通信情報保護學會論文誌
第 2 卷 第 1 號, 1992. 6

상호 신분 인증 및 디지탈 서명기법에 관한 연구

임채훈* · 이필중*

On Mutual Authentication and Digital Signature Schemes

Chae Hoon Lim and Pil Joong Lee

요 약

본 논문에서는 이산대수 문제에 바탕을 둔 Schnorr의 신분 인증 및 디지탈 서명방식을 이용하여 상호 신분 인증과 동시에 세션키를 분배할 수 있는 프로토콜 및 다자간의 회의용 키분배 프로토콜들을 제시한다. 또한 Schnorr의 서명을 이용하여 Chaum에 의해 도입된 undeniable signature의 한 예를 제시한다. 그리고 서명의 생성시에 수신자의 공개키를 결부시킴으로써 지정된 수신자만이 그 서명을 확인할 수 있고 필요하다면 수신자는 그 서명이 서명자에 의해 자신에게 발행된 서명임을 제 3 자에게 증명할 수 있는 새로운 서명방식을 제안한다. 이러한 수신자 지정 서명방식(directed signature)은 수신자의 프라이버시나 이해관계에 밀접한 관계가 있는 응용을 지향한 것으로 그 특성상 서명과 동시에 비밀보장이 요구되는 경우가 대부분인 만큼 암호화 기능과도 쉽게 결합시킬 수 있음을 보인다.

Abstract

This papar presents several protocols for mutual autherntication and key distribution and a protocal for conference key distribution using Schnorr's identification and signature scheme. We also present a selectively convertible undeniable signature scheme using modified Schnorr's signature scheme. A verifiable directed signature scheme is proposed, which is a signature scheme that only the designated neceiver can directly verify the signature and that he can prove, if necessary, its validity to any third party via an interactive protocol. This scheme is intended for those applications personally or commercially sensitive to a specific receiver.

* 포항공과대학 전자전기공학과

1. 서 론

최근의 컴퓨터 통신망에서는 고성능의 워크스테이션들이 대량 보급됨으로써 망전체의 터미날 수가 급격히 늘어나고 또한 LAN이나 다른 컴퓨터망들과의 상호접속 및 터미날의 원거리 접속(remote access) 등에 의해 훨씬 복잡한 연결성을 갖게 되었다. 이에 따라 컴퓨터망의 도처에 존재하는 위협들로부터 중요한 정보나 자원을 보호하기 위한 보다 강력한 사용자 인증기법이나 다양한 보안서비스들에 대한 요구가 증가하고 있다.

컴퓨터 통신망에서 가장 중요한 것은 원하는 상대방에 정확히 연결되었느냐를 확인하거나 연결된 상대방이 적법한 사용자인지를 확인할 수 있어야 한다는 것이며 이와같은 목적으로 Fiat-Shamir 방식[1]을 필두로 Guillou-Quisquater 방식[2], Schnorr 방식[3]등 많은 효율적인 신분인증 프로토콜(identification protocol)들이 개발되었다. 특히 이러한 신분인증 프로토콜은 중요한 건물의 출입통제나 금융기관의 신용카드, 비자(visa) 등 다양한 응용들에서 기존의 방식들을 스마트카드(smat card)로 통합할 수 있는 매우 효율적인 방법으로 주목받고 있다.

한편 컴퓨터망에서 교환되는 정보의 보호를 위한 가장 기본적인 도구로 관용 암호시스템(conventional cryptosystem)을 이용한 암호화 기법이 가장 널리 사용되고 있으나 이에 필요한 세션키(session key)를 효율적으로 분배하는 것이 중요한 과제의 하나로 많은 연구가 진행되어 왔다. 특히 최근에는 키분배 과정에서 발생될 수 있는 각종 위협들을 미연에 방지할 수 있도록 상호 신분 인증 과정을 통해 원하는 상대방과의 정확한 연결을 확인한 후 이 과정에서 서로간에 교환된 정보를 바탕으로 세션키를 분배하는 신분인증과 키 분배를 결합시킨 방식들이 활발히 연구되고 있다.[4-8]

본 논문의 전반부에서는 Schnorr의 신분인증 및 디지탈 서명을 이용하여 신분인증과 동시에 세션키를 분배할 수 있는 3-move 상호 신분 인증 방식(mutual authentication scheme)과 디지탈 서명을 이용한 회의용 키분배 방식(conference key distribution scheme)을 제시하기로 한다. 이러한 방법으로 상호 신분인증 및 키분배 프로토콜을 설계하는 것은 다른 신분인증 프로토콜을 이용하는 경우도 마찬가지로 적용될 수 있으며 Diffie-Hellman형의 키분배 방식[9]에 비해 상호 신분인증 과정을 결합시켰으므로 보다 안전하게 사용 될 수 있을 것이다. 또한 회의용 키분배 프로토콜의 경우도 안전성이 입증되었거나 그렇지 않다고 하더라도 충분한 기간 동안 분석되어져 안전한 것으로 생각되는 기존의 디지탈 서명들을 이용하는 만큼 안전성에 대한 우려를 줄일 수 있다는 잇점이 있다.

한편 메세지 및 사용자에 대한 인증기능과 동시에 메세지의 전송사실을 부인할 수 없게 하는 부인방지(unforgeability and undeniability) 기능을 갖는 일반적인 디지탈 서명(ordinary digital signature)은 각종 보안 서비스에서 필수 불가결한 도구로 사용된다. 대부분의 응용들에서는 이와같이 누구나 메세지의 출처와 메세지의 진위 여부(authentidty)를 확인할 수 있는 자체 인증기능(self-authentication)을 갖는 일반적인 디지탈 서명이 매우 유용하게 사용된다. 예를 들어 정부기관에서 고시하는 공문서(official annoucement)나 공공기관에서 발행하는 각종 증명서, 공개키의 진위 여부를 공증해 주는 공개키 증명서(public key certificate)등과 같은 일반적인 응용에서는 누구나 이를 확인할 수 있도록 하는 것은 필수적이므로 일반적인 디지탈 서명이 유용하게 사용될 수 있다.

그러나 개인적으로나 상업적으로 민감한 응용들에서는 이러한(누구나 서명의 정당성을 확인할 수 있는) 자체인증은 필요 이상의 과다한 인증기능을 제공함으로써 서명의 사본들이 악용될 수 있는 가능성을 높여 주게 된다(공갈 협박이나 밀매, 산업스파이 등). 따라서 단순한 서명의 사본만으로는 이를 확인할 수 없고 서명의 인증을 위해서는 반드시 서명자의 도움을 받아야 하거나 특정 수신자만이 서명을 확인할 수 있게 하는 방법 등에 의해 서명자나 수신자의 부당한 위협 가능성을 줄여 주고 프라이버시를 높여 줄 수 있는 서명방식이 보다 바람직한 경우가 있다. 예를 들어 소프트웨어 공급회사나 각종 제조업체들이 자사의 제품임을 보증하는 디지탈 서명을 발행하는 경우 서명자의 도움이 있어야만 그

서명을 확인할 수 있도록 함으로써 그 회사의 제품을 직접 구매한 고객만이 해당업체와의 대화(interactive protocol)를 통해 자신이 구입한 제품이 진본임을 확인할 수 있게 하고, 또한 비록 그 제품(진본)에 하자가 있을 경우라도 판매회사는 이를 부인할 수 없도록 할 수 있다. 이렇게 함으로써 서명자의 서명이 남용되는 것을 막을 수 있고 따라서 서명의 안전성에 대한 위협도 줄일 수 있을 것이다. 이와 같은 목적으로 도입된 것이 D. Chaum에 의해 제안된 undeniable signature[10),11)]이며 특히 비밀키의 일부를 노출시킴으로써 특정한 서명만 선택적으로, 혹은 전체 서명을 모두 일반적인 서명으로 변환시킬 수 있는 (selectively) convertible undeniable signature[12)]는 보다 발전된 형태로 많은 중요한 응용분야를 갖는다.

다음에는 undeniable signature와는 약간 상반되는 기능이 요구되는 응용의 예를 들어 보자. 모 정부기관에 근무하는 공직자가 신문사나 방송 등의 언론기관에 비밀정보를 제공하고자 하나 신원이 밝혀지는 것을 꺼려하여 익명을 요구하며 정보를 제공하려고 하는 경우를 생각해 보자. 언론기관에서는 허위정보를 보도할 수는 없으므로 정보제공자의 신원을 확인할 필요가 있을 것이고, 또한 정보제공자의 요구대로 그의 신원을 밝히지 않는다는 약속을 할 것이다. 이 경우에도 일반적인 디지탈 서명은 적합치 않다는 것은 분명하며, 만일 정보제공자가 undeniable signature를 사용한다고 가정해 보자. 이제 이 정보가 기사화되고 그 출처를 알아내기 위해 해당기관에서 이를 추적하는 과정에서 이 정보와 관련된 서명을 얻었다고 하자. 그러면 해당기관에서는 의심이 갈만한 모든 내부 직원들에게 comfimation protocol이나 disavowal protocol을 수행하게 함으로써 정보의 출처를 알아낼 수 있을 것이다. 즉 의심을 받는 사람은 disavowal protocol을 수행하면 쉽게 자신의 누명(?)을 벗을 수 있고, 오히려 이를 거부하는 것은 자신이 그 정보의 출처임을 시인하는 결과가 될 것이므로 이를 거부할 하등의 이유가 없을 것이다. 따라서 결국 그 정보의 제공자는 신원이 밝혀지게 될 것이므로 이와 같은 응용에서는 undeniable signature도 적합치 않다는 것을 알 수 있다. 이와 같은 가상의 시나리오를 바탕으로 T. Okamoto 등이 제안한 것이 non-transitive digital signature[13)]이다. 즉 대화형의 신분 인증 프로토콜(interactive identification protocol)을 변형하여 대화를 하는 상대방만이 서명자의 신원 및 메세지의 진위 여부를 확인할 수 있고 수신자는 제 3 자에게 이 서명이 해당 서명자의 서명임을 증명하는 것이 불가능하도록 한 것이다. 그러나 이는 디지탈 서명이라기 보다는 신분 인증 프로토콜과 메세지 인증 프로토콜(message authentication protocol)이 결합된 대화형 프로토콜로 그 서명이 문제가 되었을 때 분쟁해결(dispute resolution)의 기능이 없다는 점에서 디지탈 서명의 원래 기능을 한다고 보기는 어렵다.

한편 위의 가상 시나리오에서도 최악의 경우(그 정보제공자가 법을 위반하고 비밀정보를 누설하여 언론기관이나 해당 기자가 그 출처를 밝히지 않으면 대신 누명을 쓰고 사법처리를 받아야 할 경우)에는 확인자가 그 서명의 출처를 제판관에게 증명할 수 있도록 하는 것이 필요하다. 따라서 Okamoto 등의 non-transitive digital signature도 이런면에서 문제가 있음을 알 수 있다. 이와 같은 목적에 가장 적합한 서명방식으로 서명의 인증시에 특정 확인자의 비밀키가 요구되도록 함으로써 그 확인자만이 서명을 확인할 수 있도록 하되, 만일 그 서명이 문제가 되었을 때는 확인자의 비밀키를 노출시키지 않더라도 제 3 자(재판관)에게 서명의 출처를 증명함으로써 분쟁해결의 기능을 제공할 수 있는 서명방식이 요구된다. 이와같이 특정 수신자만이 서명을 확인할 수 있도록 하는 서명방식은 위의 가상 시나리오 이외에도 개인적인 이해관계나 프라이버시와 밀접한 관련이 있는 각종 응용에 매우 유용하게 사용될 수 있을 것이다. 예를 들어 세금고지서나 건강기록카드 등에 사용되는 서명은 해당 수신자만이 이를 확인할 수 있으면 충분하며 필요시에는 그 문서가 자신에게 발행된 것임을 증명할 수 있도록 하여 일반적인 서명을 사용했을 때 발생할 수 있는 악용들을 방지할 수 있다는 장점이 있다.

