

## Aprovechando TrainerMe para practicar con IDA.

Como nos indica la **info.txt** de este crackme el cual lo podréis encontrar en el **concurso 19 de CrackSLatinoS**, esta compilado con **Delphi 6**, esto comporta que no podamos utilizar los puntos de ruptura en las APIS. Esto puede crearnos cierto problema si utilizamos a nuestro amigo **Oly**, pero podemos solventarlo de distintas formas con nuestro amigo **IDA**. Como ya sabemos IDA nos proporciona un listado con mucha más información que Oly, ya que utiliza los archivos de firmas **FLIRT**, lo cual permite que en lugar de ver solamente instrucciones **call** sin ninguna información en todo el desensamblado podamos ver nombres de funciones conocidas.

Por lo tanto una opción, sería utilizar la característica de Ida para crear “mapeados” de un archivo, una vez que IDA ha analizado el archivo creamos un archivo **.map** el cual podremos utilizar en Oly con el plugin **GODUP**.

¿Cómo lo realizaremos? Primero cargamos el archivo en IDA y éste lo analiza, una vez analizado y con la acción **File > Produce file > Create map file**, creamos el archivo **.map** del binario. Una vez tenemos el archivo map ejecutamos Oly y cargamos el TrainerME. Veamos primero una vista de Oly sin cargar el archivo **.map** creado por IDA:

Address	Hex dump	Disassembly	Comment
0044E6E8	55	PUSH EBP	
0044E6E9	8BEC	MOV EBP,ESP	
0044E6EB	83C4 F0	ADD ESP,-10	
0044E6EE	B8 70E54400	MOV EAX,0044E570	
0044E6F3	E8 CC7EFBFF	CALL 004065C4	
0044E6F8	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E6FD	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E6FF	E8 48E4FFFF	CALL 0044CB4C	
0044E704	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E709	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E70B	BA 48E74400	MOV EDI,0044E748	ASCII "WinFan"
0044E710	E8 5BE0FFFF	CALL 0044C770	
0044E715	8B00	MOV ECX,DWORD PTR DS:[44FBFC]	TrainerM.00450C20
0044E71B	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E720	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E722	8B15 68E24400	MOV EDI,DWORD PTR DS:[44E268]	TrainerM.0044E2B4
0044E728	E8 37E4FFFF	CALL 0044CB4C	
0044E72D	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E732	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E734	E8 ABE4FFFF	CALL 0044CBE4	
0044E739	E8 2A5AFBFF	CALL 00404168	
0044E73E	0000	ADD BYTE PTR DS:[EAX],AL	
0044E740	FFFFFFFF	DD FFFFFFFF	
0044E744	06000000	DD 00000006	
0044E748	57 69 6E 46 61 6E 00	ASCII "WinFan",0	
0044E74F	00	DB 00	
0044E750	00	DB 00	

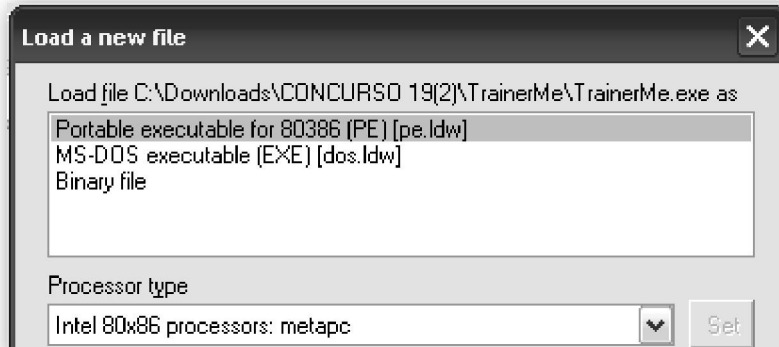
Como vemos sólo tenemos instrucciones **call** sin ningún tipo de información. Ahora seleccionemos en **Plugins > Godup > map loader > load labels**, busquemos el directorio donde está nuestro archivo **.map** y esta acción nos cargará todas las etiquetas del TrainerMe del archivo **.map** y las podremos ver así:



Address	Hex dump	Disassembly	Comment
0044E6E8	55	PUSH EBP	
0044E6E9	8BEC	MOV EBP,ESP	
0044E6EB	83C4 F0	ADD ESP,-10	
0044E6EE	B8 70E54400	MOV EAX,< dword_44E570 >	
0044E6F3	E8 CC7EFBFF	CALL < SysInit::_LinkProc_InitExe(void *) >	
0044E6F8	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E6FD	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E6FF	E8 48E4FFFF	CALL < sub_44CB4C >	
0044E704	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E709	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E70B	BA 48E74400	MOV EDI,0044E748	ASCII "WinFan"
0044E710	E8 5BE0FFFF	CALL < Forms::TApplication::SetTitle(System::Ans	
0044E715	8B00	MOV ECX,DWORD PTR DS:[44FBFC]	TrainerM.00450C20
0044E71B	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E720	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E722	8B15 68E24400	MOV EDI,DWORD PTR DS:[44E268]	< TrainerM._cls_uain_TTrainerForm >
0044E728	E8 37E4FFFF	CALL < Forms::TApplication::CreateForm(System::TI	
0044E72D	A1 ECFD4400	MOV EAX,DWORD PTR DS:[44FDEC]	
0044E732	8B00	MOV EAX,DWORD PTR DS:[EAX]	
0044E734	E8 ABE4FFFF	CALL < Forms::TApplication::Run(void) >	
0044E739	E8 2A5AFBFF	CALL < System::_LinkProc_Halt0(void) >	
0044E73E	0000	ADD BYTE PTR DS:[EAX],AL	
0044E740	FFFFFFFF	DD FFFFFFFF	
0044E744	06000000	DD 00000006	
0044E748	57 69 6E 46 61 6E 00	ASCII "WinFan",0	
0044E74F	00	DB 00	
0044E750	00	DB 00	

A partir de aquí podremos llegar a las mismas conclusiones que alcanzaremos con la siguiente opción, y que a mí me gusta más.

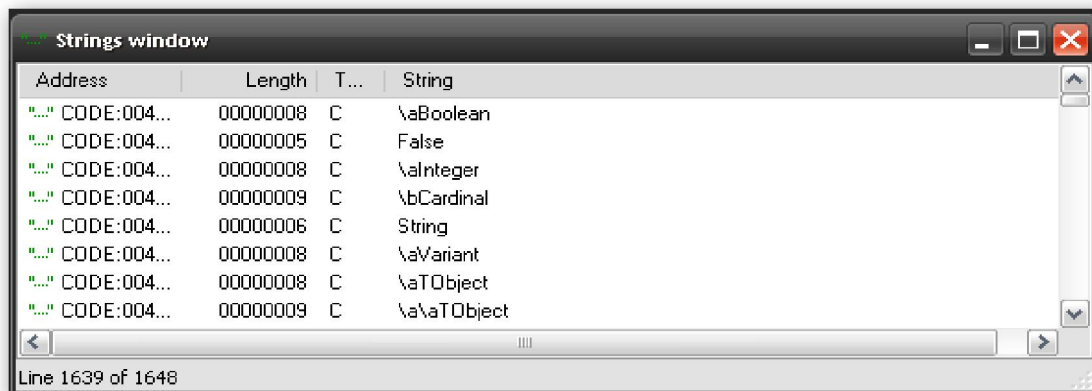
Otra opción es la siguiente, cargamos TrainerMe en IDA, y una vez cargado.



