2}\cdots x_0$ registers controller to join group, controller sends the member a group key GK. GK is shared by all group members, and used to encrypt or decrypt the messages when send ones with multicast group key. Also, a member receives n auxiliary keys, $K_{n-1}, K_{n-2}, \cdots, K_0$ set, where $K_i$ is expressed by $k_i$ if $x_i$ is 1 and by $\overline{k}_i$ else. Auxiliary keys are used to update group key by secure method. $k_i$ and $\overline{k}_i$ are complement key each other, and have meaning that are not numerical complementary relation and are not related mutually. Controller manages all auxiliary keys $\{k_0, \overline{k}_0, k_1, \overline{k}_1, \cdots, k_{n-1}, \overline{k}_{n-1}\}$.

<Fig. 1> shows keys that each member keeps in case that group size is 8. Square terminal nodes in tree stand for group members that have 3 bits UIDs, and circle nodes stand for keys. Each member keeps keys in path from a terminal node to root node. For example, member $c_5$(UID 101) keeps group key GK and auxiliary keys $k_2$, $\overline{k}_1, k_0$.

Group key and auxiliary keys are usually changed when group member leaves. Leaved member has to not be able to obtain group keys to be used future. To do so, controller changes group key and auxiliary keys that other members except the leaving members have if they leave. The leaving members can not obtain future keys using their auxiliary keys and group key. Such key changing is called rekeying. Expensive cost is needed for rekeying if controller changes group key and auxiliary keys whenever a group member leaves. Also, when 2 members leave contiguously in short time, controller must rekey for the second member without applying new group key and auxiliary keys generated by rekeying for the first member. This causes resource waste. Problem that synchronization between keys and data does not agree is called out-of-synchronization. Out-of-synchronization can be occurred if rekeying accomplishes whenever there is member's leaving [7]. When there is member join or leaving request in most applications, controller need not to handle immediately. For example, controller may not remove member immediately as soon as service time is over for group key management in VOD and on-line game.

Group controller gathers leaving members in specified period time and rekeys at once, which is known as batch rekey. We use batch rekey mode in this paper. When members to remove gather and remove periodically, the periodic time is called round. Controller can decide the periodic duration time of round random.

Because rekeying occurs every period noncontiguously, group key and auxiliary keys are expressed by $GK(r)$ and $k_i(r)$, $\overline{k}_i(r)$ respectively, where $r$ is current round.



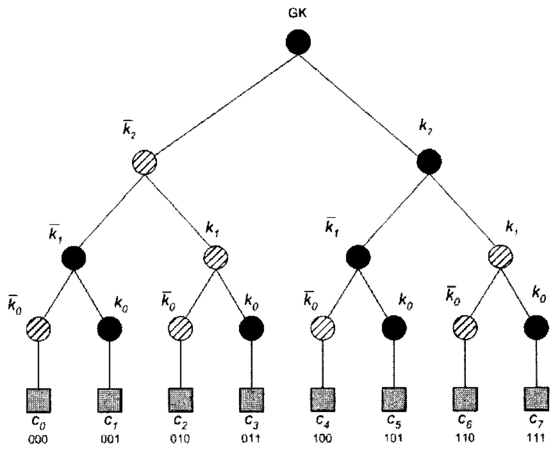
<Fig. 1> Key distribution tree of group size 8

## 2.1 Removal of One Member

New group key should be distributed all members except the member that leaves group whenever it leaves. To update current group key GK(r), controller calculates new group key GK(r+1). GK(r+1) is encrypted using complementary key of the leaving member. For example, the leaving member has auxiliary key $k_2$, $\overline{k}_1$, $k_0$ if its UID is 101. Therefore, GK(r+1) is distributed to all members of group after making 3 messages $\{GK(r+1)\}_{\overline{k}_0}$, $\{GK(r+1)\}_{k_1}$, $\{GK(r+1)\}_{\overline{k}_2}$ that is encrypted individually by complementary key $\overline{k}_2, k_1, \overline{k}_0$ of the leaving member. Where, $\{L\}_M$ is meaning that encrypts string $L$ by key $M$ and sends to whole group members.

The member to be removed receives all messages encrypted, but it can not decrypt the messages because they were encrypted by keys that the leaving member does not have. But, group remaining members can decrypt more messages than at least 1 because their UIDs differ with leaving member's UID more bits than at least 1.

<Fig. 2> represents group rekeying scheme when member $c_5$ is removed. Dark circle nodes represent auxiliary keys that the leaving member $c_5$ has. Diagonal line circle nodes represent auxiliary keys that $c_5$ does not have, the complementary key of $c_5$. All nodes on path from $c_5$ to root are dark circle nodes, but all nodes on path from others to root exist more diagonal line circle nodes than at least 1. Therefore, all members except $c_5$ can decrypt messages encrypted by auxiliary keys that $c_5$ does not have.



