

Enhancing Security Operations with S.O.C Automation using SIEM & SOAR

Md Ibtesam Hossain

| | |
|--|-----------|
| 1. INTRODUCTION | 2 |
| 1.2. Objectives and Goals | 2 |
| 2. METHODOLOGIES | 3 |
| 2.1. Description of the Tools | 3 |
| 2.2. Configuration Details | 3 |
| 3. IMPLEMENTATION | 3 |
| 3.1. Setting up the LAB Environment | 4 |
| 3.2. Installation of Wazuh and Agent | 6 |
| 3.3. Installation of TheHive | 8 |
| 3.4. Configuring Wazuh | 11 |
| 3.5. Configuring TheHive | 12 |
| 3.6. Configuring Shuffle | 18 |
| 4. TESTING DEPLOYMENTS | 20 |
| 4.1. Mimikatz | 20 |
| 4.2. Testing Shuffle | 23 |
| 5. AUTOMATION and RESPONSE | 23 |
| 5.1. Parsing Hash and Calling VirusTotal API | 23 |
| 5.2. Workflow Scenarios | 28 |
| 5.2.1. Scenario 1 | 28 |
| 5.2.2. Scenario 2 | 28 |
| 5.2.3. Other App Recommendations | 29 |
| 6. CONCLUSION | 30 |

1.INTRODUCTION

The “SOC Automation with SIEM and SOAR” project underscores my proficiency in cybersecurity by seamlessly integrating SIEM and SOAR technologies within a simulated SOC environment. Leveraging Wazuh as the SIEM solution, Shuffle as the SOAR platform, and TheHive for incident response and case management, the project showcased a robust infrastructure designed to enhance threat detection and incident response capabilities. Through the deployment of Wazuh agents on Windows clients, real-time Sysmon logs were transmitted to the Wazuh manager, enabling proactive threat monitoring. Shuffle's customized triggers facilitated the automatic generation of alerts based on predefined rules, while advanced workflows, such as the detection of mimikatz, enriched incident data by integrating VirusTotal intelligence. The project's outcomes included reduced incident response times, bolstered threat detection capabilities, and empowered analysts with enriched contextual information, emphasizing the pivotal role of SIEM and SOAR technologies in modern cybersecurity operations.

1.2. Objectives and Goals

The primary objective of the project was to enhance the effectiveness and efficiency of security operations by implementing automated workflows for threat detection and incident response. Specific goals included improving threat detection capabilities through real-time monitoring of Sysmon logs, streamlining incident response processes by automating alert generation and case management, enriching incident data with threat intelligence from external sources such as VirusTotal. Additionally, the project aimed to power security analysts with enriched contextual information and reduce incident response times, ultimately fortifying the organization's cybersecurity posture and resilience against emerging threats.

2.METHODOLOGIES

In this section I will provide an in-depth overview of the tools utilized, and the configuration details implemented throughout the project.

2.1. Description of the Tools

- **Wazuh as the SIEM Solution:** Wazuh, a robust open-source SIEM platform, was employed to collect, monitor, and analyze security event logs from various sources within the simulated environment. Its advanced capabilities facilitated real-time threat detection and response.
- **Shuffle as the SOAR Platform:** Shuffle, an open-source SOAR platform, was utilized to automate security workflows and orchestrate response actions based on predefined triggers and rules. It seamlessly integrated with Wazuh to enhance incident response capabilities.
- **TheHive for Incident Response and Case Management:** TheHive, a powerful open-source incident response platform, served as the central hub for managing and coordinating security incidents. It enabled security analysts to collaborate effectively and track the progress of investigations.

2.2. Configuration Details

- **Installation and Setup of Wazuh Agent on Windows Client:** A Wazuh agent was installed on the Windows client to facilitate the collection and transmission of Sysmon logs to the Wazuh manager. This agent served as the primary source of security event data within the environment.
- **Integration of Wazuh with Shuffle and TheHive:** Wazuh was seamlessly integrated with Shuffle and TheHive to enable automated alert generation and incident response workflows. Custom connectors and APIs were utilized to establish communication between the different components, ensuring smooth information exchange.
- **Customization of Alerts and Workflows:** Custom alerts and workflows were configured within Shuffle to automate specific security processes, such as alert triaging, enrichment, and response actions. These workflows were tailored to the organization's specific security requirements and operational workflows, enhancing overall efficiency and effectiveness.

3.IMPLEMENTATION





Detailed explanation of how the project was executed from creating each VMs to installing and configuring each application is given below.

3.1. Setting up the LAB Environment

This project required at least three machines, Wazuh manager, TheHive, and the Windows client. Wazuh manager and TheHive were installed in VM using Digital Ocean cloud. Windows 10 Pro VM was created on-prem using VMWarePlayer hypervisor.

For both Wazuh manager and TheHive was created using Ubuntu 22.04 LTS (x64) image with 8 GB Memory, 2 AMD vCPUs and a storage of 100 GB Disk.

Droplets


| Search by Droplet name | | Create Droplet | |
|---|----------------|----------------|--|
| Name | IP Address | Created ▲ | Tags |
|  thehive 8 GB / 2 AMD vCPUs / 100 GB Disk / TOR1 ... | 138.197.164.19 | 1 minute ago |  More ▼ |
|  Wazuh 8 GB / 2 AMD vCPUs / 100 GB Disk / TOR1 ... | 165.227.43.89 | 33 minutes ago |  More ▼ |

To secure these Droplets (Virtual Machines) from external adversaries while installing and configuring applications, a Firewall was deployed from the Networking Tab of the Digital Ocean. The firewall (WazuhFirewall) allowed in-bound traffic only from the on-prem Public IP.

Networking

Domains Reserved IPs Load Balancers VPC **Firewalls** PTR records

Create Firewall

| Name | Droplets | Rules | Created |
|---|----------|-------|---------------------------------|
|  WazuhFirewall | 0 | 6 | Just now More ▾ |

And finally, both the Wazuh and TheHive droplets were added to the Firewall to take advantage of the Traffic rules.

After all the VMs are in place, logged-in to both Wazuh and TheHive with SSH and performed updates and upgrades using

```
apt-get update && apt-get upgrade
```

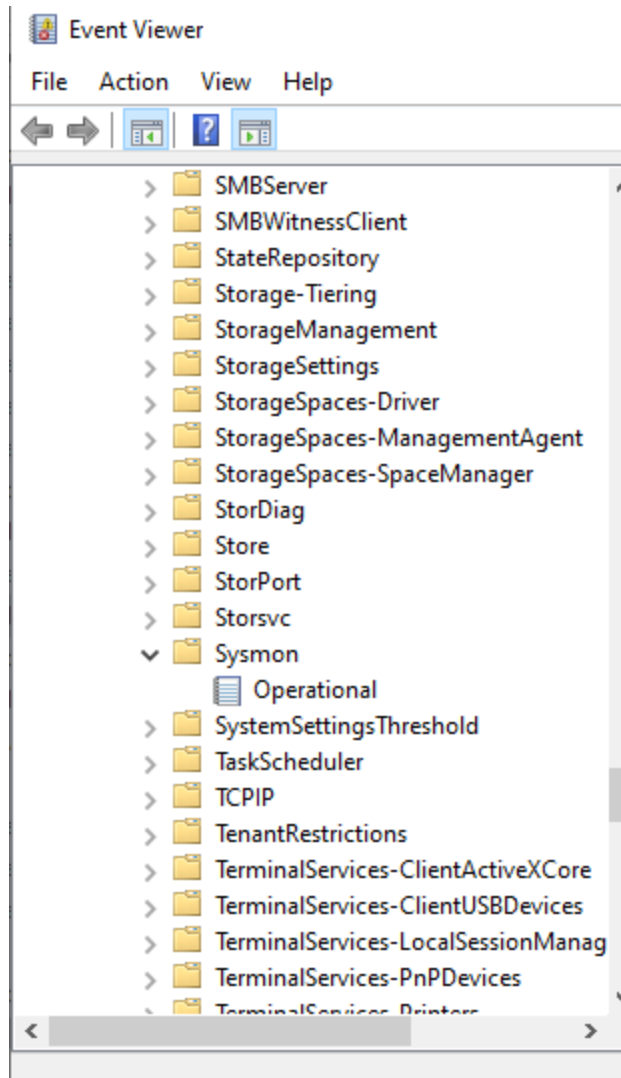
For on-prem Windows 10 pc, 50 GB Disk, 3 GB Memory and 2 AMD vCPUs were delegated from the Host system. Sysmon was downloaded from the Microsoft Sysinternals page and installed with the sysmon-config file.

```
Administrator: Windows PowerShell
PS C:\Users\Sam\Downloads\Sysmon> .\Sysmon64.exe -i '.\sysmon config.xml'

System Monitor v15.14 - System activity monitor
By Mark Russinovich and Thomas Garnier
Copyright (C) 2014-2024 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 4.90
Configuration file validated.
Sysmon64 installed.
SysmonDrv installed.
Starting SysmonDrv.
SysmonDrv started.
Starting Sysmon64..
Sysmon64 started.
PS C:\Users\Sam\Downloads\Sysmon>
```

We can verify that Sysmon was properly installed by looking through the Windows event Viewer logs given below.



