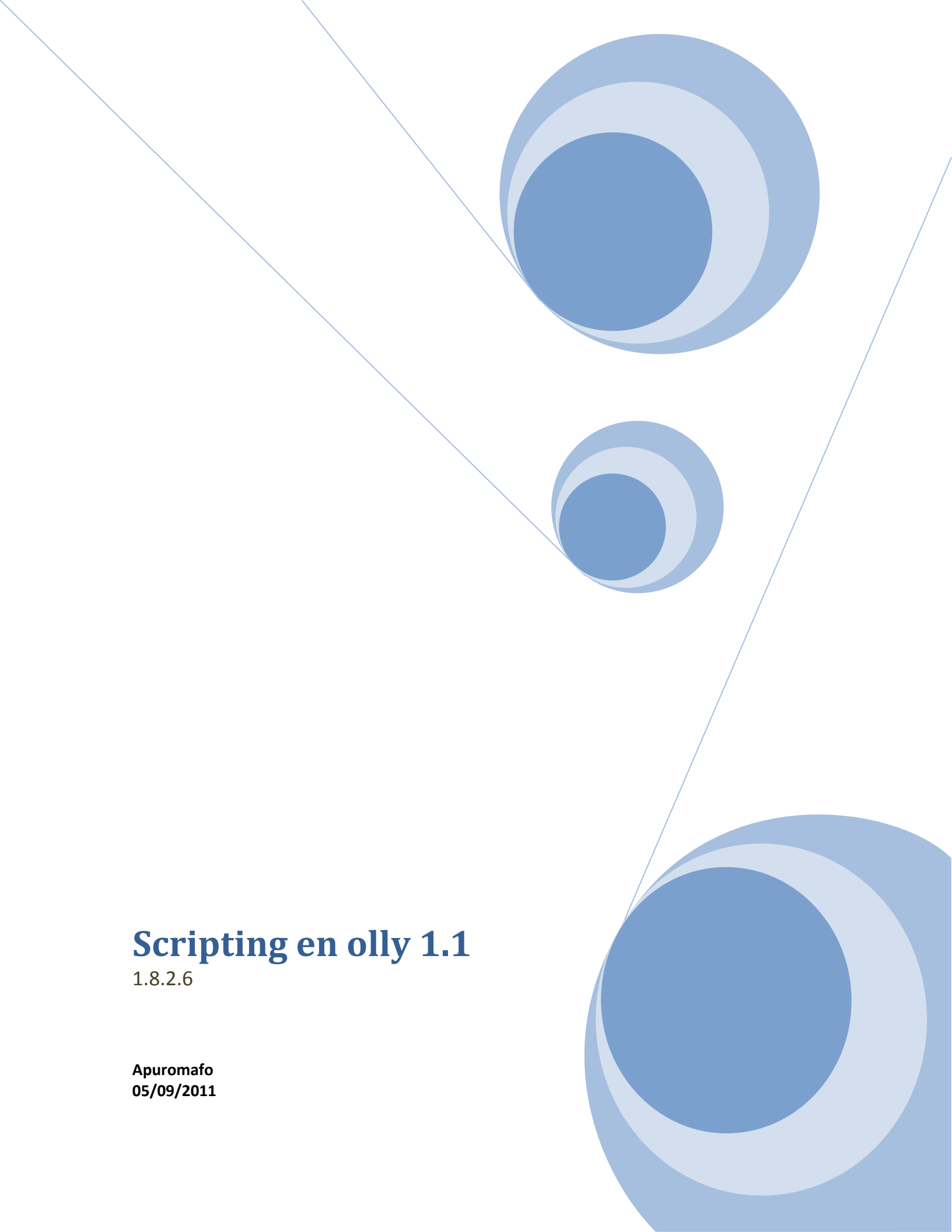


Scripting en olly 1.1

1.8.2.6

Apuromafo

05/09/2011

Como utilizar los script en OllyDbg

Bueno no es extraño saber que existen script, que en parte quieren compartir alguna novedad, alguna maniobra que explicarlo por imágenes puede ser repetitiva o bien para comparar que el procedimiento usado puede ser usado en mas de un equipo

Bueno lo mas básico que debemos saber es que sirve tanto para personas con gran conocimiento como para los newbies o los mas nuevos (aunque no lo sean)

Que es un script.

