- **Description : -**

- This project demonstrates the setup of a Security Monitoring environment using **Wazuh** (Manager, Agent, and Dashboard) integrated with **Suricata IDS**.
- The goal is to detect and visualize network intrusion attempts and custom Suricata alerts within the Wazuh Dashboard.

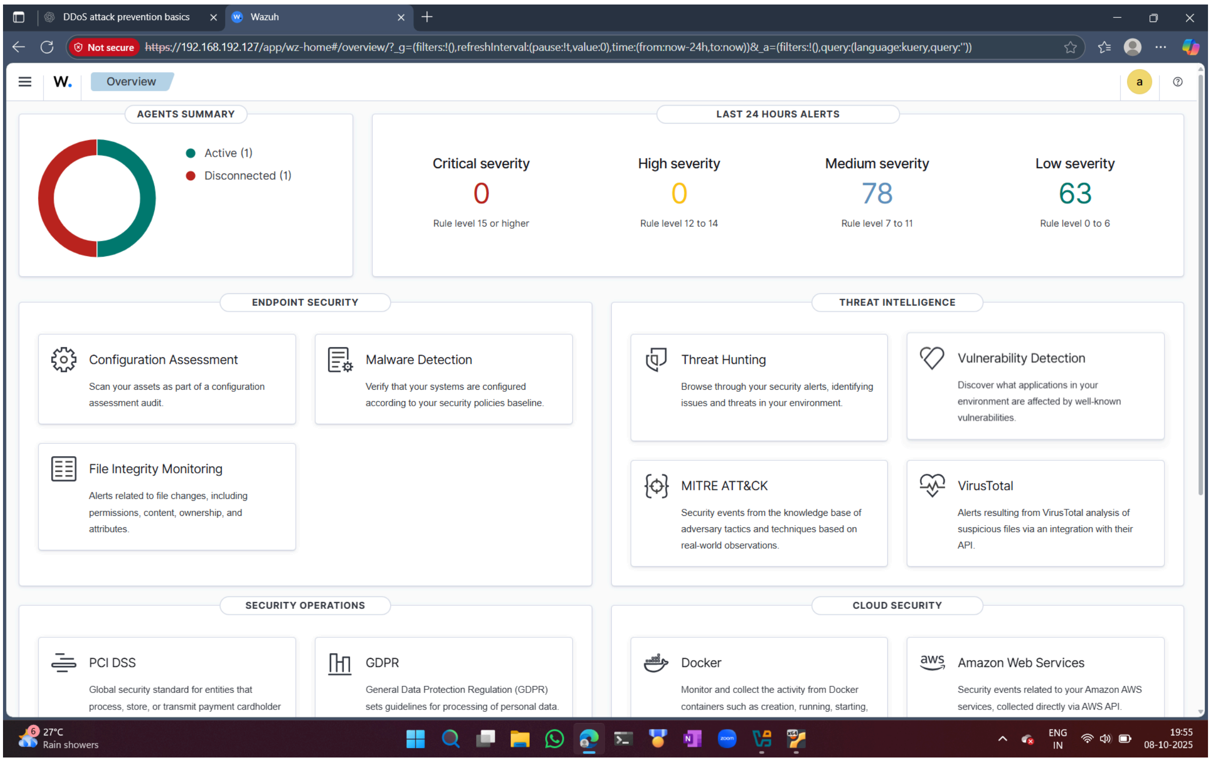
– **Environment Setup : -**

The environment consists of :

- Wazuh manager installed on Ubuntu Server (e.g., IP: 192.168.192.127)
- Wazuh dasboard installed on the same system as the manager
- Wazuh agent installed on a monitored endpoint system
- Suricata IDS installed on the agent machine to monitor traffic and generate alerts

– Installing and configuring Wazuh manager, Agent and Dashboard : -

- The Wazuh Manager is responsible for collecting, analyzing, and storing security events from all agents.
- After installation, the Wazuh Dashboard was accessed via the web interface at: https://192.168.192.127/
- The dashboard provides an overview of agent status and alerts.


DDoS attack prevention basics
Wazuh
Not secure
https://192.168.192.127/app/wz-home#/overview/?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-24h,to:now))&_a=(filters:!(),query:(language:kuery,query:''))
W.
Overview
a
27°C
Rain showers
ENG
IN
19:55
08-10-2025

AGENTS SUMMARY

● Active (1)
● Disconnected (1)

LAST 24 HOURS ALERTS

Critical severity 0 Rule level 15 or higher

High severity 0 Rule level 12 to 14

Medium severity 78 Rule level 7 to 11

Low severity 63 Rule level 0 to 6

ENDPOINT SECURITY

Configuration Assessment
Scan your assets as part of a configuration assessment audit.

Malware Detection
Verify that your systems are configured according to your security policies baseline.

File Integrity Monitoring
Alerts related to file changes, including permissions, content, ownership, and attributes.

THREAT INTELLIGENCE

Threat Hunting
Browse through your security alerts, identifying issues and threats in your environment.

Vulnerability Detection
Discover what applications in your environment are affected by well-known vulnerabilities.

MITRE ATT&CK
Security events from the knowledge base of adversary tactics and techniques based on real-world observations.

VirusTotal
Alerts resulting from VirusTotal analysis of suspicious files via an integration with their API.

SECURITY OPERATIONS

PCI DSS
Global security standard for entities that process, store, or transmit payment cardholder

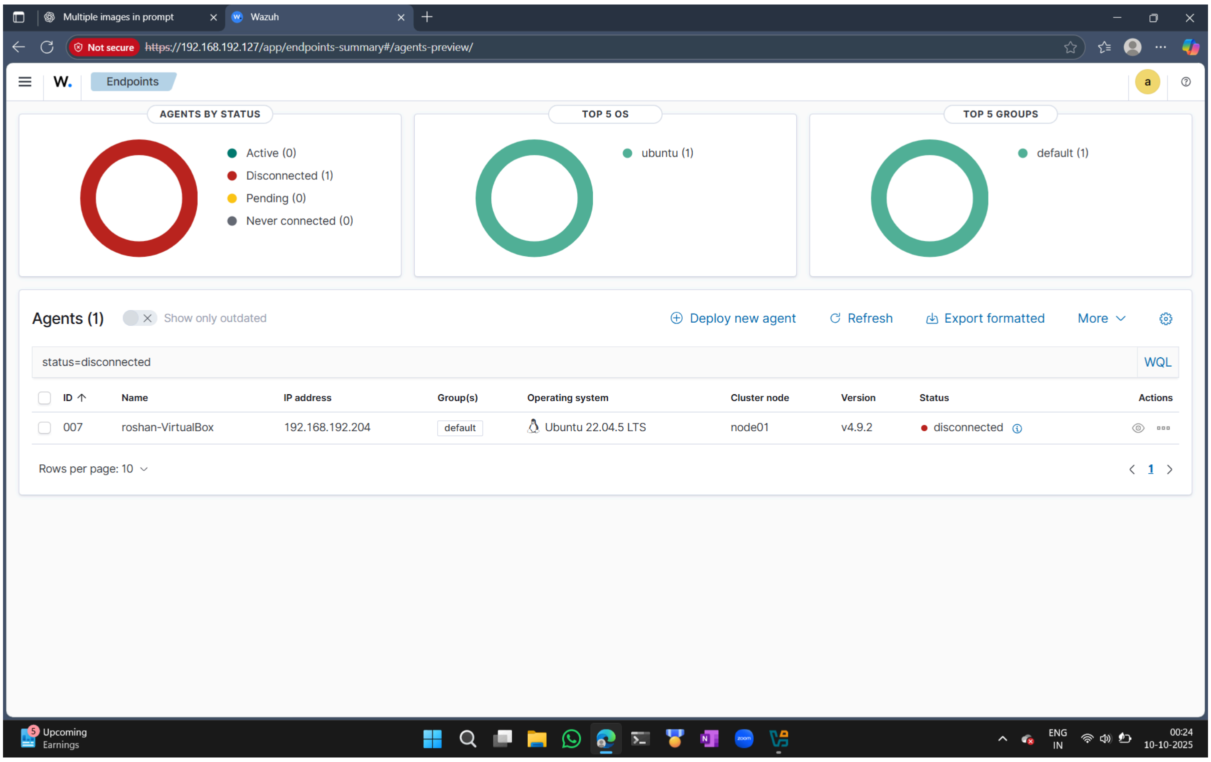
GDPR
General Data Protection Regulation (GDPR) sets guidelines for processing of personal data.