3.2. Installation of Wazuh and Agent

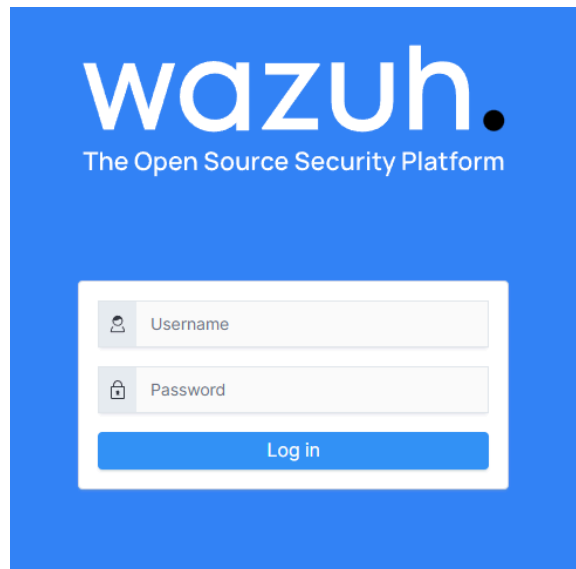
Wazuh was installed by following the documentation provided by Wazuh in their website. The step-by-step action taken is given below.

Downloaded and ran the Wazuh installation agent with the following command

```
curl -s0 https://packages.wazuh.com/4.7/wazuh-install.sh && sudo bash
./wazuh-install.sh -a
```

Once the assistant finishes the installation, the output shows the access credentials and a message that confirms that the installation was successful.

Accessed the Wazuh web interface using the `https://<wazuh-public-ip>` and the credentials.



And with successful login we can see the home page of our Wazuh SIEM.




The Wazuh SIEM home page displays a dashboard with the following components:

- Header:** Includes a menu icon, the Wazuh logo, a "Modules" dropdown, and a user profile icon.
- Agent Status Summary:** A row of five cards showing agent counts: Total agents (0), Active agents (0), Disconnected agents (0), Pending agents (0), and Never connected agents (0).
- Alerts:** A yellow banner message stating "No agents were added to this manager. Add agent".
- Security Information Management (SIM):**
 - Security events:** Browse through your security alerts, identifying issues and threats in your environment.
 - Integrity monitoring:** Alerts related to file changes, including permissions, content, ownership and attributes.
- Auditing and Policy Monitoring (APM):**
 - Policy monitoring:** Verify that your systems are configured according to your security policies baseline.
 - System auditing:** Audit users behavior, monitoring command execution and alerting on access to critical files.
 - Security configuration assessment:** Scan your assets as part of a configuration assessment audit.
- Threat Detection and Response (TDR):**
 - Vulnerabilities:** Discover what applications in your environment are affected by well-known vulnerabilities.
 - MITRE ATT&CK:** Security events from the knowledge base of adversary tactics and techniques based on real-world observations.
- Regulatory Compliance (RC):**
 - PCI DSS:** Global security standard for entities that process, store or transmit payment cardholder data.
 - TSC:** Trust Services Criteria for Security, Availability, Processing Integrity, Confidentiality, and Privacy.
 - NIST 800-53:** National Institute of Standards and Technology Special Publication 800-53 (NIST 800-53) sets guidelines for federal information systems.
 - GDPR:** General Data Protection Regulation (GDPR) sets guidelines for processing of personal data.

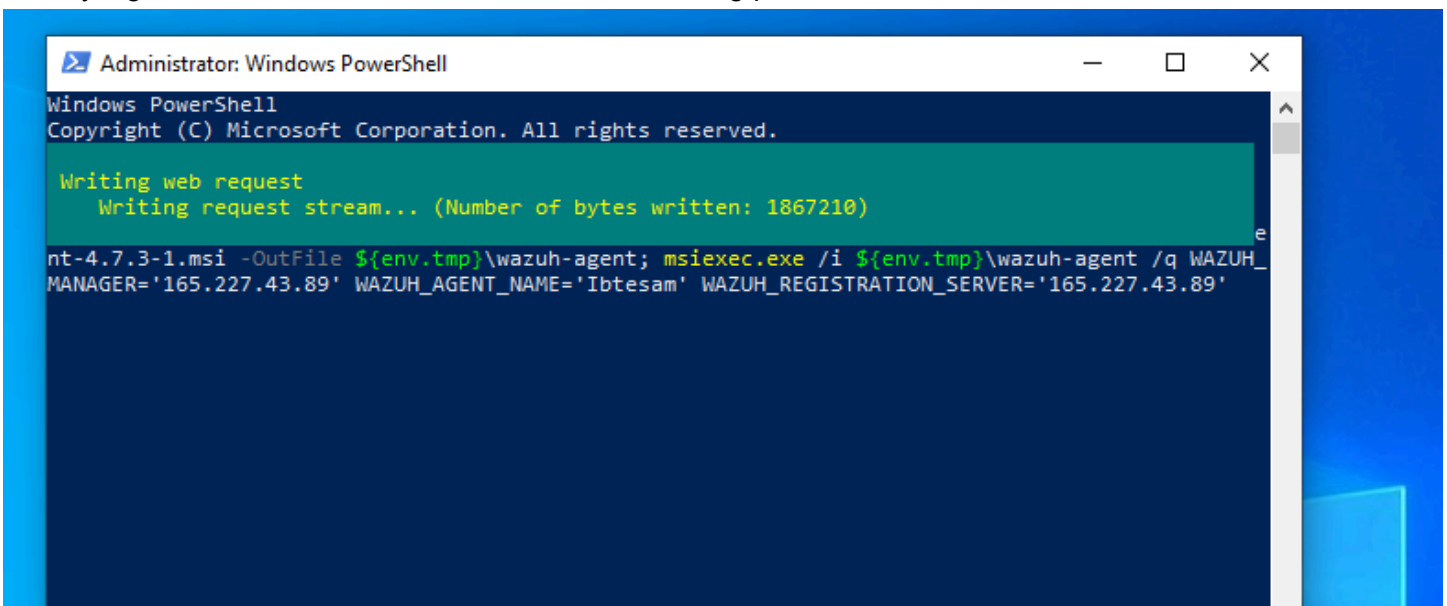
Next, agent was deployed by going to the agents option from the dropdown and selecting the Windows option.

Deploy new agent

1 Select the package to download and install on your system:

| | | |
|--|--|--|
|  LINUX <input type="radio"/> RPM amd64 <input type="radio"/> RPM aarch64 <input type="radio"/> DEB amd64 <input type="radio"/> DEB aarch64 |  WINDOWS <input type="radio"/> MSI 32/64 bits |  macOS <input type="radio"/> Intel <input type="radio"/> Apple silicon |
|--|--|--|

After including Wazuh's public IP into the server address, an agent installation command was generated. Finally, agent was installed into the Windows client using powershell.



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

Writing web request
Writing request stream... (Number of bytes written: 1867210)
nt-4.7.3-1.msi -OutFile ${env.tmp}\wazuh-agent; msixec.exe /i ${env.tmp}\wazuh-agent /q WAZUH_
MANAGER='165.227.43.89' WAZUH_AGENT_NAME='Ibtesam' WAZUH_REGISTRATION_SERVER='165.227.43.89'
```

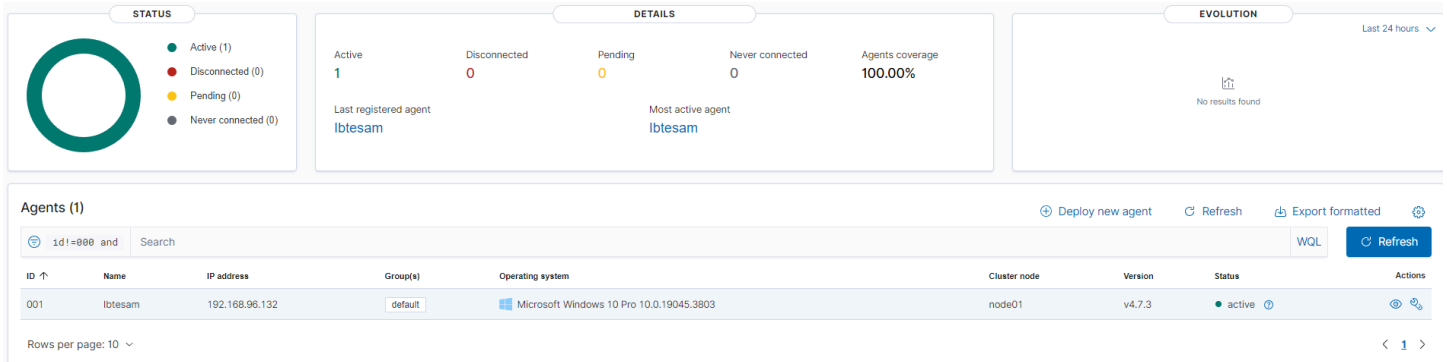
After installation, wazuh agent was started using the command

```
net start wazuhsvc
```

```
PS C:\Windows\system32> net start wazuhsvc
The Wazuh service is starting.
The Wazuh service was started successfully.

