Name: Daniela Marie D. Rabang	**Date Performed:** 08/15/2023
Course/Section: CPE232/CPE31S4	**Date Submitted:** 08/15/2023
Instructor: Engr. Jonathan Taylar	**Semester and SY:** 1st Sem SY 2023-2024

Activity 1: Configure Network using Virtual Machines

1. Objectives:

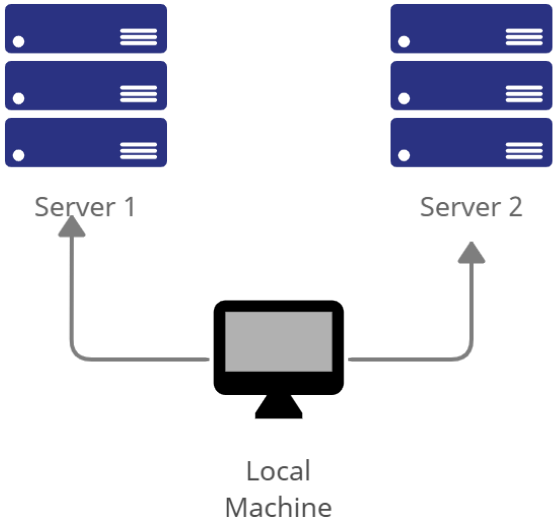
1.1. Create and configure Virtual Machines in Microsoft Azure or VirtualBox

1.2. Set-up a Virtual Network and Test Connectivity of VMs

2. Discussion:

Network Topology:

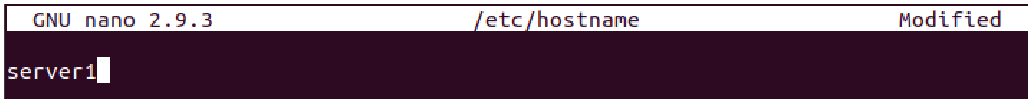
Assume that you have created the following network topology in Virtual Machines, *provide screenshots for each task*. (Note: *it is assumed that you have prior knowledge of cloning and creating snapshots in a virtual machine*).



Task 1: Do the following on Server 1, Server 2, and Local Machine. In editing the file using nano command, press control + O to write out (save the file). Press enter when asked for the name of the file. Press control + X to end.

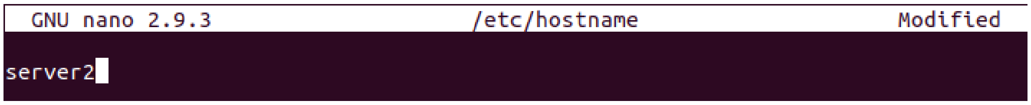
1. Change the hostname using the command *sudo nano /etc/hostname*

   1.1 Use server1 for Server 1



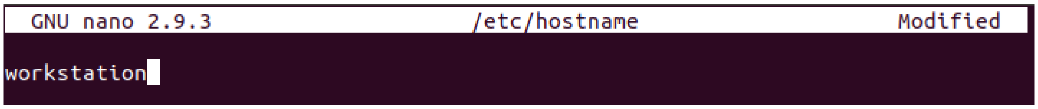
```
  GNU nano 2.9.3                  /etc/hostname                  Modified

server1
```

   1.2 Use server2 for Server 2



```
  GNU nano 2.9.3                  /etc/hostname                  Modified

server2
```

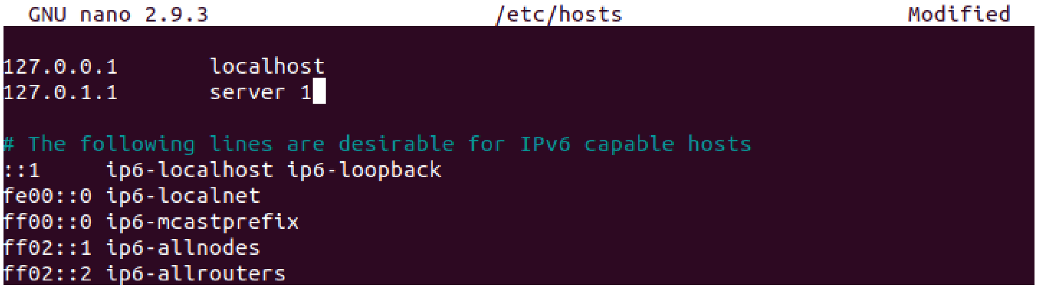
1.3 Use workstation for the Local Machine



```
  GNU nano 2.9.3                    /etc/hostname                    Modified

workstation
```

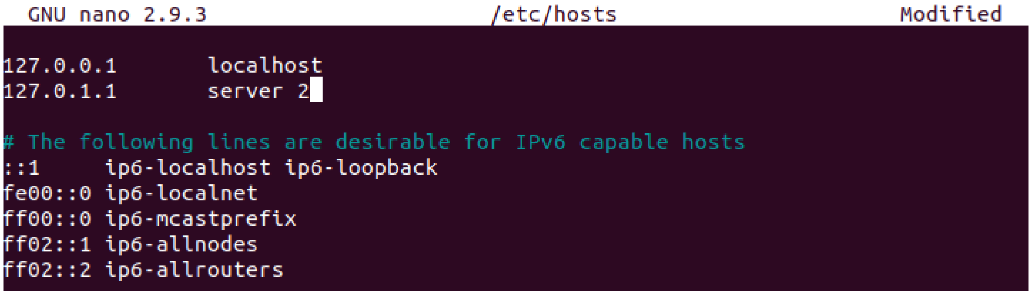
2. Edit the hosts using the command *sudo nano /etc/hosts*. Edit the second line.

