Wazuh

Task 1 Introduction

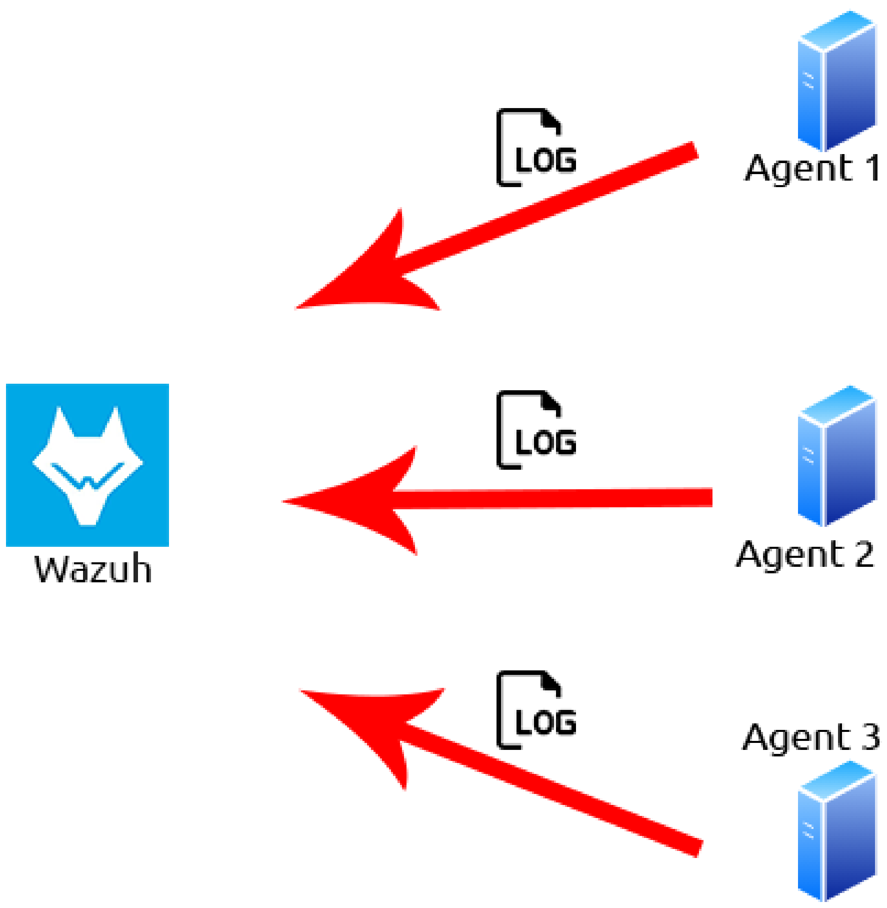
Welcome to a room showcasing the capabilities of the Wazuh EDR software solution. In this room, you can expect to learn the following things:

- What is an EDR and why are they useful solutions
- Where an EDR like Wazuh is used
- Accessing Wazuh
- Navigating Wazuh
- Learning about Wazuh rules and alerts
- Digesting logs to view specific events on devices including Linux and Windows
- How you can extend Wazuh using plugins and its API

Firstly, let's understand what EDR solutions are exactly. Endpoint detection and response (EDR) are a series of tools and applications that monitor devices for an activity that could indicate a threat or security breach. These tools and applications have features that include:

- Auditing a device for common vulnerabilities
- Proactively monitoring a device for suspicious activity such as unauthorised logins, brute-force attacks or privilege escalations
- Visualising complex data and events into neat and trendy graphs
- Recording a device's normal operating behaviour to help with detecting anomalies

Created in 2015, Wazuh is an open-source, freely available and extensive EDR solution. It can be used in all scales of environments. Wazuh operates on a management and agent module. Simply, a device is dedicated to running Wazuh named a manager, where Wazuh operates on a management and agent model where the manager is responsible for managing agents installed on the devices you'd like to monitor. Let's look at this model in the diagram below:



We can see logs from three Agents being sent to the Wazuh server.

Answer the questions below

When was Wazuh released?

2015

What is the term that Wazuh calls a device that is being monitored for suspicious activity and potential security threats?

Agent

Lastly, what is the term for a device that is responsible for managing these devices?

Manager

Task 2 **Required: Deploy Wazuh Server**

Connect to the TryHackMe network and deploy the Wazuh management server attached to this task and wait a **minimum of five minutes** before visiting the Wazuh server on HTTP://10.10.225.28.

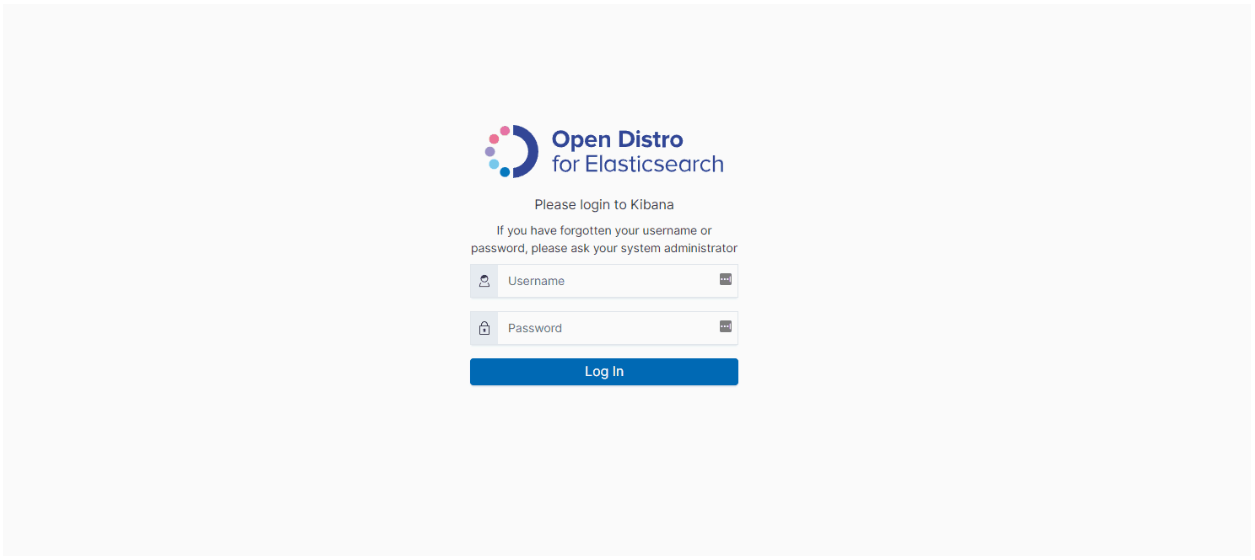
If you load the Wazuh management server too early, it will say "Kibana Server is not ready yet" Please wait a few more minutes before refreshing the page and trying again.

Once it has started, log in using the following credentials:

Username: wazuh (**make sure that this is lowercase!**)

Password: eYa0M1-hG0e7rjGi-lRB2qGYVoonsG1K

Select "**Global Tenant**" after successfully logging in. Refer to the animated gif below of the process if you are stuck.


Open Distro
for Elasticsearch
Please login to Kibana
If you have forgotten your username or password, please ask your system administrator
Username
Password
Log In

Note: The questions within the tasks of this room **will expect the data stored on this Wazuh management server**, so it is vital that you are able to connect to this server before continuing.

The Wazuh management server in this room will show the agents as being *disconnected* - this is expected.

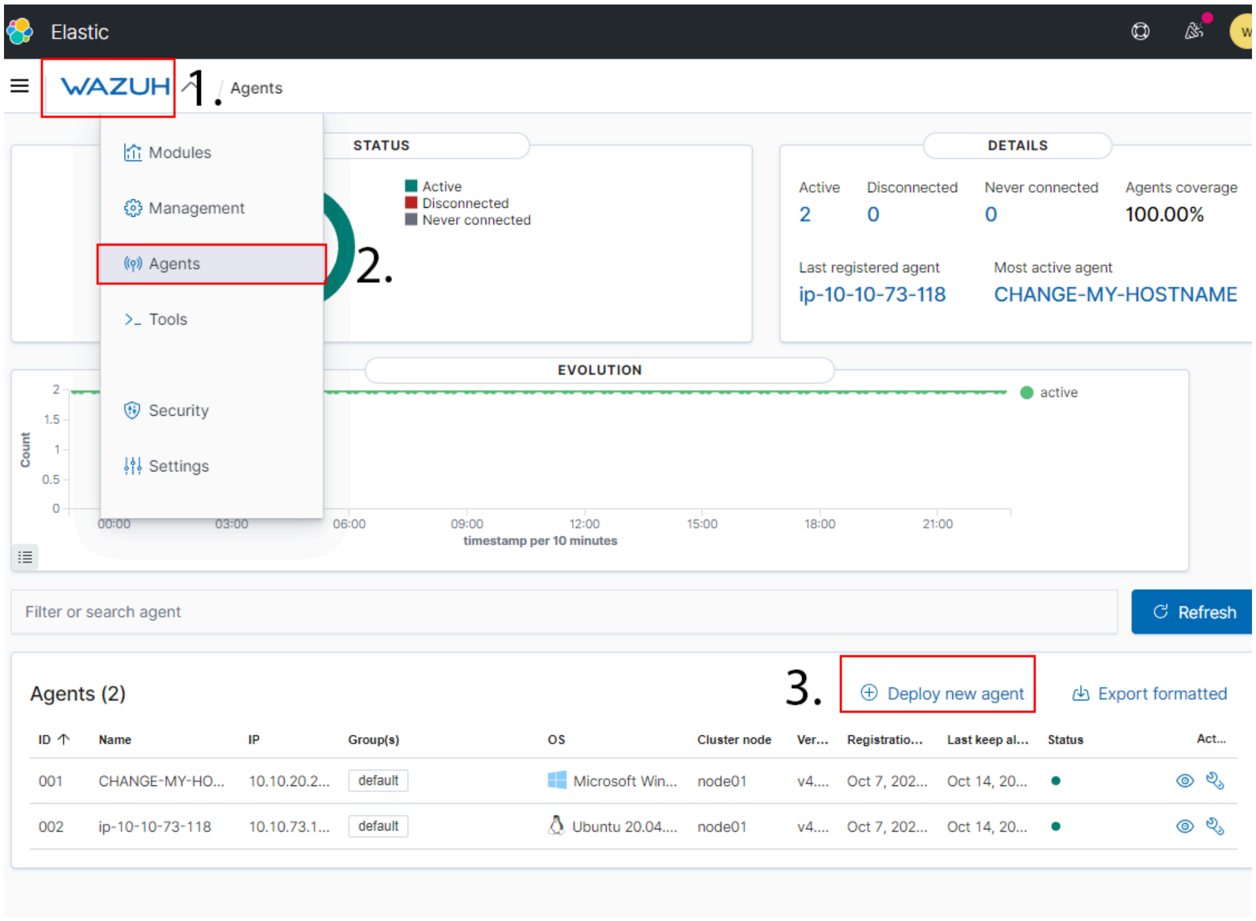
Task 3 Wazuh Agents

Devices that record the events and processes of a system are called agents. Agents monitor the processes and events that take place on the device, such as authentication and user management. Agents will offload these logs to a designated collector for processing, such as Wazuh.

In order for Wazuh to be populated, agents need to be installed onto devices to log such events. Wazuh can guide you through the agent deployment process provided you fill out some pre-requisites such as::

- Operating System
- The address of the Wazuh server that the agent should send logs to (this can be a DNS entry or an IP address)
- What group the agent will be under - you can sort agents into groups within Wazuh if you wish

This wizard can be launched by navigating to the following location on the Wazuh server: **Wazuh -> Agents -> Deploy New Agent** as illustrated in this screenshot below:



Active	Disconnected	Never connected	Agents coverage
2	0	0	100.00%

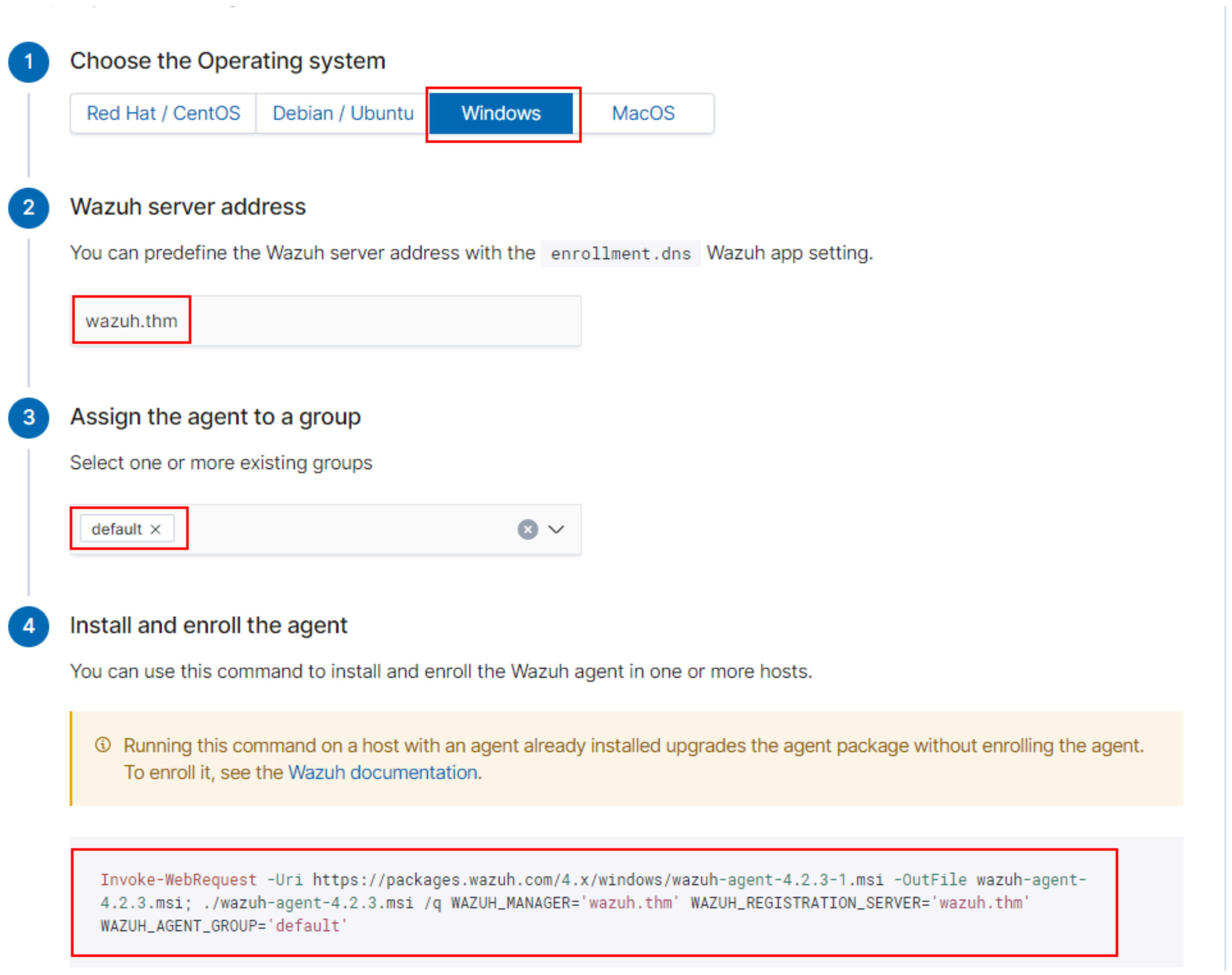
Last registered agent: ip-10-10-73-118
Most active agent: CHANGE-MY-HOSTNAME

Agents (2)

ID	Name	IP	Group(s)	OS	Cluster node	Ver...	Registratio...	Last keep al...	Status	Act...
001	CHANGE-MY-HO...	10.10.20.2...	default	Microsoft Win...	node01	v4....	Oct 7, 202...	Oct 14, 20...	●	
002	ip-10-10-73-118	10.10.73.1...	default	Ubuntu 20.04....	node01	v4....	Oct 7, 202...	Oct 14, 20...	●	

Once you navigate to this display, the intuitive wizard will be available to you. I have shared screenshots of using the wizard to install Wazhur's agent on both Windows and Debian/Ubuntu. At stage 4, you are given a command to copy and paste to your clipboard which will install & configure the agent on the device that you wish to collect logs from.

