<table>
<tr><td>College Name</td><td colspan="3">University of Hertfordshire
SEGi College <u>Subang Jaya</u></td></tr>
<tr><td>Programme Name</td><td colspan="3">BACHELOR OF <u>CYBERSECURITY AND NETWORKS</u></td></tr>
<tr><td rowspan="2">Module Name</td><td rowspan="2">CYBERSECURITY</td><td>Module Code</td><td>6COM1040</td></tr>
<tr><td>Semester</td><td>SEP 2025</td></tr>
<tr><td>Module Leader</td><td>DR ANESHKUMAR THANGAVELOO</td><td rowspan="2">Assessment Type</td><td rowspan="2">PROJECT COMPREHENSIVE REPORT</td></tr>
<tr><td>Lecturer Name</td><td>MS NUR DIANA MADINAH</td></tr>
<tr><td>Student’s declaration</td><td colspan="3">I hereby certify that this assignment is my own work and where materials have been used from other resources, they have been properly acknowledged. I also understand I will face the possibility of failing the module if the content of this assignment is plagiarized.
<table>
<tr><th>No.</th><th>Name</th><th>Student ID</th><th>Signature / Initial</th></tr>
<tr><td>1</td><td>TAN JIAJIAN</td><td>SCSJ1901781</td><td>TAN</td></tr>
</table></td></tr>
<tr><td>Release Date</td><td></td><td>Submission Due Date</td><td>8/12/2025</td><td rowspan="2">Marks obtained:</td></tr>
<tr><td>Date Received</td><td></td><td>Student’s work assessed by / date</td><td></td></tr>
</table>

Module Leader's Feedback.

Contents

List of Figure

1. Introduction

A2Z Corporation is a mid-sized company that manages sensitive customer information, including payment details and personally identifiable information. To protect its network security, the company has implemented several security measures, including encryption, firewalls, and intrusion detection systems (IDS). These mechanisms are designed to protect data in transit, manage network traffic, and detect potential threats in real time. Although the company used these measures, the company still faces various internet-facing risks, therefore requiring it to build a robust network and continuously assess its security posture.

2. System Design and Security Analysis

Internet-Facing Security Risks

A2Z Corporation faces several security threats due to its public-facing services

1. Unauthorized Access to Internal LAN: Attackers may attempt to exploit misconfigured routers or open ports to access sensitive administrative systems.
2. Attacks on Public Web Server: The DMZ-hosted web server is exposed to potential threats such as DDoS attacks, force login attempts and web application vulnerabilities.
3. Malware and Phishing Attacks: External sources could attempt to introduce malware or phishing content into the network, threatening sensitive data.
4. Reconnaissance Activities: Cyber attackers may scan open ports or exploitable services, aiming to gather information for subsequent attacks.

System Architecture Diagram

The proposed network architecture uses segmentation, controlled access and traffic filtering to mitigate these risks.

1. Network Segmentation:
    - DMZ (192.168.10.0/24): Contains the web server (192.168.10.1/24). allow only HTTP (port 80) and HTTPS (port 443) to the internet.
    - Internal LAN (192.168.20.0/24): Contains Admin-PC (192.168.20.1/24). Fully isolated from inbound internet traffic.
    - The router interfaces are isolated networks: Gig0/0 connects to the internet (203.0.113.2), Gig0/1 to DMZ, and Gig0/2 to the internal LAN.
2. Firewall Policies:
    - Inbound to DMZ: Only HTTP (80) and HTTPS (443) traffic is allowed. All other ports are blocked.
    - Inbound to Internal LAN: Completely blocked.
    - Outbound to Internal LAN: The administrator PC can access the internet and DMZ to manage the web server.
3. Device Placement and Connectivity:
    - Internet (Cloud): IP 203.0.113.1 connected to router Gig0/0 via a copper straight-through cable.
    - Router: Edge router handling routing and firewall policies. Interfaces:
        - G0/0 → Cloud

    - G0/1 → SW-DMZ
    - G0/2 → SW-LAN
- Switches:
    - SW-DMZ connects to router G0/1 and DMZ-WebServer FastEthernet0.
    - SW-LAN connects to router G0/2 and Admin-PC FastEthernet0.
- Server: Ubuntu web server in DMZ, IP 192.168.10.1.
- Admin-PC: Internal workstation, IP 192.168.20.1.
- Cables: All copper straight-through.
- Security:
    - Firewall configured on router interfaces (G0/0, G0/1, G0/2).
    - Only required ports open, DMZ exposed to HTTP/HTTPS, internal LAN isolated.

4. Encryption and Monitoring:
    - HTTPS (port 443): Ensures encrypted traffic between external users and the web server.
    - Potential IDS/IPS in Packet Tracer simulation: Can monitor unusual traffic in DMZ and internal LAN, alerting administrators to intrusion attempts.

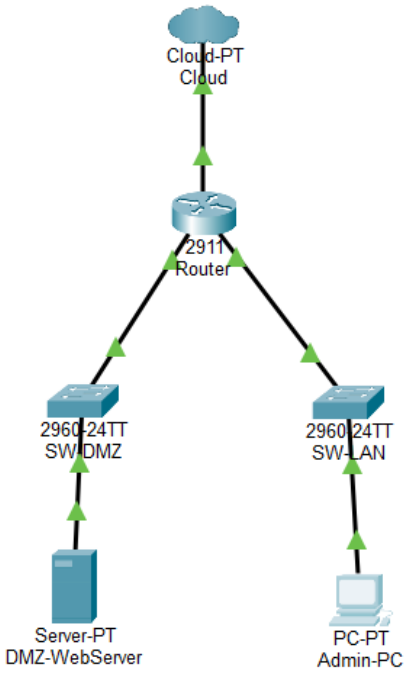

Figure 1

Justification of design choices

1. Segmentation: Separating DMZ from the internal LAN ensures that even if the web server is compromised, the core administrative systems remain protected.
2. Controlled Access: By allowing only essential ports and blocking all other traffic, the network reduces the attack surface and prevents common exploits.
3. Routing and Firewall Placement: Placing the firewall rules on router simplifies traffic control and ensures a central point of policy enforcement.
4. Encrypted Traffic: Using HTTPS ensures data confidentiality for clients accessing public services, mitigating risks of eavesdropping or data tampering.
5. Future IDS/IPS Integration: Although optional in Packet Tracer, adding monitoring systems would enhance detection of abnormal activities and support proactive security.

3. Implementation of Security Measures

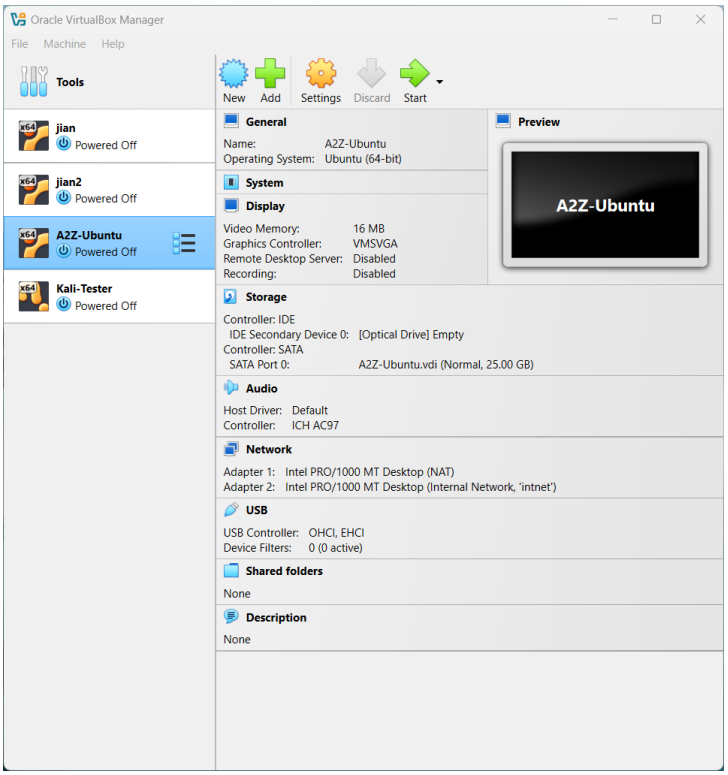

Figure 2

Figure 2 is A2Z-Ubuntu virtual machine.

