

La primera tentación en cualquier programa con NAG es ver los JMPS que dirigen a diferentes partes del programa.

```

00401200 $ 68 6C434000 PUSH killme.0040436C
00401211 . E8 F0FFFFFF CALL <JMP.>MSUBUM60.#100>
00401216 . 0000 ADD BYTE PTR DS:[EAX],AL
00401218 . 0000 ADD BYTE PTR DS:[EAX],AL
0040121A . 0000 ADD BYTE PTR DS:[EAX],AL
0040121C . 3000 XOR BYTE PTR DS:[EAX],AL
0040121E . 0000 ADD BYTE PTR DS:[EAX],AL
00401220 . 40 INC EAX
00401221 . 0000 ADD BYTE PTR DS:[EAX],AL
00401223 . 0000 ADD BYTE PTR DS:[EAX],AL
00401225 . 0000 ADD BYTE PTR DS:[EAX],AL

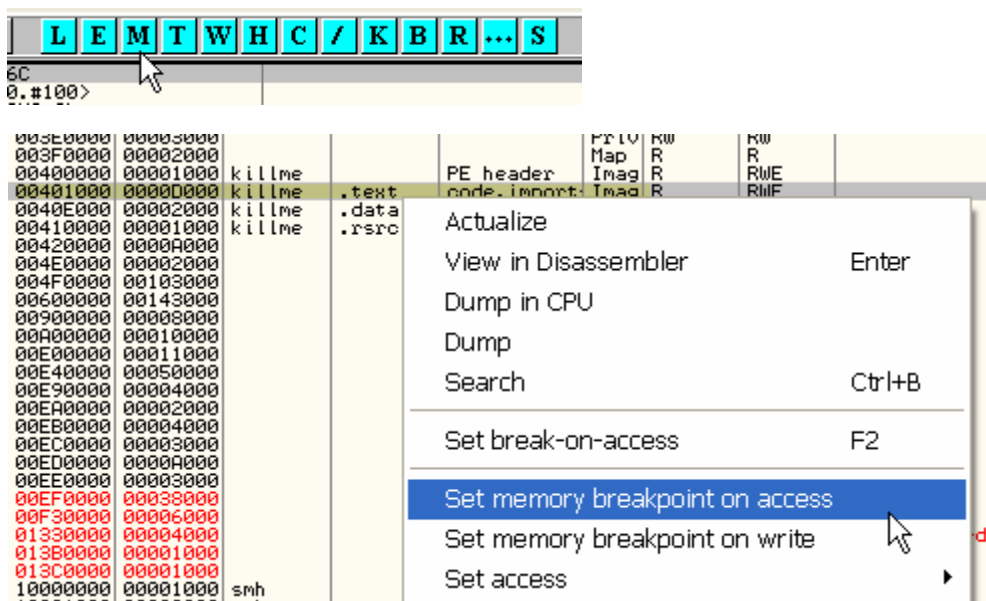
```

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

00404B58	816C24 04 3F00	SUB DWORD PTR SS:[ESP+4],3F
00404B60	✓ E9 2B850000	JMP killme3.0040D090
00404B65	816C24 04 4300	SUB DWORD PTR SS:[ESP+4],43
00404B6D	✓ E9 6E860000	JMP killme3.0040D1E0
00404B72	816C24 04 3700	SUB DWORD PTR SS:[ESP+4],37
00404B7A	✓ E9 D1860000	JMP killme3.0040D250
00404B7F	816C24 04 3B00	SUB DWORD PTR SS:[ESP+4],3B
00404B87	✓ E9 64890000	JMP killme3.0040D4F0
00404B8C	0000	ADD BYTE PTR DS:[EAX],AL
00404B8E	0000	ADD BYTE PTR DS:[EAX],AL
00404B90	40	INC EAX
00404B91	✓ E0 40	LOOPNE SHORT killme3.00404BD3
00404B93	0028	ADD BYTE PTR DS:[EAX],CH

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.

