

Project 2

Building a Secure Small Network: Design, Implementation, and Defense in a Virtualized Lab Environment

Students Name: Nwosu Onyinye Susan

Institution: TechCrush HQ

Course: Cybersecurity Capstone Project

Instructors: Jeremiah Olubunmi, Faith Olagoke

Submission Date: June, 2025

Table of Contents

- Introduction
- Project Background and Rationale
- Statement of the Problem
- Aims and Objectives
- Scope and Limitations
- Tools and Technologies
- Methodology
- Network Design and Architecture
- Implementation
- Security and Control
- Testing and Validation
- Simulated Attack Scenario
- Result and Discussion
- Challenges and Lesson Learned
- Next Step and Recommendations
- Conclusion
- References

- Appendices

1. Introduction

In the age of digital transformation, every business, small or large is at risk of cybersecurity threats. This project, titled “Building a Secure Small Network,” explores the practical aspects of designing, implementing, and securing a small network architecture in a virtualized environment using accessible open-source tools. The goal is to simulate a real-world small enterprise setup and test its resilience to network-based attacks, especially in segmented networks with access control rules.

This report provides comprehensive documentation of a defensive strategy that includes firewall implementation using pfSense, host-based protection using UFW, network reconnaissance detection using Wireshark, and simulation of attack attempts from an isolated attacker VM running Kali Linux.

2. Project Background and Rationale

In most learning environments, real-world cybersecurity experience is difficult to simulate. Many students are left with theoretical knowledge that does not adequately prepare them for practical defense scenarios. To bridge this gap, this capstone project creates a lab-based simulation using VirtualBox and free security tools to:

- Set up a segmented network using pfSense as a router and firewall
- Emulate real-world internal networks using Ubuntu Server and Desktop
- Simulate an external attacker using Kali Linux on a separate network
- Implement, test, and analyze various network security strategies

The rationale is to empower entry-level cybersecurity professionals with hands-on skills needed in securing enterprise networks, understanding threat vectors, and implementing prevention and detection mechanisms.

3. Statement of the Problem

Small networks are often seen as low-risk targets; however, they are frequently vulnerable due to poor segmentation, limited perimeter defenses, and lack of monitoring. This project aims to address the following challenges:

- How can network segmentation be enforced in a virtual lab?
- Can a firewall (pfSense) and host-based protections (UFW) effectively prevent unauthorized lateral movement?
- How visible are attack attempts from a separate subnet using standard tools like Nmap and Wireshark?

4. Aims and Objectives

The primary aim of this capstone project is to design and secure a small virtual network against unauthorized access and network reconnaissance attacks.

Objectives:

- Build a virtual lab consisting of:
 - pfSense firewall/router
 - Two internal hosts (Ubuntu Desktop and Server)
 - One external attacker VM (Kali Linux)
 - Configure separate VirtualBox networks:
 - intnet (Internal Network – LAN) with subnet [192.168.10.0/24](#)
 - Kali Attacker (Separate attacker network – [192.168.20.0/24](#))
 - Configure pfSense with 3 interfaces:
 - em0: WAN (NAT - [10.0.2.15](#))
 - em1: LAN (Internal Network - [192.168.10.1](#))

- em2: Kali Attacker (Isolated Network - 192.168.20.1)
- Apply firewall rules in pfSense to block the attacker from reaching internal hosts
- Harden Ubuntu systems using UFW
- Test connectivity and simulate scans using Nmap
- Capture and analyze traffic with Wireshark
- Reflect on effectiveness of implemented security controls

5. Scope and Limitations

Scope:

- The project simulates a small office network with basic services and limited devices.
- Focuses on layer 3 network security: segmentation, routing, and basic ACLs.
- Uses VirtualBox and open-source software exclusively.
- Designed for educational and demonstration purposes.

Limitations:

- No cloud integration or hybrid environments.
- No advanced intrusion detection/prevention systems (e.g., Suricata, Snort).
- User authentication and web-facing services are not part of this build.
- Performance metrics (e.g., latency, packet loss under stress) not captured.

6. Tools and Technologies

Tools	Description
VirtualBox Hypervisor	for creating isolated virtual machines
pfSense	Open-source firewall and router deployed as VM
Ubuntu Server/Desktop	Representing internal machines (secured via UFW) system and server
Kali Linux	Simulated attacker machine for scanning and probing
UFW	Host-based firewall for Ubuntu systems
Wireshark	Network traffic analyzer used on internal hosts
Nmap	Network scanning and reconnaissance tool and port scanning
Cisco Packet Tracer	Network design and planning tool

7. Methodology

Phase 1: Network Planning and Logical Design

- IP schema was drafted manually with clearly separated networks:
 - Internal ([192.168.10.0/24](#))
 - Kali Attacker ([192.168.20.0/24](#))
 - WAN ([10.0.2.0/24](#) via NAT)

Phase 2: VirtualBox Setup

- pfSense configured with 3 interfaces:
- em0 – NAT ([10.0.2.15](#))
- em1 – LAN ([192.168.10.1](#)) on intnet
- em2 – Attacker network ([192.168.20.1](#)) on Kali Attacker
- Ubuntu Desktop: [192.168.10.51](#) (connected to intnet)
- Ubuntu Server: [192.168.10.50](#) (connected to intnet)
- Kali Linux: [192.168.20.52](#) (connected to Kali Attacker)

Phase 3: Configuration

- Kali configured via static ip
- pfSense DHCP disabled for LAN then enabled for Vms in LAN
- Each machine could ping the gateway ([192.168.10.1](#) or [192.168.20.1](#)).
- Routing only permitted through pfSense.

Phase 4: Firewall Rules and UFW

- On pfSense, created rule on em2 interface to block all traffic from [192.168.20.0/24](#) to [192.168.10.0/24](#).

- On Ubuntu Server/Desktop, enabled UFW and only allowed SSH and ICMP.

Phase 5: Simulated Attack

- Kali attempted Nmap scans on internal network when port 80 was open and when it was closed
- Results captured and analyzed to verify if connections were blocked.
- Ubuntu desktop monitored traffic using Wireshark.

Phase 6: Documentation and Review

- All output saved with time-stamped logs and screenshots.
- Reflections and lessons noted.

8. Network Design and Architecture

The network was intentionally designed to reflect a real-world segmented architecture. Three virtual networks were configured:

- WAN ([10.0.2.0/24](#)) – Internet-facing segment (NAT in VirtualBox)
 - LAN ([192.168.10.0/24](#)) – Trusted internal network with workstations and servers
 - Kali Attacker ([192.168.20.0/24](#)) – Untrusted isolated attacker network
- pfSense connects all networks via virtual interfaces:
- em0 - NAT
 - em1 - intnet
 - em2 - Kali Attacker

This segmentation allows precise control over which IP ranges can communicate.

8.1 Network Setup Checklist

This checklist was used to verify that all virtual machines, configurations, and network interfaces were correctly prepared before implementing security controls and testing the network.

Virtual Machines

VM Name	OS	Role	Status
pfSense	Firewall pfSense 2.7+	Router/Firewall	Active
Ubuntu Desktop	Ubuntu 22.04 LTS	Internal User System	Active
Ubuntu Server	Ubuntu 22.04 LTS	Internal Server	Active
Kali Attacker	Kali Linux 2024.1	Attacker Machine	Active

VirtualBox Network Adapter Configuration

VM Name	Adapter 1	Adapter 2	Adapter 3
pfSense Firewall	NAT (WAN)	Internal Network (LAN)	Intnet Internal Network (To Attacker)
Kali Attacker	Internal Network → Kali Attacker		
Ubuntu Desktop	Internal Network → intnet		
Ubuntu Server	Internal Network → intnet		

Pre-Configuration Verification

- All VMs successfully powered on with correct IP assignments.
- pfSense LAN IP set to **192.168.10.1/24**.
- WAN side of pfSense receives IP via DHCP (e.g., **10.0.2.15**).
- Ubuntu Desktop: IP = **192.168.10.51**, Gateway = **192.168.10.1**.
- Ubuntu Server: IP = **192.168.10.50**, Gateway = **192.168.10.1**.
- Kali Attacker: IP = **192.168.20.52**, separate network (Kali_Attacker).
- pfSense able to ping both internal and external networks.
- SSH and UFW installed and enabled on Ubuntu systems.
- Static routes and DNS configured properly on all VMs.
- Screenshots taken of VirtualBox adapter settings (Insert Here).
- Screenshots taken of pfSense interface assignments and status (Insert Here).

9. Implementation

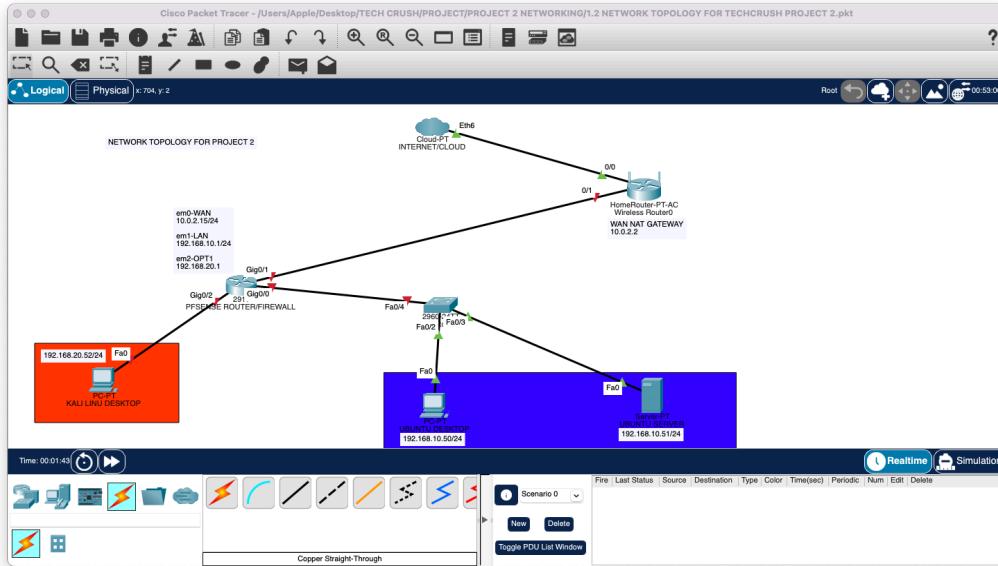
The implementation phase involved setting up the virtual environment with multiple VMs, configuring network interfaces, and applying security policies to enforce network segmentation and access controls.