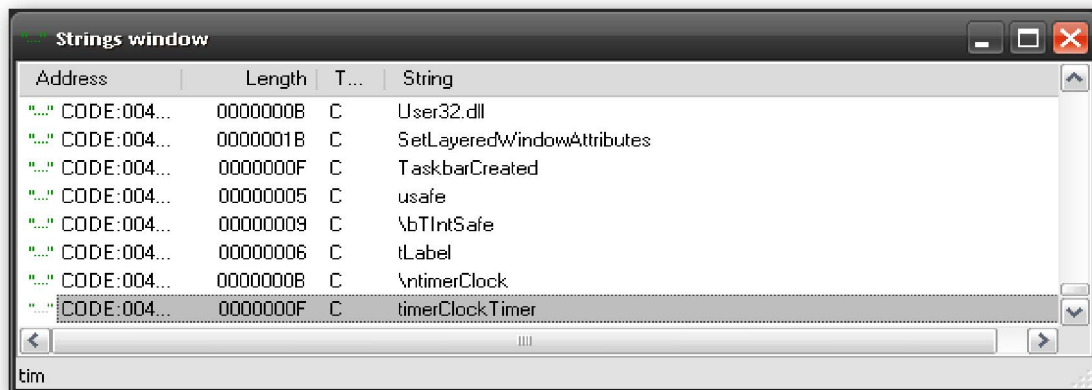
Vemos que nos reconoce las firmas como Delphi6-7, bien una vez cargado buscaremos

```
-----  
Using FLIRT signature: BDS 2005-2006 and Delphi6-7 Visual Component Library  
442DB5: can't create template name string  
Propagating type information...  
Function argument information has been propagated  
The initial autoanalysis has been finished.
```

alguna función que tenga relación con tiempo, esto es obvio ya que cuando ejecutamos TrainerMe vemos que se trata de un temporizador, en ingles algo parecido a time, timer, clock..., por lo tanto podemos buscar en la ventana **Strings** alguna cadena de este tipo lo haremos con la acción **View > Open subviews > Strings**



Una vez en la ventana realizamos una búsqueda tipo time, timer ... Como podemos ver al pulsar **tim** se nos coloca en una función tipo **timerClockTimer**,



hacemos doble click sobre ella y nos trasladamos a la siguiente ubicación

```

• CODE:0044E3CC dd offset _TTrainerForm_timerClockTimer
• CODE:0044E3D0 db 15,'timerClockTimer'

```

Otro boble click sobre dicho procedimiento y nos colocamos en su código

```

CODE:0044E470 ; ----- S U B R O U T I N E -----
CODE:0044E470 ; Attributes: bp-based frame
CODE:0044E470 ; TTrainerForm_timerClockTimer proc near ; DATA XREF: CODE:0044E3CCf0
CODE:0044E470 var_4 - dword ptr -4
CODE:0044E470 push ebp
CODE:0044E471 mov ebp, esp
CODE:0044E473 push 0
CODE:0044E475 push ebx
CODE:0044E476 mov ebx, eax
CODE:0044E478 xor eax, eax
CODE:0044E47A push ebp
CODE:0044E47B push offset loc_44E4D1
CODE:0044E480 push dword ptr fs:[eax]
CODE:0044E483 mov fs:[eax], esp
CODE:0044E486 mov eax, ebx
CODE:0044E488 call sub_44E44C
CODE:0044E48D mov edx, eax
CODE:0044E48F dec edx
CODE:0044E490 mov eax, ebx
CODE:0044E492 call sub_44E440
CODE:0044E497 mov eax, ebx
CODE:0044E499 call sub_44E458
CODE:0044E49E mov eax, ebx
CODE:0044E4A0 call sub_44E44C
CODE:0044E4A5 lea edx, [ebp+var_4]
CODE:0044E4A8 call @Sysutils@IntToStr@qqri ; Sysutils::IntToStr(int)
CODE:0044E4AB mov edx, [ebp+var_4]
CODE:0044E4AD mov eax, [ebx+2F0h]
CODE:0044E4B0 call @Controls@TControl@SetText@qqr*17System@AnsiString ; Controls::TContr
CODE:0044E4B8 xor eax, eax
CODE:0044E4BD pop edx
CODE:0044E4BE pop ecx
CODE:0044E4C0 pop ecx
CODE:0044E4C3 mov fs:[eax], edx
CODE:0044E4C8 push offset loc_44E4D8
CODE:0044E4C8 loc_44E4C8: ; CODE XREF: TTrainerForm_timerClockTimer+664j
CODE:0044E4C8 lea eax, [ebp+var_4]
CODE:0044E4CB call @System@LStrClr@qrpv ; System::__linkproc__LStrClr(void *)
CODE:0044E4D0 retn

