

Exercises: SSH (Secure SHell): IP Services Workshop SANOG 16

July 18th, 2010

Exercises

Using SSH to Admin yur Box

1. [Generate your public/private Key Pair](#)
2. [Copy Your Public Key to the admin Account](#)
3. [Copy Your Public Key to the root Account](#)
4. [Update /etc/ssh/sshd_config](#)
5. [Consider the Power of scp -r](#)

Notes (CRITICAL)

1. The "#" and "\$" characters before commands represents your system prompt and is not part of the command itself. "#" indicates a command issued as root while "\$" indicates a command issued as a normal user.
2. *italics*: Items that are in *italics* are to be replaced with something of your choice. For instance, *username* means choose your own username, don't literally choose the word "username".

1.) Generate your Public/Private Key Pair [\[Top\]](#)

We will now generate a single RSA SSH protocol 2 key of 2048 bits. To do this, issue the following command. Do this as a normal user, not as root:

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

You will be prompted for a file location for the key as well as for a passphrase to encrypt the key file. Be sure to enter a passphrase. Private key files without passphrases are a security hole. Your passphrase can be pretty much anything you want.

2.) Copy Your Public Key to the sanog Account [\[Top\]](#)

First connect to your neighbor's machine as the userid *sanog* using ssh. We'll refer to your neighbor's machine as *wsXX*. You can use the IP address of their machine in place of a name to connect.

Here's what you do (as a normal user):

```
$ ssh sanog@wsXX
```

Now you'll be faced with a prompt similar to this:

```
The authenticity of host 'wsXX.ws3.conference.sanog.org (119.2.100.2xx)' can't be established.  
RSA2 key fingerprint is 60:f7:04:8b:f7:61:c4:41:6e:9a:6f:53:7d:95:cb:29.  
Are you sure you want to continue connecting (yes/no)?
```

You should say *yes* to this prompt, but you should understand what this means. Do you? If not, please ask your instructor.

Once you say *yes*, then you see another message like this:

```
Warning: Permanently added 'wsXX.ws3.conference.sanog.org' (RSA2) to the list of known hosts.  
[/etc/ssh/ssh_host_key.pub]  
sanog@wsXX.ws3.conference.sanog.org's password:
```

At this point enter in the password for the sanog account on your neighbor's machine.

Now you'll be logged in and see a prompt like this:

```
[sanog@wsXX ~]$
```

Now you should logout of your neighbor's machine, and then immediately log back in:

```
[sanog@wsXX ~]$ exit
$ ssh sanog@wsXX
```

Now you should simply be prompted for the sanog password on your neighbor's machine. You should not see the warning message again. Now, log out of your neighbor's machine again:

```
[sanog@wsXX ~]$ exit
```

Let's copy the public_key for your user account on your machine to the /home/sanog/.ssh directory on your neighbor's machine. As usual there are several ways to do this, but here's one set of steps that should work: _

```
$ cd /home/sanog/.ssh
$ scp id_rsa.pub sanog@wsXX:/tmp/.
$ ssh sanog@wsXX
[sanog@wsXX ~]$ cd .ssh [if ".ssh" is not there do "mkdir .ssh"]
[sanog@wsXX ~]$ cat /tmp/id_rsa.pub >> authorized_keys
[sanog@wsXX ~]$ rm /tmp/id_rsa.pub
[sanog@wsXX ~]$ exit
```

If you don't understand what this meant *please* ask an instructor to explain and give you a hand.

OK, so now your public key is located in the file /home/sanog/.ssh/authorized_keys in the sanog homedir on your neighbor's machine. So, now let's try connecting to sanog on your neighbor's machine:

```
$ ssh sanog@wsXX
```

You should now see something like:

```
$ ssh sanog@wsXX
Enter passphrase for RSA key 'sanog@wsXX':
```

And, at this point you type in the *passphrase* you used when creating your public/private key pair on your machine for your account - *not* the password for the sanog account on your neighbor's machine.

If you think about this that's pretty neat! Anywhere your public key resides you can log in using one passphrase, and it won't expire.

