情報保護學會論文誌
第13券 第3號, 2003. 6

XTR 암호 시스템 기반의 대리 서명*

이 재 욱**, 전 동 호**, 최 영 근**, 김 순 자***

Proxy Signatures based on XTR Cryptosystem

Jae-Wook Lee*, Dong-Ho Jeon*, Young-Geun Choe*, Soon-Ja Kim**

요 약

Lenstra와 Verheul에 의해 제시된 XTR은 짧은 키 길이와 빠른 연산 속도의 장점을 가지고 있기 때문에 복잡한 연산에 유용하게 사용될 수 있다. 본 논문에서는 XTR에서 $Tr(g^a g^{bk})$를 연산하는 새로운 알고리즘과 이 알고리즘을 이용한 XTR 기반의 대리 서명 프로토콜을 제시하였다. $Tr(g^a g^{bk})$를 연산하는 기존의 알고리즘은 두 개의 비밀 정보가 공개되어야만 한다. 따라서 대리 서명 프로토콜의 생성 및 검증에 이용할 수 없다. 제안하는 새로운 알고리즘은 대리 서명자의 비밀키와 공개 정보로 proxy의 생성과 검증이 가능하므로 대리 서명 프로토콜을 XTR에 적용 가능하게 한다. 따라서 XTR 기반의 대리 서명 프로토콜은 XTR의 기본적인 장점을 가진다. 이러한 장점은 유선 뿐 아니라 무선에서도 이용될 수 있다.

ABSTRACT

The XTR public key system has advantage of short key length and fast computing speed. So, the XTR is used usefully in complicated operation. In this paper, we propose a new algorithm of double exponentiation operation and a proxy signature protocol based on the XTR. The double exponentiation operation should be executed to apply XTR for the proxy signature protocol. But this algorithm is inappropriate, because two secret key has to be known in existent operation algorithm. New algorithm enable double exponentiation operation with proxy signer's secret key and public information. And the XTR is used to generation and verification of proxy at proxy signature protocol. Therefore proxy signature based on the XTR has basic advantage of the XTR. These advantage can be used in internet as well as mobile.

Keyword : XTR, proxy signature

Ⅰ. 서 론

유무선 인터넷의 발달에 따라 전자 서명의 중요성은 점차 커지고 있다. 전자 서명은 서명 권한이 있는 서명자에 의해서만 이루어지고, 제 3자에 의해 그 서명의 유용성을 검증할 수 있어야 한다. 이러한 전자 서명에는 여러 가지 상황에 맞는 다양한 서명 기법이 존재한다. 특히 서명의 권한을 가진 사람(original signer)이 부득이한 사정으로 인해 서명할 수 없는 상황에 서명자가 인정하는 대리 서명자(proxy signer)를 두어 본 서명자를 대신하여 서명할 수 있다면 유용하게 사용될 수 있다. 이러한 서명 기법이 대리 서

* 본 논문은 2002년도 CISC 우수 논문임.
** 경북대학교 전자·전기 공학부 컴퓨터 통신망 연구실({llkkllkk, jdh, ind}@palgong.knu.ac.kr)
*** 경북대학교 전자·전기 공학부 교수(snjkim@ee.knu.ac.kr)

명(proxy signature)이다. 대리 서명에서는 본 서명자가 대리 서명자에게 대리 서명시 사용할 비밀 정보를 넘겨주고, 대리 서명자는 이 비밀 정보에 대한 유용성을 판별하게 된다[3]. 이 과정에서 생기는 연산의 속도를 향상시키고 또한 키 길이를 줄일 수 있다면 유선뿐만 아니라 무선에서도 효율적이다.

