

Wazuh SIEM & File Integrity Monitoring Deployment

Adewole Adebayo

25\02\2026

1. Introduction

Wazuh is a free, open-source security platform that unifies Security Information and Event Management (SIEM) and Extended Detection and Response (XDR) capabilities. It provides real-time threat detection, monitoring, and incident response for endpoints, cloud workloads, and on-premises environments.

2. Importance

- **Early Breach Detection:** FIM acts as an early warning system. Unauthorized changes to system files, registry keys, or log files are often the first signs of a malware infection, ransomware, or an active hacker attempting to hide their tracks.
- **Regulatory Compliance:** Many industry standards mandate FIM to protect sensitive data. Key regulations include:
 - **PCI DSS:** Required for organizations handling credit card data.
 - **HIPAA:** Essential for protecting electronic health records.

- **GDPR:** Necessary for ensuring the integrity of personal data.
- **Detection of Insider Threats:** While firewalls and antivirus tools focus on external threats, FIM tracks changes made by internal users. This identifies accidental or malicious modifications made by individuals who already have legitimate access to the system.
- **Incident Response & Forensics:** When a breach occurs, FIM provides a detailed "who, what, when, and how" audit trail. This data is vital for forensic investigators to determine the scope of the damage and restore systems to a trusted state.

3. Goal

- To Detect Unauthorized File Changes
- Alert Tuning & Investigation

1. Documenting VirtualBox Setup

1.1 Virtualization Platform

Platform: VirtualBox

Host OS: Windows 11

Host RAM: 16GB

Purpose: Create isolated environment for SIEM deployment and agent testing, and file monitoring.

1.2 Virtual Machine Configuration

VM 1 – Wazuh Manager (Ubuntu Server)

- OS: Ubuntu Server 22.04 LTS
- RAM: 6 GB
- CPU: 4 cores
- Storage: 70 GB dynamically allocated
- Network Mode: Bridged Adapter

VM 2 – Windows Server Agent

- OS: Windows Server 2025
- RAM: 4 GB
- CPU: 2 cores
- Storage: 50 GB
- Network Mode: Bridged Adapter

1.3 Why Bridged Network?

Bridged networking was selected to:

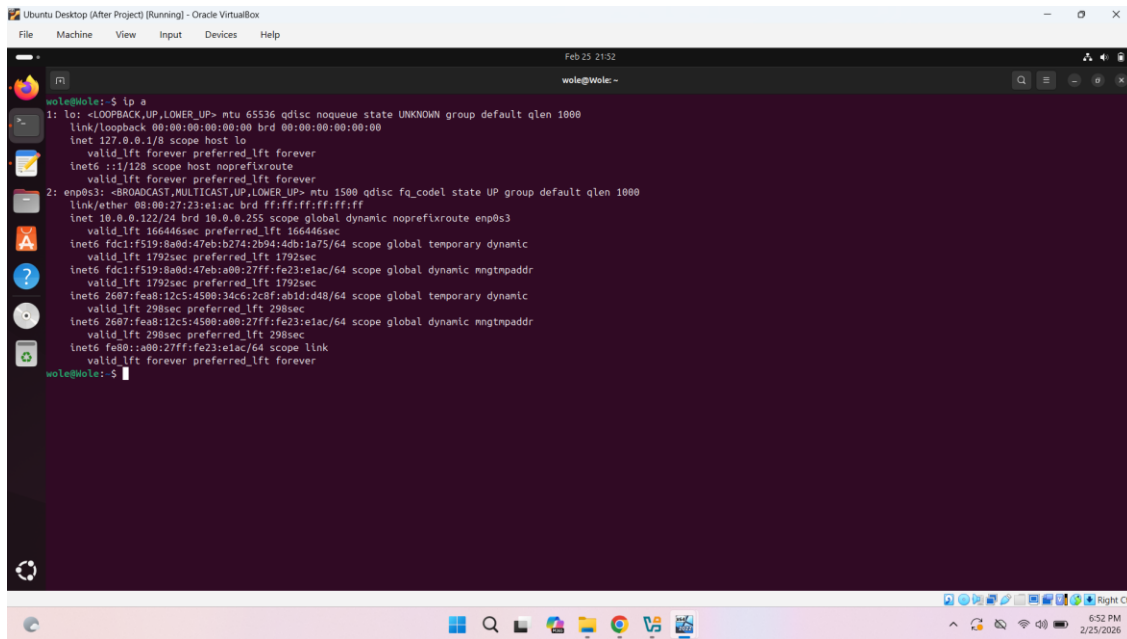
- Allow both VMs to obtain IP addresses from the same LAN
- Enable direct communication between manager and agent
- Simulate real enterprise network behavior

2. Documenting Network Configuration

2.1 IP Address Verification

On Ubuntu:

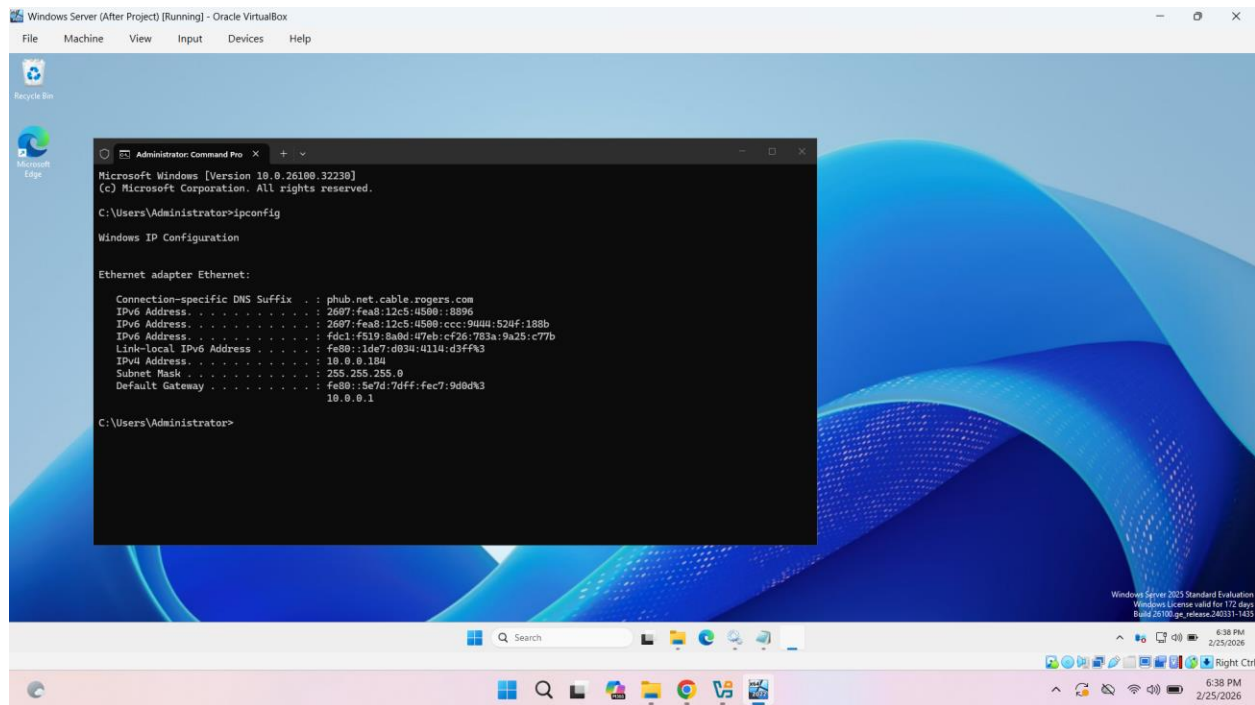
ip a



```
wale@wale:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:23:e1:ac brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.122/24 brd 10.0.0.255 scope global dynamic noprefixroute enp0s3
        valid_lft 166446sec preferred_lft 166446sec
    inet6 fdc1:f519:8a0d:47eb:b274:2b94:4db:1a75/64 scope global temporary dynamic
        valid_lft 1792sec preferred_lft 1792sec
    inet6 fdc1:f519:8a0d:47eb:a00:27ff:fe23:e1ac/64 scope global dynamic mngtnpaddr
        valid_lft 1792sec preferred_lft 1792sec
    inet6 2607:feab:12c5:4508:34c6:2c8f:abid:d48/64 scope global temporary dynamic
        valid_lft 298sec preferred_lft 298sec
    inet6 2607:feab:12c5:4508:a00:27ff:fe23:e1ac/64 scope global dynamic mngtnpaddr
        valid_lft 298sec preferred_lft 298sec
    inet6 fe80:a00:27ff:fe23:e1ac/64 scope link
        valid_lft forever preferred_lft forever
wale@wale:~$
```

On Windows:

Ipconfig



- Ubuntu Manager: 10.0.0.122
- Windows Agent: 10.0.0.184
- Both must be on the same subnet mask to communicate

2.2 Connectivity Testing

Tested communication using **ping**:

On Windows:

ping 10.0.0.122

On Ubuntu:

ping 10.0.0.184

Verified port 1514 (Wazuh agent communication port) was reachable.

3. Documenting Ubuntu Installation

3.1 ISO Installation

- Downloaded Ubuntu Server 22.04 ISO on official website
- Attached ISO in VirtualBox
- Booted VM and completed installation wizard

3.2 Initial Configuration

- Created admin user
- Configured network (automatic DHCP)

3.3 System Update

- **sudo apt update && sudo apt upgrade -y**
- **sudo apt install curl**

*Ensured system packages were fully updated before Wazuh installation.

