

# *CracksLatinos*

REGISTRAR Y DESEMPACAR

VISUAL PROTECT 3.5.4

by SNB



## 1- INTRODUCCION

Este packer es conocido y también antiguo, bueno lo que hace es crear una nag, empaquetar la aplicación a “proteger” y generar una licencia. Como este es mi primer tuto lo voy hacer lo mejor posible espero su comprensión y si hubiese errores espero que me los hagan llegar para corregirlos y seguir mejorando, entonces al ataque!

## 2- HERRAMIENTAS

- OLLYDBG 1.10
- OLLYDUMP plugins
- IMPORTREC 1.7C
- Visual Protect 3.5.4 -  
<http://ftp.cs.pu.edu.tw/Windows/Softking/soft/en/v/vp.exe>
- Process Monitor -  
<http://download.sysinternals.com/Files/ProcessMonitor.zip>

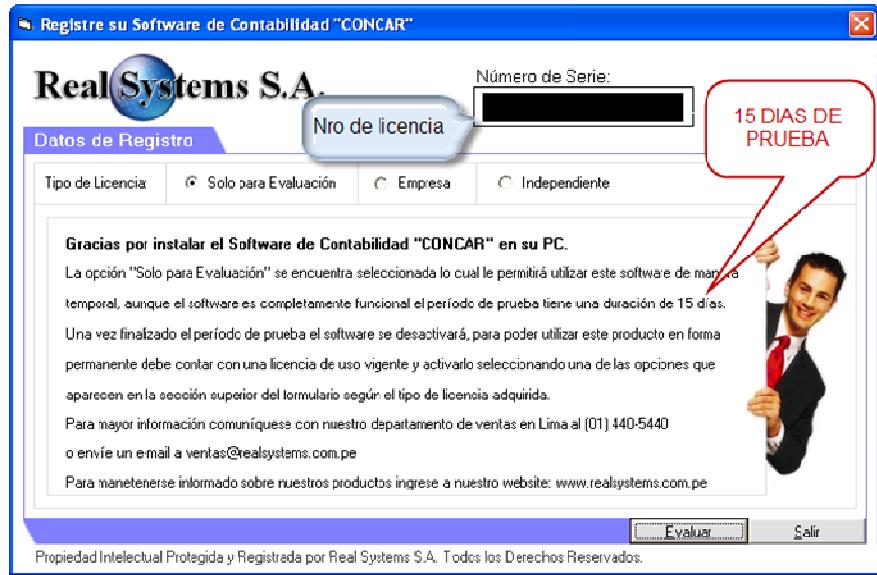
## 3- APPLICACION OBJETIVO

- CONTABILIDAD CONCAR v2009 - <http://www.realsystems.com.pe/>

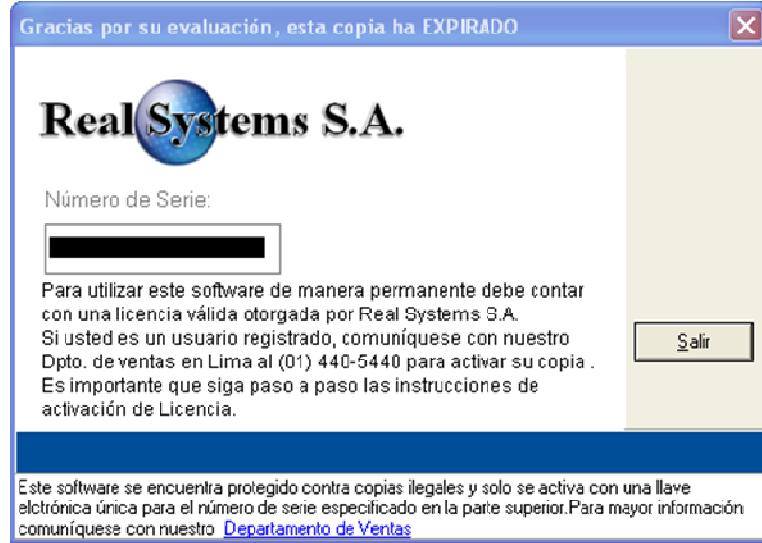
## 4- GENERAR LICENCIA (SOLUCION #01)

Una vez instalada la aplicación objetivo procedemos a realizar los reglajes respectivos:

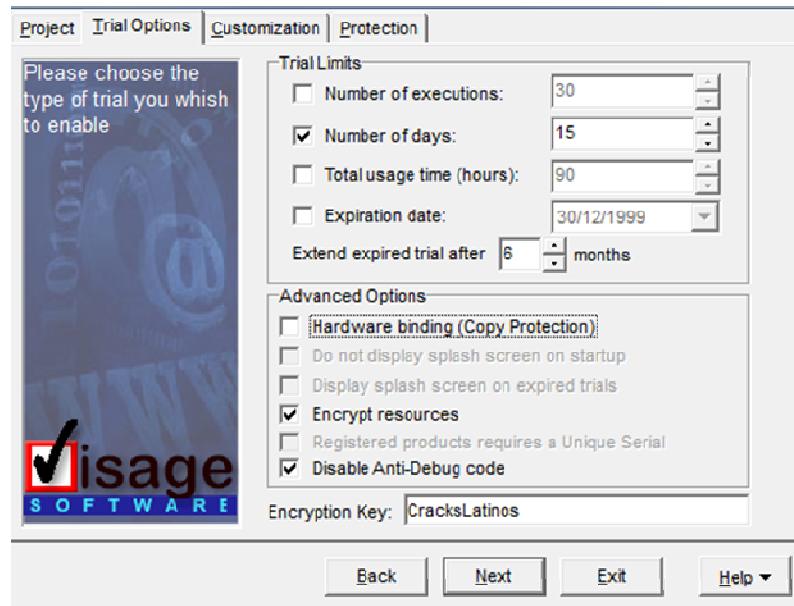
Al ejecutar la app nos muestra esto:



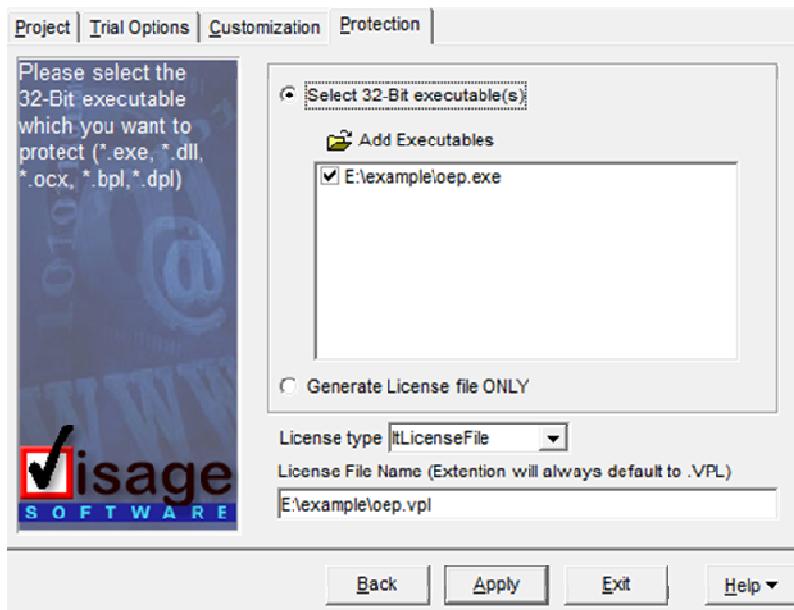
Nos dice que solo tenemos 15 días de prueba, así que corremos en el tiempo 1 mes y volvemos a ejecutar la app y nos muestra esto:



