SSH Howto

임은재
eunjea@kldp.org

SSH Howto

지은이 임은재

펴냄 2002/02/28

이 문서는 ssh 서버, 클라이언트의 설정, 사용법에 관한 문서이다.

고친 과정

고침 v0.4 2002년 02월 28일 고친이 임은재
smtp 포트 포워딩 수정
고침 v0.3 2001년 11월 9일 고친이 임은재
http proxy, sgml 오류 수정
고침 v0.2 2001년 11월 3일 고친이 임은재
fetchmail 과 imap 설정 추가
고침 v0.1 2001년 5월 21일 고친이 임은재
첫번째 문서

차례

1. SSH가 무엇이며, 어디서 구할수 있나? **4**
1.1. 왜 SSH를 사용해야만 할까? 4
1.2. 어디서 구하나? 4
2. 클라이언트 사용법 **5**
2.1. 기본적인 사용 방법 5
2.2. 인증키 사용하기 5
2.3. ssh를 이용한 파일 복사 6
2.4. ssh 터널링 7
2.5. 설정 파일 9
3. 서버 운영 **10**
3.1. 설치 10
3.2. 서버 설정 10
4. 저작권, 관련/참고 문서 **11**
4.1. 저작권 11
4.2. 관련/참고 문서 11

1장. SSH가 무엇이며, 어디서 구할수 있나?

SSH (Secure SHell)은 말 그대로 보안 로그인 쉘이다.

전통적인 ftp, pop, telnet 같은 서비스들은 잘 알려진 대로 매우 보안에 취약하다. 이런 암호화되지 않은 인증 방법은 당신의 암호가 그대로 노출될수도 있다.

ssh를 통한 모든 데이타는 암호화되며, 트래픽은 압축되어 더 빠른 전송 효율을 얻을수 있다. 또한 기존의 ftp,pop 같은 안전하지 못한 서비스들을 위한 "터널"까지 지원한다.

sshd 서버를 운영하지 않는 서버 관리자는 보안에 전혀 관심이 없는 사람이다.

1.1. 왜 SSH를 사용해야만 할까?

다음글은 www.openssh.org (http://www.openssh.com/)의 OpenSSH FAQ중에서 인용하였다.

- 강력한 보안
- 프라이버시 보호. 모든 통신은 자동으로 그리고 투명하게 암호화된다.
- 안전한 X11 세션. 원격 서버에 DISPLAY 변수를 자동으로 설정하고 모든 X11 연결을 보안 채널을 통해서 포워딩한다.
- TCP/IP 포트를 양 방향에서 다른 포트로 자유롭게 포워딩할수 있다.

1.2. 어디서 구하나?

리눅스에서 사용할수 있는 ssh 는 두가지가 존재한다. ssh의 원 제작처인 www.ssh.com (http://www.ssh.com/) (핀란드 회사) 와 BSD licence(사실 100% BSD licence는 아니다.)의 www.openssh.org (http://www.openssh.com/)가 그것이다.

나는 openSSH를 사용하며 이 문서도 openSSH를 기준으로 설명할 것이다. openSSH는 하나의 클라이언트/서버에서 ssh1,ssh2 프로토콜을 모두 지원한다.

ssh는 이미 당신의 배포본에 이미 포함되어 있을지도 모른다. 직접 컴파일 하여 사용하고 싶다면 www.openssh.org (http://www.openssh.com/portable.html)에서 소스를 받아 설치한다.

그외 ssh를 사용하기 위해 꼭 필요한 openssl 라이브러리는 www.openssl.org (http://www.openssl.org)에서 구할수 있다.

ftp.koru.org (ftp://ftp.koru.org/pub/rpm) 에는 필자가 최신 버젼의 openSSH 와 openssl을 rpm 빌드해놓은 것과 소스 rpm을 찾을수 있다.

2장. 클라이언트 사용법

이 장에서는 ssh 서버에 접속하는 ssh 클라이언트의 사용방법에 대해 알아본다.

