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 descriptos, 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.

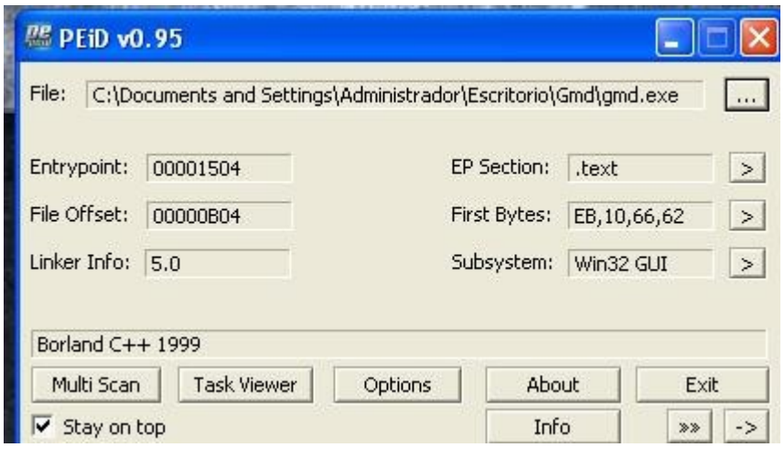
De esta manera, obtendremos un archivo de la parte del terreno que nos interesa guardar. Google Maps Images Downloader incluye un visualizador y un combinador de imágenes para fusionar varios archivos.

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



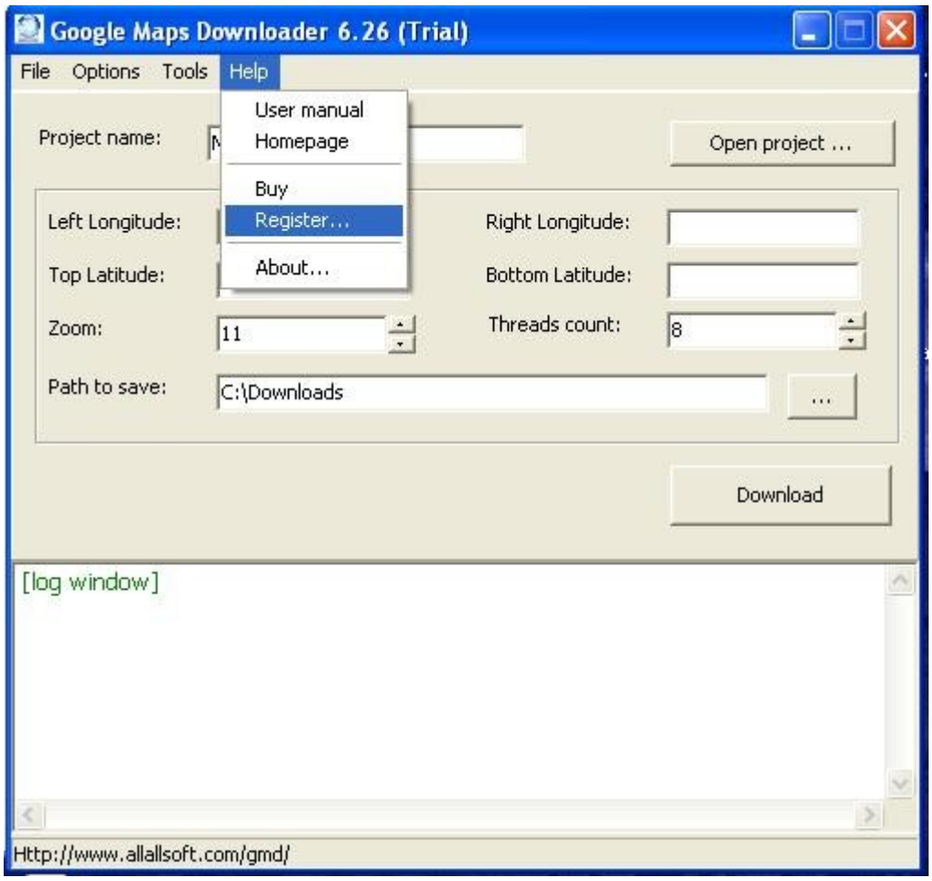
Como vemos esta compilado en C++

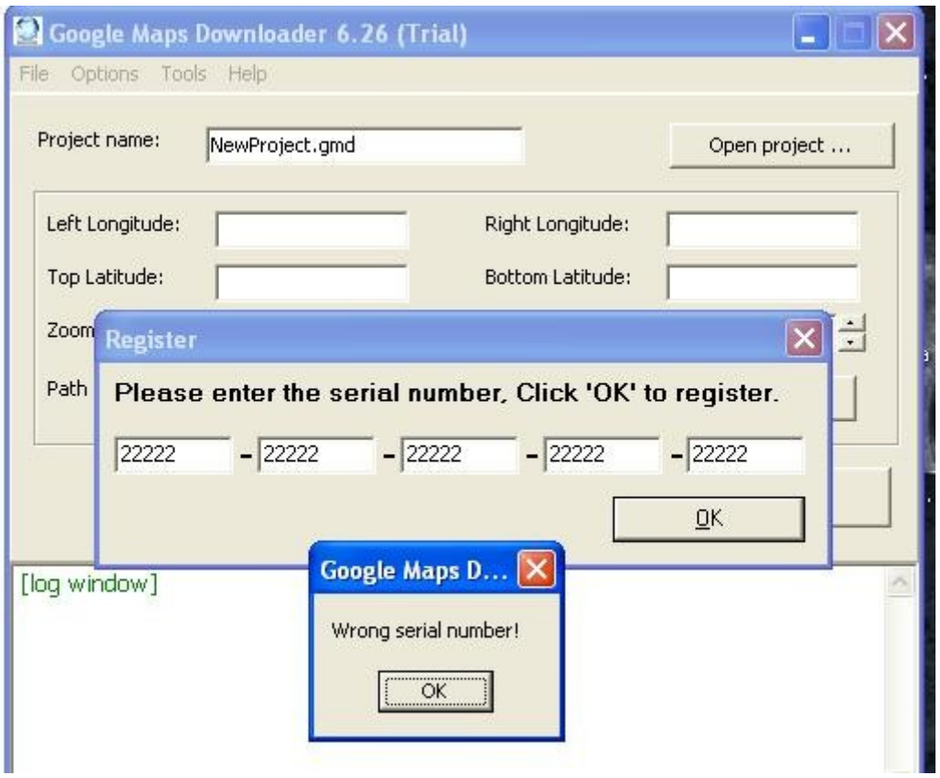
Así que llegamos a la conclusión de que no nos va a dar mucho dolor de cabeza :)

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.

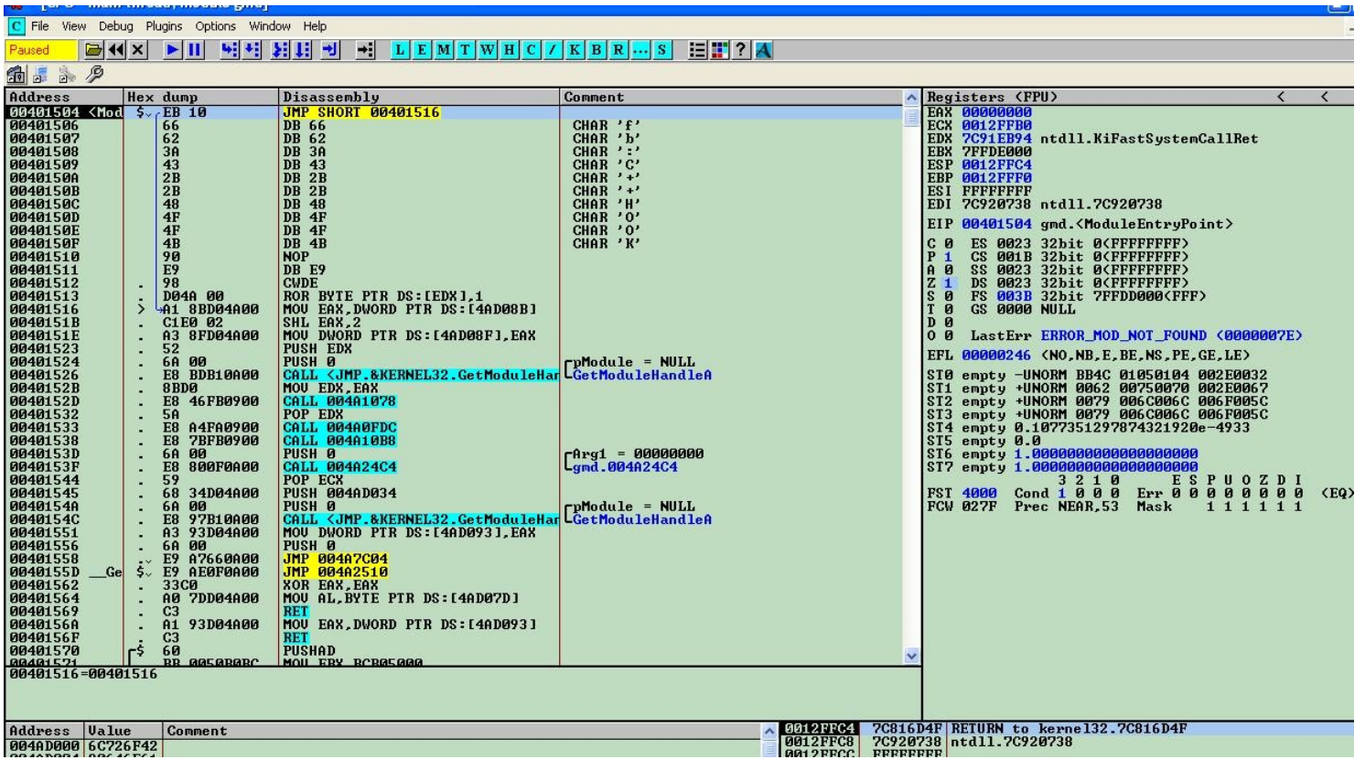
Así que vamos allí e ingresamos lo primero que se nos viene a la cabeza.