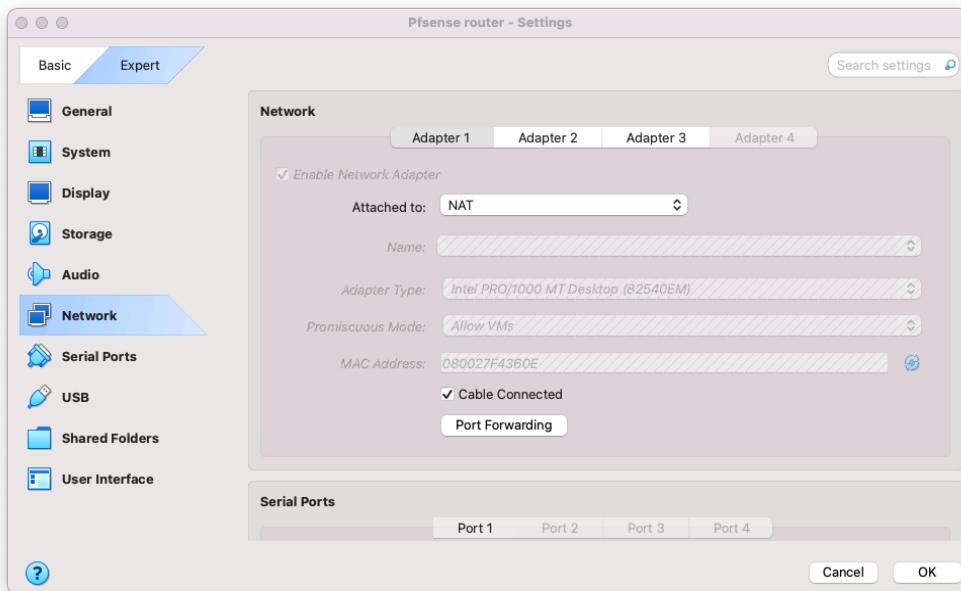
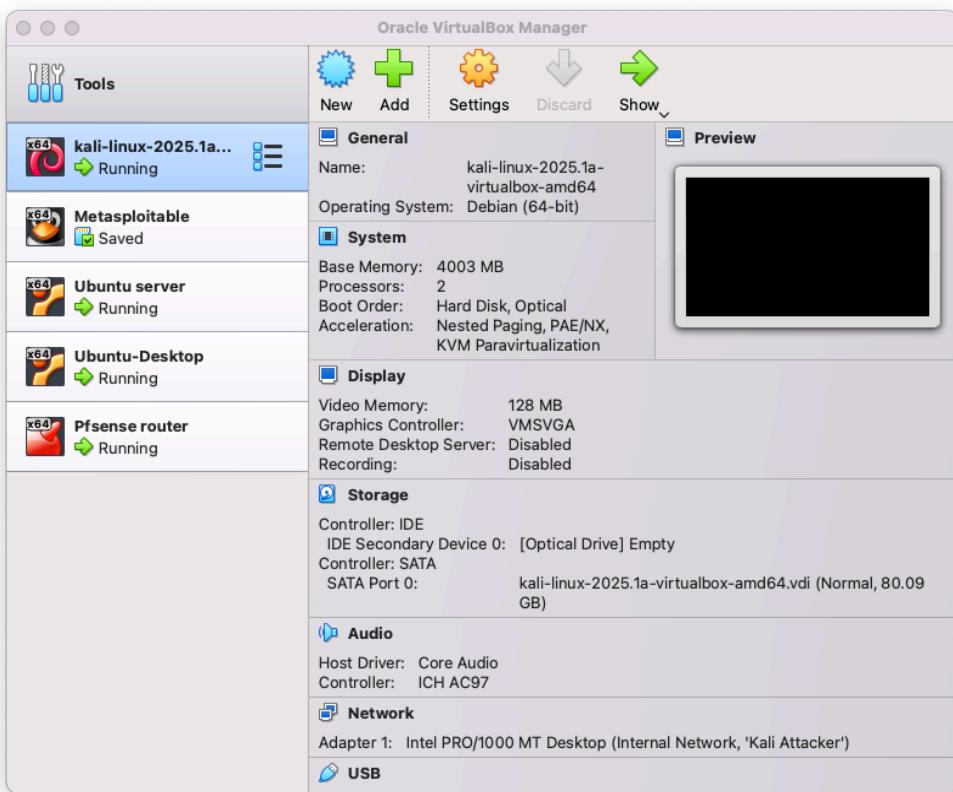
9.1 Virtual Machines Setup

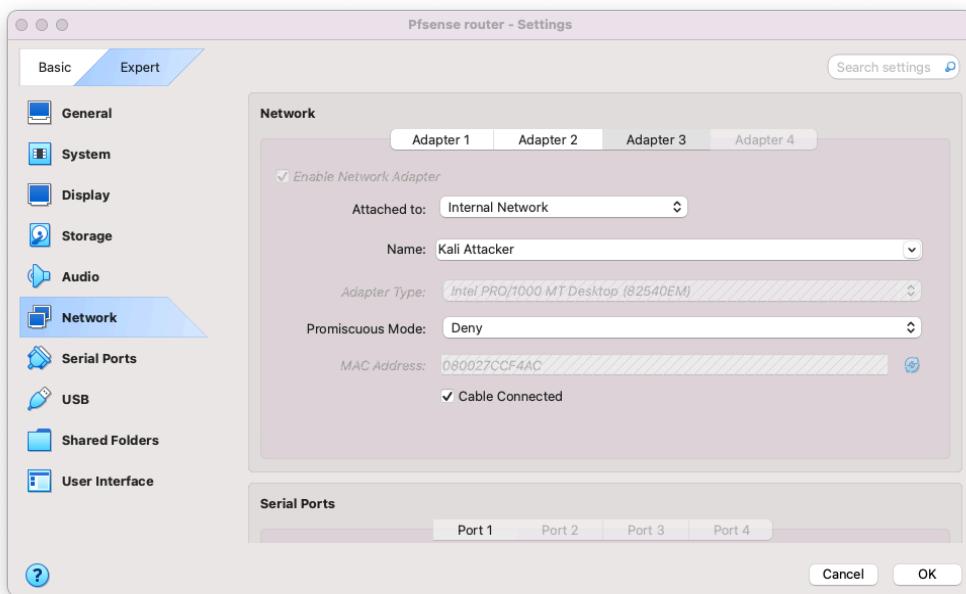
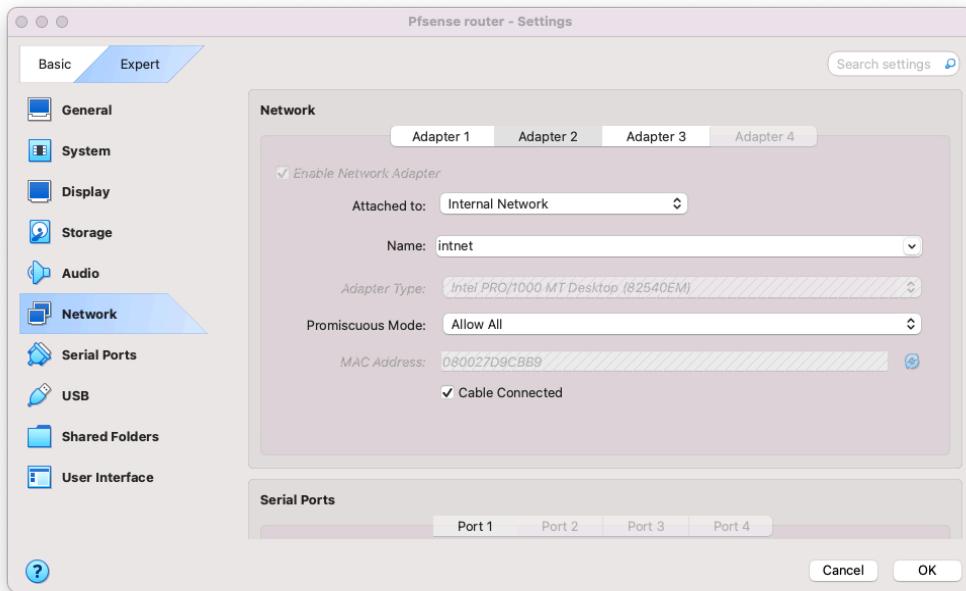
- Network Diagram and topology, IPs, interfaces (em0, em1, em2), and machine roles we displayed using Cisco Packet Tracer
- The virtual network was constructed using VirtualBox, deploying four VMs:
- pfSense Firewall/Router VM with three NICs:
- em0 → WAN interface configured as NAT, with IP **10.0.2.15** assigned by VirtualBox DHCP
- em1 → LAN interface on network named intnet (IP **192.168.10.1**)
- em2 → Kali Attacker interface on separate network named Kali Attacker (IP **192.168.20.1**)
- Ubuntu Server VM connected to the LAN network (intnet)

- Ubuntu Desktop VM connected to the LAN network (intnet)
- Kali Linux VM connected to the attacker network (Kali Attacker)

Network Topology, Vms installed and setup







pfSense Interface

Pfsense router [Running]

```
Starting CRON... done.
pfSense 2.8.0-RELEASE amd64 20250521-2312
Bootup complete

FreeBSD/amd64 (pfSense.local.lan) (ttyv0)

VirtualBox Virtual Machine - Netgate Device ID: 6b188b00a597788d1d32

*** Welcome to pfSense 2.8.0-RELEASE (amd64) on pfSense ***

WAN (wan)      -> em0 -> v4/DHCP4: 10.0.2.15/24
LAN (lan)      -> em1 -> v4: 192.168.10.1/24
KALIATTACKER (opt1) -> em2 -> v4: 192.168.20.1/24

0) Logout / Disconnect SSH          9) pfTop
1) Assign Interfaces                10) Filter Logs
2) Set interface(s) IP address     11) Restart GUI
3) Reset admin account and password 12) PHP shell + pfSense tools
4) Reset to factory defaults       13) Update from console
5) Reboot system                   14) Enable Secure Shell (sshd)
6) Halt system                     15) Restore recent configuration
7) Ping host                       16) Restart PHP-FPM
8) Shell

Enter an option: ■
```

9.2 Operating System Network Configuration (DHCP Setup)

- The pfSense DHCP server was enabled on the LAN interface (em1) to dynamically assign IP addresses to devices on the intnet network.
- Ubuntu Server and Ubuntu Desktop were configured to obtain their IP addresses via DHCP:
- Ubuntu Desktop received IP: [192.168.10.51](#)
- Ubuntu Desktop received IP: [192.168.10.50](#)
- The subnet mask is /24, and the default gateway provided via DHCP is [192.168.10.1](#) (pfSense LAN interface).
- DNS settings were also delivered by pfSense to the Ubuntu machines.

