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

SAIT

BlackMirrorSoc.

version 1.0

Table of Contents

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

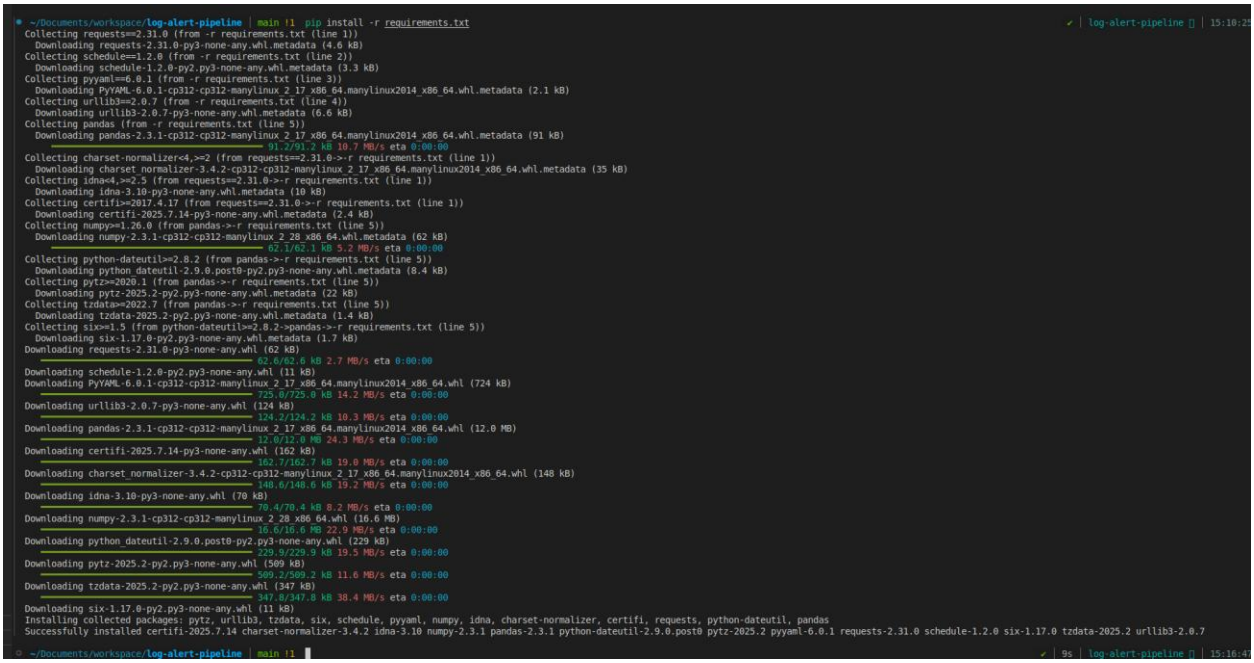
optional step, create a virtual environment

python3 -m venv .venv

source .venv/bin/activate

install dependencies

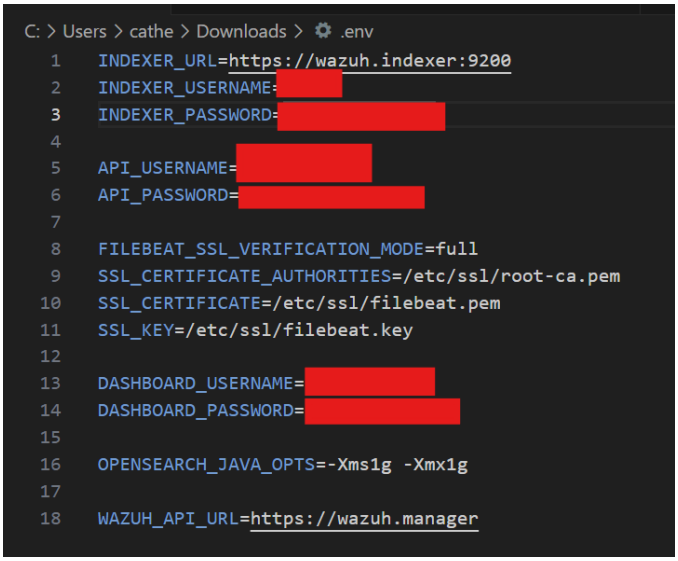
pip install -r requirements.txt



Step 3: Configure Settings Files

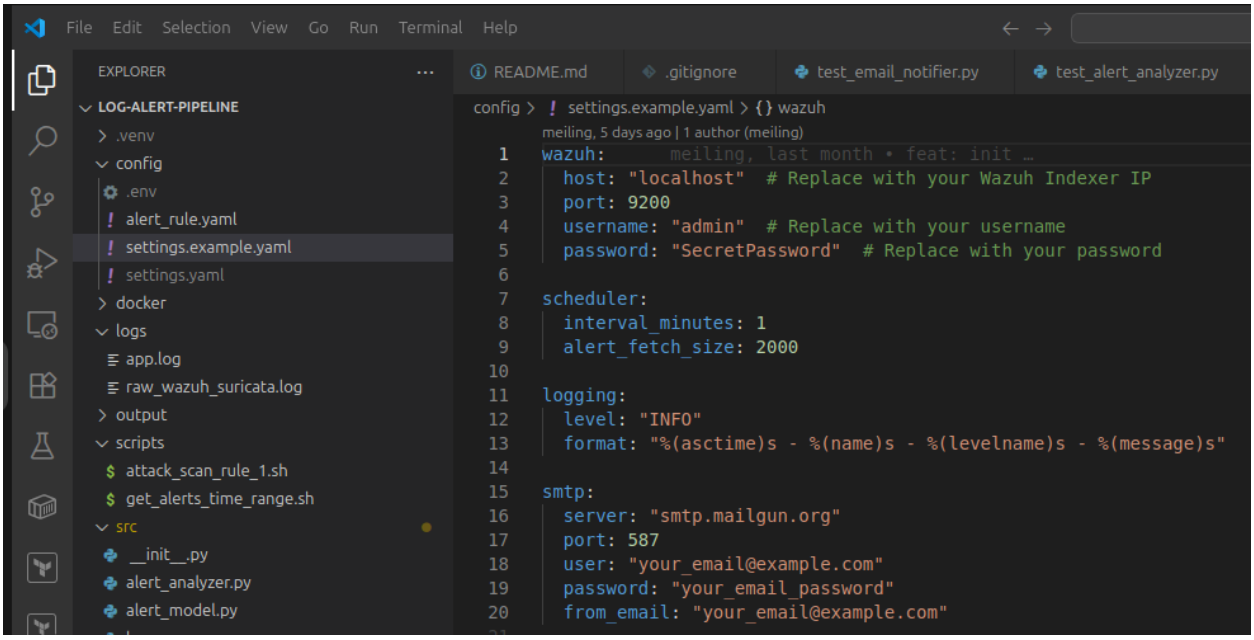
Repository 1: wazuh-suricata-elk-docker

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



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.



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

Step 4: Start Services / Applications

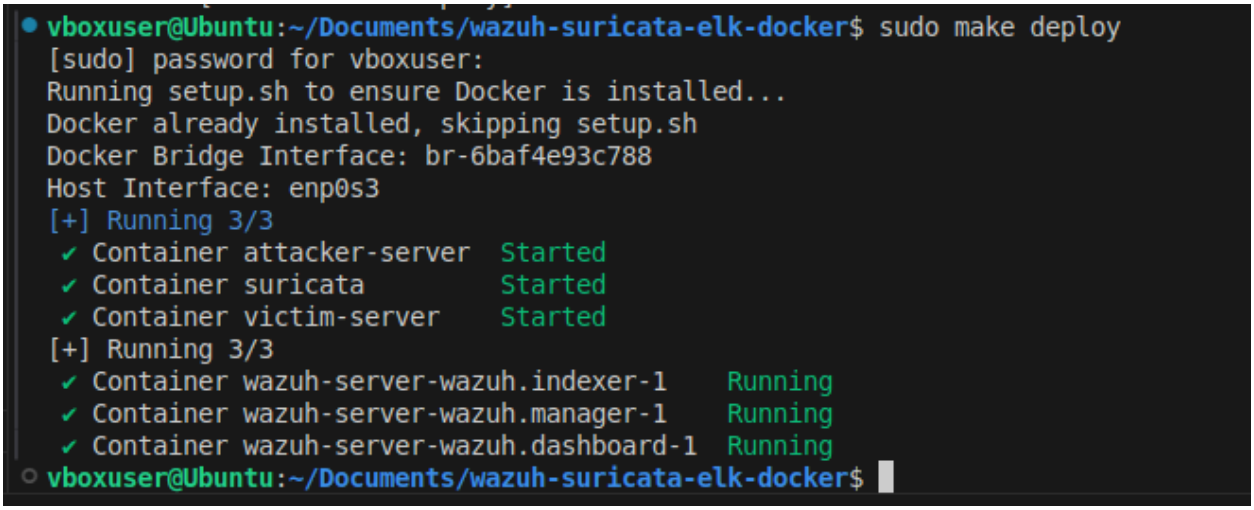
Repository 1: wazuh-suricata-elk-docker

One-Clich install

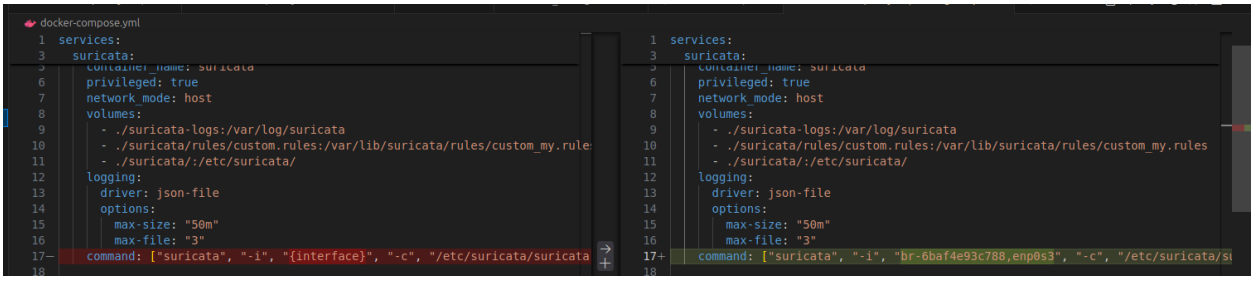
Step 1: Use “sudo make setup” to set up the environment

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

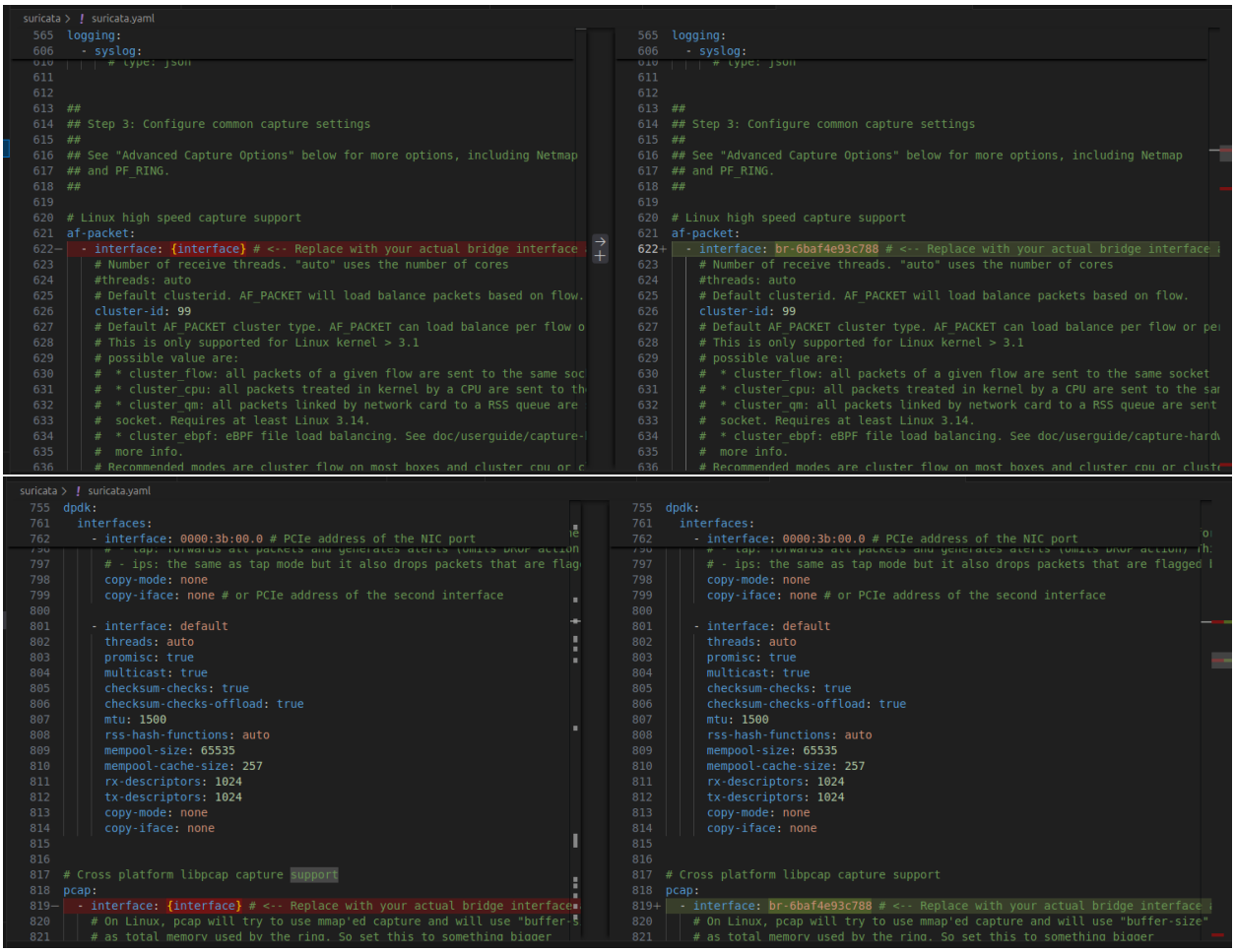
Step 3: Use “sudo make deploy” to deploy the service for the first time to run the service



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



docker-compose.yaml



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.

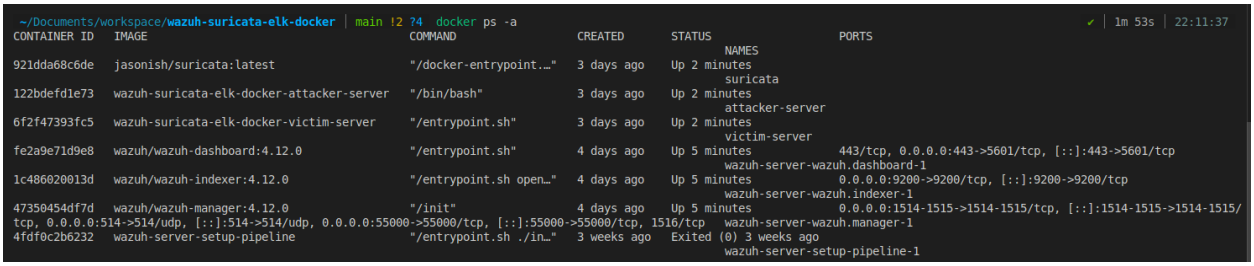
python main.py –help

3.2 Post-installation Verification

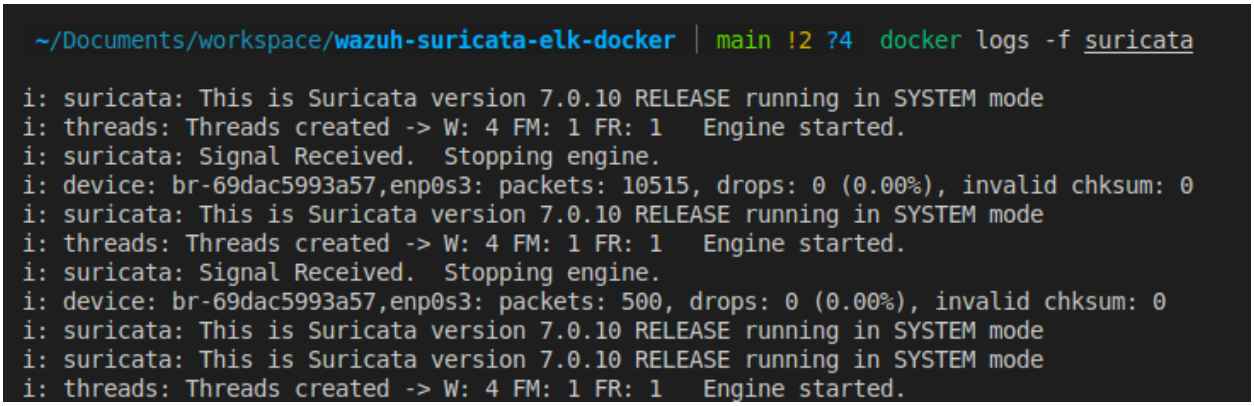
Check if the services are running correctly

Repository 1: wazuh-suricata-elk-docker

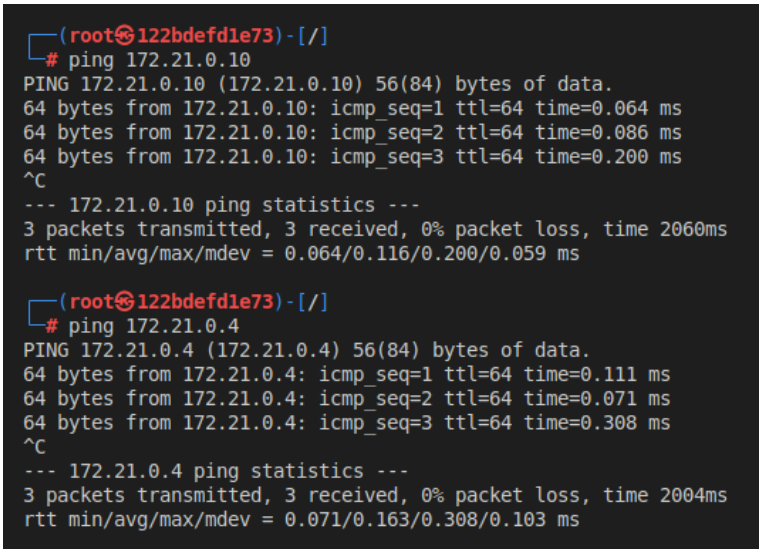
Use “docker ps -a” to check that all six services are running



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



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



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-sercer-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 "[redacted]:[redacted]" -k -X POST https://localhost:55000/security/user/auth
enticate

