

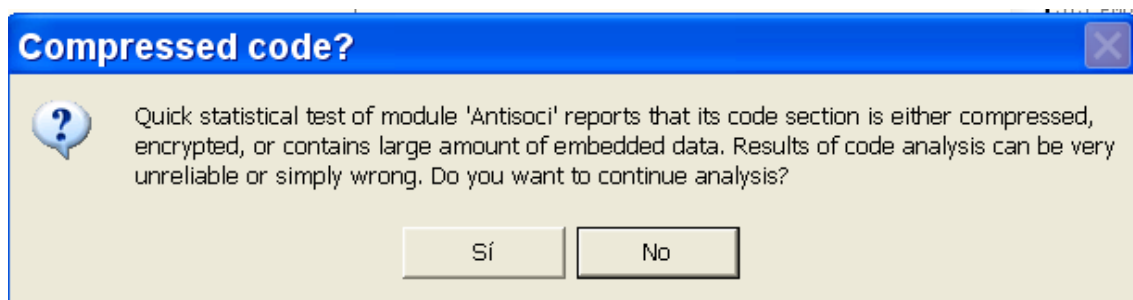
## INTRODUCCION AL CRACKING CON OLLYDBG PARTE 24

Me pidieron que antes de empezar con la parte que versa sobre excepciones, muestre como se hace correr en OLLYDBG el antisocial que deje como ejemplo, la vez anterior.

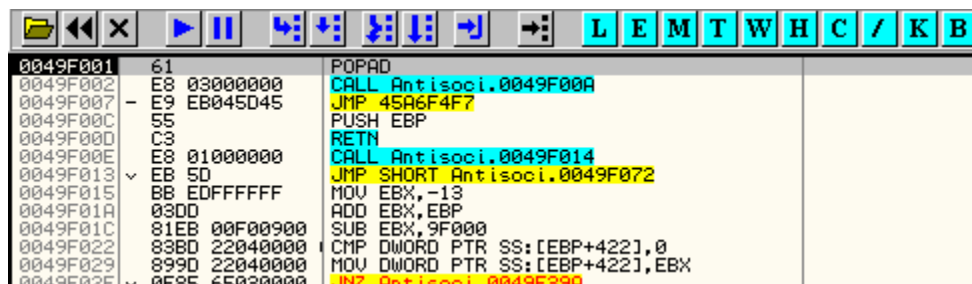
Antes que nada sabemos que es un programa empackado, y eso aun no se ha enseñado por lo cual lo haremos correr en OLLYDBG haciendo los cambios en la memoria solamente sin guardarlos, cuando ya sepamos desempacar pues podremos guardar los cambios definitivos.

Primero lo haremos correr en un OLLYDBG renombrado con todos los plugins puestos para ver porque no corre y tratar de arreglarlo.

Arrancamos el antisocial en mi OLLYDBG renombrado, parcheado y con todos los plugins habilitados.



Esto ya nos da una idea que puede estar empackado, llegamos al ENTRY POINT

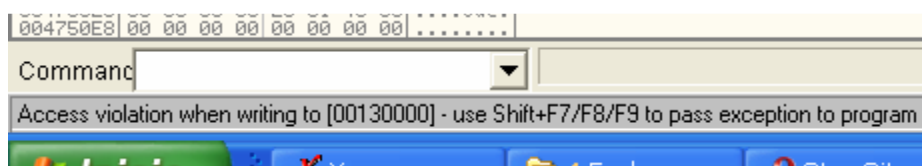


Allí ya vemos algo extraño, pues POPAD es la sentencia que se usa para recuperar del stack, los valores que guardo previamente un PUSHAD, y aquí no hay ningún PUSHAD antes, por lo tanto hay algo sospechoso allí.

Lo corremos igual a ver que pasa



Para aquí



O sea da error cuando quiere hacer el PUSH como si el stack no tuviera permiso para escribir allí, pero el stack siempre tiene permiso de escritura que ocurrió aquí, veamos el stack

|          |          |
|----------|----------|
| 00130000 | 78746341 |
| 00130004 | 00000020 |
| 00130008 | 00000001 |
| 0013000C | 00002498 |
| 00130010 | 000000C4 |
| 00130014 | 00000000 |
| 00130018 | 00000020 |
| 0013001C | 00000000 |
| 00130020 | 00000014 |
| 00130024 | 00000001 |
| 00130028 | 00000006 |
| 0013002C | 00000034 |
| 00130030 | 00000114 |

O sea el valor superior del stack es 130000, y si reiniciamos el programa

|          |          |  |             |  |      |    |      |    |
|----------|----------|--|-------------|--|------|----|------|----|
| 00010000 | 00001000 |  |             |  | Priv | RW |      | RW |
| 00020000 | 00001000 |  |             |  | Priv | RW |      | RW |
| 0012B000 | 00001000 |  |             |  | Priv | RW | Guar | RW |
| 0012C000 | 00004000 |  | stack of ma |  | Priv | RW | Guar | RW |
| 00130000 | 00003000 |  |             |  | Map  | R  |      | R  |
| 00140000 | 00003000 |  |             |  | Priv | RW |      | RW |
| 00240000 | 00006000 |  |             |  | Priv | RW |      | RW |
| 00250000 | 00003000 |  |             |  | Map  | RW |      | RW |
| 00260000 | 00016000 |  |             |  | Map  | R  |      | R  |

Y vemos las secciones, vemos que el stack va en mi maquina desde 12c000 hasta 12ffff, el error fue que el stack se salio de esa sección, y ahora apunta a la sección siguiente que empieza en 130000 y que como no es el stack, no tiene permiso de escritura y da error.

Lleguemos nuevamente a la zona del error

|          |               |                                |
|----------|---------------|--------------------------------|
| 0049F3A7 | 0BC7          | UN EAX,EAX                     |
| 0049F3A9 | 8985 A8030000 | MOV DWORD PTR SS:[EBP+3A8],EAX |
| 0049F3AF | 61            | POPAD                          |
| 0049F3B0 | 75 08         | JNZ SHORT Antisoci.0049F3BA    |
| 0049F3B2 | B8 01000000   | MOV EAX,1                      |
| 0049F3B7 | C2 0C00       | RETN 0C                        |
| 0049F3BA | 68 88494700   | PUSH Antisoci.00474988         |

Vemos que el programa ejecuta otro popad y hace un salto JNZ al PUSH que da error veamos, pongamos un BPX en ese popad antes del error.

|          |               |                                |
|----------|---------------|--------------------------------|
| 0049F3A5 | 59            | POP EAX                        |
| 0049F3A7 | 0BC9          | OR ECX,ECX                     |
| 0049F3A9 | 8985 A8030000 | MOV DWORD PTR SS:[EBP+3A8],EAX |
| 0049F3AF | 61            | POPAD                          |
| 0049F3B0 | 75 08         | JNZ SHORT Antisoci.0049F3BA    |
| 0049F3B2 | B8 01000000   | MOV EAX,1                      |
| 0049F3B7 | C2 0C00       | RETN 0C                        |
| 0049F3BA | 68 88494700   | PUSH Antisoci.00474988         |
| 0049F3BF | C3            | RETN                           |
| 0049F3C0 | 8B85 26040000 | MOV EAX,DWORD PTR SS:[EBP+426] |
| 0049F3C6 | 8D8D 3B040000 | LEA ECX,DWORD PTR SS:[EBP+43B] |

Ahora reiniciemos el programa y demos RUN para que pare allí.

|          |               |                                |
|----------|---------------|--------------------------------|
| 0049F3AF | 61            | POPAD                          |
| 0049F3B0 | 75 08         | JNZ SHORT Antisoci.0049F3BA    |
| 0049F3B2 | B8 01000000   | MOV EAX,1                      |
| 0049F3B7 | C2 0C00       | RETN 0C                        |
| 0049F3BA | 68 88494700   | PUSH Antisoci.00474988         |
| 0049F3BF | C3            | RETN                           |
| 0049F3C0 | 8B85 26040000 | MOV EAX,DWORD PTR SS:[EBP+426] |
| 0049F3C6 | 8D8D 3B040000 | LEA ECX,DWORD PTR SS:[EBP+43B] |

Veamos el stack

|          |          |                                    |
|----------|----------|------------------------------------|
| 0012FFE4 | 7C8399F3 | SE handler                         |
| 0012FFE8 | 7C816D58 | kernel32.7C816D58                  |
| 0012FFEC | 00000000 |                                    |
| 0012FFF0 | 00000000 |                                    |
| 0012FFF4 | 00000000 |                                    |
| 0012FFF8 | 0049F001 | OFFSET Antisoci.<ModuleEntryPoint> |
| 0012FFFC | 00000000 |                                    |