Ubuntu desktop ip

```

    2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
        link/ether 08:00:27:1b:16:dd brd ff:ff:ff:ff:ff:ff
        inet 192.168.10.51/24 brd 192.168.10.255 scope global dynamic noprefixroute
          valid_lft 6975sec preferred_lft 6975sec
          inet6 fe80::a00:27ff:fe1b:16dd/64 scope link
            valid_lft forever preferred_lft forever
suz@suz-VirtualBox$ 

```

Ubuntu server ip

```

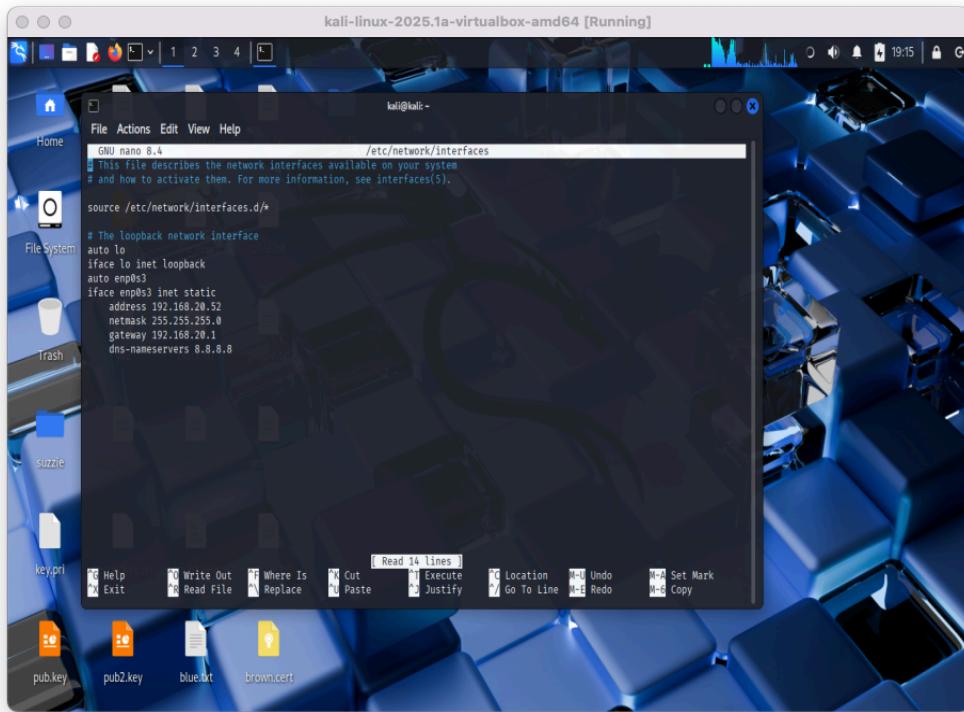
See https://ubuntu.com/esm or run: sudo pro status

officeuser@officeserver:~$ 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 brd 127.255.255.255 scope host
      valid_lft forever preferred_lft forever
      inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:14:8eff brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.50/24 brd 192.168.10.255 scope global dynamic enp0s3
      valid_lft 6975sec preferred_lft 6975sec

```

9.3 pfSense Firewall and Routing Configuration

- The WAN interface em0 was configured to receive IP dynamically via NAT from VirtualBox with IP [10.0.2.15](#).
- The LAN interface em1 was statically set to IP [192.168.10.1](#) and served as the DHCP server for Ubuntu machines.
- The attacker interface em2 was statically assigned IP [192.168.20.1](#) for the Kali Attacker network using (sudo nano /etc/network/interfaces)
- Routing and firewall policies were configured to enable controlled communication between networks and isolate the attacker network.



WAN, LAN, and ATTACKER Interface on pfSense GUI

A screenshot of the pfSense Status / Dashboard page. The page displays system information such as the system name (pfSense.local.lan), user (admin@192.168.10.51), and hardware details (Intel(R) Core(TM) i5-5350U CPU @ 1.80GHz). It also shows network interface status: WAN (10.0.2.15, 1000baseT <full-duplex>), LAN (192.168.10.1, 1000baseT <full-duplex>), and KALIATTACKER (192.168.20.1, 1000baseT <full-duplex>). A message indicates that support information is being retrieved. The browser tab shows the URL https://192.168.10.1.

pfSense Routing Table

The screenshot shows a Firefox browser window titled "Ubuntu-Desktop [Running]" with the URL "https://192.168.10.1/diag_routes.php". The page is titled "Diagnostics / Routes" and displays the "Routing Table Display Options" section. It includes checkboxes for "Resolve names" (unchecked) and "Enable", a dropdown for "Rows to display" set to 100, and a "Filter" input field. Below this is a table titled "IPv4 Routes" with the following data:

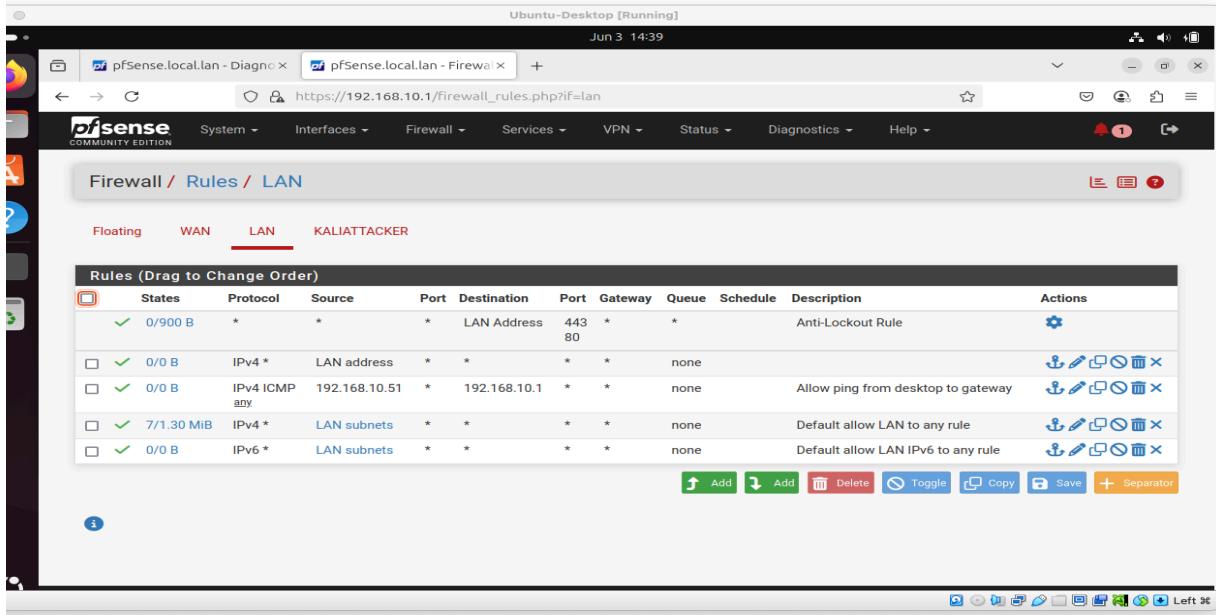
Destination	Gateway	Flags	Uses	MTU	Interface	Expire
0.0.0.0	10.0.2.2	UGS	9	1500	em0	
10.0.2.0/24	link#1	U	1	1500	em0	
10.0.2.2	link#1	UHS	8	1500	em0	
10.0.2.15	link#5	UHS	3	16384	lo0	
127.0.0.1	link#5	UH	2	16384	lo0	

10. Security Controls

10.1 pfSense Firewall Rules

- Firewall rules on the em2 interface (Kali Attacker) deny all traffic destined for the LAN subnet (192.168.10.0/24), preventing the attacker VM from reaching the internal network.
- Default allow rules were maintained on LAN (em1) to permit normal internal communications.

Allow rule on LANs

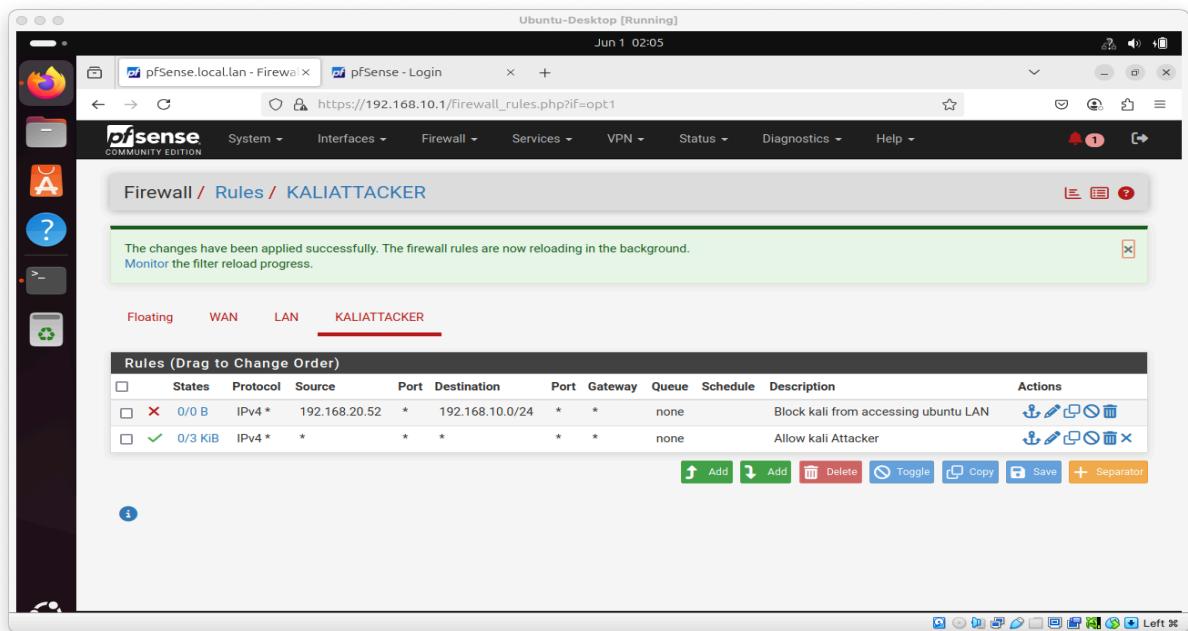


The screenshot shows the pfSense Firewall Rules configuration for the LAN interface. The LAN tab is selected. There are five rules listed:

States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✓ 0/900 B	*	*	*	LAN Address	443	*	*		Anti-Lockout Rule	
✗ 0/0 B	IPv4 *	LAN address	*	*	*	*	none			
✗ 0/0 B	IPv4 ICMP any	192.168.10.51	*	192.168.10.1	*	*	none		Allow ping from desktop to gateway	
✗ 7/1.30 MiB	IPv4 *	LAN subnets	*	*	*	*	none		Default allow LAN to any rule	
✗ 0/0 B	IPv6 *	LAN subnets	*	*	*	*	none		Default allow LAN IPv6 to any rule	

Buttons at the bottom include Add, Delete, Toggle, Copy, Save, and Separator.

