9/13/2025

Final Task

SOC Analyst Internship Task Book.

Objective 1:

- Each intern is required to individually compile all tasks assigned during the summer internship program into a single booklet.

Submitted By:
Syed Muhammad Shah
SOC TEAM ~ OMEGA

ITSOLERA PVT LTD

MY INTERNSHIP REPORT

STRUCTURED
& ORGANIZED

NOTEBOOK

1.ACKNOWLEDGMENT

I would like to thank my internship **SUPERVISOR** and **ITSOLERA** for their guidance and continuous support during the summer internship. The hands-on labs and mentorship helped me to develop practical SOC skills, including SIEM configurations, firewall integration, endpoint monitoring, malware testing, and incident response workflows.

2.INTRODUCTION

This booklet documents all tasks completed during the **SOC Analyst internship**. It consolidates the technical steps, observations, troubleshooting, and learning outcomes for the three main assignments: **(1)** installing and configuring Wazuh and an agent with File Integrity Monitoring (FIM), **(2)** installing pfSense and integrating it with Wazuh to collect perimeter logs, and **(3)** performing a controlled malware test using the EICAR sample to observe detection across ClamAV, Wazuh, and pfSense, and then drafting a *NIST-aligned* incident response plan. Each task is explained in clear steps, followed by outcomes and practical tips.

3.TABLE OF CONTENTS

4.EXECUTIVE SUMMARY

This **internship task book** presents a detailed record of the projects and exercises I completed during my *SOC Analyst internship*. Over the course of the program, I worked hands-on with industry-standard security tools including Wazuh, pfSense, and IR , ClamAV, while following structured incident response practices.

The first task focused on deploying Wazuh and enabling File Integrity Monitoring to track unauthorized changes on endpoints. **The second task** expanded the environment by setting up pfSense as a firewall, integrating it with Wazuh for centralized monitoring, and troubleshooting log forwarding pipelines. **The third task** involved simulating a malware incident using the EICAR test file, observing detection across all layers, and drafting a NIST-aligned incident response plan.

Each section of this report outlines the objectives, steps, configurations, observations, challenges, and lessons learned. Collectively, these tasks strengthened my technical knowledge of SOC operations, improved my troubleshooting ability, and enhanced my understanding of layered defense and incident handling. **This report** demonstrates both my technical execution and my ability to document findings professionally.

5. TASK 1 — WAZUH INSTALLATION, AGENT DEPLOYMENT & FILE INTEGRITY MONITORING (FIM)

Objective: Deploy a dedicated Wazuh manager (server), connect a Linux agent (Kali/Ubuntu), and enable File Integrity Monitoring to detect file additions/changes on monitored endpoints.

Environment & Notes

- I used a separate Pre build OVA VM for the Wazuh server (to avoid my OS VM slowdown). This provided me a better stability for the dashboard and My System etc.
- Key IPs used in lab: static* :)
    - Wazuh Manager = 192.168.0.100
    - Kali Linux (Agent) = 192.168.0.101

High-level steps (what I did)

1. Prepare the server environment (update packages and install required tools & Virtual Box).
2. Install Wazuh Server Pre Build OVA VM From the official Wazuh Website.
3. Deploy Wazuh agent on the endpoint (Kali Linux), configure it to point to the manager, and start the agent service.
4. Enable File Integrity Monitoring (FIM) by adding directories to ossec.conf and restarting the agent.
5. Validate alerts on the Wazuh manager & dashboard.

Key commands i mostly used ,

sudo gedit /var/ossec/etc/ossec.conf

sudo systemctl enable wazuh-agent

sudo systemctl start wazuh-agent

sudo systemctl status wazuh-agent

sudo systemctl restart wazuh-agent

Check the agent log: sudo tail -n 50 /var/ossec/logs/ossec.log

Wazuh Manager (OVA) Logs : sudo tail -n 100 /var/ossec/logs/alerts/alerts.json | grep test_fim

- **(Futher Commands & Installation Are in Appendix)**

FIM configuration (example)

- I Add monitored directories to /var/ossec/etc/ossec.conf under <directories>:
    - /root, /etc, /home, /usr/bin, /home/virus (example paths)
- Set agent polling / realtime as required and restart the agent:

<directories check_all="yes" report_changes="yes" realtime="yes">/etc</directories>

Practical note about DHCP & static IPs

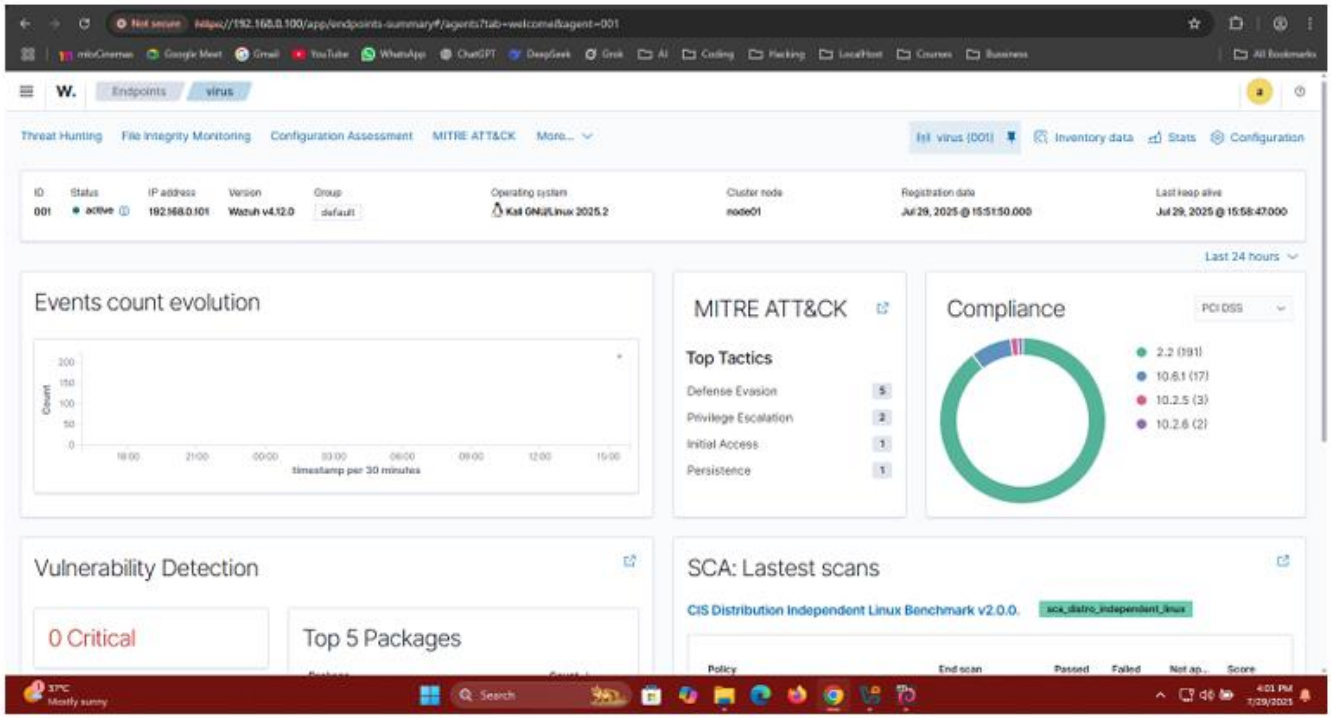
- During testing, DHCP-assigned IP changes caused agent-server mismatches. I had Two solutions in my mind :
    1. Assign static IPs to VMs, or
    2. Reserve IPs in the router DHCP using MAC addresses

