UNPROTECTING THE CRYPTER

A GENERIC APPROACH

Author :Arunpreet Singh

Blog : http://reverse2learn.wordpress.com

Tools Used

1)OllyDbg

2)Process Explorer

3)PUPE

4)PE Tools

5)Hex WorkShop

Crypter

So what is a Crypter.If   have some experience in malware Field then You must have Heard about tool called “Crypter”or may be used it.The Aim of Crypter to Protect the executables ,making difficult to analyze it or reverse engineer it .But Mostly in Malware Scene the crypters are mainly used to make malwares FUD .Here FUD stands for Fully Undetectable.

Actually the malware are basically distributed as executables ,I mean sources are gernally not available. Public malwares are gernally detected by antivurses ,that’s why crypters are used to make them FUD .

How Crypters Work .

Principle for making a crypter is very simple . Crypter Consist of Two parts

1)Builder

2)Stub

How they both parts work

1)You give your file as input to crypter,it encrypts it with any encryption algorithm (most likely RC4,AES)

By encrypting the file it defeat the static analysis done by antivirus.During static analysis the antivirises try to find the the patterns in executable and match with with signatures.Because the file is encrypted

So the antivirus can’t find patterns here.

2)Add the stub before the executable code.

When you run executable then the stub runs and decrypt the encrypted file .

Note :The decrypted file remains in memory .

3)Execute the Decrypted from Memory .

This is actually the heart of crypter.This is also called "Run PE ".There are different methods for Run PE .But Mostly the Crypter used a public method   to exectute the File from Memory ,that's what we are going to target.

Let me Explain the the method .The orginal link of this method is

http://www.security.org.sg/code/loadexe.html

I just copying the steps .i realy suggest you to once read the whole article to understand in more depth.

The steps listed in article are :

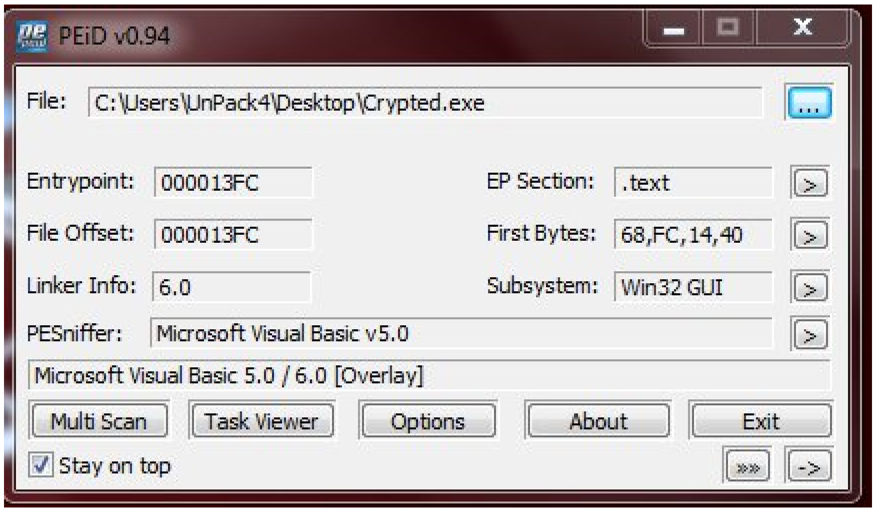
1) Use the CreateProcess API with the CREATE_SUSPENDED parameter to create a suspended process from any EXE file. (Call this the *first* EXE).
2) Call GetThreadContext API to obtain the register values (thread context) of the suspended process. The EBX register of the suspended process points to the process's PEB. The EAX register contains the entry point of the process (first EXE)
3) Obtain the base-address of the suspended process from its PEB, i.e. at [EBX+8]
4) Load the second EXE into memory (using ReadFile) and perform the neccessary alignment manually. This is required if the file alignment is different from the memory alignment
5) If the second EXE has the same base-address as the suspended process and its image-size is <= to the image-size of the suspended process, simply use the WriteProcessMemory function to write the image of the second EXE into the memory space of the suspended process, starting at the base-address
6) Otherwise, unmap the image of the first EXE using ZwUnmapViewOfSection (exported by ntdll.dll) and use VirtualAllocEx to allocate enough memory for the second EXE within the memory space of the suspended process. The VirtualAllocEx API must be supplied with the base-address of the second EXE to ensure that Windows will give us memory in the required region. Next, copy the image of the second EXE into the memory space of the suspended process starting at the allocated address (using WriteProcessMemory)

7) Patch the base-address of the second EXE into the suspended process's PEB at [EBX+8]
8) Set EAX of the thread context to the entry point of the second EXE
9) Use the SetThreadContext API to modify the thread context of the suspended process
10) Use the ResumeThread API to resume execute of the suspended process.

When you normally load a packed executable in ollydbg then it shows warning like "the code section is compressed" or "the entrypoint is outside the code section " whatever means olly give you hint that the executable is packed.But the executable crypted by crypter (which is using above method) never shows any warning when it is loaded into olly it does not show any warning .

Unpacking

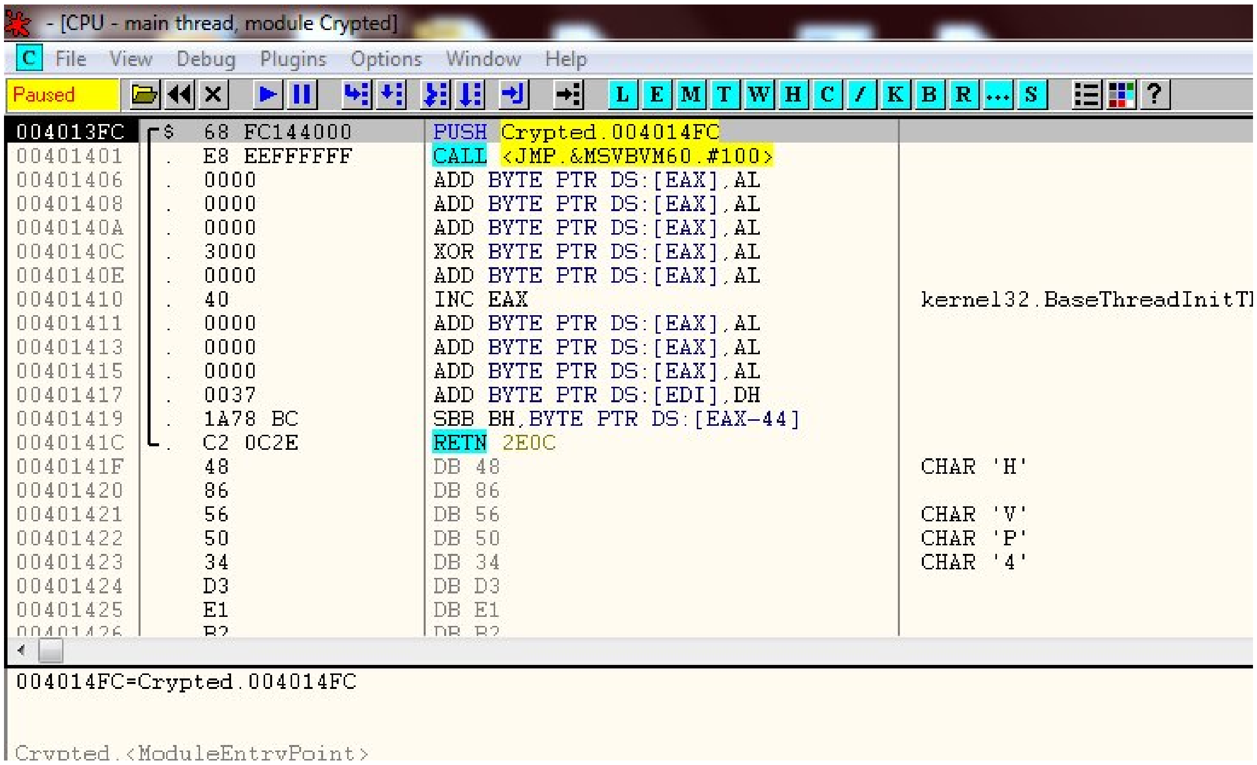
Scan it with PEID .



```
PEiD v0.94
File: C:\Users\UnPack4\Desktop\Crypted.exe
Entrypoint: 000013FC        EP Section: .text
File Offset: 000013FC       First Bytes: 68,FC,14,40
Linker Info: 6.0            Subsystem: Win32 GUI
PESniffer: Microsoft Visual Basic v5.0
Microsoft Visual Basic 5.0 / 6.0 [Overlay]
```