4. Documenting Wazuh Manager Installation

4.1 Installation Method

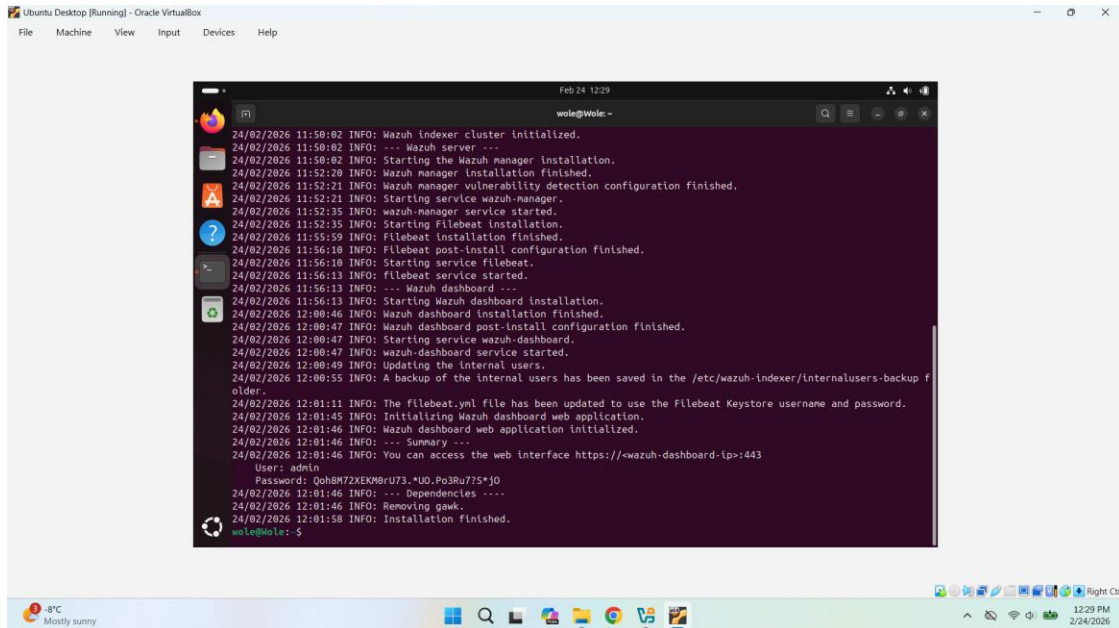
Installed Wazuh manager using official installation script from Wazuh repository.

```
curl -sO https://packages.wazuh.com/4.x/wazuh-install.sh  
sudo bash wazuh-install.sh -a
```

This installed:

- Wazuh Manager
- Wazuh Indexer
- Wazuh Dashboard

*After installation, login credentials are generated and these are used to login on the dashboard



```

Feb 24 12:29
wale@wale: ~
24/02/2026 11:58:02 INFO: Wazuh indexer cluster initialized.
24/02/2026 11:58:02 INFO: --- Wazuh server ---
24/02/2026 11:58:02 INFO: Starting the Wazuh manager installation.
24/02/2026 11:52:20 INFO: Wazuh manager installation finished.
24/02/2026 11:52:21 INFO: Wazuh manager vulnerability detection configuration finished.
24/02/2026 11:52:21 INFO: Starting service wazuh-manager.
24/02/2026 11:52:35 INFO: wazuh-manager service started.
24/02/2026 11:52:35 INFO: Starting Filebeat installation.
24/02/2026 11:55:59 INFO: Filebeat installation finished.
24/02/2026 11:56:10 INFO: Filebeat post-install configuration finished.
24/02/2026 11:56:10 INFO: Starting service filebeat.
24/02/2026 11:56:13 INFO: filebeat service started.
24/02/2026 11:56:13 INFO: --- Wazuh dashboard ---
24/02/2026 11:56:13 INFO: Starting Wazuh dashboard installation.
24/02/2026 12:00:46 INFO: Wazuh dashboard installation finished.
24/02/2026 12:00:47 INFO: Wazuh dashboard post-install configuration finished.
24/02/2026 12:00:47 INFO: Starting service wazuh-dashboard.
24/02/2026 12:00:47 INFO: wazuh-dashboard service started.
24/02/2026 12:00:49 INFO: Updating the internal users.
24/02/2026 12:00:55 INFO: A backup of the internal users has been saved in the /etc/wazuh-indexer/internalusers-backup folder.
24/02/2026 12:01:11 INFO: The filebeat.yml file has been updated to use the Filebeat Keystore username and password.
24/02/2026 12:01:45 INFO: Initializing Wazuh dashboard web application.
24/02/2026 12:01:46 INFO: Wazuh dashboard web application initialized.
24/02/2026 12:01:46 INFO: --- Summary ---
24/02/2026 12:01:46 INFO: You can access the web interface https://<wazuh-dashboard-ip>:443
User: admin
Password: Qoh8M7ZEXK8rU73.*UD.Po3Ru775*jo
24/02/2026 12:01:46 INFO: --- Dependencies ---
24/02/2026 12:01:46 INFO: Removing gawk.
24/02/2026 12:01:58 INFO: Installation finished.
wale@wale: ~
  