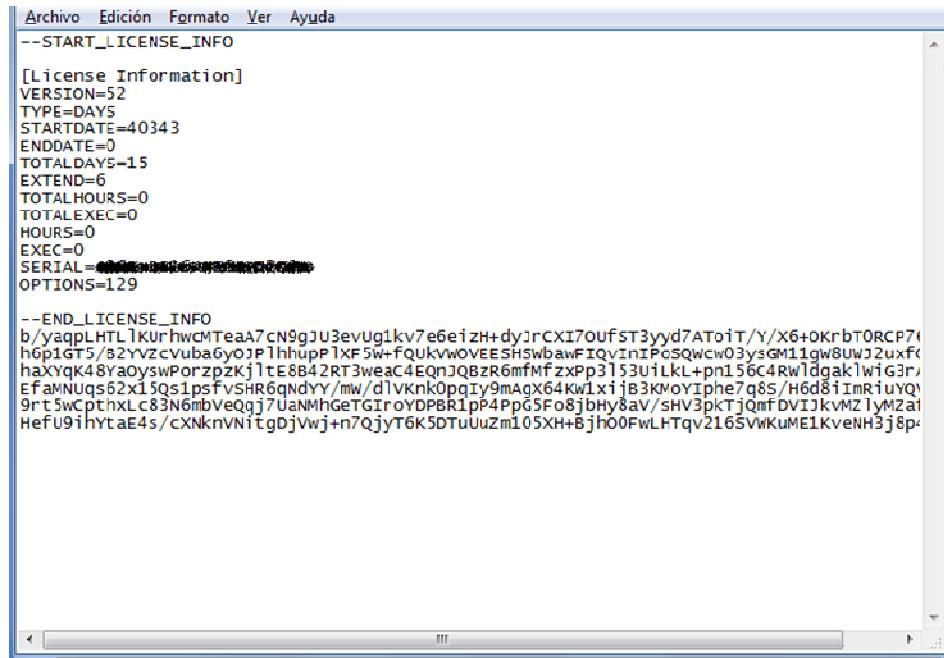
Evidentemente el packer hace su trabajo, ahora vamos a instalar y ver que hace el packer al empaquetar una aplicación para esto vamos a pasarse el packer a cualquier app que tengan, use el ope finder (oep.exe). Configure el visualprotect como se aprecia en la siguiente imagen.



Luego seleccione la app oep finder (oep.exe) y le damos en Apply.



Veo que nos genera 2 archivos vp.dll y oep.vpl, en vp.dll como es de suponerse esta el código que realiza el control de la licencia y en oep.vpl (licencia) esta los datos que pusimos al empaquetar oep.exe.

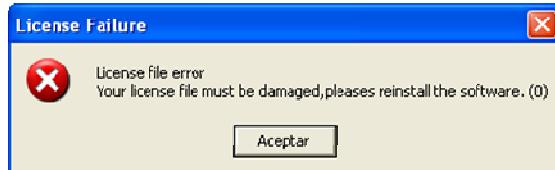


The screenshot shows a Notepad window with the following content:

```
Archivo Edición Formato Ver Ayuda
--START_LICENSE_INFO
[License Information]
VERSION=52
TYPE=DAYS
STARTDATE=40343
ENDDATE=0
TOTALDAYS=15
EXTEND=6
TOTALHOURS=0
TOTALEXEC=0
HOURS=0
EXEC=0
SERIAL=[REDACTED]
OPTIONS=129

--END_LICENSE_INFO
b/yaqlpHTL1KurhwcdTea7cN9gJU3evug1kv7e6eizH+dyJrcXI7OUfST3yyd7AToiT/Y/X6+OKrbT0RCP76
h6p1GT5/b2YvZcvuba6yoJP1hhupP1xF5w+fQukvwoVEESH5wbawF1qvInIPoSQwcw03y3gM11gW8UwJ2uxfc
haxYqk48YaOyswPorzpzkj1tE8B42T3weaC4EQnJQBzR6mfzxPp3153UiLkL+pn156c4RwldgaklwiG3r/
EfamNUqs62x15Qs1psfvSHR6QndYY/mw/d1vKnk0pqIy9maqx64Kwlxi1jB3KmoYIphe7q85/H6d8iImriuYQ/
9rt5wCpthxLc83N6mbVeQqj7uaNMGeTGIroYDPBR1pP4PpG5Fo8jbHy&aV/sHV3pkTjQmfDV1jkvMZlyM2a
HeFu9ihYtaE4s/cXMknvNitgdjvwj+n7QjyT6K5DTuUuZm105XH+Bjh0OFwLHTqv2165vwKuME1KveNH3j8p
```

Vemos que en oep.vpl está protegido por un algoritmo que verifica que los datos ahí no sean alterados sino fíjense lo que sale al intentar cambiar los 15 días (TOTALDAYS=15) de prueba por 20 días.



Me llama la atención el algoritmo y la key de encriptación que pusimos (CracksLatinos) al empaquetar el exe, entonces cargamos oep.exe empaquetado con el ollydbg. Para buscar nuestro key.

```

00400F90 55 PUSH EBP
00400F91 8BED MOV EBP,ESP
00400F93 51 PUSH ECX
00400F94 53 PUSH EBX
00400F95 56 PUSH EDI
00400F96 57 PUSH EDI
00400F97 C705 B0FF4000 00 MOV DWORD PTR DS:[40FE80],0
00400FA1 68 48E04000 PUSH oep.0040E048
00400FA6 FF15 00000000 CALL DWORD PTR DS:[40FF0C],ERX
00400FB0 A0CEFF4000 MOV DWORD PTR DS:[40FF0C],ERX
00400FB1 6A 55EE04000 PUSH oep.0040E058
00400FB6 A1 0CFF4000 MOV EXX,DWORD PTR DS:[40FF0C]
00400FB8 50 PUSH EXX
00400FBC F515 04C04000 CALL DWORD PTR DS:[40FE90],ERX
00400FC2 A3 90FF4000 MOV DWORD PTR DS:[40FE90],ERX
00400FC7 6A 00 PUSH 0
00400FC9 F515 90FE4000 CALL DWORD PTR DS:[40FE90]
00400FCF A3 ECFE4000 MOV DWORD PTR DS:[40FEEC],ERX
00400FD4 8800 ECFE4000 MOV ECX,DWORD PTR DS:[40FEEC]
00400FDA 51 PUSH ECX
00400FDB E8 C0050000 CALL oep.004095AB
00400FE0 83C4 04 ADD ESP,4
00400FE3 8945 FC MOV DWORD PTR SS:[EBP-4],ERX
00400F6 837D FC 00 CMP DWORD PTR SS:[EBP-4],0
00400FA A v74 08 JE SHORT oep.00408F8F
00400FEC FF65 FC JMP DWORD PTR SS:[EBP-4]
00400FEE 5F POP EDI
00400FF0 5E POP ESI
00400FF1 58 POP EBX
00400FF2 88E5 MOV ESP,E8P
00400FF4 50 POP EBP
00400FF5 C3 RETN
00400FF6 55 PUSH EBP
00400FF7 8BED MOV EBP,ESP
00400FF9 51 PUSH ECX
00400FFA 58 PUSH EBX
00400FFB 56 PUSH EST
00400FFC 57 PUSH EDI

