Aurora EDR

Introduction

This room will introduce you to EDRs and Aurora, a Sigma-based tool for writing detection alerts via Windows Event logs.

Learning Objectives

The objectives of this room are:

- Introduce EDRs and their functionalities.
- Introduce Event Tracing for Windows.
- Learn about Aurora and its functionalities to write alerts using event logs.
- Investigate suspicious events detected by Aurora.

Prerequisites

It is advisable to check out the following module and rooms before embarking on this room.

- Cyber Defense Module
- Windows Event Logs
- Sysmon
- Intro to Detection Engineering
- Sigma Room

EDR Definition

What is an EDR?

Securing networks and organisations does not only rely on the monitoring of external connections or defence on the perimeter but also requires the use of tools and technologies to detect and respond to threats at the endpoints. An Endpoint Detection and Response (EDR) solution provides a proactive approach toward threat detection and visibility through near real-time monitoring of events on endpoints and evaluates them based on a rules-based automated response and analysis.

The main functions of an EDR are:

- Monitor and collect activity data from endpoints that could indicate a threat.
- Analyse this data to identify threat patterns.
- Automatically respond to identified threats, remove or contain them, and notify security personnel.
- Forensics and analysis tools to research identified threats and search for suspicious activities.

- The functions of an EDR can be broken down into the aggregation of data on an endpoint and how the data is analysed and used to detect threats.

Endpoint Data Recording

- Aggregating network communication, events, process executions, file activities, commands, user operations, etc.
- Telemetry data points.
- Storage of data on endpoints, in a server, or in a hybrid approach.

Investigation of Data & Responding

- Sweep (search) for indicators of Compromise to understand the impact of detections.
- Find the root cause of detection and remediate/prevent/investigate again.
- Hunt for indicators of Attack based on behaviour rules or threat intelligence. Automatic (detection) or manual.

Components of EDR solutions

EDR vendors would classify their capabilities differently. However, the following are the common classifications:

- Detection: Fundamentally, EDR solutions are tasked with threat detection. For example, with file analysis, EDRs can flag suspicious files at the sight of any malicious behaviour. The detection process is also based on how good the threat intelligence sources are.
- Response/ Containment: EDRs provide response features that help investigate, detect, remediate and contain threats. The actions here include host segmentation, file deletion/cleanup and conducting investigations through sandboxing conditions. Advanced EDR solutions have the capability to trigger an automated response based on a set of preconfigured rules.
- Integration: EDRs extend endpoint visibility through the collection and aggregation of data. Therefore, in addressing endpoint security, EDR solutions need to work smoothly with existing security solutions in an organisation.
- Insights: Real-time analysis of events is becoming very common, providing a rapid evaluation and correlation of threat data. Through complex machine learning and artificial intelligence algorithms, EDR solutions can automate threat identification and perform behavioural analysis, mapping them to frameworks such as the MITRE ATT&CK.
- Forensics: In-depth investigation of past threats provides valuable information on the inner workings of exploits and how a breach was successful. With this, EDR solutions can outline threat timelines and identify lurking threats that go undetected.

****

Answer the questions below:

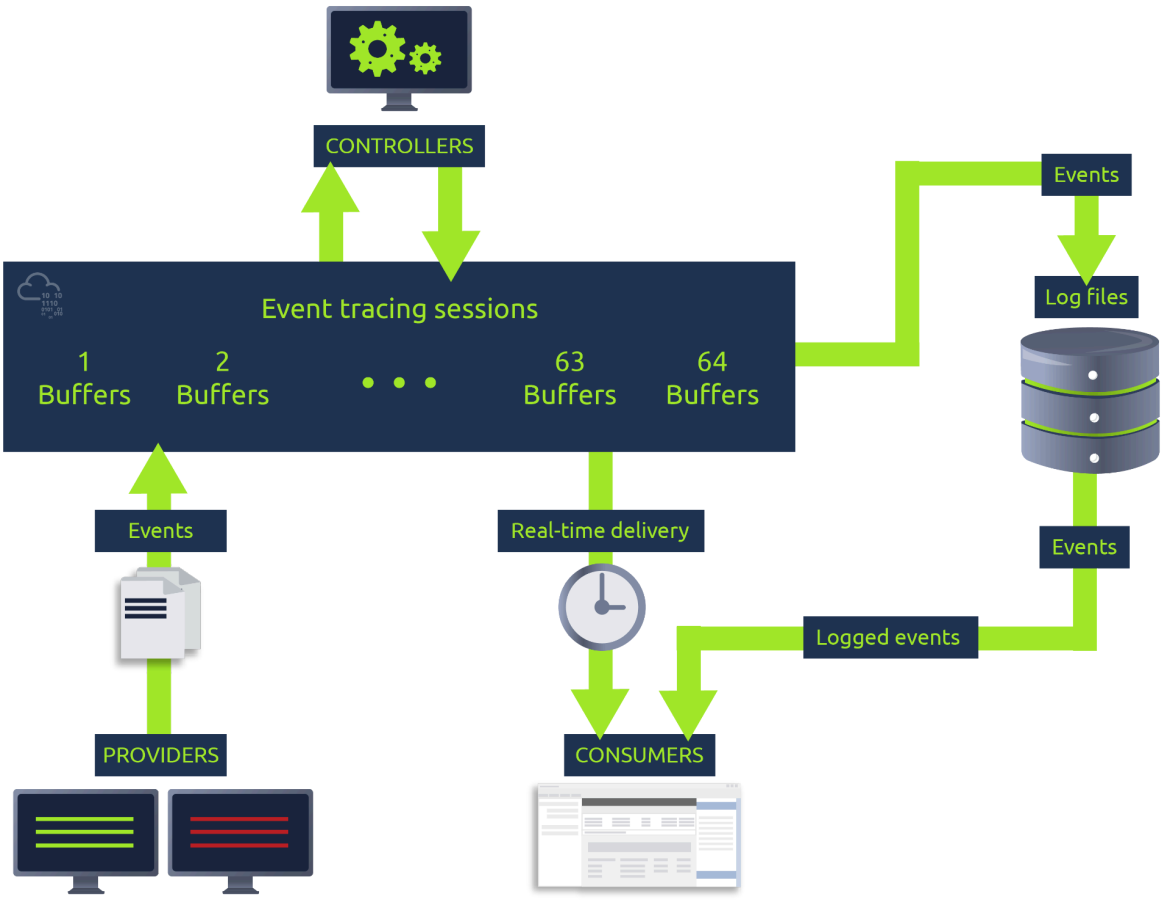
What does EDR stand for?
Answer: Endpoint Detection and Response

Event Tracing for Windows

Event Tracing for Windows (ETW) is a Windows OS logging feature that provides a mechanism to trace and log events raised by user-mode applications and kernel-mode drivers. ETW provides the capability for applications and drivers to write events. For cybersecurity defenders, this becomes a vital source of detection information.