본 논문에서는 XTR을 기반으로 하는 대리 서명 프로토콜을 제안하여 비밀 정보 생성 및 검증에 필요한 연산의 속도 향상과 키 길이를 축소 시켰다. 대리 서명 프로토콜을 XTR에 적용하기 위해서는 proxy의 생성 및 검증 과정에서 $Tr(g^a g^{bk})$를 구하는 연산이 필요하다. 기존의 연산 알고리즘은 a와 b를 모두 알고 있어야 하므로 대리 서명 프로토콜 적용 시 비밀키를 공개해야 하는 문제점이 발생한다. 이 문제점을 보완하기 위해 새로운 연산 알고리즘을 만들어 XTR을 기반으로 하는 대리 서명 프로토콜에 적용하였다.

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 Kim등이 제안한 프로토콜을 살펴본다[3,4]. 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. XTR 공개키 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].

2.1. Trace $Tr(g)$와 다항식 $F(c,X)$

$GF(p^6)$의 원소 g에 대하여 $GF(p^2)$상에서의 conjugate는 g, g^{p^2}, g^{p^4}이고, 이 값들의 합을 Trace라 한다. 즉 Trace $Tr(g)=g+g^{p^2}+g^{p^4}$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 n제곱의 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].

2.2.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. [1]에서는 행렬을 사용하여 $Tr(g^a g^{bk})$를 구하는 알고리즘을 제시하였고 이는 $GF(p)$에서 $8\log_2(a/b \bmod q)+8\log_2(b)+34$ 번의 곱셈 연산이 필요하다. 또한 [2]에서는 행렬을 사용하지 않는 방법을 제시하였고 연산시 $16\log_2 q$번의 곱셈 연산이 필요하다.

Ⅲ. 대리 서명

대리 서명은 본 서명자의 부재 등의 사유로 본 서명자가 직접 서명할 수 없을 경우 대리 서명자가 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 만족하

는 Mambo의 대리 서명 프로토콜과 Kim등이 주장한 증명서 기반의 부분 위임 (partial delegation with warrant) 대리 서명 프로토콜에 대해서 살펴본다[3,4].

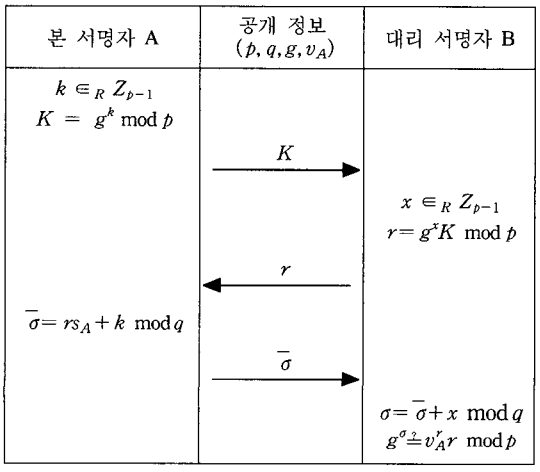
3.1. Proxy protect를 만족하는 부분 위임 대리 서명

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.

본 서명자와 대리 서명자간의 비밀 정보의 생성과 검증 프로토콜은 다음과 같다.

A는 $2^{511}<p<2^{512}$인 소수 p와 곱셈군 Z_p^*의 생성자 g를 선택한 후 공표한다. 또한 $s_A \in_R Z_{p-1}$인 A의 비밀키 난수 s_A를 생성하고, 그에 상응하는 공개키 $v_A = g^{s_A} \bmod p$를 계산한다. B는 자신의 임시 비밀키로 사용할 $x \in_R Z_{p-1}$인 난수 x를 선택한 후 $r = g^x K \bmod p$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 큰 소수이다.

B는 A에게서 받은 $\bar{\sigma}$와 자신의 임시 비밀키 x를 이용하여 $\sigma = \bar{\sigma} + x \bmod q$를 계산한 후 이 값의 유용성을 $g^{\sigma} = v_A^r r \bmod p$로 확인한다. 대리 서명자의 서명과 검증은 [4]와 동일하다[5].



