

<b>Name:</b> Kazuki A. Ogata	<b>Date Performed:</b> August 22, 2023
<b>Course/Section:</b> CPE 232 - CPE31S5	<b>Date Submitted:</b> August 23, 2023
<b>Instructor:</b> Engr. Roman Richard	<b>Semester and SY:</b> 1st semester S.Y 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 prior knowledge of cloning and creating snapshots in a virtual machine).

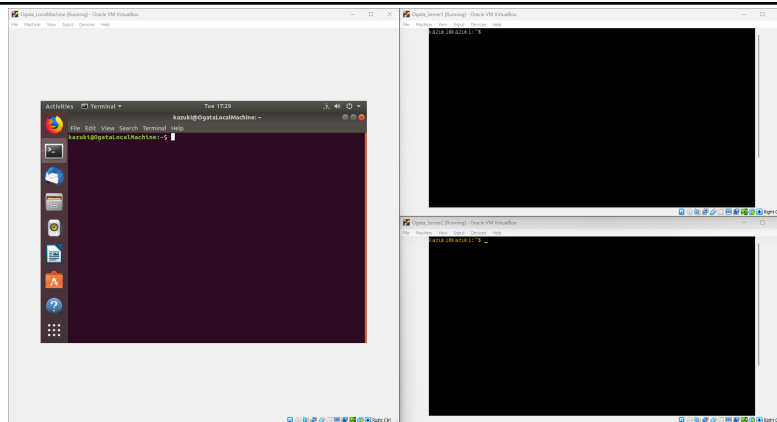
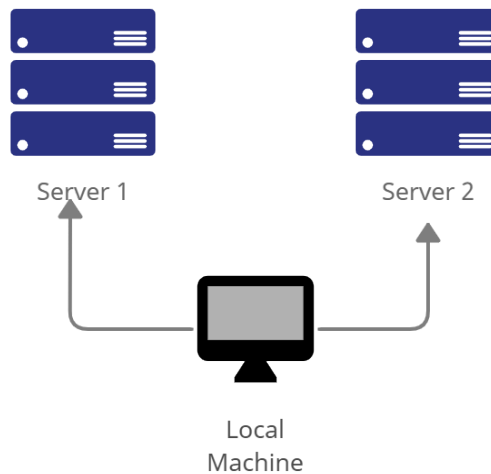


Figure 1.1: Installation of Local Machine, Server 1, and Server 2

I was able to create a workstation and two servers with the help of my past knowledge about their installation from our SysAd 1 course.

**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

```
kazuki@kazuki:~$ sudo nano /etc/hostname
GNU nano 6.2 /etc/hostname
kazuki -> server1 -> kazuki@server1:~$ hostname
server1
```

- 1.2 Use server2 for Server 2

```
kazuki@kazuki:~$ sudo nano /etc/hostname
GNU nano 6.2 /etc/hostname
kazuki -> server2 -> kazuki@server2:~$ hostname
server2
```

- 1.3 Use workstation for the Local Machine

```
root@LocalMachine0gata:/home/kazuki# sudo nano /etc/hostname
GNU nano 6.2 /etc/hostname
LocalMachine0gata -> workstation -> root@workstation:/home/kazuki# hostname
workstation
```

### Task 1.1: Configuration of Hostname File

Local Machine, Server 1, and Server 2, hostnames settings were modified using the "sudo nano /etc/hostname" command. The hostnames were changed as "server1" for Server 1, "server2" for Server 2, and "workstation" for Local Machine.

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

```
kazuki@server1:~$ sudo nano /etc/hosts
GNU nano 6.2 /etc/hosts
127.0.0.1 localhost
127.0.1.1 kazuki -> 127.0.0.1 localhost
127.0.0.1 server1
```

- 2.2 Type 127.0.0.1 server 2 for Server 2

```
kazuki@server2:~$ sudo nano /etc/hosts
GNU nano 6.2 /etc/hosts
127.0.0.1 localhost
127.0.1.1 kazuki -> 127.0.0.1 localhost
127.0.1.1 server2
```

- 2.3 Type 127.0.0.1 workstation for the Local Machine

```
root@workstation:/home/kazuki# sudo nano /etc/hosts
GNU nano 6.2 /etc/hosts
127.0.0.1 localhost
127.0.1.1 LocalMachine0gata.myquest.virtualbox.org LocalMachine0gata -> 127.0.0.1 localhost
127.0.0.1 workstation
```

### Task 1.2: Configuration of Hosts File

The hosts file was edited using the "sudo nano /etc/hosts" command, changing the second line to IP addresses and hostnames. "127.0.0.1 server1" for Server 1, "127.0.0.1 server2" for Server 2, and "127.0.0.1 workstation" for Local Machine.