<Fig. 2> Removal example of member $c_5$

Analyzing key distribution algorithm, keys managed by controller are $2\log N+1$, keys to update group key are $\log N$ after 1 member are removed when group size is N. Removed member can not obtain new group key. In case of updating group key in next round, auxiliary keys must be updated so that removed member does not decrypt group key. Each member uses hash function f of expression (1) to update auxiliary key $K_i(r)$.

$$K_i(r+1) = f(K_i(r), GK(r+1)) \quad (1)$$

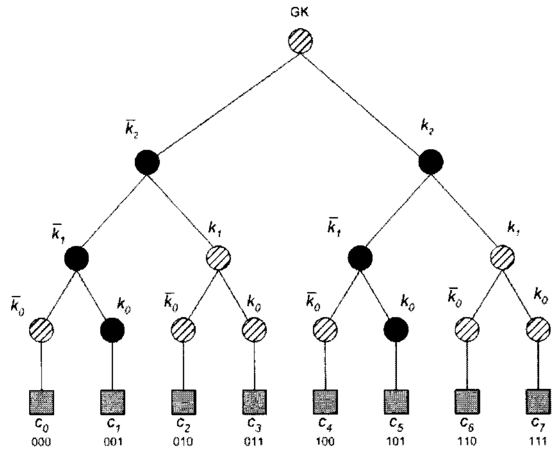
Only members that have new group key GK(r+1) can update auxiliary key $K_i(r+1)$. Removed member can not update new auxiliary key because it does not have GK(r+1), and security is kept because it can not update group key to be changed in next round, GK(r+2).

## 2.2 Removal of Several Members

Group key management can repeat k times key update process in section 2.1 to remove k members. But more effective method is removing at once after gathering members to remove. Usually member set is $S=\{c_0, c_1, \cdots, c_{N-1}\}$ and $N=2^n$. Group membership is decided by *mem*( ), Boolean function of UID in any time point. That is, $x_{n-1}x_{n-2}\cdots x_0$ is group member if $mem(x_{n-1}x_{n-2}\cdots x_0)=1$, otherwise is removed in group.

Group rekeying is meaning that updates group key and auxiliary keys of all members except members, $mem(\mathrm{UID})=0$. Group key is encoded by keys that members to be removed do not have and distributed to all members. Considering scalability and efficiency, it is desirable that rekeying has minimum operation to encrypt and has minimum messages to be distributed to group.

For example, let's suppose that UID 001 ($c_1$) and 101 ($c_5$) are removed from group in <Fig. 3>. Controller must distribute new group key, GK(r+1) to set $S=\{c_0, c_2, c_3, c_4, c_6, c_7\}$ of group remaining members. Usually member that group key must be distributed actually is subset of S, because new group key is not distributed to all UIDs. Unassigned UIDs can handle on "don't care" condition. "Don't care" condition does not influence in key distribution. 2 messages with encrypted group key, $\{GK(r+1)\}_{\bar{k}_0}$ and $\{GK(r+1)\}_{k_1}$ are distributed. The first message can be decrypted by members $(c_0, c_2, c_4, c_6)$ and the second message can be decrypted by members $(c_2, c_3, c_6, c_7)$. So group rekeying is possible because 2 messages cover all members except $c_1, c_5$ in group S. Dark circle nodes in <Fig. 3> represent keys that $c_1, c_5$ have and can not use to encrypt GK(r+1).



<Fig. 3> Removal example of 2 members $c_1, c_5$

Rekeying procedure to remove a member needs 3 messages. Controller must distribute a total of 2×3 =6 messages to remove 2 members from multicast group if rekeying is performed sequentially as key update procedure described in the previous section. But only 2 messages are distributed in case of batch rekeying. This numerical value is less numerical value than case of removal of a member. If more or 1 bit that UIDs of group remaining members are common (differing with removed UID's bit) exist, we can know intuitively that messages to be distributed are decreased. $x_0$ of members $(c_0, c_2, c_4, c_6)$ is all 0 unlike $x_0$ of members $(c_1, c_5)$ to be removed in the above example.

$\{GK(r+1)\}_{\bar{k}_0}$ can be decrypted because members $(c_0, c_2, c_4, c_6)$ has $\bar{k}_0$ while members to be removed have $k_0$. Similarly $x_1$ of members $(c_2, c_3, c_6, c_7)$ is all 1 unlike $x_1$ of UIDs of members( $c_1, c_5$) to be removed. $\{GK(r+1)\}_{k_1}$ can be decrypted $\{GK(r+1)\}_{k_1}$ because members $(c_2, c_3, c_6, c_7)$ has $k_1$ while members to be removed have $\bar{k}_1$. As there are more common bits of UIDs of members to removed at the same time, messages for rekeying decrease because there are been more common bits of group remaining members.