```

Alt+M y colocamos un bp en:

Address	Size	Owner	Section	Contains	Type	Access
00200000	00041000				Map	R
00320000	00006000				Map	R
00330000	00041000				Map	R
00380000	00005000				Pri	RW
00390000	00003000				Map	R
003A0000	00001000				Pri	RW
003B0000	00001000				Pri	RW
003C0000	00002000				Pri	RW
003D0000	00002000				Map	R
003E0000	00001000				Pri	RWE
003F0000	00001000				Map	RW
00400000	00001000	oep		PE header	Image	R
00401000	00005000	oep	VDATA	code	Image	R
00406000	0000E000	oep	VCODE	SFX,data,im	Image	R
00414000	00003000	oep	.rsro	resources	Image	R
00420000	00001000	UP		PE header	Image	R
00421000	0000C000	UP	VDATA	code	Image	R
004E1000	00009000	UP	VCODE	SFX,data,im	Image	R
004EA000	00008000	UP	.rsro	resources	Image	R
00500000	00005000				Map	R E
005C0000	00002000				Map	R E
005D0000	00103000				Map	R
006E0000	00002000				Map	R E
009E0000	00001000				Pri	RW
00A60000	0005C000				Pri	RWE
00AC0000	00004000				Pri	RW
008C0000	00005000				Map	R
00C10000	00010000				Map	RW
00CE0000	00002000				Map	RW
SB150000	00001000	uxtheme		PE header	Image	R
SB151000	00030000	uxtheme	.text	code,import	Image	R
SB181000	00001000	uxtheme	.data	data	Image	R
SB182000	00001000	uxtheme	.code	code	Image	R

Luego presionamos F9 para correr la app y se detiene en:

```
0040AFE9 CC INT3
0040AFEA CC INT3
0040AFEB CC INT3
0040AFEC CC INT3
0040AFED CC INT3
0040AFEE CC INT3
0040AFEF CC INT3
0040AFF0 55 PUSH EBP
0040AFF1 8BEC MOV EBP,ESP
0040AFF2 67 PUSH EDI
0040AFF4 56 PUSH ESI
0040AFF5 53 PUSH EBX
0040AFF6 8B4D 10 MOV ECX,DWORD PTR SS:[EBP+10]
0040AFF7 E3 26 JCXZ SHORT oep.0040B021
0040AFF8 8B09 MOV EBX,ECX
0040AFF9 8BD7 08 MOV EDI,DWORD PTR SS:[EBP+8]
0040AFFA 8B77 MOV ESI,EDI
0040AFFB 8C00 XOR EAX,EAX
0040AFFC F2:AE RSPNE SCAS BYTE PTR ES:[EDI]
0040AFFD F7D9 NEG ECX
0040AFFE 03CB ADD ECX,EBX
0040AFFF 8BFE MOV EDI,ESI
0040B000 8B75 0C MOV ESI,DWORD PTR SS:[EBP+C]
0040B001 F3:A6 REPE CMPS BYTE PTR ES:[EDI],BYTE PTR DS
0040B002 8A46 FF MOV AL,BYTE PTR DS:[ESI-1]
0040B003 83C9 XOR ECX,ECX
0040B004 8C47 FF CMP AL,BYTE PTR DS:[EDI-1]
0040B005 77 04 JG SHORT oep.0040B01F
0040B006 77 04 JE SHORT oep.0040B021
0040B007 49 DEC ECX
0040B008 49 DEC ECX
0040B009 F7D1 NOT ECX
0040B010 8B01 MOV EBX,ECX
0040B011 8BC1 POP EBX
0040B012 8B02 POP ESI
0040B013 8B03 POP EDI
0040B014 C9 LEAVE
0040B015 C3 RETN
```

Ahora que todo está en memoria buscamos nuestra key (CracksLatinos) en el dump ☺.

## Nuestro key

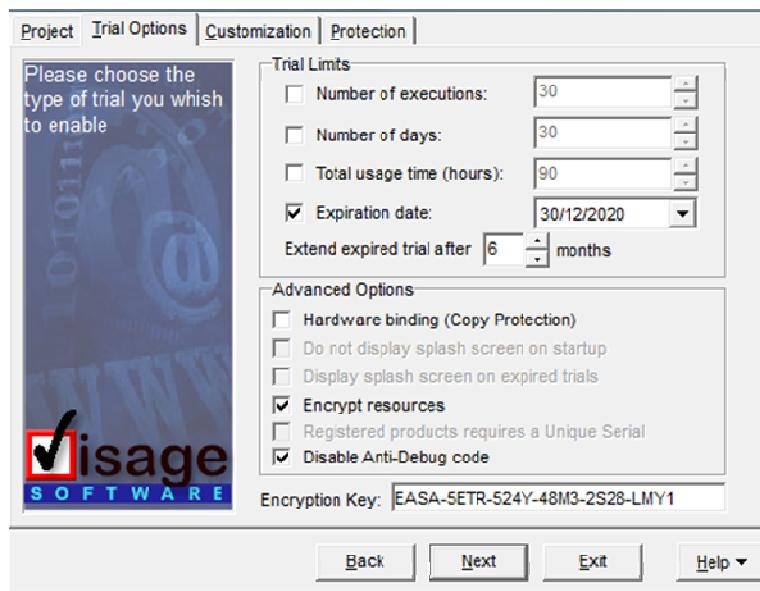
Ahora sabemos donde se encuentra la key después de {XXXXX-XXXXX-XXXX-XXXX} y antes del nombre del archivo de la licencia XXXXXX.vpl

Entonces abrimos la app objetivo (CONCAR V2009) con el ollydbg y repetimos la ultima parte pero en este caso no conocemos la key con que fue encriptado la licencia pero sabemos el nombre de la licencia así que buscamos en el dump CONCAR09.vpl y ahí tenemos la key.

