DEPLOYED WAZUH EDR AND SYSMON FOR COMPLETE CYBER DEFENSE AT INTERNEE.PK

Project by:

Ahmad Sameer

ahmadsameer0990@gmail.com

🔗 LinkedIn: Ahmad Sameer | GitHub: Ahmadx90

#1 Implement Endpoint Security & Monitoring

Table of Contents

1. Objective

I was assigned to implement an advanced endpoint security and monitoring solution designed to protect critical systems from evolving cyber threats. The objective of this project was to deploy a robust, real-time security framework using Wazuh as an Endpoint Detection and Response (EDR) platform, complemented by Sysmon for detailed event logging. The goal was to strengthen endpoint defenses against malware, unauthorized access, insider threats, and system integrity violations.

To achieve this, I deployed Wazuh Manager on an Ubuntu 20.04 server to provide centralized visibility and threat detection. A Windows 10 endpoint was integrated using the Wazuh Agent, enabling continuous monitoring of file changes, process activities, and network connections. Sysmon was installed and configured to enhance logging capabilities, ensuring granular visibility into endpoint behavior.

Additionally, I configured File Integrity Monitoring (FIM) to detect unauthorized file modifications, created custom alert rules for suspicious activities, and enabled automated responses to react to security events in real time. To further enhance the solution, I integrated MalwareBazaar threat intelligence feeds into Wazuh, enabling proactive detection of known malware indicators.

The overarching objective was to build a comprehensive security ecosystem capable of early threat detection, automated incident response, and improved organizational resilience against cyberattacks.

Lab Setup & IP Details

Lab Topology and IP Configuration

In this lab, I implement Endpoint Security & Monitoring using **Wazuh** as the EDR and **Sysmon** for advanced endpoint logging.
The environment consists of the following systems:

Device	OS	IP Address	Role
Ubuntu	Ubuntu 20.04	*192.168.137.153*	Wazuh Manager / EDR
Windows 10	Windows 10 Pro	*192.168.137.148*	Endpoint (Wazuh Agent)

2. Step 1: Installing Wazuh Manager on Ubuntu 20.04 (EDR Setup)

1.1 Update and Install Dependencies

Running the following commands to update the Ubuntu 20.04 system and install required dependencies:

```
sudo apt update && sudo apt upgrade -y
sudo apt install curl apt-transport-https unzip -y
```

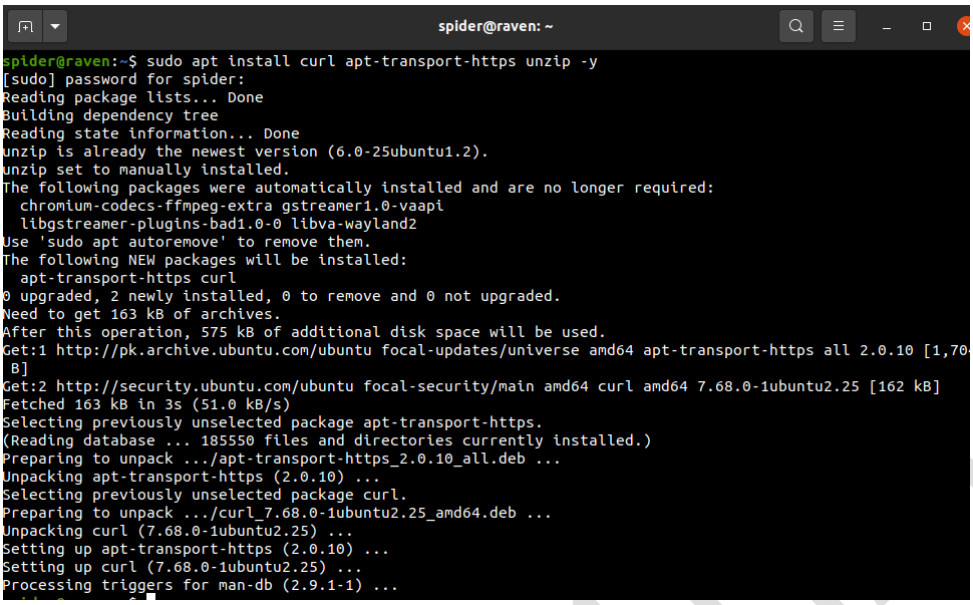


Figure 1 Install Dependencies

1.2 Download and Install Wazuh Manager

I downloaded the official Wazuh installer script and ran it with the `-a` option to deploy the all-in-one stack:

```
curl -sO https://packages.wazuh.com/4.8/wazuh-install.sh
sudo bash wazuh-install.sh -a
```

This installed:

- *Wazuh Manager*
- *Elasticsearch*
- *Kibana*
- *Filebeat*

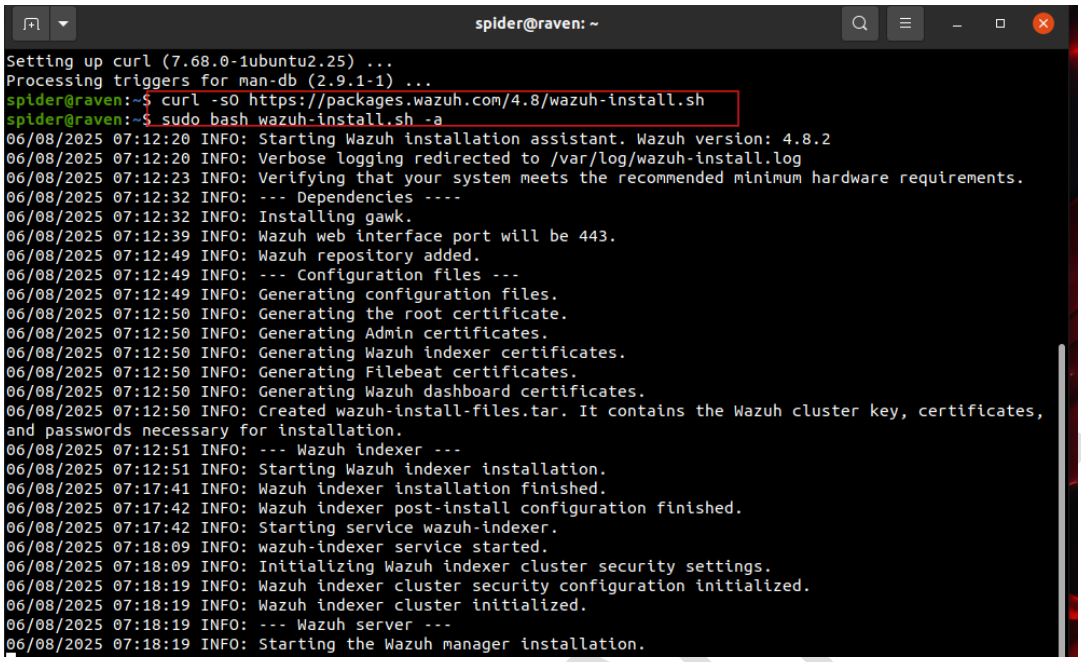


Figure 2 Installation of ELK Stack

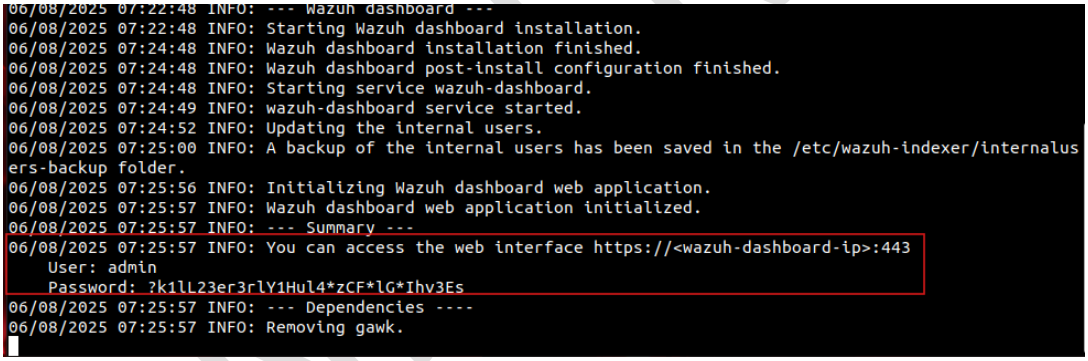


Fig 2.1 Credentials of Wazuh Web Interface

1.3 Verify Wazuh Services and Dashboard

To confirm that all Wazuh components were running properly, I ran:

```bash
sudo systemctl status wazuh-manager
```

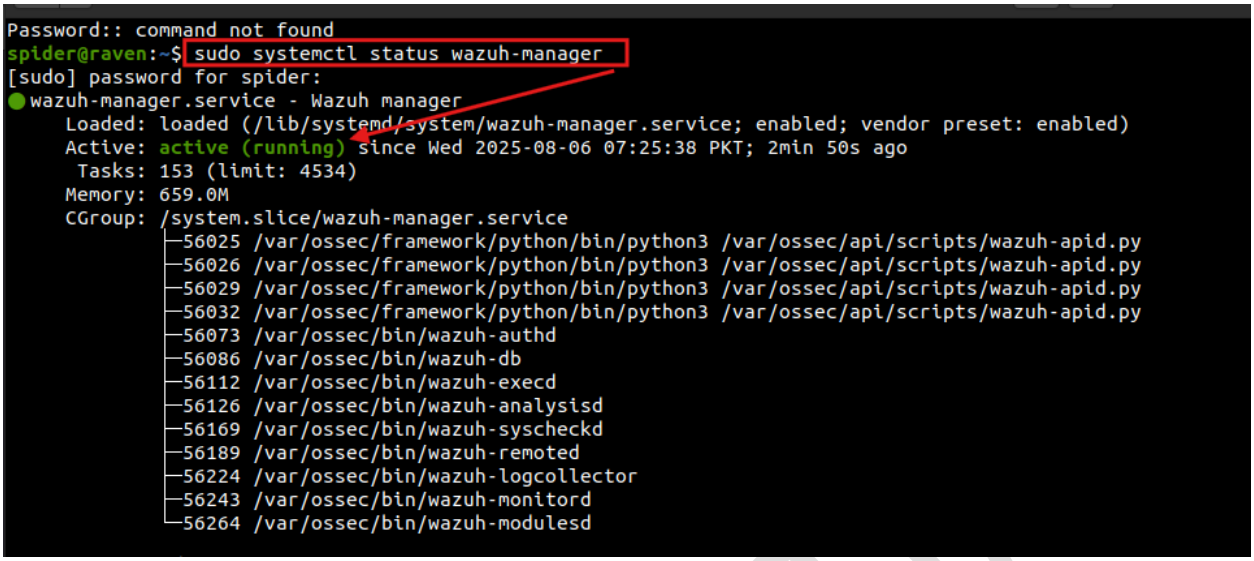


Figure 3 Checking status Wazuh Manager

- Wazuh Indexer installed and running
- Wazuh Dashboard installed and running

Then, I opened the dashboard on my browser:

```
https://192.168.137.153:443
```

Default login:

