Name: Victor B. Ortega	**Date Performed:** 09/05/23
Course/Section: CPE31S5	**Date Submitted:** 09/09/23
Instructor: Enr. Roman Richard	**Semester and SY:** 2023-2024

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.

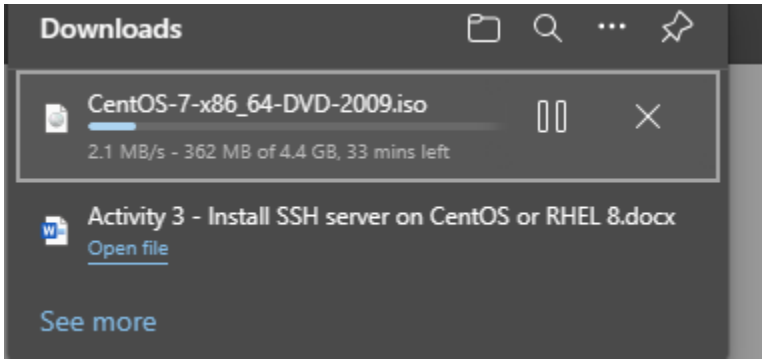
CentOS uses the RPM package format and YUM/DNF as the package manager.

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

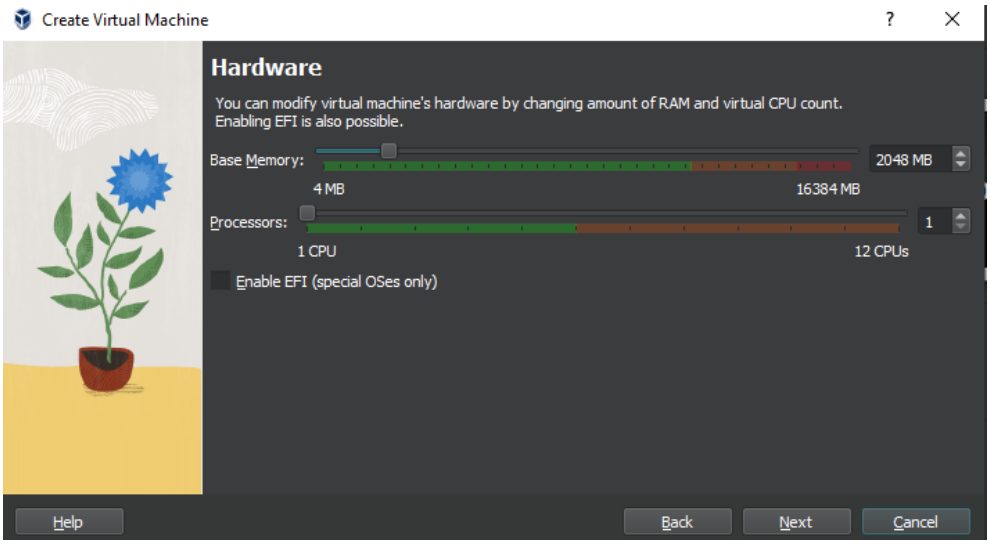
Both offer full-feature package management with network-based repository support, dependency checking and resolution, etc.. If you're familiar with one but not the other, you may have a little trouble switching over, but they're not overwhelmingly different. They both have similar features, just available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