```

4.2 Service Verification

Checked services:

sudo systemctl status wazuh-manager

sudo systemctl status wazuh-indexer

sudo systemctl status wazuh-dashboard

```

wale@wale:~$ sudo systemctl start wazuh-indexer
wale@wale:~$ sudo systemctl start wazuh-dashboard
wale@wale:~$ sudo systemctl status wazuh-manager
● wazuh-manager.service - Wazuh manager
   Loaded: loaded (/usr/lib/systemd/system/wazuh-manager.service; enabled; pr
   Active: active (running) since Tue 2026-02-24 18:54:07 EST; 29s ago
     Process: 4155 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (co
   Tasks: 227 (limit: 7150)
   Memory: 709.6M (peak: 709.9M)
   CPU: 18.828s
   CGroup: /system.slice/wazuh-manager.service
           └─3218 /var/ossec/framework/python/bin/python3 /var/ossec/apl/so
             3219 /var/ossec/framework/python/bin/python3 /var/ossec/apl/so
             3220 /var/ossec/framework/python/bin/python3 /var/ossec/apl/so
             3223 /var/ossec/framework/python/bin/python3 /var/ossec/apl/so
             3226 /var/ossec/framework/python/bin/python3 /var/ossec/apl/so
             3267 /var/ossec/bin/wazuh-autid
             3276 /var/ossec/bin/wazuh-db
             3320 /var/ossec/bin/wazuh-execd
             3331 /var/ossec/bin/wazuh-analysisd
             3347 /var/ossec/bin/wazuh-syscheckd
             3360 /var/ossec/bin/wazuh-remoted
             3441 /var/ossec/bin/wazuh-logcollector
             4450 /var/ossec/bin/wazuh-monitord
             4459 /var/ossec/bin/wazuh-modulesd

wale@wale:~$ sudo systemctl status wazuh-dashboard
● wazuh-dashboard.service - wazuh-dashboard
   Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; pres
   Active: active (running) since Tue 2026-02-24 21:32:18 EST; 2h 37min left
     Main PID: 907 (node)
   Tasks: 11 (limit: 7158)
   Memory: 350.7M (peak: 418.6M)
   CPU: 18.548s
   CGroup: /system.slice/wazuh-dashboard.service
           └─907 /usr/share/wazuh-dashboard/node/bin/node --no-warnings --na

Feb 24 18:45:00 wale opensearch-dashboards[907]: {"type":"log","@timestamp":"2026-02-24T18:45:00.000Z","message":"Starting OpenSearch Dashboards"}

```

```

wale@wale:~$ sudo systemctl status wazuh-indexer
● wazuh-indexer.service - wazuh-indexer
   Loaded: loaded (/usr/lib/systemd/system/wazuh-indexer.service; enabled; pr
   Active: active (running) since Tue 2026-02-24 18:33:21 EST; 21min ago
     Docs: https://documentation.wazuh.com
     Main PID: 1084 (java)
   Tasks: 82 (limit: 7158)
   Memory: 1.6G (peak: 1.6G)
   CPU: 1min 25.148s
   CGroup: /system.slice/wazuh-indexer.service
           └─1084 /usr/share/wazuh-indexer/jdk/bin/java -Xshare:auto -Dopensea

Feb 24 18:45:00 wale opensearch-dashboards[907]: {"type":"log","@timestamp":"2026-02-24T18:45:00.000Z","message":"Starting OpenSearch Dashboards"}
Feb 24 18:45:00 wale opensearch-dashboards[907]: {"type":"log","@timestamp":"2026-02-24T18:45:00.000Z","message":"Starting OpenSearch Dashboards"}
Feb 24 18:45:00 wale opensearch-dashboards[907]: {"type":"log","@timestamp":"2026-02-24T18:45:00.000Z","message":"Starting OpenSearch Dashboards"}
Feb 24 18:45:00 wale opensearch-dashboards[907]: {"type":"log","@timestamp":"2026-02-24T18:45:00.000Z","message":"Starting OpenSearch Dashboards"}
Notice: Journal has been rotated since unit was started, output may be incomplete.

wale@wale:~$ sudo systemctl status wazuh-indexer
● wazuh-indexer.service - wazuh-indexer
   Loaded: loaded (/usr/lib/systemd/system/wazuh-indexer.service; enabled; pr
   Active: active (running) since Tue 2026-02-24 18:33:21 EST; 21min ago
     Docs: https://documentation.wazuh.com
     Main PID: 1084 (java)
   Tasks: 82 (limit: 7158)
   Memory: 1.6G (peak: 1.6G)
   CPU: 1min 25.148s
   CGroup: /system.slice/wazuh-indexer.service
           └─1084 /usr/share/wazuh-indexer/jdk/bin/java -Xshare:auto -Dopensea

Notice: Journal has been rotated since unit was started, output may be incomplete.
wale@wale:~$ nc -zv 10.0.0.184 3389
nc: connect to 10.0.0.184 port 3389 (tcp) failed: connection refused
wale@wale:~$

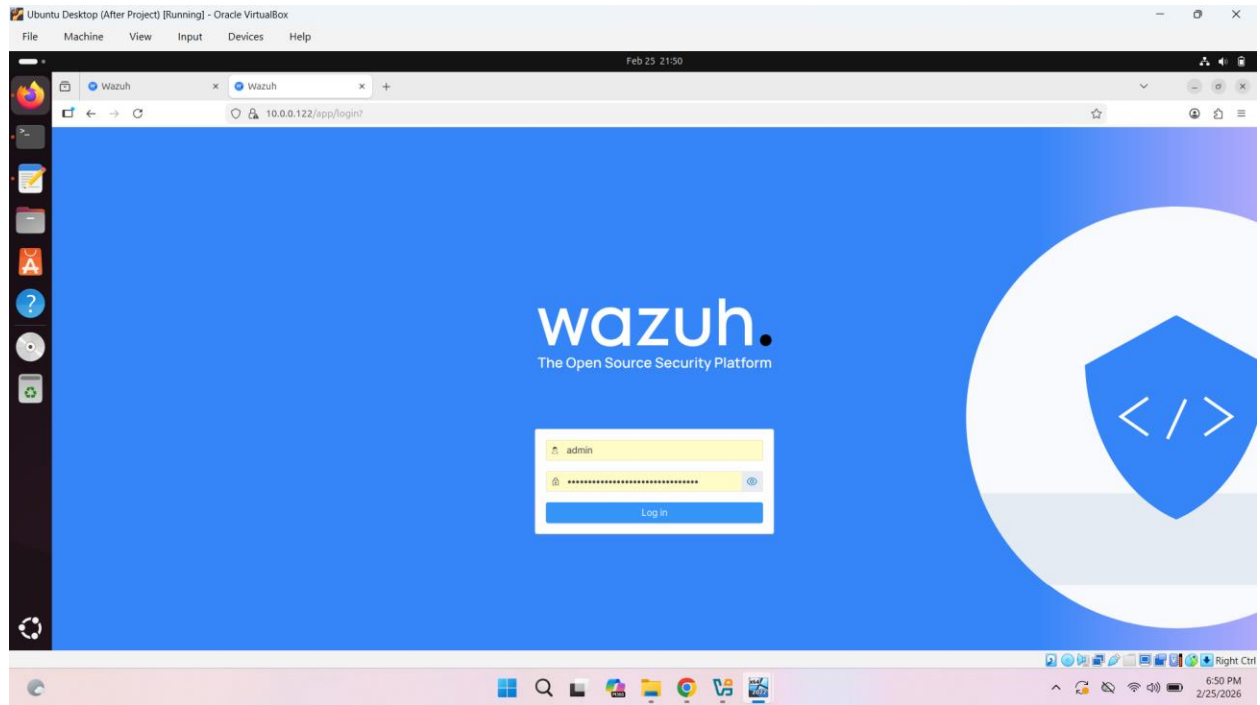
```

