안전한 암호 통신을 위한 키교환 프로토콜1)

서화정*, 김호원*
*부산대학교 컴퓨터공학과
e-mail:hwajeong@pusan.ac.kr

Authenticated Key Exchange Protocol for the Secure Communication

Hwa-Jeong Seo*, Ho-won Kim*
*Dept of Computer Engineering, Pusan National University

요 약

상호간의 보안 통신을 위해서는 서로간의 안전한 비밀키 교환이 이루어져야 한다. 이를 보장하기 위해서는 안전한 키교환 프로토콜이 사용되어야 한다. 키교환 프로토콜은 안전성을 보장함과 동시에 키의 신선도와 확신에 대한 요구사항을 모두 만족시켜야한다. 현재 대표적인 키교환 프로토콜인 Diffie-Hellman을 기본으로 하는 다양한 프로토콜이 연구 및 개발되고 있다. 최근에 연구된 EKE-E(Encrypted Key Exchange-Efficient) 프로토콜은 Diffie-Hellman 알고리즘을 통한 키교환을 제공하며 man-in-the-middle공격과 오프라인 사전공격에 대한 안정성을 보장한다. 하지만 재전송 공격에 취약성을 가진다. 본 논문에서는 최근에 제안된 키교환 프로토콜인 EKE-E의 안전성을 만족하며 재전송공격에 안전한 프로토콜을 제안한다. 동시에 연산을 줄여 보다 성능을 향상시킨다.

1. 서론

두명의 사용자간의 안전한 통신을 위한 키분배 프로토콜은 서로간의 인증을 통한 안전한 과정을 거쳐 수행되게 된다. 키 교환 문제의 신임도를 보장하기 위해 가장 최초에 제안된 EKE(Encrypted Key Exchange)은 상호간의 인증 및 안전한 키분배가 가능한 알고리즘이다[1][2]. 최근에는 해당 알고리즘을 개선한 EKE-E(Encrypted Key Exchange Efficient)프로토콜이 제안되어 보다 효율적으로 상호간의 인증이 가능하도록 하였다[3]. 하지만 재전송공격에 취약할뿐 아니라 많은 연산이 소요된다. 본논문에서는 재전송공격에 안전하며 보다 효율적인 프로토콜을 제안한다. 논문의 구성은 2장에서 A-EKE-E(Advanced Encrypted Key Exchange Efficient)프로토콜에 대해 설명한다. 3장에서는 제안된 프로토콜의 성능을 분석하며 마지막 4장에서 결론을 내린다.

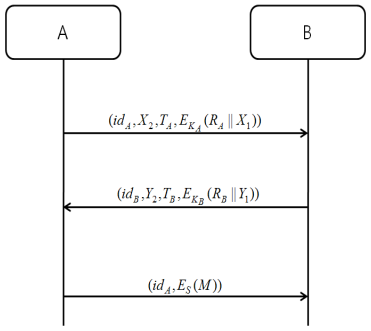
2. A-EKE-E

프로토콜에서 사용되는 용어의 표기법은 <표 1>과 같다.

<표 1> 프로토콜에 사용되는 용어의 표기법

A,B : 시스템 관리자
P : A와 B 상호 간에 미리 분배된 비밀값으로 $(X_1+X_2) \bmod N \equiv P$과 $(Y_1+Y_2) \bmod N \equiv P$를 만족한다.
N,M : 암호화과정에 사용되는 양의 정수
R_A,R_B : A와 B에 의해 생성된 임의의 난수
$H(\bullet)$: 해시함수
T_A,T_B : A와 B에 의해 생성된 타임스탬프
E_K : 비밀키 K를 통한 암호화
D_K : 비밀키 K를 통한 복호화

A-EKE-E 프로토콜은 (그림 1)과 같다. 프로토콜은 두 개의 인자간에 안전한 키교환이 수행될 수 있도록 각각의 인자는 상대방에게 비밀 값을 전달하고 이를 전달받은 상대 인자는 자신의 비밀값과 상대방의 비밀값을 통해 Diffie-Hellman기법을 사용하여 세션키를 안전하게 설립한다. 세션키가 분배되고 난 뒤에는 보내고자하는 메시지를 암호화하여 상대방에게 전송한다.



