



TARGET	GOOGLE MAPS DONWLOADER
PROTECCION	NINGUNA
DIFICULTAD	DEBE SER FÁCIL.. SI PUEDE HACERLO YO!?
HERRAMIENTAS	OLLYDBG - OPCIONAL: STUD_PE
COMPILADOR	C++
CRACKER	GUILLE3000 - (ALIAS WASSER)

ADVERTENCIA:

Al abrir este documento, el lector acepta incondicionalmente su total y exclusiva responsabilidad legal, de acuerdo a las leyes vigentes en su país, por el uso de las técnicas experimentales, educativas y/o de investigación aquí vertidas en materia de programación especializada de computadoras. En caso de discrepar con alguno de los puntos descritos, deberá eliminar inmediatamente el presente documento y todo material asociado al mismo.

Este tutorial es solo a los fines didáctico y como conocimiento de la informática en general, el autor de esta obra NO se hace responsable del uso indebido e ilegal de los conocimientos aquí explicados.

Introduccion

Este programa lo había descargado hace mucho tiempo y nunca había podido medicarlo, solo había logrado llegar hasta la primer parte que luego veremos. Así que hice lo que una vez dijo Ricardo.. más o menos era así: “..si no puedes con un programa, lo dejas y después de un tiempo vuelves a intentarlo. Si lo consigues es porque has aprendido algo.”

Nuestro proggie es uno que se llama “google maps downloader”

“ Google Maps Images Downloader graba al disco las fotografías del servidor Google Maps. Funciona introduciendo la posición de las cuatro coordenadas que deberán limitar la imagen.

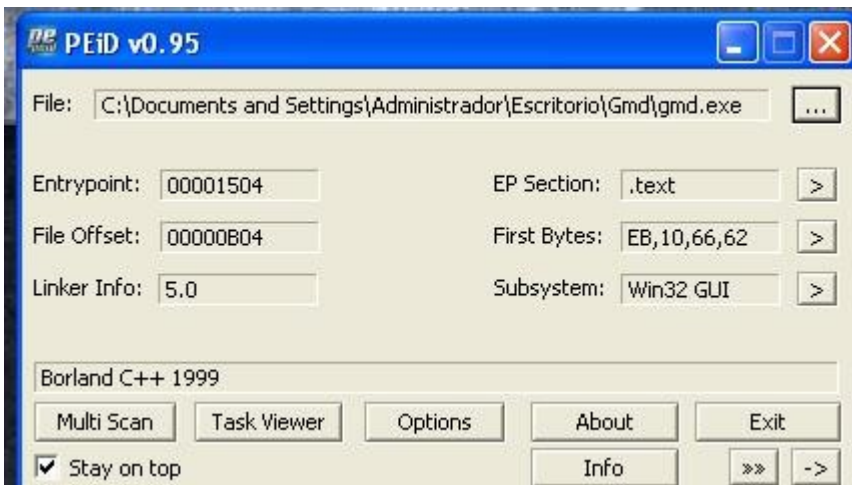
De esta manera, obtendremos un archivo de la parte del terreno que nos interesa guardar. Google Maps Images Downloader incluye un visualizador y un combinador de imágenes para fusionar varios archivos.

En resumen, Google Maps Images Downloader facilita la labor de interactuar con las fotografías de Google Maps para utilizarlas con otros programas aprovechando todas las funcionalidades de este fantástico servidor.”

Analizando:

Una vez que hemos instalado nuestro proggie vamos a la carpeta donde se encuentra el ejecutable principal y lo analizamos con el detector que deseemos, en mi caso siempre uso todos los que poseo para evitar sorpresas.

Con lo cual vemos que todos llegan a la misma conclusión, no está empaquetado ni nada por el estilo, solo código limpio



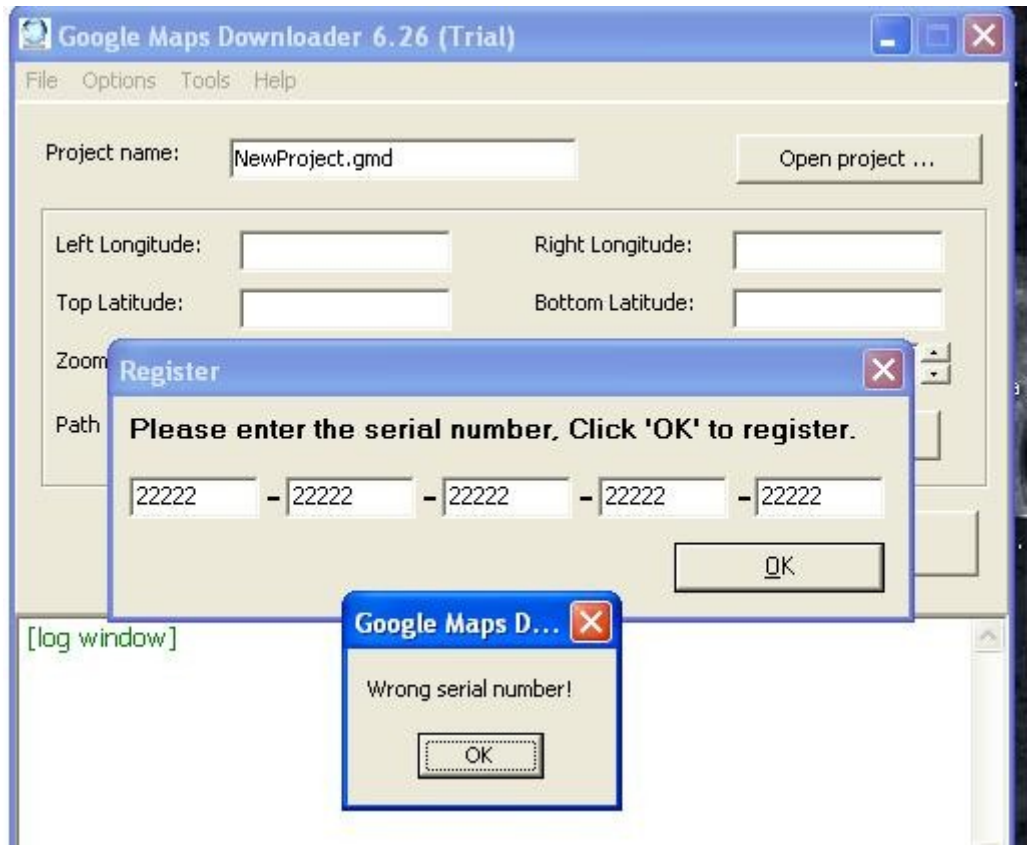
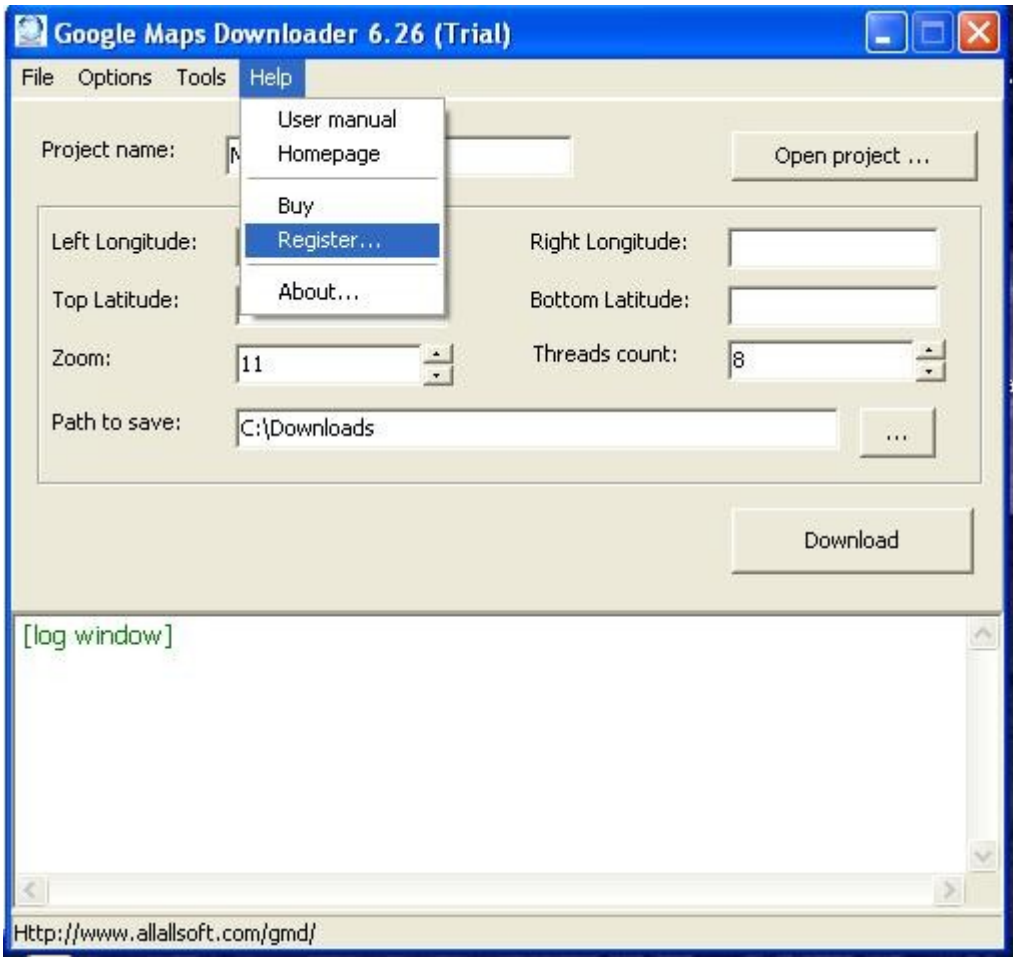
Como vemos esta compilado en C++

Así que llegamos a la conclusión de que no nos va a dar mucho dolor de cabeza :)

Ejecutamos el programa para estudiar nuestro target en búsqueda de Strings y demás cosas de interés.

Como se muestra en la imagen de abajo, ya hay varias cosas a tener en cuenta, primero nos informa que está en modo (Trial) o de prueba. Y por el otro lado vemos que nos invita a registrar.

Así que vamos allí e ingresamos lo primero que se nos viene a la cabeza.



