유효 기간을 갖는 포워드-시큐어 대리 서명†

김상희*○ 조태남* 이상호* 채기준* 박원주** 나재훈**
*이화여자대학교 컴퓨터학과, **ETRI 네트워크보안연구부
{kshee, tncho, shlee, kjchae}@ewha.ac.kr, {jwpark, jhnah}@etri.re.kr

Time-Limited Forward-Secure Proxy Signature

Sanghee Kim*○ Taenam Cho* Sang-Ho Lee* Kijoon Chae* Wonju PARK** Jaehoon Nah**
*Dept. of Computer Science and Engineering, Ewha Womans University
**Network Security Department, ETRI

요 약

대리 서명이란 원 서명자가 대리 서명자에게 서명 권한을 위임하여, 대리 서명자가 원 서명자를 대신해서 서명을 생성하는 것이다. 일반적으로 대리 서명자가 위임받은 권한은 유효 기간을 갖는다. 위임 정보에 위임 기간을 포함시키는 방법으로는 대리 서명자의 서명 생성 시간을 알 수 없기 때문에 유효 기간이 만료된 대리 서명자의 서명 위조를 막을 수 없고, 위임 기간 중에 대리 서명키가 노출되었을 경우 정당한 대리 서명자가 과거에 생성한 서명의 타당성을 보호하지 못한다. 본 논문에서는 기존 대리 서명의 보안 요구사항을 만족하면서도 원 서명자가 시간 관련 파라미터를 제어함으로써 위임 기간이 만료된 대리 서명자의 서명 위조를 막고, 서명키 노출 문제를 해결하기 위한 포워드-시큐어 서명 방법을 제안하였다.

1. 서 론

대리 서명이란 원 서명자가 대리 서명자에게 서명 권한을 위임하여, 대리 서명자가 원 서명자를 대신해서 서명을 생성하는 것이다. 수신자는 대리 서명을 검증하는 과정에서 대리 서명자의 정당한 서명임을 검증할 수 있어야 할 뿐만 아니라 원 서명자의 위임 동의를 확인할 수 있어야 한다.

일반적으로 대리 서명자가 위임받은 권한은 한시적으로만 유효하다. 위임 기간이 만료되면, 원 서명자는 권한 취소 프로토콜을 사용하여 위임했던 권한을 취소할 수 있다. 그러나 원 서명자가 권한을 위임할 당시에 위임 정보에 위임 기간을 포함시켜서 대리 서명자가 그 기간 동안만 서명을 생성할 수 있도록 하면 별도의 권한 취소 프로토콜을 사용하지 않아도 된다[1][2]. 그러나 위임 정보에 위임 기간을 명시적으로 표시한다고 하더라도 대리 서명자가 서명을 생성한 시간을 알 수 없기 때문에, 위임 기간이 만료된 대리 서명자의 서명 위조를 막을 수 없다.

또한, 서명키가 노출되었을 경우, 과거에 생성된 정당한 서명과 위조된 서명을 구분할 수 없다. 이러한 서명키 노출 문제를 해결하기 위하여 연구되고 있는 포워드-시큐어(Forward-Secure) 서명은 서명(검증)키의 사용 기한을 설정하고 현재의 서명키가 노출되더라도 이로부터 과거의 서명키를 유도해내지 못하게 함으로써 과거에 생성된 정당한 서명을 보호하는 방식이다.

위임 기간이 만료된 대리 서명자의 서명 위조를 막기 위한 방법으로 타임-스탬프(time-stamp)를 이용한 대리 서명 방법이[3] 제안되었다. 이 방법은 포워드-시큐어한 속성은 만족하지만 서명에 시간 정보가 포함되기 때문에 대리 서명자와 수신자는 정확한 시간 정보를 얻을 수 있는 메커니즘이 필요하며, 매 서명마다 타임-스탬프를 생성하여 인증된 저장 공간에 공고해야 한다.

† 본 논문은 한국전자통신연구원 네트워크보안연구부 위탁연구과제에 의한 것임.

본 논문에서는 원 서명자가 위임 기간을 제어함으로써 대리 서명자와 수신자가 정확한 시간 정보를 사용하지 않고 유효 기간을 제공하며, 보안 파라미터를 사용하여 구간별로 포워드-시큐어한 속성을 만족시키는 새로운 대리 서명을 제안한다.