Batch rekeying problem is that like systematically finding the group of members that UID bits of remaining members except members to removed are common. This problem is such as minimization[15] of Boolean function. For example, membership function can be expressed as following.

$$mem(x_2, x_1, x_0) = \bar{x}_2 \bar{x}_1 \bar{x}_0 + \bar{x}_2 x_1 \bar{x}_0 + \bar{x}_2 x_1 x_0 + x_2 \bar{x}_1 \bar{x}_0 + x_2 x_1 \bar{x}_0 + x_2 x_1 x_0 \quad (2)$$

+ is logical OR and multiplication of variables is logical AND. Each term of expression (2) corresponds to a message that encrypt group key to be delivered to each member. Letters of each term are used to generate a key by one-way function as input parameters(auxiliary keys). The generated key is known composite key. Group rekeying can be achieved by generating a message encrypted by the composite key.

For example, term $x_2 \bar{x}_1 \bar{x}_0$ corresponds to a message that is encrypted by composite key that is generated from $k_2, \bar{k}_1, \bar{k}_0$. Therefore, 6 messages must be distributed to deliver new group key to all members of group. Such each message is decrypted by 1 of 6 members.

We use example to understand how minimization method of Boolean function is applied in key update problem that have minimum messages and minimum keys. Let's suppose that $c_7$ already left group, and that UIDs were allocated to 7 members except $c_7$. If $c_1$ and $c_5$ are removed from group, membership function is like <table 1>. Output of $c_1$ and $c_5$ is 0, and $c_7$ that already left group is X (don't care), and output of remaining members is 1. $c_7$ can not decrypt messages that are encrypted by new key because auxiliary keys are updated since $c_7$ has left group. We can express as expression (3) if membership function is displayed by sum of minterms. $\Sigma$ of expression (3) stands for OR of terms, lower case m and digits stand for minterms of Boolean function. For example, minterm $\bar{x}_2 x_1 \bar{x}_0$ is expressed as m(2), $x_2 x_1 \bar{x}_0$ as m(6). Also, $d(7)$ stands for don't care term, UID 7.

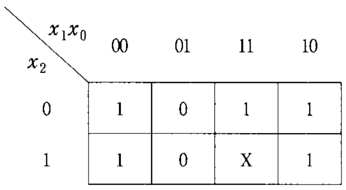
$$mem(x_2, x_1, x_0) = \Sigma m(0, 2, 3, 4, 6) + d(7) \quad (3)$$

〈Table 1〉 membership function

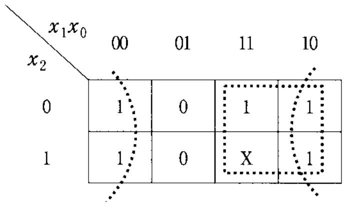
| Input $(x_2 x_1 x_0)$ | Output |
|---|---|
| 000 | 1 |
| 001 | 0 |
| 010 | 1 |
| 011 | 1 |
| 100 | 1 |
| 101 | 0 |
| 110 | 1 |
| 111 | X |

<Fig. 4> (a) shows Karnaugh map of <table 1>. A rectangular of Karnaugh map corresponds to a minterm. Minimization process

of Boolean function makes block of minterms as big as possible grouping minterms of 1 or X as shown as <Fig. 4>(b). The bigger block size is, the fewer input variables in term are. Term that number of input variable are least is called prime term. Minimization result of <table 1> is $x_1 + \bar{x}_0$.



(a) Karnaugh map



(b) Selection of prime term

<Fig. 4> Karnaugh map and minimization of membership function

We can analyze relation between minimization and batch rekeying of Boolean function as following from <Fig. 4>.

- A minterm 0 is a member that is removed in group.
- A minterm 1 is a member that remains to group.
- A minterm X is a UID that is not allocated up to now.

In the batch rekeying, group key update is closely related to problem that finds prime terms in Karnaugh map. Prime term with 1 input variable is called simple key because it is consisted of 1 key. Each prime term corresponds to a message that has a group key because it corresponds to a simple key that uses to encrypt group key. To update group key in the above example, 2 messages $\{GK(r+1)\}_{k_1}$ and $\{GK(r+1)\}_{\bar{k}_0}$ that encrypted group key are distributed to group. The messages use only 2 simple keys and operation cost to calculate composite key is low. As there are many common bits of UIDs to be removed at the same time, number and size of messages for rekeying decrease.

In the case of batch rekeying in Chang scheme, It has properties that messages and auxiliary keys that consisted of composite key can be changed according to UIDs to be removed, and also leaving members can collude and then obtain new group key. Collusion attack is that obtains new key illegally if leaving members collude and combine their keys. For example, collusion attack is possible because two members are keeping all auxiliary keys of group if UID 0 and 7 in group size 8 leave at same time. There is shortcoming that does not maintain advantage that use binary UID to key management scheme because Chang scheme can not block collusion attack.

