Sri Lanka Institute of Information Technology

Report on Linux Foundations: Virtual Machine Setup and System Administration

Gunasekara D T
IT23621138

Secure Network Programming - IE2012

May, 2025

Table of Contents

01. Basics of Linux Environments

1.1 Virtual Machine Setup:

The following steps can be taken to successfully install a virtual machine software (VirtualBox) and a Linux Distribution (Ubuntu).

Step 01:
Download VirtualBox from the official website. (https:/www.virtualbox.org/wiki/Downloads)

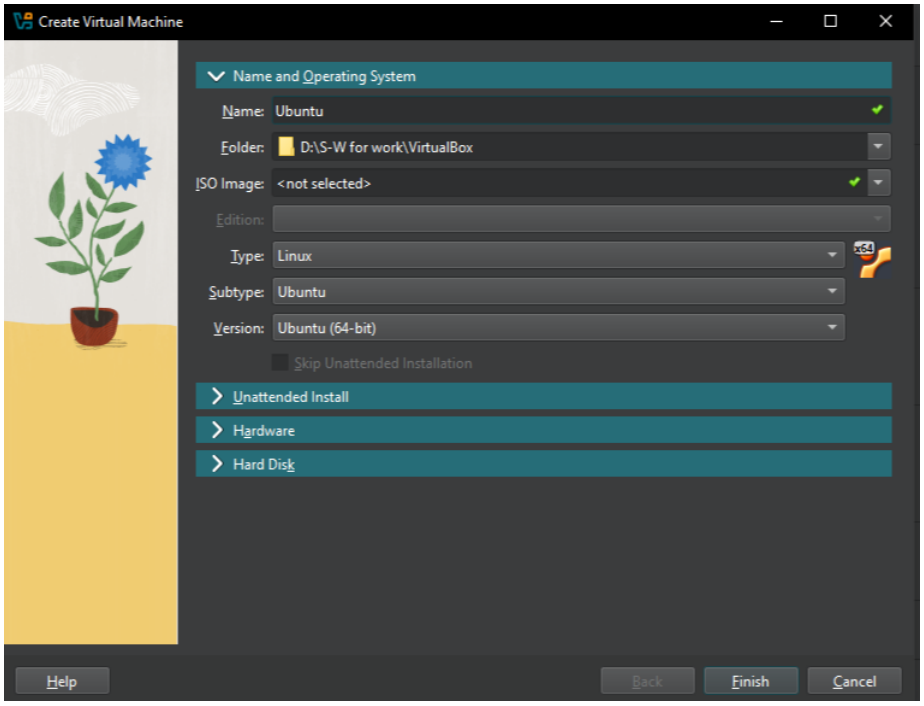
Step 02:
Download Ubuntu 22.04 LTS ISO file from the Ubuntu website. (https://ubuntu.com/download/desktop)

Step 03:
Open the VirtualBox software to create a new virtual machine.

- Select the 'New' icon in the top ribbon.

- When a new window pops up fill in the following:

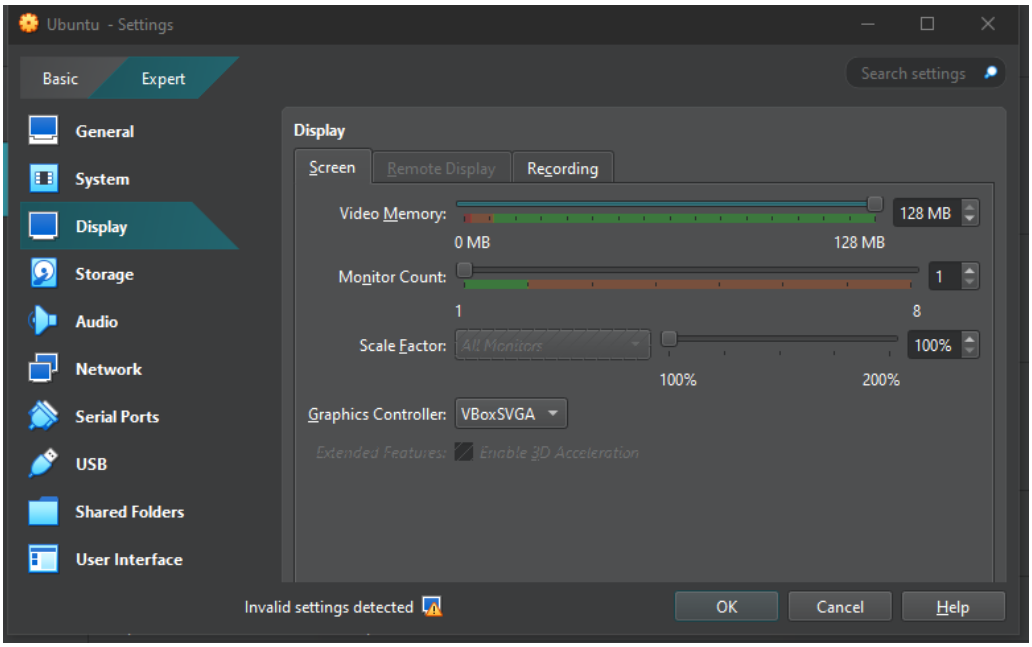
  - Set a New VM name (e.g., "Ubuntu 22.04")
  - Then select the type (Linux), subtype (Ubuntu) and version (Ubuntu 64-bit)



- Select the ‘Hardware’ section and then allocate memory for the VM. (a capacity more than 2048MB is recommended)

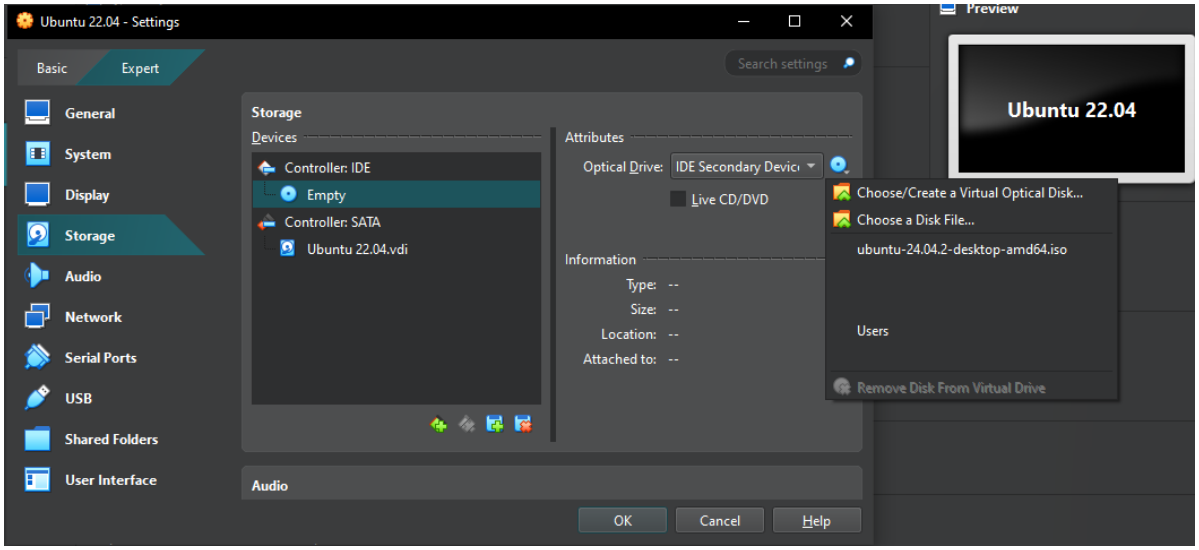
- Then move on to the ‘Hard disk’ section to create a virtual hard disk that is VDI, dynamically created and is recommended to keep the capacity at 30BG or more.

- As Ubuntu requires more power for its graphics, it is recommended to increase the video memory to the maximum for smoother operation. (It is also recommended to select a proper graphics controller like VBoxSVGA).



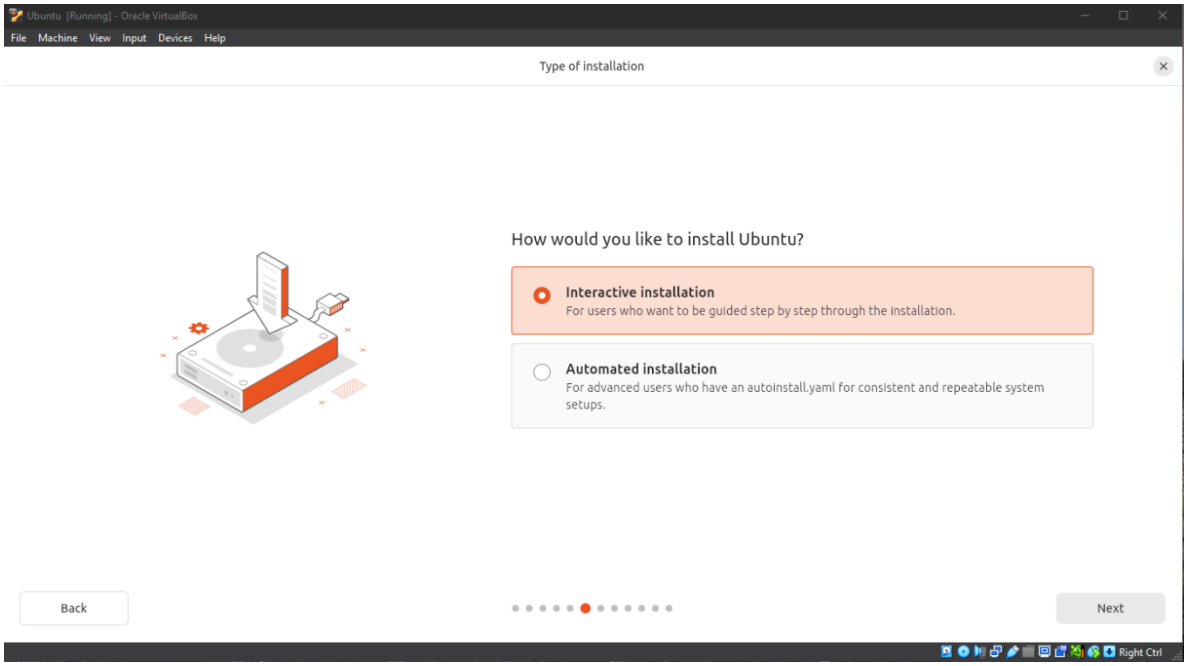
Step 04:

The downloaded ISO file can now be attached to the empty controller IDE, by selecting the CD icon at the top right of the settings pop up box.



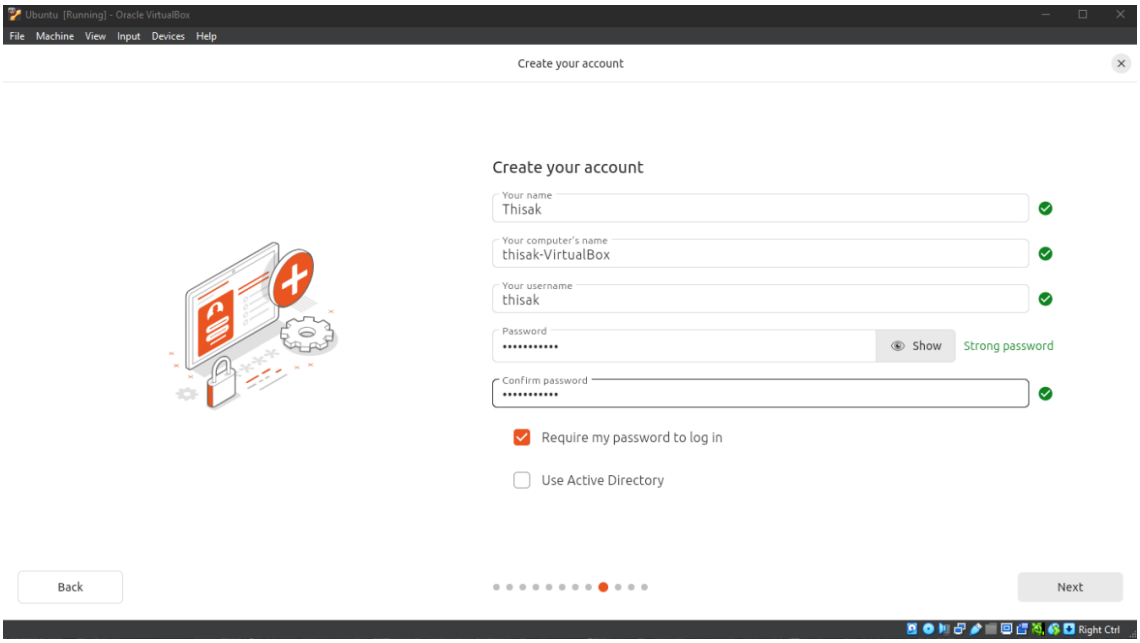
Step 05:

Now, the Ubuntu distribution can be installed by customizing the installation steps to your needs.



