

# Implementation of Security Operations Center Based on Wazuh Tool

---

## 1. Goal and Tasks of the Project

### Goal

- To build a fully functional Security Operations Center (SOC) using open-source tools (Wazuh (SIEM), MISP (Threat Intelligence Platform), IRIS (Ticketing System)).
- Enhance capabilities for detecting, analyzing, and responding to security incidents using SIEM, threat intelligence, and a ticketing system.

### Tasks and Responsibilities

#### 1. **Mohamad Nour Shahin:** Infrastructure Setup.

- Set up the virtual environment for the SOC.
- Install and configure the Wazuh SIEM tool.
- Document the installation and configuration process for Wazuh.
- Create a basic incident dashboard in Wazuh for tracking events.
- Ensure connectivity between Wazuh and other tools (e.g., MISP, IRIS).

#### 2. **Yehia Sobeh:** Integration of MISP for threat intelligence.

- Install and configure MISP as the threat intelligence platform.
- Integrate MISP with Wazuh for contextual threat information.
- Document the integration process and configuration steps.
- Test threat intelligence data flow into Wazuh and provide sample scenarios.
- Propose mechanisms to enrich threat data using MISP.

#### • **Ammar Meslmani:** Configuration of IRIS for case management and incident automation.

- Install and configure IRIS as the ticketing system for incident tracking.
- Link IRIS with Wazuh to automatically log security incidents.
- Develop and test automated responses to at least two security incidents (e.g., failed login attempts, malware detection).
- Document automation workflows and response mechanisms.
- Simulate the incident management process for the demo.

- **Ali Hamdan:** Testing, incident simulation, and documentation.
    - Test the overall integration and functionality of the SOC setup.
    - Identify and document potential issues during integration and solutions implemented.
    - Write scripts to simulate security incidents for testing (e.g., running a vulnerability scanner, mock phishing attacks).
    - Prepare the final demo presentation and record it for submission.
    - Organize and compile the documentation from all members into a cohesive report.
- 

## 2. Execution Plan and Methodology

### Plan for the Solution

1. **Setup:**
  - Deploy virtual machines or Docker containers for Wazuh, MISP, and IRIS.
  - Establish network connectivity and API integrations.
2. **Integration:**
  - Configure Wazuh to collect logs and generate alerts.
  - Integrate MISP for threat intelligence enrichment.
  - Set up IRIS for incident tracking and automation workflows.
3. **Testing:**
  - Simulate real-world security incidents (e.g., brute force attacks, malware detection).
  - Automate responses and document workflows.

### Planned Infrastructure

- Diagram illustrating connectivity between Wazuh, MISP, IRIS, and client systems.

### Methodology

- Follow official documentation for tool installations and configurations.
  - Implement and test workflows for data flow, alert generation, and automated responses.
  - Document challenges and solutions.
- 

## 3. Development of Solution and Tests (Proof of Concept)

### Environment Preparation

- Description of the SOC setup: VMs/containers, OS versions, and configurations.

1. Setup Wazuh server on Ubuntu 22.04 (Ali Device number one):

- Download and run the Wazuh installation assistant:

```
curl -s0 https://packages.wazuh.com/4.9/wazuh-install.sh
&& sudo bash ./wazuh-install.sh -a -o
```

