Global Information Assurance Certification Paper

Using SSH2 for UNIX and Windows
Paul Koppel

According to the ssh2 man pages, "*Ssh2 (Secure Shell) is a program for logging into a remote machine and executing commands in a remote machine. It is intended to replace rlogin and rsh, and provide secure,encrypted communications between two untrusted hosts over an insecure network*". Secure Shell was developed by SSH Communications Security and is one popular method for encrypting terminal connections on the Internet. The purpose of this document is to briefly describe setting up ssh2 and showing a ssh2 session between two UNIX systems (Solaris 2.7) and and also between a PC (NT4.0) and a UNIX system. Other methods for providing a secure telnet such as authentication via Kerberos or proxy by firewall are not discussed here.

UNIX Installation and Terminal Session

Installing ssh2

The distribution was downloaded from http://www.ssh.com and installed on two UNIX systems for these tests. Installing ssh2 is straightforward – first uncompressing the distribution, running configure with flags enable-debug, and then running make, and then make install. For some UNIX systems, the software can be installed using the pkgadd command. Detailed installation instructions can be found at http://www.ssh.com.

Configure sshd2

This discussion closely follows the configuration instructions at http://www.ssh.com. According to the ssh2 man pages, *"Public key authentication is based on the use of digital signatures. Each user creates a public / private key pair for authentication purposes. The server knows the user's public key, and only the user has the private key. The filenames of private keys that are used in authentication are set in $HOME/.ssh2/identification. When the user tries to authenticate himself, the server checks $HOME/.ssh2/authorization for filenames of matching public keys and sends a challenge to the user end. The user is authenticated by signing the challenge using the private key".*

In order to use ssh2, private and public keys need to be generated. Then, the public key of a particular local client is copied to the remote host. For client brian and remote host mayday, the procedure is simple. Schematically:

allow brian to establish ssh2 session on mayday	mayday has my public key that I generated on brian and copied over	mayday has sshd2 daemon running on port 22

On brian, for example (all commands are relative to my home directory) the keys are generated using the ssh-keygen2 program that comes with the distribution:

```
brian% /usr/local/bin/ssh-keygen2
Generating 1024-bit dsa key pair
  2 Oo.oOo.oOoo.
Key generated.
1024-bit dsa, koppel@brian, Wed Nov 22 2000 16:50:21
Passphrase :
Again      :
Private key saved to /opt/home/erl/koppel/.ssh2/id_dsa_1024_a
Public key saved to /opt/home/erl/koppel/.ssh2/id_dsa_1024_a.pub
```

Running the ssh-keygen2 program creates a .ssh2 subdirectory. This only needs to be done once. If we also want to allow mayday, (now acting as a client) to communicate with brian (now acting as remote host) using ssh2, then we have schematically:

allow mayday to establish ssh2 session on brian	brian has my public key that I generated on mayday and copied over	brian has sshd2 daemon running on port 22

Even if we are not interested in mayday establishing a ssh2 session with brian, we still need the .ssh2 subdirectoy on mayday to hold brian's public key. The easiest way to do this is to run the ssh-keygen2 program on mayday:

```
mayday% /usr/local/bin/ssh-keygen2
Generating 1024-bit dsa key pair
  2 Ooo.oOo.oOo.
Key generated.
1024-bit dsa, koppel@mayday, Wed Nov 22 2000 17:12:23
Passphrase :
Again      :
Private key saved to /opt/home/erl/koppel/.ssh2/id_dsa_1024_a
Public key saved to /opt/home/erl/koppel/.ssh2/id_dsa_1024_a.pub
```

We need to define an identification file, which lists the private keys that can be used for authentication. For example:

for brian to establish ssh2 session on mayday	brian% echo "IdKey id_dsa_1024_a" > identification
for mayday to establish ssh2 session on brian	mayday% echo "IdKey id_dsa_1024_a" > identification

Copy the public key of brian, "id_dsa_1024_a.pub" to the .ssh2 directory on mayday, and call it brian.pub. Copy the public key of mayday to the .ssh2 directory on brian, and call it mayday.pub. Be careful when moving the public keys around, since they have the same name on both machines.

Next, we define two authorization files – one on brian, and the other on mayday. This file lists the public keys that are accepted for authentication on a particular host. For example:

tells brian to use mayday.pub as a valid public key for logins from mayday	brian% cat authorization Key mayday.pub
tells mayday to use brian.pub as a valid public key for logins from brian	mayday% cat authorization Key brian.pub

We have the following files in .ssh2 on brian:

```
brian% ls .ssh2
authorization     id_dsa_1024_a     identification     random_seed
hostkeys          id_dsa_1024_a.pub  mayday.pub
```

The .ssh2 directory on mayday is similar, but contains the file brian.pub.

Next, we manually start up the ssh2d daemon on each system by running the sshd2.startup shell script that is included with the distribution. Normally this script would be located in /etc/rc2.d on the host, with a soft link to /etc/init.d. Since we want to establish ssh2 in both directions we need to startup the sshd2 daemon on both systems – brian and mayday:

```
# ./sshd2.startup start
-n Starting sshd2 in port 22:
done.
```

Finally, we give ssh2 a try:

```
brian% ssh2 mayday
Host key not found from database.
Are you sure you want to continue connecting (yes/no)? yes
Host key saved to /opt/home/erl/koppel/.ssh2/hostkeys/key_22_mayday.pub
host key for mayday, accepted by koppel Wed Nov 22 2000 17:38:20
Passphrase for key "/opt/home/erl/koppel/.ssh2/id_dsa_1024_a" with comment "1024-bit
dsa, koppel@brian, Wed Nov 22 2000 16:50:21":
Last login: Wed Nov 22 2000 16:06:32 from brian.wustl.edu
Sun Microsystems Inc.   SunOS 5.7       Generic October 1998
No mail.
Sun Microsystems Inc.   SunOS 5.7       Generic October 1998
mayday%
```