ETW is made up of three distinct parts:

- Controllers: These applications are used to configure event tracing sessions. They also initiate the providers. An example of a Controller is logman.exe.
- Providers: These are the applications that produce event logs.
- Consumers: These applications subscribe and listen to events in real-time or from a file.



Windows Event Viewer

Windows systems and applications provide event logs that would be useful for troubleshooting and understanding the activities being performed. These logs include system access notifications, security changes, operating system errors, hardware failures, and driver malfunctions. We shall briefly examine how event logs are presented via the Windows Event Viewer.

The event information is categorised under these types of levels:

- Information: Describes the successful operation of a driver, application or service. Basically, a service is calling home.
- Warning: Describes an event that may not be a present issue but can cause problems in the future.
- Error: Describes a significant problem with a service or application.
- Success Audit: Outlines that an audited security access operation was successful. For example, a user's successful login to the system.
- Failure Audit: Outlines that an audited security access operation failed. For example, a failed access to a network drive by a user.

Even a properly functioning host will show various logs under these classes, and as a security analyst, you will be required to comb through the logs. This ensures you can keep tabs on a system's operations and troubleshoot any problems.

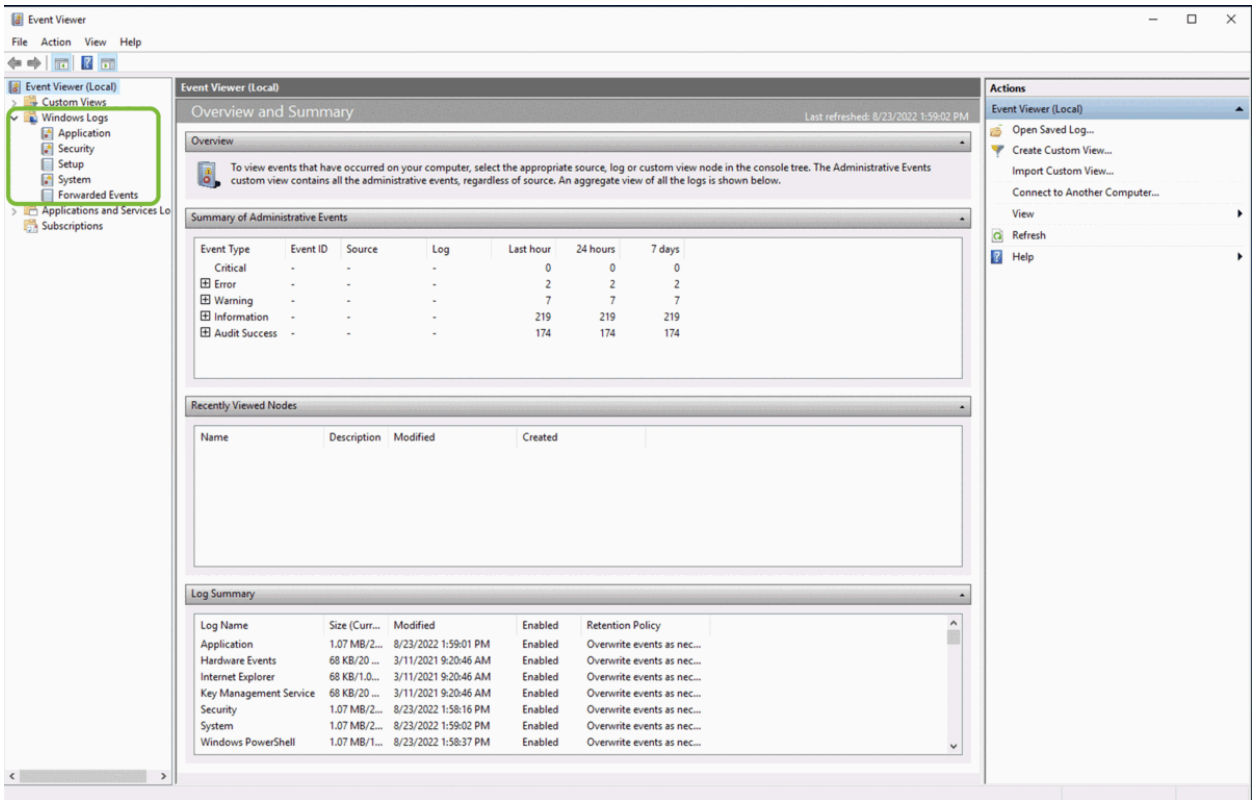
Using the Event Viewer

Windows Event Viewer is mainly found as an application on the system. We can find it simply by searching for "Event Viewer" on the Start menu.