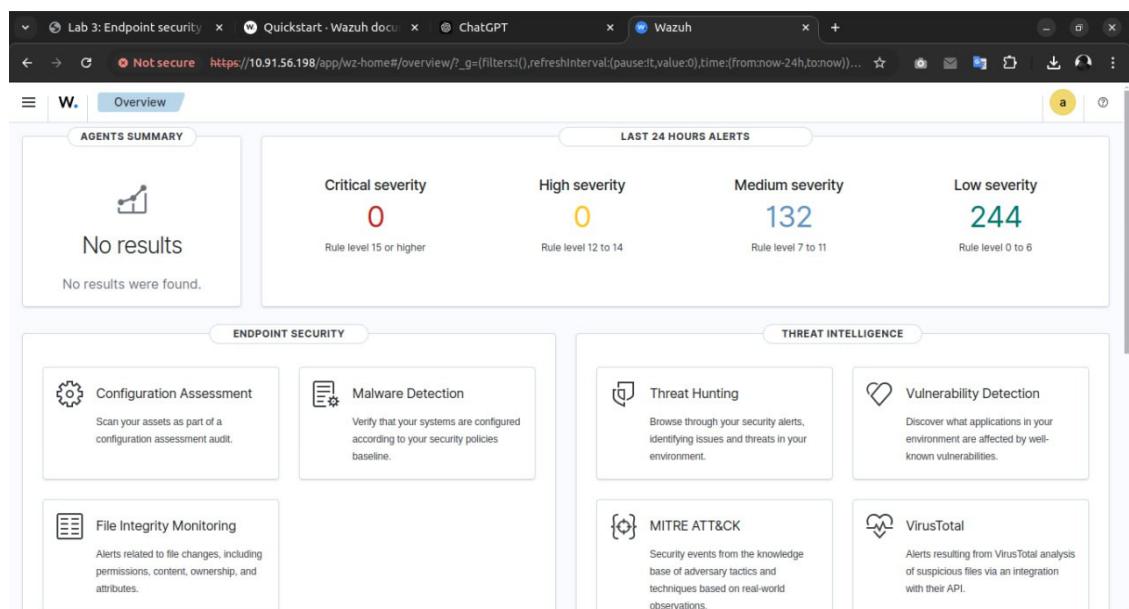
- Configure Wazuh, install Wazuh Indexer, install Wazuh Server, and install Wazuh Dashboard.

```
[root@all-workstation ~]# curl -s0 https://packages.wazuh.com/4.9/wazuh-install.sh && sudo bash ./wazuh-install.sh -a
25/12/2024 14:35:01 INFO: Starting Wazuh installation assistant. Wazuh version: 4.9.2
25/12/2024 14:35:01 INFO: Verbose logging redirected to /var/log/wazuh-install.log
25/12/2024 14:35:15 INFO: Verifying that your system meets the recommended minimum hardware requirements.
25/12/2024 14:35:15 INFO: Wazuh web interface port will be 443.
25/12/2024 14:35:34 INFO: Wazuh repository added.
25/12/2024 14:35:34 INFO: --- Configuration files ...
25/12/2024 14:35:34 INFO: Generating configuration files.
25/12/2024 14:35:35 INFO: Generating the root certificate.
25/12/2024 14:35:36 INFO: Generating admin certificates.
25/12/2024 14:35:36 INFO: Generating Wazuh indexer certificates.
25/12/2024 14:35:36 INFO: Generating filebeat certificates.
25/12/2024 14:35:36 INFO: Generating Wazuh dashboard certificates.
25/12/2024 14:35:37 INFO: Created wazuh-install-files.tar. It contains the Wazuh cluster key, certificates, and passwords necessary for installation.
25/12/2024 14:35:37 INFO: Wazuh indexer ...
25/12/2024 14:35:37 INFO: Starting Wazuh indexer installation.
25/12/2024 14:39:28 INFO: Wazuh indexer installation finished.
25/12/2024 14:39:28 INFO: Wazuh indexer post-install configuration finished.
25/12/2024 14:39:28 INFO: Starting service wazuh-indexer.
25/12/2024 14:39:47 INFO: wazuh-indexer service started.
25/12/2024 14:39:47 INFO: Initializing Wazuh indexer cluster security settings.
25/12/2024 14:39:53 INFO: Wazuh indexer cluster security configuration initialized.
25/12/2024 14:39:53 INFO: Wazuh indexer cluster initialized.
25/12/2024 14:39:53 INFO: --- Wazuh server ---
25/12/2024 14:39:53 INFO: Starting the Wazuh manager installation.
25/12/2024 14:42:26 INFO: Wazuh manager installation finished.
25/12/2024 14:42:28 INFO: Wazuh manager vulnerability detection configuration finished.
25/12/2024 14:42:28 INFO: Starting service wazuh-manager.
25/12/2024 14:42:49 INFO: wazuh-manager service started.
25/12/2024 14:42:49 INFO: Starting Filebeat installation.
25/12/2024 14:43:21 INFO: Filebeat installation finished.
25/12/2024 14:43:26 INFO: Filebeat post-install configuration finished.
25/12/2024 14:43:26 INFO: Starting service filebeat.
25/12/2024 14:43:28 INFO: filebeat service started.
25/12/2024 14:43:28 INFO: --- Wazuh dashboard ...
```

```
[root@all-workstation ~]# curl -s0 https://packages.wazuh.com/4.9/wazuh-install.sh && sudo bash ./wazuh-install.sh -a
25/12/2024 14:43:28 INFO: --- Wazuh dashboard ...
25/12/2024 14:43:28 INFO: Starting Wazuh dashboard installation.
25/12/2024 14:45:33 INFO: Wazuh dashboard installation finished.
25/12/2024 14:45:33 INFO: Wazuh dashboard post-install configuration finished.
25/12/2024 14:45:33 INFO: Starting service wazuh-dashboard.
25/12/2024 14:45:34 INFO: wazuh-dashboard service started.
25/12/2024 14:45:36 INFO: Updating the internal users.
25/12/2024 14:45:42 INFO: A backup of the internal users has been saved in the /etc/wazuh-Indexer/internalusers-backup folder.
25/12/2024 14:45:54 INFO: The filebeat.yml file has been updated to use the Filebeat Keystore username and password.
25/12/2024 14:46:29 INFO: Initializing Wazuh dashboard web application.
25/12/2024 14:46:29 INFO: Wazuh dashboard web application not yet initialized. Waiting...
25/12/2024 14:46:45 INFO: Wazuh dashboard web application not yet initialized. Waiting...
25/12/2024 14:47:00 INFO: Wazuh dashboard web application initialized.
25/12/2024 14:47:00 INFO: --- Summary ---
25/12/2024 14:47:00 INFO: You can access the web interface https://<wazuh-dashboard-ip>:443
    User: admin
    Password: scZ.E3BKusUFJ8ZqKFTIyrxH4CRkLBL+
25/12/2024 14:47:00 INFO: Installation finished.
[root@all-workstation ~]
```

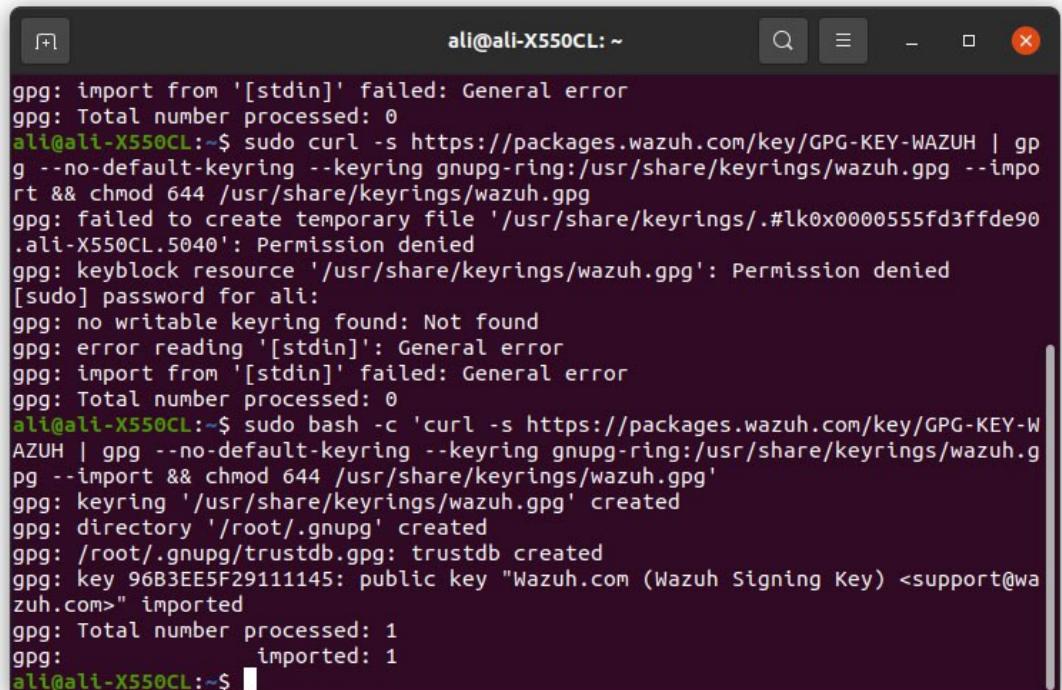
- Access the Wazuh web interface with <https://10.91.56.198:443> and my credentials:



## 2. Setup Wazuh Agent on Ubuntu 20.04 (Ali Device number two):

- Install the GPG key:

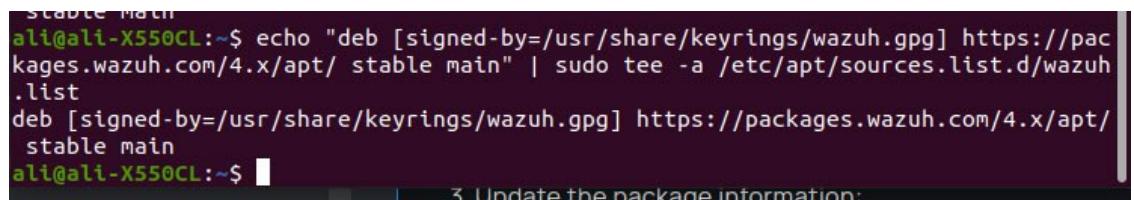
```
curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gpg  
--no-default-keyring --keyring gnupg-  
ring:/usr/share/keyrings/wazuh.gpg --import && chmod 644  
/usr/share/keyrings/wazuh.gpg
```



```
gpg: import from '[stdin]' failed: General error  
gpg: Total number processed: 0  
ali@ali-X550CL:~$ sudo curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gp  
g --no-default-keyring --keyring gnupg-ring:/usr/share/keyrings/wazuh.gpg --imp  
rt && chmod 644 /usr/share/keyrings/wazuh.gpg  
gpg: failed to create temporary file '/usr/share/keyrings/.#lk0x0000555fd3ffde90  
.ali-X550CL.5040': Permission denied  
gpg: keyblock resource '/usr/share/keyrings/wazuh.gpg': Permission denied  
[sudo] password for ali:  
gpg: no writable keyring found: Not found  
gpg: error reading '[stdin]': General error  
gpg: import from '[stdin]' failed: General error  
gpg: Total number processed: 0  
ali@ali-X550CL:~$ sudo bash -c 'curl -s https://packages.wazuh.com/key/GPG-KEY-W  
AZUH | gpg --no-default-keyring --keyring gnupg-ring:/usr/share/keyrings/wazuh.g  
pg --import && chmod 644 /usr/share/keyrings/wazuh.gpg'  
gpg: keyring '/usr/share/keyrings/wazuh.gpg' created  
gpg: directory '/root/.gnupg/trustdb.gpg' created  
gpg: /root/.gnupg/trustdb.gpg: trustdb created  
gpg: key 96B3EE5F29111145: public key "Wazuh.com (Wazuh Signing Key) <support@wa  
zuh.com>" imported  
gpg: Total number processed: 1  
gpg: imported: 1  
ali@ali-X550CL:~$
```

- Add the repository:

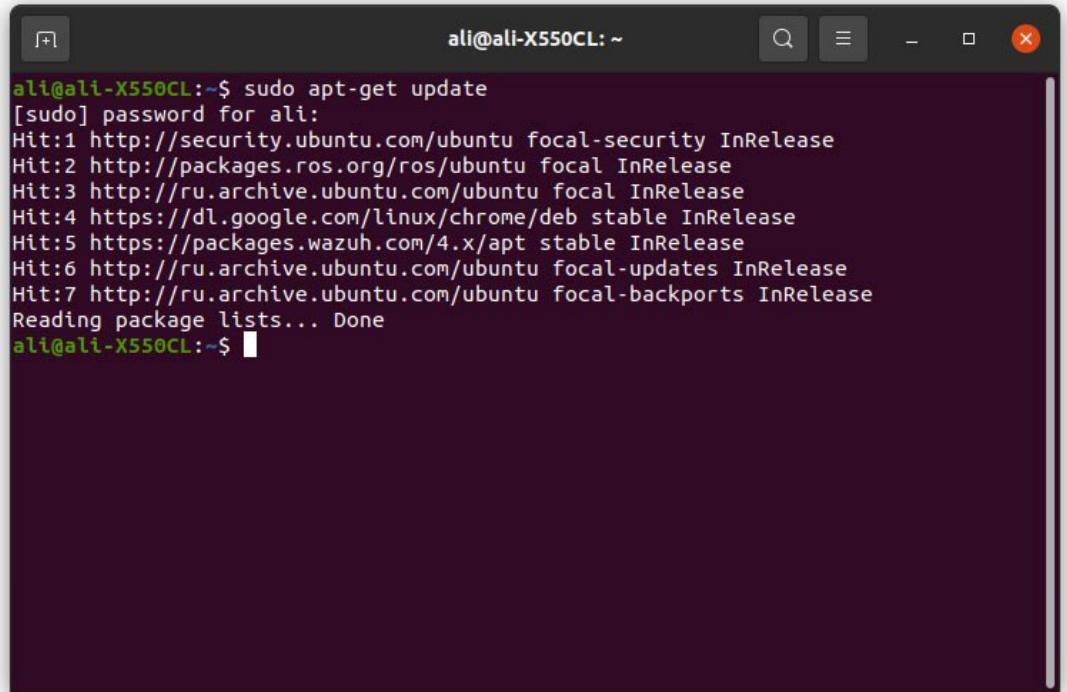
```
echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg]  
https://packages.wazuh.com/4.x/apt/ stable main" | tee -a  
/etc/apt/sources.list.d/wazuh.list
```



```
stable main  
ali@ali-X550CL:~$ echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://pac  
kages.wazuh.com/4.x/apt/ stable main" | sudo tee -a /etc/apt/sources.list.d/wazuh  
.list  
deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://packages.wazuh.com/4.x/apt/  
stable main  
ali@ali-X550CL:~$
```

