



WAZUK + ELK SOC

# OPERATION MANUAL

**PREPARED BY**

Yu Wun Chen

Meiling Xu

Shuai Jiang

**APPROVED BY**

Piyushi Singh

**DATE**

Aug 21, 2025

**Spring 2025  
Capstone Project**



---

BlackMirrorSoc.

version 1.0



## Table of Contents

1. Introduction .....	5
1.1 Project Overview and Background.....	5
1.2 Feature Summary.....	5
1.3 Purpose and Audience of this Manual.....	5
2. System Requirements .....	6
2.1 Hardware Requirements .....	6
2.2 Software Requirements.....	6
2.3 Network Requirements .....	6
3. Installation Guide .....	7
3.1 Installation Steps.....	7
Step 1: Clone the project .....	7
Step 2: Install Dependencies / Libraries.....	8
Step 3: Configure Settings Files .....	9
Step 4: Start Services / Applications .....	10
3.2 Post-installation Verification .....	13
Check if the services are running correctly.....	13
Test system connectivity .....	14
4. User Guide.....	17
4.1 Login and Main Interface Overview .....	17
Login Instructions / Default Account.....	17
Overview of Main Feature Areas .....	18
4.2 Core Function Operations .....	21
Repository 1: wazuh-suricata-elk-docker - Function 1: Wazuh detect - log-in failed.....	21
Repository 1: wazuh-suricata-elk-docker - Function 2: Wazuh detect - create a user.....	23
Repository 1: wazuh-suricata-elk-docker - Function 3: Wazuh detect - user changes the password.....	23
Repository 1: wazuh-suricata-elk-docker - Function 4: Wazuh detect - remote access alert and session opened alert.....	24
Repository 1: wazuh-suricata-elk-docker - Function 5: Wazuh detect - High privilege operation: sudo, root operation record.....	25
Repository 1: wazuh-suricata-elk-docker - Function 6: Wazuh detect - System Change: Modification or Deletion of Critical Files.....	27
Repository 1: wazuh-suricata-elk-docker - Function 7: Suricata detect - simulate ICMP/UDP/HTTP attacks with Scapy.....	28

Repository 1: wazuh-suricata-elk-docker - Function 8: Suricata detect - simulate DOS attacks .....	31
Repository 1: wazuh-suricata-elk-docker - Function 9: How to use the Kibana dashboard .....	32
Repository 2: log-alert-pipeline - Function 1: Custom Python parser filtering rules .....	46
4.3 Data Import / Export (if applicable) .....	54
Supported Import Formats .....	54
Supported Export Formats .....	54
5. Troubleshooting .....	55
5.1 Issue 1: Wazuh Dashboard cannot be accessed (Page Not Found / Connection Refused).....	55
5.2 Issue 2: No Alerts are Showing on Dashboard .....	56
5.3 Issue 3: Python Alert Parser Fails to Run.....	56
5.4 Issue 4: Exported CSV/JSON Files are Empty.....	57
5.5 Issue 5: Scheduled Fetcher Stops Running Unexpectedly.....	57
6. Additional Resources.....	58
Official Documentation Links .....	58
GitHub Repository Link and QR code.....	58
Demo Video QR Code.....	58
Contact Email: .....	58
7. References.....	59

# 1. Introduction

## 1.1 Project Overview and Background

Small-to-medium enterprises (SMEs) and educational institutions often lack access to affordable and customizable Security Operations Center (SOC) solutions for real-time threat detection, log analysis, and threat visualization. Existing solutions are either too resource-intensive, overly complex, or lack the flexibility for educational purposes. Our project addresses this gap by providing a lightweight, open-source SOC platform that integrates essential cybersecurity tools into a modular and easy-to-deploy system.

## 1.2 Feature Summary

The Plug-and-Play SOC platform offers the following key features:

- **Docker-Based Lightweight Deployment:** Simplifies installation and ensures consistent environments.
- **Open-Source Integration:** Utilizes Wazuh for host-based intrusion detection, Suricata for network threat detection, and Kibana for data visualization.
- **Custom Python Threat Analysis Module:** Enables real-time log enrichment and threat correlation.
- **Intuitive Visual Dashboards:** Provide clear, customizable visualizations for security monitoring.
- **Scalable and Portable:** Can be deployed on virtual machines or cloud environments with minimal hardware requirements.

## 1.3 Purpose and Audience of this Manual

This manual is designed to assist IT administrators, cybersecurity students, educators, and security analysts in installing, configuring, and operating the Plug-and-Play SOC platform. It serves as a comprehensive guide for:

- Setting up the system from scratch
- Understanding core functionalities
- Performing common operational tasks
- Troubleshooting potential issues

The manual ensures that both technical users and learners can independently deploy and utilize the platform for practical cybersecurity monitoring and educational purposes.

## 2. System Requirements

### 2.1 Hardware Requirements

- **CPU:** Minimum Dual-Core Processor (Recommended: Quad-Core or higher)
- **RAM:** Minimum 4GB RAM (Recommended: 8GB RAM or higher for larger log volumes)
- **Disk Space:** At least 20GB of free disk space (Recommended: SSD storage for better performance)

These specifications are for a typical deployment handling moderate log volume. For larger environments, resource allocation should scale accordingly.

### 2.2 Software Requirements

- **Operating System:** Linux-based OS (Ubuntu 24.04 LTS recommended)
- **Required Software/Frameworks:**
  - Docker Engine (version 20.10+)
  - Docker Compose (version 1.29+)
  - Python 3.8+
  - Git
  - Web Browser (Chrome, Firefox for Kibana dashboard access)

### 2.3 Network Requirements

Stable network connection for fetching updates and Docker images

#### Open ports:

- TCP 5601 (Kibana)
- TCP 9200 (Elasticsearch)
- TCP 1514 (Wazuh Agent communication)
- TCP 514 (Syslog, optional)

Ensure firewall rules allow internal network traffic between containers and external access to Kibana if remote monitoring is needed. Both inbound and outbound traffic for these ports must be configured appropriately.

## 3. Installation Guide

### 3.1 Installation Steps

#### Step 1: Clone the project

**Clone the project repository**

**Repository 1: wazuh-suricata-elk-docker**

```
git clone https://github.com/CatherineChen-CyberSecurity/wazuh-suricata-elk-docker.git
```

```
cd wazuh-suricata-elk-docker
```

**Repository 2: log-alert-pipeline**

```
git clone https://github.com/CatherineChen-CyberSecurity/log-alert-pipeline.git
```

```
cd log-alert-pipeline
```

## Step 2: Install Dependencies / Libraries

### Repository 1: wazuh-suricata-elk-docker

Install the make command

### Repository 2: log-alert-pipeline

# optional step, create a virtual environment

```
python3 -m venv .venv
```

```
source .venv/bin/activate
```

```
# install dependencies
```

```
pip install -r requirements.txt
```

```
~/.Documents/workspace/log-alert-pipeline | main | pip install -r requirements.txt
Collecting requests==2.31.0 (from -r requirements.txt (line 1))
  Downloading requests-2.31.0-py3-none-any.whl (62 kB)
Collecting schedule==1.2.0 (from -r requirements.txt (line 2))
  Downloading schedule-1.2.0-py3-none-any.whl.metadata (3.3 kB)
Collecting pyyaml==6.0.1 (from -r requirements.txt (line 3))
  Downloading PyYAML-6.0.1-cp312-cp312-manylinux_2_17_x86_64_manylinux2014_x86_64.whl.metadata (2.1 kB)
Collecting certifi==2025.7.14 (from -r requirements.txt (line 4))
  Downloading certifi-2025.7.14-py3-none-any.whl.metadata (6.0 kB)
Collecting pandas==2.3.1 (from -r requirements.txt (line 5))
  Downloading pandas-2.3.1-cp312-cp312-manylinux_2_17_x86_64_manylinux2014_x86_64.whl.metadata (91 kB)
  01.2/91.2 kB 10.7 MB/s eta 0:00:00
Collecting charset-normalizer<4,>2 (from -r requirements.txt (line 6))
  Downloading charset_normalizer-3.4.2-cp312-cp312-manylinux_2_17_x86_64_manylinux2014_x86_64.whl.metadata (35 kB)
Collecting idna==4,>2.5 (from -r requirements.txt (line 7))
  Downloading idna-4.3.10-py3-none-any.whl.metadata (10 kB)
Collecting certifi==2025.7.14 (from -r requirements.txt (line 8))
  Downloading certifi-2025.7.14-py3-none-any.whl.metadata (6.0 kB)
Collecting python-dateutil==2.31.0 (from -r requirements.txt (line 9))
  Downloading python_dateutil-2.31.0-py3-none-any.whl.metadata (62 kB)
  02.1/62.1 kB 5.2 MB/s eta 0:00:00
Collecting python-dateutil==2.9.0 (from -r requirements.txt (line 10))
  Downloading python_dateutil-2.9.0-py3-none-any.whl.metadata (8.4 kB)
Collecting tzdata==2025.1 (from -r requirements.txt (line 11))
  Downloading tzdata-2025.2-py2-py3-none-any.whl.metadata (22 kB)
Collecting tzdata==2025.2-py2-py3-none-any.whl (from -r requirements.txt (line 12))
  Downloading tzdata-2025.2-py2-py3-none-any.whl.metadata (1.4 kB)
Collecting six==1.17.3 (from -r requirements.txt (line 13))
  Downloading six-1.17.3-py2-py3-none-any.whl (1.7 kB)
Collecting requests==2.31.0-py3-none-any.whl (from -r requirements.txt (line 14))
  Downloading requests-2.31.0-py3-none-any.whl (62 kB)
  02.6/62.6 kB 2.7 MB/s eta 0:00:00
Downloaded schedule-1.2.0-py2.py3-none-any.whl (12 kB)
Downloaded PyYAML-6.0.1-cp312-cp312-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (724 kB)
  02.6/724.6 kB 14.2 MB/s eta 0:00:00
Downloaded urllib3-2.0.7-py3-none-any.whl (124 kB)
  02.2/124.2 kB 10.3 MB/s eta 0:00:00
Downloaded pandas-2.3.1-cp312-cp312-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (12.0 MB)
  02.3/12.3 MB/s 24.3 MB/s eta 0:00:00
Downloaded certifi-2025.7.14-py3-none-any.whl (162 kB)
  02.7/162.7 kB 19.0 MB/s eta 0:00:00
Downloaded charset_normalizer-3.4.2-cp312-cp312-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (148 kB)
  04.6/148.6 kB 19.2 MB/s eta 0:00:00
Downloaded idna-3.10-py3-none-any.whl (70 kB)
  04.4/70.4 kB 8.2 MB/s eta 0:00:00
Downloaded numpy-2.3.1-cp312-cp312-manylinux_2_28_x86_64.whl (16.6 kB)
  06.5/16.6 kB 22.9 MB/s eta 0:00:00
Downloaded python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
  04.7/229.7 kB 9.9 MB/s eta 0:00:00
Downloaded pytz-2025.2-py2.py3-none-any.whl (569 kB)
  06.2/569.2 kB 11.6 MB/s eta 0:00:00
Downloaded tzdata-2025.2-py2.py3-none-any.whl (347 kB)
  04.7/347.8 kB 38.4 MB/s eta 0:00:00
Downloaded six-1.17.3-py2.py3-none-any.whl (56 kB)
  04.7/56.7 kB 11.6 MB/s eta 0:00:00
Installing collected packages: pytz, urllib3, tzdata, six, schedule, pyyaml, numpy, idna, charset-normalizer, certifi, requests, python-dateutil, pandas
Successfully installed certifi-2025.7.14 charset-normalizer-3.4.2 idna-3.10 numpy-2.3.1 pandas-2.3.1 python_dateutil-2.9.0.post0 pytz-2025.2 pyyaml-6.0.1 requests-2.31.0 six-1.17.0 tzdata-2025.2 urllib3-2.0.7
~/.Documents/workspace/log-alert-pipeline | main | 9s | log-alert-pipeline.n | 15:18:4
```

## Step 3: Configure Settings Files

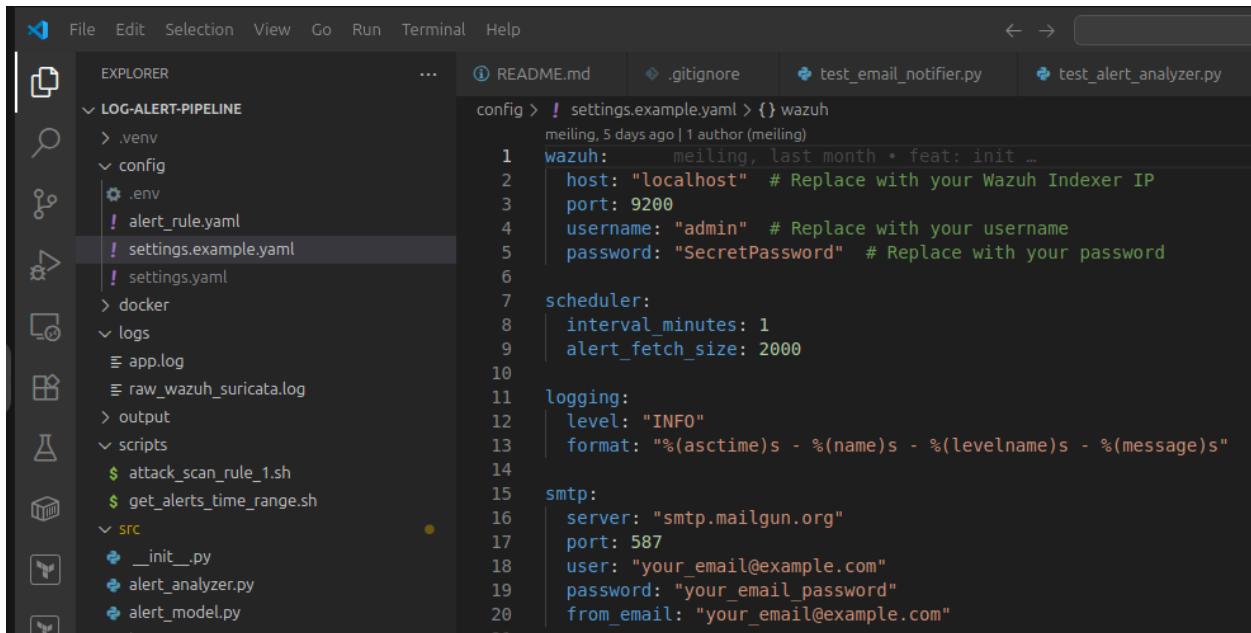
### Repository 1: wazuh-suricata-elk-docker

Add .env under the wazuh-server folder to set up the environment variable

```
C: > Users > cathe > Downloads > .env
1 INDEXER_URL=https://wazuh.indexer:9200
2 INDEXER_USERNAME=[REDACTED]
3 INDEXER_PASSWORD=[REDACTED]
4
5 API_USERNAME=[REDACTED]
6 API_PASSWORD=[REDACTED]
7
8 FILEBEAT_SSL_VERIFICATION_MODE=full
9 SSL_CERTIFICATE_AUTHORITIES=/etc/ssl/root-ca.pem
10 SSL_CERTIFICATE=/etc/ssl/filebeat.pem
11 SSL_KEY=/etc/ssl/filebeat.key
12
13 DASHBOARD_USERNAME=[REDACTED]
14 DASHBOARD_PASSWORD=[REDACTED]
15
16 OPENSEARCH_JAVA_OPTS=-Xms1g -Xmx1g
17
18 WAZUH_API_URL=https://wazuh.manager
```

### Repository 2: log-alert-pipeline

Put the settings.yaml under the config folder. Please use settings.example.yaml as an example and replace your credentials.



```
wazuh:
  host: "localhost" # Replace with your Wazuh Indexer IP
  port: 9200
  username: "admin" # Replace with your username
  password: "SecretPassword" # Replace with your password

scheduler:
  interval_minutes: 1
  alert_fetch_size: 2000

logging:
  level: "INFO"
  format: "%(asctime)s - %(name)s - %(levelname)s - %(message)s"

smtp:
  server: "smtp.mailgun.org"
  port: 587
  user: "your_email@example.com"
  password: "your_email_password"
  from_email: "your_email@example.com"
```

**Ensure sensitive information like API keys and passwords are securely managed and not hard-coded into public repositories.**

## Step 4: Start Services / Applications

### Repository 1: wazuh-suricata-elk-docker

#### One-Click install

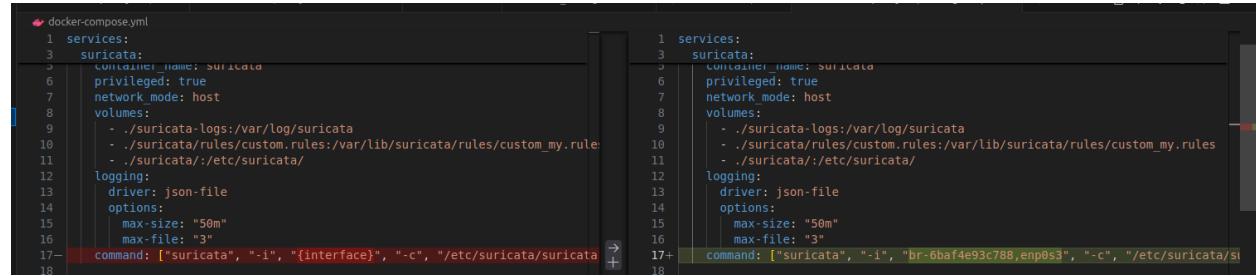
Step 1: Use “sudo make setup” to set up the environment

Step 2: Use “newgrp docker” to add the current user to the Docker group, reopen the terminal to renew the setting

Step 3: Use “sudo make deploy” to deploy the service for the first time to run the service

```
● vboxuser@Ubuntu:~/Documents/wazuh-suricata-elk-docker$ sudo make deploy
[sudo] password for vboxuser:
Running setup.sh to ensure Docker is installed...
Docker already installed, skipping setup.sh
Docker Bridge Interface: br-6baf4e93c788
Host Interface: enp0s3
[+] Running 3/3
  ✓ Container attacker-server    Started
  ✓ Container suricata          Started
  ✓ Container victim-server     Started
[+] Running 3/3
  ✓ Container wazuh-server-wazuh.indexer-1  Running
  ✓ Container wazuh-server-wazuh.manager-1  Running
  ✓ Container wazuh-server-wazuh.dashboard-1  Running
○ vboxuser@Ubuntu:~/Documents/wazuh-suricata-elk-docker$
```

