난수 재사용 기법을 이용한 다중 키 교환 프로토콜

정 익 래[†] · 이 동 훈[††]

요 약

이 논문에서 우리는 여러 사용자들이 동시에 세션키를 교환하는 키 교환 스킴에 대해서 연구한다. 이런 상황은 사용자들을 그래프 노드들로 표현하고 두 사용자간에 키를 만들어야 하는 상황을 에지로 표현하는 키 그래프에 의해서 묘사될 수 있다. 우리는 키 그래프에 있는 모든 에지들을 위한 세션키들을 하나의 세션에서 동시에 만드는 키 교환 스킴을 설계한다. 키 그래프를 위한 키 교환은 양자간 키 교환의 확장판이라고 할 수 있다. 우리는 양자간 키 교환 안전성 모델을 확장해서 키 그래프를 위한 키 교환 안전성 모델을 제안한다. 우리는 *난수 재 사용* 테크닉을 사용해서 두개의 키 그래프를 위한 키 교환 스킴을 설계하며 안전성을 증명한다. 우리가 제안하는 스킴들의 안전성은 decisional Diffie-Hellman 가정에 의존한다. 첫 번째 스킴은 일 라운드 키 교환 프로토콜이며 *키 독립성*을 보장한다. 두 번째 스킴은 일 라운드이며 *전방위 안전성*을 보장한다. 제안하는 두 가지 스킴들의 안전성은 모두 표준 모델에서 증명된다. 제안되는 프로토콜들은 최초의 다중 키 교환 프로토콜이며, 단순히 양자간 키 교환 프로토콜들을 반복 사용해서 키들을 만드는 것보다 훨씬 효율적이다. 예로서 한 사용자가 n명의 다른 사용자와 키를 만든다고 가정하자. 가장 단순한 프로토콜은 각각의 사용자들과의 키 교환을 위해서 양자간 키 교환을 사용하는 것으로서, 이 때는 계산량과 메시지 전송량이 n에 비례하게 된다. 제안되는 프로토콜들은 n개의 키를 만드는데 있어서 계산량은 n에 비례하나 전송되는 메시지의 양은 교환되는 키들의 수에 상관없이 일정하다.

키워드 : 인증된 키 교환, 다중 키 교환, Diffie-Hellman 키 교환, 난수 재활용

Pairwise Key Agreement Protocols Using Randomness Re-use Technique

Ik Rae Jeong[†] · Dong Hoon Lee[††]

ABSTRACT

In the paper we study key agreement schemes when a party needs to establish a session key with each of several parties, thus having multiple session keys. This situation can be represented by a graph, called a key graph, where a vertex represents a party and an edge represents a relation between two parties sharing a session key. The purpose of the paper is to design a *key agreement protocol for key graphs* to establish all session keys corresponding to all edges in a key graph simultaneously in a *single* session. A key agreement protocol of a key graph is a natural extension of a two-party key agreement protocol. We propose a new key exchange model for key graphs which is an extension of a two-party key exchange model. Using the so-called *randomness re-use* technique which re-uses random values to make session keys for different sessions, we suggest two efficient key agreement protocols for key graphs based on the decisional Diffie-Hellman assumption, and prove their securities in the key exchange model of key graphs. Our first scheme requires only a single round and provides *key independence*. Our second scheme requires two rounds and provides *forward secrecy*. Both are proven secure in the standard model. The suggested protocols are the first pairwise key agreement protocols and more efficient than a simple scheme which uses a two-party key exchange for each necessary key. Suppose that a user makes a session key with n other users, respectively. The simple scheme's computational cost and the length of the transmitted messages are increased by a factor of n. The suggested protocols's computational cost also depends on n, but the length of the transmitted messages are constant.

Key Words : Authenticated Key Agreement, Pairwise Key Agreement, Diffie-hellman Key Agreement, Randomness Re-use

1. 서 론

키 교환 프로토콜은 가장 기본적이고 널리 사용되는 프로토콜이다. 키 교환 프로토콜은 두 사용자[12, 19, 14, 10, 6, 21, 11] 혹은 여러 사용자들[15, 8, 2, 7, 9, 18] 사이에서 하나의 세션키를 만드는 프로토콜이다. 따라서 안전한 키 교환 프로토콜은 좀 더 복잡한 어플리케이션을 만드는 기본적인 주춧돌 역할을 한다. 어떤 응용 프로토콜에서는 사용자가 다른 사용자들과 여러 개의 세션키를 만들 필요가 있다. 예를 들어서 사용자 P_1이 사용자 P_2, P_3, P_4와 각각 세션키를 만들 필요가

※ 이 논문은 한국학술진흥재단의 해외 Post-doc. 연수지원에 의하여 연구되었음.
† 정 회 원 : 고려대학교 정보보호기술연구핸터 포닥연구원
†† 정 회 원 : 컴퓨터 이론연구회 운영위원
논문접수 : 2005년 8월 4일, 심사완료 : 2005년 11월 16일

있으며 사용자 P_2는 P_1, P_3, P_4와 각각 세션키를 만들 필요가 있다고 하자. 또한 P_3는 P_1, P_2, P_4와 각각 세션키를 만들 필요가 있으며, P_4는 P_1, P_2, P_3와 각각 세션키를 만들 필요가 있다고 하자. 이 예에서는 사용자들의 집합이 $\{P_1, P_2, P_3, P_4\}$이며, 세션키들이 만들어져야 하는 사용자 쌍의 집합은 $\{(P_1, P_2), (P_1, P_3), (P_1, P_4), (P_2, P_3), (P_2, P_4), (P_3, P_4)\}$이다.

이 예는 키 그래프에 의해서 다음처럼 묘사될 수 있다. 즉, 키 그래프를 $G = \{V, E\}$라고 하면, $V = \{P_1, P_2, P_3, P_4\}$가 되고 $E = \{(P_1, P_2), (P_1, P_3), (P_1, P_4), (P_2, P_3), (P_2, P_4), (P_3, P_4)\}$가 된다.

유용한 키 그래프들의 구조로는 링, 스타, 트리 구조 등을 들 수 있다. 예를 들어서 카드 게임을 위해서는 사용자와 딜러 사이에 키를 만들 필요가 있으며, 이런 구조는 스타 구조로 표현된다. 또 다른 예로는 원격 회의를 진행할 때 각각의 사용자들끼리 메시지를 비밀리에 주고받을 필요가 있으며, 이런 상황은 모든 컴플리트 그래프에 의해서 표현된다. [8]에서는 그룹키 교환 스킴을 만들기 위해서 링 구조를 사용하며, [8, 5]에서는 스타 구조를 사용한다.

키 그래프를 위한 키 교환 프로토콜을 만들기 위해서 우리는 난수 재사용 테크닉을 사용한다. 난수 재사용 테크닉은 전송량과 계산량을 줄이기 위해서 다중 수신자 암호 스킴에서 사용되었다[17, 4]. 난수 재 사용 테크닉은 다음과 같이 decisional Diffie-Hellman 문제의 어려움에 기반해서 만들 수 있다. 예를 들어서 P_i가 g^{α_i}를 안전한 네트웍으로 브로드캐스팅하며 P_j와 세션키 $sk_{i,j} = g^{\alpha_i \alpha_j}$를 만든다고 하자. 난수를 재사용하지 않는다면 P_1은 세 개의 난수 $\alpha_{11}, \alpha_{12}, \alpha_{13}$을 사용해서, 세 개의 세션키 $sk_{1,2} = g^{\alpha_{11}\alpha_2}, sk_{1,3} = g^{\alpha_{12}\alpha_3}, sk_{1,4} = g^{\alpha_{13}\alpha_4}$을 P_2, P_3, P_4와 만들어야 한다. 난수 재사용 테크닉을 사용한다면 P_1은 하나의 난수 α_1을 사용해서 세 개의 세션키 $sk_{1,2} = g^{\alpha_1\alpha_2}, sk_{1,3} = g^{\alpha_1\alpha_3}, sk_{1,4} = g^{\alpha_1\alpha_4}$을 만들 수 있다. 비록 난수를 재사용 하더라도 decisional Diffie- Hellman의 가정이 맞다면 세 개의 세션키 $sk_{1,2}, sk_{1,3}, sk_{1,4}$는 여전히 계산적으로 구별이 불가능하다.

