저비용 rekey를 갖는 그룹키 관리

정종인†

요 약

오늘날의 인터넷은 그룹 통신 모델인 멀티캐스트 서비스를 제공하고 있으며 멀티캐스트 통신의 보안을 유지하는 것이 중요하다. 멤버의 탈퇴는 그룹키 관리의 확장성 문제와 관련이 있다. 그룹의 한 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 위하여 연산비용을 최소화하는 것은 키 관리기법을 평가하는 중요한 기준이다. 주기적인 rekey는 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 제거한다. 여러 개의 멤버를 제거할 때 라운드 조정 알고리즘을 수행하면 rekey하기 위한 메시지의 수와 복합키를 생성하기 위한 비용을 줄일 수 있는 이점이 있다.

Group Key Management with Low Cost Rekey

Jong-In Chung†

ABSTRACT

The Internet today provides group communication model, multicast service. It is important to keep security for multicast communication. Member leaving is associated with scalability problem for group key management. If one member of the group is removed, new group key has to be changed and communicated to all remaining members of group. Modification and distribution of new group keys for rekeying is an expensive operation. Minimizing the number of messages and operation cost for generation of the composite keys are important evaluating criteria of multicast key management scheme. Periodic rekey helps reducing these important parameters rather than removing members sequentially in fashion one after another. In this paper, Hamming distance is calculated between every members to be removed. The members with Hamming distance less than threshold are selected for rekeying procedure. With running the round assignment algorithm, our model has advantages of reducing the number of message and operation cost for generation of the composite keys for rekeying.

1. 서 론

대부분의 네트워크의 응용은 유니캐스트 서비스를 사용하지만 오늘날의 인터넷은 그룹 통신 모델인 멀티캐스트 서비스를 제공하고 있다. 멀티캐스트 서비스는 1개 이상의 발신자로부터 인증된 많은 수신자들에게 메시지를 보낸다. 수신자는 IGMP 메시지를 로컬 라우터에게 보냄으로서 D급 클래스인 멀티캐스트 그룹에 가입하고 또한 탈퇴할 수 있다[1-3].

인터넷에서 멀티캐스트는 큰 그룹에 효율적이고 최선의 노력으로 전달서비스를 제공하고 있다. 그러므로 멀티캐스트 통신이 보안을 유지하

† 종신회원: 공주대학교 컴퓨터교육과 교수
논문접수: 2003년 12월 12일, 심사완료: 2004년 1월 15일
* 본 논문은 2002년도 공주대학교 자체학술연구비 지원에 의하여 연구되었음.

는 것이 중요하다. 즉, 그룹 멤버들 사이 전달되는 메시지가 무결성, 기밀성, 확장성이 제공되어야 한다[4-6].

수신자의 수를 제한하거나 데이터의 출처를 인증하여야 하는 응용에서는 멀티캐스트의 인증과 프라이버시가 지원되어야 한다. 디지털 미디어의 PPV(pay-per-view), PPU(pay-per-use) 게임, 원격회의와 같은 응용은 수신자가 정상적인 가입자들로 제한된다. 더불어 주식정보를 분배하는 응용은 데이터의 출처를 인증하는 것이 중요하다.

안전한 멀티캐스트 그룹은 키 서버에 의해 관리된다. 이러한 키 서버를 그룹제어기라 한다. 안전한 멀티캐스트 그룹에 가입하기 위하여 클라이언트는 그룹제어기에게 그룹에 접근을 요구하여야 한다. 그룹제어기는 요구를 받으면 클라이언트의 로그인명과 패스워드나 증명서에 의해 신원을 확인한다. 클라이언트가 그룹에 가입이 허용되면 제어기는 메시지가 전달될 그룹주소와 필수적인 키를 클라이언트에게 제공한다. 클라이언트에게 그룹의 모든 멤버들에 의해 공유되는 그룹키와 키분배 알고리즘에서 사용되는 보조키를 분배한다.

멤버의 가입과 탈퇴는 멀티캐스트 키 관리의 확장성(scalability) 문제와 밀접한 관계가 있다. 1개의 그룹키를 공유하며 N개의 멤버로 구성되는 멀티캐스트 그룹을 생각하여 보자. 그룹에 한 멤버가 가입하면 가입자에게만 그룹키를 암호화하여 유니캐스트 통신으로 전달하면 된다. 그룹의 한 멤버가 탈퇴한다면, 그룹키를 변경하여 그룹의 나머지 모든 $N-1$개의 멤버에게 전달되어야 한다. 그러나 새로운 그룹키를 나머지 모든 멤버들에게 안전하게 보내는 것은 간단한 문제가 아니다. 아주 간단한 해결책은 제어기와 나머지 멤버들간에 유일한 키를 가지고 유니캐스트 통신을 하는 것이다. 이 방법은 간단하지만 $N-1$개의 유니캐스트 연결과 N개의 비밀키를 요구하기 때문에 확장성이 좋지 않다.

Chang[7], Wong[8], Zhu[9]과 Noubir[10, 11]는 확장성을 제공하기 위하여 계층구조의 그룹키를 갖는 관리 구조를 제안하였다. Chang과 Wong은 2진 키 트리를, Zhu와 Noubir는 레벨에 따라 다른 노드를 갖는 트리를 사용하였다. 2진 키 트리는 관리가 간단하는 장점이 있다.

N이 그룹의 크기라 하면, Chang모델에서는 키를 갱신하기 위하여 $\log N$개의 메시지가 필요하며, 각 멤버는 $\log N$개의 키 집합을 관리해야 하고, 제어기는 $2\log N+1$개의 키를 관리하여야 한다. Wong은 키의 고정된 계층구조를 사용하는 반면에 Chang은 다른 키를 조합함으로서 동적인 키를 생성한다.

Chang과 Wong모델에서는 1개의 멤버를 그룹으로부터 제거하기 위하여 제어기가 각각 $\log N$개와 $2\log N-1$개의 메시지를 멤버들에게 보낸다. 그룹의 크기가 크고 탈퇴가 빈번할 경우, 각 멤버를 제거할 때마다 새로운 그룹키를 갱신하고 분배하는 것은 많은 연산을 요구한다[12-14]. 대부분의 응용에서는 멤버탈퇴 요구가 있을 때 즉시 처리할 필요가 없이 주기적으로 탈퇴할 멤버를 모아서 동시에 제거한다.