Installing the Wazuh agent on Windows:


1
Choose the Operating system
Red Hat / CentOS
Debian / Ubuntu
Windows
MacOS
2
Wazuh server address
3
Assign the agent to a group
4
Install and enroll the agent

You can predefine the Wazuh server address with the `enrollment.dns` Wazuh app setting.

wazuh.thm

Select one or more existing groups

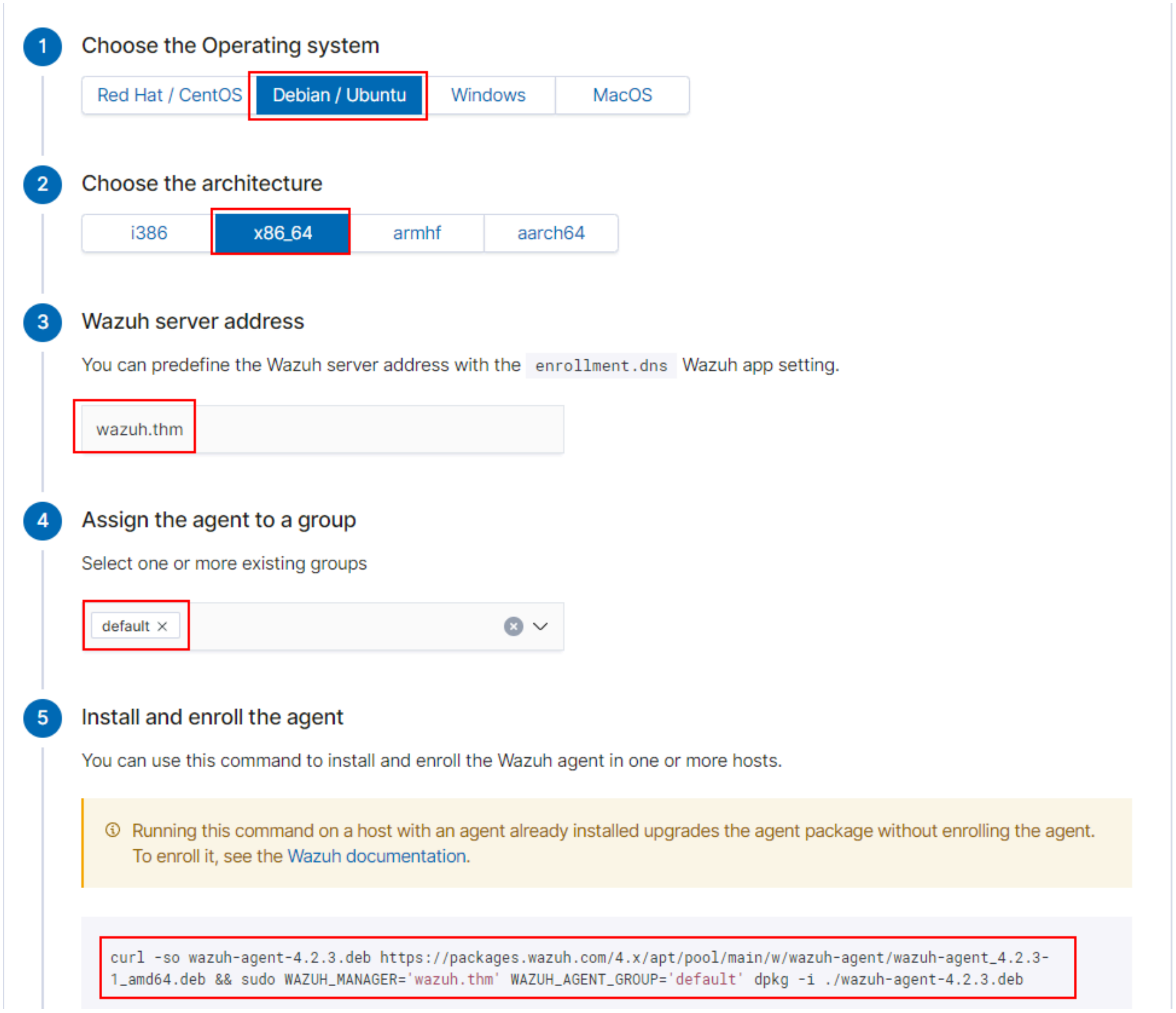
default

You can use this command to install and enroll the Wazuh agent in one or more hosts.

Running this command on a host with an agent already installed upgrades the agent package without enrolling the agent. To enroll it, see the Wazuh documentation.

```
Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.2.3-1.msi -OutFile wazuh-agent-
4.2.3.msi; ./wazuh-agent-4.2.3.msi /q WAZUH_MANAGER='wazuh.thm' WAZUH_REGISTRATION_SERVER='wazuh.thm'
WAZUH_AGENT_GROUP='default'
```

Installing the Wazuh agent on Debian/Ubuntu:


1 Choose the Operating system

Red Hat / CentOS | Debian / Ubuntu | Windows | MacOS

2 Choose the architecture

i386 | x86_64 | armhf | aarch64

3 Wazuh server address

You can predefine the Wazuh server address with the `enrollment.dns` Wazuh app setting.

wazuh.thm

4 Assign the agent to a group

Select one or more existing groups

default

5 Install and enroll the agent

You can use this command to install and enroll the Wazuh agent in one or more hosts.

Running this command on a host with an agent already installed upgrades the agent package without enrolling the agent. To enroll it, see the Wazuh documentation.

```
curl -so wazuh-agent-4.2.3.deb https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.2.3-1_amd64.deb && sudo WAZUH_MANAGER='wazuh.thm' WAZUH_AGENT_GROUP='default' dpkg -i ./wazuh-agent-4.2.3.deb
```

Answer the questions below

How many agents does this Wazuh management server manage?

2

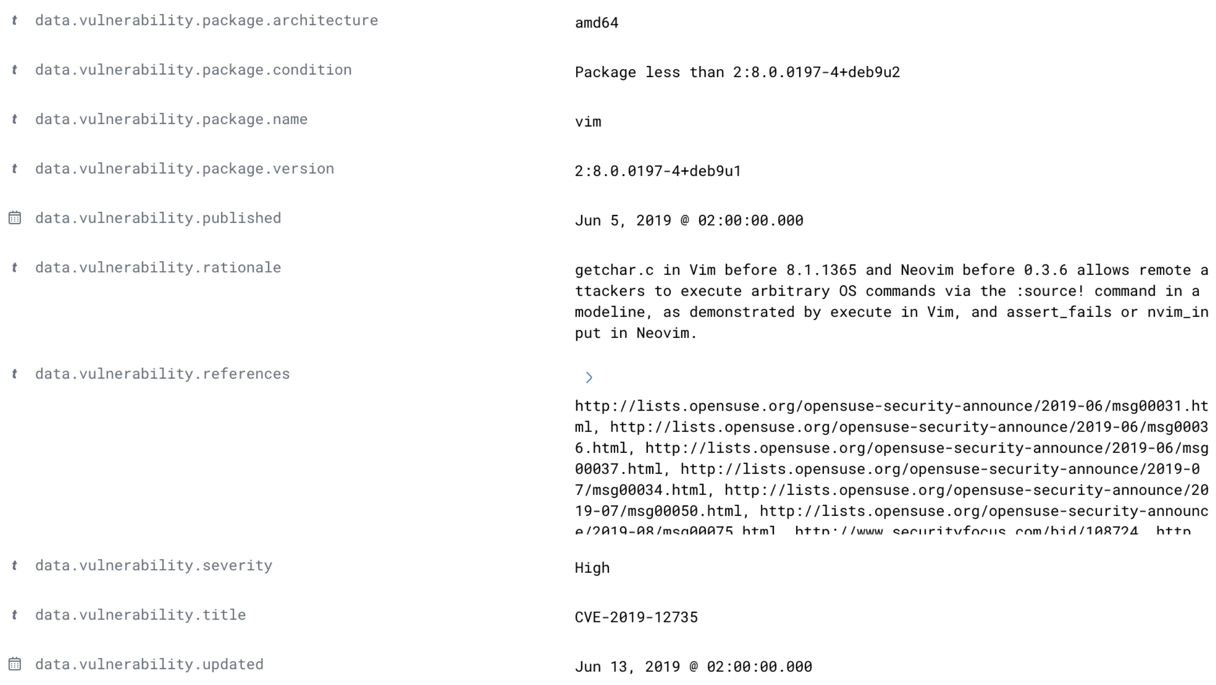
What are the status of the agents managed by this Wazuh management server?

Disconnected

Task 4 Wazuh Vulnerability Assessment & Security Events

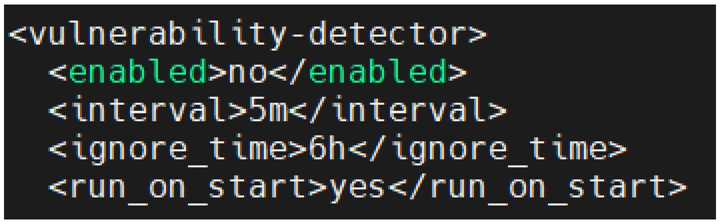
Wazuh's Vulnerability Assessment module is a powerful tool that can be used to periodically scan an agent's operating system for installed applications and their version numbers.

Once this information has been gathered, it is sent back to the Wazuh server and compared against a database of CVEs to discover potential vulnerabilities. For example, the agent in the screenshot below has a version of Vim that is vulnerable to **CVE-2019-12735**.



data.vulnerability.package.architecture	amd64
data.vulnerability.package.condition	Package less than 2:8.0.0197-4+deb9u2
data.vulnerability.package.name	vim
data.vulnerability.package.version	2:8.0.0197-4+deb9u1
data.vulnerability.published	Jun 5, 2019 @ 02:00:00.000
data.vulnerability.rationale	getchar.c in Vim before 8.1.1365 and Neovim before 0.3.6 allows remote attackers to execute arbitrary OS commands via the :source! command in a modeline, as demonstrated by execute in Vim, and assert_fails or nvim_input in Neovim.
data.vulnerability.references	http://lists.opensuse.org/opensuse-security-announce/2019-06/msg00031.html, http://lists.opensuse.org/opensuse-security-announce/2019-06/msg00036.html, http://lists.opensuse.org/opensuse-security-announce/2019-06/msg00037.html, http://lists.opensuse.org/opensuse-security-announce/2019-07/msg00034.html, http://lists.opensuse.org/opensuse-security-announce/2019-07/msg00050.html, http://lists.opensuse.org/opensuse-security-announce/2019-08/msg00075.html, http://www.securityfocus.com/bid/108724, http
data.vulnerability.severity	High
data.vulnerability.title	CVE-2019-12735
data.vulnerability.updated	Jun 13, 2019 @ 02:00:00.000

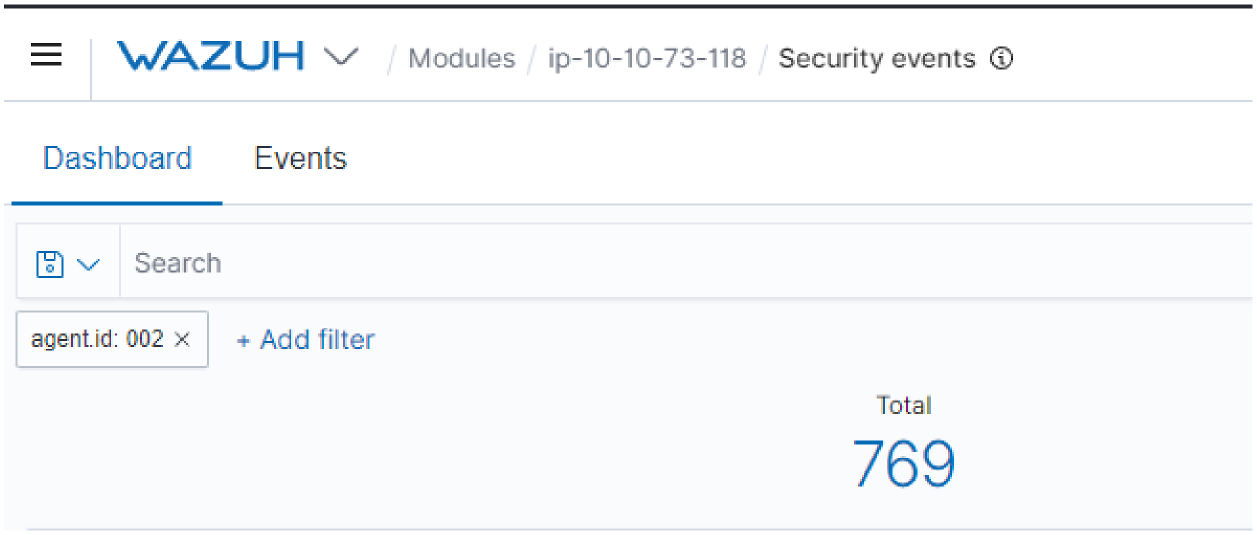
The vulnerability scanner module will perform a full scan when the Wazuh agent is first installed on a device and **must** be configured to run at a set interval then after (by default, this is set to 5 minute intervals when enabled) like so:



```
<vulnerability-detector>
  <enabled>no</enabled>
  <interval>5m</interval>
  <ignore_time>6h</ignore_time>
  <run_on_start>yes</run_on_start>
```

Configuring the Wazuh management server to audit agents for vulnerabilities frequently (/var/ossec/etc/ossec.conf)

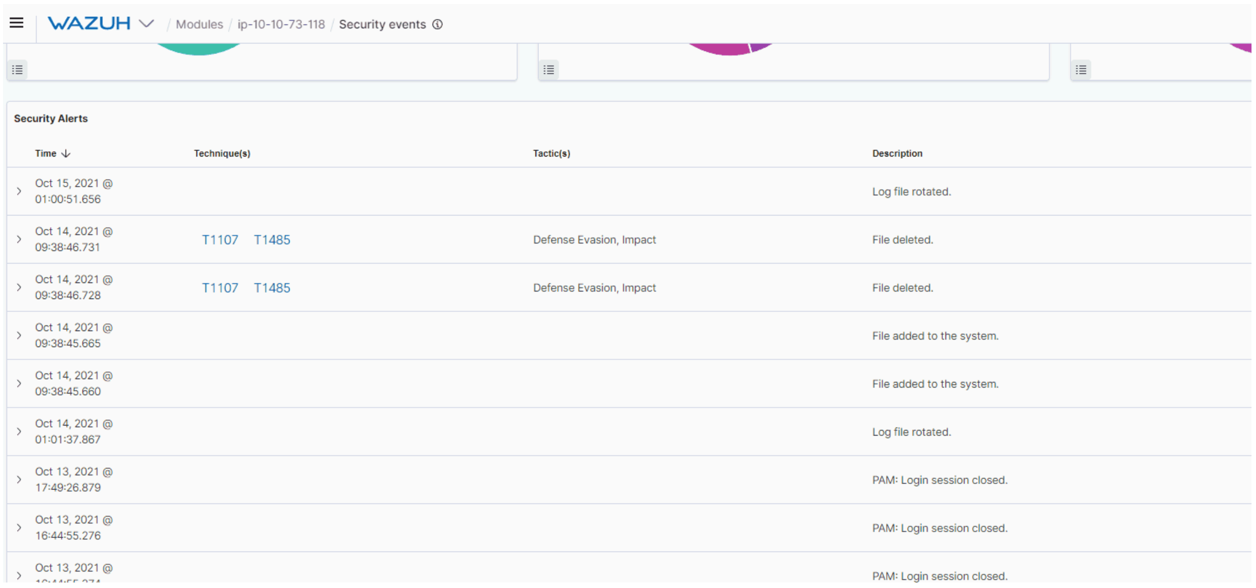
Wazuh is capable of testing an agent's configuration against certain rulesets to check for compliance. However, out of the box, it is arguably sensitive. Take, for example, this Linux host running the Wazuh agent. There have been a total of 769 events occurring that the system performs as part of its daily maintenance



Total
769

These frequent actions, such as removing files, are often detected as a security event. These events and the related severities are determined by Wazuh's rulesets, which is something that we will come on to explore adjusting in another task.

We can analyze these events individually by selecting the event's dropdown. You can sort events based upon various factors such as timestamp, tactics, or description.



Security Alerts

Time	Technique(s)	Tactic(s)	Description
Oct 15, 2021 @ 01:00:51.656			Log file rotated.
Oct 14, 2021 @ 09:38:46.731	T1107 T1485	Defense Evasion, Impact	File deleted.
Oct 14, 2021 @ 09:38:46.728	T1107 T1485	Defense Evasion, Impact	File deleted.
Oct 14, 2021 @ 09:38:45.665			File added to the system.
Oct 14, 2021 @ 09:38:45.660			File added to the system.
Oct 14, 2021 @ 01:01:37.867			Log file rotated.
Oct 13, 2021 @ 17:49:26.879			PAM: Login session closed.
Oct 13, 2021 @ 16:44:55.276			PAM: Login session closed.
Oct 13, 2021 @			PAM: Login session closed.

