Sysmon Integration with Wazuh

Sysmon:

Sysmon (System Monitor) is a Windows service developed by Microsoft that continuously monitors internal system activity and generates high-detail security logs.

> Default Windows logs tell you *that* something happened.
> Sysmon logs explain *what happened, how it happened, who initiated it, and what followed*.

Sysmon captures low-level system behavior that attackers commonly abuse, such as process execution, parent-child relationships, network connections, file creation, and registry changes details that standard Windows logging often misses.

Because of this depth, Sysmon provides visibility before, during, and after an attack, rather than showing only a single command or isolated event.

Why Wazuh with Sysmon:

Sysmon alone is a powerful log generator, but it cannot analyze or alert on events by itself. Wazuh provides the missing capabilities: log collection, processing, correlation, and alerting.

Wazuh Agent

> Collects logs from Sysmon
>
> Sends logs securely to the Wazuh Manager.
>
> Can apply local file integrity checks and basic analysis.

Wazuh Manager

> Receives logs from agents.
>
> Decodes and normalizes Sysmon events.
>
> Correlates events across endpoints to identify suspicious activity.
>
> Generates alerts and visualizes attack timelines in the Wazuh Dashboard for SOC analysts.

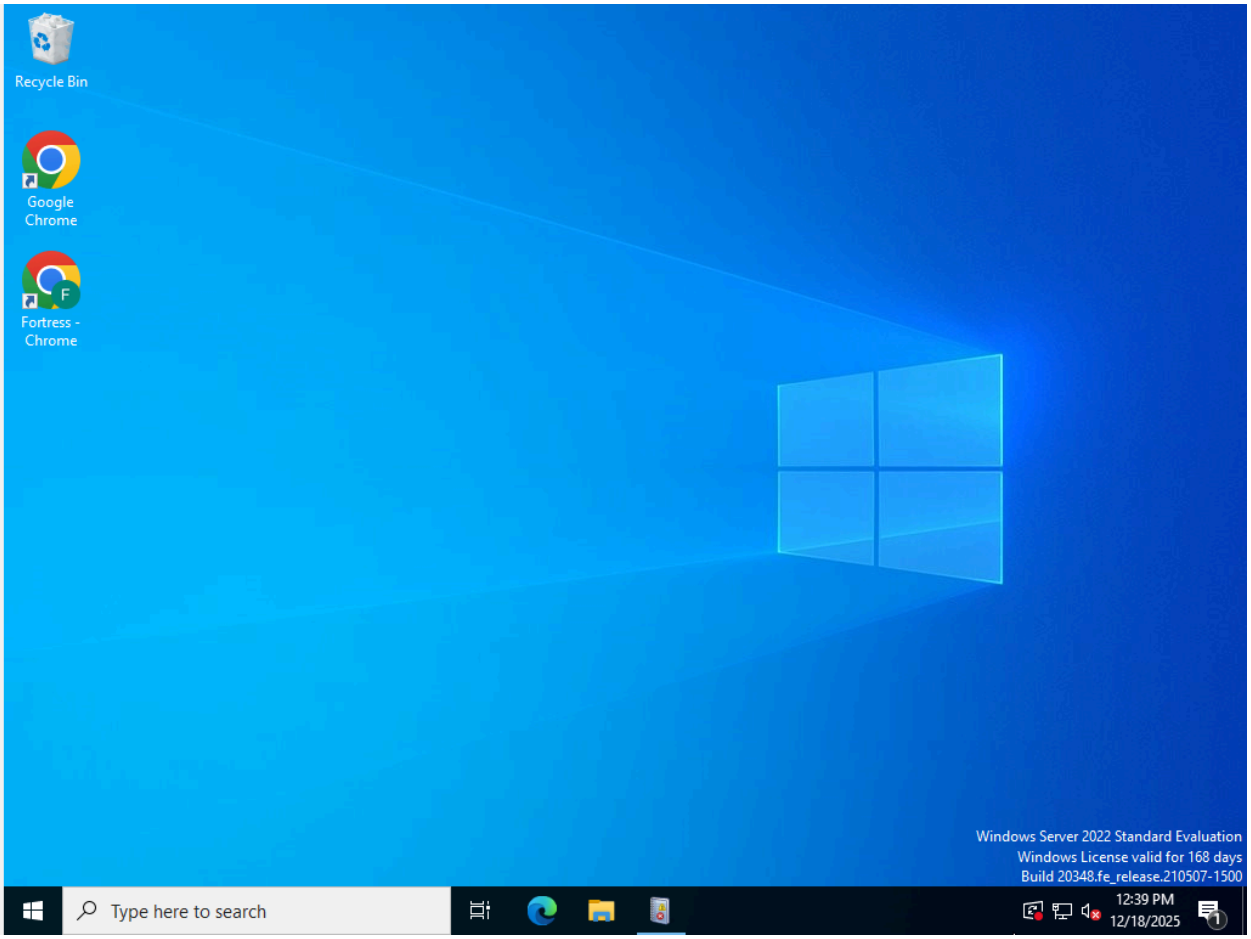
Environment Overview:

Wazuh Agent: Windows Server 2022 (monitors Sysmon logs)

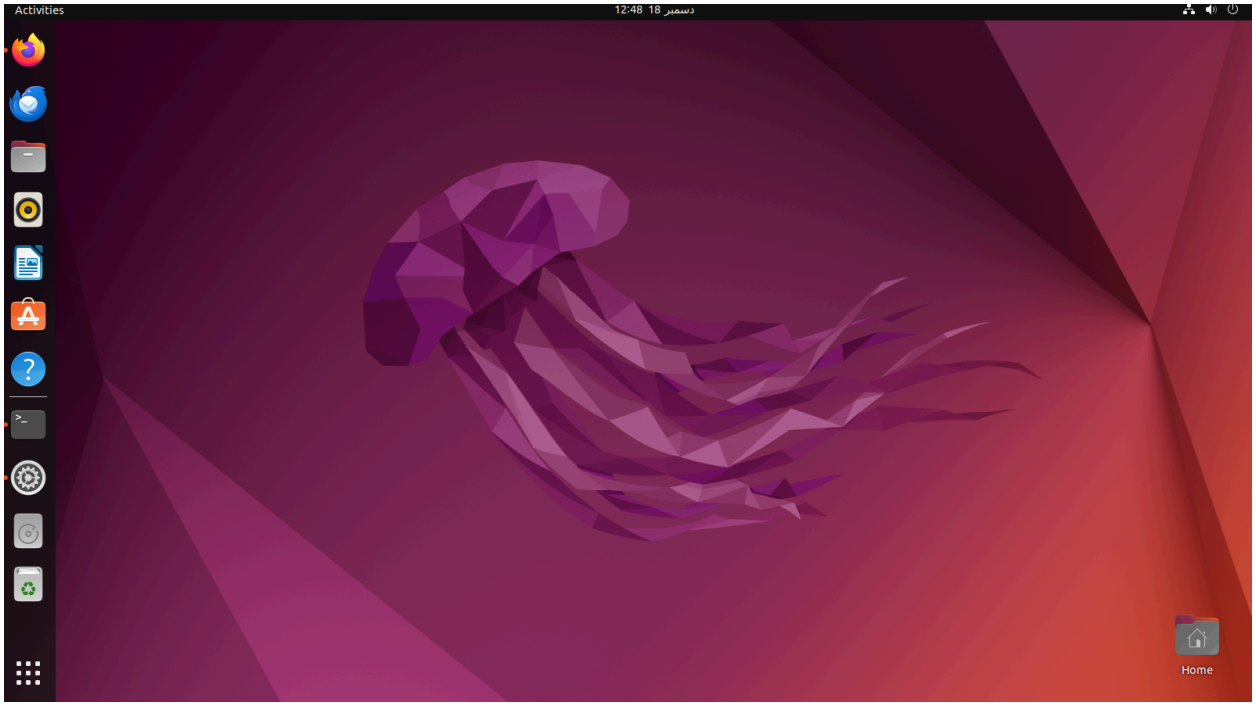
Wazuh Manager + Indexer + Dashboard: Ubuntu Desktop (central server, receives logs, visualizes alerts)
Agent collects Sysmon logs and sends them securely to the Manager.
Manager decodes, correlates, and displays events in the Dashboard.

Step by Step Methodology:

The first step is to ensure that the Wazuh Agent can communicate with the Wazuh Manager.
This is our Window Server where our Agent is installed.

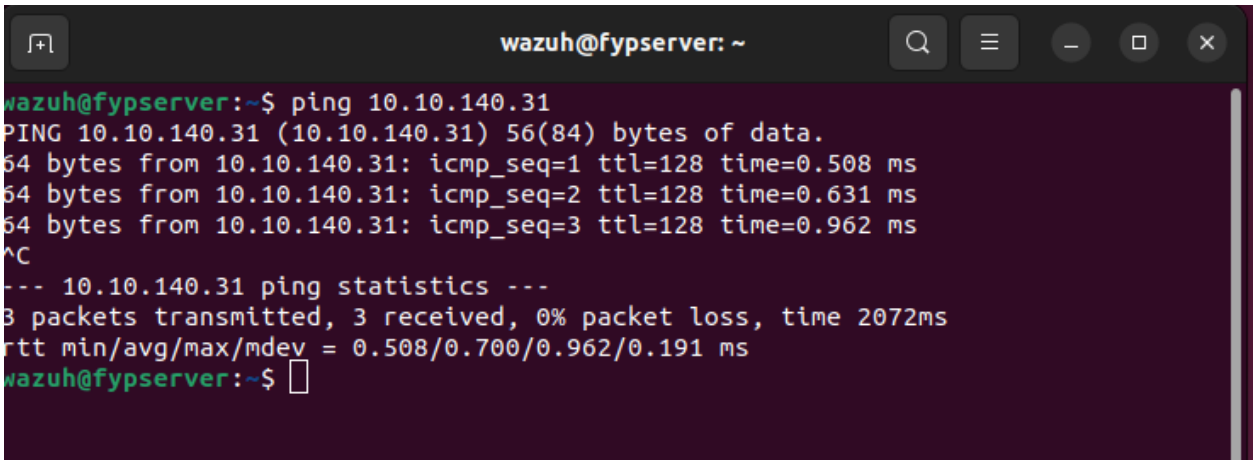


And this is our Ubuntu Desktop based where our Wazuh Manager with other components is installed.

