CODEALPHA TASK 1

PROJECT TITLE:

Intrusion Detection and Response using Suricata IDS on pfSense Firewall

NAME: Onyinye Nwosu

PROGRAM: Cybersecurity Internship Project

ORGANISATION: CodeAlpha

SUBMISSION DATE: August,2025

Table Of Content

1. INTRODUCTION

This task focused on designing, deploying, and testing a Network Intrusion Detection System (NIDS) using Suricata IDS within a virtual lab environment, secured by pfSense firewall/router. The objective was to monitor and detect malicious activities from a simulated attacker (Kali Linux) targeting a small internal network with Ubuntu Desktop and Ubuntu Server machines.

The project mirrors real-world security operations in small enterprise setups, with emphasis on detection, alerting, and response mechanisms. It provides hands-on experience with tools and technologies often used by SOC analysts and network defenders.

2. PROJECT OVERVIEW

The main goal of this project was to deploy a functional intrusion detection and response system by integrating Suricata into a virtualized pfSense firewall, configure it correctly to detect network-based threats from a known attack source, and respond appropriately.

Suricata was chosen for its real-time traffic inspection, rule-based alerting, and open-source flexibility. The project simulates an enterprise LAN setup with virtual machines, where unauthorized access attempts from an attacker system (Kali Linux) must be detected and mitigated.

This task aligns with real-world objectives of a SOC team, including:

1. Detecting malicious or suspicious behavior
2. Alert generation based on configured rules
3. Implementing response actions such as blocking offenders
4. Evaluating the effectiveness of network defense tools

Key Objectives Achieved:

Objective Description

- Network Setup Properly segmented LAN and attacker networks through internal adapters
- IDS Configuration Installed and configured Suricata on pfSense
- Rule Activation Enabled ET rules and custom rule categories
- Alert Validation Successfully triggered alerts by launching scans/attacks from Kali
- Incident Response Implemented blocking and inline IPS actions through Suricata

Network Threat Simulated:
Tool Used for Attack: Kali Linux
Attack Methods: Ping probe, Nmap scan
Targeted VMs: Ubuntu Desktop and Ubuntu Server
Response: Alert generation + IPS mode drop actions

Project Outcome:
The final setup allowed us to detect and respond to unauthorized network scans from the attacker VM, proving that Suricata was effectively inspecting traffic, logging alerts, and taking action on defined rules.

3. NETWORK SETUP & VIRTUAL INFRASTRUCTURE

Network Topology & IP Addressing

The virtual network was designed using VirtualBox, and consisted of 4 main virtual machines:

VM	Role	Network Adapter	IP Address
pfSense	Firewall Router + IDS/IPS	Adapter 1: NAT (WAN) Adapter 2: Internal Network(intnet) Adapter 3: Internal Network (KaliAttacker)	WAN: 10.0.2.15/24 (DHCP from NAT) LAN: 192.168.10.1/24 OPT1 (for Kali): 192.168.20.1/24
Ubuntu Desktop	LAN Workstation	Internal Network (intnet)	192.168.10.51/24
Ubuntu Server	LAN Server	Internal Network (intnet)	192.168.10.56/24 |
Kali Linux	Attacker Machine	Internal Network (Kali Attacker)	192.168.20.52/24

4. VIRTUALBOX VM NETWORK ADAPTER CONFIGURATION

VM Adapter Network Mode Assigned To
pfSense:

- Adapter 1 NAT Internet (default)
- Adapter 2 Internal Network: intnet - LAN VMs (Ubuntu Desktop & Server)
- Adapter 3 Internal Network: Kali Attacker- Kali Linux

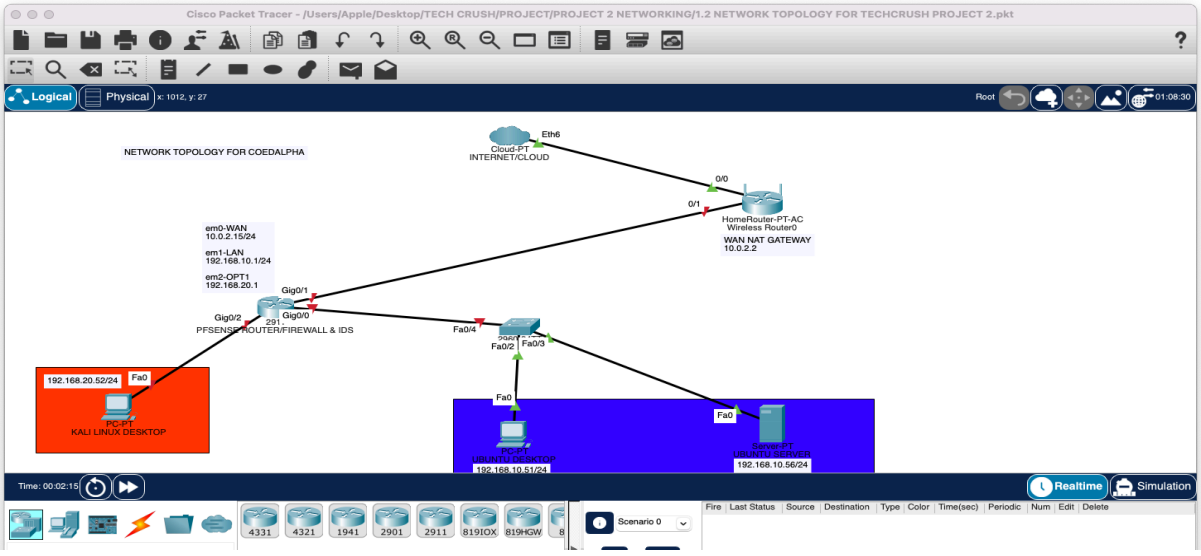
Ubuntu Desktop/Server:
Adapter 1 Internal Network (intnet), To communicate with pfSense
Kali Linux:
Adapter 1 Internal Network: Kali Attacker, To attack through pfSense

IP Address Configuration

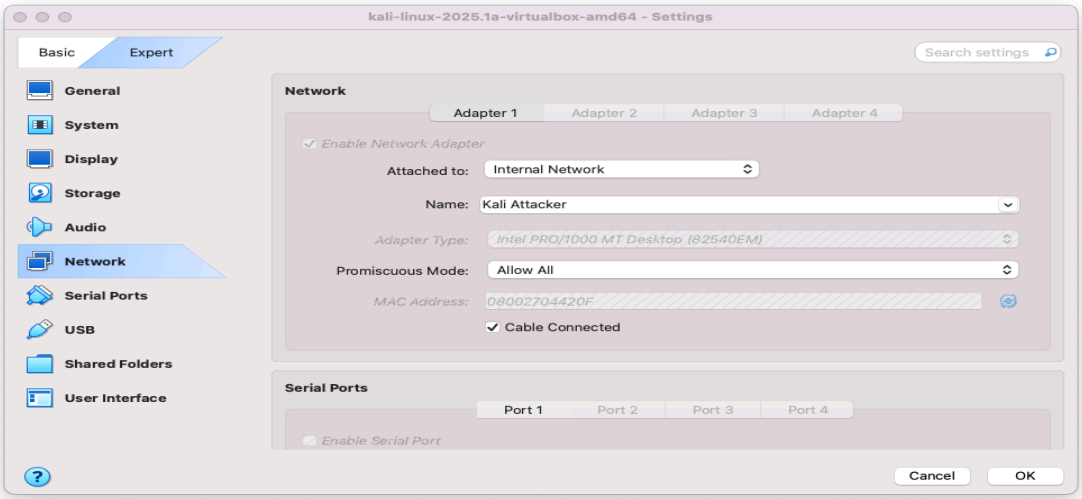
pfSense Interfaces
WAN: DHCP (gets IP like 10.0.2.15)
LAN: Static IP 192.168.10.1/24
OPT1 (Renamed to KaliAttacker): Static IP 192.168.20.1/24

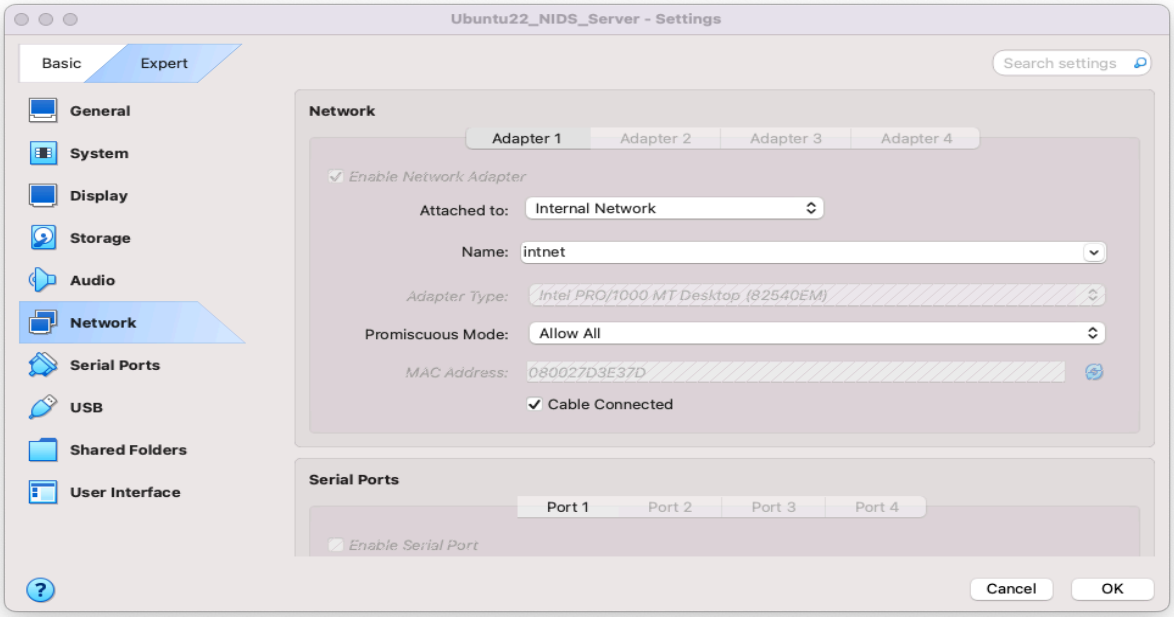
LAN Devices:
Ubuntu Desktop: DHCP from pfSense LAN: 192.168.10.50
Ubuntu Server: DHCP: 192.168.10.51
Attacker Device (Kali Linux)
Static IP: 192.168.20.52/24
Gateway: 192.168.20.1
DNS: 8.8.8.8 or use pfSense gateway