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



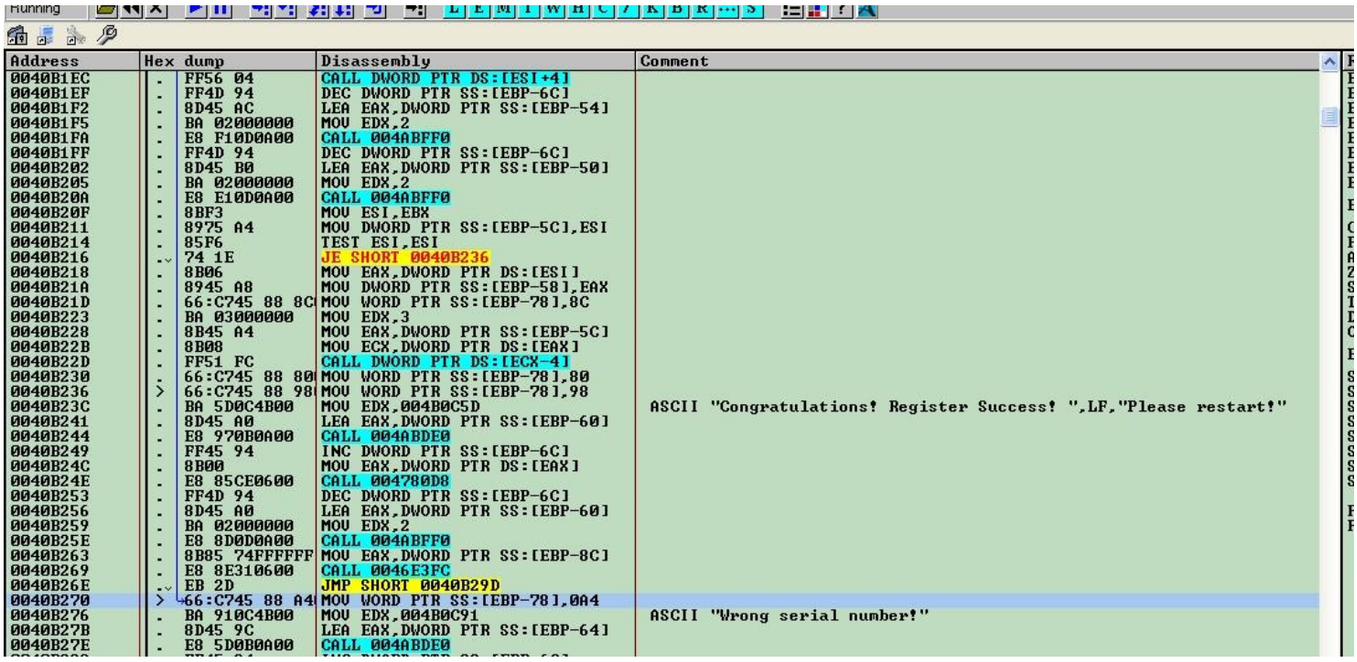
Acá hay varios caminos para tomar, y todos nos conducen a Roma. Es decir se puede elegir el método de búsqueda por las Apis, yo he intentado varios, pero el más rápido que recuerdo es el de buscar la cadena de texto que nos indicaba nuestro serial malo: “Wrong serial number”

Hacemos clic derecho en la ventana del desensamblado y search for → all referenced texts strings

Y luego en search text (obviamente vamos al inicio de la ventana para que busque desde el principio)

Una vez que encontramos nuestra cadena de texto vemos inmediatamente arriba un mensaje de felicitaciones por haber registrado el programa.

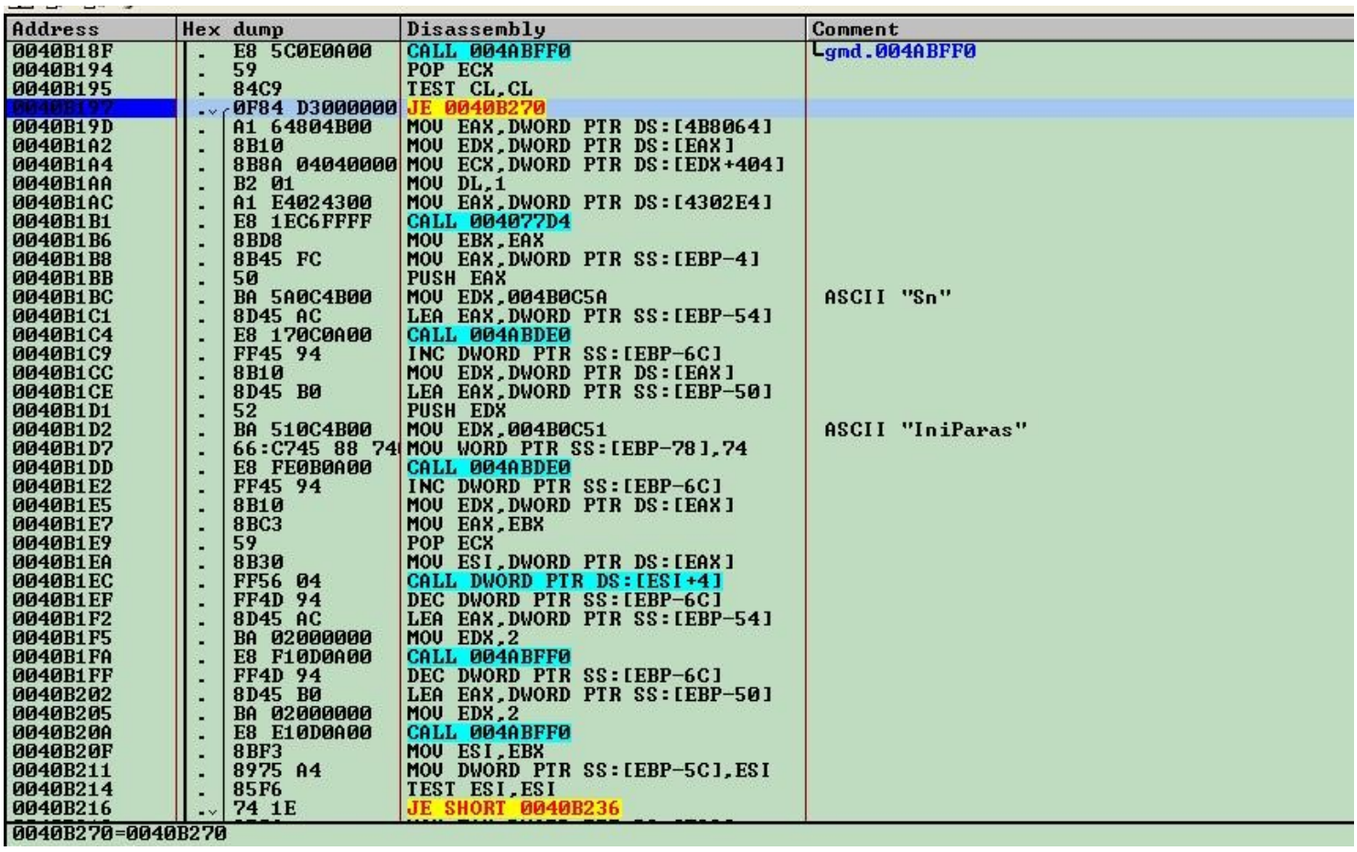
Hacemos doble clic en Wrong serial .. y llegamos a:



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.



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-C]
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

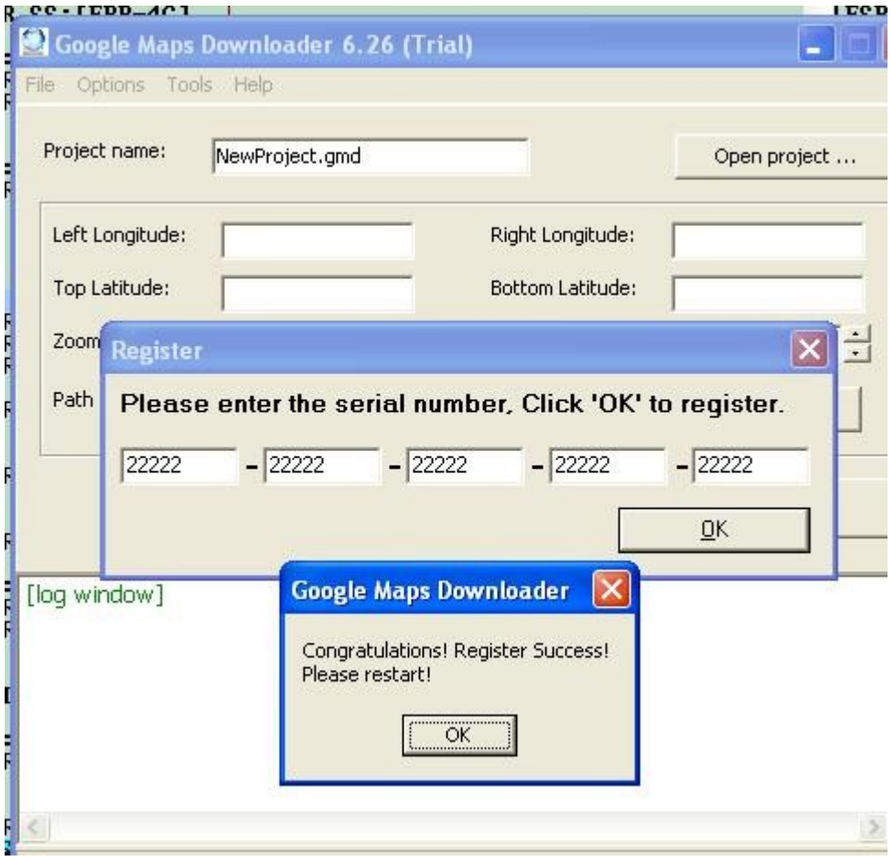
```
Address   Hex dump     Disassembly
004DC000  > 59         POP ECX
004DC001  . B1 01      MOV CL,1
004DC003  . 84C9       TEST CL,CL
004DC005  .- 0F84 65F2F2FF JE 0040B270
004DC00B  .- E9 89F1F2FF   JMP 0040B199
004DC010    90         NOP
004DC011    90         NOP
004DC012    90         NOP
```

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.

