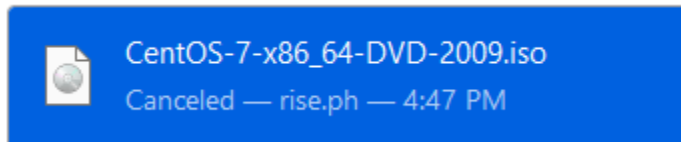


Name: Niemo, Christian Al C.	Date Performed: Sep. 7, 2023
Course/Section: CPE 232 - CPE31S6	Date Submitted: Sep. 13, 2023
Instructor: Dr. Jonathan V. Taylar	Semester and SY: 1st Sem - 2023-2024
Activity 3: Install SSH server on CentOS or RHEL 8	
1. Objectives: 1.1 Install Community Enterprise OS or Red 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.	

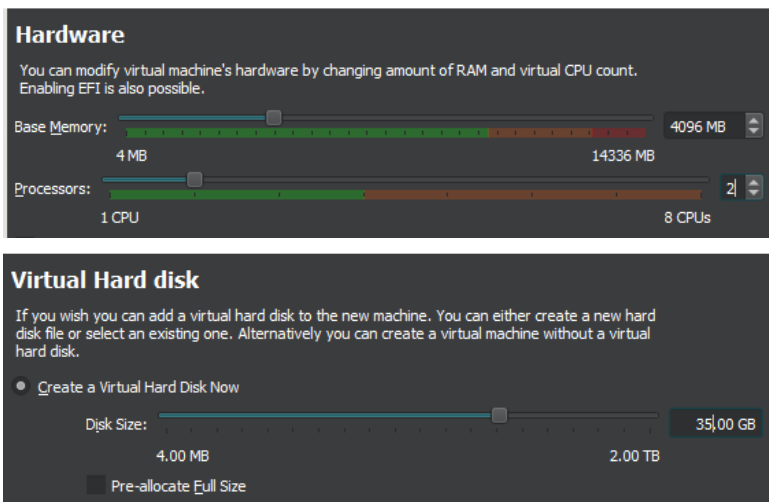
Both offer full-feature package management with network-based repository support, dependency checking and resolution, etc.. If you're familiar with one but not the other, you may have a little trouble switching over, but they're not overwhelmingly different. They both have similar features, just available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

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

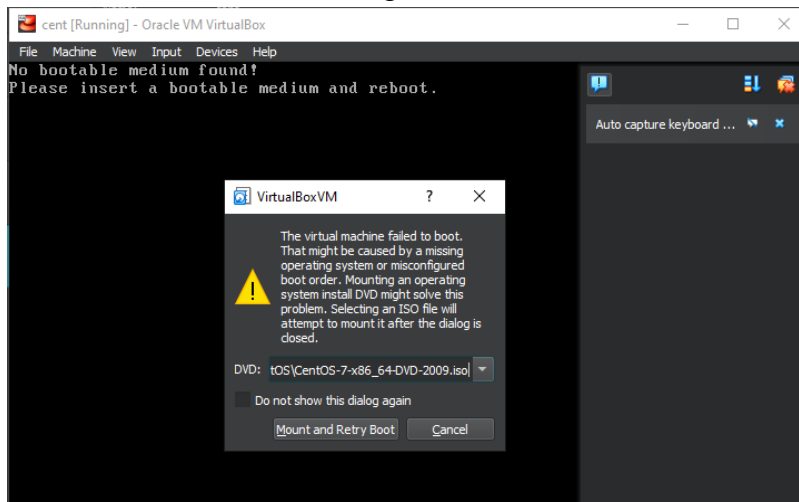
1. Download the image of the CentOS here:
http://mirror.rise.ph/centos/7.9.2009/isos/x86_64/



