정보보호개론

제12장 다자간 키 확립 프로토콜

1. 개요

온라인 실시간 강의나 방송과 같은 응용은 그룹 통신(group communication)이 필요하다. 그룹 통신이란 한 메시지를 다중 수신자에게 전송하는 것을 말한다. 보통 효율적으로 그룹 통신을 하기 위해 IP 멀티캐스트(multicast) 기법을 사용한다. IP 멀티캐스트 기법은 가장 적은 네트워크 대역폭을 사용하여 동시에 여러 수신자에게 같은 메시지를 전달하여 주는 통신 기법이다. 멀티캐스트 기법을 사용하지 않고 그룹의 개별 멤버에게 유니캐스트를 하면 공통 통신 경로에 같은 메시지를 중복하여 여러 차례 전달하게 된다.

그룹 통신을 할 때 통신 내용을 보호하고 싶을 수 있다. 비밀 그룹 통신을 하기 위해서는 그룹 멤버들이 하나의 암호키를 공유해야 한다. 그룹키는 없고 그룹 멤버 쌍마다 별도 비밀키를 공유하고 있으면 같은 메시지를 비밀스럽게 전달해야 할 때 멀티캐스트를 활용할 수 없다. 하지만 그룹키를 공유하고 있으면 일반 메시지처럼 그룹키로 메시지를 암호화한 후에 결과 암호문을 멀티캐스트하여 전달할 수 있다. 그룹 멤버 간의 공통 비밀키를 확립하기 위해 사용하는 프로토콜을 다자간 키 확립 프로토콜이라 하며, 다른 말로 그룹키, 회의키 프로토콜이라 한다.

다자간 키 확립 프로토콜에서 그룹의 크기가 매우 클 수 있으므로 확장성이 매우 중요하다. 특히, 그룹의 멤버가 빈번하게 변할 수 있으면 이 변화에 대해 확장성 있게 대처할 수 있어야 한다. 예를 들어 하나의 멤버가 탈퇴하게 되면 이 멤버는 더는 그룹의 비밀 통신 내용을 볼 수 없어야 한다. 이를 위해서는 기존에 사용한 그룹키를 변경해야 하는데, 그룹키의 변경을 위해 소요되는 비용이 그룹 크기에 비례하지 않도록 확장성을 가지게 하는 것은 쉽지 않다.

다자간 키 확립 프로토콜도 키 확립 프로토콜의 한 종류이므로 키 확립 프로토콜의 요구사항을 모두 충족해야 하지만 현실적으로 충족하는 것이 힘들다. 예를 들어 2자 간에서는 서로 같은 키를 공유하였는지 키 확인을 하지만 다자 간에서는 서로서로 같은 키를 가졌는지 확인하는 것은 비용을 고려하였을 때 현실적이지 못하다. 키 최근성의 경우 2자 간에서는 상대방을 신뢰할 필요가 없는 난스 기법을 사용할 수 있지만, 다자 간에는 난스 기법을 사용할 수 없다.

1.1 그룹의 동적성

그룹 멤버가 변할 수 있는 환경의 경우 그룹 멤버의 변화에 필요한 보안 조치를 확장성 있게 할 수 있어야 한다. 그룹 멤버의 변화는 새 멤버의 가입과 기존 멤버의 탈퇴 두 가지 경우가 있으며, 응용에 따라 다음 요구사항이 충족되어야 한다.

- 전방향 안전성(forward secrecy): 그룹을 탈퇴한 멤버를 포함하여 이전 그룹키를 알고 있는 공격자는 새 그룹키를 알 수 없어야 한다.
- 후방향 안전성(backward secrecy): 그룹에 새롭게 가입한 멤버를 포함하여 현재 그룹키를 알고 있는 공격자는 이전 그룹키를 알 수 없어야 한다.

전방향, 후방향 안전성은 장기간 키의 노출 관련 안전성을 논할 때도 사용한 용어이지만 용어만 같을 뿐 내용이 다른 것이다. 이들 두 가지 요구사항 대신에 다음 요구사항을 사용하는 경우도 있다.

- 키 독립성(key independence): 몇 개의 그룹키를 알고 있는 공격자는 이 키들을 제외한 다른 그룹키들을 알 수 없어야 한다.

키 독립성은 전방향, 후방향 안전성을 포함한 개념이다. 응용에 따라 전방향, 후방향이 모두 필요할 수 있고, 전방향 안전성이 후방향보다 상대적으로 더 중요하기 때문에 전방향 안전성만 제공하는 경우도 있다.

다자간 키 확립 프로토콜에서 전방향, 후방향 안전성을 충족하기 위해 그룹 멤버에 변화가 있을 때마다 그룹키를 바꾸어야 한다. 이때 사용하는 프로토콜을 가입과 탈퇴 프로토콜이라 한다.

1.2 다자간 키 확립 프로토콜의 분류

다자간 키 확립 프로토콜은 키 분배 서버를 활용하는 방식에 따라 다음과 같이 분류할 수 있다.

- 중앙집중형(centralized): 단일 서버가 그룹키를 분배하는 방식을 말한다.
- 탈중앙형(decentralized): 전체 그룹이 여러 개의 작은 그룹으로 나누어 관리되는 방식으로 보통 각 소그룹마다 해당 그룹의 그룹키를 분배하는 별도 서버를 사용한다.
- 분산형(distributed): 그룹키 분배하는 서버를 전혀 사용하지 않는 방식을 말한다.

분산형의 경우 키 분배 서버를 사용하지 않지만, 그룹 멤버 관리(가입 승인 등)를 위한 서버는 필요할 수 있다.

이 분류는 서버 수를 기준으로 사용하고 있다. 하지만 중앙집중, 탈중앙, 분산의 비교는 간단하지 않다. 최근 탈중앙 암호화폐의 등장으로 탈중앙, 분산 용어가 부정확하게 사용되는 경향도 있다. 3가지 개념을 가지고 분류하기보다는 통제 주제에 따라 중앙집중과 탈중앙, 물리적 위치에 따라 중앙집중과 분산으로 나누는 것이 더 직관적이다.

다자간 키 확립 프로토콜은 상태 기반(stateful)과 비상태 기반(stateless)으로도 분류할 수 있다.

- 상태 기반: 모든 키 갱신 세션에 빠짐없이 참여할 경우에만 최신 그룹키를 계산할 수 있는 방식을 말한다.
- 비상태 기반: 현재 진행 중인 키 갱신 메시지와 초기 상태에 대한 정보만 있으면 최신 그룹키를 계산할 수 있는 방식을 말한다.

예를 들어 실시간 유료 방송의 경우 각 가입자의 수신 장치가 항상 켜져 있지 않기 때문에 꺼져 있을 때 교환된 메시지는 받을 수 없다. 따라서 이와 같은 환경에서는 절대적으로 비상태 기반 기법의 사용이 필요하다.

1.3 유니캐스트와 멀티캐스트

그룹키 프로토콜의 효율성을 분석하기 위해 필요한 유니캐스트의 수와 멀티캐스트 수를 종종 비교한다. 이때 n개의 서로 다른 작은 메시지를 n명에게 유니캐스트하는 대신에 이들을 결합하여 하나의 멀티캐스트로 보낼 수 있으므로 수의 비교는 정확한 비교가 아니라고 생각할 수 있다. 하지만 서로 다른 메시지를 결합하여 멀티캐스트하는 것은 어떤 긍정적 효과도 얻을 수 없다.

서로 다른 메시지이므로 결합을 통해 메시지 크기가 줄지 않으며, 모든 통신 경로에 결합된 크기의 메시지가 지나간다. 이 때문에 오히려 사용하는 대여폭은 늘어난다. 더욱이 수신된 메시지의 상당한 부분은 수신자에게는 불필요한 데이터이다. 따라서 멀티캐스트는 같은 데이터를 다수에게 보낼 때에만 의미가 있으며, 이와 같은 상황에서만 사용된다.