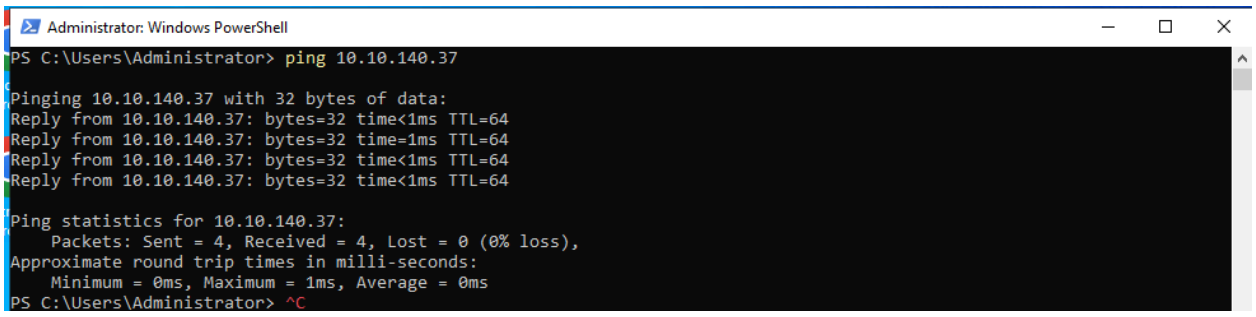


Use the ping command to verify network communication between the machines:

First Ping from Ubuntu Desktop to Windows Server is successful.

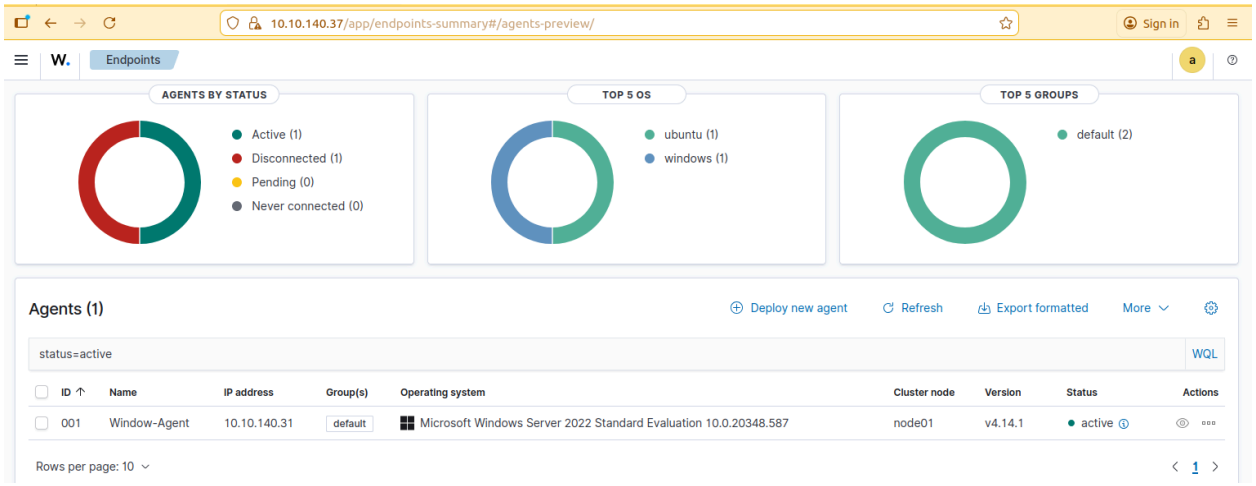


Ping from Windows Server to Ubuntu Desktop is successful.



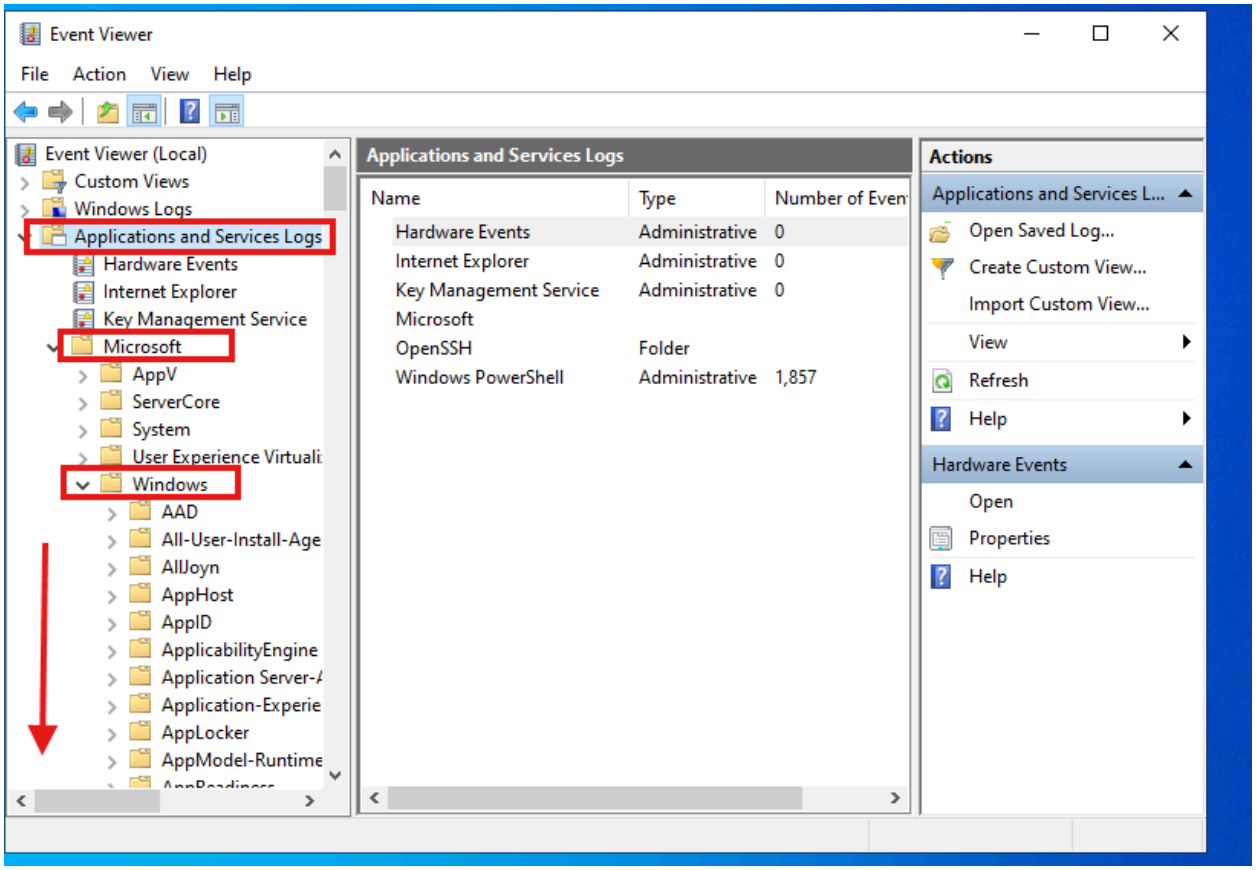
Both machines can communicate, confirming that the network connection between the Agent and Manager is working correctly.

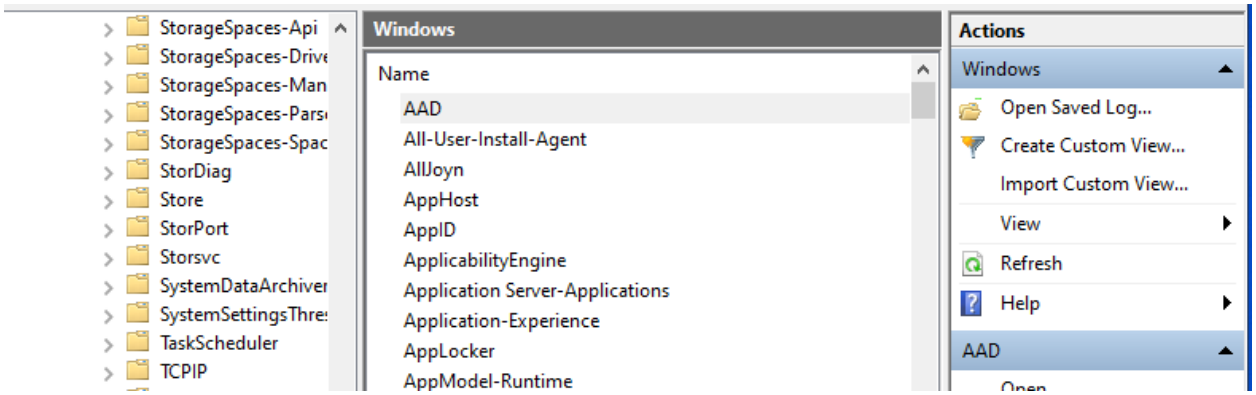
And in the Dashboard for Agent status is Active:



Step1:

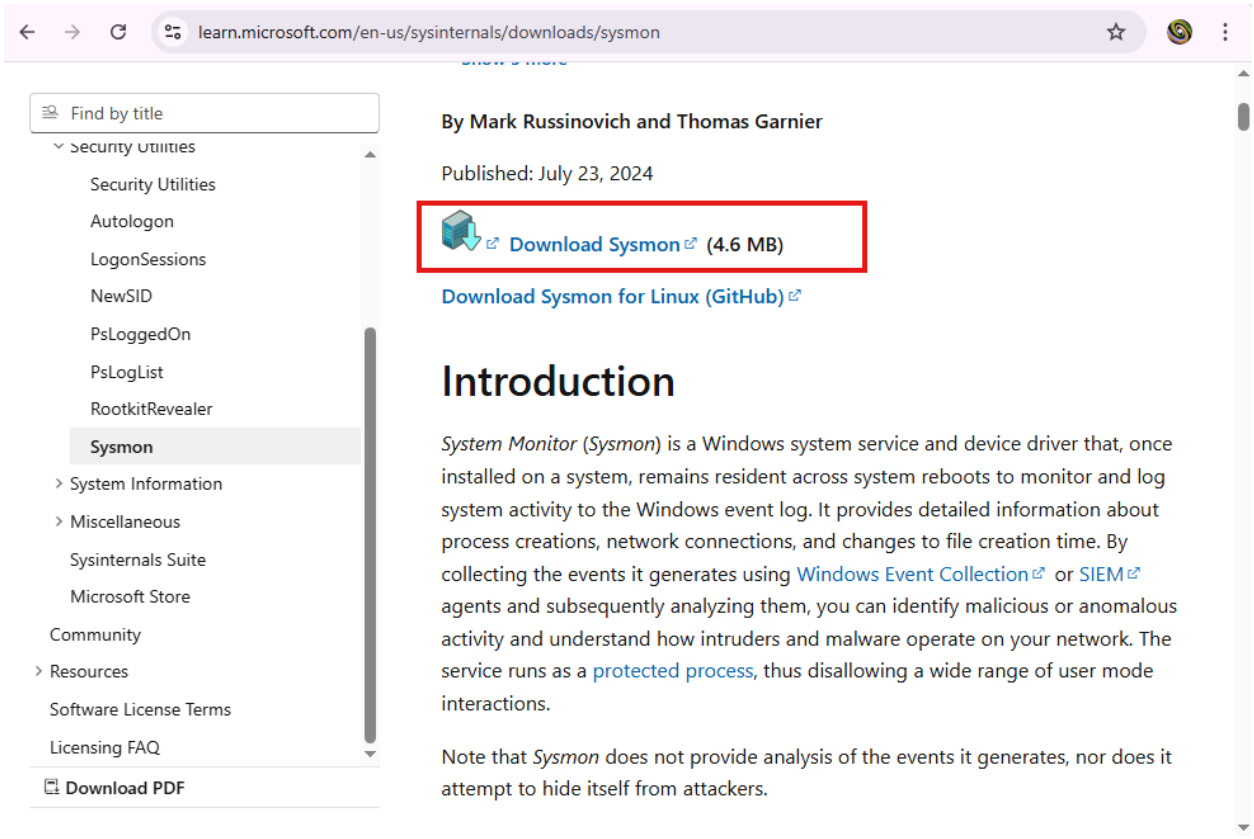
Opened the Event Viewer. In the left panel, go to Application and Services Logs > Microsoft > Windows. Scroll down the list and check. And see that Sysmon is not present by default.



