Name: Shaniah Rose Hope M. Sumaoang	**Date Performed: August 22, 2023**
Course/Section: CPE 232-CPE31S5	**Date Submitted: August 23, 2023**
Instructor: Engr. Roman Richard	**Semester and SY: 1st Semester 2023-2024**

Activity 1: Configure Network using Virtual Machines

1. **Objectives:**

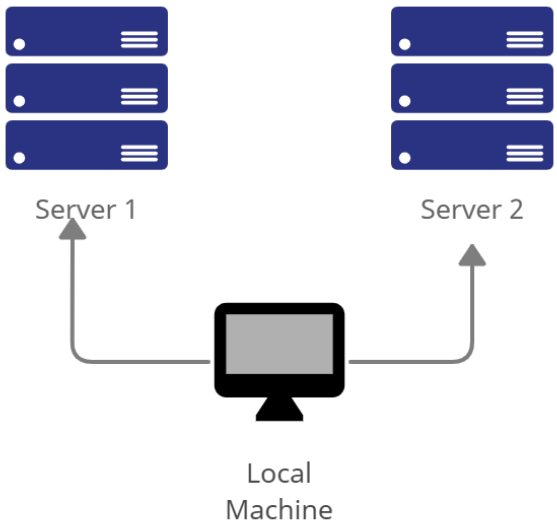
1.1. Create and configure Virtual Machines in Microsoft Azure or VirtualBox

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

2. **Discussion:**

Network Topology:

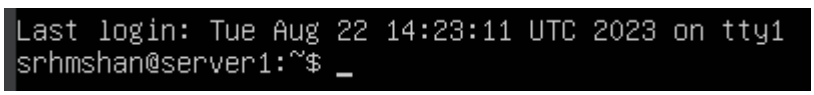
Assume that you have created the following network topology in Virtual Machines, *provide screenshots for each task*. (Note: *it is assumed that you have the prior knowledge of cloning and creating snapshots in a virtual machine*).



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



1.2 Use server2 for Server 2

```
Last login: Tue Aug 22 14:36:46 UTC 2023 on tty1
srhmshan@server2:~$
```

1.3 Use workstation for the Local Machine

```
srhmshan@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 localhost
127.0.0.1 server1-sumaoang_

# 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 server2-sumaoang_

# 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       localhost
127.0.0.1       srhmshan-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.

```
srhmshan@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
Hit:4 http://ph.archive.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Building dependency tree... 50%
```

```
srhmshan@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 cloud-init git git-man initramfs-tools initramfs-tools-bin initramfs-tools-core
  libapt-pkg6.0 libldap-2.5-0 libldap-common sosreport
12 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 7,747 kB of archives.
After this operation, 838 kB disk space will be freed.
```

```
srhmshan@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
Hit:4 http://ph.archive.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
12 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

```
srhmshan@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 cloud-init git git-man initramfs-tools initramfs-tools-bin initramfs-tools-core
  libapt-pkg6.0 libldap-2.5-0 libldap-common sosreport
12 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 7747 kB of archives.
After this operation, 838 kB disk space will be freed.
```

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

```
srhmshan@server1:~$ sudo apt install openssh-server
[sudo] password for srhmshan:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssh-server is already the newest version (1:8.9p1-3ubuntu0.3).
openssh-server set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

```
srhmshan@server2:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssh-server is already the newest version (1:8.9p1-3ubuntu0.3).
openssh-server set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

```
srhmshan@workstation:~$ sudo apt install openssh-server
[sudo] password for srhmshan:
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 26 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]
```

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

3.1 *sudo service ssh start*

```
srhmshan@server1:~$ sudo service ssh start
srhmshan@server1:~$ _
```

```
srhmshan@server2:~$ sudo service ssh start
srhmshan@server2:~$ _
```

```
srhmshan@workstation:~$ sudo service ssh start
srhmshan@workstation:~$
```