It will change the interface of docker-compose.yaml and Suricata.yaml automatically



```
diff --git a/docker-compose.yaml b/docker-compose.yaml
index 1234567..8901234 100644
--- a/docker-compose.yaml
+++ b/docker-compose.yaml
@@ -1,18 +1,18 @@
 1 services:
 2   suricata:
 3     container_name: suricata
 4     privileged: true
 5     network_mode: host
 6     volumes:
 7       - ./suricata-logs:/var/log/suricata
 8       - ./suricata/rules/custom.rules:/var/lib/suricata/rules/custom_my.rules
 9       - ./suricata/:/etc/suricata/
10     logging:
11       driver: json-file
12       options:
13         max-size: "50m"
14         max-file: "3"
15       command: ["suricata", "-i", "{interface}", "-c", "/etc/suricata/suricata
16
17+     command: ["suricata", "-i", "br-6baf4e93c788", "-c", "/etc/suricata/suricata
18
```

docker-compose.yaml

```

suricata > ! suricata.yaml
565 logging:
566   - syslog:
567     type: json
568
569   ## Step 3: Configure common capture settings
570
571   ## See "Advanced Capture Options" below for more options, including Netmap
572   ## and PF_RING.
573
574   # Linux high speed capture support
575   af-packet:
576     - interface: {interface} # <- Replace with your actual bridge interface
577       # Number of receive threads. "auto" uses the number of cores
578       #threads: auto
579       # Default clusterid. AF_PACKET will load balance packets based on flow.
580       cluster-id: 99
581       # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or
582       # This is only supported for Linux kernel > 3.1
583       # possible value are:
584       # * cluster_flow: all packets of a given flow are sent to the same socket
585       # * cluster_cpu: all packets treated in kernel by a CPU are sent to the same
586       # socket. Requires at least Linux 3.14.
587       # * cluster_rss: all packets linked by network card to a RSS queue are
588       # sent to the same socket. Requires at least Linux 3.14.
589       # * cluster_ebpf: eBPF file load balancing. See doc/ug/capture-hard
590       # more info.
591       # Recommended modes are cluster_flow on most boxes and cluster_cpu or c
592
593 dpdk:
594   interfaces:
595     - interface: 0000:3b:00.0 # PCIe address of the NIC port
596       # - cap: forwards all packets and generates alerts (unless drop default)
597       # - ips: the same as tap mode but it also drops packets that are flagged
598       copy-mode: none
599       copy-iface: none # or PCIe address of the second interface
600
601     - interface: default
602       threads: auto
603       promisc: true
604       multicast: true
605       checksum-checks: true
606       checksum-checks-offload: true
607       mtu: 1500
608       rss-hash-functions: auto
609       mempool-size: 65535
610       mempool-cache-size: 257
611       rx-descriptors: 1024
612       tx-descriptors: 1024
613       copy-mode: none
614       copy-iface: none
615
616     # Cross platform libpcap capture support
617     pcap:
618       - interface: {interface} # <- Replace with your actual bridge interface
619       # On Linux, pcap will try to use mmap'ed capture and will use "buffer-size"
620       # as total memory used by the ring. So set this to something bigger
621
622
623   ## Step 3: Configure common capture settings
624
625   ## See "Advanced Capture Options" below for more options, including Netmap
626   ## and PF_RING.
627
628   # Linux high speed capture support
629   af-packet:
630     - interface: br-6ba4e93c788 # <- Replace with your actual bridge interface
631       # Number of receive threads. "auto" uses the number of cores
632       #threads: auto
633       # Default clusterid. AF_PACKET will load balance packets based on flow.
634       cluster-id: 99
635       # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per
636       # This is only supported for Linux kernel > 3.1
637       # possible value are:
638       # * cluster_flow: all packets of a given flow are sent to the same socket
639       # * cluster_cpu: all packets treated in kernel by a CPU are sent to the same
640       # socket. Requires at least Linux 3.14.
641       # * cluster_rss: all packets linked by network card to a RSS queue are
642       # sent to the same socket. Requires at least Linux 3.14.
643       # * cluster_ebpf: eBPF file load balancing. See doc/ug/capture-hard
644       # more info.
645       # Recommended modes are cluster_flow on most boxes and cluster_cpu or clus

```

Suricata.yaml

## **Repository 2: log-alert-pipeline**

### Command-Line Usage

Navigate to the project directory and run the following commands:

#### **Scheduled Mode (Every 5 Minutes)**

Continuously fetch Wazuh alerts at a set interval (default: 5 minutes)

```
python main.py
```

#### **Single Fetch Mode**

Fetch alerts once and display/process them.

```
python main.py --once
```

#### **Export Mode (Single Fetch & Export Alerts)**

Fetch alerts once and export the results to a file.

```
python main.py --export <filename.json>
```

```
python main.py --export <filename.csv>
```

Note: Filename must end with .json or .csv.

#### **Help**

Display usage instructions.

```
python main.py -help
```

## 3.2 Post-installation Verification

Check if the services are running correctly

### Repository 1: wazuh-suricata-elk-docker

Use “docker ps -a” to check that all six services are running

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	
921dda68c6de	jasonish/suricata:latest	"/docker-entrypoint..."	3 days ago	Up 2 minutes	suricata	
122bdefd1e73	wazuh-suricata-elk-docker-attacker-server	"/bin/bash"	3 days ago	Up 2 minutes	attacker-server	
6f2f47393fc5	wazuh-suricata-elk-docker-victim-server	"/entrypoint.sh"	3 days ago	Up 2 minutes	victim-server	
fe2a9e71d9e8	wazuh/wazuh-dashboard:4.12.0	"/entrypoint.sh"	4 days ago	Up 5 minutes	443/tcp, 0.0.0.0:443->5601/tcp, [::]:443->5601/tcp wazuh-server-wazuh.dashboard-1	
1c486020013d	wazuh/wazuh-indexer:4.12.0	"/entrypoint.sh open..."	4 days ago	Up 5 minutes	0.0.0.0:9200->9200/tcp, [::]:9200->9200/tcp wazuh-server-wazuh.indexer-1	
4735045df7d	wazuh/wazuh-manager:4.12.0	"/init"	4 days ago	Up 5 minutes	0.0.0.0:1514-1515->1514-1515/tcp, [::]:1514-1515->1514-1515/tcp wazuh-server-wazuh.manager-1	
4fdf0c2b6232	wazuh-server-setup-pipeline	"/entrypoint.sh ./in..."	3 weeks ago	Exited (0)	3 weeks ago wazuh-server-setup-pipeline-1	

Use “docker logs {docker container name}” to check that the single container is running correctly

~/.Documents/workspace/wazuh-suricata-elk-docker   main !2 ?4 docker logs -f suricata
i: suricata: This is Suricata version 7.0.10 RELEASE running in SYSTEM mode
i: threads: Threads created -> W: 4 FM: 1 FR: 1 Engine started.
i: suricata: Signal Received. Stopping engine.
i: device: br-69dac5993a57, enp0s3: packets: 10515, drops: 0 (0.00%), invalid checksum: 0
i: suricata: This is Suricata version 7.0.10 RELEASE running in SYSTEM mode
i: threads: Threads created -> W: 4 FM: 1 FR: 1 Engine started.
i: suricata: Signal Received. Stopping engine.
i: device: br-69dac5993a57, enp0s3: packets: 500, drops: 0 (0.00%), invalid checksum: 0
i: suricata: This is Suricata version 7.0.10 RELEASE running in SYSTEM mode
i: suricata: This is Suricata version 7.0.10 RELEASE running in SYSTEM mode
i: threads: Threads created -> W: 4 FM: 1 FR: 1 Engine started.

Check the Attacker-server can connect to wazuh-server and victim-server

└─(root@122bdefd1e73)-[ / ]
# ping 172.21.0.10
PING 172.21.0.10 (172.21.0.10) 56(84) bytes of data.
64 bytes from 172.21.0.10: icmp_seq=1 ttl=64 time=0.064 ms
64 bytes from 172.21.0.10: icmp_seq=2 ttl=64 time=0.086 ms
64 bytes from 172.21.0.10: icmp_seq=3 ttl=64 time=0.200 ms
^C
--- 172.21.0.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2060ms
rtt min/avg/max/mdev = 0.064/0.116/0.200/0.059 ms
└─(root@122bdefd1e73)-[ / ]
# ping 172.21.0.4
PING 172.21.0.4 (172.21.0.4) 56(84) bytes of data.
64 bytes from 172.21.0.4: icmp_seq=1 ttl=64 time=0.111 ms
64 bytes from 172.21.0.4: icmp_seq=2 ttl=64 time=0.071 ms
64 bytes from 172.21.0.4: icmp_seq=3 ttl=64 time=0.308 ms
^C
--- 172.21.0.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 0.071/0.163/0.308/0.103 ms

### Repository 2: log-alert-pipeline

If there is no error, and you can see the export file under the /output folder

## Test system connectivity

### Repository 1: wazuh-suricata-elk-docker

Verify status using wazuh-control status

Use “docker exec -it wazuh-scerer-wazuh.manager-1 bash enter the wazuh manager container and type “/var/ossec/bin/wazuh-control status” to check the wazuh services are running correctly

```
~/Doc/w/wazuh-suricata-elk-docker | main ?2 docker exec -it wazuh-server-wazuh.manager-1 bash  
bash-5.2# /var/ossec/bin/wazuh-control status  
wazuh-clusterd not running...  
wazuh-modulesd is running...  
wazuh-monitord is running...  
wazuh-logcollector is running...  
wazuh-remoted is running...  
wazuh-syscheckd is running...  
wazuh-analysisd is running...  
wazuh-maild not running...  
wazuh-execd is running...  
wazuh-db is running...  
wazuh-authd is running...  
wazuh-agentlessd not running...  
wazuh-integratord not running...  
wazuh-dbd not running...  
wazuh-csyslogd not running...  
wazuh-apid is running...  
bash-5.2#
```

Copy the following command and change it to your username and password to get the token.

```
curl -u "API_USERNAME:API_PASSWORD" -k -X POST  
https://localhost:55000/security/user/authenticate
```

```
~/Doc/w/wazuh-suricata-elk-docker | main ?2 curl -u "██████████:██████████" -k -X POST https://localhost:55000/security/user/authenticate  
  
{  
    "data": {  
        "token": "eyJhbGciOiJFUzUxMiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ3YXplACIsImF1ZCI6IldehenVoIEFQSSBSRVNUIiwibmJmIjoxNzQ5NTMzMjEwLCJleHAiOjE3NDk1MzQxMTAsInNIYiI6IndhenVoLXd1aSIsInJ1bl9hcyI6ZmFsc2UsInJiYWNgcm9sZXMiOlssXSwickJhY19tb2RlIjoid2hpGUifQ.AcclhKAT3rzbsEssIyb-XT2af88vf_T55irRZ-Tre4Ux6jMLzR9CfXfhKe8J8kgrR4Km9FEcV5ZW6_2fYKaWYAVdY3AjdQA_lqK85fvncfgJTTjDHS5AxQ3L9qrTGZ00FbIkHtZNb_M73OyOK0qchQ0AYKZR1bym87u4oBgHHMwC9"},  
    "error": 0}  
}
```

Copy the following command and change it to the token you received.

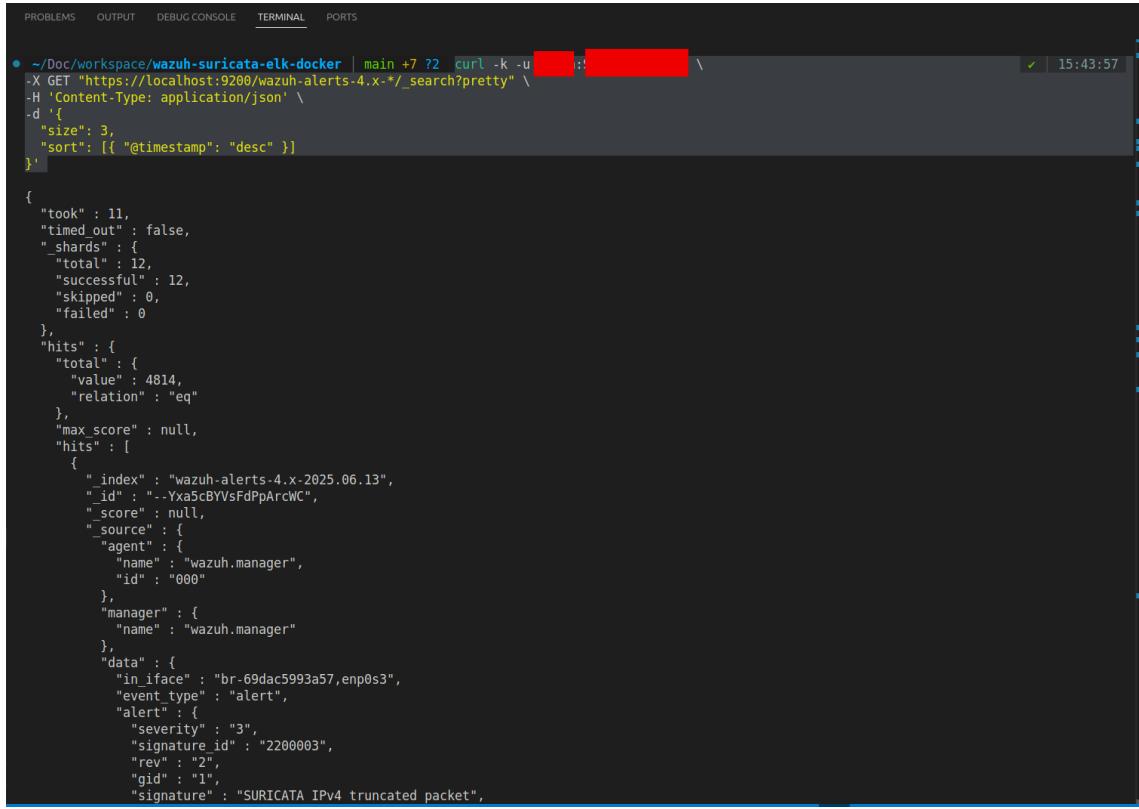
```
curl -k -X GET https://localhost:55000/agents -H  
"Authorization: Bearer  
eyJhbGciOiJFUzUxMilsInR5cCl6IkpxVCJ9.eyJpc3MiOiJ3  
YXp1aCIsImF1ZC16IldhenVoIEFQSSBSRVNULiwibmJmljo  
xNzQ5NTMzMjEwLCJleHAiOjE3NDk1MzQxMTAsInN1Yi  
I6IndhenVoLXd1aSlsInJ1bl9hcyl6ZmFsc2UsInJiYWNgcm  
9sZXMiOlxsXSwicmJhY19tb2Rlljoid2hpdGUifQ.AcclhKA  
T3rbksEsslyb-XT2aF88vf_T5SirRZ-  
Ire4Ux6jMLzR9CFgXFhKe8J8kgrR4Km9FEcV5ZW6_J2fY  
KaWYAVdY3AjdQA_IqK85fvncfgJTtjDHS5AxQ3L9qrTGZ  
OOFBikHtZNB_M7JOyQKOqchQ0AYKZR1bym87u4oBg  
HHMwC9" -H "Content-Type: application/json"
```

```
~Doc/wazuh-suricata-elk-docker | main ?2 curl -k -X GET https://localhost:55000/agents -H "Authorization: Bearer eyJhbGciOiJFUzUmIsInR5cCI6IkpXVCJ9.eyJpc3MiAiJYXp1aClIsImF1ZCI6IldhenVoIEFQSSBRSVNUiwiimhJmIxjoxNzQ5NTMzMjEwLCJleHAiOjE3NDk1MzQxMTAsInN1Yi6IndhenVoLXd1aSisInJb1hcy16MzFsc2UsInJiYWnfcm9sZXmiOlxsxSwicJhY19tb2RlIjoiid2hdGuifQ_AccLhKAT3rbksESSiyb-XT2af88vft_T5SiRZ-Ire4Ux6jMLZR9CfxFhKe8J8kgR4K9MeFcv5Zw_6_j2FyKaWYAvdY3Ajdq_Lqa_lk85fvcnfgJttJdHS5AxQ3L9qrTGZ00FbIkHtZNB_M7J0yQKoQch0AYKZR1bym87u4oBgHMwC9" -H "Content-Type: application/json"
{"data": {"affected_items": [{"os": {"arch": "x86_64", "major": "2023", "name": "Amazon Linux", "platform": "amzn", "uname": "Linux |wazuh.manager|6.11.0-26-generic |#26-24.04.1-Ubuntu SMP PREEMPT_DYNAMIC Thu Apr 17 19:20:47 UTC 2 |x86_64", "version": "2023"}, {"node_name": "node01", "version": "Wazuh v4.12.0", "manager": "wazuh.manager", "lastKeepAlive": "9999-12-31T23:59:59+00:00", "registerIP": "127.0.0.1", "group_config_status": "synced", "status": "active", "dateAdd": "2025-06-09T05:07:56+00:00", "name": "wazuh.manager", "ip": "127.0.0.1", "status_code": 0, "id": "000"}, {"node_name": "unknown", "registerIP": "any", "group_config_status": "not synced", "status": "never_connected", "dateAdd": "2025-06-10T04:25:07+00:00", "name": "2a82e879cf2e", "ip": "any", "status_code": 0, "id": "001"}, {"node_name": "unknown", "registerIP": "any", "group_config_status": "not synced", "status": "never_connected", "dateAdd": "2025-06-10T04:36:05+00:00", "name": "9d447ecf9165", "ip": "any", "status_code": 0, "id": "002"}, {"os": {"arch": "x86_64", "codename": "Jammy Jellyfish", "major": "22", "minor": "04", "name": "Ubuntu", "platform": "ubuntu", "uname": "Linux |f3ad3bb7db8 |6.11.0-26-generic |#26-24.04.1-Ubuntu SMP PREEMPT_DYNAMIC Thu Apr 17 19:20:47 UTC 2 |x86_64", "version": "22.04.5 LTS"}, "node_name": "node01", "version": "Wazuh v4.12.0", "discconection_time": "2025-06-10T05:17:18+00:00", "configSum": "ab73af4169ff13fdd81903b5f23d8d00", "mergedSum": "4a8724b20dee0124ff9656783c4904e", "group": ["default"], "manager": "wazuh.manager", "lastKeepAlive": "2025-06-10T05:18:28+00:00", "registerIP": "any", "group_config_status": "synced", "status": "disconnected", "dateAdd": "2025-06-10T04:43:18+00:00", "name": "f3ad3bb7db8", "ip": "172.21.0.6", "status_code": 5, "id": "003"}, {"os": {"arch": "x86_64", "codename": "Jammy Jellyfish", "major": "22", "minor": "04", "name": "Ubuntu", "platform": "ubuntu", "uname": "Linux |3e5aaaf5bf8db8 |6.11.0-26-generic |#26-24.04.1-Ubuntu SMP PREEMPT_DYNAMIC Thu Apr 17 19:20:47 UTC 2 |x86_64", "version": "22.04.5 LTS"}, "node_name": "node01", "version": "Wazuh v4.12.0", "configSum": "ab73af4169ff13fdd81903b5f23d8d00", "mergedSum": "4a8724b20dee0124ff9656783c4904e", "group": ["default"], "manager": "wazuh.manager", "lastKeepAlive": "2025-06-10T05:31:44+00:00", "registerIP": "any", "group_config_status": "synced", "status": "active", "dateAdd": "2025-06-10T05:22:14+00:00", "name": "3e5aaaf5bf8db8", "ip": "172.21.0.5", "status_code": 0, "id": "004"}], "total_affected_items": 5, "total_failed_items": 0, "failed_items": [], "message": "All selected agents information was returned", "error": 0}]}]
```

*How to call the API to get the alert back*

```
curl -k -u INDEX_USERNAME:INDEX_PASSWORD \
-X GET "https://localhost:9200/wazuh-alerts-4.x- \
*/_search?pretty" \
-H 'Content-Type: application/json' \
-d '{
  "size": 3,
  "sort": [{"@timestamp": "desc"}]
}'
```

Copy the following command and change it to your username and password.



The screenshot shows a terminal window with the following command history:

```
~/Doc/workspace/wazuh-suricata-elk-docker | main +7 ?2 curl -k -u [REDACTED]:[REDACTED] \ \
-X GET "https://localhost:9200/wazuh-alerts-4.x-*/_search?pretty" \
-H 'Content-Type: application/json' \
-d '{
  "size": 3,
  "sort": [{"@timestamp": "desc"}]
}'
```

The command is followed by a JSON response object. The response includes a summary of the search results and a list of hits. One hit is shown in detail, containing information such as index, id, score, source, agent, manager, data, and alert details.

```
{
  "took" : 11,
  "timed_out" : false,
  "_shards" : {
    "total" : 12,
    "successful" : 12,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4814,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [
      {
        "_index" : "wazuh-alerts-4.x-2025_06_13",
        "_id" : "--Yxa5cBYVsFdPpArcWC",
        "_score" : null,
        "_source" : {
          "agent" : {
            "name" : "wazuh.manager",
            "id" : "000"
          },
          "manager" : {
            "name" : "wazuh.manager"
          },
          "data" : {
            "in_iface" : "br-69dac5993a57,enp0s3",
            "event_type" : "alert",
            "alert" : {
              "severity" : "3",
              "signature_id" : "2200003",
              "rev" : "2",
              "gid" : "1",
              "signature" : "SURICATA IPv4 truncated packet"
            }
          }
        }
      }
    ]
}
```

## Repository 2: log-alert-pipeline

Step 1: Check the export file under the /output folder.

Step 2: Check if the email you set up in your notification\_email in the settings.example.yaml receives the alert mail.