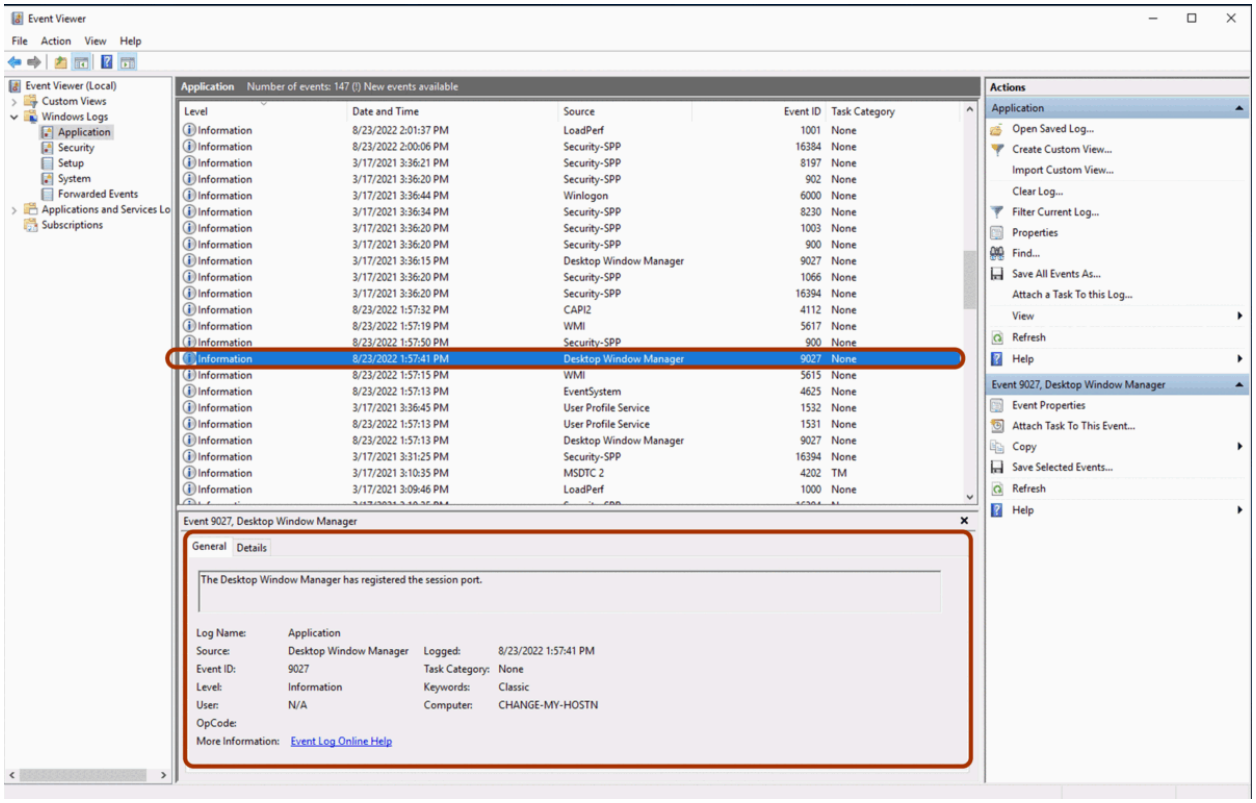
Windows logs are placed under different categories, with three major ones used for system troubleshooting and investigations:

- Application: Records log events associated with system components such as drivers and interface components that run an app.
- System: Records events related to programs installed and running on the system.
- Security: Records events associated with security, such as logon attempts and resource access.

On the main dialogue screen, we can see that the log events are presented in a tabular format which shows the levels, date and time, source of the events, event id and task category.



When we select an event, the event properties window displays information related to the event under the “General” tab. We can dig deeper via the “Details” tab.



Note that we shall focus on the application logs for the remainder of the room as Aurora writes its events in this category.

****

Answer the questions below:

Which applications produce event logs?
Answer: Providers

Applications that subscribe to event logs are called?
Answer: Consumers

Which event level would be used to describe a significant problem with a service?
Answer: Error

What is the Windows Eventlog category responsible for recording events associated with programs currently running called?
Answer: System

Aurora

Aurora is a Windows endpoint agent that uses Sigma rules and IOCs to detect threat patterns on local event streams using ETW. When a true-positive rule matches, Aurora triggers “response actions” that will be displayed under the Windows Event Log Viewer.

It has been designed to be customisable based on the Sigma rule set and function as an on-premises tool, with no data leaving the network.

Aurora comes in as an enterprise and free community version called Aurora Lite. The table below summarises the key differences in service offered by the two versions.

Aurora	**Aurora Lite**
Sigma-based event matching on ETW data	Sigma-based event matching on ETW data
An open-source rule set (1300+ rules)	An open-source rule set (1300+ rules)
Nextron’s Sigma rule set	No Nextron’s Sigma rule set
Open-source IOC set	Open-source IOC set
Nextron’s IOC set	No Nextron’s IOC set
Alert output channels: Eventlog, File, UDP/TCP	Alert output channels: Eventlog, File, UDP/TCP
Comfortable management with ASGARD	N/A
Additional detection modules	N/A
Unlimited number of response actions	N/A
Rule encryption	N/A

Aurora obtains data from different ETW channels and adds live information (for the commercial version) to enrich and recreate events similar to those generated by Sysmon. It does not create tons of logs; it only populates the viewer with events of triggered rules. Below, we can look at a comparison between Aurora and Sysmon.

	Aurora	Sysmon
Event Source	Event Tracing for Windows (ETW)	Sysmon Kernel Driver.
Sigma Rule Event Coverage	95%	100%
Relative Log Volume	Low	High
Sigma & IOC Matching	Yes	No
Response Actions	Yes	No
Resource Control (CPU Load, Output Throttling)	Yes	No
Output: Eventlog	Yes	Yes
Output: File	Yes	No
Output: TCP/UDP Target	Yes	No
Risk: Incomplete Data due to Filters	No	Yes
Risk: Blue Screen	No	Yes
Risk: High System Load	No	Yes

Aurora is supported on Windows 7/ Windows Server 2012 or newer versions and must run using administrator privileges. It must also be excluded from any running antivirus or EDR solutions. This is to avoid the application being blocked from executing its services.

Look at how to install and configure Aurora correctly via the User Manual.

Aurora Presets

Aurora can be configured to use four different configuration formats that dictate how the solution would fetch events and raise alerts. The four preset formats are:

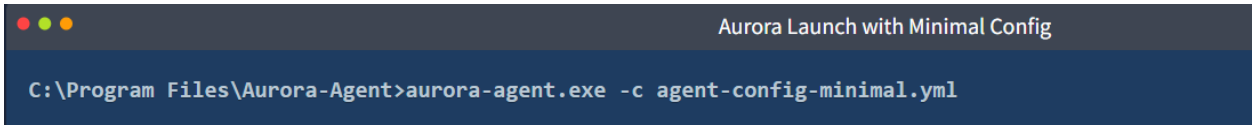
- Standard: This configuration covers events at a medium level of severity.
- Reduced: This configuration looks at events considered to be at a high minimum reporting level.
- Minimal: This configuration looks at events considered to be at a high minimum reporting level.

- Intense: This configuration looks at events considered to be at a low minimum reporting level.

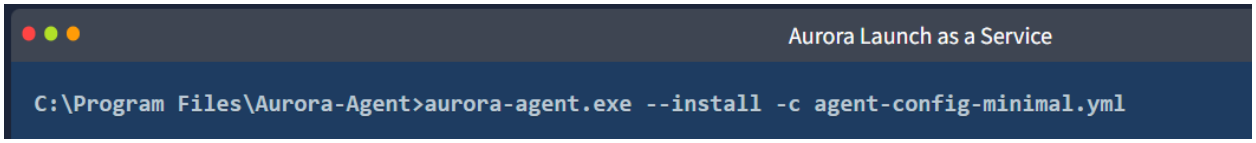
Affected Setting	Standard	Reduced	Minimal	Intense
Deactivated Sources	Registry, Raw Disk Access, Kernel Handles, Create Remote Thread	Registry, Raw Disk Access, Process Access	Registry, Raw Disk Access, Kernel Handles, Create Remote Thread, Process Access, Image Loads	-
CPU Limit	35%	30%	20%	100%
Process Priority	Normal	Normal	Low	Normal
Minimum Reporting Level	Medium	High	High	Low
Deactivated Modules	-	LSASS Dump Detector	LSASS Dump Detector, BeaconHunter	-

Running Aurora

Aurora can be started directly via the command line, with the option of selecting the preferred configuration.



