INTRODUCCION AL CRACKING CON OLLYDBG PARTE 27

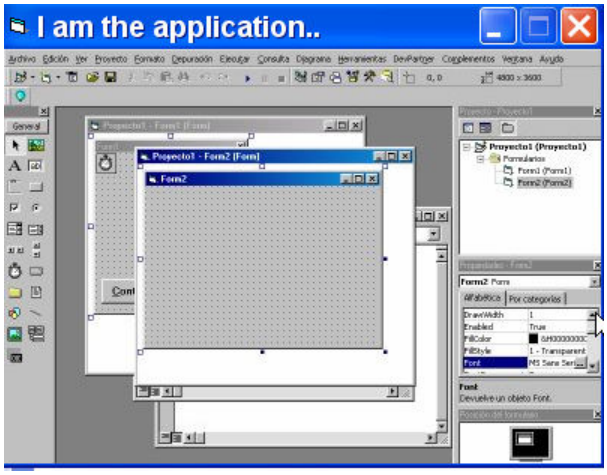
Seguimos con Visual Basic

Bueno seguiremos con Visual Basic aquí tenemos un crackme en el que hay que quitar una nag, yo lo hice a mi modo, que me sirve para cualquier caso (me gusta razonar e intentar), y quite la nag a mi forma, por supuesto use el OLLYDBG especial para VB que vimos en la parte anterior, luego de ver como lo solucione, veremos el método mecánico que ahorra tiempo, pero yo pienso que hay que saber solucionar o por lo menos intentar usando los métodos de intentar y probar, lo que uno aprende intentando es muy importante, de allí que hago este tute con mi forma de quitar la nag, al final les mostraré el método mecánico también porque hay que saber y usar de todo.

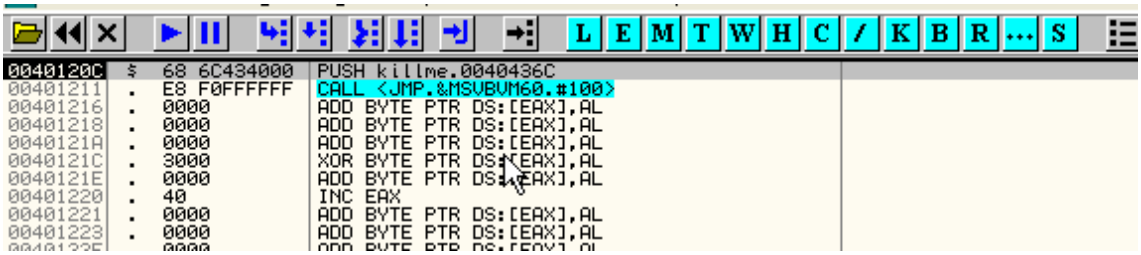
Killme es el adjunto que tiene el crackme que vamos a estudiar, en este no hay serial solo una nag molesta que hay que evitar que aparezca y que corra el programa que viene a continuación.

Si lo corro fuera de OLLYDBG veo la nag

Veo que tiene un botón deshabilitado y un timer que va desde cinco disminuyendo hasta cero y cuando llega a cero, se habilita el botón CONTINUE y al apretarlo.



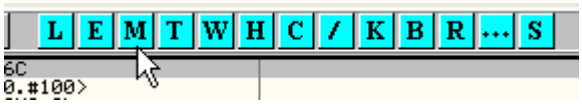
Aparece la supuesta aplicación, que debe correr directamente sin aparecer ninguna nag, iremos estudiando de a poco como funciona el crackme, todo con OLLYDBG.
La primera tentación en cualquier programa con NAG es ver los JMPS que dirigen a diferentes partes del programa.

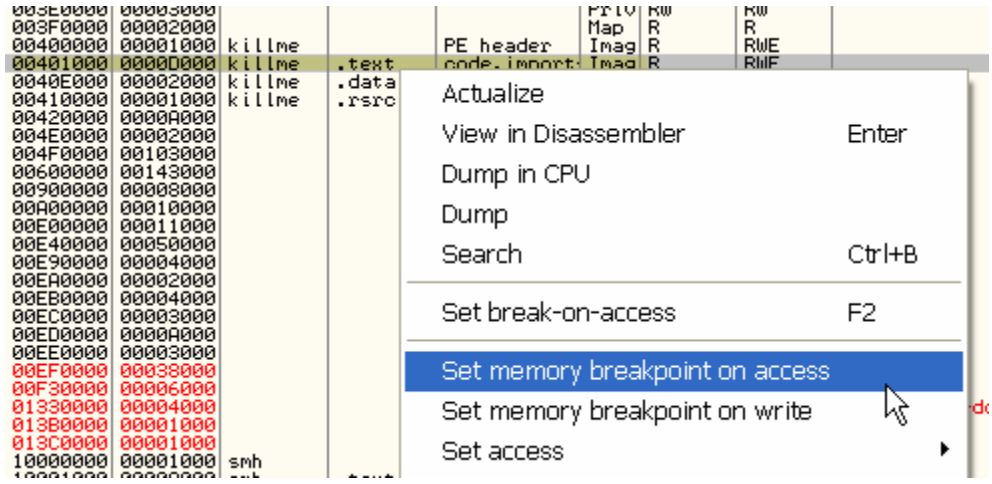


Arranco el programa y apreto F9

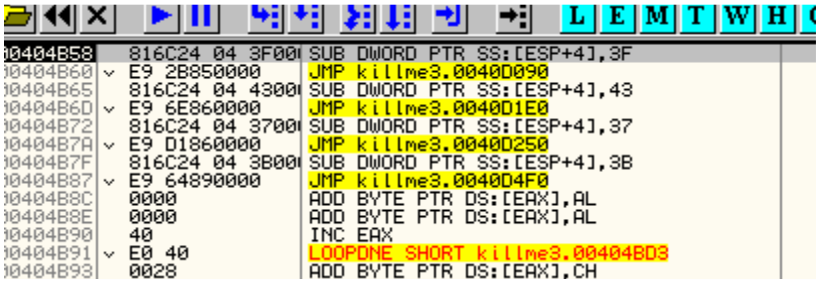
Ahí paso el tiempo y apareció habilitado el botón CONTINUE.