```

Sabemos que en esta función es donde se manipula el tiempo del programa y como sabemos que éste va disminuyendo en uno si estudiamos el procedimiento veremos una instrucción de la cual podemos deducir que es la encargada de ir disminuyendo dicho tiempo, esta es:

```

• CODE:0044E48F dec edx

```

Vayamos ahora en vivo al Olly, comprobemos que es cierta nuestra suposición. Cargamos el programa en **Olly**, hacemos **Ctrl+G** tecleamos **0044E48F** y colocamos un punto de ruptura, hacemos **F9** y veamos:

Address	Hex dump	Disassembly	Registers (FPU)
0044E46F	90	MOV	EAX 00000078
0044E470	55	PUSH EBP	ECX 00000000
0044E471	3BC8	MOV EBP, ESP	EDX 00000078
0044E473	6A 00	PUSH 0	EBX 00081AD8
0044E475	53	PUSH EBX	ESP 0012FDE0
0044E476	8BD8	MOV EBX, EAX	EBP 0012FDF4
0044E478	33C0	XOR EAX, EAX	ESI 00426E00 TrainerM.00426E0C
0044E47A	55	PUSH EBP	EDI 0012FEA0
0044E47B	68 01E44400	PUSH 0044E4D1	EIP 0044E48F TrainerM.0044E48F
0044E480	64:FF30	PUSH DWORD PTR FS:[EAX]	C 1 ES 0023 32bit 0(FFFFFFFF)
0044E483	64:8920	MOV DWORD PTR FS:[EAX], ESP	P 0 CS 001B 32bit 0(FFFFFFFF)
0044E486	3BC3	MOV EAX, EBX	A 0 SS 0023 32bit 0(FFFFFFFF)
0044E488	E8 0FFFFFFF	CALL 0044E44C	Z 0 DS 0023 32bit 0(FFFFFFFF)
0044E48D	8BD0	MOV EDX, EAX	S 1 FS 003B 32bit 7FDF000(FFF)
0044E48F	4B	DEC EDX	T 0 GS 0000 NULL
0044E490	3BC3	MOV EAX, EBX	D 0
0044E492	E8 A9FFFFFF	CALL 0044E440	O 1 LastErr ERROR_SUCCESS (00000000)
0044E497	3BC3	MOV EAX, EBX	EFL 0000A033 (O,B,NE,BE,S,PO,GE,G)
0044E499	E8 0AFFFFFF	CALL 0044E458	ST0 empty 3.2548216060144286720e-32
0044E49E	E8 07FFFFFF	MOV EAX, EBX	ST1 empty 1.0373070717531805696e-29
0044E4A0	E8 A7FFFFFF	CALL 0044E44C	ST2 empty 131.18925804927695360
0044E4A5	8D55 FC	LEA EDX, [LOCAL.1]	ST3 empty 1581.2318339100344320
0044E4A8	E8 4B9EFBFF	CALL 004082F8	ST4 empty 8081.8515955401758720
0044E4AD	3B55 FC	MOV EDX, [LOCAL.1]	ST5 empty 0.0
0044E4B0	3B53 F002000	MOV EAX, DWORD PTR DS:[EBX+2F0]	ST6 empty 0.0
0044E4B6	E8 1EFFFDFD	CALL 0040C99C	ST7 empty 15.000000000000000000
0044E4BB	33C0	XOR EAX, EAX	3 2 1 0 E S P U O Z D I
0044E4BD	5A	POP EDX	FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 (GT)
0044E4BE	59	POP ECX	FCW 1372 Prec NEAR,64 Mask 1 1 0 0 1 0
0044E4C0	59	POP ECX	
0044E4C3	64:8910	MOV DWORD PTR FS:[EAX], EDX	
0044E4C8	68 01E44400	PUSH 0044E4D8	
0044E4CB	> 8D45 FC	LEA EAX, [LOCAL.1]	
0044E4C8	E8 885DFBFF	CALL 00408258	
0044E4D0	C3	RET	

Olly para en nuestro **BP** y observamos que en el registro **EDX** tenemos el valor **78 hexa = 120 dec**, el valor del contador del timer, con lo cual si cambiamos dicho valor cambiaremos el valor del timer.