For continuous running, the agent can also run as a service through the --install flag.



Useful Flags

Some valuable flags to use and query different types of information from Aurora include:

- -status: Queries status information from the currently running service.

Aurora Status

```
C:\Program Files\Aurora-Agent>aurora-agent.exe --status
Aurora Agent
Version 1.0.7
Build Revision: afae63fe69c55
Signature Revision: 2022/08/19-071122
Sigma Revision: 0.21-1677-g3584496d4
Status: running
Uptime (in hours): 44

Active Outputs:
    Windows Application Eventlog: enabled

Active Modules: ApplyIOCs, Rescontrol, Sigma, ETWSource, ETWKernelSource, EventlogSource, PollHandles

Rule Statistics:
    Rule paths: C:\Program Files\Aurora-Agent\signatures\sigma-rules, C:\Program Files\Aurora-Agent\custom-signatures
    Loaded rules: 1548
    Rule reloads: 0
    Responses: 0

False positive filters: 0
Process excludes: 0

Missed events: 0
Sigma matches: 0
Suppressed Sigma matches of those: 0

Response Actions: disabled
```

- –trace: Queries all the events Aurora monitors from the subscribed channels. It also provides complete event statistics.

Aurora Trace

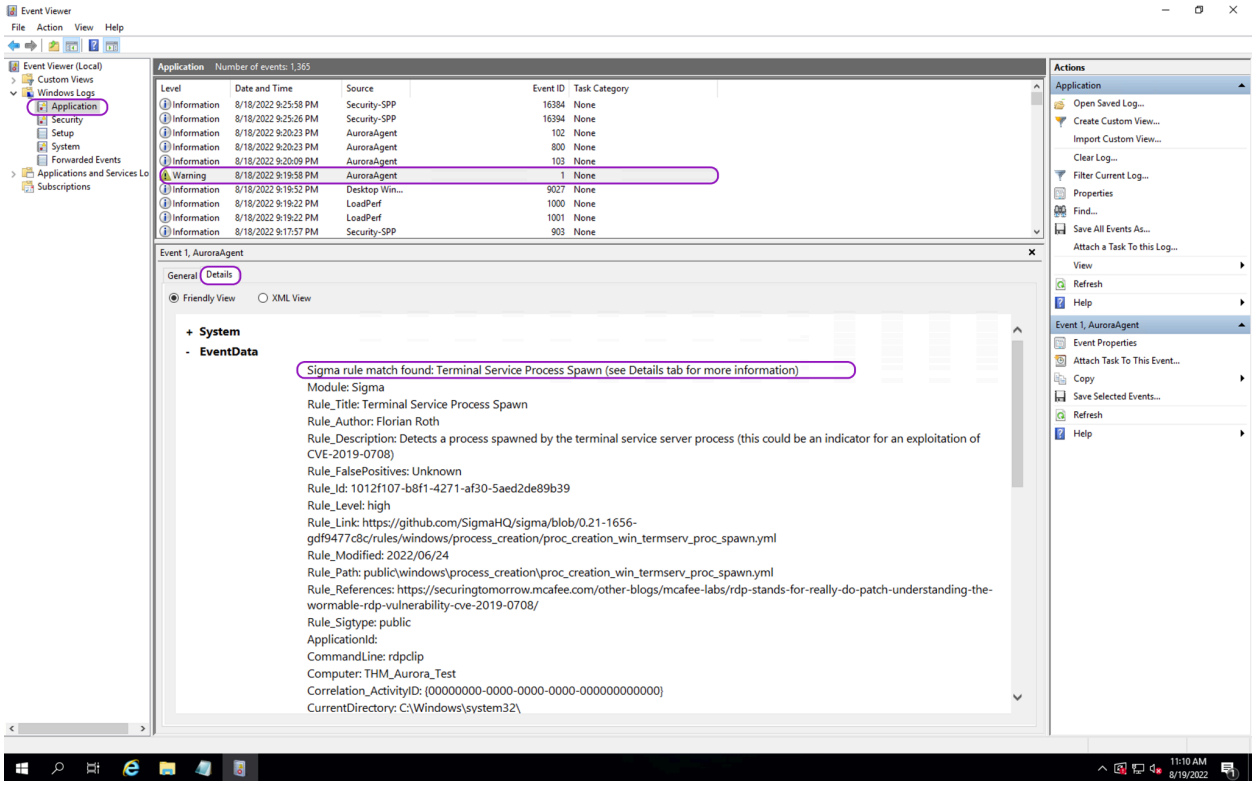
```
C:\Program Files\Aurora-Agent>aurora-agent.exe --trace > aurora-trace.log
```

- –json: Outputs information in JSON format for a more comprehensive view of the alerts that are easy to search.

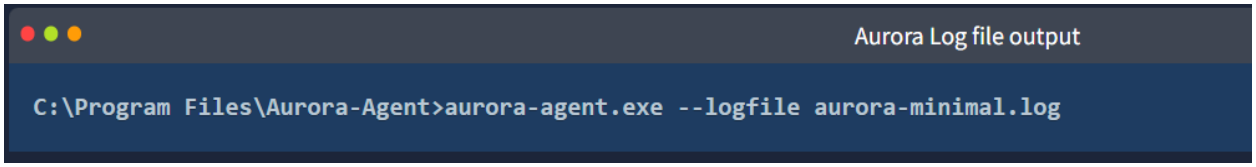
Output Options

Aurora supports the following output options:

- Windows Eventlog: On an earlier task, we looked at Event Tracing for Windows. Aurora writes its events using ETW; the details can be viewed via the EventViewer. Click the “Details” tab to see all fields and values.



- Log File: A log file can be written using the --logfile flag, which will be automatically rotated once the specified log size has been attained (by default, this is 10MB). The log files would be found under the directory where Aurora is running from.



With this command, the Aurora status will include Stdout: enabled under its configuration Active Outputs section.

- UDP/TCP Targets: Network targets direct Aurora events to an internal repository via UDP or TCP using the flags --udp-target and --tcp-target. These options require arguments in the form of host: port, such as internal.repo:8443.