CLOUD SECURITY

Docker
Monitor and collect the activity from Docker containers such as creation, running, starting,

Amazon Web Services
Security events related to your Amazon AWS services, collected directly via AWS API.

Figure 1 — Wazuh Dashboard Overview showing connected agents and alert summary.



AGENTS BY STATUS

- ● Active (0)
- ● Disconnected (1)
- ● Pending (0)
- ● Never connected (0)

TOP 5 OS

- ● ubuntu (1)

TOP 5 GROUPS

- ● default (1)

Agents (1) Show only outdated | Deploy new agent | Refresh | Export formatted | More

status=disconnected WQL

ID ↑	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
007	roshan-VirtualBox	192.168.192.204	default	Ubuntu 22.04.5 LTS	node01	v4.9.2	● disconnected	

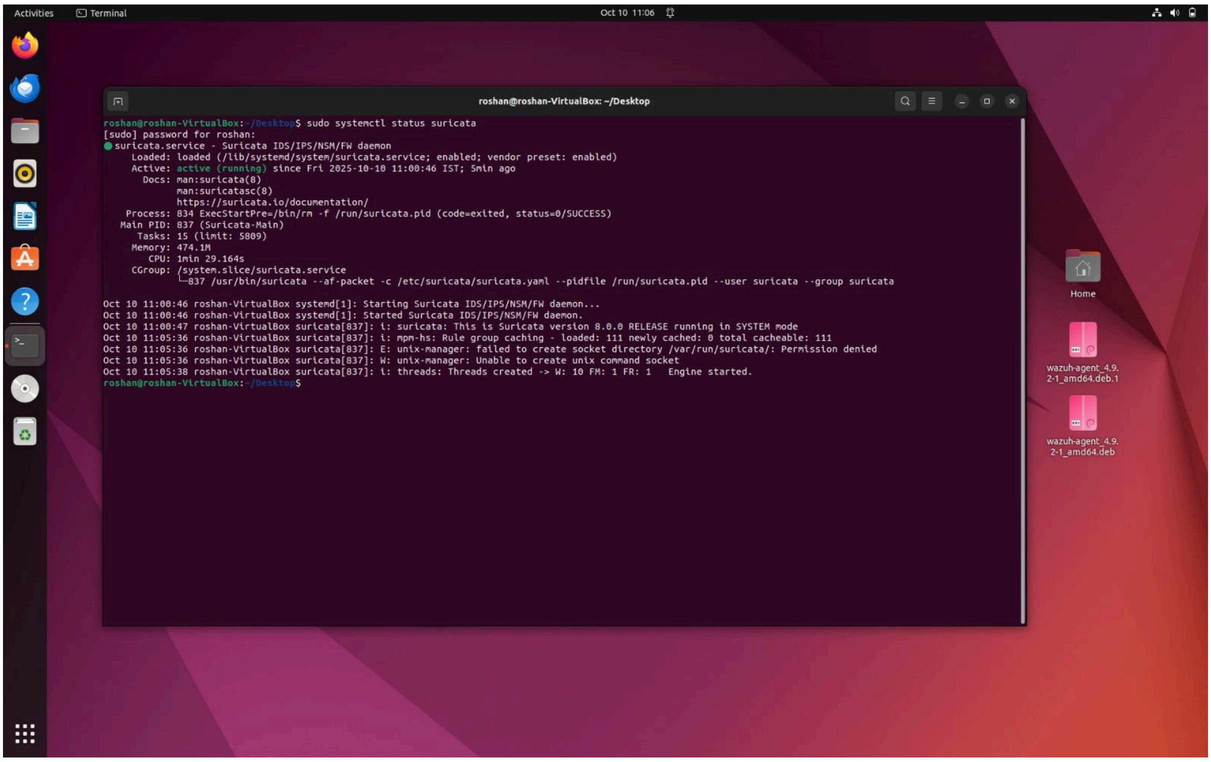
Rows per page: 10 ‹ 1 ›

Figure 2 — Agent summary showing connected Wazuh agent.

- The Wazuh Agent was installed on a monitored endpoint and successfully connected to the Wazuh Manager.
- This screenshot shows one active agent that was disconnected agent in the system.

– Installing and Configuring Suricata : -

- Suricata was installed on the endpoint where the Wazuh agent is running.
- It was configured to monitor network traffic and generate alerts stored in eve.json.
- The configuration file used: /etc/suricata/suricata.yaml


Activities
Terminal
Oct 10 11:06
roshan@roshan-VirtualBox: ~/Desktop
Home
wazuh-agent_4.9.2-1_amd64.deb.1
wazuh-agent_4.9.2-1_amd64.deb

```
roshan@roshan-VirtualBox:~/Desktop$ sudo systemctl status suricata
[sudo] password for roshan:
● suricata.service - Suricata IDS/IPS/NSM/FW daemon
     Loaded: loaded (/lib/systemd/system/suricata.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2025-10-10 11:00:46 IST; 5min ago
       Docs: man:suricata(8)
             man:suricatasc(8)
             https://suricata.io/documentation/
    Process: 834 ExecStartPre=/bin/rm -f /run/suricata.pid (code=exited, status=0/SUCCESS)
   Main PID: 837 (Suricata-Main)
      Tasks: 15 (limit: 5809)
     Memory: 474.1M
        CPU: 1min 29.164s
     CGroup: /system.slice/suricata.service
             └─837 /usr/bin/suricata --af-packet -c /etc/suricata/suricata.yaml --pidfile /run/suricata.pid --user suricata --group suricata

Oct 10 11:00:46 roshan-VirtualBox systemd[1]: Starting Suricata IDS/IPS/NSM/FW daemon...
Oct 10 11:00:46 roshan-VirtualBox systemd[1]: Started Suricata IDS/IPS/NSM/FW daemon.
Oct 10 11:00:47 roshan-VirtualBox suricata[837]: i: suricata: This is Suricata version 8.0.0 RELEASE running in SYSTEM mode
Oct 10 11:05:36 roshan-VirtualBox suricata[837]: i: mpm-hs: Rule group caching - loaded: 111 newly cached: 0 total cacheable: 111
Oct 10 11:05:36 roshan-VirtualBox suricata[837]: E: unix-manager: failed to create socket directory /var/run/suricata/: Permission denied
Oct 10 11:05:36 roshan-VirtualBox suricata[837]: W: unix-manager: Unable to create unix command socket
Oct 10 11:05:38 roshan-VirtualBox suricata[837]: i: threads: Threads created -> W: 10 FM: 1 FR: 1   Engine started.
roshan@roshan-VirtualBox:~/Desktop$
```

Figure 3 — Suricata installation and service status.

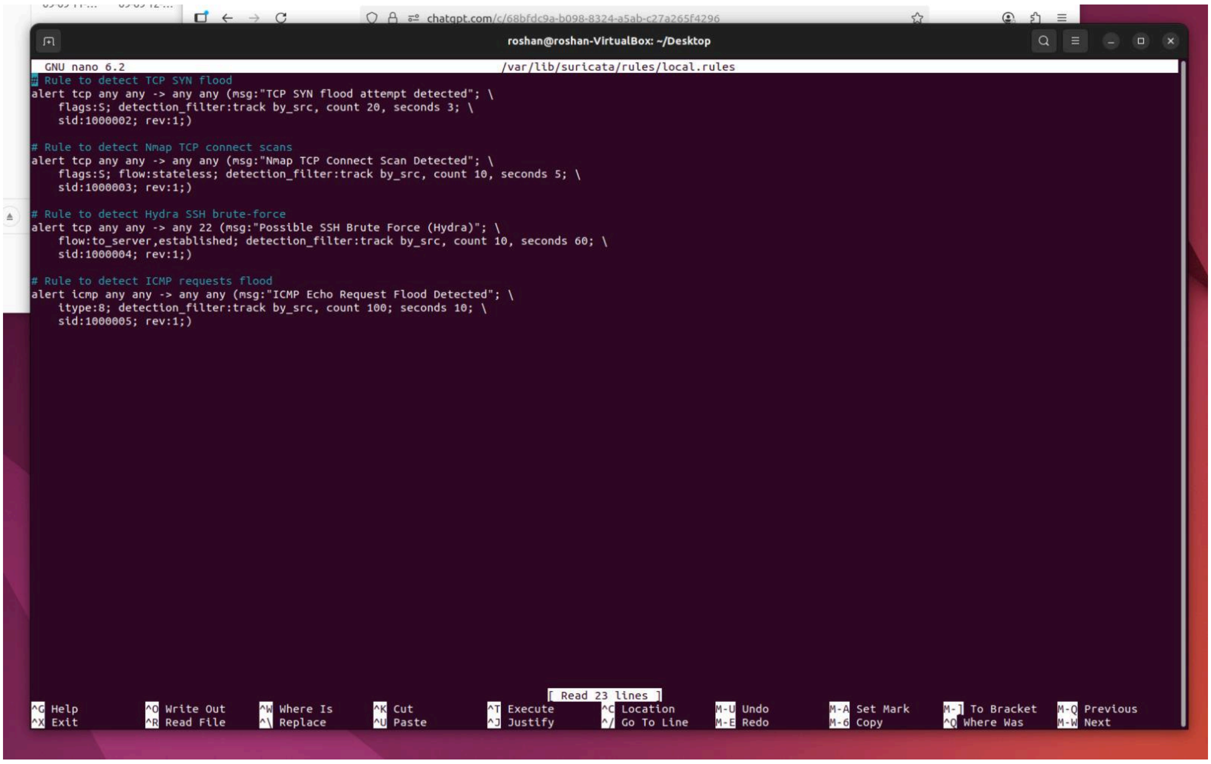
– Writing and Loading Custom Suricata rules : -

