My Private XDR & SIEM Simulation with Wazuh

1. Introduction

Overview of SIEM
Security Information and Event Management (SIEM) is a system that collects, and analyses security data from several sources within an organisation's IT infrastructure. SIEM solutions provide near real-time information on security events and activities, allowing organisations to detect, investigate, and respond to threats.

In addition to traditional features like Security Information Management (SIM) for long-term data storage and Security Event Management (SEM) for near real-time monitoring, modern SIEM solutions now include incident response capabilities. These capabilities allow security teams to not only detect threats but also automate or streamline their response to incidents. This includes features such as automated alerting, playbooks for response actions, and integrations with tools like SOAR (Security Orchestration, Automation, and Response) to enable faster containment and remediation of attacks.

That leads to the tool used in this project, Wazuh.

Overview of Wazuh

Wazuh is an open-source security platform that provides comprehensive threat detection, visibility, and response capabilities across diverse environments, making it much more than a traditional SIEM solution. It combines features such as intrusion detection, log management, file integrity monitoring, vulnerability detection, and compliance management, providing a versatile tool for security operations.[2]

One of the standout features of Wazuh is its ability to act as an Extended Detection and Response (XDR) platform, integrating data from endpoints, networks, cloud environments, and applications to detect advanced threats. It centralises data from multiple sources, allowing for cross-layer correlation and providing a unified view of security events across the entire organisation.

2. Project requisites

System Requirement
Any physical device with sufficient RAM and storage capacity can be used for this project. However, for the implementation of this project, a physical device with 16 GB of RAM and 256 GB of storage was used. This ensured optimal performance for the virtual machines running Wazuh and its components, allowing for efficient data processing and management.

Software Requirement

1. VMware Workstation Pro (hypervisor) is required to create and manage Windows and Ubuntu VMs, simulating a network environment for Wazuh deployment.
2. A Windows 10 Pro VM is required as an endpoint.
3. This setup requires two Linux VMs: one will function as an endpoint, while the other is the Wazuh server. It is important to note that the Wazuh central components can only be installed on a 64-bit Linux operating system (as stated in the Wazuh documentation).[3] For this project, two Ubuntu 20.04 VMs will be utilised.

3. Methodology

The methodology for this project is divided into three key phases: *Setup*, *Testing*, and *Review*. This phased approach ensures a structured deployment and thorough evaluation of Wazuh.

4. Implementation

Setup

Setting up Hypervisor & Virtual Machines
In this guide, I won't cover the steps for installing VMware Workstation Pro or how to set up the VMs within it. That will be addressed in a future article. However, what's most important for this project is configuring the network settings for the VMs to *"Bridged" mode*.

Why Use Bridged Mode?

By selecting the "*Bridged"* network configuration, each virtual machine operates as if it's connected directly to the same network as the host machine. This means that the VMs receive their own IP addresses from the local router or network infrastructure, making them directly accessible within the network.

This setup is crucial for simulating a real-world environment, as it allows the VMs to communicate with each other and the host as independent devices. For Wazuh, this ensures smooth communication between the Wazuh Manager on one VM and the agents on the endpoint VMs, enabling proper data collection and security monitoring. Figure 4.1 below shows the three VMs installed for this project — 2 Ubuntu (linux) and 1 Windows VMs.

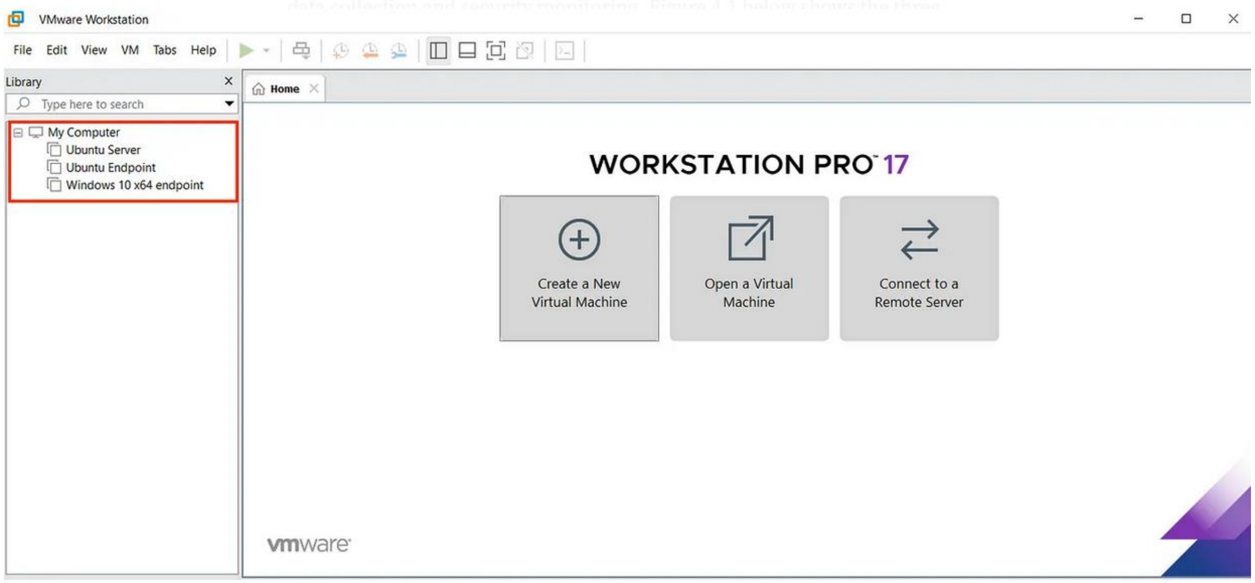


Figure 4.1 — VMs installed in VMWare Workstation Pro

Setting Up Wazuh

The best approach to setting up Wazuh is to install the Wazuh Manager first. Once the manager is up and running, you can then use it to deploy and configure the Wazuh agents on your endpoints. This centralised installation method makes it easier to manage all monitored devices from a single point.

To set up Wazuh on my server, I used the *Wazuh Installation Assistant* rather than going through the manual installation process. This method is not only simpler but ensures that all components are correctly installed and configured.

Why the Wazuh Installation Assistant?

The Wazuh Installation Assistant provides a streamlined, automated installation process for setting up Wazuh. It takes care of all the necessary components like:

- Wazuh Manager: The core component for receiving and analysing security data.
- Elasticsearch: For data storage and indexing.
- Kibana: For visualising logs and alerts in an intuitive dashboard.

Using the installation assistant saves a significant amount of time and reduces the complexity of managing dependencies and configuring each service manually. This method ensures a consistent and error-free setup, which is especially useful for those new to SIEM systems.

During the installation of Wazuh, one of the essential tools required is curl, a widely used command-line utility. curl is designed to transfer data from or to a server using various network protocols, such as HTTP, HTTPS, FTP, and others.

This is how to install curlusing either *snap* or *apt* in the terminal.

