

Securing Network Perimeters Using Firewalls, IDS/IPS, and VPNs Using Kali Linux

Problem Statement

In today's digital environment, organizations face increasing cyber threats such as unauthorized access, port scanning, malware attacks, and denial-of-service (DoS) attacks. Many networks lack properly configured firewalls, intrusion detection/prevention systems, and secure remote access mechanisms. As a result, attackers can exploit these weaknesses to compromise sensitive data and disrupt services.

Additionally, the absence of real-time network monitoring makes it difficult to detect and respond to attacks promptly. Hence, there is a need for a comprehensive network perimeter security solution that integrates firewalls, IDS/IPS, and VPNs to protect organizational networks effectively.

Proposed Solution

This project implements a layered network perimeter security approach using Kali Linux. The solution uses:

- iptables firewall to control and restrict network traffic
- Snort IDS/IPS to detect and prevent malicious activities
- OpenVPN to provide encrypted remote access
- Wireshark and Tshark for real-time traffic monitoring and analysis

This integrated approach ensures prevention, detection, and secure communication, significantly improving network security.

1 Firewall Configuration in Kali Linux

Tool Used: iptables

Step 1: Check Firewall Status

Command:

```
sudo iptables -L
```

```
(kali㉿kali)-[~]
$ sudo iptables -L
[sudo] password for kali:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

This command is used to display the current firewall rules and policies to understand the existing security state of the system.

Step 2: Flush Existing Rules

Command:

```
sudo iptables -F
```

```
(kali㉿kali)-[~]
$ sudo iptables -L
[sudo] password for kali:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

(kali㉿kali)-[~]
$ sudo iptables -F
(kali㉿kali)-[~]
```

This step removes all existing firewall rules to start with a clean and controlled firewall configuration.

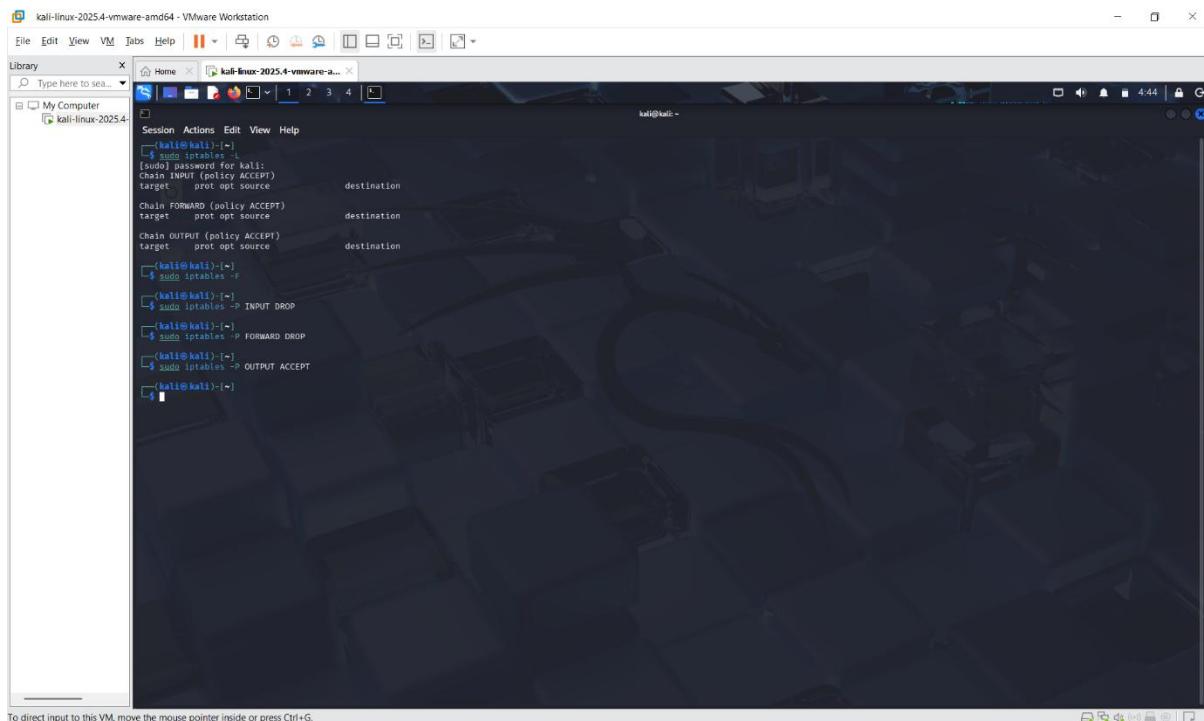
Step 3: Set Default Policies (Secure Baseline)

Commands:

```
sudo iptables -P INPUT DROP
```

```
sudo iptables -P FORWARD DROP
```

```
sudo iptables -P OUTPUT ACCEPT
```



```
[kali@kali: ~]
# sudo password for kali:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[kali@kali: ~]
$ sudo iptables -F
[kali@kali: ~]
$ sudo iptables -P INPUT DROP
[kali@kali: ~]
$ sudo iptables -P FORWARD DROP
[kali@kali: ~]
$ sudo iptables -P OUTPUT ACCEPT
[kali@kali: ~]
```

This step applies a default deny policy that blocks all incoming and forwarded traffic while allowing outgoing traffic for secure communication.

Step 4: Allow Loopback & Established Connections

Commands:

```
sudo iptables -A INPUT -i lo -j ACCEPT
```

```
sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j
ACCEPT
```

```
[kali㉿kali:~] $ sudo password for kali:  
Chain INPUT (policy ACCEPT)  
target prot opt source destination  
Chain FORWARD (policy ACCEPT)  
target prot opt source destination  
Chain OUTPUT (policy ACCEPT)  
target prot opt source destination  
[kali㉿kali:~] $ sudo iptables -F  
[kali㉿kali:~] $ sudo iptables -P INPUT DROP  
[kali㉿kali:~] $ sudo iptables -P FORWARD DROP  
[kali㉿kali:~] $ sudo iptables -P OUTPUT ACCEPT  
[kali㉿kali:~] $ sudo iptables -A INPUT -i lo -j ACCEPT  
[kali㉿kali:~] $ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT  
[kali㉿kali:~]
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

This step allows internal system communication and permits responses to already established network connections.

Step 5: Allow Essential Services

Commands:

```
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

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

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

```
[kali㉿kali:~] $ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT  
[kali㉿kali:~] $ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT  
[kali㉿kali:~] $ sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT  
[kali㉿kali:~]
```

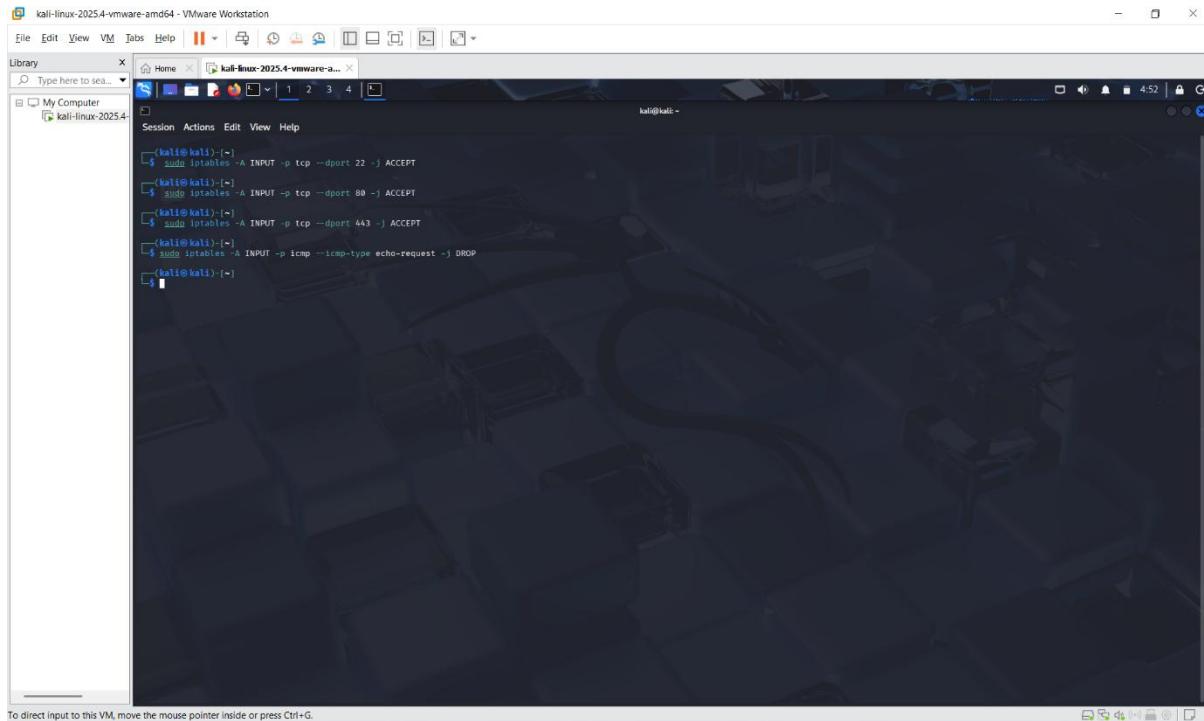
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

This step allows essential services such as secure shell (SSH) and web services to function while keeping other ports blocked.

Step 6: Block Ping & Scan Attempts

Command:

```
sudo iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
```



The screenshot shows a terminal window titled 'kali-linux-2025.4-vmware-amd64 - VMware Workstation'. The terminal is running on a Kali Linux system. The user has entered several iptables commands to set up a firewall. The commands are as follows:

```
(kali㉿kali)-[~]
$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
(kali㉿kali)-[~]
$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
(kali㉿kali)-[~]
$ sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
(kali㉿kali)-[~]
$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
(kali㉿kali)-[~]
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

This step blocks ICMP echo requests to prevent network scanning and reconnaissance attacks.