Now be sure that you log out of your neighbor's machine:

```
[sanog@wsXX ~]$ exit
```

3.) Copy Your Public Key to the root Account [[Top](#)]

You will now repeat exercise #2, with just a couple of differences. Note, you cannot log in directly to your neighbor's machine as root, so you must take advantage of the fact that you can get in as the userid *sanog* and then you can become root once you are logged in. This should work as long as your neighbor has not changed the root password as requested, and they created the sanog account correctly placing it in the wheel group.

So, here are the steps to do this:

```
$ cd /home/sanog/.ssh
$ scp id_rsa.pub sanog@wsXX:/tmp/.
$ ssh sanog@wsXX
[sanog@wsXX ~]$ su - [enter root password when requested]
# cd /root/.ssh [if ".ssh" is not there do mkdir /root/.ssh]
# cat /tmp/id_rsa.pub >> authorized_keys
# rm /tmp/id_rsa.pub
# exit
```

Now your public key is in the /root/.ssh/authorized_keys file on your neighbor's machine. You cannot log in yet to your neighbor's machine as root since the file /etc/ssh/sshd_config is configured to block all root access. Our next exercise will change this.

Be sure that everyone on your machine completes this exercises (#3).

4.) Update /etc/ssh/sshd_config [\[Top\]](#)

We have placed an sshd_config file on the noc server that you can copy to your machine to accomplish what we want to do. This configuration file only allows access to your machine via ssh if someone has their public key in the account they are trying to connect with. In addition, this file allows you to connect directly as root. This can actually be very useful, especially if you need to copy over a large number of files with root privileges. It's important that the passphrase you used on your private key is strong enough to resist brute force attacks.

For this exercise you must be root. Do the following:

```
# cd /etc/ssh
# cp sshd_config sshd_config.bak
# ftp noc
username: ftp
password: ftp
ftp> cd pub/FreeBSD/configs
ftp> lcd /etc/ssh
ftp> get sshd_config
ftp> exit
```

Now you can restart your ssh server and the new configuration will take effect, *but* you must coordinate this with your neighbors first. If they are still accessing your box to copy over keys, then wait to do this until they are done. If you don't, then they won't be able to log in and finish these exercises.

If your neighbor or neighbors are not ready, just go on to the final exercise and come back to this last step later.

To restart your ssh server (as root) do:

```
# /etc/rc.d/sshd restart
```

Once your neighbor has done this as well try logging in on their machine as root from your local account. For instance, if you are in a terminal window as root and you want to ssh to another machine as "sanog" you could do:

```
# su - sanog
[sanog@wsXX ~]$ ssh root@wsXX
```

You should be prompted for your passphrase, and you should be able to log in directly to your neighbor's machine as root! This is a very useful tool.

Be sure to exit your session on their machine:

```
# exit
```

And, have a look at the file /etc/ssh/sshd_config. Maybe compare it to /etc/ssh/sshd_config.bak to see some of the differences.

Be sure everyone on your machine completes this exercise.

5.) Consider the Power of scp [\[Top\]](#)

One of the most useful features of ssh are the scp and sftp tools that come with it. With scp (Secure CoPy) you can do some of the following:

- Securely copy files from your machine to another remote machine
- Securely copy file from a remote machine to your machine
- Securely copy files from one remote machine to another remote machine (less likely to work as ssh versions must match)
- Securely copy entire directory trees from one machine to another

We'll do one example of a directory structure copy from your neighbor's machine to your machine.

Let's copy all the files in your neighbors /usr/ports/palm directory structure to a directory in your /tmp directory.

```
$ mkdir /tmp/palm
$ scp -r sanog@wsXX:/usr/ports/palm/* /tmp/palm/.
```

That's it. Have a look at the /tmp/palm directory structure to convince yourself that things are there:

```
$ cd /tmp/palm
$ ls
$ ls -lah
$ ls -R
$ du -h
```

"ls -R" shows all directories recursively under the directory you are in. "du -h" tell you in "human readable format" how much space all the files in the directories under your current directory are using.

The "-r" option in scp can make system administration much easier.

If you want to remove /tmp/palm and all its subdirectories you can do this:

```
$ rm -rf /tmp/palm
```

Always be very careful with the "rm -rf" command as it can delete anything you have r/w access to recursively, with no warning, very quickly.

[\[Return to Top\]](#)

Hervey Allen, Phil Regnauld

Last modified: Sun Jul 18 16:13:58 BTT 2010