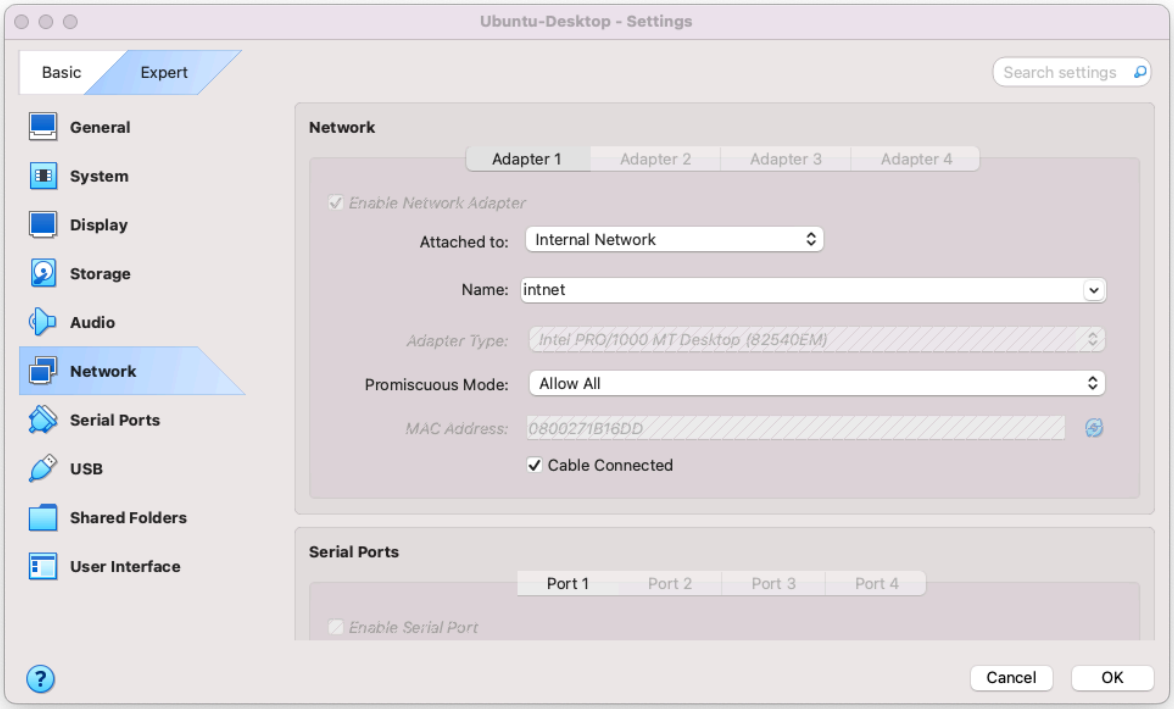
Network Topology

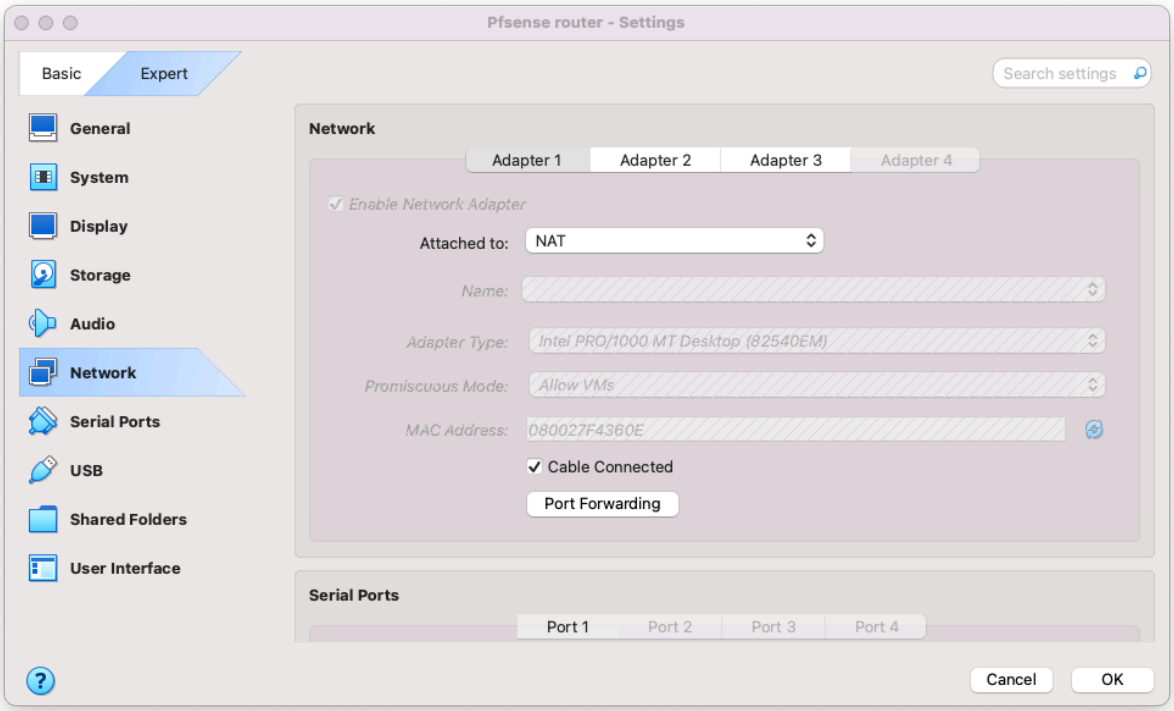


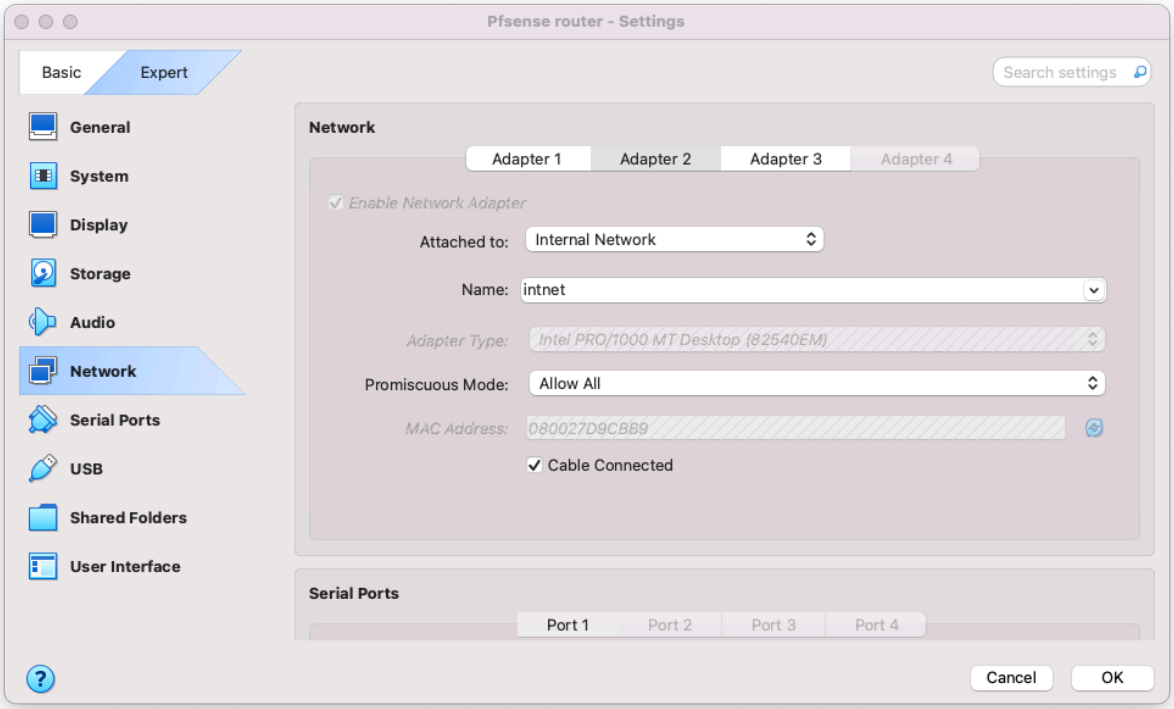
VM Settings showing network adapter types