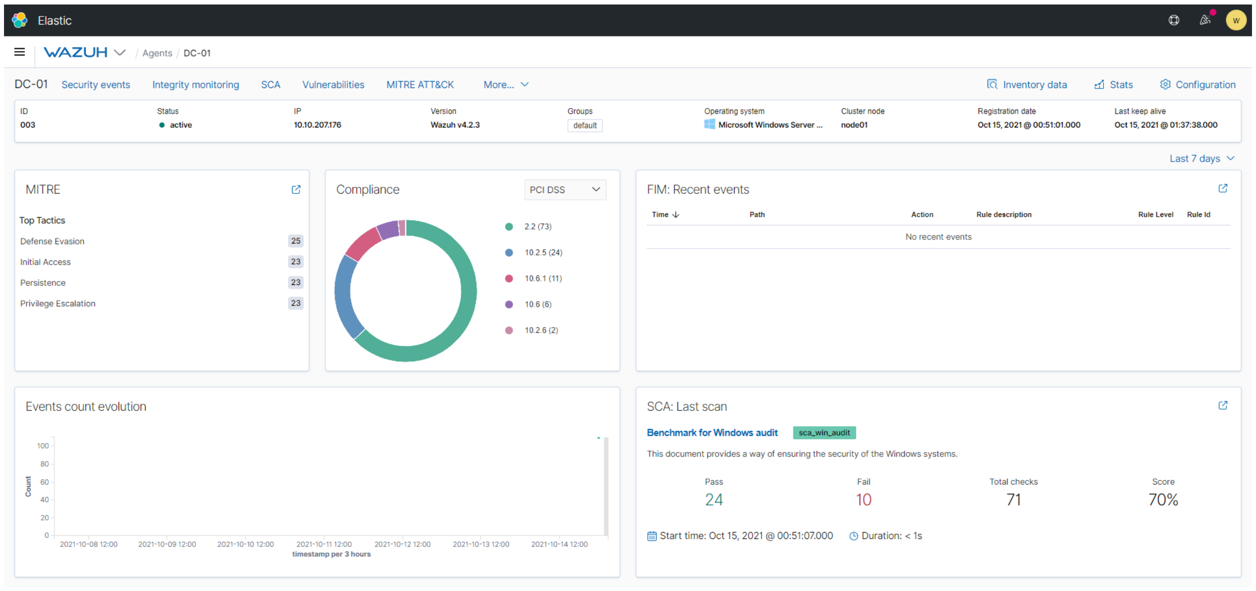
How many "Security Event" alerts have been generated by the agent "AGENT-001"?

Note: You will need to make sure that your time range includes the 11th of March 2022 *196*

Task 5 **Wazuh Policy Auditing**

Wazuh is capable of auditing and monitoring an agent's configuration whilst proactively recording event logs. When the Wazuh agent is installed, an audit is performed where a metric is given using multiple frameworks and legislations such as NIST, MITRE and GDPR.

For example, see how this agent DC-01 scores against MITRE, NIST, and SCA:



ID	Status	IP	Version	Groups	Operating system	Cluster node	Registration date	Last keep alive
003	● active	10.10.207.176	Wazuh v4.2.3	default	Microsoft Windows Server ...	node01	Oct 15, 2021 @ 00:51:01.000	Oct 15, 2021 @ 01:37:38.000

MITRE

Top Tactics

Defense Evasion	25
Initial Access	23
Persistence	23
Privilege Escalation	23

Compliance — PCI DSS

- 2.2 (73)
- 10.2.5 (24)
- 10.6.1 (11)
- 10.6 (6)
- 10.2.6 (2)

FIM: Recent events

Time	Path	Action	Rule description	Rule Level	Rule Id
No recent events					

SCA: Last scan

Benchmark for Windows audit sca_win_audit

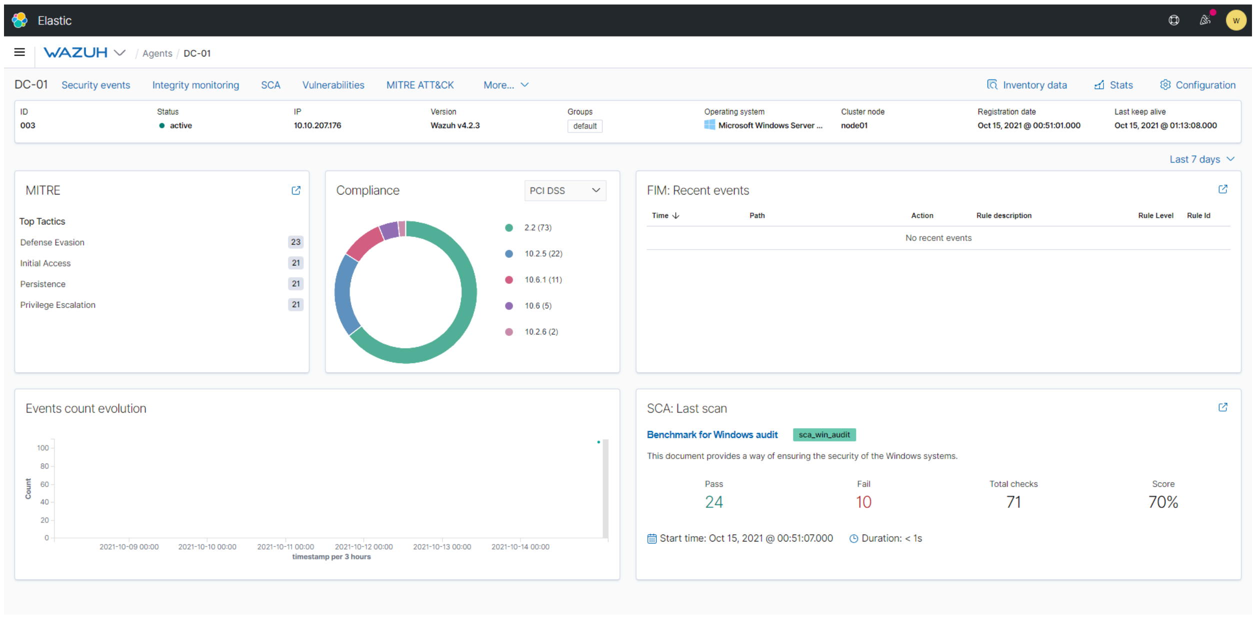
This document provides a way of ensuring the security of the Windows systems.

Pass	Fail	Total checks	Score
24	10	71	70%

Start time: Oct 15, 2021 @ 00:51:07.000 Duration: < 1s

These frameworks are outlined in the Pentesting Fundamentals room if you wish to learn more about them.

Wazuh presents a broad illustration of the logs. We can use the visualizations to break down this data and explore it further. Let's do this with the same agent. For example, see the benchmark for this domain controller running on a windows server:



ID	Status	IP	Version	Groups	Operating system	Cluster node	Registration date	Last keep alive
003	active	10.10.207.176	Wazuh v4.2.3	default	Microsoft Windows Server ...	node01	Oct 15, 2021 @ 00:51:01.000	Oct 15, 2021 @ 01:13:08.000

FIM: Recent events

Time	Path	Action	Rule description	Rule Level	Rule Id
No recent events					

SCA: Last scan

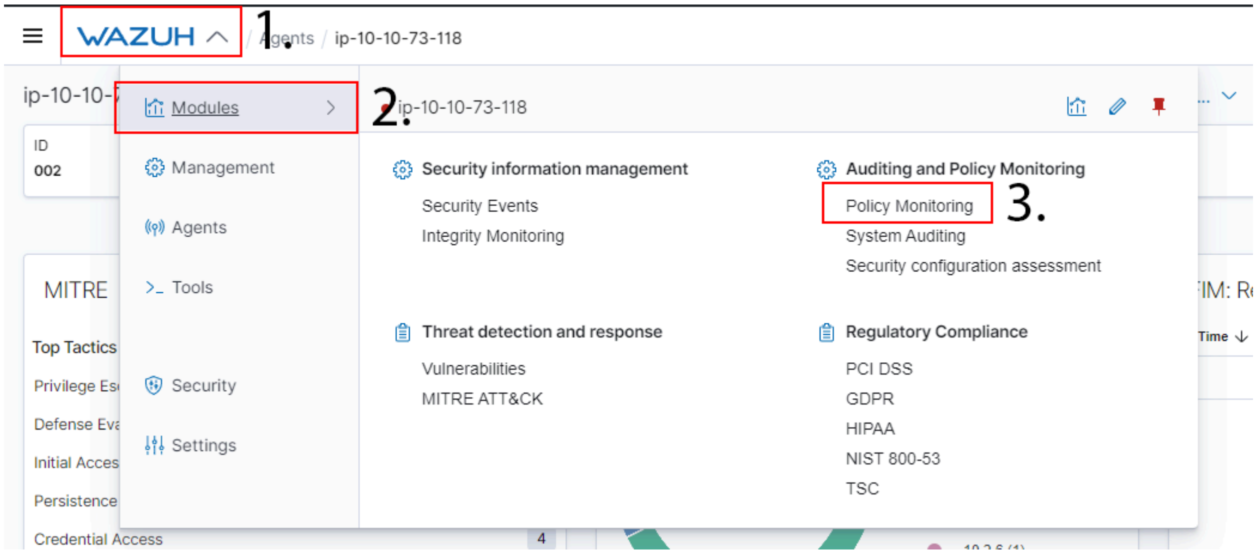
Benchmark for Windows audit sca_win_audit

This document provides a way of ensuring the security of the Windows systems.

Pass	Fail	Total checks	Score
24	10	71	70%

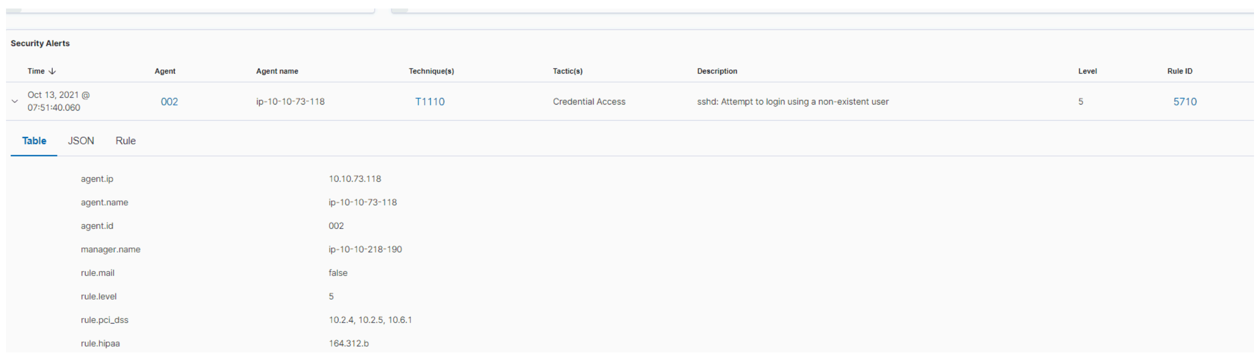
Start time: Oct 15, 2021 @ 00:51:07.000 Duration: < 1s

Navigate to the "Modules" tab by pressing **Wazuh** -> **Modules** and open the "Policy Management" module like so:



Task 6 **Monitoring Logons with Wazuh**

Wazuh's security event monitor is capable to actively record both successful and unsuccessful authentication attempts. The rule with an id of 5710 detects attempted connections that are unsuccessful for the SSH protocol. Let's look at this animated picture below as an example.


Security Alerts

Time	Agent	Agent name	Technique(s)	Tactic(s)	Description	Level	Rule ID
Oct 13, 2021 @ 07:51:40.060	002	ip-10-10-73-118	T1110	Credential Access	sshd: Attempt to login using a non-existent user	5	5710

Table JSON Rule

agent.ip	10.10.73.118
agent.name	ip-10-10-73-118
agent.id	002
manager.name	ip-10-10-218-190
rule.mail	false
rule.level	5
rule.pci_dss	10.2.4, 10.2.5, 10.6.1
rule.hipaa	164.312.b

The alert was created because someone tried to log onto the agent "**ip-10-10-73- 118**" with the user "**cmnatic**" which does not exist. I have summarized this alert into the table below:

Field	**Value**	**Description**
agent.ip	10.10.73.118	This is the IP address of the agent that the alert was triggered on.
agent.name	ip-10-10-73-118	This is the hostname of the agent that the alert was triggered on.
rule.description	sshd: Attempt to login using a non-existent user	This field is a brief description of what the event is alerting to.
rule.mitre.technique	Brute-Force	This field explains the MITRE technique that the alert pertains to.
rule.mitre.id	T1110	This field is the MITRE ID of the alert
rule.id	5710	This field is the ID assigned to the alert by Wazuh's ruleset
location	/var/log/auth.log	This field is the location of the file that the alert was generated from on the agent. In this example, it is the authentication log on the linux agent.

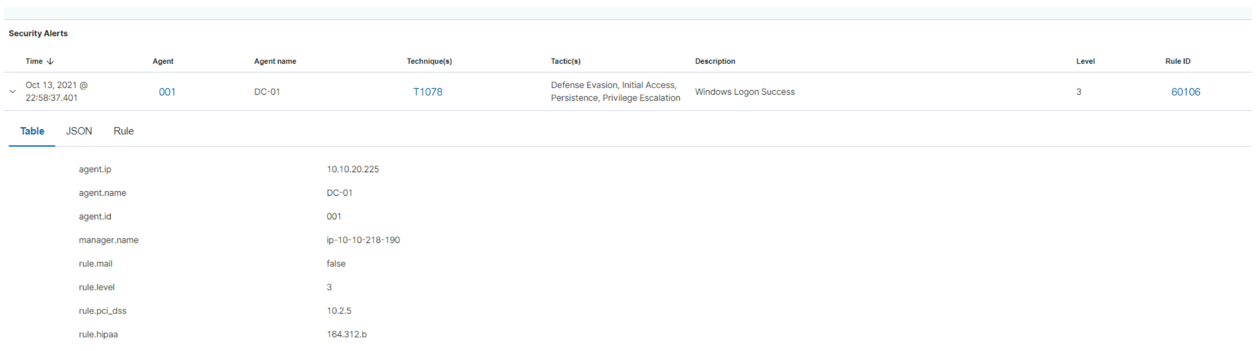
For reference, this alert is stored in a specific file on the Wazuh management server:
. We can use a command such as grep or nano to search through this file on the management server manually.

Viewing the Wazuh logon alert log for a login session (su) on the root account by the ubuntu user

```
ubuntu@wazuh-server:~$ sudo less /var/ossec/logs/alerts/alerts.log
```

```
** Alert 1634284538.566764: -
pam,syslog,authentication_success,pci_dss_10.2.5,gpg13_7.8,gpg13_7.9,gdpr_IV_32.2,hipaa_
164.312.b,ni>
2021 Oct 15 07:55:38 ip-10-10-218-190->/var/log/auth.log
Rule: 5501 (level 3) -> 'PAM: Login session opened.'
User: root
Oct 15 07:55:37 ip-10-10-218-190 sudo: pam_unix(sudo:session): session opened for user root
by ubuntu(uid=0)
uid: 0
```

Looking at the animated gif below, we can see how Wazuh has created an alert for successful login to a Window's server running the Wazuh agent. Because this attempt was successful, the severity of the alert is considered less than that of an unsuccessful login. This can, of course, be tailored to your environment. For example, if a user infrequently used is logged on, you can configure Wazuh to list this alert with higher severity.



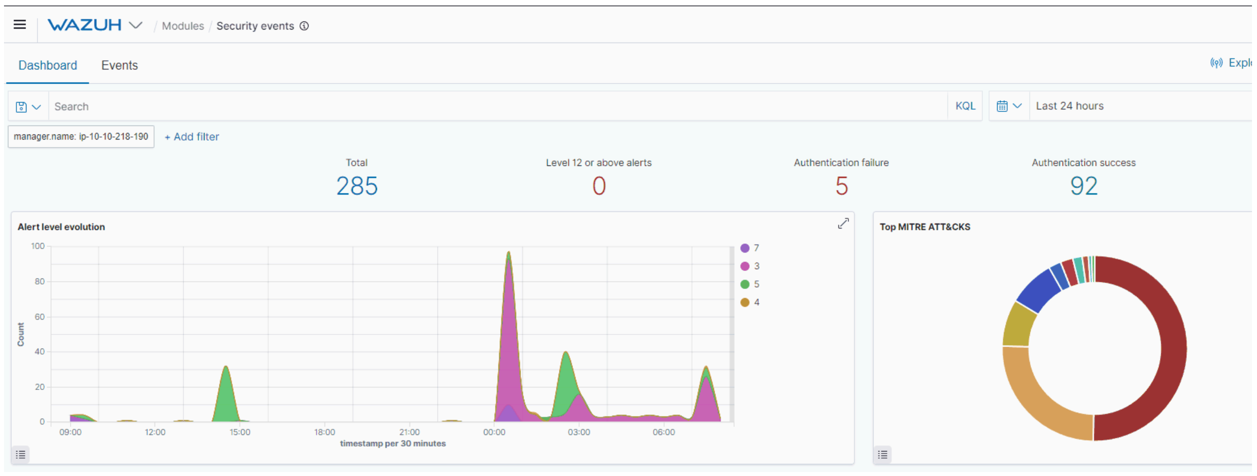
Security Alerts

Time	Agent	Agent name	Technique(s)	Tactic(s)	Description	Level	Rule ID
Oct 13, 2021 @ 22:58:37.401	001	DC-01	T1078	Defense Evasion, Initial Access, Persistence, Privilege Escalation	Windows Logon Success	3	60106

Table JSON Rule

agent.ip	10.10.20.225
agent.name	DC-01
agent.id	001
manager.name	ip-10-10-218-190
rule.mail	false
rule.level	3
rule.pci_dss	10.2.5
rule.hipaa	164.312.b

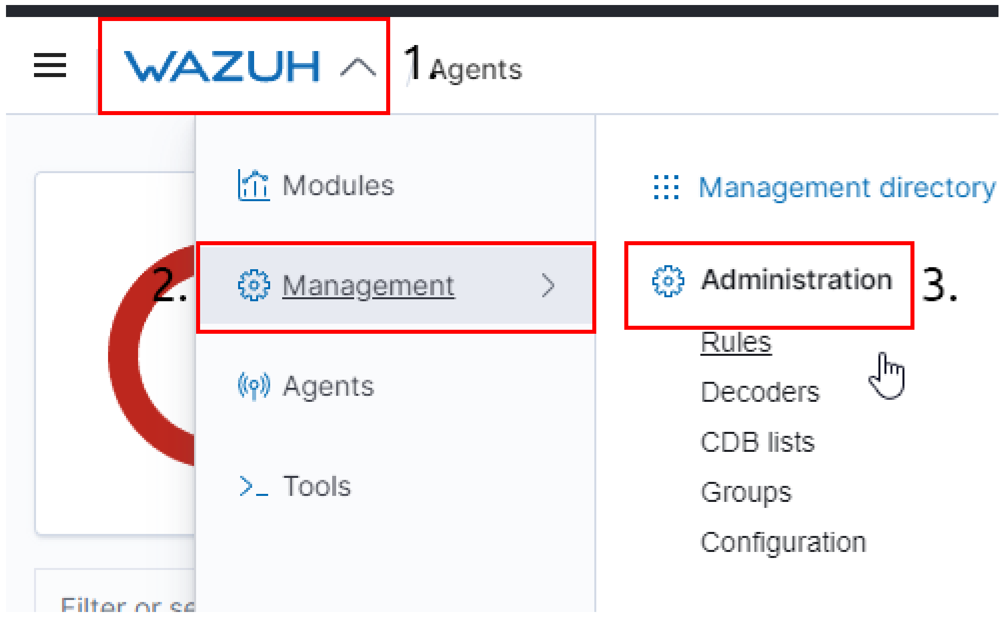
The animated gif below shows the number of Windows agent events/alerts triggered to show how many times a user has logged on. . In this case, it narrows down the total logon events of **285** to **79**.



manager.name: ip-10-10-218-190

Total	Level 12 or above alerts	Authentication failure	Authentication success
285	0	5	92

Navigate to the "Management" tab by pressing Wazuh -> **Management** and open the "Rules" module like so:



Task 7 **Collecting Windows Logs with Wazuh**

All sorts of actions and events are captured and recorded on a Windows operating system. This includes authentication attempts, networking connections, files that were accessed, and the behaviours of applications and services. This information is stored in the Windows event log using a tool called Sysmon.

