muy bien, los que quieran aprender CRACKING, por lo menos lo básico, sigan mis instrucciones, los que no quieren, simplemente no lo hagan.
existen muchos significados para la palabra CRACKING, el que vamos a utilizar es el siguiente:

CRACKING: rama del hacking que se enfoca en modificar software a nuestra conveniencia.

ya sea para eliminar una limitacion por ser version trial (lo más utilizado, de ahí viene la palabra "crack"; conseguime el "crack" de tal programa etc, etc...), para traducir un determinado programa o cambiarle las strings dejando nuestro rastro, para cambiarle alguna propiedad frente al sistema, etc, etc, etc... sencillamente podemos manosear las tripas a cualquier programa como mejor nos convenga.

para comenzar es necesario que estudien estos dos manuales.
son largos, y no es para cualkiera. de ellos aprendí yo; lo ke pretendo es que pasen por todo lo que pasé yo para que aprendan a hacer las pocas cosas que puedo hacer,que ya han visto...

el primero que tienen que leer es el manual de crackeo desde cero de Mr NOBODY--->http://rapidshare.com/files/29168839/COC_-_Completo.zip.html

luego, secundariament e o si sienten la necesidad, pueden leerlo a la par con el primero, este manual de ASSEMBLER básico para aplicar en cracking, creado por Caos Reptante--->http://rapidshare.com/files/29170001/Assembler.doc.html

bien, para continuar abanzando es necesario leer esos dos manuales, recuerden que estamos aprendiendo a hackear software desde cero... importantisimo tener un orden y disciplina.

si no leyeron estos dos manuales, o no lo estan haciendo, no me pregunten nada ni me pidan ninguna herramienta.
si los comienzan a leer y les surjen preguntas háganmelas; solo posteos porfavor.

hay mucha informacion del tema, tengan en claro que el exeso de informacion es desinformacion, no se dejen llevar por la tentacion de aprender muchas cosas diferentes de golpe. si kieren aprender cracking sigan mis posteos, este es el primero.

a pesar de las críticas que voy a recibir y que vengo recibiendo, si siguen mis posteos los voy a ayudar en todo lo que pueda, pero si me doy cuenta que ni siquiera se molestaron en leer o seguir mis instrucciones, no lo voy a hacer.

es simple, este 1er post de cracking es porque mucha gente me lo esta pidiendo.
vamos a hacer las cosas ordenadamente, ya les planteé mi orden.

no se si corresponde esto que estoy haciendo en este foro, si me equiivoco por favor administradore s corrijanme y orientenme para saber donde publicar este tema, simplemtene pienso que es el foro correcto ya que "aplicar cracking es hackear software".

graxias, espero sus preguntas, espero sus posteos, vamos a ver si vale la pena continuar con el tema.

----------By Moskito "The Moskito´s Crackers"----------

10/Sep/2007 16:33 GMT-3 Perfil · Privado · Desconectado · Web

TARCRO

ABAD

Mensajes: **459**
Desde: **22/May/2006**
#2

RE: COMENZANDO A CRACKEAR BY MOSKITO

CONTINUAMOS ...

si no leyeron los 2 manuales que dejé para que bajen en el posteo anterior, ni se molesten en leer esto porque no van a poder digerir nada, amenos que ya vengan con conocimientos. ..

bien bien, para continuar con el tema de cracking, si ya leyeron los 2 manuales que dejé en el post "Comenzando a crackear con moskito", paso a darles un par de datos más.
si no leyeron nada, lo siguiente no les va a servir... holgazanes...

la primer y fundamental herramienta a la hora de crackear: papel y lapiz. por muchas razones...

para encarar un crackeo debemos anotar con papel y lapiz la limitacion que nos molesta y el síntoma que notamos.
por ejemplo, agarro una oja y escribo:

lmitacion: versión a prueba 30 dias y cagamos.
síntoma: un cartel que me informa el dia de expiracion, o directamente me informa que el programa está expirado.

bien, cuando ya tenemos esa informacion, sabemos cual es el objetivo: liberar a el programa para que no expire nunca.
a que parte apuntamos, quual es la estrategia? la que mejor nos paresca, mientras se cumpla el objetivo todo vale;

-podemos apuntar a forzar la registracion con un numero cualkiera que le ingresemos, forzar a el programa para aceptar cualquier numero.
-o simplemente forzar a el programa para que se saltee la parte de verificar los dias que nos quedan antes de que expire. y de este modo continuar por siempre sin esa limitacion.
-modificarle el algoritmo encargado de contar cuantos dias lleva instalado, o que numero de veces se ejecutó, para que no sume más y siempre estemos como en la primer ejecucion.
-etc, etc, etc...

tenemos un programa limitado, temenos el tipo de limitacion y el sintoma, con el objetivo.
bien, el siguiente paso es verificar que el programa no está empakado, y si lo está, cual es el paker.
que significa esto? weno, nosotros vimos que podemos desensamblar un programa para verle las entrañas en codigo assembler, rastrear mediante los mensajes de error el salto condicional y, luego con un editor hexa realizar la modificacion para que salte a donde queramos.
bien, este tipo de crackeo es utilizando el codigo muerto, sin estar ejecutando el programa, sin estar debuguenadolo. osea estudiamos el codigo sin que el progrma este funcionando, solo agarramos el exe, lo abrimos y realizamos esa modificacion.

pero resulta que los muy malditos inventaron un sistema para cagarnos el dia a los crackers, este sistema se llama "portable ejecutable" o PE. y hace ke se nos complike a la hora de realizar una modificacion utilizando el codigo muerto.
como trabaja un PE? weno, un archivo exe es comprimido en si mismo y cuando se ejecuta, se descomprime en memoria (la ram) y asi funcona descomprimido.
osea que en ese exe existe un codigo de descompresion de si mismo que hace que al ejecutarse éste, se reorganicen de cierta manera los bytes dentro de la ram para ejecutarse correctamente.
osea que si desensamblamos el muy maldito, solo veremos la rutina de descompresion y luego pura basura, no vamos a encontrar nada, ninguna cadena de caracteres, nada de nada ya que todo eso aparecerá cuando este archivo este cargado en la ram.
osea que ya deducimos que no se puede crackear en codigo muerto. necesitamos de otro tipo de herramientas.
bien, ahora no es mi objetivo explicar como crackear un PE, sino continuar con la técnica básica para crackear un progrma, más adelante tocaremos el tema.
solo expliqué esto para que sepan de que no todos los programas pueden ser tratados por igual ya que existe este tipo de protecciones PE(portable executable). osea qye si desensamblamos un exe y no encontramos ninguna cadena de caracteres, así como es probable que se creen en la ejecucion, tambien es probable que aparescan en el archivo cargado en memoria por ser un PE.

retomando: tenemos un programa limitado, temenos el tipo de limitacion y el sintoma, con el objetivo.
bien, el siguiente paso es verificar que el programa no está empakado, y si lo está, cual es el paker? eso lo podemos verificar con cualquiera de stos dos progrmas...

download analizadores de archivos---
>http://rapidshare.com/files/29896107/analizadores_de_archivos.rar.html

entonces; con estos analizadores de archivos (utilizo los 2) vamos a detectar si el programa que queremos crackear esta empaketado o no, y si no lo está veremos en que lenguaje fué programado.

bueno, entonces ya vamos completando nuestro kit de herramientas, aca les dejo el desensamblador y el editor exadecimal. si ya leyeeron ya saben como funxionan y para que etapa del crackeo sirven.

download desensamblador---
>http://rapidshare.com/files/29895570/version_8.93.rar.html
download editor hexa--->http://rapidshare.com/files/29898456/UltraEdit-32.rar.html

bien amigos, ya tienen 2 manuales fundamentales y básicos, los del posteo anterior.
ya tienen el desensamblador y el editor hexa.
y ya tienen los analizadores de archivos (ke sirven para detectar si estan o no empaketados)

en todo crackeo, la primer herramienta que se utiliza el es papel y lapiz, para anotar los datos del crackeo y las direcciones, offsets, etc etc...
la segunda herramienta ke se utiliza al hacer un crackeo siempre es el analizador de archivos, porque si no sabemos si esta empacado o no, no nos podemos tirar a la deriva a desensamblar como lokos...
y la tercer herramienta ke se utiliza depende de la info recolectada por las anteriores.
a ver si adivinan cual será...

continuaremos en otro posteo de "Comenzando a crackear con MOSKITO" ...

suerte y si tienen preguntas hasta ahora, haganlas que seran respondidas...

10/Sep/2007 16:37 GMT-3 Perfil · Privado · Desconectado · Web

TARCRO

ABAD

Mensajes: **459**
Desde: **22/May/2006**
#3

RE: COMENZANDO A CRACKEAR BY MOSKITO

CONTINUANDO CON EL CRACKING desde las LIMITADAS PALABRAS DE MOSKITO...

weno, vamos a aclarar algunos puntos básicos para enriqueser la lectura de los manuales ke les dejé...

Para comprender la necesidad del cracking, y al cracking en si, tenemos ke entender el sencillo significado de la palabra "PROGRAMA", ya ke el cracking se basa en la modificacion de PROGRAMAS para nuestro beneficio:

Un programa es una serie de instrucciones ordenadas para ke una makina realice cierto trabajo.

Vamos a referirnos especificament e a una PC, sinembargo todo lo ke explico se puede aplicar a CUALKIER TIPO DE MAKINA, incluso seres vivos.

QUE TEINE KE VER EL SISTEMA BINARIO CON LA ELECTRÓNICA DE UNA MAKINA ??

