**Lab 6 Report: Using Wazuh to Add Sysmon Logging**

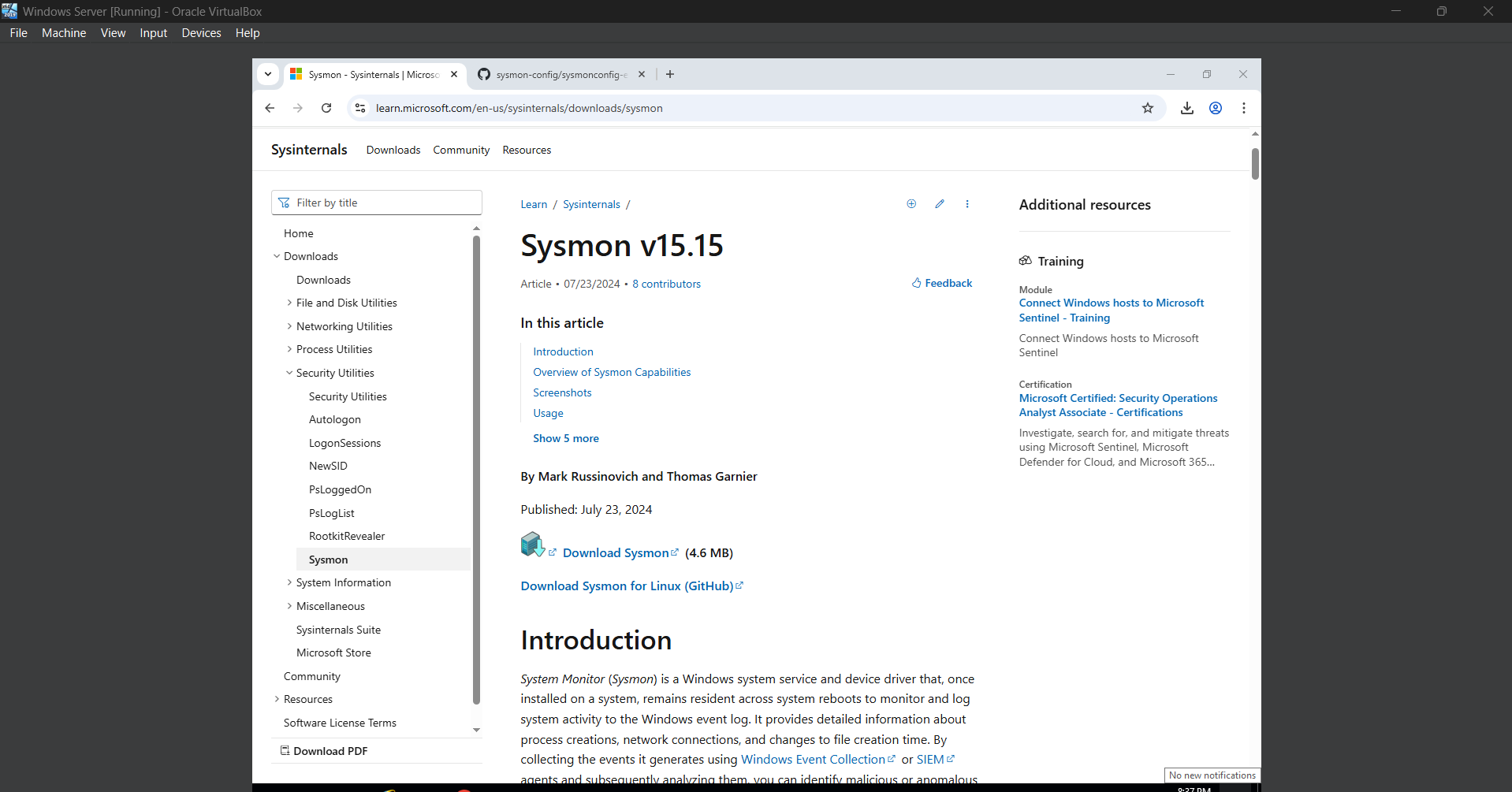
**Introduction**

In this lab, I added Sysmon logs to the Security Onion system by installing the Sysmon driver on a Windows endpoint and configuring the Wazuh agent to forward those logs. I used OT-DC1 as the primary endpoint for the installation. I have included detailed documentation along with screenshots after each step to demonstrate what I performed and observed.

**Step 1: Downloading Sysmon**

I started by downloading Sysmon from the official Sysinternals page: <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>