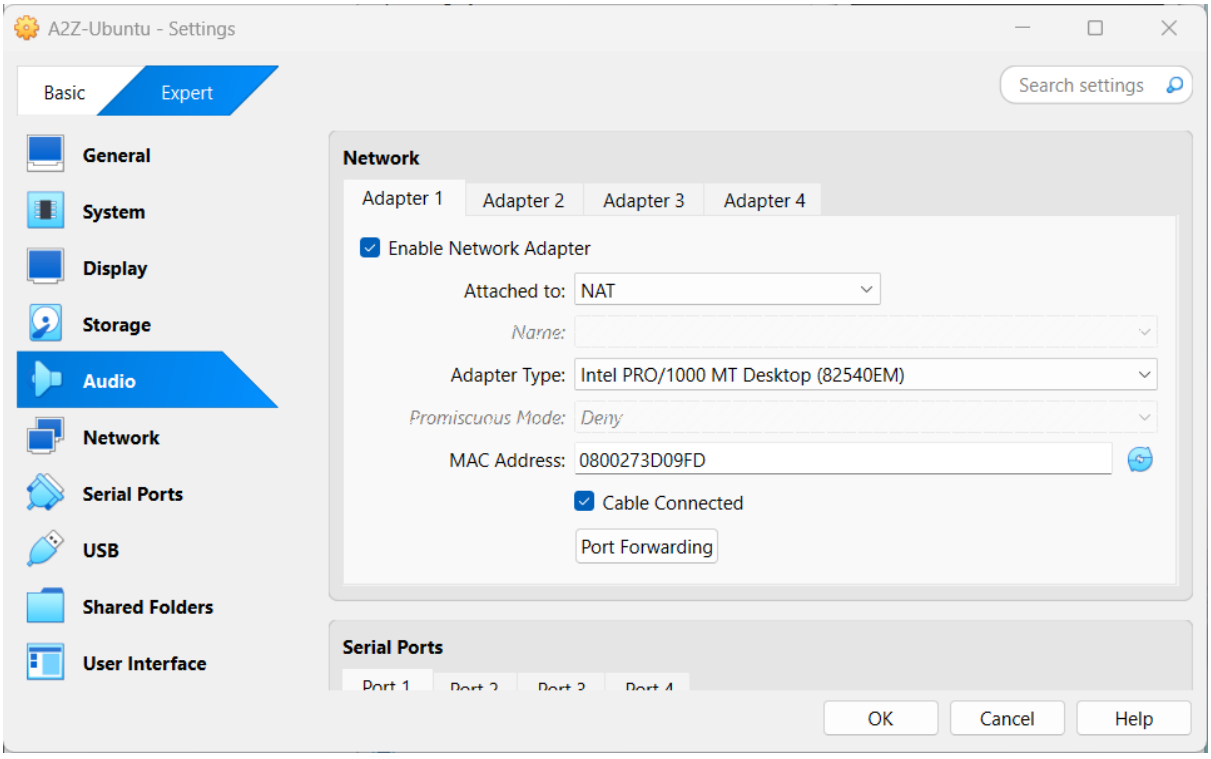

Figure 3

Figure 3 is Adapter 1 of A2Z-Ubuntu virtual machine, I set it as NAT.

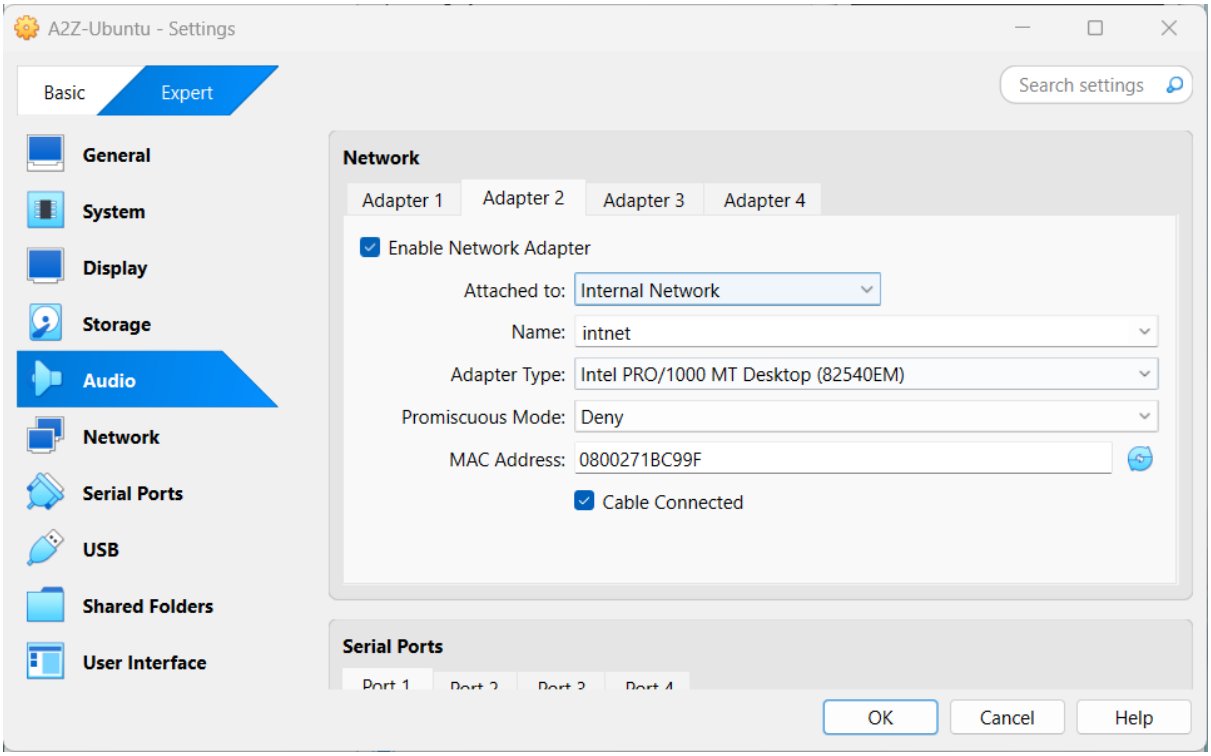

Figure 4

Figure 4 is Adapter 2 of A2Z-Ubuntu virtual machine, I set it as Internal Network because need to set an IP address to connect with Kali-Linux virtual machine. I set the IP address as 192.168.10.1.

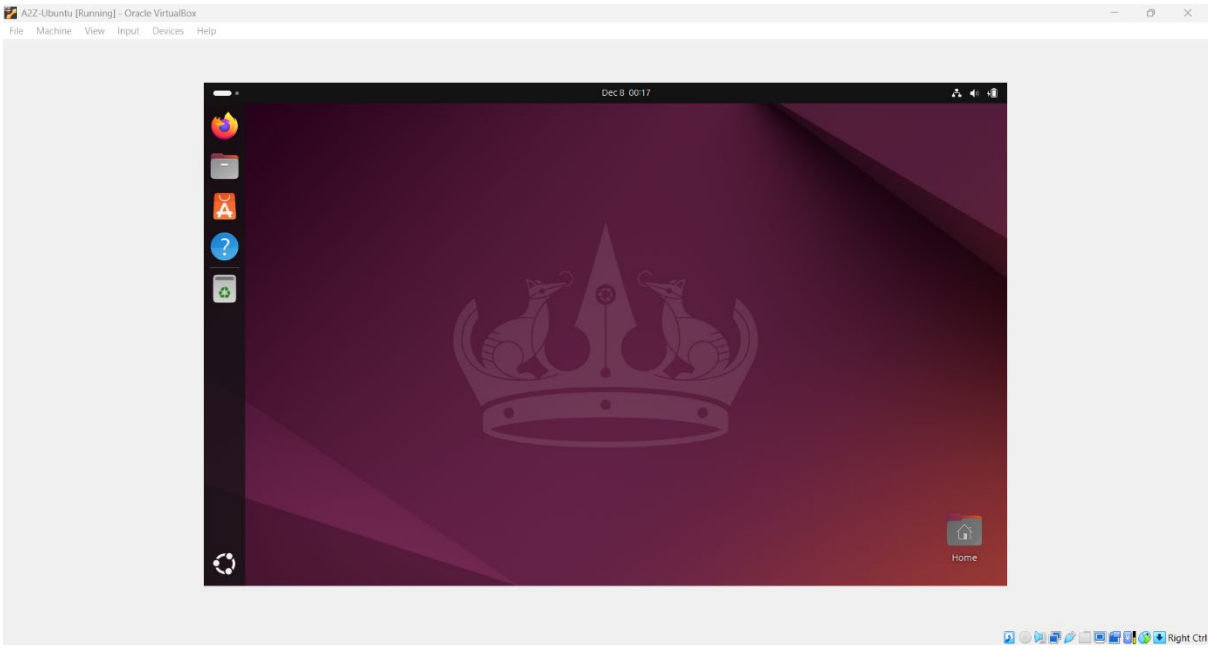

Figure 5

Figure 5 is A2Z-Ubuntu virtual machine home screen.