Looks Inocent :P

Lets Load it in Olly ..see it shows any warning or not

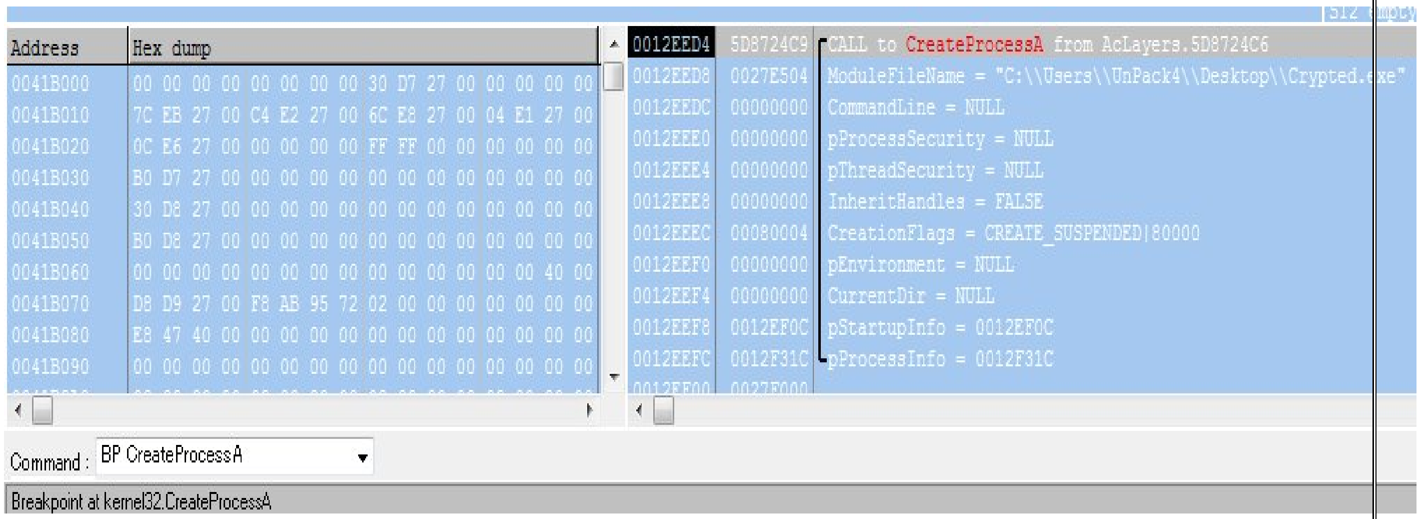

- [CPU - main thread, module Crypted]
File View Debug Plugins Options Window Help
Paused
L E M T W H C / K B R ... S

```
004013FC  $ 68 FC144000    PUSH Crypted.004014FC
00401401  . E8 EEFFFFFF    CALL <JMP.&MSVBVM60.#100>
00401406  . 0000           ADD BYTE PTR DS:[EAX],AL
00401408  . 0000           ADD BYTE PTR DS:[EAX],AL
0040140A  . 0000           ADD BYTE PTR DS:[EAX],AL
0040140C  . 3000           XOR BYTE PTR DS:[EAX],AL
0040140E  . 0000           ADD BYTE PTR DS:[EAX],AL
00401410  . 40             INC EAX                     kernel32.BaseThreadInitTl
00401411  . 0000           ADD BYTE PTR DS:[EAX],AL
00401413  . 0000           ADD BYTE PTR DS:[EAX],AL
00401415  . 0000           ADD BYTE PTR DS:[EAX],AL
00401417  . 0037           ADD BYTE PTR DS:[EDI],DH
00401419  . 1A78 BC        SBB BH,BYTE PTR DS:[EAX-44]
0040141C  . C2 0C2E        RETN 2E0C
0040141F    48             DB 48                       CHAR 'H'
00401420    86             DB 86
00401421    56             DB 56                       CHAR 'V'
00401422    50             DB 50                       CHAR 'P'
00401423    34             DB 34                       CHAR '4'
00401424    D3             DB D3
00401425    E1             DB E1

004014FC=Crypted.004014FC

Crypted.<ModuleEntryPoint>
```

Everthing Looking normal,Looks Like a normal VB excutable no warning shown by olly

First Verify If our target is realy innocent or malicious.Acc. to method described above it must call Create a new process.So Put a BP on CreateProcessA and CreateProcessW (for both ascii and unicode versions).If it Breaks then see the arguments passed check if it is in SUSPENDED MODE (Also You can Put Breakpoint on ReadProcessMemory and WriteProcessMemory APIs to check it more accurately)