[4]에서 저자들이 "내부 공격"이라 불리는 다중 수신자 암호에 대한 새로운 공격 유형을 소개했다. 이 공격 유형은 중요한 공격이 될 수 있지만 [17]에서는 고려되지 않았다. 실생활에서는 시스템 사용자들이 공격자와 공모를 할 수 있으며 이런 내부 공격자는 그들의 공개/비밀 키를 정직한 사용자들의 공개키에 기반해서 만들 수 있으며, 이렇게 만들어진 키들을 사용해서 정직한 사용자들끼리 만든 세션키에 대한 정보를 얻으려 할 수 있다. 이런 내부자 공격은 키 그래프를 위한 키 교환 스킴에서도 중요하다. 특히 난수 재사용 테크닉을 사용해서 만들어진 키 교환 스킴은 이런 유형의 공격에 취약할 수 있다. 예를 들어서 하나의 세션에서 두개의 세션키 $sk_{1,2}, sk_{2,3}$가 만들어 진다고 하자. 만약 P_1이 내부 공격자이면 $sk_{1,2}$를 알 수 있고, $sk_{2,3}$에 대한 유용한 정보를 얻을 수 있을 수도 있다. 우리는 이런 내부자 공격을 키 교환 안전성 모델에 포함시켰으며, 제안하는 스킴들이 이런 내부자 공격들에도 안전함을 보였다.

키 교환 스킴을 만들기 위해서 난수 재사용 테크닉을 사용할 수 있으며, 이에 따른 결과로서 내부 공격자들이 키 그래프에 나타나지 않는 "부가적인 키"들을 그들 사이에 만들 수 있을 수도 있다. 예를 들어서 키 그래프가 사용자 P_1, P_2, P_3, P_4들 사이에서 세션키 $sk_{1,2}, sk_{2,3}, sk_{3,4}$만들 것을 의미한다고 하자. 만약 P_1과 P_4가 내부 공격자들이면, 그 둘을 위한 세션키 $sk_{1,4}$을 만들 수도 있을지도 모른다. 하지만 이런 부가적인 키들을 가지고서도 정직한 사용자들 사이에서 만들어진 세션키들에 대한 정보를 얻을 수 없다면, 이런 공격은 키 교환 스킴에 아무런 영향을 미치지 않는다.

1.1 기존 연구 및 이 논문의 기여도

기존 연구와의 관련성을 논하기 전에 먼저 키 교환 프로토콜에서의 안전성 개념에 대해서 살펴보자. 엄밀한 정의는 다음절에 나온다. 가장 기본적으로 인증된 키 교환 프로토콜은 세션키에 대한 비밀성을 보장해야 한다. 이런 안전성 개념은 어떤 공격에 대해서 안전한지를 말해야 한다. 가장 최소한의 세션키 비밀성은 수동적 도청만 가능한 공격자에 대해 세션키의 비밀성을 유지해야 한다는 것이다. 좀 더 강한 개념으로 키 독립성(Key Independence)이 있다. 이는 세션키들이 공격자에게는 독립적으로 보여야 한다는 것을 의미하며, Denning-Sacco 공격[13]에 대해 안전해야 한다는 것을 말한다. 마지막으로 전방위 안전성(forward secrecy)을 말할 수 있다. 이 개념은 공격자의 간섭없이 만들어진 세션키는 사용자의 비밀키를 가진 공격자라고 할지라도 세션키를 알 수 없다는 것을 말한다.

보통의 네트웍에서는 통신을 할 때 사용자들이 동시에 서로 메시지를 주고받을 수 있다. 이런 동시 메시지 전송 기능을 이용하면 좀 더 효율적인 라운드 복잡도를 가지는 프로토콜을 설계할 수 있다. 특히나 그룹 키 교환 스킴을 설계할 때 동시 메시지 전송을 가정하면서 설계된다. 최근에는 효율적인 양자간 키 교환 스킴을 설계할 때 이 동시 메시지 전송 기능을 이용했다[16]. 우리는 [16]의 키 교환 모델과 스킴을 키 그래프로 확장시킨다.

우리는 두개의 키 그래프를 위한 키 교환 프로토콜인 *PKA*1와 *PKA*2을 제안한다. 이 프로토콜들은 난수 재사용 테크닉을 사용해서 설계되었으며, 프로토콜의 안전성은 decisional Diffie-Hellman 가정에 기반한다. 우리의 첫 번째 프로토콜인 *PKA*1은 키 독립성을 제공하지만 전방위 안전성은 제공하지 않는다. 만약 *PKA*1이 양자간 키 교환 프로토콜로 사용되어지면, 이 스킴은 [16]에 있는 *TS*1과 비교할 수 있다. *TS*1 역시 1-라운드 프로토콜이며 키 독립성을 보장한다. 하지만 *TS*1의 안전성은 랜덤 오라클 모델에서 증명되는데 비해 우리가 제안하는 프로토콜은 랜덤오라클을 가정하지 않

고서 증명된다. 우리의 두 번째 프로토콜 $PKA2$는 1-라운드 프로토콜로서 전방위 안전성을 제공하는데 비해서 키 독립성은 제공하지 않는다.

2. 기본적인 암호학적 도구들

이 절에서는 우리가 키 교환 프로토콜을 만드는데 사용하는 기본적이면서도 널리 사용되는 암호학 도구들의 정의에 대해서 다시 살펴본다.

2.1 의사난수 함수

$F: Keys(F) \times D \rightarrow R$를 어떤 함수 집합이라고 하고, $Rand^{D \rightarrow R}$를 정의구역을 D로 가지고 R을 공변역으로 가지는 모든 함수들의 집합이라고 하자. F가 의사난수 함수 집합임을 보이기 위해서 우리는 다음과 같은 실험을 고려한다.

$\mathrm{Exp}_{F,A}^{prf-1}$	$\mathrm{Exp}_{F,A}^{prf-0}$
$K \leftarrow Keys(F)$	$g \leftarrow Rand^{D \rightarrow R}$
$d \leftarrow A^{F_K(\cdot)}$	$d \leftarrow A^{g(\cdot)}$
return d	return d

공격자 A의 어드벤티지는 다음과 같이 정의된다:

$$Adv_{F,A}^{prf} = pr[Exp_{F,A}^{prf-1} = 1] - pr[Exp_{F,A}^{prf-0} = 1].$$

어드벤티지 함수는 다음과 같이 정의된다:

$$Adv_F^{prf}(k, t, q, \mu) = max_A\{Adv_{F,A}^{prf}\}.$$

여기서 A는 시간 복잡도가 t이고, 최대 q번의 오라클 쿼리를 수행하며, 총 쿼리들의 합이 μ 비트를 넘지 않는 임의의 공격자이다. 만약 어드벤티지 함수가 네글리져블(negligible)*하면, F는 의사난수 함수 집합이다.

2.2 DDH(Decisional Diffie-Hellman) 문제

GG를 그룹 오더가 q인 그룹 Θ와 그룹 생성자 g를 생성하는 그룹 생성 알고리즘이라고 하자. k를 안전성 패러미터라고 할 때 다음의 실험을 생각해 보자.

