공개키 암호를 이용한 bulk 데이터 암호화

신상욱†, 이경현‡
†한국전자통신연구원
‡부경대학교 컴퓨터멀티미디어공학전공

Bulk data encryption using a public key cryptography

Sang Ůk Shin†, Kyung Hyune Rhee‡
†Electronics and Telecommunications Research Institute
‡Department of Computer & Multimedia Engineering, PKNU

요 약

본 논문에서는 키 교환 단계없이 비대칭키 암호 알고리즘을 사용하여 대량의 메시지를 암호화하여 전송하는 기법을 제안한다. 제안된 기법은 전체 메시지를 스크램블링한 후 스크램블링된 메시지의 일부분만을 공개키 암호 알고리즘을 사용하여 암호화하여 전송한다. 스크램블링 함수로 신상욱[3] 등에 의해 제안된 해쉬함수를 사용한 all-or-nothing 변환을 이용한다. 그리고 제안된 기법에 약간의 추가적인 오버헤드를 부가하여 디지털 서명까지 제공하는 기법을 제안한다.

1. 서론

1990년대 이후 월드와이드웹과 전자상거래와 같은 새로운 기술이 개발된 이후 인터넷 사용자는 폭발적인 성장을 하고 있다. 인터넷과 같은 개방된 네트워크를 통해 전달되는 데이터는 항상 제3자에 의해 도청, 위조, 변경될 위험이 있기 때문에 정보보호 서비스의 필요성 역시 크게 증가하고 있다. 허가되지 않은 제3자에 의해 인터넷을 통해 전달되는 데이터에 대한 접근을 방지하기 위해 기밀성 서비스가 제공되어야 한다. 이러한 기밀성 서비스를 제공하기 위한 기본 암호 프리미티브로 대칭키 암호 알고리즘과 비대칭키 암호 알고리즘을 주로 사용한다.

대칭키 암호는 송수신자만이 알고 있는 비밀키를 사용하여 메시지를 암호화하여 전달하고 수신자는 역시 송신자와 미리 공유한 비밀키를 사용하여 메시지를 복호화한다. 이러한 대칭키 암호는 블록 암호와 스트림 암호로 분류되고, 가장 널리 사용되고 있는 DES 그리고 DES를 대신할 새로운 알고리즘 개발을 위한 AES에 제출되어 최종 선택된 5개의 후보 알고리즘인 RC6, Twofish, Serpent, Mars, Rijndael 등이 있다. 공개키 암호는 트랩도어 일방향 함수를 사용하여 암호화를 위해 사용할 공개키와 복호화를 위해 사용할 비밀키 쌍을 생성하여 공개키는 공개 디렉토리 등에 공개시켜 두고 비밀키는 타인이 알지 못하게 비밀로 유지한다. 메시지 전달을 위해 송신자는 수신자의 공개키로 메시지를 암호화하여 전송하면 수신자는 자신의 비밀키를 이용하여 메시지를 복호화한다. 대표적인 공개키 암호로 RSA와 ElGamal 공개키 암호가 있다.

대칭키 암호 알고리즘과 비대칭키 암호 알고리즘은 서로 상호 보완적인 장단점을 가진다. 대칭키 암호 알고리즘은 공개키 암호 알고리즘에 비해 매우 빠른 성능을 보이고 상대적으로 적은 키 길이를 가진다. 하지만 대칭키 암호 알고리즘은 키 교환과 키 관리의 문제가 발생한다. 비대칭키 암호 알고리즘은 대칭키 암호 알고리즘에 비해 키 교환과 키 관리에

있어 이점을 가지고 전자 서명 등에 쉽게 적용할 수 있다. 하지만, 비대칭키 암호 알고리즘은 대칭키 암호 알고리즘에 비해 성능이 매우 낮다는 단점을 가진다. 이와 같은 장단점으로 인해 일반적으로 대칭키 알고리즘과 비대칭키 알고리즘을 결합하여 사용한다. 먼저 비대칭키 알고리즘을 이용하여 사용할 비밀 세션키를 공유하고 그 후에 비밀 세션키를 대칭키 암호에 적용하여 메시지를 암호화하여 전송한다[1].