3.2 *sudo systemctl status ssh*

```
srhmshan@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 Tue 2023-08-22 15:28:23 UTC; 2min 17s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 675 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 717 (sshd)
      Tasks: 1 (limit: 4557)
     Memory: 4.4M
        CPU: 52ms
     CGroup: /system.slice/ssh.service
             └─717 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 22 15:28:22 server1 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 22 15:28:23 server1 sshd[717]: Server listening on 0.0.0.0 port 22.
Aug 22 15:28:23 server1 sshd[717]: Server listening on :: port 22.
Aug 22 15:28:23 server1 systemd[1]: Started OpenBSD Secure Shell server.
```

```
srhmshan@server2:~$ 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 Tue 2023-08-22 15:18:20 UTC; 12min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 666 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 710 (sshd)
      Tasks: 1 (limit: 4557)
     Memory: 4.4M
        CPU: 51ms
     CGroup: /system.slice/ssh.service
             └─710 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 22 15:18:19 server2 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 22 15:18:20 server2 sshd[710]: Server listening on 0.0.0.0 port 22.
Aug 22 15:18:20 server2 sshd[710]: Server listening on :: port 22.
Aug 22 15:18:20 server2 systemd[1]: Started OpenBSD Secure Shell server.
```

```
srhmshan@workstation:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: e>
     Active: active (running) since Tue 2023-08-22 23:29:19 PST; 2min 27s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 2486 (sshd)
      Tasks: 1 (limit: 4591)
     Memory: 1.7M
        CPU: 13ms
     CGroup: /system.slice/ssh.service
             └─2486 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 22 23:29:19 workstation systemd[1]: Starting OpenBSD Secure Shell server...
Aug 22 23:29:19 workstation sshd[2486]: Server listening on 0.0.0.0 port 22.
Aug 22 23:29:19 workstation sshd[2486]: Server listening on :: port 22.
Aug 22 23:29:19 workstation systemd[1]: Started OpenBSD Secure Shell server.
lines 1-16
```

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

4.1 *sudo ufw allow ssh*

```
srhmshan@server1:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
```

```
srhmshan@server2:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
```

```
srhmshan@workstation:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
```

4.2 *sudo ufw enable*

```
srhmshan@server1:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

```
srhmshan@server2:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

```
srhmshan@workstation:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

4.3 *sudo ufw status*

```
srhmshan@server1:~$ sudo ufw status
Status: active

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

```
srhmshan@server2:~$ sudo ufw status
Status: active

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

```
srhmshan@workstation:~$ 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: 192.168.56.101

```
srhmshan@server1:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAS
        inet 192.168.56.101  n
```

   1.2 Server 2 IP address: 192.168.56.105

```
srhmshan@server2:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAS
        inet 192.168.56.105  n
```

   1.3 Server 3 IP address: 192.168.56.104

```
srhmshan@workstation:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,R
        inet 192.168.56.104  netm
```

2. Make sure that they can ping each other.
   2.1 Connectivity test for Local Machine 1 to Server 1: ☑ Successful ☐ Not Successful

```
srhmshan@workstation:~$ ping 192.168.56.101
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.
64 bytes from 192.168.56.101: icmp_seq=1 ttl=64 time=0.946 ms
64 bytes from 192.168.56.101: icmp_seq=2 ttl=64 time=0.367 ms
64 bytes from 192.168.56.101: icmp_seq=3 ttl=64 time=0.410 ms
64 bytes from 192.168.56.101: icmp_seq=4 ttl=64 time=0.461 ms
64 bytes from 192.168.56.101: icmp_seq=5 ttl=64 time=0.600 ms
64 bytes from 192.168.56.101: icmp_seq=6 ttl=64 time=0.349 ms
64 bytes from 192.168.56.101: icmp_seq=7 ttl=64 time=0.537 ms
^C
--- 192.168.56.101 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6206ms
rtt min/avg/max/mdev = 0.349/0.524/0.946/0.191 ms
```

