

Indice

Introducción .....	3
Herramientas usadas: .....	3
<b>Comenzando a presentar el programa .....</b>	<b>3</b>
Limitaciones: .....	3
Comenzamos la depuración BUSCANDO LA LIMITACION.....	4
Limitaciones reales.....	8
Parchando WATERMARK COMO STRING .....	8
Parchando titulo demo .....	8
GUARDAR COMO...solo para la primera pagina.....	9
Parchando el certificado. ....	10
Agregando la nueva seccion para la extensión de código .....	11
USANDO MULTILINE ULTIMATE ASSEMBLER .....	11
USANDO MULTILINE ULTIMATE ASSEMBLER con el depurador .....	14
Ultima limitación, ejecución de web sin nuestro consentimiento. ....	19

INTRODUCCIÓN

Programa	Spdfedit 5.5
Descarga	http://www.cadkas.com/spdfedit!.exe
Dificultad	Depende de quien lo mire.
Información	http://www.cadkas.com/pdf-editor-spanish.php
Herramientas usadas	X64dbg ,PID
Fecha Creación Tutorial	22/07/2017
Fecha Compartir Tutorial	08/08/2017
Cracker	Apuromafo

“Deja que Hablen de ti. Total, tú sabes quién eres, qué haces, qué hiciste, qué dijiste y ellos no.” Anónimo

HERRAMIENTAS USADAS:

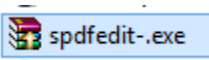
Herramienta	Descarga	Utilidad
Procesador de texto	(está incluido con el suite de office)	Para redactar el tutorial
Sharex	https://getsharex.com/	Para capturar las imágenes
Everything	http://www.voidtools.com/	Para buscar los archivos en el pc
Winrar	https://www.winrar.es/descargas	Compresor/Descompresor de archivos
7zip	http://www.7-zip.org/	Compresor/Descompresor de archivos
X64dbg	http://x64dbg.com/	Depurador
PID	https://web.archive.org/web/20170620171730/http://pid.gcwstorage.xyz/dl.php?f=ProtectionId.685.December.2016.rar	Analizador de Ejecutables
TOPO	http://ricardonarvaja.info/WEB/OTROS/HERRAMIENTAS2/Q-R-S-T-U/topo12corregido.rar	Permite Agregar una sección
Section Adder v0.1	https://reversingfiles.info/dls/tools/SecTion.Adder.0.1-Tool_CiM.zip?Submit=Download	Permite Agregar una sección
CffExplorer	www.ntcore.com/exsuite.php	PE editor para editar a gusto. Permite explorar un ejecutable x86 /x64 sin forma de ver bien los form
Pexplorer	http://www.heaventools.com/download-pe-explorer.htm	Permite explorar un ejecutable x86 , viendo bien los form
Resource hacker	http://www.angusj.com/resourcehacker/	Permite modificar ejecutables sobre todo en la parte de recursos, incluyendo los form
IDR	https://web.archive.org/web/20170501145746/http://kpnc.org/idr32/en/ https://github.com/crypto2011/IDR	Analizador de Delphi (interactive delphi ...)
Uniextractor*	http://filehippo.com/es/download_universal_extractor/	Extractor de archivos

Comenzando a presentar el programa

Bienvenidos a esta pequeña lectura e historia explorando un programa demo o capado, pero no significa que no podamos aprender cosas nuevas o extender la funcionalidad, me he dedicado a depurar este programa en breves lapsos y hasta entonces me concentré en hacer parches funcionales intentando entender o comprender lo que estaba haciendo .

¿Setup.exe?

La primera impresión es ver el setup que proviene de la web (sin análisis, vemos que usa un stub de Winrar )



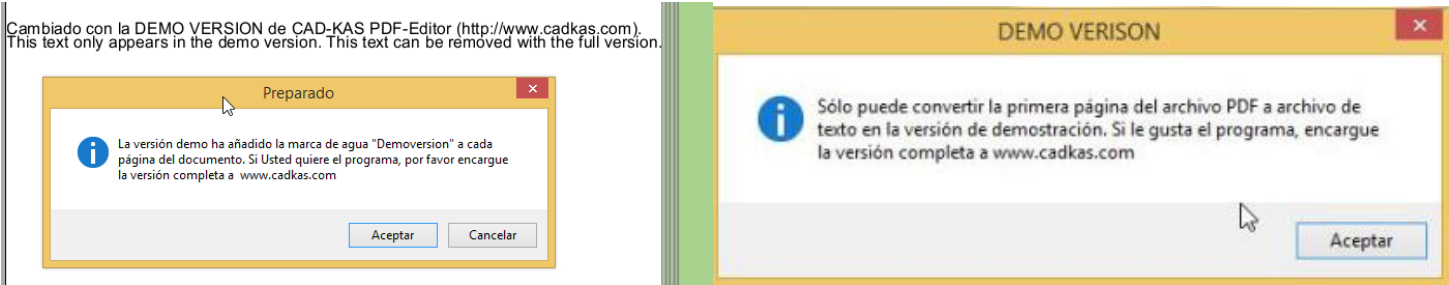
<img1 setup: Winrar sfx>

Luego es descomprimir con un descompresor, sea Winrar o 7Zip , con ello podemos ver los archivos más llamativos:

me muestra que desinstala:  <img2 unistaller: pdf edit>	Ejecutable Principal que vamos a depurar:  <img3 Programa a depurar >
---	---

LIMITACIONES:

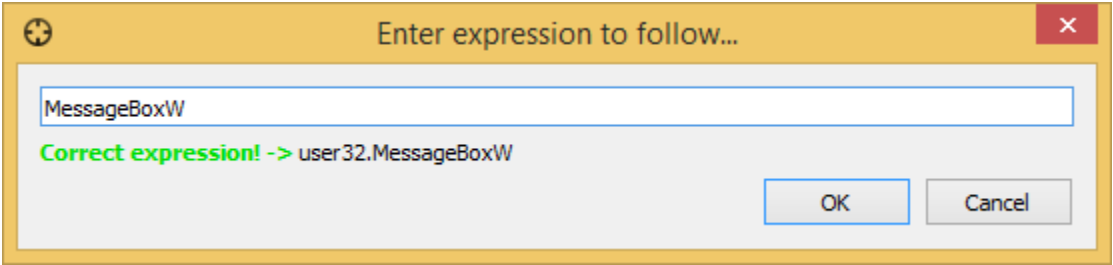
Si vamos al sitio web refiere de forma muy vaga, que la versión pro puede crear botón en el pdf , además otros tipos de checkbox y opciones (controles), pero al pulsar en el demo los botones si deja , aquí viene la pregunta, entonces ¿ cuales son las limitaciones? Solo puedo afirmar que el programador nos avisará de forma oportuna. Ahora bien, si yo creo un programa versión 5.1 y cambio algo, genero un programa 5.2, pero aquí modifican cada semana y siempre sigue siendo la versión 5.5 asumo que lo cambian por presencia de varios errores (asumo que debe ser por eso), en su web no es clara su limitación, así que solo espero encontrarlos mientras lo usemos , guardo el archivo pdf y aparece el primer mensaje.



<img4 Limitación del programa: watermark e img 5 Conversión solo de la Primera página en muchas opciones >

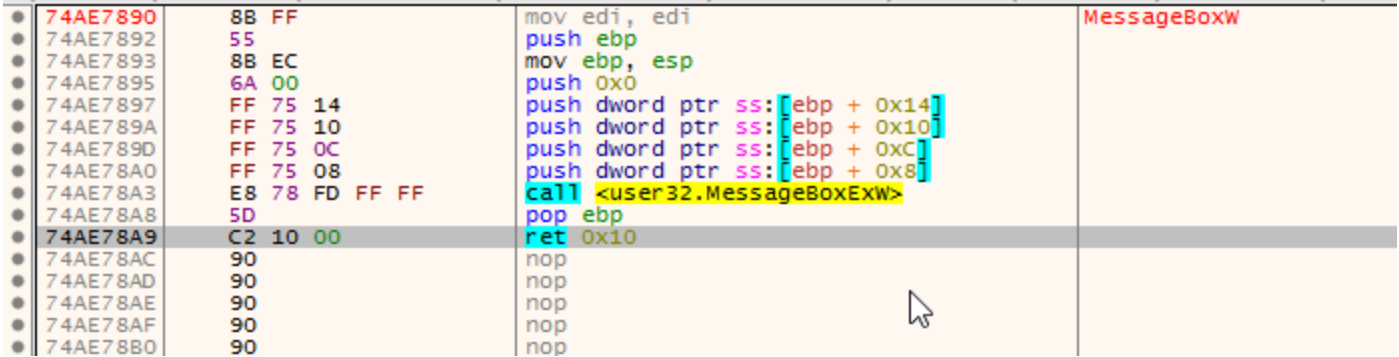
COMENZAMOS LA DEPURACIÓN BUSCANDO LA LIMITACION

Así que para poder comenzar, necesitamos un pdf cualquiera con más de una página para poder probar hasta donde llega, hacemos la acción de “guardar archivo como”, para luego leer un nag , este mensaje informativo usa MessageBoxA/W pero depurando he visto la letra W, así que comenzaré con este BP(BreakPoint). Comenzamos el programa en el depurador favorito x64dbg (que recordemos que tiene 2 opciones uno para x64 y x86, este programa está en x86 por lo cual depurará con x32dbg ) , no está empacado, se logran apreciar algunas String o palabras, pero nada muy decisivo que me diga “DEMO VERISION”, nos muestra un mensaje que han añadido una marca de agua, esto será nuestro comienzo y de ahí a retornar API para comenzar, pulsamos para seguir una expresión, en este caso la API (de mensaje)



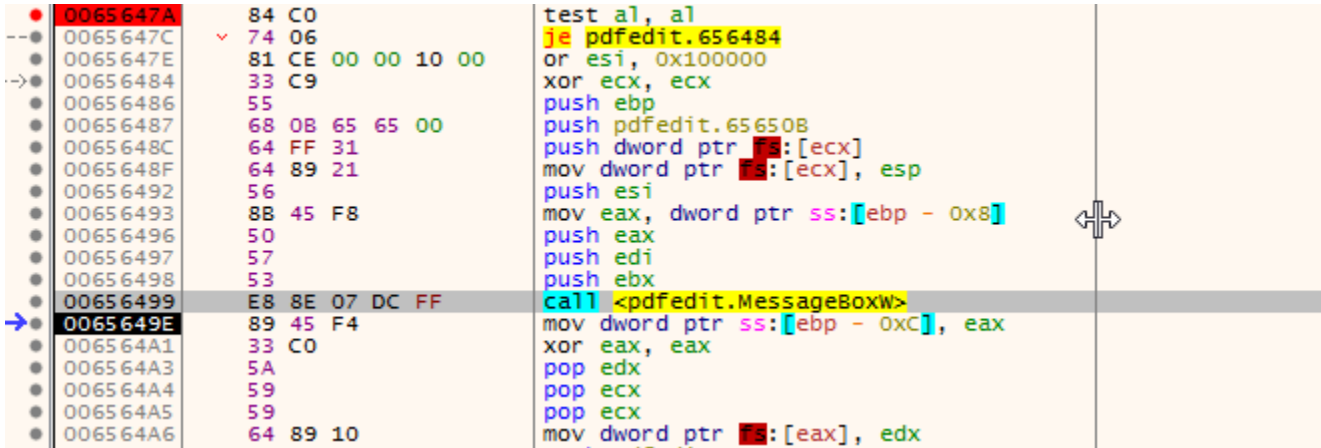
<img6 : Api MessageBox en x32dbg >

Me importa el retorno (donde termina el mensaje) para saber quien lo ha llamado.