- Update the package information:

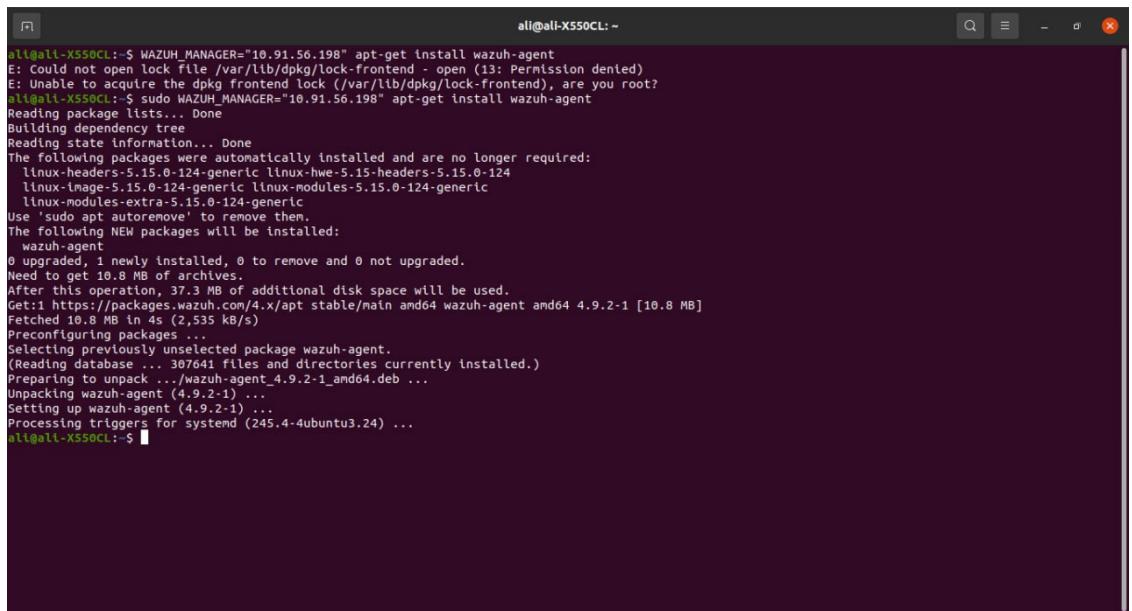
```
sudo apt-get update
```



```
ali@ali-X550CL:~$ sudo apt-get update
[sudo] password for ali:
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:2 http://packages.ros.org/ros/ubuntu focal InRelease
Hit:3 http://ru.archive.ubuntu.com/ubuntu focal InRelease
Hit:4 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:5 https://packages.wazuh.com/4.x/apt stable InRelease
Hit:6 http://ru.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:7 http://ru.archive.ubuntu.com/ubuntu focal-backports InRelease
Reading package lists... Done
ali@ali-X550CL:~$
```

- Edit the WAZUH\_MANAGER variable to contain our Wazuh manager IP address or hostname **10.91.56.198**:

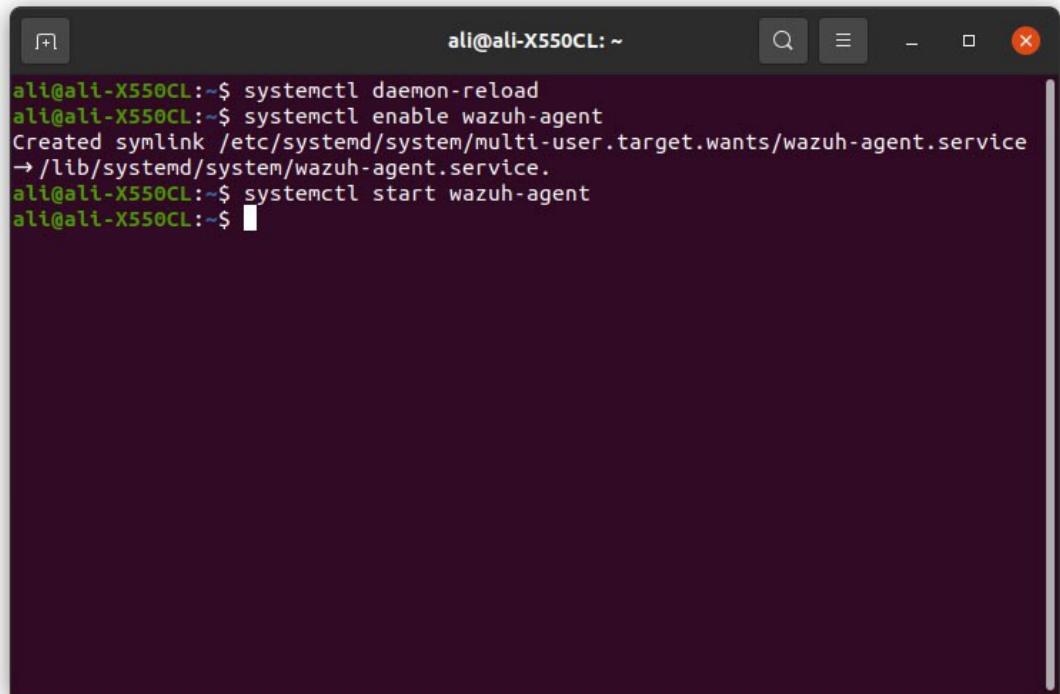
```
WAZUH_MANAGER="10.91.56.198" apt-get install wazuh-agent
```



```
ali@ali-X550CL:~$ WAZUH_MANAGER="10.91.56.198" apt-get install wazuh-agent
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission denied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root?
ali@ali-X550CL:~$ sudo WAZUH_MANAGER="10.91.56.198" apt-get install wazuh-agent
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-5.15.0-124-generic linux-hwe-5.15-headers-5.15.0-124
  linux-image-5.15.0-124-generic linux-modules-5.15.0-124-generic
  linux-modules-extra-5.15.0-124-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  wazuh-agent
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 10.8 MB of archives.
After this operation, 37.3 MB of additional disk space will be used.
Get:1 https://packages.wazuh.com/4.x/apt stable/main amd64 wazuh-agent amd64 4.9.2-1 [10.8 MB]
Fetched 10.8 MB in 4s (2,535 kB/s)
Preconfiguring packages ...
Selecting previously unselected package wazuh-agent.
(Reading database ... 307641 files and directories currently installed.)
Preparing to unpack .../wazuh-agent_4.9.2-1_amd64.deb ...
Unpacking wazuh-agent (4.9.2-1) ...
Setting up wazuh-agent (4.9.2-1) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
ali@ali-X550CL:~$
```