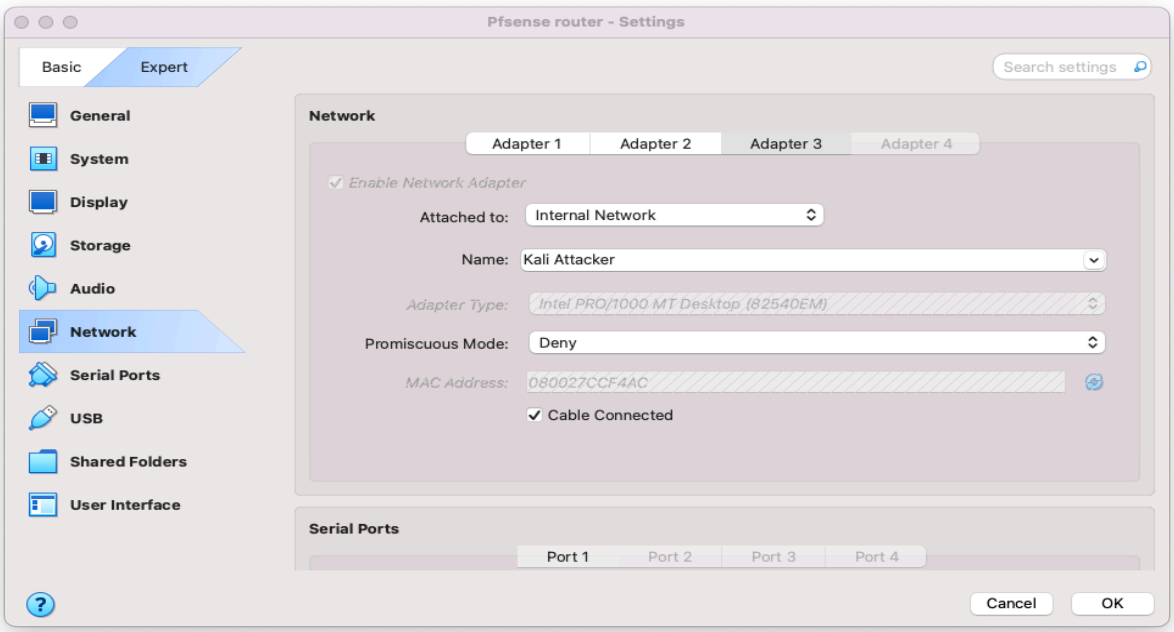




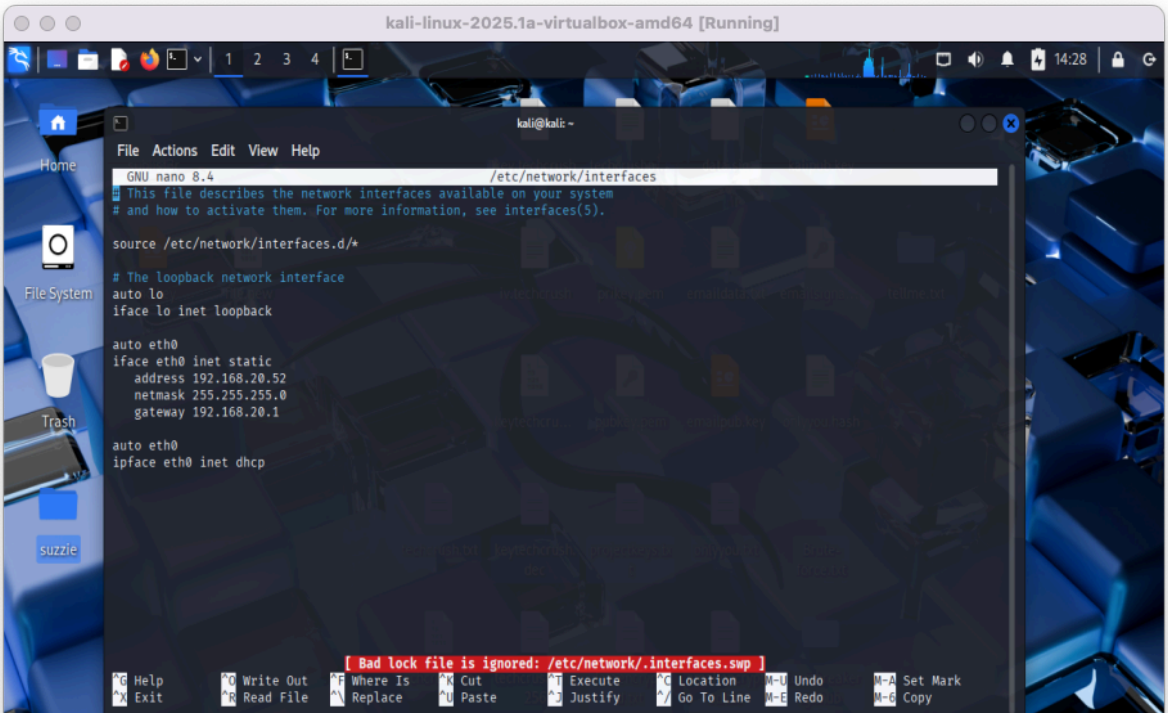

Ubuntu-Desktop - Settings
Basic
Expert
Search settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
User Interface
Network
Adapter 1
Adapter 2
Adapter 3
Adapter 4
Enable Network Adapter
Attached to:
Internal Network
Name:
intnet
Adapter Type:
Intel PRO/1000 MT Desktop (82540EM)
Promiscuous Mode:
Allow All
MAC Address:
0800271B16DD
Cable Connected
Serial Ports
Port 1
Port 2
Port 3
Port 4
Enable Serial Port
Cancel
OK

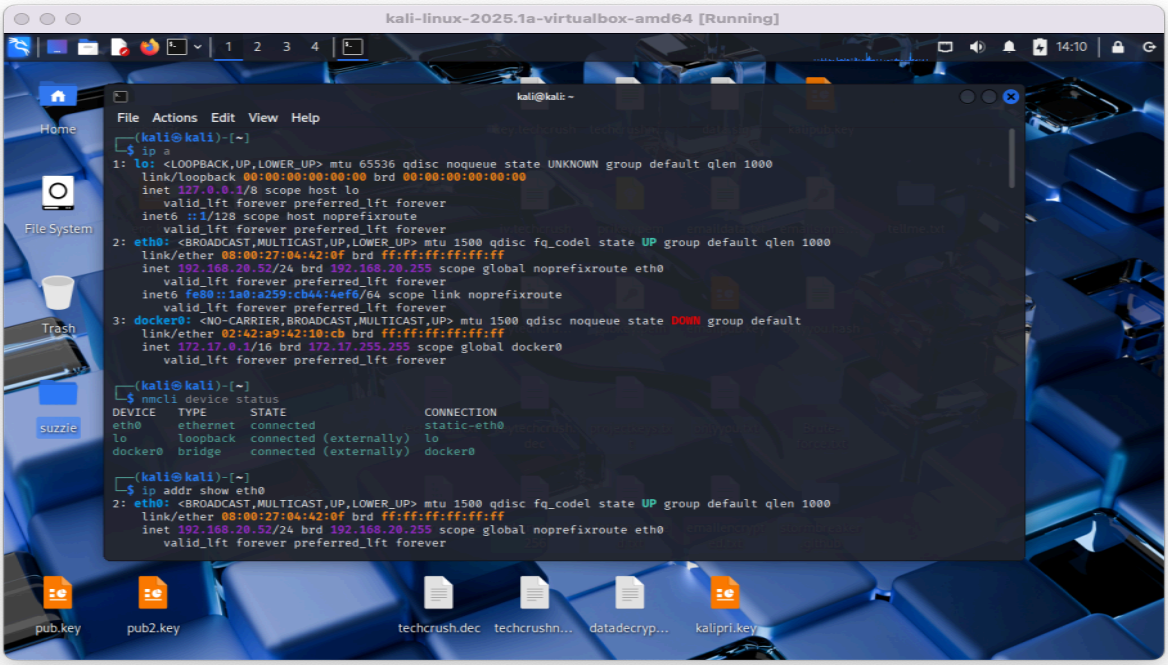

Pfsense router - Settings
Basic
Expert
Search settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
User Interface
Network
Adapter 1
Adapter 2
Adapter 3
Adapter 4
Enable Network Adapter
Attached to:
NAT
Name:
Adapter Type:
Intel PRO/1000 MT Desktop (82540EM)
Promiscuous Mode:
Allow VMs
MAC Address:
080027F4360E
Cable Connected
Port Forwarding
Serial Ports
Port 1
Port 2
Port 3
Port 4
Cancel
OK


Pfsense router - Settings
Basic
Expert
Search settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
User Interface
Network
Adapter 1
Adapter 2
Adapter 3
Adapter 4
Enable Network Adapter
Attached to:
Internal Network
Name:
intnet
Adapter Type:
Intel PRO/1000 MT Desktop (82540EM)
Promiscuous Mode:
Allow All
MAC Address:
080027D9CBB9
Cable Connected
Serial Ports
Port 1
Port 2
Port 3
Port 4
Cancel
OK

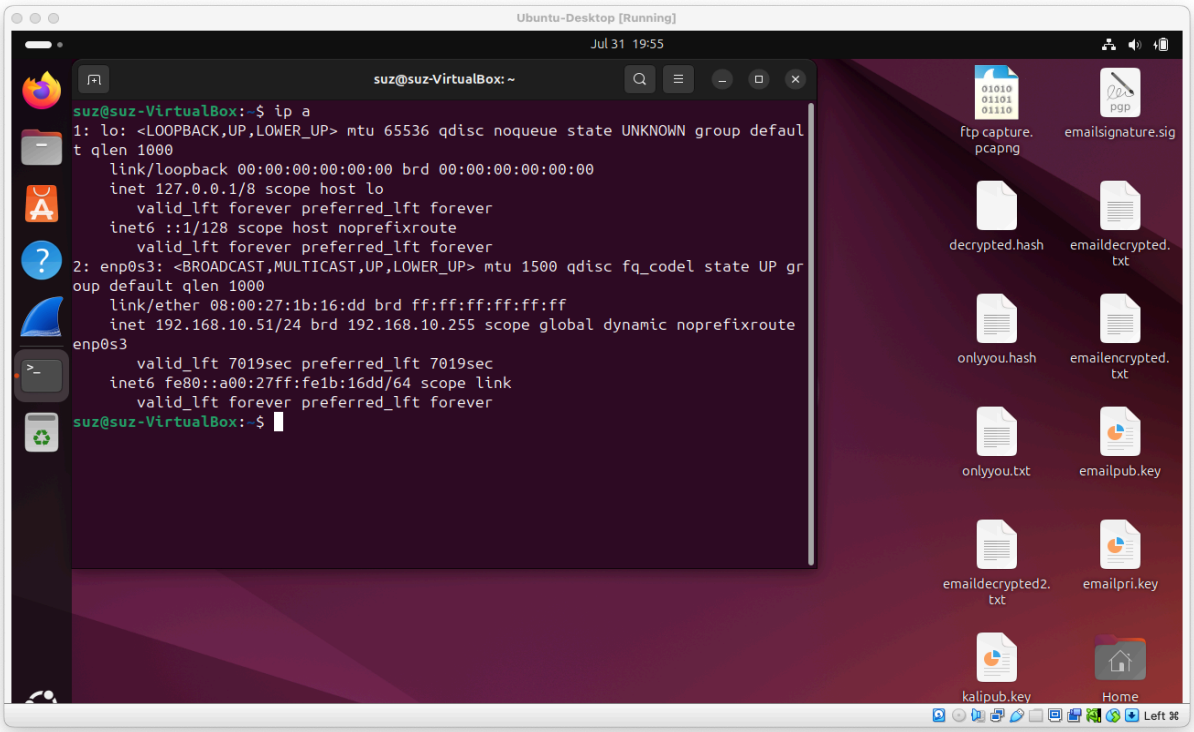

Pfsense router - Settings
Basic
Expert
Search settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
User Interface
Network
Adapter 1
Adapter 2
Adapter 3
Adapter 4
Enable Network Adapter
Attached to:
Internal Network
Name:
Kali Attacker
Adapter Type:
Intel PRO/1000 MT Desktop (82540EM)
Promiscuous Mode:
Deny
MAC Address:
080027CCF4AC
Cable Connected
Serial Ports
Port 1
Port 2
Port 3
Port 4
Cancel
OK

Network Manager & ip a output on each VM
Kali VM:

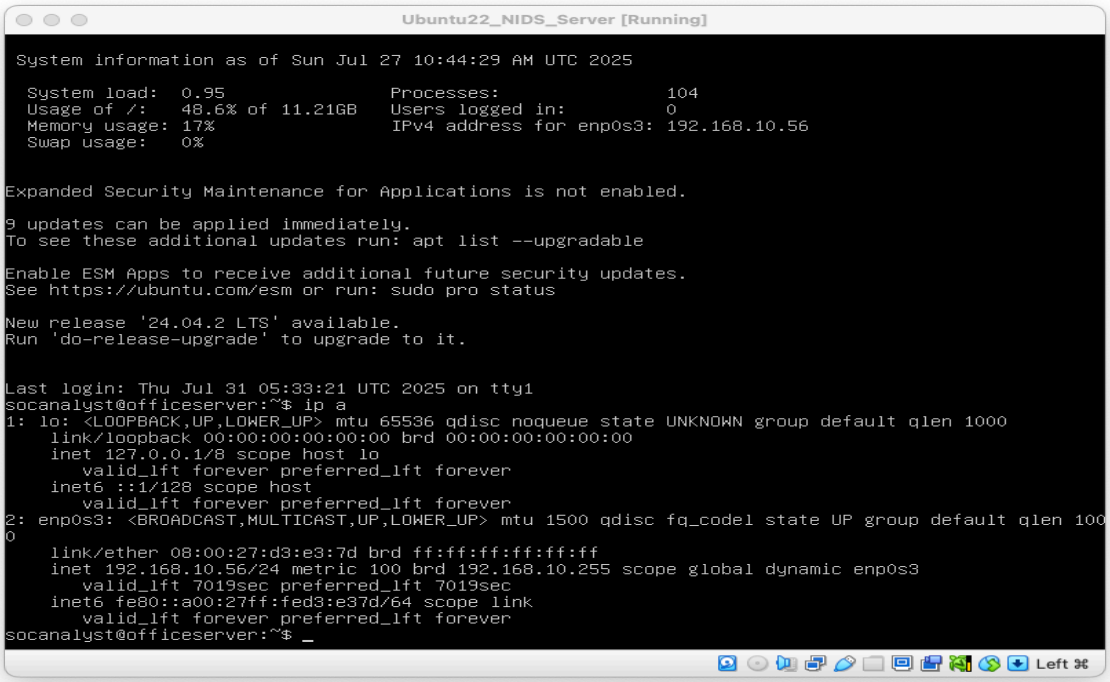




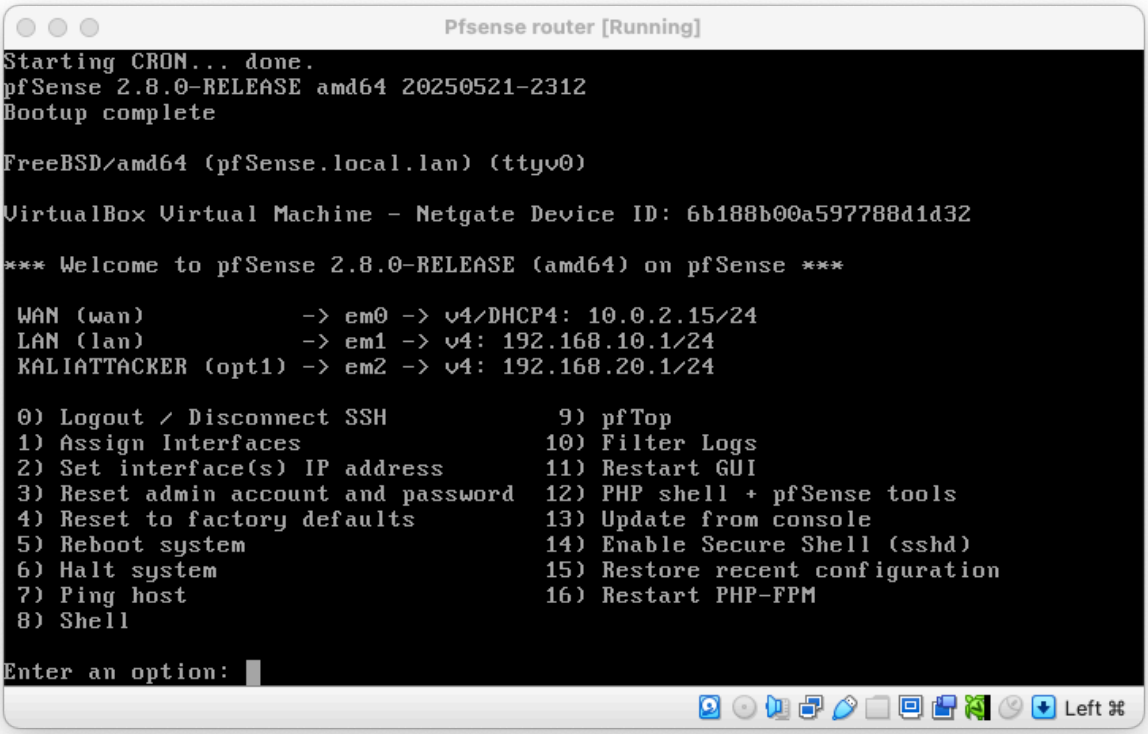
Ubuntu Desktop:



Ubuntu Server



Interfaces page in pfSense showing WAN, LAN, and OPT1



5: IDS/IPS CONFIGURATION ON PFSENSE WITH SURICATA

Installation of Suricata on pfSense

Tool Used: Suricata (via pfSense Package Manager)

Steps:

1. Login to pfSense Web UI via browser using https://192.168.10.1 from any LAN VM.

2. Navigate to System > Package Manager > Available Packages.

3. Search for Suricata and click Install.

4. After installation, go to Services > Suricata.

6. SURICATA INTERFACE CONFIGURATION

I created two separate interfaces for inspection:

Interface Purpose IP/Subnet Monitored by Suricata?

LAN Monitors internal LAN 192.168.10.1/24

KaliAttacker (OPT1) Monitors attacker subnet 192.168.20.1/24

Steps:

1. Navigate to Services > Suricata > Interfaces.

2. Click + Add to configure each interface:
Interface: LAN and KALIATTACKER
Description: Suricata_LAN and Suricata_Kali_Attacker
Enable Interface: ✔️
IPS Mode: ✔️ (to allow active blocking)
Promiscuous Mode: ✔️
Block Offenders: ✔️ (Important!)
Enabled loggings
Choose Home Net: default or define manually as 192.168.10.0/24,192.168.20.0/24
3. Scroll down to Logging Settings and enable logging for all relevant alert types.
4. Save and apply changes.

Rules Setup For Detection

I used Emerging Threats (ET Open) rules:
Steps:
1. Go to Services > Suricata > Global Settings.
2. In Rules Update Settings:
Enable ET Open Rules: ✔️
Update Frequency: 12 hours (or as needed)
3. Save and then click Update Rules button.
4. Navigate back to Interfaces > LAN > Categories.
5. Select categories like:

- Emerging-attacks
- Emerging-malware
- Emerging-scans
- Emerging-shellcode

6. Use Enable All button for the selected category.

Rule Management
Rule Category Activation:

- Under the Rules tab for each interface, categories such as Active Rules and Attack Response were enabled.
- Categories were manually reviewed to activate relevant rules against port scans, brute-force attempts, and suspicious payloads.

7. ATTACK SIMULATION, THREAT DETECTION AND ALERT GENERATION

This section outlines the simulation of a malicious scan from an untrusted host, the detection of this activity using Suricata IDS on pfSense, and the security response implemented via firewall rules. Here, I simulated unauthorized access attempts from an

attacker VM (Kali Linux) to the internal network, trigger detection using Suricata IDS, and respond by blocking the attacker via pfSense firewall rules.

Attack Simulation from Kali VM
➢ **Tools Used:**
Kali Linux VM (192.168.20.52)
nmap utility for reconnaissance
➢ **Commands Executed:**
- ping 192.168.10.1
- nmap -A -sS 192.168.10.1

ping tested host reachability.
nmap -A -sS performed a stealth scan with OS detection, version detection, script scanning, and traceroute.
➢ **Goal:**
Simulate an attacker probing a critical device — the pfSense gateway at 192.168.10.1.

Detection via Suricata on pfSense
- Suricata was active on both the LAN and Kali Interface.
- Alerts were generated for suspicious scanning behavior, visible in the pfSense > Suricata > Alerts section.

Sample alert:
[1:2024364] ET SCAN Nmap User-Agent Observed
Alerts confirmed successful detection of the scan from Kali.

8. INCIDENT RESPONSE IMPLEMENTATION

SID Management (Selective Blocking)
Due to UI limitations under the KaliAttacker interface (where SID Management was unavailable), we enabled drop actions manually using the central SID Mgmt tab.
Steps:
1. Go to Services > Suricata > SID Mgmt.
2. Click Add SID.
3. Add the SIDs (rule IDs) of signatures you want to enforce drop action on.
4. Apply changes using Apply SID Mgmt.
5. Confirm under interface settings that the rules are now enforcing drop.

SID Management Attempts & Outcome:
While Suricata's Global SID Management tool was used to attempt setting rules to drop, the expected behavior was not consistently achieved. This was due to the absence of the SID Management tab under the Kali interface (DMZ), which limited per-interface rule

control. Used the Global SID Mgmt section to add specific SIDs to be set to drop. However, these actions did not consistently result in alerts or active blocking, possibly due to interface mismatch or Suricata behavior with pfSense's packet flow handling.

SID Management Limitations:

- No SID Management section appeared under the Kali interface in Suricata.

Drop rules were added manually through Global SID Management, but this proved unreliable.

- Alerts were seen occasionally (e.g., QUIC decrypt fail), but no consistent block behavior was observed from SID actions alone.

Final Implementation That Achieved Blocking

To fulfill the detection and response objective of this project, a firewall-based mitigation strategy was adopted.

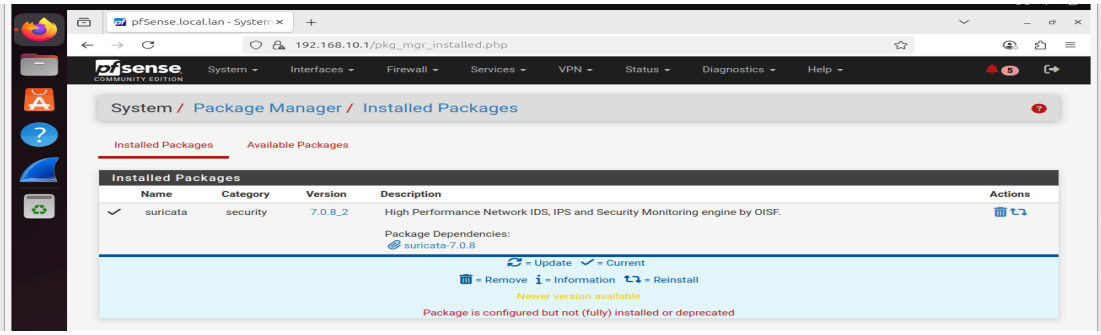
Effective Countermeasure:

A pfSense Firewall Rule was created to block all traffic originating from the attacker network.
Interface: Kali Interface (192.168.20.1/24)
Action: Block
Source: 192.168.20.0/24
Destination: 192.168.10.0/24
Protocol: Any
This rule successfully blocked Kali VM from accessing both Ubuntu Desktop and Server, achieving containment and demonstrating a security response to an identified threat.