Block rule on Kali enabled

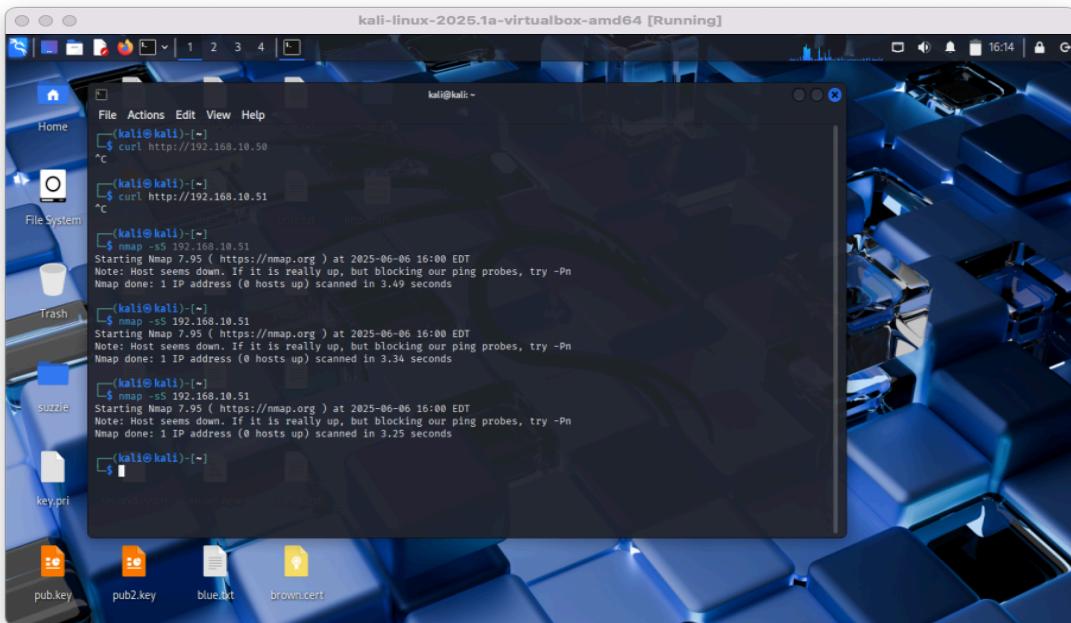
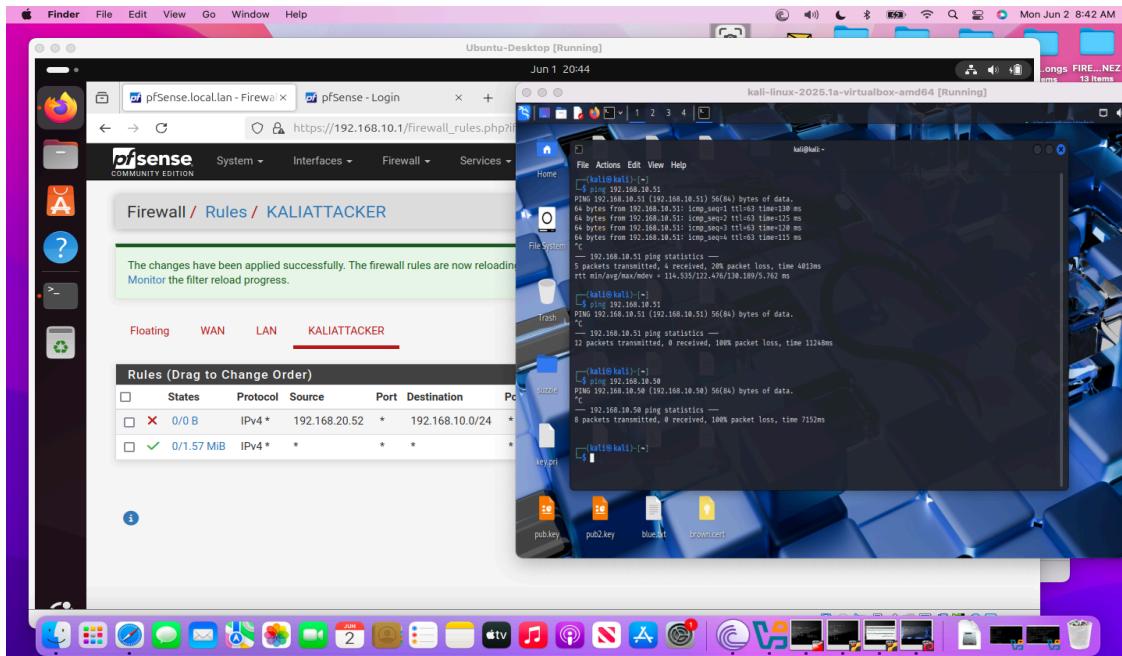


The screenshot shows the pfSense Firewall Rules configuration for the KALIATTACKER interface. The KALIATTACKER tab is selected. A message at the top states: "The changes have been applied successfully. The firewall rules are now reloading in the background. Monitor the filter reload progress." There are two rules listed:

States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✗ 0/0 B	IPv4 *	192.168.20.52	*	192.168.10.0/24	*	*	none		Block kali from accessing ubuntu LAN	
✓ 0/3 KIB	IPv4 *	*	*	*	*	*	none		Allow kali Attacker	

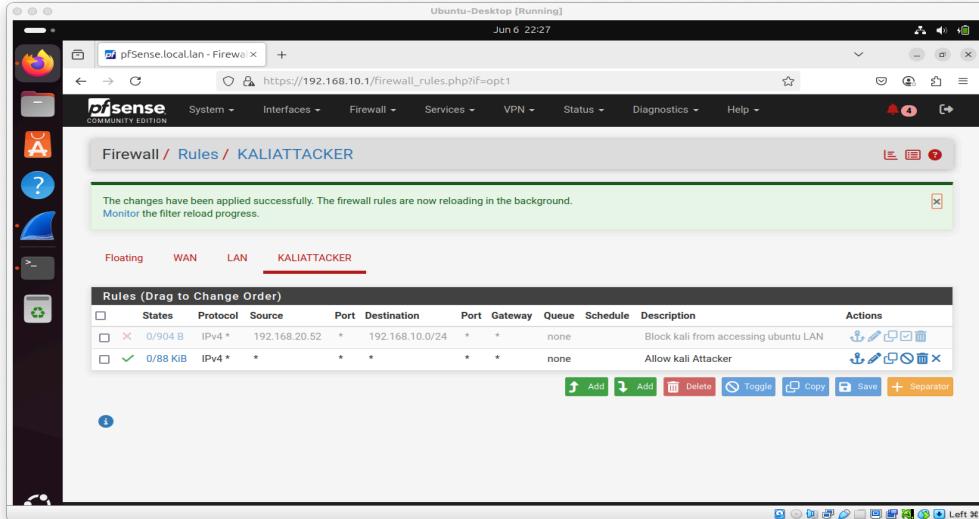
Buttons at the bottom include Add, Delete, Toggle, Copy, Save, and Separator.

It was observed that wireshark didn't capture any traffic(HTTP & TCP) when kali was blocked on pfSense, nmap scans reported host to be down. No access at all until it was disabled and we used only UFW rules then traffic could be captured



So basically when the firewall is activated on pfSense, traffic does not reach the devices on the LAN so Wireshark does not catch it when running Wireshark on any of the Ubuntus. If UFW rules are set, Wireshark will capture its traffic on the LAN devices. Also, PfSense log will show it and Wireshark on Kali too will show it and Kali will show it because it is Kali sending the requests. So to see Wireshark traffic, let us see how UFW

works, so we disabled pfSense



UFW on Ubuntu server

```
To          Action    From
Anywhere   ALLOW IN  --
Anywhere   ALLOW IN  192.168.10.0/24
Anywhere   ALLOW IN  192.168.20.0/24
22/tcp    ALLOW IN  Anywhere
80/tcp    DENY IN   Anywhere
82/tcp (v6) ALLOW IN  Anywhere (v6)
80/tcp (v6) DENY IN   Anywhere (v6)

officeuser@officeserver:~$ sudo ufw deny from 192.168.20.0
sudo: unable to resolve host officeserver.local: Name or service not known
Rule added
officeuser@officeserver:~$ sudo ufw status numbered
sudo: unable to resolve host officeserver.local: Name or service not known
Status: active

To          Action    From
[ 1] anywhere  ALLOW IN  192.168.10.0/24
[ 2] anywhere  ALLOW IN  192.168.20.0/24
[ 3] 22/tcp    ALLOW IN  Anywhere
[ 4] 80/tcp    DENY IN   Anywhere
[ 5] anywhere  DENY IN   192.168.20.0
[ 6] 82/tcp (v6) ALLOW IN  Anywhere (v6)
[ 7] 80/tcp (v6) DENY IN   Anywhere (v6)

officeuser@officeserver:~$ sudo ufw delete 2
sudo: unable to resolve host officeserver.local: Name or service not known
Deleting:
allow from 192.168.20.0/24
Proceed with operation (y/n)? y
Rule deleted
officeuser@officeserver:~$ sudo ufw status verbose
sudo: unable to resolve host officeserver.local: Name or service not known
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

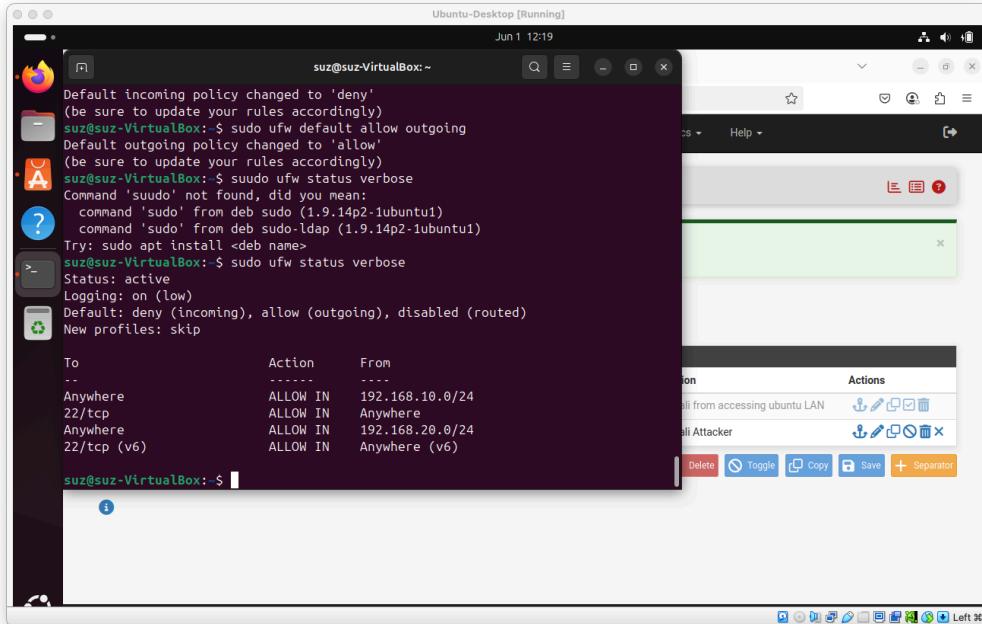
To          Action    From
Anywhere   ALLOW IN  192.168.10.0/24
22/tcp    ALLOW IN  Anywhere
80/tcp    DENY IN   Anywhere
Anywhere   DENY IN   192.168.20.0
82/tcp (v6) ALLOW IN  Anywhere (v6)
80/tcp (v6) DENY IN   Anywhere (v6)
```

10.2 Host-Based Firewall Configuration (UFW)

- First allowed in all traffic to check connectivity
- Both Ubuntu Server and Ubuntu Desktop enabled UFW with the following policies:
 - Default incoming: Deny
 - Allow incoming SSH (port 22) only from the LAN subnet (**192.168.10.0/24**)
 - Allow ICMP (ping) for diagnostics

- Deny all other inbound traffic
- This additional host-level firewall reduced the attack surface and reinforced network security.

