Casper, FDR를 이용한 SSH 프로토콜의 안전성 분석

김일곤[0], 최진영
고려대학교 컴퓨터학과
(igkim[0], choi)@formal.korea.ac.kr,

The Security Analysis of SSH protocol using Casper, FDR

Il-Gon Kim[0], Jin-Young Choi,
Dept of Computer Science & Engineering, Korea University

요약

인터넷의 비약적인 발전과 더불어, 인터넷을 활용한 각종 온라인 서비스가 활성화 되어 가고 있는 추세이다. 온라인 쇼핑몰, 온라인 뱅킹과 같은 전자 상거래 서비스는 이용자와 서비스 제공자간의 상호 신뢰를 기반으로 동작해야만 하는 서비스이다. 해당 서비스에 대한 적합한 사용자를 인증하고 인가하기 위한 다양한 인증 프로토콜(EKE, S/KEY, Kerberos등)에 대한 다양한 연구가 진행되어 오고 있다. 본 논문에서는 FDR, Casper과 같은 보안 프로토콜 분석 및 검증 도구를 이용하여 SSH 프로토콜의 위험성을 분석하여 보안 프로토콜의 안전성을 향상시키고자 한다.

1. 서론

인터넷의 비약적인 발전과 더불어, 인터넷을 활용한 각종 온라인 서비스가 활성화 되어 가고 있는 추세이다. 이런 인터넷 기술을 기반으로 하여 온라인 쇼핑몰, 온라인 뱅킹과 같은 전자 상거래 서비스가 인터넷의 사용의 편리성으로 인해 점차 각광을 받고 있다. 하지만 현재의 인터넷 서비스는 사용자의 부주의와 프로토콜상의 문제점으로 인해 악의적인 공격자들의 그 안전성 문제가 발생하기도 한다. 이에 따라 해당 서비스에 대한 적합한 사용자를 인증하고 인가하기 위한 다양한 인증 프로토콜(EKE[1], S/KEY[2], Kerberos[3]등)에 대한 다양한 연구가 진행되어 오고 있다. 하지만 이런 보안 프로토콜은 그 안전성을 분석하고 검증하기가 무척 어렵다. 예를 들면, 대표적인 보안 프로토콜인 Neddham-Schroeder 프로토콜[4]이 개발될 당시에는 누구나 이 프로토콜의 안전성을 확신하였다. 하지만 이 프로토콜이 개발된지 16년이 지나서야 비로서 그 위험성을 발견할 수 있었다.

보안 프로토콜의 안전성은 사용자들 뿐만 아니라 국가 경제에 큰 영향을 미칠 수 있기 때문에, 반듯이 보장되어야만 하는 과제이다. 하지만 우리나라에서는 아직도 이런 인증 프로토콜에 대한 안전성을 분석하고 검증하는데는 선진국에 비해 연구가 뒷쳐져 있는게 사실이다. 이미 외국에서는 오래 전부터 다양한 검증 방법을 통해 이런 보안 프로토콜의 안전성을 분석하고 검증하고자 하는 노력을 기울여 왔다. 보안 프로토콜을 검증하기 위한 주요 방법들로는 BAN, GNY, SVO등[5]과 같은 보안 프로토콜에 적합한 로직을 사용한 정리 증명 방법과 SMV, SPIN, Murphi, FDR과 같은 모델체킹[6] 방법등이 있다. 다양한 검증 방법중에서도 프로세스 알제브라 언어인 CSP(Commnunication Sequential Process)[6]를 이용해 보안 프로토콜을 명세한 다음, FDR, Casper[7,8]와 같은 도구를 이용해 보안 프로토콜을 명세하고 검증하여 다양한 보안 프로토콜의 결점을 찾아 낼 수 있었다.

따라서 본 논문에서는 FDR, Casper와 같은 자동화 도구를 사용하여 현재 사용자 인증을 위해 많이 사용되어 오고 있는 SSH(Secure Shell)[9] 보안 프로토콜의 위험성을 분석하여, 보안 프로토콜의 안전성을 향상시키고자 한다.