2.1 Type 127.0.0.1 server 1 for Server 1



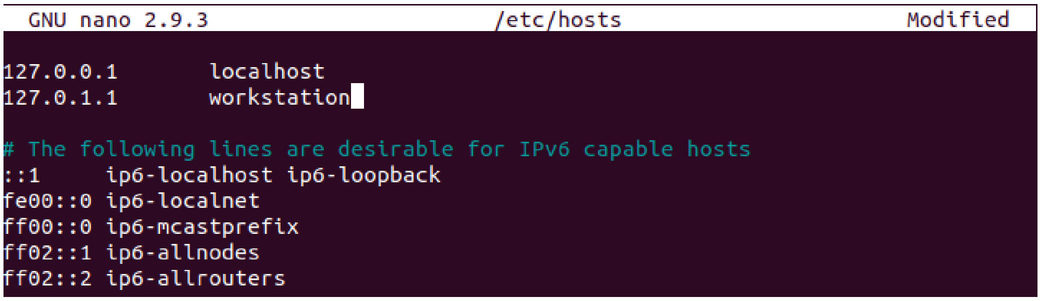
```
  GNU nano 2.9.3                      /etc/hosts                      Modified

127.0.0.1       localhost
127.0.1.1       server 1

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

2.2 Type 127.0.0.1 server 2 for Server 2



```
  GNU nano 2.9.3                      /etc/hosts                      Modified

127.0.0.1       localhost
127.0.1.1       server 2

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

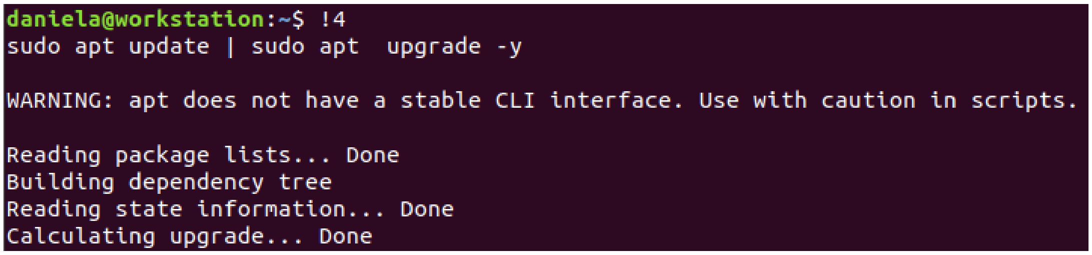
2.3 Type 127.0.0.1 workstation for the Local Machine



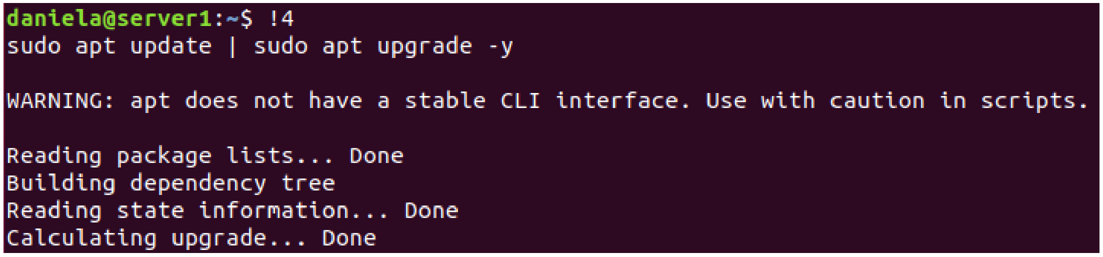
```
  GNU nano 2.9.3                      /etc/hosts                      Modified

127.0.0.1       localhost
127.0.1.1       workstation

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Task 2: Configure SSH on Server 1, Server 2, and Local Machine. Do the following:

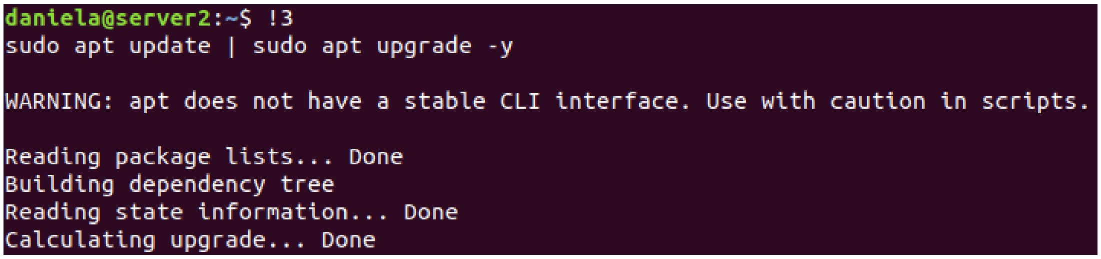
1. Upgrade the packages by issuing the command *sudo apt update* and *sudo apt upgrade* respectively.