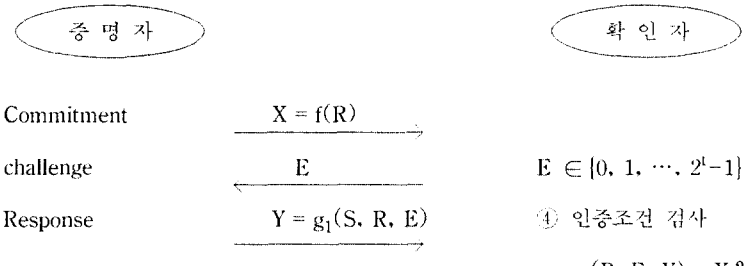
본 논문의 후반부에서는 Schnorr의 서명을 변형하여 위에서 언급한 두가지의 특수한 응용 지향적인 디지탈 서명방식들을 구현해 보고자 한다. 즉 서명자와의 대화를 통해서만 인증 가능한 Chaum의 un-

deniable signature의 한 예를 제시하고, 또한 지정된 수신자(확인자)만이 서명을 확인할 수 있고 해당 수신자는 필요시에 그 서명의 정당성을 제 3 자에게 증명할 수 있는 수신자 지정 서명방식(verifiable directed sgnature)을 제안하고자 한다. 후자의 경우 서명의 생성시에 원하는 상대방의 공개키를 결부시킴으로써 해당 비밀키를 소유한 수신자만이 이를 인증할 수 있도록 하고 수신자는 필요시에 제 3 자에게 이 서명을 확인시킬 수 있도록 함으로써 서명자가 서명한 사실을 부인하지 못하게 할 수 있다.

우선 2장에서 Schnorr의 신분인증 및 디지탈 서명방식들을 간략히 기술하고 3장에서는 이를 이용하여 세션키 분배가 가능한 상호 신분 인증 프로토콜을 구성하여 본다. 4장에서는 Schnorr의 서명을 이용한 회의용 키분배 프로토콜을 제시한다. 그리고 5, 6장에서는 Schnorr의 서명을 변형하여 구성한 undeniable signature와 수신자 지정 서명방식을 소개하기로 하며 7장에서 결론을 맺기로 한다.

2. Schnorr의 신분 인증 및 디지탈 서명방식

Schnorr의 신분 인증 프로토콜[3]은 Chaum-Everste-Graaf의 Protocol[14]을 1라운드의 병렬버전으로 압축시킨 것으로 안전성 증명은 어려우나 증명자의 비밀키에 대한 어떤 유용한 정보도 누출되지 않는 것으로 생각되며 또한 이산대수 문제를 푸는 것이 어려운 한 공격자의 성공 확률은 2^{-t}이하라는 것이 알려져 있다(즉 interactive proof system의 조건은 만족하나 zero-knowledge는 아님). 여기서 t는 안전 파라미터(security parameter)로 확인자(verifier)의 challenge값의 비트길이를 나타낸다. Fiat-Shamir 방식[1]이나 Guillou-Quisquater 방식[2] 등에서와 같이 대부분의 신분인증 프로토콜들은 증명자(prover)가 선택한 랜덤수(commitment)를 일방함수로 변환한 witness를 전송하는 1단계와 확인자가 선택한 랜덤수(challenge)를 전송하는 2단계, 마지막으로 증명자가 1단계의 commitment값과 2단계의 challenge값을 자신의 비밀키와 결합하여 계산된 값(response)으로 응답하는 3단계로 구성된다(그림 1 참조). 그리고 대부분의 경우 그 안전성은 2단계의 challenge값의 비트길이에 의존하게 된다.



f:일방 함수, g_1, $g_2$polynomial-time으로 계산 가능한 함수
R, E:증명자 및 확인자에 의해 발생된 랜덤수, P, S:증명자의 공개키 및 비밀키

그림 1. 일반적인 신분인증 프로토콜의 구성도

위와같은 신분 인증 프로토콜은 쉽게 디지탈 서명방식으로 바꿀 수 있다. 이는 서명자가 2단계의 challenge값 대신에 서명하고자 하는 메세지 m과 1단계의 witness값 X를 일방성 해쉬함수(one-way hash function) h로 압축하여 E=h(X, m)를 계산하고 이를 Y와 함께 전송함으로써 이룩된다. 다음에 Schnorr의 신분 인증 및 디지탈 서명방식을 간략히 기술한다.

우선 시스템 파라미터로 공개키 인증 센타(Key Authentication Center:KAC)는 512비트 이상의 큰 소수 p로서 p-1이 최소한 140비트 이상의 소수 q를 약수로 갖도록 선택하고 Z_p상에서 q를 위수(order)로 갖는 기본원소 α를 선택하여 p, q, α를 공개한다($p \geq 2^{512}$, $q \geq 2^{140}$, $q \mid p-1$, $\alpha^q \equiv 1 \bmod p$). 여기서 이산대수 계산시 기본원소 α의 위수 q를 알고 있을 때 매우 효율적인 알고리즘은 Pollard의 Monte

Carlo법[15]으로 그 시간복잡도는 $O(\sqrt{q})$정도로 주어지므로 대략 2^{70} 정도의 계산량(모듈라 곱셈수)이 요구되도록 하기 위해서는 q가 최소한 140비트 이상이 되도록 선택해야 할 것이다. 또한 일방성 해쉬함수로서 임의의 길이의 메세지를 압축하여 t(≥ 72) 비트길이의 랜덤수를 출력하는 h 역시 모든 사용자에게 공개한다.

각 사용자 i는 자신의 비밀키로 Z_q상에서 랜덤수 S_i를 선택하여 비밀로 간직하고 해당 공개키 $P_i \equiv \alpha^{S_i} \bmod p$를 계산, KAC에 제출하여 공개키 증명서(public key certificate)를 발급받는다. KAC는 시스템에 가입하고자 하는 각 사용자 i에 대해 그의 신원을 확인한 후 그의 ID를 공개키와 결합하여 임의의 안전한 디지탈 서명법으로 디지탈 서명한 형태의 공개키 증명서를 발급한다. 예를 들어 KAC에서 RSA를 이용하여 $C_i \equiv (P_i \oplus ID_i)^d \bmod n$ 형태의 공개키 증명서 C_i를 각 사용자 i에게 발급한다면 누구나 $C_i^e \bmod n \oplus ID_i = P_i$가 성립하는지를 검사함으로써 P_i가 그의 공개키임을 확인할 수 있다.[16] 여기서 n, e는 KAC의 RSA 서명용 공개키로 모두에게 공개하고 해당 비밀키 d($ed \equiv 1 \bmod \lambda(n)$)는 비밀리에 간직하며 모듈라 멱승(modular exponentiation)의 연산에서 reblocking이 생기지 않도록 $n > p$가 성립하도록 n을 정한다.

사용자 i가 j에게 신원을 증명하는 과정은 다음과 같다(그림 2).

① (commit) 사용자 i는 랜덤수 $R \in [1, q)$을 선택한 후 $X \equiv \alpha^R \bmod p$를 계산하여 자신의 공개키에 대한 센타의 서명과 함께 사용자 j에게 보낸다.

② (challenge) 사용자 j는 i의 공개키에 대한 서명을 확인한 후 랜덤수 $E \in [1, 2^t-1]$를 선택하여 i에게 전송한다.

③ (response) 사용자 i는 $Y \equiv R+S_iE \bmod q$를 계산하여 j에게 보낸다.

④ (verification) 사용자 j는 $\alpha^Y P_i^{-E} \equiv X \bmod p$가 성립하는지를 검사한다.

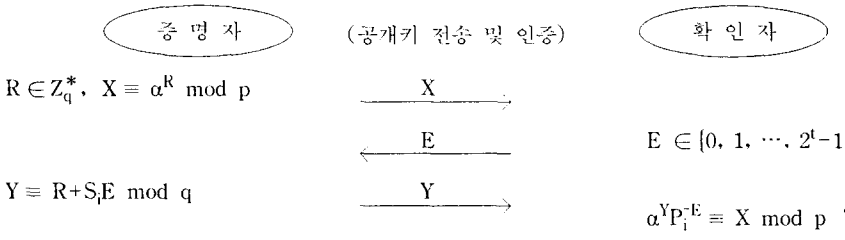


그림 2. Schnorr의 신분 인증방식

이와같은 신분인증 프로토콜을 스마트카드로 구현할 때는 계산비용에 비해 통신비용이 일반적으로 더 비싸게 먹히는 것으로 알려져 있으므로 전송량을 줄이는 것도 중요한 과제의 하나이다. 위의 프로토콜에서도 단계 ①에서 X 대신에 이를 해쉬함수로 압축한 결과인 h(X)=D를 전송하고 단계 ④에서 {E, Y}로부터 재생된 X를 같은 해쉬함수로 압축하여 이를 단계 ①에서 받은 D와 비교함으로써 상대방이 사용자 i임을 인증할 수 있을 것이다.

Schnorr의 디지탈 서명방식에서는 확인자의 질문 대신에 서명할 메세지 m과 1단계의 witness값 X에 대한 해쉬함수 h의 출력을 이용한다. 디지탈 서명의 생성 및 확인 과정은 다음과 같다.

① 서명자 i는 랜덤수 $R \in [1, q)$을 선택한 후 $X \equiv \alpha^R \bmod p$를 계산하고 $E=h(X, m) \in [1, 2^t-1]$, $Y \equiv R+S_iE \bmod q$를 계산하여 메세지 m과 디지탈 서명{E, Y}를 j에게 전송한다.

② 확인자 j는 $X \equiv \alpha^Y P_i^{-E} \bmod p$를 계산하여 $E=h(X, m)$이 성립하는지를 조사하여 메세지 m에 대한 서명을 인증한다.

만일 서명과 동시에 비밀보장이 요구된다면 다음과 같이 암호화 기능을 추가할 수 있다(그림 3 참조). 즉 서명자 i는 [1, q)에서 랜덤하게 선택한 R을 이용하여 $X \equiv \alpha^R \bmod p$를 계산하고 $E=h(X, m) \in [1, 2^t-1]$, $Y \equiv R+S_iE \bmod q$를 계산하여 메세지 m에 대한 디지탈서명 {E, Y}를 생성한다. 또한 Z

$\equiv P_j^R \equiv \alpha^{RS_j} \bmod p$를 계산하여 이를 일방성 해쉬함수로 압축한 $h(Z)=K$를 관용 암호시스템의 키로 사용하여 메세지 m을 암호화한 암호문 $C=E_K(m)$를 서명 {E, Y}와 함께 j에게 전송한다. 이를 받은 사용자 j는 $\alpha^Y P_i^{-E} \equiv X \bmod p$, $X^{S_j} \equiv Z \bmod p$, $h(Z)=K$를 계산하여 $E_K(C)=m$과 같이 복호화한 후 $h(X, m)=E$가 성립하는지의 여부를 조사하여 메시지 m에 대한 서명을 인증한다. 여기서 Z는 두 사용자 i와 j만이 계산할 수 있는 비밀 랜덤수로 디지탈 서명에 의해 특정한 메세지와 관련되어 있으며 또한 $\alpha^{ES_iS_j} \equiv Z^{-1} \cdot P_j^Y \bmod p$와 같이 Z가 알려지면 $\alpha^{S_iS_j} \bmod p$가 노출될 수 있겠지만 세션키는 $h(Z)=K$와 같이 해쉬함수의 결과를 이용하였으므로 실령 세션키가 알려진다 하더라도 다른 메세지의 서명이나 암호화에 영향을 미칠 수 없음을 알 수 있다.

