| | |
|---|---|
| **Name: Earl Kristian G. Villamor** | **Date Performed: August 16, 2022** |
| **Course/Section: CPE31S22** | **Date Submitted: August 23, 2022** |
| **Instructor: Dr. Jonathan Taylar** | **Semester and SY: 1st Sem – 2022 - 2023** |

## Activity 1: Configure Network using Virtual Machines

### 1. Objectives:

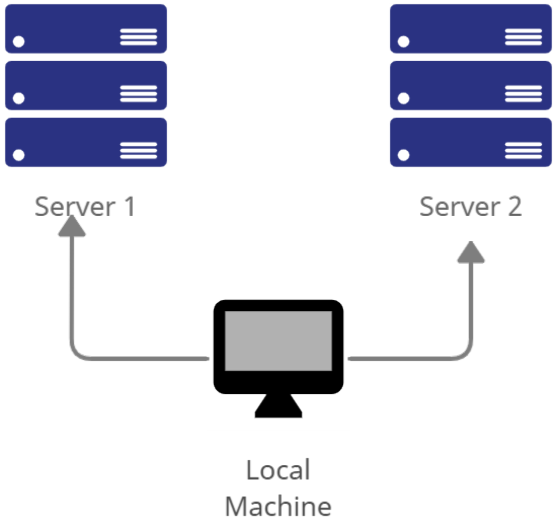
1.1. Create and configure Virtual Machines in Microsoft Azure or VirtualBox

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

### 2. Discussion:

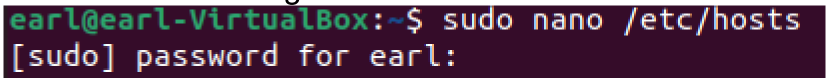
**Network Topology:**

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*).



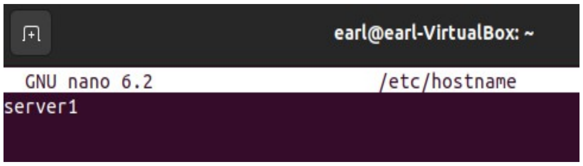
**Task 1**: Do the following on Server 1, Server 2, and Local Machine. In editing the file using nano command, press control + O to write out (save the file). Press enter when asked for the name of the file. Press control + X to end.

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



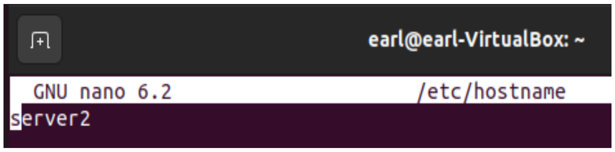
```
earl@earl-VirtualBox:~$ sudo nano /etc/hosts
[sudo] password for earl:
```

1.1 Use server1 for Server 1



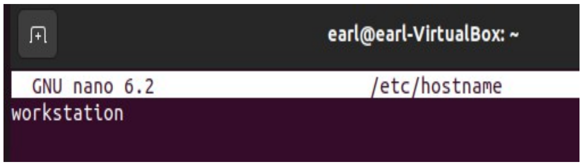
```
  GNU nano 6.2                    /etc/hostname
server1
```

1.2 Use server2 for Server 2



```
  GNU nano 6.2                    /etc/hostname
server2
```

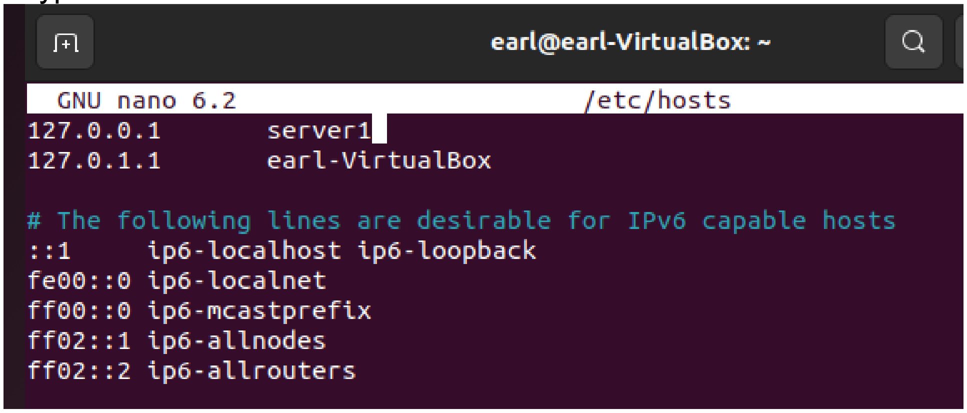
1.3 Use workstation for the Local Machine



```
  GNU nano 6.2                    /etc/hostname
workstation
```

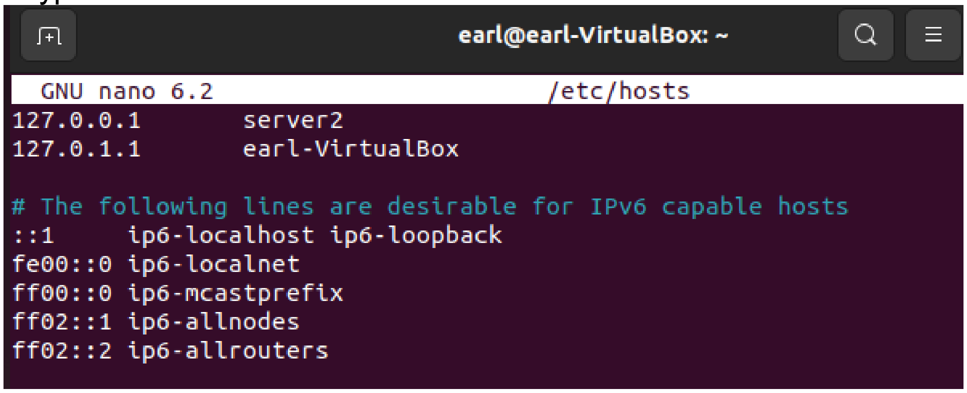
2. Edit the hosts using the command *sudo nano /etc/hosts*. Edit the second line.

