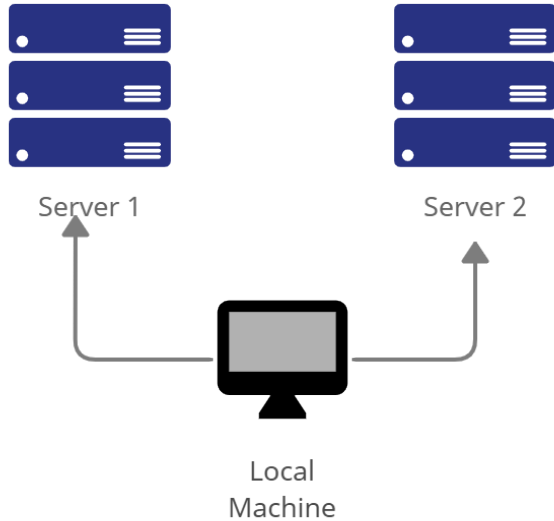


Name: San Jose, Kier Justin F.	Date Performed: 8/30/2024
Course/Section: CPE212 - CPE31S21	Date Submitted: 8/30/2024
Instructor: Engr. Robin Valenzuela	Semester and SY: SY: 2024 - 2025
Activity 1: Configure Network using Virtual Machines	
1. Objectives: 1.1. Create and configure Virtual Machines in Microsoft 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, <i>provide screenshots for each task.</i> (Note: <i>it is assumed that you have the prior knowledge of cloning and creating snapshots in a virtual machine</i>).	
 <pre> graph TD LocalMachine[Local Machine] --- Server1[Server 1] LocalMachine --- Server2[Server 2] subgraph Server1_Stack [Server 1] S1_1[] S1_2[] S1_3[] end subgraph Server2_Stack [Server 2] S2_1[] S2_2[] S2_3[] end </pre>	
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. <ol style="list-style-type: none"> 1. Change the hostname using the command <i>sudo nano /etc/hostname</i> <ol style="list-style-type: none"> 1.1 Use server1 for Server 1 1.2 Use server2 for Server 2 1.3 Use workstation for the Local Machine 	

```
root@SANJOSE:/home/kier# hostnamectl
  Static hostname: workstation
Transient hostname: SANJOSE
    Icon name: computer-vm
    Chassis: vm
  Machine ID: 7b2668e03d54446dadfd490c09ed6fe8
    Boot ID: b1f5970a7cf04c34b883854002704340
  Virtualization: oracle
Operating System: Ubuntu 18.04.2 LTS
    Kernel: Linux 4.18.0-15-generic
  Architecture: x86-64
root@SANJOSE:/home/kier#
```

```
root@SANJOSE:/home/kier# sudo nano /etc/hostname
root@SANJOSE:/home/kier# hostnamectl
  Static hostname: server1
Transient hostname: SANJOSE
    Icon name: computer-vm
    Chassis: vm
  Machine ID: 7b2668e03d54446dadfd490c09ed6fe8
    Boot ID: 96a3bc418b154481a3d5a31416ed72c8
  Virtualization: oracle
Operating System: Ubuntu 18.04.2 LTS
    Kernel: Linux 4.18.0-15-generic
  Architecture: x86-64
root@SANJOSE:/home/kier#
```

```
root@SANJOSE:/home/kier# sudo nano /etc/hostname
root@SANJOSE:/home/kier# hostnamectl
  Static hostname: server2
Transient hostname: SANJOSE
    Icon name: computer-vm
    Chassis: vm
  Machine ID: 7b2668e03d54446dadfd490c09ed6fe8
    Boot ID: 7867dca491e64cc082f06d42c12ed7ed
  Virtualization: oracle
Operating System: Ubuntu 18.04.2 LTS
    Kernel: Linux 4.18.0-15-generic
  Architecture: x86-64
root@SANJOSE:/home/kier#
```

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

```
127.0.0.1      localhost
127.0.0.1      SANJOSE.myguest.virtualbox.org  SANJOSE

# 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

```
127.0.0.1      localhost
127.0.0.1      SANJOSE.myguest.virtualbox.org  SANJOSE