Esta aun en la sección correcta ejecutemos el popad



|          |          |                                    |
|----------|----------|------------------------------------|
| 0012FFA4 | 7C920738 | ntdll.7C920738                     |
| 0012FFA8 | FFFFFFFF |                                    |
| 0012FFAC | 0012FFF0 |                                    |
| 0012FFB0 | 0012FFC4 |                                    |
| 0012FFB4 | 7FFD4000 |                                    |
| 0012FFB8 | 7C91EB94 | ntdll.KiFastSystemCallRet          |
| 0012FFBC | 0012FFB0 |                                    |
| 0012FFC0 | 00000000 |                                    |
| 0012FFC4 | 7C816D4F | RETURN to kernel32.7C816D4F        |
| 0012FFC8 | 7C920738 | ntdll.7C920738                     |
| 0012FFCC | FFFFFFFF |                                    |
| 0012FFD0 | 7FFD4000 |                                    |
| 0012FFD4 | 8054A938 |                                    |
| 0012FFD8 | 0012FFC8 |                                    |
| 0012FFDC | 84797AF8 |                                    |
| 0012FFE0 | FFFFFFFF | End of SEH chain                   |
| 0012FFE4 | 7C8399F3 | SE handler                         |
| 0012FFE8 | 7C816D58 | kernel32.7C816D58                  |
| 0012FFEC | 00000000 |                                    |
| 0012FFF0 | 00000000 |                                    |
| 0012FFF4 | 00000000 |                                    |
| 0012FFF8 | 0049F001 | OFFSET Antisoci.<ModuleEntryPoint> |
| 0012FFFC | 00000000 |                                    |

Por lo cual hacer un POPAD no lo sacara de sección, veamos pasémoslo con F8

|          |          |                                    |
|----------|----------|------------------------------------|
| 0012FFC4 | 7C816D4F | RETURN to kernel32.7C816D4F        |
| 0012FFC8 | 7C920738 | ntdll.7C920738                     |
| 0012FFCC | FFFFFFFF |                                    |
| 0012FFD0 | 7FFD4000 |                                    |
| 0012FFD4 | 8054A938 |                                    |
| 0012FFD8 | 0012FFC8 |                                    |
| 0012FFDC | 84797AF8 |                                    |
| 0012FFE0 | FFFFFFFF | End of SEH chain                   |
| 0012FFE4 | 7C8399F3 | SE handler                         |
| 0012FFE8 | 7C816D58 | kernel32.7C816D58                  |
| 0012FFEC | 00000000 |                                    |
| 0012FFF0 | 00000000 |                                    |
| 0012FFF4 | 00000000 |                                    |
| 0012FFF8 | 0049F001 | OFFSET Antisoci.<ModuleEntryPoint> |
| 0012FFFC | 00000000 |                                    |

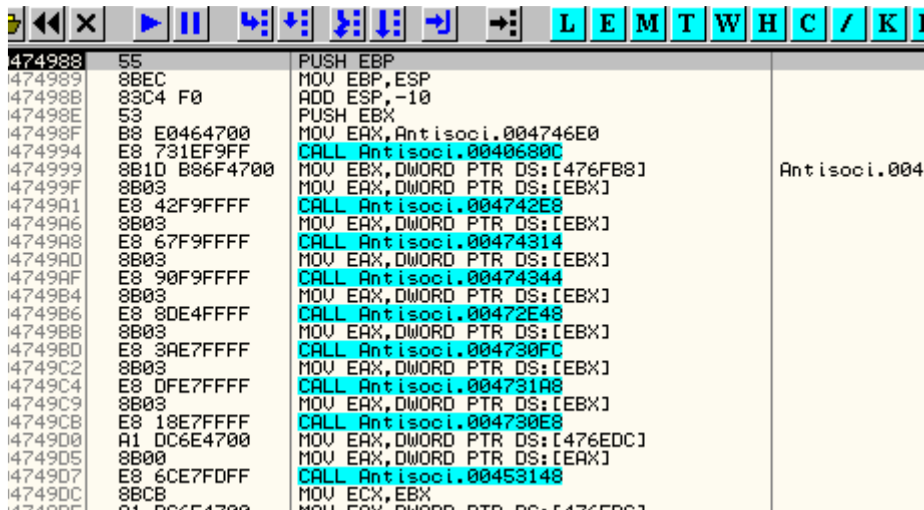
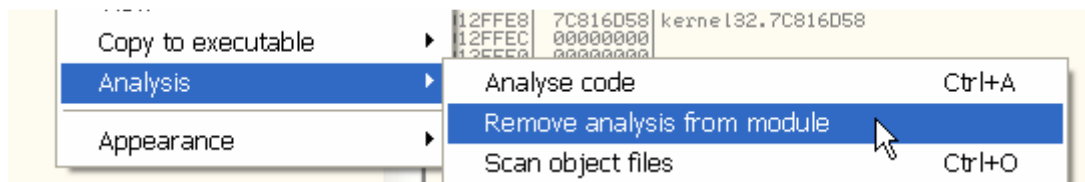
Vemos que se mantiene todo correcto

|          |               |                                |
|----------|---------------|--------------------------------|
| 0049F3BF | 61            | POPAD                          |
| 0049F3B0 | 75 08         | JNZ SHORT Antisoci.0049F3BA    |
| 0049F3B2 | B8 01000000   | MOV EAX,1                      |
| 0049F3B7 | C2 0C00       | RETN 0C                        |
| 0049F3BA | 68 88494700   | PUSH Antisoci.00474988         |
| 0049F3BF | C3            | RETN                           |
| 0049F3C0 | 8B85 26040000 | MOV EAX,DWORD PTR SS:[EBP+426] |
| 0049F3C6 | 8D8D 3B040000 | LEA ECX,DWORD PTR SS:[EBP+43B] |

Llegamos al PUSH y al RET, traceando, y con f8 pasamos el RET

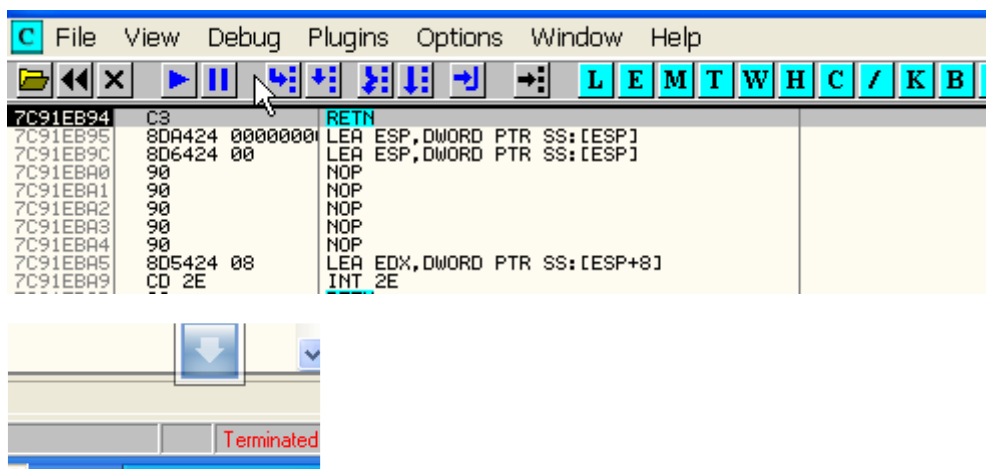
|          |    |       |          |
|----------|----|-------|----------|
| 00474988 | 55 | DB 55 | CHAR 'U' |
| 00474989 | 8B | DB 8B |          |
| 0047498A | EC | DB EC |          |
| 0047498B | 83 | DB 83 |          |
| 0047498C | C4 | DB C4 |          |
| 0047498D | F0 | DB F0 |          |
| 0047498E | 53 | DB 53 | CHAR 'S' |
| 0047498F | B8 | DB B8 |          |
| 00474990 | E0 | DB E0 |          |
| 00474991 | 46 | DB 46 | CHAR 'F' |
| 00474992 | 47 | DB 47 | CHAR 'G' |
| 00474993 | 00 | DB 00 |          |
| 00474994 | E8 | DB E8 |          |
| 00474995 | 73 | DB 73 | CHAR 's' |
| 00474996 | 1E | DB 1E |          |
| 00474997 | F9 | DB F9 |          |
| 00474998 | FF | DB FF |          |
| 00474999 | 8B | DB 8B |          |
| 0047499A | 1D | DB 1D |          |
| 0047499B | B8 | DB B8 |          |
| 0047499C | 6F | DB 6F | CHAR 'o' |
| 0047499D | 47 | DB 47 | CHAR 'o' |

