XTR 기반의 대리 서명

이재욱*, 전동호*, 최영근*, 김순자*

*경북대학교 전자공학과

Proxy Signature based on XTR

Jae-wook Lee*, Dong-ho Jeon*, Young-geun Choe*, Soon-ja Kim*

*Dept. of Electronics Engineering, Kyungpook National University

요 약

Lenstra와 Verheul에 의해 주장된 XTR은 짧은 키 길이와 빠른 연산 속도의 장점을 가지고 있기 때문에 복잡한 연산에 유용하게 사용될 수 있다. 또한 본 서명자의 부재시 위임 서명자가 대신 서명할 수 있는 대리 서명은 본 서명자가 대리 서명시 사용 될 비밀 정보를 생성하여 대리 서명자에게 전송한다. 대리 서명자는 전송된 비밀 정보의 유용성을 판별한 후 유용하다면 이 정보를 사용하여 서명하게 된다. 이러한 대리 서명 과정에서 발생하는 연산들은 XTR을 이용하여 속도의 향상을 가져올 수 있고, 짧은 키 길이로 동일한 안정성을 가진다. 따라서 유선에서 뿐만 아니라 무선에서도 효율성을 가질 수 있다.

Ⅰ. 서론

최근 유무선 인터넷의 발전에 따라 전자 서명의 중요성은 점차 커지고 있다. 전자 서명은 서명 권한이 있는 서명자에 의해서만 이루어질 수 있고 그 서명의 유용성을 검증할 수 있어야 한다.

현대 사회에서는 업무의 특성상 서명의 권한을 가진 사람이 부득이한 사정으로 인해 서명할 수 없는 상황이 생긴다. 이 경우 서명의 권한을 가진 서명자가 인정하는 대리 서명자를 통해 서명을 할 수 있는 기법이 필요하다. 이 서명 기법이 대리 서명(proxy signature)이다. 대리 서명에서는 본 서명자(original signer)가 대리 서명자(proxy signer)에게 대리 서명시 사용할 비밀 정보를 넘겨주고, 대리 서명자는 이 비밀 정보에 대한 유용성을 판별하게 된다. 이 과정에서 생기는 연산의 속도를 향상시키고 또한 키 길이를 줄일 수 있다면 유선뿐만 아니라 무선에서도 효율적이다. 본 논문에서는 XTR에 기반을 둔 대리 서명 프로토콜을 제시하였다.

본 논문의 2장에서는 XTR 공개키 시스템에 대하여 알아본다. 3장에서 대리 서명 기법 중 부분 위임을 이용한 proxy protect를 만족하는 Mambo의 대리 서명 프로토콜과 증명서 기반의 부분 위임을 사용한 Kim, Park, Won의 프로토콜을 살펴본다. 4장에서는 3장에서 제시한 대리 서명 기법을 XTR 공개키 시스템에 맞는 프로토콜로 변환하기 위한 새로운 알고리즘과 이 알고리즘을 이용하여 XTR에 기반을 둔 대리 서명 프로토콜을 제시하였다. 5장에서는 제시한 프로토콜에 대한 안정성과 효율성을 살펴보고, 끝으로 6장에서 결론을 맺는다.

Ⅱ. XTR공개키 시스템 분석

$p(\equiv 2 \bmod 3)$을 만족하는 소수 p와 p^2-p+1을 나누는 위수 $q(>6)$의 원소를 g라고 한다. p^2-p+1은 $GF(p^6)$의 위수 p^6-1을 나눌 수 있기 때문에 g는 $GF(p^6)^*$의 위수 q를 가지는 부분군을 생성한다. 또한 q는 p^s-1 ($s=1,2,3$)를 나눌 수 없으므로 g에 의해 생성된 부분군은 $GF(p^6)$의 부분체의 곱셈군에 포함되지 않는다. 또한 g의 임의의 멱승은 부분체 $GF(p^2)$의 한 원소를 사용하여 표현될 수 있다. 이러한 멱승은 $GF(p^6)$에서의 연산을 피하고 $GF(p^2)$의 원소로 연산하여 효율적으로 계산된다. 따라서 $GF(p^2)$의 원소들로 연산하여 $GF(p^6)$의 안정성을 보장한다[1].

1. Traces $Tr(g)$와 다항식 $F(c,X)$

$GF(p^6)$의 원소 g에 대하여 $GF(p^2)$상에서의 conju- gate는 g, g^{p^2}, g^{p^4}이고, 이 값들의 합을 Traces라 한다. 즉 Traces $Tr(g)=g+g^{p^2}+g^{p^4}$이다. 여기서 g의 위수가 p^2-p+1을 나눌 수 있기 때문에 $p^2=p-1$, $p^3=-1$이다. 따라서 실제 conjugate는 g, g^{p-1}, g^{-p}으로 나타난다.

