Project: Building a Secure Network Environment

By

Toniloba Ogundare - 159102

Date

November 2025

Table of Content

List of Figures

1.0 Introduction

This project demonstrates the design, deployment and evaluation of a secure network environment using a combination of defensive and offensive tools. The objective is to model a realistic organizational security setup where a protected server is exposed to both internal and external probing. The system integrates a packet-filtering firewall, intrusion-detection capabilities through packet capture, and a functional VPN service.

Through this work, we gained hands-on experience with network security concepts including traffic monitoring, service hardening, attack simulation and external connectivity testing. The environment provides a controlled platform to analyze how defensive configurations respond to legitimate traffic, malicious reconnaissance and external client interactions.

1.1 Network Architecture

The network consists of three systems: two virtual machines connected via a bridged adapter and a physical Windows host used for external validation. This setup closely models an enterprise-style network where an internal attacker, a protected server, and an external client all interact

1. Ubuntu Server (Defender)

This machine hosts the primary defensive infrastructure and acts as the target for all tests.

i. Implements a strict iptables firewall controlling inbound traffic:
Allowed:

- SSH (TCP 22)
- HTTP (TCP 80)
- OpenVPN (UDP 1194)
- ICMP echo requests/replies

Blocked:

- All other inbound TCP ports

ii. Runs an OpenVPN server on UDP 1194 to provide secure remote-access capability.

iii. Uses tcpdump as a lightweight intrusion-detection mechanism to analyze incoming ICMP packets, TCP SYN probes, Nmap scans and other reconnaissance traffic.

iv. Acts as the primary server being scanned, attacked, accessed, and externally validated.

2. Kali Linux (Attacker)

This system simulates an internal threat actor performing reconnaissance.

i. Conducts Nmap scans, including TCP SYN scans and service/port discovery.
ii. Performs ICMP ping tests and network probing to map reachable hosts.
iii. Communicates with the Ubuntu server over the same bridged Layer-2 network.
This VM represents an adversary assessing the defender’s exposed attack surface.

3. Windows Host (External Tester: PowerShell)

This is a physical machine operating outside the virtual environment, used to confirm how the server behaves from an external perspective.

i. Used to test firewall behavior using NCAT (nc) connections to:

- Allowed ports (22, 80, 1194)
- Blocked ports (e.g., 443 or any other closed port)

ii. Performs PowerShell-based VPN connection tests to verify whether OpenVPN is reachable and responsive over UDP 1194.

iii. Provides an external validation layer ensuring that Ubuntu’s firewall and services function correctly when accessed from a real machine on the bridged network.

1.2 Network Characteristics

- All systems share the same bridged network segment and can communicate directly.
- Traffic routing is handled automatically at the layer-2 level by the virtualization environment.
- The architecture simulates a realistic attacker-defender scenario with an external client verifying service behavior under controlled conditions.

2.0 Network Diagram

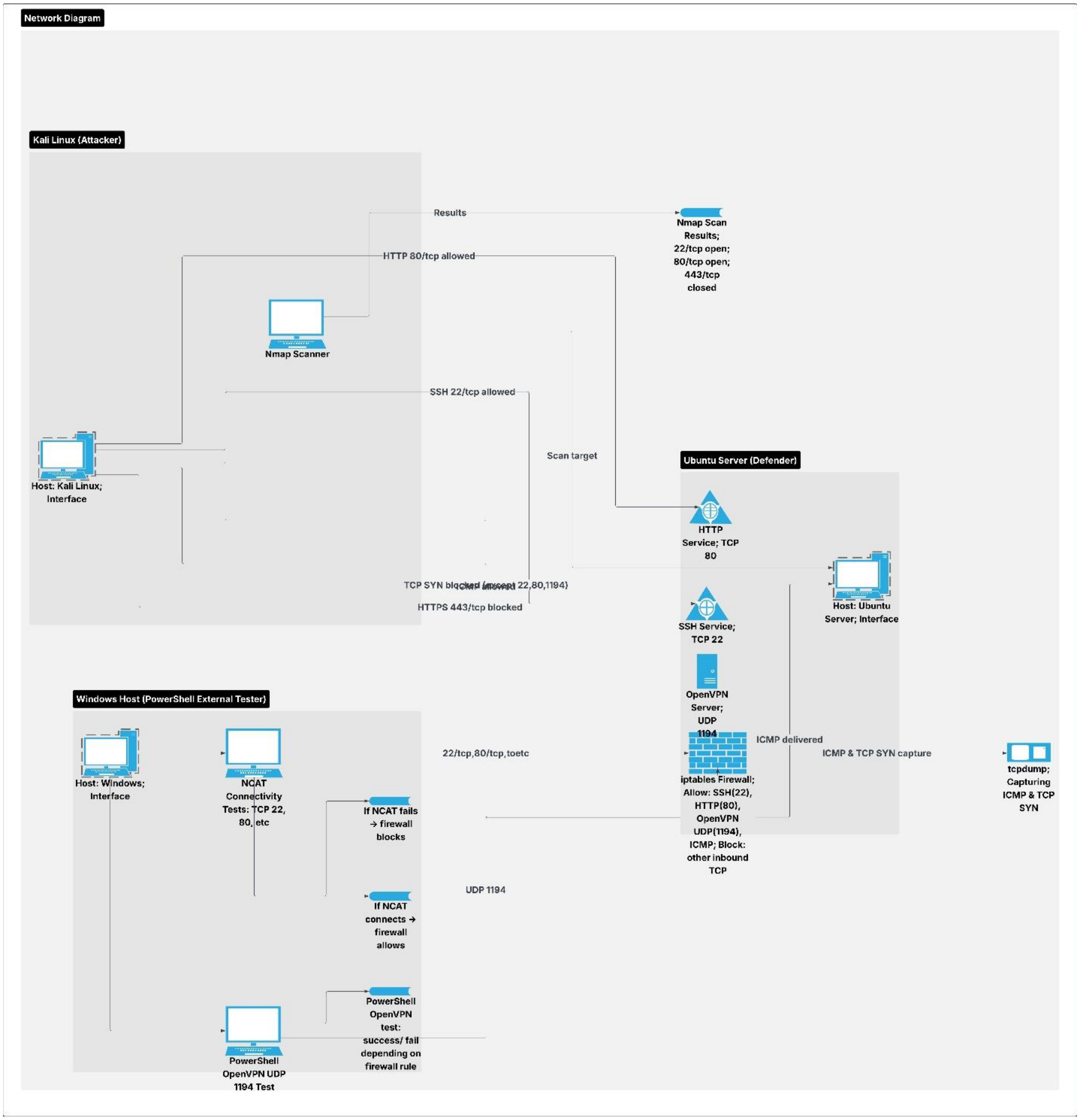
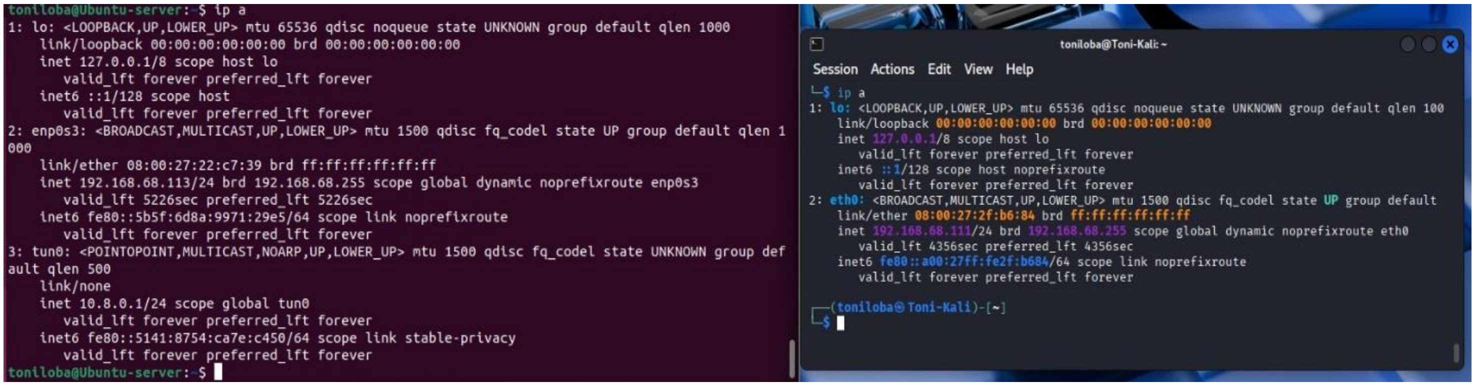