(그림 1) A-EKE-E 프로토콜

1) "이 논문 또는 저서는 2010년 교육과학기술부로부터 지원받아 수행된 연구임" (지역거점연구단육성사업/차세대물류IT기술연구사업단)

<표 2>는 인증메시지 생성과정에 대한 설명이다. [Step 1]에서 사용자 A는 $(0 \leqq X_1 \leqq N-1)$를 만족하는 인자 X_1를 선택한다. [Step 2]에서는 $X_2 = (P - X_1) \bmod N$ 특성에 따라 X_2를 계산한다. [Step 3]에서는 A의 id 값과 타임스탬프 T_A값 그리고 X_1값을 해시연산을 하여 암호화 과정에서 사용하게 될 킷값을 생성한다. [Step 4]에서는 $(0 \leqq R_A \leqq M-1)$를 만족하는 임의의 난수 R_A를 생성한다. [Step 5]에서는 생성된 난수와 X_1은 비밀값 K_A를 통해 암호화된다. 인증메시지의 작성이 모두 끝나면 [Step 6]에서 A는 B에게 메시지 $(id_A, X_2, T_A, E_{K_A}(R_A \| X_1))$를 보내게 된다.

<표 2> 사용자 A의 인증메시지 생성과정

1. A : X_1를 선택한다.
2. A : $X_2 = (P - X_1) \bmod N$ 을 계산한다.
3. A : id_A와 T_A 그리고 X_1값을 사용하여 $K_A = H(id_A, X_1, T_A)$값을 계산한다.
4. A : 난수 R_A를 생성한다.
5. A : 난수 R_A과 X_1값을 비밀값 K_A로 암호화한 $E_{K_A}(R_A \| X_1)$를 계산한다.
6. A → B : 메시지 $(id_A, X_2, T_A, E_{K_A}(R_A \| X_1))$를 User B에게 전송한다.

<표 3>는 인증메시지 검증 및 인증메시지 생성과정에 대한 설명이다. [Step 1]에서 B는 A로부터 전송받은 메시지의 타임스탬프 T_A값을 확인하여 적합한 시간 안에 도착한 메시지인지 확인한다. 만약 메시지가 시간을 초과한다면 Replay attack임을 확인할수 있다. [Step 2]에서는 전송받은 메시지 X_2를 사용하여 $X_1 = (P - X_2) \bmod N$를 계산할 수 있다. [Step 3]에서는 계산된 X_1과 타임스탬프 T_A 그리고 A의 id_A값을 해시함수의 인자값으로 사용한다. [Step 4]에서는 해시값의 결과로 도출된 K_A값은 전송된 암호문 $E_{K_A}(R_A \| X_1)$의 복호화과정의 킷값으로 사용한다. 복호화가 성공적으로 끝나면 A로부터 생성된 난수 R_A과 X_1을 추출할 수 있다. 추출된 X_1을 통해 A가 정당한 사용자임을 확인한다. [Step 5]에서 사용자 B는 $(0 \leqq Y_1 \leqq N-1)$를 만족하는 인자 Y_1를 선택한다. [Step 6]에서는 $Y_2 = (P - Y_1) \bmod N$ 특성에 따라 Y_2를 계산한다. [Step 7]에서는 B의 id값과 타임스탬프 T_B값 그리고 Y_1값을 해시연산을 하여 암호화 과정에서 사용하게 될 킷값을 생성한다. [Step 8]에서는 $(0 \leqq R_B \leqq M-1)$를 만족하는 임의의 난수 R_B를 생성한다. [Step 9]에서는 생성된 난수와 Y_1는 비밀값 K_B를 통해 암호화된다. 인증메시지의 작성이 모두 끝나면 [Step 10]에서 B는 A에게 메시지 $(id_B, Y_2, T_B, E_{K_B}(R_B \| Y_1))$를 보내게 된다.

