Dictionary Attack 방지를 위한 S/KEY 설계 및 구현

김일곤°, 방기석, 최진영
고려 대학교 컴퓨터학과
{igkim,ksbang,choi}@formal.korea.ac.kr

The Design and Implementation of S/KEY against Dictionary Attack

Il-Gon Kim°, Ki-Seok Bang, Jin-Young Choi
Dept of Computer Science & Engineering, Korea University

요약

네트워크 컴퓨팅 시스템에서 생길 수 있는 공격유형중의 하나는 로그인 아이디, 패스워드와 같은 인증정보를 네트워크상에서 가로채는 것이다. 이러한 정보를 일단 획득하면 후에 언제든지 이용할 수 있게 되는 것이다. 일회용 패스워드 시스템은 이러한 "재공격(replay attack)"을 방어하기 위해 Bellcore 사에 의해 고안되어졌다. 하지만 이 인증 시스템은 취약점을 가지고 있는데 만일 공격자가 자신이 가지고 있는 사전에서 passphrase 를 유추해 낼 수 있다면 결국 SKEY 의 결과값인 일회용 패스워드까지 알아낼 수 있게 된다. 따라서 이 passphrase 를 보다 안전하게 사용자와 시스템간에 전달할 수 있게 하기 위해 EKE(Extended Key Exchange) 프로토콜을 사용하여 키의 스니퍼링 뿐만 아니라 dictionary attack 을 방지하고자 하였다.

1. 서론

최근 인터넷의 급격한 확산과 더불어 전세계 어디에서나 손쉽게 인터넷에 연결된 기관의 전산망에 접근하여 필요한 정보를 획득, 이용하는 것이 가능해지게 되었다. 이에 대한 역기능으로 전산망에 대한 크래커들의 불법적인 침입과 위협이 날로 증대하고 있다. 특히 인터넷의 주요 구성요소인 유닉스의 운영체제와 TCP/IP 가 정보 보호 측면에서 많은 취약점을 가지고 있어 인터넷에 연결된 모든 전산망이 크게 공격자로부터 위협을 받고 있는 상태이다. 이러한 공격으로부터 시스템을 안전하기 유지하기 위한 많은 방법들이 연구되고 있으며, 그러한 방법들로서는 암호 알고리즘, 즉 대칭키, 비대칭키를 이용한 여러가지의 암호 프로토콜(예, Kerberos, IPsec, SSH 등) 및 보안장치(예, 방화벽, 침입탐지 시스템)들이 있다. 이 논문에서 설명하고자 하는 S/KEY [1] 혹은 일회용 패스워드 시스템 또한 공격자가 네트워크상에서 암호를 가로챈다 하여도 그 이후에 로그인할 때는 그 암호를 재사용할 수 없도록 하는 방식을 사용하고 있는 보안 방식을 취하고 있다. 하지만 점차적으로 컴퓨터 시스템의 연산능력의 향상으로 인한 강력한 CPU 를 갖춘 컴퓨터들이 등장함에 따라 기존의 암호 프로토콜을 부수는데 예전보다 연산시간이 점차 짧아지고 있음을 알 수 있다. 그리고 사용자가 사용하는 패스워드의 길이가 그다지 길지 않다는 부주의성을 이용하여 행해지는 이른바 "사전공격(Dictionary Attack)[6]" 또는 "무차별공격(Brute-force Attack)"으로 인해 발생하는 피해는 매우 크다고 할 수 있다. 예를 들면, 유닉스 계열에서 패스워드를 저장할 때는 "/etc/passwd" 형태로 패스워드 데이터가 저장되게 된다. 물론 저장된 패스워드는 salt 와 key 의 조합을 통한 대칭키형식으로 저장되어, 일반인들은 이 파일이 매우 안전할 것이라고 생각하게 된다. 하지만 요사이 많은 사전공격 도구[7]가 인터넷상에서 유통되어 지고 있으며, 실제로 "John the Ripper" 나 "Crack 5.0" 과 같은 도구를 이용하면 7 자리의 패스워드들을 몇 시간이내에 깰 수 있다고 한다. 이것은 절대로 간과해서는 안될 매우 위험한 사항이라고 하겠다. 이 논문의 2 장에서는 SKEY 시스템, 즉 일회용 패스워드 시스템에 대한 소개 및 취약점에 대해 다루고, 3 장에서는 dictionary attack 을 방지하기 위한 해결방안에 대해 논하고, 마지막으로 4 장에서는 결론 및 향후 연구방향을 제시한다.