Network Diagram
Kali Linux (Attacker)
Results
HTTP 80/tcp allowed
Nmap Scan Results; 22/tcp open; 80/tcp open; 443/tcp closed
Nmap Scanner
SSH 22/tcp allowed
Scan target
Host: Kali Linux; Interface
Ubuntu Server (Defender)
HTTP Service; TCP 80
TCP SYN blocked (except 22,80,1194)
ICMP allowed
HTTPS 443/tcp blocked
Host: Ubuntu Server; Interface
SSH Service; TCP 22
OpenVPN Server; UDP 1194
ICMP delivered
ICMP & TCP SYN capture
tcpdump; Capturing ICMP & TCP SYN
iptables Firewall; Allow: SSH(22), HTTP(80), OpenVPN UDP(1194), ICMP; Block: other inbound TCP
Windows Host (PowerShell External Tester)
Host: Windows; Interface
NCAT Connectivity Tests: TCP 22, 80, etc
22/tcp,80/tcp,toetc
If NCAT fails → firewall blocks
UDP 1194
If NCAT connects → firewall allows
PowerShell OpenVPN test: success/ fail depending on firewall rule
PowerShell OpenVPN UDP 1194 Test

Figure 1: Network Diagram

Explanation: This diagram illustrates the full security environment used in the project, showing how the Ubuntu Server (Defender), Kali Linux (Attacker), and the Windows Host (External Tester) communicate. The Ubuntu server hosts the firewall, IDS packet capture (tcpdump), OpenVPN service, SSH and HTTP services. The Kali Linux system performs ICMP pings, TCP SYN scans, and Nmap reconnaissance to identify open and filtered ports. The Windows host is used to validate firewall behavior externally through NCAT connectivity tests and OpenVPN UDP 1194 probing.

The arrows show the direction of traffic and the results returned from each security test, demonstrating which services are allowed (22/tcp, 80/tcp, 1194/udp) and which are blocked (e.g., 443/tcp). This visual provides a clear overview of the interactions and security controls implemented throughout the project.

2.1 Ubuntu and Kali IP addresses

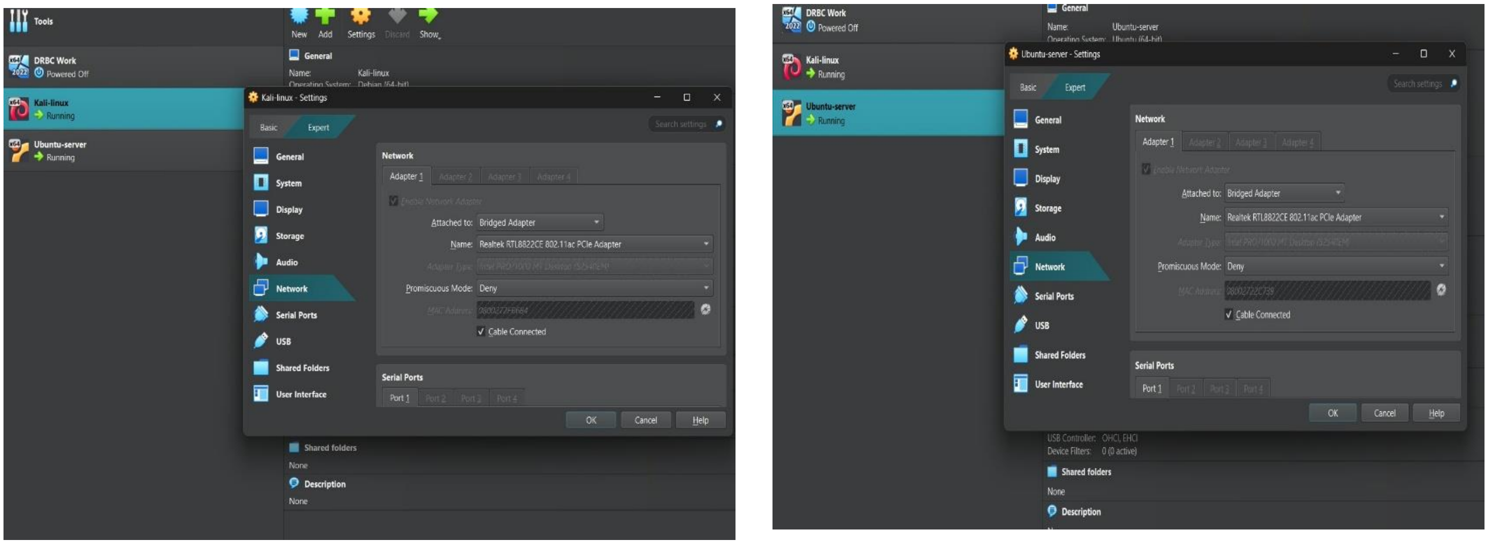


```
toniloba@Ubuntu-server:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1
000
    link/ether 08:00:27:22:c7:39 brd ff:ff:ff:ff:ff:ff
    inet 192.168.68.113/24 brd 192.168.68.255 scope global dynamic noprefixroute enp0s3
       valid_lft 5226sec preferred_lft 5226sec
    inet6 fe80::5b5f:6d8a:9971:29e5/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group def
ault qlen 500
    link/none
    inet 10.8.0.1/24 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::5141:8754:ca7e:c450/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
toniloba@Ubuntu-server:~$
```

```
toniloba@Toni-Kali: ~
Session  Actions  Edit  View  Help
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 100
    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
    link/ether 08:00:27:2f:b6:84 brd ff:ff:ff:ff:ff:ff
    inet 192.168.68.111/24 brd 192.168.68.255 scope global dynamic noprefixroute eth0
       valid_lft 4356sec preferred_lft 4356sec
    inet6 fe80::a00:27ff:fe2f:b684/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(toniloba㉿Toni-Kali)-[~]
└─$
```