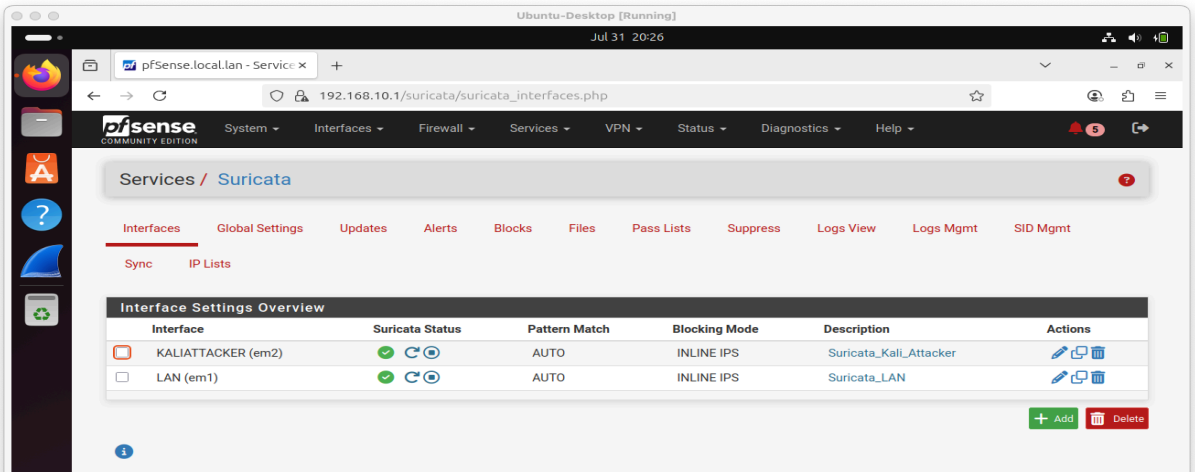
Validation of Block

- Repeated ping and nmap scans from Kali to 192.168.10.1 failed.
- No more Suricata alerts were generated for Kali after the block.
- Attacker was effectively isolated from the internal network.

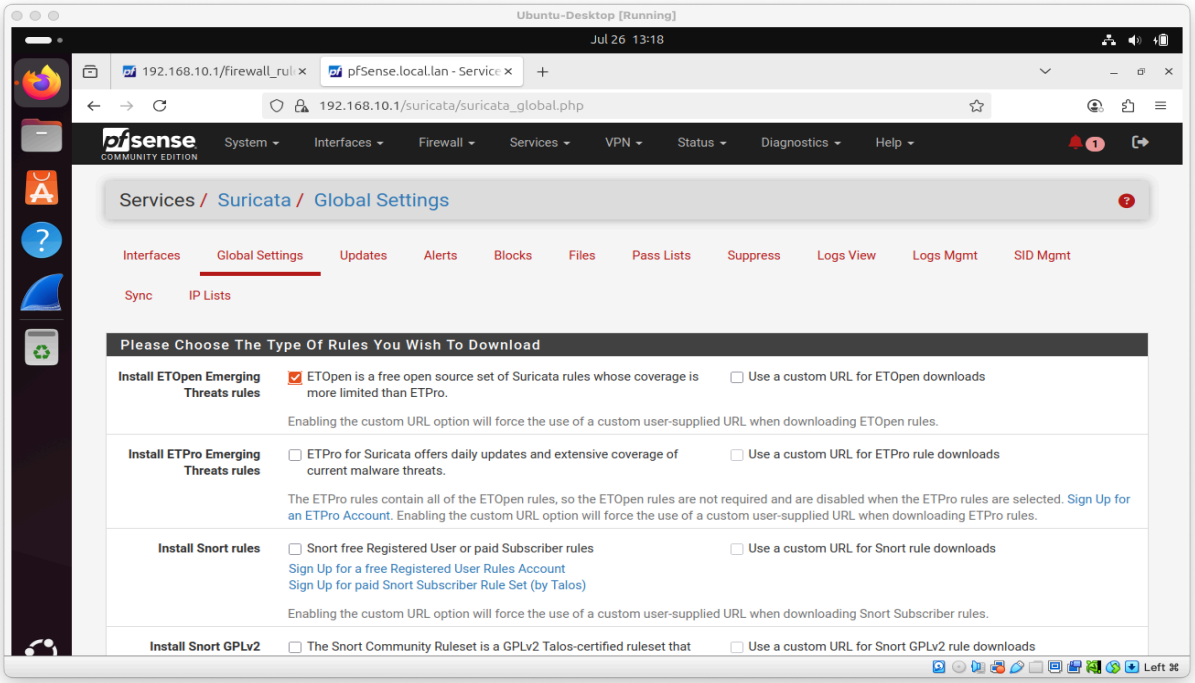
1. Suricata package installed on pfSense



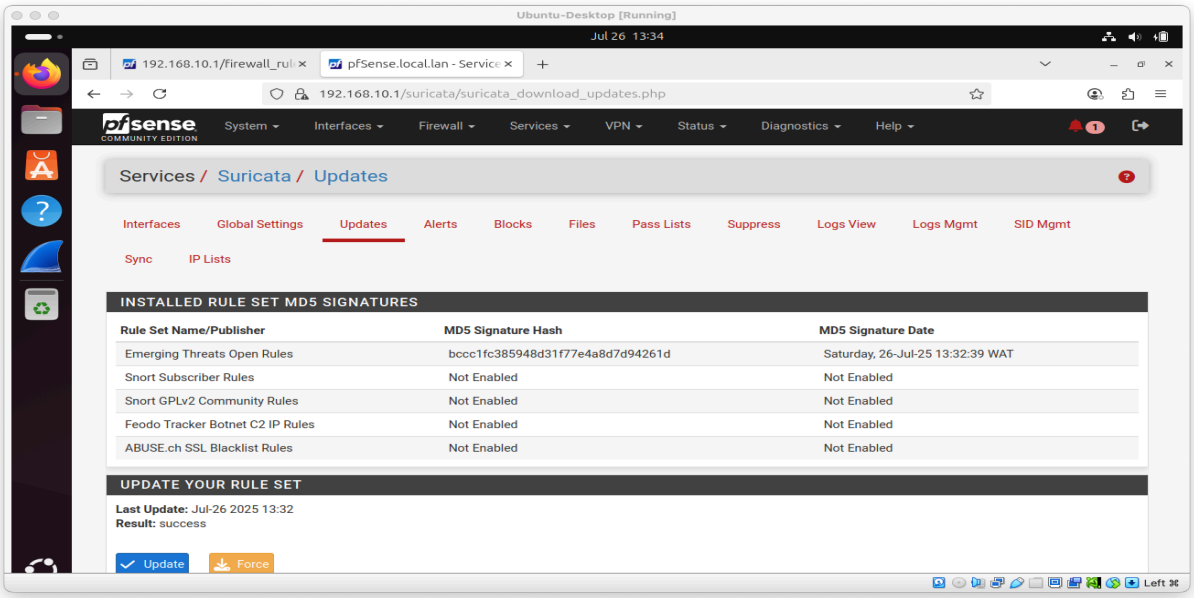
2. Interfaces where Suricata is enabled



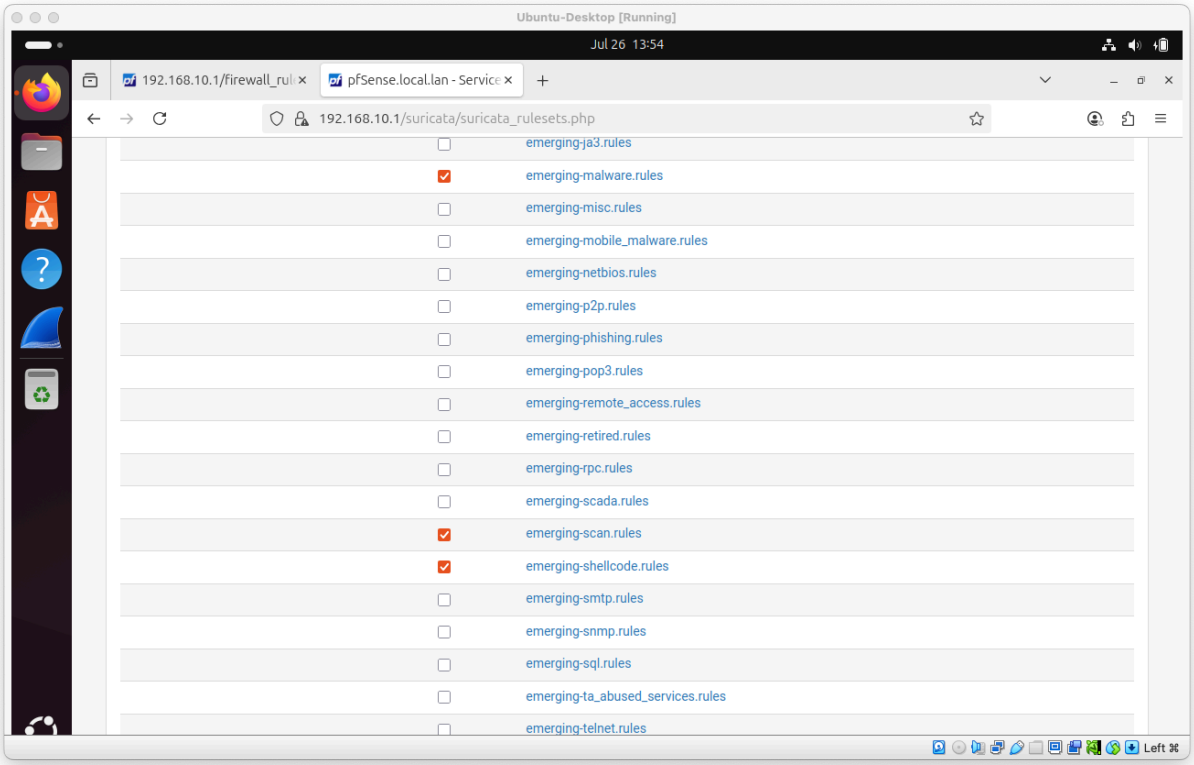
4. Rules categories and tab showing selected categories (e.g., attack-response.rules)



