Name: Erebete, Jan Kenneth F.	**Date Performed: 9/12/25**
Course/Section: CPE212 - CPE3124	**Date Submitted: 9/12/25**
Instructor: Engr. Robin Valenzuela	**Semester and SY: 2025 - 2026**

Activity 3: Install SSH server on CentOS or RHEL 8

1. **Objectives:**

1.1 Install Community Enterprise OS or Red Hat Linux OS

1.2 Configure remote SSH connection from remote computer to CentOS/RHEL-8

2. **Discussion:**

CentOS vs. Debian: Overview

CentOS and Debian are Linux distributions that spawn from opposite ends of the candle.

CentOS is a free downstream rebuild of the commercial Red Hat Enterprise Linux distribution where, in contrast, Debian is the free upstream distribution that is the base for other distributions, including the Ubuntu Linux distribution.

As with many Linux distributions, CentOS and Debian are generally more alike than different; it isn't until we dig a little deeper that we find where they branch.

CentOS vs. Debian: Architecture

The available supported architectures can be the determining factor as to whether a distro is a viable option or not. Debian and CentOS are both very popular for x86_64/AMD64, but what other archs are supported by each?

Both Debian and CentOS support AArch64/ARM64, armhf/armhfp , i386 , ppc64el/ppc64le. (Note: armhf/armhfp and i386 are supported in CentOS 7 only.)

CentOS 7 additionally supports POWER9 while Debian and CentOS 8 do not. CentOS 7 focuses on the x86_64/AMD64 architecture with the other archs released through the AltArch SIG (Alternate Architecture Special Interest Group) with CentOS 8 supporting x86_64/AMD64, AArch64 and ppc64le equally.

Debian supports MIPSel, MIPS64el and s390x while CentOS does not. Much like CentOS 8, Debian does not favor one arch over another —all supported architectures are supported equally.

CentOS vs. Debian: Package Management

Most Linux distributions have some form of package manager nowadays, with some more complex and feature-rich than others.

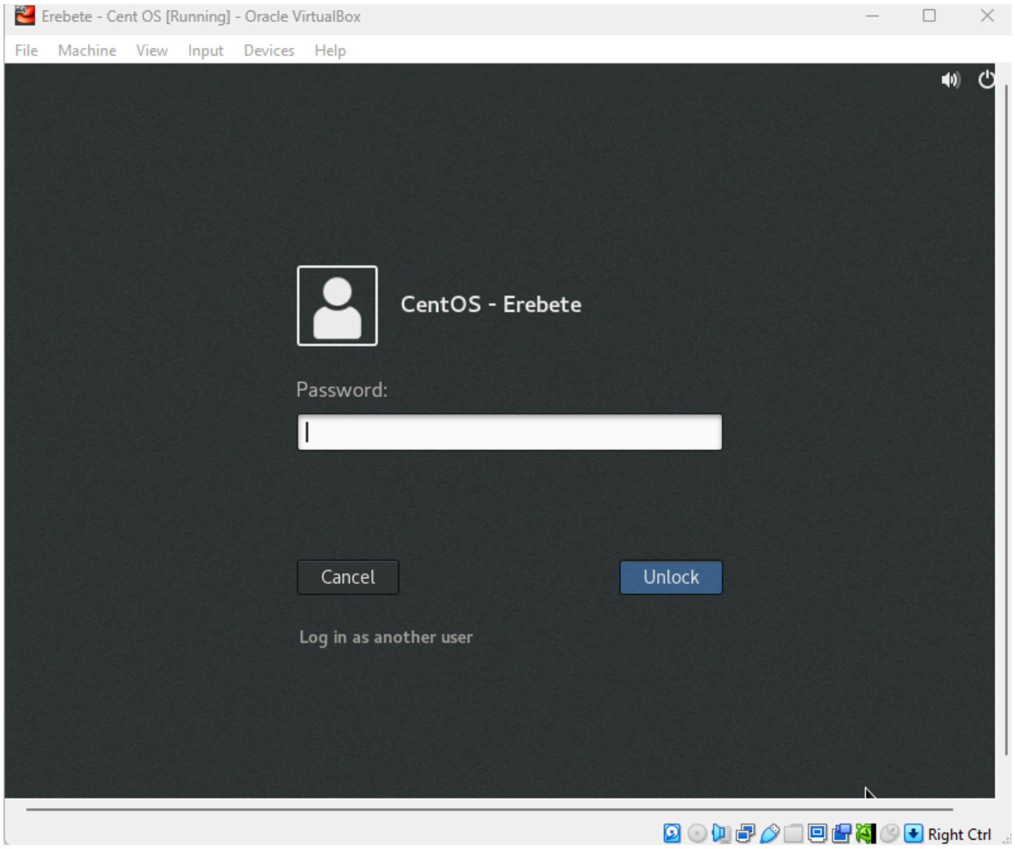
CentOS uses the RPM package format and YUM/DNF as the package manager.