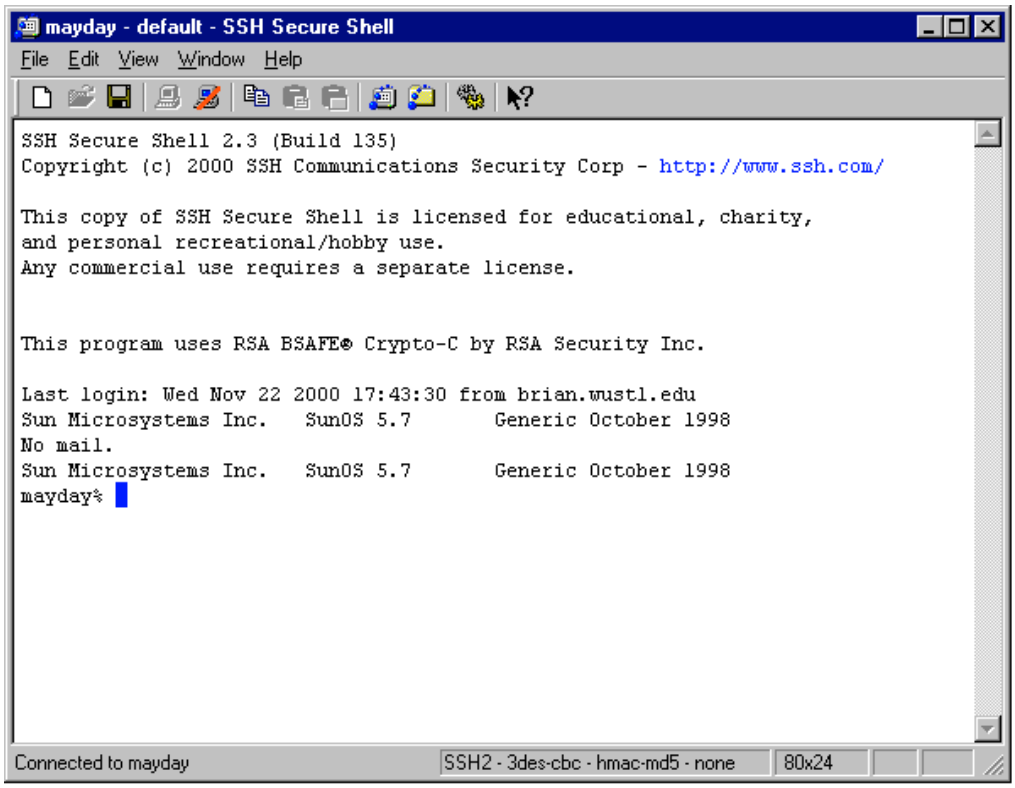
Since the host key is now saved, the next time we use ssh2, the login procedure is shorter and the Passphrase question appears first. For example, using ssh2 twice on mayday as the client:

```
mayday% ssh2 brian
Passphrase for key "/opt/home/erl/koppel/.ssh2/id_dsa_1024_a" with comment "1024-bit
dsa, koppel@mayday, Wed Nov 22 2000 17:12:23":
Last login: Wed Nov 22 2000 17:34:01 from mayday.wustl.edu
Sun Microsystems Inc.   SunOS 5.7       Generic October 1998
No mail.
Sun Microsystems Inc.   SunOS 5.7       Generic October 1998
brian%
```

NT4.0 Installation and Terminal Session

The distribution SSHWin-2.3.0.exe was installed under the Administrator account on an NT4.0 system. In addition, the scp2 program component was also installed (a Windows port of the UNIX secure copy utility). Installation was very quick and easy. After a suggested reboot, the ssh2 program was started, by clicking on the ssh2 icon. If this is the first time that ssh2 is being used for a particular host, the program pops up the following window:

After clicking “Yes”, and entering a Passphrase, a ssh2 session was established:



Also included with the package is a drag-and-drop file transfer program.

Conclusion

Information, including passwords, can be sniffed over public network segments. Installing ssh2 provides an easy method to encrypt a terminal session between a client and host, thus avoiding having to send plain-text passwords over the public Internet. In addition, by using ssh2, the ftp and telnet ports on UNIX systems can be disabled. A number of articles have shown that having these ports open leads to an insecure system.

References

1. CERT Incident Note IN-2000-09, "Systems Compromised Through a Vulnerability in the IRIX telnet daemon", Sept. 7, 2000, http://www.cert.org/incident_notes/IN-2000-09.html

2. "CERT/CC Overview Incident and Vulnerability Trends", Aug. 17, 2000, http://www.cert.org/present/cert-overview-trends/sld001.htm

3. "IRIX telnetd Environment Variable Format String Vulnerability", Nov 10, 2000, http://www.securityfocus.com/bid/1572

4. "Telnet port probe", http://www.networkice.com/advice/Intrusions/2003006/default.htm

5. Maximum Security: A Hacker's Guide to Protecting Your Internet Site and Network, Chapter 17, "UNIX, The Big Kahuna" and Chapter 29, "Telnet Based Attacks", http://security.tsu.ru/info/misc/maxsec/ch17/ch17.htm

6. “G-01: Telnetd Vulnerability”, Nov 3, 1995, http://ciac.llnl.gov/ciac/bulletins/g-01.shtml

7. SSH Communications Security, http://www.ssh.com/