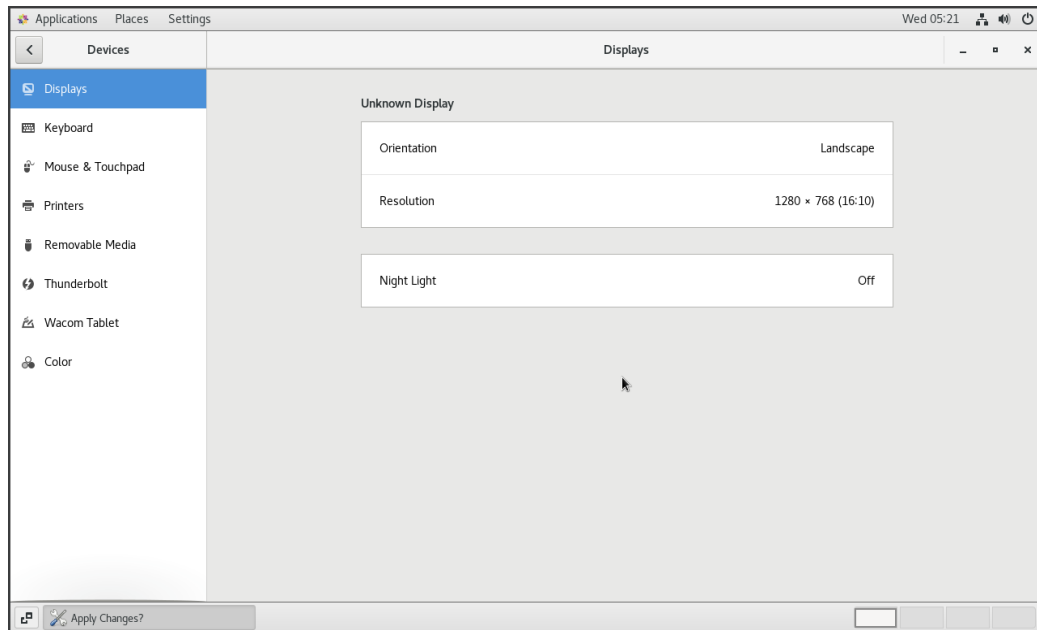
2. Create a VM machine with 2 Gb RAM and 20 Gb HD.



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

```
[workstation@localhost ~]$ yum install openssh-server
Loaded plugins: fastestmirror, langpacks
You need to be root to perform this command.
[workstation@localhost ~]$ sudo dnf install openssh-server

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for workstation:
Sorry, try again.
[sudo] password for workstation:
CentOS-7 - Base                    5.8 MB/s | 10 MB      00:01
CentOS-7 - Updates                 7.4 MB/s | 28 MB      00:03
CentOS-7 - Extras                  803 kB/s | 360 kB     00:00
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

```
[workstation@localhost ~]$ systemctl start sshd
```

\$ systemctl enable sshd

```
[workstation@localhost ~]$ systemctl enable sshd
```

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

\$ systemctl status sshd

```
[workstation@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 Thu 2023-09-07 05:30:23 EDT; 6min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
    Main PID: 4768 (sshd)
      CGroup: /system.slice/sshd.service
              └─4768 /usr/sbin/sshd -D

Sep 07 05:30:23 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
Sep 07 05:30:23 localhost.localdomain sshd[4768]: Server listening on 0.0.0.0 port 22.
Sep 07 05:30:23 localhost.localdomain sshd[4768]: Server listening on :: port 22.
Sep 07 05:30:23 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
Hint: Some lines were ellipsized, use -l to show in full.
```

4. Open the SSH port 22 to allow incoming traffic:

\$ firewall-cmd --zone=public --permanent --add-service=ssh

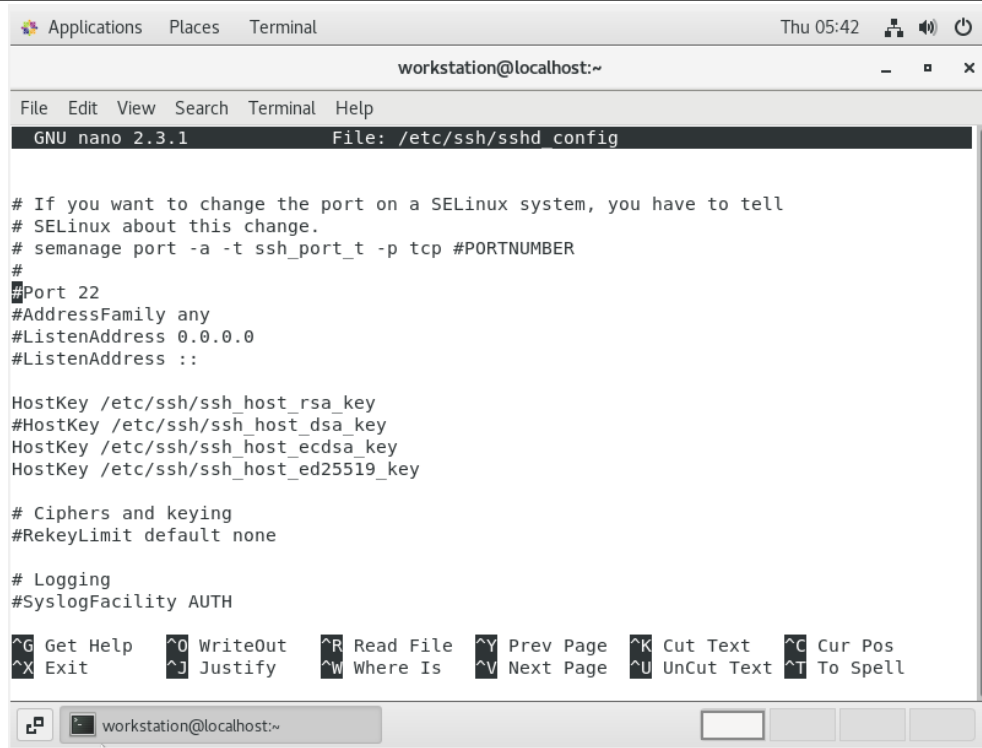
```
[workstation@localhost ~]$ firewall-cmd --zone=public --permanent --add-service=ssh
Warning: ALREADY_ENABLED: ssh
success
```

\$ firewall-cmd --reload