Sabemos que al cerrar la nag, pasara por un JMP en la sección code, por lo cual pongo un BPM ON ACCESS en dicha sección (que será en realidad ON EXECUTION) y apreto el botón Continue.

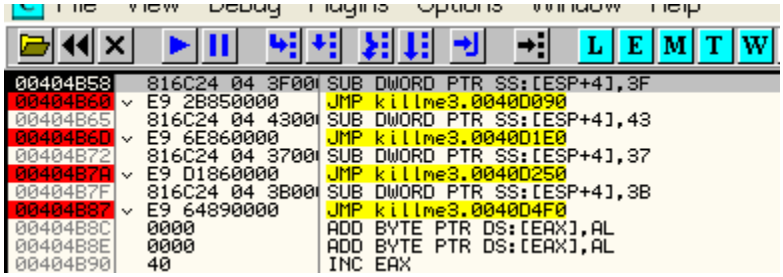




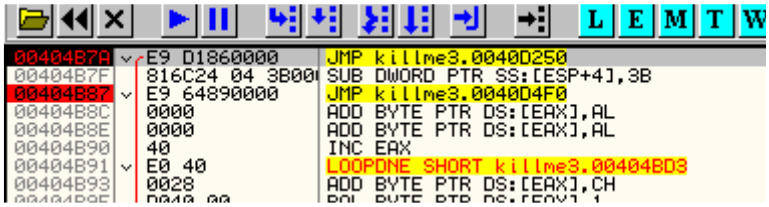
Para aquí



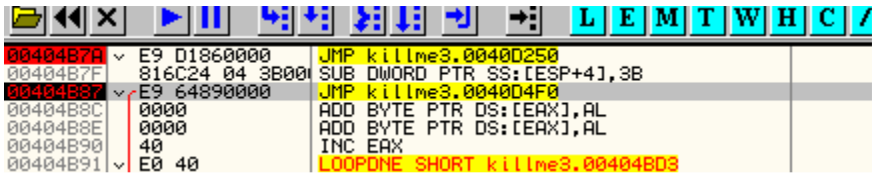
O sea que la parte del programa se comienza a ejecutar en 40d090 ya que allí vemos el salto JMP 40d090, a donde se inicia dicha parte, y vemos los restantes JMPS a las diferentes partes del programa, pongamos BP en todos estos JMPS para ver cuando para y en que caso.



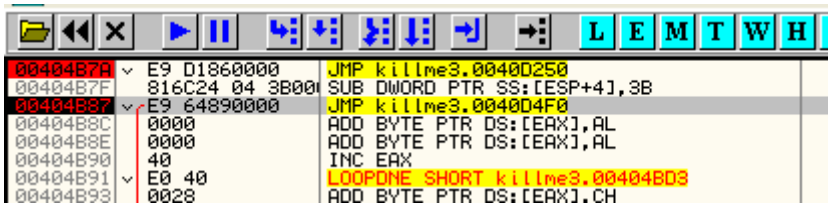
Reiniciemos el programa



Vemos que antes de salir la nag para aquí, una buena tentación es cambiar el JMP por JMP 40d090 así salta al programa directamente, pero da error y no arranca, así que demos RUN de nuevo.