I Put BP on CreateProcessW and CreateProcessA and run it in olly.As you can see this it is Breaked at CreateProcessA..Also You can see it parameters in stack ,also you can see that it is in SUSPENDED_MODE .



```
Address   Hex dump
0041B000  00 00 00 00 00 00 00 00 30 D7 27 00 00 00 00 00
0041B010  7C EB 27 00 C4 E2 27 00 6C E8 27 00 04 E1 27 00
0041B020  0C E6 27 00 00 00 00 00 FF FF 00 00 00 00 00 00
0041B030  B0 D7 27 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B040  30 D8 27 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B050  B0 D8 27 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 40 00
0041B070  D8 D9 27 00 F8 AB 95 72 02 00 00 00 00 00 00 00
0041B080  E8 47 40 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

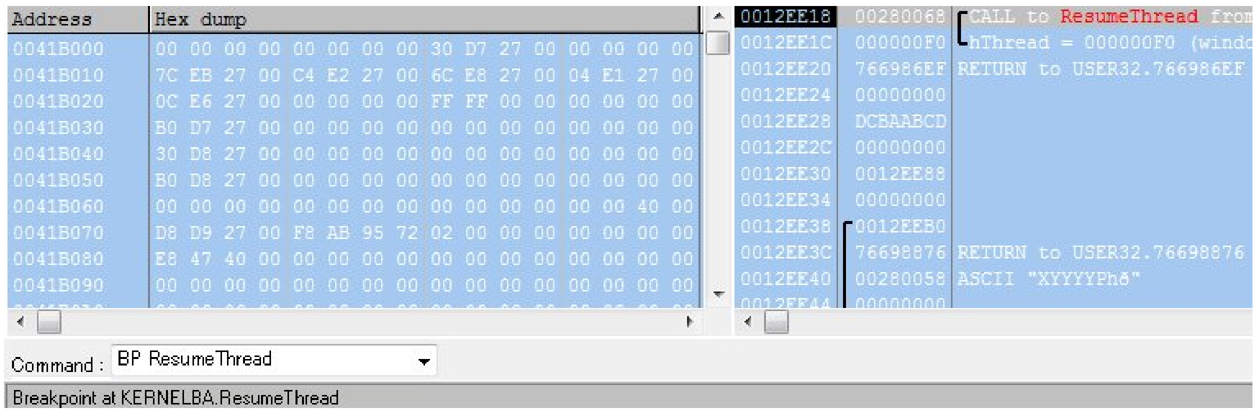
```
0012EED4  5D8724C9  CALL to CreateProcessA from AcLayers.5D8724C6
0012EED8  0027E504  ModuleFileName = "C:\\Users\\UnPack4\\Desktop\\Crypted.exe"
0012EEDC  00000000  CommandLine = NULL
0012EEE0  00000000  pProcessSecurity = NULL
0012EEE4  00000000  pThreadSecurity = NULL
0012EEE8  00000000  InheritHandles = FALSE
0012EEEC  00080004  CreationFlags = CREATE_SUSPENDED|80000
0012EEF0  00000000  pEnvironment = NULL
0012EEF4  00000000  CurrentDir = NULL
0012EEF8  0012EF0C  pStartupInfo = 0012EF0C
0012EEFC  0012F31C  pProcessInfo = 0012F31C
```

```
Command : BP CreateProcessA
Breakpoint at kernel32.CreateProcessA
```

It calles the CreateProcess In suspended mode(suspend its main thread) then decrypt the encrypted malware in newly created process address space when everything is on its place then it calls the ResumeThread API and it start running

We are going to attack at the point when It calles the ResumeThread API,because ResumeThread API is last step in executaion and before this everthing will be on its place .

So I Put BP on ResumeThread,Lets See what Happens



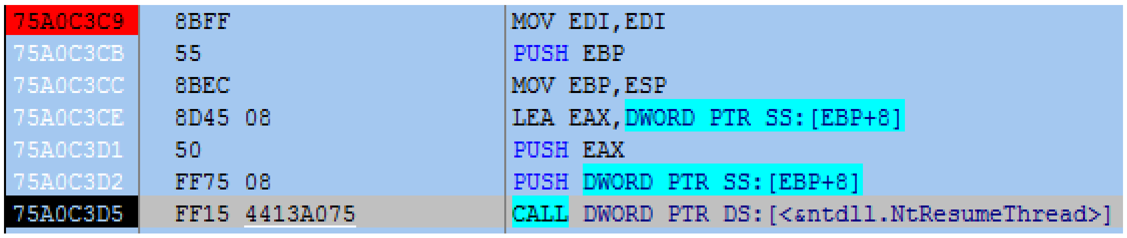
```
Address   Hex dump
0041B000  00 00 00 00 00 00 00 00 30 D7 27 00 00 00 00 00
0041B010  7C EB 27 00 C4 E2 27 00 6C E8 27 00 04 E1 27 00
0041B020  0C E6 27 00 00 00 00 00 FF FF 00 00 00 00 00 00
0041B030  B0 D7 27 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B040  30 D8 27 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B050  B0 D8 27 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 40 00
0041B070  D8 D9 27 00 F8 AB 95 72 02 00 00 00 00 00 00 00
0041B080  E8 47 40 00 00 00 00 00 00 00 00 00 00 00 00 00
0041B090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
0012EE18  00280068  CALL to ResumeThread from
0012EE1C  000000F0  hThread = 000000F0 (windo
0012EE20  766986EF  RETURN to USER32.766986EF
0012EE24  00000000
0012EE28  DCBAABCD
0012EE2C  00000000
0012EE30  0012EE88
0012EE34  00000000
0012EE38  0012EEB0
0012EE3C  76698876  RETURN to USER32.76698876
0012EE40  00280058  ASCII "XYYYYPh8"
0012EE44  00000000
```

```
Command : BP ResumeThread
Breakpoint at KERNELBA.ResumeThread
```

Wow Its Breaked on ResumeThread..

Now Step Into ResumeThread by Pressing F7.



