효율적인 임계 암호시스템 구현을 위한 능동적 비밀 분산에서의 빠른 공유 갱신에 관한 연구

이윤호[0] 김희열 이재원 정병천 윤현수
한국과학기술원 전산학 전공
(yhlee[0], hykim, jaewon, bcchung, hyoon)@kiss.or.kr

Computer Science Division, EECS Dept. , KAIST

Youn-Ho Lee[0] Jaewon Lee ByungChun Chung, Hyunsoo Yoon
Computer Science Division, EECS Dept. , KAIST

요 약

임계 암호시스템은 현대 암호학에서 중요한 한 축을 이루는 암호학의 한 분야이다. 본 논문에서는 임계 암호시스템의 근간이 되는 비밀 분산(Secret Sharing)의 한 분야인 (k,n) threshold scheme에서 능동적 비밀 분산(Proactive Secret Sharing)을 위한 공유(share)갱신 방법을 개선한 새로운 공유 갱신 방법을 제안한다. 이전 방법은 각 참여자당 $O(n^2)$ 의 모듈라 멱승 연산을 수행하는데 비하여 제안 방법은 $O(n)$ 의 모듈라 멱승 연산만으로 공유 갱신이 가능하다. 이와 함께 본 논문에서는 $k < (1/2)n - 1$ 인 경우에 대하여 제안 방법의 안전함을 증명한다.

1. 서 론

임계 암호시스템(Threshold Cryptography)은 일반적인 암호시스템에서 사용하는 하나의 키를 여러 서버에게 분산하여 보호할 수 있는 기술이다. 예를 들어 공인 인증기관의 개인키와 같은 중요한 키를 하나의 서버에 저장하여 사용하였을 경우 개인키를 갖고 있는 서버만을 공격하면 개인키가 유출되게 된다. 그러나 임계 암호시스템에서는 하나의 개인키를 여러 개의 키에 대한 공유(share)로 나누어서 다수의 서버에게 공유를 나누어 주게 되며, 원래의 하나의 개인키로써 수행하던 작업을 다수의 공유를 이용하여 수행할 수 있다. 따라서 공격자는 다수의 서버를 공격하여 일정한 수 이상의 공유를 모을 경우에만 원래의 개인키를 알 수 있게 되는 어려움이 있게 된다[1].

임계 암호시스템의 방법은 암호학의 한 분야인 비밀 분산을 근거로 한다. 비밀 분산은 다수가 어떤 비밀 정보를 분산, 공유하여 일정한 조건을 만족해야만 원래의 비밀 정보를 복구할 수 있는 것을 목표로 하는 안전한 프로토콜을 찾는 분야이다. 이 중 (k,n) threshold scheme은 비밀 정보를 n 개의 공유로 나누어서 그 중 k 개 이상의 공유가 모일 경우 비밀 정보를 복구할 수 있고 그 미만의 공유가 모일경우 비밀 정보에 관하여 어떠한 정보도 알 수 없도록 하는 비밀 분산의 한 분야이다. 이 분야는 1979년 Shamir가 제안하였으며 다항식 보간법 (polynomial interpolation)을 이용한 (k,n) threshold scheme을 구현하기 위한 방법도 동시에 제안하였다[2].

비밀 분산은 방법의 안전성의 증가를 위하여 많은 발전이 있어왔으며 검증 가능한 비밀 분산(Verifiable Secret Sharing)[3-5] , 능동적 비밀 분산 (Proactive Secret Sharing)[6]등으로 발전하여 왔다.

. 1995년 Jarecki는 능동적 비밀 분산을 위한 (k,n) threshold scheme 에서의 공유 갱신 방법을 제안하였다. 이 방법은 각 참여자 당 $O(kn)$ 의 모듈라 멱승을 필요로 한다. 일반적으로 k 값은 $O(n)$에 해당하는 값이므로 약 $O(n^2)$ 의 연산을 필요로 한다. 이것은 상당한 량의 계산량이며 실용적으로 쓰이기에는 부적합하다.

본 논문에서는 (k,n) threshold scheme에서 각 참여자의 계산량이 $O(n)$을 갖는 새로운 방법을 제안한다. 이것은 이전 방법이 각 참여자당 자신만이 알고 있는 개인값이 k 개인 반면, 제안 방법은 상수개인 것에 근거한다. 또한 제안 프로토콜의 안전성은 $k-1$ 명의 공격자가 프로토콜에 참여해도 다른 참여자가 갖게 되는 개인값을 알아 낼 수 없다는 것으로써 증명하며 이 때 이러한 k 의 값이 $(1/2)n-1$ 보다 작을 경우, 안전하다는 것을 증명한다. 증명 방식의 모델은 이전의 연구 논문을 참조한다[7-8].