- Custom Suricata rules were written inside /etc/suricata/rules/local.rules.
- These rules are designed to detect specific test attacks (e.g., ICMP, SSH brute force, HTTP SQL injection, etc.).

- Example custom rule :

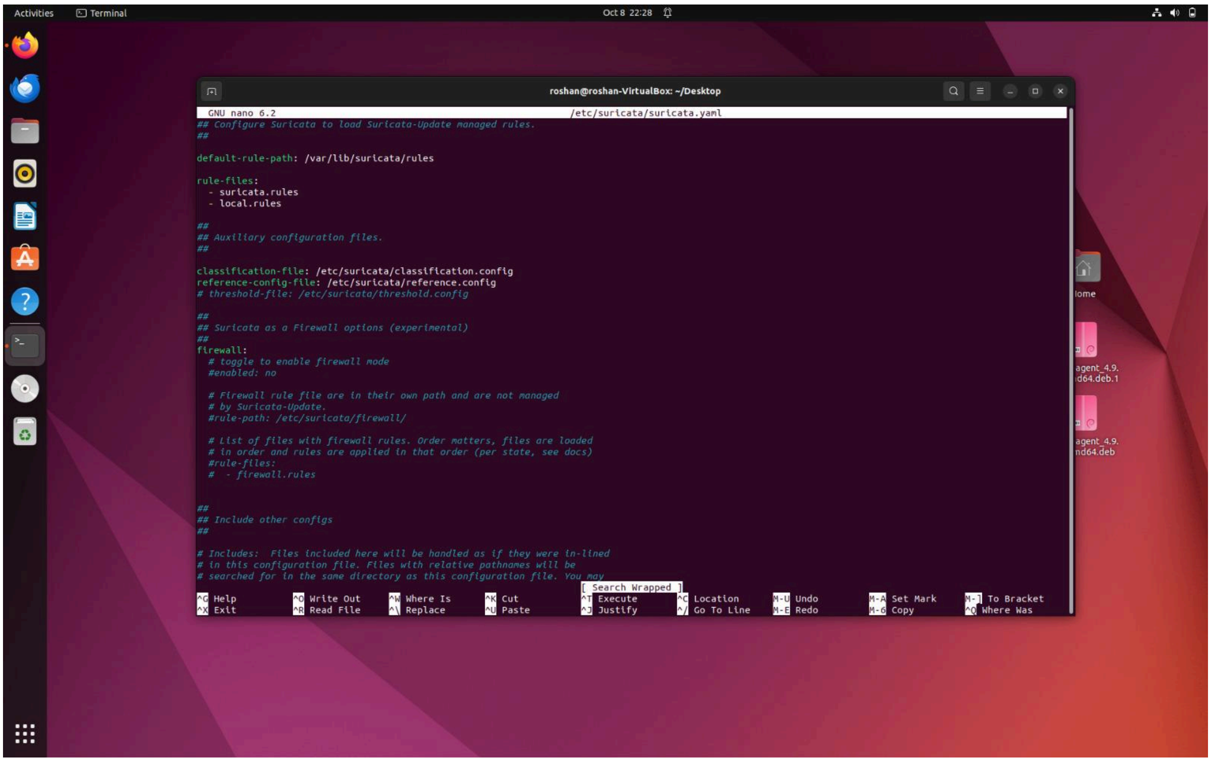
  alert icmp any any -> any any (msg:"ICMP Echo Request Detected"; sid:1000001; rev:1;)

- The rule file was configured under the rule-files section of /etc/suricata/suricata.yaml for Suricata to load it.


chatgpt.com/c/68bfdc9a-b098-8324-a5ab-c27a265f4296
roshan@roshan-VirtualBox: ~/Desktop

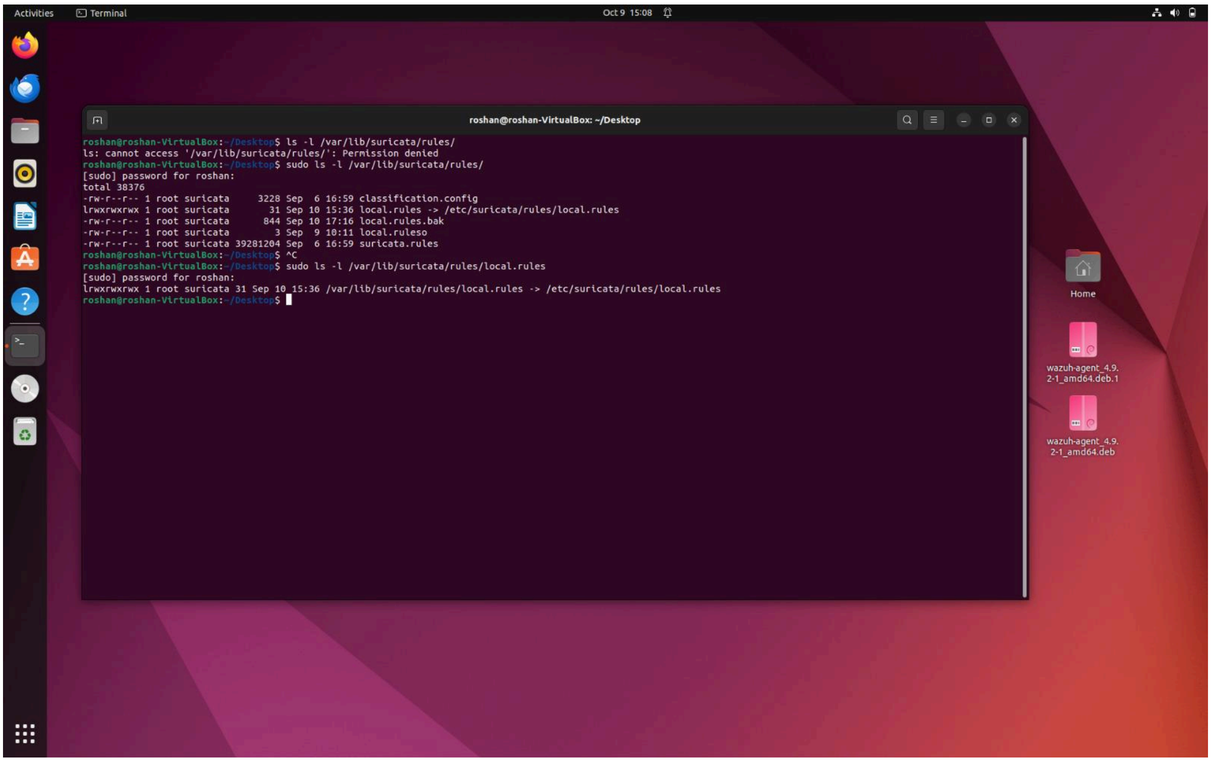
```
  GNU nano 6.2                                   /var/lib/suricata/rules/local.rules
# Rule to detect TCP SYN flood
alert tcp any any -> any any (msg:"TCP SYN flood attempt detected"; \
    flags:S; detection_filter:track by_src, count 20, seconds 3; \
    sid:1000002; rev:1;)

# Rule to detect Nmap TCP connect scans
alert tcp any any -> any any (msg:"Nmap TCP Connect Scan Detected"; \
    flags:S; flow:stateless; detection_filter:track by_src, count 10, seconds 5; \
    sid:1000003; rev:1;)

# Rule to detect Hydra SSH brute-force
alert tcp any any -> any 22 (msg:"Possible SSH Brute Force (Hydra)"; \
    flow:to_server,established; detection_filter:track by_src, count 10, seconds 60; \
    sid:1000004; rev:1;)

# Rule to detect ICMP requests flood
alert icmp any any -> any any (msg:"ICMP Echo Request Flood Detected"; \
    itype:8; detection_filter:track by_src, count 100; seconds 10; \
    sid:1000005; rev:1;)

                                   [ Read 23 lines ]
^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket M-Q Previous
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was  M-W Next
```