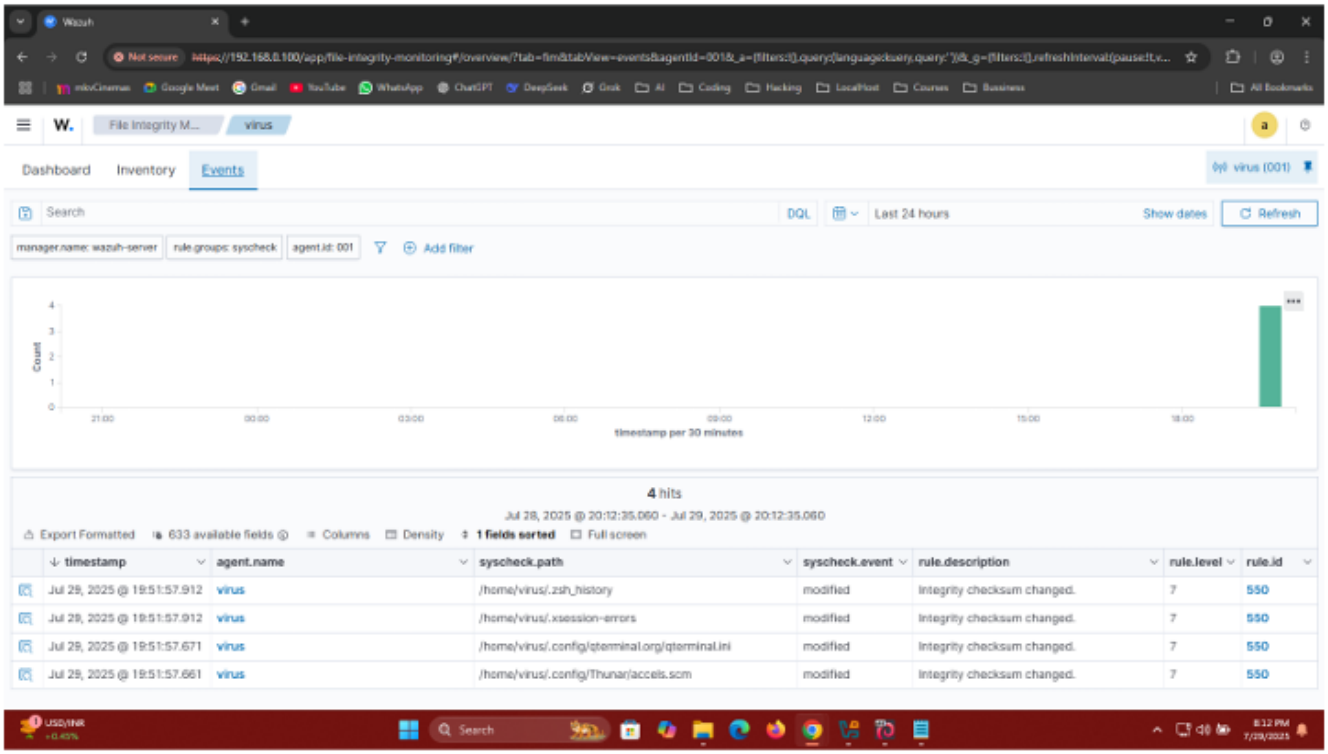
I used option 2 and it worked for me.

Validation & results

- I confirmed FIM alerts by checking manager logs and filtering alerts for test_fim. The dashboard showed real-time FIM events and threat hunting logs once everything was indexed.

Screenshots





Conclusion

Wazuh manager and agent were successfully deployed. FIM reliably reported file changes, enabling host-level detection when combined with endpoint AV alerts.

---

6. TASK 2 — PF SENSE INSTALLATION & WAZUH INTEGRATION (LOG COLLECTION & ALERTING)

Objective: Install pfSense in VirtualBox, configure network topology so lab traffic flows through pfSense, enable filtering (pfBlockerNG, DNSBL), and forward relevant logs to Wazuh for central visibility and alerting.

My Topology

- pfSense acts as perimeter between lab hosts (Kali, Ubuntu, Wazuh) and the real network/router.
- Adapter configuration:
    - pfSense Adapter1 = Bridged (WAN) — gets DHCP from router
    - pfSense Adapter2 = Internal Network (LAN) — static 192.168.0.1/24

- Kali & Wazuh on the same internal network (behind pfSense) for controlled testing.

pfSense installation highlights

- I Use the official pfSense ISO and allocate 2GB RAM, 20GB Disk & 1 Core CPU in the VM. Assign two adapters labelled above.
- On first boot pfSense asks to assign WAN/LAN — So i set them accordingly to my network setup.
- Change the admin GUI to HTTPS (System → Advanced → Admin Access) and secure access with firewall rules (allow GUI only from lab IP, block others).

My Network setup (static IPs)

- pfSense LAN: 192.168.0.1/24
- Wazuh: 192.168.0.100
- Kali: 192.168.0.101

pfSense security / blocking features i used

- **pfBlockerNG-devel** for GeoIP blocking (example: block Afghanistan & China) — validated by pinging a China IP and seeing blocks.
- **DNSBL** feature to block domains (e.g., blocking Facebook/YouTube) and log activity.
- Create firewall **Aliases** and LAN rules to block or allow specific hosts/domains (also used later for EICAR test).