{"data": {"token": "eyJhbGciOiJFUzUxMiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ3YXp1aCIsImF1ZCI6IldhenVoIEFQSSBSRVNUIiwibmJmIjoxNzQ5NTMzMjEwLCJleHA
iOjE3NDk1MzQxMTAsInN1YiI6IndhenVoLXd1aSIsInJ1bl9hcyI6ZmFsc2UsInJiYWNfcm9sZXMiOlsxXSwicmJhY19tb2RlIjoid2hpdGUifQ.AcclhKAT3rzbksEssIyb-XT2
aF88vf_T5SirRZ-Ire4Ux6jMLzR9CFgXFhKe8J8kgrR4Km9FEcV5ZW6_J2fYKaWYAVdY3AjdQA_lqK85fvncfgJTtjDHS5AxQ3L9qrTGZOOFbIkHtZNB_M7JOyQKOqchQ0AYKZR1
bym87u4oBgHHMwC9"}, "error": 0}%
```

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

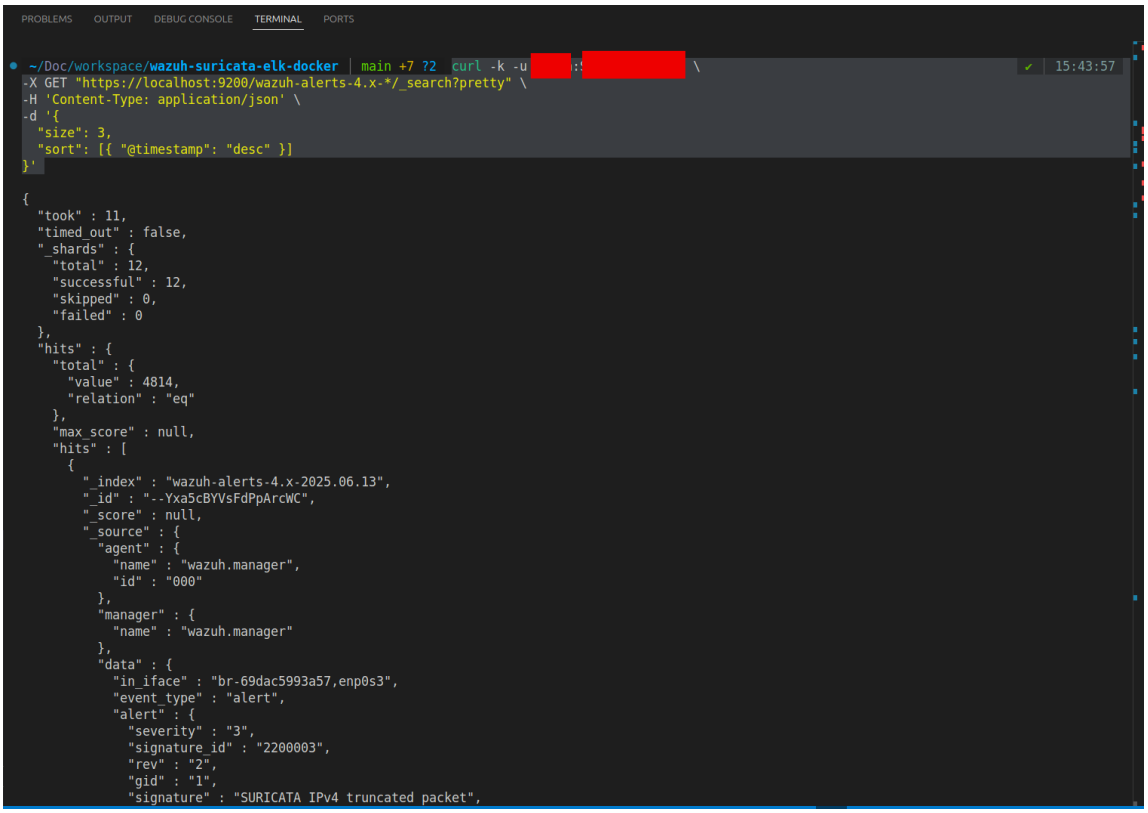
```
curl -k -X GET https://localhost:55000/agents -H "Authorization: Bearer eyJhbGciOiJFUzUxMiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ3YXp1aCIsImF1ZCI6IldhenVoIEFQSSBSRVNUIiwibmJmIjoxNzQ5NTMzMjEwLCJleHAiOjE3NDk1MzQxMTAsInN1YiI6IndhenVoLXd1aSIsInJ1bl9hcyI6ZmFsc2UsInJiYWNfcm9sZXMiOlsxXSwicmJhY19tb2RlIjoid2hpdGUifQ.AcclhKAT3rzbksEssIyb-XT2aF88vf_T5SirRZ-Ire4Ux6jMLzR9CFgXFhKe8J8kgrR4Km9FEcV5ZW6_J2fYKaWYAVdY3AjdQA_lqK85fvncfgJTtjDHS5AxQ3L9qrTGZOOFbIkHtZNB_M7JOyQKOqchQ0AYKZR1bym87u4oBgHHMwC9" -H "Content-Type: application/json"
```

```
~/Doc/w/wazuh-suricata-elk-docker | main ?2  curl -k -X GET https://localhost:55000/agents -H "Authorization: Bearer eyJhbGciOiJFUzUxMiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ3YXp1aCIsImF1ZCI6IldhenVoIEFQSSBSRVNUIiwibmJmIjoxNzQ5NTMzMjEwLCJleHAiOjE3NDk1MzQxMTAsInN1YiI6IndhenVoLXd1aSIsInJ1bl9hcyI6ZmFsc2UsInJiYWNfcm9sZXMiOlsxXSwicmJhY19tb2RlIjoid2hpdGUifQ.AcclhKAT3rzbksEssIyb-XT2aF88vf_T5SirRZ-Ire4Ux6jMLzR9CFgXFhKe8J8kgrR4Km9FEcV5ZW6_J2fYKaWYAVdY3AjdQA_lqK85fvncfgJTtjDHS5AxQ3L9qrTGZOOFbIkHtZNB_M7JOyQKOqchQ0AYKZR1bym87u4oBgHHMwC9" -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": "2a82e879cfe2", "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": "9da47ecf9165", "ip": "any", "status_code": 0, "id": "002"}, {"os": {"arch": "x86_64", "codename": "Jammy Jellyfish", "major": "22", "minor": "04", "name": "Ubuntu", "platform": "ubuntu", "uname": "Linux |f3ad33bb7db8 |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", "disconnection_time": "2025-06-10T05:17:18+00:00", "configSum": "ab73af41699f13fdd81903b5f23d8d00", "mergedSum": "4a8724b20dee0124ff9656783c490c4e", "group": ["default"], "manager": "wazuh.manager", "lastKeepAlive": "2025-06-10T05:10:28+00:00", "registerIP": "any", "group_config_status": "synced", "status": "disconnected", "dateAdd": "2025-06-10T04:43:18+00:00", "name": "f3ad33bb7db8", "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 |3e5aaf5bf8db |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": "ab73af41699f13fdd81903b5f23d8d00", "mergedSum": "4a8724b20dee0124ff9656783c490c4e", "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": "3e5aaf5bf8db", "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.



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.

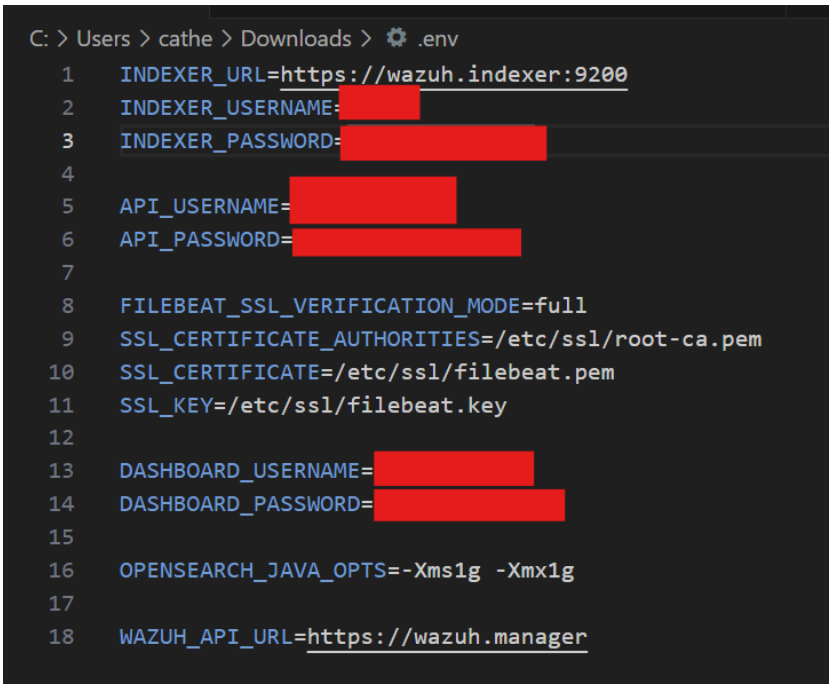
4. User Guide

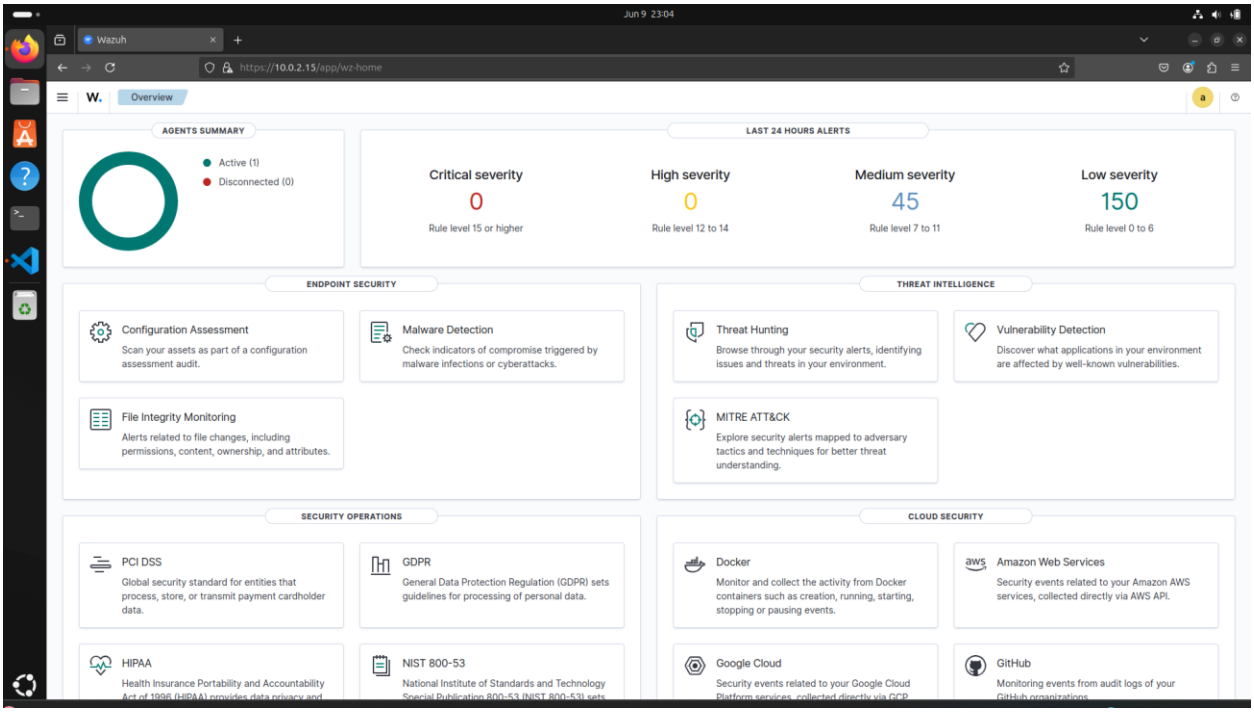
4.1 Login and Main Interface Overview

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**.





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)

- **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.

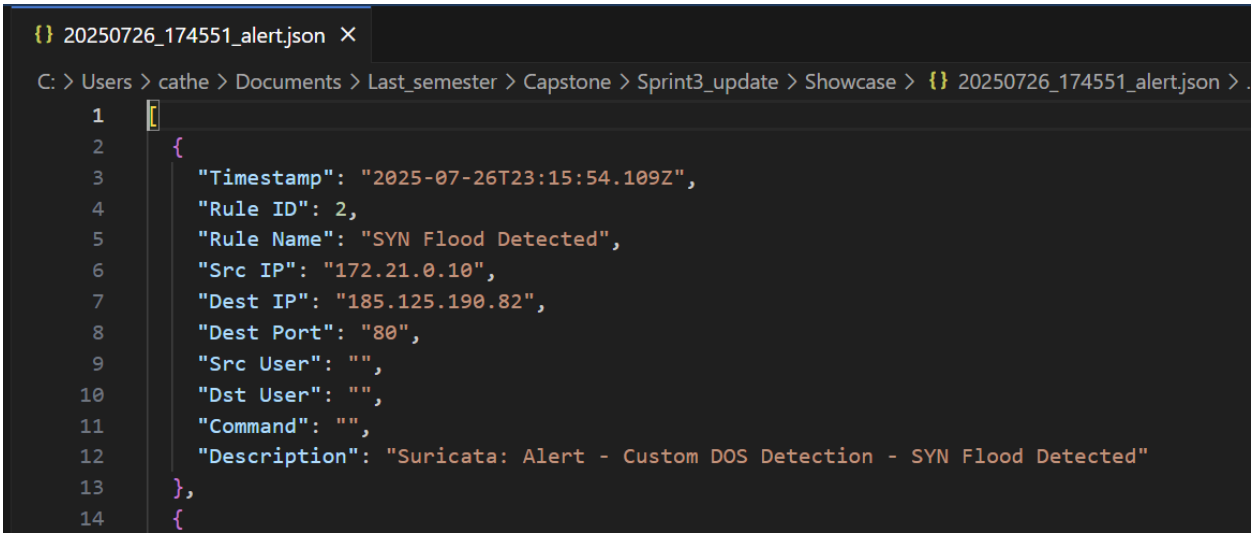
Log-Alert-Pipeline (/output folder)

- **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	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		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



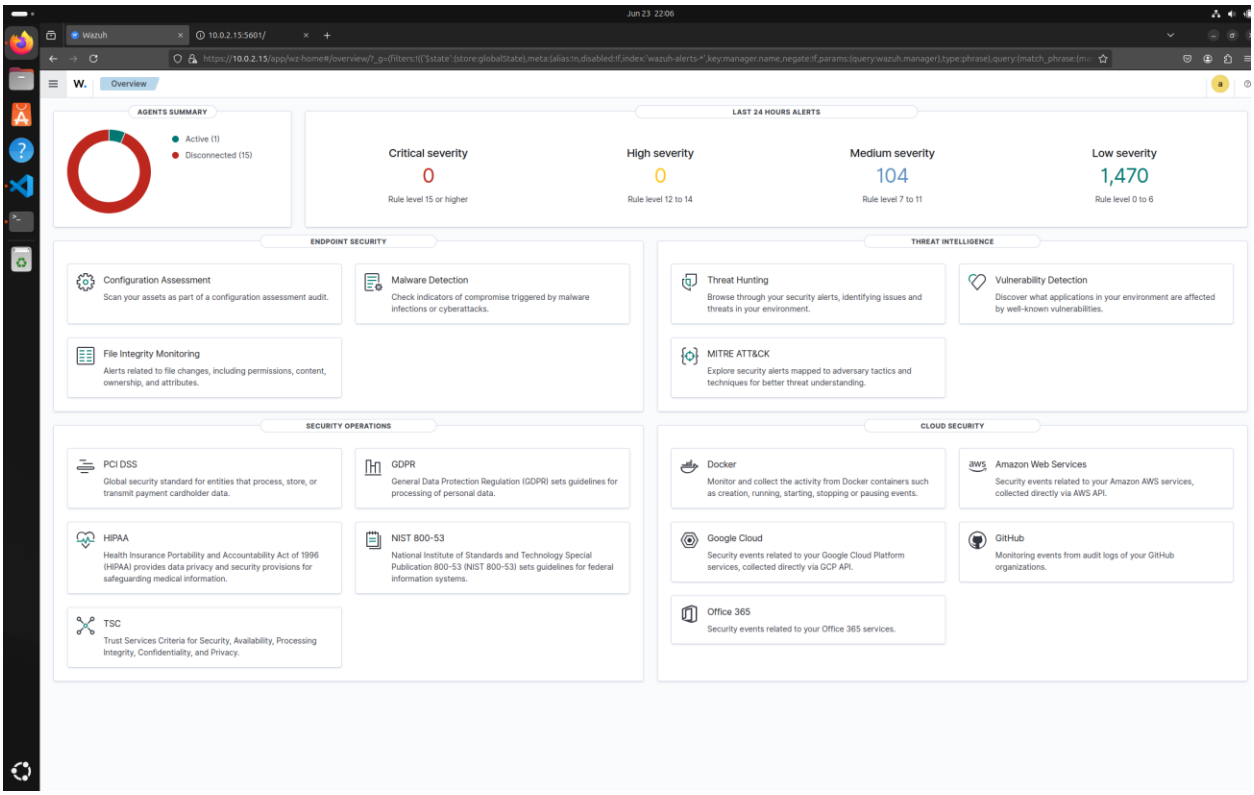
Example for JSON

4.2 Core Function Operations

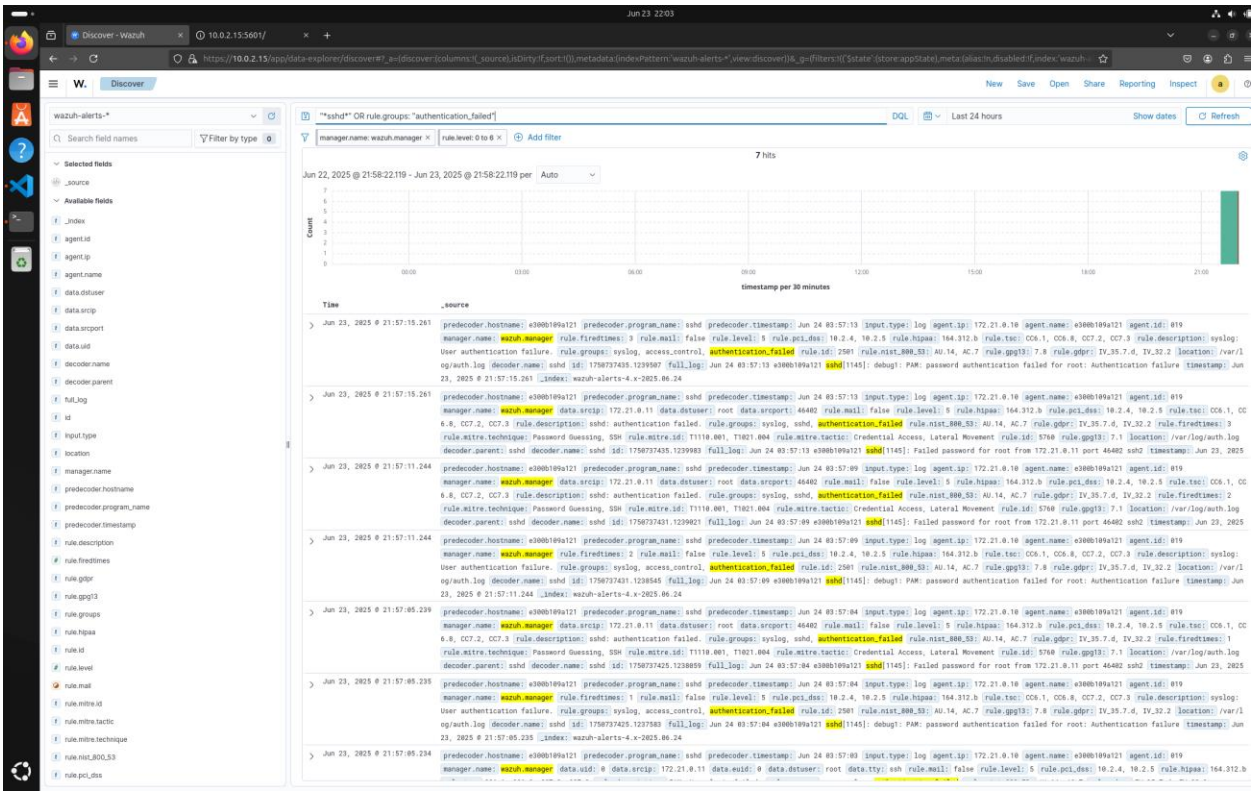
Repository 1: wazuh-suricata-elk-docker - Function 1: Wazuh detect - log-in failed