2. 관련 연구

Jareki가 제안한 능동적 비밀 분산 방법은 공격자가 특정 주기를 두어 주기마다 프로토콜에 침입하는 이동 공격자 모델[9]에서 안전한 비밀 분산을 수행 가능하도록 하기 위하여 제안되었다[6]. 능동적 비밀 분산의 핵심은 공유 갱신에 있으며 각 참여자는 특정 주기 가 지나면 공유를 갱신함으로써 공격자는 한 주기 내에 임계값 이상의 공유를 얻지 못하면 원래의 비밀 정보를 알아낼 수 없도록 하는 것이다. 이것은 각 참여자마다 자신만이 알고있는 계수를 이용하여 생성한 k 차 다항식을 사용하기 때문에 이 다항식을 다른 모든 사람들이 검증하기 위하여 각 참여자는 전체 $O(kn) = O(n^2)$ 의 연산을 수행하게 된다.

3. 제안 프로토콜

제안 프로토콜은 확장 분산 Coin Flip 프로토콜과 공유 갱신 프 일한 수열의 값을 얻는다.

로토콜로 이루어져 있다. 두 과정을 모두 수행해야 공유 갱신 작업이 끝나게 되며 복잡도 분석도 두 프로토콜을 합하여서 분석한다. 제안 프로토콜의 방법은 아래와 같다. 사용하는 모든 기호 및 알고리즘은 [10]에서 사용된 기호 및 알고리즘을 인용한다.

프로토콜 1. 확장 분산 Coin Flip 프로토콜

입력값

각 참여자 i는 다음과 같은 입력값을 생성한다.

$\{\alpha_{i1},\alpha_{i2},...,\alpha_{in}\} s.t. (\sum_{j=1}^{n}\alpha_{ij}) \bmod p \equiv 0$

$\{\beta_{i1},...,\beta_{in}\}$: GF(p)상의 난수

결과값

각 참여자 i 는 다음과 같은 결과값을 얻는다.

$\{\delta_{ij}\}_{j=1,..,n},\{\lambda_{ij}\}_{j=1,..,n}$ $s.t.$ $\sum_{l=1}^{n}\delta_{lj} (\bmod p) \equiv 0$

수행 과정

1. 각 참여자 i 는 α_{ij},β_{ij} 값을 참여자 j 에게 보낸다. ($j=1,..,n$ $j \neq i$)

2. i는 $t_i = \sum_{l=1}^{n}\beta_{il}$ 을 계산한다.

3. i 는 자신의 수열들에 대한 검증값 $g^{\alpha_{ij}} h^{\beta_{ij}}, h^{t_i}$ 값을 동보한다. ($j=1,..,n$ $j \neq i$)

4. i 는 다른 참여자들의 검증값을 이용하여 다른 참여자에게서 받은 값들을 다음과 같은 수식을 이용하여 검증한다.

$$\prod_{j=1}^{n}(g^{\alpha_{lj}} h^{\beta_{lj}}) \bmod q \stackrel{?}{=} h^{t_l} \bmod q$$

$$g^{\alpha_{li}\beta_{li}} \bmod q \stackrel{?}{=} (g)^{\alpha_{li}} \times (h)^{\beta_{li}} \bmod q$$

$(l=1,2..,i-1,i+1,..,n)$

5. 검증작업이 실패할 경우 그 값을 받은 참여자 i는 값을 보낸 참여자 j에 대한 검증이 실패했다는 것을 동보하며, 동시에 자신이 j에게 받은 값 및 검증값을 동보한다.

6. 의심을 받은 참여자 j 는 자신이 생성한 입력값 $\{\{\alpha_{jl}\}_{l=1,..,n},\{\beta_{jl}\}_{j=1,..,n}\}$을 동보하며 이 때 모든 참여자는 잘못되었다는 선언을 한 참여자 i가 동보한 값과 참여자 j가 동보한 값 및 참여자 i가 받은 검증값을 비교한다.

7. 만약 세개의 값이 동치일 경우 의심을 선언한 i가 제거되며 그렇지 않은 경우 의심 받은 참여자 j가 제거된다.

8. 5~7의 프로토콜의 과정에서 제거된 모든 참여자의 수를 제거한다. 또한 5~7의 과정에서 올바른 참여자로 판명될 경우 다시 입력값을 생성하여 1~7의 과정을 반복한다.