PS C:\Windows\system32>
```

Heading back into the Wazuh web interface it is seen that the agent is successfully integrated.



Also, the security events started to show up.

| Security Alerts | | | | | | |
|-----------------------------|--------------|-----------|--|-------|---------|--|
| Time | Technique(s) | Tactic(s) | Description | Level | Rule ID | |
| Apr 23, 2024 @ 18:23:14.989 | | | SCA summary: CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Score less than 50% (32) | 7 | 19004 | |
| Apr 23, 2024 @ 19:22:57.582 | | | SCA summary: CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Score less than 50% (32) | 7 | 19004 | |
| Apr 23, 2024 @ 19:22:50.560 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'Select when Quality Updates are received' is set to 'Enabled: 0 days'. | 3 | 19009 | |
| Apr 23, 2024 @ 19:22:50.527 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'Select when Preview Builds and Feature Updates are received' is set to 'Enabled: 180 or more days'. | 7 | 19007 | |
| Apr 23, 2024 @ 19:22:50.495 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'Manage preview builds' is set to 'Disabled'. | 3 | 19008 | |
| Apr 23, 2024 @ 19:22:50.478 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'Remove access to "Pause updates" feature' is set to 'Enabled'. | 7 | 19007 | |
| Apr 23, 2024 @ 19:22:50.472 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'Configure Automatic Updates: Scheduled install day' is set to '0 - Every day'. | 3 | 19009 | |
| Apr 23, 2024 @ 19:22:50.443 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'Configure Automatic Updates' is set to 'Enabled'. | 3 | 19008 | |
| Apr 23, 2024 @ 19:22:50.442 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'No auto-restart with logged on users for scheduled automatic updates installations' is set to 'Disabled'. | 3 | 19008 | |
| Apr 23, 2024 @ 19:22:50.424 | | | CIS Microsoft Windows 10 Enterprise Benchmark v1.12.0: Ensure 'Prevent users from modifying settings' is set to 'Enabled'. | 7 | 19007 | |

3.3. Installation of TheHive

Similar to the Wazuh, TheHive installation followed procedures outlined in their official documentation page. Step by step actions taken are given below.

TheHive is installed with the following technologies.

Standalone Server

Cluster or Hybrid Architecture

A standalone server setup involves installing all necessary components on a single server:

- Cassandra
- Elasticsearch
- File storage on the local filesystem (or MinIO if desired)
- TheHive
- Optional NGINX for managing HTTPS communications

For detailed installation instructions, refer to the step-by-step installation guide.

In this project, Java was installed first as required by TheHive, and followed by Cassandra, Elasticsearch and TheHive. For file storage, local filesystem is used.

Java installation:

```
wget -qO- https://apt.corretto.aws/corretto.key | sudo gpg --dearmor -o
/usr/share/keyrings/corretto.gpg
echo "deb [signed-by=/usr/share/keyrings/corretto.gpg] https://apt.corretto.aws stable
main" | sudo tee -a /etc/apt/sources.list.d/corretto.sources.list
sudo apt update
sudo apt install java-common java-11-amazon-corretto-jdk
echo JAVA_HOME="/usr/lib/jvm/java-11-amazon-corretto" | sudo tee -a /etc/environment
export JAVA_HOME="/usr/lib/jvm/java-11-amazon-corretto"
```

Verifying the installation by running `java-version`

```
root@thehive:~# java -version
openjdk version "11.0.23" 2024-04-16 LTS
OpenJDK Runtime Environment Corretto-11.0.23.9.1 (build 11.0.23+9-LTS)
OpenJDK 64-Bit Server VM Corretto-11.0.23.9.1 (build 11.0.23+9-LTS, mixed mode)
root@thehive:~#
```

Cassandra installation:

Adding Cassandra repository reference

```
wget -qO - https://downloads.apache.org/cassandra/KEYS | sudo gpg --dearmor -o
/usr/share/keyrings/cassandra-archive.gpg
```

Adding the repository to the system by appending the following line

```
echo "deb [signed-by=/usr/share/keyrings/cassandra-archive.gpg]
https://deb.debian.cassandra.apache.org 40x main" | sudo tee -a
/etc/apt/sources.list.d/cassandra.sources.list
```

Installing the package

```
sudo apt update
sudo apt install cassandra
```

Elasticsearch Installation:

Adding elasticsearch repository references

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o /usr/share/keyrings/elasticsearch-keyring.gpg  
sudo apt-get install apt-transport-https
```

Adding repository to the system by appending the following line

```
echo "deb [signed-by=/usr/share/keyrings/elasticsearch-keyring.gpg]  
https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee  
/etc/apt/sources.list.d/elastic-7.x.list
```

Installing the package

```
sudo apt update  
sudo apt install elasticsearch
```

TheHive installation:

Getting the repository

```
wget -O- https://archives.strangebee.com/keys/strangebee.gpg | sudo gpg --dearmor -o /usr/share/keyrings/strangebee-archive-keyring.gpg
```

Installing TheHive package by running

```
echo 'deb [arch=all signed-by=/usr/share/keyrings/strangebee-archive-keyring.gpg]  
https://deb.strangebee.com thehive-5.3 main' | sudo tee -a  
/etc/apt/sources.list.d/strangebee.list  
sudo apt-get update  
sudo apt-get install -y thehive
```

Local Filesystem:

Creating directory

```
sudo mkdir -p /opt/thp/thehive/files
```

TheHive requires that the user and group thehive:thehive has permission on the filepath of the storage. So changed the file permission by running

```
chown -R thehive:thehive /opt/thp/thehive/files
```

```

root@thehive:~# ls -la /opt/thp
total 12
drwxr-xr-x 3 root root 4096 Apr 23 22:24 .
drwxr-xr-x 5 root root 4096 Apr 23 22:24 ..
drwxr-xr-x 5 root root 4096 Apr 23 22:24 thehive
root@thehive:~# chown -R thehive:thehive /opt/thp
root@thehive:~# ls -la /opt/thp
total 12
drwxr-xr-x 3 thehive thehive 4096 Apr 23 22:24 .
drwxr-xr-x 5 root     root     4096 Apr 23 22:24 ..
drwxr-xr-x 5 thehive thehive 4096 Apr 23 22:24 thehive

```

3.4. Configuring Wazuh

For this project only Sysmon logs were selected for the log ingestion. To add this property, the agent ossec config file was edited by adding the following configuration

```

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

```

Also, to enable all logs the ossec config file in the Wazuh server was also changed by adding following parameters

```

<ossec_config>
  <global>
    <jsonout_output>yes</jsonout_output>
    <alerts_log>yes</alerts_log>
    <logall>yes</logall>
    <logall_json>yes</logall_json>

```

This **logall** option generates log archives

```

root@Wazuh:~# cd /var/ossec/logs/archives/
root@Wazuh:/var/ossec/logs/archives# ls
2024 archives.json archives.log

```

According to Wazuh documentation, this new change should be updated for the filebeat yml file as well.

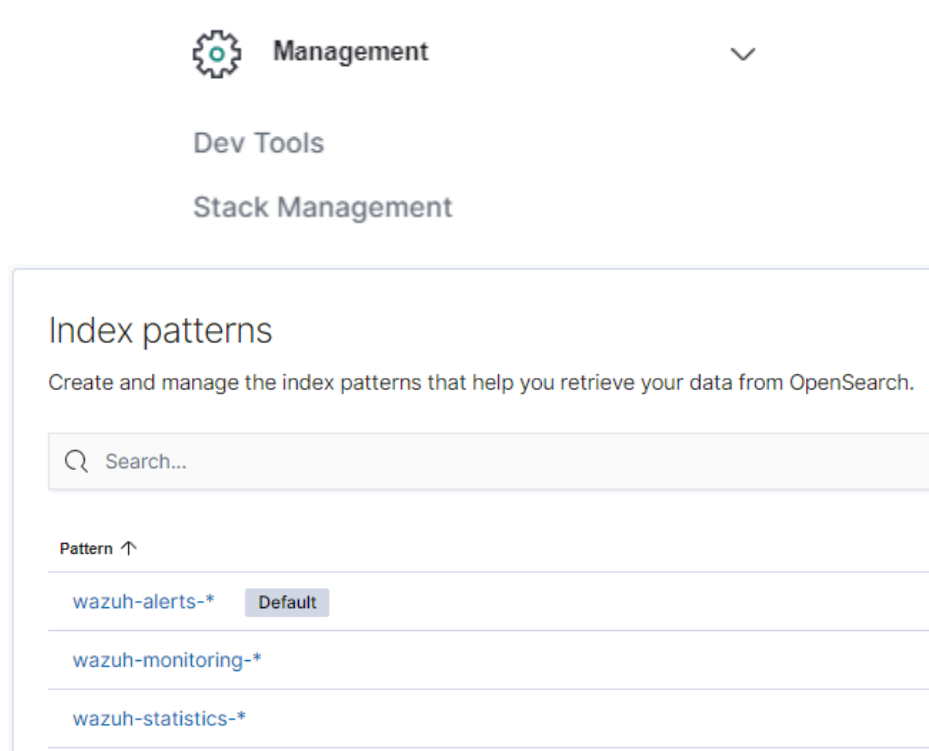
```

root@Wazuh:/var/ossec/logs/archives# nano /etc/filebeat/filebeat.yml

```

```
filebeat.modules:
- module: wazuh
  alerts:
    enabled: true
  archives:
    enabled: false
```

Returning to the Wazuh web interface a new index pattern was created to detect all the new logs from the logall archives. This option can be found at the Stack Management link under Management from the sidebar.



Adding new index: **wazuh-archives-***

Step 2 of 2: Configure settings

Specify settings for your **wazuh-archives-*** index pattern.

Select a primary time field for use with the global time filter.

