Name: Potestades, North Nygel G.	**Date Performed:** 8/8/25
Course/Section: CPE 212-CPE31S2	**Date Submitted:** 8/8/25
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st Semester 2025-2026

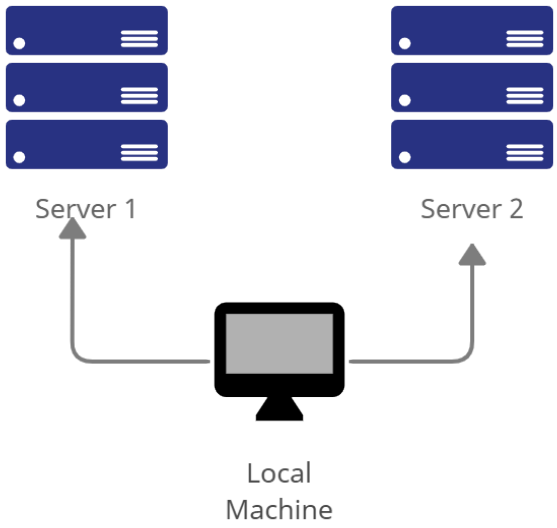
Activity 1: Configure Network using Virtual Machines

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

```
north@server1:~$
```

   1.2 Use server2 for Server 2

```
north@server2:~$
```

1.3 Use workstation for the Local Machine

```
north@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

```
127.0.0.1       localhost
127.0.0.1       server 1
```

   2.2 Type 127.0.0.1 server 2 for Server 2

```
127.0.0.1       localhost
127.0.0.1       server 2
```

   2.3 Type 127.0.0.1 workstation for the Local Machine

```
127.0.0.1       localhost
127.0.0.1       workstation
```

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.

```
north@server1:~$ sudo apt update
north@server1:~$ sudo apt upgrade
north@server2:~$ sudo apt update
north@server2:~$ sudo apt upgrade
north@workstation:~$ sudo apt update
h@workstation:~$ sudo apt upgrade
```

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

```
north@server1:~$ sudo apt install openssh-server
north@server2:~$ sudo apt install openssh-server
north@workstation:~$ sudo apt install openssh-server
```

3. Verify if the SSH service has started by issuing the following commands:
   3.1 sudo service ssh start

```
north@server1:~$ sudo service ssh start
north@server2:~$ sudo service ssh start
north@workstation:~$ sudo service ssh start
```

   3.2 sudo systemctl status ssh

```
north@server1:~$ sudo systemctl status ssh
north@server2:~$ sudo systemctl status ssh
north@workstation:~$ 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

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

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

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

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: 192.168.56.105
   1.2 Server 2 IP address: 192.168.56.106
   1.3 Local Machine IP address: 192.168.56.104
2. Make sure that they can ping each other.
   2.1 Connectivity test for Local Machine to Server 1: ✓ Successful ☐ Not Successful

```
north@workstation:~$ ping -c 4 192.168.56.105
PING 192.168.56.105 (192.168.56.105) 56(84) bytes of data.
64 bytes from 192.168.56.105: icmp_seq=1 ttl=64 time=0.577 ms
64 bytes from 192.168.56.105: icmp_seq=2 ttl=64 time=0.453 ms
64 bytes from 192.168.56.105: icmp_seq=3 ttl=64 time=0.600 ms
64 bytes from 192.168.56.105: icmp_seq=4 ttl=64 time=0.441 ms

--- 192.168.56.105 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3058ms
rtt min/avg/max/mdev = 0.441/0.517/0.600/0.076 ms
```

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

```
north@workstation:~$ ping -c 4 192.168.56.106
PING 192.168.56.106 (192.168.56.106) 56(84) bytes of data.
64 bytes from 192.168.56.106: icmp_seq=1 ttl=64 time=0.996 ms
64 bytes from 192.168.56.106: icmp_seq=2 ttl=64 time=0.484 ms
64 bytes from 192.168.56.106: icmp_seq=3 ttl=64 time=0.414 ms
64 bytes from 192.168.56.106: icmp_seq=4 ttl=64 time=0.404 ms

--- 192.168.56.106 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3058ms
rtt min/avg/max/mdev = 0.404/0.574/0.996/0.246 ms
```

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

```
north@server1:~$ ping -c 4 192.168.56.106
PING 192.168.56.106 (192.168.56.106) 56(84) bytes of data.
64 bytes from 192.168.56.106: icmp_seq=1 ttl=64 time=0.806 ms
64 bytes from 192.168.56.106: icmp_seq=2 ttl=64 time=0.390 ms
64 bytes from 192.168.56.106: icmp_seq=3 ttl=64 time=0.547 ms
64 bytes from 192.168.56.106: icmp_seq=4 ttl=64 time=0.491 ms

--- 192.168.56.106 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3054ms
rtt min/avg/max/mdev = 0.390/0.558/0.806/0.155 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*

```
north@workstation:~$ ssh north@192.168.56.105
The authenticity of host '192.168.56.105 (192.168.56.105)' can't be established
.
ECDSA key fingerprint is SHA256:HVNGpKGK5vNZ+izZLNZEpUpJcbAusL/3qSQV4UokMNc.
Are you sure you want to continue connecting (yes/no)?
```

1.2 Enter the password for server 1 when prompted

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

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

north@server1:~$
```

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

```
Connection to 192.168.56.105 closed.
north@workstation:~$
```

3. Do the same for Server 2.

```
north@workstation:~$ ssh north@192.168.56.106
The authenticity of host '192.168.56.106 (192.168.56.106)' can't be established
.
ECDSA key fingerprint is SHA256:nwdy+AQ8BmF3379JFRoO+/h/89Xkz1gJUqoF116gtWA.
Are you sure you want to continue connecting (yes/no)? yes
```

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

north@server2:~$
```

```
Connection to 192.168.56.106 closed.
north@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.

```
127.0.0.1       localhost
127.0.0.1       workstation
192.168.56.105  server1
192.168.56.106  server2
```

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.

```
north@workstation:~$ ssh north@server1
The authenticity of host 'server1 (192.168.56.105)' can't be established.
ECDSA key fingerprint is SHA256:HVNGpKGK5vNZ+izZLNZEpUpJcbAusL/3qSQV4UokMNc.
Are you sure you want to continue connecting (yes/no)? yes
Last login: Fri Aug  8 17:58:50 2025 from 192.168.56.104
north@server1:~$
```

```
north@workstation:~$ ssh north@server2
The authenticity of host 'server2 (192.168.56.106)' can't be established.
ECDSA key fingerprint is SHA256:nwdy+AQ8BmF3379JFRoO+/h/89Xkz1gJUqoF116gtWA.
Are you sure you want to continue connecting (yes/no)? yes
Last login: Fri Aug  8 18:00:23 2025 from 192.168.56.104
north@server2:~$
```

Reflections:

Answer the following:

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

This is because we added the hostnames and their associated IP addresses into the /etc/hosts file, meaning the name is now associated with the IP address assigned to it. The file acts as a local DNS server, letting the computer know which IP address is associated to each hostname.

2. How secured is SSH?

SSH is one of the safest, most secured ways to remotely access a computer. This is because it uses an encryption and authentication process to secure the connection between the two computers.