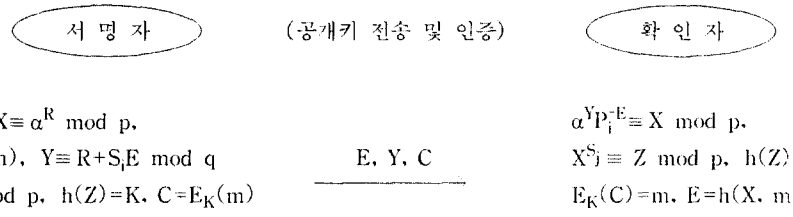


그림 3. 암호화 기능을 결합한 Schnorr의 디지탈 서명

3. 상호 신분 인증 프로토콜

이 장에서는 Schnorr의 신분인증 프로토콜 및 디지탈 서명을 이용하여 상호 신분 인증과 동시에 세션키를 분배할 수 있는 프로토콜을 제시하기로 한다. 이하 본 논문의 모든 프로토콜에서 각 사용자가 자신의 공개키 및 공개키 증명서를 상대방과 교환하여 그 진위 여부를 판단하는 공개키 인증단계는 생략하기로 한다.

신분인증 프로토콜의 안전변수(challenge 값의 비트길이)는 적용환경에 따라서 다르지만 대체로 20비트 이상이면 대부분의 경우 안전하게 사용될 수 있다. 즉 스마트카드와 단말기 사이의 인증과 같은 근접인증(loca verfication)의 경우 약 $10^{-3}-10^{-5}$ (10-17비트) 정도, 원거리 인증(remote verification)의 경우는 약 $10^{-6}-10^{-9}$(20-30비트) 정도의 속임 가능성이면 충분한 것으로 생각된다. 따라서 상호 신분인증 프로토콜에서 사용되는 해쉬함수 h는 20비트 이상의 출력을 내는 것으로 가정한다($t \geq 20$). 그러나 디지탈 서명에서 메세지를 압축하는데 사용되는 해쉬함수의 경우는 훨씬 큰 값이 요구된다. 만일 디지탈 서명이 RSA와 같이 해쉬함수의 출력이 permutation의 입력으로 사용되는 것이면 birthday attack이 적용될 수 있으므로 약 128비트 이상의 출력이 필요하며, zero-knowledge 기술로부터 생성된 디지탈 서명과 같이 해쉬함수의 입력과 출력이 서로 결합되어 있어 birthday attack이 적용되지 않는다면 약 64비트 이상의 출력이면 안전하게 사용될 수 있는 것으로 생각된다.

우선 다음의 프로토콜을 생각해 보자. 이는 Schnorr의 신분 인증 프로토콜을 두 사용자 쌍방에 적용시킨 것으로 2단계에서 각각의 challenge 값을 전송하는 대신에 1단계에서 교환한 witness를 결합하여 공통의 challenge 값으로 사용한 것이다. 이와같이 공통의 challenge 값을 1단계에서 교환한 witness 값에 의존하게 함으로써 challenge 값을 미리 예측하여 전송할 수를 결정하는 것은 불가능해지며 또한 전체 프로토콜을 2라운드로 줄임으로써 전송 효율을 높일 수 있다. 또한 후에 알겠지만 상호 신분 인증 프로토콜에서는 각 사용자가 challenge 값을 랜덤하게 선택하여 전송하는 것은 프로토콜에 대한 공격을 훨씬 용이하게 하므로 전단계에서 교환된 witness 값의 함수로 challenge값을 결정하는 것은 전체 프로토콜에 걸쳐 두 사용자 사이에 제3의 공격자가 끼어드는 것을 막는데 중요한 역할을 한다.

[프로토콜 1]

① 각 사용자 i, j는 각자가 선택한 비밀 랜덤수 R_i, $R_j \in [1, q)$를 이용하여 $X_i \equiv \alpha^{R_i} \bmod p$, $X_j \equiv \alpha^{R_j} \bmod p$를 계산, 서로 교환한다.

② 각 사용자는 ① 단계에서 교환한 X_i, X_j로 부터 $E=h(X_i) \oplus h(X_j)$를 계산하여 공통의 challenge 값으로 사용한다.

③ 각 사용자는 $Y_i \equiv R_i+S_iE \bmod q$, $Y_j \equiv R_j+S_jE \bmod q$를 계산하여 서로 교환한다.

④ 각 사용자는 $X_j \equiv \alpha^{Y_j} P_j^{-E} \bmod p$, $X_i \equiv \alpha^{Y_i} P_i^{-E} \bmod p$를 계산하여 이들이 ①단계에서 교환된 X_i, X_j와 같은지를 검사하다. 조건이 만족되지 않으면 프로토콜을 중단한다.

⑤ 이제 각 사용자는 서로의 신원이 정확함을 확인하였으므로 공통의 비밀수로 $X \equiv X_j^{R_i} \equiv X_i^{R_j} \equiv \alpha^{R_iR_j} \bmod p$를 계산한 다음 $K \equiv X \bmod q$를 그들간의 세션키로 사용한다. 여기서 q는 140 비트 이상의 수이므로 이 K값으로부터 적절한 방법으로 원하는 비트수의 세션키를 추출해 낼 수 있을 것이다.

마지막의 세션키 계산에서 $X \equiv \alpha^{R_iR_j} \equiv \alpha^{(Y_i-S_iE)(Y_j-S_jE)} \equiv \alpha^{Y_iY_j} \cdot P_j^{-Y_iE} \cdot P_i^{-Y_jE} \cdot \alpha^{S_iS_jE^2} \bmod p$와 같이 전개되며, 따라서 만일 공통의 비밀 랜덤수 X가 알려진다면 $\alpha^{S_iS_j} \bmod p$가 노출될 것이므로 known-key attack이 성공할 수 있을 것이다. 그러나 이 X를 q로 나눈 나머지를 세션키로 사용하였으므로 과거의 세션키가 노출된다고 하더라도 이로부터 X를 구하는 것은 불가능하다. 또한 세션키는 상호 신분 인증 프로토콜의 부산물로 계산된 것이므로 만일 신분 인증 과정이 안전하다면 impersonation을 하는 것도 불가능하다. 따라서 이와같이 계산된 세션키는 안전하게 사용될 수 있을 것이다.

한편 위의 프로토콜에서 신분 인증 과정의 안전성을 생각해 보자. 위의 신분 인증 프로토콜이 Schnorr의 방식과 다른 점은 앞에서도 언급했듯이 각 사용자들이 랜덤하게 선택하여 전송하던 challenge 값들 대신에 ①의 전송결과로부터 계산된 $E=h(X_i) \oplus h(X_j)$를 공통의 challenge값으로 대체한 것이다. 여기서 사용자 i를 가장한 공격자가 단계 ④의 테스트를 통과하기 위해서는 단계 ①에서 $X_i \equiv \alpha^{R_i} P_i^{-E} \bmod p$를 전송하고 단계 ③에서 $Y_i = R_i$를 전송해야 하나 단계 ①에서 전송한 X_i에 의해서 공통의 challenge 값 $E = h(X_i) \oplus h(X_j)$가 결정되므로 결국은 사용자 j로부터 X_j를 받은 후 $h(\alpha^{R_i} P_i^{-E} \bmod p) = E \oplus h(X_j)$를 만족하는 E를 구할 수 있어야 한다. 그러나 이는 해쉬함수와 이산대수 문제의 일방성이 동시에 결부된 문제이므로 이와 같은 E를 구하는 것은 불가능하다고 할 수 있다. 이러한 변형은 FS 방식이나 GQ방식 등을 상호 신분인증 프로토콜로 구현할 때에도 공히 적용될 수 있는 것으로 각각의 challenge 값을 전송하는 방식에 비해 라운드 수를 줄일 수 있으므로 보다 효율적이라 할 수 있다.

위에서 살펴보았듯이 이 프로토콜에서 증명자나 인증자 혹은 이들을 가장한 어떤 공격자도 그 자신만으로는 합법적인 사용자로 가장할 수 없음은 분명하다. 그러나 만일 공격자가 단계 ④의 테스트를 통과할 수 있는 값을 얻기 위해 합법적인 사용자를 원하는 값을 계산해주는 oracle로 사용할 수 있다면 그는 쉽게 합법적인 사용자로 가장할 수 있을 것이다(oracle session attack). 따라서 위의 프로토콜(신분인증 과정만을 생각함)은 다음과 같은 공격에 의해 쉽게 깨어질 수 있다. 여기서 공격자인 사용자 k는 사용자 i로 가장하여 사용자 j에게 신분을 인증하려고 하며 그에 필요한 전송들을 사용자 i로부터 얻어 이를 그대로 사용자 j에게로 중계한다.

[프로토콜 1에 대한 oracle session attack]

① 사용자 k는 어떤 방법으로든 사용자 i로 하여금 그에게 프로토콜을 시작하게 하고 (혹은 사용자 i가 사용자 k를 상대로 프로토콜을 시작할 때) 동시에 자신은 사용자 j에게 사용자 i를 가장하여 프로토콜을 시작한다. 그러면 사용자 i는 자신이 선택한 비밀 랜덤수 $R_i \in [1, q)$를 이용하여 $X_i \equiv \alpha^{R_i} \bmod p$를 계산하여 사용자 k에게 전송할 것이다. 이를 받은 사용자 k는 이를 그대로 사용자 j에게 전송한다. 이때 사용자 j는 상대방이 사용자 i인 것으로 생각하고 프로토콜을 진행할 것이다. 마찬가지로 사용자 j는 $R_j \in [1, q)$를 랜덤하게 선택하여 $X_j \equiv \alpha^{R_j} \bmod p$를 계산, 이를 사용자 k에게 전송할 것이며 사용자 k는 이를 그대로 사용자 i에게 전송한다.

② 사용자 i, j, k는 $E = h(X_i) \oplus h(X_j)$를 공통의 challenge 값으로 계산할 것이다.

③ 사용자 i는 $Y_i \equiv R_i + S_iE \bmod q$를 계산하여 사용자 k에게 전송할 것이고 사용자 k는 이를 그대로 j에게 전송한다. 사용자 j는 $Y_j \equiv R_j + S_jE \bmod q$를 계산하여 사용자 k에게 전송한다. 이제 사용자 k는 사용자 i로 사칭하는데 필요한 정보 Y_i를 얻었으므로 사용자 i와의 통신은 적당히 끊는다.

④ 이제 사용자 k는 $X_i \equiv \alpha^{Y_i} P_i^{-E} \bmod p$를 계산하여 이들이 ① 단계에서 받은 X_i와 같은지를 검사한다. 이는 사용자 j의 인증조건 검사와 동일할 것이므로 만일 이 검사가 통과된다면 이는 곧 사용자 k가 사용자 j에게 사용자 i로 성공적으로 가장하였음을 의미한다. 이상의 조건검사가 만족되지 않으면 각 사용자는 상대방과의 프로토콜을 중단한다.

위와같은 oracle session attack이 성공할 수 있었던 것은 크게 다음과 같은 두가지 요인에 근거한다. 첫째는 witness값의 전송시 그 값이 이를 전송하는 사용자와 무관하여 제 3 자가 이를 그대로 다른 사용자에게 중계하더라도 상대방이 이 값의 출처를 알 수 없다는 사실이다. 위의 공격에서 단계 ①의 전송시 사용자 k가 사용자 i로부터 받은 X_i를 사용자 j에게 그대로 전송해도 전 프로토콜에 걸쳐 이 값과 사용자 k와의 연관이 전혀 없으므로 아무런 문제가 발생되지 않은 것이다. 둘째는 사용자의 비밀키를 사용하여 계산된 인증 프로토콜의 마지막 전송을 상대방이 누구든 항상 받을 수 있다는 사실이다. 즉 공격자는 언제나 적법한 사용자를 oracle로 이용할 수 있는 것이다. 만일 위의 공격에서도 사용자 i가 사용자 k의 신원을 먼저 확인한 후 그가 진정한 사용자 k 일때만 응답 Y_i를 전송하게 했다면 위의 공격은 성공할 수 없을 것이다. 따라서 상호 신분 인증시에는 위와같은 대칭형의 프로토콜 보다도 비대칭형 프로토콜이 바람직함을 알 수 있다.