위에서 기술한 것처럼 하이브리드 기법은 메시지 기밀성을 제공하기 위해 비대칭키 암호를 통한 키 교환 후에 대칭키 암호를 사용하여 메시지를 암호화하여 전송하므로 키 교환 과정이 항상 필요하게 된다. 또한 구현시에 대칭키 암호 프리미티브와 비대칭키 암호 프리미티브를 모두 구현해야 한다. 따라서 본 논문에서는 키 교환 단계없이 비대칭키 암호 알고리즘을 사용하여 대량의 메시지를 암호화하여 전송하는 기법을 제안한다. 제안된 기법은 전체 메시지를 스크램블링한 후 스크램블링된 메시지의 일부분만을 공개키 암호 알고리즘을 사용하여 암호화하여 전송한다. 스크램블링 함수로 신상욱 등에 의해 제안된 해쉬함수를 사용한 all-or-nothing 변환을 이용한다[3].

2장에서는 all-or-nothing 성질을 가지는 해쉬함수를 사용하여 키 교환이 필요 없는 공개키 암호를 이용한 bulk 데이터 암호화를 제안한다. 그리고 3장은 결론이다.

2. Bulk 데이터 암호 알고리즘

인터넷과 같은 공개 네트워크를 통해 중요한 메시지를 전달하는 경우 흔히 암호화 알고리즘을 사용하여 메시지의 기밀성을 보장하게 된다. 1장에서 설명된 대칭키와 비대칭키 암호 알고리즘의 장,단점에 따라 현재 가장 많이 사용되는 메시지의 기밀성 보장 방안은 이들 두 알고리즘을 결합한 하이브리드 기법을 사용하는 것이다. 이 경우, 메시지 기밀성 제공에 사용되는 세션키는 비대칭키 암호 알고리즘을 사용하여 교환이 이루어지므로 세션 설정마다 키 교환의 절차가 필요하게 된다. 또한 하이브리드 기법은 시스템 구현시 대칭키 암호 알고리즘과 비대칭키 암호 알고리즘을 모두 구현해야 하므로 시스템 구성이 복잡해지는 단점이 존재한다. 이 장에서는 해쉬함수와 공개키 암호 알고리즘을 이용하여 세션키 교환 절차 없이 대량의 데이터에 기밀성을 제공하는 효율적인 bulk 데이터 암호 알고리즘을 제안한다. 제안된 기법은 약간의 추가적인 오버헤드를 부가하면 디지털 서명까지 제공해 줄 수 있다.

2.1 공개키 암호를 이용한 bulk 데이터 암호 알고리즘

본 절에서는 키 교환이 필요 없이 비대칭키 암호 알고리즘을 이용하여 대량의 데이터를 암호화하여 전달하는 기법을 제안한다. 제안된 기법은 Rivest[2]에 의해 제안된 all-or-nothing 변환을 이용하여 메시지를 의사 메시지로 변환한 후 변환된 의사 메시지의 일부분만을 공개키 암호를 사용하여 암호화한다. 제안된 기법은 해쉬함수와 공개키 암호 알고리즘만을 이용하고 적은 오버헤드를 가지고 메시지 기밀성과 무결성을 제공할 수 있으며, 또한 약간의 추가적인 오버헤드를 부가하여 전자서명까지 제공할 수 있다. 구체적인 알고리즘의 동작은 다음과 같다.

<기법 1>

A. 전송측

(A-1) 입력 메시지 X의 해쉬값 계산 $K = h_{IV}(X)$

(A-2) 입력 메시지 X를 s개의 n비트 블록, $(X_1, X_2, \cdots, X_s)$으로 분할(n은 해쉬함수의 출력 비트수).