Time field Refresh

timestamp
▼

3.5. Configuring TheHive

TheHive configuration starts with configuring Cassandra. This file is at the `/etc/cassandra/cassandra.yaml`

```

cluster_name: 'thp'
listen_address: 'xx.xx.xx.xx' # address for nodes
rpc_address: 'xx.xx.xx.xx' # address for clients
seed_provider:
  - class_name: org.apache.cassandra.locator.SimpleSeedProvider
    parameters:
      # Ex: "<ip1>,<ip2>,<ip3>"
      - seeds: 'xx.xx.xx.xx' # self for the first node
data_file_directories:
  - '/var/lib/cassandra/data'
commitlog_directory: '/var/lib/cassandra/commitlog'
saved_caches_directory: '/var/lib/cassandra/saved_caches'
hints_directory:
  - '/var/lib/cassandra/hints'
[...]
```

Here, the cluster_name and the seed_provider option was modified. Wazug requires either the listen address to be modified or the seed. The seed was used as it provided smooth configuration without any bugs. Everything was kept default. Finally started Cassandra and made sure it was running.

```

root@thehive:~# systemctl start cassandra.service
root@thehive:~# systemctl status cassandra.service
● cassandra.service - LSB: distributed storage system for structured data
   Loaded: loaded (/etc/init.d/cassandra; generated)
   Active: active (running) since Tue 2024-04-23 22:34:08 UTC; 12s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 25610 ExecStart=/etc/init.d/cassandra start (code=exited, status=0/SUCCESS)
    Tasks: 51 (limit: 9492)
   Memory: 2.1G
      CPU: 16.111s
   CGroup: /system.slice/cassandra.service
           └─25710 /usr/bin/java -ea -da:net.openhft... -XX:+UseThreadPriorities -X

Apr 23 22:34:08 thehive systemd[1]: Starting LSB: distributed storage system for structured data
Apr 23 22:34:08 thehive systemd[1]: Started LSB: distributed storage system for structured data
root@thehive:~#
```

After that, elasticsearch was configured using the elasticsearch.yml file in the directory

```
/etc/elasticsearch/elasticsearch.yml
```

Added cluster name "thehive" and node name as "node-1". Confirmed path data and logs are correct.

```

#
cluster.name: thehive
#
# -----
#
# Use a descriptive name for the node
#
node.name: node-1
#
# Add custom attributes to the node
#
#node.attr.rack: r1
#
# -----
#
# Path to directory where to store
#
path.data: /var/lib/elasticsearch
#
# Path to log files:
#
path.logs: /var/log/elasticsearch
#

```

Added the public IP of the thehive as network host and thehive requires port 9200 as default port.

```

# ----- Network
#
# By default Elasticsearch is only accessible
# address here to expose this node on the net
#
network.host: 138.197.164.19
#
# By default Elasticsearch listens for HTTP t
# finds starting at 9200. Set a specific HTTP
#
http.port: 9200
#
# For more information, consult the network m
#
# ----- Discovery
#
# Pass an initial list of hosts to perform di
# The default list of hosts is ["127.0.0.1",
#
#discovery.seed_hosts: ["host1", "host2"]
#
# Bootstrap the cluster using an initial set
#
cluster.initial_master_nodes: ["node-1"]
#

```

Finally started elasticsearch and confirmed it was running.

```
root@thehive:~# systemctl status elasticsearch.service
● elasticsearch.service - Elasticsearch
   Loaded: loaded (/lib/systemd/system/elasticsearch.serv
   Active: active (running) since Tue 2024-04-23 22:42:15
     Docs: https://www.elastic.co
  Main PID: 26262 (java)
    Tasks: 57 (limit: 9492)
   Memory: 4.3G
      CPU: 1min 6.245s
   CGroup: /system.slice/elasticsearch.service
           └─26262 /usr/share/elasticsearch/jdk/bin/java
             └─26451 /usr/share/elasticsearch/modules/x-pack

Apr 23 22:41:49 thehive systemd[1]: Starting Elasticsearch.
Apr 23 22:41:55 thehive systemd-entrpoint[26262]: Apr 23,
Apr 23 22:41:55 thehive systemd-entrpoint[26262]: WARNING:
Apr 23 22:42:15 thehive systemd[1]: Started Elasticsearch.
lines 1-16/16 (END)
```

TheHive is finally configured by using the application.conf file found inside the
`/etc/thehive/application.conf`

```
# Database and index configuration
# By default, TheHive is configured to connect to local Cassandra 4.x and a
# local Elasticsearch services without authentication.
db.janusgraph {
  storage {
    backend = cql
    hostname = ["127.0.0.1"]
    # Cassandra authentication (if configured)
    # username = "thehive"
    # password = "password"
    cql {
      cluster-name = thp
      keyspace = thehive
    }
  }
}
index.search {
  backend = elasticsearch
  hostname = ["127.0.0.1"]
  index-name = thehive
}
}
```

Here, changed the host name and the cluster name as following


```
# local Elasticsearch services without authentication
db.janusgraph {
  storage {
    backend = cql
    hostname = ["138.197.164.19"]
    # Cassandra authentication (if configured)
    # username = "thehive"
    # password = "password"
    cql {
      cluster-name = ibtesam
      keyspace = thehive
    }
  }
  index.search {
    backend = elasticsearch
    hostname = ["138.197.164.19"]
    index-name = thehive
  }
}
```

Lastly made sure the application base url reflects the current address.

```

# Attachment storage configuration
# By default, TheHive is configured to store files locally in the folder.
# The path can be updated and should belong to the user/group running thehive
storage {
  provider = localfs
  localfs.location = /opt/thp/thehive/files
}

# Define the maximum size for an attachment accepted by TheHive
play.http.parser.maxDiskBuffer = 1GB
# Define maximum size of http request (except attachment)
play.http.parser.maxMemoryBuffer = 10M

# Service configuration
application.baseUrl = "http://138.197.164.19:9000"
play.http.context = "/"

# Additional modules
#
# TheHive is strongly integrated with Cortex and MISP.
# Both modules are enabled by default. If not used, each one can be disabled
# commenting the configuration line.
scalligraph.modules += org.thp.thehive.connector.cortex.CortexModule
scalligraph.modules += org.thp.thehive.connector.misp.MispModule

```

Confirmed thehive was started and running.

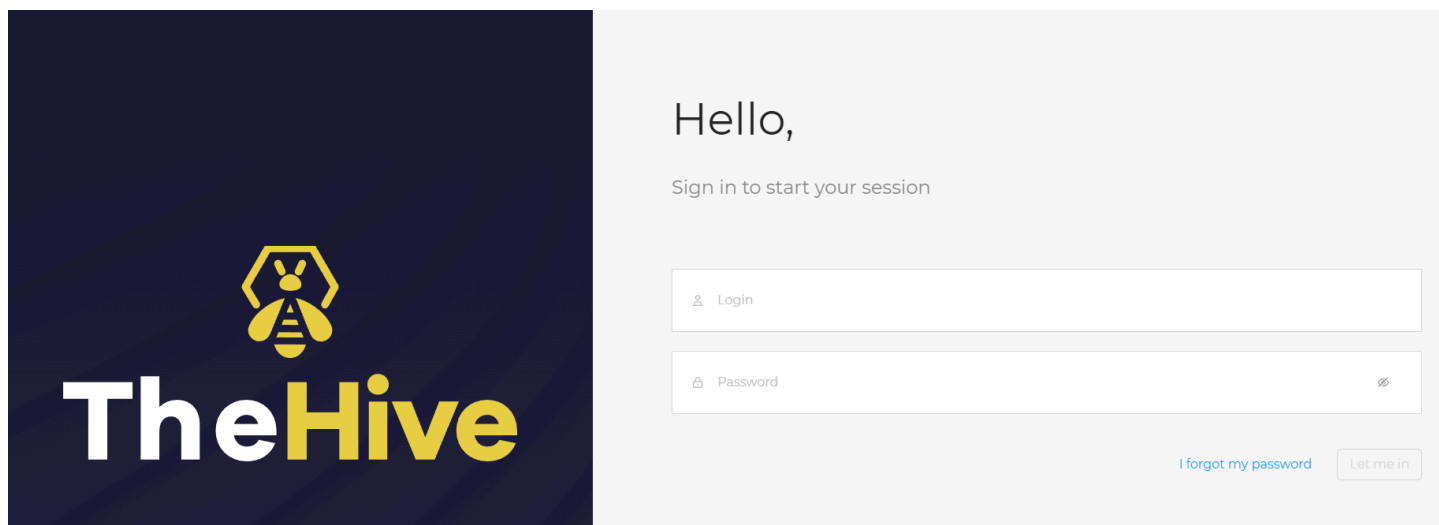
```

root@thehive:~# systemctl status thehive
● thehive.service - Scalable, Open Source a
   Loaded: loaded (/lib/systemd/system/th
   Active: active (running) since Tue 202
     Docs: https://thehive-project.org
  Main PID: 27533 (java)
    Tasks: 82 (limit: 9492)
   Memory: 586.0M
      CPU: 57.729s
   CGroup: /system.slice/thehive.service
           └─27533 java -Dfile.encoding=U

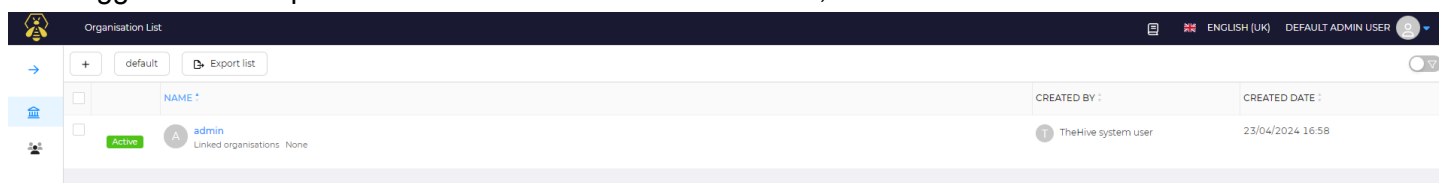
Apr 23 22:57:06 thehive systemd[1]: Started
lines 1-12/12 (END)

```

Also, TheHive web interface was active and running.



After logged-in as the provided credential in the documentation, an admin is seen on the console



Inside shuffle as admin user, added a new organization “SOC Automation” and included two new accounts. One user account is “Ibtessam” which acts as an analyst and the other is a service account named “SOAR” which will be used to integrate with Shuffle. TheHive has a predefined permission set for the **analyst** type but no template for the **service** account. It is best practice to allow Least Privilege options.

