

Linux File Integrity Monitoring using Wazuh

BY: GOKUL KRISHNA K

OBJECTIVE

The objective of this project is to implement a File Integrity Monitoring (FIM) system on a Linux environment using Wazuh to detect, track, and alert on any unauthorized or suspicious changes made to critical files and directories. This project aims to strengthen system security by ensuring that any modification to important files is immediately identified, logged, and investigated.

TOOLS USED

UBUNTU SERVER

UBUNTU DESKTOP

VRITUALBOX

WAZUH SERVER

WAZUH AGENT

Wazuh Server Setup

```
GNU nano 6.2                               /var/ossec/etc/ossec.conf *
<!-- Wazuh - Manager - Default configuration for ubuntu 22.04
  More info at: https://documentation.wazuh.com
  Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
  <global>
    <json_out></json_out>
    <alerts_log></alerts_log>
    <logall></logall>
    <logall_json>yes</logall_json>
    <email_notification>no</email_notification>
    <sntp_server>ntp://ntp.wazuh.com:123</sntp_server>
    <email_to>wazuh@example.wazuh.com</email_to>
    <email_from>wazuh@example.wazuh.com</email_from>
    <email_maxperhour>12</email_maxperhour>
    <email_source>alerts.log</email_log_source>
    <agents_disconnect_time>10</agents_disconnect_time>
    <agents_disconnection_alert_time>10</agents_disconnection_alert_time>
    <update_check>yes</update_check>
  </global>

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

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

  <remote>
    <connection>secure</connection>
    <port>1514</port>
    <proto>tcp</proto>
    <queue_size>131072</queue_size>
  </remote>

  <!-- Policy monitoring -->
  <rontcheck>
    <disabled>no</disabled>
    <check_files>yes</check_files>
    <check_procs>yes</check_procs>
    <check_trjans>yes</check_trjans>
    <check_devs>yes</check_devs>
    <check_sys>yes</check_sys>
  </rontcheck>
</ossec_config>
```

I opened server configuration file for some modifications and I enabled both `<logall>` and `<logall_json>` in the Wazuh configuration to ensure complete logging and structured storage of all events. `logall` allows the agent to record every activity, which is essential for detailed File Integrity Monitoring. `logall_json` stores these logs in JSON format, making them easier to analyze, parse, and visualize in the Wazuh dashboard or any SIEM.

Command: `sudo nano /var/ossec/etc/ossec.conf`

1. `<logall>yes</logall>` — Enables Full Logging

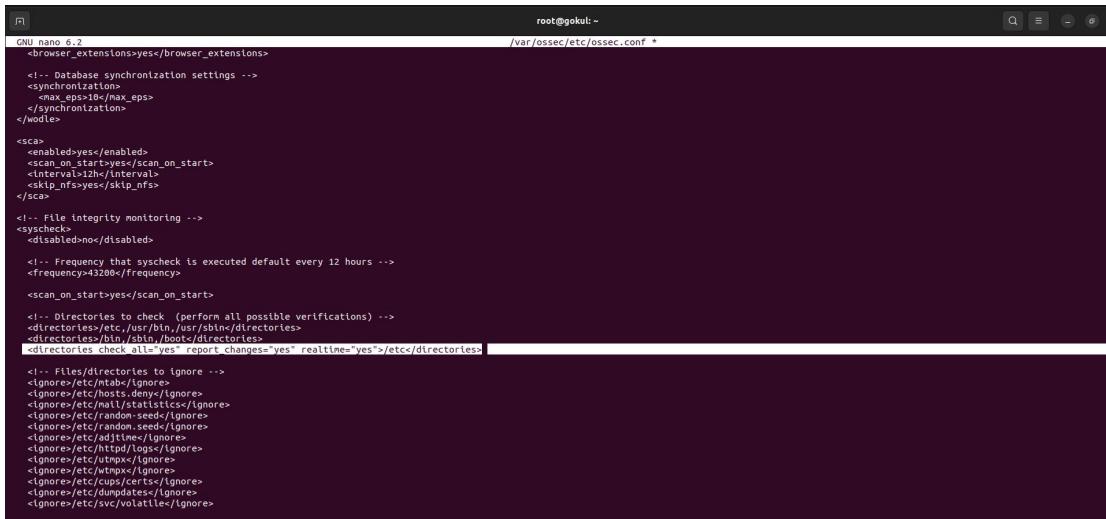
2. `<logall_json>yes</logall_json>` — Enables JSON Format Logging

```
root@gokul:/# systemctl restart wazuh-manager
root@gokul:/#
```