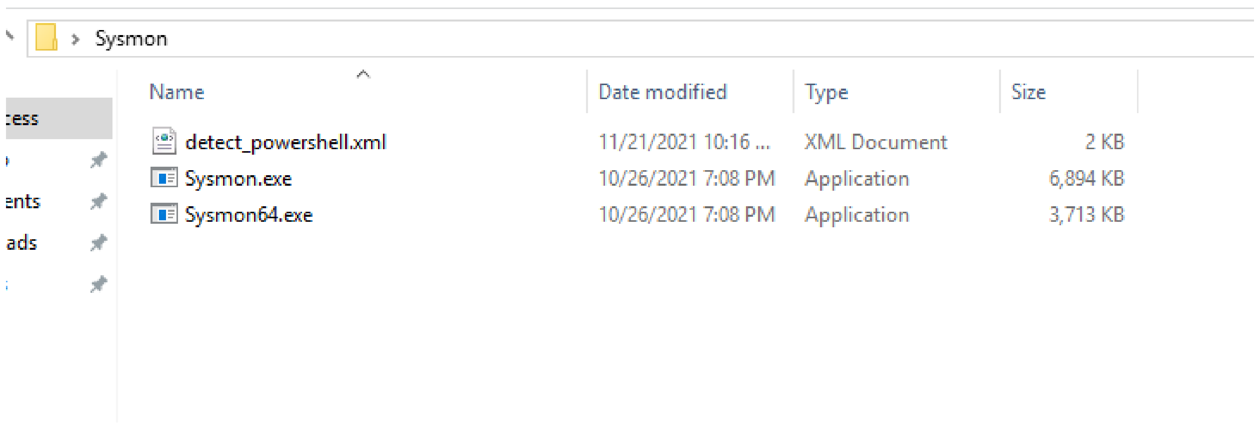
We can use the Wazuh agent to aggregate these events recorded by *Sysmon* for processing to the Wazuh manager. Now, we will need to configure both the Wazuh agent and the Sysmon application. Sysmon uses rules that are made in XML formatting to be triggered. For example, in the XML snippet below, we are telling Sysmon to monitor for the event of the powershell.exe process starting.

A Sysmon configuration file for monitoring the Powershell process

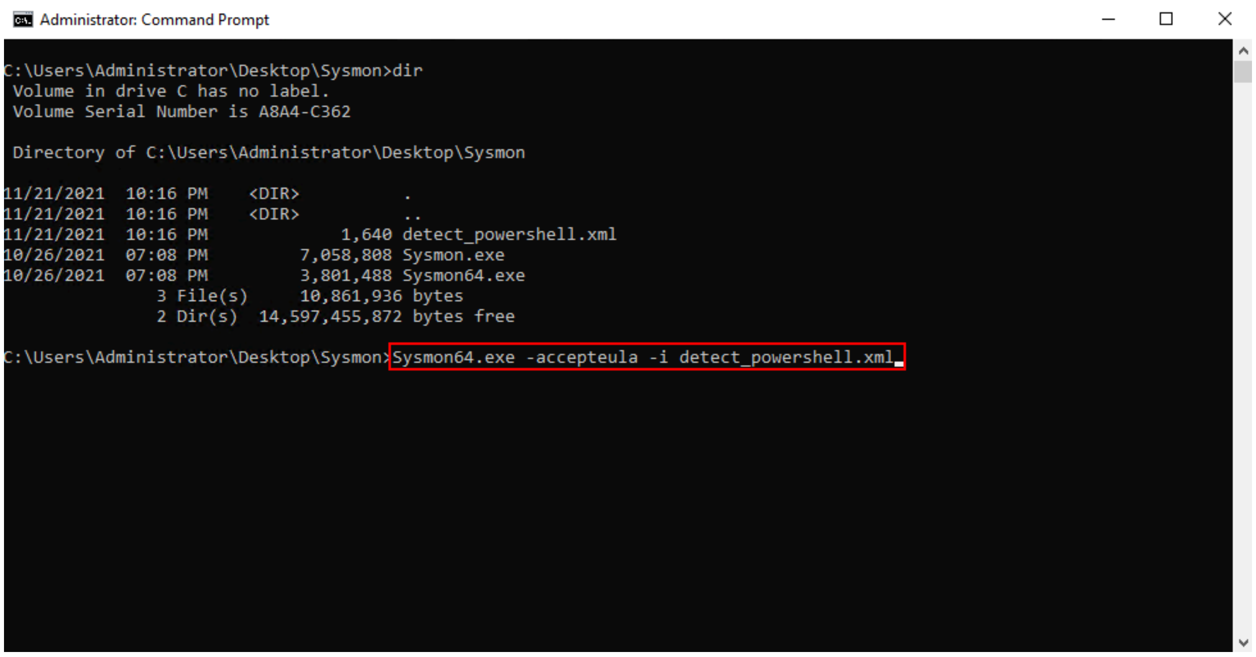
```
    Sysmon schemaversion="3.30"
      HashAlgorithms md5 /HashAlgorithms
  EventFiltering
  !--SYSMON EVENT ID 1 : PROCESS CREATION--
  ProcessCreate onmatch="include"
  Image condition="contains" powershell.exe /Image
  /ProcessCreate
  !--SYSMON EVENT ID 2 : FILE CREATION TIME RETROACTIVELY CHANGED IN THE
FILESYSTEM--
```

```
 FileCreateTime onmatch="include"  /FileCreateTime
 !--SYSMON EVENT ID 3 : NETWORK CONNECTION INITIATED--
 NetworkConnect onmatch="include"  /NetworkConnect
 !--SYSMON EVENT ID 4 : RESERVED FOR SYSMON STATUS MESSAGES, THIS LINE IS
INCLUDED FOR DOCUMENTATION PURPOSES ONLY--
 !--SYSMON EVENT ID 5 : PROCESS ENDED--
 ProcessTerminate onmatch="include"  /ProcessTerminate
 !--SYSMON EVENT ID 6 : DRIVER LOADED INTO KERNEL--
 DriverLoad onmatch="include"  /DriverLoad
 !--SYSMON EVENT ID 7 : DLL (IMAGE) LOADED BY PROCESS--
 ImageLoad onmatch="include"  /ImageLoad
 !--SYSMON EVENT ID 8 : REMOTE THREAD CREATED--
 CreateRemoteThread onmatch="include"  /CreateRemoteThread
 !--SYSMON EVENT ID 9 : RAW DISK ACCESS--
 RawAccessRead onmatch="include"  /RawAccessRead
 !--SYSMON EVENT ID 10 : INTER-PROCESS ACCESS--
 ProcessAccess onmatch="include"  /ProcessAccess
 !--SYSMON EVENT ID 11 : FILE CREATED--
 FileCreate onmatch="include"  /FileCreate
 !--SYSMON EVENT ID 12 & 13 & 14 : REGISTRY MODIFICATION--
 RegistryEvent onmatch="include"  /RegistryEvent
 !--SYSMON EVENT ID 15 : ALTERNATE DATA STREAM CREATED--
 FileCreateStreamHash onmatch="include"  /FileCreateStreamHash
 PipeEvent onmatch="include"  /PipeEvent
 /EventFiltering
/Sysmon
```

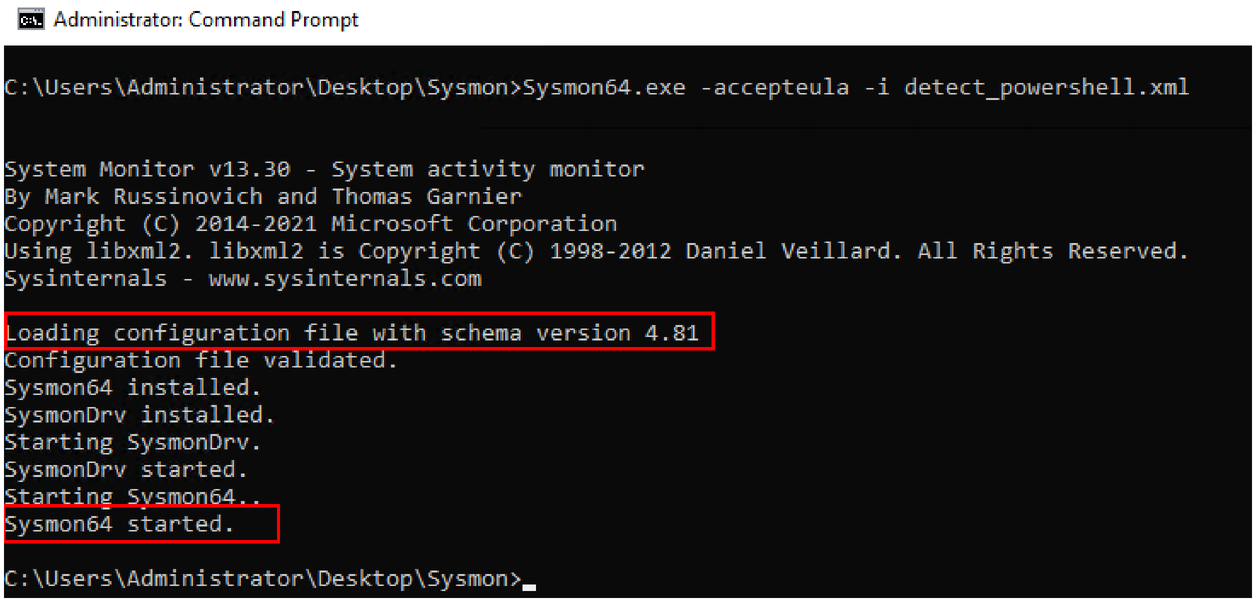
To instruct Sysmon to do, we need to execute the Sysmon application and provide the aforementioned configuration file like so:



Name	Date modified	Type	Size
detect_powershell.xml	11/21/2021 10:16 ...	XML Document	2 KB
Sysmon.exe	10/26/2021 7:08 PM	Application	6,894 KB
Sysmon64.exe	10/26/2021 7:08 PM	Application	3,713 KB

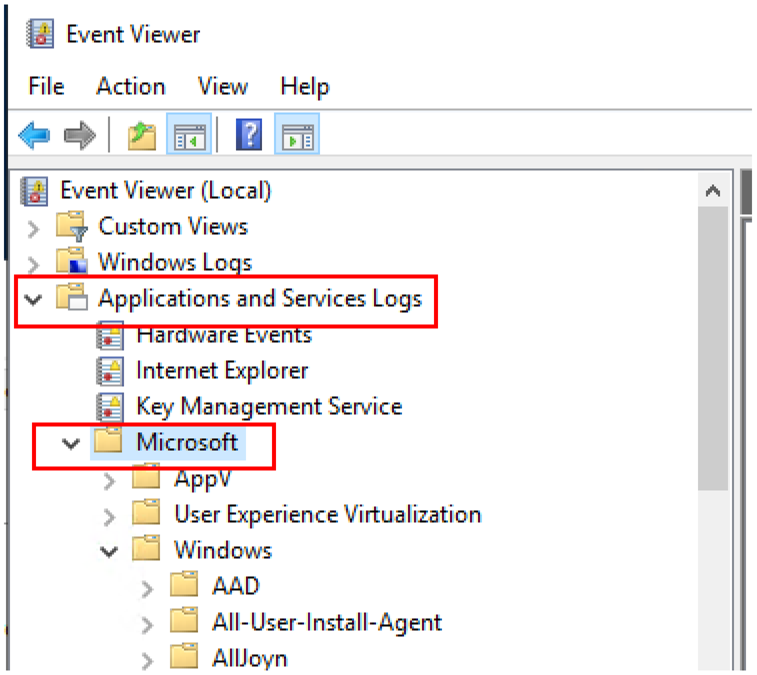


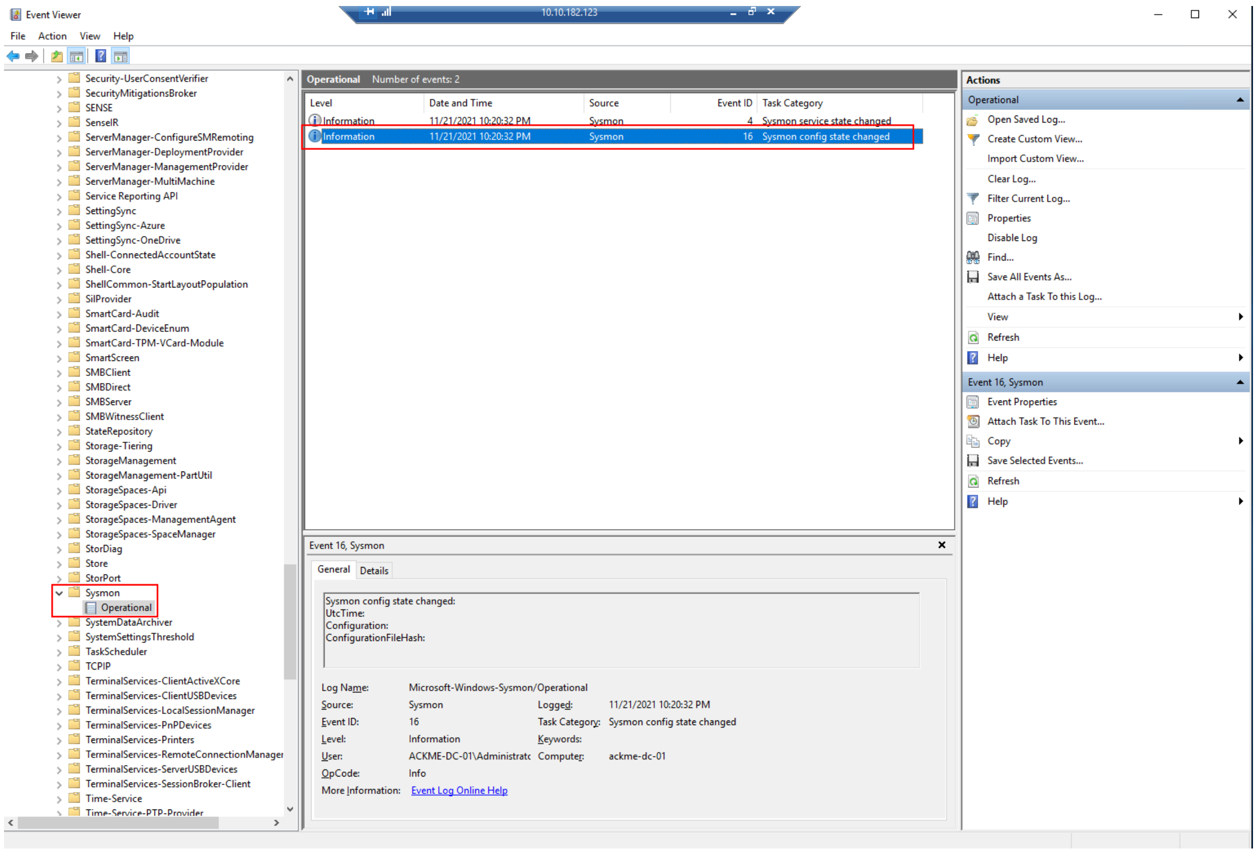
```
C:\Users\Administrator\Desktop\Sysmon>dir
 Volume in drive C has no label.
 Volume Serial Number is A8A4-C362

 Directory of C:\Users\Administrator\Desktop\Sysmon

11/21/2021  10:16 PM    <DIR>          .
11/21/2021  10:16 PM    <DIR>          ..
11/21/2021  10:16 PM             1,640 detect_powershell.xml
10/26/2021  07:08 PM         7,058,808 Sysmon.exe
10/26/2021  07:08 PM         3,801,488 Sysmon64.exe
               3 File(s)     10,861,936 bytes
               2 Dir(s)  14,597,455,872 bytes free

C:\Users\Administrator\Desktop\Sysmon>Sysmon64.exe -accepteula -i detect_powershell.xml
```



```
C:\Users\Administrator\Desktop\Sysmon>Sysmon64.exe -accepteula -i detect_powershell.xml


System Monitor v13.30 - System activity monitor
By Mark Russinovich and Thomas Garnier
Copyright (C) 2014-2021 Microsoft Corporation
Using libxml2. libxml2 is Copyright (C) 1998-2012 Daniel Veillard. All Rights Reserved.
Sysinternals - www.sysinternals.com

Loading configuration file with schema version 4.81
Configuration file validated.
Sysmon64 installed.
SysmonDrv installed.
Starting SysmonDrv.
SysmonDrv started.
Starting Sysmon64..
Sysmon64 started.

C:\Users\Administrator\Desktop\Sysmon>
```