$Exp_A^{DDH}(k)$
$(\Theta, q, g) \leftarrow GG(k)$
$(u_1, u_2, w) \leftarrow [1, q]$
$U_1 \leftarrow g^{u_1}$; $U_2 \leftarrow g^{u_2}$; $d \leftarrow [0, 1]$
if $d = 1$, $W \leftarrow g^{u_1 u_2}$
else $W \leftarrow g^{w}$
$d' \leftarrow A(U_1, U_2, W)$

* 어떤 함수 f가 네글리져블하면, f는 충분히 큰 값들에 대해, 모든 다항함수(polynomial)의 역수보다 더 빨리 감소한다. 즉, $f(n) < \frac{1}{n^c}$ for n>N, for all constant c.

공격자 A의 어드벤티지는 다음과 같이 정의된다:

$$Adv_A^{DDH} = 2pr[d = d'] - 1.$$

어드벤티지 함수는 다음과 같이 정의된다:

$$Adv^{DDH}(k, t) = max_A\{Adv_A^{DDH}\}.$$

여기서 A는 시간 복잡도가 t이다. DDH(Decisional Diffie-Hellman) 가정이란 어드벤티지 함수가 네글리져블하다는 것이다.

2.3 전자서명

전자서명 스킴 S는 세가지 알고리즘(S.key, S.gen, S.ver)로 구성된다. S.key는 사용자를 위해서 개인키-공개키 쌍을 만드는 알고리즘이며, S.gen는 개인키로 메시지를 위한 전자서명을 만드는 알고리듬이다. S.ver는 메시지와 전자서명과 공개키를 가지고서 전자서명이 올바른가를 확인하는 알고리듬이다. k를 안전성 패러미터라고 할 때, 다음의 실험을 생각해 보자.

$Exp_{S,A}^{SUF}(k)$
$(sk, pk) \leftarrow S.key$
$w \leftarrow A^{S.gen_{sk}(\cdot)}(pk)$
if $w = \perp$, then return 0
else parse w as (M, σ)
if $S.ver_{pk}(M, \sigma) = 1$ and $S.gen_{sk}(\cdot)$ never returned σ on input M, then return 1
else return 0

공격자 A의 어드벤티지는 다음과 같이 정의된다:

$$Adv_{S,A}^{SUF} = pr[Exp_A^{SUF}(k) = 1].$$

어드벤티지 함수는 다음과 같이 정의된다:

$$Adv_S^{SUF}(k, t, q_s) = max_A\{Adv_{S,A}^{SUF}\}.$$

여기서 A는 시간 복잡도가 t이며, 최대 q_s번의 오라클 쿼리를 던진다. 만약 어드벤티지 함수가 네글리져블하면, 전자서명 스킴 S는 SUF 안전성을 보장한다.

3. 키교환의 안전성 모델

우리는 [10]에서 정의된 키 교환 모델을 확장하여 키 그래프를 위한 키 교환 모델을 만든다. 우리는 키교환 시스템에서의 사용자 수를 N이라 놓고, 각각의 사용자 ID을 P_i로 놓는다. 각 사용자는 공개키-개인키 쌍을 가지고 있으며 키 교환

프로토콜에서 인증을 위해서 사용한다. 오라클 Π_i^k는 P_i의 k 번째 인스턴스를 의미한다. 키 교환 프로토콜이 종료하면 Π_i^k는 세션키 sk_i^k를 생성한다. 세션 식별자 sid_i^k는 세션들 중에서 하나의 세션을 의미할 수 있는 스트링이다. sid_i^k는 Π_i^k가 세션에서 보는 메시지들의 결합이라고 정의할 수 있다. 메시지들의 결합은 사용자들의 ID의 사전순서(lexicographic order)에 의해서 결합할 수 있다. (이 논문에서 우리는 통신하는 두 파티가 동시에 메시지를 보낼 수 있으므로, 시간의 흐름상 먼저 나타나는 순서에 따라서 메시지를 결합 순서를 정할 수 없다.) 세션 sid_i^k에 참여하는 사용자들의 집합을 $V_{\Pi_i^k}$라고 표기하자. 만약 $V_{\Pi_i^k} = V_{\Pi_j^l}$이고 $sid_i^k = sid_j^l$이면 Π_i^k와 Π_j^l은 *파트너*라고 한다. 세션 sid_i^k에 해당하는 키 그래프 $G_{\Pi_i^k}$는 $V_{\Pi_i^k}$와 $E_{\Pi_i^k}$로 이루어지며, $E_{\Pi_i^k}$는 다음과 같이 정의된다: $E_{\Pi_i^k} = \{(i,j) | P_i, P_j \in V_{\Pi_i^k} \wedge P_i$ 와 P_j는 키를 만든다.$\}$

$sk_{\Pi_i^k}^e$는 에지 $e=(i,j)$를 위해서 Π_i^k가 만드는 세션키를 의미한다. 모든 프로토콜은 다음을 만족해야 한다: 만약 Π_i^k와 Π_j^l이 파트너이면, $sk_{\Pi_i^k}^e$와 $sk_{\Pi_j^l}^e$가 같아야 한다.

안전성 개념을 정의하기 위해서 우리는 먼저 공격자의 능력을 정의한다. 우리는 공격자가 일련의 오라클 쿼리(query)를 통해서 통신상에서 흐르는 모든 메시지의 흐름을 제어한다고 가정한다. 공격자의 성공 확률인 어드벤티지를 정의하기 위해서 우리는 실험을 수행한다. 실험은 셋업과 인터액션 두 단계로 나뉜다.

셋업 단계에서는 공격자가 내부 공격자를 결정하며, 내부 공격자들의 공개키/개인키를 다음과 같은 과정으로 만든다:

(1) 사용자들의 수를 N이라고 하자. 실험자는 사용자 P_i을 위한 공개키/개인키 쌍 (y_i, x_i)을 만들어서 모든 공개키를 공격자에게 준다.

(2) 공격자는 공개키를 보고서 내부 공격자를 고른 후, 내부 공격자를 위한 공개키/개인키를 결정해서 실험자에게 보낸다. 우리는 내부 공격자의 집합을 I라고 표현하고 정직한 사용자들의 집합을 H라고 표기한다.

(3) 실험자는 내부 공격자의 공개키/개인키를 체크한다. 만약 공개키와 개인키가 일치하지 않으면 실험을 중단한다.

인터액션 단계에서는 공격자가 오라클 쿼리를 실험자에게 던지며, 실험자는 이에 답변하는 방식으로 진행된다. 오라클 쿼리들은 현실 세계에서 공격자가 행하는 공격을 모델링한다. 우리는 이 논문에서 다음과 같은 쿼리들을 고려한다:

-Initiate(i,G): 이 쿼리는 P_i로 하여금 V안에 있는 사용자들과 키 교환 프로토콜을 수행하게 한다. P_i는 이 쿼리의 응답으로서 키 교환 프로토콜의 첫 번째 메시지를 공격자에게 돌려준다.

-Send(i,k,M): 이 쿼리는 메시지 M을 오라클 Π_i^k에게 보낸다. Π_i^k는 M을 받고서 프로토콜에 따라서 반응한다. 이 쿼리를 통해서 현실 세계에서의 능동적 공격(active attack)을 모델링할 수 있다.

-Execute(G): 이 쿼리는 공격자와 실험자간에 주고받는 대화로 이루어진다. 사용자들의 집합 V를 두개의 그룹 $V_I (\in I)$와 $V_H (\in H)$로 나눈다. 키 교환 프로토콜이 n 라운드라고 하자. 실험자가 먼저 V_H들의 첫 번째 라운드 메시지를 만들어서 공격자에게 보낸다. 그러면 공격자는 V_I들의 첫 번째 라운드 메시지를 만들어서 실험자에게 보낸다. 실험자와 공격자는 남은 라운드 동안 이 과정을 반복한다. 만약 $V_I = \phi$면, 공격자가 V안에 있는 사용자들끼리의 키 교환을 단순히 도청하는 것을 의미하며, 이것은 기존의 내부 공격자를 고려하지 않는 키 교환에서의 Execute 쿼리 정의와 일치한다(비록 Execute 쿼리는 Initiate 쿼리와 Send 쿼리를 사용해서 시뮬레이션 될 수 있으나, 이 쿼리는 전방위 안전성 정위를 위해서 필요하다).