```
daniela@workstation:~$ !4
sudo apt update | sudo apt  upgrade -y

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
```

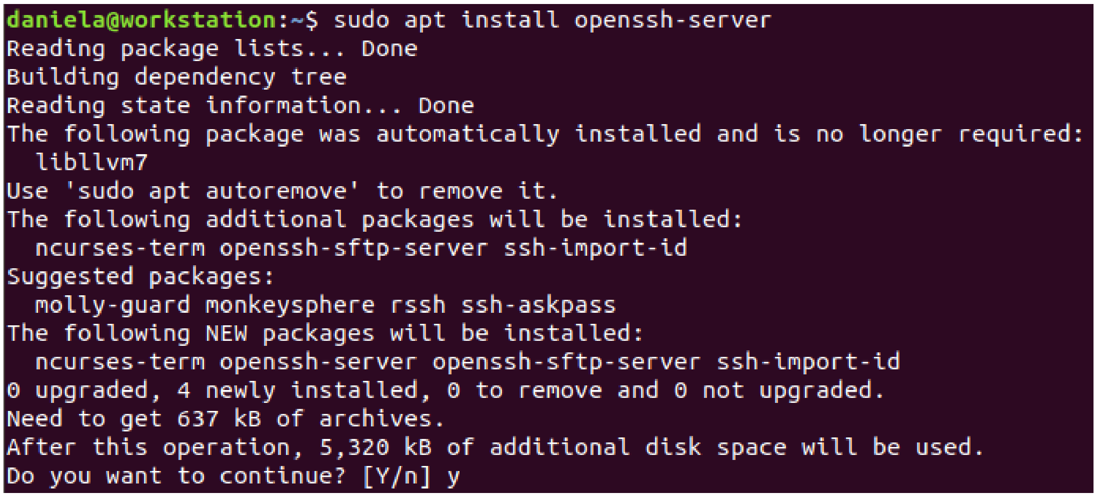


```
daniela@server1:~$ !4
sudo apt update | sudo apt upgrade -y

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
```

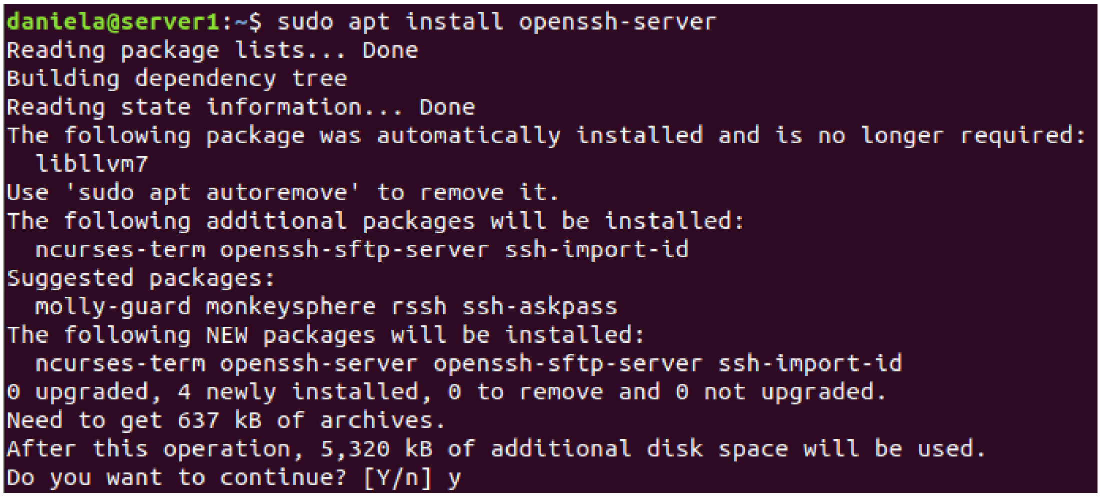


```
daniela@server2:~$ !3
sudo apt update | sudo apt upgrade -y

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
```

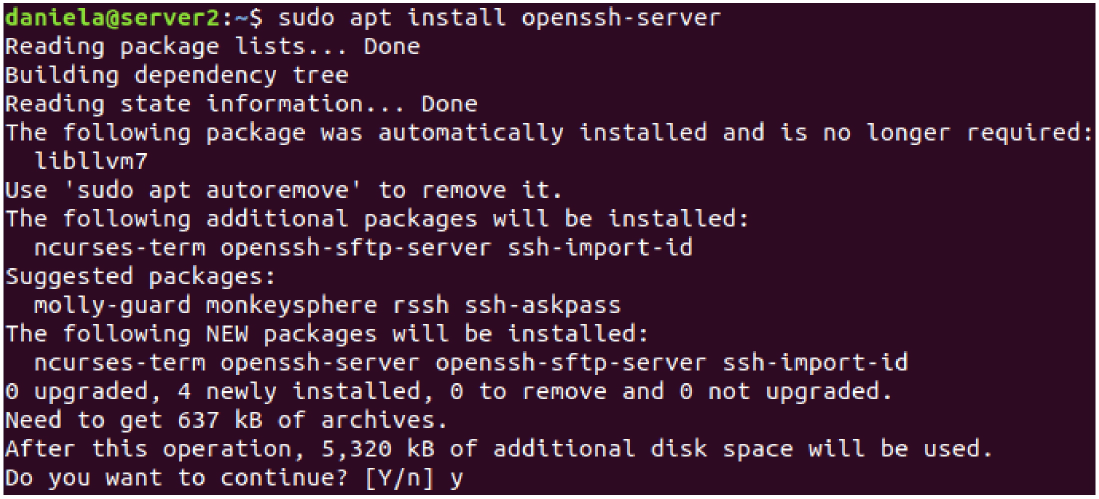
2. Install the SSH server using the command *sudo apt install openssh-server*.



