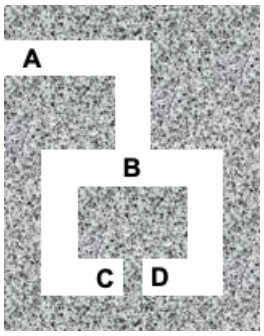
정보보호개론

제14장 고급 암호기술 2부: 고급 전자서명 기술 및 익명 인증 기술

1. 영지식 증명



<그림 14.1> 동굴의 비밀문

영지식(zero-knowledge) 증명은 알고 있는 것을 알려주지 않고, 알고 있음을 증명하여 주는 기법이다. 영지식 증명을 통해 참여자는 상대방이 해당 사실을 알고 있다는 것을 알게 되기 때문에 영지식이라는 용어는 오해의 소지가 있다. 영지식 증명의 또 다른 특징은 증명에 참여하더라도 증명자가 해당 사실을 알고 있다는 것을 다른 사용자에게 증명할 수 없다. 암호기술에서는 공개키 쌍에서 특정 공개키에 대응되는 개인키를 개인키를 노출하지 않고 알고 있음을 증명할 수 있다. 이와 같은 증명은 영지식 증명에 해당한다.

영지식 증명을 설명할 때 가장 많이 사용하는 예가 동굴의 비밀문이다[1]. 그림 14.1과 같은 구조의 동굴이 있다고 하자. 이 동굴의 가장 안쪽은 벽으로 막혀 있어 들어갔던 방향으로 다시 되돌아 나와야 한다. 하지만 Alice는 가장 안쪽 벽에 숨겨져 있는 비밀문을 여는 비밀주문을 알고 있다. Alice는 이 사실을 Bob에게 증명하고 싶지만, 비밀주문을 알려주고 싶지는 않다. 이를 위해 다음과 같은 과정을 n번 반복하여 증명하기로 하였다.

- 단계 1. Alice와 Bob은 모두 A 위치에 시작한다.
- 단계 2. Alice는 혼자 C나 D 위치로 이동한다. Bob은 동굴의 구조 특성 때문에 Alice가 어느 쪽으로 이동하였는지 알 수 없다.
- 단계 3. Bob은 일정 시간 이후 B 위치로 이동한다.
- 단계 4. Bob은 왼쪽 또는 오른쪽 중 한쪽 방향으로 나오라고 요청한다.
- 단계 5. Alice가 Bob이 요구한 방향으로 나오면 이 라운드는 성공한 라운드가 된다.

Alice가 실제 비밀주문을 알고 있다면 단계 3에서 들어간 방향과 상관없이 항상 성공할 수 있다.

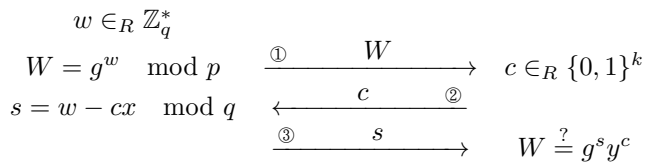
이 증명의 특징을 살펴보자. 단계 3에서 Alice가 들어간 방향과 Bob이 나오라고 요청한 방향이 같으면 비밀주문을 모르더라도 이 라운드는 성공할 수 있다. 즉, 비밀 주문을 모르더라도 한 라운드에서 성공할 수 있는 확률은 50%가 된다. 비밀주문을 모르는 상태에서 총 10번의 라운드를 수행하여 모든 라운드를 성공적으로 수행할 확률은 $1/2^{10}$이다. 따라서 10번을 연속 성공하였다면 Bob은 Alice가 비밀주문을 알고 있다고 확신(99.9%)할 수 있다. 이 증명의 또 다른 특징은 Bob이 10번의 과정을 녹화하여 다른 사람에게 보여주더라도 다른 사람은 Alice가 비밀주문을 알고 있다고 확신할 수 없다. 그 이유는 Alice와 Bob이 사전에 약속하고 녹화할 수 있기 때문이다.

영지식 증명의 요구사항은 다음과 같다.

- R1. 완전성(completeness): 증명하고자 하는 명제가 참이고 증명자와 확인자가 정직하면 증명은 통과해야 한다.
- R2. 건전성(completeness): 증명하고자 하는 명제가 거짓이면 부정한 증명자는 확인자를 속일 수 없어야 한다.
- R1. 영지식(zero knowledge): 증명하고자 하는 명제가 참이면 확인자가 얻게 되는 것은 그것이 참이다는 것 외에는 없어야 한다.

앞서 제시한 동굴 예처럼 증명자와 확인자 간에 상호작용을 통해 이루어지는 영지식 증명은 비전이성(non-transferability)을 만족한다. 비전이성이란 증명에 참여한 확인자가 다른 사람에게 증명자가 알고 있음을 증명할 수 없다는 것을 말한다.

1.1 이산대수 영지식 증명



<그림 14.2> 상호작용 방식의 이산대수 영지식 증명 프로토콜

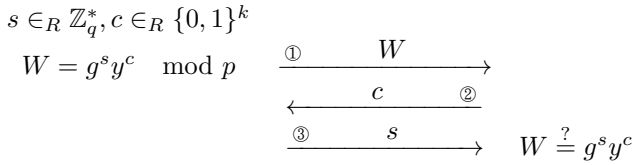
이산대수 기반 공개키 방식의 경우 이산대수 문제가 계산적으로 어려운 순환군의 생성자 g를 선택하고 그 군의 위수 범위 내에 있는 임의의 수 x를 개인키, $y = g^x$를 공개키로 사용한다. 이때 개인키의 소유자는 y에 대응되는 개인키를 알고 있다고 그림 14.2와 같은 프로토콜을 이용하여 개인키를 보여주지 않고 증명할 수 있다[2]. 그림 14.2의 프로토콜은 이산대수 문제가 계산적으로 어려운 $\mathbb{Z}_p^*$의 위수가 소수 q인 부분군 $G_q = \langle g \rangle$을 사용한다고 가정하자. 이 프로토콜의 완전성은 다음과 같이 확인할 수 있다.

$$W = g^w \stackrel{?}{=} g^s y^c = g^{w-cx} g^{xc} = g^w$$

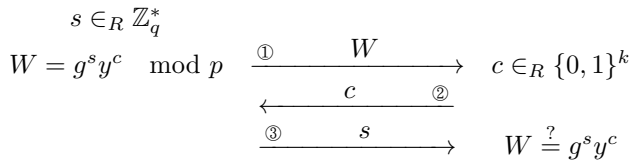
영지식 증명에 참여한 확인자도 증명자가 해당 이산대수를 알고 있다고 다른 사용자에게 재차 증명할 수 없다. 이것은 그림 14.2에 제시된 프로토콜의 트랜스크립트와 구분할 수 없는 트랜스크립트를 그림 14.3과 같이 만들어 제시할 수 있기 때문이다.

