Sri Lanka Institute of Information Technology

Assignment 1- Year 2 semester 1

Student Name – Wanasinghe N.K

Student ID – IT23221000

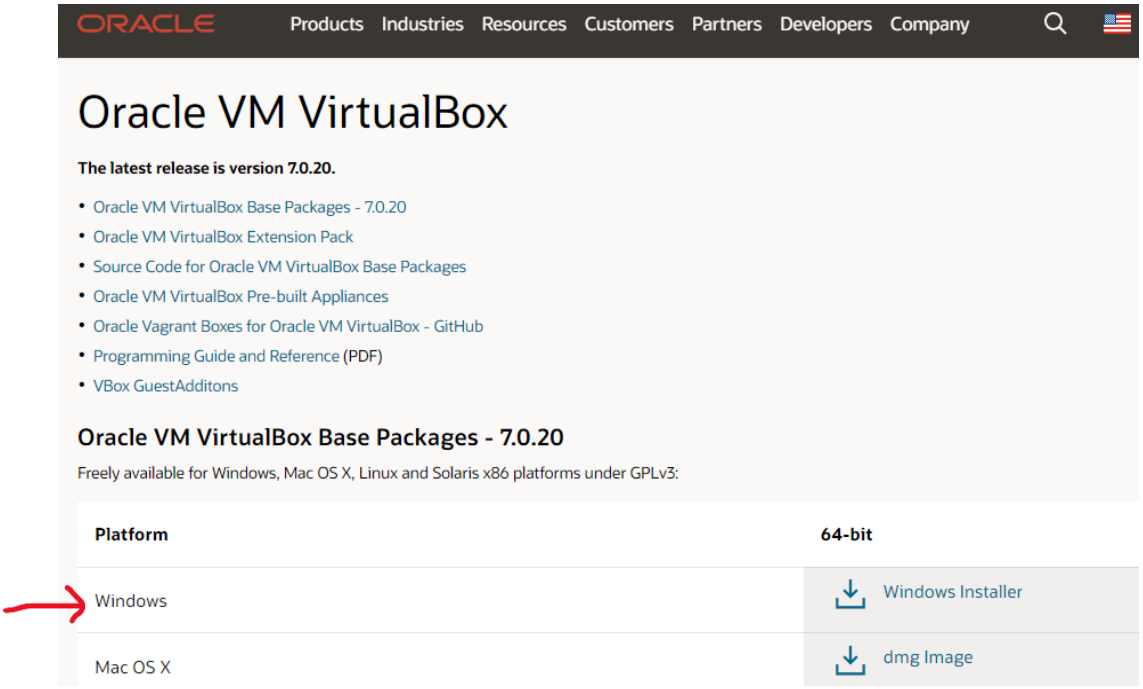
IE2012 - Systems and Network Programming

B.Sc. (Hons) in information Technology Specializing in Cyber Security

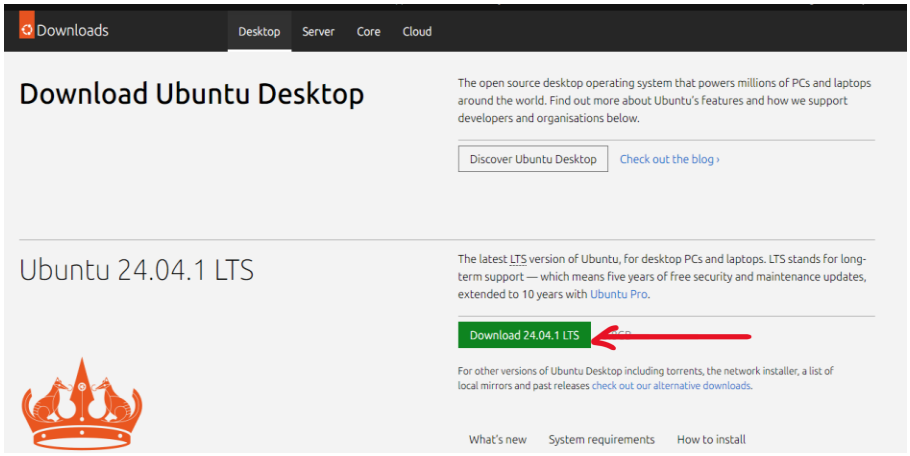
1. Basics of Linux Environments.

1. Virtual Machine Setup

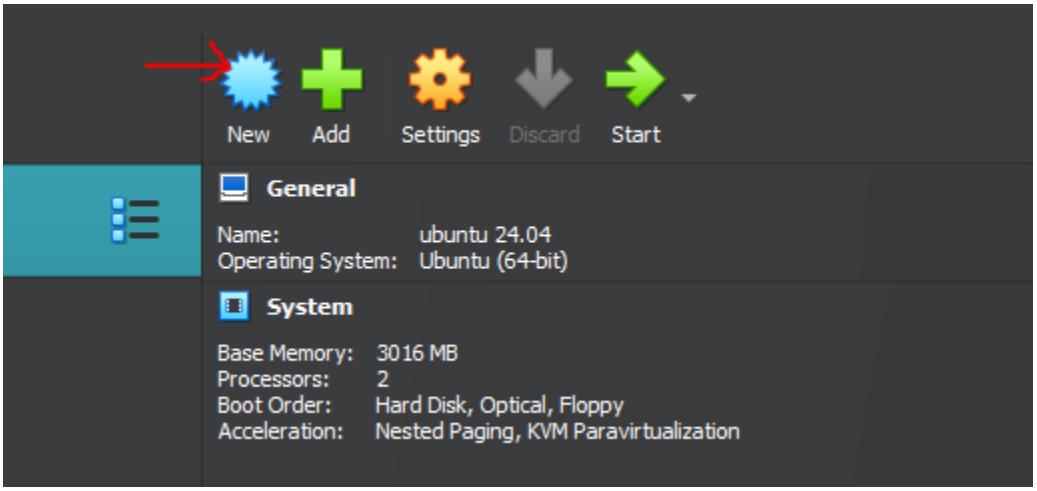
1. Initially, to setup the Virtual Machine, I have downloaded and installed Oracle VirtualBox.



2. Next, I downloaded Ubuntu Distribution (the latest version).



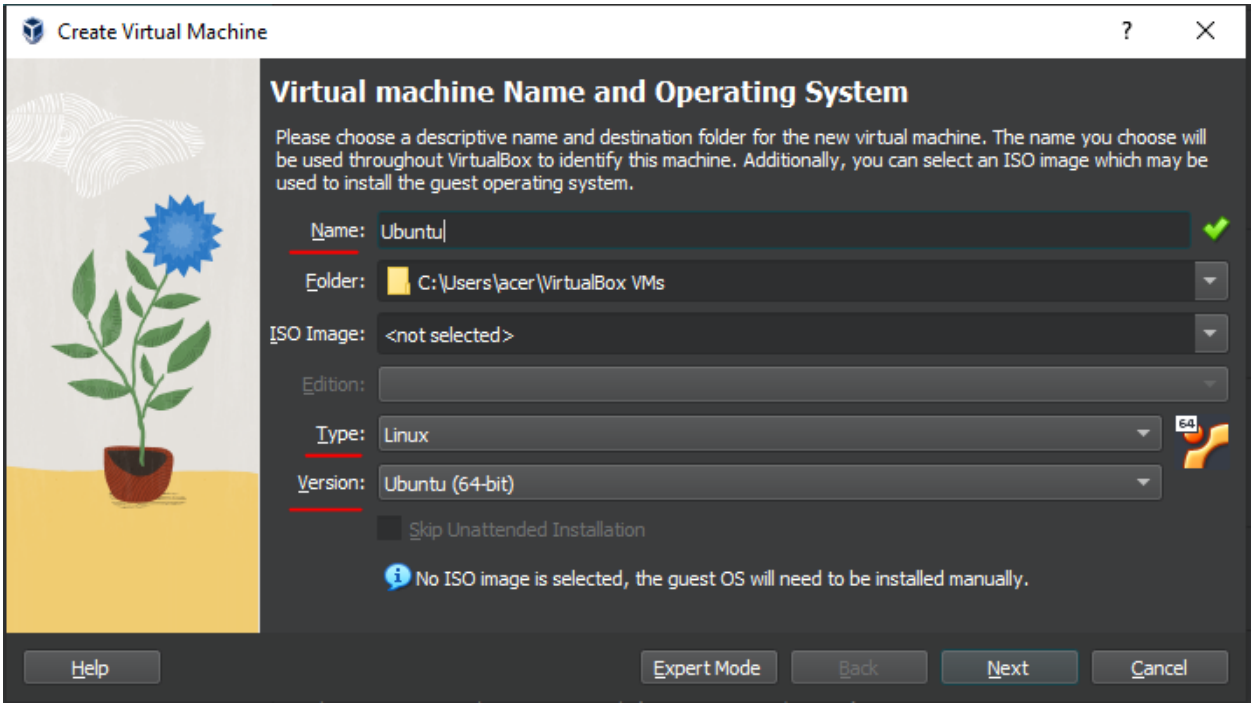
3. Open VirtualBox and click on "New" to create a new virtual machine.



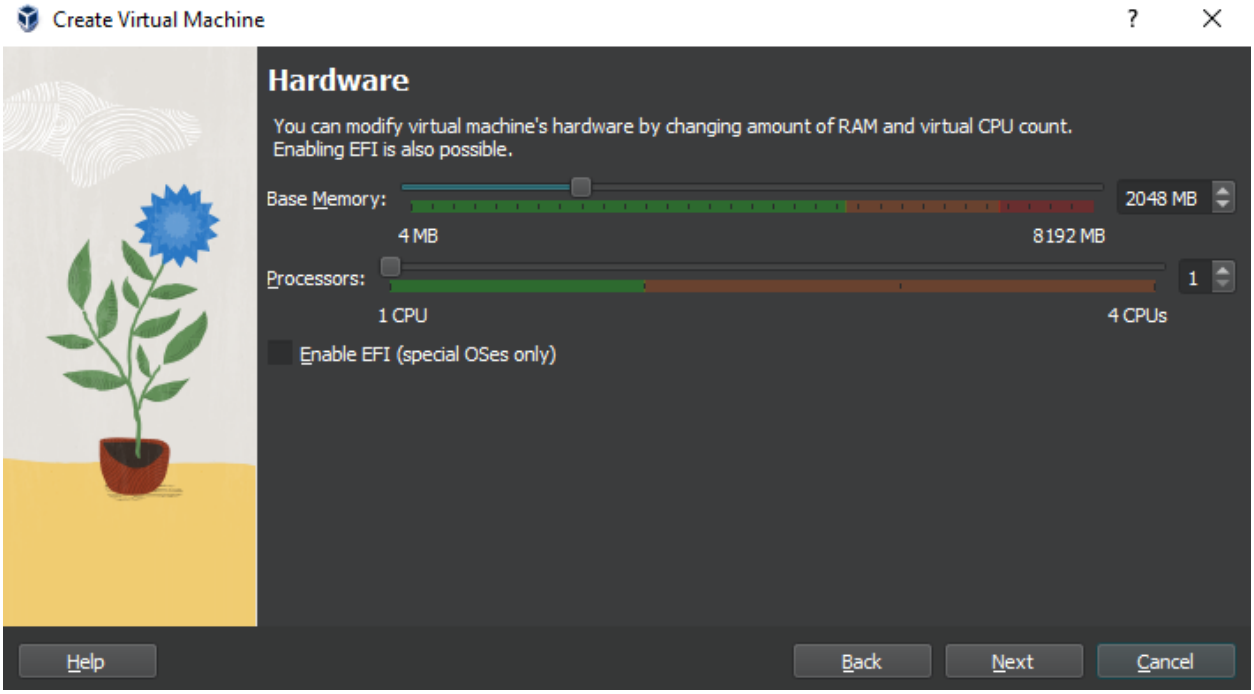
4. Type the Virtual Machine name as “Ubuntu” and the destination folder for the new Virtual Machine.

5. Select the type as “Linux’.

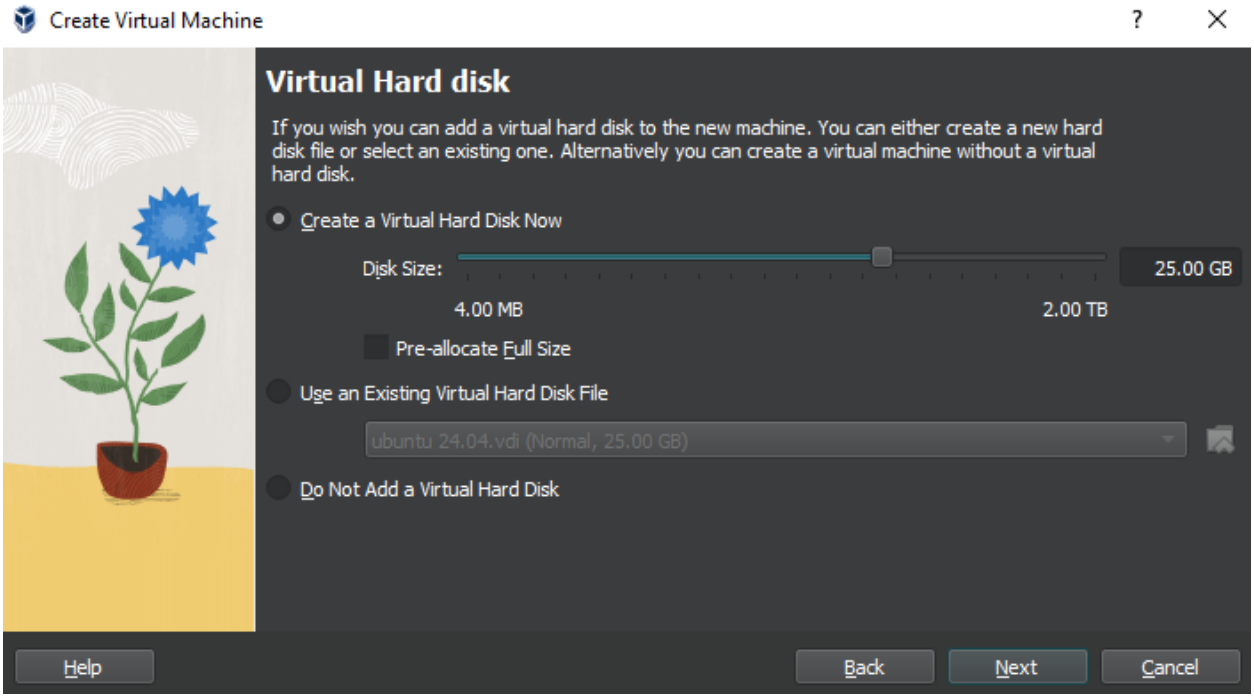
6. Select the version” Ubuntu (64-bit)”



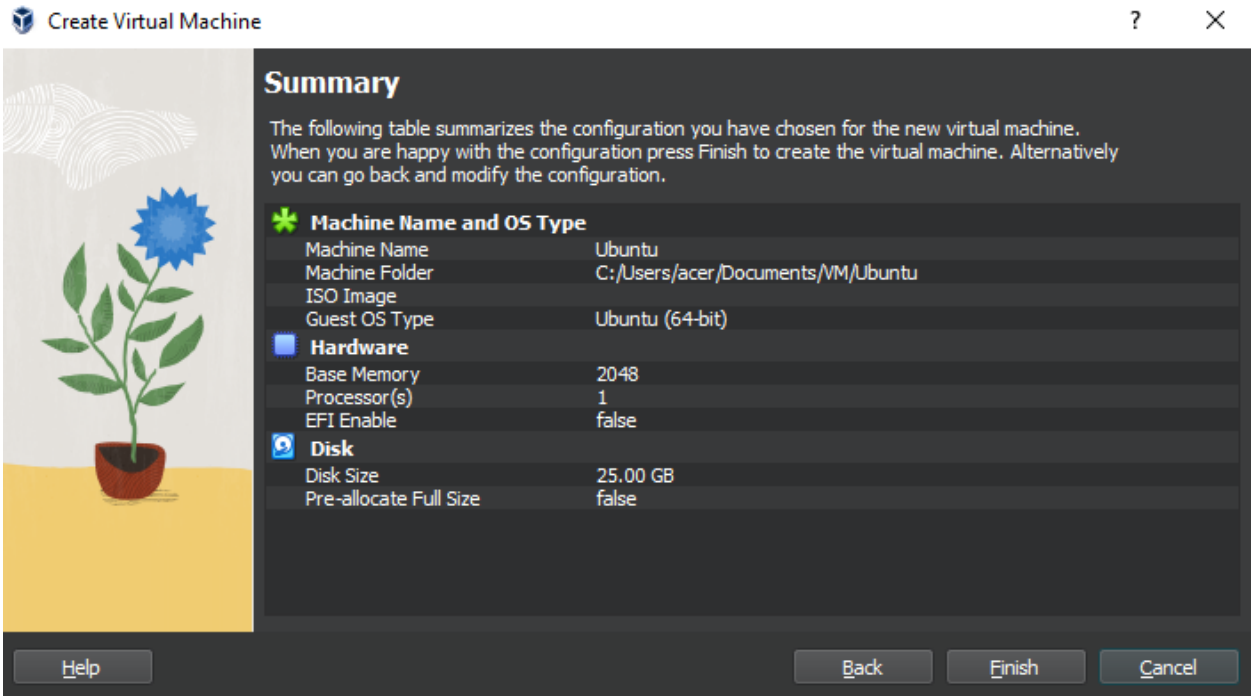
7. Choose the amount of RAM and select how many CPUs should be allocated. click "Next."