2.1. 기본적인 사용 방법

openSSH 클라이언트는 ssh1,ssh2 프로토콜을 모두 지원하므로, 서버가 지원하는 ssh 프로토콜에 상관없이 접속할수 있다. 예를 들어, 접속할 ssh서버가 gate.eunjea.org 이고 계정명이 silver 라면

```
[foo@home silver]$ ssh -l silver gate.eunjea.org
```

또는

```
[foo@home silver]$ ssh silver@gate.eunjea.org
```

이제 다음과 같은 메세지와 함께 접속이 진행될 것이다.

```
The authenticity of host 'gate.eunjea.org (192.168.1.1)' can't be estab-
lished.
RSA1 key fingerprint is e3:56:xx:b4:19:7e:xx:b1:7e:cd:xx:fe:5e:5b:17:66.
Are you sure you want to continue connecting (yes/no)?
```

위 메세지는 ssh로 해당 서버에 처음 접속할때만 나오는 메세지이며, 접속할 서버의 호스트키가 ~/.ssh/known_hosts (ssh2의 경우 known_hosts2) 파일에 저장된다. yes로 대답해주면, 다음과 같이 계정 암호를 물어오고, 이제 텔넷과 동일한 작업을 할수 있다.

```
Warning: Perma-
nently added 'gate.eunjea.org,192.168.1.1' (RSA1) to the list of known hosts.
silver@gate.eunjea.org's password:
```

2.2. 인증키 사용하기

인증키를 사용하는 것은 로그인 할때마다 암호를 직접 입력하는 것보다 더욱 안전하며, 하나의 암호로 여러 ssh서버에 접속할수 있는등의 장점을 가진다.

- 인증키 만들기

  인증키는 ssh-keygen로 만든다.

  ```
  [foo@home silver]$ ssh-keygen
  Generating public/private rsa1 key pair.
  Enter file in which to save the key (/home/foo/.ssh/identity):
  ```

키가 저장될 곳과 이름을 물어 오는데 디폴트로 그냥 엔터를 치고 넘어가면, 다음과 같이 인증키 암호를 물어온다. 원하는 암호를 두번 입력해주면 키가 생성된다.

```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/foo/.ssh/identity.
Your public key has been saved in /home/foo/.ssh/identity.pub.
The key fingerprint is:
88:0a:aa:xx:41:03:xx:62:94:fe:xx:d5:31:76:5b:b0 foo@home.eunjea.org
[foo@home silver]$
```

인증키 생성시 인증키 암호를 공백으로 (passphrase 를 물어올때 그냥 엔터를 치면 된다) 만들수도 있는데, 이것은 ssh 접속시 암호를 입력하지 않아도 그냥 접속이 되므로 편리할수는 있으나, 만약 당신의 인증키가 어떠한 경로로든 유출되었을 경우를 생각 해보면 피해야 할 것이다. 그리고 ssh-add와 ssh-agent를 사용하여 접속시마다 인증키 암호를 입력하지 않는 방법이 있다.

- 퍼블릭 키 사용하기

~/.ssh/ 안에 한쌍의 키(identity 와 identity.pub)가 생성되어 있을것이다. 이제 이 인증키를 접속할 리모트 서버들의 ~/.ssh/ 에 authorized_keys 라는 이름으로 복사해준다.

```
[foo@home silver]$ scp ~/.ssh/identity.pub sil-
ver@gate.eunjea.org:.ssh/authorized_keys
```

이제 ssh 접속을 진행 해보면 계정암호가 아닌 인증키 암호를 물어볼 것이다. 만약 계정 암호를 물어본다면 authorized_keys의 퍼미션이 잘못되어 있는 것이므로, 일단 접속후 chmod 644 .ssh/authorized_keys 해준다.

일반적으로 아무런 옵션을 주지 않고 인증키를 만들었다면 rsa1 방식의 ssh1 프로토콜의 사용을 의미한다. 서버가 ssh2 프로토콜을 지원한다면 ssh-keygen -t rsa (또는 dsa)의 옵션을 주고 인증키를 생성한다.

openSSH는 두가지 프로토콜의 인증키를 동시에 쓰는것이 가능한데, 만들어진 퍼블릭 키(예를 들어 ssh-keygen -t rsa 로 만들었다면, id_rsa.pub) 를 위와 같은 방법으로 리모트 서버에 ~/.ssh/authorized_keys2 에 저장하면 된다.

- 인증키를 메모리에 상주 시키기

다음 방법으로 인증키를 메모리에 기억시켜 두면 처음 한번만 인증키 암호를 입력하면 다음부터는 암호를 입력하지 않아도 같은 인증키를 사용하는 모든 서버들에 접속할수 있다.

```
[foo@home silver]$ eval $(ssh-agent) [Enter]
다음과 같은 메세지를 보여줄 것이다.
Agent pid 31234
이제 ssh-add 를 입력하면
Identity added: /home/silver/.ssh/identity (silver@home.eunjea.org)
```

이제 인증키를 복사해둔 ssh서버에 접속하면 이 세션에서는 더 이상 암호를 묻지 않을 것이다.