## 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.

### -sudo apt update-

```
kazuki@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... Done
Reading state information... Done
78 packages can be upgraded. Run 'apt list --upgradable' to see them.
kazuki@server1:~$
```

```
kazuki@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
78 packages can be upgraded. Run 'apt list --upgradable' to see them.
kazuki@server2:~$
```

```
root@workstation:/home/kazuki# sudo apt update
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43.0 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Metadata [40.1 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [16.5 kB]
Fetched 210 kB in 2s (93.8 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
327 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@workstation:/home/kazuki#
```

### -sudo apt upgrade-

```
78 packages can be upgraded. Run 'apt list --upgradable' to see them.
kazuki@server1:~$ sudo apt upgrade
Reading package lists... Done
After this operation, 484 kB of additional disk space will be used.
Do you want to continue? [Y/n] y_

Progress: [ 99%] [#####.]

kazuki@server2:~$ sudo apt upgrade
Reading package lists... Done
After this operation, 484 kB of additional disk space will be used.
Do you want to continue? [Y/n] y_

Progress: [ 99%] [#####.]

root@workstation:/home/kazuki# sudo apt upgrade
Reading package lists... Done
After this operation, 123 MB of additional disk space will be used.
Do you want to continue? [Y/n] y_
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43.0 kB]
Processing triggers for initramfs-tools (0.140ubuntu13.4) ...
update-initramfs: Generating /boot/initrd.img-6.2.0-26-generic
root@workstation:/home/kazuki#
```

## Task 2.1: Updating and Upgrading Packages

Successfully updated and upgraded the packages using the "sudo apt update" and "sudo apt upgrade" commands. We do this to enhance system stability.

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

```
kazuki@server1:~$ sudo apt install openssh-server
[sudo] password for kazuki:
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.
```

```
kazuki@server2:~$ sudo apt install openssh-server
[sudo] password for kazuki:
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.
```

```
root@workstation:/home/kazuki# sudo apt install openssh-server
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 2 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] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 open
```

## Task 2.2: Installing the SSH Server

Successfully Installed the SSH server using the "sudo apt install openssh-server" command. We do this to secure remote access in SSH Server.

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

### 3.1 *sudo service ssh start*

```
kazuki@server1:~$ sudo service ssh start
kazuki@server1:~$
```

```
kazuki@server2:~$ sudo service ssh start
kazuki@server2:~$
```

```
root@workstation:/home/kazuki# sudo service ssh start
root@workstation:/home/kazuki#
```

### 3.2 *sudo systemctl status ssh*

```
kazuki@server1:~$ sudo service ssh start
kazuki@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 21:20:22 UTC; 9h ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Main PID: 15994 (sshd)
     Tasks: 1 (limit: 4557)
    Memory: 1.7M
         CPU: 85ms
   CGroup: /system.slice/ssh.service
           └─15994 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 22 21:20:21 server1 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 22 21:20:22 server1 sshd[15994]: Server listening on 0.0.0.0 port 22.
Aug 22 21:20:22 server1 sshd[15994]: Server listening on :: port 22.
Aug 22 21:20:22 server1 systemd[1]: Started OpenBSD Secure Shell server.
```

```
kazuki@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 21:20:26 UTC; 9h ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Process: 15976 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 15982 (sshd)
     Tasks: 1 (limit: 4557)
    Memory: 1.7M
         CPU: 74ms
   CGroup: /system.slice/ssh.service
           └─15982 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 22 21:20:26 server2 systemd[1]: Starting OpenBSD Secure Shell server...
Aug 22 21:20:26 server2 sshd[15982]: Server listening on 0.0.0.0 port 22.
Aug 22 21:20:26 server2 sshd[15982]: Server listening on :: port 22.
Aug 22 21:20:26 server2 systemd[1]: Started OpenBSD Secure Shell server.
```

```

root@workstation:/home/kazuki# 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 Wed 2023-08-23 05:23:55 +08; 9h ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Main PID: 36446 (sshd)
    Tasks: 1 (limit: 4593)
   Memory: 1.7M
      CPU: 19ms
   CGroup: /system.slice/ssh.service
           └─36446 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Aug 23 05:23:55 workstation systemd[1]: Starting OpenBSD Secure Shell server...
Aug 23 05:23:55 workstation sshd[36446]: Server listening on 0.0.0.0 port 22.
Aug 23 05:23:55 workstation sshd[36446]: Server listening on :: port 22.
Aug 23 05:23:55 workstation systemd[1]: Started OpenBSD Secure Shell server.

