TTSSH를 이용한 OpenSSH 서버로의 원격접속에 관한 연구

강민정* · 강민수* · 박연식**

*경상대학교 · **경상대학교 해양산업연구소

Study about remote-access by using TTSSH to OpenSSH Server

Min-jung Kang* · Min-su Kang* · Yeoun-sik Park**

*Gyeong-sang National University · **GSNU-TOMI

E-mail : lundi@opentown.net, mikisoo@hitel.net, parkys@nongae.gsnu.ac.kr

요 약

현재 지구상에서 운영되고 있는 상당수의 서버들은 유닉스 기반이며, 최근엔 리눅스 기반의 서버도 꾸준히 증가하고 있는 추세다. 이러한 서버에 계정을 가진 사용자들이 원격접속을 하고자 할 때 가장 쉽게 사용하는 명령어가 telnet인데, 이는 통신 내용은 물론 로그인시 사용하는 ID와 비밀번호에 대해서도 전혀 보안이 이루어지지 못하고 있는 실정이다.

대안으로 최근 SSH에 대한 관심이 높아지고 있지만, SSH가 telnet의 명성을 따라 잡기엔 여러 가지 문제점을 안고 있다.

따라서 OpenSSH를 중심으로 윈도우 사용자가 원격접속을 시도했을 때 발생할 수 있는 문제점들과 그 해결책에 관하여 연구하였다.

ABSTRACT

Many servers that is operated in present earth are UNIX base, is trend that server of LINUX base is increasing steadily recently.

When users who have account to this server wish to do remote access, instruction that use most easily is 'telnet', security does not consist entirely about ID and password that this uses at communication substance as well as login.

The interest about latest SSH is rising by the alternative, but SSH has various kinds problem in following telnet's fame.

Therefore, We studied about problems and the solution that can happen when window users attempted remote access laying stress on OpenSSH.

키워드

SSH, OpenSSH, TTSSH, OTP

Ⅰ. 서 론

현재 네트워크 공간에는 무수히 많은 서버들이 있으며, 이 서버들을 이용하는 사용자들의 급속한 증가는 일반 사용자들의 네트워크 지식이 상당히 향상되었음을 말해준다. 그리고 네트워크상에 끊임없이 대두되는 보안 문제를 위하여 시스템과 네트워크를 안전하게 만들기 위한 노력이 많은 전문가들에 의하여 계속하고 있지만, 대부분의 사용자들은 통신 내용이 제대로 보호되지 못하고 있는 것에 대하여 상당히 불안해하고 있는 실정이다. 그래서 패스워드 같은 개인 정보의 보안을 매우 중요시하고 있는 추세이다.

하지만 현재 널리 사용되고 있는 telnet이나 ftp 같은 원격접속 프로그램들은 보안조치가 없으므로 통신내용을 보호하지 못하고 있지만, 사용자들은 오랜 관습으로 인한 편리함 때문에 이러한 프로그램들을 쉽게 접하고 있다.

일차적인 책임은 서버 운영자에게 있으므로, 견고한 서버 운영을 위하여 운영자는 끊임없이 정보를 수집하고 업그레이드를 하여 보안에 취약한 서비스들은 중단하고 보다 보안에 신뢰가 가는 서비스들을 제공하여 이를 사용자들에게 알릴 필요가 있다. 그러나 게으른 운영자들로 인하여

사용자들은 여전히 보안이 취약한 네트워크를 이용하는 경우가 많다.

이차적인 책임은 사용자 각자에게 있으므로, 운영자의 경고를 무시하고 보안에 취약한 프로그램들을 사용하는 것을 자제하여야 할 것이다.

그러나 최근엔 보안상 취약점을 드러내는 telnet 등과 같은 원격접속 프로그램 대용으로 점차적으로 SSH(Secure Shell)가 주목받고 있다.

SSH는 r* 계열 명령어 사용법에 보안 기능을 추가한 것으로 표 1과 표 2와 같은 암호기술을 갖추고 있어서 안전한 통신을 제공한다[1][2].