Nosotros, como seres humanos, realizamos los movimientos de nuestras extremidades bajo ordenes químicas de nuestro cerebro.
Una PC es una makina ke trabaja gracias a la alectricidad, electrónica (electrones en movimiento), esto significa que una PC solo interpreta informacion electronica, no química (aún).
A muy grandes razgos; para realizar cierto trabajo, la pc manipula ordenes electronicas, el procesador (cerebro) trabaja por ahora con ordenes electronicas, esto significa, ilando fino y apresuradament e, que si desde el teclado enviamos una orden al cerebro de la pc, y luego de pasar por millones de circuitos y componentes el procesador resive (siente) un pulsito de 5 volts en determinada patita, saca 5 volts por otras 5 patitas ke se comunican con otros circuitos y componentes electronicos y asi de esta manera y pasando por millones de componentes y circuitos nos llega el resultado al monitor.
Entonces debemos grabarnos el concepto de que la pc trabaja con ordenes electricas, electrónicas más especificament e, ya ke sus componentes trabajan a nivel electron.
El sistema de numeracion binario, se basa en 2 estados representados por 1 y 0. La pc trabja a nivel binario e interpreta un 1 cuando hay presencia de voltage y a un 0 cuando no hay presencia de voltaje. ësa es la relacion ke existe entre el sistema binario y el

electronico.

METIENDONOS EN EL CRACKING

bueno, recordando al definición de PROGRAMA, y analizando cómo se maneja con órdenes electrónicas una pc, llegamos a la conclusion de que un programa para una pc consiste en un listado de órdenes electrónicas ordenadas de tal manera ke el microprocesado r central y los demas componentes electrónicos puedan interpretar para realizar un trabajo y devolver un resultado producto de ese trabajo realizado.

Ahora bien, si necesitamos romper (crackear) un programa para ke haga lo ke nosotros keramos debemos cambiar el orden original de esa cadena de pulsos electronicos ke llegarán a el procesador, provenientes de un listado de instrucciones electrónicas almacenadas en algun lugar.
Pero no podemos interponer circuitos electrónicos en medio de los ya existentes para eliminar, agregar o falsear estados electronicos entendidos como órdenes ke en algún momento llegrán a el procesador principal (por lo menos YO todavia no soy capaz de lograr algo así).
Debemos trabajar en la modificacion (eliminacion, agregado, deformacion) de las instrucciones electrónicas antes de ke partan hacia los circuitos de la pc,
osea ke debemos modificar ese listado en donde se encuentra almacenado.

Los programas (listados de instrucciones electronicas, no olvidemos nunca ésto) se almacenan en unidades físicas llamadas Discos duros (si, tambien en cd's, diskettes, dvd's, pendrives, etc etc.. pero voy a explicar basandome en ke trabajamos con programas en HD´s).
Por lo tanto, sabiendo donde se encuentran los programas, vamos a saber a donde atakar (eso es fundamental).

Y vamos a atakar a los programas, para ke? para no tener ke programar y utilizar la programacion de otro en nuestro beneficio... en realidad llegamos a esa conclusion.

Como no es posible trabajar modificando directamente pulsos electronicos, tenemos ke trabajar modificando informacion binaria, pero claro, no es tan facil.
En un principio, grandes figuras inspiradoras de este arte, modificaban datos binarios directamente, pero actualmente tenemos herramientas ke nos facilitan el trabajo y nos permiten trabajar en un nivel más arriva aún (en otros sistemas de numeracion), como es el hexadecimal.

para organizarnos un poko, hablamos de 3 niveles de datos, ya van a ver a ke me estoy refiriendo:

1-el nivel de data más bajo---> pulso electrónico (existencia de éste o ausencia de éste).
2-el nivel de data ke le sigue---> Binario (traduccion de la existencia de un pulso electronico en un 1 y la ausencia del pulso en un 0).
3-el nivel de data siguiente al binario----> HEXADECIMAL (traduccion de 4 datos binarios en 1 solo en hexadecimal).

y ya vamos a analizar ...

4-El nivel de data por ensima del hexa---> Todabía no mencionado NIVEL ASSEMBLER (traduccion de 2 datos hexa en 1 palabra como minimo)

No vamos a profundizar en cada sistema de numeracion ya ke para eso hay manuales ke lo harían mucho mejor ke yo. Es cierto ke para el cracking tambien hay manuales mucho mejores (como el ke dejé para ke bajen), pero es el tema ke sí estoy desarrollando, asike a joderse o a leer el manual ke les dejé jeje...

bueno,a ke viene todo esto, a ke nosotros, como crackers, vamos a trabajar interpretando data en assembler y modificando data en hexadecimal. Dí un pantallazo de lo demás para ke entendamos un poco nadamas de lo ke en realidad estamos modificando.

bien, llegamos a lo ke personalmente keria ke lleguemos:

COMO CRACKERS VAMOS A TRABAJAR INTERPRETANDO PROGRAMAS EN ASSEMBLER Y MODIFICANDO INSTRUCCIONES A NIVEL HEXADECIMAL.

para poder hacer eso, necesitamos conocer el lenguaje assembler, el conosimiento de la data en hexa viene solo...
Para eso es ke dejé esos dos manuales en un principio, hay uno de ellos ke habla de assembler, fundamental.
Otra cosa ke vamos a necesitar para poder interpretar programas en assembler y modificar sus instrucciones en hexadecimal, son herramientas como:

en un principio:

1-DESENSAMBLADOR: para poder abrir un programa y ver su código en assembler.
2-EDITOR HEXADECIMAL: para poder modificar la data del programa a nivel hexadecimal.

si leen los 2 manuales ke dejé y bajan las herramientas ke tambien dejé y seguiré dejando, van a poder llegar a aproximarse bastante a la plenitud del conocimiento y experimentacio n de el arte del cracking.

con esta informacion ke estoy dejando, no desacredito lo ke yá he dejado, tengan en cuenta ke hay cosas ke ya dije ke no las repetí en este post. Deben leer todo lo ke fui dejando desde un principio, para poder entender y relacionar cada una de las cosas de les comparto. es cierto ke está todo muy desordenado, pero consideren ke no soy un profesional en el arte de la literatura... (en algun momento, alguien generozo editará y organizará un poco toda la informacion ke les estoy dejando y seguiré dejando)

Espero no haber causado más confucion hacerca del tema. no soy escritor, solo intento transmitir la informacion ke me "liberó" para que ustedes tambien se sientan libres... piensenlo...

continuara...

10/Sep/2007 16:38 GMT-3 Perfil · Privado · Desconectado · Web

TARCRO

ABAD

Mensajes: **459**
Desde: **22/May/2006**
#4

RE: COMENZANDO A CRACKEAR BY MOSKITO

PREPARANDO EL CAMINO PARA EL CRACKEO CON EL DEBUGUER

APIS

ke carajo son las apis bien, para entender esto tenemos ke retomar un poco el tema de el significado de "programa"...

En base a esto, debemos recordar y pensar en ke un programa puede estar dentro de uno o varios archivos, a ke me refiero ? a ke un programa puede comenzar su ejecucion en un archivo y luego continuar su ejecucion en otro, osea, ejecuto el "kaka.exe" y al comenzar la ejecucion hay un direccionamien to de las instrucciones hacia otro archivo, donde, en base a ke se cumplan ciertas condiciones, se direccionará o no hacia otro archivo o el anterior.

Tambien tenemos ke recordar, o mas bien, ser conscientes de ke en un programa hay instrucciones ke deciden ke trabajo realizar. Para ke un programa sea ordenado, debe tener ésas instrucciones ordenadas, osea agrupadas. En un programa ordenado, cada grupo de instrucciones tiene un titulo, osea un nombre. A esos grupos de instrucciones se los denomina funciones (yo las llamo funxiones).

Para ke el código de un programa no ocupe mucho espacio y sea legible y organizado, se crea solo una serie de funxiones ke más adelante seran llamadas por el código del programa. De este modo el programador se evita el trabajo de tener ke crear ése grupo de instrucciones cada vez ke las necesita y simplemente realiza un llamado a esa funxion (grupo de instr.) mediante algun salto condicional, un call, etc...

Entonces, esas funxiones no solo estan dentro del archivo ejecutable ke tiene el nombre del programa, sino ke esas funxiones están en varios archivos más.

Si no hay dudas hasta acá, seguimos, sino aclárenlas, relean y luego continuen, de este modo evitamos confuciones.

Utilizando la técnica de la repetición, como acostumbro, recordemos ke:

Un programa consta de grupos de instrucciones ke realizan cierto trabajo llamadas funxiones y pueden estar dispersadas en varios archivos no solo en el ejecutable ke tiene el nombre del programa.

y las APIS para cuando?

Un programa funxiona bajo un SO (Sistema Operativo), esto significa ke un programa utiliza los recursos del sistema operativo, y obviamente debe ser compatible con ése sistema operativo.
Por ke debe ser compatible? porke todos los programas ke funxionan, por ejemeplo, bajo windows xp utilizan
funxiones contenidas en archivos del propio sitema operativo
. Osea ke los programas no necesitan traer la mayoria de las funxiones más utilizadas ya ke dentro de los archivos del SO estan esas funxiones.
Las funxiones utilizadas por los programas ke funxionan bajo un SO ke éste mismo las contiene en sus archivos, son las llamadas funxiones API.

A ver, nunca se preguntaron por ke la mayoria de los programas utiliza el mismo entorno gráfico bajo un SO? porke los botones de ACEPTAR o CANCELAR o las venatanas son prácticamente las mismas en todos los programas? Weno, es por eso, porke todos los programas utilizan las funxiones apis de windows o el sistema operativo en el ke trabajen.

Ésas son las APIS, por ejemplo una de las apis más utilizadas es la ke se llama "messageboxa" y esta contenida dentro de un arhcivo llamado user32, y se trata de una funxion encargada de crear una ventana con un boton de aceptar (o cancelar, solo es un ejemplo), y es la más utilizada en el momento de ke un programa nos muestra un error de licencia (por ejemplo).

