Lab 6: Using Wazuh to add Sysmon logging

Introduction

As a first exercise for this chapter, we are going to add Sysmon logs to the Security Onion data, by installing the Sysmon driver on endpoints and having Wazuh forward to Security Onion the logs it creates for us.

**Submission:** You need to submit a detailed lab report, with screenshots, to describe what you have done and observed. Questions will be defined as you progress through the lab. The lab report will be compiled as a Word document and submitted on Brightspace by **March 31, 2025.**

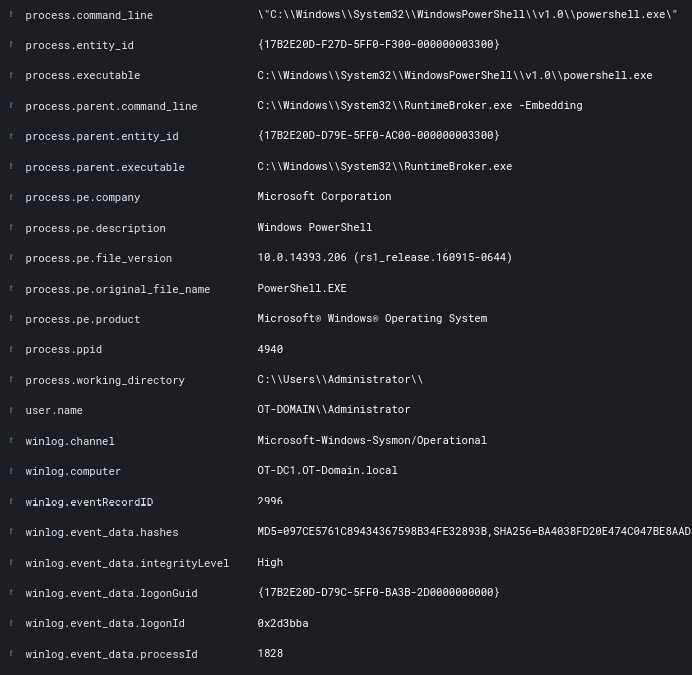
To see how Sysmon logs can be integrated with a Security Onion setup, follow along with the next instructions (note that I picked **OT-DC1** as an example system to install Sysmon on, but you should perform this operation on every system you want to install Wazuh on):

1. Download Sysmon from <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>. The ZIP file that you download will contain the binary for both 32-bit and 64-bit architectures.
2. Download the **sysmonconfig-export.xml** file from <https://github.com/SwiftOnSecurity/sysmon-config/blob/master/sysmonconfig-export.xml>. This configuration file was created by **SwiftOnSecurity** and built to allow maximal compatibility with a variety of **Security Information and Event Management** (**SIEM**) and event collection and correlation solutions. We will be applying the configuration when we install Sysmon on an endpoint.
3. Log in to an endpoint where you want to start installing Sysmon. I chose **OT-DC1** as my first endpoint. The endpoint should be a Windows machine—as of writing, Sysmon is only available for Windows; Linux is on the horizon, though. It really doesn't matter which Windows machine you start with as all your systems should eventually have Sysmon installed.
4. Open an elevated Command Prompt or PowerShell terminal, navigate to the folder where you copied the **sysmon** executable and the configuration file to, and enter the following command, shown in bold:

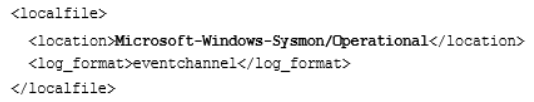


The Sysmon driver was installed and started and is now capturing interesting events and storing them in the Sysmon event log. Wazuh is monitoring that event log and will forward any new entries to Security Onion, where it is added into the Elasticsearch database and included in searches.