- These are the Suricata Custom rules that has to be written only in the file /etc/suricata/rules/local.rules
- The Suricata default in-built rules are stored in the directory /var/lib/suricata/rules This directory usually contains the default/system-installed Suricata rules, often from the community rulesets like Emerging threats.



```
  GNU nano 6.2                                   /etc/suricata/suricata.yaml
## Configure Suricata to load Suricata-Update managed rules.
##

default-rule-path: /var/lib/suricata/rules

rule-files:
  - suricata.rules
  - local.rules

##
## Auxiliary configuration files.
##

classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config
# threshold-file: /etc/suricata/threshold.config

##
## Suricata as a Firewall options (experimental)
##
firewall:
  # toggle to enable firewall mode
  #enabled: no

  # Firewall rule file are in their own path and are not managed
  # by Suricata-Update.
  #rule-path: /etc/suricata/firewall/

  # List of files with firewall rules. Order matters, files are loaded
  # in order and rules are applied in that order (per state, see docs)
  #rule-files:
  #  - firewall.rules


##
## Include other configs
##

# Includes:  Files included here will be handled as if they were in-lined
# in this configuration file. Files with relative pathnames will be
# searched for in the same directory as this configuration file. You may
                                            [ Search Wrapped ]
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy       ^Q Where Was
```

- Here the default rule-path given is /var/lib/suricata/rules in the suricata configuration file /etc/suricata/suricata.yaml
- Here Suricata does not load the suricata custom rules because custom rules file path /etc/suricata/rules/local.rules is not configured under the rule-files section in the suricata configuration file.
- If we write the suricata custom rules in the file /var/lib/suricata/rules/local.rules it may cause overwritten problem with the file /var/lib/suricata/rules/local.rules when we run suricata-update tool.
- When you run sudo suricata-update, Suricata fetches new rules and may recreate or replace /var/lib/suricata/rules/local.rules with a blank or default version.
- **Issues caused during overwritten problem :**
  – Custom rules lost **-** your manually created rules (e.g., ICMP, SSH brute force, SQLi) get replaced or deleted.
  – Alerts stop generating **-** since suricata no longer loads your custom rules, you stop getting alerts for your test attacks.
- So, to overcome this we may “Symbolic link” which is also known as symlink between the files /var/lib/suricata/rules/local.rules and /etc/suricata/rules/local.rules
- A symlink is like a *shortcut* or *pointer* in Linux.
- It links one file or directory to another — so if a program looks for a file in one location, it can actually be read from another location transparently.
- Otherwise, configure the direct suricata custom rules file as /etc/suricata/rules/lcoal.rules under the rule-files section in the suricata configuration file suircata.yaml



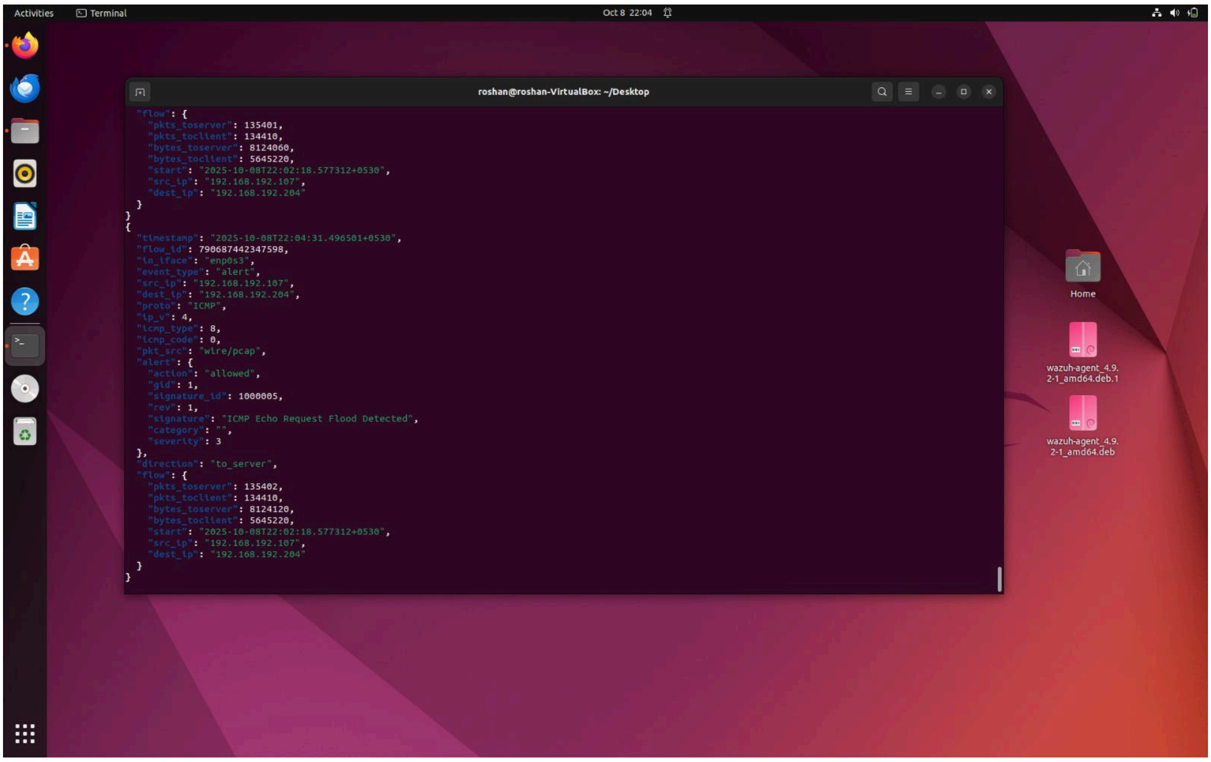
Activities
Terminal
Oct 9 15:08
roshan@roshan-VirtualBox: ~/Desktop
Home
wazuh-agent_4.9.2-1_amd64.deb.1
wazuh-agent_4.9.2-1_amd64.deb

```
roshan@roshan-VirtualBox:~/Desktop$ ls -l /var/lib/suricata/rules/
ls: cannot access '/var/lib/suricata/rules/': Permission denied
roshan@roshan-VirtualBox:~/Desktop$ sudo ls -l /var/lib/suricata/rules/
[sudo] password for roshan:
total 38376
-rw-r--r-- 1 root suricata     3228 Sep  6 16:59 classification.config
lrwxrwxrwx 1 root suricata       31 Sep 10 15:36 local.rules -> /etc/suricata/rules/local.rules
-rw-r--r-- 1 root suricata      844 Sep 10 17:16 local.rules.bak
-rw-r--r-- 1 root suricata        3 Sep  9 10:11 local.ruleso
-rw-r--r-- 1 root suricata 39281204 Sep  6 16:59 suricata.rules
roshan@roshan-VirtualBox:~/Desktop$ ^C
roshan@roshan-VirtualBox:~/Desktop$ sudo ls -l /var/lib/suricata/rules/local.rules
[sudo] password for roshan:
lrwxrwxrwx 1 root suricata 31 Sep 10 15:36 /var/lib/suricata/rules/local.rules -> /etc/suricata/rules/local.rules
roshan@roshan-VirtualBox:~/Desktop$ 
```

Figure 5 — Symlink showing link between default and custom rule directories.