Hay funxiones Apy para las ventanas, sonidos, imágenes, manoseo del registro, manoseo de archivos, colores, menoseo de memoria, monitor, etc. Incluso si ahora mismo vamos a "ayuda/ acerca de internet explorer" o vamos a el menu de nuestro borwser y le damos a "acerca de" se nos va a abrir una ventana la cual se crea a partir de el pasaje de datos a una funxion api contenida en algun archivo de windows. Esa funxion, sea la ke sea ke se utilice para mostrarnos esa ventana de créditos, necesita datos para saber ke mierda va a mostrar dentro de si y ke ará luego de ke alguien presione el boton aceptar, port ejemplo.... bien, el llamado a esa api se realiza desde nuestro programa browser el cual además de llamarla, le pasa ciertos datos y parámetros.

weno, el mejor concejo ke les puedo dar llegados a éste punto es ke pasen por acá-->http://es.wikipedia.org/wiki/Application_Programming_Interface

Tener el conocimiento de lo ke son las API nos va a servir para interceptar los datos ke manipulan en el momento en ke por ejemplo, intentamos registrar un programa. Tambien nos sirve para poder detener el programa en ese instante en ke éste llama a determinada API para mostrarnos un mensage de error (por ejemplo). Y para infinidad de cosas más.

Todo esto claro, cuando utilicemos el debuguer, el visualizador de la ejecucion de un

programa paso a paso...

vallan estudiando el tema de las API y sigan practicando el crackeo con el w32dasm y el editor hexa, el crackeo de programas sencillos ke pueden bajar de cualkier pajina, no soy partidario de practicar con crackmees ya ke es una tonteria y con el mismo ánimo podrian crackear soft de verdad.

La próxima, y antes de explicar crackeo con el debuguer, vamos a crackear el programa llamado doshttp (ke por cierto ya está en programas hacker crackeado por mi), voy a explicar el crackeo de este programa con imágenes de como lo hice, es más, si se animan, y con las cosas ke hemos visto hastá acá, ustedes mismos lo podrían crackear...

continuará... controlen la anciedad y respeten el orden, lean los manuales y practiken...

10/Sep/2007 16:39 GMT-3 Perfil · Privado · Desconectado · Web

TARCRO

ABAD

Mensajes: **459**
Desde: **22/May/2006**
#5

RE: COMENZANDO A CRACKEAR BY MOSKITO

Un ejemplo de cracking ke apliké hace poco

Ustedes conocen el malware ke hice ke te abre la lectora de cd cada 10 segundos??? bien es éste->http://foro.portalhacker.net/index.php/topic,35038.0.html

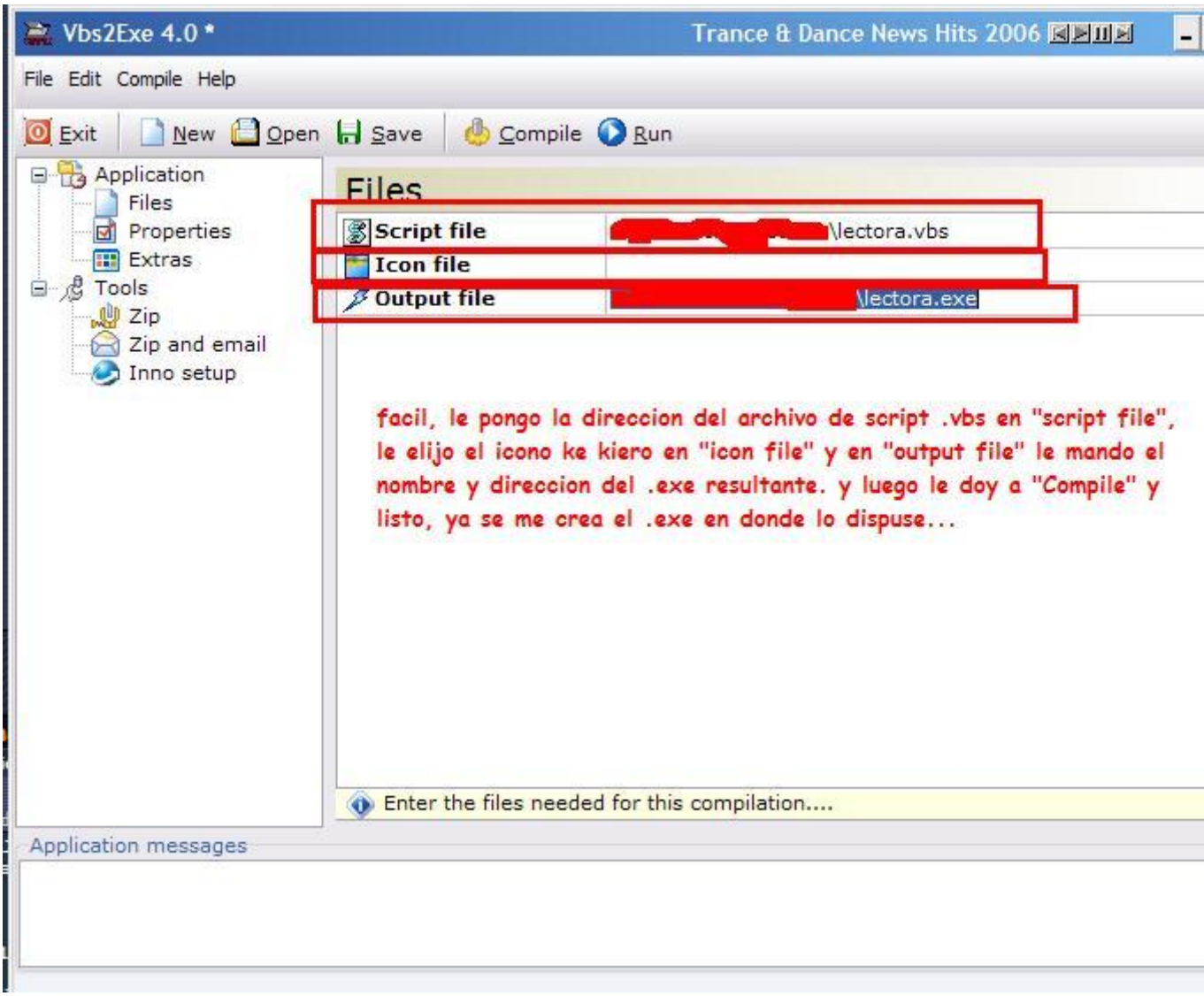
para hacerlo utilicé un script de vb ke es el ke abre la lectora, el script lo pueden descargar de acá--> download script
http://rapidshare.com/files/37562339/lectora.vbs.html

Pero el problema es ke yo necesitaba ke el archivo culpable de la apertura de la lectora sea un ".exe" y no un ".vbs" como lo és el script. Por lo tanto comencé a buscar un programa ke me pase del formato ".vbs" (Visual Basic Script) a un ".exe" (ejecutable común).

Y encontré éste ke se llama "vbs2exe" ke hace ese trabajo, te convierte un script en un exe.

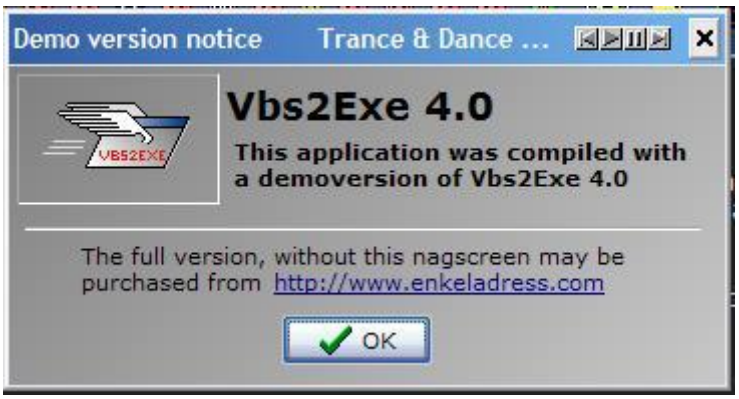
lo pueden descargar de
http://www.freedownloadscenter.com/Programming/IDEs_and_Coding_Utilities/Vbs2Exe_Download.html

hace lo siguiente...

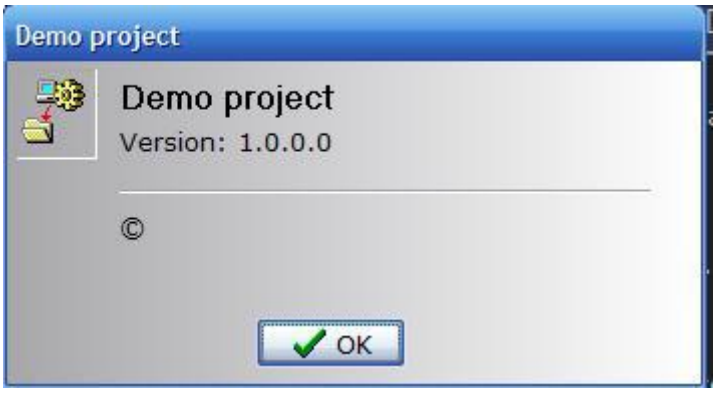


Pero el problema fué ke como el vbs2exe no estaba registrado ni lo pensaba registrar, en cada exe ke creaba me dejaba 2 venanas de mierda con la propaganda del progama y el aviso de ke la version registrada no mostrariá esas 2 nags (venatanas de mierda)...

Osea, yo ya creé el exe, lo ejecuto para probarlo esperando ke solo me abra la lectora, y me encuentro con lo siguiente...



