Name: Bolivar, Deo, M.	**Date Performed:** August 26, 2022
Course/Section: CPE232/CPE31S24	**Date Submitted:** August 27, 2022
Instructor: Engr. Jonathan V. Taylar	**Semester and SY:** 1st sem - 3rd year

Activity 1: Configure Network using Virtual Machines

1. **Objectives:**

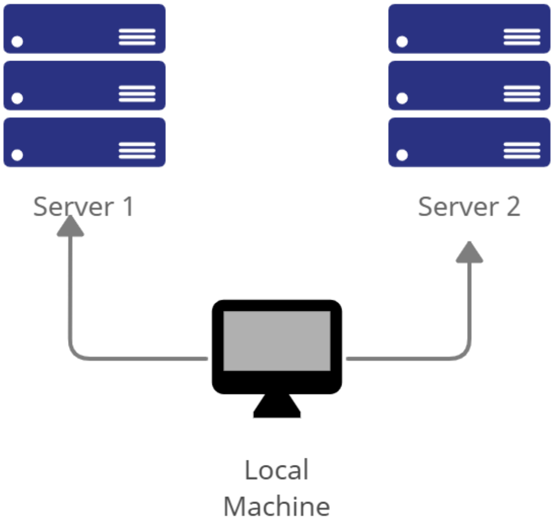
1.1. Create and configure Virtual Machines in Microsoft Azure or VirtualBox

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

2. **Discussion:**

Network Topology:

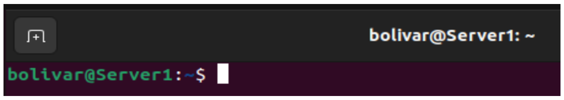
Assume that you have created the following network topology in Virtual Machines, *provide screenshots for each task*. (Note: *it is assumed that you have the prior knowledge of cloning and creating snapshots in a virtual machine*).



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.

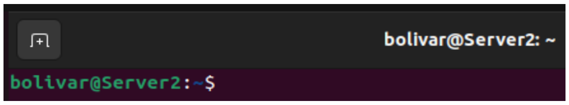
1. Change the hostname using the command *sudo nano /etc/hostname*

    1.1 Use server1 for Server 1



```
bolivar@Server1: ~
bolivar@Server1:~$ 
```

    1.2 Use server2 for Server 2



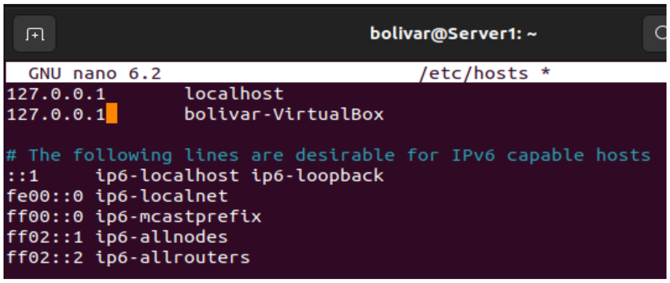
```
bolivar@Server2: ~
bolivar@Server2:~$
```

Observation: I simply use the server1 for Server1 using the command (sudo nano /etc/hostname).

1.3 Use workstation for the Local Machine

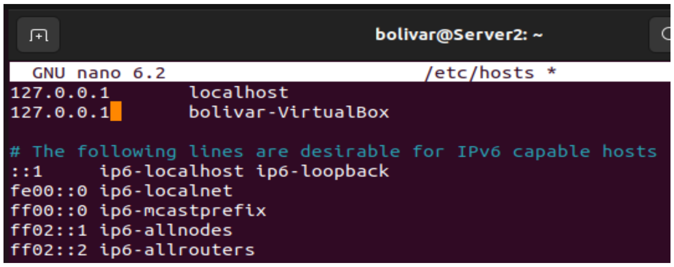
2. Edit the hosts using the command *sudo nano /etc/hosts.* Edit the second line.