9 . 프로토콜을 적법하게 수행한 참여자들을 차례로 다시 순서를 짓는다. $1,...,n'(n' \leq n)$

10. $A\lg(1,n',k+1,n')$을 수행하여 결과값이 되는 수열의 값을 $\{\gamma_{lj}\}_{j=1,..,k+1\ \ l=1,...,n'}$ 이라 한다. 이 때 모든 참여자는 동

12. $\delta_{ij} = \sum_{l=1}^{k+1}\alpha_{\gamma_{jl}i} \bmod p, \lambda_{ij} = \sum_{l=1}^{k+1}\beta_{\gamma_{jl}i} \bmod p$ 를 계산하여 프로토콜의 결과 값을 얻는다. ($j=1,2,..,n'$) □

프로토콜 2. 공유 갱신 프로토콜

입력값

확장 분산 Coin Flip 프로토콜을 올바르게 수행한 $n'(\leq n)$명이 본 프로토콜에 참여하며 프로토콜의 참가자 i는 다음과 같은 입력값을 갖는다.

- 확장 분산 Coin Flip 프로토콜의 결과값
- 확장 분산 Coin Flip 프로토콜에서 동보한 검증값

$\{g^{c_{ij}} h^{\beta_{ij}}\}_{j=1,...,n'\ i=1,...,n'}$

- 갱신되어야 할 이전 공유 s_i
- 모든 참여자가 알고 있는 참가자의 ID ($=ID_i(i=1,..,n')$)

결과값

프로토콜을 적법하게 수행한 참가자 i는 다음과 같은 결과값을 얻는다.

$s_i' = s_i + g(ID_i)$ s.t.

$g(x) = a_{k-1}x^{k-1} + ... + a_1 x$ $(a_{k-1} \neq 0)$

수행 과정

1. 각 참여자는 임의로 y_i, d_i, e_i를 생성한다.

2. $y_i + d_i$ 값을 동보한다.

3. 각 참여자에 대한 다항식을 생성한다. 각 참여자의 다항식을 $f_j(x)$라 하면 아래와 같은 방법으로 다항식을 생성한다.

$f_j(x) = c_{jk-1}x^{k-1} + c_{jk-2}x^{k-2} + .. + c_{j1}x$

$(c_{jk-1} | c_{jk-2} | ... | c_{j1}) = Hash\_Expansion(ID_j, k-1)$

$(j=1,2,..,n)$

4. 다른 참여자로부터 받은 $y_j + d_j$ 값 및 자신이 생성한 값 $y_i + d_i$를 이용하여 아래와 같은 값을 만들어 낸다.

$A_i = \sum_{l=1}^{n}(y_l + d_l) f_l(ID_i)$

5. 다른 참여자 j 에게 $d_i f_i(ID_j) + \delta_{ij} (= B_{ij})$ 및 $e_i f_i(ID_j) + \lambda_{ij} (= C_{ij})$를 보낸다. 또한 자신의 계산에 사용되는 값 B_{ii}, C_{ii}도 계산한다.

6. 5에서 보낸 값들의 검증을 위한 검증값 $g^{d_i} h^{e_i} (= V_i)$를 동보한다.

7.다른 참여자들이 동보한 값들 및 다음과 같은 수식의 만족 여부를 이용하여 다른 참여자들이 보낸 값들을 검증한다. 이 때에 확장 분산 Coin Flip 프로토콜의 4에서 동보한 값을 이용한다.

$g^{B_{ji}C_{ji}} \stackrel{?}{=} V_j^{f_j(ID_i)} * \prod_{l=1}^{k+1} g^{\alpha_{rlij}} h^{\beta_{rlij}} (j = 1,..,n' \quad j \neq i)$

8 . 7의 검증과정이 실패할 경우 확장 Coin Flip 프로토콜에서의 검증 과정이 실패할 경우와 유사한 방법으로 적법하지 못한 프로토콜 참여자를 제거한다.

9. 4 과정에서 생성한 A_i 값 중 제거된 참여자로부터 받은 값을 제거한다. 이 연산의 결과로 생성된 결과값을 A_i' 은 다음과 같은 식으로 표현할 수 있다. 여기서 $DISQUAL$ 은 프로토콜에서 적법한 행위를 하지 않아 제거된 참여자의 집합이다.

$$A_i' = A_i - \sum_{j \in DISQUAL} (y_j + d_j) f_j(ID_i)$$

10. 7의 검증 작업을 통과한 참여자를 다시 순서 짓는다. 이 때의 참여자를 $(1,...,n'')$ $(n'' \leq n')$ 이라 한다.

11. 9 과정에 생성한 A_i' 값에서 5에서 받은 값들을 빼어 주어 공유 갱신에 사용되는 값 D_i 를 생성한다.

$$D_i = A_i' - \sum_{j=1}^{n''} (B_{ij})$$

12. 기존의 공유 s_i 에 새로운 값 D_i 를 더하여 새로운 공유 s_i' 을 만든다.