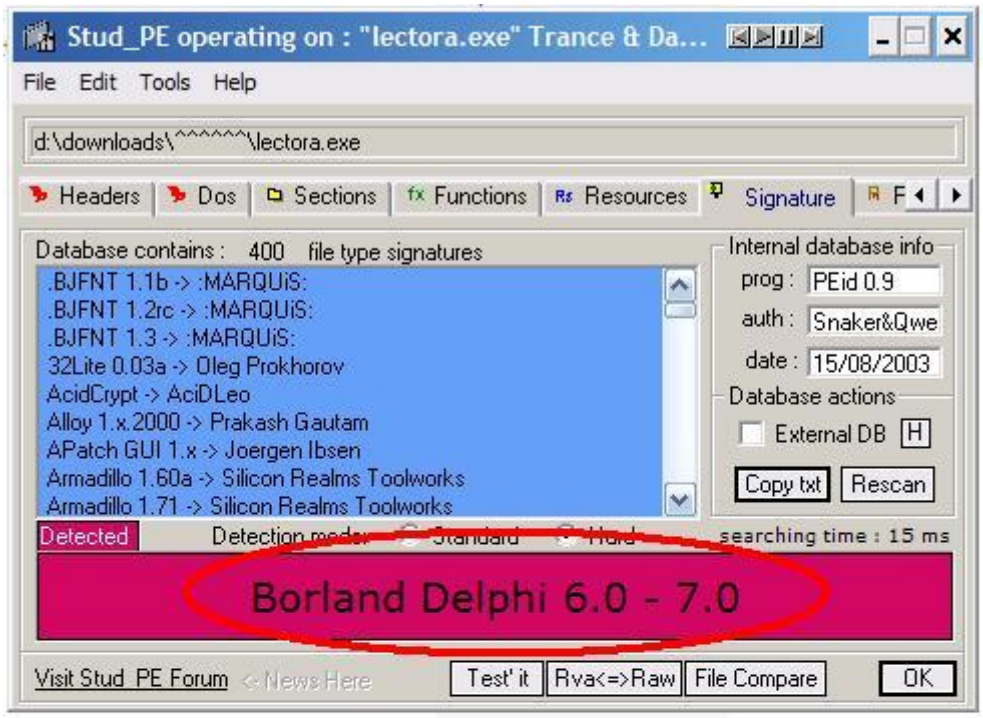
y cuando le doy a "ok" recien ahí me abre la lectora, y luego me muestra ésto...

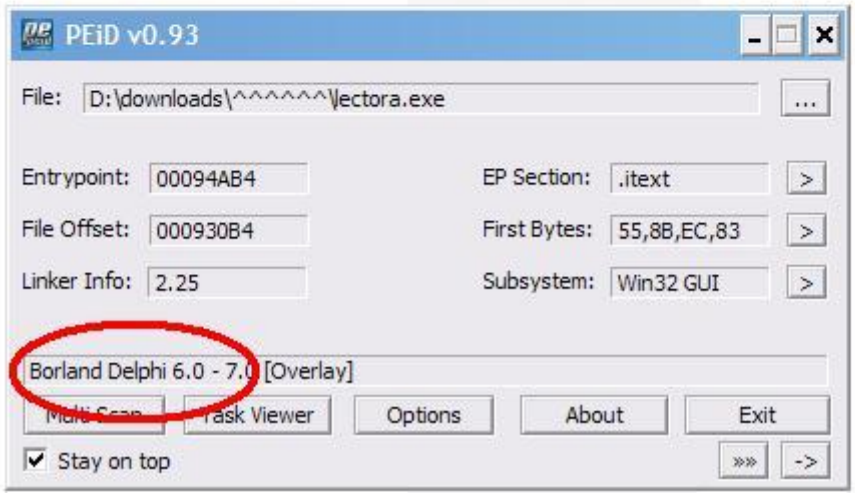


Si, 2 putas ventanas ke me dicen ke la version full bla bla bla... la puta madre ke te re parió...

Entonces llego a el jugoso pensamiento de ke (sabiendo las pocas cosas ke sé de cracking) voy a tener ke crackear el .exe para ke solo me abra la lectora y no me muestre nada, o por lo menos intentarlo.

Tonces agarro el .exe creado con el vbs2exe sin registrsar, y lo escaneo con los 2 analizadores ke uso siempre para ver si esta empaketado o algo de eso...





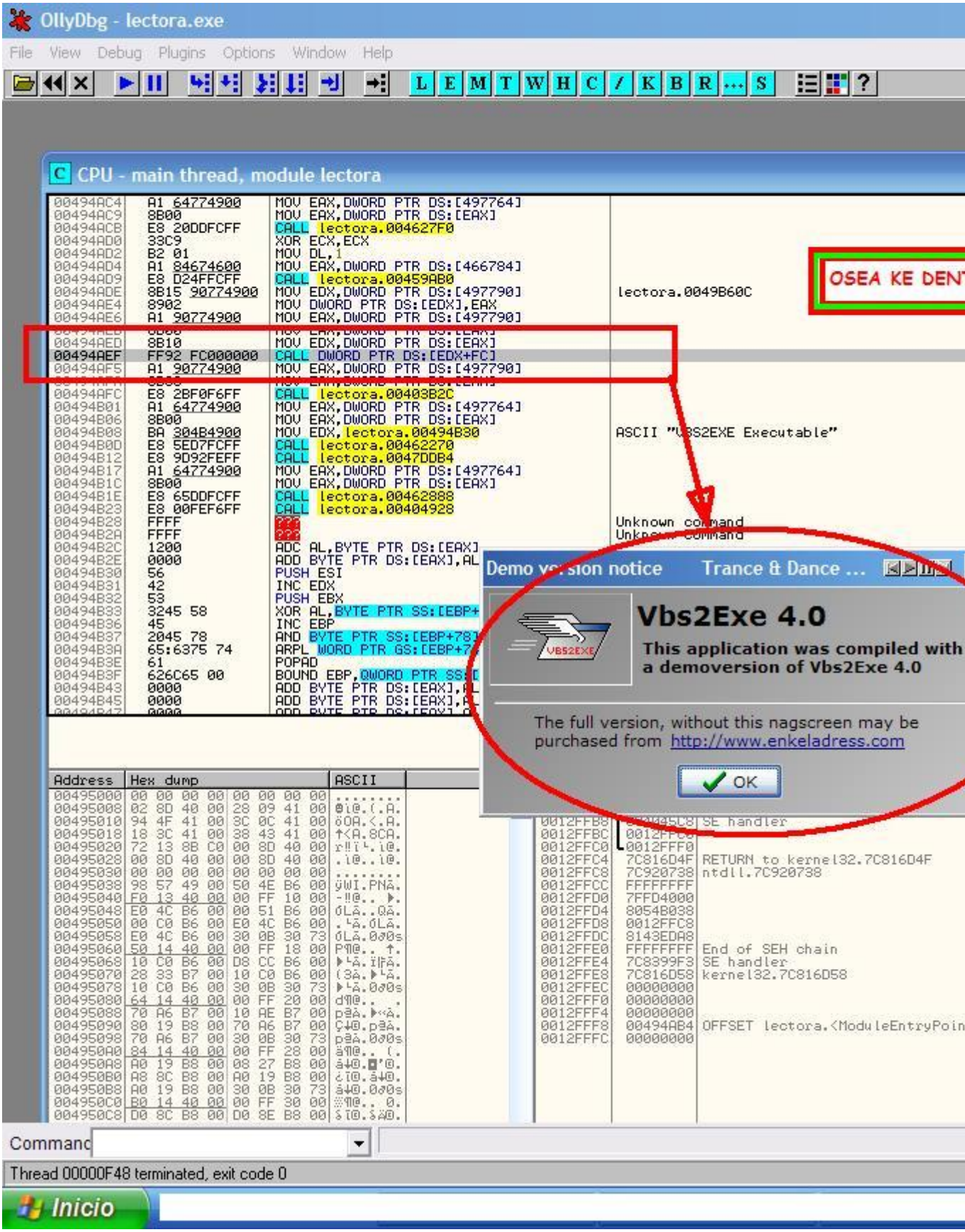
Veo ke los 2 analizadores me muestran lo mismo, ke el .exe está compilado con el Delphi, osea ke no hay empaketado ni nada de eso...

Tonces prosigo desensamblando lo con el w32dasm a ver si encuentro las strings de la maldita venatana...
y no encuentro nada, aparentemente esas strings se crean junto con la ventana durante la ejecucion, o esta compuesta de cracteres sueltos ke en el momento de la ejecucion se juntan o algo por el estilo, la cuestion es ke no encuentro con el w32dasm las strings del cartel de error, es muy comun ke suceda eso...

Que mierda hago?? bueno, abro el debuguer Olly y abro el lectora.exe para rastrear esas venatanas...

Ahora tengo ke rastrear esas 2 venatanas... Voy apretando F8 para ir "traceando" el programa, osea ke voy instruccion por instruccion sin meterme en los call, voy con F8 de a poco viendo los saltos y todo eso hasta ke le doy a F8 sobre un call y se me abre la

ventana...