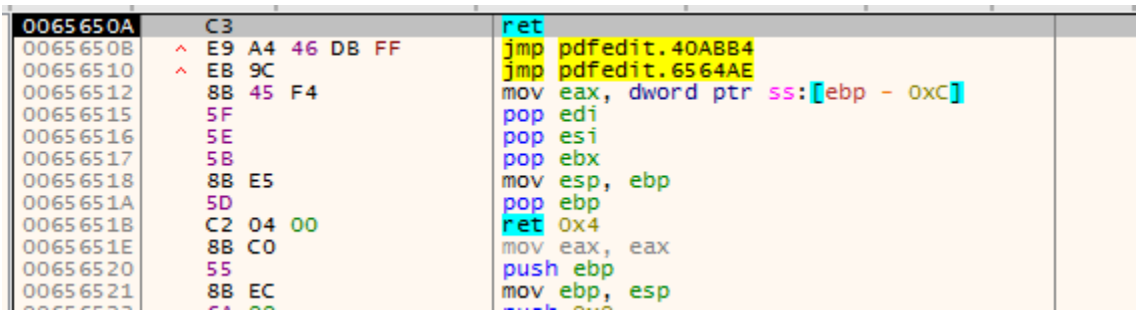
<img7 : retorno Api MessageBox en x32dbg >

al colocar ok (en el mensaje ) retorno aquí llegamos (donde fue llamado)



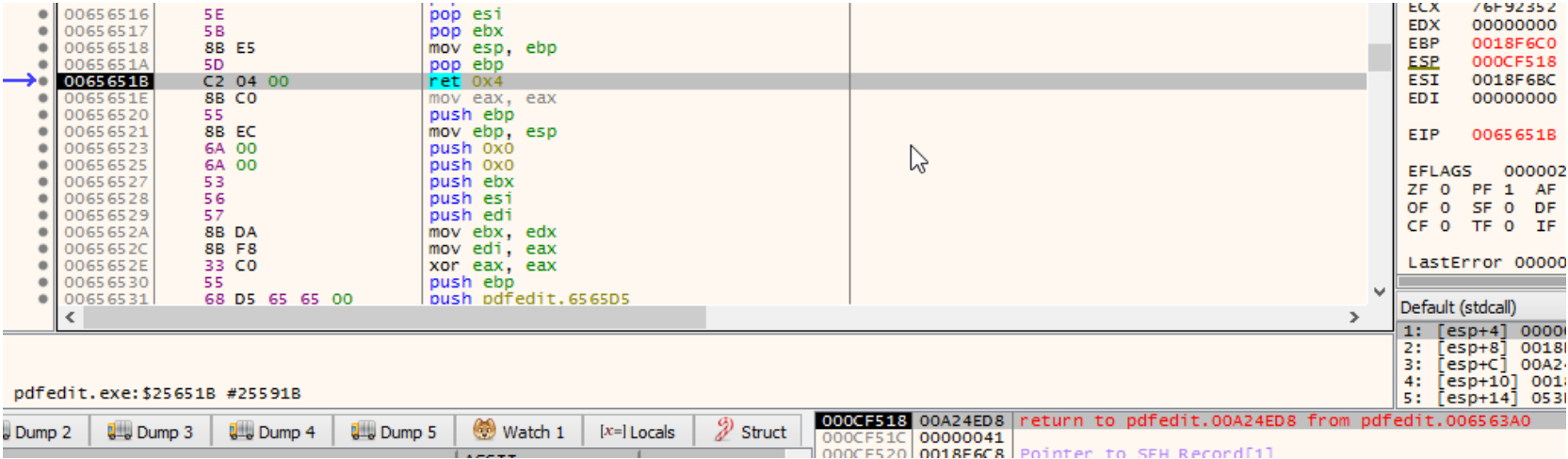
<img8: llamada de MessageBox en x32dbg >

Primer ret va hacia un poco más abajo (ret)



<img9 : retorno de función llamada en x32dbg >

Y al llegar al ret decisivo salta a donde fue invocado (ret4)



<img 10 : retorno de función 2 llamada en x32dbg >

Como se ve un call bastante estructurado (edx,eax ecx), esto es común cuando se usan translate (traducción o multi/idioma) asi que puede que el formato sea similar, hay que mirar mas arriba... para saber quien realmente ha llamado el titulo y mensaje

00A24EC4	E8 73 77 9E FF	call pdfedit.40C63C	
00A24EC9	8B D0	mov edx, eax	
00A24EC8	A1 FC 83 B7 00	mov eax, dword ptr ds:[0xB783FC]	
00A24ED0	8B 00	mov eax, dword ptr ds:[eax]	
00A24ED2	59	pop ecx	
00A24ED3	E8 C8 14 C3 FF	call pdfedit.6563A0	llamada 1
00A24ED8	48	dec eax	
00A24ED9	0F 94 45 DF	sete byte ptr ss:[ebp - 0x21]	
00A24EDD	33 C0	xor eax, eax	mm2
00A24EDF	5A	pop edx	

<img11 : retorno de función llamada en x32dbg , se transforma en la primera llamada >

Esta llamada 1, corresponde al mensaje de showmessagebox...pero todo se preparó antes...

Para cualquier persona que mira un programa que ha invocado un messagebox, intenta saber de dónde ha salido el mensaje, pero cuando no hay string, esto solo significa algo , hay que pensar que el autor ha pensado alguna forma más general o recursiva, uso de índices o algún array para usar el mismo lugar para diferentes palabras, veamos si encuentro algo al respecto, Aquí se ven como hay 2 mov eax y algo mas curioso, creo que no hay muchos escritos que hablen de esto.

A24E98	B8 4E 00 00 00	mov eax, 0x4E	mov eax,value
A24E9D	E8 32 F7 D7 FF	call pdfedit.7A45D4	
A24EA2	8B 85 7C FE F3 FF	mov eax, dword ptr ss:[ebp - 0xC0184]	
A24EA8	E8 8F 77 9E FF	call pdfedit.40C63C	
A24EAD	50	push eax	
A24EAE	8D 95 78 FE F3 FF	lea edx, dword ptr ss:[ebp - 0xC0188]	
A24EB4	B8 4D 00 00 00	mov eax, 0x4D	4D: 'M'
A24EB9	E8 16 F7 D7 FF	call pdfedit.7A45D4	
A24EBE	8B 85 78 FE F3 FF	mov eax, dword ptr ss:[ebp - 0xC0188]	
A24EC4	E8 73 77 9E FF	call pdfedit.40C63C	
A24EC9	8B D0	mov edx, eax	

<img12 : indice con eax, para armar MessageBox >

Mov eax,value, esto es sospechoso porque son valores muy pequeños, y con llamadas muy similares, esto solo significa que tiene un índice para diversos mensajes, dependiendo de ellos va creando el mensaje y luego va juntando las opciones

Seamos prácticos, si hay valores dados asi “mov eax, constante” , podemos encontrar todo lo que pasa en el programa y no tendremos que hacer gran exploración, pero bueno veamos que me enfrento , pero como tenemos tiempo , seguiremos depurando

Esta pequeña tabla posee valores entre 1 Hex y c4 (hex) , osea hay mucho que está adentro..veamos en decimal es mas fácil de entender (entre 1 y 196 mensajes)

007A45D4	53	push ebx	
007A45D5	56	push esi	
007A45D6	8B F2	mov esi, edx	
007A45D8	8B D8	mov ebx, eax	
007A45DA	83 FB 01	cmp ebx, 0x1	
007A45DD	7C 16	j1 pdfedit.7A45F5	
007A45DF	81 FB C4 00 00 00	cmp ebx, 0xC4	
007A45E5	7F 0E	jg pdfedit.7A45F5	
007A45E7	8B D6	mov edx, esi	
007A45E9	8B 04 9D 0C 0F B6 00	mov eax, dword ptr ds:[ebx * 4 + 0xB60F0C]	
007A45F0	E8 03 F2 FF FF	call pdfedit.7A37F8	
007A45F5	5E	pop esi	
007A45F6	5B	pop ebx	
007A45F7	C3	ret	
007A45F8	55	push ebp	
007A45F9	8B FC	mov ebp, esp	

<img13 : Rutina Importante para mensajes >

Colocado el bp reinicio el programa, ahora vere sus mensajes , el primero refiere optimizing, (sigo ejecutando el programa con f9) , no es relevante

007A45D1	8D 40 00	lea eax, dword ptr ds:[eax]	
007A45D4	53	push ebx	
007A45D5	56	push esi	
007A45D6	8B F2	mov esi, edx	
007A45D8	8B D8	mov ebx, eax	
007A45DA	83 FB 01	cmp ebx, 0x1	
007A45DD	7C 16	j1 pdfedit.7A45F5	
007A45DF	81 FB C4 00 00 00	cmp ebx, 0xC4	
007A45E5	7F 0E	jg pdfedit.7A45F5	
007A45E7	8B D6	mov edx, esi	
007A45E9	8B 04 9D 0C 0F B6 00	mov eax, dword ptr ds:[ebx * 4 + 0xB60F0C]	[ebx*4+B60F0C]:L"Optimizing to:"
007A45F0	E8 03 F2 FF FF	call pdfedit.7A37F8	
007A45F5	5E	pop esi	
007A45F6	5B	pop ebx	
007A45F7	C3	ret	
007A45F8	55	push ebp	
007A45F9	8B EC	mov ebp, esp	
007A45FB	6A 00	push 0x0	
007A45FD	6A 00	push 0x0	

<img14 : Verificando mensajes en x32dbg >

Es posible apreciar cadenas (mientras tanto pulso f9) que indican palabras alusivas a acciones que realizará el programa, en el primer inicio aparece la de la nag, “Image Artist”

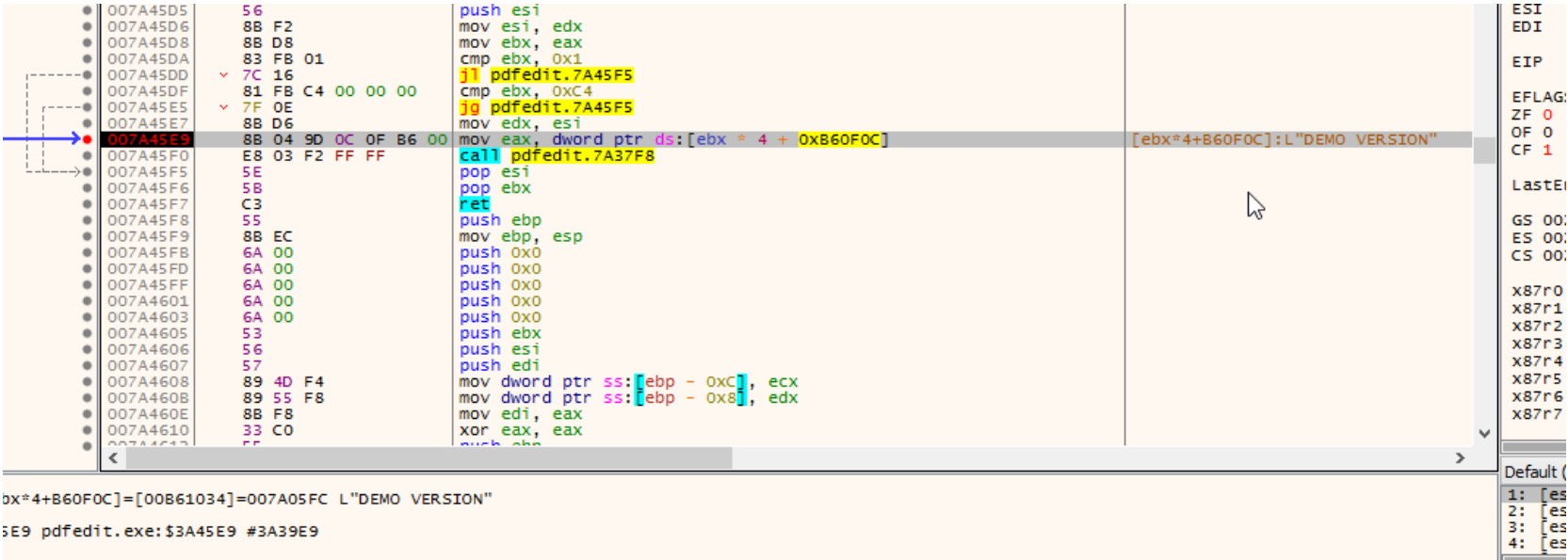
eax=55 'U'
dword ptr[ebx*4+B60F0C]=[00B61060]=007A0C0C L"PDF Editor: ImageArtist"
.text:007A45E9 pdfedit.exe:\$3A45E9 #3A39E9