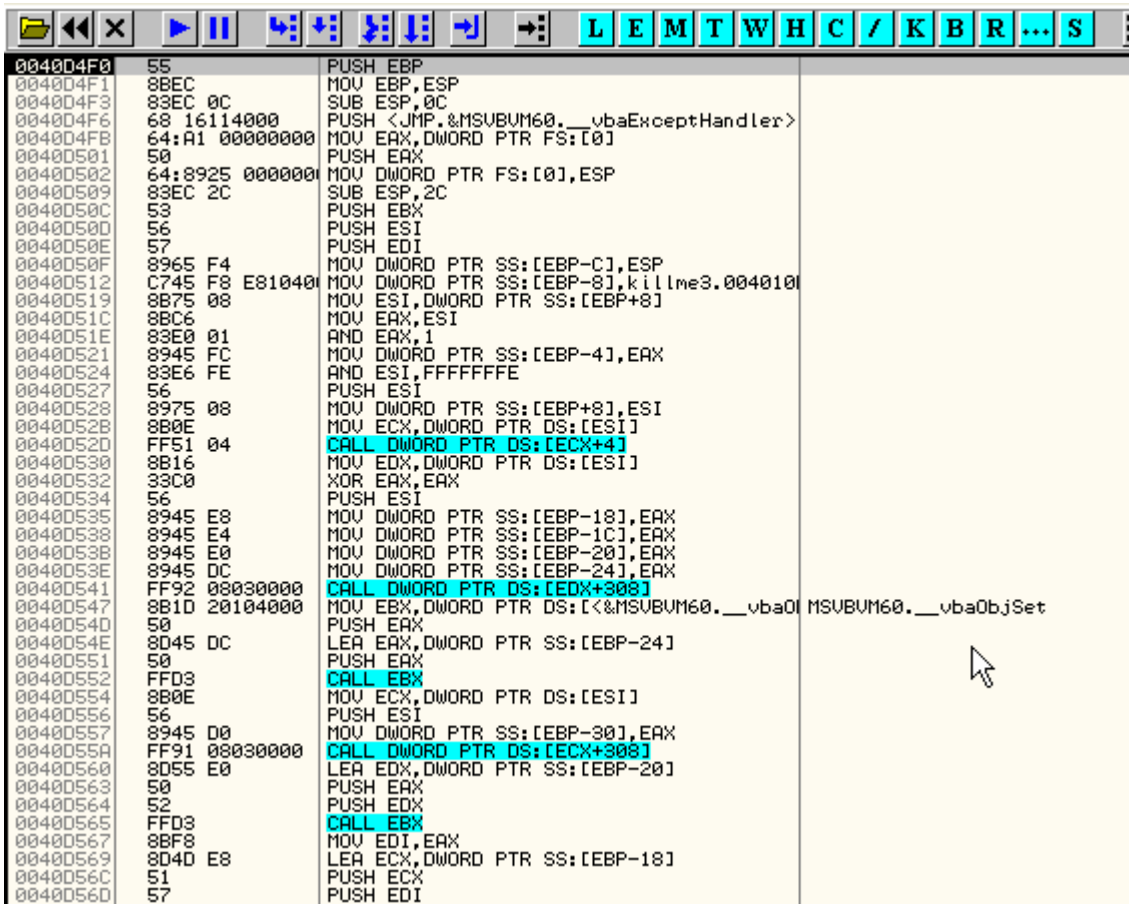


Vemos que para en el siguiente JMP y si doy RUN de nuevo.



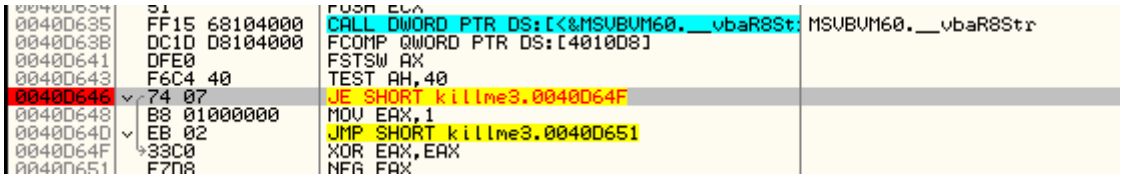
Tarda unos segundos y vuelve a parar y así 5 veces lo cual me dice que esa rutina tiene que ver con el temporizador, ya que para cada paso que el mismo disminuye.

Veamos que hay en dicha rutina traceemos en ella.

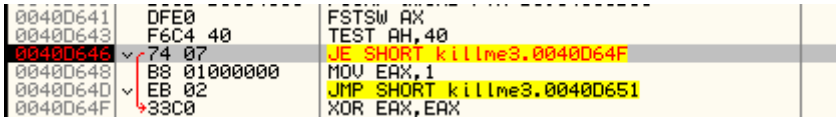


Nada lindo sigamos traceando con f8

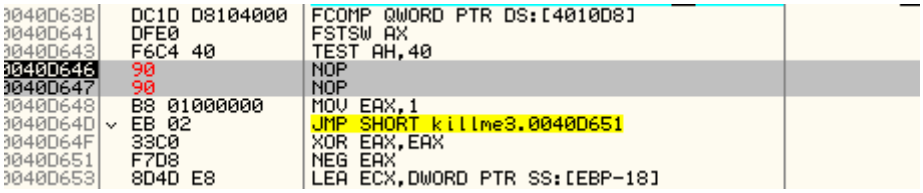
Llego a algunos saltos condicionales, puedo probar que pasa si los invierto a ver si termina el temporizador y se habilita el botón, mirando un poco e intentando veo que el que sirve es este



Es una comparación en punto flotante, que si no se que es por ahora, pues probando invertir los diferentes saltos de la rutina que son pocos, me daré cuenta.



Veo que las primeras veces que pasa por allí salta y la última vez no salta, calculo que es la comparación de si el temporizador termino de contar y llego a 0, invirtamos nopeandolo a ver que pasa.



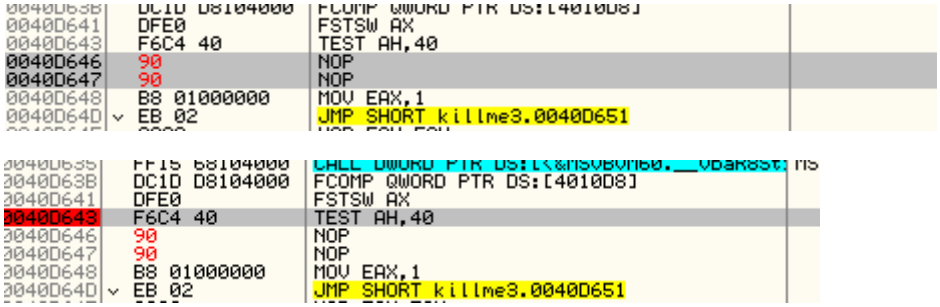
Lo nopeo para que no salte

Y doy RUN

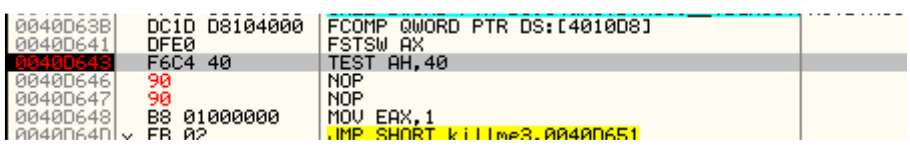
Veo que acerté pues aun siendo la segunda vez que paso por dicha rutina, al invertir el salto termino la cuenta del temporizador y habilito el botón CONTINUE, así que por ahí viene la cosa,
Ahora si ya forzamos a que el botón se habilite que ocurrirá si al retornar de esa rutina, obligamos a saltar al inicio del programa, pues ya esta todo inicializado y solo falta apretar el botón, nada mas.