-Reveal(i,k,e): $e \in E_{\Pi_i^k}$이어야 하며, 이 쿼리는 기존 세션의 세션키가 알려질 경우를 모델링한다. 즉, Denning-Sacco 공격을 모델링한다. 이 쿼리의 응답으로 세션키 $sk_{\Pi_i^k}^e$를 돌려준다.

-Corrupt(i): 이 쿼리는 사용자 P_i의 비밀키가 노출되었을 경우를 모델링한다. 하지만 공격자는 P_i의 비밀키는 알 수 있더라도 공격자의 행동까지 컨트롤하지는 못한다. 물론 공격자는 이 비밀키를 가지고서 다른 사용자에게 P_i인척 할 수 있다(impersonation attack).

-Test(i,k,e): $e \in E_{\Pi_i^k}$이어야 하며, 이 쿼리는 공격자의 성공 능력을 정의하기 위해서 사용한다. 만약 오라클 Π_i^k가 e-프레쉬 (아래에서 정의됨) 하다면, 동전 던지기를 수행한다. 동전 던지기의 결과를 b라고 하자. 만약 b가 1이면 세션키 $sk_{\Pi_i^k}^e$를 공격자에게 돌려주고, 아니면 그냥 임의의 값을 키 공간에서 뽑아서 돌려준다. 공격자는 하나의 Test 쿼리를 던질 수 있다. 이제 프레쉬 오라클을 정의한다.

[정의 1] 실험이 끝날 때까지 다음의 조건들을 만족하는 오라클 Π_i^k는 e-프레쉬하다.

(a) 모든 $j \in e$에 대해서, P_j는 내부 공격자가 아니다.

(b) 모든 $j \in e$에 대해서, 공격자는 Corrupt(j) 쿼리를 하지 않았다.

(c) 만약 Π_j^l와 Π_i^k가 파트너라면, 공격자가 Reveal(i,k,e)나 Send(j,l,e) 쿼리를 하지 않았다.

공격자는 실험이 끝날 때 b′을 출력한다. 공격자의 공격 성공 확률은 어드벤티지로 정의되며 다음과 같다:

$$Adv_A(k) = |2Pr[b' = b] - 1|.$$

만약 모든 다항시간 공격자의 어드벤티지가 네글리저블(negligible)하다면 이 프로토콜은 안전한 프로토콜이다. 다음

은 키 교환 프로토콜이 얻을 수 있는 안전성들의 종류다.

(1) 키 안전성(Implicit Authentication) 프로토콜은 기본적으로 Reveal이나 Corrupt 쿼리를 허용하지 않을 때 안전해야 한다.

(2) 키 독립성(Key Independence) : 공격자는 Reveal 쿼리는 할 수 있으나 Corrupt 쿼리는 할 수 없다. 이런 공격자들에 안전한 프로토콜은 키 독립성을 보장한다고 한다.

(3) 전방위 안전성(Forward Secrecy) : 공격자는 Reveal 쿼리는 할 수 없으나 Corrupt 쿼리는 할 수 있다. 이 때에는 프레쉬 정의에 나오는 조건 (b)가 다음처럼 바뀌어야 한다 : 공격자는 $j \in e$에 대해서 Corrupt(j)를 하지 않았거나, 또는 $j \in e$에 대해서 Send(j,l,*) 쿼리를 하지 않아야 한다*(따라서 대신에 Execute 쿼리를 사용해야 한다). 이런 공격자들에 안전한 프로토콜은 전방위 안전성을 보장한다고 한다.

프로토콜 P에 대해 공격자가 할 수 있는 공격 종류에 따라서 어드벤티지를 다음과 같이 표기한다 :

$$Adv_P^{XX}(k,t) = \max_A \{Adv_A^{XX}(k)\}.$$

여기서 XX는 IA(키 안전성), KI(키 독립성), FS (전방위 안전성)들 중에 하나이다. k는 안전성 패러미터이며, t는 공격자에게 허용되는 시간이다. 만약 모든 다항시간 t=poly(k)에 대해서 $Adv_P^{XX}(k,t)$가 네글리져블하다면 P는 XX-안전성을 보장한다고 말한다.

4. 1-라운드 키교환 프로토콜

우리가 제시하는 모든 프로토콜들은 다음을 가정한다. 사용자들은 그들의 ID에 의해서 순서대로 정렬될 수 있으며, 만약 P_i가 P_j에 앞설 경우 $P_i < P_j$로 표기한다. k는 안전성 패러미터이며, Θ는 소수 q를 오더(order)로 가지는 그룹이며, g는 그룹의 제너레이터(generator)이다. 각각의 사용자 P_i는 공개키-개인키 쌍 $(y_i = g^{x_i}, x_i)$를 가진다.

4.1 프로토콜 PKA1

Setup : F는 의사 난수 함수이며, $G = \{V, E\}$은 키 그래프이다.

Round 1 : $P_i (\in V_{\Pi_i^k})$는 난수 r_i를 k비트 스트링 공간에서 랜덤하게 뽑아서 브로드캐스트한다.

세션키 계산 : P_i는 세션식별자 sid_i^k를 만들기 위해서, 브로드캐스트된 메시지를 ID 순서대로 결합한다. P_i는 $(i,j) \in E_{\Pi_i^k}$를 위한 세션키 $sk^{(i,j)} = F_{y_j^{x_i}}(sid_i^k)$를 만든다.

PKA1의 실행 예가 (그림 1)에 나와 있다.

$G = \{V, E\}$ $V = \{P_1, P_2, P_3, P_4\}$

$E = \{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$

	$P_1(G)$	$P_2(G)$	$P_3(G)$	$P_4(G)$
Round 1	r_1	r_2	r_3	r_4

$sid = r_1 \| r_2 \| r_3 \| r_4$

$sk^{(1,2)} = F_{g^{x_1 x_2}}(sid), sk^{(1,3)} = F_{g^{x_1 x_3}}(sid), sk^{(1,4)} = F_{g^{x_1 x_4}}(sid)$

$sk^{(2,3)} = F_{g^{x_2 x_3}}(sid), sk^{(2,4)} = F_{g^{x_2 x_4}}(sid), sk^{(3,4)} = F_{g^{x_3 x_4}}(sid)$

(그림 1) PKA1의 실행 예

다음의 정리는 PKA1의 안전성에 관한 것이다.

[정리 1] 만약 Θ가 DDH 문제에 대해서 안전성을 보장하고 F가 의사난수 함수이면, PKA1은 KI (키 독립성)을 보장한다. 구체적으로,

$$Adv_{PKA1}^{KI}(k,t,q_{rs}) \le N_H(N_H-1)Adv^{DDH}(k,t) + \frac{N_H(N_H-1)}{2} Adv^{PRF}(k,t,q_s,kNq_s) + \frac{(N_H q_s)^2}{2^k}$$

$C_1 = \{1,2\}, C_2 = \{1,3\}, \ldots,$

$C_{\frac{N_H(N_H-1)}{2}} = \{\frac{N_H(N_H-1)}{2} - 1, \frac{N_H(N_H-1)}{2}\}$ 이다.

여기서 t는 공격자의 실행시간을 포함한 총 실험시간이며, q_{rs}는 Reveal 쿼리의 수이다. N은 실험에서의 사용자 수이며, N_H는 정직한 사용자의 수이고, q_s는 실험에서의 최대 세션의 수이다. DDH 문제에 대한 (t,ϵ) 안전성이란 어떤 공격자도 시간 t안에 ϵ 이상의 확률로 DDH문제를 풀 수 없다는 가정이다.

