

# . : [ CrackSLatinoS ] : .

[[ The Art of Reverse Engineering ]]

Software	Arkanoid Space Ball
Web	<a href="http://www.wegroup.org/games/arkanoid-games/arkanoid-space-ball.html">http://www.wegroup.org/games/arkanoid-games/arkanoid-space-ball.html</a>
Tools	Ollydbg 1 y 2
Fecha	25/05/10

## [ Introducion ]

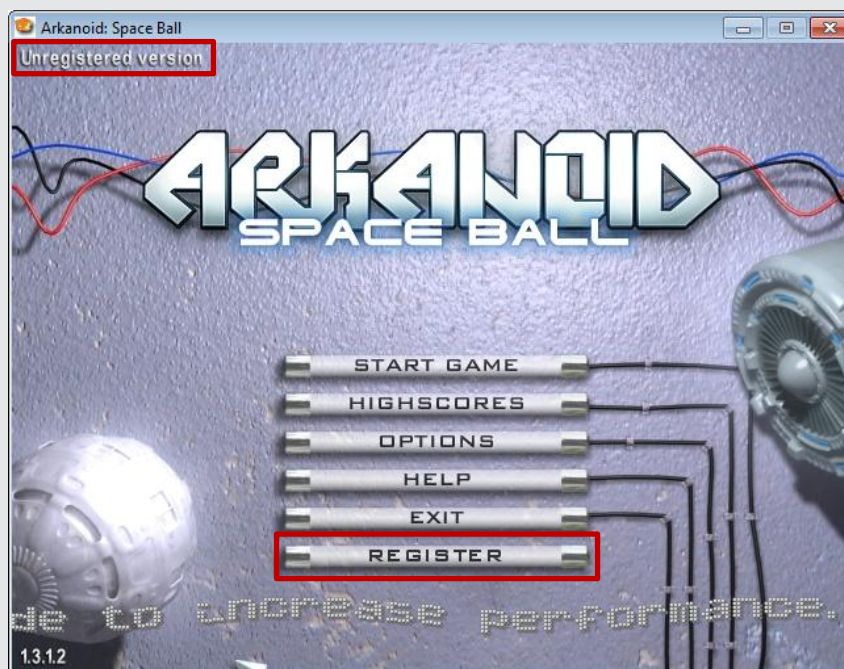
En este mi primer tutorial, veremos el análisis y crackeo de un videojuego, para ello he buscado un juego comercial, *Arkanoid Space Ball*, no tan famoso y claro sin muchas protecciones que generalmente son los desarrollado por Grupos Indie, pero nos servirá para coger bases para el crackeo de otros juegos con protecciones mas complicadas.

## [ Analisis PE ]

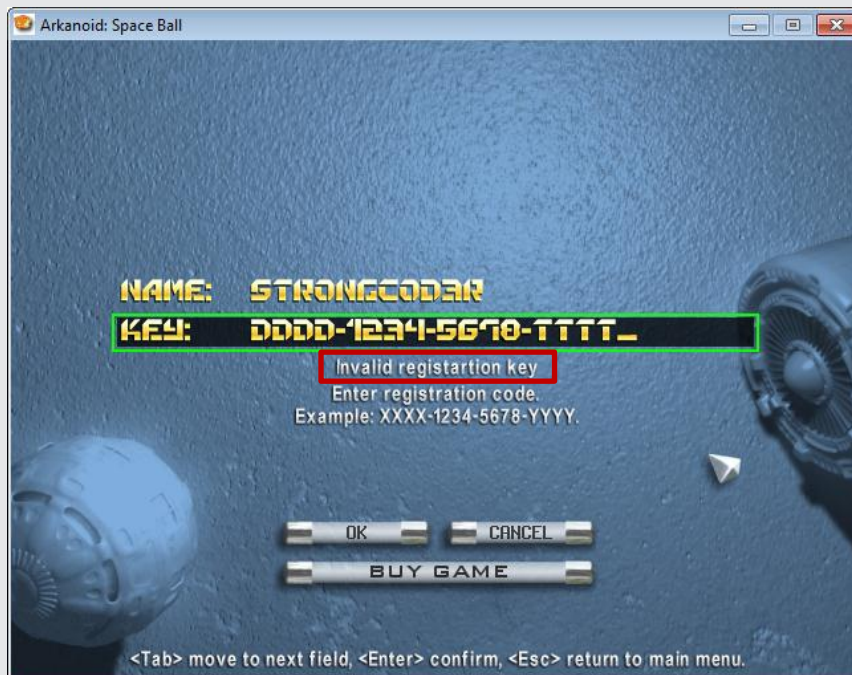
PeiD.95	Nothing found *
Detect iT Easy	Microsoft Visual C++ x.x
ExeInfo PE	Dev-C++ Compiler v4.9.9.? ( MINGW 32 vx.x.x )
FastScanner 3	MingWin32 vx.x
RGD Packer Detector 0.6.6	MingWin32 GCC 3.x
	MingWin32 Compiler

## [ Primer Contacto ]

Instalamos el juego y lo ejecutamos, y observamos esto:



Vemos que en la parte superior izquierda nos muestra el mensaje “Unregistered versión”, damos clic en “REGISTER” y probamos a meter el nombre y el serial y damos ok:



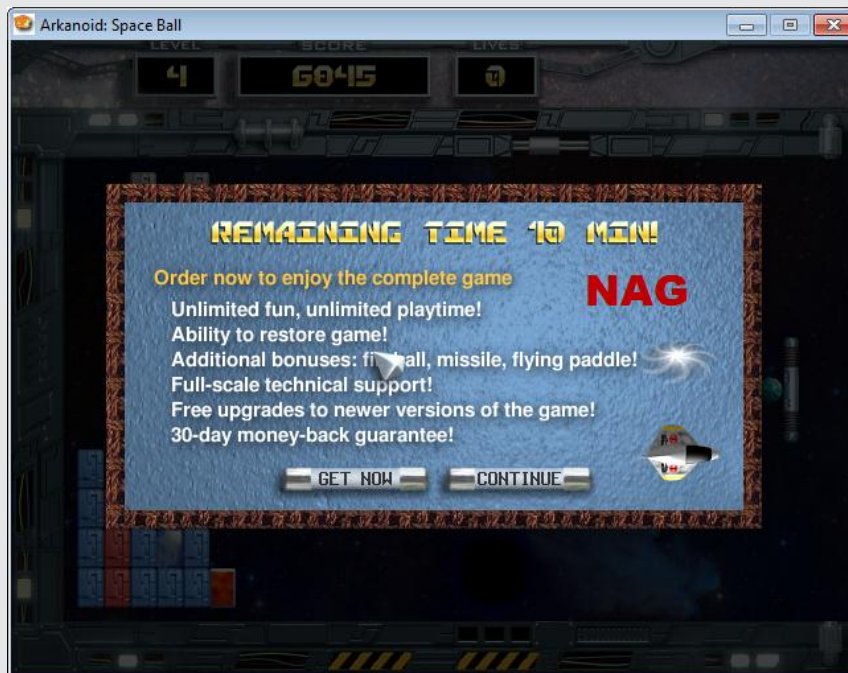
Como vemos nos tira un mensaje “**Invalid registartion key**” diciendo que la clave no es válida, bueno ya tenemos dos pistas por donde buscar:

- **Unregistered versión**
- **Invalid registartion key**

**NOTA:** Al introducir el fake serial el juego solo detecta teclados en Ingles, bug del juego, así que tuve que cambiar la configuración de mi teclado de Spanish a English en panel de control del S.O.

Me puse a jugar un poco para ver si el juego nos daba algunas sorpresitas, cuando empezamos a jugar vemos esto:





Como vemos hay muchas NAG's, el autor del juego busca de todas las maneras de que compremos el juego, bueno con esto ya estamos listos para crackearlo.

## [ Manos a la Obra ]

Abrimos el “**arkanoidsb.exe**” con Olly hacemos lo típico, clic derecho Search for → All Referenced strings y ponemos un Bp(*BreakPoint*) en las string que vimos atrás

Olly al correr el juego observaran el fullscreen del juego y se bloquea el juego a causa de los Bp puestos y no hay como minimizar ni nada ya que el juego está en primer plano, probando Ctrl + Alt + Supr para mostrar Administrador de tareas e intentar finalizar el juego, yo uso Win7 y a diferencia de WinXP crea una capa entre la aplicaciones y un menú para invocar al Administrador de tareas sin que el juego interfiera y poder finalizarlo.