```
daniela@workstation:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere rssh ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 637 kB of archives.
After this operation, 5,320 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```



```
daniela@server1:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere rssh ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 637 kB of archives.
After this operation, 5,320 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

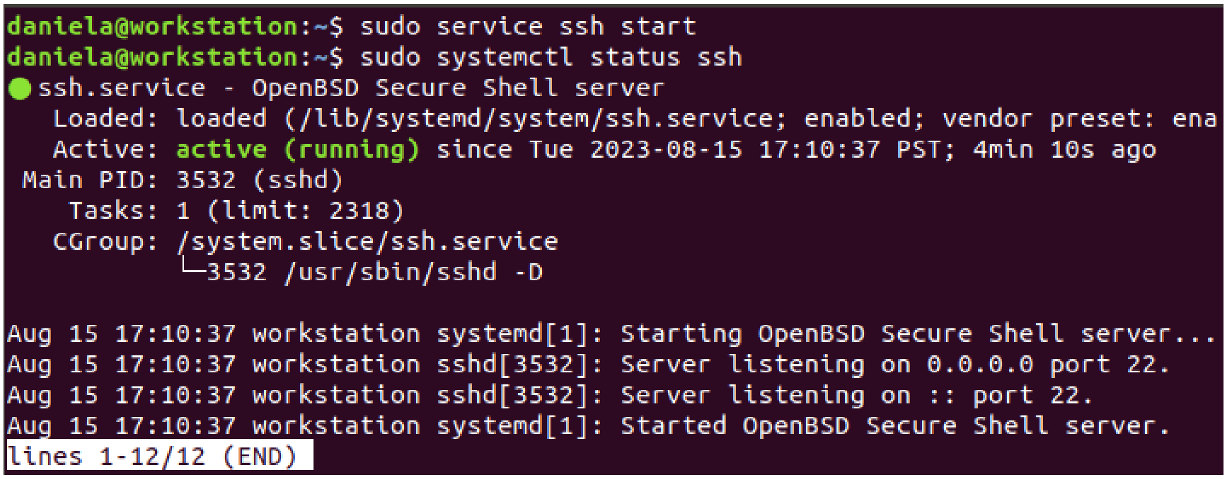


```
daniela@server2:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere rssh ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 637 kB of archives.
After this operation, 5,320 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

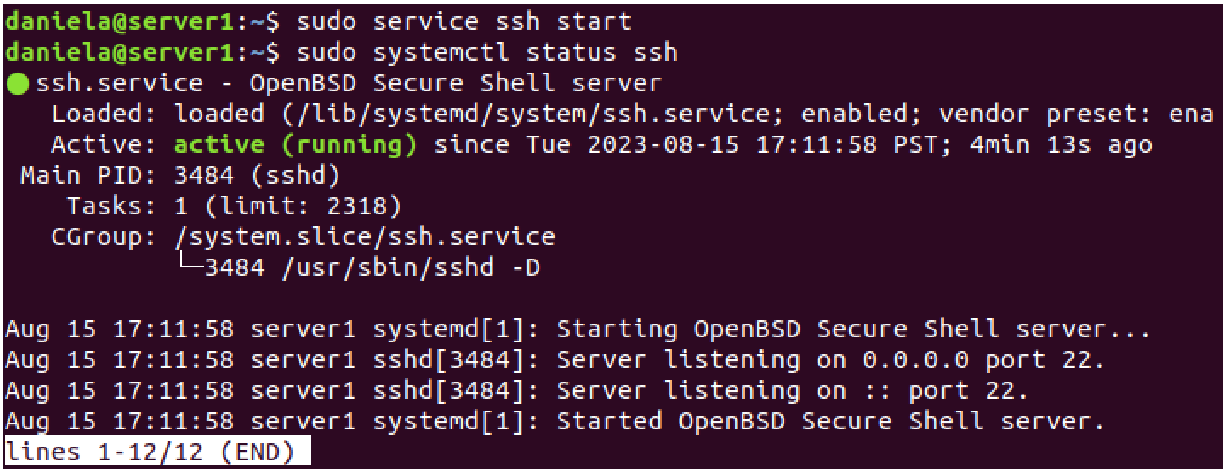
3. Verify if the SSH service has started by issuing the following commands:

   3.1 *sudo service ssh start*

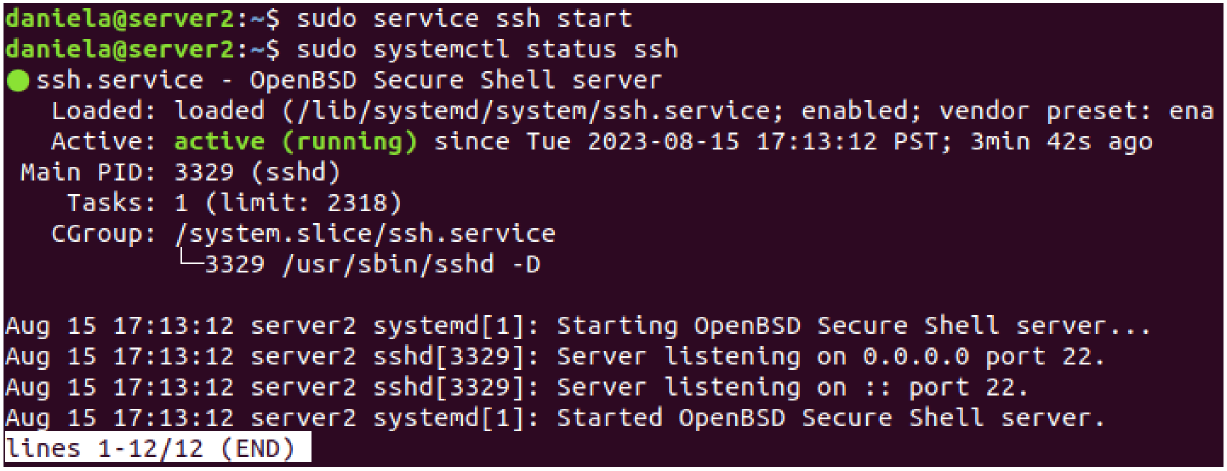
   3.2 *sudo systemctl status ssh*



```
daniela@workstation:~$ sudo service ssh start
daniela@workstation:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: ena
   Active: active (running) since Tue 2023-08-15 17:10:37 PST; 4min 10s ago
 Main PID: 3532 (sshd)
    Tasks: 1 (limit: 2318)
   CGroup: /system.slice/ssh.service
           └─3532 /usr/sbin/sshd -D

Aug 15 17:10:37 workstation systemd[1]: Starting OpenBSD Secure Shell server...
Aug 15 17:10:37 workstation sshd[3532]: Server listening on 0.0.0.0 port 22.
Aug 15 17:10:37 workstation sshd[3532]: Server listening on :: port 22.
Aug 15 17:10:37 workstation systemd[1]: Started OpenBSD Secure Shell server.
lines 1-12/12 (END)
```