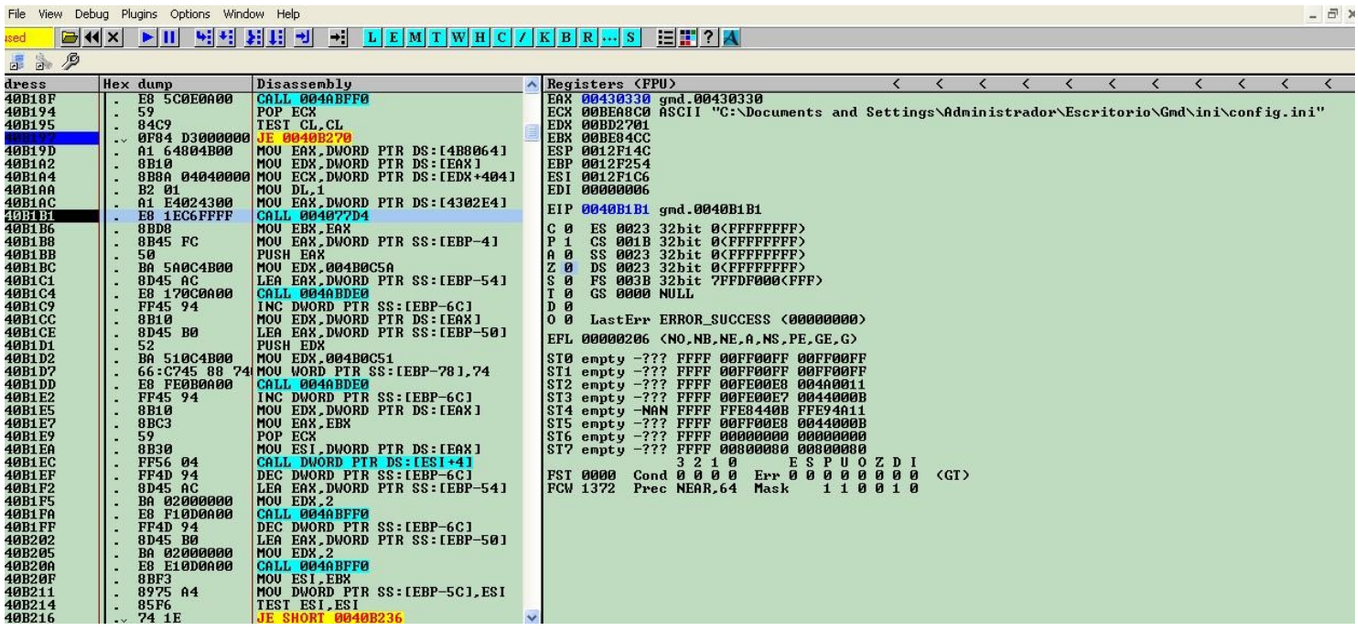
Fin Nota.

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

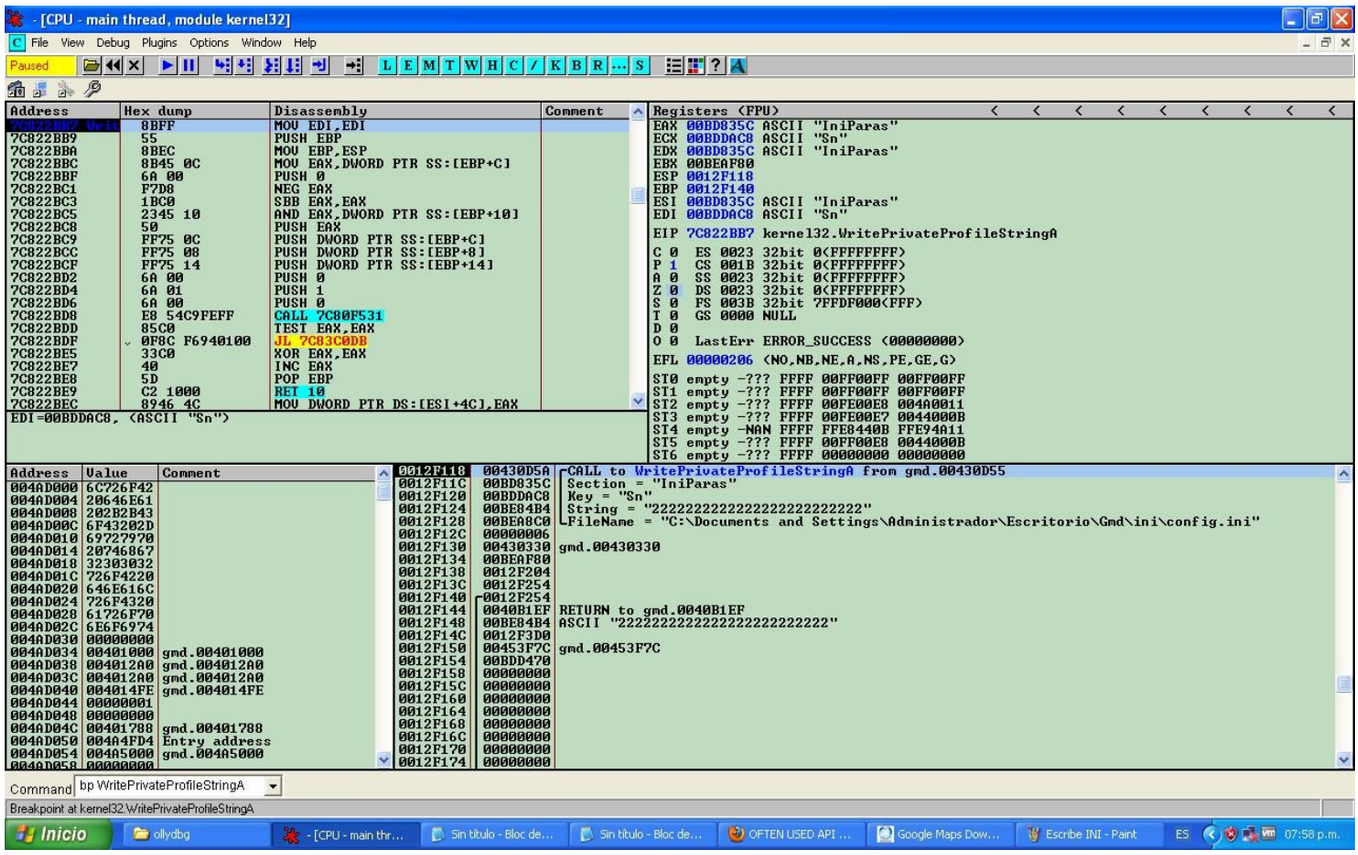


Bueno, elegimos cualquiera de los métodos que mencioné y (es más cómodo cambiar el flag!) Felicitaciones Registrado satisfactoriamente y nos invita a reiniciar!

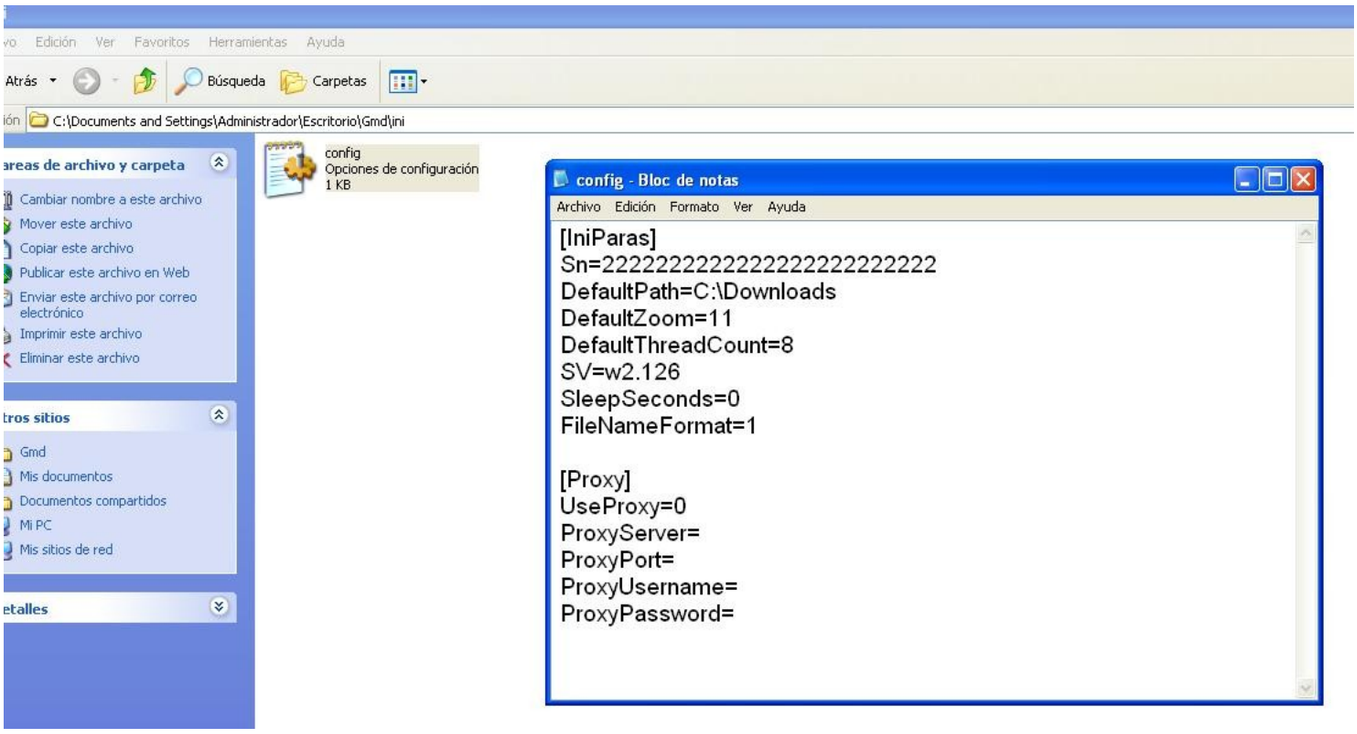
Tambien si rastreamos un poco encontramos que va a escribir config.ini en la ruta de nuestro programa



