Journal of The Korea Society of Computer and Information
Vol. 21 No. 9, pp. 73-78, September 2016

www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2016.21.9.073

Improvements of a Group key Management based on (2,2) Secret Sharing

Seunglim Yong*

Abstract

In 2014, Wuu *et al.* proposed a group key management scheme based on (2,2) secret sharing. They asserted that their scheme satisfies security requirements and mutual authentication. But this paper pointed out that their scheme does not satisfy mutual authentication and impersonating attack. In this paper, we describe the reasons and processes that a malicious group member can impersonate the Group Key Distributor. To fill the gaps, we discuss the problems, and propose an improved protocol.

▸ Keyword : Group key management, Secret sharing, Hash chain

I. Introduction

멀티캐스트는 일대다 혹은 다대다 통신 형태로 구성되는 그룹 통신에 적합한 프로토콜로서 영상회의 등의 그룹 기반의 응용에 많이 활용된다[1]. 멀티캐스트 환경에서는 하나의 송신자나 여러 송신자로부터 메시지 하나를 전송하여 멀티캐스트 그룹의 모든 멤버가 데이터를 효율적으로 수신할 수 있게 해준다. 그러나 멀티캐스트는 특정 그룹에 대한 제한적인 통신을 허락하지 않는다. 그룹 통신의 성공적인 활용을 위해서는 합법적인 권한을 부여받은 사용자들에게만 서비스 접근이 이루어질 수 있어야 하며 이는 암호화 통신으로 해결 가능하다. 그룹키를 이용한 암호화 통신은 멀티캐스트 환경에서 보호된 통신을 위하여 그룹키를 활용하여 송수신하는 메시지에 대하여 암복호화를 수행함으로 보호된 통신을 가능하게 한다. 이때 멀티캐스트 그룹키는 멤버들의 그룹 생성시 키 생성이 가능하여야 하며 가입과 탈퇴 등으로 인한 그룹 구성원 변경시 그룹키 재분배가 가능하여야 한다.

[2]에서 Wuu는 (2,2) 비밀분산법을 활용한 그룹키 관리 프로토콜을 제안하였다. 비밀분산법을 활용한 그룹키 관리 연구는 Harn과 Lin의 연구, Liu와 chen, Cao 그리고 Jian의 연구, Naranjo 등의 다양한 연구가 있다[3,4,5]. Wuu는 Shamir의 (2,2) 비밀분산법을 활용하여 그룹키 분배자가 제공한 키 생성 관련 메시지를 통하여 멤버 각자가 그 세션에 맞는 그룹키를 생성하여 이용하도록 하고 있다. 또한 키 갱신시에 하나의 메시지만을 통하여 효율적으로 키 갱신을 수행할 수 있으며 순방향 안전성, 역방향 안전성, 상호 인증, 가장 공격 등에 모두 안전하다고 분석하였다. 그러나 Wuu의 기법에서 각 멤버는 키 갱신 메시지를 GKD(Group Key Distributor)가 생성하여 송신하였음을 확인할 수 없기 때문에 그룹에 포함된 내부자에 의한 그룹키 갱신 공격이 가능하다는 문제가 발생한다.

본 논문에서는 Wuu의 기법에서 그룹 내의 정당한 사용자가 GKD를 가장하여 그룹키 갱신을 시도할 수 있음을 보인다. 그리고 이러한 시도를 막을 수 있는 새로운 방법을 제안한다. 본 논문에서 제안하는 기법에서는 해쉬 체인 방식을 이용하여 그룹의 각 멤버는 키를 생성한 주체가 GKD라는 것을 확인할 수 있도록 한다.

본 논문의 구성은 다음과 같다. 2장은 관련연구 부분으로 그룹키와 관련된 연구들에 대하여 간략히 알아본다. 3장에서는 Wuu의 (2,2) 비밀분산 기반의 그룹키 관리 기법을 살펴보고 Wuu의 기법의 공격 방법에 대하여 소개한다. 4장에서 새로운 키 관리기법을 제안하고 안전성을 평가한다. 마지막으로 5장에서 결론을 맺는다.

• First Author: Seunglim Yong, Corresponding Author: Seunglim Yong
*Seunglim Yong(slyong@inhatc.ac.kr), Dept. of Computer Systems and engineering, Inha Technical College
• Received: 2016. 05. 04, Revised: 2016. 07. 05, Accepted: 2016. 08. 31.
• This work was supported by INHA TECHNICAL COLLEGE Research Grant.