## 4. User Guide

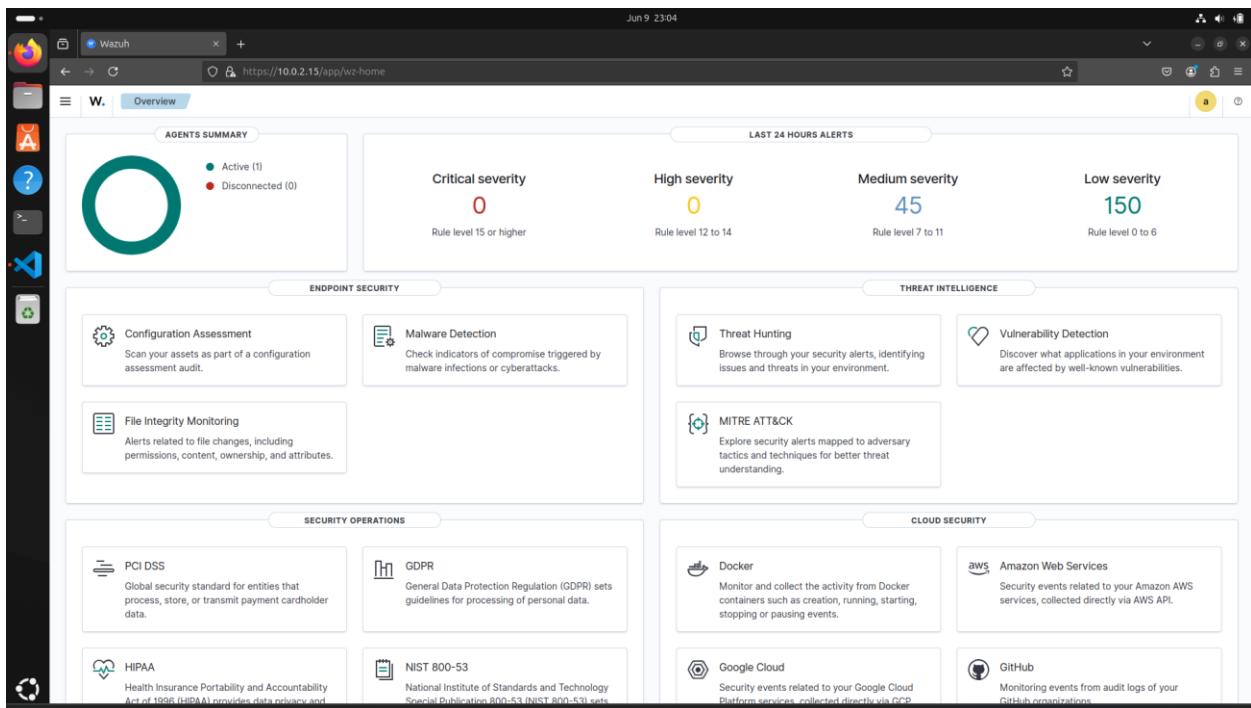
### 4.1 Login and Main Interface Overview

#### Login Instructions / Default Account

##### Repository 1: wazuh-suricata-elk-docker

1. Open a web browser (Google Chrome / Firefox).
2. In the address bar, enter the following URL:  
`https://{your_vm_IP}/app/wz-home`  
Replace {your\_vm\_IP} with the actual IP address of your VM or host system
3. You will be prompted to enter login credentials. These can be found in your project's .env file.  
Default Username: admin  
Default Password: admin (or as specified in .env)
4. Upon successful login, you will be redirected to the **Wazuh Dashboard**.

```
C: > Users > cathe > Downloads > .env
1 INDEXER_URL=https://wazuh.indexer:9200
2 INDEXER_USERNAME=[REDACTED]
3 INDEXER_PASSWORD=[REDACTED]
4
5 API_USERNAME=[REDACTED]
6 API_PASSWORD=[REDACTED]
7
8 FILEBEAT_SSL_VERIFICATION_MODE=full
9 SSL_CERTIFICATE_AUTHORITIES=/etc/ssl/root-ca.pem
10 SSL_CERTIFICATE=/etc/ssl/filebeat.pem
11 SSL_KEY=/etc/ssl/filebeat.key
12
13 DASHBOARD_USERNAME=[REDACTED]
14 DASHBOARD_PASSWORD=[REDACTED]
15
16 OPENSEARCH_JAVA_OPTS=-Xms1g -Xmx1g
17
18 WAZUH_API_URL=https://wazuh.manager
```



## Repository 2: log-alert-pipeline

No login is required for this repository.

Processed alert data will be exported to the /output directory of the project.

Navigate to the /output folder in your file explorer or terminal to access the exported .json or .csv files.

## Overview of Main Feature Areas

### Wazuh Dashboard ([https://your\\_vm\\_IP/app/wz-home](https://your_vm_IP/app/wz-home))

- Security Overview (Home Page):**  
Provides a high-level summary of active alerts, agent status, and security event trends.
- Security Events (Modules > Security Events):**  
Detailed view of all alerts generated by Wazuh Agents and Suricata, with filters for severity, date/time, and agent.
- Agents Management (Modules > Agents):**  
Displays the list of registered Wazuh agents, their current status, IP addresses, and last communication time.
- Rules and Policies (Modules > Rules):**  
Allows you to browse existing detection rules. Custom rules can be added in configuration files.
- Kibana Discover Panel (Discover):**  
Full-text search interface where you can search raw logs, filter by fields (e.g., source IP, rule ID), and visualize structured data.

- **Dashboards (Dashboard):**

Visual representation of security metrics such as Top Offenders (IP addresses), Alert Trends, Detected Attack Types, etc.

#### **Log-Alert-Pipeline (/output folder)**

- **Exported Alert Files (.json / .csv):**

Files generated by the Data Fetcher script containing simplified, correlated alert data.

- File Naming Convention: {timestamp}\_alert\_result.json or .csv
- These files can be used for reporting, further analysis, or as learning materials.

Field	Description
Timestamp	The exact time the alert was generated
Rule ID	Identifier of the matched detection rule
Rule Name	Human-readable name of the detection rule
Source IP	IP address of the attacker/source system
Destination IP	Target IP (if available)
Destination Port	Target port (if applicable)
Source User	Username associated with the event (if available)
Destination User	Target username (if applicable)
Command	The command or payload associated with the alert (if available)
Description	Short description of the alert event

Timestamp	Rule ID	Rule Name	Src IP	Dest IP	Dest Port	Src User	Dst User	Command	Description
1291	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1292	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1293	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1294	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1295	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1296	7/26/2025 0:22	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1297	7/26/2025 0:22	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1298	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1299	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1300	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1301	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1302	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1303	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1304	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1305	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1306	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1307	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1308	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1309	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1310	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1311	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1312	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1313	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1314	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1315	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1316	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1317	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1318	7/26/2025 12:26	4 UDP Flood to DNS Port Detected	172.21.0.11	172.21.0.10	53				Suricata: Alert - Custom UDP Flood to DNS Port Detected
1319	7/26/2025 0:19	4 UDP Flood to DNS Port Detected	172.21.0.11	172.21.0.10	53				Suricata: Alert - Custom UDP Flood to DNS Port Detected
1320	7/26/2025 16:41	5 SSH Authentication Failure Detected	172.21.0.11			root			syslog: User missed the password more than one time
1321	7/26/2025 17:12	6 Account Information Change Detected				testuser2			Information from the user was changed.
1322	7/26/2025 17:16	7 Successful Sudo to Root Detected				testuser2	root	/usr/bin/vim /etc/shadow	Successful sudo to ROOT executed.
1323	7/26/2025 17:15	7 Successful Sudo to Root Detected				testuser2	root	/usr/bin/apt install vim	Successful sudo to ROOT executed.
1324	7/26/2025 17:14	7 Successful Sudo to Root Detected				root	root	/usr/bin/newgrp sudo	Successful sudo to ROOT executed.

Example for CSV

```
{} 20250726_174551_alert.json X
C: > Users > cathe > Documents > Last_semester > Capstone > Sprint3_update > Showcase > {} 20250726_174551_alert.json > .
1 [
2 {
3     "Timestamp": "2025-07-26T23:15:54.109Z",
4     "Rule ID": 2,
5     "Rule Name": "SYN Flood Detected",
6     "Src IP": "172.21.0.10",
7     "Dest IP": "185.125.190.82",
8     "Dest Port": "80",
9     "Src User": "",
10    "Dst User": "",
11    "Command": "",
12    "Description": "Suricata: Alert - Custom DOS Detection - SYN Flood Detected"
13 },
14 }
```

Example for JSON

## 4.2 Core Function Operations

### Repository 1: wazuh-suricata-elk-docker - Function 1: Wazuh detect - log-in failed

On the attacker-server, attempt to enter an incorrect root password for the victim-server.

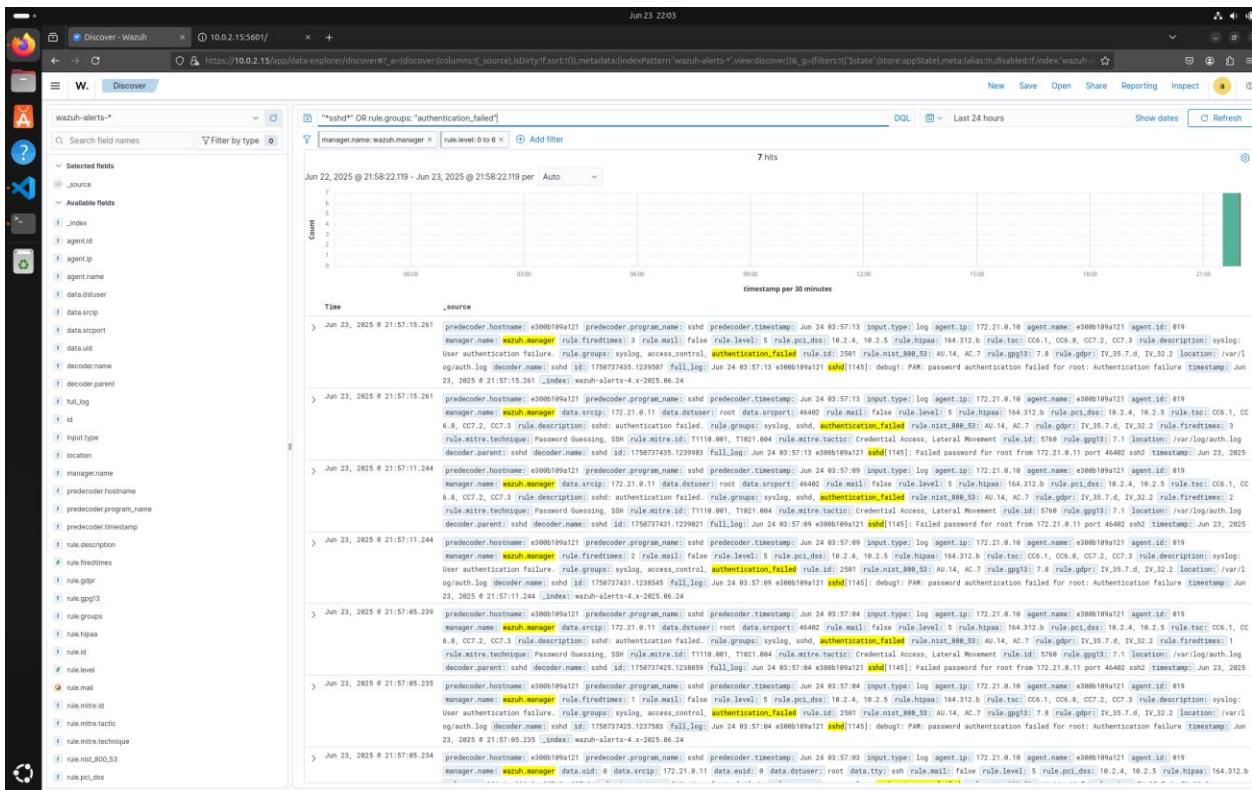
```
○ ~/Doc/w/wazuh-suricata-elk-docker | main !4 ?3 docker exec -it attacker-server bash
  └─(root@e797eb9bd20d) [~]
      # ssh root@172.21.0.10
      The authenticity of host '172.21.0.10 (172.21.0.10)' can't be established.
      ED25519 key fingerprint is SHA256:eaJWNglV65IbcL4T5tb9RRcS4KZazwjplWFcBojNHJu
      .
      This key is not known by any other names.
      Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
      Warning: Permanently added '172.21.0.10' (ED25519) to the list of known hosts
      .
      root@172.21.0.10's password:
      Permission denied, please try again.
      root@172.21.0.10's password:
      Permission denied, please try again.
      root@172.21.0.10's password:
      root@172.21.0.10: Permission denied (publickey,password).

  └─(root@e797eb9bd20d) [~]
      #
```

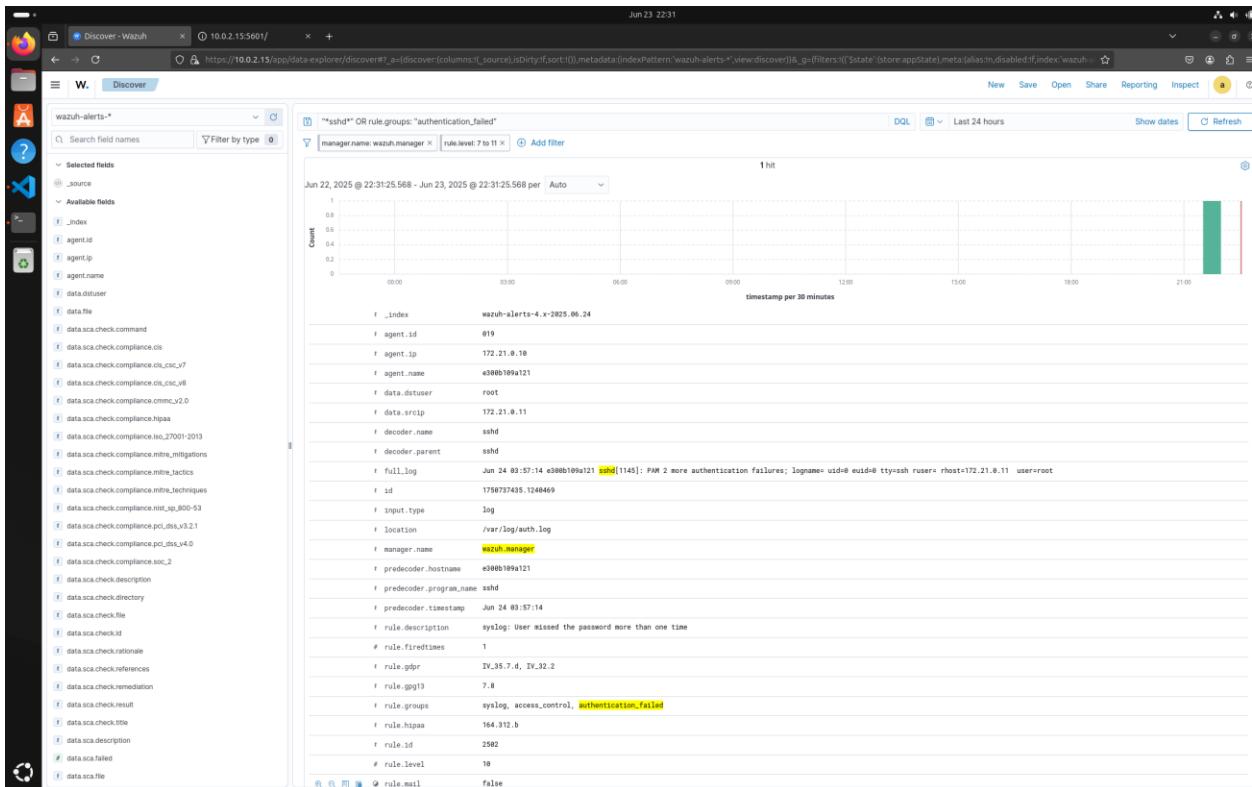
The screenshot shows the Wazuh web interface at <https://10.0.2.15/app/wz-home#/overview>. The top navigation bar includes 'Overview', 'Agents Summary', 'Logs', 'Metrics', 'Events', 'Incidents', 'Logs', 'Metrics', 'Events', and 'Incidents'. The main dashboard features several sections:

- AGENTS SUMMARY**: Shows 1 Active agent and 15 Disconnected agents.
- Critical severity**: 0 alerts (Rule level 15 or higher).
- High severity**: 0 alerts (Rule level 12 to 14).
- Medium severity**: 104 alerts (Rule level 7 to 11).
- Low severity**: 1,470 alerts (Rule level 0 to 6).
- ENDPOINT SECURITY**: Includes Configuration Assessment, Malware Detection, File Integrity Monitoring, Threat Hunting, and MITRE ATT&CK.
- THREAT INTELLIGENCE**: Includes Vulnerability Detection.
- SECURITY OPERATIONS**: Includes PCI DSS, GDPR, HIPAA, NIST 800-53, and TSC.
- CLOUD SECURITY**: Includes Docker, Google Cloud, Amazon Web Services, and GitHub.

Authentication failed is shown



Can even catch the attacker's IP address



## Repository 1: wazuh-suricata-elk-docker - Function 2: Wazuh detect - create a user

Try to add the new user to the victim-server

```
root@a741ecf32c47:/# adduser testuser
Adding user `testuser' ...
Adding new group `testuser' (1000) ...
Adding new user `testuser' (1000) with group `testuser' ...
Creating home directory `/home/testuser' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for testuser
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
root@a741ecf32c47:/#
```

The add users rule is captured

Time	_source
Jul 20, 2025 @ 21:48:48.610	<pre>timestamp per 30 minutes {"@version": "1", "@source": "wazuh-alerts-4.x-2025.07.21", "@index": "wazuh-alerts-4.x-2025.07.21", "rule_id": "T1098", "rule_name": "Account Manipulation", "rule_level": "8", "rule_nct_desc": "10.2.7, 10.2.5, 8.1.2", "rule_ipaa": "164.312.a.2.I, 164.312.a.2.II", "rule_tsc": "CC6.8, C7.2, CC7.3", "rule_description": "Information from the user was changed", "rule_groups": "syslog, adduser", "rule_nist_800_53": "AU.14, AC.7, AC.2, IA.4", "rule_gdpr": "IV.35.7.d, IV.32.2", "rule_firedtimes": 1, "rule_mitre_technique": "Persistence", "rule_mitre_id": "T1098", "rule_mitre_tactic": "Persistence", "rule_id": "5904", "rule_gpg13": "4.13", "location": "/var/log/auth.log", "decoder.parent": "chfn", "decoder.name": "chfn", "id": "1753069728.2310526", "full_log": "Jul 21 03:48:48 a741ecf32c47 chfn[661]: changed user `testuser' information", "timestamp": "Jul 20, 2025 @ 21:48:48.610", "index": "wazuh-alert"}", "agent.id": "006", "agent.ip": "172.21.0.10", "agent.name": "a741ecf32c47", "data.dstuser": "testuser", "decoder.name": "chfn", "decoder.parent": "chfn", "full_log": "Jul 21 03:48:48 a741ecf32c47 chfn[661]: changed user `testuser' information", "id": "1753069728.2310526", "input.type": "log", "location": "/var/log/auth.log", "manager.name": "wazuh_manager", "predecoder.hostname": "a741ecf32c47", "predecoder.program_name": "chfn", "predecoder.timestamp": "Jul 21 03:48:48"}, {"@version": "1", "@source": "wazuh-alerts-4.x-2025.07.21", "@index": "wazuh-alerts-4.x-2025.07.21", "rule_id": "T1098", "rule_name": "Account Manipulation", "rule_level": "8", "rule_nct_desc": "10.2.7, 10.2.5, 8.1.2", "rule_ipaa": "164.312.a.2.I, 164.312.a.2.II", "rule_tsc": "CC6.8, C7.2, CC7.3", "rule_description": "Information from the user was changed", "rule_groups": "syslog, adduser", "rule_nist_800_53": "AU.14, AC.7, AC.2, IA.4", "rule_gdpr": "IV.35.7.d, IV.32.2", "rule_firedtimes": 1, "rule_mitre_technique": "Persistence", "rule_mitre_id": "T1098", "rule_mitre_tactic": "Persistence", "rule_id": "5904", "rule_gpg13": "4.13", "location": "/var/log/auth.log", "decoder.parent": "chfn", "decoder.name": "chfn", "id": "1753069728.2310526", "full_log": "Jul 21 03:48:48 a741ecf32c47 chfn[661]: changed user `testuser' information", "timestamp": "Jul 20, 2025 @ 21:48:48.610", "index": "wazuh-alert"}", "agent.id": "006", "agent.ip": "172.21.0.10", "agent.name": "a741ecf32c47", "data.dstuser": "testuser", "decoder.name": "chfn", "decoder.parent": "chfn", "full_log": "Jul 21 03:48:48 a741ecf32c47 chfn[661]: changed user `testuser' information", "id": "1753069728.2310526", "input.type": "log", "location": "/var/log/auth.log", "manager.name": "wazuh_manager", "predecoder.hostname": "a741ecf32c47", "predecoder.program_name": "chfn", "predecoder.timestamp": "Jul 21 03:48:48"}", "expanded_document": "The expanded document shows the raw log entry from /var/log/auth.log: Jul 21 03:48:48 a741ecf32c47 chfn[661]: changed user `testuser' information. This entry is part of a larger event with ID 1753069728.2310526, which includes metadata about the decoder (chfn), agent (IP 172.21.0.10, ID 006), and the Wazuh manager (wazuh_manager). The event also includes rules and techniques from the MITRE ATT&amp;CK framework."}</pre>

## Repository 1: wazuh-suricata-elk-docker - Function 3: Wazuh detect - user changes the password

Use passwd to change the password of the testuser2 in the victim-server

```

password: password unchanged
root@56cfeabbf703:/# passwd testuser2
New password:
Retype new password:
passwd: password updated successfully

```

The alert will show “password changed for root”.

