이동성 공격자에 강인한 문턱 RSA 서명방식

박찬섭*, 염대현**, 이필중**

포항공과대학교 정보통신대학원*, 포항공과대학교 전자전기공학과**

{ballpark, dhyum, pjl}@postech.ac.kr

Robust Threshold RSA Signature Scheme Against a Mobile Adversary

Chan Sup Park*, Dae Hyun Yum**, Pil Joong Lee**

GSIT*, Department of EEE**, POSTECH

{ballpark, dhyum, pjl}@postech.ac.kr

요 약

사전 대응적 안전성은 이동성 공격자에 대해 오랜 기간 안전하게 유지 및 사용되어야 할 암호학적 키들의 안전성을 증가시킨다. 최근에 Shoup은 매우 실제적인 문턱 RSA 서명방식을 제안하였는데, 그의 논문에서는 사전 대응적 안전성에 대해서 고려하지 않았다. 본 논문에서는 Shoup이 제안한 서명방식과 사전 대응적 안전성을 고려하여 이동성 공격자에 강인한 문턱 RSA 서명방식을 제안한다.

I. 서론

(t, l)문턱 서명방식이란 l명의 참여자(또는, 서버)중에 t명의 참여자로 구성된 임의의 부분집합은 서명을 만들어 낼 수 있으나, t명 보다 작은 참여자로 구성된 임의의 부분집합은 유효한 서명을 만들어 낼 수 없도록 하는 서명방식이다. 문턱 서명방식은 Shamir[13]의 비밀분배를 이용하여 크게 이산대수 문제(Discrete Logarithm Problem, DLP)의 어려움에 기반한 방식[5]과 인수분해 문제(RSA)의 어려움에 기반한 방식[4, 9, 12]으로 나누어진다. RSA방식은 DLP방식에 비해서 기술적으로 좀 더 어려움이 있는데, 그 이유는 소수도 아니고 위수도 알려져 있지 않은 군에서 동작해야 하기 때문이다. 그럼에도 불구하고 많은 연구가 이루어지고 있는 이유는 RSA 공개키 시스템 기반이 충분히 보급되어 있기 때문이다.

문턱 서명방식에 요구되는 중요한 특성 중 하나는 강인성(Robustness)이다. 강인성이란 최대 $t-1$명의 참여자를 성공적으로 공격할 수 있는 공격자가 있다 할지라도 나머지 정직한 참여자에 의해 시스템이 안전하게 동작할 수 있다는 것이다. Shoup[14]은 이전의 RSA에 기반한 문턱 서명방식의 몇 가지 문제점들을 개선하고, RSA 문제가 어렵다는 가정하에 랜덤 오라클 모델[1]에서 안전하고 강인한 서명방식을 제안하였다. 또한, Damgard와 Koprowski[3]가 분배 프로토콜에서 RSA키를 생성하는 알고리즘[2, 8]을 이용하여 Shoup이 가정한 분배자를 제거했으나 전체적인 내용은 Shoup이 제시한 방식과 유사하다.

문턱 서명방식 뿐만 아니라 현재 사용 중인 다양한 암호시스템에서 요구되는 중요한 특성은 사전 대응성(Proactivness)[10, 11]이다. 사전 대응적 안전성이란 이미 분배된 그리고 오랜 기간 안전하게 유지 및 사용 되어야 할 암호학적 키들을 이동성 공격자(예를 들면, 해커, 바이러스, 또는 부정직한 관리자)에 대해 보다 안전하게 만들어 주는 것이다. 실제 (t, l)문턱 서명방식에 사전 대응적 안전성이 결합되면, 일반 시스템에서는 공격자가 전체 l명의 참여자 중 임의의 $t-1$명을 공격하는데 긴 시간이 주어지는 반면에 사전 대응적 시스템에서는 임의의 $t-1$명을 공격하는데 매우 짧은 시간 밖에 주어지지 않는다. 즉, 각 참여자들에게 분배된 키의 값들을 주기적으로 랜덤하게 갱신시켜(전체적인 키의 값은 유지), 과거에 최대 $t-1$명의 참여자를 공격하여 얻어냈던 이동성 공격자의 키에 대한 정보는 키를 갱신한 이후에는 사용할 수 없게 된다. 이와 같이, 사전 대응적 안전성은 오랜 기간 유지 및 사용되어야 할 암호시스템 기반구조에 매우 중요한 요소라고 할 수 있다.