II. Preliminaries

1. Related works

1.1 The group key management scheme

그룹키 관리 기법은 중앙집중형, 분산형, 중앙분산형 등 다양한 응용환경에 맞도록 다양한 방향으로 연구되어 왔다.

그룹키를 생성하는 간단한 방법 중에 하나는 키 분배 센터라고 하는 특정한 서버가 그룹키를 관리하는 것이다[5]. 중앙집중형 그룹키 관리 기법(Centralized Group Key Management Scheme)은 중앙의 키 분배 센터가 모든 그룹 멤버를 위하여 키 계층을 유지하고 모든 멤버들의 키를 관리하게 된다[6,7].

분산형 그룹키 관리 기법(Distributed Group Key Management Scheme)은 그룹의 모든 멤버들이 자신의 몫(share)를 제공하여 그룹키 생성에 참여하거나 하나의 멤버가 그룹키를 생성하고 다른 멤버들에게 분배되는 기법이다. 그룹 Diffie-Hellman 키 교환 프로토콜, 논리적 키 계층 구조 등의 연구가 있다[8,9].

중앙분산형 그룹키 관리 기법(Decentralized Group Key Management)은 하나의 그룹 관리자가 전체의 그룹을 관리하지 않고 하나의 그룹을 다수의 독립적인 하위 그룹으로 나눈다. 그리고 멤버의 변화가 생길 때 해당 그룹의 그룹키 갱신만 수행토록 하는 방식이다[10,11] 하나의 그룹을 다수의 독립적인 하위 그룹으로 나누어 그룹키 갱신을 멤버의 변화가 일어난 해당 하위 그룹 안에서만 독립적으로 이루어질 수 있도록 한다.

1.2 Requirements of the group key management

안전한 그룹키 관리를 위해서는 먼저 그룹키의 유효시간을 정의하고 키를 갱신시에는 순방향 안전성과 역방향 안전성을 만족해야 한다. 순방향 안전성(Forward Secrecy)은 새로 가입한 멤버가 이전의 멀티캐스트 데이터에 접근 권한을 제한하는 것이고 역방향 안전성(Backward Secrecy)은 기존 멤버의 탈퇴 후 멀티캐스트 데이터에 접근 권한을 제한하는 것이다.

멤버가 그룹에 포함될 때에는 멤버에 대한 접근제어를 위해 그룹에 가입 시 인증이 이루어져야 한다. 또한 키를 관리하는 관리자와 멤버들 사이에는 유니캐스트 채널을 통한 통신 이외의 방법으로 그룹키의 재분배가 가능해야 한다.

III. Group Key Management of Wuu's Scheme

1. Wuu's group key management scheme

Wuu의 논문에서는 (2,2) 비밀분산법을 활용하여 그룹키를 관리한다[2].

신뢰할 수 있는 GKD는 각각의 등록된 사용자 i와 비밀리에 미리 공유된 키 K_i를 관리하며, 각각의 사용자는 키 재설정 메시지를 받으면 각자가 그룹키를 생성하도록 한다. 그룹키 GK를 필요로 하는 그룹의 멤버가 t명이라고 가정할 때, 먼저 GKD는 그룹키 t개의 1차다항식 $f_i = a_i(x) + GK$ $(i = 1,2,...,t)$를 생성한다. 각 $f_i(x)$는 두 개의 점 $(0, GK)$, $(K_i, H(K_i \| T))$를 지나는 직선이 되도록 생성한다. 다항식의 계수 a_i는 $a_i = K_i^{-1}(H(K_i \| T) - GK)$이 되어야 한다. GKD는 임의 난수 R을 선택하고, T, R과 $f_i(R)$을 모든 그룹 멤버에게 멀티캐스트로 보낸다. 그룹 멤버는 멀티캐스트로 받은 R, $f_i(R)$, T값과 미리 공유한 키 K_i를 이용하여 $(R, f_i(R))$과 $(K_i, H(K_i \| T))$ 두 점을 지나는 1차다항식을 계산해내고 $f_i(0)$을 계산하여 그룹키 GK를 획득한다.

Wuu의 그룹키 관리 프로토콜을 단계별로 간단히 살펴보면 다음과 같다.

