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.

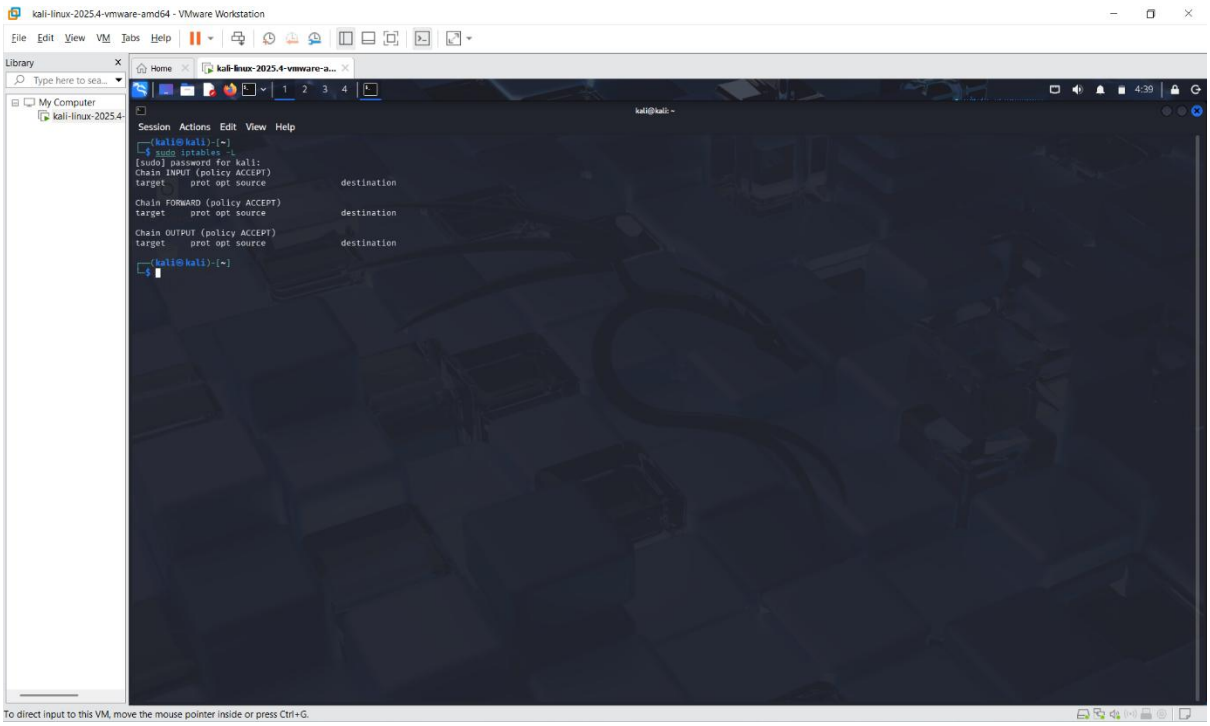
1️⃣ Firewall Configuration in Kali Linux

Tool Used: iptables

Step 1: Check Firewall Status

Command:

```
sudo iptables -L
```

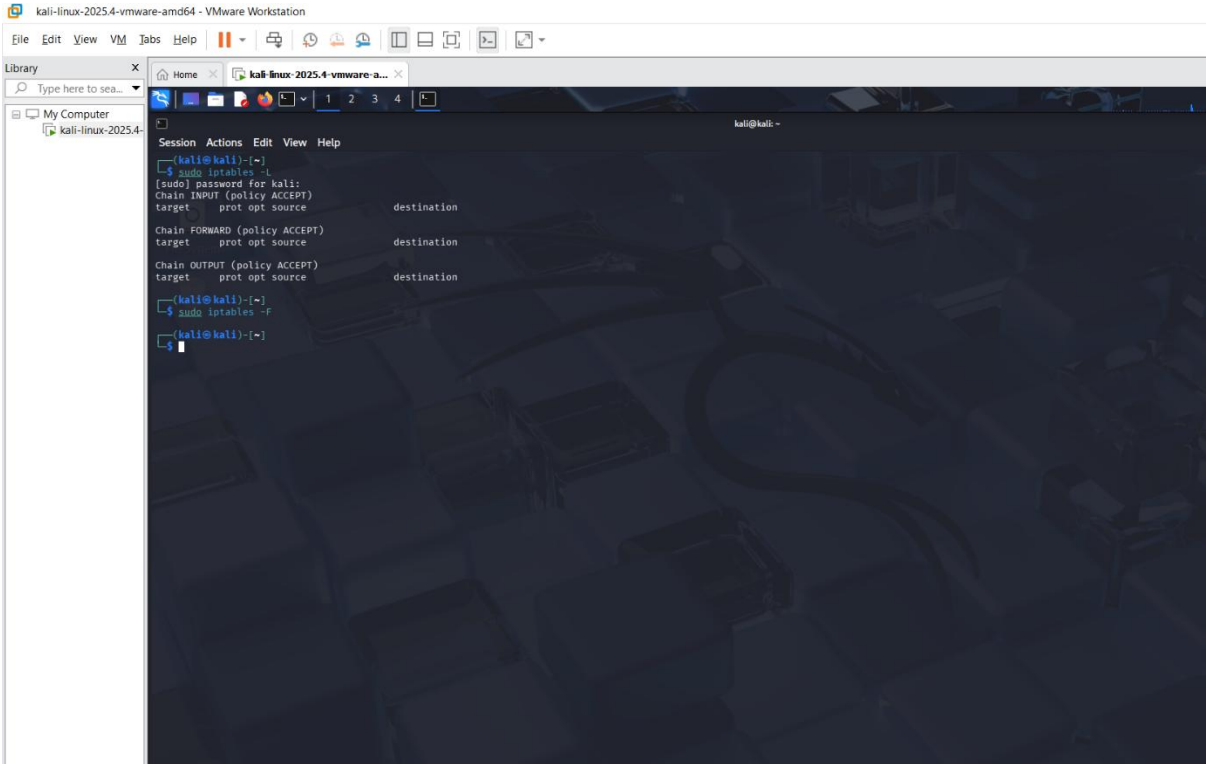


This command is used to display the current firewall rules and policies to understand the existing security state of the system.

Step 2: Flush Existing Rules

Command:

sudo iptables -F



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

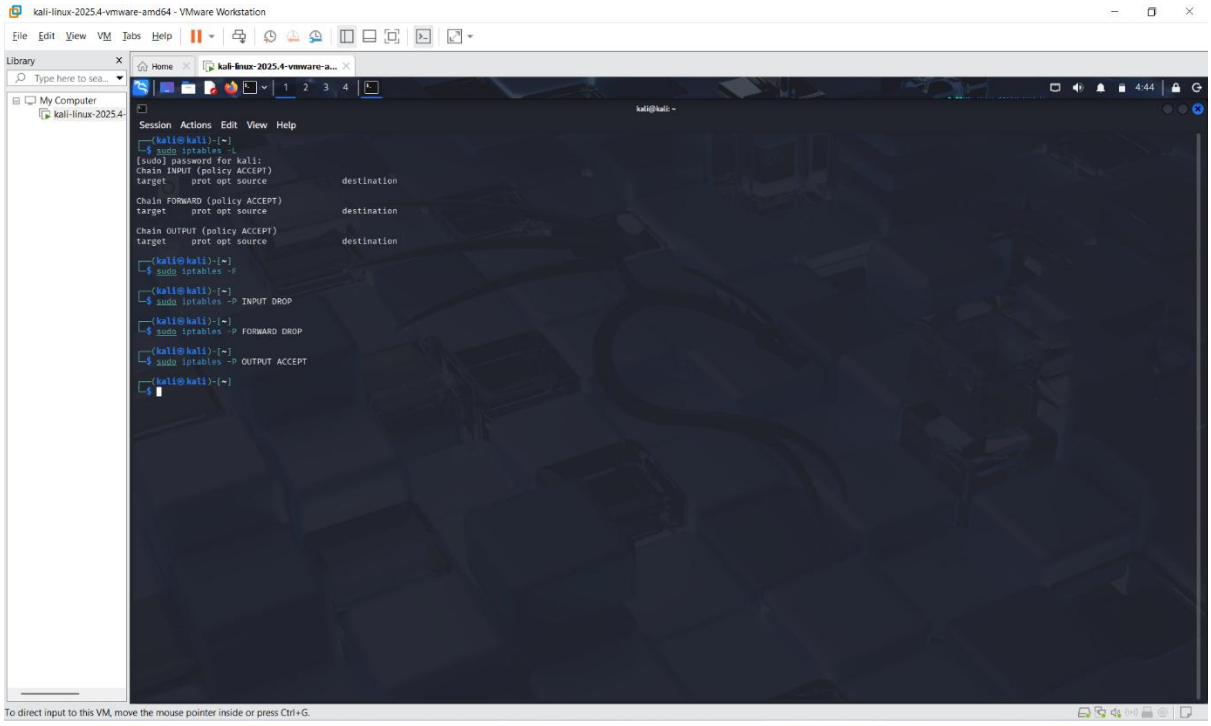
Step 3: Set Default Policies (Secure Baseline)