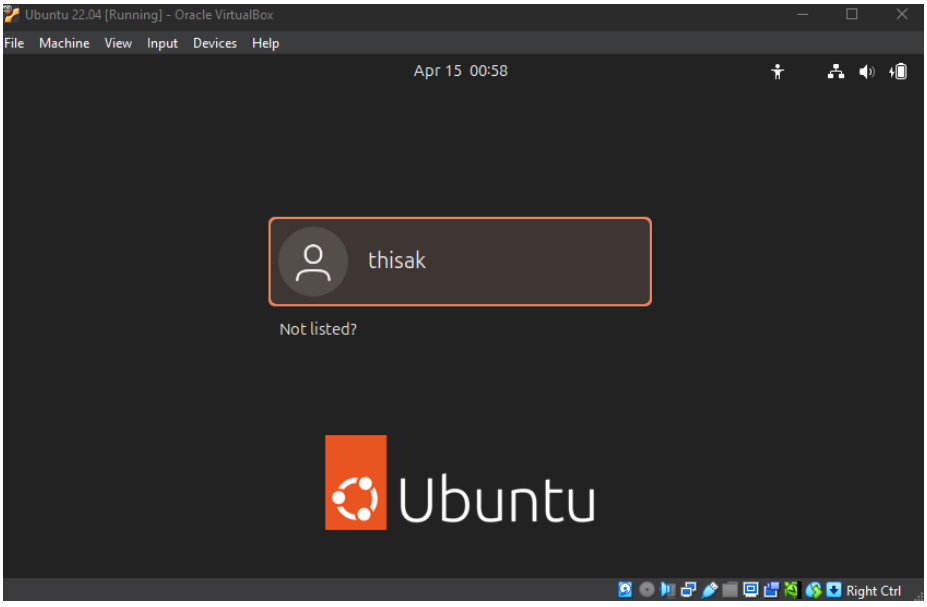
Step 06:

At the end of the installation process, a user account should be created with a username and a strong password.



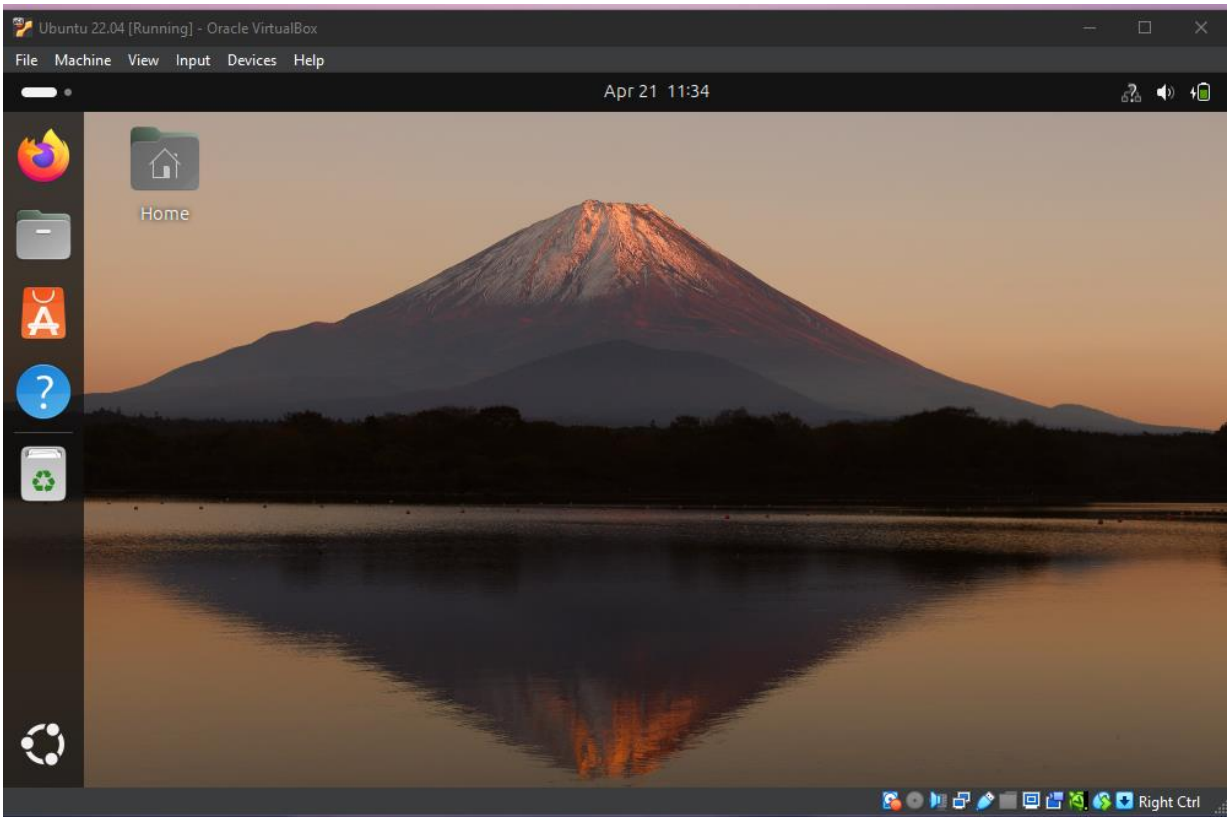
Step 07:

After completed installation, reboot the VM and login with the created user credentials.



Step 08:

A successful installation can be confirmed when a proper desktop is showing in the virtual machine.



1.2 Linux commands for basic navigation, system information retrieval and user management:

The following selection of commands can be used to learn on how to navigate the Linux environment effectively.

- **File System and Navigation:**

1. pwd	This command prints the full path of the current working directory, helping to know where you are in the file system.

```
thisak@thisak:~/Desktop$ pwd
/home/thisak/Desktop
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
```

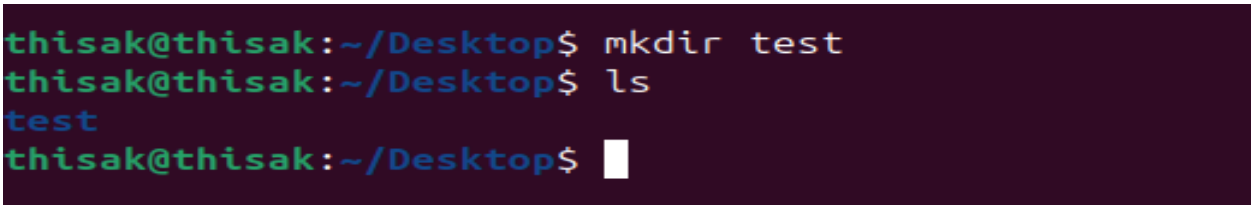
2. cd	Used to change the current directory; for example, cd /home/user moves you to the /home/user directory.

```
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$ cd /home/thisak/Desktop
thisak@thisak:~/Desktop$
```

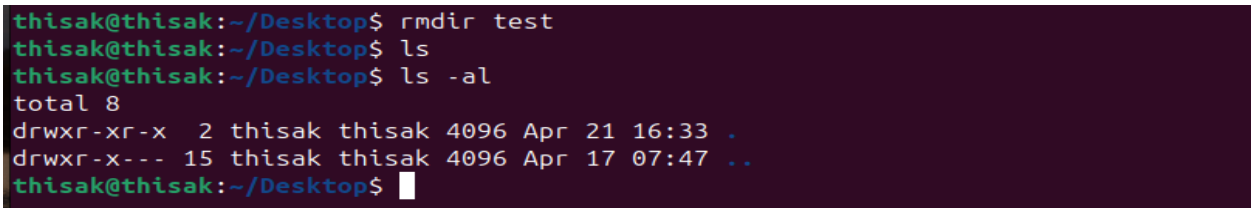
3. ls	Lists the names of files and folders in the current directory.
ls -al	Lists all files and directories, including hidden ones (those starting with a dot), in a detailed (long) format showing permissions, ownership, and the size.

```
thisak@thisak:~/Desktop$ ls
thisak@thisak:~/Desktop$ ls -al
total 8
drwxr-xr-x  2 thisak thisak 4096 Apr 21 11:34 .
drwxr-x--- 15 thisak thisak 4096 Apr 17 07:47 ..
thisak@thisak:~/Desktop$
```

4. mkdir	Creates a new directory with the specified name



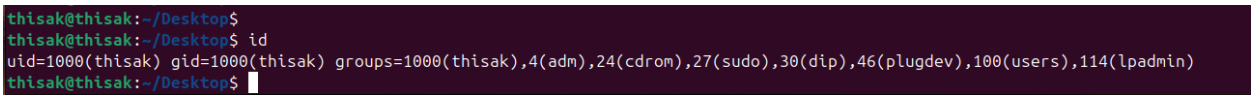
5. rmdir	Removes an empty directory; if the folder contains files, you'll need to delete them first



- **User Info and System Info:**

6. whoami	Displays a list of all users currently logged into the system.

7. id	Shows the current user's UID (User ID) and GID (Group ID) along with group memberships



8. hostnamectl	Provides information about the system's host name, operating system, and kernel, and allows changing the hostname.

```
thisak@thisak:~/Desktop$ hostnamectl
 Static hostname: thisak
       Icon name: computer-vm
         Chassis: vm
      Machine ID: 706ac1607f104380802ae59e3271fbe6
         Boot ID: 39be536c02554a66a07b81fb0a87d8d6
  Virtualization: oracle
Operating System: Ubuntu 24.04.2 LTS
          Kernel: Linux 6.11.0-24-generic
    Architecture: x86-64
 Hardware Vendor: innotek GmbH
  Hardware Model: VirtualBox
Firmware Version: VirtualBox
   Firmware Date: Fri 2006-12-01
    Firmware Age: 18y 4month 2w 2d
```

- **File Operations:**

9. cat	Reads and displays the contents of a file directly to the terminal.

```
thisak@thisak:~/Desktop$ cat test.txt
#include <stdio.h>

int main()
{
        printf("Hello World!");

        return 0;
}
```

10. cp	Copies files or directories from one location to another.

```
thisak@thisak:~/Desktop$ cp test.txt copy.txt
thisak@thisak:~/Desktop$ cat copy.txt
#include <stdio.h>

int main()
{
        printf("Hello World!");

        return 0;
}
```

11. mv	Moves or renames files or directories.

```
thisak@thisak:~/Desktop$ mv copy.txt move.txt
thisak@thisak:~/Desktop$ cat move.txt
#include <stdio.h>

int main()
{
        printf("Hello World!");

        return 0;
}
thisak@thisak:~/Desktop$ cat copy.txt
cat: copy.txt: No such file or directory
```

12. rm	Deletes files or directories permanently; rm -r is used for directories.

```
thisak@thisak:~/Desktop$ cd /home/thisak/Desktop/test
thisak@thisak:~/Desktop/test$ ls
draft.txt
thisak@thisak:~/Desktop/test$ rm draft.txt
thisak@thisak:~/Desktop/test$ ls
thisak@thisak:~/Desktop/test$
```

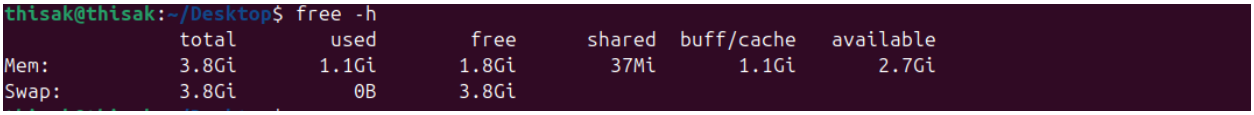
13. vi / nano	Opens a text editor to create or edit file; vi is powerful but has a steeper learning curve, while nano is simpler and beginner-friendly.

```
  GNU nano 7.2                                                  draft.txt
#include <stdio.h>

int main()
{
        printf("Hello world!");
        return 0;
}
```

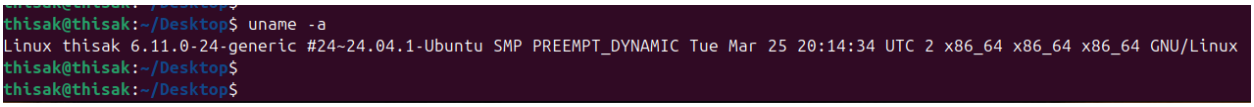
- **System and Memory:**

14. df -h	Shows the available and used disk space on all mounted file systems in a human-readable format.