Bueno ahora si que tenemos más información para trabajar. Una ventana que nos indica que hemos ingresado un serial equivocado, con una interesante cadena de texto.

Así que abrimos nuestro ejecutable con Olly

File View Debug Plugins Options Window Help									
Paused									
L E M T W H C / K B R ... S									
Address		Hex	dump	Disassembly		Comment	Registers (FPU)		
00401504		<Mod	\$	EB 10		JMP SHORT 00401516	EAX	00000000	
00401506			66	DB 66		CHAR 'f'	ECX	0012FFB0	
00401507			62	DB 62		CHAR 'b'	EDX	7C91EB94 ntdll.KiFastSystemCallRet	
00401508			3A	DB 3A		CHAR ':'	EBX	7FFDE000	
00401509			43	DB 43		CHAR 'C'	ESP	0012FFC4	
0040150A			2B	DB 2B		CHAR '+'	EBP	0012FFF0	
0040150B			2B	DB 2B		CHAR '+'	ESI	FFFFFFFF	
0040150C			48	DB 48		CHAR 'H'	EDI	7C920738 ntdll.7C920738	
0040150D			4F	DB 4F		CHAR 'O'	EIP	00401504 gmd.<ModuleEntryPoint>	
0040150E			4F	DB 4F		CHAR 'O'	C 0	ES 0023 32bit 0<FFFFFFFF>	
0040150F			4B	DB 4B		CHAR 'K'	P 1	CS 001D 32bit 0<FFFFFFFF>	
00401510			90	NOP			A 0	SS 0023 32bit 0<FFFFFFFF>	
00401511			E9	DB E9			Z 1	DS 0023 32bit 0<FFFFFFFF>	
00401512			98	CWDE			S 0	FS 003B 32bit 7FFDD000<FFF>	
00401513			D04A 00	ROR BYTE PTR DS:[EDX]1			T 0	GS 0000 NULL	
00401514		>	A1 8BD04A00	MOV EAX,DWORD PTR DS:[4AD00B1			D 0		
00401515			C1E0 02	SHL EAX,2			O 0	LastErr ERROR_MOD_NOT_FOUND <0000007E>	
00401516			A3 8FD04A00	MOV DWORD PTR DS:[4AD08F],EAX			EFL	00000246 <NO,NB,E,BE,NS,PE,GE,LE>	
00401517			52	PUSH EDX		[pModule = NULL	ST0	empty -UNORM BB4C 01050104 002E0032	
00401518			5A	PUSH 0		GetModuleHandlea	ST1	empty +UNORM 0062 00750070 002E0067	
00401519			EB BDB10A00	CALL <JMP.&KERNEL32.GetModuleHan			ST2	empty +UNORM 0079 006C006C 006F005C	
0040151A			8BD0	MOV EDX,EAX			ST3	empty +UNORM 0079 006C006C 006F005C	
0040151B			E8 46FB0900	CALL 004A1078			ST4	empty 0.1077351297874321920e-4933	
0040151C			5A	POP EDX			ST5	empty 0.0	
0040151D			E8 A4FA0900	CALL 004A0FDC			ST6	empty 1.000000000000000000000000	
0040151E			6A 00	PUSH 0		[Arg1 = 00000000	ST7	empty 1.000000000000000000000000	
0040151F			E8 800F0A00	CALL 004A24C4		gmd.004A24C4		3 2 1 0 E S P U O Z D I	
00401520			59	POP ECX			FST	4000 Cond 1 0 0 0 Err 0 0 0 0 0 0 <EQ>	
00401521			68 34D04A00	PUSH 004AD034			FCW	027F Prec NEAR,53 Mask 1 1 1 1 1 1	
00401522			6A 00	PUSH 0		[pModule = NULL			
00401523			E8 97B10A00	CALL <JMP.&KERNEL32.GetModuleHan		GetModuleHandlea			
00401524			A3 93D04A00	MOV DWORD PTR DS:[4AD093],EAX					
00401525			6A 00	PUSH 0					
00401526			E9 07660A00	JMP 004A7C04					
00401527		Ge	E9 AE0F0A00	JMP 004A2510					
00401528			33C0	XOR EAX,EAX					
00401529			A0 7DD04A00	MOV AL,BYTE PTR DS:[4AD07D]					
0040152A			C3	RET					
0040152B			A1 93D04A00	MOV EAX,DWORD PTR DS:[4AD093]					
0040152C			C3	RET					
0040152D			60	PUSHAD					
0040152E			BB 0050AB00	MOV EBX,BCR05000					
0040152F			00401516=00401516						
Address		Value		Comment					
004AD000		6C726F42				0012FFC4	7C816D4F	RETURN to kernel32.7C816D4F	
						0012FFB0	7C920738	ntdll.7C920738	
						00000000	FFFFFFFF		

Address	Hex	dump	Disassembly	Comment
0040B1EC	-	FF56 04	CALL DWORD PTR DS:[ESI+4]	
0040B1EF	-	FF4D 94	DEC DWORD PTR SS:[EBP-6C]	
0040B1F2	-	8D45 AC	LEA EAX,DWORD PTR SS:[EBP-54]	
0040B1F5	-	BA 02000000	MOV EDX,2	
0040B1FA	-	E8 F10D0A00	CALL 004ABFF0	
0040B1FF	-	FF4D 94	DEC DWORD PTR SS:[EBP-6C]	
0040B202	-	8D45 B0	LEA EAX,DWORD PTR SS:[EBP-50]	
0040B205	-	BA 02000000	MOV EDX,2	
0040B20A	-	E8 E10D0A00	CALL 004ABFF0	
0040B20F	-	8BF3	MOV ESI,EBX	
0040B211	-	8975 A4	MOV DWORD PTR SS:[EBP-5C],ESI	
0040B214	-	85F6	TEST ESI,ESI	
0040B216	-	74 1E	JE SHORT 0040B236	
0040B218	-	8B06	MOV EAX,DWORD PTR DS:[ESI]	
0040B21A	-	8945 A8	MOV DWORD PTR SS:[EBP-58],EAX	
0040B21D	-	66:C745 88 8C	MOV WORD PTR SS:[EBP-78],8C	
0040B223	-	BA 03000000	MOV EDX,3	
0040B228	-	8B45 A4	MOV EAX,DWORD PTR SS:[EBP-5C]	
0040B22B	-	8B08	MOV ECX,DWORD PTR DS:[EAX]	
0040B22D	-	FF51 FC	CALL DWORD PTR DS:[ECX-4]	
0040B230	-	66:C745 88 80	MOV WORD PTR SS:[EBP-78],80	
0040B236	>	66:C745 88 98	MOV WORD PTR SS:[EBP-78],98	
0040B23C	-	BA 5D0C4B00	MOV EDX,004B0C5D	ASCII "Congratulations! Register Success! ",LF,"Please restart!"
0040B241	-	8D45 A0	LEA EAX,DWORD PTR SS:[EBP-60]	
0040B244	-	E8 970D0A00	CALL 004ABDE0	
0040B249	-	FF45 94	INC DWORD PTR SS:[EBP-6C]	
0040B24C	-	8B08	MOV EAX,DWORD PTR DS:[EAX]	
0040B24E	-	E8 85CE0600	CALL 004780D8	
0040B253	-	FF4D 94	DEC DWORD PTR SS:[EBP-6C]	
0040B256	-	8D45 A0	LEA EAX,DWORD PTR SS:[EBP-60]	
0040B259	-	BA 02000000	MOV EDX,2	
0040B25E	-	E8 8D0D0A00	CALL 004ABFF0	
0040B263	-	8B05 74FFFFFF	MOV EAX,DWORD PTR SS:[EBP-8C]	
0040B269	-	E8 8E310600	CALL 0046E3FC	
0040B26E	-	EB 2D	JMP SHORT 0040B29D	
0040B270	>	66:C745 88 A4	MOV WORD PTR SS:[EBP-78],0A4	
0040B276	-	BA 910C4B00	MOV EDX,004B0C91	ASCII "Wrong serial number!"
0040B27B	-	8D45 9C	LEA EAX,DWORD PTR SS:[EBP-64]	
0040B27E	-	E8 5D0B0A00	CALL 004ABDE0	

Esta es la zona donde se prepara la cadena de texto que nos será mostrada.

Como vemos en la imagen se prepara “Wrong serial number!”, pero al mismo se accede a través de un salto que viene desde arriba como indica la línea azul.

Analizamos un poco de donde viene y seguimos la línea azul hasta donde se inicia.

Address	Hex	dump	Disassembly	Comment
0040B18F	-	E8 5C0E0A00	CALL 004ABFF0	Lgmd.004ABFF0
0040B194	-	59	POP ECX	
0040B195	-	84C9	TEST CL,CL	
0040B197	-	0F84 D3000000	JE 0040B270	
0040B19D	-	A1 64804B00	MOV EAX,DWORD PTR DS:[4B8064]	
0040B1A2	-	8B10	MOV EDX,DWORD PTR DS:[EAX]	
0040B1A4	-	8B8A 04040000	MOV ECX,DWORD PTR DS:[EDX+404]	
0040B1AA	-	B2 01	MOV DL,1	
0040B1AC	-	A1 E4024300	MOV EAX,DWORD PTR DS:[4302E4]	
0040B1B1	-	E8 1EC6FFFF	CALL 004077D4	
0040B1B6	-	8BD8	MOV EBX,EAX	
0040B1B8	-	8B45 FC	MOV EAX,DWORD PTR SS:[EBP-4]	
0040B1BB	-	50	PUSH EAX	
0040B1BC	-	BA 5A0C4B00	MOV EDX,004B0C5A	ASCII "Sn"
0040B1C1	-	8D45 AC	LEA EAX,DWORD PTR SS:[EBP-54]	
0040B1C4	-	E8 170C0A00	CALL 004ABDE0	
0040B1C9	-	FF45 94	INC DWORD PTR SS:[EBP-6C]	
0040B1CC	-	8B10	MOV EDX,DWORD PTR DS:[EAX]	
0040B1CE	-	8D45 B0	LEA EAX,DWORD PTR SS:[EBP-50]	
0040B1D1	-	52	PUSH EDX	
0040B1D2	-	BA 510C4B00	MOV EDX,004B0C51	ASCII "IniParas"
0040B1D7	-	66:C745 88 74	MOV WORD PTR SS:[EBP-78],74	
0040B1DD	-	E8 FE0B0A00	CALL 004ABDE0	
0040B1E2	-	FF45 94	INC DWORD PTR SS:[EBP-6C]	
0040B1E5	-	8B10	MOV EDX,DWORD PTR DS:[EAX]	
0040B1E7	-	8BC3	MOV EAX,EBX	
0040B1E9	-	59	POP ECX	
0040B1EA	-	8B30	MOV ESI,DWORD PTR DS:[EAX]	
0040B1EC	-	FF56 04	CALL DWORD PTR DS:[ESI+4]	
0040B1EF	-	FF4D 94	DEC DWORD PTR SS:[EBP-6C]	
0040B1F2	-	8D45 AC	LEA EAX,DWORD PTR SS:[EBP-54]	
0040B1F5	-	BA 02000000	MOV EDX,2	
0040B1FA	-	E8 F10D0A00	CALL 004ABFF0	
0040B1FF	-	FF4D 94	DEC DWORD PTR SS:[EBP-6C]	
0040B202	-	8D45 B0	LEA EAX,DWORD PTR SS:[EBP-50]	
0040B205	-	BA 02000000	MOV EDX,2	
0040B20A	-	E8 E10D0A00	CALL 004ABFF0	
0040B20F	-	8BF3	MOV ESI,EBX	
0040B211	-	8975 A4	MOV DWORD PTR SS:[EBP-5C],ESI	
0040B214	-	85F6	TEST ESI,ESI	
0040B216	-	74 1E	JE SHORT 0040B236	
0040B270=0040B270				