2. S/KEY

2.1 소개

S/KEY 인증 시스템은 수동적 공격(passive attack)으로부터 사용자의 패스워드를 보호하기 위해 만들어진 스킴이다. 이 인증 시스템은 추가적인 하드웨어나 별도의 정보를 추가하지 않고도, UNIX 시스템에 쉽게 부착할 수 있는 시스템이다. 처음에는 Bellcore 사의 내부 네트워크를 보호하기 위해 개발되어 졌다. 일회용 패스워드 시스템의 동작에는 두개의 중요한 동작절차가 있다. 일회용 패스워드 생성기는 서버로 부터받은 챌린지(challenge) 정보와 사용자의 비밀 "passphrase"로부터 적절한 일회용 패스워드를 생성해야 한다. 서버는 생성기에게 적절한 생성 파라미터를 포함한 챌린지(challenge)를 전달해야 하며, 수신한 일회용 패스워드를 확인하고, 일회용 패스워드 일련 숫자를 저장해야만 한다.

일회용 패스워드 생성기는 서버로부터 수신한 seed, sequence number 와 더불어 사용자가 passphrase 를 입력받아 해쉬함수를 일정량 반복하여 일회용 패스워드를 생성하게 되는 것이다. 인증이 성공적으로 이루어지면, 서버는 해쉬함수의 반복 계수를 하나만큼 감소시키게 된다. 이러한 기술은 처음 Leslie Lamport 에 의해 제안된 방식이다.

2.2 문제점

S/KEY 에서 사용하는 challenge-response scheme[2]은 ISO/IEC 9728-4 의 5.1.2 항에서 기술하고 있으며 이 프로토콜의 기본적인 방식은, B 가 A 를 확인하기 위해 다음과 같은 메시지 교환을 갖는다.

1. B 는 A 에게 challenge R 을 보내게 된다.
2. A 는 B 에게 $f_K(R)$ 에 해당하는 response 를 보내게 된다.

(K :비밀키, f:암호화 체크 함수)

이러한 인증 프로토콜의 요구사항으로서, ISO/IEC 9798-4 에서는 challenge R 이 제 3 자에 의해 예측할 수 없어야함을 명시하고 있다. 그러나 S/KEY 시스템에서 전달되어지는 일련의 seed 와 count 값은 동일값을 취해 이러한 값은 예측할 수 있다. 이러한 사항들은 다음과 같은 문제점을 야기시킬 수 있다. 처음 이러한 사실을 인지한 사람은 Gong[8]이며, 그에 따르면, 만일 challenge-response 프로토콜에서 challenge 를 예측할 수 있고, challenge 가 암호화 되어 있지 않다면 'suppress-replay attack'과 같은 종류의 공격 가능성이 있다. 공격이 행해지는 일반적인 시나리오는 다음과 같다.

username: user A
challenge: 99 k113355
response: WELD GUY CHIMP SWING GONE
user A's real password: ????

dictionary word 1: love
challenge: 99 k113355
response: BAD LOST CRUMB HIDE KNOT
(well that's not it...)

dictionary word 2: password
challenge: 99 k113355
response: FORT HARD BIKE HIT SWING
(not it either...)

dictionary word 3: secret
challenge 99 k113355
response: WELD GUY CHIMP SWING GONE
(bingo!)

사용자 C 는 B 에게 사용자 A 인 것처럼 위장한다. 만일 C 가 B 에 의해 A 에 전달되어지는 challenge R 을 가로채거나 예측할 수 있다면, C 는 A 로 위장하여 B 에게 인증요청을 하게 된다. 즉, A 가 B 에게 전달하는 response 까지 가로챈다면 그 위험성은 더욱 커지게 된다. 이러한 시나리오를 S/KEY 시스템에 적용해 보면 공격자는 서버에서 전달되는 seed & sequence number 를 포함하는 challenge 를 유추하거나 가로챈 후, passphrase 를 사전에 있는 단어들과 비교하여 결국에는 response 까지 알아낼 수 있다는 것이다. [3]

user A 의 일회용 패스워드를 알아내기 위해 악의적인 공격자는 우선 user A 에게 전달되어지는 challenge 를 가로챈 후, 사용자의 passphrase 를 알아내기 위해 공격자가 가지고 있는 사전에 있는 단어들과 맞추어 본 다음, 만일 사전에 있는 단어라면 위에서 보여준 바와 같이 쉽게 response 를 생성해낼 수 있는 것이다. 이제 user A 의 패스워드가 'secret'라는 사실을 알았다. 이러한 정보를 바탕으로 해서 challenge 가 무엇이든지에 상관없이 유효한 response 를 생성해낼 수 있는 것이다.