```
75A0C3C9   8BFF             MOV EDI,EDI
75A0C3CB   55               PUSH EBP
75A0C3CC   8BEC             MOV EBP,ESP
75A0C3CE   8D45 08          LEA EAX,DWORD PTR SS:[EBP+8]
75A0C3D1   50               PUSH EAX
75A0C3D2   FF75 08          PUSH DWORD PTR SS:[EBP+8]
75A0C3D5   FF15 4413A075    CALL DWORD PTR DS:[<&ntdll.NtResumeThread>]
```

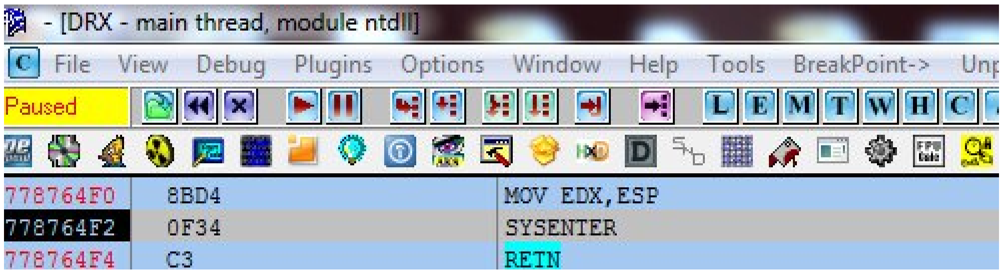
As You can see that ResumeThread internally calls window native api NtResumeThread

NOTE: NtResumeThread is Undocumneted native API . Most of windows API works this way .They provide a documented interface for main function then internally called the undocumented native APIs.This Concept is very Important Because Sometime the Crypter authors uses undocumented native APIs instead of Documented APIs.

For example they can directly use NtResumeThread instead of calling ResumeThread.In this way if you put BP on ResumeThread then it will not break .So I strongly suggest you to put breakpoint on native undocumented APIs instead of Documented APIs.

For example always put BP on NtResumeThread instead of ResumeThread ,then you will directly break at 75A0C3D5 instead of 75A0C3C9.

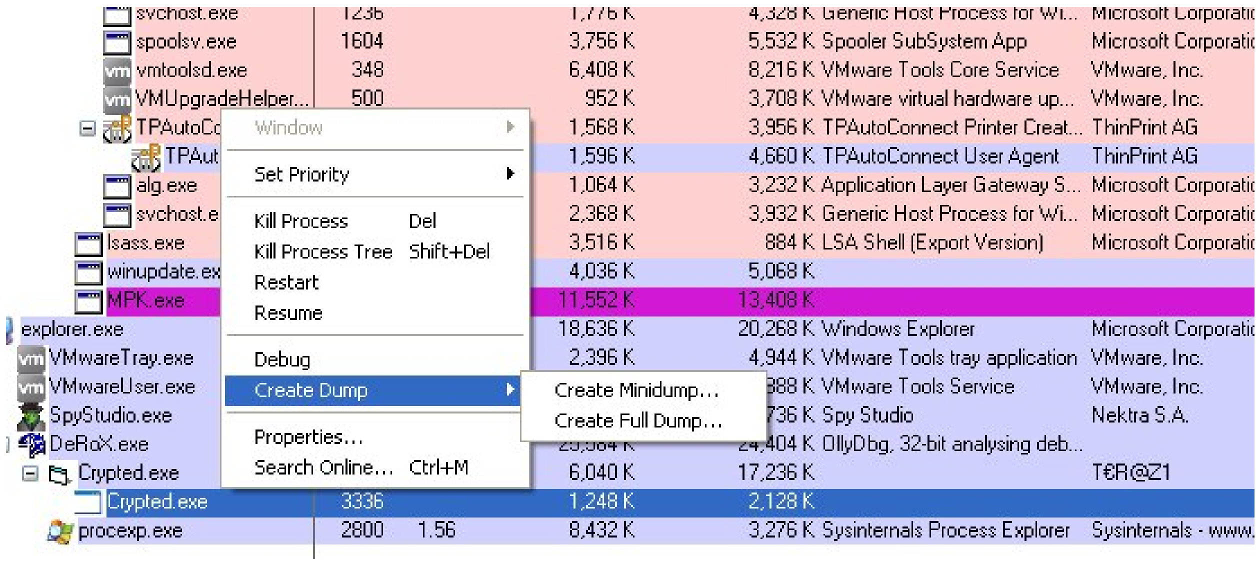
Lets Step inside NtResumeThread. By pressing F7.Contnue pressing F7 until you reach it 778764F2


- [DRX - main thread, module ntdll]
File View Debug Plugins Options Window Help Tools BreakPoint-> Unp
Paused
L E M T W H C

```
778764F0   8BD4    MOV EDX,ESP
778764F2   0F34    SYSENTER
778764F4   C3      RETN
```

This is point where the ResumeThread actually get executed and our suspended Process will start executing ,but we do not want to execute it to not get infected .So   stop Here

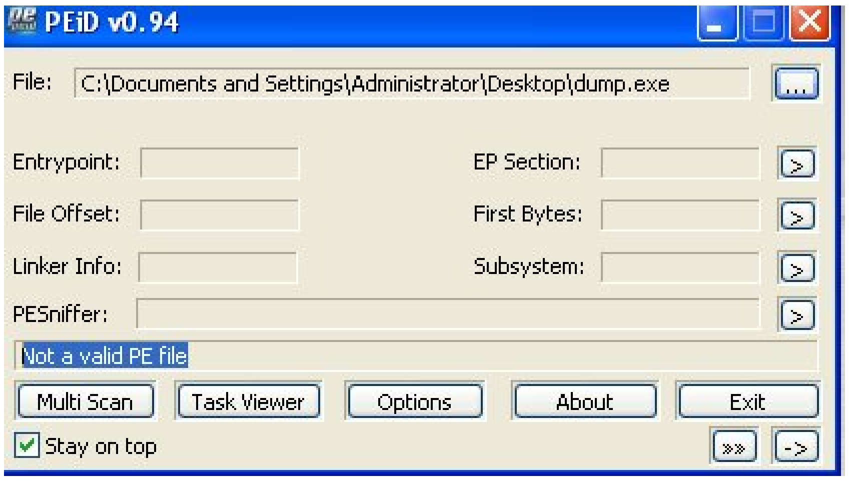
Now open the Process Explorer and dump the this process (the child process),select child process ,select full dump .



