

SOC AUTOMATION PROJECT

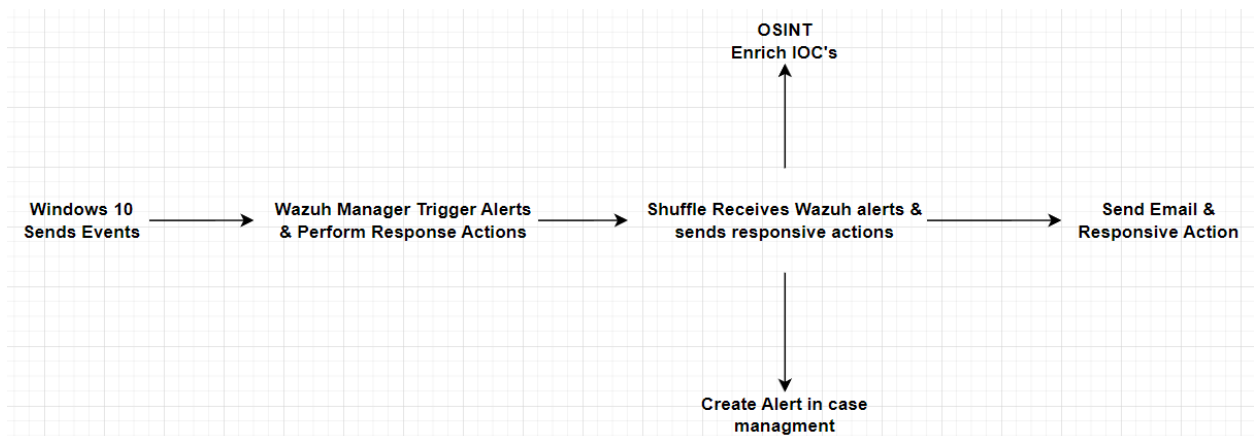
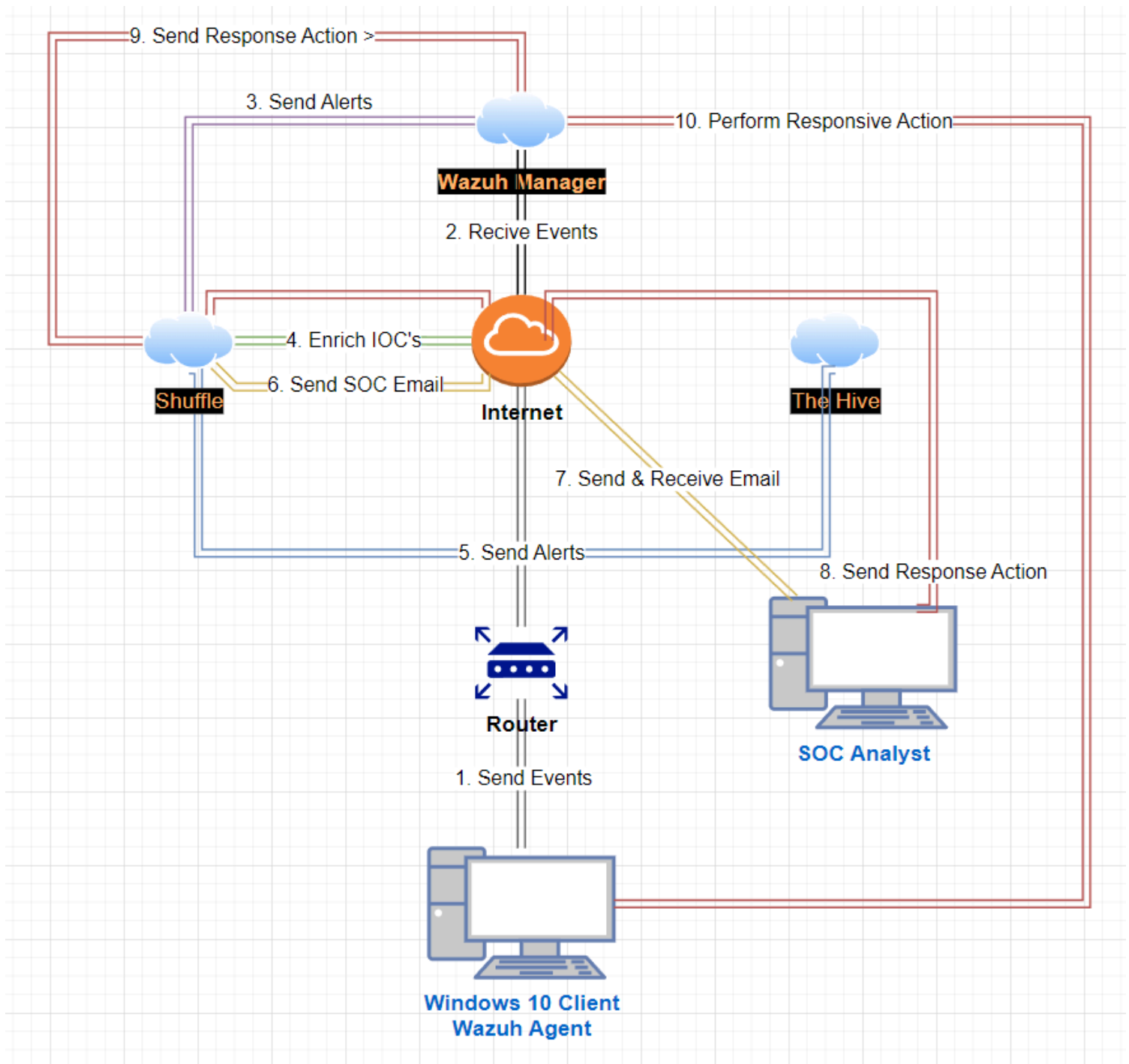
Alexander Chait

wazuh.

Shuffle

TheHive

Project Diagram



By the end of this SOC Automation Project, I'm going to have a fully functional live telemetry ingesting/digesting lab while implementing automation & orchestration

I'll be using:

- Wazuh (SIEM/XDR)
- Shuffle (SOAR)
- The Hive (CaseManagmentSystem)

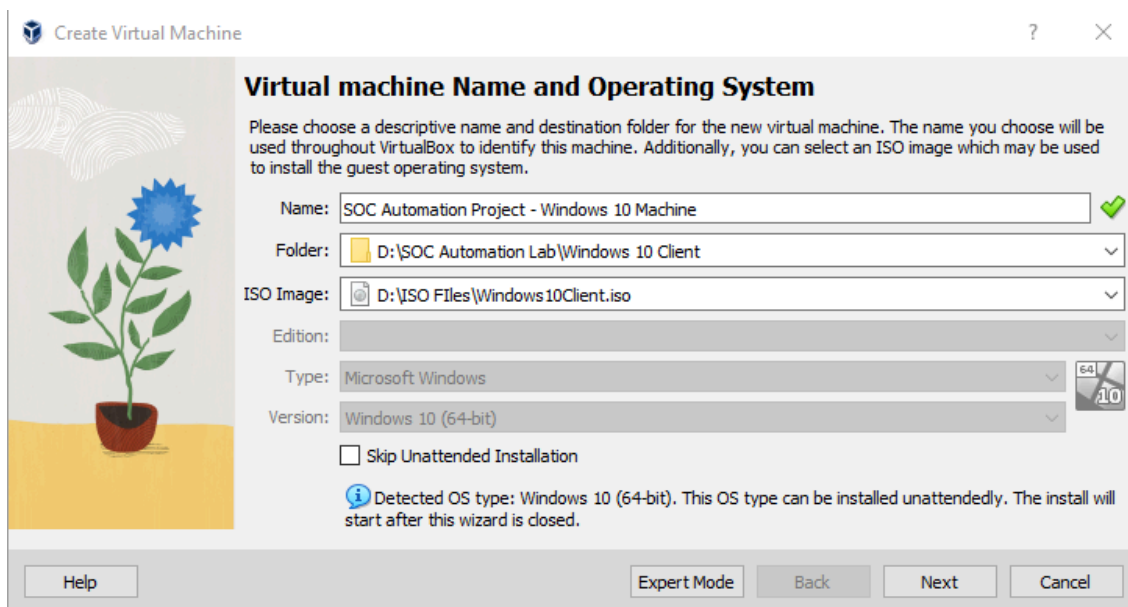
While sending alerts in real time to the SOC analyst via email

I'll also do live testing and creating custom Wazuh rules using the [Mimikatz](#) penetration testing tool & Investigating Sysmon logs

Part 1: Setting up VMs

- Windows 10 w/Sysmon
- Wazuh Server (SIEM/XDR)
- TheHive Server (IR Platform)

I'll start with the Windows 10 Client w/Sysmon using VirtualBox, this will act as the Wazuh Agent



Create Virtual Machine

Virtual machine Name and Operating System

Please choose a descriptive name and destination folder for the new virtual machine. The name you choose will be used throughout VirtualBox to identify this machine. Additionally, you can select an ISO image which may be used to install the guest operating system.

Name: SOC Automation Project - Windows 10 Machine ✓

Folder: D:\SOC Automation Lab\Windows 10 Client

ISO Image: D:\ISO Files\Windows10Client.iso

Edition:

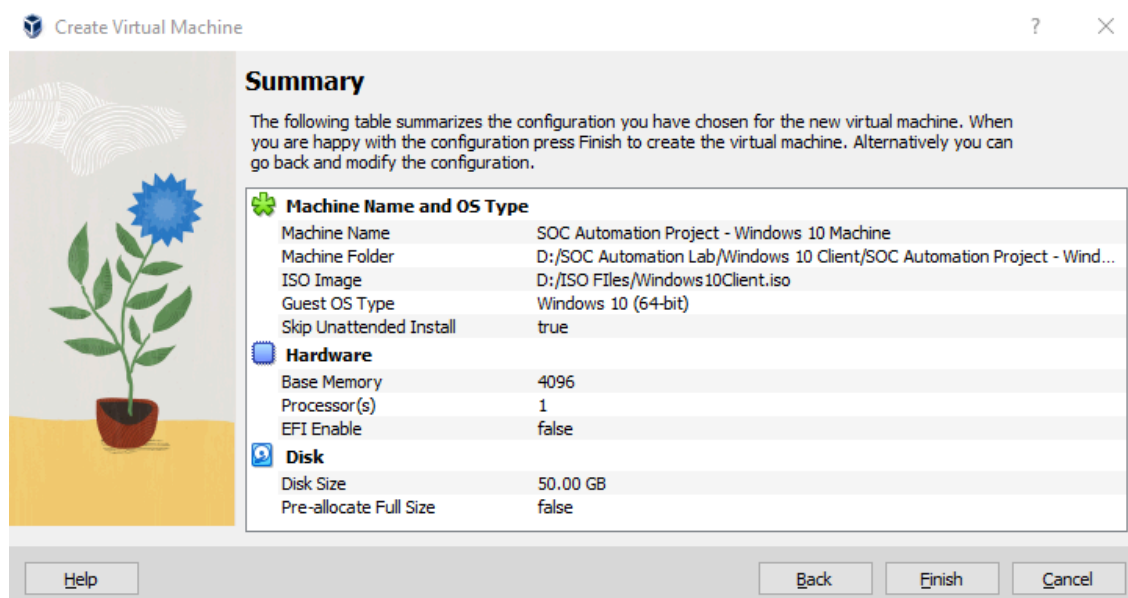
Type: Microsoft Windows

Version: Windows 10 (64-bit)

☐ Skip Unattended Installation

i Detected OS type: Windows 10 (64-bit). This OS type can be installed unattended. The install will start after this wizard is closed.

Help Expert Mode Back Next Cancel



Create Virtual Machine

Summary

The following table summarizes the configuration you have chosen for the new virtual machine. When you are happy with the configuration press Finish to create the virtual machine. Alternatively you can go back and modify the configuration.

| Machine Name and OS Type | |
|--------------------------|--|
| Machine Name | SOC Automation Project - Windows 10 Machine |
| Machine Folder | D:\SOC Automation Lab\Windows 10 Client\SOC Automation Project - Wind... |
| ISO Image | D:\ISO Files\Windows10Client.iso |
| Guest OS Type | Windows 10 (64-bit) |
| Skip Unattended Install | true |

| Hardware | |
|--------------|-------|
| Base Memory | 4096 |
| Processor(s) | 1 |
| EFI Enable | false |

| Disk | |
|------------------------|----------|
| Disk Size | 50.00 GB |
| Pre-allocate Full Size | false |

Help Back Finish Cancel

I'll be using the Olaf Hartong Sysmon Modular config file from

<https://github.com/olafhartong/sysmon-modular/blob/master/sysmonconfig.xml>



by Olaf Hartong

Running PowerShell with admin privileges to install

```
PS C:\Users\WazuhAgent\Downloads\Sysmon> dir

Directory: C:\Users\WazuhAgent\Downloads\Sysmon

Mode                LastWriteTime         Length Name
----                -
-a----            8/17/2024   1:02 AM             7490 Eula.txt
-a----            8/17/2024   1:02 AM          8480560 Sysmon.exe
-a----            8/17/2024   1:02 AM          4563248 Sysmon64.exe
-a----            8/17/2024   1:02 AM          4993440 Sysmon64a.exe
-a----            8/17/2024   1:06 AM          253169 sysmonconfig.xml

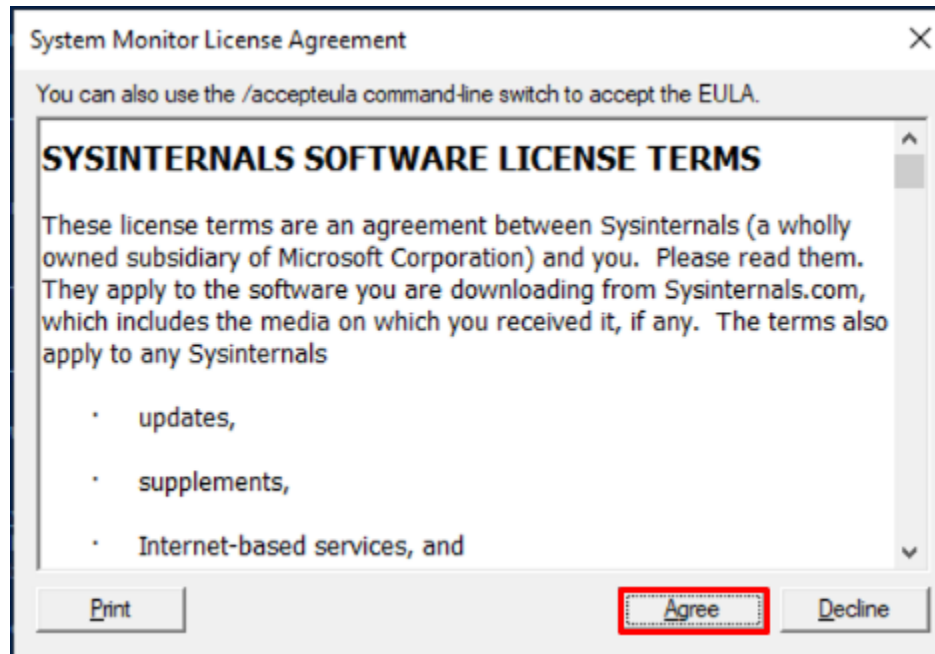
PS C:\Users\WazuhAgent\Downloads\Sysmon> .\Sysmon64.exe

System Monitor v15.15 - 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.
Sysinternals - www.sysinternals.com

Usage:
Install:                Sysmon64.exe -i [<configfile>]
Update configuration:   Sysmon64.exe -c [<configfile>]
Install event manifest: Sysmon64.exe -m
Print schema:           Sysmon64.exe -s
Uninstall:              Sysmon64.exe -u [force]
```