```
[workstation@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



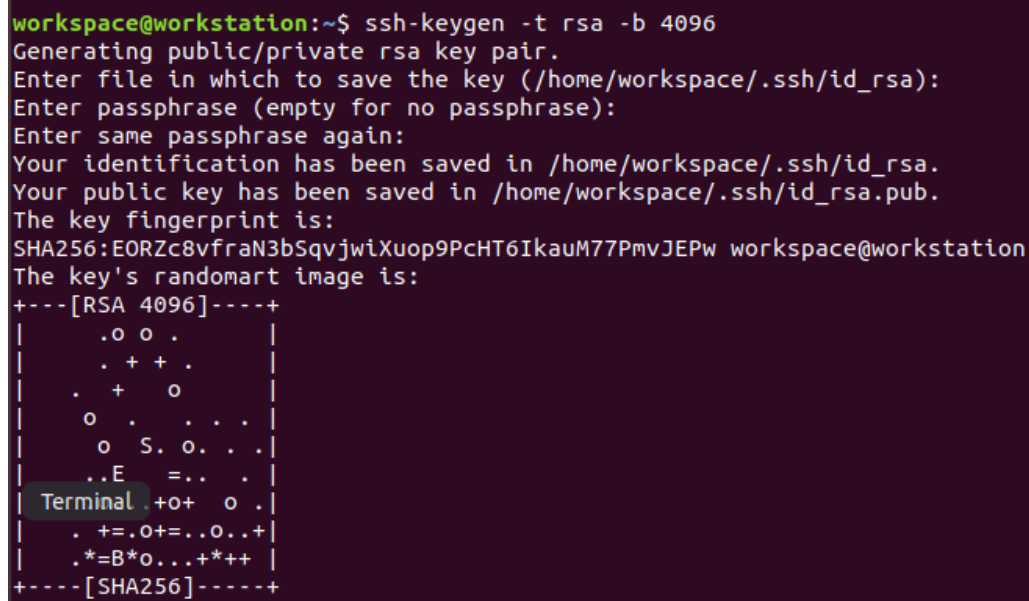
The screenshot shows a terminal window titled 'workstation@localhost:~'. The terminal is running the GNU nano 2.3.1 editor, editing the file /etc/ssh/sshd_config. The file content includes comments about SELinux, port configuration (Port 22), address family, listen address, host keys, ciphers, and logging. At the bottom, there is a status bar with various keyboard shortcuts like ^G Get Help, ^O WriteOut, ^R Read File, ^Y Prev Page, ^K Cut Text, ^C Cur Pos, ^X Exit, ^J Justify, ^W Where Is, ^V Next Page, ^U UnCut Text, and ^T To Spell.

```
workstation@localhost:~  
File Edit View Search Terminal Help  
GNU nano 2.3.1 File: /etc/ssh/sshd_config  
  
# 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  
#  
#Port 22  
#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  
  
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos  
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

```
[workstation@localhost ~]$ systemctl reload sshd
```

Task 3: Copy the Public Key to CentOS

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



The screenshot shows a terminal window with the command 'ssh-keygen -t rsa -b 4096' being executed. The output shows the generation of a public/private RSA key pair, the file path (/home/workspace/.ssh/id_rsa), the passphrase (empty), and the key fingerprint (SHA256:E0RZc8vfraN3bSqvwjXuop9PcHT6IkauM77PmvJEPw workstation@workstation). A randomart image is also displayed.