(그림 1) Proxy protect를 만족하는 부분 위임 대리 서명

3.2. 증명서 위임 대리 서명

서명 생성시 본 서명자가 대리 서명자를 인정하는 증명서(warrant) m_w를 사용하였고 프로토콜은 다음과 같다.

proxy생성시 $e = h(m_w, K)$를 계산하고 B에게 넘겨줄 비밀 정보 $\sigma = es_A + k \bmod p-1$를 계산한다. A는 B에게 m_w, σ, K를 넘겨준다. 또한 proxy verification 시 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.

4.1. $Tr(g^a g^{bk})$를 구하는 알고리즘 제안

$Tr(g^a g^{bk})$를 구하는 알고리즘인 [1, Algorithm 2. 4. 8]과 [2, 2.3]는 표1에서와 같이 a, b, $Tr(g)$, $S_k(Tr(g))$의 값을 모두 알고 있어야만 가능하다. a, b 값을 모두 안다는 것은 상대방의 비밀키 또는 임시 비밀키를 알고 있어야 하므로, 기존의 알고리즘을 대리 서명 프로토콜에 직접 사용하는 것은 불가능하다. 따라서 대리 서명에 사용될 수 있는 새로운 알고리즘이 필요하다. 제안하는 알고리즘은 $0<b<q$, $Tr(g)$, $S_k(Tr(g))$, $S_a(Tr(g))$가 주어질 경우 $Tr(g^a g^{bk})$을 구할 수 있다.

표 1. 각 알고리즘에 필요한 파라미터 비교.

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

제안하는 $Tr(g^a g^{bk})$연산 알고리즘

i. $\widetilde{Tr}(g) = Tr(g^k)$라 하면,
$\widetilde{S}_1(Tr(g)) = (3, \widetilde{Tr}(g), \widetilde{Tr}(g^2) - 2\widetilde{Tr}(g)^p)$
ii. $\widetilde{S}_b(Tr(g)) =$
$(\widetilde{Tr}(g^{b-1}), \widetilde{Tr}(g^b), \widetilde{Tr}(g^{b+1}))$
iii. $\widetilde{Tr}(g^b) = Tr(g^{bk})$
iv. $S_{bk}(Tr(g)) =$
$(Tr(g^{bk-1}), Tr(g^{bk}), Tr(g^{bk+1}))$
v. $C(A(Tr(g)^a))$
vi. $Tr(g^{a+bk}) = S_{bk}(Tr(g))^* C(A(Tr(g)^a))$

알고리즘에서 사용된 notation은 다음과 같다[1].

- $C(A(Tr(g))^e) = M_0(Tr(g))^{-1} * (S_e(Tr(g)))^T$
- $M_n(c) = \begin{pmatrix} c_{n-2} & c_{n-1} & c_n \\ c_{n-1} & c_n & c_{n+1} \\ c_n & c_{n+1} & c_{n+2} \end{pmatrix}$
- $M_0(c)^{-1}$ [1)]
- $D = c^{2p+2} + 18c^{p+1} - 4(c^{3p} + c^3) - 27$

i 에서 $\widetilde{Tr}(g)$는 $Tr(g^k)$가 c_1이다. 또한 c_2는 $\widetilde{Tr}(g^2) - 2\widetilde{Tr}(g)^p$ 이므로 [1, Lemma 2. 3.5 i]에 의해 $\widetilde{S}_1(Tr(g)) = (3, \widetilde{Tr}(g), \widetilde{Tr}(g^2) - 2\widetilde{Tr}(g)^p)$이다. b와 $Tr(g)$는 알려진 값이므로 [1, Algorithm 2. 3. 7]에 의해 ii 를 구한다.