2.1 Type 127.0.0.1 server 1 for Server 1



```
  GNU nano 6.2                                /etc/hosts *
127.0.0.1       localhost
127.0.0.1       bolivar-VirtualBox

# 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 6.2                                /etc/hosts *
127.0.0.1       localhost
127.0.0.1       bolivar-VirtualBox

# 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
```

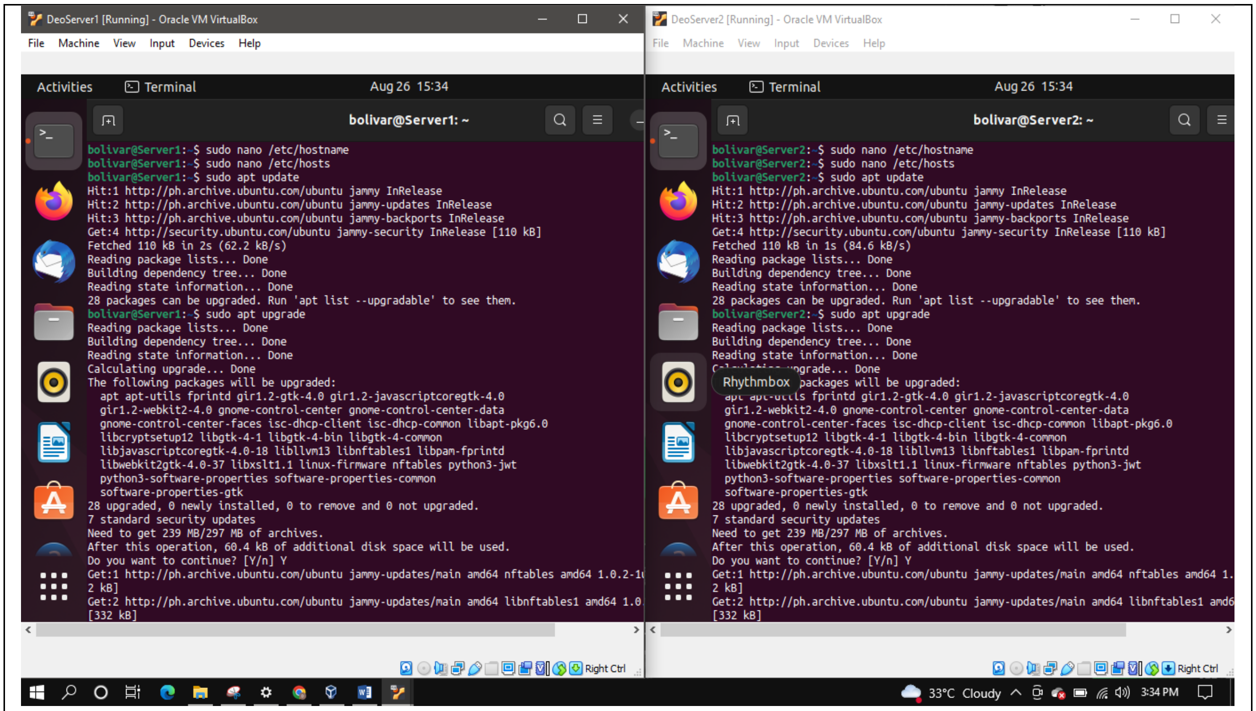
Observation: I simply edit the hosts number by typing the 127.0.0.1 for both servers.

2.3 Type 127.0.0.1 workstation for the Local Machine

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.

   Server 1 and Server 2: note: The output picture will be the same on both server.