## 2.3 Round Assignment Algorithm

We can reduce messages for rekeying because can reduce composite keys used to encrypting group key, and solving collusion attack problem occurring in Chang scheme by assigning the round of members to be removed at the same time. Round assignment is achieved by checking UIDs of leaving members. Also the operation cost of composite key is decreased because auxiliary keys used to calculate composite key are decreased.

Hamming distance[16] is number of different bits between 2 binary bit strings when 2 UIDs are expressed by binary bit string respectively. To select removing member in step 3, threshold that controller sets random is applied. We know that it can reduce messages for rekeying and operation cost of composite keys used to encrypt new group key because it can make bigger blocks in Karnaugh map of Boolean function if Hamming distance of removing member is small. This is because there are many common bits of group remaining members' UIDs as there are many common bits of UIDs that is removed at the same time. Therefore, messages and operation cost of composite key for rekeying decrease.

**Step 1** : Store in queue the UIDs of members to be removed. Calculate Hamming distance between two UIDs for all members.

**Step 2** : Check possibility of collusion for members to be removed. If there is possibility of collusion, except from removing targets the member stored in queue finally among members with the biggest Hamming distance between members that have possibility of collusion.

**Step 3**: Select as removing targets only UIDs that Hamming distance between removing targets of step 2 is below than a given threshold.

**Step 4**: Generate keys to encrypt new group key distributing to all members except selected ones.

**Step 5** : Delete selected UIDs from queue. Go to step 1.

Round Assignment Algorithm

If threshold is small, operation cost of composite key is less and number of messages for rekeying decrease, but can not remove members immediately. Controller must decide threshold according to tradeoff between efficiency for operation cost and number of messages and immediacy of member removal.

controller must set threshold low to improve efficiency for number of messages and threshold high in case of playing an important role immediacy. Application that transmit realtime data such as video conferencing, video server, stocks information, distributed game can set threshold high and applications that transmit non realtime data such as software delivery, database copy can set threshold low.

If Hamming distance is n in case that UID is expressed by n bits, 2 members have collusion possibility. Whether collusion possibility of members exist or not can be checked by following process. Collusion possibility is usually checked as expression (4) and (5) if UIDs of members to be removed are $a_{n-1}a_{n-2}\cdots a_0$, $b_{n-1}b_{n-2}\cdots b_0$, $c_{n-1}c_{n-2}\cdots c_0$, $\cdots$.

$$z_i=(a_i+b_i+c_i+\cdots)(a_ib_ic_i\cdots)',\ i=n-1,n-2\cdots 0 \tag{4}$$

$$P=z_{n-1}z_{n-2}\cdots z_1z_0 \tag{5}$$

For any $i$ in expression (4), $z_i$ is 1 if each bit $i$ of UIDs to be removed includes 0 and 1. $z_{n-1}z_{n-2}\cdots z_1z_0$ in expression (5) stands for logical AND from $z_{n-1}$ to $z_0$. If P has 1, collusion possibility exists and it does not exist otherwise. For example, for 3 UIDs, $a_3a_2a_1a_0=1100$, $b_3b_2b_1b_0=1010$, and $c_3c_2c_1c_0=0101$, it is $z_3=1$ by expression (4) because the first bits ($a_3,b_3,c_3$) of UIDs have 0 and 1. In the same way, it is $z_2=z_1=z_0=1$ by expression (4) because the second bits ($a_2,b_2,c_2$) of UIDs, the third bits ($a_1,b_1,c_1$) and the fourth bits ($a_0,b_0,c_0$) have 0 and 1. 3 members have collusion possibility because it is P = 1 by expression (5).

When group size is 16, let's give example to explain round assignment algorithm. We wish to remove UID 1111( $c_{15}$), 0011( $c_3$), 0000( $c_0$) from group in the first round. Insert these sequentially in queue. Hamming distance between members in queue is <table 2>.

<Table 2> Hamming distance between members

| | 1111( $c_{15}$ ) | 0011( $c_3$ ) | 0000( $c_0$ ) |
|---|---|---|---|
| 1111( $c_{15}$ ) | 0 | 2 | 4 |
| 0011( $c_3$ ) | 2 | 0 | 2 |
| 0000( $c_0$ ) | 4 | 2 | 0 |

There is collusion possibility because $c_{15}$ and $c_0$ have Hamming distance 4. $c_0$ is excepted from removing targets because $c_0$ out of 2 members have inserted in queue finally. If controller sets threshold by 3, $c_{15}$ and $c_3$ are removed together from group in the first round because Hamming distance between them are smaller than threshold. Membership function is same as expression (6).