- *Username: admin*
- *Password: ****

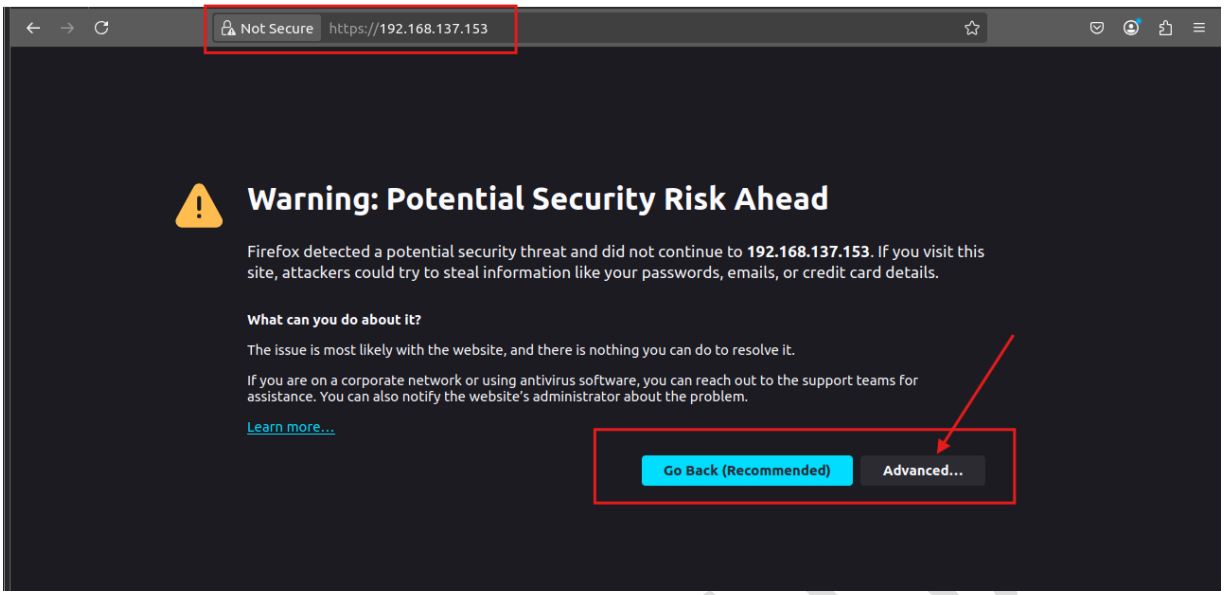


Figure 4 Lunching Wazuh by clicking on Advanced

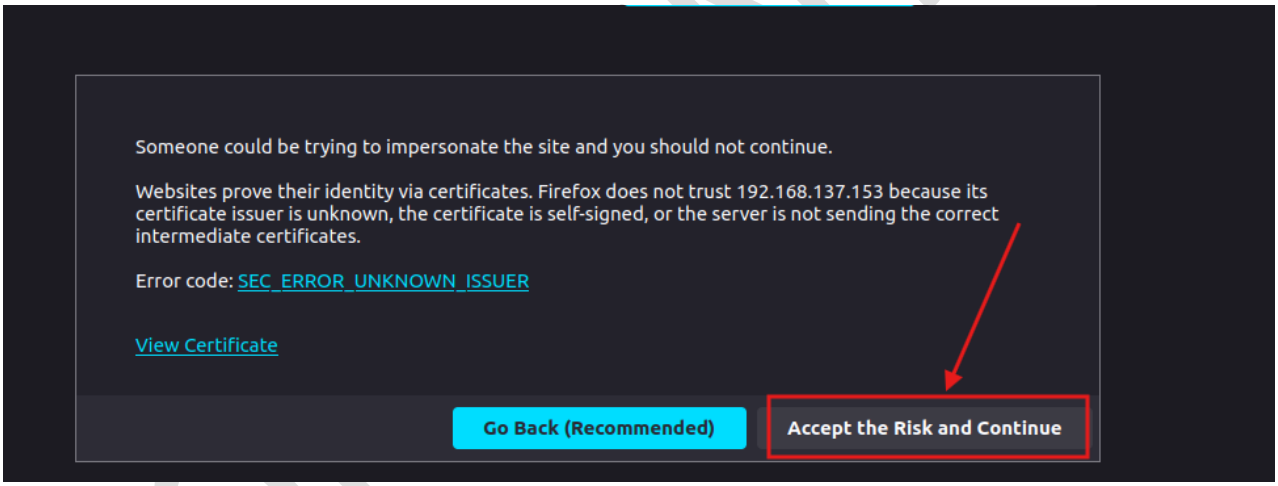


Figure 5 Click on Accept the Risk to Continue

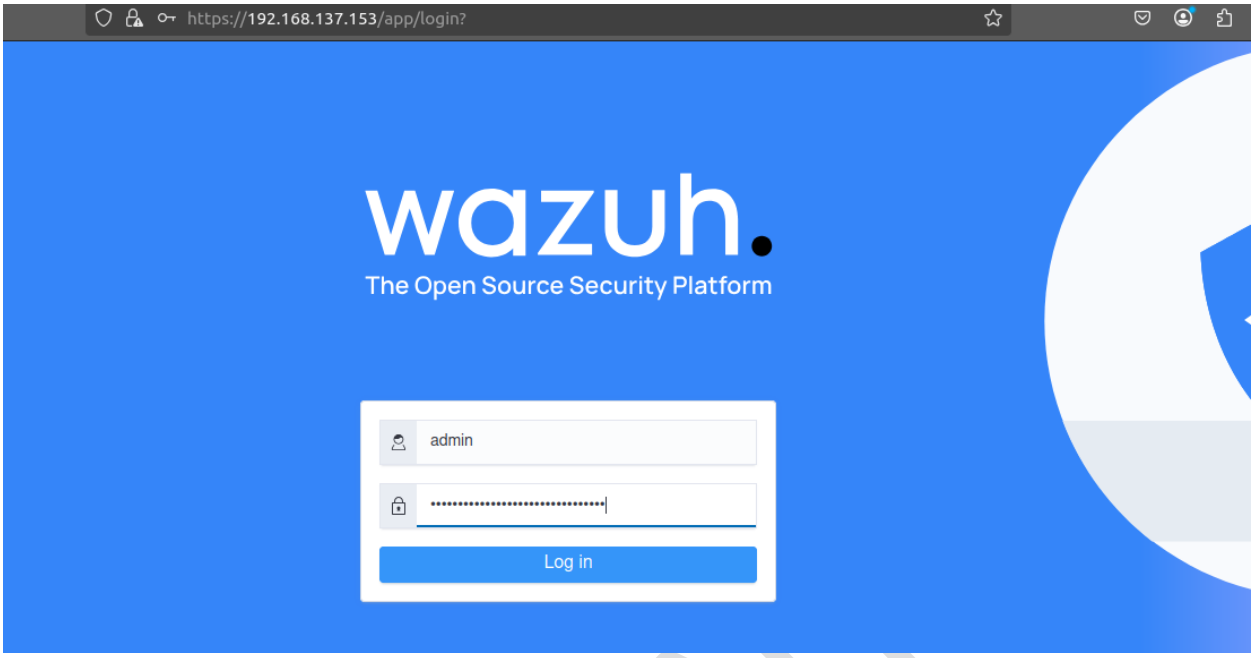


Figure 6 Login to Wazuh using default credentials which I get during Installation of Manager

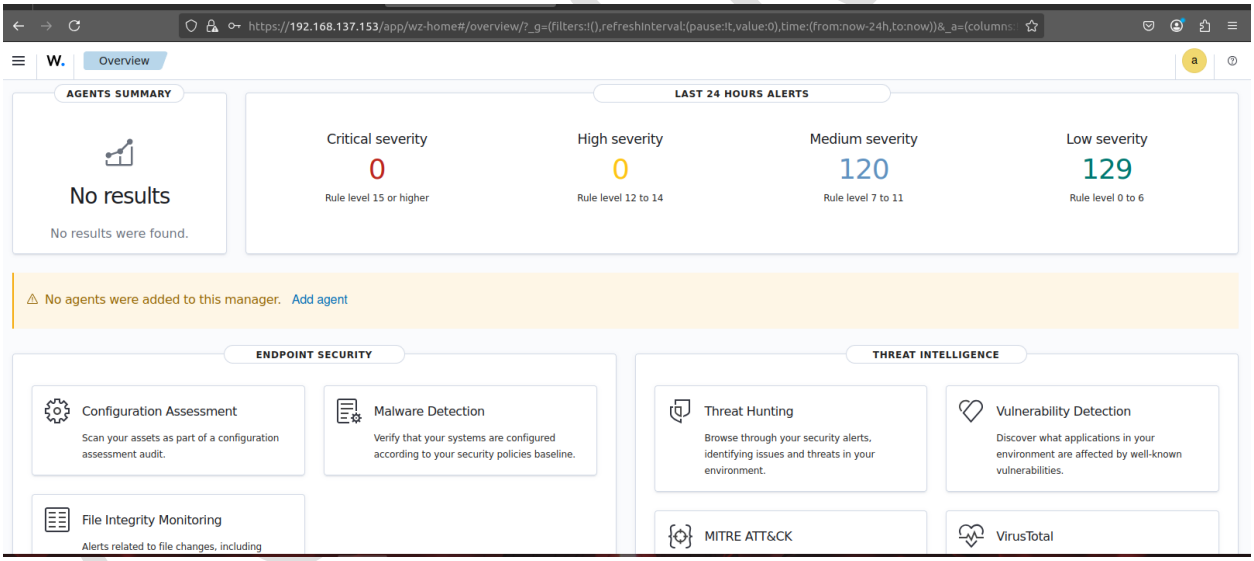


Figure 7 Wazuh Interface

Step 2: Install Wazuh Agent on Windows 10

2.1 Download Wazuh Agent

I opened my Windows 10 machine (192.168.137.148) and downloaded the latest Windows Wazuh agent from the official site:

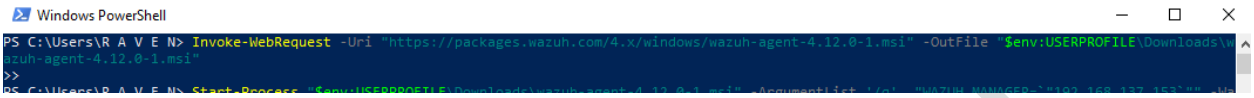


Figure 8 Installing Agent on Window 10 to use it as Endpoint System

After downloading, I double-clicked the `.msi` file to start the installation wizard.

2.2 Configure Agent to Connect to Ubuntu Wazuh Manager

During the installation wizard:

1. I selected **Next** until the **Server Address** prompt appeared.
2. I entered my **Wazuh Manager IP**: `192.168.137.153`
3. I kept the **default port 1514** for communication.
4. I completed the installation with default options.