Time	_source
Jul 20, 2025 @ 21:54:19.248	<pre> predecoder.hostname: a741ecf32c47 predecoder.program_name: passwd predecoder.timestamp: Jul 21 03:54:17 input.type: log agent.ip: 172.21.0.10 agent.name: a741ecf32c47 agent.id: 006 manager.name: <b>wazuh.manager</b> data.dstuser: root rule.firetimes: 2 rule.mail: false rule.level: 3 rule.pci_dss: 8.1.2, 10.2.5 rule.hipaa: 164.312.a.2.I, 164.312.a.2.II, 164.312.b rule.tsc: C06.8, C07.2, C07.3 rule.description: PAM: User changed password. rule.groups: pam, syslog rule.id: 5555 rule.nist_800_53: AC.2, TA.4, AU.14, AC.7 rule.gdpr: IV_35.7.d, IV_32.2 location: /var/log/auth.log decoder.parent: pam decoder.name: pam id: 1753070059.2311028 full_log: Jul 21 03:54:17 a741ecf32c47 passwd[677]: pam_unix(passwd:chauthtok): password changed for root timestamp: Jul 20, 2025 @ 21:54:19.248 _index: wazuh-alerts-4.x-2025.07.21 </pre>

Expanded document

Table JSON

t _index	wazuh-alerts-4.x-2025.07.21
t agent.id	006
t agent.ip	172.21.0.10
t agent.name	a741ecf32c47
t data.dstuser	root
t decoder.name	pam
t decoder.parent	pam
t full_log	Jul 21 03:54:17 a741ecf32c47 passwd[677]: pam_unix(passwd:chauthtok): password changed for root
t id	1753070059.2311028
t input.type	log
t location	/var/log/auth.log
t manager.name	<b>wazuh.manager</b>
t predecoder.hostname	a741ecf32c47
t predecoder.program_name	passwd
t predecoder.timestamp	Jul 21 03:54:17

## Repository 1: wazuh-suricata-elk-docker - Function 4: Wazuh detect - remote access and session opened alert

Setting the password for the root account.

```

root@56cfeabbf703:/# passwd
New password:
Retype new password:
passwd: password updated successfully
root@56cfeabbf703:/#

```

Remote login to the victim-server by using the root account successfully. It will show the message, which is “session opened for user root” and “sshd: Accepted password” on the dashboard.

Time	_source
> Jul 20, 2025 @ 21:55:53.599	<pre> predecoder.hostname: a741ecf32c47 predecoder.program_name: sshd predecoder.timestamp: Jul 21 03:55:53 input.type: log agent.ip: 172.21.0.10 agent.name: a741ecf32c47 agent.id: 006 manager.name: <b>wazuh.manager</b> data.uid: 0 data.dstuser: root(uid=0) rule.mail: false rule.level: 3 rule.pci_dss: 10.2.5 rule.hipaa: 164.312.b rule.tsc: C06.8, C07.2, C07.3 rule.description: PAM: Login session opened. rule.groups: pam, syslog, authentication_success rule.nist_800_53: AU.14, AC.7 rule.gdpr: IV_32.2 rule.firetimes: 1 rule.mitre.technique: Valid Accounts rule.mitre.id: T1078 rule.mitre.tactic: Defense Evasion, Persistence, Privilege Escalation, Initial Access rule.id: 5501 rule.mitre_id: T1078 rule.mitre_tactic: Defense Evasion, Persistence, Privilege Escalation, Initial Access, Lateral Movement rule.decoder.parent: pam decoder.name: pam id: 1753070153.2311986 full_log: Jul 21 03:55:53 a741ecf32c47 sshd[678]: pam_unix(sshd:session): session opened for user root(uid=0) by (uid=0) </pre>
> Jul 20, 2025 @ 21:55:53.599	<pre> predecoder.hostname: a741ecf32c47 predecoder.program_name: sshd predecoder.timestamp: Jul 21 03:55:53 input.type: log agent.ip: 172.21.0.10 agent.name: a741ecf32c47 agent.id: 006 manager.name: <b>wazuh.manager</b> data.scpid: 172.21.0.11 data.dstuser: root data.srport: 50454 rule.mail: false rule.level: 3 rule.hipaa: 164.312.b rule.pci_dss: 10.2.5 rule.tsc: C06.8, C07.3 rule.description: sshd: authentication success. rule.groups: syslog, sshd, authentication_success rule.nist_800_53: AU.14, AC.7 rule.gdpr: IV_32.2 rule.firetimes: 1 rule.mitre.technique: Valid Accounts, Remote Services rule.mitre.id: T1078, T1021 rule.mitre.tactic: Defense Evasion, Persistence, Privilege Escalation, Initial Access, Lateral Movement rule.id: 5715 rule.gpgid: 7.1, 7.2 location: /var/log/auth.log decoder.parent: sshd decoder.name: sshd id: 1753070153.2311527 full_log: Jul 21 03:55:53 a741ecf32c47 sshd[678]: Accepted pass </pre>

## Repository 1: wazuh-suricata-elk-docker - Function 5: Wazuh detect - High privilege operation: sudo, root operation record

Install sudo on the Victim-server using the command line “apt update && apt install sudo -y”.

```
Jul 21 04:40:58 a741ecf32c47: debug1: session_input_channel_req: session 0 req window-change
root@a741ecf32c47:/# apt update && apt install sudo -y
Hit:1 https://packages.wazuh.com/4.x/apt stable InRelease
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Hit:3 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:4 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [1267 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [3148 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [4932 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [5139 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [3461 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1572 kB]
Fetched 19.9 MB in 2s (8173 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
W: https://packages.wazuh.com/4.x/apt/dists/stable/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  sudo
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 820 kB of archives.
After this operation, 2568 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 sudo amd64 1.9.9-1ubuntu2.5 [820 kB]
Fetched 820 kB in 0s (1651 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package sudo.
```

Add the testuser to the sudo group

```
root@a741ecf32c47:/# usermod -aG sudo testuser
root@a741ecf32c47:/# newgrp sudo
root@a741ecf32c47:/# █
```

Log in to the attacker server using the testuser account and print the /etc/shadow file; the alert will be displayed on the dashboard.

```

└─(root@9074216d9feb)─[~/]
# ssh testuser@172.21.0.10
testuser@172.21.0.10's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.11.0-29-generic x86_64)

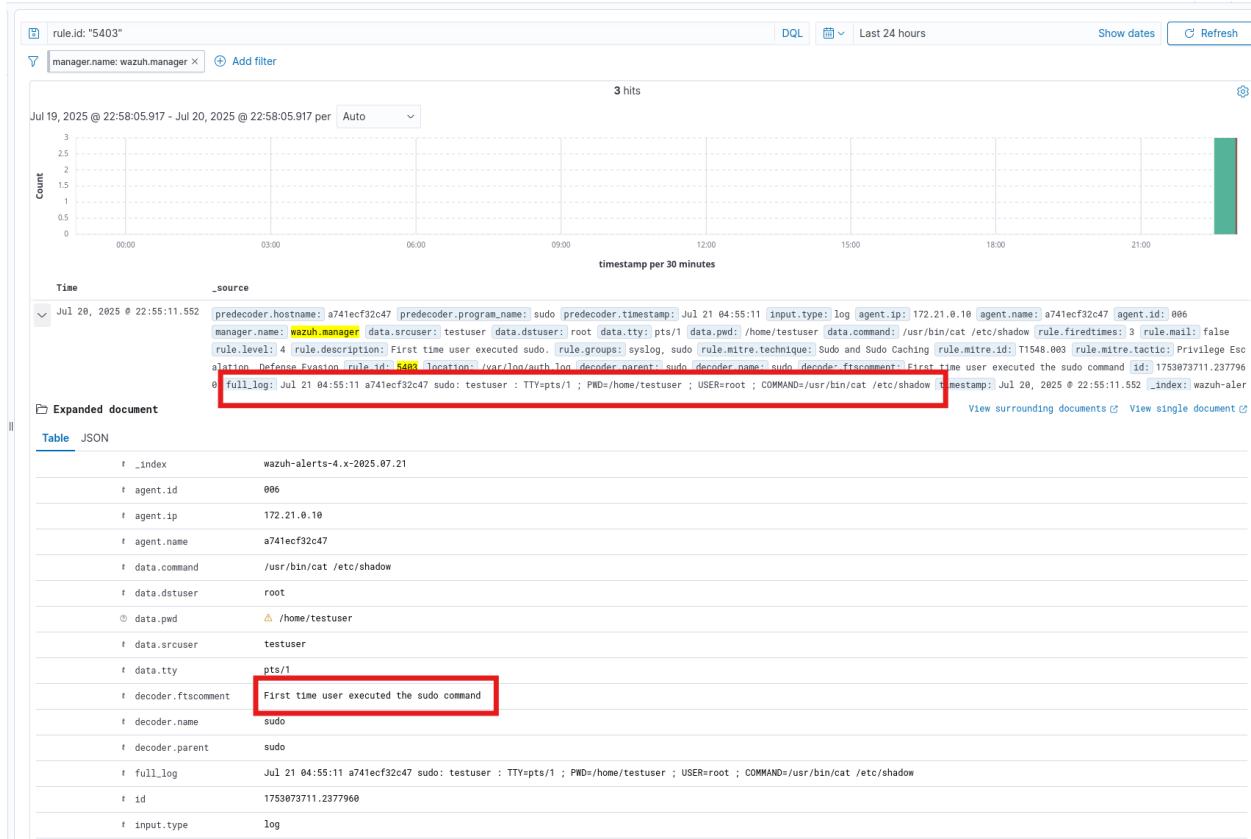
 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Mon Jul 21 04:47:14 2025 from 172.21.0.11
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

testuser@a741ecf32c47:~$ sudo cat /etc/shadow
[sudo] password for testuser:
root:$y$90M4EJgdJ5/kAJBNPf4v.$i/C.kP3F8goW3YkClNELtoQoaw4GEF8bIv/aMBInK5C:20290:0:99999:7:::
daemon:*:20283:0:99999:7:::
bin:*:20283:0:99999:7:::
sys:*:20283:0:99999:7:::
sync:*:20283:0:99999:7:::
games:*:20283:0:99999:7:::
man:*:20283:0:99999:7:::
lp:*:20283:0:99999:7:::
mail:*:20283:0:99999:7:::
news:*:20283:0:99999:7:::
uucp:*:20283:0:99999:7:::
proxy:*:20283:0:99999:7:::
www-data:*:20283:0:99999:7:::
backup:*:20283:0:99999:7:::
list:*:20283:0:99999:7:::
irc:*:20283:0:99999:7:::

```



## Repository 1: wazuh-suricata-elk-docker - Function 6: Wazuh detect - System Change: Modification or Deletion of Critical Files

Modify the /etc/shadow file on the victim-server from the attacker-server and save it, the action will be caught on the dashboard.

```
testuser@a741ecf32c47:~$ vim /etc/shadow  
testuser@a741ecf32c47:~$ sudo vim /etc/shadow  
testuser@a741ecf32c47:~$
```

Jul 20, 2025 0 23:07:30.957	predecoder.hostname: a741ecf32c47 predecoder.program_name: sudo predecoder.timestamp: Jul 21 05:07:30 input.type: log agent.ip: 172.21.0.10 agent.name: a741ecf32c47 agent.id: 006 manager.name: wazuh_manager data.srcuser: testuser data.dstuser: root data.tty: pts/1 data.pwd: /home/testuser data.command: /usr/bin/vim /etc/shadow rule.mail: false rule.level: 3 rule.pc1.dss: 10.2.5, 10.2.2 rule.hipaa: 164.312.b rule.tsc: CC6.B, CC7.2, CC7.3 rule.description: Successful sudo to ROOT executed. rule.groups: syslog, sudo rule.mist_id: B00..53; AU.14, AC.6 rule.gdpr: IV.32.2 rule.firetimes: 1 rule.mitre.technique: Sudo and Sudoc Caching rule.mitre.id: T1548 .003 rule.mitre.tactic: Privilege Escalation, Defense Evasion rule.id: 5402 rule.apq13: 7.6, 7.8, 7.13 location: /var/log/auth.log decoder.parent: sudo decoder.name: sudo decoder.ftacomment: First time user executed the sudo command id: 1753074450.2379158
Expanded document	
Table	JSON
↳ _index	wazuh-alerts-4.x-2025.07.21
↳ agent.id	006
↳ agent.ip	172.21.0.10
↳ agent.name	a741ecf32c47
↳ data.command	/usr/bin/vim /etc/shadow
↳ data.dstuser	root
↳ data.pwd	△ /home/testuser
↳ data.srcuser	testuser
↳ data.tty	pts/1
↳ decoder.ftacomment	First time user executed the sudo command
↳ decoder.name	sudo
↳ decoder.parent	sudo
↳ full_log	Jul 21 05:07:30 a741ecf32c47 sudo: testuser : TTY=pts/1 ; PWD=/home/testuser ; USER=root ; COMMAND=/usr/bin/vim /etc/shadow
↳ id	1753074450.2379158
↳ input.type	log

## Repository 1: wazuh-suricata-elk-docker - Function 7: Suricata detect - simulate ICMP/UDP/HTTP attacks with Scapy

This screenshot shows the successful deployment of a security monitoring and intrusion detection environment using Docker Compose. In the first step, three containers—*victim-server*, *attacker-server*, and *suricata*—were launched to simulate a cyber attack scenario and monitor network traffic. In the second step, within the wazuh-server directory, three Wazuh-related containers—*indexer*, *manager*, and *dashboard*—were started to enable log collection, threat detection, and centralized security visualization. Together, this setup forms a comprehensive environment for cybersecurity testing, analysis, and monitoring.

```
scapy_attack.py ●
home > vboxuser > Documents > scapy_attack.py
1  from scapy.all import *
2  import time
3
4  # Target IP [victim-server]
5  target_ip = "172.21.0.10"
6
7  # ===== 1. ICMP Flood =====
8  def icmp_flood(count=100):
9      print("[*] Starting ICMP Flood...")
10     for i in range(count):
11         pkt = IP(dst=target_ip)/ICMP()
12         send(pkt, verbose=0)
13     print("[+] ICMP Flood completed.")
14
15  # ===== 2. UDP Flood =====
16  def udp_flood(count=100):
17      print("[*] Starting UDP Flood...")
18      for i in range(count):
19          pkt = IP(dst=target_ip)/UDP(dport=53)/Raw(load="A"*100)
20          send(pkt, verbose=0)
21      print("[+] UDP Flood completed.")
22
23  # ===== 3. Abnormal HTTP (raw TCP) =====
24  def http_attack():
25      print("[*] Sending abnormal HTTP request...")
26      pkt = IP(dst=target_ip)/TCP(dport=80, flags="S")/Raw(load="GET /malicious HTTP/1.1\r\nHost: victim\r\n\r\n")
27      send(pkt, verbose=0)
28      print("[+] Abnormal HTTP request sent.")
29
30  # ===== Main Program =====
31 if __name__ == "__main__":
32     icmp_flood()
33     time.sleep(2)
34     udp_flood()
35     time.sleep(2)
36     http_attack()
```

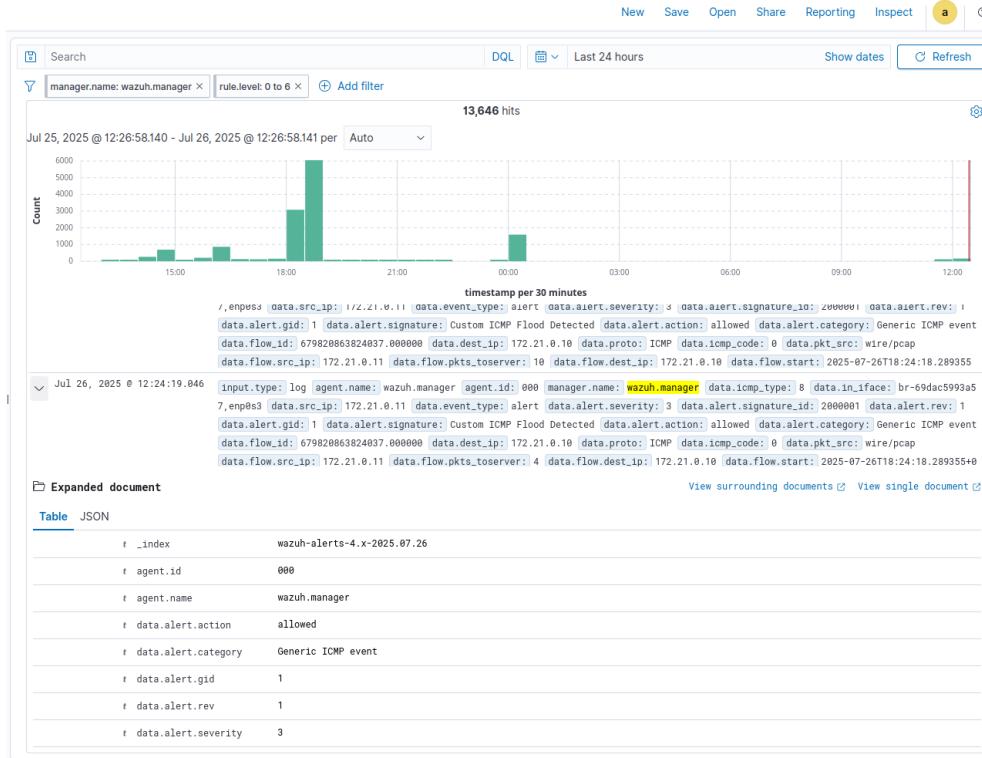
```
vboxuser@Ubuntu:~$ docker exec -it attacker-server bash
[~]# cd /root/
[~]# python3 scapy_attack.py
[*] Starting ICMP Flood...
[+] ICMP Flood completed.
[*] Starting UDP Flood...
[+] UDP Flood completed.
[*] Sending abnormal HTTP request...
[+] Abnormal HTTP request sent.
```

Next, we will use a Python script in the container to simulate ICMP Flood, UDP Flood, and abnormal HTTP request attacks to see if they can be detected by Wazuh.