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



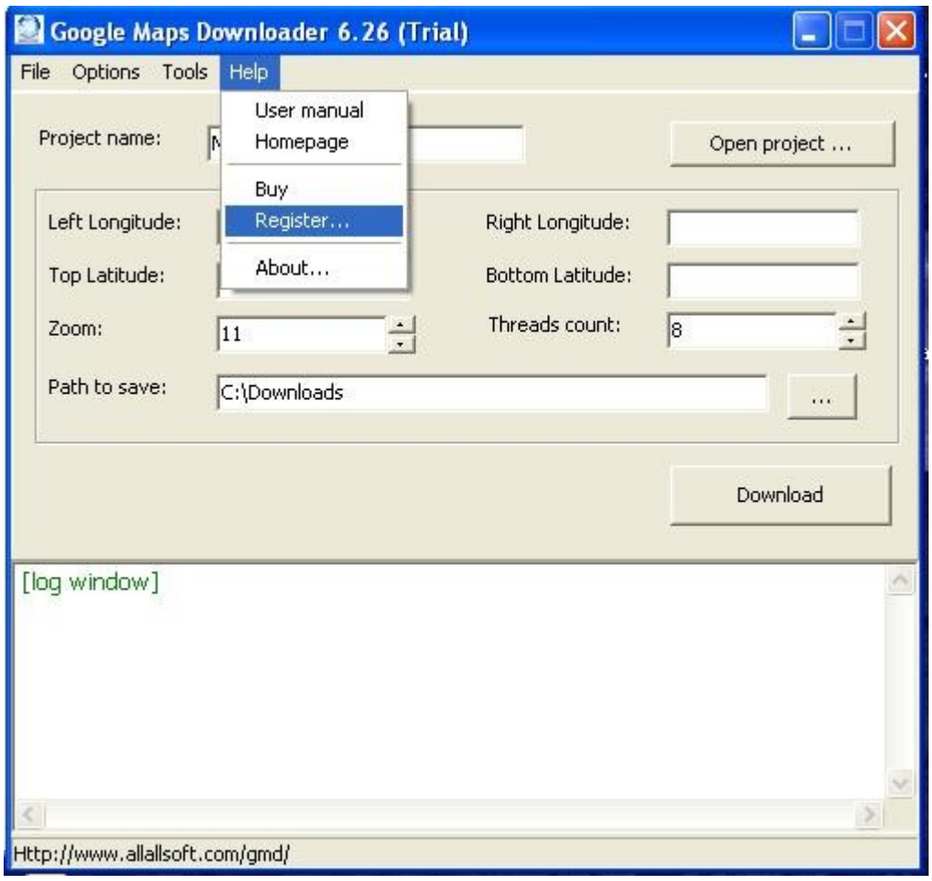
Vamos a la carpeta del programa y buscamos algún archivo ini y nos encontramos con este:



Vemos que en Sn esta nuestro serial guardado.

****

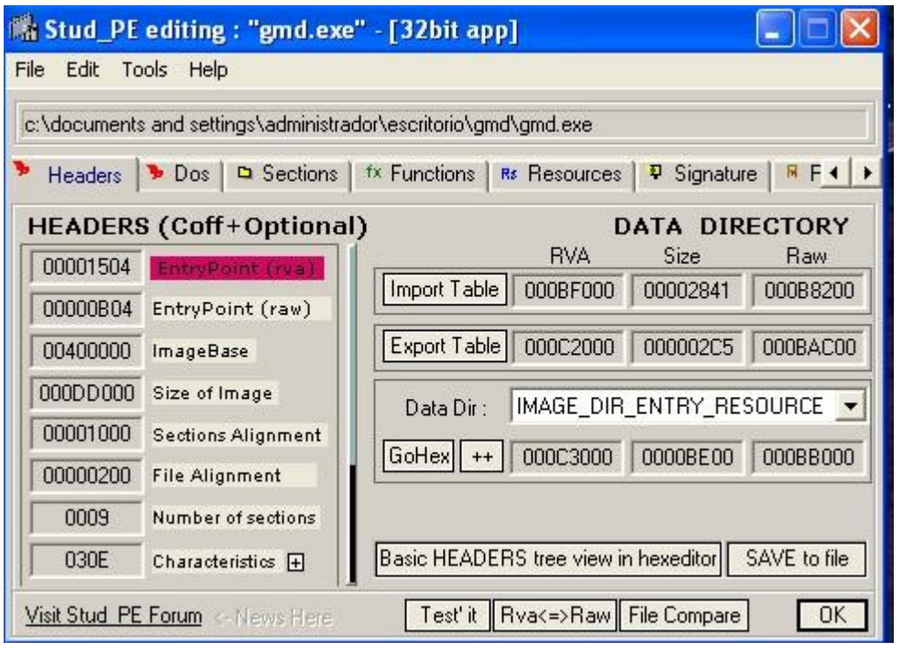
Reiniciamos contentos y vemos esto:



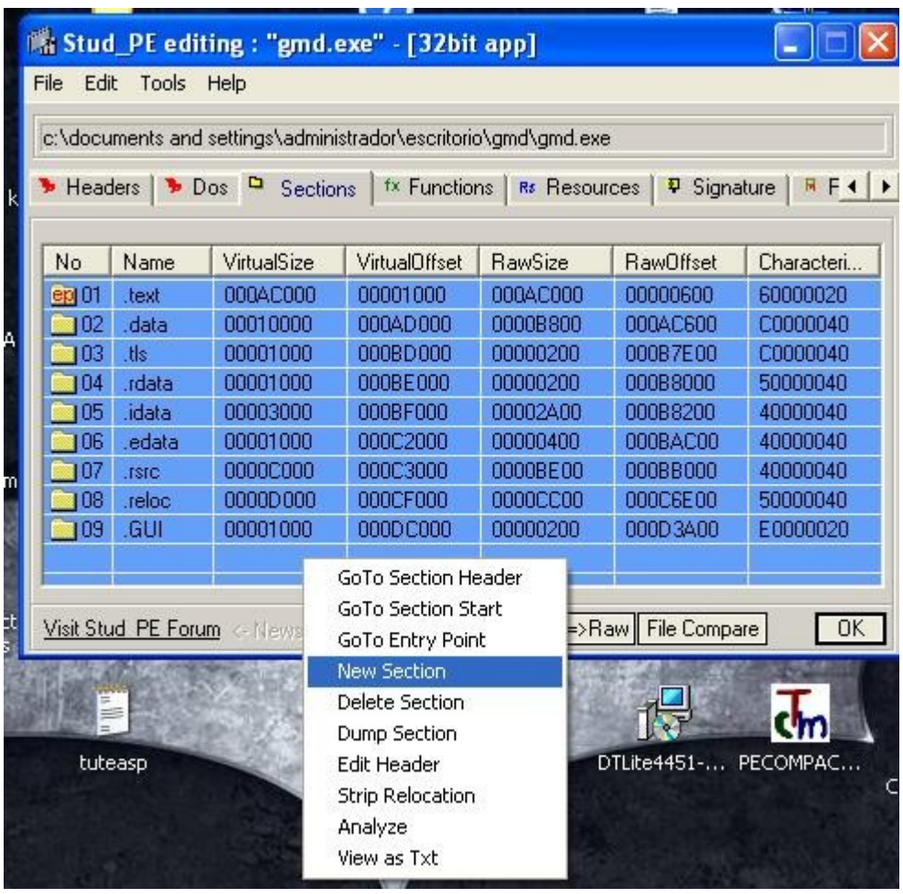
Uhh.. volvemos a cero, nuestro trabajo no fue efectivo??.. La respuesta es: Más o menos, estamos a medio camino, es decir lo que hemos hecho es forzado a que escriba en el archivo .ini del programa nuestro serial que obviamente es chequeado al reiniciar el programa y no nos registra.