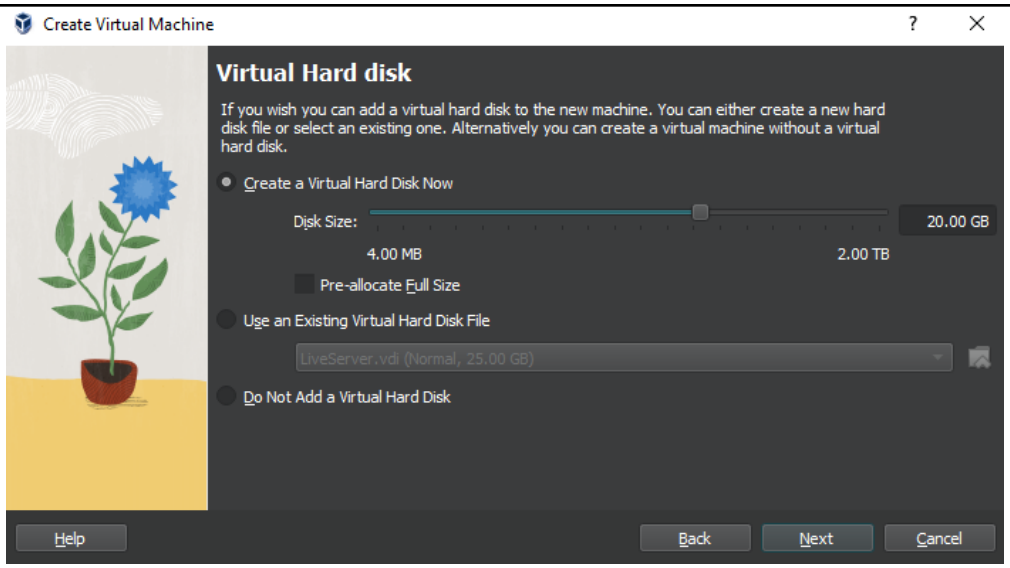
Task 1: Download the CentOS or RHEL-8 image (Create screenshots of the following)