또한 다항식 $F(c,X)=X^3-cX^2+c^pX-1$에 대한 세 근을 h_0, h_1, h_2라 하면 $c_n=h_0^n+h_1^n+h_2^n$ 이다. 즉 다항식 $F(c,X)$의 각 근의 n 제곱의 합이

c_n이다. 여기서 나타나는 세 근은 g, g^{p-1}, g^{-p}이므로 이 성질을 이용하면 c_n에 대한 여러 가지 연산을 찾을 수 있다. 이러한 연산의 특성을 통해 $S_n(c)=(c_{n-1}, c_n, c_{n+1})$을 구하는 알고리즘을 제시하였다[1].

2. Traces의 연산

[1]과 [2]에서는 $Tr(g)$, $S_k(Tr(g))$, a, b가 주어진 경우 이 값을 통해 $Tr(g^a g^{bk})$를 연산하는 알고리즘을 제시하였다. 매트릭스를 사용하여 $Tr(g^a g^{bk})$를 구하는 방법은 $GF(p)$상에서 $8\log_2(a/b \bmod q)+8\log_2(b)+34$번의 곱셈 연산이, 매트릭스를 사용하지 않는 방법은 $16\log_2 q$번의 곱셈 연산이 필요하다.

Ⅲ. 대리 서명

대리 서명은 본 서명자(original signer)의 부재 등의 사유로 본 서명자가 직접 서명할 수 없을 경우 대리 서명자(proxy signer)가 대신하여 서명할 수 있는 서명 기법이다.

대리 서명 기법에는 본 서명자의 비밀키를 대리 서명자에게 넘겨주는 기법(full delegation)과, 본 서명자의 비밀키로부터 대리 서명자가 사용할 비밀 서명 정보를 별도로 만들어서 위임 서명자에게 넘겨주는 부분 위임(partial delegation), 본 서명자가 위임 서명자로 지정한 사실을 증명서로 만들어서 넘겨주는 증명서 위임(delegation of warrant)이 있다. 이러한 기법 중 부분 위임은 대리 서명시 사용할 비밀 정보를 본 서명자 또한 알고 있는 방법(proxy *unprotect*)과 위임 서명자만이 알고 있는 방법(proxy protect)이 있다[3].

이 중 부분 위임을 이용한 proxy protect를 만족하는 Mambo의 대리 서명 프로토콜과 Kim, Park, Won이 주장한 증명서 기반의 부분 위임(partial delegation with warrant) 대리 서명 프로토콜에 대해서 살펴본다.

1. Proxy protect를 만족하는 부분 위임 대리 서명

본 서명자(A)의 비밀키와 대리 서명자(B)의 임시 비밀키가 대리 서명에 사용될 비밀 정보(σ) 생성시 사용된다. B는 A의 비밀키를 알 수 없으므로 부분 위임 방식이며, 또한 A는 σ를 알 수 없으므로 proxy protect를 만족한다.

1) 파라미터

$2^{511}<p<2^{512}$인 소수 p와 곱셈군 Z_p^*의 생성자 g를 선택한다. $s_A \in_R Z_{p-1}$인 A의 비밀키 난수 s_A를 생성하고, 그에 상응하는 공개키 $v_A=g^{s_A}$를 계산한다.

2) 프로토콜

B는 자신이 사용할 임시 비밀키로 사용할 $x \in_R Z_{p-1}$인 난수 x를 선택한 후 $r=g^x K \bmod q$를 계산하여 $r \in Z_q^*$임을 확인하여 만족하면 r을 A에게 보낸다. A는 $\bar{\sigma}=rs_A+k \bmod q$를 계산하여 B에게 보낸다. 여기서 q는 $p-1$을 나누는 임의의 수이다. B는 A에게서 받은 $\bar{\sigma}$와 자신의 임시 비밀키 x를 이용하여 $\sigma=\bar{\sigma}+x \bmod q$를 계산한 후 이 값의 유용성을 $g^{\sigma}=v_A^r r \bmod q$로 확인한다. 대리 서명자의 서명과 검증은 [4]와 동일하다[5].

