Advanced OpenSSH

Basic Usage
Authentication Methods
Keys and Agents
Remote X Windows
Tunnels and Port Forwarding
Client Configuration
Server Configuration
Secure access to remote command-line
Replaces telnet, rlogin, and rsh
Requires support on remote end
Remote shell on a firewall or “jump host”

```
ssh hostname
```

If at first you don’t succeed...

```
ssh -v user@hostname
```
Escape Sequence

ssh -e <char> user@hostname

Default escape character is the tilde: ~

~. - terminate connection
~^Z - suspend ssh
~# - list forwarded connections
~& - background ssh
~? - this message
~~ - send the escape character “upstream” by typing it twice
Basic PuTTY

PuTTY Configuration

Category:
- Session
  - Logging
- Terminal
  - Keyboard
  - Bell
  - Features
- Window
  - Appearance
  - Behaviour
  - Translation
  - Selection
  - Colours
- Connection
  - Data
  - Proxy
  - Telnet
  - Rlogin
  - SSH
  - Serial

Basic options for your PuTTY session

Specify the destination you want to connect to
Host Name (or IP address): carlos@oso.audioangler.com
Port: 22
Connection type:
- Raw
- Telnet
- Rlogin
- SSH
- Serial

Load, save or delete a stored session
Saved Sessions:
- Oso
- Default Settings

Close window on exit:
- Always
- Never
- Only on clean exit
Authentication Methods

Password *
Public Key Exchange *
GSSAPI (Kerberos)
Host equivalence
Challenge/Response

Remote host handles authentication
New “Key Pair” Creation:

```
ssh-keygen -t rsa -b 2048
```

* Choose a memorable passphrase!!

- Private key is used by client
  
  Default is ~/.ssh/id_rsa

- Public key is copied to remote servers
  
  Default is ~/.ssh/id_rsa.pub
Public Keys

Public Keys in Linux and UNIX:

~/.ssh/known_hosts

and

~/.ssh/authorized_keys

are checked by sshd on incoming connections

- Client presents a checksum based on private key
- Server uses public key to validate checksum
Authentication Agents

Many different ways of doing agent-based auth

*ssh-agent* in Linux and UNIX

Pageant for PuTTY on Windows

Some keychain facilities provide ssh agents

Can use one or many keys for different uses

Even `sudo` supports it now: `pam_ssh_agent_auth`
Linux Agent Authentication

The agent itself is a background process, invoked as

```
% eval `ssh-agent -s`
```

Runs ssh-agent, which forks and outputs shell commands to set up the environment for `ssh -A`

```
ssh-add
to add key(s)

ssh-add -l to list active keys
```

Then, subsequent ssh authentication is relegated back “up the chain” to the originating ssh client
PuTTY and WinSCP both use Pageant for agent auth.

Pageant idles in the system tray until a private key is loaded.

You can load multiple keys, but they have to be .ppk.

Puttygen is used to import an OpenSSH rsa key.

It’s harder to go the other way!
Port Forwarding

SSH “tunnel” encapsulates traffic
Encryption
Compression
Remote X Windows
VNC example (Linux and putty)
Database server example (cmd line)
X11 Forwarding

X11 support is built into OpenSSH

```
ssh -X user@host
```

The client’s DISPLAY environment variable is passed along to remote shell

ssh provides a “proxy” X server to forward remote display traffic through the tunnel

ssh also creates an Xauthority cookie and validate that the forwarded traffic uses this token, not the “real” Xauth cookie
SSH Tunnel for VNC Remote

Laptop vncviewer connecting to Firewalled Server

laptop% ssh -L 5901:localhost:5901 user@dbserv

dbserv% vncserver :1 -localhost

laptop% vncviewer localhost:5901 &

Note: TigerVNC's vncviewer includes ssh support, -via flag
PuTTY Tunnel for VNC

Category:
- Keyboard
- Bell
- Features

Window
- Appearance
- Behaviour
- Translation
- Selection
- Colours

Connection
- Data
- Proxy
- Telnet
- Rlogin

SSH
- Kex
- Auth
- TTY
- X11
- Tunnels

Options controlling SSH port forwarding

Port forwarding
- Local ports accept connections from other hosts
- Remote ports do the same (SSH-2 only)

Forwarded ports:

Add new forwarded port:
- Source port: 5901
- Destination: dbserver:5901
  - Local
  - Remote
  - Dynamic
  - Auto
  - IPv4
  - IPv6
Local Tunnel for DB

Example: Laptop to DB Server via port-forwarding

```
ssh -L 3306:localhost:3306 dbuser@dbserv
```

Now we make a “local” connection to mysql

```
mysql -h localhost -u dbuser dbname
```
Remote Tunnel for DB

Ex. 1: Jump host to DB Server
ssh -L 3306:localhost:3306 dbuser@dbserv
mysql -h localhost -u dbuser dbname

Ex. 2: Remote Laptop to DB Server
ssh -L 3306:dbserv:3306 user@jumphost
mysql -h localhost -u dbuser dbname
scp and sftp

Secure remote copy over an ssh tunnel
Replaces rcp and ftp
Supports compression, as in, ssh -C
scp files hostname:remote-dir
scp -rC dir hostname:remote-dir

Many GUI clients enable drag-n-drop
WinSCP for Windows, Cyberduck for Mac
Client Configuration

A great many ssh client options can be set in

~/.ssh/options

or /etc/ssh/ssh_config

Examples:

ForwardAgent yes
KeepAlive yes
ServerAliveInterval 12
RSAAuthentication yes
GSSAPIAuthentication no
ForwardX11 no
Server Configuration

Again, many sshd options can be set in

/etc/ssh/sshd_config

Examples:

Protocol 2
PermitRootLogin no
PasswordAuthentication yes
GSSAPIAuthentication no
X11Forwarding no
TCPKeepAlive yes
References

- http://www.openssh.org
  man pages, RFC’s, history, etc.

- SSH Mastery: OpenSSH, PuTTY, Tunnels, and Keys
  by Michael W Lucas

Time For Your Questions

Carlos Aguayo
caguayo@gmail.com