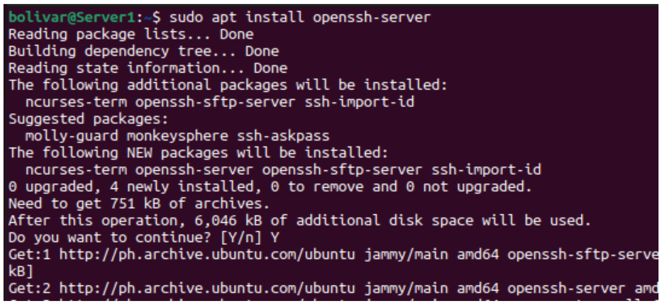

DeoServer1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Aug 26 15:34
bolivar@Server1: ~
DeoServer2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Aug 26 15:34
bolivar@Server2: ~
Rhythmbox
Right Ctrl
33°C Cloudy
3:34 PM

```
bolivar@Server1:~$ sudo nano /etc/hostname
bolivar@Server1:~$ sudo nano /etc/hosts
bolivar@Server1:~$ sudo apt update
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Fetched 110 kB in 2s (62.2 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
28 packages can be upgraded. Run 'apt list --upgradable' to see them.
bolivar@Server1:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  apt apt-utils fprintd gir1.2-gtk-4.0 gir1.2-javascriptcoregtk-4.0
  gir1.2-webkit2-4.0 gnome-control-center gnome-control-center-data
  gnome-control-center-faces isc-dhcp-client isc-dhcp-common libapt-pkg6.0
  libcryptsetup12 libgtk-4-1 libgtk-4-bin libgtk-4-common
  libjavascriptcoregtk-4.0-18 libllvm13 libnftables1 libpam-fprintd
  libwebkit2gtk-4.0-37 libxslt1.1 linux-firmware nftables python3-jwt
  python3-software-properties software-properties-common
  software-properties-gtk
28 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
7 standard security updates
Need to get 239 MB/297 MB of archives.
After this operation, 60.4 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 nftables amd64 1.0.2-1u
2 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libnftables1 amd64 1.0.
[332 kB]
```

```
bolivar@Server2:~$ sudo nano /etc/hostname
bolivar@Server2:~$ sudo nano /etc/hosts
bolivar@Server2:~$ sudo apt update
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Fetched 110 kB in 1s (84.6 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
28 packages can be upgraded. Run 'apt list --upgradable' to see them.
bolivar@Server2:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  apt apt-utils fprintd gir1.2-gtk-4.0 gir1.2-javascriptcoregtk-4.0
  gir1.2-webkit2-4.0 gnome-control-center gnome-control-center-data
  gnome-control-center-faces isc-dhcp-client isc-dhcp-common libapt-pkg6.0
  libcryptsetup12 libgtk-4-1 libgtk-4-bin libgtk-4-common
  libjavascriptcoregtk-4.0-18 libllvm13 libnftables1 libpam-fprintd
  libwebkit2gtk-4.0-37 libxslt1.1 linux-firmware nftables python3-jwt
  python3-software-properties software-properties-common
  software-properties-gtk
28 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
7 standard security updates
Need to get 239 MB/297 MB of archives.
After this operation, 60.4 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 nftables amd64 1.
2 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libnftables1 amd6
[332 kB]
```

Observation: In this output above it shows that I update and upgrade the packages of two servers.

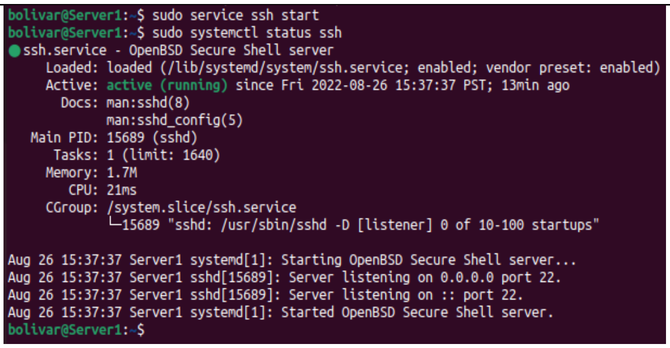
2. Install the SSH server using the command *sudo apt install openssh-server*.
   Server 1 and Server 2: note: The output picture will be the same on both server.



```
bolivar@Server1:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere 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 751 kB of archives.
After this operation, 6,046 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 openssh-sftp-serve
kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 openssh-server amd
```

Observation: In this output above it shows that I installed the SSH server for both servers.

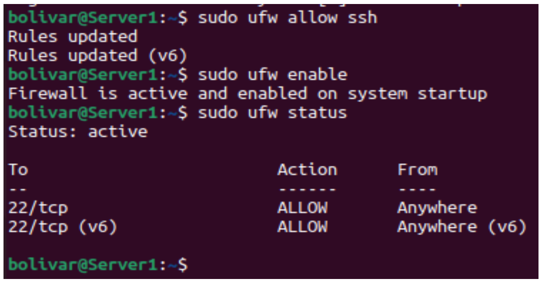
3. Verify if the SSH service has started by issuing the following commands:
   3.1 *sudo service ssh start*
   3.2 *sudo systemctl status ssh*
   Server 1 and Server 2:



```
bolivar@Server1:~$ sudo service ssh start
bolivar@Server1:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2022-08-26 15:37:37 PST; 13min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 15689 (sshd)
      Tasks: 1 (limit: 1640)
     Memory: 1.7M
        CPU: 21ms
     CGroup: /system.slice/ssh.service
             └─15689 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 26 15:37:37 Server1 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 26 15:37:37 Server1 sshd[15689]: Server listening on 0.0.0.0 port 22.
Aug 26 15:37:37 Server1 sshd[15689]: Server listening on :: port 22.
Aug 26 15:37:37 Server1 systemd[1]: Started OpenBSD Secure Shell server.
bolivar@Server1:~$
```

Observation: In this output above it shows that I'm accessing and verifying the SSH service in both servers.