Ponemos un breakpoint en el salto condicional que nos manda a la zona de serial equivocado y reiniciamos nuestro programa en el Olly.

Una vez reiniciado, lo ejecutamos con F9 y volvemos a ingresar el serial que habíamos intentado anteriormente, pero esta vez se detiene en nuestro bp.

NOTA:

Aquí quiero hacer un comentario, vemos que lo que decide si saltar o no a la zona del serial equivocado, es por el resultado que posee CL. En mi caso decidí por una cuestión experimental, incluirle una sección al exe para poder parchear y no tener que andar buscando huecos en el programa. Así que primero nopee el POP ECX hasta el Je incluido. Es decir después del call lo mandé

con un `Jmp` a mi sección que en mi caso esta `04dc000` y escribí en mi sección las líneas que había nopeado moviendo un 1 a `CL`

En resumen quedó así:

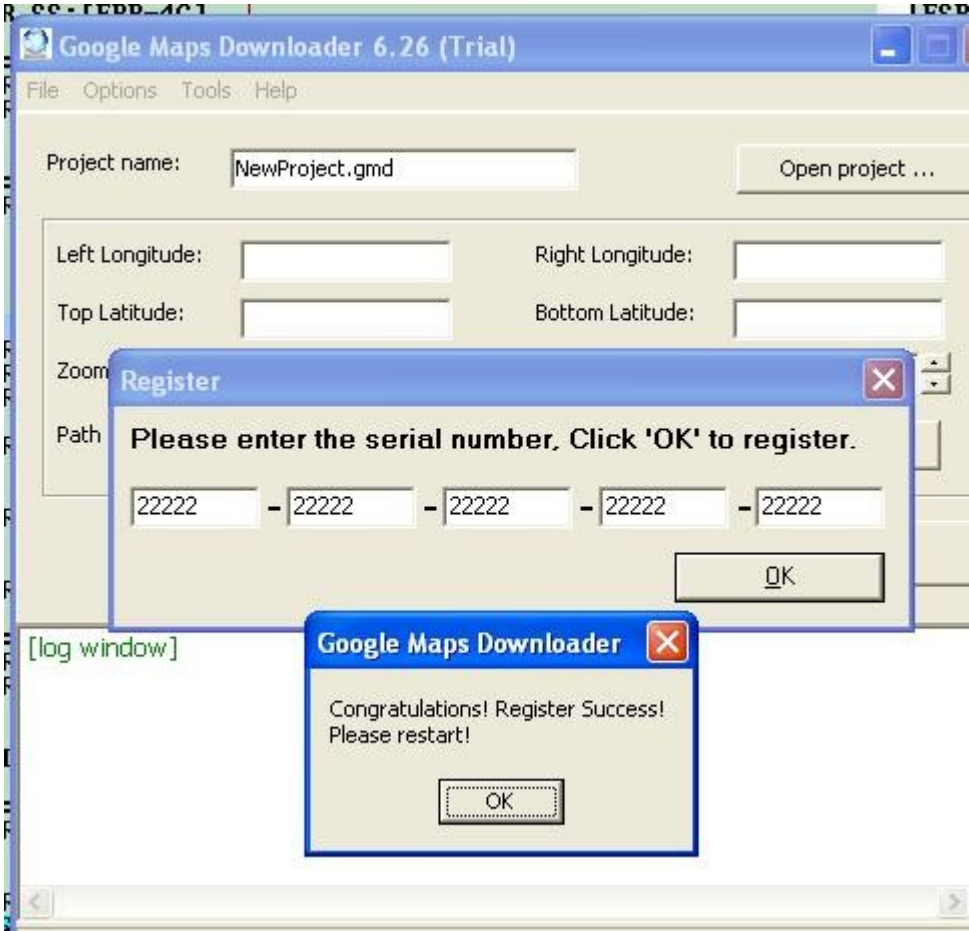
0040B17B	-	8D45 F4	LEA EAX,DWORD PTR SS:[EBP-4]
0040B17E	-	E8 510F0A00	CALL 004AC0D4
0040B183	-	50	PUSH EAX
0040B184	-	FF4D 94	DEC DWORD PTR SS:[EBP-6C]
0040B187	-	8D45 B4	LEA EAX,DWORD PTR SS:[EBP-4C]
0040B18A	-	BA 02000000	MOV EDX,2
0040B18F	-	E8 5C0E0A00	CALL 004ABFF0
0040B194	-	E9 670E0D00	JMP 004DC000
0040B199	>	90	NOP
0040B19A	-	90	NOP
0040B19B	-	90	NOP
0040B19C	-	90	NOP
0040B19D	-	A1 64804B00	MOV EAX,DWORD PTR DS:[4B8064]
0040B1A2	-	8B10	MOV EDX,DWORD PTR DS:[EAX]
0040B1A4	-	8B8A 04040000	MOV ECX,DWORD PTR DS:[EDX+404]
0040B1AA	-	B2 01	MOV DL,1
0040B1AC	-	A1 E4024300	MOV EAX,DWORD PTR DS:[4302E4]
0040B1B1	-	E8 1EC6FFFF	CALL 004077D4

Mi Sección que llamé `.GUI`

Address	Hex dump	Disassembly
004DC000	> 59	POP ECX
004DC001	- B1 01	MOV CL,1
004DC003	- 84C9	TEST CL,CL
004DC005	- 0F84 65F2F2FF	JE 0040B270
004DC00B	- E9 89F1F2FF	JMP 0040B199
004DC010	90	NOP
004DC011	90	NOP
004DC012	90	NOP

Se puede apreciar que en `4dc001` muevo un 1 a `CL` antes de mandarlo con el `jmp`.
Quizás este código puede quedar más simplificado.. pero a mí se me ocurrió hacerlo así.
Fin Nota.

Pero todo esto se puede evitar simplemente forzando flag del Olly con doble clic y evitamos el salto ☺



Bueno, elegimos cualquiera de los métodos que mencioné y (es más cómodo cambiar el flag!) Felicitaciones Registrado satisfactoriamente y nos invita a reiniciar!

Tambien si rastreamos un poco encontramos que va a escribir `config.ini` en la ruta de nuestro programa

Debugger window showing assembly code and registers. The assembly code includes instructions like CALL, MOV, PUSH, and POP, with comments indicating the use of registers like EAX, ECX, EDX, ESP, EBP, ESI, and EDI. The registers window shows the state of various registers, including EAX, ECX, EDX, ESP, EBP, ESI, and EDI, with values and flags.

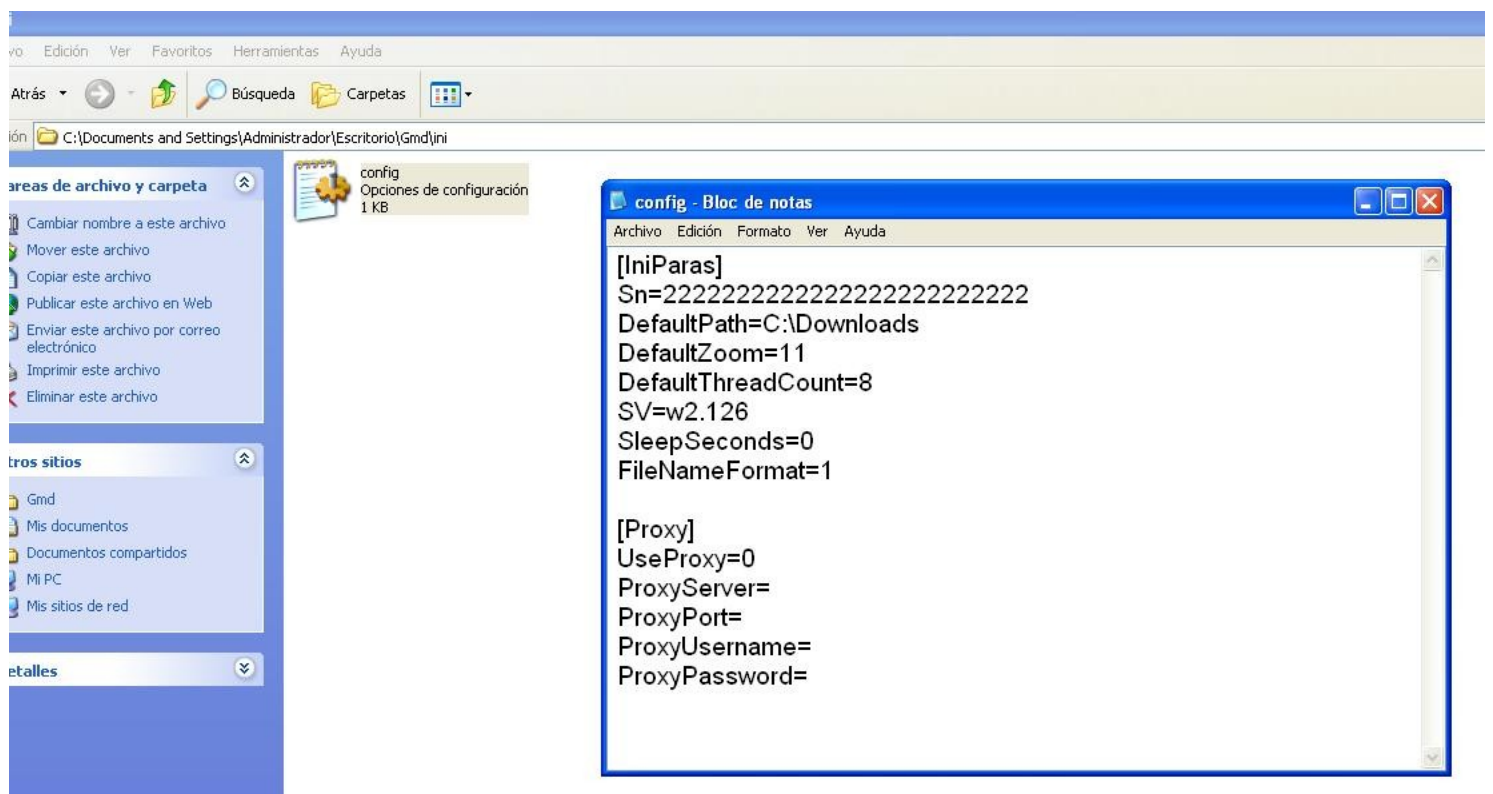
Pero si seguimos rastreando los call con F7 vemos que sigue trabajando con nuestro serial y si ponemos un bp en la api que escribe en los .ini vemos que para