Debian uses the DEB package format and dpkg/APT as the package manager.

Both offer full-feature package management with network-based repository support, dependency checking and resolution, etc.. If you're familiar with one but not the other, you may have a little trouble switching over, but they're not overwhelmingly different. They both have similar features, just available through a different interface.

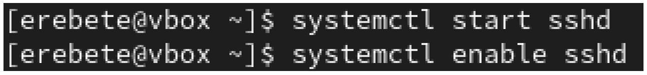
Task 1: Download the CentOS or RHEL-8 image (Create screenshots of the following)

1. Download the image of the CentOS here: http://mirror.rise.ph/centos/7.9.2009/isos/x86_64/
2. Create a VM machine with 2 Gb RAM and 20 Gb HD.
3. Install the downloaded image.
4. Show evidence that the OS was installed already.



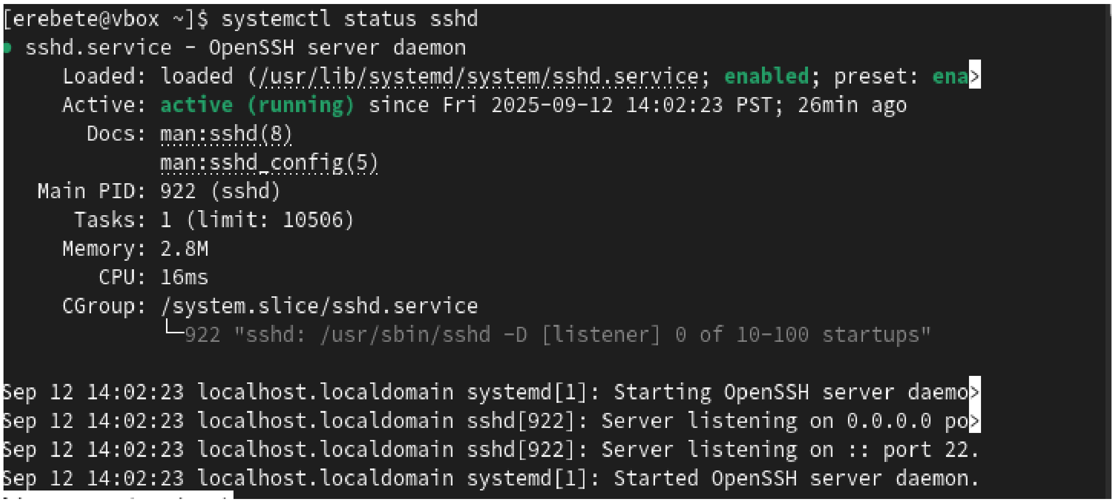
Task 2: Install the SSH server package *openssh*

1. Install the ssh server package *openssh* by using the *dnf* command:
   $ dnf install openssh-server
2. Start the *sshd* daemon and set to start after reboot:
   $ systemctl start sshd
   $ systemctl enable sshd



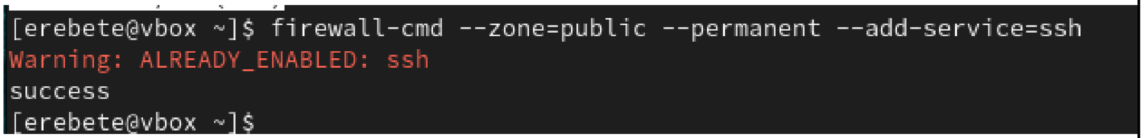
```
[erebete@vbox ~]$ systemctl start sshd
[erebete@vbox ~]$ systemctl enable sshd
```