UFW status output on Ubuntu Desktop showing active rules (sudo ufw status verbose) before deny rule



The screenshot shows a desktop environment with a terminal window and a ufw configuration interface.

Terminal Output:

```

Ubuntu-Desktop [Running]
Jun 1 12:19
suz@suz-VirtualBox:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(sure to update your rules accordingly)
suz@suz-VirtualBox:~$ sudo ufw status verbose
Command 'suudo' not found, did you mean:
  command 'sudo' from deb sudo (1.9.14p2-1ubuntu1)
  command 'sudo' from deb sudo-ldap (1.9.14p2-1ubuntu1)
Try: sudo apt install <deb name>
suz@suz-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To           Action      From
--           ----      --
Anywhere     ALLOW IN   192.168.10.0/24
22/tcp       ALLOW IN   Anywhere
Anywhere     ALLOW IN   192.168.20.0/24
22/tcp (v6)  ALLOW IN   Anywhere (v6)

suz@suz-VirtualBox:~$ 

```

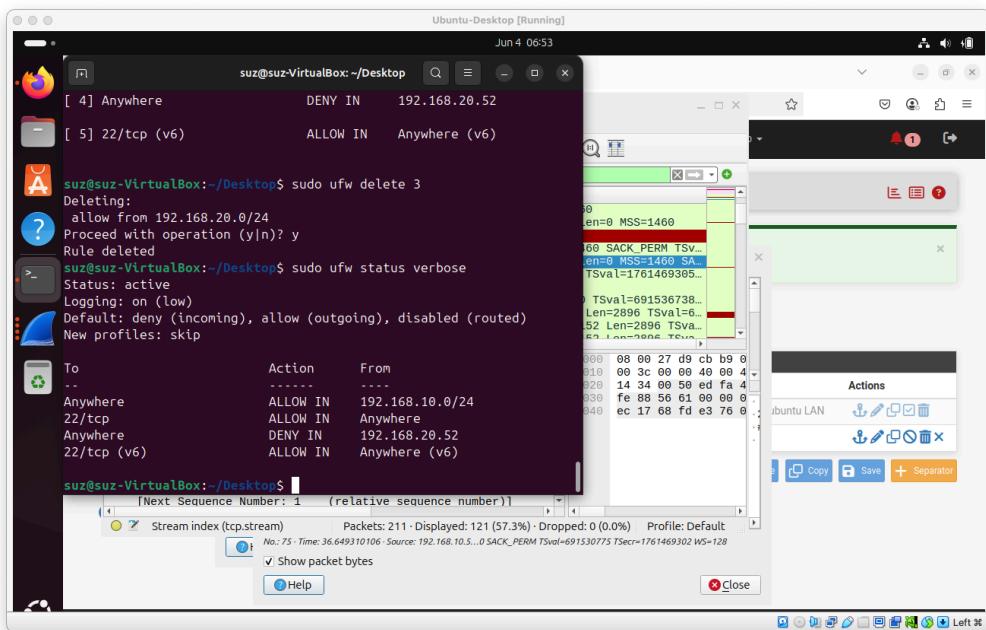
ufw Configuration GUI:

The ufw configuration interface shows two entries in the list:

Action	To	From	Comment	Actions
Allow	Anywhere	192.168.10.0/24	All from accessing ubuntu LAN	
Allow	Anywhere	192.168.20.0/24	All Attacker	

Buttons at the bottom of the GUI include: Delete, Toggle, Copy, Save, + Separator.

UFW status output on Ubuntu Desktop showing active rules (sudo ufw status verbose) after deny rule



UFW status output on Ubuntu server showing active rules (sudo ufw status verbose) before and after deny rule

```

Ubuntu server [Running]

To          Action      From
--          ----      --
Anywhere    ALLOW IN   192.168.10.0/24
Anywhere    ALLOW IN   192.168.20.0/24
22/tcp     ALLOW IN   Anywhere
80/tcp     DENY IN    Anywhere
22/tcp (v6) ALLOW IN   Anywhere (v6)
80/tcp (v6) DENY IN    Anywhere (v6)

officeuser@officeserver:~$ sudo ufw deny from 192.168.20.0/24
sudo: unable to resolve host officeserver.local: Name or service not known
Rule added
officeuser@officeserver:~$ sudo ufw status numbered
sudo: unable to resolve host officeserver.local: Name or service not known
Status: active
To          Action      From
--          ----      --
[ 1] anywhere ALLOW IN   192.168.10.0/24
[ 2] anywhere ALLOW IN   192.168.20.0/24
[ 3] 22/tcp     ALLOW IN   Anywhere
[ 4] 80/tcp     DENY IN    Anywhere
[ 5] anywhere  DENY IN    192.168.20.0/24
[ 6] 22/tcp (v6) ALLOW IN   Anywhere (v6)
[ 7] 80/tcp (v6) DENY IN    Anywhere (v6)

officeuser@officeserver:~$ sudo ufw delete 2
sudo: unable to resolve host officeserver.local: Name or service not known
Deleting...
allow from 192.168.20.0/24
Proceed with operation (y|n)? y
officeuser@officeserver:~$ sudo ufw status verbose
sudo: unable to resolve host officeserver.local: Name or service not known
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To          Action      From
--          ----      --
Anywhere    ALLOW IN   192.168.10.0/24
22/tcp     ALLOW IN   Anywhere
80/tcp     DENY IN    Anywhere
Anywhere    DENY IN    192.168.20.0/24
22/tcp (v6) ALLOW IN   Anywhere (v6)
80/tcp (v6) DENY IN    Anywhere (v6)

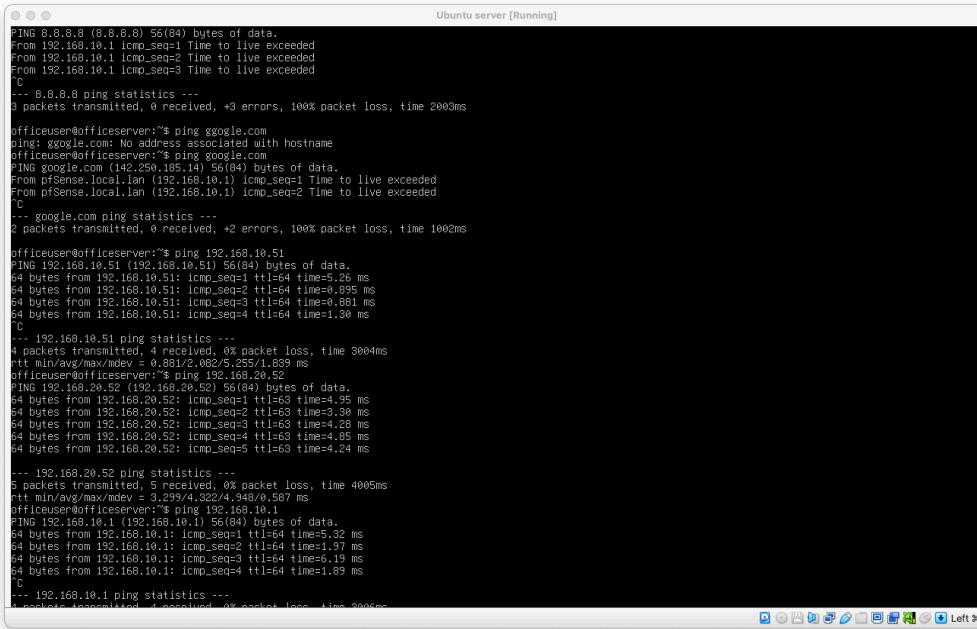
```

11. Testing and Validation

11.1 Connectivity Testing

- Ping tests from Ubuntu Server and Desktop to the pfSense LAN interface ([192.168.10.1](#)) succeeded, confirming LAN connectivity.
- Ping attempts from Kali Linux ([192.168.20.52](#)) to Ubuntu machines ([192.168.10.50](#), [192.168.10.51](#)) went through when the UFW allow rule was set but were later blocked

Successful Ping from Ubuntu Desktop to LAN machines and Kali



```
Ubuntu server [Running]
ping 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 192.168.10.1 icmp_seq=1 ttl=64 time to live exceeded
From 192.168.10.1 icmp_seq=2 ttl=64 time to live exceeded
From 192.168.10.1 icmp_seq=3 ttl=64 time to live exceeded
Tc
-- 8.8.8.8 ping statistics --
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2003ms

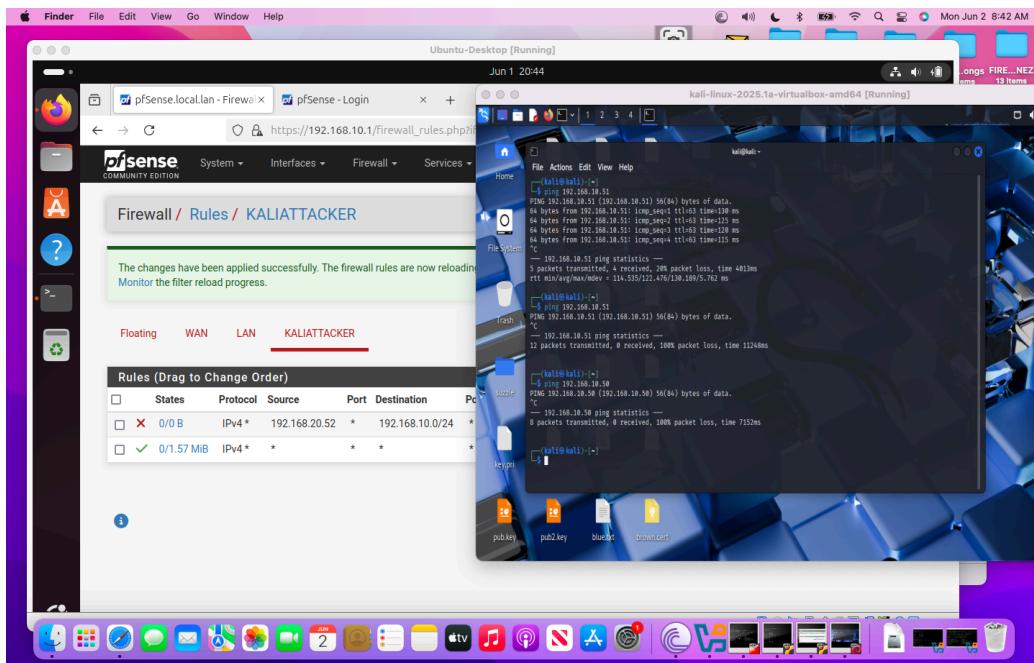
officeuser@officeserver:~$ ping google.com
PING google.com (72.237.14.14) 56(84) bytes of data.
From pfSense.local.lan (192.168.10.1) icmp_seq=1 Time to live exceeded
From pfSense.local.lan (192.168.10.1) icmp_seq=2 Time to live exceeded
Tc
-- google.com ping statistics --
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 1002ms

officeuser@officeserver:~$ ping 192.168.10.51
PING 192.168.10.51 (192.168.10.51) 56(84) bytes of data.
64 bytes from 192.168.10.51: icmp_seq=1 ttl=64 time=5.26 ms
64 bytes from 192.168.10.51: icmp_seq=2 ttl=64 time=0.895 ms
64 bytes from 192.168.10.51: icmp_seq=3 ttl=64 time=0.861 ms
64 bytes from 192.168.10.51: icmp_seq=4 ttl=64 time=1.30 ms
Tc
-- 192.168.10.51 ping statistics --
4 packets transmitted, 0 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.881/2.082/5.255/1.839 ms
officeuser@officeserver:~$ ping 192.168.20.52
PING 192.168.20.52 (192.168.20.52) 56(84) bytes of data.
64 bytes from 192.168.20.52: icmp_seq=1 ttl=63 time=4.95 ms
64 bytes from 192.168.20.52: icmp_seq=2 ttl=63 time=3.30 ms
64 bytes from 192.168.20.52: icmp_seq=3 ttl=63 time=4.28 ms
64 bytes from 192.168.20.52: icmp_seq=4 ttl=63 time=4.85 ms
64 bytes from 192.168.20.52: icmp_seq=5 ttl=63 time=4.24 ms
Tc
-- 192.168.20.52 ping statistics --
5 packets transmitted, 0 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 3.299/4.322/4.940/0.587 ms
officeuser@officeserver:~$ ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=1 ttl=64 time=5.32 ms
64 bytes from 192.168.10.1: icmp_seq=2 ttl=64 time=1.97 ms
64 bytes from 192.168.10.1: icmp_seq=3 ttl=64 time=6.19 ms
64 bytes from 192.168.10.1: icmp_seq=4 ttl=64 time=1.89 ms
Tc
-- 192.168.10.1 ping statistics --
4 packets transmitted, 0 received, 0% packet loss, time 2006ms
```