13. 이전 공유를 s_i 를 삭제한다. □

4. 프로토콜의 안전성 증명

제안 프로토콜은 공격자가 고정된 공격자(Static Adversary)로 가정했을 때에 안전하며 이것은 제안 프로토콜에 대한 시뮬레이터를 생성하여 공격자가 Decryption Oracle을 통하여 특정 값을 추측하여도 공격자의 관점을 변화시키지 않고 전혀 다른 결과값을 낼 수 있기 때문이다. 증명 과정과 시뮬레이터 의 생성 과정은 [10]에 자세히 기술되어 있으며 본 논문에서는 구체적인 방법은 생략한다.

5. 복잡도 분석과 성능 측정

Jarecki가 제안한 이전 프로토콜은 모든 참여자가 프로토콜에 적법한 행동을 할 경우 각 참여자당 $(k+1)(n+1)-1$ 번의 모듈라 멱승 연산을 수행한다[6]. 일반적으로 $k \approx (1/3)n$ 의 값을 갖게 되므로 이전 프로토콜의 모듈라 멱승 연산량은 $O(n^2)$ 이라 말할 수 있다. 이에 반하여 제안 프로토콜은 모든 참여자가 프로토콜에 적법한 행동을 할 경우 확장 분산 Coin Flip 프로토콜에서 $4n-1$ 수행하고, 제안 공유 갱신 프로토콜에서 $3n-2$ 번의 모듈라 멱승 연산을 수행한다. 따라서 전체 $7n-3$ 번의 모듈라 멱승 연산을 수행한다. 다음의 그래프는 q 의 비트 길이가 1024bit일 경우 Pentium III 450Mhz 환경에서 이전 프로토콜과 제안 프로토콜의 수행 시간을 비교한 것이다. 각 프로토콜의 수행 시간은 모듈라 멱승의 연산량과 거의 정비례한다. 따라서 프로토콜 수행 시간의 거의 대부분을 모듈라 멱승 연산이 차지한다는 것을 알 수 있다.

5. 결론

본 논문에서는 (k,n) threshold scheme의 능동적 비밀 공유

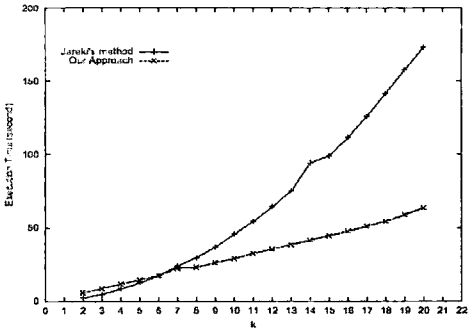

그림 1 수행시간 (n=3k)

에서 이전 공유 갱신 방법보다 연산량이 작은 새로운 공유 갱신 방법을 제안하였다. 이전 방법이 각 참여자당 연산량이 $O(n^2)$ 인 것에 비하여 제안 방법은 각 참여자당 연산량이 $O(n)$ 만으로도 공유 갱신이 가능하다. 또한 제안 방법의 안전함을 시뮬레이터를 이용한 증명 방법으로 증명하였다. 이와 함께, 이전 프로토콜과 제안 프로토콜을 실제로 구현하여 각 프로토콜의 수행시간의 거의 모든 부분을 모듈라 멱승 연산이 차지한다는 것을 보여주었다.

6. Reference

[1] S.Jarecki, " Efficient Threshold Cryptography" , Ph.D. Thesis in MIT, 2001

[2] A.Shamir, " How to Share a Secret" , vol. 22, pp.612-613, Comm. Of the ACM, 1979

[3] B.Chor, S.Goldwasser외 2명" Verifiable secret sharing and achieving simultaneous broadcast" ,pp. 335-344, Proc. Of IEEE Fund. Of Comp. Sci., 1985.

[4] P.Feldman, " A Practical Scheme for Non-interactive Verifiable Secret Sharing" , pp.427-437, Proc. 28th IEEE Symp. On Foundations of Computer Science, L.A.,1987

[5] T.Pederson," A threshold cryptosystem without a trusted third party" ,pp. 522-526, Eurocrypt' 91-LNCS,1991

[6]H.K.A. Herzberg, S.Jarecki , M.Yung," Proactive Secret Sharing Or : How to Cope With Perpetual Leakage," CRYPTO' 95-LNCS,1985

[7] R. Canneti, " Adaptive Security for Threshold Cryptosystems" , pp.98-116, Crypto' 99-LNCS,1999

[8] S. Jarecki A.Lysyanskaya, " Adaptively secure cryptography:Introducing concurrency, removing erasures." , Eurocrypt' 2000-LNCS,2000

[9] R. Ostrovsky and M.Yung, " How to withstand mobile virus attacks" , pp 51-61 In Proc. 10th ACM Symp. On Principles of Distributed Computation,1991

[10] 이윤호, " 효율적인 임계 암호시스템 구현을 위한 능동적 비밀 분산에서의 빠른 공유 갱신에 관한 연구" , M.S. Thesis in KAIST, 2002