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*
   Server 1 and Server 2:

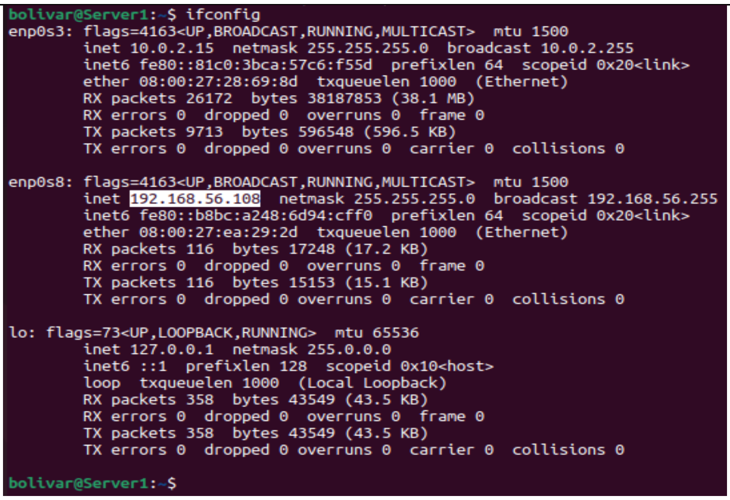


```
bolivar@Server1:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
bolivar@Server1:~$ sudo ufw enable
Firewall is active and enabled on system startup
bolivar@Server1:~$ sudo ufw status
Status: active

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

bolivar@Server1:~$
```

Observation: In this output above it shows that I enable the Firewall on port 22 on both servers.

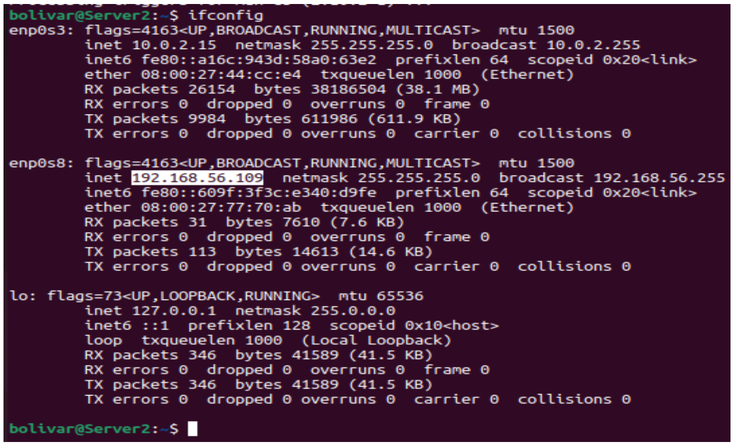
Task 3: Verify network settings on Server 1, Server 2, and Local Machine. On each device, do the following:

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: 192.168.56.108



```
bolivar@Server1:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::81c0:3bca:57c6:f55d  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:28:69:8d  txqueuelen 1000  (Ethernet)
        RX packets 26172  bytes 38187853 (38.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9713  bytes 596548 (596.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.108  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::b8bc:a248:6d94:cff0  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:ea:29:2d  txqueuelen 1000  (Ethernet)
        RX packets 116  bytes 17248 (17.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 116  bytes 15153 (15.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 358  bytes 43549 (43.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 358  bytes 43549 (43.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

bolivar@Server1:~$
```