표 1. SSH에서 사용하는 암호화 알고리즘

종 류	특 징
BlowFish	·64비트 암호화 기법 ·크기가 큰 데이터를 빠른 속도로 암호화
Triple DES	·112비트 알고리즘으로 'meet in-the-middle attack' 방지 ·ANSI X9.17, ISO 8732, PEM
IDEA	·128비트 키를 사용하는 강력한 블록 암화화 알고리즘
RSA	·공개키/개인키 암호화 알고리즘

표 2. SSH에서 이용가능한 대칭키 알고리즘

product	ver	algorithm
openssh	v1	3des, blowfish
	v2	3des, blowfish, RC4, cast, aes
ossh	v1	3des, blowfish, idea, des
ssh1	v1	3des, blowfish, RC4, idea, des
ssh2	v2	3des, blowfish, RC4, cast, des, twofish
f-secure ssh2	v2	3des, blowfish, idea
lsh	v2	3des, blowfish, RC4
TTSSH	v1	3des, blowfish, des
MindTerm	v1	3des, blowfish, RC4, idea, des
	v2	3des, blowfish, RC4, cast, idea, des, twofish, aes

SSH 프로토콜은 버전1과 버전2 두가지이나, 서로 전혀 호환성이 없다. 버전1은 공개키 교환 알고리즘으로 RSA를 사용하고 버전2는 DSA/DH를 사용하여 호스트 인증과 유저 인증 및 통신내용을 암호화한다[3].

이러한 SSH를 이용하기 위한 여러 상용 제품들이 출시되어 있다. 그 중 OpenSSH는 특허와 관련된 코드를 제거하고 각종 기능을 추가한 free software로써 버전1과 버전2를 모두 사용할 수 있다.

윈도우 사용자가 telnet과 같은 프로그램으로 원격 접속하는 것보다 TerraTerm SSH(이하 TTSSH)와 같은 SSH서버에 원격접속 가능한 클라이언트 소프트웨어프로그램으로 접속할 때 더 신뢰성 있는 보안이 이루어지는 것은 사실이다. 그러나 완벽한 보안을 기대하는 것은 어리석다.

따라서 대표적인 클라이언트 소프트웨어인 TTSSH의 기본적인 사용법과 보안상의 문제점을 찾아보고 그 해결책을 제시하고자 한다.

II. 서버에 원격접속 가능한 클라이언트 S/W와 TTSSH의 사용법

표 3은 현재 출시되어 있는 SSH 클라이언트 소프트웨어들이다[4]. 이 중 윈도우 기반에서 가장 널리 사용되는 것이 TerraTerm SSH이다.

표 3. SSH 클라이언트 소프트웨어들

종류	특 징
MacOS용 NiftySSH	매킨토시용으로 NiftyTelnet 프로그램의 확장버전, 상업용
MindTerm	자바로 개발, 버전1.5까지 지원, 매킨토시나 윈도우에서 사용가능, 상업용
TerraTerm SSH	TerraTerm 터미널 에뮬레이터에 DLL 형태로 추가되어 버전1.5까지 지원, 무료
FreeSSH	16bit 또는 32bit 윈도우 시스템 상에서 실행, 무료
SecureCRT	상업용
PilotSSH	팜파일럿 PDA에서 SSH 사용시, 상업용

TTSSH는 SSH 프로토콜 버전 1.5까지 지원하며 무료로 사용할 수 있다. 하지만 TTSSH는 자체적으로 키 생성을 할 수 없으므로 리눅스 컴퓨터 상에서 키를 생성하여 비밀키는 클라이언트 컴퓨터(윈도우 기반)에 복사하고, 공개키는 리눅스 컴퓨터(authorized_keys 파일)에 저장한다. 그 과정은 다음과 같다.

```
[lundi@inforcom lundi]$ssh-keygen
Generating public/private rsa1 key pair.
Enter file in which to save the key
(/home/lundi/.ssh/identity) :
Created directory '/home/lundi/.ssh'.
Enter passphrase (empty for no passphrase) :
Enter same passphrase again :
Your identification has been saved in
/home/lundi/.ssh/identity.
Your public key has been saved in
/home/lundi/.ssh/identity.pub.
The key fingerprint is :
d4:75:5d:63:c7:5c:da:d5:9c:b5:4c:96:02:3e:2f:f0
lundi@inforcom.gsnu.ac.kr
[lundi@inforcom lundi]$cd .ssh
[lundi@inforcom .ssh]$ls
identity  identity.pub
[lundi@inforcom .ssh]$mv identity.pub
authorized_keys
[lundi@inforcom .ssh]$chmod 700 .
[lundi@inforcom .ssh]$chmod 600 authorized_key
[lundi@inforcom lundi]$cd .ssh
```

TTSSH를 실행시키고 가장 중요한 단계는 인증방법의 선택과 passphrase를 기입하는 단계로써, 인증방법은 그림 1과 같다[5].