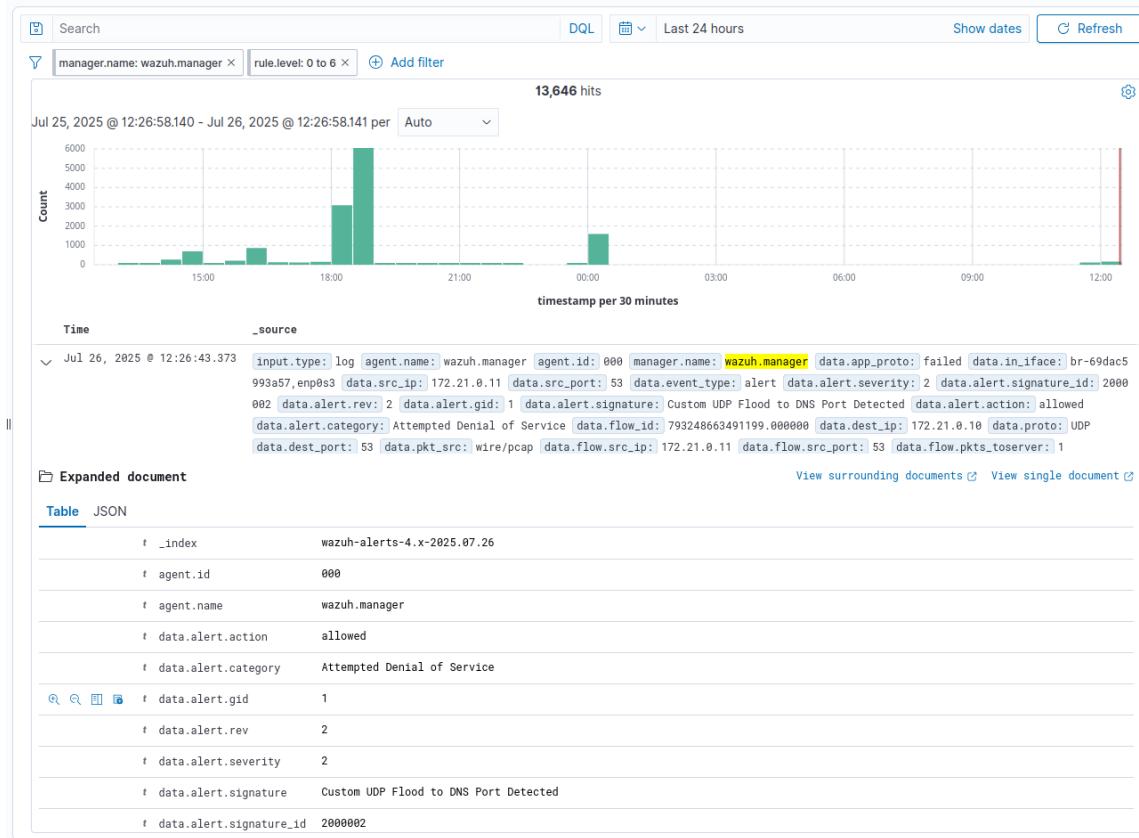
The picture below shows a screenshot of executing the attack script.

Open the Wazuh console to check the logs, and you will see that the related attack activities have been successfully recorded.

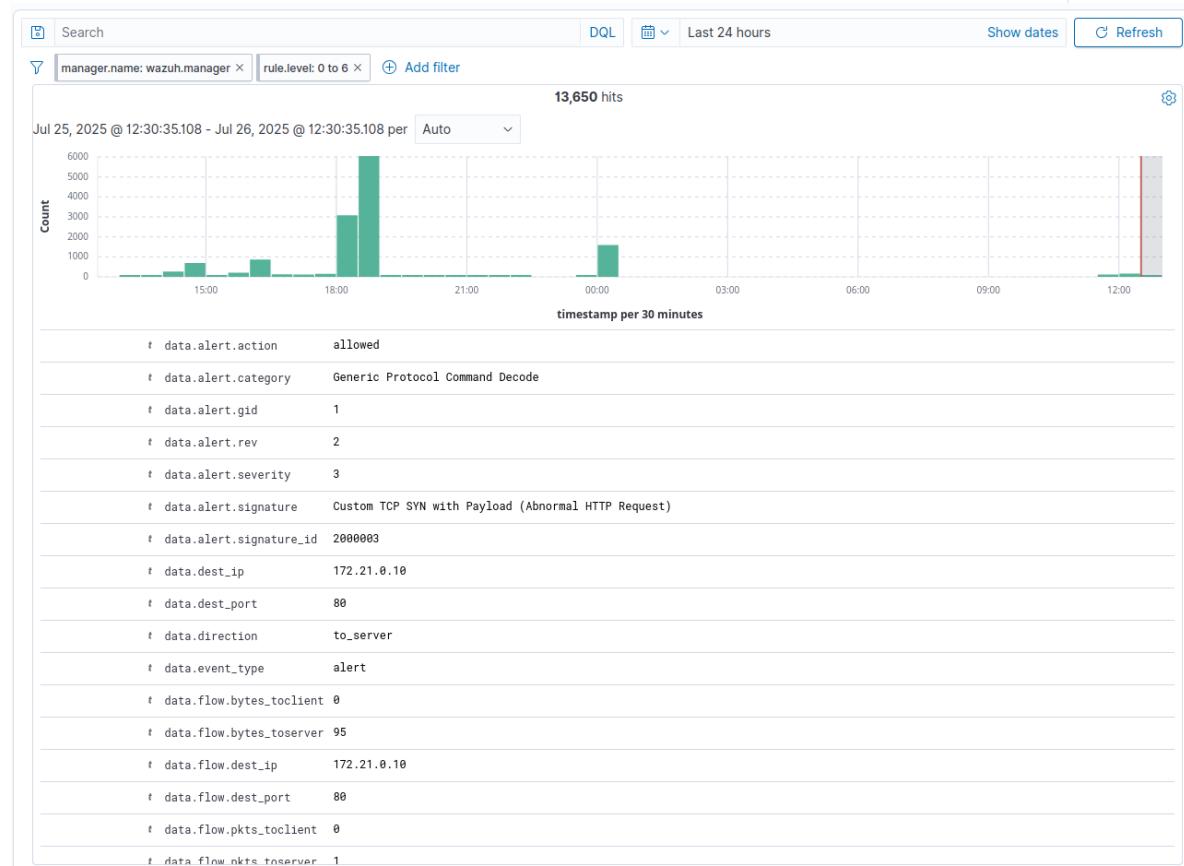
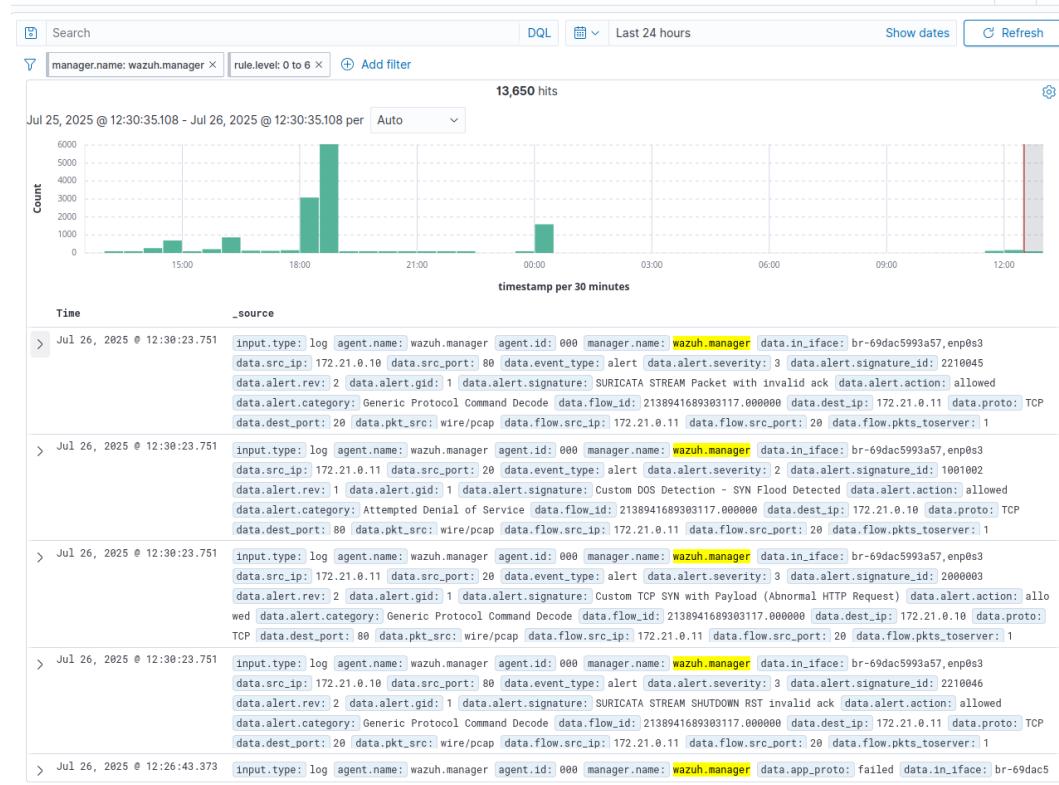
## ICMP Flood



## UDP Flood



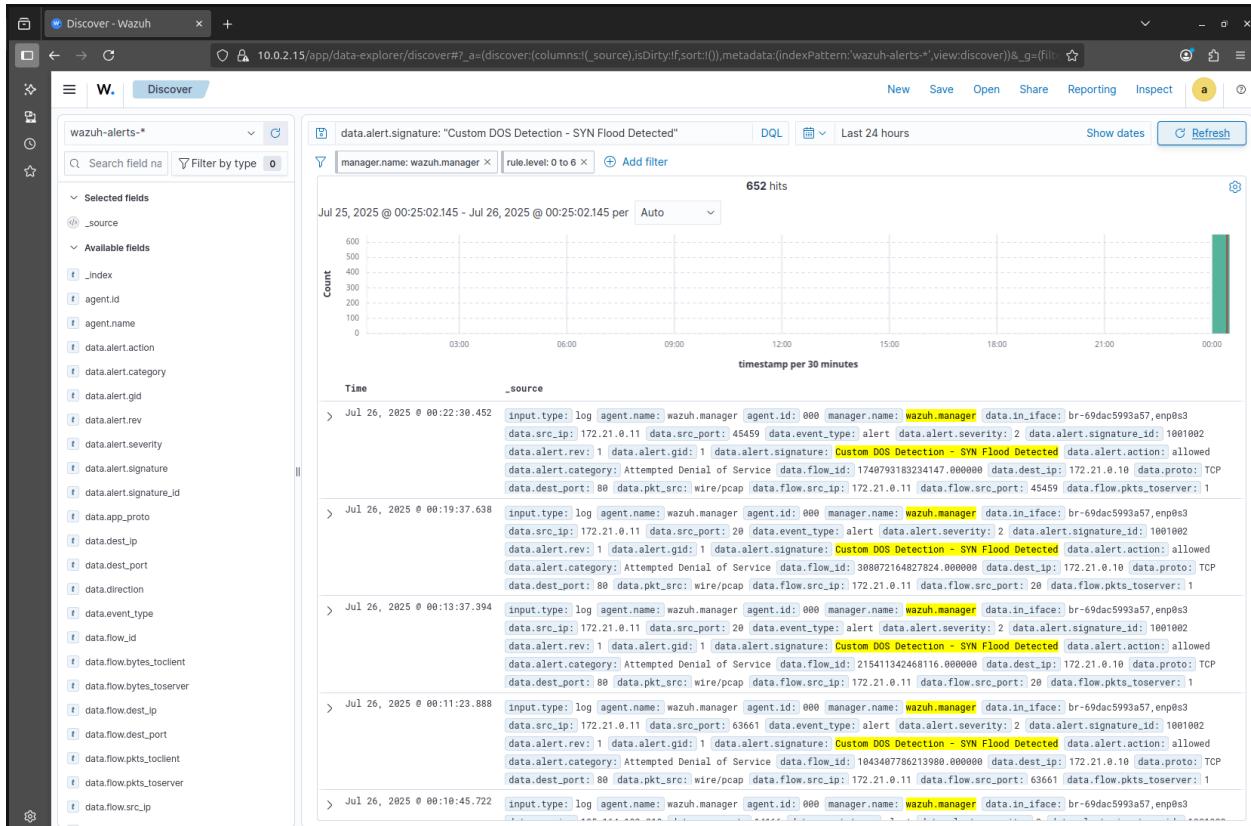
## Http attack



## Repository 1: wazuh-suricata-elk-docker - Function 8: Suricata detect - simulate DOS attacks

Upload the script to the attacker-server

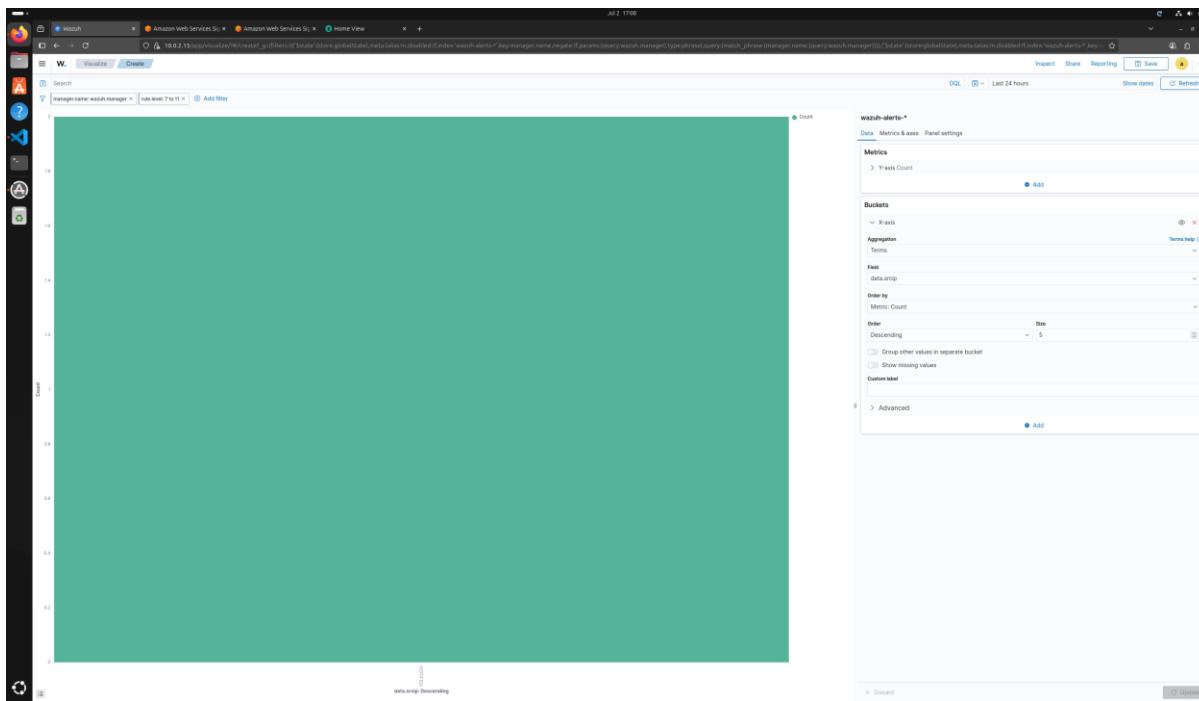
```
Open file in editor (ctrl + click)
● ~/Doc/w/wazuh-suricata-elk-docker | main ?3 docker cp /media/sf_Downloads/dos_attack_simulation.py a
ttacker-server:/opt/
Successfully copied 3.58kB to attacker-server:/opt/
○ ~/Doc/w/wazuh-suricata-elk-docker | main ?3 docker exec -it attacker-server bash ✓ | 12:24:39
```



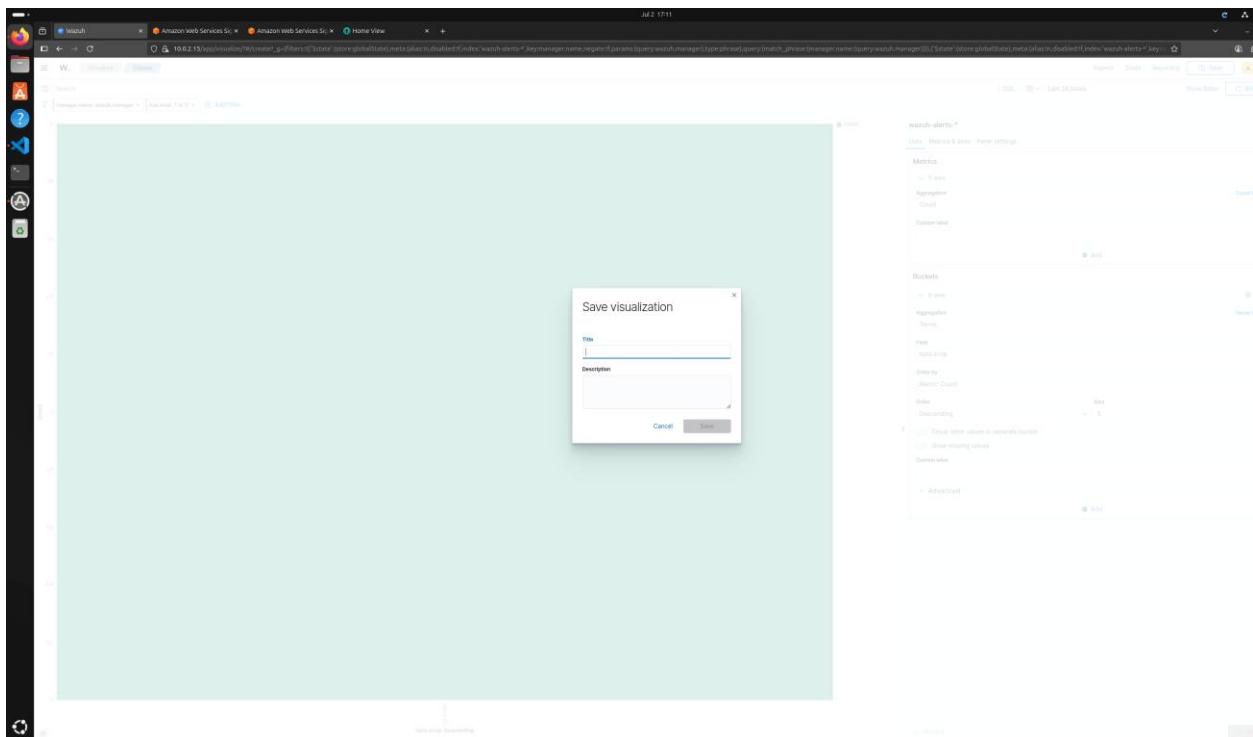
## Repository 1: wazuh-suricata-elk-docker - Function 9: How to use the Kibana dashboard

Add Top 5 IP bar chart to Kibana dashboard

Change the size of the order from 20 to 5



Click the save button after clicking the update button



×

## Save visualization

**Title**

**Description**

[Cancel](#) [Save](#)

Go to the dashboard management and click it, then go to saved objects can see the object that you added

The screenshot shows the Kibana Dashboards Management interface with the 'Saved objects' tab selected. On the left, there's a sidebar with 'Dashboards Management' and links for 'Index patterns', 'Data sources', 'Saved objects' (which is highlighted in blue), and 'Advanced settings'. The main area is titled 'Saved Objects' and contains a table of saved objects. The table has columns for 'Type', 'Title', 'Last updated', and 'Actions'. The data in the table is as follows:

Type	Title	Last updated	Actions
Advanced Settings [2.8.0]	Advanced Settings [2.8.0]	Jun 8, 2025 @ 20:39:38.419	
Advanced Settings [2.19.1]	Advanced Settings [2.19.1]	Jun 8, 2025 @ 23:10:40.968	
wazuh-monitoring-*	wazuh-monitoring-*	Jun 8, 2025 @ 20:39:32.785	
wazuh-statistics-*	wazuh-statistics-*	Jun 8, 2025 @ 20:39:32.826	
wazuh-alerts-*	wazuh-alerts-*	Jul 2, 2025 @ 16:34:52.850	
Top 5 Source IPs	Top 5 Source IPs	Jul 2, 2025 @ 17:12:21.896	

Below the table, there are buttons for 'Export all objects', 'Import', and 'Refresh'. At the bottom, there are buttons for 'Type', 'Delete', and 'Export'. The footer shows 'Rows per page: 50' and navigation arrows.