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

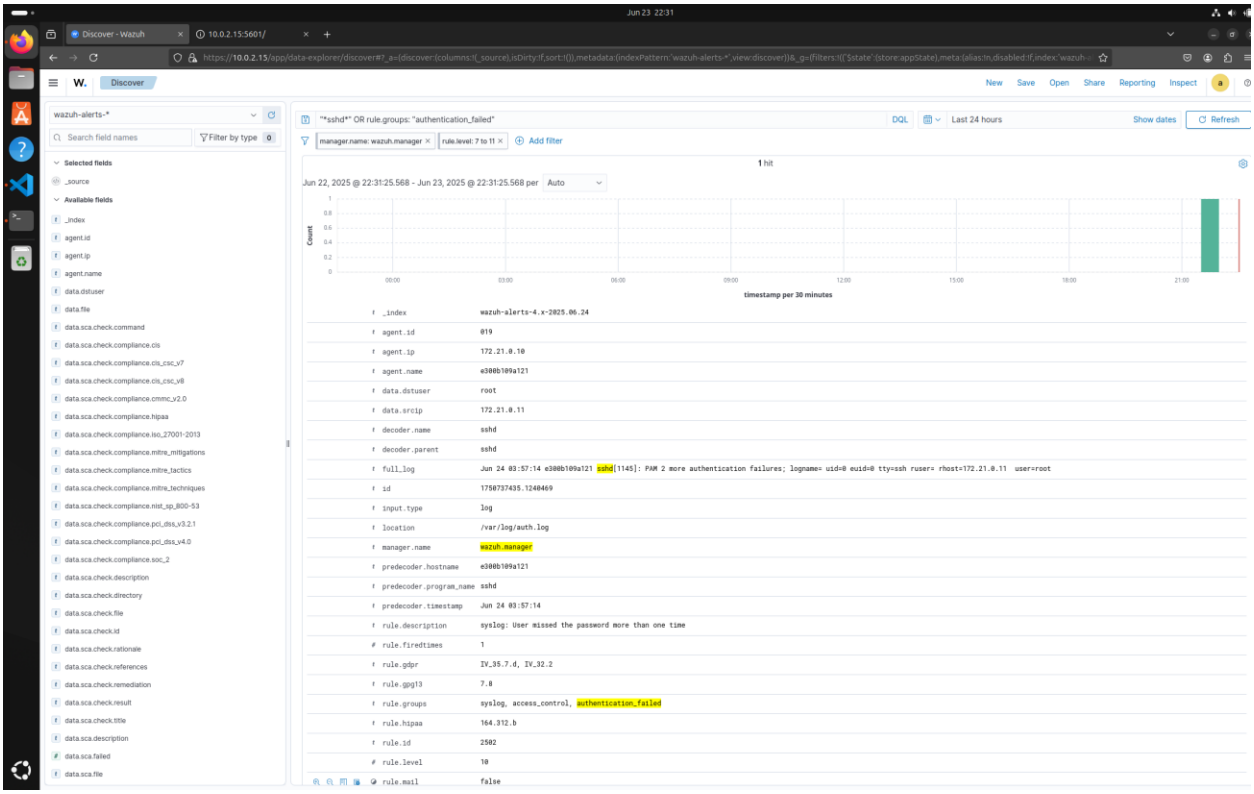
```
~/Doc/w/wazuh-suricata-elk-docker | main !4 ?3  docker exec -it attacker-ser
ver bash
┌──(root㉿e797eb9bd20d)-[/]
└─# ssh root@172.21.0.10
The authenticity of host '172.21.0.10 (172.21.0.10)' can't be established.
ED25519 key fingerprint is SHA256:eaJWNglV65IbcL4T5tb9RRcS4KZazwjpLWFcBojNHJU
.
This 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)-[/]
└─#
```



Authentication failed is shown



Can even catch the attacker's IP address

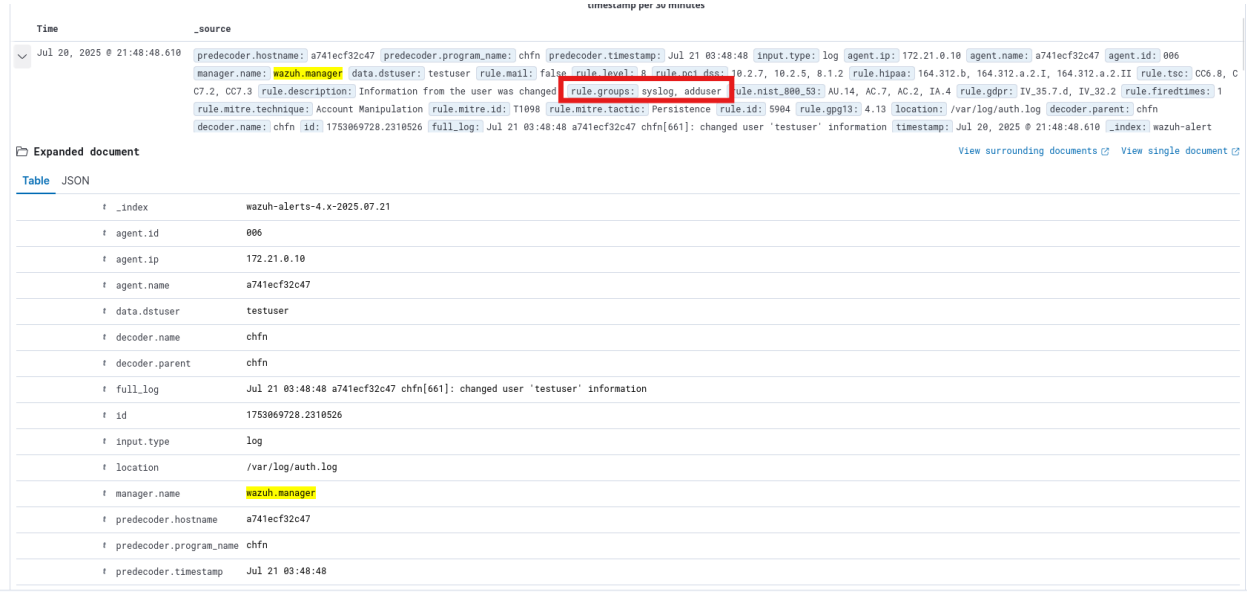


Repository 1: wazuh-suricata-elk-docker - Function 2: Wazuh detect - create a user

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

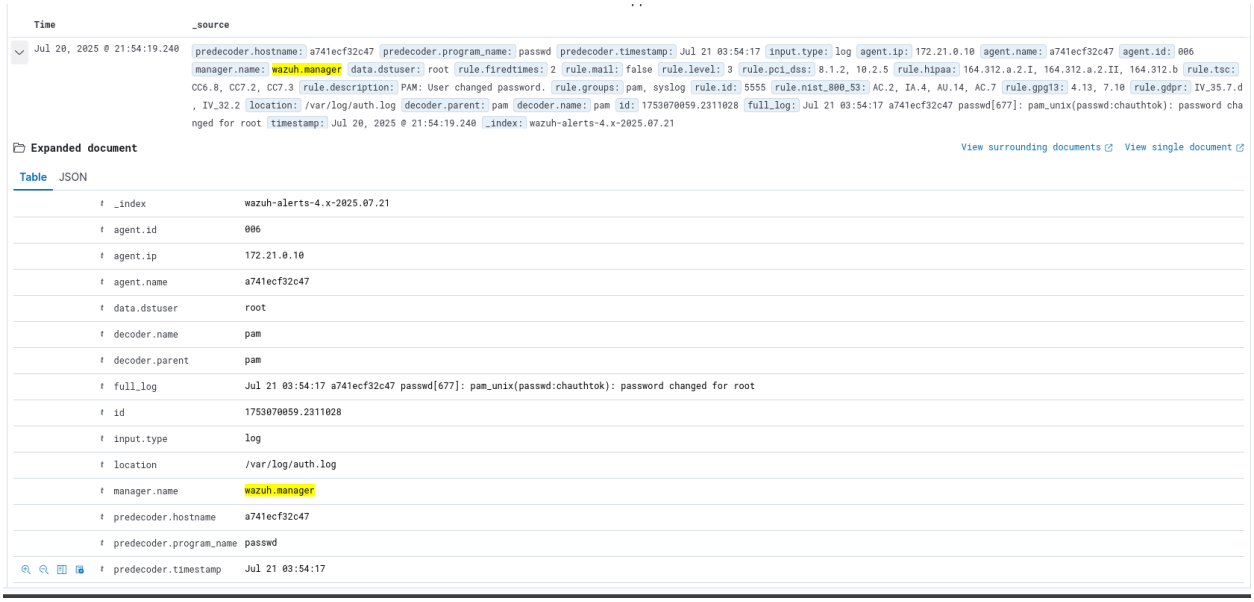


Repository 1: wazuh-suricata-elk-docker - Function 3: Wazuh detect - user changes the password

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

```
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 @ 21:54:19.240	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: wazuh.manager data.dstuser: root rule.firedtimes: 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: CC6.8, CC7.2, CC7.3 rule.description: PAM: User changed password. rule.groups: pam, syslog rule.id: 5555 rule.nist_800_53: AC.2, IA.4, AU.14, AC.7 rule.gpg13: 4.13, 7.10 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 @ 21:54:19.240 _index: wazuh-alerts-4.x-2025.07.21

Expanded document — View surrounding documents — View single document

Table | JSON

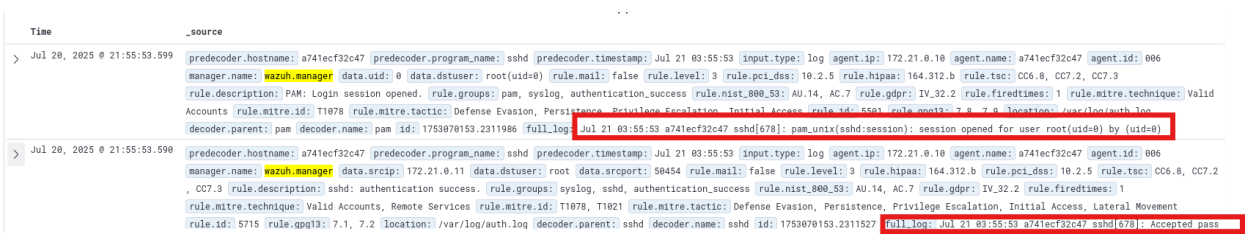
_index	wazuh-alerts-4.x-2025.07.21
agent.id	006
agent.ip	172.21.0.10
agent.name	a741ecf32c47
data.dstuser	root
decoder.name	pam
decoder.parent	pam
full_log	Jul 21 03:54:17 a741ecf32c47 passwd[677]: pam_unix(passwd:chauthtok): password changed for root
id	1753070059.2311028
input.type	log
location	/var/log/auth.log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

Repository 1: wazuh-suricata-elk-docker - Function 4: Wazuh detect - remote access alert 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:/#
```

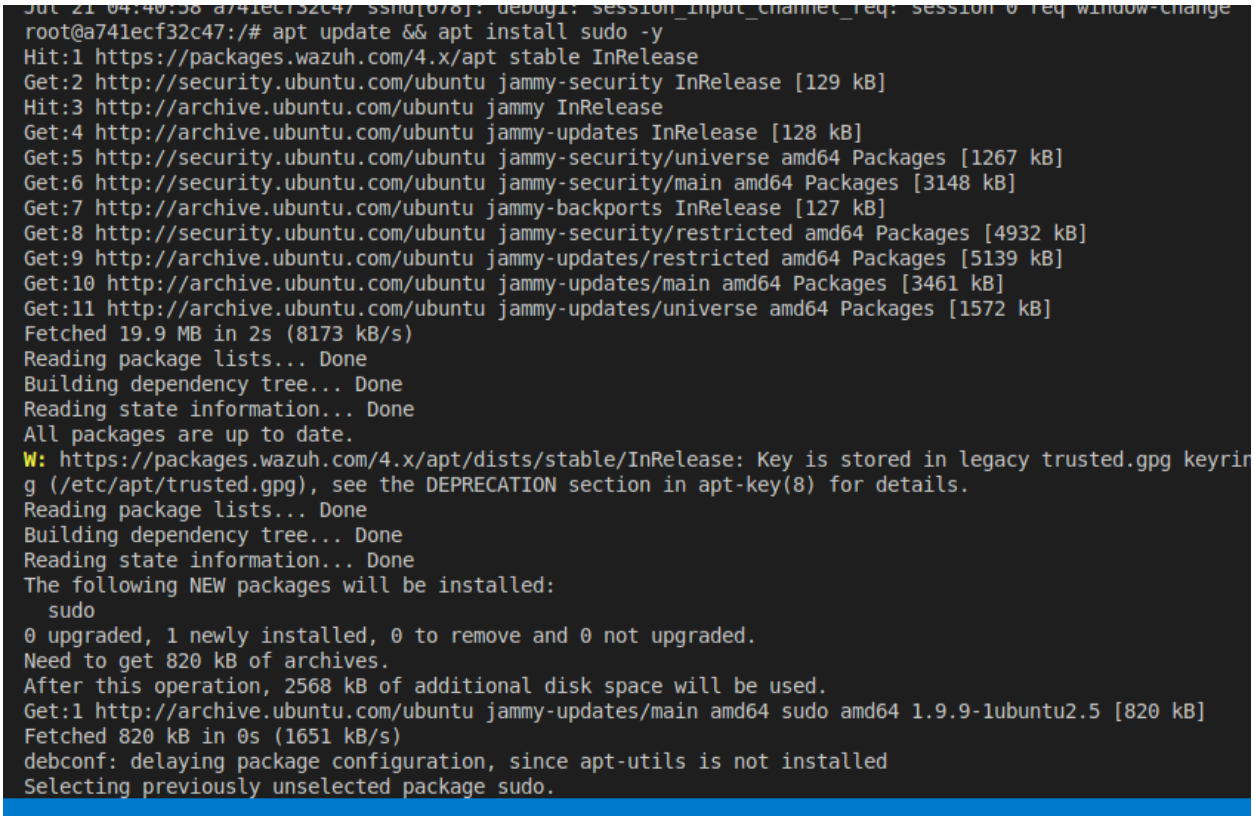
Remote login to the victim-server by using the root account successfully. It will show the message, which is “session opened for user root” and “sshd: Accepted password” on the dashboard.



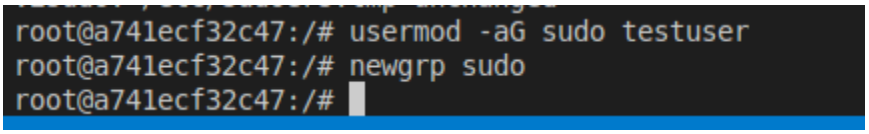
Time	_source
Jul 20, 2025 @ 21:55:53.599	predecoder.hostname: a741ecf32c47 predecoder.program_name: sshd predecoder.timestamp: Jul 21 03:55:53 input.type: log agent.ip: 172.21.0.10 agent.name: a741ecf32c47 agent.id: 006 manager.name: wazuh.manager data.uid: 0 data.dstuser: root(uid=0) rule.mail: false rule.level: 3 rule.pci_dss: 10.2.5 rule.hipaa: 164.312.b rule.tsc: CC6.8, CC7.2, CC7.3 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.firedtimes: 1 rule.mitre.technique: Valid Accounts rule.mitre.id: T1078 rule.mitre.tactic: Defense Evasion, Persistence, Privilege Escalation, Initial Access rule.id: 5501 rule.gpg13: 7.8, 7.9 location: /var/log/auth.log 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)
Jul 20, 2025 @ 21:55:53.590	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: wazuh.manager data.srcip: 172.21.0.11 data.dstuser: root data.srcport: 50454 rule.mail: false rule.level: 3 rule.hipaa: 164.312.b rule.pci_dss: 10.2.5 rule.tsc: CC6.8, CC7.2, CC7.3 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.firedtimes: 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.gpg13: 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

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”.