이 증명은 증명자가 c를 예측할 수 있으면 그림 14.4에 제시된 방법을 통해 기저 g에 대한 y의 이산대수를 모르더라도 증명이 가능하다. 따라서 c가 k 비트이면 c를 예측할 수 있는 확률이 $1/2^k$이며, 이 프로토콜의 안전성은 이 값에 의존한다.

이 증명의 또 다른 문제는 증명자가 항상 다른 w를 선택하여 프로토콜을 진행해야 한다. 만약 동일한 w를 두 번 사용하게 되면 공격자는 두 개의 트랜스크립트를 이용하여 y의 이산대수 x를 다음 두 식을 이용하여 계산할 수



<그림 14.3> 상호작용 방식의 이산대수 영지식 증명 프로토콜에 대한 가짜 트랜스크립트



<그림 14.4> c의 예측을 통한 상호작용 방식의 이산대수 영지식 증명 프로토콜

있게 된다.

$$\begin{aligned} s &= w - cx \\ s' &= w - c'x \end{aligned}$$

일반적인 영지식 증명은 증명자와 확인자 간에 상호작용하는 프로토콜이다. 하지만 대부분의 영지식 증명을 일방향 해시함수를 이용하여 그림 14.5와 같이 상호작용 없이 증명자가 홀로 증명하는 방식으로 바꿀 수 있다. 이산대수를 모르는 사용자가 $c = H_k(g||y||g^s y^c)$를 만족하는 c를 찾을 수 있으면 가짜 영시직 증명을 만들 수 있다. 이를 위해 임의의 s를 선택한 후 반복적으로 c를 바꾸어 가면서 $H_k(g||y||g^s y^c)$를 계산하여 이 값이 c가 되는 것을 찾아야 한다. 이것을 찾을 수 있는 확률은 상호작용 버전과 동일한 c의 길이에 의해 결정되기 때문에 두 버전의 안전성은 동일하다. 이 증명에서 c를 계산할 때 메시지 m을 포함하면 전자서명으로도 활용이 가능하다.

비상호작용 영지식 증명은 보통 누구나 확인할 수 있는 증명이 되므로 상호작용 버전이 가지고 있던 비전이성 특성이 사라진다. 하지만 비상호작용 영지식 증명을 만들 때 특정 확인자만 확인할 수 있도록 증명을 구성할 수 있다. 이 경우에는 비상호작용 버전도 여전히 비전이성이 제공된다.

영지식 증명은 어떤 값이 어떤 특성을 가지고 있다는 것을 그 특성을 보여주지 않고 증명하기 위한 암호기술로 많이 사용한다. 예를 들어 $A = g^a$, $B = h^a$가 있을 때 증명자는 a를 제시하지 않고 $\log_g A$와 $\log_h B$가 같음을 영지식으로 증명할 수 있다. 영지식 증명은 최근에는 프라이버시를 강화하기 위한 기술로도 많이 활용하고 있다. 예를 들어 영지식 증명을 이용하면 실제 나이를 제시하지 않고 성인임을 증명할 수 있고, 고가의 부동산 거래에서도 민감한 사적 데이터를 제시하지 않고 충분한 재산을 가지고 있음을 증명할 수 있다.

1.2 영지식 증명 관련 최신 기술 동향

비트코인의 등장으로 논문에서만 연구되던 수많은 암호 기술을 실제 현장에서 사용하고 있다. 이와 같은 기술 중 대표적인 것이 바로 영지식 증명이다. 특히, 암호화폐에서 사용자의 프라이버시를 더욱 강화하기 위한 목적으로 영지식 증명 기술을 사용하고 있다. 초기 영지식 증명은 증명할 수 있는 것이 제한적이었고, 증명하고자 하는 것에 따라 증명하는 방법이 달라졌다. 하지만 기술이 발달하여 현재는 더 이상 제한적이지 않으며, 범용적으로 사용할 수 있는 기술로 진화하였다.

현재 암호화폐에서 활용되고 있는 영지식 증명은 zk-SNARK(zero-knowledge Succint Non-interactive ARgument of Knowledge)[3]와 zk-STARK(zero-knowledge Succint Transparent ARgument of Knowledge)[4]이다. 여기서 succint라는 것은 증명이 간결하다는 것이다. 증명이 간결하기 때문에 매우 빠르게 검증할 수 있다. 그런데

$$
\begin{array}{c}
w \in_R \mathbb{Z}_q^* \\
W = g^w \mod p \\
c = H_k(g||y||W) \\
s = w - cx \mod q
\end{array}
\quad \xrightarrow{①} \quad c, s \quad \longrightarrow \quad c \stackrel{?}{=} H_k(g||y||g^s y^c)
$$

<그림 14.5> 비상호작용 방식의 이산대수 영지식 증명

zk-SNARK는 증명자의 능력을 제한하고 있다. 즉, 증명자가 무한한 컴퓨팅 자원과 시간이 있으면 증명을 위조할 수 있다. zk-SNARK는 특정 수학적 기반의 영지식 증명이기 때문에 일반 내용을 영지식으로 증명할 때 바로 사용할 수 없다. 이 때문에 front-end, back-end 개념으로 확장되었다. zk-SNARK에서 front-end는 컴파일러 역할을 하며, 각 응용에서 증명하고자 하는 것을 zk-SNARK를 구현한 back-end가 처리할 수 있도록 번역하여 준다.

2. 은닉채널

합법적인 메시지에 송신자와 수신자만이 알 수 있는 내용을 포함하여 교환할 수 있으면 송신자와 수신자 사이에 **은닉 채널**(subliminal channel)이 있다고 말한다. Simmons는 합법적인 전자서명 메시지에 은닉 채널을 포함할 수 있다는 것을 발견하였다[5]. 정보보호 측면에서는 서명에 은닉채널을 포함할 수 없도록 만들어야 한다.

3. 특수 서명 프로토콜

3.1 부인불가 서명

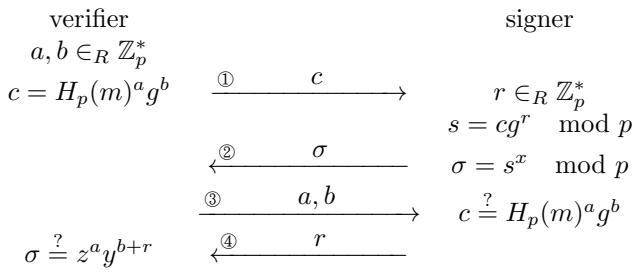
보통의 전자서명은 서명의 확인키를 가지고 있으면 누구든지 서명을 확인할 수 있다. 이와 달리 **부인불가 서명**(undeniable signature)은 서명자의 협조가 없으면 서명을 확인할 수 없다[6]. 이 기능을 이용하면 서명을 확인할 수 있는 사용자를 제한할 수 있다. 하지만 확인을 해야 하는 확인자 입장에서는 서명자의 도움이 없으면 확인할 수 없기 때문에 서명자가 서명하였음에도 불구하고 부인할 수 있다. 따라서 부인불가 서명은 이름에 나타나 있듯이 서명자는 확인자와 프로토콜을 진행하여 서명의 유효성을 확인해 줄 수 있을 뿐만 아니라 본인이 서명한 것이 아니라면 부인을 할 수 있다. 부인한다는 것은 주어진 서명값이 메시지 m에 대한 자신의 서명이 아니라는 것을 증명할 수 있다는 것이다.