* Confirmed that all services were running.

4.3 Dashboard Access

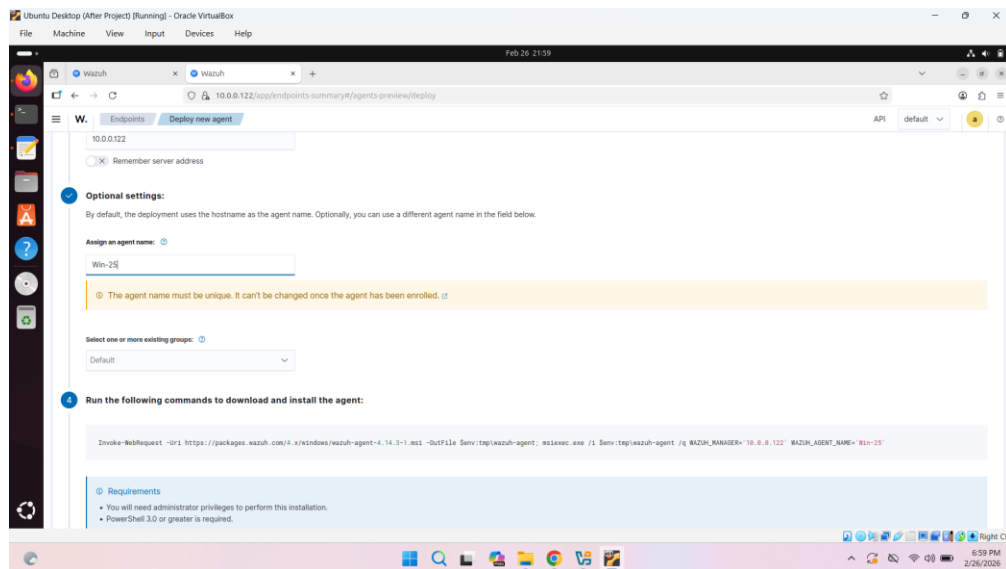
Accessed dashboard via:

<https://10.0.0.122>



Logged in using generated credentials.

4.4 Agent Deployment



Deployed a new agent on Windows server using the command from the Wazuh dashboard on Powershell

Invoke-WebRequest -Uri

**`https://packages.wazuh.com/4.x/windows/wazuh-agent-4.14.3-1.msi -
OutFile $env:tmp\wazuh-agent; msisexec.exe /i $env:tmp\wazuh-agent /q
WAZUH_MANAGER='10.0.0.122' WAZUH_AGENT_NAME='Win-25'`**

Architecture Explanation

The Ubuntu Server hosts the Wazuh Manager, which receives logs and FIM alerts from the Windows Server agent over TCP port 1514. Alerts are indexed and visualized via the Wazuh Dashboard.

