DRAFT

IT341 Introduction to System Administration

Project VI: Using ***ssh***, ***scp***, and ***sftp*** with Key-Based Authentication

This material here is mostly for **example** purposes ...

scp and sftp

When you install **ssh**, you also get **scp**, a secure copy for doing secure cp's from one machine to another (actually, it is a secure **rcp** – remote copy), and **sftp**, a secure version of ftp, that is a secure file transfer protocol. You can learn about both of these by looking at their man pages:

man scp

or

man sftp

Of course, there is also a man page for **ssh**.

scp is useful for quickly copying a file from one host to another. For example, say we are on **it20** and we wish to copy our (i.e., **it20**'s) **/etc/hosts** to **itvm2x-yz**. Rather than copy **hosts** directly to directory **/etc** on **itvm2x-yz**, it is safer to copy it to **itvm2x-yz**'s **/tmp** – a directory for holding files temporarily; then, once we log on to **itvm2x-yz**, we can move it into place (perhaps after saving **itvm2x-yz**'s original **/etc/hosts**). Anyway, we can use **scp** to do the copy:

```
abird@it20:~$ scp /etc/hosts itvm2x-yz:/tmp
abird@itvm2x-yz's password:
hosts 100% 628 0.6KB/s 00:00
abird@it20:~$
```

1. The first argument is the file we want to copy. Because it is on the host we are currently logged into we needn't specify the host.
2. The second argument tells **scp** where it should copy the file to:
   a. the host: **itvm2x-yz**:
   b. the target directory on that host: **/tmp**.
3. Notice **scp** needs **abird**'s password on **itvm2x-yz**. (Of course, because of NIS, **abird**'s password is the same on all hosts on the network – a good thing.)

We can also copy files from elsewhere to our own host. For example, to copy itvm2x-yz's **/etc/hosts** file to our (**it20**'s) **/tmp**, we *could* say:

```
abird@it20:~$ scp itvm2x-yz:/etc/hosts /tmp
abird@itvm2x-yz's password:
hosts 100% 624 0.6KB/s 00:00
abird@it20:~$
```

Again, we are asked for ***abird***’s password on itvm2x-yz.

We can recursively copy whole directories from one host to another. For example, to copy itvm2x-yz’s entire **`/etc`** to our (**it20**’s) **`/tmp`**, we would say

```
abird@it20:~$ scp -r itvm2x-yz:/etc /tmp
abird@itvm2x-yz's password:
defaultdomain 100% 6 0.0KB/s 00:00
adjtime 100% 48 0.1KB/s 00:00
global 100% 459 0.5KB/s 00:00
config 100% 1568 1.5KB/s 00:00
mtab 100% 629 0.6KB/s 00:00
scp: /etc/shadow: Permission denied
…
a whole lot of files
…
README 100% 371 0.4KB/s 00:00
K16dhcdbd 100% 1506 1.5KB/s 00:00
abird@it20:~$
```

Notice that **`scp`** will not copy **`/etc/shadow`** across; if it did allow it, anyone could take a look at a host’s **`/etc/shadow`**, whether they were ***sudo***ers or not.

If you want to have full access, you should work as user **root**. (Or, you should ask yourself if you really want to have such full access – you can really do damage to your system!)

Key-Based Authentication

One thing you may have noticed is that it would be a lot easier if we could push stuff (common files, etc.) from **it20** out to the client **itvm2x-yz**. *And we would like to do so without having to supply a password every time.*

So we will set up key-based authentication. Following the text, we will use a non-empty passphrase. Of course, this puts us back in the position of having to supply a pass phrase in place of a password. But we can then use **`ssh-agent`** for managing the pass phrase exchange whenever we are challenged. As you have read in the text, **`ssh-agent`** caches the pass phrase in memory while the current shell is active; when the shell dies, the pass phrase goes with it.

OK, so now our **`ssh`** client is **it20** and our **`ssh`** servers (from whom we want to push out files) are the itvm2x-yz hosts. In our example, we will set up key-based authentication with **itvm2x-yz**; we use it here only as an example.

Stuff that YOU do starts *HERE*

(***Each*** *of you should do these things!*)

On the virtual server:

1. You should first read the section on key-based authentication (pages 257 – 261) in *Beginning Ubuntu LTS Server Administration.*

2. Log in to your virtual server as yourself (for Al Bird and in the examples below, it's **abird**).

3. The first thing we have to do is generate a public/private key pair with **ssh-keygen**. We will use the passphrase **qazxsw** (which is easier to type than you might think).

4. Note: Although the instructions below use **dsa**, you may wish to use **rsa** instead because the former is being deprecated and may cause issues when using version 16.04. You may also wish to use a key size of **2048** instead of 1024.