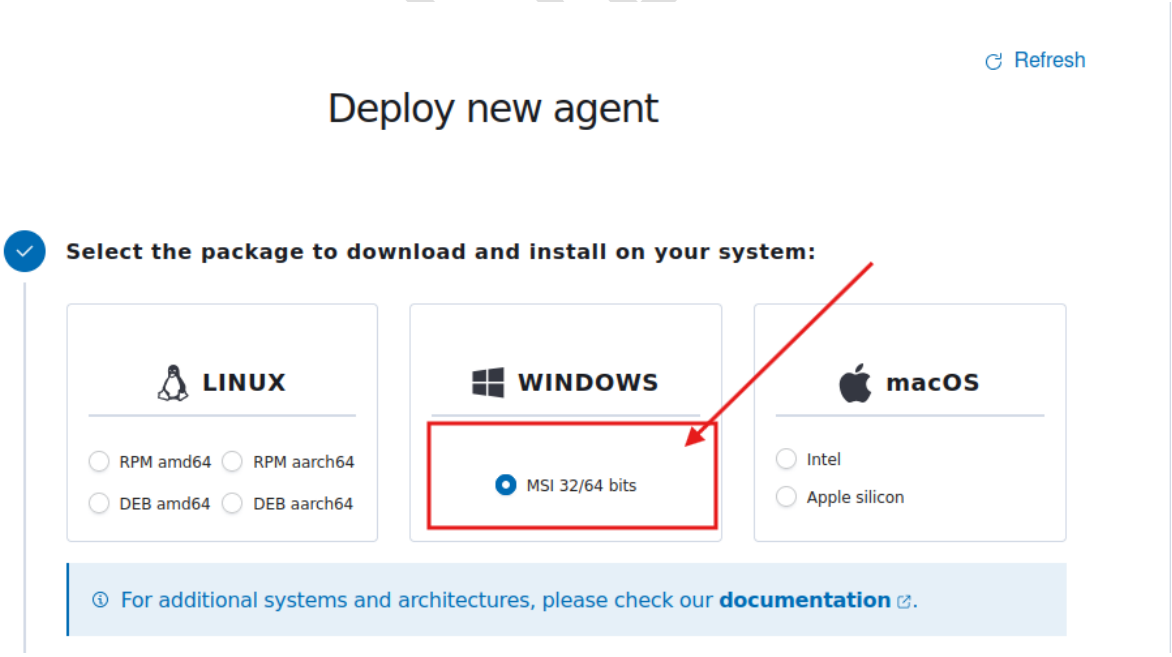


Figure 9 Deploying MSI 32/64 bits window Agent to connect Manager and Agent

Figure 10 Putting Server address which is Host(Ubuntu where Wazuh manager is running)

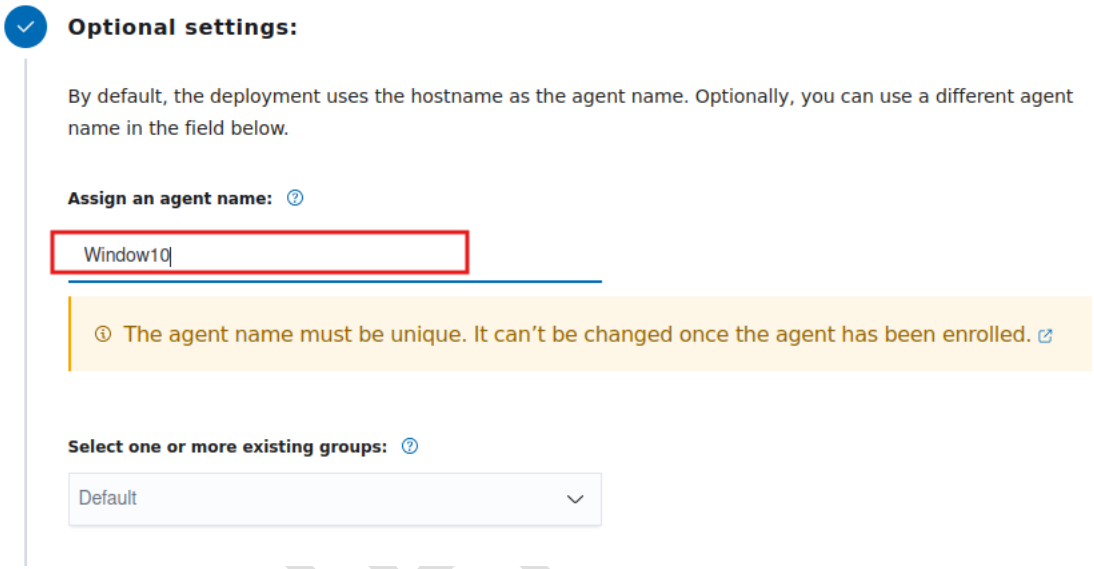


Figure 11 Setting name of Agent "Window 10"

Figure 12 Getting Built in command form Wazuh manager to connect Window Agent

2.3 Start Agent and Verify Connection

After installation, I started the Wazuh agent service:

"net start wazuhsvc"

Figure 13 Copying the command and pasting in Shell of Window to Start the Wazuh service

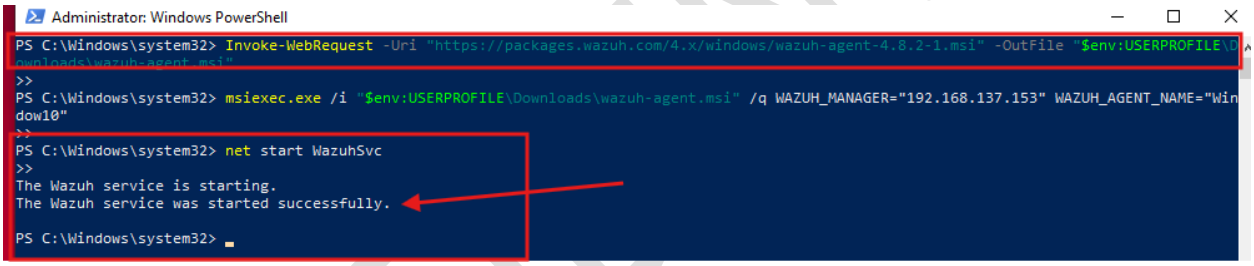


Figure 14 Installation and starting services of Wazuh in Window 10 Administrator PowerShell

Or, I could start it through **Services.msc** by locating **Wazuh Agent** and clicking **Start**.

To verify the agent connection to the manager:

1. **I opened the Wazuh Dashboard on my browser:**

"http://192.168.137.153:443"

2. I navigated to **Wazuh → Agents** and saw the **Windows 10 machine listed** as "***active***".

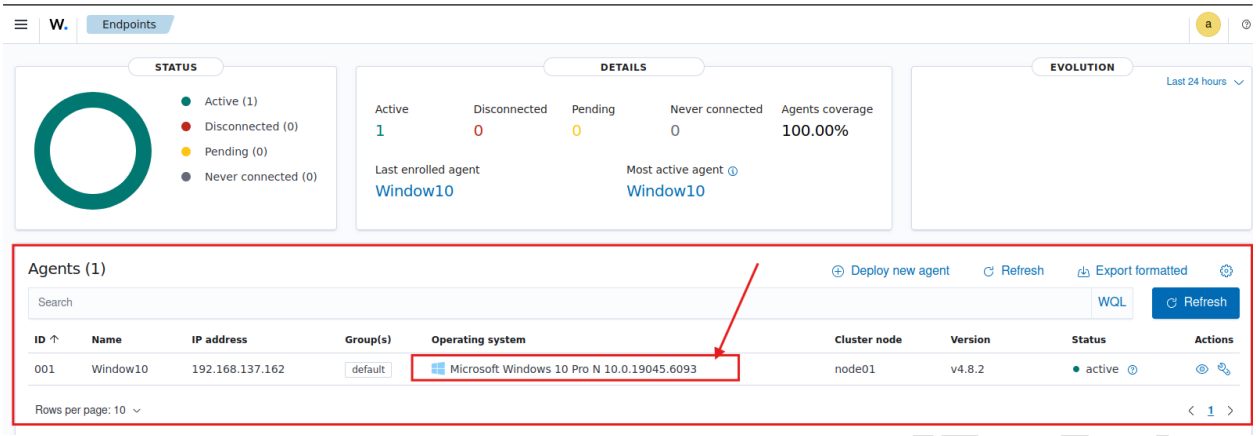


Figure 15 Actie Wazuh Agent and Analyzing the system derails

Step 2 is complete. *My Windows 10 endpoint is now securely connected to the Wazuh Manager.*

Step 3: Install and Configure Sysmon on Windows 10

3.1 Download Sysmon

I started by downloading **Sysmon** from the official Microsoft Sysinternals page:

"https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon"

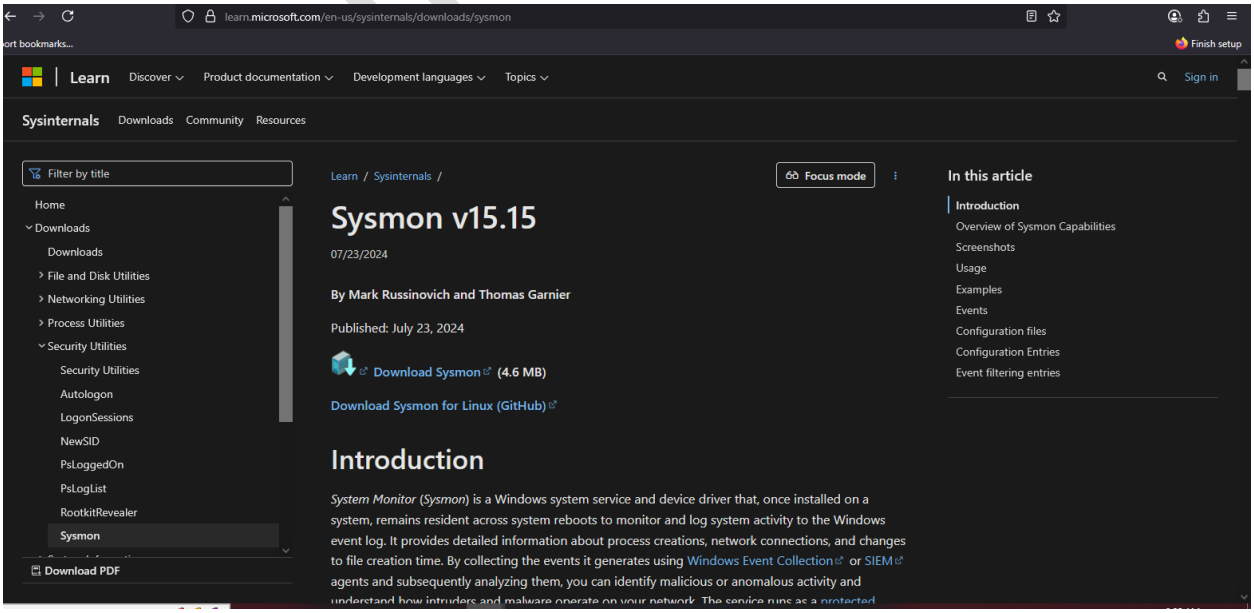


Figure 16 Installing Sysmon V15.15 in Window to monitor Logs and Activity

I extracted the ZIP file to a folder.