3. 수정된 S/KEY 시스템의 설계

S/KEY 시스템에서 생길 수 있는 passphrase 에 대한 dictionary attack 을 방지하기 위해서, 무엇보다도 passphrase 를 어떻게 하면 보다 안전하게 사용자와 호스트사이에 전달할 수 있는가에 중점을 두었다. passphrase 가 안전하게 전달되기 위해서는 우선적으로 공격자에 의해 생길 수 있는 스니퍼링과 같은 가로채기 공격을 막아야 한다. 이를 위해서는 물리적 보안은 물론 이거니와 채널사이에 지나가는 패킷을 암호화하는 SSH 와 같은 안전한 채널을 유지해야 함은 필수적인 사항이다. 하지만 오프라인상에서 행해지는 사전공격을 막기위해서는 passphrase 에 대한 길이를 최대한 확장하여 그 공격 가능성을 방지하고자 하였다.

그러기 위해 이 논문에서는 EKE(Encrypted Key Exchange) protocol[4]을 S/KEY 시스템에 적용해 보았다. 사용자가 선택한 키에 바탕을 둔 전형적인 암호 프로토콜의 문제점은 공격자가 패스워드를 추측해서 알아낼 수 있다는 것이었다. 특히 이러한 문제점은 패스워드에 기반한 키 교환방식에서 두드러지게 되었다. 사용자들은 누구나 쉽게 기억할 수 있게 하기 위해 패스워드를 설정할 때, 그만 그 안정성을 고려하지 않고 매우 짧은 길이의 패스워드를 설정해 버리게 되는 것이다. 그럼 과연 어떻게 하면 짧은 패스워드를 가지고서도 상호가 안전하게 키 전달을 통해 서로를 인증하고 통신할 수 있을까 하는 문제가 관심을 가지게 되었다. 그래서 나온 방안이 기존의 Diffie Hellman 의 키교환 방식[9]과 유사한 형태로, 대칭키(비밀키)와 비대칭키(공개키)를 새롭게 조합한 형태의 암호방식을 통해 불안전한 채널에서 보다 안전하게 서로가 정보를 주고 받을 수 있는 연구가 진행되어 오고 있다. 이러한 프로토콜중의 하나가 바로 EKE(Extended Key Exchange) 프로토콜이다. 이 EKE 를 보다 확장 변형한 형태의 프로토콜들이 생겨나고 있는데, SPEKE(Simple Password Exponential Key Exchange) , OKE(Open Key Exchange), A-EKE, DH-EKE 등[5]이 이에 속한다. EKE 는 대칭과 방식과 비대칭키방식을 조합하여 사용되어진다. 또한 비대칭키 방식에 다양한 조합을 통하여 변형을 가져올 수 있다. 비대칭키 방식으로 RSA 와 ElGamal 방식을 사용할 수 있으며 각각 그 장단점을 가지고 있다. 이를 S/KEY 에 적용시켜 살펴보면, challenge R 은 count 와 seed 의 조합이며, $f_K(R)$은 $g^e(K \oplus D)$, g 는 단방향 함수, K 는 64 비트 키를 의미한다. 수정후 S/KEY 시스템의 동작은 수정전과 거의 유사하다 다만 passphrase 를 user 와 host 상호간에 암호화하여 전달한다는데 그 차이점을 두고 있으며, 이 프로토콜이 사전공격으로부터 안전하기 위해서는 무엇보다도 상호간에 공유되는 임의의 비밀키 R 의 길이에 달려있다.

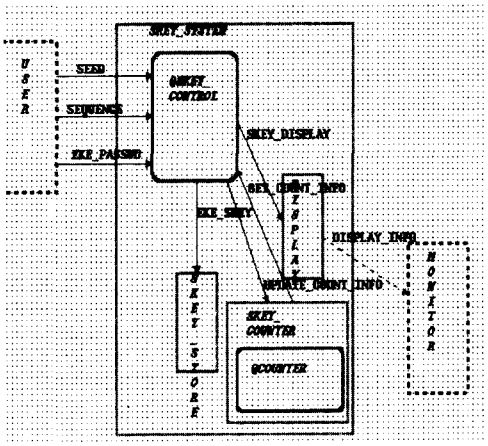

그림 1. 수정된 S/KEY 시스템

다음은 EKE 프로토콜의 동작절차를 나타내 주고 있다.