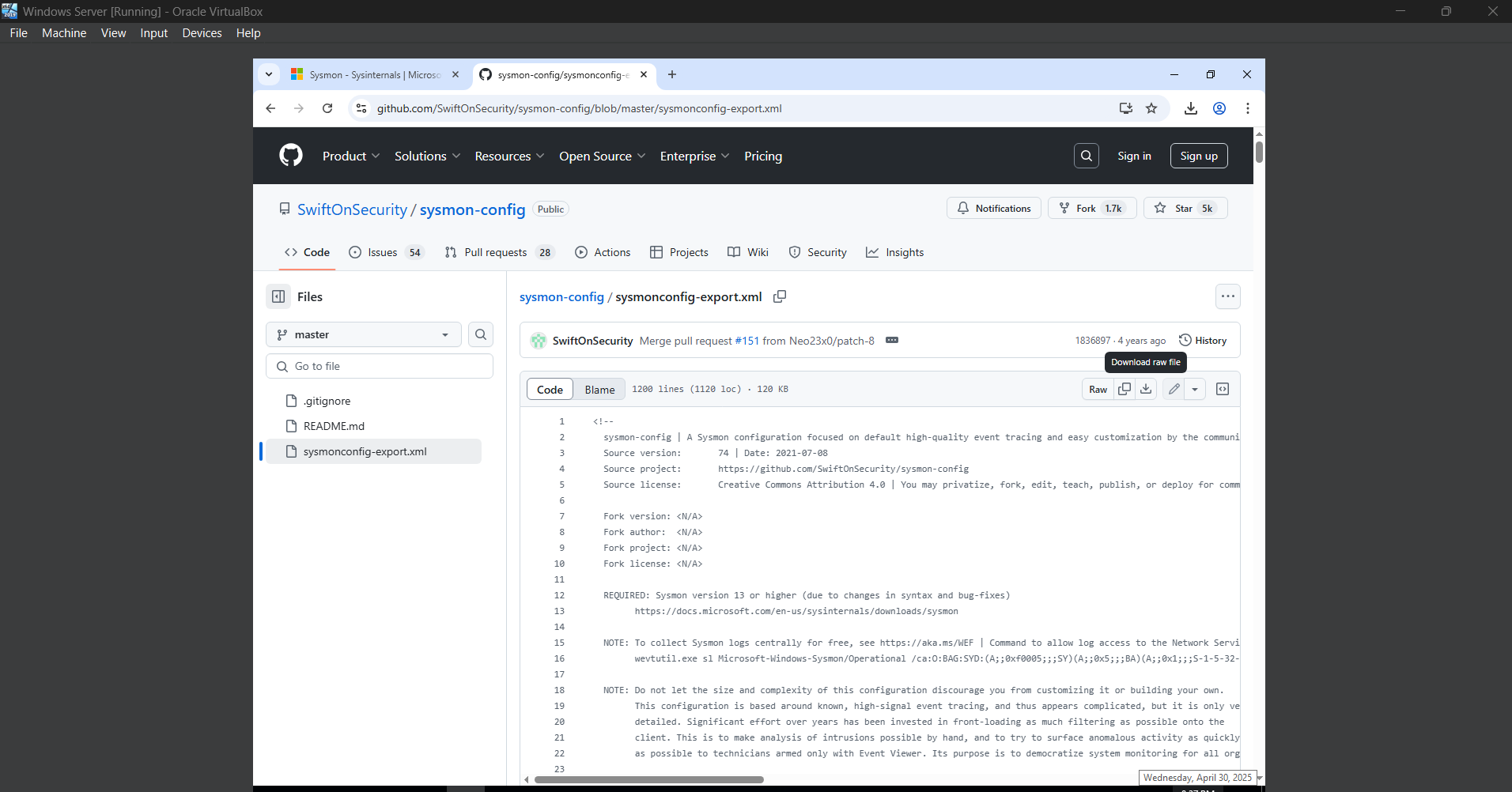
The ZIP file I downloaded contained both 32-bit and 64-bit binaries.



