Reverse Engineering & Malware Analysis Training

Part 5 – Reverse Engineering Tools Basics

Swapnil Pathak

www.SecurityXploded.com

Disclaimer

The Content, Demonstration, Source Code and Programs presented here is "AS IS" without any warranty or conditions of any kind. Also the views/ideas/knowledge expressed here are solely of the trainer’s only and nothing to do with the company or the organization in which the trainer is currently working.

However in no circumstances neither the trainer nor SecurityXploded is responsible for any damage or loss caused due to use or misuse of the information presented here.

Acknowledgement

- Special thanks to **null** & **Garage4Hackers** community for their extended support and cooperation.
- Thanks to all the **Trainers** who have devoted their precious time and countless hours to make it happen.

Reversing & Malware Analysis Training

This presentation is part of our **Reverse Engineering & Malware Analysis** Training program. Currently it is delivered only during our local meet for FREE of cost.

For complete details of this course, visit our Security Training page.

Who am I

Swapnil Pathak

- Member SecurityXploded
- Security Researcher
- RE, Malware Analysis, Network Security
- Email: swapnilpathak101@gmail.com

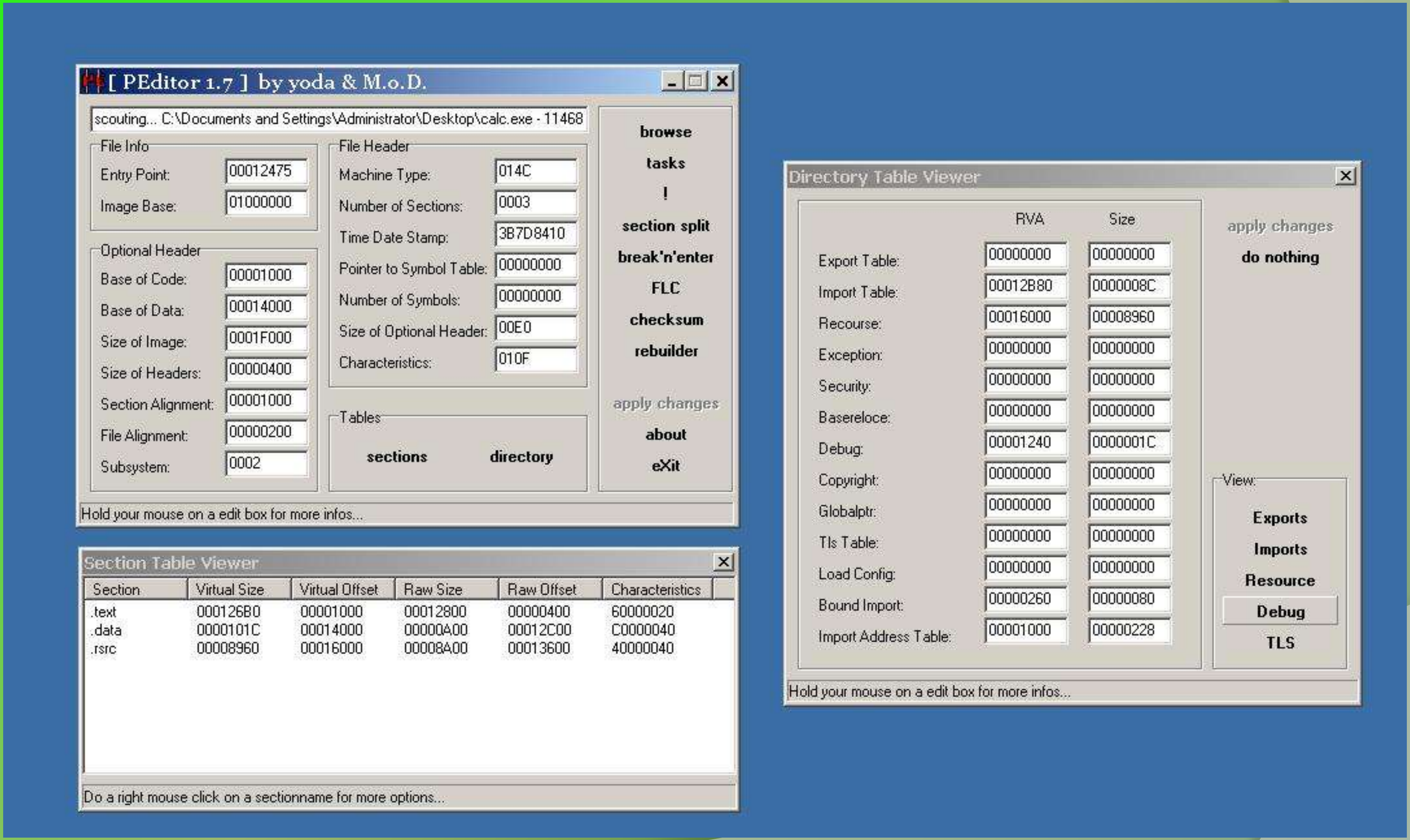
Presentation Outline

- PE Tools
  - PE Editor
- Disassemblers
  - IDA Pro
- Debuggers
  - OllyDbg
- Other Useful Tools

PE Tools

- Portable Executable Editor
  - Allow user to view, edit data structures present in PE file format
- Some Common Use Cases:
  - Change entry point of the executable
