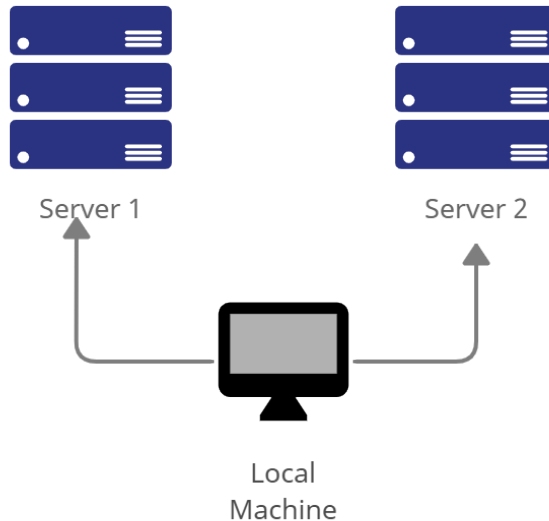
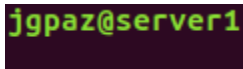


Name: Paz, John Glen L.	Date Performed: Aug. 17, 2023
Course/Section: CPE 232 - CPE31S6	Date Submitted: Aug. 17, 2023
Instructor: Dr. Jonathan F. Taylar	Semester and SY: 1st Sem. - 2023
Activity 1: Configure Network using Virtual Machines	
1. Objectives: 1.1. Create and configure Virtual Machines in 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, <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] </pre> <p>The diagram illustrates a network topology where a central 'Local Machine' (represented by a monitor icon) is connected to two separate server stacks. 'Server 1' and 'Server 2' are each represented by three stacked server rack icons. Arrows point from the 'Local Machine' to both 'Server 1' and 'Server 2'.</p>	
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 <i>sudo nano /etc/hostname</i> 1.1 Use server1 for Server 1	
 <pre> jgpaz@server1 </pre> <p>A terminal window screenshot showing the prompt 'jgpaz@server1' in green text on a black background.</p>	

1.2 Use server2 for Server 2

```
jgpaz@server2
```

1.3 Use workstation for the Local Machine

```
jgpaz@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.

```
jgpaz@server1:~$ sudo apt update | sudo apt upgrade
```

```
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
```

```
Reading package lists... Done
```

```
jgpaz@server2:~$ sudo apt update | sudo apt upgrade
```

```
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
```

```
Reading package lists... Done
```

```
Building dependency tree
```

```
Reading state information... Done
```

```
Calculating upgrade... Done
```

```
The following package was automatically installed and is no longer required:
```

```
liblvm7
```

```
jgpaz@workstation:~$ sudo apt update | sudo apt upgrade
```

```
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
```

```
Reading package lists... Done
```

```
Building dependency tree
```

```
Reading state information... Done
```

```
Calculating upgrade... Done
```

```
The following package was automatically installed and is no longer required:
```

```
liblvm7
```

```
Use 'sudo apt autoremove' to remove it.
```

```
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

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

```
jgpaz@server1:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
```

```
jgpaz@server2:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere rssh ssh-askpass
```

```
jgpaz@workstation:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
```