본 논문에서는 Chang과 Wong의 키 분배 구조와 유사한 구조를 사용하지만, 제거대상 멤버들의 제거 라운드를 조정함으로서 이들 모델보다 메시지의 수와 메시지를 암호화하는 복합키의 연산 비용을 줄일 수 있는 키 관리구조를 제안하고 이들 모델과의 성능을 비교분석한다.

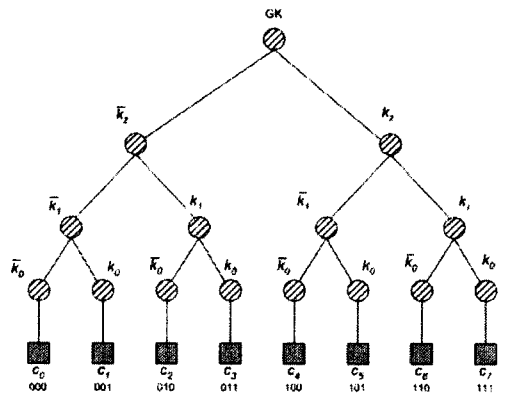
2. 키관리 체제

그룹의 각 멤버에는 n비트 길이의 2진수 문자열을 가진 사용자 ID(UID)가 부여된다. UID는 $x_{n-1}x_{n-2}\cdots x_0$로 표현되어지며 각 x_i는 0이나 1이다. UID의 길이는 그룹 크기에 의존한다. 예를 들면 그룹의 크기가 300이면 9비트 UID를 사용한다.

UID $x_{n-1}x_{n-2}\cdots x_0$인 멤버가 그룹에 가입하기 위하여 그룹 제어기에게 등록을 하면 제어기는 멤버에게 공용 그룹키 GK를 보낸다. GK는 그룹의 모든 멤버들이 공유하며, 멀티캐스트 그룹에게 메시지를 보낼 때 암호화하거나 복호화할 때 사용된다. 또한 멤버는 n 개의 보조키 세트들 중에서 1개의 세트 $K_{n-1}, K_{n-2}, \cdots, K_0$를

받는다. 여기서 K_i는 x_i가 1이면 k_i, 0이면 $\overline{k}_i$로 표현된다. 보조키는 그룹키를 안전한 방법으로 갱신하는데 사용된다. 키의 쌍 $\{k_i, \overline{k}_i\}$은 UID의 i번째 비트에 대응된다. k_i와 $\overline{k}_i$는 서로 보수키(complement key)이며, 수치적인 보수관계가 아니라 서로 관계가 없는 키라는 의미이다. 멤버는 n 개의 키의 쌍으로부터 각각 1개의 키를 제어기로부터 받는다. 제어기는 모든 보조키 $\{k_0, \overline{k}_0, k_1, \overline{k}_1, \cdots, k_{n-1}, \overline{k}_{n-1}\}$를 관리한다.

<그림 1>은 그룹의 크기가 8인 경우에 각 멤버가 보관하는 키를 보여주고 있다. 트리에서 사각형 단말 노드는 3비트의 UID를 가지는 그룹의 멤버를 나타내고, 트리의 원형 노드는 키를 나타낸다. 각 멤버는 단말 노드에서 루트 노드까지 가는 노드에 있는 키를 보관한다. 예를 들면, 멤버 c_5(UID 101)은 그룹키 GK와 보조키 $k_2, \overline{k}_1, k_0$를 보관한다.



<그림 1> 그룹크기 8의 키 분배트리

일반적으로 그룹키와 보조키는 그룹 멤버가 탈퇴하거나 강제 추방할 때 변경된다. 이와 같이 키의 변경을 그룹 rekey이라 한다. 그룹 제어기는 일정한 주기마다 그룹에서 탈퇴할 멤버를 제거하기 위하여 rekey를 한다. 이러한 주기를 라운드라고 하고, 라운드의 주기는 제어기가 임의로 정할 수 있다. rekey는 불연속적으로 일어나므로 그룹키와 보조키를 각각 $GK(r)$과 $k_i(r)$, $\overline{k}_i(r)$로 표현한다. 여기서 r은 현재의 라운드를 나타낸다.

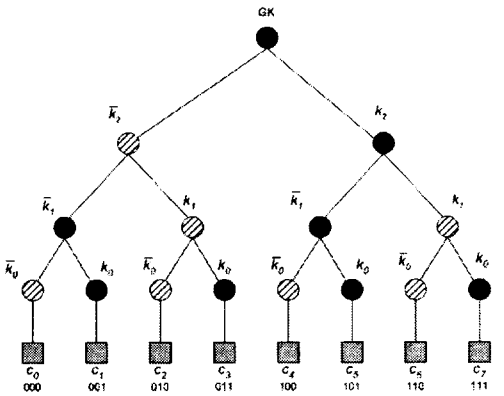
2.1. 한 개의 멤버 제거

가입이 만기가 되어 그룹의 한 멤버를 그룹으로부터 강제적으로 제거할 때마다 새로운 그룹키가 그룹을 떠나는 멤버를 제외한 모든 멤버들에게 분배되어야 한다. 유사하게 멤버가 자발적으로 그룹을 떠날 때 그룹 제어기의 rekey정책에 따라 그룹키가 갱신되어야 한다.

현재의 그룹키 $GK(r)$를 갱신하기 위하여 제어기는 새로운 그룹키 $GK(r+1)$를 계산한다. $GK(r+1)$는 제거되는 멤버의 보수키를 사용하여 암호화한다. 예를 들면 제거되는 멤버의 UID가 101이라면 그 멤버는 보조키 k_2, $\overline{k}_1$, k_0을 가지고 있다. 그러므로 $GK(r+1)$는 그 멤버의 보수키인 $\overline{k}_2$, k_1, $\overline{k}_0$에 의해 각각 암호화한 3개의 메시지 $\{GK(r+1)\}_{\overline{k}_0}$, $\{GK(r+1)\}_{k_1}$, $\{GK(r+1)\}_{\overline{k}_2}$를 만들어 그룹의 모든 멤버들에게 멀티캐스트한다. 여기서 $\{L\}_M$은 문자열 L을 키 M에 의해 암호화하여 그룹 전체멤버에게 보낸다는 의미이다. 제거될 멤버도 암호화된 모든 메시지를 수신하지만 모든 메시지는 그 멤버가 가지고 있지 않은 키로 암호화되어 있기 때문에 복호화할 수 없다. 그러나 그룹의 나머지 멤버들의 UID는 제거될 멤버의 UID와 적어도 1비트 이상 다르므로 그룹의 나머지 멤버들은 암호화된 메시지 중의 적어도 1개 이상을 복호화할 수 있다.

