How to Install / Configure OpenSSH on Linux

Introduction:

Secure Shell (SSH) is a cryptographic protocol that allows a client machine to interact with a remote server in a secure environment.

Over **Secure Shell (SSH**) communication a high-level of encryption protects the exchange of information and allows file transfer or issue commands between remote machines securely.

During this LAB work we used CentOS 7 as an Operating system

Prerequisites:

- CentOS 7 system to act as an SSH server
- A user with necessary permissions
- Access to a command line (Ctrl-Alt-T)
- **yum** utility (included by default)

Installing and Enabling OpenSSH:

SSH software packages are included on CentOS by default. However, if these packages are not present on your system, you can easily install them by completing Step 1, outline as below.

Step 1: Install OpenSSH Server Software Package

Enter the following command from your terminal to start the installation process:

```
sudo yum -y install openssh-server openssh-clients
```

This command installs both the OpenSSH client applications, as well as the OpenSSH server daemon, **sshd**.

```
[phoenixnap@localhost ~]$ sudo yum -y install openssh-server openssh-clients
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.wwfx.net
 * extras: mirror.wwfx.net
 * updates: mirror.slu.cz
Package openssh-server-7.4p1-16.el7.x86_64 already installed and latest version
Package openssh-clients-7.4p1-16.el7.x86_64 already installed and latest version
Nothing to do
```

In this example, the system informs us that the latest version is already present.

Step 2: Starting SSH Service

To start the **SSH daemon** on the OpenSSH server:

```
sudo systemctl start sshd
```

When active, **sshd** continuously listens for client connections from any of the client tools. When a connection request occurs, **sshd** sets up the correct connection.

Step 3: Check sshd status

Check the status of the SSH daemon:

```
sudo systemctl status sshd
```

As we have previously started the service, the output confirms that it is active.

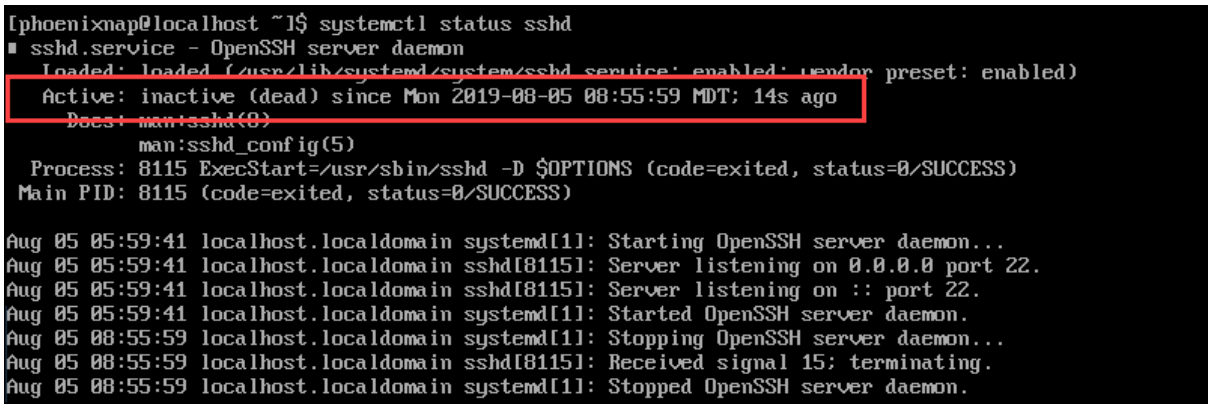
```
[phoenixnap@localhost ~]$ systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2019-08-05 02:36:58 MDT; 29min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1094 (sshd)
   CGroup: /system.slice/sshd.service
           └─1094 /usr/sbin/sshd -D

Aug 05 02:36:58 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
Aug 05 02:36:58 localhost.localdomain sshd[1094]: Server listening on 0.0.0.0 port 22.
Aug 05 02:36:58 localhost.localdomain sshd[1094]: Server listening on :: port 22.
Aug 05 02:36:58 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
Aug 05 02:37:51 localhost.localdomain sshd[1386]: Accepted password for phoenixnap from ::1 por...h2
Aug 05 02:45:54 localhost.localdomain sshd[1466]: Accepted password for phoenixnap from ::1 por...h2
Hint: Some lines were ellipsized, use -l to show in full.
```

To stop the SSH daemon enter:

```
systemctl stop sshd
```

We can check if the service has stopped by verifying the status. The output shows that the service is inactive and the time and date when the status last changed.



Step 4: Enable OpenSSH Service

Enable SSH to start automatically after each system reboot by using the **systemctl** command:

```
sudo systemctl enable sshd
```

To disable SSH after reboot enter:

```
sudo systemctl disable sshd
```

OpenSSH Server Configuration:

Properly configuring the **sshd** configuration file hardens server security. The most common settings to enhance security are changing the port number, disabling root logins, and limiting access to only certain users.

To edit these setting access the /**etc/ssh/sshd_config** file:

```
sudo vim /etc/ssh/sshd_config
```

Once you access the file by using a text editor (in this example we used **vim**), you can disable root logins and edit the default port number:

- To disable root login:

  PermitRootLogin no

- Change the SSH port to run on a non-standard port. For example:

  Port 2002

- Allowing Specific User. For example:

  AllowUsers Arif islam

```
Port 2002
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
SyslogFacility AUTHPRIV
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Remember to uncomment the lines that you edit by removing the hashtag.

Save and close the file. Restart **sshd**: error

By default, SELinux only allows port **22** for SSH. So, what we need to do is enablling newly created port through **SELinux**. To do that, **run** the commands below:

```
semanage port -a -t ssh_port_t -p tcp 2002
```

If you run the commands above and get an error that **semanage command not found**, run the commands below to install it.

```
yum -y install policycoreutis-python
```

Now we can run the ***semange*** commend again to allow the new port through SELinux.

After that, run the commands below to allow the new port through the firewall.

```
firewall-cmd --permanent --zone=public --add-port=2002/tcp
```

Reload the firewall configurations

```
sudo firewall-cmd –reload
```

Restart SSH by running the command as below

```
service sshd restart / systemctl restart sshd.service
```

Now verify that SSH is now running on the new port by running the commands below

```
ss -tnlp | grep ssh
```

Exit and try signing in using the new port number.

```
ssh root@192.168.0.115 -p 2002
```