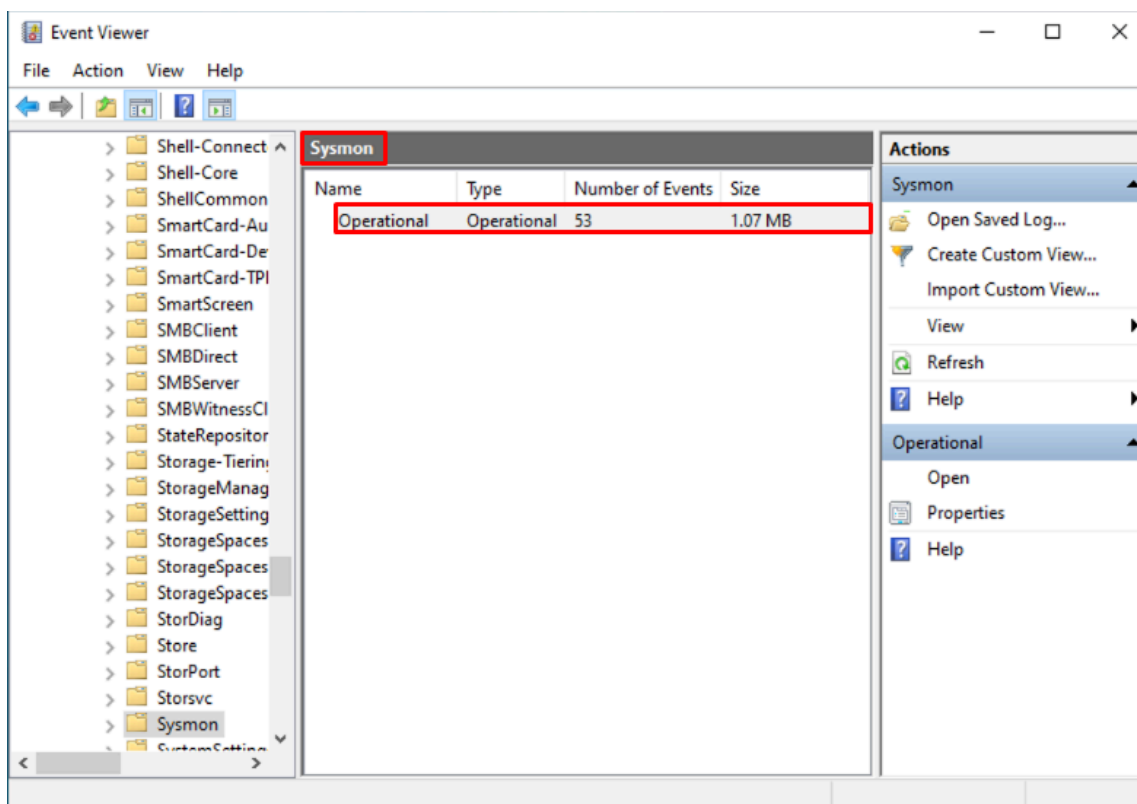
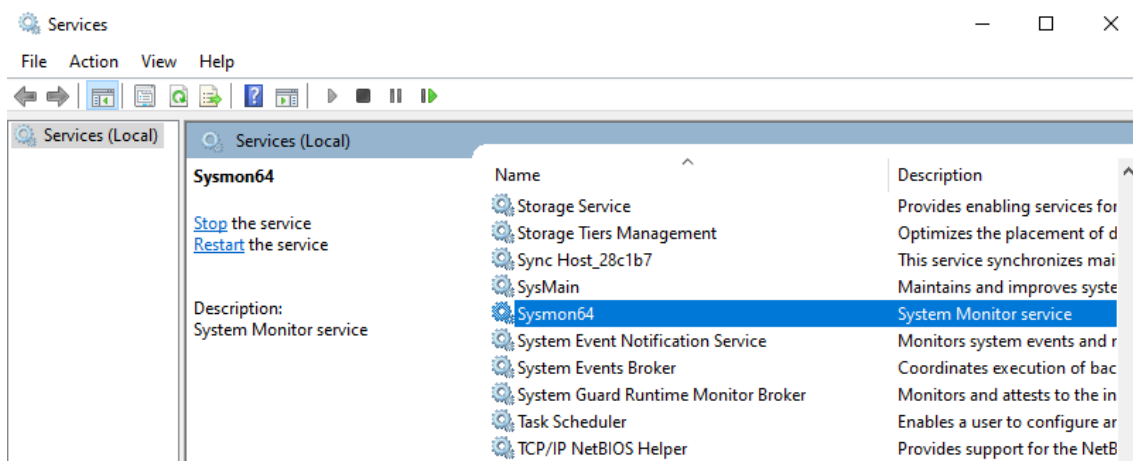

Making sure the right configurations are installed

```
PS C:\Users\WazuhAgent\Downloads\Sysmon> .\Sysmon64.exe -i .\symonconfig.xml
```



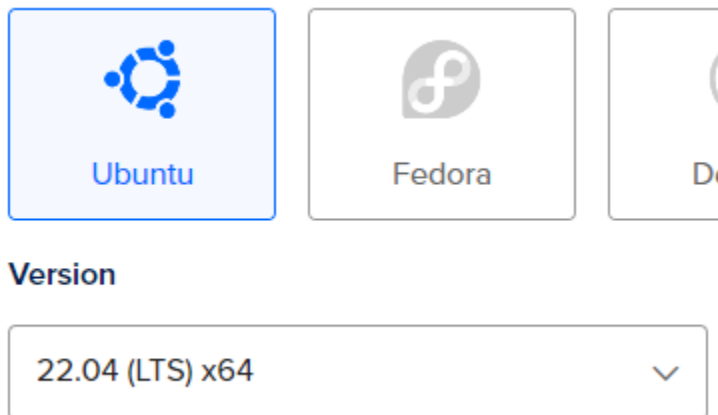
```
Loading configuration file with schema version 4.90
Configuration file validated.
Sysmon64 installed.
SysmonDrv installed.
Starting SysmonDrv.
SysmonDrv started.
Starting Sysmon64..
Sysmon64 started.
PS C:\Users\WazuhAgent\Downloads\Sysmon>
```

I'll make sure that Sysmon is up and running, by checking EventViewer & Services



Now that I have my Windows Client machine installed along with Sysmon, I'll start installing the Wazuh Server

I'll be using Digital Ocean, which is a cloud infrastructure provider, I'll use Ubuntu 22.04



I'll configure the firewall first, I'll be blocking all inbound TCP/UDP traffic except for my personal Lab network


| | | |
|------|-----------------------------|---|
| Name | SOC-Automation-Lab-Firewall | ✓ |
|------|-----------------------------|---|

Inbound Rules

Set the Firewall rules for incoming traffic. Only the specified ports will accept inbound connections. All other traffic

| Type | Protocol | Port Range | Sources |
|---------|----------|------------|---------|
| All TCP | TCP | All ports | |
| All UDP | UDP | All ports | |
| ICMP | ICMP | | |


I'll link the firewall rules to the Wazuh-Server

**SOC-Automation-Lab-Firewall**
6 Rules / 1 Droplet

Rules **Droplets** Destroy

[Learn](#)

Add Droplets

| Name | IP Address | State | Added |
|---|------------|------------|-------------------------------|
|  Wazuh-Server 8 GB / 2 Intel vCPUs / 160 GB / FRA1 | | Up-to-date | Just now More |

I'll use [PuTTY](#) for my SSH to IaaS Cloud connection

```
root@Wazuh-Server: ~  
login as: root  
root@ [REDACTED] password:  
Access denied  
root@ [REDACTED] password:  
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-113-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/pro  
  
System information as of Sat Aug 17 10:09:58 UTC 2024  
  
System load:  0.0                Processes:            101  
Usage of /:   1.1% of 154.88GB   Users logged in:     0  
Memory usage: 2%                IPv4 address for eth0: [REDACTED]  
Swap usage:   0%                IPv4 address for eth0: [REDACTED]  
  
Expanded Security Maintenance for Applications is not enabled.  
  
39 updates can be applied immediately.  
29 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status
```

I'll start by updating and upgrading

```
root@Wazuh-Server:~# dir  
snap  
root@Wazuh-Server:~# apt-get update && apt-get upgrade -y  
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]  
Hit:2 http://mirrors.digitalocean.com/ubuntu jammy InRelease  
Hit:3 https://repos-droplet.digitalocean.com/apt/droplet-agent main InRelease  
Hit:4 http://mirrors.digitalocean.com/ubuntu jammy-updates InRelease  
Hit:5 http://mirrors.digitalocean.com/ubuntu jammy-backports InRelease  
Fetched 129 kB in 4s (36.2 kB/s)  
Reading package lists... Done  
Reading package lists... Done  
Building dependency tree... Done
```

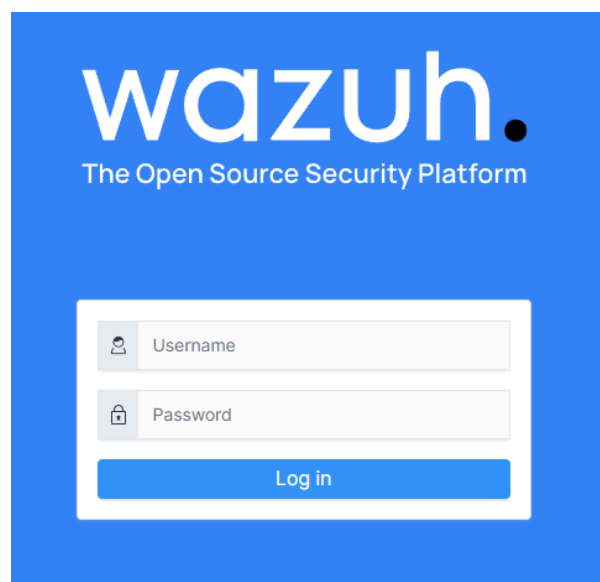
Using this curl command, I'll install Wazuh 4.7

```
root@Wazuh-Server:~#  
root@Wazuh-Server:~#  
root@Wazuh-Server:~#  
root@Wazuh-Server:~# curl -sO https://packages.wazuh.com/4.7/wazuh-install.sh &&  
sudo bash ./wazuh-install.sh -a  
17/08/2024 10:34:10 INFO: Starting Wazuh installation assistant. Wazuh version:  
4.7.5  
17/08/2024 10:34:10 INFO: Verbose logging redirected to /var/log/wazuh-install.l
```

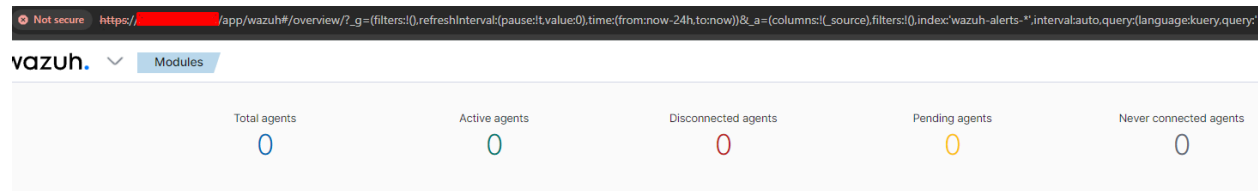
Post installation, I've received a Username & Password for the Wazuh dashboard

```
17/08/2024 10:38:49 INFO: Initializing Wazuh dashboard web application.  
17/08/2024 10:38:49 INFO: Wazuh dashboard web application initialized.  
17/08/2024 10:38:49 INFO: --- Summary ---  
17/08/2024 10:38:49 INFO: You can access the web interface https://<wazuh-dashboard-ip>:443  
User:   
Password:   
17/08/2024 10:38:49 INFO: Installation finished  
root@Wazuh-Server:~#
```

Now I can log in into the dashboard of my Wazuh Server





After entering the credentials I was provided, I'm able to use the Wazuh dashboard using the public IP address of my Wazuh server



Next I'll repeat the same process for TheHive Server, and I will apply the firewall rules to it

The screenshot shows the 'SOC-Automation-Lab-Firewall' interface. At the top, there's a header with the title and '6 Rules / 2 Droplets'. Below this, there are three tabs: 'Rules', 'Droplets' (which is selected), and 'Destroy'. On the right side, there's a 'Learn' link and an 'Add Droplets' button. The main content area displays a table of droplets.

| Name | IP Address | State | Added |
|---|---------------|------------|-----------------------------------|
|  Wazuh-Server 8 GB / 2 Intel vCPUs / 160 GB / FRA1 | [Redacted IP] | Up-to-date | 1 hour ago More ▾ |
|  TheHive-Server 8 GB / 2 Intel vCPUs / 160 GB / FRA1 | [Redacted IP] | Up-to-date | Just now More ▾ |

Before installing the TheHive I'll be installing needed dependencies, I'll also be installing Java/Cassandra/ElasticSearch as prerequisites

Installing Dependencies: CommandLine:

```
8 Dependencies
9 apt install wget gnupg apt-transport-https git ca-certificates ca-certificates-java curl software-properties-common python3-pip lsb-release
```

```
root@TheHive-Server:~#
root@TheHive-Server:~# apt install wget gnupg apt-transport-https git ca-certificates ca-certificates-java curl software-properties-common python3-pip lsb-release
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
ca-certificates is already the newest version (20230311ubuntu0.22.04.1).
git is already the newest version (1:2.34.1-1ubuntu1.11).
```

Installing Java CommandLine:

```
12 wget -qO- https://apt.corretto.aws/corretto.key | sudo gpg --dearmor -o /usr/share/keyrings/corretto.gpg
13 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
14 sudo apt update
15 sudo apt install java-common java-11-amazon-corretto-jdk
16 echo JAVA_HOME="/usr/lib/jvm/java-11-amazon-corretto" | sudo tee -a /etc/environment
17 export JAVA_HOME="/usr/lib/jvm/java-11-amazon-corretto"
```

```
root@TheHive-Server:~#
root@TheHive-Server:~#
root@TheHive-Server:~# 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"
```

Installing Casandra

CommandLine:

```
20  wget -qO - https://downloads.apache.org/cassandra/KEYS | sudo gpg --dearmor -o /usr/share/keyrings/cassandra-archive.gpg
21  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
22  sudo apt update
23  sudo apt install cassandra
```

```
After this operation, 57.4 MB of additional disk space will be used.
Get:1 https://apache.jfrog.io/artifactory/cassandra-deb 40x/main amd64 cassandra all 4.0.13 [46.6 MB]
Fetched 46.6 MB in 7s (6585 kB/s)
Selecting previously unselected package cassandra.
(Reading database ... 72736 files and directories currently installed.)
Preparing to unpack .../cassandra_4.0.13_all.deb ...
Unpacking cassandra (4.0.13) ...
Setting up cassandra (4.0.13) ...
Adding group `cassandra' (GID 121) ...
```

Installing ElasticSearch

CommandLine:

```
26  wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o /usr/share/keyrings/elasticsearch-keyring.gpg
27  sudo apt-get install apt-transport-https
28  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
29  sudo apt update
30  sudo apt install elasticsearch
```

```
(Reading database ... 72905 files and directories currently installed.)
Preparing to unpack .../elasticsearch_7.17.23_amd64.deb ...
Creating elasticsearch group... OK
Creating elasticsearch user... OK
Unpacking elasticsearch (7.17.23) ...
Setting up elasticsearch (7.17.23) ...
```


Installing TheHive

CommandLine:

```
39 wget -O- https://archives.strangebee.com/keys/strangebee.gpg | sudo gpg --dearmor -o /usr/share/keyrings/strangebee-archive-keyring.gpg
40 echo 'deb [signed-by=/usr/share/keyrings/strangebee-archive-keyring.gpg] https://deb.strangebee.com thehive-5.2 main' | sudo tee -a /etc/apt/sources.list.d/strangebee.list
41 sudo apt-get update
42 sudo apt-get install -y thehive
```

```
root@TheHive-Server:~# wget -O- https://archives.strangebee.com/keys/strangebee.gpg | sudo gpg --dearmor -o /usr/share/keyrings/strangebee-archive-keyring.gpg
echo 'deb [signed-by=/usr/share/keyrings/strangebee-archive-keyring.gpg] https://deb.strangebee.com thehive-5.2 main' | sudo tee -a /etc/apt/sources.list.d/strangebee.list
sudo apt-get update
sudo apt-get install -y thehive
--2024-08-17 13:14:21-- https://archives.strangebee.com/keys/strangebee.gpg
Resolving archives.strangebee.com (archives.strangebee.com)... 5.196.134.251
Connecting to archives.strangebee.com (archives.strangebee.com)|5.196.134.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3179 (3.1K) [text/plain]
Saving to: 'STDOUT'

-                               100%[=====>]    3.10K  --.-KB/s    in 0s
```

Part 2: Configurations

- Configure TheHive Server
- Configure The Wazuh Server
- Configure Windows 10 Reporting to Wazuh Server