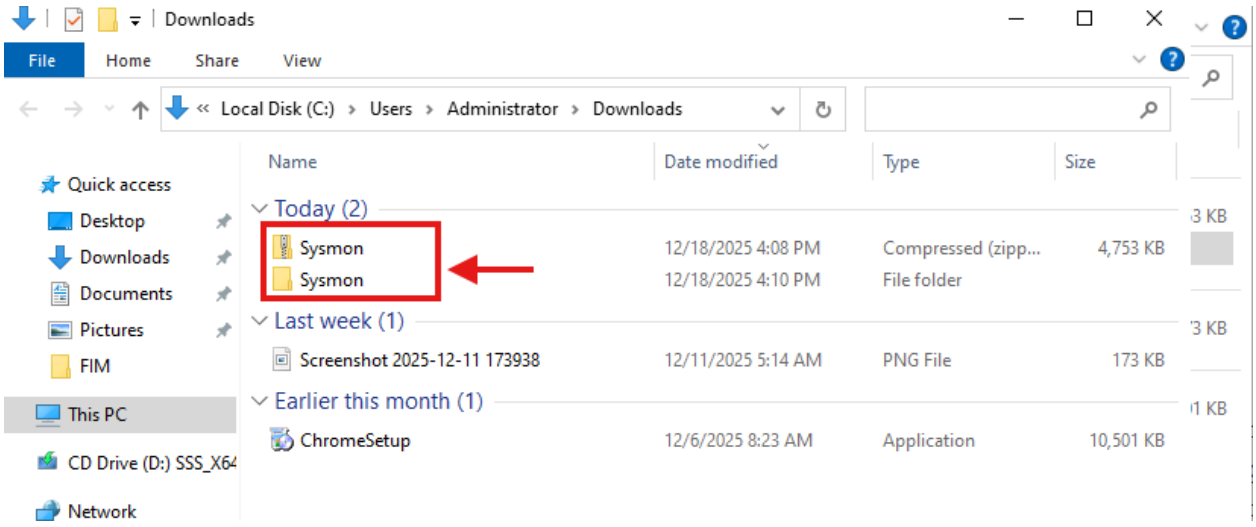


Sysmon is not installed here and so downloaded it from the following official Microsoft source:

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



After downloading the Sysmon, extract the folder.

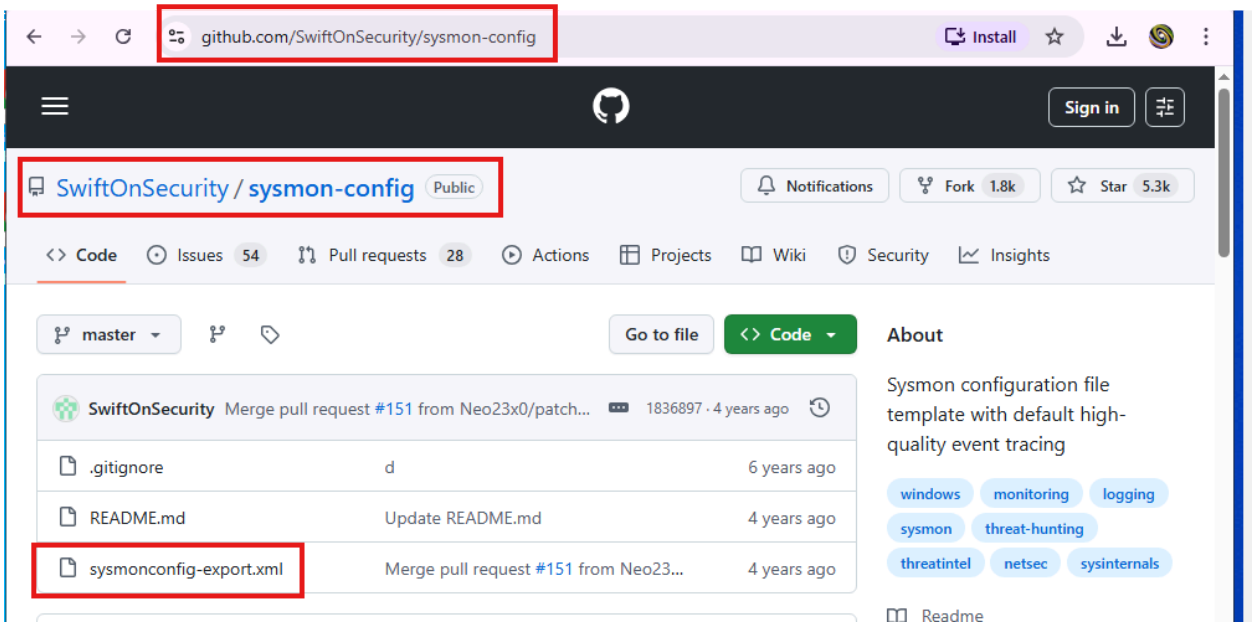


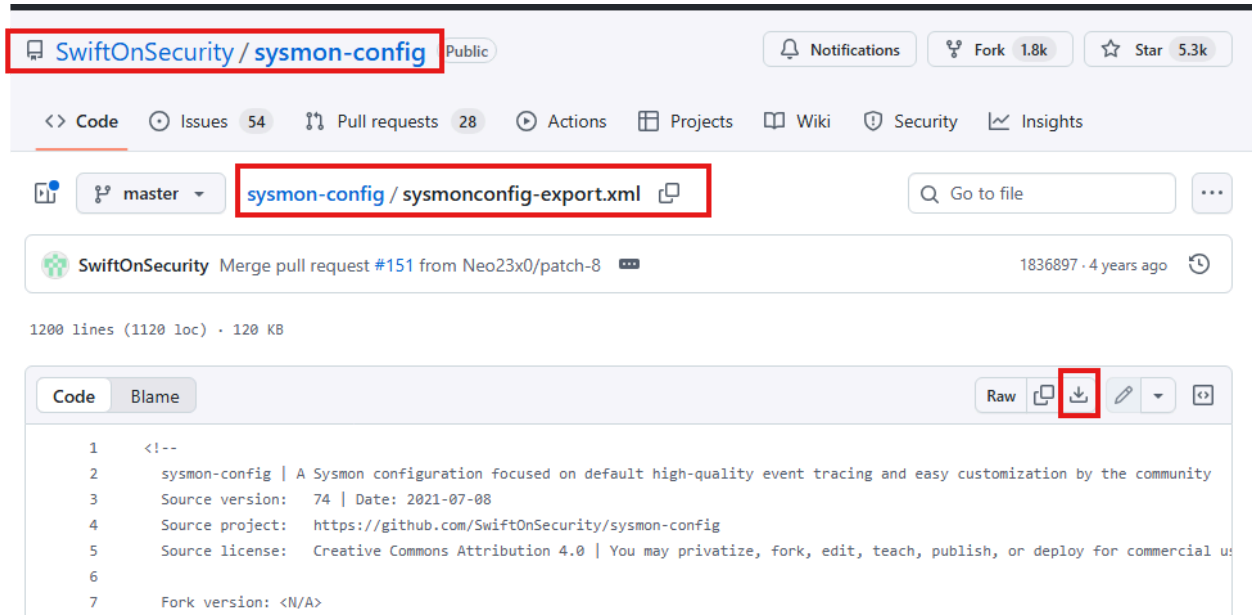
After downloading Sysmon, we also need to download its configuration file from GitHub. The configuration file helps Sysmon know what events to monitor and log.

We can get the official Sysmon configuration file from the following link:

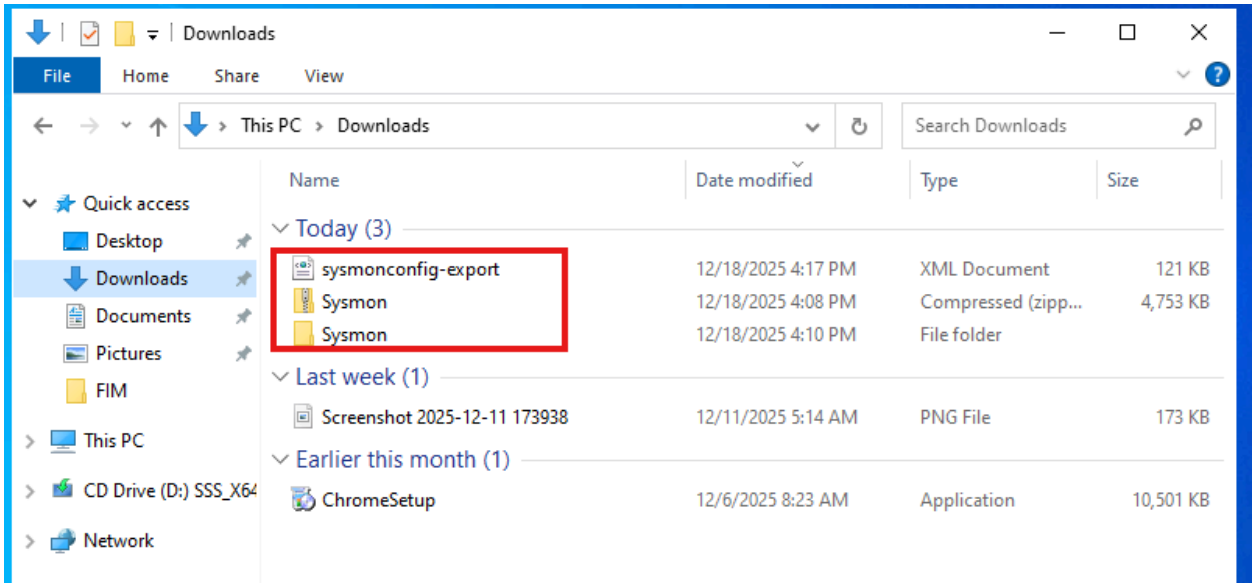
https://github.com/SwiftOnSecurity/sysmon-config

Now, download the “sysmonconfig-export.xml” file.