본 서명자 A	공개 정보 (p, q, g, v_A)	대리 서명자 B
$k \in_R Z_{p-1}$ $K = g^k \bmod p$		
	K →	
		$x \in_R Z_{p-1}$ $r = g^x K \bmod p$
	← r	
$\bar{\sigma}=rs_A+k \bmod q$		
	$\bar{\sigma}$ →	
		$\sigma=\bar{\sigma}+x \bmod q$ $g^{\sigma} \stackrel{?}{=} v_A^r r \bmod q$

그림1: Proxy protect를 만족하는 부분 위임 대리 서명.

2. 증명서 기반의 부분 위임 대리 서명

서명 생성시 본 서명자가 대리 서명자를 인정하는 증명서(warrant) m_w를 사용하였다.

1) 프로토콜

proxy 생성시 $e=h(m_w, K)$를 계산하고 B에게 넘겨줄 비밀 정보 $\sigma=es_A+k \bmod p-1$를 계산한다. A는 B에게 m_w, σ, K를 넘겨준다. 또한 proxy verification시 B는 A에게 받은 m_w와 K를 이용하여 $e=h(m_w, K)$를 생성한 후 $g^{\sigma} \stackrel{?}{=} v_A^e K \bmod p$를 확인하여 σ의 유용성을 판별한다. 대리 서명자는 σ를 사용하여 서명을 생성한다. 생성된 서명은 $(m_p, Sign_{\sigma}(m_p), K, m_w)$이다. 즉 Mambo의 프로토콜에 m_w가 추가된다[3].

Ⅳ. XTR 기반의 대리 서명

3장에서 살펴본 두 개의 대리 서명 프로토콜을 연산 속도 향상과 키 길이 단축을 위해 XTR에 맞는 프로토콜을 제시한다.

1. $Tr(g^a g^{bk})$를 구하는 알고리즘 제안

1) 알고리즘의 필요성

$Tr(g^a g^{bk})$를 구하는 알고리즘인 [1, Algorithm

2.4.8]과 [2, 2.3]는 표1 에서와 같이 $Tr(g)$, $S_k(Tr(g))$, a, b의 값을 알고 있어야만 가능하다. 그러나 이 알고리즘에서는 a, b를 모두 알고 있어야 한다. 이 값을 안다는 것은 상대방의 비밀키 또는 임시 비밀키를 알고 있어야 하므로, 이 알고리즘을 대리 서명 프로토콜에 직접 사용하는 것은 불가능하다. 따라서 대리 서명에 사용될 수 있는 새로운 알고리즘이 필요하다. 이러한 알고리즘이 알고리즘 4.1.1이다. 알고리즘 4.1.1은 $0<b<q$, $Tr(g)$, $S_k(Tr(g))$, $S_a(Tr(g))$가 주어질 경우 $Tr(g^a g^{bk})$을 구할 수 있다.

구 분	$Tr(g^a g^{bk})$ 연산시 필요한 파라미터
매트릭스를 사용하는 알고리즘[1]	$Tr(g)$, $S_k(Tr(g))$, a, b
매트릭스를 사용하지 않는 알고리즘[2]	$Tr(g)$, $S_k(Tr(g))$, a, b
제안하는 알고리즘 4.1.1	$Tr(g)$, $S_k(Tr(g))$, $S_a(Tr(g))$, b

표1: 각 알고리즘에 필요한 파라미터 비교.

2) 알고리즘 제안

알고리즘 4.1.1 ($Tr(g^a g^{bk})$연산)

i . $Tr(g) = Tr(g^k)$라 하면,

$S_1(Tr(g)) = (3, Tr(g), Tr(g^2) - 2Tr(g)^k)$이다.

ii . $S_b(Tr(g)) = (Tr(g^{b-1}), Tr(g^b), Tr(g^{b+1}))$

iii . $Tr(g^b) = Tr(g^{bk})$

iv . $S_{bk}(Tr(g)) = (Tr(g^{bk-1}), Tr(g^{bk}), Tr(g^{bk+1}))$

v . $C(A(Tr(g)^a))$

vi . $Tr(g^{a+bk}) = S_{bk}(Tr(g)) * C(A(Tr(g)^a))$

i 에서 $Tr(g)$는 $Tr(g^k)$가 c_1이고 c_0는 3이다. 또한 c_2는 [1, Lemma 2.3.5 i]에 의해 $Tr(g^2) - 2Tr(g)^k$이므로 $S_1(Tr(g)) = (3, Tr(g), Tr(g^2) - 2Tr(g)^k)$이다. b와 $Tr(g)$는 알려진 값이므로 [1, Algorithm 2.3.7]에 의해 ii 를 구한다. iii에서는 $S_b(Tr(g)) = S_b(Tr(g^k))$이고 $S_b(Tr(g)) = (Tr(g^{b-1}), Tr(g^b), Tr(g^{b+1}))$이이므로 [2, 2.1 Facts.2(c)]에 의해 $Tr(g^b) = Tr(g^{bk})$를 구한다[1, Algorithm 2.4.8].