Bueno aquí hay un problema de análisis

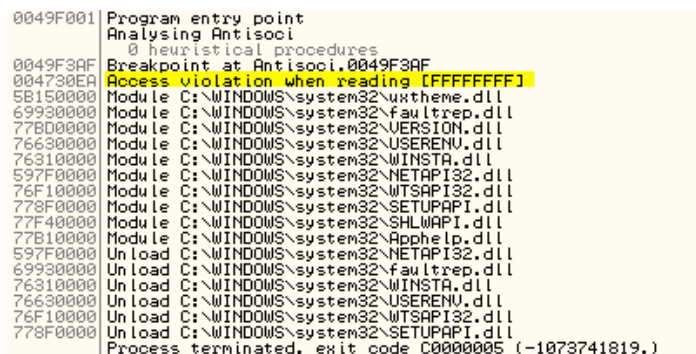


Ahora si se ve bien

Veamos que ocurre si damos RUN



Veamos en el LOG del OLLYDBG si vemos algo ya que el programa se termina

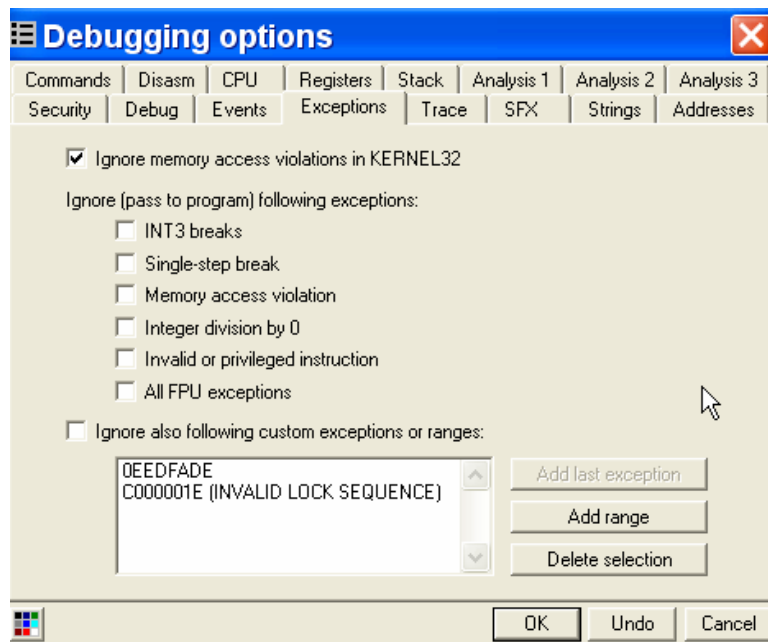


Vemos que hay una excepción, luego de parar en el BPX del popad

Reiniciemos y repitamos los pasos para llegar nuevamente adonde estábamos.

|          |               |                                |                   |
|----------|---------------|--------------------------------|-------------------|
| 00474988 | 55            | PUSH EBP                       |                   |
| 00474989 | 8BEC          | MOV EBP, ESP                   |                   |
| 0047498B | 83C4 F0       | ADD ESP, -10                   |                   |
| 0047498E | 53            | PUSH EBX                       |                   |
| 0047498F | B8 E0464700   | MOV EAX, Antisoci.004746E0     |                   |
| 00474994 | E8 731EF9FF   | CALL Antisoci.004746E8         |                   |
| 00474999 | 8B1D B86F4700 | MOV EBX, DWORD PTR DS:[476FB8] | Antisoci.00478D7C |
| 0047499F | 8B03          | MOV EAX, DWORD PTR DS:[EBX]    |                   |
| 004749A1 | E8 42F9FFFF   | CALL Antisoci.004742E8         |                   |
| 004749A6 | 8B03          | MOV EAX, DWORD PTR DS:[EBX]    |                   |
| 004749A8 | E8 67F9FFFF   | CALL Antisoci.00474314         |                   |
| 004749AD | 8B03          | MOV EAX, DWORD PTR DS:[EBX]    |                   |
| 004749AF | E8 90F9FFFF   | CALL Antisoci.00474344         |                   |
| 004749B4 | 8B03          | MOV EAX, DWORD PTR DS:[EBX]    |                   |
| 004749B6 | E8 80E4FFFF   | CALL Antisoci.00472E48         |                   |
| 004749BB | 8B03          | MOV EAX, DWORD PTR DS:[EBX]    |                   |
| 004749BD | E8 3AE7FFFF   | CALL Antisoci.004730FC         |                   |
| 004749C2 | 8B03          | MOV EAX, DWORD PTR DS:[EBX]    |                   |
| 004749C4 | E8 DFE7FFFF   | CALL Antisoci.004731A8         |                   |
| 004749C9 | 8B03          | MOV EAX, DWORD PTR DS:[EBX]    |                   |
| 004749CB | E8 18E7FFFF   | CALL Antisoci.004730E8         |                   |

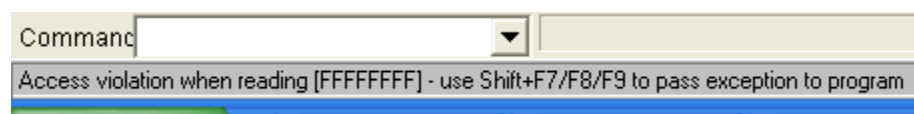
Quitemos todas las tildes, menos la primera para que pare en las excepciones



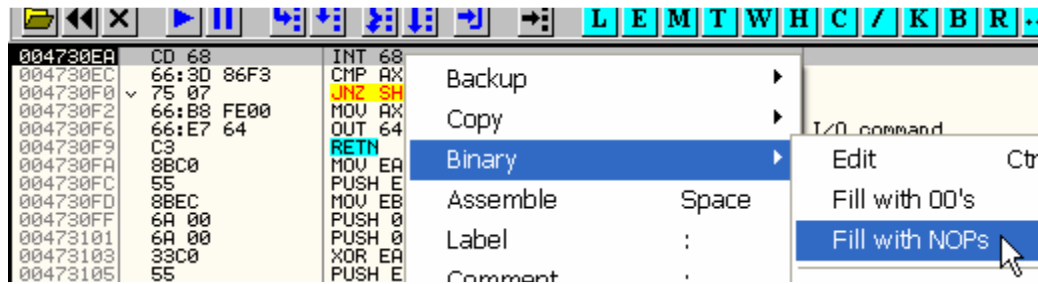
Demos RUN

|          |            |                             |         |
|----------|------------|-----------------------------|---------|
| 004730EA | CD 68      | INT 3                       |         |
| 004730EC | 66:3D 86F3 | CMP AX, 0F386               |         |
| 004730F0 | 75 07      | JNZ SHORT Antisoci.004730F9 |         |
| 004730F2 | 66:B8 FE00 | MOV AX, 0FE                 |         |
| 004730F6 | 66:E7 64   | OUT 64, AX                  | I/O cop |
| 004730F9 | C3         | RETN                        |         |
| 004730FA | 8BC0       | MOV EAX, EAX                |         |

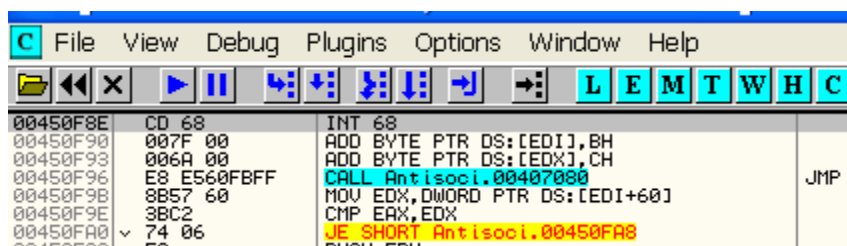
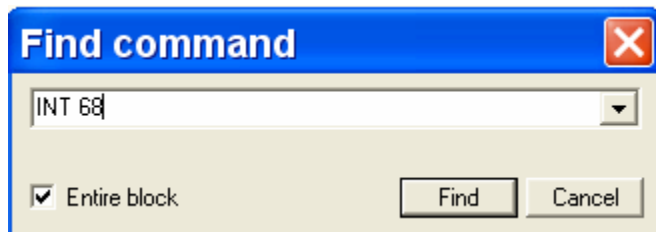
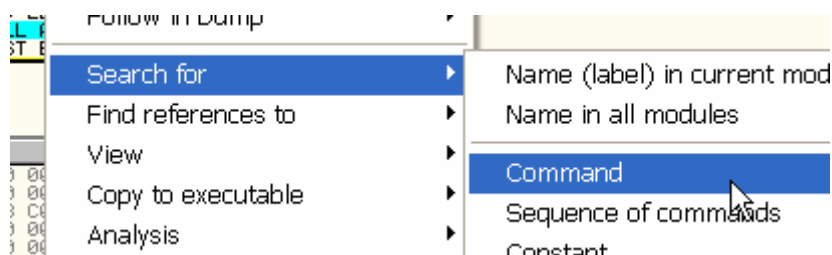
Para en la excepción que vimos en el LOG



INT68 es una de las pocas excepciones que el OLLYDBG no puede manejar, podemos pasarla nopeandola.