  - View Import/Export/Sections within EXE
  - Change characteristics of a file i.e. Dll to Exe
  - Fix anomalies of PE files
- PE editors
  - Hiew, PE Editor, CFF Explorer, StudPE, LordPE etc

PE Editor



[PEditor 1.7] by yoda & M.o.D.

scouting... C:\Documents and Settings\Administrator\Desktop\calc.exe - 11468

File Info

Entry Point:	00012475
Image Base:	01000000

Optional Header

Base of Code:	00001000
Base of Data:	00014000
Size of Image:	0001F000
Size of Headers:	00000400
Section Alignment:	00001000
File Alignment:	00000200
Subsystem:	0002

File Header

Machine Type:	014C
Number of Sections:	0003
Time Date Stamp:	3B7D8410
Pointer to Symbol Table:	00000000
Number of Symbols:	00000000
Size of Optional Header:	00E0
Characteristics:	010F

Hold your mouse on a edit box for more infos...

Section Table Viewer

Section	Virtual Size	Virtual Offset	Raw Size	Raw Offset	Characteristics
.text	000126B0	00001000	00012800	00000400	60000020
.data	0000101C	00014000	00000A00	00012C00	C0000040
.rsrc	00008960	00016000	00008A00	00013600	40000040

Do a right mouse click on a sectionname for more options...

Directory Table Viewer

	RVA	Size
Export Table:	00000000	00000000
Import Table:	00012B80	0000008C
Recourse:	00016000	00008960
Exception:	00000000	00000000
Security:	00000000	00000000
Basereloce:	00000000	00000000
Debug:	00001240	0000001C
Copyright:	00000000	00000000
Globalptr:	00000000	00000000
Tls Table:	00000000	00000000
Load Config:	00000000	00000000
Bound Import:	00000260	00000080
Import Address Table:	00001000	00000228

Hold your mouse on a edit box for more infos...