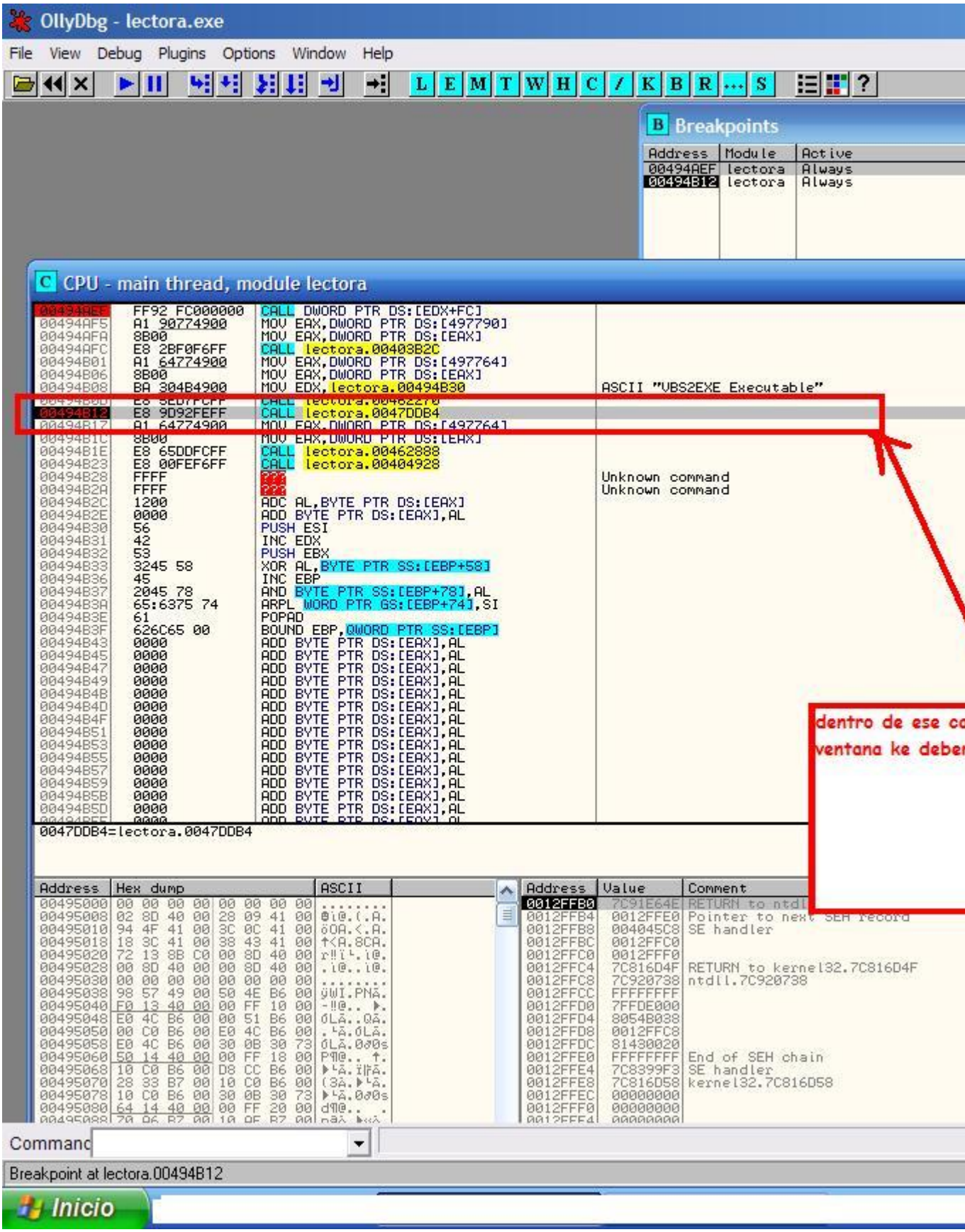

OllyDbg - lectora.exe
File View Debug Plugins Options Window Help
CPU - main thread, module lectora
OSEA KE DENT
Demo version notice
Trance & Dance ...
Vbs2Exe 4.0
This application was compiled with a demoversion of Vbs2Exe 4.0
The full version, without this nagscreen may be purchased from http://www.enkeladress.com
OK
Command
Thread 00000F48 terminated, exit code 0
Inicio

Weno tonces pongo un breakpoint (una marca para ke frene la ejecucion ahí), me paro sobre ese call y apreto F2 y dejo ese break ahí, luego le doy a revovinar con el boton de rewind ke tenemos en el menú, y cuando esta nuevamente listo de doy a "run" osea F9 o el play ke hay en el menú, y perfectamente la ejecucion frena ahí, en ese call donde pusimos el break.
Ahora ke hago ?? weno, necesito llegar a el call o la instruccion encargada de crear esa ventana, osea ke necesito meterme en ese call, cuando el programa frena ahí, le doy a F7 para entrar en ese call (recuerden ke F8 es para tracear sin meterme en los call), y luego continuo con F8 para ver si hay otro call...
Y si, lo hay, asike hago lo mismo, pongo el breakpoint, le doy a revovinar, y a F9 para ke comience denuevo, vemos ke frena en el primer breakpoint ke pusimos, le damos nuevamente a F9 y frena en el segundo break, el último ke pusimos y para ver si ése call es el maldito problema, lo eliminamos, con nops, osea ke lo noopeamos para ke siga la ejecucion del programa sin meterse en ese call, como lo noopeamos? le damos doble clik a el call y se nos abre la ventana "assemble", en ésa ventana borramos lo ke dice y escribimos "nop" en su lugar y apretamos el boton assemble y seguidamente apretamos el boton cancel de esa misma ventanita.

Le damos a "run" para ke continue la ejecucion a ver si ahora se saltea esa ventana... y efectivamente la ventana no aparece y la lectora se abre, sinembargo me aparece la otra ventana.

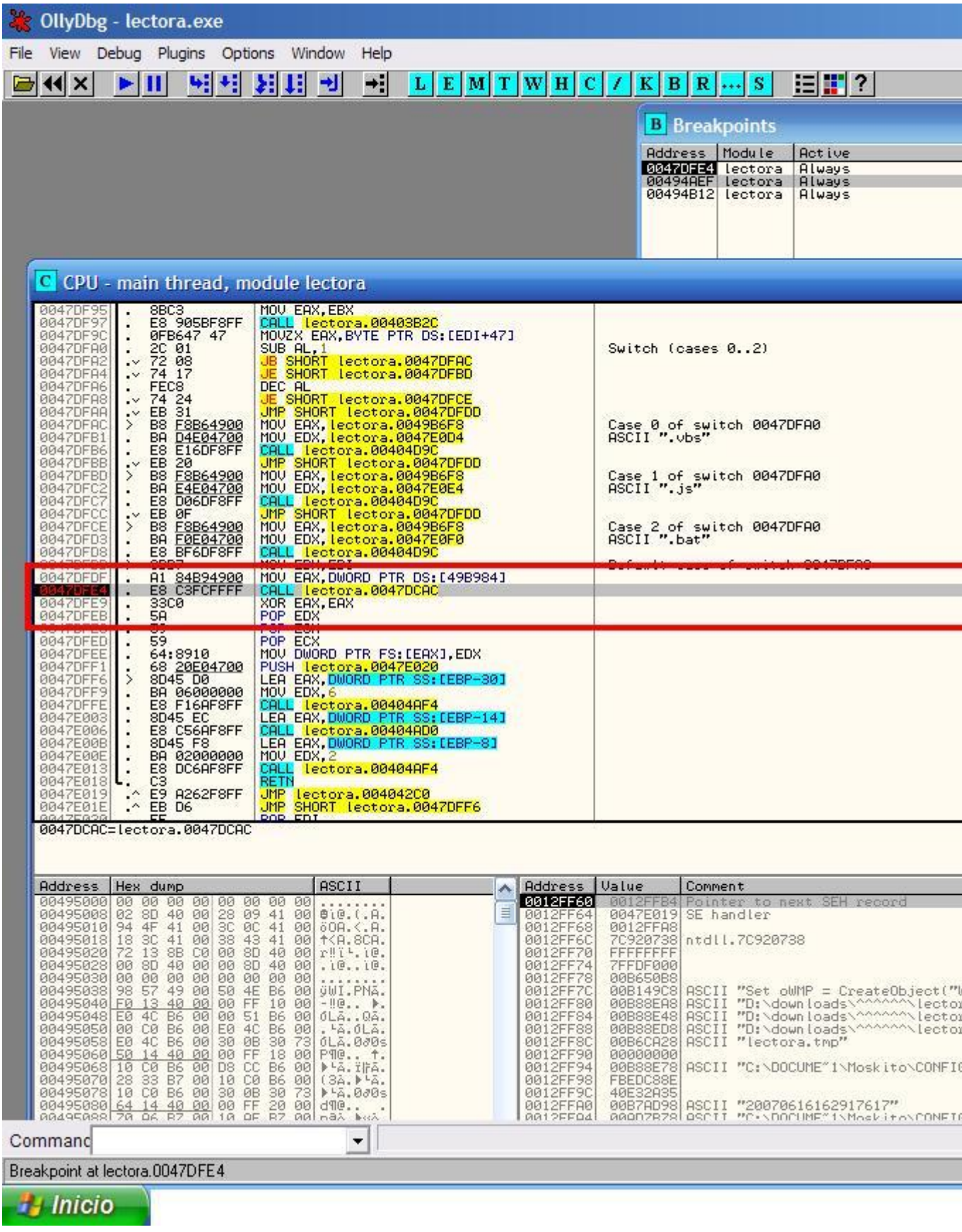
Perfecto, ya descubrimos ke hay un call en la direccion 494aef encargado de hacer la primer ventana, debemos anotar la direccion de esa instruccion para más adelante noopearla, porke el olly no deja grabada la modificacion ke le hemos hecho (amenos ke hagamos algo ke haora no vamos a hacer porke necesitan practicar).

Ahora tenemos ke ir en busca de la ultima ventana, esa puta ventana...
Ponemos un break en el call ke descubrimos y le damos a "run" (revovinamos obviamente sino no va a arrancar la ejecucion), bien, la ejecucion se frena en ése call, weno, aparece la venatana de mierda fabricada en ese call, apretamos el boton "ok" de la venatana y seguimos traceando con F8, porke estamos buscando la segunda ventana... hasta ke llegamos a el siguiente call en el cual se realiza la apertura de la lectora y la puta ventana...