2. 관련 연구

2.1 대리 서명의 보안 요구 사항[2][4]

① 강력한 위조 불가능성(Strong Unforgeability) : 대리 서명자만이 대리 서명을 생성할 수 있어야 한다.
② 검증성(Verification) : 검증자는 대리 서명으로부터 원 서명자의 위임 동의를 확인할 수 있어야 한다.
③ 강력한 신원 확인성(Strong Identifiability) : 누구나 대리 서명으로부터 대리 서명자의 신원을 확인할 수 있어야 한다.
④ 강력한 부인 불가능성(Strong Undeniability) : 대리 서명자는 자신이 서명한 사실을 부인할 수 없어야 한다.
⑤ 오용 방지(Prevention of Misuse) : 대리 서명자는 원 서명자로부터 위임받은 권한 이외의 목적으로 대리 서명키를 사용할 수 없어야 한다. 만약, 대리 서명키 오용의 문제가 발생하면 대리 서명자의 책임이 명시적으로 드러나야 한다.

2.2 대리 서명

[4]에서 처음 소개된 대리 서명 스킴에서는 대리 서명자가 무한하게 서명을 생성할 수 있기 때문에 위임 권한을 취소하기 위해서 별도의 권한 취소 프로토콜을 사용해야 한다. 그래서 [1]에서는 위임 정보에 위임 기간을 포함시킨 대리 서명 방법을 제안하였다. 그러나 대리 서명자의 서명 생성 시간을 알 수 없기 때문에, 위임 기간이 만료된 대리 서명자의 서명 위조 가능성이 있다[1][2].

[3]에서는 대리 서명에 타임-스탬프를 이용하여 이 문제를 해결하였다. 대리 서명자는 시간 정보가 포함된 이

진 연결 스킴(binary linking scheme)의[5] 연결 정보(linking information)를 생성하여 서명에 포함시키고, 연결 정보는 인증된 저장 공간에 공고한다. 그리고 위임 기간이 만료되면, 대리 서명자는 종료(ending) 연결 정보를 생성하여 인증된 저장 공간에 공고한다. 대리 서명자와 수신자의 공모가 의심될 때에는 공고된 연결 정보를 검증한다. 현재의 연결 정보에는 과거 연결 정보의 해쉬 함수 값이 포함되기 때문에 과거의 서명을 위조할 수 없으므로 포워드-시큐어하다. 그러나 서명 생성 및 검증시에 대리 서명자와 수신자가 정확한 시간 정보를 생성할 수 있는 메커니즘이 필요하며, 생성되는 모든 연결 정보를 실시간으로 인증된 저장 공간에 공고해야 한다.

2.3 포워드-시큐어 서명

포워드-시큐어한 가장 간단한 서명 방법은 보안 강도에 따라 일정 시간 구간마다 서명키와 서명 검증키를 변경하는 것이다. 그러나 이것은 매우 비효율적이기 때문에, 고정된 서명 검증키를 사용하면서 매 시간 구간마다 서명키를 갱신한다. 이 때, 현재의 서명키로 과거의 서명키를 유도해낼 수 없도록 하여 과거 서명의 타당성을 보호한다.

Fiat-Shamir 스킴을 이용한 방법은[6] 원 서명 스킴 자체의 계산량이 많기 때문에, 포워드-시큐어 서명 스킴 역시 계산량이 많고, Guillou-Quisquater 서명을 이용한 방법은[7] 서명키 생성과 갱신의 계산량이 시간 구간의 수에 비례하는 대신, 서명 생성과 검증의 계산량을 최적화시켰다. 또한, 마스터 공개키를 이용하여 특정 시간 구간에 사용할 공개키들을 체인(chain)의 형태로 인증함으로써 임의의 서명 스킴을 포워드-시큐어하게 만드는 방법은[8] 시간 구간에 비례하는 키 생성 계산량과 부가적인 저장 공간이 필요하다.

포워드-시큐어 서명 스킴들은 효율성 측면에서 저장 공간이나 계산량 등에서 상충 관계(trade-off)가 있지만, 기본적으로 주요-파라미터(main-parameter)인 공개키, 비밀키, 서명의 크기가 시간 구간의 수에 독립적이어야 한다[6][7][8].

3. 유효 기간을 갖는 포워드-시큐어 대리 서명 방법 제안

원 서명자는 보안 파라미터로서 대리 서명키의 갱신 주기와 위임 기간을 설정한다. 그리고 이를 설명하는 위임 정보를 생성하여 대리 서명자에게 전송하고, 위임 정보와 시간 구간에 관련된 파라미터들을 공고한다.

대리 서명자는 위임 정보의 타당성을 검증하고, 매 갱신 구간마다 새로운 대리 서명키를 생성한다. 서명을 할 때에는 해당 구간의 대리 서명키를 사용하여 효율적인 포워드-시큐어 대리 서명을 생성한다. 수신자는 대리 서명을 검증하는 과정에서 대리 서명자의 정당한 서명임을 검증하고, 원 서명자의 위임 동의와 함께 위임 권한의 유효 기간을 확인한다.

위임 기간이 만료되면, 원 서명자는 공고했던 파라미터들을 삭제한다.