Commands:

sudo iptables -P INPUT DROP

sudo iptables -P FORWARD DROP

sudo iptables -P OUTPUT ACCEPT



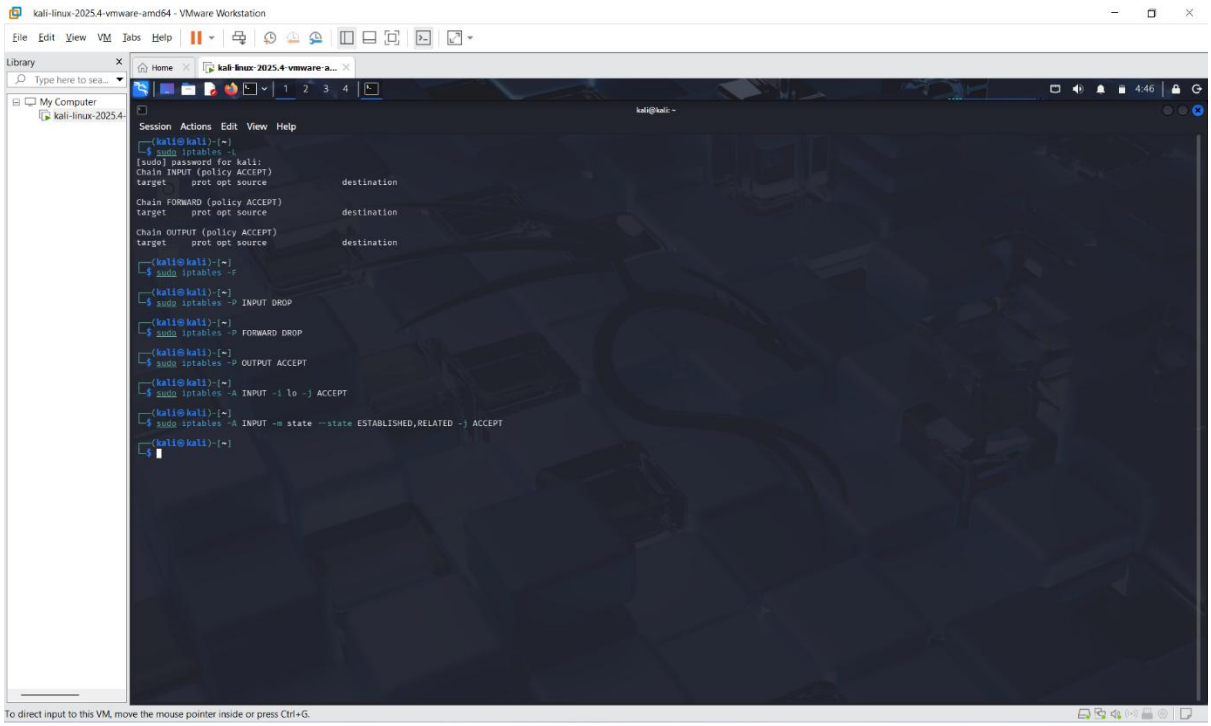
This step applies a default deny policy that blocks all incoming and forwarded traffic while allowing outgoing traffic for secure communication.

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



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

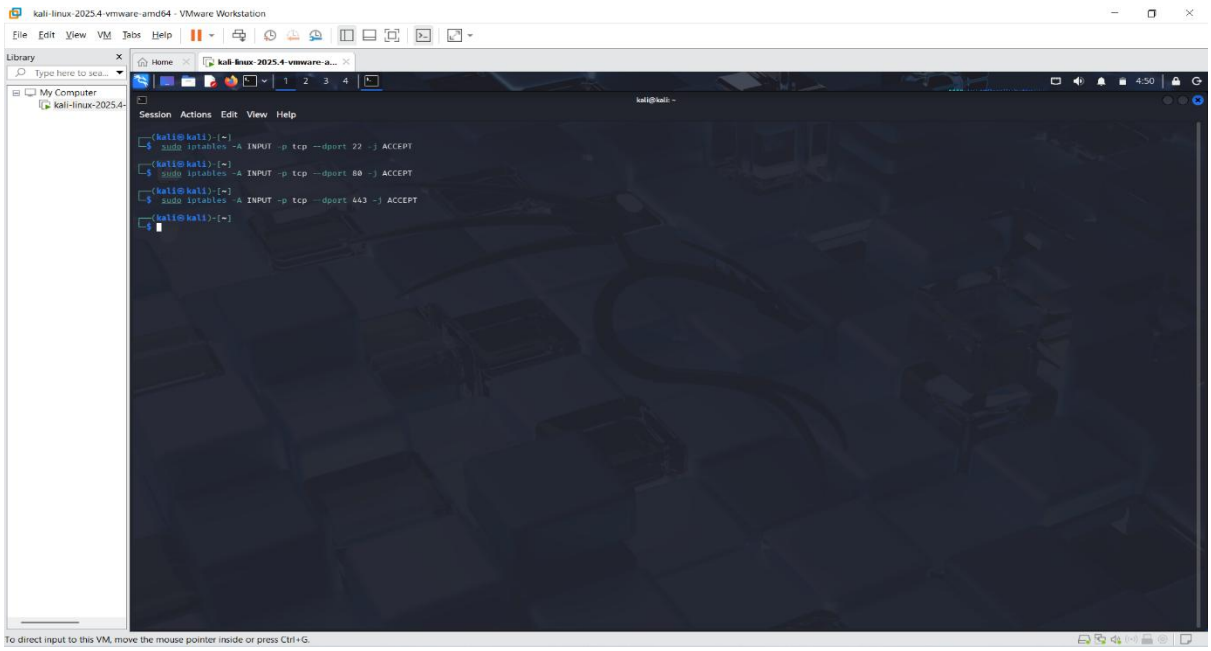
Step 5: Allow Essential Services

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

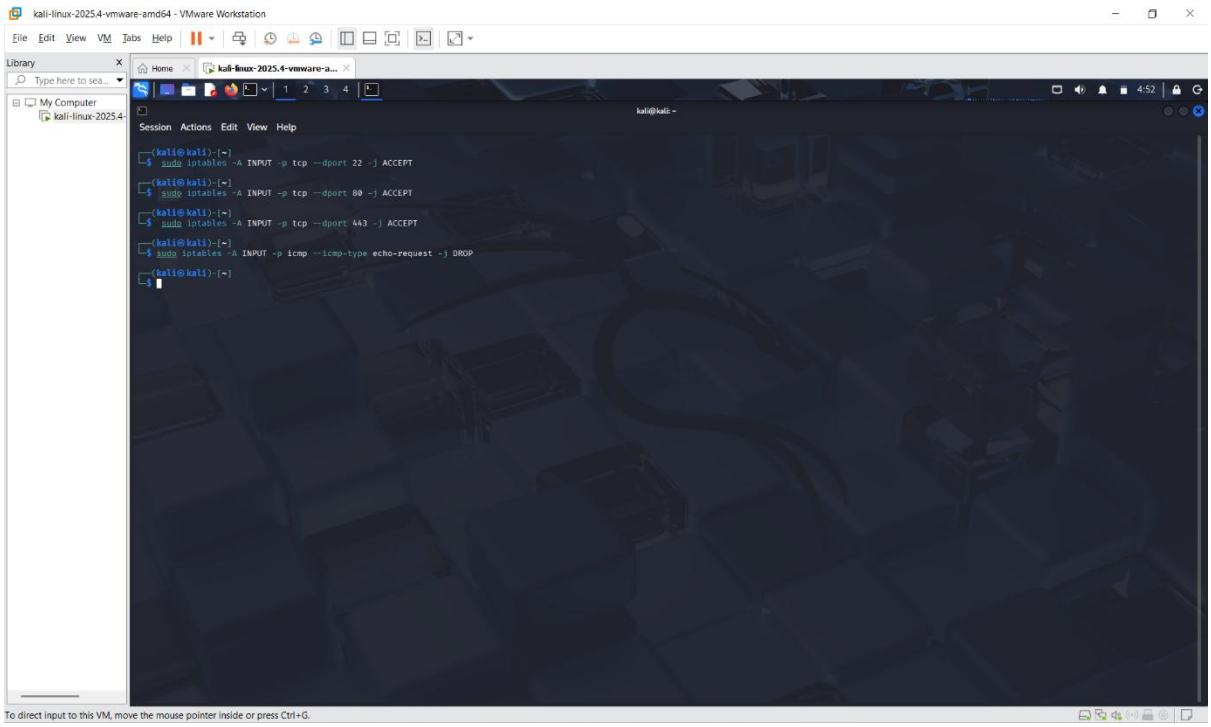


This step allows essential services such as secure shell (SSH) and web services to function while keeping other ports blocked.

Step 6: Block Ping & Scan Attempts

Command:

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



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

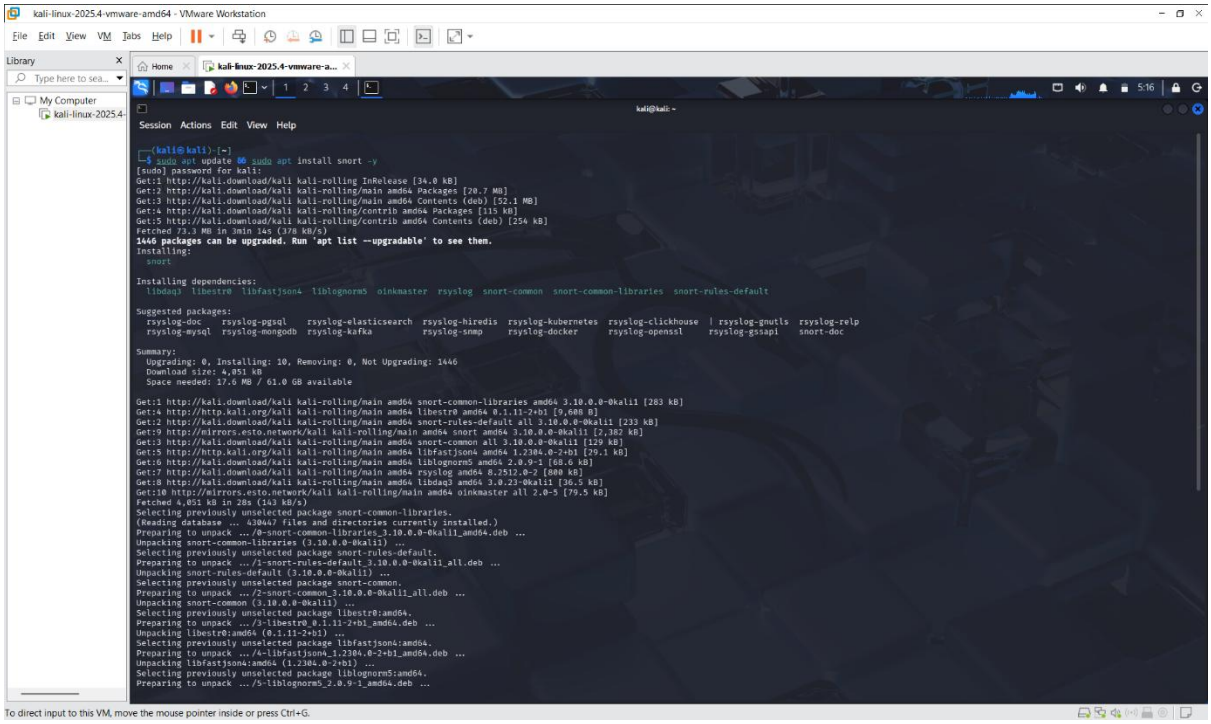
✓ Firewall configured successfully

2 IDS / IPS Setup Using Snort (Kali Linux)

Tool Used: Snort

Step 1: Install Snort

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



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

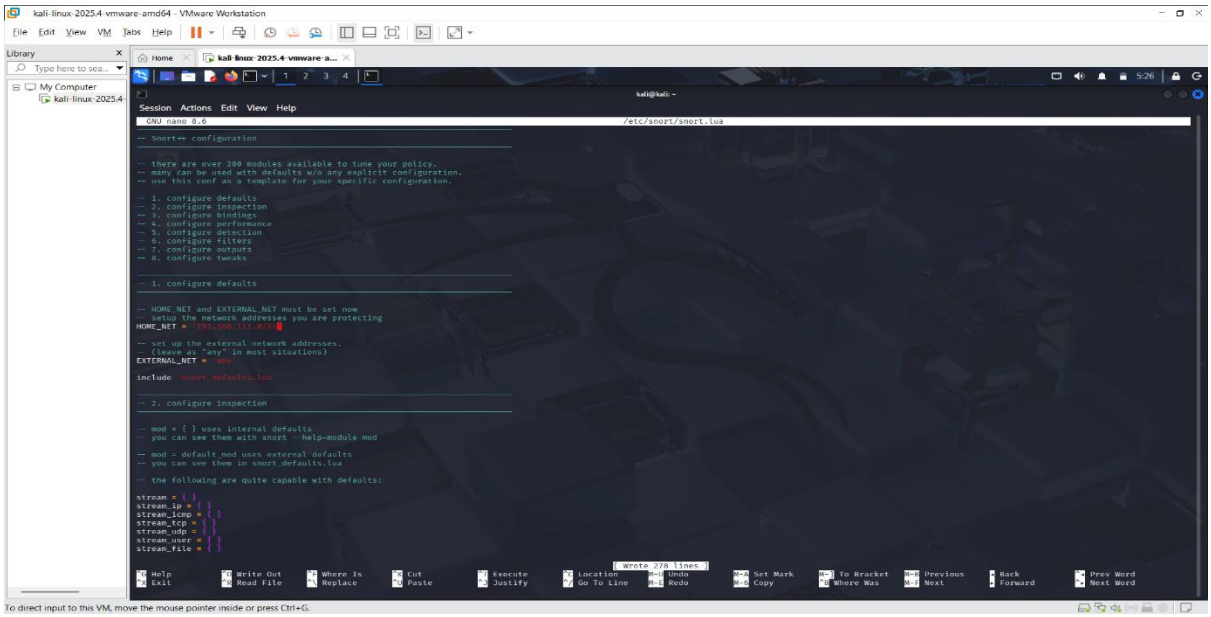
Step 2: Configure Network Interface

Commands:

sudo nano /etc/snort/snort.lua

Change it to:

HOME_NET = '192.168.111.0/24'

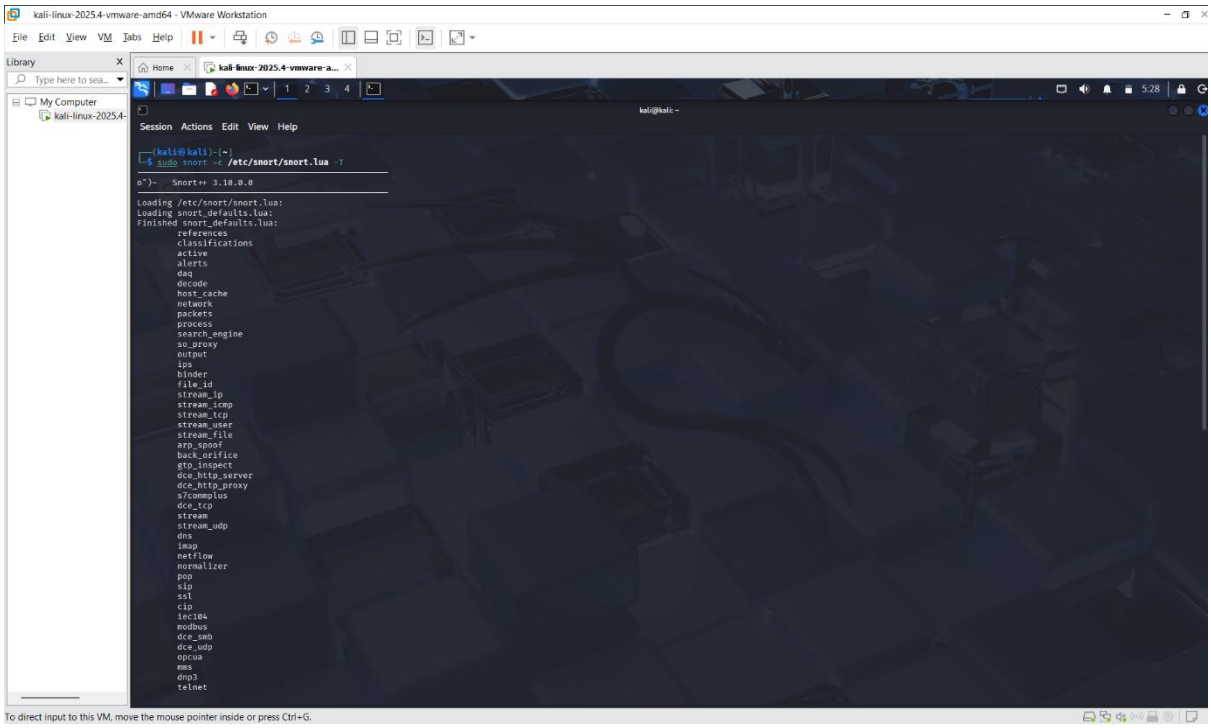


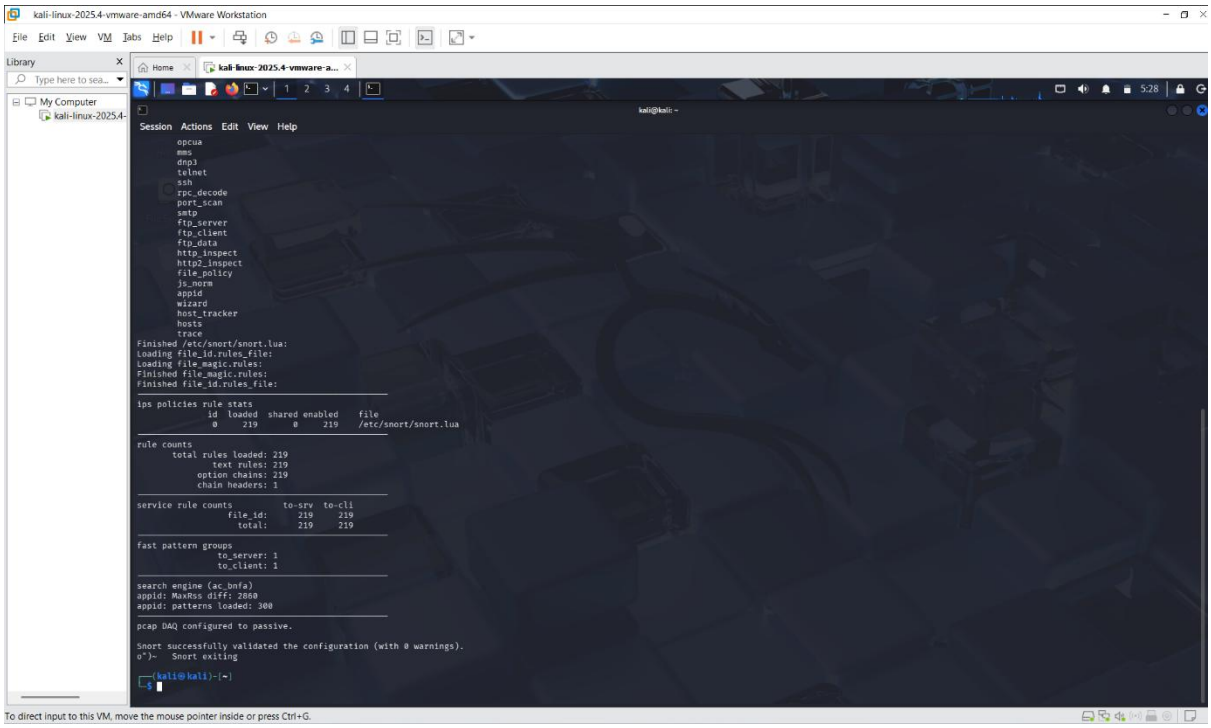
This step defines the local network (HOME_NET) so Snort can monitor and analyze internal traffic accurately.