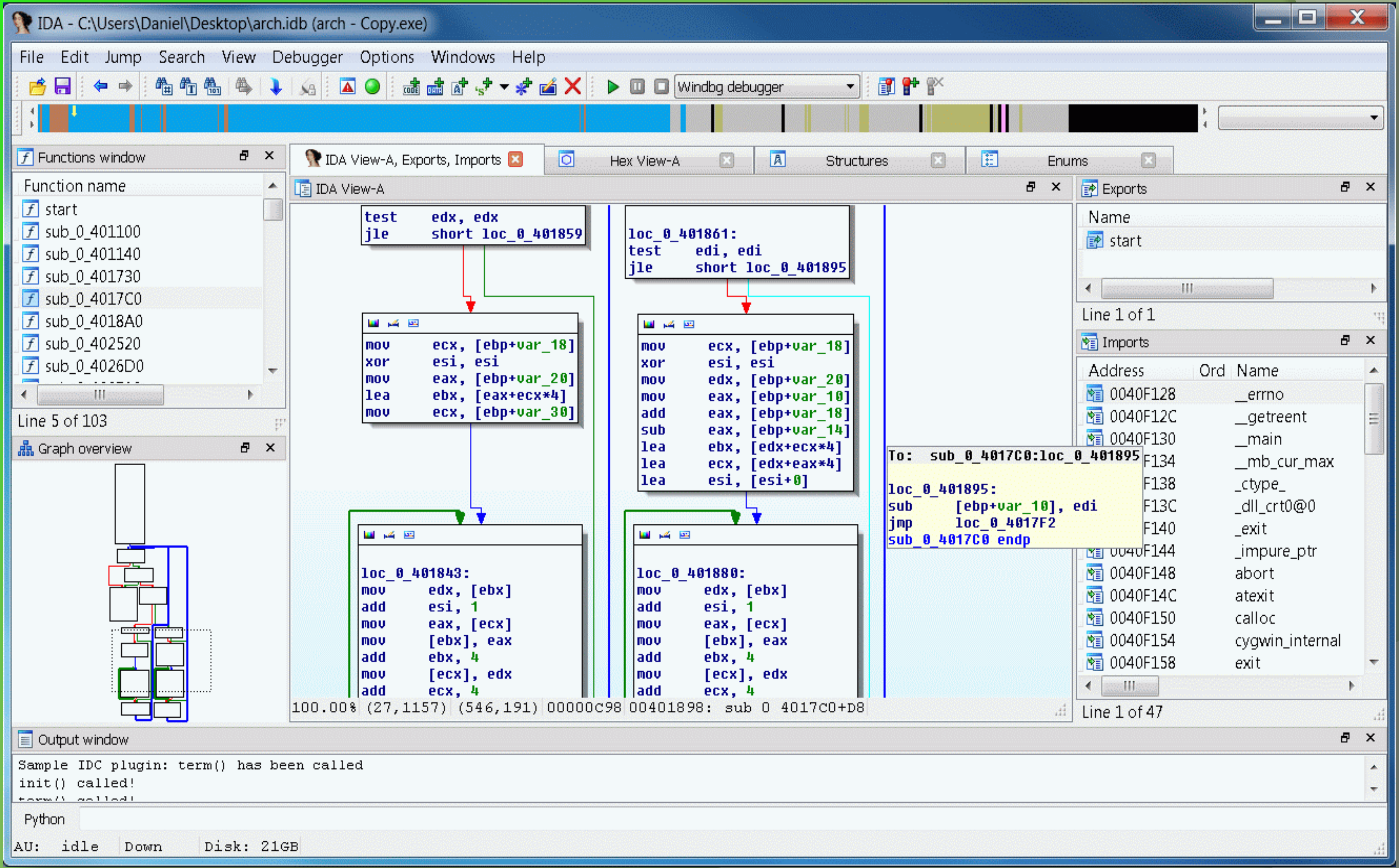
IDA Pro

- Popular commercial software used for Reverse Engineering.
- Disassembler and Debugger in one tool
- Supports variety of executable formats for different processors and OS
- Provides both Text & Graphical view of the code flow.
- View strings, Imports, Exports referenced in the executable
- Support Plugins
- Some of the useful plugins
  - X86 emulator
  - IDAPython
  - IDARub
  - IDA Scripts
  - Windbg

IDA Pro Functions

- IDA Windows & purpose – IDA View, Hex View, Imports, Strings, Functions windows etc.
- Graphs & Text view (use “space” to switch between views)
- Xref To & Xref From – powerful feature
- Jump & Search
- Edit function names (N), Add comments (;)

IDA Pro in Action



IDA - C:\Users\Daniel\Desktop\arch.idb (arch - Copy.exe)
File Edit Jump Search View Debugger Options Windows Help
Windbg debugger
Functions window
IDA View-A, Exports, Imports
Hex View-A
Structures
Enums
IDA View-A
Exports
Imports
Graph overview
Output window
Python

Function name
- start
- sub_0_401100
- sub_0_401140
- sub_0_401730
- sub_0_4017C0
- sub_0_4018A0
- sub_0_402520
- sub_0_4026D0