- Enable and start the Wazuh agent service:

```
systemctl daemon-reload
systemctl enable wazuh-agent
systemctl start wazuh-agent
```



```
ali@ali-X550CL:~$ systemctl daemon-reload
ali@ali-X550CL:~$ systemctl enable wazuh-agent
Created symlink /etc/systemd/system/multi-user.target.wants/wazuh-agent.service
→ /lib/systemd/system/wazuh-agent.service.
ali@ali-X550CL:~$ systemctl start wazuh-agent
ali@ali-X550CL:~$
```

### 3. Setup IRIS on Ubuntu 22.04 (Ammar Device):

- Setup Docker on the device:

(DOCKER PART)

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Add the repository to Apt sources:
echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
sudo docker run hello-world
sudo groupadd docker
sudo gpasswd -a $USER docker
newgrp docker
docker run hello-world
```

- Intalling the IRIS on the device:

```

# Clone the iris-web repository
git clone https://github.com/dfir-iris/iris-web.git
cd iris-web

# Checkout to the last tagged version
git checkout v2.4.16

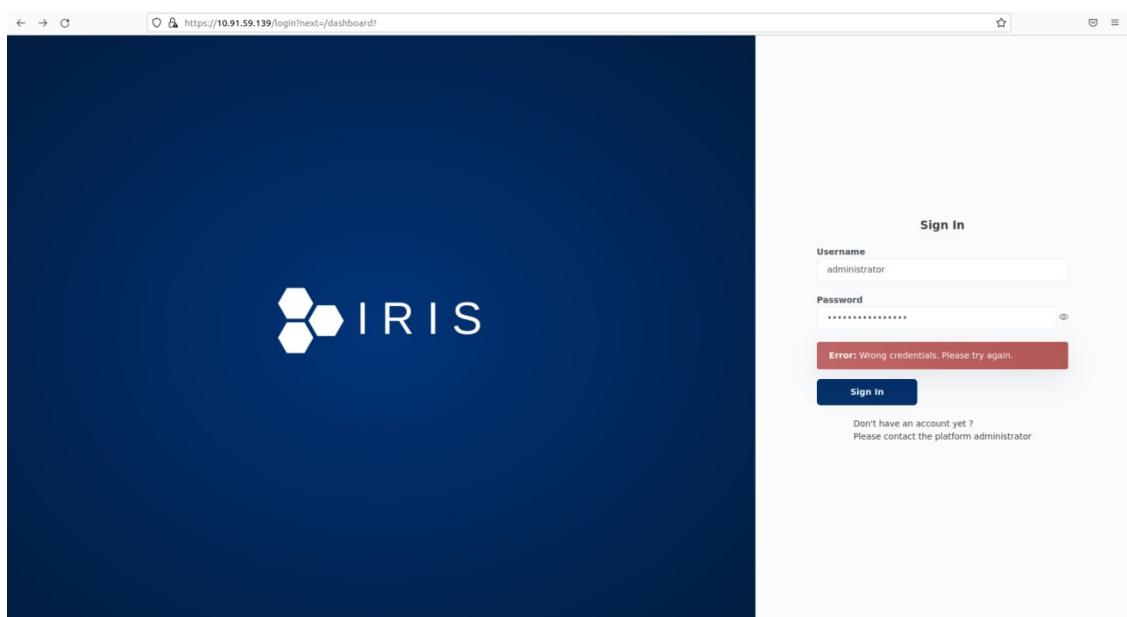
# Copy the environment file
cp .env.model .env