00404B58	816C24 04 3F00	SUB DWORD PTR SS:[ESP+4],3F
00404B60	✓ E9 2B850000	JMP killme3.0040D090
00404B65	816C24 04 4300	SUB DWORD PTR SS:[ESP+4],43
00404B6D	✓ E9 6E860000	JMP killme3.0040D1E0
00404B72	816C24 04 3700	SUB DWORD PTR SS:[ESP+4],37
00404B7A	✓ E9 D1860000	JMP killme3.0040D250
00404B7F	816C24 04 3B00	SUB DWORD PTR SS:[ESP+4],3B
00404B87	✓ E9 64890000	JMP killme3.0040D4F0
00404B8C	0000	ADD BYTE PTR DS:[EAX],AL
00404B8E	0000	ADD BYTE PTR DS:[EAX],AL
00404B90	40	INC EAX

Reiniciemos el programa

00404B7A	✓ E9 D1860000	JMP killme3.0040D250
00404B7F	816C24 04 3B00	SUB DWORD PTR SS:[ESP+4],3B
00404B87	✓ E9 64890000	JMP killme3.0040D4F0
00404B8C	0000	ADD BYTE PTR DS:[EAX],AL
00404B8E	0000	ADD BYTE PTR DS:[EAX],AL
00404B90	40	INC EAX
00404B91	✓ E0 40	LOOPNE SHORT killme3.00404BD3
00404B93	0028	ADD BYTE PTR DS:[EAX],CH

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.

00404B7A	✓ E9 D1860000	JMP killme3.0040D250
00404B7F	816C24 04 3B00	SUB DWORD PTR SS:[ESP+4],3B
00404B87	✓ E9 64890000	JMP killme3.0040D4F0
00404B8C	0000	ADD BYTE PTR DS:[EAX],AL
00404B8E	0000	ADD BYTE PTR DS:[EAX],AL
00404B90	40	INC EAX
00404B91	✓ E0 40	LOOPNE SHORT killme3.00404BD3

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

00404B7A	✓ E9 D1860000	JMP killme3.0040D250
00404B7F	816C24 04 3B00	SUB DWORD PTR SS:[ESP+4],3B
00404B87	✓ E9 64890000	JMP killme3.0040D4F0
00404B8C	0000	ADD BYTE PTR DS:[EAX],AL
00404B8E	0000	ADD BYTE PTR DS:[EAX],AL
00404B90	40	INC EAX
00404B91	✓ E0 40	LOOPNE SHORT killme3.00404BD3
00404B93	0028	ADD BYTE PTR DS:[EAX],CH

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.

Address	Disassembly	Comment
004004F0	55	PUSH EBP
004004F1	8BEC	MOV EBP,ESP
004004F3	83EC 0C	SUB ESP,0C
004004F6	68 16114000	PUSH <JMP.&MSUBUM60.__vbaExceptionHandler>
004004FB	64:A1 00000000	MOV EAX,DWORD PTR FS:[0]
00400501	50	PUSH EAX
00400502	64:8925 00000000	MOV DWORD PTR FS:[0],ESP
00400509	83EC 2C	SUB ESP,2C
0040050C	53	PUSH EBX
0040050D	56	PUSH ESI
0040050E	57	PUSH EDI
0040050F	8965 F4	MOV DWORD PTR SS:[EBP-C],ESP
00400512	C745 F8 E81040	MOV DWORD PTR SS:[EBP-8],killme3.00401040
00400519	8B75 08	MOV ESI,DWORD PTR SS:[EBP+8]
0040051C	8BC6	MOV EAX,ESI
0040051E	83E0 01	AND EAX,1
00400521	8945 FC	MOV DWORD PTR SS:[EBP-4],EAX
00400524	83E6 FE	AND ESI,FFFFFFFE
00400527	56	PUSH ESI
00400528	8975 08	MOV DWORD PTR SS:[EBP+8],ESI
0040052B	8B0E	MOV ECX,DWORD PTR DS:[ESI]
0040052D	FF51 04	CALL DWORD PTR DS:[ECX+4]
00400530	8B16	MOV EDX,DWORD PTR DS:[ESI]
00400532	33C0	XOR EAX,EAX
00400534	56	PUSH ESI
00400535	8945 E8	MOV DWORD PTR SS:[EBP-18],EAX
00400538	8945 E4	MOV DWORD PTR SS:[EBP-1C],EAX
0040053B	8945 E0	MOV DWORD PTR SS:[EBP-20],EAX
0040053E	8945 DC	MOV DWORD PTR SS:[EBP-24],EAX
00400541	FF92 08030000	CALL DWORD PTR DS:[EDX+308]
00400547	8B1D 20104000	MOV EBX,DWORD PTR DS:[<&MSUBUM60.__vbaObjSet
0040054D	50	PUSH EAX
0040054E	8D45 DC	LEA EAX,DWORD PTR SS:[EBP-24]
00400551	50	PUSH EAX
00400552	FFD3	CALL EBX
00400554	8B0E	MOV ECX,DWORD PTR DS:[ESI]
00400556	56	PUSH ESI
00400557	8945 D0	MOV DWORD PTR SS:[EBP-30],EAX
0040055A	FF91 08030000	CALL DWORD PTR DS:[ECX+308]
00400560	8D55 E0	LEA EDX,DWORD PTR SS:[EBP-20]
00400563	50	PUSH EAX
00400564	52	PUSH EDX
00400565	FFD3	CALL EBX
00400567	8BF8	MOV EDI,EAX
00400569	8D4D E8	LEA ECX,DWORD PTR SS:[EBP-18]
0040056C	51	PUSH ECX
0040056D	57	PUSH EDI