Y ahora que hacemos para poder debugearlo si el juego se queda bloqueado y no nos deja hacer nada, una solución sería ir en el juego Menu→ Options poner al “**Fullscreen Mode = NO**” y problema solucionado:





Pero eso sería lo más fácil a hacer y pensando que no todos los juegos tienen esa opción, así que vamos hacer este tutorial más interesante y solucionar el problema desde Olly.

La pregunta es cómo hacerlo, así que pongámonos a pensar desde lo más básico, que se necesita para programar videojuegos:

- Librería Grafica
- Librería Sonido
- Librería Física
- etc

Creo que la más importante es la librería Grafica ya que controla todo lo visual en el juego como primitivas, imágenes, fonts, Modos de Video etc. Así que cargamos de nuevo el juego en Olly y nos dirigimos a View → Log (Alt + L):

```
OllyDbg v2.00 (intermediate version - under development!)
File 'C:\Program Files\Arkanoid Space Ball\arkanoidsb.exe'
New process (ID 00001EC8) created
Main thread (ID 00001200) created
004012A0 Module C:\Program Files\Arkanoid Space Ball\arkanoidsb.exe
00400000 Module C:\Program Files\Arkanoid Space Ball\SDL_image.dll
002C0000 Module C:\Program Files\Arkanoid Space Ball\libpng13.dll
002E0000 Module C:\Program Files\Arkanoid Space Ball\zlib1.dll
00320000 Module C:\Program Files\Arkanoid Space Ball\SDL_mixer.dll
10000000 Module C:\Program Files\Arkanoid Space Ball\SDL.dll
64F40000 Module C:\Program Files\Arkanoid Space Ball\jpeg.dll
73310000 Module C:\Windows\system32\WINMM.dll
```

Vemos que Olly ha cargado algunas librerías dinámicas, ósea las que el juego va a utilizar en ejecución y entre ellas vemos las de nombre SDL:

- SDL.dll ← Librería Grafica principal
- SDL\_image.dll ← Extensión de SDL para soporte de formato imágenes
- SDL\_mixer.dll ← Extensión de SDL para soporte de formatos de sonido

La Librería SDL([Simple DirectMedia Layer](#)) es una librería muy conocida para el desarrollo de juegos 2D, hay que tener en cuenta que no es la única, hay muchas más, aquí un listado de las conocidas en el desarrollo de videojuegos:

- DirectX,
- OpenGL,
- Allegro,
- IrrLich,
- Ogre,
- IndieLib,
- ClanLib,
- XNA.Net

Seguimos. En todas las anteriores librerías antes comentadas hay una API que tienen en común todas ellas, es la de establecer el modo de video en el juego. Viendo la documentación de la librería de SDL ([Link](#)) encontramos el API que se encarga de establecer el modo de video del juego, aquí el prototipo de la API:

```
// Para mas flags ver documentacion
//Flag: SDL_FULLSCREEN = Pantalla completa

SDL_Surface* SDL_SetVideoMode(int width,      //Ancho de la ventana
                              int height,     //Alto de la ventana
                              int bitsperpixel, //Profundidad de color
                              Uint32 flags);   //Flags
```

Aquí un ejemplos de cómo usarla:

(1)

```
// Juego en FullScreen(pantalla completa)
SDL_SetVideoMode( 1024, 768, 32, SDL_FULLSCREEN);
```

(2)

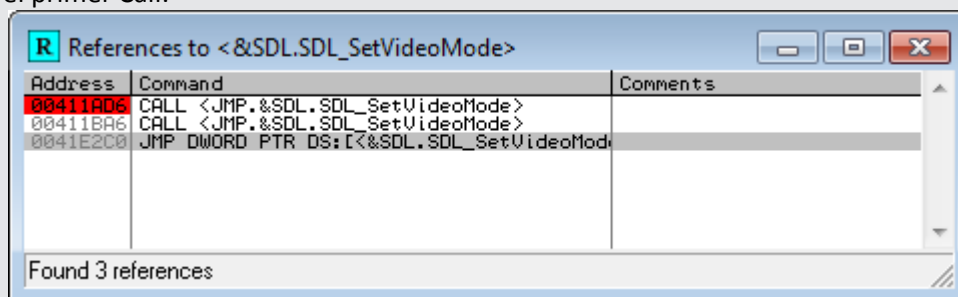
```
// Juego en WindowMode(modos ventana)
SDL_SetVideoMode( 800, 600, 32, SDL_SWSURFACE);
```

Como vemos actualmente el juego se ejecutaría como el ejemplo 1, lo que necesitamos es que se ejecute como el ejemplo 2, así que cuando Olly llame a esas API le cambiamos los parámetros para que se ejecute en pantalla normal, pero como sabemos qué valor tiene cada flag, la respuesta a eso podríamos tenerla si hiciéramos un pequeño ejemplo en C/C++ con SDL con eso parámetros y debugearlo con Olly, pero sería muy lioso, ya que tendríamos que bajar la librería SDL, configurar las librerías con el compilador etc, ese método es recomendado para cuando no tenga el código fuente de las librerías como DirectX, también podríamos intentar meterle algún numero cualquiera por probar, pero como quiero hacer las cosas bien vamos a lo seguro, así que lo que aremos es leer el código fuente de SDL(libre distribución) y ponernos analizar donde tiene definidos esos flags y ver el valor, entonces vamos alla.

Después de bajarme la librería, la descomprimimos y vemos una tonelada de carpetas, pero dónde buscar?, normalmente estos flag en C/C++ se saben definir en las header .h, no siempre, así que vamos a la carpeta **include** y abrimos el archivo **SDL\_video.h** y vemos en las líneas **131-155** definidas las constantes con sus valores y vemos que nuestro flag tiene de valor:

SDL\_FULLSCREEN = 0x80000000  
 SDL\_SWSURFACE = 0x00000000 // Vale 0, tanta búsqueda para esto ☺

Como podemos ver nuestro flag vale 0, ahora si ya estamos listos, cargamos el juego en Olly y damos clic derecho Search For→Name y ponemos damos clic derecho en la Api **SDL.SDL\_SetVideoMode**→**Find References** y vemos 2 Call, ponemos un Bp en el primer Call:



El segundo Call lo usa el juego cuando cambiamos la configuración en Menu→Options dentro del juego y aplica los cambios, seguimos, damos al botón Run y para aquí:

00411A90	55	PUSH EBP	
00411A91	89E5	MOV EBP,ESP	
00411A93	57	PUSH EDI	
00411A94	56	PUSH ESI	
00411A95	53	PUSH EBX	
00411A96	83EC 3C	SUB ESP,3C	
00411A99	A1 30B34300	MOV EAX,DWORD PTR DS:[43B330]	
00411A9E	8B7D 10	MOV EDI,DWORD PTR SS:[EBP+10]	
00411AA1	8B75 14	MOV ESI,DWORD PTR SS:[EBP+14]	
00411AA4	85C0	TEST EAX,EAX	
00411AA6	0F84 84030000	JE 00411E30	
00411AAC	A1 20B34300	MOV EAX,DWORD PTR DS:[43B320]	
00411AB1	85C0	TEST EAX,EAX	
00411AB3	0F85 4A030000	JNE 00411E03	
00411AB9	F7C6 00001000	TEST ESI,00100000	
00411ABF	75 33	JNE SHORT 00411AF4	
00411AC1	897424 0C	MOV DWORD PTR SS:[ESP+0C],ESI	
00411AC5	897C24 08	MOV DWORD PTR SS:[ESP+08],EDI	
00411AC9	8B45 0C	MOV EAX,DWORD PTR SS:[EBP+0C]	
00411ACC	894424 04	MOV DWORD PTR SS:[ESP+04],EAX	
00411AD0	8B55 08	MOV EDX,DWORD PTR SS:[EBP+08]	
00411AD3	891424	MOV DWORD PTR SS:[ESP],EDX	
00411AD6	E8 E5C70000	CALL <JMP.&SDL.SDL_SetVideoMode>	Jump to SDL.SDL_SetVideoMode