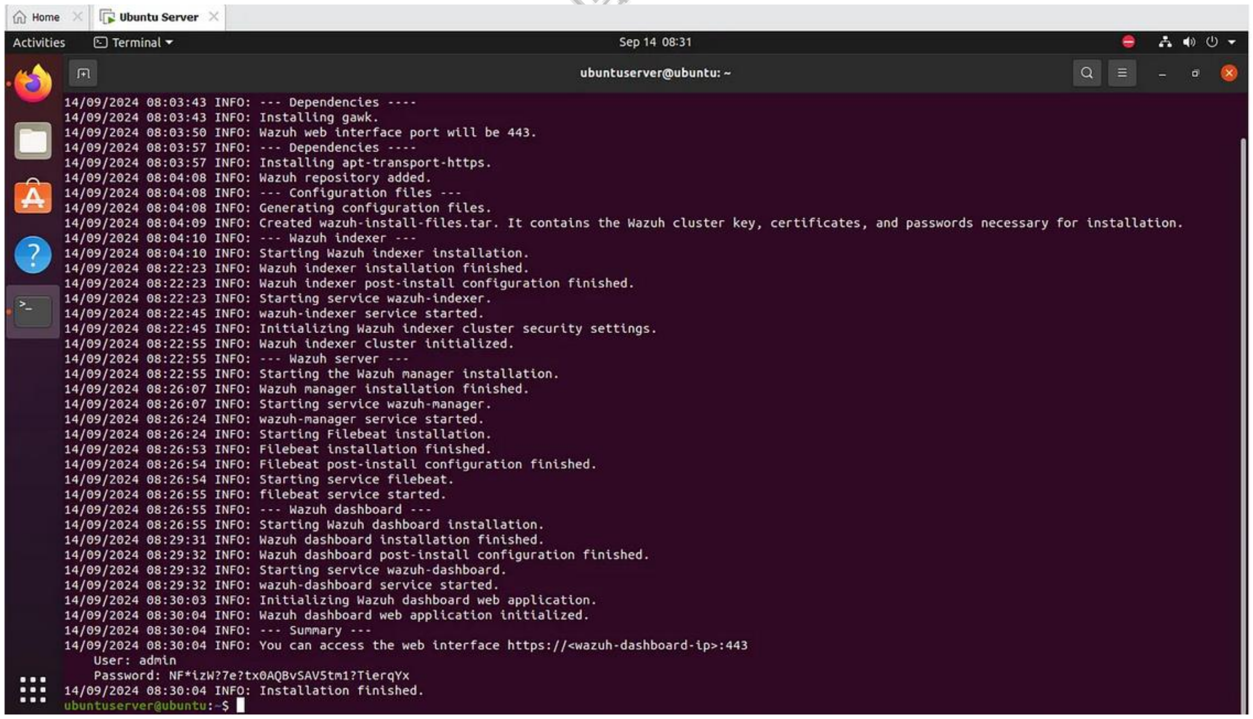
sudo snap install curl

sudo apt install curl

Begin by downloading the installation script directly from the Wazuh repository by executing the following command in the terminal. (The code can be obtained from Wazuh's official website). Keep in mind that the version may vary when you retrieve the code. For this project, version 4.5 was used.

curl -sO https://packages.wazuh.com/4.5/wazuh-install.sh && sudo bash ./wazuh-install.sh -a

Once the assistant finishes the installation, the output shows the access credentials (username and password) and a message that confirms that the installation was successful as shown in Figure 4.2.



```
14/09/2024 08:03:43 INFO: --- Dependencies ----
14/09/2024 08:03:43 INFO: Installing gawk.
14/09/2024 08:03:50 INFO: Wazuh web interface port will be 443.
14/09/2024 08:03:57 INFO: --- Dependencies ----
14/09/2024 08:03:57 INFO: Installing apt-transport-https.
14/09/2024 08:04:08 INFO: Wazuh repository added.
14/09/2024 08:04:08 INFO: --- Configuration files ---
14/09/2024 08:04:08 INFO: Generating configuration files.
14/09/2024 08:04:09 INFO: Created wazuh-install-files.tar. It contains the Wazuh cluster key, certificates, and passwords necessary for installation.
14/09/2024 08:04:10 INFO: --- Wazuh indexer ---
14/09/2024 08:04:10 INFO: Starting Wazuh indexer installation.
14/09/2024 08:22:23 INFO: Wazuh indexer installation finished.
14/09/2024 08:22:23 INFO: Wazuh indexer post-install configuration finished.
14/09/2024 08:22:23 INFO: Starting service wazuh-indexer.
14/09/2024 08:22:45 INFO: wazuh-indexer service started.
14/09/2024 08:22:45 INFO: Initializing Wazuh indexer cluster security settings.
14/09/2024 08:22:55 INFO: Wazuh indexer cluster initialized.
14/09/2024 08:22:55 INFO: --- Wazuh server ---
14/09/2024 08:22:55 INFO: Starting the Wazuh manager installation.
14/09/2024 08:26:07 INFO: Wazuh manager installation finished.
14/09/2024 08:26:07 INFO: Starting service wazuh-manager.
14/09/2024 08:26:24 INFO: wazuh-manager service started.
14/09/2024 08:26:24 INFO: Starting Filebeat installation.
14/09/2024 08:26:53 INFO: Filebeat installation finished.
14/09/2024 08:26:54 INFO: Filebeat post-install configuration finished.
14/09/2024 08:26:54 INFO: Starting service filebeat.
14/09/2024 08:26:55 INFO: filebeat service started.
14/09/2024 08:26:55 INFO: --- Wazuh dashboard ---
14/09/2024 08:26:55 INFO: Starting Wazuh dashboard installation.
14/09/2024 08:29:31 INFO: Wazuh dashboard installation finished.
14/09/2024 08:29:32 INFO: Wazuh dashboard post-install configuration finished.
14/09/2024 08:29:32 INFO: Starting service wazuh-dashboard.
14/09/2024 08:29:32 INFO: wazuh-dashboard service started.
14/09/2024 08:30:03 INFO: Initializing Wazuh dashboard web application.
14/09/2024 08:30:04 INFO: Wazuh dashboard web application initialized.
14/09/2024 08:30:04 INFO: --- Summary ---
14/09/2024 08:30:04 INFO: You can access the web interface https://<wazuh-dashboard-ip>:443
    User: admin
    Password: NF*izW?7e?tx0AQBvSAV5tm1?TierqYx
14/09/2024 08:30:04 INFO: Installation finished.
ubuntuserver@ubuntu:~$
```

Figure 4.2 — Successful installation of Wazuh Manager with access credentials

The next step is to log in to the Wazuh dashboard to monitor and manage. your security environment. You can access the dashboard by typing the IP address of the VM into your web browser's address bar. For example, if the VM's IP is 192.168.1.100, you would enter:

http://192.168.1.100
http://localhost

After typing the IP address or localhost, you will be prompted to log in using the access credentials created during the installation process as shown in Figure 4.3. These credentials provide secure access to the dashboard, ensuring that only authorised users can monitor and manage security alerts, logs, and configurations.

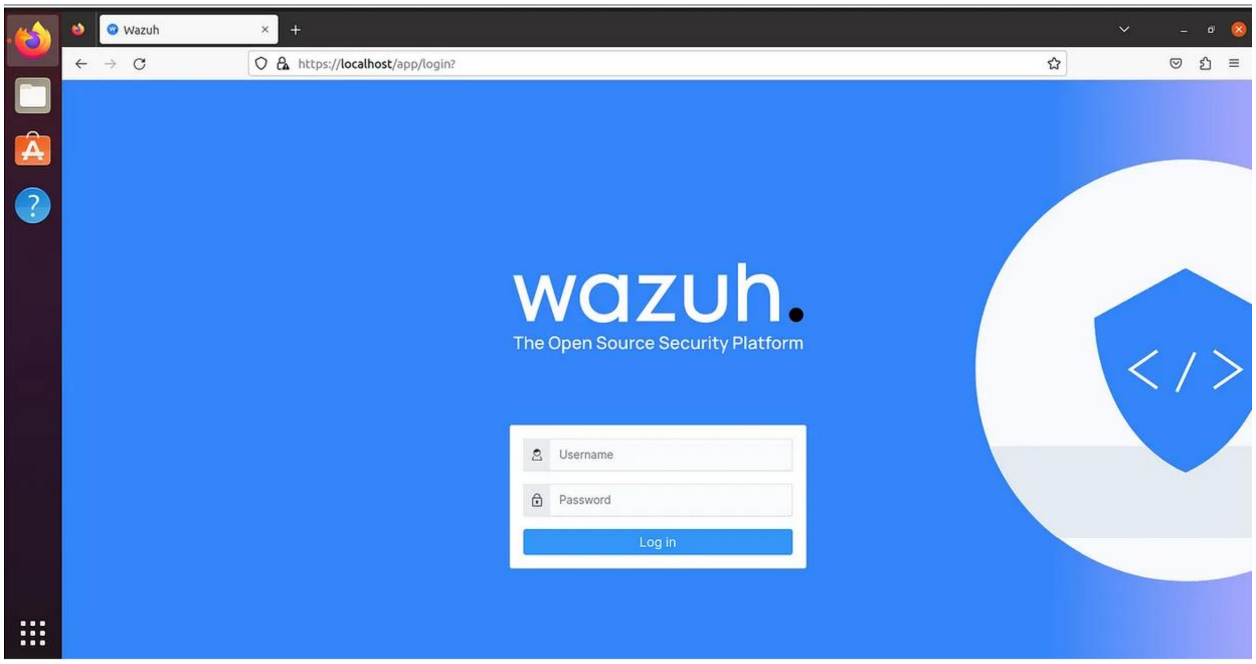

Wazuh
https://localhost/app/login?
wazuh.
The Open Source Security Platform
Username
Password
Log in