<img15: Nag: ImageArtist desde x32dbg>

Luego refiere un “of”, luego proviene uno mas interesante “demo”, para saber entonces donde estamos, debemos llegar al “ret”, vemos lo siguiente

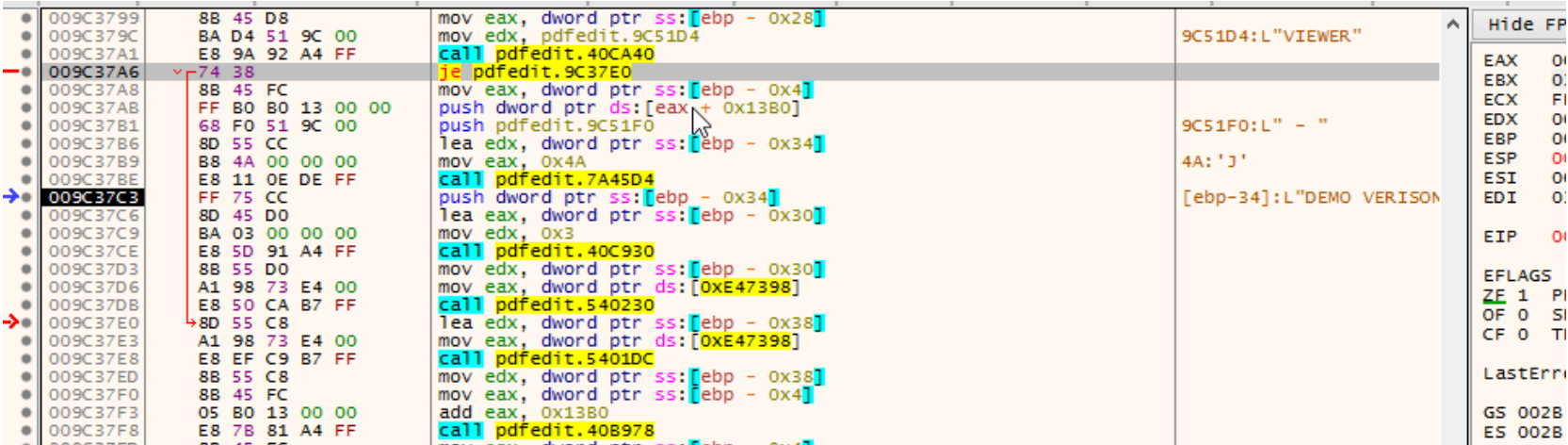


7A45f0 se puede apreciar la string “DEMO VERSION”



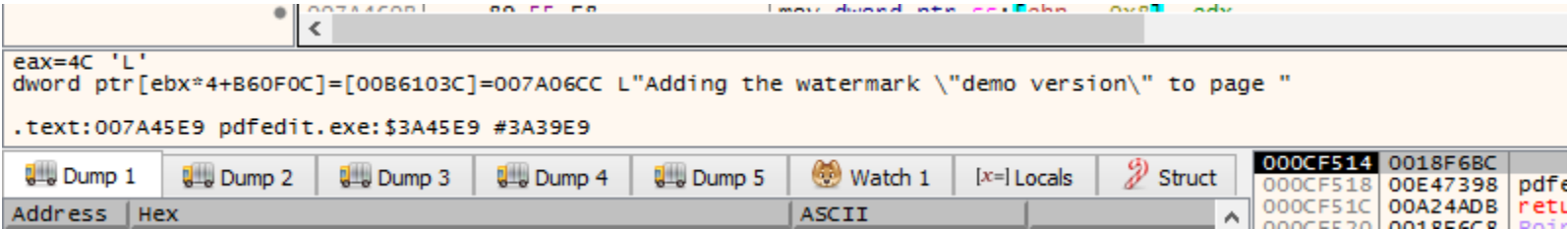
<img16 : Demo versión, posible limitación >

Sobre este demo, llego al ret, al volver vemos la siguiente rutina, la cual claramente tiene un condicional fácil de modificar , debemos cambiar ese salto de je a jmp para que no tengamos en el titulo del programa esta marca.



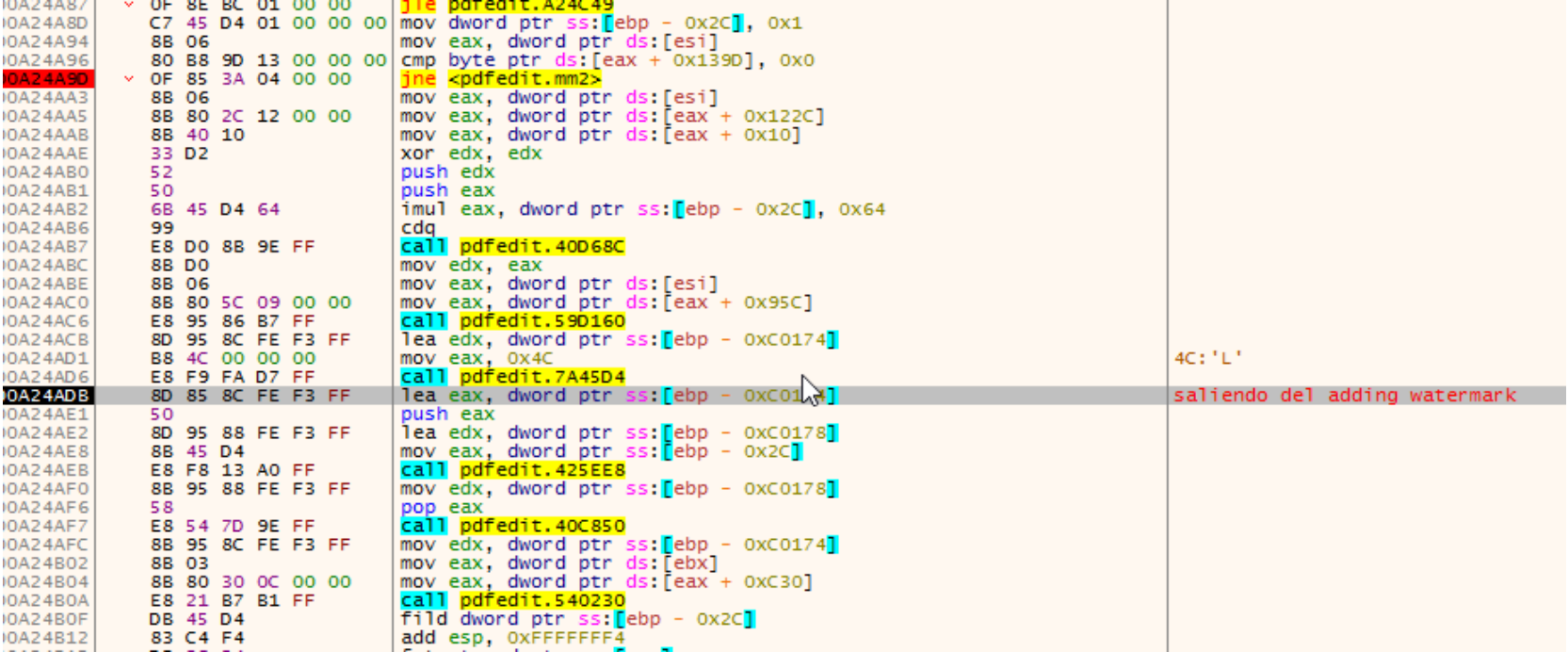
<img17 : Demo versión, Título del programa >

cuando abrimos el programa agregamos el pdf, podemos ver como nos va diciendo cuantas paginas tiene, verifica paginas borradas y más, llega el minuto final donde no hay mas string llamativas, procedo a guardar como (guardar archivo) en el programa y muestra así:



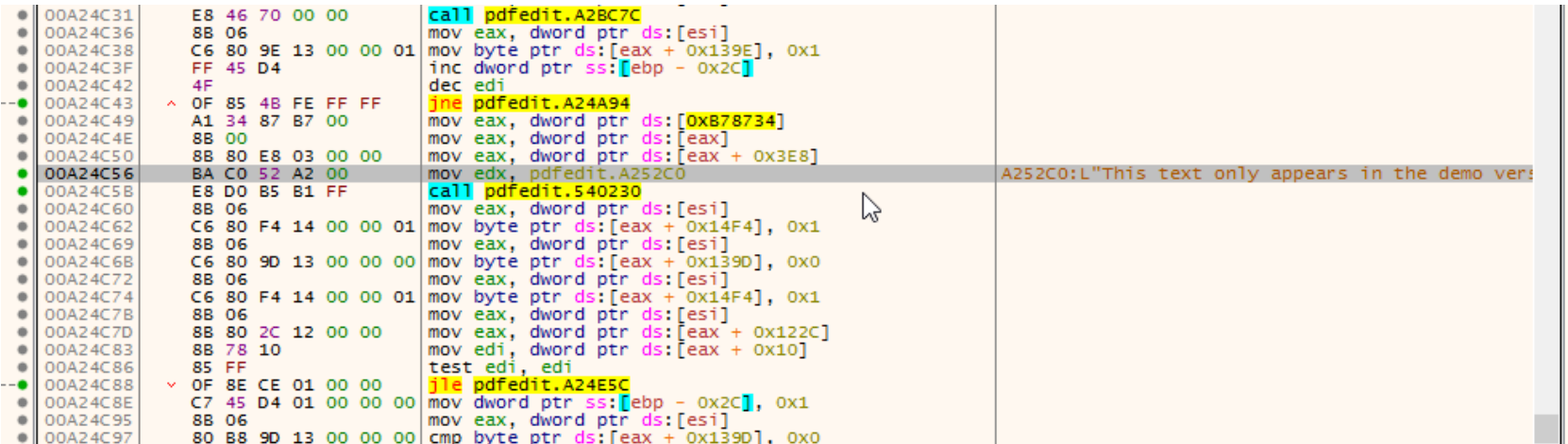
<img18: Segunda Limitación,Watermark >

Si traduzco dice agregando marca de agua “demo version a la pagina”, ósea nos dice claramente que nos está agregando el watermark, wow , que expresivo, retorno para ver de dónde viene:



<img19: saliendo de la generación de palabra add Watermark >

Por lo que al mirar mas arriba vemos entre ensayo error, para que sirve , si asumo que son saltos chico bueno y malo como en el anterior, el programa no procesa, así que realmente estos saltos son para saber si procesa o no la página en esa porción de código, por lo cual no hay nada que me permita pensar que realmente no me crea el watermark con la presencia de algún archivo o licencia pues realmente lo hará si o si, asi que comenzamos a parchar este watermark, vemos hacia un pequeño scroll (bajando) :