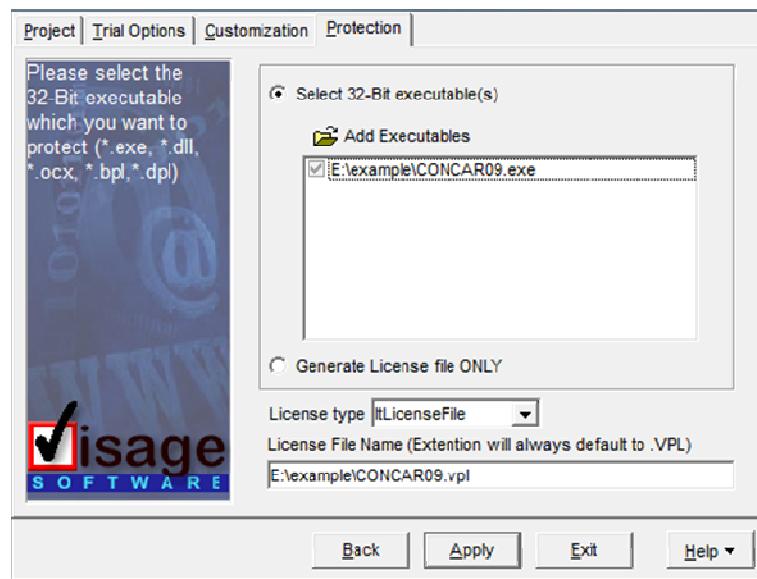
Address	ASCII dump
01B4F7F0	.s8+3 4s06(pF&HnNk·Dw··gI7e·F·#6+
01B4F810	T<B"06·1>aRa,-▲3_061_601Y<YT.Ud
01B4F830	L~.Y@0aa@VVU@J@e+A@X··?d+~41·g@]
01B4F850	S@A·A'·0@e~M·@0A9@g"·@·@p@·A"f@
01B4F870	-@s(@6JE@0a·=T·.t@·0@·0@#@;@=Sp
01B4F890	ud@+@e@-U@·@sc@{#@#5J@+@?i@{#h@·
01B4F8B0	@UEUWV@39924.5918305093.....
01B4F8D0	.....
01B4F8F0	.....@2h0..@.(S..i@u
01B4F910	03.....@E@0150141628-8
01B4F930	E0R-4979-8C7C-B51EA14RA3E2.....
01B4F950	.....EASA-5ETR-
01B4F970	524Y-48M3-2S28-LMY1.....
01B4F990	CONCAR09.
01B4F9B0	vpl.....
01B4F9D0	00wy2uFZ
01B4F9F0	TINw03FXHnxFdTim4cw1yv9puYEzuq24
01B4FA10	LK2zNUss@5wumX0S8y+Gw@jQLTx@FRsn
01B4FA30	EK42uJgLmDhYsyvrKCMUR64Mo08i4fx
01B4FA50	TERyKXcJ2eU/+@+utIhC@psJg++RUH4
01B4FA70	2sRFaA4u6GF40KM2hrRRRWg@0u04SDE
01B4FA90	Jj1PUkzUyFUEic1G5+NQenXJSQJ0UMJu
01B4FAB0	yD2AE0SSST2rgX2uwU2n26geRvQTUpXFx
01B4FAD0	e5vBtI1TjwvZwkoA9MzILxuJ.....
01B4FAF0	.....
01B4FB10	.....
01B4FB30	.....
01B4FB50	.....

key

Entonces ahora vamos a generar una licencia que expire en el 30/12/2020 para ello tomamos otra vez cualquier app para empaquetarlo y lo renombramos como se llama el app objetivo en este caso CONCAR09.EXE.



Le damos APPLY y nos genera el archivo CONCAR09.vpl



Seguidamente remplazamos la licencia por el que generamos el cual expira en el año 2020 y procedemos a ejecutar la app OOHHH SORPRESA!!!! corre con total normalidad para asegurarnos viajamos en el tiempo 1 mes y volvemos a ejecutar y la app aun no ha expirado vamos por buen camino, nuevamente viajamos en el tiempo ahora un par de años y la app sigue trabajando con total normalidad, en conclusión tenemos app para 10 años ☺.



## 5- DESEMPACAR VISUALPROTECT (SOLUCION #02)

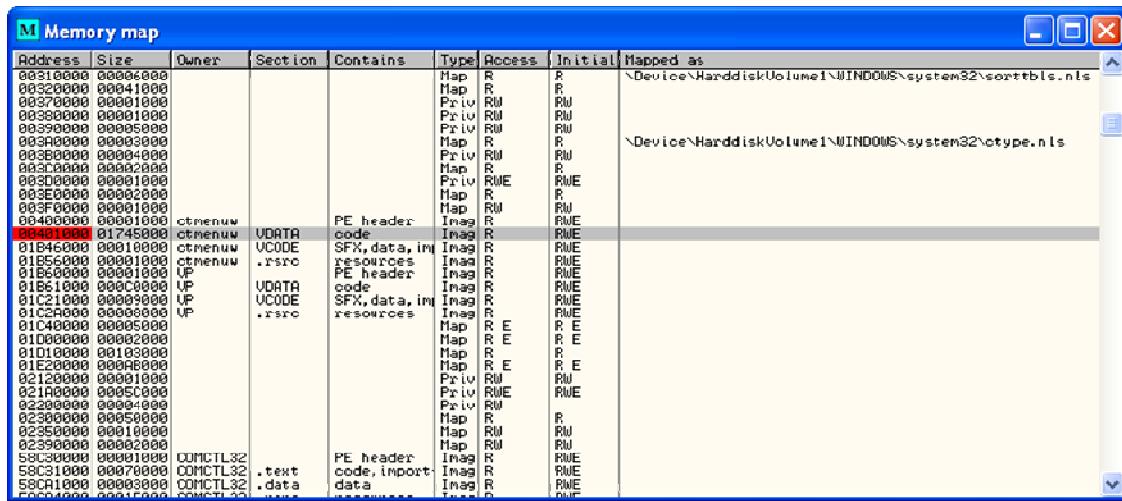
Al ataque caballeros,



La imagen nos confirma que esta con visual protect, cargamos el app objetivo con el ollydbg e iniciamos aquí.