서버가 지원한다면 되도록 SSH2 프로토콜을 사용하도록 한다. SSH2는 SSH1과는 전혀 다른 프로토콜이며 더욱 안전하고, 성능이 좋다.

2.3. ssh를 이용한 파일 복사

- scp

  위에서 인증키를 리모트 서버에 복사할때 사용한 scp에 대해서 알아보자

  예를 들어, 복사하려는 파일명이 'dumb' 라고 하고 접속하려는 원격 서버의 주소는 www.foobar.com, 당신의 쉘 계정은 babo 라고 한다면

  dumb 파일을 www.foobar.com 의 babo 계정 홈 디렉토리에 복사하기

  ```
  [foo@home silver]$ scp dumb babo@www.foobar.com:.
  ```

  www.foobar.com 의 babo 계정 홈 디렉토리에 있는 dumb 파일을 로컬로 복사하기

  ```
  [foo@home silver]$ scp babo@www.foobar.com:dumb .
  ```

  만약 ~/.ssh/config 파일에 다음과 같이 www.foobar.com 의 계정을 설정해 놓았다면,

  ```
  Host *fbc
  HostName www.foobar.com
  User babo
  ForwardAgent yes
  ```

  다음과 같이 더 간단하게 할수 있다.

  ```
  [foo@home silver]$ scp dumb fbc:.
  ```

  또한 scp 는 -r 옵션도 가지고 있는데 이것은 디렉토리를 통채로 복사 할때 사용한다. 예를 들어 test/ 디렉토리안의 모든 파일과 하위 디렉토리를 서버 계정의 www 디렉토리 안에 복사 하려면 다음과 같이 한다.

  ```
  [foo@home silver]$ scp -r test/ babo@www.foobar.com::www/
  ```

- sftp

  sftp는 ssh하에서 전통적인 ftp 환경을 제공하며, 리모트상의 프로그램을 실행시킬수도 있다.

  openSSH 클라이언트 패키지에는 sftp가 포함되어 있다.

2.4. ssh 터널링

ssh 터널링이란 ssh 접속을 다른 프로그램이 사용할수 있도록 port forwarding해주는 것을 말한다. 이 ssh 터널링을 이용해 암호화 접속을 사용하지 않는 네트워크 접속을 보다 안전하게 사용할수 있다.

- POP

  fetchmail을 사용하면 간단하게 ssh 터널안에서의 pop 메일 긁어오기를 구현할수 있다.

  .fetchmailrc 설정예

  ```
  poll localhost with protocol pop3 and port 11110:
       preconnect "ssh -C -f 계정@메일서버.com -L 11110:메일서
  버.com:110 sleep 5"
       password xxxxx
  ```

  자세한 문서는 : SSH 를 이용한 보안 POP (http://kltp.kldp.org/eunjea/securepop/Secure-POP+SSH.html)

  원격 계정의 이메일을 아예 복사해오는 방법도 생각해 볼수 있다. (Compressed TCP/IP-Sessions using SSH-like tools (http://www.linuxdoc.org/HOWTO/mini/Compressed-TCP-3.html) 참조)

- IMAP

  ssh 터널링과 fetchmail을 사용해서 imap 서버로부터 메일을 가져오려면, 다음과 같은 .fetchmailrc를 만들어 사용하면 된다.

  ```
  poll 메일서버.com with proto imap:
       plugin "ssh %h /usr/sbin/imapd" auth ssh;
       user babo is babo here
  ```

- SMTP

  역시 같은 문서에서 SSH 접속을 이용한 SMTP 사용법을 제시했는데 방법은 다음과 같이 간단하다.

  ```
  ssh -C -l loginid mailserver -L2525:mailserver:25
  ```

  후에 메일 클라이언트를 localhost port 2525 를 통해 메일을 보내도록 하면 된다. 예를 들어 pine을 사용한다면, .pinerc의 smtp-server=localhost:2525 와 같이 해주면 되겠다.

  ssh 윈도우 클라이언트인 SecureCRT를 사용해도 가능한데 Session Option -> Connection -> Hostname -> Advanced 탭을 선택해서, 같은 요령으로 사용할 로컬 포트와 원격 호스트 이름, 포워딩할 원격 포트를 선택한다. ssh 접속 후에는 OE의 경우 SMTP 서버를 127.0.0.1 로 지정하고 사용할 포트만 위에서 선택한 로컬 포트로 지정하면 된다. POP 포트도 같은 방법으로 사용 가능 하다.

  SSH를 이용한 SMTP는 몇가지 장점을 가지는데 네트워크 트래픽의 감소와 계정 사용자만이 SMTP 서버를 사용할수 있으므로 함부로 릴레이를 열어놓지 않아도 된다는데 의미가 있겠다.