I'll begin with configuring the Cassandra.yaml File

Using the nano command

```
root@TheHive-Server:~# nano /etc/cassandra/
cassandra-env.sh      cassandra.yaml      hotspot_compiler
cassandra-rackdc.properties  commitlog_archiving.properties  jvm-clients.options
cassandra-topology.properties  cqlshrc.sample      jvm-server.options
root@TheHive-Server:~# nano /etc/cassandra/cassandra.yaml
GNU nano 6.2
# Cassandra storage config YAML
```

I'll change the Cluster name to "SocAutoLab"

```
# The name of the cluster. This is mainly used to prevent machines in
# one logical cluster from joining another.
cluster_name: 'SocAutoLab'
```

And I'll change the Listen Address to the public IP address of TheHive Server

```
# Setting listen_address to 0.0.0.0 is always wrong.
#
listen_address: 1.
# Set listen_address OR listen interface, not both. Interfaces must correspond
# to a single address, IP aliasing is not supported.
# listen_interface: eth0
```

Change the RPC address to TheHive Public IP address

```
# For security reasons, you should not expose this port to the internet. Firewall it if needed.
rpc_address: [REDACTED]

# Set rpc_address OR rpc_interface, not both. Interfaces must correspond
# to a single address, IP aliasing is not supported.
# rpc_interface: eth1
```

I'll do the same with the Seed Provider

```
seed_provider:
  # Addresses of hosts that are deemed contact points.
  # Cassandra nodes use this list of hosts to find each other and learn
  # the topology of the ring. You must change this if you are running
  # multiple nodes!
  - class_name: org.apache.cassandra.locator.SimpleSeedProvider
    parameters:
      # seeds is actually a comma-delimited list of addresses.
      # Ex: "<ip1>,<ip2>,<ip3>"
      - seeds: "64.227.119.244,7000"
```

Next I'll stop Cassandra temporarily in order to remove old files

```
root@TheHive-Server:~#
root@TheHive-Server:~# systemctl stop cassandra.service
```

Removing the old files in the var/lib/cas dir

```
root@TheHive-Server:~# rm -rf /var/lib/cassandra/*
root@TheHive-Server:~#
```

Start Cassandra again after removing old files

```
root@TheHive-Server:~#
root@TheHive-Server:~# systemctl start cassandra.service
root@TheHive-Server:~#
root@TheHive-Server:~#
```

Checking if its running, and it is

```
root@TheHive-Server:~# systemctl status cassandra.service
● cassandra.service - LSB: distributed storage system for structured data
   Loaded: loaded (/etc/init.d/cassandra; generated)
   Active: active (running) since Sat 2024-08-17 13:43:37 UTC; 1min 3s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 16632 ExecStart=/etc/init.d/cassandra start (code=exited, status=0/SUCCESS)
    Tasks: 55 (limit: 9478)
   Memory: 2.2G
      CPU: 14.828s
   CGroup: /system.slice/cassandra.service
           └─16735 /usr/bin/java -ea -da:net.openhft... -XX:+UseThreadPriorities -XX:+HeapDumpOnOutOfMemor

Aug 17 13:43:37 TheHive-Server systemd[1]: Starting LSB: distributed storage system for structured data...
Aug 17 13:43:37 TheHive-Server systemd[1]: Started LSB: distributed storage system for structured data.
lines 1-13/13 (END)
```

Now that Cassandra is up & running and configured, I'll setup Elasticsearch which is used to manage querying data

```
root@TheHive-Server: ~
root@TheHive-Server:~# nano /etc/elasticsearch/elasticsearch.yml
```

Activating & Changing Cluster.Name and Node.Name
And to the public IP for the HiveServer

```
# Use a descriptive name for your cluster:
#
cluster.name: SocAutoLab
#
# ----- Node -----
#
# Use a descriptive name for the node:
#
node.name: node-1
#
# Add custom attributes to the node:
#
#node.attr.rack: r1
#
# ----- Paths -----
```

```
network.host: 64.227.119.244
#
# By default Elasticsearch li
# finds starting at 9200. Set
#
http.port: 9200
```

Cluster Node will be set to 1 node, but this can be scaled but adding more nodes

```
# Bootstrap the cluster using an initial set
#
cluster.initial_master_nodes: ["node-1"]
#
```

Now I'll start and Enable ElasticSearch

```
root@TheHive-Server:~# systemctl start elasticsearch
root@TheHive-Server:~# systemctl enable elasticsearch
Synchronizing state of elasticsearch.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable elasticsearch
Created symlink /etc/systemd/system/multi-user.target.wants/elasticsearch.service → /lib/systemd/system/elasticsearch.service.
root@TheHive-Server:~#
```

Status is Running

```
root@TheHive-Server:~# systemctl status elasticsearch.service
● elasticsearch.service - Elasticsearch
   Loaded: loaded (/lib/systemd/system/elasticsearch.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2024-08-17 13:58:28 UTC; 1min 35s ago
     Docs: https://www.elastic.co
  Main PID: 17718 (java)
    Tasks: 57 (limit: 9478)
   Memory: 4.3G
      CPU: 57.745s
   CGroup: /system.slice/elasticsearch.service
           └─17718 /usr/share/elasticsearch/jdk/bin/java -Xshare:auto -Des.networkaddress.cache.ttl=60 -Des.networkaddress.cache.negative
              17905 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/bin/controller

Aug 17 13:58:04 TheHive-Server systemd[1]: Starting Elasticsearch...
Aug 17 13:58:11 TheHive-Server systemd-entrypoint[17718]: Aug 17, 2024 1:58:11 PM sun.util.locale.provider.LocaleProviderAdapter <clinit>
Aug 17 13:58:11 TheHive-Server systemd-entrypoint[17718]: WARNING: COMPAT locale provider will be removed in a future release
Aug 17 13:58:28 TheHive-Server systemd[1]: Started Elasticsearch.
```

The Hive Configurations:

```
root@TheHive-Server:~# ls -la /opt/thp
total 12
drwxr-xr-x 3 root root 4096 Aug 17 13:14 .
drwxr-xr-x 5 root root 4096 Aug 17 13:14 ..
drwxr-xr-x 5 root root 4096 Aug 17 13:14 thehive
root@TheHive-Server:~#
```

Changing the hive root directory

```
root@TheHive-Server:~# chown -R thehive:thethehive /opt/thp
root@TheHive-Server:~# ls -la /opt/thp
total 12
drwxr-xr-x 3 thehive thehive 4096 Aug 17 13:14 .
drwxr-xr-x 5 root root 4096 Aug 17 13:14 ..
drwxr-xr-x 5 thehive thehive 4096 Aug 17 13:14 thehive
root@TheHive-Server:~#
```

Now it's TheHive user & TheHive group

Next I'll configure TheHive .conf file

```
root@TheHive-Server:~# nano /etc/thehive/application.conf
GNU nano 6.2
TheHive configuration - application.conf
#
```

Change ClusterName + IP Addresses

```
db.janusgraph {
  storage {
    backend = cql
    hostname = ["64.227.119.244"]
    # Cassandra authentication (if configured)
    # username = "thethehive"
    # password = "password"
    cql {
      cluster-name = SocAutoLab
      keyspace = thehive
    }
  }
}

index.search {
  backend = elasticsearch
  hostname = ["64.227.119.244"]
  index-name = thehive
}
```


For the service as well

```
# Service configuration
application.baseUrl = "http://64.227.119.244:9000"
play.http.context = "/"
```

Cortex provides response capabilities

Misp Used as a CTI (CyberThreatIntelligence) Plat

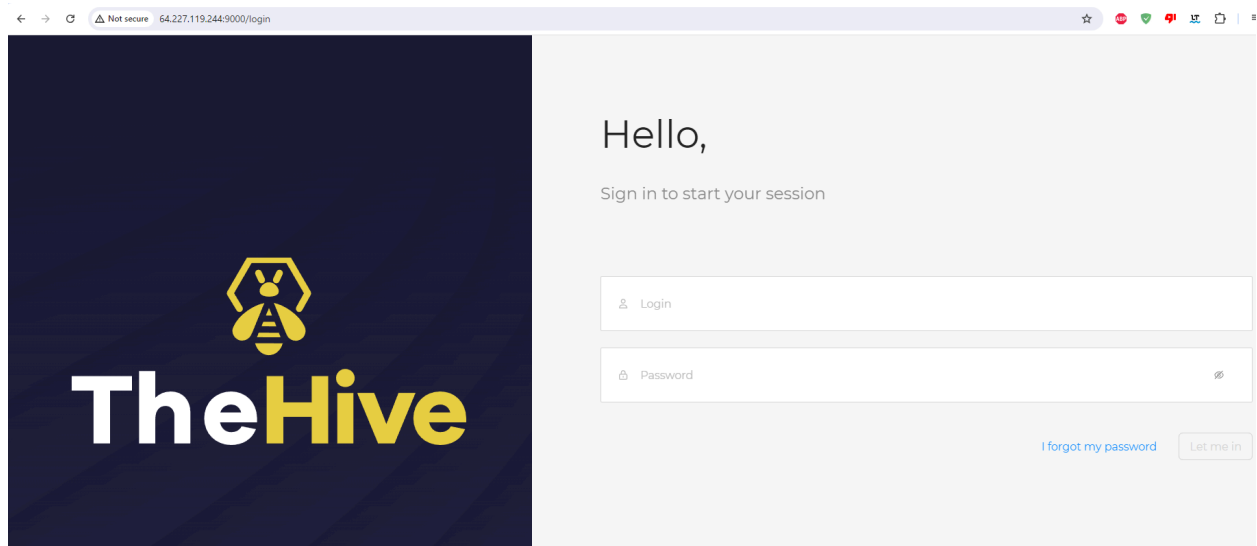
```
# Both modules are enabled by default. If not used, each one can be disabled by
# commenting the configuration line.
scalligraph.modules += org.thp.thehive.connector.cortex.CortexModule
scalligraph.modules += org.thp.thehive.connector.misp.MispModule
```

TheHive is configured and up and running

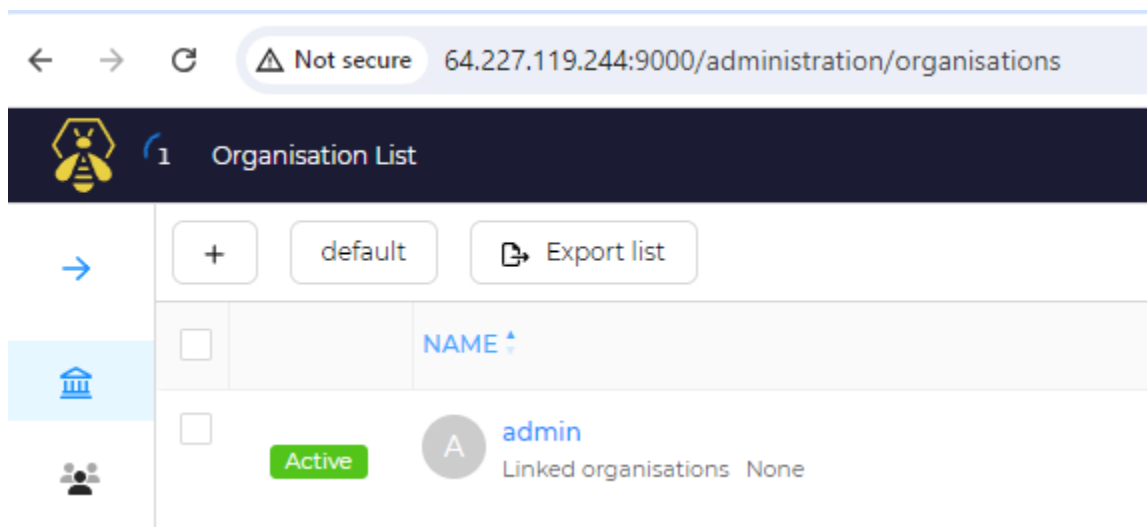
```
root@TheHive-Server:~# systemctl start thehive
root@TheHive-Server:~# systemctl enable thehive
Created symlink /etc/systemd/system/multi-user.target.wants/thehive.service - /lib/systemd/system/thehive.service.
root@TheHive-Server:~# systemctl status thehive
● thehive.service - Scalable, Open Source and Free Security Incident Response Solutions
   Loaded: loaded (/lib/systemd/system/thehive.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2024-08-17 14:16:27 UTC; 32s ago
     Docs: https://thehive-project.org
  Main PID: 19170 (java)
    Tasks: 61 (limit: 9478)
  Memory: 568.2M
     CPU: 33.851s
   CGroup: /system.slice/thehive.service
           └─19170 java -Dfile.encoding=UTF-8 -Dconfig.file=/etc/thehive/application.conf -Dlogger.file=/etc/thehive/log

Aug 17 14:16:27 TheHive-Server systemd[1]: Started Scalable, Open Source and Free Security Incident Response Solutions.
root@TheHive-Server:~#
```

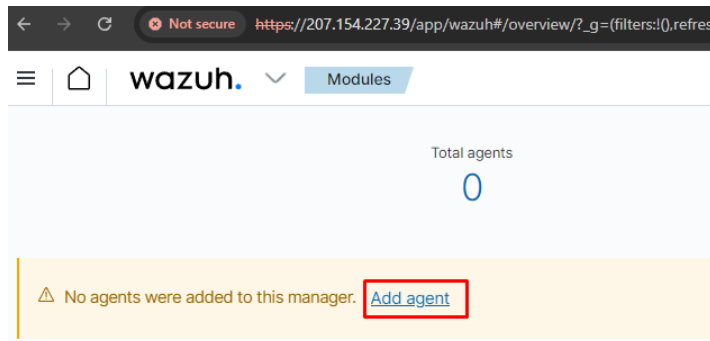
Now TheHive Dashboard is active, after activating and configuring elastic search and Cassandra




And we have a TheHive dashboard!





Now I'll configure an agent on the Wazuh Dashboard



Select the package to download and install on your system:

 **LINUX**
☐ RPM amd64 ☐ RPM aarch64
☐ DEB amd64 ☐ DEB aarch64

 **WINDOWS**
☒ MSI 32/64 bits

 **macOS**
☐ Intel
☐ Apple silicon

① For additional systems and architectures, please check our documentation [here](#).

Server address:

This is the address the agent uses to communicate with the server. Enter an IP address or a fully qualified domain name (FDQN).

Assign a server address: [?](#)

207.154.227.39

Wazuh Public IP Address

Optional settings:

By default, the deployment uses the hostname as the agent name. Optionally, you can use a different agent name in the field below.

Assign an agent name: [?](#)

Agent

① The agent name must be unique. It can't be changed once the agent has been enrolled. [?](#)

Now I will run these following commands with AdminPriv using PowerShell on the Windows10 Client



Run the following commands to download and install the agent:

```
Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.7.5-1.msi -OutFile ${{env.tmp}}\wazuh-agent; msixec.exe /i ${{env.tmp}}\wazuh-agent /q WAZUH_MANAGER='207.154.227.39' WAZUH_AGENT_NAME='Agent' WAZUH_REGISTRATION_SERVER='207.154.227.39'
```

④ Requirements

- You will need administrator privileges to perform this installation.
- PowerShell 3.0 or greater is required.

Keep in mind you need to run this command in a Windows PowerShell terminal.