Adding an Organisation

*

Name

SOC Automation

*

Description

Security Operations Center

Tasks sharing rule

manual

Observables sharing rule

manual

Type

Normal

Service users are essentially used for bots (API key authentication).

Organisation

SOC Automation

* Login

ibtesam@test.com

* Name

Ibtesam

* Profile

analyst

Permissions

accessTheHiveFS

manageAction

manageAlert/create

manageAlert/delete

manageAlert/import

manageAlert/reopen

manageAlert/update

manageAnalyse

manageCase/changeOwnership

manageCase/create

manageCase/delete

manageCase/merge

manageCase/reopen

manageCase/update

manageCaseReport

manageComment

manageCustomEvent

manageFunction/invoke

manageKnowledgeBase

manageObservable

managePage

manageProcedure

manageShare

manageTask

Type

Service

Service users are essentially used for bots (API key authentication).

Organisation

SOC Automation

* Login

shuffle@test.com

* Name

SOAR

* Profile

analyst

Permissions

accessTheHiveFS

manageAction

manageAlert/create

manageAlert/delete

manageAlert/import

manageAlert/reopen

manageAlert/update

manageAnalyse

manageCase/changeOwnership

manageCase/create

manageCase/delete

manageCase/merge

manageCase/reopen

manageCase/update

manageCaseReport

manageComment

manageCustomEvent

manageFunction/invoke

manageKnowledgeBase

manageObservable

managePage

manageProcedure

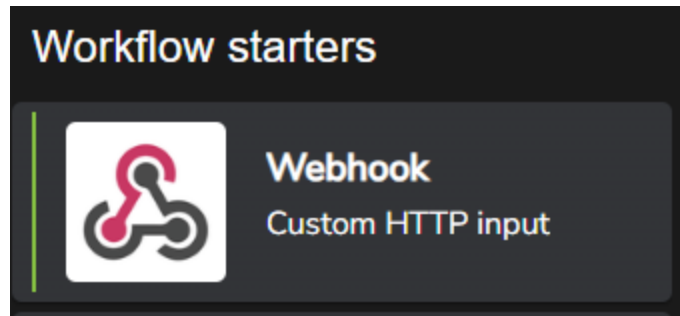
manageShare

manageTask

Created a new password for the analyst account and logged in to start receiving alerts. From the service account, an API key can be found and copied to later integrate with shuffle.

3.6. Configuring Shuffle

Shuffle can be used by going to their website shuffler.io and after logging an workflow tab can be found. All the automation actions are then configured using the workflow section. To configure shuffle first a webhook was selected from the triggers section.



By dragging to add it in the workflow and clicking on it opens further settings.

Webhook: uninitialized
[What are webhooks?](#)


Name
Wazuh-Alerts

Find Associated App (optional) ▼

Environment
cloud ▼

Parameters

● **Webhook URI**

https://shuffler.io/api/v1/hooks/w 

START STOP

● **Authentication headers**

AUTH_HEADER=AUTH_VALUE1

The Webhook URI is copied and added to Wazuh to create a connection between both of them. Config shown below was added into the ossec.conf file inside the Wazuh Manager server.

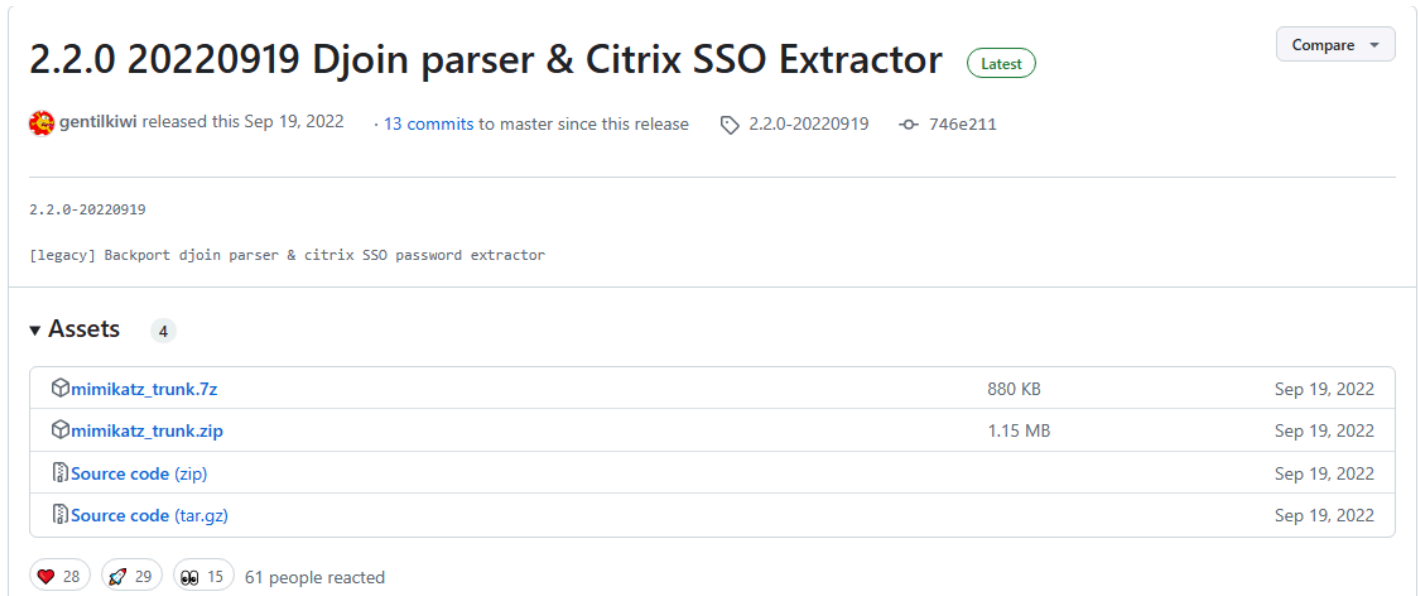
```
<integration>
  <name>shuffle</name>
  <hook_url>https://shuffler.io/api/v1/hooks/webhook_74155535-f145-42ca-8019-be574dd10d8b </hook_url>
  <rule_id>100002</rule_id>
  <alert_format>json</alert_format>
</integration>
```

Restarted the service as the config file was modified and confirmed the status running.

4. TESTING DEPLOYMENTS

4.1. Mimikatz

The Windows client was sending Sysmon logs to Wazuh and it was successfully showing up in the security events tab. To simulate an attack to the windows pc, mimikatz was used. Mimikatz is a malicious tool known for its credential dumping capabilities. Mimikatz was downloaded from GitHub and extracted.



2.2.0 20220919 Djoin parser & Citrix SSO Extractor Latest

gentilkiwi released this Sep 19, 2022 · 13 commits to master since this release 2.2.0-20220919 746e211

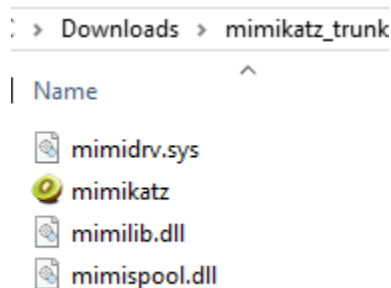
2.2.0-20220919

[legacy] Backport djoin parser & citrix SSO password extractor

▼ Assets 4

| | | |
|----------------------|---------|--------------|
| mimikatz_trunk.7z | 880 KB | Sep 19, 2022 |
| mimikatz_trunk.zip | 1.15 MB | Sep 19, 2022 |
| Source code (zip) | | Sep 19, 2022 |
| Source code (tar.gz) | | Sep 19, 2022 |

