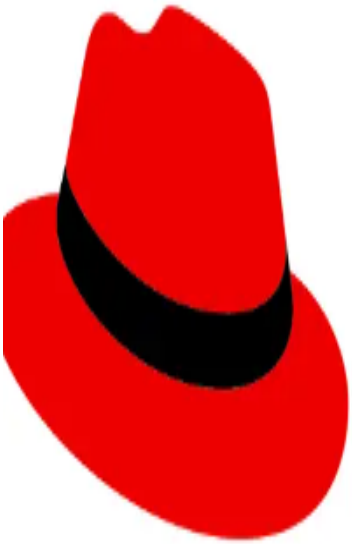

# Red Hat Enterprise Linux 9

# RHCSA-SA1 LAB-Book

# Chapter 10 - Configuring and Securing SSH

## What is the OpenSSH Secure Shell (SSH)?

- ⇨ The OpenSSH Secure Shell **.ssh** is used to securely run a shell on a remote system.
- ⇨ If you have a user account on a remote Linux system providing SSH services **.ssh** is the command normally used to remotely log into that system.
- ⇨ The **.ssh** command can also be used to run an individual command on a remote system.

## Why use OpenSSH?

- ⇨ SSH (Secure Shell) is a tool for secure system administration, file transfers, and other communication across the Internet or other untrusted network.
- ⇨ It encrypts identities, passwords, and transmitted data so that they cannot be eavesdropped and stolen. OpenSSH is an open-source implementation of the SSH protocol.

## Where OpenSSH Secure Shell is used?

- ⇨ SSH is typically used to log into a remote machine and execute commands, but it also supports tunnelling, forwarding TCP ports and X11 connections; it can transfer files using the associated SSH file transfer (SFTP) or secure copy (SCP) protocols.
- ⇨ SSH uses the client-server model.

## Secure Shell Examples:

**Eg1:** Create a remote interactive shell as the current user, then return to your previous shell when done with the exit command?

- ⇨ ssh 172.25.0.11

**Eg2:** Connect to a remote shell as a different user (remote user) on a selected host (remotehost)?

⇨ ssh student@172.25.0.11

remote user  remotehost (IP)

⇨ password: student

**Eg3:** Command: w: Displays a list of users currently logged into the computer. This is especially useful to show which users are logged in using ssh from which remote locations, and what they are doing?

⇨ w -f

## SSH Host Keys:

- ✓ When a **ssh** client connects to an SSH Server, before the client logs in. the server sends it a copy of its public key.
- ✓ This is used to set up the secure encryption for the communication channel and to authenticate the server to the client.
- ✓ The first time a user uses ssh to connect to a particular server, the ssh command stores the server's public key in the user's ~/.ssh/known_hosts file.
- ✓ Every time the user connects after that, the client makes sure it gets the same public key from the server by comparing the server's entry in the ~/.ssh/known_hosts file to the public key the server sent.
- ✓ If the keys do not match, the client assumes that the network traffic is being hijacked or that the server has been compromised, and breaks the connection.
- ✓ This means that if server's public key is changed (because the key was lost due to hard drive failure, or replaced for some legitimate reason), users will need to update their ~/.ssh/known_hosts files and remove the old entry in order to

log in.

**Practical 1:** Host ID's are stored in "~/.ssh/known_hosts on your local client system?

⇨ cat ~/.ssh/known_hosts

**Practical 2:** Host keys are stored in /etc/ssh/ssh_host_key* on the SSH server?

⇨

```
[student@gandhar ~]$ ls /etc/ssh/*key*
/etc/ssh/ssh_host_ecdsa_key      /etc/ssh/ssh_host_ed25519_key      /etc/ssh/ssh_host_rsa_key
/etc/ssh/ssh_host_ecdsa_key.pub  /etc/ssh/ssh_host_ed25519_key.pub  /etc/ssh/ssh_host_rsa_key.pub
```

⇨

## Practical based on Accessing the remote command line:

**Step 1:** Log in as student on your desktop machine.

**Step 2:** ssh student@serverX

**Step 3:** Run the w command. The output of the w clearly indicates we have logged in as user student from desktop.

**w -f**

**Step 4:** Execute the exit command terminate the source shell connection.

**exit**

**Step 5:** This time, ssh to your serverX machine as user root.

**ssh root@serverX**

**Password: redhat**

**Step 6:** Run the w command again. This time, the output of the w shows the active connection to the root user account from desktop

**w -f**

**Step 7:** Run the exit to terminate the secure shell connection.

**exit**

**Step 8:** In this, there is only one host entry in the know_hosts. So it can be removed completely. Remove the known_hosts file for the user student.