2. 중앙집중형

2.1 단순 접근 방법

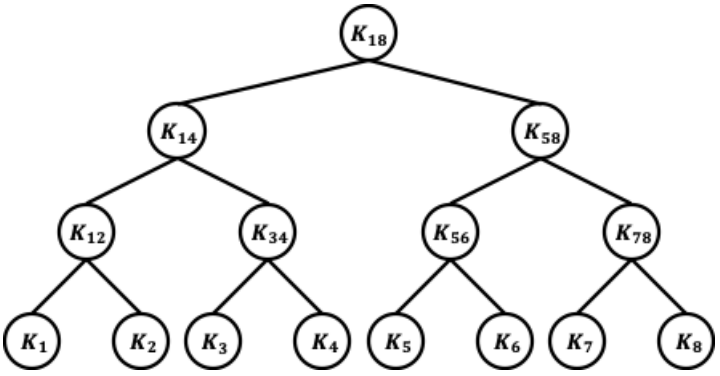
어떤 문제에 대한 해결책을 찾을 때 가장 단순한 방법부터 생각해보는 것이 필요하다. 가장 단순한 방법의 문제점을 통해 문제에 대한 더 올바른 이해를 할 수 있다.

처음으로 생각해 볼 해결책은 중앙 서버가 그룹의 각 멤버와 비밀키를 공유하고 그룹에서 사용할 그룹키를 각 멤버와 공유한 비밀키로 암호화하여 전달하는 방식이다. 이 방식은 그룹의 크기가 n일 때 그룹키를 확립하기 위해 n개의 암호문 생성이 필요하고, n개의 유니캐스트가 필요하다. 그룹에 새 멤버가 가입하거나 기존 멤버가 탈퇴할 때에도 전후방향 안전성을 제공하기 위해 같은 방법을 사용한다면 그룹의 크기에 비례한 비용이 필요하다. 따라서 확장성이 있는 방식은 아니다.

위 방법에서 새 멤버 A가 가입할 경우에는 아주 효과적으로 새 그룹키 K_{new}를 확립하는 방법이 있다. 기존 멤버들은 모두 이전 그룹키 K_{old}를 알고 있으므로 새 멤버에게는 해당 멤버와 중앙서버가 공유한 비밀키로 새 그룹키를 암호화하여 유니캐스트하고, 나머지 멤버에게는 이전 그룹키로 새 그룹키를 암호화하여 멀티캐스트하면 된다. 따라서 가입의 경우에는 항상 고정된 비용을 이용하여 새 그룹키를 확립할 수 있다. 실제 이보다 더 저렴한 비용으로 그룹키를 확립할 방법은 없다. 문제는 탈퇴의 경우에는 이와 유사한 방법으로 문제를 해결할 수 없다는 것이다.

지금의 설명에서 하나 기억해야 하는 것은 확장성의 초점이 개별 참여자의 비용이 아니라 서버 비용이라는 것이다. 서버가 생성해야 하는 암호문의 수, 전송해야 하는 메시지의 수나 형태를 확장성 있게 만들고자 하는 것이다. 물론 그룹키 프로토콜을 설계하면서 각 참여자의 비용도 확장성이 있어야 하지만 서버 비용이 확장성이 있으면 자동으로 각 참여자의 비용도 확장성이 있게 된다. 이 절에서 소개한 단순 접근 방법에서 각 참여자는 하나의 암호문만 수신하며, 서버와 공유한 장기간 키와 현재 그룹키만 유지하면 된다.

2.2 LKH



<그림 12.1> LKH 트리

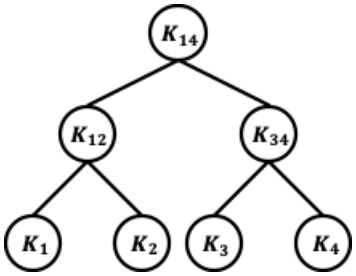
단순 접근 방법을 사용하면 가입은 상수 비용으로, 탈퇴는 그룹 수에 비례한 선형 비용으로 처리할 수 있다. 이를 개선하기 위해서는 탈퇴 비용의 개선이 필요하다. 따라서 선형 비용을 로그 비용이나 상수 비용으로 줄일 수

있는 방법을 고안해야 하는데, 가입 비용을 상수 비용으로 유지하면서 탈퇴 비용을 줄이는 것이 가능하지 않을 수 있다. 하지만 가입과 탈퇴를 모두 로그 비용에 할 수 있으면 최악의 비용이 줄어드는 것이므로 단순 접근 방법보다는 효과적인 방법이 된다.

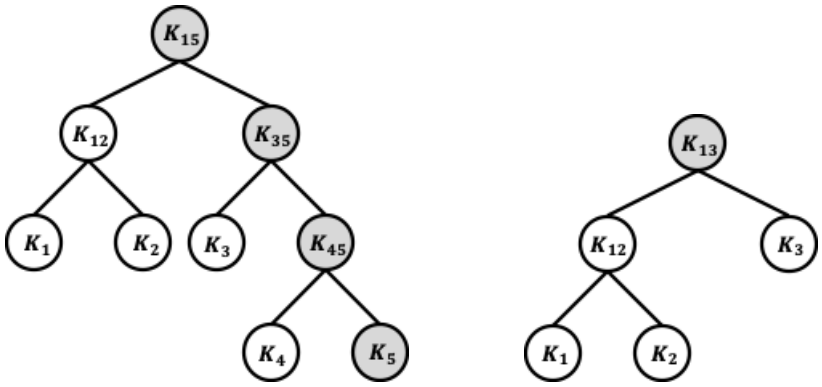
기본적으로 선형 시간이 필요한 문제에 대한 확장성 있는 해결책을 찾을 때 가장 흔하게 사용하는 기법은 해당 문제를 이진 트리로 모델링하고 문제를 해결하는 비용이 트리의 높이에 비례하도록 하는 것이다. 이때 트리의 균형이 유지되어야 한다. 트리의 균형이 유지될 수 없으면 필요한 비용이 로그 비용이라 말하기 어렵다.

LKH(Logical Key Hierarchy) 기법은 바로 그룹키 문제를 이진 트리를 이용하여 확장성 문제를 해결한 기법이다[1, 2]. 이 기법에서 중앙 서버는 그림 12.1처럼 각 그룹의 멤버들과 공유한 키를 이진 트리의 단말 노드 값으로 할당하여 트리를 구성한다. 이때 중간 노드에는 독립적인 비밀키를 할당한다. 각 사용자는 자신의 단말부터 루트까지 경로 있는 모든 노드에 할당된 키를 중앙 서버로부터 받는다. 예를 들어 멤버 U_i가 K_i 노드에 할당되어 있다고 가정하였을 때 U_1은 4개의 대칭키 K_1, K_{12}, K_{14}, K_{18}을 받아야 한다. 여기서 K_1은 사용자와 중앙서버 간에 공유된 비밀키이기 때문에 나머지 키들은 K_1을 이용하여 암호화하여 U_1에게 분배할 수 있다.

이와 같이 트리를 구성하여 키를 배포하면 각 노드는 트리 높이만큼의 키를 유지해야 한다. n개의 노드가 있으면 $\log n$ 정도의 키를 유지해야 하므로 확장성을 가지고 있다. 모든 그룹 멤버가 루트 노드에 할당된 키를 가지고 있기 때문에 이 키가 그룹키가 된다. 또한 나머지 키도 소그룹 키로 활용이 가능하다. 예를 들어 K_{14}는 U_1부터 U_4에 비밀 메시지를 전송하기 위해 사용이 가능하다.