```
svchost.exe          1236         1,776 K    4,328 K Generic Host Process for Wi...  Microsoft Corporatio
spoolsv.exe          1604         3,756 K    5,532 K Spooler SubSystem App           Microsoft Corporatio
vmtoolsd.exe          348         6,408 K    8,216 K VMware Tools Core Service       VMware, Inc.
VMUpgradeHelper...    500           952 K    3,708 K VMware virtual hardware up...   VMware, Inc.
TPAutoCo                          1,568 K    3,956 K TPAutoConnect Printer Creat...  ThinPrint AG
TPAut                             1,596 K    4,660 K TPAutoConnect User Agent        ThinPrint AG
alg.exe                           1,064 K    3,232 K Application Layer Gateway S...  Microsoft Corporatio
svchost.e                         2,368 K    3,932 K Generic Host Process for Wi...  Microsoft Corporatio
lsass.exe                         3,516 K      884 K LSA Shell (Export Version)      Microsoft Corporatio
winupdate.ex                      4,036 K    5,068 K
MPK.exe                          11,552 K   13,408 K
explorer.exe                     18,636 K   20,268 K Windows Explorer                Microsoft Corporatio
VMwareTray.exe                    2,396 K    4,944 K VMware Tools tray application   VMware, Inc.
VMwareUser.exe                               388 K VMware Tools Service            VMware, Inc.
SpyStudio.exe                                736 K Spy Studio                      Nektra S.A.
DeRoX.exe                                 24,404 K OllyDbg, 32-bit analysing deb...
Crypted.exe                       6,040 K   17,236 K                                 T€R@Z1
Crypted.exe          3336         1,248 K    2,128 K
procexp.exe          2800  1.56   8,432 K    3,276 K Sysinternals Process Explorer   Sysinternals - www.
```

It will be saved as filename .dmp format ,I rename it to dump.exe

I named file as dump.exe ,and I scan it with PEID


PEiD v0.94

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

Entrypoint:

EP Section:

File Offset:

First Bytes:

Linker Info:

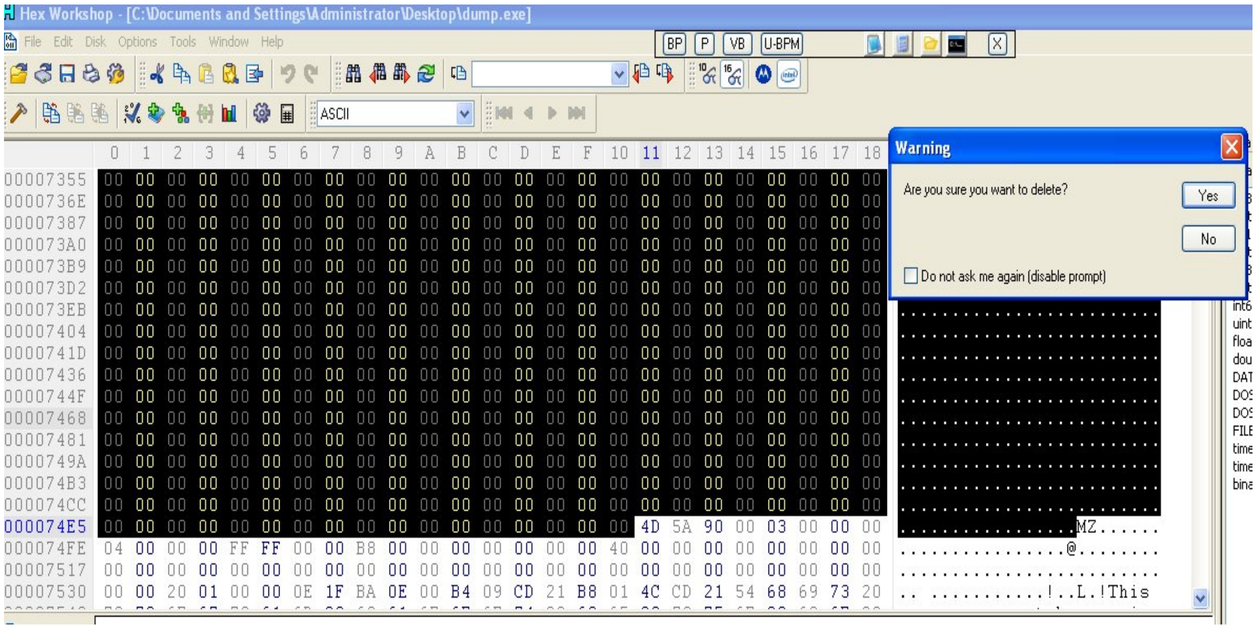
Subsystem:

PESniffer:

Not a valid PE file

Ah, not a Valid PE file..seems scary ..lets Fix this..The PE File start With Letter “MZ “.The File Analyzer like PEID gernally first check if the file contain MZ in starting or not ..if not that mean not a valid PE file(Also they do some extra tests ..but check for ”MZ” is first one.)

Open Up it dump.exe in Hex Workshop,search for “MZ”.Delte Everything above “MZ”. Save It ,Then our file become valid executable .

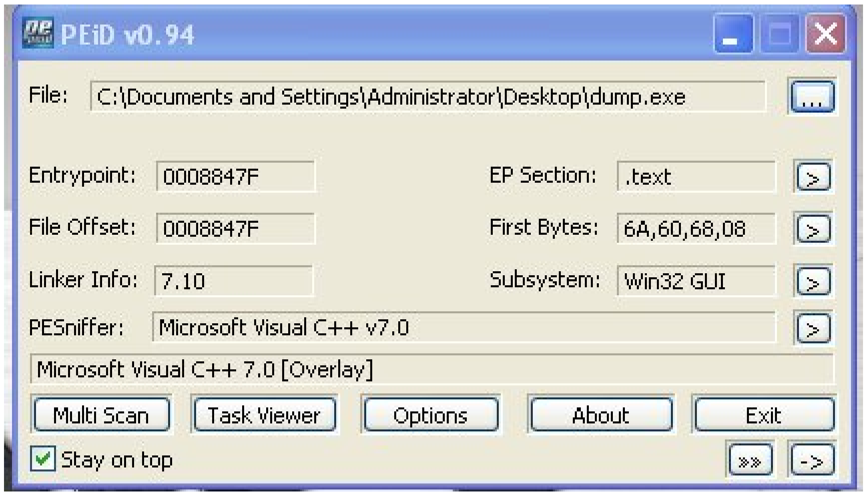

Hex Workshop - [C:\Documents and Settings\Administrator\Desktop\dump.exe]

```
          0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  10 11 12 13 14 15 16 17 18
00007355 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000736E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00007387 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000073A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000073B9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000073D2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000073EB 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
00007404 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
0000741D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
00007436 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
0000744F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
00007468 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
00007481 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
0000749A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
000074B3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
000074CC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
000074E5 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 4D 5A 90 00 03 00 00 00  .................MZ......
000074FE 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00  ................@........
00007517 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .........................
00007530 00 00 20 01 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 69 73 20  .. ...........!..L.!This
```

Warning

Are you sure you want to delete?

Now You can scan your modified File with PEID


PEiD v0.94

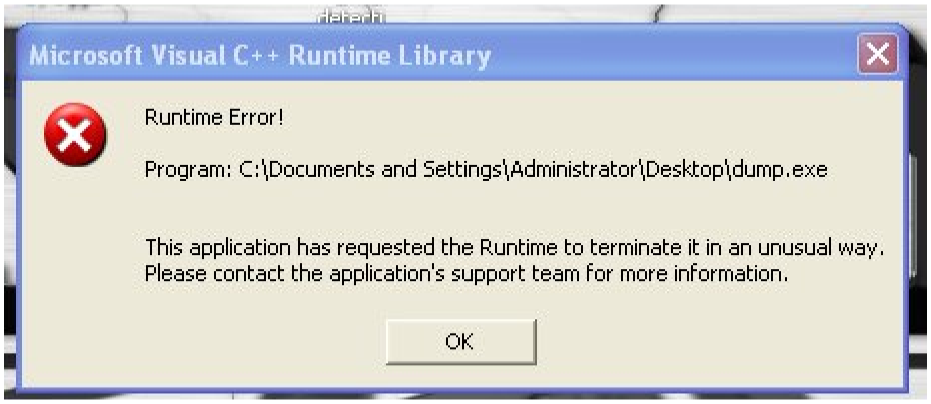
File: C:\Documents and Settings\Administrator\Desktop\dump.exe

Entrypoint: 0008847F
EP Section: .text
File Offset: 0008847F
First Bytes: 6A,60,68,08
Linker Info: 7.10
Subsystem: Win32 GUI
PESniffer: Microsoft Visual C++ v7.0

Microsoft Visual C++ 7.0 [Overlay]

Now Look Like Valid PE :D

But this is Not gonna run and giving the C++ Run time Error.


Microsoft Visual C++ Runtime Library

Runtime Error!

Program: C:\Documents and Settings\Administrator\Desktop\dump.exe

This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.

The Purpose of making this valid PE is to Find Its OEP by Loading it into Olly or by using other PE utlity tools

Note : You can find directly Calculate OEP from Hex Workshop without Deleting the Bytes If You know PE Header, I want to make it simple so I do it by this simple ad long way.

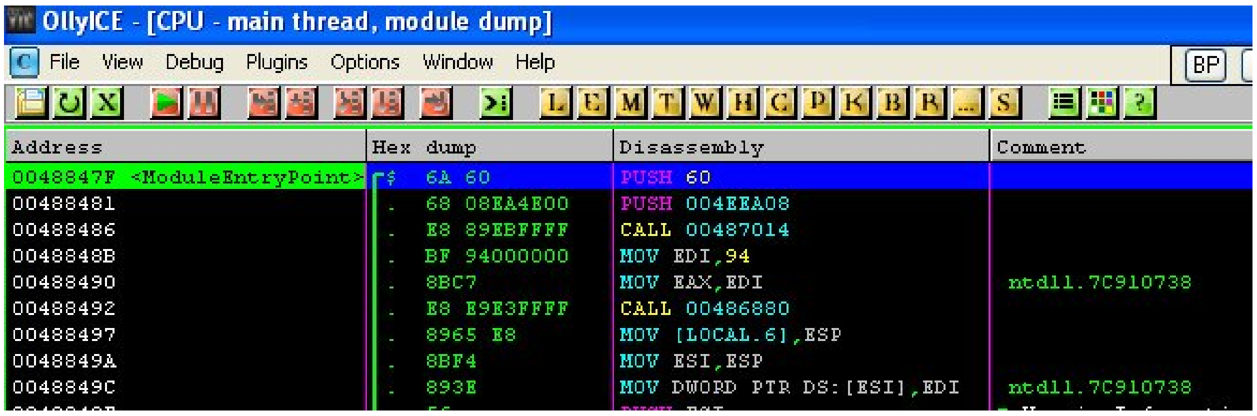
OEP :Orginal Entry Point .It is the address from which the program start execution.

Why we need OEP ?

WE Dump the program before the ResumeThread execute but it is not working.I am supposing the the crypted program is malware so I do not want to run it,then how I am going to to get it working .The idea is

Change the First Two Bytes at Program Entry point so that it trapped in infinite LOOP ,this way it will not able to get executed and everthing will be placed correctly and we will have a gud chance to dump it .

Lets Find the OEP by of our dumped file by opening it in olly.Also Note Down the starting bytes at entry point


OllyICE - [CPU - main thread, module dump]
File View Debug Plugins Options Window Help
BP