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*

```
jgpaz@server1:~$ sudo service ssh start
jgpaz@server1:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: ena
   Active: active (running) since Thu 2023-08-17 17:53:39 PST; 3min 49s ago
     Main PID: 2645 (sshd)
       Tasks: 1 (limit: 4656)
      CGroup: /system.slice/ssh.service
              └─2645 /usr/sbin/sshd -D

Aug 17 17:53:39 server1 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 17 17:53:39 server1 sshd[2645]: Server listening on 0.0.0.0 port 22.
Aug 17 17:53:39 server1 sshd[2645]: Server listening on :: port 22.
Aug 17 17:53:39 server1 systemd[1]: Started OpenBSD Secure Shell server.
lines 1-12/12 (END)
```

```
jgpaz@server2:~$ sudo service ssh start
jgpaz@server2:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: ena
   Active: active (running) since Thu 2023-08-17 17:55:46 PST; 2min 28s ago
   Main PID: 4241 (sshd)
     Tasks: 1 (limit: 4656)
    CGroup: /system.slice/ssh.service
            └─4241 /usr/sbin/sshd -D

Aug 17 17:55:46 server2 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 17 17:55:46 server2 sshd[4241]: Server listening on 0.0.0.0 port 22.
Aug 17 17:55:46 server2 sshd[4241]: Server listening on :: port 22.
Aug 17 17:55:46 server2 systemd[1]: Started OpenBSD Secure Shell server.
lines 1-12/12 (END)
```

```
jgpaz@workstation:~$ sudo service ssh start
jgpaz@workstation:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: ena
   Active: active (running) since Thu 2023-08-17 17:43:55 PST; 15min ago
   Main PID: 2606 (sshd)
     Tasks: 1 (limit: 4656)
    CGroup: /system.slice/ssh.service
            └─2606 /usr/sbin/sshd -D

Aug 17 17:43:55 workstation systemd[1]: Starting OpenBSD Secure Shell server...
Aug 17 17:43:55 workstation sshd[2606]: Server listening on 0.0.0.0 port 22.
Aug 17 17:43:55 workstation sshd[2606]: Server listening on :: port 22.
Aug 17 17:43:55 workstation systemd[1]: Started OpenBSD Secure Shell server.
lines 1-12/12 (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*

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

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

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

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

```
jgpaz@workstation:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
jgpaz@workstation:~$ sudo ufw enable
Firewall is active and enabled on system startup
jgpaz@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: 192.168.56.102

```
inet 192.168.56.102
```

1.2 Server 2 IP address: 192.168.56.103

```
inet 192.168.56.103
```

1.3 Server 3 IP address: 192.168.56.101

```
inet 192.168.56.101
```

2. Make sure that they can ping each other.

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

```
jgpaz@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=0.879 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=2.11 ms
64 bytes from 192.168.56.102: icmp_seq=3 ttl=64 time=1.49 ms
64 bytes from 192.168.56.102: icmp_seq=4 ttl=64 time=0.685 ms
64 bytes from 192.168.56.102: icmp_seq=5 ttl=64 time=1.79 ms
```

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

```
jgpaz@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=1.76 ms
64 bytes from 192.168.56.103: icmp_seq=2 ttl=64 time=0.453 ms
64 bytes from 192.168.56.103: icmp_seq=3 ttl=64 time=0.725 ms
64 bytes from 192.168.56.103: icmp_seq=4 ttl=64 time=0.579 ms
64 bytes from 192.168.56.103: icmp_seq=5 ttl=64 time=0.424 ms
```

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

```
64 bytes from 192.168.56.103: icmp_seq=93 ttl=64 time=1.10 ms
64 bytes from 192.168.56.103: icmp_seq=94 ttl=64 time=0.935 ms
64 bytes from 192.168.56.103: icmp_seq=95 ttl=64 time=0.579 ms
64 bytes from 192.168.56.103: icmp_seq=96 ttl=64 time=1.59 ms
64 bytes from 192.168.56.103: icmp_seq=97 ttl=64 time=1.79 ms
64 bytes from 192.168.56.103: icmp_seq=98 ttl=64 time=0.550 ms
^C
--- 192.168.56.103 ping statistics ---
98 packets transmitted, 98 received, 0% packet loss, time 97878ms
rtt min/avg/max/mdev = 0.380/1.598/22.209/2.629 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`

```
jgpaz@workstation:~$ ssh jgpaz@192.168.56.102
The authenticity of host '192.168.56.102 (192.168.56.102)' can't be established.
ECDSA key fingerprint is SHA256:noG7j7p0EMx4+DdQ0B8wVV/4qnGay2jJD/mvH0g7mo8.
Are you sure you want to continue connecting (yes/no)? yes
```

1.2 Enter the password for server 1 when prompted

```
jgpaz@192.168.56.102's password: █
```

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

```
jgpaz@server1:~$ █
```

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

```
jgpaz@server1:~$ logout
Connection to 192.168.56.102 closed.
jgpaz@workstation:~$
```

3. Do the same for Server 2.