Forwarding pfSense logs to Wazuh (two approaches, adopted syslog method)

- Attempted syslog-ng but GUI update failed; instead used the built-in remote syslog configuration in pfSense (Status → System Logs → Settings → Remote Logging Options).
    - Remote log server: 192.168.0.100:514 (Wazuh VM).
    - Enable categories: Firewall Events, System Events, Authentication, pfBlockerNG/DNSBL etc.

Wazuh (rsyslog) side configuration

1. Install rsyslog on the Wazuh manager:

➢ sudo yum install -y rsyslog

- sudo systemctl enable --now rsyslog

2. Create /etc/rsyslog.d/10-pfsense.conf with UDP/TCP 514 listeners and route logs from 192.168.0.1 to /var/log/pfsense.log:

$ModLoad imudp

$UDPServerRun 514

$ModLoad imtcp

$InputTCPServerRun 514

if $fromhost-ip == "192.168.0.1" then /var/log/pfsense.log

& stop

3. Ensure Wazuh manager reads /var/log/pfsense.log by adding a <localfile> entry to /var/ossec/etc/ossec.conf, then restart wazuh-manager.

Also create file for it ,

- sudo touch /var/log/pfsense.log
- sudo chown root:root /var/log/pfsense.log
- sudo chmod 0640 /var/log/pfsense.log
- sudo systemctl restart rsyslog

Decoder & rules testing

- Use /var/ossec/bin/wazuh-logtest to verify decoders produce pf events and that rules fire (example: rule id 87701 for pfSense firewall drop event). The lab demonstrates pre-decoding → decoding → filtering phases.
- Also I used custom Decoders and Rules . Pre Decoders were also working for me .

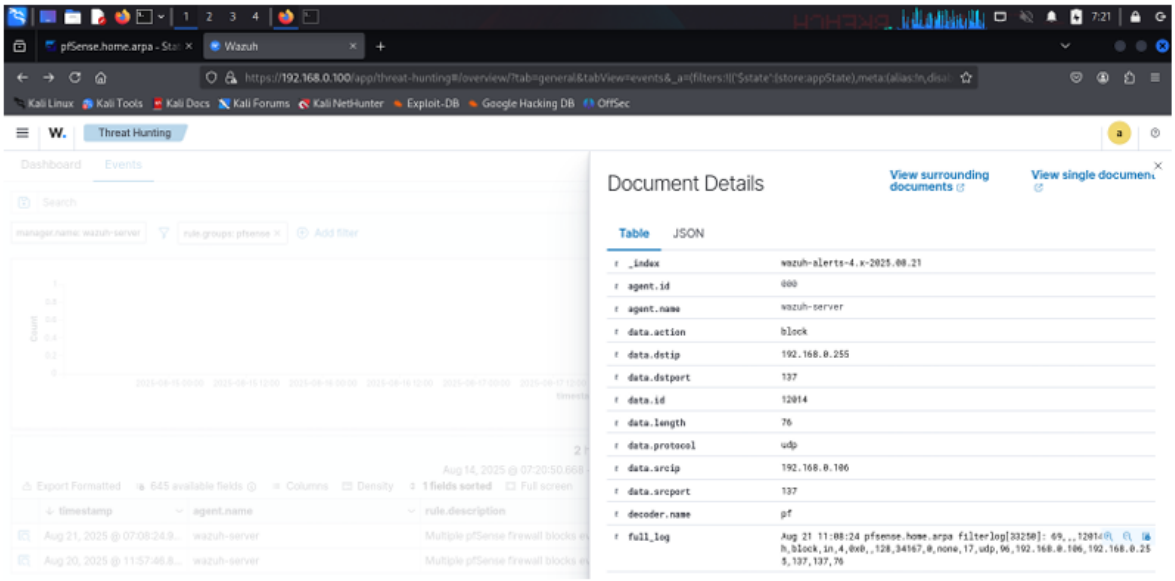
Commands I mostly used,

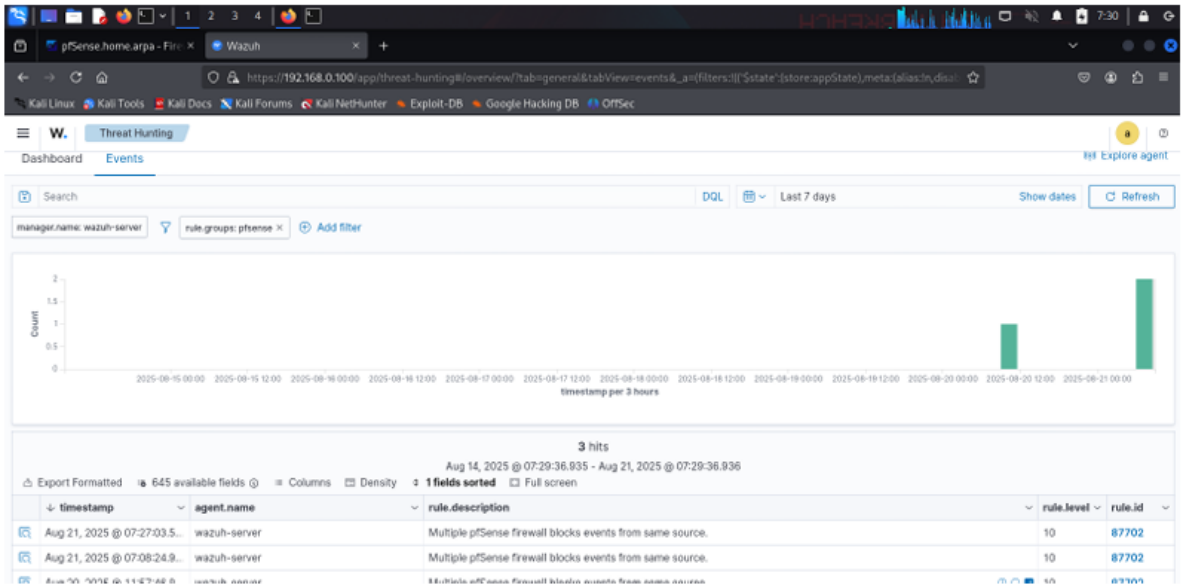
1. sudo nano /var/ossec/etc/decoders/local_decoder.xml
2. sudo nano /var/ossec/etc/rules/local_rules.xml
3. sudo /var/ossec/bin/wazuh-analysisd -t
4. sudo /var/ossec/bin/wazuh-logtest
5. sudo systemctl restart wazuh-manager
6. sudo systemctl status wazuh-manager
7. sudo nano /etc/rsyslog.d/10-pfsense.conf
8. sudo tail -f /var/log/pfsense.log
9. sudo tail -n 200 /var/log/pfsense.log