Vemos a la API **SetVideoMode** que se encuentra en el librería **SDL.dll**, la api va hacer llamada con el Call, tecleamos F7 para entrar en el Call, luego miramos al Stack y vemos algo como esto:

0022F89C	C00411ADB	+A.	RETURN from SDL.SDL_SetVideoMode to arkanoidsb.00411ADB
0022F8A0	00000280	00..	
0022F8A4	000001E0	00..	
0022F8A8	00000020	0...l	
0022F8AC	C0000001	0...l	
0022F8B0	00000001	0...l	
0022F8B4	77122920	l#w	
0022F8B8	7709C620	\$.w	RETURN from msvcrt.7708987B to msvcrt.7709C620
0022F8BC	89105E00	0U>e	

Intepretamos esto poniendo comentarios y pasando los valores a decimal(sufijo d):

Address	Value	Comments
0022F89C	00411ADB	; RETURN from SDL.SDL_SetVideoMode to arkanoidsb.00411ADB
0022F8A0	00000280	; 640d Ancho
0022F8A4	000001E0	; 480d Alto
0022F8A8	00000020	; 32 bits Profundidad de color
0022F8AC	C0000001	; Valor con la suma de Flags – FullScreen

Como vemos nos haría cambiar el valor de la address de los flags poniéndolo a "0" para que se inicie en WindowMode así que lo modificamos, para ello damos doble click en Value y podemos modificar:

Address	Value	Comments
0022F89C	00411ADB	; RETURN from SDL.SDL_SetVideoMode to arkanoidsb.00411ADB
0022F8A0	00000280	; 640d Ancho
0022F8A4	000001E0	; 480d Alto
0022F8A8	00000020	; 32 bits Profundidad de color
0022F8AC	00000000	; Valor con la suma de Flags - WindowsMode

Listo ahora si damos F9 y vemos que inicia en ventana normal podemos poner un Bp y cualquier lado del juego y vemos que ya no tendremos los problemas del bloqueo.

Seguimos, con mi amigo REForCE y yo nos hemos propuesto un reto que consiste en buscar distintas soluciones para el crackeo del juego, a continuación mi solución.

### [ Solución 1 ]

**Método :** Por Parcheo

**Autor:** StrongCod3r

Bueno se que esta solución es la más fácil y rápida y menos elegante, pero igual es una solución :D.  
Cargamos el juego vamos a Search for → All Referenced Strings y ponemos un Bp en:

004181CF	MOV EAX,OFFSET	arkanoidsb.00432528	ASCII "TIP: Try change video mode to increase performance. TIP: Kam
0041823A	MOV EBX,OFFSET	arkanoidsb.004315DC	ASCII "1.3.1.2"
0041826C	MOV EAX,OFFSET	arkanoidsb.004326E7	ASCII "Registered to: %s"
00418374	MOV EAX,OFFSET	arkanoidsb.004326F9	ASCII "Unregistered version"
0041944B	MOV EDX,OFFSET	arkanoidsb.00432750	ASCII "Kamikaze impact paddle"
00419458	MOV EDX,OFFSET	arkanoidsb.00432767	ASCII "Meteor impact paddle"
0041B807	MOV EBX,OFFSET	arkanoidsb.00432800	ASCII "%d"

Damos F9 Run y vemos que inicia el juego y para aquí:

0041835D	C70424 005541	MOV DWORD PTR SS:[ESP],OFFSET arkanoidsb.004326F9	ASCII "0:C"
00418364	E8 274D0000	CALL 0041D090	
00418369	E9 7FEFFFFF	JMP 004181ED	
0041836E	931C0	XOR EAX,EAX	
00418370	894424 10	MOV DWORD PTR SS:[ESP+10],EAX	
00418374	B8 F9264300	MOV EAX,OFFSET arkanoidsb.004326F9	ASCII "Unregistered version"
00418379	894424 0C	MOV DWORD PTR SS:[ESP+0C],EAX	
0041837D	E9 05FFFFFF	JMP 00418287	
00418382	90	NOP	
00418383	90	NOP	
00418384	90	NOP	
00418385	90	NOP	
00418386	90	NOP	
00418387	90	NOP	
00418388	90	NOP	
00418389	90	NOP	
0041838A	90	NOP	
0041838B	90	NOP	
0041838C	90	NOP	
0041838D	90	NOP	
0041838E	90	NOP	
0041838F	90	NOP	
00418390	55	PUSH EBP	
00418391	89E5	MOV EBP,ESP	
Jump from 418257			

Vemos que hemos saltado desde 418257 , así retrocedemos a esa offset, aquí:

0041823F	895C24 0C	MOV DWORD PTR SS:[ESP+0C],EBX	
00418243	894C24 08	MOV DWORD PTR SS:[ESP+08],ECX	
00418247	895424 04	MOV DWORD PTR SS:[ESP+4],EDX	
0041824B	E8 F02F0000	CALL 0041B240	
00418250	803D A5404300	CMP BYTE PTR DS:[4340A5],0	
00418257	BF84 101000	JE 0041836E	NAG
0041825D	E8 40404300	MOV EAX,OFFSET arkanoidsb.00434040	
00418262	809D A8F0FFFF	LEA EBX,[EBP-2581]	
00418268	894424 08	MOV DWORD PTR SS:[ESP+8],EAX	
0041826C	B8 E7264300	MOV EAX,OFFSET arkanoidsb.004326E7	ASCII "Registered to: %s"
00418271	894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
00418275	891C24	MOV DWORD PTR SS:[ESP],EBX	
00418279	E8 435B0000	CALL 00418287	
0041827D	895C24 0C	MOV DWORD PTR SS:[ESP+0C],EBX	
00418281	31C0	XOR EAX,EAX	
00418283	894424 10	MOV DWORD PTR SS:[ESP+10],EAX	
00418287	C70424 A07F41	MOV DWORD PTR SS:[ESP],OFFSET arkanoidsb.004326F9	
0041828F	B8 05000000	MOV EAX,5	

Como vemos caemos en JE 0041836E el que nos lanza a la **NAG Bad** y vemos que más debajo de JE vemos la **NAG Good**, así que la cuestión de que el juego no muestre la NAG buena esta en ese JE, lo parcharíamos allí pero seria un grave error ya que si parcheamos solo arreglaríamos la NAG mala por la buena y el juego todavía estaría sin registrar, ya que el juego no solo se fija en la NAG sino en muchos lados más para comprobar si de verdad estamos registrados, así que eso de parchar allí no es la solución así vamos a tomar otro camino, arriba del JE vemos :

00418250 |. 803D A5404300 CMP BYTE PTR DS:[4340A5],0

Vemos que compara esta address 4340A5 con 0 si son iguales se activaría el Flag Z y JE se cumpliría y saltaría a la NAG Mala, así que para que no se cumpla debe haber algún valor en 4340A5, seguimos, así que ponemos un Bp en 00418250 y reiniciamos el juego y damos run de nuevo y caemos aquí:

00418247	895424 04	MOV DWORD PTR SS:[ESP+4],EDX	
00418248	E9 F02F0000	CALL 00418248	
00418250	803D A5404300	CMP BYTE PTR DS:[4340A5],0	
00418257	0F84 11010000	JE 0041836E	NAG
0041825D	B8 40404300	MOV EAX,OFFSET arkanoidsb.00434040	
00418262	8D9D A8FDFFF	LEA EBX,[EBP-258]	
00418268	894424 08	MOV DWORD PTR SS:[ESP+8],EAX	
00418269	B8 E7264300	MOV EAX,OFFSET arkanoidsb.004326E7	ASCII "Registered to: %s"
00418271	894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
00418275	891C24	MOV DWORD PTR SS:[ESP],EBX	
00418278	E9 43BB0000	CALL <JMP.&msvcrt.sprintf>	
0041827D	895424 04	MOV DWORD PTR SS:[ESP+4],EAX	

Y vemos en el Dump la que tiene la address 4340A6:

Address	Hex dump	ASCII
004340A5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
004340B5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
004340C5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
004340D5	00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 00	.....
004340E5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
004340F5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....