Debugger window showing assembly code and registers. The assembly code includes instructions like MOV, PUSH, and POP, with comments indicating the use of registers like EAX, ECX, EDX, ESP, EBP, ESI, and EDI. The registers window shows the state of various registers, including EAX, ECX, EDX, ESP, EBP, ESI, and EDI, with values and flags.

Command: bp WritePrivateProfileStringA

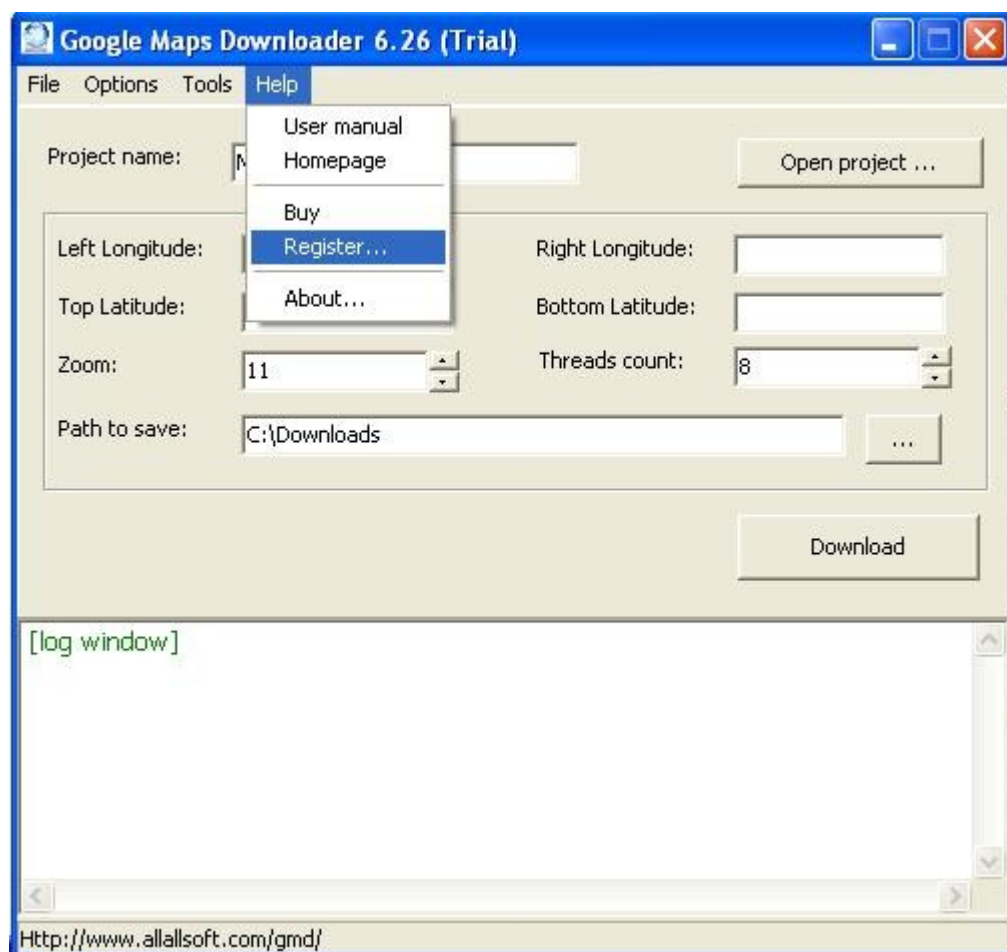
Breakpoint at kernel32.WritePrivateProfileStringA

Vamos a la carpeta del programa y buscamos algún archivo ini y nos encontramos con este:



Vemos que en Sn esta nuestro serial guardado.

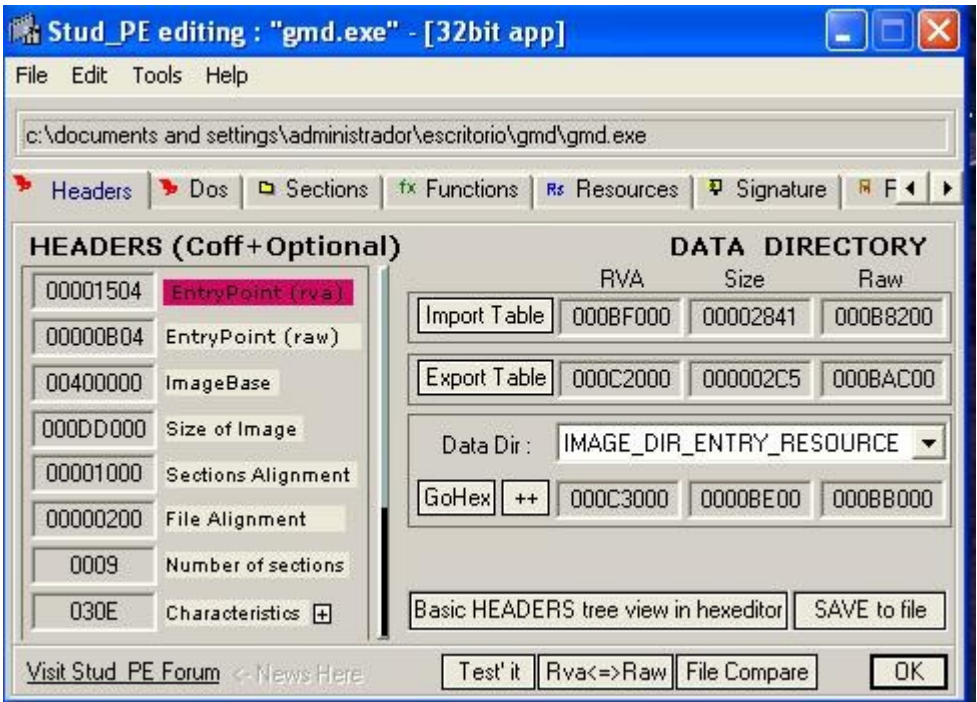
Reiniciamos contenidos y vemos esto:



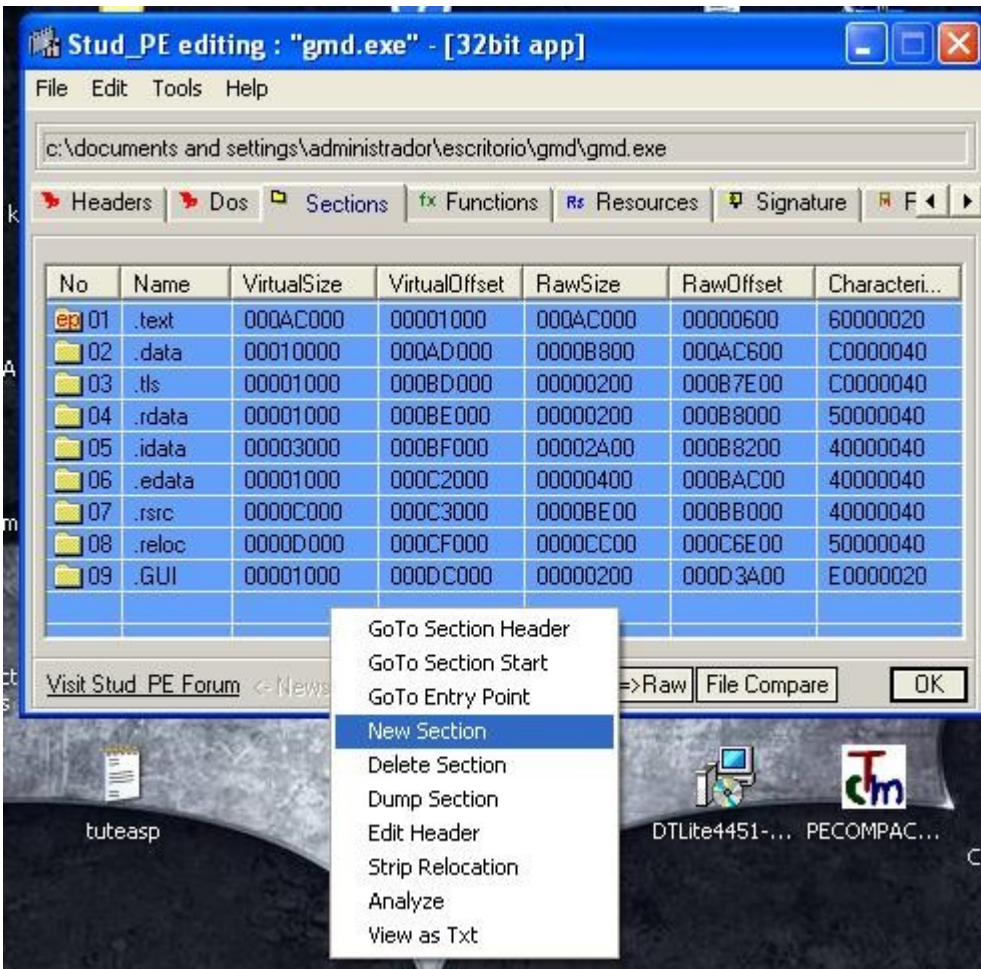
Uhh.. volvemos a cero, nuestro trabajo no fue efectivo??.. La respuesta es: Más o menos, estamos a medio camino, es decir lo que hemos hecho es forzado a que escriba en el archivo .ini del programa nuestro serial que obviamente es chequeado al reiniciar el programa y no nos registra.