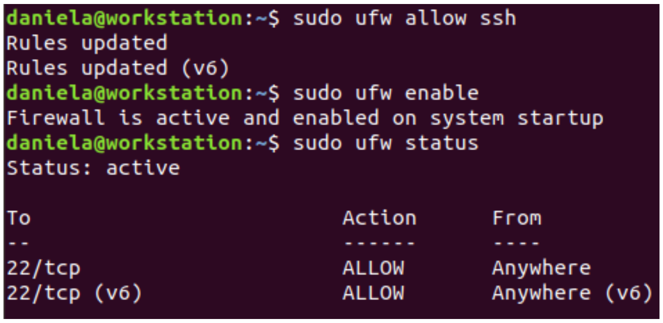


```
daniela@server1:~$ sudo service ssh start
daniela@server1:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: ena
   Active: active (running) since Tue 2023-08-15 17:11:58 PST; 4min 13s ago
 Main PID: 3484 (sshd)
    Tasks: 1 (limit: 2318)
   CGroup: /system.slice/ssh.service
           └─3484 /usr/sbin/sshd -D

Aug 15 17:11:58 server1 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 15 17:11:58 server1 sshd[3484]: Server listening on 0.0.0.0 port 22.
Aug 15 17:11:58 server1 sshd[3484]: Server listening on :: port 22.
Aug 15 17:11:58 server1 systemd[1]: Started OpenBSD Secure Shell server.
lines 1-12/12 (END)
```

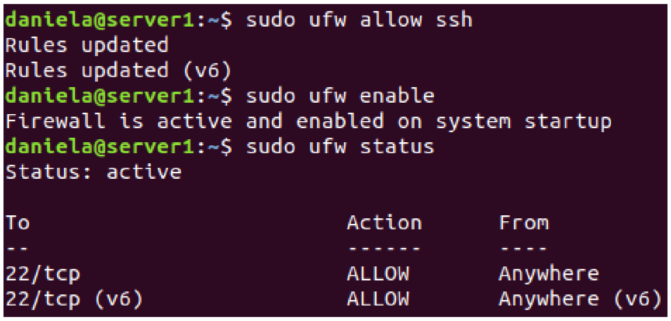


```
daniela@server2:~$ sudo service ssh start
daniela@server2:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: ena
   Active: active (running) since Tue 2023-08-15 17:13:12 PST; 3min 42s ago
 Main PID: 3329 (sshd)
    Tasks: 1 (limit: 2318)
   CGroup: /system.slice/ssh.service
           └─3329 /usr/sbin/sshd -D

Aug 15 17:13:12 server2 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 15 17:13:12 server2 sshd[3329]: Server listening on 0.0.0.0 port 22.
Aug 15 17:13:12 server2 sshd[3329]: Server listening on :: port 22.
Aug 15 17:13:12 server2 systemd[1]: Started OpenBSD Secure Shell server.
lines 1-12/12 (END)
```

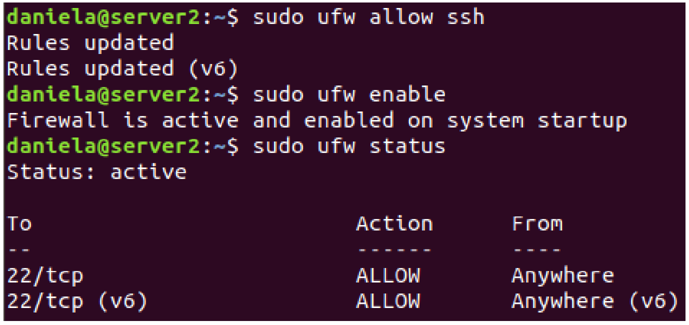
4. Configure the firewall to all port 22 by issuing the following commands:
    4.1 *sudo ufw allow ssh*
    4.2 *sudo ufw enable*
    4.3 *sudo ufw status*



```
daniela@workstation:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
daniela@workstation:~$ sudo ufw enable
Firewall is active and enabled on system startup
daniela@workstation:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```



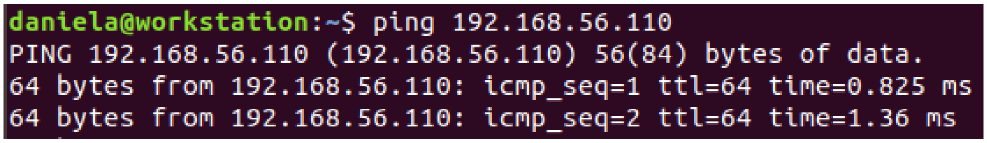
```
daniela@server1:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
daniela@server1:~$ sudo ufw enable
Firewall is active and enabled on system startup
daniela@server1:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```