# 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

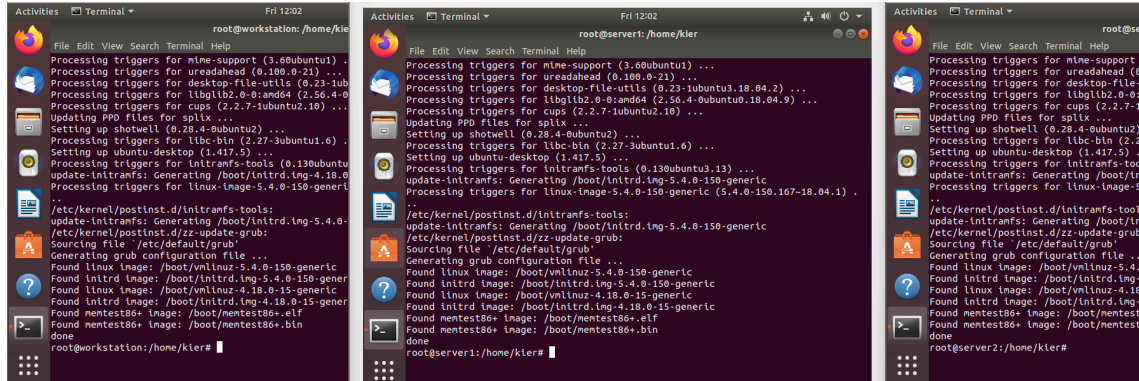
```
GNU nano 2.9.3 /etc/hosts

127.0.0.1      localhost
127.0.0.1      SANJOSE.myguest.virtualbox.org  SANJOSE

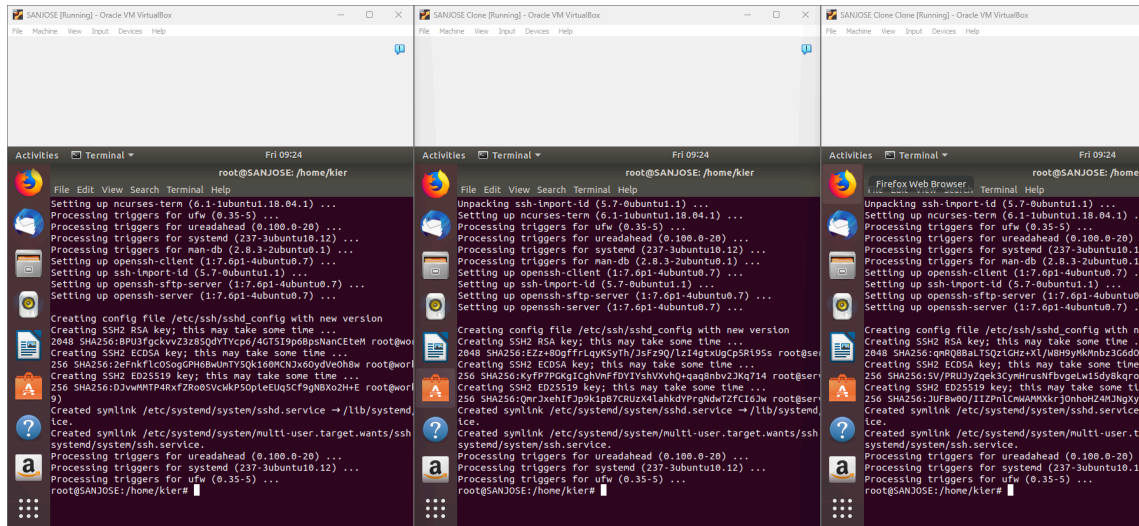
# The following lines are desirable for IPv6 capable hosts
::1          ip6-localhost ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
```

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.



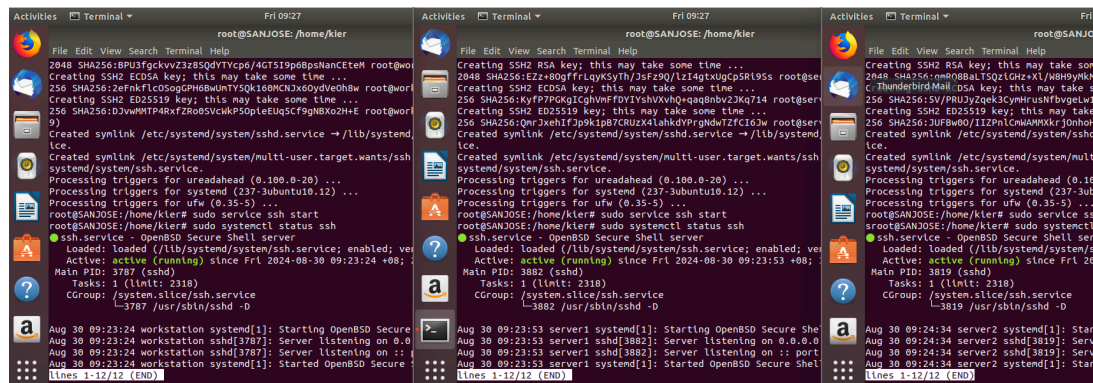
2. Install the SSH server using the command *sudo apt install openssh-server*.