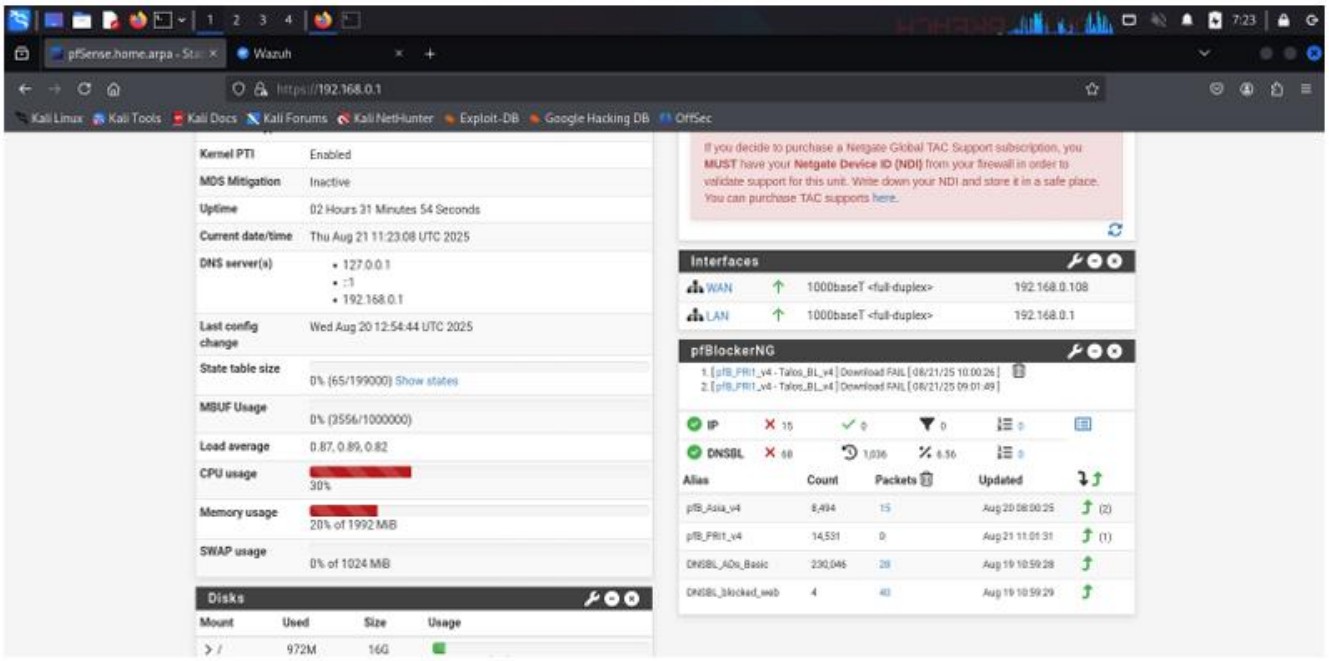
10. sudo tail -n 200 /var/ossec/logs/ossec.log
11. sudo tail -n 200 /var/ossec/logs/alerts/alerts.log
12. sudo tail -n 200 /var/ossec/logs/alerts/alerts.json
13. sudo tail -n 200 /var/ossec/logs/archives/archives.log
14. sudo nano /var/ossec/etc/ossec.conf
15. sudo nano /etc/network/interfaces
16. sudo nano /etc/sysconfig/network-scripts/ifcfg-eth0

- **(Futher Commands & Installation Are in Appendix)**

Screenshots







Conclusion

pfSense was fully functional as a perimeter device and logs were successfully forwarded to Wazuh.

---

7. TASK 3 — EICAR MALWARE TEST, CLAMAV INTEGRATION & INCIDENT RESPONSE PLAN

Objective: Download a Malware test file (EICAR) in a controlled lab VM, observe detection across ClamAV, Wazuh, and pfSense, analyze the logs, extract IOCs & IOAs, and document a NIST-aligned incident response plan.

My Environment

- Ubuntu VM (victim): 192.168.0.103
- Wazuh: 192.168.0.100
- pfSense LAN: 192.168.0.1/24
- ClamAV installed on Ubuntu and its logs forwarded to Wazuh.

I download Ubuntu as a fresh start instead of using kali which was already setup.

Preparation/Setup

1. Configure Ubuntu network (netplan) to use static IP in the lab.
2. (Pre-requisites were full-filled as Done in Task 1,2)
3. Install ClamAV and clamd: sudo apt install -y clamav clamav-daemon and update definitions.
4. Configure ClamAV to log to syslog (LogSyslog true in clamd.conf), and ensure Wazuh reads relevant syslog files by adding <localfile> entries to /var/ossec/etc/ossec.conf. Restart services as needed.

pfSense lab changes for detection

- Create an alias (name: EICAR) for www.eicar.org and add a LAN rule to block access from Ubuntu VM to that domain/IP so firewall logs show blocked outbound attempts.

Test & detection

- The lab attempted to download eicar.com.txt. pfSense logged outbound connections to 89.238.73.97 (EICAR host) and blocked them (logged). ClamAV on the host detected the EICAR file as Win.Test.EICAR_HDB-1. Wazuh aggregated these signals: ClamAV alert (rule ID 52502) and grouped pfSense firewall-block alerts (rule 87702).

Key evidence captured (IOCs)

- File: /home/ubuntu/malware/eicar.com.txt
- MD5: 44d88612fea8a8f36de82e1278abb02f
- SHA-256: 275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f
- pfSense destination IP: 89.238.73.97
- Wazuh ClamAV detection: rule.id = 52502 and grouped pfSense alert 87702 (timestamps as recorded in logs).

Log (example)

- ClamAV full_log captured in Wazuh:

Sep 01 15:08:17 ubuntu2204 clamd[17638]: … /home/ubuntu/malware/eicar.com.txt: Win.Test.EICAR_HDB-1(44d88612…:68) FOUND

This exact line appears in the Wazuh full_log field for ClamAV alerts.

Incident Response Plan (NIST SP 800-61 aligned) – Example Snippet

Executive summary:

On 2025-09-01 at ~11:08 a test file (EICAR) was downloaded on ubuntu2204 (192.168.0.103). ClamAV detected the file and pfSense logged the outbound attempt; Wazuh correlated these events. The incident was contained without impact.