✓ Firewall configured successfully

2 IDS / IPS Setup Using Snort (Kali Linux)

Tool Used: Snort

Step 1: Install Snort

```
sudo apt update
```

```
sudo apt install snort -y
```

```

kali@kali: ~
[~]# sudo apt update
[~]# sudo apt install snort -y
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [34.6 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [20.7 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [52.1 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [115 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [254 kB]
Fetched 75.1 MB in 1min 14s (7.8 kB/s)
1446 packages can be upgraded. Run 'apt list --upgradable' to see them.
Installing:
snort

Installing dependencies:
[libdmg] libfastjson5 liblognorm5 oinkmaster rsyslog snort-common snort-common-libraries snort-rules-default

Suggested packages:
rsyslog-doc rsyslog-pgsql rsyslog-elasticsearch rsyslog-hiredis rsyslog-kubernetes rsyslog-clickhouse | rsyslog-gnutls rsyslog-repl
rsyslog-mysql rsyslog-mongodb rsyslog-smp rsyslog-docker rsyslog-openssl rsyslog-gssapi snort-doc

Summary:
Upgrading: 0, Installing: 10, Removing: 0, Not Upgrading: 1446
Download size: 4.851 kB
Space needed: 17.6 MB
Space available: 51.0 GB available

Get:10 http://mirrors.estointernet.in/kali kali-rolling/main amd64 oinkmaster all 2.0-5 [79.5 kB]
Get:11 http://mirrors.estointernet.in/kali kali-rolling/main amd64 libdmg 0.1.11-2+b1 [9.688 kB]
Selecting previously unselected package snort-common-libraries.
(Reading database ... 4394/47 files and directories currently installed.)
Preparing to unpack .../0-snort-common-libraries_3.10.0.0-0kali1_amd64.deb ...
Unpacking snort-common-libraries (3.10.0.0-0kali1) ...
Selecting previously unselected package snort-rules-default.
Preparing to unpack .../1-snort-rules-default_3.10.0.0-0kali1_all.deb ...
Unpacking snort-rules-default (3.10.0.0-0kali1) ...
Selecting previously unselected package snort-common.
Preparing to unpack .../2-snort-common_3.10.0.0-0kali1_all.deb ...
Unpacking snort-common (3.10.0.0-0kali1) ...
Selecting previously unselected package libfastjson5:amd64.
Preparing to unpack .../3-libfastjson5_0.1.11-2+b1_amd64.deb ...
Unpacking libfastjson5:amd64 (0.1.11-2+b1) ...
Selecting previously unselected package liblognorm5:amd64.
Preparing to unpack .../4-liblognorm5_2.0.9-1_amd64.deb ...
Unpacking liblognorm5:amd64 (2.0.9-1) ...
Selecting previously unselected package libdmg:amd64.
Preparing to unpack .../5-libdmg_0.1.11-2+b1_amd64.deb ...

```

This step installs Snort on Kali Linux to enable intrusion detection and prevention capabilities.

Step 2: Configure Network Interface

Commands:

`sudo nano /etc/snort/snort.lua`

Change it to:

`HOME_NET = '192.168.111.0/24'`

```

kali@kali: ~
[~]# nano /etc/snort/snort.lua
GNU nano 0.6
-- Snort++ configuration
there are over 200 modules available to tune your policy.
use the help module for any syntax configuration.
use this conf as a template for your specific configuration.

1: configure defaults
2: configure inspection
3: configure performance
4: configure detection
5: configure detection
6: configure outputs
7: configure tweaks

1: configure defaults
-- HOME_NET and EXTERNAL_NET must be set now
setup the network addresses you are protecting
HOME_NET = '192.168.111.0/24'
setup the external network addresses.
(leave as "Any" in most situations)
EXTERNAL_NET = 'any'

include 'snort_defaults.lua'

2: configure inspection
mod = {} uses internal defaults
you can see them with snort -h --help-module mod
mod = default.mod uses external defaults
you can see them in snort_defaults.lua

the following are quite capable with defaults:
stream {
    stream_ip = {}
    stream_tcp = {}
    stream_udp = {}
    stream_user = {}
    stream_file = {}
}

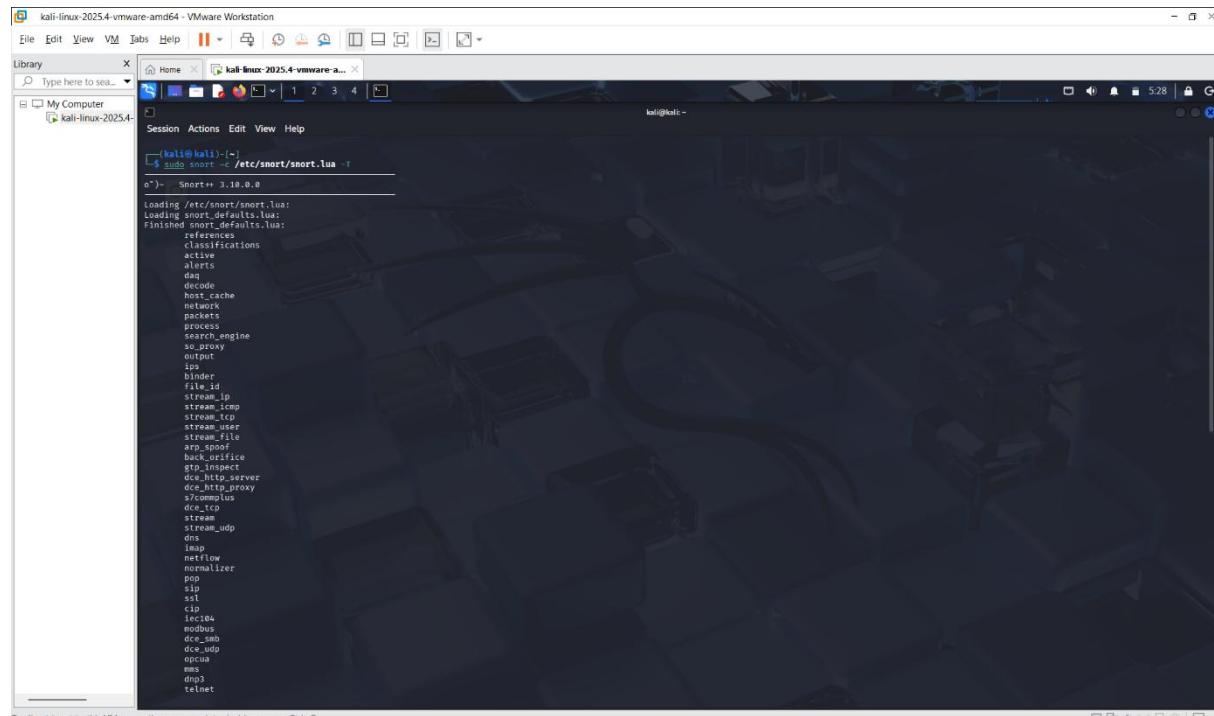
```

This step defines the local network (HOME_NET) so Snort can monitor and analyze internal traffic accurately.