**Step 2: Downloading the Sysmon Configuration File**

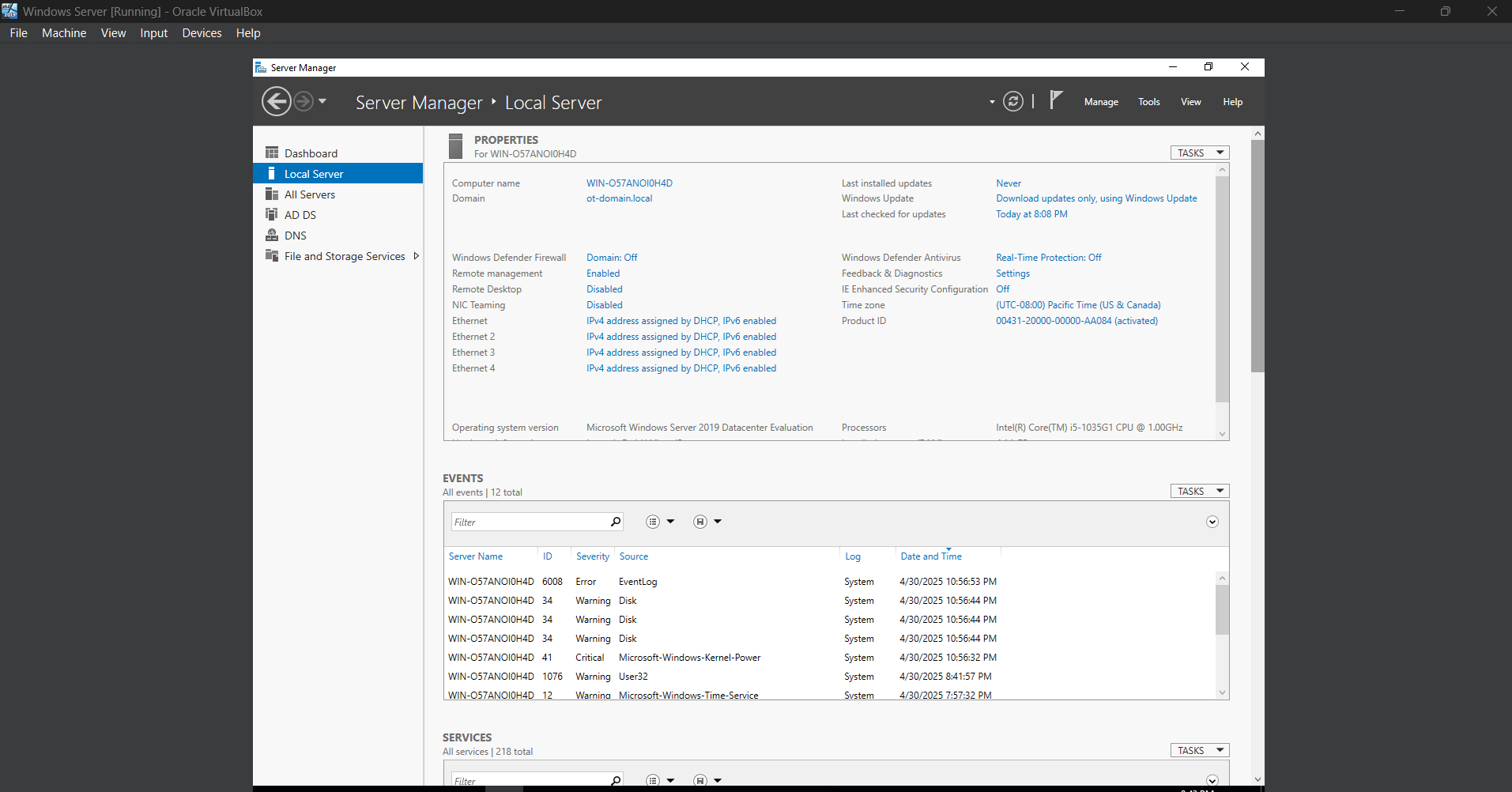
Next, I downloaded the Sysmon configuration file from the following GitHub repository: <https://github.com/SwiftOnSecurity/sysmon-config/blob/master/sysmonconfig-export.xml>

This configuration was built by SwiftOnSecurity and designed for compatibility with SIEM and event correlation tools.



**Step 3: Logging into Endpoint OT-DC1**

I logged into the Windows machine OT-DC1, which I used as my target endpoint for installing Sysmon. Sysmon is supported only on Windows, so I used this system as my starting point.