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 [1, Algorithm 2.4.8]에 $a=2$를 대입하여 $Tr(g^{bk+2})$의 값을 구한다. $Tr(g^{bk})$, $Tr(g^{bk+1})$, $Tr(g^{bk+2})$으로 $Tr(g^{bk+2}) - Tr(g)^* Tr(g^{bk+1}) + Tr(g)^p * Tr(g^{bk})$ [1, Corollary2.3.5 ii]를 통해 $Tr(g^{bk-1})$을 찾을 수 있다. i 에서 iv의 과정을 통해 얻은 $Tr(g)$와 $S_a(Tr(g))$로 v에서 $C(A(Tr(g)^a))$을 구한다. vi에서 $Tr(g^{a+bk}) = S_{bk}(Tr(g))^* C(A(Tr(g)^a))$을 연산하여 $Tr(g^{a+bk})$를 구한다[1].

기존의 알고리즘에서는 iv의 연산을 하지 않기 때문에 $S_k(Tr(g))$값으로 만 vi의 연산을 실행하여 노출된 a값을 사용하여 $Tr(g^{a+bk})$를 구한다. 그러나 제안하는 알고리즘에서는 iv의 연산을 함으로써, a 값과 관계없이 공개 정보 $S_a(Tr(g))$로 $Tr(g^{a+bk})$를 구할 수 있다.

4.2. XTR 기반의 proxy protect를 만족하는 부분 위임 대리 서명

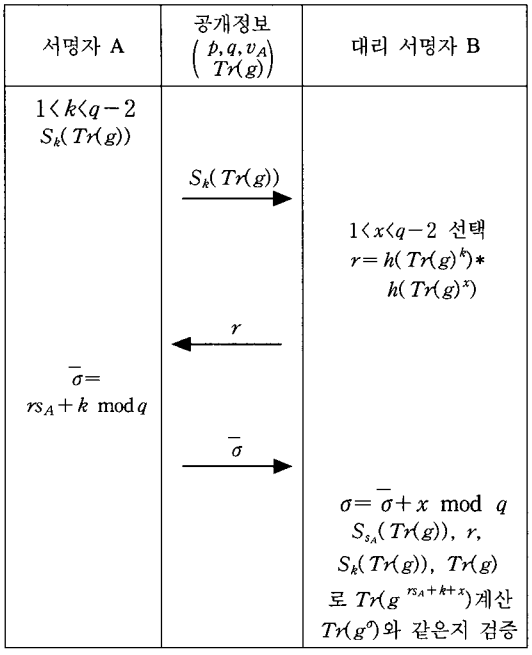
$p \equiv 2 \bmod 3$를 만족하는 $2^{169} < p < 2^{170}$인 소수 p와 $p^2 - p + 1$을 나누는 위수 $q(>6)$의 원소 g를 선택하여 $Tr(g)$를 계산한 후 공개한다. $s_A \in_R Z_{q-2}$인 A의 비밀키 난수 s_A를 생성하고, 그에 상응하는 공개키 $v_A = g^{s_A}$를 계산한다.

본 서명자 A가 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 사용하여 $\bar{\sigma} = rs_A + k \bmod q$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 제안하는 알고리즘에 의해 $Tr(g^{rs_A+k})$를 계산한다. 이 값과 대리서명자가 서명 생성시 계산한 $S_x(Tr(g))$값을 제안하는 알고리즘의 vi을 이용하여 $Tr(g^{rs_A+k+x})$를 계산한다. 이 값과 $Tr(g^\sigma)$로 얻은 값과 비교하여 σ의 유용성을 판별한다. B는 문서 m_p에 σ를 이용하여 대리 서명을 생성한다. 대리 서명자에 의해 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.

---

1) $M_0(c)^{-1} = \frac{1}{D}\begin{pmatrix} 2c^2 - 6c^p & 2c^{2p} + 3c - c^{p+2} & c^{p+1} - 9 \\ 2c^{2p} + 3c - c^{p+2} & (c^2 - 2c^p)^{p+1} - 9 & (2c^{2p} + 3c - c^{p+2})^p \\ c^{p+1} - 9 & (2c^{2p} + 3c - c^{p+2})^p & (2c^2 - 6c^p)^p \end{pmatrix}$



〔그림 2〕 XTR 기반의 proxy protect를 만족하는 부분 위임 대리 서명

4.3. XTR 기반의 증명서 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 만든 후 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

5.1. 안전성 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[11].