<그림 12.2> LKH/OFT: 가입/탈퇴 전 모습



1) 그림 12.1에서 U_5가 가입한 후 모습 2) 그림 12.1에서 U_4가 탈퇴한 후 모습

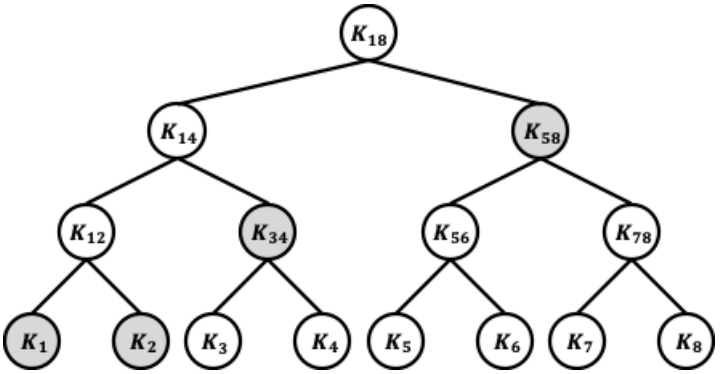
<그림 12.3> LKH: 가입/탈퇴 후 모습

LKH에서 그림 12.2처럼 4명의 멤버가 있을 때 새 멤버의 가입은 그림 12.3.1)처럼 진행된다. 단말 중 가장 낮은 레벨에 있는 단말의 형제로 가입한다. 기존 단말은 새 노드의 형제 노드가 되고, 원래 단말이 있던 위치는 중간 노드가 된다. 이때 후방향 안전성을 제공하기 위해서는 새 노드부터 루트까지 경로에 있는 기존 중간 노드에 할당되어 있는 키를 변경해야 한다. 노드의 키 값이 변경되면 그 값들을 기존 노드들에게 알려 주어야 한다. 이를 위해 LKH에서는 트리 높이에 비례하는 암호문을 생성하여 멀티캐스트나 유니캐스트로 전달한다. 여기서 중요한 것은 계산 비용이나 통신 비용이 모두 트리 높이에 비례한다는 것이다. 이처럼 기존 키를 이용하여 바뀐 키들을 전달하기

때문에 한 번 키 갱신 과정에 참여하지 못하면 그 이후 키 갱신 과정에 참여하더라도 필요한 키를 얻을 수 없다. 따라서 이 기법은 상태 기반 프로토콜이다.

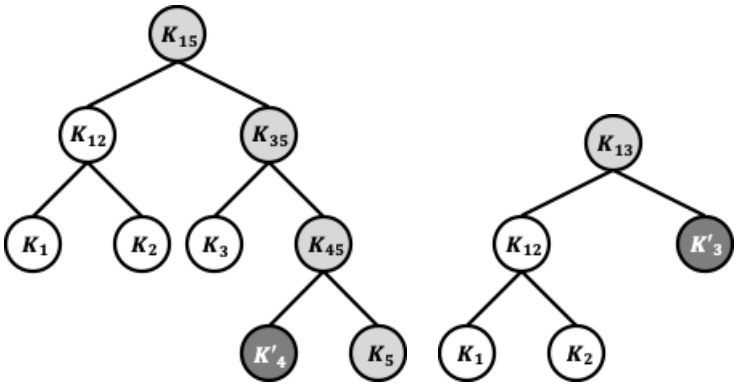
LKH에서 그림 12.2처럼 4명의 멤버가 있을 때 U_4의 탈퇴는 그림 12.3.2)처럼 진행된다. 탈퇴한 노드의 형제 노드는 부모 노드를 대체하게 되며, 전방향 안전성을 보장하기 위해 탈퇴한 노드가 알고 있는 노드의 키 값을 변경해야 한다. 가입 프로토콜과 마찬가지로 변경된 키들을 자식 노드의 키로 암호화하여 멀티캐스트 또는 유니캐스트 한다. 필요한 비용은 가입과 마찬가지로 트리 높이에 비례한다. 이와 같은 방법으로 탈퇴가 진행되면 이진 트리의 균형은 깨질 수 있다. 하지만 가입은 낮은 레벨부터 진행되기 때문에 평균적으로 가입과 탈퇴 비용을 $O(\log n)$으로 분석할 수 있다.

2.3 OFT



<그림 12.4> OFT 트리

OFT(One-way Function Tree) 기법[3]은 기존 LKH 기법에서 필요한 비용을 반으로 줄인 기법으로써 중간 노드에 키를 독립적으로 할당하지 않고, 자식 노드의 키를 이용하여 계산하는 방식을 사용하고 있다. 예를 들어 K_1과 K_2가 두 형제 단말 노드에 할당된 키일 때 그것의 부모 노드 키는 $f(g(K_1)||g(K_2))$가 되며, 여기서 f와 g는 일방향 해시함수이다. 11장에서 학습한 머클 트리와 동일한 형태이다. LKH에서 사용자는 자신이 할당된 단말 노드부터 루트 노드까지의 키를 유지하는 반면에 OFT에서는 루트를 제외한 해당 노드들의 형제 노드 키의 해시값을 유지한다. 즉, 그림 12.4에서 U_1의 경우 K_1, $g(K_2)$, $g(K_{34})$, $g(K_{58})$을 유지한다. 이처럼 유지하면 LKH와 동일하게 자신의 노드부터 루트 노드까지의 키를 계산할 수 있다. 여기서 핵심은 자신이 유지해야 하는 키 값을 계산하기 위한 절반의 정보를 항상 가지고 있다는 것이다.



1) 가입 프로토콜 2) 탈퇴 프로토콜

<그림 12.5> OFT: 가입/탈퇴 후 모습

OFT에서 가입 위치는 LKH와 동일하다. 하지만 필요한 암호문의 수가 반으로 감소한다. 그 이유는 노드의 값을

계산하기 위한 정보 중 반을 항상 노드들이 가지고 있기 때문이다. 또 다른 중요한 차이점은 가입하는 노드의 형제 노드가 되는 사용자, 탈퇴하는 노드의 형제 노드는 자신의 키를 바꾸어야 한다. 만약 바꾸지 않으면 새 사용자가 기존 키를 계산할 수 있거나 탈퇴한 사용자가 이전 키를 계속 알 수 있게 된다.

2.4 Naor 등의 기법

Naor 등의 기법[4]은 LKH, OFT처럼 논리적 키 계층 구조를 사용하지만, 루트 노드에 할당된 키가 그룹키가 아니다. 이 기법은 트리에 있는 키(트리에 있는 키가 변하지 않음)를 바꾸지 않고 계속 이용한다. 즉, 멤버의 동적 변화를 고려하지 않는다. 하지만 매번 메시지를 전송할 때 전체 그룹 멤버 중 수신해야 하는 그룹 멤버를 바꿀 수 있다.

중앙 서버의 역할도 기존 LKH, OFT와 다르다. LKH, OFT에서 중앙 서버의 핵심 역할은 사용자의 가입과 탈퇴에 따른 그룹키의 갱신과 분배이다. 이 기법에서 중앙 서버는 사용자로부터 메시지를 받아 멀티캐스팅 해주어야 한다. LKH, OFT에서 그룹 통신은 각 사용자가 직접 보유한 그룹키를 이용하지만, 이 기법에서 사용자가 비밀 그룹 통신을 하고 싶으면 대상 부분 그룹과 메시지를 서버에게 알리면 서버가 해당 메시지를 사용자 대신 적절한 키로 암호화하여 멀티캐스팅한다. 이와 같은 방식으로 동작하기 때문에 이 기법은 비상태 기반 기법이다. Naor 등은 이와 같은 방식의 완전 부분 트리(CS, Complete Subtree) 기법과 부분 트리 차이(SD, Subtree Difference) 기법, 두 가지 기법을 제안하였다.