Entender un script (sea un txt, un osc) es un segment de código escrito para hacer un trabajo dependiendo de las intenciones, accede en forma que uno logra depurar, muchas veces veremos algun programa que tenga muchos antidebug y cueste un poco desempacar luego conforme sabemos un poco mas, lo usamos de aliado para evitar hacer acciones repetitivas.

Porque las personas usan el script?

Obviamente para la gente que jamás ha hecho algún script no es útil, es normalmente usado para manipular tareas repetitivas, tal cual uno lo puede hacer en forma igual "que podría haberlo hecho otro" .Ejemplos comunes serian (puhad/popad), UPX /aspack/,y otros, parece difícil creer que hace un tiempo solo existían packers como estos, y actualmente hay centenas, realmente a la actualidad existen tools, que automatizan este proceso

Tengamos presente que aveces ciertos progresos se olvidan y luego ejecutar estos script le recuerdan como hacerlo, desde 5 minutos a hacerlo en segundos., realmente es guardar un poco ciertos resultados

Caracteristicas de los script:

Podemos decir que quien ha programado o no ha programado, sabe que para compilar un programa en cierto programa debe algún moemento ejecutarlo, el script no esta tan lejos de aquello, tiene un botón de ejecutar y ver paso a paso que esta pasando y se detiene en un posible error.

Instalacion del plugin:

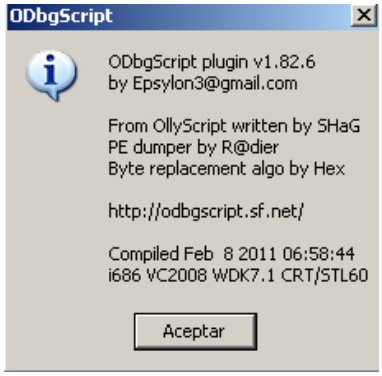
Se puede descargar desde: http://odbgscript.sf.net/ o encontrarlo desde foros ingleses como tuts4you.com , después de descargar la copia, tenemos un archivo ODbgScript.dll el cual debe colocarse en la carpeta de plugins de Olly, luego al reiniciar, la instalación estará completa.

Plugin para ejecutar script

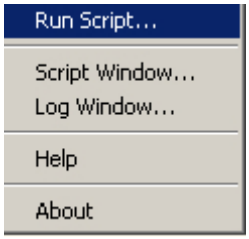
Por ciertos años se uso ollyscript escrito por SHaG, el cual fue modificado varias veces por otros autores y luego un coder Epsylon pues reimplemento todo usando reemplazamientos hexadecimales, pe dumpers y otros , por lo cual no es solamente colocar bp, actualmente puede escribir en archivos, leer archivos , comparar variables, convertir variables, todo dependiendo de la habilidad que logremos con esto.

El programa o plugin a agregar a ollydbg es OdbgScript, el cual en la versión 1.1 esta alojado en SF, y el de la versión 2.x de olly esta en etapa de desarrollo por usar valores Unicode (al contrario del otro que usaba ascii), y en cuanto al uso, deberíamos colocarlo en extensión *. txt y *. OSC

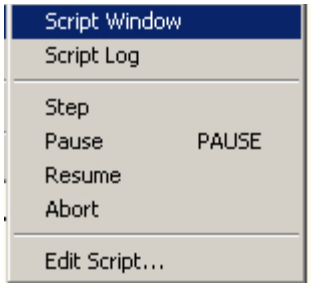
Acerca del plugin:



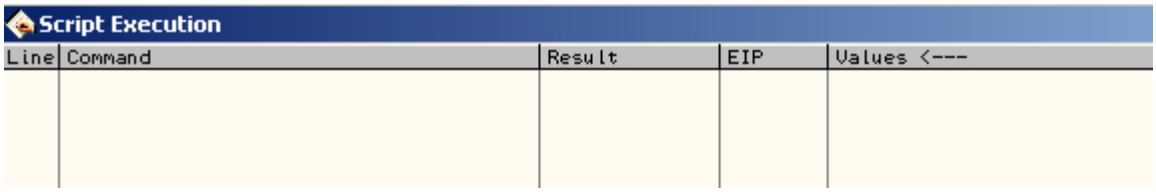
Menus del plugin desde la parte superior(toolbar/:



Desde la parte inferior (cpu/main tread):



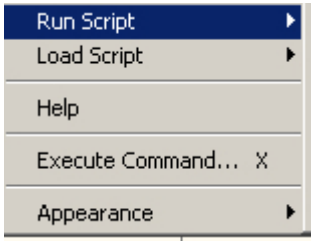
Menú depuración de script/de ejecucion:



Usar los script.:

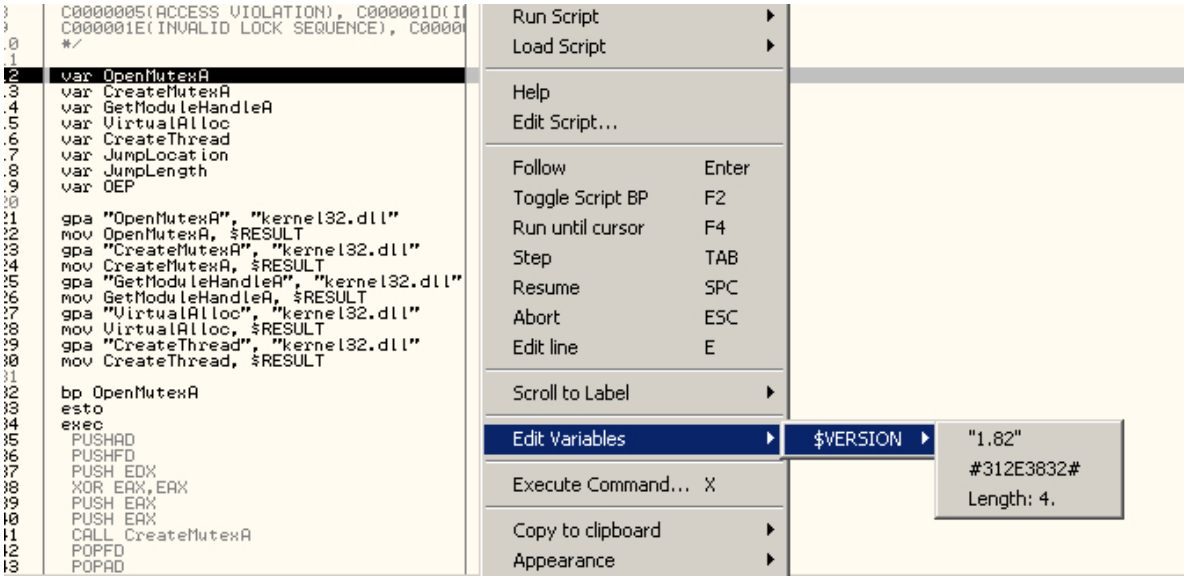
Ya tenemos en el menú de ollydbg, la opción de ejecutar el script (run script), luego en los menus contextuales de la ventana de código, donde podemos detenerlo, abortar, y otros.

Pero además existe otro menú en la ventana misma del scripting



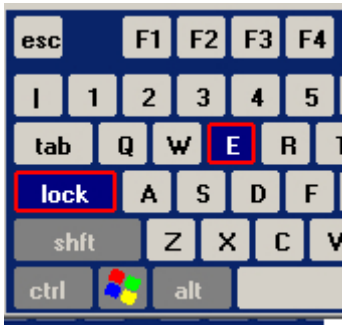
Cuando queremos ejecutar un script, debemos colocar Run script, hacemos el click en la opción, y aparecerá un OPEN, el cual esperara un tiempo para completar las actividades.

Luego de cargado mostrara un menú similar a este:



Donde la primera variable es la variable $VERSION, Para verificar que utilizen esta versión.

vemos como en esta ventana se pueden usar comandos para editar la línea (E)

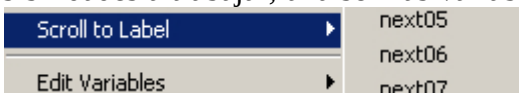


,

paso a paso (TAB/tabulación)

,

Y otras mas.

Trabajo del script en 1.82 Existen 2 posibilidades a trabajar, una son las variables y otros los labels. Y podemos saber los labels: 

Las variables son los valores, y los labels, son los nombres que tenemos ciertos comienzos/terminus de rutina

Para declarar una variable

Var variable ,
pero últimamente para evitar gran procedimiento de variables, también es permitido declarar una variable usando mediante un mov con valor o mov con labels inclusive con $RESULT,
ejemplo:
Mov nombrevariable,0
Mov label, $RESULT //
Mov label2, label

Lo mas básico a comentar en el scripting, es declarar una variable, realizar una acción y guardar el resultado en $RESULT, el cual puede ser transferido a otra variable.