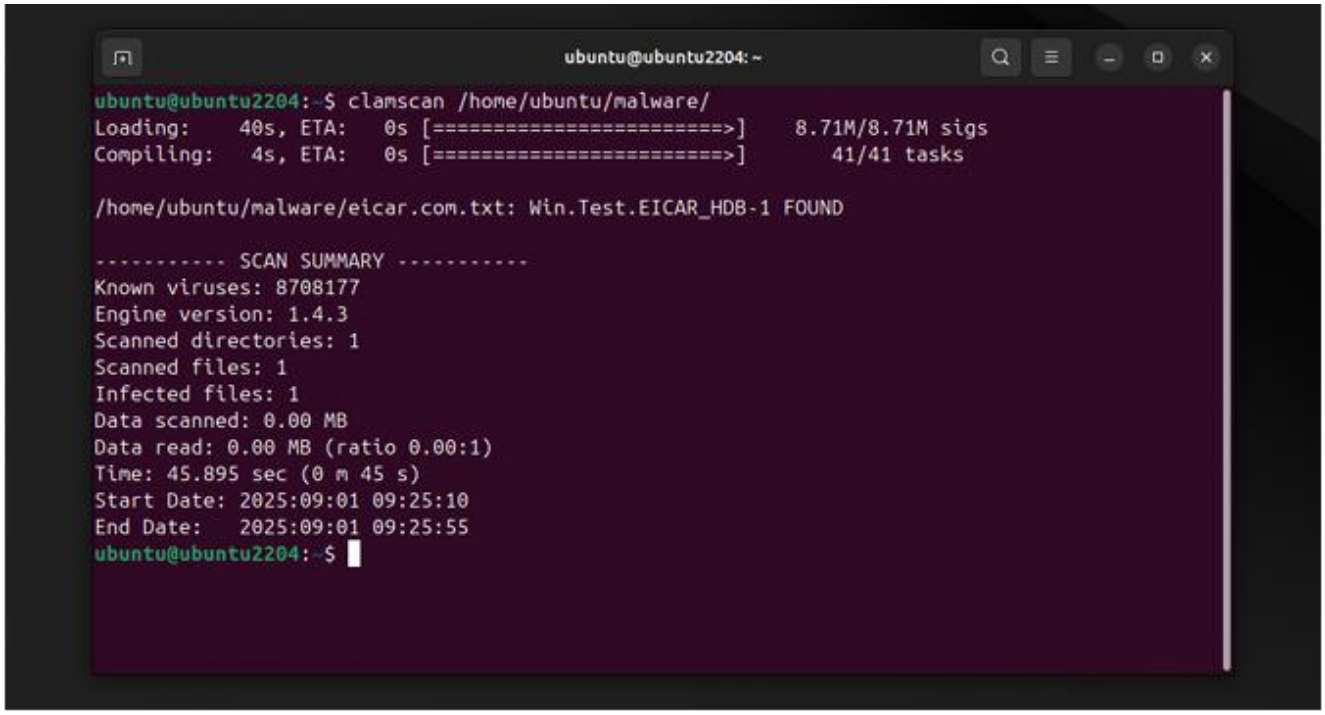
1. **Detection & Analysis**
    - Confirm the ClamAV detection, inspect Wazuh alerts, and review pfSense logs to determine the pattern and time window.
2. **Containment**
    - Immediately isolate the affected VM (restrict network access) and maintain snapshots for forensics.
    - Apply firewall blocks (already blocked in lab via alias/rule).
3. **Eradication**
    - Remove the test file: sudo rm /home/ubuntu/malware/eicar.com.txt.
    - Run full scans and verify no other malicious artifacts remain (clamscan, clamdscan).
4. **Recovery**
    - Restore from a known-good snapshot, re-enable network, and validate monitoring continues to work.
5. **Post-Incident**
    - Document IOCs / IOAs and update detection/correlation rules to alert on grouped repeated attempts.
    - Maintain snapshots before malware testing and tighten configuration (e.g., add correlation rules in Wazuh).