Figure 2: From ip a on both systems



Kali-linux - Settings

Network — Adapter 1

☑ Enable Network Adapter

Attached to: Bridged Adapter

Name: Realtek RTL8822CE 802.11ac PCIe Adapter

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Deny

MAC Address: 0800272FB684

☑ Cable Connected

Ubuntu-server - Settings

Network — Adapter 1

☑ Enable Network Adapter

Attached to: Bridged Adapter

Name: Realtek RTL8822CE 802.11ac PCIe Adapter

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Deny

MAC Address: 08002722C739

☑ Cable Connected

USB Controller: OHCI, EHCI
Device Filters: 0 (0 active)

Figure 3: Ubuntu and Kali network configuration using bridged mode

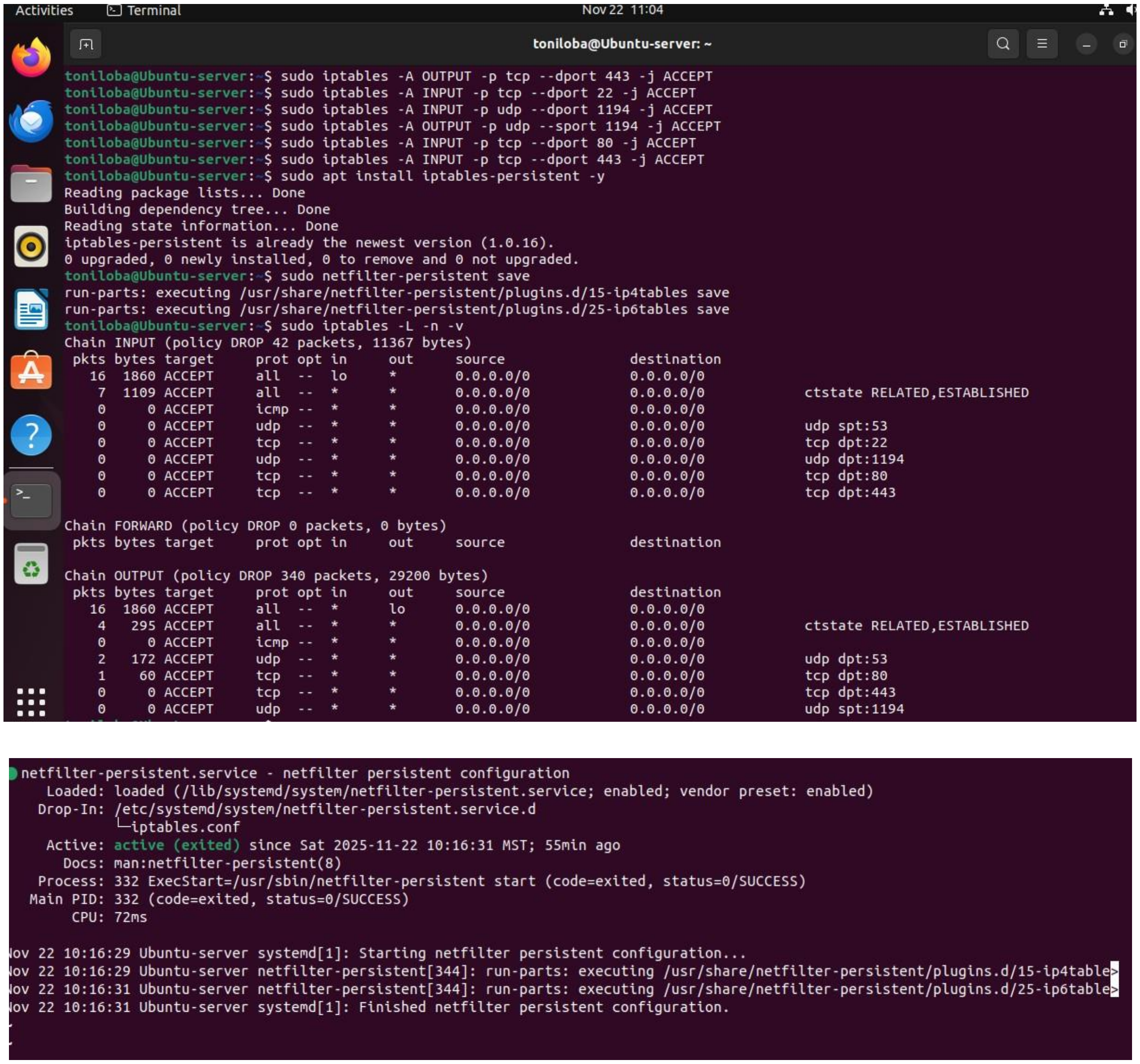
3.0 Firewall Configuration

3.1 Ubuntu (Defender) Firewall

The firewall on Ubuntu uses iptables to restrict inbound TCP traffic while allowing essential services and ICMP traffic:

i. Allowed: SSH (22), HTTP (80), VPN (1194), ICMP (ping)
ii. Blocked: All other inbound TCP ports