<표 3> 사용자 A의 인증메시지 검증 및 사용자 B의 인증메시지 생성

1. B : 전송받은 타임스탬프 T_A가 적합한지 확인한다.
2. B : $X_1 = (P - X_2) \bmod N$ 을 계산한다.
3. B : $K_A = H(id_A, X_1, T)$을 계산한다.
4. B : $D_{K_A}(E_{K_A}(R_A))$을 수행하여 A로부터 전달받은 난수 R_A를 추출한다.
5. B : Y_1를 선택한다.
6. B : $Y_2 = (P - Y_1) \bmod N$ 을 계산한다.
7. B : id_B와 Y_1값 그리고 T_B을 사용하여 $K_B = H(id_B, Y_1, T_B)$ 값을 계산한다.
8. B : 난수 R_B를 생성한다.
9. B : 난수 R_B를 비밀값 K_B로 암호화한 $E_{K_B}(R_B \| Y_1)$를 계산한다.
10. B → A : 메시지 $(id_B, Y_2, T_B, E_{K_B}(R_B \| Y_1))$를 User A에게 전송한다.

<표 4>는 인증메시지 검증 및 세션키를 설립하는 과정이다. [Step 1]에서 A는 B로부터 전송받은 메시지의 타임스탬프 T_B값을 확인하여 적합한 시간 안에 도착한 메시지인지 확인한다. 만약 메시지가 시간을 초과한다면 재전송공격임을 확인할수 있다. [Step 2]에서는 전송받은 메시지 Y_2를 사용하여 $Y_1 = (P - Y_2) \bmod N$를 계산할 수 있다. [Step 3]에서는 계산된 Y_1과 타임스탬프 T_B 그리고 B의 id_B값을 해시함수의 인자값으로 사용한다. 해시값의 결과로 도출된 K_B값은 [Step 4]에서 전송된 암호문 $E_{K_B}(R_B \| Y_1)$의 복호화 과정의 킷값으로 사용된다. 복호화가 성공적으로 끝나면 B로부터 생성된 난수 R_B과 Y_1을 추출할 수 있다. 추출된 Y_1을 통해 B가 정당한 사용자임을 확인한다. [Step 5]에서는 서로간에 교환한 난수값 R_A, R_B는 해시연산을 통해 세션키를 생성하게 된다. 이는 상호간에 비밀정보값을 교환하여 세션키를 생성하는 Diffie-Hellman기법을 적용한 것이다[4].

<표 4> 사용자 B의 인증메시지 검증 및 세션키 설립

1. A : 전송받은 타임스탬프 T_B가 적합한지 확인한다.
2. A : $Y_1 = (P - Y_2) \bmod N$ 을 계산한다.
3. A : $K_B = H(id_B, Y_1, T_B)$을 계산한다.
4. A : $D_{K_B}(E_K(R_B))$을 수행하여 B로부터 전달받은 난수 R_B를 추출한다.

5. A : 서로간에 교환한 난수값을 해시연산하여 세션키 $SK=H(R_A \| R_B)$를 생성한다.

<표 5>는 세션키의 분배이후 안전한 보안통신과정에 대한 설명이다. [Step 1]에서 B와 세션키를 생성한 A는 전송하고자 하는 메시지 M을 세션키 S로 암호화하여 암호문 $E_S(M)$를 생성하게 된다. 생성된 암호문은 [Step 2]에서 자신의 아이디 정보와 함께 B에게 안전하게 전달되며 이를 통해 보안 통신이 가능하다.

<표 5> 세션키를 통한 암호화 이후 안전한 보안통신

1. A : 전송할 메시지 M을 세션키 S로 암호화하여 $E_S(M)$를 생성한다. 2. A → B : A는 B에게 전송하기 위해 암호화한 메시지와 자신의 id를 포함한 메시지 $(id_A, E_S(M))$를 B에게 전송한다.

3. 프로토콜 분석

3.1 성능 분석

<표 6>에서와 같이 제안된 프로토콜 A-EKE-E는 기존의 EKE-E보다 적은 해시연산을 사용한다. 그 이유는 기존에 필요했던 상호인증 과정을 키 생성과정으로 통합함으로써 가능하다. 따라서 전체 연산 중 2번의 해시 연산이 적게 사용되므로 성능이 향상된다.