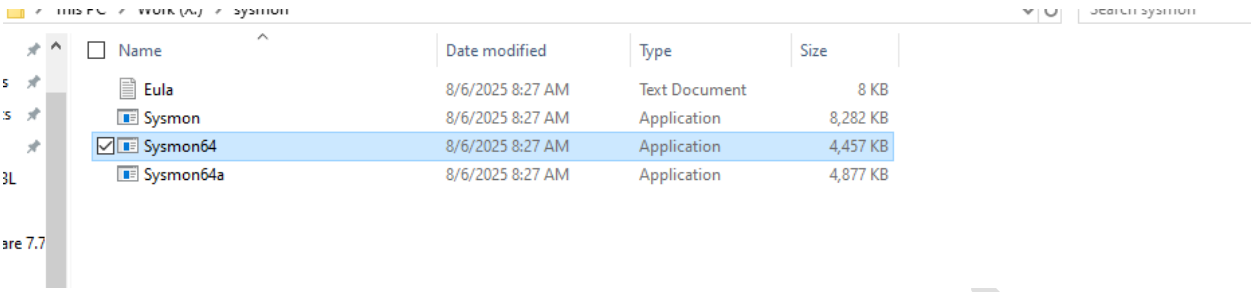


Figure 17 Sysmon flies Extracted

3.2 Download Sysmon Configuration File

To capture high-quality logs, I downloaded the **SwiftOnSecurity Sysmon configuration file**, which is widely used in enterprise environments:

"https://raw.githubusercontent.com/SwiftOnSecurity/sysmon-config/master/sysmonconfig-export.xml"

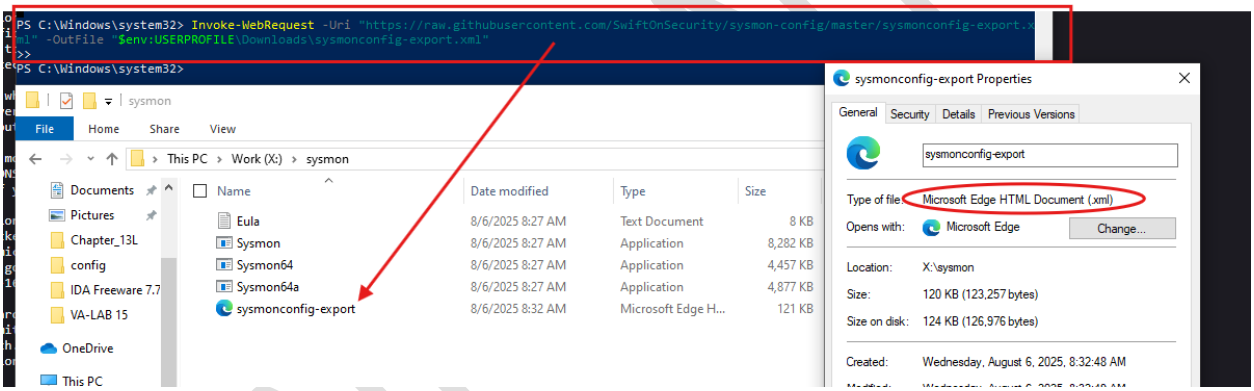


Figure 18 Installing Sysmon Configuration File.xml

I saved it in the same folder as Sysmon binaries for easy access.

3.3 Install Sysmon with Configuration

I opened **PowerShell as Administrator** in the Sysmon folder and installed Sysmon with the configuration file using:

```
powershell

Sysmon64.exe -accepteula -i sysmonconfig-export.xml
```

- **-accepteula**: Automatically accepts the license agreement.
- **-i**: Installs Sysmon as a service with the specified configuration.

After installation, I verified that Sysmon was running as a Windows service.

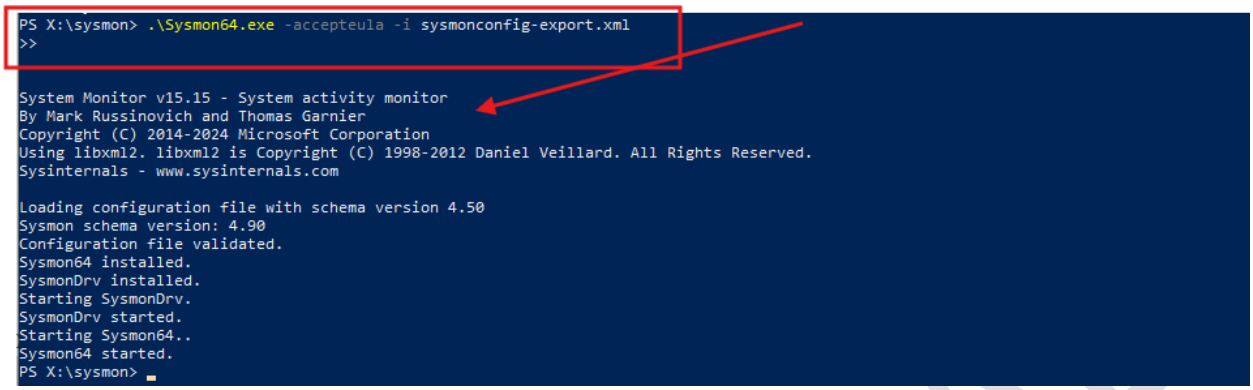


Figure 19 Installing Sysmon .exe and sysconfig-export.xml. After Installation starting Sysmon

3.4 Verify Sysmon Event Logging

To confirm Sysmon was generating logs, I opened **Event Viewer**:

1. Pressed **Win + R**, typed ***eventvwr.msc***, and hit Enter.
2. Navigated to:

   "Applications and Services Logs → Microsoft → Windows → Sysmon → Operational"

3. I verified that Sysmon was logging events such as **Process Creation, Network Connections, and File Changes**.

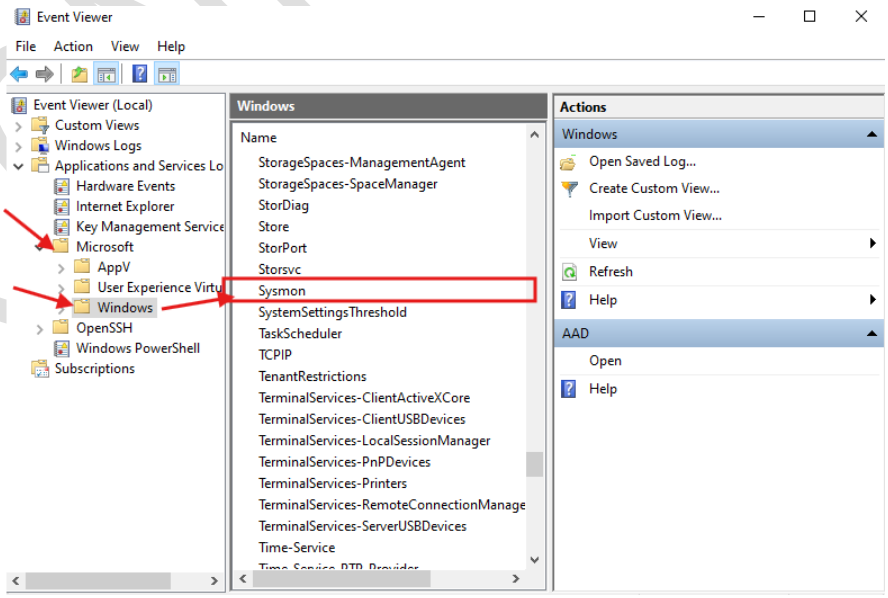


Figure 20 Verifying the service of Sysmon Logging

Step 3 is complete.
*I now have **Sysmon installed and actively monitoring process, network, and file activities** on my Windows 10 endpoint. These logs will enhance threat detection in Wazuh.*

Step 4: Implement File Integrity Monitoring (FIM) in Wazuh

4.1 Purpose of File Integrity Monitoring

I implemented **File Integrity Monitoring (FIM)** to detect **unauthorized modifications, creations, or deletions** of critical files and directories on my Windows 10 endpoint. This enhances security by alerting me to potential malware activity or insider threats.

- ***Wazuh Dashboard → Security Events → File Integrity Monitoring section***

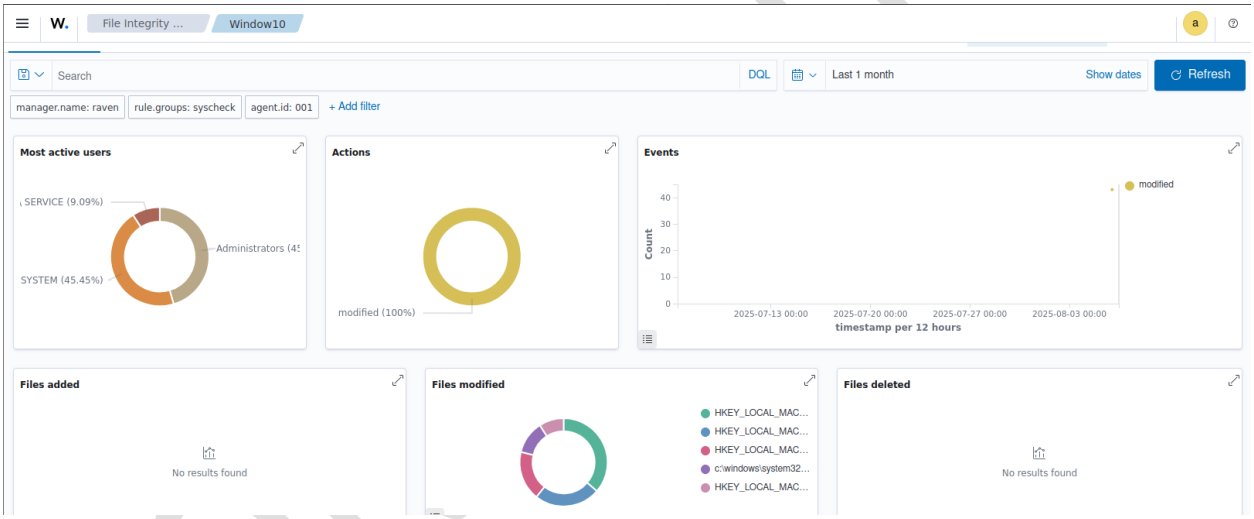


Figure 21 Checking security Events of Window 10. Most Active, Actions and file modifications

4.2 Configure FIM on Wazuh Manager

I configured the Wazuh Manager to monitor specific directories.

1. I logged in to my **Ubuntu 20.04 Wazuh Manager (192.168.137.153)**.
2. I edited the ***ossec.conf*** file using nano:

"sudo nano /var/ossec/etc/ossec.conf"

3. I located the ***<syscheck>*** section and added directories to monitor.
   Example configuration for **Windows system files**:
4. I saved the file and exited nano.

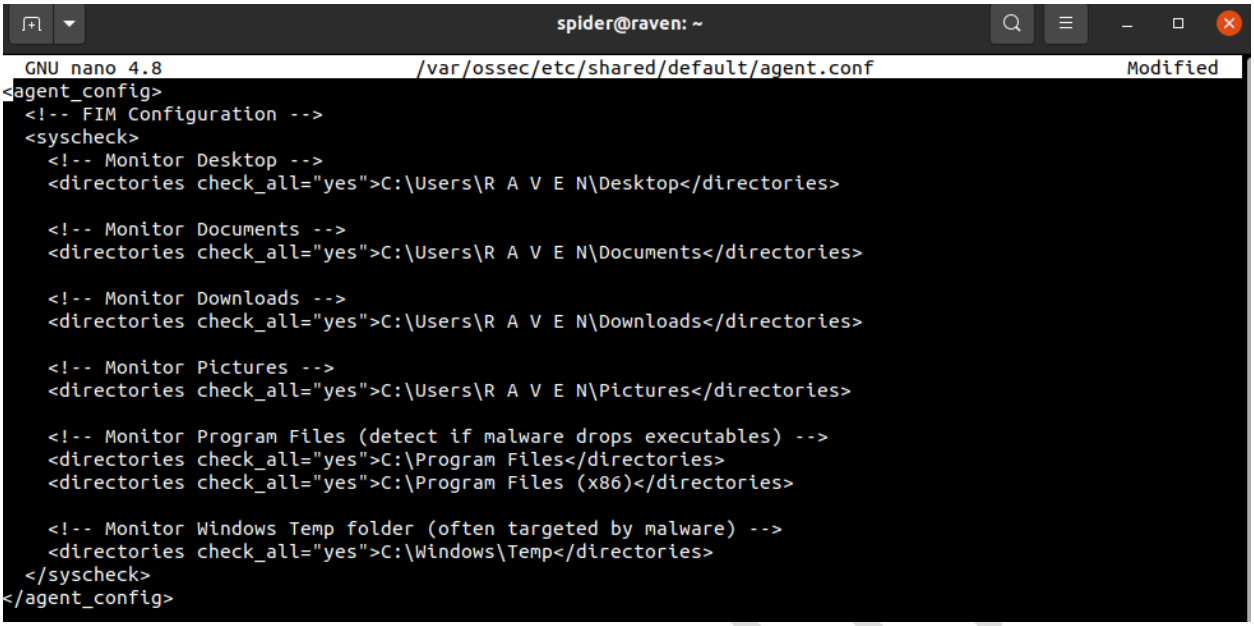


Figure 22 Adding Directories to monitor all activity

4.4 Verify FIM Events on Wazuh Dashboard

I tested the FIM feature by **creating a new file** in one of the monitored directories on my Windows 10 machine:

