공개키 기반 구조에서의 키 복구 지원 메커니즘

이용호° 이임영
순천향대학교 정보기술공학부
abysskey@catholic.or.kr° imylee@sch.ac.kr

Key Recovery Mechanism in Public Key Infrastructure

Yong-Ho Lee° Im-Yeong Lee
Division of Information Technology Engineering Soonchunhyang Univ.

요 약

암호 사용이 증가하는 현 시점에서 키 복구 기능은 PKI(Public Key Infrastructure)의 부가 서비스로써 제공되어야 한다. 그러나 현재 제안된 키 복구 시스템들은 PKI에서 이용하기에는 부적절한 면을 많이 가지고 있다. 본 논문에서는 공개키 기반 구조에서 인증서를 발행하는 일련의 순서를 따라 진행하면서 키 복구를 지원하는 시스템을 제안한다. 제안된 시스템에서 사용자의 공개키는 인증기관에 의해서 생성되고, 사용자의 비밀키는 자신이 안전하게 생성하게 된다. 인증기관은 사용자의 비밀키를 모르면서 사용자의 인증서를 발행하게 된다.

1. 서 론

정보보호의 중요성이 인식되면서 암호의 사용이 빠르게 성장하고 있다. 이러한 암호의 사용이 증가함에 따라 이에 대한 역기능 또한 증가하고 있다. 예로써 암호화에 사용된 키를 분실하거나 자연 재해가 발생하여 키가 없어졌다면 암호화된 문서를 회수하지 못하는 경우가 많이 발생하게 될 것이다. 또한 국가적 차원에서 범죄 수사에 대한 법집행 능력이 저하될 것이다. 현재 이러한 문제점을 해결할 수 있는 가장 좋은 대안으로서 키 복구 시스템이 있다. 키 복구 방식은 사용자의 정보를 최대한 보호하면서 국가의 적법한 절차에 의한 법집행 권한을 유지해 주어야 한다. 따라서 다음과 같은 요구사항을 기본적으로 만족하도록 구성되어야 한다.[1]

- 데이터에 대한 무결성과 기밀성을 유지해야 한다.
- 사용자들의 많은 참여를 유도하기 위해 상세한 부분에 대해 공개적으로 검증 가능해야 한다.
- 범죄자들의 암호화 통신시 적법한 절차를 거치면 암호문을 복호화 할 수 있어야 한다.
- 시스템을 구성하는 참여 개체들의 오용이나 담합과 같은 문제점에 대한 방지책이 있어야 한다.
- 사용자의 키를 복구할 경우, 시스템을 재 초기화하는 비효율성을 제거하기 위해 시간 제한에 대한 보장이 필요하다.

본 논문의 구성은 2장에서 키 복구 시스템과 공개키 기반 구조에 대해 알아보고, 3장에서는 인증서 발행 과정에서 키 복구 기능을 지원하는 새로운 메커니즘을 제안한다. 마지막으로 4장에서 결론을 맺도록 한다.

2. 기존 기술 분석

2.1 키 복구 시스템

키 복구란 기본적인 암호에서 제공하는 방법 이외에 키에 관련된 또 다른 접근 방법을 제공해줌으로써 암호문의 생성자와 수신자 이외에 다른 주체가 유사시에 키를 복구할 수 있는 능력을 주는 것이다. 또한 항상 사용자의 프라이버시 보호와 정부의 법 집행 능력 보장이라는 두 가지 상반된 목적을 만족 시켜야 한다. 즉, 복구 능력을 갖는 주체라도 합법적인 절차를 따르지 않는다면 복구가 불가능하도록 구성되어 한다.[3]

2.2 공개키 기반 구조

공개키 기반 구조(PKI;Public Key Infrastructure)는 공개키 암호 방식을 사용하는 암호시스템에서 사용자의 공개키를 안전하고 신뢰성 있게 공표하는 수단을 제공하고 있다. 정보보호의 중요성이 인식되면서 암호의 사용이 급격히 증가하게 되었다. 따라서 기반 구조는 현재 없어서는 안 되는 중요한 역할을 차지하고 있다.

공개키 기반 구조의 구성 객체는 다음과 같다.[2]

- 사용자(User) : 인증서를 발행 받아 공개키 기반 구조를 이용하려는 객체
- 등록기관(RA; Registration Authority) : 인증서를 요청한 사용자를 확인하고 등록하는 객체
- 인증기관(CA; Certification Authority) : 사용자의 공개키 인증서를 생성/보관/발급하는 기능을 제공하는 객체
- 디렉토리 서버(DS; Directory Server) : 사용자의 인증서 등을 안전하게 보관하고, 각 구성 객체들에게 인증서를 배포하는 기능을 제공하는 객체

공개키 기반 구조에서 인증서(Certificate)를 발급 받기 위한 과정은 일반적으로 다음과 같다.