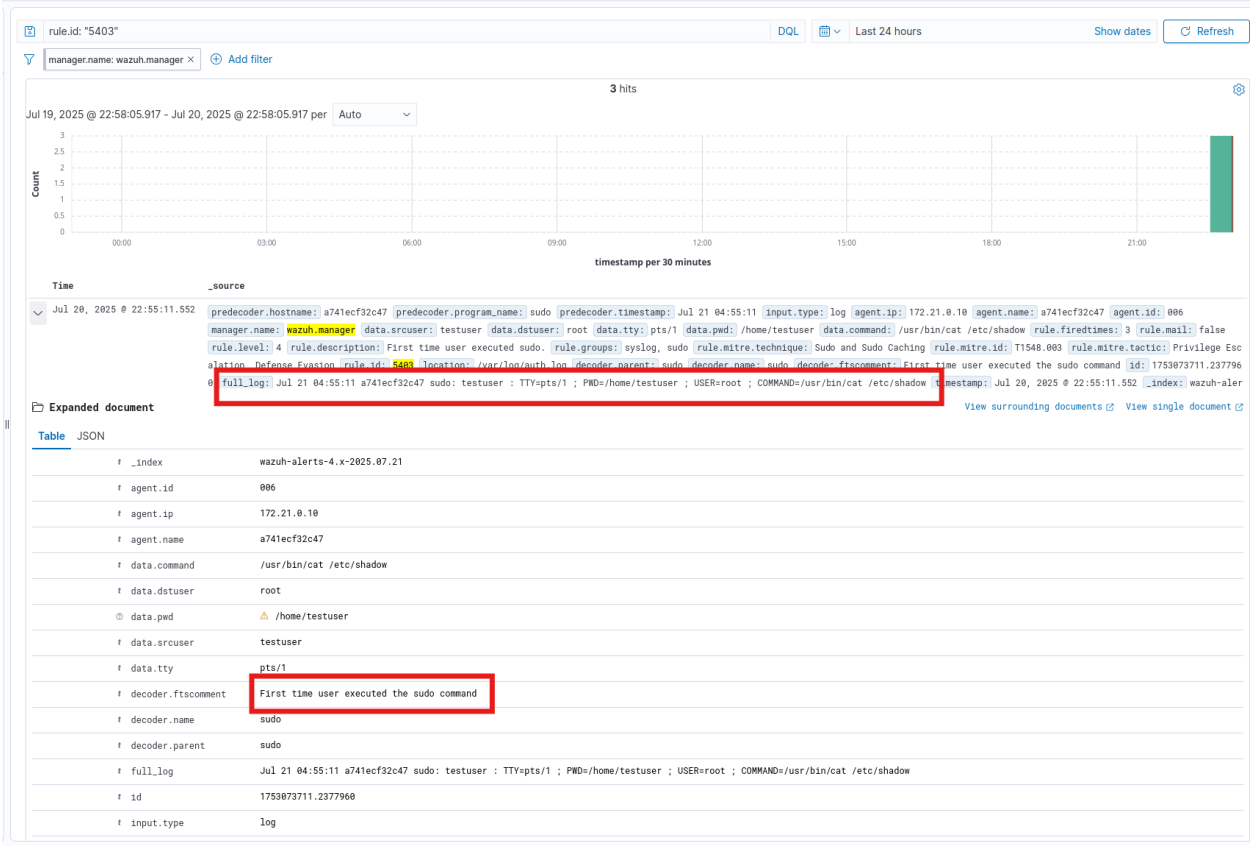


Add the testuser to the sudo group



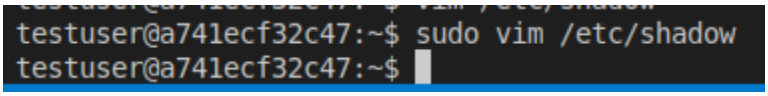
Log in to the attacker server using the testuser account and print the /etc/shadow file; the alert will be displayed on the dashboard.

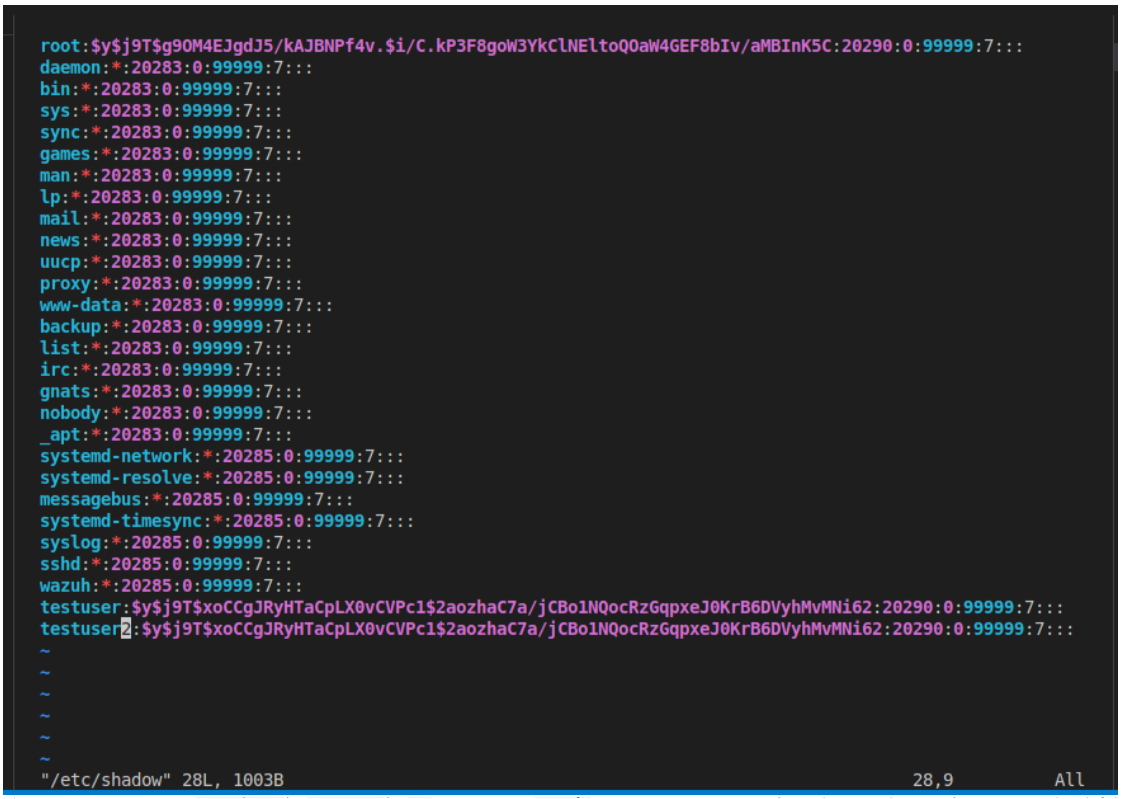
```
┌──(root㉿9074216d9feb)-[/]
└─# ssh testuser@172.21.0.10
testuser@172.21.0.10's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.11.0-29-generic x86_64)

 * 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:$y$j9T$g9OM4EJgdJ5/kAJBNPf4v.$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:::
```

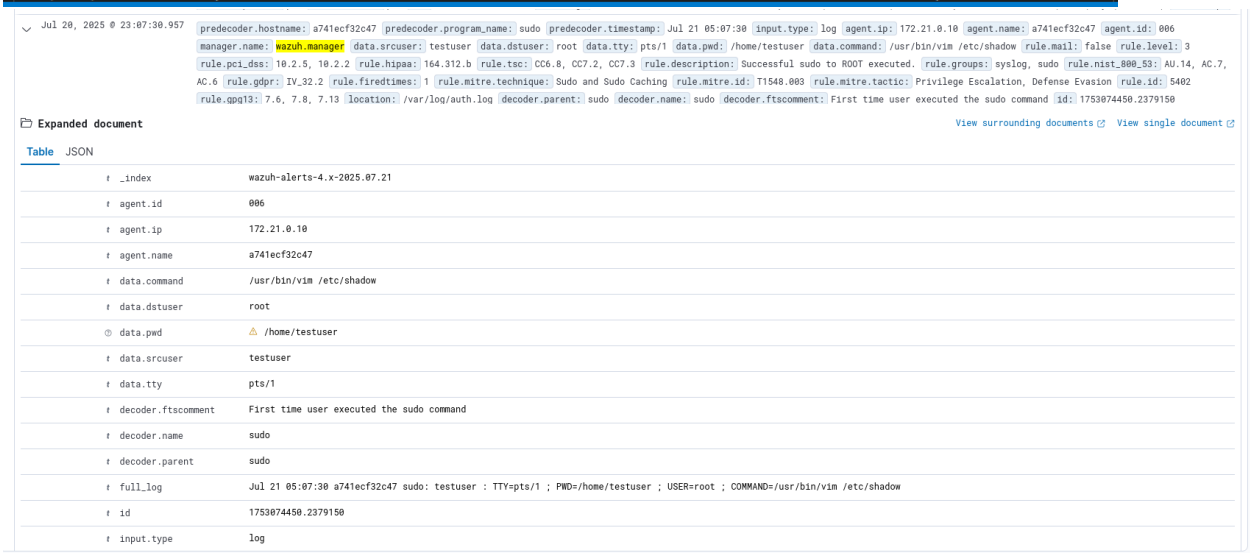


Repository 1: wazuh-suricata-elk-docker - Function 6: Wazuh detect - System Change: Modification or Deletion of Critical Files

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

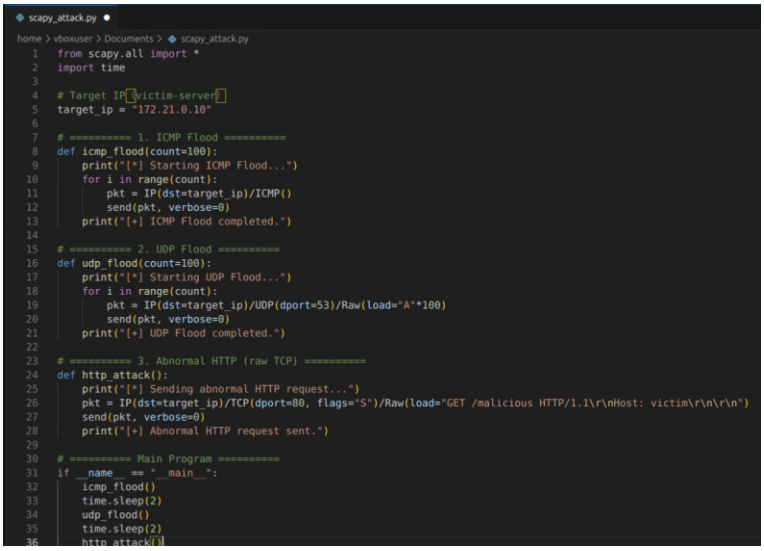






Repository 1: wazuh-suricata-elk-docker - Function 7: Suricata detect - simulate ICMP/UDP/HTTP attacks with Scapy

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.

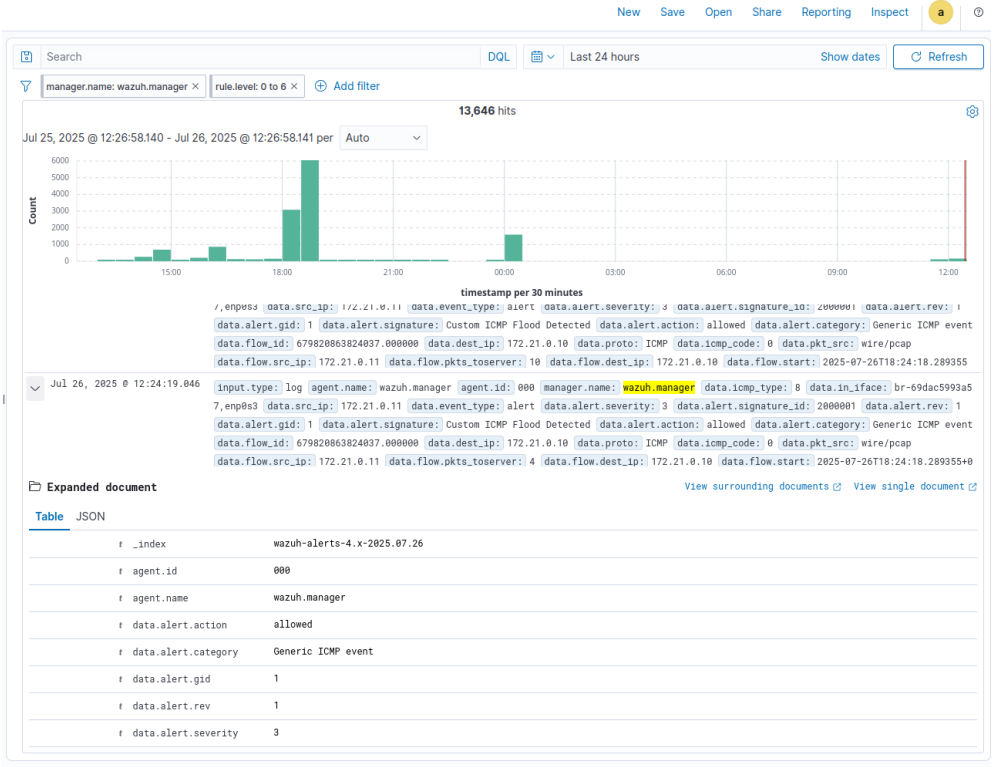


Next, we will use a Python script in the container to simulate ICMP Flood, UDP Flood, and abnormal HTTP request attacks to see if they can be detected by Wazuh.

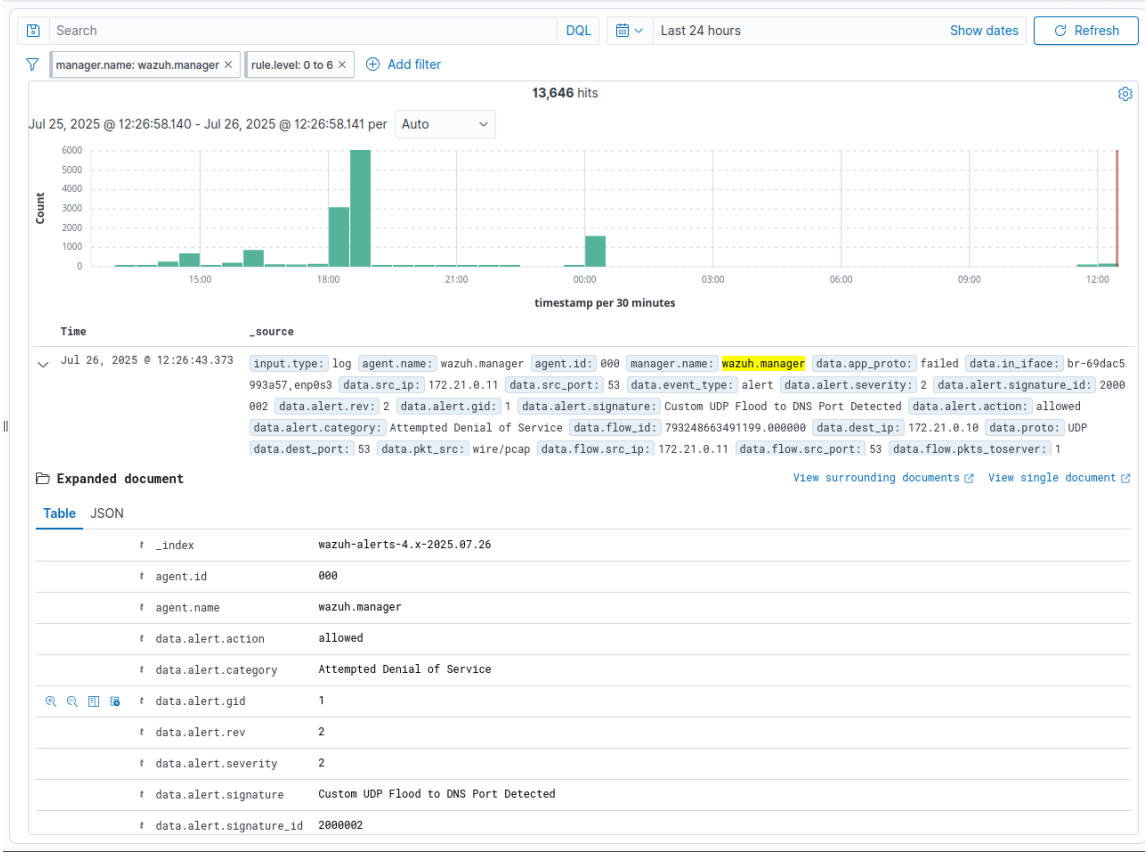
The picture below shows a screenshot of executing the attack script.