28 29 15 61 people reacted



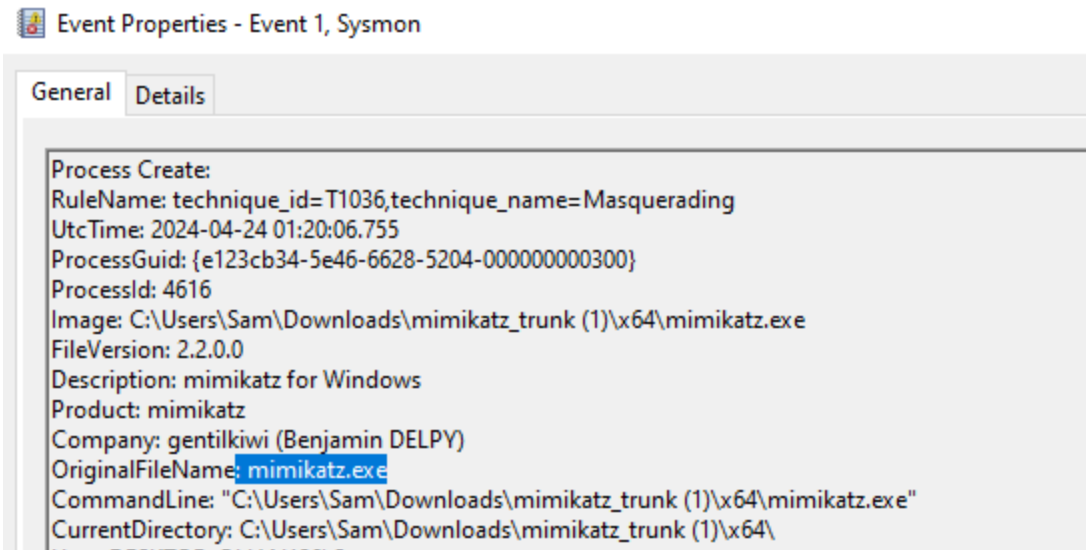
After that, ran powershell in the administrator mode and executed mimikatz.

```
PS C:\Users\Sam\Downloads\mimikatz_trunk (1)\x64> .\mimikatz.exe

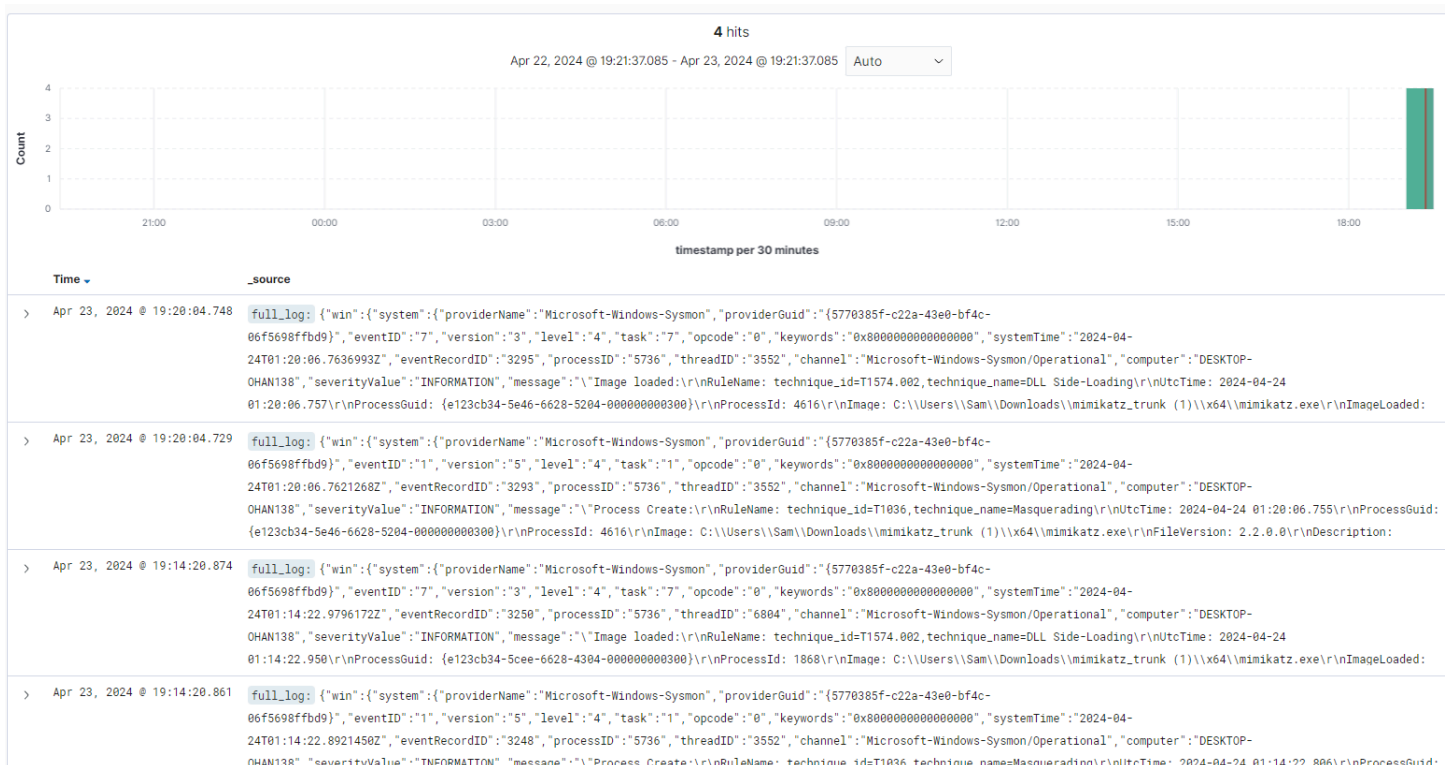
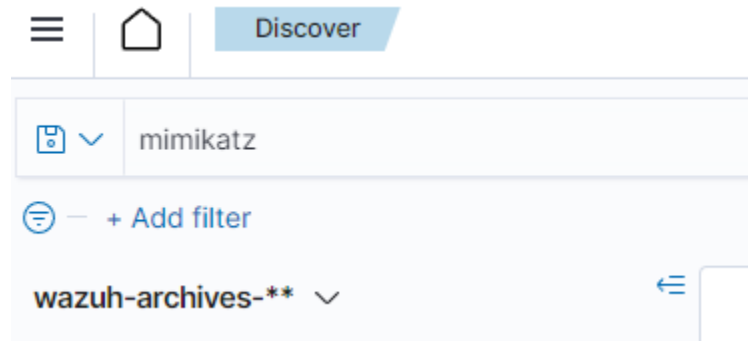
.#####.  mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
.## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
## / \ ##  /**/ Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##   > https://blog.gentilkiwi.com/mimikatz
'## v ##'   Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####'   > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz #
```

Identified that mimikatz was successfully logged in the windows event viewer with the mimikatz.exe as the OriginalFileName field.



Heading back into Wazuh under the newly created index wazuh-archives-*, it was seen that wazuh logged mimikatz.



Apr 23, 2024 @ 19:20:04.748

full_log: { "win": { "system": { "providerName": "Microsoft-Windows-Sysmon", "providerGuid": "{5770385f-c22a-43e0-bf4c-06f5698ffbd9}", "eventId": "7", "version": "3", "level": "4", "task": "7", "opcode": "0", "keywords": "0x8000000000000000", "systemTime": "2024-04-24T01:20:06.7636993Z", "eventRecordID": "3295", "processID": "5736", "threadID": "3552", "channel": "Microsoft-Windows-Sysmon/Operational", "computer": "DESKTOP-OHAN138", "severityValue": "INFORMATION", "message": "\\\"Image loaded:\\\"\\nRuleName: technique_id=T1574.002, technique_name=DLL Side-Loading\\\"\\nUtcTime: 2024-04-24 01:20:06.757\\\"\\nProcessGuid: {e123cb34-5e46-6628-5204-000000000300}\\\"\\nProcessId: 4616\\\"\\nImage: C:\\\\Users\\Sam\\Downloads\\mimikatz_trunk (1)\\\\x64\\mimikatz.exe\\\"\\nImageLoaded:

Expanded document

TableJSON

| | @timestamp | |
|--|-------------------------------------|--|
| | Apr 23, 2024 @ 19:20:04.748 | |
| | _index | wazuh-archives-4.x-2024.04.24 |
| | agent.id | 001 |
| | agent.ip | 192.168.96.132 |
| | agent.name | Ibtesam |
| | data.win.eventdata.company | gentilkiwi (Benjamin DELPY) |
| | data.win.eventdata.description | mimikatz for Windows |
| | data.win.eventdata.fileVersion | 2.2.0.0 |
| | data.win.eventdata.hashes | SHA1=E3B6EA8C46FA831CE6F235ACF48B38A4AE8D69, MD5=29EFD640D3C7FE1E2B02287AD73A18A5, SHA256=61C0810A23580CF492A8BA4F7654566108331E7A4134C968C2D6A05261B2D8A1, IMPHASH=55EE500B84BDFC49F27A98AE456D8EDF |
| | data.win.eventdata.image | C:\\Users\\Sam\\Downloads\\mimikatz_trunk (1)\\\\x64\\mimikatz.exe |
| | data.win.eventdata.imageLoaded | C:\\Users\\Sam\\Downloads\\mimikatz_trunk (1)\\\\x64\\mimikatz.exe |
| | data.win.eventdata.originalFileName | mimikatz.exe |
| | data.win.eventdata.processGuid | {e123cb34-5e46-6628-5204-000000000300} |
| | data.win.eventdata.processId | 4616 |

In addition, a similar field named `win.eventdata.originalFileName` also picked `mimikatz.exe`. This provided the pathway for creating a custom rule that can detect `mimikatz` even if the filename is modified by the adversaries as the original file name will always stay the same.

data.win.eventdata.originalFileName mimikatz.exe

[Manage rules files](#)
[⊕ Add new rules file](#)
[↻ Refresh](#)
[⬇ Export formatted](#)

Accessing add new rules functionality, added a new rule that will detect any original file name matched to mimikatz.exe. Additionally, MITRE id of T1003 was selected which is designated as Credential Dumping.

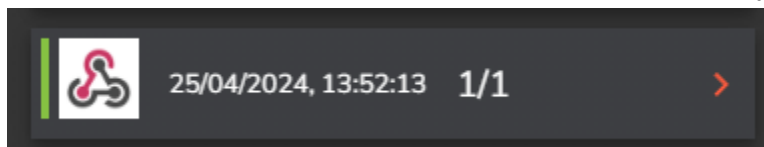
```
<rule id="100002" level="15">
  <if_group>sysmon_event1</if_group>
  <field name="win.eventdata.originalFileName" type="pcr2">(?)mimikatz\.exe</field>
  <description>Mimikatz Detected</description>
  <mitre>
    <id>T1003</id>
  </mitre>
</rule>
```

Executing mimikatz once more from the Windows client showed that the rule is correctly detecting mimikatz.