$$mem(x_3, x_2, x_1, x_0) = \Sigma m(0,1,2,4,5,6,7,8,9,10,11,12,13,14) \tag{6}$$

Karnaugh map for membership function is same as <Fig. 5> (a). Controller distributes 4 messages $\{GK(r+1)\}_{\overline{k}_0}$, $\{GK(r+1)\}_{\overline{k}_1}$, $\{GK(r+1)\}_{f(\overline{k}_3, k_2)}$, $\{GK(r+1)\}_{f(k_3, \overline{k}_2)}$ to update group key because minimizing Karnaugh map results $\overline{x}_0 + \overline{x}_1 + \overline{x}_3 x_2 + x_3 \overline{x}_2$. Where, $f(\overline{k}_3, k_2)$ is composite key that is generated by one-way function that has $\overline{k}_3$ and $k_2$ as input parameter. 2 simple keys and 2 composite keys can be used to encrypt group key. $\{GK(r+1)\}_{\overline{k}_0}$ can be decrypted by $\{c_0, c_2, c_4, c_6, c_8, c_{10}, c_{12}, c_{14}\}$, $\{GK(r+1)\}_{\overline{k}_1}$ by $\{c_0, c_1, c_4, c_5, c_8, c_9, c_{12}, c_{13}\}$, $\{GK(r+1)\}_{f(\overline{k}_3, k_2)}$ by $\{c_4, c_5, c_6, c_7\}$, and $\{GK(r+1)\}_{f(k_3, \overline{k}_2)}$ by $\{c_8, c_9, c_{10}, c_{11}\}$. Let's suppose that $c_2, c_{10}$ were newly inserted in queue to be removed from group before the second round begins. $c_0, c_2, c_{10}$ were stored sequentially in current queue. Hamming distance between members in queue is <table 3>.

<Table 3> Hamming distance between members

| | 0000( $c_0$ ) | 0010( $c_2$ ) | 1010( $c_{10}$ ) |
|---|---|---|---|
| 0000( $c_0$ ) | 0 | 1 | 2 |
| 0010( $c_2$ ) | 1 | 0 | 1 |
| 1010( $c_{10}$ ) | 2 | 1 | 0 |

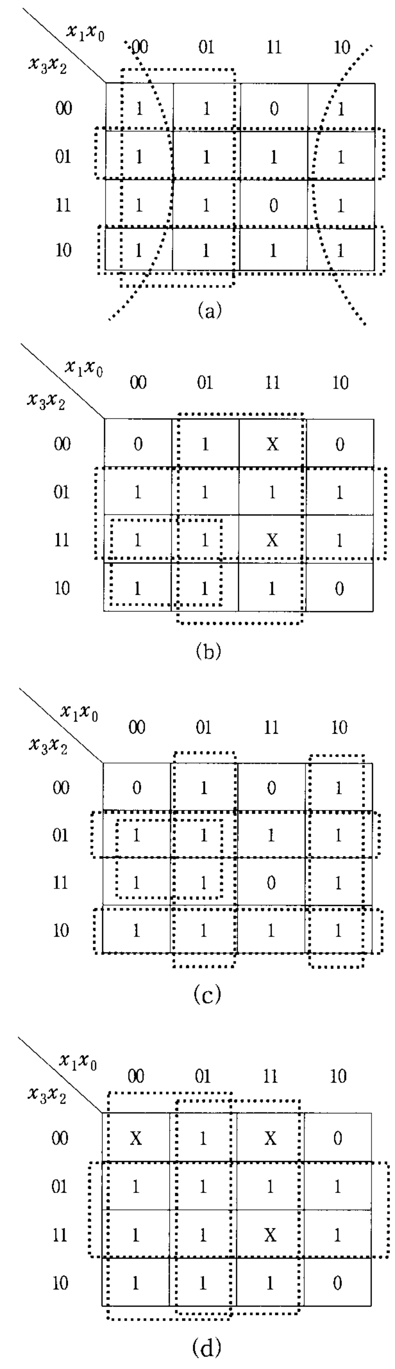
In the second round, all members in queue are selected as removing target because it is no collusion possibility between members in queue and Hamming distance between members is smaller than given threshold. Membership function is expression (7).

$$mem(x_3, x_2, x_1, x_0) = \Sigma m(1,4,5,6,7,8,9,11,12,13,14) + \Sigma d(3, 15) \tag{7}$$

Karnaugh map for expression (7) is <Fig. 5> (b). Controller distributes 3 messages $\{GK(r+1)\}_{k_0}$, $\{GK(r+1)\}_{k_2}$, $\{GK(r+1)\}_{f(k_3, \overline{k}_1)}$ to update group key because minimizing Karnaugh map results $x_0 + x_2 + x_3 \overline{x}_1$. 2 simple keys and 1 composite key can be used to encrypt group key. Chang scheme requires 5 messages encrypted 5 composite keys respectively because minimization result in the first round is $x_3 \overline{x}_2 + x_2 \overline{x}_1 + x_1 \overline{x}_0 + x_0 \overline{x}_1 + x_2 \overline{x}_3$ as <Fig. 5>(c). Also, the most important problem is that $c_0$ and $c_{15}$ can collude together.