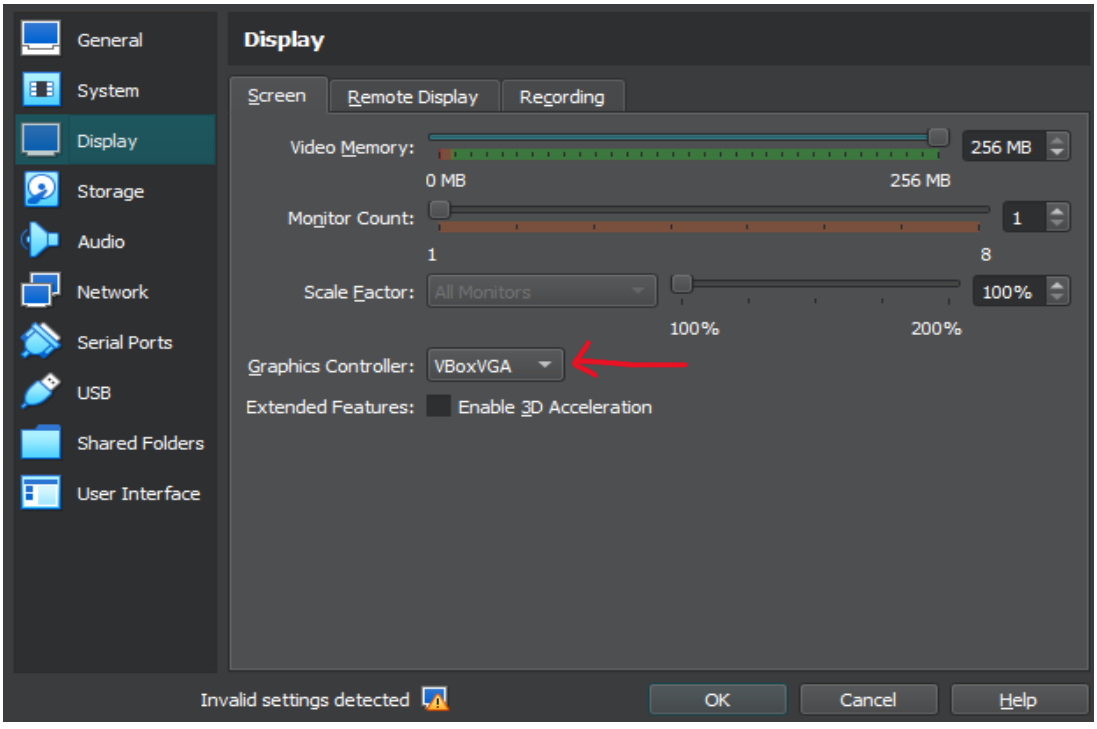
8. Choose "Create a virtual hard disk now" and set the size of the virtual hard disk (at least 20 GB) and click "Next."



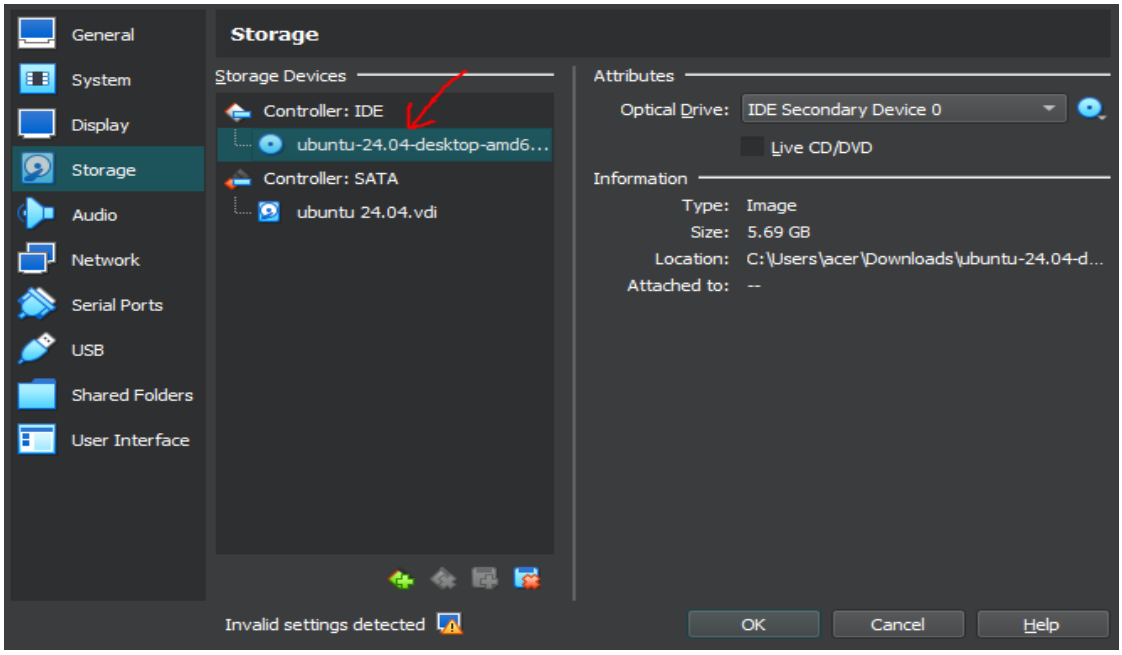
9. After checking the summary of the newly created virtual machine, click “Finish.”



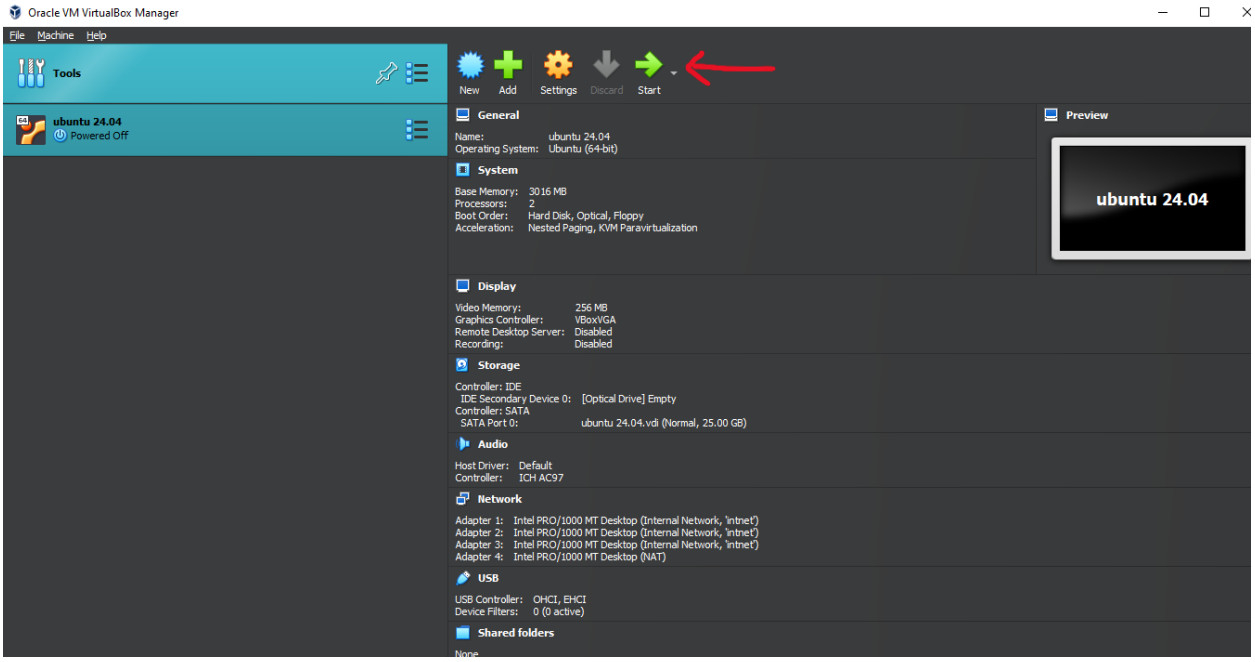
10. Go to **Settings < Display** and set video memory to 256MB and Graphic Controller to VBoxVGA.

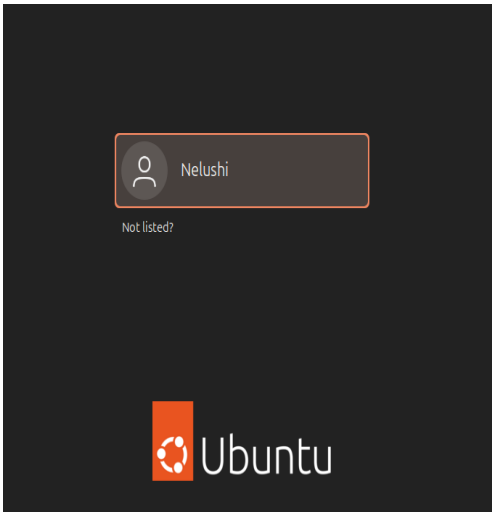


11. Go to **Storage.** Under "Controller: IDE," click on the empty disk icon.
    On the right, click the disk icon and select "Choose a disk file."
    Navigate to the ISO file you downloaded and select it.
    Click "OK" to save the settings.



12. Finally, select the virtual machine name and click "Start.".





2. Command Line Introduction.

1. Navigation Commands.

- **cd (change directory):** Allows to navigate between directories.

  cd . stays in the current directory.

  cd .. takes up one directory level.

  cd - switches to previous directory

  cd / cd ~ / cd $HOME goes to the home directory.

```
nelushi@nelushi-VirtualBox:~$ pwd
/home/nelushi
nelushi@nelushi-VirtualBox:~$ cd ..
nelushi@nelushi-VirtualBox:/home$ pwd
/home
nelushi@nelushi-VirtualBox:/home$ cd .
nelushi@nelushi-VirtualBox:/home$ pwd
/home
nelushi@nelushi-VirtualBox:/home$ cd $HOME
nelushi@nelushi-VirtualBox:~$ pwd
/home/nelushi
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ pwd
/home/nelushi/Desktop
nelushi@nelushi-VirtualBox:~/Desktop$ cd GDB
nelushi@nelushi-VirtualBox:~/Desktop/GDB$ pwd
/home/nelushi/Desktop/GDB
nelushi@nelushi-VirtualBox:~/Desktop/GDB$ cd -
/home/nelushi/Desktop
```

- **ls:** Lists the names of files and directories in the current directory (without hidden files).

  ls -l lists the file content in long format like file type, file size, permissions, number of links, owner etc.

  ls -a lists all the files including hidden files (files that start with a dot.).

  ls -al combines -a (show all files, including hidden) and -l (long format) options.
  It provides a detailed listing of all files, including hidden ones**.**

```
nelushi@nelushi-VirtualBox:~$ ls
Desktop  Documents  Downloads  IT23221000  Music  Pictures  Public  snap  sos  so
nelushi@nelushi-VirtualBox:~$ ls -a
.              .bashrc  Documents   .lesshst  .profile  soslab02
..             .cache   Downloads   .local    Public    .ssh
.bash_history  .config  IT23221000  Music     snap      student
.bash_logout   Desktop  .lesshsQ    Pictures  sos       .sudo_as_admin_successful
nelushi@nelushi-VirtualBox:~$ ls -l
total 52
drwxr-xr-x 5 nelushi nelushi 4096 Sep  5 00:25 Desktop
drwxr-xr-x 2 nelushi nelushi 4096 Jul 20 14:45 Documents
drwxr-xr-x 2 nelushi nelushi 4096 Jul 20 14:45 Downloads
drwxrwxr-x 2 nelushi nelushi 4096 Jul 23 15:23 IT23221000
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ ls -al
total 40
drwxr-xr-x  5 nelushi nelushi  4096 Sep 26 09:08  .
drwxr-x--- 20 nelushi nelushi  4096 Sep 26 09:21  ..
-rwxrwxr-x  1 nelushi nelushi 16032 Sep  5 00:25  ex
-rw-rw-r--  1 nelushi nelushi   243 Sep  5 00:25  ex.c
drwxrwxr-x  2 nelushi nelushi  4096 Sep 26 10:15  GDB
drwxrwxr-x  3 nelushi nelushi  4096 Aug 22 10:29 'shell scripting'
drwxrwxr-x  2 nelushi nelushi  4096 Sep 22 16:57  snpAssignment
```

- **pwd (Print Working Directory):** Displays the full path of the current working directory.

```
nelushi@nelushi-VirtualBox:~/Desktop$ pwd
/home/nelushi/Desktop
```

- **mkdir (Make directory)**: Creates a new directory.
  rmdir (Remove directory): Removes a new directory.

```
nelushi@nelushi-VirtualBox:~$ mkdir snp
nelushi@nelushi-VirtualBox:~$ cd snp
nelushi@nelushi-VirtualBox:~/snp$
```

```
nelushi@nelushi-VirtualBox:~$ rmdir snp
nelushi@nelushi-VirtualBox:~$ cd snp
bash: cd: snp: No such file or directory
```

2. File Manipulation Commands.

- **cp**: Copy a file or a directory from one location to another.