Reiniciemos el programa

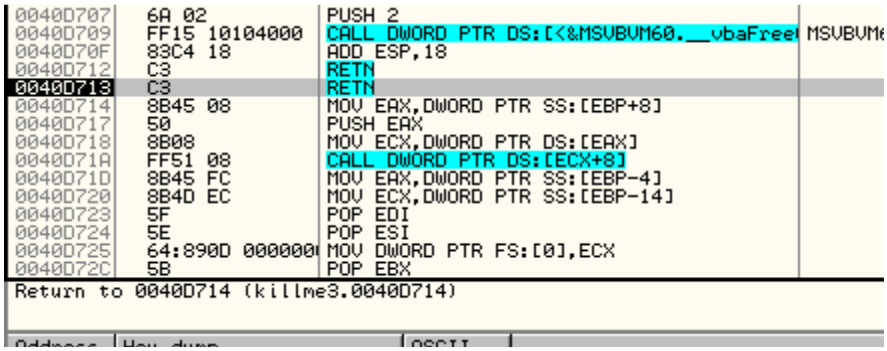
Vayamos a la línea que nopeamos antes de ejecutar nada



y quitemos todos los BP y pongamos uno allí, demos RUN

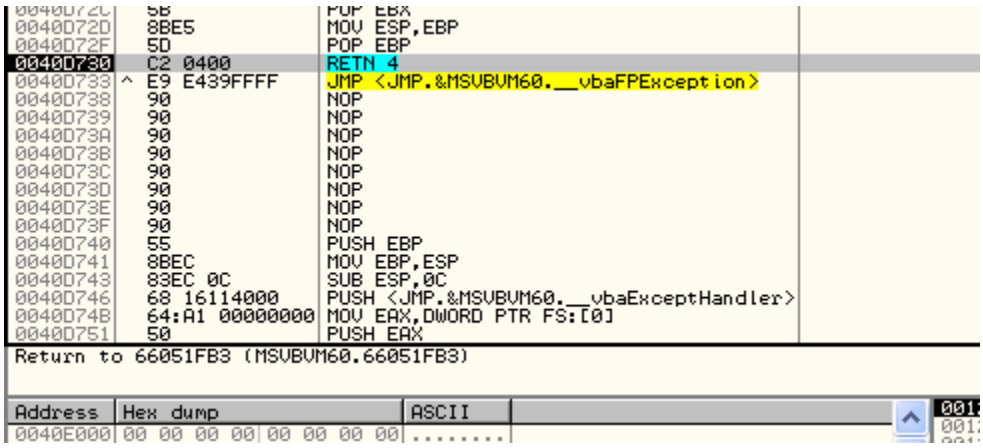


Allí paro traceemos a ver cuando la rutina termina y vuelve a la dll de visual Basic