1) Desktop:

```
powershell

cd "C:\Users\R A V E N\Desktop"
echo "Test Desktop Event" > fim_desktop.txt
```

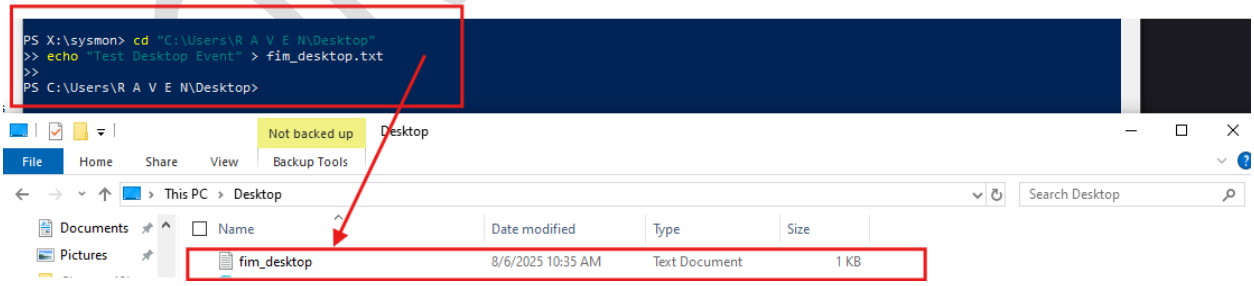


Figure 23 creating random file in window to insure and grab activity in Wazuh

2) Downloads:

```
powershell

cd "C:\Users\R A V E N\Downloads"
echo "Test Download Event" > fim_download.txt
```

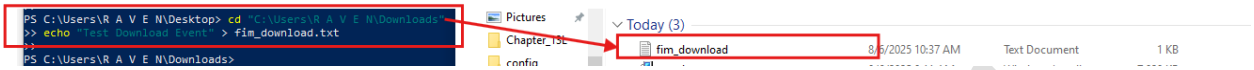


Figure 24 Creating File in downloader folder

Temp folder:

```
powershell

cd C:\Windows\Temp
echo "Test Temp Event" > fim_temp.txt
```

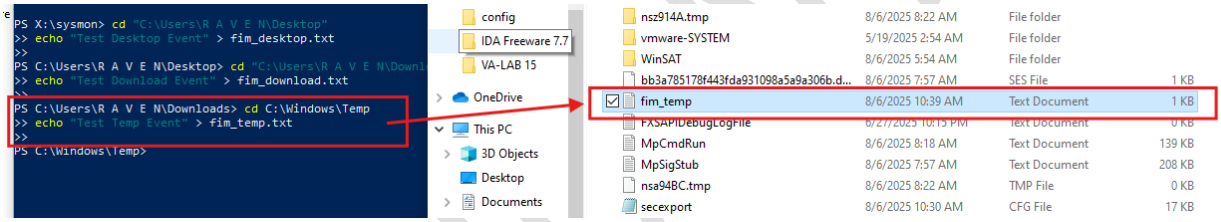


Figure 25 creating fim_temp file in temp directory of window to make sure the activity captured in Events

Then **modify and delete** to generate more events:

```
powershell

echo "Modified Event" >> fim_temp.txt
del fim_temp.txt
```

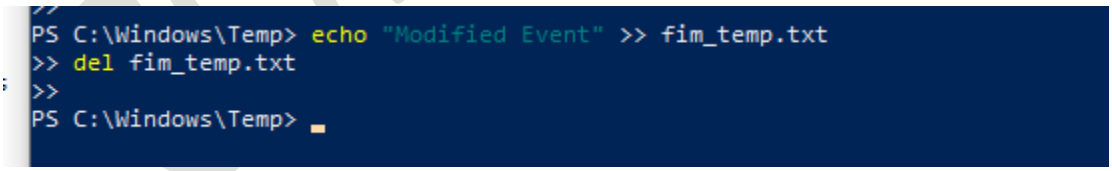


Figure 26 Modifying the file and after that deleting to monitor all Events of Agent Wazuh as an Endpoint Device

Then I logged in to the **Wazuh Dashboard → Security Events → File Integrity Monitoring** and verified that:

- **A new file creation event** appeared.
- The agent **Windows10** generated the alert successfully.

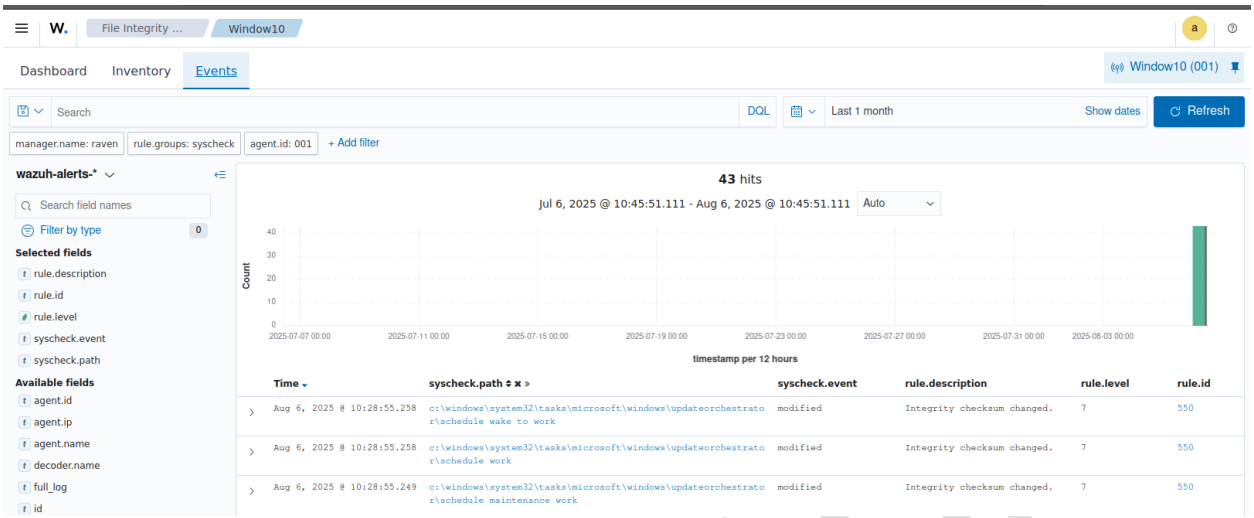


Figure 27 43 hits (shows what type of changes are made on Endpoint)

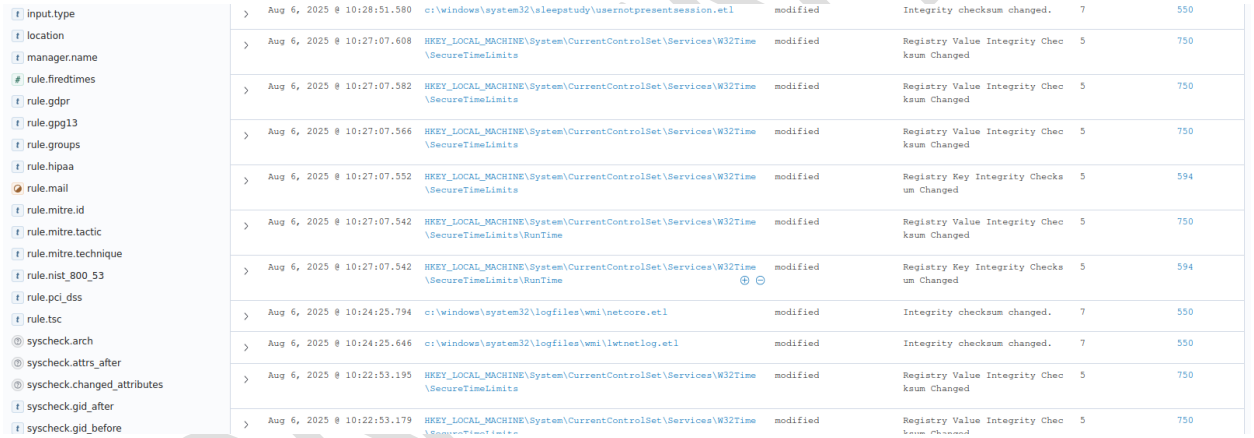


Figure 27.1 Another further details of Figure 27.

- **Top Bar:**
  - Dashboard shows **File Integrity Monitoring → Events Tab**.
  - **Agent:** `Window10 (001)` (your Windows 10 endpoint is successfully connected).
  - **Time Filter:** Last 1 month.

- **Middle Graph:**
  - **43 hits** → This means **43 file integrity events** have been detected in the selected time range.
  - **Bar at the right side of the graph** → Indicates **recent events**, likely triggered today.

Event Table Explanation

The lower section lists **each FIM event**. Columns mean:

- **Time:**
  - Timestamp when the file change was detected.
  - Example: `Aug 6, 2025 @ 10:28:55.258` → Happened today at 10:28 AM.
- **syscheck.path:**
  - Full path of the file that changed.
  - Example:

"c:\windows\system32\tasks\microsoft\windows\updateorchestrator\schedule wake to work"

- This shows that a **Windows scheduled task** file was modified.
- **syscheck.event:**
  - Type of change detected:
    - **added** → New file created.
    - **modified** → File was changed.
    - **deleted** → File was removed.
  - In your snapshot: **modified** → File content or checksum changed.
- **rule.description:**
  - Wazuh's interpretation of the event.
  - Example: **Integrity checksum changed** → Means the file's hash changed (indicating modification).
- **rule.level:**
  - Severity of the event (1–15).
  - Here it's **7**, which is **medium priority**.
  - Changes in ***`C:\Windows\System32`*** are important to watch because malware often targets this directory.
- **rule.id (550):**
  - ID of the rule in Wazuh that triggered the alert.