<표 6> EKE-E와 A-EKE-E의 성능비교

	A	B	Total
EKE-E	1P+2S+3H+1R	1P+2S+3H+1R	2P+4S+6H+2R
A-EKE-E	1P+2S+2H+1R	1P+2S+2H+1R	2P+4S+4H+2R

P : Pass
S : Symmetric key crypto system
H : Hash function
R : Random number generation

3.2 안전성 분석

• **Replay attack :** 공격자는 프로토콜 중간에 사용된 메시지를 재사용하여 인증 및 암호화과정을 수행할 수 있는 공격이다. 제안된 프로토콜에서는 타임스탬프를 사용하여 시간이 경과된 메시지는 다음 과정을 수행하지 못한다. 따라서 공격자는 메시지를 재사용할 수 없다.

• **Pre-play attack :** 이전에 사용된 메시지를 기록하고 특정메시지를 통해 정보를 얻는 공격이다. 하지만 타임스탬프에 의해 정보의 기간이 한정되므로 공격이 불가능하다.

• **Eavesdropping :** 공격자는 통신라인상에 메시지를 듣고 중요한 정보를 얻으려고 한다. 하지만 통신상에 사용되는 메시지는 암호화 과정을 거친 메시지로써 비밀값을 알지 못하는 공격자는 중요한 정보를 얻는 것이 불가능하다.

• **Man-in-the-middle attack :** 공격자는 통신이 수행되는 라인의 중간에 위치하여 중요한 정보를 얻는다. 프로토콜을 수행하는 사용자들은 중간에 위치한 공격자를 발견할 수 없으며 안전한 보안 통신을 하고 있다고 생각하게 된다. 제안된 프로토콜은 비밀값 X_1과 Y_1을 암호화하여 보내게 된다. 공격자는 사용자들간에 분배된 P값에 대한 정보를 얻을 수 없으므로 인증메시지를 생성하는 것이 불가능하다.

• **Password guessing attack (Off-line dictionary attack) :** 공격자는 사용자가 사용하게 되는 패스워드를 자주 사용하는 문구와 기록된 통신을 통해 예측할 수 있다. 하지만 P값에 대한 정보를 알 수 없으므로 공격자는 패스워드를 예측할 수 없다.

• **Password file compromise :** 공격자는 사용자임을 가장하여 패스워드 정보를 가지고 있는 자료들에 접근한다. 제안된 프로토콜은 통신상에 이루어 지는 정보의 보안은 안전하게 유지된다. 단 여기서 정보가 저장되는 매체에 대한 고려는 하지 않는다.

• **Server compromise :** 공격자가 서버임을 가장하여 중요한 정보를 얻는 공격이다. 하지만 공격자는 P값에 대한 정보를 가지고 있지 못하므로 상대방에게 자신이 서버임을 증명할 수 없다.

4. 결론

제안된 프로토콜은 단위추가 N수에 의하여 나타낼 수 있는 암호를 사용한 EKE-E 프로토콜을 개선하여 해시연산을 수행횟수를 2번 감소시켜 효율적인 연산이 가능하도록 하였다. 또한 기존의 EKE-E에서 보장하는 Man-in-the-middle-attack과 오프라인 사전공격 그리고 재전송공격에 대해서도 안전하다. 본 논문에서 제안한 프로토콜은 다양한 공격에 대해 안전하며 기존의 프로토콜과 비교해 우수한 성능을 가진 키교환 프로토콜이다.

참고문헌

[1] E. Bach, Algorithmic Number Theory, Vol 1 : Efficient Algorithms, MIT Press Cambridge, Massachusetts, 1996.

[2] M Bellare, D. Pointcheaval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," Advances in Cryptology Eurocrypt'00, LNCS Vol. 1807, Springer-Verlag, pp. 139-155, 2000.

[3] Jong-Min Park and Byung-Jun Park "Authenticated Key Exchange Protocol for the Secure and Efficient," The korea institute of maritime information & communication sciences, vol. 14, no 8, Sep. 2010.

[4] W.Diffe and M. Hellman "New Direction in Cryptography," IEEE Transaction on Information Theory, vol. 22, no. 6, pp. 664-654, Nov. 1976.