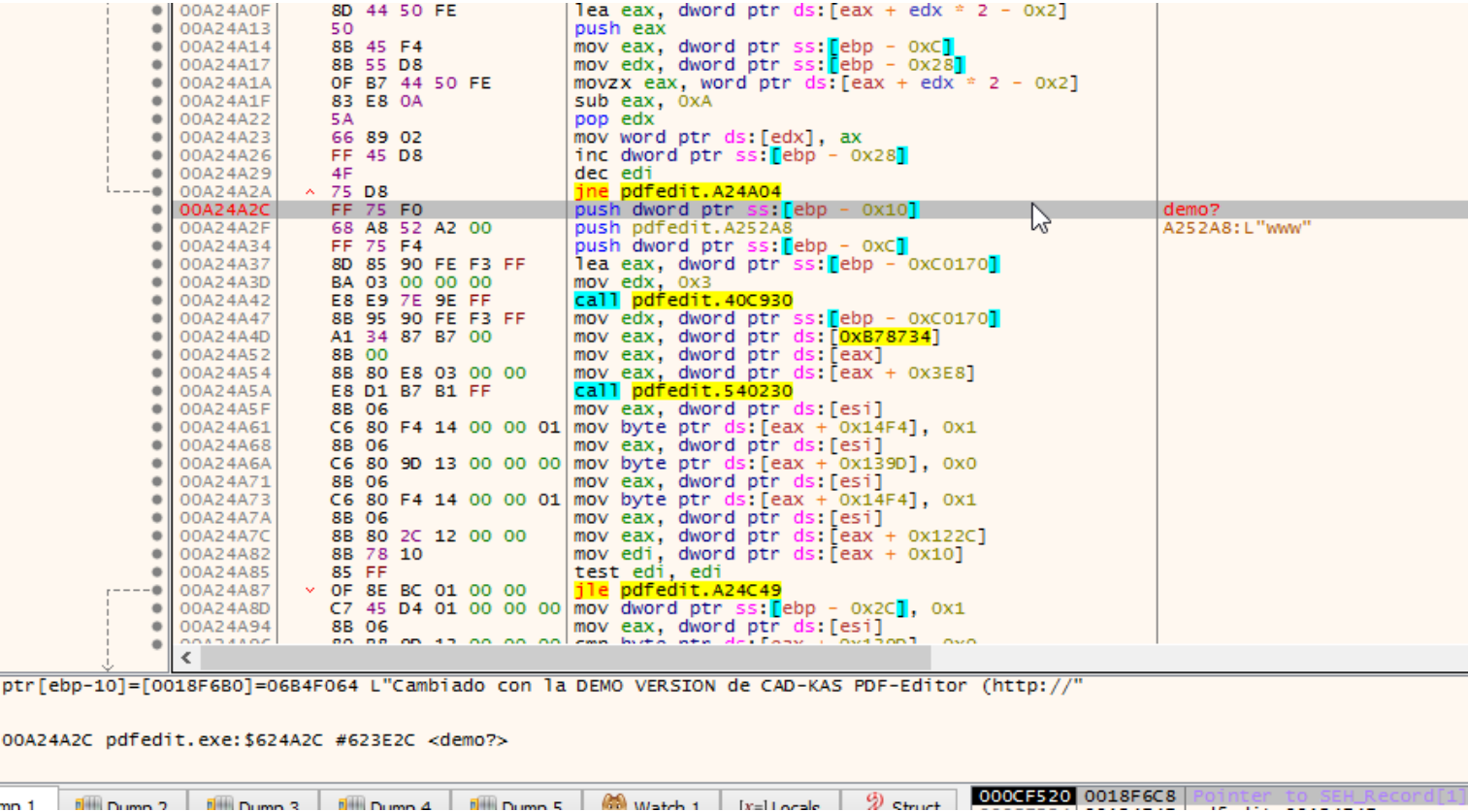
<img20: Watermark este texto...>

En traducción dice este texto solo aparece en la versión demo del programa., pero no veo donde lo ha escrito, esto significa otra cosa, hay además algún encode/decode si subo mucho mas arriba encuentro al responsable tiene el mismo largo ☺



<img21: Watermark cambiado con la versión...>

Ahí está todo cifrado y más encima el descifrado arriba , no hay misterio, ahora También se aprecia un www



<img22: Watermark www...>

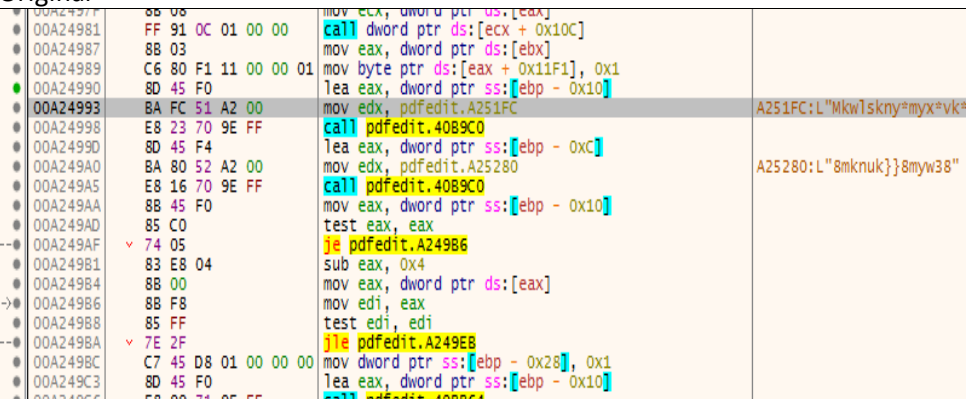
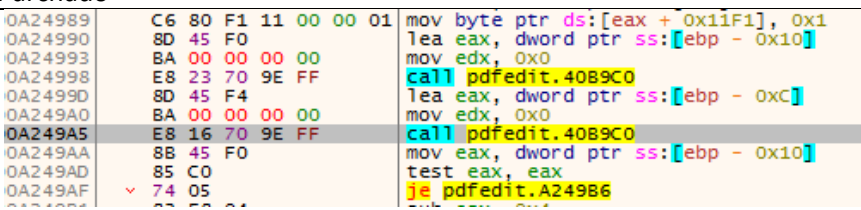

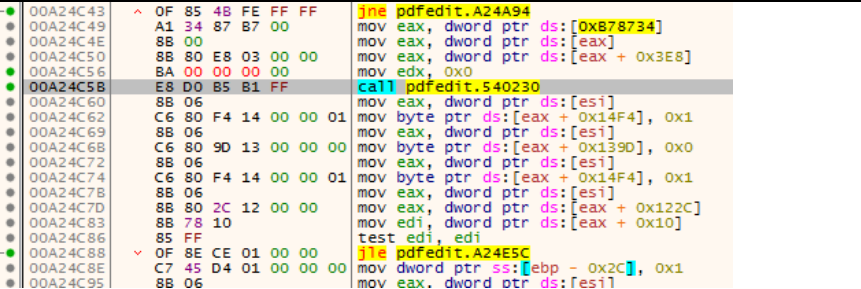
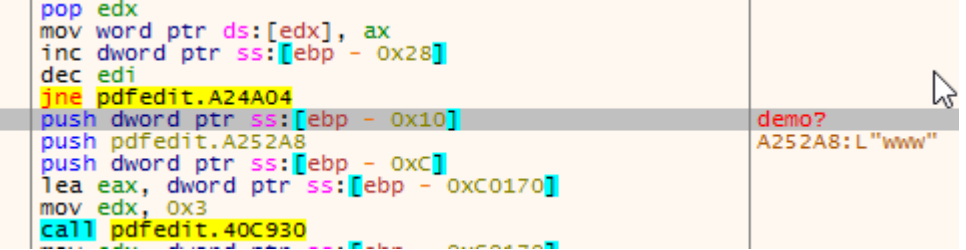
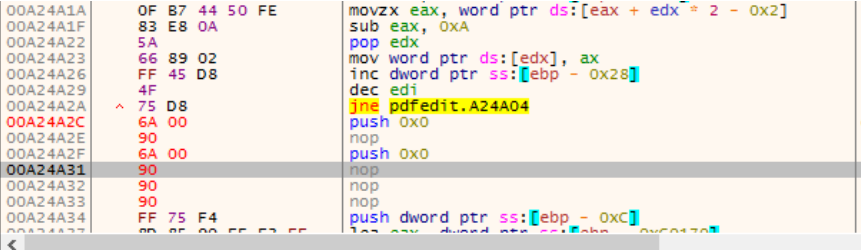
LIMITACIONES REALES

Es bueno hacer una lista y que haremos:

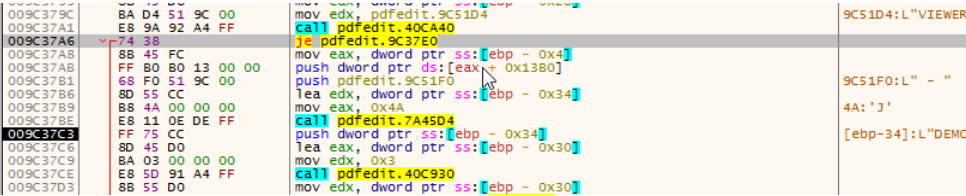
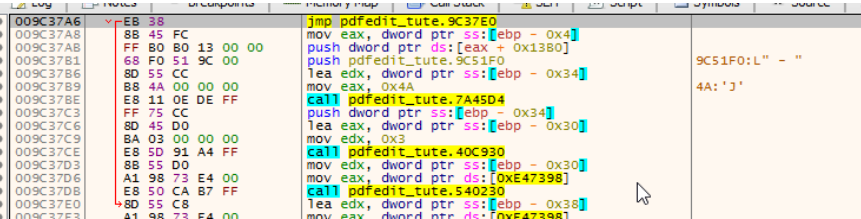
- 1) Titulo con mensaje demo (se cambia de je a jmp)
- 2) Watermark( se cambia anulando la string ,como lo haré ahí veremos)
- 3) Limitación de páginas (se extiende mediante Multiline Ultimate Assembler en x32dbg) y topo o alguna tool similar.
- 4) Certificado Digital (se anula con cffexplorer o pexplorer)
- 5) Menús refiriendo solo pro(se edita con resource hacker)
- 6) Funciones capadas(se edita con resource hacker para anularlas)
- 7) Abre la pagina web del autor al cerrar el programa

PARCHANDO WATERMARK COMO STRING

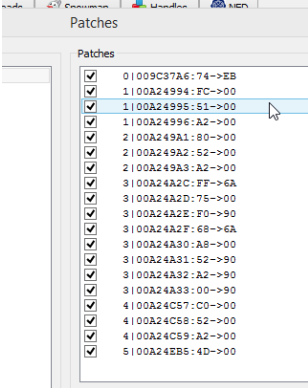
Creo que en resumen eso tenemos, asi que debemos comenzar a anular de la mejor forma Si anulo el salto, no habrá donde guardar(pero si anulo el string, no puede escribir bytes nulos, buena idea) el parche será colocar el valor a 0 , Respecto mas arriba el cifrado anulamos igual con edx, 0

Original	Parchado
<div></div> <div>&lt;img23: watermark cifrado &gt;</div>	<div></div> <div>&lt;img24: watermark anulado &gt;</div>
<div></div> <div>&lt;img25: watermark this text &gt;</div>	<div></div> <div>&lt;img26: this text anulado &gt;</div>
<div></div> <div>&lt;img27: www. (watermark) &gt;</div>	<div></div> <div>&lt;img28: www anulado &gt;</div>

PARCHANDO TITULO DEMO

<div></div> <div>&lt;img29: Titulo Demo &gt;</div>	<div></div> <div>&lt;img30: Titulo Demo anulado &gt;</div>
--	---

En resumen llevamos parchado 6 lugares, (de string y de titulo demo)





GUARDAR COMO...SOLO PARA LA PRIMERA PAGINA

La estrategia para encontrar de donde venia el mensaje me permite esta vez saber donde ha sido llamado, encontramos la llamada aquí:

00A74DCA	E8 9D D8 F4 FF	call pdfedit.9C266C	
00A74DCF	6A 40	push 0x40	
00A74DD1	8D 95 E4 FE FF FF	lea edx, dword ptr ss:[ebp - 0x11C]	
00A74DD7	88 4A 00 00 00	mov eax, 0x4A	4A: 'J'
00A74DDC	E8 F3 F7 D2 FF	call pdfedit.7A45D4	
00A74DE1	88 85 E4 FE FF FF	mov eax, dword ptr ss:[ebp - 0x11C]	
00A74DE7	E8 50 78 99 FF	call pdfedit.40C63C	
00A74DEC	50	push eax	
00A74DED	8D 95 E0 FE FF FF	lea edx, dword ptr ss:[ebp - 0x120]	
00A74DF3	88 71 00 00 00	mov eax, 0x71	71: 'q'
00A74DF8	E8 D7 F7 D2 FF	call pdfedit.7A45D4	
00A74DFD	88 85 E0 FE FF FF	mov eax, dword ptr ss:[ebp - 0x120]	
00A74E03	E8 34 78 99 FF	call pdfedit.40C63C	
00A74E08	88 D0	mov edx, eax	
00A74E0A	A1 FC 83 87 00	mov eax, dword ptr ds:[0x8783FC]	
00A74E0F	88 00	mov eax, dword ptr ds:[eax]	
00A74E11	59	pop ecx	
00A74E12	E8 89 15 BE FF	call pdfedit.6563A0	llamada de txt
00A74E17	88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74E1A	88 80 4C 09 00 00	mov eax, dword ptr ds:[eax + 0x94C]	
00A74E20	B2 01	mov dl, 0x1	
00A74E22	88 08	mov ecx, dword ptr ds:[eax]	
00A74E24	FF 91 88 00 00 00	call dword ptr ds:[ecx + 0x88]	
00A74E2A	33 C0	xor eax, eax	
00A74E2C	5A	pop edx	
00A74E2D	59	pop ecx	
00A74E2E	59	pop ecx	
00A74E2F	64 89 10	mov dword ptr ds:[eax], edx	
00A74E32	68 7F 4E A7 00	push pdfedit.A74E7F	
00A74E37	8D 85 E0 FE FF FF	lea eax, dword ptr ss:[ebp - 0x120]	
00A74E3D	BA 08 00 00 00	mov edx, 0x8	

<img32: Buscando la limitación 3 >

Si vemos la función que nos da la limitación también usa indice y la otra que tenemos pendiente de su llamada a messagebox

00A74DB3	> 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74DB6	. 88 80 D0 03 00 00	mov eax, dword ptr ds:[eax + 0x3D0]	
00A74DBC	. E8 97 F8 B1 FF	call <pdfedit.sub_594658>	
00A74DC1	. 48	dec eax	
00A74DC2	. 75 08	jne pdfedit.A74DCF	
00A74DC4	. 88 55 E4	mov edx, dword ptr ss:[ebp - 0x1C]	
00A74DC7	. 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74DCA	. E8 9D D8 F4 FF	call <pdfedit.sub_9C266C>	
00A74DCF	> 6A 40	push 0x40	
00A74DD1	. 8D 95 E4 FE FF FF	lea edx, dword ptr ss:[ebp - 0x11C]	
00A74DD7	. 88 4A 00 00 00	mov eax, 0x4A	4A: 'J'
00A74DDC	. E8 F3 F7 D2 FF	call <pdfedit.sub_7A45D4>	
00A74DE1	. 88 85 E4 FE FF FF	mov eax, dword ptr ss:[ebp - 0x11C]	[ebp-11C]:L"DEMO VERISON"
00A74DE7	. E8 50 78 99 FF	call <pdfedit.sub_40C63C>	
00A74DEC	. 50	push eax	
00A74DED	. 8D 95 E0 FE FF FF	lea edx, dword ptr ss:[ebp - 0x120]	
00A74DF3	. 88 71 00 00 00	mov eax, 0x71	71: 'q'
00A74DF8	. E8 D7 F7 D2 FF	call <pdfedit.sub_7A45D4>	
00A74DFD	. 88 85 E0 FE FF FF	mov eax, dword ptr ss:[ebp - 0x120]	
00A74E03	. E8 34 78 99 FF	call <pdfedit.sub_40C63C>	
00A74E08	. 88 D0	mov edx, eax	

<img33: Buscando la limitación 3 >

Pillamos denuevo el demo versión

00A74DC1	. 48	dec eax	
00A74DC2	. 75 08	jne pdfedit.A74DCF	
00A74DC4	. 88 55 E4	mov edx, dword ptr ss:[ebp - 0x1C]	
00A74DC7	. 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74DCA	. E8 9D D8 F4 FF	call <pdfedit.sub_9C266C>	
00A74DCF	> 6A 40	push 0x40	
00A74DD1	. 8D 95 E4 FE FF FF	lea edx, dword ptr ss:[ebp - 0x11C]	
00A74DD7	. 88 4A 00 00 00	mov eax, 0x4A	4A: 'J'
00A74DDC	. E8 F3 F7 D2 FF	call <pdfedit.sub_7A45D4>	demo
00A74DE1	. 88 85 E4 FE FF FF	mov eax, dword ptr ss:[ebp - 0x11C]	
00A74DE7	. E8 50 78 99 FF	call <pdfedit.sub_40C63C>	
00A74DEC	. 50	push eax	
00A74DED	. 8D 95 E0 FE FF FF	lea edx, dword ptr ss:[ebp - 0x120]	
00A74DF3	. 88 71 00 00 00	mov eax, 0x71	71: 'q'
00A74DF8	. E8 D7 F7 D2 FF	call <pdfedit.sub_7A45D4>	
00A74DFD	. 88 85 E0 FE FF FF	mov eax, dword ptr ss:[ebp - 0x120]	you can only
00A74E03	. E8 34 78 99 FF	call <pdfedit.sub_40C63C>	
00A74E08	. 88 D0	mov edx, eax	
00A74E0A	. A1 FC 83 87 00	mov eax, dword ptr ds:[0x8783FC]	
00A74E0F	. 88 00	mov eax, dword ptr ds:[eax]	
00A74E11	. 59	pop ecx	
00A74E12	. E8 89 15 BE FF	call pdfedit.6563A0	llamada de txt
00A74E17	. 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74E1A	. 88 80 4C 09 00 00	mov eax, dword ptr ds:[eax + 0x94C]	
00A74E20	. B2 01	mov dl, 0x1	
00A74E22	. 88 08	mov ecx, dword ptr ds:[eax]	

<img34: Buscando la limitación 3 >

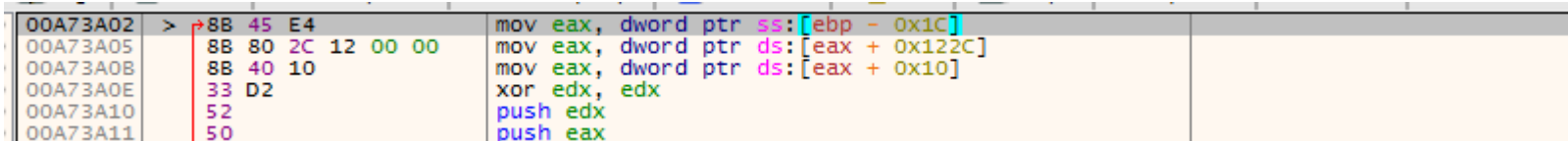
El tema es aprovechar bien el espacio que tenemos, porque si alcanza no necesitamos una nueva sección, pero si crearemos una nueva sección tendrá un espacio suficiente para crear lo que podamos imaginar, solo haremos jmp (codecave o nuevo espacio) o push label +ret y ahí ejecutaremos las instrucciones faltantes... demo versión hace alusión que hay un mensajito o messagebox que me dice que es demo y solo procesará la primera página. ahora vamos a lo importante luego se dedica a procesar el archivo abierto y mueve valores a través de ebp, eax, hasta aquí cualquiera que hubiera depurado archivos de abrir, guardar, verá que es algo similar siempre.

00A74CD3	. 98	wait	
00A74CD4	> 4B	dec ebx	
00A74CD5	. 83 FB FF	cmp ebx, 0xFFFFFFFF	
00A74CD8	. 75 98	jne pdfedit.A74C72	
00A74CDA	> 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	importante
00A74CDD	. 88 80 30 12 00 00	mov eax, dword ptr ds:[eax + 0x1230]	
00A74CE3	. C6 80 E8 04 00 00 00	mov byte ptr ds:[eax + 0x4E8], 0x0	
00A74CEA	. 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74CED	. 88 80 30 12 00 00	mov eax, dword ptr ds:[eax + 0x1230]	
00A74CF3	. 88 00	mov eax, dword ptr ds:[eax]	
00A74CF5	. 33 D2	xor edx, edx	
00A74CF7	. E8 14 1F D7 FF	call <pdfedit.sub_7E6C10>	
00A74CFC	. 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74CFF	. 88 80 30 12 00 00	mov eax, dword ptr ds:[eax + 0x1230]	
00A74D05	. 88 00	mov eax, dword ptr ds:[eax]	
00A74D07	. 33 D2	xor edx, edx	
00A74D09	. 89 50 48	mov dword ptr ds:[eax + 0x48], edx	
00A74D0C	. FF 85 5C FF FF FF	inc dword ptr ss:[ebp - 0xA4]	
00A74D12	. 83 BD 5C FF FF FF 02	cmp dword ptr ss:[ebp - 0xA4], 0x2	
00A74D19	. 75 EC FF FF	jne pdfedit.A73A02	
00A74D1F	> 88 45 E4	mov eax, dword ptr ss:[ebp - 0x1C]	
00A74D22	. 88 80 5C 09 00 00	mov eax, dword ptr ds:[eax + 0x95C]	
00A74D28	. 33 D2	xor edx, edx	

<img35:Encontrada una la limitación 3 : Solo la primera pagina >

La porción de código dice que solo permite la primera página, el programa si maneja cuanto es el total, pero compara con una constante, los números mas comunes a buscar (si es solo la primera página...), o es el 1 o es el 2 o al revez 1 , 0 y menos 1 ,

en este caso busca en la porción (A74D09 a 00A74D12) el 2 (dado que mueve el dword a edx y luego lo incrementa 1 vez) ,dado que la pagina 0 no es ninguna al estar lista la primera, incrementa con 1 y tiene en hex el primer valor, si el valor es igual (jne) entonces no salta..y termina la nag..  
si el valor salta continua, pero donde salta, nada más ni nada menos hay una llamada a punteros del total de páginas (en esta parte donde salta, es tomado el final de las paginas.(segunda parte importante para ver)