Design Considerations

- Allocated sufficient RAM and Storage to prevent indexer crash
- Used bridged networking to simulate enterprise deployment
- Reduced FIM scan interval to 60 seconds for rapid lab validation
- Segmented monitoring responsibilities between manager and agent

5. Configuration

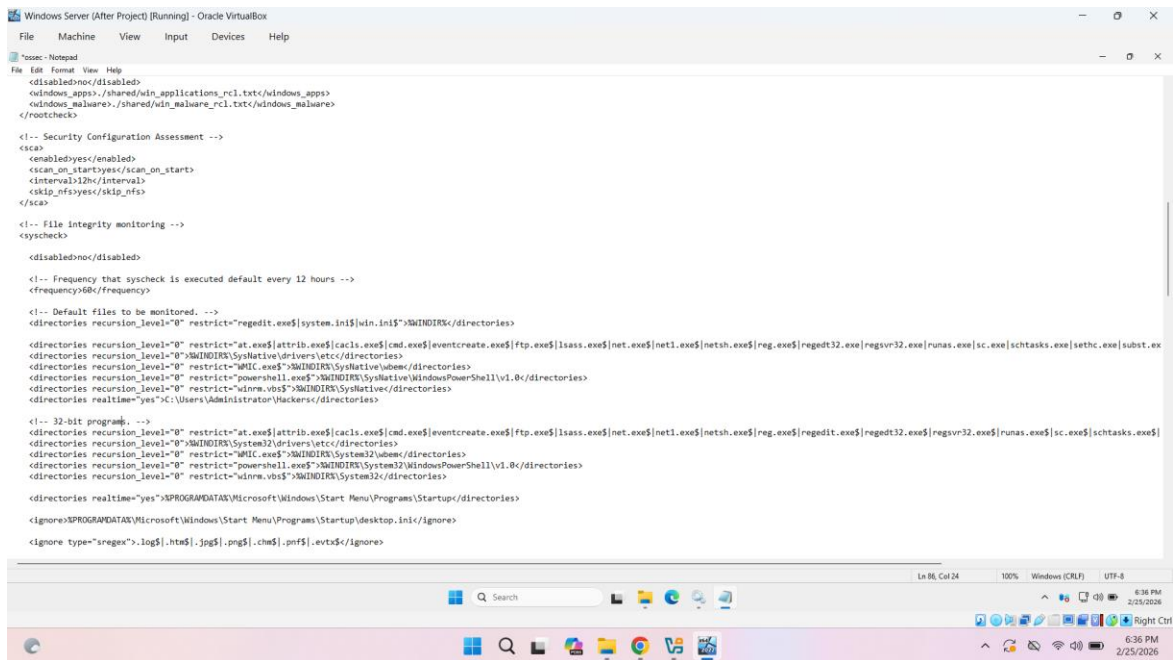
<syscheck>

<disabled>no</disabled>

<frequency>60</frequency>

<directories realtime="yes">C:\Users\Administrator\Hackers</directories>

<syscheck>



```
File Edit Format View Help
File Edit Format View Help
<!-- Security Configuration Assessment -->
<scap>
  <enabled>yes</enabled>
  <scan_on_start>yes</scan_on_start>
  <interval>12h</interval>
  <skip_nfs>yes</skip_nfs>
</scap>

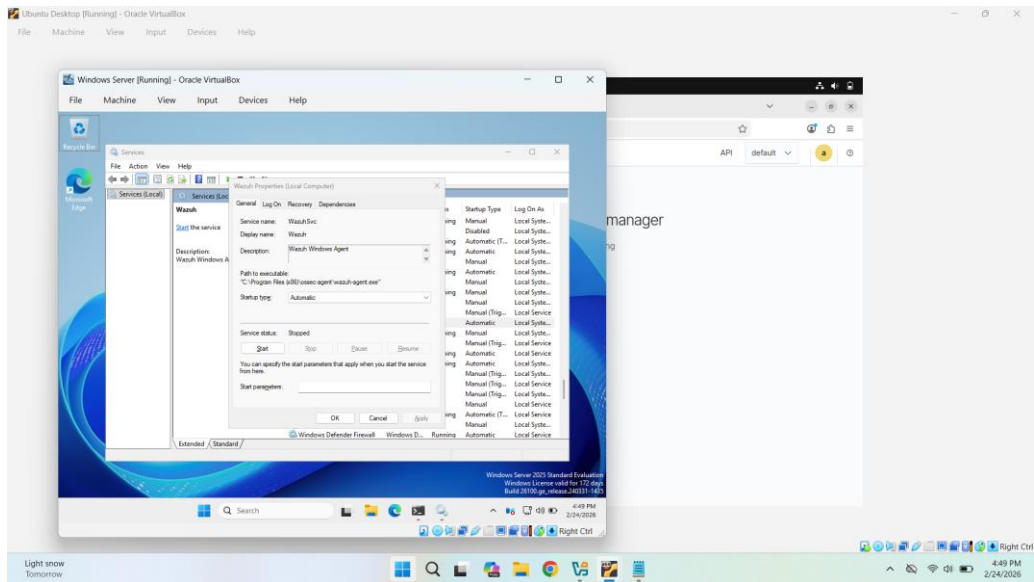
<!-- File Integrity Monitoring -->
<syscheck>

  <disabled>no</disabled>

  <!-- Frequency that syscheck is executed default every 12 hours -->
  <frequency>60</frequency>

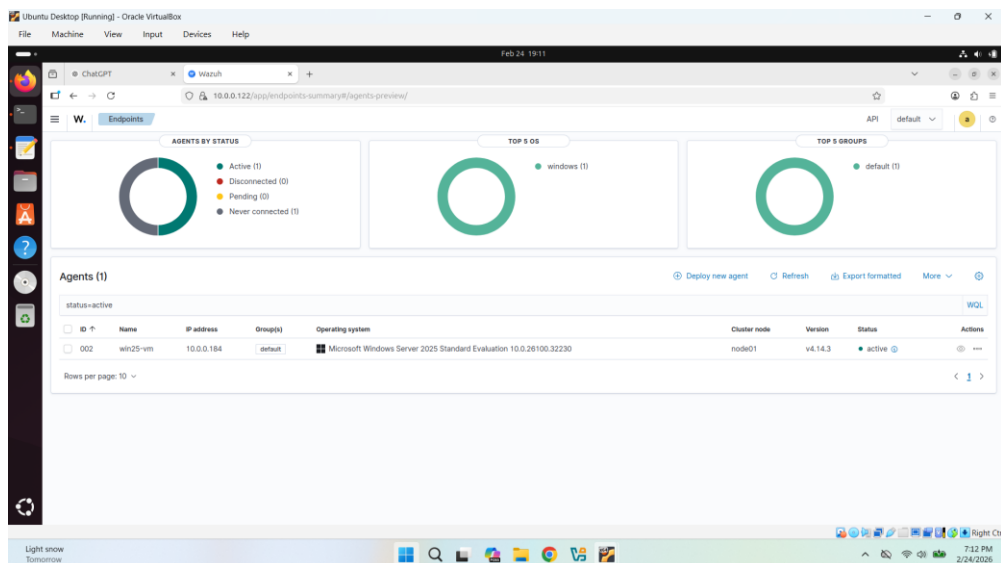
  <!-- Default files to be monitored -->
  <directories recursion_level="0" restrict="regedit.exe|system.ini|win.ini">%WINDIR%\directories>
  <directories recursion_level="0" restrict="at.exe|attrib.exe|cactls.exe|cmd.exe|eventcreate.exe|ftp.exe|lsass.exe|net.exe|net1.exe|netsh.exe|reg.exe|regedit32.exe|regsvr32.exe|runas.exe|sc.exe|schtasks.exe|sethc.exe|subst.exe|wmic.exe">%WINDIR%\System32\drivers\etc\directories>