Line 5 of 103

```
test    edx, edx
jle     short loc_0_401859
```

```
mov     ecx, [ebp+var_18]
xor     esi, esi
mov     eax, [ebp+var_20]
lea     ebx, [eax+ecx*4]
mov     ecx, [ebp+var_30]
```

```
loc_0_401843:
mov     edx, [ebx]
add     esi, 1
mov     eax, [ecx]
mov     [ebx], eax
add     ebx, 4
mov     [ecx], edx
add     ecx, 4
```

```
loc_0_401861:
test    edi, edi
jle     short loc_0_401895
```

```
mov     ecx, [ebp+var_18]
xor     esi, esi
mov     edx, [ebp+var_20]
mov     eax, [ebp+var_10]
add     eax, [ebp+var_18]
sub     eax, [ebp+var_14]
lea     ebx, [edx+ecx*4]
lea     ecx, [edx+eax*4]
lea     esi, [esi+0]
```

```
loc_0_401880:
mov     edx, [ebx]
add     esi, 1
mov     eax, [ecx]
mov     [ebx], eax
add     ebx, 4
mov     [ecx], edx
add     ecx, 4
```

```
To:  sub_0_4017C0:loc_0_401895

loc_0_401895:
sub     [ebp+var_10], edi
jmp     loc_0_4017F2
sub_0_4017C0 endp
```

```
100.00% (27,1157) (546,191) 00000C98 00401898: sub_0_4017C0+D8
```

Exports

Name
start

Line 1 of 1

Imports

Address	Ord	Name
0040F128		_errno
0040F12C		_getreent
0040F130		__main
F134		__mb_cur_max
F138		_ctype_
F13C		_dll_crt0@0
F140		_exit
0040F144		_impure_ptr
0040F148		abort
0040F14C		atexit
0040F150		calloc
0040F154		cygwin_internal
0040F158		exit

Line 1 of 47

```
Sample IDC plugin: term() has been called
init() called!
term() called!
```

```
AU:  idle   Down    Disk: 21GB
```

