Practice: Using SSH Key-based Authentication

Guided exercise

In this lab, you will set up SSH key-based authentication.

Outcomes:

Students will set up SSH user key-based authentication to initiate SSH connections.

□ 1. Create an SSH key pair as **student** on desktopX using no passphrase.

```
[student@desktopX ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/student/.ssh/id_rsa): Enter
Created directory '/home/student/.ssh'.
Enter passphrase (empty for no passphrase): Enter
Enter same passphrase again: Enter
Your identification has been saved in /home/student/.ssh/id_rsa.
Your public key has been saved in /home/student/.ssh/id_rsa.pub.
...
```

□ 2. Send the SSH public key to the **student** account on serverX.

```
[student@desktopX ~]$ ssh-copy-id serverX
The authenticity of host 'serverX (172.25.X.11)' can't be established.
ECDSA key fingerprint is 33:fa:a1:3c:98:30:ff:f6:d4:99:00:4e:7f:84:3e:c3.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
 out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are
 prompted now it is to install the new keys
student@serverX's password: student

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'student@serverX'"
and check to make sure that only the key(s) you wanted were added.
```

□ 3. Run the **hostname** command by using **ssh** to display the hostname of the serverX.example.com machine without the need to enter a password.

```
[student@desktopX ~]$ ssh serverX 'hostname'
serverX.example.com
```

Customizing SSH Service Configuration

Objective

After completing this section, students should be able to customize sshd configuration to restrict direct logins as root or to disable password-based authentication.

The OpenSSH server configuration file

While OpenSSH server configuration usually does not require modification, additional security measures are available.

Various aspects of the OpenSSH server can be modified in the configuration file **/etc/ssh/sshd_config**.

Prohibit the root user from logging in using SSH

From a security standpoint, it is advisable to prohibit the root user from directly logging into the system with **ssh**.

- The username root exists on every Linux system by default, so a potential attacker only has to guess the password, instead of a valid username and password combination.
- The root user has unrestricted privileges.

The OpenSSH server has an internal configuration file setting to prohibit a system login as user root, which is commented out by default in the **/etc/ssh/sshd_config** file:

```
#PermitRootLogin yes
```

By enabling the previous option in the **/etc/ssh/sshd_config** configuration file as follows, the root user will be unable to log into the system using the **ssh** command after the sshd service has been restarted:

```
PermitRootLogin no
```

The sshd service has to be restarted to put the changes into effect:

```
[root@serverX ~]# systemctl restart sshd
```

Another option is to only allow key-based ssh login as root with:

```
PermitRootLogin without-password
```

Prohibit password authentication using SSH

Only allowing key-based logins to the remote command line has various advantages:

- SSH keys are longer than an average password, which adds security.
- Less effort to initiate remote shell access after the initial setup.

There is an option in the **/etc/ssh/sshd_config** configuration file which turns on password authentication by default:

```
PasswordAuthentication yes
```

To prevent password authentication, the **PasswordAuthentication** option has to be set to **no** and the sshd service needs to be restarted:

```
PasswordAuthentication no
```

Keep in mind that whenever you change the **/etc/ssh/sshd_config** file, the sshd service has to be restarted:

```
[root@serverX ~]# systemctl restart sshd
```

References

ssh(1), **sshd_config**(5) man pages

Practice: Restricting SSH Logins

Guided exercise

In this lab, you will enable additional security features in OpenSSH.

Outcomes:

Prohibit direct SSH login as root on serverX; prohibit users from using passwords to login through SSH to serverX; public key authentication should still be allowed for regular users.

Before you begin...

Reset the desktopX and serverX systems.

Run **lab ssh setup** on both desktopX and serverX. This will create a user account called **visitor** with a password of **password**.

```
[student@desktopX ~]$ lab ssh setup
```

```
[student@serverX ~]$ lab ssh setup
```

□ 1. Generate SSH keys on desktopX, copy the public key to the **student** account on serverX, and verify that the keys are working.

□ 1.1. Generate the SSH keys on desktopX.

```
[student@desktopX ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/student/.ssh/id_rsa):  Enter
Created directory '/home/student/.ssh'.
Enter passphrase (empty for no passphrase):   Enter
Enter same passphrase again:  Enter
Your identification has been saved in /home/student/.ssh/id_rsa.
Your public key has been saved in /home/student/.ssh/id_rsa.pub.
...
```

□ 1.2. Copy the SSH public key to the **student** account on serverX.

```
[student@desktopX ~]$ ssh-copy-id serverX
The authenticity of host 'serverX (172.25.X.11)' can't be established.
ECDSA key fingerprint is 33:fa:a1:3c:98:30:ff:f6:d4:99:00:4e:7f:84:3e:c3.
Are you sure you want to continue connecting (yes/no)? yes

/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to
 filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are
 prompted now it is to install the new keys
student@serverX's password: student

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'student@serverX'"
and check to make sure that only the key(s) you wanted were added.
```

□ 1.3. Verify that key-based SSH authentication is working for user student on serverX.

```
[student@desktopX ~]$ ssh student@serverX
[student@serverX ~]$
```

□ 2. Log into the serverX machine and obtain superuser privileges.

```
[student@desktopX ~]$ ssh student@serverX
[student@serverX ~]$ su -
Password: redhat
[root@serverX ~]#
```

□ 3. Configure SSH on serverX to prevent root logins.

□ 3.1. As user root, edit **/etc/ssh/sshd_config** on serverX so that "PermitRootLogin" is uncommented and set to "no."

```
PermitRootLogin no
```

□ 3.2. Restart the SSH service on the serverX machine.

```
[root@serverX ~]# systemctl restart sshd
```

□ 3.3. Confirm that **root** cannot log in with SSH, but **student** is permitted to log in.

```
[student@desktopX ~]$ ssh root@serverX
Password: redhat
Permission denied, please try again.
Password: redhat
Permission denied, please try again.
Password: redhat
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password)

[student@desktopX ~]$ ssh student@serverX
[student@serverX ~]$
```

□ 4. Configure SSH on serverX to prevent password authentication.

□ 4.1. Edit the configuration file **/etc/ssh/sshd_config** as user root so that the "PasswordAuthentication" entry is set to "no":

```
PasswordAuthentication no
```

□ 4.2. Restart the SSH service.

```
[root@serverX ~]# systemctl restart sshd
```

□ 4.3. Confirm that **visitor** cannot log in using a password, but **student** is permitted to log in using the SSH keys created earlier.

```
[student@desktopX ~]$ ssh visitor@serverX
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
```

```
[student@desktopX ~]$ ssh student@serverX
[student@serverX ~]$
```