Por otro lado sabemos que puede haber mas INT68 que molesten busquemos a ver si halla alguna mas, así lo nopeamos directamente.



Vemos que halla otro lo nopeamos



Buscamos nuevamente hay otro mas, lo nopeamos

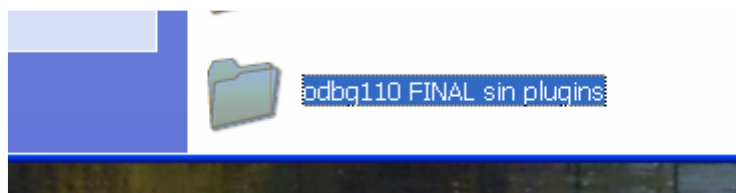
|          |            |                             |         |
|----------|------------|-----------------------------|---------|
| 00473EB7 | 90         | NOP                         |         |
| 00473EB8 | 90         | NOP                         |         |
| 00473EB9 | 66:3D 86F3 | CMP AX,0F386                |         |
| 00473EBD | 75 07      | JNZ SHORT Antisoci.00473EC6 |         |
| 00473EBF | 66:B8 FE00 | MOV AX,0FE                  |         |
| 00473EC3 | 66:E7 64   | OUT 64,AX                   | I/O com |
| 00473EC7 | 66:00 00   | MOV ECX,00                  |         |

Si hacemos CTRL + L continua buscando a partir del ultimo que hallo

Ya no halla mas ahora si damos RUN

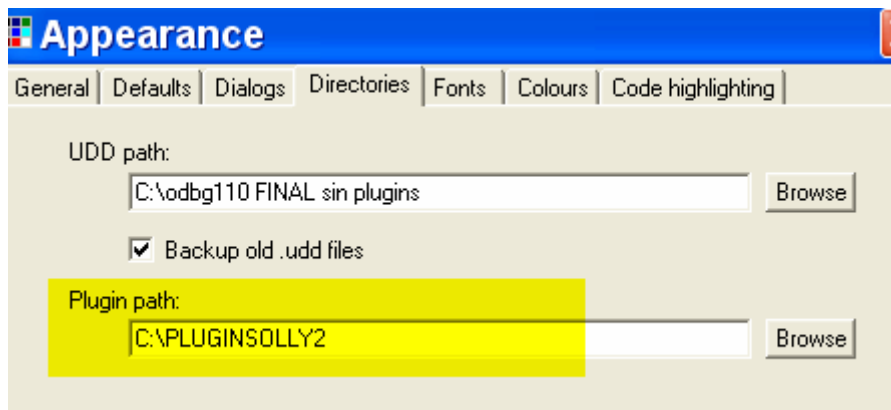


Como ven todo eso hay que hacer para que corra el OLLYDBG con los plugins, ahora tratemos de correrlo en un OLLYDBG sin plugins, y sin renombrar ni parchear.

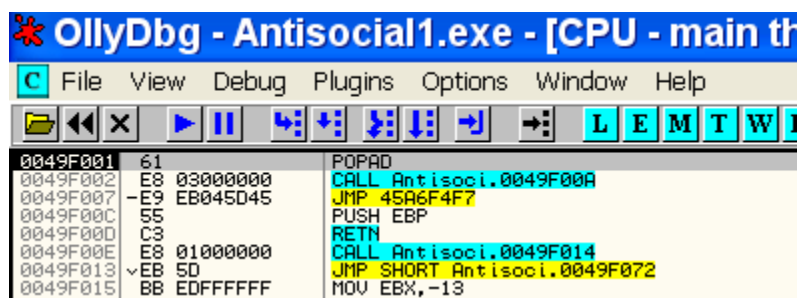


Allí descomprimí un OLLYDBG sin plugins, bah el único que tiene es el command bar, ya que ese no protege de nada es solo por comodidad, a este OLLYDBG, le hice una carpeta diferente de plugins y el path a los plugins lo apunte allí.

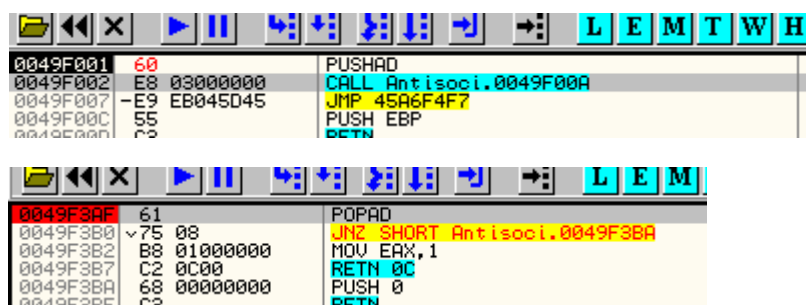




Como vemos allí hay solo dos plugins, lo arranco en ese OLLYDBG



Repito las operaciones de cambiar el popad



Poner el BPX en el POPAD veamos si llega allí demos RUN

```

0049F3AF 61          POPAD
0049F3B0 75 08       JNZ SHORT Antisoci.0049F3BA
0049F3B2 B8 01000000 MOV EAX,1
0049F3B7 C2 0C00     RETN 0C
0049F3BA 68 88494700 PUSH Antisoci.00474988
0049F3BF C3          RETN
0049F3C0 8B85 26040000 MOV EAX,DWORD PTR SS:[EBP+426]

```

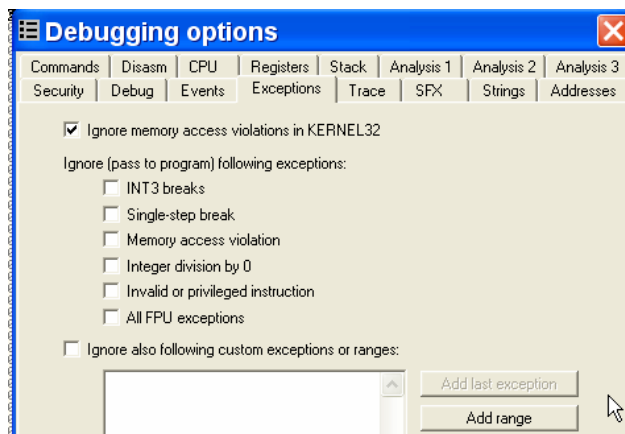
Lleguemos hasta el código del programa desempacado.

```

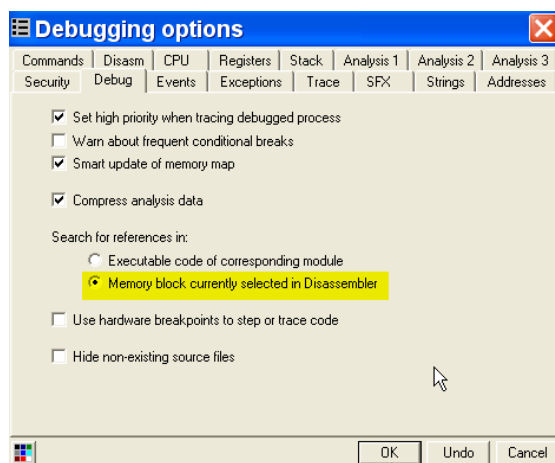
00474988 55          PUSH EBP
00474989 8BEC        MOV EBP,ESP
0047498B 83C4 F0     ADD ESP,-10
0047498E 53          PUSH EBX
0047498F B8 E0464700 MOV EAX,Antisoci.004746E0
00474994 E8 731EF9FF CALL Antisoci.00406800
00474999 8B1D B86F4700 MOV EBX,DWORD PTR DS:[476FB8]
0047499F 8B03        MOV EAX,DWORD PTR DS:[EBX]
004749A1 E8 42F9FFFF CALL Antisoci.004742E8
004749A6 8B03        MOV EAX,DWORD PTR DS:[EBX]
004749A8 E8 67F9FFFF CALL Antisoci.00474314
004749AD 8B03        MOV EAX,DWORD PTR DS:[EBX]
004749AF E8 90F9FFFF CALL Antisoci.00474344
004749B4 8B03        MOV EAX,DWORD PTR DS:[EBX]
004749B6 E8 80E4FFFF CALL Antisoci.00472E48
004749BB 8B03        MOV EAX,DWORD PTR DS:[EBX]
004749BD E8 3AE7FFFF CALL Antisoci.004730FC
004749C2 8B03        MOV EAX,DWORD PTR DS:[EBX]
004749C4 E8 DFE7FFFF CALL Antisoci.004731A8

```

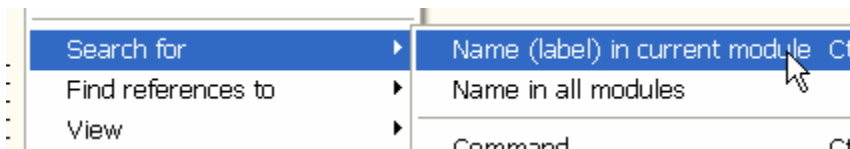
Podemos nopearle los INT68 o quitar las tildes en exceptions y cada vez que pare si es un INT68 nopearlo y si no pasarlo con SHIFT +f9 como a cualquier excepción.