[증명] 표기를 쉽게 하기 위해서 $H = \{P_1, \ldots, P_{N_H}\}$이고 $I = \{P_{N_H+1}, \ldots, P_N\}$라고 가정하자. 숫자들의 집합 $\{1, \ldots, N_H\}$의 2개의 원소로 이루어지는 조합들을 사전적 순서로 늘어 놓으면 다음과 같이 된다 : 우리는 다음처럼 게임들 $Game_0, \ldots, Game_{\frac{N_H(N_H-1)}{2}}$을 정의한다. 우리는 사건 A가 $Game_t$에서 일어날 확률을 $pr_t[A]$라고 표기한다. $Game_t$에서의 공격자 A의 어드벤티지는 $Adv_{t,A}$로 표기한다. 정리 증명은 표준적인 하이브리드 아귀먼트를 이용해서 증명된다. $Game_0$는 공격자의 어드벤티지를 정의하는 실험과 같다 :

$Game_0$

1. H에 있는 사용자들을 위한 공개키를 선택한다. 즉, $1 \le i \le N_H$에 대해서, 난수 x_i를 선택한 다음, P_i의 공개키를 g^{x_i}로 한다.
2. H에 있는 모든 공개키를 A에게 준다.

* 이 정의는 논문들에서 사용되는 전방위 안전성의 두 가지 정의들 중 약한 것이다. 더 강한 전방위 안전성 정의에서는 만약 공격자가 Corrupt(j) 쿼리를 한 후에, Send(j,l,*) 쿼리를 하지 않으면 프레쉬한 오라클로 간주된다.

3. A로부터 I에 있는 모든 내부 공격자들의 공개키/개인키 쌍을 받아서 올바른가를 검사한다. 만약 올바르지 않다면 실험을 끝낸다.
4. k번째 조합 $C_k=\{i,j\}$ $(1\le k\le \frac{N_H(N_H-1)}{2})$에 대해서,
   -세션키를 만들기 위해서 $g^{x_ix_j}$를 사용하라. 즉, $sk^{(i,j)}=F_{g^{x_ix_j}}(sid)$
5. 두 사용자 중에 한 사용자가 H에 있고, 다른 사용자가 I에 있다면, $sk^{(i,j)}=F_{g^{x_ix_j}}(sid)$.
6. 다른 모든 오라클 쿼리들에 대해서는 프로토콜에 명시된대로 응답한다.

$Game_t$ $(1\le t\le \frac{N_H(N_H-1)}{2})$는 다음과 같다:

$Game_t$

1. H에 있는 사용자들을 위한 공개키를 선택한다. 즉, $1\le i\le N_H$에 대해서, 난수 x_i를 선택한 다음, P_i의 공개키를 g^{x_i}로 한다.
2. H에 있는 모든 공개키를 A에게 준다.
3. A로부터 I에 있는 모든 내부 공격자들의 공개키/개인키 쌍을 받아서 올바른가를 검사한다. 만약 올바르지 않다면 실험을 끝낸다.
4. k번째 조합 $C_k=\{i,j\}$ $(1\le k\le t)$에 대해서,
   -난수 $r_{i,j}$를 선택해서, 세션키를 만들기 위해서 $r_{i,j}$를 사용하라. 즉, $sk^{(i,j)}=F_{r_{i,j}}(sid)$.
5. k번째 조합 $C_k=\{i,j\}$ $(t+1\le k\le \frac{N_H(N_H-1)}{2})$에 대해서,
   -세션키를 만들기 위해서 $g^{x_ix_j}$를 사용하라. 즉, $sk^{(i,j)}=F_{g^{x_ix_j}}(sid)$.
6. 두 사용자 중에 한 사용자가 H에 있고, 다른 사용자가 I에 있다면, $sk^{(i,j)}=F_{g^{x_ix_j}}(sid)$.
7. 다른 모든 오라클 쿼리들에 대해서는 프로토콜에 명시된대로 응답한다.

우리는 먼저 다음의 클레임에서 이웃한 두 게임 $Game_{t-1}$과 $Game_t$에서의 공격자 어드벤티지 차이는 네글리져블함을 보인다.

Claim (1). $Adv^{PKA1}_{t-1,A}-Adv^{PKA1}_{t,A}\le 2Adv^{DDH}$ $(1\le t\le \frac{N_H(N_H-1)}{2})$.

다음으로 우리는 $Game_{\frac{N_H(N_H-1)}{2}}$에서의 모든 공격자들의 어드벤티지는 네글리져블함을 보인다.

Claim (2). $Adv^{PKA1}_{\frac{N_H(N_H-1)}{2},A}\le \frac{N_H(N_H-1)}{2}Adv^{prf}_F+\frac{(N_Hq_s)^2}{2^k}$.

Claim (1)으로부터 표준적인 하이브리드 아귀먼트를 사용해서, 다음의 수식을 얻는다:

$$Adv^{PKA1}_{0,A}-Adv^{PKA1}_{\frac{N_H(N_H-1)}{2},A}\le N_H(N_H-1)Adv^{DDH}.$$

Claim (2)와 위의 식으로부터 우리는 다음처럼 정리를 증명할 수 있다:

$$Adv^{PKA1}_{0,A}\le N_H(N_H-1)Adv^{DDH}+\frac{N_H(N_H-1)}{2}Adv^{prf}_F+\frac{(N_Hq_s)^2}{2^k}.$$ ❑

(Proof of Claim (1))

우리는 게임 $Game_{t-1}$과 $Game_t$에서의 공격자 A의 어드벤티지 차이를 가지고서 decisional Deffie-Hellman 문제를 푸는 알고리즘 $D_{t-1,t}$를 만들 수 있음을 보인다. $D_{t-1,t}$는 입력값으로 (U_1, U_2, W)를 받는다. $C_t=\{i',j'\}$이라고 하면, $D_{t-1,t}$는 U_1과 U_2를 $P_{i'}$과 $P_{j'}$의 공개키로 사용하고, W를 세션키를 만드는데 사용한다. 즉, 세션키는 $sk^{(i',j')}=F_W(sid)$이다. 결과적으로 $D_{t-1,t}$는 W가 Diffie-Hellman 값인지 아닌지에 따라서 게임 $Game_{t-1}$ 또는 $Game_t$을 공격자 A에게 시뮬레이션하게 된다. 우리는 $D_{t-1,t}$를 자세히 기술한다:

$D_{t-1,t}(U_1, U_2, W)$

1. t번째 조합 $C_t=\{i',j'\}$에 대해서,
   -U_1과 U_2를 $P_{i'}$과 $P_{j'}$의 공개키로 한다.
   -W를 세션키 $sk^{(i',j')}$를 만드는데 사용한다. 즉, $sk^{(i',j')}=F_W(sid)$.
2. H에 있는 사용자들을 위한 공개키를 선택한다. 즉, $1\le i\le N_H(i\ne i',j')$에 대해서, 난수 x_i를 선택한 다음, P_i의 공개키를 g^{x_i}로 한다.
3. H에 있는 모든 공개키를 A에게 준다.
4. A로부터 I에 있는 모든 내부 공격자들의 공개키/개인키 쌍을 받아서 올바른가를 검사한다. 만약 올바르지 않다면 알고리듬을 종료한다.
5. k번째 조합 $C_k=\{i,j\}$ $(1\le k\le t)$에 대해서,
   -난수 $r_{i,j}$를 선택해서, 세션키를 만들기 위해서 $r_{i,j}$를 사용하라. 즉, $sk^{(i,j)}=F_{r_{i,j}}(sid)$.
6. k번째 조합 $C_k=[i,j]$ $(t+1\le k\le \frac{N_H(N_H-1)}{2})$에 대해서,
   -만약 $i'\in C_k$ 또는 $j'\in C_k$이면, 세션키로 $U_1^{x_o}$ 또는 $U_2^{x_o}$를 사용한다. 여기서 x_o는 C_k 안에 있는 다른 사용자의 비밀키이다.