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

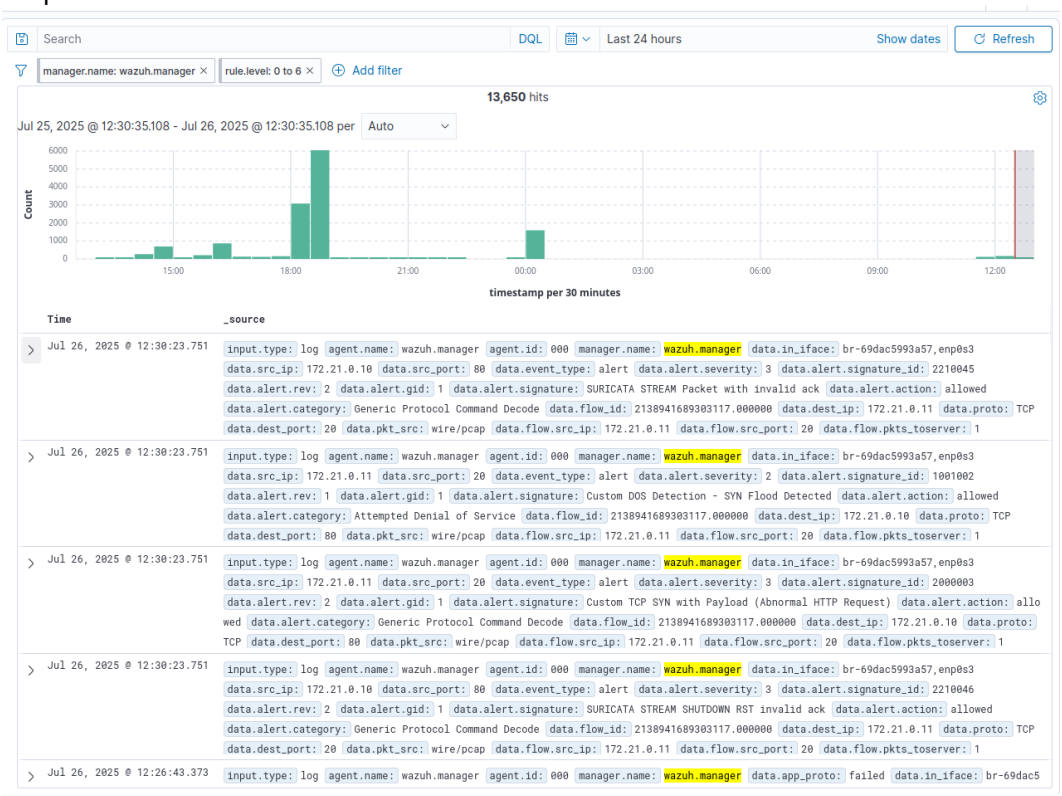
ICMP Flood

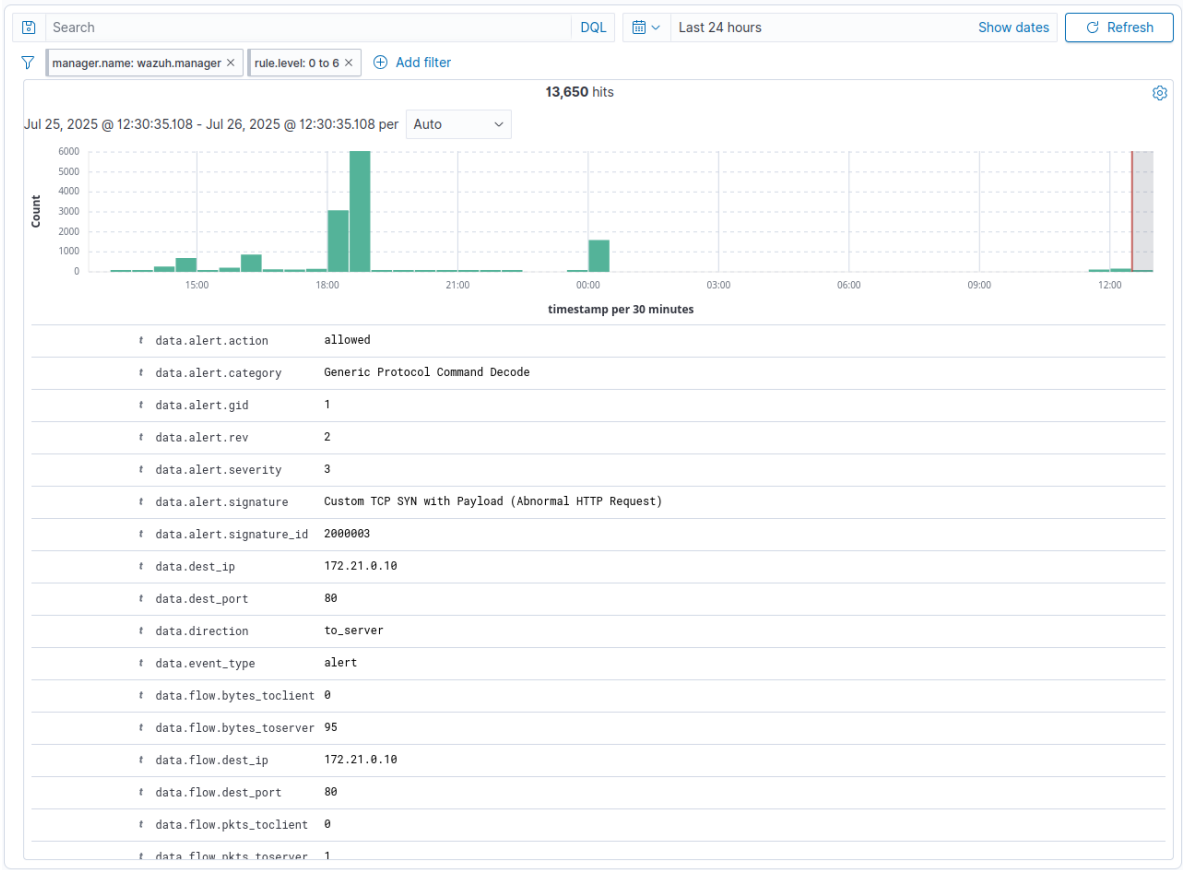


UDP Flood



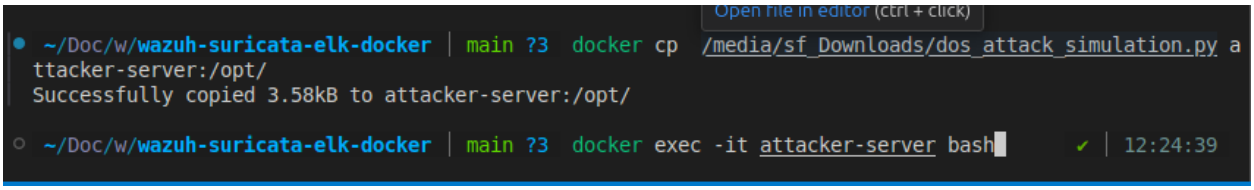
Http attack

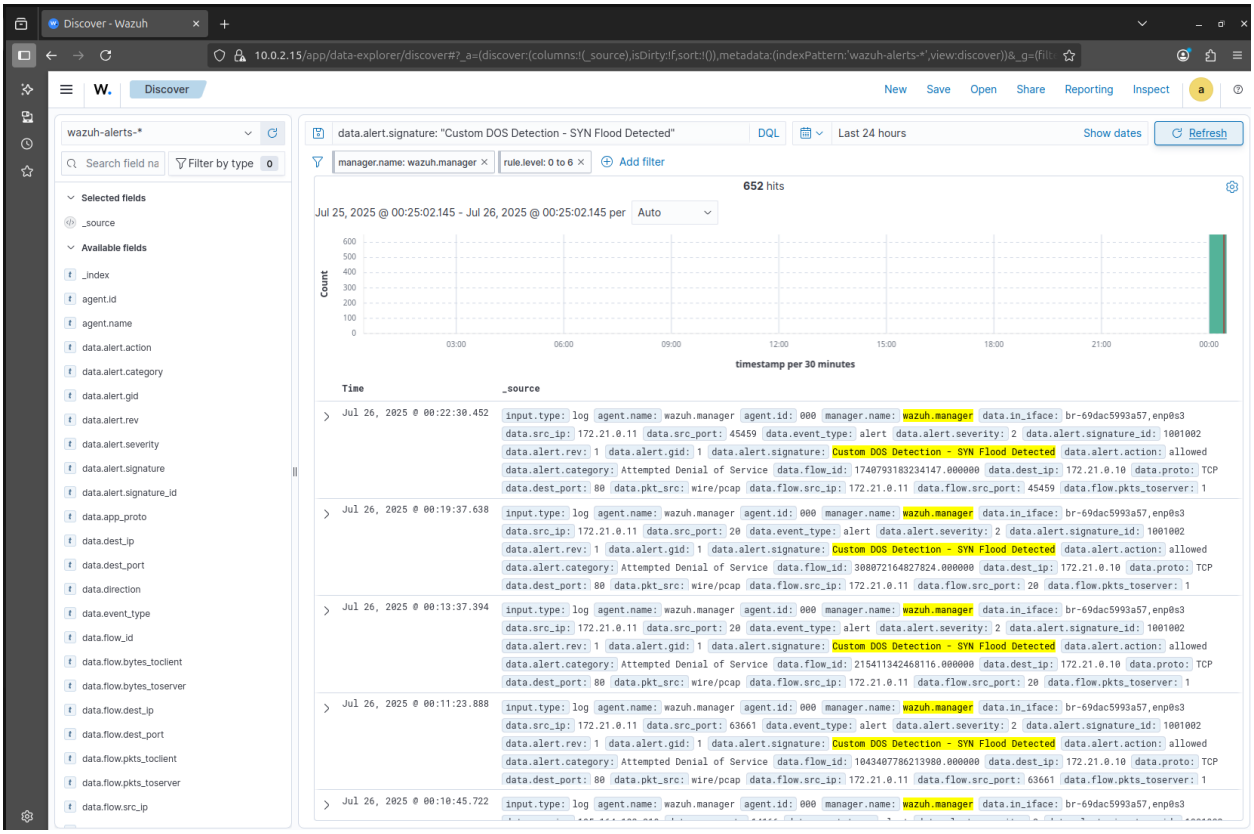

Search
DQL
Last 24 hours
Show dates
Refresh
manager.name: wazuh.manager
rule.level: 0 to 6
Add filter
13,650 hits
Jul 25, 2025 @ 12:30:35.108 - Jul 26, 2025 @ 12:30:35.108 per Auto
Count
timestamp per 30 minutes
Time
_source
Jul 26, 2025 @ 12:30:23.751
input.type: log agent.name: wazuh.manager agent.id: 000 manager.name: wazuh.manager data.in_iface: br-69dac5993a57,enp0s3 data.src_ip: 172.21.0.10 data.src_port: 80 data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 2210045 data.alert.rev: 2 data.alert.gid: 1 data.alert.signature: SURICATA STREAM Packet with invalid ack data.alert.action: allowed data.alert.category: Generic Protocol Command Decode data.flow_id: 2138941689303117.000000 data.dest_ip: 172.21.0.11 data.proto: TCP data.dest_port: 20 data.pkt_src: wire/pcap data.flow.src_ip: 172.21.0.11 data.flow.src_port: 20 data.flow.pkts_toserver: 1
Jul 26, 2025 @ 12:30:23.751
input.type: log agent.name: wazuh.manager agent.id: 000 manager.name: wazuh.manager data.in_iface: br-69dac5993a57,enp0s3 data.src_ip: 172.21.0.11 data.src_port: 20 data.event_type: alert data.alert.severity: 2 data.alert.signature_id: 1001002 data.alert.rev: 1 data.alert.gid: 1 data.alert.signature: Custom DOS Detection - SYN Flood Detected data.alert.action: allowed data.alert.category: Attempted Denial of Service data.flow_id: 2138941689303117.000000 data.dest_ip: 172.21.0.10 data.proto: TCP data.dest_port: 80 data.pkt_src: wire/pcap data.flow.src_ip: 172.21.0.11 data.flow.src_port: 20 data.flow.pkts_toserver: 1
Jul 26, 2025 @ 12:30:23.751
input.type: log agent.name: wazuh.manager agent.id: 000 manager.name: wazuh.manager data.in_iface: br-69dac5993a57,enp0s3 data.src_ip: 172.21.0.11 data.src_port: 20 data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 2000003 data.alert.rev: 2 data.alert.gid: 1 data.alert.signature: Custom TCP SYN with Payload (Abnormal HTTP Request) data.alert.action: allowed data.alert.category: Generic Protocol Command Decode data.flow_id: 2138941689303117.000000 data.dest_ip: 172.21.0.10 data.proto: TCP data.dest_port: 80 data.pkt_src: wire/pcap data.flow.src_ip: 172.21.0.11 data.flow.src_port: 20 data.flow.pkts_toserver: 1
Jul 26, 2025 @ 12:30:23.751
input.type: log agent.name: wazuh.manager agent.id: 000 manager.name: wazuh.manager data.in_iface: br-69dac5993a57,enp0s3 data.src_ip: 172.21.0.10 data.src_port: 80 data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 2210046 data.alert.rev: 2 data.alert.gid: 1 data.alert.signature: SURICATA STREAM SHUTDOWN RST invalid ack data.alert.action: allowed data.alert.category: Generic Protocol Command Decode data.flow_id: 2138941689303117.000000 data.dest_ip: 172.21.0.11 data.proto: TCP data.dest_port: 20 data.pkt_src: wire/pcap data.flow.src_ip: 172.21.0.11 data.flow.src_port: 20 data.flow.pkts_toserver: 1
Jul 26, 2025 @ 12:26:43.373
input.type: log agent.name: wazuh.manager agent.id: 000 manager.name: wazuh.manager data.app_proto: failed data.in_iface: br-69dac5


Search
DQL
Last 24 hours
Show dates
Refresh
manager.name: wazuh.manager
rule.level: 0 to 6
Add filter
13,650 hits
Jul 25, 2025 @ 12:30:35.108 - Jul 26, 2025 @ 12:30:35.108 per Auto
Count
timestamp per 30 minutes
data.alert.action allowed
data.alert.category Generic Protocol Command Decode
data.alert.gid 1
data.alert.rev 2
data.alert.severity 3
data.alert.signature Custom TCP SYN with Payload (Abnormal HTTP Request)
data.alert.signature_id 2000003
data.dest_ip 172.21.0.10
data.dest_port 80
data.direction to_server
data.event_type alert
data.flow.bytes_toclient 0
data.flow.bytes_toserver 95
data.flow.dest_ip 172.21.0.10
data.flow.dest_port 80
data.flow.pkts_toclient 0
data.flow.pkts_toserver 1

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

Upload the script to the attacker-server

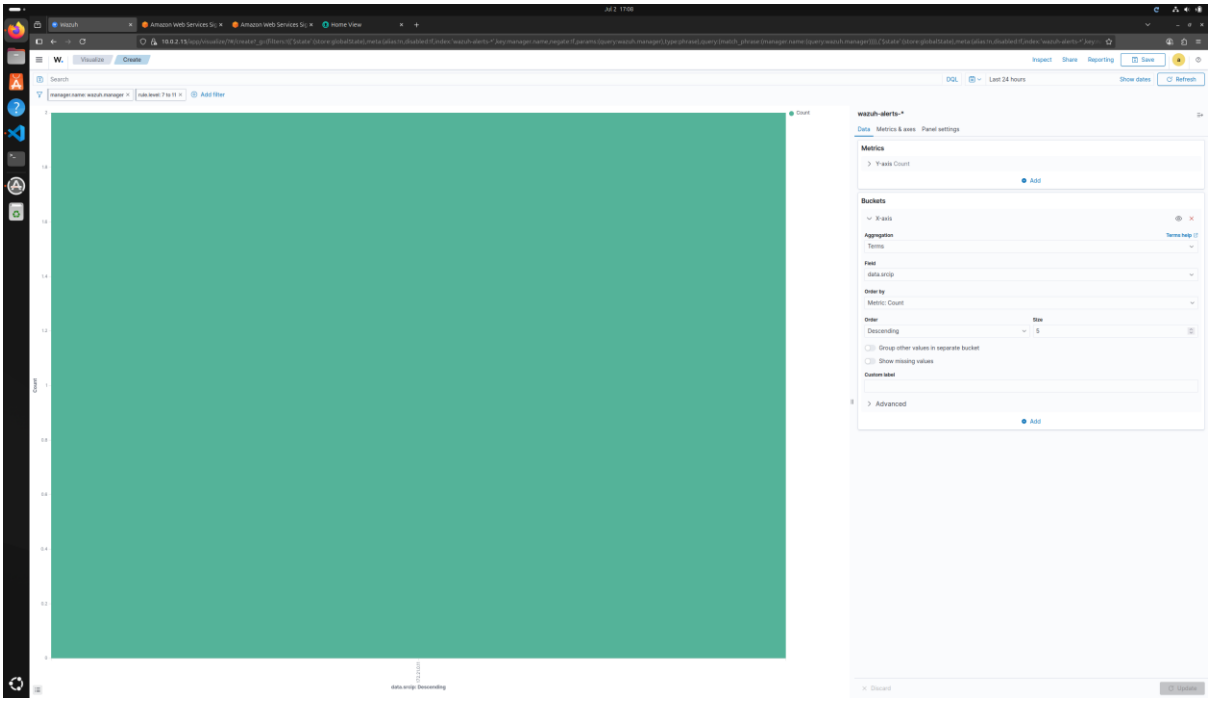




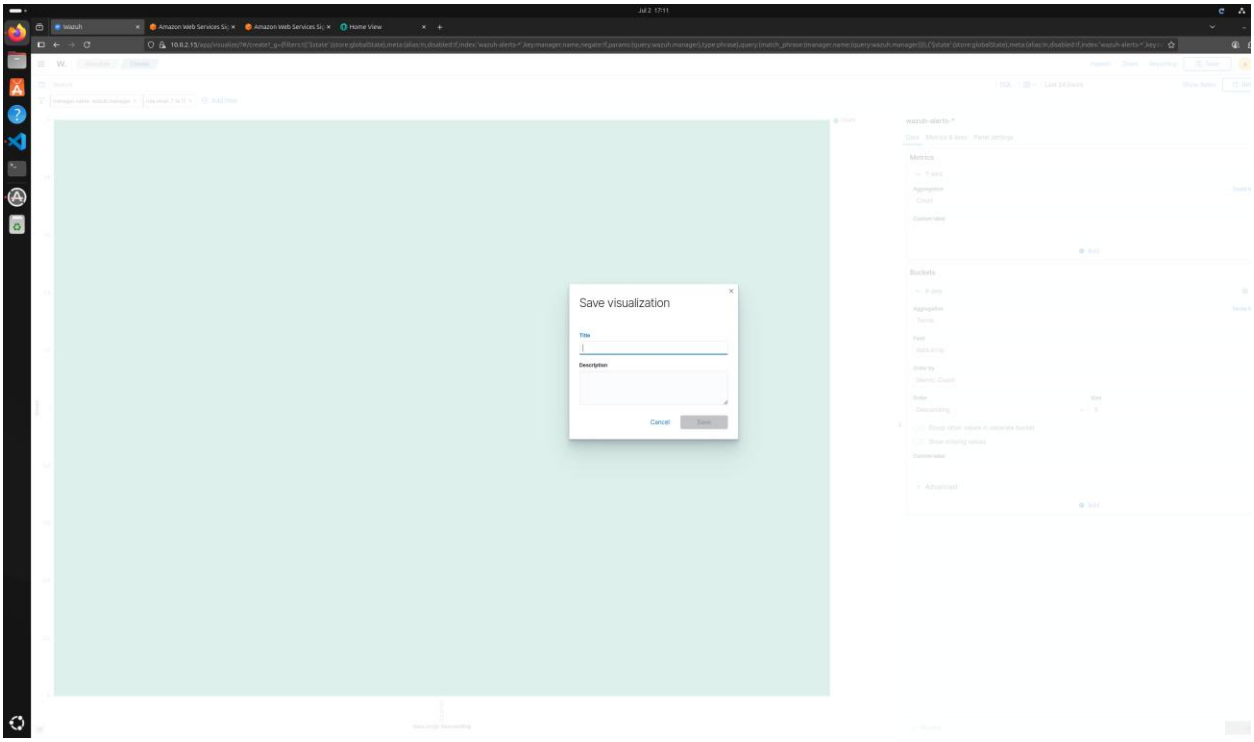
Repository 1: wazuh-suricata-elk-docker - Function 9: How to use the Kibana dashboard

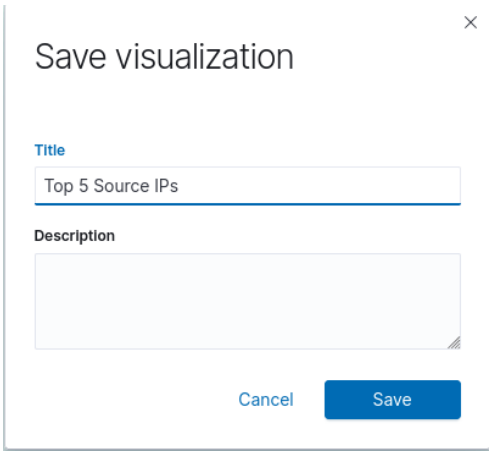
Add Top 5 IP bar chart to Kibana dashboard