  Copy a file to another file – **cp text1.txt text2.txt**

```
#text2.txt
IT number - IT23221000
~
```

```
#text2.txt
IT number - IT23221000
~
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ cp text1.txt text2.txt
nelushi@nelushi-VirtualBox:~/Desktop$ cat text2.txt
#text1.txt
This is snp assignment
```

Copy a file to a directory – **cp text1.txt snp (directory name)**

```
nelushi@nelushi-VirtualBox:~$ mkdir snp
nelushi@nelushi-VirtualBox:~$ touch text
nelushi@nelushi-VirtualBox:~$ cp text snp
nelushi@nelushi-VirtualBox:~$ cd snp
nelushi@nelushi-VirtualBox:~/snp$ ls
text
```

Copy a directory to another directory – **cp -r snp1 snp2**

```
nelushi@nelushi-VirtualBox:~/Desktop$ mkdir snp1
nelushi@nelushi-VirtualBox:~/Desktop$ mkdir snp2
nelushi@nelushi-VirtualBox:~/Desktop$ cp -r snp1 snp2
nelushi@nelushi-VirtualBox:~/Desktop$ cd snp2
nelushi@nelushi-VirtualBox:~/Desktop/snp2$ ls
snp1
```

- **mv**: move a file and rename a file.

```
nelushi@nelushi-VirtualBox:~$ mkdir snp2
nelushi@nelushi-VirtualBox:~$ mv text snp2
nelushi@nelushi-VirtualBox:~$ cd snp2
nelushi@nelushi-VirtualBox:~/snp2$ ls
text
nelushi@nelushi-VirtualBox:~/snp2$ mv text text1
nelushi@nelushi-VirtualBox:~/snp2$ ls
text1
```

- **cat**: concatenate and displays a content of a file.

  cat < text1.txt – displays and concatenate a file
  cat > text1.txt – overwrites existing information**.**
  cat >> text1.txt – appends whatever we type to the end of text1.txt instead of overwriting it.

```
nelushi@nelushi-VirtualBox:~$ cat < test.txt
Nelushi Wanasinghe
IT23221000
Sri Lanka Institute of Information Technology
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ cat < text1.txt
#text1.txt
This is snp assignment
nelushi@nelushi-VirtualBox:~/Desktop$ cat > text1.txt
hello world
^C
nelushi@nelushi-VirtualBox:~/Desktop$ cat text1.txt
hello world
```

- **touch**: creates an empty file.

```
nelushi@nelushi-VirtualBox:~$ touch test2.txt
nelushi@nelushi-VirtualBox:~$ ls
checkpermission  Documents  IT23221000  Pictures  snap  soslab02  Templates  test.txt
Desktop          Downloads  Music       Public    sos   student    test2.txt  Videos
```

- **head**: prints the first specified number of lines of a file.
  tail: prints the last specified number of lines of a file.

```
nelushi@nelushi-VirtualBox:~$ head -2 test.txt
Nelushi Wanasinghe
IT23221000
nelushi@nelushi-VirtualBox:~$ tail -2 test.txt
IT23221000
Sri Lanka Institute of Information Technology
```

3. System Information and User Management commands.

1. **uname -a:** displays details like kernel name, version, and the system architecture.

```
nelushi@nelushi-VirtualBox:~$ uname -a
Linux nelushi-VirtualBox 6.8.0-39-generic #39-Ubuntu SMP PREEMPT_DYNAMIC Fri Jul
  5 21:49:14 UTC 2024 x86_64 x86_64 x86_64 GNU/Linux
```

2. **cat /proc/version:** displays the kernel version and some additional details about GNU compiler and Linux distribution.

```
nelushi@nelushi-VirtualBox:~$ cat /proc/version
Linux version 6.8.0-39-generic (buildd@lcy02-amd64-112) (x86_64-linux-gnu-gcc-13
 (Ubuntu 13.2.0-23ubuntu4) 13.2.0, GNU ld (GNU Binutils for Ubuntu) 2.42) #39-Ub
untu SMP PREEMPT_DYNAMIC Fri Jul  5 21:49:14 UTC 2024
```

3. **df -h:** displays the available and used disk space on all mounted filesystems in a human-readable format.

```
nelushi@nelushi-VirtualBox:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           292M  1.6M  291M   1% /run
/dev/sda2        25G   12G   12G  49% /
tmpfs           1.5G     0  1.5G   0% /dev/shm
tmpfs           5.0M  8.0K  5.0M   1% /run/lock
tmpfs           292M  128K  292M   1% /run/user/1000
```

4. **free -m:** displays the total, used, and available memory in MB.

```
nelushi@nelushi-VirtualBox:~$ free -m
               total        used        free      shared  buff/cache   available
Mem:            2916        1109         709          33        1295        1807
Swap:           2915           0        2915
```

5. **id**: displays the user ID, group ID, and groups associated with a user.

```
nelushi@nelushi-VirtualBox:~$ id
uid=1000(nelushi) gid=1000(nelushi) groups=1000(nelushi),4(adm),24(cdrom),27(sud
o),30(dip),46(plugdev),100(users),114(lpadmin)
```

6. **whoami**: displays the username of the current logged-in user.

```
nelushi@nelushi-VirtualBox:~$ whoami
nelushi
```

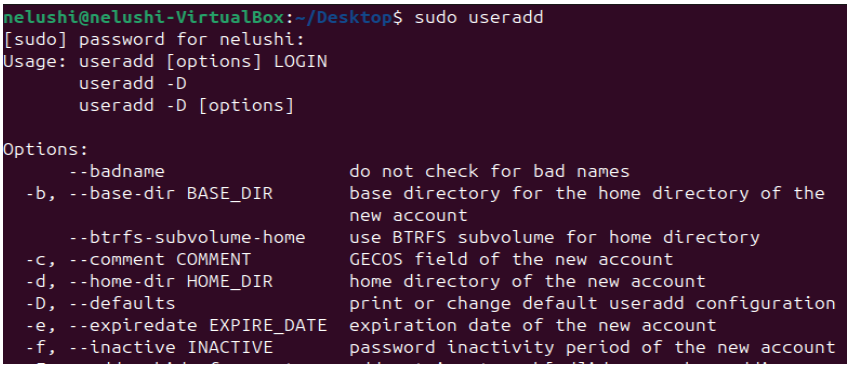
7. **who**: shows information about the users who are currently logged into the system.

```
nelushi@nelushi-VirtualBox:~$ who
nelushi  seat0        2024-09-22 04:04 (login screen)
nelushi  tty2         2024-09-22 04:04 (tty2)
```

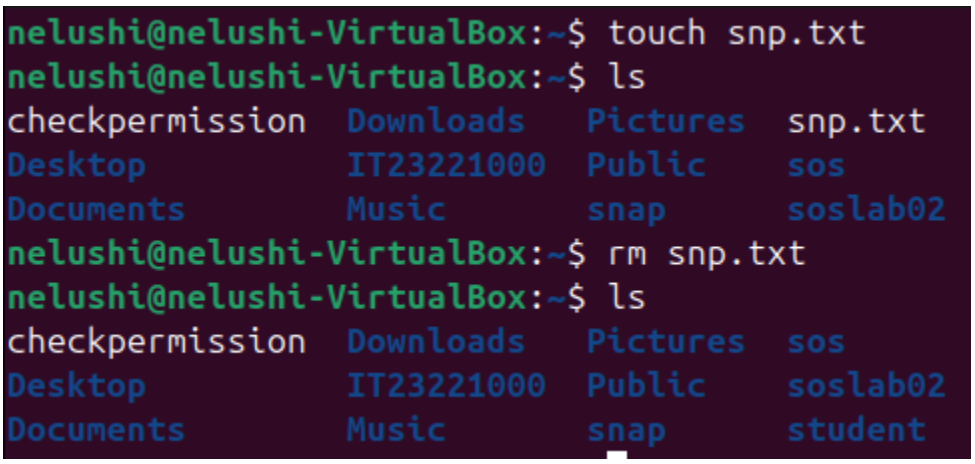
8. **passwd**: changes the password of the current user (if another specified user).

```
nelushi@nelushi-VirtualBox:~/Desktop$ passwd
Changing password for nelushi.
Current password:
New password:
```

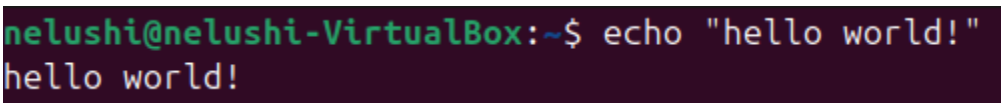
9. **sudo useradd**: creates a new user account.



10. **rm**: deletes a file or a folder that is not empty.



11. **echo**: prints the provided test or variable to the terminal.



12. **chmod**: modifies the file permissions of a file or a directory.

chmod a+rwx text1.txt – gives read, wirte and execute permission for all.

chmod u=rwx,g=rw,o=rx text1.txt – gives read, write and execute permissions for users, only read,write permissions to groups and read,execute permissions for others.

chmod u-x text1.txt – remove execute permission of users.

chmod 755 text1.txt – gives read, write, execute permission for users, read,execute permissions for groups amd others.

```
nelushi@nelushi-VirtualBox:~/Desktop$ ls -l text1.txt
-rw-rw-r-- 1 nelushi nelushi 12 Oct  1 20:31 text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ chmod a+rwx text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ ls -l text1.txt
-rwxrwxrwx 1 nelushi nelushi 12 Oct  1 20:31 text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ chmod u=rwx,g=rx,o=rw text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ ls -l text1.txt
-rwxr-xrw- 1 nelushi nelushi 12 Oct  1 20:31 text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ chmod u-w text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ ls -l text1.txt
-r-xr-xrw- 1 nelushi nelushi 12 Oct  1 20:31 text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ chmod 755 text1.txt
nelushi@nelushi-VirtualBox:~/Desktop$ ls -l text1.txt
-rwxr-xr-x 1 nelushi nelushi 12 Oct  1 20:31 text1.txt
```

13. **man**: displays the manual for a specified command, showing a detailed description on how to use it and available options.

```
SUDO(8)                    System Manager's Manual                    SUDO(8)

NAME
       sudo, sudoedit — execute a command as another user

SYNOPSIS
       sudo -h | -K | -k | -V
       sudo -v [-ABkNnS] [-g group] [-h host] [-p prompt] [-u user]
       sudo  -l [-ABkNnS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
            [command [arg ...]]
       sudo  [-ABbEHnPS]  [-C  num]  [-D  directory]  [-g  group]  [-h  host]
            [-p  prompt]  [-R  directory]  [-r  role]  [-t  type] [-T timeout]
            [-u user] [VAR=value] [-i | -s] [command [arg ...]]
       sudoedit [-ABkNnS]  [-C  num]  [-D  directory]  [-g  group]  [-h  host]
            [-p  prompt]  [-R  directory]  [-r  role]  [-t  type] [-T timeout]
            [-u user] file ...

DESCRIPTION
       sudo allows a permitted user to execute a command as the  superuser  or
       another user, as specified by the security policy.  The invoking user's
       real  (not  effective)  user-ID is used to determine the user name with
       which to query the security policy.

       sudo supports a plugin architecture for  security  policies,  auditing,
       and  input/output  logging.   Third  parties can develop and distribute
 Manual page sudo(8) line 1 (press h for help or q to quit)
```

14. **whatis**: provides a brief description of a command or system call, summarizing what the command does.

```
nelushi@nelushi-VirtualBox:~$ whatis sudo
sudo (8)             - execute a command as another user
```

15. **top**: displays all active processes in real time.

```
top - 16:36:22 up  1:03,  1 user,  load average: 0.00, 0.00, 0.03
Tasks: 193 total,   1 running, 192 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.0 us,  0.8 sy,  0.0 ni, 97.8 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0 st
MiB Mem :   2916.9 total,    715.5 free,   1096.0 used,   1302.3 buff/cache
MiB Swap:   2916.0 total,   2916.0 free,      0.0 used.   1820.9 avail Mem
```

2. DHCP, DNS and NTP Services

1. DHCP (Dynamic Host Configuration Protocol)

What is DHCP?

Dynamic Host Configuration Protocol (DHCP) is a network management protocol that automate the process of assigning IP addresses and related network configuration information to devices on a network. DHCP dynamically assigns IP addresses from a predefined range whenever a device connects to the network, instead of manual IP address configuration.

Key Points

- **Automatic IP management** – When devices join and leave the network, DHCP automatically manages the assignment and reassignment of IP addresses.
- **DHCP scope** – A DHCP server assigns IP addresses to computers on a network from its scope. It is a range of IP addresses that a DHCP server can hand out.
- **DHCP lease** – Lease is the amount of time an IP address is assigned to a computer (Eg:- One day, one week, one month) to help make sure the DHCP server does not run out of IP addresses and its scope.
- **Configuration Provided:** IP address, subnet mask, default gateway, and DNS servers.
- **Benefits:** Reduces manual configuration errors such as IP conflicts, centralizes management, and scales easily with network growth.