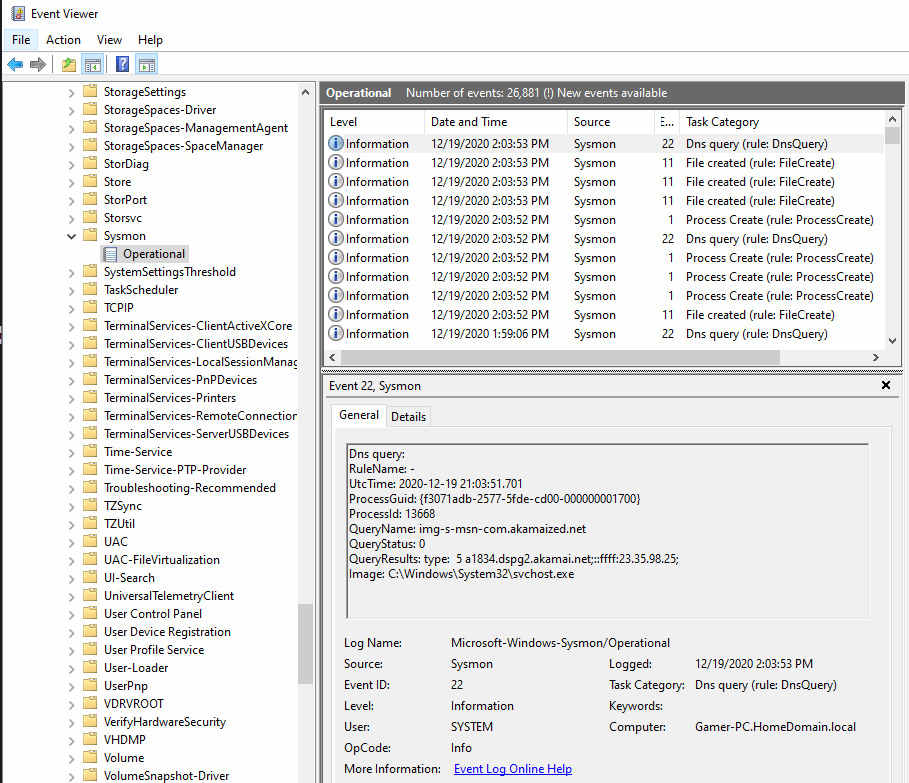
We can verify things are properly configured and being forwarded by opening the Security Onion web portal and opening the Kibana tool. Next, navigate to **Home** | **Host** | **Sysmon** dashboard and view the event logs at the bottom of the dashboard screen. The following screenshot shows details of starting a PowerShell Terminal process on **OT-DC1**:

*Figure 9.1 – Exercise 1: Sysmon log for starting the PowerShell process*

As you can see, the Sysmon tool adds a tremendous amount of detail around the process start event. We can see who started the PowerShell process (**user.name**), from what application the process was started (**\*.parent\***), where the process executable file is located (**process.executable**), the command-line arguments (**process.command\_line**), the process ID, the process and parent executable file hashes (**winlog.event\_data.hashes**), and so on.

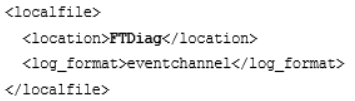


The configuration snippet instructs Wazuh to look inside the Windows event logs at **Microsoft-Windows-Sysmon/Operational** and forward any entries to the Wazuh service, running on the Security Onion VM. The **Microsoft-Windows- Sysmon/Operational** location that Wazuh is monitoring corresponds with the location in the Windows event viewer utility—**Applications and Services Logs** | **Microsoft** | **Windows** | **Sysmon** | **Operational**, as illustrated in the following screenshot:

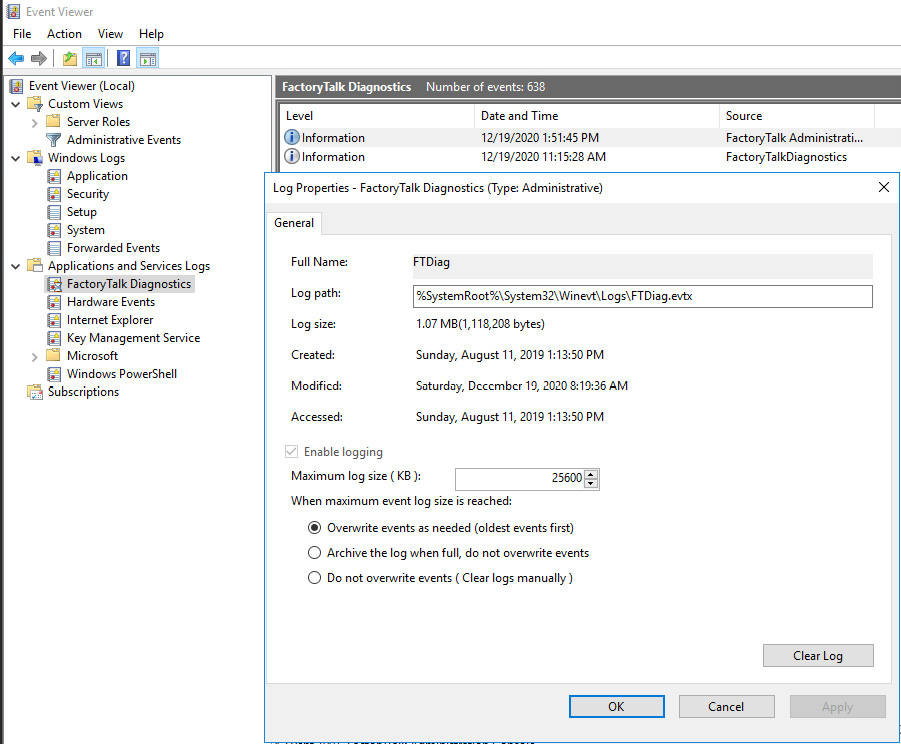
*Figure 9.2 – Exercise 1: Sysmon Windows event log location*

You can add other interesting event logs to Wazuh's configuration as well so that they can become part of the Elasticsearch database. For example, if we want to start sending **Rockwell Automation FactoryTalk Diagnostics** logs (**Applications and Services Logs** | **FactoryTalk Diagnostics**) to Security

Onion, we can add the following snippet to the Wazuh agent configuration (and restart the agent):

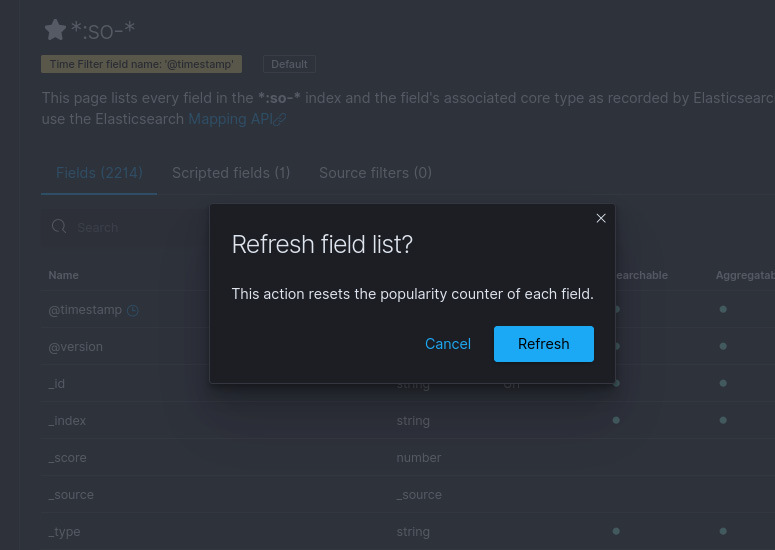


Notice I am specifying the location as **FTDiag**. This is done because that is the **full name** of the log file, which you can discover by right-mouse clicking on the event log and selecting **Properties**, as illustrated in the following screenshot:

*Figure 9.3 – Exercise 1: FactoryTalk Diagnostics log – filename*

This configuration change will start pulling in the **FactoryTalk Diagnostics** log into Elasticsearch. **FactoryTalk Diagnostic** logs are a treasure trove for Rockwell Automation applications and security events, as well as for application and performance data.