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*

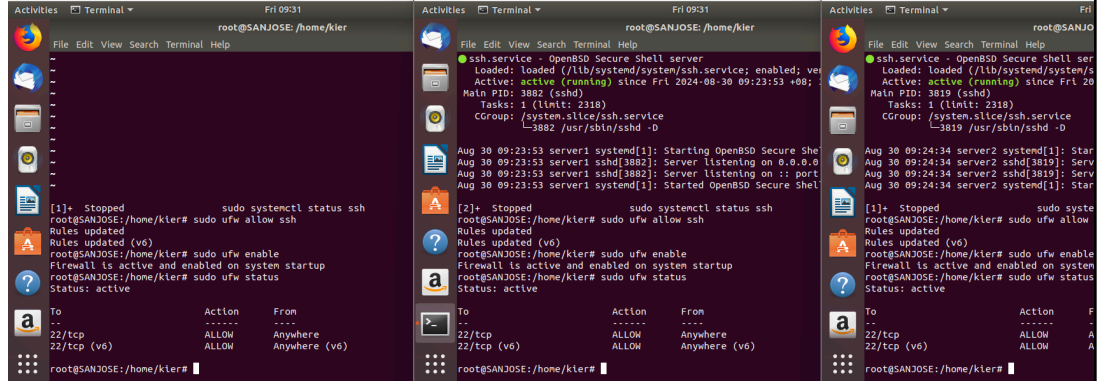


4. Configure the firewall to all port 22 by issuing the following commands:

4.1 *sudo ufw allow ssh*

4.2 *sudo ufw enable*

4.3 *sudo ufw status*



The image shows three terminal windows side-by-side, all with the prompt `root@SANJOSE:/home/kier`. The first terminal shows the command `sudo systemctl status ssh` being run, followed by `sudo ufw allow ssh`, `sudo ufw enable`, and `sudo ufw status`. The second terminal shows the same sequence of commands. The third terminal shows the same sequence of commands. The output of `sudo ufw status` in all three terminals is:

```
Status: active
```

To	Action	From
22/tcp	ALLOW	Anywhere (v6)
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: 192.168.56.138

```
Flags=4103<UP,BROADCAST,RUNNING,MULTICAST>
inet 192.168.56.138 netmask 255.255.255.0 b
inet6 fe80::44f:e849:78dd:5ef5 prefixlen 6
ether 08:00:27:c6:e1:81 txqueuelen 1000 (E
RX packets 519 bytes 88318 (88.3 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 120 bytes 15479 (15.4 KB)
TX errors 0 dropped 0 overruns 0 carrier 0
```

1.2 Server 2 IP address: 192.168.56.139

```
inet 192.168.56.139 netmask 255.255.255.0 b
inet6 fe80::58c3:3df1:2575:8ea9 prefixlen 64
ether 08:00:27:0d:4e:3f txqueuelen 1000 (Et
RX packets 486 bytes 84903 (84.9 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 114 bytes 14751 (14.7 KB)
TX errors 0 dropped 0 overruns 0 carrier 0
```

1.3 Server 3 IP address: 192.168.56.140

```
inet 192.168.56.140 netmask 255.255.255.0
inet6 fe80::1fdf:309e:655c:a56e prefixlen 6
ether 08:00:27:7e:c4:db txqueuelen 1000 (B
RX packets 458 bytes 81197 (81.1 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 111 bytes 14579 (14.5 KB)
TX errors 0 dropped 0 overruns 0 carrier 6
```

2. Make sure that they can ping each other.

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

```
root@workstation:/home/kier# ping 192.168.56.139
PING 192.168.56.139 (192.168.56.139) 56(84) bytes of data.
64 bytes from 192.168.56.139: icmp_seq=1 ttl=64 time=0.299 ms
64 bytes from 192.168.56.139: icmp_seq=2 ttl=64 time=0.162 ms
```