<img36: Manejo del total de paginas >

En resumen estas 3 instrucciones me dan información adecuada

el total de paginas del archivo 00A73A02 mov eax, dword ptr ss:[ebp - 0x1C] 00A73A05 mov eax, dword ptr ds:[eax + 0x122C] 00A73A0B mov eax, dword ptr ds:[eax + 0x10]	la cantidad de pagina actual que estamos (ya procesó) 00A74CFC mov eax, dword ptr ss:[ebp - 0x1C] 00A74CFF mov eax, dword ptr ds:[eax + 0x1230] 00A74D05 mov eax, dword ptr ds:[eax]
--	---

como comparo las 2, si no hay más espacio...pues generando un espacio nuevo, aquí debemos entrar en algo no tan usado, llamado inline, debemos luego que llame a la página actual, que compara con las 3 instrucciones (que llama al total de páginas) y luego continuar el flujo a debajo de las 3 instrucciones, me es difícil de explicar pero lo intentaré hay 2 tipos de secuencias que llamare instrucción a y b , sea el a el que procesa la página actual y b el que me dice el total de páginas, debo comparar a con b para que sea un valor real y no sea el 2 que me ofrece entonces debo ordenar esta información, si llego y modifico el flujo sin mantenerlo crashear, debo hacer que la instrucción a, siga tal cual, luego agregar una comparación con una instrucción b que guardaré en otro lugar, luego que compare con un número relativamente grande para que pueda seguir procesando, el nuevo código en cuestión será:

```
//inline patch que indica que tenemos el flujo número total
mov eax, dword ptr [ebp - 0x1c]
    mov eax, dword ptr [eax + 0x122c]
    mov eax, dword ptr [eax + 0x10]
//y usamos la misma constante de la 1 para comparar con el valor total
    cmp dword ptr [ebp - 0xa4], eax
//si son distintos que siga procesando (pasara al loop normal), pero si son igual al final del documento entonces saltara al a73a02 donde es que termina
    jne 0xa73a02
```

Hasta aquí es entendible la idea, pero pensemos si salta se desajustará stack , entonces debemos ordenar en pushad/popad las nuevas instrucciones

La comparación con el 2, ahora será modificada a un valor mucho mayor..., y la comparación dentro del pushad/popad servirá para salir del bucle (del número mayor) Para este inline haremos uso del MultimateAssembler en x64dbg debemos definir algunas cosas eso si, dirección y las instrucciones, tenemos esto:

```
<00A74D0C> //dirección donde esta la comparación del texto

    inc dword ptr [ebp - 0xa4] //aquí le dice que termino la pagina 1
    pushad //pushad guarda registros
mov eax, dword ptr [ebp - 0x1c] // instrucción 1
    mov eax, dword ptr [eax + 0x122c] // instrucción 2
    mov eax, dword ptr [eax + 0x10] // instrucción 3
    cmp dword ptr [ebp - 0xa4], eax //comparación que realmente necesito
    je @L00000001 //si son iguales devuelve el flujo de lo que continua
    popad //devuelve el valor antes del injerto

@L00000008:
    cmp dword ptr [ebp - 0xa4],99999 //comparación con 99999 hex paginas
    jne 0xa73a02 //instrucción original que comparaba pagina actual versus numero gigante (antes decía 2)

@L00000001:
    mov eax, dword ptr [ebp - 0x1c] //instrucción que continua el flujo
```

Comenzando a parchar dado que ya tenemos las nociones básicas del programa y con esto podemos comenzar bien

## PARCHANDO EL CERTIFICADO.

Primer paso, debemos usar cff explorer para anular la sección de firma digital, recordemos que el programa posee una firma digital. Si usamos pexplorer basta colocar set to zero,

DATA DIRECTORIES

Certificate Table

00BBC20000001878✔Set to Zero

Directory Name	Virtual Address	Size
Export Table	00E67000h	00000099h
Import Table	00E5D000h	00008B88h
Resource Table	00E6A000h	0043BA00h
Exception Table		
Certificate Table	00BBC200h	00001878h
Relocation Table		

<img38: Anulando la sección del certificado digital con Pexplorer >

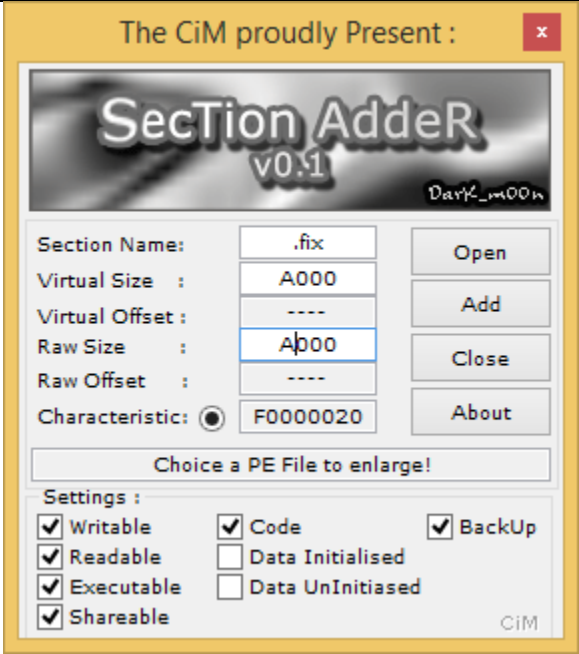
Security Directory RVA	00000198	Dword	00000000
Security Directory Size	0000019C	Dword	00000000
Relocation Directory RVA	000001A0	Dword	00000000

<img39: Anulando la sección del certificado digital con cffExplorer >

Si queremos hacerlo en cff explorer es agregar nueva sección, establecer los permisos y luego el tamaño.. es casi lo mismo, pero esto es mas visual y ordenado.

AGREGANDO LA NUEVA SECCION PARA LA EXTENSION DE CÓDIGO

Agregamos la sección con Section Adder,

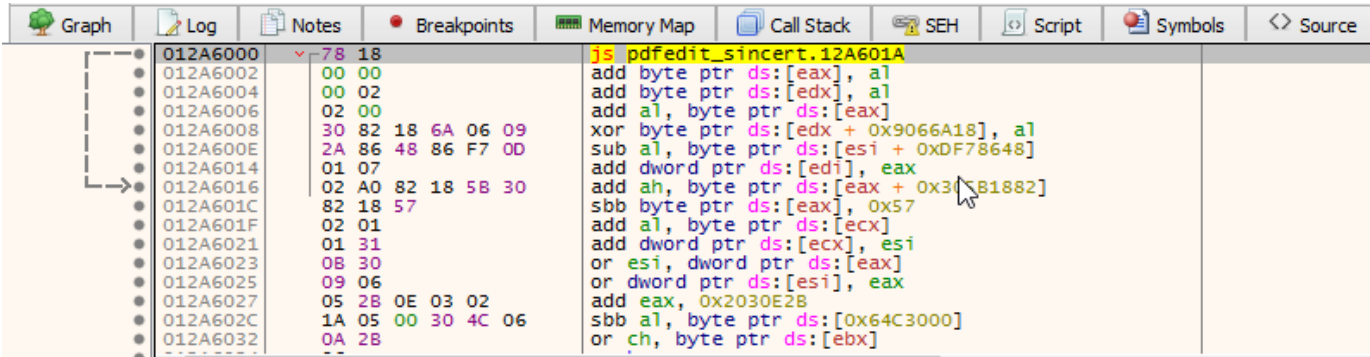


Comenzamos definimos nombre sección “.fix”, luego el tamaño (yo eleji A000 ) luego pulso El orden será (Open), luego Add (mostrará okey y que leamos el about xD) , luego Close para los permisos solo es pulsar el botón y elegir.

<img40: Agregando una nueva sección >

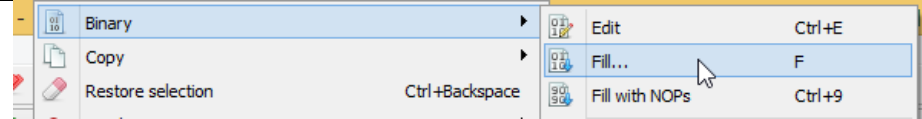
El único principio que tenemos base es que podemos definir label , comentarios sin mayor problema (leer el faq del Multiline Ultimate Assembler)

Ahora al ir a la sección nueva, nos llevamos esta sorpresa que está escrito

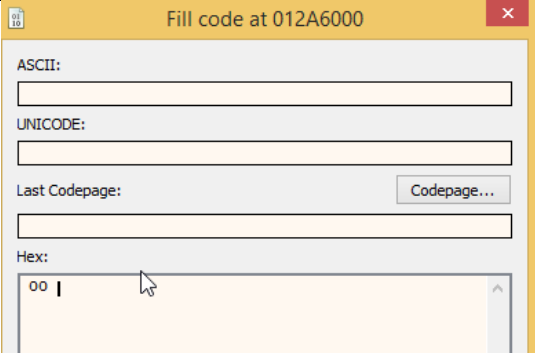


<img41: Viendo la nueva sección >

Está escrito, jaja, bueno no es mala noticia, esto es el certificado digital (recuerden que luego del ejecutable , es ingresado el certificado digital sin asignar ese espacio, una vez que creamos una nueva sección con un espacio este está incluido ahí, lo hacemos fill con 00 (hex 00), solo eliminamos que dijera que la sección no está, pero su contenido aun está ahí.



<img42:comandos en x32dbg >

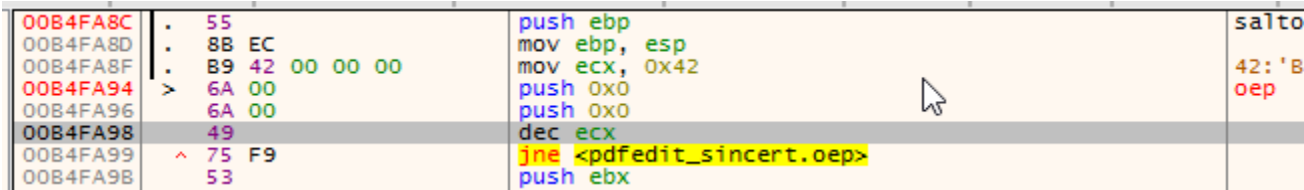


<img43:comandos en x32dbg >

USANDO MULTILINE ULTIMATE ASSEMBLER PARA EJERCITAR

Luego comenzamos a crear el código de inline, esta vez desde el ejecutable original solo que hemos quitado el certificado.

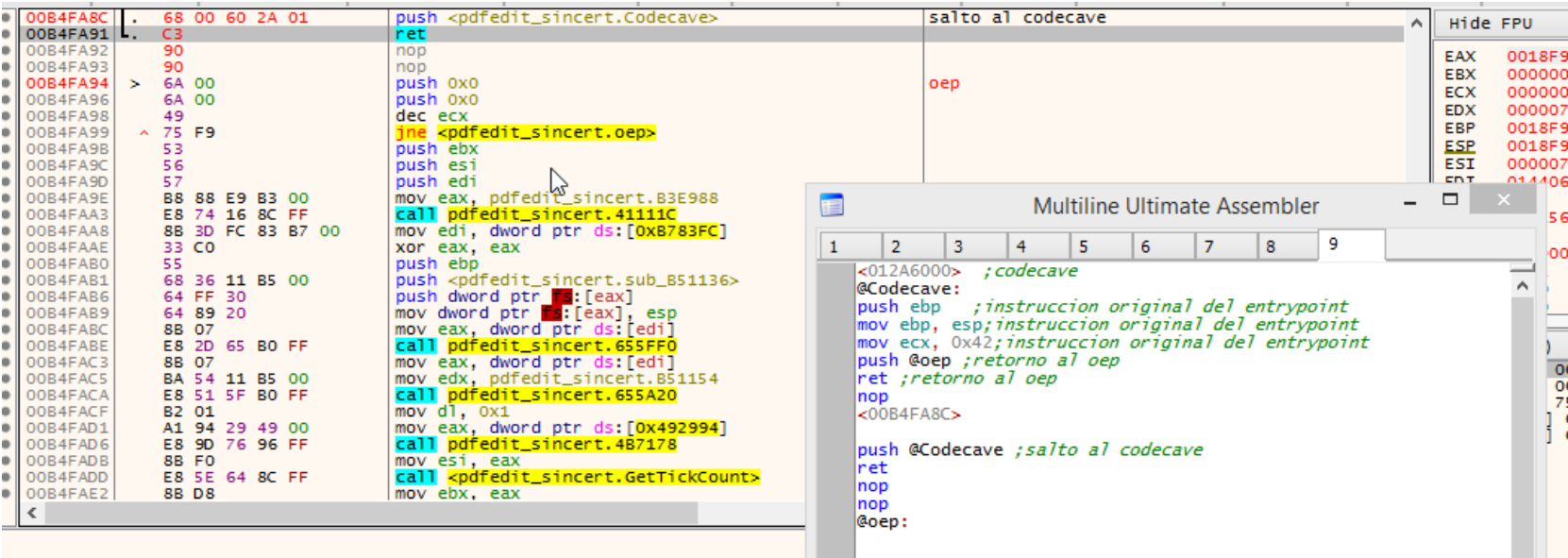
Antes asi es el entryptoint, debemos ensayar antes de cambiar algo



<img44:Entrypoint antes del cambio en x32dbg >

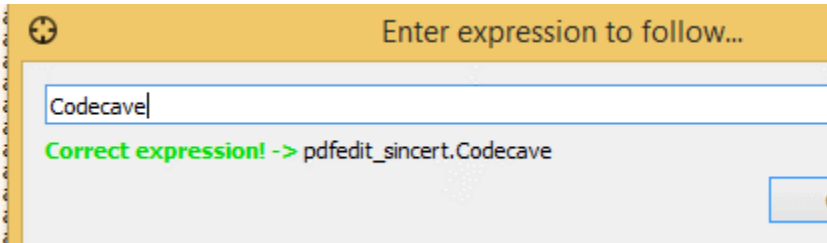
Después de una prueba de inline.