```
Address                      Hex dump        Disassembly                     Comment
0048847F <ModuleEntryPoint>  $ 6A 60         PUSH 60
00488481                     . 68 08EA4E00   PUSH 004EEA08
00488486                     . E8 89EBFFFF   CALL 00487014
0048848B                     . BF 94000000   MOV EDI,94
00488490                     . 8BC7          MOV EAX,EDI                     ntdll.7C910738
00488492                     . E8 E9E3FFFF   CALL 00486880
00488497                     . 8965 E8       MOV [LOCAL.6],ESP
0048849A                     . 8BF4          MOV ESI,ESP
0048849C                     . 893E          MOV DWORD PTR DS:[ESI],EDI      ntdll.7C910738
```

0048847F <ModuleEntryPoint>   6A 60   PUSH 60

EntryPoint 0048847F

The First Two Bytes are 6A 60

Show Time

Lets Finally Fix this

Run the Crypted.exe in olly ,Continue Untill the last instruction inside ResumeThread Executes Like we did before.

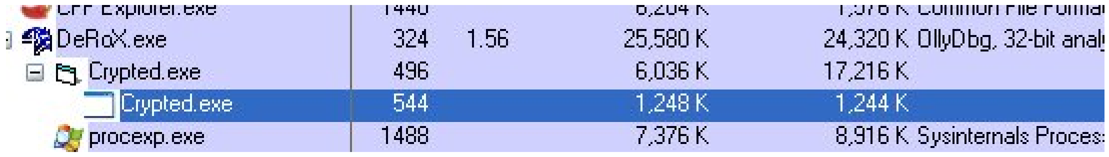
That is

countinue Stepping into ResumeThread API until this instruction

```
7C90EB8D      0F34                    SYSENTER
```

That's point where the actually execuation takes place

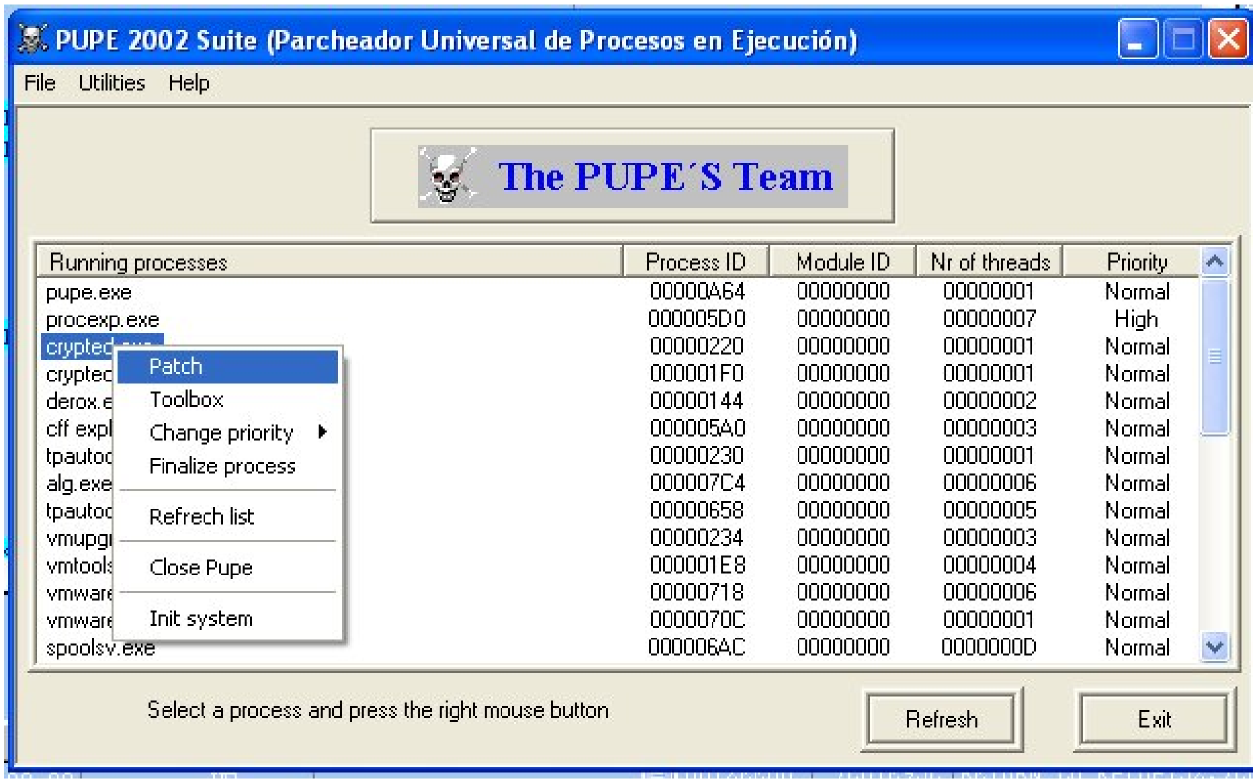
Now We Have to change first two bytes at EntryPoint to trap the program in infinite Loop,we olly use the little program PUPE for this