1.2 Server 2 IP address: 192.168.56.109

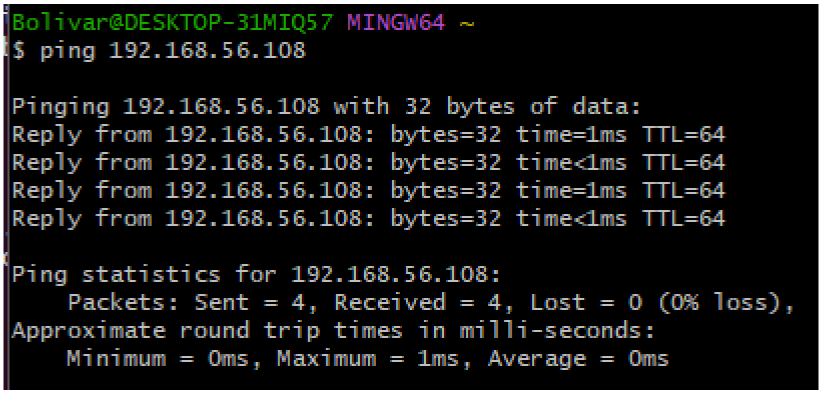


```
bolivar@Server2:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a16c:943d:58a0:63e2  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:44:cc:e4  txqueuelen 1000  (Ethernet)
        RX packets 26154  bytes 38186504 (38.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9984  bytes 611986 (611.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.109  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::609f:3f3c:e340:d9fe  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:77:70:ab  txqueuelen 1000  (Ethernet)
        RX packets 31  bytes 7610 (7.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 113  bytes 14613 (14.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 346  bytes 41589 (41.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 346  bytes 41589 (41.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

bolivar@Server2:~$
```

1.3 Server 3 IP address: 192.168.56.___

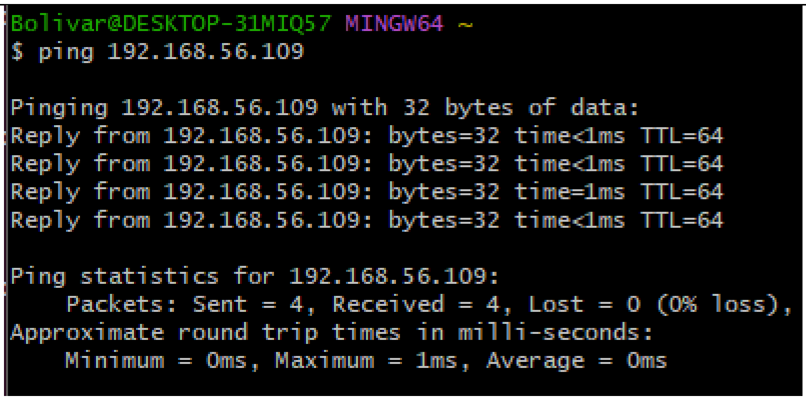
2. Make sure that they can ping each other.

2.1 Connectivity test for Local Machine 1 to Server 1: ☒ Successful ☐ Not Successful



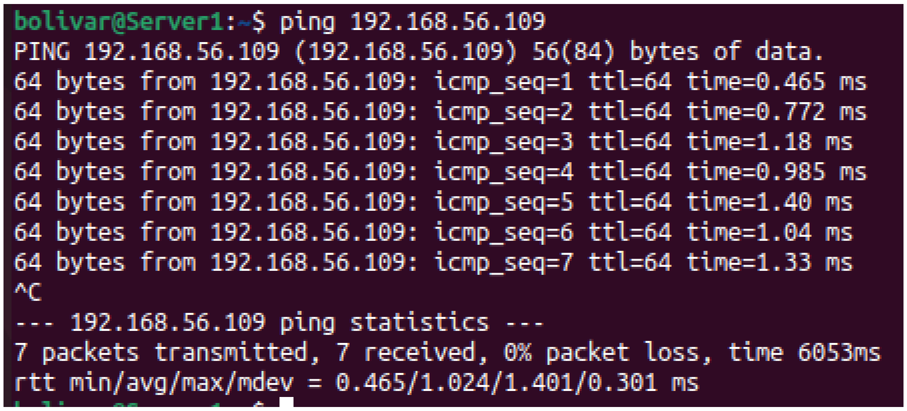
```
Bolivar@DESKTOP-31MIQ57 MINGW64 ~
$ ping 192.168.56.108

Pinging 192.168.56.108 with 32 bytes of data:
Reply from 192.168.56.108: bytes=32 time=1ms TTL=64
Reply from 192.168.56.108: bytes=32 time<1ms TTL=64
Reply from 192.168.56.108: bytes=32 time=1ms TTL=64
Reply from 192.168.56.108: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.56.108:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

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