2.4.1 CS 기법

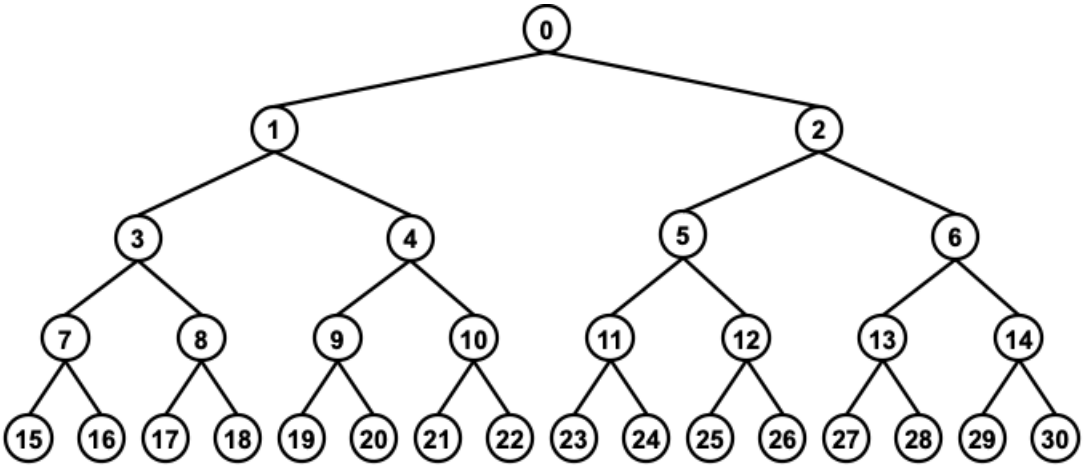
CS 기법은 LKH와 동일한 논리적 키 트리를 만들며, 각 사용자는 LKH와 동일하게 단말 노드부터 루트 노드까지의 키를 유지한다. 그림 12.1에서 사용자 U_1, U_2, U_6를 제외하고 나머지 사용자들에게 비밀 메시지를 전달하고 싶으면 메시지를 새 그룹키로 암호화하고 새 그룹키를 K_{34}, K_5, K_{78}로 암호화하여 전달하는 방식이다. 이처럼 제외되지 않은 사람들만 포함되는 가장 큰 부분 트리들을 찾고, 해당 부분 트리의 루트 노드에 할당된 키로 그룹키를 암호화하는 방식이다. 수신해야 하는 사용자들을 가장 크게 묶을 수 있는 부분 트리들을 찾고, 그것의 루트 노드에 할당된 키를 이용하여 비밀 통신을 하는 것이 CS 기법이다.

2.4.2 SD 기법

CS 기법은 제외하는 사용자들이 어떻게 배치되어 있는지에 따라 사용되는 부분 트리의 개수가 많아질 수 있는 단점이 있다. 이를 극복하기 위해 사용자들이 자신이 할당된 노드부터 루트 노드까지의 키를 유지하는 것이 아니라 그들을 제외한 나머지 키를 모두 알 수 있도록 하는 방법을 생각해 볼 수 있다. 이 경우 U_1만 제외하고 싶을 경우, CS 기법에서는 새 그룹키를 K_2, K_{34}, K_{58}로 암호화하여 보내면 된다. 반대로 제안한 기법에서는 새 그룹키를 K_1으로만 암호화하여 보내면 된다. 그러나 한 사용자가 자신의 노드부터 루트 노드까지의 키들을 제외한 모든 키를 유지해야 하면 한 사용자가 유지해야 하는 키의 개수가 너무 많아 확장성 있는 해결책이 되지 못한다. 또한 U_1, U_5를 제외하고 싶을 경우 U_1이 모르는 키는 U_5가 알고 있고, 반대도 마찬가지이므로 이와 같은 방법으로는 문제를 해결할 수 없다.

SD 기법은 이 문제를 해결하기 위해 부분집합 개념을 사용한다. SD 기법에서 사용하는 부분집합은 항상 두 개의 집합 G_i와 G_j를 이용하여 정의한다. 이때 $G_j \subset G_i$이다. 이 때문에 이 기법의 이름이 부분 집합 차이 기법이다. 여기서 G_i는 해당 노드 아래에 할당된 사용자 집합을 나타낸다. 따라서 G_0는 모든 사용자를 포함하는 집합이다. 이 기법에서 부분집합 $S_{i,j} = G_i - G_j$이다.

이 기법은 각 부분집합마다 키가 할당되어 있으며, 제외한 사용자들이 포함되지 않도록 서로 독립된 부분집합의 키를 이용하여 랜덤한 새 대칭키를 암호화하고, 새 대칭키로 메시지를 암호화하여 멀티캐스트한다. 예를 들어 그림 12.6과 같이 16명으로 구성된 트리가 있다고 하자. 여기서 U_1만 제외하고 싶으면 $S_{0,15}$ 부분집합에 할당된



<그림 12.6> SD 기법

키로 새 대칭키를 암호화하고 새 대칭키로 메시지를 암호화하여 멀티캐스트하면 U_1을 제외한 나머지 사용자들이 메시지를 얻을 수 있다.

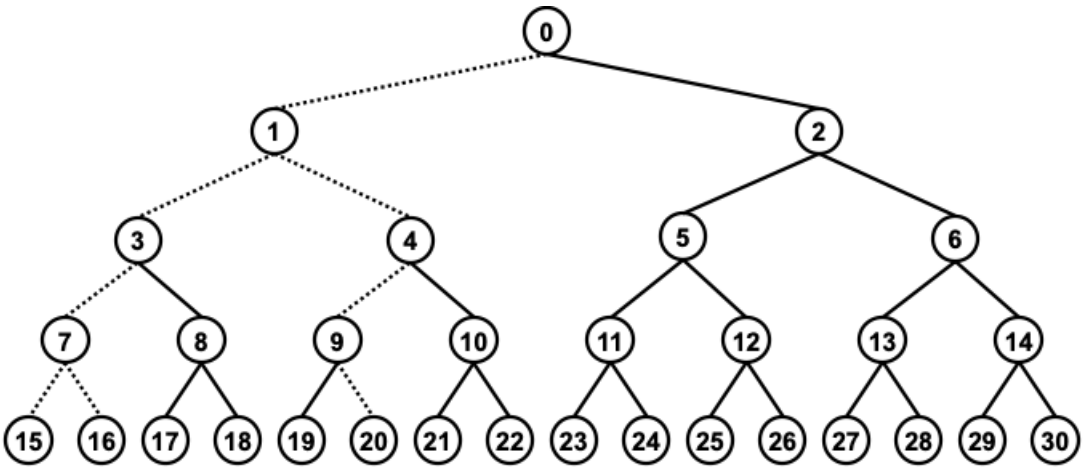
이 방식에서 각 사용자는 자신이 소속된 모든 부분집합의 키를 가지고 있어야 한다. 하지만 이 키들이 독립적이면 유지해야 하는 키가 너무 많아 확장성이 없다. 실제 노드가 유지해야 하는 키는 자신이 할당된 단말부터 루트까지의 노드를 중심으로 이 노드를 제외한 노드의 수만큼의 키가 필요하다. 예를 들어 U_1의 경우 다음과 같은 키들이 필요하다.

- 노드 0: (0,2), ..., (0,30), (0.4), ..., (0,22), (0,8), ..., (0,18), (0, 16)
- 노드 1: (1,4), ..., (1,22), (1.8), ..., (1,18), (1,16)
- 노드 3: (3,8), ..., (3,18), (3,16)
- 노드 7: (7,16)

앞서 언급한 바와 같이 이 키들을 모두 독립적으로 만들어 각 사용자에게 보내는 것은 확장성이 없다. SD 기법은 이 문제를 해결하기 위해 각 부분집합 $S_{i,j}$마다 노드 i에 K_i를 하나 할당한 후에 이 키를 이용하여 모든 $S_{i,j}$를 계산할 수 있도록 만들었다. 두 개의 해시 함수 H_L, H_R를 이용하여 왼쪽과 오른쪽 자식의 값을 계산한다. 예를 들어 $S_0 = K_0$이면 $S_{0,1} = H_L(S_0)$, $S_{0,2} = H_R(S_0)$이 되며, 이것을 이용하여 반복적으로 부분집합의 키를 계산한다. 이 방식을 사용하면 U_1에게는 $S_{0,2}$, $S_{0,4}$, $S_{0,8}$, $S_{0,16}$, $S_{1,4}$, $S_{1,8}$, $S_{1,16}$, $S_{3,8}$, $S_{3,16}$, $S_{7,16}$만 제공해 주면 된다. 각 사용자에게 전달해야 하는 키는 $O((\log n)^2)$개 정도가 된다.