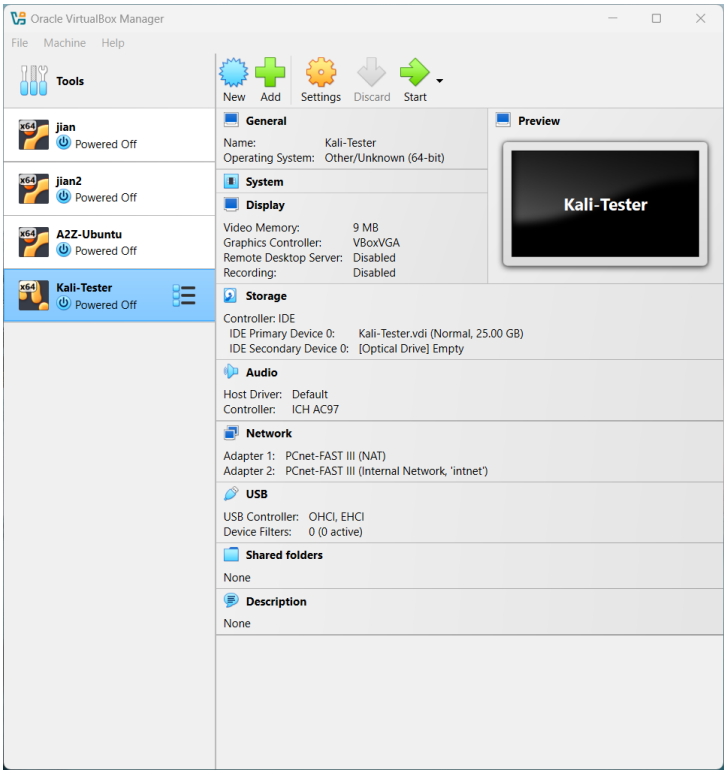

Figure 6

Figure 6 is Kali-Tester virtual machine.

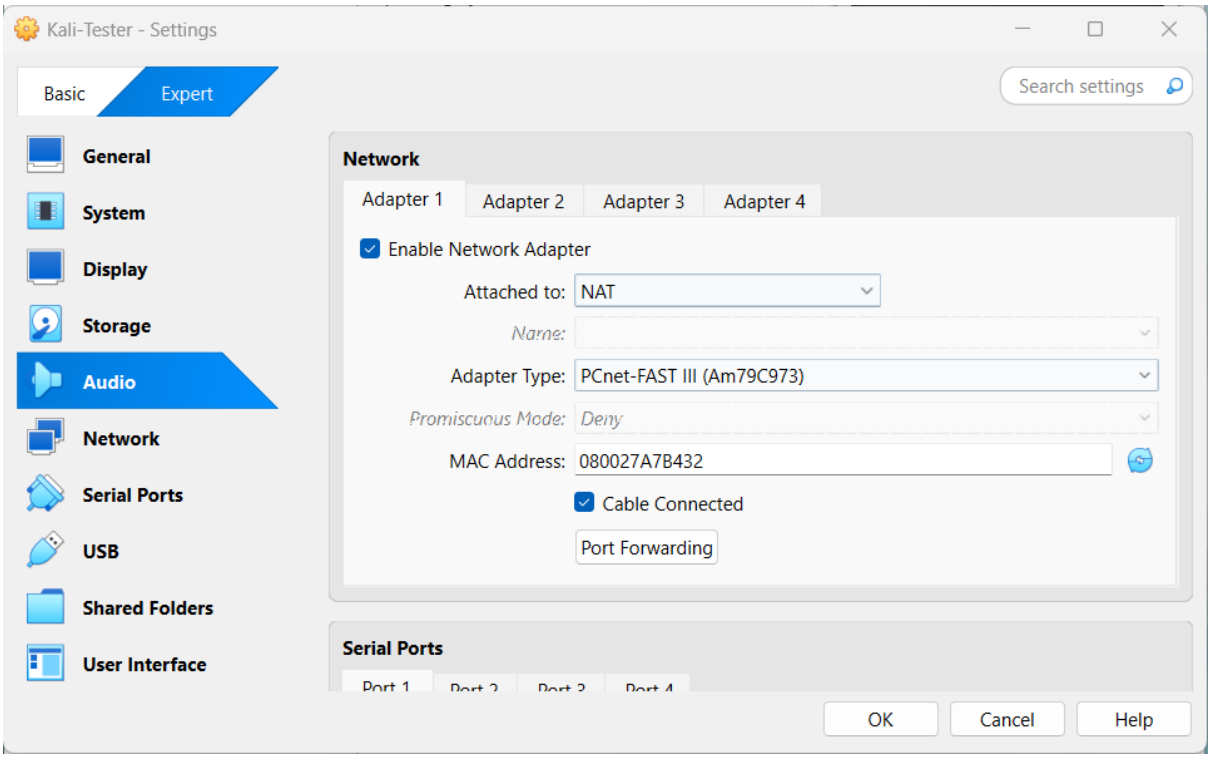

Figure 7

Figure 7 is Adapter 1 of Kali-Tester virtual machine, I set it as NAT.

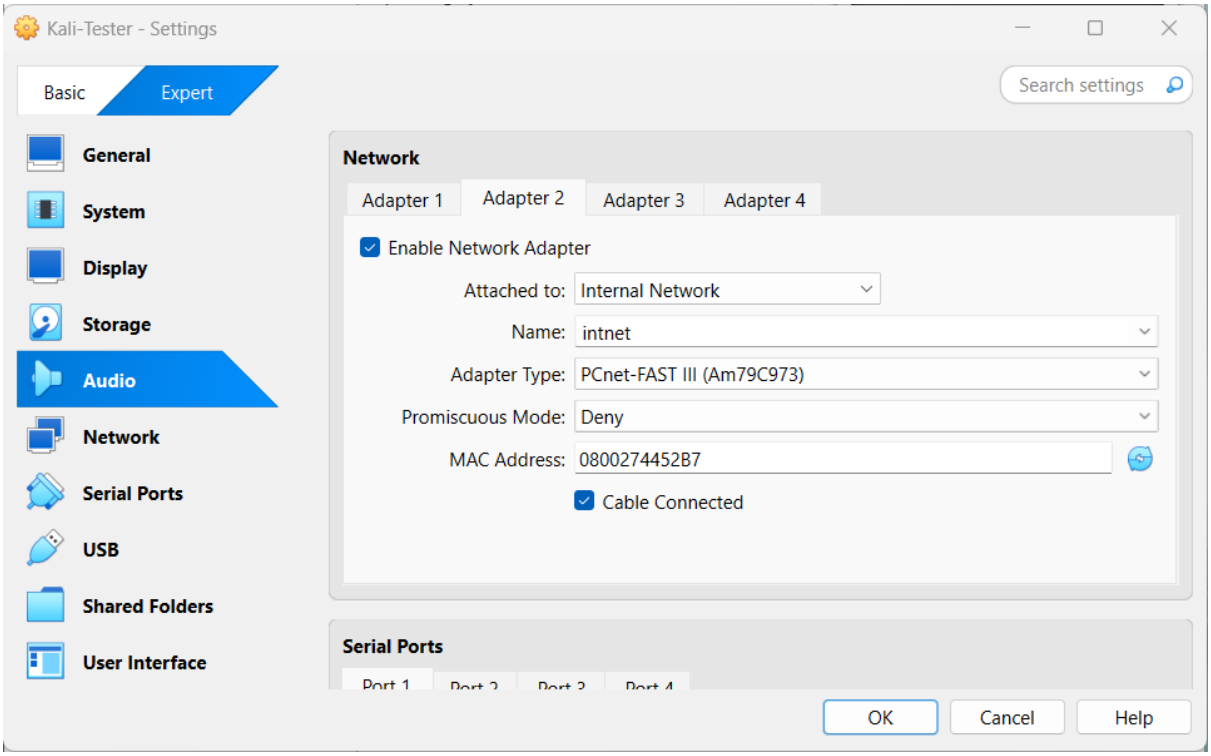

Figure 8

Figure 8 is Adapter 2 of Kali-Tester virtual machine, I set it as Internal Network because need to set an IP address to connect with Ubuntu virtual machine. I set the IP address as 192.168.10.11.

