

Name:Tracey Dee Bringuela	Date Performed:8/25/24
Course/Section:CPE31S2	Date Submitted:8/25/24
Instructor: Robin Valenzuela	Semester and SY: 1st semester 2024-2025

### Activity 1: Configure Network using Virtual Machines

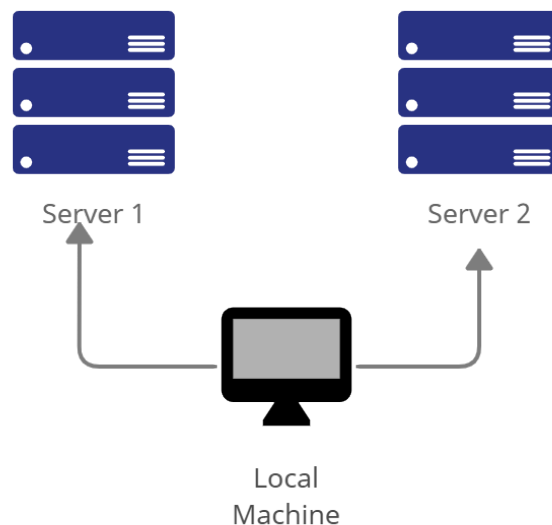
#### 1. Objectives:

- 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, *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

```
vboxuser@server1: ~  
GNU nano 6.2 /etc/hostname  
server1
```

```
vboxuser@server1:~$ sudo nano /etc/hostname
```

1.2 Use server2 for Server 2

```
vboxuser@server2:~$ sudo nano /etc/hostname
```

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

1.3 Use workstation for the Local Machine

```
vboxuser@workstation:~$ sudo nano /etc/hostname
```

```
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

```
vboxuser@server1:~$ sudo nano /etc/hosts
```

```
GNU nano 6.2 /etc/hosts  
127.0.0.1 localhost  
127.0.0.1 server1
```

2.2 Type 127.0.0.1 server 2 for Server 2

```
vboxuser@server2:~$ sudo nano /etc/hosts
```

```
GNU nano 6.2 /etc/hosts  
127.0.0.1 localhost  
127.0.0.1 server2
```

2.3 Type 127.0.0.1 workstation for the Local Machine

```
vboxuser@workstation:~$ sudo nano /etc/hosts
```

```
GNU nano 6.2 /etc/hosts  
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.

```
vboxuser@server1: ~  
vboxuser@server1:~$ sudo apt update  
[sudo] password for vboxuser:  
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease  
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]  
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease  
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]  
Get:5 http://ph.archive.ubuntu.com/ubuntu jammy/main Translation-en_GB [483 kB]  
Get:6 http://ph.archive.ubuntu.com/ubuntu jammy/restricted Translation-en_GB [5,768 B]  
Get:7 http://ph.archive.ubuntu.com/ubuntu jammy/universe Translation-en_GB [838 kB]  
Get:8 http://ph.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en_GB [102 kB]  
Fetched 1,686 kB in 2s (1,006 kB/s)  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
326 packages can be upgraded. Run 'apt list --upgradable' to see them.  
vboxuser@server1:~$ sudo apt upgrade -y  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
Calculating upgrade... Done  
The following packages were automatically installed and are no longer required:
```

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

```
vboxuser@server1:~$ sudo apt install openssh-server  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following packages were automatically installed and are no longer  
  libwpe-1.0-1 libwpebackend-fdo-1.0-1  
Use 'sudo apt autoremove' to remove them.  
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 to upgrade, 4 to newly install, 0 to remove and 6 not to upgrade.  
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-updates/main amd64 openssh-server amd64 1:8.9p1-3ubuntu0.10 [38.9 kB]  
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 openssh-sftp-server amd64 1:8.9p1-3ubuntu0.10 [435 kB]  
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 ncurses-term all 6.3-2ubuntu0.1 [267 kB]  
Get:4 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 ssh-import-id
```

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*

```
vboxuser@server1:~$ sudo service ssh start
vboxuser@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 Sun 2024-08-25 16:07:48 CEST; 1min 15s ago
     Docs: man:sshd(8)
           man:sshd_config(5)
    Main PID: 32013 (sshd)
      Tasks: 1 (limit: 2272)
     Memory: 1.7M
        CPU: 24ms
    CGroup: /system.slice/ssh.service
            └─32013 "sshd: /usr/sbin/sshd -D [listener] 0 of 128 max 1024"

Aug 25 16:07:48 server1 systemd[1]: Starting OpenBSD Secure Shell server: sshd.
Aug 25 16:07:48 server1 sshd[32013]: Server listening on 0.0.0.0 port 22.
Aug 25 16:07:48 server1 sshd[32013]: Server listening on :: port 22.
Aug 25 16:07:48 server1 systemd[1]: Started OpenBSD Secure Shell server: sshd.
lines 1-16/16 (END)
```

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*

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

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

vboxuser@server1:~$
```

**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 netwDork 192.168.31.XX.