Configuration Steps for DHCP

1. First, open the terminal in Ubuntu and update package list by executing the **sudo apt update** command.

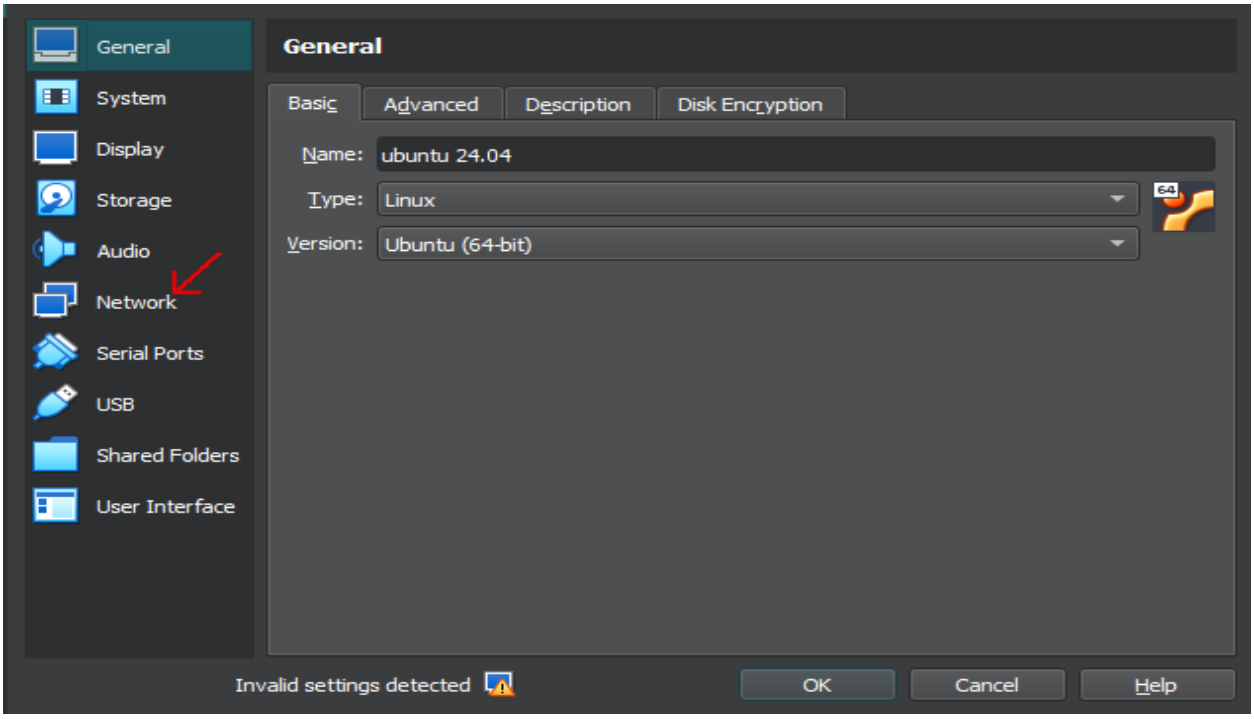
2. Now, Install the DHCP Server using **sudo apt install isc-dhcp-server** command.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo apt install isc-dhcp-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
isc-dhcp-server is already the newest version (4.4.3-P1-4ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 247 not upgraded.
nelushi@nelushi-VirtualBox:~/Desktop$
```

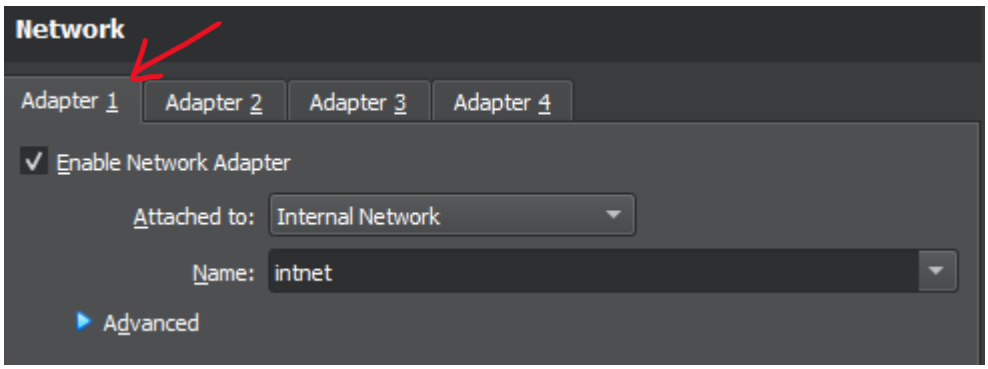
3. Now, let's create Virtual Network Interfaces for two clients.

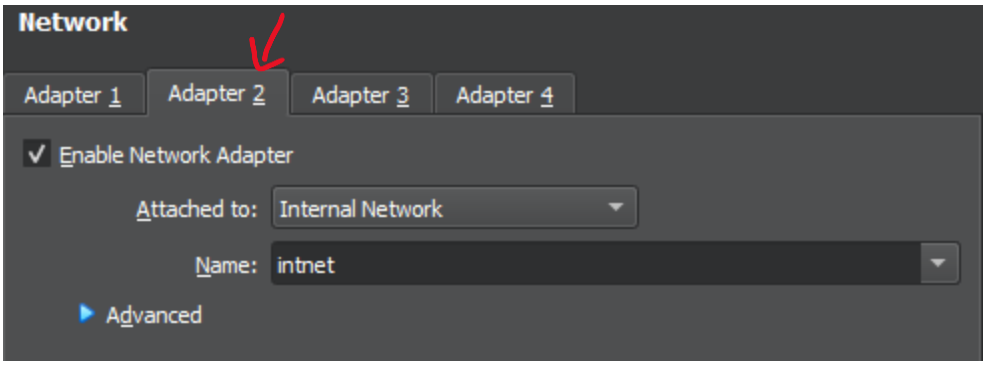
   In VirtualBox, we can configure multiple network adapters to act as clients.

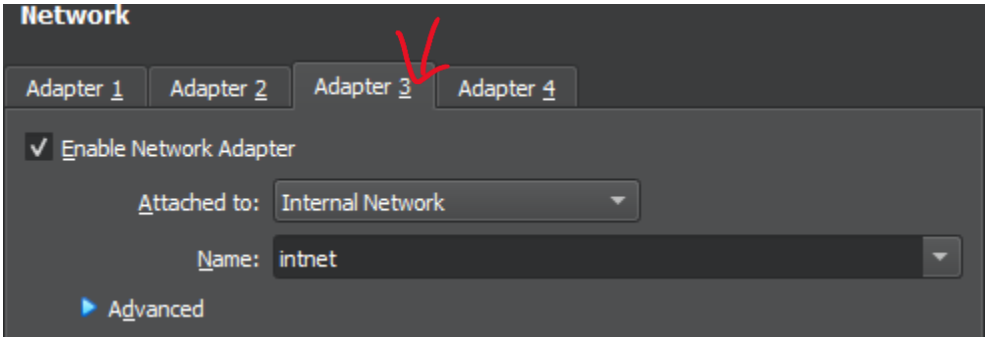
   Go to **Settings > Network** in the virtual machine.



Set **Adapter 2 and 3** to **Internal Network** (same network name used for Adapter 1/DHCP server, e.g., intnet).







4. Check the network interfaces using the commands **ip a, ip addr, or ifconfig**.

- **ip a**: Lists all network interfaces and their details. This is the modern recommended command.
- **ip addr**: Displays IP addresses and network interfaces, similar to ip a.
- **ifconfig**: Older command to display network interfaces and their configurations. Need to install it using **sudo apt install net-tools.**

```
nelushi@nelushi-VirtualBox:~/Desktop$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 08:00:27:20:89:65 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:c4:af:70 brd ff:ff:ff:ff:ff:ff
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:c9:1e:19 brd ff:ff:ff:ff:ff:ff
nelushi@nelushi-VirtualBox:~/Desktop$
```

In the output, you can see the Loopback Interface as lo. It has an IP address of 127.0.0.1/8.

Other interfaces such as enp0s3, enp0S8 and enp0s9 represent the network connections (wired or wireless). **enp0s3 is where I have decided to run the DHCP server and enp0s8, enp0s9 are the clients I have created**.

5. Check the IP address and the range of the enp0s3 Interface (DHCP Server).

If the interface does not show an IP address, we can manually assign a static IP address using the following commands.

sudo ip addr add 192.168.100.1/24 dev enp0s3

sudo ip link set enp0s3 up

ip addr show enp0s3 – To verify the IP assignment

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ip addr add 192.168.100.1/24 dev enp0s3
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ip link set enp0s3 up
nelushi@nelushi-VirtualBox:~/Desktop$ ip addr show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:20:89:65 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.1/24 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe20:8965/64 scope link
       valid_lft forever preferred_lft forever
nelushi@nelushi-VirtualBox:~/Desktop$
```

6. Next, specify the network interface for DHCP Server by editing the DHCP server default configuration file to set the network interface as **enp0s3** for **INTERFACESv4**.

sudo nano /etc/default/isc-dhcp-server

```
# Additional options to start dhcpd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="enp0s3"
INTERFACESv6=""
```

Press ctrl+o to save, enter and ctrl+x to exit from the file.

7. Next, configure the subnet and range by adding or modifying the subnet configuration for an internal network.

sudo nano /etc/dhcp/dhcpd.conf

```
subnet 192.168.100.0 netmask 255.255.255.0 {
        range 192.168.100.10 192.168.100.100;
        option subnet-mask 255.255.255.0;
        option routers 192.168.100.1;
        default-lease-time 600;
        max-lease-time 7200;
}

option domain-name-servers 8.8.8.8, 8.8.4.4;
```

8. To configure the DHCP Server, uncomment the **authoritative** line.

```
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
authoritative;
```

Press ctrl+o to save, enter and ctrl+x to exit from the file.

9. Next, we can start the DHCP Server and Verify.

Start DHCP service**: sudo systemctl start isc-dhcp-server**

Enable DHCP service at boot**: sudo systemctl enable isc-dhcp-server**

Verify service status**: sudo systemctl status isc-dhcp-server**

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl start isc-dhcp-server
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl enable isc-dhcp-server
Synchronizing state of isc-dhcp-server.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable isc-dhcp-server
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl status isc-dhcp-server
● isc-dhcp-server.service - ISC DHCP IPv4 server
     Loaded: loaded (/usr/lib/systemd/system/isc-dhcp-server.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-09-16 04:24:34 +0530; 1min 15s ago
       Docs: man:dhcpd(8)
   Main PID: 2998 (dhcpd)
      Tasks: 1 (limit: 3416)
     Memory: 3.8M (peak: 4.0M)
        CPU: 9ms
     CGroup: /system.slice/isc-dhcp-server.service
             └─2998 dhcpd -user dhcpd -group dhcpd -f -4 -pf /run/dhcp-server/dhcpd.pid -cf /etc/dhcp/dhcpd.conf enp0s3

Sep 16 04:24:34 nelushi-VirtualBox dhcpd[2998]: PID file: /run/dhcp-server/dhcpd.pid
Sep 16 04:24:34 nelushi-VirtualBox dhcpd[2998]: Wrote 2 leases to leases file.
Sep 16 04:24:34 nelushi-VirtualBox sh[2998]: Wrote 2 leases to leases file.
Sep 16 04:24:34 nelushi-VirtualBox dhcpd[2998]: Listening on LPF/enp0s3/08:00:27:20:89:65/192.168.100.0/24
Sep 16 04:24:34 nelushi-VirtualBox sh[2998]: Listening on LPF/enp0s3/08:00:27:20:89:65/192.168.100.0/24
Sep 16 04:24:34 nelushi-VirtualBox sh[2998]: Sending on   LPF/enp0s3/08:00:27:20:89:65/192.168.100.0/24
Sep 16 04:24:34 nelushi-VirtualBox sh[2998]: Sending on   Socket/fallback/fallback-net
Sep 16 04:24:34 nelushi-VirtualBox dhcpd[2998]: Sending on   LPF/enp0s3/08:00:27:20:89:65/192.168.100.0/24
Sep 16 04:24:34 nelushi-VirtualBox dhcpd[2998]: Sending on   Socket/fallback/fallback-net
Sep 16 04:24:34 nelushi-VirtualBox dhcpd[2998]: Server starting service.
```

10. Now, we can request IP addresses for our clients from the DHCP Server by,

For client 1 - **sudo dhclient enp0s8**

ip addr show enp0s8 – To see the assigned IP address from the given range.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo dhclient enp0s8
Setting LLMNR support level "yes" for "3", but the global support level is "no".
nelushi@nelushi-VirtualBox:~/Desktop$ ip addr show enp0s8
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:c4:af:70 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.12/24 brd 192.168.100.255 scope global dynamic enp0s8
       valid_lft 584sec preferred_lft 584sec
    inet6 fe80::ae60:5d2f:a5c:55aa/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

DHCP Server has assigned the IP address 192.168.100.12 from the specified range (192.168.100.10 – 192.168.100.100).

For client 2 - **sudo dhclient enp0s9**

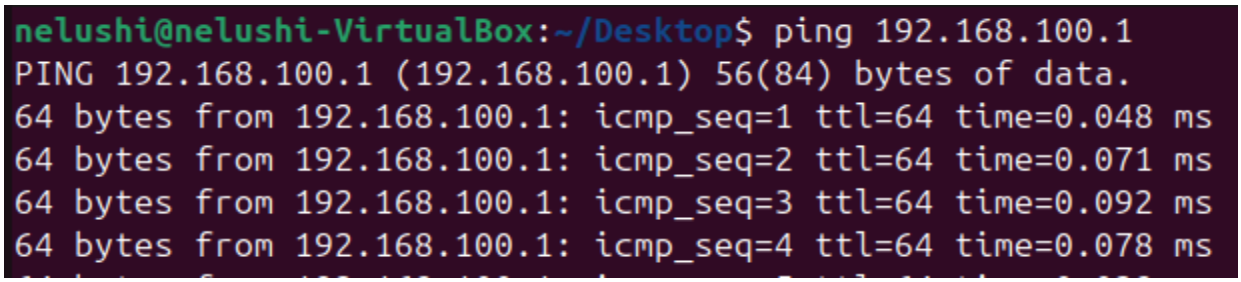
ip addr show enp0s9

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo dhclient enp0s9
Setting LLMNR support level "yes" for "4", but the global support level is "no".
nelushi@nelushi-VirtualBox:~/Desktop$ ip addr show enp0s9
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:c9:1e:19 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.13/24 brd 192.168.100.255 scope global dynamic enp0s9
       valid_lft 582sec preferred_lft 582sec
    inet6 fe80::a425:6f8b:eab9:d42e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

DHCP Server has assigned the IP address 192.168.100.13 from the specified range (192.168.100.10 – 192.168.100.100).

11. Additionally, we can test the network connectivity by,

    Ping DHCP server from client**: ping 192.168.100.1**



    The output shows responses like “64 bytes from 192.168.100.1: icmp_seq=1 ttl=64 time=0.048 ms”, it means Adapter 2 and 3 is successfully communicating with Adapter 1 (DHCP Server).

2. DNS (Domain Name System)

Role of DNS

Domain Name Server translates human-friendly domain names (like www.google.com) into IP addresses (like 93.184.216.34) that computers use to communicate with each other over the network.

How DNS Works

- **Query:** when we type a domain name into our browser, the computer asks a DNS server for the IP address of the specified domain name.
- **Resolution:** The DNS server finds the IP address by checking its records or asking other servers.
- **Response:** The IP address is sent back to your computer by the DNS Server, which connects to the website.

Configuration steps for DNS

1. open the terminal in Ubuntu and update package list by executing the **sudo apt update** command.

2. Now, Install the DNS Server using **sudo apt install bind9** command.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo apt install bind9
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
bind9 is already the newest version (1:9.18.28-0ubuntu0.24.04.1).
0 upgraded, 0 newly installed, 0 to remove and 251 not upgraded.
nelushi@nelushi-VirtualBox:~/Desktop$
```

3. Make sure that you have a NAT adapter to connect to the internet.

4. Next, open the primary configuration file for BIND.

sudo nano /etc/bind/named.conf.options

Uncomment and modify the **forwarders** section to use Google DNS.

```
forwarders {
        8.8.8.8;
};
```

The forwarders section is where we specify external DNS servers that our DNS server will query if it cannot resolve a domain name on its own.