15. free -h	Displays available and used memory in a human-readable format.



16. uname -a	Prints all the system information, including the kernel name, version, machine type, and OS.



02. DHCP, DNS and NTP services

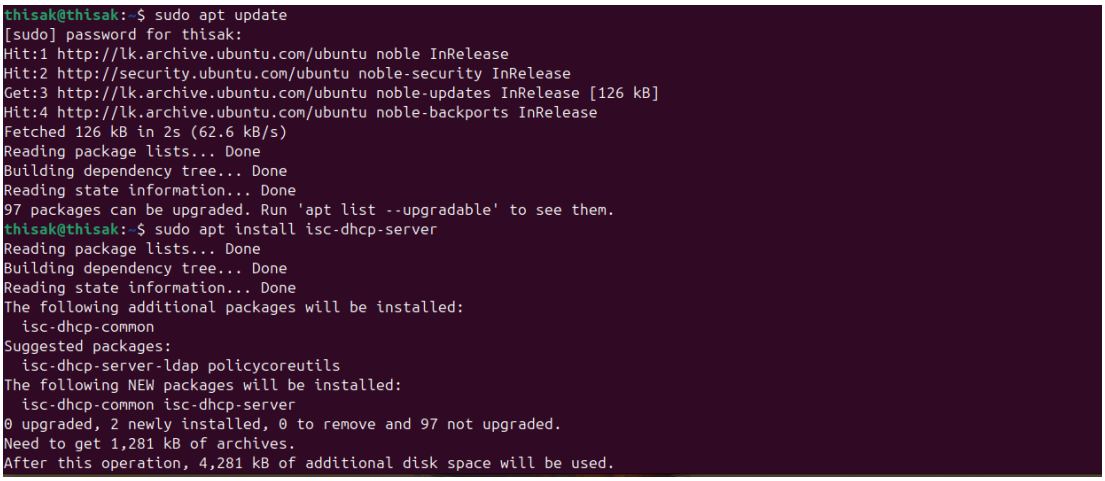
2.1 DHCP Service:

DHCP (Dynamic Host Configuration Protocol) automates the process assigning IP addresses, subnet masks, gateways, and DNS servers to devices in a network. This eliminates the need for manual IP configuration on each device.

Step 01:
Install the DHCP server by executing the following commands:

- *Sudo apt update* (ensures that the latest version of the software is installed)
- *Sudo apt install isc-dhcp-server*

- Sudo provides admin rights for the actions.



Step 02:
Edit the DHCP Configuration file:

- *Sudo nano /etc/dhcp/dhcpd.conf*

- The main config file is opened using the nano text editor.
- Here the following should be defined;
    - Network range
    - Gateway
    - DNS servers
    - Lease times

- Add the following to the end of the config file:

  subnet 192.168.56.0 netmask 255.255.255.0 {
  range 192.168.56.10 192.168.56.100;
  option routers 192.168.56.1;
  option subnet-mask 255.255.255.0;
  option domain-name-servers 8.8.8.8, 8.8.4.4;
  default-lease-time 600;
  max-lease-time 7200;
  }

- Save the configuration and exit the editor.

```
  GNU nano 7.2                    /etc/dhcp/dhcpd.conf *
#    deny members of "foo";
#    range 10.0.29.10 10.0.29.230;
#  }
#}

subnet 192.168.56.0 netmask 255.255.255.0 {
        range 192.168.56.10 192.168.56.100;
        option routers 192.168.56.100;
        option subnet-mask 255.255.255.0;
        option domain-name-servers 8.8.8.8, 8.8.4.4;
        default-lease-time 600;
        max-lease-time 7200;
}
```

Step 03:
Set the network interface for DHCP:

- *Sudo nano /etc/default/isc-dhcp-server*

- The above command opens a file where which interface the DCHP server should listen on is specified.

- Set: INTERFACESv4="enp0s3"

- The above setting tells the server to only listen on interface ‘enp0s3’

```
  GNU nano 7.2                /etc/default/isc-dhcp-server
# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPDv4_PID=/var/run/dhcpd.pid
#DHCPDv6_PID=/var/run/dhcpd6.pid

# 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=""
```

Step 04:
Start and enable the DHCP server:

- *Sudo systemctl restart isc-dhcp-server*
- *Sudo systemctl enable isc-dhcp-server*

- The first command restarts the service to apply the changes while the second command enables to the service to start automatically at boot

```
thisak@thisak:~$ sudo nano /etc/default/isc-dhcp-server
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$ sudo systemctl restart isc-dhcp-server
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$ 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
thisak@thisak:~$
```

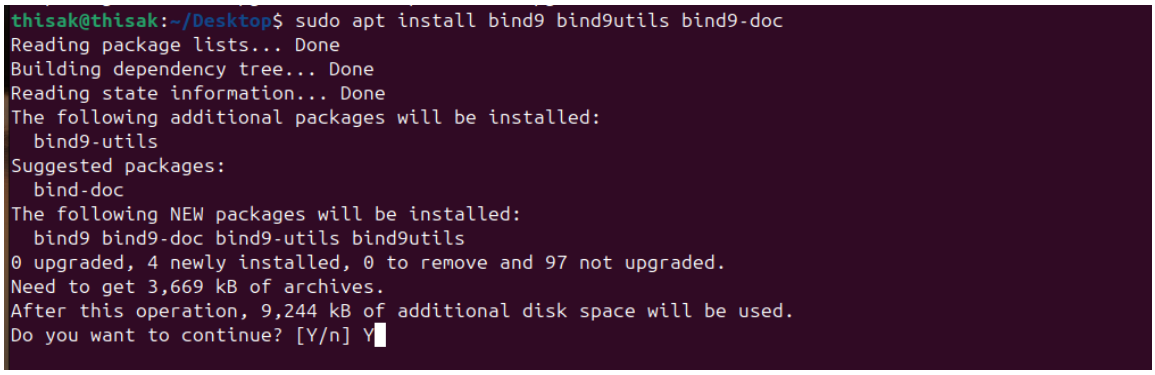
2.2 DNS Service:

The DNS server translates human-readable domain names, for example ‘google.com’ into IP addresses like ‘142.250.190.15’.

Step 01:
Install BIND and DNS utilities by executing the following commands:

- *Sudo apt install bind9 bind9utils bind9-doc*

- ‘*bind9’* is the most widely used DNS server software
- ‘*bind9utils’* are tools to check and manage DNS configurations
- ‘bind9-*doc’* contains all the documentation needed on BIND



Step 02:
Configure the zone files / map files:

- *Sudo nano /etc/bind/named.conf.local*

- This file allows for custom zone definition
- The following commands can be used to add a zone:

```
zone "example.local" {
    type master;
    file "/etc/bind/db.example.local";
};
```

Here, the zone declares the domain (human readable name), the type master being the primary DNS for that domain and the file being the file that stores the DNS records.

```
 GNU nano 7.2                          /etc/bind/named.conf.local
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "example.local"{
        type master;
        file "etc/bind/db.example.local";
};
```

Step 03:
Create and Edit the Zone file:

- *sudo cp /etc/bind/db.local /etc/bind/db.example.local*

- The above command copies a sample zone file to a new file.

- *sudo nano /etc/bind/db.example.local*

- after accessing the new file through the nano text editor, edit the following:

```
$TTL    604800
@       IN      SOA     ns1.example.local. admin.example.local. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns1.example.local.
ns1     IN      A       192.168.56.5
www     IN      A       192.168.56.10
```

This section of the DNS zone file defines how the domain is managed:

- ✓ setting caching rules
- ✓ identifying authoritative server
- ✓ creating mappings so domain names can be converted to IP addresses
- ✓ allowing other devices to and connect to those services on the network

```
 GNU nano 7.2                /etc/bind/db.example.local
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     nsl.example.local. admin.example.local.  (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      nsl.example.local
@       IN      A       192.168.56.5
nsl     IN      A       192.168.56.5
www     IN      A       192.168.56.10
```

Step 04:
Check for errors and restart BIND:

- *sudo named-checkconf*

- The above command checks the global BIND configuration for any syntax errors, and if there is no output, all is well.

- *sudo systemctl restart bind9*

```
thisak@thisak:~/Desktop$ sudo named-checkconf
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$ sudo systemctl restart bind9
[sudo] password for thisak:
thisak@thisak:~/Desktop$
```

- The above command restarts the BIND DNS server so the changes take effect.

Step 05:
Point the VM to use the DNS server:

- *sudo nano /etc/systemd/resolved.conf*

  - Edit or add the line : DNS= *192.168.56.5* and restart the service

- *sudo systemctl restart systemd-resolved*

```
[Resolve]
# Some examples of DNS servers which may be used for DNS= and FallbackDNS=:
# Cloudflare: 1.1.1.1#cloudflare-dns.com 1.0.0.1#cloudflare-dns.com 2606:4700:
# Google:     8.8.8.8#dns.google 8.8.4.4#dns.google 2001:4860:4860::8888#dns.g
# Quad9:      9.9.9.9#dns.quad9.net 149.112.112.112#dns.quad9.net 2620:fe::fe#
#DNS=192.168.1.5
#FallbackDNS=
#Domains=
#DNSSEC=no
#DNSOverTLS=no
#MulticastDNS=no
#LLMNR=no
#Cache=no-negative
#CacheFromLocalhost=no
#DNSStubListener=yes
```

- By running '*nslookup example.local'*, if everything is configured successfully, the IP address for the domain can be obtained.

```
thisak@thisak:~/Desktop$ host nsl.example.local 127.0.0.1
Using domain server:
Name: 127.0.0.1
Address: 127.0.0.1#53
Aliases:

Host nsl.example.local not found: 2(SERVFAIL)
```

```
● named.service - BIND Domain Name Server
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset: enabled)
     Active: active (running) since Thu 2025-04-24 10:29:10 +0530; 2h 30min ago
       Docs: man:named(8)
   Main PID: 3399 (named)
     Status: "running"
      Tasks: 8 (limit: 4609)
     Memory: 5.7M (peak: 6.2M)
        CPU: 217ms
     CGroup: /system.slice/named.service
             └─3399 /usr/sbin/named -f -u bind

Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './NS/IN': 198.97.190.53#53
Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './DNSKEY/IN': 198.97.190.53#53
Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './NS/IN': 199.7.83.42#53
Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './DNSKEY/IN': 199.7.83.42#53
Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './NS/IN': 199.7.91.13#53
Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './DNSKEY/IN': 199.7.91.13#53
Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './NS/IN': 198.41.0.4#53
Apr 24 12:29:10 thisak named[3399]: network unreachable resolving './DNSKEY/IN': 198.41.0.4#53
Apr 24 12:29:10 thisak named[3399]: managed-keys-zone: Unable to fetch DNSKEY set '.': failure
lines 1-21
```

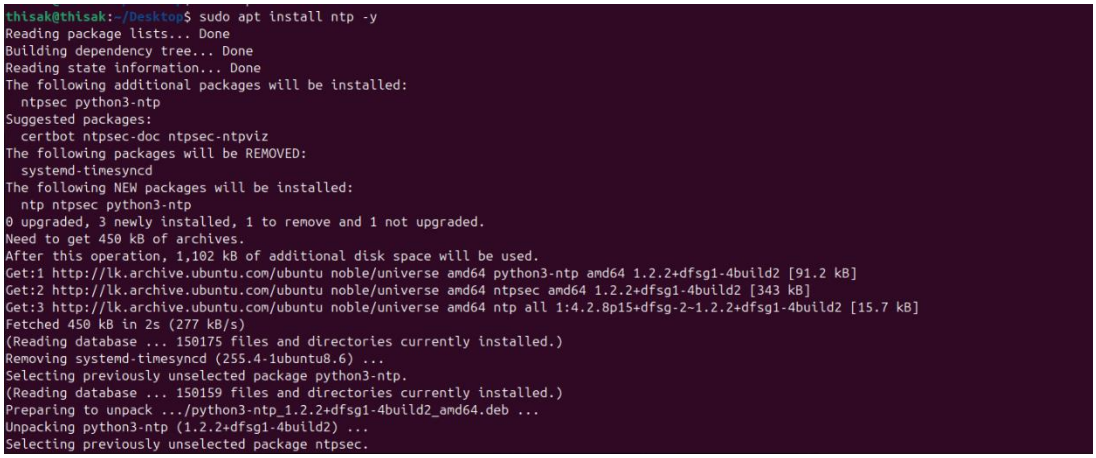
2.3 NTP Service:

The NTP service synchronizes system time with the internet time servers which is essential for logs, certificates and network co-ordination.