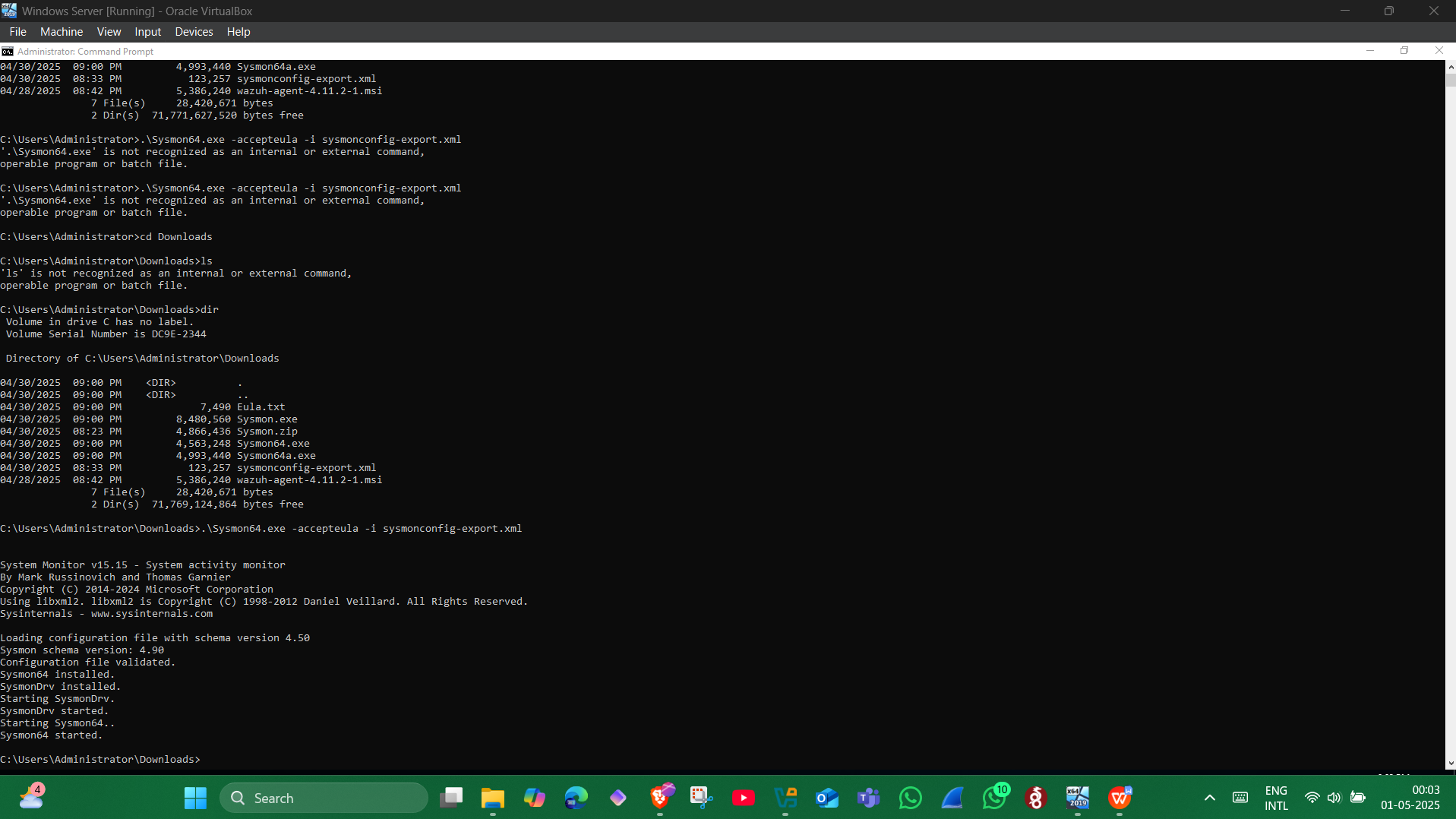


**Step 4: Installing Sysmon**

I opened an elevated PowerShell terminal and navigated to the folder containing the Sysmon executable and the configuration file. I ran the following command:

sysmon.exe -accepteula -i sysmonconfig-export.xml

This successfully installed and started the Sysmon service. It began capturing events immediately and logging them under the Sysmon log channel.

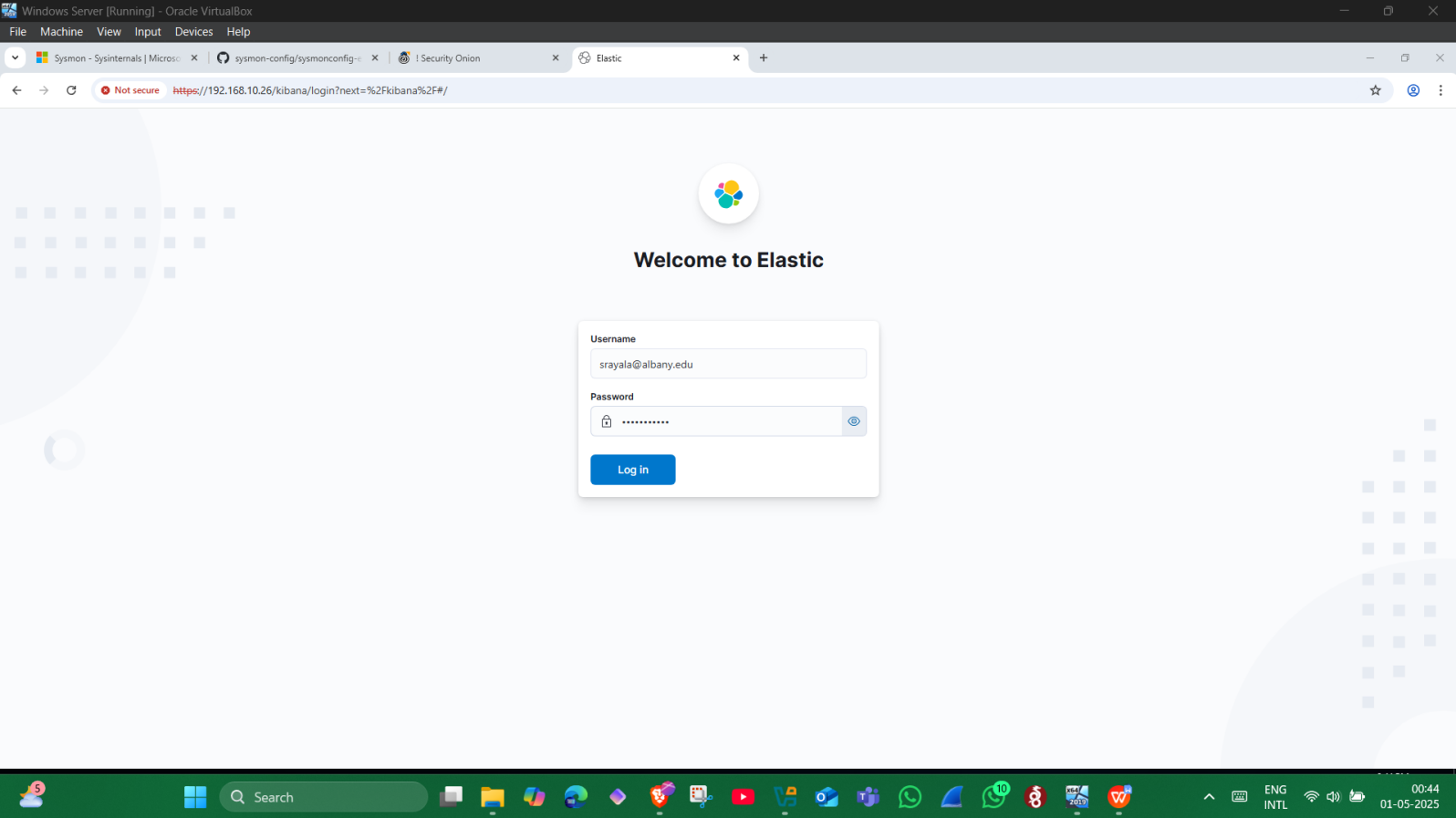


**Step 5: Verifying Log Forwarding via Wazuh and Security Onion**

I confirmed that the Wazuh agent was monitoring the Sysmon event log and forwarding entries to Security Onion. I accessed the Kibana dashboard through the Security Onion web interface. I navigated to:

**Home | Host | Sysmon dashboard**

There, I verified that the Sysmon event logs were visible, particularly the logs for a PowerShell terminal process initiation.