부인불가 서명은 서명 알고리즘, 서명 확인 프로토콜, 부인 프로토콜 3가지로 구성되어 있다. 부인불가에서 서명 확인 프로토콜은 다음과 같은 요구사항을 가지고 있다.

- R1. 확인자는 주어진 서명이 특정 메시지에 대한 서명자의 유효한 서명임을 확인할 수 있다.
- R2. 프로토콜에 참여한 확인자도 다른 사람에게 주어진 서명이 특정 메시지에 대한 서명자의 유효한 서명임을 확인해 줄 수 없다.

부인 프로토콜도 유사한 요구사항을 가지고 있다. 즉, 해당 서명이 특정 메시지 m에 대한 본인의 서명이 아닌 경우 이를 확인자에게 확인시켜 줄 수 있다. 하지만 확인자는 이를 다른 사람에게 재차 확인해 줄 수는 없다.

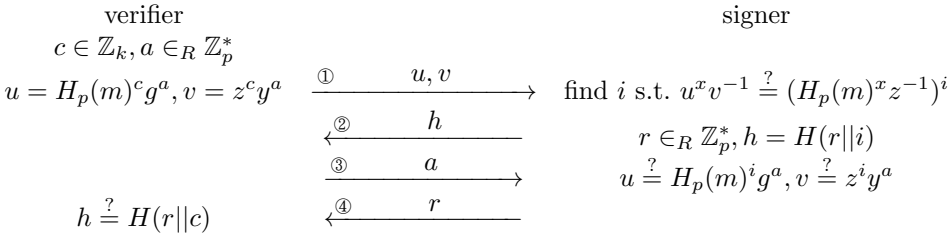
Chaum의 부인 불가 서명에서 서명자의 서명키는 $x \in \mathbb{Z}_p^*$이며, 공개키 $y = g^x \mod p$이다. 메시지 m에 대한 서명 프로토콜은 $z = H_p(m)^x \mod p$이다. 확인 프로토콜은 그림 14.6과 같이 진행된다. 여기서 다음이 성립하므로



<그림 14.6> Chaum의 부인 불가 서명의 서명 확인 프로토콜

이 프로토콜은 정확하다.

$$\sigma = s^x = (cg^r)^x = (H_p(m)^a g^b)^x g^{rx} = (H_p(m)^x)^a (g^x)^{b+r} = z^a y^{b+r}$$



<그림 14.7> Chaum의 부인 불가 서명의 서명 부인 프로토콜

Chaum의 부인 불가 서명의 서명 부인 프로토콜은 그림 14.7과 같다. 여기서 $z \neq H_p(m)^x$임을 서명자는 증명하고 있다. 다음이 성립하므로

$$u^x v^{-1} = (H_p(m)^c g^a)^x (z^c g^{xa})^{-1} = (H_p(m)^x)^c (z^{-1})^c = (H_p(m)^x z^{-1})^c$$

$z = H_p(m)^x$이면 $u^x v^{-1} = 1$이므로 i를 찾을 수 없다. 거꾸로 $z \neq H_p(m)^x$일 경우에는 추측하는 방법 외에는 찾을 수 있는 방법이 없다. 이 프로토콜에서 k를 1,023으로 결정하였고, 이 프로토콜을 10번 수행한다면 z가 유효한 자신의 서명임에도 우연히 i를 찾아 모두 성공할 확률은 매우 낮다.

Boyar 등은 부인불가 서명을 일반 서명으로 전환할 수 있는 전환 가능 부인불가 서명(convertible undeniable signature)을 제안하였다[7]. 서명자가 특정한 값을 공개하면 부인불가 서명이 일반 서명으로 전환되며, 누구든지 서명자의 협조 없이 서명의 유효성을 확인할 수 있다.

3.2 지정된 확인자 서명

지정된 확인자 서명(designated confirmer signature)[8]은 부인불가 서명과 일반 서명의 절충안이다. 부인불가 서명은 서명자의 협조가 없으면 서명을 확인할 수 없는 반면에 일반 서명은 누구나 확인키를 이용하여 확인할 수 있다. 지정된 확인자 서명에서 확인자는 부인 불가 서명과 마찬가지로 서명자의 협조가 있으면 서명의 유효성을 확인할 수 있으며, 서명자가 지정한 사용자의 협조가 있어도 서명의 유효성을 확인할 수 있다.

3.3 프록시 서명

프록시 서명(proxy signature)은 서명자가 자신의 서명키를 지정된 프록시에게 주지 않고, 자신을 대신하여 서명할 수 있도록 해주는 기법이다[9]. 프록시 서명의 요구사항은 다음과 같다.

- R1. 구별가능성: 프록시 서명은 일반 서명을 구분할 수 있어야 한다.
- R2. 위조불가능성: 원 서명자와 지정된 프록시 서명자만 유효한 프록시 서명을 생성할 수 있어야 한다.
- R3. 프록시 서명자는 프록시 서명이 아닌 실제 서명은 할 수 없어야 한다.
- R4. 확인가능성: 확인자는 프록시 서명으로부터 원 서명자의 서명된 메시지에 대한 동의를 확인할 수 있어야 한다.
- R5. 식별가능성: 원래 서명자는 프록시 서명을 통해 프록시 서명자를 확인할 수 있어야 한다.
- R6. 부인불가능성: 프록시 서명자는 자신이 서명한 프록시 서명을 부인할 수 없어야 한다.

요구사항 2에 따라 원 서명자는 프록시 서명을 할 수 있으면 책임을 프록시 서명자에게 전가할 수 있는 문제점이 있다. 특히, 이 요구사항은 요구사항 6과 충돌되는 측면이 있다. 요구사항 5는 다수의 프록시가 존재할 수 있다는 것을 의미한다.

3.4 일괄 서명

일괄 서명(batch signature)는 한 서명자가 동시에 여러 개의 다른 메시지를 효율적으로 서명할 수 있도록 해준다. n개의 서로 다른 메시지에 대해 각각 개별 서명하는 것보다 저렴한 비용으로 n개의 메시지에 대한 서명을 생성할 수 있도록 해준다. Pavloski와 Boyd[10]는 머클 해시트리를 이용한 일괄 서명 기법을 제안하였으며, 이 기법은 문서의 개수와 상관없이 항상 한 번의 서명만 진행한다. 해시 트리에 포함된 특정 문서에 대한 서명을 제시하고 싶으면 루트에 대한 서명, 문서가 포함된 단말 노드부터 루트노드까지 가는 경로에 있는 형제 노드의 값을 제시해야 한다. 총 $\log n$개 정보를 루트에 대한 서명값과 함께 제시해야 하므로 확장성이 있는 기법이다.