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

00400634	51	PUSH EAX
00400635	FF15 68104000	CALL DWORD PTR DS:[<&MSUBUM60.__vbaR8Str
00400638	DC1D D8104000	FCOMP QWORD PTR DS:[4010D8]
00400641	DFE0	FSTSW AX
00400643	F6C4 40	TEST AH,40
00400646	74 07	JE SHORT killme3.0040D64F
00400648	B8 01000000	MOV EAX,1
0040064D	EB 02	JMP SHORT killme3.0040D651
0040064F	33C0	XOR EAX,EAX
00400651	F7D8	NEG EAX

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 dará cuenta.

00400641	DFE0	FSTSW AX
00400643	F6C4 40	TEST AH,40
00400646	74 07	JE SHORT killme3.0040D64F
00400648	B8 01000000	MOV EAX,1
0040064D	EB 02	JMP SHORT killme3.0040D651
0040064F	33C0	XOR EAX,EAX

Veó 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.

00400638	DC1D D8104000	FCOMP QWORD PTR DS:[4010D8]
00400641	DFE0	FSTSW AX
00400643	F6C4 40	TEST AH,40
00400646	90	NOP
00400647	90	NOP
00400648	B8 01000000	MOV EAX,1
0040064D	EB 02	JMP SHORT killme3.0040D651
0040064F	33C0	XOR EAX,EAX
00400651	F7D8	NEG EAX
00400653	8D4D E8	LEA ECX,DWORD PTR SS:[EBP-18]

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

0040063B	DC1D 08104000	FCOMP QWORD PTR DS:[4010D8]	
00400641	DFF0	FSTSW AX	
00400643	F6C4 40	TEST AH,40	
00400646	90	NOP	
00400647	90	NOP	
00400648	B8 01000000	MOV EAX,1	
0040064D	EB 02	JMP SHORT killme3.00400651	

0040063B	FF15 68104000	CALL QWORD PTR DS:[68104000]	ITS
0040063B	DC1D 08104000	FCOMP QWORD PTR DS:[4010D8]	
00400641	DFF0	FSTSW AX	
00400643	F6C4 40	TEST AH,40	
00400646	90	NOP	
00400647	90	NOP	
00400648	B8 01000000	MOV EAX,1	
0040064D	EB 02	JMP SHORT killme3.00400651	

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

0040063B	DC1D 08104000	FCOMP QWORD PTR DS:[4010D8]	
00400641	DFF0	FSTSW AX	
00400643	F6C4 40	TEST AH,40	
00400646	90	NOP	
00400647	90	NOP	
00400648	B8 01000000	MOV EAX,1	
0040064D	FR 02	JMP SHORT killme3.00400651	

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

0040D707	6A 02	PUSH 2	
0040D709	FF15 10104000	CALL DWORD PTR DS:[<&MSUBUM60.__vbaFree	MSUBUM60
0040D70F	83C4 18	ADD ESP,18	
0040D712	C3	RETN	
0040D713	C3	RETN	
0040D714	8B45 08	MOV EAX,DWORD PTR SS:[EBP+8]	
0040D717	50	PUSH EAX	
0040D718	8B08	MOV ECX,DWORD PTR DS:[EAX]	
0040D71A	FF51 08	CALL DWORD PTR DS:[ECX+8]	
0040D71D	8B45 FC	MOV EAX,DWORD PTR SS:[EBP-4]	
0040D720	8B4D EC	MOV ECX,DWORD PTR SS:[EBP-14]	
0040D723	5F	POP EDI	
0040D724	5E	POP ESI	
0040D725	64:890D 00000000	MOV DWORD PTR FS:[0],ECX	
0040D72C	5B	POP EBX	
Return to 0040D714 (killme3.0040D714)			
Address Hex dump ASCII			