Step 3: Enable and Verify Snort Rules

Commands:

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



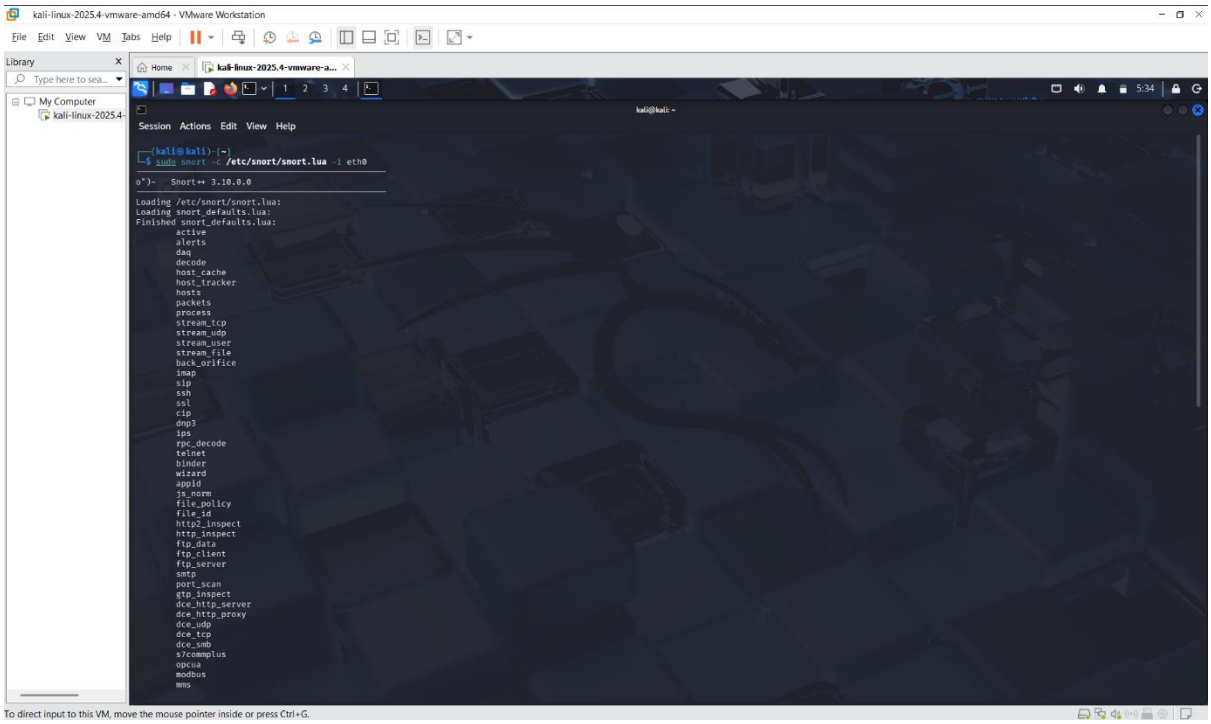


This step checks and validates the Snort configuration and rule files to ensure there are no errors.

Step 4: Run Snort in IDS Mode

Commands:

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

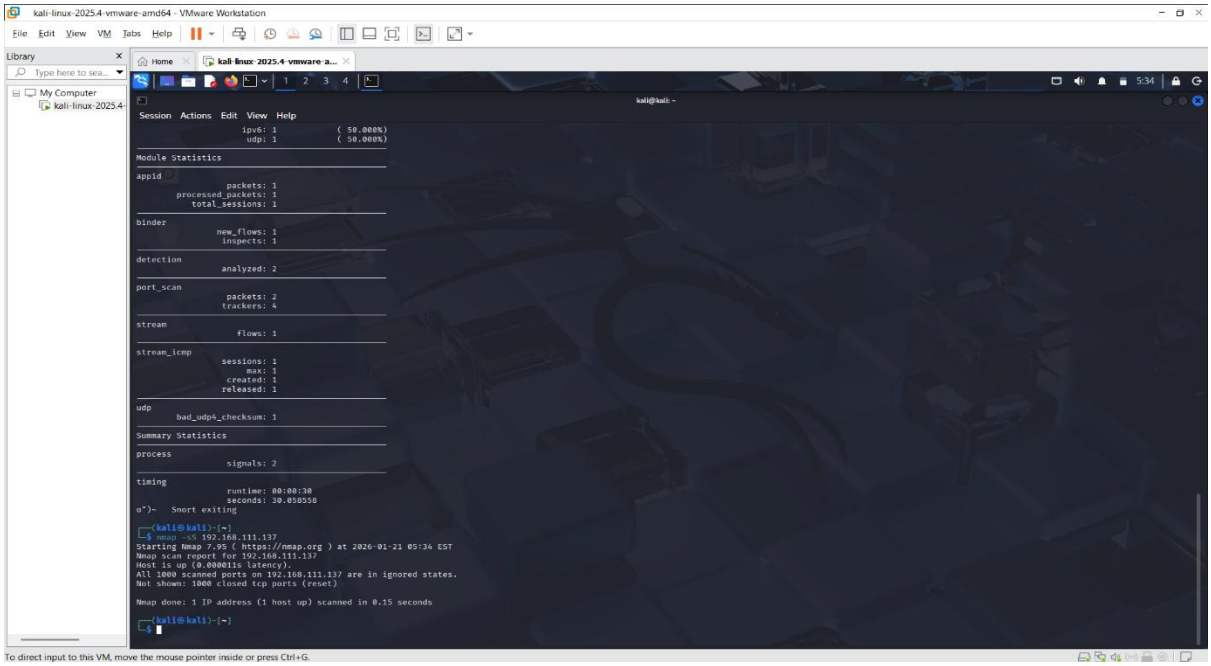


This step runs Snort in Intrusion Detection System (IDS) mode to monitor network traffic and generate alerts.

Step 5: Test IDS Functionality

Commands:

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



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

✓ Attacks are now blocked automatically

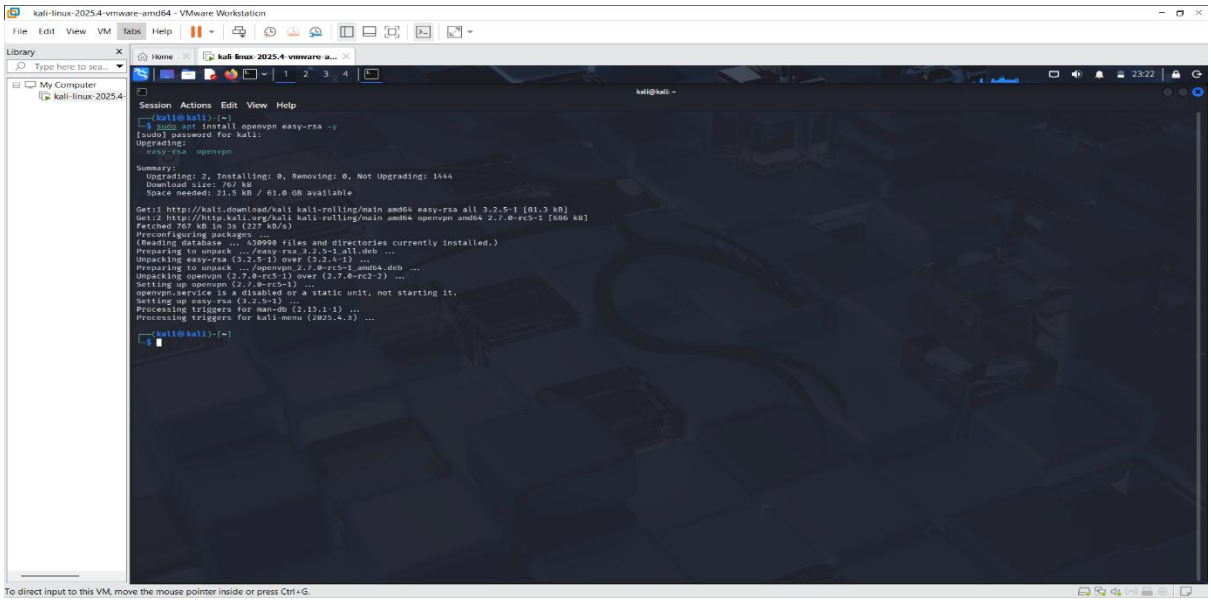
3 VPN Setup Using OpenVPN (Kali Linux)

Tool Used: OpenVPN

Step 1: Install OpenVPN

Commands:

sudo apt install openvpn easy-rsa -y



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

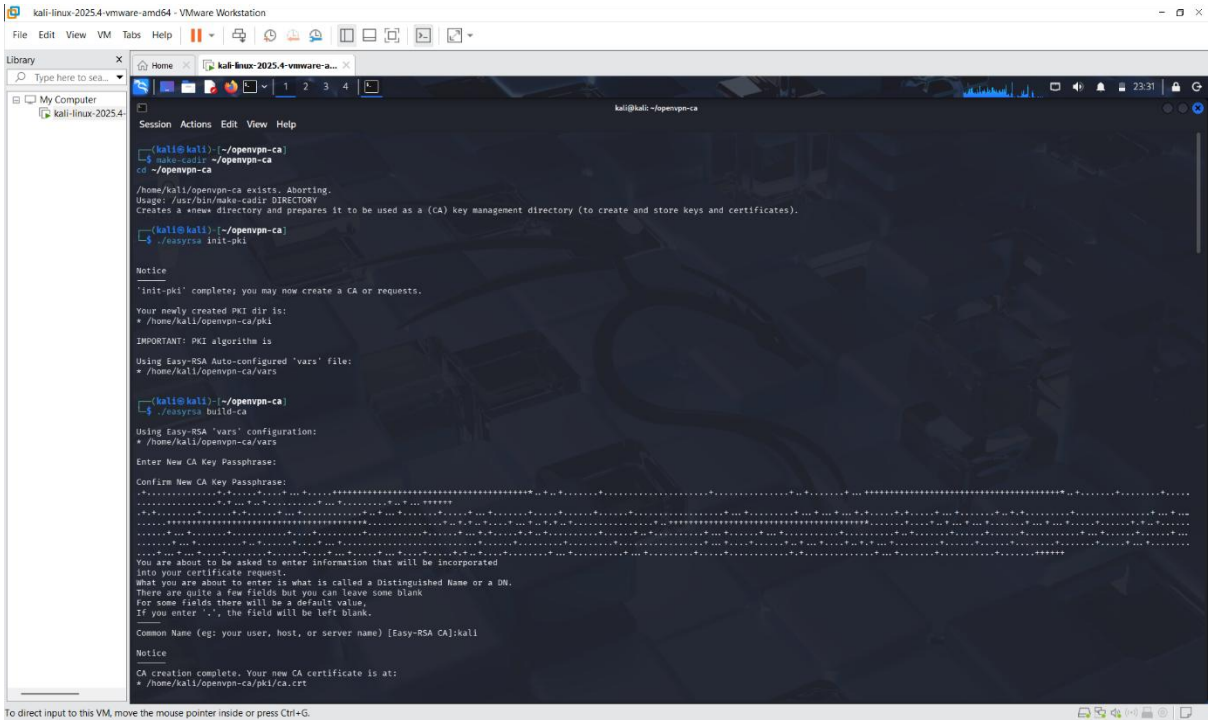
./easyrsa init-pki

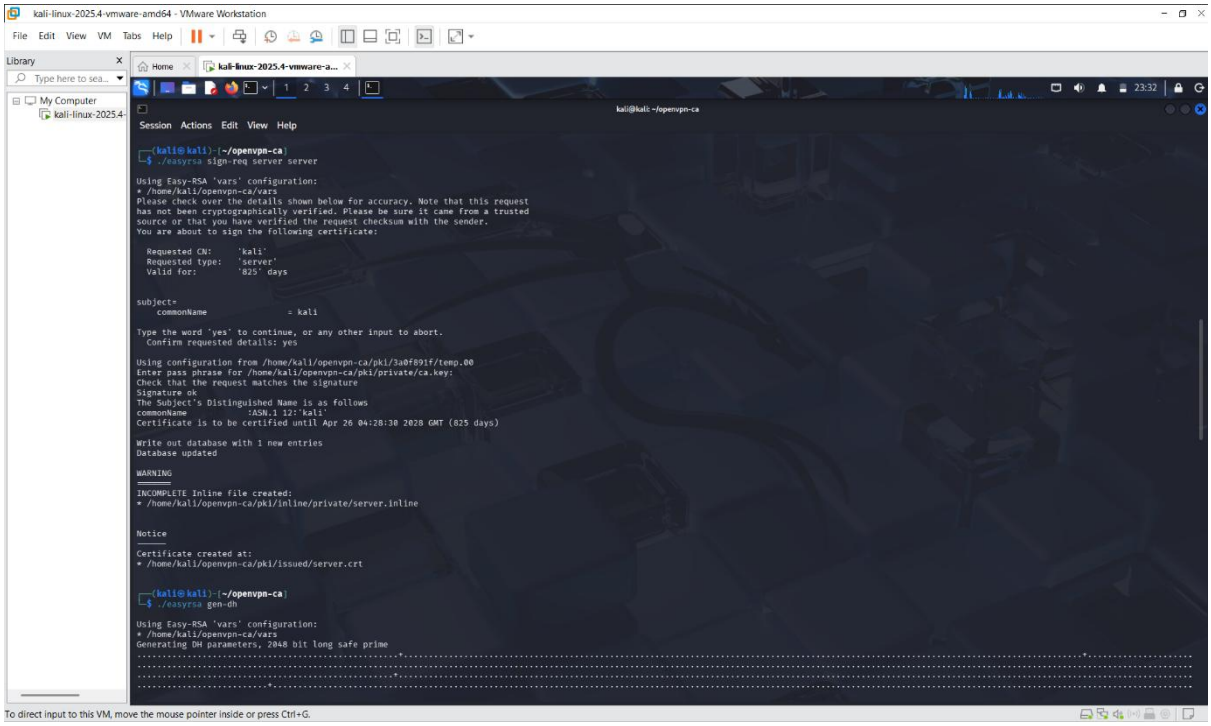
./easyrsa build-ca

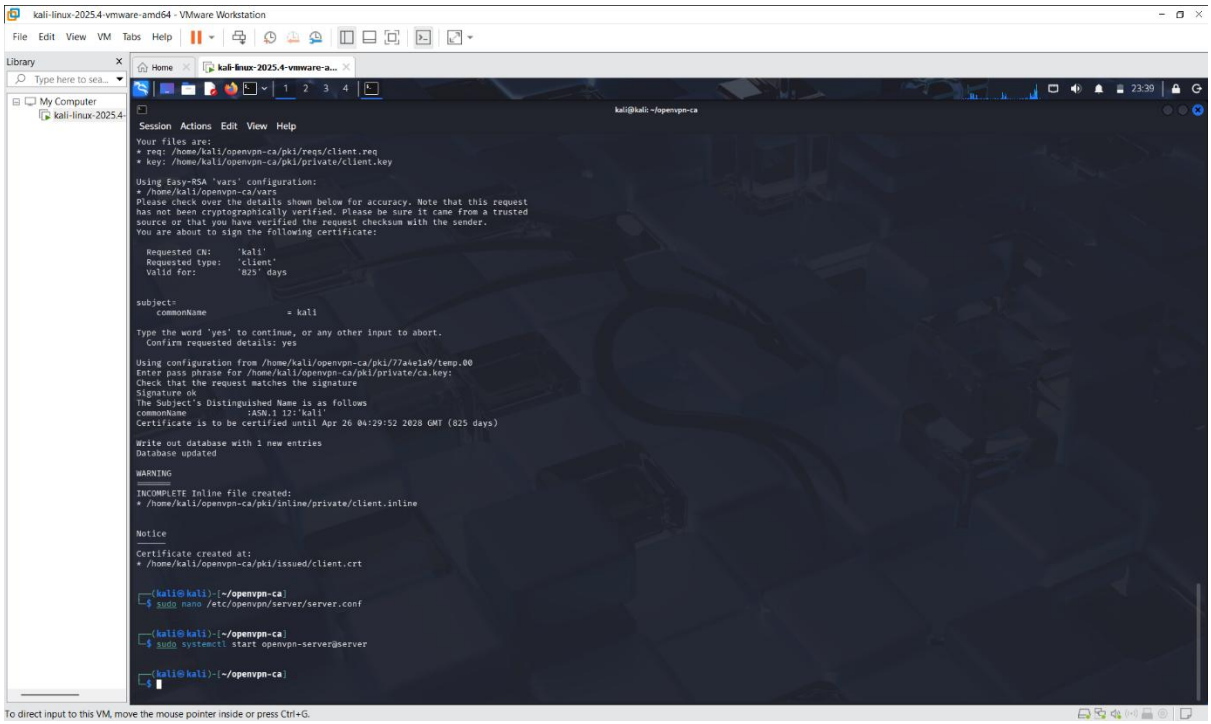
./easyrsa build-server-full server nopass

./easyrsa build-client-full client nopass

./easyrsa gen-dh





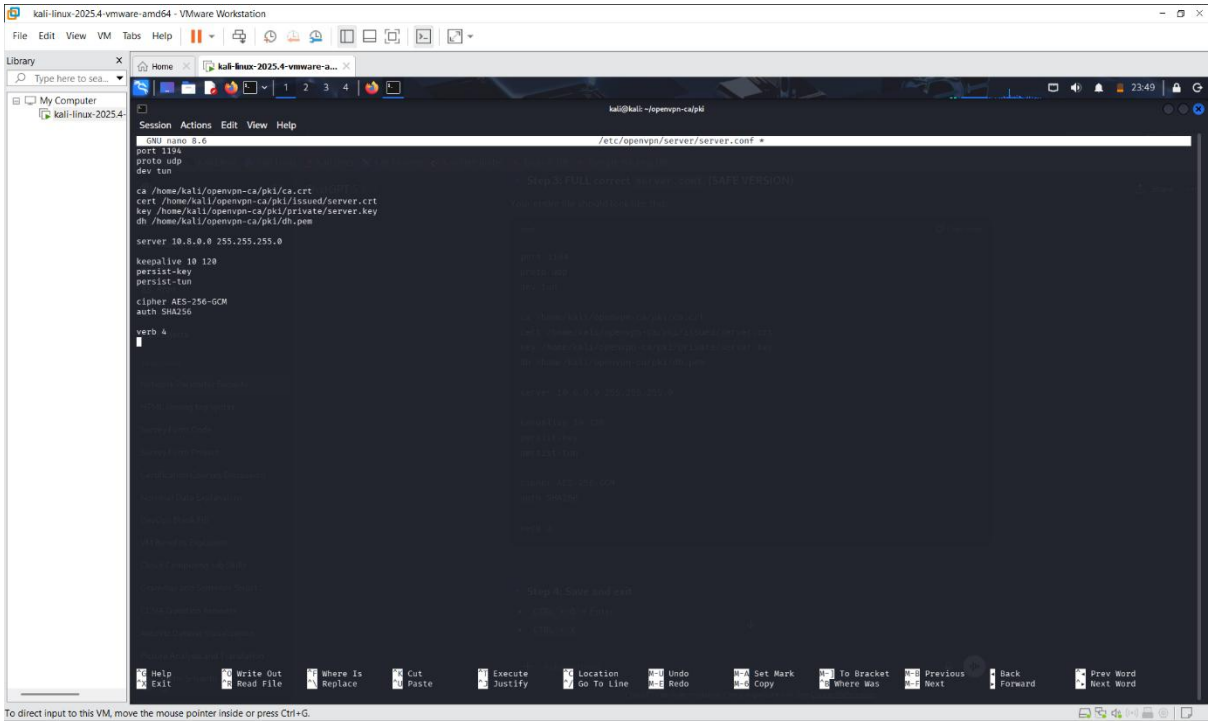


This step generates the Certificate Authority, server certificate, client certificate, and Diffie-Hellman parameters for secure authentication.

Step 3: Configure OpenVPN Server

Commands:

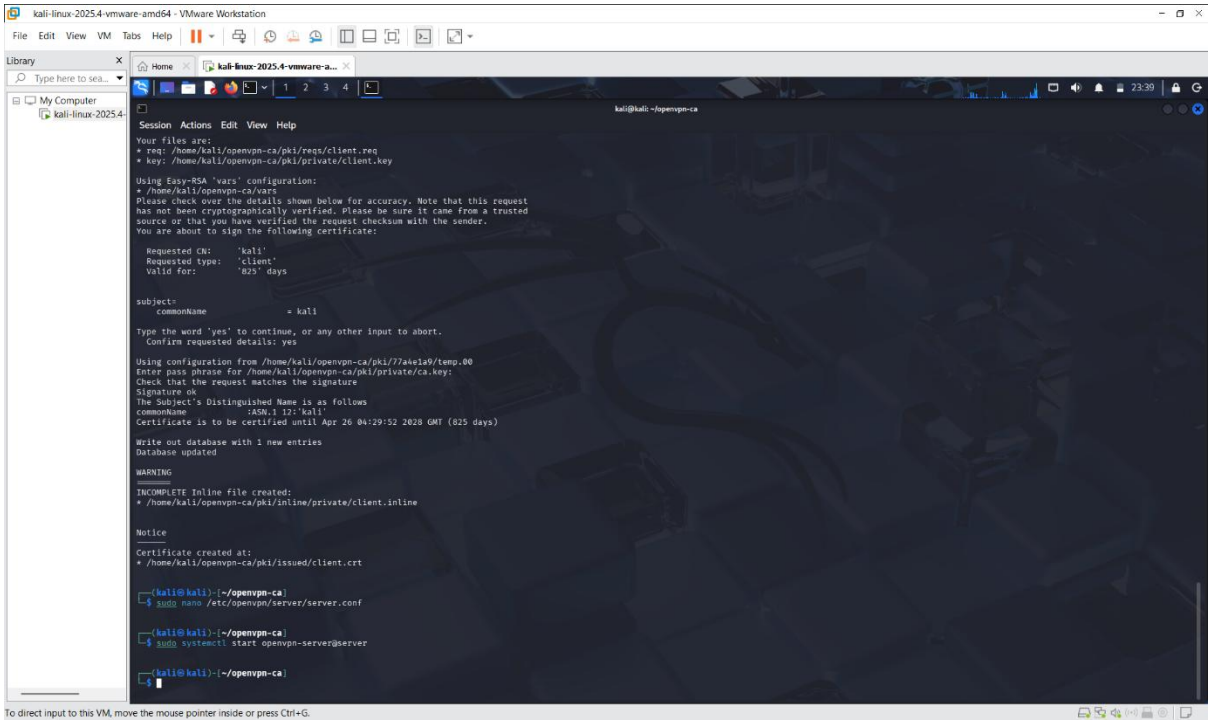
sudo nano /etc/openvpn/server/server.conf