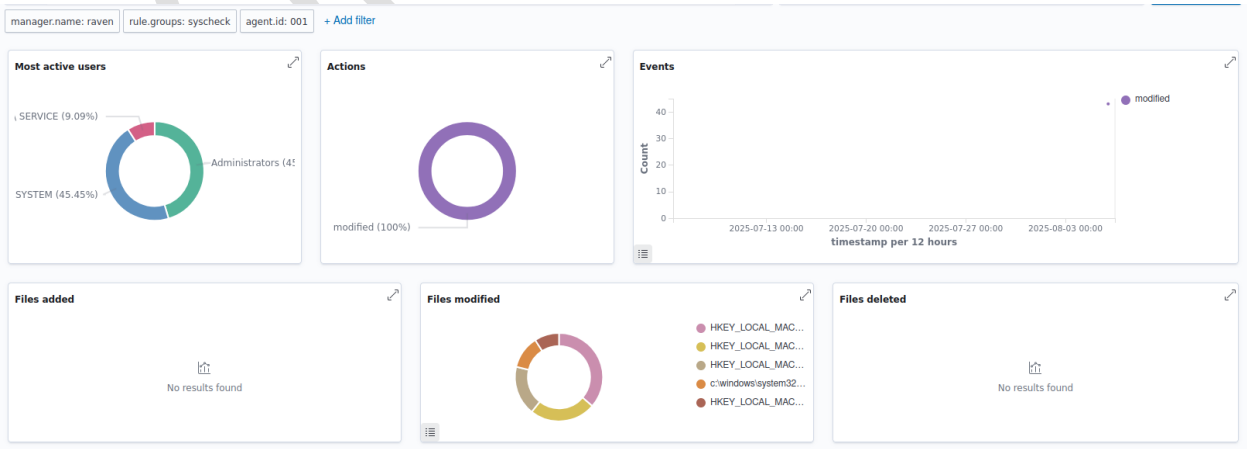


Figure 28 Captured Activity of End Point system

Step 4 is complete.
*I now have **real-time file integrity monitoring** active, and any unauthorized change will trigger an alert in Wazuh.*

Step 5: Configure Security Alerts and Automated Responses in Wazuh

5.1 Objective of Alerts and Automation

I configured **alerts and automated responses** in Wazuh to:

- Detect **suspicious behavior** like failed logins, file tampering, or malware indicators.
- Automatically **trigger notifications** or **run responses** when threats are detected.

This ensures **real-time visibility** and **quick reaction** to incidents.

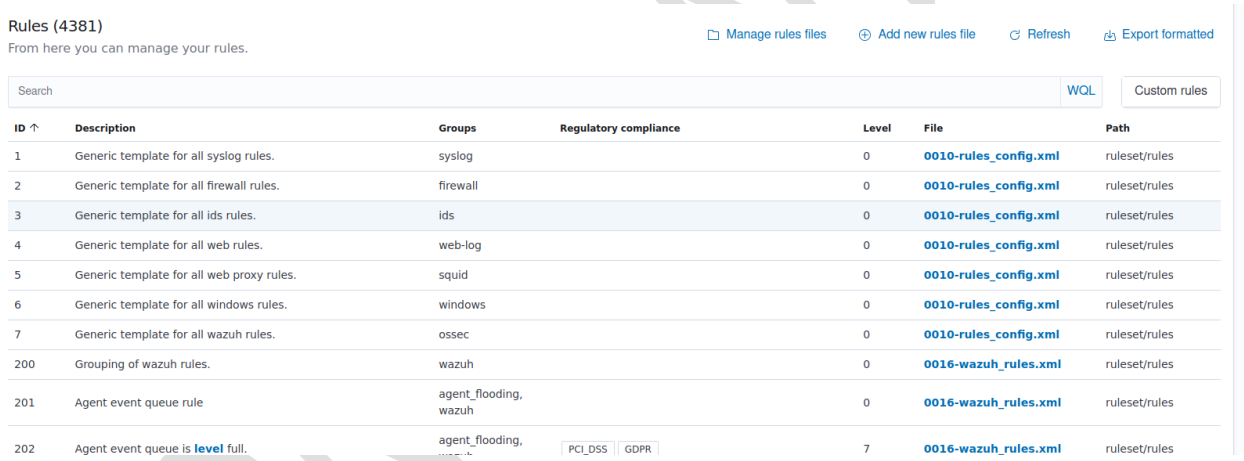


Figure 29 Rules to Alert which shows real-time visibility and quick reaction to incidents

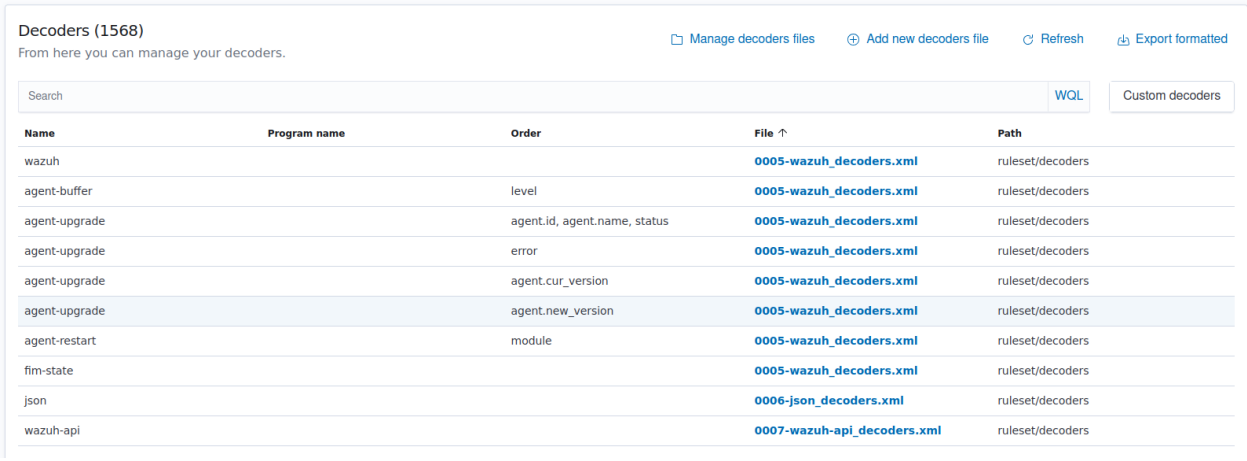


Figure 29.1 Decoders Alerts

5.2 Review Default Wazuh Rules

Wazuh comes with **pre-configured rules** that generate alerts for:

- **File Integrity Monitoring events (FIM)**
- **System log anomalies**
- **Unauthorized access attempts**
- **Malware or suspicious behavior detected via Sysmon logs**

To view these rules, I navigated to:

"cd /var/ossec/rules

ls"

- Default rules:

```
 GNU nano 4.8                    /var/ossec/etc/rules/local_rules.xml
<!-- Local rules -->

<!-- Modify it at your will. -->
<!-- Copyright (C) 2015, Wazuh Inc. -->

<!-- Example -->
<group name="local,syslog,sshd,">

  <!--
  Dec 10 01:02:02 host sshd[1234]: Failed none for root from 1.1.1.1 port 1066 ssh2
  -->
  <rule id="100001" level="5">
    <if_sid>5716</if_sid>
    <srcip>1.1.1.1</srcip>
    <description>sshd: authentication failed from IP 1.1.1.1.</description>
    <group>authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5,</group>
  </rule>

</group>
```

Figure 30 Pre define defaults rules

FIM Rules:

```
root@raven: ~
 GNU nano 4.8                    /var/ossec/etc/rules/local_rules.xml
<!-- Local custom rules -->
<group name="local,syscheck,">

  <!-- Trigger when a file is added -->
  <rule id="100100" level="7">
    <if_group>syscheck</if_group>
    <field name="syscheck.event">added</field>
    <description>File ADDED: $(file) in monitored directory</description>
  </rule>

  <!-- Trigger when a file is deleted -->
  <rule id="100101" level="7">
    <if_group>syscheck</if_group>
    <field name="syscheck.event">deleted</field>
    <description>File DELETED: $(file) from monitored directory</description>
  </rule>

  <!-- Trigger when a file is modified -->
  <rule id="100102" level="7">
    <if_group>syscheck</if_group>
    <field name="syscheck.event">modified</field>
    <description>File MODIFIED: $(file) in monitored directory</description>
  </rule>

</group>
```

Figure 31 Writing new FIM rules to ADD, Delete and Modify file

5.3 Configure Email or Slack Alerts (Optional)

To **receive notifications**, I can configure:

1. **Email Alerts** – by editing the Wazuh configuration:

"sudo nano /var/ossec/etc/ossec.conf"

I ensured **<global>** section includes my SMTP settings:

```
 GNU nano 4.8                          /var/ossec/etc/ossec.conf
->

ossec_config>
 <global>
   <jsonout_output>yes</jsonout_output>
   <alerts_log>yes</alerts_log>
   <logall>no</logall>
   <logall_json>no</logall_json>
   <email_notification>no</email_notification>
   <smtp_server>smtp.example.wazuh.com</smtp_server>
   <email_from>wazuh@example.wazuh.com</email_from>
   <email_to>recipient@example.wazuh.com</email_to>
   <email_maxperhour>12</email_maxperhour>
   <email_log_source>alerts.log</email_log_source>
   <agents_disconnection_time>10m</agents_disconnection_time>
   <agents_disconnection_alert_time>0</agents_disconnection_alert_time>
   <update_check>yes</update_check>
 </global>

 <alerts>
   <log_alert_level>3</log_alert_level>
   <email_alert_level>12</email_alert_level>
 </alerts>

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

Figure 32 SMTP Rules setting for Emails

5.4 Enable Active Response

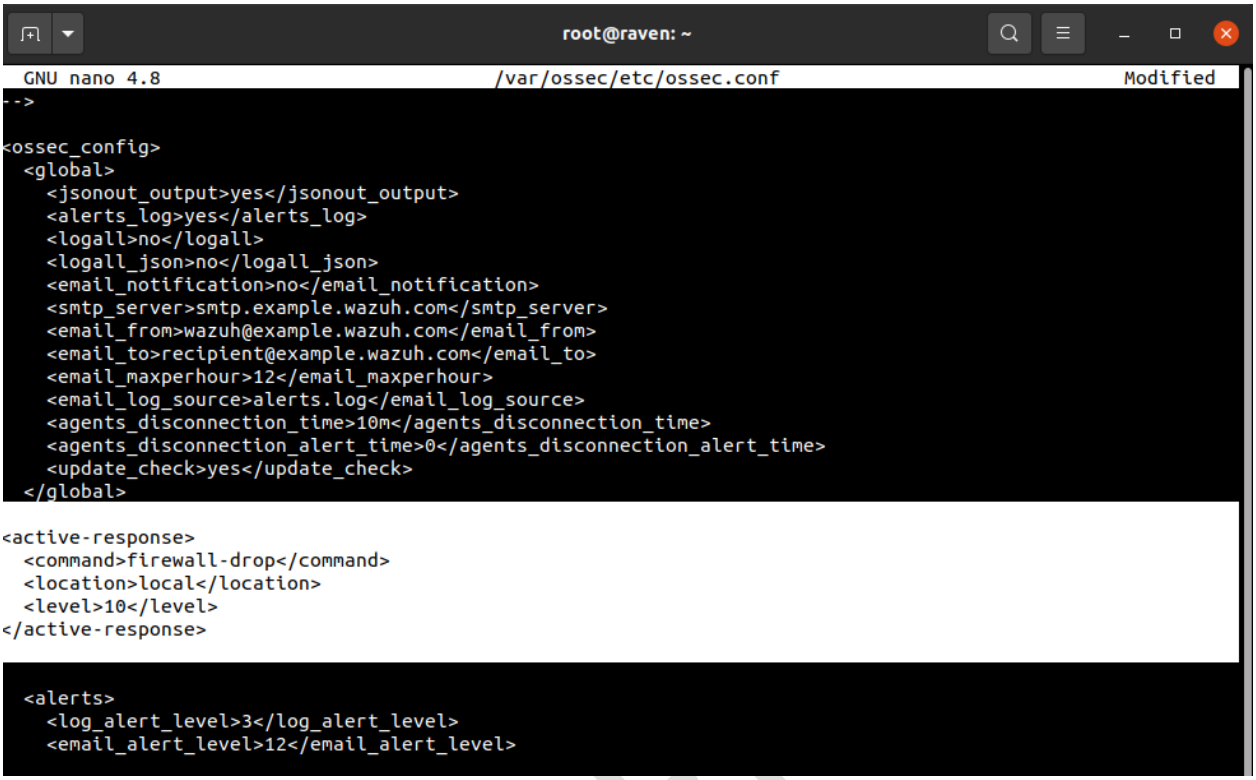
I enabled **Active Responses** to **automatically block malicious behavior**, such as IP addresses performing brute force attacks.

Example: Block repeated SSH failures on Linux or RDP on Windows:

1. Edit active-response configuration:

"sudo nano /var/ossec/etc/ossec.conf"

- **Add:**