```
CFF Explorer.exe      1440             6,204 K      1,576 K Common File Format
DeRoX.exe              324   1.56     25,580 K     24,320 K OllyDbg, 32-bit anal
  Crypted.exe          496             6,036 K     17,216 K
    Crypted.exe        544             1,248 K      1,244 K
  procexp.exe         1488             7,376 K      8,916 K Sysinternals Proces
```

We can see our child process Crypted.exe in process Explorer. Its process id is 544 in decimal

Process id in Hex =220


PUPE 2002 Suite (Parcheador Universal de Procesos en Ejecución)

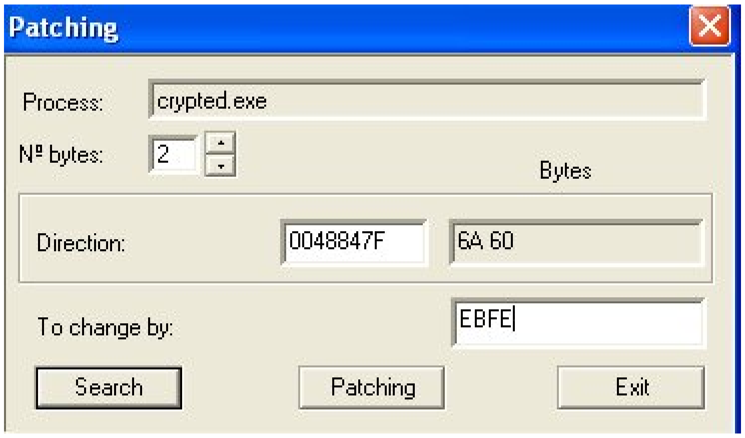
File Utilities Help

The PUPE´S Team

Running processes	Process ID	Module ID	Nr of threads	Priority
pupe.exe	00000A64	00000000	00000001	Normal
procexp.exe	000005D0	00000000	00000007	High
crypted.exe	00000220	00000000	00000001	Normal
crypted	000001F0	00000000	00000001	Normal
derox.e	00000144	00000000	00000002	Normal
cff expl	000005A0	00000000	00000003	Normal
tpautoc	00000230	00000000	00000001	Normal
alg.exe	000007C4	00000000	00000006	Normal
tpautoc	00000658	00000000	00000005	Normal
vmupgr	00000234	00000000	00000003	Normal
vmtools	000001E8	00000000	00000004	Normal
vmware	00000718	00000000	00000006	Normal
vmware	0000070C	00000000	00000001	Normal
spoolsv.exe	000006AC	00000000	0000000D	Normal

Select a process and press the right mouse button

Select the Target Process and click Patch . Then You will see the patch window Like this


Patching

Process: crypted.exe

Nº bytes: 2

Bytes

Direction: 0048847F 6A 60

To change by: EBFE

Change the Number of bytes to 2

Put the OEP in the Direction option and click search we get 6A 60 as bytes (these are ogrinal bytes .note it)

Put EB FE in change by .

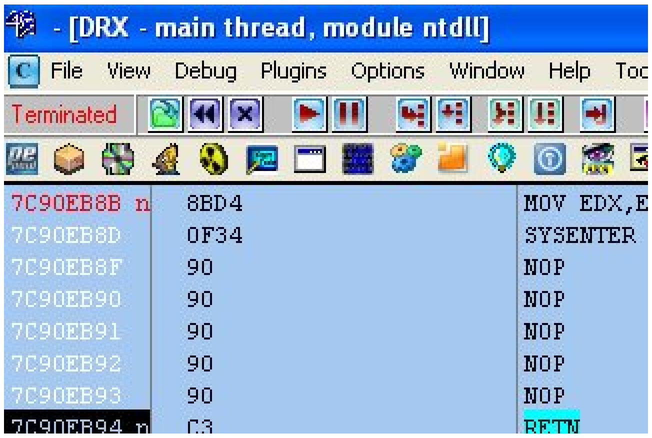
EB FE will instruction will make the jump to to same instruction again and again and hence trap it in infinite loop

Now click on patching

After that the orginal bytes are replaced by EB FE .

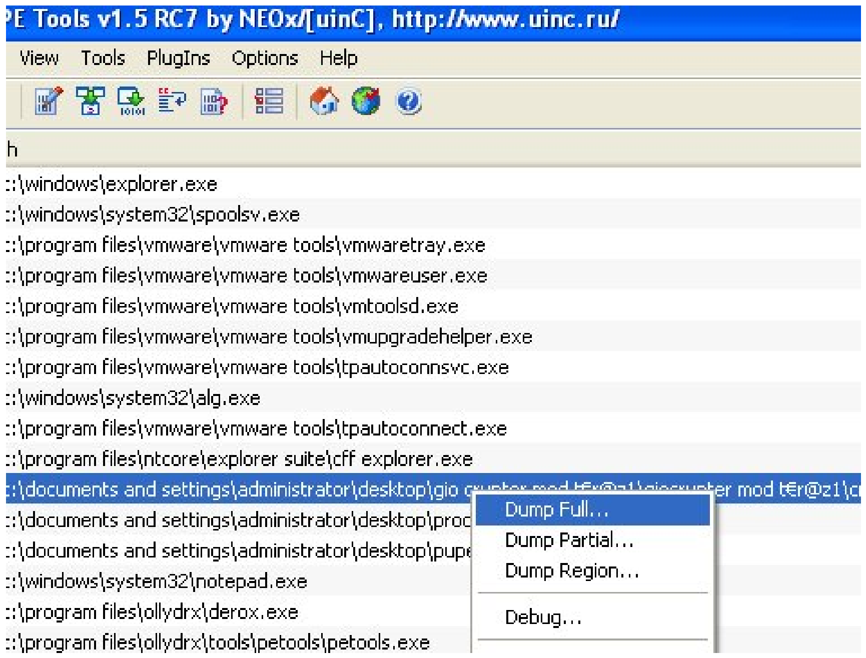
Now Go to our ollly again and click and Run the Program

After Clicking on Run button you will see that that your process is terminated in olly .


- [DRX - main thread, module ntdll]
File View Debug Plugins Options Window Help Toc
Terminated