**Step 6: Understanding the Logged Data**

The logs generated by Sysmon included detailed metadata:

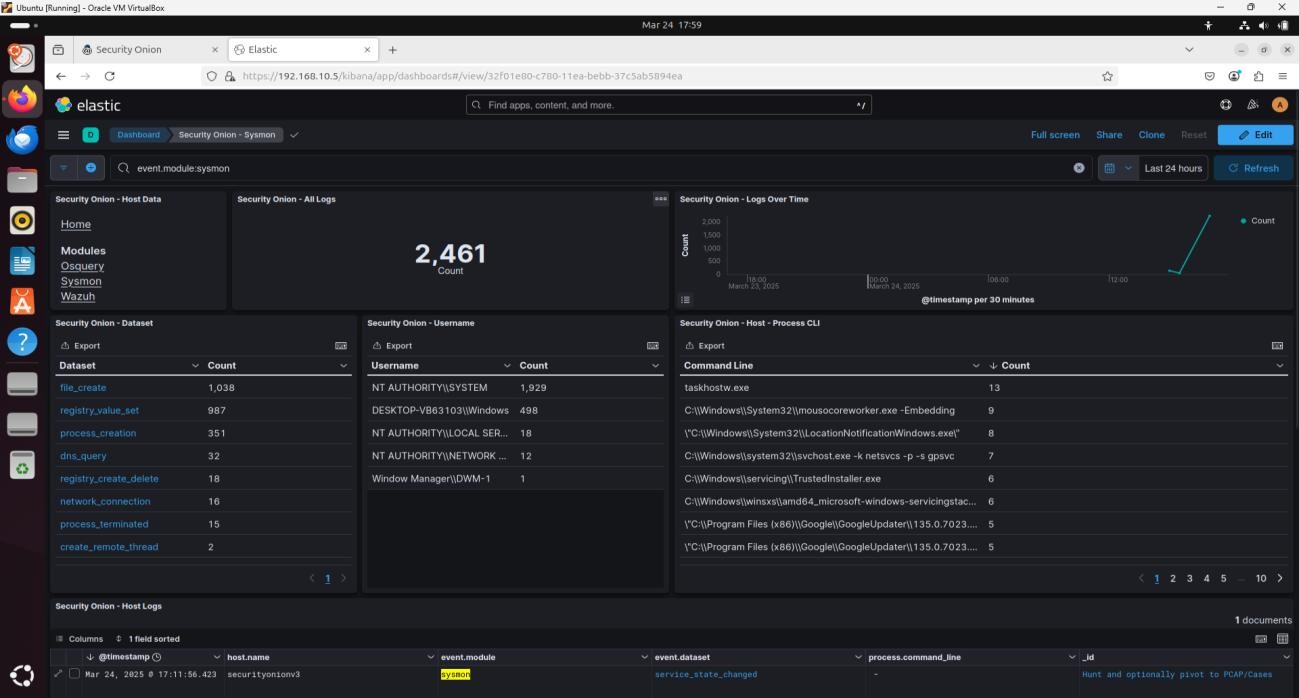
The user who launched the PowerShell session (user.name)

The parent application (parent.process)

File path of the executable (process.executable)

Command-line arguments (process.command\_line)