이제 프로토콜 1을 다음과 같이 변형하면 위에서 제시한 oracle session attack을 막을 수 있다. 즉 위에서 언급한 첫째 항목에 주목하여 witness의 전송시 상대방의 ID를 결합하여 이 값을 그대로 중계했을 때 마지막의 인증조건 검사에서 모순을 유도할 수 있다. 위의 프로토콜 1의 단계 ①에서 사용자 i는 $X_i \equiv \alpha^{R_i} \bmod p$를 그대로 전송하는 대신에 $T_i \equiv \alpha^{R_i} \bmod p$로 두고 이를 상대방의 ID인 $ID_i(<p)$와 결합하여 $X_i \equiv (T_i \oplus ID_j) \bmod q$를 전송한다면 위의 프로토콜은 oracle session attack에 대해 안전함을 쉽게 알 수 있다(여기서 X_i로부터 T_i를 유도해 내는 것은 불가능함을 주목하자).

[프로토콜 2]

① 사용자 i, j는 각자가 선택한 비밀 랜덤수 R_i, $R_j \in [1, q)$로 $T_i \equiv \alpha^{R_i} \bmod p$, $T_j \equiv \alpha^{R_j} \bmod p$를 계산한 다음 통신하고자 하는 상대방의 ID를 이용하여 $X_i \equiv (T_i \oplus ID_j) \bmod q$, $X_j \equiv (T_j \oplus ID_i) \bmod q$를 계산, 상대방에게 전송한다.

② 각 사용자는 ① 단계에서 교환한 X_i, X_j로부터 $E = h(X_i) \oplus h(X_j)$를 계산하여 공통의 challenge값으로 사용한다.

③ 각 사용자는 $Y_i \equiv R_i + S_iE \bmod q$, $Y_j \equiv R_j + S_jE \bmod q$를 계산하여 서로 교환한다.

④ 각 사용자는 $T_j \equiv \alpha^{R_j} P_i^{-E} \bmod p$를 계산한 다음 이 값과 자신들의 ID로 부터 $\underline{X_j} \equiv (T_j \oplus ID_i) \bmod q$, $\underline{X_i} \equiv (T_i \oplus ID_j) \bmod q$를 계산한다. 만일 이들이 ①단계에서 교환된 X_j, X_i와 같다면 상대방은 적법한 사용자임을 확신하고 조건이 만족되지 않으면 프로토콜을 중단한다.

⑤ 이제 각 사용자는 서로의 신원이 정확함을 확인하였으므로 공통의 비밀수로 $T \equiv T_j^{R_i} \equiv T_i^{R_j} \equiv \alpha^{R_iR_j} \bmod p$를 계산한 다음 $K \equiv T \bmod q$를 그들간의 세션키로 사용한다.

위의 프로토콜 2에 oracle session attack을 적용한다면 사용자 j가 단계 ①에서 사용자 i를 가장한 사용자 k로부터 받는 것은 $X_i \equiv (T_i \oplus ID_k) \bmod q$가 될 것이고 단계 ④에서는 자신의 ID인 ID_j를 결합하여 $\underline{X_i} \equiv (T_i \oplus ID_j) \bmod q$를 계산할 것이므로 두 값이 같을 수 없다. 또한 사용자 k가 단계 ①에서 사용자 j로부터 받은 X_j를 그대로 사용자 i에게로 중계하지 않는 한 단계 ②에서 사용자 i가 계산하는 challenge값이 사용자 j가 계산하는 challenge값과 같을 수는 없으므로 단계 ④에서 사용자 j에게로 전송할 적법한 응답 Y_i를 얻을 수 없을 것이다. 만일 원래의 Schnorr 방식에서와 같이 challenge 값 E를 각 사

용자들이 랜덤하게 선택하여 전송하도록 했다면 역시 위의 프로토콜은 oracle session attack에 의해 깨어질 것이다. 따라서 challenge 값을 두 사용자의 witness 값의 함수로 계산하도록 한 것은 위 프로토콜의 안전성 유지에 중요한 역할을 함을 알 수 있다. 한편 적법한 사용자 i를 oracle로 이용하지 않는 한 사용자 i의 비밀키를 모르는 공격자가 단계 ④의 전송에 필요한 응답을 계산할 수 없다는 사실은 Schnorr 인증방식의 안전성에 근거하므로 위의 프로토콜 2는 일단 위에서 제시한 oracle session attack하에서는 안전한 것으로 생각할 수 있다.

그러나 위의 프로토콜 2가 앞에서 언급한 것처럼 대칭형 프로토콜로서 어느 사용자가 먼저 통신을 시작한다는 조건이 없다는 사실은 여전히 문제가 될 수 있다. 즉 프로토콜 1에 대한 oracle session attack에서 공격자인 사용자 k는 사용자 i와의 대화에서도 그를 사용자 j로 가장할 수 있다. 프로토콜이 완전 대칭이므로 공격자는 사용자 j의 공개키 및 공개키 증명서를 사용자 i에게 전송하고 사용자 i가 먼저 통신을 시작하여 witness값을 전송하기를 기다리며 마찬가지로 사용자 j에게도 사용자 i의 공개키 및 공개키 증명서를 전송하여 사용자 i로 가장, 통신을 시작한다. 이제 사용자 i로부터 witness 값을 받은 후부터는 두 사용자 사이의 모든 전송을 그대로 중계하면 결국 공격자는 사용자 j에게 성공적으로 사용자 i를 사칭할 수 있을 것이다. 따라서 위의 프로토콜 2도 통신 주도권의 애매성으로 인해 이러한 공격하에서는 안전하지 않을 수 있다.

이제 위의 프로토콜 2를 한 사용자에게 initiative를 주어 그가 원하는 상대방에게만 프로토콜을 시작하게 하는 비대칭형 프로토콜로 바꾸어 위에서 언급한 공격들에 대해 안전한 프로토콜로 변형시켜 보자. 이는 프로토콜 2를 다음과 같이 비대칭형의 3-move protocol로 변형시킴으로써 쉽게 이룩될 수 있다 (그림 4 참조).

[프로토콜 3]

① 사용자 i는 랜덤하게 선택한 비밀 랜덤수 R_i, $\in[1, q)$로 $T_i \equiv \alpha^{R_i} \bmod p$를 계산하고 통신 하고자 하는 상대방인 사용자 j의 ID를 이용, $X_i \equiv T_i \oplus ID_j$ mod q를 계산한 후 이를 사용자 j에게 전송한다.

② 사용자 j 역시 마찬가지 방법으로 $T_j \equiv \alpha^{R_j} \bmod p$, $X_j \equiv (T_j \oplus ID_i) \bmod q$를 계산한다. 또한 $E = h(X_i) \oplus h(X_j)$와 랜덤수 R_j, 자신의 비밀키 S_j를 이용하여 $Y_j \equiv R_j + S_jE \bmod q$를 계산하여 $\{X_j, Y_j\}$를 사용자 i에게 전송한다.

③ 사용자 i는 단계 ②에서 사용자 j로부터 받은 $\{X_j, Y_j\}$를 이용하여 $E = h(X_i) \oplus h(X_j)$를 구하고 $\underline{T_j} \equiv \alpha^{Y_j} P_j^{-E} \bmod p$, $\underline{X_j} \equiv (T_j \oplus ID_i) \bmod q$를 계산한다. 이 값이 단계 ②에서 받은 X_j와 같은지를 조사하여 만일 이 조건이 만족되면 사용자 i는 상대방이 사용자 j임을 확신하고 응답 $Y_j \equiv R_j + S_jE \bmod q$를 계산하여 전송한다. 그리고 사용자 j와의 세션키로 $K \equiv T \bmod q$, $T \equiv T_j^{R_i} \bmod p$를 계산한다. 조건이 만족되지 않으면 프로토콜을 중단한다.

④ 사용자 j는 $\underline{T_i} \equiv \alpha^{Y_i} P_i^{-E} \bmod p$, $\underline{X_i} \equiv (T_i \oplus ID_j) \bmod q$를 계산하여 ① 단계에서 받은 X_i와 같은지를 조사하여 사용자 i를 인증한다. 인증조건이 만족되면 $T \equiv T_i^{R_j} \equiv \alpha^{R_iR_j} \bmod p$를 계산한 다음 $K \equiv T \bmod q$를 사용자 i와의 세션키로 사용한다.

이제 위의 프로토콜 3은 만일 사용자 i가 그가 원하는 상대방에게만 통신을 시작한다면 앞에서 언급한 어떤 공격하에서도 안전함을 알 수 있다. 각 사용자는 자신이 원하는 상대방에게만 통신을 시작해야 한다는 조건은 비대칭형 프로토콜에서는 당연한 가정이며 이러한 프로토콜을 이용하는 사용자가 곡 지켜야할 기본 수칙이라 할 수 있다. 만일 공격자가 임의의 사용자로 가장하여 사용자 i로 하여금 필요할 때는 언제든지 그와의 통신을 시작하게 할 수 있다면 어떤 상호 신분 인증 프로토콜도 공격자의 중계에 의한 사칭(사용자 i로부터의 전송을 자신이 사용자 i로 가장하고자 하는 임의의 다른 사용자에게 그대로 전송하고 또한 그로부터의 모든 전송을 사용자 i에게로 중계하는 공격) 아래에서는 안전하지 않을 것이기 때문이다. 이러한 가정하에서 위의 프로토콜 3은 프로토콜 2에서 존재하던 공격자의 동시 사칭 문제가 자연히 해결된다.

이상에서 살펴본 것처럼 zero-knowledge technique으로부터 생성된 각종 신분인증 프로토콜들은

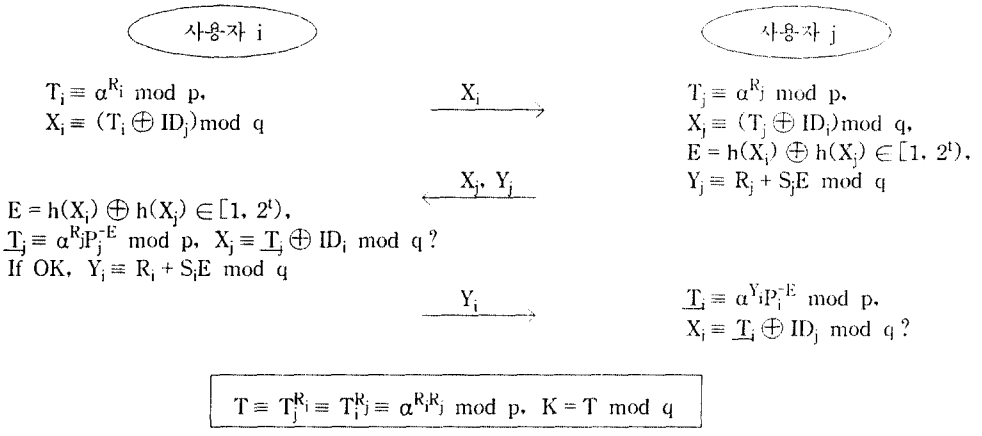


그림 4. Schnorr 인증방식을 이용한 상호 신분인증 및 키 분배

쉽게 상호 신분 인증 프로토콜로 바꿀 수 있으며 동시에 후속되는 통신에서 사용될 세션키를 분배할 수도 있다. 그러나 이들 신분 인증 프로토콜을 두 사용자 쌍방에 그대로 적용함으로써 얻어지는 상호 신분 인증 프로토콜은 결코 안전할 수 없으며 위에서 제시한 것처럼 적법한 사용자를 oracle로 이용하여 원하는 응답을 얻어내는 oracle session attack을 막을 수 있도록 주의를 요함을 지적하고자 한다.