3.5 다중 서명

다중 서명(multi-signature)은 n명의 서로 다른 서명자가 같은 메시지에 대해 서명하지만, 그 결과가 n개의 서명이 아니라 하나의 서명을 얻게 되는 서명 기법을 말한다. 따라서 n개의 서명 대신에 하나의 서명을 확인하여 n명이 서명한 사실을 확인할 수 있다. 다음은 Boldyreva가 겹선형 사상을 이용한 다중 서명 기법이다[11]. 이 기법은 위수가 소수 q인 타원곡선 군 $G_1 = \langle P \rangle$와 곱셈군 $G_2 = \langle g \rangle$를 사용하며, 두 군의 위수는 같다. $H : \{0,1\}^* \rightarrow G_1$는 충돌회피 해시함수이며, $\hat{e} : G_1 \times G_1 \rightarrow G_2$는 겹선형 사상이다.

- 각 사용자 i의 개인키, 공개키: $x_i \in_R \mathbb{Z}_q^*$, $Y_i = x_i P$
- 다중 서명 공개키: $Y = \sum_i Y_i = (\sum_i x_i)P$
- 메시지 m에 대한 각 사용자의 서명: $S_i = x_i H(m)$
- 다중 서명: $S = \sum_i S_i = (\sum_i x_i)H(m)$
- 다중 서명에 대한 확인: $\hat{e}(P, S) \stackrel{?}{=} \hat{e}(Y, H(m))$

참고로 P는 위수가 소수 q인 타원곡선 군의 생성자이고, $x \in \mathbb{Z}_q^*$가 주어졌을 때 xP는 P를 x번 더한 값이다. $Q = xP$일 때, 곱셈군에서 이산대수 문제와 마찬가지로 군 정보, Q, P가 주어지더라도 x를 찾는 것은 계산적으로 어렵다. 이를 타원곡선 기반 DH 계산 문제라 한다. 겹선형 사상 $\hat{e}$은 타원 곡선 군의 원소 두 개를 받아 곱셈군 원소로 매핑하여 주는 함수이며, $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$가 성립한다. 겹선형 사상은 원래 이산대수 문제를 사용하는 암호기술을 공격하기 위한 도구로 사용되었지만 지금은 암호 기술을 구현하는 기초 기술로 더 널리 사용하고 있다. 실제 Joux가 제한한 3자간 자체 강화 방식의 키 동의 프로토콜도 겹선형 사상을 이용하고 있다.

3.6 결합 서명

결합 서명(multi-signature)은 n명의 서로 다른 서명자가 n개의 서로 다른 메시지에 대해 서명하지만, 그 결과가 n개의 서명이 아니라 하나의 서명을 얻게 되는 서명 기법을 말한다. 따라서 n개의 서명 대신에 하나의 서명을 확인하여 n개의 서로 다른 메시지에 대한 n명의 서명을 확인할 수 있다. 다음은 Boneh 등이 겹선형 사상을 이용한 결합 서명 기법이다[12]. 이 기법에서 G_1, G_2, H, $\hat{e}$는 이전 절에서 제시한 Boldyreva의 다중 서명과 동일하다.

- 각 사용자 i의 개인키, 공개키: $x_i \in_R \mathbb{Z}_q^*$, $Y_i = x_i P$
- 결합 서명 공개키: $Y = \sum_i Y_i = (\sum_i x_i)P$
- 메시지 m_i에 대한 각 사용자의 서명: $S_i = x_i H(m_i)$
- 결합 서명: $S = \sum_i S_i = \sum_i x_i H(m_i)$
- 결합 서명에 대한 확인: $\hat{e}(P, S) = \prod_{i=1}^{n} \hat{e}(Y_i, H(m))$

3.7 일괄 확인

일괄 서명은 서명자의 서명 계산 비용을 줄이기 위한 수단이다. 다중 서명과 결합 서명은 서명 크기를 줄어주는 효과도 있지만 확인하는 비용을 줄여주는 측면도 있다. 따라서 다중 서명과 결합 서명은 일괄 확인 기능을 가지고 있는 서명 기법이다. 일괄 확인은 개별 서명을 별도 확인하지 않고, 결합한 다음에 한번에 확인할 수 있다. 보통 여러 개의 서명을 확인해야 하는 확인자는 확인 비용을 줄이기 위해 이들을 개별 확인하지 않고 결합한 후에 확인할 수 있다.

응용에 따라 차이가 있을 수 있지만 일괄 확인하였을 때 결과가 실패이면 어떤 서명 또는 서명들 때문에 실패하였는지 알아야 할 수 있다. 이와 같은 응용에서는 이 측면 때문에 일괄 확인이 원하는 효과를 얻기 힘들 수 있으며, 공격자는 일부로 유효하지 않은 서명을 포함해 방해 공격을 할 수 있다. 이 때문에 보통 개별 확인한 서명을 결합하여 결합 서명을 만들며, 최초 개별 확인한 사용자를 제외하고는 다른 사용자들은 매우 효과적으로 증명을 확인할 수 있도록 해주는 용도로 많이 사용한다. 블록체인을 이용한 암호화폐에서도 일괄 확인을 이용하여 블록에 포함된 모든 서명을 개별 확인하는 것이 아니라 이들 서명을 결합한 하나의 서명만 확인하여 한 블록에 있는 모든 서명의 유효성을 효율적으로 확인하는데 활용하고 있다.

4. 전자서명과 프라이버시

전자서명은 인증을 목적으로 하므로 본질적으로 서명한 주체를 명확하게 알 수 있어야 한다. 하지만 때에 따라 적법한 사용자가 서명한 것이지만 어떤 특정 사용자가 서명한 것인지 모르게 하고 싶을 수가 있다. 이때 사용하는 기술이 익명 인증 기술이다.

4.1 익명ID 또는 익명 인증서

전자서명은 공개키 기반 기술이며, 공개키 방식은 크게 인증서 기반 방식과 신원 기반 방식으로 나누어질 수 있다. 인증서 기반의 경우 인증서 주체 정보에 사용자의 실명을 보통 사용한다. 하지만 사용자의 실명 대신에 다양한 정보와 공개키를 바인딩하여 활용할 수 있다. 예를 들어 청소년들이 특정 사이트를 사용하기 위해서는 부모 동의가 필요한 경우가 있다. 자녀들이 부모 동의 없이 쉽게 해당 사이트를 사용할 수 없도록 가족 관계를 증명하는 인증서를 부모에게 발급할 수 있다, 자녀가 부모 동의가 필요한 사이트에 ID를 발급받고 싶을 때 이 인증서를 활용할 수 있다. 예를 들어 해당 사이트는 청소년의 부모에게 이메일이나 SMS를 보낼 수 있으며, 부모는 링크를 따라 가족 관계를 증명하는 인증서를 보내고, 대응되는 개인키로 ID 발급에 동의하는 서명을 할 수 있다.