본 논문의 2장에서는 SSH 보안 프로토콜에 대한 소개 및 보안 기능을 보여주고, 3장에서는 FDR과 Casper 도구에 대해 설명하고, 4장에서는 FDR과 Casper를 이용하여 SSH 보안 프로토콜을 분석한 결과를 보여주고, 마지막으로 5장에서는 결론 및 향후 연구 방향을 제시하고자 한다.

2. SSH(Secure Shell)

SSH 보안 프로토콜은 RSA[3] 암호 매커니즘을 사용하여 암호와 호스트 인증을 통해 클라이언트와 서버간에 안전한 통신 채널을 제공한다. SSH의 장점은 공개키 기반의 암호화 방식을 사용하여 안전하지 않은 통신 채널을 갖고 있는 사용자들간에 보다 안전한 암호화 통신을 해주기 때문에, 악의적인 공격자가 스니핑 도구를 이용해 사용자의 아이디와 패스워드를 쉽게 가로채지 못하게 한다는 것이다.

SSH 프로토콜의 호스트 인증은 크게 4종류로 나눌 수 있다. 첫번째 방식은 "RSA-Host Based Authentication", 두번째 방식은 "Rhosts and RSA Authenitcaion", 세번째 방식은 "RSA Authentication"으로 이 세가지 인증 방식은 패스워드 기반 인증 방식과 달리 클라이언트와 서버간의 공개키와 비밀키, 난수를 이용하여 상호 인증하는 비패스워드 인증 방식이고, 마지막으로 네번째 방식은 클라이언트로 서버간에 미리 설정해 놓은 공유키인 패스워드를 통해 인증하는 패스워드 기반 인증 방식을 사용하고 있다.

본 논문에서는 이 4가지 인증 방법중에서 "RSA Authentication" 방법을 선택하여, 그 안정성을 분석하였다.

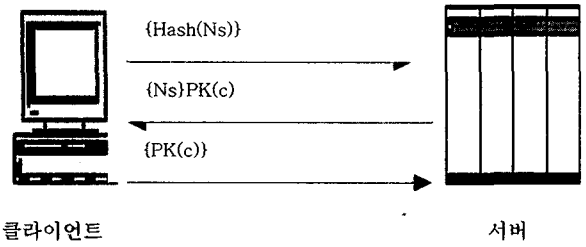

그림 1. SSH 프로토콜의 RSA Authentication 절차

위의 그림 1은 "RSA Authentication" 절차를 나타내고 있다. 클라이언트와 서버간에 인증을 위해 전달되는 메시지는 다음과 같다.

- PK(c) : 클라이언트의 공개키
- Ns : 서버에서 생성되는 임의 수
- Hash : MD5 해쉬 함수

```
[igkim@paragon igkim]$ ssh paragon.korea.ac.kr
Enter passphrase for key '/home/igkim/.ssh/id_dsa':
Last login: Sat Aug 17 16:12:10 2002 from paragon.korea.ac.kr
```

인증 성공

```
[igkim@paragon igkim]$ ssh paragon.korea.ac.kr
Enter passphrase for key '/home/igkim/.ssh/id_dsa':
Enter passphrase for key '/home/igkim/.ssh/id_dsa':
Enter passphrase for key '/home/igkim/.ssh/id_dsa':
Permission denied (publickey,keyboard-interactive,hostbased).
```

인증 실패

그림 2. SSH 프로토콜을 이용한 인증 예제

그림 2는 SSH 프로토콜의 "RSA Authentication" 방법을 사용하여 서버에 인증을 요구하였을시, 인증이 성공한 경우와 실패한 경우를 보여주는 그림이다.

3. Casper, FDR

Casper(A Compile for the Analysis of Security Protocols)[8]는 CSP로 프로토콜을 명세하는데 걸리는 시간을 단축시켜주고 보다 모델링을 쉽게 하도록 개발되어진 컴파일러이다. Casper에서 입력 파일은 프로토콜의 동작과 체킹해야할 시스템을 표현한다. 따라서 입력 파일은 다음과 같이 두 부분으로 나뉘어진다.

- 프로토콜이 동작하는 방법을 정의한다. (호스트들간에 전달되는 메시지와 데이터 아이템의 타입등)
- 체킹해야할 실제 시스템을 정의한다. (실제 시스템에서 동작하는 호스트의 역할, 실제 데이터 타입, 공격자의 배경지식등)

Casper에서 위의 사항들을 8개의 세부항목으로 분류하고 있으며, 각 항목의 헤더 부분은 "#"로 시작한다.