5. Next, restart the BIND service using **sudo systemctl restart bind9**.

6. Check the status using **sudo systemctl status bind9**.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl status bind9
● named.service - BIND Domain Name Server
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset: enabled)
     Active: active (running) since Sun 2024-09-22 06:07:30 +0530; 12h left
       Docs: man:named(8)
   Main PID: 967 (named)
     Status: "running"
      Tasks: 8 (limit: 3416)
     Memory: 11.0M (peak: 11.5M)
        CPU: 77ms
     CGroup: /system.slice/named.service
             └─967 /usr/sbin/named -f -u bind

Sep 22 06:09:01 nelushi-VirtualBox named[967]: no longer listening on fe80::209d:7a82:e2c5:4317%3#53
Sep 22 06:09:01 nelushi-VirtualBox named[967]: no longer listening on fe80::a791:4c1:64f4:7ef8%4#53
Sep 22 06:09:01 nelushi-VirtualBox named[967]: listening on IPv6 interface enp0s8, fe80::209d:7a82:e2c5:4317%3#53
Sep 22 06:09:01 nelushi-VirtualBox named[967]: listening on IPv6 interface enp0s9, fe80::a791:4c1:64f4:7ef8%4#53
Sep 22 06:09:46 nelushi-VirtualBox named[967]: no longer listening on fe80::209d:7a82:e2c5:4317%3#53
Sep 22 06:09:46 nelushi-VirtualBox named[967]: no longer listening on fe80::a791:4c1:64f4:7ef8%4#53
Sep 22 06:09:46 nelushi-VirtualBox named[967]: listening on IPv6 interface enp0s8, fe80::209d:7a82:e2c5:4317%3#53
Sep 22 06:09:46 nelushi-VirtualBox named[967]: listening on IPv6 interface enp0s9, fe80::a791:4c1:64f4:7ef8%4#53
Sep 22 06:10:31 nelushi-VirtualBox named[967]: no longer listening on fe80::209d:7a82:e2c5:4317%3#53
Sep 22 06:10:31 nelushi-VirtualBox named[967]: no longer listening on fe80::a791:4c1:64f4:7ef8%4#53
```

7. Now, let's configure the vitual machine to use our local DNS server. First, find our server's IP address.

   ip addr show

```
enp0s10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
 link/ether 08:00:27:0d:8f:ad brd ff:ff:ff:ff:ff:ff
 inet 10.0.5.15/24 brd 10.0.5.255 scope global dynamic noprefixroute enp0s10
    valid_lft 86083sec preferred_lft 86083sec
 inet6 fe80::e526:70c0:49e8:5530/64 scope link noprefixroute
    valid_lft forever preferred_lft forever
```

8. Set the IP address as the DNS server in the /etc/resolv.conf file on the clients.
   sudo nano /etc/resolv.conf Add the line **nameserver 10. 0.5.15** (IP address of your server) or use **nameserver 8.8.8.8 (**if using Google's DNS)

```
nameserver 10.0.5.15
options edns0 trust-ad
search .
```

9. Finally, to test the DNS server is resolving hostnames correctly,

   Use **nslookup google.com**

   It should return information about the IP addresses of google.com as shown in the below picture. It means the DNS server is working correctly.

```
Server:         10.0.5.15
Address:        10.0.5.15#53

Non-authoritative answer:
Name:   google.com
Address: 74.125.200.139
Name:   google.com
Address: 74.125.200.101
Name:   google.com
Address: 74.125.200.100
Name:   google.com
Address: 74.125.200.102
```

   Or **dig google.com**

   The answer section shows the IP address of google.com and the server shows the IP address of the DNS server being used.

```
;; ANSWER SECTION:
google.com.             43      IN      A       74.125.200.138
google.com.             43      IN      A       74.125.200.100
google.com.             43      IN      A       74.125.200.102
google.com.             43      IN      A       74.125.200.101
google.com.             43      IN      A       74.125.200.113
google.com.             43      IN      A       74.125.200.139

;; Query time: 0 msec
;; SERVER: 10.0.5.15#53(10.0.5.15) (UDP)
;; WHEN: Mon Sep 16 22:28:57 +0530 2024
;; MSG SIZE  rcvd: 163
```

3. NTP (Network Time Protocol)

What is NTP?

NTP (Network Time Protocol) is a networking protocol used to synchronize the clocks of computers over a network. It ensures that all systems in a network have the same time, which is important for time-sensitive operations like logging events, scheduling tasks, and securing network transactions.

Importance of accurate system time

- **System synchronization:** Makes sure all computers and devices work together smoothly**.**
- **Ensures Correct Data:** Keeps data organized and prevents errors.
- **Improves Performance:** Helps applications and processes run efficiently.
- **Enhances Security:** Protects against issues with security certificates and passwords**.**

Configuration steps for NTP.

1. First, open the terminal in Ubuntu and update package list by executing the **sudo apt update** command.

2. Now, Install the NTP Client using **sudo apt install ntp** command.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo apt install ntp
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ntp is already the newest version (1:4.2.8p15+dfsg-2~1.2.2+dfsg1-4bu
ild2).
0 upgraded, 0 newly installed, 0 to remove and 251 not upgraded.
```

3. Next, open the NTP configuration file using **sudo nano /etc/ntp.conf** or **sudo nano /etc/ntpsec/ntp.conf.**

4. Optionally, add or ensure the following lines are present to use public NTP servers.

   server 0.pool.ntp.org iburst
   server 1.pool.ntp.org iburst
   server 2.pool.ntp.org iburst
   server 3.pool.ntp.org iburst

```
pool 0.ubuntu.pool.ntp.org iburst
pool 1.ubuntu.pool.ntp.org iburst
pool 2.ubuntu.pool.ntp.org iburst
pool 3.ubuntu.pool.ntp.org iburst

# Use Ubuntu's ntp server as a fallback.
server ntp.ubuntu.com

server 0.pool.ntp.org iburst
server 1.pool.ntp.org iburst
server 2.pool.ntp.org iburst
server 3.pool.ntp.org iburst
```

5. Next, restart the NTP service using **sudo systemctl restart ntp.**

6. Check the status using **sudo systemctl status ntp.**

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl start ntp
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl status ntp
● ntpsec.service - Network Time Service
     Loaded: loaded (/usr/lib/systemd/system/ntpsec.service; enabled; preset: enabled)
     Active: active (running) since Sun 2024-09-22 06:07:30 +0530; 12h left
       Docs: man:ntpd(8)
    Process: 998 ExecStart=/usr/libexec/ntpsec/ntp-systemd-wrapper (code=exited, status=0/SUCCESS)
   Main PID: 1006 (ntpd)
      Tasks: 1 (limit: 3416)
     Memory: 12.2M (peak: 12.8M)
        CPU: 246ms
     CGroup: /system.slice/ntpsec.service
             └─1006 /usr/sbin/ntpd -p /run/ntpd.pid -c /etc/ntpsec/ntp.conf -g -N -u ntpsec:ntpsec

Sep 21 17:44:18 nelushi-VirtualBox ntpd[1006]: DNS: Pool skipping: 162.159.200.1
Sep 21 17:44:18 nelushi-VirtualBox ntpd[1006]: DNS: Pool skipping: 162.159.200.123
Sep 21 17:44:18 nelushi-VirtualBox ntpd[1006]: DNS: dns_take_status: 1.ubuntu.pool.ntp.org=>good, 8
Sep 21 17:44:19 nelushi-VirtualBox ntpd[1006]: DNS: dns_probe: 0.ubuntu.pool.ntp.org, cast_flags:8, flags:101
Sep 21 17:44:19 nelushi-VirtualBox ntpd[1006]: DNS: dns_check: processing 0.ubuntu.pool.ntp.org, 8, 101
Sep 21 17:44:19 nelushi-VirtualBox ntpd[1006]: DNS: Pool skipping: 162.159.200.123
Sep 21 17:44:19 nelushi-VirtualBox ntpd[1006]: DNS: Pool skipping: 162.159.200.1
Sep 21 17:44:19 nelushi-VirtualBox ntpd[1006]: DNS: dns_take_status: 0.ubuntu.pool.ntp.org=>good, 8
Sep 21 17:47:10 nelushi-VirtualBox ntpd[1006]: IO: Deleting interface #13 enp0s8, fe80::209d:7a82:e2c5:4317%3#123,
Sep 21 17:47:10 nelushi-VirtualBox ntpd[1006]: IO: Deleting interface #14 enp0s9, fe80::a791:4c1:64f4:7ef8%4#123,
```

7. Use the below command to check if the system is synchronized with a public NTP server.

   ntpq -p

```
nelushi@nelushi-VirtualBox:~/Desktop$ ntpq -p
     remote                                refid      st t when poll reach   delay   offset   jitter
==============================================================================================================
 0.ubuntu.pool.ntp.org                    .POOL.          16 p    -  256    0   0.0000   0.0000   0.0001
 1.ubuntu.pool.ntp.org                    .POOL.          16 p    -  256    0   0.0000   0.0000   0.0001
 2.ubuntu.pool.ntp.org                    .POOL.          16 p    -  256    0   0.0000   0.0000   0.0001
 3.ubuntu.pool.ntp.org                    .POOL.          16 p    -  256    0   0.0000   0.0000   0.0001
*prod-ntp-3.ntp1.ps5.canonical.com        79.243.60.50     2 u   55   64   17 146.0214  -0.3113   1.4848
 0.pool.ntp.org                           .DNS.           16 u    -  512    0   0.0000   0.0000   0.0001
 1.pool.ntp.org                           .DNS.           16 u    -  512    0   0.0000   0.0000   0.0001
 2.pool.ntp.org                           .DNS.           16 u    -  512    0   0.0000   0.0000   0.0001
 3.pool.ntp.org                           .DNS.           16 u    -  512    0   0.0000   0.0000   0.0001
+time.cloudflare.com                      10.111.8.4       3 u   51   64   17   9.3680   1.0829   1.8600
+time.cloudflare.com                      10.111.8.4       3 u   53   64   17   9.6431   2.5612   0.7042
 time.cloudflare.com                      .INIT.          16 u    -   64    0   0.0000   0.0000   0.0001
 time.cloudflare.com                      .INIT.          16 u    -   64    0   0.0000   0.0000   0.0001
```

This will output a list of NTP servers and their status.
(*): Indicates the server currently being used for synchronization with your machine.
(+): Indicates a server that is reachable but not currently being used for synchronization.

My machine is now remotely connected to prod-ntp-3.ntp1.ps5.canonical.com server for synchronization.

3.Shell Scripting and Security

1. Shell scripting

Variables

```
#!/bin/usr/bash

name="Nelushi Wanasinghe"
age=22

echo "My name is $name and I am $age years old."
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ touch test.txt
nelushi@nelushi-VirtualBox:~/Desktop$ nano test.txt
nelushi@nelushi-VirtualBox:~/Desktop$ bash test.txt
My name is Nelushi Wanasinghe and I am 22 years old.
```

if-else

```
#!/bin/usr/bash

echo "Enter a number: "
read num

if [ $num -gt 0 ]
then
        echo "The number is positive"
elif [ $num -lt 0 ]
then
        echo "The number is negetive"
else
        echo "The number is zero"
fi
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ bash test.txt
Enter a number:
2
The number is positive
```

For loop

```
#!/bin/usr/bash

for item in "Apple" "Banana" "Orange"
do
        echo "I like $item"
done
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ bash test.txt
I like Apple
I like Banana
I like Orange
```

While loop

```
#!/bin/usr/bash

counter=1

while [ $counter -le 5 ]
do
        echo "Counter: $counter"
        counter=$((counter + 1))
done
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ bash test.txt
Counter: 1
Counter: 2
Counter: 3
Counter: 4
Counter: 5
```

Function

```
#!/bin/usr/bash

add_numbers() {
        result=$(( $1 + $2 ))
        echo "The sum of $1 and $2 is: $result"
}

add_numbers 5 10
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ bash test.txt
The sum of 5 and 10 is: 15
```

Arrays

```
#!/bin/usr/bash

fruits=("Apple" "Banana" "Cherry")

echo "First fruit: ${fruits[0]}"
echo "Second fruit: ${fruits[1]}"

for fruit in "${fruits[@]}"
do
        echo "I have a $fruit."
done
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ bash test.txt
First fruit: Apple
Second fruit: Banana
I have a Apple.
I have a Banana.
I have a Cherry.
```

Arithmetic operations

```
#!/bin/usr/bash

a=10
b=5

sum=$((a + b))
sub=$((a - b))
mul=$((a * b))
div=$((a / b))
mod=$((a % b))

echo "Sum: $sum"
echo "Substraction: $sub"
echo "Multiplication: $mul"
echo "Division: $div"
echo "modulus: $mod"
```

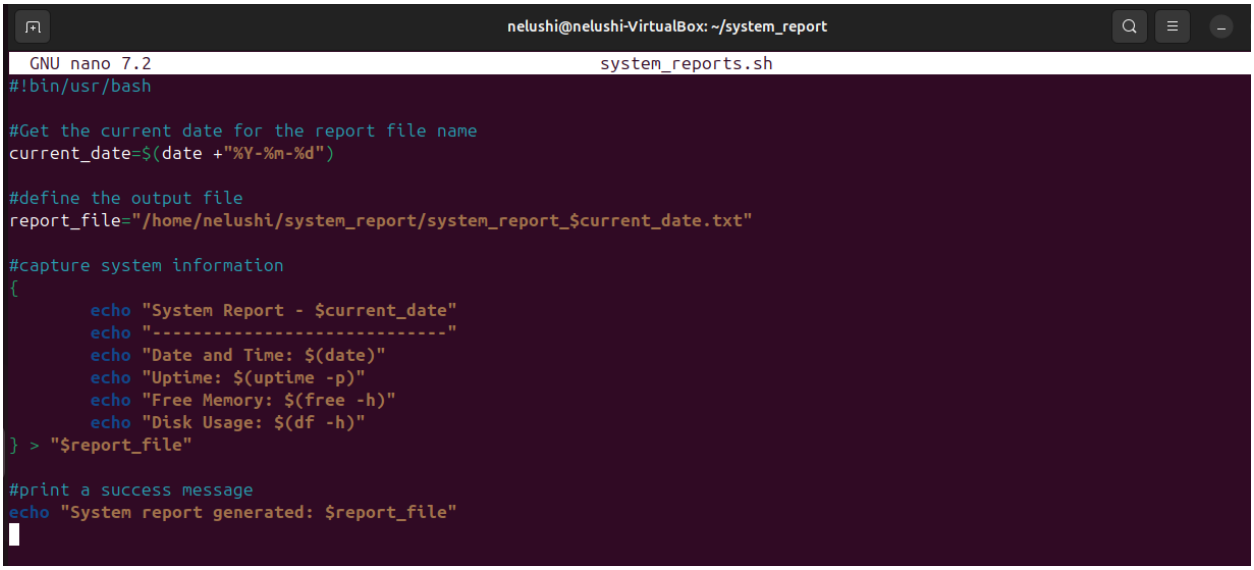
```
nelushi@nelushi-VirtualBox:~/Desktop$ bash test.txt
Sum: 15
Substraction: 5
Multiplication: 50
Division: 2
modulus: 0
```

i) <u>Script to Capture System Information</u>

1. Go to home directory – **cd ~**
   Create a directory – **mkdir system_report**
   Go to the directory **– cd system_report**
   Create a nano file and open the file **– nano system_reports.sh**