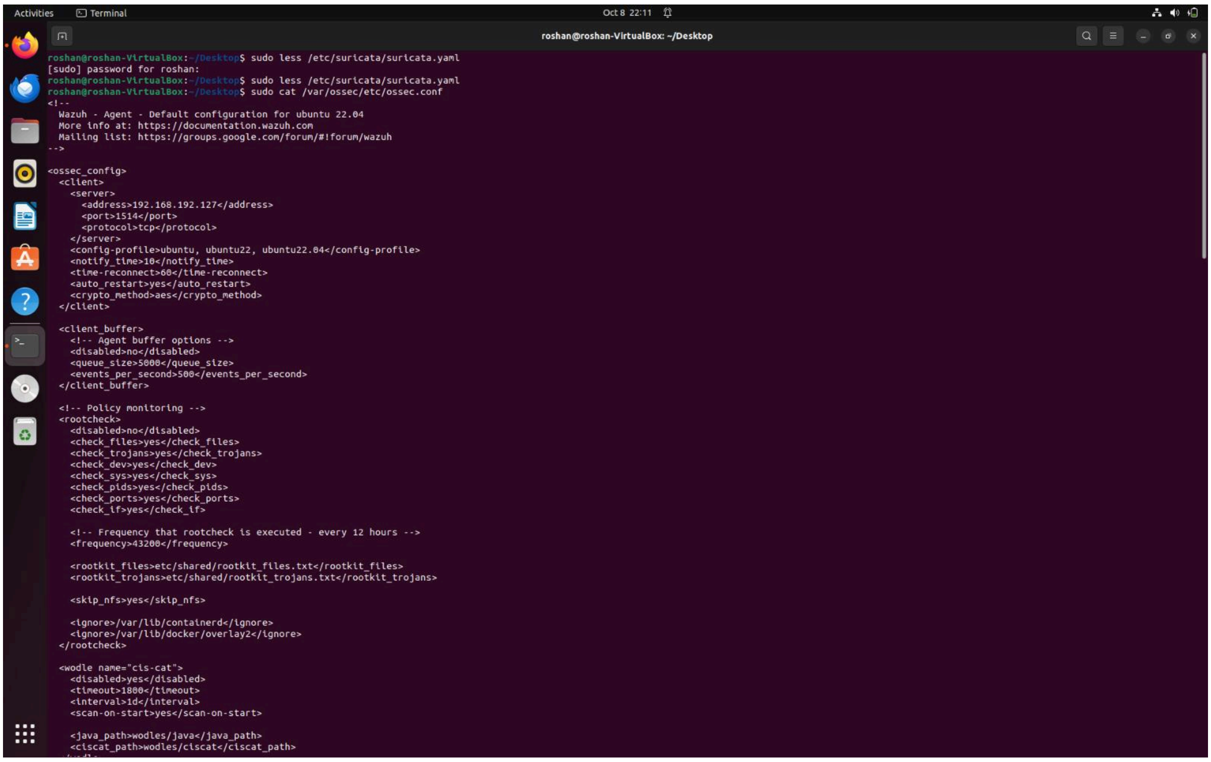
– Triggering and Capturing Alerts : -

- To test the Suricata setup, sample attacks were simulated (such as ICMP ping or TCP port scans) against the monitored host.
- These actions triggered alerts that were logged in /var/log/suricata/eve.json.
- To read the logs in human readable format we install the tool ‘jq’.


Activities
Terminal
Oct 8 22:04
roshan@roshan-VirtualBox: ~/Desktop
Home
wazuh-agent_4.9.2-1_amd64.deb.1
wazuh-agent_4.9.2-1_amd64.deb

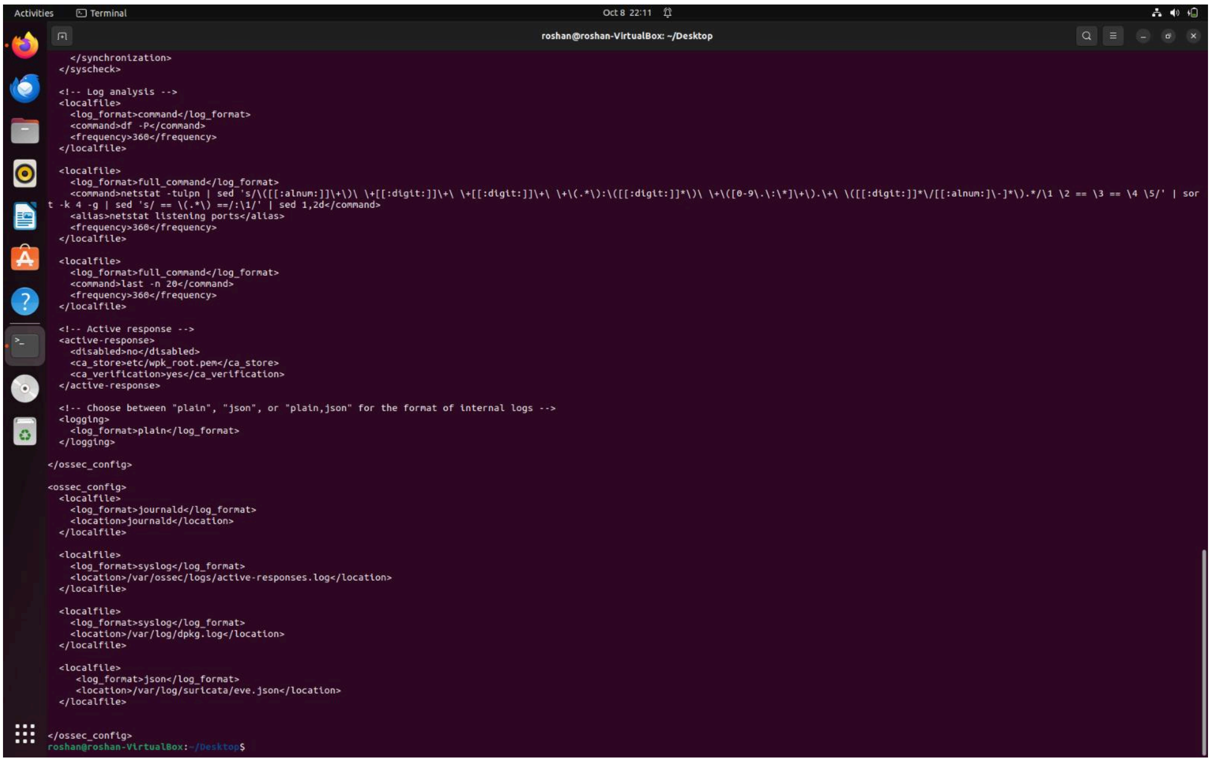
```
  "flow": {
    "pkts_toserver": 135401,
    "pkts_toclient": 134410,
    "bytes_toserver": 8124060,
    "bytes_toclient": 5645220,
    "start": "2025-10-08T22:02:18.577312+0530",
    "src_ip": "192.168.192.107",
    "dest_ip": "192.168.192.204"
  }
}
{
  "timestamp": "2025-10-08T22:04:31.496501+0530",
  "flow_id": 790687442347598,
  "in_iface": "enp0s3",
  "event_type": "alert",
  "src_ip": "192.168.192.107",
  "dest_ip": "192.168.192.204",
  "proto": "ICMP",
  "ip_v": 4,
  "icmp_type": 8,
  "icmp_code": 0,
  "pkt_src": "wire/pcap",
  "alert": {
    "action": "allowed",
    "gid": 1,
    "signature_id": 1000005,
    "rev": 1,
    "signature": "ICMP Echo Request Flood Detected",
    "category": "",
    "severity": 3
  },
  "direction": "to_server",
  "flow": {
    "pkts_toserver": 135402,
    "pkts_toclient": 134410,
    "bytes_toserver": 8124120,
    "bytes_toclient": 5645220,
    "start": "2025-10-08T22:02:18.577312+0530",
    "src_ip": "192.168.192.107",
    "dest_ip": "192.168.192.204"
  }
}
```