1.U(user)는 임의의 키쌍 E_A와 D_A를 생성하고 상호간에 공유하고 있는 대칭키 P(passphrase)로 E_U를 암호화, $P(E_U)$를 생성한다. U는 $\{U, P(E_U)\}$를 H(host)에게 전송한다.
2.H는 복호화하여 $P^{-1}(P(E_U))=E_U$를 얻은 후, 임의의 비밀키 R을 생성하고 이를 키 E_U를 이용하여 암호화하여 $P(E_U(R))$생성한다. U는 $\{P(E_U(R))\}$를 H에 전송한다.
3.U는 메시지를 복호화하여 R을 구한다.
$D_U(P^{-1}(P(E_U(R))))=R$

```
plaintext1 is :Zf4iX95w7vyGA1G
ciphertext is:WZP4TUMa34GwJiw
des_input1 :0101011101011010010101000000110100010101000101010100110101100001
des_input2 :0011001100110100010001110111011101010101100101011101110000000
outBlk1:0110110101110000011001101001001001010111101010001101001001100l
outBlk2:0110001011110010010000100111101000010110001101110001001110001010
final_encryption:0110110101110000011001101001001001010111101010001101001001100
1011000101111001001000010011110100001011000110111000100111000101001010111111

root 0099 so480201          0010101000001000001101000001100111011101110101010000
1010110000000100100111000001111001001011011110001110011001110011110011101110110
0010000000001100101011000010110001100110100100100  Sep 01,2001 15:01:56
```

그림 2. 암호화된 S/KEY의 입력값과 출력값

지금까지 위의 사항을 토대로 설계한 S/KEY 시스템은 아래 [그림 1] 과 같다. 즉 S/KEY 시스템에 사용자 인증을 위하여 입력하는 값중 passphrase를 EKE 프로토콜을 이용하여 암호화하여 전달할 뿐만아니라 [그림 2]에서 보는 바와 같이 S/KEY의 결과값을 다시금 EKE 프로토콜에 적용한 알고리즘으로 암호화하여 S/KEY 시스템의 입력값과 출력값 모두를 dictionary attack으로 부터 보호하도록 설계하였다.

4.결론 및 향후 연구 방향

S/KEY 시스템에 로그인할 때 사용하는 비밀키인 passphrase를 dictionary attack으로 부터 방지하기 위해, 수정된 S/KEY 시스템에서는 128 비트 크기의 임의의 passphrase를 생성하도록 구현하였으며, 또한 /etc/skeykeys에 저장되어진 S/KEY의 결과값인 skey.val의 값을 EKE 프로토콜에 적용된 알고리즘으로 암호화하여 저장하였다. 128 크기의 바이너리 배열로 저장하였으므로 수학적으로 공격자의 사전에서 passphrase와 skey.val의 값을 산출해 내기란 매우 힘들다고 할 수 있다. 이로써 기존의 S/KEY 시스템에서 생길 수 있는 dictionary attack을 방지하여 보다 안전한 시스템을 설계하고 구현하였다.
향후 연구 방향으로는 시스템의 안전성을 검증하기 위해서, 수정된 S/KEY 시스템에 적용된 EKE 프로토콜의 안전성을 정형검증 도구를 통하여 검증하여 시스템에 공격 모델을 첨가하였을때도 프로토콜이 안전한지를 검증해 보고자 한다.

5.참고 문헌

[1] THE S/KEY™ ONE-TIME PASSWORD SYSTEM, Neil M,Haller , Bellcore Morristown, New Jersey
[2] A One-Time Password System,rfc 1938, N.Haller
[3] Comments on the S/KEY user authentication scheme, Liqun Chen and Chris J.Mitchell
[4] Extended Password Key Exchange Protocols Immune to Dictionary Attack. David P.Jablon, Integrity Sciences,Inc
[5] Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks, Steven M.Bellovin & Michael Merritt, AT&T Bell Laboratories
[6] Authenticated Key Exchange Secure Against Dictionary Attacks. Mihir Bellare, David Pointcheval and Phillip Rogaway
[7] HACKING EXPOSED, Network Security Secrets & Solutions. Brian Hatch, James Lee, George Kurtz
[8] L.Gong, M.Lomas, R.Needham, J.Saltzer, "Protecting Poorly Chosen Secrets from Guessing Attacks", IEEE Journal on Selected Areas in Communications, Vol. 11, No. 5, (1993), pp. 648--656.
[9] IEEE P1363 Standard for RSA, Diffie-Hellman and Related Publ ic-Key Cryptography - Elliptic Curve Systems, February 6, 1997.