# Pull the dockers
docker compose pull

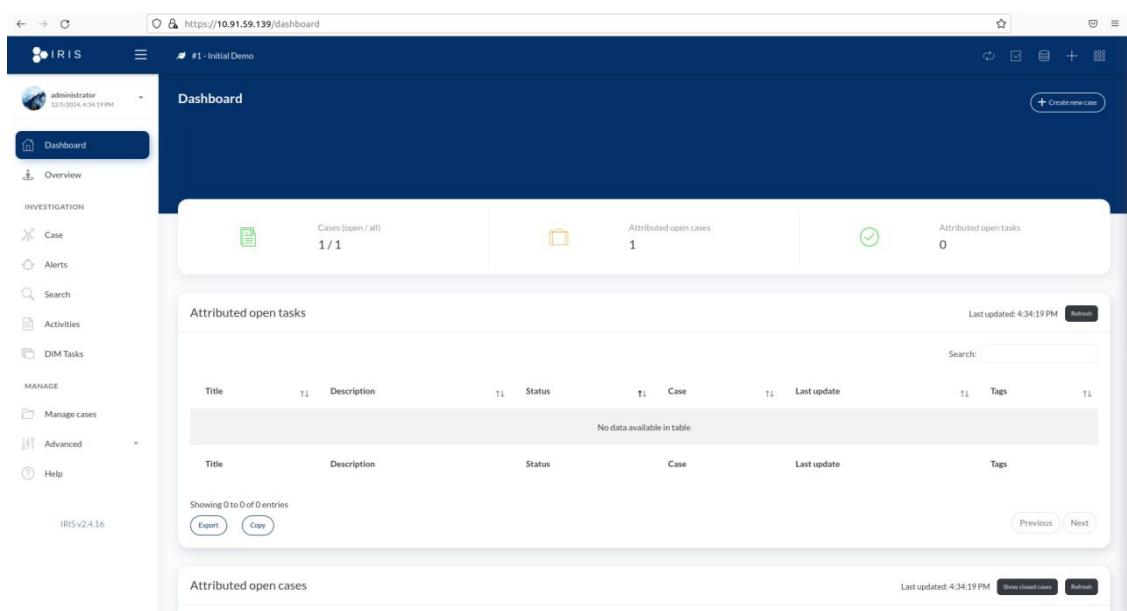
# Run IRIS
docker compose up

```

- After Installing, Access the web interface:



- Login and see the Dashboard of IRIS:

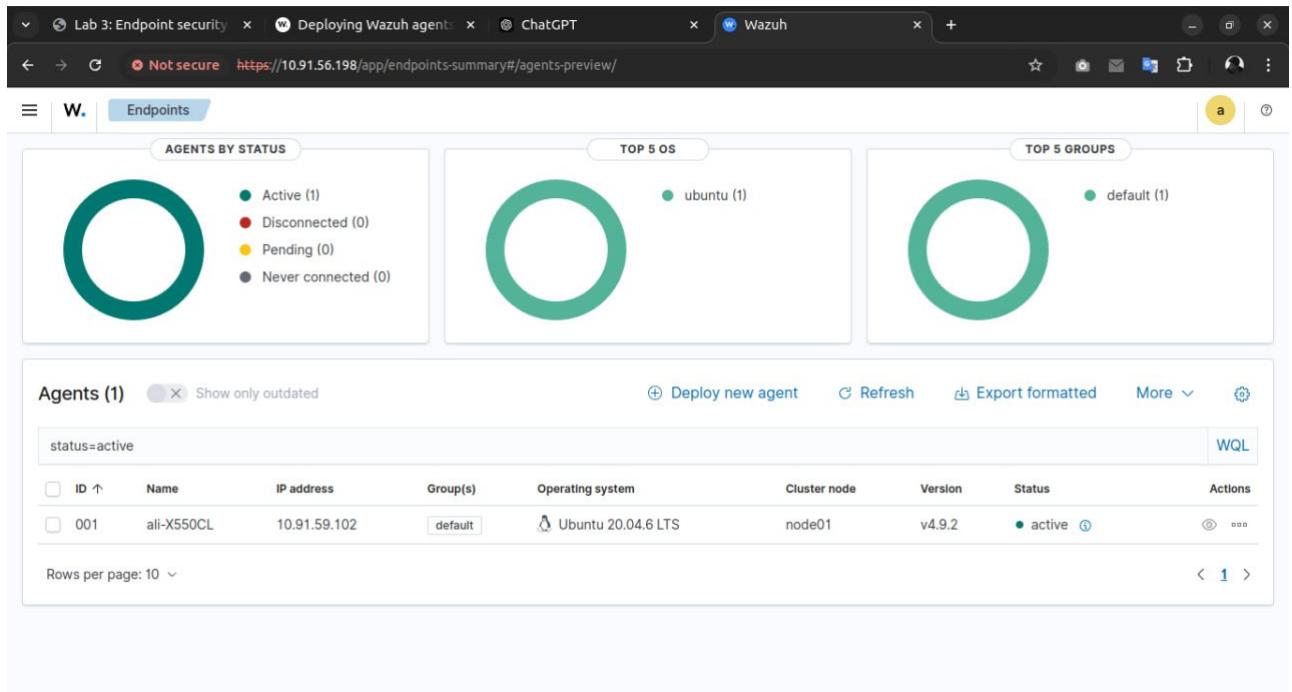


- Network architecture with diagrams showing relationships between components.

## Working Instances of Tools

- **Wazuh:** Configured to collect logs, generate alerts, and display dashboards.

here you can see the connection between the Wazuh Server and Wazuh Agent



- **MISP:** Integrated with Wazuh for threat intelligence enrichment.

- Clone the repository of [IRIS-Wazuh integration](#) and update `verify=False` inside the script `custom-iris.py`.
- Copy the updated script to `custom-iris.py` to the directory `/var/ossec/integrations/custom-iris.py`, Sets permissions for the file so the owner can read, write, and execute, and Changes the ownership of the file to the user root and group wazuh.