Figure 4.3 — Wazuh Manager dashboard login page

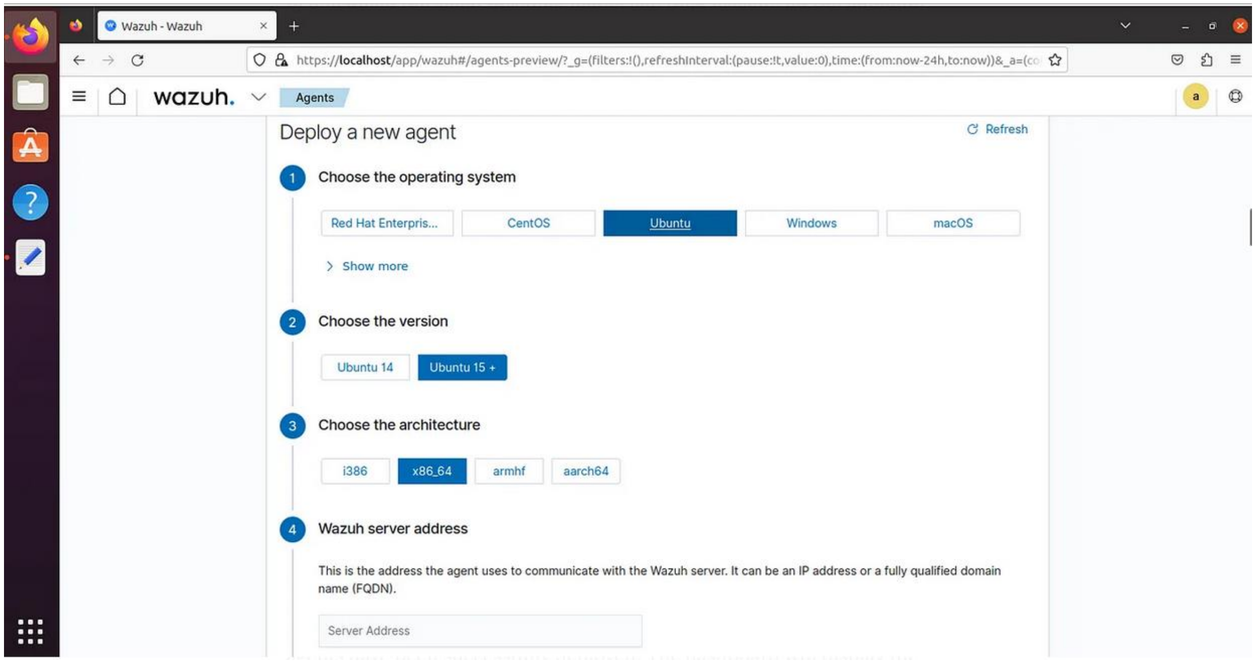
The next thing is to deploy the Linux (Ubuntu) and Windows agents.

Using the Wazuh Manager dashboard to set up agents is by far the simplest and most efficient method compared to manual configurations or other installation methods. The dashboard provides a user-friendly interface that automates much of the process, removing the need for manual file edits and complicated command-line operations.

Deploying the agents

You'll need to know the operating system, version, architecture (32-bit or 64-bit), and the IP address of the Wazuh Manager before deploying an agent. These details are essential to ensure the agent installs correctly and can communicate with the Wazuh Manager.

Next, click on *Add agent* and select the appropriate operating system (Linux or Windows). The dashboard will generate a registration command or a link to download the agent installer, which you then execute on the endpoint machine. Figure 4.4 and Figure 4.5 show an example.


Wazuh - Wazuh
https://localhost/app/wazuh#/agents-preview/?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-24h,to:now))&_a=(co
wazuh.
Agents
Refresh

Deploy a new agent

1 Choose the operating system

Red Hat Enterpris... | CentOS | **Ubuntu** | Windows | macOS

Show more

2 Choose the version

Ubuntu 14 | **Ubuntu 15 +**

3 Choose the architecture

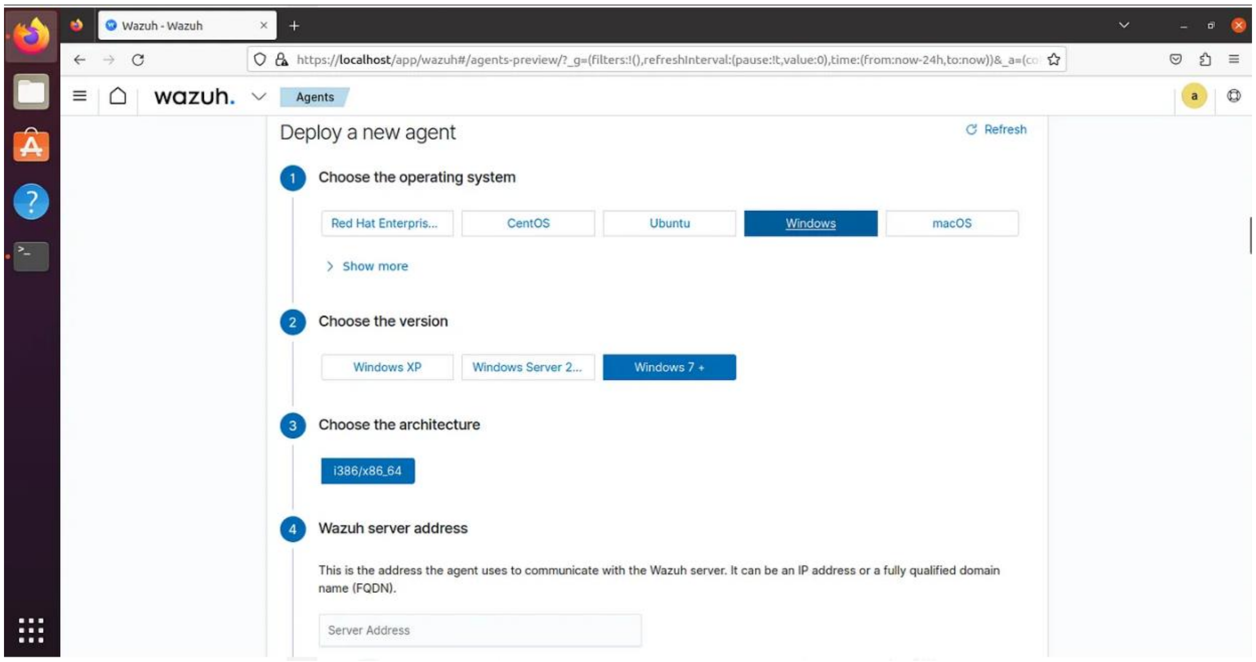
i386 | **x86_64** | armhf | aarch64

4 Wazuh server address

This is the address the agent uses to communicate with the Wazuh server. It can be an IP address or a fully qualified domain name (FQDN).