1.1 System initialization phase

GKD는 소수 p, 일방향 해쉬함수 $H()$, 암호 알고리즘 E를 공개하고 사용자는 그룹에 포함되기 전에 GKD에 자신의 실제 ID를 등록하고 사전공유키 k_i를 안전한 채널을 통하여 받는다.

1.2 Group creation

새로운 그룹을 생성하려는 멤버를 '그룹 생성자(Group Initiator)'라고 하며 간단히 ID_1이라 가정한다. 생성자는 GKD에게 타임스탬프 T, 그룹 ID gid, 그룹 멤버 리스트 $\{ID_1, ID_2, ..., ID_t\}$, 인증 코드 $Auth_1 = H(T \| gid \| ID_1 \| ID_2 \| ... \| ID_t \| K_1)$를 그룹 생성 메시지 요청과 함께 유니캐스트한다.

GKD는 요청 메시지를 확인하고 다음의 절차를 통하여 그룹키를 생성한다.

step 1. 타임스탬프 T와 $Auth_1$의 유효성을 확인한다.

step 2. 그룹키 $GK \neq H(K_i \| T)$와 임의난수 $R \neq K_i$ 를 생성한다.

step 3. 각 멤버를 위하여 $(0, GK)$와 $(K_i, H(K_i \| T))$의 두 점을 지나는 1차다항식 $f_i(x) = K^{-1}(H(K_i \| T) - GK)x + GK \bmod p$을 생성한다. 그리고나서 $y_i = f_i(R)$을 계산한다.

step 4. $Auth_G = H(GK \| T \| gid \| R)$을 계산하고 그룹 멤버에게 $\{T, gid, R, y_1, ..., y_t, Auth_G\}$를 멀티캐스트한다.

step 5. 각 멤버는 $(K_i, H(K_i \| T))$를 스스로 계산하고 받은 메시지에 있는 (R, y_i)를 이용하여 그룹키 $GK = f_i(0) = f_i((K_i \cdot y_i) - (R \cdot H(K_i \| T))) \cdot (K_i - R)^{-1} \bmod p$를 계산해낸다. 계산된 GK를 이용하여 $Auth_G$가 $H(GK \| T \| gid \| R)$과 같은지 확인한다.

1.3 Member join

멤버 j가 gid그룹에 조인하기 원할 때 조인 요청 메시지와 함께 타임스탬프 T, 그룹 ID gid, 인증코드 $Auth_j = H(T \| gid \| K_j)$를 GKD에 보낸다.

GKD는 그룹 생성 단계 step 1, step 2를 수행하고 $(0, GK')$과 $(K_j, H(K_j \| T))$를 지나는 새로운 1차다항식을 생성한다.

GKD는 $Auth_G = H(GK' \| T \| gid \| R)$을 생성하여 메시지 $\{T, gid, R, y_j, E_{GK}[GK'], Auth_G\}$를 새로 조인하는 멤버 j와 기존 멤버에게 보내준다.

멤버 j는 GK'을 계산해내면 되고 기존 멤버는 자신들이 소유한 GK 키를 이용하여 메시지를 복호화하면 GK'을 $D_{GK}[E_{GK}[GK']]$를 통하여 얻어낼 수 있다.

1.4 Member leave

멤버가 떠나면 새로운 멤버를 구성하는 방식으로 처음부터 다시 생성한다.

2. The problem with Wuu's scheme

Wuu가 제안한 기법은 그룹의 멤버와 GKD 사이에 그룹키를 생성하고 새 멤버가 그룹에 조인할 때 상호 인증이 이루어지며 안전하다고 분석하였다. 그러나 주장하는 바와 달리, 멤버가 조인되어 키 갱신이 이루어져야 할 때 멤버들과 GKD는 상호인증이 이루어지지 않는다. 즉, Wuu의 기법에서는 그룹의 한 멤버가 GKD를 가장하여 기존 그룹의 멤버들에게 키 갱신 메시지를 보낼 수 있으며, 기존 그룹의 멤버들은 수신한 메시지를 통하여 키를 갱신하는 문제가 발생하게 된다.