Ubuntu server unable to ping KALI ATTACKER 192.168.20.52 after block rule added

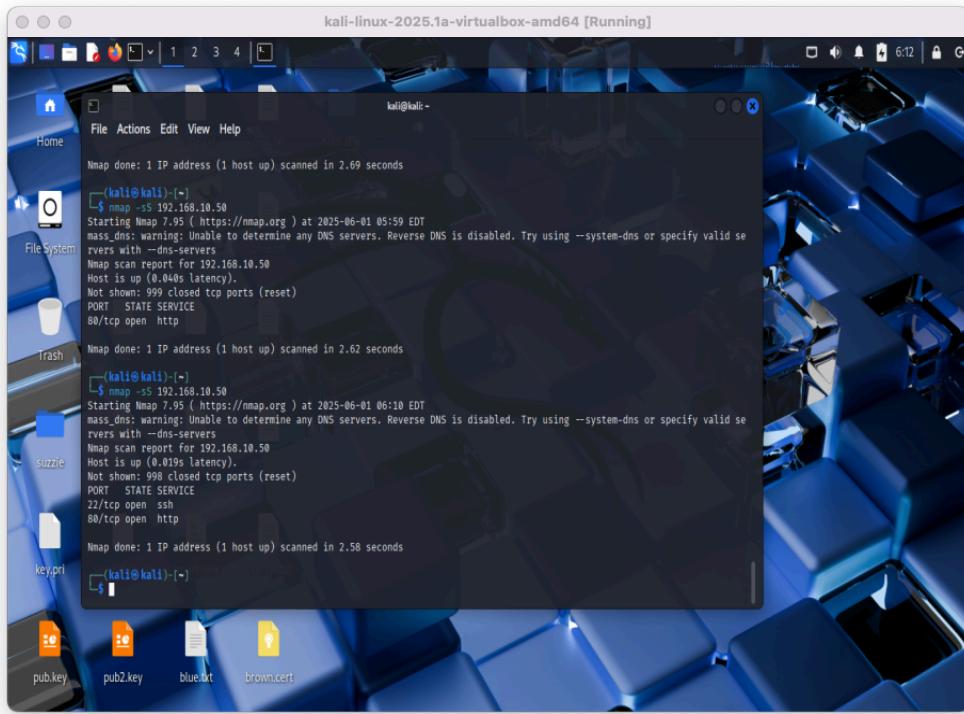
```
valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:4f:4f:ff brd ff:ff:ff:ff:ff:ff
      inet 192.168.20.52/24 brd 192.168.20.255 scope global dynamic enp0s3
        valid_lft 7019sec preferred_lft 7019sec
      inet6 fe80::a00c:29ff:fe4f:4f%enp0s3 brd ff:ff:ff:ff:ff:ff scope link
        valid_lft forever preferred_lft forever
oficeuser@officeserver:~$ ping 192.168.20.52
PING 192.168.20.52 (192.168.20.52) 56(84) bytes of data.
^C
--- 192.168.20.52 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 2561ms
```

Kali terminal showing ping attempts to Ubuntu server 192.168.10.50 and ubuntu desktop 192.168.10.51 before and after block rule added again



11.2 Network Scanning with Nmap

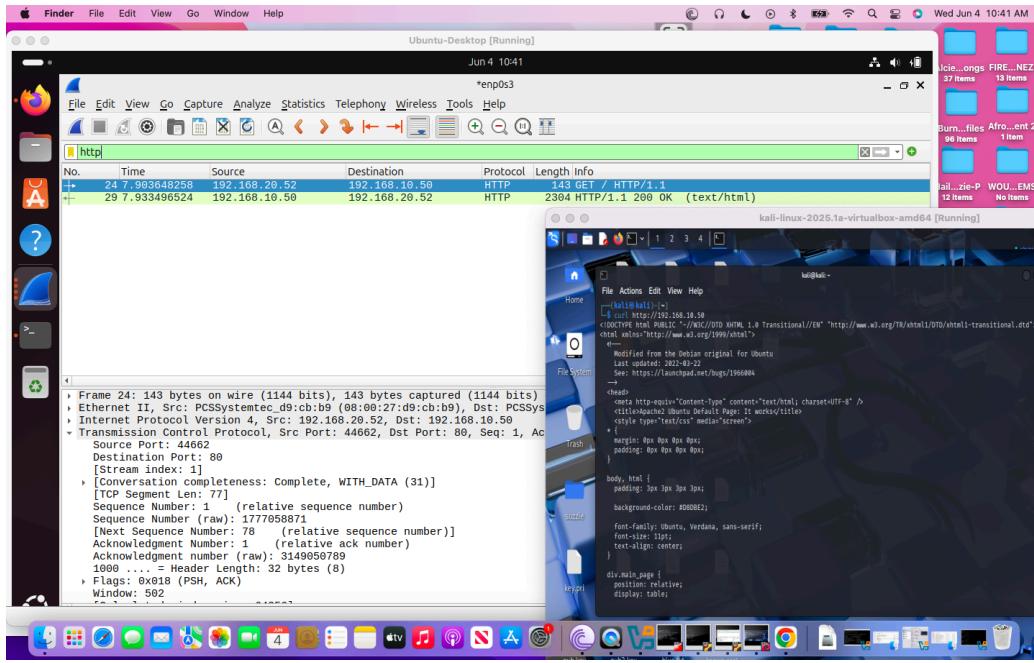
- Kali Linux executed various scans on the Ubuntu machines:
- No open ports were discovered on Ubuntu machines due to effective firewall policies at both pfSense and host level.
- KALI ATTACKER Nmap scan after block rule added
- Had to allow KALI ATTACKER access the LAN machines, also made port 80 and 22 vulnerable on both LAN machines for port scanning using Nmap and network traffic monitoring using Wireshark in the image below:



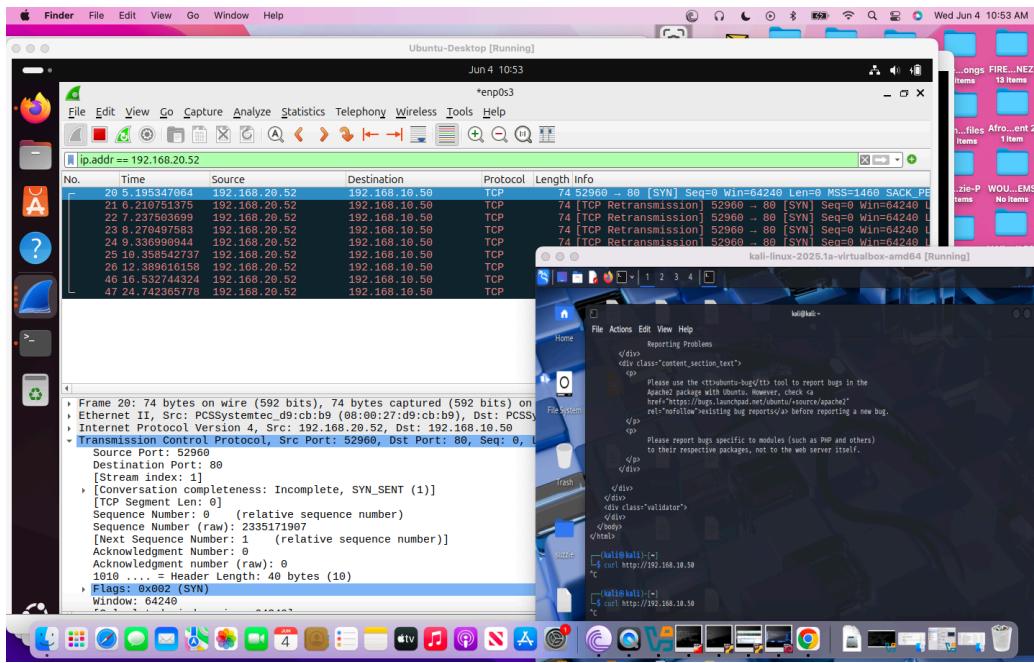
11.3 Traffic Capture and Analysis with Wireshark

- Wireshark running on the Ubuntu Server captured the scan attempts from Kali Linux.
- The captures confirmed that HTTP and TCP connection attempts (SYN packets) from the attacker IP (192.168.20.52) were at first allowed and connected until they were dropped indicating retransmission
- It is important to note that UFW allow rule apply open ports and wireshark captures it as SYN-ACK which means ubuntu server is listening and accepts connection, UFW deny rule apply closed ports and wireshark captures it as RST-ACK, which means ubuntu server actively rejects connection. UFW default deny rule makes ports filtered, so on wireshark it is reported as SYN and no reply which means the server silently dropped packets and they are unresponded to.