iv . $S_{bk}(Tr(g)) = (Tr(g^{bk-1}), Tr(g^{bk}), Tr(g^{bk+1}))$을 구하기 위해 $Tr(g^{bk-1})$과 $Tr(g^{bk+1})$을 계산해야 한다. [1, Algorithm 2.4.8]에 $a=1$을 대입하여 $Tr(g^{bk+1})$을 구한다. $Tr(g^{bk-1})$을 구하기 위해서 $Tr(g^{-1}) = Tr(g^p)$의 값을 얻게 되고[1, Lemma2.3.2 v], [1, Algorithm 2.4.8]에 $a=p$를 대입하여 $Tr(g^{bk-1})$의 값을 구한다.

i 에서 iv의 과정을 통해 얻은 $Tr(g)$와 $S_a(Tr(g))$으로 v 에서 $C(A(Tr(g)^a))$을 구한다. 이 과정은 a값을 알지 못해도 가능하다. vi에서 $Tr(g^{a+bk}) = S_{bk}(Tr(g)) * C(A(Tr(g)^a))$을 연산함으로써 $Tr(g^{a+bk})$를 구한다[1, Algorithm 2.4.8].

2. XTR을 기반으로 하고 proxy protect를 만족하는 부분 위임 대리 서명

1) 파라미터

$p \equiv 2 \bmod 3$를 만족하는 $2^{169} < p < 2^{170}$인 소수 p와 $p^2 - p + 1$을 나누는 위수 $q(>6)$의 원소 g를 선택하여 $Tr(g)$를 계산한 후 공개한다. $s_A \in_R Z_{q-2}$인 A의 비밀키 난수 s_A를 생성하고, 그에 상응하는 공개키 $v_A = g^{s_A}$를 계산한다.

2) 프로토콜

서명자 A	공개정보 $\left(\begin{matrix} p, q, v_A \\ Tr(g) \end{matrix}\right)$	대리 서명자 B
$1<k<q-2$ $S_k(Tr(g))$		
	$S_k(Tr(g))$ →	
		$1<x<q-2$ 선택 $r = h(Tr(g)^k) * h(Tr(g)^x)$
	← r	
$\bar{\sigma} =$ $rs_A + k \bmod q$		
	$\bar{\sigma}$ →	
		$\sigma = \bar{\sigma} + x \bmod q$ $S_{s_A}(Tr(g))$, r, $S_k(Tr(g))$, $Tr(g)$로 $Tr(g^{rs_A+k+x})$ 계산 $Tr(g^\sigma)$와 같은지 검증

그림 2: XTR을 기반으로 하고 proxy protect를 만족하는 부분 위임 대리 서명.

본 서명자 A는 임시 비밀키로 사용할 k $(0<x<q-2)$를 선택한 후 $S_k(Tr(g))$를 계산하고 대리 서명자 B에게 보낸다. 대리 서명자 B는 자신의 임시 비밀키로 사용될 $x(0<x<q-2)$를 선택한 후 $S_x(Tr(g))$를 계산하고 $Tr(g^x)$을 해쉬한 후, 받은 $S_k(Tr(g))$중 $Tr(g^k)$를 해쉬한 값을 곱하여 $r = h(Tr(g)^k) * h(Tr(g)^x) \bmod p$을 만들어 A에게 보낸다. A는 이 값과 자신의 비밀키 (s_A)와 k를 사

용하여 $\bar{\sigma}=rs_A+k \bmod q$를 B에게 보낸다. B는 받은 $\bar{\sigma}$와 x를 이용하여 $\sigma=\bar{\sigma}+x \bmod q$를 생성한다. σ의 유용성을 판별하기 위해 먼저 $Tr(g^{\sigma})$를 계산한다. 또한 A의 공개키 $S_{s_A}(Tr(g))$, 자신이 생성한 r, A에게서 받은 $S_k(Tr(g))$, 공개정보 $Tr(g)$를 이용하여 알고리즘 4.1.1에 의해 $Tr(g^{rs_A+k+x})$를 계산한다. 이 값과 $Tr(g^{\sigma})$로 얻은 값과 비교하여 σ의 유용성을 판별한다. B는 문서 m_p에 σ를 이용하여 대리 서명을 생성한다. 생성된 서명값은 $(m_p, Sign_{\sigma}(m_p), S_k(Tr(g))$이다. 대리 서명의 검증을 위해 검증자는 공개키 v_A와 비밀리에 받은 $S_k(Tr(g))$를 이용하여 proxy 검증과 동일한 방법으로 v'를 생성하여 검증한다. 이 과정은 그림 2와 같다.