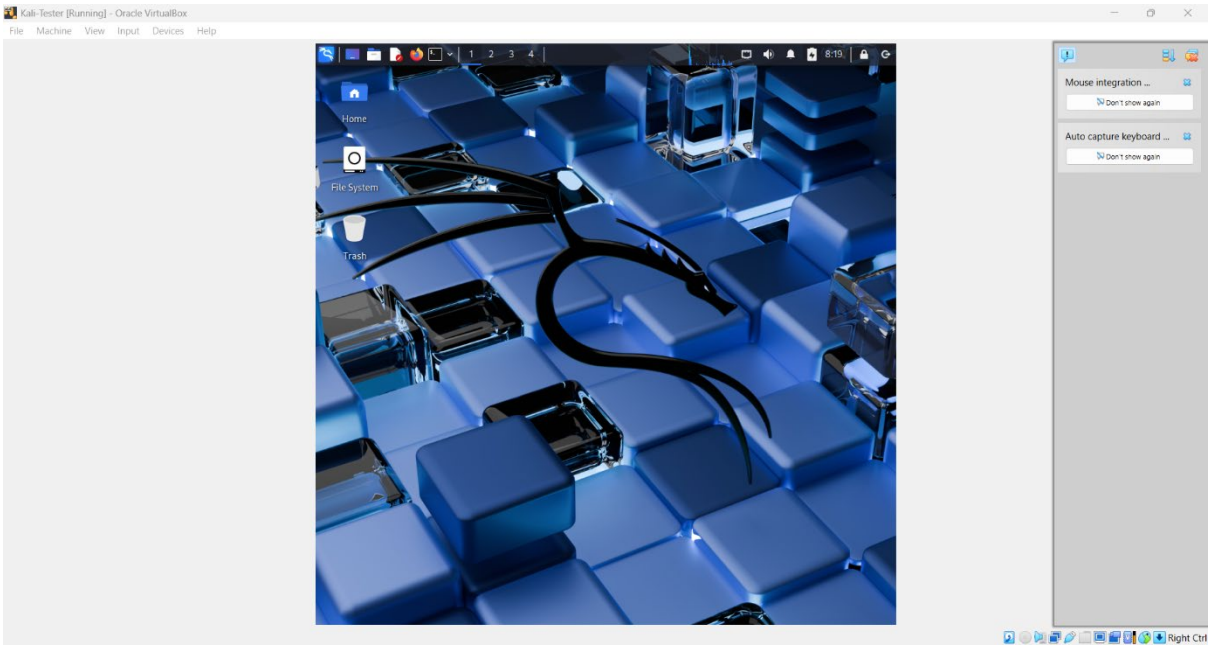

Figure 9

Figure 9 is Kali-Tester virtual machine home screen.

```
jiajan@jiajian-VirtualBox:~$ sudo apt update
[sudo] password for jiajan:
Hit:1 http://my.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://my.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://my.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
203 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Figure 10

Figure 10 is sudo apt update command to refreshes the system's list of available software packages from online repositories.

```
jiajan@jiajian-VirtualBox:~$ sudo apt install -y docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd git git-man liberror-perl pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools btrfs-progs cgroupfs-mount | cgroup-lite debootstrap
  docker-buildx docker-compose-v2 docker-doc rinse zfs-fuse | zfsutils
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb
  git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  bridge-utils containerd docker.io git git-man liberror-perl pigz runc
  ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 203 not upgraded.
Need to get 80.4 MB of archives.
After this operation, 312 MB of additional disk space will be used.
Get:1 http://my.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1
[65.6 kB]
Get:2 http://my.archive.ubuntu.com/ubuntu noble/main amd64 bridge-utils amd64 1.
7.1-1ubuntu2 [33.9 kB]
Get:3 http://my.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd am
d64 1.7.28-0ubuntu1~24.04.1 [38.4 MB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 runc amd64 1.3
```

Figure 11

Figure 11 sudo apt install -y docker.io command is to download the docker.

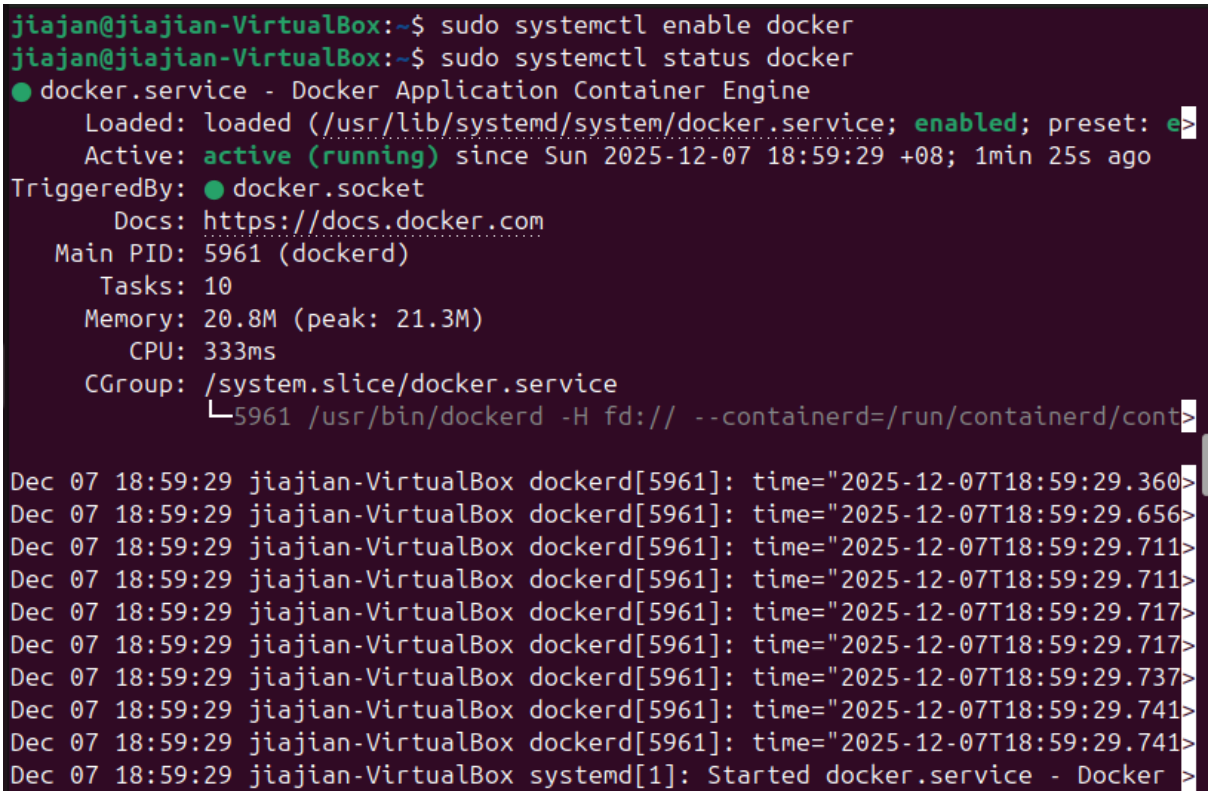

Figure 12

Figure 12 sudo systemctl enable docker to start the docker application container engine and sudo systemctl status docker is check whether docker engine is active or not.

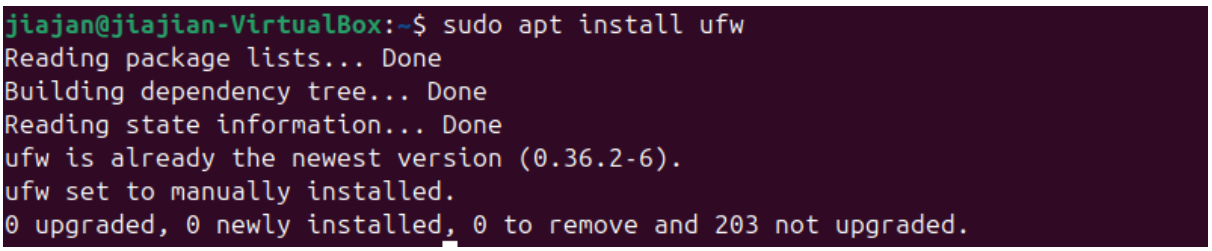

Figure 13

Figure 13 sudo apt install ufw is installed the ufw firewall.

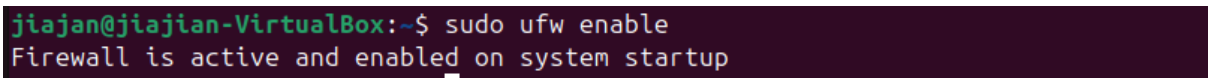

Figure 14

Figure 14 sudo ufw enable is open the ufw firewall.

```
jiajan@jiajian-VirtualBox:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
jiajan@jiajian-VirtualBox:~$ sudo ufw allow 443/tcp
Rule added
Rule added (v6)
```

Figure 15

Figure 15 sudo ufw allow 80 & 443 tcp is only allow required ports.

```
jiajan@jiajian-VirtualBox:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
jiajan@jiajian-VirtualBox:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
```

Figure 16

Figure 16 sudo ufw default deny incoming is deny anything else from outside and sudo ufw default allow outgoing is allow anything else from inside.

```
jiajan@jiajian-VirtualBox:~$ sudo ufw allow from 192.168.10.11 to any port 22
Rule added
jiajan@jiajian-VirtualBox:~$ sudo ufw deny 22/tcp
Rule added
Rule added (v6)
```

Figure 17

Figure 17 sudo ufw allows from 192.168.10.11 to any port 22 and sudo ufw deny 22/tcp is restrict SSH so only Kali-Tester IP addresses can access.