Step 4: Start VPN Server

Commands:

sudo systemctl start openvpn@server

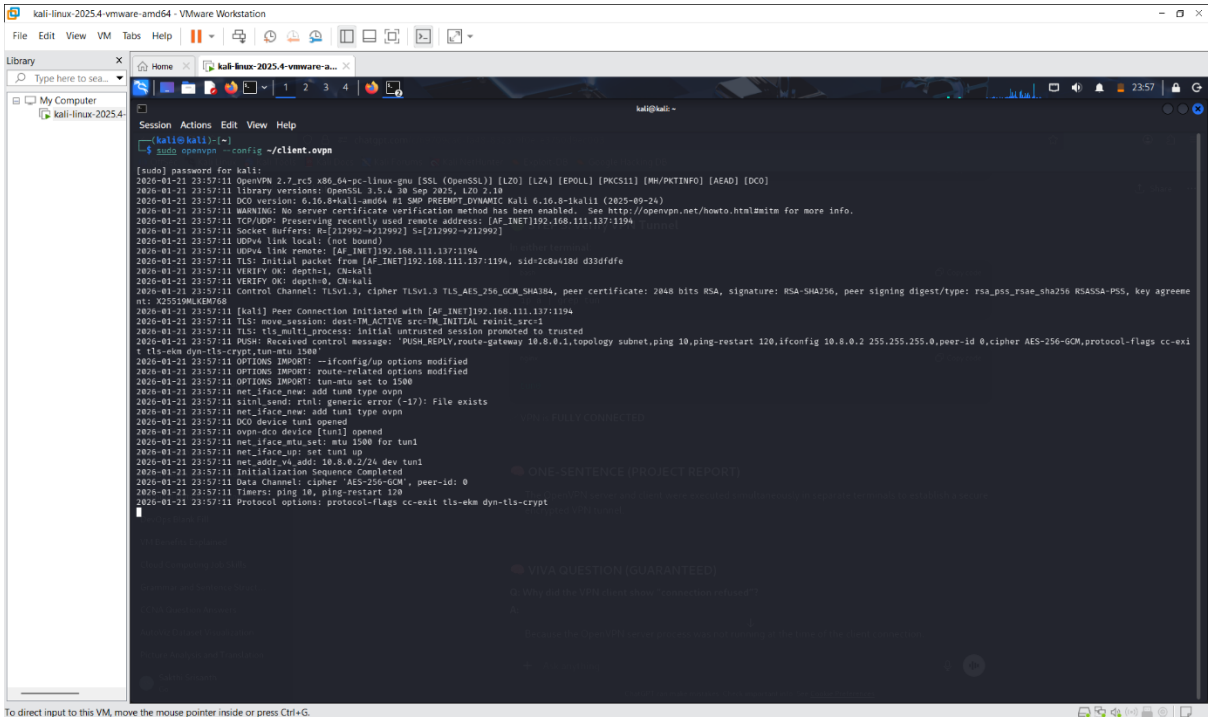


This command starts the OpenVPN server service, allowing the system to accept secure VPN connections.

Step 5: Connect VPN Client

Commands:

sudo openvpn --config client.ovpn



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.

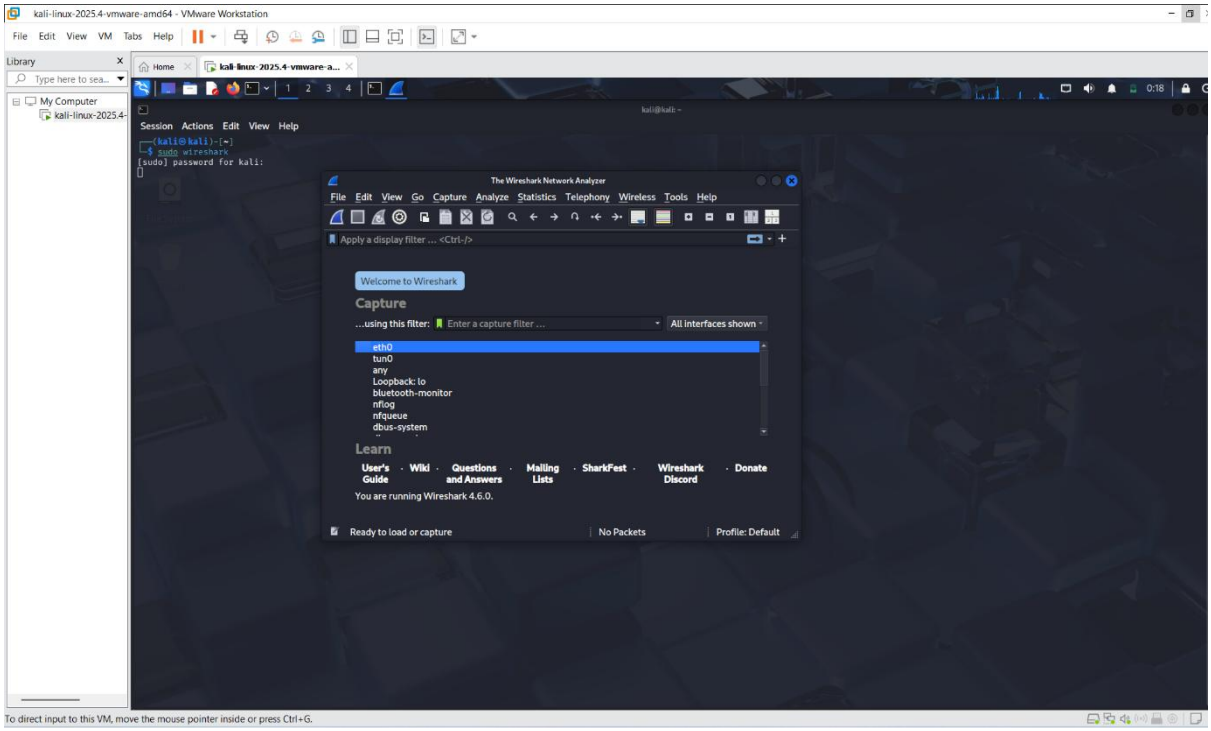
4️⃣ Traffic Monitoring Using Wireshark & Tshark

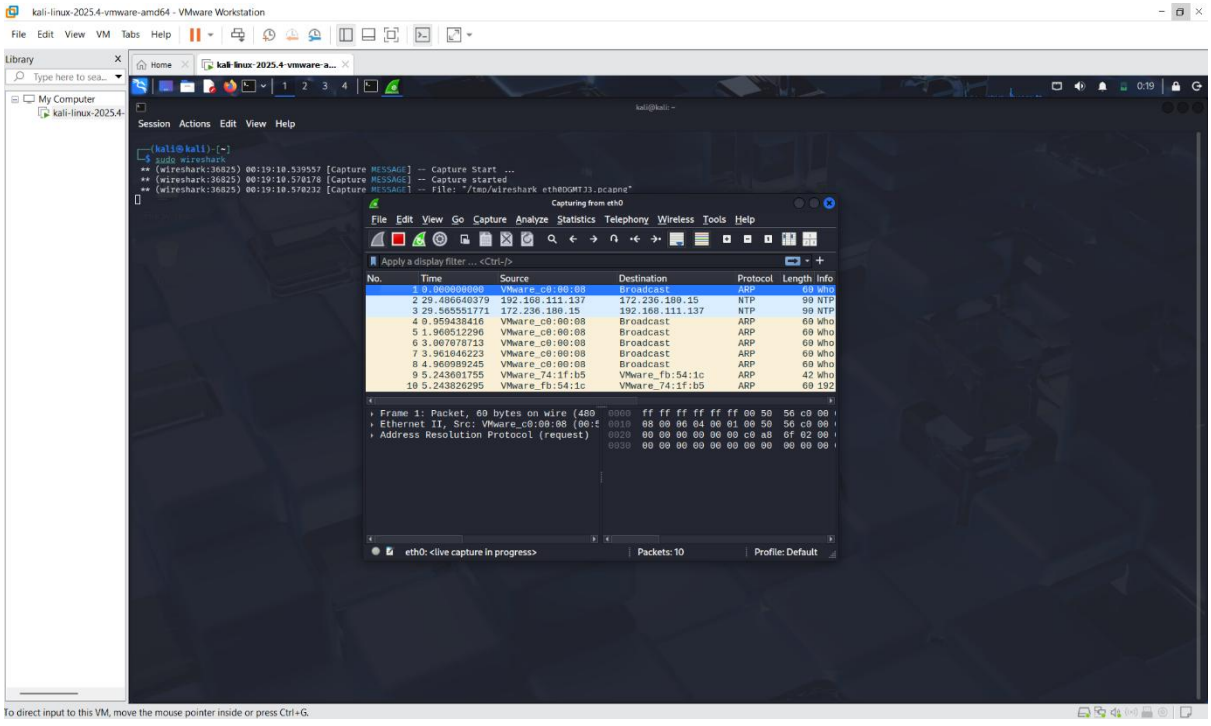
Tools Used: Wireshark, Tshark

Step 1: Start Traffic Capture (GUI Mode)