먼저 Wuu가 제안한 방식에서 상호 인증이 이루어지는 방식은 다음과 같다. 그룹의 생성 단계에서는 멤버들과 GKD가 사전 공유한 K_i의 값, 그리고 $Auth_G$의 값을 통하여 상대방을 인증할 수 있다. 그룹의 멤버 조인 단계에서는 조인하는 멤버 j와 GKD는 사전에 공유한 K_j값과 GKD가 생성하여 멀티캐스트한 메시지 $Auth_G$를 통하여 상호 인증이 가능하다. 그룹의 기존 멤버들과 GKD는 이전 그룹키 GK를 통하여 상호 인증이 가능하다. 멤버 j가 조인 시에 멤버 j는 $Auth_G$를 통해서 수신 메시지가 GKD의 메시지임을 확인할 수 있다. 멤버 j의 비밀 공유값 K_j는 GKD만 알고 있으며 그 값을 알고 있는 사람만이 정당한 GK'을 생성할 수 있기 때문이다.

그러나, 주장한 바와 달리 기존 멤버는 $Auth_G$ 또는 수신 메시지 $\{T, gid, R, y_j, E_{GK}[GK'], Auth_G\}$의 내용만으로는 메시지의 생성자가 GKD임을 확인할 방법이 없다. GK를 아는 악의적인 기존 멤버가 있다고 가정하자. 이 멤버는 가짜로 $Auth'_G = H(GK' \| T \| gid \| R)$, R, y_j값을 만들 수 있다. 악의적인 기존 멤버가 GKD가 생성한 메시지인 것으로 가장하여 메시지 $\{T, gid, R, y_j, E_{GK}[GK'], Auth'_G\}$를 다른 그룹 멤버에게 멀티캐스트 한다면, 기존의 멤버들은 가지고 있는 GK로 $E_{GK}[GK']$를 복호화하고 GK'으로 키를 갱신하게 된다. 기존의 다른 멤버들은 $Auth'_G$값을 통하여 GKD의 메시지 여부를 확인할 수 없기 때문에 악의적인 내부 멤버 중 한 명에 의하여 키 갱신 메시지가 정당한 메시지로 가장되어 전송될 수 있다. 또한 다른 멤버들은 그 메시지를 통해 키를 갱신하게 됨으로써 정상적인 그룹통신이 어려워질 수 있다. 따라서 상호 인증이 성립되지 않으며 악의적인 사용자는 경우에 따라 GKD를 가장할 수도 있게 된다.

악의적인 멤버	다른 그룹 멤버
임의의 T, R, y_j생성 $Auth'_G = H(GK' \| T \| gid \| R)$	
$T, gid, R, y_j, E_{GK}[GK'], Auth'_G$ (multicast) →	

Fig.1 Impersonating attack

이러한 문제점을 해결하기 위하여 멀티캐스트된 키 갱신 메시지가 GKD로부터 생성된 메시지임을 모든 멤버가 확인할 수 있는 인증 방법이 추가되어야 한다.

IV. The Proposed Scheme

1. Proposed group key management scheme

본 논문에서는 멀티캐스트된 메시지가 정당한 GKD의 메시지인지를 확인하기 위하여 해쉬 체인 기법을 활용하고자 한다.

해쉬 체인 기법은 해쉬값을 생성하는 순서의 역순으로 해쉬값을 사용하는 방식이다[12]. 일방향 해쉬 함수 H는 주어진 값 x에 대하여 $y = H(x)$를 계산하기는 쉬우나, y 값이 주어졌을 때 $y = H(x)$를 만족하는 x를 찾아내는 것이 계산상으로 불가능한 특징을 가진다. 해쉬 체인이란 해쉬값 $\{x_n, ..., x_j, ..., x_0\}$을 종자값 x_n을 이용하여 식(1)과 같이 만들어낸다.

$$x_{i-1} = H(x_i),\ (i = 1...n) \tag{1}$$

즉, $x_1 = H(x_2) = ... = H^{n-2}(x_{n-1}) = H^{n-1}(x_n)$의 관계가 된다. 해쉬 체인은 해쉬함수의 특성으로 인하여 주어진 x_i에 대하여 $x_j (j < i)$값은 계산상 불가능하지만 $x_j (j > i)$값은 쉽게 계산해낼 수 있는 특징이 있다.

제안하는 기법에서는 해쉬 체인을 GKD의 생성메시지 확인에 적용하여 GKD를 가장하는 공격으로부터의 안전성을 확보하고자 한다. GKD는 인증메시지 $Auth_G$를 생성할 때 생성된 해쉬 체인을 역순으로 사용함으로써 GKD임을 인증하게 된다. 이때 해쉬값은 $Auth_G$를 생성할 때마다 변경하여 사용함으로써 재사용공격을 막는다. 본 논문에서 제안하는 개선된 프로토콜의 상세 내용은 다음과 같다.