Server Address

Figure 4.4 — Deploying Linux(Ubuntu) agent



Deploy a new agent

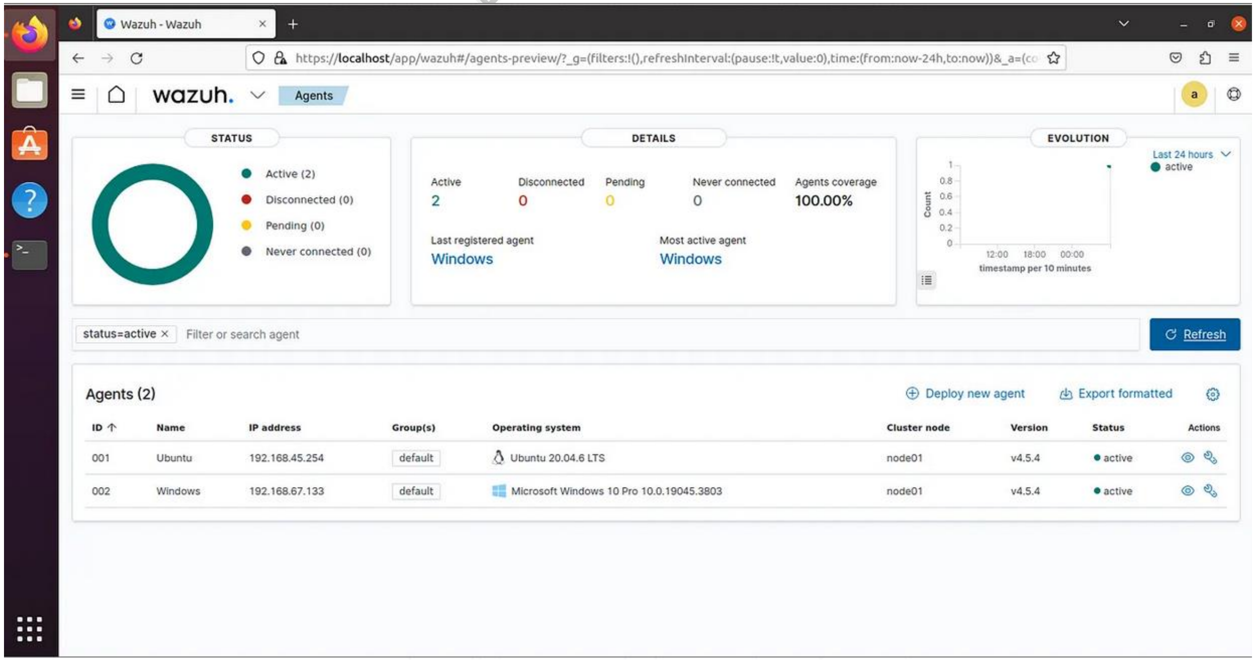
1. Choose the operating system: Red Hat Enterpris... | CentOS | Ubuntu | **Windows** | macOS
2. Choose the version: Windows XP | Windows Server 2... | **Windows 7 +**
3. Choose the architecture: **i386/x86_64**
4. Wazuh server address

This is the address the agent uses to communicate with the Wazuh server. It can be an IP address or a fully qualified domain name (FQDN).

Server Address

Figure 4.5 — Deploying Windows agent

After completing the steps, you can check the Wazuh dashboard to see if the agents have been successfully deployed. The dashboard will display the status of each agent, confirming whether they are connected and actively reporting data to the Wazuh Manager. This allows you to ensure that your security monitoring is operational and effectively collecting data from all monitored endpoints as shown in Figure 4.6.



STATUS: Active (2), Disconnected (0), Pending (0), Never connected (0)

DETAILS

Active	Disconnected	Pending	Never connected	Agents coverage
2	0	0	0	100.00%

Last registered agent: Windows
Most active agent: Windows

status=active

Agents (2)

ID	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
001	Ubuntu	192.168.45.254	default	Ubuntu 20.04.6 LTS	node01	v4.5.4	● active	
002	Windows	192.168.67.133	default	Microsoft Windows 10 Pro 10.0.19045.3803	node01	v4.5.4	● active	

Figure 4.6 — Agents successfully deployed

Testing

Here are some ways to test the capabilities of Wazuh:

- File Integrity Monitoring (FIM).
- Log Monitoring.
- Intrusion Detection.

When testing Wazuh's capabilities, it's important to note that most tests you run on one operating system can be replicated on another. Whether you're working with Linux or Windows, Wazuh agents function similarly across both platforms. For example, if you conduct a File Integrity Monitoring (FIM) test on Linux, the same test can be performed on a Windows machine, yielding comparable results. This means there's no need to repeat the same test on both operating systems in this project unless you're specifically interested in platform-specific behaviour.

Starting with File Integrity Monitoring (FIM)

To begin testing, I'll start with File Integrity Monitoring (FIM). Wazuh monitors files for any changes, such as modifications, deletions, or new file creation. For this test, I'll create a few files in my **Documents** directory and manipulate them to see how Wazuh detects these changes.

Before running the test, I'll edit the Wazuh configuration file to specifically monitor the **Documents** directory. This involves updating the configuration to watch that directory for any file modifications or alterations. Once the configuration is set, Wazuh will track any changes I make in my **Documents** directory, logging them and generating alerts on the dashboard. This allows me to test how Wazuh responds to various file activities in real time.

I will test the FIM on the Windows endpoint.

First, you need to edit the Wazuh agent configuration to monitor specific directories on your Windows machine. Open the Wazuh agent configuration file located at:

C:\Program Files (x86)\ossec-agent\ossec.conf

Add a new section to monitor your desired directory (e.g., Documents):

```
<syscheck>
  <directories>
```

```
    <directory>C:\Users\<YourUsername>\Documents</directory>
  </directories>
</syscheck>
```

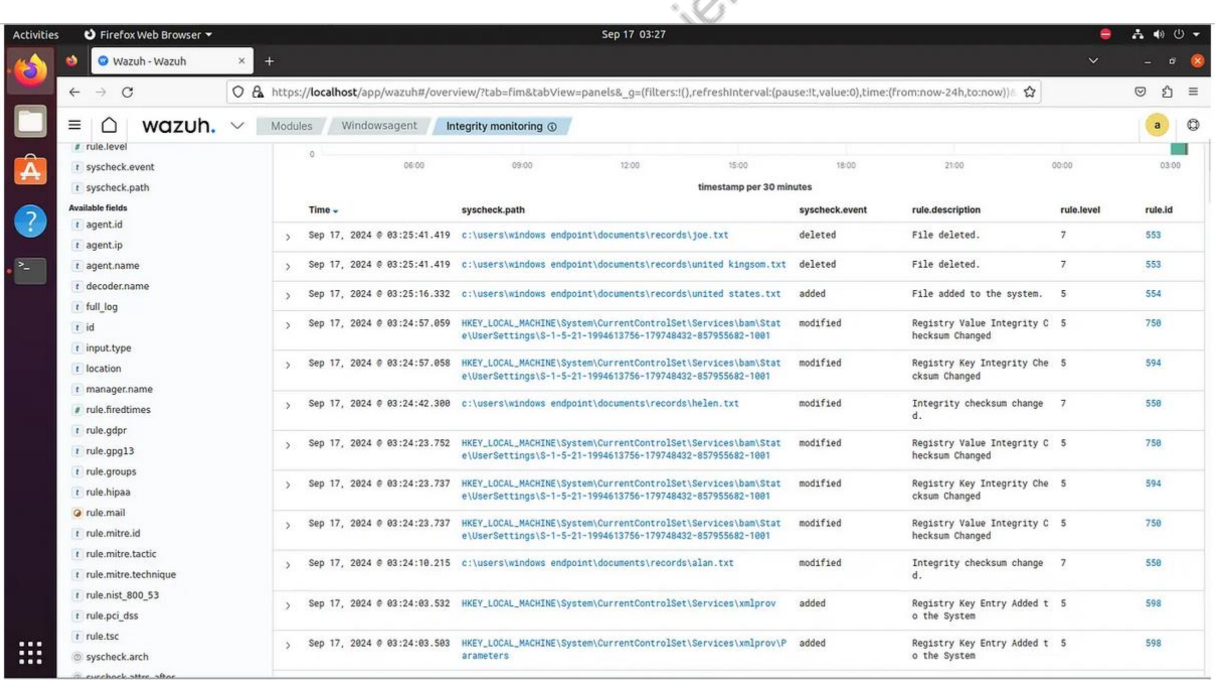
Save the file and restart the Wazuh agent service from the Windows Services menu or using the command prompt:

```
net stop wazuh
net start wazuh
```

Now that the **Documents** directory is being monitored, make some changes within the folder:

- Create, modify, rename, or delete files within the monitored directory.

After performing the file changes, log into the Wazuh Manager dashboard and go to the Security Events or File Integrity Monitoring section. You should see detailed logs of the changes made to the files in the monitored directory.