2.1 Type 127.0.0.1 server 1 for Server 1



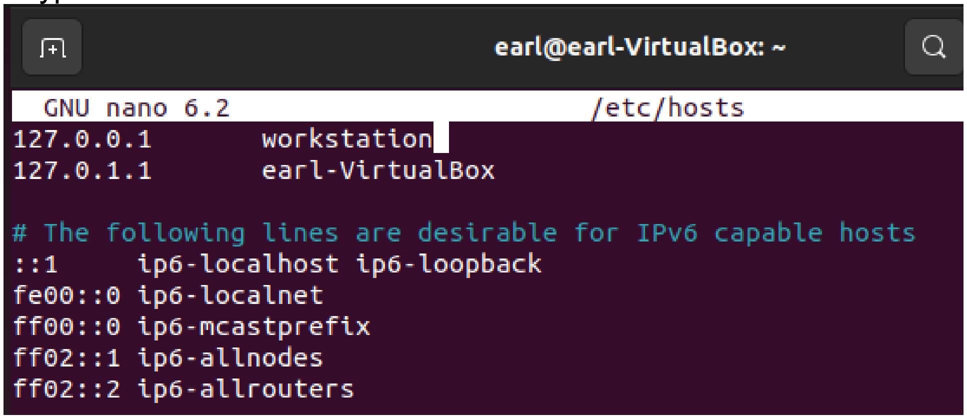
```
  GNU nano 6.2                                /etc/hosts
127.0.0.1       server1
127.0.1.1       earl-VirtualBox

# 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       server2
127.0.1.1       earl-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
```

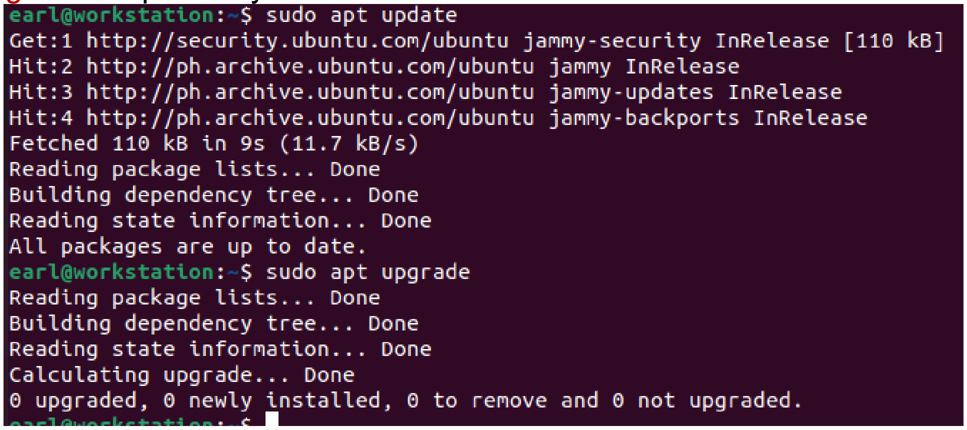
2.3 Type 127.0.0.1 workstation for the Local Machine



```
  GNU nano 6.2                          /etc/hosts
127.0.0.1       workstation
127.0.1.1       earl-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
```

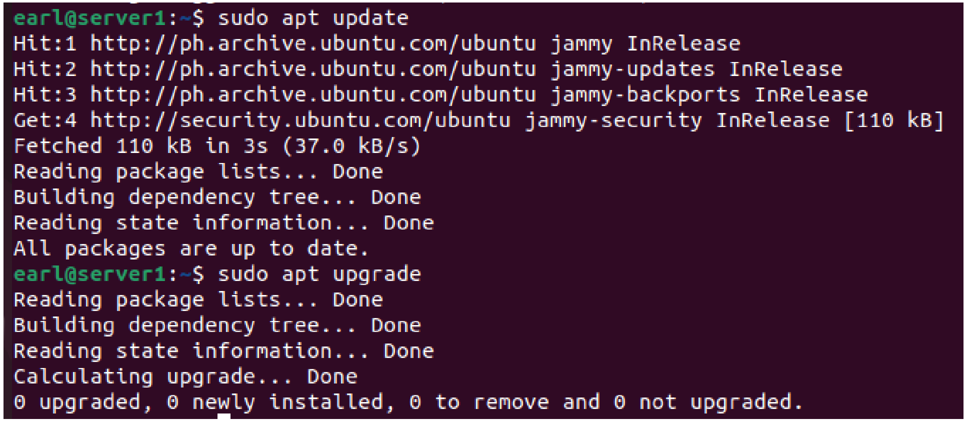
**Task 2**: Configure SSH on Server 1, Server 2, and Local Machine. Do the following:

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



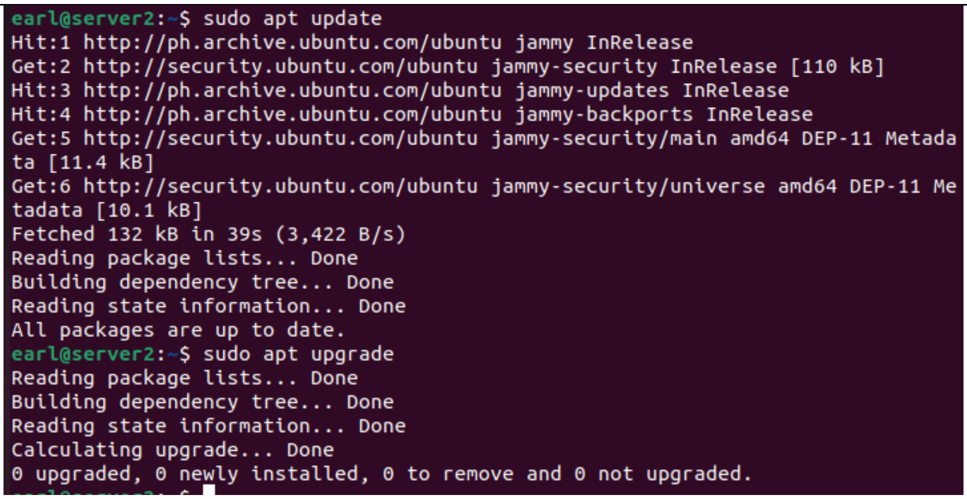
```
earl@workstation:~$ sudo apt update
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:2 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Fetched 110 kB in 9s (11.7 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
earl@workstation:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
earl@workstation:~$
```