Desmedt 등은 위에서 고려한 oracle session attack과 유사하게 인증자(사용자 j)가 제 3 자(사용자 A)와 결탁하여 그를 다른 사용자(사용자 B)에게 증명자 (사용자 i)로 사칭할 수 있도록 모든 전송들을 중간에서 중계해 주는 소위 mafia fraud 혹은 middle person attack 하에서는 상호 신분 인증 프로토콜을 포함하여 기존의 어떤 인증 프로토콜도 안전할 수 없다고 주장하였다.[17),18)] 그리고 이를 막는 방법으로 스마트카드와 단말기를 인증기간 동안 외부와 완전히 격리시키는 Faraday cage(identification cage)를 이용하는 방법[18]이나 게임이론에서 Chess Grandmaster problem을 해결하는 방법(게임을 하는 두사람 사이에 시간간격 t를 미리 정해 두고 자신의 차례가 되었을때 정확히 t초가 경과된 후에 이동)을 인증 프로토콜에 적용한 방법[19]등이 제안되었다. 그러나 이들은 모두 스마트카드와 단말기 사이의 인증과 같은 근접인증의 경우에만 적용가능한 방법이며 보다 일반적인 통신환경하의 원거리 인증에서는 적용될 수 없다. 일반적인 원거리 인증의 경우에는 위에서 살펴보았듯이 1단계의 witness 값을 자신이 통신하고자 하는 상대방의 ID와 결합시키고 2단계의 challenge 값을 1단계에서 교환된 witness 값들의 함수로 계산하며 또한 프로토콜을 비대칭형으로 구성하여 각 사용자가 자신이 원하는 상대방에게만 통신을 시작하게 한다면 이러한 relay attack을 쉽게 막을 수 있다.

한편 인증자와 제 3 자의 결탁에 의한 mafia fraud는 위에서 설명한 것처럼 막을 수 있으나 만일 증명자인 사용자 i가 의도적으로 제 3 자인 사용자 k와 결탁하여 그를 다른 사용자 j에게 사용자 i인 것처럼 사칭할 수 있도록 도와 주는 것은 항상 가능하다 (terrorist fraud). 이는 비밀키의 소유자인 사용자 i가 기꺼이 자신을 가장하도록 사용자 k를 도와 주겠다는 것이므로 어떤 신분 인증 프로토콜에서도 마찬가지로 적용될 수 있으며 이를 막기 위해서는 앞서 언급한 참고문헌[18),19)]에서 제안한 것과 같은 특수한 격리시설이나 시간제한 등의 방법을 이용할 수 밖에 없다. 그러나 이는 근접인증의 경우에만 적용가능하며 원거리 인증의 경우에는 이는 곧 사용자 i가 사용자 j와의 인증과정을 마친후 사용자 k로 하여금 자신의 이름으로 통신을 하도록 자리를 비켜주는 것과 다를 바가 없으므로 이를 공격의 한 유형으로 분류하는 것은 의미가 없을 것이다.

이러한 공격은 신분 인증 프로토콜이 실제 구현

되었을 때 응용에 따라서는 중대한 위협이 될 수 있으므로 여기에 대한 대책이 반드시 마련되어야 할 것이다. 이 공격을 악용할 수 있는 가장 치명적인 예가 신분 인증 프로토콜이 비자 프로토콜로 사용되고 여기에 이 공격이 적용되었을 때이며 이로부터 terrorist fraud라는 악명이 유래한다. 즉 특정 국가의 입국에 아무런 하자가 없는 제 3 자(terrorist의 일행)가 무선링크를 이용하여 입국 금지된 terrorist가 그 나라로 입국할 수 있도록 도와 줄 수 있다는 것이다.

한편 Schnorr의 서명과 timetamp를 이용하면 다음과 같이 1라운드로 압축된 신분인증 및 키분배 프로토콜을 구성할 수 있다. 이 경우 프로토콜은 완전 대칭이므로 한 사용자 i에 대해서만 기술하기로 한다(그림 5 참조).

[프로토콜 4]

① 사용자 i는 랜덤하게 선택한 비밀수 R_i, $\in [1, q)$를 이용하여 $X_i \equiv \alpha^{R_i} \bmod p$를 계산한 다음 이를 당시의 시간, 날짜 등으로 구성된 timetamp T_i, 그리고 통신하고자 하는 상대방의 ID인 ID_j와 함께 공개된 해쉬함수 h로 압축한 결과인 $E_i = h(X_i, ID_j, T_i)$를 구한다. 이 E_i와 비밀 랜덤수 R_i를 이용하여 Schnorr의 서명 $Y_i \equiv R_i + E_iS_i \bmod q$를 계산하여 (T_i, E_i, Y_i)를 사용자 j에게 전송한다.

② 사용자 i는 ① 단계에서 사용자 j로부터 받은 (T_j, E_j, Y_j)로부터 timestamp T_j가 현재의 시간과 비교하여 정해진 한계(예를들면 1분)를 넘지 않는다면 $X_j \equiv \alpha^{Y_j} P_j^{-E_j} \bmod p$를 계산하여 $h(X_j, ID_i, T_j) = E_j$가 성립하는지를 검사하여 상대방이 사용자 j임을 확인한다. 만일 T_j가 정해진 한계를 넘는다면 ① 단계의 전송이 제 3 자에 의한 과거 정보의 재전송이라고 간주하고 프로토콜을 중단한다.

③ 이제 각 사용자는 서로의 신원이 정확함을 확인하였으므로 공통의 비밀수로 $X \equiv X_j^{R_i} \equiv X_i^{R_j} \equiv \alpha^{R_iR_j} \bmod p$를 계산하여 $K \equiv X \bmod q$를 그들간의 세션키로 사용한다.

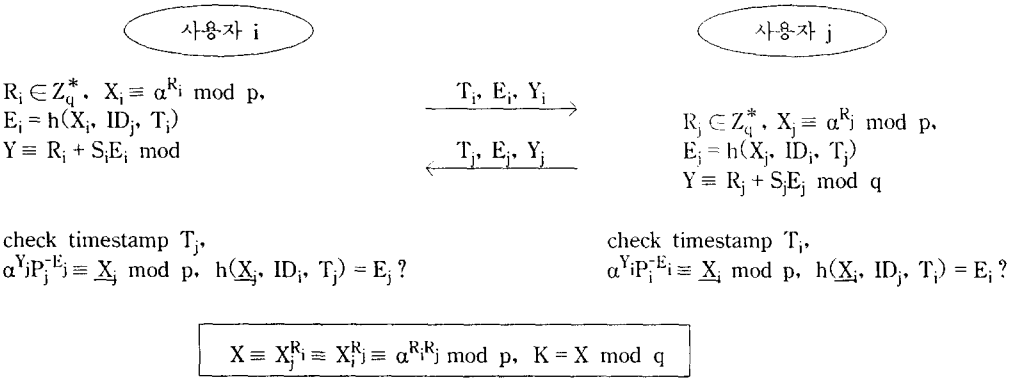


그림 5. Schnorr의 서명을 이용한 상호 신분 인증 및 키분배

위의 프로토콜은 쉽게 알 수 있듯이 이에 사용된 서명이 안전하다면 상호 신분 인증과 동시에 세션키 분배기능을 안전하게 제공할 수 있다. 여기서 timestamp를 사용함으로써 과거 전송정보의 재전송(replay attack)을 막을 수 있고 또한 서명의 생성시 상대방의 ID를 결합시킴으로써 공격자가 이를 제 3 자에게 중계하여 적법한 사용자로 가장하는 것을 막을 수 있다.

4. 회의용 키분배 프로토콜

3장에서는 두 사용자간에 상호 신분 인증과 동시에 세션키를 분배할 수 있는 프로토콜들을 제시하였으나 이들은 세명 이상의 다자간의 비밀회의를 위한 용

도로는 사용될 수 없으므로 여기서는 이와같은 회의용 키분배 프로토콜(conference key distribution scheme)을 다루기로 한다. 지금까지 몇몇 프로토콜들이 제안되었다가 해독되고 다시 수정되는 등[20-25] 우여곡절을 격어온 과정에서 알 수 있듯이 회의용 키분배 프로토콜은 두 사용자간의 키분배를 위한 프로토콜 보다 설계가 용이하지 않으며 그 안전성 분석 또한 어렵다. 이러한 측면에서 안전성이 증명되었거나 그렇지 않다고 하더라도 충분한 기간동안 분석되어져 안전한 것으로 생각되는 기존의 디지탈 서명방식들을 이용하여 프로토콜을 구성하는 것은 일단 그 설계가 용이할 뿐더러 안전성 분석 또한 거의 자명해지므로 회의용 키분배 방식을 설계하는 다른 한 방법이 될 수 있다.

여기서는 Schnorr의 서명을 이용하여 간단히 회의용 세션키를 분배할 수 있는 프로토콜을 제시한다. 즉 n명의 사용자들이 성형구조(star network)로 연결되어 있을 때 1대 (n-1)의 키분배 프로토콜을 구성해 보기로 한다. 회의용 키분배 방식으로는 프로토콜의 진행상 통신형태에 따라 크게 고리형(ring network), 성형 그리고 그물형(mesh or complete graph network) 등으로 분류할 수 있으나 대부분의 경우 성형구조의 프로토콜이 간단하면서도 효율적이다.[20] 아래에 제시하는 프로토콜은 Schnorr의 디지탈 서명을 이용한 일방향의 1대 (n-1) 프로토콜로 편의상 n명의 사용자를 사용자 1, …, 사용자 n으로 지칭하고 사용자 1이 통신을 시작하는 것으로 가정한다.

[사용자 1]

① Z_q^* 상에서 n개의 랜덤수 $\{R_i, \rho\}$ $(2 \le i \le n)$를 발생시킨다.

② 각 i에 대하여 $X_i \equiv \alpha^{R_i} \bmod p$, $Z_i \equiv P_i^{\rho} \equiv \alpha^{ps_i} \bmod p$를 계산한다.

③ 공개된 해쉬함수를 이용하여 $D_i = h(ID_i, X_i, Z_i, T_i)$를 계산한다. 여기서 ID_i는 사용자 i의 ID이며 T_i는 계산 당시의 시간/날짜 등을 나타내는 timestamp이다.

④ 자신의 비밀키 S_1을 이용하여 $Y_i \equiv R_i + S_1 D_i \bmod q$를 계산한 다음 $\{D_i, Y_i, Z_i, T_i\}$를 각 사용자 i에게 전송한다.

[사용자 i] $(2 \le i \le n)$

① 각 사용자 i는 timestamp T_i와 당시의 시간과의 차이가 정해진 한계를 넘지 않는지의 여부를 조사하여 자신이 받은 정보가 불법적인 제3자에 의한 과거 전송정보의 재전송이 아님을 확인한다. 조건이 만족되지 않으면 프로토콜을 중단한다.

② 각 사용자 i는 사용자 1의 공개키 P_1을 이용하여 $\alpha^{Y_i} P_1^{-D_i} \equiv X_i \bmod p$를 계산한 후 $h(ID_i, X_i, Z_i, T_i) = D_i$를 만족하는지의 여부를 조사한다. 이 조건이 만족되지 않으면 프로토콜을 중단한다.