Time	syscheck.path	syscheck.event	rule.description	rule.level	rule.id
Sep 17, 2024 @ 03:25:41.419	c:\users\windows endpoint\documents\records\joe.txt	deleted	File deleted.	7	553
Sep 17, 2024 @ 03:25:41.419	c:\users\windows endpoint\documents\records\united kingsom.txt	deleted	File deleted.	7	553
Sep 17, 2024 @ 03:25:16.332	c:\users\windows endpoint\documents\records\united states.txt	added	File added to the system.	5	554
Sep 17, 2024 @ 03:24:57.059	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-1994613756-179748432-857955682-1001	modified	Registry Value Integrity Checksum Changed	5	750
Sep 17, 2024 @ 03:24:57.058	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-1994613756-179748432-857955682-1001	modified	Registry Key Integrity Checksum Changed	5	594
Sep 17, 2024 @ 03:24:42.300	c:\users\windows endpoint\documents\records\helen.txt	modified	Integrity checksum changed.	7	550
Sep 17, 2024 @ 03:24:23.752	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-1994613756-179748432-857955682-1001	modified	Registry Value Integrity Checksum Changed	5	750
Sep 17, 2024 @ 03:24:23.737	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-1994613756-179748432-857955682-1001	modified	Registry Key Integrity Checksum Changed	5	594
Sep 17, 2024 @ 03:24:23.737	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-1994613756-179748432-857955682-1001	modified	Registry Value Integrity Checksum Changed	5	750
Sep 17, 2024 @ 03:24:10.215	c:\users\windows endpoint\documents\records\alan.txt	modified	Integrity checksum changed.	7	550
Sep 17, 2024 @ 03:24:03.532	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\xmlprov	added	Registry Key Entry Added to the System	5	598
Sep 17, 2024 @ 03:24:03.503	HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\xmlprov\Parameters	added	Registry Key Entry Added to the System	5	598

Figure 4.7 — Integrity Monitoring on Wazuh dashboard

After testing FIM in Wazuh, you will encounter several important pieces of information in the logs or the Wazuh dashboard as shown in Figure 4.7. Here's an overview of key fields and what they represent:

Time

- The **time** field records the exact timestamp when a change occurred in the monitored files or directories. This is crucial for tracking when a file was created, modified, or deleted. The timestamp helps correlate events and identify patterns or anomalies in file activity.

syscheck.path

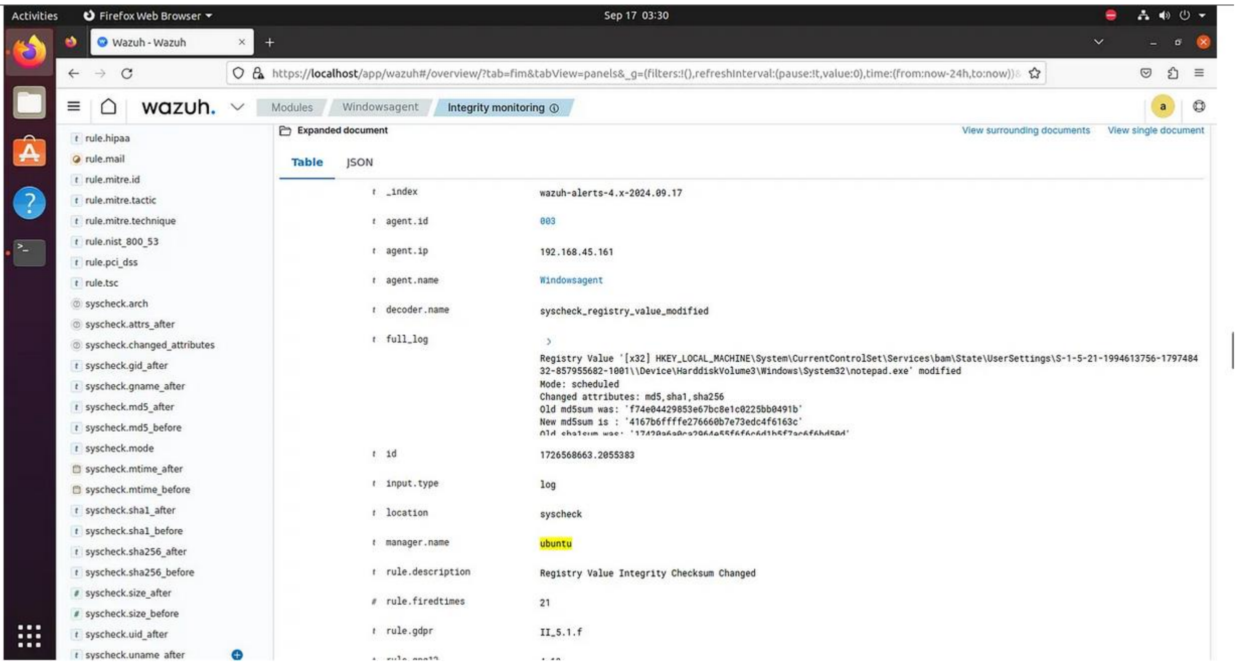
- The **syscheck.path** field shows the specific path of the file or directory that was modified. This tells you exactly where the change took place on the monitored endpoint, allowing you to pinpoint which files or folders were altered.

syscheck.event

- The **syscheck.event** field describes the type of change that occurred, such as added, modified, deleted, or renamed. This information is essential for understanding what kind of action was performed on the file, helping you distinguish between different types of file activities.

Rule Description

- Each file change triggers a specific rule within Wazuh's rule set, and the **rule description** provides a human-readable explanation of the event. For example, if a file is modified, the rule description might state something like *"File integrity checksum changed"* or *"File was deleted"*. This gives context to the event and helps you assess whether the activity is normal or suspicious.



```
rule.hipaa
rule.mail
rule.mitre.id
rule.mitre.tactic
rule.mitre.technique
rule.nist_800_53
rule.pci_dss
rule.tsc
syscheck.arch
syscheck.attrs_after
syscheck.changed_attributes
syscheck.gid_after
syscheck.gname_after
syscheck.md5_after
syscheck.md5_before
syscheck.mode
syscheck.mtime_after
syscheck.mtime_before
syscheck.sha1_after
syscheck.sha1_before
syscheck.sha256_after
syscheck.sha256_before
syscheck.size_after
syscheck.size_before
syscheck.uid_after
syscheck.uname_after
```

```
Expanded document                                   View surrounding documents   View single document

Table   JSON

_index              wazuh-alerts-4.x-2024.09.17
agent.id            003
agent.ip            192.168.45.161
agent.name          Windowsagent
decoder.name        syscheck_registry_value_modified
full_log            Registry Value '[x32] HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-1994613756-1797484
                    32-857955682-1001\\Device\HarddiskVolume3\Windows\System32\notepad.exe' modified
                    Mode: scheduled
                    Changed attributes: md5,sha1,sha256
                    Old md5sum was: 'f74e04429853e67bc8e1c0225bb0491b'
                    New md5sum is : '4167b6ffffe276660b7e73edc4f6163c'
id                  1726568663.2055383
input.type          log
location            syscheck
manager.name        ubuntu
rule.description    Registry Value Integrity Checksum Changed
rule.firedtimes     21
rule.gdpr           II_5.1.f
```

Figure 4.8 — FIM showing details of a security event

In the Wazuh Manager dashboard, each event logged during File Integrity Monitoring (FIM) can be explored further using a *drop-down button*. This feature allows you to check detailed information about each event, giving you deeper insights into what exactly occurred as shown in Figure 4.8.

Key Details Revealed by the Drop-Down Button:

1. **Timestamp**: You can see the precise time the event took place, useful for tracking the exact moment a file was modified, created, or deleted.
2. **Affected File Path**: The drop-down will show the full path of the file or directory where the event happened, making it clear which specific file or folder was involved.
3. **Event Type**: It provides details on whether the file was modified, added, deleted, or renamed. This helps you distinguish between various types of file activity.
4. **Rule and Alert Info**: The drop-down reveals the specific Wazuh rule that was triggered, including a description explaining the nature of the event (e.g., "File integrity checksum changed"). This helps you understand why the event was flagged and how it aligns with your security policies.
5. **Additional Metadata**: In some cases, you may also see metadata like file hashes or checksums, which are used to track file integrity and changes in file contents. This is particularly useful when identifying unauthorized modifications.