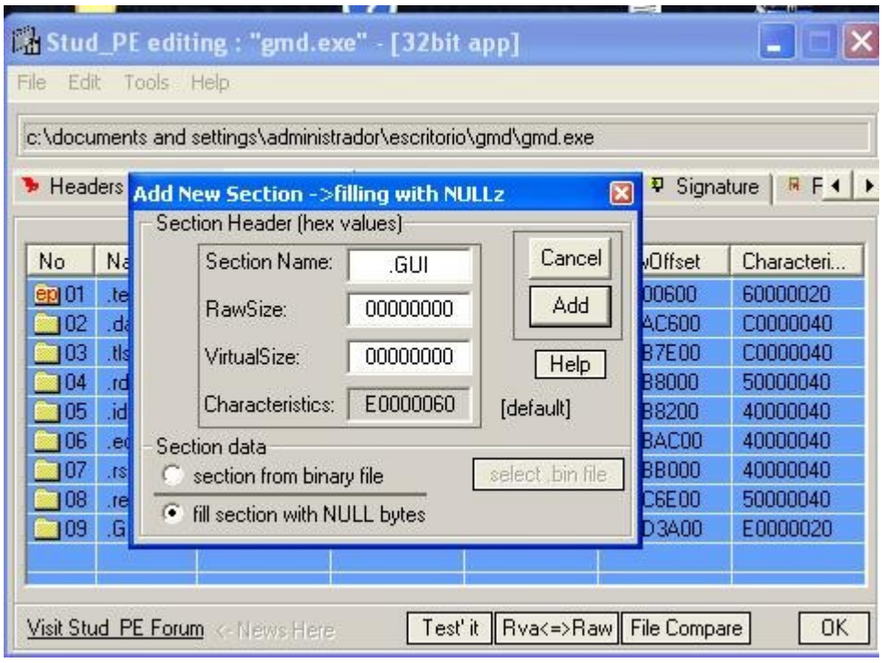
Llegué a la conclusión de que hay que parchear para arreglar este lío! y como dije antes agregar una sección para no tener que andar buscando huecos y que a veces se sobrescribe con información no prevista.

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



Clic derecho y elegimos crear nueva sección



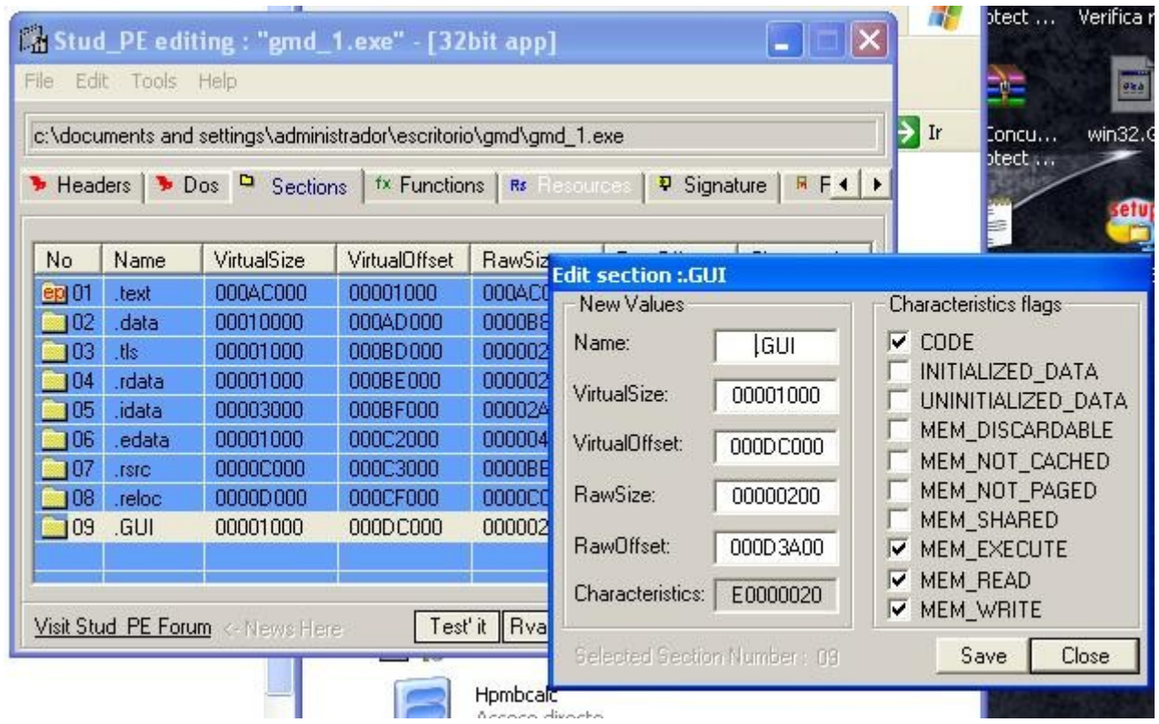


Elegimos el nombre que queremos y luego add, y ok

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

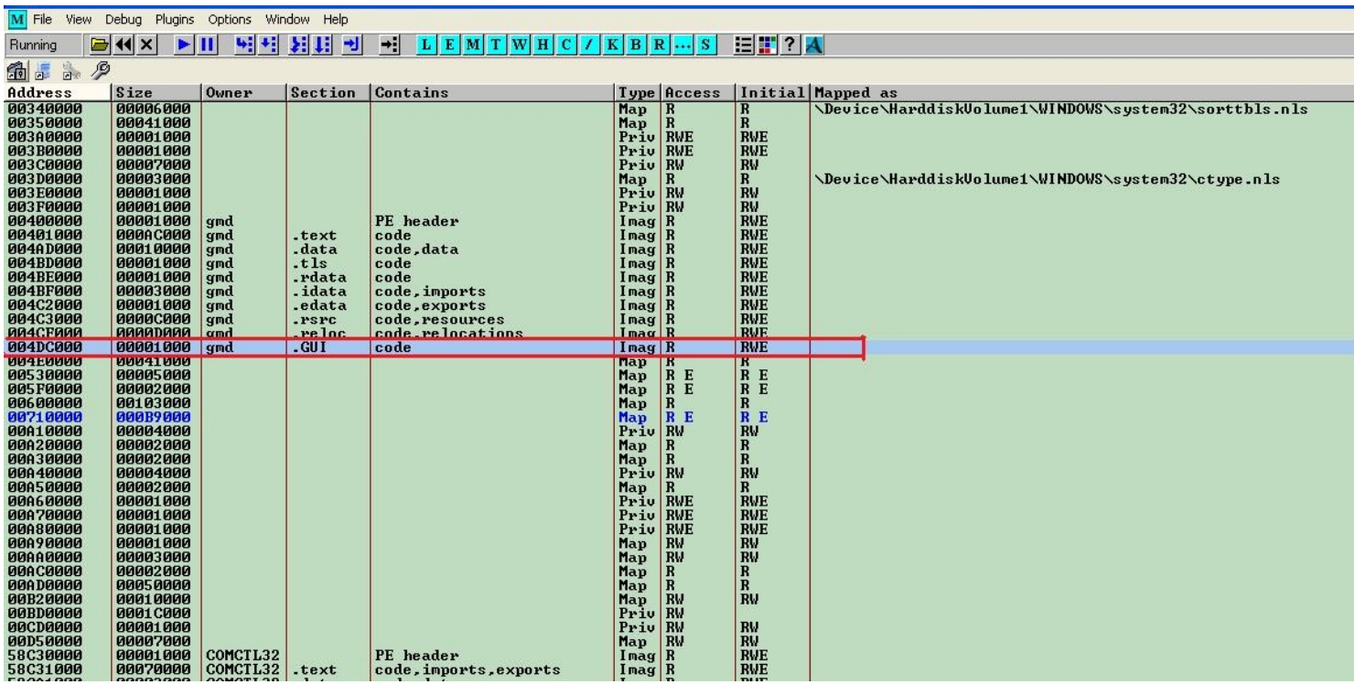
Y le asignamos los bytes que deseemos en mi caso le puse 100 bytes tanto en rawsize como en virtual (virtual y disco) y cliqueamos en donde dice que llene la sección con null bytes.

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:



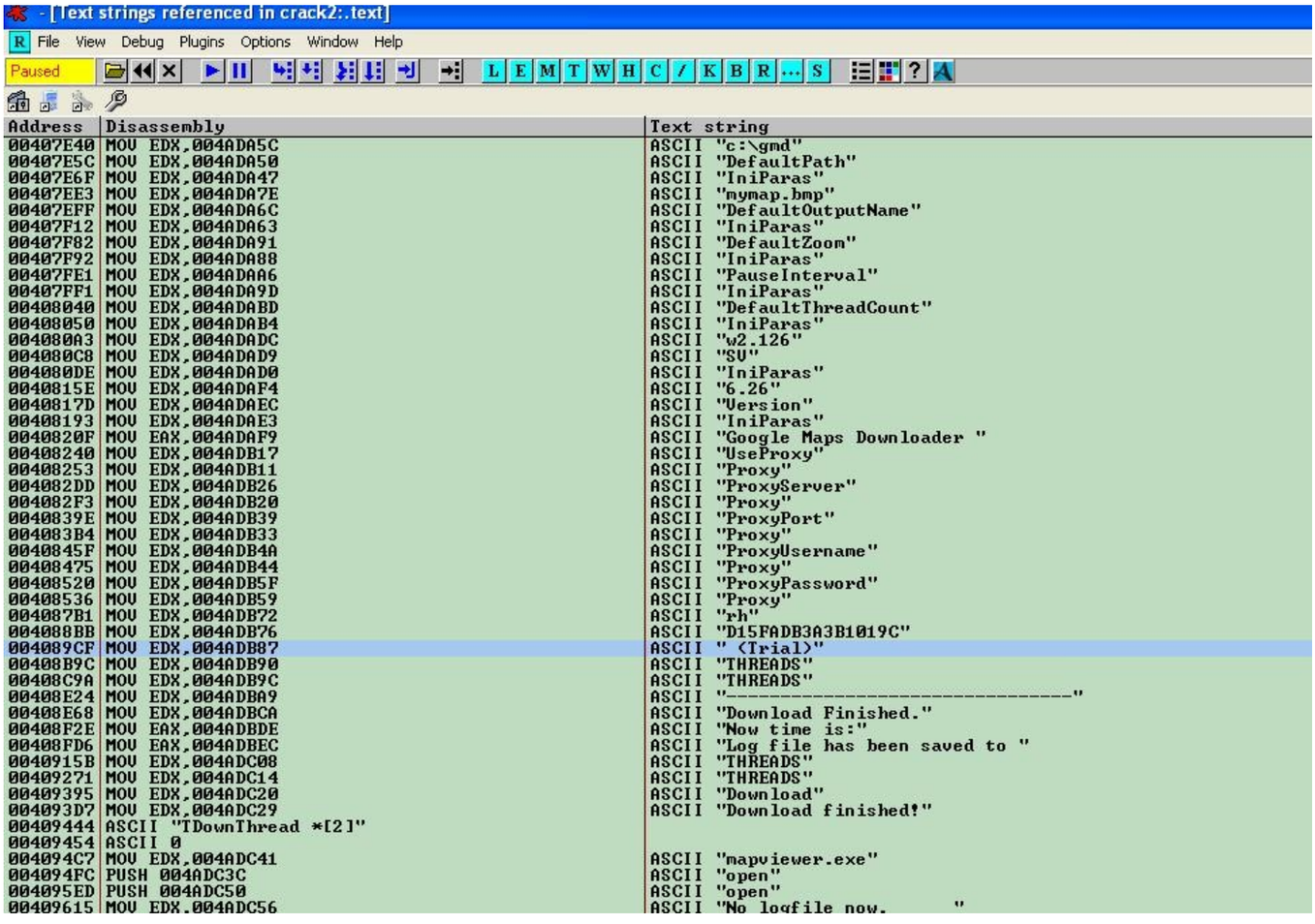
Ahí se ve en 4DC000 empieza.