Figure 6 — Test attack traffic generated to trigger Suricata rule.



```
roshan@roshan-VirtualBox:~/Desktop$ sudo less /etc/suricata/suricata.yaml
[sudo] password for roshan:
roshan@roshan-VirtualBox:~/Desktop$ sudo less /etc/suricata/suricata.yaml
roshan@roshan-VirtualBox:~/Desktop$ sudo cat /var/ossec/etc/ossec.conf
<!--
  Wazuh - Agent - Default configuration for ubuntu 22.04
  More info at: https://documentation.wazuh.com
  Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
  <client>
    <server>
      <address>192.168.192.127</address>
      <port>1514</port>
      <protocol>tcp</protocol>
    </server>
    <config-profile>ubuntu, ubuntu22, ubuntu22.04</config-profile>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
    <crypto_method>aes</crypto_method>
  </client>

  <client_buffer>
    <!-- Agent buffer options -->
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
  </client_buffer>

  <!-- Policy monitoring -->
  <rootcheck>
    <disabled>no</disabled>
    <check_files>yes</check_files>
    <check_trojans>yes</check_trojans>
    <check_dev>yes</check_dev>
    <check_sys>yes</check_sys>
    <check_pids>yes</check_pids>
    <check_ports>yes</check_ports>
    <check_if>yes</check_if>

    <!-- Frequency that rootcheck is executed - every 12 hours -->
    <frequency>43200</frequency>

    <rootkit_files>etc/shared/rootkit_files.txt</rootkit_files>
    <rootkit_trojans>etc/shared/rootkit_trojans.txt</rootkit_trojans>

    <skip_nfs>yes</skip_nfs>

    <ignore>/var/lib/containerd</ignore>
    <ignore>/var/lib/docker/overlay2</ignore>
  </rootcheck>

  <wodle name="cis-cat">
    <disabled>yes</disabled>
    <timeout>1800</timeout>
    <interval>1d</interval>
    <scan-on-start>yes</scan-on-start>

    <java_path>wodles/java</java_path>
    <ciscat_path>wodles/ciscat</ciscat_path>
```

Figure 7 — Wauzh agent Configuration file send lpgs to the Wazuh Manager IP Address 192.168.192.127

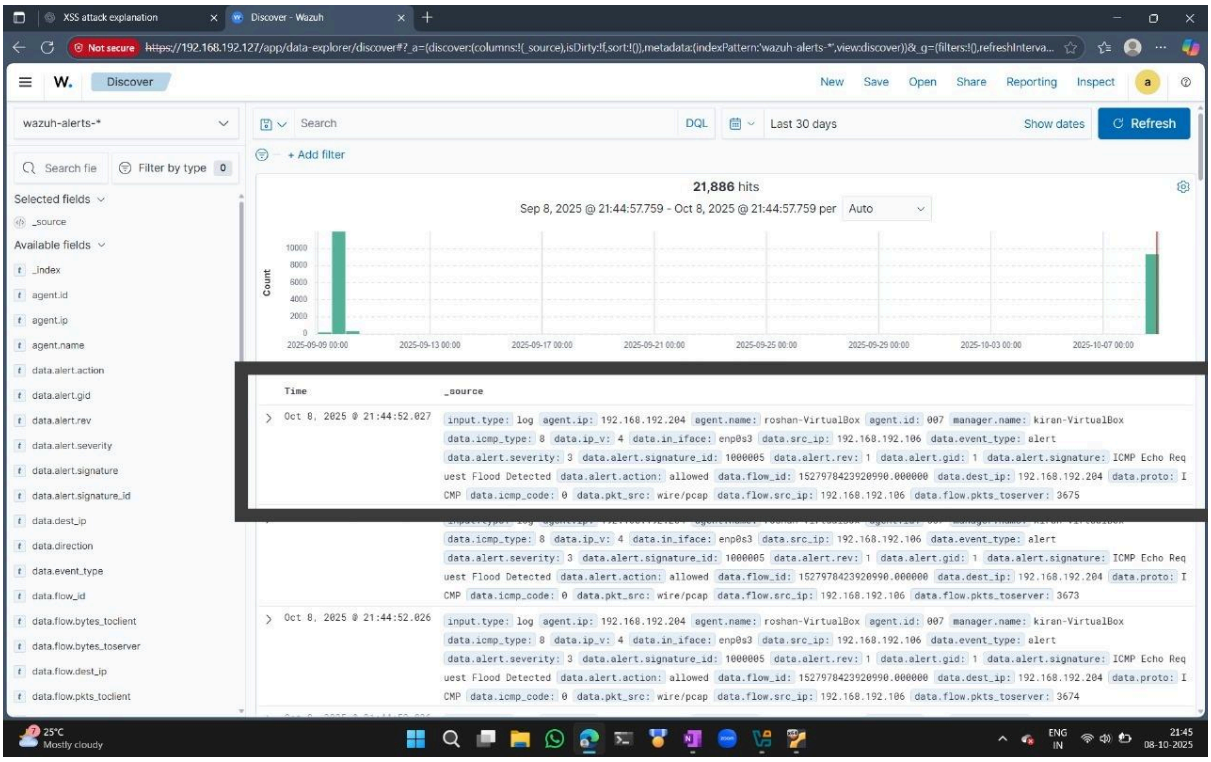


```
    </synchronization>
  </syscheck>

  <!-- Log analysis -->
  <localfile>
    <log_format>command</log_format>
    <command>df -P</command>
    <frequency>360</frequency>
  </localfile>

  <localfile>
    <log_format>full_command</log_format>
    <command>netstat -tulpn | sed 's/\([[:alnum:]]\+\)\ \+[[:digit:]]\+\ \+[[:digit:]]\+\ \+\(.*\):\([[:digit:]]*\)\ \+\([0-9\.\:\*]\+\).\+\ \([[:digit:]]*\/[[:alnum:]\-]*\).*/\1 \2 == \3 == \4 \5/' | sor
t -k 4 -g | sed 's/ == \(.*\) ==/:\1/' | sed 1,2d</command>
    <alias>netstat listening ports</alias>
    <frequency>360</frequency>
  </localfile>

  <localfile>
    <log_format>full_command</log_format>
    <command>last -n 20</command>
    <frequency>360</frequency>
  </localfile>

  <!-- Active response -->
  <active-response>
    <disabled>no</disabled>
    <ca_store>etc/wpk_root.pem</ca_store>
    <ca_verification>yes</ca_verification>
  </active-response>

  <!-- Choose between "plain", "json", or "plain,json" for the format of internal logs -->
  <logging>
    <log_format>plain</log_format>
  </logging>

</ossec_config>

<ossec_config>
  <localfile>
    <log_format>journald</log_format>
    <location>journald</location>
  </localfile>

  <localfile>
    <log_format>syslog</log_format>
    <location>/var/ossec/logs/active-responses.log</location>
  </localfile>

  <localfile>
    <log_format>syslog</log_format>
    <location>/var/log/dpkg.log</location>
  </localfile>

  <localfile>
     <log_format>json</log_format>
     <location>/var/log/suricata/eve.json</location>
  </localfile>


</ossec_config>
roshan@roshan-VirtualBox:~/Desktop$
```

Figure 8 — Wazuh agent stores different formats of logs in different file locations and