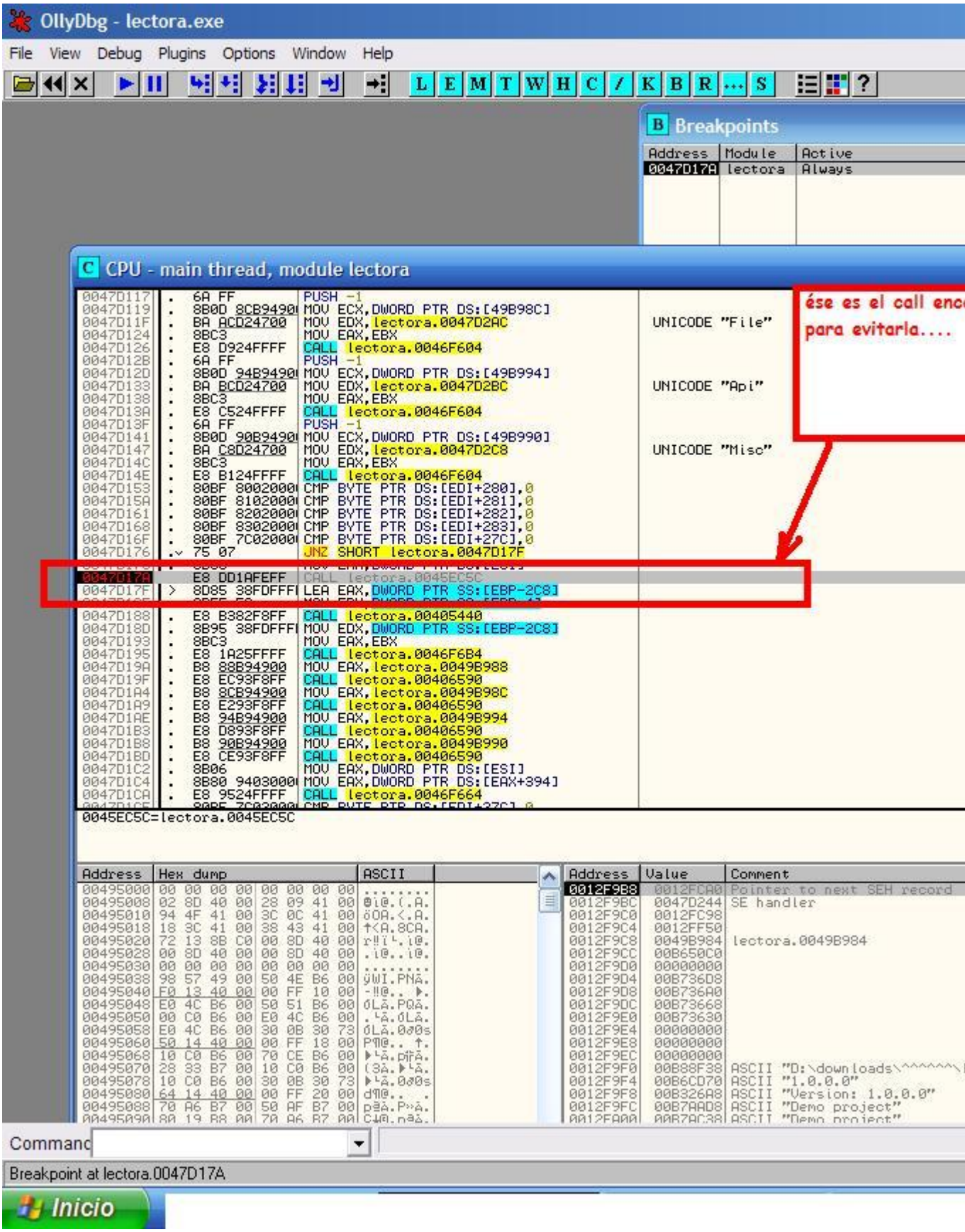
bien, seguimos con la técnica y llegamos a otro call donde sucede lo mismo ke sucedia

en el anterior, osea, se me abre la lectora y me muestra la ventana, todo al mismo timepo, el nuevo call ke hace eso es...

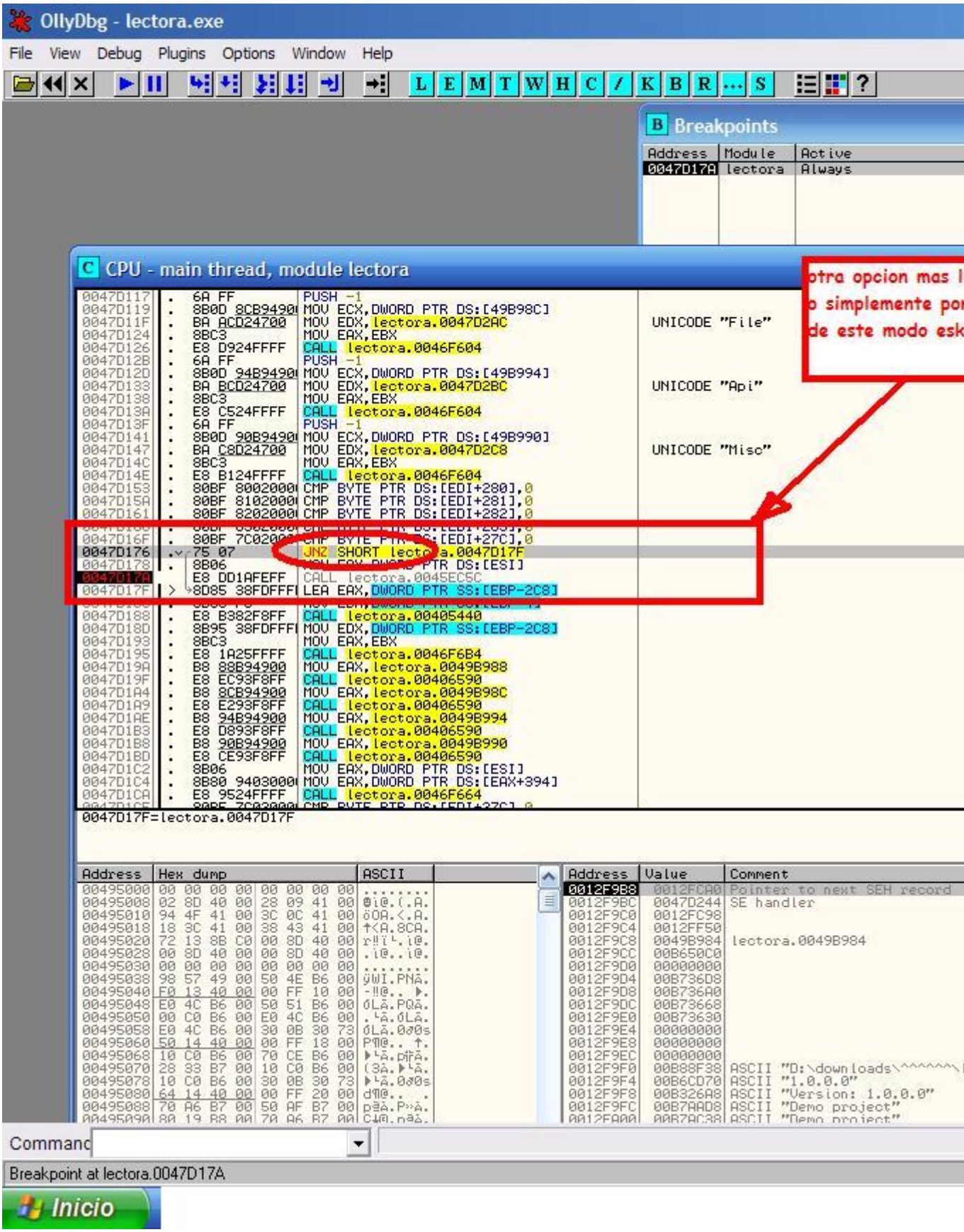


Entonces seguimos aplicando la técnica de poner un break en ese call, revovinar y volver a llegar hasta ése call donde al llegar seguimos traceando con F7 para entrar en él y luego con F8 ver lo ke hay adentro hasta llegar al call responsable de la creacion de la ventana. Hay ke tener paciencia y disfrutarlo... sino no hay ke perder el tiempo...

Luego de aplicar ésta técnica varias veces más, llego a descubrir el call responsable de llamar a la creacion de la ventana. Lo confirmo cuando noopeo y veo ke se abre la lectora y no aparece la ventana. Curiosamente el call ke crea la nag (ventana de mierda) esta antes ke el responsable de abrir la lectora.
y es el siguiente:



otra opcion posible para el crackeo...