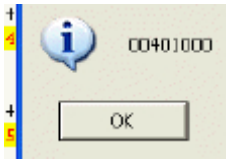
Este ejemplo proviene de At4re:

codebaseshow.txt - Bloc de notas

Archivo Edición Formato Ver Ayuda

```
//This is an example of retrieving the codebase and display it in a msgbox.
var cbase //declares the variable cbase
GMI eip, CODEBASE // Now $RESULT is the address to the codebase
MOV cbase, $RESULT //moves the result to our variable
msg cbase //Msgbox with the value
ret //End script
```

El mensaje puede mostrar un valor de codebase



Ejemplo personal:
Diferencias en el manejo de los string:

Diferencias importantes:
Mov valor1,61
Mov valor2,#61#
Mov valor3,"61"
/* notese es " y no ” */
Mov valor4,eip
Mov valor5,[eip]
RET //**final del script**
Visto desde la ventana:

```
Mov valor1,61
Mov valor2,#61#
Mov valor3,"61"
/*  notese es " y no ö      */
Mov valor4,eip
Mov valor5,[eip]
RET
```

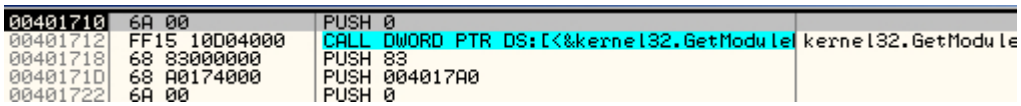
Resultados:

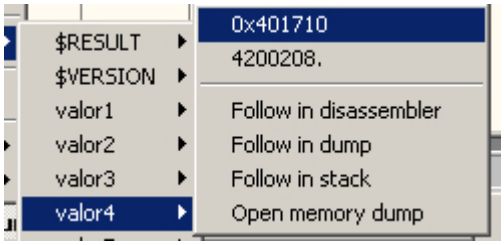
valor1	0x61
valor2	97.

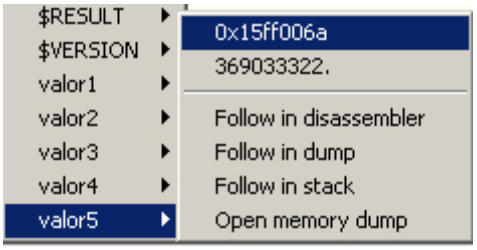
valor2	"a"
valor3	#61#
valor4	Length: 1.

valor3	"61"
valor4	#3631#
valor5	Length: 2.

Es 0x61 decimal 97 ,el segundo es como letra “a” #61# , el tercero es como “61”->#3631#





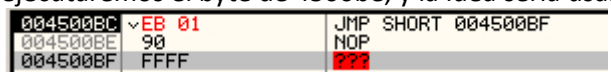


Ver valores de variables
Msg //envía un mensaje
Log //envía a log
Wrt/Wrta/escribe cierto valor en un archivo

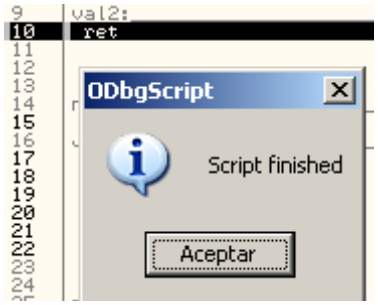
TAMAÑOS
Método de buscar y nopear(usando fill) 2 bytes en un salto estilo junk,:

```
mov offset,eip
val1:
findop offset,#EB01#
cmp $RESULT,0
je val2
fill $RESULT,3,90
mov addr,$RESULT
jmp val1
val2:
ret
```

Aclaracion: si hago un nop de 2 bytes, eliminare el EB01(jmp 4500bf) , pero salta +1, por lo cual ejecutaremos el byte de 4500be, y la idea seria usar el 4500bF



Recordemos cuando termina estamos avisados:



Este es el comienzo de revisión, es posible hacer muchas otras cosas más,
Dejo aquí el escrito para no herir sensibilidades Espero se animen a complementar este escrito
He revisado armadillo, asprotect, y otros packers y mediante el scripting es donde mas tiempo he guardado, mientras el resultado de hacerlo a mano era igual), el tema es como optimizarlo.
Ahora bien, con solo conocer el menú es cosa de explorar, este escrito es solo como ayuda de memoria pues ya a la fecha aun siguen agregándose comandos y variables.

Saludos Apuromafo

Pd: siempre antes de ejecutar un script, es bueno intentarlo uno, codificarlo y luego verificar saludos