Leave blank and press Enter

```
abird@it29vm-6:~$ ssh-keygen -t ~~dsa~~ -b ~~1024~~
Generating public/private dsa key pair.
Enter file in which to save the key (/home/abird/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):

Enter same passphrase again:
Your identification has been saved in /home/abird/.ssh/id_dsa.
Your public key has been saved in /home/abird/.ssh/id_dsa.pub.
The key fingerprint is:

2e:98:6f:72:9f:70:9c:37:11:c1:fc:ed:91:9b:b8:09 abird@it29vm-
6 The key's randomart image is:
+--[ DSA 1024]----+
| o.              |
| o.              |
| ..  .  .        |
| ..  +           |
| S  .  o +       |
| o o .E.. +      |
| o o = o. o      |
| ..o+ o .o       |
| +..o            |
+-----------------+
abird@it29vm-6:~$
```

4. Append the content of /**home/abird/.ssh/id_dsa.pub** to **/home/abird/.ssh/authorized_keys**, thus insuring that any file there already is not overridden; if the **authorized_keys** file doesn't already exist, it is created.

NOTE: If you used **rsa** instead of dsa, change the commands accordingly.

```
abird@itvm2x-yz:~$ cd .ssh
abird@itvm2x-yz:~$ cat id_dsa.pub >> authorized_keys
abird@itvm2x-yz:~$ ls -l
total 20
-rw-r--r-- 1 abird abird 598 2011-03-22 14:01
authorized_keys
-rw------- 1 abird abird 736 2011-03-22 13:56 id_dsa
-rw-r--r-- 1 abird abird 598 2011-03-22 13:56 id_dsa.pub
-rw-r--r-- 1 abird abird 7096 2011-03-21 10:12 known_hosts
abird@itvm2x-yz:~$
```

Recall, the >> denotes an append.

5. Now **ssh** to another machine to see if it works.

```
abird@itvm2x-yz:~$ ssh it20
Enter passphrase for key '/home/abird/.ssh/id_dsa':
Linux it20 2.6.32-29-generic-pae #58-Ubuntu SMP Fri Feb 11
19:15:25 UTC 2011 i686 GNU/Linux
Ubuntu 10.04 LTS

Welcome to Ubuntu!
* Documentation: https://help.ubuntu.com/

System information as of Tue Mar 22 14:04:04 EDT 2011

System load: 0.0 Memory usage: 13% Processes: 85

Usage of /: 7.3% of 18.82GB Swap usage: 0% Users logged
in: 0

Graph this data and manage this system at
https://landscape.canonical.com/

Last login: Tue Mar 22 14:01:02 2011 from itvm2x-
yz.it.cs.umb.edu
abird@it20:~$
```

Instead of asking for **abird**'s password, it asks for the pass phrase for the authentication key from **it20**. We haven't made too much progress. We would like to be able to get to

itvm2x-yz without having to supply the pass phrase. **ssh-agent** allows us to do this for a single shell process.

6. Make sure you understand **why** this works. Write about it in your notebook.

7. Again, let us log out and return to **itvm2x-yz**. Here we invoke **ssh-agent** with the name of the shell we want to use as its argument:

```
abird@it20:~$ exit logout
Connection to it20 closed.
abird@itvm2x-yz:~$ ssh-agent /bin/bash
abird@itvm2x-yz:~$
```

This will open a *subshell* that is *nested within* your current session

8. We now invoke **ssh-add**

```
abird@itvm2x-yz:~$ ssh-add
Enter passphrase for /home/abird/.ssh/id_dsa:
Identity added: /home/abird/.ssh/id_dsa (/home/abird/.ssh/id_dsa)
abird@itvm2x-yz:~$
```

ssh-add adds RSA or DSA identities to the authentication agent, **ssh-agent**. When run without arguments, it adds the files

~/.ssh/id_rsa, **~/.ssh/id_dsa** and **~/.ssh/identity**. Alternative file names can be given on the command line. If any file requires a passphrase, **ssh-add** asks for the pass phrase from the user.

9. Now, let's try to log into **it20** again.

```
abird@itvm2x-yz:~$ ssh it20
Linux it20 2.6.32-29-generic-pae #58-Ubuntu SMP Fri Feb 11
19:15:25 UTC 2011 i686 GNU/Linux
Ubuntu 10.04 LTS

Welcome to Ubuntu!
* Documentation: https://help.ubuntu.com/

System information as of Tue Mar 22 14:09:50 EDT 2011

System load: 0.0 Memory usage: 13% Processes: 84
Usage of /: 7.3% of 18.82GB Swap usage: 0% Users logged
in: 0
```

```
Graph this data and manage this system at
https://landscape.canonical.com/

Last login: Tue Mar 22 14:04:04 2011 from itvm2x-
yz.it.cs.umb.edu
abird@it20:~$
```

Yahoo!

The point of this is that, once your authentication key has been distributed to all hosts, you can use **`ssh-agent`** and **`ssh-add`** to set up a shell from which you can perform a task that is based on ssh (**`ssh`**, **`scp`**, **`sftp`**, **`rdist`**, etc) without being challenged for a password or pass phrase.

NOTE:

The agent is *only* usable within the login session where it is started. It ***will not*** carry over to concurrent or sequential sessions!