

Verificando una app con Stolen bytes DAP 0.2 by Apuromafo

Este nivel fue resuelto en el concurso 94, pero quedaba pendiente resolver el **antidump**, es **simple** como quitarlo, pero en su tiempo no lo era, espero colaborar con un granito de arena

*El ollydbg que uso tiene strongOD, así que no detecta ningún truco.*

Yo simplemente lo ejecute y revise, pero no pensaba que se podría hacer algo más.

Quienes quieran leer el escrito de solid:

[http://ricardonarvaja.info/WEB/CONCURSOS%20VIEJOS/CONCURSOS%202004-2006/CONCURSO%2094/C\\_94\\_N3\\_solid.rar](http://ricardonarvaja.info/WEB/CONCURSOS%20VIEJOS/CONCURSOS%202004-2006/CONCURSO%2094/C_94_N3_solid.rar)

yo para escribir este escrito no lo usé, pero quien este este packer, siempre es bueno resolver los temas pendientes.

este es la apariencia del packer en si, Yo quiero acotar al antidump, agregar un manifest y borrar el título que acompaña, para que se vea mas bonito ☺

este es la apariencia inicial:



Si ejecutado y al pulsar el logo de river veo un mensaje

DAPv0.2 - [WinXP] - Hecho en MASM/RadAsm

dapaf - 4 de Julio de 2006

Agradecimientos:

Emadicius, por 'prestarme' parte de su agregaSeccion.asm ;)

Marciano, por la ayuda de siempre, en especial con esta mierda.

SolidSnake y Arapumk, por tomarse siempre la molestia de mirar las boludeces que les envio XD

Y a todos mis amigos de CracksLatinoS, en especial a la gente yahoolzera

dapaf@tutopia.com

Aceptar

Veamos que me muestra en el ollydbg con este mensaje vemos:

0012F578	77D3EA24	USER32.WaitMessage	USER32.77D3EA1F	0012F5A8
0012F5AC	77D2688A	USER32.77D3E895	USER32.77D26885	0012F5A8
0012F5D4	77D3B7C5	USER32.77D267D4	USER32.77D3B7C0	0012F5D0
0012F894	77D3B128	USER32.SoftwareModalMessageBox	USER32.77D3B126	0012F890
0012F9E4	77D65FDF	USER32.77D3AFB6	USER32.77D65FDA	0012F9E0
0012FA3C	77D66084	USER32.MessageBoxTimeoutW	USER32.77D6607F	0012FA38
0012FA70	77D50598	? USER32.MessageBoxTimeoutA	USER32.77D50593	0012FA6C
0012FA90	77D50550	? USER32.MessageBoxExA	USER32.77D5054B	0012FA8C
0012FA94	004301AC	hOwner = 004301AC ("DAP v0.2		
0012FA98	00405104	Text = "DAPv0.2 - [WinXP] - Hecho		
0012FA9C	00405297	Title = "DAPv0.2 - [WinXP] - Hecho		
0012FAA0	00000000	Style = MB_OK MB_APPLMODAL		
0012FAA4	00000000	LanguageID = 0 (LANG_NEUTRAL)		
0012FAAC	0040212D	? <JMP.&user32.MessageBoxA>	Copia_(2.00402128	0012FAA8
0012FAB0	004301AC	hOwner = 004301AC ("DAP v0.2		
0012FAB4	00405104	Text = "DAPv0.2 - [WinXP] - Hecho		
0012FAB8	00405297	Title = "DAPv0.2 - [WinXP] - Hecho		
0012FABC	00000000	Style = MB_OK MB_APPLMODAL		
0012FAC4	77D18709	Includes Copia_(2.0040212D	USER32.77D18706	0012FAC0
0012FAF0	77D24CA6	? USER32.77D186E1	USER32.77D24CA1	0012FAEC
0012FAF4	0040208D	<JMP.&kernel32.ExitProcess>	Copia_(2.00402088	0012FB58
0012FAF8	004301AC	ExitCode = 4301AC		