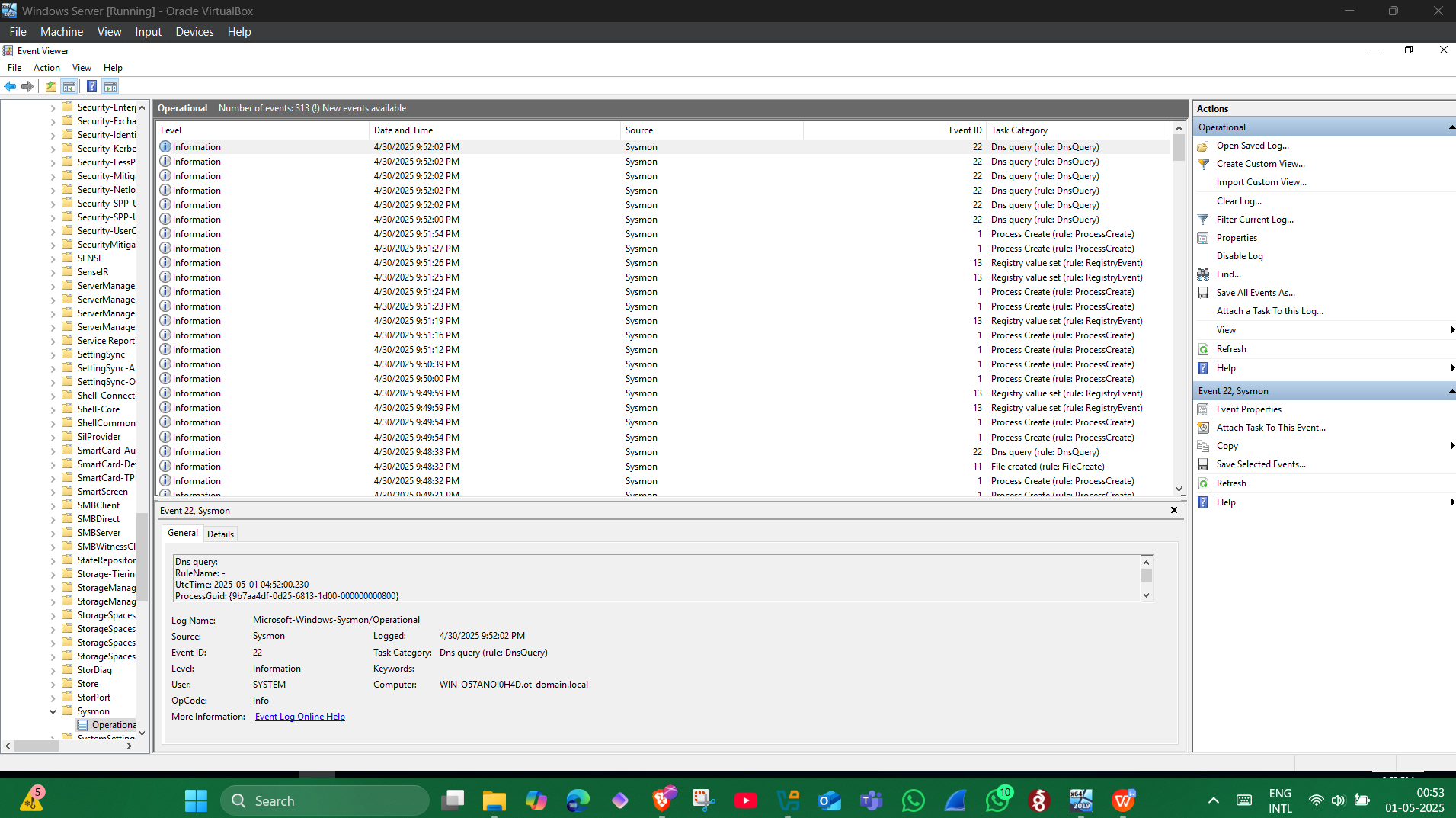
Process ID and hashes (winlog.event\_data.hashes)



**Step 7: Locating Sysmon Event Log in Windows Event Viewer**

On the OT-DC1 system, I opened the Event Viewer and confirmed that the Sysmon logs were being recorded under:

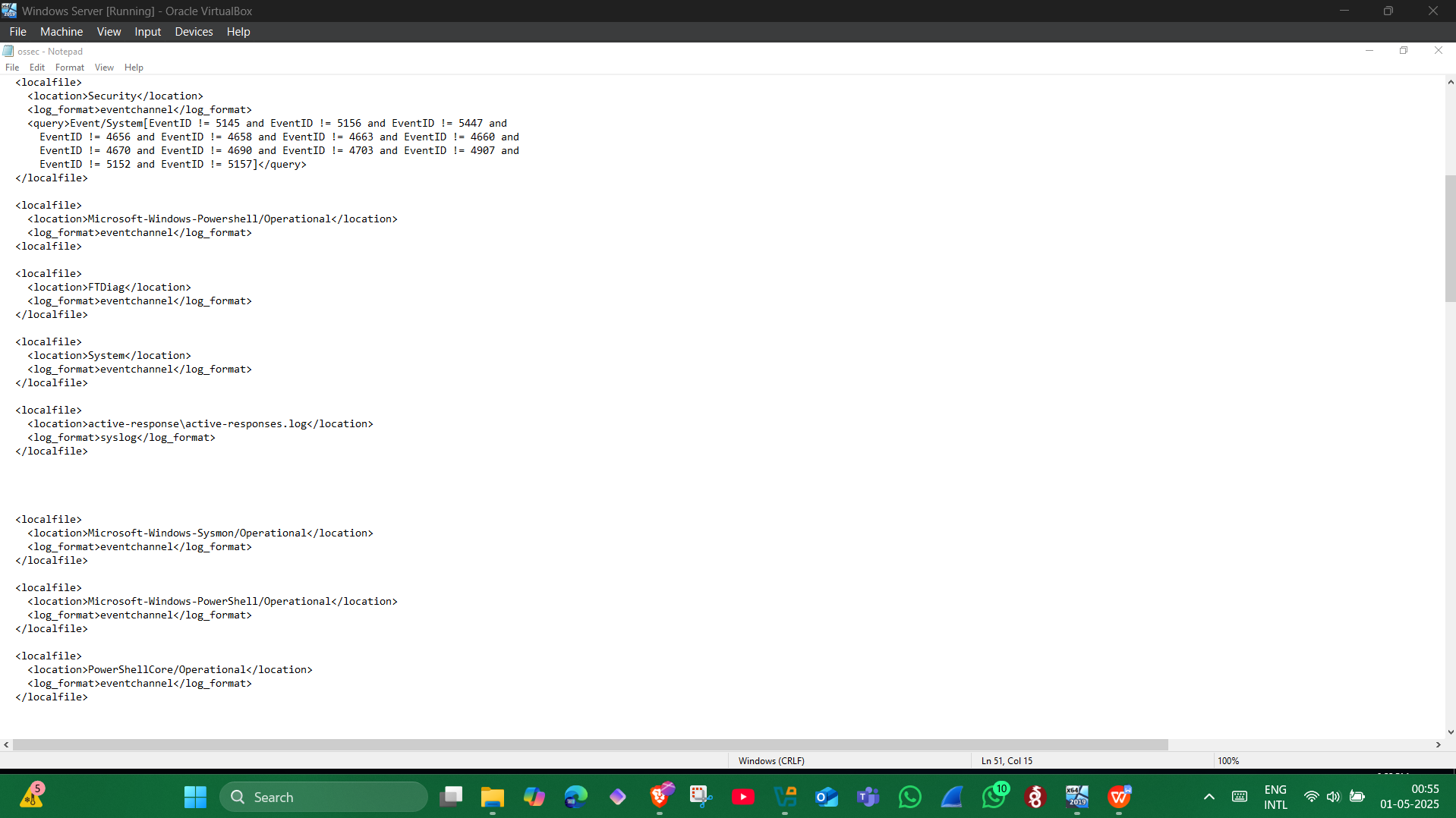
**Applications and Services Logs | Microsoft | Windows | Sysmon | Operational**