(A-3) all-or-nothing 변환을 통해 의사 메시지 $Y = (Y_1, \cdots, Y_s)$을 계산.

$Y_0 = IV,\ X_0 = K$

$Y_i = X_i \oplus h_{X_{i-1}}(Y_{i-1} \| (K \oplus i)),\ i = 1, \cdots, s$

(A-4) 마지막 의사 메시지 블록 Y_{s+1} 계산(n비트). 여기서 K_b는 자신의 공개키이다. 예로, 1024비트 공개키를 사용하는 경우 n 비트로 분할한 후 이들을 xor하여 사용한다.

$MD = h_{K_b}(Y_1 \| \cdots \| Y_s \| h_{IV}(K_b \oplus (s+1))),$

$Y_{s+1} = K \oplus MD$

(A-5) $(Y_{s+1} \| h_{MD}(Y_{s+1}))$을 수신자의 공개키 R_b로 암호화.

$$C = AE_{R_b}(Y_{s+1} \| h_{MD}(Y_{s+1}))$$

여기서, AE는 RSA와 같은 비대칭키 암호 알고리즘이다.

(A-6) $(Y(= Y_1, \cdots, Y_s) \| C)$를 전송.

B. 수신측

(B-1) $(Y \| C)$을 수신.

(B-2) 자신의 비밀키 R_s로 C를 복호화, $(LB \| Z) = AE_{R_s}(C)$

(B-3) 메시지 Y을 s 개의 n비트 블록, $Y_1, Y_2, \cdots, Y_s$로 분할.

(B-4) K 복원. 여기서 K_p는 송신자의 공개키를 n 비트로 분할한 후 이들을 xor한 값이다.

$MD' = h_K(Y_1 \| \cdots \| Y_s \| h_{IV}(K_p \oplus (s+1)))$,

$K = LB \oplus MD$

(B-5) $Z = h_{MD'}(LB)$ 인지 확인.

(B-6) 원래 메시지 $X = (X_1, X_2, \cdots, X_s)$ 복구.

$Y_0 = IV$, $X_0 = K$

$X_i = Y_i \oplus h_{X_{i-1}}(Y_{i-1} \| (K \oplus i))$, $i = 1, \cdots, s$

(B-7) $K = h_{IV}(X_1 \cdots X_s)$ 인지 확인.

제안된 기법은 해쉬함수와 RSA와 같은 비대칭키 암호 알고리즘을 사용하여 구성된다. 입력 메시지는 먼저 해쉬함수를 사용한 all-or-nothing 변환을 통해 의사 메시지로 변환되고, 변환된 의사 메시지 중에서 마지막 두 블록만을 비대칭키 암호 알고리즘을 사용하여 암호화한다. all-or-nothing 변환 자체는 비밀키 요소를 가지지 않는 공개된 변환이므로 송수신자 사이에 비밀 세션키 교환 과정이 필요없게 된다.

제안된 기법의 안전성을 고려해보면, 공격자는 수신자의 비밀키를 알지 못하기 때문에 전달되는 메시지 $(Y \| C)$ 중에서 C를 복호화할 수 없다. C를 복호화하지 못하면, $(Y_{s+1} \| Z)$를 알지 못하게 되고, 결국 K를 알지 못한다. 그러므로 제안된 기법은 사용된 비대칭키 암호 알고리즘의 안전성에 의존한다.

K를 알지 못하는 공격자가 C를 복호화하지 않고 평문을 알기 위해서는 다음을 계산해야 한다.

$$X_i = Y_i \oplus h_{X_{i-1}}(Y_{i-1} \| (K \oplus i))$$

이것을 풀기 위해서는 공격자는 $h_{X_{i-1}}(Y_{i-1} \| (K \oplus i))$ 값을 추측하거나 K 값을 추측해야 한다. 이 경우 K는 평문의 해쉬값이고 원 평문을 알지 못하기 때문에, 공격자는 단지 K를 추측할 수 밖에 없다. 의사 난수 성질을 만족하는 해쉬함수를 사용한다면, 공격이 성공할 확률은 2^{-n}이다. 160비트 출력을 가지는 해쉬함수를 사용한다면, 공격이 성공할 확률은 2^{-160}이다.