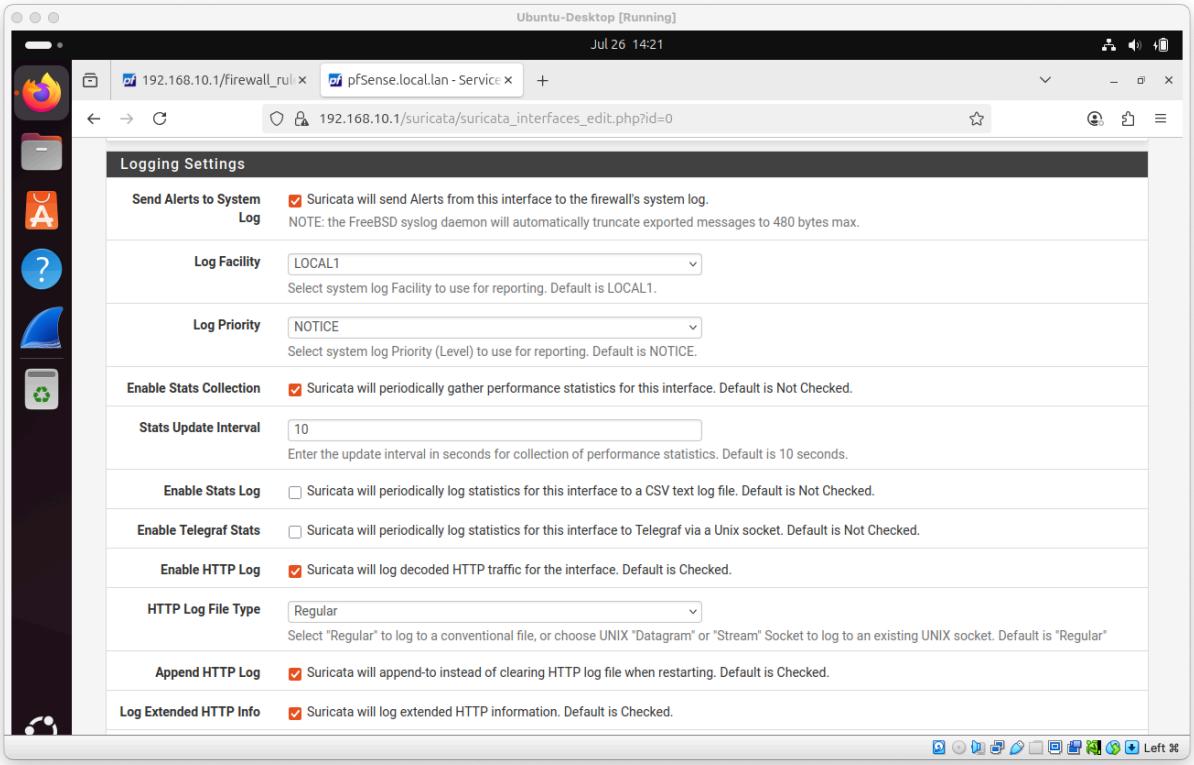
5. Force rule update

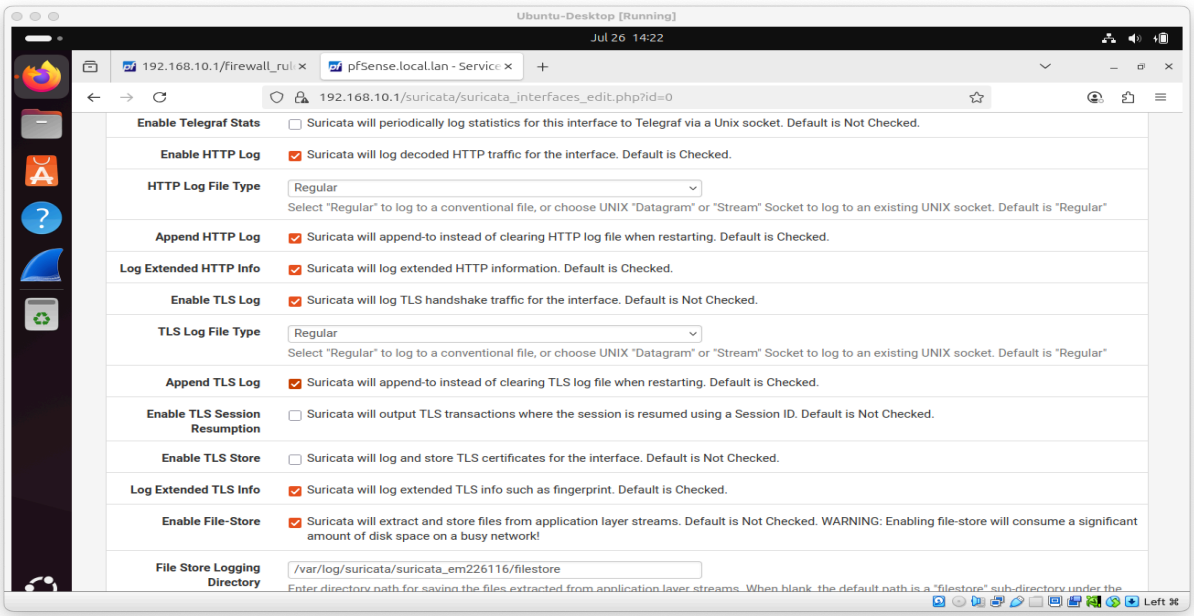


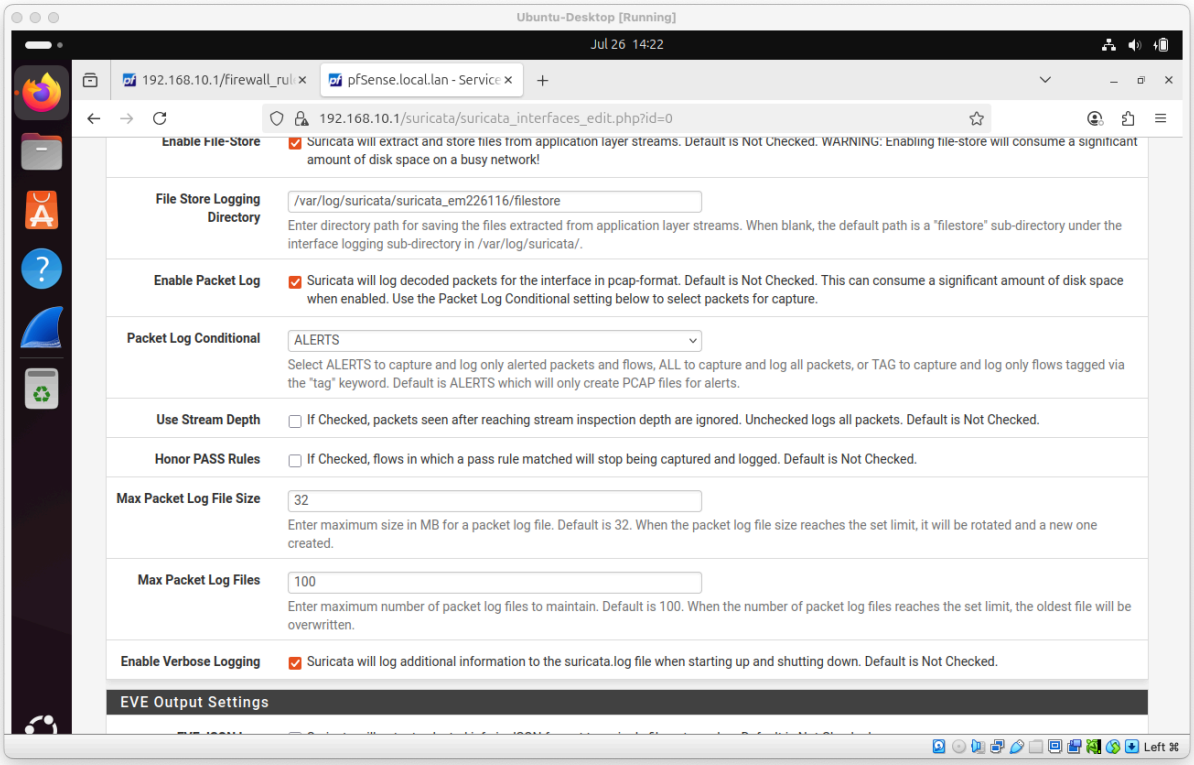
6. Enable ET rules



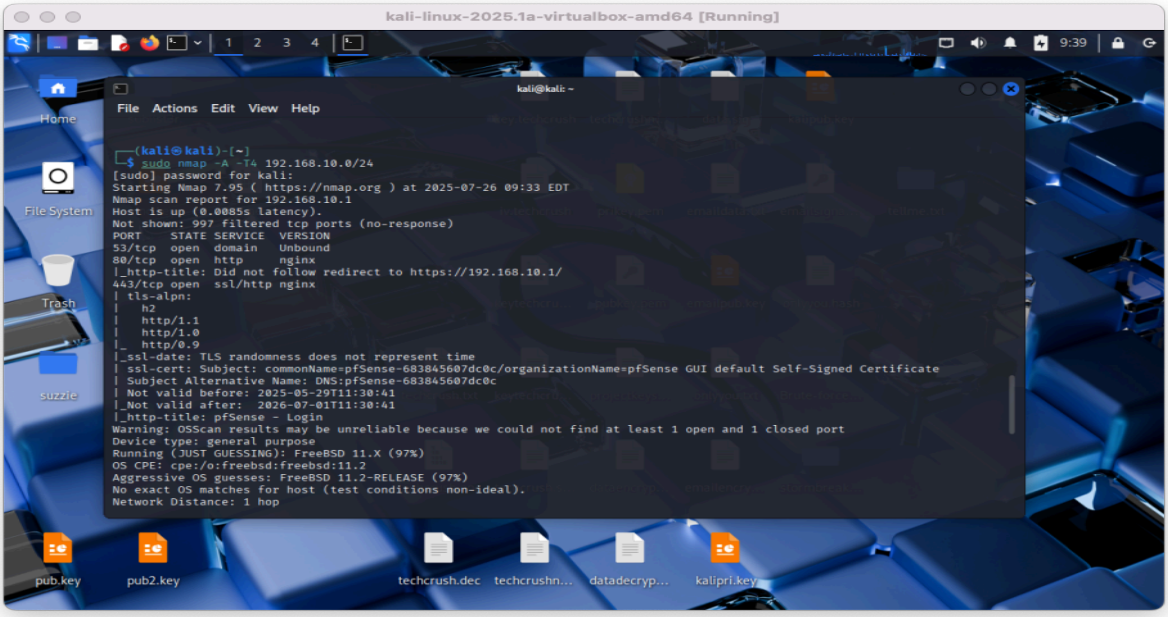
6B. Enabled loggings and alerts

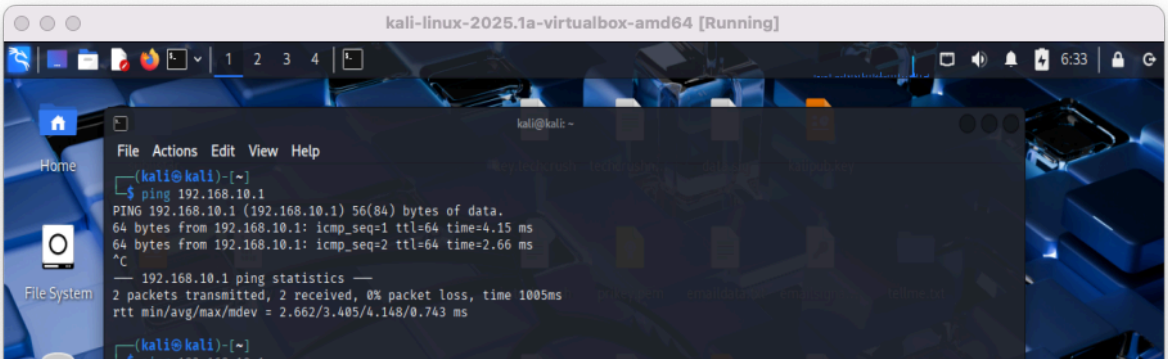




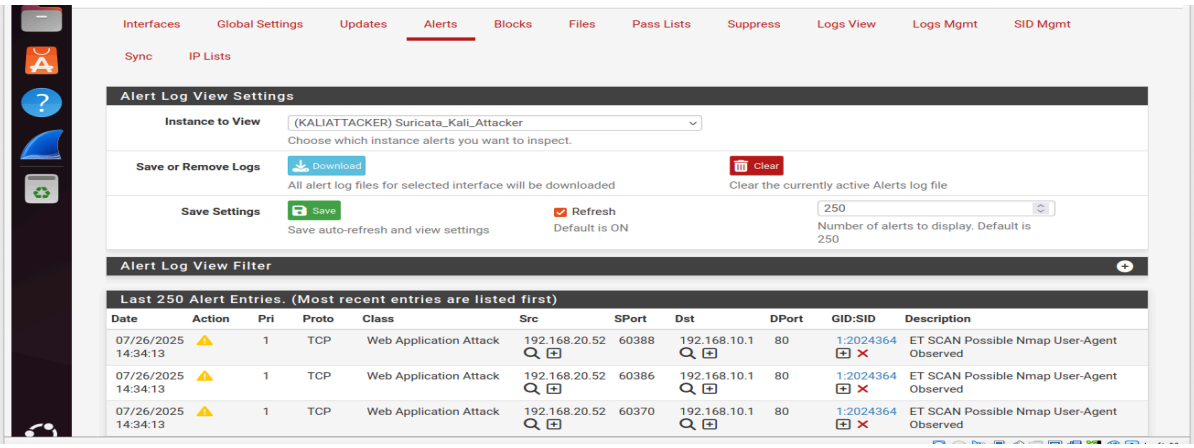


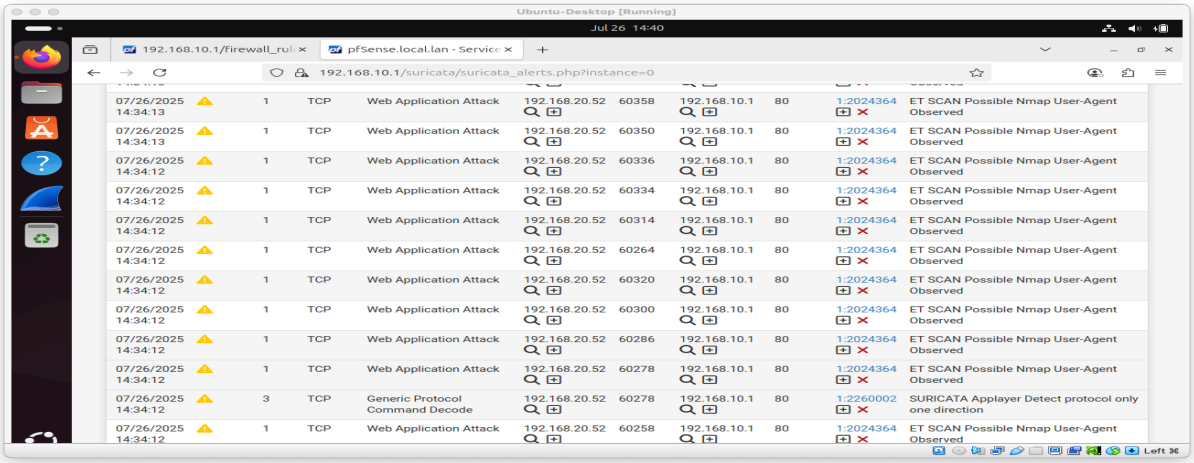
7. Terminal output showing ping and nmap -A -sS 192.168.10.1 from Kali before rule