첫째, password only방식이다. 보안이 된 채널을 통하여 password를 서버에게 직접 전송한다.

둘째, RSA방식이다. ssh-keygen에 의해서 생성된 identity(사용자 비밀키) 파일을 선택한다. 대부분 이 파일은 passphrase에 의해 보호되고 있으므로 passphrase box에 passphrase를 기입한다. 가장 많이 이용되는 방법이다.

셋째, rhosts and rhosts with RSA방식이다. RSA 키를 사용하는 서버에 접근하는 local computer의 identity를 증명하기 위해서 host private key를 선택한다. RSA 방식과 마찬가지로 passphrase는 이 키에 접근하기 위하여 사용된다. 하지만 이 방식은 firewall를 사용하는 사용자들에게 발생되는 문제점들 때문에 기본적으로 사용할 수 없다.

넷째, TIS challenge/response방식이다. 패스워드 없이 서버가 먼저 challenge 메시지를 전송하면 클라이언트가 응답하는 방식으로, 기본적으로 사용할 수 없다.

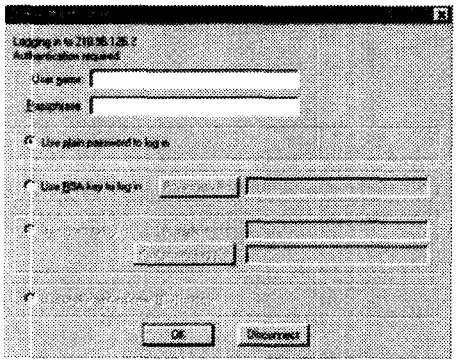

그림 1. TTSSH의 Authentication

III. TTSSH의 문제점

TTSSH 사용자들은 인증방법으로 대부분 RSA를 이용한다. 즉, SSH 인증에서는 공개키를 사용하여 인증을 하기 때문에 키 관리 문제가 곧 보안과 직결된다.

TTSSH는 호스트인증과 사용자 인증을 위한 각각의 공개키 페어가 필요하며, 이를 통해서 통신내용이 암호화된다.

먼저 호스트인증용 키는 OpenSSH 설치시 자동으로 /etc/ssh/ssh_hosts_key파일과 /etc/ssh/ssh_hosts_key.pub

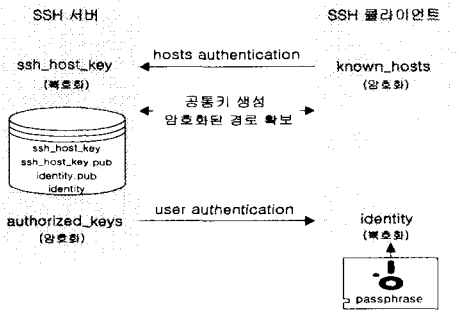

그림 2. SSH 인증구조

파일로 작성된다. 사용자인증용 키는 리눅스컴퓨터에서 작성하여 공개키는 호스트 컴퓨터에 보관하고 비밀키는 로컬 컴퓨터로 전달한다. 이 과정에서 아래와 같은 문제점을 발견할 수 있다.

첫째, 사용자인증용 키를 생성하기 위해 주로 사용자들은 telnet으로 원격접속을 한다. 하지만 telnet은 전혀 통신내용이 보호되지 않기 때문에, 키 생성과정이 외부에 노출될 수 도 있다.

둘째, 생성된 identity를 로컬 컴퓨터로 가져오는 방법에 관한 문제이다. 보안상 허점이 많은 ftp를 이용하여 다운받게 되면 무결성을 보장받을 수 없다.

셋째, 로컬 컴퓨터에 저장된 identity 키는 누구라도 사용할 수 있다. 가능한 찾기 어려운 곳에 숨겨두는 방법이외의 조금 더 안전한 방법이 필요하다.

넷째, 그림 2에서 보는 바와 같이 SSH 인증은 먼저 호스트 공개키를 이용한 호스트인증이 이루어진다. 이 호스트 공개키는 자동으로 로컬컴퓨터에 known_hosts라는 이름으로 만들어지는데 이 파일 역시 누구라도 접근할 수 있다.

다섯째, TTSSH는 SSH 버전1.5까지만 지원하므로 버전2를 사용할 수 없다. OpenSSH가 버전1보다 버전2가 더 많은 암호 알고리즘을 제공한다는 것을 감안한다면 TTSSH 제품이 다른 제품에 비하여 보안수준이 낮다.