```
git clone https://github.com/nateuribe/Wazuh-IRIS-integration.git
cd Wazuh-IRIS-integration/
cp custom-iris.py /var/ossec/integrations/custom-iris.py
chmod 750 /var/ossec/integrations/custom-iris.py
chown root:wazuh /var/ossec/integrations/custom-iris.py
```

```
78 # Send request to IRIS
79 response = requests.post(hook_url, verify=False, data=payload, headers={"Authorization": "Bearer " + api_key, "content-type": "application/json"})
80
```

- Add the following snippet into the `/var/ossec/etc/ossec.conf` config file, Adjust `<hook_url>` and `<api_key>` to your environment, and change `<level>` to the desired threshold for alerts.

```
<!-- ... Rest of config ... -->
<!-- IRIS integration -->
```

```

<integration>
    <name>custom-iris.py</name>
    <hook_url>http://10.91.59.139/alerts/add</hook_url>
    <level>4</level>
    <api_key>d1n0Dx-1qQvef982I8aAKY9R11CflVZrC0zlcNXqARXn5-
IdcOUNHZSiPbgBTStzmDDSGYw6Z8m3_KZ8p1P5Ww</api_key>
    <alert_format>json</alert_format>
</integration>

<! -- ... Rest of config ... -->

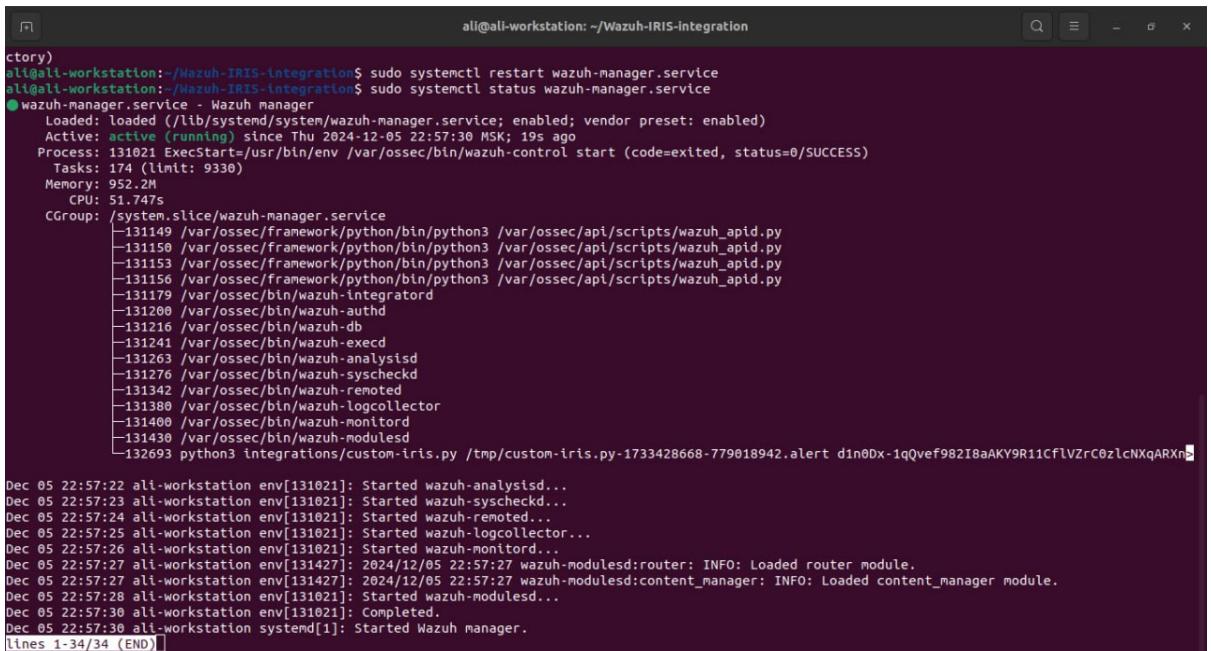
```

```

327 <!-- IRIS integration -->
328 <integration>
329     <name>custom-iris.py</name>
330     <hook_url>https://10.91.59.139/alerts/add</hook_url>
331     <level>4</level>
332     <api_key>d1n0Dx-1qQvef982I8aAKY9R11CflVZrC0zlcNXqARXn5-
IdcOUNHZSiPbgBTStzmDDSGYw6Z8m3_KZ8p1P5Ww</api_key>
333     <alert_format>json</alert_format>
334 </integration>

```

- Restart the wazuh-manager service after making the above settings, and check the status:



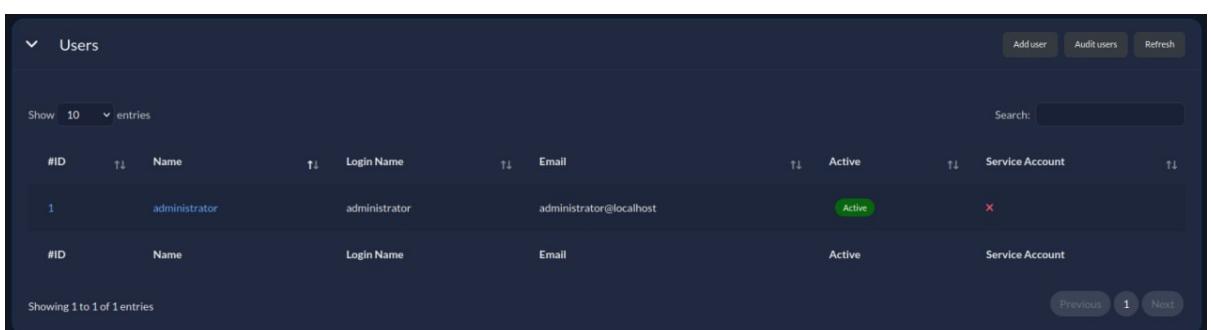
```