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

Sigamos traceando

0040D720	5B	POP EBX	
0040D72D	8BE5	MOV ESP,EBP	
0040D72F	5D	POP EBP	
0040D730	C2 0400	RETN 4	
0040D733	^ E9 E439FFFF	JMP <JMP.&MSUBUM60.__vbaFPException>	
0040D738	90	NOP	
0040D739	90	NOP	
0040D73A	90	NOP	
0040D73B	90	NOP	
0040D73C	90	NOP	
0040D73D	90	NOP	
0040D73E	90	NOP	
0040D73F	90	NOP	
0040D740	55	PUSH EBP	
0040D741	8BEC	MOV EBP,ESP	
0040D743	83EC 0C	SUB ESP,0C	
0040D746	68 16114000	PUSH <JMP.&MSUBUM60.__vbaExceptionHandler>	
0040D74B	64:A1 00000000	MOV EAX,DWORD PTR FS:[0]	
0040D751	50	PUSH EAX	
Return to 66051FB3 (MSUBUM60.66051FB3)			
Address Hex dump ASCII			
0040E000	00 00 00 00 00 00 00 00	.....	001

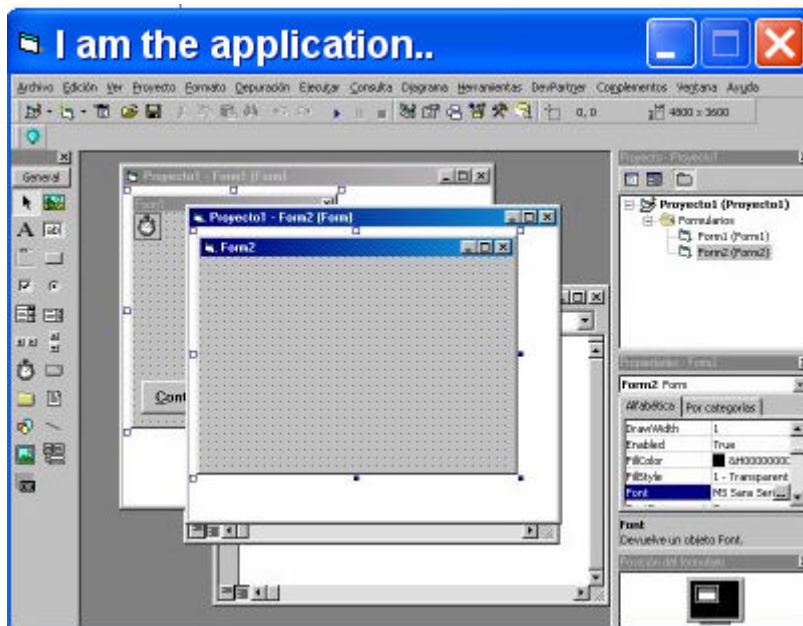
Allí si llega al RETN que vemos en la aclaración que volverá al dll de visual, aquí ya esta todo inicializado para arrancar el programa, así que puedo intentar cambiar ese RETN 4 por un salto al inicio del programa veamos.

0040D720	5B	POP EBX	
0040D72D	8BE5	MOV ESP,EBP	
0040D72F	5D	POP EBP	
0040D730	EB 07	JMP SHORT killme3.0040D739	
0040D732	90	NOP	
0040D733	^ E9 E439FFFF	JMP <JMP.&MSUBUM60.__vbaFPException>	
0040D738	90	NOP	
0040D739	90	NOP	
0040D73A	90	NOP	
0040D73B	90	NOP	
0040D73C	90	NOP	
0040D73D	90	NOP	
0040D73E	90	NOP	
0040D73F	90	NOP	
0040D740	55	PUSH EBP	

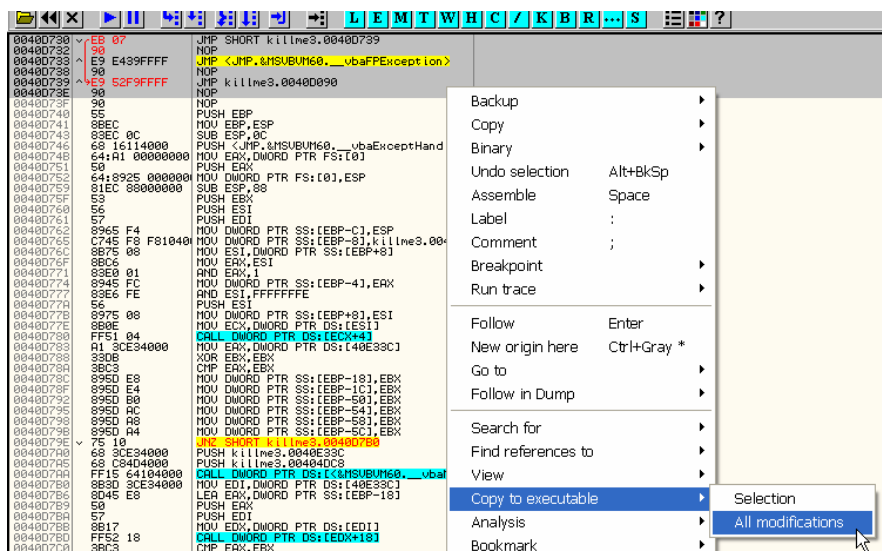
Salto allí que hay mas lugar para escribir y luego salto a 40d090 que era el inicio del programa.