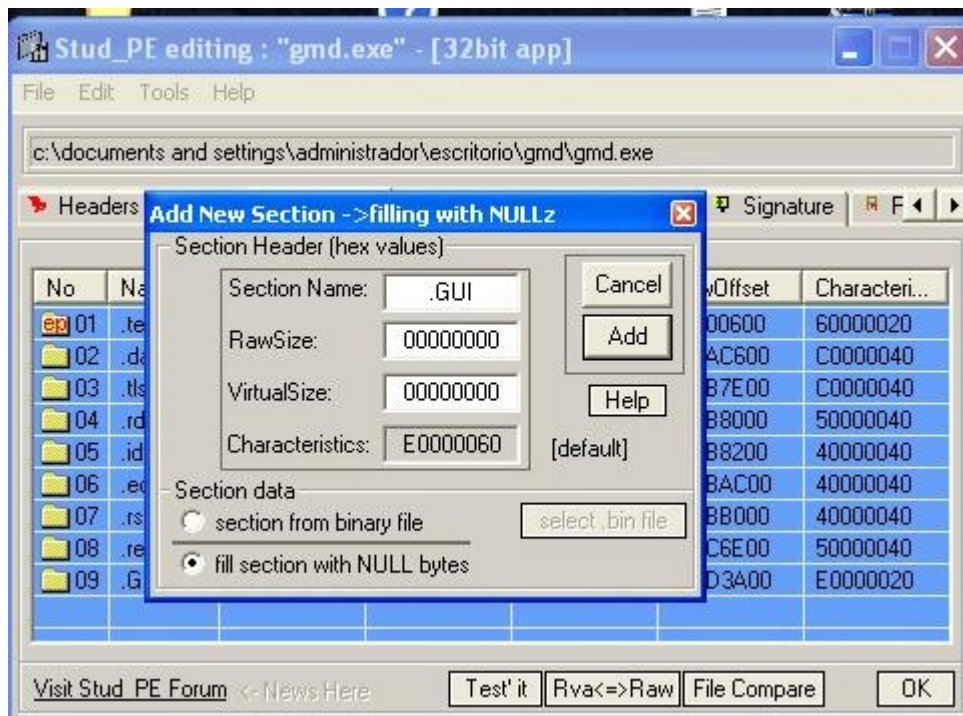
Llegué a la conclusión de que hay que parchear para arreglar este lío! y como dije antes agregar una sección para no tener que andar buscando huecos y que a veces se sobrescribe con información no prevista.

Usamos Stud PE, así que lo abrimos y cargamos nuestro gmd.exe



Clic derecho y elegimos crear nueva sección



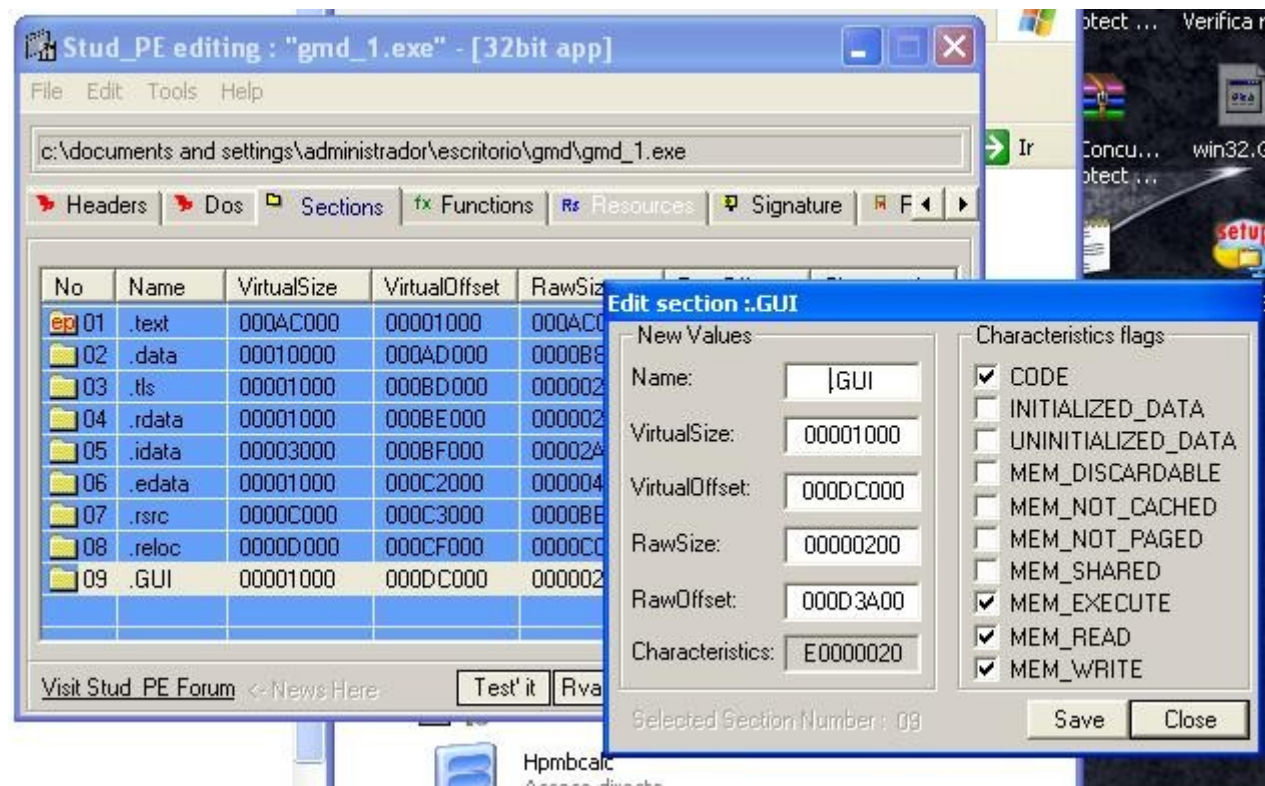


Elegimos el nombre que queremos y luego add, y ok

Después vamos a la pestaña sections, buscamos la nuestra (.GUI) y clic derecho – Edit header

Y le asignamos los bytes que deseemos en mi caso le puse 100 bytes tanto en rawsize como en virtual (virtual y disco) y cliqueamos en donde dice que llene la sección con null bytes.

Nota: 100 bytes son muchos... puede ser menos y el ejecutable final pesará menos.. yo le puse 100 por costumbre... para no errarle y como no se cuanto tengo que parchear.. je je!, pero si quieren pongan menos 40 – 50 pero ojo que no les falte.



Y aquí lo que hacemos es editar en Characteristics, y cambiar E0000060 por E0000020 como se ve en la imagen. Es decir le damos permisos de lectura y escritura. Luego en Save y Ok

Listo tenemos nuestra sección creada, para ven la dirección de la misma vamos en el Olly vamos a la ventana de memoria y la buscamos:

M File View Debug Plugins Options Window Help									
Running									
L E M T W H C / K B R ... S									
AddressSizeOwnerSectionContainsTypeAccessInitialMapped as									
00340000	00006000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\sortthls.nls	
00350000	00041000				Map	R	R		
003A0000	00001000				Priv	RWE	RWE		
003B0000	00001000				Priv	RWE	RWE		
003C0000	00007000				Priv	RW	RW		
003D0000	00003000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\ctype.nls	
003E0000	00001000				Priv	RW	RW		
003F0000	00001000				Priv	RW	RW		
00400000	00001000	gmd		PE header	Imag	R	RWE		
00401000	0000A000	gmd	.text	code	Imag	R	RWE		
004AD000	00010000	gmd	.data	code,data	Imag	R	RWE		
004BD000	00001000	gmd	.tls	code	Imag	R	RWE		
004BE000	00001000	gmd	.rdata	code	Imag	R	RWE		
004BF000	00003000	gmd	.idata	code,imports	Imag	R	RWE		
004C2000	00001000	gmd	.edata	code,exports	Imag	R	RWE		
004C3000	0000C000	gmd	.rsrc	code,resources	Imag	R	RWE		
004CF000	0000D000	gmd	.reloc	code,relocations	Imag	R	RWE		
004DC000	00001000	gmd	.GUI	code	Imag	R	RWE		
004E0000	00041000				Map	R	R		
00530000	00005000				Map	R E	R E		
005F0000	00002000				Map	R E	R E		
00600000	00103000				Map	R	R		
00710000	000B9000				Map	R E	R E		
00A10000	00004000				Priv	RW	RW		
00A20000	00002000				Map	R	R		
00A30000	00002000				Map	R	R		
00A40000	00004000				Priv	RW	RW		
00A50000	00002000				Map	R	R		
00A60000	00001000				Priv	RWE	RWE		
00A70000	00001000				Priv	RWE	RWE		
00A80000	00001000				Priv	RWE	RWE		
00A90000	00001000				Map	RW	RW		
00AA0000	00003000				Map	RW	RW		
00AC0000	00002000				Map	R	R		
00AD0000	00050000				Map	R	R		
00B20000	00010000				Map	RW	RW		
00D00000	0001C000				Priv	RW	RW		
00CD0000	00001000				Priv	RW	RW		
00D50000	00007000				Map	RW	RW		
58C30000	00001000	COMCTL32		PE header	Imag	R	RWE		
58C31000	00070000	COMCTL32	.text	code,imports,exports	Imag	R	RWE		

Ahí se ve en 4DC000 empieza.