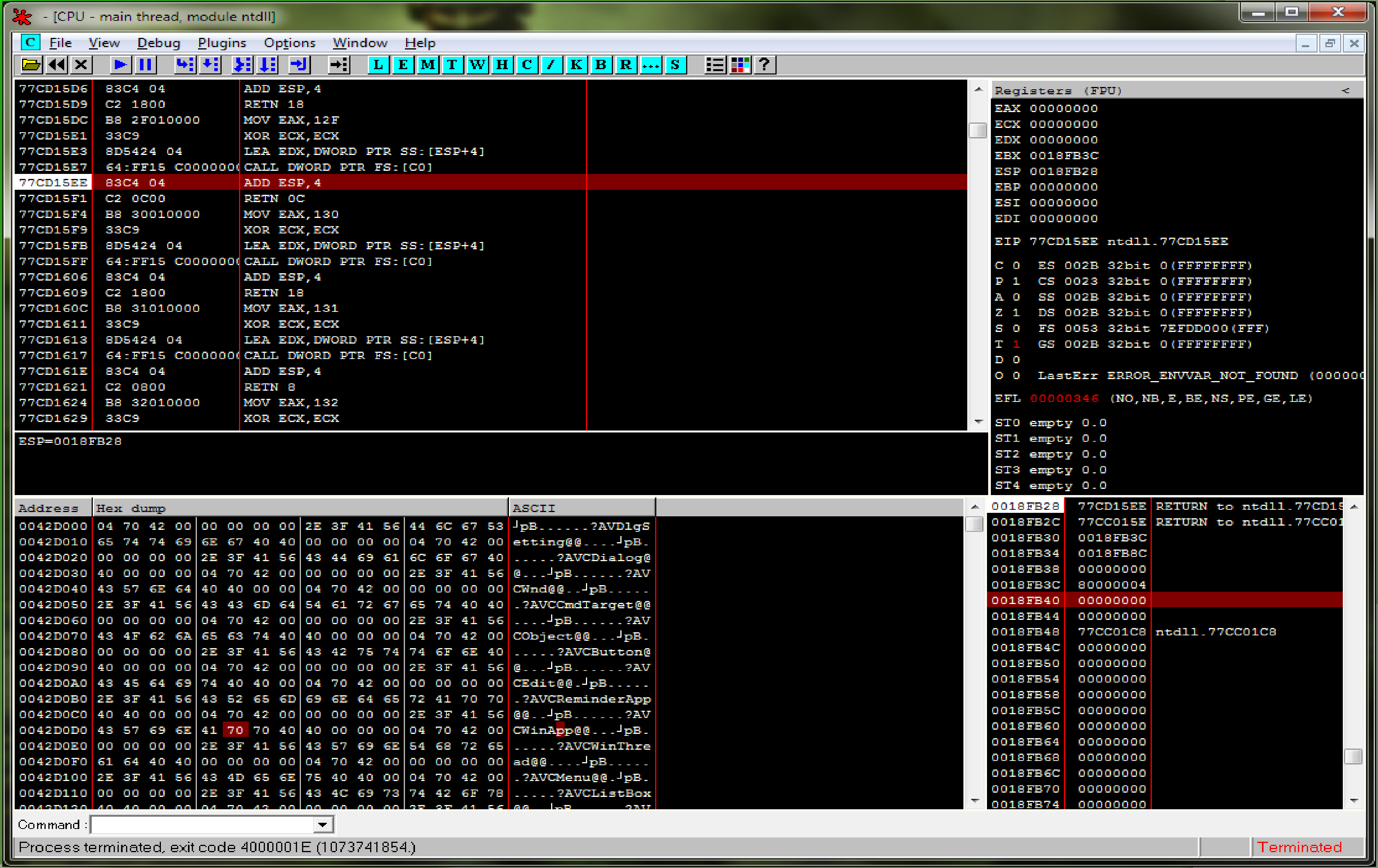
Ollydbg

- Most Popular Ring 3 Debugger used in Reversing
- Provides the below mentioned functionalities
  - Debugging program step by step (Single Stepping)
  - Software, Hardware and Memory based Breakpoints
  - Examine the current state of the program wrt variables, memory etc.
  - Change the flow or state of Program by directly editing Instructions, Registers or Memory.
- Support Plugins, Here are popular ones,
  - OllyDump
  - OllyAdvanced
  - OllyScript

Ollydbg Cont.

- Different Windows – CPU, Registers, Dump, Stack screens
- Trace Into, Trace Over, Exceptions
- Integrate windows API help file in ollydbg
- Shortcuts (imp. Only)
  - F7 – Step into [call]
  - F8 – Single step [execute call]
  - F9 – Run
  - F2 – Breakpoint