3 messages are required because Karnaugh map minimization in the second round is $\overline{x}_0 + x_1 + x_2$ as <Fig. 5> (d). <Table 4> shows comparison of number of auxiliary keys needed for composite keys and number of messages

distributed for rekeying between Chang scheme and CAB(Collusion Attack Block) scheme proposed in this paper.



<Fig. 5> Minimization of membership function

CAB scheme must distribute total 7 messages to update group key in the first and the second round. 4 messages of them are encrypted by simple key, 3 ones are encrypted using composite key generated by 2 auxiliary keys. Chang scheme must distribute total 8 messages in the first and the second round. 3 messages of them are encrypted by simple key, 5 ones are encrypted using composite key generated by 2 auxiliary keys. Number of prime terms in Boolean function represent messages, number of letters that consist of prime term represents auxiliary keys used to encrypt group key. It can reduce operation cost to generate composite key if number of letters that consist of prime term is few. CAB scheme can reduce messages and operation cost to generate composite keys than Chang scheme.

<Table 4> operation cost of composite key and number of messages between CAB and Chang scheme

| | number of messages | | | operation cost of composite key | |
|---|---|---|---|---|---|
| | 1st round | 2nd round | total | number of messages encrypted simple key | number of messages encrypted 2 auxiliary keys |
| CAB | 4 | 3 | 7 | 4 | 3 |
| Chang | 5 | 3 | 8 | 3 | 5 |

Number of blocks(prime terms) or number of letters in prime term decrease because can bind members remaining to group in bigger block as Hamming distance between members to be removed is small. This is same meaning that messages that encrypt group key and operation cost of composite key are decreased.

## 3. Performance and Analysis

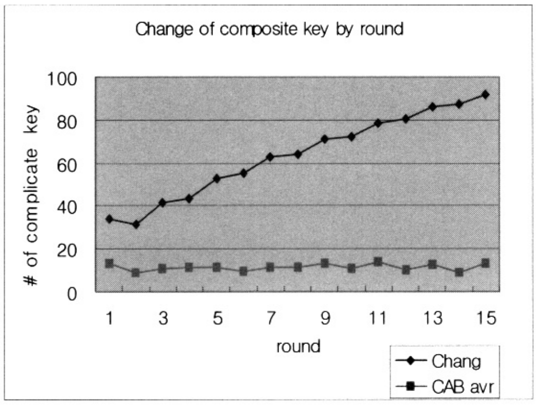
We evaluate performance for 3 schemes, Wong scheme, Chang scheme that had existent good scalability, and CAB scheme. And we show performance of CAB scheme is excellent.

In case of N = 1024, we compare messages and operation cost of composite key for 3 schemes through simulation results, do number of messages and operation cost of composite key according to threshold in only CAB scheme.

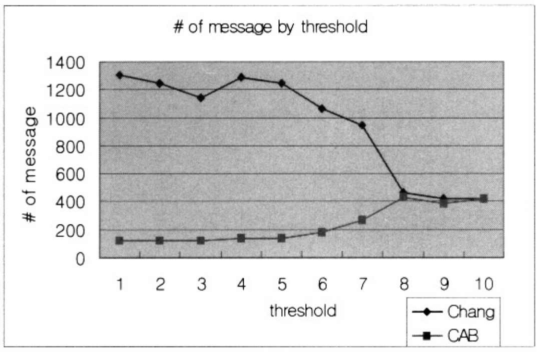
Wong scheme delivers unique private key to member using secure channel and delivers auxiliary keys and group key that exist a path from terminal node to root node in key tree if a member joins. Secure channel can use Diffie-Hellman [17] key exchange method. Group controller should manage a lot of key because Wong scheme must deliver private keys unlike CAB and Chang scheme when group size is big.

<Fig. 6> and <Fig. 7> represent change of simple key and composite key by round in Chang scheme and CAB scheme. Simple key and composite key are always generated at the same time. Graph change of simple key and composite key applied average of thresholds in CAB scheme. With analyzing change of generated keys, Chang scheme is little changes of simple key and most composite keys are generated. Generated composite keys are shown increasing trend as round is repeated in Chang scheme but is generated constant number being less far than Chang scheme in CAB scheme.