1.1. System initialization phase

시스템 초기화단계에서 GKD는 소수 p, 일방향 해쉬함수 $H()$, 암호 알고리즘 E를 공개한다.

GKD는 해쉬 체인의 확인값 h_0^G를 생성하기 위하여 임의의 해쉬 체인의 종자값 h_n을 선택한 후 종자값에 해쉬 체인을 n번 수행하여 생성한다. 즉, $h_0^G = H(h_1) = ... = H^{n-1}(h_{n-1}) = H^n(h_n)$을 계산한다. 사용자는 그룹에 포함되기 전에 GKD에 자신의 실제 ID를 등록하고 사전공유키 K_i, 해쉬 체인의 확인값 h_0^G을 안전한 채널을 통하여 받는다. GKD의 해쉬 체인 확인값은 비밀값은 아니나 GKD가 생성한 값임을 확인하기 위하여 다른 서명 등을 추가하지 않고 안전한 채널을 통하여 받기로 한다.

1.2. Group creation

새로운 그룹을 생성하는 기본 방법과 step 1, step 2의 생성절차는 Wuu의 방식과 동일하다. 그러나 GKD가 멀티캐스트 메시지를 작성할 때 GKD가 작성한 메시지인지를 모든 멤버가 확인할 수 있도록 $Auth_G$의 값을 달리 생성한다. 그룹 생성자(Group Initiator)는 편의상 ID_1으로 가정한다.

step 1. 그룹 생성자 ID_1은 GKD에게 그룹 생성 메시지 요청과 함께 타임스탬프 T, 그룹 ID gid, 그룹 멤버리스트 $\{ID_1, ID_2, ..., ID_t\}$와 함께 인증코드 $Auth_1 = H(T \| gid \| ID_1 \| ID_2 \| ... \| ID_t | K_1)$를 생성하여 보낸다.

step 2. GKD는 타임스탬프 T와 $Auth_1$의 유효성을 확인하고 $GK \neq H(K_i \| T)$인 그룹키 $GK \in Z_p^*$와 $R \neq K_i (i = 1,2,...,t)$인 임의난수 $R \in Z_p^*$을 생성한다.

step 3. GKD는 각 멤버를 위하여 $(K_i, H(K_i \| T)), (0, GK)$ 두 점을 지나는 1차다항식 $f_i(x)$를 생성하고, $y_i = f_i(R)$과 인증코드 $Auth_G = H(GK \| T \| gid \| R \| h_k \| k)$을 계산한다. 그리고 나서 그룹 멤버에게 $\{T, gid, k, h_k, R, y_1, y_2, ..., y_t, Auth_G\}$를 멀티캐스트 한다. 그룹 생성 단계의 k값은 1로 한다.

step 4. 각 멤버는 수신한 메시지 중 $\{k, h_k\}$값을 이용하여 $h_0^G = H^k(h_k)$을 계산하여 GKD로부터 온 메시지가 맞는지 확인한다. 그리고 GKD로부터 얻은 메시지를 통하여 그룹키 $GK = f_i(0) = f_i((K_i \cdot y_i) - (R \cdot H(K_i \| T))) \cdot (K_i - R)^{-1} \bmod p$를 만들어낸다. 만들어낸 GK를 이용하여 $Auth_G$가 $H(GK \| h_k \| T \| gid \| R \| k)$과 같은지 확인한다.