Ollydbg in Action



- [CPU - main thread, module ntdll]
C File View Debug Plugins Options Window Help
L E M T W H C / K B R ... S
?

```
77CD15D6  83C4 04          ADD ESP,4
77CD15D9  C2 1800          RETN 18
77CD15DC  B8 2F010000      MOV EAX,12F
77CD15E1  33C9             XOR ECX,ECX
77CD15E3  8D5424 04        LEA EDX,DWORD PTR SS:[ESP+4]
77CD15E7  64:FF15 C000000( CALL DWORD PTR FS:[C0]
77CD15EE  83C4 04          ADD ESP,4
77CD15F1  C2 0C00          RETN 0C
77CD15F4  B8 30010000      MOV EAX,130
77CD15F9  33C9             XOR ECX,ECX
77CD15FB  8D5424 04        LEA EDX,DWORD PTR SS:[ESP+4]
77CD15FF  64:FF15 C000000( CALL DWORD PTR FS:[C0]
77CD1606  83C4 04          ADD ESP,4
77CD1609  C2 1800          RETN 18
77CD160C  B8 31010000      MOV EAX,131
77CD1611  33C9             XOR ECX,ECX
77CD1613  8D5424 04        LEA EDX,DWORD PTR SS:[ESP+4]
77CD1617  64:FF15 C000000( CALL DWORD PTR FS:[C0]
77CD161E  83C4 04          ADD ESP,4
77CD1621  C2 0800          RETN 8
77CD1624  B8 32010000      MOV EAX,132
77CD1629  33C9             XOR ECX,ECX
```

```
ESP=0018FB28
```

```
Registers (FPU)
EAX 00000000
ECX 00000000
EDX 00000000
EBX 0018FB3C
ESP 0018FB28
EBP 00000000
ESI 00000000
EDI 00000000

EIP 77CD15EE ntdll.77CD15EE

C 0  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 0  SS 002B 32bit 0(FFFFFFFF)
Z 1  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 7EFDD000(FFF)
T 1  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr ERROR_ENVVAR_NOT_FOUND (000000

EFL 00000346 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
```

```
Address   Hex dump                                         ASCII
0042D000  04 70 42 00 00 00 00 00 2E 3F 41 56 44 6C 67 53  ┘pB......?AVDlgS
0042D010  65 74 74 69 6E 67 40 40 00 00 00 00 04 70 42 00  etting@@....┘pB.
0042D020  00 00 00 00 2E 3F 41 56 43 44 69 61 6C 6F 67 40  .....?AVCDialog@
0042D030  40 00 00 00 04 70 42 00 00 00 00 00 2E 3F 41 56  @...┘pB......?AV
0042D040  43 57 6E 64 40 40 00 00 04 70 42 00 00 00 00 00  CWnd@@..┘pB.....
0042D050  2E 3F 41 56 43 43 6D 64 54 61 72 67 65 74 40 40  .?AVCCmdTarget@@
0042D060  00 00 00 00 04 70 42 00 00 00 00 00 2E 3F 41 56  ....┘pB......?AV
0042D070  43 4F 62 6A 65 63 74 40 40 00 00 00 04 70 42 00  CObject@@...┘pB.
0042D080  00 00 00 00 2E 3F 41 56 43 42 75 74 74 6F 6E 40  .....?AVCButton@
0042D090  40 00 00 00 04 70 42 00 00 00 00 00 2E 3F 41 56  @...┘pB......?AV
0042D0A0  43 45 64 69 74 40 40 00 04 70 42 00 00 00 00 00  CEdit@@.┘pB.....
0042D0B0  2E 3F 41 56 43 52 65 6D 69 6E 64 65 72 41 70 70  .?AVCReminderApp
0042D0C0  40 40 00 00 04 70 42 00 00 00 00 00 2E 3F 41 56  @@..┘pB......?AV
0042D0D0  43 57 69 6E 41 70 70 40 40 00 00 00 04 70 42 00  CWinApp@@...┘pB.
0042D0E0  00 00 00 00 2E 3F 41 56 43 57 69 6E 54 68 72 65  .....?AVCWinThre
0042D0F0  61 64 40 40 00 00 00 00 04 70 42 00 00 00 00 00  ad@@....┘pB.....
0042D100  2E 3F 41 56 43 4D 65 6E 75 40 40 00 04 70 42 00  .?AVCMenu@@.┘pB.
0042D110  00 00 00 00 2E 3F 41 56 43 4C 69 73 74 42 6F 78  .....?AVCListBox
```

```
0018FB28   77CD15EE  RETURN to ntdll.77CD15
0018FB2C   77CC015E  RETURN to ntdll.77CC01
0018FB30   0018FB3C
0018FB34   0018FB8C
0018FB38   00000000
0018FB3C   80000004
0018FB40   00000000
0018FB44   00000000
0018FB48   77CC01C8  ntdll.77CC01C8
0018FB4C   00000000
0018FB50   00000000
0018FB54   00000000
0018FB58   00000000
0018FB5C   00000000
0018FB60   00000000
0018FB64   00000000
0018FB68   00000000
0018FB6C   00000000
0018FB70   00000000
0018FB74   00000000
```

Command :

Process terminated, exit code 4000001E (1073741854.)