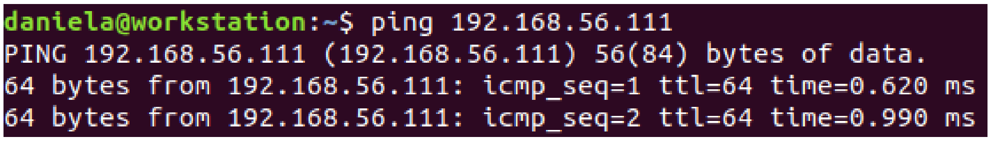
```
daniela@server2:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
daniela@server2:~$ sudo ufw enable
Firewall is active and enabled on system startup
daniela@server2:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```

Task 3: Verify network settings on Server 1, Server 2, and Local Machine. On each device, do the following:

1. Record the ip address of Server 1, Server 2, and Local Machine. Issue the command *ifconfig* and check network settings. Note that the ip addresses of all the machines are in this network 192.168.56.XX.
   1.1 Server 1 IP address: 192.168.56.**109**
   1.2 Server 2 IP address: 192.168.56.**110**
   1.3 Server 3 IP address: 192.168.56.**111**
2. Make sure that they can ping each other.
   2.1 Connectivity test for Local Machine 1 to Server 1: □ Successful □ Not Successful

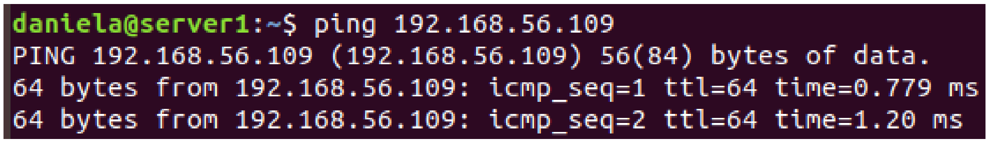


```
daniela@workstation:~$ ping 192.168.56.110
PING 192.168.56.110 (192.168.56.110) 56(84) bytes of data.
64 bytes from 192.168.56.110: icmp_seq=1 ttl=64 time=0.825 ms
64 bytes from 192.168.56.110: icmp_seq=2 ttl=64 time=1.36 ms
```

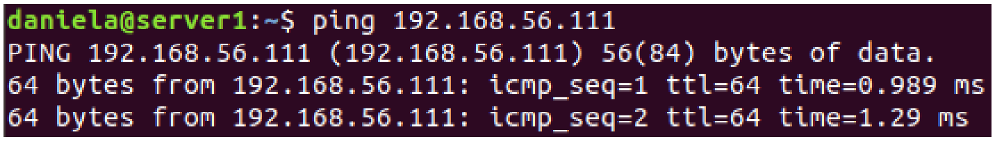


```
daniela@workstation:~$ ping 192.168.56.111
PING 192.168.56.111 (192.168.56.111) 56(84) bytes of data.
64 bytes from 192.168.56.111: icmp_seq=1 ttl=64 time=0.620 ms
64 bytes from 192.168.56.111: icmp_seq=2 ttl=64 time=0.990 ms
```

   2.2 Connectivity test for Local Machine 1 to Server 2: □ Successful □ Not Successful

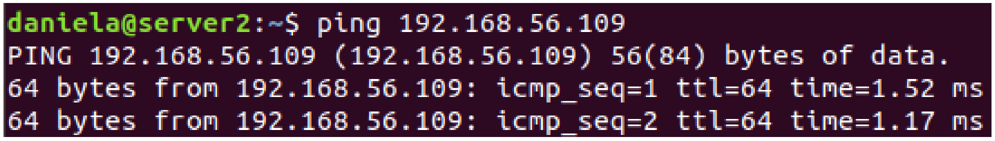


```
daniela@server1:~$ ping 192.168.56.109
PING 192.168.56.109 (192.168.56.109) 56(84) bytes of data.
64 bytes from 192.168.56.109: icmp_seq=1 ttl=64 time=0.779 ms
64 bytes from 192.168.56.109: icmp_seq=2 ttl=64 time=1.20 ms
```

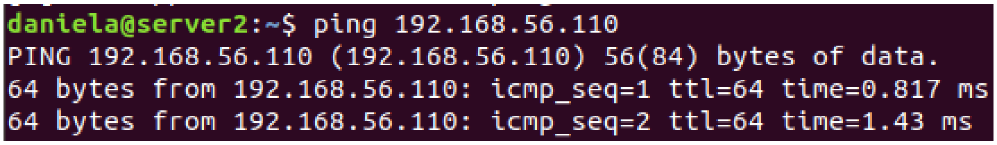


```
daniela@server1:~$ ping 192.168.56.111
PING 192.168.56.111 (192.168.56.111) 56(84) bytes of data.
64 bytes from 192.168.56.111: icmp_seq=1 ttl=64 time=0.989 ms
64 bytes from 192.168.56.111: icmp_seq=2 ttl=64 time=1.29 ms
```

   2.3 Connectivity test for Server 1 to Server 2: □ Successful □ NoSuccessful