Veamos que apis usa el programa para esto debemos recordar que la tilde en

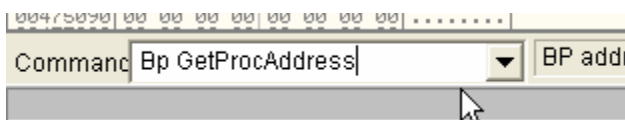


Debe estar colocada alli para que muestre la información de la seccion en que estamos.



| Names in Antisoci |         |        |                                |
|-------------------|---------|--------|--------------------------------|
| Address           | Section | Type   | Name                           |
| 004A0005          | .....   | Import | user32.GetKeyboardType         |
| 0049FF60          | .....   | Import | kernel32.GetModuleHandleA      |
| 0049FF5C          | .....   | Import | kernel32.GetProcAddress        |
| 004A0100          | .....   | Import | comctl32.ImageList_SetIconSize |
| 0049FF64          | .....   | Import | kernel32.LoadLibraryA          |
| 0049F001          | .....   | Export | <ModuleEntryPoint>             |
| 004A000D          | .....   | Import | advapi32.RegQueryValueExA      |
| 004A00ED          | .....   | Import | advapi32.RegQueryValueExA      |
| 004A0105          | .....   | Import | oleaut32.SafeArrayPtrOfIndex   |
| 004A00E5          | .....   | Import | oleaut32.SysFreeString         |
| 004A00F5          | .....   | Import | gdi32.UnrealizeObject          |
| 004A00FD          | .....   | Import | user32.WindowFromPoint         |

Bueno vemos que no hay apis de las sospechosas, pero usa muy pocas apis y esta alli GetProcAddress para cargar mas apis nuevas, así que pongamos un BPX en la api GetProcAddress.



Demos RUN

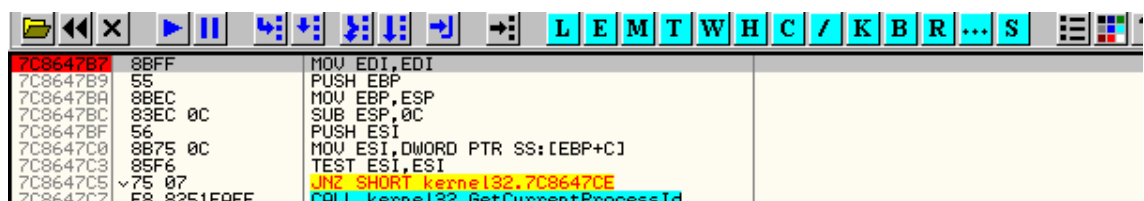
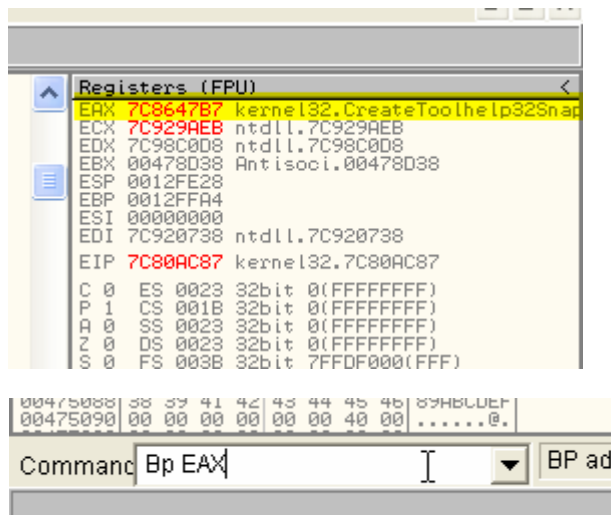
|          |          |  |
|----------|----------|--|
| 0012FF58 | 0040B324 | CALL to GetProcAddress from Antisoci.0040B31F      |
| 0012FF5C | 7C800000 | hModule = 7C800000 (kernel32)                      |
| 0012FF60 | 0040B350 | ProcNameOrOrdinal = "GetDiskFreeSpaceExA"          |
| 0012FF64 | 00000008 |  |
| 0012FF68 | 0040BFE1 | RETURN to Antisoci.0040BFE1 from Antisoci.0040B308 |
| 0012FF6C | 0012FF80 | Pointer to next SEH record                         |
| 0012FF70 | 0040BFF4 | SE handler   |
| 0012FF74 | 0012FF78 |  |

No es sospechosa sigamos dando RUN hasta que pare en alguna sospechosa

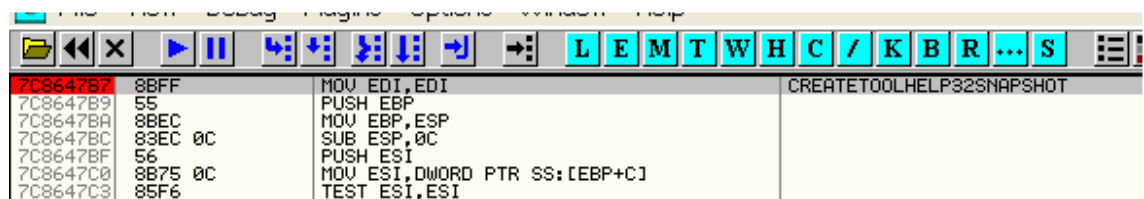
|          |          |  |
|----------|----------|--|
| 0012FE28 | 004725E9 | CALL to GetProcAddress from Antisoci.004725E4      |
| 0012FE2C | 7C800000 | hModule = 7C800000 (kernel32)                      |
| 0012FE30 | 00472724 | ProcNameOrOrdinal = "CreateToolhelp32Snapshot"     |
| 0012FE34 | 0000000F |  |
| 0012FE38 | 0047283F | RETURN to Antisoci.0047283F from Antisoci.004725B8 |
| 0012FE3C | 0012FE78 |  |
| 0012FE40 | 00478D7C | Antisoci.00478D7C                                  |
| 0012FE44 | 00472EAB | RETURN to Antisoci.00472EAB from Antisoci.00472834 |

Aquí vemos la primera sospechosa la api que nos hace la foto de todos los procesos que están corriendo en nuestra maquina, lleguemos al RET y pongamos un BP EAX ya que en EAX estará la dirección de la api en nuestra maquina.

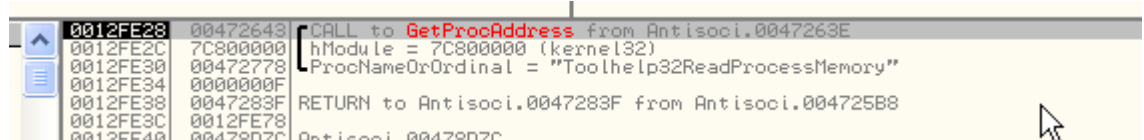
|          |                |                             |
|----------|----------------|-----------------------------|
| 7C80AC84 | 5F             | POP EDI                     |
| 7C80AC85 | 5B             | POP EBX                     |
| 7C80AC86 | C9             | LEAVE                       |
| 7C80AC87 | C2 0800        | RET 8                       |
| 7C80AC8A | 837D 10 00     | CMP DWORD PTR SS:[EBP+10],0 |
| 7C80AC8E | ^0F85 81E6FFFF | JNZ kernel32.7C809315       |
| 7C80AC94 | 33FF           | XOR EDI,EDI                 |



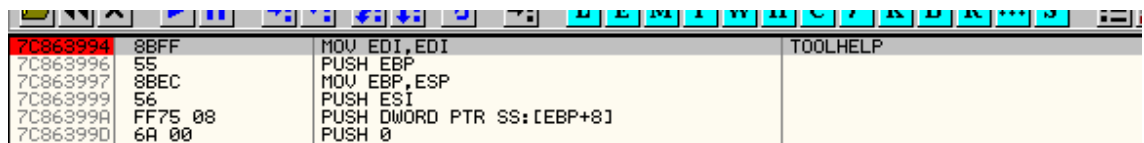
Allí esta el BP le pondré el comentario para saber que api es