```
jiajan@jiajian-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
80/tcp                     ALLOW IN    Anywhere
443/tcp                    ALLOW IN    Anywhere
22                         ALLOW IN    192.168.10.11
22/tcp                     DENY IN     Anywhere
80/tcp (v6)                ALLOW IN    Anywhere (v6)
443/tcp (v6)               ALLOW IN    Anywhere (v6)
22/tcp (v6)                DENY IN     Anywhere (v6)
```

Figure 18

Figure 18 sudo ufw status verbose is check firewall status.

```
jiajan@jiajian-VirtualBox:~$ sudo docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
0e4bc2bd6656: Pull complete
b5feb73171bf: Pull complete
108ab8292820: Pull complete
53d743880af4: Pull complete
77fa2eb06317: Pull complete
192e2451f875: Pull complete
de57a609c9d5: Pull complete
Digest: sha256:553f64aecdc31b5bf944521731cd70e35da4faed96b2b7548a3d8e2598c52a42
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
```

Figure 19

Figure 19 sudo docker pull nginx is pull the open-source web server image.

```
jiajan@jiajian-VirtualBox:~$ sudo docker run -d --name webserver -p 80:80 nginx
20043ec978317e7deec6220d33dbba6c771ac00bd924d3364f7af245189c47cb
```

Figure 20

Figure 20 sudo docker run -d --name webserver -p 80:80 nginx is run container with correct port mapping. After running the container, from Kali browser type http://192.168.10.1 will shows “Welcome to nginx!”, so the server is working.

```
jiajan@jiajian-VirtualBox:~$ sudo docker pull vulnerables/web-dvwa
Using default tag: latest
latest: Pulling from vulnerables/web-dvwa
3e17c6eae66c: Pull complete
0c57df616dbf: Pull complete
eb05d18be401: Pull complete
e9968e5981d2: Pull complete
2cd72dba8257: Pull complete
6cff5f35147f: Pull complete
098cffd43466: Pull complete
b3d64a33242d: Pull complete
Digest: sha256:dae203fe11646a86937bf04db0079adef295f426da68a92b40e3b181f337daa7
Status: Downloaded newer image for vulnerables/web-dvwa:latest
docker.io/vulnerables/web-dvwa:latest
```

Figure 21

Figure 21 sudo docker pull vulnerables/web-dvwa is pull the DVWA docker image.

```
jiajan@jiajian-VirtualBox:~$ sudo docker run -d --name dvwa -p 8080:80 vulnerabl
es/web-dvwa
1d84e9072e264b45c06cbb1df46e906d75143e8d1d42e7509aac7af810362ed9
```

Figure 22

Figure 22 is run DVWA container.

Next, type the sudo apt-get install snort -y to install the snort.

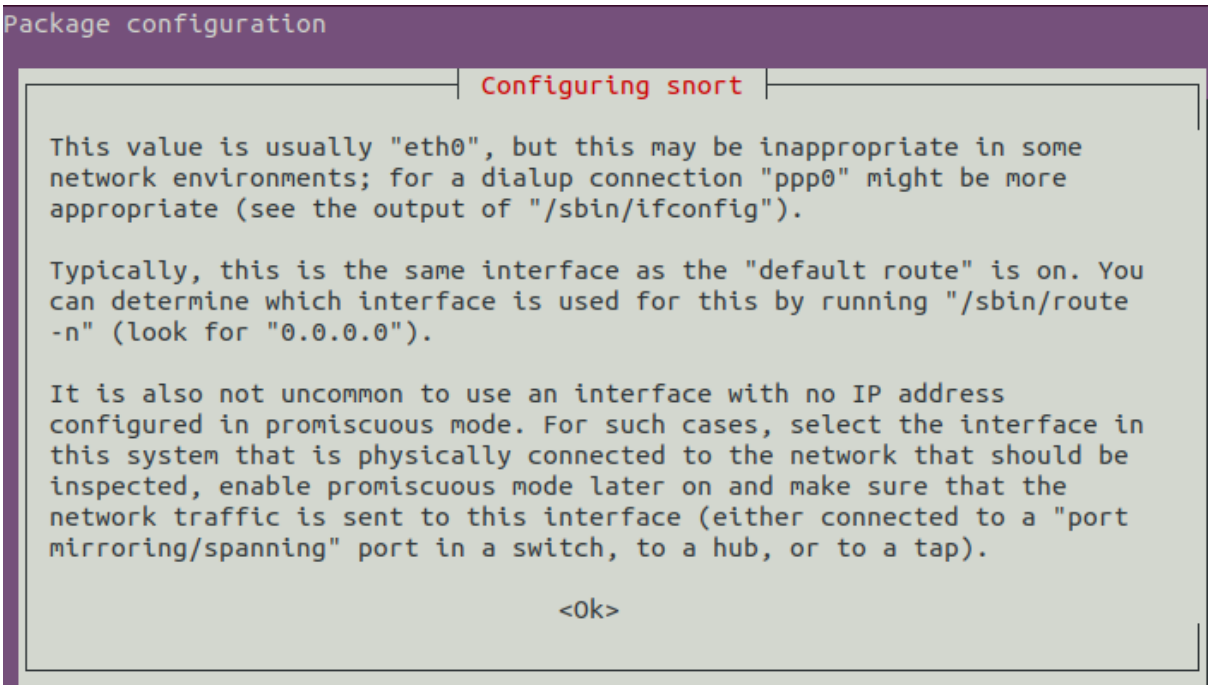

Figure 23

Figure 23 is after type the command, the system shows package configuration window to configure the snort.

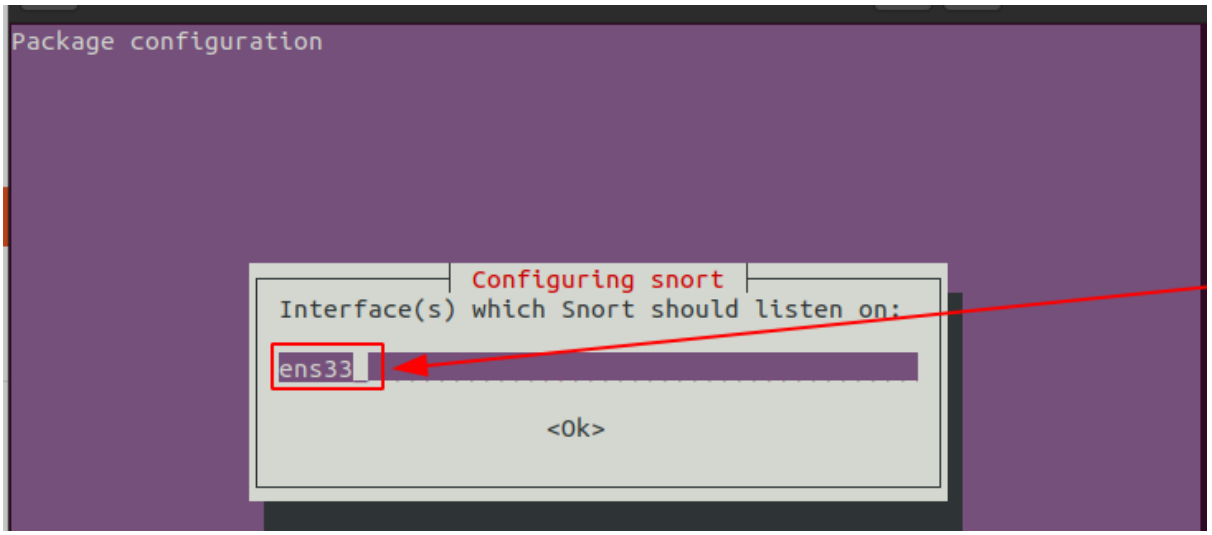

Figure 24

Figure 24 is typing the interface which Snort should listen on, in this part I put the interface as enp0s8.

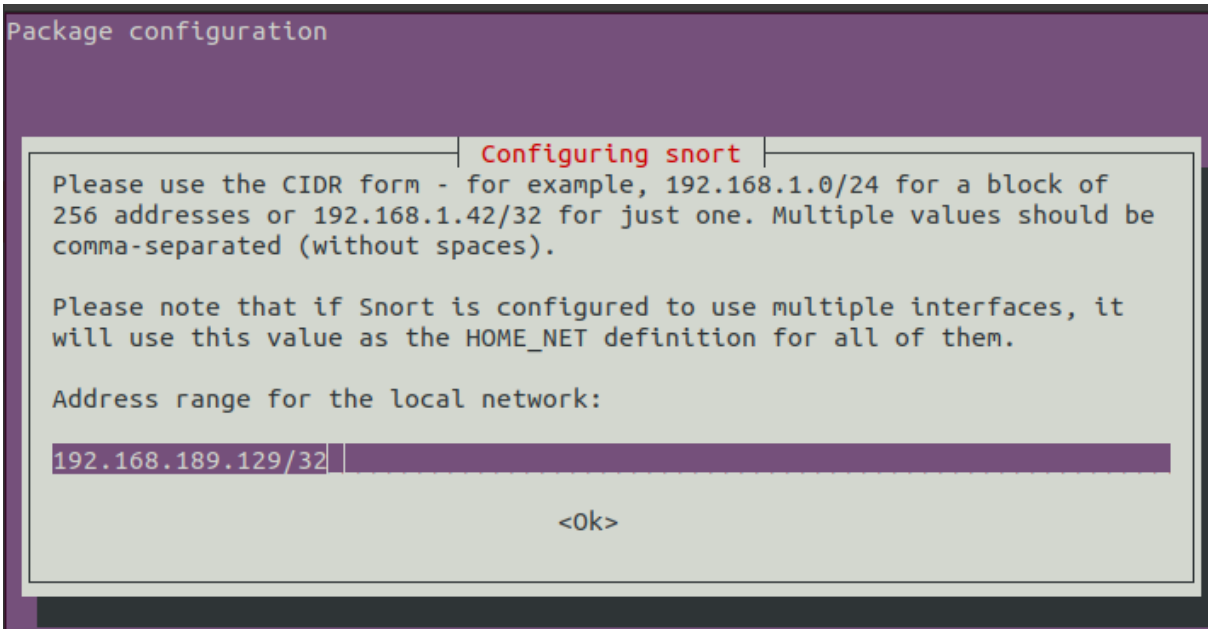

Figure 25

Figure 25 is typing the IP address for the local network, at this part I put the IP address as 192.168.10.1/24.

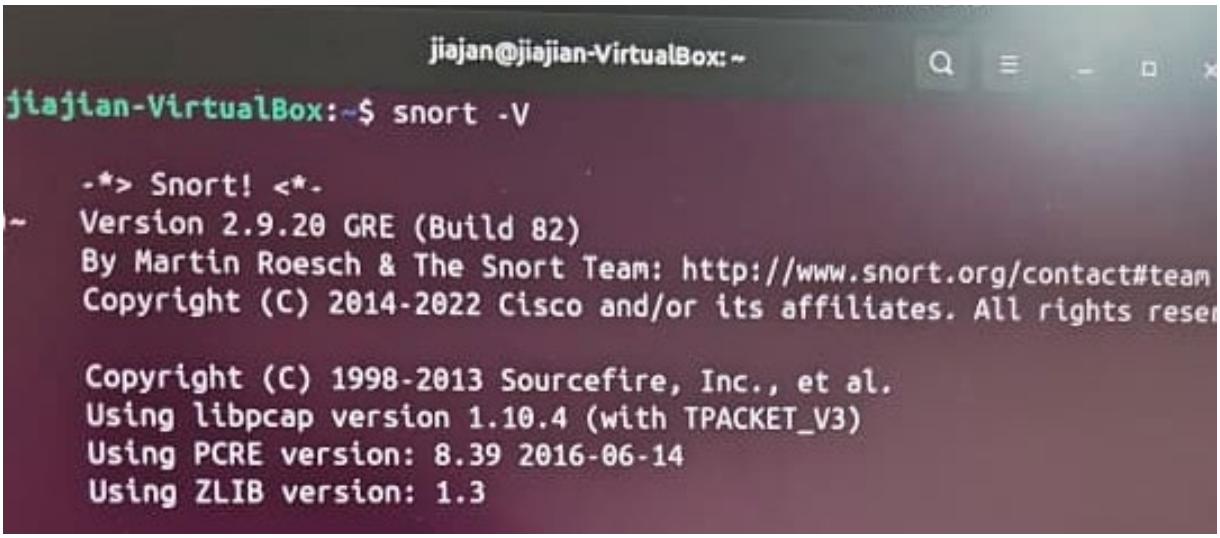

Figure 26

Figure 26 is after setting all the interface and IP address, snort -V is to determine which version of the Snort is installed.

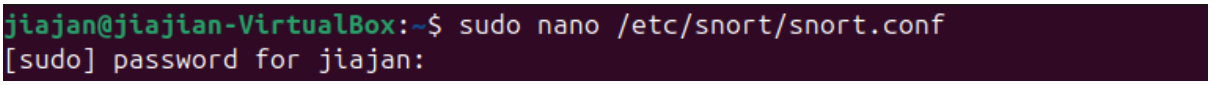

Figure 27

Figure 27 is a I should type the Ubuntu virtual machine's IP address as a HOME_NET, so I type this command.