**Step 8: Adding FactoryTalk Diagnostics Logs to Wazuh**

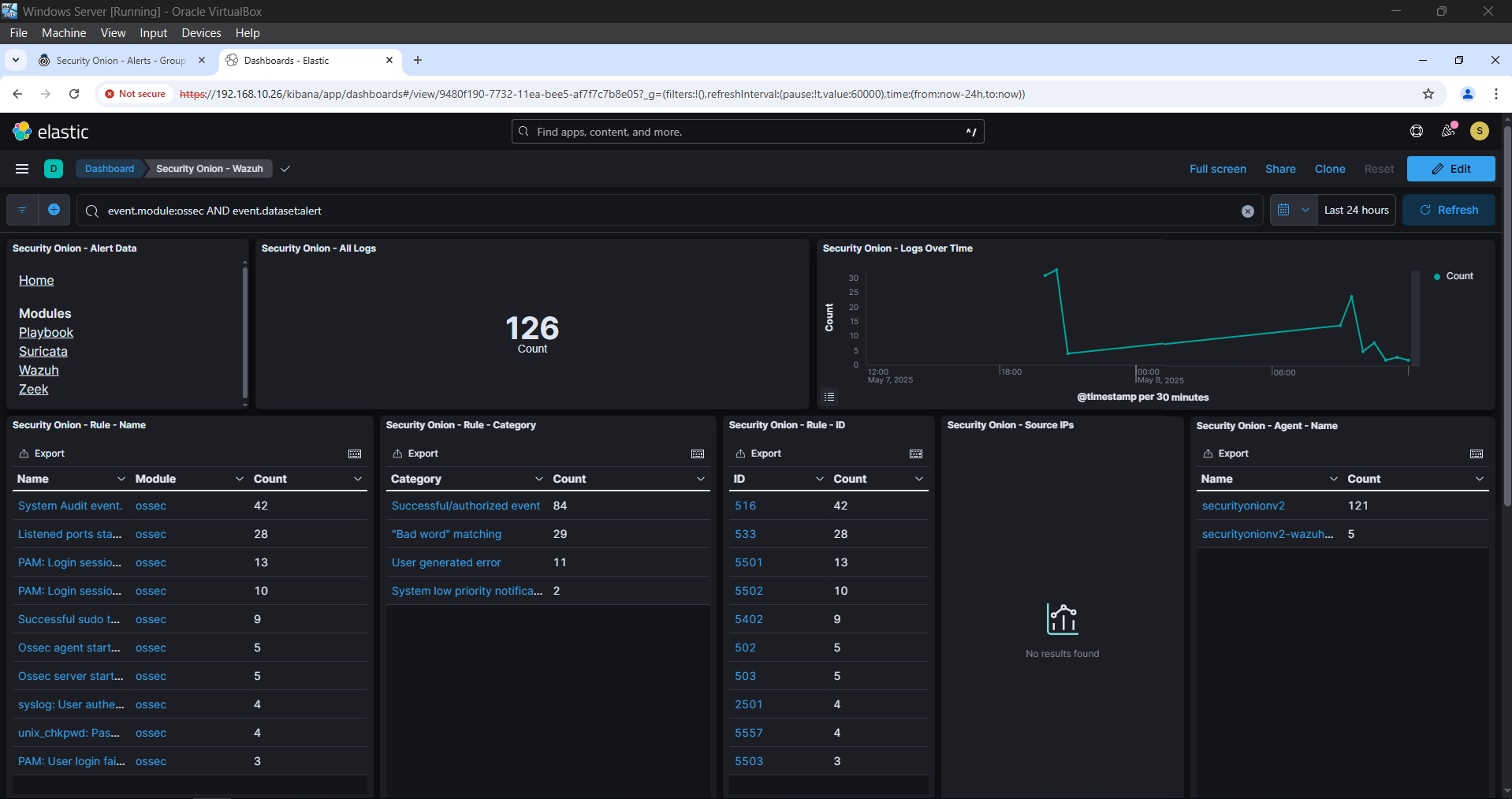
I wanted to extend log monitoring further. I added Rockwell Automation FactoryTalk Diagnostics logs by modifying the Wazuh agent configuration file on OT-DC1. I specified the log name as FTDiag.