Why It's Useful?

The drop-down button enhances visibility into the event, offering more context and helping you assess the severity of the activity. Whether it's a minor file modification or a potential security breach, this feature allows you to make informed decisions and take appropriate action.

Log monitoring

I do this test on the Ubuntu machine.

During my testing with Wazuh, I performed several actions, and I monitored how Wazuh captured each event as shown in Figure 4.9. The following is what I did.

1. Stopping and Starting the Wazuh Agent

First, I stopped and started the Wazuh agent service. This action was crucial as it changed the operational state of the agent. I noticed that Wazuh logged these service status changes, allowing me to monitor the agent's availability and ensure it was running correctly.

2. User Lockout and Failed Login Attempts

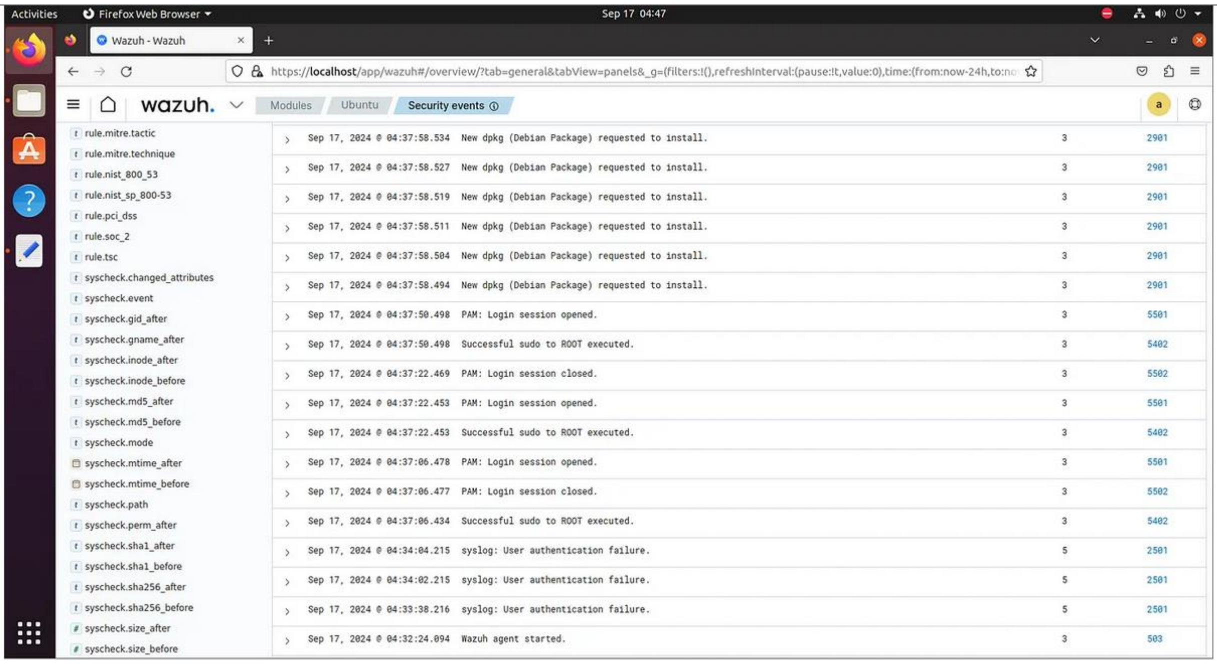
Next, I locked myself out of the user account and then attempted to log back in with incorrect passwords several times. Wazuh tracked these failed login attempts as potential security incidents. It was reassuring to see how it could detect such behaviour, which is essential for identifying brute force attacks or unauthorized access attempts. The logs included the timestamp, my username, and the source of the login attempt.

3. Successful Login

After a few attempts, I successfully logged back in, and Wazuh logged this event as well. This log entry was important for maintaining a security audit trail, as it provided insights into my access patterns and confirmed that my login was legitimate following the failed attempts.

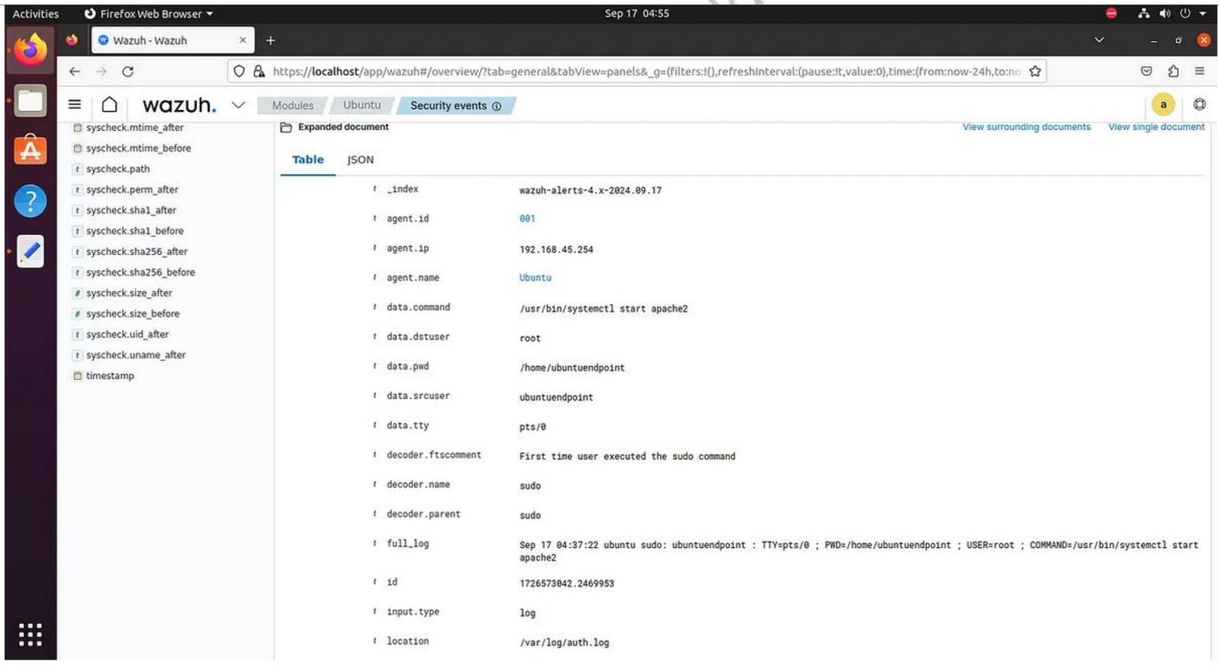
4. Installing Apache2 with Root Access

Finally, I decided to install Apache2 after granting my user account root access. Wazuh captured the entire installation process, including the commands I executed and the system changes made during the installation as shown in Figure 4.10. This logging was particularly helpful in tracking significant modifications to the server configuration and services being enabled.



```
Sep 17, 2024 @ 04:37:58.534  New dpkg (Debian Package) requested to install.   3   2901
Sep 17, 2024 @ 04:37:58.527  New dpkg (Debian Package) requested to install.   3   2901
Sep 17, 2024 @ 04:37:58.519  New dpkg (Debian Package) requested to install.   3   2901
Sep 17, 2024 @ 04:37:58.511  New dpkg (Debian Package) requested to install.   3   2901
Sep 17, 2024 @ 04:37:58.504  New dpkg (Debian Package) requested to install.   3   2901
Sep 17, 2024 @ 04:37:58.494  New dpkg (Debian Package) requested to install.   3   2901
Sep 17, 2024 @ 04:37:50.498  PAM: Login session opened.                        3   5501
Sep 17, 2024 @ 04:37:50.498  Successful sudo to ROOT executed.                 3   5402
Sep 17, 2024 @ 04:37:22.469  PAM: Login session closed.                        3   5502
Sep 17, 2024 @ 04:37:22.453  PAM: Login session opened.                        3   5501
Sep 17, 2024 @ 04:37:22.453  Successful sudo to ROOT executed.                 3   5402
Sep 17, 2024 @ 04:37:06.478  PAM: Login session opened.                        3   5501
Sep 17, 2024 @ 04:37:06.477  PAM: Login session closed.                        3   5502
Sep 17, 2024 @ 04:37:06.434  Successful sudo to ROOT executed.                 3   5402
Sep 17, 2024 @ 04:34:04.215  syslog: User authentication failure.              5   2501
Sep 17, 2024 @ 04:34:02.215  syslog: User authentication failure.              5   2501
Sep 17, 2024 @ 04:33:38.216  syslog: User authentication failure.              5   2501
Sep 17, 2024 @ 04:32:24.094  Wazuh agent started.                              3   503
```