Y vemos que no está a cero osea el CMP se cumpliría pero si estuviera algún valor allí el CMP no se cumpliría y JE no saltaría a la Nag Mala sino que pasaría de largo a la Nag buena. Jojo...ósea la solución está en esa Address, vamos a poner un Bpm(BreakPoint Memory) en 4340A5, seleccionamos el valor de la address clic derecho Breakpoint→Memory:

Edit memory breakpoint at arkanoidsb:.bss:004340A5..004340A8

Break on:

☒ Read access
☒ Write access
☐ Execution
☐ Disabled

OK
Cancel

Marcamos esas Read y Write y damos ok, listo ahora vamos a reiniciar el juego y damos run de nuevo y parara aquí:

004026EA	8D45 08	LEA EAX,[EBP-28]	
004026ED	BF 40404300	MOV EDI,OFFSET arkanoidsb.00434040	
004026F2	894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004026F6	8D45 A8	LEA EAX,[EBP-58]	
004026F9	C605 A5404300	MOV BYTE PTR DS:[4340A5],0	
00402700	890424	MOV DWORD PTR SS:[ESP],EAX	
00402703	E8 A0160200	CALL <JMP.&msvcrt strcpy>	
00402708	90	NOP	
00402709	8DB426 000000	LEA ESI,[ESI]	

Y vemos que está moviendo 0 a nuestra dirección y por consecuencia de eso saltaríamos más después a la NAG mala, la solución es cambiar ese 0 por un 1 y quedaría así:

004026EA	8D45 08	LEA EAX,[EBP-28]	
004026ED	BF 40404300	MOV EDI,OFFSET arkanoidsb.00434040	
004026F2	894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004026F6	8D45 A8	LEA EAX,[EBP-58]	
004026F9	C605 A5404300	MOV BYTE PTR DS:[4340A5],1	
00402700	890424	MOV DWORD PTR SS:[ESP],EAX	
00402703	E8 A0160200	CALL <JMP.&msvcrt strcpy>	
00402708	90	NOP	

Una vez que hemos hecho eso más adelante veremos que el JE ya no saltara a la NAG mala sino que pasara de largo, así que quitaremos todos los Bp y Bpm y damos Run y veremos esto:



Vemos que ya nos sale registrado y el botón de “register” ha desaparecido, para comprobar si el juego es bien crackeado me puse a jugar un rato y ni rastro de las Nag’s ni limite de tiempo de jugar, ahora si podemos guardar una copia del exe parchado. Como hemos visto hemos cambiado 1 byte y se a registrado el juego jejej, aquí acaba mi solución vamos a dar paso a la siguiente solución.

## [ Solucion 2 ]

**Método:** Serial

**Autor:** REFoRCE

Bueno esta solución es la elegante como dicen lo maestros, así que allí va el va el escrito REFoRCE.

Hola! .Pues me toca la continuación del tute de StrongCod3r acerca de Arkanoïd v-1.3.1.2,.La forma en que StrongCod3r registra el programa es una buena opción, demuestra lo simple que puede hacer registrar el juego y es la más fácil sin modificar tanto la aplicación. Bueno la siguiente parte es de cómo registrar el juego atreves de la búsqueda de su serial.

Bueno pues manos a la obra. >)

Como ya se vio en el tutorial de StrongCod3r las screenshot del menú principal con el botón:

“Register” y el texto “Unregistered versión” se sabe también que debemos agregar un “name” y un “serial” o “key” tipo: XXXX-XXXX-XXXX-XXXX. Esta vez no agregare ni clave ni name solo lo cargo en ollydbg y empiezo a buscar por pistas.

Bueno entonces lo cargo en ollydbg. Doy Search for-All Referenced text string y veo:

Text strings referenced in arkanoïd.text				
Address	Dis4ss3mbl5		Text string	
00401EAA	MOV	DWORD PTR SS:[ESP], 43062B	ASCII	"mainmenuicons.png"
00401EC4	MOV	DWORD PTR SS:[ESP], 430630	ASCII	"cursor.png"
00401EDE	MOV	DWORD PTR SS:[ESP], 430648	ASCII	"transp.png"
00401EF8	MOV	DWORD PTR SS:[ESP], 430653	ASCII	"tutorial_dlg.png"
00401F12	MOV	DWORD PTR SS:[ESP], 430664	ASCII	"sinusstring.png"
00401F2C	MOV	DWORD PTR SS:[ESP], 430674	ASCII	"options.png"
00401F65	MOV	EAX, 430167	ASCII	"yes"
00402199	ASCII	" %", 0		
00402637	MOV	[LOCAL.6], 43068C	ASCII	"EKW3-4X54-Z72W-N7H5"
0040263E	MOV	[LOCAL.5], 4306D0	ASCII	"26ZE-95X9-WNFZ-7NY4"
00402773	MOV	EAX, 4306F4	ASCII	"%2X"
004027AC	MOV	EAX, 4306F4	ASCII	"%2X"
00402916	MOV	ESI, 430758	ASCII	"%sconfig"

Bueno al parecer son 2 claves en 00402637 y 0040263E así que doy doble clic en alguno de ellas para que me retorne al desensamblado justo aquí:



0040261F	90	NOP		
00402620	55	PUSH	EBP	
00402621	89E5	MOV	EBP, ESP	
00402623	83EC 28	SUB	ESP, 28	
00402626	897D FC	MOV	[LOCAL.1], EDI	
00402629	8B7D 08	MOV	EDI, [ARG.1]	
0040262C	895D F4	MOV	[LOCAL.3], EBX	
0040262F	310B	XOR	EBX, EBX	
00402631	8975 F8	MOV	[LOCAL.2], ESI	
00402634	8D75 E8	LEA	ESI, [LOCAL.6]	
00402637	C745 E8 8C064300	MOV	[LOCAL.6], 4306BC	ASCII "EKW3-4X54-Z72W-N7H5"
0040263E	C745 EC 00064300	MOV	[LOCAL.5], 4306D0	ASCII "26ZE-95X9-WNFZ-7NY4"
00402645	897C24 04	MOV	DWORD PTR SS:[ESP+4], EDI	
00402649	8B049E	MOV	EAX, DWORD PTR DS:[ESI+EBX*4]	
0040264C	890424	MOV	DWORD PTR SS:[ESP], EAX	
0040264F	E8 FC160200	CALL	00423D50	strcmp
00402654	85C0	TEST	EAX, EAX	
00402656	74 15	JE	SHORT 0040266D	arkanoid.0040266D
00402658	43	INC	EBX	
00402659	83FB 01	CMP	EBX, 1	
0040265C	76 E7	JBE	SHORT 00402645	arkanoid.00402645
0040265E	31C0	XOR	EAX, EAX	
00402660	8B5D F4	MOV	EBX, [LOCAL.3]	
00402663	8B75 F8	MOV	ESI, [LOCAL.2]	
00402666	8B7D FC	MOV	EDI, [LOCAL.1]	
00402669	89EC	MOV	ESP, EBP	
0040266B	5D	POP	EBP	
0040266C	C3	RETN		
0040266D	B8 01000000	MOV	EAX, 1	
00402672	EB EC	JMP	SHORT 00402660	arkanoid.00402660

OK. Todo bien pongo un breakpoint or F2 en 00402620 PUSH para no perderme de algunos movimientos que se hagan antes de las claves, doy Run o F9. :) y el breakpoint se activa.

En los registros aparece un valor específicamente en EAX.

R3g1st3rs (FPU)	
EAX	0022F800 ASCII "FN7W5FW452ZK4F5"
ECX	00000010
EDX	00000035
EBX	00000000
ESP	0022F7BC
EBP	0022F858
ESI	00000065

Ok anoto ese valor “FN7W5FW452ZK4F5” en el block de notas o en algún editor de textos, y analizo la pieza de código de arriba a partir del push, mientras voy traceando voy haciendo una traducción más o menos de los valores que se toman y de lo que se hace.