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

```
srhmshan@workstation:~$ ping 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=0.688 ms
64 bytes from 192.168.56.105: icmp_seq=2 ttl=64 time=0.472 ms
64 bytes from 192.168.56.105: icmp_seq=3 ttl=64 time=0.376 ms
64 bytes from 192.168.56.105: icmp_seq=4 ttl=64 time=0.583 ms
64 bytes from 192.168.56.105: icmp_seq=5 ttl=64 time=0.451 ms
64 bytes from 192.168.56.105: icmp_seq=6 ttl=64 time=0.892 ms
64 bytes from 192.168.56.105: icmp_seq=7 ttl=64 time=0.746 ms
^C
--- 192.168.56.105 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6122ms
rtt min/avg/max/mdev = 0.376/0.601/0.892/0.170 ms
```

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

```
srhmshan@server1:~$ ping 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=4.94 ms
64 bytes from 192.168.56.105: icmp_seq=2 ttl=64 time=0.986 ms
64 bytes from 192.168.56.105: icmp_seq=3 ttl=64 time=0.678 ms
64 bytes from 192.168.56.105: icmp_seq=4 ttl=64 time=0.521 ms
64 bytes from 192.168.56.105: icmp_seq=5 ttl=64 time=0.975 ms
64 bytes from 192.168.56.105: icmp_seq=6 ttl=64 time=0.764 ms
64 bytes from 192.168.56.105: icmp_seq=7 ttl=64 time=0.740 ms
^C
--- 192.168.56.105 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6015ms
rtt min/avg/max/mdev = 0.521/1.371/4.937/1.463 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*

```
srhmshan@workstation:~$ ssh srhmshan@192.168.56.101
The authenticity of host '192.168.56.101 (192.168.56.101)' can't be established.
ED25519 key fingerprint is SHA256:AgzbbiTEs9IIypidSJAzYuB5pC6rozMAFT7t1Ac+Nw4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.56.101' (ED25519) to the list of known hosts
.
```

1.2 Enter the password for server 1 when prompted

```
srhmshan@192.168.56.101's password:
```

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

```
srhmshan@server1:~$
```

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

```
srhmshan@server1:~$
logout
Connection to 192.168.56.101 closed.
```

3. Do the same for Server 2.

```
srhmshan@workstation:~$ ssh srhmshan@192.168.56.105
srhmshan@192.168.56.105's password:
Last login: Tue Aug 22 17:10:35 2023
srhmshan@server2:~$
```

```
srhmshan@server2:~$
logout
Connection to 192.168.56.105 closed.
```

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       localhost
127.0.0.1       srhmshan-VirtualBox
192.168.56.101  server1
192.168.56.105  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.

```
srhmshan@workstation:~$ ssh srhmshan@server1
The authenticity of host 'server1 (192.168.56.101)' can't be established.
ED25519 key fingerprint is SHA256:AgzbbiTEs9IIypidSJAzYuB5pC6rozMAFT7t1Ac+Nw4.
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.
srhmshan@server1's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)
```

```
srhmshan@workstation:~$ ssh srhmshan@server2
The authenticity of host 'server2 (192.168.56.105)' can't be established.
ED25519 key fingerprint is SHA256:G53wJvrR1NQcS2x1seqXwpNpKJfNIFt0CiuF+o2W5XU.
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.
srhmshan@server2's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)
```

Reflections:

Answer the following:

1. How are we able to use the hostname instead of IP address in SSH commands?
   I opened /etc/hosts to encode the IP address to assign it as the servers' IP address.
2. How secured is SSH?

   SSH in Ubuntu is secure when used properly. It can be used to communicate to others within the same network. Regardless, its security and management need to be maintained and facilitated.

Conclusion:

In conclusion, SSH in Ubuntu can achieve strong security through accurate setup and consistent management. It's essential to adhere to recommended security measures, keep your system up to date, and remain aware of any possible security issues affecting SSH and Ubuntu. In this activity, I had a quick review on how to create and configure virtual machines in VirtualBox. I also learned the basics of setting up a virtual network and testing connectivity between virtual machines. I had encountered some problems but I managed to troubleshoot them and complete the activity.