Step 01:
Install the NTP service:

- *Sudo apt install ntp*

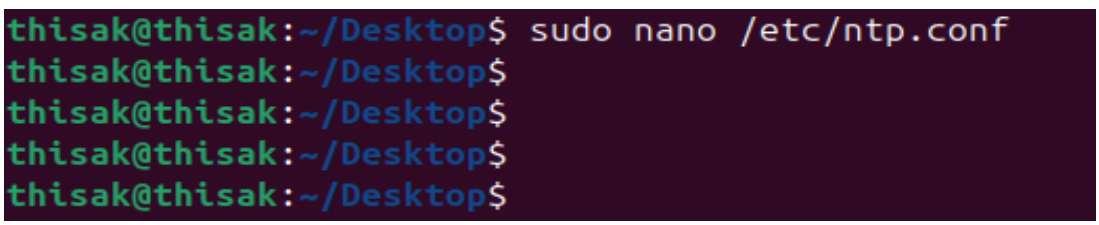
- The '*ntp'* package is a system daemon (background service) that syncs the computer's system with internet time servers.



Step 02:
Edit the NTP service:

- *Sudo nano /etc/ntp.conf*

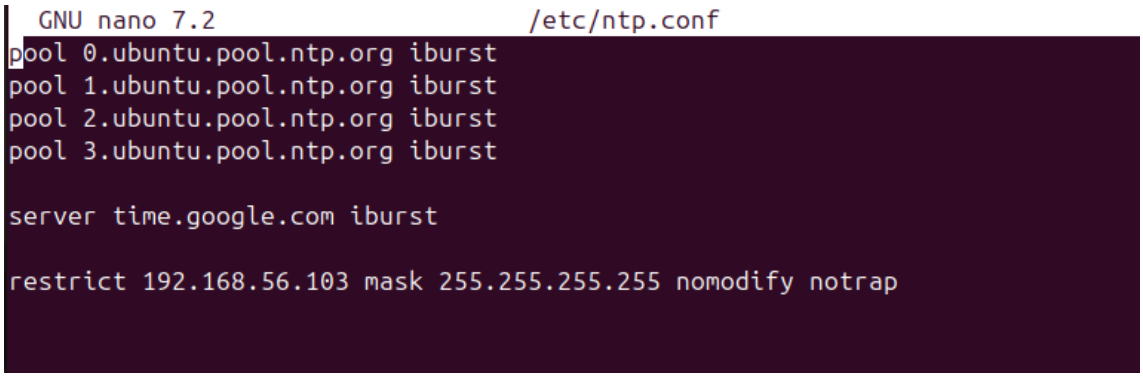
- By default NTP will use its built-in server list to synchronize time. However we can use the above command to edit the NTP configuration file to specify custom NTP's as needed.



Step 03:
Add or change time servers:

- *server time.google.com 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*

- The above command tells the NTP client to use Google's public time server as the source.
- The 'iburst' speeds up the time synching process when connecting initially.
- Some other NTP servers that can be used are;
    - Time.nist.gov
    - 0.pool.ntp.org.



Step 04:
Restart the NTP service:

- *Sudo systemctl restart ntp*

- By restarting the changes done to the ntp.conf file will take place

- *Sudo systemctl enable ntp*

- The above command makes sure that the service automatically start at boot.

- A good recommendation is to check the status of the NTP just to make sure that theCorrect file is named when enabling the service.

  - *Sudo systemctl status ntp.service*

```
thisak@thisak:~/Desktop$ systemctl status ntp.service
● ntpsec.service - Network Time Service
     Loaded: loaded (/usr/lib/systemd/system/ntpsec.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-04-22 19:44:18 +0530; 4min 19s ago
       Docs: man:ntpd(8)
    Process: 21050 ExecStart=/usr/libexec/ntpsec/ntp-systemd-wrapper (code=exited, status=0/SUCCESS)
   Main PID: 21053 (ntpd)
      Tasks: 1 (limit: 4609)
     Memory: 10.5M (peak: 11.0M)
        CPU: 201ms
     CGroup: /system.slice/ntpsec.service
             └─21053 /usr/sbin/ntpd -p /run/ntpd.pid -c /etc/ntpsec/ntp.conf -g -N -u ntpsec:ntpsec

Apr 22 19:48:35 thisak ntpd[21053]: DNS: Pool skipping: 222.165.180.134
Apr 22 19:48:35 thisak ntpd[21053]: DNS: Pool skipping: 162.159.200.1
Apr 22 19:48:35 thisak ntpd[21053]: DNS: Pool skipping: 162.159.200.123
Apr 22 19:48:35 thisak ntpd[21053]: DNS: dns_take_status: 0.ubuntu.pool.ntp.org=>good, 8
Apr 22 19:48:36 thisak ntpd[21053]: DNS: dns_probe: 1.ubuntu.pool.ntp.org, cast_flags:8, flags:101
Apr 22 19:48:36 thisak ntpd[21053]: DNS: dns_check: processing 1.ubuntu.pool.ntp.org, 8, 101
Apr 22 19:48:36 thisak ntpd[21053]: DNS: Pool skipping: 162.159.200.1
Apr 22 19:48:36 thisak ntpd[21053]: DNS: Pool skipping: 162.159.200.123
Apr 22 19:48:36 thisak ntpd[21053]: DNS: Pool skipping: 222.165.180.134
Apr 22 19:48:36 thisak ntpd[21053]: DNS: dns_take_status: 1.ubuntu.pool.ntp.org=>good, 8
```

```
thisak@thisak:~/Desktop$ sudo systemctl enable ntpsec
Synchronizing state of ntpsec.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ntpsec
thisak@thisak:~/Desktop$
```

Step 05:

Check if the NTP service is working or not:

  - *ntpq -p*

- The above command displays a list of NTP servers the system is connected to and how well they're synching.

  - *timedatectl*

```
thisak@thisak:~/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-5.ntp1.ps5.canonical.com         79.243.60.50     2 u   15   64  177 172.0852  -9.1077   1.8802
+time.cloudflare.com                       10.35.0.7        3 u    5   64  177  46.5441  -2.0266   1.8932
+time.cloudflare.com                       10.121.8.10      3 u    7   64  177  39.5176   0.2218   1.7098
*ntp.sltidc.lk                            216.239.35.12    2 u    9   64  177   7.9021   2.8401   2.1898
 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
thisak@thisak:~/Desktop$
```

- The above command displays the time status, time zone, and whether the time is synchronized

```
thisak@thisak:~/Desktop$ timedatectl
               Local time: Thu 2025-04-24 09:45:26 +0530
           Universal time: Thu 2025-04-24 04:15:26 UTC
                 RTC time: Thu 2025-04-24 04:09:17
                Time zone: Asia/Colombo (+0530, +0530)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

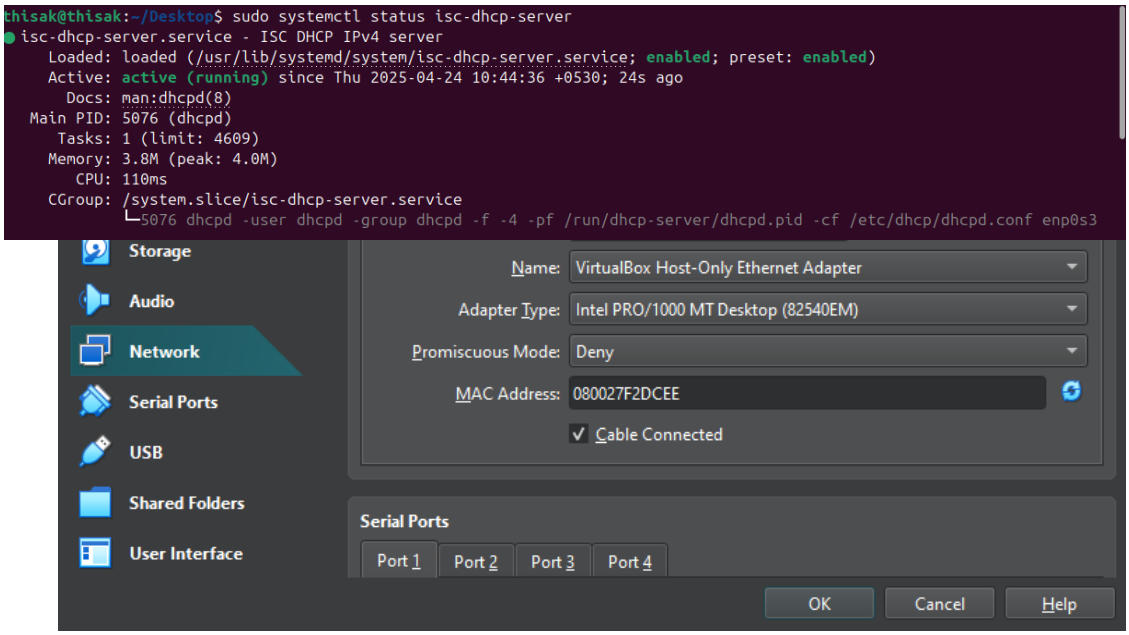
2.4 Simulating a small network environment with DHCP, DNS & NTP:

In this section the necessary steps that needs to be taken to successfully simulate a small virtual environment and to configuring to obtain an IP address automatically from the server will be discussed.

Step 01:
On the server VM (the Ubuntu distro) make sure the config files and the DHCP interface contain the commands explained in section 2.1.

Step 02:
In VirtualBox, configure both the server and client VMs to use the same internal network by;

- Going to network settings in both VMs.
- Setting adapter 1 to 'Host only adapter' and making sure the name is the same as well.
- This creates a small network environment within the Virtual machines.



Step 03:

On the client VM (in this instance, Kali Linux), see what IP address is assigned to the client from the DHCP server.

Apply the same command to get the IP address from the server VM as well

- *ip a*

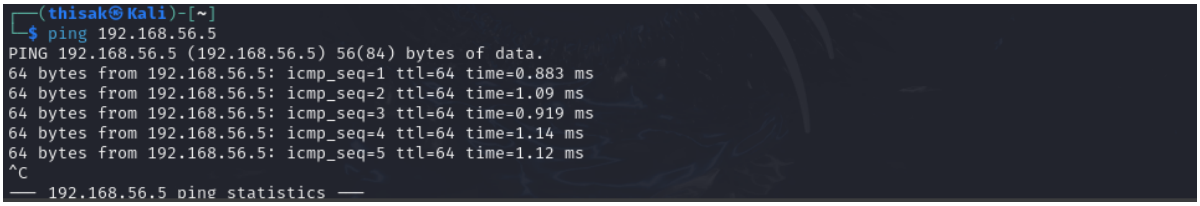
```
thisak@thisak:~/Desktop$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:cd:25:6d brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.5/24 scope global enp0s3
       valid_lft forever preferred_lft forever
```

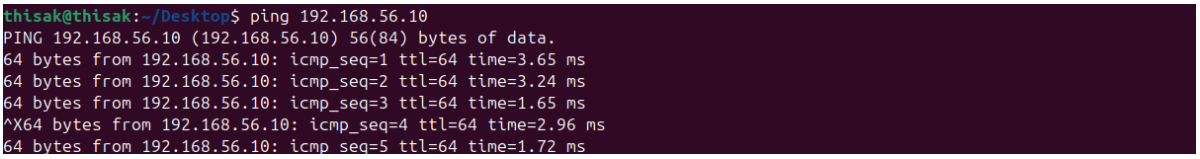
```
thisak@Kali: ~
File  Actions  Edit  View  Help
ping: connect: Network is unreachable

┌──(thisak㉿Kali)-[~]
└─$ 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: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:f2:dc:ee brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.10/24 brd 192.168.56.255 scope global dynamic noprefixroute eth0
       valid_lft 523sec preferred_lft 523sec
    inet6 fe80::a00:27ff:fef2:dcee/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Step 04:

Now the connectivity can be checked by pinging the assigned IP addresses of the server and client virtual machines.