00402620	PUSH EBP	
00402621	MOV EBP, ESP	
00402623	SUB ESP, 28	
00402626	MOV [LOCAL 1] , EDI	
00402629	MOV EDI, [ARG 1]	EL valor “FN7W5FW452ZK4F5” a EDI
0040262C	MOV [LOCAL 3] EBX	
0040262F	XOR EBX, EBX	EBX = 0
00402631	MOV [LOCAL 2] , ESI	
00402634	LEA ESI, [LOCAL 6]	
00402637	MOV [LOCAL 6] 004306BC	ASCII “EKW3-4X54-Z72W-N7H5”
0040263E	MOV [LOCAL 5] 004306D0	ASCII “ 26ZE-95X9-WNFZ-7YN4”
00402645	MOV [ESP + 4] EDI	offset de “FN7W5FW452ZK4F5” a ESP +4 = 0022f794
00402649	MOV EAX [ESI + EBX * 4]	EKW3-4X54-Z72W-N7H5 a EAX
0040264C	MOV [ESP] EAX	
0040264F	CALL 00423D50	{ strcmp → step into F7 :)
00402654	TEST EAX , EAX	
00402656	JE SHORT 0040266D	
00402658	INC EBX	
00402659	CMP EBX, 1	
0040265C	JBE SHORT 00402645	
0040265E	XOR EAX, EAX	
00402660	MOV EBX [LOCAL 3]	
00402663	MOV ESI [LOCAL 2]	
00402666	MOV EDI [LOCAL 1]	
00402669	MOV ESP, EBP	
0040266B	POP EBP	
0040266C	RETN	
0040266D	MOV EAX, 1	
00402672	JMP 00402660	

Step into → 0040264F CALL strcmp lo que hará aquí será comparar:

0022F790	0043068C	s1 = "EKW3-4X54-Z72W-N7H5"
0022F794	0022F800	s2 = "FN7W5FW452ZK4F5"
0022F798	0022F822	

Está claro que no son iguales, dentro del call se sé hace la comparación del primer byte del valor: “EKW3-4X54-Z72W-N7H5” con el primer byte de “FN7W5FW452ZK4F5” como no son iguales F y E , EAX hace una SBB quedando en FFFFFFFF y después un SHL ,1 quedando en FFFFFFFE después se Incrementa quedando en -1. Osea [FFFFFFF]. .

#### Saliendo del CALL

00402654	TEST EAX, EAX	EAX = -1 [FFFFFFF]
00402656	JE SHORT 0040266D	sin tomarse.
00402658	INC EBX	EBX a 1
00402659	CMP EBX, 1	Cmp 1, 1
0040265C	JBE SHORT 00402645	Salto tomado : 0
0040265E	XOR EAX, EAX	
00402660	MOV EBX [LOCAL 3]	
00402663	MOV ESI [LOCAL 2]	
00402666	MOV EDI [LOCAL 1]	
00402669	MOV ESP, EBP	
0040266B	POP EBP	
0040266C	RETN	
0040266D	MOV EAX, 1	
00402672	JMP 00402660	

Ok. Como el JBE fue tomado al tracear con F8 me retorna a 00402645 y esta vez tomara la segunda clave: “26ZE-95X9-WNFZ-7NY4” y lo comparara con “FN7W5FW452ZK4F5” lo puedo checar en el stack al llegar traceando con **step over** sobre :

0040264F | CALL 00423D50 strcmp ← :)

0022F790	00430600	s1 = "26ZE-95X9-WNFZ-7NY4"
0022F794	0022F800	s2 = "FN7W5FW452ZK4F5"

← Stack

Entro con **step into** or **F7** in strcmp veo que hace lo mismo:

77C17730	MOV	EDX, DWORD PTR SS:[ESP+4]	
77C17734	MOV	ECX, DWORD PTR SS:[ESP+8]	
77C17738	TEST	EDX, 3	
77C1773E	JNZ	SHORT 77C1777C	
77C17740	MOV	EAX, DWORD PTR DS:[EDX]	
77C17742	CMP	AL, BYTE PTR DS:[ECX]	← Comparación de los siguientes
77C17744	JNZ	SHORT 77C17774	valores al tracear hasta aquí con F7
			Se pueden ver las comparaciones en los registros.

EAX	455A3632	
ECX	0022F800	ASCII "FN7W5FW452ZK4F5"
EDX	00430600	ASCII "26ZE-95X9-WNFZ-7NY4"
EBX	00000001	

← REGISTROS.

Más claro que el agua ! Las 2 claves fueron comparadas con el valor “FN7W5FW452ZK4F5” probablemente hará lo mismo con mi clave “key” falsa haciendo lo mismo en strcmp. Ok traceo hasta salir del CALL por lo tanto doy: **Execute till user code** or **Alt+ F9**.

Para llegar aquí:

00402654	TEST EAX, EAX	Igual EAX = FFFFFFFF (-1)
00402656	JE SHORT 0040266D	el salto no es tomado y se incrementa EBX
00402658	INC EBX	que anteriormente era 1 al incrementarse será 2
00402659	CMP EBX, 1	se compara con 1 por lo tanto el JBE tampoco
0040265C	JBE SHORT 00402645	es tomado permitiéndome tracear hasta el RETN.
0040265E	XOR EAX, EAX	Ok. Traceo hasta el RETN con ctrl+F9
00402660	MOV EBX [LOCAL 3]	
00402663	MOV ESI [LOCAL 2]	

```

00402666    MOV EDI [LOCAL 1]
00402669    MOV ESP, EBP
0040266B    POP EBP
0040266C    RETN
0040266D    MOV EAX, 1
00402672    JMP 00402660

```

Llegando hasta el RETN en 0040266C doy *step in* or **F7** para llegar aquí:

004027FF	. 84C0	TEST	AL, AL	
00402801	. 75 10	JNZ	SHORT 00402820	arkanoid.00402820
00402803	. 8045 88	LEA	EAX, DWORD PTR SS:[EBP 78]	
00402806	. 8055 A8	LEA	EDX, DWORD PTR SS:[EBP 58]	
00402809	. 894424 04	MOV	DWORD PTR SS:[ESP+4], EAX	
0040280D	. 891424	MOV	DWORD PTR SS:[ESP], EDX	
00402810	. E8 3B150200	CALL	00423050	strcmp
00402815	. 85C0	TEST	EAX, EAX	
00402817	. 75 07	JNZ	SHORT 00402820	arkanoid.00402820

Sip otro strcmp,.....:

Ok. Todo bien, es tiempo de ejecutar por completo doy F9 y pongo mi “key” falsa:

ABCD-1234-5678-ABCD

Y hago lo mismo ingreso mi nickname:

REFORCE

Presiono el botoncillo para aceptar los valores y veo que mi Bp en el PUSH se activo

00402620    PUSH EBP    ← BP F2 Activado y traceo hasta llegar a:

00402645	. > 897C24 04	MOV	DWORD PTR SS:[ESP+4], EDI	
00402649	. 8B049E	MOV	EAX, DWORD PTR DS:[ESI+EBX*4]	
0040264C	. 890424	MOV	DWORD PTR SS:[ESP], EAX	
0040264F	. E8 FC160200	CALL	00423050	strcmp

Sip en strcmp de nuevo. Ahora veo el stack y un nuevo valor comparando la clave:  
“EKW3-4X54-Z72W-N7H5”

0022F600	004306BC	s1 = "EKW3-4X54-Z72W-N7H5"
0022F604	0022F670	s2 = "XWK43F236H5KYH74"

← Un nuevo valor! : ).

Wtf! Okay se supone que compara también: “26ZE-95X9-WNFZ-7YN4” con el nuevo valorcito:  
“XWK43F236H5KYH74” y para asegurarme pongo un Breakpoint en:

00402645 MOV DWORD PTR SS: [ESP + 4], EDI

Un poco arriba de strcmp en 0040264F. Y doy run o F9.

OK al hacer esto mi Breakpoint se activa y traceo con F8 de nuevo hasta strcmp en 0040264F.

Por lo tanto puedo ver el stack y checar mi suposición que es correcta.

0022F600	00430600	s1 = "26ZE-95X9-WNFZ-7NY4"
0022F604	0022F670	s2 = "XWK43F236H5KYH74"

← Igual con el valor nuevo  
Se realizó la comparación.