File View Debug Plugins Options Window Help			
[Icons] [L] [E] [M] [T] [W] [H] [C]			
0040D730	EB 07	JMP SHORT killme3.0040D739	
0040D732	90	NOP	
0040D733	^ E9 E439FFFF	JMP <JMP.&MSUBUM60.__vbaFPException>	
0040D738	90	NOP	
0040D739	^ E9 52F9FFFF	JMP killme3.0040D090	
0040D73A	90	NOP	
0040D73B	90	NOP	
0040D73C	90	NOP	
0040D73D	90	NOP	
0040D73E	90	NOP	
0040D73F	90	NOP	
0040D740	55	PUSH EBP	
0040D741	8BEC	MOV EBP,ESP	

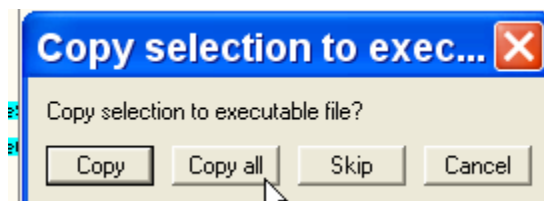
Probemos si arranca demos RUN



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

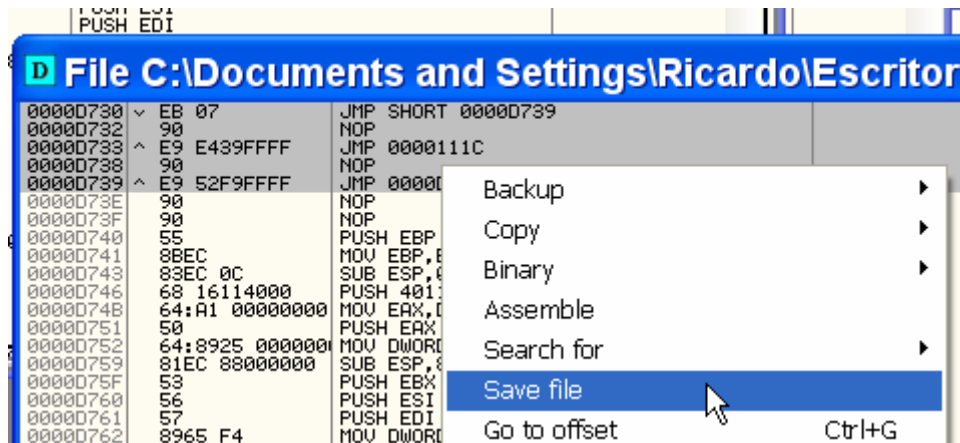


Hago click derecho COPY TO EXECUTABLE – ALL MODIFICATIONS y guardara todo lo que cambie.