01B48F90	55	PUSH EBP
01B48F91	88EC	MOV EBP, ESP
01B48F93	S1	PUSH ECX
01B48F94	S9	PUSH EBX
01B48F95	S6	PUSH ESI
01B48F96	S7	PUSH EDI
01B48F97	C705 B0FEB401 01	MOV DWORD PTR DS:[1B4FEB0], 0
01B48F98	68 48E0B401	PUSH ctmenuuw.01B4E048
01B48F99	FF15 00C6B401	CALL DWORD PTR DS:[&KERNEL32.LoadLibraryA]
01B48F9A	A3 0CEFB401	MOV DWORD PTR DS:[1B4FF0C], EAX
01B48F9B	68 58E0B401	PUSH ctmenuuw.01B4E058
01B48F9C	A1 0CEFB401	MOV EDX, DWORD PTR DS:[1B4FF0C]
01B48F9D	S0	PUSH EDK
01B48F9E	FF15 04C0B401	CALL DWORD PTR DS:[&KERNEL32.GetProcAddress]
01B48F9F	A3 90FEB401	MOV DWORD PTR DS:[1B4FE90], EAX
01B48FC0	6A 00	PUSH 0
01B48FC1	FF15 90FEB401	CALL DWORD PTR DS:[1B4FE90]
01B48FC2	A3 ECFEB401	MOV DWORD PTR DS:[1B4FEEC], EAX
01B48FC3	88D0 ECFEB401	MOV ECX, DWORD PTR DS:[1B4FEEC]
01B48FC4	S1	PUSH ECX
01B48FC5	E8 C0050000	CALL ctmenuuw.01B495A0
01B48FC6	09C4 04	ADD ESP, 4
01B48FC7	8945 FC	MOV DWORD PTR SS:[EBP-4], EAX
01B48FC8	8970 FC 00	CMP DWORD PTR SS:[EBP-4], 0
01B48FC9	-74 03	JE SHORT ctmenuuw.01B49FEE
01B48FCB	FF65 FC	JMP DWORD PTR SS:[EBP-4]
01B48FCC	SF	POP EDI
01B48FCD	S5	POP ESI
01B48FCE	S8	POP EDX
01B48FCF	88E5	MOV ESP, EBP
01B48FCF	S0	POP EBP
01B48FF0	C3	RET
01B48FF1	S5	PUSH EBP
01B48FF2	88EC	MOV EBP, ESP
01B48FF3	S1	PUSH ECX
01B48FF4	S6	PUSH ESI
01B48FF5	S7	PUSH EDI

Luego presionamos Alt+M para y colocamos un bp en la siguiente posición.



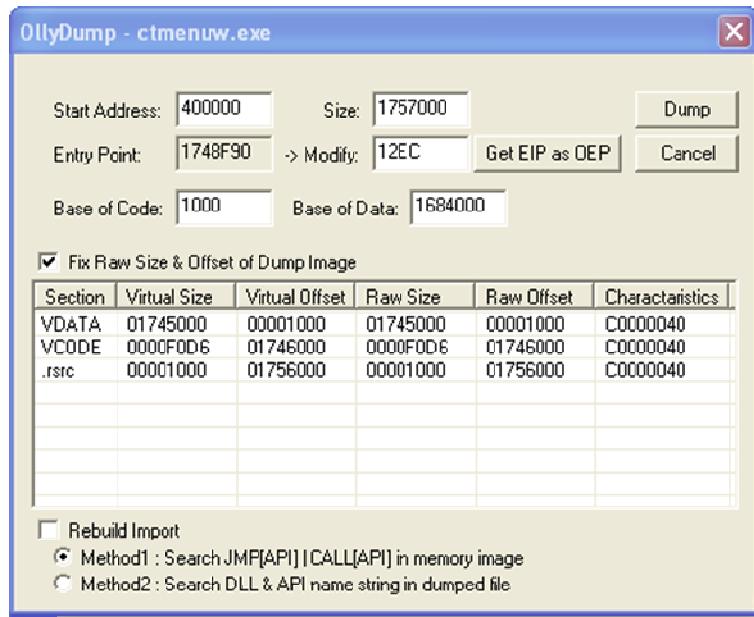
Presionamos F9 ejecutar el app y nos denemos en:

01B4AFEC	CC	INT3
01B4AFED	CC	INT3
01B4FEE	CC	INT3
01B4AFFE	CC	INT3
01B4AFF0	55	PUSH EBP
01B4AFF1	8BEC	MOV EBP,ESP
01B4AFF3	57	PUSH EDI
01B4AFF4	56	PUSH ESI
01B4AFF5	53	PUSH EBX
01B4AFF6	8B4D 10	MOV ECX,DWORD PTR SS:[EBP+10]
01B4AFF9	v E9 26	JE CXZ SHORT ctmenuw.01B4B021
01B4AFFB	8BD9	MOV EBX,ECX
01B4AFFD	8B70 08	MOV EDI,DWORD PTR SS:[EBP+8]
01B4AFFE	8B77 FF	MOV ESI,EDI
01B4AFF2	33C0	XOR EAX,EAX
01B4B004	F2:AЕ	REPNE SCAS BYTE PTR ES:[EDI]
01B4B006	F7D9	NEG ECX
01B4B008	03CB	ADD ECX,EBX
01B4B00A	8BFE	MOV EDI,ESI
01B4B00C	8B75 0C	MOV ESI,DWORD PTR SS:[EBP+C]
01B4B00F	F3:A6	REPE CMPS BYTE PTR ES:[EDI],BYTE PTR DS
01B4B010	8A46 FF	MOV AL,BYTE PTR DS:[ESI-1]
01B4B014	33C9	XOR ECX,ECX
01B4B016	3A47 FF	CMP AL,BYTE PTR DS:[EDI-1]
01B4B019	v 77 04	JR SHORT ctmenuw.01B4B01F
01B4B018	v 74 04	JE SHORT ctmenuw.01B4B021
01B4B01D	49	DEC ECX
01B4B01E	49	DEC ECX
01B4B01F	F7D1	NOT ECX
01B4B021	8BC1	MOV EAX,ECX
01B4B023	5B	POP EBX
01B4B024	5E	POP ESI
01B4B025	5F	POP EDI
01B4B026	C9	LEAVE
01B4B027	C3	RETN
01B4B028	CC	INT3
01B4B029	CC	INT3
01B4B02A	CC	INT3

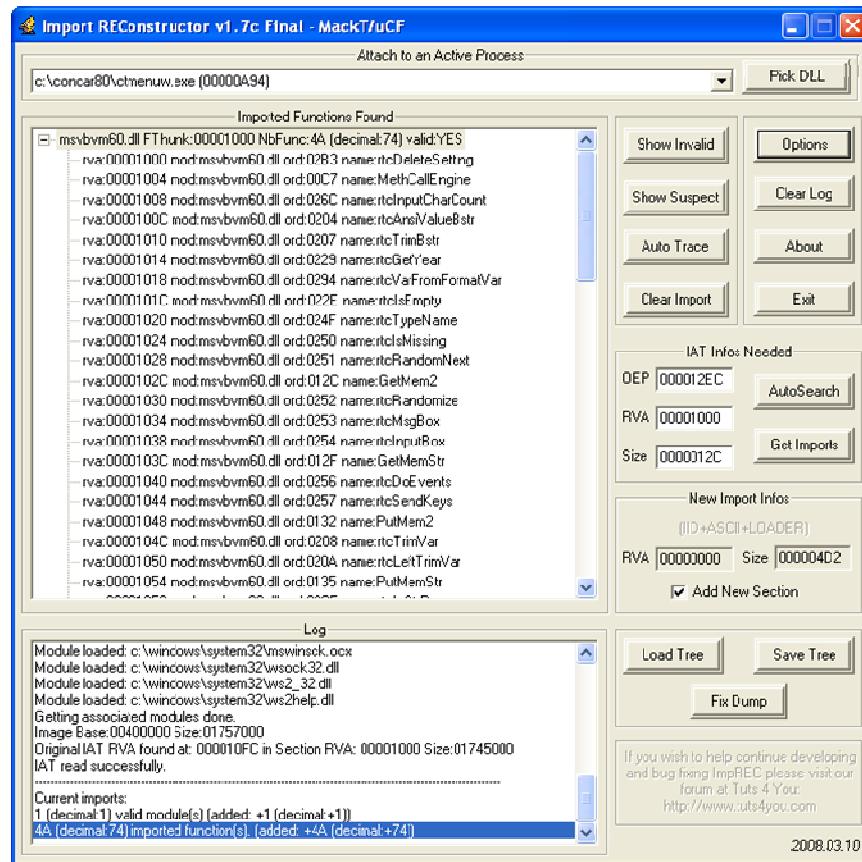
Aun nada raro, traceamos el código presionado F8 hasta llegar a este punto que evidentemente es raro el código y cambia el OFFSET drásticamente esto quiere decir que hallamos el OEP