Figure 2.1



```
earl@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 3s (37.0 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
earl@server1:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

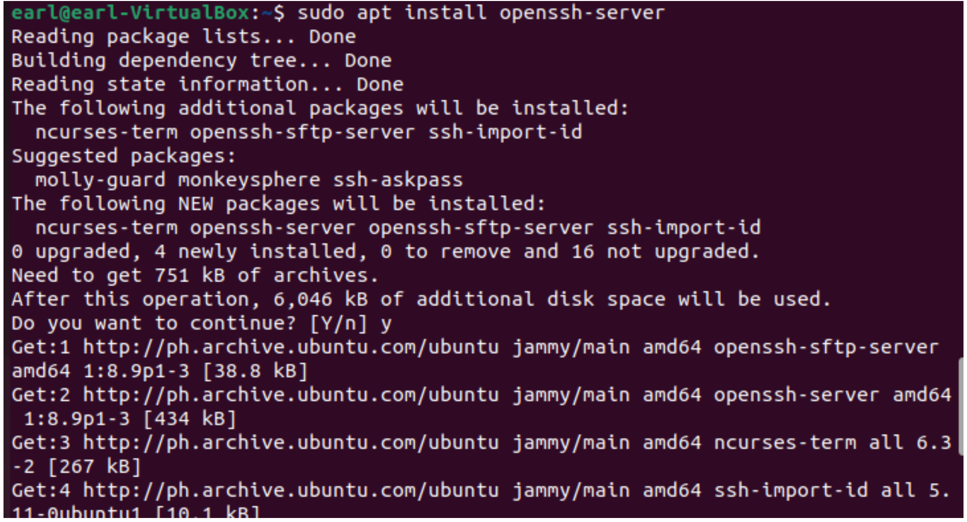
Figure 2.2



```
earl@server2:~$ sudo apt update
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metada
ta [11.4 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Me
tadata [10.1 kB]
Fetched 132 kB in 39s (3,422 B/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
earl@server2:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

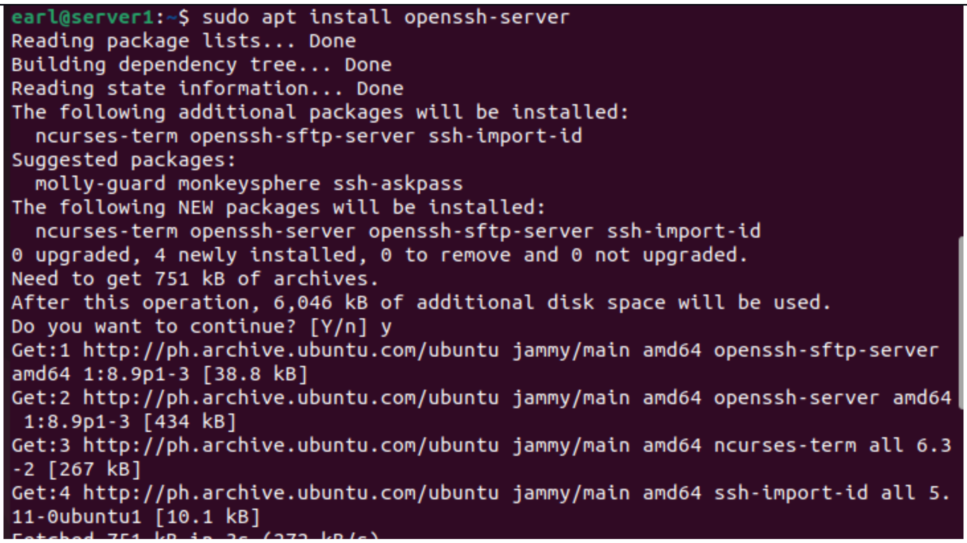
Figure 2.3

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



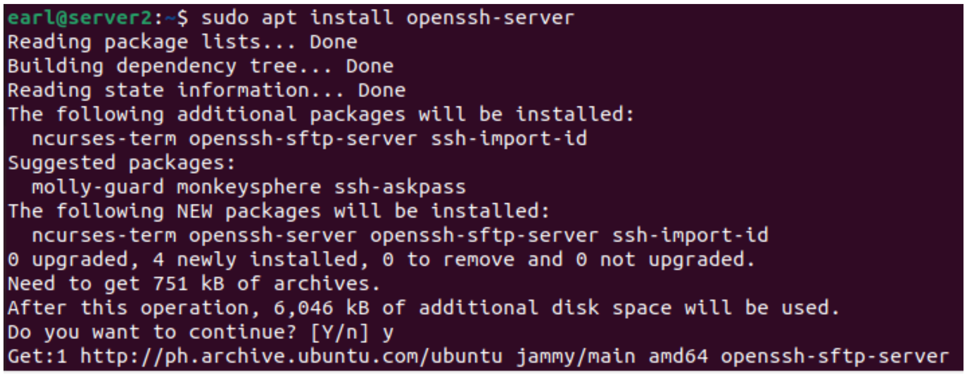
```
earl@earl-VirtualBox:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere 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 16 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-server
amd64 1:8.9p1-3 [38.8 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 openssh-server amd64
 1:8.9p1-3 [434 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 ncurses-term all 6.3
-2 [267 kB]
Get:4 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 ssh-import-id all 5.
11-0ubuntu1 [10.1 kB]
```

Figure 2.4



```
earl@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-server
amd64 1:8.9p1-3 [38.8 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 openssh-server amd64
 1:8.9p1-3 [434 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 ncurses-term all 6.3
-2 [267 kB]
Get:4 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 ssh-import-id all 5.
11-0ubuntu1 [10.1 kB]
Fetched 751 kB in 3s (273 kB/s)
```