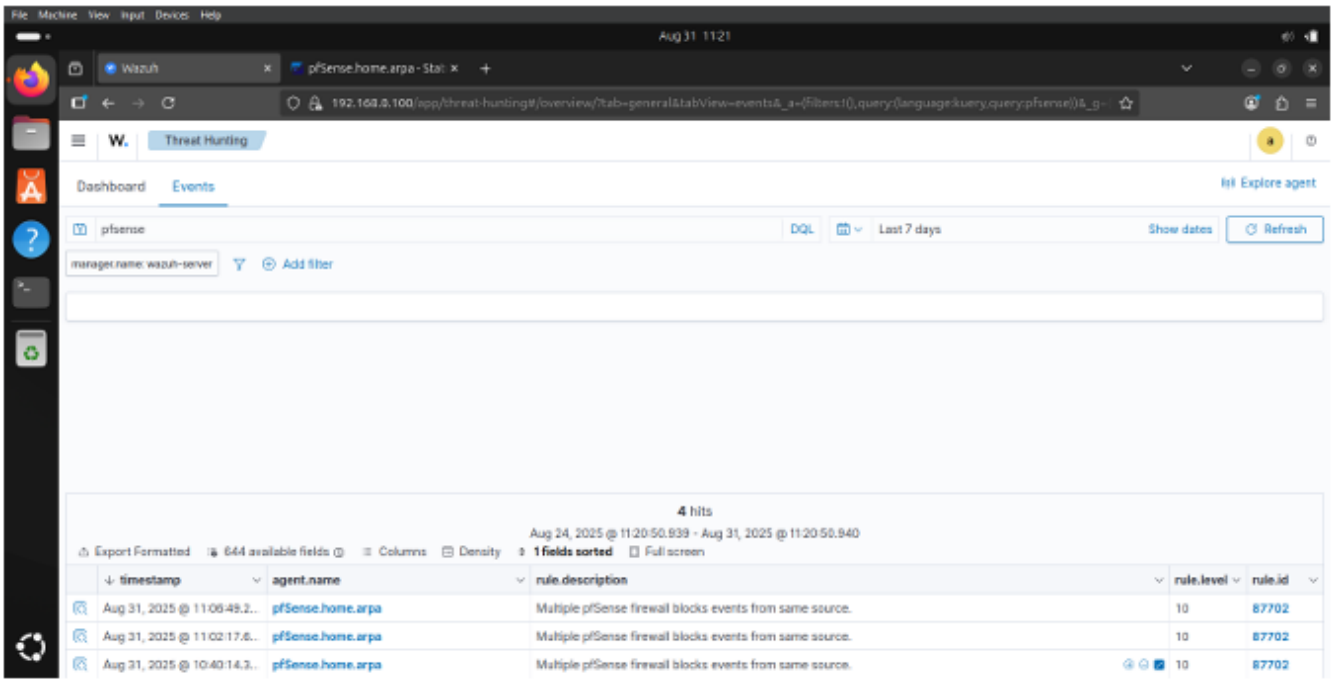
Commands I mostly used here,

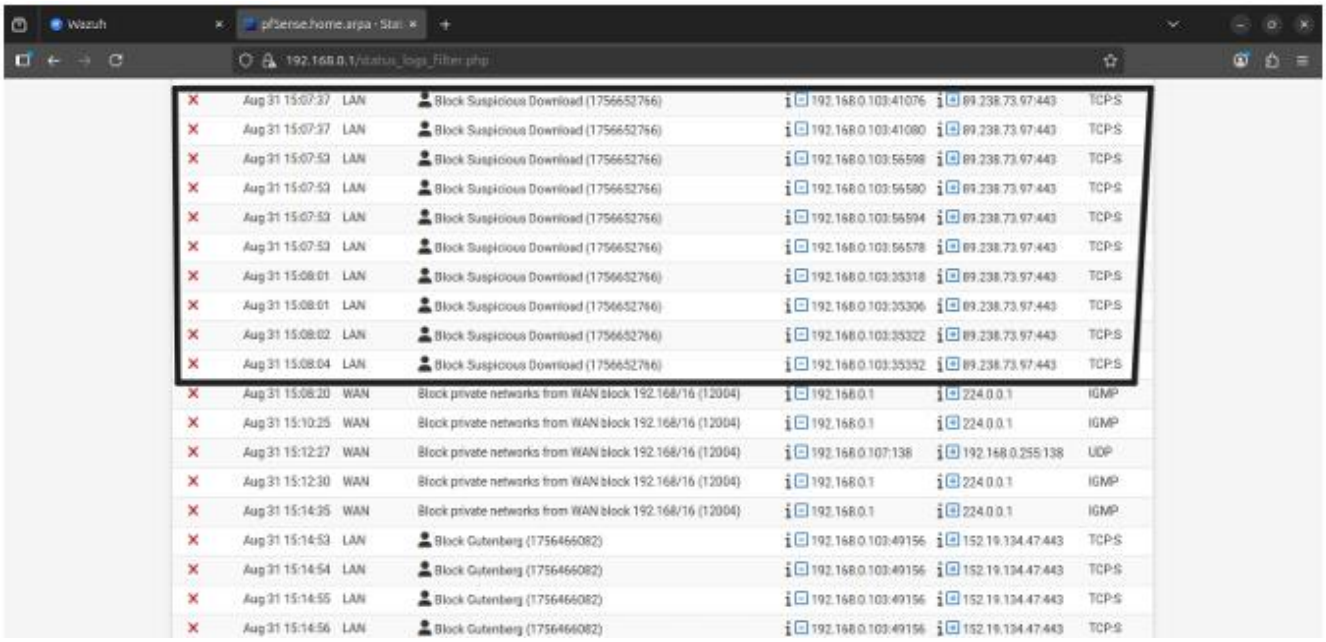
1. sudo tail -n 50 /var/ossec/logs/alerts/alerts.log
2. sudo tail -n 100 /var/ossec/logs/alerts/alerts.log | grep clamav
3. sudo tail -n 100 /var/ossec/logs/alerts/alerts.json | grep clamav
4. sudo tail -f /var/ossec/logs/ossec.log

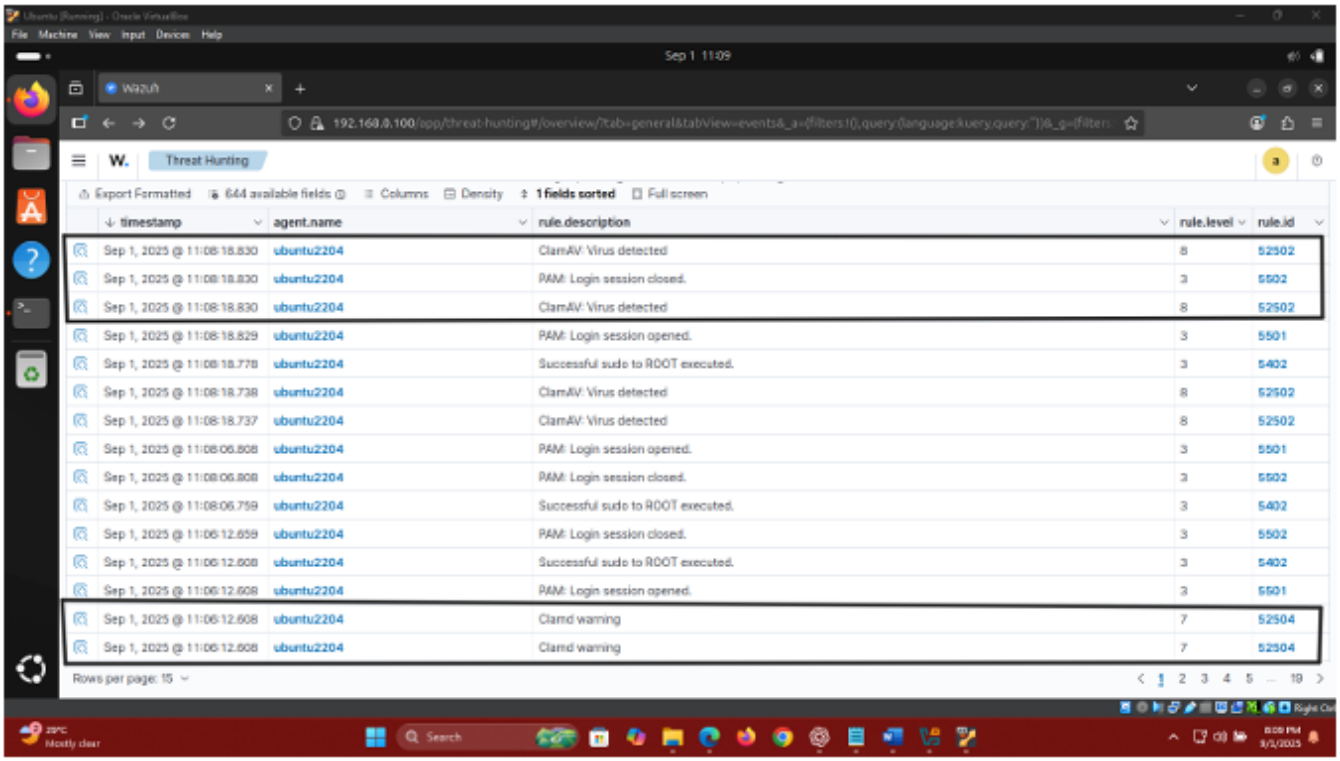
5. sudo tail -f /var/ossec/logs/alerts/alerts.log | grep -i eicar
6. sudo gedit /var/ossec/etc/ossec.conf | sudo cat /var/ossec/etc/ossec.conf
7. sudo cat /etc/clamav/clamd.conf | sudo gedit /etc/clamav/clamd.conf
8. sudo tail -n 50 /var/log/syslog | grep -i 'clamd\|FOUND'
9. sudo tail -n 50 /var/ossec/logs/alerts/alerts.log | grep -i clamav