Ahora bien si queremos obtener el primer lugar en que se pasa dicho valor haremos lo siguiente, cuando ejecutamos el TrainerMe, nos damos cuenta que el valor del tiempo ya está en el **Form**. Esto por lógica nos indica que éste es colocado en el momento de creación del form "**FormCreate**", averigüemos si existe un procedimiento con dicho nombre o parecido. Si recordamos cuando estábamos en la ventana **Strings** buscando el timer al hacer el primer doble click se nos desplazamos a esta ubicación:

```

* CODE:0044E3CC      dd offset _TTrainerForm_timerClockTimer
* CODE:0044E3D0      db 15,'timerClockTimer'
* CODE:0044E3E0      dw 11h
* CODE:0044E3E2      dd offset _TTrainerForm_FormCreate
* CODE:0044E3E6      db 10,'FormCreate'
* CODE:0044E3F1      dw 12h
* CODE:0044E3F3      dd offset _TTrainerForm_FormDestroy
* CODE:0044E3F7      db 11,'FormDestroy'

```

Si nos fijamos existe el procedimiento **FormCreate** y también **FormDestroy**, hagamos doble click en **FormCreate**, nos conduce hasta aquí:

```

CODE:0044E4DC ; ===== S U B R O U T I N E =====
CODE:0044E4DC
CODE:0044E4DC
CODE:0044E4DC TTrainerForm_FormCreate proc near ; DATA XREF: CODE:0044E3E2↑
* CODE:0044E4DC      push     ebx
* CODE:0044E4DD      mov     ebx, eax
* CODE:0044E4DF      mov     ecx, 400h
* CODE:0044E4E4      mov     dl, 1
* CODE:0044E4E6      mov     eax, off_44E058
* CODE:0044E4E8      call   sub_44E0C0
* CODE:0044E4F0      mov     [ebx+2F8h], eax
* CODE:0044E4F6      mov     edx, 78h
* CODE:0044E4FB      mov     eax, ebx
* CODE:0044E4FD      call   sub_44E440
* CODE:0044E502      pop     ebx
* CODE:0044E503      retn
CODE:0044E503 TTrainerForm_FormCreate endp
CODE:0044E503
CODE:0044E504

```

Veamos estudiemos el procedimiento, justo en la dirección **0044E4F6** vemos que a **EDX** se le pasa el valor **78h=120d**, con lo cual si buscamos dicha dirección en el Olly, colocamos un punto de ruptura, ejecutamos y en el momento que pare le cambiamos el valor **78** por el valor **F1ACA** tendremos un contador de **989898** segundos, comprobémoslo:

0044E4DF	B9 0040000	MOV ECX,400
0044E4E4	B2 01	MOV DL,1
0044E4E6	A1 58E04400	MOV EAX,DWORD PTR DS:[44E058]
0044E4E8	E8 00BFFFFF	CALL 0044E0C0
0044E4F0	8993 F020000	MOV DWORD PTR DS:[EBX+2F8],EAX
0044E4F6	B6 F800000	MOV EDI,78
0044E4F8	8BC3	MOV EAX,EBX
0044E4FD	E8 3EFFFFFF	CALL 0044E440
0044E502	5B	POP EBX
0044E503	C3	RET
0044E504	05 F020000	ADD EAX,2F8
0044E509	E8 BACEBFFF	CALL 0040B9C8
0044E50E	C3	RET

Assemble at 0044E4F6

MOV EDI,0F1ACA

Unknown identifier

Fill with NOP's

Assemble Cancel



Los segundos que faltan hasta 989898, son los que he tardado en tomar la foto jeje. Bueno todas estas líneas son lubricaciones mentales, que se escriben para que no se pierdan en la nada. Si a alguien le sirve para saber cosas nuevas perfecto.

Saludos.  
Bigundill@