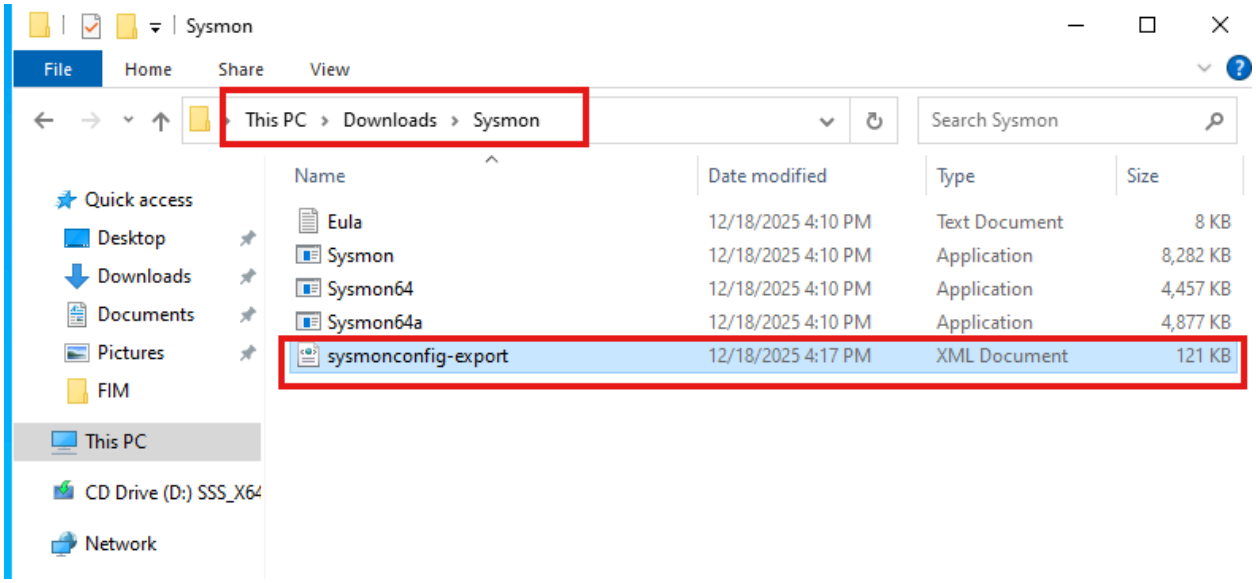




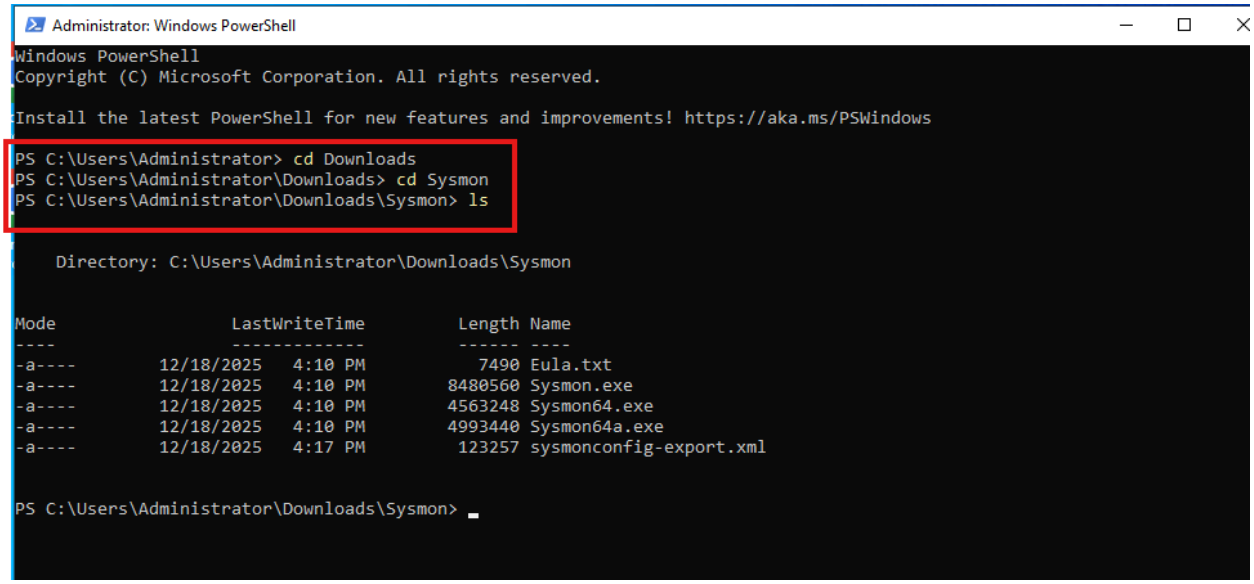
After this step, both Sysmon and the **"**sysmonconfig-export.xml**"** file will be downloaded successfully.



Now, paste the configuration file into the Sysmon folder.

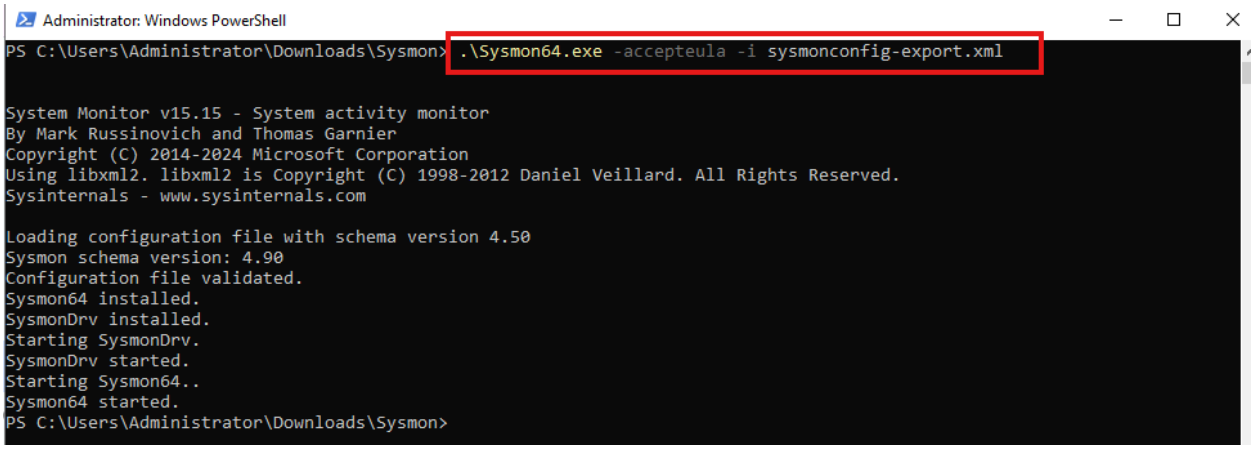


Run PowerShell as Administrator in the Sysmon folder:



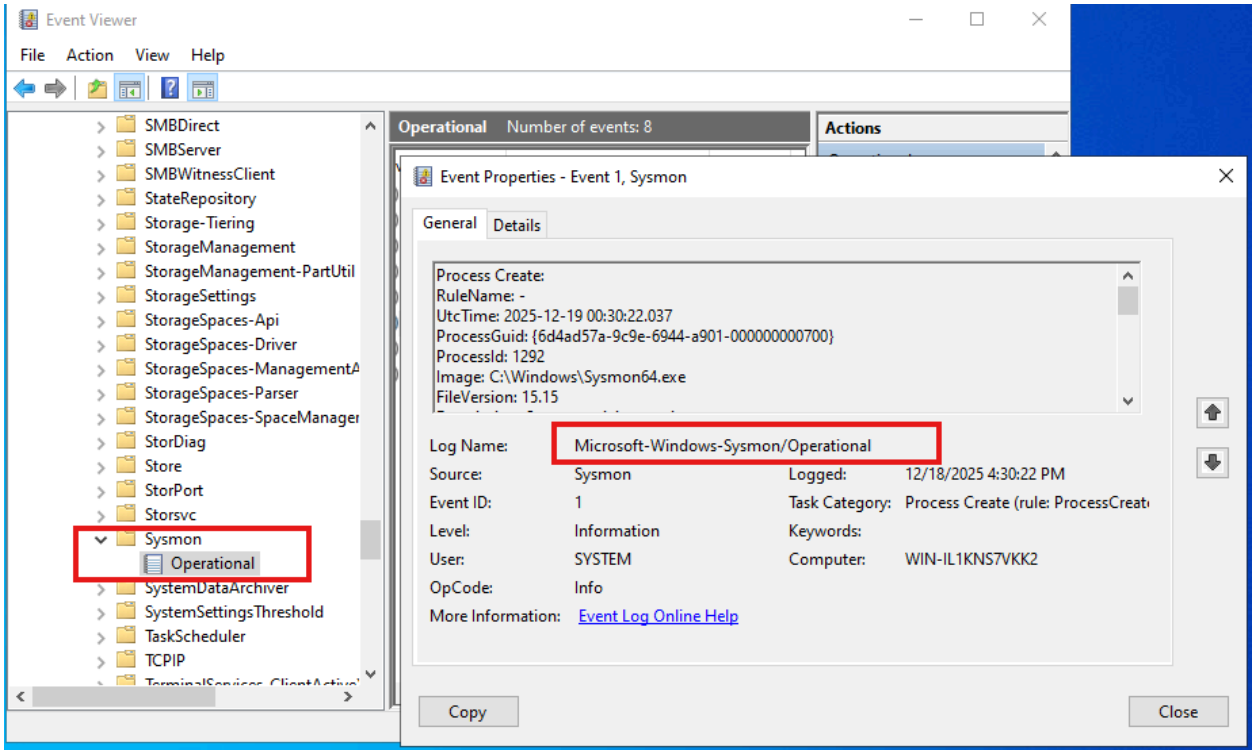
Now run the following command:

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



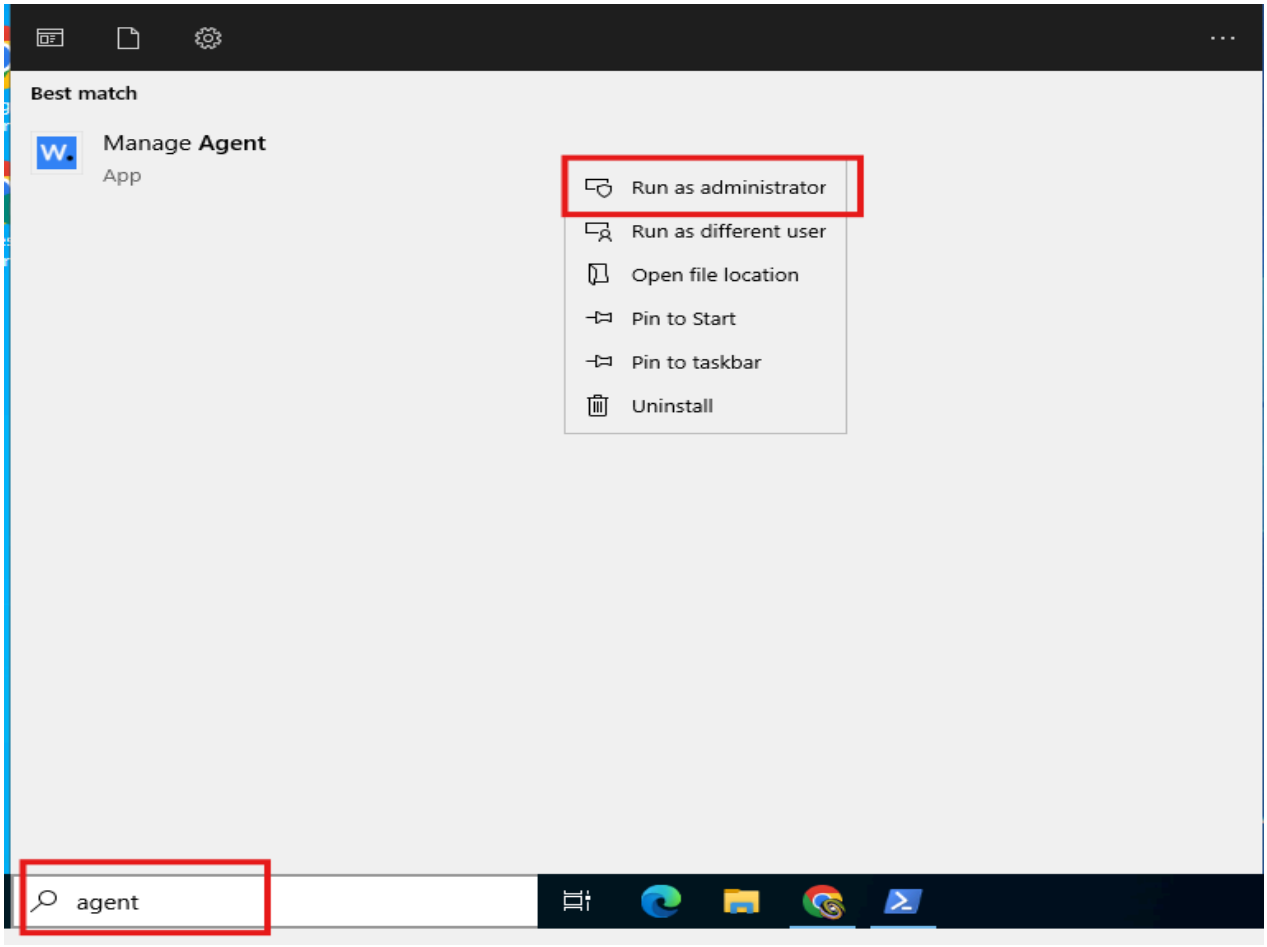
After the installation is complete, I need to verify it. Open Event Viewer and check if Sysmon logs are now visible under:

Application and Services Logs > Microsoft > Windows > Sysmon.

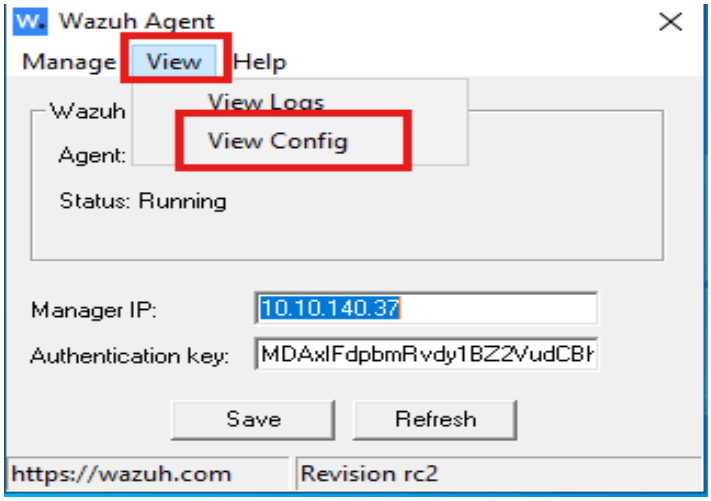


Now Sysmon is successfully installed and logs are appearing .

Now we need to forward Sysmon logs to Wazuh Manager. For this, open the **Wazuh agent** on your Windows Server. Search Agent in menu bar and “Run as administrator.



Next, open the "ossec.conf" file. We can do this by clicking on "View Config" in the Wazuh agent.



In the "ossec.conf" file, search for the "localfile" section.

After that, we have to specify the location of the Sysmon logs.

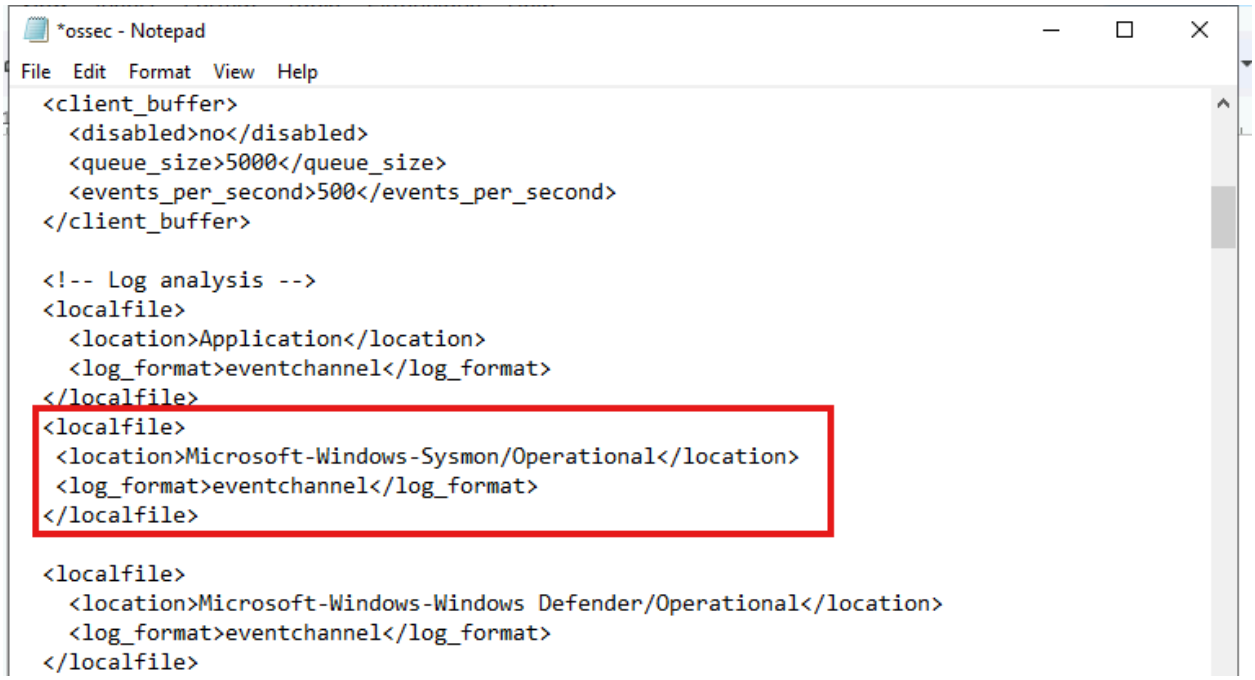
To configure Sysmon logs in the "ossec.conf" file, add the following lines:

<localfile>

  <location>Microsoft-Windows-Sysmon/Operational</location>

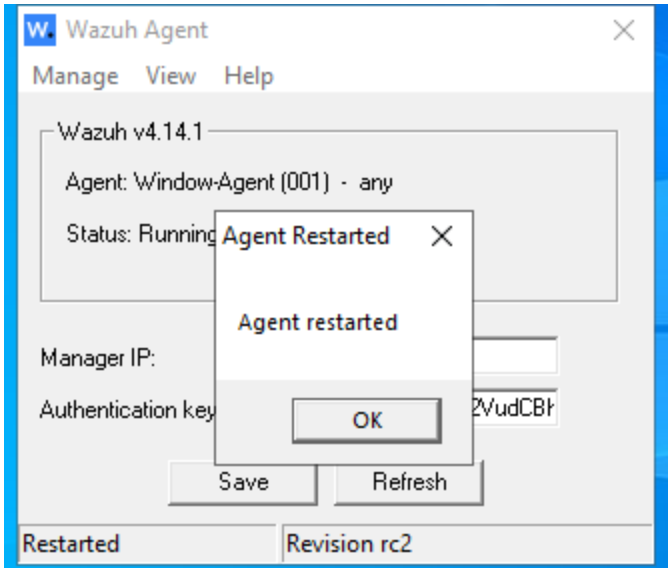
  <log_format>eventchannel</log_format>

</localfile>



Now, save the configuration file.

After saving, restart the Wazuh agent



Setup on Wazuh Server

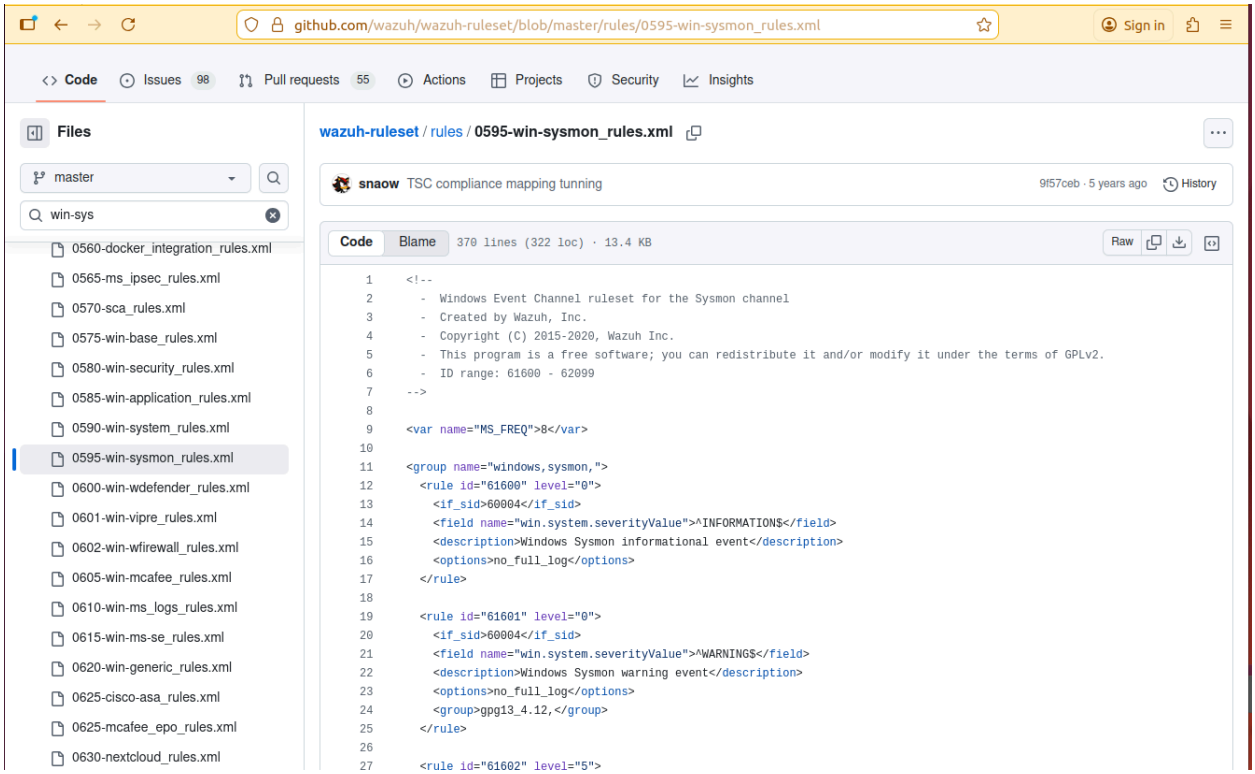
After configuring and restarting the Wazuh agent, go to the Wazuh server to download the Sysmon rules.