프라이버시를 위해 인증기관이 실명 대신에 익명으로 인증서를 발급할 수 있다. 신원 기반 방식의 경우에는 사용자의 신원 정보 대신에 익명 정보를 이용하여 개인키를 발급하여 줄 수 있다. 이 방식은 항상 같은 익명이나 같은 인증서를 사용하면 불연결성을 제공하지 못한다는 문제점을 가지고 있다. 따라서 강한 프라이버시를 제공하기 위해서는 다수의 익명 인증서나 다수의 개인키를 발급받아야 하는 불편함이 있다. 사용자 입장에서는 다수의 인증서나 다수의 개인키를 관리 및 유지하여야 하며, 주기적으로 재발급을 받아야 하기 때문에 사용하는 것이 불편할 수 있다. 이와 같은 기법에서 조건부 익명성은 발급하는 기관이 실제 신원과 익명 쌍을 유지하여 제공할 수 있다. 하지만 이 경우에도 해당 기관이 익명을 철회할 수 있는 권한을 남용하지 못하기 위한 조치가 필요하다.

4.2 그룹 서명

그룹 서명(group signature)은 그룹 관리자와 그룹 멤버가 참여하며, 다음과 같이 진행한다[13]. 그룹 관리자는 각 그룹 멤버에게 그룹 개인키를 발급한다. 각 사용자가 받게 되는 키는 다르다. 각 그룹 멤버는 자신이 받은 그룹 개인키를 이용하여 그룹 서명을 할 수 있다. 그룹 공개키를 이용하면 누구나 그룹 서명을 확인할 수 있으나 어떤 멤버가 서명하였는지 알 수 없다.

그룹 서명의 요구사항은 다음과 같다.

- R1. 그룹의 멤버만 서명을 할 수 있다.
- R2. 확인자는 그룹의 멤버 중 한 명이 서명하였다는 것을 확인할 수 있지만 누가 실제로 서명하였는지는 알 수 없다.
- R3. 두 개의 그룹 서명이 주어졌을 때 동일한 멤버가 서명한 것인지 알 수 없다.
- R4. 그룹 멤버들이 공모하여도 다른 특정 그룹 멤버가 서명한 것으로 옮아맬 수 없다.
- R5. 분쟁이 발생한 경우에는 실제 서명자를 밝힐 수 있다.

요구사항 3은 불연결성을 제공한다는 것을 의미하므로 그룹 서명은 강한 프라이버시를 제공하는 기법이다. 또 요구사항 5 때문에 그룹 서명은 기본적으로 조건부 익명성을 제공하는 기법이다.

그룹 개념이므로 그룹 멤버의 변경을 적절히 처리할 수 있어야 한다. 특히, 그룹 멤버가 탈퇴하면 그 멤버는 더 이상 그룹 서명을 할 수 없어야 한다. 보통 철회된 멤버를 제외한 다른 모든 멤버들에게 값을 전달해 주며, 이들은 이 값을 이용하여 자신의 그룹 개인키를 갱신한다. 이처럼 철회는 확장성이 부족하여 응용에 따라 효과적인 철회 방법이 될 수 없다. CRL을 사용하는 경우도 있지만, 이 경우에도 CRL에 있는 철회된 사용자 수에 비례한 노력(공개키 연산)이 필요하다. 그룹 서명은 기존 일반 서명보다 상대적으로 계산 비용이 높거나 서명 값의 크기가 크다는 문제도 있다.

$$
\begin{array}{ccc}
r \in_R \mathbb{Z}_n^* & & \\
\tilde{m} = H(m)r^e \mod n & \xrightarrow{①\quad \tilde{m}} & \tilde{s} = \tilde{m}^d \mod n \\
s = \tilde{s}/r \mod n & \xleftarrow{\tilde{s}\quad ②} & \\
H(m) \stackrel{?}{\equiv} s^e \pmod n & &
\end{array}
$$

<그림 14.8> RSA 은닉서명

4.3 링 서명

링 서명(ring signature)[14]에서 각 사용자는 서명하기 전에 여러 개의 공개키를 이용하여 링을 만든다. 실제 서명은 자신의 개인키를 이용하게 되지만 확인자는 형성된 링에 포함된 멤버 중 누가 서명하였는지를 알 수 없다. 그룹 서명과 유사하지만, 링은 보통 사용자가 매번 직접 만들수 있기 때문에 사용자마다 또는 매 순간 사용하는 링이 다르다. 반면에 그룹 서명에서 그룹에 있는 멤버들은 동일 그룹 하에서 서명하게 된다.

링의 크기는 그룹의 크기보다 보통 작은 단위이며, 링은 항상 새롭게 형성될 수 있으므로 링 서명을 전달할 때 링 정보를 함께 전달하여 한다. 따라서 서명의 크기가 링의 크기에 비례한다고 할 수 있으며, 이것이 큰 단점이 될 수 있다. 특정 사용자가 만든 링 서명은 항상 해당 사용자의 공개키가 링에 포함되므로 서로 중첩된 것이 없는 링을 이용한 두 개의 서명은 서로 다른 사용자가 한 서명이 된다. 따라서 완벽하게 불연결성이 제공되는 기법은 아니다. 링 서명도 익명성 보장이 필요할 때 사용할 수 있는 서명기법이며, 철회가능 링 서명도 있다[15]. 현재 모네로 암호화폐에서 링 서명을 활용하고 있다.

4.4 은닉 서명

보통 서명자는 서명하는 내용을 확인하고 서명해야 한다. 하지만 **은닉서명**(blind signature)에서 서명자는 자신이 서명하는 내용을 모른 채 서명하게 된다. 은닉서명은 주로 전자화폐, 전자선거 등에서 사용자의 익명성을 제공하기 위해 사용된다. 특이한 점은 서명자의 익명성을 제공하는 것이 아니라 서명 받은 사용자의 익명성에 초점을 두고 있다. 은닉서명을 발급받은 사용자가 나중에 이 서명을 제시하면 서명자는 이것을 언제 누구에게 발급해 준 것인지 알 수 없다.