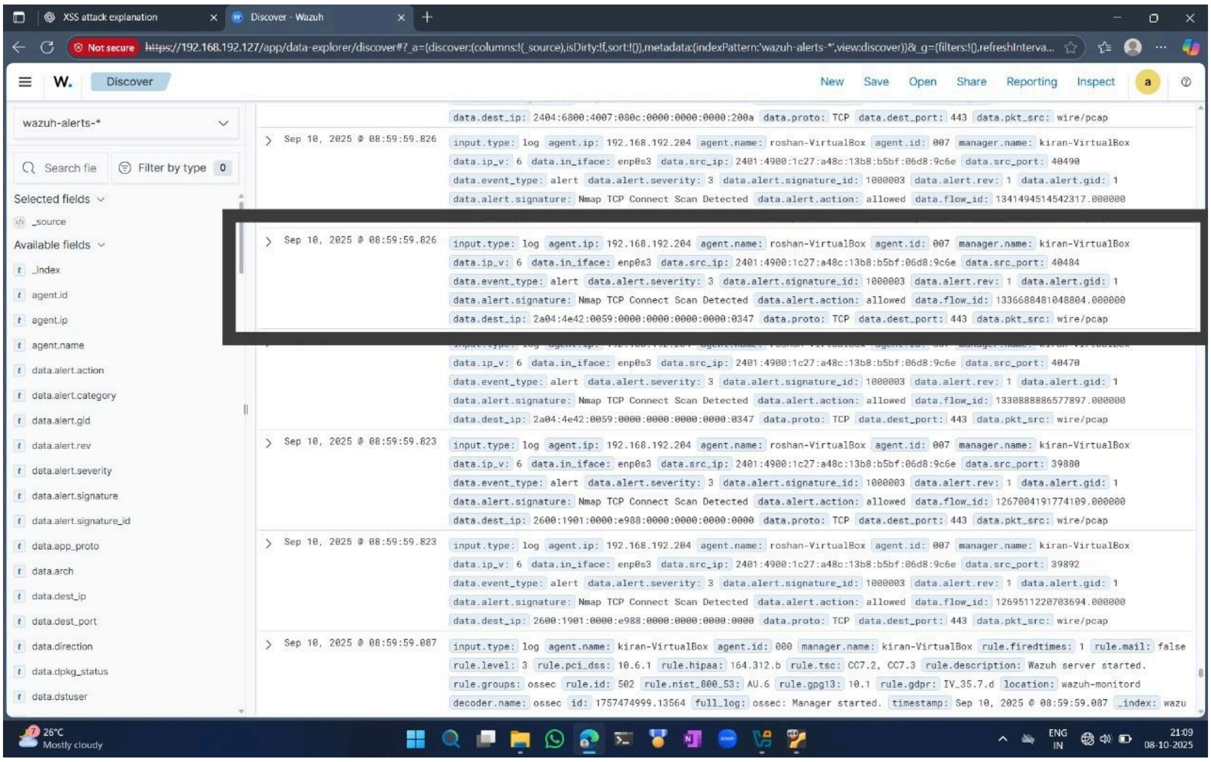
– Viewing Alerts in Wazuh Dashboard : -

- The alerts generated by Suricata were successfully forwarded to the Wazuh Manager via the Wazuh Agent.
- Wazuh parsed the Suricata logs and displayed the corresponding alerts in the Dashboard under *Security events* → *Suricata*.
- The alert contains fields such as timestamp, source/destination IPs, protocol, and custom message.



```
Time                         _source
> Oct 8, 2025 @ 21:44:52.027 input.type: log agent.ip: 192.168.192.204 agent.name: roshan-VirtualBox agent.id: 007 manager.name: kiran-VirtualBox
                             data.icmp_type: 8 data.ip_v: 4 data.in_iface: enp0s3 data.src_ip: 192.168.192.106 data.event_type: alert
                             data.alert.severity: 3 data.alert.signature_id: 1000005 data.alert.rev: 1 data.alert.gid: 1 data.alert.signature: ICMP Echo Req
                             uest Flood Detected data.alert.action: allowed data.flow_id: 1527978423920990.000000 data.dest_ip: 192.168.192.204 data.proto: I
                             CMP data.icmp_code: 0 data.pkt_src: wire/pcap data.flow.src_ip: 192.168.192.106 data.flow.pkts_toserver: 3675

                             data.icmp_type: 8 data.ip_v: 4 data.in_iface: enp0s3 data.src_ip: 192.168.192.106 data.event_type: alert
                             data.alert.severity: 3 data.alert.signature_id: 1000005 data.alert.rev: 1 data.alert.gid: 1 data.alert.signature: ICMP Echo Req
                             uest Flood Detected data.alert.action: allowed data.flow_id: 1527978423920990.000000 data.dest_ip: 192.168.192.204 data.proto: I
                             CMP data.icmp_code: 0 data.pkt_src: wire/pcap data.flow.src_ip: 192.168.192.106 data.flow.pkts_toserver: 3673
> Oct 8, 2025 @ 21:44:52.026 input.type: log agent.ip: 192.168.192.204 agent.name: roshan-VirtualBox agent.id: 007 manager.name: kiran-VirtualBox
                             data.icmp_type: 8 data.ip_v: 4 data.in_iface: enp0s3 data.src_ip: 192.168.192.106 data.event_type: alert
                             data.alert.severity: 3 data.alert.signature_id: 1000005 data.alert.rev: 1 data.alert.gid: 1 data.alert.signature: ICMP Echo Req
                             uest Flood Detected data.alert.action: allowed data.flow_id: 1527978423920990.000000 data.dest_ip: 192.168.192.204 data.proto: I
                             CMP data.icmp_code: 0 data.pkt_src: wire/pcap data.flow.src_ip: 192.168.192.106 data.flow.pkts_toserver: 3674
```

Figure 9 — ICMP Flood Suricata rule alerts successfully displayed in Wazuh Dashboard.



```
                             data.dest_ip: 2404:6800:4007:080c:0000:0000:0000:200a data.proto: TCP data.dest_port: 443 data.pkt_src: wire/pcap
> Sep 10, 2025 @ 08:59:59.826 input.type: log agent.ip: 192.168.192.204 agent.name: roshan-VirtualBox agent.id: 007 manager.name: kiran-VirtualBox
                             data.ip_v: 6 data.in_iface: enp0s3 data.src_ip: 2401:4900:1c27:a48c:13b8:b5bf:06d8:9c6e data.src_port: 40490
                             data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000003 data.alert.rev: 1 data.alert.gid: 1
                             data.alert.signature: Nmap TCP Connect Scan Detected data.alert.action: allowed data.flow_id: 1341494514542317.000000
> Sep 10, 2025 @ 08:59:59.826 input.type: log agent.ip: 192.168.192.204 agent.name: roshan-VirtualBox agent.id: 007 manager.name: kiran-VirtualBox
                             data.ip_v: 6 data.in_iface: enp0s3 data.src_ip: 2401:4900:1c27:a48c:13b8:b5bf:06d8:9c6e data.src_port: 40484
                             data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000003 data.alert.rev: 1 data.alert.gid: 1
                             data.alert.signature: Nmap TCP Connect Scan Detected data.alert.action: allowed data.flow_id: 1336688481048804.000000
                             data.dest_ip: 2a04:4e42:0059:0000:0000:0000:0000:0347 data.proto: TCP data.dest_port: 443 data.pkt_src: wire/pcap

                             data.ip_v: 6 data.in_iface: enp0s3 data.src_ip: 2401:4900:1c27:a48c:13b8:b5bf:06d8:9c6e data.src_port: 40470
                             data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000003 data.alert.rev: 1 data.alert.gid: 1
                             data.alert.signature: Nmap TCP Connect Scan Detected data.alert.action: allowed data.flow_id: 1330888886577897.000000
                             data.dest_ip: 2a04:4e42:0059:0000:0000:0000:0000:0347 data.proto: TCP data.dest_port: 443 data.pkt_src: wire/pcap
> Sep 10, 2025 @ 08:59:59.823 input.type: log agent.ip: 192.168.192.204 agent.name: roshan-VirtualBox agent.id: 007 manager.name: kiran-VirtualBox
                             data.ip_v: 6 data.in_iface: enp0s3 data.src_ip: 2401:4900:1c27:a48c:13b8:b5bf:06d8:9c6e data.src_port: 39880
                             data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000003 data.alert.rev: 1 data.alert.gid: 1
                             data.alert.signature: Nmap TCP Connect Scan Detected data.alert.action: allowed data.flow_id: 1267004191774109.000000
                             data.dest_ip: 2600:1901:0000:e988:0000:0000:0000:0000 data.proto: TCP data.dest_port: 443 data.pkt_src: wire/pcap
> Sep 10, 2025 @ 08:59:59.823 input.type: log agent.ip: 192.168.192.204 agent.name: roshan-VirtualBox agent.id: 007 manager.name: kiran-VirtualBox
                             data.ip_v: 6 data.in_iface: enp0s3 data.src_ip: 2401:4900:1c27:a48c:13b8:b5bf:06d8:9c6e data.src_port: 39892
                             data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000003 data.alert.rev: 1 data.alert.gid: 1
                             data.alert.signature: Nmap TCP Connect Scan Detected data.alert.action: allowed data.flow_id: 1269511220703694.000000
                             data.dest_ip: 2600:1901:0000:e988:0000:0000:0000:0000 data.proto: TCP data.dest_port: 443 data.pkt_src: wire/pcap
> Sep 10, 2025 @ 08:59:59.087 input.type: log agent.name: kiran-VirtualBox agent.id: 000 manager.name: kiran-VirtualBox rule.firedtimes: 1 rule.mail: false
                             rule.level: 3 rule.pci_dss: 10.6.1 rule.hipaa: 164.312.b rule.tsc: CC7.2, CC7.3 rule.description: Wazuh server started.
                             rule.groups: ossec rule.id: 502 rule.nist_800_53: AU.6 rule.gpg13: 10.1 rule.gdpr: IV_35.7.d location: wazuh-monitord
                             decoder.name: ossec id: 1757474999.13564 full_log: ossec: Manager started. timestamp: Sep 10, 2025 @ 08:59:59.087 _index: wazu
```