```
daniela@server2:~$ ping 192.168.56.109
PING 192.168.56.109 (192.168.56.109) 56(84) bytes of data.
64 bytes from 192.168.56.109: icmp_seq=1 ttl=64 time=1.52 ms
64 bytes from 192.168.56.109: icmp_seq=2 ttl=64 time=1.17 ms
```



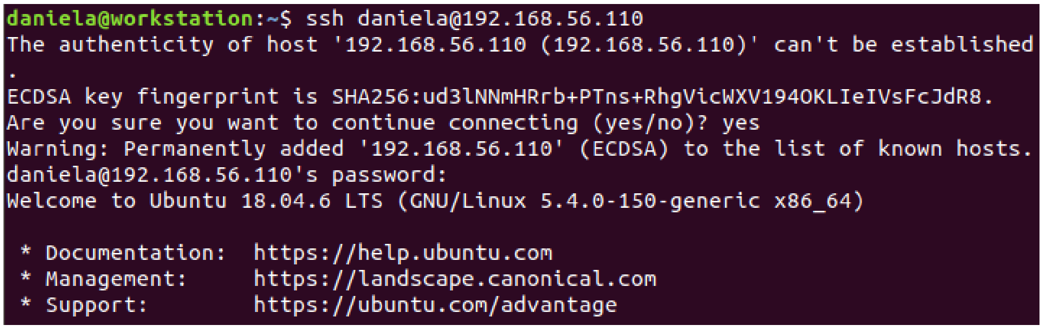
```
daniela@server2:~$ ping 192.168.56.110
PING 192.168.56.110 (192.168.56.110) 56(84) bytes of data.
64 bytes from 192.168.56.110: icmp_seq=1 ttl=64 time=0.817 ms
64 bytes from 192.168.56.110: icmp_seq=2 ttl=64 time=1.43 ms
```

Task 4: Verify SSH connectivity on Server 1, Server 2, and Local Machine.

1. On the Local Machine, issue the following commands:

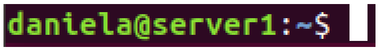
1.1 ssh username@ip_address_server1 for example, *ssh jvtaylar@192.168.56.120*

1.2 Enter the password for server 1 when prompted



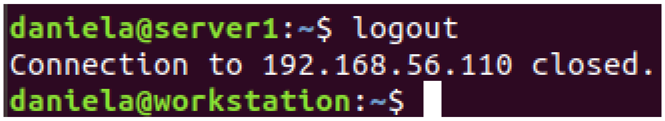
```
daniela@workstation:~$ ssh daniela@192.168.56.110
The authenticity of host '192.168.56.110 (192.168.56.110)' can't be established
.
ECDSA key fingerprint is SHA256:ud3lNNmHRrb+PTns+RhgVicWXV194OKLIeIVsFcJdR8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.110' (ECDSA) to the list of known hosts.
daniela@192.168.56.110's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

1.3 Verify that you are in server 1. The user should be in this format user@server1. For example, *jvtaylar@server1*



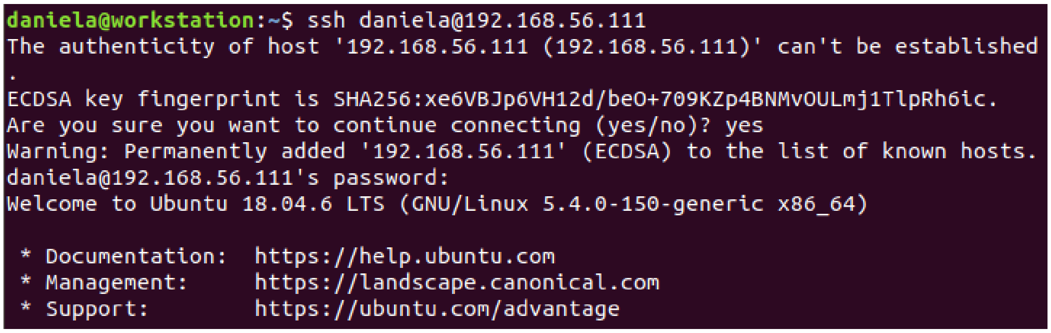
```
daniela@server1:~$
```

2. Logout of Server 1 by issuing the command *control + D.*

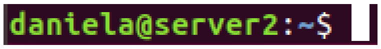


```
daniela@server1:~$ logout
Connection to 192.168.56.110 closed.
daniela@workstation:~$
```

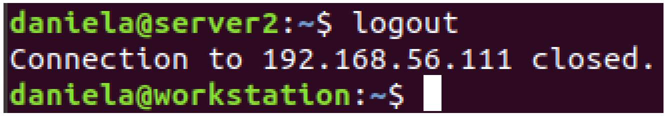
3. Do the same for Server 2.



```
daniela@workstation:~$ ssh daniela@192.168.56.111
The authenticity of host '192.168.56.111 (192.168.56.111)' can't be established
.
ECDSA key fingerprint is SHA256:xe6VBJp6VH12d/beO+709KZp4BNMvOULmj1TlpRh6ic.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.111' (ECDSA) to the list of known hosts.
daniela@192.168.56.111's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```



