

RHSA (Admin 1) Red Hat Enterprise Linux 8 Essentials

Prepared by Eng. Taghreed Elsayed

Lecture 7

جامعة الجميع الذكية

تسعى لتكون الكيان الأول الذي يتبنى مبدأ المجتمع التعليمي
التعاوني ويقدم العلم مقابل العمل لتنمية وبناء المجتمعات.

CONFIGURING AND SECURING OPENSSSH



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

WHAT IS THE OPENSSSH SECURE SHELL (SSH)?

- ❑ The term **OpenSSH** refers to the software implementation of the Secure Shell software used in the system.
- ❑ 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.

➤ SECURE SHELL EXAMPLES

- ❑ Here are some examples of **ssh command syntax for remote login** and remote execution:
 - Create a remote interactive shell as the current user, then return to your previous shell when done with the exit command.



```
[student@host ~]$ssh remotehost  
student@remotehost's password:  
[student@remotehost ~]$ exit  
Connection to remotehost closed.  
[student@host ~]$
```

- Connect to a remote shell as a different user (remoteuser) on a selected host (remotehost):

```
[student@host ~]$ssh remoteuser@remotehost  
remoteuser@remotehost's password:  
[remoteuser@remotehost ~]$
```



- Execute a single command (hostname) on a remote host (remotehost) and as a remote user (remoteuser) in a way that returns the output to the local display:

```
[student@host ~]$ssh remoteuser@remotehost hostname
remoteuser@remotehost's password:
remotehost.example.com
[student@host ~]$
```

- The w command 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.

```
[student@host ~]$w -f
```

USER	TTY	FROM	LOGIN@	IDLE	JCPU	PCPU	WHAT
student	tty1	:0	Wed08	2days	1:52m	0.07s	pam: gdm-passwo
root	tty6	-	12:33	4:14m	16.27s	15.74s	-bash
student	pts/0	:0.0	Wed08	5:11	1.63s	1.60s	/usr/bin/gnome-
student	pts/1	:0.0	Wed08	43:44	14.48s	13.81s	vim hello.c
student	pts/3	:0.0	Wed14	0.00s	0.06s	0.06s	w
visitor	pts/6	server2.example.	09:22	3:14	0.02s	0.02s	-bash

- ❖ **In this example**, user student logged in on virtual console 1 (tty1) through the graphical login (:0) at about 08:00 on Wednesday.
- ❖ User student currently has three pseudo-terminals open (pts/0, pts/1, and pts/3) started by the graphical environment; these are almost certainly terminal windows.
- ❖ In one window, student is editing hello.c.



- ❖ User root is logged in on virtual console 6, starting at 12:33 today. User visitor logged in on pseudo-terminal 6 at 09:22 today from the host server2.
- ❖ example.com (note that the name has been truncated), probably using ssh, and has been sitting idle at a shell prompt for three minutes and 14 seconds.

SSH HOST KEYS

- ☐ **SSH secures** communication through public-key encryption.
- ☐ When an 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 a 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.



- Host IDs are stored in `~/.ssh/known_hosts` on your local client system:

```
$ cat ~/.ssh/known_hosts
remotehost,192.168.0.101 ssh-rsa AAAAB3Nzac...
```

- Host keys are stored in `/etc/ssh/ssh_host_key*` on the SSH server.

```
$ ls /etc/ssh/*key*
ssh_host_dsa_key      ssh_host_key          ssh_host_rsa_key
ssh_host_dsa_key.pub  ssh_host_key.pub      ssh_host_rsa_key.pub
```



SSH KEY-BASED AUTHENTICATION

- ☐ Users can authenticate ssh logins without a password by using **public key** authentication. **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.
- ☐ The private key file is used as the authentication credential, and like a password, must be kept secret and secure.
- ☐ The **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 holding your private key.
- ☐ As a result, you can authenticate using the presence of your key.

- ❑ This allows you to access systems in a way that doesn't require typing a password every time, but is still secure. 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`.
- ❑ Before key-based authentication can be used, the public key needs to be copied to the destination system. This can be done with `ssh-copy-id`.

```
[student@desktopX ~]$  
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.

USING SSH KEY-BASED AUTHENTICATION

❑ 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

❖ 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**
 - 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 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
```

- 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  
  
sshd
```

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

```
PermitRootLogin without-password
```

CONFIGURING AND SECURING OPENSSSH SERVICE

- ❖ you will add security measures to the ssh service.
- ❖ Run lab ssh setup as the student user 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
```

Unless specified, all steps are to be performed as user visitor.

- ❑ 1. Generate SSH keys on desktopX for user visitor and copy the public key to the visitor account on serverX.

1.1. Generate a SSH public key on desktopX as user visitor.

```
[visitor@desktopX ~]$  
ssh-keygen
```

1.2. Install the SSH public key generated previously on desktopX to the visitor account on serverX.

```
[visitor@desktopX ~]$ ssh-copy-id serverX
The authenticity of host 'serverX (172.25.X.11)' can't be established.
ECDSA key fingerprint is xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx.
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
visitor@serverX's password: password

Number of key(s) added: 1

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



2. Disable ssh login for the root user and password-based SSH authentication on serverX.

2.1. Log into the serverX virtual machine as user root.

```
PermitRootLogin no  
PasswordAuthentication no
```

2.2. Customize the ssh service on serverX by disabling SSH connections for the user root and only allow key-based login.

Set the necessary config file parameters in /etc/ssh/sshd_config:

2. Disable ssh login for the root user and password-based SSH authentication on serverX.

2.1. Log into the serverX virtual machine as user root.

```
PermitRootLogin no  
PasswordAuthentication no
```

2.2. Customize the ssh service on serverX by disabling SSH connections for the user root and only allow key-based login.

Set the necessary config file parameters in /etc/ssh/sshd_config:



2. Disable ssh login for the root user and password-based SSH authentication on serverX.

2.1. Log into the serverX virtual machine as user root.

```
[visitor@desktopX ~]$  
ssh root@serverX
```

2.2. Customize the ssh service on serverX by disabling SSH connections for the user root and only allow key-based login.

Set the necessary config file parameters in /etc/ssh/sshd_config:

```
PermitRootLogin no  
PasswordAuthentication no
```

2.3. Restart the sshd service on serverX.

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

3. Verify that user root is not allowed to login to serverX by using ssh, while user visitor is with the private key.

3.1. On a different terminal window on desktopX, validate that user root cannot connect to serverX with the ssh command. It should fail because we disabled root logins with the ssh service.

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



3.2. Try logging in as user student to serverX from desktopX by using ssh. It should fail because we did not add the public key from that user to the student account on the serverX machine.

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

3.3. Verify the ssh service is still accepting key-based authentication by successfully connecting to serverX as user visitor with the ssh command.

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





جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