Pero si pauso sin ese mensaje veo

Call stack of main thread			
Address	Stack	Procedure / arguments	Called from
0012FE90	77D193F5	Includes ntdll.KiFastSystemCallRet	USER32.77D193F3
0012FE94	77D3EA24	USER32.WaitMessage	USER32.77D3EA1F
0012FEC8	77D2688A	USER32.77D3E895	USER32.77D26885
0012FEF0	77D268CC	USER32.77D267D4	USER32.77D268C7
0012FF10	77D2892D	USER32.DialogBoxIndirectParamAorW	USER32.77D28928
0012FF3C	00402086	<JMP.&user32.DialogBoxParamA>	Copia_(2.00402081
0012FF40	00400000	hInst = 00400000	
0012FF44	00000065	pTemplate = 65	
0012FF48	00000000	hOwner = NULL	
0012FF4C	0040208D	DlgProc = Copia_(2.0040208D	
0012FF50	00000000	lParam = NULL	

El dialogo comienza en 40208d

Veamos por ahí

00402087	7E	DEC ECX	
00402089	21E9	AND ECX,EBP	
0040208B	41	INC ECX	
0040208D	-0F89 E7E85E14	JNS 149F0955	
0040208E	0000	ADD BYTE PTR DS:[EAX],AL	
00402090	6A 00	PUSH 0	
00402092	68 8D204000	PUSH Copia_(2.0040208D	
00402094	6A 00	PUSH 0	
00402096	6A 65	PUSH 65	
00402098	FF35 B05A4000	PUSH DWORD PTR DS:[405AB0]	
0040209A	E8 0C140000	CALL <JMP.&user32.DialogBoxParamA>	Copia_(2.00400000
0040209C	6A 00	PUSH 0	
0040209E	E8 C3130000	CALL <JMP.&kernel32.ExitProcess>	
004020A0	55	PUSH EBP	
004020A2	8BEC	MOV EBP,ESP	

Y al revisar a la iat se ve InitCommon

004034C2	-FF25 64404000	JMP DWORD PTR DS:[&user32.SetLayeredWindowAttributes]	USER32.SetLayeredWindowAttri
004034C8	-FF25 68404000	JMP DWORD PTR DS:[&user32.SetWindowLongA]	USER32.SetWindowLongA
004034CE	-FF25 00404000	JMP DWORD PTR DS:[&comctl32.InitCommonControls]	COMCTL32.InitCommonControls
004034D4	0000	ADD BYTE PTR DS:[EAX],AL	
004034D6	0000	ADD BYTE PTR DS:[EAX],AL	
004034D8	0000	ADD BYTE PTR DS:[EAX],AL	

Veamos cuando lo llama

Hardware, on execution

Al reiniciar vemos

0012FF50	00402070	CALL to InitCommonControls from Copia_(2.0040206B
0012FF54	00000015	

40206b

Tenemos entonces el OEP por aquí, pero habrá algún **valor escrito antes?**

0040206B	E8 5E140000	CALL <JMP.&comctl32.InitCommonControls>
00402070	6A 00	PUSH 0
00402072	68 8D204000	PUSH Copia_(2.0040206B
00402077	6A 00	PUSH 0

Si busco alguna referencia de esta dirección vemos como se hace:

0040206B	CALL <JMP.&comctl32.InitCommonControls>	(Initial CPU selection)
00414618	MOV DWORD PTR DS:[EAX],Copia_(2.0040206B	0040206B=Copia_(2.0040206B