Aurora Responses

Responses extend the Sigma services and can be used to set specific actions to be performed and respond when an event matches. These actions can help contain a threat actor or limit their damage to the system host. The intended use cases for the responses are worm containment, ransomware containment, and the hard blocking of applications.
Aurora supports two responses: Predefined and Custom.

Predefined Responses
The following responses are available by default with the installation of Aurora:

- Suspend: Used to stop a specified process.
- Kill: Used to kill a specified process.
- Dump: Used to create a dump file found in the dump-path folder.

Custom Responses
Custom responses are meant to call an internal program to execute a function based on a raised alert. The program has to be available from PATH and the answer would be a command-line query.

With these responses, a set of flags can be used to modify and relay different types of information. The flags have been summarised in the table below:

Flag	**Definition**
Simulate	Used to test out rules, and responses that won't be triggered. A log will be created to indicate the type of response that would be triggered.
Recursive	It is used to specify that the response will affect descendent processes. It is usually true by default.
Low privilege only	Marked by the flag lowprivonly. The flag specifies that the response will be triggered if the target process does not run as LOCAL SYSTEM or at an elevated role.
Ancestor	The ancestors flag specifies that the response will affect a process's ancestor, not itself. The key: value pair is indicated by integers to show the level of ancestors, e.g. one (1) is for parent process, 2 for grand-parent, and so on.
Process ID field	The processidfield flag specifies the field contains the process ID that shall be affected by the response.

Response examples

Aurora Kill Parent Process Response

```
response:
    type: predefined
    action: kill
    processidfield: ParentProcessId
```

Aurora Suspend Response

```
response:
    type: predefined
    action: suspend
```

Aurora Copy image to backup folder Custom Response

```
response:
    type: custom
    action: cmd /c copy %Image% "%%ProgramData%%\Aurora\Image-%ProcessId%.bin"
```

Aurora Event IDs

If you may have noticed through the screenshots and navigating through the VM, Aurora uses event IDs to log to the Windows Eventlog. The tables below list the IDs related to Sigma, internal and other notable modules used.

Event ID	Description	EventID	Description
1	A process creation Sigma rule matched.	**100**	A license file was found.
2	A set file creation time sigma rule matched.	**101**	Status message (from --report-stats)
3	A network connection sigma rule matched.	**102**	Aurora Agent started.
4	A sysmon status Sigma rule matched.	**103**	Aurora Agent is terminating.

5	A process termination Sigma rule matched.	**104**	The current license expired.
6	A driver-loaded Sigma rule matched.	**105**	No valid license file was found.
7	An image-loaded Sigma rule matched.	**107**	A process created a large number of events.
8	An image-loaded Sigma rule matched.	**108**	An internal panic occurred.
9	A raw disk access Sigma rule matched.	**200**	BeaconHunter
10	A process access Sigma rule matched.	**300**	Lsass Dump Detector
11	A file creation Sigma rule matched.	**400**	ETW Canary
12	A registry event Sigma rule matched.	**500**	Process Tampering Detector
15	A create stream hash Sigma rule matched.	**600**	Temporary Driver Load Detector
17	A pipe event Sigma rule matched.	**700**	Command Line Mismatch Detector
19	A WMI event Sigma rule matched.	**800**	Event Distributor
21	A registry event	**900**	ETW

	Sigma rule matched.		Provider
22	A DNS query Sigma rule matched.	**1000**	Eventlog Provider
23	A file deletion Sigma rule matched.	**1100**	Handle Polling Provider
95	An error occurred while loading the Sigma rules.	**1200**	Resource Control
96	Sigma rules were reloaded.		
97	No Sigma rule files were found.		
98	Unspecified log message from Sigma module.		
99	Another Sigma rule (that did not belong to one of the above categories) matched.		
6000	A response for a sigma match was executed.		
6001	A response for a sigma match was simulated.		

****

Answer the questions below:

Which Aurora preset supports the highest CPU limit?
Answer: intense

Which process would be terminated when the response flag ancestors:3 is used?

Answer: great grandparent

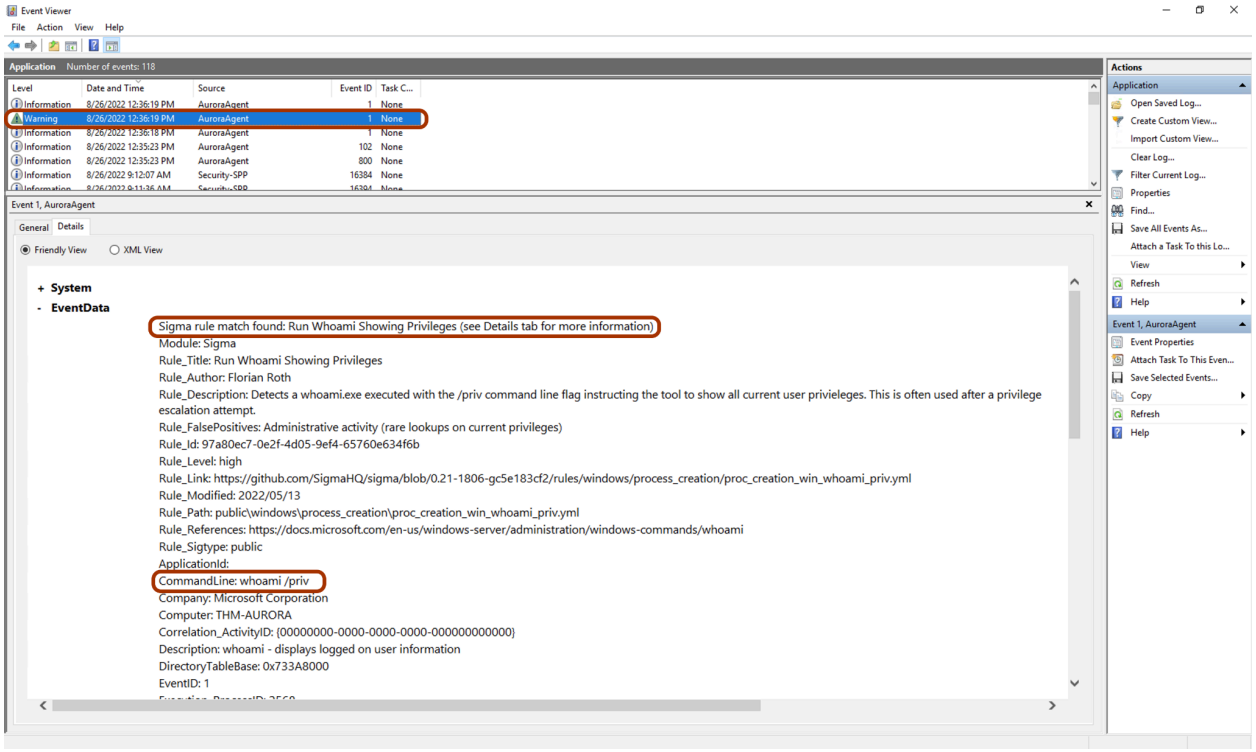
When the Aurora agent is terminated, which event id will be used?
Answer: 103