모든 사용자에게 필요한 부분집합의 키를 배포하였으면 필요한 사용자들을 쉽게 제외하고 암호화된 메시지를 멀티캐스트할 수 있다. 특정 사용자들을 제외하고 싶을 때 사용할 부분집합은 그림 12.7에 제시된 점선으로 구성된 트리처럼 제외할 사용자의 노드만 포함하는 트리를 먼저 구해야 한다. 이 트리를 Steiner 트리라 한다. 이 트리에서 최대 체인(maximal chain)을 구하면 필요한 부분집합을 구할 수 있다. Steiner 트리에서 체인이란 트리에 있는 경로로 경로에 있는 마지막 노드를 제외하고 노드의 자식이 하나만 있는 경로를 말한다. 그림 12.7에 제시된 Steiner 트리에서는 (0,1), (3,7), (4,20) 3개의 최대 체인이 있다. 따라서 U_1, U_2, U_6를 제외하고 싶으면 $S_{0,1}$, $S_{3,7}$, $S_{4,20}$의 키로 새 대칭키를 암호화하고 메시지를 새 대칭키로 암호화하여 멀티캐스트하면 된다.

CS와 SD 기법은 그룹의 멤버가 고정된 상태에서 전송할 때마다 일부 멤버를 제외하는 환경에서는 사용할 수 있는 기법이며, 기존 LKH, OFT처럼 멤버가 빈번하게 탈퇴하고 가입하는 환경에서는 사용하기 어려운 기법이다.



<그림 12.7> SD 기법에서 3명의 사용자를 제외할 경우 만들어지는 Steiner 트리

3. 탈중앙형

그룹키에서 탈중앙형이란 단일 중앙 서버를 사용하지 않고 여러 개의 서버를 사용하는 방식이다. 이와 같은 방식에서 하나의 서버가 동작을 중단하여도 나머지 소그룹 동작에 영향을 주지 않는 이점이 있다. 이와 같은 탈중앙형 그룹키는 다음과 같은 추가적인 요구사항이 있다.

- 지역키 독립성: 부분그룹의 멤버 변화는 전체에 영향을 주지 않아야 한다.
- 그룹 간 독립성: 한 멤버는 여러 부분 그룹에 동시에 가입할 수 없어야 한다.
- 키와 데이터 간의 관계: 키 관리 경로와 데이터 전달 경로가 독립적이어야 한다. 즉, 부분 그룹의 멤버 변화가 데이터 전달을 방해하거나 지연시키지 않아야 한다.

Mittra는 Iolus라는 탈중앙 방식을 제안하였으며[5], 이 기법에서 전체 그룹은 여러 개의 부분 그룹으로 나누어지며, 각 부분 그룹은 GSA(Group Security Agency)라는 키 서버가 관리한다. 또한 전체 그룹을 총괄하는 GSC(Group Security Control)라는 하나의 서버를 두고 있다. 각 GSA는 GSC를 통해 그룹키를 확립하고, 각 부분 그룹의 멤버들은 GSA를 통해 해당 소그룹의 그룹키를 확립한다. 확립하는 그룹키 방식은 소그룹마다 다른 방식을 사용할 수 있다. 특정 소그룹에 소속된 멤버가 전체 멤버에게 비밀 그룹 통신을 하고 싶으면 다음 과정을 통해 이루어진다.

- 단계 1. 부분 그룹키로 자신이 속한 부분 그룹에 멀티캐스트한다.
- 단계 2. 해당 그룹의 GSA는 이를 받아 복호화한 후에 GSA 간에 그룹키로 암호화하여 다른 GSA에게 멀티케스트한다.
- 단계 3. 수신한 다른 GSA는 이를 복호화한 후에 자신의 속한 소그룹의 그룹키를 이용하여 소그룹 멤버들에게 멀티캐스트한다.

Iolus에서 가입과 탈퇴는 해당 소그룹에만 영향을 주고 다른 소그룹과는 독립적으로 진행된다. 이때 해당 소그룹에서 사용하는 그룹키 확립 방식에 따라 가입과 탈퇴가 이루어진다.

4. 분산형

분산형은 그룹키를 분배하기 위한 별도 키 서버를 전혀 사용하지 않는 방식이다. 실제 현장에서 분산형에 대한 요구가 있어 연구된 것이라고 보기는 힘들다. 키 분배를 위한 서버를 사용하지 않더라도 멤버의 가입과 탈퇴를 관리하는 서버가 필요할 수 있으며, 가입과 탈퇴 관련 규칙이나 약속을 자체 강화 방식으로 멤버 간에 자율적으로 이루어지기도 힘들다. 하지만 연구는 필요성이 없더라도 연구 차원에서 살펴볼 수 있는 것이다.

Perrig[6]은 Diffie-Hellman 키 동의 프로토콜을 형제 노드 간에 수행함으로써 분산형 LKH가 가능하다는 것을 보였다. 하지만 노드들 간에 지속적으로 키 동의 프로토콜을 진행해야 하며, 해당 프로토콜에 참여하지 않은 다른 사용자에게 필요한 키를 전달해 주어야 한다. Kim 등[7]은 Perrig의 제안에 대해 2가지 개선 방안을 제시하였다. 첫째는 논리 키 트리가 완벽 이진 트리일 때에 멤버의 가입은 기존처럼 단말 노드의 형제 노드로 가입하지 않고 루트 노드의 형제 노드가 되도록 가입하는 개선안을 제안하였다. 이 개선안은 기존 LKH, OFT에서도 사용할 수 있는 기법이다. 둘째는 sponsor 노드라는 개념을 도입하여 키 동의 프로토콜을 수행해야 하는 노드를 결정하는 규칙을 제안하였다. Ren 등[8]은 기존 기법들이 중간 노드의 키를 계산할 때 무조건 Diffie-Hellman 키 동의 프로토콜을 수행하고 있는 문제점을 발견하였다. 단말 형제 노드들을 제외하고는 키 동의 프로토콜을 사용하지 않더라도 논리 키 트리를 구축할 수 있으며, 단말 형제 노드들도 서로 다른 프로토콜을 사용하여 키 확립도 가능하다는 것을 보였다.

5. 기타 프로토콜

Chiou와 Chen은 "secure lock"이라는 중국인 나머지 정리를 이용한 중앙집중 방식의 그룹키 프로토콜을 제안하였다[9]. 중국인 나머지 정리란 연립합동식의 해를 구하는 수학 정리이며, 이 정리를 이용하면 서로 다른 값을 하나의 값으로 바꾸어 전달할 수 있게 해준다. 그룹의 크기가 m이고 각 멤버 U_i와 중앙 서버 간의 K_i를 공유하고 있다고 하자. 그러면 단순 접근 방법에서 설명한 첫 번째 방식을 사용하면 총 m개의 $\{K\}.K_i$ 형태의 암호문이 필요하다. 각 사용자가 전달할 암호문의 크기보다 큰 정수 n_i를 가지고 있고, 서로 다른 n_i가 서로소이면 연립합동식 $x \equiv \{K\}.K_1 \pmod{n_1}$, ..., $x \equiv \{K\}.K_m \pmod{n_m}$의 해를 중국인 나머지 정리로 구한 다음 각 사용자에게 동일한 x 값을 전달하여 그룹키를 분배할 수 있다. 하지만 x 값은 $n_1 n_2 \cdots n_m$ 법의 한 값이므로 매우 큰 값이며, m개의 암호문을 합친 크기에 비례하므로 중국인 나머지 정리를 이용하는 것이 큰 효과를 발휘하지는 못한다. 이전에 설명한 것처럼 서로 다른 여러 개의 메시지를 결합하여 하나의 멀티캐스트로 보내 얻어지는 긍정적 효과는 하나도 없으며, 이것은 중국인 나머지 정리를 이용하여도 차이가 없다.