OllyDbg - lectora.exe
File View Debug Plugins Options Window Help
Breakpoints
Address Module Active
0047D17A lectora Always
CPU - main thread, module lectora
otra opcion mas
o simplemente por
de este modo esk
0047D117 . 6A FF PUSH -1
0047D119 . 8B0D 8CB94900 MOV ECX,DWORD PTR DS:[49B98C]
0047D11F . BA ACD24700 MOV EDX,lectora.0047D2AC UNICODE "File"
0047D124 . 8BC3 MOV EAX,EBX
0047D126 . E8 D924FFFF CALL lectora.0046F604
0047D12B . 6A FF PUSH -1
0047D12D . 8B0D 94B94900 MOV ECX,DWORD PTR DS:[49B994]
0047D133 . BA BCD24700 MOV EDX,lectora.0047D2BC UNICODE "Api"
0047D138 . 8BC3 MOV EAX,EBX
0047D13A . E8 C524FFFF CALL lectora.0046F604
0047D13F . 6A FF PUSH -1
0047D141 . 8B0D 90B94900 MOV ECX,DWORD PTR DS:[49B990]
0047D147 . BA C8D24700 MOV EDX,lectora.0047D2C8 UNICODE "Misc"
0047D14C . 8BC3 MOV EAX,EBX
0047D14E . E8 B124FFFF CALL lectora.0046F604
0047D153 . 80BF 80020000 CMP BYTE PTR DS:[EDI+280],0
0047D15A . 80BF 81020000 CMP BYTE PTR DS:[EDI+281],0
0047D161 . 80BF 82020000 CMP BYTE PTR DS:[EDI+282],0
0047D16F . 80BF 7C020000 CMP BYTE PTR DS:[EDI+27C],0
0047D176 . 75 07 JNZ SHORT lectora.0047D17F
0047D178 . 8B06 MOV EAX,DWORD PTR DS:[ESI]
0047D17A E8 DD1AFEFF CALL lectora.0045EC5C
0047D17F > 8D85 38FDFFFF LEA EAX,DWORD PTR SS:[EBP-2C8]
0047D188 . E8 B382F8FF CALL lectora.00405440
0047D18D . 8B95 38FDFFFF MOV EDX,DWORD PTR SS:[EBP-2C8]
0047D193 . 8BC3 MOV EAX,EBX
0047D195 . E8 1A25FFFF CALL lectora.0046F6B4
0047D19A . B8 88B94900 MOV EAX,lectora.0049B988
0047D19F . E8 EC93F8FF CALL lectora.00406590
0047D1A4 . B8 8CB94900 MOV EAX,lectora.0049B98C
0047D1A9 . E8 E293F8FF CALL lectora.00406590
0047D1AE . B8 94B94900 MOV EAX,lectora.0049B994
0047D1B3 . E8 D893F8FF CALL lectora.00406590
0047D1B8 . B8 90B94900 MOV EAX,lectora.0049B990
0047D1BD . E8 CE93F8FF CALL lectora.00406590
0047D1C2 . 8B06 MOV EAX,DWORD PTR DS:[ESI]
0047D1C4 . 8B80 94030000 MOV EAX,DWORD PTR DS:[EAX+394]
0047D1CA . E8 9524FFFF CALL lectora.0046F664
0047D17F=lectora.0047D17F
Address Hex dump ASCII
00495000 00 00 00 00 00 00 00 00 ....
00495008 02 8D 40 00 28 09 41 00 @i@.(.A.
00495010 94 4F 41 00 3C 0C 41 00 öOA.<.A.
00495018 18 3C 41 00 38 43 41 00 ↑<A.8CA.
00495020 72 13 8B C0 00 8D 40 00 r‼ï└.ì@.
00495028 00 8D 40 00 00 8D 40 00 .ì@..ì@.
00495030 00 00 00 00 00 00 00 00 ....
00495038 98 57 49 00 50 4E B6 00 ÿWI.PNÂ.
00495040 F0 13 40 00 00 FF 10 00 -‼@.. ►.
00495048 E0 4C B6 00 50 51 B6 00 óLÂ.PQÂ.
00495050 00 C0 B6 00 E0 4C B6 00 .└Â.óLÂ.
00495058 E0 4C B6 00 30 0B 30 73 óLÂ.0∂0s
00495060 50 14 40 00 00 FF 18 00 P¶@.. ↑.
00495068 10 C0 B6 00 70 CE B6 00 ►└Â.pîÂ.
00495070 28 33 B7 00 10 C0 B6 00 (3À.►└Â.
00495078 10 C0 B6 00 30 0B 30 73 ►└Â.0∂0s
00495080 64 14 40 00 00 FF 20 00 d¶@.. .
00495088 70 A6 B7 00 50 AF B7 00 p¦À.P»À.
Address Value Comment
0012F9B8 0012FCA0 Pointer to next SEH record
0012F9BC 0047D244 SE handler
0012F9C0 0012FC98
0012F9C4 0012FF50
0012F9C8 0049B984 lectora.0049B984
0012F9CC 00B650C0
0012F9D0 00000000
0012F9D4 00B736D8
0012F9D8 00B736A0
0012F9DC 00B73668
0012F9E0 00B73630
0012F9E4 00000000
0012F9E8 00000000
0012F9EC 00000000
0012F9F0 00B88F38 ASCII "D:\downloads\^^^^^^\
0012F9F4 00B6CD70 ASCII "1.0.0.0"
0012F9F8 00B326A8 ASCII "Version: 1.0.0.0"
0012F9FC 00B7AAD8 ASCII "Demo project"
Command
Breakpoint at lectora.0047D17A
Inicio

Bien, anotamos esa direccion tambien.

Ahora lo ke tenemos ke hacer es realizar el cambio en el archivo y ke kede fijo. Agarramos las direcciones del codigo ke debemos modificar y abrimos el archivo a cracekar con el w32dasm, buscamos cada una de las direcciones con el w32dasm y nos fijamos abajo, en la bara de estado, al lado de la palabra offset tenemos la direccion ke vamos a buscar con el editor hexadecimal, ésa va a ser la direccion ke vamos a buscar, no la ke nos muestra a la izkierda del desensamblado. Bien, las anotamos a un lado de las ke ya anotamos anteriormente.

Abrimos el editor hexadecimal, buscamos las direcciones y en la primera, la correspondient es al call, debemos poner en cada byte correspondient e a toda la cadena de bytes del call, un 90 ke significa nop en assembler.
Luego vamos a la otra direccion y en donde vemos el byte correspondient e al salto, ponemos un "eb" ke corresponde a un jmp, osea un salto incondicional.
Lo guardamos y listo, ya nos keda el exe sin esas molestas ventanas.

Es simplemente un ejemplo de aplicar el cracking en algo ke surjió en el momento... piensen ke en 5 minutos ya tenia el tema resuelto, parecerá estupido tal vez hacer tantas cosas para lograr lo ke buscaba, pero lo pude hacer graxias a mis pocos conociemientos de cracking, y me satisfizo...

Eso si, luego de haber terminado el programa y de haberlo publicado, resivi frases como "ese programa es lammer, es inutil, de parte de aldebaran_taur us, entre otros" etc, etc... Pero es porke no saben lo ke hay detras de cada aplicacion ke realizo, por suerte puedo experimentar, aprender y hasta compartir con ustedes las cosas ke hago... aunke no les sirva para nada.... jajajaj es así....

y bueno, si no entendieron algo, lo postean, es muy probable ke se hallan mareado ya ke és algo nuevo lo del olly, tengo pensado explicar como crackeé otro de los programas ke publiké utilizando solo el w32dasm y luego voy a continuar explicando algunas cosas del crackeo con el olly, pero recuerden ke tienen ke leer las ultimas cosas ke fuí agrgando, en especial lo de las API.

graxias, denada

continuará

10/Sep/2007 16:40 GMT-3 Perfil · Privado · Desconectado · Web

TARCRO

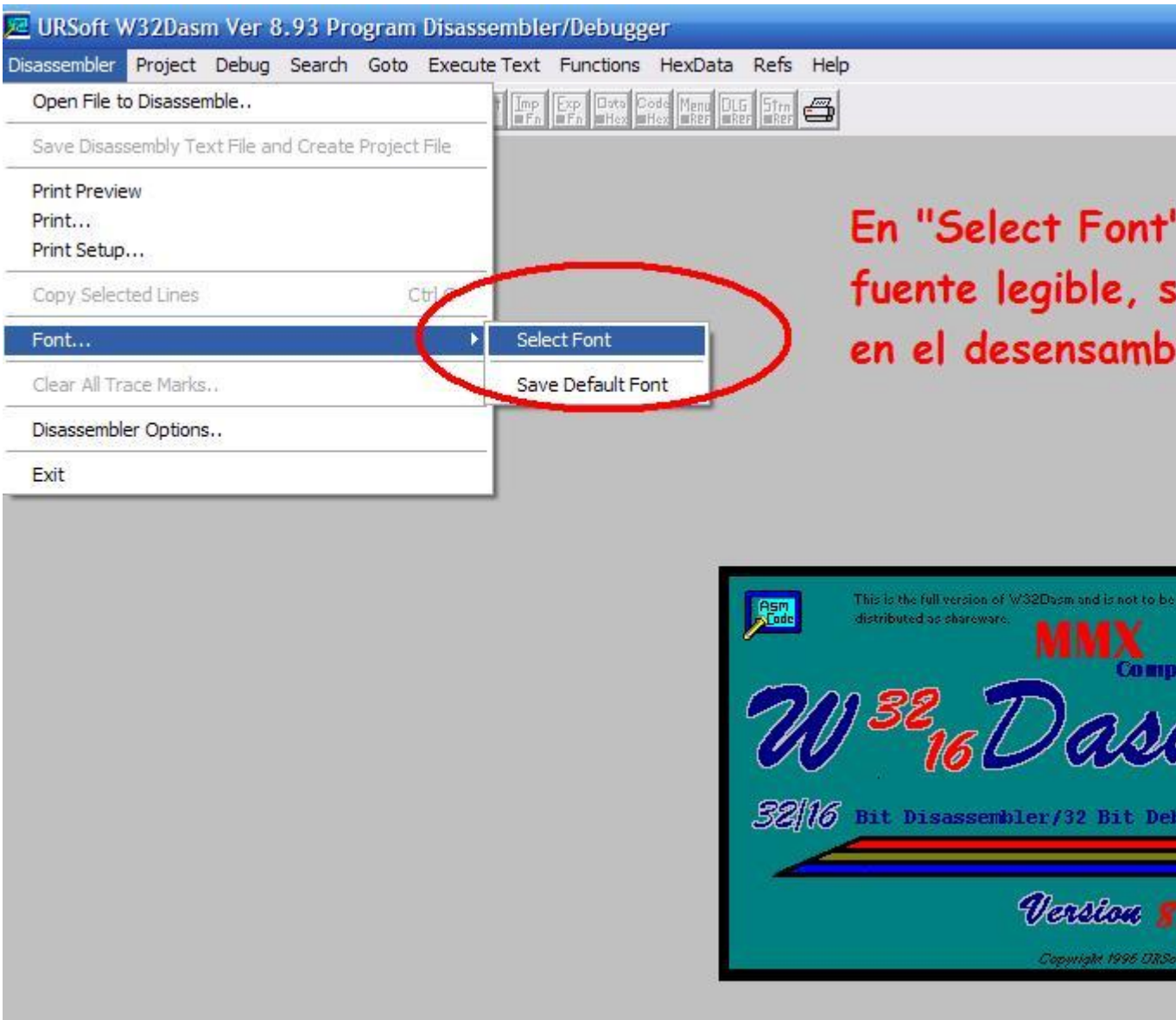
ABAD

Mensajes: **459**
Desde: **22/May/2006**
#6

RE: COMENZANDO A CRACKEAR BY MOSKITO

RESPONDIENDO UNA PREGUNTA TONTA/FUNDAMENTAL HACERCA DEL W32DASM

Resulta ke me han esxrito varios de ustedes preguntando sobre los códigos raros ke aparecen cuando abren por primera vez el w32dasm, resulta ke solo deben configurarlo con la fuente legible ke mejor les paresca, solo éso amigos.
Miren la imagen, ahí les indico donde se configura:



graxias x su atención y disculpen ke publike esto en medio del minimanual, pero lo considero util para muchos... para muchos ke seguramente van a abrir el w32dasm y van a decir "ésta mierda es un despelote, mejor aprendo otra cosa..." jejeje...