```
<ossec_config>
  <global>
    <jsonout_output>yes</jsonout_output>
    <alerts_log>yes</alerts_log>
    <logall>no</logall>
    <logall_json>no</logall_json>
    <email_notification>no</email_notification>
    <smtp_server>smtp.example.wazuh.com</smtp_server>
    <email_from>wazuh@example.wazuh.com</email_from>
    <email_to>recipient@example.wazuh.com</email_to>
    <email_maxperhour>12</email_maxperhour>
    <email_log_source>alerts.log</email_log_source>
    <agents_disconnection_time>10m</agents_disconnection_time>
    <agents_disconnection_alert_time>0</agents_disconnection_alert_time>
    <update_check>yes</update_check>
  </global>

<active-response>
  <command>firewall-drop</command>
  <location>local</location>
  <level>10</level>
</active-response>

  <alerts>
    <log_alert_level>3</log_alert_level>
    <email_alert_level>12</email_alert_level>
```

Figure 33 Updating Script by adding Firewall level 10 active response

Restart Wazuh Manager:

"sudo systemctl restart wazuh-manager"

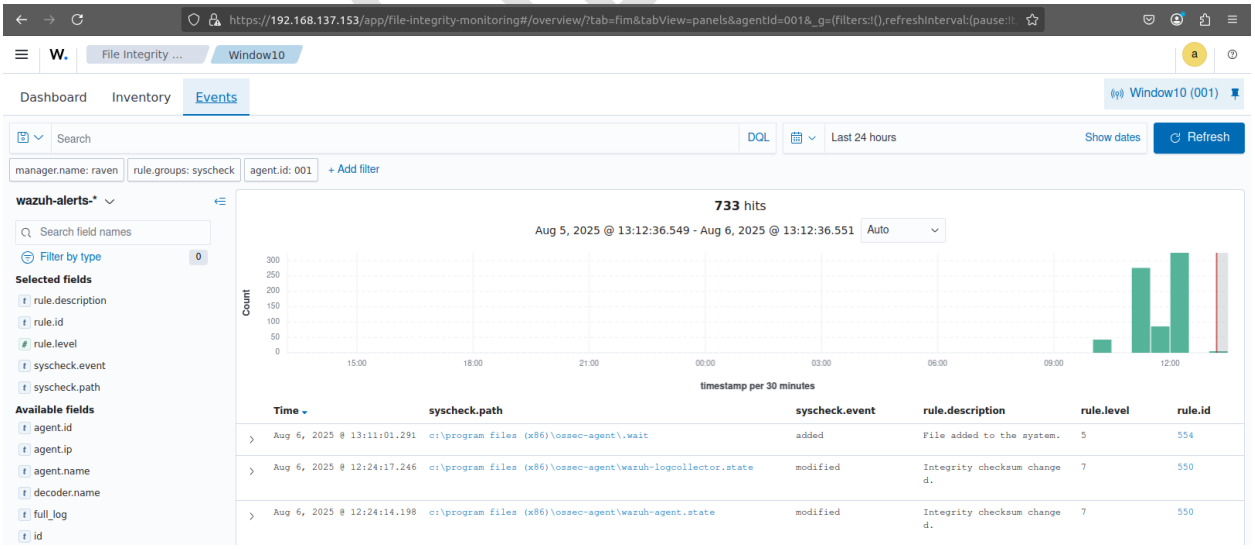


Figure 34 733 Hits occurs after changing the rules and creating files to Analyze Event.

5.5 Validate Alerts

To test my alerts, I performed a **failed login attempt** on Windows 10:

1. I deliberately entered the wrong password 3 times.
2. Wazuh dashboard showed a **security event alert** for multiple failed logins.
3. If active response was enabled, the IP could be **blocked automatically**.
4. **Step 5 is complete.**
   I now have **real-time alerts and optional automated responses** configured to monitor and react to suspicious activity efficiently.

Step 6: Integrate Threat Intelligence with MalwareBazaar in Wazuh

6.1 Objective of Threat Intelligence Integration

I integrated **MalwareBazaar threat intelligence feeds** into Wazuh to:

- Detect **known malware** using file hashes or indicators of compromise (IOCs).
- Enhance Wazuh alerts with **external intelligence**, improving incident response.

6.2 Download MalwareBazaar Threat Feed

I used **Ubuntu 20.04 Wazuh Manager (192.168.137.153)** to fetch the **daily hash feed** from MalwareBazaar:

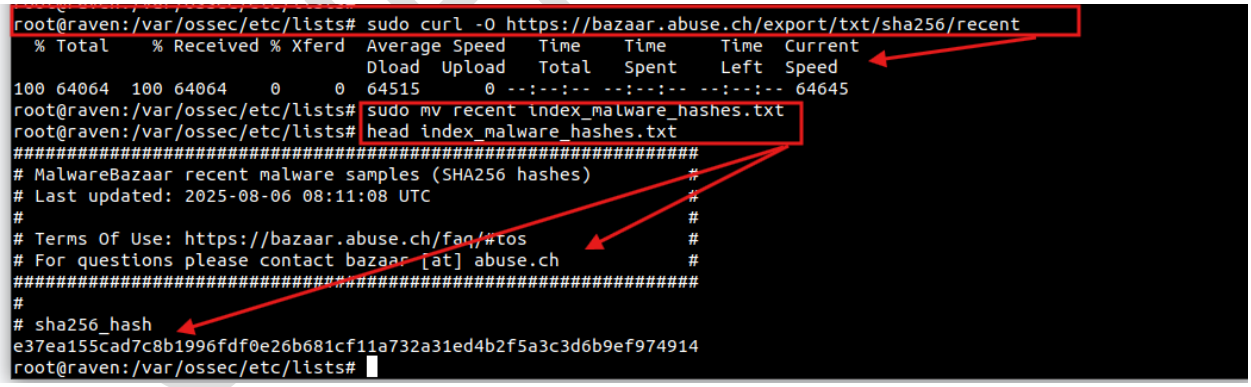


Figure 35

- This creates a **list of malicious hashes** that Wazuh can use for detection.

6.3 Configure Wazuh to Use Threat Feed

I linked the downloaded feed with Wazuh's **CDB list**:

1. Edit ***ossec.conf***:

"sudo nano /var/ossec/etc/ossec.conf"

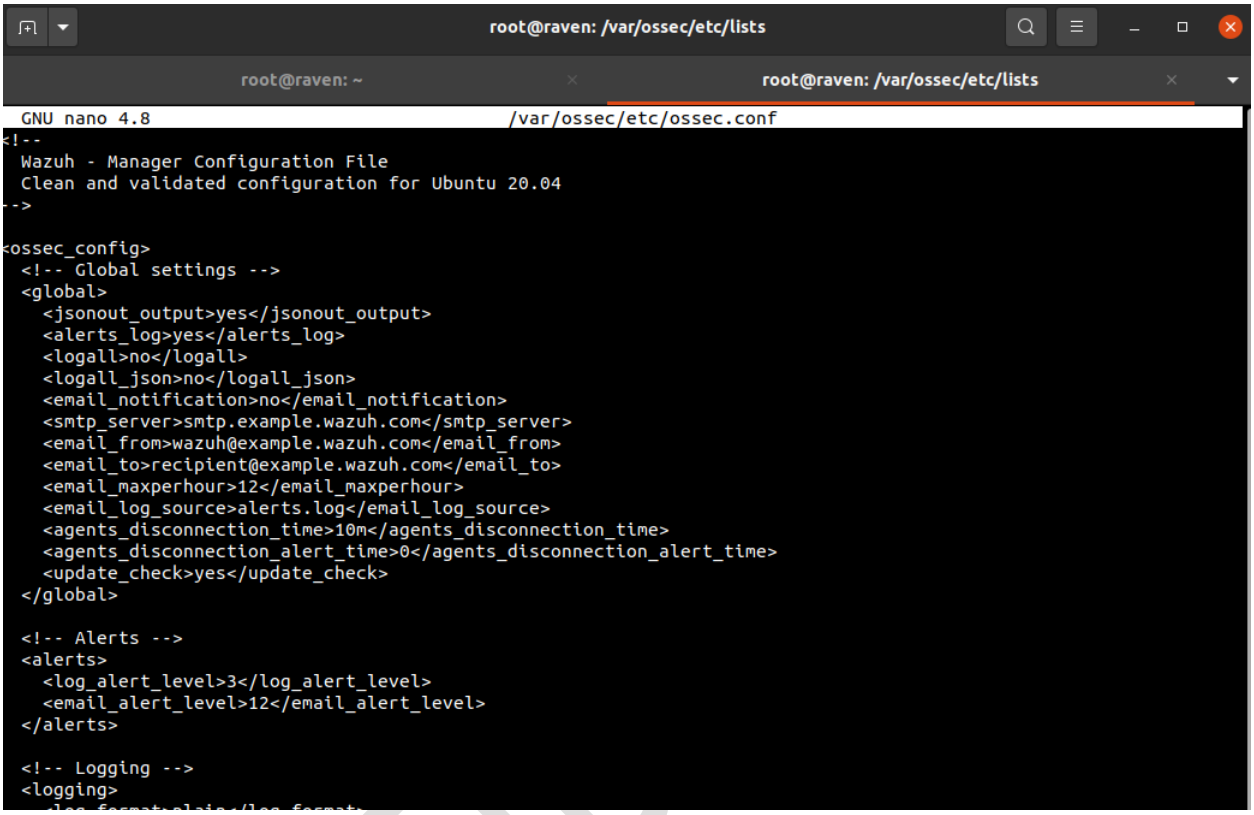


Figure 36 ossec.conf which contains the list/script of Alerts, Logging SMTP etc

2. Add the new list under the ***<ruleset>*** section:

"<ruleset>
<list>lists/index_malware_hashes.txt</list>
</ruleset>"

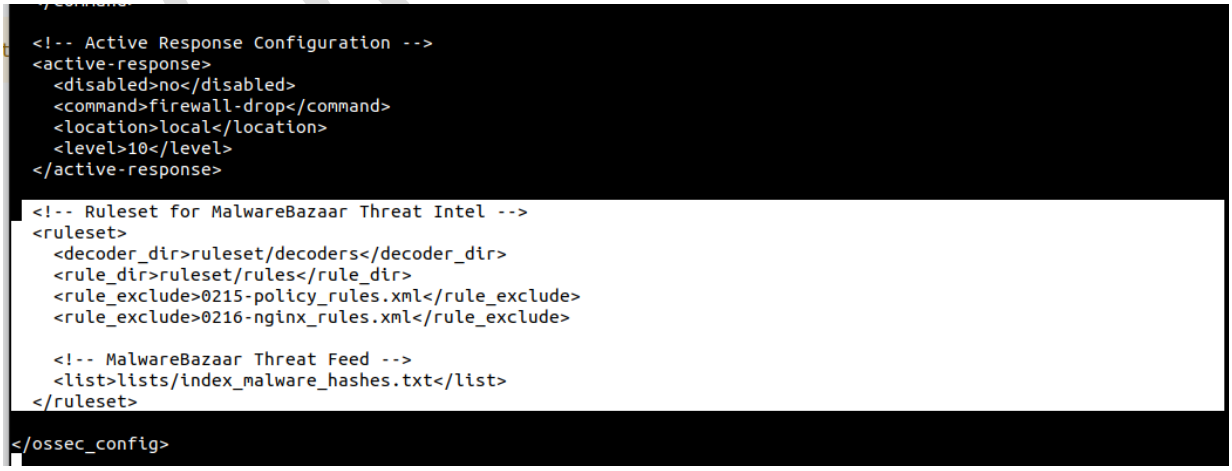


Figure 37 Adding and setting new Rules of Malware Threat feed

6.4 Create a Detection Rule for Malware Hashes

To trigger alerts when a **file hash matches MalwareBazaar IOCs**, I created a **custom rule**:

1. **Create a new rule file:**

"sudo nano /var/ossec/etc/rules/local_rules.xml"

2. **Add the rule:**