Con la sección creada ahora nos dedicamos a parcher. Quiero aclara que hay muchos métodos para llegar a encontrar la comprobación de si estamos registrados o no, luego de probar buscar por Apis de Windows he encontrado que la manera más rápida es buscar la cadena de texto (Trial) que es la que nos pone en la barra de titulo de la ventana principal del programa

Es evidente que si estamos registrados no tendría que mostrarla.. así que la buscamos entre las strings en el Olly.

Clic derecho en la ventana del desensamblado y la buscamos como hicimos antes con Wrong serial number... pero ahora buscamos (Trial).

Y vemos esto:

- [text strings referenced in crack2:.text]		
R File View Debug Plugins Options Window Help		
Paused		
L E M T W H C / K B R ... S		
AddressDisassemblyText string		
00407E40	MOV EDX,004ADA5C	ASCII "c:\gmd"
00407E5C	MOV EDX,004ADA50	ASCII "DefaultPath"
00407E6F	MOV EDX,004ADA47	ASCII "IniParas"
00407EE3	MOV EDX,004ADA7E	ASCII "mymap.bmp"
00407EFF	MOV EDX,004ADA6C	ASCII "DefaultOutputName"
00407F12	MOV EDX,004ADA63	ASCII "IniParas"
00407F82	MOV EDX,004ADA91	ASCII "DefaultZoom"
00407F92	MOV EDX,004ADA88	ASCII "IniParas"
00407FE1	MOV EDX,004ADAF6	ASCII "PauseInterval"
00407FF1	MOV EDX,004ADA9D	ASCII "IniParas"
00408040	MOV EDX,004ADABD	ASCII "DefaultThreadCount"
00408050	MOV EDX,004ADAB4	ASCII "IniParas"
004080A3	MOV EDX,004ADADC	ASCII "w2.126"
004080C8	MOV EDX,004ADAD9	ASCII "SU"
004080DE	MOV EDX,004ADAD0	ASCII "IniParas"
0040815E	MOV EDX,004ADAF4	ASCII "6.26"
0040817D	MOV EDX,004ADAE3	ASCII "Version"
00408193	MOV EDX,004ADAE3	ASCII "IniParas"
0040820F	MOV EAX,004ADAF9	ASCII "Google Maps Downloader "
00408240	MOV EDX,004ADB17	ASCII "UseProxy"
00408253	MOV EDX,004ADB11	ASCII "Proxy"
004082DD	MOV EDX,004ADB26	ASCII "ProxyServer"
004082F3	MOV EDX,004ADB20	ASCII "Proxy"
0040839E	MOV EDX,004ADB39	ASCII "ProxyPort"
004083B4	MOV EDX,004ADB33	ASCII "Proxy"
0040845F	MOV EDX,004ADB4A	ASCII "ProxyUsername"
00408475	MOV EDX,004ADB44	ASCII "Proxy"
00408520	MOV EDX,004ADB5F	ASCII "ProxyPassword"
00408536	MOV EDX,004ADB59	ASCII "Proxy"
004087B1	MOV EDX,004ADB72	ASCII "rh"
004088BB	MOV EDX,004ADB76	ASCII "D15FADB3A3B1019C"
004089CF	MOV EDX,004ADB87	ASCII " (Trial)"
00408B9C	MOV EDX,004ADB90	ASCII "THREADS"
00408C9A	MOV EDX,004ADB9C	ASCII "THREADS"
00408E24	MOV EDX,004ADBA9	ASCII "-----"
00408E68	MOV EDX,004ADBCA	ASCII "Download Finished."
00408F2E	MOV EAX,004ADBDE	ASCII "Now time is:"
00408FD6	MOV EAX,004ADBEC	ASCII "Log file has been saved to "
0040915B	MOV EDX,004ADC08	ASCII "THREADS"
00409271	MOV EDX,004ADC14	ASCII "THREADS"
00409395	MOV EDX,004ADC20	ASCII "Download"
004093D7	MOV EDX,004ADC29	ASCII "Download finished!"
00409444	ASCII "TDownThread *[2]"	
00409454	ASCII 0	
004094C7	MOV EDX,004ADC41	ASCII "mapviewer.exe"
004094FC	PUSH 004ADC3C	ASCII "open"
004095ED	PUSH 004ADC50	ASCII "open"
00409615	MOV EDX,004ADC56	ASCII "No logfile now. "

Le hacemos doble clic y llegamos

File View Debug Plugins Options Window Help			
Paused			
L E M T W H C / K B R ... S			
Address	Hex dump	Disassembly	Comment
0040894F	> 74 2C	JE SHORT 0040897D	
00408951	. 33D2	XOR EDX,EDX	
00408953	. 8B86 20030000	MOV EAX,DWORD PTR DS:[ESI+320]	
00408959	. E8 06F70400	CALL 00458064	
0040895E	. 33D2	XOR EDX,EDX	
00408960	. 8B86 24030000	MOV EAX,DWORD PTR DS:[ESI+324]	
00408966	. E8 F9F60400	CALL 00458064	
0040896B	. 33D2	XOR EDX,EDX	
0040896D	. 8B86 1C030000	MOV EAX,DWORD PTR DS:[ESI+31C]	
00408973	. E8 ECF60400	CALL 00458064	
00408978	> E9 CA000000	JMP 00408A47	
0040897D	> B2 01	MOV DL,1	
0040897F	. 8B86 20030000	MOV EAX,DWORD PTR DS:[ESI+320]	
00408985	. E8 DAF60400	CALL 00458064	
0040898A	. B2 01	MOV DL,1	
0040898C	. 8B86 24030000	MOV EAX,DWORD PTR DS:[ESI+324]	
00408992	. E8 CDF60400	CALL 00458064	
00408997	. B2 01	MOV DL,1	
00408999	. 8B86 1C030000	MOV EAX,DWORD PTR DS:[ESI+31C]	
0040899F	. E8 C0F60400	CALL 00458064	
004089A4	. 66:C743 10 A0	MOV WORD PTR DS:[EBX+10],1A0	
004089AA	. 33C9	XOR ECX,ECX	
004089AC	. 898D E8FEFFFF	MOV DWORD PTR SS:[EBP-118],ECX	
004089B2	. 8D95 E8FEFFFF	LEA EDX,DWORD PTR SS:[EBP-118]	
004089B8	. FF43 1C	INC DWORD PTR DS:[EBX+1C]	
004089BB	. 8B35 AC864B00	MOV ESI,DWORD PTR DS:[_Form1]	
004089C1	. 8BC6	MOV EAX,ESI	
004089C3	. E8 9C4F0700	CALL 0047D964	
004089C8	. 8D95 E8FEFFFF	LEA EDX,DWORD PTR SS:[EBP-118]	
004089CE	. 52	PUSH EDX	
004089CF	. BA 87DB4A00	MOV EDX,004ADB87	ASCII " <Trial>"
004089D4	. 8D85 E4FEFFFF	LEA EAX,DWORD PTR SS:[EBP-11C]	
004089DA	. E8 01340A00	CALL 004ABDE0	
004089DF	. FF43 1C	INC DWORD PTR DS:[EBX+1C]	
004089E2	. 33C0	XOR EAX,EAX	
004089E4	. 8985 E0FEFFFF	MOV DWORD PTR SS:[EBP-120],EAX	
004089EA	. 8D95 E4FEFFFF	LEA EDX,DWORD PTR SS:[EBP-11C]	
004089F0	. FF43 1C	INC DWORD PTR DS:[EBX+1C]	
004089F3	. 8D8D E0FEFFFF	LEA ECX,DWORD PTR SS:[EBP-120]	
004089F9	. 58	POP EAX	
004089FA	. E8 49360A00	CALL 004AC048	
00408A47=00408A47			

Si subimos un poco vemos como se muestra en la imagen un salto bastante largo que se saltea el trial y un poco más arriba un salto condicional. Con lo cual lo que queremos es que ese salto condicional no salte, sino que siga hasta el jump.

Subimos un poco y lo vemos más claro:

- [CPU - main thread, module gmd_2]			
File View Debug Plugins Options Window Help			
Paused			
L E M T W H C / K B R ... S			
Address	Hex dump	Disassembly	Comment
0040890F	> 66:C743 10 58	MOV WORD PTR DS:[EBX+10],158	
00408915	. C686 39040000	MOV BYTE PTR DS:[ESI+439],0	
0040891C	> FF4B 1C	DEC DWORD PTR DS:[EBX+1C]	
0040891F	. 8D85 04FFFFFF	LEA EAX,DWORD PTR SS:[EBP-FC]	
00408925	. BA 02000000	MOV EDX,2	
0040892A	. E8 C1360A00	CALL 004ABFF0	
0040892F	. FF4B 1C	DEC DWORD PTR DS:[EBX+1C]	
00408932	. 8D85 08FFFFFF	LEA EAX,DWORD PTR SS:[EBP-F8]	
00408938	. BA 02000000	MOV EDX,2	
0040893D	. E8 AE360A00	CALL 004ABFF0	
00408942	. 66:C743 10 14	MOV WORD PTR DS:[EBX+10],14	
00408948	> 80BE 39040000	CMP BYTE PTR DS:[ESI+439],0	
0040894F	> 74 2C	JE SHORT 0040897D	
00408951	. 33D2	XOR EDX,EDX	
00408953	. 8B86 20030000	MOV EAX,DWORD PTR DS:[ESI+320]	
00408959	. E8 06F70400	CALL 00458064	
0040895E	. 33D2	XOR EDX,EDX	
00408960	. 8B86 24030000	MOV EAX,DWORD PTR DS:[ESI+324]	
00408966	. E8 F9F60400	CALL 00458064	
0040896B	. 33D2	XOR EDX,EDX	
0040896D	. 8B86 1C030000	MOV EAX,DWORD PTR DS:[ESI+31C]	
00408973	. E8 ECF60400	CALL 00458064	
00408978	> E9 CA000000	JMP 00408A47	
0040897D	> B2 01	MOV DL,1	
0040897F	. 8B86 20030000	MOV EAX,DWORD PTR DS:[ESI+320]	
00408985	. E8 DAF60400	CALL 00458064	
0040898A	. B2 01	MOV DL,1	
0040898C	. 8B86 24030000	MOV EAX,DWORD PTR DS:[ESI+324]	
00408992	. E8 CDF60400	CALL 00458064	
00408997	. B2 01	MOV DL,1	
00408999	. 8B86 1C030000	MOV EAX,DWORD PTR DS:[ESI+31C]	
0040899F	. E8 C0F60400	CALL 00458064	
004089A4	. 66:C743 10 A0	MOV WORD PTR DS:[EBX+10],1A0	
Jump from 0040868B			