```
nelushi@nelushi-VirtualBox:~$ cd system_report
nelushi@nelushi-VirtualBox:~/system_report$ nano system_reports.sh
```

2. Create the shell script to capture the system information inside the nano file.



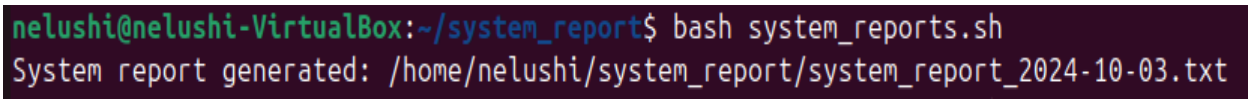
Press ctrl+o to save, enter and ctrl+x to exit from the file.

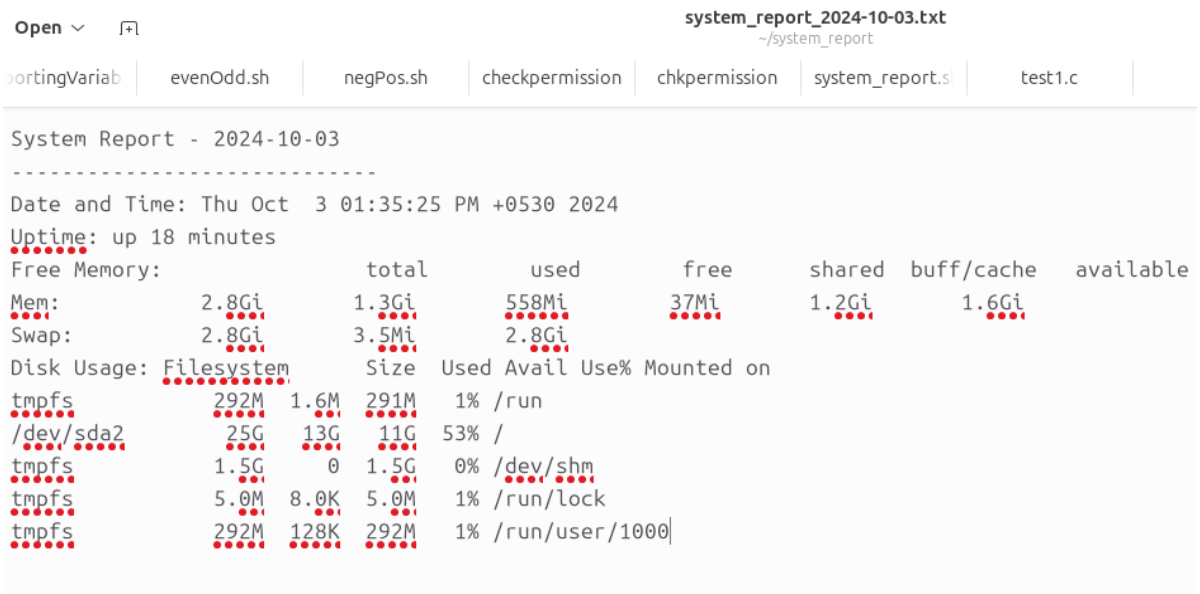
3. Use the following command make the script executable.

chmod +x system_reports.sh

4. Run the script to see if it's returning the system information.

bash system_reports.sh

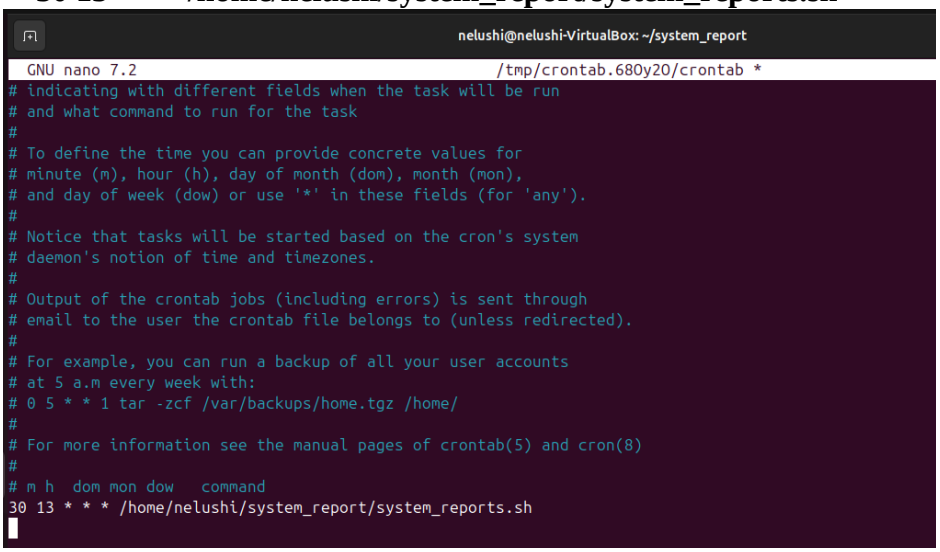




To automate the script using Cron Jobs,

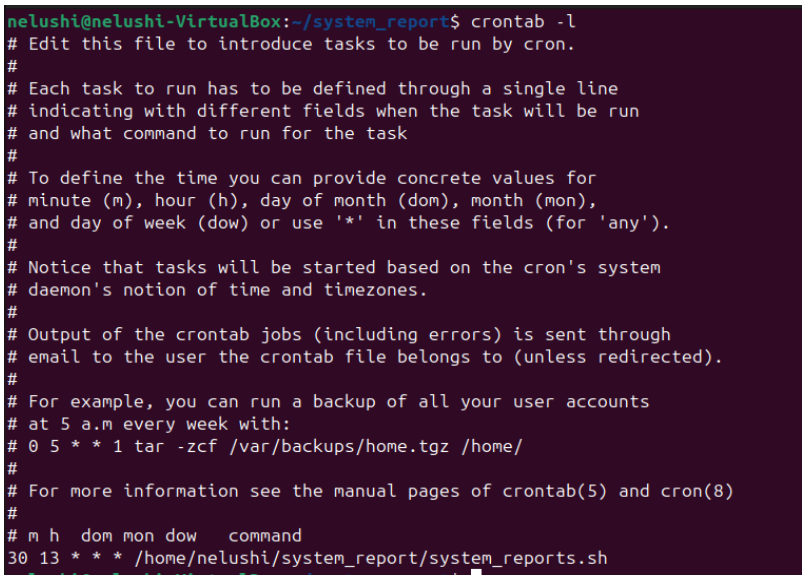
5. Open the crontab for editing using **crontab -e**
6. Add the Cron Job by adding the following line at the end of the file to schedule the script to run every day at **1:30 PM**

   30 13 * * * /home/nelushi/system_report/system_reports.sh



   30: Minute (30th minute)
   13: Hour (13:00 for 1 PM)
   ***:** Every day of the month
   ***:** Every month
   ***:** Every day of the week

7. Run **crontab -l** to verify the Cron Job. Now the script will run every day at 1:30 PM.



ii) Script to backup of a critical directory

1. Create the Backup Directory - /home/nelushi/backup/documents

   Make sure that the backup directory exists. If it doesn't exist, create it.

   mkdir -p /home/user/backup/documents
   mkdir -p ensures that if the directory doesn't exist, it will create it.

2. Navigate to home directory and create the shell script using nano.

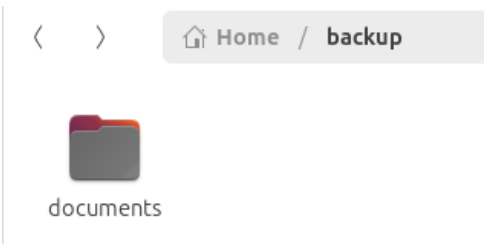
```
nelushi@nelushi-VirtualBox:~/Desktop$ mkdir -p /home/nelushi/backup/documents
nelushi@nelushi-VirtualBox:~/Desktop$ cd /home/nelushi
nelushi@nelushi-VirtualBox:~$ nano backup_script.sh
nelushi@nelushi-VirtualBox:~$ chmod +x backup_script.sh
```

```
nelushi@nelushi-VirtualBox: ~
 GNU nano 7.2                                  backup_script.sh
#!/bin/usr/bash

#define the source and destination directories
SRC_DIR="/home/nelushi/Documents"
DEST_DIR="/home/nelushi/backup/documents"

#get the current date
DATE=$(date +"%Y-%m-%d")

#create the backup file with the current date in it's name
tar -czf "$DEST_DIR/backup_$DATE.tar.gz" "$SRC_DIR"

#print a success message
echo "Backup completed successfully at $DEST_DIR/backup_$DATE.tar.gz"
```

SRC_DIR is the source directory (the directory we want to back up).

DEST_DIR is the destination directory (where the backup will be saved).

tar -czf creates a compressed .tar.gz file of the source directory.



3. Use the following command make the script executable.

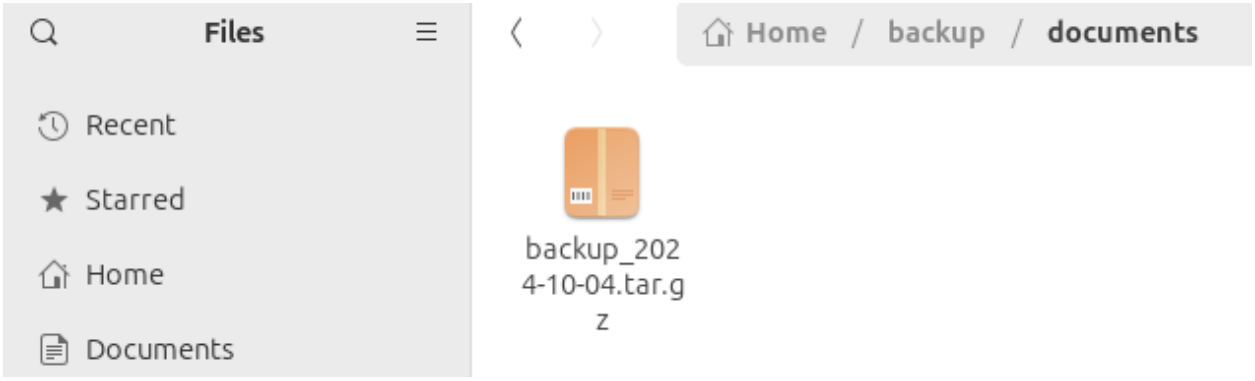
chmod +x system_reports.sh

4. Run the script to see if it's returning the system information.

bash system_reports.sh

```
nelushi@nelushi-VirtualBox:~$ bash backup_script.sh
tar: Removing leading `/' from member names
Backup completed successfully at /home/nelushi/backup/documents/backup_2024-10-03.tar.gz
```

This is the backup file of the source directory. It is saved in /home/nelushi/backup/documents.

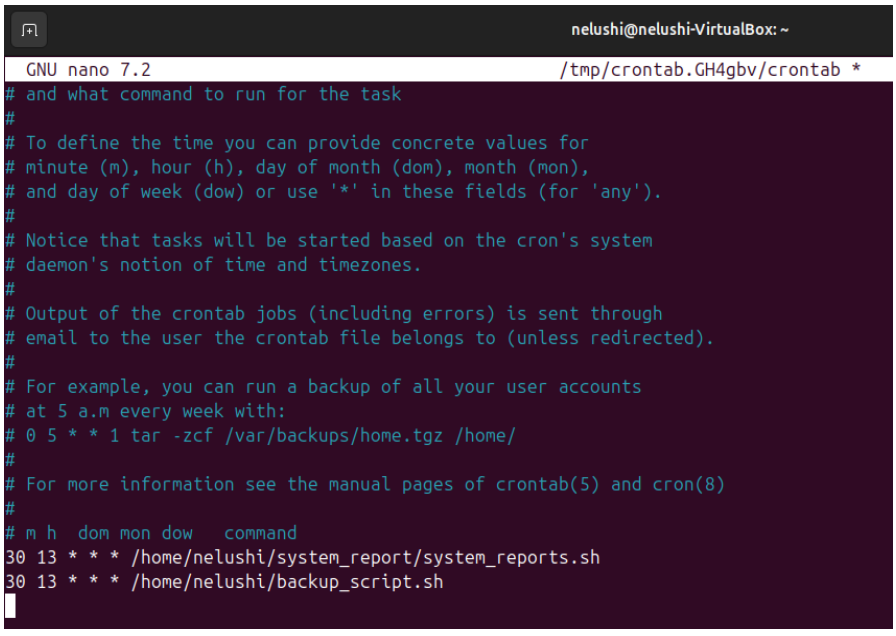


To Schedule the Backup with Cron Jobs periodically,

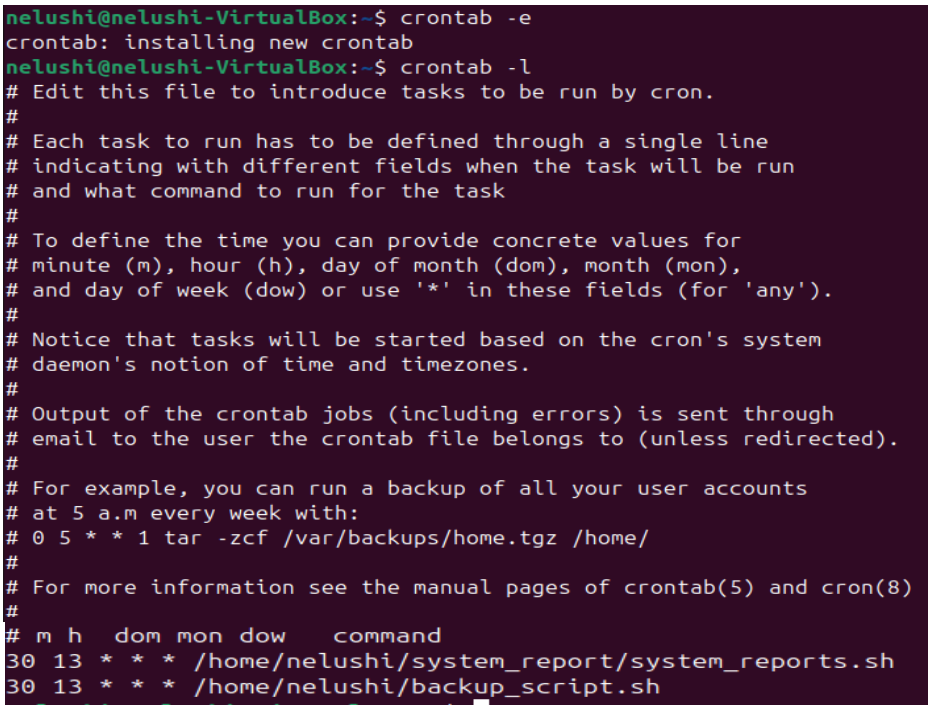
5. Open the crontab for editing: **crontab -e**

6. let's schedule it to run every day at 1:30 PM.

30 13 * * * /home/user/backup_script.sh



7. Run **crontab -l** to verify the Cron Job. Now the script will run every day at 1:30 PM.



2. SSH (Secure Shell)

Understanding the Importance of SSH (Secure Shell)

SSH, or secure shell, is a protocol used to remotely log into and manage devices over an encrypted data stream. It ensures that data transmitted between two systems like our computer and a remote server is protected against potential eavesdropping or man-in-the-middle attacks.

SSH is used for,

- Secure remote login and administration.
- Secure file transfer via SCP or SFTP.
- Tunneling and port forwarding.

Configuration steps for SSH

1. open the terminal in Ubuntu and Update Package List by executing the **sudo apt update** command.

2. Now, Install the SSH Server package (openssh-server) using **sudo apt install openssh-server** command.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssh-server is already the newest version (1:9.6p1-3ubuntu13.5).
0 upgraded, 0 newly installed, 0 to remove and 272 not upgraded.
```

3. Allow SSH Through Firewall – **sudo ufw allow ssh**
4. Check the applied firewall rules **– sudo ufw status**

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ufw status
[sudo] password for nelushi:
Status: active

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

5. Start the server and check if it's running successfully using,

   sudo systemctl start ssh
   sudo systemctl status ssh

6. Ensure the SSH service starts automatically at boot using **sudo systemctl enable ssh**.

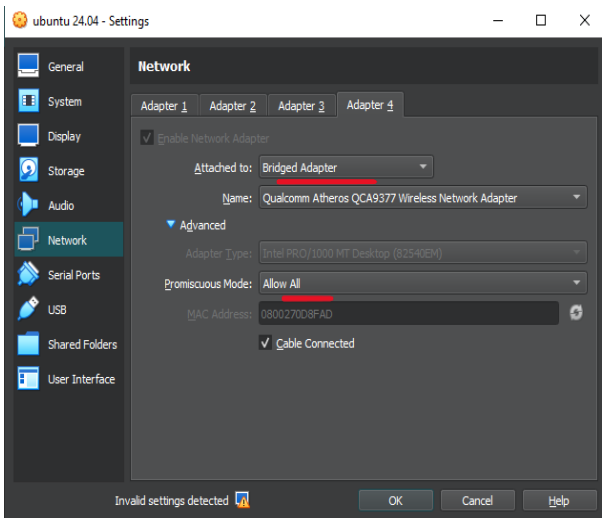
```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
Created symlink /etc/systemd/system/sshd.service → /usr/lib/systemd/system/ssh.service.
Created symlink /etc/systemd/system/multi-user.target.wants/ssh.service → /usr/lib/systemd/system/ssh.service.
```