5

Start the agent:

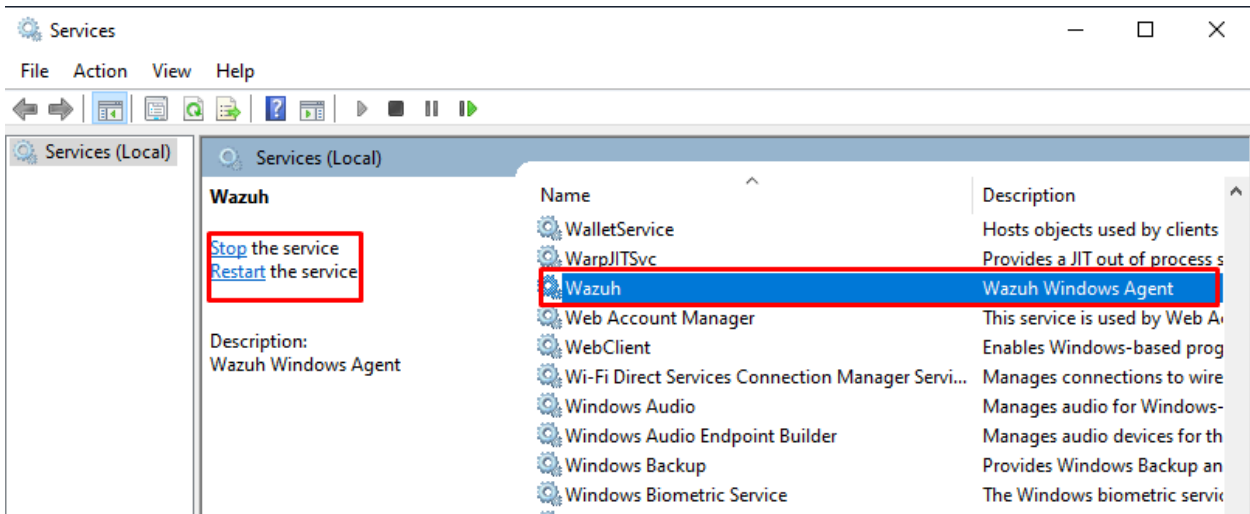
```
NET START WazuhSvc
```

Close

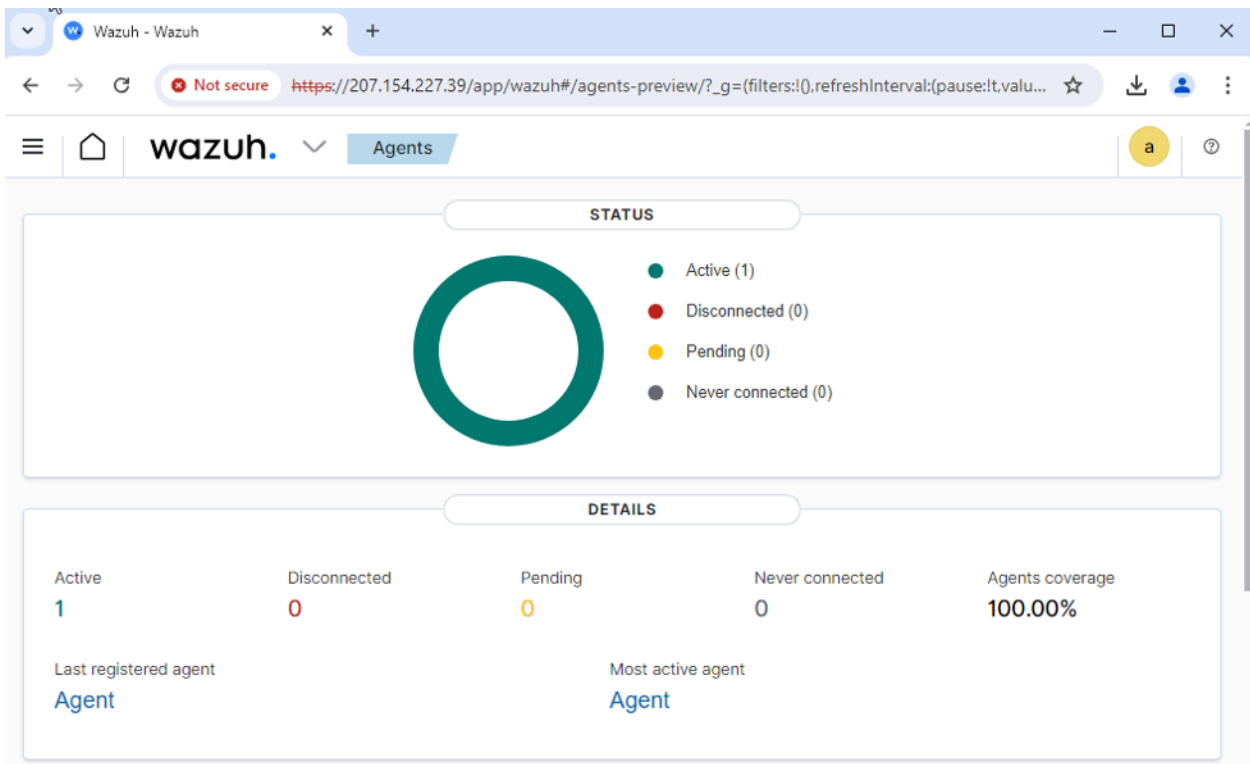
The Wazuh Service is now running on our Client

```
PS C:\Windows\system32> Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.7.5-1.msi -OutFile ${{env.tmp}}\wazuh-agent; msixec.exe /i ${{env.tmp}}\wazuh-agent /q WAZUH_MANAGER='207.154.227.39' WAZUH_AGENT_NAME='Agent' WAZUH_REGISTRATION_SERVER='207.154.227.39'
PS C:\Windows\system32>
PS C:\Windows\system32> NET START WazuhSvc
The Wazuh service is starting.
The Wazuh service was started successfully.
PS C:\Windows\system32>
```

We can also see its running in the services



Lets check the Wazuh Dashboard

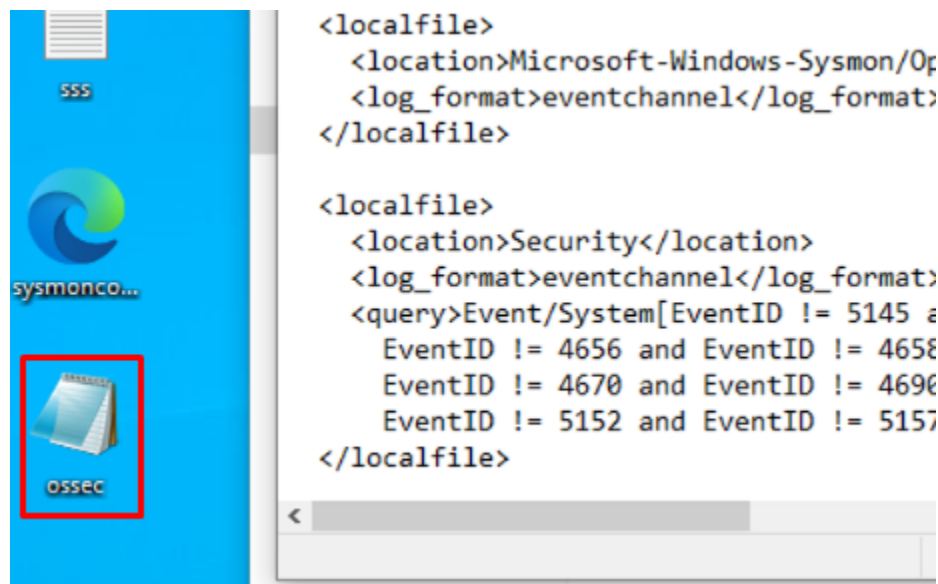


And the Agent is Active and integrated

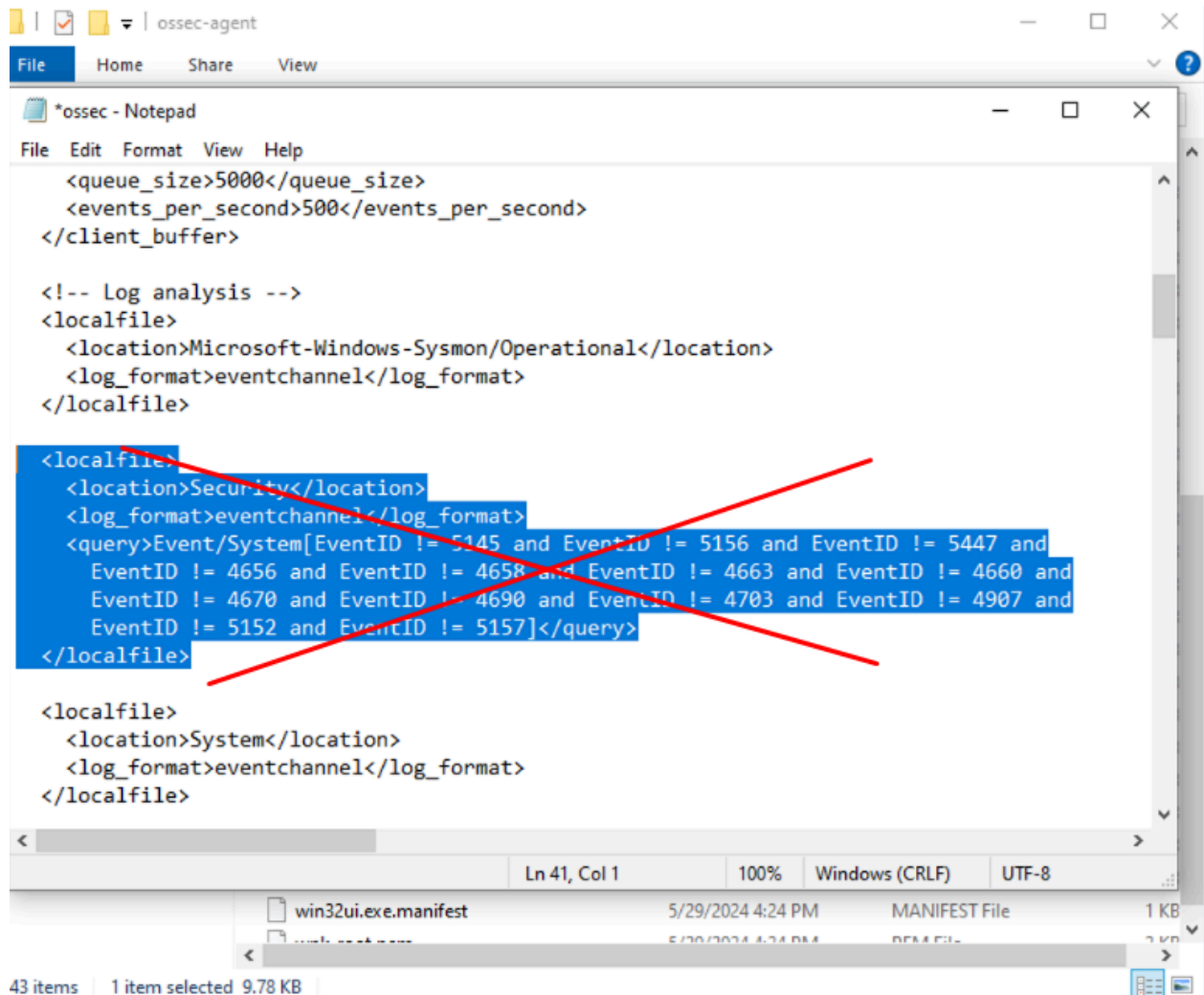
Part 3: Generate & Ingest Telemetry

I'll start by configuring the ossec.conf file on the Client Machine, I'll enable mimikatz log collection as well

I'll start by making a backup of the ossec.conf file, in case something is misconfigured



Security+System logs removed, I want to focus on Sysmon logs and implementing mimikats



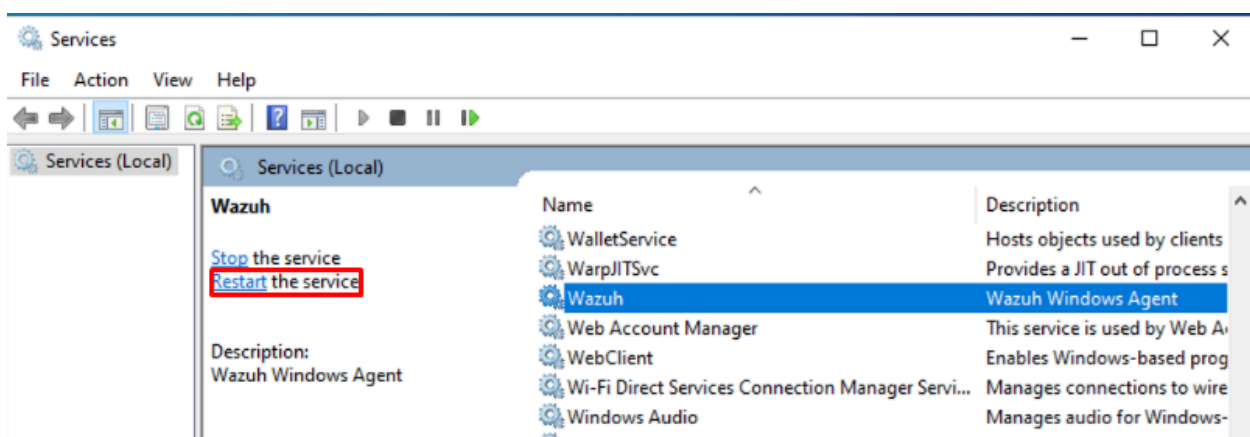
```
*ossec - Notepad
File Edit Format View Help
<queue_size>5000</queue_size>
<events_per_second>500</events_per_second>
</client_buffer>

<!-- Log 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 EventID != 5447 and
    EventID != 4656 and EventID != 4658 and EventID != 4663 and EventID != 4660 and
    EventID != 4670 and EventID != 4690 and EventID != 4703 and EventID != 4907 and
    EventID != 5152 and EventID != 5157]</query>
</localfile>

<localfile>
  <location>System</location>
  <log_format>eventchannel</log_format>
</localfile>
```

Next I will restart Wazuh to apply the changes

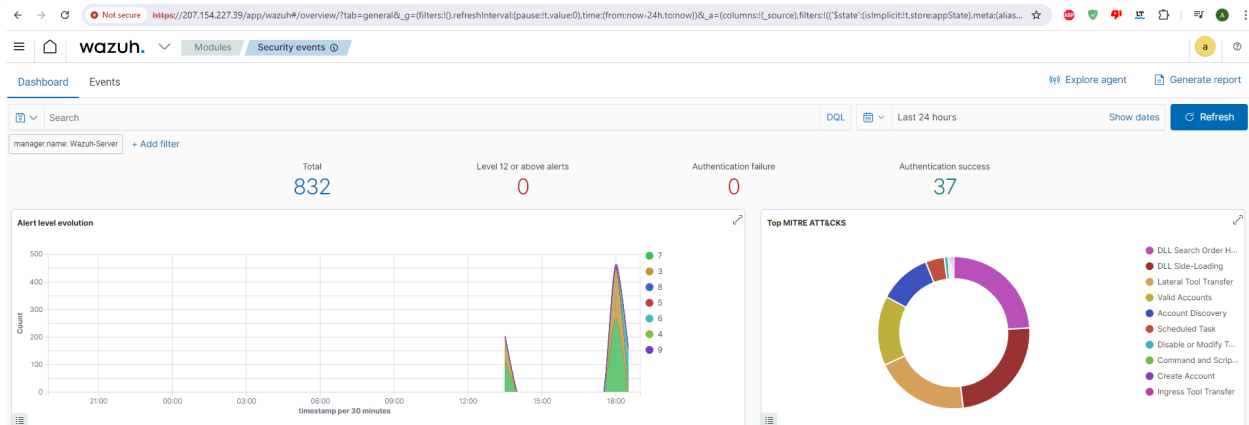


We see the agent is active

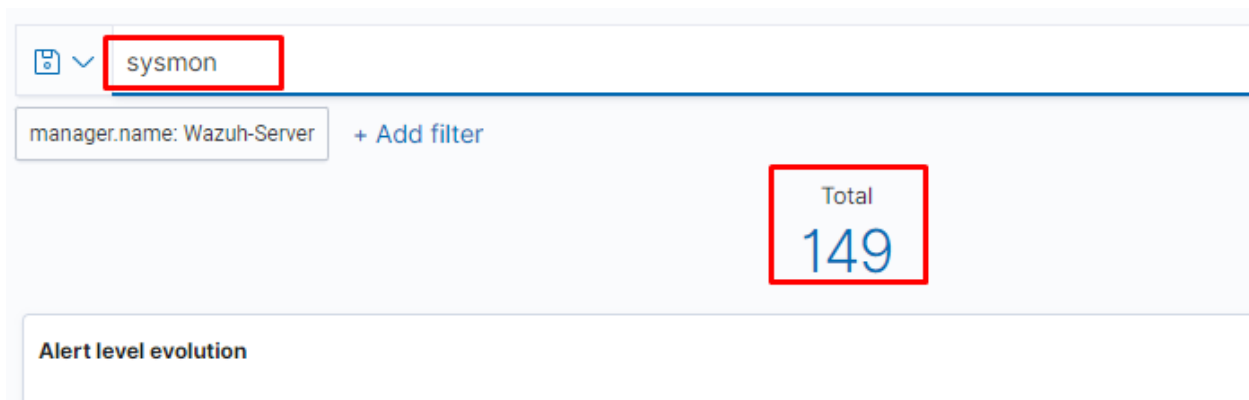
| ID ↑ | Name | IP address | Group(s) | Operating system | Cluster node | Version | Status | Actions |
|------|-------|------------|----------|--|--------------|---------|----------|-------------------------------------|
| 001 | Agent | 10.0.2.15 | default | Microsoft Windows 10 Pro 10.0.19045.3803 | node01 | v4.7.5 | ● active | 🔍 🔗 |