We need to download the Sysmon ruleset from Wazuh's official GitHub repository. These rules help Wazuh to properly analyze and generate alerts based on Sysmon logs.

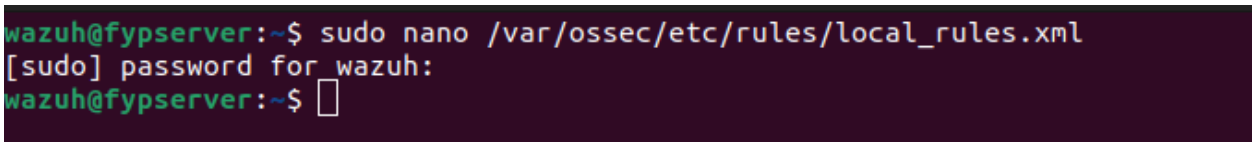
We can download the Sysmon rules from the following link:

https://github.com/wazuh/wazuh-ruleset/tree/master/rules

Download or copy the file "win-sysmon_rules.xml" from the Wazuh ruleset folder. This file contains the necessary rules for analyzing Sysmon logs.



Add these rules in following file:
/var/ossec/etc/rules/local_rules.xml



Currently, I have added only some specific rules in this document.

<group name="windows,sysmon,">

<rule id="61600" level="0">

<if_sid>60004</if_sid>

<field name="win.system.severityValue">^INFORMATION$</field>

<description>Windows Sysmon informational event</description>

<options>no_full_log</options>

</rule>

```
  <rule id="61601" level="0">
    <if_sid>60004</if_sid>
    <field name="win.system.severityValue">^WARNING$</field>
    <description>Windows Sysmon warning event</description>
    <options>no_full_log</options>
    <group>gpg13_4.12,</group>
  </rule>

  <rule id="61602" level="5">
    <if_sid>60004</if_sid>
    <field name="win.system.severityValue">^ERROR$</field>
    <description>Windows Sysmon error event</description>
    <options>no_full_log</options>
    <group>system_error,gpg13_4.3,gdpr_IV_35.7.d,</group>
  </rule>

  <rule id="61603" level="0">
    <if_sid>61600</if_sid>
    <field name="win.system.eventID">^1$</field>
                    <description>Sysmon - Event 1: Process creation
$(win.eventdata.description)</description>
    <options>no_full_log</options>
    <group>sysmon_event1,</group>
```

```
  </rule>


  <rule id="61604" level="0">
    <if_sid>61600</if_sid>
    <field name="win.system.eventID">^2$</field>
        <description>Sysmon - Event 2: A process changed a file creation time by $(win.eventdata.sourceImage)</description>
    <options>no_full_log</options>
    <group>sysmon_event2,</group>
  </rule>


  <rule id="61605" level="0">
    <if_sid>61600</if_sid>
    <field name="win.system.eventID">^3$</field>
                <description>Sysmon - Event 3: Network connection by $(win.eventdata.sourceImage)</description>
    <options>no_full_log</options>
    <group>sysmon_event3,</group>
  </rule>


  <rule id="61606" level="0">
    <if_sid>61600</if_sid>
    <field name="win.system.eventID">^4$</field>
          <description>Sysmon - Event 4: Sysmon service state changed by $(win.eventdata.sourceImage)</description>
```

```
    <options>no_full_log</options>
    <group>sysmon_event4,</group>
  </rule>

  <rule id="61607" level="0">
    <if_sid>61600</if_sid>
    <field name="win.system.eventID">^5$</field>
                <description>Sysmon - Event 5: Process terminated by
$(win.eventdata.sourceImage)</description>
    <options>no_full_log</options>
    <group>sysmon_event5,</group>
  </rule>
    </group>
```

```
GNU nano 6.2                                    /var/ossec/etc/rules/local_rules.xml *
<group name="windows,sysmon,">
  <rule id="61600" level="0">
    <if_sid>60004</if_sid>
    <field name="win.system.severityValue">^INFORMATION$</field>
    <description>Windows Sysmon informational event</description>
    <options>no_full_log</options>
  </rule>

  <rule id="61601" level="0">
    <if_sid>60004</if_sid>
    <field name="win.system.severityValue">^WARNING$</field>
    <description>Windows Sysmon warning event</description>
    <options>no_full_log</options>
    <group>gpg13_4.12,</group>
  </rule>

  <rule id="61602" level="5">
    <if_sid>60004</if_sid>
    <field name="win.system.severityValue">^ERROR$</field>
    <description>Windows Sysmon error event</description>
    <options>no_full_log</options>
    <group>system_error,gpg13_4.3,gdpr_IV_35.7.d,</group>
  </rule>

  <rule id="61603" level="0">
    <if_sid>61600</if_sid>
    <field name="win.system.eventID">^1$</field>
    <description>Sysmon - Event 1: Process creation $(win.eventdata.description)</description>
    <options>no_full_log</options>
    <group>sysmon_event1,</group>
  </rule>

  <rule id="61604" level="0">
    <if_sid>61600</if_sid>
    <field name="win.system.eventID">^2$</field>
    <description>Sysmon - Event 2: A process changed a file creation time by $(win.eventdata.sourceImage)</description>
    <options>no_full_log</options>
    <group>sysmon_event2,</group>
  </rule>
```

Now saved the configuration file and restart Wazuh manager.

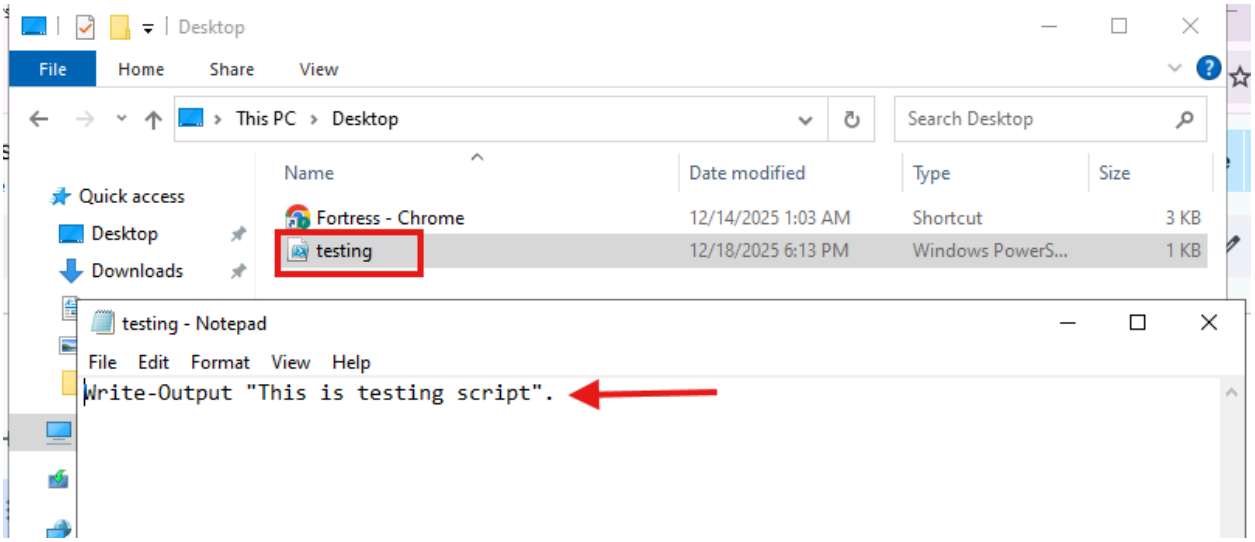
Sudo systemctl restart wazuh-manager

```
wazuh@fypserver:~$ sudo systemctl restart wazuh-manager
wazuh@fypserver:~$
```

Testing Phase:

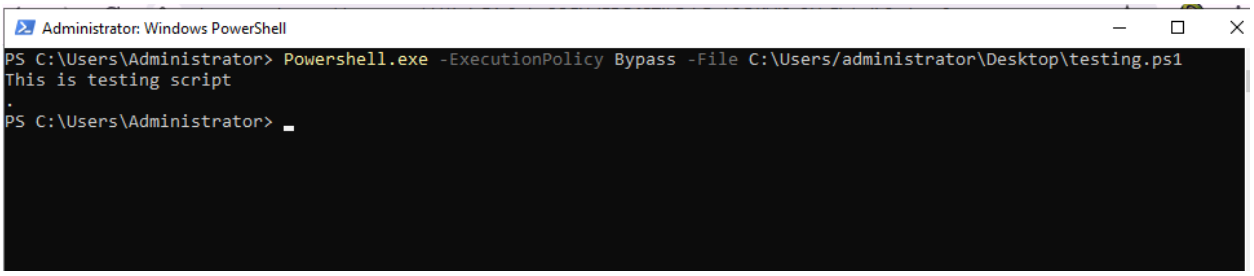
First I create the file in my Desktop folder in window server where my agent is installed.