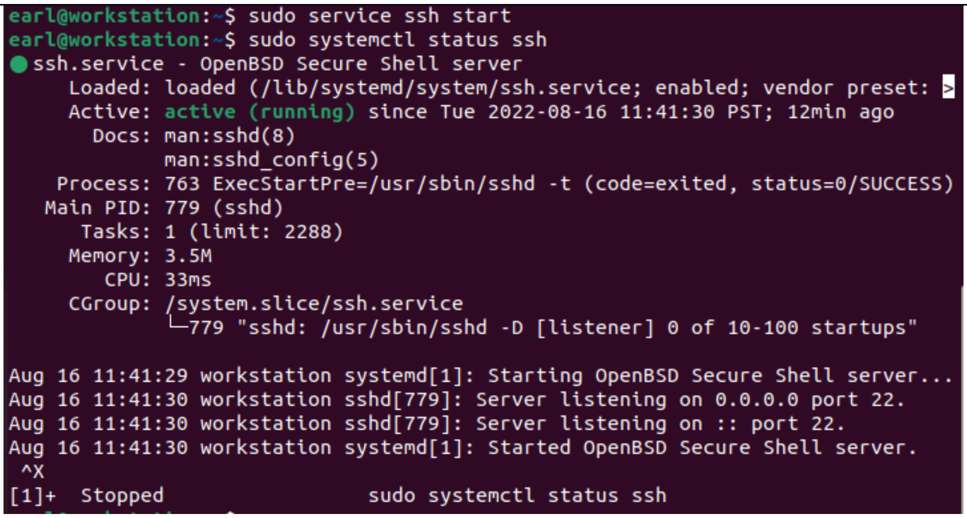
Figure 2.5



```
earl@server2:~$ 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-server
```

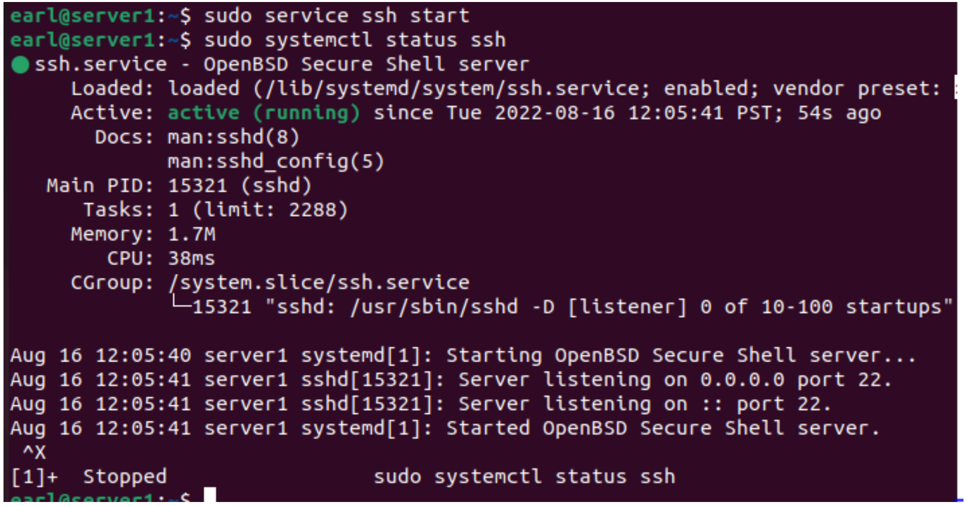
Figure 2.6

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*



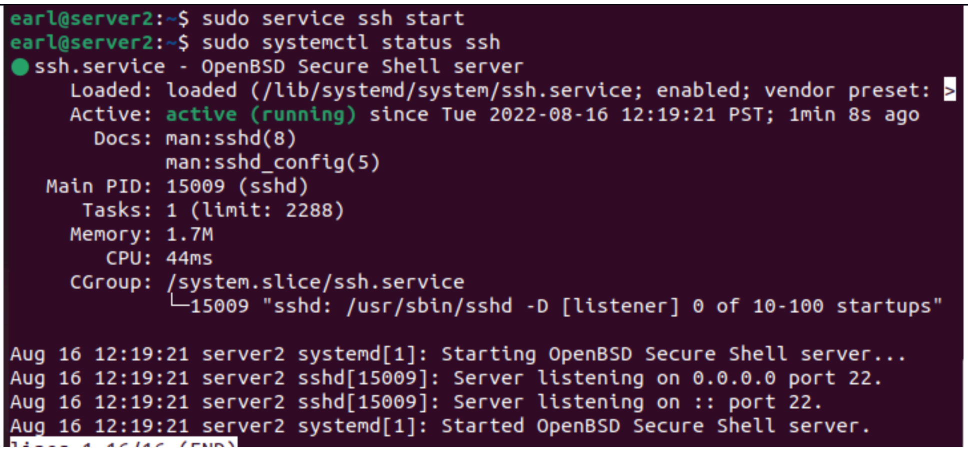
```
earl@workstation:~$ sudo service ssh start
earl@workstation:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: >
     Active: active (running) since Tue 2022-08-16 11:41:30 PST; 12min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 763 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 779 (sshd)
      Tasks: 1 (limit: 2288)
     Memory: 3.5M
        CPU: 33ms
     CGroup: /system.slice/ssh.service
             └─779 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 16 11:41:29 workstation systemd[1]: Starting OpenBSD Secure Shell server...
Aug 16 11:41:30 workstation sshd[779]: Server listening on 0.0.0.0 port 22.
Aug 16 11:41:30 workstation sshd[779]: Server listening on :: port 22.
Aug 16 11:41:30 workstation systemd[1]: Started OpenBSD Secure Shell server.
 ^X
[1]+  Stopped                 sudo systemctl status ssh
```

Figure 2.7



```
earl@server1:~$ sudo service ssh start
earl@server1:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset:
     Active: active (running) since Tue 2022-08-16 12:05:41 PST; 54s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 15321 (sshd)
      Tasks: 1 (limit: 2288)
     Memory: 1.7M
        CPU: 38ms
     CGroup: /system.slice/ssh.service
             └─15321 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 16 12:05:40 server1 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 16 12:05:41 server1 sshd[15321]: Server listening on 0.0.0.0 port 22.
Aug 16 12:05:41 server1 sshd[15321]: Server listening on :: port 22.
Aug 16 12:05:41 server1 systemd[1]: Started OpenBSD Secure Shell server.
 ^X
[1]+  Stopped                 sudo systemctl status ssh
```