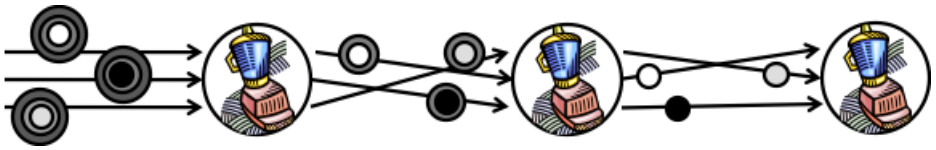
RSA 기반 은닉서명은 그림 14.8과 같다[16]. 메시지 m에 대한 서명을 받기 위해 메시지 m을 전달하는 대신에 랜덤 요소 r을 선택하여 메시지에 r^e을 곱하여 서명자에게 주게 된다. 서명자는 서명하는 메시지의 내용을 모르는 상태에서 서명하게 되며, 수신자는 받은 서명에서 r를 제거하면 메시지 m에 대한 서명자의 유효한 서명을 얻게 된다.

은닉서명은 서명자가 서명하는 내용을 볼 수 없으므로 매우 위험할 수 있다. 이런 위험을 줄이기 위해 cut-and-choose라는 기법을 사용할 수 있다. Cut-and-choose는 서양에서 두 명의 사용자가 케이크를 나눌 때 사용하는 방법으로써 케이크를 자르는 사람과 선택하는 사람이 다를 경우 자르는 사람은 손해를 보지 않기 위해 정확하게 반으로 자르게 된다는 논리를 담고 있다. 은닉서명에 이 기술에 적용하면 은닉서명을 받고 싶은 사용자는 10개의 메시지를 만들고, 이 메시지를 모두 은닉하여 서명자에게 전달한다. 서명자는 이 중 하나를 제외하고 나머지의 은닉 요소를 요구한다. 수신자가를 이를 전달하면 서명자는 10개 중 9개는 개방하여 올바르게 구성되어 있는지 확인한 후 문제가 없으면 개방하지 않은 값에 전자서명하여 은닉서명 프로토콜을 진행하게 된다. 따라서 수신자가 서명자를 속일 수 있는 확률은 10%가 된다. Cut-and-choose를 사용하면 수신자의 부정을 막을 수 있지만 교환되는 값들이 너무 많아질 수 있어 효율적이지 못하다. 현재는 cut-and-choose 기법을 사용하지 않아도 수신자가 부정할 수 없는 은닉서명[17]도 개발되어 있다.

5. 익명 통신

익명 통신(anonymous communication)은 트래픽 분석을 하더라도 교환되는 메시지의 소스 노드와 목적 노드를 식별할 수 없도록 메시지를 교환하는 통신 방식을 말한다. 일반 통신에서 노드는 IP 주소, MAC 주소처럼 노드를 특정할 수 있는 식별자를 사용하여 경로를 찾고 메시지를 목적 노드까지 전달한다. 하지만 익명 통신에서는 이 정보를 사용하더라도 이 정보를 바탕으로 소스 노드와 목적 노드를 식별할 수 없어야 한다. 이를 위해 소스 노드에서 목적 노드로 메시지를 바로 전달하지 않고, 메시지의 내용뿐만 아니라 헤더 정보까지 암호화한 다음 여러 중간 노드를 거쳐 전달한다.

익명 통신이 제공되기 위해서는 노드의 관찰을 통해 소스 노드나 목적 노드를 식별할 수 없어야 한다. 일반적으로 메시지는 그것의 내용, 크기, 송수신 시점에 의해 식별할 수 있다. 내용에 의한 식별은 메시지 내용뿐만 아니라 헤더도 암호화하여 식별하지 못하도록 해야 하며, 크기에 의한 식별은 채우기를 통해 항상 일정한 크기의 메시지를 전송해야 한다. 소스 노드의 익명성을 위해 수신된 메시지가 이전 노드에서 시작된 것인지 중계된 것인지 구분할 수 없어야 한다. 목적 노드의 익명성은 중계의 중단이나 응답 메시지를 보내는 행위 때문에 노출될 수 있다. 이와 같은 노출을 막기 위해 불필요한 메시지의 송신이나 지연 중계가 필요할 수 있다. 이 모든 것은 사용하는 네트워크 환경에 영향을 받는다. 사용하는 네트워크 환경에 따라 고려해야 하는 것이 다를 수 있고, 불가피하게 노출될 수밖에 없는 정보도 있다.



<그림 14.9> 믹스넷

현재 익명 통신에서 가장 널리 사용하는 기술은 믹스넷(mixnet)[18]과 양파 라우팅(onion routing)[19]이다. 둘 다 여러 겹으로 암호화한다는 측면에서 유사하다. 믹스넷은 그림 14.9와 같이 여러 개의 믹스로 구성되어 있으며, 각 믹스는 수신된 여러 겹으로 암호화된 데이터의 가장 바깥층을 제거하여 전송한다. 믹스가 수신한 메시지와 전송하는 메시지는 한 겹의 암호화를 제거하였기 때문에 메시지 자체로는 연결할 수 없지만, 시간을 이용하여 연결할 수 있다. 이 때문에 믹스는 일정 기간 수신된 메시지를 버퍼에 유지하고 충분한 수의 메시지를 확보하면 이들을 섞어 다시 중계한다. 이 때문에 믹스를 사용하면 통신 지연이 불가피하게 발생할 수밖에 없다. 믹스넷을 사용하기 위해 전송자는 중간 노드들이 복호화하여 다음 중간 노드에 전달할 수 있도록 여러 겹으로 메시지를 암호화하여야 한다. 시간뿐만 아니라 메시지 크기로 연결할 수 없도록 모든 사용자는 미리 약속된 크기의 메시지만 주고받아야 한다.

양파 라우팅은 소스 노드와 목적 노드가 메시지의 비밀성을 보장한 상태로 익명 통신을 할 수 있도록 해주며, 원하면 목적 노드로부터 메시지의 소스 노드를 숨길 수 있다. 소스 노드가 전달한 메시지는 지정된 중간 노드를 거쳐 목적 노드에 도달하게 되는데, 중간 노드들이 서로 협조하지 않으면 중간 노드를 포함하여 제3자에게 통신의 비밀성과 익명성을 제공해 준다. 소스 노드는 사용할 중계 노드의 개수(>= 3)와 순서를 결정하고 이 순서에 맞게 양파를 만들어 전달한다. 이 과정을 소스 노드가 하지 않고 프록시 노드가 대신할 수 있다. 이와 같이 양파 라우팅을 이용하면 제3자는 소스 노드가 프록시 노드와 통신하는 것으로 보이며, 목적 노드는 마지막 중계 노드와 통신하는 것으로 보이게 된다. 이때 마지막 중계 노드를 꼬리 노드라 하며, 이와 같은 방법으로 통신하기 때문에 꼬리 노드 기반 기법이라 한다.

양파 라우팅에서 양파는 이전 미스넷 설명과 동일하게 여러 겹으로 암호화되어 있다. 실제 소스 노드에서 목적 노드로 메시지를 전달할 때에는 소스 노드가 여러 겹으로 암호화하여 전달하며, 각 중간 노드는 수신한 암호문의 가장 바깥층을 제거하여 중계한다. 반대로 목적 노드에서 소스 노드로 메시지를 전달할 때에는 중간 노드들이 계속 받은 것을 암호화하여 중계하며, 소스 노드는 수신한 양파의 모든 겹을 제거하여 메시지를 얻게 된다.