  <directories recursion_level="0" restrict="WMIC.exe">%WINDIR%\System32\wbem\directories>
  <directories recursion_level="0" restrict="powershell.exe">%WINDIR%\System32\WindowsPowerShell\v1.0\directories>
  <directories recursion_level="0" restrict="winrm.vbs">%WINDIR%\System32\WindowsPowerShell\v1.0\directories>
  <directories realtime="yes">C:\Users\Administrator\Hackers</directories>

  <!-- 32-bit programs -->
  <directories recursion_level="0" restrict="at.exe|attrib.exe|cactls.exe|cmd.exe|eventcreate.exe|ftp.exe|lsass.exe|net.exe|net1.exe|netsh.exe|reg.exe|regedit.exe|regedit32.exe|regsvr32.exe|runas.exe|sc.exe|schtasks.exe|wmic.exe">%PROGRAMDATA%\Microsoft\Windows\Start Menu\Programs\Startup\directories>
  <directories recursion_level="0" restrict="WMIC.exe">%WINDIR%\System32\wbem\directories>
  <directories recursion_level="0" restrict="powershell.exe">%WINDIR%\System32\WindowsPowerShell\v1.0\directories>
  <directories recursion_level="0" restrict="winrm.vbs">%WINDIR%\System32\WindowsPowerShell\v1.0\directories>
  <directories realtime="yes">%PROGRAMDATA%\Microsoft\Windows\Start Menu\Programs\Startup\desktop.ini</ignore>
  <ignore type="sregex">*.log$|.hta$|.jpg$|.png$|.chm$|.pif$|.evtx</ignore>
```