일반적으로 메커니즘의 안전성을 높이기 위해서는 빈번한 키 변경이 요구된다. 즉, 같은 키를 여러 번 반복해서 사용하지 않고 세션마다(또는 메시지마다) 비밀키를 새롭게 설정하게 된다. 이런 점을 고려해보면, 제안된 기법은 평문의 해쉬값이 비밀키로 작용하기 때문에 키는 메시지마다 한 번만 사용되게 된다. 따라서 공격자는 새로운 메시지가 전달될 때마다 K를 추측하는 작업을 반복해야 한다. 이것은 송수신측이 각 메시지를 교환할 때마다 사용할 비밀키를 교환하는 것과 비교하면 상당한 통신 오버헤드를 절약하게 된다. 또한 보통의 대칭키 암호 알고리즘이 메시지의 무결성을 제공하지 않지만 제안된 기법은 (B-5) 단계와 (B-7) 단계에서 수신된 메시지에 대한 무결성을 검사한다.

제안된 기법 1은 약 13.21Mbyte/sec의 성능을 보인다. RC6 블록 암호 알고리즘을 이용한 경우에는 약 8.7Mbyte/sec의 성능을 보이고, RSA 공개키 암호 알고리즘를 이용한 경우에는 약 0.68Mbyte/sec의 성능을 보인다. 이와 같은 결과는 해쉬함수의 특징 중의 하나가 빠른 성능을 가진다는 것을 보면 제안된 기법이 성능면에서 이점을 가지는 것을 알 수 있다. 더욱이 블록 암호 알고리즘을 이용한 경우에는 추가적으로 공개키 암호 알고리즘을 이용한 세션키 교환 절차가 필요하기 때문에 이를 처리하기 위한 성능 저하가 추가적으로 발생한다.

2.2 공개키 암호를 이용한 bulk 데이터의 암호화와 디지털 서명

2.1절에서 제안된 기법 1에 약간의 오버헤드를 부가함으로써 디지털 서명 서비스를 제공할 수 있다. 일반적으로 메시지에 기밀성과 디지털 서명을 함께 제공하려면, 먼저 메시지의 해쉬값을 취한 후 이를 공개키 암호를 사용하여 디지털 서명하여 메시지에 추가한 후 (메시지, 디지털 서명)을 대칭키 암호 알고리즘에 적용하여 암호화한다. 디지털 서명을 추가한 제안된 기법은 다음처럼 동작한다.

<기법 2>

A. 전송측

(A-1) 입력 메시지 X의 해쉬값 계산 $K = h_{IV}(X)$

(A-2) 입력 메시지 X를 s개의 n비트 블록, $(X_1, X_2, \cdots, X_s)$으로 분할.

(A-3) all-or-nothing 변환을 통해 의사 메시지 $Y = (Y_1, \cdots, Y_s)$을 계산.

$Y_0 = IV$, $X_0 = K$

$Y_i = X_i \oplus h_{X_{i-1}}(Y_{i-1} \| (K \oplus i))$, $i = 1, \cdots, s$

(A-4) 마지막 의사 메시지 블록 Y_{s+1} 계산(n 비트). 여기서 K_p는 자신의 공개키이다. 예로, 1024비트 공개키를 사용하는 경우 n 비트로 분할한 후 이들을 xor하여 사용한다.

$$MD = h_{K_p}(Y_1 \| \cdots \| Y_s \| h_{IV}(K_p \oplus (s+1))),$$

$$Y_{s+1} = K \oplus MD$$

(A-5) $(Y_{s+1} \| h_{MD}(Y_{s+1}))$을 자신의 비밀키 S_s로 서명한 후 수신자의 공개키 R_p로 암호화.

$$C = AE_{R_p}(AE_{S_s}(Y_{s+1} \| h_{MD}(Y_{s+1})))$$

(A-6) $(Y(= Y_1, \cdots, Y_s) \| C)$를 전송.