File Name: testing.ps1



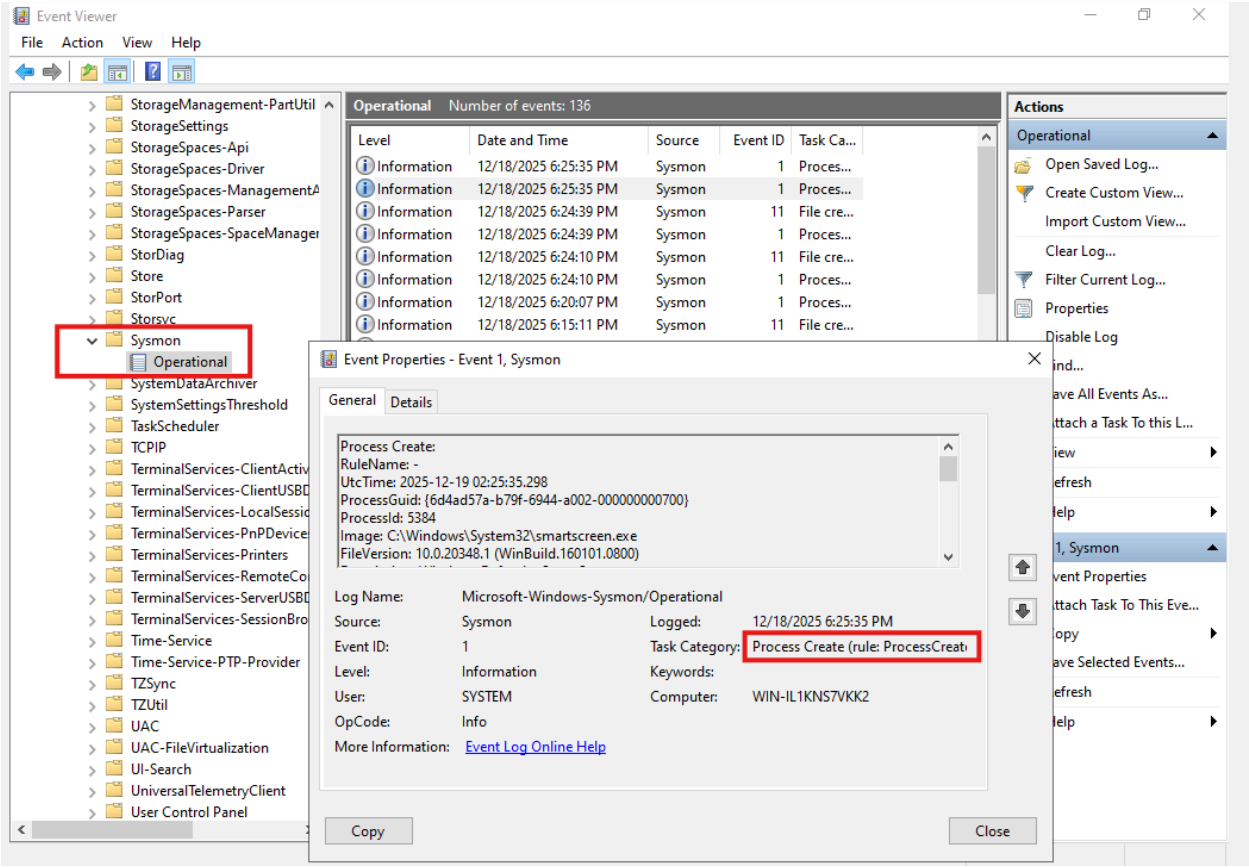
Run this command on powershell.

Powershell.exe -ExecutionPolicy Bypass -File “C:\Users/administrator\Desktop\testing.ps1”



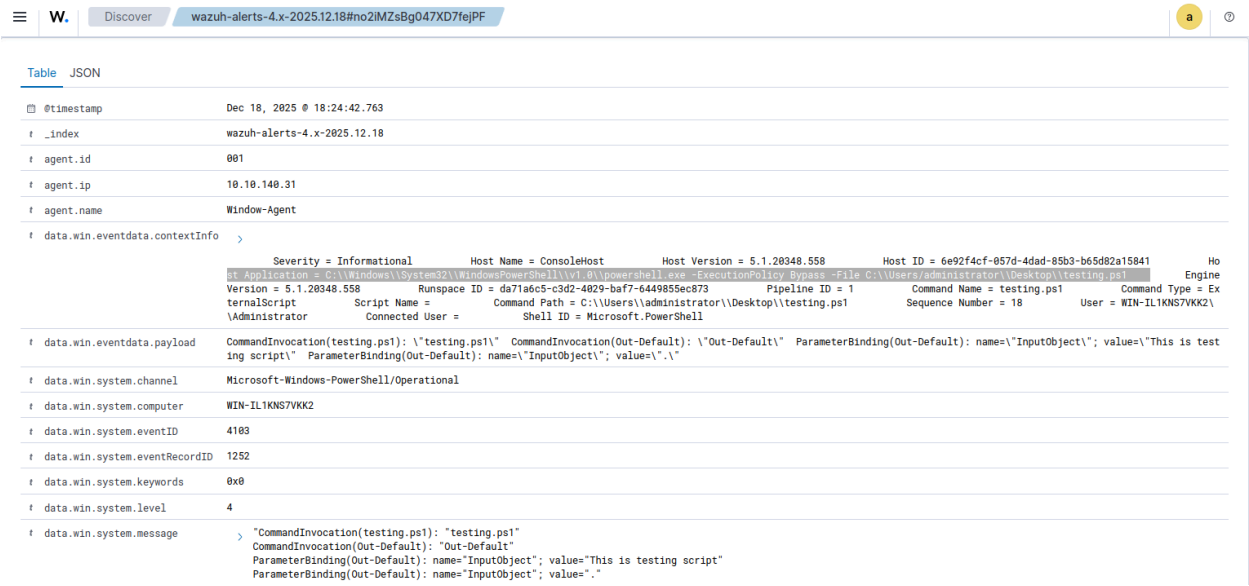
Now check it in the event viewer for process creation.

Applications and Services Logs > Microsoft > Windows > Sysmon > Operational



Now we see these logs in the Wazuh Dashboard.

Detailed Logs:



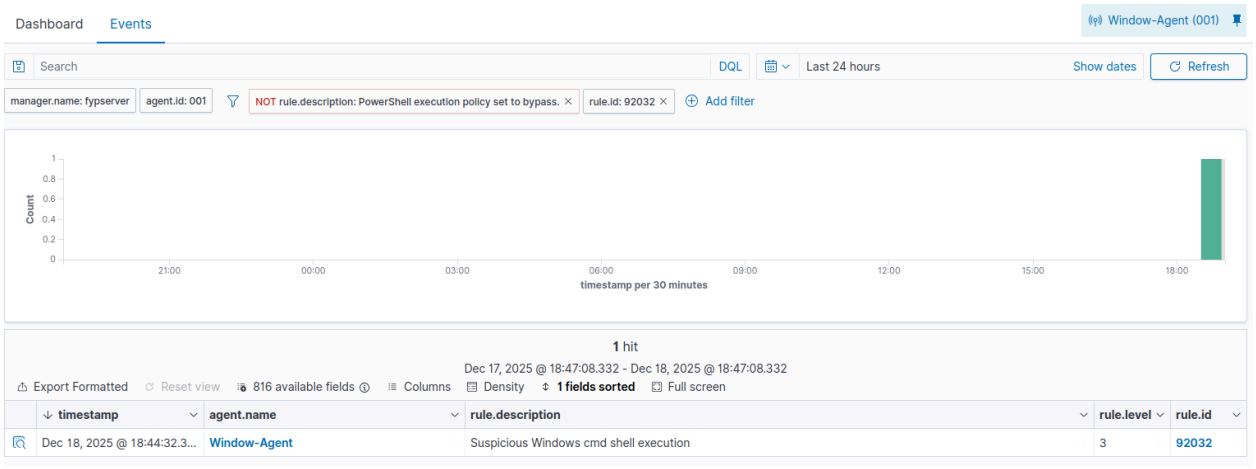
t data.win.system.opcode	20
t data.win.system.processID	3840
t data.win.system.providerGuid	{a0c1853b-5c40-4b15-8766-3cf1c58f985a}
t data.win.system.providerName	Microsoft-Windows-PowerShell
t data.win.system.severityValue	INFORMATION
t data.win.system.systemTime	2025-12-19T02:24:40.0910533Z
t data.win.system.task	106
t data.win.system.threadID	5992
t data.win.system.version	1
t decoder.name	windows_eventchannel
t full_log	{"win":{"system":{"providerName":"Microsoft-Windows-PowerShell","providerGuid":"{a0c1853b-5c40-4b15-8766-3cf1c58f985a}","eventID":"4103","version":"1","level":"4","task":"106","opcode":"20","keywords":"0x0","systemTime":"2025-12-19T02:24:40.0910533Z","eventRecordID":"1252","processID":"3840","threadID":"5992","channel":"Microsoft-Windows-PowerShell/Operational","computer":"WIN-IL1KNS7VKK2","severityValue":"INFORMATION","message":"\"CommandInvocation(testing.ps1): \"testing.ps1\"\r\nCommandInvocation(Out-Default): \"Out-Default\"\r\nParameterBinding(Out-Default): name=\"InputObject\"; value=\"This is testing script\"\r\nParameterBinding(Out-Default): name=\"InputObject\"; value=\".\"\r\n\r\n\r\nContext:\r\n Severity = Informational\r\n Host Name = ConsoleHost\r\n Host Version = 5.1.20348.558\r\n Host ID = 6e92f4cf-057d-4dad-85b3-b65d82a15841\r\n Host Application = C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe -ExecutionPolicy Bypass -File C:\\Users\\administrator\\Desktop\\testing.ps1\r\n Engine Version = 5.1.20348.558\r\n Runspace ID = da71a6c5-c3d2-4029-baf7-6449855ec873\r\n Pipeline ID = 1\r\n C
t id	1766064282.832419
t input.type	log
t location	EventChannel
t manager.name	fypserver
t rule.description	PowerShell execution policy set to bypass.

Test 2:

cmd.exe /c whoami

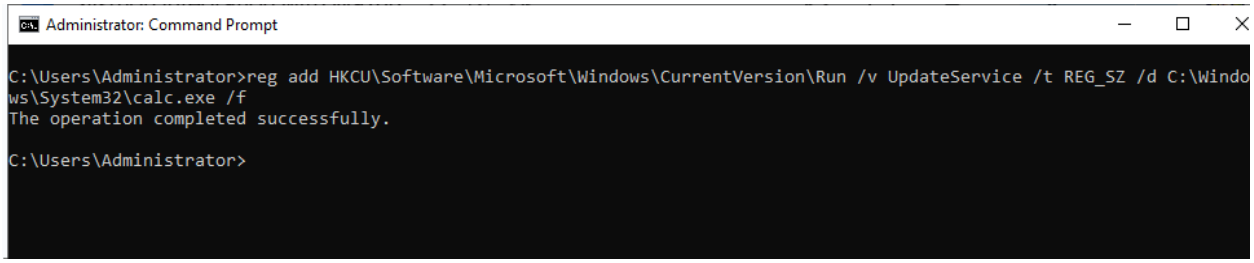
Run this command on cmd:

And its alerts appear in Dashboard.