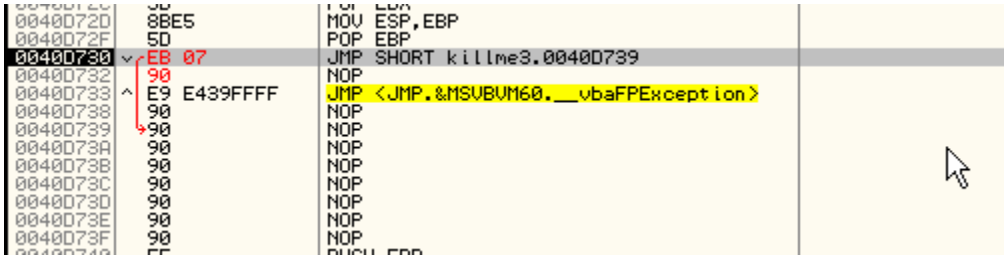


Vemos que llega al RETN pero no sale a la dll de visual sino que sigue a 40d714

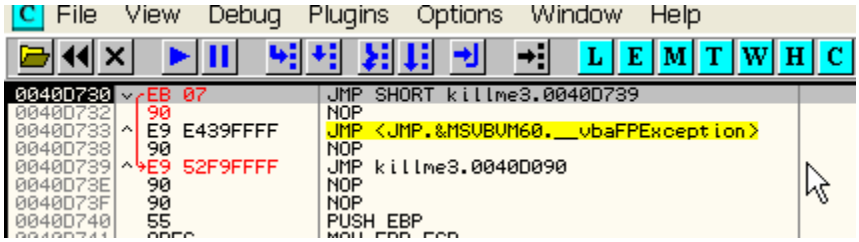
Sigamos traceando



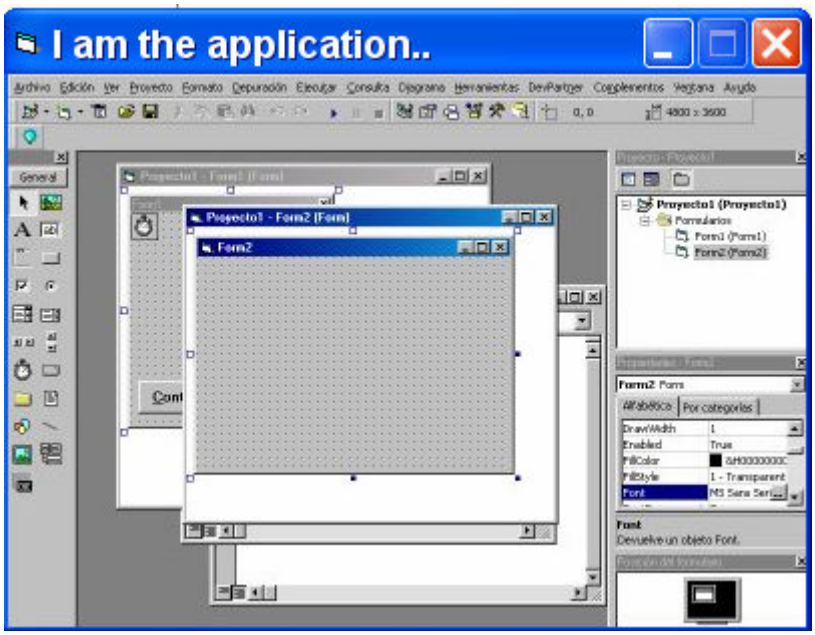
Allí si llega al RETN que vemos en la aclaración que volverá al dll de visual, aquí ya esta todo inicializado para arrancar el programa, así que puedo intentar cambiar ese RETN 4 por un salto al inicio del programa veamos.



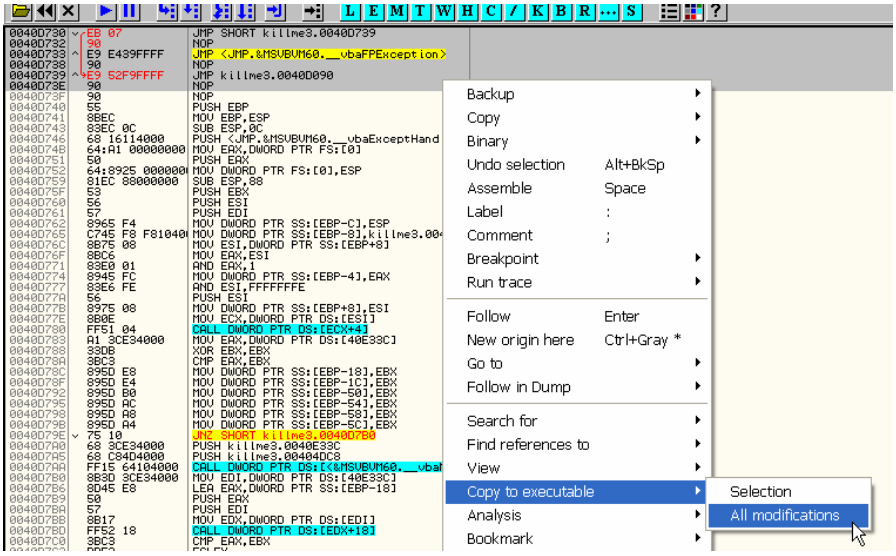
Salto allí que hay mas lugar para escribir y luego salto a 40d090 que era el inicio del programa.



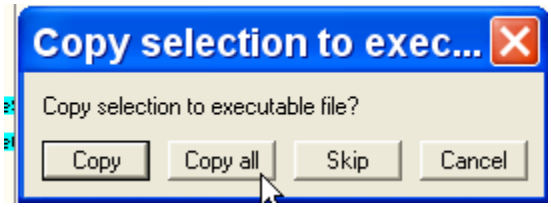
Probemos si arranca demos RUN



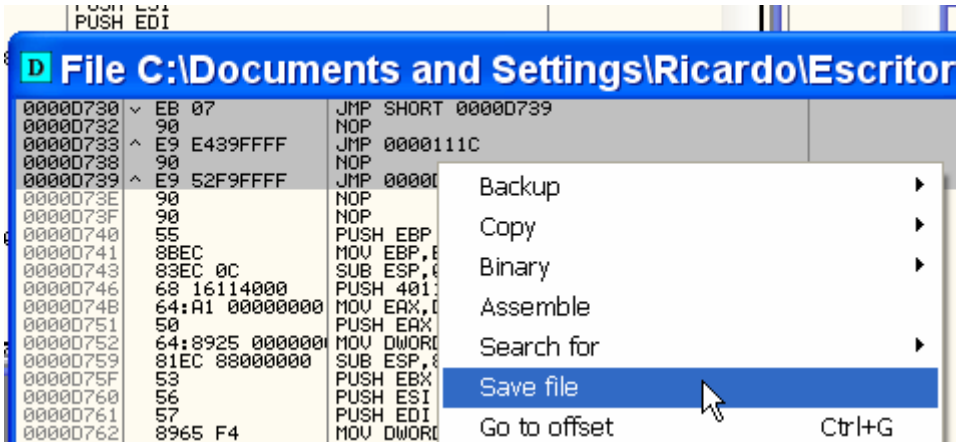
Ahora podemos guardar los cambios, el nop inicial y estos dos saltos



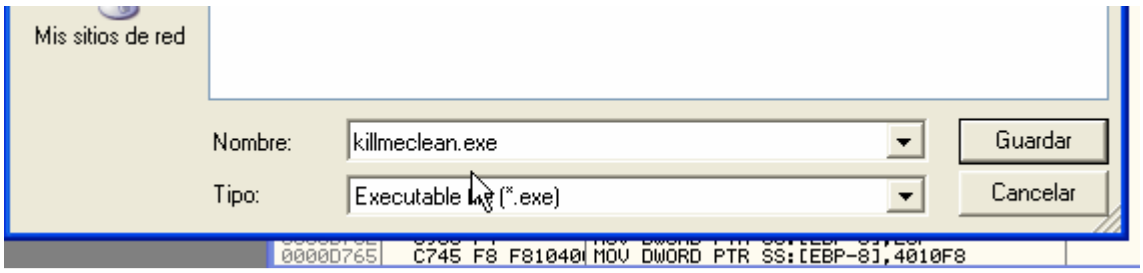
Hago click derecho COPY TO EXECUTABLE – ALL MODIFICATIONS y guardara todo lo que cambie.