3. Confirm that the sshd daemon is up and running:
   $ systemctl status sshd



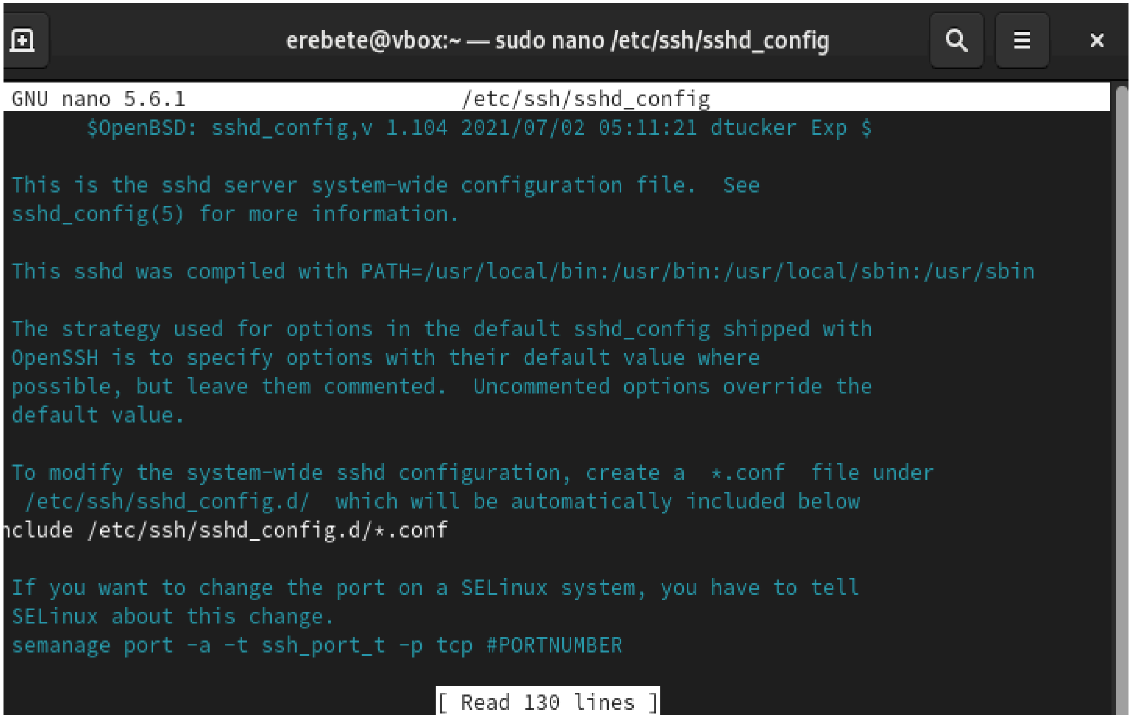
```
[erebete@vbox ~]$ systemctl status sshd
● sshd.service - OpenSSH server daemon
     Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; preset: ena>
     Active: active (running) since Fri 2025-09-12 14:02:23 PST; 26min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 922 (sshd)
      Tasks: 1 (limit: 10506)
     Memory: 2.8M
        CPU: 16ms
     CGroup: /system.slice/sshd.service
             └─922 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Sep 12 14:02:23 localhost.localdomain systemd[1]: Starting OpenSSH server daemo>
Sep 12 14:02:23 localhost.localdomain sshd[922]: Server listening on 0.0.0.0 po>
Sep 12 14:02:23 localhost.localdomain sshd[922]: Server listening on :: port 22.
Sep 12 14:02:23 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
```

4. Open the SSH port 22 to allow incoming traffic:
   $ firewall-cmd --zone=public --permanent --add-service=ssh
   $ firewall-cmd --reload

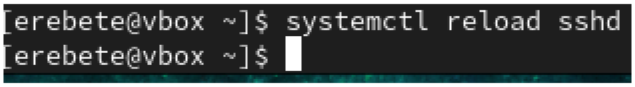


```
[erebete@vbox ~]$ firewall-cmd --zone=public --permanent --add-service=ssh
Warning: ALREADY_ENABLED: ssh
success
[erebete@vbox ~]$
```

5. Locate the ssh server man config file /*etc/ssh/sshd_config* and perform custom configuration. Every time you make any change to the /*etc/ssh/sshd-config* configuration file reload the *sshd* service to apply changes:
   $ systemctl reload sshd