```
Bolivar@DESKTOP-31MIQ57 MINGW64 ~
$ ping 192.168.56.109

Pinging 192.168.56.109 with 32 bytes of data:
Reply from 192.168.56.109: bytes=32 time<1ms TTL=64
Reply from 192.168.56.109: bytes=32 time<1ms TTL=64
Reply from 192.168.56.109: bytes=32 time=1ms TTL=64
Reply from 192.168.56.109: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.56.109:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

2.3 Connectivity test for Server 1 to Server 2: ☒ Successful ☐ Not Successful

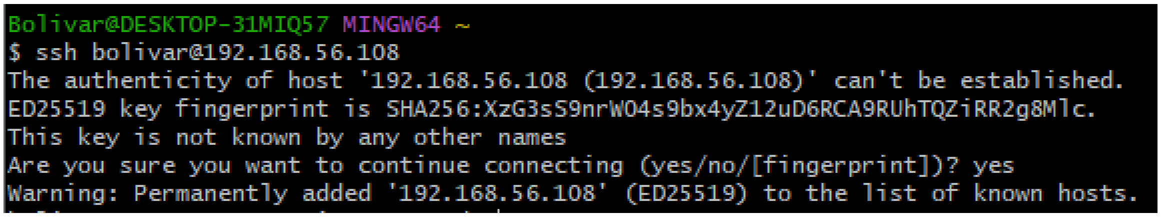


```
bolivar@Server1:~$ ping 192.168.56.109
PING 192.168.56.109 (192.168.56.109) 56(84) bytes of data.
64 bytes from 192.168.56.109: icmp_seq=1 ttl=64 time=0.465 ms
64 bytes from 192.168.56.109: icmp_seq=2 ttl=64 time=0.772 ms
64 bytes from 192.168.56.109: icmp_seq=3 ttl=64 time=1.18 ms
64 bytes from 192.168.56.109: icmp_seq=4 ttl=64 time=0.985 ms
64 bytes from 192.168.56.109: icmp_seq=5 ttl=64 time=1.40 ms
64 bytes from 192.168.56.109: icmp_seq=6 ttl=64 time=1.04 ms
64 bytes from 192.168.56.109: icmp_seq=7 ttl=64 time=1.33 ms
^C
--- 192.168.56.109 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6053ms
rtt min/avg/max/mdev = 0.465/1.024/1.401/0.301 ms
```

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

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

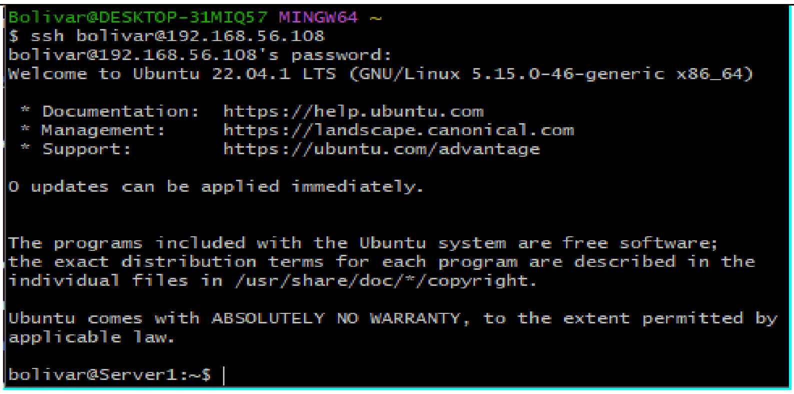
1.1 ssh username@ip_address_server1 for example, *ssh jvtaylar@192.168.56.120*



```
Bolivar@DESKTOP-31MIQ57 MINGW64 ~
$ ssh bolivar@192.168.56.108
The authenticity of host '192.168.56.108 (192.168.56.108)' can't be established.
ED25519 key fingerprint is SHA256:XzG3sS9nrWO4s9bx4yZ12uD6RCA9RUhTQZiRR2g8Mlc.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.108' (ED25519) to the list of known hosts.
```

Observation: In this output above it shows that I'm accessing the SSH server of server1 through the local machine.

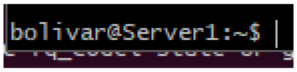
1.2 Enter the password for server 1 when prompted



```
Bolivar@DESKTOP-31MIQ57 MINGW64 ~
$ ssh bolivar@192.168.56.108
bolivar@192.168.56.108's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-46-generic x86_64)

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

0 updates can be applied immediately.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

bolivar@Server1:~$ |
```

Observation: I successfully enter the SSH server of server 1 by entering the password on the local machine.