본 논문은 매우 효율적이고 강인한 것으로 알려진 Shoup[14]의 문턱 서명방식과 사전 대응적 안전성을 얻는 방법을 제시한 [7]을 변형하여 이동성 공격자에 강인한 문턱 RSA 서명 방식을 제안한다.

II. 제안된 서명 방식

우리는 l명의 참여자로 구성된 임의의 집합 $\{1,2,\ldots,l\}$과 분배자 그리고 공격자를 가정한다. 임의의 t명의 참여자로 구성된 부분집합을 S라고 하자. 분배자는 공개키 PK를 생성하고 각 참여자들에게 비밀키를 분배한다. 분배가 완료된 후, 분배자는 프로토콜에 참여하지 않는다. 각 참여자들은 비밀채널로 연결되어 있으며 인증된 공개채널에도 접근할 수 있다.

1. 키 생성과 비밀분배

분배자가 키를 생성하고 Shamir[13]의 비밀분배 방식을 이용하여 각 참여자들에게 비밀키를 분배한다.

분배자는 같은 크기의 두 큰 소수 p와 q를 랜덤하게 선택한다. $p=2p'+1, q=2q'+1$, p'와 q' 또한 소수이다. RSA 모듈러스 n은 p와 q의 곱이되고 p'와 q'의 곱은 $\Phi(n)/4$가 된다. RSA 공개 지수 $e(>l)$가 소수가 되도록 선택한다. 공개키 PK는 (n,e)가 된다. 다음에, $de\equiv 1 \bmod \Phi(n)/4$이 되도록 $d\in Z$를 계산하고 다음의 다항식을 생성한다.

$$f(X)=\sum_{i=0}^{t-1} a_i X^i \in Z[X]$$

$a_0=d$ 이고 $a_i \in_R Z_{\Phi(n)/4}$ $(1\le i\le t-1)$.

분배자가 다음과 같이 계산해서 각 참여자들에게 s_i를 분배한다.

$$s_i=f(i) \bmod \Phi(n)/4 \quad (1\le i\le l) \qquad (1)$$

s_i는 참여자 i의 비밀키가 된다. Z_n^*에서 가장 큰 위수를 갖는 g를 선택해서 공개한다. 그리고, Z_n^*에서 제곱이 되는 부분군을 Q_n이라 표시하면, $v\in_R Q_n$을 선택해서 $v_i=v^{s_i\cdot \Delta^2}$ $(1\le i\le l)$를 계산한다. 여기서, Δ는 $l!$이다. 분배자는 검증키 $VK=v$와 참여자 i의 부분 검증키 $VK_i=v_i$를 공개한다.

위와 같이 각 참여자들에게 분배된 s_i로부터 (t,l)문턱 서명을 얻을 수 있는데 그 원리는 다음과 같다. l명의 참여자 중 t명의 참여자로 구성된 임의의 부분집합 S, 그리고 $i\in\{1,2,\ldots..l\}\setminus S$와 $j\in S$에 대해서, 아래와 같은 식을 정의하자.

$$\lambda_{i,j}^S=\Delta\frac{\prod_{j'\in S\setminus\{j\}}(i-j')}{\prod_{j'\in S\setminus\{j\}}(j-j')} \in Z \qquad (2)$$

식(2)의 $\lambda_{i,j}^S$은 Lagrange 보간공식으로부터 유도되고 다음과 같은 식을 얻을 수 있다.

$$\Delta\cdot f(i)\equiv \sum_{j\in S}\lambda_{i,j}^S f(j) \bmod \Phi(n)/4 \qquad (3)$$

2. 사전 대응적 프로토콜

이동성 공격자를 다루기 위해 통상적으로 사용되는 시각(하루, 일주일, 한달 등)을 사용한다. 사전 대응적 프로토콜이 수행되기 전의 시각을 $u-1$이라고 하면, 프로토콜이 성공적으로 완료된 후의 시각은 u가 된다. 따라서, 사전 대응적 프로토콜을 수행하기 위해서는 임의의 시각 $u-1$에서 t명의 참여자로 구성된 임의의 부분집합 S가 결정된다. 그 다음에 "Poly-to-Sum" 프로토콜이 수행된 후 "Sum-to-Poly" 프로토콜이 수행된다. 위의 두 프로토콜이 성공적으로 완료된 후, 각 참여자의 비밀키는 시각 u에서 새로운 비밀키 값으로 갱신된다.