004145E5	. 33C0	XOR EAX,EAX
004145E7	. 33C9	XOR ECX,ECX
004145E9	. 05 00000E00	ADD EAX,0E0000
004145EE	. C700 B8000040	MOV DWORD PTR DS:[EAX],400000B8
004145F4	. 83C0 04	ADD EAX,4
004145F7	. C700 00000000	MOV DWORD PTR DS:[EAX],0
004145FD	. 83C0 01	ADD EAX,1
00414600	. C700 A3B05A40	MOV DWORD PTR DS:[EAX],405AB0A3
00414606	. 83C0 04	ADD EAX,4
00414609	. C700 00000000	MOV DWORD PTR DS:[EAX],0
0041460F	. 83C0 01	ADD EAX,1
00414612	. C600 BA	MOV BYTE PTR DS:[EAX],0BA
00414615	. 83C0 01	ADD EAX,1
00414618	. C700 6B204000	MOV DWORD PTR DS:[EAX],Copia_(2.0040206B
0041461E	. 83C0 04	ADD EAX,4
00414621	. C700 C7050000	MOV DWORD PTR DS:[EAX],5C7
00414627	. 83C0 04	ADD EAX,4
0041462A	. C700 0E009077	MOV DWORD PTR DS:[EAX],7790000E
00414630	. 83C0 04	ADD EAX,4
00414633	. C700 90EC0000	MOV DWORD PTR DS:[EAX],0EC90
00414639	. 83C0 04	ADD EAX,4
0041463C	. C700 C7051600	MOV DWORD PTR DS:[EAX],1605C7
00414642	. 83C0 04	ADD EAX,4
00414645	. C700 0E009090	MOV DWORD PTR DS:[EAX],9090000E
00414648	. 83C0 04	ADD EAX,4
0041464E	. C700 90E80000	MOV DWORD PTR DS:[EAX],0E890
00414654	. 83C0 02	ADD EAX,2
00414657	. 91	XCHG EAX,ECX
00414658	. 66:C701 FFE2	MOV WORD PTR DS:[ECX],0E2FF
0041465D	. 61	POPAD
0041465E	. EB 21	JMP SHORT Copia_(2.00414681

Bien teníamos un ejemplo de copiar trozos de memoria con rep es lo mas común en trozos de stolen , pero aquí es con MOV, mueve a 0e0000 , veamos ese trozo

Asi esta cuando ya estamos en el oep

000E0000	90	NOP	
000E0001	-77 90	JN SHORT 0000FF93	
000E0003	EC	IN AL,DX	I/O command
000E0004	00A3 B05A4000	ADD BYTE PTR DS:[EBX+405AB0],AH	
000E000A	BA 6B204000	MOV EDX,40206B	
000E000F	C705 00000E00	MOV DWORD PTR DS:[E0000],90909090	
000E0019	E8 00C70516	CALL 1613C71E	
000E001E	000E	ADD BYTE PTR DS:[ESI],CL	
000E0020	0090 9090E8FF	ADD BYTE PTR DS:[EAX+FFE89090],DL	
000E0026	✓E2 00	LOOPD SHORT 000E0028	
000E0028	0000	ADD BYTE PTR DS:[EAX],AL	
000FA02A	0000	ADD BYTE PTR DS:[EAX],AL	

Y así es cuando estaba, y claramente se auto/edita en la dirección 0e0000 notese la diferencia

000E0000	B8 00004000	MOV EAX,400000
000E0005	A3 005A4000	MOV DWORD PTR DS:[405AB0],EAX
000E000A	BA 6B204000	MOV EDX,40206B
000E000F	C705 00000E00 9	MOV DWORD PTR DS:[E0000],EC907790
000E0019	0000	ADD BYTE PTR DS:[EAX],AL
000E001B	C705 16000E00 9	MOV DWORD PTR DS:[E0016],E8909090
000E0025	FFE2	JMP EDX
000E0027	0000	ADD BYTE PTR DS:[EAX],AL