- *Ping 192.168.56.5* (Ubuntu – server VM)
- *Ping 192.168.56.10* (Kali – client VM)


┌──(thisak㉿Kali)-[~]
└─$ ping 192.168.56.5
PING 192.168.56.5 (192.168.56.5) 56(84) bytes of data.
64 bytes from 192.168.56.5: icmp_seq=1 ttl=64 time=0.883 ms
64 bytes from 192.168.56.5: icmp_seq=2 ttl=64 time=1.09 ms
64 bytes from 192.168.56.5: icmp_seq=3 ttl=64 time=0.919 ms
64 bytes from 192.168.56.5: icmp_seq=4 ttl=64 time=1.14 ms
64 bytes from 192.168.56.5: icmp_seq=5 ttl=64 time=1.12 ms
^C
── 192.168.56.5 ping statistics ──


thisak@thisak:~/Desktop$ ping 192.168.56.10
PING 192.168.56.10 (192.168.56.10) 56(84) bytes of data.
64 bytes from 192.168.56.10: icmp_seq=1 ttl=64 time=3.65 ms
64 bytes from 192.168.56.10: icmp_seq=2 ttl=64 time=3.24 ms
64 bytes from 192.168.56.10: icmp_seq=3 ttl=64 time=1.65 ms
^X64 bytes from 192.168.56.10: icmp_seq=4 ttl=64 time=2.96 ms
64 bytes from 192.168.56.10: icmp_seq=5 ttl=64 time=1.72 ms

2.5 The need and use of DHCP, DNS and NTP services in network administration:

2.5.1 DHCP service:

- Manually configuring unique IP addresses for each device is inefficient, error-prone and impractical in large scale environments. With the introduction of the DHCP service the allocation of IP addresses, subnet masks, gateways and server information has become automated, effectively reducing administrative overhead.

- DHCP allows for centralized control over address assignment to ensure that IP conflicts are not met.

- The DHCP service also facilitates DHCP reservations to assign fixed IP addresses to specific devices like servers and printers.

2.5.2 DNS service:

- While devices use IP addresses to communicate, remembering each and every address is a tedious task. The DNS server aids by translating human-readable domain names into IP addresses, allowing for easier network navigation.

- The DNS service makes accessing internal and external websites user-friendly.

- This service also facilitates network troubleshooting by identifying and resolving DNS related issues.

2.5.3 NTP service:

- The NTP service synchronizes the clocks of all network devices with a trusted time source; either from the internet or a local NTP server. Accurate time synchronization is much needed for;
    - Log file accuracy
    - Certificate validation
    - Distributed systems

- The NTP server also makes sure to avoid issues with time-sensitive operations like security token expirations, cron jobs as well as database transactions.

03. Security and Other Services

3.1 Shell scripting:

A shell script is a set of instructions written in a file so that the computer is able to execute the commands one by one. A bash script is a type of shell script for Linux distros, and within this script the following tasks will be implemented:

- Deleting log files older than 7 days
- Archiving the remaining by compressing them
- Provide a summary of the actions

Step 01:

Create custom logs in a new directory named *'/var/log/custom_logs'* and set up a few dummy files.

- *sudo mkdir /var/log/custom_logs*
- *sudo touch /var/log/custom_logs/test.log*

```
thisak@thisak:~$ sudo mkdir /var/log/custom_logs
thisak@thisak:~$ sudo touch /var/log/custom_logs/test.log
thisak@thisak:~$ sudo touch /var/log/custom_logs/test01.log
thisak@thisak:~$ sudo touch /var/log/custom_logs/test02.log
thisak@thisak:~$ sudo touch /var/log/custom_logs/test03.log
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$
thisak@thisak:~$ cd /var/log/custom_logs
thisak@thisak:/var/log/custom_logs$ ls
test01.log  test02.log  test03.log  test.log
```

Step 02:

Create a new script file (.sh file) to write the bash script.

- *sudo nano /var/log/log_cleanup.sh*

The script is as follows:

```
#!/bin/bash
# This script deletes log files older than 7 days, archives the rest, and prints a summary.

# Directory where log files are stored
LOG_DIR="/var/log/custom_logs"

# Backup file name with timestamp (optional enhancement)
BACKUP_FILE="/var/log/logs_backup.tar.gz"
```

```
# Find and delete log files older than 7 days
echo "Deleting log files older than 7 days..."
DELETED_FILES=$(find "$LOG_DIR" -name "*.log" -type f -mtime +7 -print -delete)

# Archive remaining log files
echo "Archiving remaining log files..."
tar -czf "$BACKUP_FILE" "$LOG_DIR"/*.log 2>/dev/null

# Print summary
echo "Summary:"
echo "Deleted files:"
echo "$DELETED_FILES"
echo "Archived files:"
ls "$LOG_DIR"/*.log 2>/dev/null

echo "Backup saved as $BACKUP_FILE"
```

Script explanation:

- The **LOG_DIR** and **BACKUP_FILE** are variables that holds the full path and name of the files to be deleted and backed-up respectively.
- The **echo** commands are used to display text on the terminal.
- The **DELETED_FILES** variable is used to store the result of the command that is within the **$()**.
- The **find** command is used to search for files and folders, and since the LOG_DIR is attached, that directory is the one to be searched.
- -**name "*.log**" looks for files ending with .log.
- **-type f** command makes sure to only find the files not folders.
- -**mtime +7** specifies to find files that were modified more than 7 days ago.
- -**print** command outputs the names of the found files while the –delete command deletes said files.

- Tar is a tool to collect files into one single file.
- The **–czf** option, creates a new **archive (c),** compresses it using **gzip (z)** and use the **filename** specified by user **(f).**
- The **2 >/dev/null** command hides error messages.
- The **ls** command lists the files matching the requirements.

```
  GNU nano 7.2                                        /var/log/log_cleanup.sh
#! /bin/bash

#This bash script does the following tasks:
        # 1. delete log files older than 7 days in the /var/log/custom_logs directory
        # 2. Archive the remaining log files into  a compressed .tar.gz file
        # 3. Prints a summary of the deleted and archived files

# setting the directory containing the log files
LOG_DIR="/var/log/custom_logs"

# setting the backup archive file name and location
BACKUP_FILE="/var/log/logs_backup.tar.gz"

# commands to implement task 1
echo "Deleting log files older than 7 days from $LOG_DIR"
DELETED_FILES=$(find "$LOG_DIR" -name "*.log" -type -f -mtime +7 -print -delete)

# commands to implement task 2
echo "Archiving remanining log files"
tar -czf "$BACKUP_FILE" "$LOG_DIR"/*.log 2>/dev/null

#commands to implement task 3
echo "Summary:"
echo "--------------------------------------------"
echo "Deleted files:"
echo "$DELETED_FILES"
echo "--------------------------------------------"
echo "Archived files:"
ls "$LOG_DIR"/*.log 2>/dev/null
echo "--------------------------------------------"
echo "Backup saved as $BACKUP_FILE"
```

Step 03:
Provide permission to run the script via chmod.

- *sudo chmod +x /usr/local/bin/log_cleanup.sh*

Step 04:
Test the script manually.

- *sudo /usr/local/bin/log_cleanup.sh*

```
thisak@thisak:~$ sudo /var/log/log_cleanup.sh
Deleting log files older than 7 days from /var/log/custom_logs
Archiving remanining log files
Summary:
--------------------------------------------
Deleted files:

--------------------------------------------
Archived files:
/var/log/custom_logs/test02.log  /var/log/custom_logs/test03.log  /var/log/custom_logs/test.log
--------------------------------------------
Backup saved as /var/log/logs_backup.tar.gz
```

(There are no deleted files as there aren't any log files that are older than 7 days.)

Step 05:

Schedule the script using the cron job editor.

- *sudo crontab –e*

- Select an editor option from the given list [1] – [3].

```
thisak@thisak:~$ sudo crontab -e
no crontab for root - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.tiny
  3. /bin/ed

Choose 1-3 [1]: 1
```

- Add the following command to the end of the file.

- *0 0 * * 0 /usr/local/bin/log_cleanup.sh*

```
  GNU nano 7.2                                          /tmp/crontab.3AOKeq/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command

0 0 * * 0 /var/log/log_cleanup.sh

# Minute 0, Hour 0, Anby day, any month, Day 0 (sunday) is what is implemented
```

The above command specifies that the script '/usr/local/bin/log_cleanup.sh' will run at minute 0, hour 0 on every Sunday.

3.2 SSH (Secure Shell):

SSH (Secure Shell) is a widely-used network protocol that provides a secure, encrypted channel for remote login and command execution over an unsecured network.

Step 01:
Use the update and install commands to install the SSH service.

- *Sudo apt update*
- *Sudo apt install openssh-server*

```
thisak@thisak:~/Desktop$ 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 31 not upgraded.
Need to get 832 kB of archives.
After this operation, 6,743 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://lk.archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-sftp-server amd64 1:9.6p1-3ubuntu13.11 [37.3 kB]
Get:2 http://lk.archive.ubuntu.com/ubuntu noble-updates/main amd64 openssh-server amd64 1:9.6p1-3ubuntu13.11 [509 kB]
Get:3 http://lk.archive.ubuntu.com/ubuntu noble/main amd64 ncurses-term all 6.4+20240113-1ubuntu2 [275 kB]
Get:4 http://lk.archive.ubuntu.com/ubuntu noble-updates/main amd64 ssh-import-id all 5.11-0ubuntu2.24.04.1 [10.1 kB]
Fetched 832 kB in 3s (328 kB/s)
```

Step 02:
Verify that the SSH server (sshd daemon) is running properly.

- *sudo systemctl status ssh*

```
thisak@thisak:~/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 Thu 2025-05-01 15:17:48 +0530; 22s ago
TriggeredBy: ● ssh.socket
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 4332 (sshd)
      Tasks: 1 (limit: 4609)
     Memory: 1.2M (peak: 1.5M)
        CPU: 48ms
     CGroup: /system.slice/ssh.service
             └─4332 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

May 01 15:17:48 thisak systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
May 01 15:17:48 thisak sshd[4332]: Server listening on :: port 22.
May 01 15:17:48 thisak systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
```

Step 03:
Obtain the IP address of the VM to connect to it from a different machine.

- *ip a*

```
thisak@thisak:~/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,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:cd:25:6d brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 brd 192.168.56.255 scope global dynamic noprefixroute enp0s3
       valid_lft 435sec preferred_lft 435sec
    inet6 fe80::a00:27ff:fecd:256d/64 scope link
       valid_lft forever preferred_lft forever
```

Step 04:
It is recommended to configure ssh server settings to make sure that the server is working in order. After accessing the config file, make sure the port numbers, logins and permissions are in order.
Restart the server after making any changes in order to implement them.

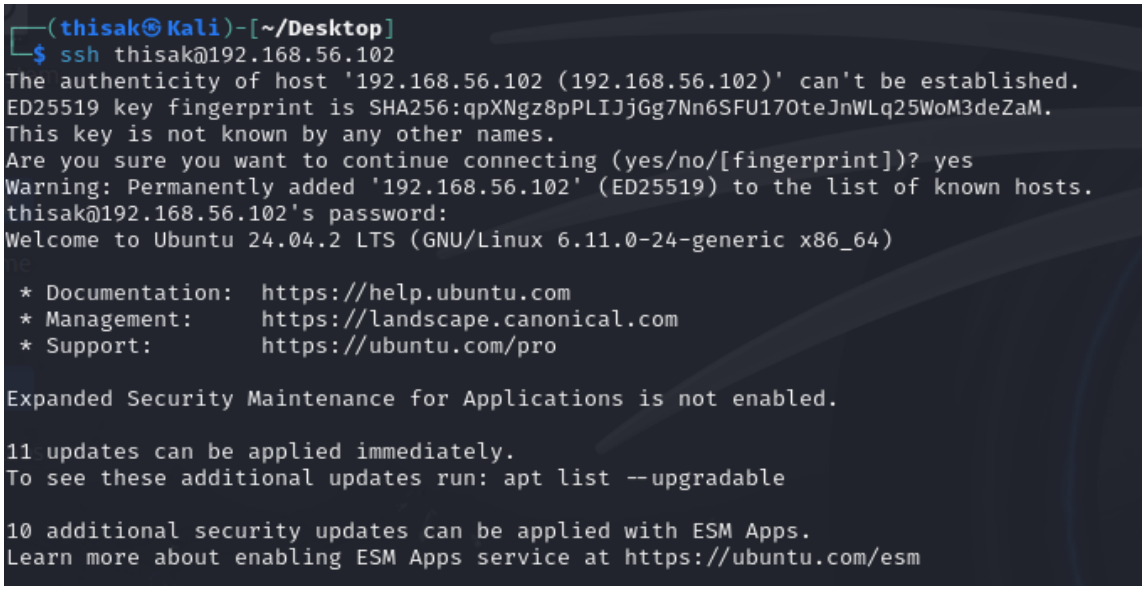
- *Sudo nano /etc/ssh/sshd_config*
- *Sudo systemctl restart ssh*

```
  GNU nano 7.2                                    /etc/ssh/sshd_config

# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

#Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin no
#StrictModes yes
```

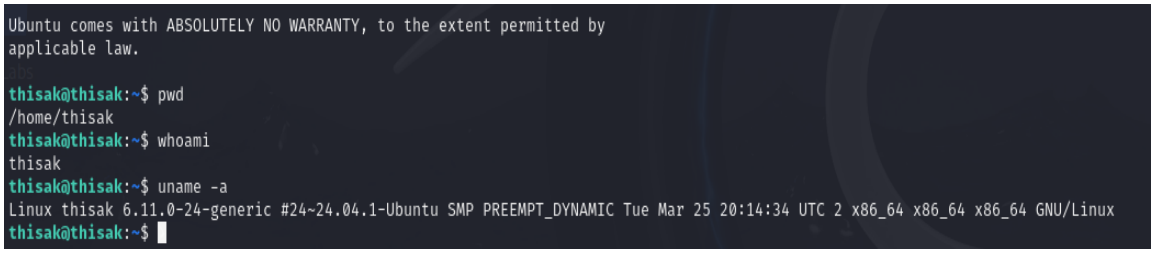
Step 05:

Connect to Ubuntu VM via ssh from another machine (this instance kali VM).