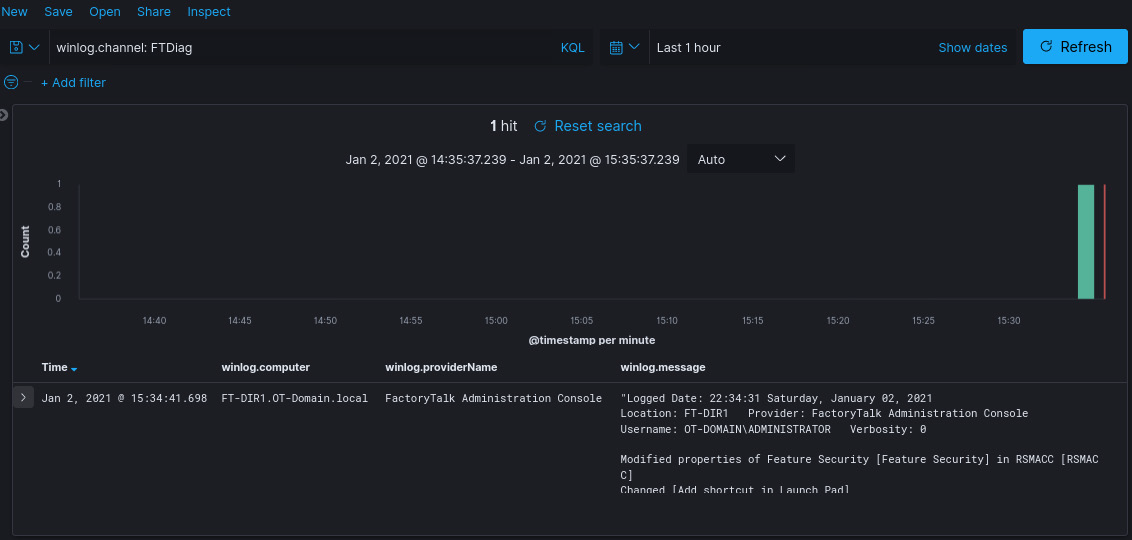
To show you an example, we will look at some event logs that are part of the FactoryTalk Diagnostics service. But first, after the Wazuh agent starts sending the FactoryTalk logs to Security Onion, we need to instruct Elasticsearch to update its **index patterns** to include the newly created attributes that are sent as part of the additional logs. To do this, in the Kibana web portal, click on the menu icon (top-left corner) and select **Stack Management**. Next, from the **Stack Management** page, navigate to **Index Patterns - \*:so-\*** and click on the **Refresh** button, then confirm you wish to refresh by clicking **Refresh** on the pop- up screen, as illustrated in the following screenshot:

*Figure 9.4 – Exercise 1: Refreshing Elasticsearch field list*

This will instruct Elasticsearch to update (refresh) the field list, which in turn allows fields to be queried (through search attributes).

We can now start searching within the refreshed data fields. To do this, click on the **Kibana** menu icon and select **Discover**. This brings up the **Discovery** page for Kibana. Here, we can search for data and view the raw logs as they are recorded in Elasticsearch.

With a search for **winlog.channel: FTDiag**, we can see several categories of event logs sent by the FactoryTalk Diagnostics logging service, as illustrated in the following screenshot:

*Figure 9.5 – Exercise 1: FactoryTalk Diagnostics example log entries*

In this small section of the logs returned, we can see valuable information such as a **login** by **OT-DOMAIN\ADMINISTRATOR** to the FactoryTalk directory, followed by a successful change to a feature property (**OT- DOMAIN\ADMINISTRATOR** changed to permit the **RSMACC** software to be opened through FactoryTalk Security).

Likewise, logs for other **industrial control systems** (**ICS**) vendors can be added in the same way—I will leave that as a homework assignment for you.

In this exercise, we learned how to extend the event logging capabilities of the ELK stack with the Wazuh host-based event log collection agent. More specifically, we added the many advanced capabilities of the Sysinternals Sysmon driver, as well as implementing ICS vendor-specific event log forwarding. These additional event log capabilities will allow us to discover security-related incidents and anomalies for a broader set of circumstances and technologies.

Next, we are going to look at how to add some insight into PowerShell scripts and commands.