```

### Task 2.3: Verifying the SSH Service

When we run the command "sudo service ssh start" we are literally starting the SSH service in our system. But, based on my research, in many cases, if the SSH service starts successfully, we might not see any output like what is shown in the screenshot. So, it does not mean that we did not successfully execute the command and it shows us that we successfully verified the SSH service. The command "sudo systemctl status ssh" is also one of the ways to verify SSH service, since it displays the current status and information about the SSH service in our system.

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

#### 4.1 *sudo ufw allow ssh*

```

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

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

root@workstation:/home/kazuki# sudo ufw allow ssh
Rules updated
Rules updated (v6)
root@workstation:/home/kazuki#

```

#### 4.2 *sudo ufw enable*

```

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

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

root@workstation:/home/kazuki# sudo ufw enable
Firewall is active and enabled on system startup
root@workstation:/home/kazuki#

```

#### 4.3 *sudo ufw status*

```

kazuki@server1:~$ sudo ufw status
Status: active

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

kazuki@server2:~$ sudo ufw status
Status: active

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

root@workstation:/home/kazuki# sudo ufw status
Status: active

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

```

### Task 2.4: Firewall Configuration

The command "sudo ufw allow ssh" adds a rule to the ufw and it literally allows incoming SSH connection, this is important to enable remote access to our system using the SSH. The command "sudo ufw enable" literally enables the ufw, once this is enabled it starts enforcing the rules you've created, without this the rules you added in the first command will be nothing or it would not be in effect. And the command "sudo ufw status" just displays the current status of the ufw.

**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: 192.168.56.104

```
kazuki@server1:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.104 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::a00:27ff:fe59:4468 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:59:44:68 txqueuelen 1000 (Ethernet)
    RX packets 2 bytes 1180 (1.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 10 bytes 1334 (1.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

1.2 Server 2 IP address: 192.168.56.103

```
kazuki@server2:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.103 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::a00:27ff:fed3:a699 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:d3:a6:99 txqueuelen 1000 (Ethernet)
    RX packets 18 bytes 6181 (6.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 13 bytes 1544 (1.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

1.3 Server 3 IP address: 192.168.56.102

```
root@workstation:/home/kazuki# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.102 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::b1c8:1112:eb11:40c2 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:24:92:e6 txqueuelen 1000 (Ethernet)
    RX packets 64 bytes 12649 (12.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 75 bytes 9702 (9.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.101 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::4976:6481:f66a:eef8 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:49:46:45 txqueuelen 1000 (Ethernet)
    RX packets 63 bytes 11674 (11.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 77 bytes 9669 (9.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

### Task 3.1: Verifying Network

After running the command “ifconfig”, the IP Addresses were shown for Local Machine, Server 1, and Server 2. All IP addresses are within the designated network range 192.168.56.\_.

## 2. Make sure that they can ping each other.

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

```
64 bytes from 192.168.56.104: icmp_seq=310 ttl=64 time=0.859 ms
64 bytes from 192.168.56.104: icmp_seq=311 ttl=64 time=1.69 ms
64 bytes from 192.168.56.104: icmp_seq=312 ttl=64 time=1.20 ms
^C
--- 192.168.56.104 ping statistics ---
312 packets transmitted, 312 received, 0% packet loss, time 311682ms
rtt min/avg/max/mdev = 0.557/1.319/8.828/0.633 ms
```

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

```
64 bytes from 192.168.56.103: icmp_seq=411 ttl=64 time=1.125 ms
64 bytes from 192.168.56.103: icmp_seq=412 ttl=64 time=1.02 ms
64 bytes from 192.168.56.103: icmp_seq=413 ttl=64 time=1.18 ms
64 bytes from 192.168.56.103: icmp_seq=414 ttl=64 time=1.03 ms
64 bytes from 192.168.56.103: icmp_seq=415 ttl=64 time=1.39 ms
^C
--- 192.168.56.103 ping statistics ---
415 packets transmitted, 415 received, 0% packet loss, time 414822ms
rtt min/avg/max/mdev = 0.751/1.238/2.663/0.289 ms
```

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