3장에서 제시된 대리 서명 프로토콜은 이산 대수 문제(DLP)를 기반으로 한다. 이러한 이산 대수 문제를 기반으로 하는 프로토콜에 대한 XTR 적용은 프로토콜 자체의 안정성을 그대로 유지하게 된다.

대리 서명의 요구사항 중 위조 방지는 비밀 정보 σ를 비밀 채널을 이용하여 전송하므로 만족할 수 있다. 또한 제 3자가 서명 값에 있는 $S_k(Tr(g))$, 공개 정보 $Tr(g)$와 v_{s_A}를 이용하여 공개키 v'를 생성하여 서명을 검증한다. 4장에서 제시된 2개의 프로토콜중 XTR 기반의 증명서 위임 대리 프로토콜은 DLP기반 프로토콜과 동일하게 proxy unprotect이므로 신원 확인과 부인 방지를 만족하지 못한다. 그러나 XTR 기반의 proxy protect를 만족하는 부분위임 서명 프로토콜에서는 비밀 정보 σ에 대리 서명자의 개인키 x가 포함되어 있으므로 검증 시 대리 서명을 작

〔표 2〕 XTR기반의 대리서명 프로토콜의 효율성 비교

구 분		DLP기반의대리서명	XTR기반의 대리서명
p		1024비트	170비트
q		512비트	160비트
키 길 이		1024비트	160비트
연산량	본서명자 (proxy 생성시)	pow mod 연산 1번(1024비트)	곱셈 연산 $8\log_2 k$번 (160비트)
	대리서명자 (proxy 생성시)	곱셈 연산 1번 pow mod 연산 1번(1024비트)	곱셈 연산 $8\log_2 x+1$번 (160비트)
	대리서명자 (proxy 검증시)	곱셈 연산 1번 pow mod 연산 1번(1024비트)	곱셈 연산 $48\log_2 p+136$번 (160비트)
전 송 량		1024비트의 K	1020비트의 $S_k(Tr(g))$

성한 사람의 신원 확인이 가능하고 또한 부인 방지를 만족한다.

즉 XTR기반의 대리 서명 프로토콜은 그 자체로서 DLP 기반의 프로토콜 안전성을 그대로 유지하게 되고, 따라서 대리 서명의 기본적인 요구사항 또한 만족한다.

5.2. 효율성 분석

본 논문에서는 Mambo와 Kim 등이 제안한 대리 서명 기법에 XTR 암호 시스템을 적용함으로써 프로토콜 진행 시 키길이의 감소와 연산량의 효율성을 높일 수 있다.

XTR은 160 비트의 키 길이로 1024 비트의 키 길이를 갖는 이산대수 문제에 기반한 암호 시스템과 동일한 안전성을 갖고 빠른 연산속도를 가진다[1,6,7].

표2는 XTR 기반과 DLP 기반에 대한 proxy protect를 만족하는 부분 위임 대리 서명 프로토콜의 효율성을 비교 분석하였다. DLP 기반의 프로토콜에서는 본 서명자가 proxy 생성시 K값을 생성하기 위해 pow mod 연산을 1번 하게된다. 또한 대리 서명자가 proxy 생성시 r을 생성하기 위해, 또한 서명의 검증을 위해 각각 pow mod 연산 1번과 곱셈 연산 1번이 필요하다.

XTR 기반의 프로토콜에서는 본 서명자의 proxy 생성 시 필요한 $S_k(Tr(g))$연산에는 $8\log_2 k$번의 곱셈 연산이 필요하다[1]. 대리 서명자 역시 proxy 생성시 r을 생성하기 위해 $8\log_2 k$+1번의 곱셈 연산이 필요하다. 또한 proxy 검증시 대리 서명자는 제안하는 알고리즘을 사용하게 된다. 제안하는 알고리즘은 ii에서 $8\log_2 p$번의 곱셈 연산, iv의 과정에서 $32\log_2 p$+68번의 곱셈 연산, vi의 과정에서 $8\log_2 p$+34번의 곱셈 연산이 필요하다. 즉 proxy검증시 총 $48\log_2 p$+136번의 곱셈 연산이 필요하다.