- Webmin

  Webmin (http://www.webmin.com/webmin/)는 웹상에서 브라우저로 서버 관리를 하는 툴이며, 당연히 보안에 민감하다.

  webmin은 일반적으로 10000 포트를 사용하므로 다음과 같이 ssh 접속을 연다.

```
ssh -f -l [원격 유저] [원격 서버] -L 1234:[원격 서
버]:10000 tail -f /etc/motd
```

이제 브라우저에서 http://localhost:1234 로 접속할수 있다.

- X

  리모트 서버상의 X 어플리케이션들을 실행하고자 한다면 계정 홈 디렉토리의 ~/.ssh/environment 파일을 만들고 다음과 같은 내용을 넣어준다.

  ```
  XAUTHORITY=/home/계정 이름/.Xauthority
  ```

  이제 로그아웃한후에 ssh를 다음과 같이 실행해본다. (계정 이름이 silver이고 서버는 gate.eunjea.org 라고 한다면)

  ```
  ssh -f -X -l silver gate.eunjea.org xterm
  ```

  이제 xterm 이 로컬의 X에서 실행될 것이다. 다른 X 어플리케이션들도 이와 같이 실행시킬수 있다.

2.5. 설정 파일

ssh 설정 파일은 ~/.ssh/config 파일 이다. 또는 전체 유저의 설정파일은 /etc/ssh/ssh_config 로 설정할수 있다.

다음은 내가 사용하는 설정 파일의 일부분이다. Host 지시자를 사용하여 접속할 서버마다 다른 옵션을 사용할수 있다.

```
# *.eunjea.org 도메인을 가진 서버에 접속할때는 SSH2 프로토콜을 사용한다.
Host *.eunjea.org
Protocol 2

# koru.org 에 접속할때는 SSH2 와 압축 옵션을 사용한다.
Host koru.org
Protocol 2
Compression yes
CompressionLevel 9

# kldp.org에 접속할때는 SSH1 프로토콜을 사용하고
# Cipher는 blowfish, 압축을 켠다.
Host kldp.org
Protocol 1
Cipher blowfish
Compression yes
```

그외 중요한 옵션으로는 CheckHostIP 가 있는데 이것은 접속할때 마다 리모트 서버의 IP 주소를 known_hosts 파일과 대조해본다. 이것은 DNS spoofing에 의해 호스트키의 변경을 알수 있는 옵션이다. 디폴트는 yes이다.

이외에도 많은 옵션들이 있는데 ssh의 man 페이지를 참고하라.

3장. 서버 운영

3.1. 설치

서버는 간단하게 패키지를 설치하거나 직접 소스를 사용하여 빌드할 경우 다음 옵션정도로 빌드하면 무리가 없겠다. 만약 tcp wrapper 를 사용할 것이라면 configure 옵션에 --with-tcp-wrappers 를 사용한다.

```
configure \
--with-pam \
--with-ipv4-default \
--with-rsh=/usr/bin/rsh \
--with-default-path=/bin:/usr/bin:/usr/local/bin:/usr/X11R6/bin
```

3.2. 서버 설정

서버 설치가 끝난후 설정 파일(/etc/ssh/sshd_config)의 옵션들을 살펴보자. 대부분의 경우 기본 설정파일 그대로 사용하여도 좋지만, 특정 그룹이나 유저들에게만 로그인을 허용하도록 할 경우 다음 지시자를 사용할수 있다.

- AllowGroups

  ssh 로그인을 해당 그룹으로 제한한다. 각각의 그룹명은 공백으로 구분한다. 와일드 카드(* 와 ?)를 사용할수 있다.

- AllowUsers

  ssh 로그인을 해당 유저로 제한한다. 사용법은 AllowGroups과 같다.

- DenyGroups

  AllowGroups의 반대 역할을 한다. 지정된 그룹은 로그인이 거부된다.

- DenyUsers

  AllowUsers의 반대 역할을 한다. 지정된 사용자는 로그인이 거부된다.

이외 사용자들의 sftp 사용을 허용하려면 다음과 같은 라인이 있는지 확인한다.

```
Subsystem    sftp   /usr/lib/openssh/sftp-server
```

이외 옵션들은 sshd 의 man 페이지를 참고한다.

4장. 저작권, 관련/참고 문서

4.1. 저작권

4.2. 관련/참고 문서

- http://www.linuxdoc.org/HOWTO/mini/Compressed-TCP.html
- http://www.mandrakeuser.org/docs/secure/sssh.html