-만약 $i', j' \notin C_k$이면, 세션키를 만들기 위해서 $g^{x_i x_j}$를 사용하라. 즉, $sk^{(i,j)} = F_{g^{x_i x_j}}(sid)$.

7. 다른 모든 오라클 쿼리들에 대해서는 프로토콜에 명시된대로 응답한다.
8. test 쿼리에서의 동전 던지기의 결과를 b라 하고, A의 출력값을 b'이라고 하자. 만약 b=b'이면, $D_{t-1,t}$는 1을 출력하고 끝낸다. 그렇지 않으면 $D_{t-1,t}$는 0을 출력하고 끝낸다.

$D_{t-1,t}$의 어드벤티지는 다음과 같다:

$$\begin{aligned} Adv^{DDH}_{D_{t-1,t}} &= pr[U_1 \leftarrow g^{u_1}; U_2 \leftarrow g^{u_2}; W \leftarrow g^{u_1 u_2} : D_{t-1,t}(U_1, U_2, W) \\ &\quad = 1] - pr[U_1 \leftarrow g^{u_1}; U_2 \leftarrow g^{u_2}; W \leftarrow g^{w} : D_{t-1,t}(U_1, U_2, W) = 1] \\ &= pr_{t-1}[b = b'] - pr_t[b = b'] \\ &= 1/2\,(2pr_{t-1}[b = b'] - 1) - (2pr_t[b = b'] - 1)) \\ &= 1/2\,(Adv^{PKA1}_{t-1,A} - Adv^{PKA1}_{t,A}) \end{aligned}$$

이것으로부터 Claim (1)이 증명된다. ❑

(Proof of Claim (2))

우리는 $Game_{\frac{N_H(N_H-1)}{2}}$에서의 공격자 A의 어드벤티지를 가지고서 의사난수 함수 F를 공격하는 공격자 D를 만들 수 있음을 보인다. D는 의사 난수 함수 집합 F에 대한 어드벤티지를 측정하기 위한 실험에서 주어진 오라클 f를 이용할 수 있다. D는 임의의 조합 $C_t = \{i', j'\}$을 선택한 다음 $P_{i'}$와 $P_{j'}$ 사이의 세션키를 만들기 위해서 f를 이용한다. 즉, $sk^{(i',j')} = f(sid)$이다. 결과적으로 D는 f가 의사난수 함수 F에서 뽑힌건지 아닌지에 따라, Exp^{prf-1}_F이거나 Exp^{prf-0}_F을 공격자 A에게 시뮬레이션 하게 된다. 만약 A가 P_i와 P_j 사이의 세션키에 대해서 Test 쿼리를 던지게 되면 D는 이것을 이용해서 F를 공격할 수 있다.

$D^{f(\cdot)}$

1. H에 있는 사용자들을 위한 공개키를 선택한다. 즉, $1 \le i \le N_H$에 대해서, 난수 x_i를 선택한 다음, P_i의 공개키를 g^{x_i}로 한다.
2. H에 있는 모든 공개키를 A에게 준다.
3. A로부터 I에 있는 모든 내부 공격자들의 공개키/개인키 쌍을 받아서 올바른가를 검사한다. 만약 올바르지 않다면 알고리듬을 종료한다.
4. 먼저 $t \leftarrow [1, \frac{N_H(N_H-1)}{2}]$를 랜덤하게 선택한다. t번째 조합 $C_t = \{i', j'\}$에 대해서,
   -$P_{i'}$와 $P_{j'}$ 사이의 세션키를 만들기 위해서 f를 사용하라. 즉, $sk^{(i',j')} = f(sid)$.
5. k번째 조합 $C_k = \{i, j\}$ $(1 \le k(\neq t) \le \frac{N_H(N_H-1)}{2})$에 대해서,
   -난수 $r_{i,j}$를 선택해서, 세션키를 만들기 위해서 $r_{i,j}$를 사용하라. 즉, $sk^{(i,j)} = F_{r_{i,j}}(sid)$.
6. 만약 $P_i \in H$이고 $P_j \in I$이면, $sk^{(i,j)} = F_{g^{x_i x_j}}(sid)$.
7. 다른 모든 오라클 쿼리들에 대해서는 프로토콜에 명시된대로 응답한다.
8. A의 출력값을 b'이라고 하면, $D^{f(\cdot)}$는 b'을 출력하고 종료한다.

공격자 A는 세션 식별자 sid가 중복되면 거기서 어드벤티지를 얻을 수도 있다. 하지만 세션 식별자가 중복할 확률은 $\frac{(N_H q_s)^2}{2^k}$임을 쉽게 알 수 있다. col을 세션 식별자가 중복되는 사건이라고 하자. 그러면 $pr_{\frac{N_H(N_H-1)}{2}}[col] \le \frac{(N_H q_s)^2}{2^k}$이 된다. 이런 경우를 제외한 경우의 공격자 A의 어드벤티지는 다음과 같이 D의 공격 성공 확률과 관련된다:

$$\begin{aligned} Adv^{prf}_D &= pr[K \leftarrow Keys(F) : D^{F_K(\cdot)} = 1] \\ &\quad - pr[g \leftarrow Rand^{D \to R} : D^{g(\cdot)} = 1] \\ &= \frac{2}{N_H(N_H-1)} pr_{\frac{N_H(N_H-1)}{2}}[b = 1 : A() = 1 \wedge \overline{col}] \\ &\quad - \frac{2}{N_H(N_H-1)} pr_{\frac{N_H(N_H-1)}{2}}[b = 0 : A() = 1 \wedge \overline{col}] \\ &\ge \frac{2}{N_H(N_H-1)} (pr_{\frac{N_H(N_H-1)}{2}}[b = 1 : A() = 1] \\ &\quad - pr_{\frac{N_H(N_H-1)}{2}}[b = 0 : A() = 1] - pr_{\frac{N_H(N_H-1)}{2}}[col]) \\ &= \frac{2}{N_H(N_H-1)} (Adv^{PKA1}_{\frac{N_H(N_H-1)}{2},A} - pr_{\frac{N_H(N_H-1)}{2}}(col)) \\ &\ge \frac{2}{N_H(N_H-1)} (Adv^{PKA1}_{\frac{N_H(N_H-1)}{2},A} - \frac{(N_H q_s)^2}{2^k}) \end{aligned}$$

이것으로부터 Claim (2)가 증명된다. ❑

4.2. 프로토콜 PKA2

Setup: S는 전자 서명 스킴이며, $G = \{V, E\}$은 키 그래프이다.

Round 1: $P_i (\in V_{\Pi_i^k})$는 난수 α_i를 $[1, q]$에서 뽑은 다음 $\sigma_i = S.gen(g^{\alpha_i})$을 만들어서 $g^{\alpha_i} \| \sigma_i$를 브로드캐스트한다.

세션키 계산: P_i는 받은 전자서명들이 올바른가를 확인한다. P_i는 $(i,j) \in E_{\Pi_i^k}$를 위한 세션키 $sk^{(i,j)} = (g^{\alpha_j})^{\alpha_i}$를 만든다. PKA2의 실행 예가 그림 2에 나와 있다.

$G = \{V, E\}$ $V = \{P_1, P_2, P_3, P_4\}$

$E = \{(1,2), (1,3), (1,4), (2,3), (2,4), (3,4)\}$

	$P_1(G)$	$P_2(G)$	$P_3(G)$	$P_4(G)$
Round1	$g^{\alpha_1}\|\|\sigma_1$	$g^{\alpha_2}\|\|\sigma_2$	$g^{\alpha_3}\|\|\sigma_3$	$g^{\alpha_4}\|\|\sigma_4$

$$\sigma_i = S.gen(g^{\alpha_i})$$

$$sk^{(1,2)} = g^{\alpha_1\alpha_2}, sk^{(1,3)} = g^{\alpha_1\alpha_3}, sk^{(1,4)} = g^{\alpha_1\alpha_4}$$
$$sk^{(2,3)} = g^{\alpha_2\alpha_3}, sk^{(2,4)} = g^{\alpha_2\alpha_4}, sk^{(3,4)} = g^{\alpha_3\alpha_4}$$

(그림 2) PKA2의 실행 예

다음의 정리는 PKA2의 안전성에 관한 것이다.