프로토콜에서 사용되는 파라미터는 다음과 같다.

- x_O , $y_O = g^{x_O}$: 원 서명자 O의 개인키와 공개키
- x_P , $y_P = g^{x_P}$: 대리 서명자 P의 개인키와 공개키
- p, q : $q|p-1$을 만족하는 큰 소수
- g : 위수가 q인 Z_p^*상의 서브 그룹 생성자
- $h()$, $H()$: 충돌 회피성 해쉬 함수
- $S()$, $V()$: 이산대수 문제의 어려움에 기반한 서명 생성 알고리즘과 서명 검증 알고리즘

프로토콜은 다음과 같이 5단계로 이루어진다.

① 원 서명자에 의한 위임 정보 생성

1	갱신 주기 u, 위임 기간 $L(=u \cdot T)$을 설정하고, 보증 정보 m_w를 생성한다.
2	해쉬 체인의 초기 입력값 $A \in_R Z_q^*$를 선택한다.
3	위임 권한이 만료되는 시간에 대한 해쉬값 $H^{T+1}(A) = E$를 계산한다.
4	$k \in_R Z_q^*$를 선택하고, $r = g^k \pmod p$과 위임 정보 $s = x_O \cdot h(m_w , r) + k \pmod q$를 계산한다.
5	r , T , E를 공고한다.
6	대리 서명자에게 (m_w, s, A, r, E)를 전송한다.

보증 정보 m_w에는 원 서명자 및 대리 서명자의 신원, 갱신 구간 횟수 T, 갱신 주기 u에 대한 설명, 서명 권한에 대한 세부적인 사항 등이 포함된다.

② 대리 서명자에 의한 위임 정보 검증

1	$g^s = y_O^{\,h(m_w , r)} \cdot r \pmod p$의 등식이 성립하는지 확인함으로써 위임 정보의 타당성을 검증한다.

③ 대리 서명자에 의한 대리 서명키 생성 및 갱신

매 시간 구간 t $(1 \le t \le T)$마다 다음을 수행한다.

1	$H^t(A) = H(H^{t-1}(A))$를 계산하여 시간 구간을 갱신한다.
2	$b_t \in_R Z_q^*$를 선택하고, $B_t = g^{b_t} \pmod p$를 계산한다.
3	b_{t-1}를 삭제하여 포워드-시큐어한 속성을 만족시킨다.
4	시간 구간 t동안 사용할 대리 서명키 x_t와 대리 서명 검증키 y_t를 생성한다. $x_t = s + x_P \cdot h(H^t(A) , B_t , r) + b_t \pmod q$ $y_t = g^{x_t} \pmod p$

④ 대리 서명자에 의한 서명 생성

1	보증 정보 m_w를 통해서 메시지 m이 서명 위임된 사항인지 확인한다.
2	현재 시간 구간 t의 대리 서명키 x_t로 메시지 m에 대한 서명 $\sigma = S(x_t , m)$을 생성한다.
3	수신자에게 대리 서명 $(m, \sigma, m_w, r, E, t, H^t(A), B_t)$를 전송한다.

⑤ 수신자에 의한 서명 검증

1	보증 정보 m_w를 통해서 메시지 m이 서명 위임된 사항인지 확인한다.
2	$H^{T-t+1}(H^t(A)) = E$인지 확인함으로써 위임 권한의 유효 기간을 검증한다.
3	대리 서명 검증키 y_t를 계산한다. $y_t = y_O^{h(m_w, r)} \cdot r \cdot y_P^{h(H^t(A), B_t, r)} \cdot B_t \pmod p$
4	현재 시간 구간 t의 대리 서명 검증키 y_t로 $V(y_t, m, \sigma) = true$인지 검증한다.

위임 권한이 만료되면 원 서명자는 공고했던 파라미터들을 삭제하고, 대리 서명자는 위임 기간동안 생성한 서명의 해쉬값들을 원 서명자에게 전송한다.