<그림 2>는 멤버 c_5가 제거될 때의 그룹 rekey체제를 나타낸다. 짙은 원형노드는 그룹에서 제거되는 멤버 c_5가 가지고 있는 보조키를 나타낸다. 빗금친 원형노드는 c_5가 가지고 있지 않는 보조키 즉, c_5의 보수키를 나타낸다. c_5에서 루트까지 가는 노드는 모두 짙은 원형 노드이지만 그 외의 노드에서 루트까지 가는 노드에는

적어도 1개 이상의 빗금친 노드가 존재한다. 그러므로 c_5를 제외한 모든 멤버는 c_5가 가지고 있지 않은 보조키에 의해 암호화된 적어도 1개 이상의 메시지를 복호화할 수 있다.



<그림 2> 1개의 멤버 c_5가 제거되는 예

키 분배 알고리즘을 분석하면, 그룹 멤버수가 N인 경우, 제어기에 의해 관리되는 키의 수는 $2\log N+1$개이며, 1개의 멤버가 제거된 후에 그룹키를 갱신하기 위하여 보내는 메시지의 수는 $\log N$개 이다.

제거되는 멤버는 새로운 그룹키를 획득할 수 없다. 그룹키를 다음 라운드에서 갱신할 경우 제거되는 멤버가 이 그룹키를 복호화하지 못하도록 보조키를 갱신하여야 한다. 각 멤버는 보조키 $K_i(r)$를 식 (1)과 같이 갱신하기 위하여 해쉬함수 f를 사용한다.

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

새로운 그룹키 $GK(r+1)$를 가지고 있는 멤버만이 보조키 $K_i(r+1)$를 갱신할 수 있다. 제거되는 멤버는 $GK(r+1)$를 가지고 있지 않기 때문에 새로운 보조키를 갱신할 수 없으며 다음 라운드에서 변경될 그룹키, $GK(r+2)$를 갱신할 수 없다.

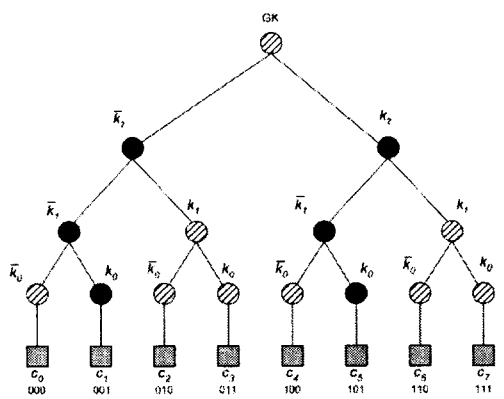
2.2. 여러 개의 멤버 제거

k개의 멤버를 제거하기 위하여 2.1절의 키 갱신 과정을 k번 반복할 수 있다. 그러나 더욱 효과적인 방법은 제거할 멤버들을 모아서 한번에 제거하는 것이다.

멤버의 집합, $S=\{c_0, c_1, \cdots, c_{N-1}\}$이며 $N=2^n$이다. 멤버 c의 UID를 $x_{n-1}x_{n-2}\cdots x_0$로 표현한다. 여기서 x_i, $i=0, 1, \cdots, n-1$는 0이나 1의 값을 가진다. 어떤 시점에서 그룹의 멤버쉽은 UID의 부울함수 $mem()$에 의해 결정된다. 즉, $mem(x_{n-1}x_{n-2}\cdots x_0)=1$이라면 $x_{n-1}x_{n-2}\cdots x_0$는 그룹의 멤버이고, 그렇지 않으면 그룹에서 제거된다.

그룹 rekey는 $mem(\mathrm{UID})=0$인 멤버를 제외한 모든 멤버에 대하여 그룹키와 보조키를 갱신하는 것이다. 이 키들은 제거될 멤버에게 알려지지 않은 키에 의해 암호화하여 모든 멤버들에게 전송된다. 확장성과 효율성을 고려하여 rekey는 그룹에 전송될 최소의 메시지와 암호화하는데 최소의 연산이 소요되는 것이 바람직하다.

예를 들어보면, <그림 3>에서 UID 001(c_1)과 101(c_5)가 그룹으로부터 제거된다고 가정하자.



<그림 3> 2개의 멤버 c_1, c_5가 제거되는 예

그룹의 나머지 멤버의 집합 $S=\{c_0, c_2, c_3, c_4,$

$c_6, c_7\}$에게 새로운 그룹키, $GK(r+1)$을 분배하여야 한다. 일반적으로 모든 UID가 할당되는 것이 아니므로, 실제 그룹키를 분배하여야 되는 멤버는 S의 부분집합이다. UID가 할당되지 않은 멤버는 키 분배에 영향을 주지 않는 "don't care" 조건으로 취급할 수 있다. 암호화된 그룹키, $\{GK(r+1)\}_{\bar{k}_0}$와 $\{GK(r+1)\}_{k_1}$를 갖는 2개의 메시지를 멀티캐스트하면 된다. 첫 번째 메시지는 멤버 c_0, c_2, c_4, c_6에 의해 복호화될 수 있으며, 두 번째 메시지는 멤버 c_2, c_3, c_6, c_7에 의해 복호화될 수 있다. 그리하여 2개의 메시지는 그룹의 나머지 멤버의 집합 S에서 c_1, c_5를 제외한 그룹을 커버하므로 그룹 rekey가 가능하다. <그림 3>에서 짙은 원형노드는 c_1, c_5가 가지고 있는 키를 나타내며, 이들은 $GK(r+1)$을 암호화하는데 사용할 수 없다.

1개의 멤버를 제거하기 위한 rekey는 3개의 메시지가 필요하였다. 1개의 멤버를 제거하고 나서 다음 다른 멤버를 제거하는 식의 순서적으로 rekey가 이루어진다면, 2개의 멤버가 제거될 경우에는 2×3=6개의 메시지를 멀티캐스트하여야 한다. 그러나 2개의 멤버를 모아서 동시에 제거할 경우에는 단지 2개의 메시지만 멀티캐스트하면 된다. 이 수치는 1개의 멤버를 제거할 경우보다 더 적은 수치이다. 직관적으로 그룹의 나머지 멤버들의 UID가 공통인 비트가 1개 이상(공통인 비트는 제거되는 UID의 비트와 다름)이면 멀티캐스트될 메시지의 수는 줄어들게 됨을 알 수 있다. 위의 예에서 c_0, c_2, c_4, c_6의 UID의 x_0은 제거되는 멤버 (c_1, c_5)의 UID의 x_0와 달리 모두 0이다. 제거되는 멤버는 k_0를 가지고 있는 반면에 c_0, c_2, c_4, c_6는 $\bar{k}_0$를 가지고 있기 때문에 $\{GK(r+1)\}_{\bar{k}_0}$를 복호화할 수 있다. 유사하게 c_2, c_3, c_6, c_7의 x_1은 제거되는 멤버 (c_1, c_5)의 UID의 x_1와 달리 모두 1이다. 제거되는 멤버는 $\bar{k}_1$를 가지고 있는 반면에 c_2, c_3, c_6, c_7은 k_1을 가지고 있기 때문에 $\{GK(r+1)\}_{k_1}$을 복호화할 수 있다.