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

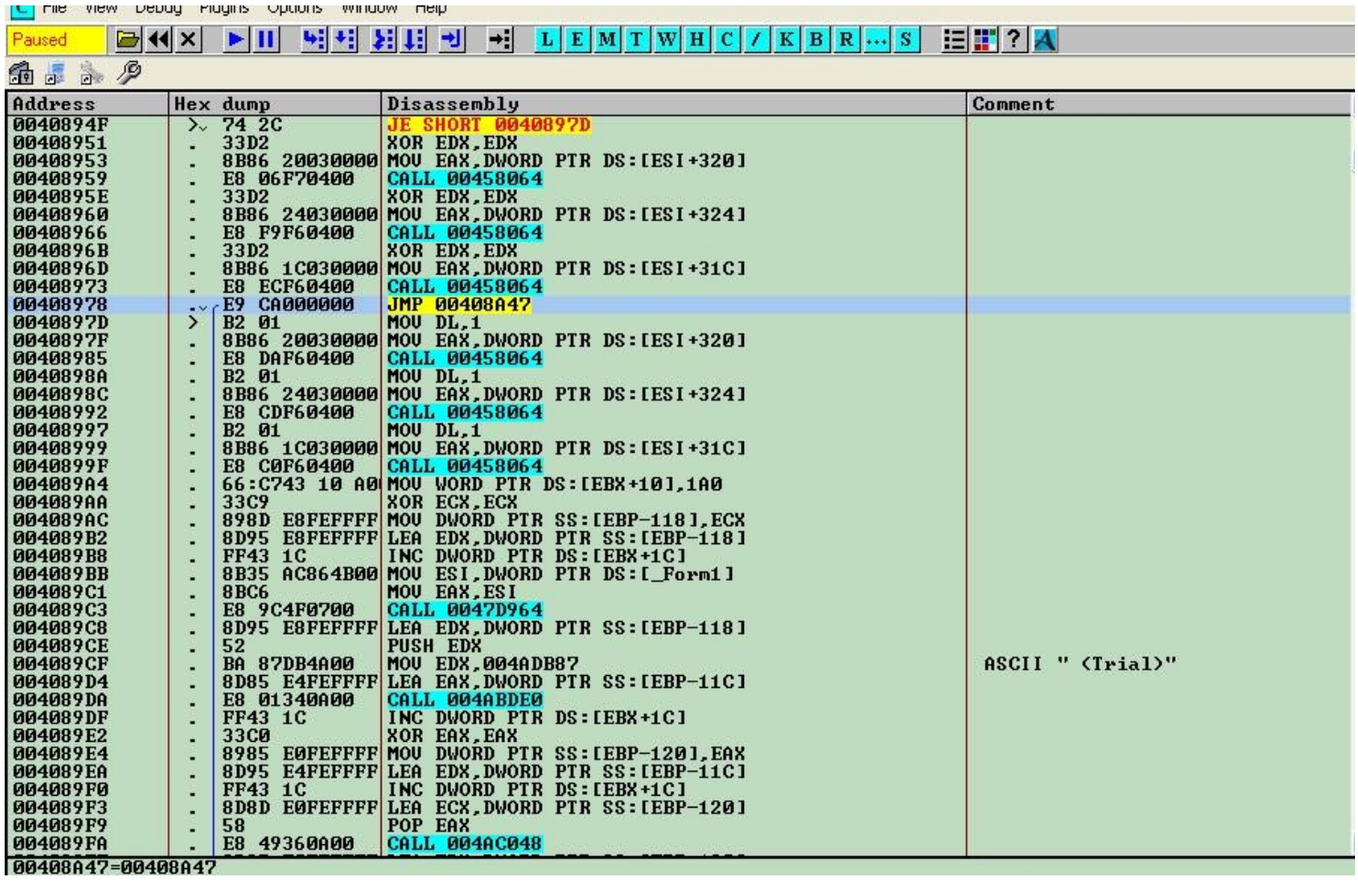
Es evidente que si estamos registrados no tendría que mostrarla.. así que la buscamos entre las strings en el Olly.

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

Y vemos esto:

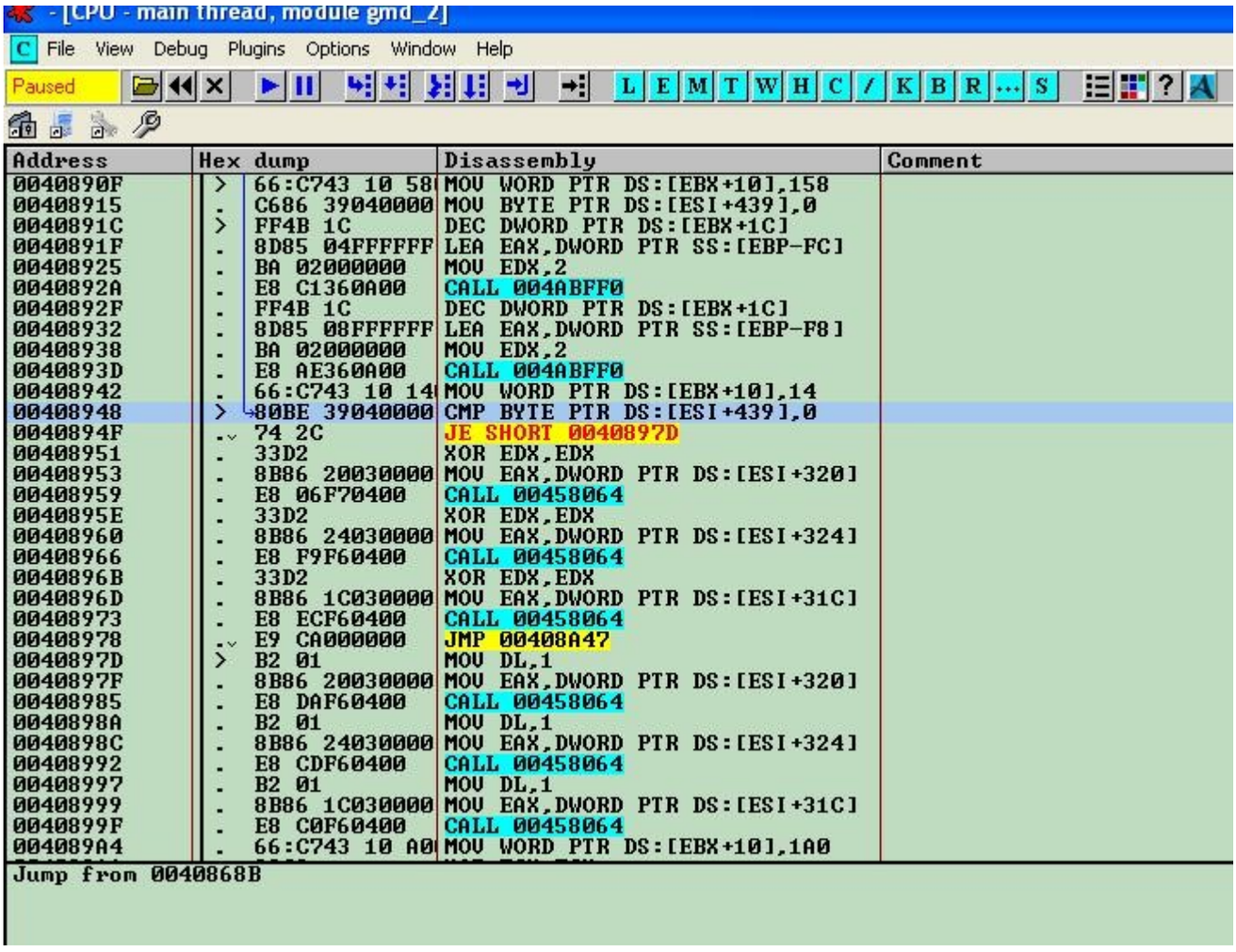


Le hacemos doble clic y llegamos



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.