00401286	-FF25 00114000	JMP DWORD PTR DS:[401100]
0040128C	-FF25 00104000	JMP DWORD PTR DS:[401000]
00401292	-FF25 10104000	JMP DWORD PTR DS:[401010]
00401298	-FF25 88104000	JMP DWORD PTR DS:[401088]
0040129E	-FF25 A0104000	JMP DWORD PTR DS:[4010A0]
004012A4	-FF25 9C104000	JMP DWORD PTR DS:[40109C]
004012A9	-FF25 70104000	JMP DWORD PTR DS:[401070]
004012B0	-FF25 90104000	JMP DWORD PTR DS:[401090]
004012B6	-FF25 04104000	JMP DWORD PTR DS:[401004]
004012BC	-FF25 CC104000	JMP DWORD PTR DS:[4010CC]
004012C2	-FF25 3C104000	JMP DWORD PTR DS:[40103C]
004012C8	-FF25 54104000	JMP DWORD PTR DS:[401054]
004012D4	-FF25 2C104000	JMP DWORD PTR DS:[40102C]
004012D9	-FF25 48104000	JMP DWORD PTR DS:[401048]
004012DA	-FF25 BC104000	JMP DWORD PTR DS:[4010BC]
004012E0	-FF25 C4104000	JMP DWORD PTR DS:[4010C4]
004012E5	-FF25 FC104000	JMP DWORD PTR DS:[4010FC]
<b>004012EC</b>	<b>68 38144000</b>	PUSH ctmenuw.00401438
004012F1	E8 F0FFFFFF	CALL ctmenuw.004012E6
004012F6	0000	ADD BYTE PTR DS:[EAX], AL
004012F8	0000	ADD BYTE PTR DS:[EAX], AL
004012FA	0000	ADD BYTE PTR DS:[EAX], AL
004012FC	3000	XOR BYTE PTR DS:[EAX], AL
004012FE	0000	ADD BYTE PTR DS:[EAX], AL
00401300	3800	CMP BYTE PTR DS:[EAX], AL
00401302	0000	ADD BYTE PTR DS:[EAX], AL
00401304	0000	ADD BYTE PTR DS:[EAX], AL
00401306	0000	ADD BYTE PTR DS:[EAX], AL
00401308	SC	POP ESP
00401309	A0 C62DE5A6	MOV AL,BYTE PTR DS:[A6E52DC6]
0040130E	<b>&lt;E2 4A</b>	LOOP SHORT ctmenuw.0040135H
00401310	9C	PUSHFD
00401311	12D6	ADC DL,DH
00401313	D269 A1	SHR BYTE PTR DS:[ECX-5F],CL
00401316	3121	XOR DWWORD PTR DS:[ECX],ESP
00401319	0000	ADD BYTE PTR DS:[EAX], AL
0040131D	0000	ADD BYTE PTR DS:[EAX], AL
0040131E	0000	ADD BYTE PTR DS:[EAX], AL

Una vez hallado el oep nos disponemos a dumper para eso hacemos uso del OllyDump, 1ro desmarcamos “Rebuild Import”, 2do como EIP esta apuntando a 004012EC el cual es nuestro OEP presionamos “Get EIP as OEP” luego “Dump” y guardamos con el nombre de Cdump.exe



Seguidamente nos disponemos a reparar la IAT para lo cual haremos uso del IMPORT REC ejecutamos el app objetivo original luego ejecutamos el Import REC y seleccionamos el proceso activo (ctmenuw.exe), seguidamente cambiamos el OEP = 00400000 – 004012EC = 000012EC, luego presionamos “AutoSearch” y luego “Get Imports” todo esta OK



Luego presionamos en "Fix Dump" y seleccionamos nuestro archivo dumpeado (Cdmp.exe) , ya tenemos corregida la IAT, vemos que lo guardo con otro nombre (Cdmp\_.exe).

```
IAT read successfully
-----
Current imports:
1 [decimal:1] valid module(s) [added: +1 [decimal:+1]]
4A [decimal:74] imported function(s). [added: +4A [decimal:+74]]
-----
Fixing a dumped file...
1 [decimal:1] module(s)
4A [decimal:74] imported function(s).
*** New section added successfully. RVA:01757000 SIZE:00001000
Image Import Descriptor size: 14; Total length: 4D2
C:\Concar80\cdump_.exe saved successfully.
```

Ejecutamos nuestro app corregido (Cdmp\_.exe) para ver si todo va bien y nos sale el siguiente mensaje.



Cargamos el app corregido con el ollydbg y presionamos F9 y vemos que carga el archivo | “vp.dll” así que el app corregido tiene aun esta protección, veamos donde llama a “vp.dll” para esto usaremos “PROCESS MONITOR” lo ejecutamos y luego ejecutamos nuestro app corregido para ver en que momento llama a la vp.dll o ejecuta alguno de sus modulos, buscamos en el process monitor vp.dll y tenemos esto:

Process Name	PID	Operation	Path
cdump_.exe	3600	RegQueryValue	HKLM\System\CurrentControlSet\Control\Nls\CodePage\949
cdump_.exe	3600	RegQueryValue	HKLM\System\CurrentControlSet\Control\Nls\CodePage\950
cdump_.exe	3600	RegQueryValue	HKLM\System\CurrentControlSet\Control\Nls\CodePage\936
cdump_.exe	3600	RegOpenKey	HKLM\SOFTWARE\Microsoft\WBA\Monitors
cdump_.exe	3600	RegOpenKey	HKLM\SOFTWARE\Microsoft\WBA\Monitors
cdump_.exe	3600	QueryOpen	C:\Concar80\vp.dll
cdump_.exe	3600	QueryOpen	C:\WINDOWS\system32\vp.dll
cdump_.exe	3600	CreateFile	C:\WINDOWS\system32\vp.dll
cdump_.exe	3600	CreateFileMapping	C:\WINDOWS\system32\vp.dll
cdump_.exe	3600	CreateFileMapping	C:\WINDOWS\system32\vp.dll
cdump_.exe	3600	CloseFile	C:\WINDOWS\system32\vp.dll
cdump_.exe	3600	QueryOpen	C:\Concar80\OLEPRO32.DLL
cdump_.exe	3600	QueryOpen	C:\WINDOWS\system32\olepro32.dll
cdump_.exe	3600	CreateFile	C:\WINDOWS\system32\olepro32.dll
cdump_.exe	3600	CreateFileMapping	C:\WINDOWS\system32\olepro32.dll
cdump_.exe	3600	CreateFileMapping	C:\WINDOWS\system32\olepro32.dll
cdump_.exe	3600	CloseFile	C:\WINDOWS\system32\olepro32.dll
cdump_.exe	3600	QueryOpen	C:\Concar80\WINMM.DLL
cdump_.exe	3600	QueryOpen	C:\WINDOWS\system32\winmm.dll
cdump_.exe	3600	CreateFile	C:\WINDOWS\system32\winmm.dll
cdump_.exe	3600	CreateFileMapping	C:\WINDOWS\system32\winmm.dll
cdump_.exe	3600	CreateFileMapping	C:\WINDOWS\system32\winmm.dll
cdump_.exe	3600	CloseFile	C:\WINDOWS\system32\winmm.dll

Precionamos Ctrl+K para ver el STACK

Frame	Module	Location	Address	Path
K 4	ntoskrnl.exe	ntoskrnl.exe + 0x8c86c	0x8056386c	C:\WINDOWS\sys
K 5	ntoskrnl.exe	ntoskrnl.exe + 0x90c63	0x80567e63	C:\WINDOWS\sys
K 6	ntoskrnl.exe	ntoskrnl.exe + 0x9abc	0x80571bc	C:\WINDOWS\sys
K 7	ntoskrnl.exe	ntoskrnl.exe + 0x806b	0x804d06b	C:\WINDOWS\sys
U 8	ntdll.dll	ntdll.dll + 0x16d33	0x7c926d33	C:\WINDOWS\sys
U 9	ntdll.dll	ntdll.dll + 0x16f03	0x7c926f03	C:\WINDOWS\sys
U 10	ntdll.dll	ntdll.dll + 0x1ce16	0x7c92ce16	C:\WINDOWS\sys
U 11	ntdll.dll	ntdll.dll + 0x1cc02	0x7c92cc02	C:\WINDOWS\sys
U 12	ntdll.dll	ntdll.dll + 0x16071	0x7c926071	C:\WINDOWS\sys
U 13	ntdll.dll	ntdll.dll + 0x162da	0x7c9262da	C:\WINDOWS\sys
U 14	kernel32.dll	kernel32.dll + 0x1bb9	0x7c801bb9	C:\WINDOWS\sys
U 15	kernel32.dll	kernel32.dll + 0x1d5e	0x7c801d5e	C:\WINDOWS\sys
U 16	kernel32.dll	kernel32.dll + 0x1d44	0x7c801d44	C:\WINDOWS\sys
U 17	msvbm60.dll	msvbm60.dll + 0x1b7ee	0x733bb7ee	C:\WINDOWS\sys
U 18	msvbm60.dll	msvbm60.dll + 0x1b79a	0x733bb79a	C:\WINDOWS\sys
U 19	cdump_exe	cdump_exe + 0x1c4f7f	0x5c4f7f	C:\Corcar80\cdur
U 20	msvbm60.dll	msvbm60.dll + 0xf7f5	0x7349f7f5	C:\WINDOWS\sys
U 21	msvbm60.dll	msvbm60.dll + 0x147da	0x733b47da	C:\WINDOWS\sys
U 22	msvbm60.dll	msvbm60.dll + 0x12099	0x733b2099	C:\WINDOWS\sys
U 23	msvbm60.dll	msvbm60.dll + 0xe24f	0x733ae24f	C:\WINDOWS\sys
U 24	msvbm60.dll	msvbm60.dll + 0xd93	0x733ad93	C:\WINDOWS\sys
U 25	msvbm60.dll	msvbm60.dll + 0xde39	0x733ade99	C:\WINDOWS\sys

Configure the symbol Properties... Search... Source... Save...

Como vemos llama a vp.dll o alguno de sus módulos en el OFFSET 005C4F7F, así que cargamos en el ollydgb nuestro app corregido colocamos un bp en 005C4F7F y le damos F9.

005C4F53	003C4F	ADD BYTE PTR DS:[EDI+ECX*2],BH	
005C4F56	5C	POP ESP	
005C4F57	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F59	008400	ADD BYTE PTR DS:[EAX+EAX],AL	
005C4F5C	44	INC ESP	
005C4F5D	E8 A0010000	ES A0010000	
005C4F62	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F64	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F66	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F68	A1 4CE8A801	HOU EAX,DWORD PTR DS:[1A8E84C]	
005C4F6D	00C0	OR EAX,EAX	
005C4F6F	v74 02	JE SHORT cdump_.005C4F73	
005C4F71	FFE0	JMP EAX	
005C4F73	68 504F5000	PUSH cdump_.005C4F50	
005C4F78	88 30114000	MOU EAX,<JMP.&msvbm60.DLLFunctionCal	
005C4F7D	FFD0	CALL EAX	
005C4F7F	-FFE0	JMP EAX	vp.VPGetLicenseInfo
005C4F81	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F82	0009	ADD BYTE PTR DS:[ECX],CL	
005C4F85	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F87	006B 65	ADD BYTE PTR DS:[EBX+65],CH	
005C4F88	v72 6E	JB SHORT cdump_.005C4FFA	
005C4F8C	65:6C	INS BYTE PTR ES:[EDI],DX	
005C4F8E	3332	XOR ESI,DWORD PTR DS:[EDX]	
005C4F90	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F92	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F94	1800	SBB BYTE PTR DS:[EAX],AL	
005C4F96	0000	ADD BYTE PTR DS:[EAX],AL	
005C4F98	47	INC EDI	
005C4F99	65:74 45	JE SHORT cdump_.005C4FE1	Superfluous prefix
005C4F9C	6E	OUTS DX,BYTE PTR ES:[EDI]	I/O command
005C4F9D	v76 69	JBE SHORT cdump_.005C5008	
005C4F9F	v72 6F	JB SHORT cdump_.005C5010	
005C4FA1	6E	OUTS DX,BYTE PTR ES:[EDI]	I/O command
005C4FA2	60	INS DWORD PTR ES:[EDI],DX	I/O command
005C4FA3	65:6E	OUTS DX,BYTE PTR ES:[EDI]	I/O command
005C4FA5	v74 66	JB SHORT cdump_.005C4FFD	
005C4FA7	61	POPAO	

Y vemos que llama a "vp.VPGetLicenceInfo" le damos F7 .