Bresson 등은 저전력 모바일 이동 노드를 고려한 중앙집중형 그룹키 프로토콜을 제안하였다[10]. 주된 목적은 그룹키를 확립할 때 각 이동 노드의 비용을 줄이는 것이다. 실제 그룹키 확립할 때 각 이동 노드는 한 번의 해시 연산과 XOR 연산만 필요하다. 하지만 이동 노드의 비용을 줄이기 위해 초기 설정 비용이나 키 서버의 비용은 상대적으로 높다. 더 구체적으로 설명하면 이동노드 MN_i는 서버와 설정 과정에서 Diffie-Hellman 키 동의 프로토콜을 진행한다. 이를 통해 각 이동 노드와 서버는 비밀 정보 α_i를 공유하게 된다. 서버는 이들 비밀정보를 모두 이용하고, 여기에 카운터 값을 포함하여 그룹키 $K = H(c||\alpha_1||\alpha_2||\ldots||\alpha_n)$를 계산한다. 그다음 $K_i = K \oplus H(c||\alpha_i)$를 계산하여 각 이동 노드에게 전달한다. 각 이동 노드는 $K = K_i \oplus H(c||\alpha_i)$를 계산하여 그룹키를 얻는다.

90년 중후반 모바일 컴퓨팅에 대한 연구가 시작될 무렵에는 기존 프로토콜을 모바일 이동 환경에 적합하도록 수정하는 것에 대한 연구가 많이 진행되었다. 이들 연구에서는 모바일 기기의 에너지를 절약하고 통신 메시지의 크기를 줄이는 대신 서버의 역할을 높이는 방식으로 프로토콜을 수정하였으며, 오프라인에서 사전 계산한 값을 유지하여 활용하는 방법 등도 사용하였다. Bresson 등은 기법은 이와 같은 연구의 결과 중 하나이다.

6. 다자간 키 확립 프로토콜의 응용

6.1 유료 실시간 방송 서비스

유료 실시간 방송 서비스에서는 요금을 지불한 가입자만이 방송을 볼 수 있어야 한다. 이를 위해 수신제한시스템(CAS, Conditional Access System)이라는 다자간 키 확립 프로토콜을 사용한다. 이 CAS는 지금도 사용하고 있는 기술이지만 초창기 유료 방송 서비스부터 활용된 기술이다. 즉, 기존 방송 통신 환경이 일방향 통신 채널이라는 제한점을 고려하여 개발된 기술이다. 일방향 통신 채널이란 방송국에서 가입자로는 정보의 전달이 가능하지만, 가입자가 방송국에게는 정보를 전달할 수 없는 환경을 말한다. CAS는 통신 채널만 보호하는 기술이며, 암호화된 방송 데이터가 복호화된 이후에는 이를 보호하기 위한 다른 보안 메커니즘이 필요하다. CAS에서 방송 데이터의 암호화와 복호화하는 과정을 스크램블(scramble)과 디스크램블(descramble)이라는 용어를 사용한다.

CAS의 기본적인 원리는 매우 간단하다. 방송 콘텐츠를 스크램블하여 방송하고 합법적인 가입자들만 이를 디스크램블하여 볼 수 있게 하는 것이다. 그런데 CAS에서 고려되는 가입자의 규모는 매우 크다. 또한 가입자들이 수신기를 항상 켜 놓는 것도 아니다. 이와 같은 특성들 때문에 보안 측면보다 완전성과 효율성이 더욱 중요시되어 개발된 기법이다. CAS에서 가입자들은 적절한 키가 포함된 스마트카드를 발급받으며, 이 스마트카드는 방송 수신기인 셋톱박스에 설치되어 사용된다. 최근에는 스마트카드를 사용하지 않고 CAS 소프트웨어 자체에 키를 포함하는 방식을 사용한다.

CAS 기법을 이해하기 위해서는 유료 방송 환경에 대한 이해가 먼저 선행되어야 한다. 유료 방송 서비스도 가입과 탈퇴가 빈번하게 계속 이루어진다. 하지만 한 가입자의 가입과 탈퇴가 있을 때마다 그룹키를 갱신하는 것은 현실적이지 못하다. 또한 전방향, 후방향 안전성을 고려하였을 때 후방향보다는 전방향이 더 중요한 서비스이다. 전방향이 중요하더라도 탈퇴가 발생한 시점에 즉시 갱신하지 않고 일정 기간 후에 갱신하는 것은 방송이란 특성 때문에 큰 문제가 되지 않을 수 있다. 또 후방향이 중요하지 않다고 생각하면 가입의 경우에는 기존 그룹키를 바꾸지 않고 새로운 가입자에게 기존 키를 주는 방법도 문제가 되지 않을 수 있다. CAS의 동작 원리는 대략적으로 알려졌지만 각 업체가 실제 사용하는 자세한 내부 메커니즘은 비즈니스 측면에서 공개하지 않고 있다. 앞서 언급한 바와 같이 가입자들이 언제 자신의 수신기를 켤지 모르기 때문에 상태 기반이면 갱신 메시지를 받지 못한 가입자는 시청이 가능하지 않게 되므로 비상태 기반 기법을 사용해야 한다. 참고로 방송의 특성상 방송국은 제공하는 모든 채널의 콘텐츠를 항상 모두 전달하는 방식을 사용하고 있다.

CAS는 마스터 개인키(MPK, Master Private Key), AK(Authorization Key), CW(Control Word), 총 세 종류의 키를 사용한다, 사용자는 서비스에 가입하면 마스터 개인키가 발급된다. 이 키는 공개키 방식이며, 각 사용자의 스마트카드에 저장되어 배포되거나 셋톱박스에 설치된 소프트웨어에 안전하게 포함되어 있다. 따라서 가입 이후 탈퇴할 때까지 보통 바뀌지 않는다. 각 방송 채널마다 다른 AK을 사용한다. 이 키는 대칭키이며, 갱신 주기는 보통 몇 주 이상이다. 실제 방송 콘텐츠는 CW를 이용하여 스크램블/디스크램블된다. 이 키의 갱신 주기는 5–20초이며, 키의 길이는 48bit 또는 60bit이다.

CW의 갱신주기를 이렇게 짧게 한 가장 큰 이유는 가입자가 언제 필요할지 모르기 때문이다. 가입자는 자신의 수신기를 언제 켤지 알 수 없다. 따라서 수신기를 켜는 순간 수신한 콘텐츠를 보기 위해서는 가입자는 항상 최신의 CW가 필요하다. 따라서 갱신 주기가 길더라도 해당 주기 처음에만 보내는 것이 아니라 계속 같은 값을 보내야 하므로 갱신 주기를 늘려 비용을 줄일 수 있지 않다. 그러므로 차라리 갱신 주기를 짧게 하면 키 길이도 짧게 할 수 있는 장점이 있기 때문에 갱신 주기를 짧게 한 것이다.

이 3가지 키를 이용하여 방송 콘텐츠를 어떻게 전달하는지 살펴보자. 각 채널의 콘텐츠는 다른 CW를 이용하여 스크램블된다. 스크램블된 콘텐츠에는 가입자들이 CW를 얻을 수 있도록 AK로 CW를 암호화한 암호문이 포함된다. 이 암호문을 ECM(Entitlement Control Message)이라 한다. 따라서 각 채널 콘텐츠마다 하나의 ECM이 포함된다. 처음 수신기를 켜면 MPK만 가지고 있으므로 각 가입자의 공개키로 AK를 암호화한 암호문을 콘텐츠에 포함한다. 이 암호문을 EMM(Entitlement Management Message)이라 하며, 해당 채널의 가입자 수가 n이면 n개의 EMM이

해당 채널 콘텐츠에 포함된다. 따라서 지금까지 살펴본 확장성 개념과는 거리가 먼 방식을 사용하고 있음을 알 수 있다.

이와 같은 방식에서 주기가 어떤 역할을 하는지 살펴보자. AK의 주기는 몇 주 이상으로 길기 때문에 EMM은 자주 변하지 않는다. 따라서 스크램블된 콘텐츠마다 n개의 EMM을 포함하더라도 매번 n개의 암호문을 생성해야 하는 것은 아니다. 반대로 CW의 주기는 매우 짧기 때문에 ECM은 매우 빈번하게 바뀐다. 하지만 각 스크램블된 콘텐츠에는 하나의 ECM만 포함해야 하기 때문에 서버에 부담되지 않는다. 이와 같이 3개의 키를 사용하는 것이 최소이다. 최소한 2개(사용자별 키, 콘텐츠 암호키)가 필요하지만 2개만 사용할 경우에는 짧은 주기마다 n개의 암호문을 생성해야 하므로 서버에게 너무 부담이 되는 기법이다. 반대로 4개, 5개의 키를 사용하도록 기법을 확장할 수 있다. 키가 많아지면 서비스 제공자의 노력을 감소시킬 수 있지만 전달하는 정보량 측면에서는 감소하는 효과는 없으며, 각 참여자의 비용도 차이가 없다.