Vemos que en 408948 compara el byte el puntero de esi + 439 con 0

Asi que como el nuestro da 0, necesitamos que valgo 1

Si lo seguimos en el dump confirmamos el valor:

Address

Hex dump

Disassembly

Comment

0040893D

- E8 AE360A00

CALL 004ABFF0

00408942

- 66:C743 10 14

MOV WORD PTR DS:[EBX+10],14

00408948

> 80BE 39040000

CMP BYTE PTR DS:[ESI+320],0

0040894F

- 74 2C

JE SHORT 0040897D

00408951

- 33D2

XOR EDX,EDX

00408953

- 8B86 20030000

MOV EAX,DWORD PTR DS:[ESI+320]

00408959

- E8 06F70400

CALL 00458064

0040895E

- 33D2

XOR EDX,EDX

00408960

- 8B86 24030000

MOV EAX,DWORD PTR DS:[ESI+324]

00408966

- E8 F9F60400

CALL 00458064

0040896B

- 33D2

XOR EDX,EDX

0040896D

- 8B86 1C030000

MOV EAX,DWORD PTR DS:[ESI+31C]

00408973

- E8 ECF60400

CALL 00458064

00408978

- E9 CA000000

JMP 00408A47

0040897D

> B2 01

MOV DL,1

0040897F

- 8B86 20030000

MOV EAX,DWORD PTR DS:[ESI+320]

00408985

- E8 DAF60400

CALL 00458064

0040898A

- B2 01

MOV DL,1

DS:[00BD2BC9]=00

Jump from 0040868B

Address

Value

Comment

00BD2BC9

00000000

00BD2BCD

00000000

00BD2BD1

00000000

00BD2BD5

00000000

00BD2BD9

00000000

00BD2BDD

00000000

00BD2BE1

00000000

00BD2BE5

00000000

00BD2BE9

00000000

00BD2BED

00000000

00BD2BF1

00000000

00BD2BF5

16000000

00BD2BF9

00000000

00BD2BFD

00000000

00BD2C01

00000000

00BD2C05

00000000

00BD2C09

2A000000

Backup

Copy

Binary

Assemble

Label

Comment

Breakpoint

Hit trace

Run trace

Go to

Follow in Dump

View call tree

Search for

Find references to

View

Copy to executable

Analysis

Detach Process

Process Patcher

Analyze This!

Asm2Clipboard

Bookmark

Code Ripper

Data Rinner

Selection

Memory address

Ahí lo subrayé, 000000000 ☹

Tomamos nota del mov y del cmp porque lo vamos a nopear para reescribirlo en nuestra sección.

Nopeamos y ponemos un salto a nuestra seccion:

CPU - main thread, module crackZ

File View Debug Plugins Options Window Help

Paused

L E M T W H C / K B R ... S

Address

Hex dump

Disassembly

Comment

00408932

- 8D85 08FFFFFF

LEA EAX,DWORD PTR SS:[EBP-F8]

00408938

- BA 02000000

MOV EDI,2

0040893D

- E8 AE360A00

CALL 004ABFF0

00408942

- E9 CE360D00

JMP 004DC015

00408947

- 90

NOP

00408948

> 90

NOP

00408949

- 90

NOP

0040894A

- 90

NOP

0040894B

- 90

NOP

0040894C

- 90

NOP

0040894D

- 90

NOP

0040894E

- 90

NOP

0040894F

> 74 2C

JE SHORT 0040897D

00408951

- 33D2

XOR EDI,EDI

00408953

- 8B86 20030000

MOV EAX,DWORD PTR DS:[ESI+320]

00408959

- E8 06F70400

CALL 00458064

0040895E

- 33D2

XOR EDI,EDI

00408960

- 8B86 24030000

MOV EAX,DWORD PTR DS:[ESI+324]

00408966

- E8 F9F60400

CALL 00458064

0040896B

- 33D2

XOR EDI,EDI

0040896D

- 8B86 1C030000

MOV EAX,DWORD PTR DS:[ESI+31C]

00408973

- E8 ECF60400

CALL 00458064

00408978

- E9 CA000000

JMP 00408A47

0040897D

> B2 01

MOV DL,1

0040897F

- 8B86 20030000

MOV EAX,DWORD PTR DS:[ESI+320]

00408985

- E8 DAF60400

CALL 00458064

0040898A

- B2 01

MOV DL,1

0040898C

- 8B86 24030000

MOV EAX,DWORD PTR DS:[ESI+324]

00408992

- E8 CDF60400

CALL 00458064

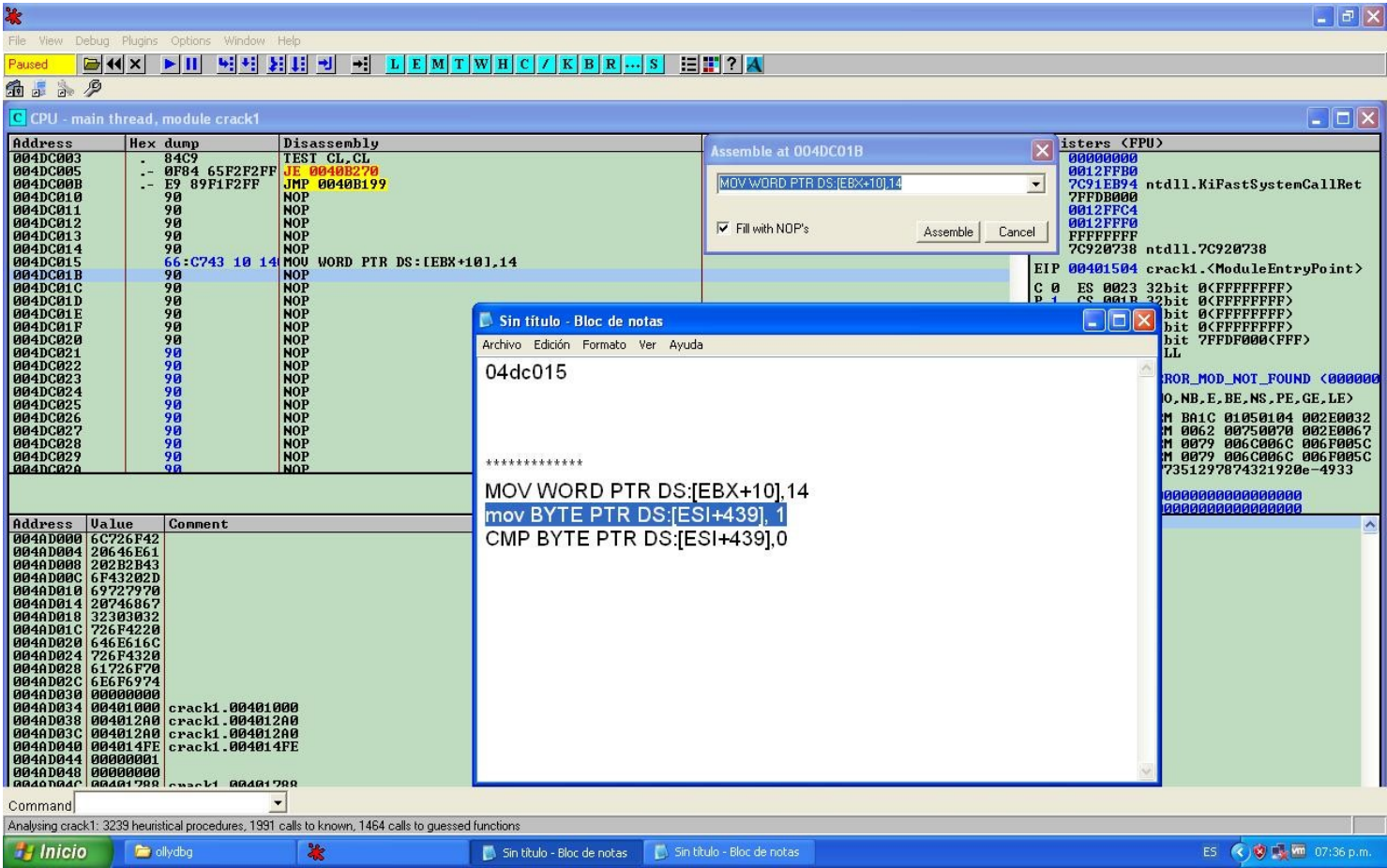
00408997

- B2 01

MOV DL,1

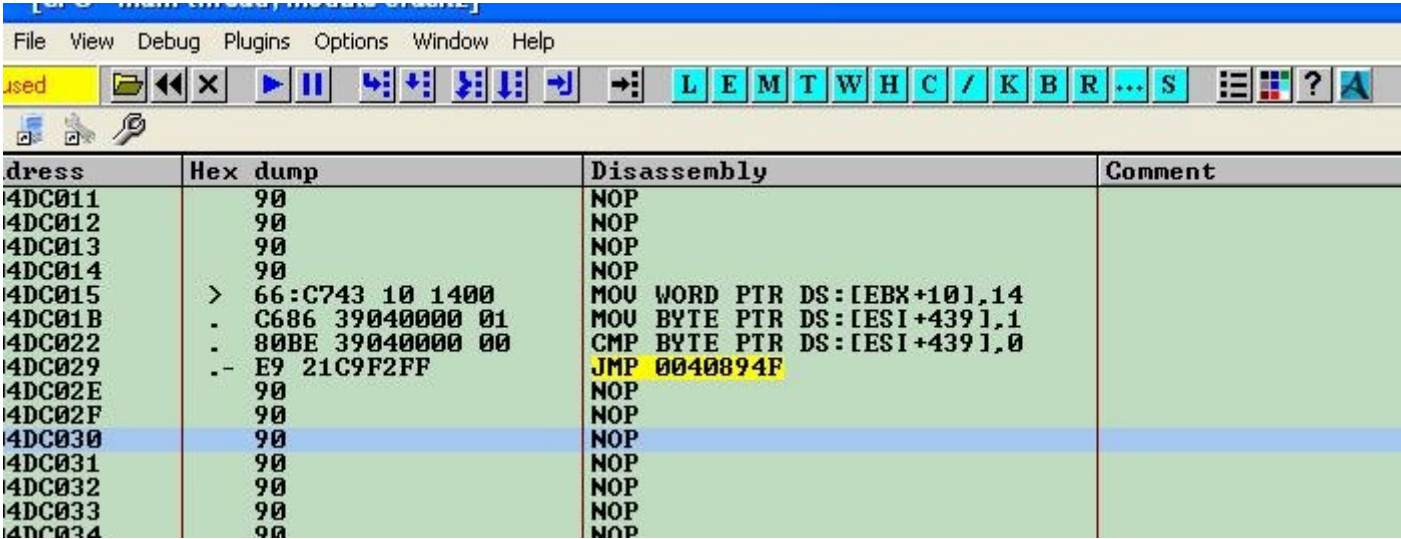
0040897D=0040897D

Asi que como habíamos tomado nota vamos a nuestra sección y empezamos a escribir el código:



Ahí en la captura resalté el mov esi+439, 1 que incluí .. como queremos que valga 1 y no 0 lo hacemos valer 1.