- ssh thisak@192.168.56.102



- After successful connection, the client machine will ask the password of the machine to be connected.
- When the correct password is entered, access to the Ubuntu terminal will be granted via the kali terminal.



3.3 iptables:

iptables is a packet filtering firewall built into the Linux kernel. It allows to define rules about what network traffic is allowed or denied based on: protocol, Port number, IP address and Network interface.

Step 01:

Check the existing iptables in the VM.

- *Sudo iptables –L –n -v*

```
thisak@thisak:~/Desktop$ sudo iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

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
```

Step 02:

Blocking social media can be done using IP addresses or domain name resolution. Since IP addresses often change in social media platforms, using domain names is recommended.

- *sudo iptables -A OUTPUT -m string --algo bm --string "facebook.com" -j DROP*
- *sudo iptables -A OUTPUT -m string --algo bm --string "instagram.com" -j DROP*
- *sudo iptables -A OUTPUT -m string --algo bm --string "twitter.com" -j DROP*

- The **'-A OUTPUT'** command handles outgoing traffic from the machine.
- The **'-m string'** command matches traffic based on a string pattern in the packet payload.
- The **'- - algo bm'** command specifies the string matching algorithm; in this case the Boyer-Moore algorithm.
- The **'-j DROP'** command is used to silently discard the packet without replying or forwarding it.

```
thisak@thisak:~/Desktop$ sudo iptables -A OUTPUT -m string --algo bm --string "facebook.com" -j DROP
thisak@thisak:~/Desktop$ sudo iptables -A OUTPUT -m string --algo bm --string "instagram.com" -j DROP
thisak@thisak:~/Desktop$ sudo iptables -A OUTPUT -m string --algo bm --string "twitter.com" -j DROP
thisak@thisak:~/Desktop$ sudo iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

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

Chain OUTPUT (policy ACCEPT 128 packets, 10048 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       0    --  *      *       0.0.0.0/0            0.0.0.0/0           STRING match  "facebook.com" ALGO name bm
    0     0 DROP       0    --  *      *       0.0.0.0/0            0.0.0.0/0           STRING match  "instagram.com" ALGO name bm
    0     0 DROP       0    --  *      *       0.0.0.0/0            0.0.0.0/0           STRING match  "twitter.com" ALGO name bm
```

Step 02:

To allow only secure web browsing (HTTPS) and block unsecure web browsing (HTTP), allowing outgoing connections to port 443 should be enabled and blocking outgoing connections to port 80 should be implemented.

- *sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT*
- *sudo iptables -A OUTPUT -p tcp --dport 80 -j DROP*

- The **'p –tcp'** command specifies the protocol the rule applies to; in this case TCP
- The **'- -dport'** command matches packets destined for the port

- *sudo iptables -A OUTPUT -p udp --dport 53 -j ACCEPT*

- The above command allows DNS traffic to resolve freely.

- *sudo iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT*

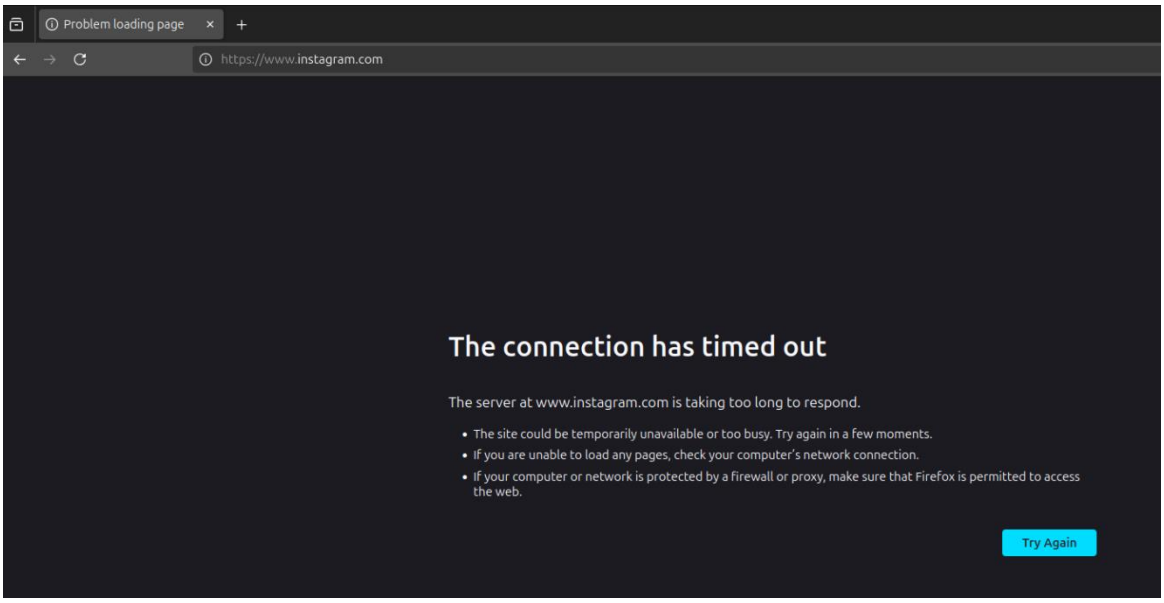
- The above command allows already existing connections and packets related those existing connections to exist without blocking.

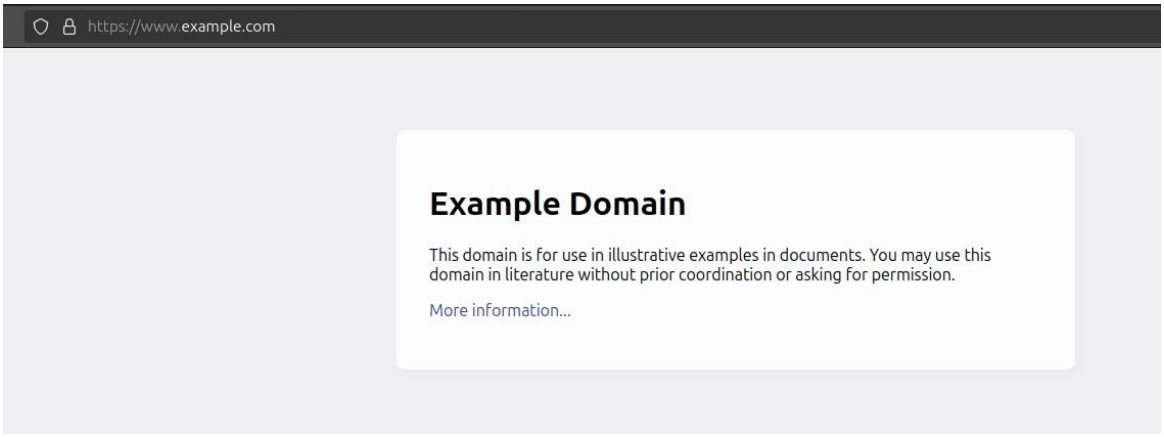
- *sudo iptables -A OUTPUT -j DROP*

- The above command block any other traffic present.

```
thisak@thisak:~/Desktop$ sudo iptables -L -v --line-numbers
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination

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

Chain OUTPUT (policy ACCEPT 1402 packets, 109K bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 DROP       all  --  any    any     anywhere             anywhere             STRING match  "facebook.com" ALGO name bm
2        0     0 DROP       all  --  any    any     anywhere             anywhere             STRING match  "instagram.com" ALGO name bm
3        0     0 DROP       all  --  any    any     anywhere             anywhere             STRING match  "twitter.com" ALGO name bm
4        0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:https
5        0     0 DROP       tcp  --  any    any     anywhere             anywhere             tcp dpt:http
6      131  9177 ACCEPT     udp  --  any    any     anywhere             anywhere             udp dpt:domain
7       52  4585 ACCEPT     all  --  any    any     anywhere             anywhere             state RELATED,ESTABLISHED
8        0     0 DROP       all  --  any    any     anywhere             anywhere
thisak@thisak:~/Desktop$
```

- Dropped connection and HTTPS accepted connection:





3.4 Web Server (Apache):

Popular web servers like Apache and Nginx are widely used due to their reliability, flexibility, and ease of configuration. They support web content such as HTML files, images, and other resources requested by a user's web browser.

Step 01:

Update the system and install apache server.

- *Sudo apt update*
- *Sudo apt install apache2 -y*

```
thisak@thisak:~/Desktop$ sudo apt install apache2 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom ufw
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
0 upgraded, 8 newly installed, 0 to remove and 26 not upgraded.
Need to get 1,900 kB of archives.
```

Step 02:

Verify that the server is up and running.

- *Sudo systemctl status apache2*

```
thisak@thisak:~/Desktop$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Thu 2025-05-01 18:27:54 +0530; 6min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 9363 (apache2)
      Tasks: 55 (limit: 4609)
     Memory: 5.3M (peak: 5.5M)
        CPU: 181ms
     CGroup: /system.slice/apache2.service
             ├─9363 /usr/sbin/apache2 -k start
             ├─9365 /usr/sbin/apache2 -k start
             └─9366 /usr/sbin/apache2 -k start

May 01 18:27:54 thisak systemd[1]: Starting apache2.service - The Apache HTTP Server...
May 01 18:27:54 thisak apachectl[9362]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the
May 01 18:27:54 thisak systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Step 03:

Verify the Apache server listening ports to see what ports the server is bound to.

- *Sudo ss –tuln | grep 80*

```
thisak@thisak:~/Desktop$ sudo ss -tuln | grep 80
udp   UNCONN 0      0      [fe80::a00:27ff:fecd:256d]%enp0s3:53            [::]:*
udp   UNCONN 0      0      [fe80::a00:27ff:fecd:256d]%enp0s3:53            [::]:*
udp   UNCONN 0      0            [fd00::78ff:980a:8fb:db98]:53            [::]:*
udp   UNCONN 0      0            [fd00::78ff:980a:8fb:db98]:53            [::]:*
udp   UNCONN 0      0      [fe80::a00:27ff:fecd:256d]%enp0s3:123           [::]:*
udp   UNCONN 0      0            [fd00::78ff:980a:8fb:db98]:123           [::]:*
tcp   LISTEN 0      511                                    *:80              *:*
tcp   LISTEN 0      10     [fe80::a00:27ff:fecd:256d]%enp0s3:53            [::]:*
tcp   LISTEN 0      10     [fe80::a00:27ff:fecd:256d]%enp0s3:53            [::]:*
tcp   LISTEN 0      10           [fd00::78ff:980a:8fb:db98]:53            [::]:*
tcp   LISTEN 0      10           [fd00::78ff:980a:8fb:db98]:53            [::]:*
thisak@thisak:~/Desktop$
```

Step 04:

The next step is to create sample web page (index.html) to verify that the server is properly responding.

- *Cd /var/www/html*

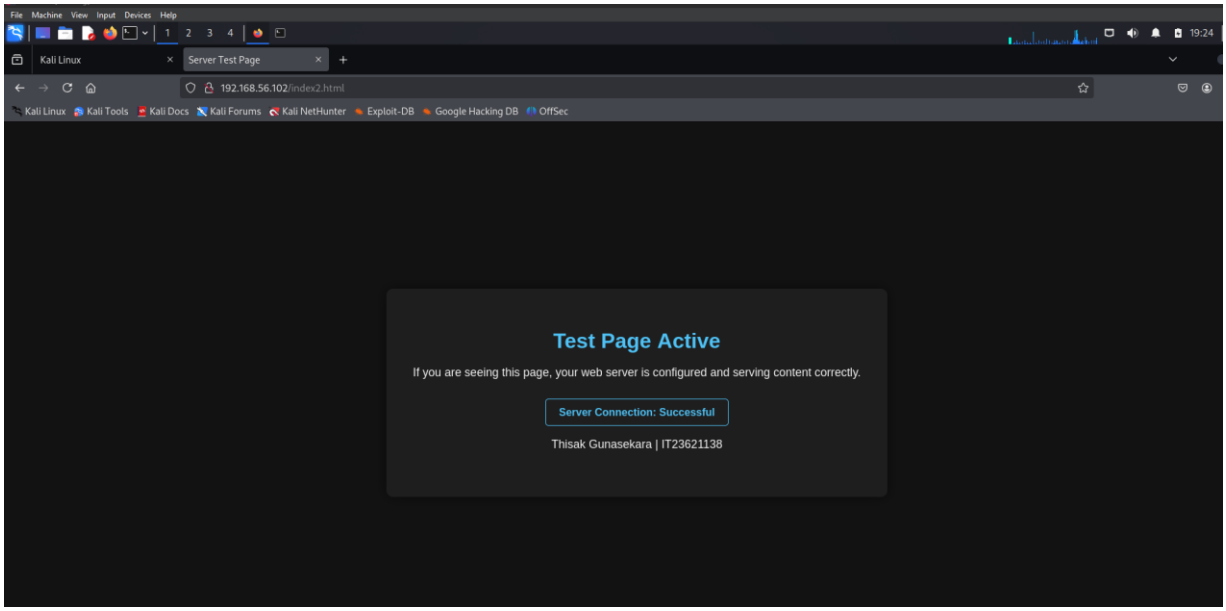
- The above command allows access to Apache's default web root.
- It is recommended to create a new index.html file by removing the existing file in the directory and creating one of your own.

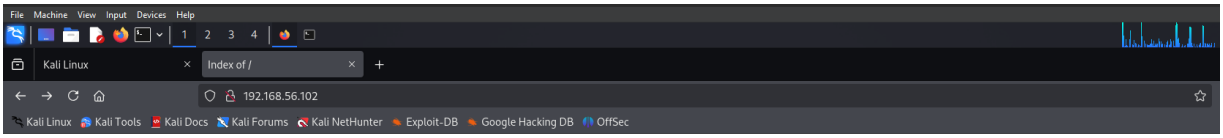
- *sudo nano /var/www/html/index.html*