```
# Note to Debian users: this value is overriden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 192.168.10.1/24

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET
```

Figure 28

Figure 28 shows the HOME_NET already set as 192.168.10.1/24.

```
# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules
```

Figure 29

Figure 29 shows The RULE_PATH variable in the snort.conf file determines the location of the snort rule files.

```
# site specific rules
include $RULE_PATH/local.rules

# The include files commented below have been disabled
# because they are not available in the stock Debian
# rules. If you install the Sourcefire VRT please make
# sure you re-enable them again:

#include $RULE_PATH/app-detect.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/blacklist.rules
#include $RULE_PATH/botnet-cnc.rules
#include $RULE_PATH/browser-chrome.rules
#include $RULE_PATH/browser-firefox.rules
#include $RULE_PATH/browser-ie.rules
#include $RULE_PATH/browser-other.rules
#include $RULE_PATH/browser-plugins.rules
#include $RULE_PATH/browser-webkit.rules
include $RULE_PATH/chat.rules
#include $RULE_PATH/content-replace.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/experimental.rules
#include $RULE_PATH/exploit-kit.rules
include $RULE_PATH/exploit.rules
```

Figure 30

Figure 30 shows all the community rules.

```
jiajan@jiajian-VirtualBox:~$ sudo snort -T -i ens33 -c /etc/snort/snort.conf
Running in Test mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined :  [ 80:81 311 383 591 593 901 1220 1414 1741 1830
2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777
7779 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300
8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002
55555 ]
```

Figure 31

Figure 31 is a command that can test the configuration. If make any mistakes in the configuration file will show error.

```
Total snort Fixed Memory Cost - MaxRss:103956
Snort successfully validated the configuration!
Snort exiting
```

Figure 32

Figure 32 shows if everything is configured correctly, will shows a “Snort successfully validated the configuration!” message.

```
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.

alert icmp any any -> $HOME_NET any (msg:"ICMP Detection Rule"; sid:1000001;)
alert tcp any any -> $HOME_NET 22 (msg:"SSH Connection Attempts"; sid:1000002)
alert tcp any any -> $HOME_NET 80 (msg:"Command Execution Attempt"; content:"GET"; content:"/etc/passwd"; sid:1000003;)
```

Figure 33

Figure 33 shows I set three alerts, first is generate an alert when an ICMP traffic is received from any source IP Address to Ubuntu. Second is create a snort rule to detect SSH connection attempts and last create a snort rule to detect a command execution attack which contains /etc/passwd in the http GET request.

4. Testing

In this assessment, I used Nmap, tcpdump and Snort IDS to identify vulnerabilities in the test network. The target machine Ubuntu server with IP address 192.168.10.1 was hosting two services:

- Nginx on port 80
- DVWA on port 8080
- Snort IDS running in docker on Ubuntu
- Kali Linux used as attacker machine.

From these tools, I identified several vulnerabilities explained below.

Vulnerability 1 - Port 80 (HTTP) is Open Without Encryption

Evidence

From nmap

80/tcp open http

Reasoning

HTTP on port 80 sends data in plain text, including:

- Cookies
- Login information
- Session tokens
- Web requests

An attacker can capture this traffic using tcpdump or Wireshark

Risk Level

Medium to High

Because attackers in the same network can sniff traffic and steal sensitive data.

Recommended Fix

- Enable HTTPS with SSL/TLS certificates
- Redirect all HTTP traffic to HTTPS
- Disable plain HTTP if not needed

Vulnerability 2 - DVWA (Port 8080) Running in “Low Security Mode”

Evidence

From tcpdump on port 8080:

HTTP/1.1 200 OK

DVWA login page

From the browser: DVWA accessible without restriction.

Reasoning

DVWA in low security mode contains intentional vulnerabilities:

- SQL Injection
- XSS
- Insecure file upload
- Command execution
- CSRF

If a malicious user accesses DVWA, they can exploit these vulnerabilities to attack the server or compromise network.

Risk Level

High

DVWA is purposely vulnerable.

Recommended Fix

- Never expose DVWA to a real production network
- Use firewall rules to only allow trusted hosts
- Run DVWA in an isolated environment / VM only

Vulnerability 3 - Port 80 (Nginx) Traffic Not Captured by tcpdump

Evidence

- tcpdump -i enp0s8 port 80 shows no packets when Kali accesses http://192.168.10.1.
- But tcpdump -i enp0s8 port 8080 shows GET / 200 OK when accessing DVWA.
- This means port 80 traffic is not passing through enp0s8.

Reasoning

Nginx on port 80 is likely bound to the NAT interface, not the internal network interface enp0s8.

Because of this, tcpdump and Snort cannot see or monitor port 80 traffic, creating a monitoring gap.

Risk Level

Critical - Attacks on port 80 bypass Snort and tcpdump, allowing undetected exploitation of the web server.

Recommended Fix

- Bind Nginx to the internal network IP (192.168.10.1).
- Re-run the container with correct port mapping.
- Verify that port 80 traffic goes through enp0s8 so Snort and tcpdump can inspect it.