```
workspace@workstation:~$ ssh-keygen -t rsa -b 4096  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/workspace/.ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/workspace/.ssh/id_rsa.  
Your public key has been saved in /home/workspace/.ssh/id_rsa.pub.  
The key fingerprint is:  
SHA256:E0RZc8vfraN3bSqvwjXuop9PcHT6IkauM77PmvJEPw workstation@workstation  
The key's randomart image is:  
+---[RSA 4096]-----+  
| .o o . |  
| . + + . |  
| . + o |  
| o . . . |  
| o S. o. . |  
| ..E =. . |  
| Terminal .+o+ o . |  
| . +=.O+=..O..+ |  
| .*=B*O...+*++ |  
+----[SHA256]-----+
```

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

```
workspace@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa workstation@192.168.56.109
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/workspace/.ssh/id_rsa.pub"
The authenticity of host '192.168.56.109 (192.168.56.109)' can't be established.
ECDSA key fingerprint is SHA256:XJMLmVy80wuKzJYXBWP8UWz6JqJJ0uGi32hxOxL0jc.
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
workstation@192.168.56.109's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'workstation@192.168.56.109'"
and check to make sure that only the key(s) you wanted were added.

workspace@workstation:~$
```

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

```
Desktop Documents Downloads Music Pictures Public Templates Videos
[workstation@localhost ~]$ cd ~/.ssh
[workstation@localhost .ssh]$ ls
authorized_keys
[workstation@localhost .ssh]$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQACQ7yvDYXkM+H7ZWM3BNx1FRlMURrsRDVHcwr2ZZPcI5yp
ujA/Gig7wmeW2EbJ54to5VwIZgLttzlz0xFD/kqU0VW2andFlaFz6anAWm+e2/SV6ZHZND7I9t9hxmwmrU
bEadP8RJQqvFQkN5zQR7d1VVWabyrhZ6RprxJ6Yp08n7pqR9T9e3tIqpmWtZLNiZo05z1MDFT0XxeUbIlt
VI++DRt0K0mEQhq+QB77TwW9IiTjJ0lwCDcERvydN15MJfXV9KZ9opAtKy0YcXU3pp/Jw/1/4jwmKV369d
GarupobuEp22iypT6hplVRXKMRFGN+D6Q5I7hk3xdymyQoE34GvLcct92au3gcLZ+KI+6WU9ukZzlkuZeB
X7en9s1DxiJ5L6u8TyT4+Lep3Qc5T3cZnWqpXphqPH7a03RwxsCBG7zN/0stgxUBG8RMK0s9/eIcVl7jHV
M56MY33I8m/2JCNLC41ZUdL3rARHBxX0y1NlRepMsmuilt/x9VrecxgvfsS52y8Ni/gYwZVRfThmWedeFS
Aw9TFYUpy7iaesmpkIJW3ti7SLt+patdEy+07V8QHPXJzRgiGNi1ArRkBwBXDTqsyZkd+X+5i2oHEDoF6/
KnmMFPfqa1a2N5GVH8Hwr004sQ== workstation
```

Task 4: Verify ssh remote connection

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

```
workspace@workstation:~$ ssh workstation@192.168.56.109

Last login: Thu Sep  7 05:08:46 2023

[workstation@localhost ~]$
[workstation@localhost ~]$
```

2. Show evidence that you are connected.

```
[workstation@localhost ~]$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
```

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 deciding between Debian and Red Hat Linux distributions, it's important to consider factors such as your specific use case, the level of support you require, your familiarity with package management systems, the balance between stability and cutting-edge features, licensing preferences, integration with ecosystems, community vs. commercial backing, budget constraints, security considerations, and hardware compatibility. Ultimately, your choice should align with your specific needs and priorities, as both Debian and Red Hat have their strengths and are widely used in various scenarios.
2. What are the main differences between Debian and Red Hat Linux distributions?
 - Debian and Red Hat Linux distributions differ in their origins and licensing, package management systems, support structures, stability versus cutting-edge features, ecosystems, costs, security approaches, community versus commercial backing, and their release cycles, with Debian being community-driven and emphasizing free software principles, while Red Hat offers a commercial product, such as Red Hat Enterprise Linux (RHEL), with a balanced approach to stability and new features, backed by commercial support options and a well-established ecosystem.

Conclusion:

As a student, accomplishing the objectives of installing either CentOS or Red Hat Linux OS and configuring a remote SSH connection holds significant importance in building a robust Linux environment. The decision between CentOS, a community-supported option, and Red Hat, which offers commercial support, should be made with factors like support availability, stability, and licensing in mind. Configuring SSH access is essential for secure remote management, and it involves implementing security best practices such as shifting from password-based to key-based authentication and establishing firewall rules to restrict SSH access to trusted IP addresses. These objectives not only serve as foundational steps in Linux system administration but also contribute to improved security and the overall reliability of the Linux infrastructure, making them crucial tasks for a student working with Linux systems.