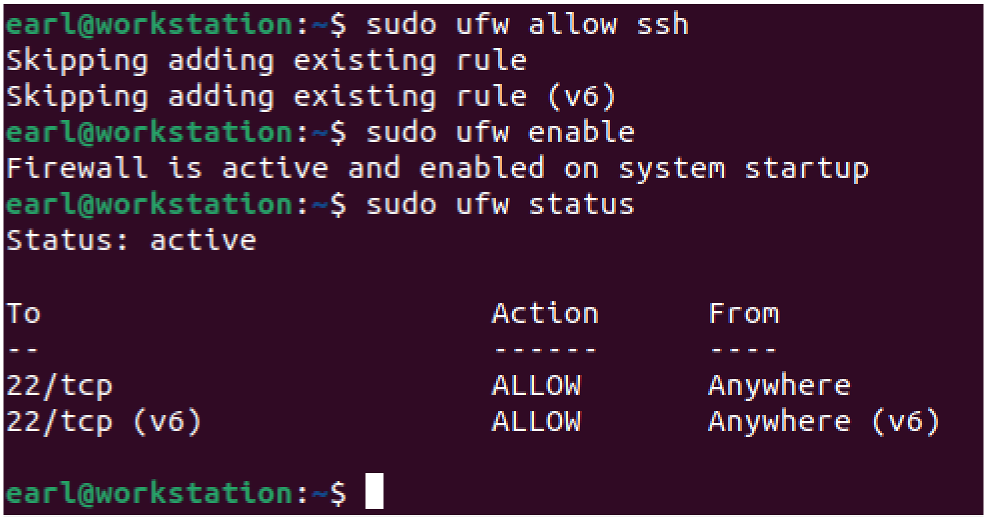
Figure 2.8



```
earl@server2:~$ sudo service ssh start
earl@server2:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: >
     Active: active (running) since Tue 2022-08-16 12:19:21 PST; 1min 8s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 15009 (sshd)
      Tasks: 1 (limit: 2288)
     Memory: 1.7M
        CPU: 44ms
     CGroup: /system.slice/ssh.service
             └─15009 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 16 12:19:21 server2 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 16 12:19:21 server2 sshd[15009]: Server listening on 0.0.0.0 port 22.
Aug 16 12:19:21 server2 sshd[15009]: Server listening on :: port 22.
Aug 16 12:19:21 server2 systemd[1]: Started OpenBSD Secure Shell server.
lines 1-16/16 (END)
```

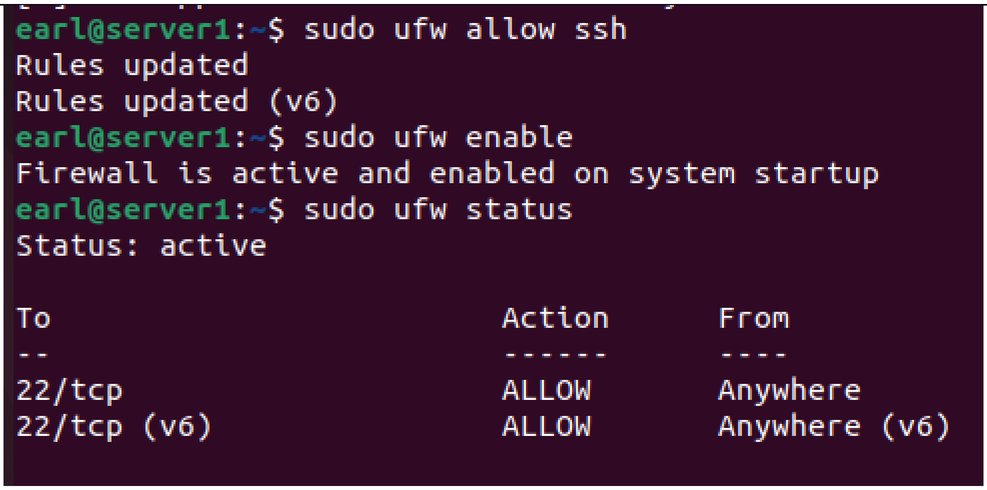
Figure 2.9

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*



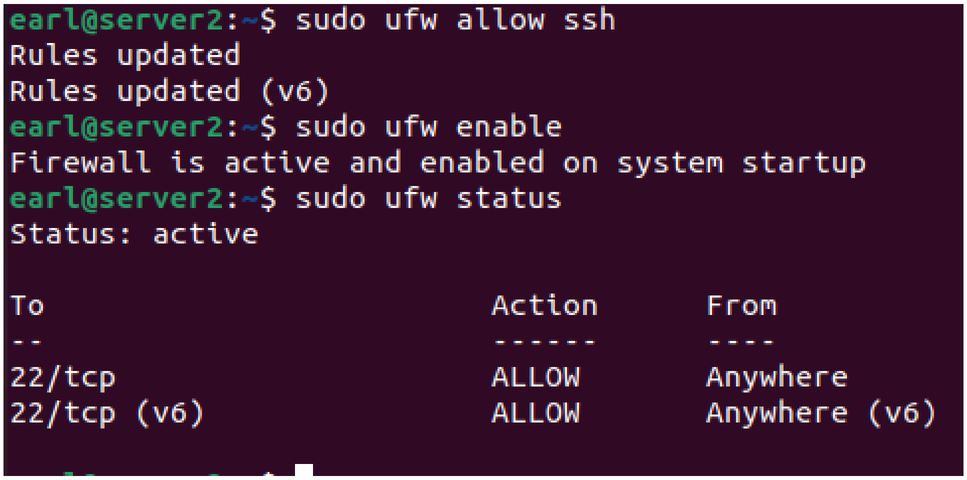
```
earl@workstation:~$ sudo ufw allow ssh
Skipping adding existing rule
Skipping adding existing rule (v6)
earl@workstation:~$ sudo ufw enable
Firewall is active and enabled on system startup
earl@workstation:~$ sudo ufw status
Status: active

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

earl@workstation:~$ 
```