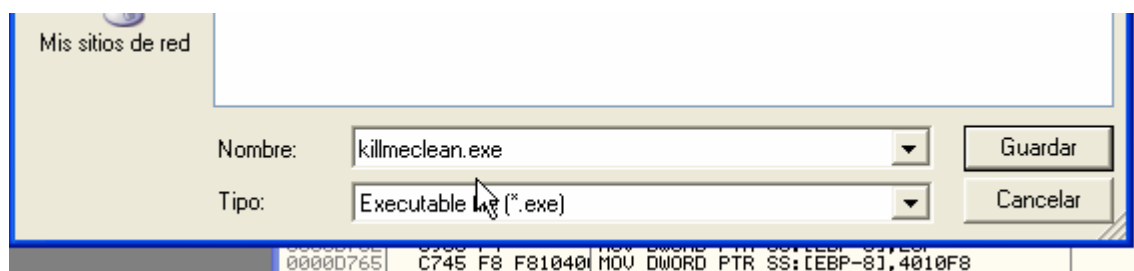


Apreto COPY ALL

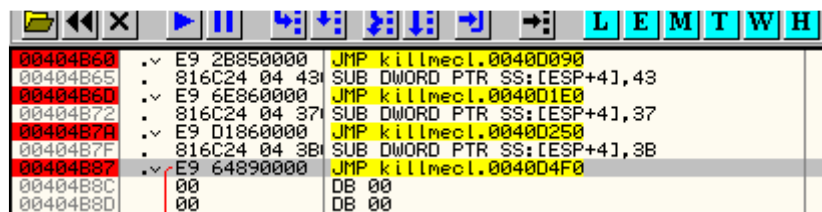




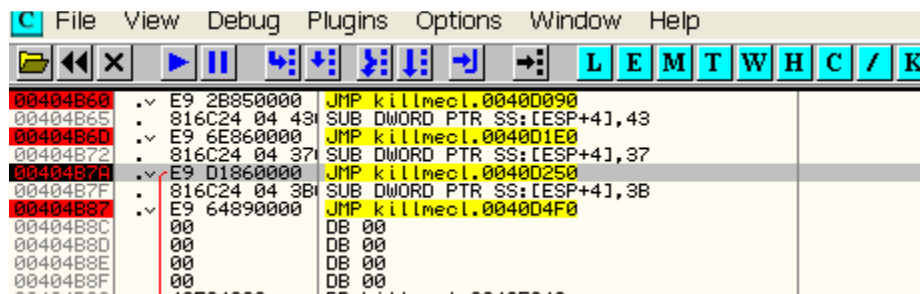
Ahora click derecho SAVE FILE



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

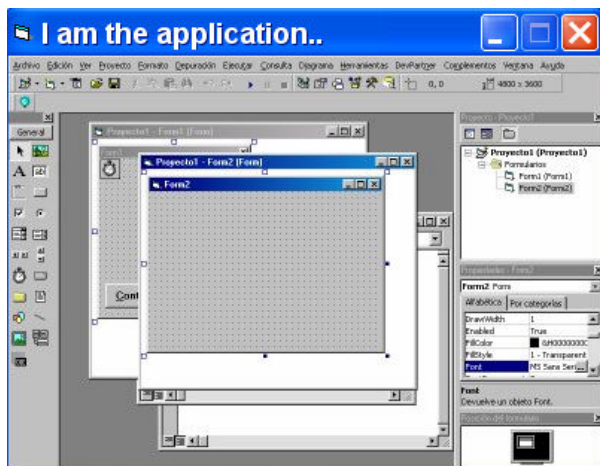


Veó que el primero que para es el que crea la nag el de 404b7a, y recuerdo que luego de pasar unos segundos de que sale la nag, para en el de 404b87 y de allí, ya arranca el programa directo gracias a que lo parchee, así que cambio el salto que crea la nag, por el que maneja de la rutina parcheada y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

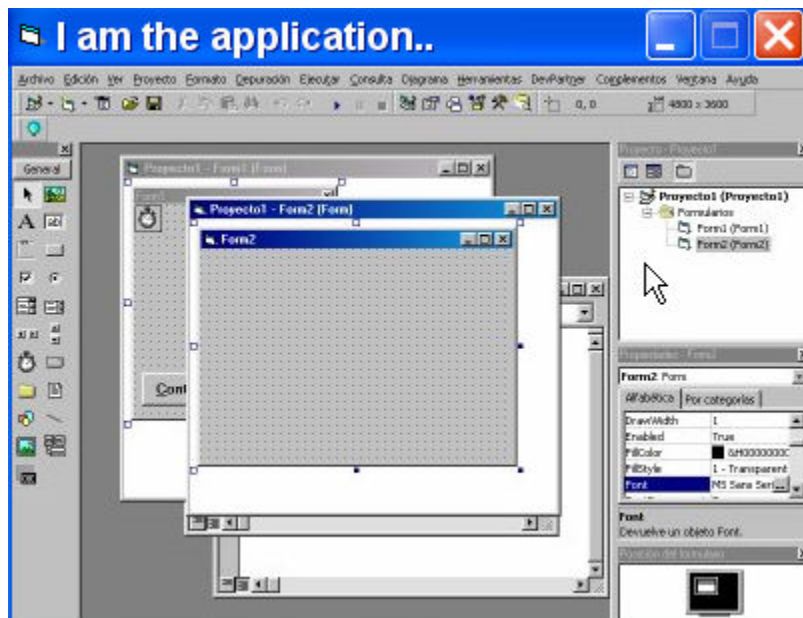


00404B72	816C24 04 37	SUB DWORD PTR SS:[ESP+41],37
00404B7A	90	NOP
00404B7B	90	NOP
00404B7C	90	NOP
00404B7D	90	NOP
00404B7E	90	NOP
00404B7F	90	NOP
00404B80	90	NOP
00404B81	90	NOP
00404B82	90	NOP
00404B83	90	NOP
00404B84	90	NOP
00404B85	90	NOP
00404B86	90	NOP
00404B87	E9 64890000	JMP killmecl.0040D4F0
00404B8C	00	DB 00
00404B8D	00	DB 00