<img45:Entrypoint despues del cambio en x32dbg >

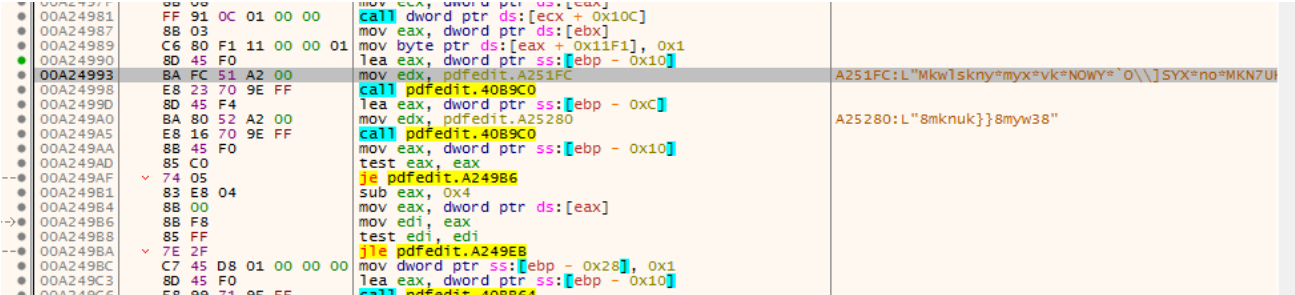
Lo especial y lo que me gusta cuando usas label, es que puedes ir fácilmente con x32dbg al nuevo label creado



<img46:comandos en x32dbg permiten ir a la dirección del label>

Hemos hecho una pausa para practicar como podemos robar unas pequeñas instrucciones, hasta aquí lo defino como Codecave , Esto es para probar solamente que podemos raptarnos código de donde queramos, el principio será el mismo en todos los lados que robemos instrucción original y me permitirá extender la funcionalidad , ósea si robamos 10 instrucciones, esas instrucciones, deberán estar en el lugar del inline, pero además debemos mantener balanceado nuestros valores para esp por lo que podemos hacer uso de pushad o bien push (registro), aquí definimos labels para hacer los llamados al igual como hacíamos con el messagebox, aquí usamos una etiqueta creada por nosotros , un label es una etiqueta que se realiza en la dirección definido en MLA. (multiline ultimate assembler.) , es similar al principio de comentario, pero el label es definido automáticamente o por el usuario como un arroba @, así como una api es llamada por su nombre ,este label es llamado solo si lo necesitamos usar (en el depurador es llamado sin el arroba).

La acción de parcharlo de forma ordenada desde el oep, requiere tiempo, comenzamos con la primera idea (no es la que usaremos al final, pero es para demostrar cómo se hace. Veamos aquí necesitamos hacer fill de mov edx, valor a mov edx,0)

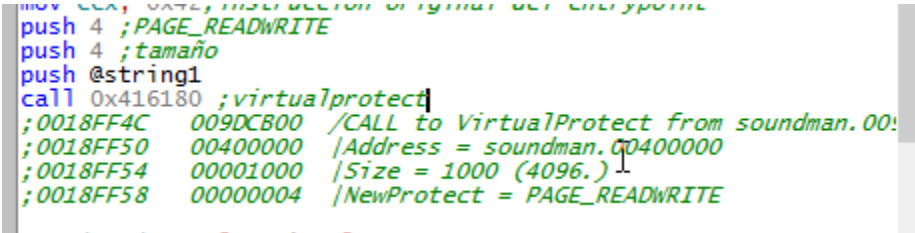


<img47 watermark encryptado>

El primer byte en el caso de mov (es el BA) (mov edx), los siguientes bytes son de la string, así que las direcciones reales a parchar serian A24994 y 0A249a1 con 0 , Aquí comenzaremos directamente con multi line Multiline Ultimate Assembler en x32dbg, Comenzamos a definir las como string1, string2

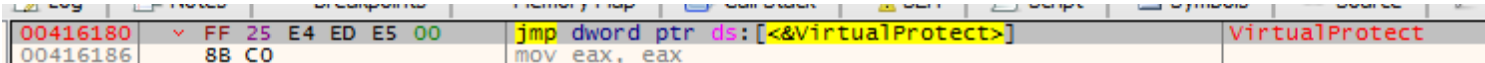
Definimos label llamadas String1 y String2 <pre>&lt;00A24994&gt; @string1: &lt;00A249A1&gt; @string2:</pre>	Y ahora el parche será <pre>mov dword ptr [string1], 0 mov dword ptr [string2], 0 push @oep ;retorno al oep</pre> <img49 usando Multiline Ultimate Assembler>	Veamos como estamos al minuto : no puede ensamblar porque no hay permisos : <pre>First chance exception on 012A601B (C0000005, EXCEPTION_ACCESS_VIOLATION)!</pre> <img50 usando Multiline Ultimate Assembler>
--	---	---

Entonces invocaré a virtualprotect, con el orden determinado, dirección, tamaño , protección, se ensamblan en inverso, para tener permisos en la sección



<img51 invocando a Virtualprotect en Multiline Ultimate Assembler>

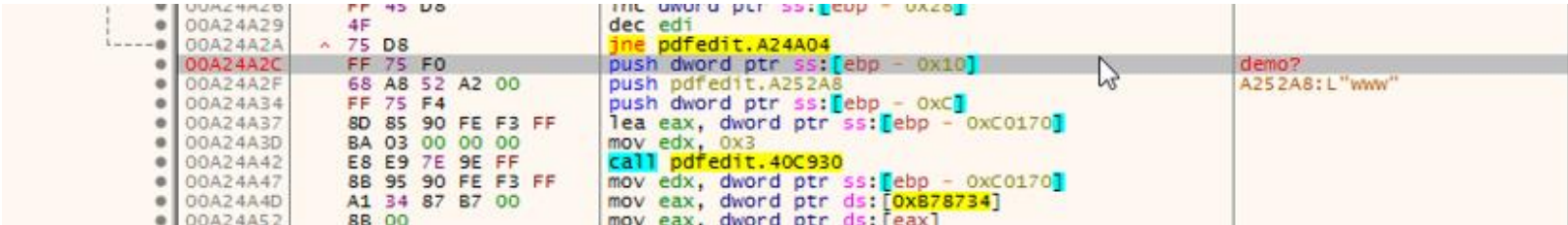
En la aplicación disponemos de virtualprotect



<img52 explicación de porque call 416180 >

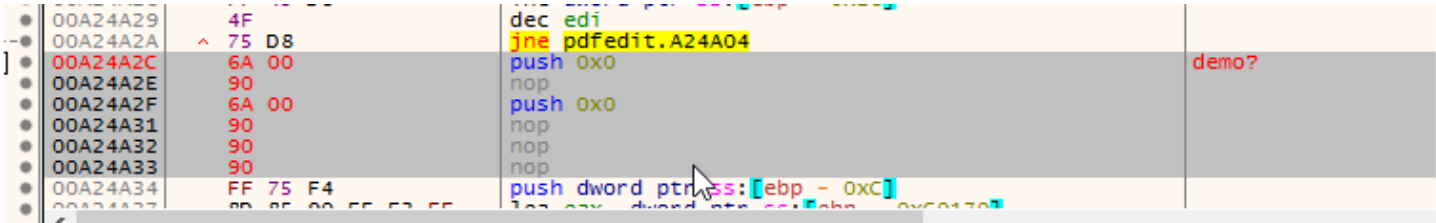


Luego identifico el objetivo a cambiar



<img53 watermark>

Para que quede así:



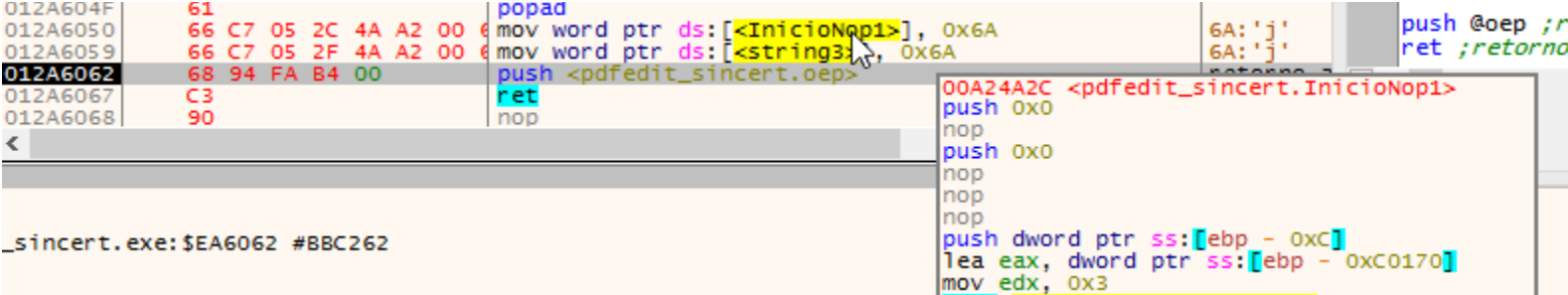
<img54 modificado (sin watermark)>

Vamos a colocar desde A24A2c hasta A24A33 NOP (la zona de finalizar el nop seria A24A34 y el el dword esperado 6A00 en las 2 posiciones que quiero

Para el bucle del nop coloco

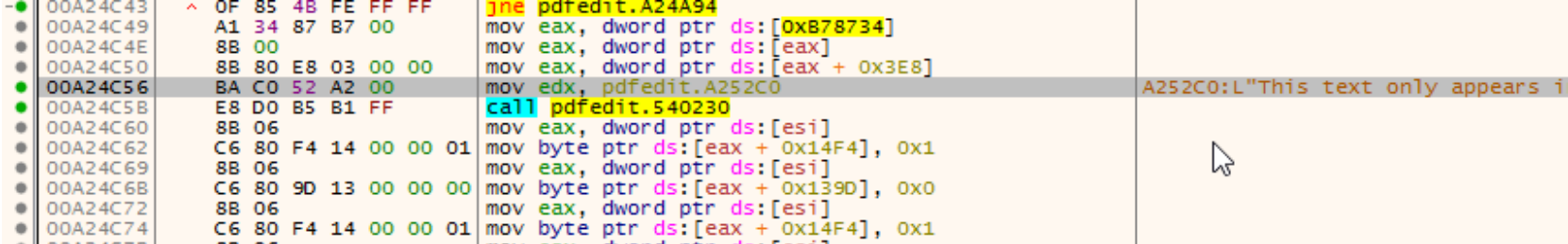
```
pushad
xor eax,eax
mov eax,@InicioNop1
@bucle1:
mov byte ptr [eax], 0x90
inc eax
cmp eax,@FinNop1
jne @bucle1
popad
```