```
jgpaz@workstation:~$ ssh jgpaz@192.168.56.103
The authenticity of host '192.168.56.103 (192.168.56.103)' can't
.
ECDSA key fingerprint is SHA256:1+jS2pMQmvmsPpPWv5CgmWavrse4/JA
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.103' (ECDSA) to the list
jgpaz@192.168.56.103's password: █
```

```
jgpaz@server2:~$ sudo nano /etc/hosts
[sudo] password for jgpaz:
jgpaz@server2:~$ sudo nano /etc/hosts
jgpaz@server2:~$ ^C
jgpaz@server2:~$ logout
Connection to 192.168.56.103 closed.
jgpaz@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)

```
jgpaz@workstation:~$ sudo nano /etc/hosts
jgpaz@workstation:~$ ssh jgpaz@server1
The authenticity of host 'server1 (192.168.56.102)' can't be established.
ECDSA key fingerprint is SHA256:noG7j7pOEMx4+DdQ0B8wVV/4qnGay2jJD/mvH0g7mc
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server1' (ECDSA) to the list of known hosts.
jgpaz@server1's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

78 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 18.04 at
https://ubuntu.com/18-04

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Thu Aug 17 18:21:47 2023 from 192.168.56.101
jgpaz@server1:~$ █
```



```
jgpaz@workstation:~$ ssh jgpaz@server2
The authenticity of host 'server2 (192.168.56.103)' can't be established.
ECDSA key fingerprint is SHA256:1+jS2pMQmvmsPpPWv5CgmWavrse4/JA3PNrtAPv.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server2' (ECDSA) to the list of known hosts.
jgpaz@server2's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

78 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 18.04 at
https://ubuntu.com/18-04

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Thu Aug 17 18:23:45 2023 from 192.168.56.101
jgpaz@server2:~$
```



```

jgpaz@workstation:~$ ssh jgpaz@server2
The authenticity of host 'server2 (192.168.56.103)' can't be established
ECDSA key fingerprint is SHA256:1+jS2pMQmvmsPpPWv5CgmWavrse4/JA3PNrtAPvI
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server2' (ECDSA) to the list of known hosts.
jgpaz@server2's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

78 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 18.04 at
https://ubuntu.com/18-04

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Thu Aug 17 18:23:45 2023 from 192.168.56.101
jgpaz@server2:~$ logout
Connection to server2 closed.
jgpaz@workstation:~$ █

```

4.3 Save the file and exit.

- 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.

Reflections:

Answer the following:

- How are we able to use the hostname instead of IP address in SSH commands?
The system uses a procedure known as Domain Name System (DNS) resolution to convert the hostname into the associated IP address when you use the hostname rather than an IP address in SSH commands.
- How secured is SSH? - As we know, It contains powerful encryption and authentication features, making it a standard choice for secure system

communication. It's crucial to comprehend a technology's advantages as well as any potential weaknesses, though, just as with any other technology.

Conclusion:

In this laboratory activity, I remember from my previous activities in Cisco 1 on how to ping the ip addresses and I learned that the process of cloning servers is slightly long. I also learned that SSH provides a reliable and secure method of duplicating server settings and it offers a solid basis for protecting the server cloning process by encapsulating secure remote access, encrypted data transfer, and secure command execution. Administrators can confidently duplicate server configurations, applications, and data from source to target servers by using SSH, while reducing the risks of data breaches, illegal access, and tampering. Only authorized workers can start and manage the cloning operation thanks to the inclusion of SSH's encryption and authentication protocols, which also guarantees the confidentiality of sensitive data.