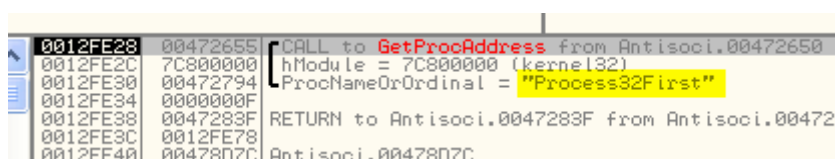
Listo continuemos



Hmm otra parecida a la anterior por si acaso pongámosle un BP también



Demos RUN



Bueno esta es conocida BP en ella

|          |               |                                 |       |
|----------|---------------|---------------------------------|-------|
| 7C863A8D | 8BFF          | MOV EDI,EDI                     | FIRST |
| 7C863A8F | 55            | PUSH EBP                        |       |
| 7C863A90 | 8BEC          | MOV EBP,ESP                     |       |
| 7C863A92 | 81EC 30020000 | SUB ESP,230                     |       |
| 7C863A98 | A1 CC36887C   | MOV EAX,DWORD PTR DS:[7C8836CC] |       |

|          |          |  |  |
|----------|----------|--|--|
| 0012FE28 | 00472667 | CALL to <b>GetProcAddress</b> from Antisoci.00472662 |  |
| 0012FE2C | 7C800000 | hModule = 7C800000 (kernel32)                        |  |
| 0012FE30 | 004727A4 | ProcNameOrOrdinal = "Process32Next"                  |  |
| 0012FE34 | 0000000F |  |  |
| 0012FE38 | 0047283F | RETURN to Antisoci.0047283F from Antisoci.004725B8   |  |
| 0012FE3C | 0012FE78 |  |  |

Lo mismo

|          |               |                                 |      |
|----------|---------------|---------------------------------|------|
| 7C863C00 | 8BFF          | MOV EDI,EDI                     | NEXT |
| 7C863C02 | 55            | PUSH EBP                        |      |
| 7C863C03 | 8BEC          | MOV EBP,ESP                     |      |
| 7C863C05 | 81EC 30020000 | SUB ESP,230                     |      |
| 7C863C0B | A1 CC36887C   | MOV EAX,DWORD PTR DS:[7C8836CC] |      |

|          |          |  |  |
|----------|----------|--|--|
| 0012FE28 | 00472679 | CALL to <b>GetProcAddress</b> from Antisoci.00472674 |  |
| 0012FE2C | 7C800000 | hModule = 7C800000 (kernel32)                        |  |
| 0012FE30 | 004727B4 | ProcNameOrOrdinal = "Process32FirstW"                |  |
| 0012FE34 | 0000000F |  |  |
| 0012FE38 | 0047283F | RETURN to Antisoci.0047283F from Antisoci.004725B8   |  |
| 0012FE3C | 0012FE78 |  |  |

|          |          |  |  |
|----------|----------|--|--|
| 0012FE28 | 0047268B | CALL to <b>GetProcAddress</b> from Antisoci.00472686 |  |
| 0012FE2C | 7C800000 | hModule = 7C800000 (kernel32)                        |  |
| 0012FE30 | 004727C4 | ProcNameOrOrdinal = "Process32NextW"                 |  |
| 0012FE34 | 0000000F |  |  |
| 0012FE38 | 0047283F | RETURN to Antisoci.0047283F from Antisoci.004725B8   |  |

|          |          |  |  |
|----------|----------|--|--|
| 0012FE30 | 0047284B | CALL to <b>CreateToolhelp32Snapshot</b> from Antisoci.00472844 |  |
| 0012FE34 | 0000000F | Flags = TH32CS_SNAPALL   |  |
| 0012FE38 | 00000000 | ProcessID = 0  |  |
| 0012FE3C | 0012FE78 |  |  |
| 0012FE40 | 00478D7C | Antisoci.00478D7C  |  |

Allí para en la api que saca foto y como toda la protección depende de la foto, desde la cual de donde trabaja con la lista de procesos, threads, y no se cuantas cosas mas testea todo de la bendita foto, podemos intentar parchear esta api de la foto para que no devuelva el handle de la misma sino cero, para ellos vamos al ret de la api.

|          |         |                              |  |
|----------|---------|------------------------------|--|
| 7C86482D | 83C8 FF | OR EAX,FFFFFFFF              |  |
| 7C864830 | EB 03   | JMP SHORT kernel32.7C864835  |  |
| 7C864832 | 8B45 0C | MOV EAX,DWORD PTR SS:[EBP+C] |  |
| 7C864835 | 5E      | POP ESI                      |  |
| 7C864836 | C9      | LEAVE                        |  |
| 7C864837 | C2 0800 | RETN 8                       |  |
| 7C86483A | 90      | NOP                          |  |
| 7C86483B | 90      | NOP                          |  |
| 7C86483C | 90      | NOP                          |  |
| 7C86483D | 90      | NOP                          |  |
| 7C86483E | 90      | NOP                          |  |
| 7C86483F | 8BFF    | MOV EDI,EDI                  |  |
| 7C864841 | 55      | PUSH EBP                     |  |

Vemos que allí hay un espacio vacío así que podemos hacer EAX igual a cero antes de volver

|          |         |                              |  |
|----------|---------|------------------------------|--|
| 7C86482D | 83C8 FF | OR EAX,FFFFFFFF              |  |
| 7C864830 | EB 03   | JMP SHORT kernel32.7C864835  |  |
| 7C864832 | 8B45 0C | MOV EAX,DWORD PTR SS:[EBP+C] |  |
| 7C864835 | 5E      | POP ESI                      |  |
| 7C864836 | C9      | LEAVE                        |  |
| 7C864837 | 33C0    | XOR EAX,EAX                  |  |
| 7C864839 | C2 0800 | RETN 8                       |  |
| 7C86483C | 90      | NOP                          |  |
| 7C86483D | 90      | NOP                          |  |
| 7C86483E | 90      | NOP                          |  |
| 7C86483F | 8BFF    | MOV EDI,EDI                  |  |

Con eso devolverá cero y el programa no tendrá handle para poder manejar las fotos ni averiguar sobre ellas, esa es una posibilidad la otra cambiando en el código del programa llegamos al ret de la api la cual no modificamos.

|          |         |                              |
|----------|---------|------------------------------|
| 7C864830 | EB 03   | JMP SHORT kernel32.7C864835  |
| 7C864832 | 8B45 0C | MOV EAX,DWORD PTR SS:[EBP+C] |
| 7C864835 | 5E      | POP ESI                      |
| 7C864836 | C9      | LEAVE                        |
| 7C864837 | C2 0800 | RETN 8                       |
| 7C86483A | 90      | NOP                          |
| 7C86483B | 90      | NOP                          |
| 7C86483C | 90      | NOP                          |
| 7C86483D | 90      | NOP                          |
| 7C86483E | 90      | NOP                          |

Allí vemos un poco mas abajo unos saltos JNZ y justo abajo los call a TerminateProcess que cerrarian el OLLYDBG con la llamada anterior a OpenProcess para obtener el handle, y vemos que hay como cinco de estas partes justo una debajo de otra