☰ | W. | Dashboards Ma... | Saved objects

**Recently viewed** > X

- Configuration Assessment
- Malware Detection
- File Integrity Monitoring

**Threat intelligence** >

**Security operations** > ▼

- PCI DSS
- GDPR
- HIPAA
- NIST 800-53
- TSC

**Cloud security** > ▼

- Docker
- Amazon Web Services
- Google Cloud
- GitHub
- Office 365

**Agents management** > ▼

- Summary
- Groups

**Server management** > ▼

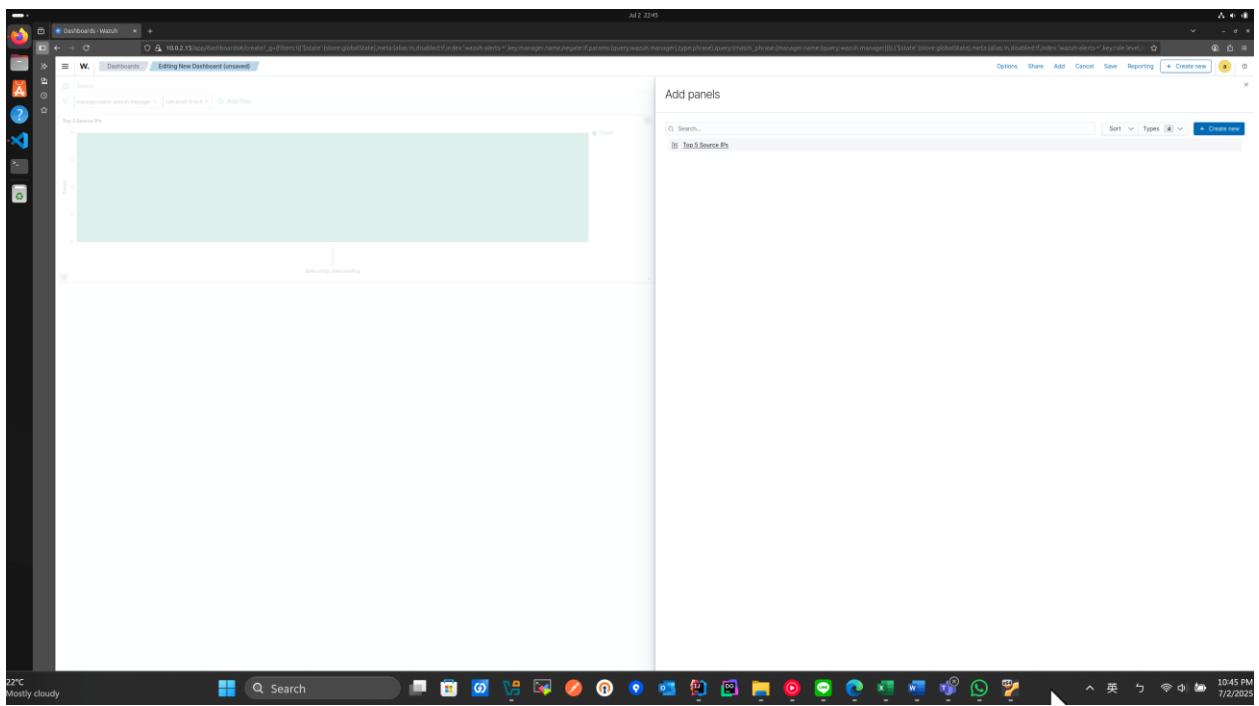
- Rules
- Decoders
- CDB Lists
- Status
- Cluster
- Statistics
- Logs
- Settings
- Dev Tools
- Ruleset Test
- Security

**Indexer management** >

**Dashboard management** > ▼

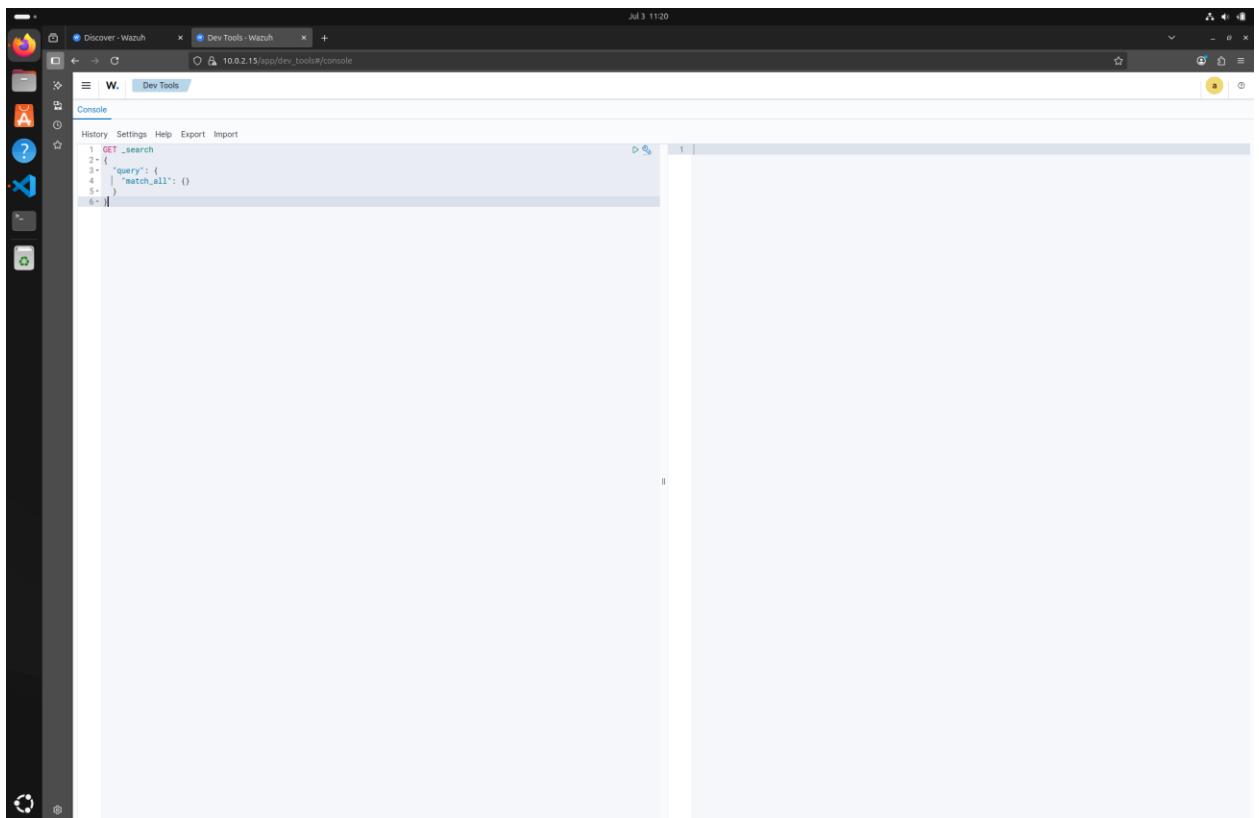
- Dashboards Management
- Reporting
- Server APIs
- App Settings
- About

Add the chart by clicking the Add button and selecting the one we created from the list.

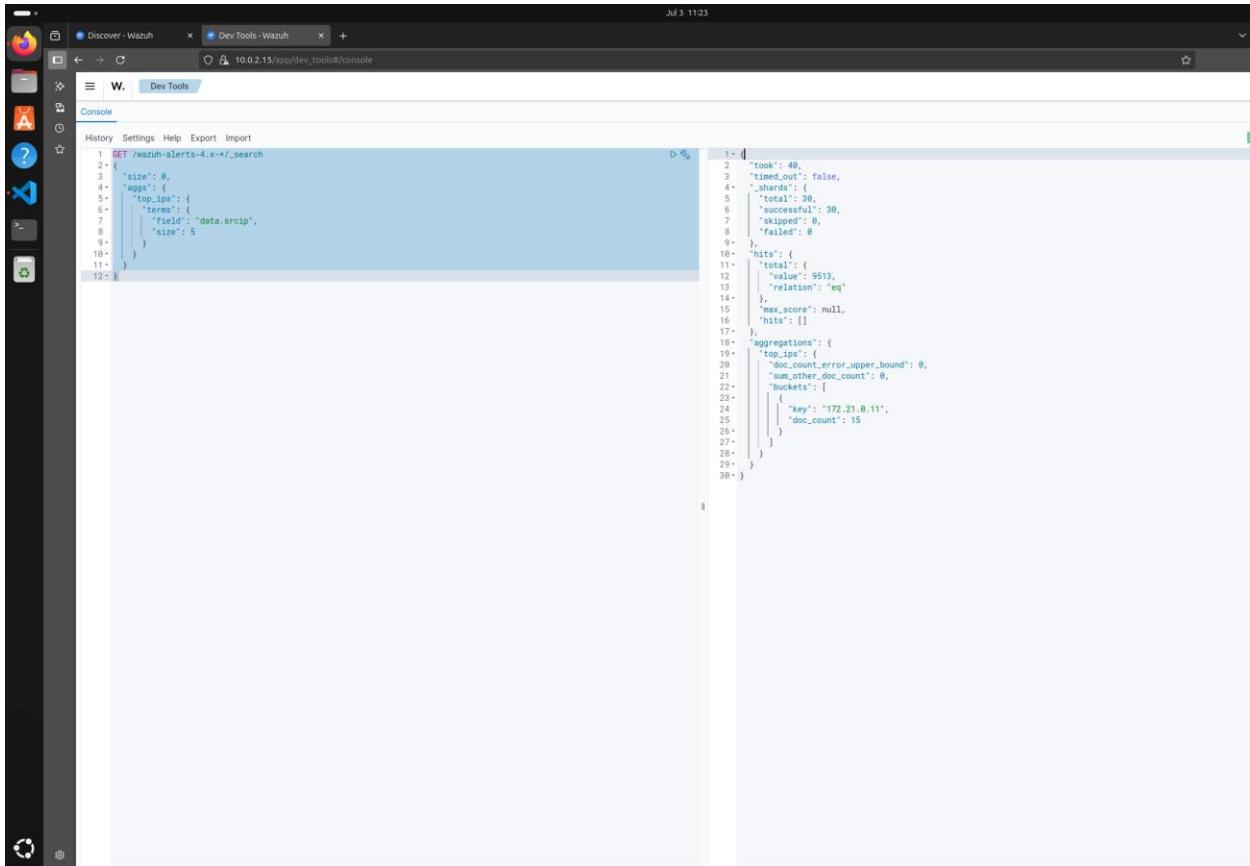


### Configure Elasticsearch aggregation for Top 5 IPs

Open the dev tools by typing the URL([https://10.0.2.15/app/dev\\_tools#/console](https://10.0.2.15/app/dev_tools#/console)).



Enter the query information to search for the top 5 IPs, and we will retrieve the collected data. Successfully queried wazuh-alerts-4.x-\* using Elasticsearch terms aggregation on data.srcip field. Verified IP frequency counts correctly, showing 172.21.0.11 appeared 15 times.



The screenshot shows the Wazuh Dev Tools interface. In the top bar, there are tabs for 'Discover - Wazuh' and 'Dev Tools'. The URL in the address bar is '10.0.2.15/app/dev\_tools#/console'. The main area is a code editor titled 'Console' with the following Elasticsearch query:

```
1 GET /wazuh-alerts-4.x-*/_search
2 {
3   "size": 0,
4   "aggs": {
5     "top_ips": {
6       "terms": {
7         "field": "data.srcip",
8         "size": 5
9       }
10      }
11    }
12 }
```

To the right of the code editor, the response from the Elasticsearch search is displayed as a JSON object:

```
1 {
2   "took": 48,
3   "timed_out": false,
4   "_shards": {
5     "total": 10,
6     "successful": 30,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 9513,
13      "relation": "eq"
14    },
15    "max_score": null,
16    "hits": []
17  },
18  "aggregations": {
19    "top_ips": {
20      "doc_count_error_upper_bound": 0,
21      "sum_other_doc_count": 0,
22      "buckets": [
23        {
24          "key": "172.21.0.11",
25          "doc_count": 15
26        }
27      ]
28    }
29  }
30 }
```

## *Test Kibana time and region filter*

### Time filter

The screenshot shows the Wazuh SOC Dashboard interface. At the top, there are tabs for 'Discover - Wazuh' and 'SOC Dashboard - Wazuh'. The main area is titled 'Top 5 Source IPs' and contains a message 'No results found'. The bottom right corner of the dashboard header shows the date and time as 'Jul 3 11:57'. The top right of the dashboard has buttons for 'Full screen', 'Share', 'Clone', 'Reporting', 'Edit', and a refresh icon.

### region filter

The screenshot shows the Dev Tools - Wazuh interface, specifically the 'Console' tab. The URL in the address bar is '10.0.2.15/app/dev\_tools/console'. The console window displays a single line of code: '1 GET /\_ingest/pipeline/geolp-info'. To the right of the code, the response pane shows the JSON configuration for the 'geolp-info' pipeline stage. The configuration includes a description ('Add geolp info to source IP'), processors, and a 'geolp' processor block with fields like 'data.geolp' and 'target\_field'. The status bar at the bottom right indicates a response time of '200 OK 18 ms'.

```
1 GET /_ingest/pipeline/geolp-info
2 {
3     "description": "Add geolp info to source IP",
4     "processors": [
5         {
6             "geolp": {
7                 "data_field": "data.geolp",
8                 "target_field": "data.geolp",
9                 "ignore_missing": true
10            }
11        }
12    ]
13}
14}
```

```
1- {
2-   "index_templates": [
3-     {
4-       "name": "wazuh-alerts-template",
5-       "index_template": {
6-         "index_patterns": [
7-           "wazuh-alerts-*"
8-         ],
9-         "template": {
10-           "mappings": {
11-             "geoip": {
12-               "index": {
13-                 "default_pipeline": "geoip-info"
14-               }
15-             }
16-           }
17-         },
18-         "composed_of": []
19-       }
20-     }
21-   ]
22- }
```

### Integrate GeoIP DB or API for IP location

```
curl -k -u
[REDACTED]:[REDACTED]
-X PUT
"https://wazuh.ind
exer:9200/_ingest/
pipeline/geoip-
info" -H 'Content-
Type:
application/json' -
d '{
application/json' -
d
"description":
```

You can add a GeoIP enrichment pipeline to transform IP addresses into geographic coordinates (latitude and longitude) by using Elasticsearch Ingest Pipelines with the GeoIP processor.

Here's how you can set up a pipeline called geoip-info that enriches documents with latitude and longitude from an IP address field.  
(Replace your INDEX\_USERNAME and INDEX\_PASSWORD)

"Add the geoip-info pipeline in filebeat.yml to ensure every Wazuh index document passes through this pipeline for IP-to-geolocation transformation."

```
bash-5.2# sed -i '/output.elasticsearch:/a \  pipeline: "geoip-info"' /etc/filebeat/filebeat.yml
bash-5.2# cat /etc/filebeat/filebeat.yml

# Wazuh - Filebeat configuration file
filebeat.modules:
- module: wazuh
  alerts:
    enabled: true
  archives:
    enabled: false

setup.template.json.enabled: true
setup.template.overwrite: true
setup.template.json.path: '/etc/filebeat/wazuh-template.json'
setup.template.json.name: 'wazuh'
setup.ilm.enabled: false
output.elasticsearch:
  pipeline: "geoip-info"
  hosts: ['https://wazuh.indexer:9200']
  username: '████████'
  password: '████████'
  ssl.verification_mode: 'full'
  ssl.certificateAuthorities: ['/etc/ssl/root-ca.pem']
  ssl.certificate: '/etc/ssl/filebeat.pem'
  ssl.key: '/etc/ssl/filebeat.key'

logging.metrics.enabled: false

seccomp:
  default_action: allow
  syscalls:
  - action: allow
    names:
    - rseq
bash-5.2# pkill filebeat
```

restart the wazuh-server-wazuh.manager-1

Add fake data (Since the Docker environment is isolated, I used mock data to simulate global IP traffic. If you want to test external traffic, you need to modify the Docker configuration to expose port 22 on the victim-server container, and also adjust the firewall settings to allow incoming traffic on port 22.)

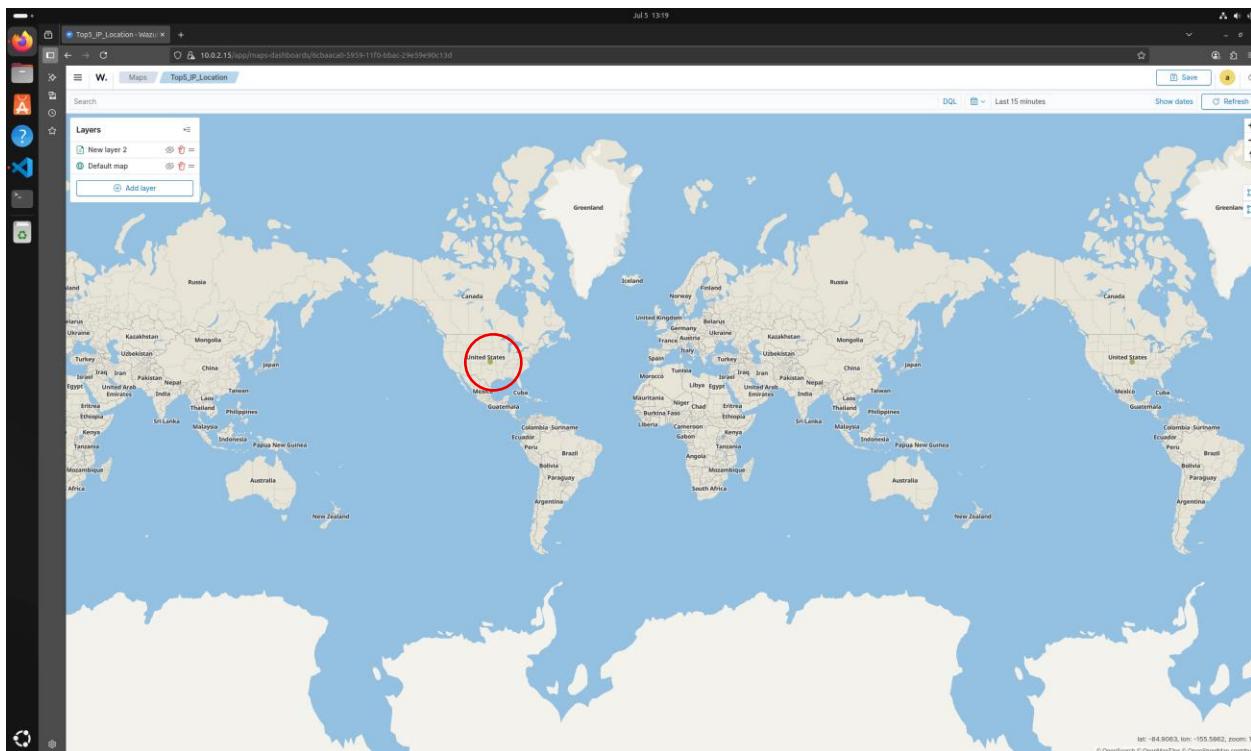
```
curl -k -u INDEX_USERNAME:INDEX_PASSWORD -X
POST "https://wazuh.indexer:9200/wazuh-alerts-4.x-
2025.07.26/_doc?pipeline=geoip-info" -H 'Content-
Type: application/json' -d '{
    "@timestamp": """$(date -u +%Y-%m-
%dT%H:%M:%SZ)""",
    "timestamp": """$(date -u +%Y-%m-
%dT%H:%M:%SZ)""",
    "data": {
        "srcip": "2.2.3.4"
    }
}'
```

Check that the fake data on the dashboard includes longitude and latitude.