```
  <!-- Trigger when a file is modified -->
  <rule id="100102" level="7">
    <if_group>syscheck</if_group>
    <field name="syscheck.event">modified</field>
    <description>File MODIFIED: $(file) in monitored directory</description>
  </rule>

</group>

<!-- MalwareBazaar Threat Intelligence Rule -->
<group name="malwarebazaar,syscheck,">
  <rule id="100200" level="12">
    <if_sid>550</if_sid> <!-- Syscheck file integrity event -->
    <list field="hash_sha256" lookup="match">lists/index_malware_hashes.txt</list>
    <description>Malware file detected using MalwareBazaar Threat Intelligence</description>
    <mitre>
      <id>T1059</id> <!-- Example MITRE technique -->
    </mitre>
  </rule>
</group>
```

Figure 38 Adding Levels to check systems integrity's

3. **Save the file and restart Wazuh:**

"sudo systemctl restart wazuh-manager"

6.5 Test Threat Intelligence Detection

I verified the integration by creating a **dummy file** with a hash from MalwareBazaar's feed:

```
PS C:\Users\R A V E N\Desktop> New-Item -ItemType Directory -Path C:\Temp
>>


    Directory: C:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         8/6/2025   1:57 PM                Temp


PS C:\Users\R A V E N\Desktop> echo "malware_test" > C:\Temp\malwaretest.txt
>>
PS C:\Users\R A V E N\Desktop> Get-FileHash C:\Temp\malwaretest.txt -Algorithm SHA256
>>

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA256          A6CFC960757A08649DAF4BB7DECCAF2BFC64BF0845D0C7BB05EAFBBCCC23C6B2       C:\Temp\malwaretest.txt
```

Figure 39 Creating Temp MalwareBazaar file "malware_test"

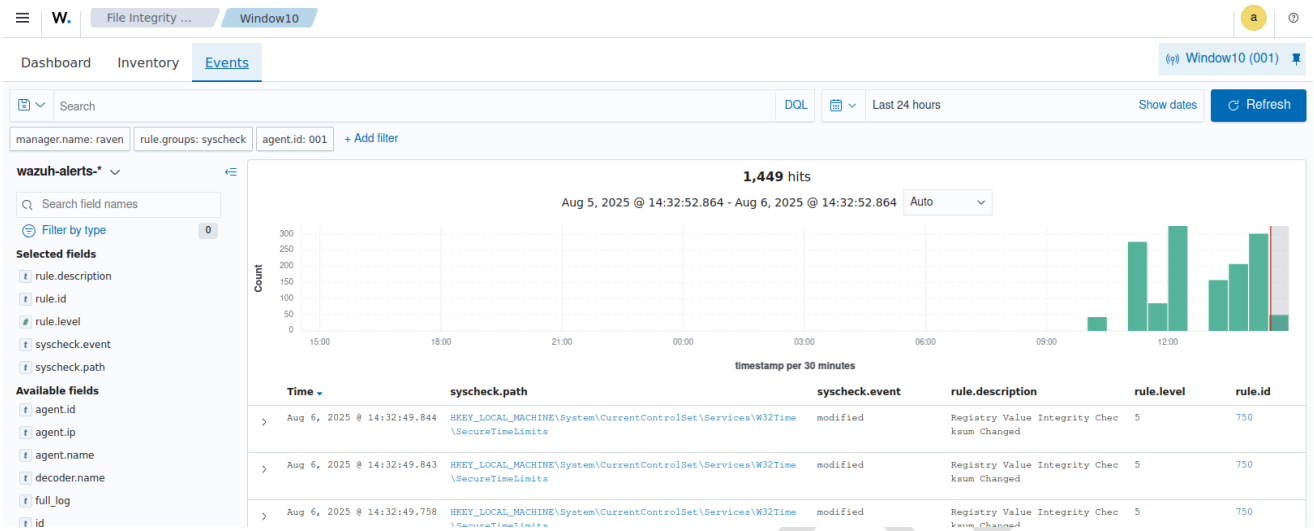


Figure 40 Analyzing the last stage of Hit Events after Performing ALL 1-6 required and assigned steps

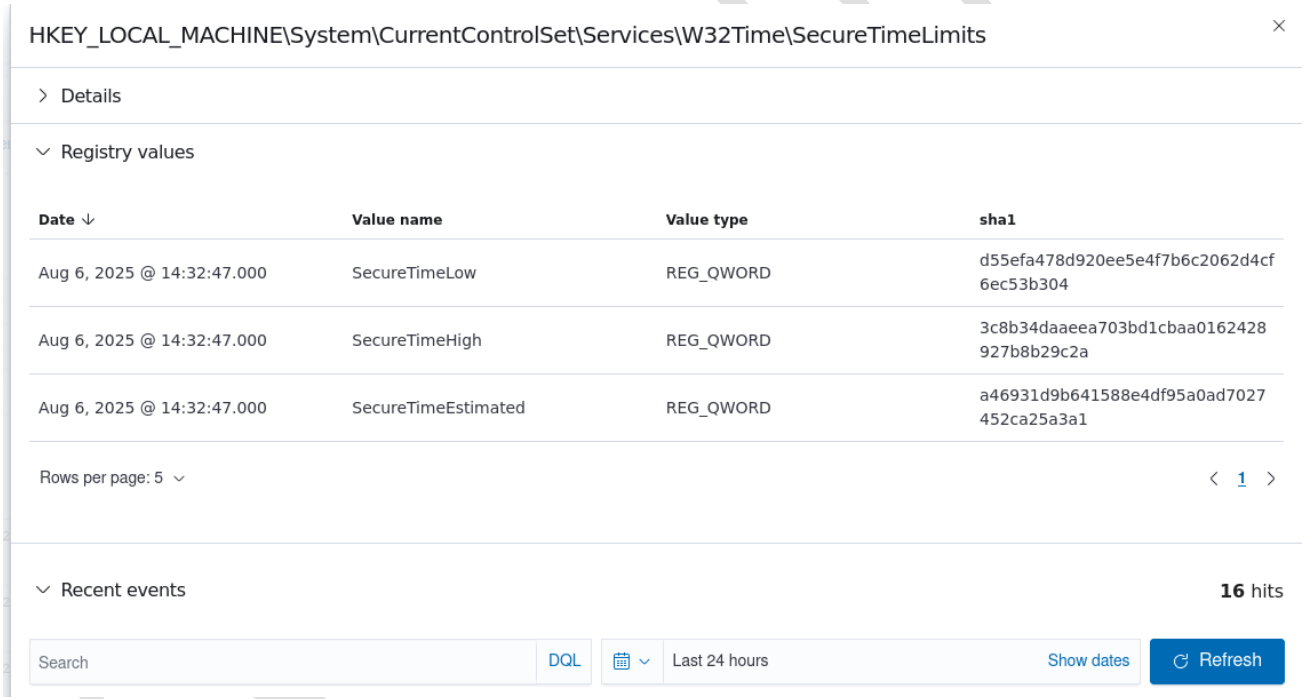


Figure 41.1 The following snapshot shows the detail information of Sha1 form the file which is created temp directory on Endpoint system.

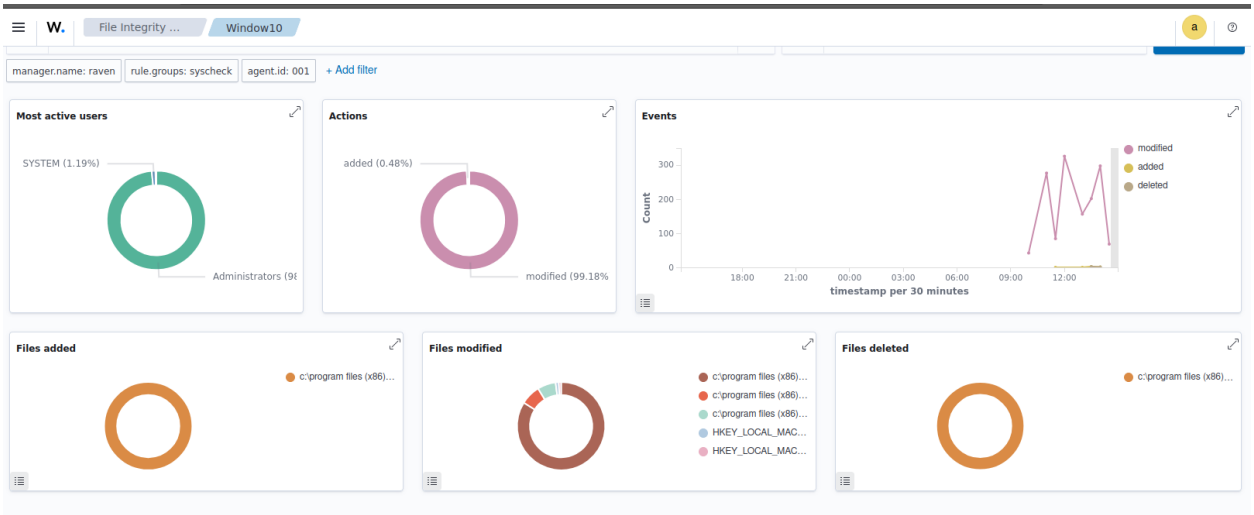


Figure 41.2 The Final Snapshot shows File Integrity's Most active, Actions, Events like Files added, Modified and Deleted.

7. Conclusion

The implementation of Wazuh as an EDR platform, coupled with Sysmon logging, successfully delivered a layered defense system that strengthens endpoint security and visibility. By deploying the Wazuh Manager on Ubuntu and integrating a Windows 10 endpoint, I achieved centralized monitoring and real-time alerting for critical security events. Sysmon logs enhanced detection capabilities by providing deep insights into process executions, network activities, and file changes, which significantly reduced blind spots in endpoint monitoring.

File Integrity Monitoring was configured to detect unauthorized file changes, and automated alerting mechanisms were introduced to respond swiftly to abnormal behaviors. Active response capabilities further ensured that specific threats, such as brute-force attempts, could be mitigated instantly without manual intervention. The integration of MalwareBazaar threat intelligence added an additional layer of proactive defense, allowing for rapid identification of known malware through hash-based detection.

This project demonstrates the importance of combining real-time monitoring, behavioral analysis, and external threat intelligence for a robust cybersecurity posture. By completing this implementation, I successfully created an operational framework that enables early detection, automated responses, and improved situational awareness—critical components for reducing risk and ensuring system resilience against modern cyber threats.