Command to view rules: sudo iptables -L -v --line-numbers



```
toniloba@Ubuntu-server:~$ sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT
toniloba@Ubuntu-server:~$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
toniloba@Ubuntu-server:~$ sudo iptables -A INPUT -p udp --dport 1194 -j ACCEPT
toniloba@Ubuntu-server:~$ sudo iptables -A OUTPUT -p udp --sport 1194 -j ACCEPT
toniloba@Ubuntu-server:~$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
toniloba@Ubuntu-server:~$ sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
toniloba@Ubuntu-server:~$ sudo apt install iptables-persistent -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
iptables-persistent is already the newest version (1.0.16).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
toniloba@Ubuntu-server:~$ sudo netfilter-persistent save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
toniloba@Ubuntu-server:~$ sudo iptables -L -n -v
Chain INPUT (policy DROP 42 packets, 11367 bytes)
 pkts bytes target     prot opt in     out     source               destination
   16  1860 ACCEPT     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
    7  1109 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0            ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp spt:53
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:22
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:1194
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:80
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:443

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

Chain OUTPUT (policy DROP 340 packets, 29200 bytes)
 pkts bytes target     prot opt in     out     source               destination
   16  1860 ACCEPT     all  --  *      lo      0.0.0.0/0            0.0.0.0/0
    4   295 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0            ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0
    2   172 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:53
    1    60 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:80
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:443
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp spt:1194
```

```
● netfilter-persistent.service - netfilter persistent configuration
     Loaded: loaded (/lib/systemd/system/netfilter-persistent.service; enabled; vendor preset: enabled)
    Drop-In: /etc/systemd/system/netfilter-persistent.service.d
             └─iptables.conf
     Active: active (exited) since Sat 2025-11-22 10:16:31 MST; 55min ago
       Docs: man:netfilter-persistent(8)
    Process: 332 ExecStart=/usr/sbin/netfilter-persistent start (code=exited, status=0/SUCCESS)
   Main PID: 332 (code=exited, status=0/SUCCESS)
        CPU: 72ms

Nov 22 10:16:29 Ubuntu-server systemd[1]: Starting netfilter persistent configuration...
Nov 22 10:16:29 Ubuntu-server netfilter-persistent[344]: run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4table>
Nov 22 10:16:31 Ubuntu-server netfilter-persistent[344]: run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6table>
Nov 22 10:16:31 Ubuntu-server systemd[1]: Finished netfilter persistent configuration.
```

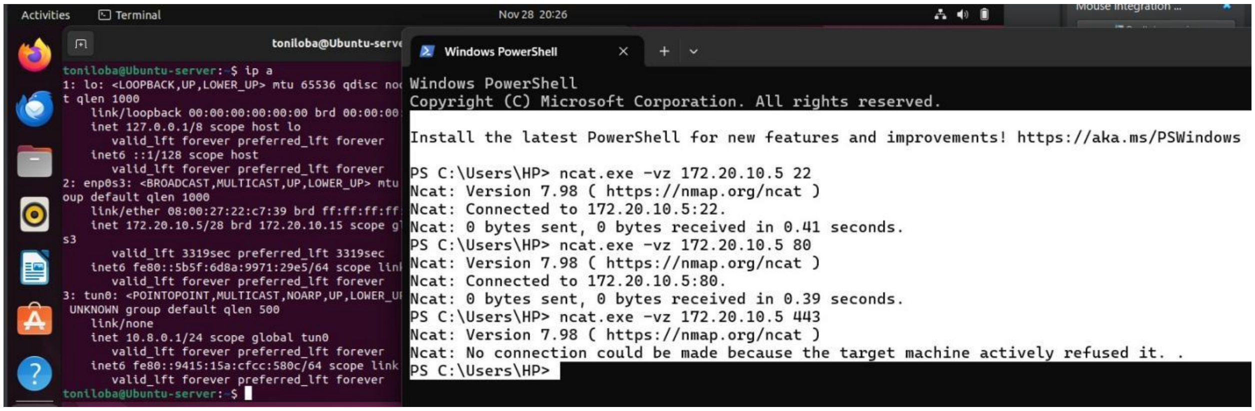
Figure 4: Ubuntu iptables firewall rules

Explanation:
The firewall ensures only required services are accessible externally, providing a baseline defense against unauthorized access. The firewall is configured using a strict allow-list model that permits only essential inbound services: SSH (22/tcp), HTTP (80/tcp), and OpenVPN (1194/udp). All other TCP ports are blocked to prevent unauthorized access or lateral movement. These active iptables rules establish the primary defensive boundary for the Ubuntu server.

ICMP is intentionally allowed to support basic connectivity checks and to enable packet-level monitoring during IDS demonstrations.

3.2 Firewall Test

To validate the firewall configuration from an external perspective, PowerShell on the Windows host was used to test connectivity to the Ubuntu server. Allowed services (HTTP 80, SSH 22) responded successfully, while restricted ports such as HTTPS 443 returned failure, confirming that the firewall rules were functioning as intended.



```
toniloba@Ubuntu-server:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noc
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu
oup default qlen 1000
    link/ether 08:00:27:22:c7:39 brd ff:ff:ff:ff:
    inet 172.20.10.5/28 brd 172.20.10.15 scope gl
s3
       valid_lft 3319sec preferred_lft 3319sec
    inet6 fe80::5b5f:6d8a:9971:29e5/64 scope lin
       valid_lft forever preferred_lft forever
3: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UF
 UNKNOWN group default qlen 500
    link/none
    inet 10.8.0.1/24 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::9415:15a:cfcc:580c/64 scope link
       valid_lft forever preferred_lft forever
toniloba@Ubuntu-server:~$
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\HP> ncat.exe -vz 172.20.10.5 22
Ncat: Version 7.98 ( https://nmap.org/ncat )
Ncat: Connected to 172.20.10.5:22.
Ncat: 0 bytes sent, 0 bytes received in 0.41 seconds.
PS C:\Users\HP> ncat.exe -vz 172.20.10.5 80
Ncat: Version 7.98 ( https://nmap.org/ncat )
Ncat: Connected to 172.20.10.5:80.
Ncat: 0 bytes sent, 0 bytes received in 0.39 seconds.
PS C:\Users\HP> ncat.exe -vz 172.20.10.5 443
Ncat: Version 7.98 ( https://nmap.org/ncat )
Ncat: No connection could be made because the target machine actively refused it. .
PS C:\Users\HP>
```

Figure 5: Firewall rules testing

Explanation:

Allowed Port (HTTP 80): PowerShell connectivity test showing that TCP port 80 on the Ubuntu Server is reachable (allowed by the firewall).