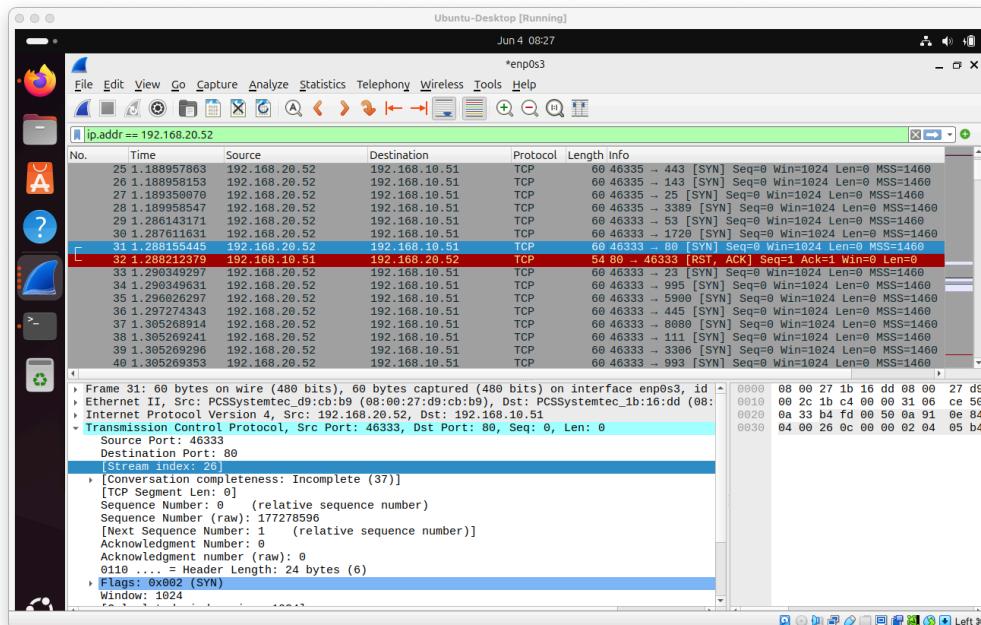
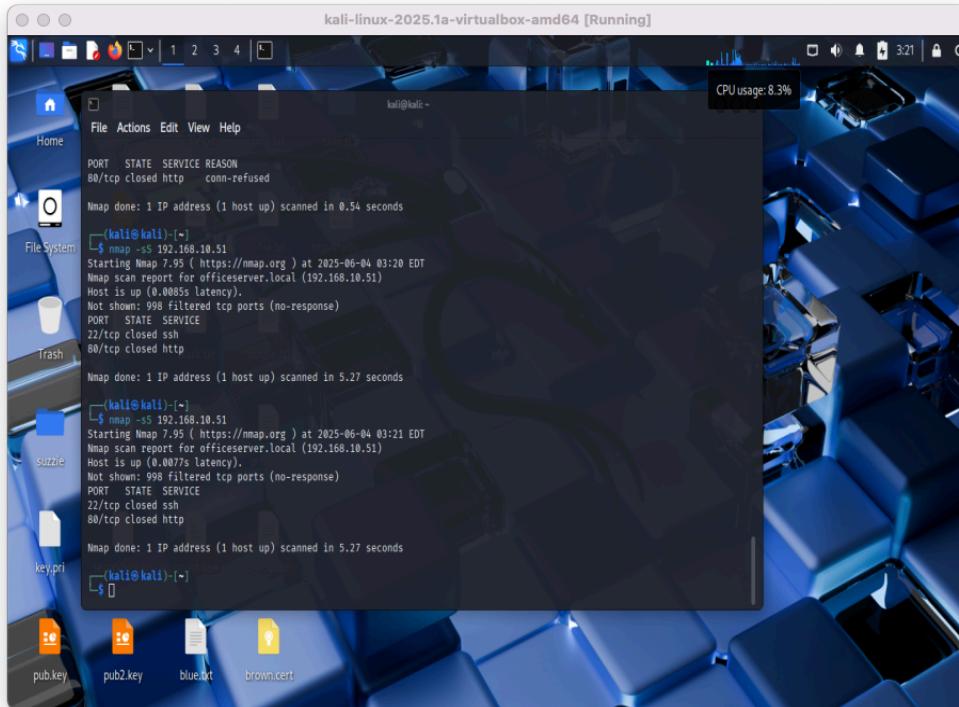
Kali attacker HTTP scan to Ubuntu server before UFW block rule



Wireshark HTTP traffic capture showing SYN request blocked, indicating retransmission after UFW deny rule



Kali Nmap scan to ubuntu desktop, wireshark capture on Ubuntu desktop. SSH socket disabled, after which nmap scanned traffic from kali to ubuntu desktop showed port no connection to port 80 (SYN) and traffic blocked on port 22 (RST-ACK) because of hardening even though ping was allowed



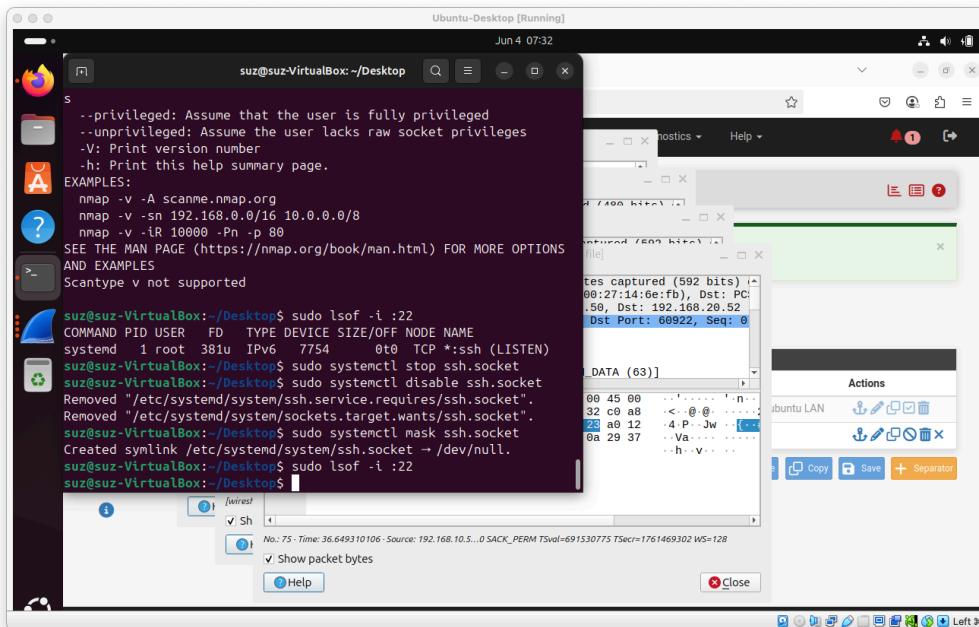
12. Simulated Attack Scenario

12.1 Attack Description

- Kali Linux simulated an internal reconnaissance attack by scanning and pinging Ubuntu hosts.
- This test emulates a common adversary tactic for network mapping before exploitation.

12.2 Defense Outcomes

- Due to network segmentation and firewall rules, Kali's scans revealed no services or open ports on the Ubuntu hosts.
- Ping requests were dropped, confirming strict network isolation.
- Host-based UFW rules added an additional barrier, ensuring no unauthorized access even if firewall rules were bypassed.
- SSH not deactivated on ubuntu desktop and server fully because systemd is still listening and ssh.socket is still active until disabled
- All ports were closed after simulated attack



```

Ubuntu server [Running]
officeuser@officeserver:~$ sudo ufw status verbose
sudo: unable to resolve host officeserver.local: Name or service not known
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To           Action      From
Anywhere     ALLOW IN   192.168.10.0/24
22/tcp       ALLOW IN   Anywhere
80/tcp       DENY IN    Anywhere
Anywhere     DENY IN    192.168.20.0
22/tcp (v6)  ALLOW IN   Anywhere (v6)
80/tcp (v6)  DENY IN    Anywhere (v6)

officeuser@officeserver:~$ sudo systemctl disable ssh
sudo: unable to resolve host officeserver.local: Name or service not known
[sudo] password for officeuser:
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install disable ssh
Disabling ssh.service , but its triggering units are still active:
ssh.socket
officeuser@officeserver:~$ sudo systemctl status ssh
sudo: unable to resolve host officeserver.local: Name or service not known
* ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/usr/lib/systemd/system/ssh.service; disabled; preset: enabled)
    Active: inactive (dead)
   TriggeredBy: • ssh.socket
     Docs: man:sshd(8)
          man:sshd_config(5)
officeuser@officeserver:~$ sudo lsof -i :22
sudo: unable to resolve host officeserver.local: Name or service not known
[sudo] password for officeuser:
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME
systemd 1 root 18u IPv6 7408      0t0  TCP >*:ssh (LISTEN)
officeuser@officeserver:~$ sudo systemctl stop ssh.socket
sudo: unable to resolve host officeserver.local: Name or service not known
[sudo] password for officeuser:
Removed "/etc/systemd/system/ssh.service.requires/ssh.socket".
Removed "/etc/systemd/system/sockets.target.wants/ssh.socket".
officeuser@officeserver:~$ sudo systemctl mask ssh.socket
sudo: unable to resolve host officeserver.local: Name or service not known
Created symlink /etc/systemd/system/ssh.socket → /dev/null.
officeuser@officeserver:~$ sudo lsof -i :22
sudo: unable to resolve host officeserver.local: Name or service not known
officeuser@officeserver:~$ sudo lsof -i :22

```

Logs from pfSense showing blocked traffic entries

Jun 2 08:49						
Jun 1 10:05:37	WAN	veraunt deny rule IPv4 (1000000103)	1 ↗ 54.217.10.153:443	1 ↖ 10.0.2.15:43398	TCP:FA	
Jun 1 10:05:50	WAN	Default deny rule IPv4 (1000000103)	1 ↗ 54.217.10.153:443	1 ↖ 10.0.2.15:43398	TCP:FA	
Jun 1 10:06:02	WAN	Default deny rule IPv4 (1000000103)	1 ↗ 54.217.10.153:443	1 ↖ 10.0.2.15:43398	TCP:RA	
Jun 1 10:46:30	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:44786	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 11:00:13	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:56725	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 11:17:12	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:55439	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 11:30:03	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:61175	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 11:34:38	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:48894	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 11:37:43	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:63656	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 11:48:05	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:55853	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 12:00:53	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:44905	1 ↖ 192.168.10.51:80	TCP:A	
Jun 1 12:01:37	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:51759	1 ↖ 192.168.10.51:80	TCP:A	
Jun 1 12:03:29	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:58753	1 ↖ 192.168.10.51:80	TCP:A	
Jun 1 12:22:34	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:58467	1 ↖ 192.168.10.51:80	TCP:A	
Jun 1 12:49:24	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:47372	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 12:50:03	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:57653	1 ↖ 192.168.10.51:80	TCP:A	
Jun 1 13:53:36	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:36434	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 13:54:43	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:37963	1 ↖ 192.168.10.50:80	TCP:A	
Jun 1 15:51:16	KALIATTACKER	Default deny rule IPv4 (1000000103)	1 ↗ 192.168.20.52:43341	1 ↖ 192.168.10.51:80	TCP:A	

13. Results and Discussion

The layered security approach demonstrated effective prevention of unauthorized access from an attacker on a separate subnet. Segmentation via pfSense and host hardening with UFW successfully blocked reconnaissance and lateral movement attempts.