Finalmente nos quedaría así, con un salto final de regreso a la comparación:

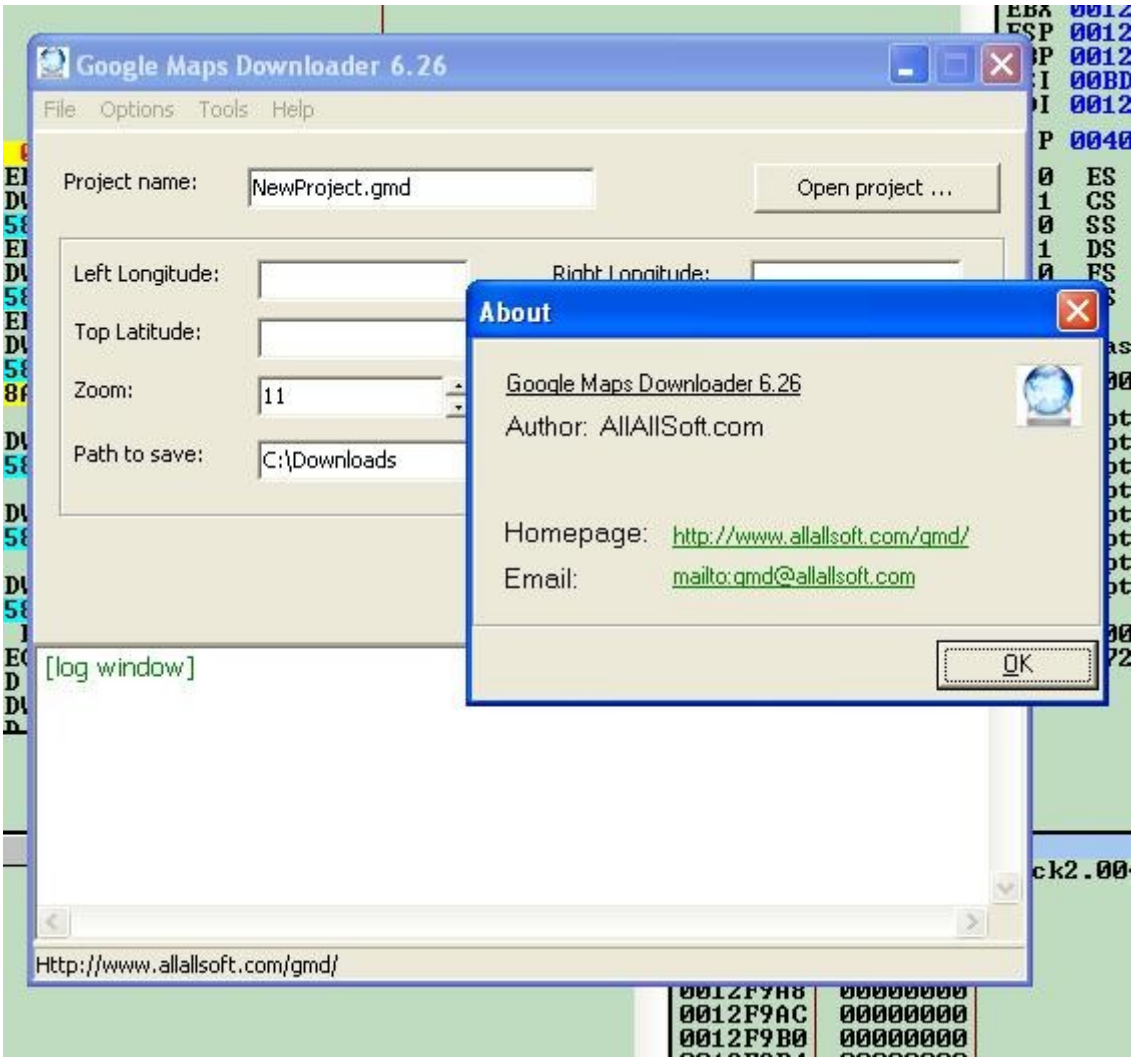


A continuación guardamos todos los cambios en save file—all modifications y lo guardamos con otro nombre

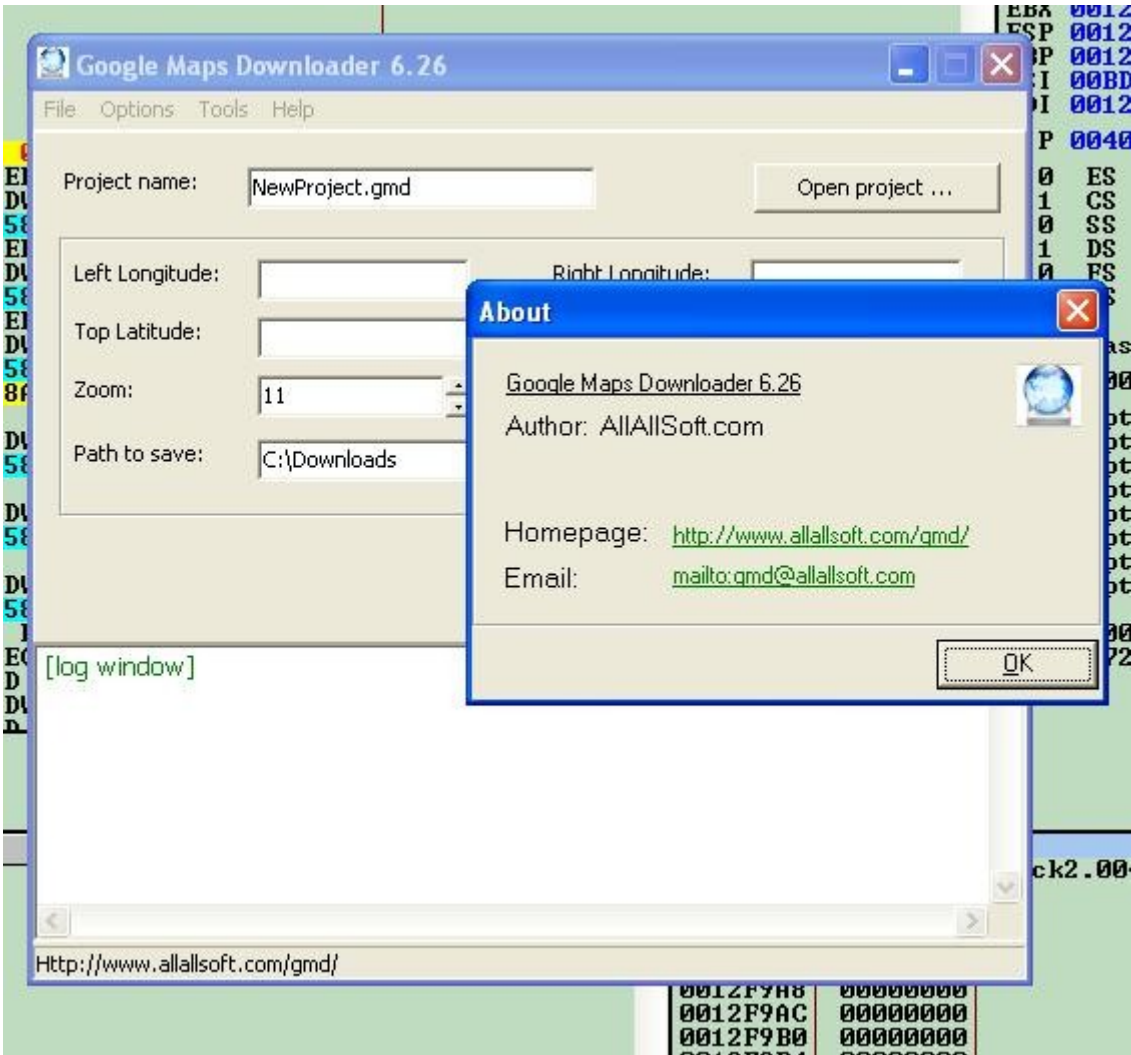
Bueno terminamos, ahora es hora de la prueba de fuego, esperemos no quemarnos!!

Cargamos nuestro trabajo y lo ejecutamos:

Bueno hay poco para decir, desapareció el trial y además en la carpeta help no nos permite registrarnos porque ya lo estamos!!



Si vamos a about vemos que no nos dice trial ni nada.



Conclusión :El programa quedó full, quizás haya una manera más fácil de registrarlo, opté por este camino porque quería explorar como crear nuevas secciones y probar con ellas.

Sepan disculpar errores tanto en la confección de este tute como de conocimientos ya que estoy aprendiendo con mucho entusiasmo todo esto.

Agradecimientos:

A todos CracksLatinos por ser un excelente grupo de trabajo y un lugar de amigos muy ameno!

A Ricardo Narvaja por darnos este espacio!. Gracias a él y a CracksLatinos he aprendido todo lo que sé

Un agradecimiento Especial a InDuLgEo por enseñarme todo lo que sabe y por la paciencia que me tiene siempre que lo consulto y además por haberme prestado la advertencia de este documento .

Y gracias a ti por leer este tute.