Change the size of the order from 20 to 5

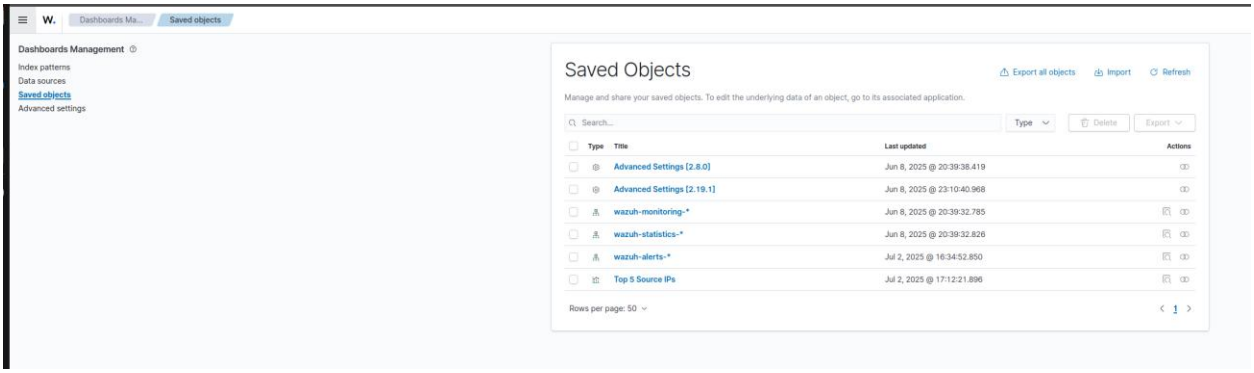


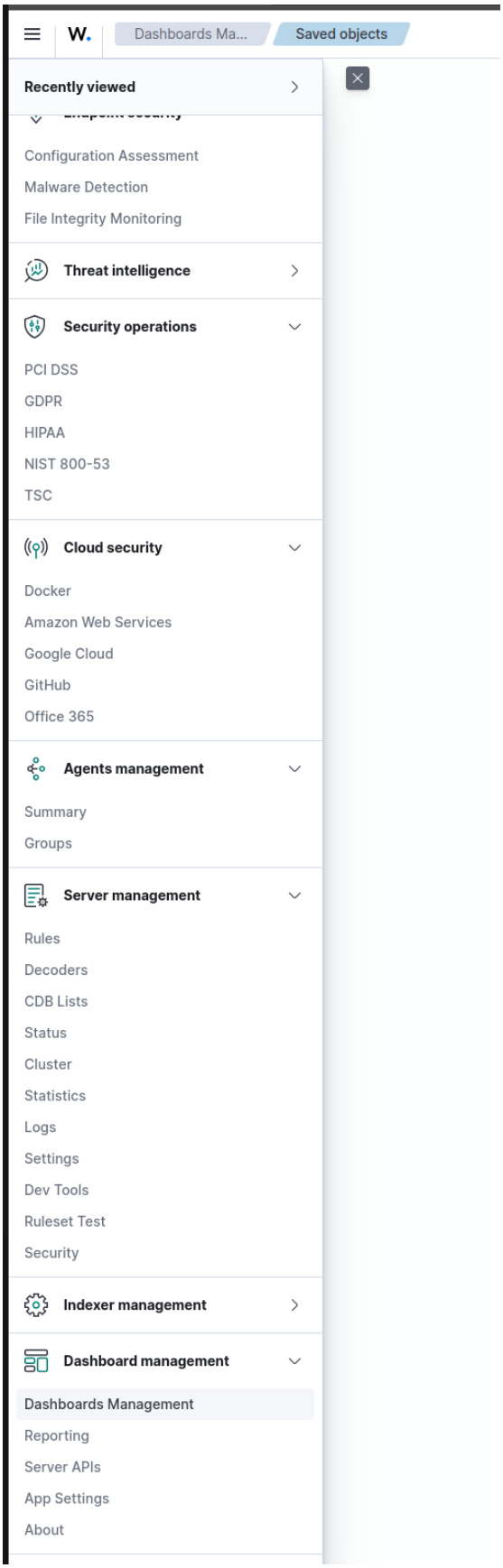
Click the save button after clicking the update button



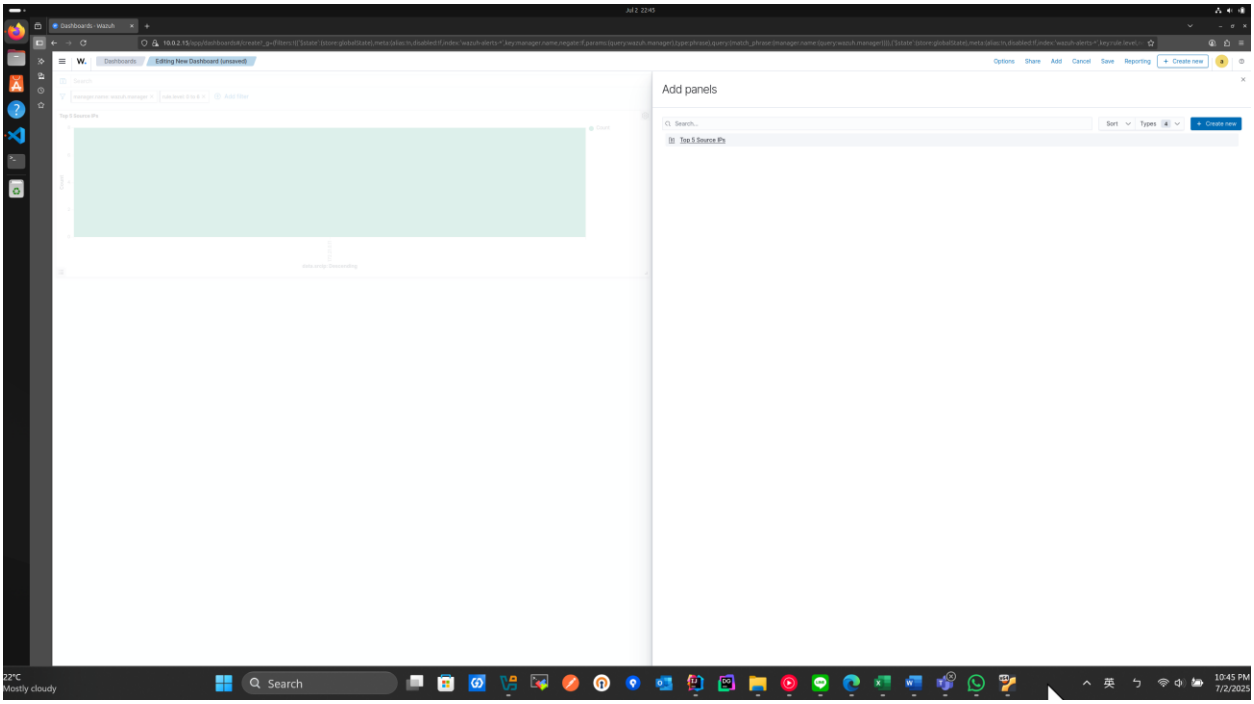


Go to the dashboard management and click it, then go to saved objects can see the object that you added



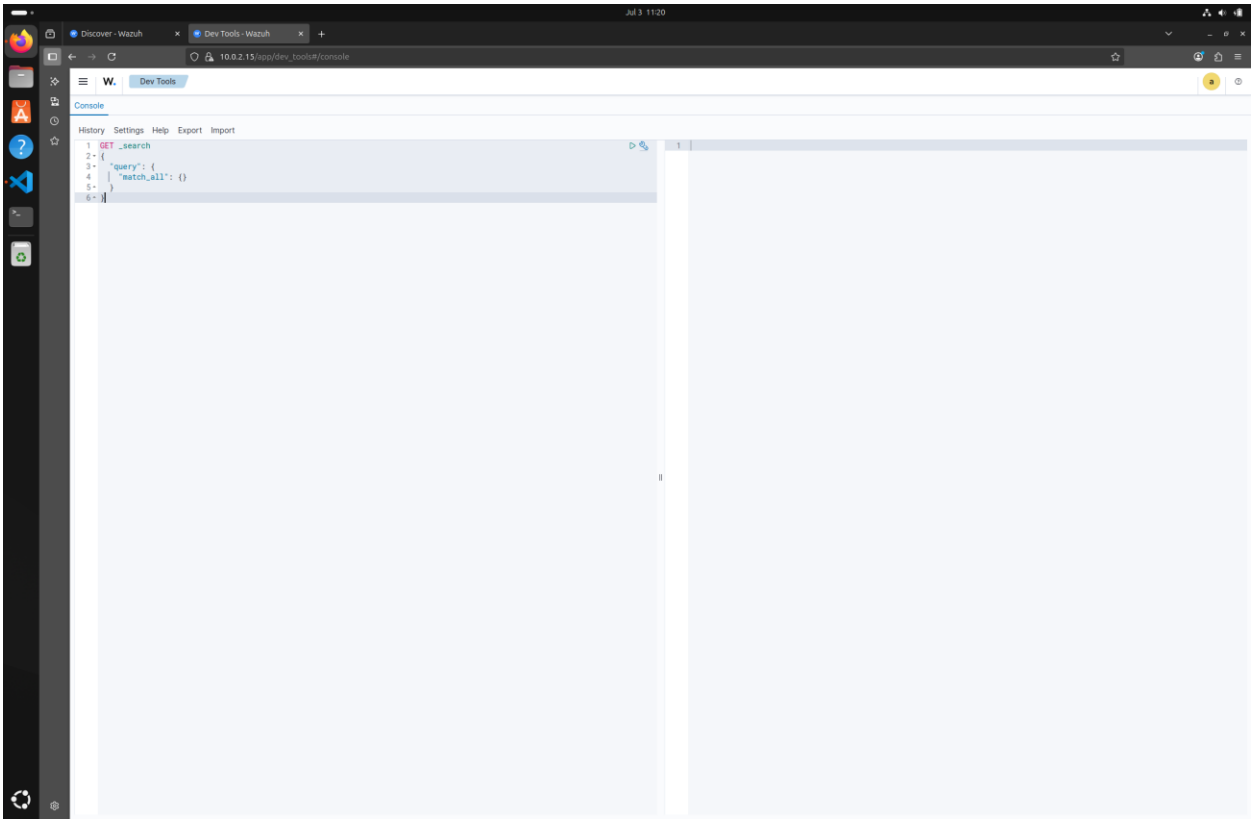

Dashboards Ma...
Saved objects
Recently viewed
Configuration Assessment
Malware Detection
File Integrity Monitoring
Threat intelligence
Security operations
PCI DSS
GDPR
HIPAA
NIST 800-53
TSC
Cloud security
Docker
Amazon Web Services
Google Cloud
GitHub
Office 365
Agents management
Summary
Groups
Server 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.

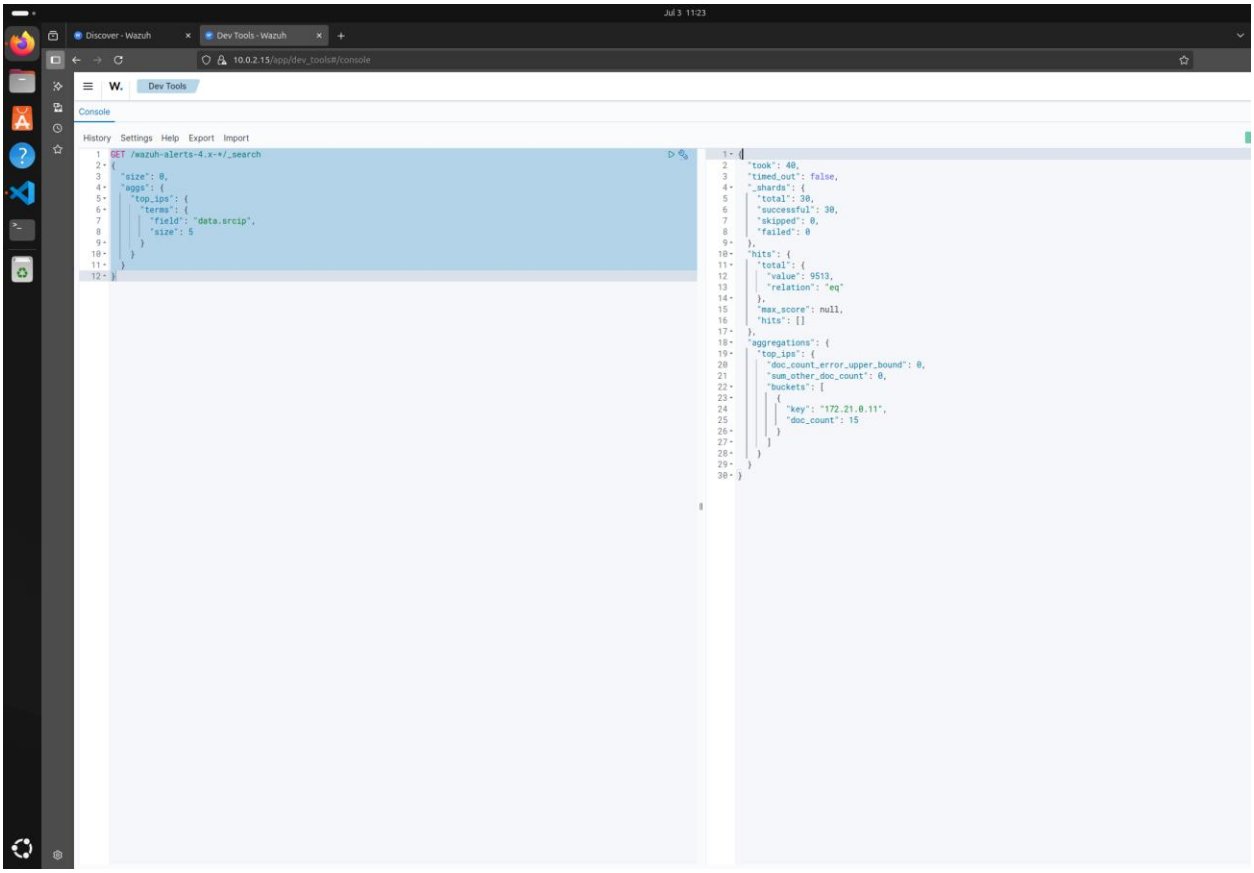


Configure Elasticsearch aggregation for Top 5 IPs

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

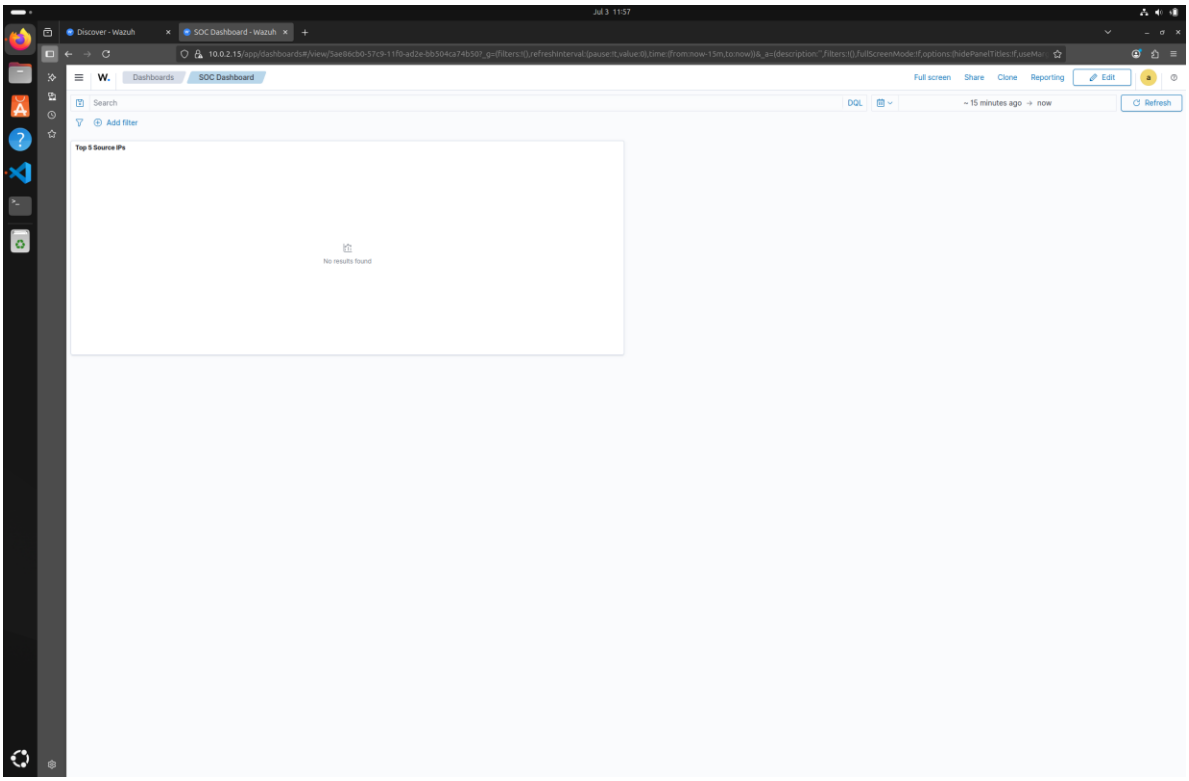


Enter the query information to search for the top 5 IPs, and we will retrieve the collected data. Successfully queried wazuh-alerts-4.x-* using Elasticsearch terms aggregation on data.srcip field. Verified IP frequency counts correctly, showing 172.21.0.11 appeared 15 times.