Wireshark analysis provided real-time verification that traffic from Kali Linux was denied access and was unable to interact with Ubuntu machines. This approach validates the importance of using multiple security layers to mitigate threats in small network environments.

14. Challenges and Lessons Learned

- Properly configuring multiple network interfaces on pfSense was essential for effective segmentation.
- Using DHCP simplified network management and IP assignments but required monitoring to ensure consistent addressing.
- Coordinating firewall rules between pfSense and UFW to avoid conflicts was critical.
- Simulating realistic threats like port scanning and HTTP probing from kali needed controlled testing to avoid noise or false assumption
- Practical testing using real attack tools highlighted the importance of layered security via pfSense (network-level) and UFW (host-level)
- Understanding VirtualBox networking modes (NAT, internal, bridged) was fundamental to network design, planning IP addressing and mapping traffic path is important to avoid connectivity issues.

15. Next Steps and Recommendations

- Explore advanced intrusion detection/prevention systems (IDS/IPS) for enhanced threat detection.
- Implement VPN services for secure remote access to internal network resources.

- Expand the lab to include Windows machines and simulate phishing or malware attacks for broader testing.
- Automate firewall monitoring and logging to improve incident response efficiency.

Conclusion, References, Appendices, and Final Notes

16. Conclusion

This capstone project aimed to design, implement, and secure a small network environment using VirtualBox virtual machines, the pfSense firewall/router, and Ubuntu Linux servers, with Kali Linux serving as an attacker VM on a segregated network.

By assigning three network interfaces to pfSense (em0 as NAT WAN, em1 as LAN intnet, and em2 as Kali Attacker network), the project demonstrated effective network segmentation, isolation, and controlled routing.

Dynamic IP address assignment via pfSense DHCP simplified network management for Ubuntu Server and Desktop while maintaining consistent connectivity within the internal LAN.

Through comprehensive firewall configurations on pfSense and host-based firewalls (UFW) on Ubuntu systems, all unauthorized access attempts and network reconnaissance efforts by the Kali attacker VM were successfully blocked. Testing with ping, nmap scanning, and Wireshark packet analysis validated the implemented security controls.

The layered defense approach implemented is a best practice in network security, providing multiple barriers against potential intrusions, reconnaissance, and lateral movement.

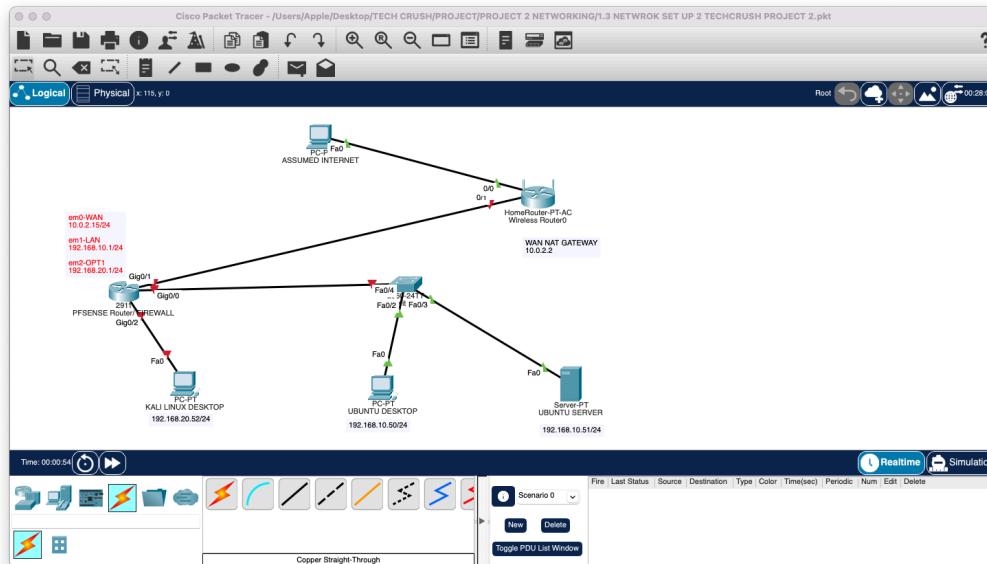
This project highlights the value of combining network segmentation, firewall hardening, and continuous monitoring to secure small and medium-sized networks from internal and external threats.

17. References

- TechCrush Cybersecurity Cohort , March, 2025- Network security, Network protocols, Wireshark, Operating systems by Jeremiah Olubunmi and Faith Olagoke
- pfSense Documentation
Netgate. (n.d.). pfSense Documentation. Retrieved from <https://docs.netgate.com/pfsense/en/latest/>
- Ubuntu Server Administration
Ubuntu Community. (n.d.). Ubuntu Server Guide. Retrieved from <https://help.ubuntu.com/lts/serverguide/>
- Ubuntu UFW Firewall
Ubuntu Community Help Wiki. (n.d.). UFW - Uncomplicated Firewall. Retrieved from <https://help.ubuntu.com/community/UFW>
- Nmap Network Scanning
Lyon, G. F. (2009). Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. Insecure.Com LLC.
- Wireshark User Guide
Combs, G. (n.d.). Wireshark User's Guide. Retrieved from https://www.wireshark.org/docs/wsug_html_chunked/
- VirtualBox Networking
Oracle. (n.d.). VirtualBox User Manual. Retrieved from <https://www.virtualbox.org/manual/ch06.html>
- General Cybersecurity Principles
Stallings, W. (2018). Network Security Essentials: Applications and Standards (6th ed.). Pearson.

18. Appendices

Configuration Files and Command Outputs



The screenshot shows the 'Edit Gateway Group Entry' configuration page in a web browser. The 'Group Name' is set to 'WAN_Default'. Under 'Gateway Priority', there are three entries:

Gateway	Tier	Virtual IP	Description
WAN_DHCP	Tier 1	Interface Address	Interface WAN_DHCP Gateway
WANGW	Never	Interface Address	MANUAL_WAN_GW
LAN_DHCP	Never	Interface Address	Interface LAN_DHCP Gateway

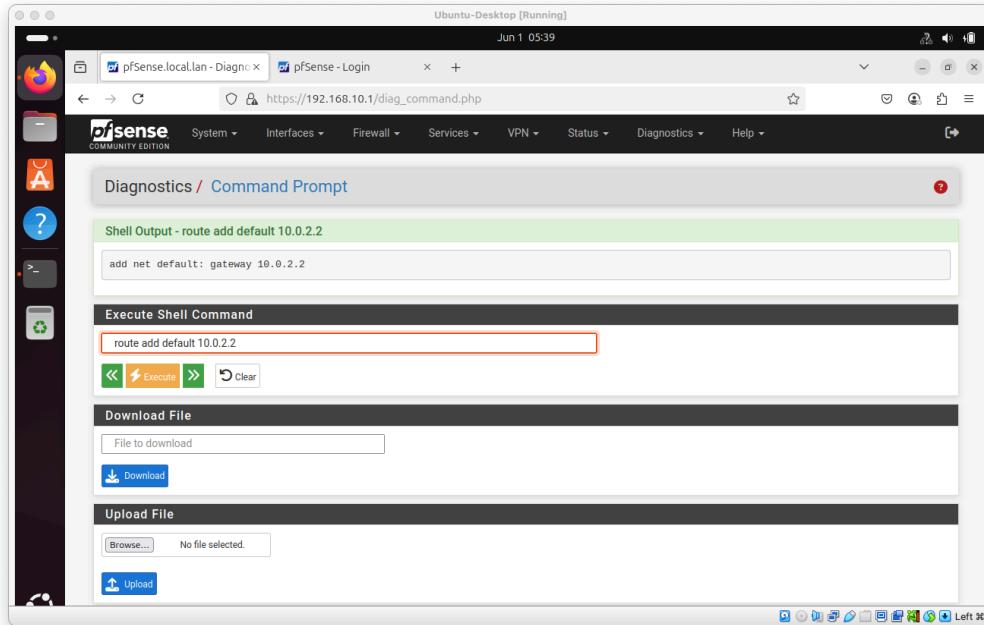
The 'Link Priority' section notes: "The priority selected here defines in what order failover and balancing of links will be done. Multiple links of the same priority will balance connections until all links in the priority will be exhausted. If all links in a priority level are exhausted then the next available link(s) in the next priority level will be used."

The 'Virtual IP' section specifies: "The virtual IP field selects which (virtual) IP should be used when this group applies to a local Dynamic DNS, IPSec or OpenVPN endpoint."

The 'Keep Failover States' section includes: "Use global behavior (default)" and "Kill states on gateway recovery: states created for lower-priority gateways by policy-routing firewall rules will be killed." It also notes: "Note: changes to the gateway priority configuration only affect states created after settings are applied - states which already exist may need to be manually killed."

The 'Trigger Level' is set to 'Member Down'.

Static gateway added at first on pfSense via system/routing/gateway, checked the box for WAN DHCP and applied changes then it was permanent.



The screenshot shows the pfSense System / Routing / Gateways configuration page. The 'Gateways' tab is selected, displaying a table of configured gateways:

Name	Default	Interface	Gateway	Monitor IP	Description	Actions
WAN_DHCP	✓	WAN	10.0.2.2	10.0.2.2	Interface WAN_DHCP Gateway	
WAN_DHCP6	✓	WAN	dynamic	dynamic	Interface WAN_DHCP6 Gateway	
WANGW		WAN	192.168.1.1	192.168.1.1	MANUAL_WAN_GW	
LAN_DHCP6	✓	LAN	dynamic	dynamic	Interface LAN_DHCP6 Gateway	
LAN_DHCP	✓	LAN	dynamic	dynamic	Interface LAN_DHCP Gateway	

Below the table, there are sections for 'Default gateway IPv4' (set to 'Automatic') and 'Default gateway IPv6' (set to 'Automatic').

```
Pfsense router [Running]
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address    11) Restart GUI
3) Reset admin account and password 12) PHP shell + pfSense tools
4) Reset to factory defaults     13) Update from console
5) Reboot system                14) Enable Secure Shell (sshd)
6) Halt system                  15) Restore recent configuration
7) Ping host                     16) Restart PHP-FPM
8) Shell

Enter an option: 7

Enter a host name or IP address: 192.168.20.52

PING 192.168.20.52 (192.168.20.52): 56 data bytes
64 bytes from 192.168.20.52: icmp_seq=0 ttl=64 time=1.826 ms
64 bytes from 192.168.20.52: icmp_seq=1 ttl=64 time=3.766 ms
64 bytes from 192.168.20.52: icmp_seq=2 ttl=64 time=2.577 ms

--- 192.168.20.52 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 1.826/2.723/3.766/0.799 ms

Press ENTER to continue.
```