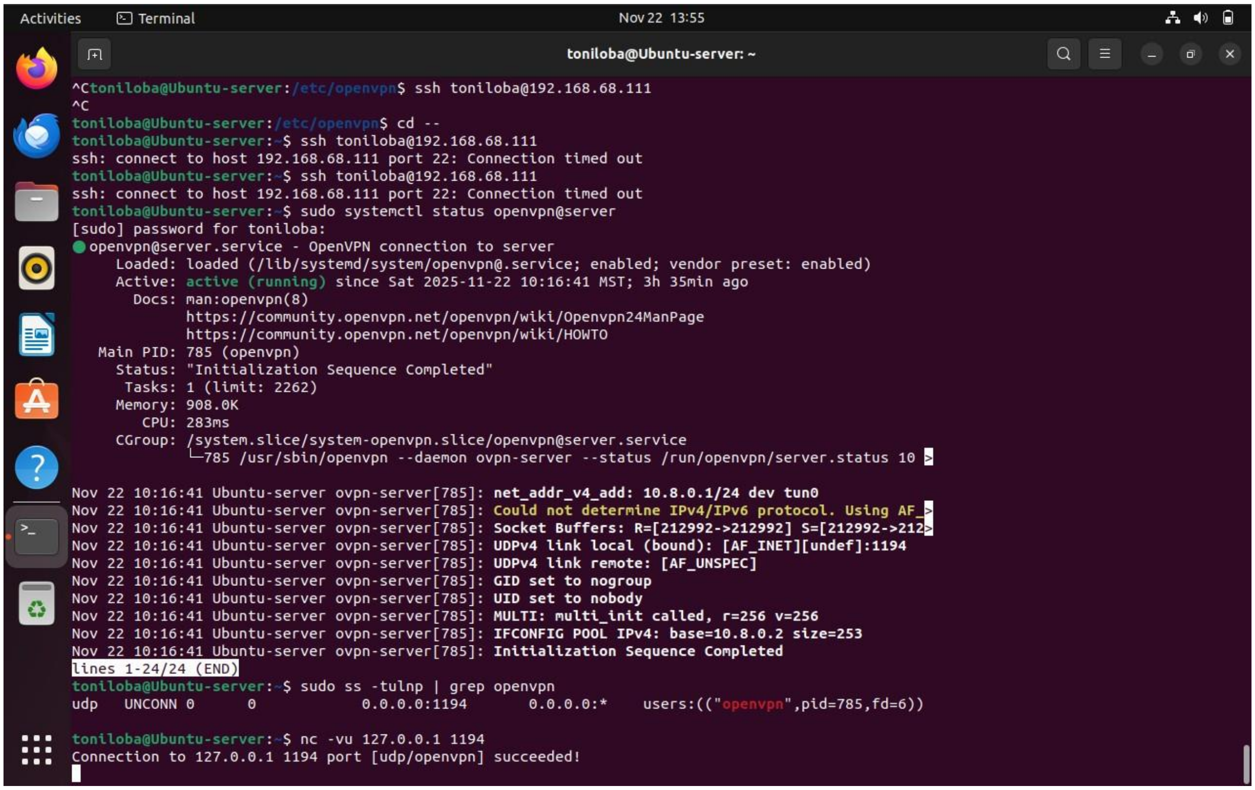
Allowed Port (SSH 22): PowerShell test confirming successful connectivity to TCP port 22 (SSH), verifying that remote access is permitted through the firewall.

Blocked Port (HTTPS 443): PowerShell test to TCP port 443 returning *No connection could be made because the target machine refused it*, demonstrating that the firewall correctly blocks unauthorized inbound ports.

4.0 VPN Configuration

An OpenVPN server was installed and configured on Ubuntu, The server successfully initialized, generated server keys and is fully operational and ready to accept external clients

i. UDP port 1194 is open and reachable (verified using netcat)
ii. Firewall rules allow VPN traffic alongside SSH and HTTP
iii. External connectivity tests confirm the server is reachable from outside the virtual network
iv. Server installed
v. Easy-RSA PKI initialized
vi. Server certificates generated



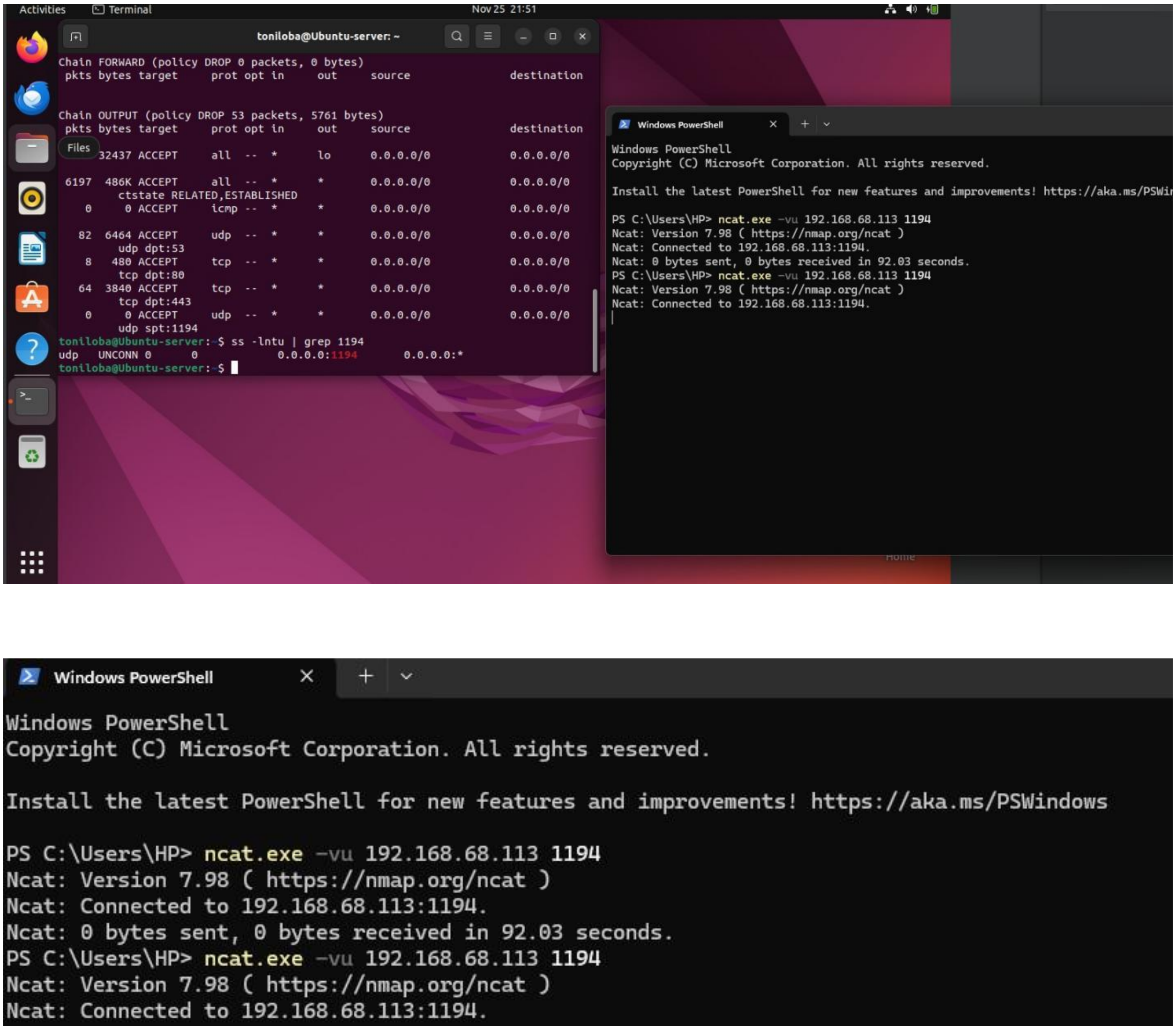
```
^Ctoniloba@Ubuntu-server:/etc/openvpn$ ssh toniloba@192.168.68.111
^C
toniloba@Ubuntu-server:/etc/openvpn$ cd --
toniloba@Ubuntu-server:~$ ssh toniloba@192.168.68.111
ssh: connect to host 192.168.68.111 port 22: Connection timed out
toniloba@Ubuntu-server:~$ ssh toniloba@192.168.68.111
ssh: connect to host 192.168.68.111 port 22: Connection timed out
toniloba@Ubuntu-server:~$ sudo systemctl status openvpn@server
[sudo] password for toniloba:
● openvpn@server.service - OpenVPN connection to server
     Loaded: loaded (/lib/systemd/system/openvpn@.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2025-11-22 10:16:41 MST; 3h 35min ago
       Docs: man:openvpn(8)
             https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
             https://community.openvpn.net/openvpn/wiki/HOWTO
   Main PID: 785 (openvpn)
     Status: "Initialization Sequence Completed"
      Tasks: 1 (limit: 2262)
     Memory: 908.0K
        CPU: 283ms
     CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
             └─785 /usr/sbin/openvpn --daemon ovpn-server --status /run/openvpn/server.status 10 >

Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: net_addr_v4_add: 10.8.0.1/24 dev tun0
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: Could not determine IPv4/IPv6 protocol. Using AF_>
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: Socket Buffers: R=[212992->212992] S=[212992->212>
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: UDPv4 link local (bound): [AF_INET][undef]:1194
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: UDPv4 link remote: [AF_UNSPEC]
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: GID set to nogroup
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: UID set to nobody
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: MULTI: multi_init called, r=256 v=256
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: IFCONFIG POOL IPv4: base=10.8.0.2 size=253
Nov 22 10:16:41 Ubuntu-server ovpn-server[785]: Initialization Sequence Completed
lines 1-24/24 (END)
toniloba@Ubuntu-server:~$ sudo ss -tulnp | grep openvpn
udp   UNCONN 0      0            0.0.0.0:1194       0.0.0.0:*    users:(("openvpn",pid=785,fd=6))

toniloba@Ubuntu-server:~$ nc -vu 127.0.0.1 1194
Connection to 127.0.0.1 1194 port [udp/openvpn] succeeded!
```