Aurora Function Tests

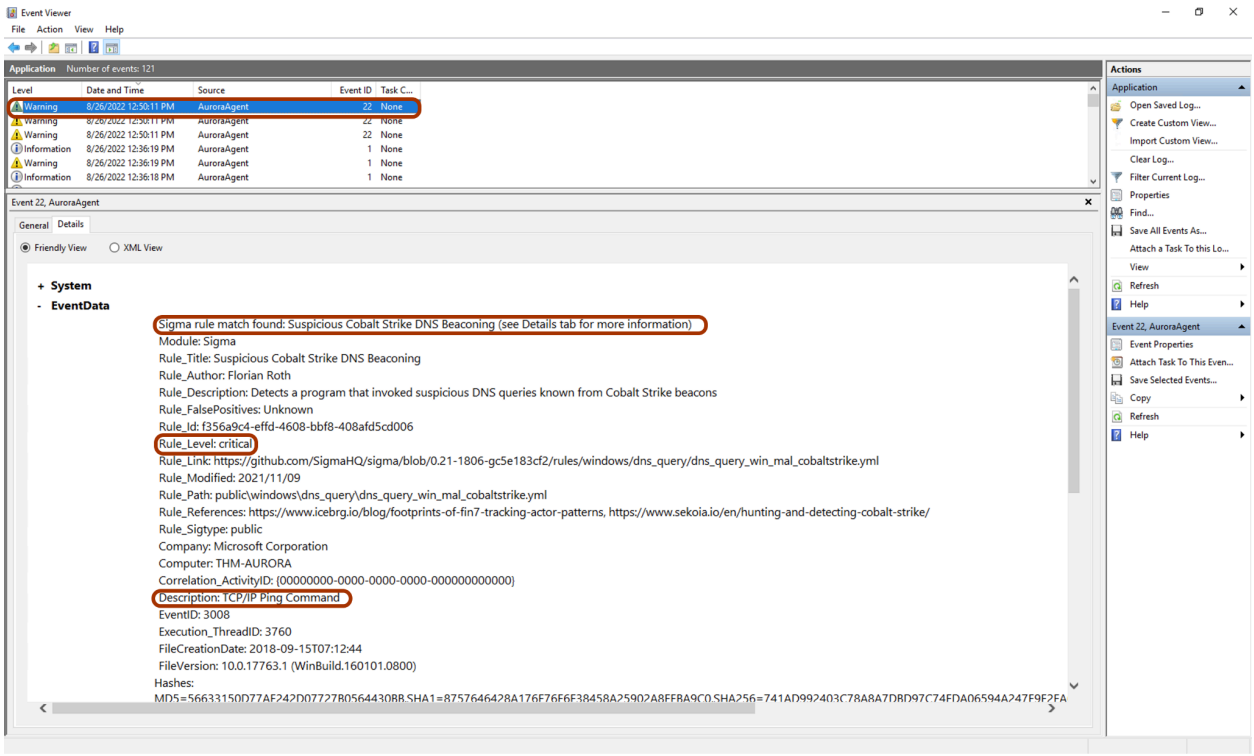
Function Tests

Once Aurora has been installed and configured, several function tests can be executed to check its various functionalities. Let's look at a few Sigma matching and IOC matching examples.

- Listing user account privileges: Running a simple command whoami /priv to list the current user privileges will trigger a Sigma rule with a level high and create a WARNING level message on the Eventlog. This will be a process creation alert.



- Suspicious network communication: Running a suspicious DNS beaconing request will result in a critical rule being triggered. Below is an example that matches a suspicious cobalt strike DNS beaconing.



****

Answer the questions below:

Read the above.

No answer needed

Aurora Detection Gaps

Since Aurora uses ETW to observe and monitor Windows system events, there are some sections where ETW events aren't available or not easily usable. These areas cover Aurora's detection gaps, and we shall look at them in this task.

Named Pipes

Named pipes are a one-way communication channel between processes that are subject to security checks in the memory. ETW has no provider to gather information about the creation or connection to named pipes. Observing named pipe events is through the Kernel Object Handle, which is noisy and can provide unnecessary information.

Solution

- Using Aurora under "Intense" configurations.
- Complement Aurora configuration with Sysmon to capture the events.

Registry Events

Registry events are generated on the ETW by creating keys or writing values, primarily via the Microsoft-Windows-Kernel-Registry. However, this information may not be directly usable as all registry handles must be tracked individually as keys are referenced by their handle. For value setting too.

Solution

- Using Aurora under “Intense” configurations.
- Complement Aurora configuration with Sysmon to capture the events.

ETW Disabled

Attackers have the ability to disable ETW events by patching the system calls that Windows would use to create the events from user space. Writing detection rules based on events that originate from the process and are caused by a provider that is not Microsoft-Windows-Kernel should be done with care.

Solution

- Aurora’s full version uses the ETW Canary module to detect any manipulations of ETW.
- Using the flag --report-stats allows for reporting the agent’s status to your SIEM and will include stats of the observed, processed and dropped events that can indicate signs of manipulations.

****

Answer the questions below:

Read the above.