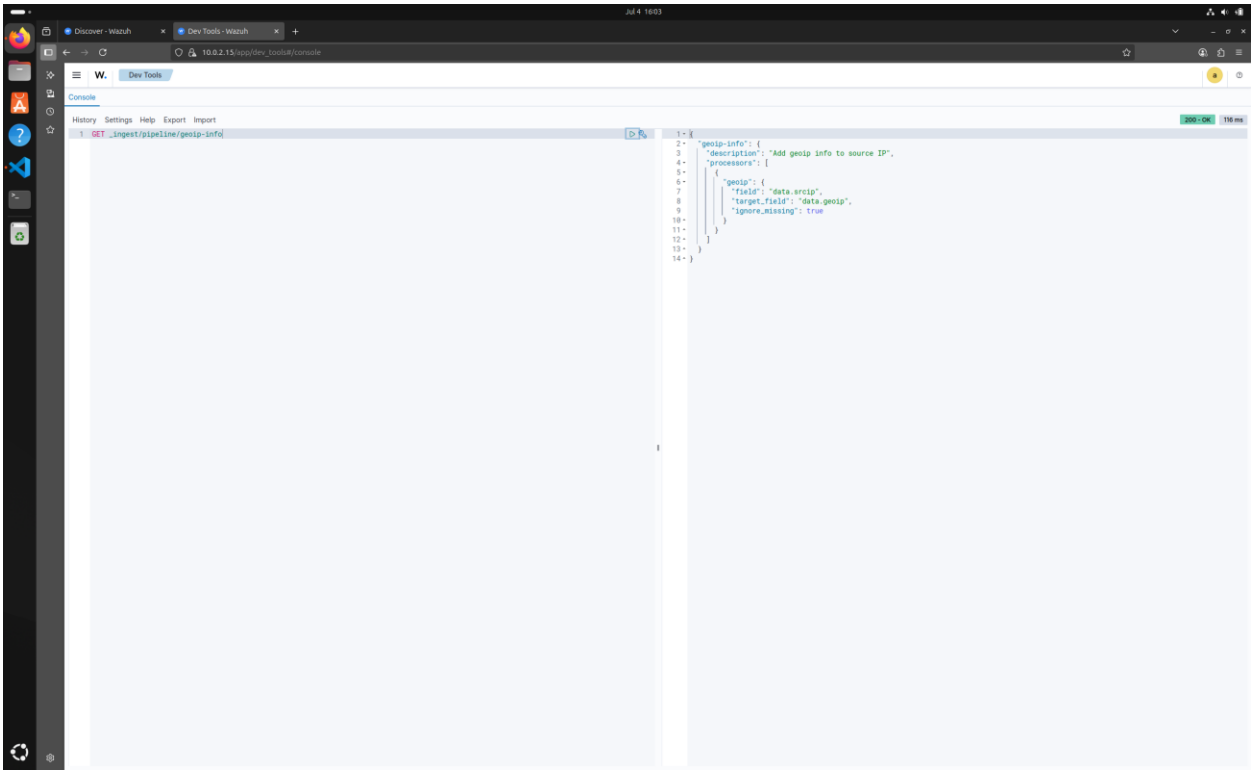


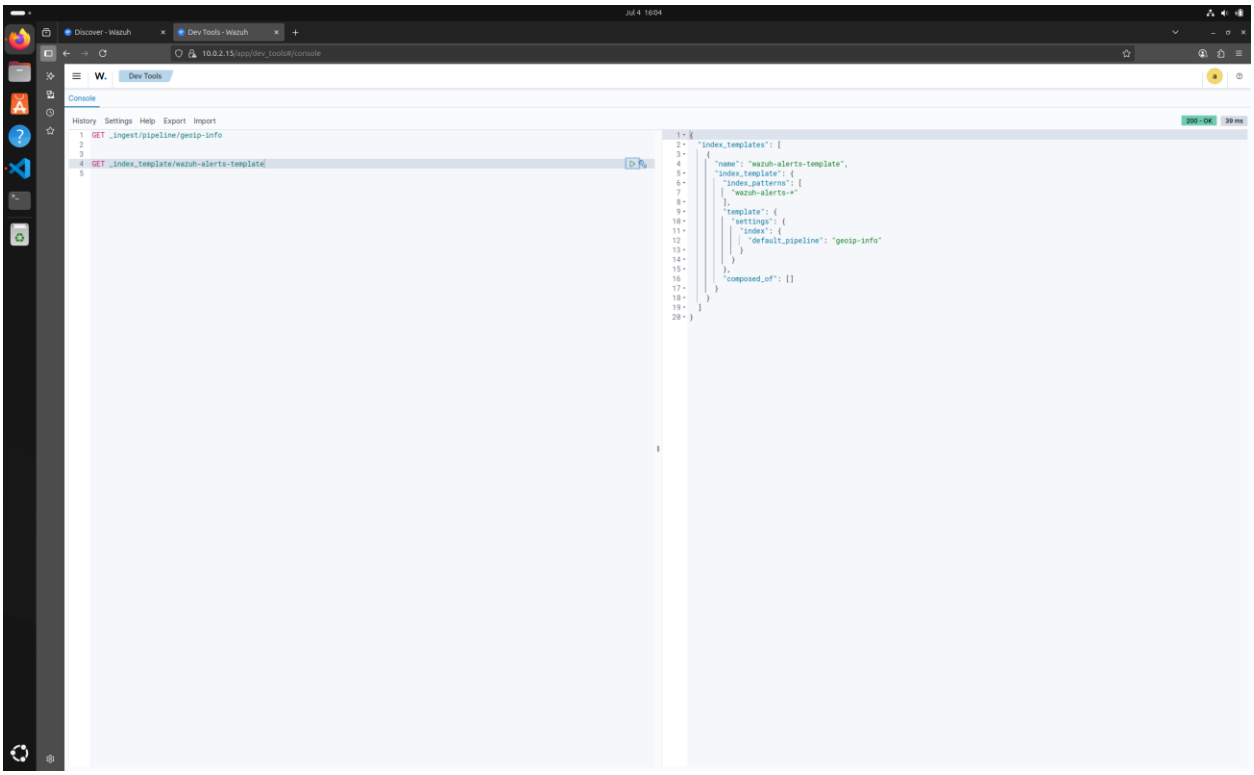
Test Kibana time and region filter

Time filter



region filter





Integrate GeoIP DB or API for IP location

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

  "description":
```

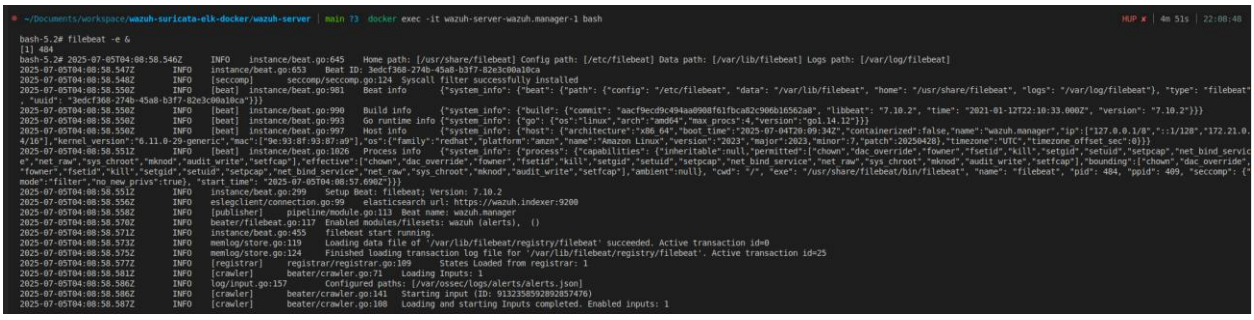
You can add a GeoIP enrichment pipeline to transform IP addresses into geographic coordinates (latitude and longitude) by using Elasticsearch Ingest Pipelines with the GeoIP processor.

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: '[redacted]'
  password: '[redacted]'
  ssl.verification_mode: 'full'
  ssl.certificate_authorities: ['/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
```

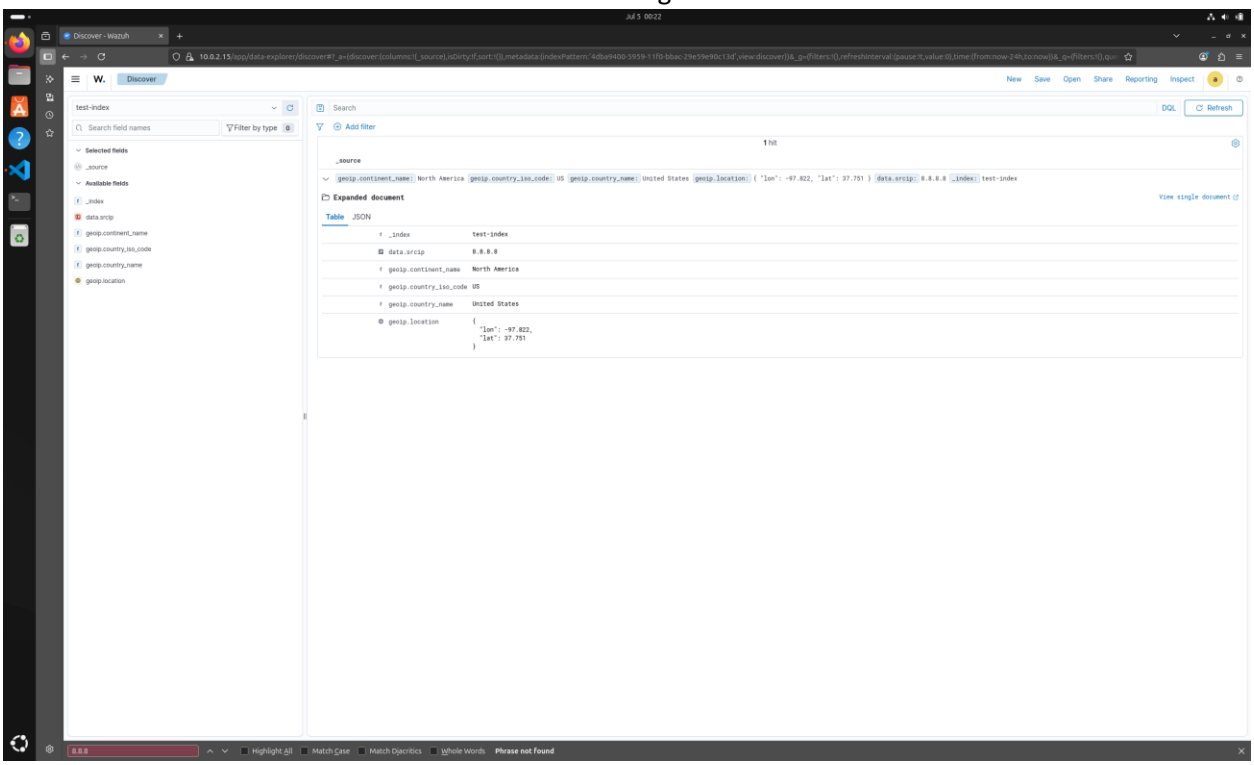
restart the wazuh-server-wazuh.manager-1



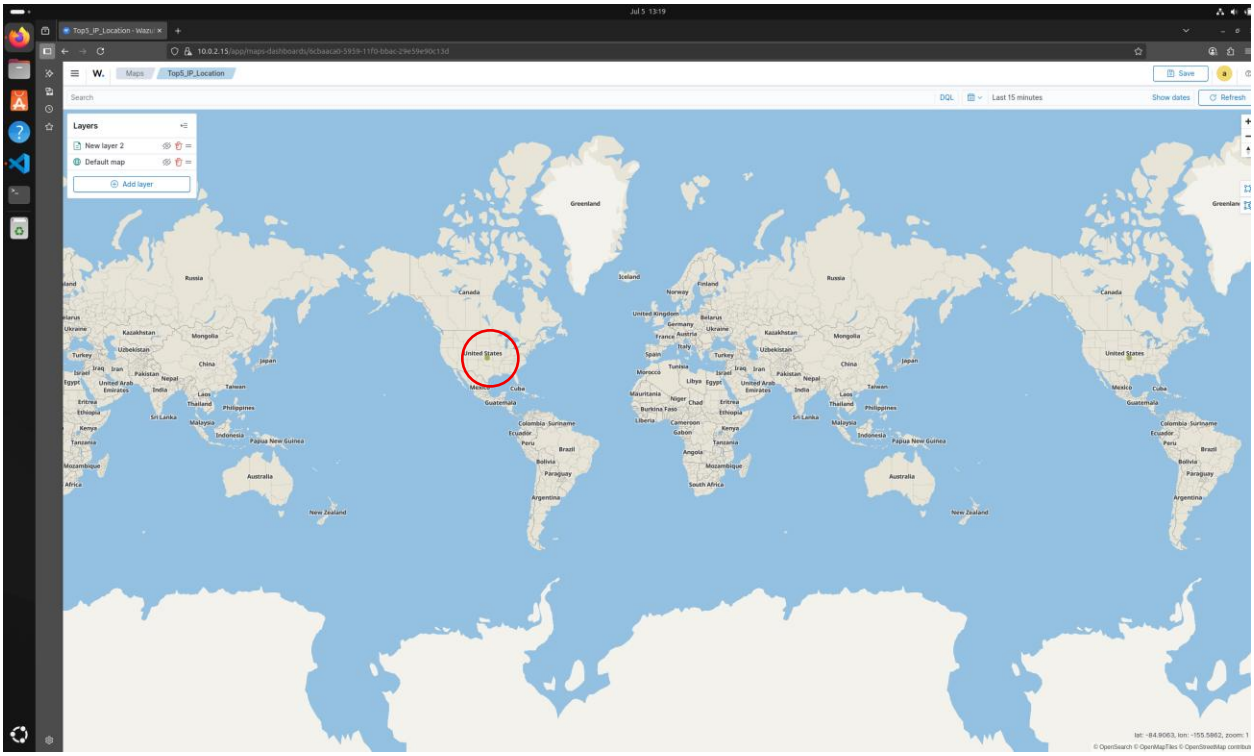
Add fake data (Since the Docker environment is isolated, I used mock data to simulate global IP traffic. If you want to test external traffic, you need to modify the Docker configuration to expose port 22 on the victim-server container, and also adjust the firewall settings to allow incoming traffic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

  }

}'
```

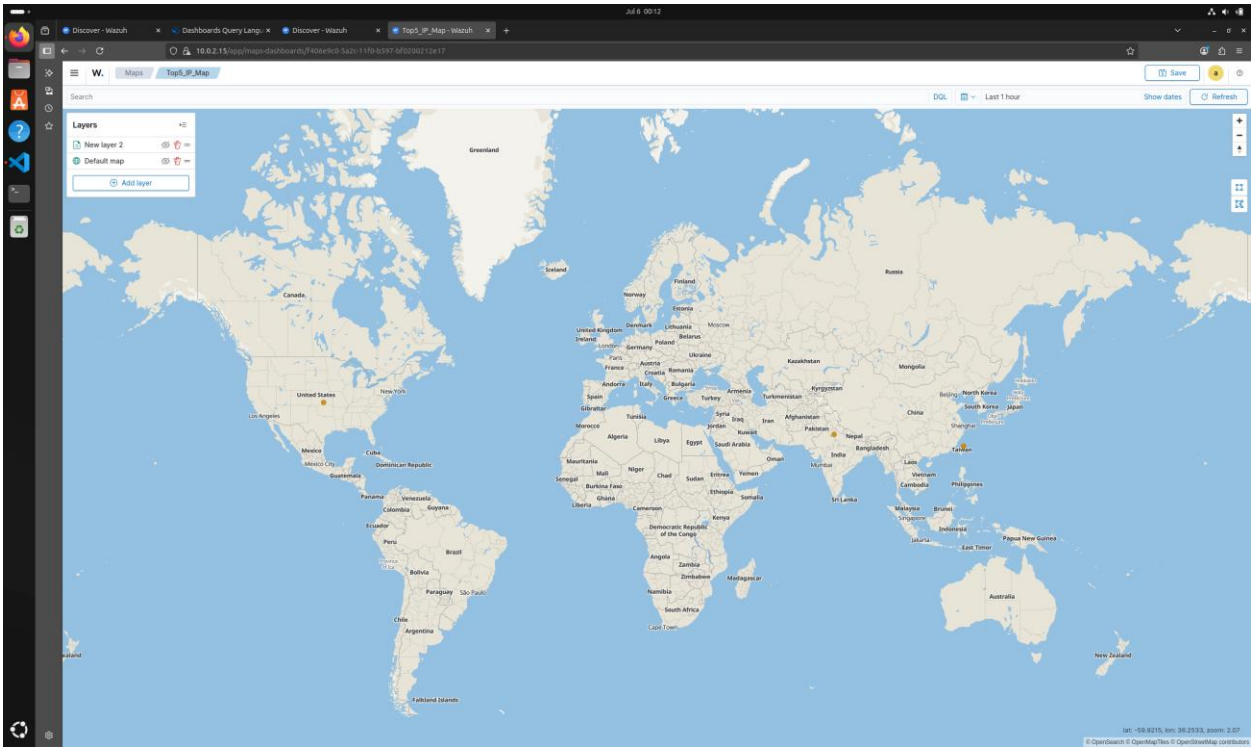
Check that the fake data on the dashboard includes longitude and latitude.



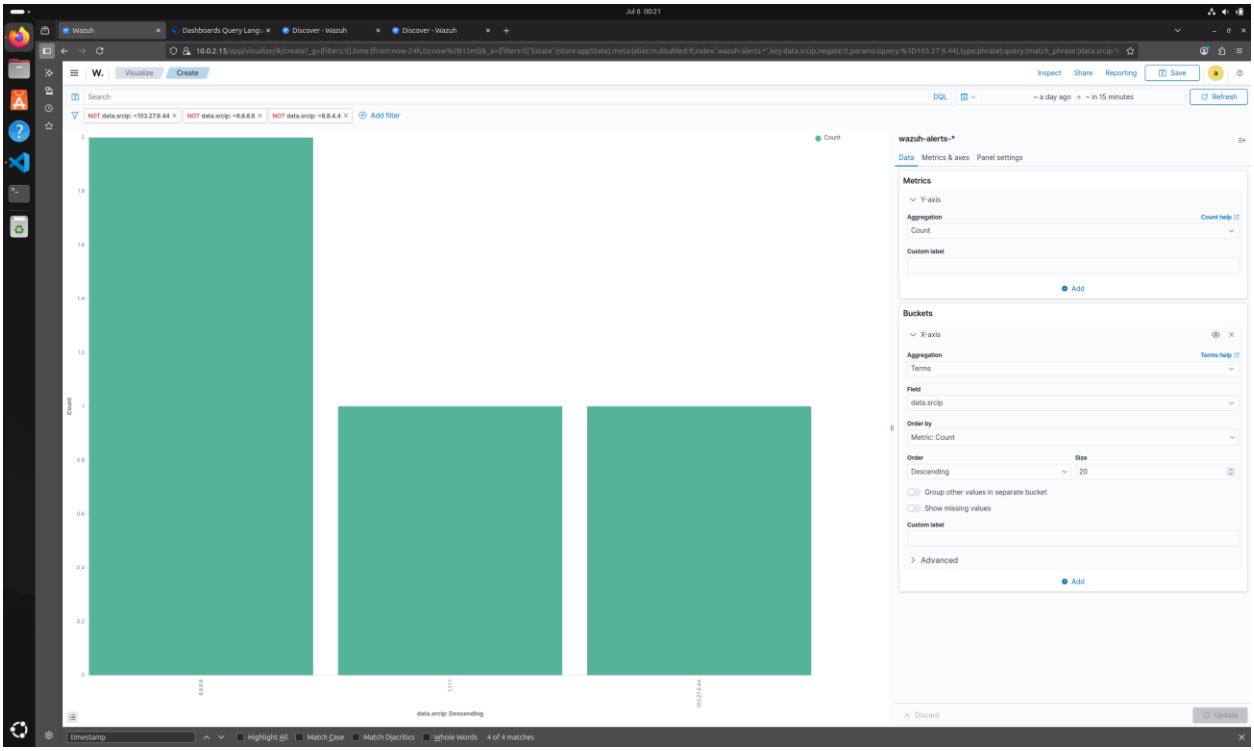
It will also show on the Map.



Add geographic map panel in Kibana



Configure bar chart parameters



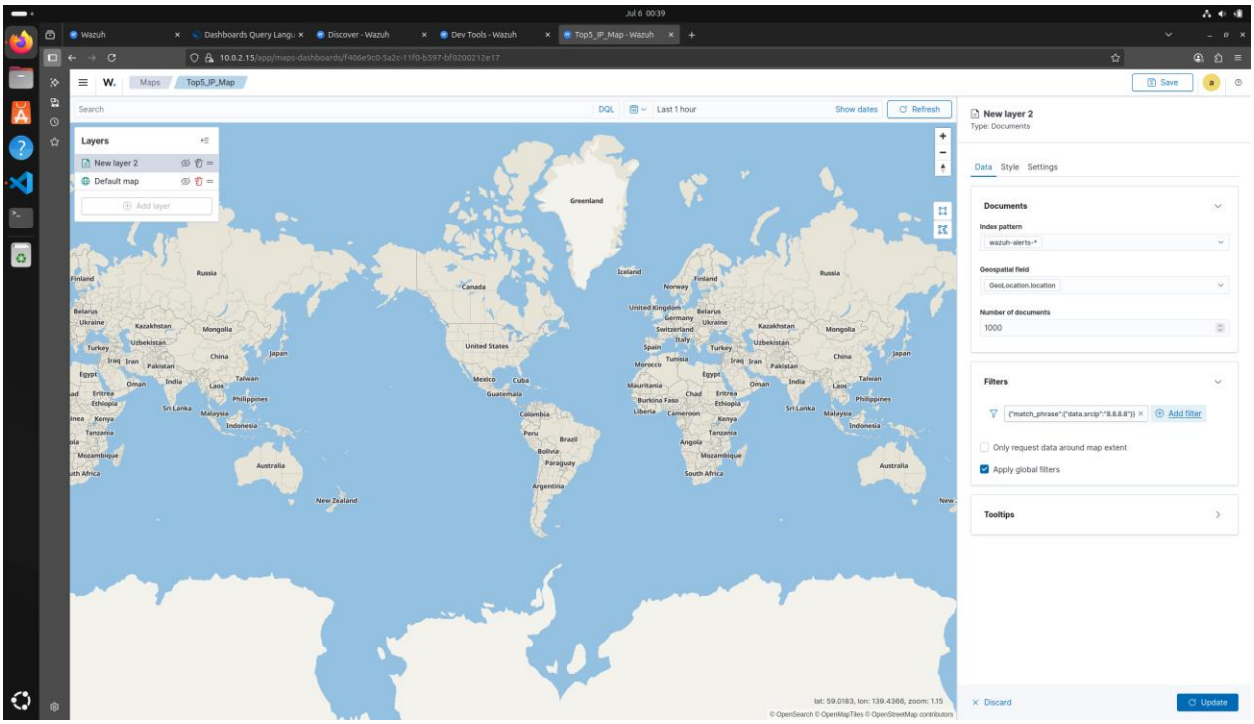
Go to the index dashboard and search for the top 5 most frequent IP addresses.

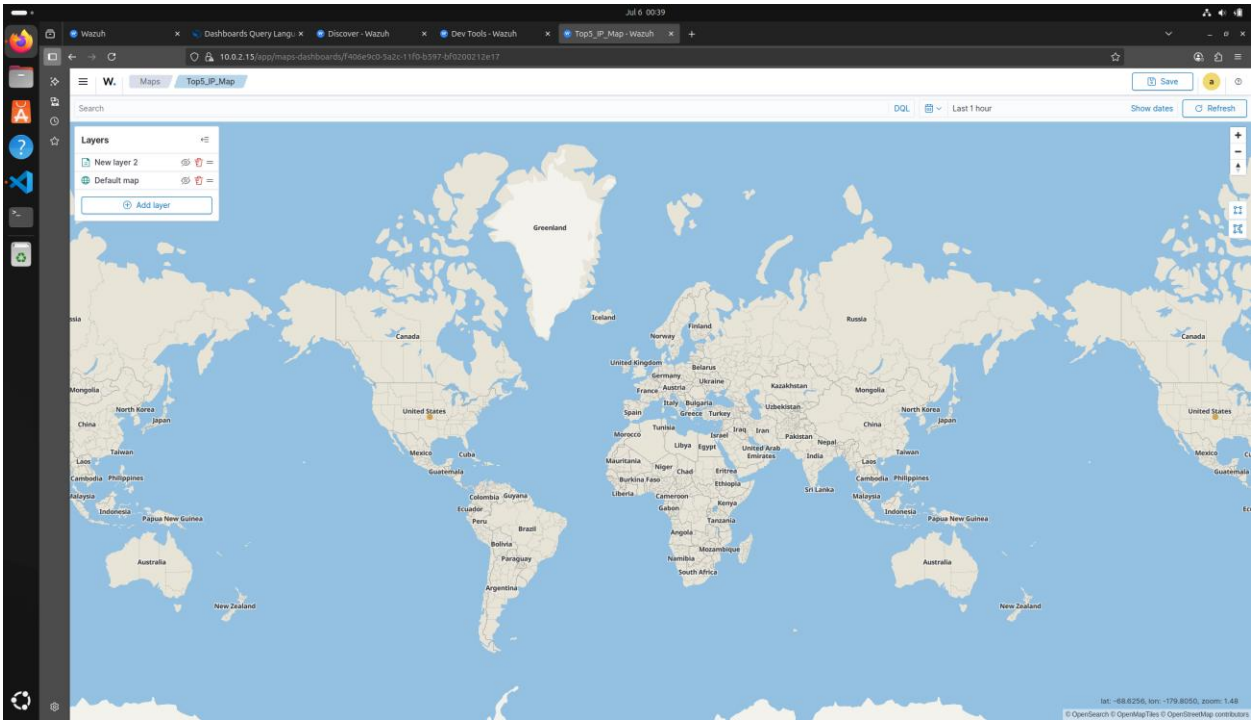
```
GET /wazuh-alerts-4.x-*/_search
{
  "size": 0,
  "aggs": {
    "top_ips": {
      "terms": {
        "field": "data.srcip",
        "size": 5
      }
    }
  }
}
```


