SSH

the Secure Shell

Simon Law

History

- Network computing prevalent
- Need for remote logins
- Tools were developed for the early networks
  - rsh
  - rexec
  - rcp
  - rlogin
  - telnet

Problems

- Tools first appeared in 4.2 BSD
- Assumed the network was secure
- Trade-off at the time

Birth of SSH

- Tatu Ylönen at Helsinki U. of Technology
- First implementation in 1995
- Version 1.2.12 was the last free release
- Tatu co-founded SSH Communications Security
- Two commercial implementations:
  - SSH Communications Security
  - Datafellows F-Secure

OpenSSH

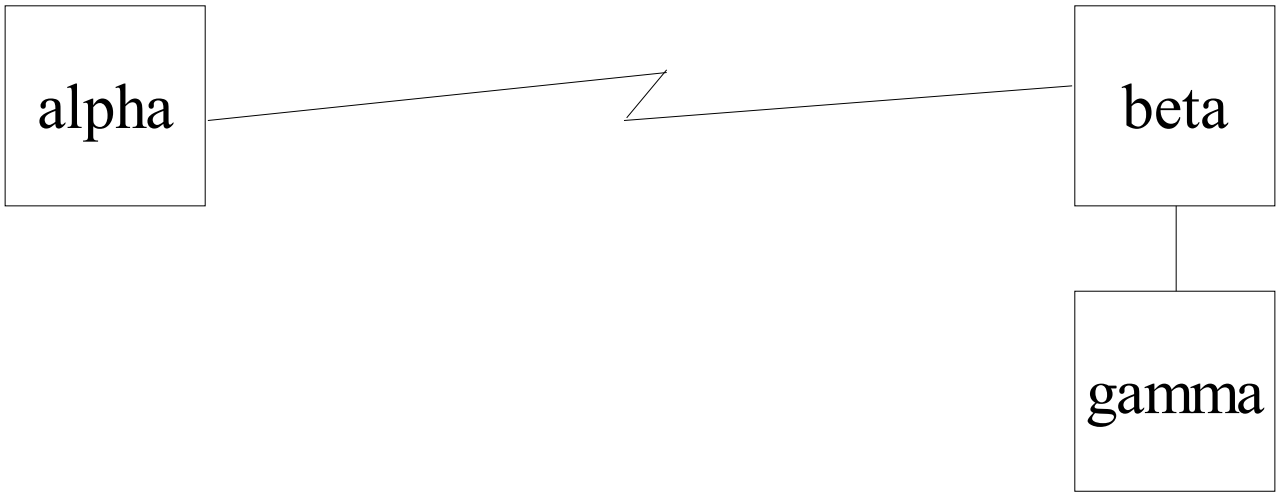
- OpenBSD team forked OpenSSH
- IETF published SSH2 protocol
- Complete working implementation by 2000
- http://www.openssh.com

What is SSH?

- Protocol suite
- Strong Encryption
- Strong Authentication
- Data Compression

Network

- Three machines: alpha, beta, gamma



Secure shell

```
alpha$ ssh beta
jdoe@beta's password:
beta$

alpha$ ssh -l root beta
root@beta's password:
beta#

alpha$ ssh root@beta
root@beta's password:
beta#
```

Secure execution

```
alpha$ ssh beta hostname
jdoe@beta's password:
beta.some.network.example
alpha$

alpha$ ssh beta ls -l '|' wc -l
jdoe@beta's password:
      57
alpha$
```

Secure copy

```
alpha$ scp myfile.txt beta:
jdoe@beta's password:
myfile.txt 100% |**********************|  123 00:02
alpha$

alpha$ scp beta:myfile.txt .
jdoe@beta's password:
myfile.txt 100% |**********************|  123 00:02
alpha$
```

Secure File Transfer Protocol

```
alpha$ sftp beta
jdoe@beta's password:
sftp>
```

Compression

```
alpha$ ssh -C beta
jdoe@beta's password:
beta$
```

X11 forwarding

```
alpha$ ssh -X beta
jdoe@beta's password:
beta$ xterm

alpha$ echo $DISPLAY
:0.0
alpha$ ssh -X beta
jdoe@beta's password:
beta$ echo $DISPLAY
localhost:10.0
beta$
```

Backgrounding

```
alpha$ ssh -X -f beta xterm
jdoe@beta's password:
alpha$
```

- Recommended way to start X11 applications
- Over a slow link:

```
alpha$ ssh -C -X -f beta xterm
jdoe@beta's password:
alpha$
```

Chaining SSH

```
alpha$ ssh -X -f -t beta \
          ssh -X gamma xterm
jdoe@beta's password:
jdoe@gamma's password:
alpha$
```

Local forwarding

- Forward a local port to a remote system

```
alpha$ ssh -L 8080:localhost:80 beta
jdoe@beta's password:
beta$
```

- Now we can connect to localhost, port 8080; to get to beta, port 80

Local forwarding

- We can also redirect other machines

```
alpha$ ssh -L 8080:gamma:80 beta
jdoe@beta's password:
beta$
```

- Now we can connect to localhost, port 8080; in order to reach gamma, port 80.

Remote forwarding

- You can do the reverse as well

```
alpha$ ssh -R 2222:localhost:22 beta
jdoe@beta's password:
beta$
```

- Now, whenever anyone connects to beta, port 2222; he actually gets to alpha, port 22.

SOCKS proxying

- OpenSSH can also be a SOCKS4 proxy

```
alpha$ ssh -D 1080 beta
jdoe@beta's password:
beta$
```

- Now, we can use any SOCKS4 aware application, as if he were connecting from beta.

Public key authentication

- Typing passwords can be tedious
- You can create a public key infrastructure through SSH to authenticate seamlessly.
- SSH1: ~/.ssh/identity ~/.ssh/identity.pub
- SSH2: ~/.ssh/id_rsa ~/.ssh/id_rsa.pub
  ~/.ssh/id_dsa ~/.ssh/id_dsa.pub

Generating keys

- Use the ssh-keygen(1) command

```
alpha$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/jdoe/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jdoe/.ssh/id_dsa.
Your public key has been saved in /home/jdoe/.ssh/id_dsa.pub.
The key fingerprint is:
d1:60:dc:00:5a:09:d0:34:58:f8:ca:9d:8f:cf:d2:87 jdoe@alpha
```

Authorised List

- Now you must add this public key to the authorised list.

```
alpha$ ssh beta cat '>>' ~/.ssh/authorized_keys \
            < ~/.ssh/id_dsa.pub
jdoe@beta's password:
alpha$ ssh beta
beta$
```

Questions?

- Find more information from:
  - http://www.openssh.org
  - man 1 ssh
  - man 1 ssh-keygen
  - man 1 ssh-agent
  - man 1 ssh-keyscan