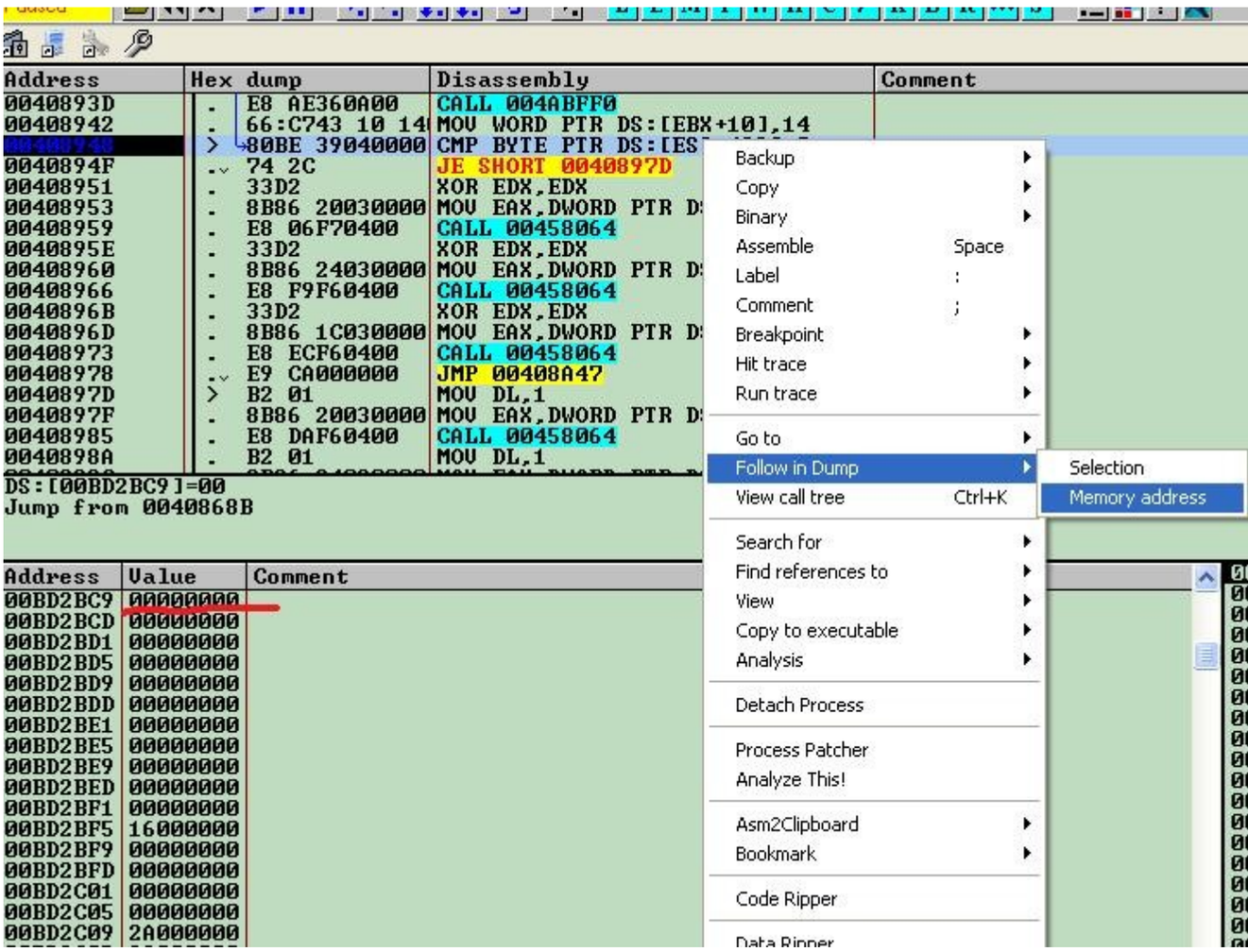
Subimos un poco y lo vemos más claro:



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

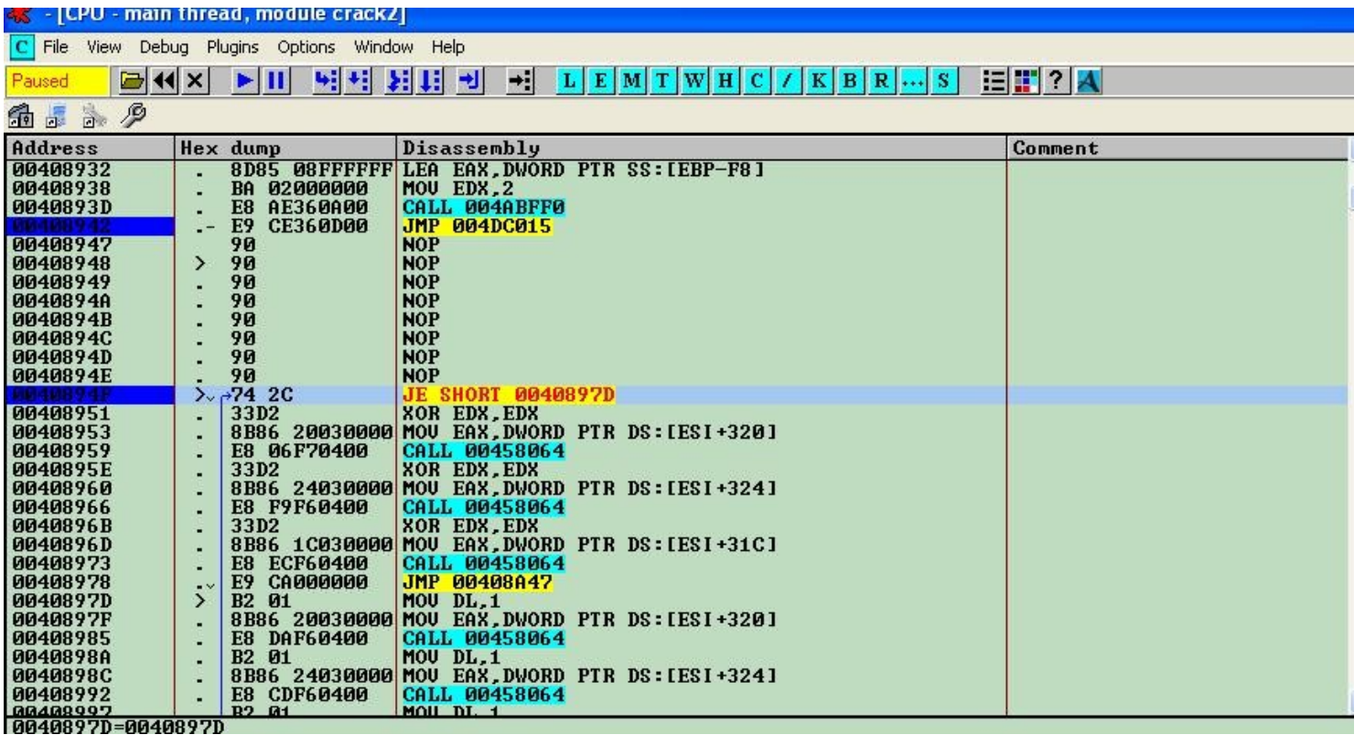
Si lo seguimos en el dump confirmamos el valor:



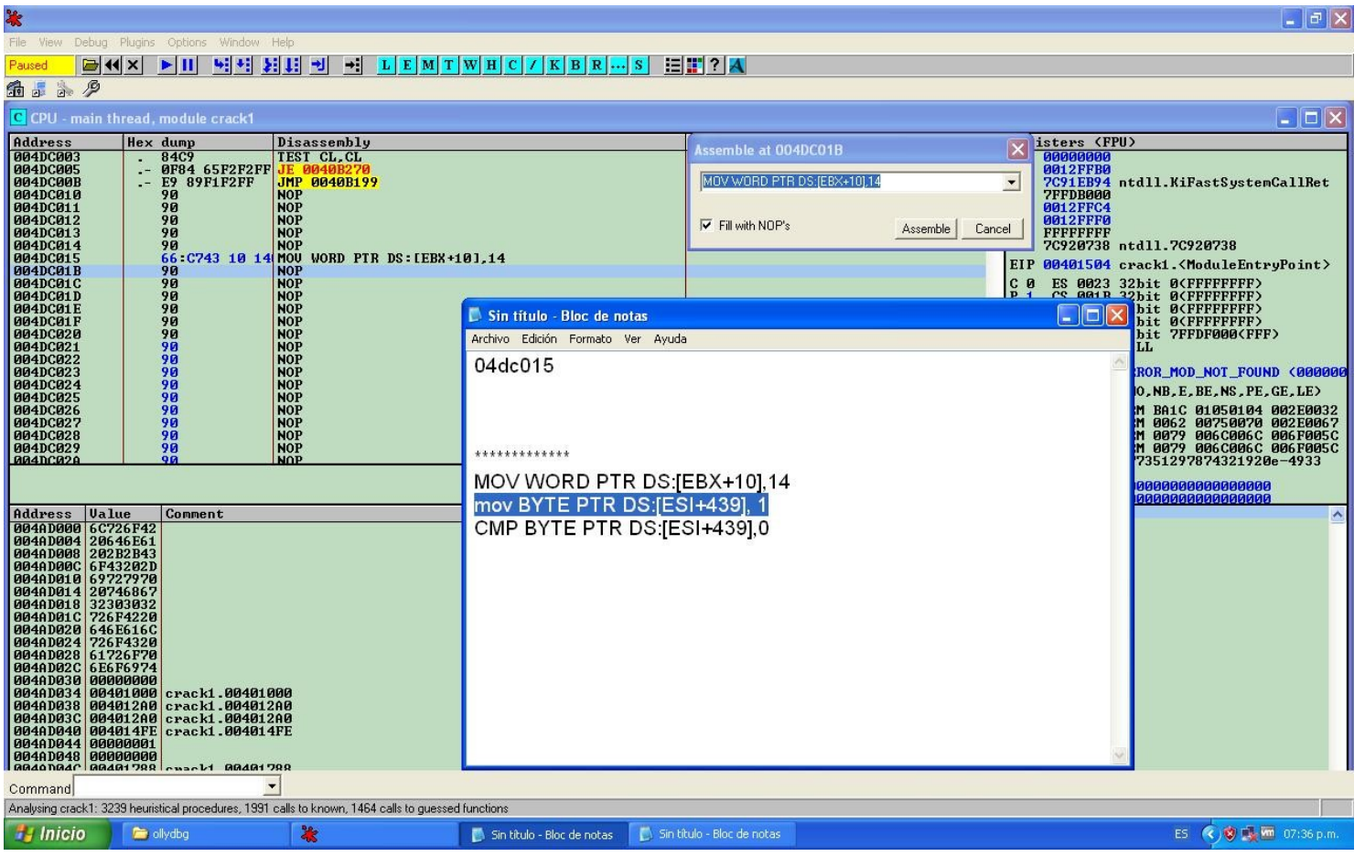
Ahí lo subrayé, 000000000 ☹

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:

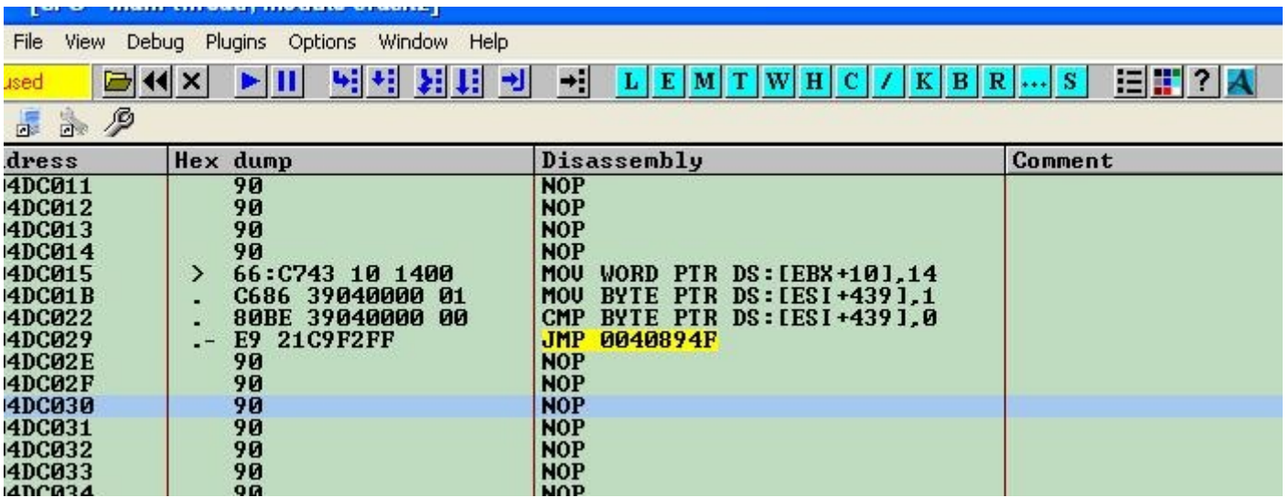


Asi que como habíamos tomado nota vamos a nuestra sección y empezamos a escribir el código:



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 asi, con un salto final de regreso a la comparación:

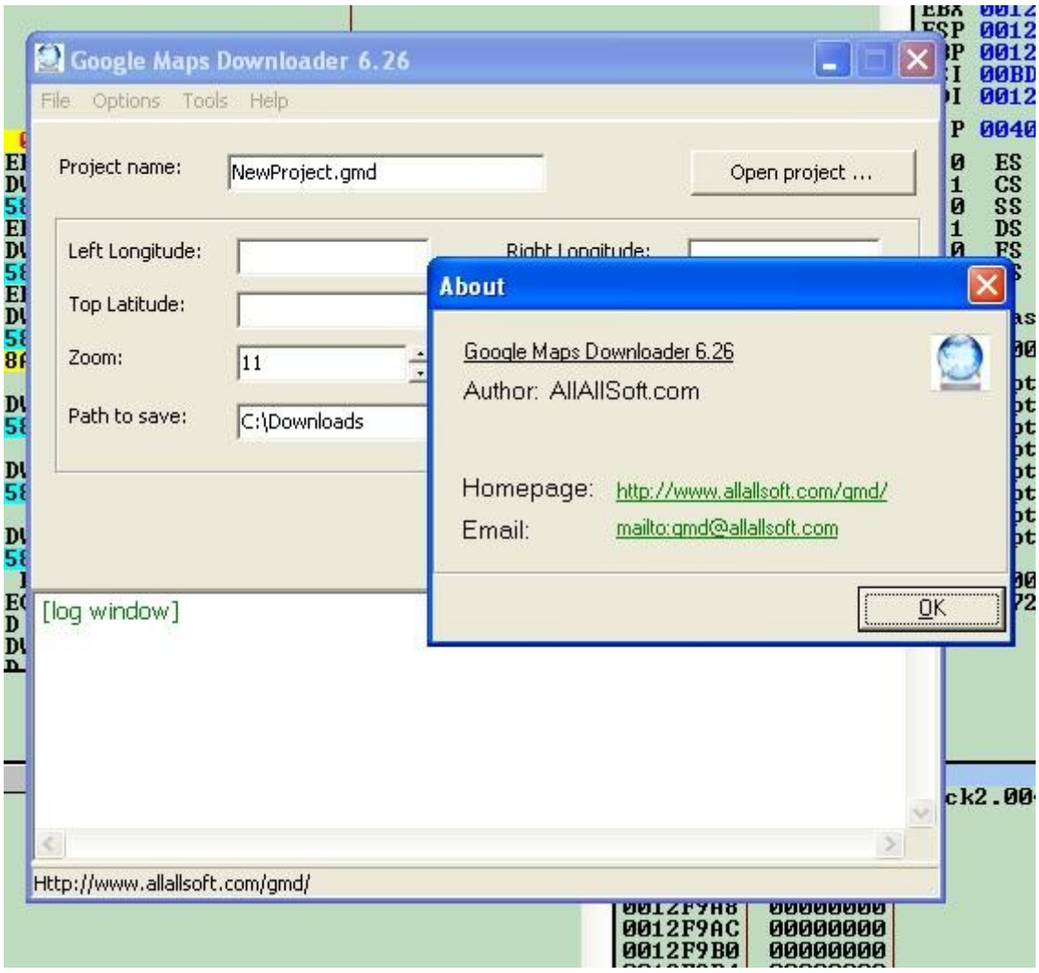


A continuación guardamos todos los cambios en save file—all modifications y lo guardamos con otro nombre

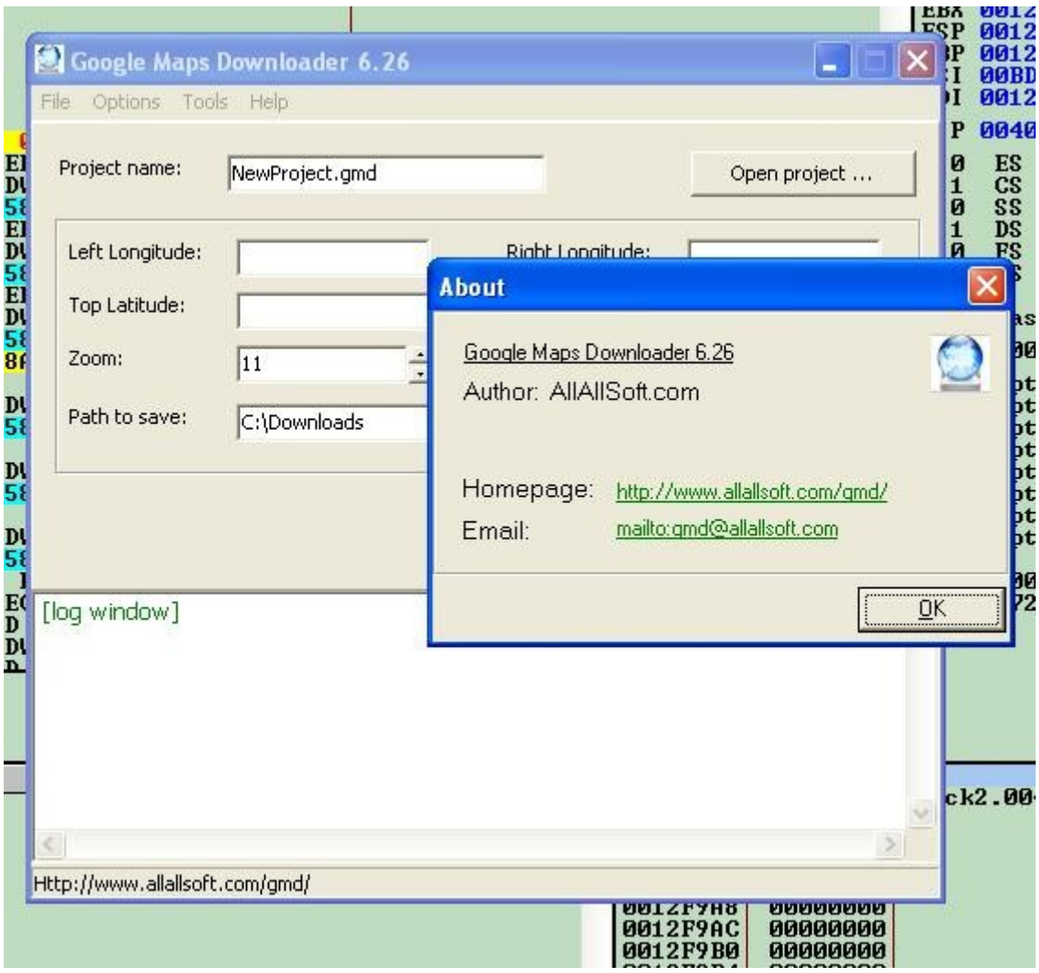
Bueno terminamos, ahora es hora de la prueba de fuego, esperemos no quemarnos!!

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.