예를 들어 4개의 키를 사용하는 방식을 생각하여 보자. AK와 MPK 사이에 GK를 두어 채널 그룹별 같은 GK를 사용한다고 하자. 이 경우 각 채널 콘텐츠에 AK로 CW를 암호화한 암호문, AK를 GK로 암호화한 암호문, GK를 각 MPK로 암호화한 암호문이 필요하다. 즉, 암호문 수 측면에서는 기존보다 하나가 더 증가한다. 하지만 서비스 제공자 입장에서는 채널 그룹마다 포함해야 하는 EMM이 같아지기 때문에 비용을 절약할 수 있다.

예를 들어 3계층 키를 사용할 때 3개 채널로 전달되는 암호문은 다음과 같다.

$$\begin{aligned} CH1: &\quad \{CW\}.AK_1, \{AK_1\}.MPK_1, \ldots, \{AK_1\}.MPK_n \\ CH2: &\quad \{CW\}.AK_2, \{AK_2\}.MPK_1, \ldots, \{AK_2\}.MPK_n \\ CH3: &\quad \{CW\}.AK_3, \{AK_3\}.MPK_1, \ldots, \{AK_3\}.MPK_n \end{aligned}$$

여기서 CW는 모두 같게 표현했지만 CW는 매번 새롭게 임의로 선택되는 값이며, 각 채널에 포함된 CW는 다르다. 이때 이 3개 채널이 같은 채널 그룹에 소속되어 있다고 가정하고 4계층 키를 사용하면 각 채널로 전달되는 암호문은 다음과 같이 바뀐다.

$$\begin{aligned} CH1: &\quad \{CW\}.AK_1, \{AK_1\}.GK, \{GK\}.MPK_1, \ldots, \{GK\}.MPK_n \\ CH2: &\quad \{CW\}.AK_2, \{AK_2\}.GK, \{GK\}.MPK_1, \ldots, \{GK\}.MPK_n \\ CH3: &\quad \{CW\}.AK_3, \{AK_3\}.GK, \{GK\}.MPK_1, \ldots, \{GK\}.MPK_n \end{aligned}$$

이 예를 통해 알 수 있듯이 사용자 측면에서 얻어지는 이득은 없지만, 서비스 제공자 측면에서 생성해야 하는 암호문 수를 줄일 수 있다.

초창기 CAS는 하드웨어 칩으로 구현되어 셋톱박스에 설치되어 사용하였지만 최근에 펌웨어 형태의 소프트웨어로 구현되어 사용된다. 이를 DCAS(Downloadable CAS)라 한다. XCAS(eXchangeable CAS), ICAS(Interchangble CAS)도 유사한 개념이다. 이들은 기존과 달리 키를 스마트카드에 유지하지 않고 소프트웨어 자체에 내장하는 방식을 사용한다. 이와 같이 다운받아 설치하는 소프트웨어 형태로 사용하면 CAS 기법을 동적으로 변경할 수 있는 이점도 있다. 비용 측면에서 상대적으로 저렴한 방식이다. 보안 측면에서 DCAS는 외부의 물리적 해킹으로부터 보호되어야 하며, 새 버전의 소프트웨어를 안전하게 다운받을 수 있어야 하고, 다운받은 소프트웨어의 무결성을 검증할 수 있어야 한다.

CAS는 일방향 통신 환경을 고려하여 설계된 방식이지만 지금은 양방향 통신이 가능한 방송 환경이기 때문에 이를 고려하여 여러 개선이 가능하지만 환경이 바뀌었음에도 불구하고 여전히 동일 방식을 사용하고 있다. 예를 들어 IPTV 이후에는 가입자에서 방송국 간의 양방향 통신이 가능하기 때문에 수신기가 부팅될 때 방송국과 프로토콜을 수행하여 필요한 키를 각 기기마다 개별적으로 보내줄 수 있다. 이와 같은 방식을 사용하면 n개의 EMM을 매번 포함하지 않고 서비스가 가능할 수 있다. 참고로 VOD는 CAS 기술을 사용하지 않고 DRM 기술을 사용하여 보호되고 있으며, DRM은 CAS와 보안 목적과 방식이 전혀 다르다. DRM은 다운된 디지털 콘텐츠의 시청 제한과 복제 방지가 목적이다.

6.2 메신저 보안

메신저는 SMS와 달리 그룹 채팅 기능을 제공하며, 일반 채팅 프로그램과 달리 모든 그룹 멤버가 동시에 온라인 상태가 아니더라도 그룹 채팅이 가능하다. 기본적으로 메신저들은 정부 기관에 의한 감시 활동과 서비스 제공자에 대한 신뢰 문제로 사용자들이 다양한 메신저를 선택할 때 메신저에서 제공하는 보안 서비스가 선택의 중요한 기준이 되었다. 현재는 기본적으로 종단간 암호화 서비스를 제공하여 중앙서버도 2자간 주고 받은 메시지를 볼 수 없게 되었다. 하지만 그룹 채팅의 경우에는 종단간 암호화 서비스를 제공하는 것이 간단한 문제가 아니다[11].

메신저 보안 프로토콜 중 가장 유명한 시그널의 경우에는 2자간을 기계적으로 그룹 채팅으로 확장하여 사용하고 있다. 이 때문에 시그널에서 그룹의 한 멤버가 n명이 참여한 그룹에 비밀 메시지를 보내기 위해서는 $n-1$개의 암호문을 생성하여야 한다. 따라서 확장성 있는 기법이라고 보기 어렵다. 물론 CAS 환경과 비교하면 그룹의 크기는 상대적으로 적을 수 있으며, 인라인 통신 방식을 사용한다는 측면도 고려해야 한다.

WhatsApp의 경우 기본적으로 시그널 프로토콜을 사용하지만, 다자간에는 시그널과 달리 사용자는 단일 암호문을 서버에 보내면 서버는 암호화하지 않은 메시지를 그룹에 보내는 방법과 같은 방법을 이용하여 나머지 멤버들에게 전달하여 준다. 이를 위해 각 그룹의 멤버는 처음으로 암호화된 그룹 메시지를 보낼 때 그룹키를 생성한 다음 이 그룹키를 그룹의 모든 멤버에게 2자 간 시그널 프로토콜을 이용하여 안전하게 전달한다. 또한 원래 시그널 프로토콜과 달리 이중 톱니바퀴 형태로 키를 갱신하지 않고 그룹키만 갱신하는 방식을 사용한다. 따라서 특정 메시지를 암호화하기 위한 키가 노출되면 과거 키들은 노출되지 않지만, 미래 키는 노출되는 문제점이 있다. 즉, 효율성을 위해 안전성을 희생한 경우로 볼 수 있다.

참고문헌

[1] D. Wallner, E. Harder, R. Agee, "Key Management for Multicast: Issues and Architectures," IETF RFC 2627, June 1999.

[2] C. K. Wong, M. G. Gouda, S. S. Lam, "Secure Group Communications using Key Graphs," IEEE/ACM Transactions on Networking, Vol. 8, No. 1, pp. 16–30, 2000.

[3] A. T. Sherman, D. A. McGrew, "Key Establishment in Large Dynamic Groups using One-way Function Trees," IEEE Transactions on Software Engineering, Vol. 29, No. 5, pp. 444–458, May 2003.

[4] Dalit Naor, Moni Naor, Jeff Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers," Advances in Cryptology, Crypto 2001, LNCS 2139, pp. 41–62, Springer, 2001.

[5] Suvi Mittra, "Iolus: A Framework for Scalable Secure Multicasting," Proc. of the ACM SIGCOMM '97, pp. 277–288, Sept. 1997.