We can verify that Sysmon has accepted our configuration file by navigating to the Event Viewer and searching for the “**Sysmon**” module like so:




Event Viewer
10.10.182.123
File
Action
View
Help
Security-UserConsentVerifier
SecurityMitigationsBroker
SENSE
SenseIR
ServerManager-ConfigureSMRemoting
ServerManager-DeploymentProvider
ServerManager-ManagementProvider
ServerManager-MultiMachine
Service Reporting API
SettingSync
SettingSync-Azure
SettingSync-OneDrive
Shell-ConnectedAccountState
Shell-Core
ShellCommon-StartLayoutPopulation
SilProvider
SmartCard-Audit
SmartCard-DeviceEnum
SmartCard-TPM-VCard-Module
SmartScreen
SMBClient
SMBDirect
SMBServer
SMBWitnessClient
StateRepository
Storage-Tiering
StorageManagement
StorageManagement-PartUtil
StorageSpaces-Api
StorageSpaces-Driver
StorageSpaces-ManagementAgent
StorageSpaces-SpaceManager
StorDiag
Store
StorPort
Sysmon
Operational
SystemDataArchiver
SystemSettingsThreshold
TaskScheduler
TCPIP
TerminalServices-ClientActiveXCore
TerminalServices-ClientUSBDevices
TerminalServices-LocalSessionManager
TerminalServices-PnPDevices
TerminalServices-Printers
TerminalServices-RemoteConnectionManager
TerminalServices-ServerUSBDevices
TerminalServices-SessionBroker-Client
Time-Service
Time-Service-PTP-Provider
Actions
Operational
Open Saved Log...
Create Custom View...
Import Custom View...
Clear Log...
Filter Current Log...
Properties
Disable Log
Find...
Save All Events As...
Attach a Task To this Log...
View
Refresh
Help
Event 16, Sysmon
Event Properties
Attach Task To This Event...
Copy
Save Selected Events...
Refresh
Help

Operational Number of events: 2

Level	Date and Time	Source	Event ID	Task Category
Information	11/21/2021 10:20:32 PM	Sysmon	4	Sysmon service state changed
Information	11/21/2021 10:20:32 PM	Sysmon	16	Sysmon config state changed

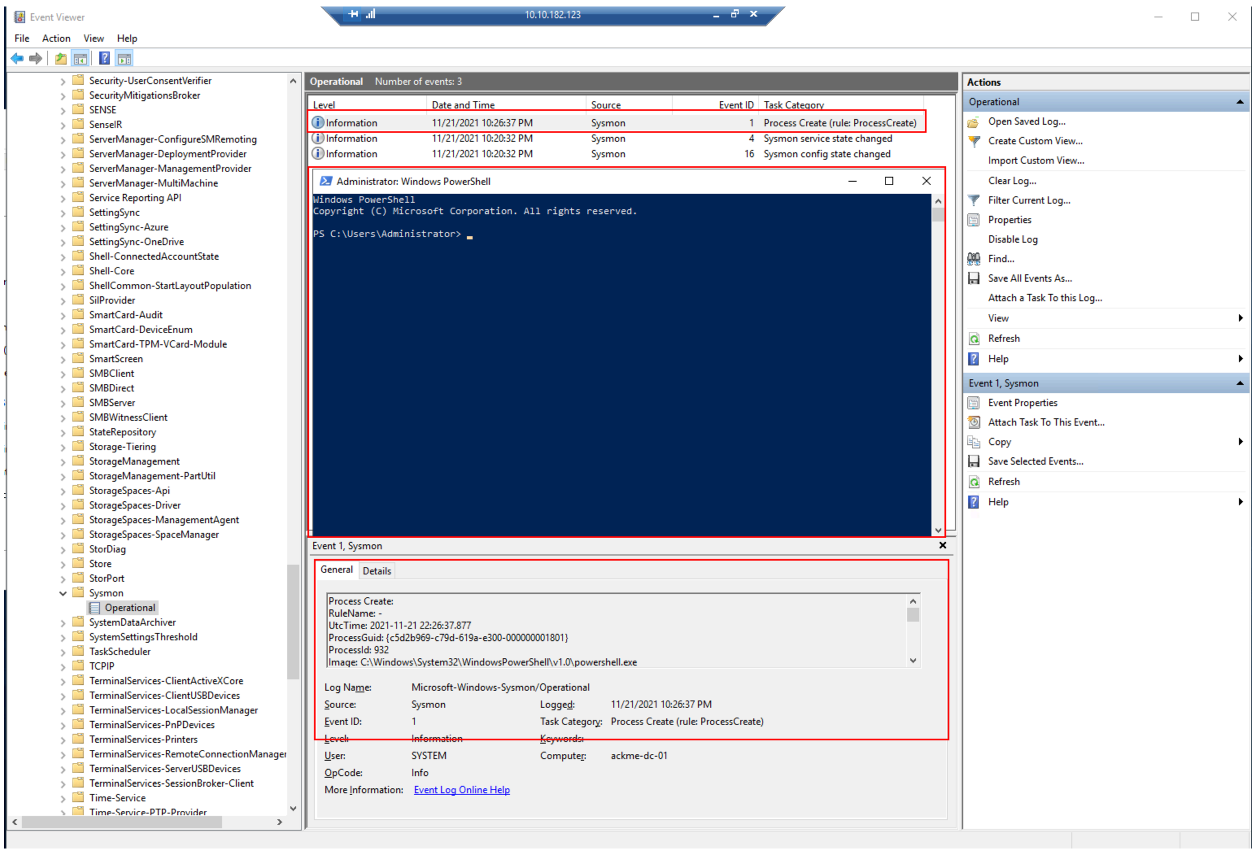
Event 16, Sysmon

General | Details

```
Sysmon config state changed:
UtcTime:
Configuration:
ConfigurationFileHash:
```

Log Name:	Microsoft-Windows-Sysmon/Operational		
Source:	Sysmon	Logged:	11/21/2021 10:20:32 PM
Event ID:	16	Task Category:	Sysmon config state changed
Level:	Information	Keywords:	
User:	ACKME-DC-01\Administratc	Computer:	ackme-dc-01
OpCode:	Info		
More Information:	Event Log Online Help		

Let’s launch a powershell prompt on the Windows Server and return to our Event Viewer. We can now see a record of this powershell prompt being opened, kept within the Event Viewer.


Event Viewer
10.10.182.123
File Action View Help
Security-UserConsentVerifier
SecurityMitigationsBroker
SENSE
SenseIR
ServerManager-ConfigureSMRemoting
ServerManager-DeploymentProvider
ServerManager-ManagementProvider
ServerManager-MultiMachine
Service Reporting API
SettingSync
SettingSync-Azure
SettingSync-OneDrive
Shell-ConnectedAccountState
Shell-Core
ShellCommon-StartLayoutPopulation
SilProvider
SmartCard-Audit
SmartCard-DeviceEnum
SmartCard-TPM-VCard-Module
SmartScreen
SMBClient
SMBDirect
SMBServer
SMBWitnessClient
StateRepository
Storage-Tiering
StorageManagement
StorageManagement-PartUtil
StorageSpaces-Api
StorageSpaces-Driver
StorageSpaces-ManagementAgent
StorageSpaces-SpaceManager
StorDiag
Store
StorPort
Sysmon
Operational
SystemDataArchiver
SystemSettingsThreshold
TaskScheduler
TCPIP
TerminalServices-ClientActiveXCore
TerminalServices-ClientUSBDevices
TerminalServices-LocalSessionManager
TerminalServices-PnPDevices
TerminalServices-Printers
TerminalServices-RemoteConnectionManager
TerminalServices-ServerUSBDevices
TerminalServices-SessionBroker-Client
Time-Service
Time-Service-PTP-Provider
Actions
Operational
Open Saved Log...
Create Custom View...
Import Custom View...
Clear Log...
Filter Current Log...
Properties
Disable Log
Find...
Save All Events As...
Attach a Task To this Log...
View
Refresh
Help
Event 1, Sysmon
Event Properties
Attach Task To This Event...
Copy
Save Selected Events...
Refresh
Help
Administrator: Windows PowerShell
General
Details

Operational   Number of events: 3

Level	Date and Time	Source	Event ID	Task Category
Information	11/21/2021 10:26:37 PM	Sysmon	1	Process Create (rule: ProcessCreate)
Information	11/21/2021 10:20:32 PM	Sysmon	4	Sysmon service state changed
Information	11/21/2021 10:20:32 PM	Sysmon	16	Sysmon config state changed

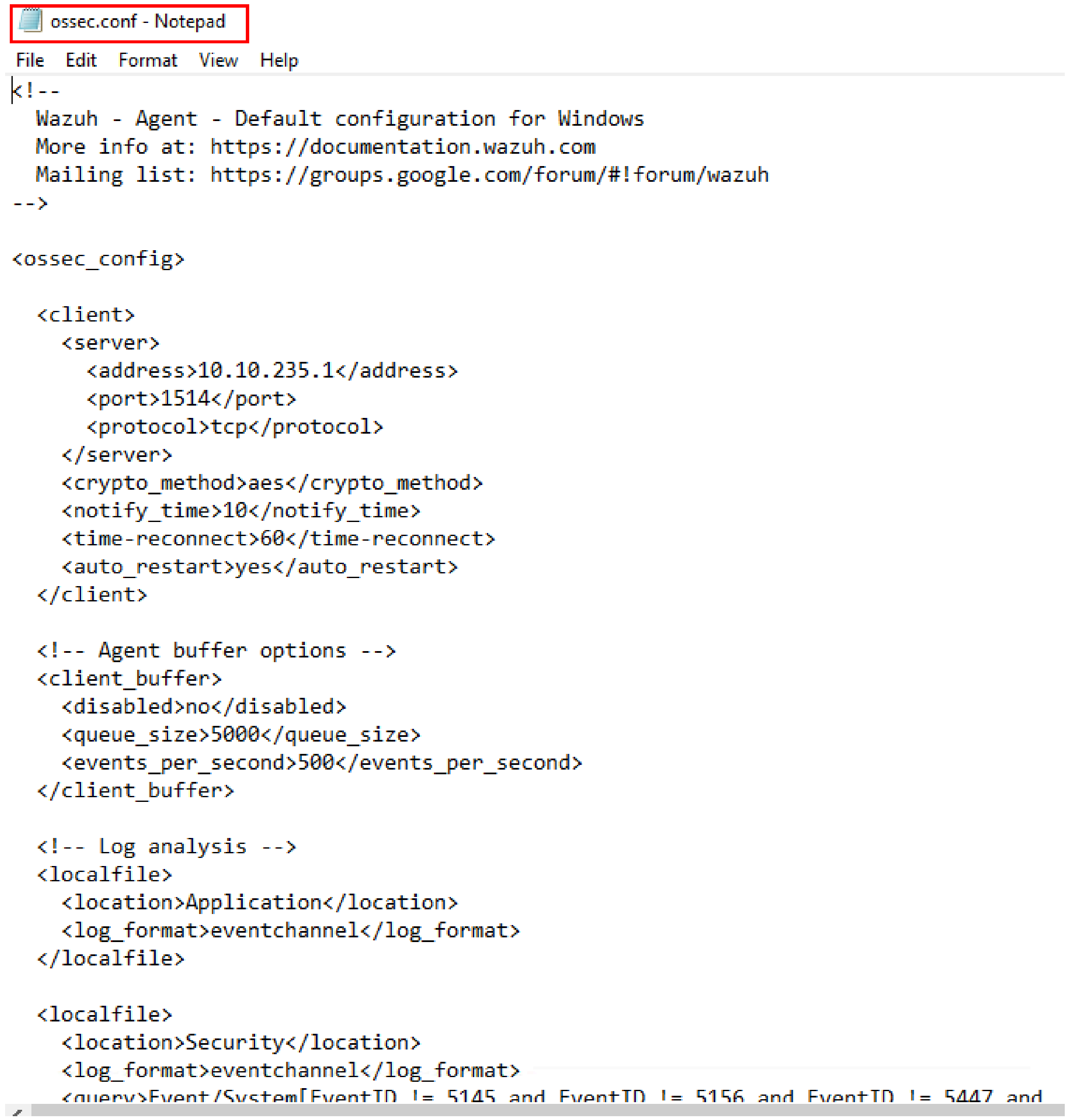
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> _
```

Event 1, Sysmon

```
Process Create:
RuleName: -
UtcTime: 2021-11-21 22:26:37.877
ProcessGuid: {c5d2b969-c79d-619a-e300-000000001801}
ProcessId: 932
Image: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
```

Log Name:	Microsoft-Windows-Sysmon/Operational		
Source:	Sysmon	Logged:	11/21/2021 10:26:37 PM
Event ID:	1	Task Category:	Process Create (rule: ProcessCreate)
Level:	Information	Keywords:	
User:	SYSTEM	Computer:	ackme-dc-01
OpCode:	Info		
More Information:	Event Log Online Help		

Now we will need to configure the Wazuh agent on this Window Server to instruct it to send these events to the Wazuh management server. To do so, we need to open the Wazuh agent file located at:



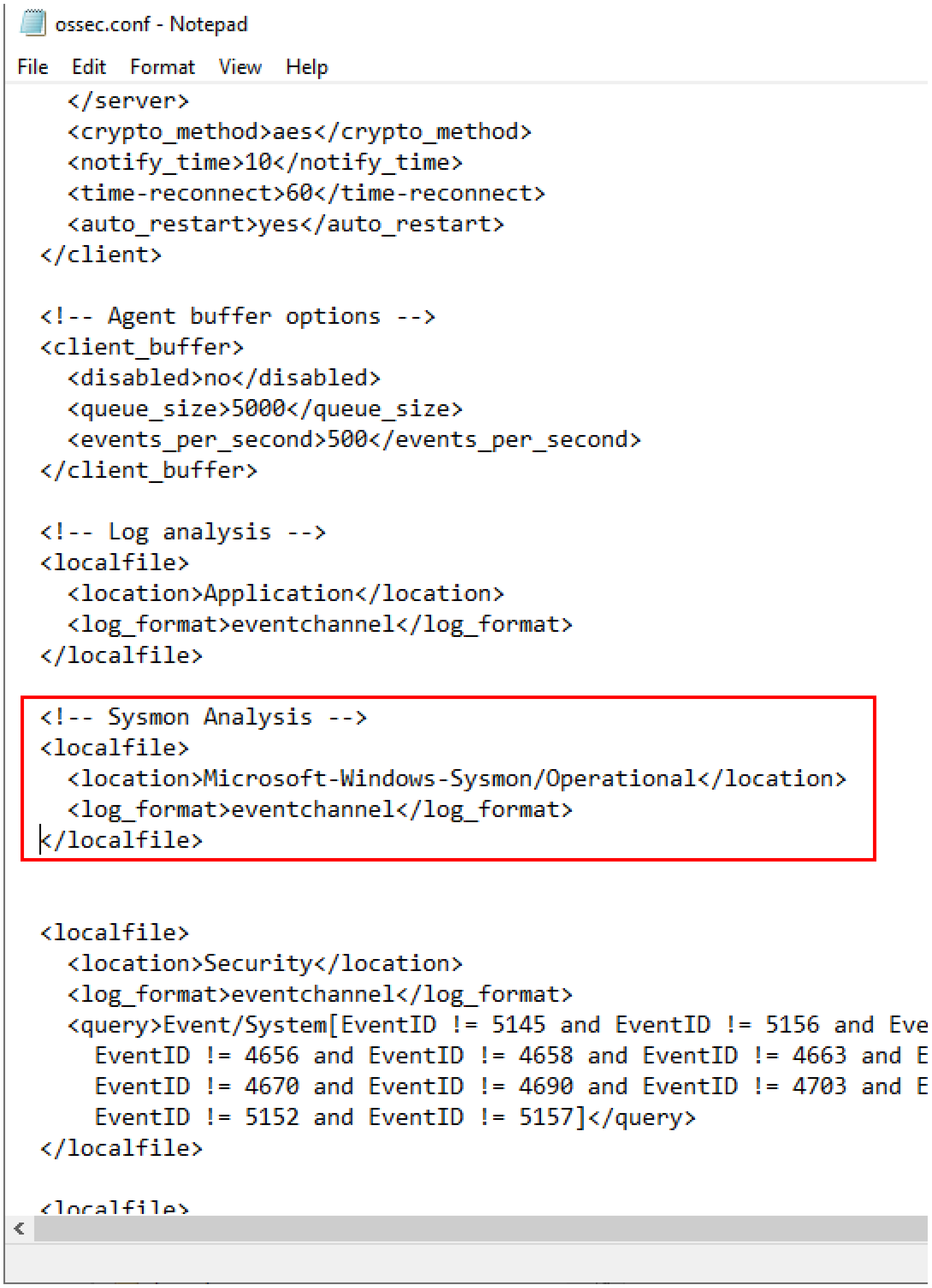
```
<!--
  Wazuh - Agent - Default configuration for Windows
  More info at: https://documentation.wazuh.com
  Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>