제거할 멤버를 모아서 주기적으로 동시에 rekey하는 것을 주기적 rekey(periodic rekey)라 한다. 주기적 rekey는 제거되는 맴버를 제외한 나머지 멤버들의 UID의 비트가 서로 공통인 멤버들의 그룹을 체계적으로 찾는 것과 같은 문제이다. 이러한 문제는 부울 멤버쉽 함수의 최소화와 같은 문제이다. 예를 들면, 멤버쉽 함수는 다음과 같이 표현할 수 있다.

$$mem(x_2, x_1, x_0) = \bar{x}_2\bar{x}_1\bar{x}_0 + \bar{x}_2x_1\bar{x}_0 + x_2\bar{x}_1x_0 + x_2x_1x_0 + x_2\bar{x}_1\bar{x}_0 + x_2x_1\bar{x}_0 + x_2x_1x_0 \quad (2)$$

여기서 +는 논리적인 OR이며 변수들의 곱은 논리적인 AND를 나타낸다. 식(2)의 각 항은 1개의 메시지에 대응된다. 각 항을 구성하고 있는 키들을 입력으로 하는 일방향 함수에 의해 생성된 복합키로 암호화하여 1개의 메시지를 생성한다. 예를 들면, $x_2\bar{x}_1\bar{x}_0$항은 $k_2, \bar{k}_1, \bar{k}_0$로부터 생성되는 복합키에 의해 암호화되는 메시지에 대응된다. 그러므로 그룹의 모든 멤버들에게 새로운 그룹키를 전달하기 위하여 6개의 메시지를 멀티캐스트하여야 한다. 이러한 각 메시지들은 그룹에 남아 있는 6개의 멤버들 중의 1개에 의해 복호화된다.

그러므로 멤버쉽 함수를 최소화하는 필요성이 나오게 된다. 이와 같은 유사한 문제는 논리회로 설계에서 사용되는 부울함수의 최소화문제와 같다. 논리회로설계에서 부울함수의 최소화의 목적은 게이트의 수와 게이트의 크기를 줄이는 것이다. 논리회로 설계방법을 이용하여 더욱 효율적으로 rekey를 한다. 앞으로 사용될 부울함수에서 나오는 용어의 정의는 다음과 같다.

리터럴(literal): 변수나 그의 보수, 예를 들면, $x_1, \bar{x}_1$ 등

곱의 항(product term): AND항, 예를 들면, $x_1\bar{x}_2x_3$

합의 항(sum term): OR항, 예를 들면, $x_1 + \bar{x}_2$

최소항(minterm): 함수의 모든 변수를 사용한 곱의 항, 예를 들면 식(2)의 6개의 항은 모두 최

소항이다.

일반적으로 부울함수의 최소화에 사용되는 표준형은 곱의 합(sum of products)형태이다. 논리회로설계에서 곱의 항은 리터럴을 입력으로 하는 AND게이트에 해당한다. 곱의 항은 그룹 rekey문제에서 메시지에 대응되며, 각 리터럴은 메시지의 암호/복호키를 생성하는 함수의 입력으로 사용되는 키에 대응된다. 곱의 항의 수를 더 이상 줄일 수 없거나 곱의 항의 수는 같지만 각 항의 리터럴 수를 더 이상 줄일 수 없을 때 최소의 곱의 합 형태이다.

부울함수를 최소화하기 위하여 카르노맵(Karnaugh map)[15]이 사용된다. 카르노맵은 도식적으로 최소의 곱의 합을 구하는 방법이다. 그러나 변수의 수가 많을 때는 카르노맵을 사용할 수 없기 때문에 Quine-McCluskey 알고리즘[15]을 사용하여 컴퓨터 프로그램으로 최소화하여야 한다. 여러 개의 멤버가 같은 라운드에서 그룹을 떠날 때, 제어기는 그룹에 남아 있는 멤버들에게 보낼 메시지를 계산하기 위하여 Quine-McCluskey알고리즘을 실행한다.

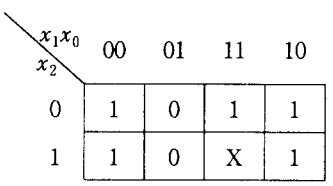
부울함수의 최소화방법이 최소의 메시지와 최소의 키를 가지는 키 갱신 문제에 어떻게 적용되는지를 이해하기 위하여 예를 들어 설명한다. c_7이 그룹을 이미 떠났다고 가정하고, c_7를 제외한 나머지 7개의 멤버들에게는 UID가 할당되어 있다고 가정하자. 그룹으로부터 c_1과 c_5를 제거한다면 멤버쉽 함수는 <표 1>과 같다. c_1과 c_5의 출력은 0, 그룹을 이미 떠난 c_7은 X(don't care), 나머지 멤버들의 출력은 1이다. c_7은 그룹을 떠난 후 보조키를 갱신시켰기 때문에 새로운 키에 의해 암호화된 메시지를 복호화할 수 없다. 멤버쉽 함수를 최소항으로 나타내면 식 (3)과 같이 표현할 수 있다. m과 d는 각각 최소항과 don't care항을 나타낸다.

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

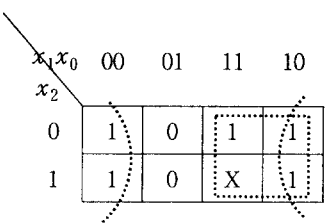
<표 1> 멤버쉽 함수

입력 ($x_2x_1x_0$)	출력
000	1
001	0
010	1
011	1
100	1
101	0
110	1
111	X

<그림 4(a)>는 <표 1>의 카르노맵을 나타내고 있다. 카르노맵의 사각형은 최소항이다. 최소화하는 과정은 <그림 4(b)>와 같이 인접해 있는 1이나 X의 최소항들의 블럭을 가능한 크게 만든다. 블럭에 속하는 최소항의 수는 2의 멱승이 되도록 하여야 한다. 이러한 블럭을 주항(prime implicant)라 한다. 주항의 수를 최소로 함으로서 함수의 최소화가 이루어진다. <표 1>은 $x_1 + \bar{x}_0$로 최소화된다.