1. Protocol description
2. Free variables, Processes
3. Specifications
4. Algebraic equivalences
5. Type definition
6. Functions
7. System definition
8. The intruder

FDR(Failure Divergence Refinements)[7] 도구는 CSP(Communication Sequential Process)[6]를 입력 언어로 받아, 모델이 해당 속성을 만족하는지 않는지를 체킹하는 모델 체킹 도구이다. 즉 보안 프로토콜 모델을 프로세스 알제브라 언어인 CSP로 명세한 후, 보안 프로토콜이 반드시 갖추어야 하는 요구사항인 비밀성, 무결성, 인증, 부인방지와 같은 보안 속성을 만족하는지 검사하는 도구라고 할 수 있다. FDR은 다음과 같은 세가지 모델을 지원해 준다.

1. trace model

프로세스는 그 프로세스가 갖는 행위에 의해 유한 순서 집합으로 표현되며, P의 프로세스가 Q 프로세스의 모든 행위들을 포함할 때 $P \sqsubseteq_T Q$라고 표기한다

$$P \sqsubseteq_T Q \triangleq traces(Q) \subseteq traces(P)$$

2. failure model

failure는 (s, X)의 쌍으로, s는 *traces(P)* 에서의 trace를 말하고 X는 s 이후에 프로세스가 거부하는 모든 이벤트의 집합을 말한다. 즉 dead lock 상태를의미하며, 다음과 같이 표기한다.

$$P \sqsubseteq_T Q \triangleq traces(Q) \subseteq traces(P)$$

3. failures/divergence model

프로세스의 divergence는 livelock을 의미한다. 즉 failure/divegence model은 dead lock 상태 이면서 livelock 상태를 의미하며, 다음과 같이 표기한다.

$$P \sqsubseteq_{FD} Q \triangleq Failures(Q) \subseteq failures(P) \wedge divergences(Q) \subseteq divergences(P)$$

4. Casper, FDR를 이용한 SSH 프로토콜 위험성 분석

4.1 SSH 프로토콜 모델링

본 논문에서는 SSH 프로토콜의 "RSA Authentication"을 Casper를 이용해 모델링하였다. 아래 그림3은 8개의 세부 항목중 Free variables 항목을 보여주고 있다.

```
#Free variables
a, b : Agent
PK : Agent -> PublicKey
SK : Agent -> SecretKey
F: HashFunction
nb: Nonce
InverseKeys = (PK,SK),(F,F)
```

그림 3. RSA Authentication의 Free variables

a,b 는 각각의 두 호스트, PK는 공개키, SK는 개인키, F는 해쉬함쉬, nb는 b호스트의 임의 수, InverserKeys는 암호화 함수에 대응되는 복호화 함수를 나타낸다.

4.2 위험성 분석

Casper를 이용한 컴파일하여 생성된 CSP 코드를 FDR 모델체커에 입력한 후, 디버거를 통해 다음과 같은 행위를 찾아낼 수 있었다.

```
env.Alice(Env0,Mallory,<>)
intercept.Alice.Mallory(Msg1,PK_.Alice,<>)
fake.Alice.Bob(Msg1,PK_Alice,<>)
intercept.Bob.Alice(Msg2,Encrypt.(PK_.Alice, <Nb>),<>)
fake.Mallory.Alice(Msg2,Encrypt.(PK_Alice,<Nb>),<>)
signal.Running1.INITIATOR_role.Alice.Mallory.Nb
signal.Commit1.RESPONDER_role.Bob.Alice.Nb
```

위의 내용에 대한 공격 시나리오는 그림 4에 나타나 있다.