  <client>
    <server>
      <address>10.10.235.1</address>
      <port>1514</port>
      <protocol>tcp</protocol>
    </server>
    <crypto_method>aes</crypto_method>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
  </client>

  <!-- Agent buffer options -->
  <client_buffer>
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
  </client_buffer>

  <!-- Log analysis -->
  <localfile>
    <location>Application</location>
    <log_format>eventchannel</log_format>
  </localfile>

  <localfile>
    <location>Security</location>
    <log_format>eventchannel</log_format>
    <query>Event/System[EventID != 5145 and EventID != 5156 and EventID != 5447 and
```

To include the following snippet:

Configuring the Wazuh Agent's configuration

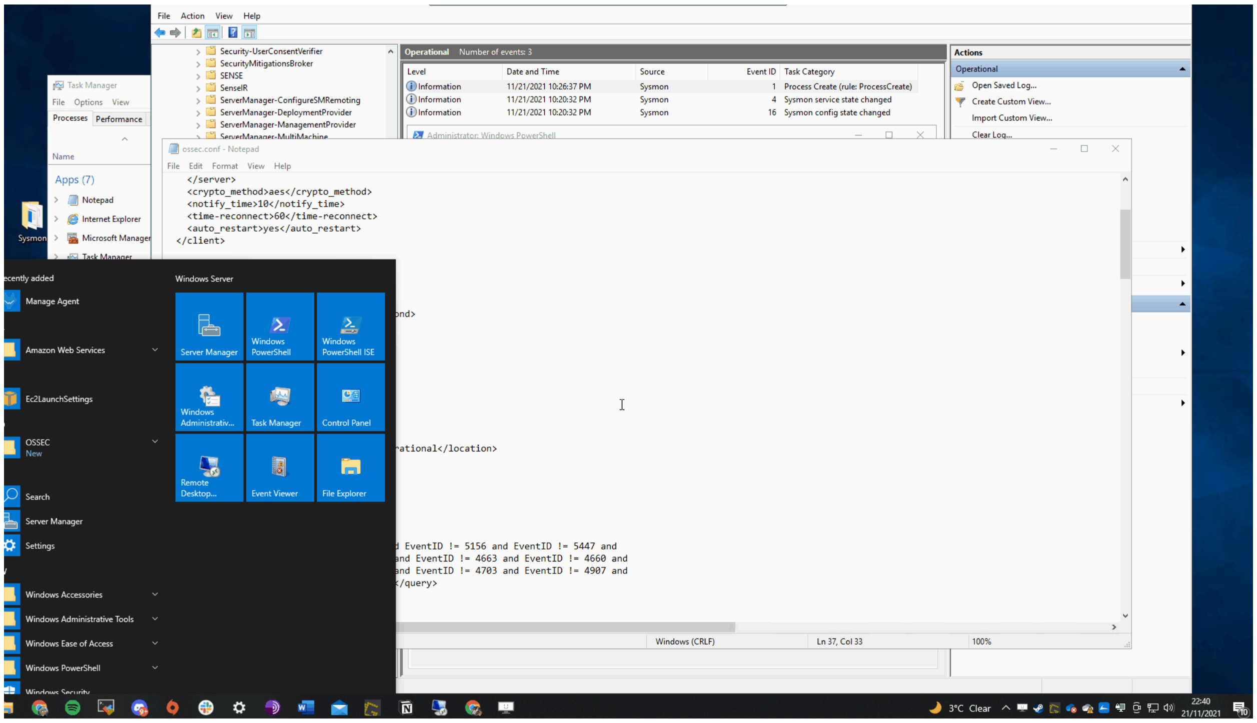
```
        <localfile>
<location>Microsoft-Windows-Sysmon/Operational</location>
<log_format>eventchannel</log_format>
</localfile>
```

Looking like so:



```
    </server>
    <crypto_method>aes</crypto_method>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
  </client>

  <!-- Agent buffer options -->
  <client_buffer>
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
  </client_buffer>

  <!-- Log analysis -->
  <localfile>
    <location>Application</location>
    <log_format>eventchannel</log_format>
  </localfile>

  <!-- Sysmon Analysis -->
  <localfile>
    <location>Microsoft-Windows-Sysmon/Operational</location>
    <log_format>eventchannel</log_format>
  </localfile>


  <localfile>
    <location>Security</location>
    <log_format>eventchannel</log_format>
    <query>Event/System[EventID != 5145 and EventID != 5156 and Eve
      EventID != 4656 and EventID != 4658 and EventID != 4663 and E
      EventID != 4670 and EventID != 4690 and EventID != 4703 and E
      EventID != 5152 and EventID != 5157]</query>
  </localfile>

  <localfile>
```

Now, we will need to restart the Wazuh agent. In this instance, I am restarting the operating system just to be sure that these changes have taken place.



Operational Number of events: 3

Level	Date and Time	Source	Event ID	Task Category
Information	11/21/2021 10:26:37 PM	Sysmon	1	Process Create (rule: ProcessCreate)
Information	11/21/2021 10:20:32 PM	Sysmon	4	Sysmon service state changed
Information	11/21/2021 10:20:32 PM	Sysmon	16	Sysmon config state changed

```
  </server>
  <crypto_method>aes</crypto_method>
  <notify_time>10</notify_time>
  <time-reconnect>60</time-reconnect>
  <auto_restart>yes</auto_restart>
</client>
```

```
ond>
rational</location>
d EventID != 5156 and EventID != 5447 and
and EventID != 4663 and EventID != 4660 and
and EventID != 4703 and EventID != 4907 and
</query>
```

Once this is done, we need to tell the Wazuh Management server to add Sysmon as a rule to visualize these events. This can be done by adding an XML file to the local rules located in

Configuring the Wazuh Server to ingress Ssymon events

```
        <group name="sysmon,">
 <rule id="255000" level="12">
 <if_group>sysmon_event1</if_group>
 <field name="sysmon.image">\\powershell.exe||\\.ps1||\\.ps2</field>
 <description>Sysmon - Event 1: Bad exe: $(sysmon.image)</description>
 <group>sysmon_event1,powershell_execution,</group>
 </rule>
</group>
```

You will need to restart the Wazuh Management server for this to apply. Once done, we can refer back to our Wazuh Management server and notice that data has been retrieved from an agent.

Answer the questions below

What is the name of the tool that we can use to monitor system events?

Sysmon

What standard application on Windows do these system events get recorded to?

Event Viewer

Task 8 Collecting Linux Logs with Wazuh

Capturing logs from a Linux agent is a simple process similar to capturing events from a Windows agent. We will be using Wazuh's log collector service to create an entry on the agent to instruct what logs should be sent to the Wazuh management server.

For example, in this task, we will be monitoring the logs of an Apache2 web server. To begin, let's configure the log collector service on a Linux server running the Wazuh agent.

Wazuh comes with many rules that enable Wazuh to analyze log files and can be found in . Some common applications include:

- Docker
- FTP
- WordPress
- SQL Server
- MongoDB
- Firewalld
- And many, many more (approximately 900).

However, you can always make your own rules. In this task, Wazuh will digest *Apache2* logs using the ruleset.

This ruleset can analyze apache2 logs for warnings and error messages like so: We will need to insert this into the Wazuh's agent that is sending logs to the Wazuh management servers configuration file located in :

Apache2 Log Analysis

```
      <!-- Apache2 Log Analysis -->
 <localfile>
  <location>/var/log/example.log</location>
  <log_format>syslog</log_format>
 </localfile>
```

We will now need to restart the Linux agent running the Apache2 service.

Answer the questions below

What is the full file path to the rules located on a Wazuh management server?

/var/ossec/ruleset/rules

__Task 9__ Auditing Commands on Linux with Wazuh

Wazuh utilises the package that can be installed on Wazuh agents running on Debian/Ubuntu and CentOS operating systems. In this task, we will be using on a Ubuntu system. monitors the system for certain actions and events and will write this to a log file.

We can then use the log collector module on a Wazuh agent to read this log file and send it to the Wazuh management server for processing.

First, we will need to install the package and an plugin. This may already be installed on your system; however, let's install it to make sure. Let's run the command and enable this service to run currently as well as on boot. &

We will need to configure to create a rule for the commands and events that we wish for it to monitor. In this task, we will be telling to monitor for any commands executed as root.

You can extend this to monitor commands such as , , or *catting* files such as */etc/passwd*, which are all hallmarks of a breach.

rules are located in the following directory: . We will be adding our rules manually.

For this task, we will need to open this *audit.rules* file and append our rule ourselves. First, let's edit the file using and appending

Monitoring commands executed as root

## First rule - delete all

-D

```
## Increase the buffers to survive stress events.

## Make this bigger for busy systems

-b 8192


## This determine how long to wait in burst of events

--backlog_wait_time 0


## Set failure mode to syslog

-f 1


-a exit,always -F arch=b64 -F euid=0 -S execve -k  audit-wazuh-c
```

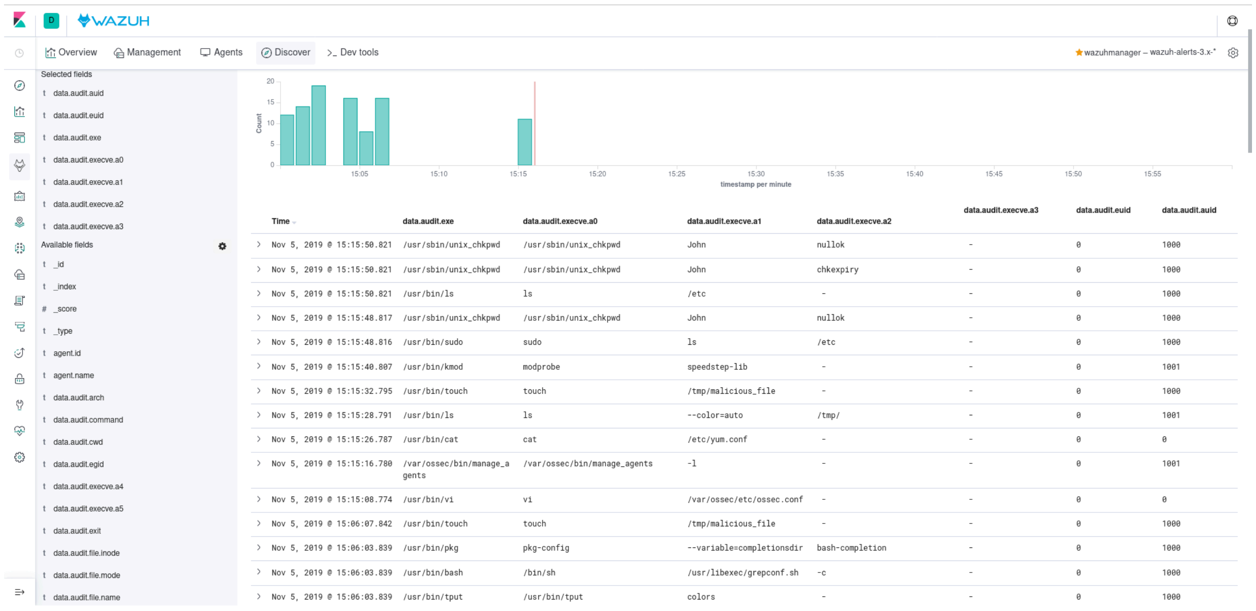
We will now need to inform audits of this new rule, so let's run this command
to now read the new *audit.rules* file that we appended to in the previous task.

Now, let’s configure the system that is running a Wazuh agent that we wish to monitor these events on. We’ll be monitoring a Linux host in this case, so like in our previous tasks, we will need to configure the Wazuh agent to detect this new log file that is generated by like so and add the log like so:

Configuring the Wazuh agent to add the auditd log as a log file to send to the Wazuh management server

```
<localfile>

   <location>/var/log/audit/audit.log</location>

   <log_format>audit</log_format>
```

</localfile>



Time	data.audit.exe	data.audit.execve.a0	data.audit.execve.a1	data.audit.execve.a2	data.audit.execve.a3	data.audit.euid	data.audit.auid
Nov 5, 2019 @ 15:15:50.821	/usr/sbin/unix_chkpwd	/usr/sbin/unix_chkpwd	John	nullok	-	0	1000
Nov 5, 2019 @ 15:15:50.821	/usr/sbin/unix_chkpwd	/usr/sbin/unix_chkpwd	John	chkexpiry	-	0	1000
Nov 5, 2019 @ 15:15:50.821	/usr/bin/ls	ls	/etc	-	-	0	1000
Nov 5, 2019 @ 15:15:48.817	/usr/sbin/unix_chkpwd	/usr/sbin/unix_chkpwd	John	nullok	-	0	1000
Nov 5, 2019 @ 15:15:48.816	/usr/bin/sudo	sudo	ls	/etc	-	0	1000
Nov 5, 2019 @ 15:15:40.807	/usr/bin/kmod	modprobe	speedstep-lib	-	-	0	1001
Nov 5, 2019 @ 15:15:32.795	/usr/bin/touch	touch	/tmp/malicious_file	-	-	0	1000
Nov 5, 2019 @ 15:15:28.791	/usr/bin/ls	ls	--color=auto	/tmp/	-	0	1001
Nov 5, 2019 @ 15:15:26.787	/usr/bin/cat	cat	/etc/yum.conf	-	-	0	0
Nov 5, 2019 @ 15:15:16.780	/var/ossec/bin/manage_agents	/var/ossec/bin/manage_agents	-l	-	-	0	1001
Nov 5, 2019 @ 15:15:08.774	/usr/bin/vi	vi	/var/ossec/etc/ossec.conf	-	-	0	0
Nov 5, 2019 @ 15:06:07.842	/usr/bin/touch	touch	/tmp/malicious_file	-	-	0	1000
Nov 5, 2019 @ 15:06:03.839	/usr/bin/pkg	pkg-config	--variable=completionsdir	bash-completion	-	0	1000
Nov 5, 2019 @ 15:06:03.839	/usr/bin/bash	/bin/sh	/usr/libexec/grepconf.sh	-c	-	0	1000
Nov 5, 2019 @ 15:06:03.839	/usr/bin/tput	/usr/bin/tput	colors	-	-	0	1000

Answer the questions below

What application do we use on Linux to monitor events such as command execution?

auditd

What is the full path & filename for where the aforementioned application stores rules?

/etc/audit/rules.d/audit.rules

Task 10 Wazuh API

Wazuh utilises the package that can be installed on Wazuh agents running on Debian/Ubuntu and CentOS operating systems. In this task, we will be using on a Ubuntu system. monitors the system for certain actions and events and will write this to a log file.

We can then use the log collector module on a Wazuh agent to read this log file and send it to the Wazuh management server for processing.

First, we will need to install the package and an plugin. This may already be installed on your system; however, let's install it to make sure. Let's run the command

and enable this service to run currently as well as on boot. &

We will need to configure to create a rule for the commands and events that we wish for it to monitor. In this task, we will be telling to monitor for any commands executed as root.

You can extend this to monitor commands such as , , or *catting* files such as */etc/passwd*, which are all hallmarks of a breach.

rules are located in the following directory: . We will be adding our rules manually.

For this task, we will need to open this *audit.rules* file and append our rule ourselves. First, let's edit the file using and appending

Monitoring commands executed as root