```
thisak@thisak: ~/Desktop
thisak@thisak:~/Desktop$ sudo cp index.html /var/www/index.html
thisak@thisak:~/Desktop$ sudo cat /var/www/index.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Server Test Page</title>
    <style>
        body {
            background-color: #121212;
            color: #f5f5f5;
            font-family: 'Arial', sans-serif;
            margin: 0;
            padding: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }
```

- After creating the index.html file, restart the apache server, and access the webpage via a client VM (kali machine) with the ip address of the Host VM.

  - *http://192.168.56.102*





Index of /

Name	Last modified	Size	Description
index2.html	2025-05-01 19:17	1.3K	

Apache/2.4.58 (Ubuntu) Server at 192.168.56.102 Port 80

3.5 Email Server (Postfix):

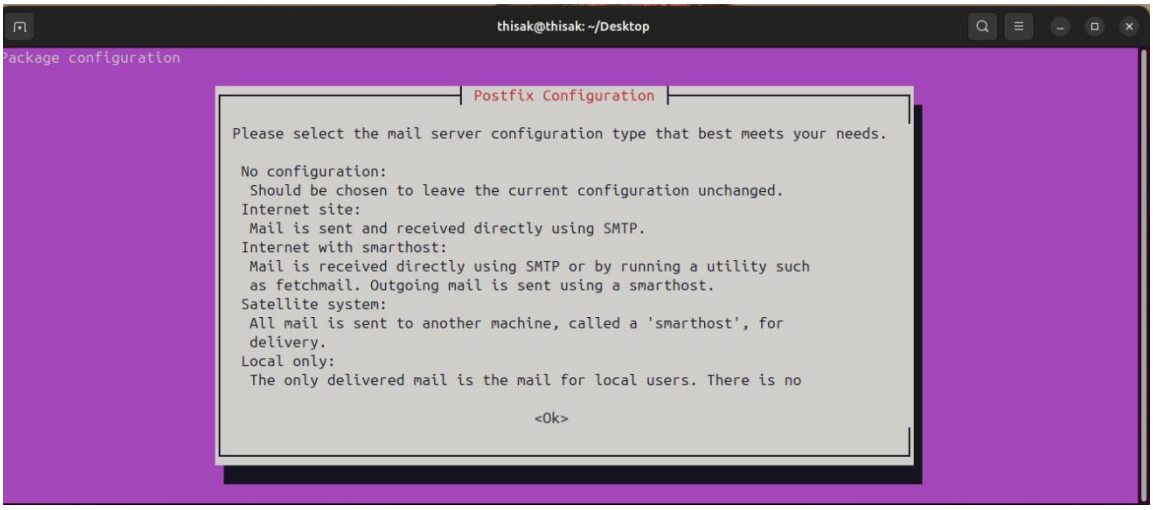
Among the various email server software available, Postfix is one of the most popular choices for Unix-like systems due to their reliability, performance, and ease of configuration.

Step 01:

Update the system and install postfix.

- *sudo apt update*
- *Sudo apt install postfix*

```
thisak@thisak:~/Desktop$ sudo apt install postfix
[sudo] password for thisak:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libnsl2
Suggested packages:
  mail-reader postfix-cdb postfix-doc postfix-ldap postfix-lmdb postfix-mta-sts-resolver postfix-mysql postfix-pcre postfix-pgsql
  postfix-sqlite procmail sasl2-bin | dovecot-common ufw
```



Step 02:

Change initial configurations of the postfix service.

- Configure the general type of mail to be ‘Local only’.
- Change the system mail name to be the VM’s hostname or type ‘localhost’

Step 03:
Test the server by creating a dummy user and sending that user a simple mail.

- Create a new user in the VM to send the test mail:
  - *Sudo adduser testuser*

- When prompted, set a password and create the user.

- From the current user, send a mail to test user using the following command
  - *echo "Hello testuser, this is a local mail test!" | mail -s "Test Mail" testuser*

```
thisak@thisak:~/Desktop$ sudo adduser testuser
info: Adding user `testuser' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `testuser' (1001) ...
info: Adding new user `testuser' (1001) with group `testuser (1001)' ...
info: Creating home directory `/home/testuser' ...
info: Copying files from `/etc/skel' ...
New password:
BAD PASSWORD: The password fails the dictionary check - it is too simplistic/systematic
Retype new password:
Sorry, passwords do not match.
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for testuser
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
info: Adding new user `testuser' to supplemental / extra groups `users' ...
info: Adding user `testuser' to group `users' ...
```

Step 04:
Switch to the test user to verify that the mail has been delivered.

- *Sudo su - testuser*

- After getting into the test user account, enter ‘mail’ command to see if there has been any mail received.

```
thisak@thisak:~/Desktop$ sudo su- testuser
[sudo] password for thisak:
sudo: su-: command not found
thisak@thisak:~/Desktop$ sudo su - testuser
testuser@thisak:~$ mail
"/var/mail/testuser": 1 message 1 unread
>U   1 thisak              Thu May  1 19:42  16/520   test mail
?
```

04. Linux GDB

This section documents the analysis of an executable file focusing on how it executes, the debugging of the file, file system interactions and the analysis of data files to understand the functionality of the program, it's interaction with the system and how it affects local files.

1. **Execution process:**

Step 01:
To select the correct executable file, first check the system architecture using the following command.

- *Uname –m*

Step 02:
From the downloaded executable files, select the one that matches the VM system architecture.

```
thisak@thisak:~/Desktop$ uname -m
x86_64
thisak@thisak:~/Desktop$ cd /home/thisak/Downloads
thisak@thisak:~/Downloads$ ls
Executables  Executables.zip
thisak@thisak:~/Downloads$ cd Executables
thisak@thisak:~/Downloads/Executables$ ls
Executables  __MACOSX
```

Step 03:
After selecting the correct executable file, change file permissions for it to be executable and then run the program with sudo in the current directory that the file is in.

- *Chmod +x x86_64*
- *Sudo ./x86_64*

- Input the IT number when prompted.

```
thisak@thisak:~/Downloads/Executables/Executables$ chmod +x x86_64
thisak@thisak:~/Downloads/Executables/Executables$ sudo ./x86_64
Enter the student IT number: IT23621138
```

Step 04:
After an executable file with the IT number has been created, execute the program using sudo, and see what the output is using the 'ls' command in the current directory.

- *Sudo ./IT23621138*

```
thisak@thisak:~/Downloads/Executables/Executables$ sudo ./IT23621138
thisak@thisak:~/Downloads/Executables/Executables$ ls
ARM  data.txt  IT23621138
thisak@thisak:~/Downloads/Executables/Executables$
```

2. Debugging process:

Step 01:

To start the debugging process, open the executable file with the IT number in GDB.

- gdb ./IT23621138

```
thisak@thisak:~/Downloads/Executables/Executables$ gdb ./IT23621138
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./IT23621138...
(gdb)
```

Step 02:

To see what functions are present in the executable file, enter the following command to gdb.

- *Info functions*

```
File IT23621138.c:
13:     int main();
5:      void xor_encrypt_decrypt(char *, const char *);

Non-debugging symbols:
0x0000000000001000  _init
0x00000000000010d0  __cxa_finalize@plt
0x00000000000010e0  puts@plt
0x00000000000010f0  fclose@plt
0x0000000000001100  strlen@plt
0x0000000000001110  __stack_chk_fail@plt
0x0000000000001120  pclose@plt
0x0000000000001130  fputs@plt
0x0000000000001140  strcspn@plt
0x0000000000001150  fgets@plt
0x0000000000001160  popen@plt
0x0000000000001170  fopen@plt
0x0000000000001180  _start
0x00000000000011b0  deregister_tm_clones
0x00000000000011e0  register_tm_clones
0x0000000000001220  __do_global_dtors_aux
--Type <RET> for more, q to quit, c to continue without paging--
0x0000000000001260  frame_dummy
0x0000000000001418  _fini
(gdb)
```

- From the above output, the following can be observed;
  - There are two main sections in the source code, the *main()* function and the *xor_encrypt_decrypt()* function.
  - The source code contains system calls like *fclose()* and *fopen()*.
  - The source code also has system calls like *popen()* and *pclose()*.
  - The source code seems to implement a XOR encryption, which can be observed in the initial output lines. It is probable that the data.txt file, which is the output of this executable file, be encrypted using XOR.

Step 03:

To further understand the source code of this executable file, the following command can be used to see the assembly line of the main function in the source code.

- *Disassemble main*

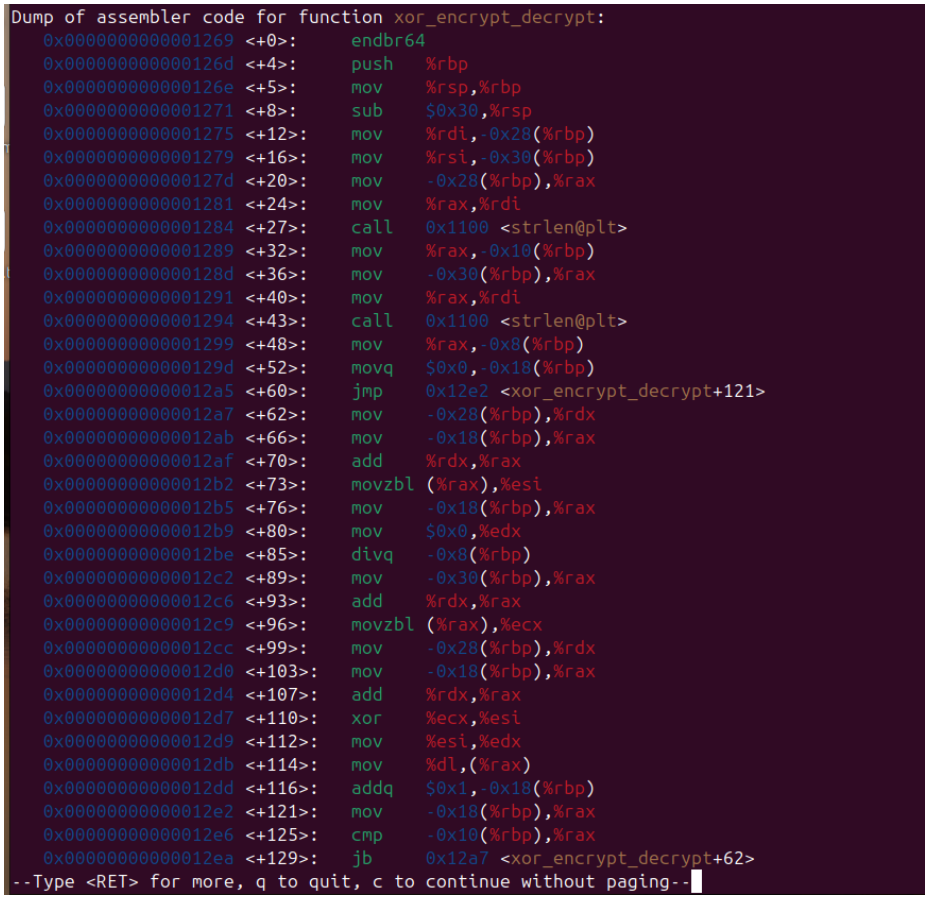
```
(gdb) disassemble main
Dump of assembler code for function main:
   0x00000000000012f0 <+0>:     endbr64
   0x00000000000012f4 <+4>:     push   %rbp
   0x00000000000012f5 <+5>:     mov    %rsp,%rbp
   0x00000000000012f8 <+8>:     sub    $0x60,%rsp
   0x00000000000012fc <+12>:    mov    %fs:0x28,%rax
   0x0000000000001305 <+21>:    mov    %rax,-0x8(%rbp)
   0x0000000000001309 <+25>:    xor    %eax,%eax
   0x000000000000130b <+27>:    lea    0xcf6(%rip),%rax        # 0x2008
   0x0000000000001312 <+34>:    mov    %rax,%rsi
   0x0000000000001315 <+37>:    lea    0xcf4(%rip),%rax        # 0x2010
   0x000000000000131c <+44>:    mov    %rax,%rdi
   0x000000000000131f <+47>:    call   0x1160 <popen@plt>
   0x0000000000001324 <+52>:    mov    %rax,-0x58(%rbp)
   0x0000000000001328 <+56>:    cmpq   $0x0,-0x58(%rbp)
   0x000000000000132d <+61>:    jne    0x1348 <main+88>
   0x000000000000132f <+63>:    lea    0xd02(%rip),%rax        # 0x2038
   0x0000000000001336 <+70>:    mov    %rax,%rdi
   0x0000000000001339 <+73>:    call   0x10e0 <puts@plt>
   0x000000000000133e <+78>:    mov    $0x1,%eax
   0x0000000000001343 <+83>:    jmp    0x1400 <main+272>
   0x0000000000001348 <+88>:    mov    -0x58(%rbp),%rdx
   0x000000000000134c <+92>:    lea    -0x40(%rbp),%rax
--Type <RET> for more, q to quit, c to continue without paging--S
```

- The above stated observations can be confirmed from this output as;
  - The memory address in <+47> uses the call assembly instruction to implement a *popen()* function in the C library, perhaps to read certain data and information.
  - The memory address in <+73> uses the call assembly instruction to implement a *puts()* function in the C library, which is used to write a string as output and move the cursor to the next line.

Step 04:

To see what more information can be gained, the assembly line code of the other function can be seen as well.

- *Disassemble xor_encrypt_decrypt*



- The above stated observations can be confirmed from this output as;
  - The memory address in <+27> and <+43> uses the call assembly instruction to implement a *strlen()* C library function, which seems to get the length of a certain piece of information.
  - The *xor_encrypt_decrypt()* function present in memory addresses <+60> and <+129> may have been used to encrypt in the same information that the *strlen()* function was a part of, and even the results of the *strlen()* function.

3. File System Analysis:

Observation of file system changes:

Step 01:

To observe the data.txt file use the following file system and file information retrieval commands to see file information before execution.

- *Ls –l*
- *Stat data.txt*