Commands:

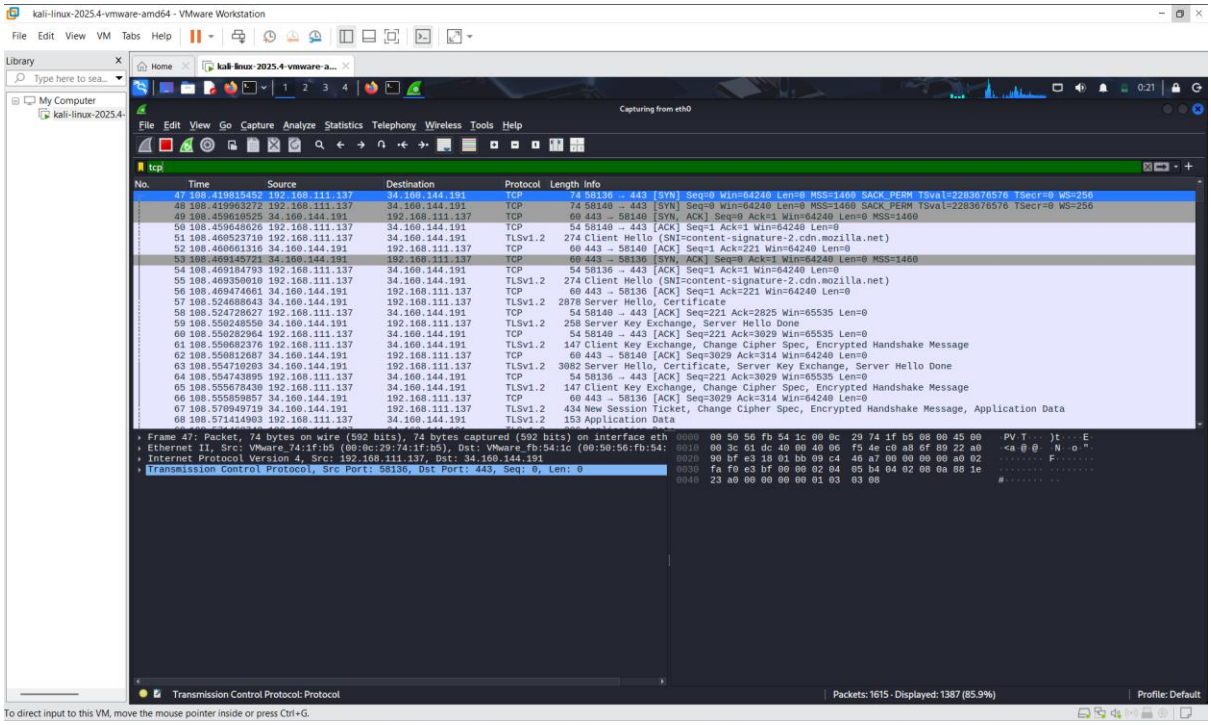
sudo wireshark





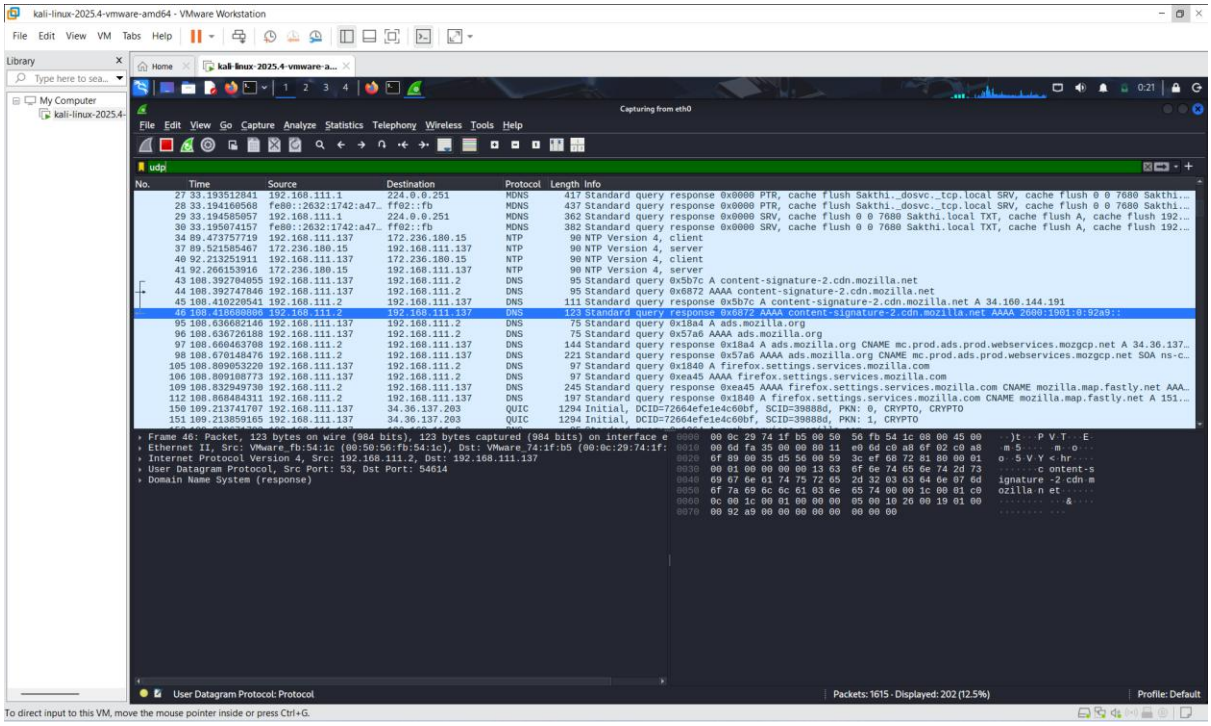
This command launches Wireshark to capture and visually analyze network packets in real time.

Step 2: Apply TCP Filter



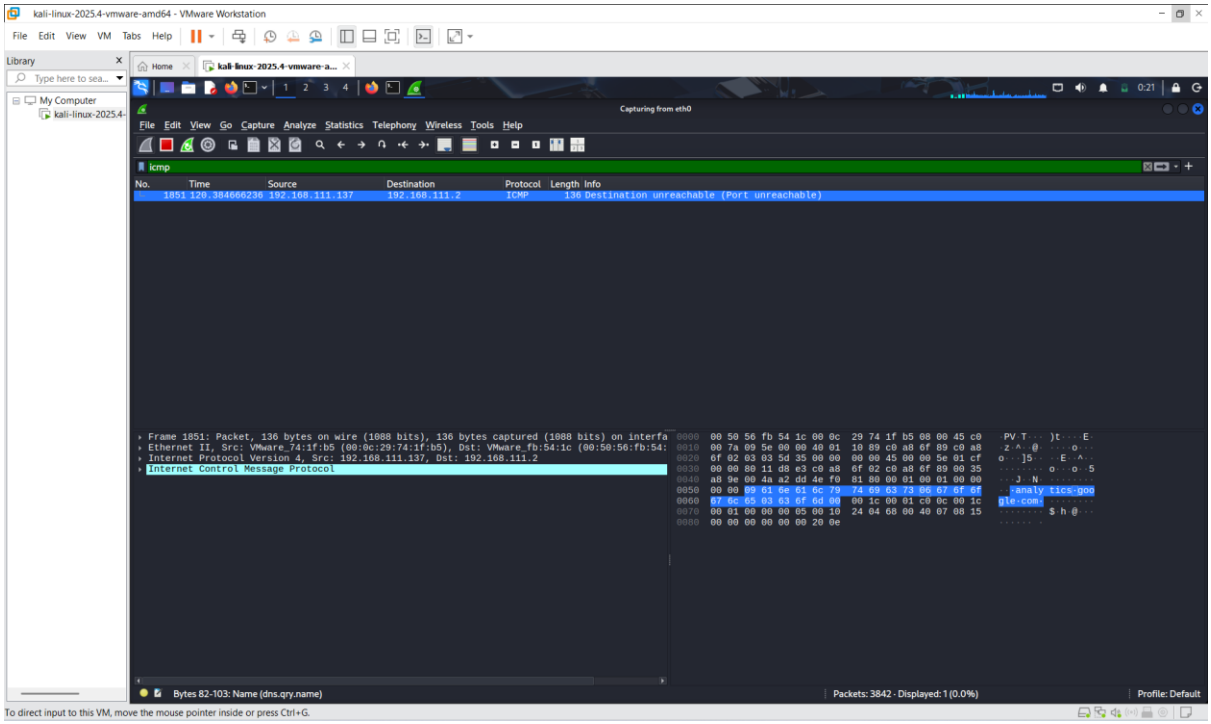
This filter displays only TCP packets to analyze connection-oriented traffic such as HTTP and HTTPS.