To identify the exact name, I right-clicked the FactoryTalk Diagnostics log in Event Viewer and selected Properties. This showed the full log name as FTDiag.



**Step 9: Restarting Wazuh Agent**

After updating the configuration, I restarted the Wazuh agent so it would begin forwarding the new event logs to Security Onion.

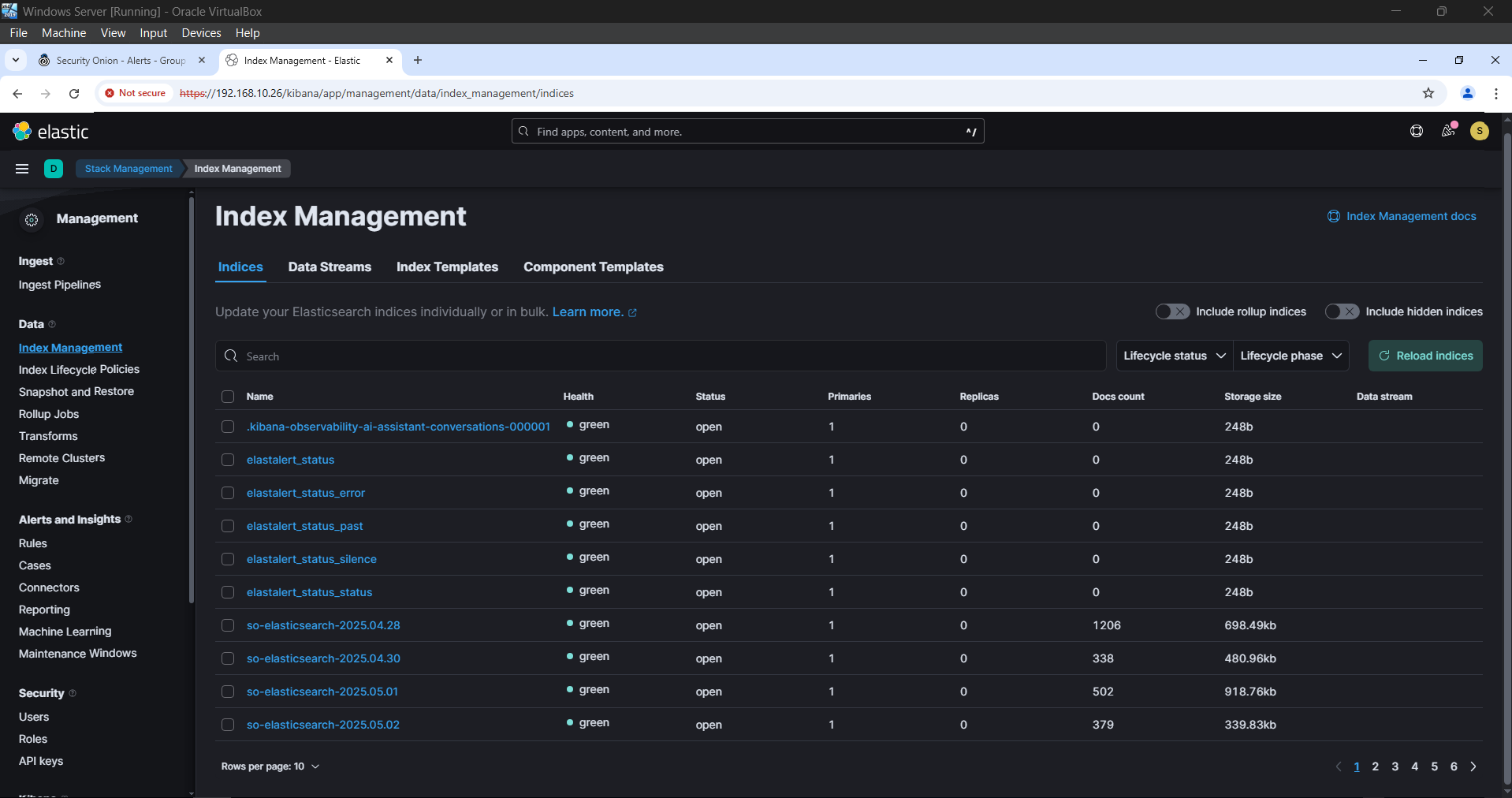


**Step 10: Refreshing Index Patterns in Elasticsearch**

To include new attributes in searches, I refreshed Elasticsearch index patterns via the Kibana web interface. I navigated to:

**Menu > Stack Management > Index Patterns**

I clicked the **Refresh** button and confirmed the operation.



**Conclusion**

In this lab, I successfully extended the logging capabilities of Security Onion by integrating Sysmon logs and additional ICS event logs using Wazuh. The Sysmon driver captured detailed system events, which were forwarded to Elasticsearch through Wazuh and visualized using Kibana. This integration significantly enhances our ability to detect and respond to a wider variety of security-related events.

Next, I will proceed to explore how to enhance visibility into PowerShell scripts and command usage.