**rm ~/.ssh/known_hosts**

**Step 9:** ssh to serverX as root again. Accept the key, log in, and then exit the session.

**ssh root@serverX**

**password: Redhat**

**exit**

**Step 10**: Use ssh non-interactively to run the hostname command on serverX as root.

**ssh root@serverX 172.25.0.11.**

Hostname

## Configuring SSH-KEY-BASED Authentication:

- Users can authenticate ssh logins without a password by using public key authentications .ssh allows users to authenticate using a private-public key scheme.
- **This means that two keys are generated, a private key and a public key.**

  **Private key:** It is used as the authentication credential, and like a password, must to kept secret and secure.

  **Public key:**
  - ✓ Is copied to systems the user wants to log into, and is used to verify the private key.
  - ✓ The public key does not need to be secret.
  - ✓ An SSH server that has the public key can issue a challenge that can only be answered by a system holiday your private key.
  - ✓ Key generation is done using the ssh-keygen command.
  - ✓ This generates the private key ~/.ssh/id_rsa and the public key ~/.ssh/id_rsa.pub.
  - ✓ Once the SSH keys have been generated, they are stored by default in the .ssh/ directory of your home directory.

- Permissions should be 600 on the private key and 644 on the public key.
- Before key-based authentication can be used, the public key needs to copied to the destination system. This can be done with ssh-copy-id.

  **Command:** ssh-copy-id root@desktopY

- When the key is copied to another system using **ssh-copy-id** it copies the **~/.ssh/id_rsa.pub** file by default.

- **Practical based on ssh-keygen (Passphrase):**

  **Step 1:** ssh-keygen

  **Step 2:**

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/student/.ssh/id_rsa): Enter
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in Enter.
Your public key has been saved in Enter.pub.
The key fingerprint is:
SHA256:5qL8omanoDVcRbh00eYFYQ5RV/vqmGBuiBGj/Eb66bE student@gandhar
The key's randomart image is:
+---[RSA 2048]----+
|      .===o...    |
|     o..+o..  .   |
|    . o.o..  .    |
|     +.  .    .   |
| . ..o  S     .   |
| .o.o  o      .   |
|. ++.o..+    .    |
|.o+o*=.+.. +      |
|.ooBEo... o .     |
+----[SHA256]-----+
```

**Note:**

To check the sshd status command is systemctl status sshd

To reset the desktopX and serverX systems command is las ssh setup

- **Practical based on ssh-keygen (With Password):**

  **Step 1:** ssh root@172.25.0.10

  **Step 2:** cd .ssh

  **Step 3:** cat ~/.ssh/known_hosts

  **Step 4:** logout

## Restricting SSH Logins:

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 desktop.

⇨ Ssh-keygen

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

⇨ ssh-copy-id serverX

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

⇨ ssh student@serverX

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

⇨ ssh student@serverX

⇨ su –

⇨ password: Redhat

3. Configure SSH service on the serverX machine.

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.

⇨ systemctl reload sshd

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

⇨ ssh root@serverX
⇨ password: Redhat
⇨ password: Redhat
⇨ password: Redhat
⇨ ssh student@serverX

4. Configure SSH on serverX to present password authentication.
   4.1. Edit the configuration file /etc/ssh/sshd_config an user root so that the “PasswordAuthentication” entry is set to “no”.

⇨ PasswordAuthentication no

4.2 Restart the SSH service.

⇨ systemctl reload 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.

⇨ ssh visitor@serverX

**END**