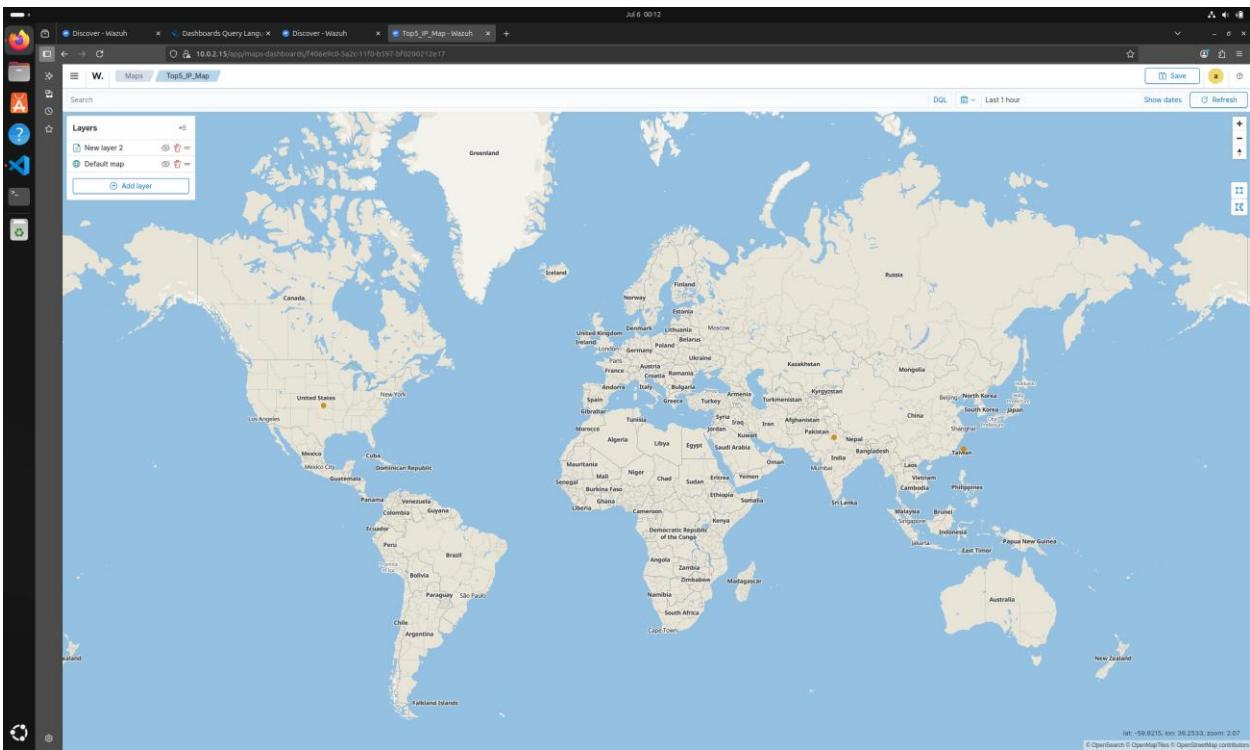
The screenshot shows the Vizual search interface with a document titled "test-index". The document contains the following JSON data:

```
{
  "_index": "test-index",
  "data_srcip": "8.8.8.8",
  "geop.continent_name": "North America",
  "geop.country_iso_code": "US",
  "geop.country_name": "United States",
  "geop.location": {
    "lon": -97.822,
    "lat": 37.751
  }
}
```

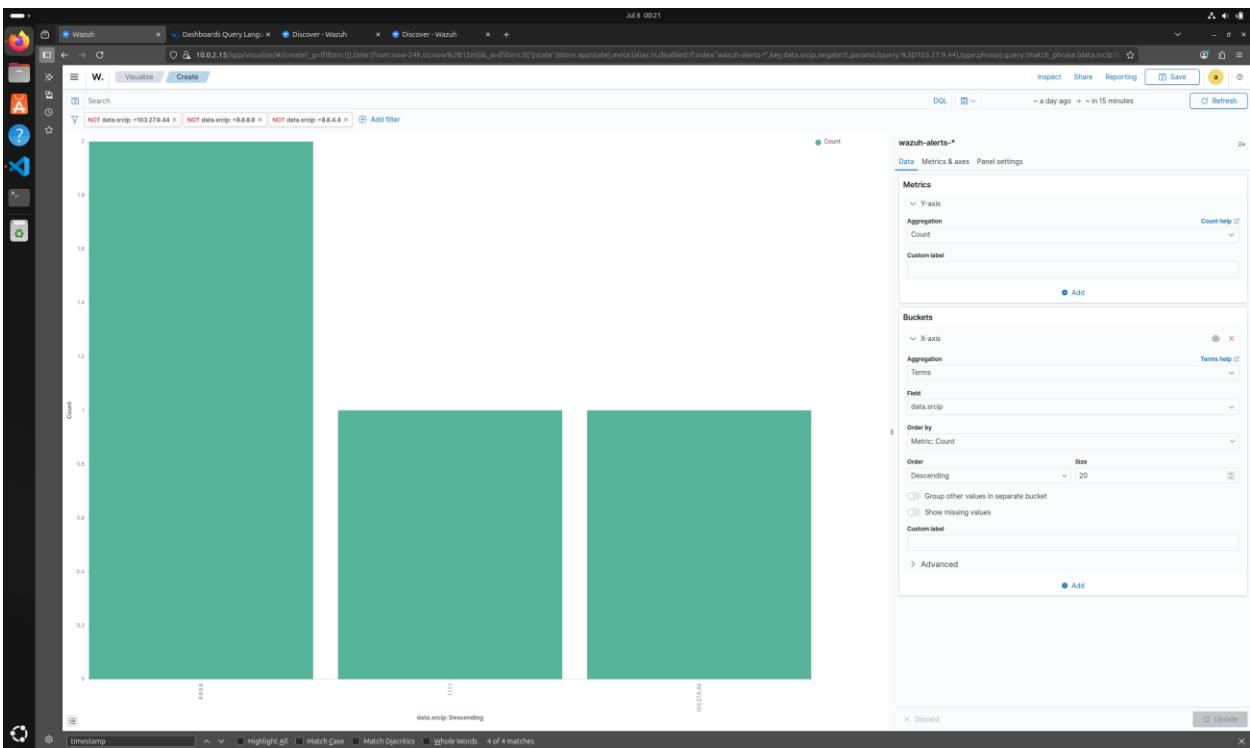
It will also show on the Map.



## Add geographic map panel in Kibana



## Configure bar chart parameters



Go to the index dashboard and search for the top 5 most frequent IP addresses.

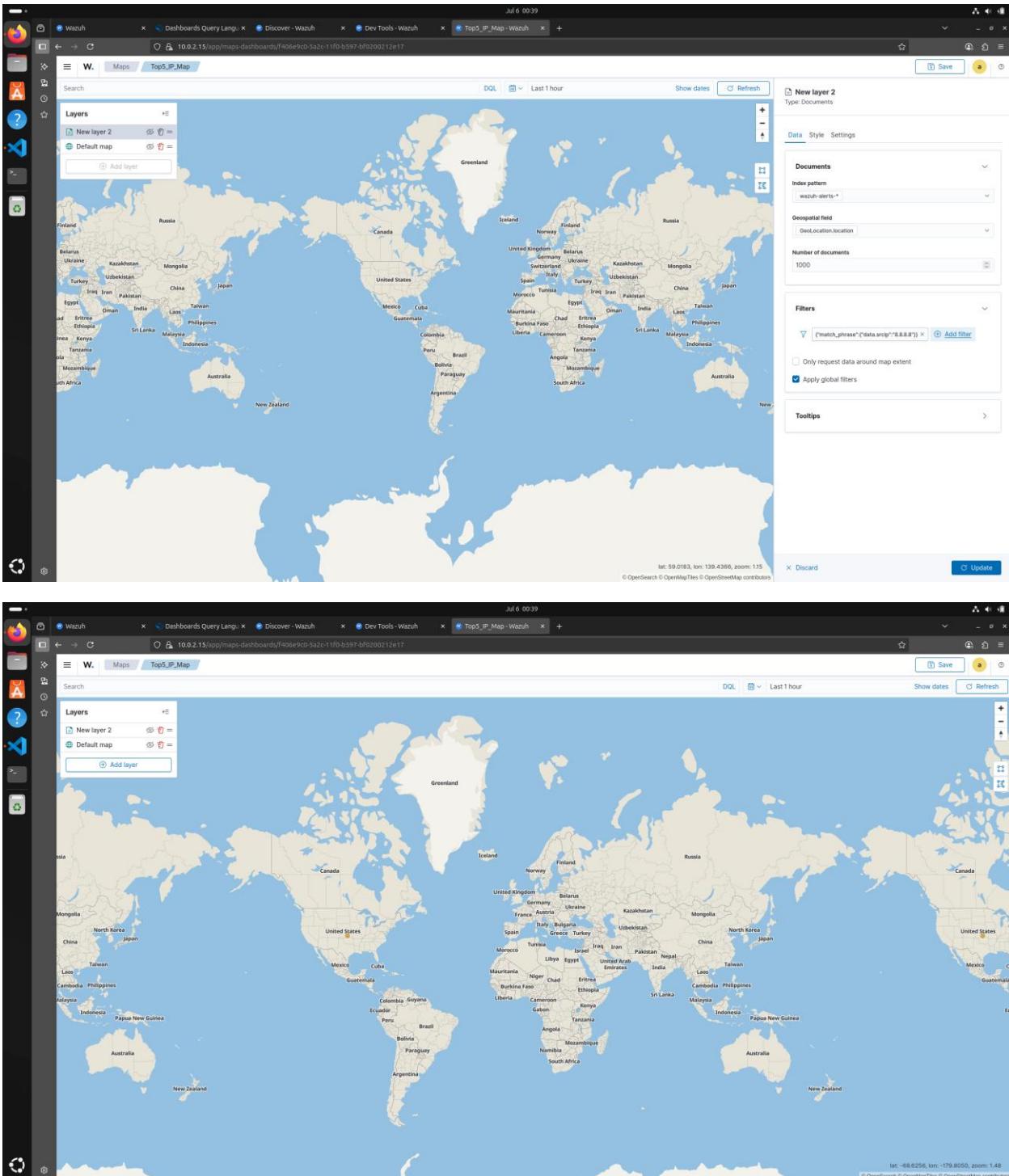
```
17
18 GET /wazuh-alerts-4.x-*/_search
19 {
20   "size": 0,
21   "aggs": {
22     "top_ips": {
23       "terms": {
24         "field": "data.srcip",
25         "size": 5
26       }
27     }
28   }
29 }
30
```

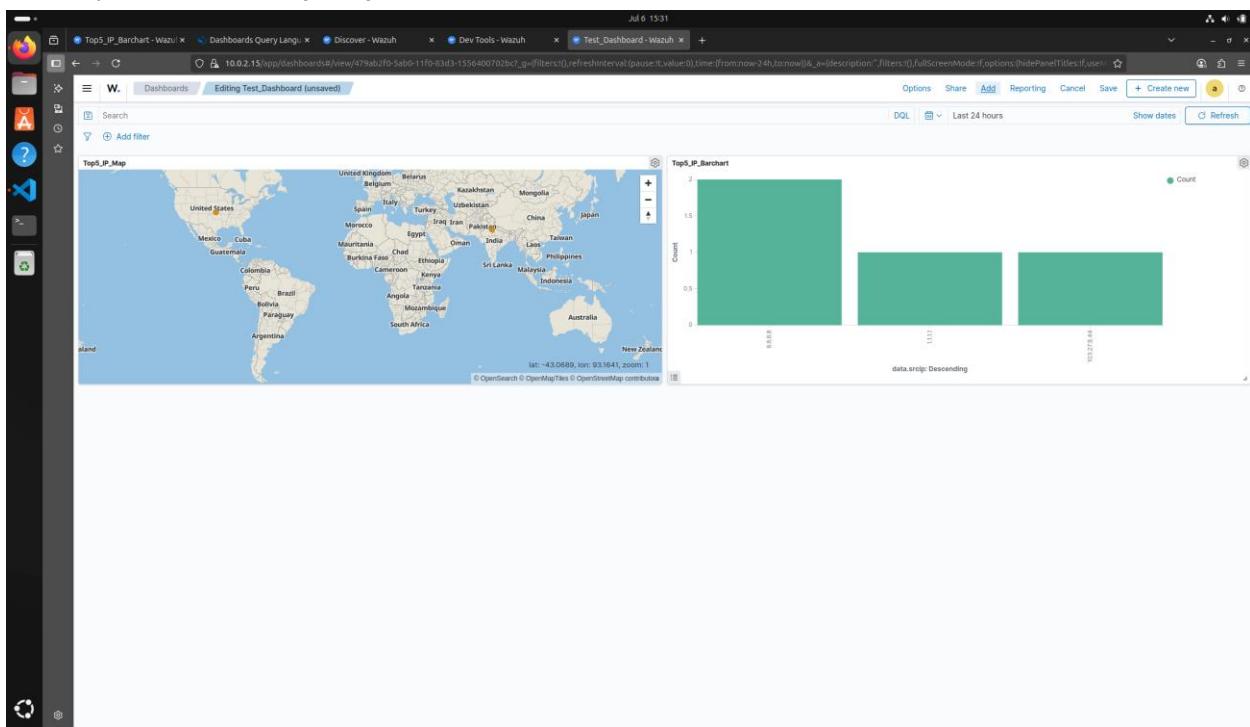
```
1 {
2   "took": 6,
3   "timed_out": false,
4   "_shards": {
5     "total": 6,
6     "successful": 6,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 6514,
13      "relation": "eq"
14    },
15    "max_score": null,
16    "hits": []
17  },
18  "aggregations": {
19    "top_ips": {
20      "doc_count_error_upper_bound": 0,
21      "sum_other_doc_count": 4,
22      "buckets": [
23        {
24          "key": "8.8.8.8",
25          "doc_count": 10
26        },
27        {
28          "key": "172.21.0.1",
29          "doc_count": 5
30        },
31        {
32          "key": "172.21.0.11",
33          "doc_count": 5
34        },
35        {
36          "key": "1.1.1.1",
37          "doc_count": 2
38        },
39        {
40          "key": "=8.8.8.8",
41          "doc_count": 2
42        }
43      ]
44    }
45  }
46 }
```

## Configure map to filter charts on region click

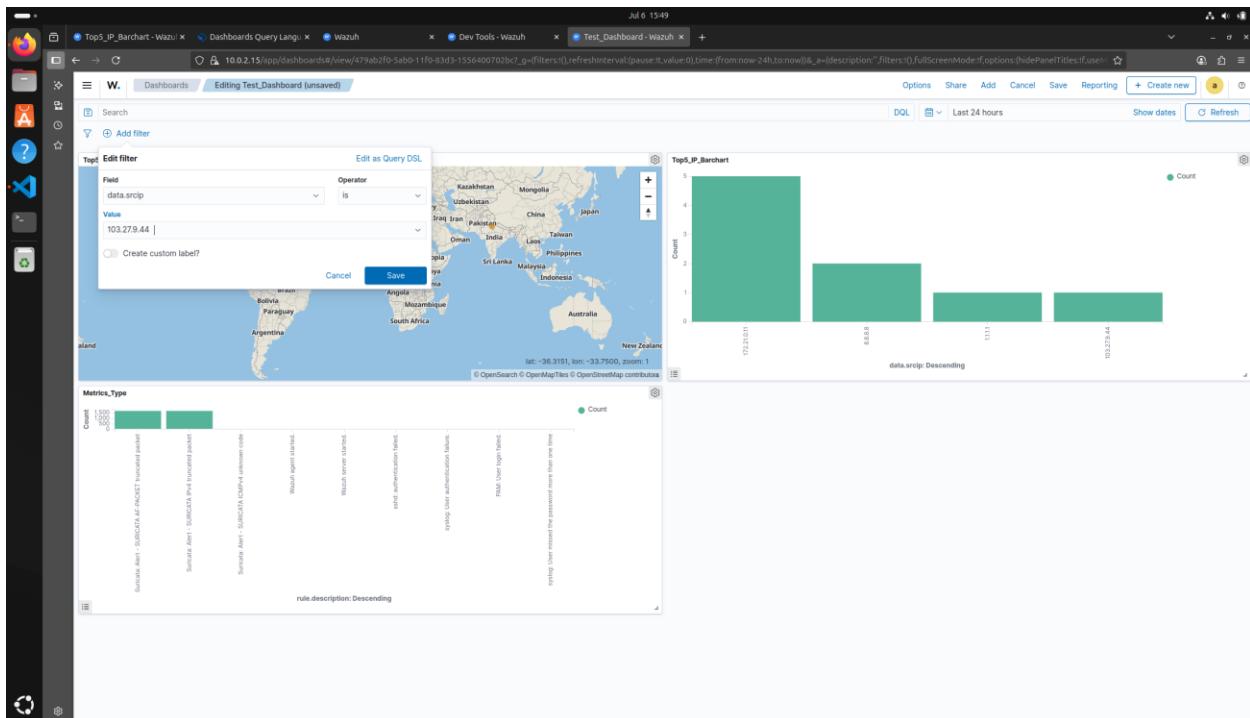
Add the filter, which is data. srchip to 8.8.8.8 and click the update button; it will only show the point in the US.

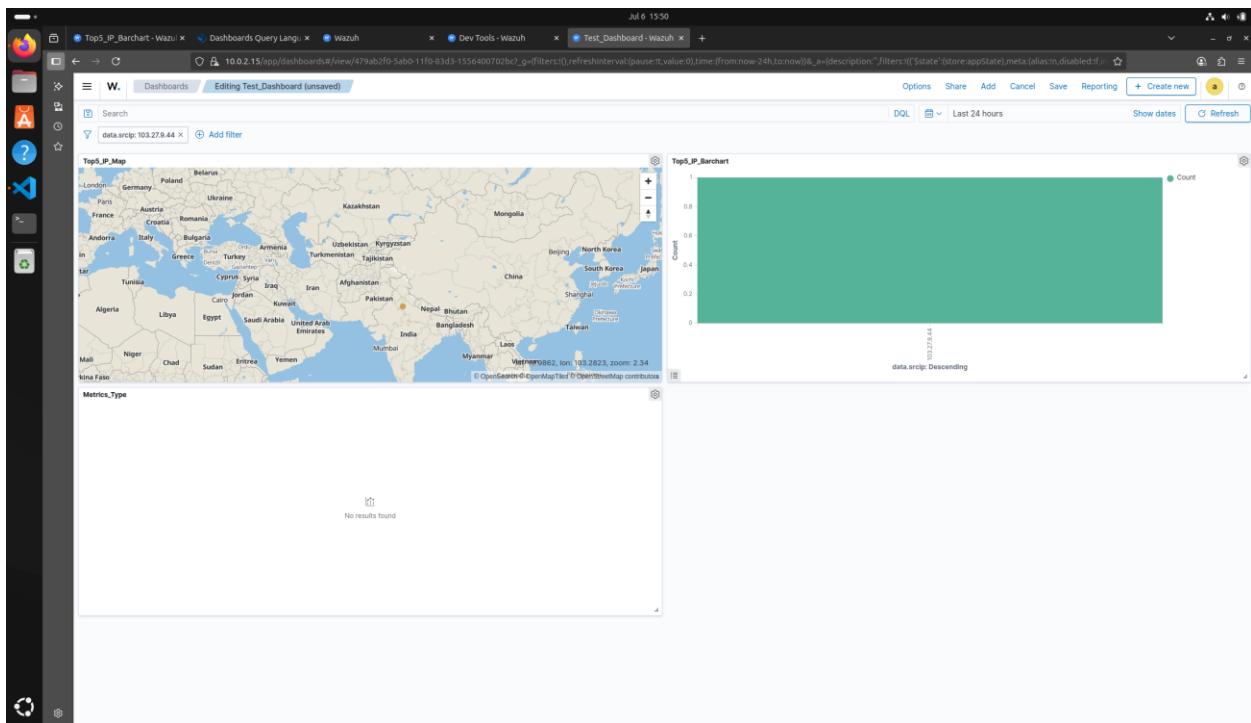


## Test map interaction and filter function



Add the filter (data.srcip = 103.27.9.44) and click the save button, and it will only show the related information on the dashboard.