Terminated

UPX Unpacking Ollyscript

Here is example for Unpacking UPX based Malwares using OllyScript in OllyDbg

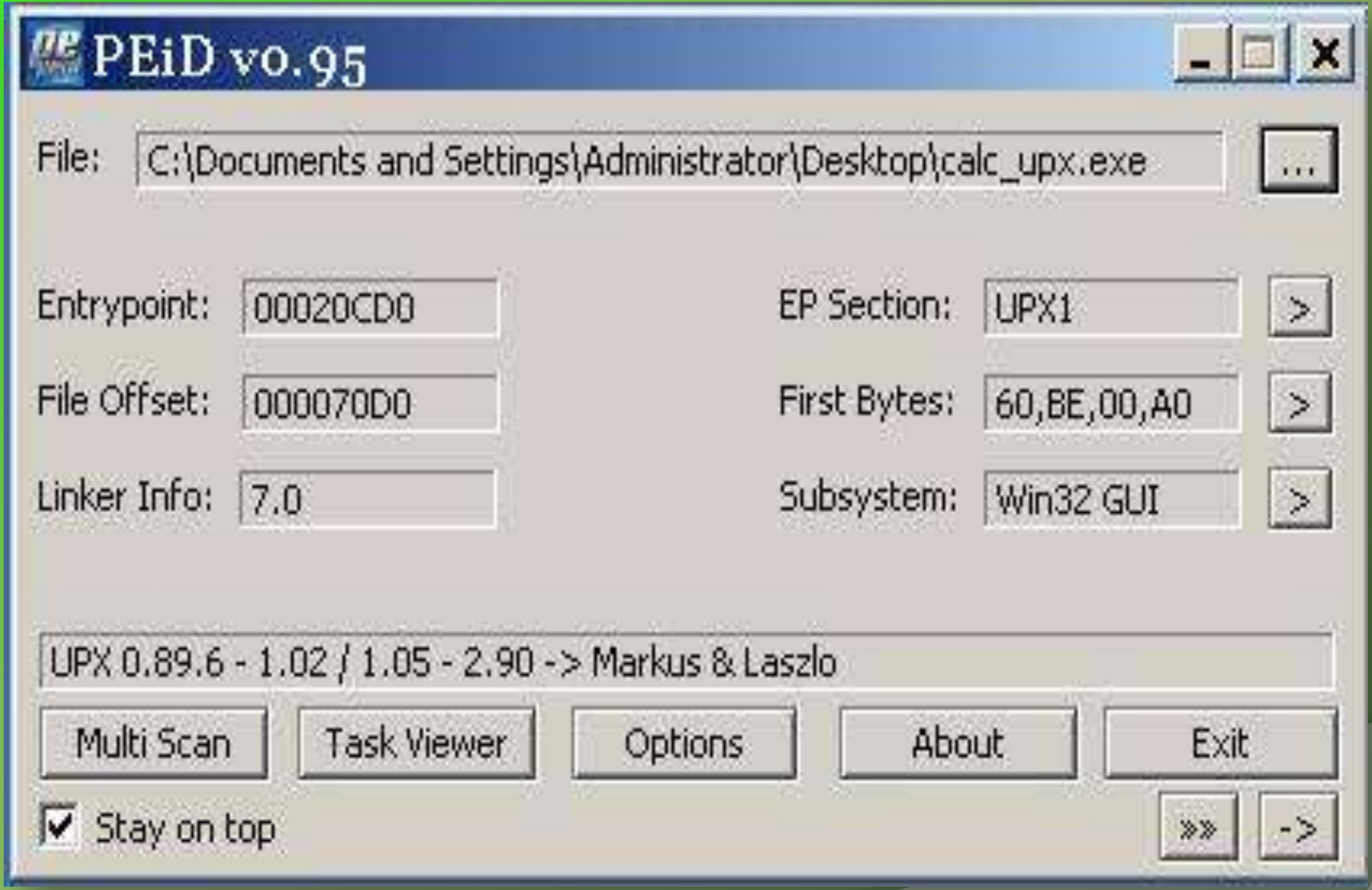
```
var hwdBP                       // Local variable to store hardware breakpoint
var softBP                      // Local variable to strore software breakpoint
sti                             // Step into F7 command
findop eip, #61#                // find next POPAD
mov hwdBP, $RESULT               // Store $RESULT to hardware breakpoint local variable
bphws hwdBP, "x“                 // Set hardware breakpoint (execute) on the next POPAD
run                             // Run F9 command
findop eip, #E9????????#        // Find the next JMP
mov softBP, $RESULT              // Store $RESULT to software breakpoint local variable
bp softBP
run                             // Run to JMP instruction
sti                             // Step into the OEP
cmt eip, "<<>>"
msg "OEP found, you can dump the file starting from this address"
ret
```

Source : http://x9090.blogspot.in/2009/07/ollyscript-tutorial-unpack-upx.html

Useful Tools

- Packer Identifier Tools
  - RDG packer detector
  - PEID
  - ExeScan

PEiD – PE Packer Identifier Tool



File: C:\Documents and Settings\Administrator\Desktop\calc_upx.exe

Entrypoint: 00020CD0

File Offset: 000070D0

Linker Info: 7.0

EP Section: UPX1

First Bytes: 60,BE,00,A0

Subsystem: Win32 GUI

UPX 0.89.6 - 1.02 / 1.05 - 2.90 -> Markus & Laszlo

Reference

- Complete Reference Guide for Reversing & Malware Analysis Training

Thank You !

www.SecurityXploded.com