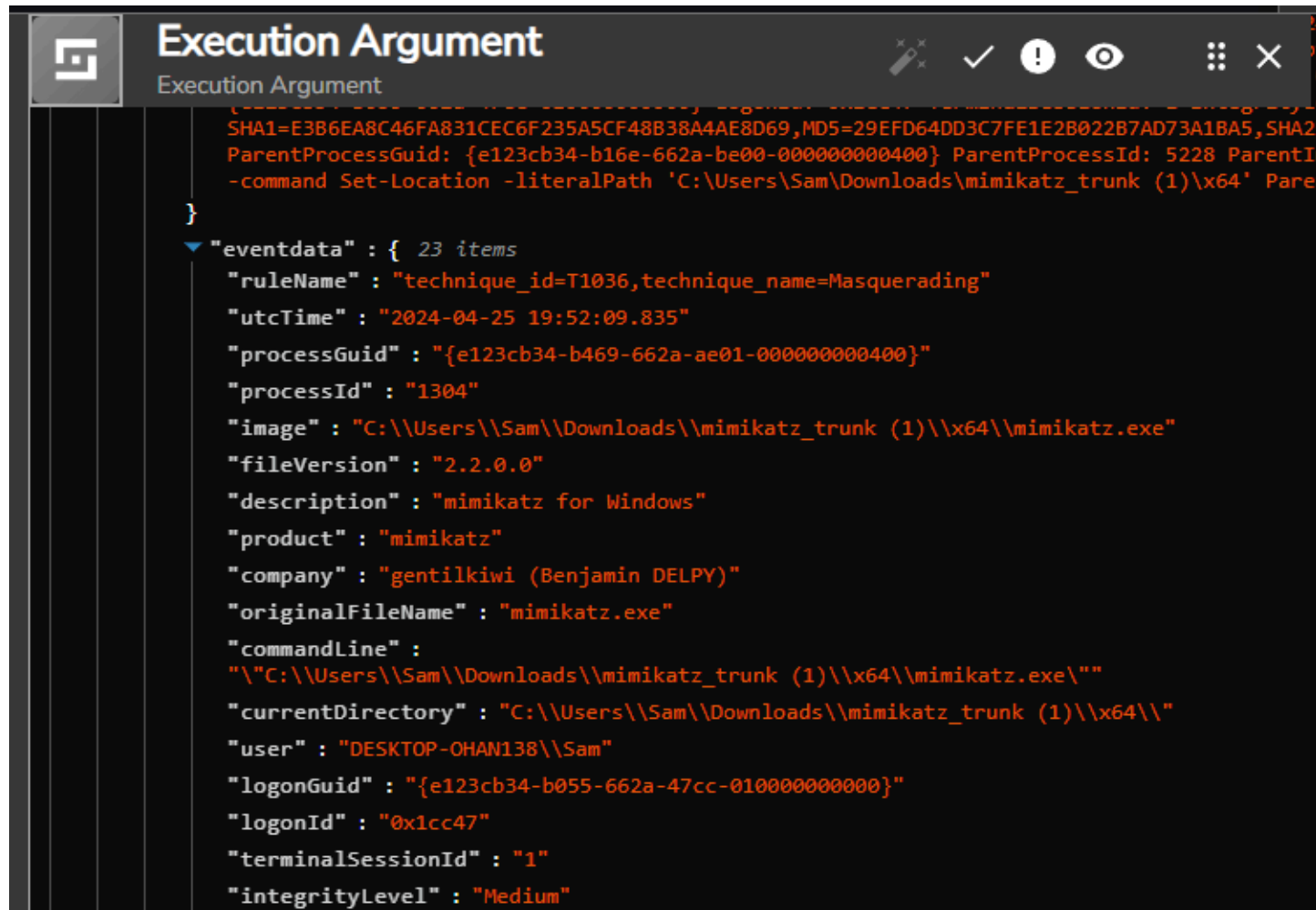
| | | | | | | | | |
|---|-----------------------------|-----|---------|-------|-------------------|-------------------|----|--------|
| > | Apr 25, 2024 @ 13:40:27.180 | 001 | Ibtesam | T1003 | Credential Access | Mimikatz Detected | 15 | 100002 |
|---|-----------------------------|-----|---------|-------|-------------------|-------------------|----|--------|

4.2. Testing Shuffle

Under Shuffle's execution tab, it was seen that the webhook was successfully generated.



And the execution argument showed mimikatz.exe with the originalFileName parameter.



5. AUTOMATION and RESPONSE

With all the components including Wazuh, Shuffle, and TheHive properly configured and tested, it was time for automation workflows. As previously stated this project's goal was to enhance the effectiveness and efficiency of security operations, VirusTotal was used as a threat intelligence medium to enrich IOCs and provide a detailed alert to the analyst inside TheHive console.

5.1. Parsing Hash and Calling VirusTotal API

To use this workflow, created an account with VirusTotal and utilized their API to integrate with Shuffle. From the available applications inside Shuffle searched for VirusTotal. However, the workflow cannot send a full execution argument to the VirusTotal. Using the documentation provided by the VirusTotal, it was found that it accepts file hash as an input. Using Shuffle's tool a "Regex capture group" action was selected to parse the SHA256 Hash from the generated webhook execution argument. This step was necessary as the execution argument provided results in this format "**SHA256Hash=hash**" whereas VirusTotal takes only the hash portion of the result.

The image shows the configuration interface for the 'Regex capture group' tool in Shuffle. The interface is dark-themed with orange highlights. At the top, there is a 'Find Actions' search bar containing the text 'Regex capture group'. Below this is a 'Parameters' section. Under 'Parameters', there is an 'Input data' section with a text field containing '\$exec.text.win.eventdata.hash' and a plus icon. Below the 'Input data' section is a 'Regex' section with a text field containing 'SHA256=([0-9A-Fa-f]{64})' and a plus icon. The entire configuration area is enclosed in a dark box with orange borders around the input fields.

Ran the execution of the workflow one more time and Shuffle tool successfully parsed the hash.

Status SUCCESS

```
▼ "Results for Change Me" : { 3 items
  "success" : true
  ▼ "group_0" : [ 1 item
    0 : "61C0810A23580CF492A6BA4F7654566108331E7A4134C968C2D6A05261B2D8A1"
  ]
  "found" : true
}
```

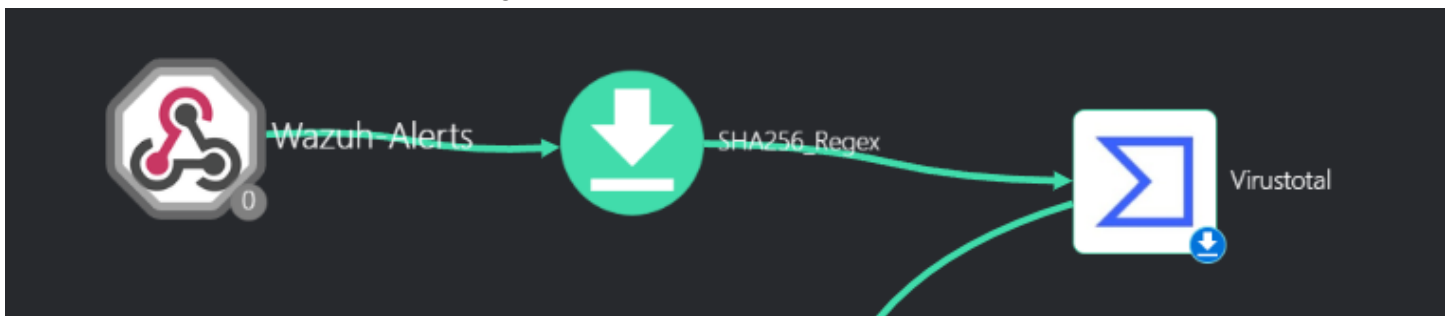
Variables (click to expand)


input_data

regex: SHA256=([0-9A-Fa-f]{64})

shuffle_action_logs

This hash can now be used with the VirusTotal API using the “**Get A Hash Report**” endpoint. The workflow so far and the execution and response is given below.



**Virustotal**
get_a_hash_report_

Status SUCCESS

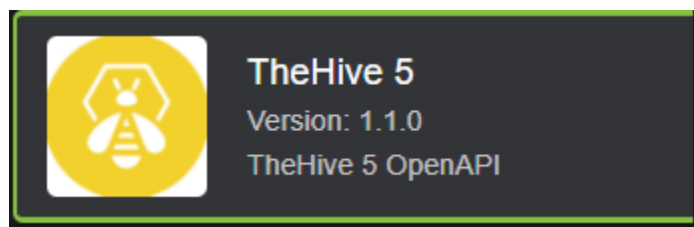
```
▶ "Results for Virustotal" : [...]  
1 item
```

```

  "popular_threat_classification": { 3 items
    "popular_threat_category": [ 3 items
      0: { 2 items
        "value": "hacktool"
        "count": 22
      }
      1: { 2 items
        "value": "trojan"
        "count": 16
      }
      2: { 2 items
        "value": "pua"
        "count": 3
      }
    ]
    "popular_threat_name": [ 3 items
      0: { 2 items
        "value": "mimikatz"
        "count": 31
      }
      1: { ... } 2 items
      2: { ... } 2 items
    ]
    "suggested_threat_label": "hacktool.mimikatz/hacktoolx"
  }
  "sha1": "e3b6ea8c46fa831cec6f235a5cf48b38a4ae8d69"

```

Adding TheHive into the workflow to streamline the whole process. Similar to the VirusTotal, TheHive can also be found by searching for apps inside Shuffle.





Used TheHive's service account API to connect Shuffle with TheHive. Selected "Create Alert" as the intended action.

The screenshot shows the 'TheHive 5' configuration window. At the top, there are icons for back, document, and edit. Below, the 'Name' field is set to 'TheHive_5_1' and the 'Delay' field is set to '0'. The 'Authentication' section shows 'Latest' and '1.1.0' buttons, followed by a dropdown menu labeled 'Auth for Th...' with a plus sign. At the bottom, there is a 'Find Actions' search bar with 'Create alert' selected from the dropdown.