(1) 사용자는 공개키 쌍(공개키와 비밀키)을 생성한다.
(2) 인증기관에게 자신의 공개키와 공개키에 대응하는 비밀키 소유 확인 정보를 포함한 인증 요구 메시지를 전송한다.
(3) 인증기관은 인증 요구를 확인한다.
(4) 인증기관은 사용자의 공개키에 대한 인증서를 발행한다.
(5) 인증서를 발행한 후 사용자는 인증서를 수신하여 사용하게 된다.

3. 제안방식

본 장에서는 사용자가 인증서를 발행 받는 과정에서 키 복구를 지원하는 새로운 메커니즘을 제안한다. 이 후 모든 수식에서 법(mod) 연산의 표기는 생략하고 진행하며, 표기되는 Alice와 Bob은 사용자를 의미한다.

3.1 시스템 계수

- $\#$: A(Alice; 사용자), B(Bob; 사용자), EA_j(Escrow Agent j; j번째 위탁기관), RA(Registration Authority; 등록기관), CA(Certification Authority; 인증기관), LEA(Law Enforcement Agency; 법집행기관)
- $*$: $\#$의 개인키 또는 공개키, D_A
- m : 사용자들에게 제공되는 부분키 쌍의 개수
- *Part_key_list* : CA에서 생성하여 사용자에게 제공하는 부분키 쌍 리스트
- *Sig_list* : CA의 서명이 붙은 K_i들의 리스트
- $Sig_*( )$: $*$를 키로 사용해 생성한 서명문
- $E_*( )$: $*$를 키로 사용해 생성한 암호문
- $KE_{SK}( )$: 대칭키 SK로 암호화한 암호문
- k_i, t_*, d_* : $*$의 개인키를 구성하는 비밀 난수
- ID_* : $*$의 식별자
- (Y_*, X_*) : $*$의 공개키 쌍, (공개키, 개인키)
- $Cert(*)$: $*$의 인증서
- R_1, R_2, R_3 : Alice와 Bob간의 세션키 설정에 사용된 난수
- *KR_factor* : 암호문과 함께 전송되는 인자로써 수신자의 메시지 복호와 법집행기관의 세션키 복구에 사용된다.
- SK : Alice와 Bob간의 통신에 사용되는 세션키
- SK_info_1, SK_info_2 : Alice와 Bob간의 세션키 SK를 복구하기 위해 사용되는 정보

3.2 프로토콜

본 프로토콜은 초기 설정 단계와 인증서 신청 및 생성 단계 그리고 암호화 통신 및 데이터 복구 단계로 구성되고 각각의 단계는 다음과 같다. 여기서 CA, RA 그리고 EA들은 이미 공개키 쌍을 생성하였으며, 각각의 인증서는 CA의 디렉토리 서버에 안전하게 등록되어 있다고 가정한다.

3.2.1 초기 설정 단계

이 단계는 CA와 RA가 수행하는 단계로써 사용자들의 부분 개인키를 생성하여 공표하는 단계이다.

(1) CA는 m개의 부분키쌍 리스트를 생성한다.

$$Part\_key\_list = (k_i, K_i)$$
$$K_i = g^{k_i} \quad (단,\ 1 \le i \le m) \qquad (식\text{-}1)$$

(2) CA는 부분키쌍 중 K_i를 서명하여 RA에게 전송한다.

$$Sig\_list = Sig_{X\_CA}(K_i) \quad (단,\ 1 \le i \le m) \qquad (식\text{-}2)$$

(3) RA는 CA에서 전송된 *Sig_list*를 안전하게 공표한다.

3.2.2 인증서 신청 및 생성 단계

이 단계는 Alice, RA, EA_i 그리고 CA가 수행하는 단계이다. 최종적으로 Alice는 자신의 공개키쌍을 생성하게 되고, CA는 Alice의 인증서를 발행하게 된다. 이러한 진행 과정 중에서 Alice는 EA_i에게 자신의 부분 개인키 정보를 위탁하게 되고, 이를 통하여 적법한 절차가 수행되면 Alice의 키를 복구할 수 있는 키 복구 기능이 추가된다.