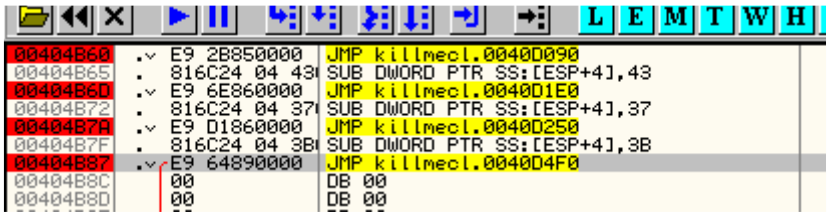
Apreto COPY ALL



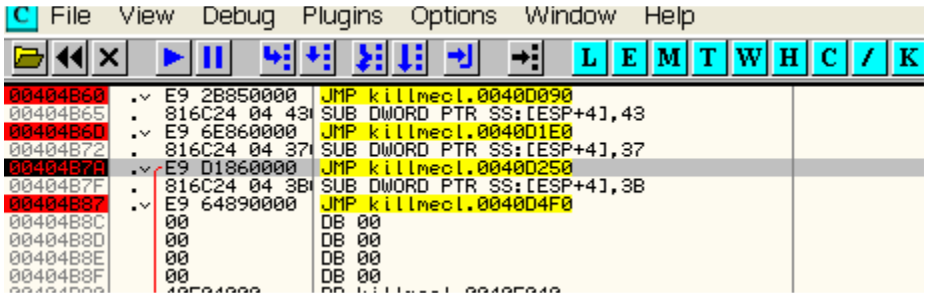
Ahora click derecho SAVE FILE



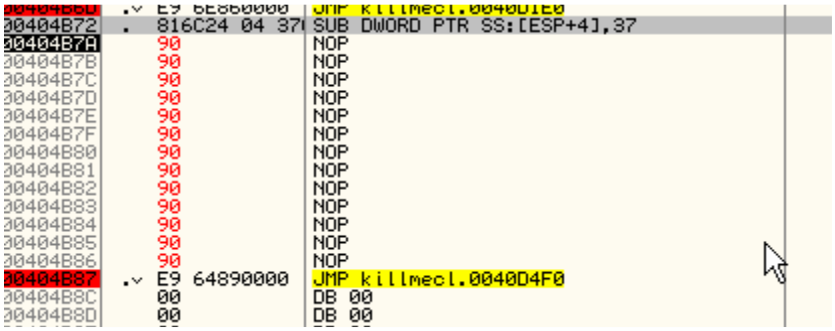
Bueno ya estoy mas cerca, ahora arranca la nag queda unos segundos y sola desaparece, así que pongo nuevamente los BPX en los JMPS



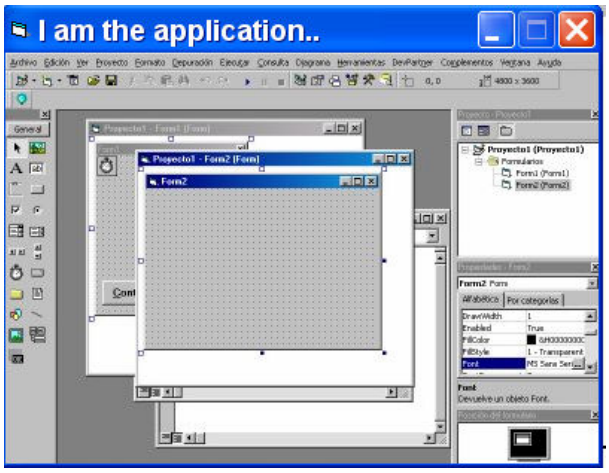
Doy RUN



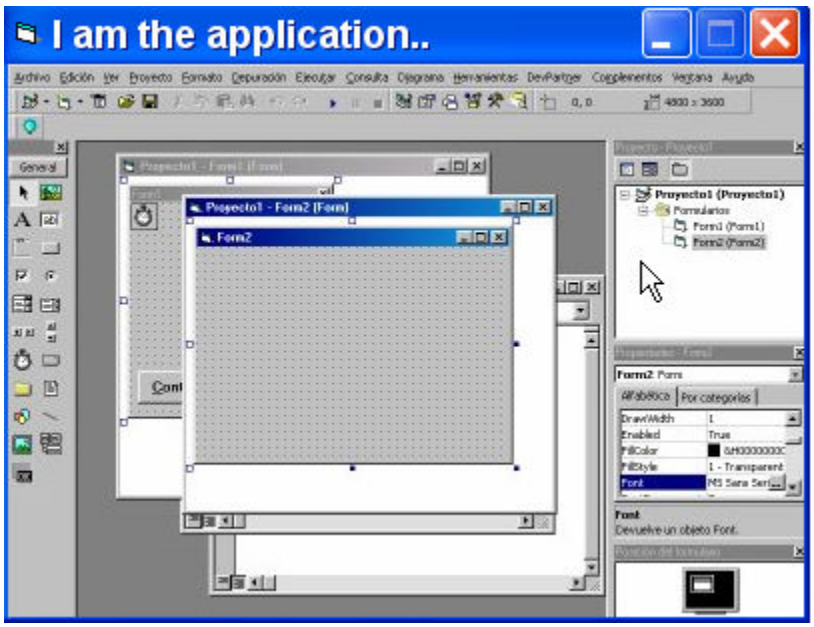
Veo que el primero que para es el que crea la nag el de 404b7a, y recuerdo que luego de pasar unos segundos de que sale la nag, para en el de 404b87 y de alli, ya arranca el programa directo gracias a que lo parchee, así que cambio el salto que crea la nag, por el que maneja de la rutina parcheada y va al inicio del programa.