Edited Alert fields as below:

Title:

```
Mimikatz Alert on host $exec.text.win.system.computer
```

```
Mimikatz Alert on host DESKTOP-OHAN138
```

Summary:

```
VirusTotal Suggested Threat Label: $virustotal.#.body.data.attributes.popular_threat_classification  
.suggested_threat_label Malicious labeled by $virustotal.#.body.data.attributes.last_analysis_stats  
.malicious
```

```
VirusTotal Suggested Threat Label: hacktool.mimikatz/hacktoolx Malicious labeled by 63
```

Description:

```
Mimikatz Detected on host DESKTOP-OHAN138
```

Tags: T1003

Status: New

Date and time:

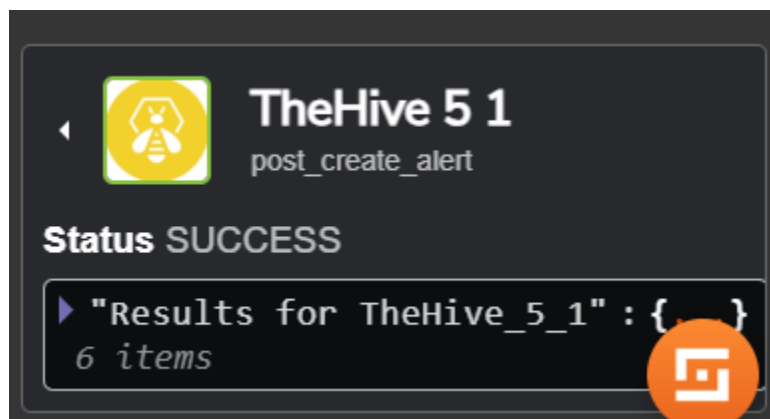
```
$exec.text.win.eventdata.utcTime
```

```
2024-04-25 19:52:09.835
```

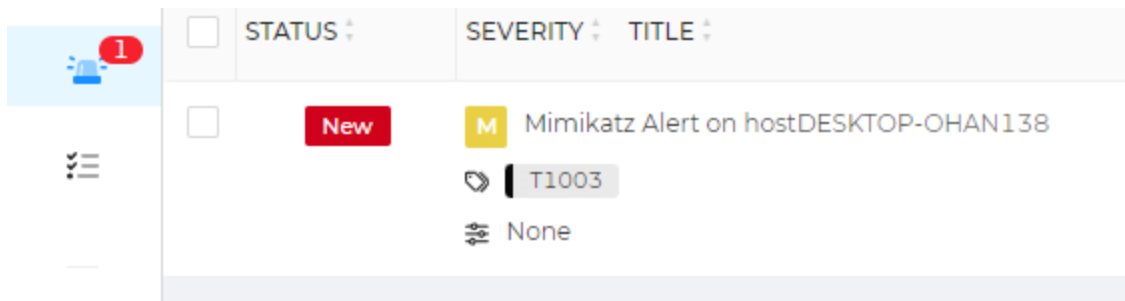
Source: Wazuh

Sourceref: Rule:100002

Finally Shuffle executed the workflow successfully.



Logged-in as the analyst user lbtesam, an alert was seen in the case management section.

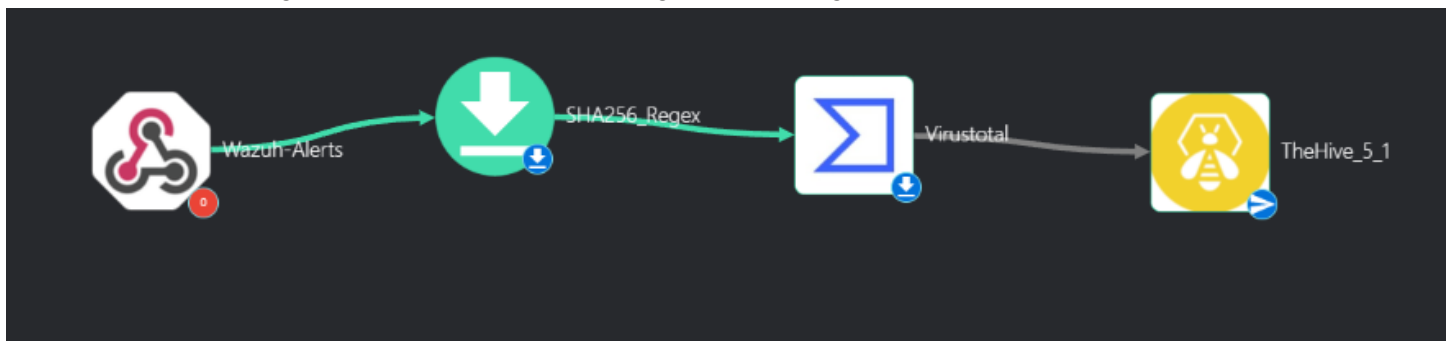


5.2. Workflow Scenarios

Depending on the organization's needs, this SIEM SOAR platform can be customized for hundreds of possible scenarios. With Shuffle's vast application connectors sky is really the limit. Below some possible automation scenarios are listed.

5.2.1. Scenario 1

As demonstrated above, our Windows client sent security events through Sysmon logs which generated a security event inside Wazuh (SIEM) which then triggered Shuffle (SOAR) to create an alert in TheHive (Case Management) while also performing threat intelligence on the event.



5.2.2. Scenario 2

In this scenario, instead of alerting analysts through TheHive, we can use an active response command using the Wazuh application. So, the scenario is, someone scanning or pinging or even trying to SSH into our Windows client and we want to block the IP at the firewall. To add a custom active response command, Wazuh needs to be configured like below.

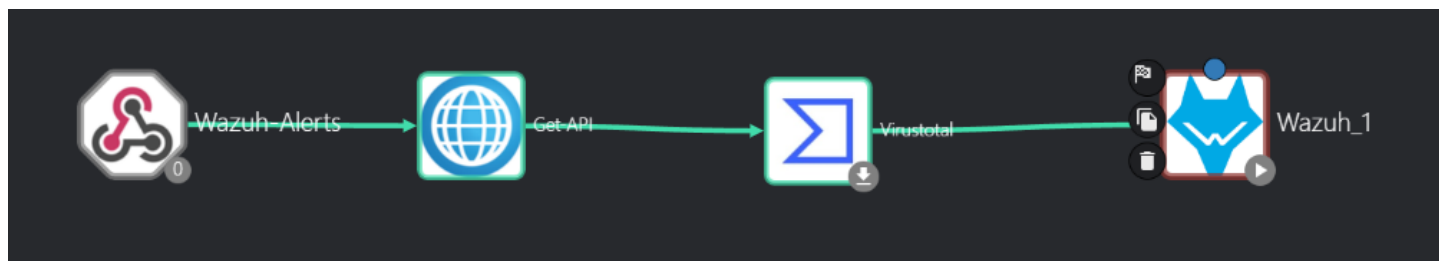
```

<active-response>
  <command>firewall-drop</command>
  <location>local</location>
  <level>5</level>
  <timeout>no</timeout>
</active-response>
  
```

And the actual command to be used is: firewall-drop0 as shown below

```
root@Wazuh:/var/ossec/bin# ./agent_control -L  
  
Wazuh agent_control. Available active responses:  
  
Response name: firewall-drop0, command: firewall-drop
```

In Shuffle, we need to use the Wazuh API to execute the command. The workflow looks like this:

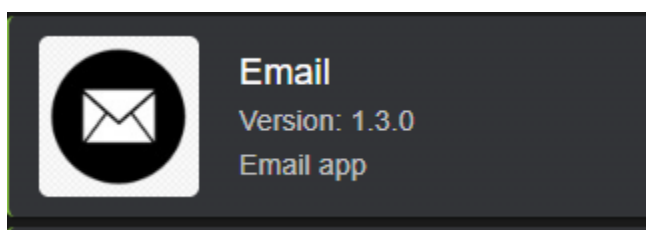


651 Host Blocked by firewall-drop Active Response

5.2.3. Other App Recommendations

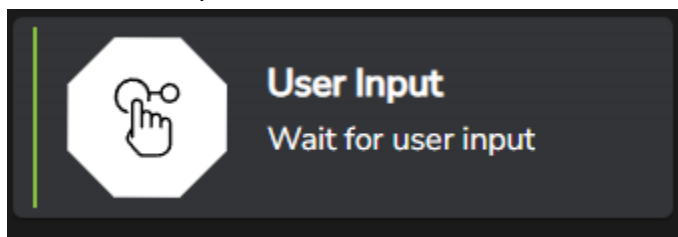
These are some recommended app for SOC Automation workflows other than demonstrated above. These includes:

Email: This allows sending emails.



User Input:

This allows anyone selected through email or text message to take any specified action. For example, an email can be sent out asking the analyst to block an IP and based on the response it can forward the decision to Wazuh to block that IP using the firewall-drop0 command.



User Input
What is the user input trigger?

Name
User_Input_1

Information
The information you want to show the user.
Supports variables.
Do you want to block the IP?

Input options
Use subflows to connect to any app you want, or
use the default email and sms options
☐ Subflow ☒ Email ☐ SMS
Email *
test@test.com

6. CONCLUSION

In conclusion, the "SOC Automation with SIEM & SOAR" project has been a resounding success, demonstrating the tangible benefits of implementing advanced security automation solutions. By leveraging SIEM and SOAR technologies, the project has significantly enhanced threat detection capabilities, streamlined incident response processes, and empowered security analysts with enriched contextual information. The project's impact extends beyond the confines of the simulated environment, serving as a testament to the transformative power of automation in modern cybersecurity operations. As organizations continue to face increasingly sophisticated cyber threats, the lessons learned from this project will undoubtedly shape the future of security operations and pave the way for more agile and resilient security frameworks.