(4) Alice는 두 개의 난수 t_A와 d_A를 생성한다. 그 후 아래와 같이 t_A를 n개로 나누고, T_A와 D_A를 계산한다.

$$t_A = t_{A1} + t_{A2} + \cdots + t_{An}$$
$$T_{Aj} = g^{t_{Aj}} \quad (단,\ 1 \le j \le n)$$
$$T_A = \prod_{j=1}^{n} T_{Aj}, \quad D_A = g^{d_A} \qquad (식\text{-}3)$$

(5) Alice는 RA가 공표한 *Sig_list* 중 하나를 획득하고, n개의 EA와 CA 그리고 RA의 공개키를 이용하여 *(식-4)*를 계산하여 RA에게 전송한다.

$$E_{Y\_RA}(ID_A||T_A||T_{A1}||\cdots||T_{An}||K_i)||E_{Y\_CA}(D_A) \qquad (식\text{-}4)$$

(6) RA는 전송된 값 중 $E_{Y_RA}(ID_A||T_A||T_{A1}||\cdots||T_{An}||K_i)$를 복호화하여 다음을 검증한다.

$$T_A \stackrel{?}{=} \prod_{j=1}^{n} T_{Aj} \qquad (식\text{-}5)$$

(7) RA는 검증이 성공하면 T_{Aj}를 EA_j의 공개키로 암호화하여 각 EA_j에게 전송하고, 다음 값을 계산하여 CA에게 전송한다.

$$E_{Y\_CA}(T_A||K_i)||E_{Y\_CA}(D_A) \qquad (식\text{-}6)$$

(8) 각 EA_j는 전송된 값들을 안전하게 저장하고, CA는 Alice가 선택한 K_i에 대응하는 k_i를 검색한다.

(9) CA는 전송된 D_A와 T_A 그리고 검색된 k_i를 이용하여 *(식-7)*과 같이 Alice의 최종 공개키 Y_A를 생성하고, 인증서 $Cert(A)$를 발행한다.

$$Y\_A = T_A * D_A^{k_i} \qquad (식\text{-}7)$$

(10) CA는 D_A를 키로 이용하여 k_i를 암호화하고, Alice의 인증서와 함께 RA를 통하여 Alice에게 전송한다.

$$Cert(A)||E_{D_A}(k_i) \qquad (식\text{-}8)$$

(11) Alice는 전송된 $E_{D_A}(k_i)$를 복호화 한다. 그리고 k_i와 d_A 그리고 t_A를 이용하여 *(식-9)*와 같이 자신의 최종 개인키 X_A를 생성한다.

$$X\_A = t_A + d_A * k_i \qquad (식\text{-}9)$$

(12) Alice는 생성된 개인키를 이용하여 *(식-10)*과 같이 계산하여 $Cert(A)$에 있는 공개키 Y_A와 비교한다. 만약 같으면 Y_A를 공개키로 인정한다.

$$Y\_A = g^{X\_A} \qquad (식\text{-}10)$$

이러한 과정을 통하여 Alice는 안전하게 공개키쌍 *(Y_A,*

X_A)를 생성하게 되고, 공개키 인증서를 발급 받게 된다.

3.2.3 암호화 통신 단계

여기서는 상기 시스템을 통하여 인증서를 발행 받은 Alice와 Bob간의 암호화 통신 단계를 기술한다. Bob의 공개키쌍과 인증서 생성 단계는 Alice와 동일하다고 가정한다. Alice와 Bob은 암호화 통신을 위해서 세션키를 공유하고, 대칭키 암호 알고리즘을 이용하여 암호화 통신을 수행한다.

(13) Bob은 난수 R_1을 생성하고, Alice에게 암호 통신 요구 메시지를 전송한다.

$$Cert(B)||E_{Y\_A}(R_1) \qquad (식-11)$$

(14) Alice는 난수 R_2를 생성하여 R_3을 계산하고, Bob 의 공개키를 이용하여 *(식-12)*와 같이 암호문을 구성한다. 이것을 Bob에게 전송한다.

$$R_3 = R_1 \oplus R_2$$
$$E_{Y\_B}(Cert(A)||R_2||R_3||K_i||T_A) \qquad (식-12)$$

(15) Bob은 R_2와 R_3을 XOR 연산한 결과와 처음 생성한 R_1을 비교한다. 만약 같으면 $Cert(A)$와 K_i 그리고 T_A를 이용하여 KR_factor를 계산한다.

$$R_1 \overset{?}{=} R_2 \oplus R_3$$
$$KR\_factor = g^{k_i(1-d_A)} \qquad (식-13)$$

KR_factor는 다음과 같이 계산된다.

$$\begin{aligned} KR\_factor &= K_i * (Y_A * T_A^{-1})^{-1} \\ &= g^{k_i} * (g^{t_A - t_A})^{-1} \\ &= g^{k_i} * (g^{d_A * k_i})^{-1} \\ &= g^{k_i(1-d_A)} \end{aligned}$$

(16) Bob은 대칭키 암호화에 사용되는 세션키 SK를 랜덤하게 생성한다. *(식-14)*와 같이 구성하여 Alice에게 전송한다.

$$KE_{SK}(M)||SK * KR\_factor \qquad (식-14)$$

(17) Alice는 K_i와 d_A를 이용하여 SK를 획득하고, 메시지를 복호화한다.