3. XTR을 기반으로 하고 증명서를 기반으로 하는 부분 위임 대리 서명

본 서명자 A는 k를 선택한 후 $S_k(Tr(g))$를 계산한다. m_w와 $K=Tr(g^k)$을 해쉬하여 $e=h(m_w, K)$를 만든후 비밀 정보 σ를 생성한다. $\sigma=es_A+k \bmod p-1$를 계산한 후 m_w, σ, K를 대리 서명자 B에게 보낸다. B는 A에게서 받은 m_w와 $Tr(g^k)$을 해쉬하여 $e=h(m_w, K)$를 얻는다. 비밀 정보 σ의 유용성을 확인하기 위해 $Tr(g^{\sigma})=Tr(g^{es_A+k})$를 계산한다. 또한 $S_k(Tr(g))$, e, $Tr(g)$, $S_{s_A}(Tr(g))$를 사용하여 알고리즘 4.1.1에 의해 $Tr(g^{es_A+k})$를 계산한다. 이 값과 $Tr(g^{\sigma})$를 비교하여 유용성을 판별한다.

V. 안전성 및 효율성 분석

XTR의 안전성은 $S_n(Tr(g))$의 값으로부터 n을 찾을 수 없어야 한다는 것에 기반을 둔다. 이것은 이산 대수 문제에 기반을 두는 것과 동일한 안전성을 가진다[1]. 3장에서 제시된 대리 서명 프로토콜은 이산 대수 문제를 기반으로 한다. 이러한 이산 대수 문제를 기반으로 하는 프로토콜에 대한 XTR의 적용은 프로토콜 자체의 안정성을 그대로 유지하게 한다.

또한 XTR은 170 비트의 키길이로 1024 비트의 키길이를 갖는 이산대수 문제에 기반한 암호 시스템과 동일한 안전성을 갖는다. 본 논문에서는 대리 서명 기법에 XTR 암호 시스템을 적용함으로써 프로토콜 진행시 키길이의 감소와 연산량의 효율성을 높일 수 있다.

VI. 결론

XTR의 장점인 짧은 키 길이와 빠른 연산 속도를 이용하기 위해 대리 서명에 맞는 XTR 알고리즘을 제시하였고, 이 알고리즘을 이용한 대리 서명 프로토콜을 제시하여 키 길이의 단축을 가져왔다. 연산 속도 향상과 키 길이 단축이라는 장점은 유선뿐만 아니라 무선인터넷에서도 효율적으로 이용될 수 있다.

참고 문헌

[1] Arjen K. Lenstra and Eric R. Verheul, "The XTR public key system," Proceedings of Crypto 2000, LNCS 1880, Springer-Verlag, pp.1-19, 2000.

[2] Martijn Stam and Arjen K. Lenstra, "Speeding up XTR," Proceedings of Asiacrypt 2001, pp.125-143, 2001.

[3] S. Kim, S. Park and D.Won, "Proxy signatures, re visited," Proceedings of International Conference on Information and Communications Security (ICISC'97), pp.223-232, 1997.

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

[5] N. Lee, T. Hwang, and C. Wang, "On Zhang's Nonrepudiable Proxy Signature Scheme," in Proceedings of ACISP '98-Australasian Conference on Information Security and Privacy (C. Boyd and E. Dawson, eds.), vol. 1438 of Lecture Notes in Computer Science, pp.415-422, 1998.

[6] J. Herranz and G. Saez, "Fully Distributed Proxy Signature Scheme," 2002.

[7] C.P. Schnorr, "Efficient signature generation by smart cards," Journal of Cryptology Vol 4, pp.161-174, 1991.

[8] Eric R. Verheul, "Evidence that XTR is more secure than supersingular elliptic curve cryptosystems," Advances in Cryptology-Eurocrypt 2001, pp.195-210, 2001.

[9] A.E. Brouwer, R. Pellikaan and E.R. Verheul, "Doing more with fewer bits," Proceedings Asiacrypt99, LNCS 1716, Springer-Verlag 1999, pp.321-332, 1999.