No answer needed

Interactive Scenario

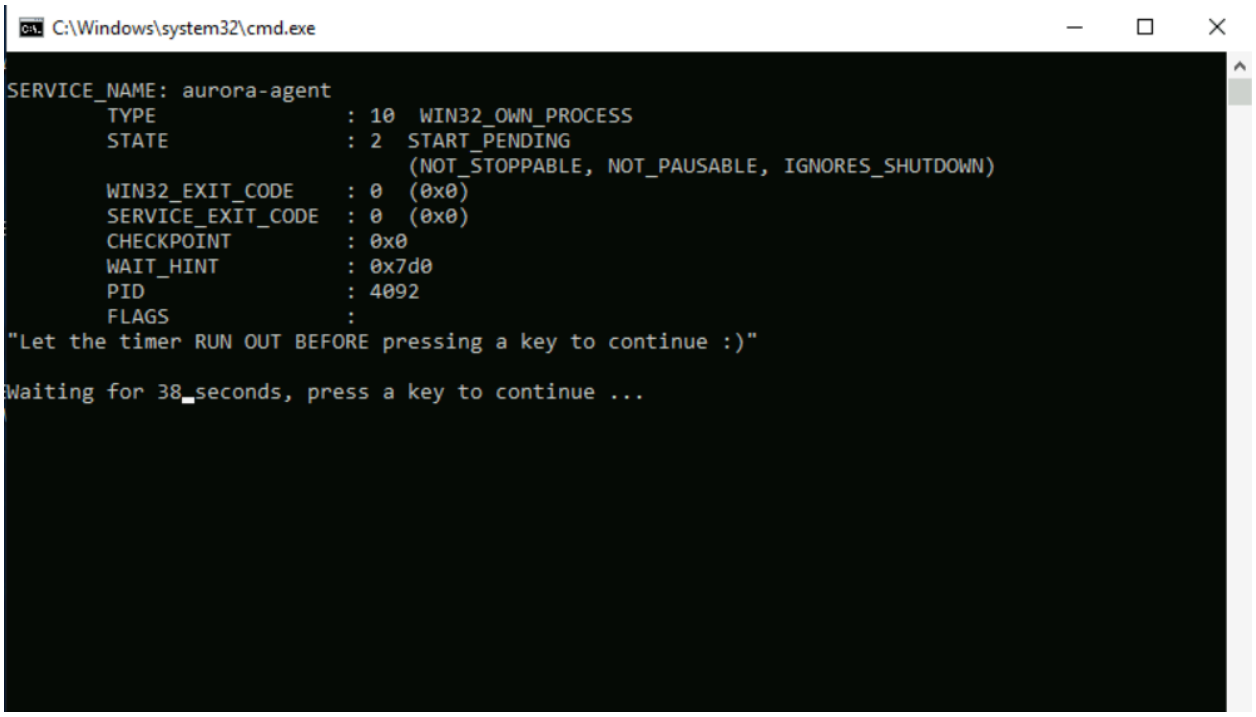
Recently, a series of suspicious activities have been observed within THM_Acme's systems. The CISO has recently heard about Aurora and has tasked you to carry out tests on one of the Windows hosts and assess its detection capabilities. Aurora has been installed on the Windows host. However, it is not running. Execute the batch file attached on the Desktop to launch and investigate any produced events starting with Event ID: 1 associated with Aurora to answer the questions below. In case no events appear, rerun the batch file.

You may also use the following information if you prefer accessing the machine via RDP:

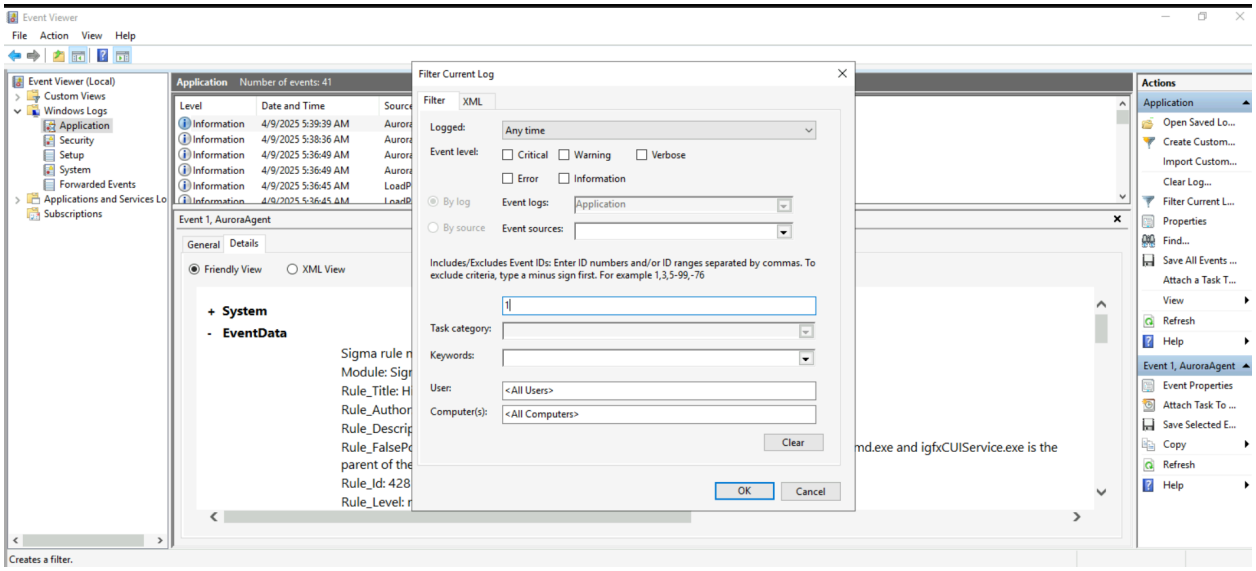
Machine IP: MACHINE_IP
User: Administrator
Password: THM_AUR0RA

****

Answer the questions below:

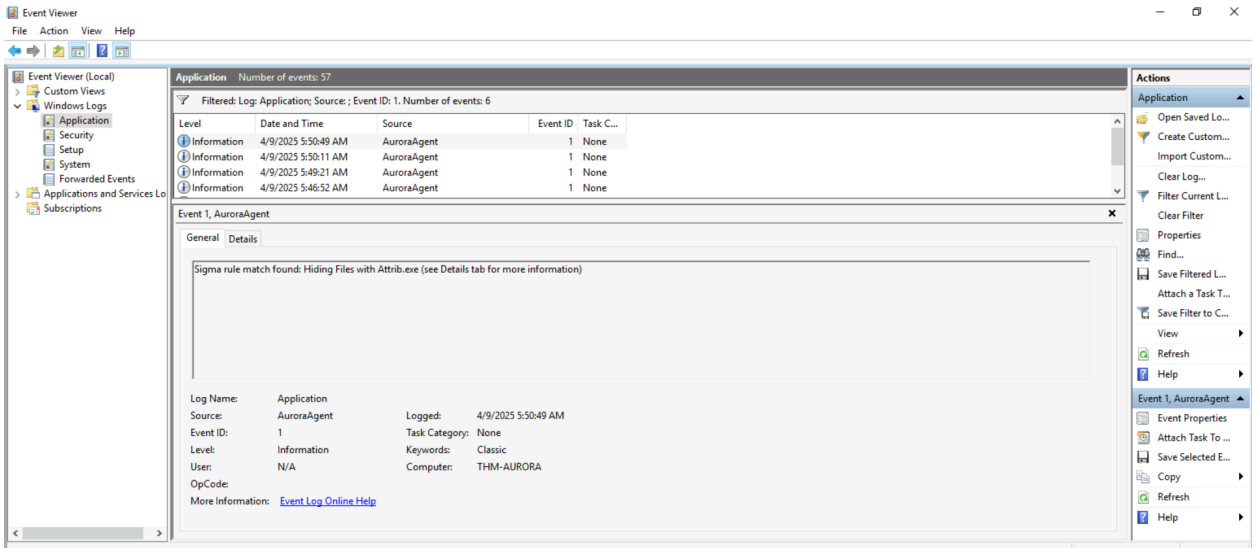


First run the .bat file as an administrator.