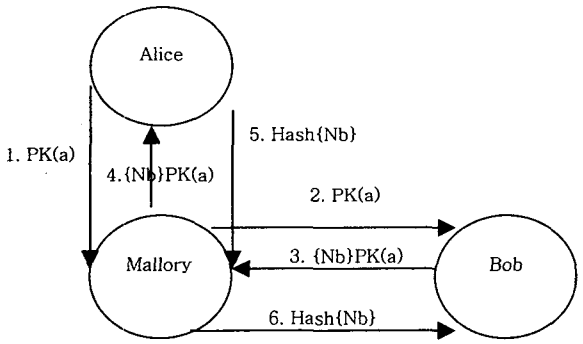

그림 4. RSA Authentication 공격 시나리오

위의 분석 내용을 토대로 보면 Alice는 공격자 Mallory가 Bob인줄 알고 자신의 공개키 PK(a)를 보내게 되며 Mallory는 자신이 가로챈 공개키 PK(a)를 Alice로 가장하여 Bob에 전달하고 Bob은 자신이 생성한 임의 수 Nb를 PK(a)로 암호화하여 전달하게 되고 Mallory는 이 암호화된 메시지를 Alice에게 전달하게 되면 Alice는 인증을 위해 Bob의 임의 수를 해쉬화한 메시지 Hash{Nb}를 Bob인줄 알고 전달하지만 이 역시 Mallory가 가로채게 됨으로써 Mallory는 Bob에 인증을 받을 수 있게 된다. 이런 취약점은 Needham shroeder 프로토콜에서 발견된 man-in-the-middle-attack[10]방법과 유사하다. 이런 공격방법은 스니퍼링과 스푸핑 방법을 이용하여 상대방을 속일 수 있기 때문에 사용자는 SSH 프로토콜을 사용할 경우, 나머지 3가지 인증 기능을 혼합하여 보다 높은 안정성을 갖도록 해야 한다.

5. 결론 및 향후 연구 방향

보안 프로토콜의 안정성을 분석하고 검증하는 일은 개인 정보의 유출막기 위해서 뿐만 아니라 국가 경제, 보안에 중대한 관심사이다. 하지만 보안 프로토콜의 취약점을 알아내기란 쉽지 않다. 그 첫번째 이유는 보안 프로토콜이 갖는 특수성에 있다. 즉, 전문가들 조차도 분석하는데 어려움을 갖고 있다는 것이다. 이런 문제를 해결하기 위해 본 논문에서는 Casper, FDR와 같은 자동화 도구를 사용하여 실제 인증을 위해 많이 사용되어지는 SSH 프로토콜의 공격 가능성을 찾아내어, 상호간에 보다 안전한 통신을 유도할 수 있었다. 하지만 보안 프로토콜의 안전성을 분석하고 검증하기 위해서는 무엇보다도 공격자의 공격 행동을 정형화하는 일이 필요하다. 공격자의 다양한 공격 행동을 모델링하기 위한 연구가 요구된다.
향후 연구방향으로는 차세대 모바일 환경에서 사용될 DIAMETER 프로토콜[12]을 다양한 정형 검증 도구를 이용해 그 안정성을 분석해 보고자 한다.

6. 참고문헌

[1] Steven M. Bellovin, Michael Merrritt, Encrypted key Exchange: Password-Based Protocols Secure Against Dictionary Attacks
[2] THE S/KEYTM ONE-TIME PASSWORD SYSTEM, Neil M,Haller , Bellcore Morristown, New Jersey
[3] William Stallings, Network Security Essentials
[4] Gavin Lowe, Breaking and Fixing the Needham-Schroeder Public-Key Protocol using FDR
[5] A.D. Rubin, P. Honeyman, Formal Methods for the Analysis of Authenitcation Protocols CITI Technical Report 93-7, November 8, 1993
[6] C.A.R Hoare, Communicating Sequential Processes,
[7] Formal Systems(Europe) Ltd. Failure Divergence Refinement-FDR2 User Manual, Aug. 1999
[8] Gavin Lowe, Casper(A Compiler for Analysis of Security Protocols) User Manual and Trutorial, version 1.3, July 13th, 1999
[9] Scott Mann, Ellen L.Mitchell, Linux System Security The Administrators's Guide to Open Source Security Tools
[10] Joel Scambray, Stuart McClure, George Kurtz, Hacking Exposed(Second Editiion)
[11] Joel Scambray, Stuart McClure, Hacking
[12] AAA Working Group Internet-Draft Category Standards Track <draft-ietf-aaa-diameter-12.txt>