1. Download the image of the CentOS here: http://mirror.rise.ph/centos/7.9.2009/isos/x86_64/

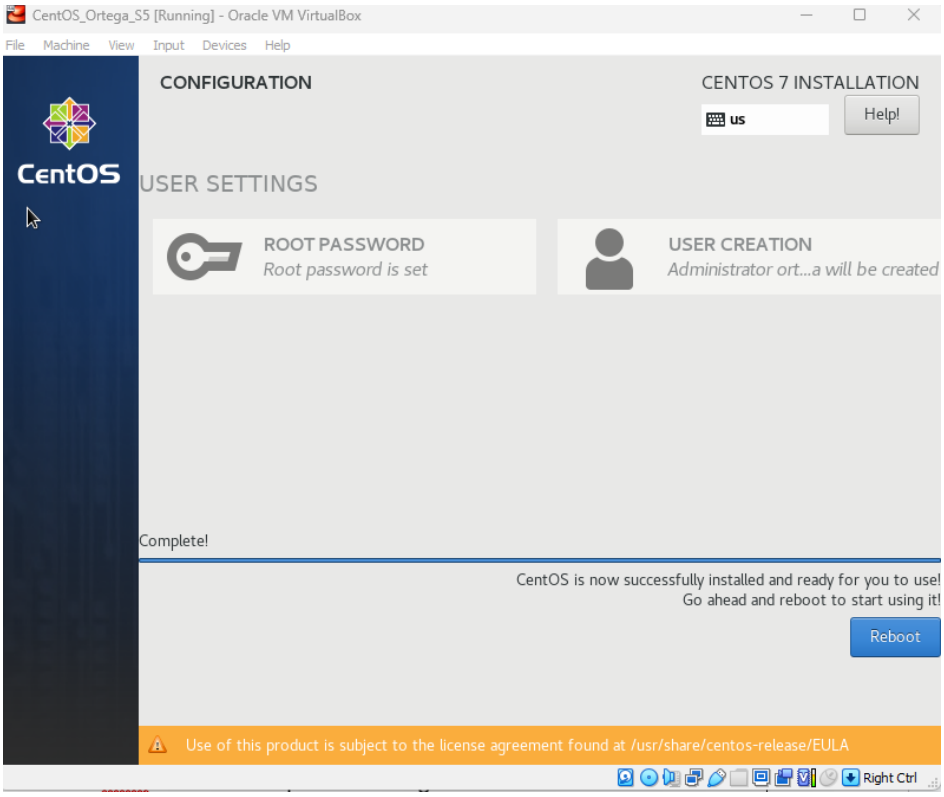


2. Create a VM machine with 2 Gb RAM and 20 Gb HD.

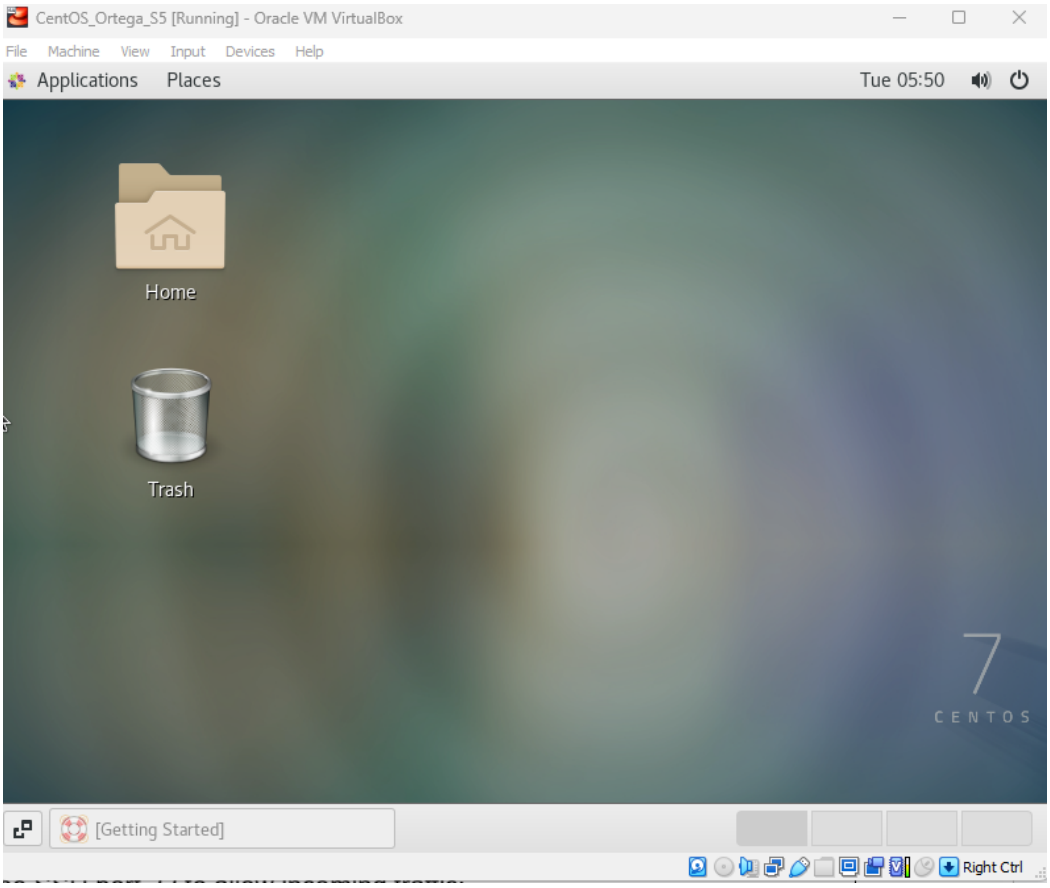




3. Install the downloaded image.



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

```
[root@localhost ortega]# dnf install openssh-server
CentOS-7 - Base                                  1.5 MB/s |  10 MB     00:06
CentOS-7 - Updates                               1.2 MB/s |  28 MB     00:22
CentOS-7 - Extras                                225 kB/s | 360 kB     00:01
Package openssh-server-7.4p1-21.el7.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

2. Start the *sshd* daemon and set to start after reboot:
   $ systemctl start sshd
   $ systemctl enable sshd

```
exit
[ortega@localhost ~]$ systemctl start sshd
[ortega@localhost ~]$ systemctl enable  sshd
```

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

```
ortega@localhost ~]$ systemctl status sshd
 sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enable
d)
   Active: active (running) since Tue 2023-09-05 06:10:23 EDT; 6min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1140 (sshd)
   CGroup: /system.slice/sshd.service
           └─1140 /usr/sbin/sshd -D

Sep 05 06:10:23 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
Sep 05 06:10:23 localhost.localdomain sshd[1140]: Server listening on 0.0.0.0 port 22.
Sep 05 06:10:23 localhost.localdomain sshd[1140]: Server listening on :: port 22.
Sep 05 06:10:23 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
Hint: Some lines were ellipsized, use -l to show in full.
[ortega@localhost ~]$
```

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

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

   $ firewall-cmd --reload

```
[ortega@localhost ~]$ firewall-cmd --reload
success
```

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