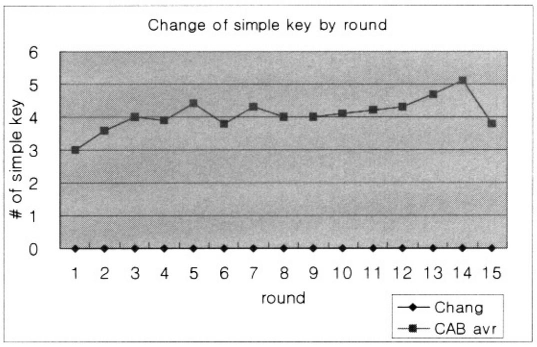
<Fig. 8> is shown comparison of messages in side of operation cost of composite key for CAB and Chang scheme. According to the result, we can know that the operation cost of message in CAB scheme has a lot of generations of simple key and a few of generation of composite key. Also, we can know that messages generated are few as threshold is low and messages are many in CAB scheme. Therefore, when compare number of messages with operation cost, we can know that CAB scheme is superior to Chang scheme. In N = 1024, we can conclude from <Fig. 8> that threshold 5 is the best in rekeying efficiency and immediacy side.



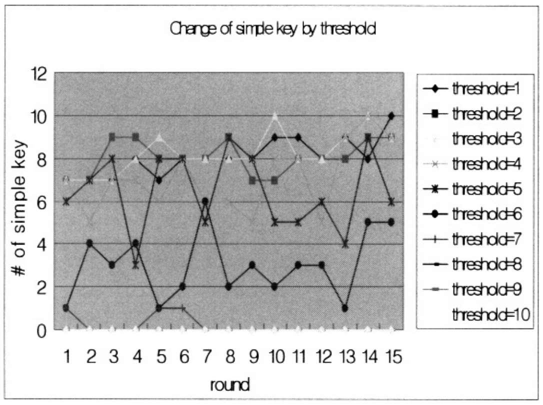
<Fig. 6> Change of composite keys by round



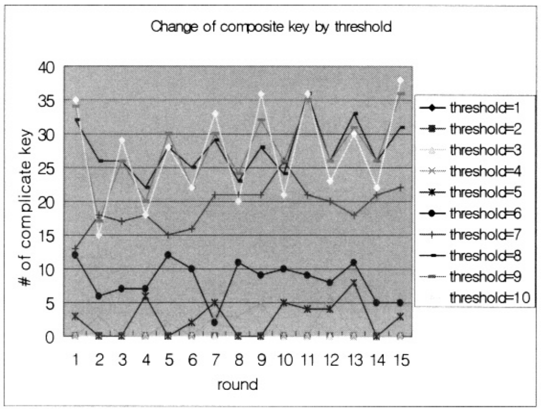
<Fig. 7> Change of message by threshold in CAB and Chang scheme



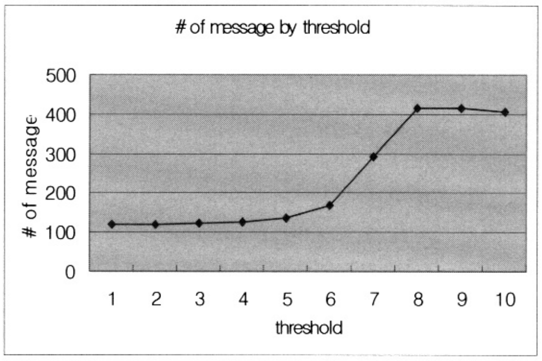
<Fig. 8> Change of simple keys by round



<Fig. 9> Change of simple key by round for any threshold in CAB scheme



<Fig. 10> Change of composite key by round for any threshold in CAB scheme



<Fig. 11> Change of message by threshold in CAB scheme

<Fig. 9> and <Fig. 10> show measurement of change of simple keys and composite keys that is generated every round according to various threshold in CAB scheme. With comparing simple keys and composite keys as round going for any threshold, few irregular state output because we input any threshold with repeating random join and leaving but we can know results that most simple keys are generated as threshold is low and most composite keys are generated as threshold is high. We can know from <Fig. 11> that number of message are few as threshold is low and there are a lot of distributing messages as threshold is high. Therefore, CAB scheme can reduce operation cost of composite key used to encrypt a new group and number of messages for rekeying because can make bigger block in Karnaugh map as threshold is low.

## 4. Conclusions

Controller sends members a group key to rekey. Group key is shared by group members and used to encrypt or decrypt the message when send ones with encrypted group key. Also, a member receives one set among auxiliary key sets. Auxiliary key is used to update group key on secure method.

When rekeying is achieved, messages or auxiliary keys that encrypt message are required much as Hamming distance between members to be removed is far. CAB scheme can have less than worst messages of Chang scheme because remove only member that Hamming distance between UIDs of members is below than a given threshold by round assignment algorithm. And CAB scheme needs less operation cost of composite key than Chang scheme because number of auxiliary keys to encrypt messages are few. Also, our scheme can solve collusion attack problem that is shortcoming of Chang scheme.