```
7C90EB8B n  8BD4   MOV EDX,E
7C90EB8D    0F34   SYSENTER
7C90EB8F    90     NOP
7C90EB90    90     NOP
7C90EB91    90     NOP
7C90EB92    90     NOP
7C90EB93    90     NOP
7C90EB94 n  C3     RETN
```

Don’t Worry it does not matter to us .Only child process matter to us that is still running (trapped in infinite loop) . Now you just have to Dump it with Your Favourate Dumping tool. I Will dump it with my favourate that is PE tools

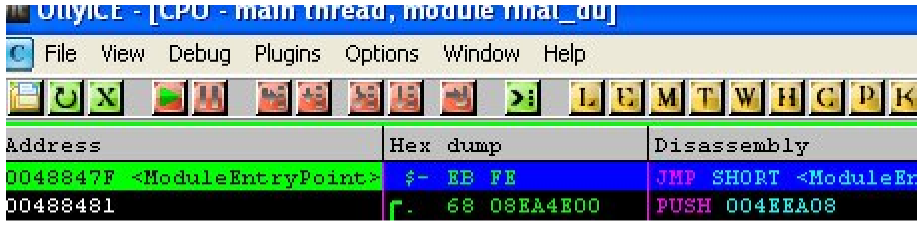


```
PE Tools v1.5 RC7 by NEOx/[uinC], http://www.uinc.ru/
View   Tools   PlugIns   Options   Help
h
::\windows\explorer.exe
::\windows\system32\spoolsv.exe
::\program files\vmware\vmware tools\vmwaretray.exe
::\program files\vmware\vmware tools\vmwareuser.exe
::\program files\vmware\vmware tools\vmtoolsd.exe
::\program files\vmware\vmware tools\vmupgradehelper.exe
::\program files\vmware\vmware tools\tpautoconnsvc.exe
::\windows\system32\alg.exe
::\program files\vmware\vmware tools\tpautoconnect.exe
::\program files\ntcore\explorer suite\cff explorer.exe
::\documents and settings\administrator\desktop\gio c            ter mod t€r@z1\c
::\documents and settings\administrator\desktop\proc
::\documents and settings\administrator\desktop\pupe
::\windows\system32\notepad.exe
::\program files\ollydrx\derox.exe
::\program files\ollydrx\tools\petools\petools.exe
```

Click on Dump Full and save it with any name you want .i saved it with final_dump.exe

After Dumping Also Kill the process.

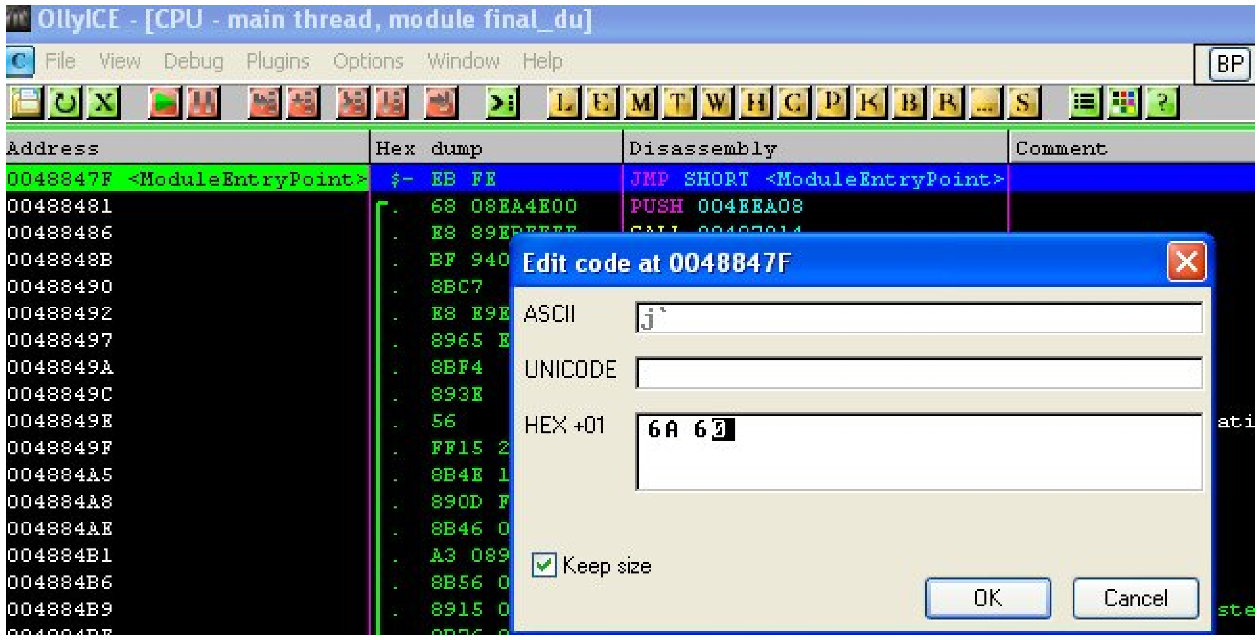
Now open the final_dump.exe in olly



```
OllyICE - [CPU - main thread, module final_du]
File  View  Debug  Plugins  Options  Window  Help
Address                     Hex dump          Disassembly
0048847F <ModuleEntryPoint>  $- EB FE          JMP SHORT <ModuleEr
00488481                     ┌. 68 08EA4E00    PUSH 004EEA08
```

As You can see the first two bytes are EB FE   ,they will always trp the program in infinite loop to fix it replace these two bytes with orginal two bytes that are   6A 60

Right   click on instruction then go to binary -> edit options and replace it with orginal bytes as shown in pic


OllyICE - [CPU - main thread, module final_du]
File View Debug Plugins Options Window Help
BP
L E M T W H C P K B R ... S
OK
Cancel

```
Address                      Hex dump        Disassembly                     Comment
0048847F <ModuleEntryPoint>  $-  EB FE       JMP SHORT <ModuleEntryPoint>
00488481                     .   68 08EA4E00 PUSH 004EEA08
00488486                     .   E8 89E
0048848B                     .   BF 940
00488490                     .   8BC7
00488492                     .   E8 E9E
00488497                     .   8965 E
0048849A                     .   8BF4
0048849C                     .   893E
0048849E                     .   56
0048849F                     .   FF15 2
004884A5                     .   8B4E 1
004884A8                     .   890D F
004884AE                     .   8B46 0
004884B1                     .   A3 089
004884B6                     .   8B56 0
004884B9                     .   8915 0
```

Edit code at 0048847F

ASCII: j`

UNICODE:

HEX +01: 6A 60

[✓] Keep size

Now click on the copy to executable option and save this file .Now You have Your orginal file back .

Congrats You just Unpck the crypted file successfully.

You can verify it by running .

Important :

As I already mention the crypter coders now days use the windows undcoumneted native APIs instead of documented API

FOR example Use of NtResumeThread instead of ResumeThread.

So I suggest to Put BP on NtResmeThread instead of Resume Thread.

Apply same to all other API that you want to break on .

These crypters gernally add junk code to make them undtectbale but don't worry if they are using the same RUN PE method they will get unpacked by using this method because adding junk code did not matter at the end they have to to call ResumeThread :P

NOTE :This Method works on the crypter who are using the above method written .I found that more than 60 % crypters use the method.

If You like My tute then leave comments or you can mail me at

Email : arunpreet90@gmail.com