Step 3: Enable and Verify Snort Rules

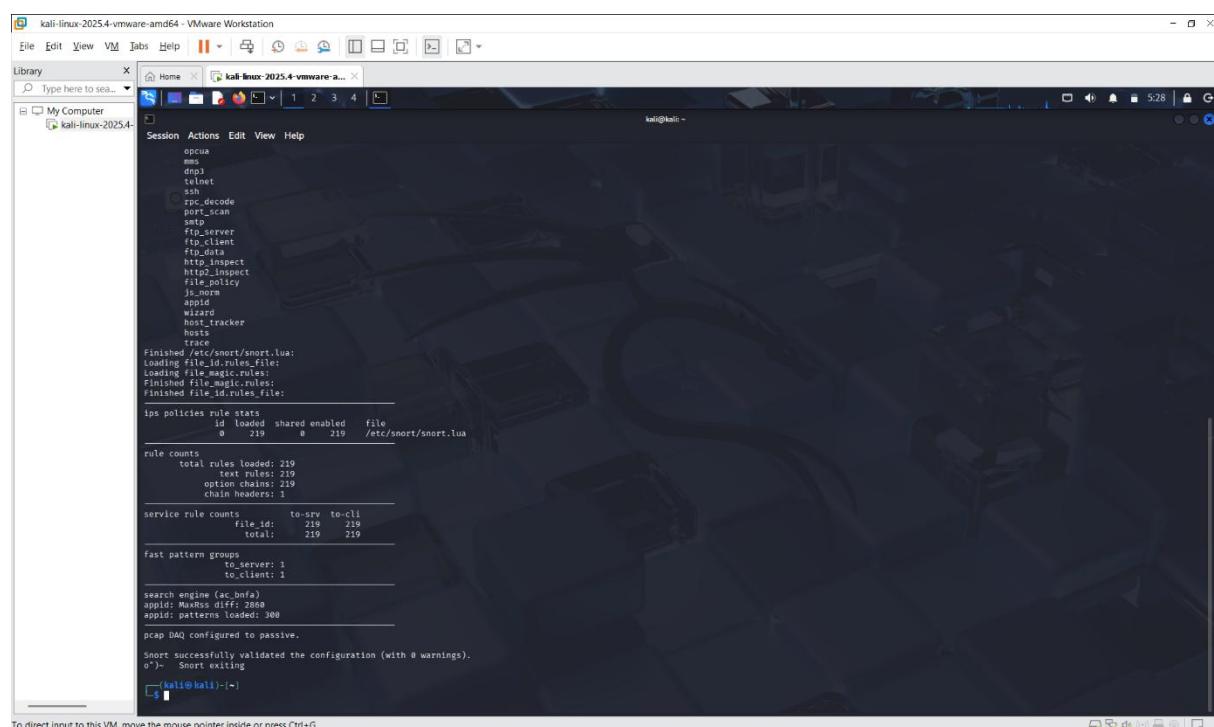
Commands:

```
sudo snort -c /etc/snort/snort.lua -T
```



```
kali@kali:~$ sudo snort -c /etc/snort/snort.lua -T
o)->  Snort= 3.19.0.8
Loading /etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
    references
    classifications
    active
    alerts
    tags
    decode
    host_cache
    network
    packets
    process
    search_engine
    so_proxy
    output
    ips
    binder
    file_id
    stream_ip
    stream_icmp
    stream_tcp
    stream_udp
    stream_file
    arp_spoofer
    httpd_inspect
    dce_http_server
    dce_http_proxy
    stream_dos
    dce_tcp
    stream
    stream_udp
    dns
    imap
    netflow
    normalizer
    pod
    sip
    ssl
    tcp
    iec184
    mosh
    dce_udp
    opcuia
    max
    dns3
    telnet

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
```



```
kali@kali:~$ sudo snort -c /etc/snort/snort.lua -T
o)->  Snort= 3.19.0.8
Loading /etc/snort/snort.lua:
Loading file_magic.rules:
Loading file_id.rules:
Finished file_id.rules_file:
Finished /etc/snort/snort.lua:
Loading file_id.rules_file:
Loading file_magic.rules:
Finished file_magic.rules:
Finished file_id.rules_file:
ips policies rule stats
    id loaded shared enabled   file
    0     219      0     219   /etc/snort/snort.lua
rule counts
    total rules loaded: 219
        text rules: 219
        option chains: 219
        chain headers: 1
service rule counts
    file_id:  219  219
    total:   219  219
fast pattern groups
    to_server: 1
    to_client: 1
search engine (ac_bnf)
apid: MaxRss diff: 2888
apid: patterns loaded: 300
pcap DAO configured to passive.

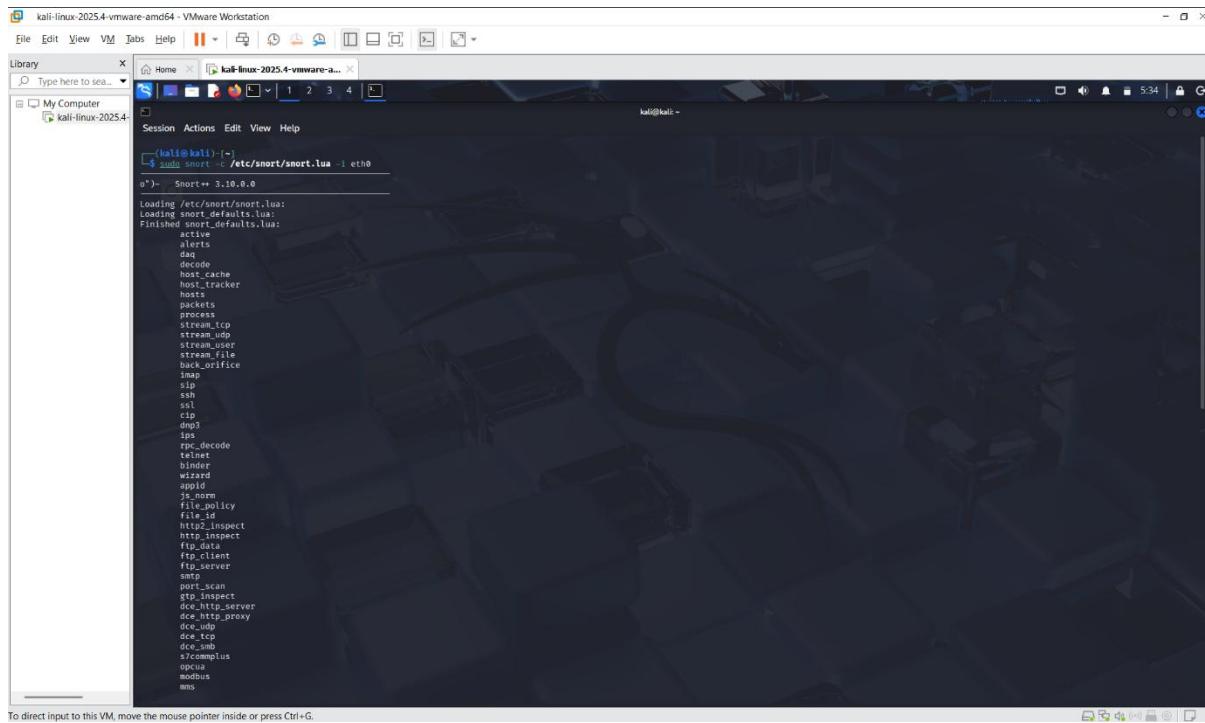
Snort successfully validated the configuration (with 0 warnings).
o)->  Snort exiting
[4]
```

This step checks and validates the Snort configuration and rule files to ensure there are no errors.

Step 4: Run Snort in IDS Mode

Commands:

```
sudo snort -c /etc/snort/snort.lua -i eth0
```



The screenshot shows a terminal window titled "kali-linux-2025.4-vmware-amd64 - VMware Workstation". The terminal displays the output of the command "sudo snort -c /etc/snort/snort.lua -i eth0". The output shows the loading of configuration files and various modules, including alert, daf, decode, host_exch, host_tracker, hosts, packets, process, stream_tcp, stream_udp, stream_sip, stream_file, back_orifice, imap, sip, ssh, ssl, cdp, dns3, ipp, rpc_decode, telnet, binder, wizard, apollo, ls_norm, file_policy, file_type, http_inspect, http_inspect, ftp_data, ftp_client, ftp_server, smtp, port_scan, ipp_inspect, dce_http_server, dce_http_proxy, dce_ntp, dce_tcp, dce_smb, s7complus, opentdb, modbus, mms. The terminal also includes a message: "To direct input to this VM, move the mouse pointer inside or press Ctrl+G."

This step runs Snort in Intrusion Detection System (IDS) mode to monitor network traffic and generate alerts.

Step 5: Test IDS Functionality

Commands:

```
nmap -sS 192.168.111.0/24
```

The screenshot shows a Kali Linux desktop environment within a VMware Workstation window. The desktop has a dark theme with a 3D cube background. A terminal window is open in the foreground, displaying Snort module statistics and a completed nmap scan. The terminal output includes:

```

Module Statistics
apid
    packets: 1
    processed_packets: 1
    total_sessions: 1

binder
    new_flows: 1
    inspects: 1

detection
    analyzed: 2

port_scan
    packets: 2
    trackers: 4

stream
    flows: 1

stream_icmp
    sessions: 1
    max: 1
    created: 1
    released: 1

udp
    bad_udp4_checksum: 1

Summary Statistics
process
    signals: 2

timing
    runtime: 00:00:38
    seconds: 38.058558
o> Snort exiting!
<-[kali㉿kali]->
$ nmap -sS 192.168.111.137
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-21 09:34 EST
Nmap scan report for 192.168.111.137
Host is up (0.00001s latency).
All 1000 scanned ports on 192.168.111.137 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds

```

This step simulates a port scanning attack to verify that Snort detects and alerts suspicious activities.