1) Poly-to-Sum 프로토콜

프로토콜은 어떤 시각에 임의의 t명의 참여자에 의해서 수행되고, (t,l) 다항식 기반의 비밀공유를 (t,t) 덧셈식 기반의 비밀공유로 변환한다. 이 프로토콜의 목적은 현재 상태의 랜덤화를 제공하는 것이며, 다음과 같이 수행된다.

- 비밀분배: 참여자 $i\in S$

• $r_{i,j}\in_R Z_n$ $(j\in S\setminus\{i\})$를 선택한다.

• 다음과 같이 $r_{i,i}$를 계산한다.

$$r_{i,i}=s_i\cdot\lambda_{0,i}^S-\sum_{j\in S\setminus\{i\}} r_{i,j}\in Z$$

• $r'_{i,j}\in_R Z_n$선택하고 $r_{i,j}$와 $r'_{i,j}$를 참여자 j에게 안전하게 전송한다.

• $R_{i,j}\equiv v^{r_{i,j}\cdot\Delta^2} g^{r'_{i,j}}$와 $U_i\equiv\prod_{j\in S} g^{r'_{i,j}}$를 공개한다.

- 비밀검증과 발생:

• 모든 $i\in S\setminus\{j\}$에 대해서 각 참여자 $j\in S$는 다음과 같이 검증한다.

$$(v_i)^{V_1}\equiv^{?}(U_i^{-1}\prod_{b\in S}R_{i,b})^{V_2}$$

여기서, $V_1=\prod_{b\in S\setminus\{i\}}(0-x_b)$이고 $V_2=\prod_{b\in S\setminus\{i\}}(x_i-x_b)$ 이다. 그리고, $r_{i,j}\le^{?} n-1$, $v^{r_{i,i}\cdot\Delta^2}g^{r'_{i,i}}\equiv^{?}R_{i,j}$를 검사한다.

만약 검증과정이 통과되지 않으면 참여자 i는 제거되고 새로운 S가 선택된다.

• 모든 검증이 통과하면 참여자 j의 새로운 비밀은 $s'_j = -r''_j + \sum_{i \in S} r_{i,j}$ 가 된다. 여기서, r''_j은 Δ^2이 s'_j을 나누고 $0 \le r''_j < \Delta^2$이다. 그리고 참여자 j는 r''_j를 공개한다.

2) Sum-to-Poly 프로토콜

이 프로토콜의 목적은 "Poly-to-Sum"을 통해 얻은 덧셈식 비밀공유의 값을 $t-1$차 다항식의 값으로 랜덤하게 재공유 하는 것이다. 프로토콜은 덧셈식 비밀공유 s'_i와 공개값 $W = \sum_{j \in S} r''_j$을 계산해서 다음과 같이 수행된다.

- 참여자 $i \in S$:

• 가장 작은 번호를 가진 참여자만 다음과 같이 계산한다.

$s'_i = s'_i + W$

그리고 참여자 i는 $t-1$차 랜덤 다항식을 다음과 같이 생성한다.

$r'_i(x) = \sum_{k=0}^{t-1} a_{i,k} \cdot x^k$, (여기서, $a_{i,0} = s'_i$이고 $a_{i,k} \in_R \{0, \Delta, \ldots, 2\Delta^3 t n^{2+\varepsilon}\}$.)

- 참여자 $i \in S$:

• 검증을 위해 모든 $v^{a_{i,k} \cdot \Delta^2}$를 공개하고 안전하게 $w_{i,j}$를 참여자 j에게 전송한다.