```
thisak@thisak:~/Downloads/Executables/Executables$ ls -l
total 96
-rw-rw-r-- 1 thisak thisak 70824 Mar 17 06:26 ARM
-rw-r--r-- 1 root   root      36 May  1 20:52 data.txt
-rwxr-xr-x 1 root   root   20368 May  1 20:52 IT23621138
thisak@thisak:~/Downloads/Executables/Executables$
thisak@thisak:~/Downloads/Executables/Executables$
thisak@thisak:~/Downloads/Executables/Executables$
thisak@thisak:~/Downloads/Executables/Executables$ stat data.txt
  File: data.txt
  Size: 36              Blocks: 8          IO Block: 4096   regular file
Device: 8,2     Inode: 1314799     Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2025-05-01 20:53:01.691887974 +0530
Modify: 2025-05-01 20:52:40.454897092 +0530
Change: 2025-05-01 20:52:40.454897092 +0530
 Birth: 2025-05-01 20:52:40.454897092 +0530
```

Step 02:

Run the executable file.

Step 03:

To see the file changes after running the executable, re use the commands mentioned in step 01.

```
thisak@thisak:~/Downloads/Executables/Executables$ ls -l
total 96
-rw-rw-r-- 1 thisak thisak 70824 Mar 17 06:26 ARM
-rw-r--r-- 1 root   root      36 May  1 21:04 data.txt
-rwxr-xr-x 1 root   root   20368 May  1 20:52 IT23621138
```

```
thisak@thisak:~/Downloads/Executables/Executables$ stat data.txt
  File: data.txt
  Size: 36              Blocks: 8          IO Block: 4096   regular file
Device: 8,2     Inode: 1314799     Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2025-05-01 20:53:01.691887974 +0530
Modify: 2025-05-01 21:04:11.822600260 +0530
Change: 2025-05-01 21:04:11.822600260 +0530
 Birth: 2025-05-01 20:52:40.454897092 +0530
```

- When comparing the two outputs;
  - The file size remains unchanged at 36 bytes before and after execution.
  - Access time has been changed indicating that the file was accessed after execution.
  - The Modify time, Change time and Birth time has remained unchanged, suggesting that the file content and metadata was not changed.

Metadata analysis of generated files:

Use file manipulation and inspection commands to gain more information about the data.txt file created from the

- *xxd data.txt*

```
thisak@thisak:~/Downloads/Executables/Executables$ xxd data.txt
00000000: 5251 1a0f 541d 5a54 5409 031a 0948 180e  RQ..T.ZTT....H..
00000010: 5140 465c 1c0a 5254 0903 4e5c 0440 5e51  Q@F\..RT..N\.@^Q
00000020: 4c08 501a                                L.P.
thisak@thisak:~/Downloads/Executables/Executables$
```

- *hexdump –C data.txt*

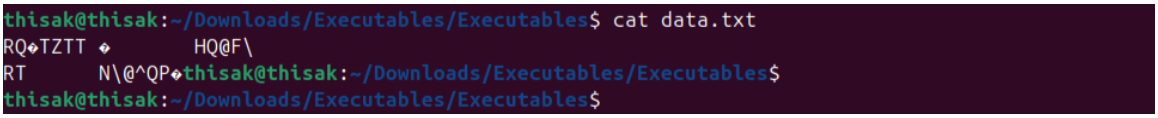
```
thisak@thisak:~/Downloads/Executables/Executables$ hexdump -C data.txt
00000000  52 51 1a 0f 54 1d 5a 54  54 09 03 1a 09 48 18 0e  |RQ..T.ZTT....H..|
00000010  51 40 46 5c 1c 0a 52 54  09 03 4e 5c 04 40 5e 51  |Q@F\..RT..N\.@^Q|
00000020  4c 08 50 1a                                      |L.P.|
00000024
```

- From the above outputs the following observations can be seen;
  - Both the outputs are essentially the same, only the format is slightly different as both show the same hexadecimal values and the same ASCII representation.
  - The file contains a combination of printable and non-printable characters which are shown as '.' in the ASCII column.
  - Characters such as 'R', 'T', '@' and '/' are repetitive in the ACII column.
  - The content in the data.txt file is not human readable and indicates that it might be an encoded file or a file with encrypted data.

4. **Analysis of ‘data.txt’:**

Content explanation:

In the directory it can be seen that a ‘data.txt’ file has been created after the executable file has been run. The ‘cat’ command can be used to see the content of that text file.



- Upon analysis of the content of the .txt file it can be seen that the output of the executable file is encrypted.
- The cipher-text seems to be contain certain repetitive characters like ‘R’, ‘T’, ‘Q’, ‘\’, ‘@’ and a ‘?’ character.
- This observation can be used to come to a conclusion that the encrypted content contains repetitive characters and that the context of the encrypted content is obtainable with the right methods and tools.
- Another conclusion is that the encryption method is one that is not highly complex, but a method that can be deciphered upon efficient effort.

Source of the content of data.txt:

To see what sort of source code could give an encoded or encrypted data.txt file, it is best to see the readable strings of the executable file, to see if any meaningful information can be gained.

- *Strings ./IT23621138*

- The below attached image shows the output of the strings command which revealed the source code.

```
stdio.h
string.h
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void xor_encrypt_decrypt(char *data, const char *key) {
    size_t data_len = strlen(data);
    size_t key_len = strlen(key);
    for (size_t i = 0; i < data_len; i++) {
        data[i] ^= key[i % key_len];
    }
int main() {
    FILE *fp = popen("sudo cat /sys/class/dmi/id/product_uuid", "r");
    if (fp == NULL) {
        printf("Error retrieving data\n");
        return 1;
    }
    char uuid[50];
    fgets(uuid, sizeof(uuid), fp);
    pclose(fp);
    uuid[strcspn(uuid, "\n")] = 0; // Remove newline
    const char *key = "key";
    xor_encrypt_decrypt(uuid, key);
    FILE *out = fopen("data.txt", "w");
    if (out == NULL) {
        printf("Error creating data.txt!\n");
        return 1;
    }
    fprintf(out, "%s", uuid);
    fclose(out);
    return 0;
```

- The function **'xor_encrypt_decrypt'** takes a message and a key word, in this case **'key'** and scrambles each letter of the message using the XOR encryption method.
- In the main function, the **popen()** system call gets the unique ID of the system and prepares to read it, and if there are any errors, an error message is printed and the execution is stopped.
- Then the ID number is read into a string named **'UUID'** and closes the file it opened used to get the ID number (**pclose(fp)**).
- The remove newline syntax eliminates the invisible enter character at the end of the ID number.
- The **'const char *key = "key"'** sets the password for the XOR scramble as 'key'.
- The function calling **'xor_encrypt_decrypt(uuid,key)'** encrypts the UUID using the XOR function.
- The **fopen()** system call creates a writable text file named **'data.txt'**.
- The final **'if'** condition stops the execution if the file cannot be created.
- The **fprintf** prints the encrypted UUID into the data.txt file, closes the file using **'fclose(out)'** and exits the program.

Decoded/Processed information:

To verify that the source code that was identified actually creates a data.txt, the source code was extracted and was run separately.

```
thisak@thisak:~/Desktop$ nano temp.c
thisak@thisak:~/Desktop$ gcc temp.c -o temp
thisak@thisak:~/Desktop$ sudo ./temp
[sudo] password for thisak:
thisak@thisak:~/Desktop$ ls
data.txt  index.html  temp  temp.c  webpage.txt
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$ cat data.txt
RQ◆TZTT ◆       HQ@F\
RT      N\@^QP◆thisak@thisak:~/Desktop$
```

- It can be seen that the source code does indeed create the data.txt, and no other factors are affected, as well as the encrypted content of the data.txt is the same.

Now to verify that the UUID is the one that is being encrypted in the data.txt file, a simple XOR decryption python script can be implemented.

```
 GNU nano 7.2                                                  decrpyt.py
def xor_encrypt_decrypt(data, key):
    key_len = len(key)
    return ''.join(chr(ord(c) ^ ord(key[i % key_len])) for i, c in enumerate(data))

# Read encrypted data
with open("data.txt", "r") as file:
    encrypted_data = file.read()

# Decrypt it
key = "key"
decrypted_uuid = xor_encrypt_decrypt(encrypted_data, key)

print("Decrypted UUID:", decrypted_uuid)
```

- The above python code reads the content of the data.txt and decrypts the XOR encryption to show the UUID using the ‘key’ key.
- The image attached below shows the output and the confirmation that it is indeed the UUID of the system.

```
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$ python3 decrpyt.py
Decrypted UUID: 94cd1d11-bfcb-ae49-9ea7-bf77a9545c5c
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$
thisak@thisak:~/Desktop$ sudo cat /sys/class/dmi/id/product_uuid
94cd1d11-bfcb-ae49-9ea7-bf77a9545c5c
thisak@thisak:~/Desktop$
```

5. **Tools used for analysis:**

The following tools and commands were used during this process:

- **cat** → to view contents of files.
- **strings** →to extract readable text strings from the executable file.
- **xxd** and **hexdump** → to view hexadecimal representations of file content.
- **GDB** → to attempt debugging the executable file to understand its functionality.
- Custom **Python XOR decryption script** →to decrypt the content in data.txt.
- **ls -l** and **stat** →for file metadata and modification timestamps.