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

0040120C	\$	68 6C434000	PUSH killmecl.0040436C
00401211	.	E8 F0FFFFFF	CALL <JMP.&MSUBUM60.#100>
00401216	.	0000	ADD BYTE PTR DS:[EAX],AL
00401218	.	0000	ADD BYTE PTR DS:[EAX],AL
0040121A	.	0000	ADD BYTE PTR DS:[EAX],AL
0040121C	.	0000	XOR BYTE PTR DS:[EAX],AL

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.

Address	Hex dump	ASCII
0040436C	56 42 35 21 F0 1F 56 42	VB5!-!UB
00404374	36 45 53 2E 44 4C 4C 00	6ES.DLL.
0040437C	00 00 00 00 2A 00 00 00	....*...
00404384	00 00 00 00 00 00 00 00	.....
0040438C	00 00 0A 00 0A 0C 00 00	.....
00404394	09 04 00 00 00 00 00 00	.....
0040439C	FC 45 40 00 12 F8 30 00	!E0.00.
004043A4	00 FF FF FF 08 00 00 00	.....
004043AC	01 00 00 00 02 00 00 00	0...0...
004043B4	E9 00 00 00 0C 44 40 00	ü...D0.
004043BC	64 43 40 00 18 12 40 00	dC0.00.
004043C4	78 00 00 00 7F 00 00 00	h...0...
004043CC	95 00 00 00 96 00 00 00	o...0...
004043D4	00 00 00 00 00 00 00 00	.....

Que como vemos es el header de Visual Basic

A esa dirección hay que sumarle 4C o sea

40436c + 4c

0040E050	00 00 00 00 00 00 00 00	.....
0040E060	00 00 00 00 00 00 00 00	.....
0040E070	00 00 00 00 00 00 00 00	.....
Command	40436c + 4c	HEX: 4043B8 -
Program entry point		

Es 4043b8

Address	Hex dump	ASCII
004043B8	0C 44 40 00 64 43 40 00	.D0.dC0.
004043C0	18 12 40 00 78 00 00 00	00.00.00
004043C8	7F 00 00 00 95 00 00 00	0...0...
004043D0	96 00 00 00 00 00 00 00	ü.....
004043D8	00 00 00 00 00 00 00 00	.....
004043E0	00 00 00 00 6B 69 6C 6C	....kill
004043E8	6D 65 00 4B 69 6C 6C 4D	me.KillM
004043F0	45 20 62 79 20 44 65 6D	E by Dem
004043F8	69 61 6E 2F 54 4E 54 21	ian/TNT!
00404400	00 00 50 72 6F 79 65 63	..Proyec
00404408	74 6F 31 00 50 00 00 00	to1.P...
00404410	C3 68 59 BB 17 DE 04 11	hV000iE4
00404418	A6 C4 C7 10 4B BB C9 3B	0-A0K0F;
00404420	00 00 00 00 00 00 00 00	.....
00404428	00 00 00 00 00 00 00 00	.....

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

Address	Hex dump	ASCII
004043B8	0C 44 40 00 64 43 40 00	.D0.dC0.
004043C0	18 12 40 00 78 00 00 00	00.00.00
004043C8	7F 00 00 00 95 00 00 00	0...0...
004043D0	96 00 00 00 00 00 00 00	ü.....
004043D8	00 00 00 00 00 00 00 00	.....
004043E0	00 00 00 00 6B 69 6C 6C	....kill
004043E8	6D 65 00 4B 69 6C 6C 4D	me.KillM
004043F0	45 20 62 79 20 44 65 6D	E by Dem
004043F8	69 61 6E 2F 54 4E 54 21	ian/TNT!
00404400	00 00 50 72 6F 79 65 63	..Proyec
00404408	74 6F 31 00 50 00 00 00	to1.P...
00404410	C3 68 59 BB 17 DE 04 11	hV000iE4
00404418	A6 C4 C7 10 4B BB C9 3B	0-A0K0F;
00404420	00 00 00 00 00 00 00 00	.....
00404428	00 00 00 00 00 00 00 00	.....