Rows per page: 10

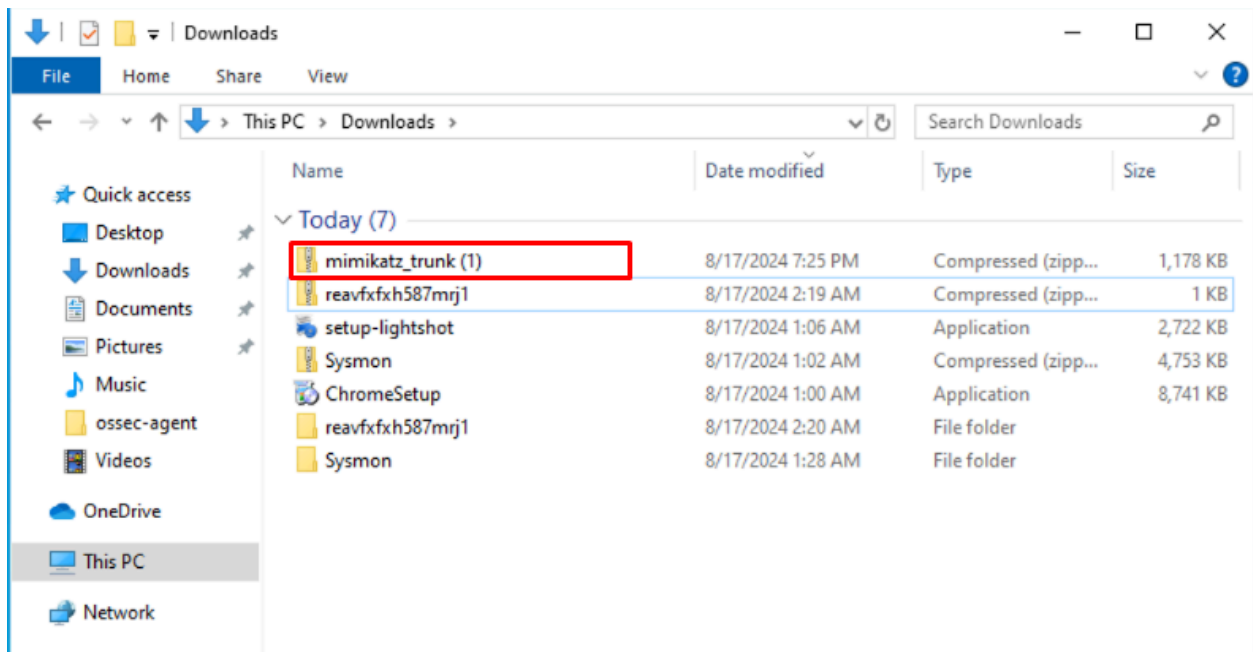
The Dashboard is active and responding



Sysmon Logs are querying



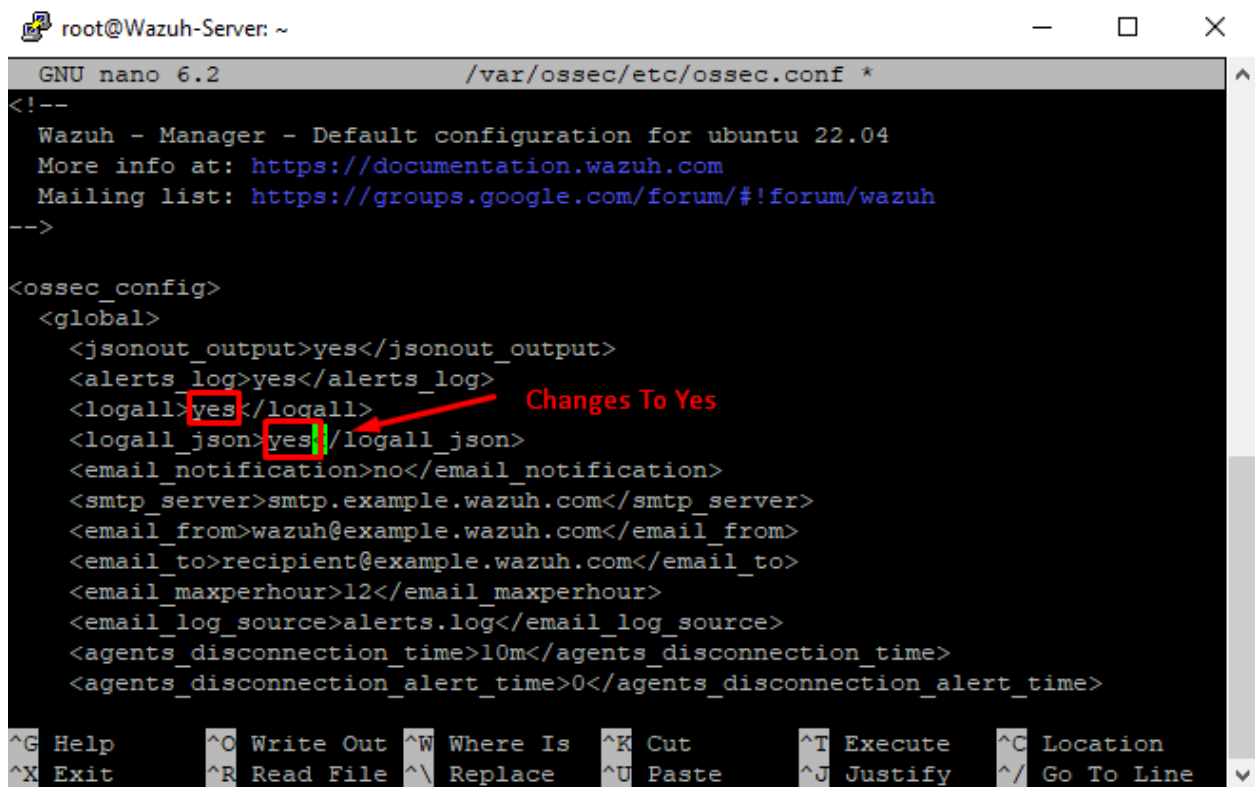
Next I'll download mimikatz, mimikatz is a PT tool used to extract credentials, I will need to disable the windows defender/create exclusions in order to do this



Mimikatz is downloaded after using exclusion and temp disabling chrome defenses

Now I'll configure the Wazuh Server ossec file for better alert filtering, but first a BACKUP

```
Last login: Sat Aug 17 10:29:19 2024 from 147.235.207.170
root@Wazuh-Server:~#
root@Wazuh-Server:~# cp /var/ossec/etc/ossec.conf ~/ossec-backup.conf
root@Wazuh-Server:~#
```



```
GNU nano 6.2 /var/ossec/etc/ossec.conf *
<!--
Wazuh - Manager - Default configuration for ubuntu 22.04
More info at: https://documentation.wazuh.com
Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

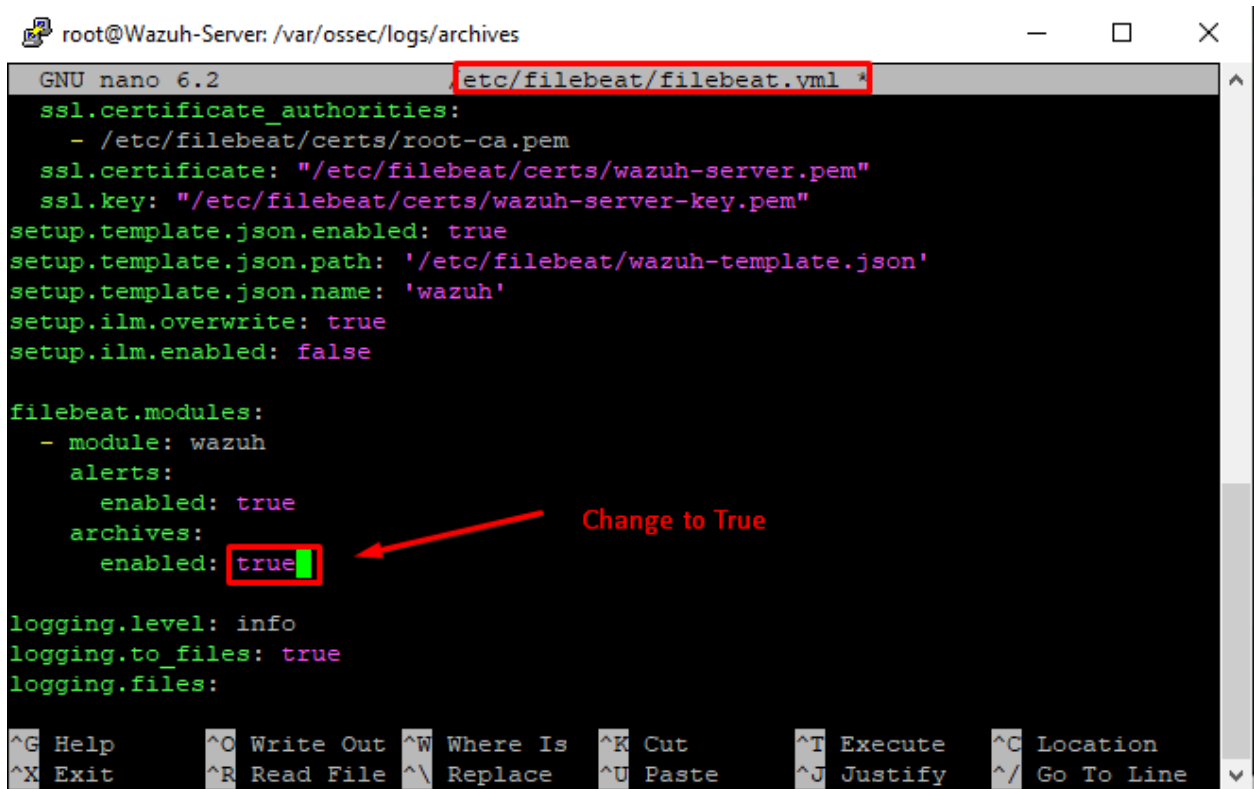
<ossec_config>
  <global>
    <jsonout_output>yes</jsonout_output>
    <alerts_log>yes</alerts_log>
    <logall>yes</logall>
    <logall_json>yes</logall_json>
    <email_notification>no</email_notification>
    <smtp_server>smtp.example.wazuh.com</smtp_server>
    <email_from>wazuh@example.wazuh.com</email_from>
    <email_to>recipient@example.wazuh.com</email_to>
    <email_maxperhour>12</email_maxperhour>
    <email_log_source>alerts.log</email_log_source>
    <agents_disconnection_time>10m</agents_disconnection_time>
    <agents_disconnection_alert_time>0</agents_disconnection_alert_time>

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

This will make Wazuh archive the log files

```
Last login: Sat Aug 17 10:29:19 2024 from 147.235.207.170
root@Wazuh-Server:~#
root@Wazuh-Server:~# cp /var/ossec/etc/ossec.conf ~/ossec-backup.conf
root@Wazuh-Server:~# nano /var/ossec/etc/ossec.conf
root@Wazuh-Server:~# systemctl restart wazuh-manager.service
root@Wazuh-Server:~# cd /var/ossec/logs/archives/
root@Wazuh-Server:/var/ossec/logs/archives# ls
2024 archives.json archives.log
root@Wazuh-Server:/var/ossec/logs/archives#
```

Now we want Wazuh to ingest the files, we edit the filebeat.vml file




```
root@Wazuh-Server: /var/ossec/logs/archives
GNU nano 6.2 /etc/filebeat/filebeat.vml
ssl.certificate_authorities:
  - /etc/filebeat/certs/root-ca.pem
ssl.certificate: "/etc/filebeat/certs/wazuh-server.pem"
ssl.key: "/etc/filebeat/certs/wazuh-server-key.pem"
setup.template.json.enabled: true
setup.template.json.path: '/etc/filebeat/wazuh-template.json'
setup.template.json.name: 'wazuh'
setup.ilm.overwrite: true
setup.ilm.enabled: false

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

logging.level: info
logging.to_files: true
logging.files:
```

After configuring the Wazuh Manager, I can edit the Wazuh index patterns

 Create index pattern

I want Wazuh to be able to Index the archives regardless if they triggered an alert or not

For everything

Step 1 of 2: Define an index pattern

Index pattern name

wazuh-archives-*

Use an asterisk (*) to match multiple indices. Spaces and the ch

☐ Include system and hidden indices

✓ Your index pattern matches 1 source.

The new Archive Index is aggregating logs as expected



Wazuh is unique because on default it doesn't collect logs unless a specific rule has been met, that's why I changed the configs to make it collect logs regardless if a rule is met.

Now its time to test it, I'll run mimikatz on the client and see if we get the alert and logs from our SIEM/XDR Wazuh

```
PS C:\Users\WazuhAgent\Downloads\mimikatz_trunk\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 # exit
Bye!
PS C:\Users\WazuhAgent\Downloads\mimikatz_trunk\x64> get-date

Saturday, August 17, 2024 8:23:43 PM

PS C:\Users\WazuhAgent\Downloads\mimikatz_trunk\x64>
```

Now let's check if we can see the logs



NICE!

_source

```
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": "2024-08-18T03:22:06.5629576Z", "eventRecordID": "4189", "processID": "2904", "threadID": "4524", "channel": "Microsoft-Windows-Sysmon/Operational", "computer": "DESKTOP-6QQ4GGF", "severityValue": "INFORMATION", "message": "\\Image loaded:\\r\\nRuleName: technique_id=T1574.002, technique_name=DLL Side-Loading\\r\\nUtcTime: 2024-08-18 03:22:06.555\\r\\nProcessGuid: {984dd485-68de-66c1-ae03-00000000200}\\r\\nProcessId: 7960\\r\\nImage: C:\\\\Users\\WazuhAgent\\Downloads\\mimikatz_trunk\\x64\\mimikatz.exe\\r\\nImageLoaded:
```

Now I'll create some specific sysmon rules using the Wazuh GUI

Rules (4372)
From here you can manage your rules.

Manage rules files

Add new rules file

Refresh

Export formatted

Search

WQL

Custom rules

| ID ↑ | Description | Groups | Regulatory compliance | Level | File | Path |
|------|---|-----------------------|-----------------------|-------|-----------------------|---------------|
| 1 | Generic template for all syslog rules. | syslog | | 0 | 0010-rules_config.xml | ruleset/rules |
| 2 | Generic template for all firewall rules. | firewall | | 0 | 0010-rules_config.xml | ruleset/rules |
| 3 | Generic template for all ids rules. | ids | | 0 | 0010-rules_config.xml | ruleset/rules |
| 4 | Generic template for all web rules. | web-log | | 0 | 0010-rules_config.xml | ruleset/rules |
| 5 | Generic template for all web proxy rules. | squid | | 0 | 0010-rules_config.xml | ruleset/rules |
| 6 | Generic template for all windows rules. | windows | | 0 | 0010-rules_config.xml | ruleset/rules |
| 7 | Generic template for all wazuh rules. | ossec | | 0 | 0010-rules_config.xml | ruleset/rules |
| 200 | Grouping of wazuh rules. | wazuh | | 0 | 0016-wazuh_rules.xml | ruleset/rules |
| 201 | Agent event queue rule | agent_flooding, wazuh | | 0 | 0016-wazuh_rules.xml | ruleset/rules |
| 202 | Agent event queue is level full. | agent_flooding, wazuh | PCIDSSGDPR | 7 | 0016-wazuh_rules.xml | ruleset/rules |

Rows per page: 10

< 1 2 3 4 5 ... 438 >

I'll build a custom rule using 0800-sysmon_id_1

sysmon

| File ↑ | Path | Action |
|---------------------------|---------------|--------|
| 0330-sysmon_rules.xml | ruleset/rules | 👁 |
| 0595-win-sysmon_rules.xml | ruleset/rules | 👁 |
| 0800-sysmon_id_1.xml | ruleset/rules | 👁 |
| 0810-sysmon_id_3.xml | ruleset/rules | 👁 |
| 0820-sysmon_id_7.xml | ruleset/rules | 👁 |
| 0830-sysmon_id_11.xml | ruleset/rules | 👁 |
| 0860-sysmon_id_13.xml | ruleset/rules | 👁 |
| 0870-sysmon_id_8.xml | ruleset/rules | 👁 |
| 0945-sysmon_id_10.xml | ruleset/rules | 👁 |
| 0950-sysmon_id_20.xml | ruleset/rules | 👁 |

Rows per page: 10

I'll copy one of these default Wazuh Sysmon_id_1 rules and build my own custom rule detecting mimikatz

< local_rules.xml

```
1 <!-- Local rules -->
2
3 <!-- Modify it at your will. -->
4 <!-- Copyright (C) 2015, Wazuh Inc. -->
5
6 <!-- Example -->
7 <group name="local,syslog,sshd,">
8
9 <!--
10 Dec 10 01:02:02 host sshd[1234]: Failed none for root from 1.1.1.1 port 1066 ssh2
11 -->
12 <rule id="100001" level="5">
13   <if_sid>5716</if_sid>
14   <srcip>1.1.1.1</srcip>
15   <description>sshd: authentication failed from IP 1.1.1.1.</description>
16   <group>authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5,</group>
17 </rule>
18
19 <rule id="92000" level="4">
20   <if_group>sysmon_event1</if_group>
21   <field name="win.eventdata.parentImage" type="pcr2">(?i)\\(c|w)script\.exe</field>
22   <options>no_full_log</options>
23   <description>Scripting interpreter spawned a new process</description>
24   <mitre>
25     <id>T1059.005</id>
26   </mitre>
27 </rule>
28
29 </group>
30
```

Custom Rules

Added this as the base for my New rule

The New Custom Mimikatz Rule

```
<!--
Dec 10 01:02:02 host sshd[1234]: Failed none for root from 1.1.1.1 port 1066 ssh2
-->
<rule id="100001" level="5">
  <if_sid>5716</if_sid>
  <srcip>1.1.1.1</srcip>
  <description>sshd: authentication failed from IP 1.1.1.1.</description>
  <group>authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5,</group>
</rule>

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

</group>
```

Alert will triggered regardless if attacker changed the file name

credential dumping

Now to test if this works, I'll run Mimikatz after changing the file name to memecats and see if the new rule is triggered

```
PS C:\Users\WazuhAgent\Downloads\mimikatz_trunk\x64> ./memecats.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 # ?
ERROR mimikatz_doLocal ; "?" command of "standard" module not found !