I restarted wazuh manager

Command: sudo systemctl restart wazuh-manager

Wazuh Agent Setup



```
GNU nano 6.2
<browser_extensions>yes</browser_extensions>
<!-- Database synchronization settings -->
<synchronization>
  <max_eps>10</max_eps>
</synchronization>
</wodle>
<scas>
  <enabled>yes</enabled>
  <scan_on_start>yes</scan_on_start>
  <interval>12h</interval>
  <skipt_nfs>yes</skipt_nfs>
</scas>
<!-- File integrity monitoring -->
<syscheck>
  <disabled>no</disabled>
  <!-- Frequency that syscheck is executed default every 12 hours -->
  <frequency>43200</frequency>
  <scan_on_start>yes</scan_on_start>
  <!-- Directories to check (perform all possible verifications) -->
<directories></etc>,</usr/bin>,</usr/sbin>,</directories>
  <directories></etc>,</var>,</sbin>,</boot>,</directories>
  <!-- Directories to ignore -->
  <ignore></etc>,</hosts>,</etc/deny>
  <ignore></etc>,</mail/statistics>
  <ignore></etc>,</random>,</seed>
  <ignore></etc>,</random>,</seed>
  <ignore></etc>,</utmp>
  <ignore></etc>,</utmpx>
  <ignore></etc>,</utmpx>
  <ignore></etc>,</cups>,</certs>
  <ignore></etc>,</updates>
  <ignore></etc>,</xvc>,</volatile>
```

I opened the agent configuration file same like server configuration file and I added `<directories check_all="yes" report_changes="yes" realtime="yes"/>` inside configuration file of wazuh agent. This configuration line in Wazuh File Integrity Monitoring (FIM) tells Wazuh to monitor the `/etc` directory for security-related file changes. This directive is used to enable File Integrity Monitoring on the `/etc` directory.

It instructs Wazuh to:

`check_all="yes"` → Monitor all types of changes, including file creation, deletion, modification, and permission changes.

`report_changes="yes"` → Generate detailed reports showing what exactly changed inside a file (diff output).

`realtime="yes"` → Detect and alert on changes immediately using real-time monitoring instead of periodic scans.

By applying this configuration, Wazuh continuously tracks any unauthorized or suspicious modifications in the `/etc` directory, which is one of the most critical locations in a Linux system because it contains system configurations, user account details, and security settings.

```
Command : <directories check_all="yes" report_changes="yes"  
realtime="yes">/etc</directories>
```

```
root@gokul:~# systemctl restart wazuh-agent  
root@gokul:~#
```

I restart wazuh agent for getting fresh start

Command: sudo systemctl restart wazuh-agent

My First Attack Simulation

Append a fake root user into /etc/passwd:

```
root@gokul:~# echo "hacker:x:0:0:Hacker:/root:/bin/bash" | sudo tee -a /etc/passwd  
hacker:x:0:0:Hacker:/root:/bin/bash  
root@gokul:~#
```

I had run the echo "hacker:x:0:0:Hacker:/root:/bin/bash" | sudo tee -a /etc/passwd command this is for adds a malicious backdoor user with root privileges, which is commonly used in attacks. This will trigger Wazuh FIM alerts because /etc/passwd is a monitored file and the system's integrity is compromised.

Command: echo "hacker:x:0:0:Hacker:/root:/bin/bash" | sudo tee -a /etc/passwd

My Second Attack Simulation

Modify SSH configuration to allow root login:

```
root@gokul:~# echo "PermitRootLogin yes" | sudo tee -a /etc/ssh/sshd_config
PermitRootLogin yes
root@gokul:~#
```

It enables remote login as root using SSH (which is normally disabled for security reasons). I appended the line PermitRootLogin yes to the SSH configuration file /etc/ssh/sshd_config.

This change enables SSH login for the root account, which is a high-risk security setting. Wazuh File Integrity Monitoring detects this modification and generates a security alert indicating that a critical system configuration file was altered.

Command: echo "PermitRootLogin yes" | sudo tee -a /etc/ssh/sshd_config

I accessed the Wazuh server to review the security alerts triggered by the attack I performed

The screenshot shows the Wazuh MITRE ATT&CK interface. The top navigation bar includes 'Dashboard', 'Intelligence', 'Framework', and 'Events'. The 'Events' tab is selected. A search bar at the top has the query 'manager.name:gokul AND rule.mitre.id:events AND agent.id:001'. Below the search bar is a timeline from Dec 9, 2023, to Dec 10, 2023, with a count of 6 events. The event list table has columns for timestamp, agent.name, rule.mitre.id, and rule.mitre.tactic. The first event listed is a sudo command executed by user 'root' on 'gokul' at 22:19:53 on Dec 10, 2023. The 'Document Details' panel on the right shows the JSON structure of the event, including fields like '_index', 'agent.id', 'agent.ip', 'agent.name', 'data.command', 'data.dstuser', 'data.pwd', 'data.srouser', 'data.tty', 'decoder.ftcomment', 'decoder.name', 'decoder.parent', and 'full_log'. The 'full_log' field contains the command: 'Dec 10 22:19:53 [gokul sudo] root : TTY pts/0 ; PWD=/root ; USER=root ; COMMAND=echo "PermitRootLogin yes" | sudo tee -a /etc/ssh/sshd_config'.

Both alerts were visible on the Wazuh server during my analysis.

CONCLUSION

The implementation of File Integrity Monitoring (FIM) using Wazuh successfully demonstrated how critical system files in a Linux environment can be continuously monitored for unauthorized changes. By configuring Wazuh agents to track directories such as /etc, the system was able to detect and alert on high-risk activities, including modifications to /etc/passwd and /etc/ssh/sshd_config. These actions simulated real-world attack scenarios such as privilege escalation and insecure configuration changes.