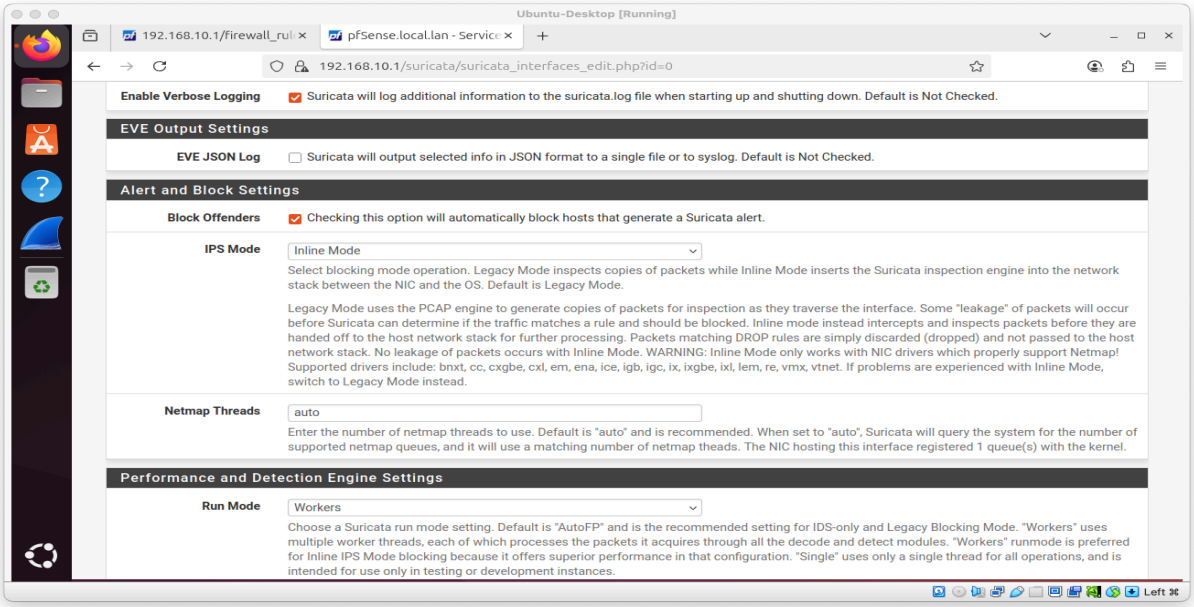


8. Suricata alert log showing triggered detection

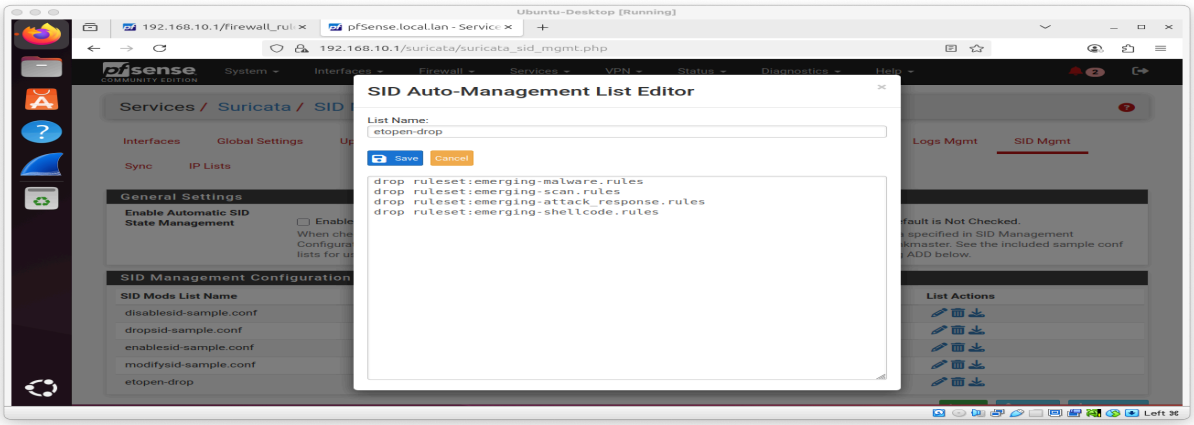


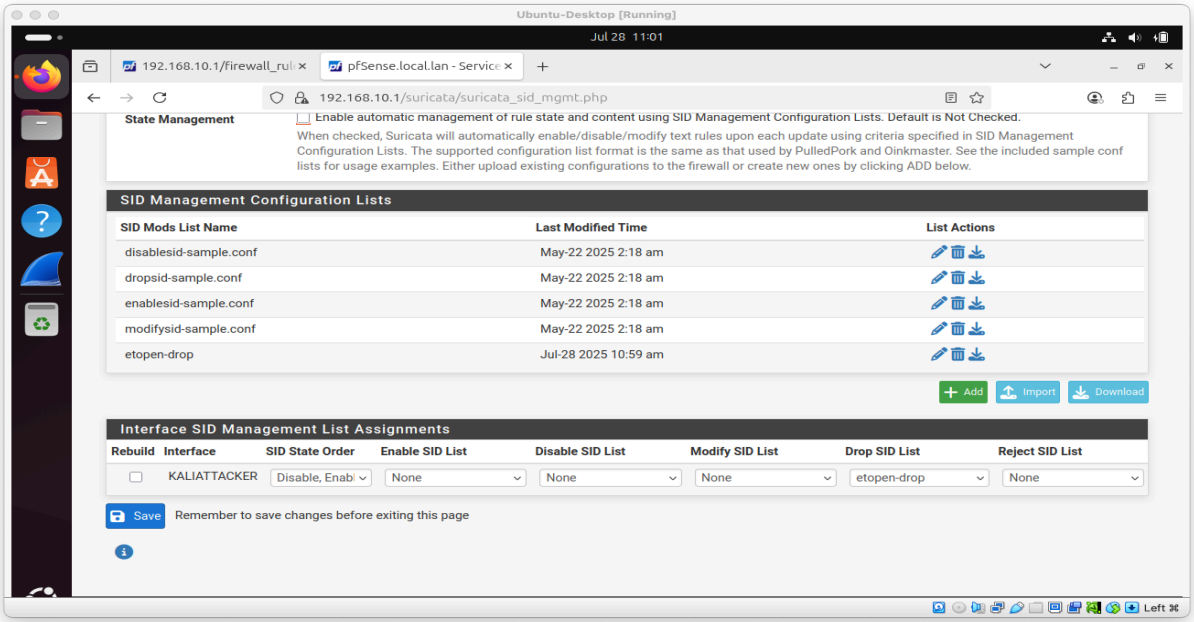


9. IPS mode and Block Offenders checkbox activated

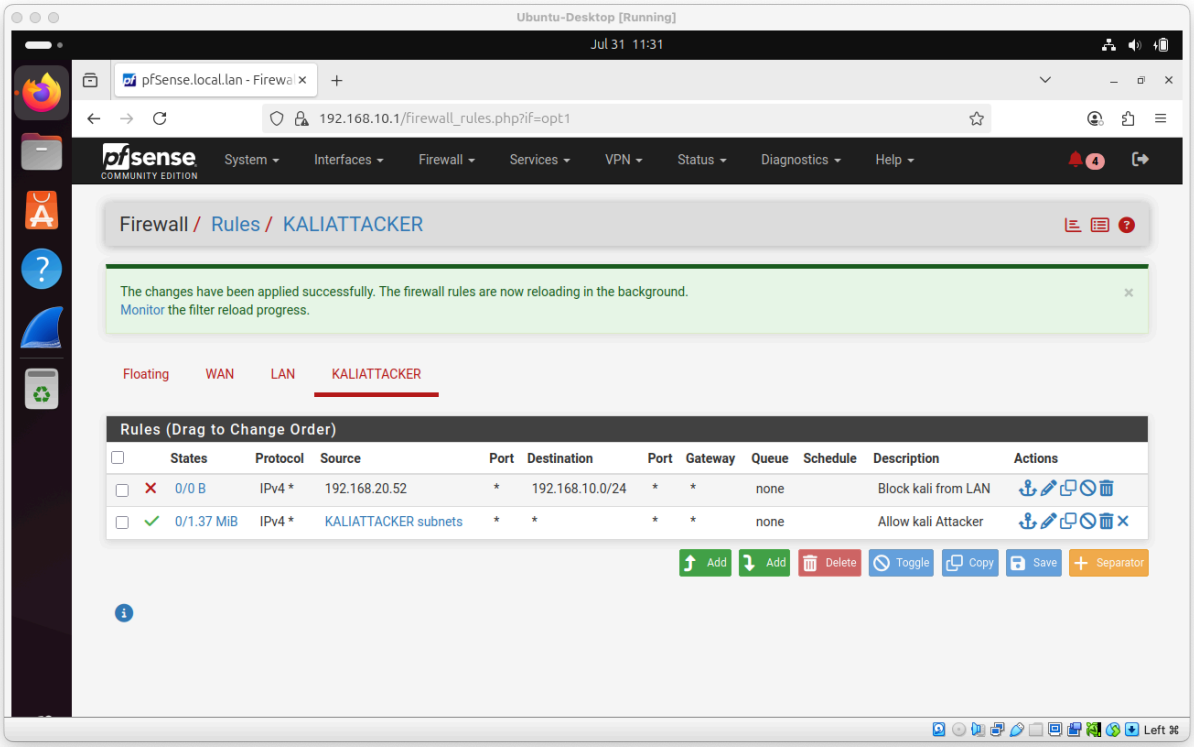


10. Global SID Management area with manually added SIDs and drop rules





11. pfSense Firewall Rule blocking Kali traffic



12. Failed re-attempted scan or ping from Kali

9. LESSON LEARNED AND RECOMMENDATIONS

Lessons Learned

1. Correct Network Segmentation Enables Effective Isolation

Placing the Kali attacker VM on a separate internal network (192.168.20.0/24) made it easier to detect and control traffic flows, proving how segmentation enhances visibility and security.

2. Suricata on pfSense Can Detect Basic Threats Efficiently

Suricata successfully picked up stealth scans and ping probes, reinforcing its value as a lightweight and powerful intrusion detection tool when properly configured on pfSense.

3. Detection Alone Isn’t Enough - Response is Key

While Suricata detected attacks, it didn’t block them automatically. Implementing a targeted pfSense firewall rule on the Kali interface provided an effective and immediate response mechanism.

4. SID Management Was Not Essential for This Use Case

Though Suricata supports SID Management for rule tuning and inline blocking, this project achieved its goal without using that feature. Blocking was instead implemented via firewall, showing there's more than one effective response path.

5. Simplicity and Clarity in Configuration Prevents Downtime

Keeping the Suricata configuration straightforward (monitoring only relevant interfaces) and using minimal, targeted scanning from Kali ensured a focused learning experience with observable outcomes.

6. Tool Synergy Is Powerful
Combining Suricata for detection with pfSense firewall for enforcement demonstrates a layered defense strategy — a foundational concept in real-world network security.

Recommendations for Future Implementations
1. Enable Logging and Monitoring on All Interfaces Early
Ensure the correct interfaces are monitored by Suricata from the beginning (e.g., LAN, attacker interface) to avoid blind spots during attack simulation.
2. Consider SID Management for Production Environments
For real-time blocking or deeper customization, SID Management can enhance Suricata's effectiveness by controlling rule actions (alert/drop/pass) at a granular level.
3. Use Wireshark for Deeper Traffic Analysis
While not needed here, incorporating Wireshark in future simulations can help with deeper traffic inspection and forensic analysis, especially when IDS alerts seem unclear.
4. Automate Response Where Possible
Integration of IDS with automated response systems (e.g., scripts, firewall APIs) can reduce manual intervention and enhance real-time threat mitigation.
5. Expand Threat Simulation Variety
Future tests should explore other forms of malicious traffic such as brute-force login attempts, malware downloads, or DNS tunneling to test detection rules more extensively.

This project successfully demonstrates the deployment and use of Suricata IDS on pfSense to detect and respond to malicious network activity in a controlled virtual lab environment. By simulating a basic reconnaissance attack using nmap and ICMP ping probes from a Kali attacker machine, we confirmed the ability of Suricata to identify suspicious activity on monitored interfaces.
Despite challenges such as the absence of SID Management options under the Kali interface, we effectively implemented a responsive mitigation strategy using pfSense firewall rules to block the attacker from further interacting with the internal LAN.
The entire process reinforced key cybersecurity concepts, including network segmentation, traffic monitoring, intrusion detection, and layered defense. The project also highlighted the value of simplicity, precision, and adaptability in configuring tools for specific network scenarios.