We evaluated performance for 3 schemes, Wong scheme, Chang scheme that have had existent good scalability, and CAB. And we showed performance of CAB scheme is excellent. And we compared number of messages and operation cost of composite key for 3 schemes through simulation results.

Comparing with number of messages between three schemes, when group's size is big, Wong scheme has a lot of number of keys that group controller should manage because Wong scheme must distribute individual keys unlike CAB and Chang scheme.

Also comparing with CAB and Chang scheme, we can know that CAB scheme has few messages to be distributed than Chang scheme as threshold is low because which is achieved by round assignment algorithm. Comparing about number of keys, we can know that CAB scheme has much generation of simple keys as threshold is low and Chang scheme has much generation of composite keys. We can also know that there are a few number of messages as threshold is low and there are a lot of messages as threshold is high because CAB scheme removes members who are below given threshold by round arrangement algorithm. There are a lot of simple keys and composite keys are few as threshold is low. On the other hand, we can know that there are a few simple keys and there are a lot of composite keys as threshold is high.

It is important criteria that it evaluates group key management scheme to reduce messages that distribute to rekey and operation cost to generate composite key. Scheme that propose in this paper reduces messages and operation cost of composite key for rekeying because runs round assignment algorithm when several members are removed and there is also advantage that can solve collusion attack problem.

## References

[1] R. Canetti, J. Garay, G.Itkis, D.Micciancio, M.Naor, B. Pinkas(1999). Multicast Security: A Taxonomy and Some Efficient Constructions. IEEE INFOCOM 99.

[2] H. Harney, E. Harder(1999). Logical Key Hierarchy Protocol(LKH). I-draft Harney-sparta-lkhp-sec-00.

[3] C. K. Wong, M. Gouda, S. S. Lam(1998). Secure Group Communications using Key Graphs. Proceedings of ACM SIGCOMM, Vancouver, British Columbia.

[4] I. Chang, R. Engel, D. Kandlur, D. Pendarakis, D. Saha(1999). Key Management for Secure Internet Multicast using Boolean Function Minimization Techniques. Proceedings of Infocom, New York.

[5] F. Zhu, A. Chan, G. Noubir(2003). Optimal Tree Structure for Key Management of Simultaneous Join/Leave in Secure Multicast. MILCOM 2003.

[6] G. Noubir(1999). A Scalable Key-Distribution Scheme for Dynamic Multicast Groups. The 6th ACM Conference on Computer and Communication Security.

[7] G. Noubir, F. Zhu, H.Chan(2002). Key Management for Simultaneous Join/Leave in Secure Multicast. IEEE Int'l Symposium on Information Theory.

[8] S. Banerjee, B. Bhattacharjee(2001). Scalable secure Group Communication over IP Multicast. Proceedings of ICNP.

[9] M. Steiner, G. Tsudik, M. Waidne(2000). Key Agreement in Dynamic Peer Groups. IEEE Tr. on Parallel and Distributed Systems, Vol 11, No 8, pp.769-780.

[10] X.S.Li, Y.R.Yang, M.G.Gouda, S.S.Lam(2001). Batch Rekeying for Secure Group Communications. WWW10.

[11] J. Pegueroles, W. Bin, M. Soriano, F.

Rico-Novella(2004), "Group Rekeying Algorithm Using Pseudo-Random Functions and Modular Reduction," LNCS 3032, p. 875-882.

[12] J. Pegueroles, F. Rico-Novella, L. Hernandez-Serrano, M. Soriano(2003), "Improved LKH for Batch Rekeying in Multicast Groups, " IEEE ITRE 2003.

[13] J. Pegueroles, F. Rico-Novella(2003), "Balanced Batch LKH New Proposal, Implementation and Performance Evaluation," ISCC 2003.

[14] J. Pegueroles, J. Hernandez-Serrano, F. Rico-Novella, M. Soriano(2003), "Adapting GDOI for Balanced Batch-LKH," draft-irtf-gsec-gdoi-batch-lkh-00.txt, June.

[15] J. F. Wakerly(1990). Digital Design Principles and Practices. Prentice-Hall.

[16] http://www.nist.gov/dads/HTML/hammingdist.html.

[17] W. Diffie and M. Hellman(1976). New Directions in Cryptography. IEEE Tr. on Information Theory, IT-22, pp.644- 654.

## 정 종 인

1981 경북대학교 전자공학과 (전산전공)(공학사)
1985 경북대학교 대학원 전자공학과 (공학석사)
1995 서강대학교 전자계산학과(공학박사)
1985~1997 우송공업대학 전산과 교수
1997~현재 공주대학교 컴퓨터교육과 교수
1999~2000 미국 USC post-doc
관심분야: 정보보안, 병렬처리구조, 네트워크
E-Mail: jichung@kongju.ac.kr