```
        ## First rule - delete all

-D


## Increase the buffers to survive stress events.

## Make this bigger for busy systems

-b 8192


## This determine how long to wait in burst of events

--backlog_wait_time 0


## Set failure mode to syslog

-f 1
```

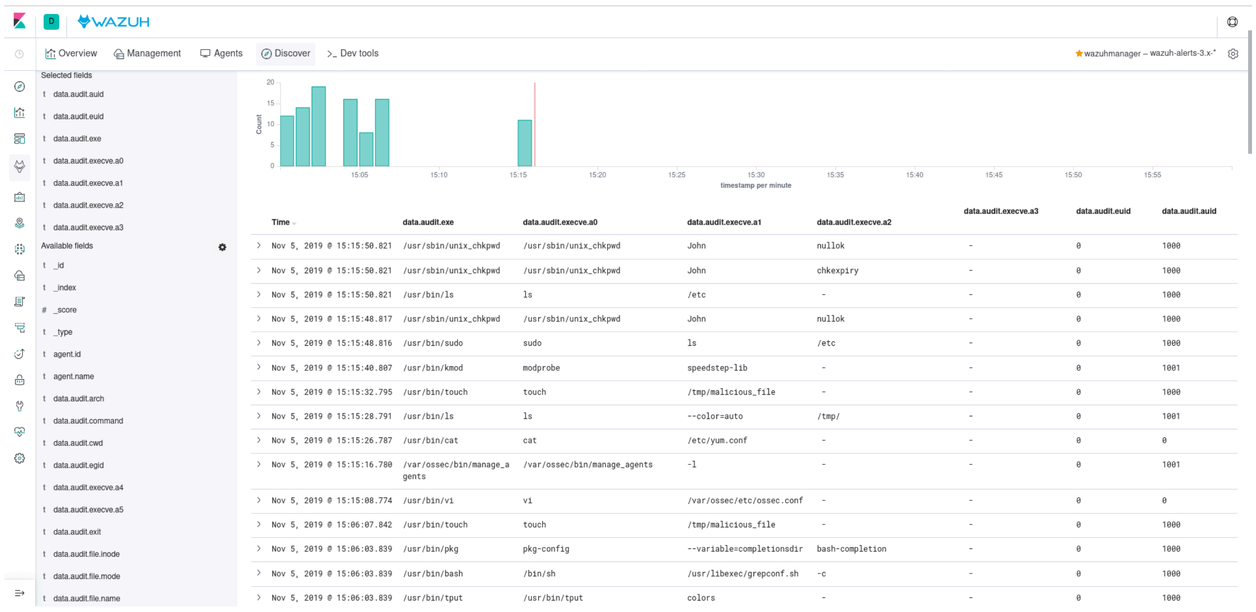
```
-a exit,always -F arch=b64 -F euid=0 -S execve -k  audit-wazuh-c
```

We will now need to inform audits of this new rule, so let's run this command
to now read the new *audit.rules* file that we appended to in the previous task.

Now, let's configure the system that is running a Wazuh agent that we wish to monitor these events on. We'll be monitoring a Linux host in this case, so like in our previous tasks, we will need to configure the Wazuh agent to detect this new log file that is generated by like so and add the log like so:

Configuring the Wazuh agent to add the auditd log as a log file to send to the Wazuh management server

```
<localfile>

    <location>/var/log/audit/audit.log</location>

    <log_format>audit</log_format>

</localfile>
```



Time	data.audit.exe	data.audit.execve.a0	data.audit.execve.a1	data.audit.execve.a2	data.audit.execve.a3	data.audit.euid	data.audit.auid
Nov 5, 2019 @ 15:15:50.821	/usr/sbin/unix_chkpwd	/usr/sbin/unix_chkpwd	John	nullok	-	0	1000
Nov 5, 2019 @ 15:15:50.821	/usr/sbin/unix_chkpwd	/usr/sbin/unix_chkpwd	John	chkexpiry	-	0	1000
Nov 5, 2019 @ 15:15:50.821	/usr/bin/ls	ls	/etc	-	-	0	1000
Nov 5, 2019 @ 15:15:48.817	/usr/sbin/unix_chkpwd	/usr/sbin/unix_chkpwd	John	nullok	-	0	1000
Nov 5, 2019 @ 15:15:48.816	/usr/bin/sudo	sudo	ls	/etc	-	0	1000
Nov 5, 2019 @ 15:15:40.807	/usr/bin/kmod	modprobe	speedstep-lib	-	-	0	1001
Nov 5, 2019 @ 15:15:32.795	/usr/bin/touch	touch	/tmp/malicious_file	-	-	0	1000
Nov 5, 2019 @ 15:15:28.791	/usr/bin/ls	ls	--color=auto	/tmp/	-	0	1001
Nov 5, 2019 @ 15:15:26.787	/usr/bin/cat	cat	/etc/yum.conf	-	-	0	0
Nov 5, 2019 @ 15:15:16.780	/var/ossec/bin/manage_agents	/var/ossec/bin/manage_agents	-l	-	-	0	1001
Nov 5, 2019 @ 15:15:08.774	/usr/bin/vi	vi	/var/ossec/etc/ossec.conf	-	-	0	0
Nov 5, 2019 @ 15:06:07.842	/usr/bin/touch	touch	/tmp/malicious_file	-	-	0	1000
Nov 5, 2019 @ 15:06:03.839	/usr/bin/pkg	pkg-config	--variable=completionsdir	bash-completion	-	0	1000
Nov 5, 2019 @ 15:06:03.839	/usr/bin/bash	/bin/sh	/usr/libexec/grepconf.sh	-c	-	0	1000
Nov 5, 2019 @ 15:06:03.839	/usr/bin/tput	/usr/bin/tput	colors	-	-	0	1000

Answer the questions below

What application do we use on Linux to monitor events such as command execution?

auditd

What is the full path & filename for where the aforementioned application stores rules?

/etc/audit/rules.d/audit.rules

Using Our Own Client

The Wazuh management server features a rich and extensive API to allow the Wazuh management server to be interacted with using the command line. Because the Wazuh management server requires authentication, we must first authenticate our client.

In this task, we will be using a Linux machine with the tool installed to interact with the Wazuh management server API. First, we will need to authenticate ourselves by providing a valid set of credentials to the authentication endpoint.

Once we are authenticated, the Wazuh management server will give us a token (similar to a session) that we will need to provide for any further interaction. We can store this token as an environment variable on our Linux machine like the snippet below:

(replacing *WAZUH_MANAGEMENT_SERVER_IP* with the IP address of the Wazuh management server (i.e. 10.10.192.44):

Let's confirm that we have authenticated okay and have been given a token by the Wazuh management server:

Wazuh API Verify Authentication

```
{
    "data": {
        "title": "Wazuh API",
        "api_version": "4.0.0",
        "revision": 4000,
        "license_name": "GPL 2.0",
        "license_url": "https://github.com/wazuh/wazuh/blob/master/LICENSE",
        "hostname": "wazuh-master",
        "timestamp": "2021-10-25T07:05:00+0000"
    },
    "error": 0
}
```

We can use the standard HTTP request methods such as   by providing the relevant option after a   i.e.

For example, let’s use the Wazuh API to list some statistics and important information about the Wazuh management server, including what services are being monitored and some general settings about the Wazuh management server:

Getting information about the Wazuh manager

```
          {
  "data": {
    "affected_items": [
      {
```

```
      "wazuh-agentlessd": "running",
      "wazuh-analysisd": "running",
      "wazuh-authd": "running",
      "wazuh-csyslogd": "running",
      "wazuh-dbd": "stopped",
      "wazuh-monitord": "running",
      "wazuh-execd": "running",
      "wazuh-integratord": "running",
      "wazuh-logcollector": "running",
      "wazuh-maild": "running",
      "wazuh-remoted": "running",
      "wazuh-reportd": "stopped",
      "wazuh-syscheckd": "running",
      "wazuh-clusterd": "running",
      "wazuh-modulesd": "running",
      "wazuh-db": "running",
      "wazuh-apid": "stopped"
    }
  ],
  "total_affected_items": 1,
  "total_failed_items": 0,
  "failed_items": []
},
"message": "Processes status were successfully read in specified node",
  "error": 0
}
```

Or perhaps, we can use the Wazuh management server’s API to interact with an agent:

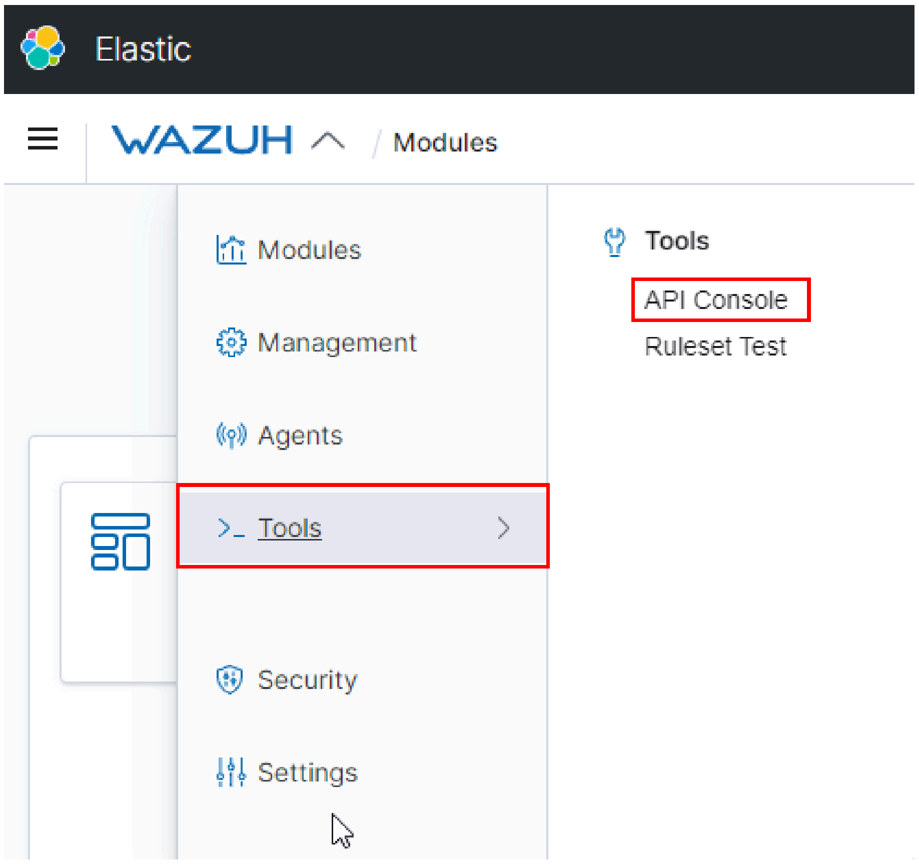
Using the Wazuh management server’s API to interact with an agent

```
        {
  "data": {
    "affected_items": [
      {
        "node_name": "worker2",
        "status": "active",
        "manager": "wazuh-worker2",
        "version": "Wazuh v3.13.1",
        "id": "001",
        "name": "wazuh-agent1"
      }
],
    "total_affected_items": 9,
    "total_failed_items": 0,
    "failed_items": []
  },
  "message": "All selected agents information was returned",

  "error": 0

}
```

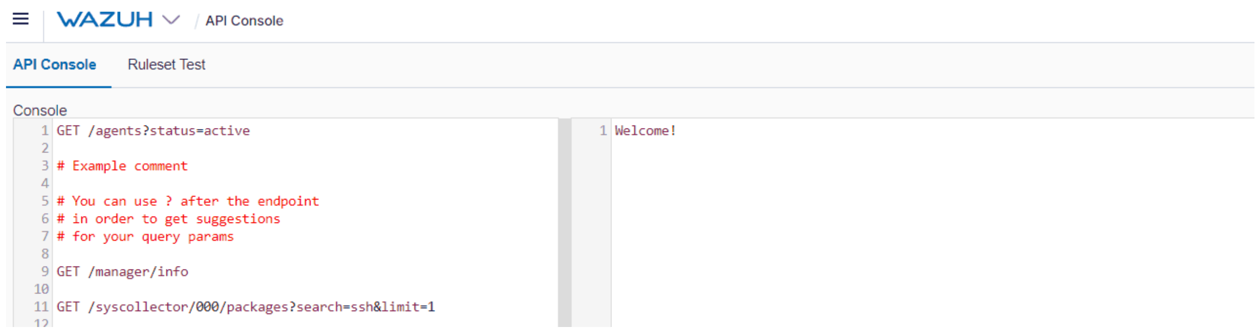
Using Wazuh's API Console

Wazuh has a powerful, integrated API console within the Wazuh website to query management servers and agents. Whilst it is not as extensive as using your own environment (where you can create and run scripts using python, for example), it is convenient.

To find this API console, we need to open the "Tools" category within the Wazuh heading at the top:



You will be greeted with a few sample queries that you can run. Simply *select* the line and *press* the green run arrow to run the query as demonstrated below:



```
GET /agents?status=active

# Example comment

# You can use ? after the endpoint
# in order to get suggestions
# for your query params

GET /manager/info

GET /syscollector/000/packages?search=ssh&limit=1
```

```
Welcome!
```

Reminder, the syntax for running queries uses the same web methods (i.e. GET/PUT/POST) and endpoints (i.e. /manager/info) as you would use with curl. You can view some more options about API endpoints by following Wazuh's detailed API documentation here

Answer the questions below

What is the name of the standard Linux tool that we can use to make requests to the Wazuh management server?

curl

What HTTP method would we use to retrieve information for a Wazuh management server API?

GET

What HTTP method would we use to perform an action on a Wazuh management server API?

PUT

Navigate to Wazuh's API console.

Use the API console to find the Wazuh server's version.

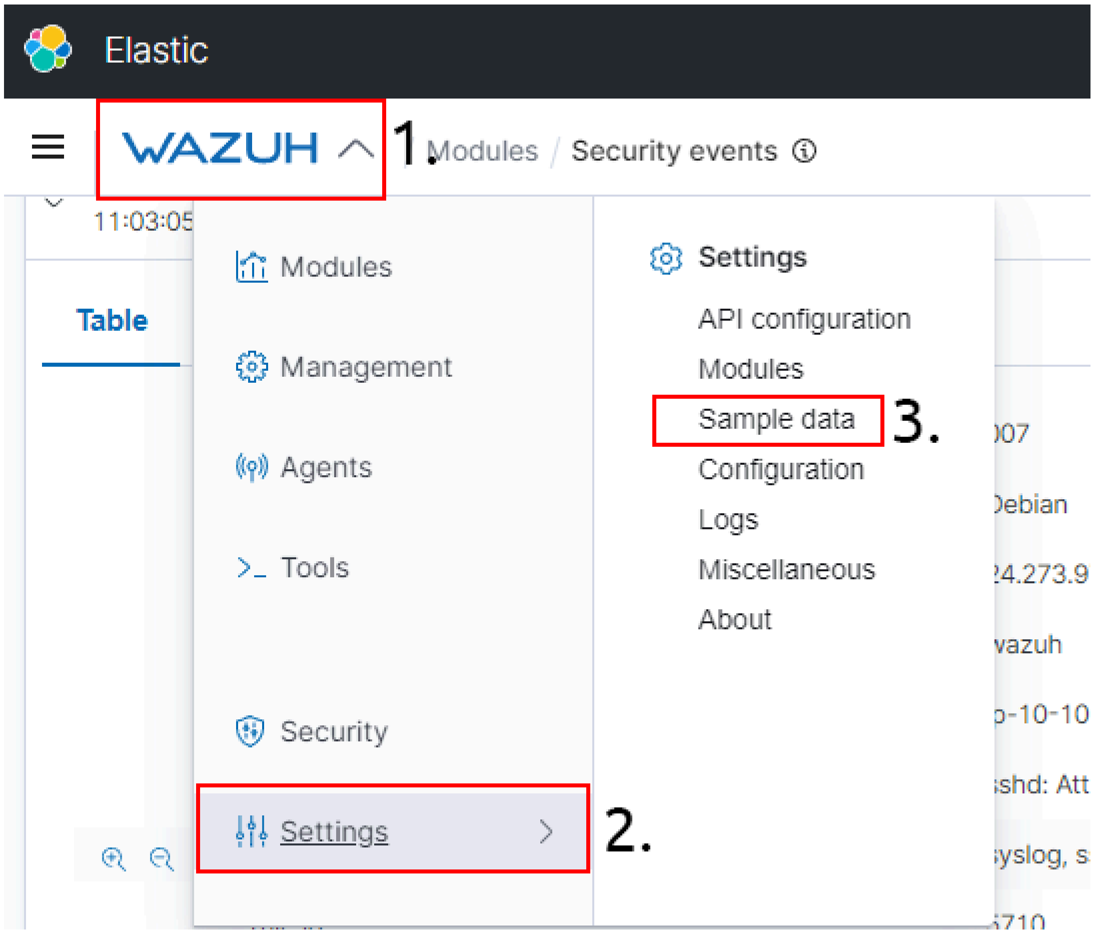
Note: You will need to add the "v" prefix to the number for this answer. For example v1.2.3

V4.2.5

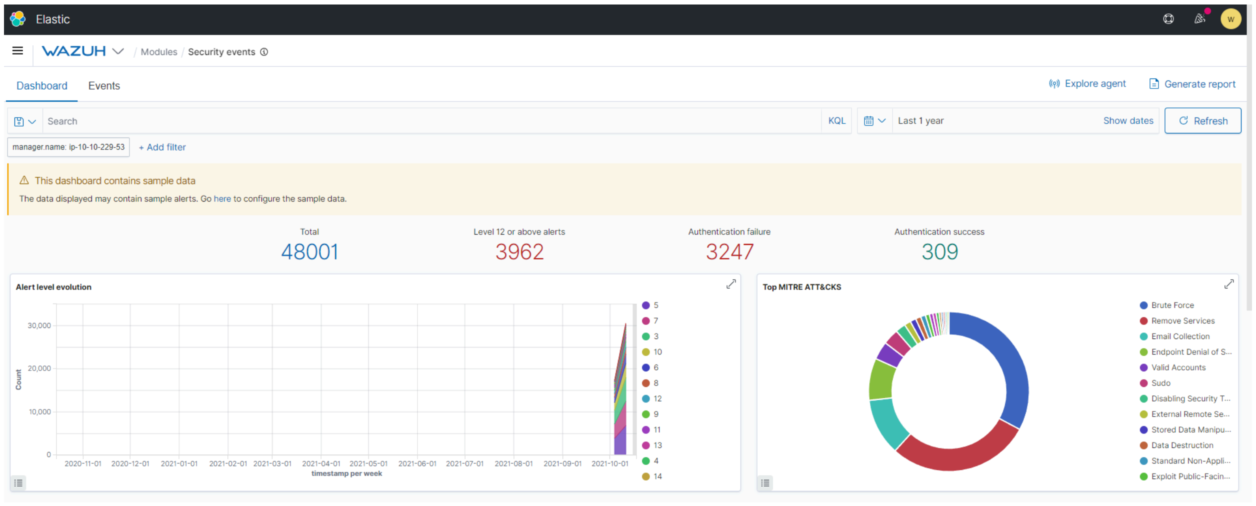
Task 11 Generating Reports with Wazuh

The Wazuh management server comes with sample data bundled with the installation that can be loaded at your convenience. I have not enabled this by default to improve the performance of the server. However, if you wish to import much more data to showcase the extensibility of Wazuh further, follow the steps below. Navigate to the module to load the sample data:

1. Open the "**Wazuh"** tab in the heading.
2. Highlight over "**Settings"**.
3. Select the "**Sample Data"** heading.
4. Press the "**Add Data**" button on the respective three cards to import the data.

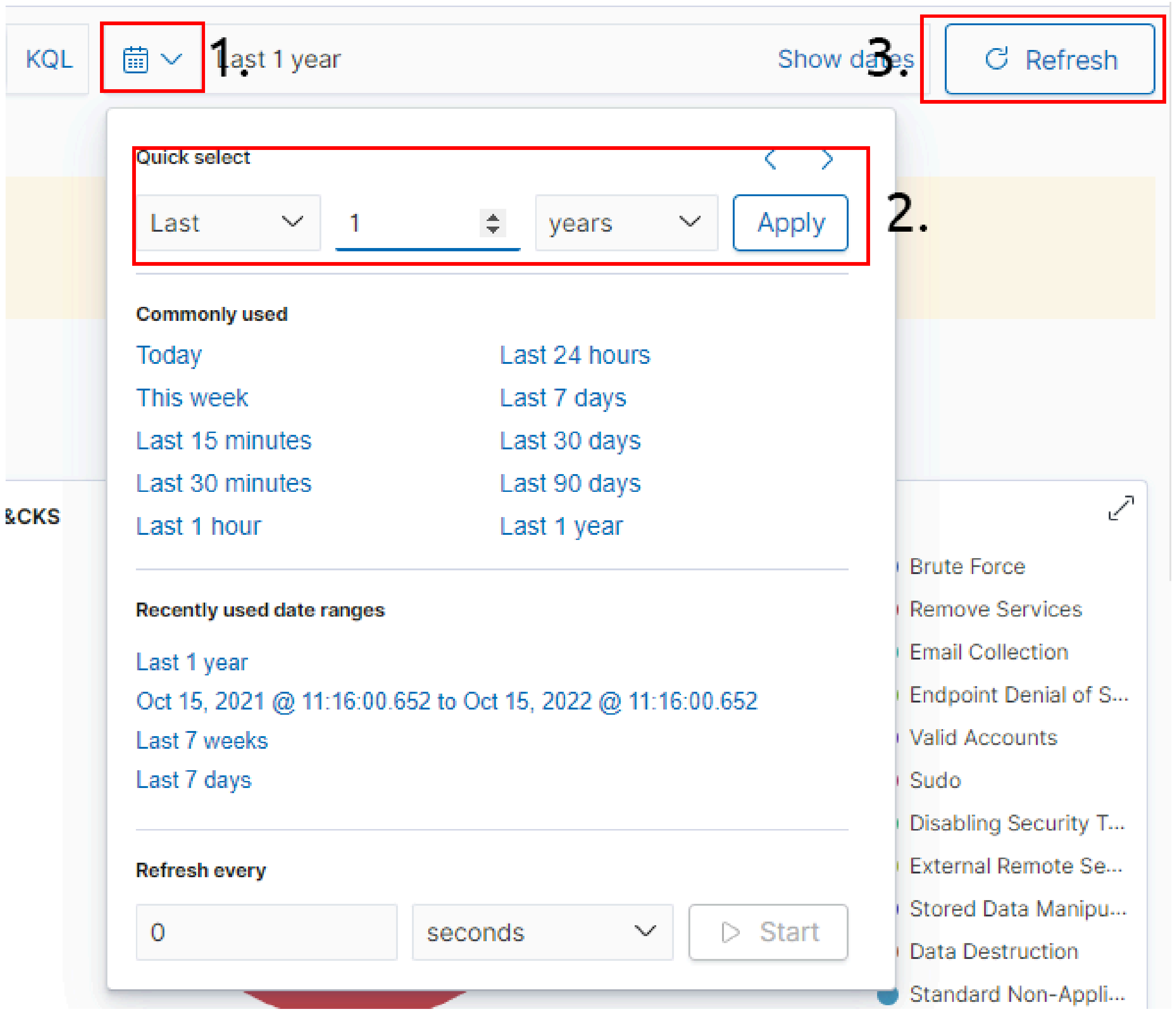


- Note that this may take up to a minute for each. Refer to this animated picture below for example. The data will have successfully imported when the button on the card says "Remove data"



Return to the Wazuh dashboard to see the newly imported data. For example, we can now see that the "Security Events" module has a tonne more data for us to explore.

Please note that you will need to play with the date range. The absolute minimum required to show the sample will need to be Last 7 days+ and refresh the dashboard for this to apply.



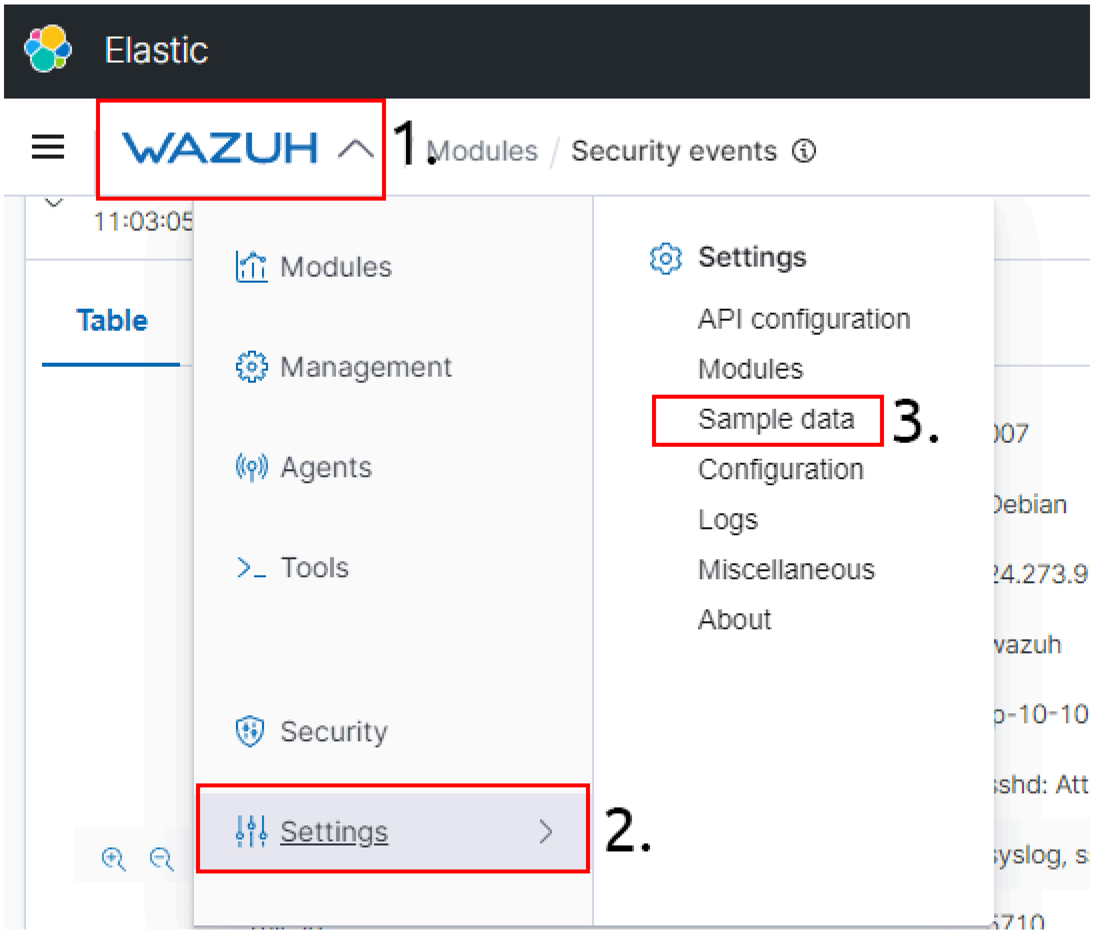
Analyse the report. What is the name of the agent that has generated the most alerts?

Agent-001

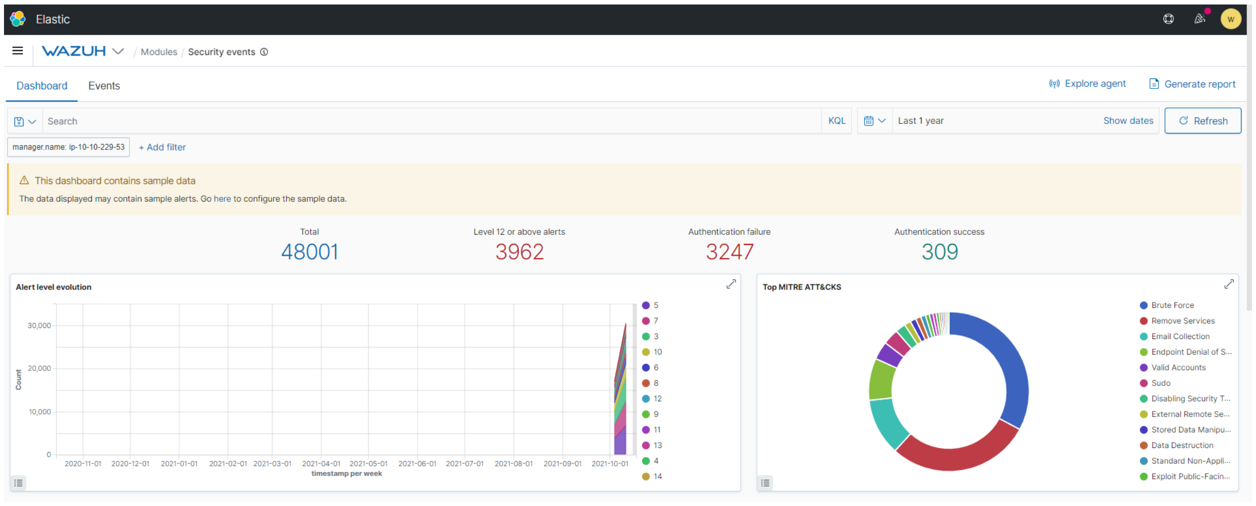
Task 12 **Loading Sample Data**

The Wazuh management server comes with sample data bundled with the installation that can be loaded at your convenience. I have not enabled this by default to improve the performance of the server. However, if you wish to import much more data to showcase the extensibility of Wazuh further, follow the steps below. Navigate to the module to load the sample data:

1. Open the "**Wazuh"** tab in the heading.
2. Highlight over "**Settings"**.
3. Select the "**Sample Data"** heading.
4. Press the "**Add Data**" button on the respective three cards to import the data.

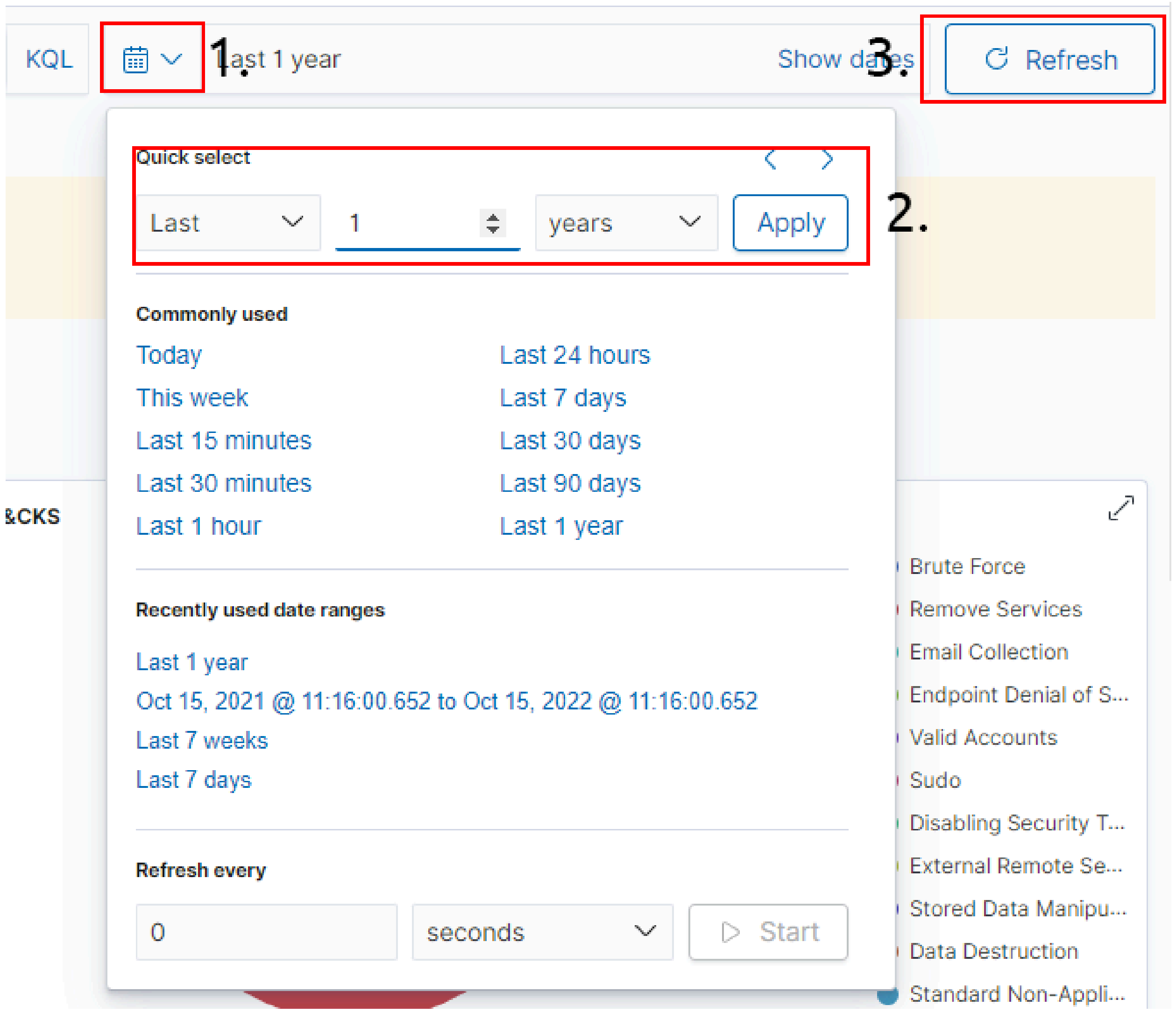


- Note that this may take up to a minute for each. Refer to this animated picture below for example. The data will have successfully imported when the button on the card says "Remove data"



Return to the Wazuh dashboard to see the newly imported data. For example, we can now see that the "Security Events" module has a tonne more data for us to explore.

Please note that you will need to play with the date range. The absolute minimum required to show the sample will need to be Last 7 days+ and refresh the dashboard for this to apply.



KQL | 1. Last 1 year | Show dates 3. | Refresh

Quick select 2.

Last | 1 | years | Apply

Commonly used

Today	Last 24 hours
This week	Last 7 days
Last 15 minutes	Last 30 days
Last 30 minutes	Last 90 days
Last 1 hour	Last 1 year

Recently used date ranges

Last 1 year
Oct 15, 2021 @ 11:16:00.652 to Oct 15, 2022 @ 11:16:00.652
Last 7 weeks
Last 7 days

Refresh every

0 | seconds | Start

&CKS

Brute Force
Remove Services
Email Collection
Endpoint Denial of S...
Valid Accounts
Sudo
Disabling Security T...
External Remote Se...
Stored Data Manipu...
Data Destruction
Standard Non-Appli...