Nopeando el salto que crea la nag, va directo al otro que maneja el timer y arranca el programa, veamos probemos con f9.



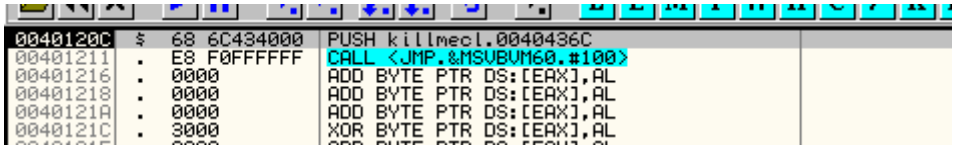
Guardemos los cambios y probemoslo



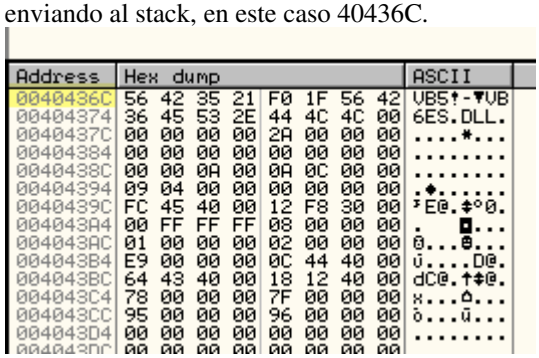
Allí corre limpio y ni se ve la nag, quedo limpio como culo de bebe, jeje.y a mano je.

El método del 4c es un método mecánico, muy útil para hacerlo rápido y sencillo y se basa en conocer como esta compuesto un archivo en VISUAL BASIC.

Lo usaremos vemos el entry point



Vemos que los programas en VB empiezan por un PUSH y un CALL (si no encontramos esta estructura, pues el exe ha sido modificado, pues necesitamos encontrar el PUSH y anotar la dirección que esta enviando al stack, en este caso 40436C.

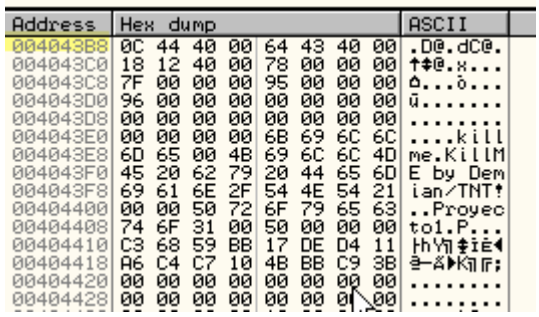


Que como vemos es el header de Visual Basic

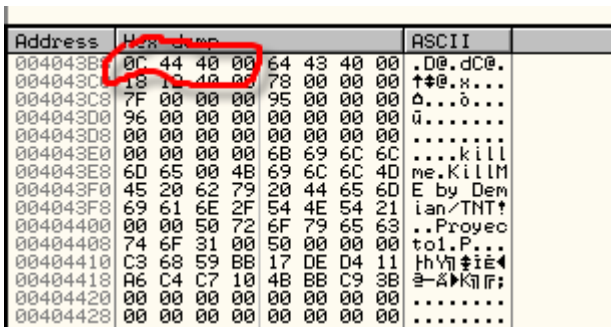
A esa dirección hay que sumarle 4C o sea

40436c + 4c

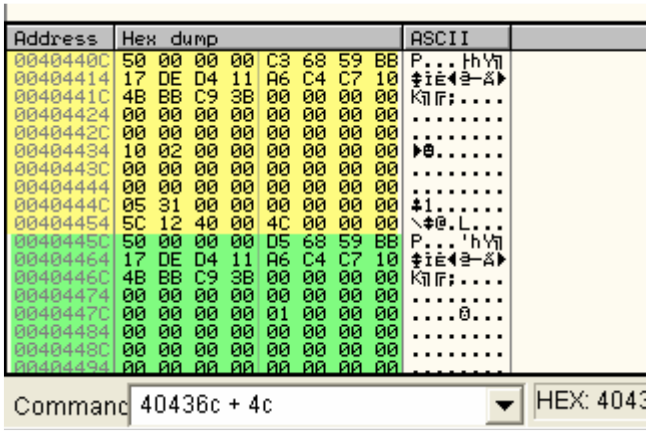
Es 4043b8



Allí esta, ahora buscamos la dirección que encontramos allí en 4043b8 que es 4044c0