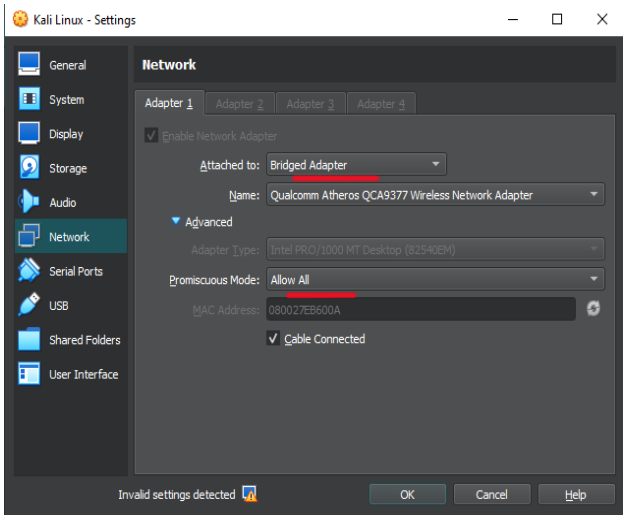
```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Wed 2024-10-02 01:41:23 +0530; 29s ago
TriggeredBy: ● ssh.socket
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 7447 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 7448 (sshd)
      Tasks: 1 (limit: 3416)
     Memory: 1.2M (peak: 1.4M)
        CPU: 24ms
     CGroup: /system.slice/ssh.service
             └─7448 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Oct 02 01:41:23 nelushi-VirtualBox systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Oct 02 01:41:23 nelushi-VirtualBox sshd[7448]: Server listening on :: port 22.
Oct 02 01:41:23 nelushi-VirtualBox systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
```

Now, let's connect to our Virtual Machine remotely using SSH Client.

I'm using **kali linux, as the ssh host machine** and **Ubuntu as the ssh client**.

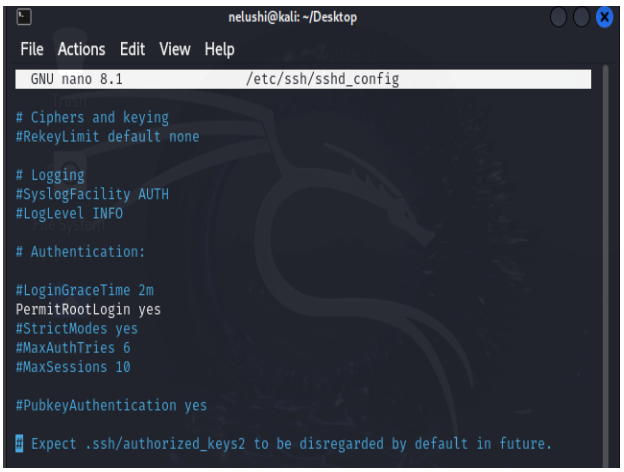
7. Set the network adapters of both kali linux and ubuntu to **bridged Adapter.**
   Set promiscuous mode to **allow all** in both.

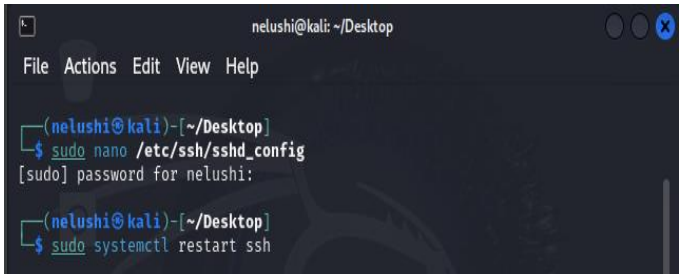




8. Open the terminal in kali linux (the host machine) and type **sudo nano /etc/ssh/sshd_config** to configure the file.

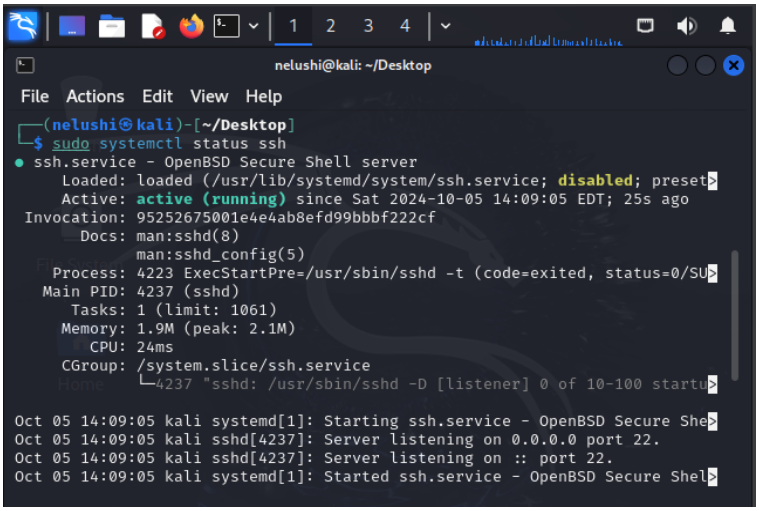
9. In Authentication section, set **PermitRootLogin** to **yes.**



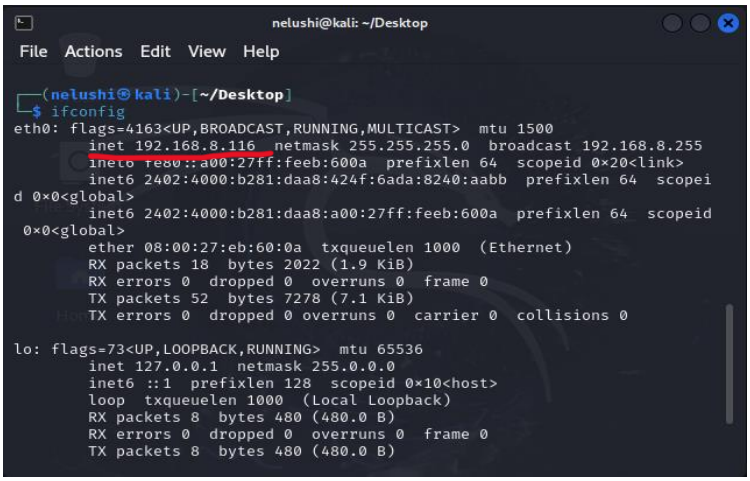


10. Start the ssh server – **sudo systemctl start ssh**

    Check the status to verify – **sudo systemctl status ssh**



11. Check the IP address **– ip addr**



12. Go to the client machine (ubuntu) and execute this command to connect to the ssh server (kali).

    ssh nelushi@192.168.8.116

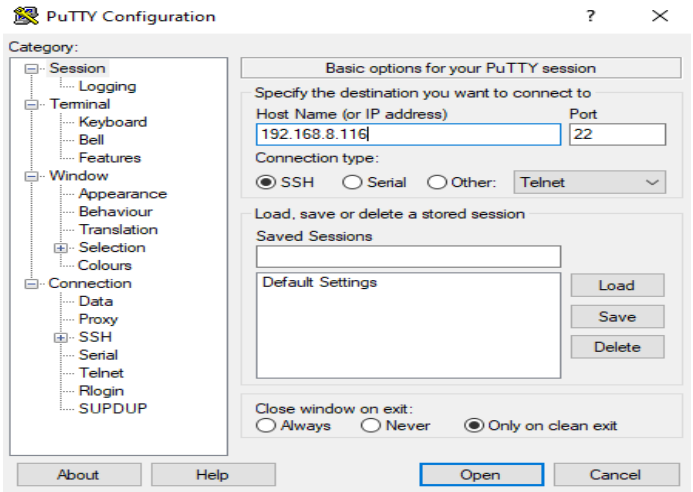
    nelushi – username in kali linux(ssh server)
    192.168.8.116 – ip address of the ssh server

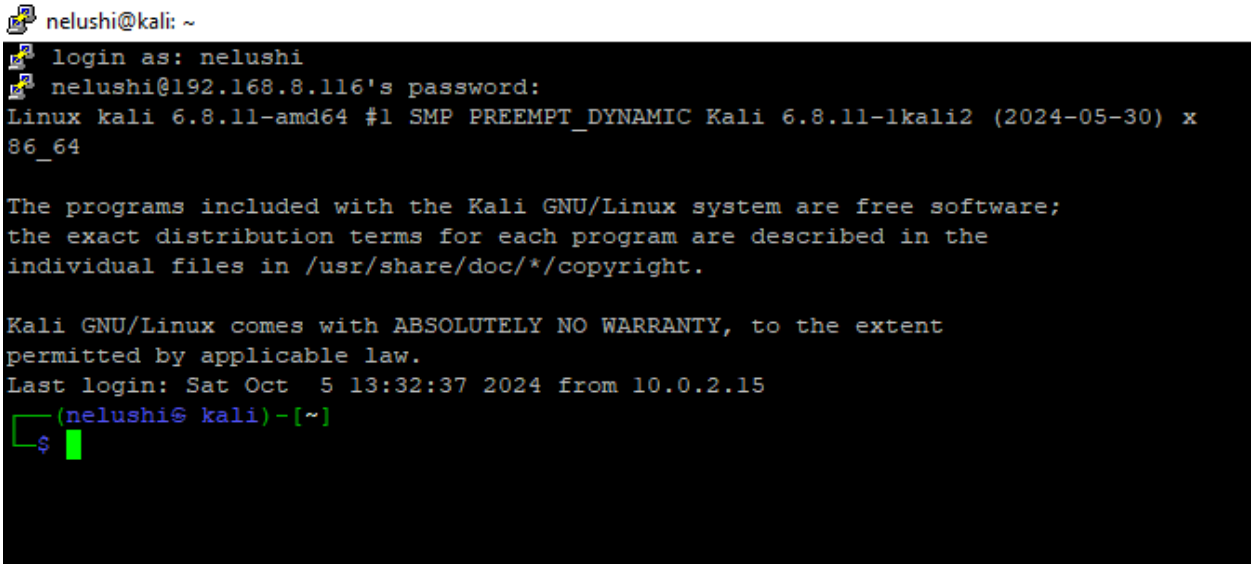
Then it prompts to enter the password of the username.

```
nelushi@nelushi-VirtualBox:~/Desktop$ ssh nelushi@192.168.8.116
The authenticity of host '192.168.8.116 (192.168.8.116)' can't be established.
ED25519 key fingerprint is SHA256:vMj1nZvGSBJU0az0qM2VgAtV1ESgxtsHmogEzkSjY4A.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.8.116' (ED25519) to the list of known hosts.
nelushi@192.168.8.116's password:
Linux kali 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.8.11-1kali2 (2024-05-30) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Oct  5 14:12:25 2024 from 192.168.8.146
┌──(nelushi㉿kali)-[~]
└─$
```

Now the ubuntu machine (ssh client) is successfully connected to the ssh server (kali linux) remotely using ssh.

13. We can do the same process using **windows putty software as the ssh client** instead of using ubuntu.
14. Open putty software and enter the ip address of the ssh server (kali linux).





Now the windows putty (ssh client) is successfully connected to the ssh server (kali linux) remotely using ssh.

3. Iptables and ACLs

1. Ensure iptables is installed (it usually is by default). To check, run

   sudo iptables -L

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

2. If want to start fresh by clearing any existing firewall rules: **sudo iptables -F**
3. Set the default policy to drop all incoming traffic by default.

   Explanation: This ensures that by default, no incoming traffic is allowed unless specified by additional rules.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -F
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -P INPUT DROP
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -P FORWARD DROP
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -P OUTPUT ACCEPT
```

4. To prevent breaking existing connections or related services, allow established connections.

   sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

i) Web Server Security

To configure web server security using iptables so that only incoming traffic on port 80 (HTTP) and port 443 (HTTPS) is allowed while blocking all other incoming traffic, we need to,

1. Allow HTTP (Port 80) and HTTPS (Port 443) Traffic

sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT

2. Allow Loopback Interface (Localhost): Ensure that our system can communicate with itself (important for local services)

sudo iptables -A INPUT -i lo -j ACCEPT

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -i lo -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     all  --  anywhere             anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
```

ii) Remote Administration Access

1. Allow SSH (Port 22) from specific IP address. Let's assume the trusted machine's IP address is 192.168.8.146.

sudo iptables -A INPUT -p tcp --dport 22 -s 192.168.8.146 -j ACCEPT

This rule allows SSH traffic only from the IP address 192.168.8.146.

2. Block all other SSH traffic.

sudo iptables -A INPUT -p tcp --dport 22 -j DROP

This rule blocks all other SSH traffic except from the trusted machine we allowed.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp --dport 22 -s 192.168.8.146 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp --dport 22 -j DROP
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     tcp  --  192.168.8.146        anywhere             tcp dpt:ssh
DROP       tcp  --  anywhere             anywhere             tcp dpt:ssh

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

iii) Allow Specific Applications

If, the application uses port 443 (Ex: for a video conferencing app or HTTPS traffic),

1. Allow incoming and outgoing traffic for video conferencing app. Port (443)

sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
```

2. Other than the video conferencing application that uses port 443 to communicate, we can allow other applications with the specific port they use for communications.

HTTP: Port 80
HTTPS: Port 443
SSH: Port 22
FTP: Port 21
SMTP (Email): Port 25
DNS: Port 53
Video Conferencing (ex: Zoom): uses ports 80, 443, and dynamic UDP ports.

3. If we want to **allow specific applications that use port 21 (FTP) and port 25 (SMTP),** we can set rules as follows.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp --dport 21 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A OUTPUT -p tcp --dport 21 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp --dport 25 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A OUTPUT -p tcp --dport 25 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ftp
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:smtp

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ftp
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:smtp
```

iv) <u>Allow Pings (ICMP Echo Request)</u>

1. Allow incoming ICMP Echo Requests

sudo iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT

- -A INPUT: Append this rule to the INPUT chain to allow incoming traffic.
- -p icmp: Specify the protocol as ICMP.
- --icmp-type echo-request: Target the ICMP echo request type, which is used for ping.
- -j ACCEPT: Jump to the ACCEPT target to allow the traffic.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  DESKTOP-83T6A3G      anywhere             tcp dpt:ssh
DROP       tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     icmp --  anywhere             anywhere             icmp echo-request

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
```

v) Printer Server Access

1. Allow printing traffic from specific IP addresses within local network.

   sudo iptables -A INPUT -p tcp -s 10.0.5.15 --dport 9100 -j ACCEPT
   sudo iptables -A INPUT -p tcp -s 192.168.8.146 --dport 9100 -j ACCEPT

2. Block all other incoming traffic to Port 9100. This rule ensures that no other IP addresses can access the printer server.

   sudo iptables -A INPUT -p tcp --dport 9100 -j DROP

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp -s 10.0.5.15 --dport 9100 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp -s 192.168.8.146 --dport 9100 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -p tcp --dport 9100 -j DROP
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ftp
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:smtp
ACCEPT     icmp --  anywhere             anywhere             icmp echo-request
ACCEPT     tcp  --  10.0.5.15            anywhere             tcp dpt:9100
ACCEPT     tcp  --  192.168.8.146        anywhere             tcp dpt:9100
DROP       tcp  --  anywhere             anywhere             tcp dpt:9100

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:https
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ftp
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:smtp
```

3. Save all the firewall rules using - **sudo iptables-save**
4. Verify the existing rules using - **sudo iptables -L -n -v**

4. Best practices

New strategies have appeared in the process of protecting a Linux system from external attacks and intrusions, and one of the key elements of the concept is the proper configuration of the network interfaces available on the computer system. The security approaches and measures for network interface configuration are being specified here.