Figure 2.10



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

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

Figure 2.11



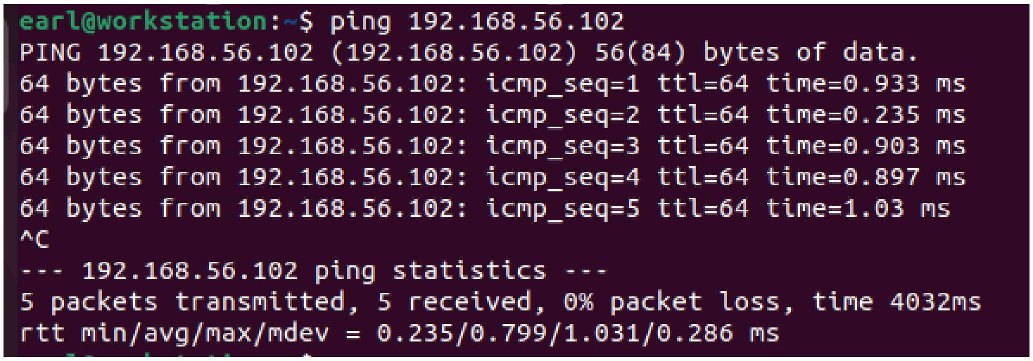
```
earl@server2:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
earl@server2:~$ sudo ufw enable
Firewall is active and enabled on system startup
earl@server2:~$ sudo ufw status
Status: active

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

Figure 2.12

**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: 192.168.56.101
   1.2 Server 2 IP address: 192.168.56.102
   1.3 Server 3 IP address: 192.168.56.103
2. Make sure that they can ping each other.
   2.1 Connectivity test for Local Machine 1 to Server 1:

= Successful

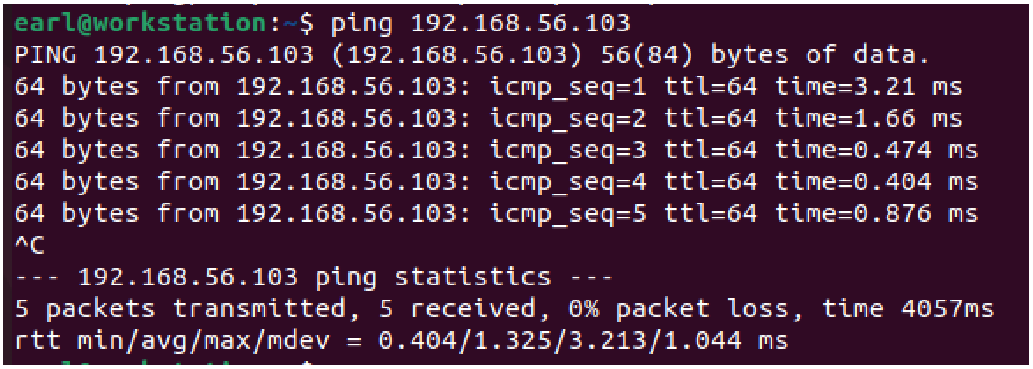


```
earl@workstation:~$ ping 192.168.56.102
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data.
64 bytes from 192.168.56.102: icmp_seq=1 ttl=64 time=0.933 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=0.235 ms
64 bytes from 192.168.56.102: icmp_seq=3 ttl=64 time=0.903 ms
64 bytes from 192.168.56.102: icmp_seq=4 ttl=64 time=0.897 ms
64 bytes from 192.168.56.102: icmp_seq=5 ttl=64 time=1.03 ms
^C
--- 192.168.56.102 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4032ms
rtt min/avg/max/mdev = 0.235/0.799/1.031/0.286 ms
```

Figure 3.1

2.2 Connectivity test for Local Machine 1 to Server 2:

= Successful

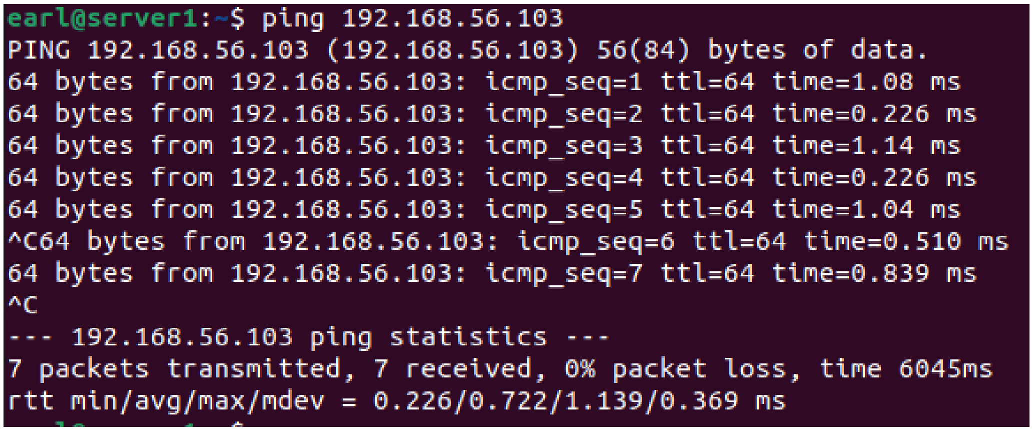


```
earl@workstation:~$ ping 192.168.56.103
PING 192.168.56.103 (192.168.56.103) 56(84) bytes of data.
64 bytes from 192.168.56.103: icmp_seq=1 ttl=64 time=3.21 ms
64 bytes from 192.168.56.103: icmp_seq=2 ttl=64 time=1.66 ms
64 bytes from 192.168.56.103: icmp_seq=3 ttl=64 time=0.474 ms
64 bytes from 192.168.56.103: icmp_seq=4 ttl=64 time=0.404 ms
64 bytes from 192.168.56.103: icmp_seq=5 ttl=64 time=0.876 ms
^C
--- 192.168.56.103 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4057ms
rtt min/avg/max/mdev = 0.404/1.325/3.213/1.044 ms
```

Figure 3.2

2.3 Connectivity test for Server 1 to Server 2:

= Siccessful