$w_{i,j} = r'_i(j) \in Z \ (j \in \{1, 2, \ldots, l\})$

- 각 참여자 j는 다음을 수행한다:

• 다음의 방법을 이용해서 적절한 비밀인지 검증한다. 이것은 [6]에서 사용한 방법과 유사하다.

$v^{w_{i,j} \cdot \Delta^2} \equiv^{?} \prod_{k=0}^{t-1} (v^{a_{i,k} \cdot \Delta^2})^{j^k} \bmod n$

$v^{a_{i,0} \cdot \Delta^2} =^{?} v^{s'_i \cdot \Delta^2}$.

$|w_{i,j}| \le^{?} \Delta^3 t^2 n^{2+\varepsilon} + 2\Delta^3 n^{2+\varepsilon} t^2 l^{t+1} + ln^2$

$w_{i,j}$가 Δ의 배수이어야 한다.

- 만약 검증이 통과되지 않으면 "Poly-to-Sum"프로토콜로 돌아간다.

- 이제 이전의 값을 지우고 새로운 비밀키 값 s_i를 갖게 되고 $v^{s_i \cdot \Delta^2}$를 공개한다.

$$s_i = \sum_{j \in S} w_{j,i} = \sum_{j \in S} r'_j(i) \qquad (4)$$

3. 부분 서명의 발생과 검증

각 참여자가 어떤 메시지 M에 대해서 부분 서명을 어떻게 발생시키고 검증하는지 살펴보자. 메시지 M에 해쉬한 결과를 x라고 하자. 여기서, H는 임의의 길이의 입력에 대해서 고정된 길이의 출력을 만들어 내는 해쉬 함수[15]이다. 각 참여자 i의 부분 서명은 "검증의 증거"와 함께 다음과 같이 계산된다.

$$x_i = x^{2\Delta \cdot s_i} \in Q_n \qquad (5)$$

검증의 증거는 기저 $\tilde{x} = x^{4\Delta}$에 x_i^2의 이산대수와 기저 v^{Δ^2}에 v_i의 이산대수가 같다는 것이다. 즉, 해쉬 함수를 이용함으로써 프로토콜을 상호반응적이지 않게 만들고, 검증의 증거를 통해서 서명 방식에 강인성이 보장된다. 좀 더 자세히 살펴보면 다음과 같다. H를 해쉬 함수라 하고, 그것의 출력은 L_1비트 길이의 정수라고 하자.

검증의 증거를 구성하기 위해 각 참여자 i는 $r \in_R \{0, \ldots, 2^{L(n)+2L_1} - 1\}$을 선택하고 다음과 같이 계산한다. 여기서, $L(n)$은 n의 크기이다.

$v' = (v^{\Delta^2})^r, x' = \tilde{x}^r, c = H(v^{\Delta^2}, \tilde{x}, v_i, x_i^2, v', x')$

$z = s_i \cdot c + r$

검증의 증거는 (z, c)가 된다. 검증의 증거를 확인하기 위해 각 참여자는 다음과 같이 확인할 수 있다.

$c = H(v^{\Delta^2}, \tilde{x}, v_i x_i^2, v^{z \cdot \Delta^2} v_i^{-c}, \tilde{x}^z x_i^{-2c})$

4. 부분 서명의 결합과 검증

l명의 참여자 중 임의의 t명의 참여자로 구성된 부분집합 S로부터 유효한 서명값을 가지고 있다고 하자. $S = \{i_1, i_2, \ldots, i_t\} \subset \{1, 2, \ldots, l\}$.