(a) 카르노맵



(b) 주항의 선택

<그림 4> 멤버쉽 함수의 카르노맵과 최소화

<그림 4>에서 부울함수의 최소화와 주기적인 rekey간의 관계를 다음과 같이 해석할 수 있다.

- 0인 최소항은 그룹에서 제거되어야 하는 멤버
- 1인 최소항은 그룹에 남아 있는 멤버
- X인 최소항은 아직까지 할당받지 못한 UID

주기적인 rekey에서 갱신할 그룹키는 카르노맵에서 가장 적은 수의 주항을 찾는 문제와 밀접한 관계가 있다. 각 주항은 그룹키를 암호화하는데 사용하는 1개의 복합키에 대응하므로 그룹키를 갖는 1개의 메시지에 대응한다. 위의 예에서 그룹키를 갱신하기 위하여, 암호화된 그룹키 $\{GK(r+1)\}_{k_1}$와 $\{GK(r+1)\}_{\overline{k}_0}$를 포함하는 2개의 메시지를 그룹에 멀티캐스트한다.

2.3. 라운드 조정 알고리즘

본 논문에서는 그룹 탈퇴멤버들의 UID를 체크하여 동시에 제거될 멤버들의 라운드를 조정하는 알고리즘을 사용하면 Chang모델보다 그룹키를 암호화하는데 사용할 키의 수를 줄일 수 있다. 그룹키를 암호화하는데 사용할 복합키의 수를 줄일 수 있으며 복합키의 연산비용을 줄일 수 있다.

해밍거리는 2개의 UID를 이진 비트로 표현할 때 서로 다른 비트의 수이다. 제거될 멤버를 선택하기 위하여 제어기가 임의로 설정한 임계치가 적용된다. 제거될 멤버들의 해밍거리가 작으면 그룹의 나머지 멤버들에게 분배할 그룹키를 작은 연산비용으로 암호화할 수 있으며, 메시지 수를 줄일 수 있다.

[라운드 조정 알고리즘]

단계 1: 제거될 멤버의 UID를 큐에 저장한다. 모든 UID끼리의 해밍거리를 계산한다.
단계 2: 큐에 저장된 제거 대상들 간의 해밍거리가 주어진 임계치(threshold)보다 작은 UID의 멤버만 제거 대상으로 선택한다.
단계 3: 선택된 UID를 제외한 나머지 그룹멤버들에게 분배할 새로운 그룹키를 암호화할 키를 생성한다.
단계 4: 선택된 UID는 큐에서 제거된다. 단계 1로 간다.

라운드 조정 알고리즘을 설명하기 위하여 예를 들어 보자. 첫 번째 라운드에서 그룹의 크기가 16이고 UID 1111(c_{15}), 0011(c_3), 0000(c_0)을 그룹으로부터 제거하고자 한다. 이들을 큐에 순서대로 입력한다. 큐에 있는 멤버들간의 해밍 거리를 계산하면 다음과 같다.

$c_{15} \leftrightarrow c_3$: 2

$c_{15} \leftrightarrow c_0$: 4

$c_3 \leftrightarrow c_0$: 2

제어기가 임계치를 3으로 설정하였다면, c_{15}와 c_3간, c_3와 c_0 간의 해밍거리는 임계치보다 작고, c_{15}와 c_0의 해밍거리는 임계치보다 크므로 마지막에 입력한 c_0를 제외한 c_{15}와 c_3가 함께 제거대상으로 선택되어 첫번째 라운드에서 그룹으로부터 제거된다. 멤버쉽 함수는 다음과 같다.

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

멤버쉽 함수에 대한 카르노 맵은 <그림 5(a)>와 같다. 카르노 맵을 최소화하면 $\overline{x}_0 + \overline{x}_1 + \overline{x}_3 x_2 + x_3 \overline{x}_2$이므로 그룹키를 갱신하기 위하여 4개의 메시지 $\{GK(r+1)\}_{\overline{k}_0}$, $\{GK(r+1)\}_{\overline{k}_1}$, $\{GK(r+1)\}_{f(\overline{k}_3, k_2)}$, $\{GK(r+1)\}_{f(k_3, \overline{k}_2)}$를 멀티캐스트한다. 여기서 $f(\overline{k}_3, k_2)$는 $\overline{k}_3$와 k_2을 입력으로 하는 함수에 의해 생성된 복합키이다. $\{GK(r+1)\}_{\overline{k}_0}$는 $\{c_0, c_2, c_4, c_6, c_8, c_{10}, c_{12}, c_{14}\}$, $\{GK(r+1)\}_{\overline{k}_1}$는 $\{c_0, c_1, c_4, c_5, c_8, c_9, c_{12}, c_{13}\}$, $\{GK(r+1)\}_{f(\overline{k}_3, k_2)}$는 $\{c_4, c_5, c_6, c_7\}$, $\{GK(r+1)\}_{f(k_3, \overline{k}_2)}$는 $\{c_8, c_9, c_{10}, c_{11}\}$에 의해 각각 복호화할 수 있다.

두 번째 라운드가 시작하기 전에 새로 c_2, c_{10}이 그룹으로부터 제거되기 위하여 큐에 입력되었다고 가정하자. 현재 큐에는 c_0, c_2, c_{10}이 순서대로 기억되어 있다. 두 번째 라운드에서는 큐에 있는 멤버들간의 모든 해밍거리가 임계치보다 작으므로 큐에 있는 모든 멤버들이 제거대상

으로 선택된다. 멤버쉽 함수는 다음과 같다.

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

멤버쉽 함수에 대한 카르노맵은 <그림 5(b)>와 같다. 카르노맵을 최소화하면 $x_0 + x_2 + x_3 \overline{x}_1$ 이므로 그룹키를 갱신하기 위하여 3개의 메시지 $\{GK(r+1)\}_{k_0}$, $\{GK(r+1)\}_{k_2}$, $\{GK(r+1)\}_{f(k_3, \overline{k}_1)}$를 멀티캐스트한다.