```
vboxuser@server1:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.31.75 netmask 255.255.255.0 broadcast 192.168.31.255
    inet6 fe80::b2f9:59ae:199c:1ebb prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:6e:45:54 txqueuelen 1000 (Ethernet)
    RX packets 16522 bytes 4133361 (4.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1046 bytes 88671 (88.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collision 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 225 bytes 21275 (21.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 225 bytes 21275 (21.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collision 0

vboxuser@server1:~$
```

2.

1.1 Server 1 IP address: 192.168.31.75

1.2 Server 2 IP address: 192.168.31.133

1.3 Server 3 IP address: 192.168.31.58

3. Make sure that they can ping each other.

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

```
vboxuser@workstation:~$ ping -c 4 192.168.31.75
PING 192.168.31.75 (192.168.31.75) 56(84) bytes of data:
64 bytes from 192.168.31.75: icmp_seq=1 ttl=64 time=2.74 ms
64 bytes from 192.168.31.75: icmp_seq=2 ttl=64 time=0.563 ms
64 bytes from 192.168.31.75: icmp_seq=3 ttl=64 time=0.928 ms
64 bytes from 192.168.31.75: icmp_seq=4 ttl=64 time=0.769 ms

--- 192.168.31.75 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 0.563/1.250/2.741/0.870 ms
vboxuser@workstation:~$ ss
```

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

```
vboxuser@workstation:~$ ping -c 4 192.168.31.133
PING 192.168.31.133 (192.168.31.133) 56(84) bytes of data.
64 bytes from 192.168.31.133: icmp_seq=1 ttl=64 time=1.63 ms
64 bytes from 192.168.31.133: icmp_seq=2 ttl=64 time=1.14 ms
64 bytes from 192.168.31.133: icmp_seq=3 ttl=64 time=0.893 ms
64 bytes from 192.168.31.133: icmp_seq=4 ttl=64 time=0.642 ms
```

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

```
vboxuser@server1:~$ ping -c 4 192.168.31.58
PING 192.168.31.58 (192.168.31.58) 56(84) bytes of data.
64 bytes from 192.168.31.58: icmp_seq=1 ttl=64 time=0.933 ms
64 bytes from 192.168.31.58: icmp_seq=2 ttl=64 time=1.90 ms
64 bytes from 192.168.31.58: icmp_seq=3 ttl=64 time=0.573 ms
64 bytes from 192.168.31.58: icmp_seq=4 ttl=64 time=0.598 ms

--- 192.168.31.58 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3022ms
rtt min/avg/max/mdev = 0.573/0.999/1.895/0.536 ms
```

**ATask 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`

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, `jvtaylor@server1`

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

```
vboxuser@workstation:~$ ssh vboxuser@192.168.31.75
The authenticity of host '192.168.31.75 (192.168.31.75)' can't be es
ED25519 key fingerprint is SHA256:ng4lFU3yqrpLWCBjRs7yYfS0AkcmF6shIt
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.31.75' (ED25519) to the list of
.
vboxuser@192.168.31.75's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.8.0-40-generic x86_64)

 * Documentation:  https://help.ubuntu.com
Ubuntu Software   https://landscape.canonical.com
                  https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

*** System restart required ***
Last login: Sun Aug 25 14:37:24 2024
vboxuser@server1:~$
logout
Connection to 192.168.31.75 closed.
vboxuser@workstation:~$
```

3. Do the same for Server 2.



```
vboxuser@workstation:~$ ssh vboxuser@192.168.31.133
The authenticity of host '192.168.31.133 (192.168.31.133)' can't be
ED25519 key fingerprint is SHA256:Plx4E2QYrDewfeqXhhlAdXhdDDAiBf4c29
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Warning: Permanently added '192.168.31.133' (ED25519) to the list of
S.
vboxuser@192.168.31.133's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.8.0-40-generic x86_64)

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

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

*** System restart required ***
Last login: Sun Aug 25 14:37:24 2024
vboxuser@server2:~$
logout
Connection to 192.168.31.133 closed.
vboxuser@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.



```

GNU nano 6.2 /etc/hosts *
127.0.0.1    localhost
127.0.0.1    workstation
192.168.31.75 server1
192.168.31.133 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 jvtaylor@server1`. Enter the password when prompted. Verify that you have entered Server 1. Do the same for Server 2.

server 1

```

vboxuser@workstation:~$ ssh vboxuser@server1
The authenticity of host 'server1 (192.168.31.75)' can't be established.
ED25519 key fingerprint is SHA256:ng4lFU3yqrpLWCBjRs7yYfS0AkcmF6shIf
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])?
Warning: Permanently added 'server1' (ED25519) to the list of known hosts.
vboxuser@server1's password:
Permission denied, please try again.
vboxuser@server1's password:
Permission denied, please try again.
vboxuser@server1's password:
vboxuser@server1:~$
Ubuntu 22.04.4 LTS (GNU/Linux 6.8.0-40-generic x86_64)
LibreOffice Writer
* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

*** System restart required ***
Last login: Sun Aug 25 18:36:36 2024 from 192.168.31.58
vboxuser@server1:~$

```

server 2

```
vboxuser@workstation:~$ ssh vboxuser@server2
The authenticity of host 'server2 (192.168.31.133)' can't be established.
ED25519 key fingerprint is SHA256:Plx4E2QYrDewfeqXhhlAdXhdDDAiBf4c29
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])?
Warning: Permanently added 'server2' (ED25519) to the list of known hosts.
vboxuser@server2's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.8.0-40-generic x86_64)

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

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

*** System restart required ***
Last login: Sun Aug 25 18:38:50 2024 from 192.168.31.58
vboxuser@server2:~$
```

### Reflections:

Answer the following:

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

Using the hostname instead of the IP address in SSH commands works because the hostname is mapped to the correct IP address in the `/etc/hosts` file or via DNS. When you set up the `/etc/hosts` file with the server's IP address and hostname on your local machine, it enables the system to resolve the hostname to the appropriate IP, allowing you to use the hostname in SSH commands.

2. How secured is SSH?

SSH is very secure due to its encryption of data between the client and server, which prevents interception and tampering. It also supports robust authentication methods, including passwords, public keys, and two-factor authentication, adding extra security layers. Moreover, SSH safeguards against various network threats like IP spoofing and DNS spoofing.