```
{
  "took": 6,
  "timed_out": false,
  "_shards": {
    "total": 6,
    "successful": 6,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 6514,
      "relation": "eq"
    },
    "max_score": null,
    "hits": []
  },
  "aggregations": {
    "top_ips": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 4,
      "buckets": [
        {
          "key": "8.8.8.8",
          "doc_count": 10
        },
        {
          "key": "172.21.0.1",
          "doc_count": 5
        },
        {
          "key": "172.21.0.11",
          "doc_count": 5
        },
        {
          "key": "1.1.1.1",
          "doc_count": 2
        },
        {
          "key": "=8.8.8.8",
          "doc_count": 2
        }
      ]
    }
  }
}
```

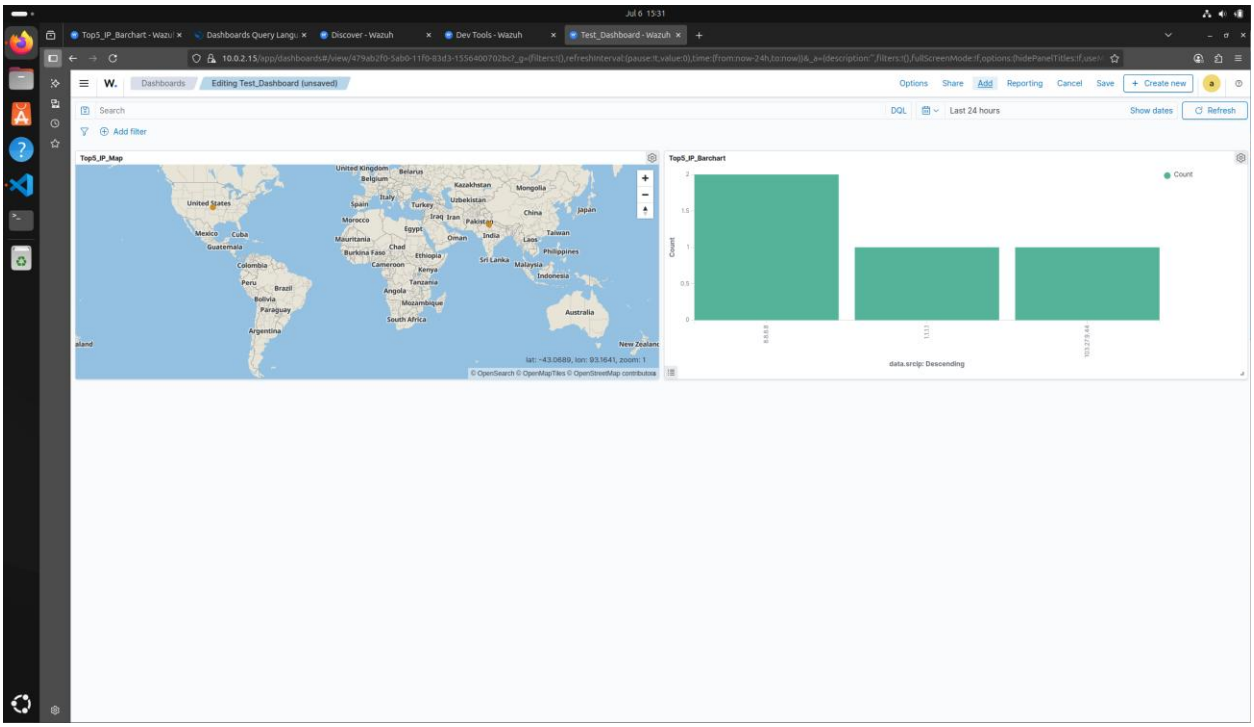
Configure map to filter charts on region click

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

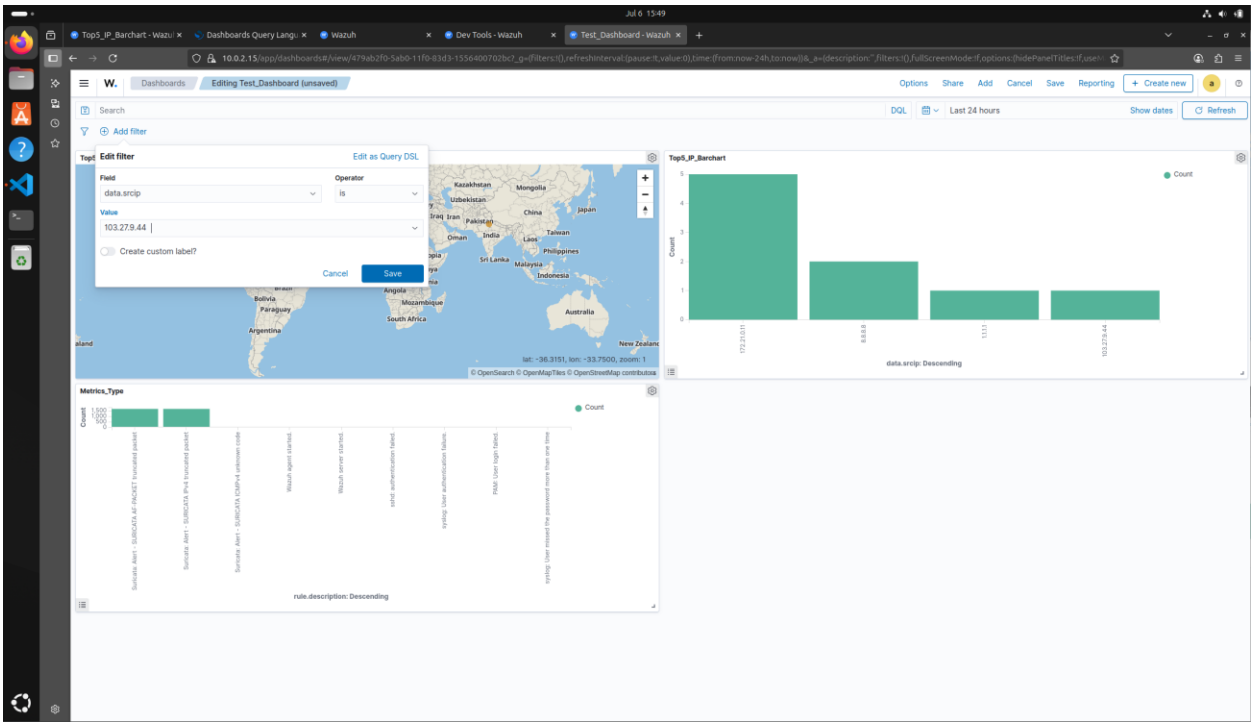


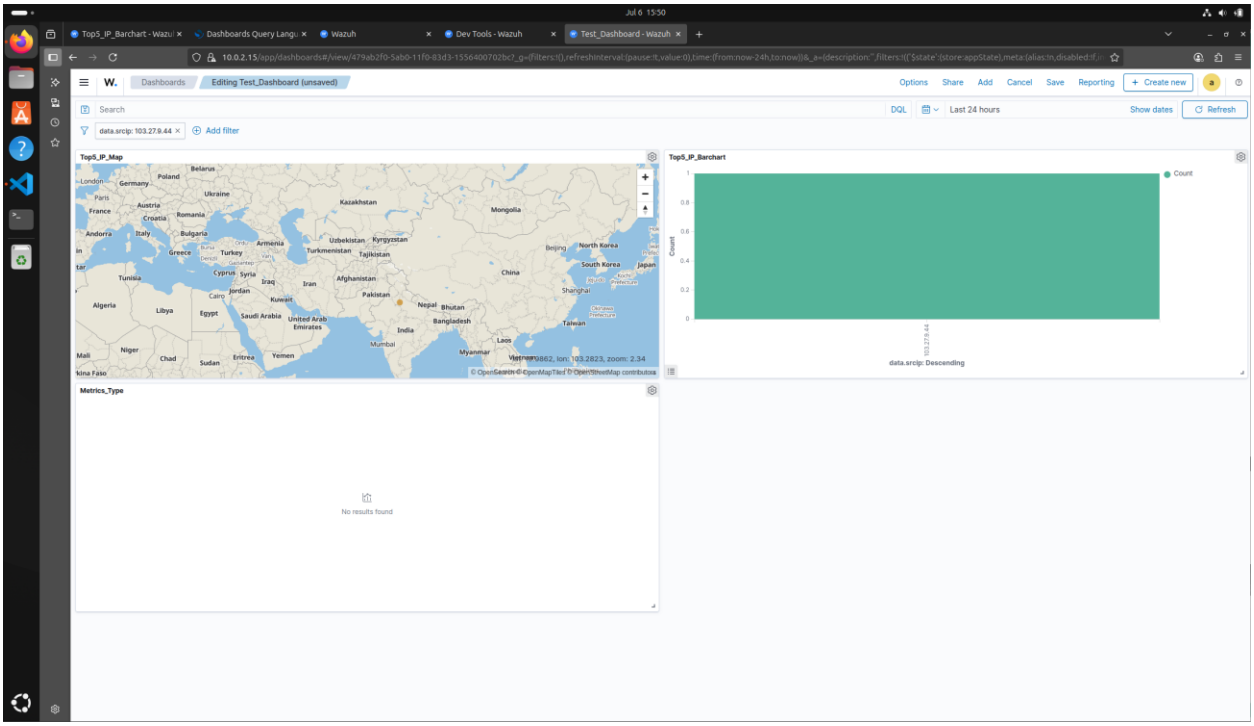


Test map interaction and filter function



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

```yaml
rules:
  - rule_id: 1
    rule_name: "Suspicious Port Scan"
    rule_description: "Detects if a source IP is scanning multiple ports on one or more destinations within a time window."
    rule_actions:
      - action: "email"
        email: "maggie.555x@gmail.com"
        template: "Suspicious Scan Detected: $(src_ip) is scanning $(dest_ip). Hit $(dest_port_count) unique ports in the last 60 seconds."
    rule_score: 5
    filter:
      - field: "data.flow.reason"
        operator: "eq"
        value: "timeout"
      - field: "data.tcp.rst"
        operator: "eq"
        value: "true"
    aggregation:
      type: "port_scan"
      time_window: 60
      group_by: "data.src_ip"
      unique_count_field: "data.dest_port"
      threshold: 5
```

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㉿4c8b503fcca2)-[/]
└─# 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 5
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 cythbgy@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 cythbgy@gmail.com
```

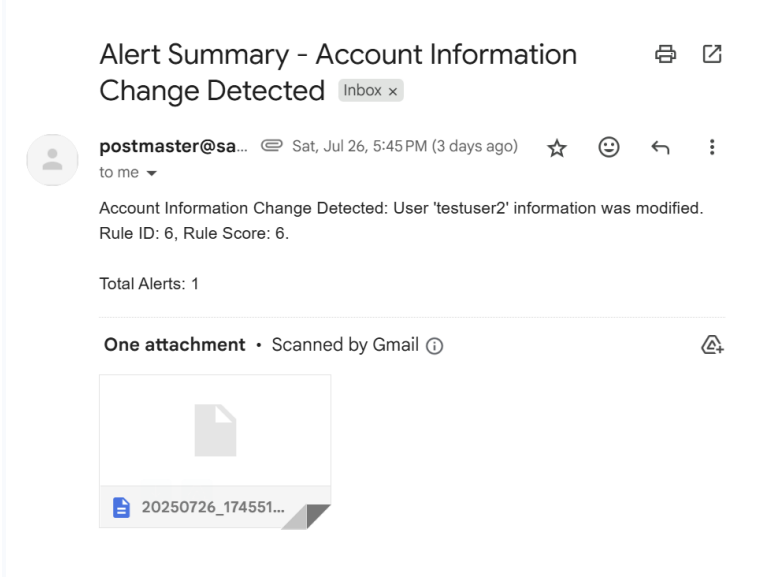
Schedule running effects

main.py alert_rule.yaml settings.yaml

config > alert_rule.yaml

```yaml
rules:
  - rule_id: 1
    rule_name: "Suspicious Port Scan"
    rule_description: "Detects if a source IP is scanning multiple ports on one or more destinations within a time window."
    rule_actions:
      - action: "email"
        email: "maggie.555x@gmail.com"
        template: "Suspicious Scan Detected: $(src_ip) is scanning $(dest_ip). Hit $(dest_port_count) unique ports in the last 60 seconds."
    rule_score: 5
    filter:
      - field: "data.flow.reason"
        operator: "eq"
        value: "timeout"
      - field: "data.tcp.rst"
        operator: "eq"
        value: "true"
    aggregation:
      type: "port_scan"
      time_window: 60
      group_by: "data.src_ip"
      unique_count_field: "data.dest_port"
      threshold: 5
```

alert rule



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		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.

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 Type	Description
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

[Demo Video: System Setup & Attack Simulation Walkthrough]

Contact Email:

catherine.chen.canadait@gmail.com

maggie.555x@gmail.com

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