alibaba@ali-workstation:~/Wazuh-IRIS-integration$ sudo systemctl restart wazuh-manager.service
alibaba@ali-workstation:~/Wazuh-IRIS-integration$ sudo systemctl status wazuh-manager.service
● wazuh-manager.service - Wazuh manager
   Loaded: loaded (/lib/systemd/system/wazuh-manager.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2024-12-05 22:57:30 MSK; 19s ago
     Process: 131021 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
       Tasks: 174 (limit: 9330)
      Memory: 952.2M
        CPU: 51.747s
       CGroup: /system.slice/wazuh-manager.service
               └─131149 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
                  ├─131150 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
                  ├─131153 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
                  ├─131156 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
                  ├─131179 /var/ossec/bin/wazuh-integratord
                  ├─131200 /var/ossec/bin/wazuh-authd
                  ├─131216 /var/ossec/bin/wazuh-db
                  ├─131241 /var/ossec/bin/wazuh-execd
                  ├─131263 /var/ossec/bin/wazuh-analysisd
                  ├─131276 /var/ossec/bin/wazuh-syscheckd
                  ├─131342 /var/ossec/bin/wazuh-remoted
                  ├─131380 /var/ossec/bin/wazuh-logcollector
                  ├─131400 /var/ossec/bin/wazuh-monitord
                  ├─131430 /var/ossec/bin/wazuh-modulesd
                  └─132693 python3 integrations/custom-iris.py /tmp/custom-iris.py-1733428668-779018942.alert d1n0Dx-1qQvef982I8aAKY9R11CflVZrC0zlcNXqARXn5

Dec 05 22:57:22 ali-workstation env[131021]: Started wazuh-analysisd...
Dec 05 22:57:23 ali-workstation env[131021]: Started wazuh-syscheckd...
Dec 05 22:57:24 ali-workstation env[131021]: Started wazuh-remoted...
Dec 05 22:57:25 ali-workstation env[131021]: Started wazuh-logcollector...
Dec 05 22:57:26 ali-workstation env[131021]: Started wazuh-monitord...
Dec 05 22:57:27 ali-workstation env[131427]: 2024/12/05 22:57:27 wazuh-modulesd:router: INFO: Loaded router module.
Dec 05 22:57:27 ali-workstation env[131427]: 2024/12/05 22:57:27 wazuh-modulesd:content_manager: INFO: Loaded content_manager module.
Dec 05 22:57:28 ali-workstation env[131021]: Started wazuh-modulesd...
Dec 05 22:57:30 ali-workstation systemd[1]: Started Wazuh manager.
lines 1-34/34 (END)

```

- Now everything configured from Wazuh server side and we need to check it on the IRIS side:
- Navigate to the Advanced Settings in Iris, then Access Control Section:



#ID	Name	Login Name	Email	Active	Service Account
1	administrator	administrator	administrator@localhost	Active	

Showing 1 to 1 of 1 entries

- Locate Customers Management:

Edit user

Info Permissions Groups Customers Cases access

Full name  
administrator

Login  
administrator

Email  
administrator@localhost

Password (optional for service accounts)

- Must contain at least 12 chars
- Must contain at least an upper case
- Must contain at least a lower case
- Must contain at least a digit

User ID  
1

User UUID  
c9823b6a-93d5-472d-816c-5f7d151b39aa

User API Key  
d1n0Dx-1qQvef982l8aAKY9R11CflVZrC0zlcNXqARXn5-IdcOUNHZSiPbgBTStzmDDSGYw6Z8m3\_KZ8p1P5Ww

Renew

- Assign Wazuh Server to a Customer:

Edit user

Customers the user belongs to

Show 10 entries

Customer

IrisInitialClient

Showing 1 to 1 of 1 entries

Manage

Manage customers membership

Customers membership \*

IrisInitialClient

Search

Select all

IrisInitialClient

Save

- Showing the alerts of that's coming from wazuh agent:

- **IRIS:** Configured for case management and linked with Wazuh.

## Testing Scenarios

- **Incident 1:** Brute force attack simulation detected by Wazuh, automated IP block as response.
- **Incident 2:** Malware detection with alert escalation to IRIS.

## Test Results

- Logs or screenshots demonstrating successful detection and automated responses.

## 4. Difficulties Faced and New Skills Acquired

### Difficulties Faced

- Challenges in tool integration and resolving API incompatibilities.
- Troubleshooting network configurations and performance issues.

### New Skills Acquired

- Hands-on experience with Wazuh, MISP, and IRIS.
- Skills in integrating SOC components and automating responses.
- Enhanced understanding of incident workflows and response automation.

## 5. Conclusion, Contemplations, and Judgement

### Conclusion

- Evaluation of the SOC setup's effectiveness in detecting and responding to incidents.
- Reflection on the strengths of the open-source tools used.
- Acknowledgment of the limitations and areas for improvement.

## Recommendations

- Suggestions for future work, such as adding more tools or refining workflows.
  - Potential use cases for scaling the solution in real-world scenarios.
- 

## 6. Appendices

### A. Scripts and Configurations

- Links to all scripts and configuration files in the repository:  
[GitHub Repository](#)

### B. Proof of Concept Demonstration Video

- [YouTube Link to Demo Video](#)

### C. Extended Documentation

- Step-by-step installation and configuration details for all tools.
  - Troubleshooting steps for resolving challenges.
- 

## Integration of New Sections with Previous Requirements:

This report structure includes:

1. **Goal and Tasks of the Project:** Describes what the project aims to solve and specifies the division of tasks among members.
2. **Execution Plan/Methodology:** Contains the solution plan, architecture diagram, and methodology for implementing and testing the solution.
3. **Development of Solution/Tests:** Explains the setup and PoC testing, including results and insights from incident simulations.
4. **Difficulties Faced/New Skills:** Reflects on challenges and the knowledge gained during the project.
5. **Conclusion and Judgement:** Wraps up with reflections, evaluations, and suggestions for improvement.
6. **Appendices:** Contains all important links to code, configuration files, and demonstration materials for proof of concept.

Let me know if you need help with any specific section or step in implementation!