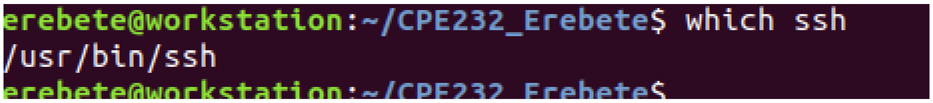
```
GNU nano 5.6.1                    /etc/ssh/sshd_config
      $OpenBSD: sshd_config,v 1.104 2021/07/02 05:11:21 dtucker Exp $

This is the sshd server system-wide configuration file.  See
sshd_config(5) for more information.

This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin

The strategy used for options in the default sshd_config shipped with
OpenSSH is to specify options with their default value where
possible, but leave them commented.  Uncommented options override the
default value.

To modify the system-wide sshd configuration, create a  *.conf  file under
 /etc/ssh/sshd_config.d/  which will be automatically included below
nclude /etc/ssh/sshd_config.d/*.conf

If you want to change the port on a SELinux system, you have to tell
SELinux about this change.
semanage port -a -t ssh_port_t -p tcp #PORTNUMBER

                                   [ Read 130 lines ]
```



```
[erebete@vbox ~]$ systemctl reload sshd
[erebete@vbox ~]$ 
```

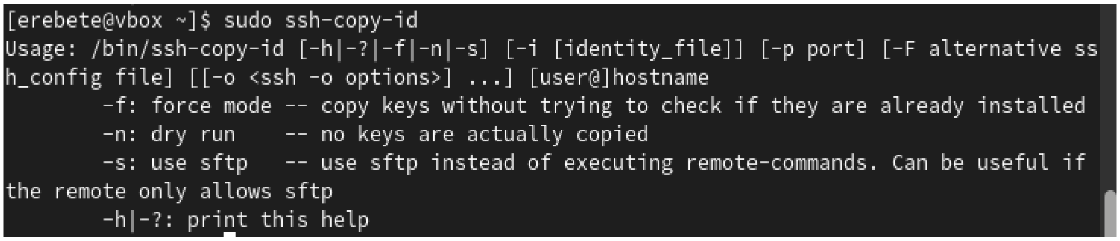
Task 3: Copy the Public Key to CentOS

1. Make sure that *ssh* is installed on the local machine.



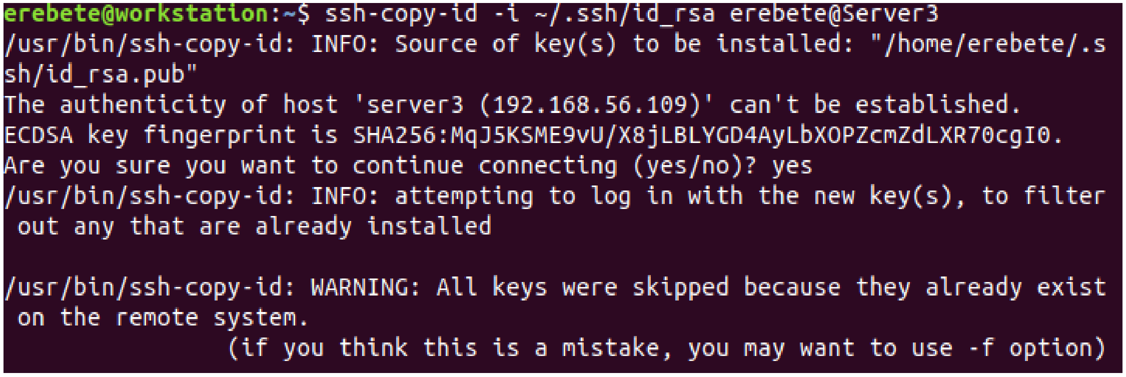
```
erebete@workstation:~/CPE232_Erebete$ which ssh
/usr/bin/ssh
erebete@workstation:~/CPE232_Erebete$
```

2. Using the command *ssh-copy-id*, connect your local machine to CentOS.

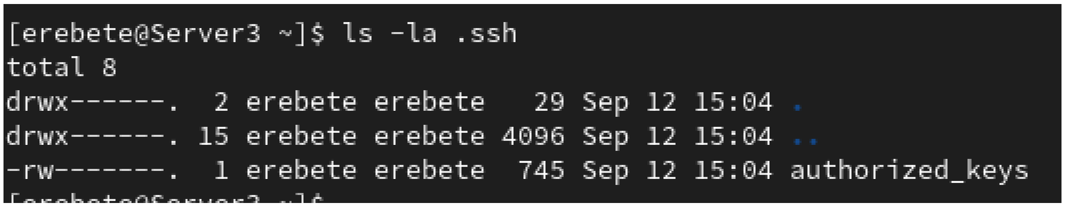


```
[erebete@vbox ~]$ sudo ssh-copy-id
Usage: /bin/ssh-copy-id [-h|-?|-f|-n|-s] [-i [identity_file]] [-p port] [-F alternative ss
h_config file] [[-o <ssh -o options>] ...] [user@]hostname
        -f: force mode -- copy keys without trying to check if they are already installed
        -n: dry run    -- no keys are actually copied
        -s: use sftp   -- use sftp instead of executing remote-commands. Can be useful if
the remote only allows sftp
        -h|-?: print this help
```