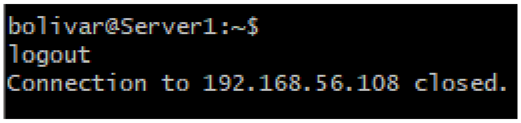
1.3 Verify that you are in server 1. The user should be in this format user@server1. For example, *jvtaylar@server1*



```
bolivar@Server1:~$ |
```

Observation: This is the verification that my local machine is connected to server1.

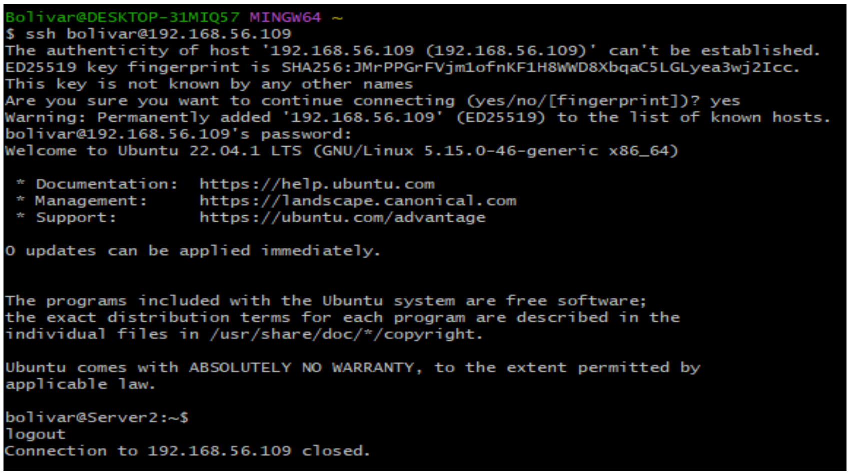
2. Logout of Server 1 by issuing the command *control + D.*



```
bolivar@Server1:~$
logout
Connection to 192.168.56.108 closed.
```

Observation: Logging out the server1 in the local machine.

3. Do the same for Server 2.

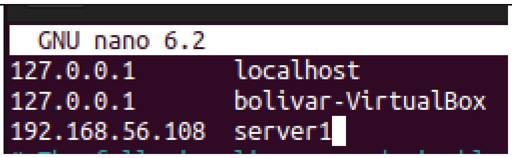


```
Bolivar@DESKTOP-31MIQ57 MINGW64 ~
$ ssh bolivar@192.168.56.109
The authenticity of host '192.168.56.109 (192.168.56.109)' can't be established.
ED25519 key fingerprint is SHA256:JMrPPGrFVjm1ofnKF1H8WWD8XbqaC5LGLyea3wj2Icc.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.109' (ED25519) to the list of known hosts.
bolivar@192.168.56.109's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-46-generic x86_64)

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

0 updates can be applied immediately.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

bolivar@Server2:~$
logout
Connection to 192.168.56.109 closed.
```

Observation: This is all the same process on the task 1.3.

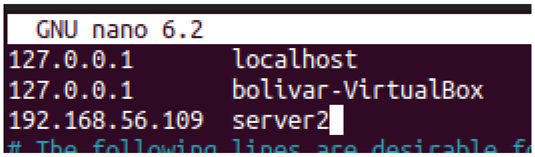
4. Edit the hosts of the Local Machine by issuing the command *sudo nano /etc/hosts.* Below all texts type the following:

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



```
 GNU nano 6.2
127.0.0.1       localhost
127.0.0.1       bolivar-VirtualBox
192.168.56.108  server1
```