Busquemos dicha dirección en el dump

Address	Hex dump	ASCII
0040440C	50 00 00 00 C3 68 59 BB	P...hYŋ
00404414	17 DE D4 11 A6 C4 C7 10	ŋiē←ā→
0040441C	4B BB C9 3B 00 00 00 00	Kŋf;....
00404424	00 00 00 00 00 00 00 00	.....
0040442C	00 00 00 00 00 00 00 00	.....
00404434	10 02 00 00 00 00 00 00	0.....
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	\#0.L...
0040445C	50 00 00 00 05 68 59 BB	P...hYŋ
00404464	17 DE D4 11 A6 C4 C7 10	ŋiē←ā→
0040446C	4B BB C9 3B 00 00 00 00	Kŋf;....
00404474	00 00 00 00 00 00 00 00	.....
0040447C	00 00 00 00 01 00 00 00	...0....
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	.....

Command: 40436c + 4c      HEX: 4043

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

Address	Hex dump	ASCII
0040440C	50 00 00 00 C3 68 59 BB	P...hYŋ
00404414	17 DE D4 11 A6 C4 C7 10	ŋiē←ā→
0040441C	4B BB C9 3B 00 00 00 00	Kŋf;....
00404424	00 00 00 00 00 00 00 00	.....
0040442C	00 00 00 00 00 00 00 00	.....
00404434	10 02 00 00 00 00 00 00	0.....
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	\#0.L...
0040445C	50 00 00 00 05 68 59 BB	P...hYŋ
00404464	17 DE D4 11 A6 C4 C7 10	ŋiē←ā→
0040446C	4B BB C9 3B 00 00 00 00	Kŋf;....
00404474	00 00 00 00 00 00 00 00	.....
0040447C	00 00 00 00 01 00 00 00	...0....
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	E0.....
004044A4	44 4F 40 00 9C 00 00 00	D00.0...
004044AC	01 00 01 00 8C 4B 40 00	0.0.1k0.
004044B4	00 00 00 00 E8 CF 40 00	...0000.
004044BC	FF FF FF FF 00 00 00 00	.....
004044C4	10 4C 40 00 1C E0 40 00	0L0.L00.
004044CC	00 00 00 00 C0 CB 60 00	...000.
004044D4	00 00 00 00 00 00 00 00	.....
004044DC	00 00 00 00 00 00 00 00	.....

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.

Address	Hex dump	ASCII
0040440C	50 00 00 00 C3 68 59 BB	P...hYŋ
00404414	17 DE D4 11 A6 C4 C7 10	ŋiē←ā→
0040441C	4B BB C9 3B 00 00 00 00	Kŋf;....
00404424	00 00 00 00 00 00 00 00	.....
0040442C	00 00 00 00 01 00 00 00	...0....
00404434	10 02 00 00 00 00 00 00	0.....
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	\#0.L...
0040445C	50 00 00 00 05 68 59 BB	P...hYŋ
00404464	17 DE D4 11 A6 C4 C7 10	ŋiē←ā→
0040446C	4B BB C9 3B 00 00 00 00	Kŋf;....
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	E0.....
004044A4	44 4F 40 00 9C 00 00 00	D00.0...
004044AC	01 00 01 00 8C 4B 40 00	0.0.1k0.
004044B4	00 00 00 00 E8 CF 40 00	...0000.
004044BC	FF FF FF FF 00 00 00 00	.....
004044C4	10 4C 40 00 1C E0 40 00	0L0.L00.
004044CC	00 00 00 00 C0 CB 60 00	...000.
004044D4	00 00 00 00 00 00 00 00	.....
004044DC	00 00 00 00 00 00 00 00	.....

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...hVh
00404414	17 DE D4 11 A6 C4 C7 10	tié-À»
0040441C	4B BB C9 3B 00 00 00 00	Kf;....
00404424	00 00 00 00 00 00 00 00	.....
0040442C	00 00 00 00 01 00 00 00	...0...
00404434	10 02 00 00 00 00 00 00	0.....
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	\0.L...
0040445C	50 00 00 00 D5 68 59 BB	P...hVh
00404464	17 DE D4 11 A6 C4 C7 10	tié-À»
0040446C	4B BB C9 3B 00 00 00 00	Kf;....
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	E.....
004044A4	44 4F 40 00 9C 00 00 00	D00.f...
004044AC	01 00 01 00 8C 4B 40 00	0.0.iK0.
004044B4	00 00 00 00 F0 CF 40 00	0.0.fK0.

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