1. Disable Unused Network Interfaces

Explanation - If systems contain multiple network interfaces (Ex: Ethernet, WiFi) that are switched on but are not in use, it can create vulnerabilities for the machine and make it expose to attacks.

How - Check available network interfaces – **ip addr show**

Disable the interfaces that are not in use – **sudo ip link set enp0s9 down** (if we want to disable enp0s9 network interface)

Verify if its disabled – **sudo ip link show enp0s9**

Benefit - By doing this, we can prevent potential attackers from using the unused interfaces to gain unauthorized access to the system.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ip link show enp0s9
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:c9:1e:19 brd ff:ff:ff:ff:ff:ff
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ip link set enp0s9 down
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ip link show enp0s9
4: enp0s9: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN mode DEFAULT group default qlen 1000
    link/ether 08:00:27:c9:1e:19 brd ff:ff:ff:ff:ff:ff
```

We can also disable the unnecessary services.

First, identify running services - **systemctl list-units --type=service**

```
nelushi@nelushi-VirtualBox:~/Desktop$ systemctl list-units --type=service
  UNIT                                LOAD   ACTIV>
  accounts-daemon.service             loaded activ>
  alsa-restore.service                loaded activ>
  apparmor.service                    loaded activ>
  apport.service                      loaded activ>
  avahi-daemon.service                loaded activ>
  colord.service                      loaded activ>
  console-setup.service               loaded activ>
  cron.service                        loaded activ>
  cups-browsed.service                loaded activ>
  cups.service                        loaded activ>
  dbus.service                        loaded activ>
  fwupd.service                       loaded activ>
  gdm.service                         loaded activ>
  gnome-remote-desktop.service        loaded activ>
lines 1-15...skipping...
  UNIT                                LOAD   ACTIVE SUB     DESCRIPTION
  accounts-daemon.service             loaded active running Accounts Service
  alsa-restore.service                loaded active exited  Save/Restore Sound Card State
  apparmor.service                    loaded active exited  Load AppArmor profiles
  apport.service                      loaded active exited  automatic crash report generation
  avahi-daemon.service                loaded active running Avahi mDNS/DNS-SD Stack
  colord.service                      loaded active running Manage, Install and Generate Color Profiles
  console-setup.service               loaded active exited  Set console font and keymap
```

We can disable unwanted services – **sudo systemctl disable colord.service**

Benefit - Disabling unnecessary services reduces the system's attack surface, improving security, and frees up resources, enhancing performance.

2. Secure Network Interface with Firewall Rules (iptables)

Explanation - Configuring firewall rules to specific network interfaces ensures that only authorized traffic can access or leave the system.

How – We can use **iptables or ufw** to define strict access rules.

Ex - allowing only HTTP and HTTPS traffic on enp0s9 network interface.

Using iptables,

sudo iptables -A INPUT -i enp0s9 -p tcp --dport 80 -j ACCEPT

sudo iptables -A INPUT -i enp0s9 -p tcp --dport 443 -j ACCEPT

sudo iptables -A INPUT -i enp0s9 -j DROP to block all other traffic

sudo iptables -L -v -n to check the applied rules.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -i enp0s9 -p tcp --dport 80 -j ACCEPT
[sudo] password for nelushi:
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -i enp0s9 -p tcp --dport 443 -j ACCEPT
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -A INPUT -i enp0s9 -j DROP
nelushi@nelushi-VirtualBox:~/Desktop$ sudo iptables -L -v -n
Chain INPUT (policy ACCEPT 75 packets, 8303 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     6    --  enp0s9 *       0.0.0.0/0            0.0.0.0/0            tcp dpt:80
    0     0 ACCEPT     6    --  enp0s9 *       0.0.0.0/0            0.0.0.0/0            tcp dpt:443
    0     0 DROP       0    --  enp0s9 *       0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Also, we can use ufw to define these rules.

First, we have to enable ufw.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ufw enable
[sudo] password for nelushi:
Firewall is active and enabled on system startup
```

Apply the rules.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ufw allow ssh
Rule added
Rule added (v6)
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ufw allow http
Rule added
Rule added (v6)
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ufw allow https
Rule added
Rule added (v6)
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ufw deny from any
Rule added
Rule added (v6)
```

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo ufw status
Status: active

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

Benefit - Proper firewall configurations defend the system from unauthorized access and possible attacks by controlling network traffic.

3. Enable Network Interface Hardening

Explanation - Strengthen the network stack by disabling certain protocols and features that may be unnecessary or insecure.

How - Edit the **/etc/sysctl.conf** file and add the following lines to disable IP forwarding, source routing, and redirects.

1. **Disable IP forwarding** - net.ipv4.ip_forward=0

   Benefit: This setting prevents the system from routing packets between interfaces, which is useful if our machine is not intended to act as a router. Disabling IP forwarding reduces the attack surface by limiting the potential for routing attacks.

2. **Disable Source routing** - net.ipv4.conf.all.accept_source_route=0

Benefit: Source routing allows the sender of a packet to specify the route that a packet takes. Disabling this feature helps prevent certain types of network attacks, such as IP spoofing and man-in-the-middle attacks.

3. **Disable accepting redirects** - net.ipv4.conf.all.accept_redirects=0

   net.ipv6.conf.all.accept_redirects=0

   Benefit: Disabling redirects prevents the system from accepting ICMP redirect messages, which can be used in certain attacks to mislead the routing of packets.

To apply the changes, use **sudo sysctl -p**

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo nano /etc/sysctl.conf
[sudo] password for nelushi:
nelushi@nelushi-VirtualBox:~/Desktop$ sudo sysctl -p
net.ipv4.ip_forward = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.all.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0
```

To verify that the settings have been applied correctly, we can check the current values by executing each line separately. Each command should return 0, indicating that the settings are correctly applied.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
nelushi@nelushi-VirtualBox:~/Desktop$ sysctl net.ipv4.conf.all.accept_source_route
net.ipv4.conf.all.accept_source_route = 0
nelushi@nelushi-VirtualBox:~/Desktop$ sysctl net.ipv4.conf.all.accept_redirects
net.ipv4.conf.all.accept_redirects = 0
nelushi@nelushi-VirtualBox:~/Desktop$ sysctl net.ipv6.conf.all.accept_redirects
net.ipv6.conf.all.accept_redirects = 0
```

4. Monitor network traffic.

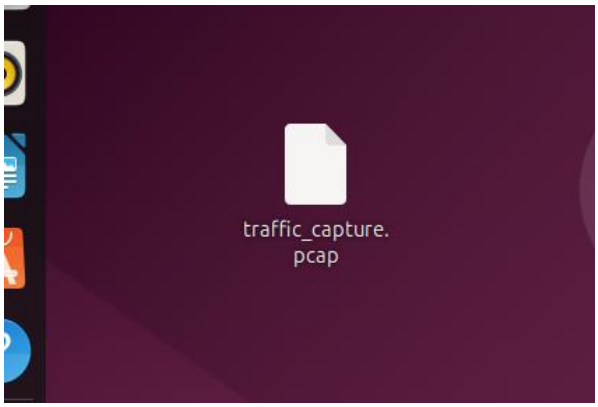
Explanation - network traffic monitoring is retrieving, viewing, and analyzing data packets flowing through our network. This security practice makes it possible to detect any attempts of unauthorized access or incidents tthat may represent an attack.

- First, install monitoring tools such as tcpdump or wireshark.
- Use tcpdump to capture traffic on our primary network interface.

```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo apt install tcpdump
[sudo] password for nelushi:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
tcpdump is already the newest version (4.99.4-3ubuntu4).
tcpdump set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 272 not upgraded.
```

sudo tcpdump -i enp0s10 -w traffic_capture.pcap is used to capture network traffic on a specified network interface and write the captured packets to a file named traffic_capture.pcap. The .pcap extension indicates that the file is in the Packet Capture format, which can be analyzed later using tools like Wireshark.

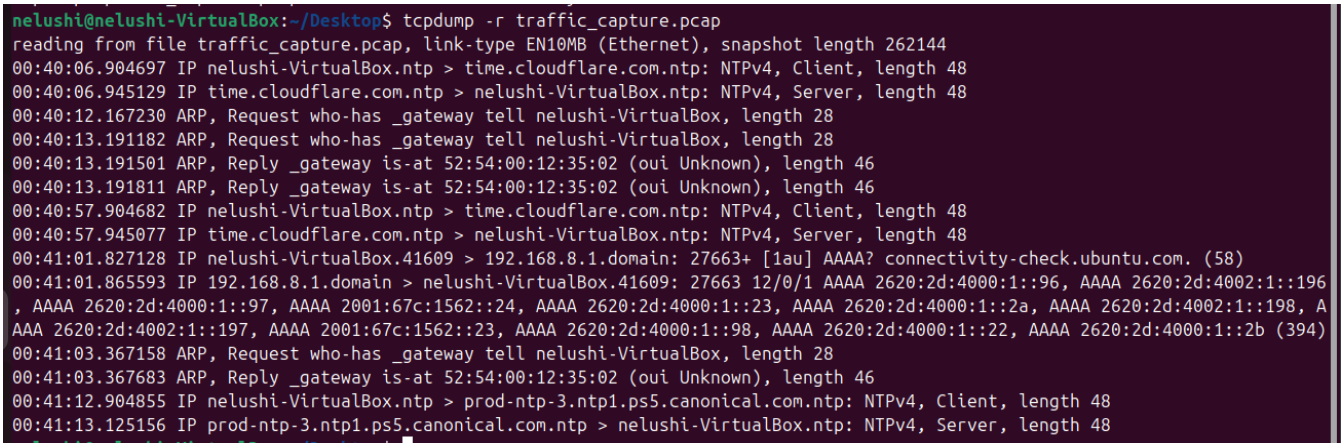
```
nelushi@nelushi-VirtualBox:~/Desktop$ sudo tcpdump -i enp0s10 -w traffic_capture.pcap
tcpdump: listening on enp0s10, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C14 packets captured
14 packets received by filter
0 packets dropped by kernel
```



- Moreover, we can filter specific traffics.

Ex: To filter HTTP traffic **- sudo tcpdump -i enp0s10 port 80 -w http_traffic.pcap**

- Next, we can analyze the captured data and look for anomalies such as unusual traffic patterns or unknown IP addresses using **tcpdump -r traffic_capture.pcap.**



Benefits:

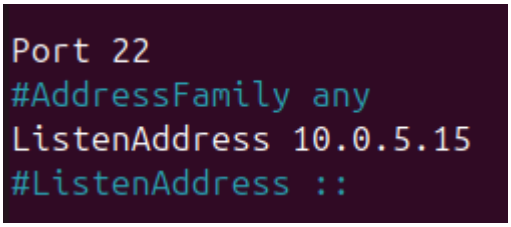
Better Incident Response: The detailed network activity logs help us to more effectively address potential security incidents

Better Visibility: With monitoring tools, we can see the status of our network like if there are some legit or malicious activities.

5. Implement Network Interface Binding

Explanation: to ensure that certain services or applications are accessible only on specific network interfaces, we can bind a service to a private or internal interface (ex: 10.0.0.0/8). Therefore, we can ensure it's only accessible within a trusted network, reducing the risk of attacks from external sources.

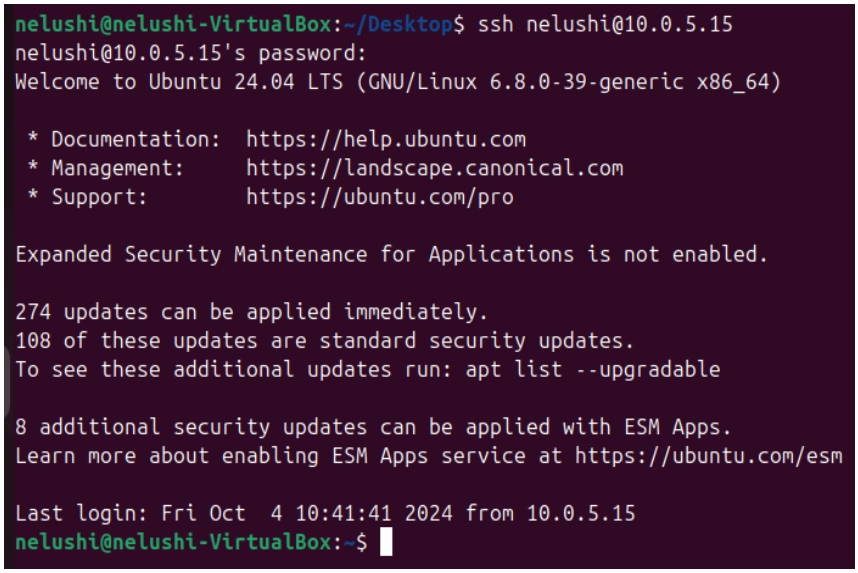
- First, identify the service we want to bind. I choose ssh service.
- Open the SSH configuration file - **/etc/ssh/sshd_config** and find the line that starts with ListenAddress and modify it to ListenAddress 10.0.5.15 (your ip address).



- Restart the service - **sudo systemctl restart sshd**

- To verify that the SSH service is properly bound to our internal IP address and not accessible from external networks, we can use,

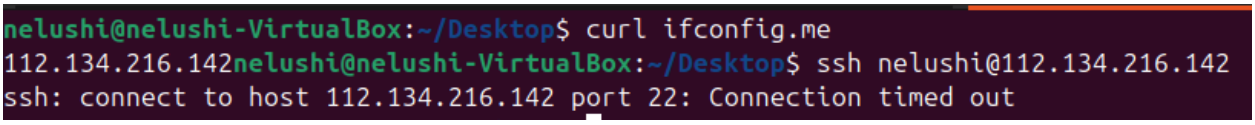
  ssh nelushi@10.0.5.15 – my private ip address



- To test accessibility from external networks, use your public ip address.
  To find the public ip address – **curl ifconfig.me**

  ssh nelushi@112.134.216.142 – my public ip address

  If the SSH service is properly configured to listen only on the internal IP address, we should see a message indicating that the connection is refused or timed out.



Benefits:

Limiting access: It ensures that only reliable network segments have access to sensitive services.
Increasing control: It aids in traffic flow management and makes sure services are only accessible via approved interfaces.
Reducing vulnerabilities: By restricting services to internal or private interfaces, it lowers the possibility of unauthorized access or attacks.

Enhanced Security: we can limit the exposure of services to the network and increase the difficulty of unauthorized access by binding them to certain interfaces.
Better Traffic Management: This enhances traffic management by limiting access to services to approved interfaces and making the most use of available resources.