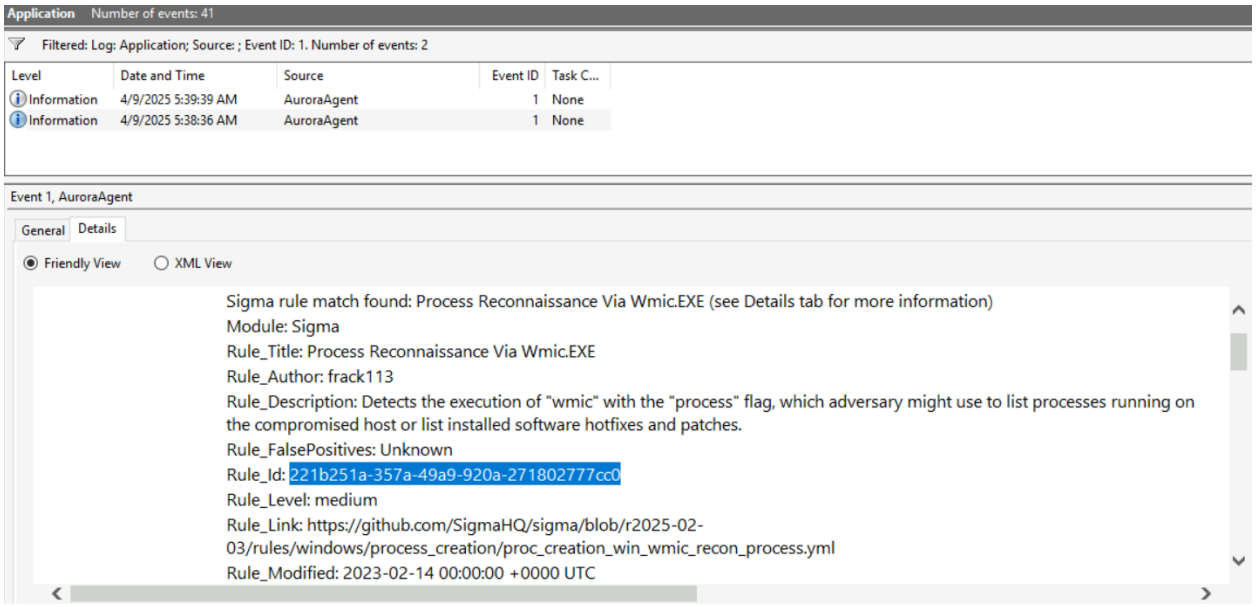
Open event viewer, go to Application Logs, and filter for Event ID 1.

What is the title of the first event rule?



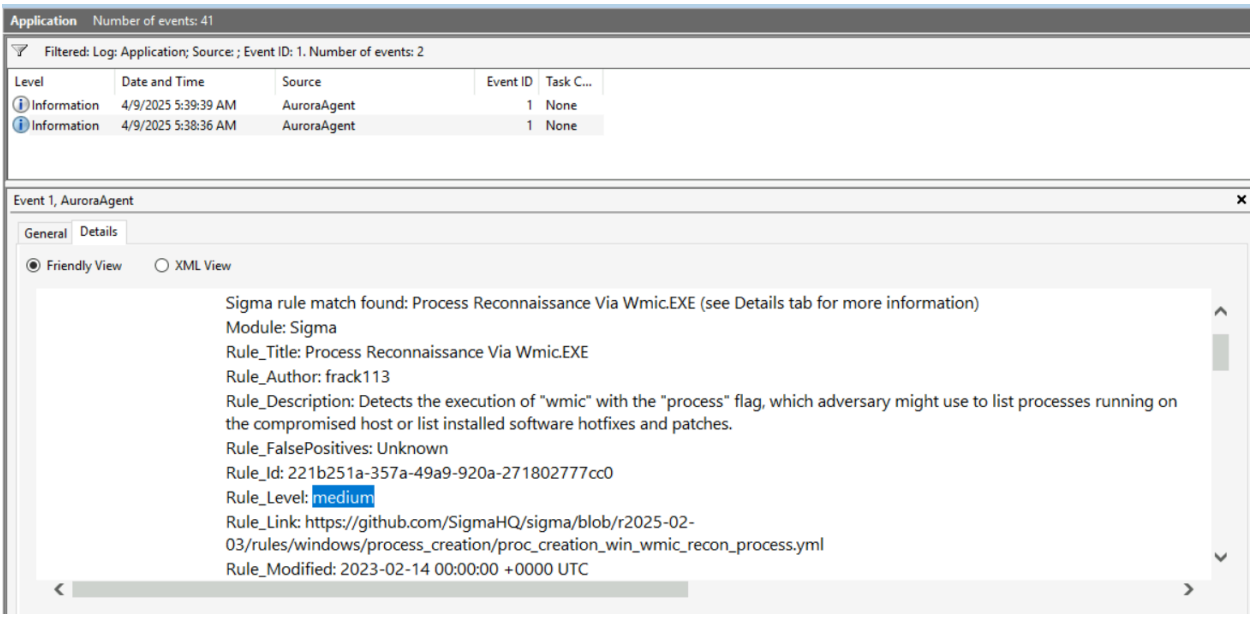
Answer: Process Reconnaissance Via Wmic.EXE

What is the Rule ID of the matched Sigma rule?



Answer: 221b251a-357a-49a9-920a-271802777cc0

What is the Sigma rule level of the event?



Answer: medium

What is the Rule Title of the second Event?

For some reason I couldn't get this event to pop up. I only got the two events of Process Reconnaissance Via Wmic.EXE and Hiding Files with Attrib.exe so I had to look this answer up online.

Answer: Suspicious Creation TXT File in User Desktop

Based on the Rule's characteristics, what malicious activity is associated with the event?

Answer: Ransomware

Conclusion