Figure 6: OpenVPN server status, ss output showing listening port 1194

Explanation:
This configuration demonstrates secure remote access to the internal network and confirms that firewall rules properly allow VPN traffic without compromising security



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


Chain OUTPUT (policy DROP 53 packets, 5761 bytes)
 pkts bytes target     prot opt in     out     source               destination
     32437 ACCEPT     all  --  *      lo      0.0.0.0/0            0.0.0.0/0

 6197  486K ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0
         ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0

   82  6464 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
         udp dpt:53
    8   480 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
         tcp dpt:80
   64  3840 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
         tcp dpt:443
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
         udp spt:1194
toniloba@Ubuntu-server:~$ ss -lntu | grep 1194
udp   UNCONN 0      0            0.0.0.0:1194       0.0.0.0:*
toniloba@Ubuntu-server:~$
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWi

PS C:\Users\HP> ncat.exe -vu 192.168.68.113 1194
Ncat: Version 7.98 ( https://nmap.org/ncat )
Ncat: Connected to 192.168.68.113:1194.
Ncat: 0 bytes sent, 0 bytes received in 92.03 seconds.
PS C:\Users\HP> ncat.exe -vu 192.168.68.113 1194
Ncat: Version 7.98 ( https://nmap.org/ncat )
Ncat: Connected to 192.168.68.113:1194.
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\HP> ncat.exe -vu 192.168.68.113 1194
Ncat: Version 7.98 ( https://nmap.org/ncat )
Ncat: Connected to 192.168.68.113:1194.
Ncat: 0 bytes sent, 0 bytes received in 92.03 seconds.
PS C:\Users\HP> ncat.exe -vu 192.168.68.113 1194
Ncat: Version 7.98 ( https://nmap.org/ncat )
Ncat: Connected to 192.168.68.113:1194.
```

Figure 7: External connectivity test success

Explanation:

This shows external test from my host laptop acting as an external device, I tested connectivity to the VPN server using Ncat.

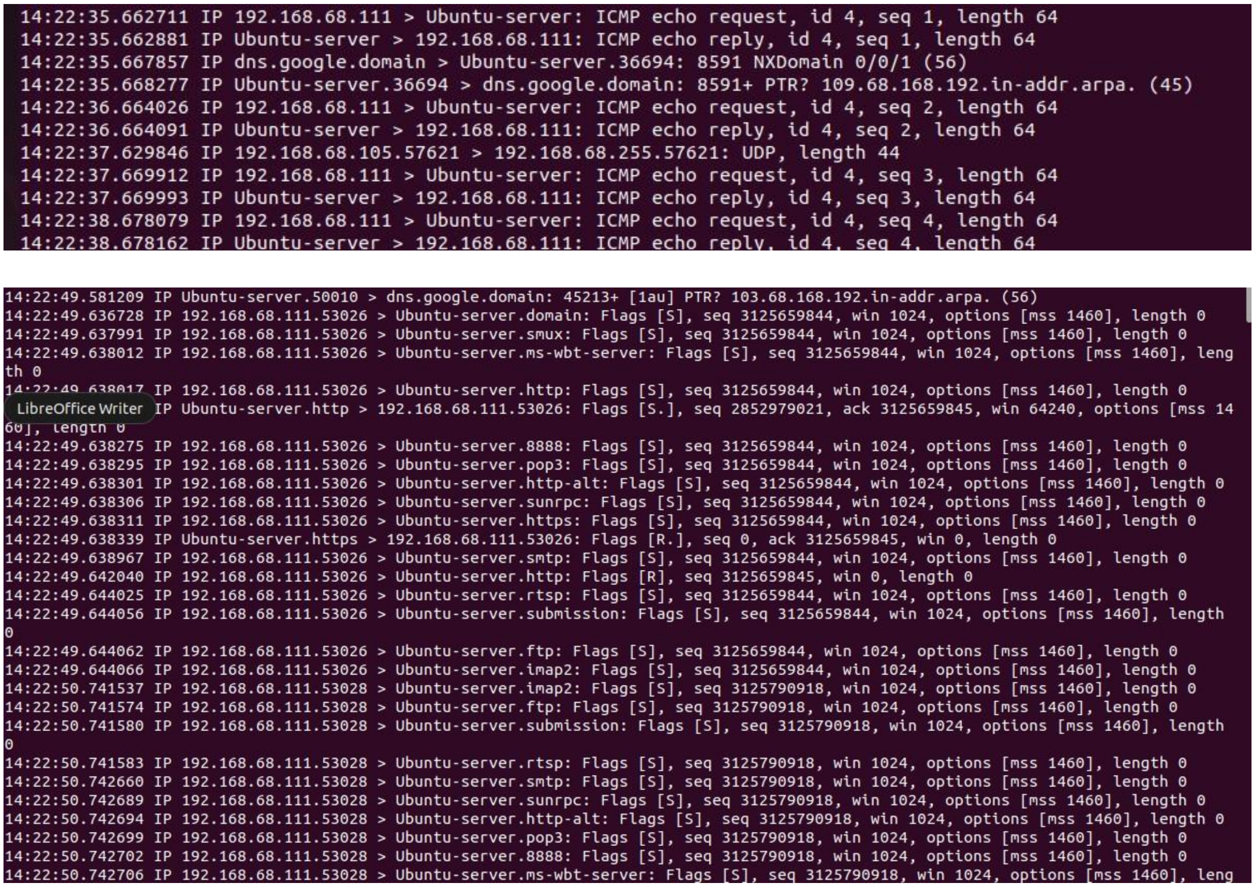
Result: “The output was: Ncat: Connected to 192.168.68.113:1194” **-** This shows that packets from an external device successfully reached the VPN server and the firewall allowed the traffic.

5.0 Intrusion Detection (IDS)

To simulate attack detection capabilities, tcpdump was used on Ubuntu to capture incoming packets, while Nmap active scans were run from Kali.

5.1 Packet Capture/ICMP Test Ping

i. ICMP packets (ping) were captured as echo requests and replies
ii. TCP SYN packets from Nmap scans were captured