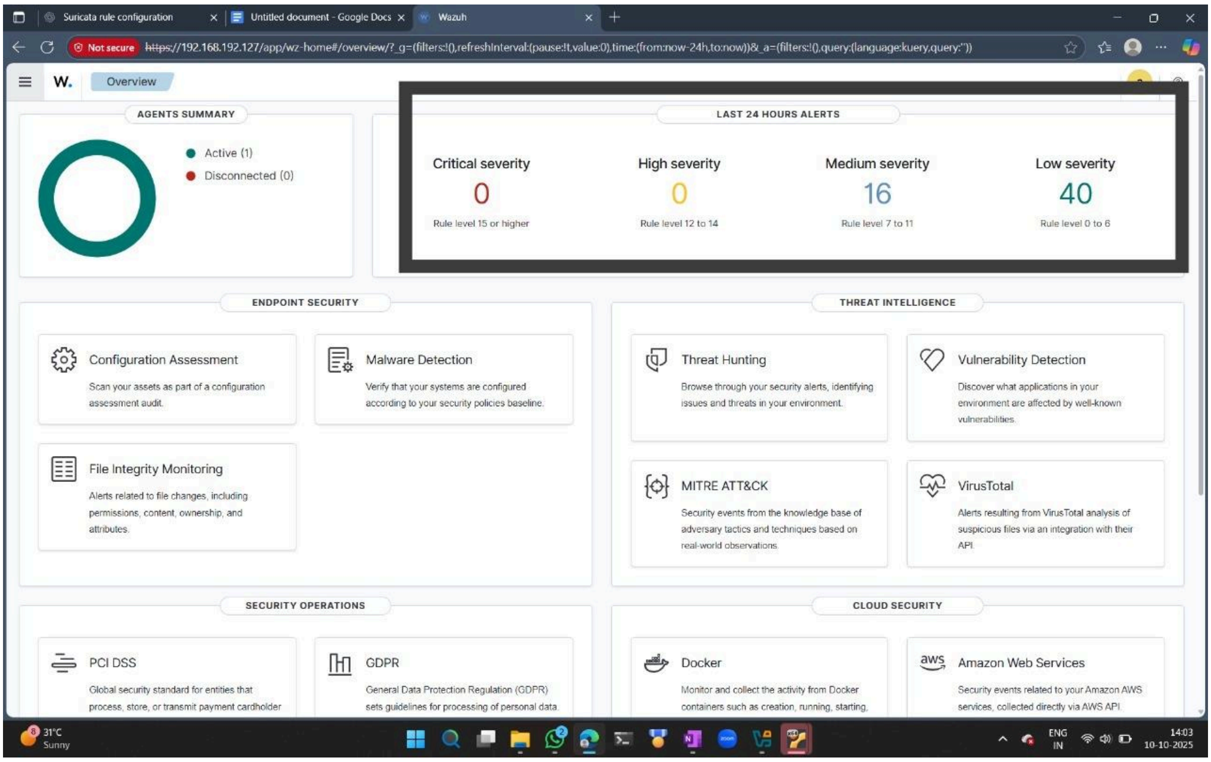
Figure 10 — Nmap Scan detection Suricata rule alerts successfully displayed in Wazuh Dashboard.

– Understanding Alerts Severities : -

Wazuh categorizes alerts based on rule levels **(0–15)** :

- 0–6 → Low severity
- 7–11 → Medium severity
- 12–14 → High severity
- 15+ → Critical severity

The dashboard displays alert counts by severity level over the last 24 hours.


Suricata rule configuration
Untitled document - Google Docs
Wazuh
Not secure
https://192.168.192.127/app/wz-home#/overview/?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-24h,to:now))&_a=(filters:!(),query:(language:kuery,query:''))
W.
Overview
31°C
Sunny
ENG
IN
14:03
10-10-2025

AGENTS SUMMARY

● Active (1)
● Disconnected (0)

LAST 24 HOURS ALERTS

Critical severity	High severity	Medium severity	Low severity
0	0	16	40
Rule level 15 or higher	Rule level 12 to 14	Rule level 7 to 11	Rule level 0 to 6

ENDPOINT SECURITY

Configuration Assessment – Scan your assets as part of a configuration assessment audit.

Malware Detection – Verify that your systems are configured according to your security policies baseline.

File Integrity Monitoring – Alerts related to file changes, including permissions, content, ownership, and attributes.

THREAT INTELLIGENCE

Threat Hunting – Browse through your security alerts, identifying issues and threats in your environment.

Vulnerability Detection – Discover what applications in your environment are affected by well-known vulnerabilities.

MITRE ATT&CK – Security events from the knowledge base of adversary tactics and techniques based on real-world observations.

VirusTotal – Alerts resulting from VirusTotal analysis of suspicious files via an integration with their API.

SECURITY OPERATIONS

PCI DSS – Global security standard for entities that process, store, or transmit payment cardholder

GDPR – General Data Protection Regulation (GDPR) sets guidelines for processing of personal data

CLOUD SECURITY

Docker – Monitor and collect the activity from Docker containers such as creation, running, starting,

Amazon Web Services – Security events related to your Amazon AWS services, collected directly via AWS API.

Figure 11 — Severity distribution of alerts in the Wazuh Dashboard.

– Threat Hunting and MITRE ATT&CK Mapping : -

- Wazuh integrates with **MITRE ATT&CK** to classify detected events based on known adversary tactics and techniques.
- This helps security analysts understand the context and purpose of detected behaviors.

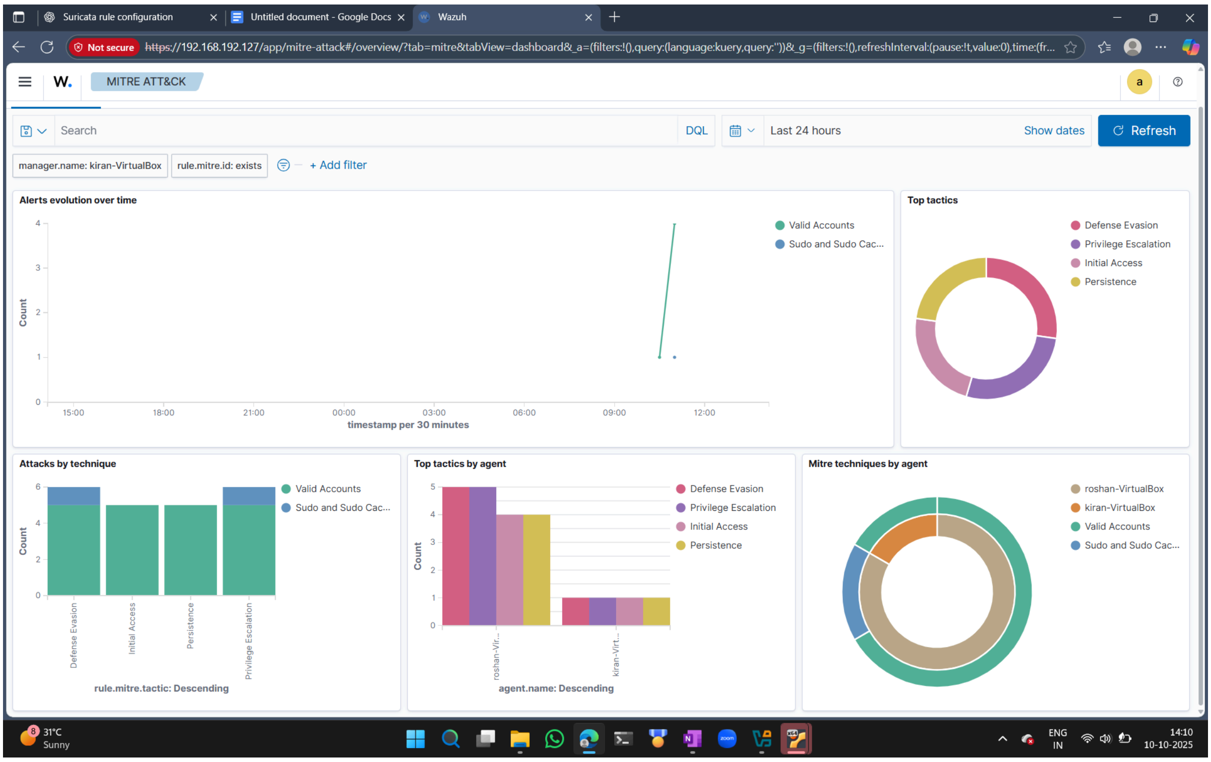


Figure 12 — MITRE ATT&CK mapping of detected threats in Wazuh.

– Conclusion : -

- This project demonstrated how Wazuh and Suricata can be integrated for real-time intrusion detection and alert visualization.
- By creating and loading custom Suricata rules, network attacks were successfully detected and displayed on the Wazuh Dashboard.
- The setup provides a complete open-source SIEM + IDS monitoring environment.