Screenshots









Conclusion

The EICAR exercise validated the detection chain — endpoint AV (ClamAV), perimeter firewall (pfSense), and centralized SIEM (Wazuh). The lab confirmed that layered defenses provide actionable telemetry for an analyst.

8. Task 4 — Malware Breach Report & Internship Task book

Objective 1: Comprehensive Internship Task book

- Each intern is required to individually compile all tasks assigned during the summer internship program into a single booklet. This booklet should be well-designed, properly formatted, and written in your own words. Ensure originality in your submissions—no plagiarism will be tolerated.

    - **Instructions:**
        1. Collect all tasks from the beginning of the internship up to the final task.
        2. Rewrite the tasks in your own words.
        3. Design the booklet professionally (use tools like Canva, Adobe, etc.).
        4. Each intern must submit their own version individually.

SOLUTION : Here is the task book I created by my self , rewrite with my own words . I don't use or compile all my tasks pdf . instead I rewrite the main snippets and conclusions in my own words as concise as possible . and following the instructions I did by Best .

Objective 2: Malware Breach Report

- Research and create a report on recent or current malware attacks, breaches, and the well-known companies affected by these incidents.

    - **Instructions:**
        1. Research at least 10 major breaches that occurred recently.
        2. Analyze the details of each breach, focusing on the nature of the attack, the company affected, and the aftermath.
        3. Read articles from credible sources and summarize the findings.
        4. The report should be clear, concise, and insightful.

SOLUTION: I did research and explore a lot of websites to gather information. And to be honest I use different Ai tools to make my report and research more precise . By following the instructions i write 18 recent breaches , made a professional report & submit With this task.

---

9. CHALLENGES & SOLUTIONS

1. **DHCP IP changes caused agent-server mismatches**
    - *Solution:* Reserve IPs in router DHCP by MAC or assign static IPs to VMs. (Reported in Task 1.)
2. **pfSense → syslog-ng package failed to update via GUI**
    - *Solution:* I Used pfSense builtin remote syslog forwarding to send logs to Wazuh and configure rsyslog on the Wazuh manager to receive them.
3. **pfSense alerts not visible in Wazuh Dashboard (Fixed but again issue caused in Task 3)**
    - *Root cause:* Filebeat output misconfiguration and authentication/connectivity between Filebeat and OpenSearch/Wazuh indexer.
    - *Solution:* Fix filebeat.yml output.elasticsearch section (correct host, username/password, certs), restart Filebeat, and ensure dashboard opensearch_dashboards.yml points to the indexer with valid credentials. Verify indices appear.
    - ***But in Task 3 :*** i used agent method , I install wazuh agent for free BSD in pfSense and setup and configure and then it was fine .
4. **Grouping hides many low-level events (noise reduction vs. visibility)**
    - *Solution:* Understand grouping rules (e.g., 87702) and use previous_output field to see the raw lines; tune grouping thresholds and add custom rules decoders etc.
5. **ClamAV logging to syslog required changes**
    - *Solution:* Set LogSyslog true in clamd.conf, ensure Wazuh reads /var/log/syslog and ClamAV logs are parsed by built-in decoders.

10. KEY LEARNINGS

- **Layered defense works:** Host AV + FIM + perimeter logs + SIEM correlation provide better visibility than any single control.
- **Configuration reliability:** Use static or DHCP reservations for lab VMs to avoid IP mismatches.
- **Testing safely:** Use EICAR to practice detection without real risk; always snapshot VMs before malware testing.
- **Documentation is part of the job:** Clear write-ups enable repeatability and faster troubleshooting.
- **Networking:** Network Topology Matters alot !
- **Troubleshooting is a SOC skill:** Many issues (like pfSense syslog errors or Filebeat misconfigurations) required step-by-step debugging, showing that SOC work is as much about problem-solving as it is about detection.
- **Log correlation is powerful:** Collecting logs from multiple sources (host, firewall, AV) into Wazuh highlighted how cross-correlation improves detection accuracy.
- **Firewalls are more than traffic blockers:** pfSense demonstrated advanced features like GeoIP filtering, DNSBL, and integration with SIEM, showing how firewalls support broader SOC operations.
- **Importance of decoders and rules:** Understanding how Wazuh decoders and rules interpret raw logs taught me the value of tuning detection for accuracy and reducing false positives.
- **Regular updates are critical:** Tools like ClamAV require up-to-date definitions, and Wazuh requires maintained indices — reinforcing the importance of keeping systems current for reliable detection.
- **Security is teamwork:** Even though tasks were individual, sharing troubleshooting experiences and learning from mentors emphasized collaboration in SOC environments.
- **Hands-on practice builds confidence:** Setting up real VMs, configuring agents, and analyzing alerts provided practical confidence that theoretical study alone cannot offer.

11. CONCLUSION

This internship allowed me to gain hands-on exposure to the core responsibilities of a SOC Analyst. Starting with the deployment of Wazuh and its File Integrity Monitoring feature, I learned how crucial it is to monitor endpoints for unauthorized changes. The process of installing, configuring, and troubleshooting the SIEM taught me not only technical commands but also the patience and structured thinking needed in real-world security operations.