Module :      standard
Full name :    Standard module
Description :  Basic commands (does not require module name)

    exit - Quit mimikatz
    cls  - Clear screen (doesn't work with redirections, like PsExec)
    answer - Answer to the Ultimate Question of Life, the Universe, and Everything
    coffee - Please, make me a coffee!
    sleep - Sleep an amount of milliseconds
    log   - Log mimikatz input/output to file
    base64 - Switch file input/output base64
    version - Display some version informations
    cd    - Change or display current directory
    localtime - Displays system local date and time (OJ command)
    hostname - Displays system local hostname

mimikatz # exit
Bye!
PS C:\Users\WazuhAgent\Downloads\mimikatz_trunk\x64> get-date

Saturday, August 17, 2024 8:53:54 PM

PS C:\Users\WazuhAgent\Downloads\mimikatz_trunk\x64>
```

The new custom rule is working as intended, it was able to alert about mimikatz usage despite a deception attempt

| | | | | | | | | |
|---|-----------------------------|-----|-------|-------|-------------------|-------------------------|----|--------|
| > | Aug 17, 2024 @ 20:53:23.052 | 001 | Agent | T1003 | Credential Access | Mimikatz Usage Detected | 15 | 100002 |
|---|-----------------------------|-----|-------|-------|-------------------|-------------------------|----|--------|

Looking deeper into the log we can see the Original File name mimikatz has triggered the alert

| | |
|--------------------------------------|---|
| @timestamp | 2024-08-17T17:53:23.052Z |
| _id | xkJ4YZEBWkWDxbv6_Ds |
| agent.id | 001 |
| agent.ip | 10.0.2.15 |
| agent.name | Agent |
| data.win.eventdata.commandLine | "C:\\Users\\WazuhAgent\\Downloads\\mimikatz_trunk\\x64\\memecats.exe" |
| data.win.eventdata.company | gentilkiwi (Benjamin DELPY) |
| data.win.eventdata.currentDirectory | C:\\Users\\WazuhAgent\\Downloads\\mimikatz_trunk\\x64\\ |
| data.win.eventdata.description | mimikatz for Windows |
| data.win.eventdata.fileVersion | 2.2.0.0 |
| data.win.eventdata.hashes | SHA1=E3B6EA8C46FA831CEC6F235A5CF48B38A4AE8D69,MD5=29EFD64DD3CFC49F27A98AE456D8EDF |
| data.win.eventdata.image | C:\\Users\\WazuhAgent\\Downloads\\mimikatz_trunk\\x64\\memecats.exe |
| data.win.eventdata.integrityLevel | Medium |
| data.win.eventdata.logonGuid | {984dd485-4b8f-66c1-3452-040000000000} |
| data.win.eventdata.logonId | 0x45234 |
| data.win.eventdata.originalFileName | mimikatz.exe |
| data.win.eventdata.parentCommandLine | "C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe" |
| data.win.eventdata.parentImage | C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe |
| data.win.eventdata.parentProcessGuid | {984dd485-6fd1-66c1-2b04-000000000200} |
| data.win.eventdata.parentProcessId | 5536 |

| | |
|----------------------|------------------------------|
| input.type | log |
| location | EventChannel |
| manager.name | Wazuh-Server |
| rule.description | Mimikatz Usage Detected |
| rule.firedtimes | 1 |
| rule.groups | local, syslog, sshd |
| rule.id | 100002 |
| rule.level | 15 |
| rule.mail | true |
| rule.mitre.id | T1003 |
| rule.mitre.tactic | Credential Access |
| rule.mitre.technique | OS Credential Dumping |
| timestamp | 2024-08-17T17:53:23.052+0000 |

Part 4: IMPLEMENTING AUTOMATION (SOAR)

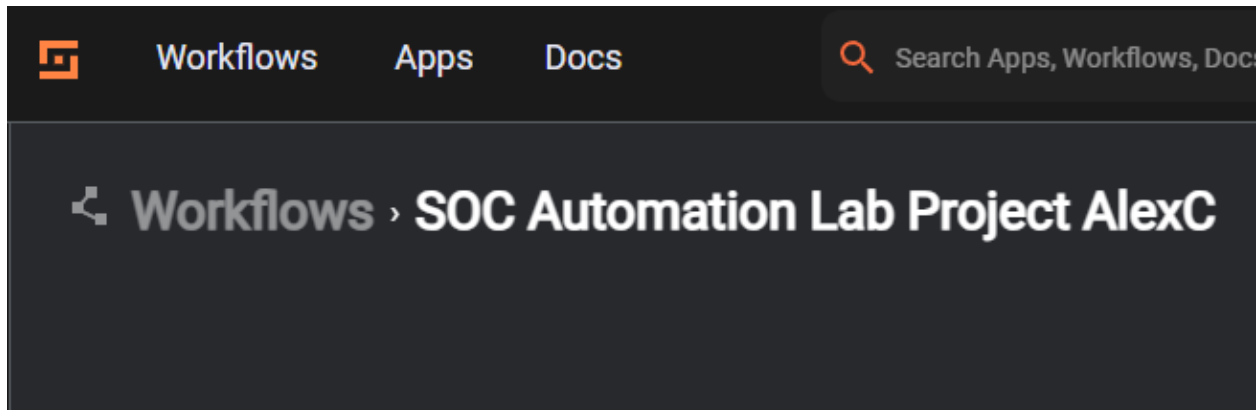
- CONNECT SHUFFLE (SOAR)
- SEND ALERTS TO THE HIVE
- SEND ALERTS VIA EMAIL TO THE SOC ANALYST

By the end of this, I'm going to have a fully functional live telemetry ingesting/digesting lab
While implementing automation & orchestration
Using:

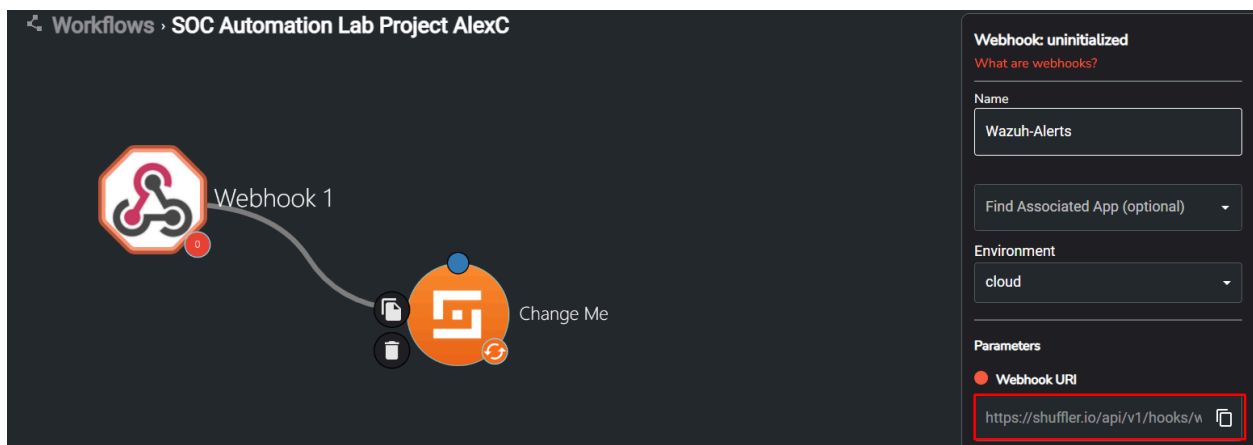
- Wazuh (SIEM/XDR)
- Shuffle (SOAR)
- The Hive (CaseManagmentSystem)

Also sending alerts in real time to the SOC analyst
via email

Starting by using a Shuffle account to create
“apps” or “triggers”



Next I'm going to copy the Webhook URI, to add it
to my ossec conf file on the Wazuh-manager



Now I'm going to use the Wazuh-Server CLI to connect to shuffle by adding an integration tag in the ossec config file, and adding the hook_url

```
<integration>
  <name>shuffle</name>
  <hook_url>http://https://shuffler.io/api/v1/hooks/webhook_61f355e0-492e-4011-b1db-6bda0ef332a4 </hook_url>
  <level>3</level>
  <alert_format>json</alert_format>
</integration>
```

And I'll change it from sent all alerts above 3 to the special mimikatz rule I created

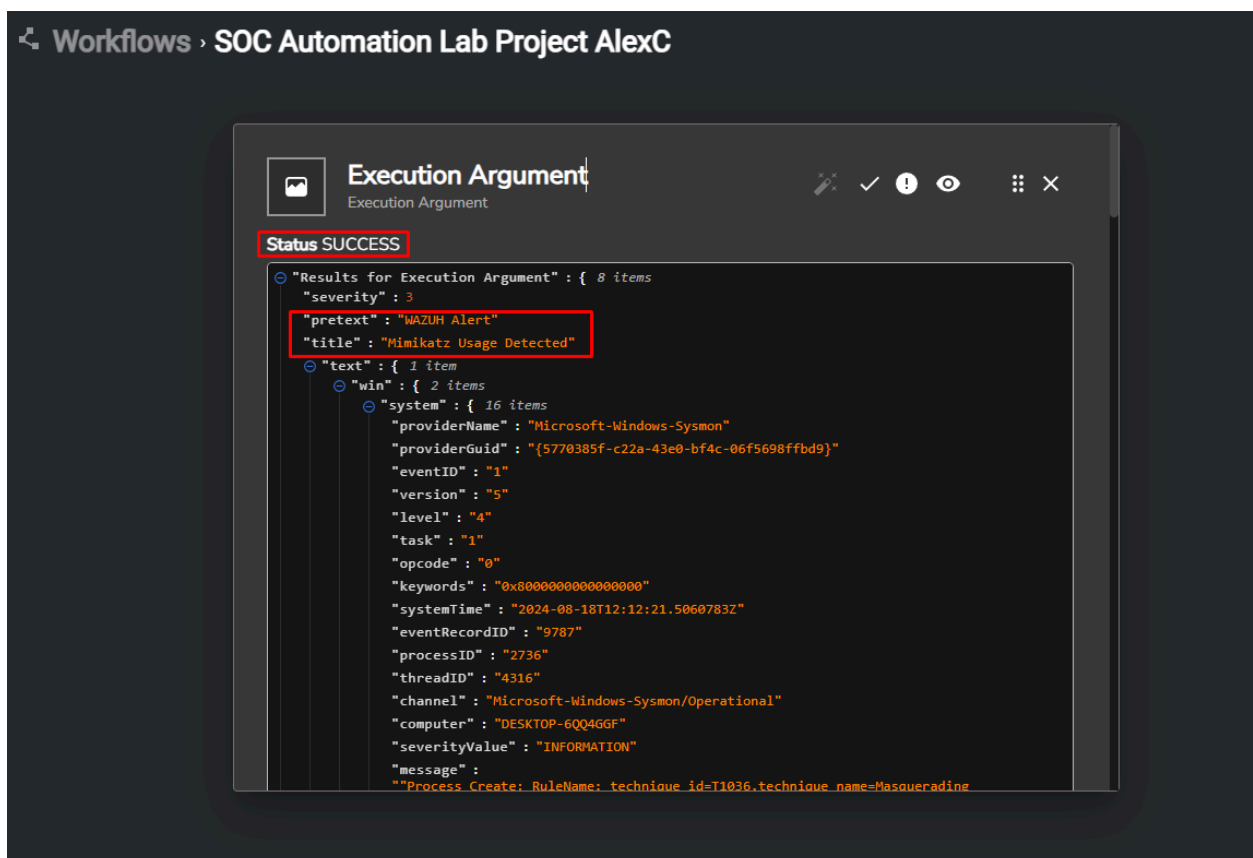
```
<integration>
  <name>shuffle</name>
  <hook_url>http://https://shuffler.io/
  <rule_id>100002</rule_id>
  <alert_format>json</alert_format>
</integration>
```

Troubleshooting by changing to https:// instead of http://https://

```
<integration>
  <name>shuffle</name>
  <hook_url>https://shuffler.io/api/v1
  <rule_id>100002</rule_id>
  <alert_format>json</alert_format>
</integration>
```


I'll start the webhook and run memecats.exe again on the Client

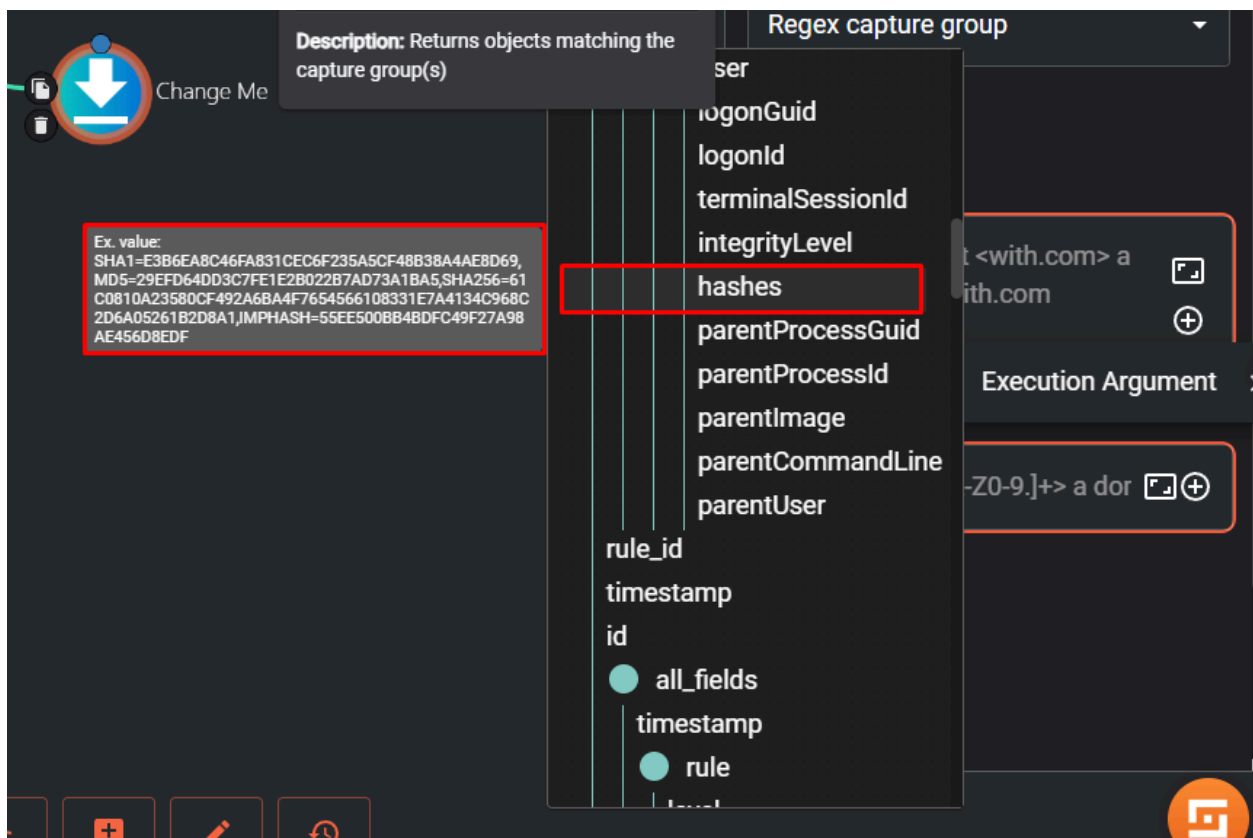
And It's working, I'm getting the 100002 alert to shuffle + all the information generated by Wazuh itself into shuffle, pretty cool!



Next I will be creating this workflow:

1. Mimikatz alert sent to shuffle
2. Shuffle receives alert and extracts SHA256 from the file
3. Check the reputation score using VirusTotal
4. Send Details to TheHive to create an Alert
5. Send Email to the SOC Analyst to begin investigation

I'll start with creating a hash argument



Next, I will use chatGPT to parse the value of SHA256 of the hash



create a regex to parse the sha256 value Ex. value:

```
"hashes": "SHA1=E3B6EA8C46FA831CEC6F235A5CF48B38A4AE8D69,M  
D5=29EFD64DD3C7FE1E2B022B7AD73A1BA5,SHA256=61C0810A235  
80CF492A6BA4F7654566108331E7A4134C968C2D6A05261B2D8A1,IM  
PHASH=55EE500BB4BDFC49F27A98AE456D8EDF"
```




To create a regex that extracts the SHA256 value from the given string, you can use the following pattern:







regex

Copy code

```
SHA256=([A-Fa-f0-9]{64})
```

Success The SHA256 is parsed

 **Change Me**
regex_capture_group

Status SUCCESS

```
"Results for Change_Me" : { 3 items
  "success" : true
  "group_0" : [ 1 item
    "61C0810A23580CF492A6BA4F7654566108331E7A4134C968C2D6A05261B2D8A1"
  ]
  "found" : true
}
```

Variables (click to expand)

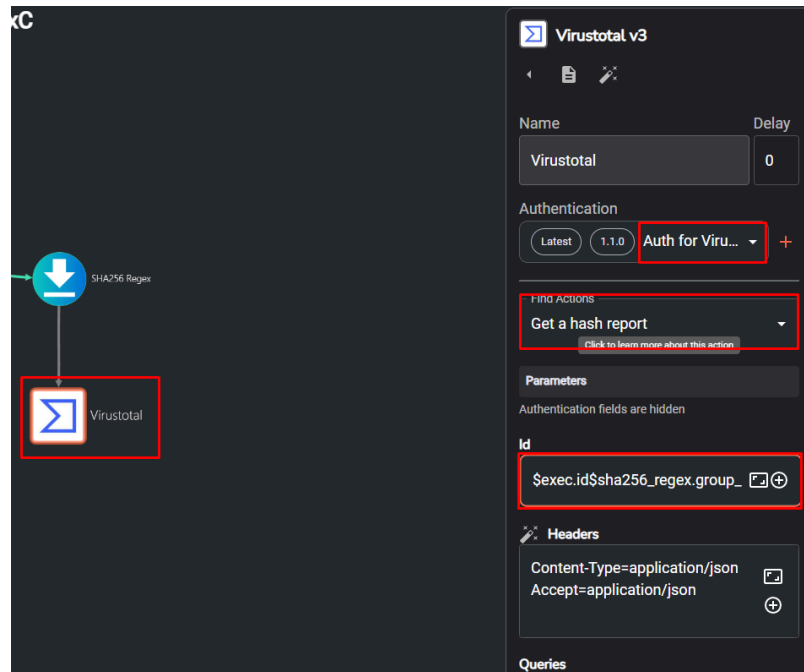
input_data

regex: SHA256=([A-Fa-f0-9]{64})

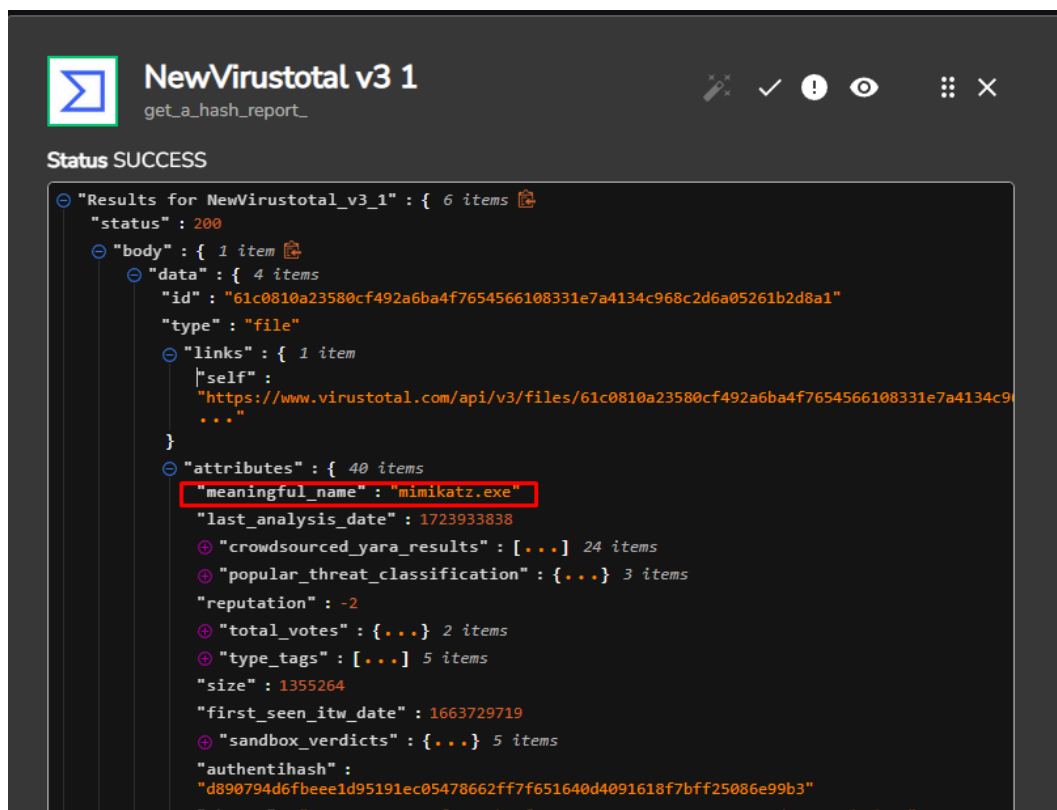
Action Logs

Logs for an action are not available without [an onprem environment](#) with the [SHUFFLE_LOGS_DISABLED](#) environment variable set to false: SHUFFLE_LOGS_DISABLED=false.
Logs are enabled by default, except in scale mode.

Next I will use the VirusTotal API Key and integrated it into shuffle



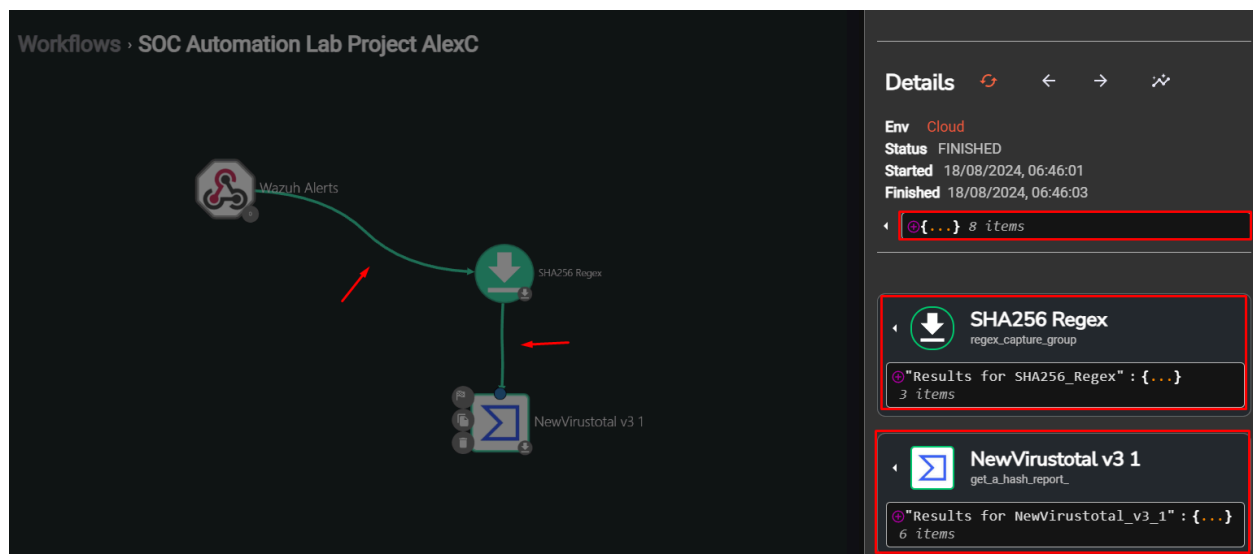
After alot of time spend on troubleshooting, the Virustotal API is able to integrate with Shuffle



We can see that 67 vendors flagged it as Malicious


```
"creation_date" : 1663602279
⊖ "last_analysis_stats" : { 8 items
  "malicious" : 67
  "suspicious" : 0
  "undetected" : 8
  "harmless" : 0
  "timeout" : 0
  "confirmed-timeout" : 0
  "failure" : 0
  "type-unsupported" : 4
}
```

Everything is integrated, Great!




Quick recap: I setup my SOAR platform to receive the Wazuh alerts than performed regex to parse the SHA256 hash & integrated VirusTotal to check the reputation, Next I will send the details to TheHive


I'll login into TheHive GUI, I'll create a new org+new users



Organisation List


→





+

default

 Export list

| <input type="checkbox"/> | NAME ↕ |
|--------------------------|---|
| <input type="checkbox"/> | <div><div>A</div><div>admin</div><div>Linked organisations None</div></div> |

Adding an Organisation

* Name

AlexC

* Description

SOC Automation Project

I'll create a new Analyst profile

Adding a User

×

Type

Normal

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

Organisation

AlexC

* Login

SOC1@test.com

* Name

SOC1

* 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

And I'll create a shuffle user, in a real world scenario there would be least privilege unique profiles for each user/groups, for the purpose of this Lab Project I'll be using the analyst profile

Adding a User

Type

Service

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

Organisation

AlexC

* Login

Shuffle@test.com

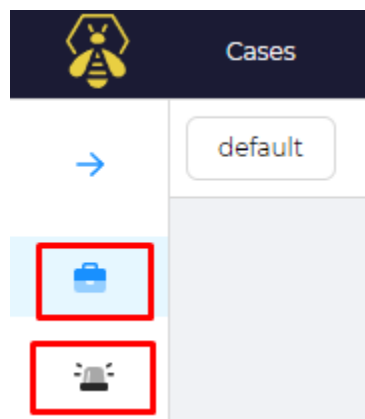
* Name

SOAR

* Profile

analyst

I'll copy the SOAR user API key & save it, next I will login into the SOC1@test.com user




Next I will auth TheHive in Shuffle with the APIkey


Workflows › SOC Automation Lab Project AlexC

Authentication for TheHive

[What is app authentication?](#)
These are required fields for authenticating with TheHive

Name - what is this used for?
Auth for TheHive

 apikey
.....

 url
http://64.227.119.244:9000

CANCEL SUBMIT

There is
Docum
con

And I'll create an mimikatz alert

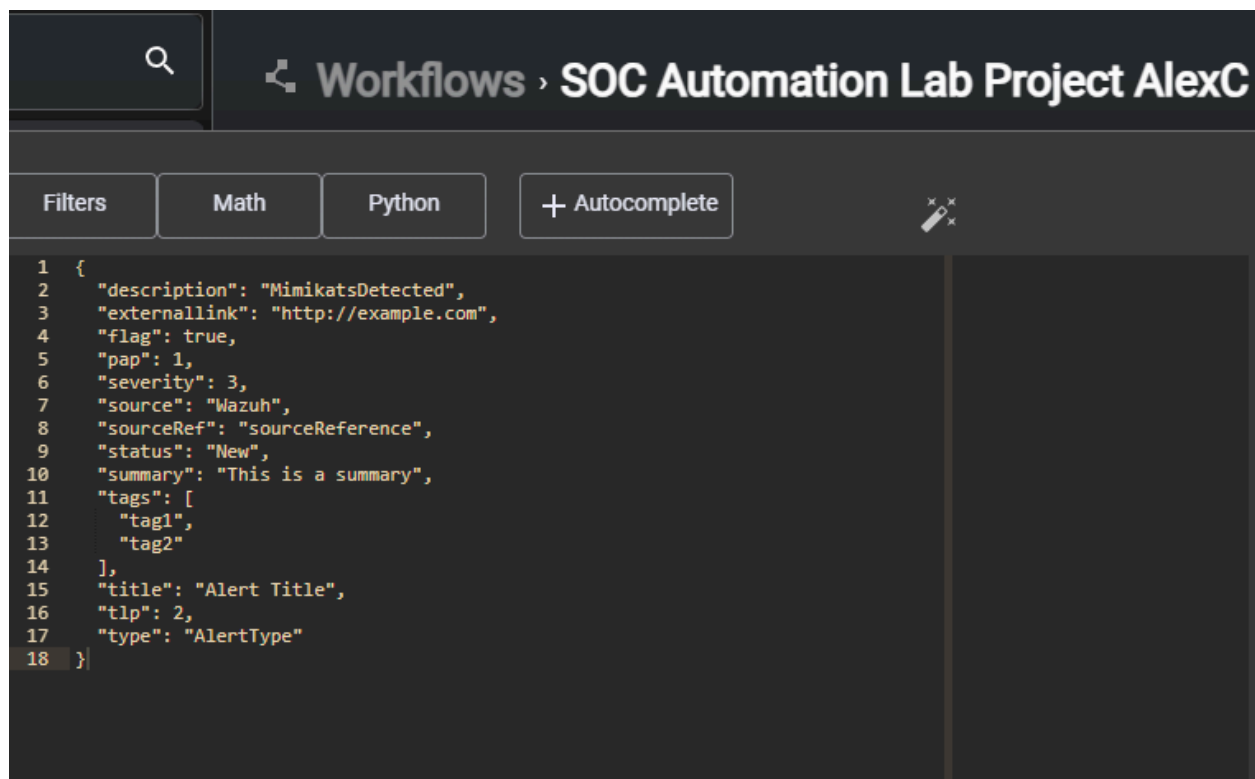
```
Filters Math Python + Autocomplete
```