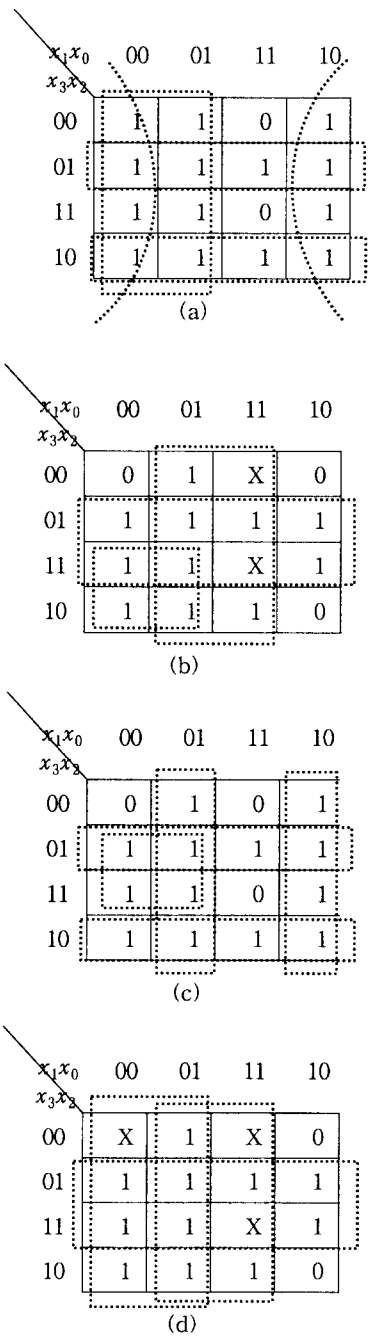
Chang 모델과 같이 그룹을 rekey하면 첫 번째 라운드에서는 <그림 5(c)>와 같이 카르노맵을 최 소 화 하 면 $x_3 \overline{x}_2 + x_2 \overline{x}_1 + x_1 \overline{x}_0 + x_0 \overline{x}_1 + x_2 \overline{x}_3$이므로 rekey하기 위하여 5개의 메시지가 필요하며 각 메시지는 2개의 보조키에 의해 생성된 복합키를 사용하여 암호화된다. 두 번째 라운드에서의 카르노맵의 최소화는 <그림 5(d)>와 같이 $\overline{x}_0 + x_1 + x_2$이므로 3개의 메시지가 필요하며 각 메시지는 1개의 보조키에 의해 암호화된다.

라운드 조정 알고리즘을 수행하면 첫 번째와 두 번째 라운드에서 그룹키를 갱신하기 위하여 총 7개의 메시지를 분배하여야 한다. 그 중에서 1개의 보조키를 사용하여 암호화한 메시지는 4개이며, 2개의 보조키에 의해 생성된 복합키를 사용하여 암호화한 메시지는 3개의 메시지이다.

Chang모델에서는 첫 번째와 두 번째 라운드에서 총 8개의 메시지를 분배하여야 한다. 그 중에서 1개의 보조키를 사용하여 암호화한 메시지는 3개이며, 2개의 보조키에 의해 생성된 복합키를 사용하여 암호화한 메시지는 5개의 메시지이다.

부울함수에서 주항의 수는 메시지의 수, 주항을 구성하고 있는 리터럴의 수는 그룹키를 갱신하는데 사용할 보조키의 수이다. 주항을 구성하는 리터럴의 수가 적으면 복합키를 생성하는데 필요한 연산의 비용을 줄일 수 있다. 첫 번째와 두 번째 라운드에서는 메시지의 수를 줄일 수 있을 뿐 아니라 복합키를 생성하는 연산 비용도 줄었다.

제거될 멤버들간의 해밍거리가 작을수록 그룹에 남아있는 멤버들을 더 큰 블록으로 묶을 수 있게 되어 블록(주항)의 수가 줄어들거나 주항의 리터럴수가 줄어들게 된다. 이것은 그룹키를 암호화하는 메시지의 수와 복합키의 연산비용이 줄어들게 된다.



<그림 5> 멤버쉽 함수의 최소화

3. 성능분석

이 장에서는 제안된 키 관리체제에 대한 성능을 분석한다. 여러 개의 멤버가 그룹을 떠날 때 그룹키를 갱신하기 위하여 요구되는 메시지의 총수의 관점에서 성능분석을 한다. 2개의 멤버가 그룹을 떠나는 경우와 $N/2=2^{n-1}$개의 멤버들이 떠날 때 rekey를 하기 위하여 분배하여야 하는 최악의 메시지 복잡도를 유도한다.

3.1. 두 개의 멤버 동시제거

2개의 멤버 c_1, c_2가 그룹으로부터 제거된 후에 새로운 키를 분배하여야 한다. c_1, c_2의 UID를 각각 2진수로 $x_{n-1}x_{n-2}\cdots x_0$와 $y_{n-1}y_{n-2}\cdots y_0$로 표현한다. UID가 n비트이므로 2^n개의 멤버가 있을 수 있다.

2개의 UID의 해밍거리는 다음과 같을 때 최대가 된다.

$$y_i = \overline{x}_i, \quad i=0, 1, \cdots, n-1$$

여기서 $\overline{x}_i$는 x_i의 보수이다. 그러므로 c_1은 키 $k_{n-1}, k_{n-2}, \cdots, k_1, k_0$, c_2는 $\overline{k}_{n-1}, \overline{k}_{n-2}, \cdots, \overline{k}_1, \overline{k}_0$를 가지고 있다. 이와 같은 경우가 그룹의 나머지 멤버들에게 새로운 키를 보내기 위하여 가장 많은 메시지를 보내야 하는 최악의 상황이다. 최악의 경우에 그룹 rekey를 위하여 식(4)이나 식(5)과 같은 메시지가 필요하다. 각 식은 많아야 n개의 메시지가 필요하다.

$$\{GK(r+1)\}_{f(k_{n-1}, \overline{k}_{n-2})}, \{GK(r+1)\}_{f(k_{n-2}, \overline{k}_{n-3})}, \cdots, \{GK(r+1)\}_{f(k_0, \overline{k}_{n-1})} \quad (4)$$

$$\{GK(r+1)\}_{f(\overline{k}_{n-1}, k_{n-2})}, \{GK(r+1)\}_{f(\overline{k}_{n-2}, k_{n-3})}, \cdots, \{GK(r+1)\}_{f(\overline{k}_0, k_{n-1})} \quad (5)$$

식(4)과 식(5)의 n개의 각 항은 1개의 복합키를 사용하여 암호화하는 그룹키 $GK(r+1)$를 포함하는 메시지를 나타낸다. 암호화에 사용되는 복합키는 2개의 보조키를 입력으로하는 일방향 함수로부터 유도된다. 일방향 함수 $f(k_1, k_2)$는 k_1k_2와 같이 간단하게 구현되거나 해쉬함수를 사용하여 구현될 수 있다. k_1k_2는 $GK(r+1)$을 첫 번째 보조키 k_1로 암호화한 후에 그 결과를 두 번째 보조키 k_2로 암호화한다는 의미이다.

c_1과 c_2는 각 메시지를 암호화하는데 사용되는 2개의 키 중에서 단지 1개만 가지고 있기 때문에 이러한 메시지를 복호화할 수 없다. c_1, c_2를 제외한 그룹의 나머지 멤버들은 식(4)이거나 식(5)의 n개의 메시지 중에서 적어도 1개 이상의 메시지를 복호화할 수 있다.