Ⅵ. 결 론

본 논문에서는 XTR 암호 시스템을 기반으로 하는 대리 서명 프로토콜을 제시하였다. 대리 서명 프로토콜을 XTR에 적용하기 위해서는 $Tr(g^a g^{bk})$의 연산을 반드시 해야한다. 기존의 XTR에서는 대리 서명의 생성 및 검증에 사용될 $Tr(g^a g^{bk})$의 연산을 구하기 위해서는 a와 b의 값을 알고 있어야 한다. a, b의 값을 알게 된다는 것은 본 서명자의 비밀키 또는 임시 비밀키 중 한 개가 대리 서명자가 알아야 한다는 것이다.

따라서 $Tr(g^a g^{bk})$의 연산을 b와 공개 정보를 이용하여 구할수 있는 알고리즘을 제시하였다.

이 알고리즘을 사용하여 XTR을 대리 서명에 적용하여 대리 서명의 비밀 정보 생성 및 검증 속도를 향상 시켰다. 또한 키 길이의 단축과 연산량 감소의 효율을 가져왔다. 이러한 연산 속도 향상과 키 길이 단축은 유선뿐만 아니라 무선인터넷에서도 효율적으로 사용될 수 있다.

향후 과제로서 기존의 대리 서명 프로토콜을 XTR에 적용하는 과정에서 새롭게 제안된 알고리즘의 연산량 축소가 필요하다. 또한 프로토콜 진행시 전송되는 전송량을 줄이기 위한 새로운 프로토콜 연구가 필요하다.

참 고 문 헌

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

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

[3] S. Kim, S. Park and D. Won, "Proxy signatures, revisited", Proceedings of International Conference on Information and Communications Security, pp.223~232, 1997.

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

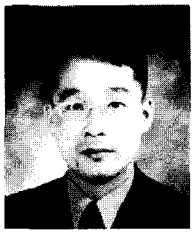
[5] N. Lee, T. Hwang, and C. Wang, "On Zhang's Nonrepudiable Proxy Signature Scheme", in Proceedings of ACISP'98-Australasian Conference on Information Security and Privacy, Vol. 1438 of Lecture Notes in Computer Science, pp.415~422, 1998.

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

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

〈著者紹介〉

이 재 욱 (Jae-wook Lee) 학생회원
2001년 2월 : 경북대학교 전자공학과 학사
2003년 2월 : 경북대학교 전자공학과 석사
2003년 3월~현재 : 경북대학교 전자공학과 박사과정
<관심분야> XTR 암호 시스템, 정보보호



전 동 호 (Dong-ho Jeon) 학생회원
2000년 2월 : 밀양대학교 컴퓨터공학과 학사
2002년 2월 : 경북대학교 정보통신학과 석사
2002년 3월~현재 : 경북대학교 정보보호학과 박사과정
<관심분야> 스마트카드, 무선랜보안, 정보보호

최 영 근 (Young-Geun Choe) 학생회원
1995년 2월 : 경북대학교 전자공학과 졸업
2001년 2월 : 경북대학교 전자공학과 석사 졸업
2001년 3월~현재:경북대학교 전자공학과 박사과정
<관심분야> 정보보호 및 보안 기술, 정보 보호 응용 기술

김 순 자 (Soon-Ja Kim) 정회원
1975년 2월 : 경북대학교 수학 교육학과 학사
1977년 2월 : 경북대학교 수학과 석사
1988년 2월 : 계명대학교 수학과 박사
1993년 4월~현재 : 경북대학교 전자·전기 공학부 교수
<관심분야> 정보보호 및 보안 기술, 정보 보호 응용 기술