```
┌──(jiajian㉿vbox)-[~]
└─$ nmap 192.168.10.1
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-07 05:42 PST
Nmap scan report for 192.168.10.1 (192.168.10.1)
Host is up (0.00039s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE  SERVICE
22/tcp   closed ssh
80/tcp   open   http
443/tcp  closed https
8080/tcp open   http-proxy
MAC Address: 08:00:27:1B:C9:9F (PCS Systemtechnik/Oracle VirtualBox virtual N
IC)

Nmap done: 1 IP address (1 host up) scanned in 4.98 seconds
```

Figure 34

Figure 34 is finding the open ports on Ubuntu.

```
jiajan@jiajian-VirtualBox:~$ sudo snort -q -l /var/log/snort/ -i enp0s8 -A conso
le -c /etc/snort/snort.conf
12/08-00:51:38.859009  [**] [1:1000002:0] SSH Connection Attempts [**] [Priority
: 0] {TCP} 192.168.10.11:42853 -> 192.168.10.1:22
12/08-00:51:41.372170  [**] [1:1000002:0] SSH Connection Attempts [**] [Priority
: 0] {TCP} 192.168.10.11:42858 -> 192.168.10.1:22
12/08-00:51:41.982796  [**] [1:1418:11] SNMP request tcp [**] [Classification: A
ttempted Information Leak] [Priority: 2] {TCP} 192.168.10.11:42853 -> 192.168.10
.1:161
12/08-00:51:42.083863  [**] [1:1418:11] SNMP request tcp [**] [Classification: A
ttempted Information Leak] [Priority: 2] {TCP} 192.168.10.11:42855 -> 192.168.10
.1:161
12/08-00:51:42.685164  [**] [1:1000002:0] SSH Connection Attempts [**] [Priority
: 0] {TCP} 192.168.10.11:42860 -> 192.168.10.1:22
12/08-00:51:43.001298  [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 192.168.10.11:42853 -> 192
.168.10.1:705
12/08-00:51:43.104510  [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 192.168.10.11:42855 -> 192
.168.10.1:705
```

Figure 35

Figure 35 shows all the log.

```
┌──(jiajian㉿vbox)-[~]
└─$ nmap -sV 192.168.10.1
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-07 05:46 PST
Nmap scan report for 192.168.10.1 (192.168.10.1)
Host is up (0.00046s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE  SERVICE VERSION
22/tcp   closed ssh
80/tcp   open   http    nginx 1.29.3
443/tcp  closed https
8080/tcp open   http    Apache httpd 2.4.25 ((Debian))
MAC Address: 08:00:27:1B:C9:9F (PCS Systemtechnik/Oracle VirtualBox virtual N
IC)
```

Figure 36

Figure 36 commands identify what services and versions are running.

```
┌──(jiajian㉿vbox)-[~]
└─$ nmap -A 192.168.10.1
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-07 05:52 PST
Nmap scan report for 192.168.10.1 (192.168.10.1)
Host is up (0.00037s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE  SERVICE VERSION
22/tcp   closed ssh
80/tcp   open   http    nginx 1.29.3
|_http-title: Welcome to nginx!
|_http-server-header: nginx/1.29.3
443/tcp  closed https
8080/tcp open   http    Apache httpd 2.4.25 ((Debian))
|_http-open-proxy: Proxy might be redirecting requests
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
| http-title: Login :: Damn Vulnerable Web Application (DVWA) v1.10 *Develop.
..
|_Requested resource was login.php
|_http-server-header: Apache/2.4.25 (Debian)
| http-robots.txt: 1 disallowed entry
|_/
MAC Address: 08:00:27:1B:C9:9F (PCS Systemtechnik/Oracle VirtualBox virtual N
IC)
Aggressive OS guesses: Linux 5.0 - 5.14 (98%), MikroTik RouterOS 7.2 - 7.5 (L
inux 5.6.3) (98%), Linux 4.15 - 5.19 (94%), OpenWrt 21.02 (Linux 5.4) (94%),
Linux 2.6.32 - 3.13 (93%), Linux 5.1 - 5.15 (93%), Linux 6.0 (93%), Linux 2.6
.39 (93%), OpenWrt 22.03 (Linux 5.10) (93%), Linux 4.19 (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   0.37 ms 192.168.10.1 (192.168.10.1)
```

Figure 37

Figure 37 commands used to aggressive scan and this scan performs OS detection, version detection and traceroute.

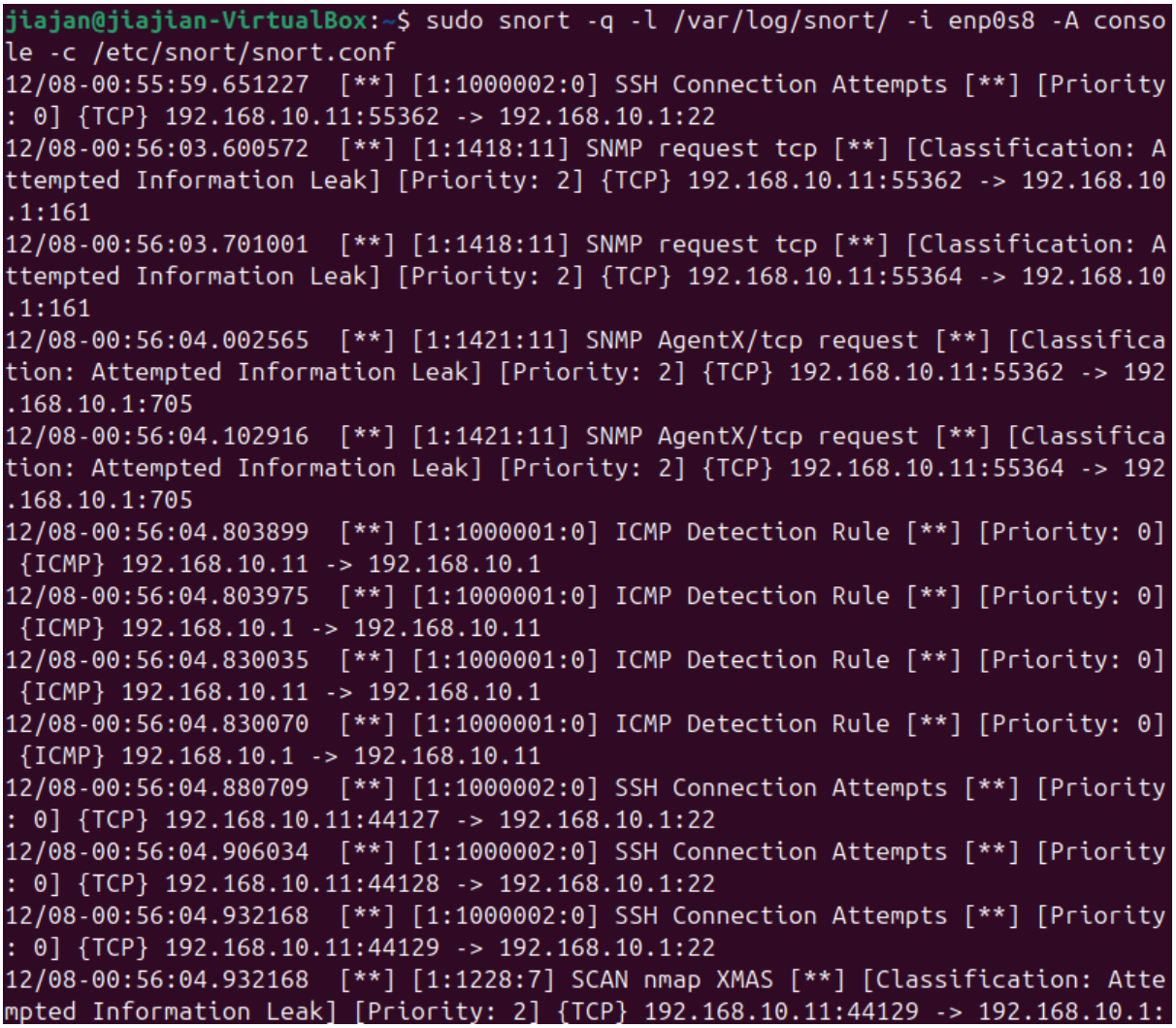

Figure 38

Figure 38 also shows all the logs.

Next, to capture HTTP Traffic using tcpdump, run tcpdump on Ubuntu and on Kali-Tester open the browser and type http://192.168.10.1:8080. The output will show GET/HTTP/1.1 and HTTP/1.1 200 OK.