③ 이상의 모든 과정을 통과하면 각 사용자 i는 자신의 비밀키 S_i를 이용하여 $K \equiv Z_i^{S_i^{-1}} \equiv \alpha^{\rho} \bmod p$와 같이 회의용 비밀키 K를 계산한다. 여기서 S_i^{-1}는 법 q로 계산한 S_i의 곱셈에 대한 역원을 나타낸다.

위의 프로토콜은 쉽게 알 수 있듯이 사용자 1의 메세지 Z_i에 대한 Schnorr의 서명을 각 사용자 i에게 전송한 것으로 여기에 과거 전송정보의 재전송을 막기 위한 timestamp가 첨가된 것이다. 이 timestamp의 사용은 일방향의 전송만으로 이루어지는 모든 프로토콜에서 재전송 공격을 막기 위해서는 필수불가결한 요소이다. 여기서 메세지 $Z_i \equiv P_i^{\rho} \equiv \alpha^{ps_i} \bmod p$는 회의용 비밀키 $K \equiv \alpha^{\rho} \bmod p$가 사용자 i의 공개키 P_i와 Difie-Hellman 형으로 결합된 형태이므로 (Diffie-Hellman 문제가 어려운 한) 오직 공개키 P_i에 대한 비밀키 S_i를 알고 있는 사용자 i만이 이로부터 회의용 비밀키 K를 계산할 수 있음을 알 수 있다. 따라서 위의 프로토콜의 안전성은 Diffie-Hellman 문제와 Schnorr의 서명의 안전성에 전적으로 의존한다고 할 수 있다.

5. Undeniable signature

이 장에서는 D. Chaum[10]에 의해 최초로 제안된 undeniable signature의 일종으로 Schnorr의 서명을 변형하여 하나의 공개키만으로 (selectively) convertible한 undeniable signature를 구성해 보기로 한다. 서명자의 비밀키, 공개키 쌍은 (S, P), P

$\equiv \alpha^S \bmod p$로 주어지고 이외에도 서명자는 관용 암호시스템을 위한 비밀키로 K를 갖는다. 이는 MAC(Message Authentication Code)값을 발생시키는 방법과 같은 원리로 관용 암호시스템을 이용하여 임의의 길이의 메세지를 [1, q) 사이의 랜덤한 값으로 압축시키는 알고리즘 f의 키로 사용될 것이다.

메세지 m에 대한 undeniable signature의 생성 과정은 다음과 같다.

① 랜덤수 $R \in [1, q)$를 선택하여 $Z \equiv \alpha^R \bmod p$를 계산한다.

② 이 Z와 메세지 m을 K를 비밀키로 하는 알고리즘 f로 압축하여 $f(K, Z, m)=\rho \in [1, q)$를 구한다.

③ 다음으로 $W \equiv \alpha^\rho \bmod p$를 계산하고 공개된 해쉬함수로 $E=h(W, m)$을 구한다.

④ $Y \equiv RE+S\rho \bmod q$를 계산하면 (Z, W, Y)가 메세지 m에 대한 서명이 된다.

즉 (Z, W, Y)는 $\alpha^Y Z^{-E} \equiv W^S \equiv P^\rho \bmod p$를 만족할때만 메세지 m에 대한 유효한 서명이 된다. 여기서 만일 ρ를 공개한다면 이 ρ에 대응하는 하나의 메세지에 대한 서명은 보통의 디지탈 서명으로 바뀌어짐을 쉽게 알 수 있다(selectively convertible). 한편 비밀 키 K 자체를 공개한다면 임의의 메세지에 대한 서명에 대해서도 $\rho=f(K, Z, m)$과 같이 누구나 ρ를 계산할 수 있으므로 이때까지 발행된 모든 undeniable signature를 보통의 디지탈 서명으로 변환시킬 수 있다(convertible). 따라서 두개의 공개키를 이용하여 ElGamal의 서명으로 구성한 Chaum등의 selectively convertible undeniable signature[12]에 비해 단 하나의 공개키만으로 같은 기능의 서명을 구현할 수 있으므로 보다 효율적이라 할 수 있다.

이 서명에 대한 인증 프로토콜는 Chaum의 confirmation/disavowal protocol[11), 12)]을 그대로 이용할 수 있다. 즉 서명자가 임의의 확인자로부터 (Z, W, Y)가 주어진 메세지 m에 대한 유효한 서명인지를 인증해 줄 것을 요청받았다면 서명자는 $\log_W V=\log_\alpha P$가 성립하는지의 여부를 확인자와의 대화형 프로토콜인 confirmation/disavowal protocol을 통하여 실현할 수 있다. 여기서 $\log_X Y$는 X를 밑으로 하는 Y의 이산대수, 즉 $Y \equiv X^Z \bmod p$를 만족하는 Z를 나타내며 V는 주어진(Z, W, Y)와 m으로부터 계산된 $V \equiv \alpha^Y Z^{-E} \bmod p$를 나타낸다. 다음에 Chaum의 zero-knowledge protocol을 간략히 기술한다(그림 6 참조).

[Confirmation protocol]

① 확인자는 두 랜덤수 $a, b \in [1, q)$를 선택하여 $T \equiv W^a \cdot \alpha^b \bmod p$를 증명자에게 전송한다.

② 증명자는 랜덤수 $t \in [1, q)$를 선택하여 $D_1 \equiv T.\alpha^t \bmod p$, $D_2 \equiv D_1^S \bmod p$를 계산, 확인자에게 전송한다.

③ 확인자는 자신의 랜덤수 a, b를 증명자에게 전송한다.

④ 증명자는 확인자로부터 받은 a, b를 이용하여 $T \equiv W^a \cdot \alpha^b \bmod p$가 성립하는지를 검사하여 단계 ①에서 확인자가 적법한 challenge값을 전송했는지를 확인한다. 만일 이 합동식이 성립한다면 자신의 랜

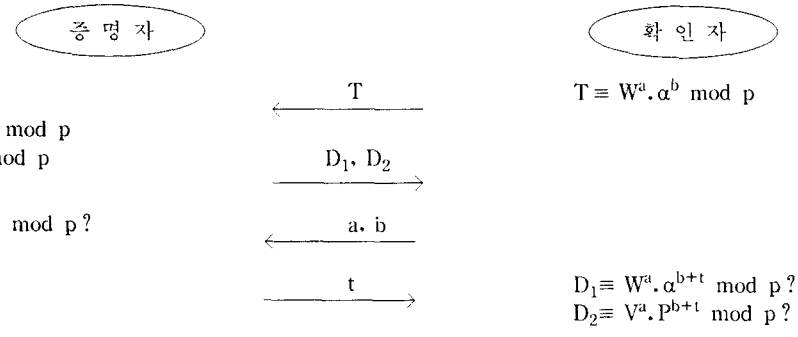


그림 6. Chaum의 confirmation protocol

덤수 t를 확인자에게 전송하고 그렇지 않다면 프로토콜을 종료한다.

⑤ 확인자는 증명자로부터 받은 t를 이용하여 $D_1 \equiv W^a \cdot \alpha^{b+t} \bmod p$, $D_2 \equiv V^a \cdot P^{b+t} \bmod p$가 성립하는지를 조사한다.

만일 단계 ⑤의 테스트를 통과한다면 (Z, W, Y)는 m에 대한 증명자의 서명임이 입증된 것이며 그렇지 않다면 다음의 두가지 가능성을 생각할 수 있다. 즉 (Z, W, Y)가 m에 대한 유효한 서명이 아니거나 유효한 서명인데도 서명자가 이를 부인하려고 하는 경우이다. 이 두 가능성은 후술하는 disavowal protocol에 의해 구분 가능하다. 위에서 기술한 confimation protocol은 쉽게 알 수 있듯이 증명자와의 대화없이도 통신내용들을 simulation하는 것이 가능하며 또한 비밀키를 모르는 제 3 자가 ⑤의 테스트를 통과할 확률은 기껏해야 1/q로 랜덤하게 추측하는 방법뿐이라는 사실이 참고문헌 12.에 상세히 증명되어 있다(zero-knowledge interactive proof system).

한편 zero-knowledge는 아니나 보다 효율적인 confirmation protocol로 Chaum-Eveste-Graaf의 Protocol 3(simultaneous discrete log.)[14]의 병렬 버전을 사용할 수 있을 것이다. 즉 $P \equiv \alpha^S \bmod p$, $V \equiv W^S \bmod p$를 동시에 만족하는 이산대수 S를 알고 있다는 사실을 보이기 위해 증명자는 [1, q) 구간에서 랜덤하게 선택한 R을 이용하여 $X_1 \equiv \alpha^R \bmod p$, $X_2 \equiv W^R \bmod p$를 계산해 X_1, X_2를 확인자에게 전송하고 확인자는 [1, 2^l) 구간에서 선택한 랜덤수 E로 질문하며 이에 대한 응답으로 증명자는 $Y \equiv R+SE \bmod q$를 전송한다. 이를 받은 확인자는 $\alpha^Y P^{-E} \equiv X_1 \bmod p$, $W^Y V^{-E} \equiv X_2 \bmod p$의 두 합동식이 성립하는지를 검사하여 증명자가 $P \equiv \alpha^S \bmod p$, $V \equiv W^S \bmod p$를 동시에 만족하는 이산대수 S를 알고 있는지의 여부를 확인할 수 있다.

위에서 언급했듯이 confimation protocol이 실패했을 때의 두가지 가능성을 판별해 줄 수 있는 disavowal protocol[2]은 다음과 같다(그림 7 참조). 우선 여기에 필요한 파라미터로 KAC에서는 위수가 q인 랜덤수 β를 공통의 밑으로 공개해야 하며(α를 밑으로 하는 β의 이산대수는 모두에게 비밀이어야 한다) 안전 파라미터인 k 역시 공통의 상수로 공개하거나 두 통신 당사자들 사이에 미리 협의되어야 한다. 여기서는 증명자가 속일 가능성이 1/(k+1) 이므로 이 가능성을 원하는 레벨 이하로 낮추기 위해서는 아래의 프로토콜을 필요한 수만큼 반복 시행해야 할 것이다.

[Disavowal protocol]

① 확인자는 [0, k] 구간에서 랜덤하게 선택한 r과 [0, q) 구간에서 랜덤하게 선택한 a를 이용하여 $T_1 \equiv W^r \cdot \alpha^a \bmod p$, $T_2 \equiv V^r \cdot P^a \bmod p$를 계산, 증명자에게 전송한다.

② 증명자는 T_1, T_2로부터 trial and error로 r을 결정한 후(만일 r을 찾을 수 없다면 랜덤하게 선택) 랜덤수 t를 선택하여 $D \equiv \alpha^r \cdot \beta^t$를 계산, 확인자에게 전송한다.

③ 확인자는 단계 ①에서 사용한 랜덤수 a를 전송한다.

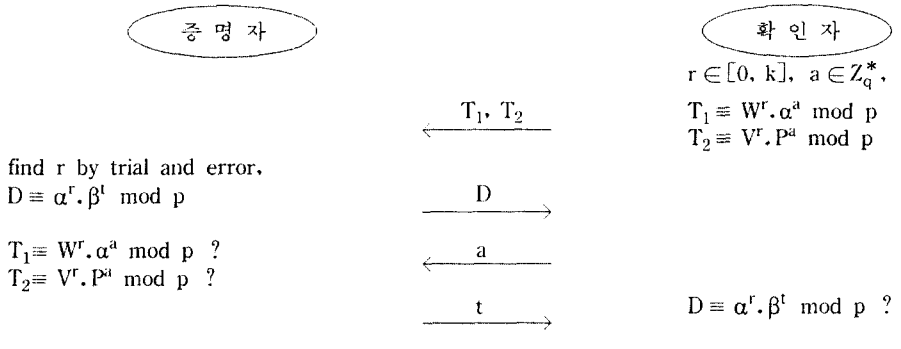


그림 7. Chaum의 disavowal protocol

④ 증명자는 이 a가 $T_1 \equiv W^r \cdot \alpha^u \bmod p$, $T_2 \equiv V^r \cdot P^u \bmod p$을 만족하는지 조사하여 이를 만족한다면 단계 ②에서 사용한 랜덤수 t를 전송한다. 이를 만족하지 않는다는 것은 확인자가 프로토콜을 따르지 않는다는 사실을 의미하므로 프로토콜을중단한다.