3. On CentOS, verify that you have the *authorized_keys*.



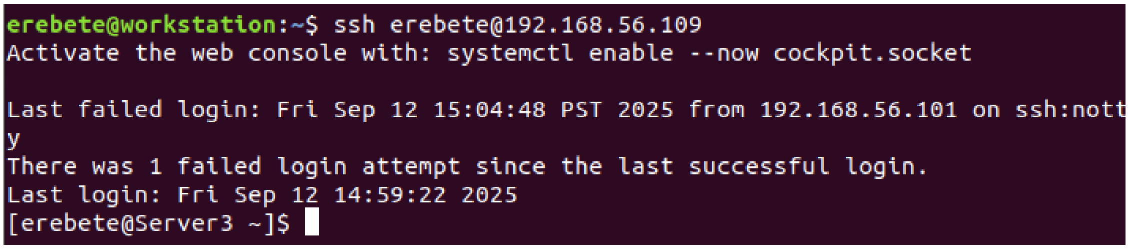
```
erebete@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa erebete@Server3
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/erebete/.s
sh/id_rsa.pub"
The authenticity of host 'server3 (192.168.56.109)' can't be established.
ECDSA key fingerprint is SHA256:MqJ5KSME9vU/X8jLBLYGD4AyLbXOPZcmZdLXR70cgI0.
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: WARNING: All keys were skipped because they already exist
 on the remote system.
                (if you think this is a mistake, you may want to use -f option)
```



```
[erebete@Server3 ~]$ ls -la .ssh
total 8
drwx------.  2 erebete erebete   29 Sep 12 15:04 .
drwx------. 15 erebete erebete 4096 Sep 12 15:04 ..
-rw-------.  1 erebete erebete  745 Sep 12 15:04 authorized_keys
[erebete@Server3 ~]$
```

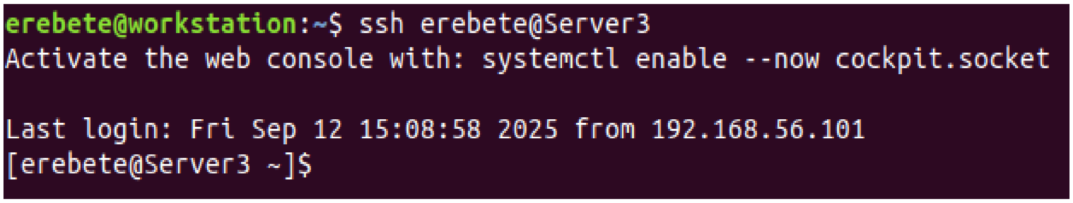
Task 4: Verify ssh remote connection

1. Using your local machine, connect to CentOS using ssh.



```
erebete@workstation:~$ ssh erebete@192.168.56.109
Activate the web console with: systemctl enable --now cockpit.socket

Last failed login: Fri Sep 12 15:04:48 PST 2025 from 192.168.56.101 on ssh:nott
y
There was 1 failed login attempt since the last successful login.
Last login: Fri Sep 12 14:59:22 2025
[erebete@Server3 ~]$ █
```

2. Show evidence that you are connected.



```
erebete@workstation:~$ ssh erebete@Server3
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Sep 12 15:08:58 2025 from 192.168.56.101
[erebete@Server3 ~]$
```

Reflections:

Answer the following:

1. What do you think we should look for in choosing the best distribution between Debian and Red Hat Linux distributions?
   - When choosing between Debian and Red Hat Linux distributions, you should consider some factors such as your intended use case, the type of package management system, the level of system stability versus access to newer software, the availability of commercial support, licensing preferences, and whether you value a community-driven or corporate-backed development model.

2. What are the main differences between Debian and Red Hat Linux distributions?

   - The main differences between Debian and Red Hat Linux distributions lie in their package management systems, update philosophies, and support models: Debian uses APT with debian packages and emphasizes community-driven stability, while Red Hat relies on DNF/YUM with .rpm packages and offers enterprise-grade support and tools tailored for business environments.