## Repository 2: log-alert-pipeline - Function 1: Custom Python parser filtering rules

```

main.py      alert_rule.yaml ● settings.yaml
config > ! alert_rule.yaml
config > ! alert_rule.yaml
1  rules:
2    - rule_id: 1
3      rule_name: "Suspicious Port Scan"
4      rule_description: "Detects if a source IP is scanning multiple ports on one or more destinations within a time window."
5      rule_actions:
6        - action: "email"
7          email: "maggie.555x@gmail.com"
8          template: "Suspicious Scan Detected: ${src_ip} is scanning ${dest_ip}. Hit ${dest_port_count} unique ports in the last 60 seconds."
9      rule_score: 5
10     filter:
11       - field: "data.flow.reason"
12         operator: "eq"
13         value: "timeout"
14       - field: "data.tcp.rst"
15         operator: "eq"
16         value: "true"
17     aggregation:
18       type: "port_scan"
19       time_window: 60
20       group_by: "data.src_ip"
21       unique_count_field: "data.dest_port"
22       threshold: 5
23

```

This document explains the structure and usage of the alert\_rule.yaml file. This configuration file is used to define a set of rules for a log analysis and alerting pipeline. Each rule specifies conditions to detect specific patterns in log data and actions to take when these patterns are found.

### Rule Schema

The configuration file contains a list of rules under the rules key. Each rule is a dictionary with the following fields:

Key	Type	Required	Description
rule_id	Integer	Yes	A unique identifier for the rule.
rule_name	String	Yes	A human-readable name for the rule.
rule_description	String	No	A detailed description of what the rule is designed to detect.
rule_actions	List of Objects	Yes	A list of actions to be executed when the rule's conditions are met.
rule_score	Integer	Yes	A severity score for the alert (e.g., 1-10), which can be used for prioritization.
filter	List of Objects	Yes	A list of filter conditions that log entries must satisfy. All conditions must be met (logical AND).
aggregation	Object	Yes	Defines the logic for aggregating data over a time window to detect complex patterns.

### rule\_actions

This section defines what happens when a rule is triggered.

Key	Type	Required	Description
action	String	Yes	The type of action to perform. Currently, email is a supported value.
email	String	Yes	The recipient's email address for the notification. Required if action is email.
template	String	Yes	A message template for the notification. It can contain variables that will be substituted with values from the aggregated alert data.

**Template Variables:** Variables in the template string are denoted by \${variable\_name}. These are dynamically replaced with context from the triggered alert. For example, \${src\_ip} would be replaced by the source IP address that triggered the rule.

## filter

The filter section specifies the criteria for selecting log entries that are relevant to the rule before aggregation. It's a list of conditions, and a log entry must match all of them.

Key	Type	Required	Description
field	String	Yes	The dot-notation path to a field within the log data (e.g., data.tcp.rst).
operator	String	Yes	The comparison operator. Currently, eq (equals) is supported.
value	String	Yes	The value to compare the field against.

## aggregation

The aggregation block defines how filtered log entries are grouped and counted over a period to detect patterns.

Key	Type	Required	Description
type	String	Yes	The type of aggregation. The value port_scan is used for detecting port scanning activity.
time_window	Integer	Yes	The duration in seconds over which to aggregate the data.
group_by	String	Yes	The field to group the log entries by (e.g., data.src_ip).
unique_count_field	String	Yes	The field for which to count the number of unique values within the group (e.g., data.dest_port).
threshold	Integer	Yes	The minimum number of unique values for the unique_count_field that must be met to trigger the alert.

### *Example Rule Explained: "Suspicious Port Scan"*

Let's break down the example rule provided in alert\_rule.yaml:

#### **How it works:**

1. **Filter:** The system first looks for log entries where data.flow.reason is timeout **OR** data.tcp.rst is true. (Assuming the logic combines these with OR for this specific scan detection, if it's AND the log must have both).
2. **Aggregation:** It then aggregates these filtered entries over a **60-second window**.
  - It groups the entries by the source IP address (data.src\_ip).
  - Within each group, it counts the number of unique destination ports (data.dest\_port).
3. **Trigger:** If, for any single source IP, the count of unique destination ports exceeds **5** within the 60-second window, the rule is triggered.
4. **Action:** When triggered, an email is sent to Maggie.555x@gmail.com with a subject and body generated from the template, providing details about the scan.

#### *How to Add a New Rule*

To add a new rule, follow these steps:

1. Copy and paste an existing rule block within the rules list.
2. Assign a new, unique rule\_id.
3. Update rule\_name, rule\_description, and rule\_score to reflect the new rule's purpose and severity.
4. Modify the filter list to define the specific conditions for the log data you want to analyze.
5. Configure the aggregation parameters (time\_window, group\_by, unique\_count\_field, and threshold) to define the pattern you want to detect.
6. Set up the desired rule\_actions to specify what should happen when the rule is triggered.
7. Save the alert\_rule.yaml file. The alerting pipeline should automatically pick up the new rule.

#### *Run the Python Parser*

Put the attack\_scan\_rule\_1.sh on the attacker-server.

```
~/Documents/workspace/log-alert-pipeline | main !1 ?2 docker cp scripts/attack_scan_rule_1.sh attacker-server:/opt/
Successfully copied 2.56kB to attacker-server:/opt/
```

```
~/Documents/workspace/wazuh-suricata-elk-docker | main ?3 docker exec -it attacker-server bash
└─(root㉿4c8b503fccca2)-[/>
# ls /opt/
attack_scan_rule_1.sh  venv
```

```

venv -- (venv) (maggie㉿kali)-[~/ttt/log-alert-pipeline]
└─$ python src/main.py
2025-06-14 16:47:26,481 - alert_analyzer - INFO - Loaded 1 alert rules from config/alert_rule.yaml
2025-06-14 16:47:26,482 - __main__ - INFO - Starting Wazuh data fetcher with 1-minute interval
Wazuh Data Fetcher started - fetching every 1 minutes
Press Ctrl+C to stop
2025-06-14 16:47:26,482 - __main__ - INFO - Fetching 20 recent alerts...
2025-06-14 16:47:26,483 - wazuh_client - INFO - Getting recent alerts with query: {'query': {'match_all': {}}, 'sort': [{"@timestamp": {"order": "desc"}}], 'size': 20}
2025-06-14 16:47:26,593 - alert_analyzer - INFO - Analyzing 20 alerts against 1 rules
2025-06-14 16:47:26,601 - alert_analyzer - INFO - Found 0 matching alerts

```

□

timestamp per 30 minutes			
Time	data.dest_ip	data.src_ip	rule.description
> Jun 14, 2025 @ 13:37:39.507	10.0.0.131	172.21.0.3	Suricata: Alert - SURICATA STREAM excessive retransmissions
> Jun 14, 2025 @ 13:37:39.503	10.0.0.131	172.21.0.3	Suricata: Alert - SURICATA STREAM excessive retransmissions
Jun 14, 2025 @ 10:00:01.031	172.21.0.4	172.21.0.3	Suricata: Alert - SURICATA STREAM Packet with invalid timestamp
> Jun 14, 2025 @ 05:10:20.439	10.0.0.131	172.21.0.3	Suricata: Alert - SURICATA STREAM suspected RST injection
> Jun 14, 2025 @ 05:09:53.121	-	-	Wazuh server started.
> Jun 14, 2025 @ 05:06:20.008	8.8.8.8	172.21.0.11	Suricata: Traffic observed from monitored IP 172.21.0.11 to 8.8.8.8.
> Jun 14, 2025 @ 05:05:49.961	8.8.8.8	172.21.0.11	Suricata: Traffic observed from monitored IP 172.21.0.11 to 8.8.8.8.
> Jun 14, 2025 @ 05:02:41.509	172.21.0.10	172.21.0.11	Suricata: Traffic observed from monitored IP 172.21.0.11 to 172.21.0.10.

The data inside the red box will be filtered out.

```

Problems Output Debug Console Terminal Ports ⓘ
Wazuh Data Fetcher started - fetching every 1 minutes
Press Ctrl+C to stop
2025-07-12 03:26:12,607 - __main__ - INFO - Fetching 2000 recent alerts...
2025-07-12 03:26:12,608 - wazuh_client - INFO - Getting recent alerts with query: {'query': {'match_all': {}}, 'sort': [{"@timestamp": {"order": "desc"}}, 'size': 2000}
2025-07-12 03:26:15,740 - alert_analyzer - INFO - Analyzing 2000 alerts against 1 rules
2025-07-12 03:26:16,026 - alert_analyzer - INFO - Found 1 matching alerts
2025-07-12 03:26:17,283 - notifier - INFO - Successfully sent alert email to cythhy@gmail.com

2025-07-12 03:27:13,513 - __main__ - INFO - Fetching 2000 recent alerts...
2025-07-12 03:27:13,526 - wazuh_client - INFO - Getting alerts by time range (2025-07-12T07:26:13.000Z - 2025-07-12T07:27:13.000Z) with query: {'query': {'range': {'@timestamp': {'gte': '2025-07-12T07:26:13.000Z', 'lt': '2025-07-12T07:27:13.000Z'}}}, 'sort': [{"@timestamp": {"order": "desc"}}, 'size': 2000}
2025-07-12 03:27:13,880 - alert_analyzer - INFO - Analyzing 2000 alerts against 1 rules
2025-07-12 03:27:14,019 - alert_analyzer - INFO - Found 1 matching alerts
2025-07-12 03:27:15,263 - notifier - INFO - Successfully sent alert email to cythhy@gmail.com

```

Schedule running effects

```

main.py ! alert_rule.yaml ! settings.yaml
config > ! alert_rule.yaml
config > ! settings.yaml
config > ! alert_rule.yaml
  1 > rules:
  2   - rule_id: 1
  3     rule_name: "Suspicious Port Scan"
  4     rule_description: "Detects if a source IP is scanning multiple ports on one or more destinations within a time window."
  5     rule_actions:
  6       - action: "email"
  7         email: "maggie.555x@gmail.com"
  8         template: "Suspicious Scan Detected: $(src_ip) is scanning $(dest_ip). Hit $(dest_port_count) unique ports in the last 60 seconds."
  9     rule_score: 5
 10    filter:
 11      - field: "data.flow.reason"
 12        operator: "eq"
 13        value: "timeout"
 14      - field: "data.tcp.rst"
 15        operator: "eq"
 16        value: "true"
 17    aggregation:
 18      type: "port_scan"
 19      time_window: 60
 20      group_by: "data.src_ip"
 21      unique_count_field: "data.dest_port"
 22      threshold: 5
 23

```

alert rule

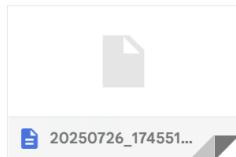
## Alert Summary - Account Information Change Detected

**postmaster@sa...** Sat, Jul 26, 5:45PM (3 days ago)

to me ▾  
Account Information Change Detected: User 'testuser2' information was modified.  
Rule ID: 6, Rule Score: 6.

Total Alerts: 1

One attachment • Scanned by Gmail



### Alert Email

Timestamp	Rule ID	Rule Name	Src IP	Dest IP	Dest Port	Src User	Dst User	Command	cription
1291	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1292	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1293	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1294	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1295	7/26/2025 12:18	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1296	7/26/2025 0:22	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1297	7/26/2025 0:22	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1298	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1299	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1300	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1301	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1302	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1303	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1304	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1305	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1306	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1307	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1308	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1309	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1310	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1311	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1312	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1313	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1314	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1315	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1316	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1317	7/26/2025 0:19	3 ICMP Flood Detected	172.21.0.11	172.21.0.10					Suricata: Alert - Custom ICMP Flood Detected
1318	7/26/2025 12:26	4 UDP Flood to DNS Port Detected	172.21.0.11	172.21.0.10	53				Suricata: Alert - Custom UDP Flood to DNS Port Detected
1319	7/26/2025 0:19	4 UDP Flood to DNS Port Detected	172.21.0.11	172.21.0.10	53				Suricata: Alert - Custom UDP Flood to DNS Port Detected
1320	7/26/2025 16:41	5 SSH Authentication Failure Detected	172.21.0.11			root			slog: User missed the password more than one time Information from the user was changed.
1321	7/26/2025 17:12	6 Account Information Change Detected				testuser2			
1322	7/26/2025 17:16	7 Successful Sudo to Root Detected				testuser2	root	/usr/bin/vim /etc/shadow	Successful sudo to ROOT executed.
1323	7/26/2025 17:15	7 Successful Sudo to Root Detected				testuser2	root	/usr/bin/apt install vim	Successful sudo to ROOT executed.
1324	7/26/2025 17:14	7 Successful Sudo to Root Detected				root	root	/usr/bin/newgrp sudo	Successful sudo to ROOT executed.

Csv file

```

},
{
  "Timestamp": "2025-07-26T18:26:43.373Z",
  "Rule ID": 4,
  "Rule Name": "UDP Flood to DNS Port Detected",
  "Src IP": "172.21.0.11",
  "Dest IP": "172.21.0.10",
  "Dest Port": "53",
  "Src User": "",
  "Dst User": "",
  "Command": "",
  "Description": "Suricata: Alert - Custom UDP Flood to DNS Port Detected"
},
{
  "Timestamp": "2025-07-26T06:19:33.001Z",
  "Rule ID": 4,
  "Rule Name": "UDP Flood to DNS Port Detected",
  "Src IP": "172.21.0.11",
  "Dest IP": "172.21.0.10",
  "Dest Port": "53",
  "Src User": "",
  "Dst User": "",
  "Command": "",
  "Description": "Suricata: Alert - Custom UDP Flood to DNS Port Detected"
},
{
  "Timestamp": "2025-07-26T22:41:36.473Z",
  "Rule ID": 5,
  "Rule Name": "SSH Authentication Failure Detected",
  "Src IP": "172.21.0.11",
  "Dest IP": "",
  "Dest Port": "",
  "Src User": "",
  "Dst User": "root",
  "Command": "",
  "Description": "syslog: User missed the password more than one time"
},
{
  "Timestamp": "2025-07-26T23:12:43.060Z",
  "Rule ID": 6,
  "Rule Name": "Account Information Change Detected",
  "Src IP": "",
  "Dest IP": "",
  "Dest Port": "",
  "Src User": "",
  "Dst User": "testuser2",
  "Command": "",
  "Description": "Information from the user was changed."
},
{
  "Timestamp": "2025-07-26T23:16:19.984Z",
  "Rule ID": 7,
  "Rule Name": "Successful Sudo to Root Detected",
  "Src IP": "",
  "Dest IP": "",
  "Dest Port": "",
  "Src User": "testuser2",
  "Dst User": "root",
  "Command": "/usr/bin/vim /etc/shadow",
  "Description": "Successful sudo to ROOT executed."
}

```

JSON file

### *Detection Logic (Aggregation-Based)*

The core detection mechanism for this rule is based on the **aggregation** of network event data. It does not use the filter block, as explained in the final section.

The logic operates as follows:

- **Grouping:** It aggregates all incoming data by the source IP address (group\_by: "data.src\_ip").
- **Time Window:** The analysis is performed within a 60-second time window (time\_window: 60).
- **Trigger Condition:** The rule is triggered if a single source IP is observed communicating with a number of unique destination ports (unique\_count\_field: "data.dest\_port") that exceeds the threshold of 5.

In essence, if one source IP address attempts to connect to more than five unique ports on any destination(s) within a 60-second period, the system identifies this behavior as a suspicious port scan.

### *Alerting Mechanism (Rule Actions)*

When the rule's conditions are met, it can execute a predefined action. The rule\_actions section is configurable to define this response.

- **Action Type:** The currently supported action is email.
- **Configuration:** You can specify the recipient's email address and a message template for the alert. The template supports dynamic variables, such as \$(src\_ip) and \$(dest\_port\_count), to provide rich, contextual information in the notification.

### *Important Implementation Note: Logic Priority*

A critical aspect of our rule engine's architecture is the evaluation priority between logic blocks.

- **aggregation has higher priority than filter.**
- When both an aggregation block and a filter block are defined within the same rule, the engine will **only** evaluate and execute the aggregation logic. The filter block will be ignored. For this rule, the aggregation logic is what takes effect.

## 4.3 Data Import / Export (if applicable)

### Supported Import Formats

N/A

### Supported Export Formats

Format	Description
Type	
JSON (.json)	Exports complete alert information, suitable for advanced analysis and integration with other systems.
CSV (.csv)	Exports simplified alert records, ideal for importing into Excel for report analysis.

## 5. Troubleshooting

### 5.1 Issue 1: Wazuh Dashboard cannot be accessed (Page Not Found / Connection Refused)

- **Symptoms:**

When accessing `https://{{your_vm_IP}}/app/wz-home`, the browser shows "Page Not Found" or "Connection Refused".

- **Causes:**

1. **The Wazuh container is not running.**
2. **The VM IP address is incorrect or network is unreachable.**
3. **Port 5601 (Kibana) is blocked by firewall settings.**

- **Solutions:**

1. **Check Docker container status:**

```
docker ps -a
```

**Restart Wazuh and Kibana containers if stopped:**

```
make up
```

2. **Verify that you are using the correct VM IP.**
3. **Ensure port 5601 is open in firewall and security group settings (for cloud deployments).**

## 5.2 Issue 2: No Alerts are Showing on Dashboard

- **Symptoms:**  
The dashboard loads correctly, but no alerts or events are displayed even when attacks are simulated.
- **Causes:**
  1. Suricata is not generating logs.
  2. The log pipeline (Filebeat or API input) is misconfigured or down.
  3. Wazuh agent is not connected or not receiving logs.
- **Solutions:**
  1. Check Suricata log output in the container:  
`docker logs suricata`
  2. Ensure Suricata rules are properly loaded and log paths are correct.
  3. Restart log pipeline services and ensure connectivity.
  4. Re-run the attack simulation script on the attacker VM to generate logs.

## 5.3 Issue 3: Python Alert Parser Fails to Run

- **Symptoms:**  
Running `python3 main.py --export result.json` throws a `ModuleNotFoundError` or crashes.
- **Causes:**
  1. Required Python libraries are not installed.
  2. Configuration file `settings.yaml` is missing or malformed.
  3. API credentials in `settings.yaml` are incorrect.
- **Solutions:**
  1. Install required libraries:  
`pip3 install -r requirements.txt`
  2. Verify the existence and correctness of config/`settings.yaml`.
  3. Double-check Wazuh API credentials in the YAML file.

## 5.4 Issue 4: Exported CSV/JSON Files are Empty

- **Symptoms:**  
After running the parser, the exported CSV/JSON file has headers but no data rows.
- **Causes:**
  1. The filtering rules in alert\_rule.yaml are too strict and no log entries match.
  2. The time window for aggregation does not include any events.
  3. Logs are not correctly ingested into Wazuh/Elasticsearch.
- **Solutions:**
  1. Temporarily relax filtering conditions in alert\_rule.yaml to test matching.
  2. Increase the time window duration in aggregation block.
  3. Verify logs are ingested using Kibana Discover panel.

## 5.5 Issue 5: Scheduled Fetcher Stops Running Unexpectedly

- **Symptoms:**  
The Python scheduled fetcher starts but stops after a few cycles or crashes.
- **Causes:**
  1. Memory resource limits exceeded.
  2. Unhandled exception in data fetching process.
  3. Logs directory or output path does not exist.
- **Solutions:**
  1. Monitor system resource usage using htop or docker stats.
  2. Check fetcher logs in logs/wazuh\_fetcher.log for detailed error messages.
  3. Ensure /logs and /output directories exist:  
`mkdir -p logs output`

## 6. Additional Resources

### Official Documentation Links

[Documentation Link](#)

### GitHub Repository Link and QR code

Link – [Wazuh ELK SOC](#)

Link – [Log Alert Pipeline](#)

QR code



### Demo Video QR Code

[Link](#)

QR Code



### Contact Email:

[catherine.chen.canadait@gmail.com](mailto:catherine.chen.canadait@gmail.com)

[maggie.555x@gmail.com](mailto:maggie.555x@gmail.com)

[alexjiang1129@gmail.com](mailto:alexjiang1129@gmail.com)

## 7. References

- **Referenced APIs**
  - Wazuh API - Postman Collection  
[Postman](#)
  - Wazuh API Reference Documentation  
[API reference · Wazuh documentation](#)
- **Frameworks**
  - Docker  
[Docker Docs](#)
  - Python  
[Our Documentation | Python.org](#)
  - Elasticsearch  
[Getting Started with Elasticsearch | Elastic Videos](#)
  - Kibana  
[Kibana: Explore, Visualize, Discover Data | Elastic](#)
- **Open-source Resources**
  - Suricata EVE JSON Output Format Documentation  
[15.1.2. Eve JSON Format — Suricata 6.0.3 documentation](#)