Figure 4.9 — Monitoring logs on Wazuh dashboard



```
_index              wazuh-alerts-4.x-2024.09.17
agent.id            001
agent.ip            192.168.45.254
agent.name          Ubuntu
data.command        /usr/bin/systemctl start apache2
data.dstuser        root
data.pwd            /home/ubuntuendpoint
data.srcuser        ubuntuendpoint
data.tty            pts/0
decoder.ftscomment  First time user executed the sudo command
decoder.name        sudo
decoder.parent      sudo
full_log            Sep 17 04:37:22 ubuntu sudo: ubuntuendpoint : TTY=pts/0 ; PWD=/home/ubuntuendpoint ; USER=root ; COMMAND=/usr/bin/systemctl start
                    apache2
id                  1726573042.2469953
input.type          log
location            /var/log/auth.log
```

Figure 4.10 — Details of Apache2 installation log

Intrusion detection

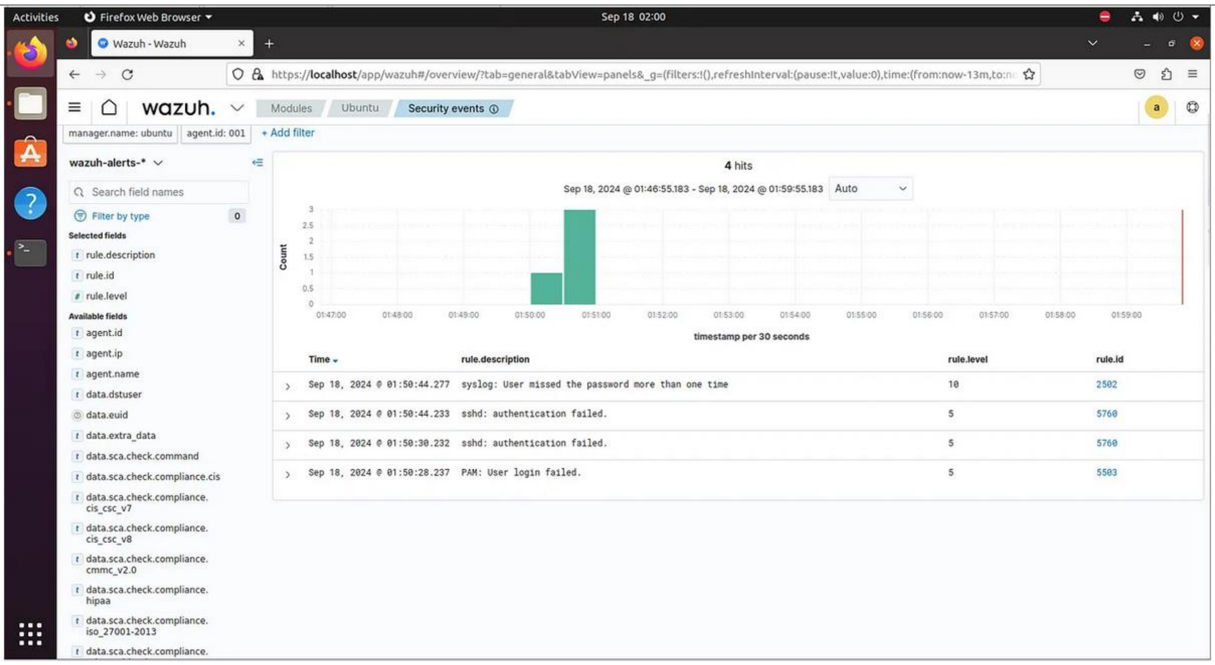
I decided to test Wazuh's intrusion detection capabilities by simulating an attack using failed SSH login attempts. This test demonstrates how Wazuh can detect potential unauthorised access attempts on a network.

I initiated an SSH connection from the Wazuh server to the Ubuntu endpoint but deliberately entered the wrong password multiple times. This simulated an attempted unauthorised login, similar to a brute force attack where an attacker tries multiple passwords to gain access.

To do that is simple, on the source machine, use the following command to SSH into the destination machine:

ssh username@destination_ip

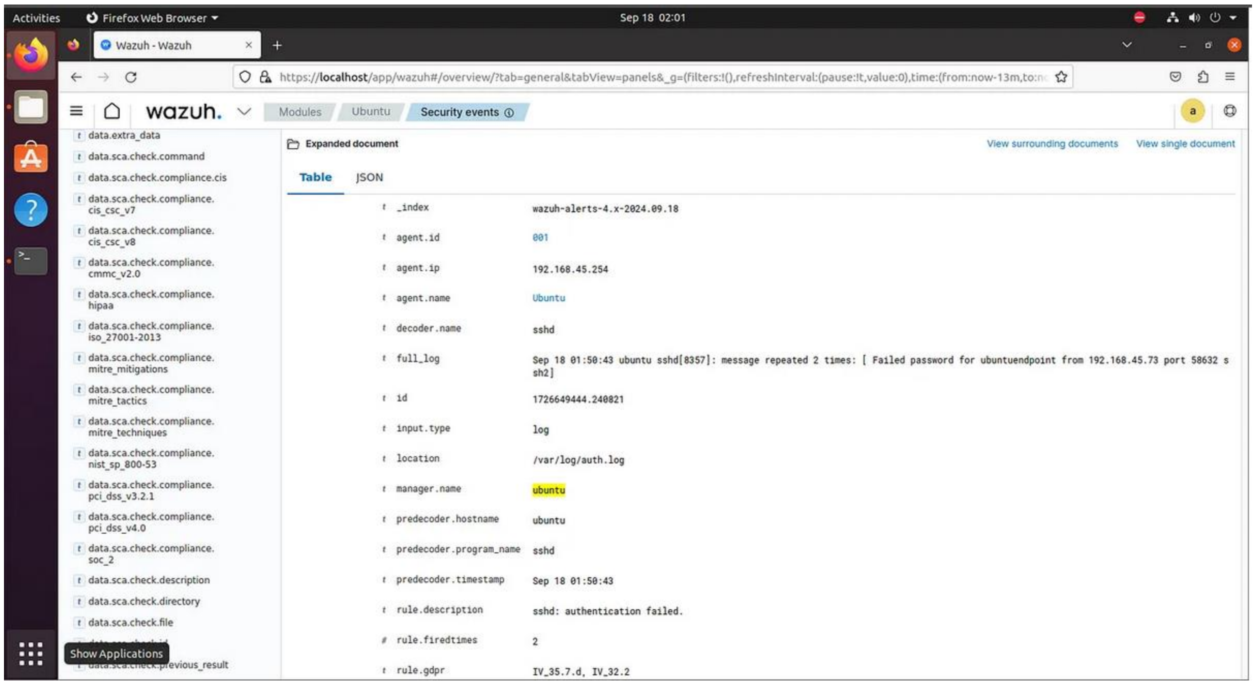
Figure 4.11 below shows the log generated on the Wazuh dashboard. It notifies administrators about unauthorised access attempts, so they can take preemptive security measures.