4. 안전성 및 효율성

- 강력한 위조 불가능성 : $S()$, $V()$가 안전하다고 가정하였을 때, 서명을 위조하려면 대리 서명키를 위조해야 한다. 이것은 대리 서명 검증키 y_t로부터 $y_t = g^{x_t}$를 만족하는 x_t를 알아내는 문제와 같으므로 계산상 불가능하다. 또한, 대리 서명키 x_t는 보증 정보 m_w를 포함하는 위임 정보 s와 대리 서명자의 비밀키 x_P를 알아야만 계산할 수 있다. 따라서, 원 서명자를 포함한 어떤 제3자도 타당한 대리 서명을 생성할 수 없다.
- 검증성 : 수신자가 서명을 검증할 때, 원 서명자의 공개키 y_O를 통하여 원 서명자의 위임 동의가 검증된다.
- 강력한 신원 확인성 : 수신자가 서명을 검증할 때, 대리 서명자의 공개키 y_P를 통하여 대리 서명자의 신원이 확인된다.
- 강력한 부인 불가능성 : 강력한 위조 불가능성에 의해서 대리 서명자는 자신의 서명을 부인할 수 없다.
- 오용 방지 : 보증 정보 m_w에 명시되지 않은 목적으로 대리 서명키가 사용되었다면, 강력한 위조 불가능성에 의해서 이것은 대리 서명자 P의 책임이다.
- 위임 기간 : 위임 기간이 만료되면, 원 서명자는 시간 구간에 관련된 파라미터들을 삭제한다. 따라서, 수신자는 원 서명자가 공고한 파라미터로부터 위임 권한의 유효 기간을 검증할 수 있다.
- 공모 불가능성 : 위임 기간이 만료되면, 대리 서명자는 위임 기간동안 생성한 서명의 해쉬값들을 원 서명자에게 전송하기 때문에 위임 기간이 만료된 대리 서명자와 수신자의 공모는 불가능하다.
- 포워드-시큐어 : 시간 구간 t의 서명키 x_t가 노출되더라도 공격자는 x_P, b_t 값을 알아낼 수 없고, 이들이 노출되더라도 b_j $(1 \le j < t)$는 b_t와 연관성이 없는 난수이기 때문에 이로부터 b_j를 알아낼 수 없다. 또한, 과거의 B_j $(1 \le j < t)$들이 공개되어있다 하더라도 이산대수 문제의 어려움에 근거하여 B_j로부터 b_j $(1 \le j < t)$를 알아내는 것은 계산상 불가능하다. 따라서 현재의 서명키로부터 과거의 서명키를 유도해 낼 수 없다.
- 효율성 : 대리 서명키, 대리 서명 검증키, 대리 서명의 크기가 시간 구간의 수에 독립적일 뿐만 아니라, 수신자의 서명 검증시 해쉬 함수 계산을 제외하고는 대리 서명키 생성, 갱신, 대리 서명 생성의 계산량이 시간 구간의 수에 독립적이다. 또한, 포워드-시큐어한 속성을 제공하기 위한 부가적인 저장 공간을 필요로 하지 않는다.

5. 결론

대리 서명은 대리 서명자가 원 서명자의 권한을 위임받아 대신 행사하는 것이기 때문에, 대리 서명자의 권한 남용 문제는 중요하게 다뤄져야 한다.

일반적으로 원 서명자가 위임한 권한은 유효 기간을 갖는다. 위임 기간을 위임 정보에 명시적으로 표시한다고 하더라도, 대리 서명자의 서명 생성 시간을 알 수 없기 때문에 완전하지 못하며 서명키 노출 문제도 해결하지 못한다.

본 논문에서는 기존 대리 서명의 보안 요구사항을 만족하면서도 위임 기간이 만료된 대리 서명자의 서명 위조를 막고, 서명키 노출 문제를 해결하기 위한 포워드-시큐어 서명을 생성하는 방법을 제안하였다. 제안한 방법은 원 서명자가 시간 관련 파라미터를 공고하여 제어하기 때문에 대리 서명자와 수신자가 정확한 시간 정보를 필요로 하지 않고, 서명키 갱신 주기를 보안 파라미터로 사용하여 시간 구간별로 포워드-시큐어한 속성을 만족시킨다.

6. 참고문헌

[1] S. Kim, S. Park, and D. Won, "Proxy Signatures, revisited," Proc. of ICICS 97, pp.223-232, 1997.

[2] B. Lee, H. Kim, and K. Kim, "Strong Proxy Signature and its Applications," Proc. of SCIS 2001, pp.603-608, 2001.

[3] H. M. Sun, "Design of time-stamped proxy signature with traceable receivers," Proc. of IEE Computers and Digital Techniques, Vol. 147, No. 6, pp.462-466, 2000.

[4] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: Delegation of the power to sign message," IEICE Trans. Fundamentals, Vol. E79-A, No. 9, pp.1338-1353, 1996.

[5] A. Buldas, P. Laud, H. Lipmaa, and J. Villemson, "Time-stamping with binary linking schemes," Proc. of Crypto'98, pp.486-501, 1998.

[6] M. Bellare and S. Miner, "A forward-Secure digital signature scheme," Crypto'99, 1999.

[7] G. Itkis and L. Reyzin, "Forward-Secure Signatures with Optimal Signing and Verifying," Crypto'01, 2001.

[8] H. Krawczyk, "Simple forward-secure signatures from any signature scheme," 7th ACM Conference on Computer and Communication Security, 2000.