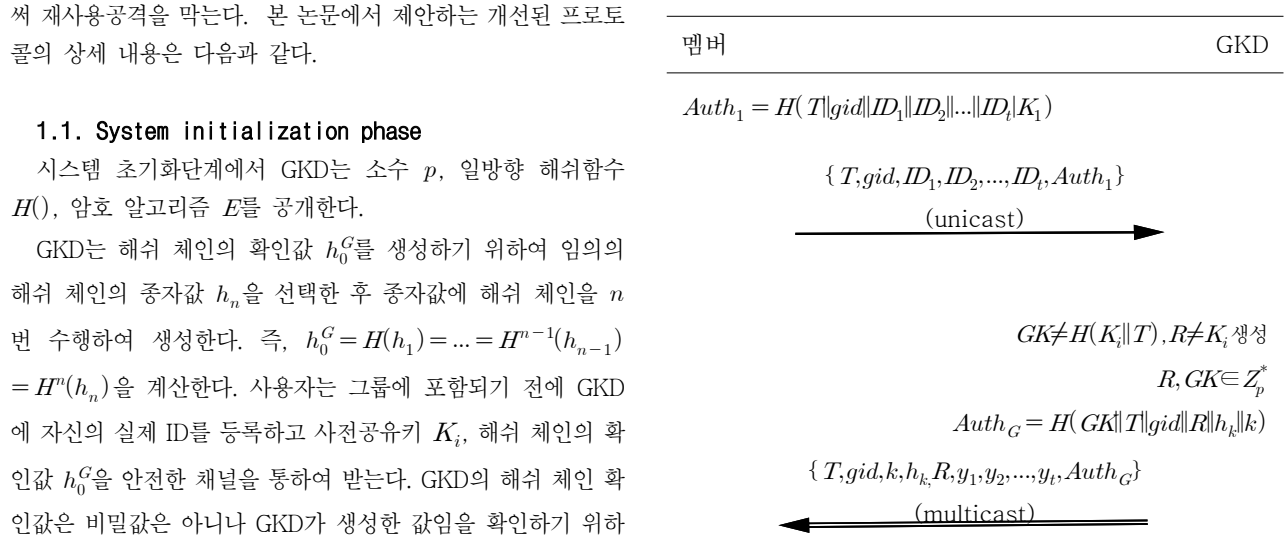

Fig 2. Group creation

1.3. Member join

멤버 j가 gid그룹에 조인하기 원할 때 다음의 절차를 따른다.

step 1. 멤버 j는 GKD에게 조인 요청 메시지와 함께 그룹 id gid, 타임스탬프 T', 인증코드 $Auth_j = H(T' \| gid \| ID_j \| K_j)$를 GKD에 보낸다.

step 2. GKD는 타임스탬프 T'와 $Auth_j$의 유효성을 확인하고 $GK' \neq H(K_j \| T')$인 그룹키 $GK' \in Z_p^*$와 $R' \neq K_j$인 임의난수 $R' \in Z_p^*$를 생성한다.

step 3. GKD는 $(0, GK')$와 $(K_j, H(K_j \| T'))$의 두 점을 지나는 1차다항식 $f_j(x) = K^{-1}(H(K_j \| T') - GK')x + GK' \bmod p$을 생성한다. 멤버 j를 위한 $y_j = f_j(R')$ 값과 GKD 메시지임을 확인할 수 있는 $Auth_G = H(GK' \| T' \| gid \| R' \| h_{k'} \| k')$을 계산하고 그룹 멤버에게 $\{T', gid, k', h_{k'}, R, y_j, E_{GK}[GK'], Auth_G\}$를 멀티캐스트 한다. 그룹 멤버 조인시 $k' = k+1$이며 그룹 키를 갱신할 때마다 $k' = k' + 1$하여 이용하도록 한다. 또한 각 멤버는 k'의 값을 저장하도록 하여 재사용을 방지하도록 한다.

step 4. 각 멤버는 수신한 메시지 중 $\{k', h_{k'}\}$값을 이용하여 해쉬함수를 k'번 적용한 값 $H^{k'}(h_{k'})$과 h_0^G값이 동일한지 확인

함으로써 GKD로부터 온 메시지가 맞는지 확인한다.

step 5. 멤버 j는 수신 메시지를 통하여 그룹키 $GK' = f_j(0) = f_j((K_j \cdot y_j) - (R' \cdot H(K_j \| T'))) \cdot (K_j - R')^{-1} \bmod p$를 만들어낸다. 모든 멤버들은 $E_{GK}[GK']$값을 복호화하여 GK'를 계산하고 $Auth_G$가 $H(GK' \| T' \| gid \| R' \| h_{k'} \| k')$과 같은지 확인하여 갱신된 그룹키를 검증한다.