In real-world scenarios, integrating IDS with automated response systems and wider traffic analysis tools such as Wireshark would enhance visibility and security posture. However, for a lab-based educational task, this setup met the core objectives and provided a solid foundation in network threat detection and response.

Next Steps (Optional Enhancements for Future Work)

- Integrate Wireshark for full packet capture and traffic flow analysis.
- Simulate more advanced attacks (e.g., brute force, malware injection) for comprehensive rule validation.
- Explore automated blocking using Suricata's inline mode or integration with firewall APIs.
- Implement logging and alert forwarding to a central SIEM (e.g., Wazuh or ELK Stack) for enterprise-style threat monitoring.

10. CONCLUSION AND REFLECTION

Conclusion

This project was executed as Task 4 of the Code Alpha Internship Program, focusing on practical network defense through the use of Suricata IDS on pfSense. The goal was to set up a small virtualized network environment, simulate an attacker using Kali Linux, and implement detection and response mechanisms.

Through proper configuration of Suricata on pfSense, we achieved accurate detection of Nmap reconnaissance scans and ping probes. Although we initially explored Suricata's inline blocking capabilities, we opted for a manual response using pfSense firewall rules to block the attacker's IP address — demonstrating practical defensive action in a networked environment.

Key tasks completed:

- Setup of a multi-VM virtual network using Internal Networks on VirtualBox
- Installation and configuration of Suricata IDS on the pfSense router/firewall
- Execution of simulated attacks from Kali Linux
- Successful generation of detection alerts via Suricata
- Manual response by applying firewall rules on the attacker's interface

Reflection

This task provided hands-on exposure to critical cybersecurity operations including network segmentation, threat simulation, intrusion detection, and incident response. The challenge of ensuring proper interface selection and troubleshooting alert failures emphasized the importance of precise system configuration and real-time problem-solving.

Key Takeaways:

- **Traffic direction matters:** Suricata must monitor the correct interface where the threat enters
- **Tool limitations require adaptability:** Lack of SID Management on interface settings was mitigated by leveraging pfSense firewall
- **Basic attacks are useful for validation:** Simple tools like nmap and ping can effectively test detection pipelines
- **Firewall rules are still powerful:** Even without inline blocking from Suricata, we implemented solid mitigation

Overall, this task improved my technical agility, strengthened my IDS configuration skills, and deepened myunderstanding of how to pair detection tools with enforcement controls. It was a vital learning experience in building foundational competence for real-world cybersecurity roles.

11. REFERENCES

1. pfSense Documentation
Netgate Documentation. Configuring and Managing pfSense Firewall.
https://docs.netgate.com/pfsense/en/latest/
2. Suricata IDS Official Documentation
Open Information Security Foundation (OISF). Suricata - Open Source Threat Detection Engine.
https://docs.suricata.io/
3. Nmap Network Scanning Tool
Gordon Lyon (Fyodor). Nmap Security Scanner Manual and Examples.
https://nmap.org/book/man.html
4. VirtualBox User Manual
Oracle. VirtualBox Networking and VM Configuration.
https://www.virtualbox.org/manual/UserManual.html
5. Network Security Concepts
Cisco Press. Introduction to Intrusion Detection Systems and Basic Network Security.
(Used for background knowledge on IDS vs IPS, not directly cited)