IV. 해결방안

OpenSSH가 완전한 보안을 제공하지는 않지만 현재로서는 최선의 대책이라고 할 수 있다. 하지만 윈도우사용자가 OpenSSH에 원격접속하기 위하여 사용하는 소트프웨어인 TTSSH는 살펴본 바와 같이 몇가지 문제점을 안고 있다. 그 해결책을 다음과 같이 제시한다.

첫째, 사용자인증용 키를 생성할 때 -f옵션을 붙여 identity 이외의 다른 이름으로 생성하여 쉽게 찾을 수 없도록 만든다.

둘째, identity 파일을 플로피디스켓에 별도 보관한다. 디스켓을 들고 다녀야 하는 번거러움이 있지만 identity 파일을 보호하기 위한 훨씬 더 안전한 방법이다.

셋째, 리모트 컴퓨터(리눅스컴퓨터)에서 만든 identity 파일을 사용자가 보안을 유지하면서 로컬컴퓨터로 가져올 수 있는 방법은 없다. 따라서 관리자가 각 사용자의 identity 파일을 디스켓에 담아 우편으로 전송하는 방법이 가장 안전하다.

하지만 이러한 방법보다 가장 근본적인 해결책은 사용자인증용 키를 클라이언트에서 생성하는 것이다. 즉, 키 생성 기능이 없는 TTSSH 1.5.4 버전으로서는 만족할만한 해결책을 찾을 수 없다.

그래서 One-Time Password(OTP)의 개념을 도입할 것을 제안한다. 윈도우 사용자가 OTP를 이용하여 직접 사용자 인증키를 생성하여 원격접속이 이루어진다면, 기존의 방식보다 더 안전하게 이용할 수 있을 것으로 생각된다. 그림 3은 S/Key 방식을 이용한 OTP 프로그램 중 하나이며, 그림 4는 OTP개념을 도입한 업그레이드 된 SSH의 인증구조이다.

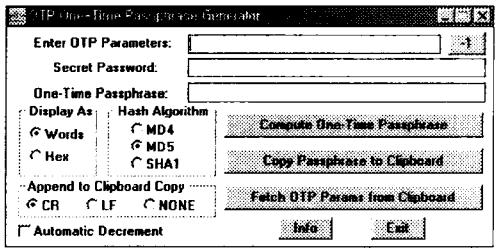

그림 3. S/Key 방식의 OTP

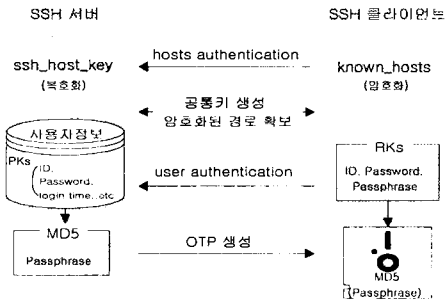

그림 4. OTP를 적용한 SSH 인증구조

V. 결 론

OpenSSH는 원격접속을 위한 제품 중에서 가장 신뢰할 만하다. 물론 완벽한 보안이 이루어질 수는 없지만 현재로서는 최선의 선택임은 틀림없다. OpenSSH는 무료로 사용할 수 있는 SSH 제품으로 SSH 버전1과 버전2 모두 지원한다. 윈도우 사용자가 OpenSSH에 원격접속하기 위한 제품도 많이 출시되어 있다. 그 중에서 TTSSH는 무료로 사용 가능한 대표적인 제품으로 버전1.5까지만 지원한다.

하지만 TTSSH도 문제점이 존재하며, 이런 문제점들을 해결하기 위해서는 누구보다 서버 관리자가 부지런해야 한다. 그러나 이를 해결하기 위한 가장 근본적인 해결책은 OTP를 도입한 SSH 인증구조를 TTSSH에 적용하는 것이다. 따라서 향후 TTSSH의 오픈 소스를 이용하여 TTSSH의 인증구조를 변경하여 연구를 계속 하여야 할 것이다.

참고문헌

[1] 박창섭, 암호이론과 보안, 대영사, P45~P57, 1999

[2] 리눅스매거진편집부, SSH, Linux magazine 7월호, P78, 2001

[3] Anonymous, Linux Security, 인포북, P322, 1999

[4] Steve Shah, LINUX Administration : A Beginner's Guide 2/e, scitech, P461, 2001

[5] Robert O'Callahan, TTSSH:An SSH Extension to Teraterm Version 1.5, http://www.zipworld.com.au/~roca/ttssh.html