[정리 2] 만약 Θ가 DDH 문제에 대해서 안전성을 보장하고 S가 전자 서명 스킴이면, PKA2은 FS (전방위 안전성)을 보장한다. 구체적으로,

$$Adv^{FS}_{PKA2}(k,t,q_{rs},q_{rl}) \le N \cdot Adv^{SUF}_{S}(k,t,q_s) + q_s N^2 \cdot Adv^{DDH}(k,t) + \frac{(N_H q_s)^2}{2^k}$$ 이다.

여기서 t는 공격자의 실행시간을 포함한 총 실험시간이며, q_{rl}는 Corrupt 쿼리의 수이다. N은 실험에서의 사용자 수이며, N_H는 정직한 사용자의 수이고, q_s는 실험에서의 최대 세션의 수이다.

[증명] PKA2를 공격하는 공격자 A가 높은 확률로 공격에 성공한다고 가정하자. 공격자 A는 전송되는 메시지들이 중복되면 거기서 어드벤티지를 얻을 수도 있다.

하지만 전송되는 메시지들이 중복될 확률, 즉 sid가 중복할 확률은 $\frac{(N_H q_s)^2}{2^k}$임을 쉽게 알 수 있다. 이런 경우를 제외한 경우의 공격자 A의 어드벤티지는 DDH 문제를 풀거나 서명을 위조하는 데 사용될 수 있음을 보인다. col을 sid가 중복되는 사건이라고 하고, forgery를 적어도 하나의 위조된 서명이 실험에 나타나는 사건이라고 하자.

공격자 A의 성공 확률은 다음과 같이 구분될 수 있다:

$$\begin{aligned} Adv^{FS}_{PKA2,A} &= pr[b=1:A()=1] - pr[b=0:A()=1] \\ &\le pr[col] + pr[\overline{col} \wedge forgery] \\ &\quad + pr[b=1:A()=1 \wedge \overline{col} \wedge \overline{forgery}] \\ &\quad - pr[b=0:A()=1 \wedge \overline{col} \wedge \overline{forgery}]. \end{aligned}$$

우리는 $pr[col] \le \frac{(N_H q_s)^2}{2^k}$임을 쉽게 알 수 있다.

Claim (3). $pr[\overline{col} \wedge forgery] \le N \cdot Adv^{SUF}_{S}$.

Claim (4). $pr[b=1:A()=1 \wedge \overline{col} \wedge \overline{forgery}] - pr[b=0:A()=1 \wedge \overline{col} \wedge \overline{forgery}] \le N_H^2 \cdot Adv^{DDH}$.

Claim (3)과 Claim (4)로부터 우리는 다음처럼 정리를 증명할 수 있다:

$$Adv^{FS}_{PKA2}(k,t,q_{rs},q_{rl}) \le N_H^2 \cdot Adv^{DDH}(k,t) + N Adv^{SUF}_{S}(k,t,q_s) + \frac{(N_H q_s)^2}{2^k}.$$ ❑

이제 우리는 claim들을 증명한다.

(Proof of Claim (3))

우리는 forgery 사건이 발생할 경우에 공격자 A를 사용해서 서명 스킴 S를 공격하는 알고리즘 F를 만들 수 있음을 보인다. 서명에 관한 위조 실험에서 F는 공개키 y가 주어지며 서명 오라클 $S.gen(\cdot)$을 사용할 수 있다. F는 임의로 사용자 한 명을 선택해서 그 사용자의 공개키로 y를 이용하며 서명을 만들기 위해서는 서명 오라클을 사용한다.

$F^{S.gen(\cdot)}(y)$

1. $i' \leftarrow [1,N]$을 선택해서 y를 $P_{i'}$의 공개키로 설정한다. 다른 사용자들의 공개키는 정상적으로 생성한다.
2. 모든 공개키를 A에게 준다.
3. A로부터 I에 있는 모든 내부 공격자들의 공개키/개인키 쌍을 받아서 올바른가를 검사한다. 만약 올바르지 않다면 실험을 끝낸다. 만약 $P_{i'} \in I$이면, 알고리듬을 종료한다.
4. A의 오라클 쿼리들에 대해서 다음처럼 응답한다.
   - -Send와 Execute 쿼리 : $P_{i'}$의 서명을 만들기 위해서 서명 오라클 $S.gen(\cdot)$을 사용한다.
   - -Corrupt(i) 쿼리 : 만약 $P_i = P_{i'}$이면, 알고리듬을 종료한다.
   - -다른 모든 오라클 쿼리들에 대해서는 프로토콜에 명시된 대로 응답한다.
5. 만약 $P_{i'}$의 공개키에 대한 위조된 서명 σ가 발견되면, σ을 출력하고 끝낸다.

F의 성공 확률은 A가 $P_{i'}$의 공개키에 대한 서명을 위조하는 가의 여부에 달려있다. 따라서 F의 어드벤티지는 다음과 같다:

$$Adv^{SUF}_{S,F} \ge \frac{1}{N} \cdot pr[\overline{col} \wedge forgery].$$

이로부터 Claim(3)이 증명된다. ❑

(Proof of Claim (4))

우리는 공격자 A의 어드벤티지를 가지고서 DDH 문제를 푸는 알고리즘 D를 만들 수 있음을 보인다. D는 입력값으로 (U_1, U_2, W)를 받아서 이 입력값들을 DDH 문제의 랜덤 재생성 기법 (random self-reducibility)을 사용해서 프로토콜들

의 메시지로 삽입한다. 표기를 쉽게 하기 위해서 $H=\{P_1,...,P_{N_H}\}$이고 $I=\{P_{N_H+1},...,P_N\}$라고 가정하자.

$D(U_1, U_2, W)$

1. 사용자들을 위한 공개키를 선택한다. 즉, $1 \le i \le N_H$에 대해서, 난수 x_i를 선택한 다음, P_i의 공개키를 g^{x_i}로 한다.
2. H에 있는 모든 사용자의 공개키를 A에게 준다.
3. A로부터 I에 있는 모든 내부 공격자들의 공개키/개인키 쌍을 받아서 올바른가를 검사한다. 만약 올바르지 않다면 실험을 끝낸다.
4. $s_1, s_2 \leftarrow [1, q_s]$을 선택한다.
5. A의 오라클 쿼리들에 대해서 다음처럼 응답한다.
   - Send와 Execute 쿼리 : 만약 s_1번째 세션이면 $i' \leftarrow V$를 $i' \in [1, N_H]$이 되도록 선택한다. 만약 조건을 만족하는 i'이 존재하지 않으면 실험을 끝낸다. 조건을 만족하는 i'이 존재하면 U_1을 $P_{i'}$의 라운드 메시지로 사용한다. 만약 s_2번째 세션이면 $j' \leftarrow V$를 $j' (\neq i') \in [1, N_H]$이 되도록 선택한다. 만약 조건을 만족하는 j'이 존재하지 않으면 실험을 끝낸다. 조건을 만족하는 j'이 존재하면 U_2을 $P_{j'}$의 라운드 메시지로 사용한다.
   - Test(i,k,(i,j)) 쿼리 : 만약 i가 에지 (i,j)를 위한 키를 만들기 위해서 U_1과 U_2를 사용해야 한다면 W를 돌려준다. 그렇지 않다면 알고리즘을 종료한다.
   - 다른 모든 오라클 쿼리들에 대해서는 프로토콜에 명시된 대로 응답한다.