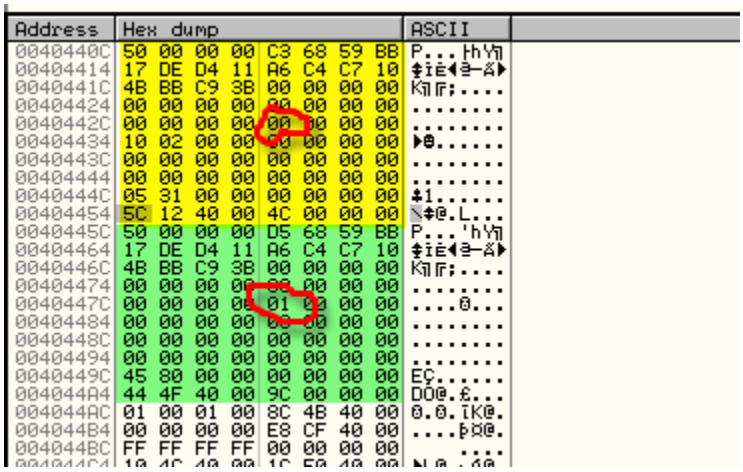


Busquemos dicha direccion en el dump

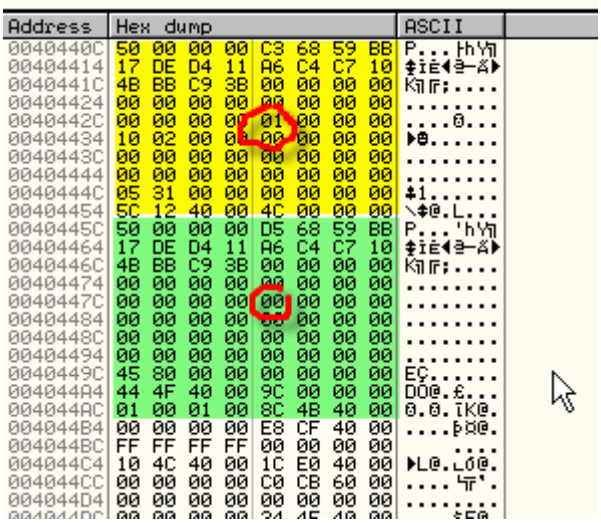


Vemos entradas de 50 hexa de largo, cada una corresponde a un FORM, en cada una de esas partes el byte 24 nos muestra el orden que tienen las forms para aparecer, veamos

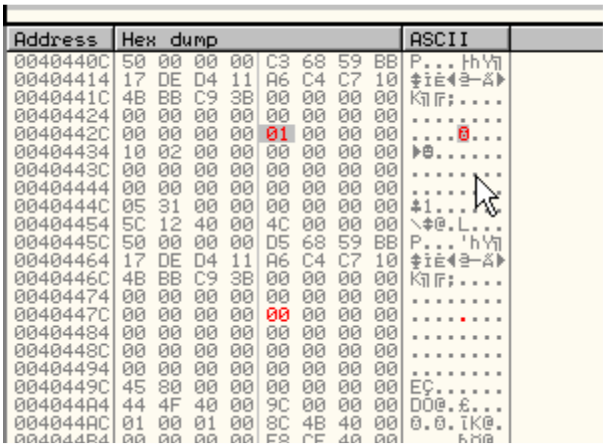
40440c +24=404430



Allí esta el 00 significa que esa form es la primera que aparecerá y el 01 significa que es la segunda que aparecerá, así que podemos alterar el orden.



Ahí esta ahora lo primero que aparecerá será el programa jeje y la nag segundo o nunca ya que al cerrar el programa ya se cierra y no sale una segunda form.



Address	Hex dump	ASCII
0040440C	50 00 00 00 C3 68 59 BB	P...├hY╗
00404414	17 DE D4 11 A6 C4 C7 10	↨ÞÔ◄ª─Ç►
0040441C	4B BB C9 3B 00 00 00 00	K╗╔;....
00404424	00 00 00 00 00 00 00 00	....
0040442C	00 00 00 00 01 00 00 00	....☺...
00404434	10 02 00 00 00 00 00 00	►☻......
0040443C	00 00 00 00 00 00 00 00	....
00404444	00 00 00 00 00 00 00 00	....
0040444C	05 31 00 00 00 00 00 00	♣1......
00404454	5C 12 40 00 4C 00 00 00	\↕@.L...
0040445C	50 00 00 00 D5 68 59 BB	P...╒hY╗
00404464	17 DE D4 11 A6 C4 C7 10	↨ÞÔ◄ª─Ç►
0040446C	4B BB C9 3B 00 00 00 00	K╗╔;....
00404474	00 00 00 00 00 00 00 00	....
0040447C	00 00 00 00 00 00 00 00	....
00404484	00 00 00 00 00 00 00 00	....
0040448C	00 00 00 00 00 00 00 00	....
00404494	00 00 00 00 00 00 00 00	....
0040449C	45 80 00 00 00 00 00 00	EÇ......
004044A4	44 4F 40 00 9C 00 00 00	DO@.£...
004044AC	01 00 01 00 8C 4B 40 00	☺.☺.îK@.
004044B4	00 00 00 00 F8 CF 40 00	....°Ï@.

Listo solucionado con el método 4c, ya saben las dos formas, por supuesto esta segunda es mas sencilla y rápida, pero es bueno siempre razonar y pelear un programa, por eso les mostré ambos métodos ya que no siempre habrá un método automático para cada caso y siempre estaremos nosotros razonando e intentado delante del programa.

Les dejare una tarea allí adjuntos hay dos crackmes uno sencillo y uno mas difícil, practiquen a ver que pueden hacer con ellos si pueden hallar los seriales, quitar las nags etc, en la próxima parte los solucionaremos

Hasta la parte 28
Ricardo Narvaja
17 de enero de 2006