중간 노드들이 양파를 중계하기 위해서는 수신한 양파를 전달할 다음 노드를 알아야 한다. 이 정보가 트래픽 분

석에 도움이 될 수 있으므로 각 중간 노드는 노드 간의 암호 채널을 구축하여 양파를 교환한다. 실제 양파는 암호화된 데이터이므로 헤더 정보만 노드 간의 암호 채널로 암호화한다. 헤더 정보 뿐만 아니라 메시지 크기에 의한 노출을 막기 위해 항상 정해진 크기의 셀 단위로 메시지를 교환하며, 양파가 전달되는 과정에서 양파 크기의 변화에 의한 노출을 막기 위해 각 중간 노드는 랜덤 채우기를 통해 항상 동일한 크기의 메시지를 다음 중계 노드로 전달한다. 또 전송 시점에 의한 노출을 줄이기 중계 노드가 믹스넷 역할(지연 통신, 가짜 트랙픽 발생)을 수행할 수 있다.

실제 양파 라우팅에서 메시지의 교환은 크게 양파 경로 설정, 데이터 교환 두 단계로 나누어진다. 양파 경로 설정은 메시지 중계에 사용할 중계 노드의 개수와 순서를 결정한 다음, 이 순서에 맞게 키 확립을 위한 양파를 구성하여 전달하게 된다. 예를 들어 X, Y, Z 노드를 중간 노드로 사용하기로 소스 노드가 결정하였다면 소스 노드는 다음과 같은 순서로 양파를 구성한다.

$$\begin{aligned} O_Z &= \{\_||K_Z||\_\}.+K_Z \\ O_Y &= \{Z||K_Y||O_Z\}.+K_Y \\ O_X &= \{Y||K_X||O_Y\}.+K_X \end{aligned}$$

소스 노드가 O_X를 X에 전달하면 X는 자신의 개인키로 메시지를 복호화하여 K_X를 얻을 수 있고, O_Y를 Y에게 전달한다. 이것이 꼬리 노드까지 반복된다. 꼬리 노드는 양파의 특성 때문에 자신이 꼬리 노드임을 알 수 있다.

양파를 통해 확립된 키는 실제 데이터를 암호화할 때 사용한다. 이 예에서는 K_X로부터 4개의 키를 만들어 각 방향마다 2개의 키를 사용하여 인증 암호화한다고 가정한다. 소스 노드가 메시지 M을 목적 노드에 전달하기 위해 다음과 같이 암호문을 생성하여 O_X를 X에 전달한다.

$$\begin{aligned} O_Z &= [\_, C_Z = \{M\}.K_{FZ}, \mathrm{MAC}.K'_{FZ}(C_Z)] \\ O_Y &= [Z, C_Y = \{O_Z\}.K_{FY}, \mathrm{MAC}.K'_{FY}(C_Y)] \\ O_X &= [Y, C_X = \{O_Y\}.K_{FX}, \mathrm{MAC}.K'_{FX}(C_X)] \end{aligned}$$

예를 들어 M은 한 블록 크기라 가정하고, 암호화하는 CTR 모드, MAC의 크기는 두 블록 크기라 가정하고, 필요한 헤더는 한 블록이라 하면 O_Z는 총 5블록 크기가 된다. 같은 방법으로 계산하면 O_Y는 9블록, O_X는 13블록이 필요하다. 이 경우 X가 O_Y를 전달할 때 4블록을 추가로 보내 O_X와 크기를 일치시켜 보내게 되며, Y는 O_X를 전달할 때 8블록을 추가로 붙여 전달한다.

반대 방향의 메시지는 다음과 같이 중계되면서 암호화되며, 이때에도 총 13블록이 되도록 랜덤 블록을 뒤에 추가하여 전달한다.

$$\begin{aligned} Z \to Y: &\quad O_Z = [\_, C_X = \{M\}.K_{BZ}, \mathrm{MAC}.K'_{BZ}(C_X)] \\ Y \to X: &\quad O_Y = [X, C_Y = \{O_Z\}.K_{BY}, \mathrm{MAC}.K'_{BY}(C_Y)] \\ X \to S: &\quad O_X = [Y, C_S = \{O_Y\}.K_{BX}, \mathrm{MAC}.K'_{BX}(C_X)] \end{aligned}$$

참고문헌

[1] Jean-Jacques Quisquater, M. Quisquater, M. Quisquater, M. Quisquate, L. Guillou, M. A. Guillou, G. Guillou, A. Guillou, G. Guillou, S. Guillou, "How to Explain Zero-Knowledge Protocols to Your Children," Advances in Cryotology, Crypto 1989, LNCS 435, pp. 628–631, 1990.

[2] C. P. Schnorr, "Efficient signature generation for smart cards," Journal Of Cryptology, Vol. 4, No. 3, pp. 239–252, 1991.

[3] Nir Bitansky, Ran Canetti, Alessandro Chiesa, Eran Tromer, "From Extractable Collision Resistance to Succint Non-Interactive Arguments of Knowledge and Back Again," Proc. of the 3rd Innovations in Theoretical Computer Science Conf. ACM, pp. 326-349, Jan. 2012.

[4] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, Michael Riabzev, "Scalable, Transparent, and Post-Quantum Secure Computational Integrity," IACR Cryptology ePrint Archive 2018-046, Mar. 2018

[5] Gustavus J. Simmons, “The Prisoners Problem and the Subliminal Channel,” Advances in Cryptology, CRYPTO ’83, pp. 51–67, 1984.

[6] David Chaum, Hans van Antwerpen, “Undeniable Signatures,” Advances in Cryptology, CRYPTO ’89, LNCS. 435, pp. 212–216, 1990.

[7] Joan Boyar, David Chaum, Ivan Damgard, Torben Pedersen, “Convertible Undeniable Signature,” Advances in Cryptology, CRYPTO ’90, LNCS 537, pp. 189–205, Springer, 1991.

[8] David Chaum, Hans van Antwerpen, “Desginated Confirmer Signatures,” Advances in Cryptology, Eurocrypt ’94, LNCS. 950, pp. 86–91, 1995.

[9] Masahiro Mambo, Keisuke Usuda, Eiji Okamoto, “Proxy Signatures for Delegating Signing Operation,” Proc. of the 3rd ACM Conf. on Computer and Communications Security, pp. 48–57, Jan. 1996.

[10] Christopher J. Pavlovski, Colin Boyd, “Efficient Batch Signature Generation Using Tree Structure,” Int’l Workshop on Cryptographic Techniques and E-Commerce, pp. 70–77, 1999.