Vemos como realmente mueve el valor del

**Mov eax,400000 y**

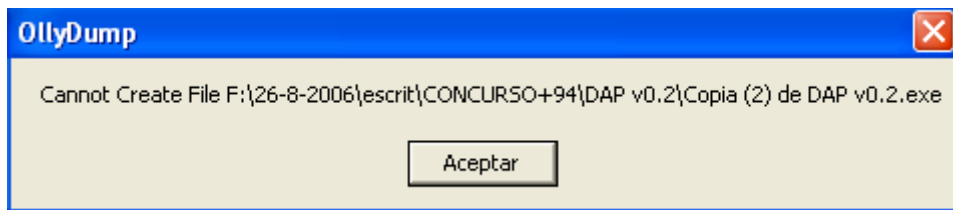
**MOV DWORD PTR DS:[405AB0],EAX**

Lo cual es lo mismo que **MOV DWORD PTR DS:[405AB0], 400000**

Ahora bien, es necesario ese valor pues es usado como hInst en el dialogo

00402077	. 6A 00	PUSH 0	hOwner = NULL
00402079	. 6A 65	PUSH 65	pTemplate = 65
0040207B	. FF35 B05A4000	PUSH DWORD PTR DS:[405AB0]	hInst = 00400000
00402081	. E8 0C140000	CALL <JMP.&user32.DialogBoxParamA>	DialogBoxParamA
00402086	. 6A 00	PUSH 0	ExitCode = 0

Ahora que sabemos el oep, tenemos el stolen byte, y ahora queda dumpear



Solución:

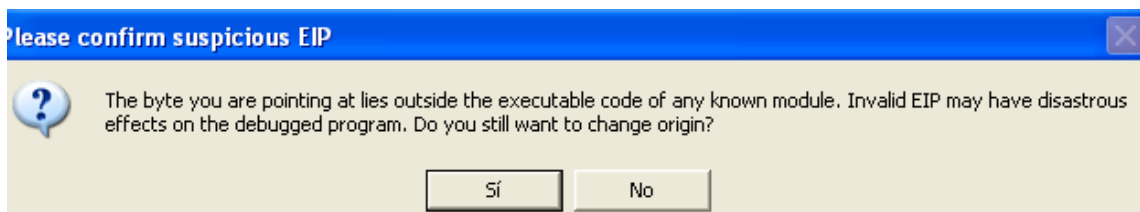
**Esto pasa por usar createFileA, la forma de anular es buscar el handle y anularlo**

00000038	File	2.	00120089	Size 80896.	f:\26-8-2006\escri\CONCURSO+94\DAp v0.2\Copia (2) de DAP v0.2.exe
0000008C	File (dir)	2.	00100020		f:\26-8-2006\escri\CONCURSO+94\DAp v0.2
0000009C	File (dir)	2.	00100020		c:\WINDOWS\Microsoft Windows Common-Controls_659586411A466146_x-ww

En este caso es 38 y llamar a CloseHandle mediante

push 38 y call a esa api,

cambio el origen de eip



00402057	6A 38	PUSH 38
00402059	E8 197B407C	CALL kernel32.CloseHandle
0040205E	C705 B05A4000 0	MOV DWORD PTR DS:[405AB0],Copia_(2.00400000
00402068	90	NOP
00402069	90	NOP
0040206A	90	NOP
0040206B	E8 5E140000	CALL <JMP.&comet132.InitCommonControls>
00402070	6A 00	PUSH 0

Y ahora podemos **dumpear tranquilamente** con cualquier dumper. al llegar a 40205e:

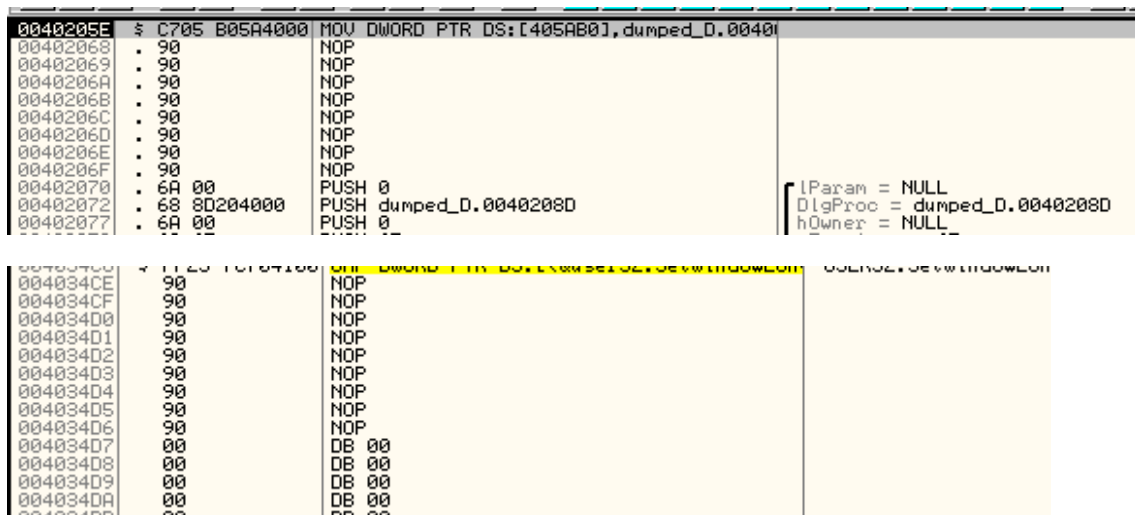
Section	Virtual Size	Virtual Offset	Raw Size	Raw Offset	Characteristics
.text	000024D4	00001000	000024D4	00001000	E0000020
.rdata	00000324	00004000	00000324	00004000	40000040
.data	00000AD8	00005000	00000AD8	00005000	C0000040
.rsrc	0000C010	00006000	0000C010	00006000	40000040
	00004000	00013000	00004000	00013000	E0000020

Usar import rec

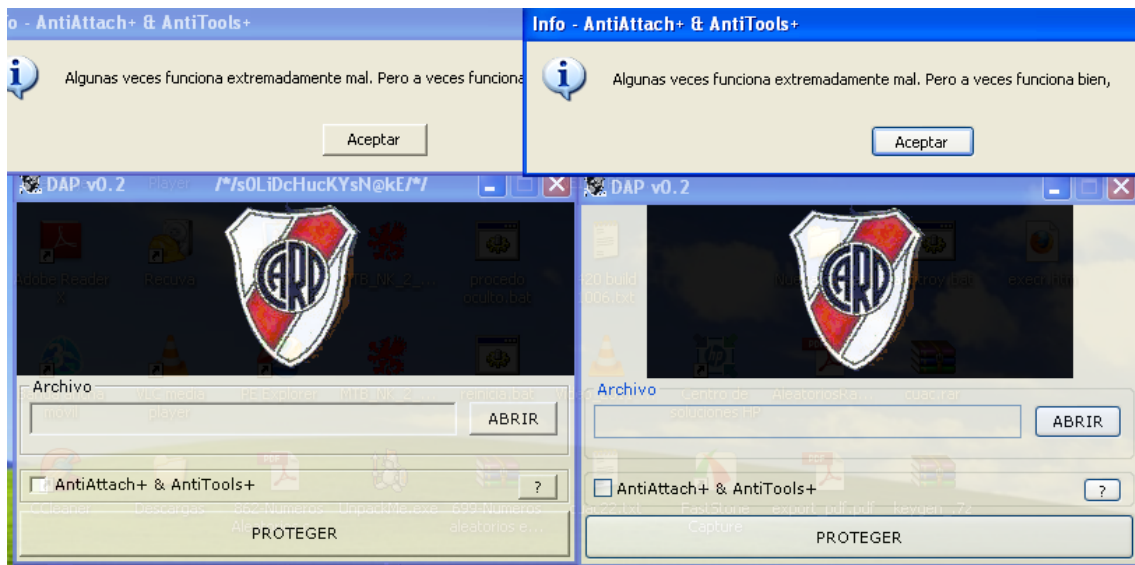
Y al objetivo: unpacked, cambiamos el titulo con PExplorer R6

Y le agregamos el manifiest

Ahora elimino el init, porque a la larga es como un call +ret=nop



Y ahora tengo tranquilamente el Dap 0.2 unpacked y con mejor apariencia en los botones.



Eso es todo, saludos Apuromafo