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<MOSKITO>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

MAS HERRAMIENTAS PARA NO PERDER EL CONTROL DE LO KE PASA EN LA PC

Bien, todavia no he sacado tiempo para continuar con el tutorial, ya habrán escuchado o leido sobre los monitores de regsitro y de archivos, de ke se trata?

Monitor de archivos:

Espia todo lo ke hace determinado programa con los archivos de la pc, kuales son los

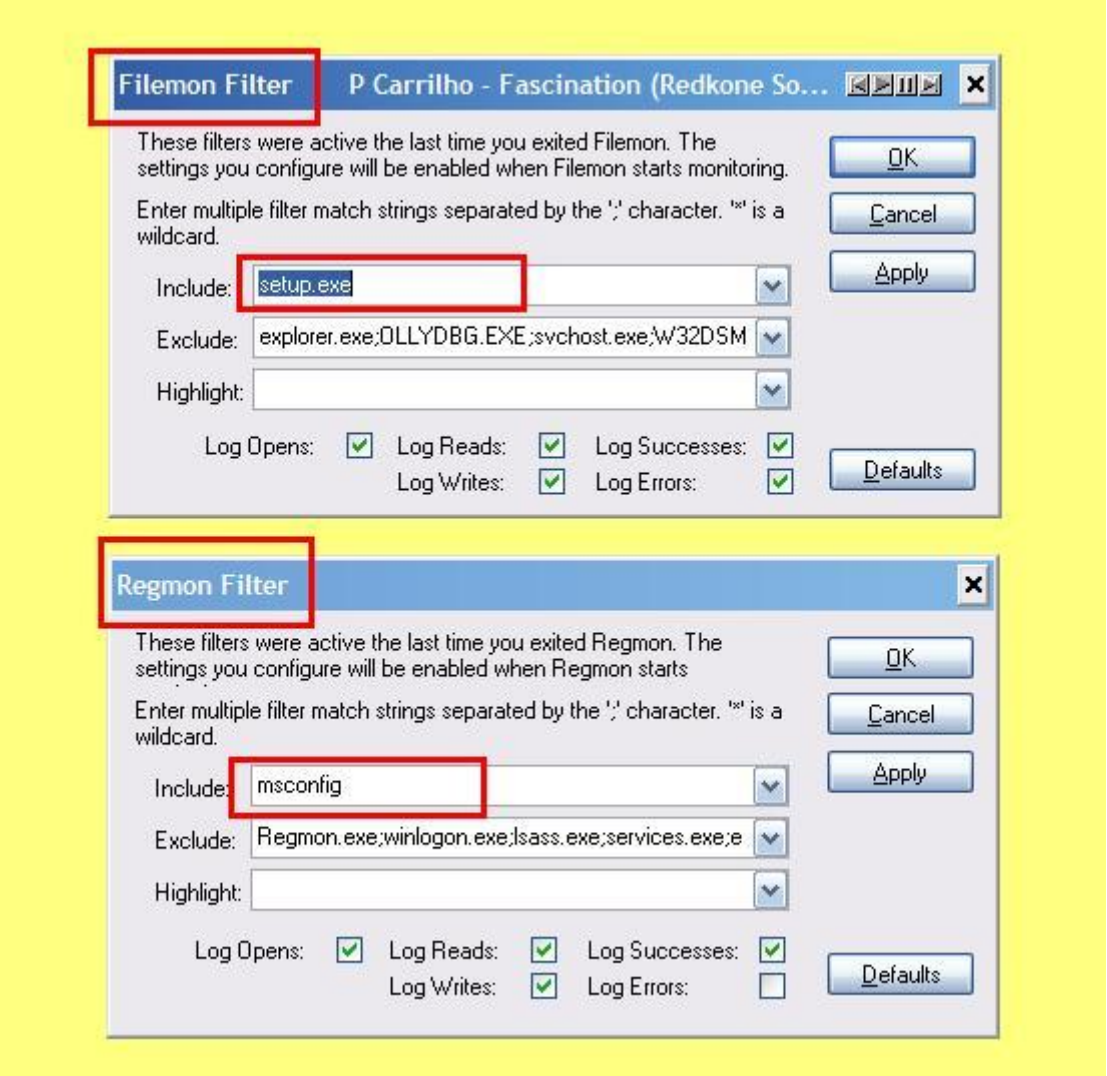
archivos ke modifica, crea o borra, etc. Monitoriza los archivos ke sufren cambios x el programa ke elijamos espiar.

Monitor de Registro:

Espia todo lo ke hace determinado programa con el regsitro de windows, kuales son las keys o claves ke modifica, crea o borra, etc. Monitoriza las keys y/o claves ke sufren cambios x el programa ke elijamos espiar.

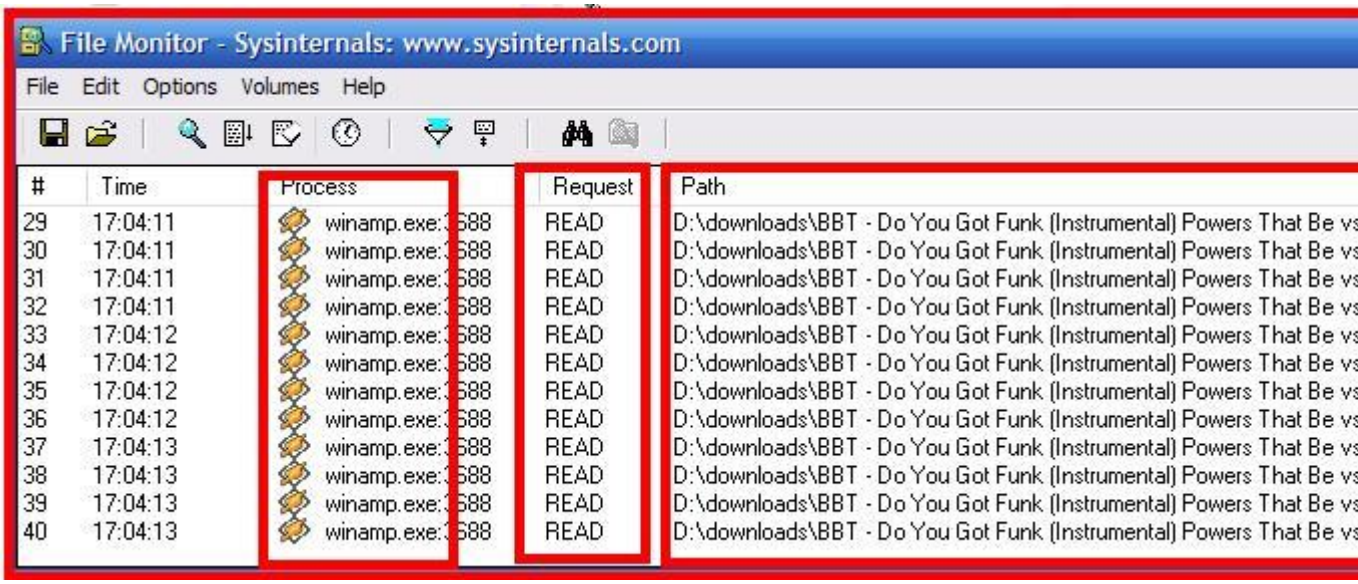
Son muy útiles para descubrir si algun programa esconde alguna clave, contador de tiempo o lo ke sea en algun archivo o registro ke no pensamos ke fuera a utilizar o modificar. Son precisamente para no poerder el control de lo ke sucede en la pc.

Al iniciar cualkiera de los dos, el Filemon o el Regmon, veremos lo siguiente:



Que corresponde al filtro de partida, para filtrar todo lo ke no keremos ke sea monitorizado y para ke en la ventanita marcada pongamos el programa ke keremos espiar.

Y el panel principal es el siguiente (es lo mismo para los dos monitores):



y podemos distinguir el nombre del proceso o procesos ke estamos monitoreando, lo ke hace y a ke archivos o claves con la ruta y todo y si tuvo éxito o no. Hermoso.

DOWNLOAD MONITORES->http://rapidshare.com/files/44238771/Monitores_de_windows.rar.html

denada, espero pronto tener algo de tiempo para continuar con el tuto...

10/Sep/2007 16:41 GMT-3 Perfil · Privado · Desconectado · Web

TARCRO

ABAD

Mensajes: **459**
Desde: **22/May/2006**
#7

RE: COMENZANDO A CRACKEAR BY MOSKITO

MANUAL RÁPIDO DE CRACKEO DE STRINGS (CAMBIAR TEXTO EN PROGRAMAS)

Pasen por acá http://foro.portalhacker.net/index.php/topic,45085.0.html

Eso fué por la insistencia de todos...

Continuará