Activities
Firefox Web Browser
Sep 18 02:00
Wazuh - Wazuh
https://localhost/app/wazuh#/overview/?tab=general&tabView=panels&_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-13m,to:n
wazuh.
Modules
Ubuntu
Security events
manager.name: ubuntu
agent.id: 001
+ Add filter
wazuh-alerts-*
Search field names
Filter by type
Selected fields
rule.description
rule.id
rule.level
Available fields
agent.id
agent.ip
agent.name
data.dstuser
data.euid
data.extra_data
data.sca.check.command
data.sca.check.compliance.cis
data.sca.check.compliance.cis_csc_v7
data.sca.check.compliance.cis_csc_v8
data.sca.check.compliance.cmmc_v2.0
data.sca.check.compliance.hipaa
data.sca.check.compliance.iso_27001-2013
data.sca.check.compliance.
4 hits
Sep 18, 2024 @ 01:46:55.183 - Sep 18, 2024 @ 01:59:55.183
Auto
Count
timestamp per 30 seconds

```
Time                          rule.description                                   rule.level  rule.id
Sep 18, 2024 @ 01:50:44.277   syslog: User missed the password more than one time   10        2502
Sep 18, 2024 @ 01:50:44.233   sshd: authentication failed.                          5         5760
Sep 18, 2024 @ 01:50:30.232   sshd: authentication failed.                          5         5760
Sep 18, 2024 @ 01:50:28.237   PAM: User login failed.                               5         5503
```

Figure 4.11 — ssh attempts logged by Wazuh

Figure 4.12 also gives details about the ssh attempt.


Activities
Firefox Web Browser
Sep 18 02:01
Wazuh - Wazuh
https://localhost/app/wazuh#/overview/?tab=general&tabView=panels&_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-13m,to:n
wazuh.
Modules
Ubuntu
Security events
Expanded document
View surrounding documents
View single document
Table
JSON
data.extra_data
data.sca.check.command
data.sca.check.compliance.cis
data.sca.check.compliance.cis_csc_v7
data.sca.check.compliance.cis_csc_v8
data.sca.check.compliance.cmmc_v2.0
data.sca.check.compliance.hipaa
data.sca.check.compliance.iso_27001-2013
data.sca.check.compliance.mitre_mitigations
data.sca.check.compliance.mitre_tactics
data.sca.check.compliance.mitre_techniques
data.sca.check.compliance.nist_sp_800-53
data.sca.check.compliance.pci_dss_v3.2.1
data.sca.check.compliance.pci_dss_v4.0
data.sca.check.compliance.soc_2
data.sca.check.description
data.sca.check.directory
data.sca.check.file
Show Applications
data.sca.check.previous_result

```
_index                   wazuh-alerts-4.x-2024.09.18
agent.id                 001
agent.ip                 192.168.45.254
agent.name               Ubuntu
decoder.name             sshd
full_log                 Sep 18 01:50:43 ubuntu sshd[8357]: message repeated 2 times: [ Failed password for ubuntuendpoint from 192.168.45.73 port 58632 s
                         sh2]
id                       1726649444.240821
input.type               log
location                 /var/log/auth.log
manager.name             ubuntu
predecoder.hostname      ubuntu
predecoder.program_name  sshd
predecoder.timestamp     Sep 18 01:50:43
rule.description         sshd: authentication failed.
rule.firedtimes          2
rule.gdpr                IV_35.7.d, IV_32.2
```

Figure 4.12 — Details about ssh attempt

Review

As part of the security tests conducted so far, I explored several core functionalities of Wazuh, including File Integrity Monitoring (FIM), log monitoring, and intrusion detection. These tests provided valuable insights into Wazuh's strengths and areas where it can improve.

File Integrity Monitoring (FIM)

Wazuh's File Integrity Monitoring (FIM) feature is designed to detect changes to files or directories and flag any unexpected or unauthorised modifications. For the test, I configured Wazuh to monitor the /Documents directory on both my Windows and Linux machines. After deliberately creating, modifying, and deleting files, Wazuh successfully logged each event, detailing the time, type of modification (e.g., file added, changed, deleted), and the exact file path. The drop-down menu on the Wazuh dashboard allowed me to examine the finer details of each event, such as timestamps, user actions, and rule descriptions. This feature is particularly useful for monitoring critical system files and ensuring no malicious tampering occurs.

Log Monitoring

For log monitoring, I focused on my Ubuntu machine. The first tests involved stopping and starting the Wazuh agent, logging in and out of user accounts, deliberately entering incorrect passwords multiple times, and attempting to install software as a root user. Wazuh's log monitoring capability captured all these activities, offering real-time detection of suspicious behaviour. The logs were easy to view and analyse through the dashboard, where details such as user actions, IP addresses, and event severity were clearly displayed.

Intrusion Detection

For the intrusion detection test, I focused on SSH activity between my Wazuh server and an Ubuntu endpoint. After intentionally entering an incorrect password several times, Wazuh flagged the failed authentication attempts and recorded them as potential brute-force attempts. This prompt detection of suspicious login activity highlighted the value of Wazuh's built-in rules for intrusion detection, helping to protect against attacks such as brute-force and credential stuffing.

Downsides of Wazuh

1. Manual Dashboard Refresh
One of the notable downsides of the Wazuh dashboard is that it does not auto-refresh. This can be a frustrating experience when monitoring real-time events or reviewing system status. Users must manually refresh the page to see the latest logs, alerts, or status changes. In fast-paced environments where real-time information is crucial, this lack of automation can slow down incident response and create inefficiencies. It would be more effective if the dashboard provided auto-refresh capabilities to continuously display up-to-date data without requiring manual intervention.

2. Technical Complexity of Editing Config Files
Wazuh's configuration files are essential for fine-tuning the system, but editing them requires a certain level of technical know-how. The configuration involves writing and modifying scripts in XML format, which can be tedious and error-prone for users who aren't familiar with scripting or system administration. Even small syntax mistakes in these config files can cause the system to malfunction or not apply desired rules, adding a layer of difficulty for less experienced users. A more user-friendly interface or guided editing tools

for managing configuration settings would significantly improve usability and reduce errors.

3. Inability to Change Automatically Generated Access Credentials
When setting up Wazuh, access credentials for the dashboard are automatically generated during the installation process. Unfortunately, there is no straightforward way to change these credentials later. This limitation poses a security concern because it violates best practices, which recommend changing and personalising credentials to improve security hygiene. Not having the ability to rotate or update these access credentials leaves the system vulnerable if the default credentials are compromised. Wazuh should provide an easier way for administrators to reset and manage access credentials to enhance security.

Other Capabilities of Wazuh

While FIM, log monitoring, and intrusion detection are some of the most commonly tested features, Wazuh offers a variety of other security tools that can enhance an organisation's security posture:

Vulnerability Detection: Wazuh scans for known vulnerabilities in both operating systems and applications. It integrates with CVE databases to detect vulnerabilities and provide recommendations on mitigation steps. This feature is critical for maintaining up-to-date software and preventing known exploits.

Configuration Assessment: This feature checks system configurations against security best practices and compliance standards. It helps identify weak configurations, such as disabled firewalls or weak password policies.

Cloud Security Monitoring: Wazuh also extends its capabilities to cloud environments, providing monitoring and threat detection for cloud workloads and services.

Active Response: Wazuh includes an active response feature that automates defensive actions against threats in real time. For example, it can block suspicious IP addresses, stop malicious processes, or prevent further unauthorised access. This capability allows security teams to mitigate threats faster and more efficiently, reducing the time between detection and response.

5. Conclusion

Wazuh proves to be a powerful and versatile SIEM and XDR tool, offering essential security capabilities such as File Integrity Monitoring, log monitoring, intrusion detection, and vulnerability detection. Throughout the testing process, I observed Wazuh's effectiveness in capturing critical events and providing comprehensive visibility across multiple endpoints. However, like any robust security solution, Wazuh has its downsides.

Despite these challenges, Wazuh remains a highly valuable tool for organisations seeking an open-source, cost-effective solution to monitor and secure their systems. With the right configuration and tuning, it can significantly strengthen security operations and incident response efforts. As with any security tool, understanding its limitations and continuously improving your setup will ensure that Wazuh can meet your security needs effectively.

6. Sources

1. https://www.gartner.com/en/information-technology/glossary/security-information-and-event-management-siem
2. https://wazuh.com/platform/overview/
3. https://documentation.wazuh.com/current/installation-guide/wazuh-server/index.html