Moving forward, integrating pfSense into the environment expanded my understanding of network-level defense. Initially, I forwarded logs using the syslog approach, but later I also explored the **agent method by installing the Wazuh FreeBSD agent directly on pfSense**. This gave me deeper visibility and proved more stable for continuous log forwarding. Alongside pfSense, I also practiced using the **UFW firewall** on Linux systems to reinforce the concept of layered security. These parallel setups helped me compare different firewall solutions and appreciate how multiple defense layers complement each other when monitored in a SIEM.

Finally, the EICAR malware simulation tied everything together. Observing detections across ClamAV, Wazuh, and pfSense highlighted the value of log correlation and reinforced the concept that no single tool is enough — security comes from integration and coordination. Beyond the technical skills, I also gained a strong appreciation for documentation, troubleshooting, and clear communication. Overall, this internship has been a journey of learning, experimentation, and problem-solving that has prepared me to confidently step into real SOC environments.

12. MY FINAL THOUGHTS

Completing this internship has been a valuable and transformative experience for me. Working directly with SOC technologies, troubleshooting real-world issues, and practicing malware detection gave me confidence in my technical and analytical skills. More importantly, I realized that documentation, clear communication, and structured processes are as essential as technical execution in security operations.

I am grateful for the guidance and support of my mentors, and I believe the skills I have gained will serve as a strong foundation for my future career as a cybersecurity professional. This internship confirmed my passion for SOC operations, and I am motivated to continue building expertise in threat detection, response, and security monitoring.

13. APPENDIX

1) **Wazuh manager install (example)**
   - sudo apt update && sudo apt upgrade -y
   - sudo apt install curl apt-transport-https lsb-release gnupg -y
   - curl -sO https://packages.wazuh.com/4.12/wazuh-install.sh
   - sudo bash ./wazuh-install.sh -a
   - # open firewall
   - sudo ufw allow 1515/tcp
   - sudo ufw allow 1514/tcp
   - sudo ufw allow 443/tcp
   - sudo ufw reload
   - AS MENTIONED , I USED PRE BUILD OVA VM , DOWNLOADED FROM OFFICIAL WAZUH WEBSITE .

2) **Agent install (example)**
   - # On agent
   - curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gpg --dearmor | sudo tee /usr/share/keyrings/wazuh.gpg > /dev/null
   - echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://packages.wazuh.com/4.x/apt/ stable main" | sudo tee /etc/apt/sources.list.d/wazuh.list
   - sudo apt update
   - sudo apt install wazuh-agent
   - # Edit /var/ossec/etc/ossec.conf and add <client><server><address>192.168.0.100</address></server></client>
   - sudo systemctl enable --now wazuh-agent
   - sudo systemctl restart wazuh-agent

3) **rsyslog config for pfSense logs (on Wazuh manager)**
   - Create /etc/rsyslog.d/10-pfsense.conf:
     $ModLoad imuxsock
     $ModLoad imklog
     $ModLoad imudp
     $UDPServerRun 514

```
$ModLoad imtcp
$InputTCPServerRun 514

if $fromhost-ip == "192.168.0.1" then /var/log/pfsense.log
& stop
```

- Then:
- sudo touch /var/log/pfsense.log
- sudo chown root:root /var/log/pfsense.log
- sudo chmod 0640 /var/log/pfsense.log
- sudo systemctl restart rsyslog

Add to Wazuh /var/ossec/etc/ossec.conf:

```
<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/pfsense.log</location>
</localfile>
```

4) ClamAV logging & scan commands

- In /etc/clamav/clamd.conf set LogSyslog true.
- Restart services:
  sudo systemctl restart clamav-daemon
  sudo systemctl restart rsyslog
  sudo systemctl restart wazuh-agent

- Scan:
  sudo clamdscan /home/ubuntu/malware
  sudo clamscan /home/ubuntu/malware

5) Quick verification commands

- Watch pfsense log: sudo tail -f /var/log/pfsense.log
- Wazuh alerts log: sudo tail -n 200 /var/ossec/logs/alerts/alerts.json

6) Network Topology and Configurations

- We want: All traffic from Kali → pfSense LAN → pfSense WAN → Router → Internet.
  That way pfSense can filter traffic, log, and send logs to Wazuh.

- pfSense VM:
  - o Adapter 1 = Bridged (WAN) → gets IP from router (DHCP keep as is).
  - o Adapter 2 = Internal Network (LAN) → keep (192.168.0.1/24).

For 2 , we go to pfsense VM > option 2 > setup static LAN ip v4 to 192.168.0.1/24 (follow instructions)

- Kali VM:
  o Adapter 1 = Internal Network Adapter (same as pfSense LAN).
  o Remove Bridged (or disable).

For 1, after setup adapter in VBox > in kali linux

- sudo nano /etc/network/interfaces
- add this to file ,

```
auto eth0
iface eth0 inet static
 address 192.168.0.101
 netmask 255.255.255.0
 gateway 192.168.0.1
 dns-nameservers 8.8.8.8
```

Save.

- sudo systemctl restart networking

- Wazuh VM:
  o Adapter 1 = Internal Network Adapter (same as pfSense LAN).
  o If Wazuh needs Internet → you can add second adapter = Bridged/NAT (optional).
  But no need for me .

For 1 , in wazuh VM

- sudo nano /etc/sysconfig/network-scripts/ifcfg-eth0
- remove default and add this .

```
DEVICE=eth0
BOOTPROTO=static
ONBOOT=yes
TYPE=Ethernet
IPADDR=192.168.0.100
NETMASK=255.255.255.0
GATEWAY=192.168.0.1
DNS1=8.8.8.8
DNS2=1.1.1.1
```

Save .

➢ sudo systemctl restart network

7) For Ubuntu The Network Files For static ip is different , so setting up ,

i. Using "ip a" to see adapter name (enp0s3).

ii. Replace the file (sudo gedit /etc/netplan/50-cloud-init.yaml) content with below content.

```
network:
 version: 2
 renderer: networkd
 ethernets:
  enp0s3:          # check with `ip a` if your NIC has a different name
   dhcp4: no
   addresses:
    - 192.168.0.103/24
   routes:
    - to: default
      via: 192.168.0.1
   nameservers:
    addresses:
     - 192.168.0.1
```

iii. sudo netplan apply.

iv. sudo systemctl restart systemd-networkd.

THE END

“This marks the completion of my SOC Analyst Internship journey With ITSOLERA.”
“Every task, every challenge, and every solution has added a new layer to my learning.”
“This is not the end, but the beginning of a stronger path in cybersecurity.”

~ *Thanks* ~