[6] Adrian Perrig, "Efficient Collaborative Key Management Protocols for Secure Autonomous Group Communication," Int'l Workshop on Cryptographic Techniques and E-Commerce, CrypTEC '99, pp. 192–202, Jul. 1999.

[7] Yongdae Kim, Adrian Perrig, Gene Tsudik, "Simple and Fault-tolerant Key Agreement for Dynamic Collaborative Groups," 7th ACM Conf. on Computer and Communications Security, pp. 235–244, Nov. 2000.

[8] Kui Ren, Hyunrok Lee, Kwangjo Kim, Taewhan Yoo, "Efficient Authenticated Key Agreement Protocol for Dynamic Groups," Int'l Workshop on Information Security Applications, WISA 2004, LNCS 3325, pp. 144–159, Springer, 2005.

[9] Guang-huei Chiou, Wen-Tsuen Chen, "Secure Broadcasting Using the Secure Lock," IEEE Trans. on Software Engineering, Vol. 15, No. 8, pp. 929–934, Aug. 1989.

[10] Emmanuel Bresson, Olivier Chevassut, Abdelilah Essiari, David Pointcheval, "Mutual Authentication and Group Key Agreement for Low-power Mobile Devices," Computer Communications, Vol. 27, No. 17, pp. 1730–1737, Nov. 2004.

[11] Paul Rosler, Christian Mainka, Jorg Schwenk, "More is Less: On the End-to-End Security of Group Chats in Signal, Whatsapp, and Threema," IEEE European Symp. on Security and Privacy, 2018.

퀴즈

1. LKH와 OFT는 모두 이진 트리 형태의 논리적 키 계층구조를 사용하는 중앙집중 그룹키 프로토콜이다. 두 기법과 관련된 다음 설명 중 틀린 것은?

   ① LKH와 OFT에서 각 사용자는 모두 자신의 단말부터 루트까지 노드에 해당하는 키를 알아야 하지만 유지하는 값은 서로 다르다.

   ② LKH는 각 노드에 독립적인 키를 할당하지만 OFT는 머클 해시 트리처럼 두 개의 자식 노드의 값을 이용하여 중간 노드의 값을 계산한다.

   ③ 전방향, 후방향 안전성을 위해 가입한 사용자가 알게 되는 노드 값이나 탈퇴한 사용자가 알고 있던 값을 바꾸어 그룹 멤버들에게 알려주어야 한다. 사용자 수가 n이면 갱신된 값들을 알려주기 위해 총 $\log n$에 비례한 수의 암호문이 필요하다.

   ④ 두 방법 모두 가입한 사용자의 형제 노드나 탈퇴한 사용자의 형제 노드에 해당하는 노드의 값을 바꾸지 않아도 된다.

2. LKH와 OFT에서는 원래 사용자를 추가할 때에는 단말 노드의 형제 노드가 되도록 추가하였다. 분산 LKH 기법을 제안하면서 논리 키 트리가 완벽 이진 트리이면 루트의 형제 노드가 되도록 트리를 갱신하면 키 갱신에 필요한 암호문의 수를 줄일 수 있다는 것을 보였다. 이와 관련된 다음 설명 중 틀린 것은?

   ① 완벽 이진 트리일 때에만 루트의 형제 노드가 되도록 트리를 갱신해야 한다. 항상 루트의 형제 노드가 되도록 가입하면 트리의 높이가 선형적으로 계속 증가하기 때문에 키 갱신 메시지의 확장성을 보장할 수 없다.

   ② OFT는 형제 노드의 키 갱신이 필요하기 때문에 이 방법을 적용할 수 없다.

   ③ LKH는 각 키가 독립적이므로 이 방법을 적용하는데 문제가 없다.

   ④ LKH에서 루트의 형제 노드로 가입하면 새 루트 값을 기존 그룹키로 암호화하여 기존 멤버들에게 주고, 새 루트 값을 새 멤버에게 암호화하여 주어 키 갱신을 완료할 수 있다.

3. 수신제한시스템(CAS, Conditional Access System)과 관련된 다음 설명 중 틀린 것은?

   ① 가입자마다 접근 제어가 다를 수 있기 때문에 같은 키로 암호화된 방송된 콘텐츠를 모든 가입자에게 보내기 위해서는 최소한 2계층 키가 필요하다.

   ② 3계층 키를 사용할 경우 방송 콘텐츠를 암호화할 때 사용하는 CW 갱신 주기마다 총 $n+1$개의 암호문을 전송해야 한다. 하지만 이 중에 AK로 CW를 암호화한 ECM만 변하고 나머지 n개 암호문은 AK 주기 동안 변하지 않는다.

   ③ CW의 갱신 주기가 짧은 이유는 어차피 짧은 주기로 계속 CW를 전달해야 하기 때문이다. 각 가입자는 언제 시청을 시작할지 모르기 때문에 계속 CW를 보내주어야 한다.

   ④ 한 계층을 추가해 여러 채널을 관리하는 키를 하나 더 사용하면 CW 갱신 주기마다 총 $n+2$개의 암호문을 전송해야 하지만 방송국 입장에서는 여러 채널에 동일 n개 암호문을 전달할 수 있으므로 계산 비용을 줄일 수 있다. 가입자도 같은 그룹에 속한 채널 간 변경할 때에는 CW를 얻기 위한 노력이 감소한다.

연습문제

1. IP multicast에 대해 간단히 설명하시오.

2. 다자간 키 확립 프로토콜은 키 확립 프로토콜 중 하나이므로 키 확립 프로토콜의 기본 요구사항(키의 비밀성, 최근성, 용도, 확인)을 만족하여야 한다. 하지만 2자간과 같이 적은 인원이 참여하는 방식이 아니기 때문에 기존과 유사한 방법으로 요구사항을 충족시키기 어렵다. 키 확인 측면에서 만족시킬 수 있는지 여부를 논하시오.

3. 상태 기반과 비상태 기반 다자간 키 확립 프로토콜에 대하여 설명하시오. 왜 실시간 방송과 같은 환경에서는 비상태 기반이 필요한지 설명하시오.

4. 현재 논리 트리의 모습이 그림 12.2와 같다고 가정하자. LKH, OFT 방식을 각각 사용하여 사용자 1명이 가입하였다고 하자. 이때, 단말 노드에 가입하는 방식 대신에 루트 노드에 가입하는 방식을 사용한다고 가정하고, 중앙서버가 분배해야 하는 메시지와 해당 메시지를 유니캐스트로 또는 멀티캐스트로 전달하는지 설명하시오.

5. 상태 기반으로 소개된 LKH, OFT 기법과 비상태 기반으로 소개된 Naor 등의 CS, SD 기법은 사용하는 방식이 다르다. 어떤 차이점이 있는지 설명하시오.

6. Perrig 등이 제안한 분산형 그룹키 프로토콜은 트리의 단말부터 루트까지 올라가면서 계속 Diffie-Hellman 키 동의 프로토콜을 진행한다. Ren 등은 단말의 조부모 노드부터는 Diffie-Hellman을 할 필요가 없다고 주장하였다. 예를 들어 U_1, U_2가 키 동의 프로토콜을 진행하여 확립한 키가 K_{12}이고, U_3와 U_4가 확립한 키가 K_{34}라 할 때, U_2와 U_4는 이들을 이용하여 K_{14}를 확립해야 한다. 이 과정에서 Diffie-Hellman을 진핸하는 것과 OFT 방식을 사용하는 것의 차이를 구체적으로 비교 설명하시오.

7. CAS는 3 계층 키 구조를 사용한다. 중간 층에 해당하는 AK가 채널별 키라고 할 때, 중간 층을 사용하지 않으면 어떤 문제가 있는지 설명하시오.

8. CAS는 일방향 통신(방송국에서 셋톱박스)만 가능한 환경을 가정하여 만든 프로토콜이다. 현재 IPTV는 양방향 통신이 가능하다. 양방향 통신이 가능하면 CAS를 어떻게 개선할 수 있는지 설명하시오.

9. LKH나 OFT를 메신저 그룹 채팅에 적용이 가능한지 설명하시오. 가능하면 어떻게, 문제가 있다면 어떤 문제가 있는지 설명하시오.