그러므로 2^n 멀티캐스트 그룹에서 2개의 멤버가 그룹을 떠날 경우 rekey하기 위하여 많아야 n개의 메시지가 필요하다.

3.2. $N/2$ 개의 멤버 동시제거

$N=2^n$의 그룹크기에서 $N/2$개의 멤버가 제거될 때, 나머지 $N/2$개의 멤버들에게 새로운 키를 분배하여야 하는 경우를 보자. 그룹에 남아있는 나머지 멤버들의 임의의 2개의 멤버들간의 해밍거리가 2이상이 될 때 최악의 복잡도가 나오게 된다. UID가 $x_{n-1}x_{n-2}\cdots x_0$인 그룹에서 1개의 멤버가 <그림 6>의 카르노맵에서 보여주는 것처럼 그 멤버와 한 비트가 다른 제거될 n개의 멤버들로 둘러싸여 있을 때 최악의 경우이다. <그림 6>으로부터 부울함수를 더 이상 최소화할 수 없음을 직관적으로 알 수 있다. 즉, rekey를 하기 위하여 각 멤버 당 1개의 메시지를 보내야 하므로 $N/2$개의 메시지가 필요하다. 그러므로 그룹의 크기가 $N=2^n$일 때, rekey하기 위하여 많아야 $N/2$ 메시지를 분배하여야 한다.

x_3x_2 \ x_1x_0	00	01	11	10
00	1	0	1	0
01	0	1	0	1
11	1	0	1	0
10	0	1	0	1

<그림 6> $N/2$ 개의 멤버 동시제거시 최악의 경우

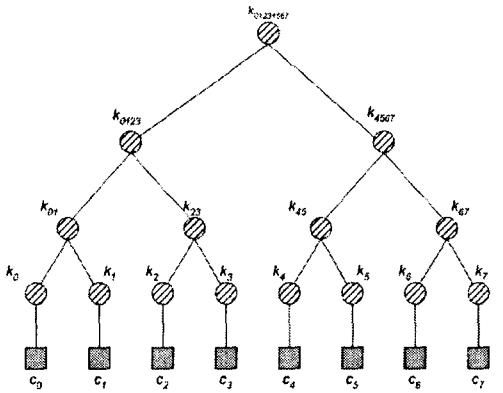
3.3 비교분석

본 논문에서 제안한 모델(LCR, Low Cost Rekey)과 기존의 좋은 확장성을 보여준 모델인 Wong, Chang의 모델을 비교분석함으로서 LCR 모델의 성능이 우수함을 보인다. 3개의 모델간에 메시지 복잡도, 제어기의 키 복잡도, 멤버의 키 복잡도, 복합키의 연산비용에 대하여 비교한다.

메시지 복잡도는 멤버 탈퇴시 rekey를 위하여 최악의 경우에 얼마나 많은 메시지를 분배하여야 하는지를 나타내며 확장성의 중요한 척도이다. 제어기의 키 복잡도는 그룹키 관리를 위하여 모든 멤버들이 가지고 있는 키들을 생성하여 분배하여야 한다. Rekey를 위하여 제어기가 얼마나 많은 키를 관리하여야 하는지를 나타낸다. 멤버의 키 복잡도는 그룹키를 관리하기 위하여 각 멤버가 제어기로부터 얼마나 많은 키를 받아서 관리하여야 하는지를 나타낸다. 복합키의 연산비용은 그룹키를 암호화하는데 필요한 복합키를 연산하는 비용을 나타낸다.

<그림 7>은 그룹의 크기가 8인 Wong의 키 관리체제를 나타낸다.

트리에서 빗금친 원형 노드는 키, 사각형 노드는 멤버를 나타내며 트리의 루트노드는 그룹키를 나타낸다. 1개의 멤버가 제거될 때 트리의 단말 노드로부터 루트노드까지의 모든 키가 변경되어야 한다. 갱신된 키는 그룹의 각 멤버들에게 유니캐스트하는 것이 아니라 그룹에 멀티캐스트한다. 예를 들면, 멤버 c_5가 그룹을 떠날 때, 키 k_{45}, k_{4567}, $k_{01234567}$이 갱신되어야 한다. 키 k_{45}는 k_4에 의해 암호화되며, 키 k_{4567}은 k_{67}과 새로 갱신된 k_{45}에 의해 암호화된다. 또한 키 $k_{01234567}$은 k_{0123}과 새로 갱신된 k_{4567}에 의해 암호화되어 전체 그룹에 멀티캐스트한다.



<그림 7> Wong모델의 키 관리체제

크기가 N인 Wong모델에서 1개의 멤버를 제거할 때, rekey를 위하여 $2\log N-1$개의 메시지를 멀티캐스트하여야 하며, 각 멤버는 $\log N$개의 키를 관리하여야 한다. 그리고 제어기는 $2N-1$개의 키를 가지는 트리를 관리하여야 한다. 그룹 크기가 8일 때 Wong모델은 15개의 복합키가 사용되며, LCR과 Chang모델은 6개의 보조키와 1개의 그룹키가 사용된다.

멤버의 키 복잡도는 Wong과 Chang, LCR모델이 모두 같으며 $O(\log N)$이다. 제어기가 관리하는 키의 복잡도는 Wong, Chang과 LCR모델이 각각 $O(2N)$, $O(2\log N)$와 $O(2\log N)$이다. 1개의 멤버를 제거할 때 분배하는 메시지의 복잡도는 Wong, Chang, LCR모델의 경우 각각 $O(2\log N)$과 $O(\log N)$, $O(\log N)$이다.