```
64 bytes from 192.168.56.103: icmp_seq=324 ttl=64 time=1.04 ms
64 bytes from 192.168.56.103: icmp_seq=325 ttl=64 time=1.66 ms
64 bytes from 192.168.56.103: icmp_seq=326 ttl=64 time=1.22 ms
64 bytes from 192.168.56.103: icmp_seq=327 ttl=64 time=1.26 ms
^C
--- 192.168.56.103 ping statistics ---
327 packets transmitted, 327 received, 0% packet loss, time 332689ms
rtt min/avg/max/mdev = 0.619/1.330/7.832/0.474 ms
```

## Task 3.2: Pinging

Connectivity tests were executed in Local Machine, Server 1, and Server 2. Local Machine was able to ping both servers 1 and 2, and also server 1 was also able to ping server 2. Since all of the ping were successful. it indicates that network connectivity is working among all of them.

## 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`

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`

```
root@workstation:/home/kazuki# ssh kazuki@192.168.56.104
kazuki@192.168.56.104's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)

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

System information as of Wed Aug 23 01:24:54 PM UTC 2023

System load:  0.06689453125   Processes:    116
Usage of /:   46.0% of 11.21GB Users logged in:  1
Memory usage: 6%             IPv4 address for enp0s3: 192.168.56.104
Swap usage:   0%

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

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

Last login: Wed Aug 23 13:24:55 2023 from 192.168.56.104
kazuki@server1:~$
Logout
```



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

```
Last login: Wed Aug 23 13:24:33 2023
kazuki@server1:~$
logout
Connection to 192.168.56.104 closed.
root@workstation:/home/kazuki#
```

3. Do the same for Server 2.

```
root@workstation:/home/kazuki# ssh kazuki@192.168.56.103
The authenticity of host '192.168.56.103 (192.168.56.103)' can't be established.
ED25519 key fingerprint is SHA256:oBCMp24rQR1SDxZKMBYVS2IzIaOQV8ySPe6FvzBNWMU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.103' (ED25519) to the list of known hosts.
kazuki@192.168.56.103's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)

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

System information as of Wed Aug 23 02:50:32 PM UTC 2023

System load:  0.05126953125   Processes:            112
Usage of /:   44.4% of 11.21GB Users logged in:      1
Memory usage: 6%             IPv4 address for enp0s3: 192.168.56.103
Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

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

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

Last login: Wed Aug 23 12:00:34 2023
kazuki@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)

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
192.168.56.104 server1
192.168.56.103 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.

```
root@workstation:/home/kazuki# ssh kazuki@server1
The authenticity of host 'server1 (192.168.56.104)' can't be established.
ED25519 key fingerprint is SHA256:DVoAm58nArG701KAMCpqMVAhznA01S/CYmGdW9vWVs.
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.
kazuki@server1's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)

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

System information as of Wed Aug 23 01:24:54 PM UTC 2023

System load:  0.06689453125      Processes:           116
Usage of /:   46.0% of 11.21GB   Users logged in:     1
Memory usage: 6%                IPv4 address for enp0s3: 192.168.56.104
Swap usage:   0%

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

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

Last login: Wed Aug 23 13:12:47 2023 from 192.168.56.102
kazuki@server1:~$

root@workstation:/home/kazuki# ssh kazuki@server2
The authenticity of host 'server2 (192.168.56.103)' can't be established.
ED25519 key fingerprint is SHA256:oBCMp24rQR1SDxZKMBYV52IzIaOQV8ySPe6FvzBNWMU.
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.
kazuki@server2's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)

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

System information as of Wed Aug 23 02:57:56 PM UTC 2023

System load:  0.0                Processes:           112
Usage of /:   44.4% of 11.21GB   Users logged in:     1
Memory usage: 6%                IPv4 address for enp0s3: 192.168.56.103
Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

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

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

Last login: Wed Aug 23 14:50:34 2023 from 192.168.56.102
kazuki@server2:~$
```

### Task 4.1-5:Verifying SSH Connectivity

I was able to verify SSH connectivity by using the command "ssh username@ip address of servers 1 and 2" on the Local machine. To make this easier, we just need to edit the "/etc/hosts" file to add the IP addresses of servers 1 and 2. And then we can change the command "ssh username@ip address of servers 1 and 2" to "ssh username@server1/server2".

## Reflections:

Answer the following:

1. How are we able to use the hostname instead of IP address in SSH commands?
  - By using the command “`sudo nano /etc/hosts`” we can edit the file to add the IP addresses of the servers 1 and 2 and their corresponding hostnames. Once we are done editing the file, we can now use their hostnames in SSH commands.
2. How secure is SSH?
  - For me, I think SSH is highly secure due to its authentication methods. Also, it asks us if we want to continue (answerable by yes or no) and then asks for a password to continue with some commands, which makes SSH really secure. But we should not be overconfident with this, we must practice and maintain its security measures, like using strong passwords. Additionally, it uses encryption to secure the connection between a client and a server.