3.2.4 암호 세션키 복구 단계

이 단계에서는 Alice와 Bob간의 암호화 통신시에 사용된 세션키를 복구하는 단계이다. Alice의 세션키 복구는 법원의 허가서를 통해 EA들과 CA의 도움을 받아 LEA에 의해서 이루어진다. 여기서 LEA는 정당한 과정을 통하여 법원의 허가서를 받았다고 가정한다.

(18) 법원의 허가서를 받은 LEA는 전송되는 암호 데이터를 획득한 후 Alice의 공개키를 이용하여 *(식-15)*와 같이 SK_info_1을 계산한다.

$$SK\_info_1 = SK * g^{t_A + k_i} \qquad (식-15)$$

SK_info_1은 다음과 같이 계산된다.

$$\begin{aligned} SK\_info_1 &= SK * KR\_factor * Y_B \\ &= SK * g^{k_i(1-d_A)} * g^{t_A + d_A * k_i} \\ &= SK * g^{t_A + k_i} \end{aligned}$$

(19) LEA는 EA들과 CA에게 법원의 허가서를 제출하고 Alice의 위탁 정보를 요청한다. EA_j와 CA는 각각 T_A와 K_i를 LEA에게 안전하게 전송한다. LEA는 전송된 값을 *(식-16)*과 같이 계산하여 SK_info_2를 계산한다.

$$SK\_info_2 = T_A * K_i \qquad (식-16)$$

(20) LEA는 SK_info_1과 SK_info_2를 *(식-17)*과 같이 계산하여 세션키 SK를 획득하고, 이것을 이용하여 Bob이 Alice에게 전송한 암호 데이터를 복호한다.

$$SK = SK\_info_1 \,/\, SK\_info_2 \qquad (식-17)$$

3.3 비교분석

제안된 방식은 PKI하에서 인증서를 발행 받는 과정을 그대로 유지하면서 키 복구를 지원할 수 있도록 구성되었다. 사용자의 개인키는 3개의 비밀 정보로 구성되고, 이 중 2개는 각각 CA와 EA에게 알려지게 된다. 그러나 나머지 1개의 비밀 정보는 사용자만이 알고 있으므로, 사용자는 자신의 개인키를 유일하게 소유하게 된다. CA는 사용자의 공개키를 직접 생성하고 이에 대한 인증서를 발행하므로 사용자의 부정에 대한 소지를 제거할 수 있다.

인증서를 발행 받은 후 사용자간에 암호화 통신을 할 경우 전송되는 메시지에 대한 복구가 가능하다. 특정 조건을 만족하는 기관 (즉, 법원의 허가를 받은 법집행기관)은 인증기관과 위탁기관들의 협조를 받아 사용자의 암호문에 사용된 키를 복구할 수 있다. 여기서 사용자가 생성한 세션키는 사용기간을 임의대로 조정할 수 있으므로 한 번 복구 되더라고 시스템을 재 초기화하기 않아도 된다는 특징을 가지고 있다.

4. 결 론

우리는 키 복구 기능의 중요성과 현재 널리 사용되고 있는 PKI에 대해 알아보았고, PKI에서 키 복구 기능의 중요성에 대하여 설명하였다. 본 논문에서는 기존의 인증서 발급 과정을 진행하면서 최소의 변경으로 키 복구를 지원하는 시스템을 제안하였다. 이렇게 키 복구 기능을 PKI에 적용함으로써 효율성을 극대화하고, 신뢰할 수 있는 정보보호 서비스를 제공할 수 있다. 향후 좀더 효율적이고 안전한 시스템의 연구가 진행되어야 할 것이다.

참고문헌

[1] J. Kilian and T. Leighton, Fair Cryptosystems, Revisited, CRYPTO 95, 1995

[2] J. Menezes, C. Oorschot, A. Vanstone, HANDBOOK of APPLIED CRYPTOGRAPHY, CRC Press LLC, 1997

[3] M. Bellare and S. Goldwasser, Verifiable Partial Key Escrow, Annual Conference on Computer and Communications Security, ACM, 1996