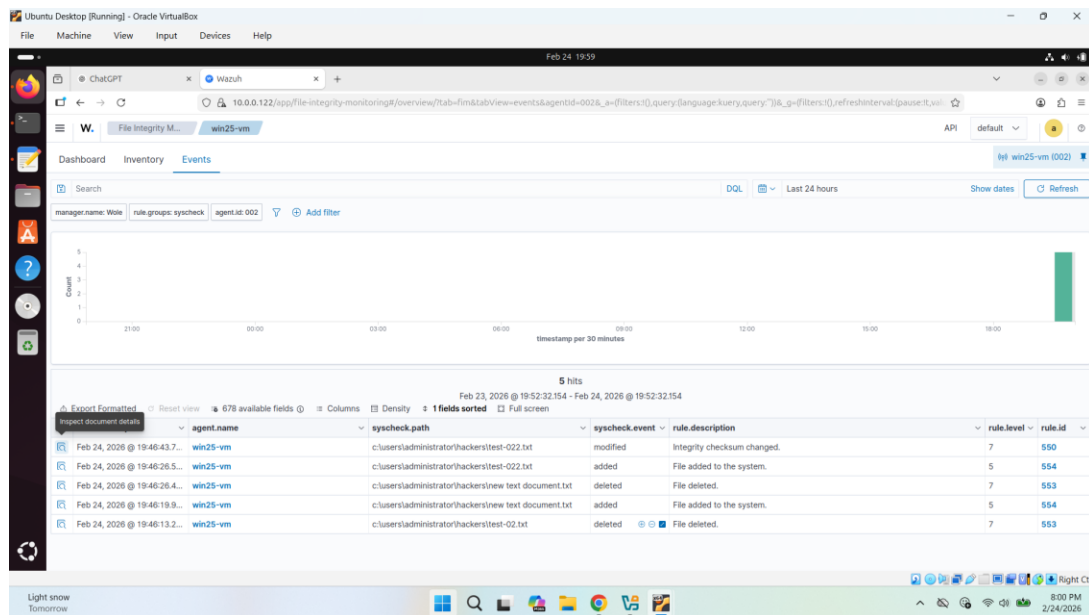


- Scan frequency set to 60 seconds so alerts can be quickly generated when changes are made to a file
- New Directory was created for solely monitoring the changes made to it
- Restarted Wazuh service so Wazuh reloads updated **ossec.conf** file

6. Testing & Validation



Dashboard showing Agent is active



File Creation

When a new file is created in a monitored directory, **Wazuh** detects that the file did not exist in its previous baseline scan. It records the file's metadata and hash, then generates an **“added”** alert as shown in the screenshot above.

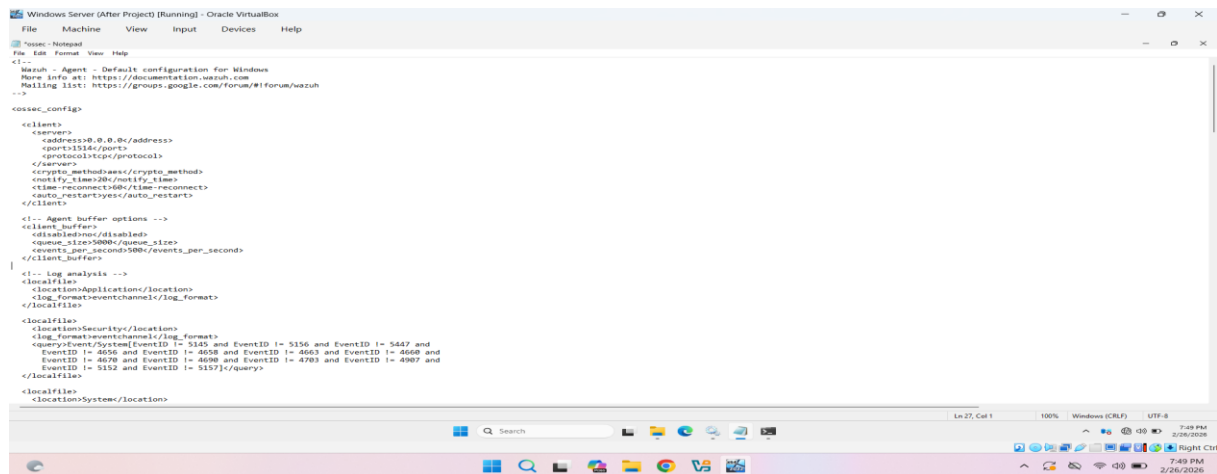
File Modification

When a monitored file is changed, Wazuh recalculates its hash and compares it to the stored baseline value. If the hash or attributes differ, it generates a **“modified”** alert indicating possible tampering as shown in the screenshot above.

File Deletion

When a previously recorded file is removed, Wazuh detects that it is missing during the next scan. It then generates a **“deleted”** alert showing the file path and prior recorded details as shown in the screenshot above.

7. Troubleshooting



At first, the deployed agent wasn't showing up on the Wazuh dashboard; it was because the IP address in the ossec file was set to 0.0.0.0 by default. Edited the IP address to 10.0.0.122, the Ubuntu Manager address to fix this.

Conclusion

What I Learned

I learned how to configure and validate File Integrity Monitoring using **Wazuh**, including editing the <syscheck> configuration and adjusting scan frequency. I also learned how file hashes, timestamps, and baselines are used to detect changes and generate security alerts.

Additionally, I gained hands-on experience analyzing “added,” “modified,” and “deleted” alerts in the Wazuh dashboard, which improved my understanding of how file-level monitoring works in a SIEM environment.

Why FIM Matters in Real SOC Environments

File Integrity Monitoring is critical because attackers often modify, create, or delete files after gaining access to a system. Monitoring these changes helps SOC analysts detect malware drops, configuration tampering, web shell uploads, and log deletion attempts.

In real SOC environments, FIM provides visibility into unauthorized system changes and supports compliance requirements (such as monitoring sensitive system files). It acts as an early warning system for post-compromise activity.

How This Simulates Real-World Attack Detection

By creating, modifying, and deleting test files, I simulated behaviors commonly seen during real cyberattacks. For example, attackers may create malicious scripts, modify configuration files for persistence, or delete logs to cover their tracks.

This lab demonstrates how a SOC team would receive alerts from Wazuh and investigate suspicious file activity, replicating real-world detection, and response workflows.