Step 3: Apply UDP Filter



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

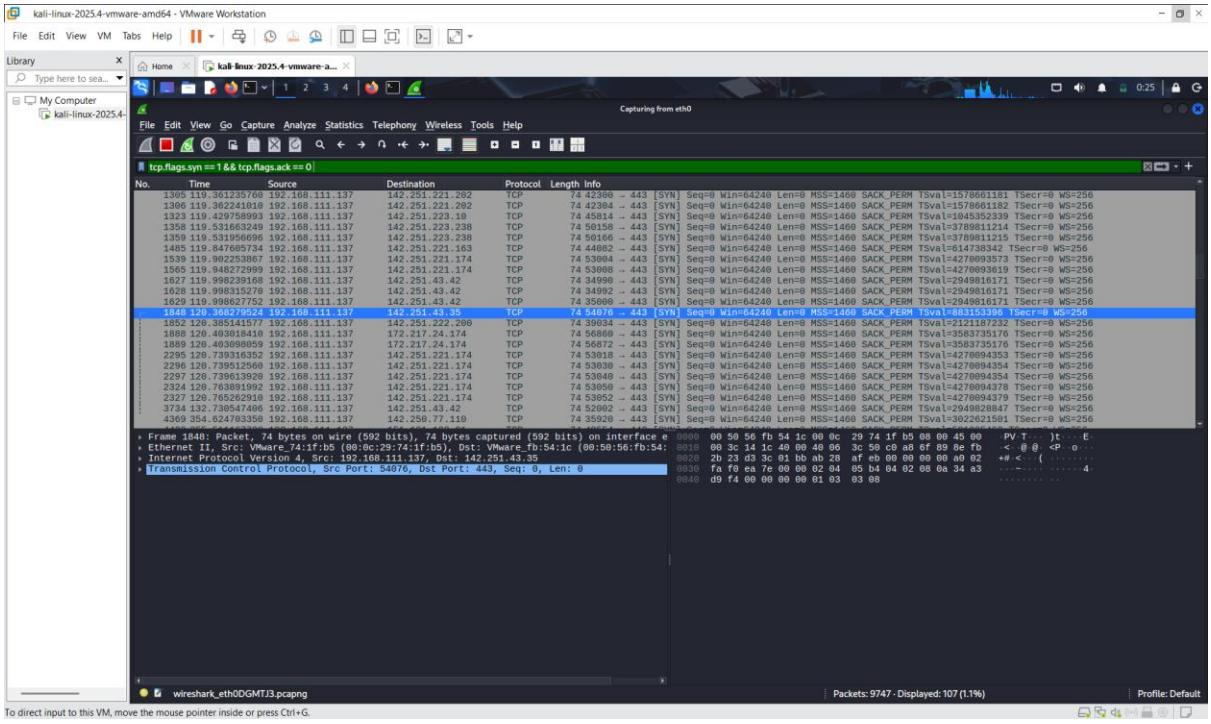
Step 4: Apply ICMP Filter



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)

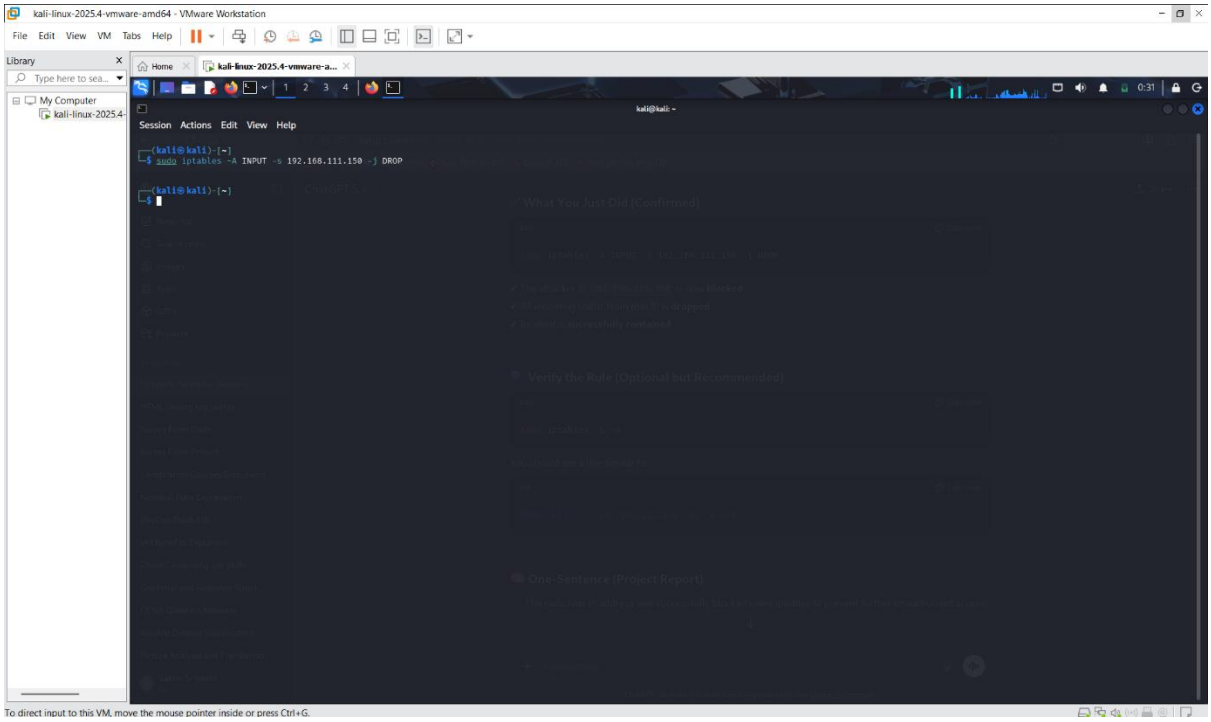
Detection

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.

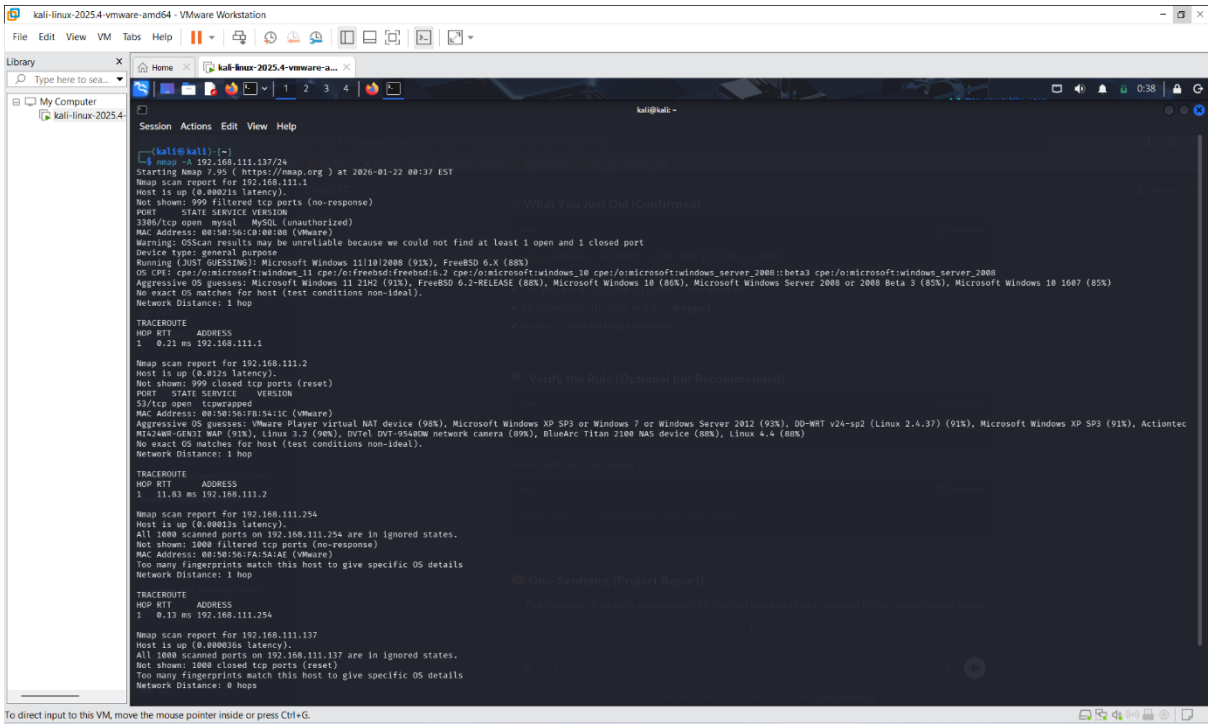
The incident response plan ensures timely detection, containment, eradication, recovery, and documentation of network security incidents.

6 Testing & Demonstration (Mandatory)

Simulated Attacks

Commands:

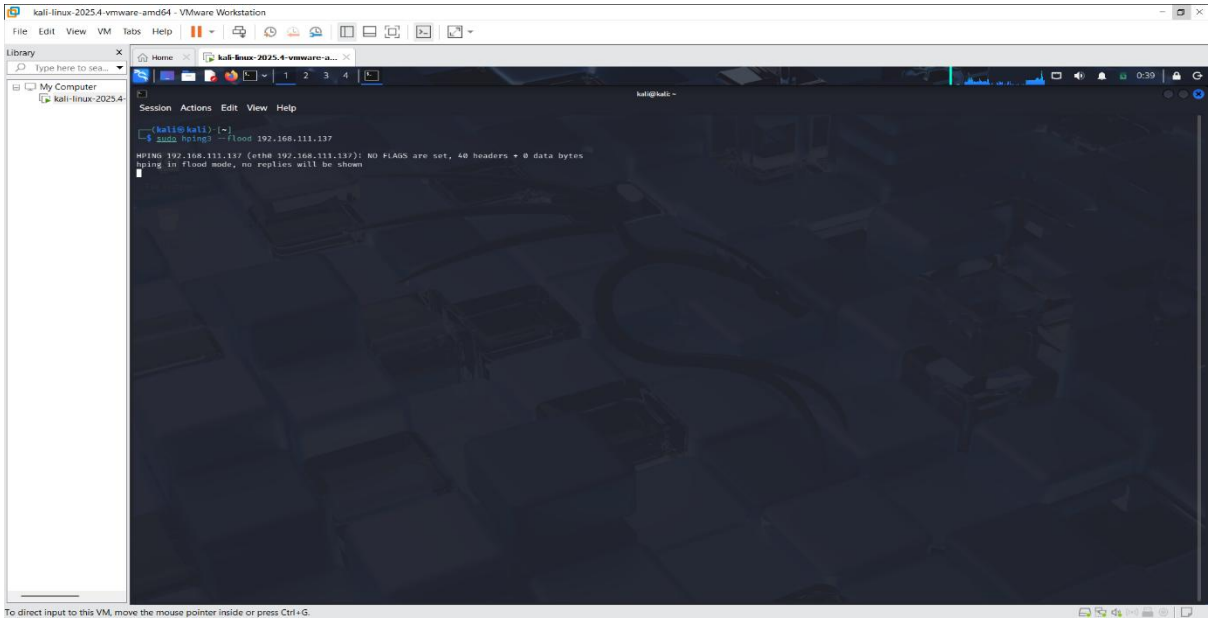
sudo nmap -A 192.168.111.137/24



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



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