⑤ 확인자는 단계 ②에서 받은 D와 단계 ④에서 받은 t를 이용하여 $D \equiv \alpha^r \cdot \beta^t \bmod p$를 만족하는지를 검사한다.

위 프로토콜의 단계 ②에서 증명자가 r을 결정할 수 있는 것은 $\log_W V \neq \log_\alpha P$인 경우이다. 만일 $\log_W V = \log_\alpha P$가 성립한다면 결국 $T_2 \equiv T_1^S \bmod p$이므로 이같은 경우는 증명자의 계산능력에 관계없이 정보이론적으로 T_1, T_2는 r에 대한 정보를 전혀 제공하지 않기 때문이다. 따라서 단계 ⑤의 테스트를 통과한다면 그 서명은 증명자의 유효한 서명이 아니라는 사실이 증명되는 것이므로 confirmation protocol의 실패시의 두 가능성을 구분할 수 있다. 파라미터 k가 커지면 단계 ②에서 증명자의 계산량이 k에 비례하여 증가하므로 k를 임의로 크게 잡는 것은 불가능하며 실제로 약 1023정도가 적절하다. 여기서 $\log_W V \neq \log_\alpha P$인 경우 r을 계산하는 효율적인 방법중의 하나는 $T_1^S/T_2 \equiv (W^S/V)^r \bmod p$의 관계식에 주목하여 우선 r의 모든 가능한 값에 대해 $(W^S/V)^r \bmod p$를 계산하여 저장해 두고 이로부터 $T_1^S/T_2 \bmod p$의 값을 찾는 것이다. $(W^S/V)^r \bmod p$의 값은 확인자의 랜덤수 a와는 무관하므로 한번 계산하여 두면 프로토콜을 반복 시행하는 경우에도 계속 이용할 수 있을 것이다.

6. 수신자 지정 서명방식(Directed signature)

이 장에서는 지정된 수신자만이 서명을 인증할 수 있고 필요시 제 3 자에게 그 서명이 자신에게 발행된 유효한 서명임을 증명할 수 있게 함으로써 자신에게 발행된 서명의 남용을 통제할 수 있는 수신자 지정 서명방식을 다루기로 한다. 이는 서론에서도 언급했듯이 서명된 메세지가 수신자의 이해관계나 프라이버시에 관련된 내용인 경우 서명의 수신자가 서명의 사본들이 불법적으로 사용되는 것을 통제할 수 있도록 하자는 것이다. 서명자는 서명을 생성할 때 특정 수신자의 공개키를 결부시킴으로써 그 공개키에 대응하는 비밀키의 소유자만이 서명을 인증할 수 있도록 하고 또한 지정 수신자는 자신이 그 서명을 제시해야 할 필요가 있을 때에는 제 3 자에게 그 정당성을 증명할 수 있다.

Schnorr의 서명을 변형하면 이와 같은 수신자 지정 서명방식을 구성할 수 있다. 사용자 i가 사용자 j만이 확인할 수 있도록 메세지 m을 서명하여 보내고자 하는 경우 다음과 같이 서명을 생성할 수 있다.

① 사용자 i는 랜덤수 $R \in [1, q)$를 선택하여 $W \equiv \alpha^R \bmod p$, $K \equiv P_j^R \bmod p$를 계산한다.

② $Z \equiv \alpha^K \bmod p$, $E = h(W, Z, m)$를 계산하고 $Y \equiv RK + S_i E \bmod q$를 구하면 (Z, W, Y)가 메세지 m에 대한 서명이 된다.

③ 이를 받은 사용자 j는 $h(W, Z, m) = E$와 $W^{S_j} \equiv K \bmod p$를 계산하여 $\alpha^Y P_i^{-E} \equiv W^K \bmod p$를 만족하는지 검사함으로써 메세지 m에 대한 서명을 확인할 수 있다.

여기서 $K \equiv P_j^R \equiv W^{S_j} \bmod p$는 서명자와 지정 수신자만이 계산할 수 있으며 그외의 어떤 제 3 자도 (Z, W, Y)와 메세지 m으로부터 서명의 진위 여부를 판별할 수는 없으므로 undeniable signature와 마찬가지로 서명의 사본들이 남용되는 것을 막을 수 있다. 이 서명방식에서 특정한 K를 공개한다면 해당 메세지에 대한 서명은 보통의 디지탈 서명으로 바뀐다는 것을 알 수 있다(selectively convertible). 단계 ②에서 $Z \equiv \alpha^K \bmod p$를 서명의 일부로 포함시킨 것은 이 값이 아래에 설명되는 제 3 자에 대한 서명의 정당성 증명 프로토콜을 위해 필요하기 때문이다.

만일 메세지에 대한 비밀보장이 요구된다면 서명자는 서명을 생성한 후 수신자와의 공통의 비밀 랜덤수 K를 공개된 해쉬함수로 압축한 결과인 $h(K) = KS$를 세션키로 메세지를 암호화하여 전송할 수도 있다. 그러면 수신자 역시 KS를 계산할 수 있으므로 암호문을 복호화한 후 서명을 인증할 수 있을 것이다.

한편 디지탈 서명의 가장 중요한 기능 중의 하나인 부인방지 기능을 위해서는 이 서명이 문제가 되었을 때 서명자가 이를 부인할 수 없도록 서명의 수신자가

임의의 제 3 자에게 그 서명의 정당성을 증명할 수 있는 프로토콜이 필수적이다. 즉 수신자(사용자 j)는 제 3 자에게 $\alpha^{Y}P_i^{-E}(=V) \equiv W^{K} \bmod p$, $P_j \equiv \alpha^{S_j} \bmod p$를 만족하는 이산대수 K와 S_j를 알고 있다는 사실을 증명할 수 있어야 한다. 여기서 $K \equiv P_j^{R} \equiv W^{S_j} \equiv \alpha^{RS_j} \bmod p$이므로 비밀키 S_j를 알면 K를 계산할 수 있는 것은 당연하지만 이 K 값을 알고 있다고 해서 그가 서명의 수신자임이 증명되는 것은 아니다. 예를 들어 서명자가 사용자 j에게 서명한 사실을 부인하기 위해 K를 제 3 자에게 은밀히 누출시킬 수 있을 것이다. 따라서 사용자 j가 그 서명의 정당한 수신자임을 제 3 자에게 증명하기 위해서는 그가 $V \equiv W^{K} \bmod p$와 $P_j \equiv \alpha^{S_j} \bmod p$를 만족하는 두 이산대수 K와 S_j를 모두 알고 있다는 사실을 증명하도록 해야 할 것이다. 이러한 목적으로 사용될 프로토콜로 앞에서 소개한 Chaum의 zero-knowledge confirmation protocol을 변형하여 다음과 같은 프로토콜을 구성할 수 있다(확인자=제 3 자, 증명자=서명의 수신자 즉 사용자 j : 그림 8 참조).

[제 3 자에 대한 증명 프로토콜]

① 확인자는 세 랜덤수 a, b, $c \in [1, q)$를 선택하여 $T_1 \equiv W^{a} \cdot P_j^{c} \bmod p$, $T_2 \equiv \alpha^{b} \cdot Z^{-c} \bmod p$를 계산하여 $\{T_1, T_2\}$를 증명자에게 전송한다.

② 증명자는 랜덤수 $t_1, t_2 \in [1, q)$를 선택하여 $D_1 \equiv T_1 \cdot W^{t_1} \bmod p$, $D_2 \equiv T_2 \cdot \alpha^{t_2} \bmod p$, $D_3 \equiv D_1^{K} \cdot D_2^{S_j} \bmod p$를 계산, 확인자에게 전송한다.

③ 확인자는 자신의 랜덤수 a, b, c를 서명자에게 전송한다.

④ 증명자는 확인자로부터 받은 a, b, c를 이용

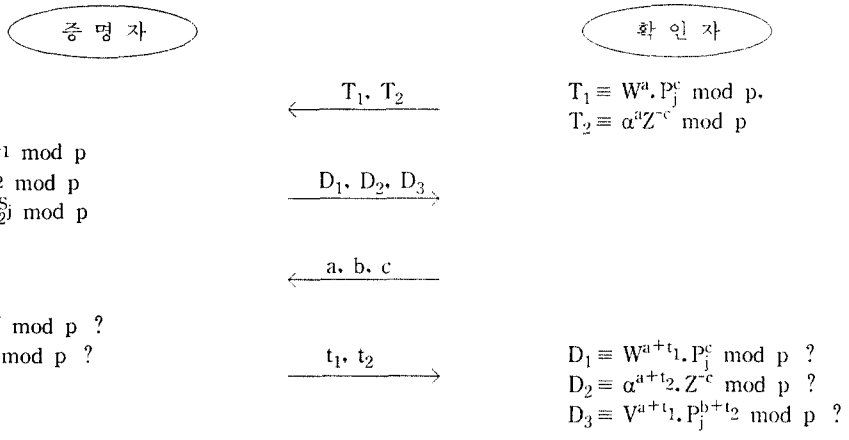


그림 8. 제 3 자에 대한 증명 프로토콜 1

하여 $T_1 \equiv W^{a} \cdot P_j^{c} \bmod p$, $T_2 \equiv \alpha^{b} \cdot Z^{-c} \bmod p$가 성립하는지를 검사하여 단계 ①에서 확인자가 적법한 challenge값을 전송했는지를 확인한다. 만일 이 합동식이 성립한다면 자신의 랜덤수 t_1, t_2를 확인자에게 전송하고 그렇지 않다면 프로토콜을 종료한다.

⑤ 확인자는 증명자로부터 받은 t_1, t_2를 이용하여 $D_1 \equiv W^{a+t_1} \cdot P_j^{c} \bmod p$, $D_2 \equiv \alpha^{b+t_2} \cdot Z^{-c} \bmod p$, $D_3 \equiv V^{a+t_1} \cdot P_j^{b+t_2} \bmod p$가 성립하는지를 조사한다.

위의 프로토콜에서도 Chaum의 프로토콜과 마찬가지로 K와 S_j를 모르는 공격자가 이 프로토콜을 성공적으로 통과할 수 있는 가능성은 1/q로 이들을 랜덤하게 추측하는 것과 마찬가지라는 것을 보일 수 있으며 또한 유사한 방법으로 simulator를 구성할 수 있다(참고문헌 12.의 증명과정 참조). 또한 위와 같은 zero-knowledge protocol 대신에 아래의 간단한 challenge-response protocol을 이용할 수도 있다(그림 9 참조). 물론 이 프로토콜이 zero-knowledge가 될 수는 없으나 증명자의 비밀키에 대한 어떤 유용한 정보도 누출되지 않는다는 것은 쉽게 알 수 있다. 또한 S_j를 모르는 공격자가(어떤 경로로든 K를

알고 있다고 가정하더라도) 확인자가 계산할 E값을 얻기 위해서는 해쉬함수 h의 일방성을 깰 수 있어야 하므로 안전성 조건에도 문제가 없음을 알 수 있다 (증명자의 응답은 적법한 증명자와 확인자만이 계산할 수 있는 D에 대한 K를 비밀키로 하는 Schnorr의 서명으로 구성된다).

① 확인자는 랜덤수 $\rho \in [1, q)$를 선택하여 $T \equiv \alpha^{\rho} \bmod p$를 계산, 증명자에게 전송한다.