식(5)에 의해서, $x_{i_j}^2 = x^{4\Delta s_{i_j}}$ 가 되고 부분 서명값을 결합하기 위해 다음과 같이 계산한다.

$$\begin{aligned} w &= x_{i_1}^{2\lambda_{0,i_1}^S} \cdot \cdots \cdot x_{i_t}^{2\lambda_{0,i_t}^S} \\ &= (x_{i_1}^2)^{\lambda_{0,i_1}^S} \cdot \cdots \cdot (x_{i_t}^2)^{\lambda_{0,i_t}^S} \\ &= x^{4\Delta s_{i_1} \cdot \lambda_{0,i_1}^S} \cdot \cdots \cdot x^{4\Delta s_{i_t} \cdot \lambda_{0,i_t}^S} \\ &= x^{4\Delta (s_{i_1} \cdot \lambda_{0,i_1}^S + \cdots + s_{i_t} \cdot \lambda_{0,i_t}^S)} \end{aligned} \qquad (6)$$

여기서, λ의 값들은 식(2)에서 정의된 정수이다. 식(6)은 식(3)으로부터 $w = x^{4\Delta^2 \cdot d}$로 계산되고, $w^e = x^{e'}$을 얻게 된다. 여기서, e'은 $4\Delta^2$이다. 그런데, $\gcd(e, e') = 1$이므로 Extended Euclidean Algorithm을 이용하여 $e'a + eb = 1$이 되는 a와 b를 구할 수 있다. 따라서, x의 서명값을 y라 하

고 $y=w^a x^b$ 라 두면 $y^e=x$가 됨을 확인할 수 있다.

III. 결론

본 논문에서는 문턱 RSA서명방식과 사전 대응적 안전성을 결합하여 안전하고 강인한 서명방식을 제안하였다. 제안한 방식은 랜덤 오라클 모델에서 증명 가능한 안전성과 강인성을 갖는 Shoup의 방식과 사전 대응적 안전성을 갖는 [7]을 변형하여 분배된 비밀키에 "Poly-to-Sum"과 "Sum-to-Poly"프로토콜을 수행시킴으로써 이전에 가지고 있던 비밀키를 새롭게 갱신하여 이동성 공격자에 강인하게 설계 되었다.

참고문헌

[1] M. Bellare and P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols," 1st ACM Conference on Computer and Commnications Security, pp. 62-73, 1993.

[2] D. Boneh and M. Franklin, "Efficient Generation of Shared RSA keys," CRYPTO '97, LNCS 1233, pp. 425-439, 1997.

[3] I. Damgard and M. Koprowski, "Practical threshold RSA signatures without a trusted dealer," EUROCRYPT '01, LNCS 2045, pp. 152-165, 2001.

[4] Y. Desmedit and Y. Frankel, "Shared generation of authenticators and signatures," CRYPTO '91, pp. 457-469, 1991.

[5] Y. Desmedt and Y. Frankel, "Threshold cryptosystems," CRYPTO '89, pp. 307-315, 1989.

[6] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," 28th Annual Symposium on Foundations of Computer Science, pp. 427-437, 1987.

[7] Y. Frankel, P. Gemmell, P. D. MacKenzie, and M. Yung, "Optimal-Resilience proactive public-key cryptosystems," 38th Annual Symposium on Foundations of Computer Science, pp. 384-393, 1997.

[8] Y. Frankel, P. D. MacKenzie, and M. Yung, "Robust Efficient Distributed RSA-Key Generation," Proc. of STOC '98, pp. 663-672, 1998.

[9] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust and Efficient sharing of RSA functions," CRYPTO '96, LNCS 1109, pp. 157-172, 1996.

[10] A. Herzberg, S. Jarecki, H. Krawczyk, M. Yung, "Proactive Secret Sharing, or: how to cope with perpetual leakage," CRYPTO '95, LNCS 963, pp. 339-352, 1995.

[11] R. Ostrovsky and M. Yung, "How to withstand mobile virus attacks," 10th Annual ACM Symposium on Principles of Distributed Computing, pp. 51-61, 1991.

[12] A. De Saints, Y. Desmedt, Y. Frankel, and M. Yung, "How to share a function securely," 26th Annual ACM Symposium on the Theory of Computing, pp. 522-533, 1994.

[13] A. Shamir, "How to share a secret," Comm. of ACM, pp. 612-613, 1979.

[14] V. Shoup., "Practical Threshold Signatures," EUROCRYPT '00, LNCS 1807, pp. 207-220, 2000.

[15] TTA.IS-10118: 해쉬함수표준 - 제2부: 해쉬함수알고리즘(HAS-160), 한국정보통신기술협회, 1998.