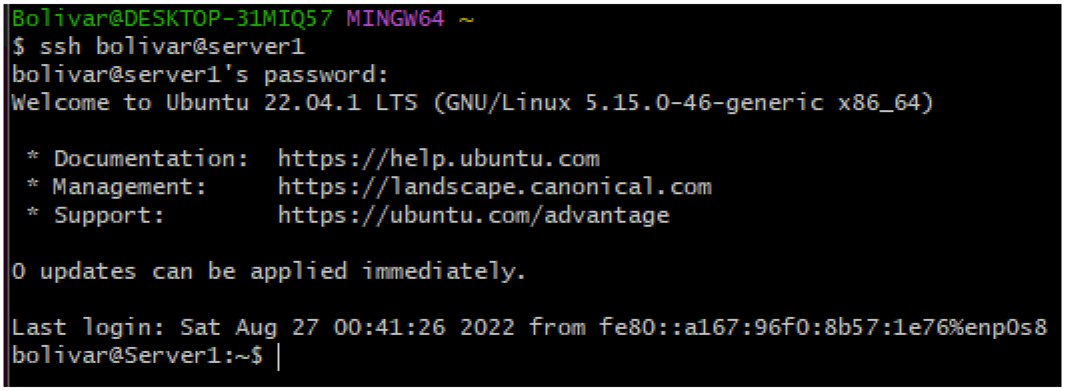
4.2 IP_address server 2 (provide the ip address of server 2 followed by the hostname)



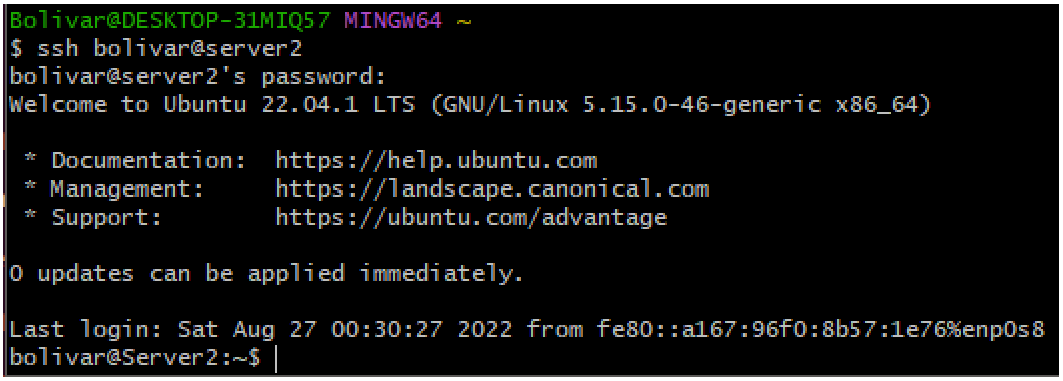
```
 GNU nano 6.2
127.0.0.1       localhost
127.0.0.1       bolivar-VirtualBox
192.168.56.109  server2
# The following lines are desirable f
```

4.3 Save the file and exit.

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.



```
Bolivar@DESKTOP-31MIQ57 MINGW64 ~
$ ssh bolivar@server1
bolivar@server1's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-46-generic x86_64)

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

0 updates can be applied immediately.

Last login: Sat Aug 27 00:41:26 2022 from fe80::a167:96f0:8b57:1e76%enp0s8
bolivar@Server1:~$ 
```



```
Bolivar@DESKTOP-31MIQ57 MINGW64 ~
$ ssh bolivar@server2
bolivar@server2's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-46-generic x86_64)

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

0 updates can be applied immediately.

Last login: Sat Aug 27 00:30:27 2022 from fe80::a167:96f0:8b57:1e76%enp0s8
bolivar@Server2:~$ 
```

Observation: In this output above it shows that I successfully access and verify that my local machine is connected to server1 and server2.

Reflections:

Answer the following:

1. How are we able to use the hostname instead of IP address in SSH commands?

   I am able to use the hostname in SSH commands instead of using IP address because of the addition of hostname associated to the IP address of a certain server in /etc/hosts. As a result, we could issue SSH instructions to a server and access it by just specifying its hostname because it had been added to a registry or file that contained IP addresses for servers and other units.

2. How secured is SSH?

   SSH, or Safe Shell, employs encryption and authentication for all possible connections, making it extremely secure. SSH's high level of security is also a result of its remote client management system. It simply tells us that accessing SSH connections between devices is difficult when secure or SSH keys/credentials are used for SSH connections. However, if SSH is not properly maintained, it won't be safe at the time, leading to the normal SSH brute force assaults.

Conclusion:

After I finished this activity, first is I learned how to clone the Virtual Machines for the different servers that I will use in this activity. I also learned how to use the SHH commands on the local machine and I'm able to observed the connection between my two servers through my local machine. Lastly I'm able to perform this activity while having so many complication why I'm doing the cloning.

"I affirm that I will not give or receive any unauthorized help on this activity/examand that all workwill be my own."

Bolivar, Deo