② 증명자는 랜덤수 $R_j \in [1, q)$를 선택하여 $Z \equiv W^{R_j} \bmod p$를 계산하고 $D \equiv W^{S_j} \bmod p$, $E=h(Z, D)$, $Y \equiv R_j + KE \bmod p$를 계산, (Z, Y)를 확인자에게 전송한다.

③ 확인자는 $P_j^{\rho} \equiv D$, $h(Z, D)=E$를 계산하여 $W^Y V^{-E} \equiv Z \bmod p$가 성립하는지를 조사한다.

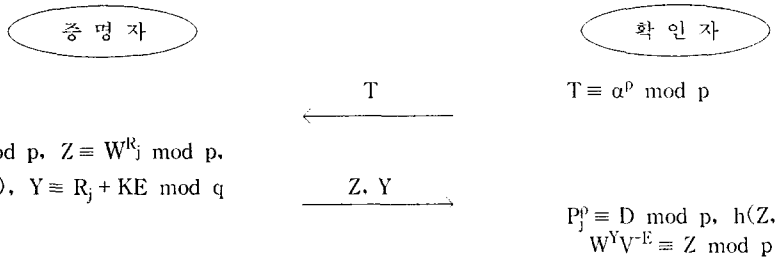


그림 9. 간단한 Challenge-Response protocol

다음에는 위에서 소개한 방식 보다 간단한 수신자 지정 서명방식으로 다음의 프로토콜을 생각해 보자.

① 사용자 i는 랜덤수 $R \in [1, q)$를 선택하여 $W \equiv \alpha^R \bmod p$, $K \equiv P_j^R \bmod p$를 계산한다.

② 이제 $E=h(W, m)$를 구한 후 $Y \equiv K+R+S_iE \bmod q$를 계산하면 (W, Y)가 메세지 m에 대한 서명이 된다.

③ 이를 받은 사용자 j는 $h(W, m)=E$와 $W^{S_j} \equiv K \bmod p$를 계산하여 $\alpha^{Y-K} P_i^{-E} \equiv W \bmod p$를 만족하는지 검사함으로써 메세지 m에 대한 서명을 확인할 수 있다.

이 서명방식에서 특정 서명과 관련된 K를 공개하더라도 비밀키에 대한 아무런 정보도 누출되지 않으므로 특정한 메세지에 대한 수신자 지정 서명을 일반적인 서명으로 바꿀 수 있음을 알 수 있다. 서명의 수신자인 사용자 j가 메세지 m에 대한 서명 (W, Y)가 사용자 i에 의해 그에게 발행된 것임을 제 3 자에게 증명하기 위해서는 $V(\equiv \alpha^Y P_i^{-E} W^{-1}) \equiv \alpha^K \bmod p$와 $P_j \equiv \alpha^{S_j} \bmod p$를 만족하는 이산대수 K와 S_j를 알고 있다는 사실을 증명할 수 있어야 한다. 이를 위해 Chaum-Everste-Graaf의 Protocol 2(multiple discrete log)[14]의 t-라운드 프로토콜을 병렬로 처리한 다음과 같은 프로토콜을 사용할 수 있다(그림 10 참조). 물론 이 병렬버전의 프로토콜은 Schnorr의 신분 인증 방식과 마찬가지로 zero-knowledge가 되지는 않는다.

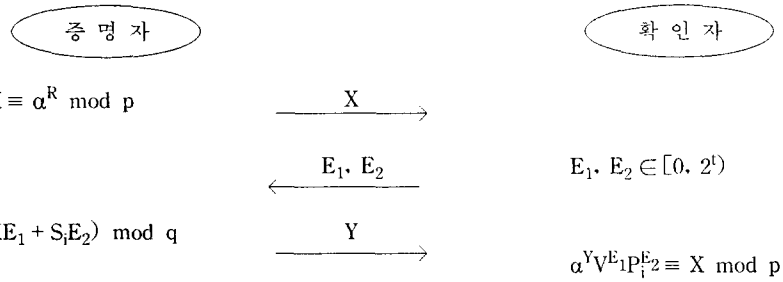


그림 10. 제 3 자에 대한 증명 프로토콜 2

① 증명자는 랜덤수 $R \in [1, q)$을 선택한 후 $X \equiv \alpha^R \bmod p$를 계산하여 확인자에게 보낸다.

② 확인자는 랜덤수 $E_1, E_2 \in [1, 2^t-1]$를 선택하여 증명자에게 전송한다.

③ 증명자는 $Y \equiv R-(KE_1+S_iE_2) \bmod q$를 계산하여 확인자에게 전송한다.

④ 확인자는 $\alpha^Y V^{E_1} P_i^{E_2} \equiv X \bmod p$가 성립하는지를 검사한다.

이상의 5, 6장에서는 서명자나 특정 확인자가 서명의 확인과정에 직접 개입하도록 하여 서명의 남용을 통제할 수 있도록 함으로써 일반적인 서명을 사용했을 때 발생될 수 있는 서명의 불법 사용을 방지할 수 있는 응용 지향적인 특수한 디지탈 서명법들을 제시하였다. 즉 최근에 와서 활발히 연구되고 있는 서명방식으로 서명의 인증을 위하여는 서명자와의 대화가 필요하도록 함으로써 서명자가 자신이 발행한 서명의 불법 사용을 통제할 수 있는 (selectively) convertible undeniable signature와, Okamoto의 non-transitive signature의 개념에서 출발하여 그 결함을 보강하고 이를 일반화시킨 것으로 서명의 수신자가 자신에게 발행된 서명의 사용을 통제할 수 있도록 하는 수신자 지정 서명방식(directed signature)을 Schnorr의 서명을 이용하여 구현하여 보았다. 이 두가지 서명법 모두가 서명의 사본들의 불법사용을 통제할 수 있도록 한다는 데에서는 공통되지만 전자의 경우는 서명자의 이해관계나 프라이버시가 관계되는 응용들을 지향한 것인 반면 후자의 경우는 서명된 메세지의 내용이 지정된 수신자의 이해관계나 프라이버시와 밀접한 관계가 있는 응용들에 적절하다고 할 수 있겠다.

7. 결 론

본 논문에서는 Schnorr의 신분인증 및 디지탈 서명방식을 변형, 확장시킨 몇가지 프로토콜들을 제시하였다. 두 사용자 쌍방이 서로의 신분을 인증함과 동시에 후속되는 통신에서의 세션키를 분배할 수 있는 상호 신분인증 프로토콜로 Schnorr 방식을 변형한 3-move mutual anthentication scheme을 제안하였다. 또한 Schnorr의 서명을 이용하면 쉽게 회의용의 세션키를 분배할 수 있음도 보였다. 서명의 불법 사용을 서명자가 통제할 수 있는 undeniable signature로 Schnorr의 서명을 변형하여 하나의 공개키만으로(selectively) convertible한 방식을 구성하여 보았다. 또한 특정한 수신자만을 상대로 서명을 발행하여 수신자가 자신에게 발행된 서명을 통제할 수 있는 수신자 지정 서명방식을 제안하였으며 그 서명의 정당성을 제3자에게 증명할 수 있는 프로토콜들도 제시하였다. 이러한 서명방식들은 보통의 서명방식이 누구나 인증 가능하다는 사실로 인해 이를 악용 할 수 있는 가능성이 높다는 사실에 근거하여 서명자나 특정 수신자의 개입없이는 그 서명을 인증할 수 없도록 함으로써 서명자나 수신자의 프라이버시를 높여줄 수 있으므로 여러가지 응용들에서 매우 유용하게 사용될 수 있을 것이다.

참 고 문 헌

1. A. Fiat and A. Shamir, "How to prove yourself: Practical solutions of identification and signature problems," Proc. Crypto'86.

2. L.S. Guillou and J.J. Quisquater, "A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory," Proc. Eurocrypt'88.

3. C.P. Schnorr, "Efficient identification and signatures for smart cards," Proc. Crypto'89; J. Cryptology, Vol. 4, 1991, pp. 161-174.

4. J. Brant, I. Damgard, P. Landrock, and T. Pedersen, "Zero-knowledge authentication scheme with secret key exchange," Proc. Crypto'88.

5. C.G. Gunther, "An identity-based key-exchange protocol," Proc. Eurocrypt'89.

6. F. Bauspieß and H.J. Knobloch, "How to keep authenticity alive in a computer network," Proc. Eurocrypt'89.

7. 이윤호, 양형규, 장청룡, 원동호, "영지식 증명을 이용한 키 분배방식에 관한 연구," '91 정보보호학술발표논문집, pp. 85-94.

8. P.J. Lee, "Secure access control for public networks," Proc. Auscrypt'90.

9. W. Diffie and M. Hellman, "New directions in cryptography," IEEE Trans. Inform. Theory, IT-22, 1976, pp.644-654.

10. D. Chaum and H. Antwerpen, "Undeniable signature," Proc. Crypto'89.

11. D.Chaum. "Zero-knowledge undeniable signatures," Proc. Eurocrypt'90.

12. J. Boyar, D. Chaum and I. Damgard, "Convertible undeniable signatures," Proc. Crypto'90.

13. T. Okamoto and K. Ohta, "How to utilize the randomness of zero-knowledge proofs," Proc. Crypto'90.

14. D. Chaum, J.H. Everste, and J. Graaf, "An improved protocol for demonstrating possession of discrete logarithms and some generalization," Proc. Eurocrypt'87.

15. J.M. Pollard, "Monte Carlo methods for index computation mod p," Math. Comp. 32, 1978, pp.918-924.

16. M. Girault, "Self-certified public keys," Proc. Eurocrypt'91.

17. Y. Desmedt, C. Goutier, and S. Bengio, "Special uses and abuses of the Fiat-Shamir passport protocol," Proc. Crypto'87.

18. S. Bengio, G. Brassard, Y. Desmedt, and C. Goutier, "Secure implementation of identification systems," J. Cryptology, No. 3, Vol. 4, 1991, pp.175-183.

19. T. Beth and Y. Desmedt, "Identification token-or: solving the chess grandmaster problem," Proc. Crypto'90.

20. K. Koyama and K.Otha, "Identity-based conference key distribution systems." Proc. Crypto'87.

21. Y. Yacobi, "Attack on the Koyama-Otha identity based key distribution scheme," Proc. Crypto'87.

22. K. Koyama and K. Otha, "Security of improved identity-based conference key distribution systems," Proc. Eurocrypt'88.

23. T. Chikazawa and T. Inoue, "A new key sharing system for global telecommunication," Proc. Gobecom'90.

24. A. Simbo and S. Kawamura, "Cryptanalysis of several conference key distribution schemes," Proc. Asiacrypt'91.

25. T. Chikazawa and A. Yamagishi, "Improved identity-based key sharing system for multiaddress communication," Elect. Letters, Vol. 28, No. 11, 1992, pp.1015-1017.

□ 著者紹介

林 采 薰(正會員)

1963年生
1989年 2月 韓國데이타通信(株) 技術本部 勤務
1989년 3月 서울大學校 電子工學科 學士
1992年 2月 浦項工科大學 電子電氣工學科 碩士
現 在: 浦港工科大學 電子電氣工學科 博士過程 在學中

李 弼 中(正會員)

1951年生
1974年 2月 서울大學校 電子工學科 學士
1977年 2月 서울大學校 電子工學科 碩士
1982年 6月 U.C.L.A System Science, Engineer
1985年 6月 U.C.L.A Electrical Engineering, Ph.D,
1980年 6月～1985年 8月 : Jet Propulsion Laboratory, Senior Engineer
1985年 8月～1990年 2月 : Bell Communications Research, M.T.S.
1990年 2月～現在 : 浦項工科大學 電子電氣工學科, 副教授