```
earl@server1:~$ ping 192.168.56.103
PING 192.168.56.103 (192.168.56.103) 56(84) bytes of data.
64 bytes from 192.168.56.103: icmp_seq=1 ttl=64 time=1.08 ms
64 bytes from 192.168.56.103: icmp_seq=2 ttl=64 time=0.226 ms
64 bytes from 192.168.56.103: icmp_seq=3 ttl=64 time=1.14 ms
64 bytes from 192.168.56.103: icmp_seq=4 ttl=64 time=0.226 ms
64 bytes from 192.168.56.103: icmp_seq=5 ttl=64 time=1.04 ms
^C64 bytes from 192.168.56.103: icmp_seq=6 ttl=64 time=0.510 ms
64 bytes from 192.168.56.103: icmp_seq=7 ttl=64 time=0.839 ms
^C
--- 192.168.56.103 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6045ms
rtt min/avg/max/mdev = 0.226/0.722/1.139/0.369 ms
```

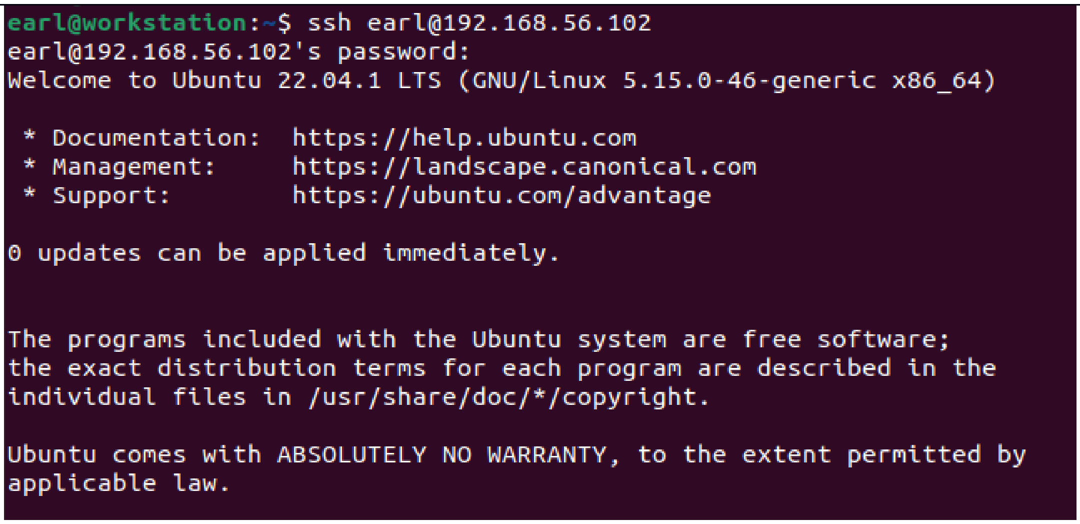
Figure 3.3

**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



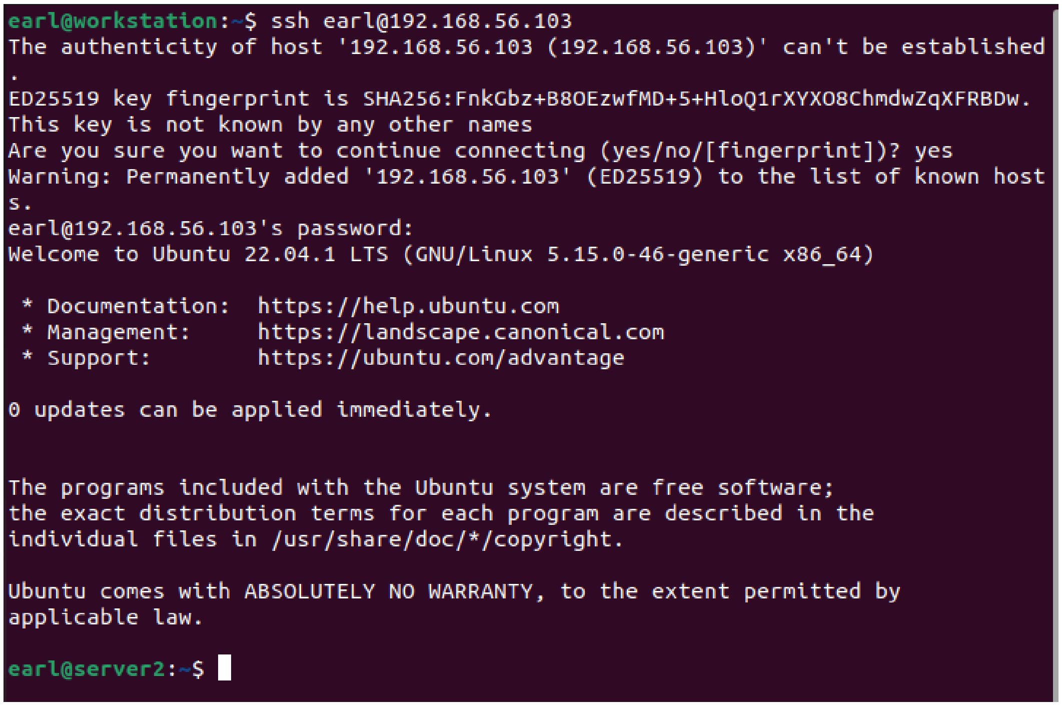
```
earl@workstation:~$ ssh earl@192.168.56.102
earl@192.168.56.102's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-46-generic x86_64)

 * 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.
```

Figure 4.1

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

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

3. Do the same for Server 2.