|          |               |                                       |  |  |  |
|----------|---------------|---------------------------------------|--|--|--|
|          |               |                                       | L E M T W H C / K B R ... S  |  |  |
| 00472EF7 | 75 15         | JNZ SHORT jejebuen.00472F0E           | ExitCode = 0   |  |  |
| 00472EF9 | 6A 00         | PUSH 0                                | ProcessId<br>Inheritable = TRUE<br>Access = TERMINATE<br>OpenProcess<br>hProcess<br>TerminateProcess<br>ASCII "FVF/HCEZMMP"  |  |  |
| 00472EFB | 8B46 08       | MOV EAX,DWORD PTR DS:[ESI+8]          |  |  |  |
| 00472EFE | 50            | PUSH EAX                              |  |  |  |
| 00472EFF | 6A FF         | PUSH -1                               |  |  |  |
| 00472F01 | 6A 01         | PUSH 1                                |  |  |  |
| 00472F03 | E8 883BF9FF   | CALL <JMP.&kernel32.OpenProcess>      |  |  |  |
| 00472F08 | 50            | PUSH EAX                              |  |  |  |
| 00472F09 | E8 EA3BF9FF   | CALL <JMP.&kernel32.TerminateProcess> |  |  |  |
| 00472F0E | 8D95 C8FEFFFI | LEA EDX,DWORD PTR SS:[EBP-138]        |  |  |  |
| 00472F14 | B8 84304700   | MOV EAX,jejebuen.00473084             |  |  |  |
| 00472F19 | E8 86FEFFFF   | CALL jejebuen.00472DA4                |  |  |  |
| 00472F1E | 8B95 C8FEFFFI | MOV EDX,DWORD PTR SS:[EBP-138]        |  |  |  |
| 00472F24 | 8B45 FC       | MOV EAX,DWORD PTR SS:[EBP-4]          |  |  |  |
| 00472F27 | E8 C418F9FF   | CALL jejebuen.004047F0                |  |  |  |
| 00472F2C | 75 15         | JNZ SHORT jejebuen.00472F43           | ExitCode = 0   |  |  |
| 00472F2E | 6A 00         | PUSH 0                                | ProcessId<br>Inheritable = TRUE<br>Access = TERMINATE<br>OpenProcess<br>hProcess<br>TerminateProcess<br>ASCII "FVF/1113XSU"  |  |  |
| 00472F30 | 8B46 08       | MOV EAX,DWORD PTR DS:[ESI+8]          |  |  |  |
| 00472F33 | 50            | PUSH EAX                              |  |  |  |
| 00472F34 | 6A FF         | PUSH -1                               |  |  |  |
| 00472F36 | 6A 01         | PUSH 1                                |  |  |  |
| 00472F38 | E8 533BF9FF   | CALL <JMP.&kernel32.OpenProcess>      |  |  |  |
| 00472F3D | 50            | PUSH EAX                              |  |  |  |
| 00472F3E | E8 B53BF9FF   | CALL <JMP.&kernel32.TerminateProcess> |  |  |  |
| 00472F43 | 8D95 C4FEFFFI | LEA EDX,DWORD PTR SS:[EBP-13C]        |  |  |  |
| 00472F49 | B8 98304700   | MOV EAX,jejebuen.00473098             |  |  |  |
| 00472F4E | E8 51FEFFFF   | CALL jejebuen.00472DA4                |  |  |  |
| 00472F53 | 8B95 C4FEFFFI | MOV EDX,DWORD PTR SS:[EBP-13C]        |  |  |  |
| 00472F59 | 8B45 FC       | MOV EAX,DWORD PTR SS:[EBP-4]          |  |  |  |
| 00472F5C | E8 8F18F9FF   | CALL jejebuen.004047F0                |  |  |  |
| 00472F61 | 75 15         | JNZ SHORT jejebuen.00472F78           | ExitCode = 0   |  |  |
| 00472F63 | 6A 00         | PUSH 0                                | ProcessId<br>Inheritable = TRUE<br>Access = TERMINATE<br>OpenProcess<br>hProcess<br>TerminateProcess<br>ASCII "FVF/34SFE8PM" |  |  |
| 00472F65 | 8B46 08       | MOV EAX,DWORD PTR DS:[ESI+8]          |  |  |  |
| 00472F68 | 50            | PUSH EAX                              |  |  |  |
| 00472F69 | 6A FF         | PUSH -1                               |  |  |  |
| 00472F6B | 6A 01         | PUSH 1                                |  |  |  |
| 00472F6D | E8 1E3BF9FF   | CALL <JMP.&kernel32.OpenProcess>      |  |  |  |
| 00472F72 | 50            | PUSH EAX                              |  |  |  |
| 00472F73 | E8 803BF9FF   | CALL <JMP.&kernel32.TerminateProcess> |  |  |  |
| 00472F78 | 8D95 C0FEFFFI | LEA EDX,DWORD PTR SS:[EBP-140]        |  |  |  |
| 00472F7E | B8 AC304700   | MOV EAX,jejebuen.004730AC             |  |  |  |
| 00472F83 | E8 1CFEFFFF   | CALL jejebuen.00472DA4                |  |  |  |
| 00472F88 | 8B95 C0FEFFFI | MOV EDX,DWORD PTR SS:[EBP-140]        |  |  |  |
| 00472F8E | 8B45 FC       | MOV EAX,DWORD PTR SS:[EBP-4]          |  |  |  |

Por supuesto cambiando todos los saltos JNZ por JMP, evitamos que el programa llegue a TerminateProcess evitamos y que cierre el OLLYDBG.

Esta primera parte esta solucionada ahora sigamos con la segunda parte de la protección, demos RUN

|          |            |                             |
|----------|------------|-----------------------------|
| 004730EA | CD 68      | INT 68                      |
| 004730EC | 66:3D 86F3 | CMP AX,0F386                |
| 004730F0 | 75 07      | JNZ SHORT Antisoci.004730F9 |
| 004730F2 | 66:B8 FE00 | MOV AX,0FE                  |
| 004730F6 | 66:E7 64   | OUT 64,AX                   |
| 004730F9 | C3         | RETN                        |

Nopeamos y damos RUN y corre pero se cierra el programa, si le pongo solo el plugin HideDebugger 1.23f y lo protejo contra FindWindows/EnumWindows corre bien con los pasos anteriores que vimos.

El tema del lugar de donde se cierra es aquí

|          |               |                                |  |
|----------|---------------|--------------------------------|--|
| 004732EB | . 8B95 DCFEFF | MOV EDX,DWORD PTR SS:[EBP-124] |  |
| 004732F1 | . 58          | POP EAX                        |  |
| 004732F2 | . E8 F914F9FF | CALL jejebuen.004047F0         |  |
| 004732F7 | 75 0C         | JNZ SHORT jejebuen.00473305    |  |
| 004732F9 | A1 DC6E4700   | MOV EAX,DWORD PTR DS:[476EDC]  |  |
| 004732FE | 8B00          | MOV EAX,DWORD PTR DS:[EAX]     |  |
| 00473300 | E8 C7FFDFFF   | CALL jejebuen.004532CC         |  |
| 00473305 | 8D85 D4FEFF   | LEA EAX,DWORD PTR SS:[EBP-12C] |  |
| 0047330B | 8B06          | MOV EDX,ESI                    |  |
| 0047330D | B9 00010000   | MOV ECX,100                    |  |

Cuando corremos el programa cuando para alli en el call superior.

|          |               |                            |  |
|----------|---------------|----------------------------|--|
| 004047F0 | 53            | PUSH EBX                   |  |
| 004047F1 | 56            | PUSH ESI                   |  |
| 004047F2 | 57            | PUSH EDI                   |  |
| 004047F3 | 89C6          | MOV ESI,EAX                |  |
| 004047F5 | 89D7          | MOV EDI,EDX                |  |
| 004047F7 | 39D0          | CMP EAX,EDX                |  |
| 004047F9 | 0F84 8F000000 | JE jejebuen.0040488E       |  |
| 004047FF | 85F6          | TEST ESI,ESI               |  |
| 00404801 | 74 50         | JE SHORT jejebuen.0040485B |  |

| Registers (FPU) |                            |
|-----------------|----------------------------|
| EAX             | 00D923F8                   |
| ECX             | FFFFFFFF                   |
| EDX             | 00D9242C ASCII "OllYDbg"   |
| EBX             | 01680468                   |
| ESP             | 0012FE44                   |
| EBP             | 0012FFA4                   |
| ESI             | 00D923F8                   |
| EDI             | 00D9242C ASCII "OllYDbg"   |
| EIP             | 004047F7 jejebuen.004047F7 |
| C 0             | ES 0023 32bit 0(FFFFFFFF)  |
| P 0             | CS 001B 32bit 0(FFFFFFFF)  |
| A 0             | SS 0023 32bit 0(FFFFFFFF)  |
| 3 0             | DS 0023 32bit 0(FFFFFFFF)  |

Vemos que en una de esas tantas comparaciones, que para alli.

| Registers (FPU) |                            |
|-----------------|----------------------------|
| EAX             | 00D92EA0 ASCII "OllYDbg"   |
| ECX             | FFFFFFFF                   |
| EDX             | 00D92ED4 ASCII "OllYDbg"   |
| EBX             | 020304BE                   |
| ESP             | 0012FE44                   |
| EBP             | 0012FFA4                   |
| ESI             | 00D92EA0 ASCII "OllYDbg"   |
| EDI             | 00D92ED4 ASCII "OllYDbg"   |
| EIP             | 004047F7 jejebuen.004047F7 |

|        |          |  |  |
|--------|----------|--|--|
| 12FE44 | 7C920738 | ntdll.7C920738   |  |
| 12FE48 | 0012FEA4 | ASCII "OllYDbg - jejebueno.exe - [CPU - main thread, module jejebuen]" |  |
| 12FE4C | 020304BE |  |  |
| 12FE50 | 004732F7 | RETURN to jejebuen.004732F7 from jejebuen.004047F0                     |  |
| 12FE54 | 0012FFB4 | Pointer to next SEH record   |  |
| 12FE58 | 004733E7 | SE handler   |  |
| 12FE5C | 0012FFA4 |  |  |
| 12FE60 | FFFFFFFF |  |  |
| 12FE64 | 00478D7C | jejebuen.00478D7C  |  |
| 12FE68 | 00D92CDC | ASCII "URSoft"   |  |
| 12FE6C | 00D92CA4 | ASCII "WinDowse"   |  |
| 12FE70 | 00D908EC | ASCII "WinDow"   |  |
| 12FE74 | 00D92C90 | ASCII "DeDe"   |  |
| 12FE78 | 00D92C44 | ASCII "WinDowse"   |  |
| 12FE7C | 00D92C5C | ASCII "WinD"   |  |
| 12FE80 | 00D92ED4 | ASCII "OllYDbg"  |  |
| 12FE84 | 00D92E54 | ASCII "OllYDbg - jejebueno.exe - [CPU - main thread, module jejebuen]" |  |
| 12FE88 | 00D92EA0 | ASCII "OllYDbg"  |  |
| 12FE8C | 00D92E38 | ASCII "Symbol Loader"  |  |
| 12FE90 | 00D92DCC | ASCII "OllYDbg - jejebueno.exe - [CPU - main thread, module jejebuen]" |  |
| 12FE94 | 00D91BE8 | ASCII "ule jejebuen]"  |  |
| 12FE98 | 00D92DA8 | ASCII "TRW2000 for Windows 9x"   |  |
| 12FE9C | 00D92D3C | ASCII "OllYDbg - jejebueno.exe - [CPU - main thread, module jejebuen]" |  |
| 12FEA0 | 00D92CF0 | ASCII "OllYDbg - jejebueno.exe - [CPU - main thread, module jejebuen]" |  |
| 12FEA4 | 796C6C4F |  |  |
| 12FEA8 | 20676244 |  |  |
| 12FEAC | 2F000000 |  |  |