025A71BC	55	PUSH EBP
025A71BD	8BEC	MOV EBP, ESP
025A71BF	83C8 08	MOV EDI, EDI
025A71C2	83D2	XOR EDX, EDX
025A71C4	83D0 0C 29	CMP DWORD PTR SS:[EBP+C], 29
025A71C9	C7 4F	JNZ SHORT up_025A7219
025A71CA	803D 04BB5A02 00	CMP BYTE PTR DS:[25ABB04], 0
025A71D1	74 13	JE SHORT up_025A71E6
025A71D3	8B08	MOV EDX, EDX
025A71D5	A1 C0BB5A02	MOV EAX, DWORD PTR DS:[25ABB02]
025A71D9	B9 29000000	MOV ECX, 29
025A71DF	F8 2C88F5FF	CALL up_02502910
025A71E4	EB 31	JMP SHORT vp_025A7217
025A71E6	33D2	XOR EDX, EDX
025A71E8	8910	MOV DWORD PTR DS:[EAX], EDX
025A71E9	8958 04	MOV DWORD PTR DS:[EAX+4], EDX
025A71ED	83D2	XOR EDX, EDX
025A71EF	8958 08	MOV DWORD PTR DS:[EAX+8], EDX
025A71F2	83D2	XOR EDX, EDX
025A71F4	8958 0C	MOV DWORD PTR DS:[EAX+C], EDX
025A71F7	83D2	XOR EDX, EDX
025A71F9	8958 10	MOV DWORD PTR DS:[EAX+10], EDX
025A71FB	83D2	XOR EDX, EDX
025A71FE	8950 14	MOV DWORD PTR DS:[EAX+14], EDX
025A7201	8950 18	MOV DWORD PTR DS:[EAX+18], EDX
025A7204	33D2	XOR EDX, EDX
025A7206	8958 1C	MOV DWORD PTR DS:[EAX+1C], EDX
025A7209	83D2	XOR EDX, EDX
025A720B	8958 20	MOV DWORD PTR DS:[EAX+20], EDX
025A720E	83D2	XOR EDX, EDX
025A7210	8958 24	MOV DWORD PTR DS:[EAX+24], EDX
025A7213	C640 28 00	MOV BYTE PTR DS:[EAX+28], 0
025A7217	R2_A1	MOU DL, 1
025A7219	8BC2	MOV EAX, EDX
025A721B	5D	POP EBP
025A721C	C2 0800	RETN 8
025A721F	90	NOP
025A7220	55	PUSH EBP

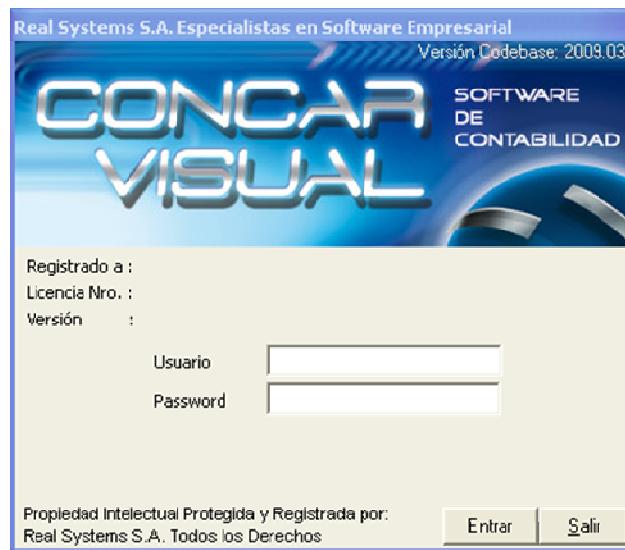
Ya que vimos el código anterior traceamos hasta llegar a retn 8 y no salta en JNZ pero si en JE evitando el CALL Y EL JMP si que todo está en este pequeño segmento de código, así que analizando que hace línea por línea y en especial **MOV BYTE PTR DS:[EAX+28],0** traceamos un poco y vemos en el stack nuestro chico(UNREGISTERED) malo en la dirección 0015E1FC.

0012F7BC	00140000	
0012F7C0	00000000	
0012F7C4	00000000	
0012F7C8	00140000	
0012F7CC	00000001	
0012F7D0	00000000	
0012F7D4	00000001	
0012F7D8	00000000	
0012F7DC	00000000	
0012F7E0	00000000	
0012F7E4	00140000	
0012F7E8	0012F838	
0012F7EC	7C97E686	RETURN to ntdll.7C97E686 from ntdll.7C97CDC9
0012F7F0	00140000	
0012F7F4	00000000	
0012F7F8	00140000	
0012F7FC	00149AF8	
0012F800	00176B50	
0012F804	00000000	
0012F808	00000000	
0012F80C	0015E1FC	UNICODE "Unregistered"
0012F810	00140000	
0012F814	0151A5BF	cdump_.0151A5BF
0012F818	FFFFFFFFFF	
0012F81C	01140000	cdump_.01140000
0012F820	0012F7F8	
0012F824	0012F7FC	
0012F828	0012F984	Pointer to next SEH record
0012F82C	7C91EE18	SE handler

Para no hacerlo extenso este tute modificamos **MOV BYTE PTR DS:[EAX+28],0** por **BYTE PTR DS:[EAX+28],1** y traceamos un poco y tenemos en el stack el chico bueno (REGISTERED) .

```
0012F808 00000000| 0015E1FC UNICODE "Registered"  
0012F80C 00140000| 0151A5BF cdump_.0151A5BF  
0012F810 FFFFFFFF|  
0012F814 0151A5BF cdump_.0151A5BF  
0012F818 FFFFFFFF|  
0012F81C 01140000 cdump_.01140000  
0012F820 0012F7F8|  
0012F824 0012F7FC|  
0012F828 0012F984 Pointer to next SEH record  
0012F82C 7C91EE18 SE handler  
0012F830 7C97E728 ntdll.7C97E728  
0012F834 00000001|  
0012F838 0012F850|  
0012F83C 7C95976B RETURN to ntdll.7C95976B from ntdll.7C97E60B  
0012F840 00140000|  
0012F844 50000061|  
0012F848 00176B50|  
0012F84C 00000002|  
0012F850 0012F864|  
0012F854 77402590 RETURN to ole32.77402580 from ntdll.RtlSizeHeap  
0012F858 00140000|  
0012F85C 00000000|  
0012F860 00176B50|  
0012F864 0012F884|  
0012F868 770F4A64 RETURN to OLEAUT32.770F4A64  
0012F86C 775D5034 ole32.775D5034  
0012F870 00176B50|  
0012F874 00000002|  
0012F878 FFFFFFF10|
```

Para terminar le damos F9 y el app corre de maravillas.



Así que solo tenemos que hacer un loader para modificar vp.dll en tiempo de ejecución y listo.

## 6- AGRADEMIENTOS

En primer lugar a **Eddy -=InDuLgEo=-**, por animarme a realizar este tuto, para mí que estoy empezando me ha costado hacer todo esto. Gracias Eddy.

En segundo lugar a la lista **CracksLatinos** liderado por el maestro Ricardo Narvaja, espero que les haya gustado este tuto... Gracias



By SNB