D의 공격 성공 확률은 D가 정확하게 s', i', j'을 추측했는가의 여부에 의해 결정된다. 따라서 D의 어드벤티지는 다음과 같다:

$$Adv_D^{DDH} \ge \frac{1}{q_s^2 N_H^2} \cdot (pr[b=1 : A()=1 \wedge \overline{col} \wedge \overline{forgery}] - pr[b=0 : A()=1 \wedge \overline{col} \wedge \overline{forgery}]).$$

이로부터 Claim(4)가 증명된다.

5. 결 론

우리는 이 논문에서 여러 사용자들과 키 교환이 필요한 경우에 이를 좀 더 효율적으로 할 수 있는 다중 키 교환 프로토콜들을 최초로 제안했으며, 제안되는 프로토콜들의 안전성을 증명했다. 제안되는 프로토콜들의 효율성 비교를 위해서 각각의 사용자들과의 키 교환을 위해서 양자간 키 교환을 사용하는 단순한 프로토콜과 비교한다(<표 1>). 양자간 키 교환 프로토콜이 2-KE라고 하고 이를 반복 사용하는 단순한 다중 키 교환 프로토콜을 SKE 라고 하자. SKE의 라운드 수나 연산량, 전송되는 메시지의 길이가 계산되는 키들의 개수에 비례함을 알 수 있다. 제안되는 프로토콜인 PKA1과 PKA2는 연산량은 계산되는 키들의 개수에 비례하나 전송되는 메시지의 길이는 키들의 개수에 상관없이 일정함을 알 수 있다. SKE가 우리가 제안하는 안전성 모델에서 안전하기 위해서는 2-KE가 내부 공격자들에 대해서도 안전해야 한다. 그러나 기존의 양자간 키 교환 프로토콜들은 내부 공격자를 고려하지 않고 증명되었으므로 우리가 제안하는 내부 공격자를 고려하는 더 강한 모델에서는 안전하지 않을 수 있다. 반면에 우리가 제안하는 프로토콜들은 내부 공격자를 고려하는 환경에서도 안전성을 제공하는 양자간 키 교환 프로토콜로 사용될 수 있다.

<표 1> 사용자가 n 개의 키를 만들 경우의 프로토콜 분석

	SKE	PKA1	PKA2
라운드수	Round(2-KE)	1	1
연산량	n • Time(2-KE)	n(Time(Exp) + Time(PRF))	(n+1) • Time(Exp) + Time(S.gen) + n • Time(S.ver)
브로드캐스팅 메시지 길이	n • Length(2-KE)	k	$2k + \lvert\sigma\rvert$
안전성	-	키 독립성	전방위 안전성
안전성 모델	-	스탠다드 모델	스탠다드 모델

k는 안전성 패러미터로서 소수 p의 길이로 볼 수 있으며, $|\sigma|$는 서명의 길이다.

Time(Exp)는 지수연산에 들어가는 연산량이며, Time(PRF)는 의사난수 함수의 실행시간이다.

Time(S.gen)는 서명을 생성하는데 필요한 시간이며, Time(S.ver)는 서명을 확인하는데 필요한 시간이다.

Round(2-KE)는 2-KE의 라운드 수, Time(2-KE)는 연산량, Length(2-KE)는 전송되는 메시지의 길이이다.

참 고 문 헌

[1] R. Ankney, D. Johnson, and M. Matyas, The Unified Model, Contribution to ANSI X9F1, 1995.

[2] G. Ateniese, M. Steiner, and G. Tsudik, New Multi-Party Authentication Services and Key Agreement Protocols, IEEE Journal of Selected Areas in Communications 18 (4) (2000) 628~639.

[3] C. Boyd, On Key Agreement and Conference Key Agreement, in : Proceedings of ACISP'97, 1997, pp.294~302.

[4] M. Bellare, A. Boldyreva, and J. Staddon, Randomness Re-use in Multi-recipient Encryption Schemes, in : Proceedings of PKC'03, 2003, pp.85~99.

[5] E. Bresson, O. Chevassut, A. Essiari and D. Pointcheval.

Mutual Authentication and Group Key Agreement for Low-Power Mobile Devices, in : Proceedings of MWCN'03, 2003, pp.59~62.

[6] M. Bellare, R. Canetti, and H. Krawczyk, A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols, in : Proceedings of 30th Annual Symposium on the Theory of Computing, 1998, pp.419~428.

[7] E. Bresson, O. Chevassut, and D. Pointcheval, Provably Authenticated Group Diffie-Hellman Key Exchange --- The Dynamic Case, in : Proceedings of ASIACRYPT'01, 2001, pp.290~309.

[8] M. Burmester and Y. Desmedt, A Secure and Efficient Conference Key Distribution System, in : Proceedings of EUROCRYPT'94, 1994, pp.275~286.

[9] C. Boyd and J.M.G. Nieto, Round-Optimal Contributory Conference Key Agreement, in : Proceedings of PKC'03, 2003, pp.161~174.

[10] M. Bellare and P. Rogaway, Entity Authentication and Key Distribution, in : Proceedings of CRYPTO'93, 1993, pp.232~249.

[11] R. Canetti and H. Krawczyk, Universally Composable Notions of Key Exchange and Secure Channels, in : Proceedings of EUROCRYPT'02, 2002, pp.337~351.

[12] W. Diffie and M. Hellman, New Directions in Cryptography, IEEE Transactions on Information Theory 22 (6) (1976), 644-654.

[13] D. Denning and G. M. Sacco, Timestamps in Key Distribution Protocols, Comm. ACM 24 (8) (1981) 533~536.

[14] W. Diffie, P. van Oorschot, and M. Wiener, Authentication and Authenticated Key Exchanges, Designs, Codes, and Cryptography 2 (2) (1992) 107~125.

[15] I. Ingemarasson, D.T. Tang, and C.K. Wong, A Conference Key Distribution System, IEEE Transactions on Information Theory 28 (5) (1982) 714-720.

[16] I. Jeong, J. Katz, and D. Lee, One-Round Protocols for Two-Party Authenticated Key Exchange, in : to appear in ACNS'04, 2004.

[17] Kaoru Kurosawa, Multi-recipient Public-Key Encryption with Shortened Ciphertext, in : Proceedings of PKC'02, 2002, pp.48~63.

[18] J. Katz and M. Yung, Scalable Protocols for Authenticated Group Key Exchange, in : Proceedings of CRYPTO'03, 2003, pp.110~125.

[19] L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, An Efficient Protocol for Authenticated Key Agreement, Technical report CORR 98-05, University of Waterloo, 1988.

[20] V. Shoup, On Formal Models for Secure Key Exchange, Available at http://eprint.iacr.org.

[21] S. Blake-Wilson, D. Johnson, and A. Menezes, Key Agreement Protocols and their Security Analysis, in : Proceedings of Sixth IMA International Conference on Cryptography and Coding, 1997, pp.30~45.

정 익 래

e-mail : jir@cist.korea.ac.kr

1998년 고려대학교 전산학과(학사)

2000년 고려대학교 전산학과(이학석사)

2004년 고려대학교 정보보호대학원(공학박사)

2004년 9월 2005년 9월 Maryland 대학교 포닥

2005년 9월 현재 고려대학교 정보보호기술연구센터 포닥연구원

관심분야 : 암호이론, 암호 프로토콜, 계산 복잡도 이론 등

이 동 훈

e-mail : donghlee@korea.ac.kr

1984년 고려대학교 경제학과 학사

1987년 Oklahoma Univ. 전산학과 석사

1992년 Oklahoma Univ. 전산학과 박사

1993년 2001년 고려대학교 전산학과 교수

2001년 현재 고려대학교 정보보호대학원 교수

1998년 현재 한국정보과학회 교육위원

1999년 현재 한국정보보호학회 논문지 편집위원

1999년 현재 컴퓨터 이론연구회 운영위원

관심분야 : 암호이론, 암호 프로토콜, 계산 복잡도 이론