La cuestión, es que aquí compara si hay cosas malas, jeje y que todo eso se evita con el jnz que esta a la salida del primer CALL si lo cambio por JMP

|          |             |                                |
|----------|-------------|--------------------------------|
| 004732F2 | E8 F914F9FF | CALL jejebuen.004047F0         |
| 004732F7 | 75 0C       | JNZ SHORT jejebuen.00473305    |
| 004732F9 | A1 DC6E4700 | MOV EAX,DWORD PTR DS:[476EDC]  |
| 004732FE | 8B00        | MOV EAX,DWORD PTR DS:[EAX]     |
| 00473300 | E8 C7FFDFFF | CALL jejebuen.004532CC         |
| 00473305 | 8D85 D4FEFF | LEA EAX,DWORD PTR SS:[EBP-12C] |
| 0047330B | 8BD6        | MOV EDI,ESI                    |
| 0047330D | B9 00010000 | MOV ECX,100                    |
| 00473312 | E8 4513F9FF | CALL jejebuen.0040465C         |

Ese es el salto clave si lo cambio por JMP

|          |             |                                |
|----------|-------------|--------------------------------|
| 004732F2 | E8 F914F9FF | CALL jejebuen.004047F0         |
| 004732F7 | EB 0C       | JMP SHORT jejebuen.00473305    |
| 004732F9 | A1 DC6E4700 | MOV EAX,DWORD PTR DS:[476EDC]  |
| 004732FE | 8B00        | MOV EAX,DWORD PTR DS:[EAX]     |
| 00473300 | E8 C7FFDFFF | CALL jejebuen.004532CC         |
| 00473305 | 8D85 D4FEFF | LEA EAX,DWORD PTR SS:[EBP-12C] |
| 0047330B | 8BD6        | MOV EDI,ESI                    |
| 0047330D | B9 00010000 | MOV ECX,100                    |

Evita el segundo call que si lo vemos dentro

|          |             |                                    |
|----------|-------------|------------------------------------|
| 004532CC | E8 FB7FFBFF | CALL jejebuen.0040B2CC             |
| 004532D1 | 84C0        | TEST AL,AL                         |
| 004532D3 | 74 07       | JE SHORT jejebuen.004532DC         |
| 004532D5 | 6A 00       | PUSH 0                             |
| 004532D7 | E8 043EFBFF | CALL <JMP.&user32.PostQuitMessage> |
| 004532DC | C3          | RETN                               |
| 004532DD | 8D40 00     | LEA EAX,DWORD PTR DS:[EAX]         |
| 004532E0 | 53          | PUSH EBX                           |

Es el que te lleva a PostQuitMessage y va cerrando la ventana pues esta por terminar el programa lo cual lo hace mas adelante, pero aquí ya la decisión esta tomada, el PostQuitMessage es que se va a cerrar el loop de mensajes y no queda otra que el crackme se cierre ya que no tiene mas ventanas.

Por supuesto si me dicen como halle ese JNZ pues es fácil.

Una vez que paso la primera parte de la protección, pongo un BPX en PostQuitMessage.

Command: Bp PostQuitMessage

Breakpoint at user32.PostQuitMessage

|          |             |                           |                |
|----------|-------------|---------------------------|----------------|
| 77D21211 | 8BFF        | MOV EDI,EDI               | ntdll.7C920738 |
| 77D21213 | 55          | PUSH EBP                  |                |
| 77D21214 | 8BEC        | MOV EBP,ESP               |                |
| 77D21216 | 6A 33       | PUSH 33                   |                |
| 77D21218 | FF75 08     | PUSH DWORD PTR SS:[EBP+8] |                |
| 77D2121B | E8 8672FFFF | CALL user32.77D184A6      |                |
| 77D21220 | 5D          | POP EBP                   |                |
| 77D21221 | C2 0400     | RETN 4                    |                |
| 77D21224 | 33C0        | XOR EAX,EAX               |                |
| 77D21226 | 40          | INC EAX                   |                |
| 77D21227 | E9 EA230000 | JMP user32.77D23616       |                |
| 77D2122C | 33C0        | XOR EAX,EAX               |                |
| 77D2122E | E9 7DC5FFFF | JMP user32.77D1D7B0       |                |
| 77D21233 | 90          | NOP                       |                |
| 77D21234 | 90          | NOP                       |                |
| 77D21235 | 90          | NOP                       |                |

Para en la api miro el stack a ver de donde fue llamado



```

0012FE48 004532DC CALL to PostQuitMessage from jejebuen.004532D7
0012FE4C 00000000 ExitCode = 0
0012FE50 00473305 RETURN to jejebuen.00473305 from jejebuen.004532CC
0012FE54 0012FFB4 Pointer to next SEH record
0012FE58 004733E7 SE handler
0012FE5C 0012FFA4
0012FE60 FFFFFFFF
0012FE64 00478D7C jejebuen.00478D7C
0012FE68 00D90BEC ASCII "URSoft"
0012FE6C 00D92C78 ASCII "WinDowse"
0012FE70 00D93C98 ASCII "WinDowse"

```

Voy a 4532d7 ya que el stack me dice que fue llamado de alli

```

004532C9 . 5D POP EBP
004532CA . C3 RETN
004532CB . 90 NOP
004532CC . E8 FB7FFBFF CALL jejebuen.0040B2CC
004532D1 . 84C0 TEST AL,AL
004532D3 . 74 07 JE SHORT jejebuen.004532DC
004532D5 . 6A 00 PUSH 0
004532D7 . E8 043EFBFF CALL <JMP.&user32.PostQuitMessage>
004532DC . C3 RETN
004532DD . 8D40 00 LEA EAX,DWORD PTR DS:[EAX]
004532E0 . 53 PUSH EBX
004532E1 . 56 PUSH ESI
004532E2 . 55 PUSH EBP

```

Por supuesto probé ese JE que esta antes del PostQuitMessage y no evitaba que se cierre al invertirlo, así que fui al segundo RETURN TO que encontré en el stack.

```

0012FE48 004532DC CALL to PostQuitMessage from jejebuen.004532D7
0012FE4C 00000000 ExitCode = 0
0012FE50 00473305 RETURN to jejebuen.00473305 from jejebuen.004532CC
0012FE54 0012FFB4 Pointer to next SEH record
0012FE58 004733E7 SE handler
0012FE5C 0012FFA4
0012FE60 FFFFFFFF
0012FE64 00478D7C jejebuen.00478D7C
0012FE68 00D90BEC ASCII "URSoft"
0012FE6C 00D92C78 ASCII "WinDowse"
0012FE70 00D93C98 ASCII "WinDowse"

```

Este viene de 4532CC

```

004732F1 . 58 POP EAX
004732F2 . E8 F914F9FF CALL jejebuen.004047F0
004732F7 . 75 0C JNZ SHORT jejebuen.00473305
004732F9 . A1 DC6E4700 MOV EAX,DWORD PTR DS:[476EDC]
004732FE . 8B00 MOV EAX,DWORD PTR DS:[EAX]
00473300 . E8 C7FFDFFF CALL jejebuen.004532CC
00473305 . 8D85 D4FEFF LEA EAX,DWORD PTR SS:[EBP-12C]
0047330B . 8BD6 MOV EDI,ESI
0047330D . B9 00100000 MOV ECX,100
00473312 . F8 4513F9FF CALL jejebuen.00404650

```

Como vemos este call se evita con el JNZ anterior con lo cual no entra y no va a PostQuitMessage y al cambiarlo por JMP veo que corre perfecto sin ningún plugin.

Adjunto el crackme desempacado que corre en cualquier OLLYDBG por supuesto ustedes aun no aprendieron a desempacar en este curso, por lo cual aun eso no se pide, pero si quieren verlo por curiosidad, allí esta adjunto pueden probar que corre en cualquier OLLYDBG se llame como se llame y sin ningún plugin, ya que además de desempacarlo, le hice los cambios permanentes que vimos en este tute y con eso quedo limbito como culito de bebe.

Hasta la parte 25 ahora si empezamos con las excepciones

Ricardo Narvaja

06 de enero de 2006