```
1 {
2   "description": "{{ '${mimikatz.detected.on.host.computer}' | replace: '\n', '\\r\\n' }}",
3   "externallink": "${externallink}",
4   "flag": ${false},
5   "pap": ${2},
6   "severity": "${2}",
7   "source": "${Wazuh}",
8   "sourceRef": "${Rule: 100002}",
9   "status": "${New}",
10  "summary": "${mimikatz activitey detected on host}",
11  "tags": "${["T1003"]}",
12  "title": "${mimikatz detected}",
13  "tlp": ${2},
14  "type": "${interal}"
15 }
```


Now I'll add a new firewall rule in my cloud server in order to allow the Hive proper querying

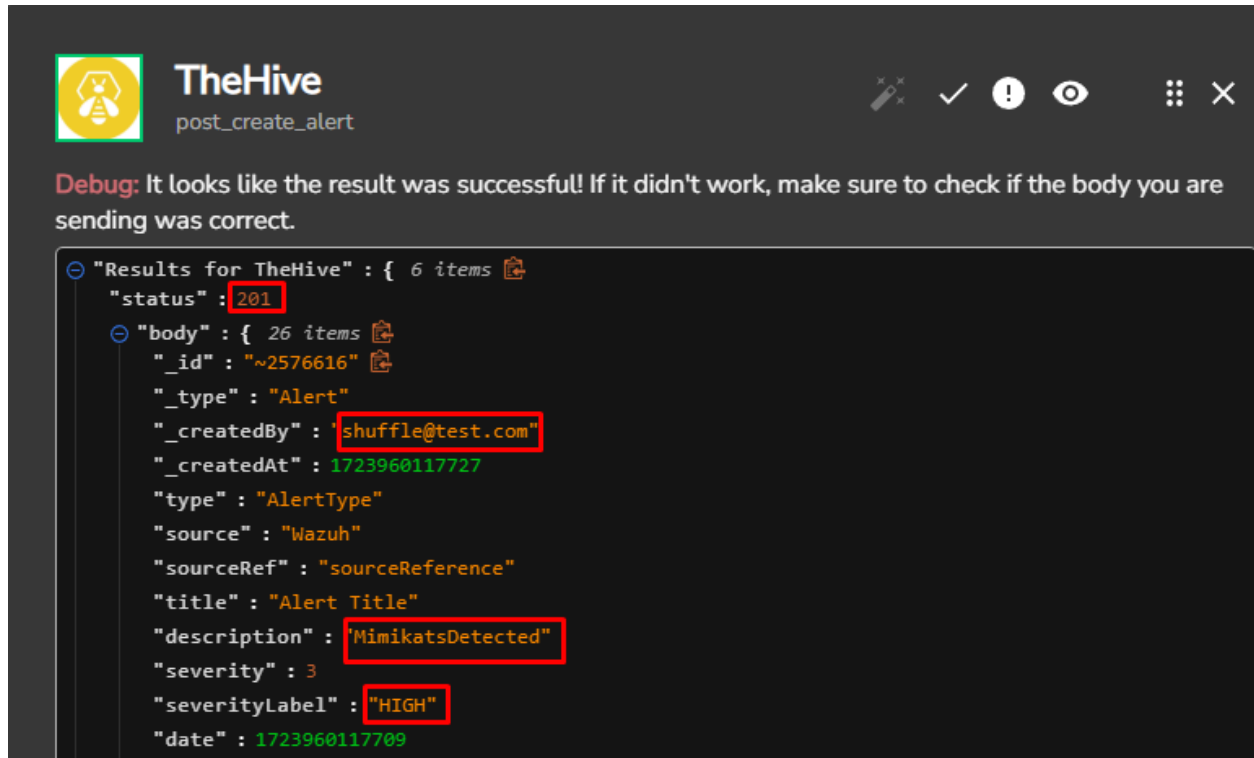
| Type | Protocol | Port Range | Sources | |
|---------|----------|------------|-----------------|-------------------------------------|
| All TCP | TCP | All ports | 147.235.207.170 | More ▼ |
| SSH | TCP | 22 | 147.235.207.170 | More ▼ |
| Custom | TCP | 9000 | All IPv4 | More ▼ |
| All UDP | UDP | All ports | 147.235.207.170 | More ▼ |

I'll create a custom create alert rule and test to see if the SOC analyst is getting live alerts

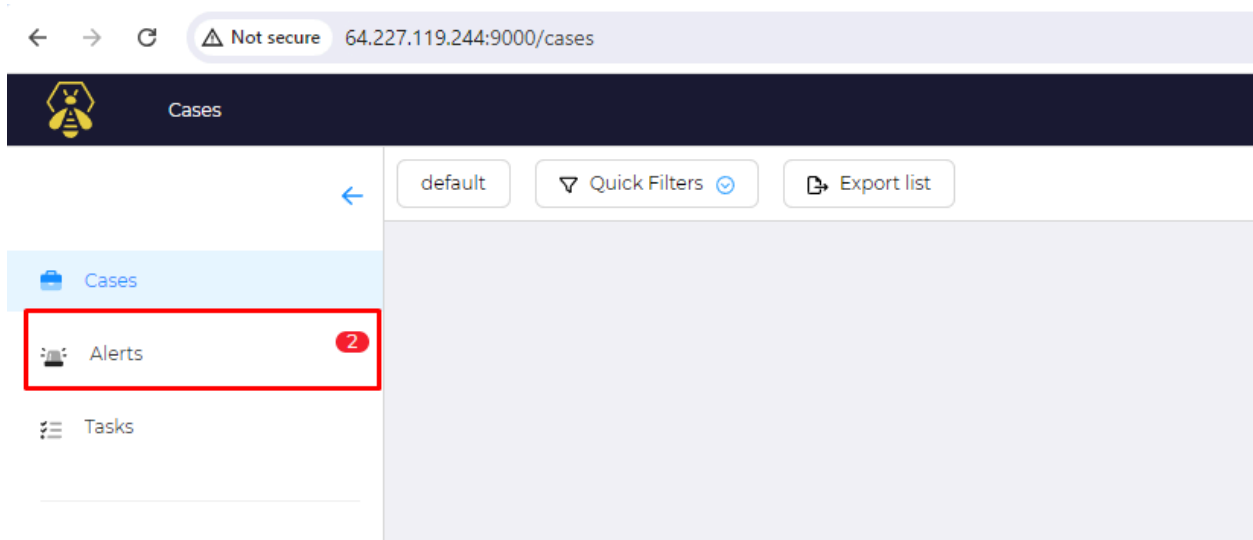


```
1 {
2   "description": "MimikatsDetected",
3   "externallink": "http://example.com",
4   "flag": true,
5   "pap": 1,
6   "severity": 3,
7   "source": "Wazuh",
8   "sourceRef": "sourceReference",
9   "status": "New",
10  "summary": "This is a summary",
11  "tags": [
12    "tag1",
13    "tag2"
14  ],
15  "title": "Alert Title",
16  "tlp": 2,
17  "type": "AlertType"
18 }
```

Great, it looks like it worked, now to check the SOC analyst station



The SOAR has forwarded the alerts automatically to the SOC analyst, AWESOME!



I can see the new alerts from the SOC Analyst dashboard, automatically created and forwarded using our SOAR & Wazuh/TheHive/VirusTotal

| <input type="checkbox"/> | STATUS | SEVERITY | TITLE | # CASE | TYPE | SOURCE | REFERENCE | DETAILS | ASSIGNEE | DATES | O. | C. | U. |
|--------------------------|--------|----------|-------------------|--------|-----------|------------|-----------------|---------------------|----------|-------|--|----|----|
| <input type="checkbox"/> | New | H | Alert Title | - | AlertType | Wazuh | sourceReference | Observables TTPs | 0 0 | ? | O. 18/08/2024 08:48 C. 18/08/2024 08:48 | | |
| | | | tag1 tag2 None | | | | | | | | | | |
| <input type="checkbox"/> | New | H | Alert Title | - | AlertType | sourceName | sourceReference | Observables TTPs | 0 0 | ? | O. 18/08/2024 08:46 C. 18/08/2024 08:46 | | |
| | | | tag1 tag2 None | | | | | | | | | | |

id ~2576616

Created by shuffle@test.com

Created at 18/08/2024 08:48

SEVERITY:HIGH

TLP:AMBER

PAP:GREEN

Assignee [Assign to me](#)

Unassigned

Source

Wazuh

Reference

sourceReference

Type

AlertType

Occurred date

18/08/2024 08:48

Status

New

General

Observ

Tags

tag1 tag2

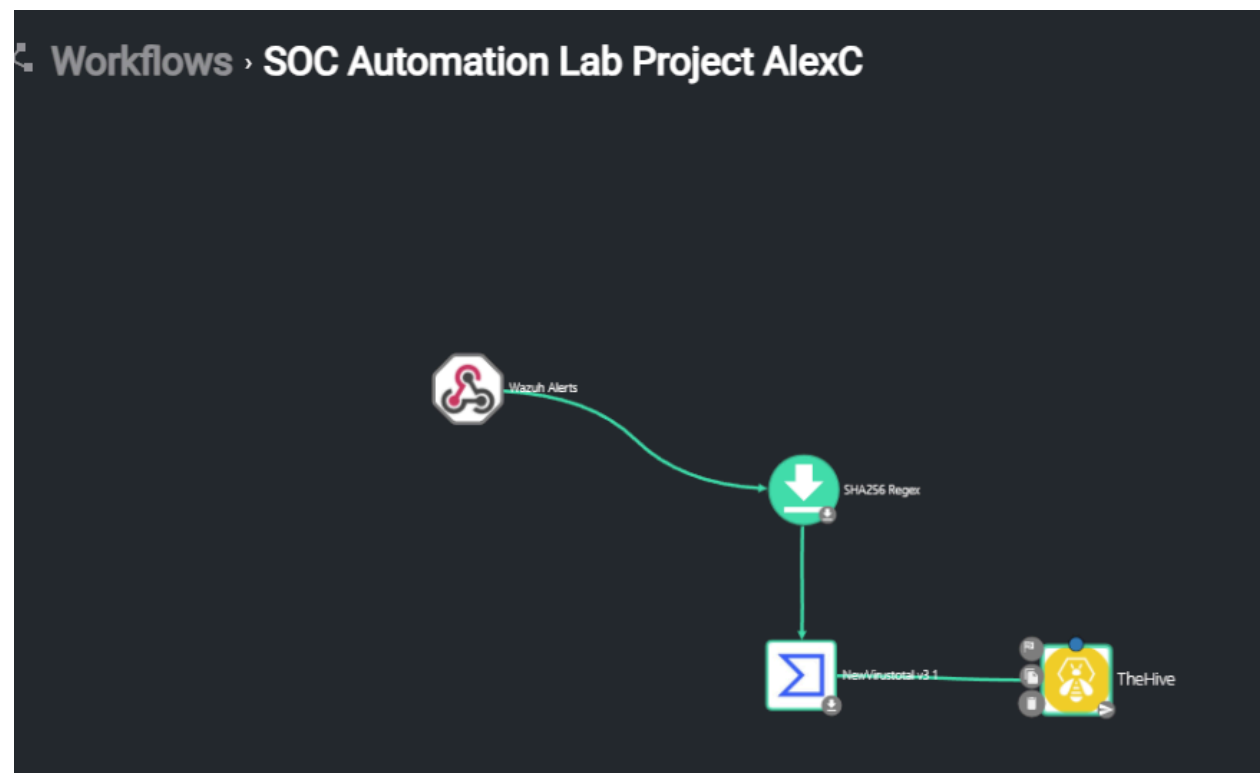
Description

MimikatsDetected

Summary

This is a summary

The workflow is working as intended!



In the next step, I'll add an automated email alert for the SOC Analyst

The screenshot shows a workflow in the SOC Analyst interface. On the left, a workflow diagram includes a 'SHA256 Regex' step, a 'NewVirusTotal v3.1' step, and 'Email 1'. On the right, the 'Email' configuration panel is open, showing the following details:

- Name:** EmailServer
- Delay:** 0
- Find Actions:** Send email shuffle
- Parameters:** (empty)
- Recipients:** (redacted)
- Subject:** WAKEUP! Mimikatz Detected
- Body:** \$exec.text.win.eventdata.utcti me

An email was sent now lets check if it was received

The screenshot shows the SOC Analyst interface with the 'SOCEmail' action highlighted. The status is 'SUCCESS'. The results for the action are as follows:

```
"Results for SOCEmail" : { 2 items
  "success" : true
  "reason" : "Email was successfully sent"
}
```

The variables section shows the following values:

- apikey:** (redacted)
- recipients:** (redacted)
- subject:** WAKEUP! Mimikatz Detected
- body:** (redacted)

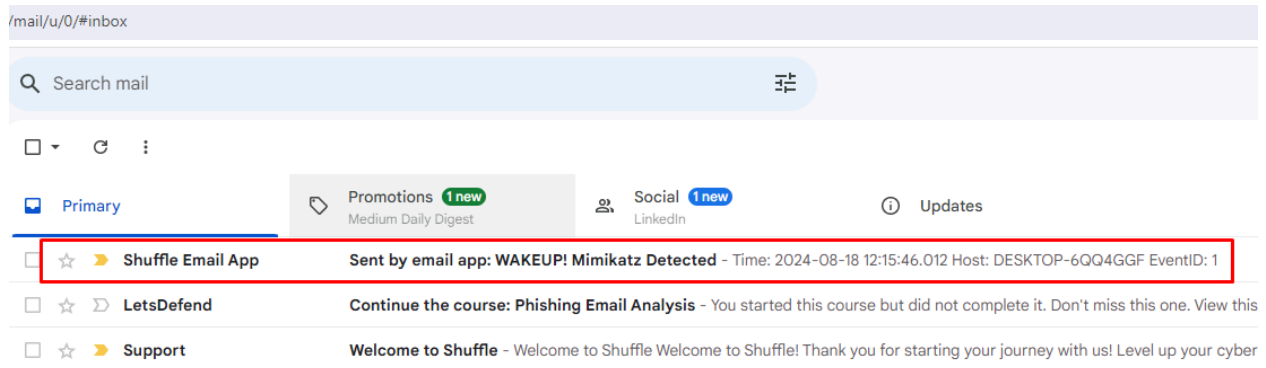
The action logs section shows the following message:

Logs for an action are not available without an onprem environment with the SHUFFLE_LOGS_DISABLED environment variable set to false: SHUFFLE_LOGS_DISABLED=false. Logs are enabled by default, except in scale mode.

The results list on the right shows the following items:

- SHA256 Regex (regex_capture_group) - 3 items
- NewVirusTotal v3.1 (get_a_hash_report_) - 6 items
- TheHive (post_create_alert) - 6 items
- SOCEmail (send_email_shuffle) - 2 items

Success, our SOC Analyst got an automated email for the mimikatz attack



Pretty cool! :)

Sent by email app: WAKEUP! Mimikatz Detected

Shuffle Email App <email-app@shuffler.io>

to me ▼

Time: 2024-08-18 12:15:46.012

Host: DESKTOP-6QQ4GGF

End of Project Summary

By the conclusion of this SOC Automation Project, I have successfully established a fully operational live telemetry ingesting and digesting environment. This project leverages a robust stack of security tools and systems to enhance SOC (Security Operations Center) capabilities through advanced automation and orchestration.

Components Implemented:

- **Wazuh (SIEM/XDR):** Deployed as the core SIEM and XDR solution, Wazuh efficiently collects, analyzes, and correlates security data. I have tailored custom rules and configurations to adapt to specific use cases, ensuring comprehensive monitoring and detection of threats.
- **Shuffle (SOAR):** Integrated Shuffle to automate security workflows and orchestrate responses across the security stack. This includes the creation of automated playbooks and incident response processes, streamlining the handling of security alerts and reducing manual intervention.
- **The Hive (Case Management System):** Implemented The Hive to manage and track security incidents. This system facilitates detailed case management, including alert aggregation, investigation tracking, and collaboration among SOC analysts.

Real-Time Alerting:

Configured the system to send real-time alerts to SOC analysts via email & live updated dashboards, ensuring that critical security events are promptly communicated and addressed. This real-time notification system is essential for rapid incident response and effective security posture management.

Live Testing and Customization:

- **Mimikatz:** Employed Mimikatz, validating the effectiveness of Wazuh's detection capabilities and refining custom rules based on the test results. This hands-on approach enhanced the SIEM's resilience.
- **Sysmon Log Implementation:** Conducted analysis of Sysmon logs to enhance threat detection & incident investigation. The analysis contributed to the development of more precise detection rules and improved overall system accuracy.

Skills learned:

- **Automation and Orchestration:** Successfully implemented advanced automation and orchestration techniques, significantly enhancing operational efficiency and response times within the SOC environment.
- **Custom Rule Development:** Created and optimized custom rules within Wazuh, tailored to the specific needs of the organization and the threats identified during testing.
- **Incident Management:** Established effective case management practices with The Hive, ensuring thorough documentation and tracking of security incidents.
- **Integration and Testing:** Demonstrated proficiency in integrating multiple security tools and performing rigorous live testing to ensure system reliability and effectiveness.