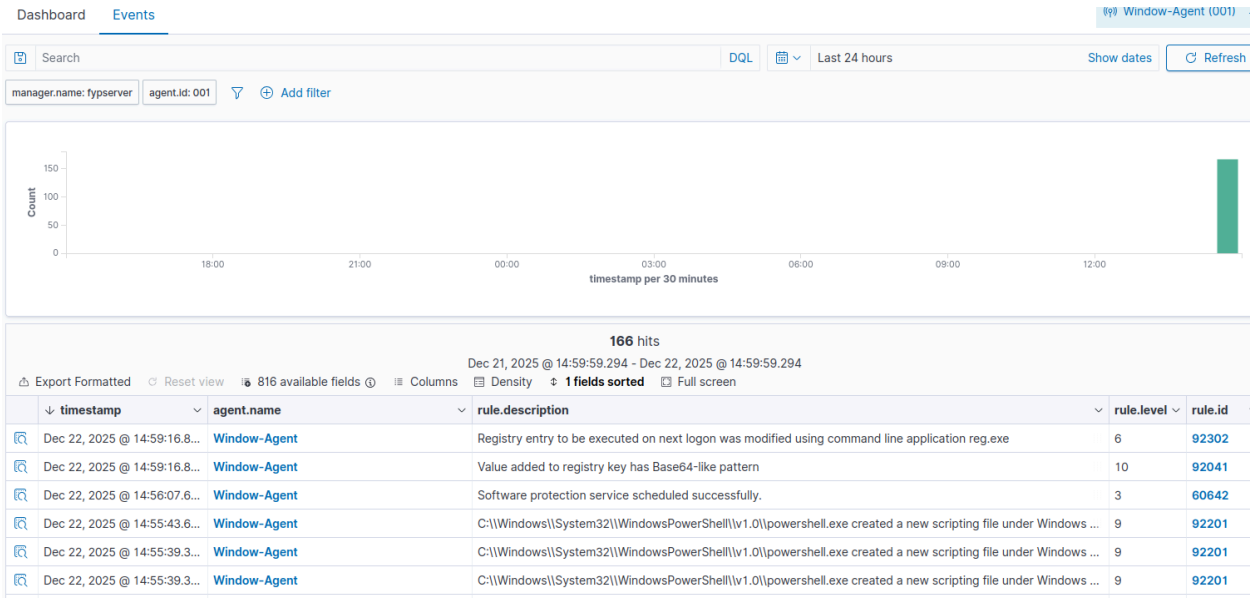


Test3: Registry changes detection alerts:

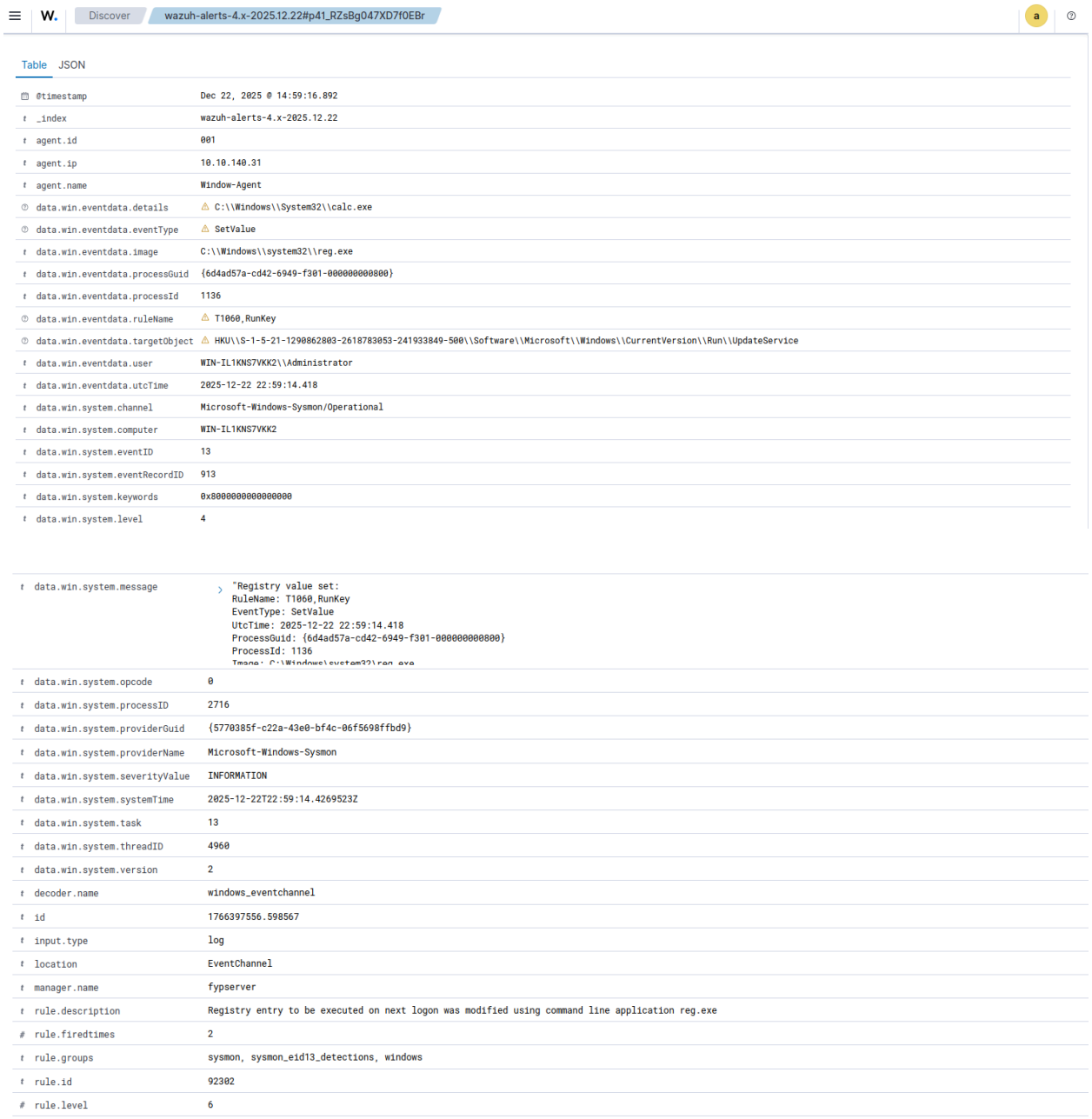
reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v UpdateService /t REG_SZ /d C:\Windows\System32\calc.exe /f



Alerts appear in the Dashboard.



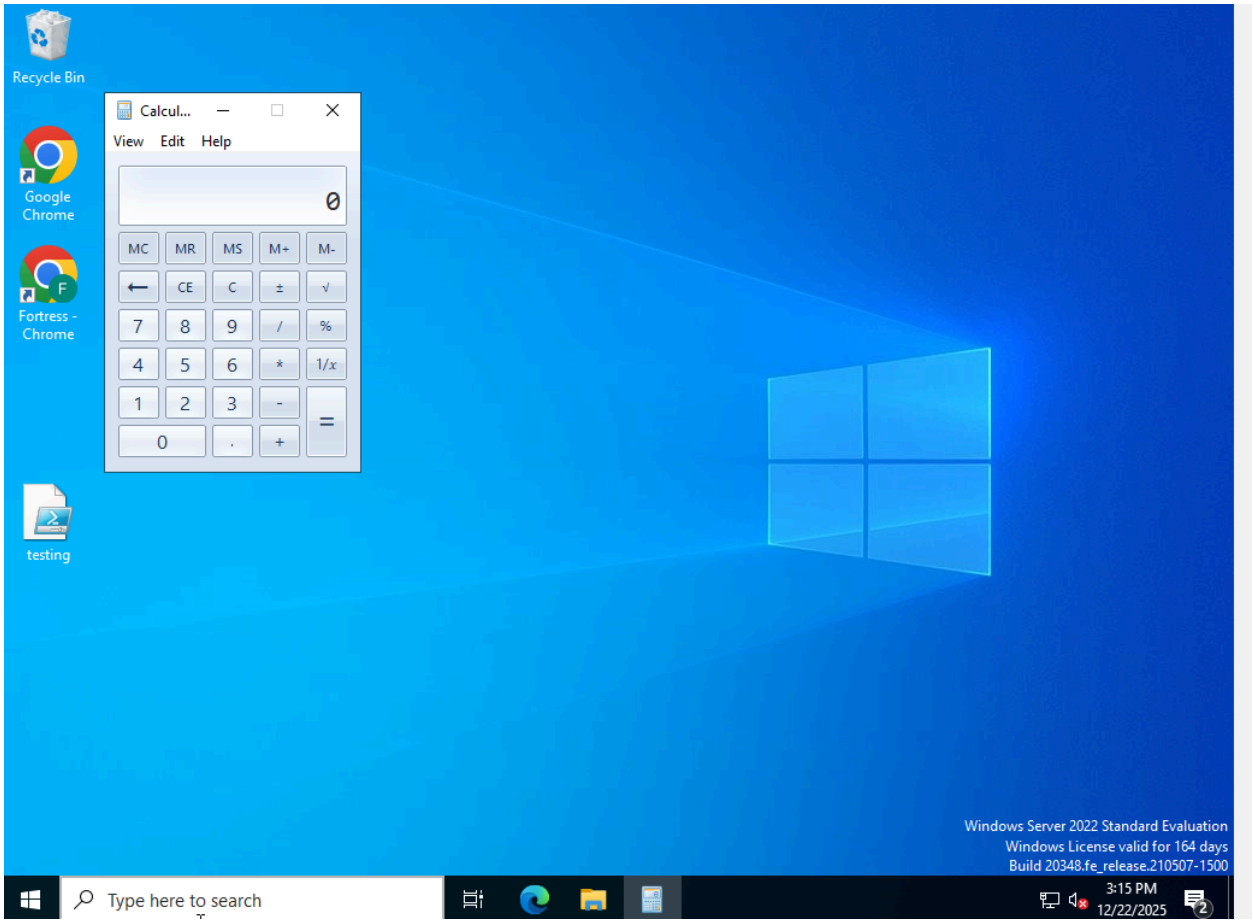
Details:



After adding the registry entry using the Command Prompt (CMD), the system behavior was tested.

During the next login to the Windows Server, the Calculator application opened automatically.

This confirms that the registry Run key was successfully modified and executed as expected during user login. .



Summary:

We successfully connected Sysmon with Wazuh to watch what happens on a Windows system. Sysmon collects detailed actions like running programs, using commands, and changing the registry. The Wazuh Agent sends these logs to the Wazuh Manager, which checks them and shows alerts on the Dashboard.

During testing, running PowerShell scripts, CMD commands, and adding registry entries were all detected correctly. This shows that Sysmon with Wazuh can help see suspicious activity and keep the system secure.