Ok. Traceo hasta llegar al RETN que se encuentra un poco más debajo de strcmp.

0040266C    RETN    ← aqui doy step into.

Entonces llego de Nuevo aquí y traceo hasta strcmp:

004027FF	. 84C0	TEST	AL, AL	
00402801	. 75 10	JNZ	SHORT 00402820	arkanoid.00402820
00402803	. 8045 88	LEA	EAX, DWORD PTR SS:[EBP 78]	
00402806	. 8055 A8	LEA	EDX, DWORD PTR SS:[EBP 58]	
00402809	. 894424 04	MOV	DWORD PTR SS:[ESP+4], EAX	
0040280D	. 891424	MOV	DWORD PTR SS:[ESP], EDX	
00402810	. E8 3B150200	CALL	00423D50	strcmp
00402815	. 85C0	TEST	EAX, EAX	
00402817	. 75 07	JNZ	SHORT 00402820	arkanoid.00402820

Traceando hasta strcmp veo en el stack mi “key” falsa comparada con: XWK43F236H5KYH74

0022F630	0022F670	ASCII	"XWK43F236H5KYH74"
0022F634	0022F650	ASCII	"ABCD12345678ABCD"

Interesante!.

Hasta aquí se sabe que hay 2 valores con los cuales las claves son comparadas

- 1.- XWK43F236H5KYH74
- 2.- FN7W5FW452ZK4F5

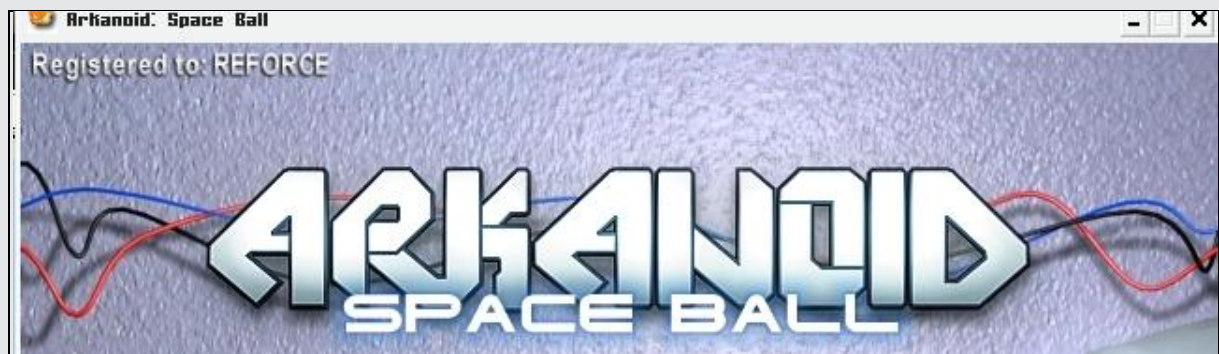
Bueno es tiempo de tomar un riesgo. Testear los dos valores, mientras tanto reinicio. Ctrl-F2.

Hasta aquí se que mi “key” fue comparada con “XWK43F236H5KYH74” okay.

Empiezo con esa quito todos los Bp. Doy Run F9. Presiono el botón de “Register” eh ingreso la clave de esta manera:

XWK4-3F23-6H5K-YH74

Y también mi name: REFORCE jeje. Clic para aceptar los valores y:



Yep, >) Registered!

Testeando: FN7W5FW452ZK4F5 quedaría así la clave FN7W-5FW4-52ZK-4F5

Esta clave funciona sin agregar un nombre y:



Ok. Se ha registrado el juego.



Bueno espero haberme entendido ☺ en este tutorial, no soy buen escritor y creo hice mi mejor esfuerzo jeje, Saludos a todos los que se conectan los fines de semana en el IRC-Hispano en el canal # **CrackslatinoS** aunque no pertenezco a la lista de CLS ahí me encuentro generalmente en el canal.

**StrongCod3r** : Bueno ya tenemos las 2 soluciones pero además de crackear vamos a jugar un rato a ser dios en el juego, seguid leyendo.

## [ GameHack con Olly ]

Viendo en las teorías tutoriales en lo que se refiere al GameHack(trainers y formas de hacer cheats) hacían uso de programas externos para hacer dicha tarea, ahora vamos a ver cómo hacerlo solo con olly + plugin sin ningún programa, otro método más de hacer gamehack.

Para dicha tarea vamos hacer uso de unos plugins que encontrado para olly que nos va ayudar a encontrar los valores en la memoria.

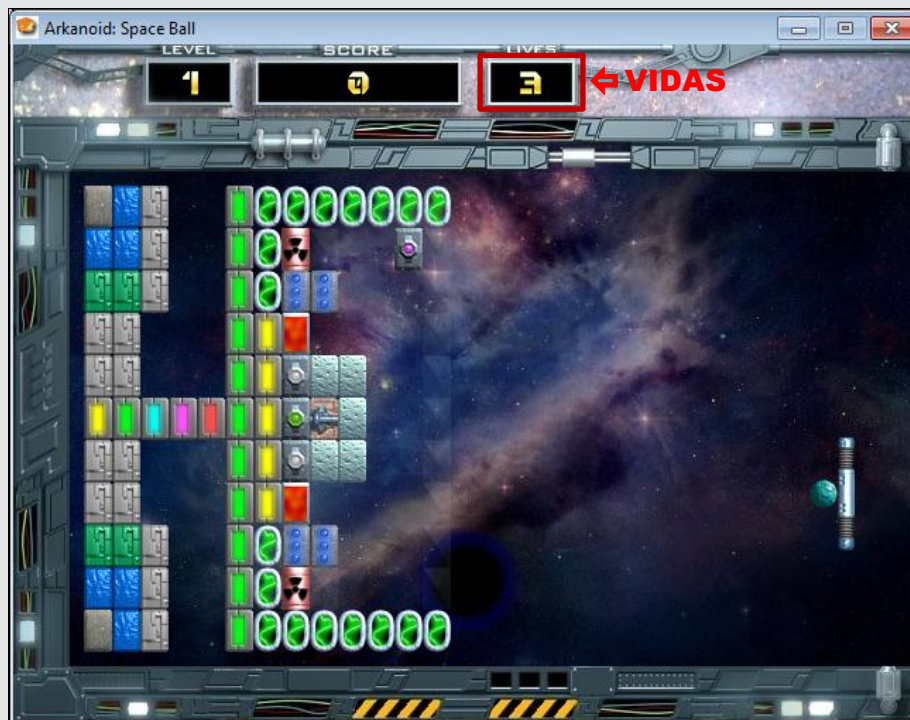
Bueno los plugins son 2 :

Games Invader.dll

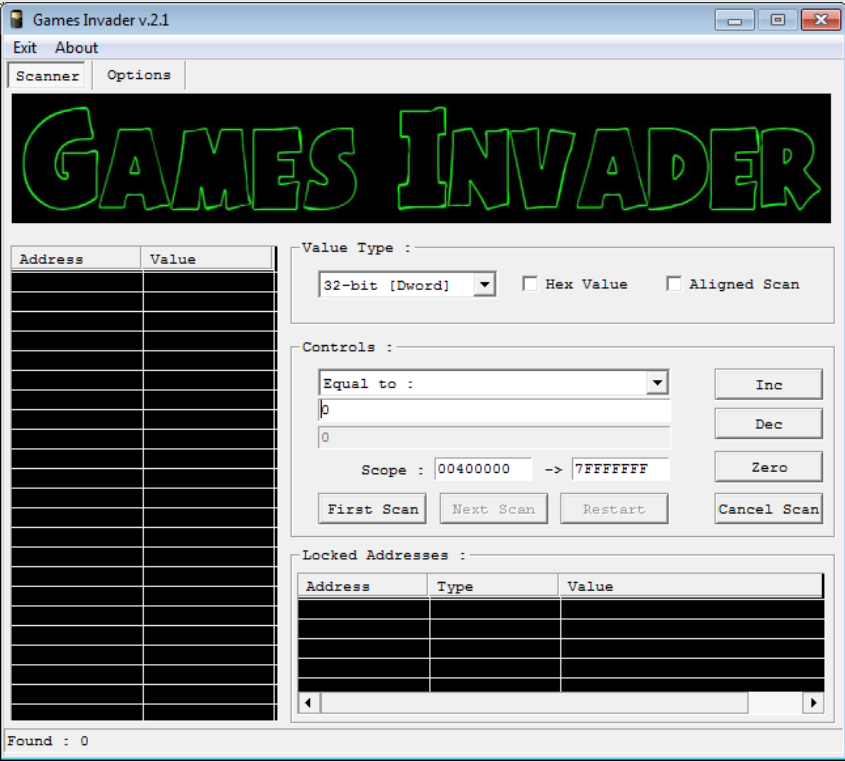
CheatUtility.dll

Los dos son muy bueno , pertenecen al mismo autor *GamingMasteR* , pero personalmente uso el GameInvader.dll porque tiene más opciones, se las adjuntado con el tutorial, cópienlo en la carpeta de plugins de olly.