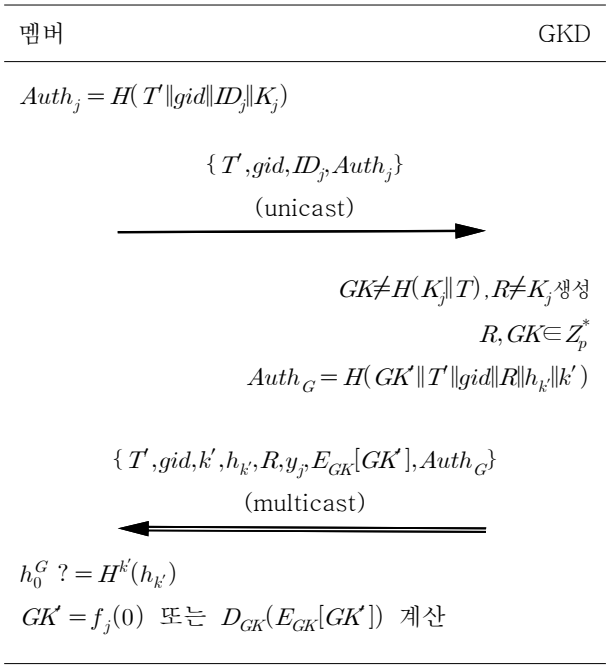

Fig. 3 Member Join

1.4. Member leave

멤버가 떠나면 새로운 멤버를 구성하는 방식으로 처음부터 다시 수행한다.

2. Security analysis

제안한 기법에서는 GKD의 인증을 위한 수단으로 해쉬 체인을 이용한다. 해쉬 체인은 해쉬 함수의 일방향성으로 인하여 종자값을 알고 있는 GKD 이외에는 정당한 해쉬 체인값을 생성해 낼 수 없기 때문에 GKD 인증이 가능하다. 또한 해쉬 체인은 역순으로 사용하게 되며 매번 인증을 수행해야 하는 경우마다 역순의 해쉬 체인값으로 인증을 하기 때문에 GKD를 가장한 공격을 막을 수 있다.

2.1 Mutual authentication between GKD and Group member i

제안한 기법은 Wuu의 기법과 마찬가지로 각 멤버와 GKD 사이에 사전 공유한 키 K_i를 가지고 상호 인증을 수행하게 된다. 그룹 생성 단계에서 GKD는 그룹 생성자를 $Auth_1$ 값으로 확인할 수 있으며, 각 멤버는 GKD가 생성한 $(K_i, H(K_i \| T'))$와 (R, y_i), 그리고 h_0^G값의 통하여 확인할 수 있다.

멤버 조인 단계에서 GKD는 $Auth_j$값을 통하여 새로 조인하는 멤버 j를 인증하고 멤버 j는 GKD가 생성한 $(K_j, H(K_j \| T'))$와 (R, y_j), 그리고 h_0^G값의 통하여 GKD의 메시지임을 확인한다. 기존의 그룹 멤버들은 $Auth_G$ 값과 GKD의 해쉬 체인 h_0^G값을 통하여 키 갱신 메시지가 GKD로부터 전송된 것임을 확인할 수 있다.

2.2. Forward secrecy

그룹 멤버가 탈퇴할 때, 제안한 기법은 Wuu의 기법과 마찬가지로 새로운 그룹 키를 생성하고 분배하는 단계를 수행하게 된다. 따라서 탈퇴한 멤버는 기존 그룹의 갱신된 새로운 키의 어떠한 정보도 알 수 없게 되며 따라서 순방향 안전성을 만족한다.

2.3 Backward secrecy

새 멤버가 조인할 때 GKD는 임의의 새로운 그룹키를 생성하게 된다. 따라서 이전의 그룹키를 통하여 새로운 그룹 키를 유추해내는 것은 불가능하다.

2.4 Replay attack

제안한 기법은 타임스탬프 T를 이용하기 때문에 재전송 공격을 수행할 수 없다. 또한 모든 멤버는 k값을 저장하고 확인하기 때문에 해쉬 체인의 값을 재전송하여 사용하지 못한다.

2.5 Impersonating attack

그룹 외부의 공격자가 그룹 멤버 i로 가장하려면 GKD와 사전 공유한 공유키 K_i를 알아내야 가능하며, GKD로 가장하고자 할 때는 멤버들과의 사전 공유키를 전부 알고 있어야 하기 때문에 가장 공격으로부터 안전할 수 있다.