```
earl@workstation:~$ ssh earl@192.168.56.103
The authenticity of host '192.168.56.103 (192.168.56.103)' can't be established
.
ED25519 key fingerprint is SHA256:FnkGbz+B8OEzwfMD+5+HloQ1rXYXO8ChmdwZqXFRBDw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.103' (ED25519) to the list of known host
s.
earl@192.168.56.103'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.

earl@server2:~$ 
```

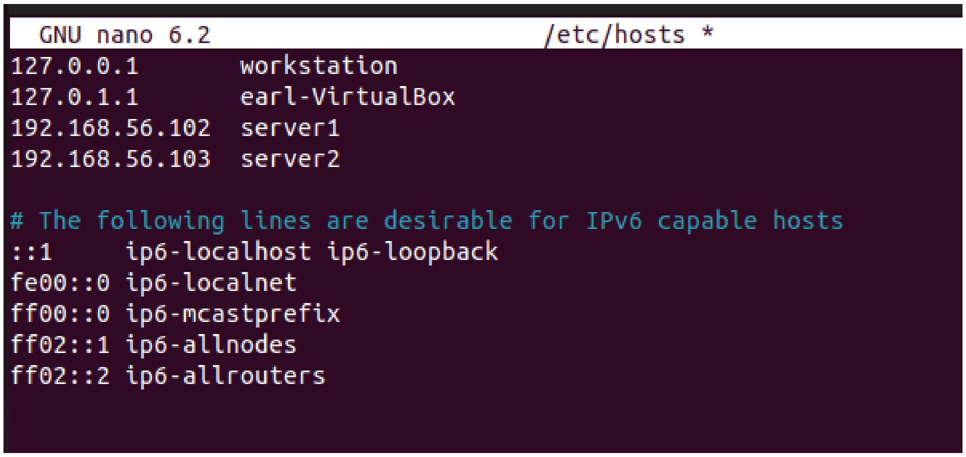
Figure 4.2

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)

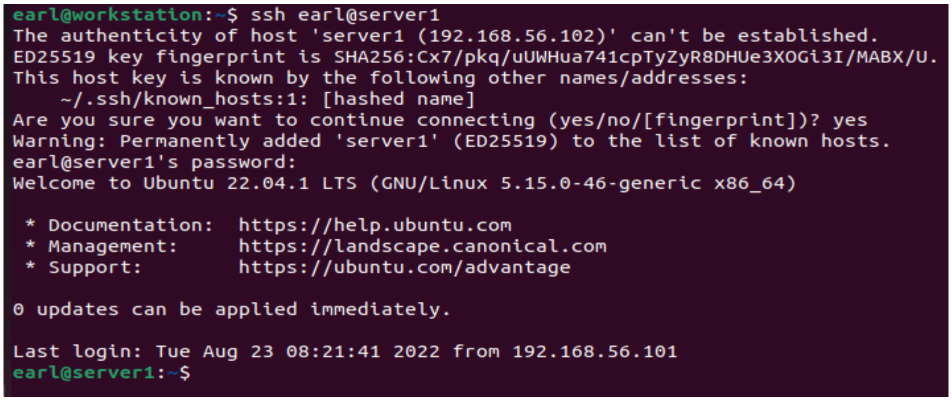
4.3 Save the file and exit.



```
  GNU nano 6.2                            /etc/hosts *
127.0.0.1       workstation
127.0.1.1       earl-VirtualBox
192.168.56.102  server1
192.168.56.103  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
```

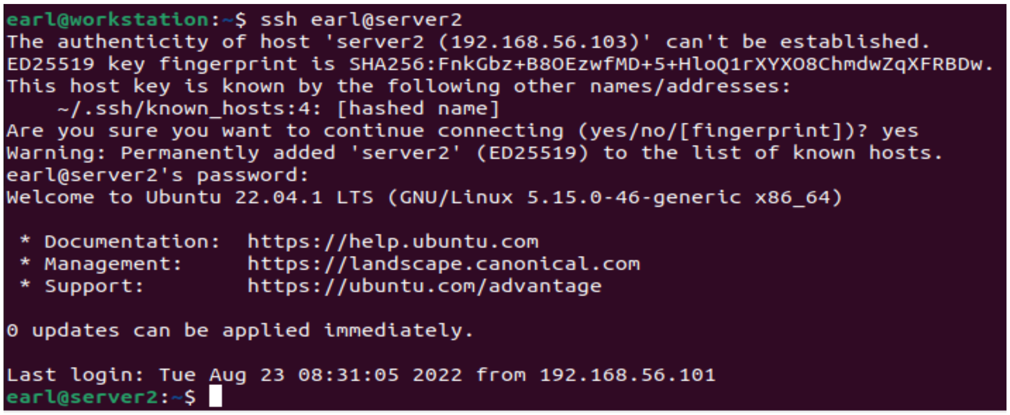
Figure 4.3

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.



```
earl@workstation:~$ ssh earl@server1
The authenticity of host 'server1 (192.168.56.102)' can't be established.
ED25519 key fingerprint is SHA256:Cx7/pkq/uUWHua741cpTyZyR8DHUe3XOGi3I/MABX/U.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'server1' (ED25519) to the list of known hosts.
earl@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: Tue Aug 23 08:21:41 2022 from 192.168.56.101
earl@server1:~$
```

Figure 4.4



```
earl@workstation:~$ ssh earl@server2
The authenticity of host 'server2 (192.168.56.103)' can't be established.
ED25519 key fingerprint is SHA256:FnkGbz+B8OEzwfMD+5+HloQ1rXYXO8ChmdwZqXFRBDw.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:4: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'server2' (ED25519) to the list of known hosts.
earl@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: Tue Aug 23 08:31:05 2022 from 192.168.56.101
earl@server2:~$
```

Figure 4.5

**Reflections:**

Answer the following:

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

- After configuring it on the command *sudo nano /etc/host,* I wrote the IP address of each corresponding servers, after saving and exit the hosts of the local server I'm able to reach out onto the servers just through the hostname instead of IP address.

2. How secured is SSH?

**-** The SSH protocol is regarded as being extremely secure when used with conventional security measures. However, maintaining the security of SSH connections depends in large part on human factors. On SSH servers, brute force attacks are a typical occurrence. Attackers utilize popular identities and passwords to attempt connections to several SSH servers.