```
success
[ortega@localhost ~]$ systemctl reload sshd
[ortega@localhost ~]$
```

Task 3: Copy the Public Key to CentOS

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

```
[ortega@localhost ~]$ ssh
usage: ssh [-1246AaCfGgKkMNnqsTtVvXxYy] [-b bind_address] [-c cipher_spec]
           [-D [bind_address:]port] [-E log_file] [-e escape_char]
           [-F configfile] [-I pkcs11] [-i identity_file]
           [-J [user@]host[:port]] [-L address] [-l login_name] [-m mac_spec]
           [-O ctl_cmd] [-o option] [-p port] [-Q query_option] [-R address]
           [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
           [user@]hostname [command]
```

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

```
victor@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa ortega@192.168.56.104
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/victor/.ssh/id_rsa.pub"
The authenticity of host '192.168.56.104 (192.168.56.104)' can't be established.
ED25519 key fingerprint is SHA256:Rgbqu+oWi6joubZDSm3A2g92KNMPTeL1AqV2er3ZxkA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? 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
ortega@192.168.56.104's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ortega@192.168.56.104'"
and check to make sure that only the key(s) you wanted were added.
```

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

```
[ortega@localhost ~]$ ls -la .ssh
total 8
drwx------.  2 ortega ortega   29 Sep  5 09:39 .
drwx------. 16 ortega ortega 4096 Sep  5 09:39 ..
-rw-------.  1 ortega ortega  572 Sep  5 09:39 authorized_keys
[ortega@localhost ~]$
```

Task 4: Verify ssh remote connection

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

```
victor@workstation:~$ ssh ortega@192.168.56.104
Last login: Tue Sep  5 09:26:33 2023
[ortega@localhost ~]$
```

2. Show evidence that you are connected.

Ping CentOS

```
victor@workstation:~$ ping 192.168.56.104
PING 192.168.56.104 (192.168.56.104) 56(84) bytes of data.
64 bytes from 192.168.56.104: icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from 192.168.56.104: icmp_seq=2 ttl=64 time=0.299 ms
64 bytes from 192.168.56.104: icmp_seq=3 ttl=64 time=0.364 ms
64 bytes from 192.168.56.104: icmp_seq=4 ttl=64 time=0.308 ms
64 bytes from 192.168.56.104: icmp_seq=5 ttl=64 time=0.588 ms
64 bytes from 192.168.56.104: icmp_seq=6 ttl=64 time=0.560 ms
^Z
[5]+  Stopped                 ping 192.168.56.104
```

Ping Workstation

```
[ortega@localhost ~]$ ping 192.168.56.101
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.
64 bytes from 192.168.56.101: icmp_seq=1 ttl=64 time=0.233 ms
64 bytes from 192.168.56.101: icmp_seq=2 ttl=64 time=0.351 ms
64 bytes from 192.168.56.101: icmp_seq=3 ttl=64 time=0.291 ms
64 bytes from 192.168.56.101: icmp_seq=4 ttl=64 time=0.514 ms
^Z
[1]+  Stopped                 ping 192.168.56.101
```

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?

Answer: Consider your specific needs and preferences. Debian tends to prioritize stability and free software, while Red Hat offers enterprise support and a more standardized ecosystem.

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

Answer: CentOS and Debian stand out as widely-used Linux distributions with notable resemblances and distinctions. CentOS serves as a cost-free derivative of Red Hat's enterprise Linux, whereas Debian functions as the foundational open-source distribution from which other variants, such as Ubuntu, originate.

Conclusion:

CentOS stands out as a strong SSH choice in enterprise setups, thanks to its renowned stability and enduring support. It benefits from the backing of Red Hat, ensuring a secure and dependable environment for SSH use in critical business applications.

On the flip side, Debian presents a versatile, community-driven distribution that also proves effective for SSH purposes. It grants you the freedom to tailor and adjust as needed, making it applicable to various scenarios, including SSH use. However, it lacks the commercial backing and the same level of predictability found in CentOS.

Ultimately, your decision between CentOS and Debian for SSH should align with your precise needs. If stability, support, and a clearly defined release schedule are paramount, CentOS emerges as a dependable choice. Conversely, if you prize adaptability and uphold open-source values, Debian could emerge as the favored option.