여러 개의 멤버를 제거할 때 최악의 메시지 복잡도를 비교하여보자. Wong모델에서 최악의 메시지 복잡도는 키 트리의 최하의 모든 서브 트리에 있는 2개의 멤버중 1개가 탈퇴할 경우이다. 이때 키 트리의 모든 보조키가 갱신되어야 하므로 최악의 경우이며 복잡도는 $O(N)$이다. Chang 모델에서 최악의 메시지 복잡도는 각 멤버간의 해밍거리가 2이상인 $N/2$개의 멤버가 동시에 제거 될 때이며 복잡도는 $O(N/2)$이다. LCR모델

에서 최악의 메시지 복잡도는 라운드 조정 알고리즘을 수행하므로 Chang모델의 최악의 메시지 복잡도보다 낮다.

<표 2>는 그룹에서 1개의 멤버가 제거될 때, rekey를 하기 위하여 제어기가 멀티캐스트하는 최악의 메시지 복잡도, 제어기가 관리하는 키 복잡도, 멤버가 관리하는 키 복잡도에 대하여 Wong, Chang, LCR간의 비교를 보여 주고 있다. 또한 <표 3>은 여러 개의 멤버가 제거될 때, 제어기가 멀티캐스트하는 최악의 메시지 복잡도, 제어기가 관리하는 키 복잡도, 멤버가 관리하는 키 복잡도, 복합키를 생성하기 위한 연산비용에 대하여 3 개의 모델간의 비교를 나타내고 있다.

제거될 멤버들간의 해밍거리가 클수록 많은 메시지와 메시지를 암호화하는 보조키의 수가 많이 요구된다. LCR은 라운드 조정알고리즘에 의해 제거될 멤버들간의 해밍거리가 임계치 이하인 멤버만 제거할 수 있으므로 Chang의 최악의 메시지 복잡도보다 낮은 복잡도를 가질 수 있으며 메시지를 암호화하는데 필요한 키의 수가 적으므로 Chang보다 적은 키 연산 비용이 소요된다. <표 3>에서 복합키를 생성하기 위한 연산비용 항목은 LCR과 Chang모델간의 비교만 가능하나 Wong모델과의 비교는 불가능하다.

새로운 키를 생성하는 것은 많은 연산을 요구하므로 rekey하기 위하여 보내는 메시지의 수를 줄이거나 복합키를 생성하기 위한 연산을 줄이는 것은 키 관리구조를 평가하는 중요한 척도이다. 여러 개의 멤버들을 제거할 때 라운드 조정 알고리즘을 수행하면 rekey를 위하여 메시지와 복합키 연산의 비용을 줄일 수 있는 이점이 있다.

<표 2> 한 개의 멤버 제거시 모델간의 비교

	메시지 복잡도	제어기의 키 복잡도	멤버의 키 복잡도
Wong	$O(2\log N)$	$O(2N)$	$O(\log N)$
Chang	$O(\log N)$	$O(2\log N)$	$O(\log N)$
LCR	$O(\log N)$	$O(2\log N)$	$O(\log N)$

<표 3> 여러 개의 멤버 제거시 모델간의 비교

	메시지 복잡도	제어기의 키 복잡도	멤버의 키 복잡도	복합키 연산비용
Wong	$O(N)$	$O(2N)$	$O(\log N)$	N/A
Chang	$O(N/2)$	$O(2\log N)$	$O(\log N)$	> LCR
LCR	< $O(N/2)$	$O(2\log N)$	$O(\log N)$	<Chang

4. 결 론

그룹 탈퇴멤버들의 UID를 체크하여 동시에 제거될 멤버들의 라운드를 조정하는 알고리즘을 사용하여 그룹키를 암호화하는데 사용할 키의 수를 줄일 수 있었다.

제거될 멤버들간의 해밍거리가 클수록 많은 메시지의 수와 메시지를 암호화하는 보조키의 수가 많이 요구된다. LCR는 라운드 조정알고리즘에 의해 제거될 멤버들간의 해밍거리가 임계치 이하인 멤버만 제거할 수 있으므로 Chang의 최악의 메시지 복잡도보다 좋은 복잡도를 가질 수 있으며 메시지를 암호화하는데 필요한 보조키의 수가 적으므로 Chang보다 적은 복합키 연산 비용이 소요된다.

LCR모델과 기존의 좋은 확장성을 보여준 모델인 Wong, Chang의 모델을 비교분석함으로서 LCR의 성능이 우수함을 보였다. 3개의 모델간에 메시지복잡도, 제어기의 키 복잡도, 멤버의 키 복잡도, 복합키 연산비용에 대하여 비교하였다.

LCR모델의 멤버의 키 복잡도는 Wong과 Chang모델과 모두 같으며, 제어기가 관리하는 키의 복잡도는 Wong모델보다 낮았다. 또한 LCR모델의 경우 1개의 멤버를 제거할 때 분배하는 메시지의 복잡도는 Wong모델보다 낮았으며, 여러 개의 멤버를 제거할 때 최악의 메시지 복잡도는 Wong과 Chang모델보다 낮았다.

LCR, Chang모델은 최소한의 보조키를 사용하여 그룹키를 관리하지만 그룹을 탈퇴하는 멤버들끼리 공모공격(collusion attack)이 발생할 수 있다. 공모공격은 그룹을 탈퇴하는 멤버의 집합이 공모하여 그들이 가지고 있는 키를 조합하면 새로운 키를 불법으로 획득하는 것이다. 향후에는

공모공격을 차단하는 연구가 필요하다.

참 고 문 헌

[1] B. Cain, S. Deering, I. Kouvelas, B. Fenner, A. Thyagarajan(2002). Internet Group Management Protocol, version 3. RFC 2236.

[2] H. Harney, C. Muckenhim(1997). Group Key Management Protocol(GKMP) Architecture. RFC 2093.

[3] H. Harney, C. Muckenhim(1997). Group Key Management Protocol(GKMP) Architecture. RFC 2094.

[4] T.Hardjono, G. Tsudik(2000). IP Multicast Security: Issues and Directions. Annales De Telecom, France.

[5] A. Ballardie(1996). Scalable Multicast Key Distribution. RFC 1949.

[6] D. Wallner, E. Harder, R. Agee(1999). Key Management for Multicast: Issues and Architectures. RFC 2627.

[7] 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.

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

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

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

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

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

[13] M. Steiner, G. Tsudik, M. Waidner(2000). Key Agreement in Dynamic Peer Groups. IEEE Tr. on Parallel and Distributed Systems.

[14] 서진원, 정종인(2003). 주기적인 Rekey를 사용한 멀티캐스트 키관리구조. 한국통신학회 하계학술대회.

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

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
1999~현재 멀티미디어기술사
관심분야: 정보보안, 병렬처리구조, 네트워크
E-Mail: jichung@kongju.ac.kr