1.113.7 Set up secure shell (OpenSSH) Weight 4

Linux Professional Institute Certification — 102

Geoffrey Robertson `ge@ffrey.com` Nick Urbanik
`nicku@nicku.org`

2005 July

Outline

- Context
- Objectives
- What is SSH?
- Versions of SSH
- SSH commands and utilities
  - ssh
  - scp
  - sftp
- Advanced usage
  - Redirecting stdin and stdout
  - X forwarding
  - Port Forwarding
- Authenticating with User Keys
  - ssh-agent
- SSH from Windows
- Other frontends
- License Of This Document

Topic 113 Networking Services [24]

Where we are up to

1.113.1 Configure and manage inetd, xinetd, and related services [4]

1.113.2 Operate and perform basic configuration of sendmail [4]

1.113.3 Operate and perform basic configuration of Apache [4]

1.113.4 Properly manage the NFS, smb, and nmb daemons [4]

1.113.5 Setup and configure basic DNS services [4]

1.113.7 **Set up secure shell (OpenSSH) [4]**

Description of Objective

1.113.7 Set up secure shell (OpenSSH)

The candidate should be able to obtain and configure OpenSSH. This objective includes basic OpenSSH installation and troubleshooting, as well as configuring `sshd` to start at system boot.

Key files, terms, and utilities include:

`/etc/hosts.allow` — TCP Wrappers file: sshd is linked against TCP Wrappers

`/etc/hosts.deny` — TCP Wrappers file: sshd is linked against TCP Wrappers

`/etc/nologin` — if present, only `root` can log in; others shown content of this file

`/etc/ssh/sshd_config` — `sshd` configuration

`/etc/ssh_known_hosts` — holds public host keys of hosts to verify their identity when connect to them

`/etc/sshrc` — global login script for SSH

`sshd` — SSH server program

`ssh-keygen` — Generates public/provate key pairs for host keys, user keys

“Secure SHell”

A functional replacement of the ancient `rsh` command, except with encryption and authentication.

Versions

Commercial SSH Finnish company. Original authors of SSH.

OpenSSH Split from last free version of commercial SSH. Development led by OpenBSD team.

Draft “secsh” RFC.
Alternative implementations exist (Putty, `Net::SSH::Perl`, etc.)

Commands

`ssh`	Run a shell command on a remote host
`sshd`	SSH server daemon
`scp`	Copy files using SSH
`sftp`	An ftp-like interface into `scp`
`ssh-keygen`	Generate an SSH key pair
`ssh-agent`, `ssh-add`	SSH key forwarding

Commands

ssh

```
ssh [options] host [command]
```

Run a shell command on a remote host.
Acts like a normal shell command. ie: STDIN, STDOUT work as normal.
Without a *command*, ssh runs an interactive login.

Commands

scp

```
scp user@host:path/file user2@host2:path/file2
```

Copy a file over ssh.
user defaults to current login, *user@host* maybe omitted for local files, *path* is relative to $HOME

Commands

sftp

```
sftp user@host:path
```

ftp-like command line interface to scp.
Only provided with more recent ssh versions.

Redirecting stdin and stdout

Advanced Usage

Remember that STDIN and STDOUT still work as normal (unlike `telnet`):

```
ssh remote tar zcf - /remotepath > localfile.tar.gz
```

Advanced Usage

X Forwarding

```
ssh -X host
```

Login to *host* and “forward” X11 connections back to the local Xserver.
A “fake” $DISPLAY and `xauth` environment are created, and the X11 data is passed back over the same SSH connection.

Advanced Usage

X Forwarding *ctd...*

- Forwarding X over SSH is secure and easy, but slower than not doing it.
- On a local LAN, the encryption is probably unnecessary—use normal X methods such as `rstart` instead (`rstart` can use ssh for authentication anyway).
- Specialised X11 caching methods (eg: LBX) can get better performance than ssh compression over slow links.
- There are concerns over connecting to a hostile remote machine and forwarding X back again, so don’t forward X by default. A hostile remote site may forward damaging commands back down the link to your X server (ie your screen and keyboard).

Advanced Usage

Port Forwarding

Arbitrary ports can also be forwarded over the SSH connection, to add security to other protocols (or bypass poor firewall policies…)

```
# .fetchmailrc example
poll localhost protocol pop3 port 11110:
  preconnect "ssh -C -f user@host.com \
    -L 11110:host.com:110 sleep 10"
```

Advanced Usage

SSH Keys

- Public key authentication. More secure alternative to password login.
- Generate a public/private “key pair” with `ssh-keygen`.
- **Keep the private key secret.**
- Append the public key into your (remote) `~/.ssh/authorized_keys` to allow access.

  ```
  $ cat identity.pub »
  ~/.ssh/authorized_keys ↩
  ```
- More powerful automation (scripting) possibilities.

Advanced Usage

SSH Authentication Agent

- `ssh-agent` allows key information to be “forwarded” between its child processes—even across nested ssh sessions.
- Start `ssh-agent` in your X-session or login scripts, and run `ssh-add` to add keys.
- `ssh-askpass` is (basically) an X11 version of `ssh-add`.

Advanced Usage

SSH from win32

Putty Includes command line `pscp.exe` `scp` clone too.
`http://www.chiark.greenend.org.uk/~sgtatham/putty/`

Winscp Graphical SCP client.
`http://winscp.vse.cz/eng`

Advanced Usage

Other “frontends”

KDE kio_fish Provides `ssh://` konquerer paths.

tramp.el Transparent access to remote files for emacs.

rsh-compatible Anything that can use rsh (eg: CVS)

Topics Covered

Context

Objectives

What is SSH?

Versions of SSH

SSH commands and utilities

- `ssh`
- `scp`
- sftp

Advanced usage

- Redirecting stdin and stdout
- X forwarding
- Port Forwarding

Authenticating with User Keys

- ssh-agent

SSH from Windows

Other frontends

License Of This Document

License Of This Document