```
14:22:35.662711 IP 192.168.68.111 > Ubuntu-server: ICMP echo request, id 4, seq 1, length 64
14:22:35.662881 IP Ubuntu-server > 192.168.68.111: ICMP echo reply, id 4, seq 1, length 64
14:22:35.667857 IP dns.google.domain > Ubuntu-server.36694: 8591 NXDomain 0/0/1 (56)
14:22:35.668277 IP Ubuntu-server.36694 > dns.google.domain: 8591+ PTR? 109.68.168.192.in-addr.arpa. (45)
14:22:36.664026 IP 192.168.68.111 > Ubuntu-server: ICMP echo request, id 4, seq 2, length 64
14:22:36.664091 IP Ubuntu-server > 192.168.68.111: ICMP echo reply, id 4, seq 2, length 64
14:22:37.629846 IP 192.168.68.105.57621 > 192.168.68.255.57621: UDP, length 44
14:22:37.669912 IP 192.168.68.111 > Ubuntu-server: ICMP echo request, id 4, seq 3, length 64
14:22:37.669993 IP Ubuntu-server > 192.168.68.111: ICMP echo reply, id 4, seq 3, length 64
14:22:38.678079 IP 192.168.68.111 > Ubuntu-server: ICMP echo request, id 4, seq 4, length 64
14:22:38.678162 IP Ubuntu-server > 192.168.68.111: ICMP echo reply, id 4, seq 4, length 64
```