[11] Alexandra Boldyreva, “Threshold Signatures, Multisignatures, and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme,” Public Key Cryptography, PKC 2003, LNCS 2567, pp. 31–46, Springer, 2003.

[12] Dan Boneh, Craig Gentry, Hovav Shacham, Ben Lynn, “Aggregate and Verifiably Encrypted Signatures from Bilinear Maps,” Advances in Cryptology, Eurocrypt 2003, LNCS 2656, pp. 416–432, Springer, 2003.

[13] David Chaum, Eugene van Heyst, “Group signatures,” Advances in Cryptology, Eurocrypt ’91. LNCS 547, pp. 257–265, Springer 1991.

[14] Ronald L. Rivest, Adi Shamir, Yael Tauman, “How to Leak a Secret,” Advances in Cryptology, Asiacrypt 2001, LNCS 2248, pp. 552–565, Springer, 2001.

[15] Dennis Y.W. Liu, Joseph K. Liu, Yi Mu, Willy Susilo, Duncan S. Wong, “Revocable Ring Signature,” Journal of Computer Science and Technology, Vol. 22, No. 6, pp. 785–794, Springer, Nov. 2007.

[16] David Chaum, “Blind Signatures for Untraceable Payments,” Advances in Cryptology, CRYPTO ’82, pp. 199–203, 1983.

[17] S. A. Brands, “Untraceable Off-line Cash in Wallets with Observers,” Advances in Cryptology, Crypto ’93, LNCS 773, pp. 302–318. Springer, 1994.

[18] David Chaum, “Untraceable Electronic Mail, Return Addresses, and Digital Pseudonym,” Communications of ACM, Vol. 24, No. 2, pp. 84–90, Feb. 1981.

[19] Paul F. Syverson, David M. Goldschlag, Michael G. Reed, “Anonymous Connections and Onion Routing,” Proc. of the IEEE Symp. on Security and Privacy, pp. 45–54, May 1997.

퀴즈

1. 익명 통신할 때 사용할 수 있는 믹스넷과 양파라우팅 관련된 다음 설명 중 틀린 것은?

   ① 둘 다 소스 노드가 메시지를 전송한 것과 목적 노드가 수신한 것은 노출되지만 서로 연결되지 않는다.

   ② 둘 다 소스 노드가 중간 노드를 결정하고 여러 겹으로 암호화해야 한다.

   ③ 중간 노드들이 모두 협력하여도 메시지의 송수신 노드를 알 수 없다.

   ④ 믹스넷에서 믹스 역할을 하는 노드는 메시지를 수신하면 즉시 중계하지 않고 버퍼에 정해진 수의 메시지가 도착할 때까지 기다린 후에 수신된 순서와 무관한 임의의 순서로 메시지를 중계한다.

2. 네트워크를 관찰하면 얻을 수 있는 정보가 많기 때문에 완벽한 익명 통신을 제공하기 어렵다. 기본적으로 통신 메시지를 서로 구분할 수 없어야 익명 통신을 제공할 수 있다. 다음 중 통신 메시지를 식별할 수 없도록 하기 위해 기본적으로 사용해야 하는 기법이 잘못된 것은?

   ① 메시지 내용을 통해 식별을 할 수 없도록 메시지는 기본적으로 암호화되어야 한다.

② 메시지 크기를 통해 식별할 수 없도록 채우기를 이용하여 모든 참여자가 항상 같은 크기의 메시지를 교환해야 한다.

③ 헤더 정보 없이는 통신 자체가 가능하지 않기 때문에 헤더 정보의 노출은 불가피하다.

④ 송수신 시점에 의해 식별할 수 없도록 불필요한 메시지의 송신이나 지연 중계가 필요하다.

3. 링 서명과 관련된 다음 설명 중 틀린 것은?

① 링 서명에는 그룹 개념이 없어 탈퇴 개념이 없다.

② 링 서명에서 링은 매번 다르게 만들어 사용할 수 있지만 서명자 자신은 항상 링에 포함되어야 한다.

③ 링 서명은 두 서명이 있을 때 같은 서명자가 서명한 것임을 알 수 없으며, 두 서명의 링 정보를 통해 노출되는 정보도 없기 때문에 강한 프라이버시를 제공하는 기법이다.

④ 링 서명은 링을 항상 새롭게 형성할 수 있으므로 링 정보를 항상 전달해야 하므로 서명의 크기가 링 크기에 비례한다.

4. 영지식 증명은 원래 증명자와 확인자 간에 프로토콜을 진행한다. 하지만 대부분의 영지식 증명은 상호작용 없이 증명자가 홀로 증명을 수행하는 비상호작용 버전으로 바꿀 수 있다. 다음에 제시된 영지식 증명 요구사항 중 상호작용 버전과 비상호작용 버전 간의 가장 큰 차이가 있는 것은?

① 완전성(completness): 증명하고자 하는 명제가 참이고, 증명자와 확인자가 정직하면 증명은 통과해야 한다.

② 건전성(soundness): 증명하고자 하는 명제가 거짓이면 부정한 증명자가 확인자를 속인 수 없어야 한다.

③ 비전이성(non-trasferability): 증명을 확인한 확인자는 증명을 다른 확인자에게 제시하여 명제가 참이라는 것을 증명할 수 없어야 한다.

④ 영지식(zero knowledge): 증명하고자 하는 명제가 참이면 확인자가 얻게 되는 것은 명제가 참이다는 것 외에는 없어야 한다.

연습문제

1. 영지식 증명은 실제로는 정보가 전혀 노출되는 것은 아니다. 항상 노출되는 정보는 무엇인지 설명하시오.

2. 1 절에서 영지식 증명을 설명하기 위해 동굴의 비밀문 예를 이용하였다. 이 예에서 증명자와 확인자는 설명된 과정을 여러 차례 반복해야 한다. 이보다 더 효과적인 방법을 제시하시오.

3. 1.1 절에 제시된 이산대수를 증명하는 영지식 프로토콜을 다음과 같이 바꾸었다.

$$\begin{array}{l} \qquad\qquad w \in_R \mathbb{Z}_q^* \\ W = g^w \mod p \\ s = w + x \mod q \end{array} \quad \xrightarrow{\qquad W, s \qquad} \quad Wy \stackrel{?}{=} g^s$$

이 프로토콜의 문제점을 설명하시오.

4. 여러 서명을 각각 확인하지 않고 결합하여 한번 확인해주는 기법이 있다. 이 기법의 근본적인 문제점을 설명하시오.

5. 익명인증서나 익명ID를 이용할 수 있다. 이 경우 하나의 익명인증서 또는 익명ID를 사용할 경우의 문제점을 설명하시오.

6. 익명성을 논할 때 그룹 개념이 꼭 필요하다. 익명성에서 필요한 그룹이란 무엇이며, 그룹의 크기가 어떤 의미가 있는지 설명하시오.