만약 기존의 그룹 멤버가 GKD로 가장하여 키 갱신을 시도하고자 한다면 그룹 멤버는 h_{k+1}값을 알고 있어야 정당한 $Auth_G$값을 계산할 수 있다. 그러나 기존 멤버가 알고 있는 값은 h_0^G와 h_k값이 전부이며, 해쉬 체인의 성질에 따라 $h_k = H(h_{k+1})$의 값은 쉽게 구할 수 있지만 h_k값을 통하여 h_{k+1}의 값은 알아낼 수 없기 때문에 정당한 $Auth_G$값을 생성할 수 없다. 따라서 기존 멤버가 GKD를 가장하여 그룹키 갱신을 시도할 수 없다.

V. Conclusions

본 논문에서는 기존의 Wuu의 기법에서 그룹 멤버가 조인하는 단계에서 상호 인증과 GKD 가장 공격이 가능하다는 것을

보였다. 또한 이를 해결하기 위하여 해쉬 체인 기법을 이용하여 순방향 안전성과 역방향 안전성을 만족하면서도 다른 PKI기법 이용 없이 GKD를 확인할 수 있는 방법을 제안하였다.

다만 해쉬 체인을 이용할 경우 GKD의 메시지 전송 횟수가 제한이 생기는 문제가 발생할 수 있어 제한된 수의 키 갱신에 사용되어야 하는 문제가 발생된다. 이러한 문제는 멤버가 탈퇴할 때 새로이 키 생성 단계부터 실시하게 되므로 이때 h'^{G}_{0}값을 새로 배포하는 것도 해결의 하나의 방법이 될 수 있다.

REFERENCES

[1] Ng WHD, Howarth M, Sun Z, Cruickshank H. "Dynamic balanced key tree management for secure multicast communications," IEEE Transactions on Computers, Vol. 56, No. 5, pp. 590-605, 2007.

[2] Lih-Chyau Wuu, Chi-Hsiang Hung, Wen-Chung Kuo, “Group Key Management based on (2,2) Secret Sharing”, KSII Transactions on Internet and Information Systems, Vol.8 , No. 3, pp. 1144-1156, 2014.

[3] L. Harn, C. Lin, “Authenticated group key transfer protocol based on secret sharing,” IEEE. Trans. Computers, Vol. 59, No. 6, pp. 842-846, 2010.

[4] Y. Liu, C. Cheng, J. Cao, T. Jiang, “An improved authenticated group key transfer protocol based on secret sharing,” IEEE Trans. Computers, Vol. 62, No. 11, pp. 2335-2336, 2013.

[5] Harney H, Muckenhirn C, Rivers T, "Group key management protocol(GKMP) architecture," RFC 2094, IETF 1997.

[6] C. Blundo, Luiz A. Frota Mattos, and D.R. Stinson, “ Generalized Beimel-Chor Schemes for Broadcast Encryption and Interactive Key Distribution”, Thoretical Computer Science V. 200, No. 1-2, pp. 313-334, 1998.

[7] Ng WHD, Cruickshank H, Sun Z. "Scalable balanced batch rekeying for secure group communication," Computers and Security, Vol. 25, No. 4, pp. 265-273, 2006.

[8] M. Stener, G. Tsydik, M. Waidner, “Diffie-Hellman Key Distribution Extended to Group Communication”, In Proceedings of ACM CCS, pp. 31-37,1996

[9] O. Rodeh, K.P. Briman, D. Dolev, “Optimized Group Rekey for Group Communication Systems”, In Proceeding of Network and Distributed Systems Security Symposium, 2000.

[10] Junbeom Hur, Hyunsoo Yoon, “Decentralized Group Key Management for Untrusted Dynamic Networks”, Journal of Computing Science and Engineering, Vol. 36, No. 4, pp. 263-275, 2009.

[11] Kwak DW, Kim J., "A decentralized group key management scheme for the decentralized P2P environment," IEEE Communications Letters, Vol. 11, No. 6, pp. 555-557, 2007.

[12] Thomas Page, “The application of hash chains and hash structures to cryptography”, PhD thesis, Citeseer, 2009.

Authors

Seunglim Yong received the B.S., M.S. and Ph.D. degrees in Computer Science and Engineering from Ewha Womans University, Korea, in 1998, 2000 and 2006, respectively.

Dr. Yong joined the faculty of the Department of Computer Systems and Engineering at Inha Technical College, Korea, in 2008. He is currently a Associate Professor in the Department of Computer Systems and Engineering, Inha Technical College. She is interested in computer security, cryptography, and cryptographic protocols.