```
14:22:49.581209 IP Ubuntu-server.50010 > dns.google.domain: 45213+ [1au] PTR? 103.68.168.192.in-addr.arpa. (56)
14:22:49.636728 IP 192.168.68.111.53026 > Ubuntu-server.domain: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.637991 IP 192.168.68.111.53026 > Ubuntu-server.smux: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.638012 IP 192.168.68.111.53026 > Ubuntu-server.ms-wbt-server: Flags [S], seq 3125659844, win 1024, options [mss 1460], leng
th 0
14:22:49.638017 IP 192.168.68.111.53026 > Ubuntu-server.http: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
[illegible] IP Ubuntu-server.http > 192.168.68.111.53026: Flags [S.], seq 2852979021, ack 3125659845, win 64240, options [mss 14
60], length 0
14:22:49.638275 IP 192.168.68.111.53026 > Ubuntu-server.8888: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.638295 IP 192.168.68.111.53026 > Ubuntu-server.pop3: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.638301 IP 192.168.68.111.53026 > Ubuntu-server.http-alt: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.638306 IP 192.168.68.111.53026 > Ubuntu-server.sunrpc: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.638311 IP 192.168.68.111.53026 > Ubuntu-server.https: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.638339 IP Ubuntu-server.https > 192.168.68.111.53026: Flags [R.], seq 0, ack 3125659845, win 0, length 0
14:22:49.638967 IP 192.168.68.111.53026 > Ubuntu-server.smtp: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.642040 IP 192.168.68.111.53026 > Ubuntu-server.http: Flags [R], seq 3125659845, win 0, length 0
14:22:49.644025 IP 192.168.68.111.53026 > Ubuntu-server.rtsp: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.644056 IP 192.168.68.111.53026 > Ubuntu-server.submission: Flags [S], seq 3125659844, win 1024, options [mss 1460], length
0
14:22:49.644062 IP 192.168.68.111.53026 > Ubuntu-server.ftp: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:49.644066 IP 192.168.68.111.53026 > Ubuntu-server.imap2: Flags [S], seq 3125659844, win 1024, options [mss 1460], length 0
14:22:50.741537 IP 192.168.68.111.53028 > Ubuntu-server.imap2: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.741574 IP 192.168.68.111.53028 > Ubuntu-server.ftp: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.741580 IP 192.168.68.111.53028 > Ubuntu-server.submission: Flags [S], seq 3125790918, win 1024, options [mss 1460], length
0
14:22:50.741583 IP 192.168.68.111.53028 > Ubuntu-server.rtsp: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.742660 IP 192.168.68.111.53028 > Ubuntu-server.smtp: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.742689 IP 192.168.68.111.53028 > Ubuntu-server.sunrpc: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.742694 IP 192.168.68.111.53028 > Ubuntu-server.http-alt: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.742699 IP 192.168.68.111.53028 > Ubuntu-server.pop3: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.742702 IP 192.168.68.111.53028 > Ubuntu-server.8888: Flags [S], seq 3125790918, win 1024, options [mss 1460], length 0
14:22:50.742706 IP 192.168.68.111.53028 > Ubuntu-server.ms-wbt-server: Flags [S], seq 3125790918, win 1024, options [mss 1460], leng
```

Figure 8: tcpdump capturing ICMP and TCP SYN packets during connectivity test.

5.2 Attack Simulation and Analysis

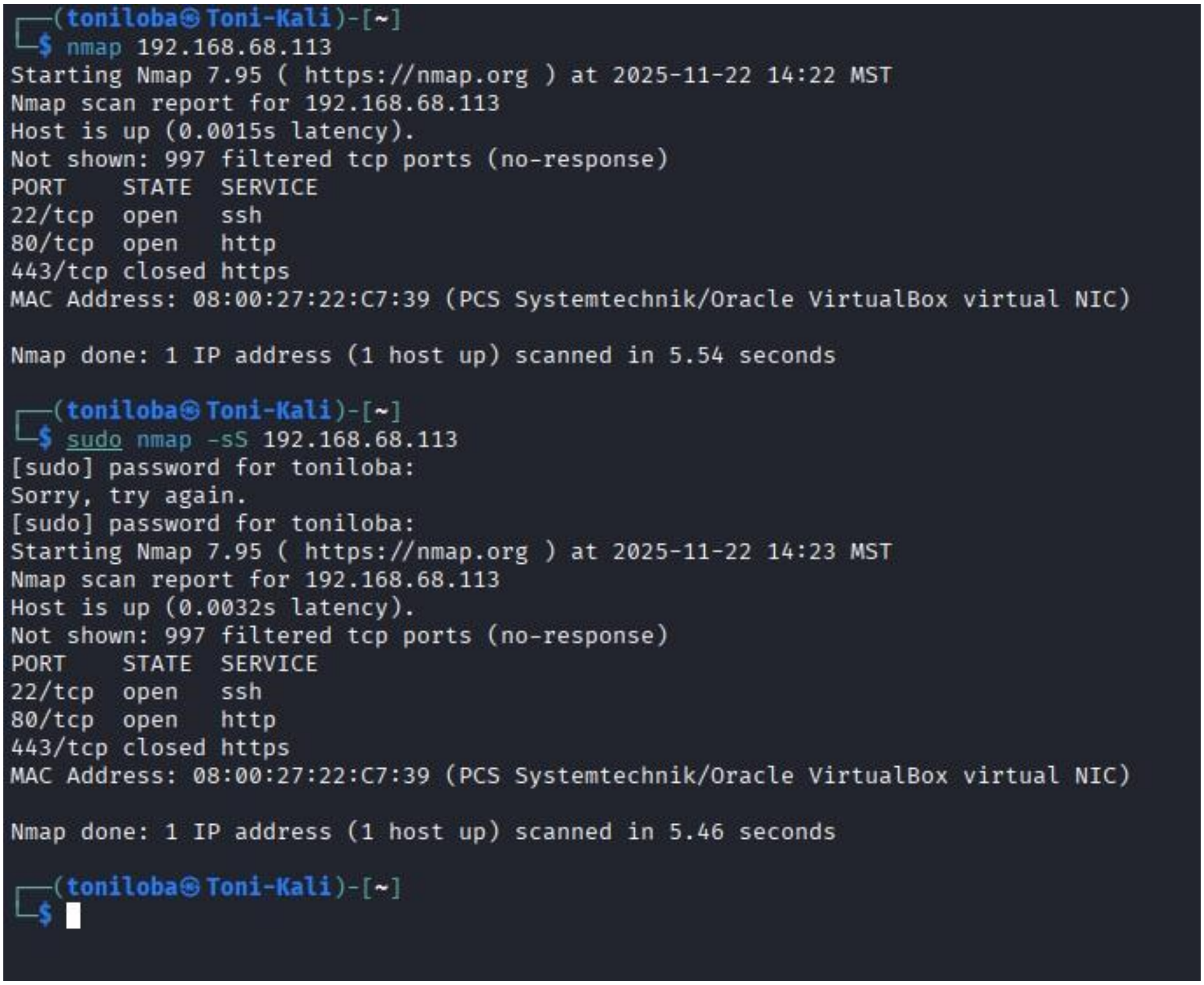
From Kali, the following scans were performed:

ping -c 4 <Ubuntu_IP>

nmap <Ubuntu_IP>

sudo nmap -sS <Ubuntu_IP>

Results:



```
┌──(toniloba㉿Toni-Kali)-[~]
└─$ nmap 192.168.68.113
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-22 14:22 MST
Nmap scan report for 192.168.68.113
Host is up (0.0015s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT    STATE  SERVICE
22/tcp  open   ssh
80/tcp  open   http
443/tcp closed https
MAC Address: 08:00:27:22:C7:39 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 5.54 seconds

┌──(toniloba㉿Toni-Kali)-[~]
└─$ sudo nmap -sS 192.168.68.113
[sudo] password for toniloba:
Sorry, try again.
[sudo] password for toniloba:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-22 14:23 MST
Nmap scan report for 192.168.68.113
Host is up (0.0032s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT    STATE  SERVICE
22/tcp  open   ssh
80/tcp  open   http
443/tcp closed https
MAC Address: 08:00:27:22:C7:39 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 5.46 seconds

┌──(toniloba㉿Toni-Kali)-[~]
└─$ 
```

Figure 9: Nmap scan results from Kali

Observations:

i. ICMP echo requests/replies confirm allowed ping traffic
ii. TCP SYN packets targeting ports 22, 80, 443 were captured in tcpdump
iii. Firewall allowed only necessary ports: blocked ports show no replies, matching Nmap results
iv. IDS effectively detected scan attempts, providing evidence of potential reconnaissance activity

Explanation:
This demonstrates the IDS functionality, capturing suspicious packets in real-time and validating firewall rules. The combination of packet monitoring and filtered ports provides a basic but effective defense simulation.

6.0 Testing, Analysis and Security Observations

i. **Firewall Strengths:** selectively allows necessary services and blocks unnecessary traffic
ii. **VPN Strengths:** ensures secure remote access while permitting only authorized services
iii. **IDS Strengths:** detects reconnaissance attempts like ping sweeps and SYN scans

7.0 Conclusion

This project successfully implemented a secure and controlled network environment integrating three core security mechanisms: a packet-filtering firewall, a VPN service and an intrusion detection simulation. The Ubuntu server consistently enforced strict access control by allowing only essential services while blocking all unnecessary TCP ports. OpenVPN provided a reliable and encrypted remote-access pathway, demonstrating how secure connectivity can be maintained without exposing internal services.

Through tcpdump monitoring, the IDS component effectively identified reconnaissance behavior such as ICMP probes and TCP SYN scans originating from Kali, validating the visibility and detection capability of the setup. Combined with external testing from the Windows host, the project confirmed that defensive configurations behaved correctly across multiple attack and access paths.

Overall, this work strengthened practical cybersecurity skills and reinforced key concepts in network hardening, traffic analysis and secure service deployment within a realistic, hands-on environment.