```
daniela@server2:~$
```



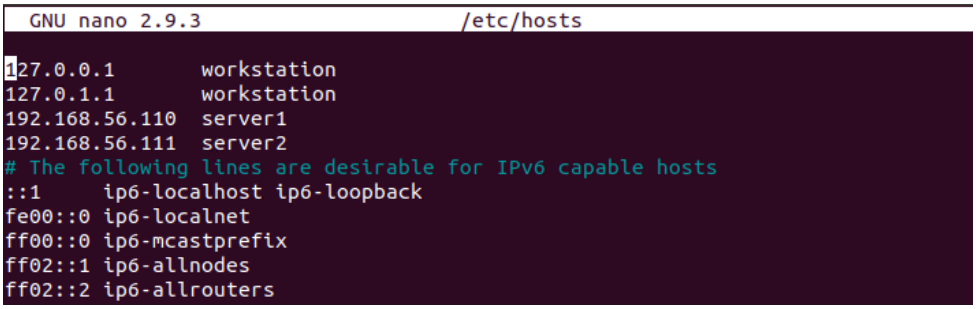
```
daniela@server2:~$ logout
Connection to 192.168.56.111 closed.
daniela@workstation:~$
```

4. Edit the hosts of the Local Machine by issuing the command *sudo nano /etc/hosts.* Below all texts type the following:

4.1 IP_address server 1 (provide the ip address of server 1 followed by the hostname)

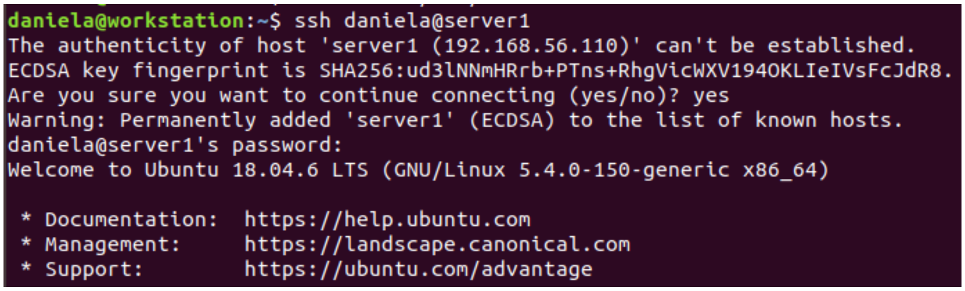
4.2 IP_address server 2 (provide the ip address of server 2 followed by the hostname)

4.3 Save the file and exit.

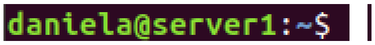


```
  GNU nano 2.9.3                        /etc/hosts

127.0.0.1       workstation
127.0.1.1       workstation
192.168.56.110  server1
192.168.56.111  server2
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

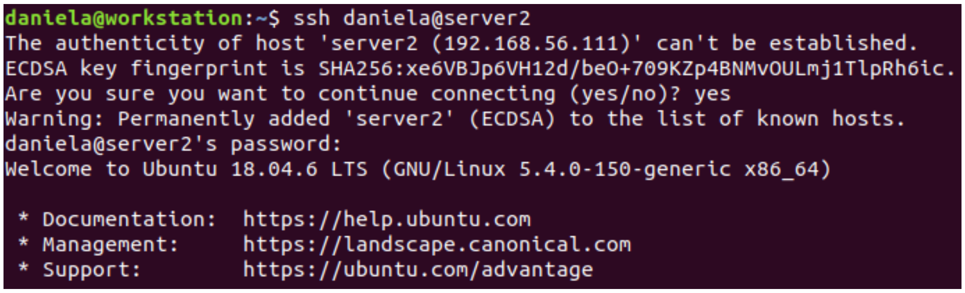
5. On the local machine, verify that you can do the SSH command but this time, use the hostname instead of typing the IP address of the servers. For example, try to do *ssh jvtaylar@server1*. Enter the password when prompted. Verify that you have entered Server 1. Do the same for Server 2.



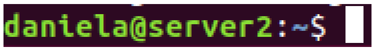
```
daniela@workstation:~$ ssh daniela@server1
The authenticity of host 'server1 (192.168.56.110)' can't be established.
ECDSA key fingerprint is SHA256:ud3lNNmHRrb+PTns+RhgVicWXV194OKLIeIVsFcJdR8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server1' (ECDSA) to the list of known hosts.
daniela@server1's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```



```
daniela@server1:~$
```



```
daniela@workstation:~$ ssh daniela@server2
The authenticity of host 'server2 (192.168.56.111)' can't be established.
ECDSA key fingerprint is SHA256:xe6VBJp6VH12d/beO+709KZp4BNMvOULmj1TlpRh6ic.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server2' (ECDSA) to the list of known hosts.
daniela@server2's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```



```
daniela@server2:~$
```

Reflections:

Answer the following:

1. How are we able to use the hostname instead of IP address in SSH commands?
    - We are able to use the hostname instead of the IP address because we had changed the /etc/hosts and declared the hostname as well as its ip address.

2. How secured is SSH?
    - SSH is secured for the authentication process on every steps and action that the user is making.

Conclusion:

This activity contains the following task that I had done for the step by step procedure of the hands-on activity. First I cloned the main virtual machine or the local host two times so I can achieve the network topology that is shown in the first page. Then I had changed the hostname of the three by the names of server 1, server 2, and the workstation. Then for task 2 I had upgraded and updated the three and installed the SSH server as well. After I had done the upgrade and installation process I had verified if the SSH service has been started, and yes it had been started. Then I had configured the firewall to all the port 22. For task 3, first I identify the IP Address of the three and ping the two other machine for the three machine. For the last task I had verified SSH connectivity. This is all done in the in the local machine that is called the workstation.