B. 수신측

(B-1) $(Y \| C)$을 수신.

(B-2) 자신의 비밀키 R_s로 C를 복호화,

$$(DS) = AE_{R_s}(C)$$

(B-3) DS를 송신자의 공개키 S_p로 복호,

$$(LB \| Z) = AE_{S_p}(DS)$$

(B-4) 메시지 Y을 s 개의 n비트 블록, $Y_1, Y_2, \cdots, Y_s$로 분할.

(B-5) K 복원. 여기서 K_p는 송신자의 공개키를 n 비트로 분할한 후 이들을 xor한 값이다.

$$MD' = h_{K_p}(Y_1 \| \cdots \| Y_s \| h_{IV}(K_p \oplus (s+1))),$$

$$K = LB \oplus MD$$

(B-6) $Z = h_{MD'}(LB)$ 인지 확인(디지털 서명 검증).

(B-7) 원래 메시지 $X = (X_1, X_2, \cdots, X_s)$ 복구.

$$Y_0 = IV,\ X_0 = K$$

$$X_i = Y_i \oplus h_{X_{i-1}}(Y_{i-1} \| (K \oplus i)),\ i = 1, \cdots, s$$

(B-8) $K = h_{IV}(X_1 \cdots X_s)$ 인지 확인.

디지털 서명이 추가된 기법 2는 기법 1과 비교할 때 (A-5) 단계에서 송신측의 비밀키로 서명하는 부분과 (B-3) 단계에서 송신측의 공개키로 복호하는 과정이 추가되어 계산량에서 비대칭키 암호 알고리즘의 적용이 2번 더 추가되었다. 하지만 이것은 단지 한 블록에 대한 디지털 서명과 검증 과정이기 때문에 송수신자에게 많은 계산적인 오버헤드를 부가하지 않는다. 이 기법 역시 해쉬함수와 공개키 암호 알고리즘만으로 메시지의 기밀성과 디지털 서명 서비스를 제공해 줄 수 있다.

3. 결론

본 논문에서는 키 교환 단계없이 비대칭키 암호 알고리즘을 사용하여 대량의 메시지를 암호화하여 전송하는 기법을 제안하였다. 제안된 기법은 전체 메시지를 스크램블링한 후 스크램블링된 메시지의 일부분만을 공개키 암호 알고리즘을 사용하여 암호화하여 전송한다. 스크램블링 함수로 신상욱 [3] 등에 의해 제안된 해쉬함수를 사용한 all-or-nothing 변환을 이용하였다. 제안된 기법은 평문의 해쉬값이 비밀키로 작용하기 때문에 키는 메시지마다 한 번만 사용되게 된다. 따라서 공격자는 새로운 메시지가 전달될 때마다 K를 추측하는 작업을 반복해야 한다. 이것은 송수신측이 각 메시지를 교환할 때마다 사용할 비밀키를 교환하는 것과 비교하면 상당한 통신 오버헤드를 절약하게 된다. 또한 제안된 기법에 약간의 추가적인 오버헤드를 부가하여 해쉬함수와 공개키 암호 알고리즘만으로 메시지의 기밀성과 디지털 서명 서비스를 제공해 줄 수 있다.

[참고문헌]

[1] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997

[2] R.L. Rivest, "All-Or-Nothing Encryption and The Package Transform", The Proceedings of the 1997 Fast Software Encryption Conference, LNCS, vol.1267, pp.210-218, 1997

[3] Sang Uk Shin, Kyung Hyune Rhee, Jae Woo Yoon, "Hash Functions and the MAC Using All-Or-Nothing Property", PKC'99(International Workshop on Practice and Theory in Public Key Cryptography), LNCS, vol.1560, pp.263-275, 1999