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

```
root@workstation:/home/kier# ping 192.168.56.140
PING 192.168.56.140 (192.168.56.140) 56(84) bytes of data.
64 bytes from 192.168.56.140: icmp_seq=1 ttl=64 time=0.319 ms
64 bytes from 192.168.56.140: icmp_seq=2 ttl=64 time=0.208 ms
64 bytes from 192.168.56.140: icmp_seq=3 ttl=64 time=0.253 ms
```

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

```
root@server1:/home/kier# ping 192.168.56.140
PING 192.168.56.140 (192.168.56.140) 56(84) bytes of data.
64 bytes from 192.168.56.140: icmp_seq=1 ttl=64 time=0.302 ms
64 bytes from 192.168.56.140: icmp_seq=2 ttl=64 time=0.370 ms
64 bytes from 192.168.56.140: icmp_seq=3 ttl=64 time=0.230 ms
64 bytes from 192.168.56.140: icmp_seq=4 ttl=64 time=0.193 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 jvtaylor@192.168.56.120*

```
root@server1:/home/kier# ssh kier@192.168.56.139
The authenticity of host '192.168.56.139 (192.168.56.139)' can't be established.
ECDSA key fingerprint is SHA256:KyfP7PGKgICghVmFfDYIYshVXvhQ+qaq8nbv2JKq714.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.139' (ECDSA) to the list of known hosts.
kier@192.168.56.139's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.18.0-15-generic x86_64)
```

1.2 Enter the password for server 1 when prompted

```
kier@server1
```

1.3 Verify that you are in server 1. The user should be in this format `user@server1`.

For example, *jvtaylor@server1*

```
kier@server1
```

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


```
Connection to 192.168.56.139 closed.
```

3. Do the same for Server 2.

```
root@server2:/home/kier# ssh kier@192.168.56.140
kier@192.168.56.140's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.18.0-15-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

Enable ESM Infra to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Failed to connect to https://changelogs.ubuntu.com/meta-release-
Internet connection or proxy settings

Your Hardware Enablement Stack (HWE) is supported until April 20
*** System restart required ***
Last login: Fri Aug 30 12:27:42 2024 from 192.168.56.140
kier@server2:~$
```

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 2.9.3 /etc/hosts

127.0.0.1    localhost
127.0.0.1    SANJOSE.myguest.virtualbox.org  SANJOSE
192.168.56.139  server1
192.168.56.140  server2
```

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

```
GNU nano 2.9.3 /etc/hosts

127.0.0.1    localhost
127.0.0.1    SANJOSE.myguest.virtualbox.org  SANJOSE
192.168.56.139  server1
192.168.56.140  server2
```

- 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 jvtaylor@server1`. Enter the password when prompted. Verify that you have entered Server 1. Do the same for Server 2.

```
root@workstation:/home/kier# ssh kier@192.168.56.139
kier@192.168.56.139's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.18.0-15-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

Enable ESM Infra to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Failed to connect to https://changelogs.ubuntu.com/meta-release-l
Internet connection or proxy settings

Your Hardware Enablement Stack (HWE) is supported until April 202
*** System restart required ***
Last login: Fri Aug 30 12:28:13 2024 from 192.168.56.139
kier@server1:~$ ssh kier@192.168.56.140
```



```
kier@server1:~$ ssh kier@192.168.56.140
The authenticity of host '192.168.56.140 (192.168.56.140)' can't be
determined. ECDSA key fingerprint is SHA256:5V/PRUJyZqek3CymHrusNfbvgeLw15dy.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.140' (ECDSA) to the list of known hosts.
kier@192.168.56.140's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.18.0-15-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.
No security updates can be applied immediately.

To enable ESM Infra to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts.
Check your Internet connection or proxy settings

Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***
Last login: Fri Aug 30 12:30:50 2024 from 192.168.56.140
kier@server2:~$
```

Reflections:

Answer the following:

1. How are we able to use the hostname instead of IP address in SSH commands?
 - It relies on DNS so that we can use its hostname instead of IP address.
2. How secured is SSH?
 - SSH uses a strong encryption method that secures our logged information within the system and it can remotely access which offers convenience to its users.