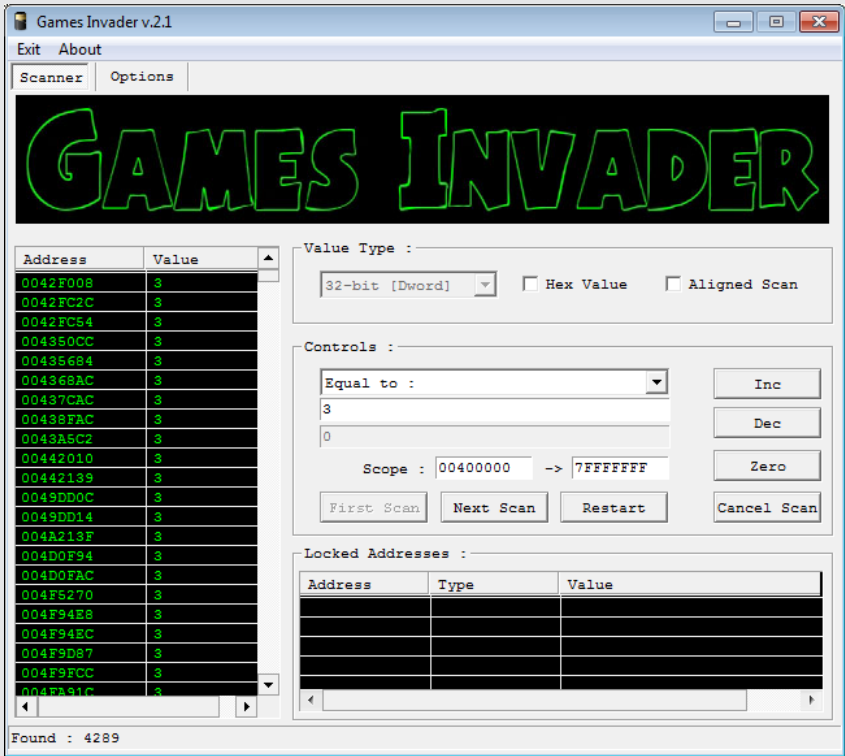
Bueno como estaba cansado de morir y gastarme esas pocas 3 vidas que me dan y que me tire el GameOver, vamos arreglar eso. Cargamos el juego y jugamos una partida y mostrara esto:



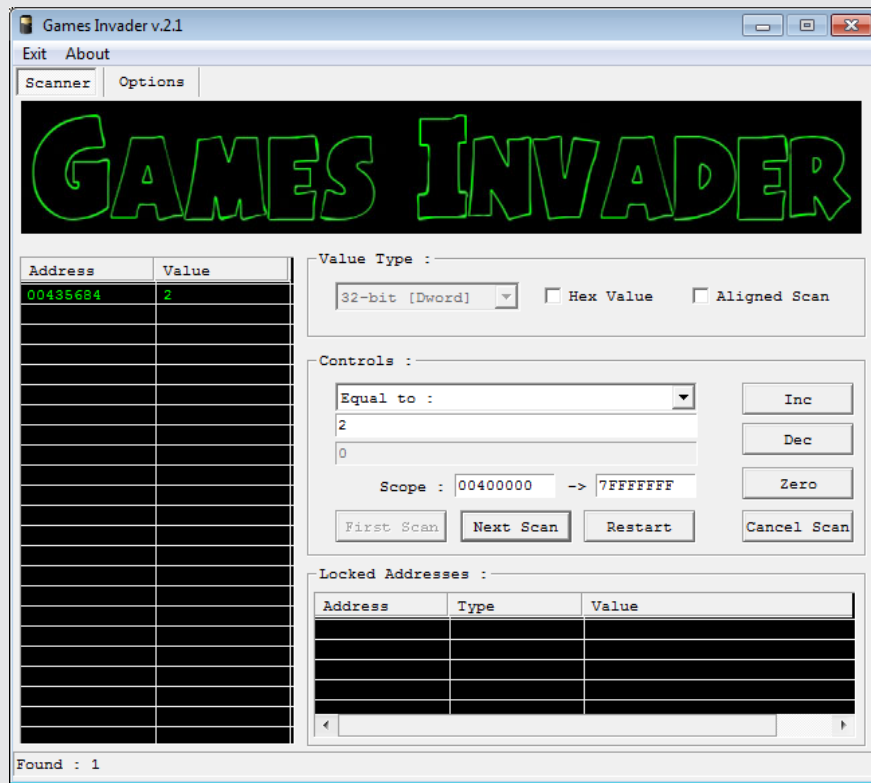
Hay vemos las 3 vidas que nos dan al empezar el juego, bueno vamos al olly en el “Menu→Plugins→ GamesInvader→ BeginScan”, y nos aparecerá la siguiente ventana:



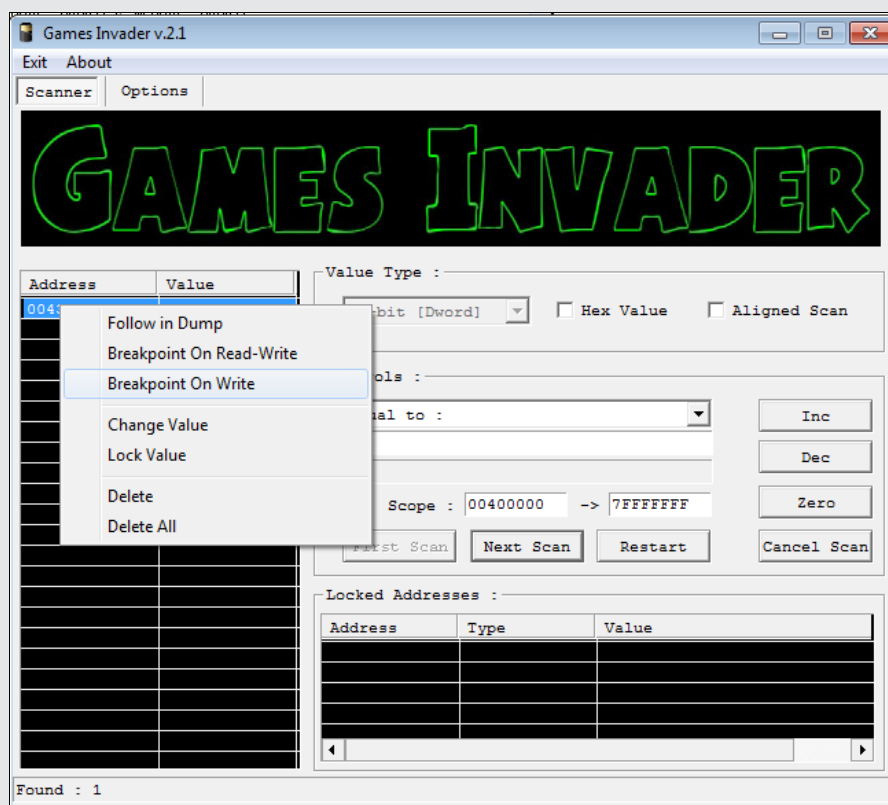
En el 0 ponemos el número de vidas que tenemos actualmente, ósea 3 y damos clic en **First Scan** y vemos esto:



Ha buscado en toda la memoria el numero 3 y vemos que hay bastantes Address con ese valor, sin cerrar la ventana del GamesInvader regresamos al juego y seguimos jugando y perdemos intencionalmente una vida para que nos quede 2 vidas, volvemos al olly sin cerrar el juego y en la ventana GamesInvaders , en el cuadro de texto donde estaba 3 lo ponemos a 2 y damos al Boton **Next Scan** y vemos esto:



Vemos que ha descartado las demás Address y solo ha quedado una (en su pc puede ser otra dirección), en esa address damos clic derecho → Breakpoint On Write, así:

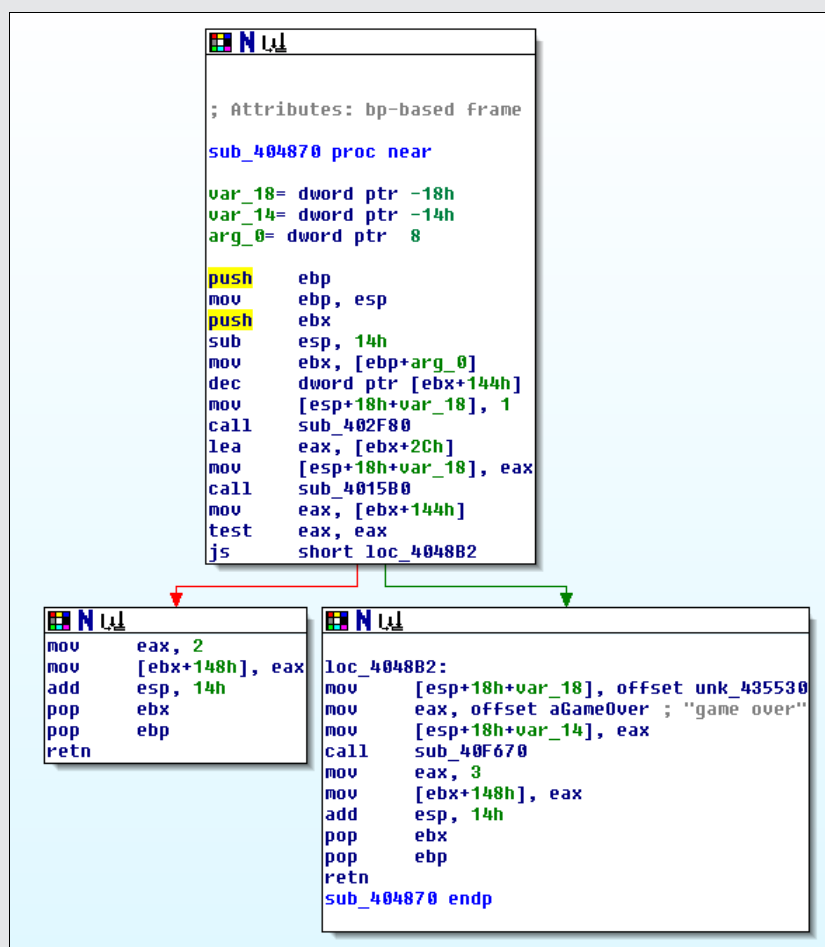


Listo ahora ya podemos cerrar el GameInvaders y seguimos jugando el juego y otras ves perdemos intencionalmente otra vida y parara en el Bp que pusimos antes, aquí:

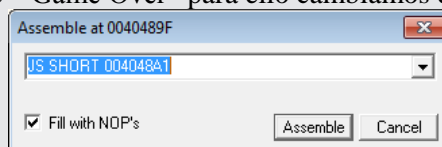
00404870	55	PUSH EBP	
00404871	89E5	MOV EBP,ESP	
00404873	53	PUSH EBX	
00404874	83EC 14	SUB ESP,14	
00404877	8B5D 08	MOV EBX,DWORD PTR SS:[EBP+8]	
00404878	FF8B 44010000	DEC DWORD PTR DS:[EBX+144]	
00404880	C70424 01000000	MOV DWORD PTR SS:[ESP],1	
00404887	E8 F4E6FFFF	CALL arkanoid.00402F80	
0040488C	8D43 2C	LEA EAX,DWORD PTR DS:[EBX+2C]	
0040488F	890424	MOV DWORD PTR SS:[ESP],EAX	
00404892	E8 19CDFFFF	CALL arkanoid.004015B0	
00404897	8B83 44010000	MOV EAX,DWORD PTR DS:[EBX+144]	
0040489D	85C0	TEST EAX,EAX	
0040489F	78 11	JS SHORT arkanoid.004048B2	
004048A1	B8 02000000	MOV EAX,2	
004048A6	8983 48010000	MOV DWORD PTR DS:[EBX+148],EAX	
004048AC	83C4 14	ADD ESP,14	
004048AF	5B	POP EBX	
004048B0	5D	POP EBP	
004048B1	C3	RETN	
004048B2	C70424 305543	MOV DWORD PTR SS:[ESP],arkanoid.00435530	
004048B9	B8 880A4300	MOV EAX,arkanoid.00430A88	ASCII "game over"
004048BE	894424 04	MOV DWORD PTR SS:[ESP+4],EAX	arkanoid.0040F670
004048C2	E8 A9AD0000	CALL arkanoid.0040F670	
004048C7	B8 03000000	MOV EAX,3	
004048CC	8983 48010000	MOV DWORD PTR DS:[EBX+148],EAX	
004048D2	83C4 14	ADD ESP,14	
004048D5	5B	POP EBX	
004048D6	5D	POP EBP	
004048D7	C3	RETN	

Yo creo que no hace falta explicar nada aquí lo que hace desde la offset **00404870** a **004048D7**, bueno porsiacaso algún despistado no lo ve todavía vamos a dar algunas pistas de lo que hace.

Hay vemos al DEC es el que se encarga de restar las vidas, más abajo vemos un **test eax, eax** si estuviera a cero eax el JS saltaría al “Game Over” :D, para que lo vean más claro yo recomiendo que cuando lleguen a uno de estos puntos y halla mucho código(que no es nuestro caso pero lo vamos a ver, es más útil en juegos de mayor tamaño) , aunque se puede hacer todo con olly sin ningún problema, veamos como se ve e IDA en modo Graph(grafico):



Hay vemos el flujo del juego cuando pasa por el JS el que decide si seguimos jugado o nos manda al Game Over, vamos hacer algo como lo hizo Marciano en su tute del NES, pero nosotros no vamos a restar o sumar las vidas, vamos hacer que el juego nunca nos tire “Game Over” para ello cambiamos el JS con esto :





0040489D	. 85C0	TEST EAX,EAX	
0040489F	78 00	JS SHORT arkanoid.004048A1	
004048A1	B8 02000000	MOV EAX,2	
004048A6	8983 48010000	MOV DWORD PTR DS:[EBX+148],EAX	
004048AC	83C4 14	ADD ESP,14	
004048AF	5B	POP EBX	
004048B0	5D	POP EBP	
004048B1	C3	RET	
004048B2	> C70424 305543	MOV DWORD PTR SS:[ESP],arkanoid.00435531	
004048B9	B8 880A4300	MOV EAX,arkanoid.00430A88	ASCII "game over"
004048BE	894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004048C2	E8 A9AD0000	CALL arkanoid.0040F670	arkanoid.0040F670
004048C7	B8 03000000	MOV EAX,3	
004048CC	8983 48010000	MOV DWORD PTR DS:[EBX+148],EAX	
004048D2	83C4 14	ADD ESP,14	
004048D5	5B	POP EBX	
004048D6	5D	POP EBP	
004048D7	C3	RET	

Vemos que cambiamos la el Offset del JS que nos tiraba al “GameOver” por la siguiente, ósea continuaría el flujo juego normalmente aunque ya no tengamos vidas seguiremos jugando y no perderemos nunca.

Bueno al final me he divertido más crackeandolo que jugando propio juego :D, con esto damos por terminado este MegaHiperNewbie-Tute, pido disculpas desde ya por los errores en el tute y la escritura del tute que seguro los hay.

Saludos los que siempre están en el IRC Zapper, KrozmiCCLS, Asphyxia, jackgris y a toda la lista y en especial a Ricardo por estar tantos años al pie del cañón.

**By StrongCod3r - REFoRCE**