```
jiajan@jiajian-VirtualBox:~$ sudo tcpdump -i enp0s8 port 8080
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
22:25:46.087013 IP 192.168.10.11.57130 > 192.168.10.1.http-alt: Flags [S], seq 4
47195380, win 64240, options [mss 1460,sackOK,TS val 1645604483 ecr 0,nop,wscale
 7], length 0
22:25:46.087158 IP 192.168.10.1.http-alt > 192.168.10.11.57130: Flags [S.], seq
3060772556, ack 447195381, win 65160, options [mss 1460,sackOK,TS val 2034182523
 ecr 1645604483,nop,wscale 7], length 0
22:25:46.087401 IP 192.168.10.11.57130 > 192.168.10.1.http-alt: Flags [.], ack 1
, win 502, options [nop,nop,TS val 1645604483 ecr 2034182523], length 0
22:25:46.087887 IP 192.168.10.11.57130 > 192.168.10.1.http-alt: Flags [P.], seq
1:338, ack 1, win 502, options [nop,nop,TS val 1645604484 ecr 2034182523], lengt
h 337: HTTP: GET / HTTP/1.1
22:25:46.088002 IP 192.168.10.1.http-alt > 192.168.10.11.57130: Flags [.], ack 3
38, win 507, options [nop,nop,TS val 2034182523 ecr 1645604484], length 0
22:25:46.088805 IP 192.168.10.1.http-alt > 192.168.10.11.57130: Flags [P.], seq
1:480, ack 338, win 507, options [nop,nop,TS val 2034182524 ecr 1645604484], len
gth 479: HTTP: HTTP/1.1 302 Found
22:25:46.089153 IP 192.168.10.11.57130 > 192.168.10.1.http-alt: Flags [.], ack 4
80, win 501, options [nop,nop,TS val 1645604485 ecr 2034182524], length 0
22:25:46.106755 IP 192.168.10.11.57130 > 192.168.10.1.http-alt: Flags [P.], seq
338:744, ack 480, win 501, options [nop,nop,TS val 1645604503 ecr 2034182524], l
ength 406: HTTP: GET /login.php HTTP/1.1
22:25:46.109372 IP 192.168.10.1.http-alt > 192.168.10.11.57130: Flags [P.], seq
480:1530, ack 744, win 504, options [nop,nop,TS val 2034182545 ecr 1645604503],
length 1050: HTTP: HTTP/1.1 200 OK
22:25:46.151073 IP 192.168.10.11.57130 > 192.168.10.1.http-alt: Flags [.], ack 1
530, win 493, options [nop,nop,TS val 1645604549 ecr 2034182545], length 0
22:25:51.129365 IP 192.168.10.1.http-alt > 192.168.10.11.57130: Flags [F.], seq
1530, ack 744, win 504, options [nop,nop,TS val 2034187565 ecr 1645604549], leng
th 0
```

Figure 39

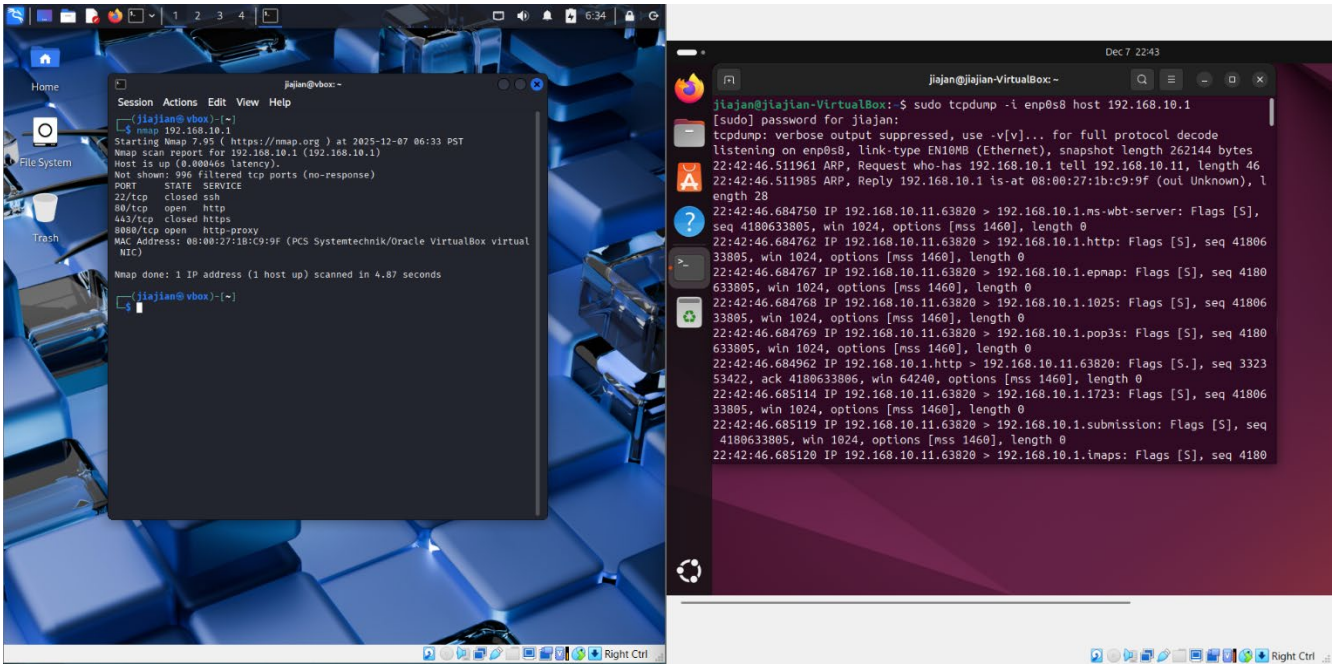

Figure 40

Figure 40 is capture nmap scan traffic, from Ubuntu run the command and on Kali-Tester run namp 192.168.10.1. The tcpdump will show the log.

5. Conclusion

In this project, I used open-source tools such as UFW, Docker, Nginx, DVWA, Snort, Nmap, and tcpdump to secure and test A2Z Corporation's network system. By setting up the firewall, running web services in containers, and monitoring traffic with Snort and tcpdump, I was able to understand how different security layers protect a real environment.

During testing, I identified three main vulnerabilities. First, port 80 was open and using HTTP without encryption, which means data could be intercepted or modified. Second, DVWA running on port 8080 was in low security mode, making it easy for attackers to exploit common web vulnerabilities. Third, the traffic to Nginx on port 80 was not captured by tcpdump, meaning Snort and monitoring tools could not detect attacks on that service. These weaknesses showed how important correct configuration and monitoring are for network security.

Based on these results, I recommended using HTTPS instead of HTTP, running DVWA only in a controlled testing environment, and making sure Nginx is bound to the correct network interface so traffic can be monitored.

Overall, the security design improved A2Z Corporation's system by reducing risks, increasing visibility, and ensuring better protection against attacks.

6. Link

GitHub Link - https://github.com/jiajian4680/6COM1040-CYBERSECURITY-PROJECT.git

7. References

1. Oracle. (2023). Oracle VM VirtualBox. VirtualBox. https://www.virtualbox.org/

2. Cisco. (2024). Cisco Packet Tracer. Netacad.com. https://www.netacad.com/cisco-packet-tracer

3. Ubuntu. (2019, June 19). The leading operating system for PCs, IoT devices, servers and the cloud | Ubuntu. Ubuntu. https://ubuntu.com/

4. Kali. (2018, December 4). Our Most Advanced Penetration Testing Distribution, Ever. Kali.org. https://www.kali.org/

5. Docker. (2020, August 14). Install Docker Engine on Ubuntu. Docker Documentation. https://docs.docker.com/engine/install/ubuntu/

6. Ahmet Numan Aytemiz. (2024, August 30). How to Install and Configure Snort on Ubuntu. Letsdefend.io; LetsDefend. https://letsdefend.io/blog/how-to-install-and-configure-snort-on-ubuntu

7. Singh Walia, A. (2024, April 16). Top 50+ Linux Commands You MUST Know | DigitalOcean. Www.digitalocean.com. https://www.digitalocean.com/community/tutorials/linux-commands

8. Boucheron, B., Camisso, J., & Abid, E. (2024, February 27). How to Set Up a Firewall with UFW on Ubuntu | DigitalOcean. Www.digitalocean.com. https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-with-ufw-on-ubuntu

9. How to install NMAP on Ubuntu? (2024). AccuWeb Cloud. https://accuweb.cloud/resource/articles/installing-nmap-on-ubuntu

10. Canonical. (2020). Ubuntu Manpage: tcpdump - dump traffic on a network. Ubuntu.com. https://manpages.ubuntu.com/manpages/jammy/man8/tcpdump.8.html

11. Cloudflare. (n.d.). Why is HTTP not secure? | HTTP vs. HTTPS. Cloudflare. https://www.cloudflare.com/learning/ssl/why-is-http-not-secure/