✓ Attacks are now blocked automatically

3 VPN Setup Using OpenVPN (Kali Linux)

Tool Used: OpenVPN

Step 1: Install OpenVPN

Commands:

```
sudo apt install openvpn easy-rsa -y
```

The screenshot shows a Kali Linux desktop environment within a VMware Workstation window. A terminal window is open, displaying the command `sudo apt install openvpn easy-rsa -y` being run. The terminal output shows the package manager fetching files from the Kali repository and installing the specified packages. The output includes:

```

[kali㉿kali]-> sudo apt install openvpn easy-rsa -y
[sudo] password for kali:
Upgrading: 2, Installing: 0, Removing: 0, Not Upgrading: 1444
Download size: 767 kB
All files already downloaded / 0.0 GB available

Get:1 http://kali.download/kali kali-rolling/main amd64 easy-rsa all 3.2.5-1 [81.3 kB]
Get:2 http://http.kali.org/kali kali-rolling/main amd64 openvpn amd64 2.7.0-rc5-1 [686 kB]
Preconfiguring packages ...
(Reading database ... 439990 files and directories currently installed.)
Preparing to unpack .../easy-rsa_3.2.5-1_all.deb ...
Unpacking easy-rsa (3.2.5-1) over (3.2.4-1) ...
Preparing to unpack .../openvpn_2.7.0-rc5-1_amd64.deb ...
Unpacking openvpn (2.7.0-rc5-1) over (2.7.0-rc5-2) ...
Setting up easy-rsa (3.2.5-1) ...
Processing triggers for man-db (2.19.1-1) ...
Processing triggers for kali-menu (2020.4-3) ...

```

This step installs OpenVPN and Easy-RSA tools required to create encrypted VPN tunnels and manage certificates.

Step 2: Create Certificate Authority (CA) and VPN Certificates

Commands:

```
make-cadir ~/openvpn-ca  
cd ~/openvpn-ca  
.easyrsa init-pki  
.easyrsa build-ca  
.easyrsa build-server-full server nopass  
.easyrsa build-client-full client nopass  
.easyrsa gen-dh
```

```
(kali㉿kali)-[~/openvpn-ca]
$ ./openvpn-ca
Using Easy-RSA 'vars' configuration:
* /home/kali/openvpn-ca/openssl.cnf
Please check over the details shown below for accuracy. Note that this request has not been cryptographically verified. Please be sure it came from a trusted source or that you have verified the request checksum with the sender.
You are about to sign the following certificate:

Requested CN: 'kali'
Requested type: 'server'
Valid for: '925' days

subject:
    commonName = kali
Type the word 'yes' to continue, or any other input to abort.
Confirm requested details: yes

Using configuration from /home/kali/openvpn-ca/pki/2a8f89cf/temp.00
Enter pass phrase for /home/kali/openvpn-ca/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName: ASN.1 12: "kali"
Certificate is to be certified until Apr 26 04:28:30 2028 GMT (925 days)
Write out database with 1 new entries
Database updated
WARNING
=====

INCOMPLETE Inline file created:
* /home/kali/openvpn-ca/pki/inline/private/server.inline

Notice
-----
Certificate created at:
* /home/kali/openvpn-ca/pki/issued/server.crt

(kali㉿kali)-[~/openvpn-ca]
$ ./easyrsa gen-dh
Using Easy-RSA 'vars' configuration:
* /home/kali/openvpn-ca/vars
Generating DH parameters, 2048 bit long safe prime
.....
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

```
(kali㉿kali)-[~/openvpn-ca]
File Edit View VM Tabs Help | 1 2 3 4
Library Type here to sea...
My Computer kali-linux-2025.4...
Session Actions Edit View Help
(kali㉿kali)-[~/openvpn-ca]
$ ./openvpn-ca
Your files are:
* req: /home/kali/openvpn-ca/pki/reqs/client.req
* key: /home/kali/openvpn-ca/pki/private/client.key

Using Easy-RSA 'vars' configuration:
* /home/kali/openvpn-ca/vars
please check over the details shown below for accuracy. Note that this request has not been cryptographically verified. Please be sure it came from a trusted source or that you have verified the request checksum with the sender.
You are about to sign the following certificate:

Requested CN: 'kali'
Requested type: 'client'
Valid for: '925' days

subject:
    commonName = kali
Type the word 'yes' to continue, or any other input to abort.
Confirm requested details: yes

Using configuration from /home/kali/openvpn-ca/pki/77aee1a9/temp.00
Enter pass phrase for /home/kali/openvpn-ca/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName: ASN.1 12: "kali"
Certificate is to be certified until Apr 26 04:29:52 2028 GMT (925 days)
Write out database with 1 new entries
Database updated
WARNING
=====

INCOMPLETE Inline file created:
* /home/kali/openvpn-ca/pki/inline/private/client.inline

Notice
-----
Certificate created at:
* /home/kali/openvpn-ca/pki/issued/client.crt

(kali㉿kali)-[~/openvpn-ca]
$ sudo nano /etc/openvpn/server/server.conf
(kali㉿kali)-[~/openvpn-ca]
$ sudo systemctl start openvpn-server@server
(kali㉿kali)-[~/openvpn-ca]
```

This step generates the Certificate Authority, server certificate, client certificate, and Diffie-Hellman parameters for secure authentication.

Step 3: Configure OpenVPN Server

Commands:

```
sudo nano /etc/openvpn/server/server.conf
```

Step 4: Start VPN Server

Commands:

```
sudo systemctl start openvpn@server
```

```
kali-linux-2025.4-vmware-amd64 - VMware Workstation
File Edit View VM Tab Help ||| ☰ 🌐 🕒 📁 🗃 🖼 🖼 🖼 🖼 🖼

Library Type here to search... kali-linux-2025.4-vmware-a...
File My Computer kali-linux-2025.4 Session Actions Edit View Help
Your files are:
* req: /home/kali/openvpn-ca/pki/reqs/client.req
* key: /home/kali/openvpn-ca/pki/private/client.key

Using Easy-RSA 'vars' configuration:
/home/kali/openvpn-ca/vars
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.
You are about to sign the following certificate:

Requested CN: 'kali'
Requested type: 'client'
Valid for: "825" days

subject:
    commonname = kali

Type the word 'yes' to continue, or any other input to abort.
Confirm requested details: yes

Using configuration from /home/kali/openvpn-ca/pki/77a4e1a9/temp_00
Enter pass phrase for /home/kali/openvpn-ca/pki/private/ca.key:
Checking the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName : ASN.1 12:'Kali'
Certificate is to be certified until Apr 26 04:29:52 2028 GMT (825 days)

Write out database with 1 new entries
Database updated

WARNING

INCOMPLETE Inline file created:
* /home/kali/openvpn-ca/pki/inline/private/client.inline

Notice

Certificate created at:
* /home/kali/openvpn-ca/pki/issued/client.crt

└─(kali㉿kali)-~/openvpn-ca
└─$ sudo nano /etc/openvpn/server.conf

└─(kali㉿kali)-~/openvpn-ca
└─$ sudo systemctl start openvpn-server@server

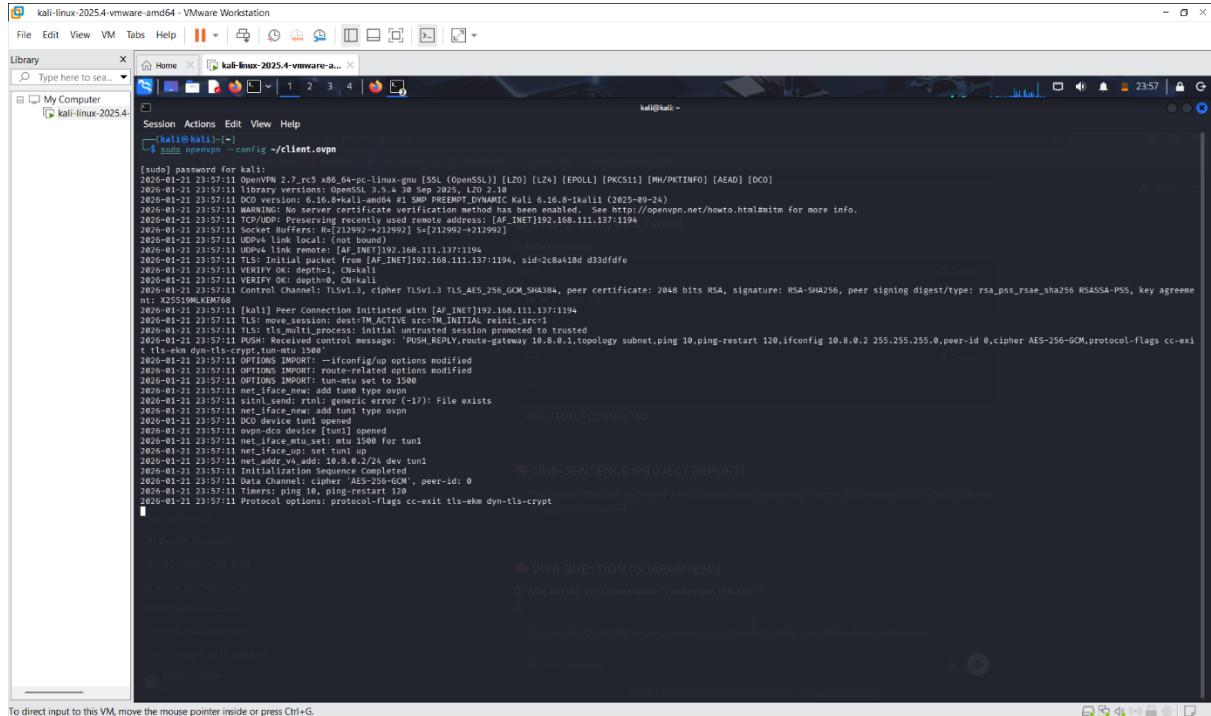
└─(kali㉿kali)-~/openvpn-ca
```

This command starts the OpenVPN server service, allowing the system to accept secure VPN connections.

Step 5: Connect VPN Client

Commands:

```
sudo openvpn --config client.ovpn
```



```
[kali@kali:~] kali@kali:~$ sudo openvpn --config ~/client.ovpn
[sudo] password for kali:
2026-01-21 23:57:11 OpenVPN 2.7_rc5 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] [DCC]
2026-01-21 23:57:11 Library Version: OpenSSL 3.0.4 50 Sep 2025, LZO 2.10
2026-01-21 23:57:11 DCC Peer Version: 6.16.8-1kali-0md5 #1 SMP PREEMPT_DYNAMIC Kali 6.16.8-1kali1 (2025-09-24)
2026-01-21 23:57:11 WARNING: No server certificate verification method has been enabled. See http://openvpn.net/howto.html#mitm for more info.
2026-01-21 23:57:11 TCP/UDP: Preserving recently used remote address: [AF_INET]192.168.1.113:1194
2026-01-21 23:57:11 DNS: 192.168.1.113#1194 [AF_INET]192.168.1.113:1194
2026-01-21 23:57:11 UDPv4 Link local: (not bound)
2026-01-21 23:57:11 UDPv4 Link remote: ([AF_INET]192.168.1.113:1194
2026-01-21 23:57:11 UDPv4 Route: [AF_INET]192.168.1.113:1194 [AF_INET]192.168.1.113:1194, sid:2c8a41bd d33dfdfde
2026-01-21 23:57:11 VERIFY OK: depth=0, CN=kali
2026-01-21 23:57:11 Control Channel: TLSv1.3 TLS_AES_256_GCM_SHA384, peer certificate: 2048 bits RSA, signature: RSA-SHA256, peer signing digest/type: rsa_pss_rsae_sha256 RSASSA-PSS, key agreement: RSA-2551MLUEM68
2026-01-21 23:57:11 [Kali] Peer Connection Initiated with [AF_INET]192.168.1.113:1194
2026-01-21 23:57:11 move_session: dest:TM_ACTIVE src:TM_INITIAL reinit_src:1
2026-01-21 23:57:11 Received control message: PUSH_REPLY,route-gateway 10.8.0.1,topology subnet,ping 10,ping-restart 120,ifconfig 10.8.0.2 255.255.255.0,peer-id 0,cipher AES-256-GCM,protocol-flags cc-exit ttls-ekm dyn-tls-crypt,tun-mtu 1584
2026-01-21 23:57:11 OPTIONS IMPORT: --ifconfig/up --script-security 2 options modified
2026-01-21 23:57:11 OPTIONS IMPORT: --proto udp options modified
2026-01-21 23:57:11 PUSH: Received control message: PUSH_REPLY,route-gateway 10.8.0.1,topology subnet,ping 10,ping-restart 120,ifconfig 10.8.0.2 255.255.255.0,peer-id 0,cipher AES-256-GCM,protocol-flags cc-exit ttls-ekm dyn-tls-crypt,tun-mtu 1584
2026-01-21 23:57:11 net_iface_new: add tun type ovpn
2026-01-21 23:57:11 net_iface_set: set tun type ovpn
2026-01-21 23:57:11 net_iface_up: set tun type ovpn
2026-01-21 23:57:11 DCC device tun1 opened
2026-01-21 23:57:11 ovpn-dcc device [tun1] opened
2026-01-21 23:57:11 net_iface_up: set tun1 up
2026-01-21 23:57:11 net_iface_v4_addr: 10.8.0.2/24 dev tun1
2026-01-21 23:57:11 net_iface_v4_dhcp: 10.8.0.2/24 dev tun1
2026-01-21 23:57:11 Data Channel cipher: "AES-256-GCM", peer-id: 0
2026-01-21 23:57:11 Timers: ping 10, ping-restart 120
2026-01-21 23:57:11 Protocol options: protocol-flags cc-exit ttls-ekm dyn-tls-crypt
[...]
```

This command connects the VPN client to the server and establishes a secure encrypted tunnel.

A secure encrypted VPN tunnel was successfully established between the client and the server using OpenVPN.

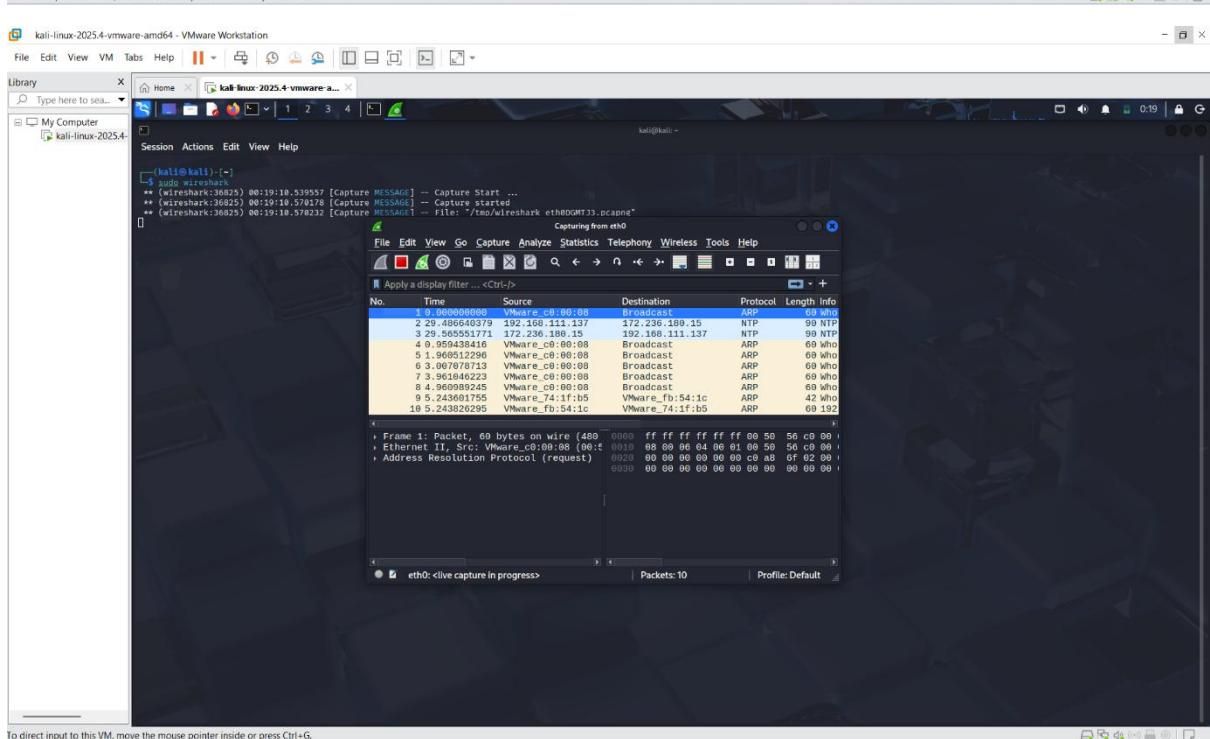
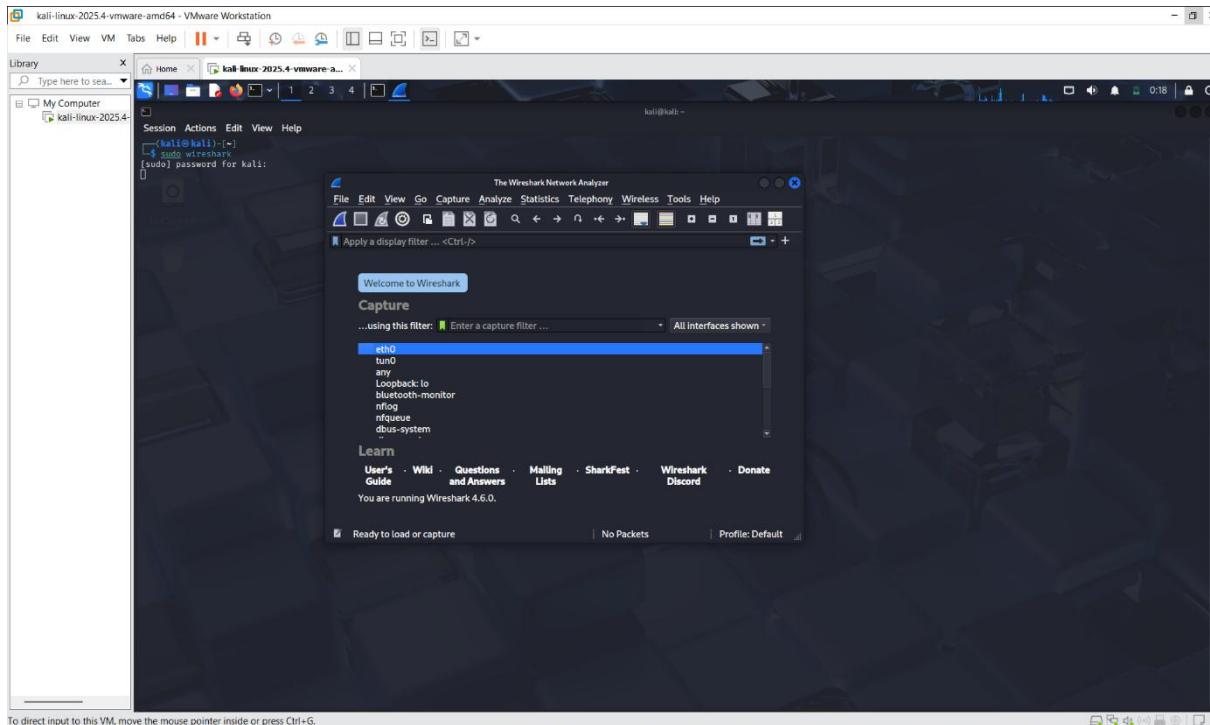
4 Traffic Monitoring Using Wireshark & Tshark

Tools Used: Wireshark, Tshark

Step 1: Start Traffic Capture (GUI Mode)

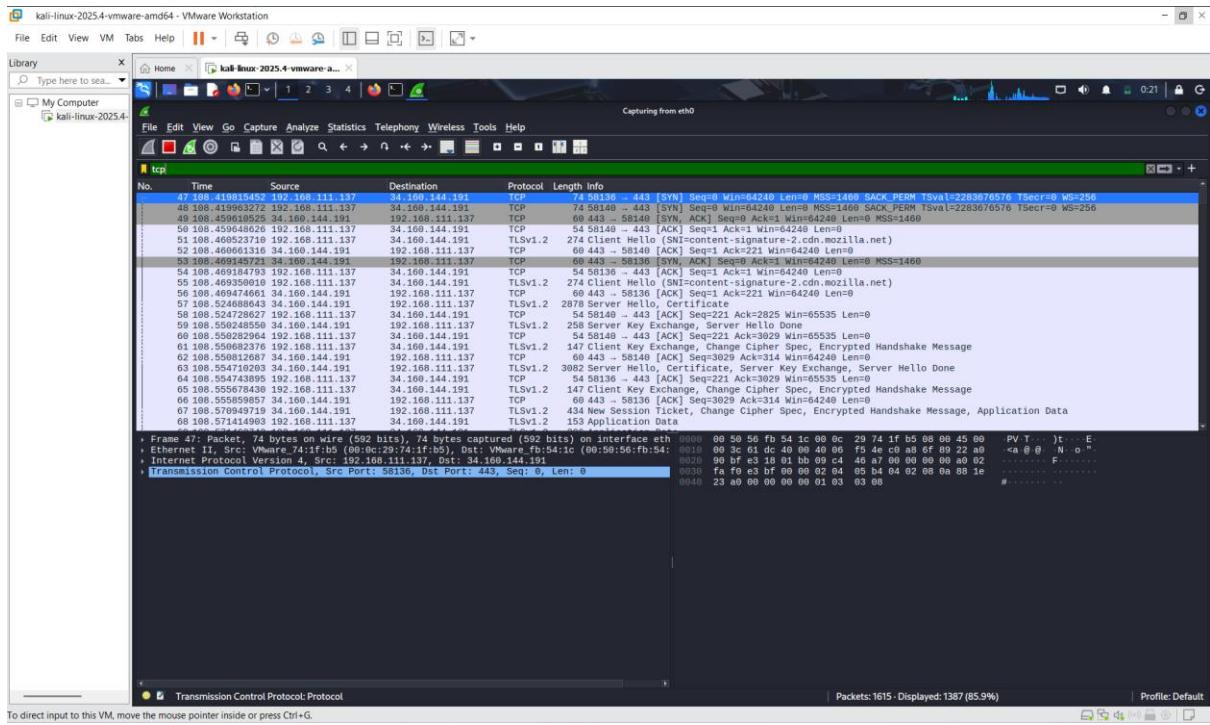
Commands:

```
sudo wireshark
```



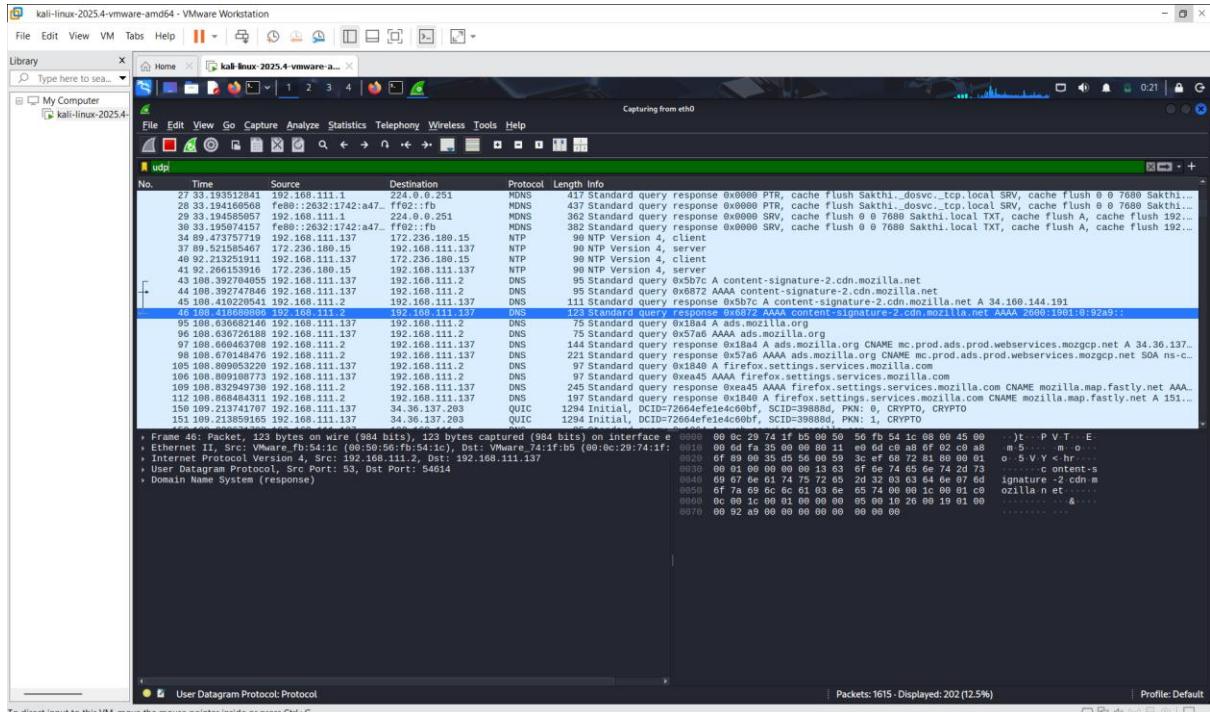
This command launches Wireshark to capture and visually analyze network packets in real time.

Step 2: Apply TCP Filter



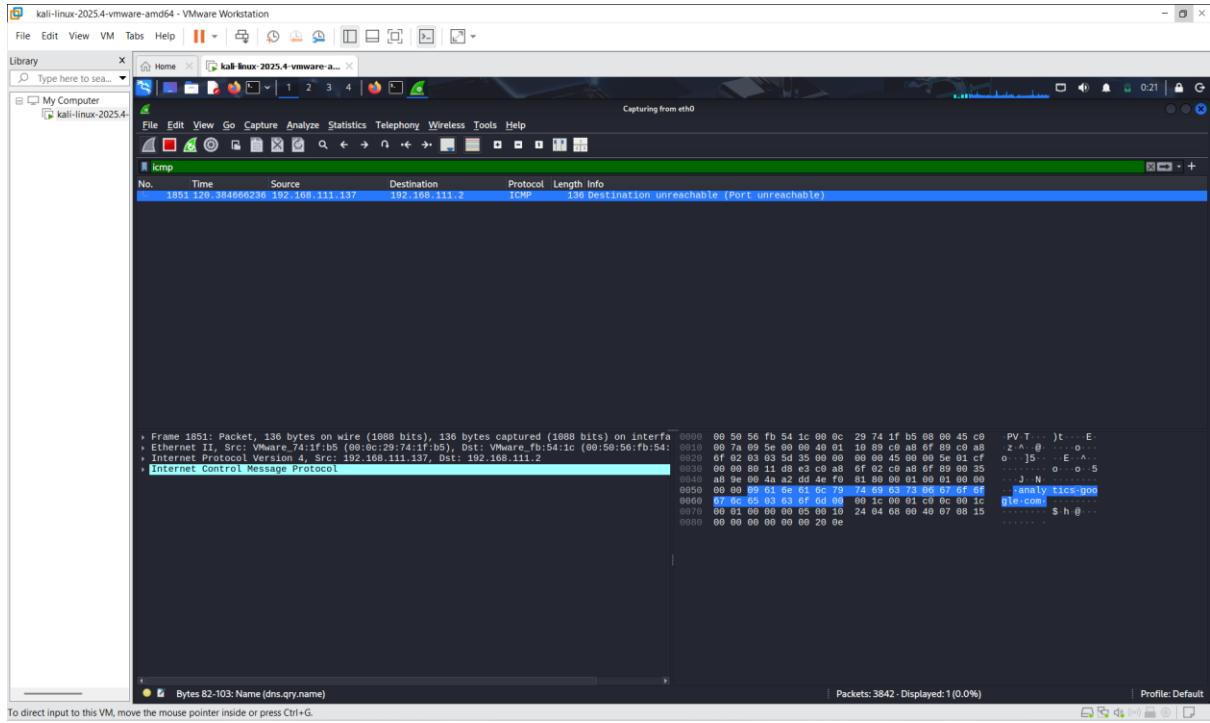
This filter displays only TCP packets to analyze connection-oriented traffic such as HTTP and HTTPS.

Step 3: Apply UDP Filter



This filter shows UDP traffic to monitor services like DNS and VPN communication.

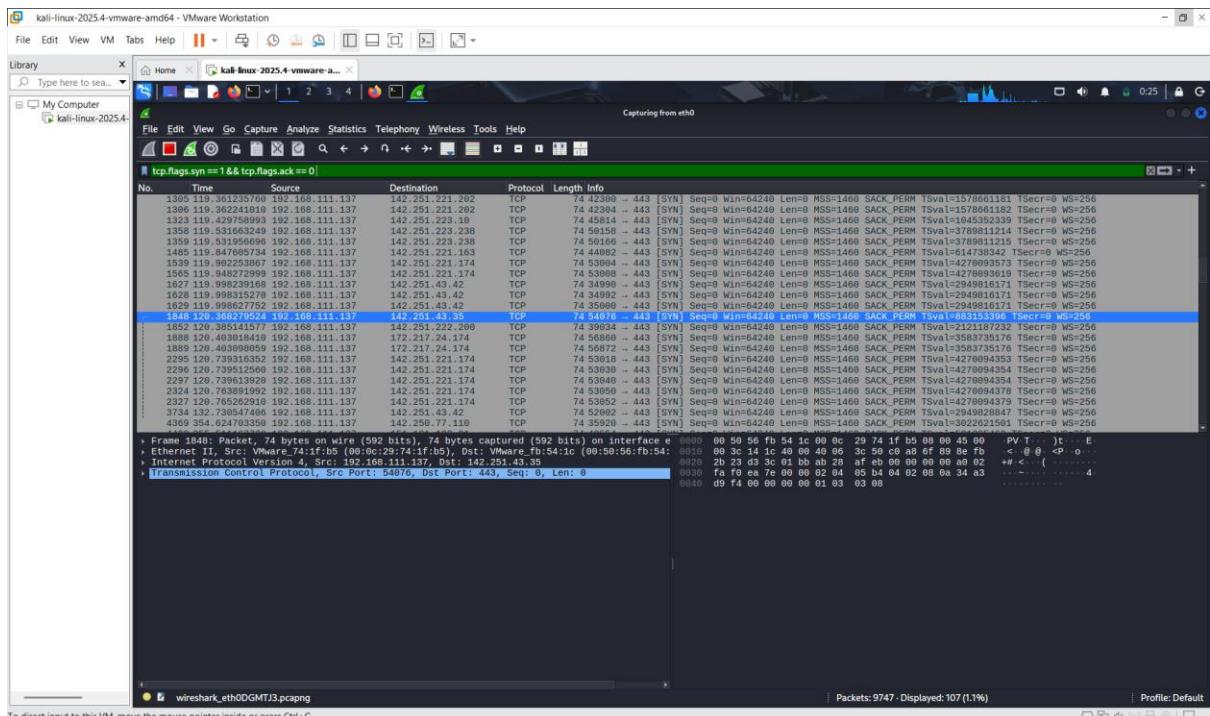
Step 4: Apply ICMP Filter



This filter is used to identify ICMP packets such as ping requests and network scans.

Step 5: Identify Suspicious Traffic

`tcp.flags.syn == 1 && tcp.flags.ack == 0`



This filter helps detect potential port-scanning attacks by identifying repeated TCP SYN packets.

Network traffic was successfully captured and analyzed using Wireshark and Tshark to identify suspicious and malicious activities.

5 Incident Response Plan (Kali Linux Based)

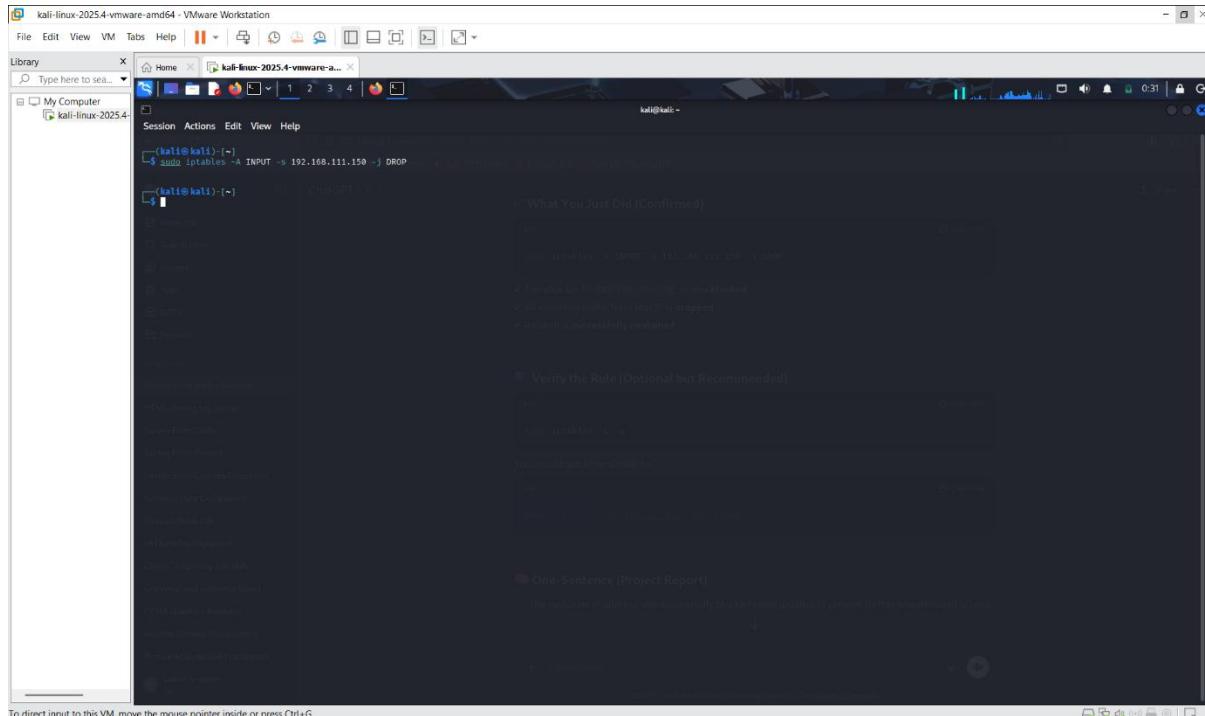
Detection

Security incidents are detected using Snort alerts and abnormal traffic patterns observed in Wireshark.

Containment

Commands:

```
sudo iptables -A INPUT -s 192.168.111.150 -j DROP
```



This step blocks the attacker's IP address using firewall rules to prevent further malicious access.

Eradication

Malicious services are stopped and compromised files are removed to eliminate the threat from the system.

Recovery

Affected services are restarted and the network is continuously monitored to ensure normal operation is restored.

Documentation

All incident details, logs, and actions taken are documented to prepare an incident response report.

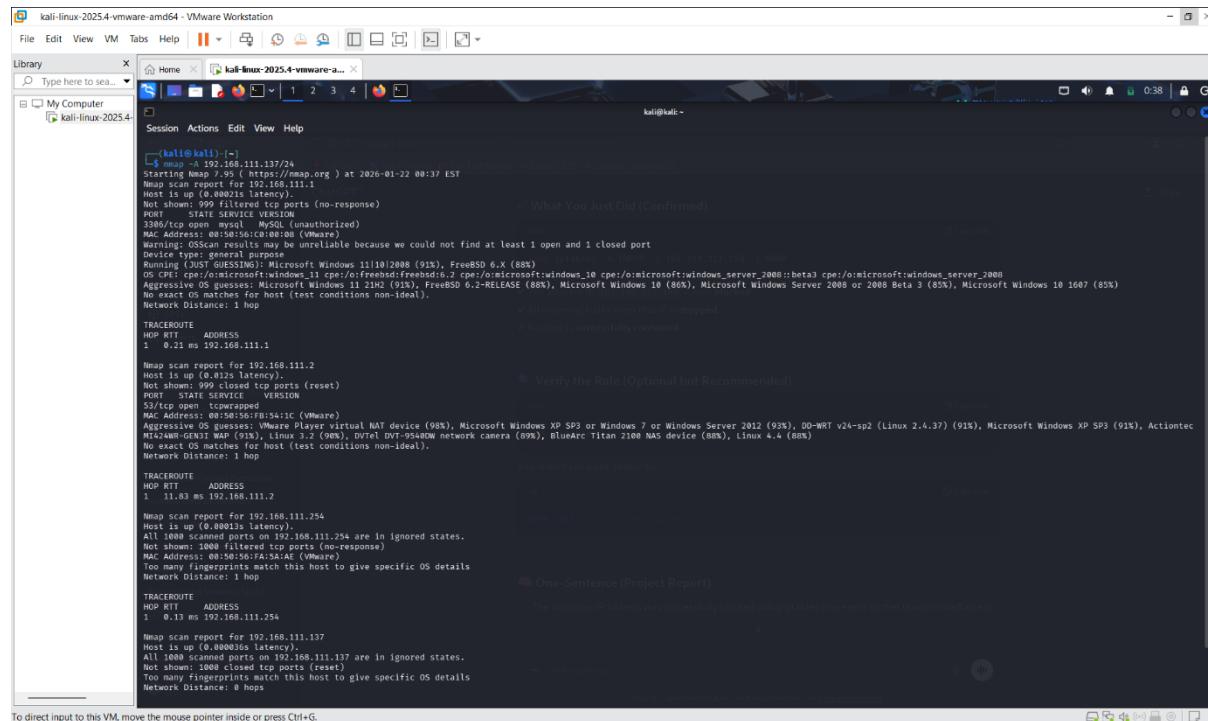
The incident response plan ensures timely detection, containment, eradication, recovery, and documentation of network security incidents.

6 Testing & Demonstration (Mandatory)

Simulated Attacks

Commands:

```
sudo nmap -A 192.168.111.137/24
```



```
[kali㉿kali] ~- [+] Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-22 08:37 EST
Nmap scan report for 192.168.111.1
Host is up (0.00021s latency).
Not shown: 1000 filtered ports (no-response)
PORT      STATE SERVICE VERSION
3306/tcp  open  mysql  MySQL (unauthorized)
80/tcp    open  http   Apache httpd/2.4.42.1
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSED): Microsoft Windows 11/10 2008 (91%), FreeBS 6.X (88%)
OSCPE: cpe:/microsoft:windows_10_cpe:/microsoft:windows_server_2008_beta3 cpe:/microsoft:windows_server_2008
Aggressive OS guesses: Microsoft Windows 11 21H2 (91%), FreeBS 0.2-RELEASE (88%), Microsoft Windows Server 2008 or 2008 Beta 3 (85%), Microsoft Windows 10 1607 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1  0.21 ms  192.168.111.1

Nmap scan report for 192.168.111.2
Host is up (0.0027s latency).
Not shown: 1000 filtered ports (reset)
PORT      STATE SERVICE VERSION
53/tcp    open  tcptraceroute
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSED): VMware Player virtual NAT device (98%), Microsoft Windows XP SP3 or Windows 7 or Windows Server 2012 (93%), DD-WRT v24-sp2 (Linux 2.4.37) (91%), Microsoft Windows XP SP3 (91%), Actiontec MT42WR-GEN3 WAP (91%), Linux 3.2 (98%), D-Link DWT-954DN network camera (89%), BlueArc Titan 210B NAS device (88%), Linux 4.4 (88%)
Aggressive OS guesses: Microsoft Windows 11 21H2 (91%), FreeBS 0.2-RELEASE (88%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1  0.13 ms  192.168.111.2

Nmap scan report for 192.168.111.254
Host is up (0.0003s latency).
All 1000 scanned ports on 192.168.111.254 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
MAC Address: VMware Player (VMware)
Too many fingerprints match this host to give specific OS details
Network Distance: 0 hops

Verify the Rule (Optional but Recommended)

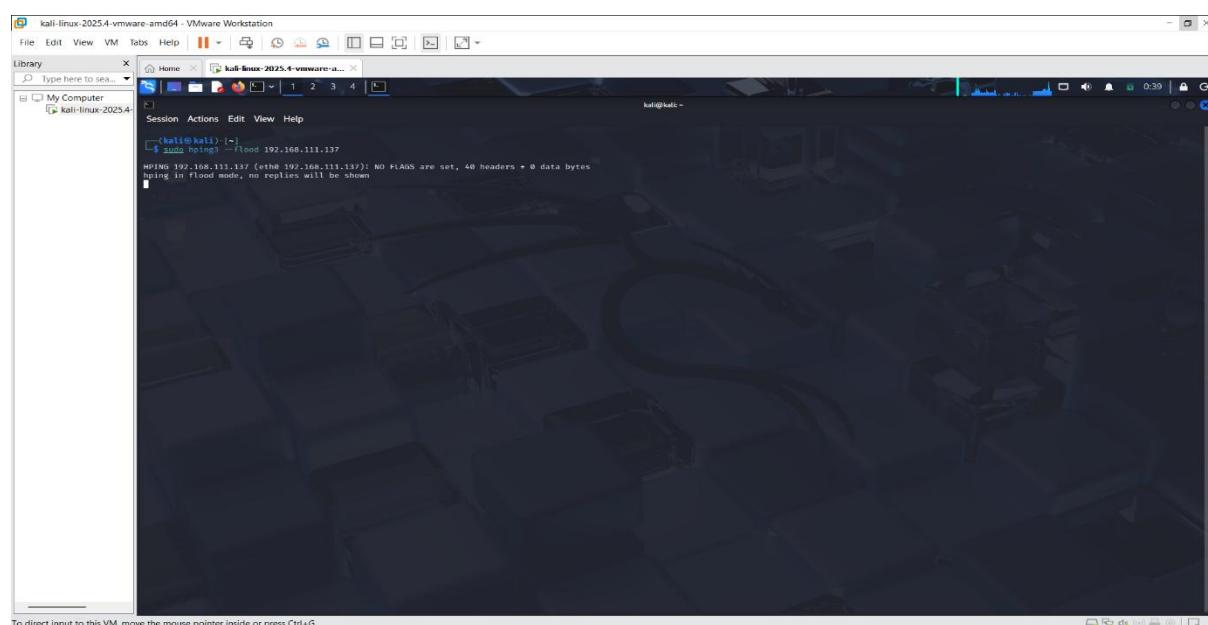
One-Sentence (Project Report)

The malicious IP address was successfully blocked using iptables to prevent further unauthorized access.
```

An aggressive Nmap scan was performed to identify open ports, services, and operating systems within the network.

Commands:

```
sudo hping3 --flood 192.168.111.137
```



```
[kali㉿kali] ~- [+] sudo hping3 --flood 192.168.111.137
HPING 192.168.111.137 (eth0 192.168.111.137): NO FLAGS are set, 40 headers + 0 data bytes
Flood in Flood mode, replies will be shown
■
```

A SYN flood attack was simulated using hping3 to test firewall and intrusion prevention mechanisms.

Firewall Response

The firewall successfully blocked unauthorized and excessive traffic based on predefined security rules.

IDS Response

Snort generated real-time alerts upon detecting suspicious scanning and attack patterns.

IPS Response

The intrusion prevention system automatically dropped malicious packets to prevent further attacks.

VPN Security

All remote communication was securely encrypted through the VPN tunnel, ensuring data confidentiality.

Final Conclusion (For Report)

This project successfully demonstrates the use of Kali Linux to design and implement a robust network perimeter security system by integrating firewalls, IDS/IPS, and VPN technologies, providing layered protection through prevention, detection, and secure communication.

727723EUCY046

Sakthi Sri Santh M