Luego para parchar los bytes 6A00 debo moverlos en sentido inverso al push 0 , osea 0x006A



<img55 probando capacidad de Multiline Ultimate Assembler>

Seguimos



<img56 watermark>

Esta zona requiere también que sea mov edx,0

Repetimos la forma anterior (darle permisos, luego mover a 0)

Asi que al minuto vamos así

```
<00A24994>
@string1:
<00A249A1>
@string2:
<00A24a2F>
@string3:
<00A24c57>
@string4:
<009C37A6> ;je a jmp
jmp 0x9c37e0
<00A24a2C>
@InicioNop1;@string3
<00A24a34>
@FinNop1:
<012A6000> ;codecave
@Codecave:
```



```
<00A24993>
mov edx,0 ;string1
<00A249A0>
mov edx,0 ;string2
<00A24a2F>
push 0 ; ;string3
push 0 ; ;string4
nop
nop
nop
nop
<00A24c56>
mov edx,0 ; ;string5
<009C37A6> ;je a jmp
jmp 0x9c37e0 ;demo1
;ahora el lugar para definir el espacio de los nuevos parches
<012A6000> ;codecave
@Codecave:
```

Hasta aquí diríamos que estamos relativamente listo para la parte final y comienzo a ver si realmente logro extender la funcionalidad

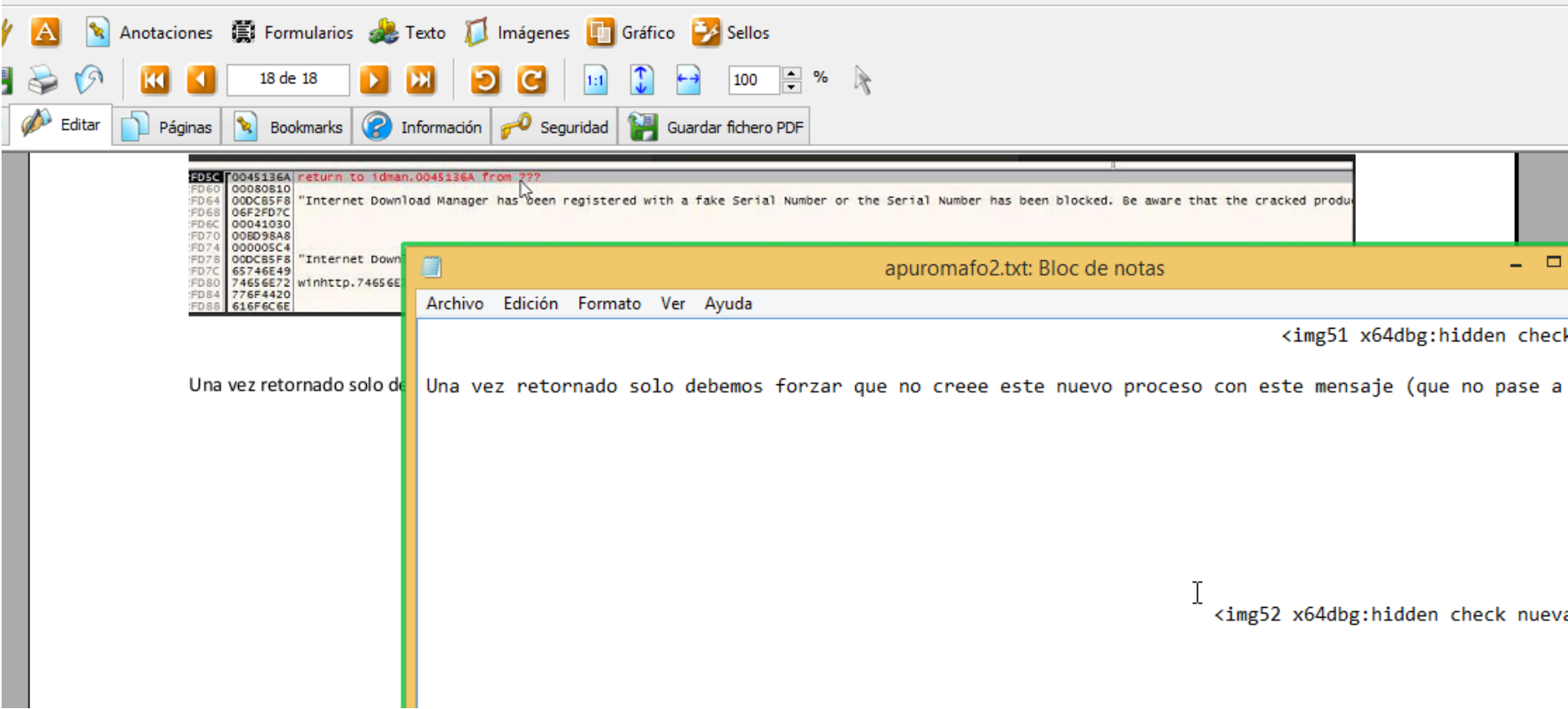
```
<00A74D0C> //dirección donde esta la comparación del texto

    inc dword ptr [ebp - 0xa4] //aquí le dice que termino la pagina 1
    pushad //pushad guarda registros
mov eax, dword ptr [ebp - 0x1c] // instrucción 1
    mov eax, dword ptr [eax + 0x122c] // instrucción 2
    mov eax, dword ptr [eax + 0x10] // instrucción 3
    cmp dword ptr [ebp - 0xa4], eax //comparación que realmente necesito
    je @L00000001 //si son iguales devuelve el flujo de lo que continua
    popad //devuelve el valor antes del injerto

@L00000008:
    cmp dword ptr [ebp - 0xa4],99999 //comparación con 99999 hex paginas
    jne 0xa73a02 //instrucción original que comparaba pagina actual versus numero gigante (antes decía 2)

@L00000001:
    mov eax, dword ptr [ebp - 0x1c] //instrucción que continua el flujo
```

Hasta aquí estaría feliz Pero aun hay un error, si el valor a comparar con 99999 hay un pushad, cuando salte no hara el popad, por lo que debemos forzar devolver el flujo luego del popad y luego recién saltar al flujo normal Con eso reparado nuestro injerto funciona, pero pasa algo, aun asi no tenemos todas las paginas(porque si tenemos que comparar debe ser con 1 mas xD)



<img58 verificando cuantas paginas llevaba convertida >

(estoy probando sobre el pdf que escribi sobre idm tiene 18 páginas ) Le falta solo 1 pagina, probemos 2 casos (antes de la comparación, el valor de eax, tiene el total de páginas)

Agregando dec eax, antes de la comparación: Me convierte 2 paginas menos	Agregando inc eax,antes de la comparación Me convierte con éxito las paginas antes buscado
---	---

Con esto estaría vencida la limitación de txt

[illegible]



```

    nop
    nop
    nop
@sigue_txt:

```

Luego me propongo ir a la versión de exportar en excel (es muy similar a la de txt), ahora bien luego me propongo ir a la versión de exportar imágenes y aquí si lo pensamos de forma crítica, no nos advierte ningún mensaje, por lo que deberemos buscar como eliminamos el watermark en el pasado

aquí por ejemplo llamaba a 40b9c0



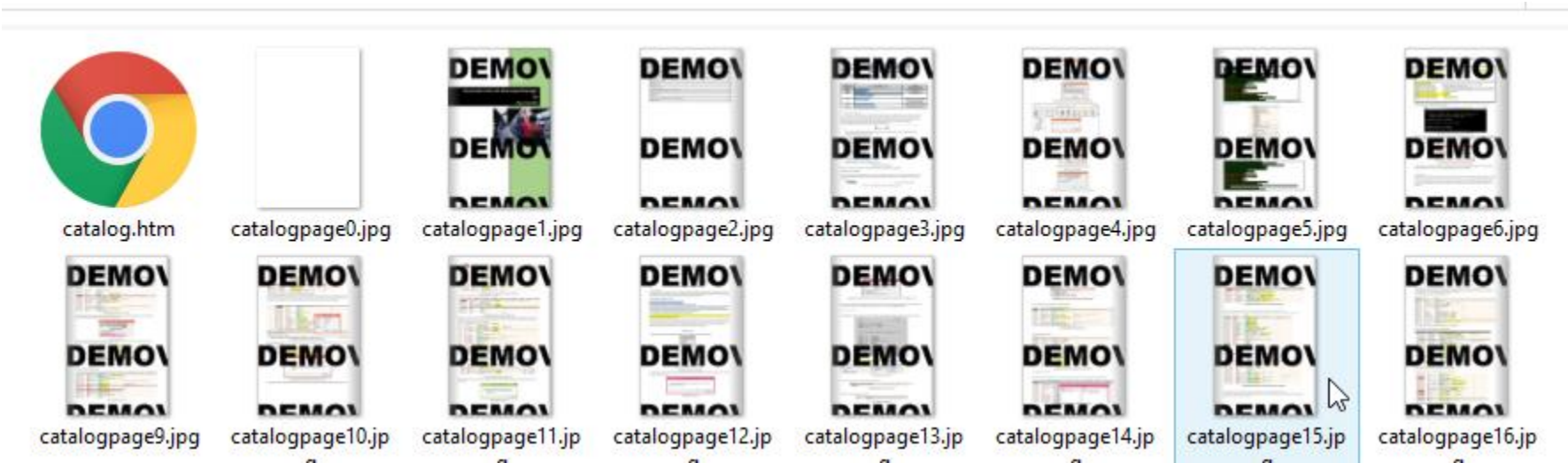
<img 59 usando el watermark del pasado para pillar el call encriptador/desencriptador >

Probamos exportar como imagen y llegamos al lugar en cuestión 9BD3F4 aquí tenemos la palabra en otro orden

DE	MO	VE	RS	IO	N
9DB5AC	9DB5D4	9DB5FC	9DB5C0	9DB5AC	9DB610
0098D3F4	BA AC D5 98 00		mov edx, apuromafo_tute.98D5AC	98D5AC:L"DE"	
0098D3F9	E8 C2 E5 A4 FF		call apuromafo_tute.40B9C0		
0098D3FE	8B C7		mov eax, edi		
0098D400	BA C0 D5 98 00		mov edx, apuromafo_tute.98D5C0	98D5C0:L"RS"	
0098D405	E8 B6 E5 A4 FF		call apuromafo_tute.40B9C0		
0098D40A	8B C6		mov eax, esi		
0098D40C	BA D4 D5 98 00		mov edx, apuromafo_tute.98D5D4	98D5D4:L"MO"	
0098D411	E8 3A F4 A4 FF		call apuromafo_tute.40C850		
0098D416	8B C7		mov eax, edi		
0098D418	BA E8 D5 98 00		mov edx, apuromafo_tute.98D5E8	98D5E8:L"IO"	
0098D41D	E8 2E F4 A4 FF		call apuromafo_tute.40C850		
0098D422	8B C6		mov eax, esi		
0098D424	BA FC D5 98 00		mov edx, apuromafo_tute.98D5FC	98D5FC:L"VE"	
0098D429	E8 22 F4 A4 FF		call apuromafo_tute.40C850		
0098D42E	8B C7		mov eax, edi		
0098D430	BA 10 D6 98 00		mov edx, apuromafo_tute.98D610		
0098D435	E8 16 F4 A4 FF		call apuromafo_tute.40C850		

<img 60 nuevo watermark, bastante difícil de encontrar >

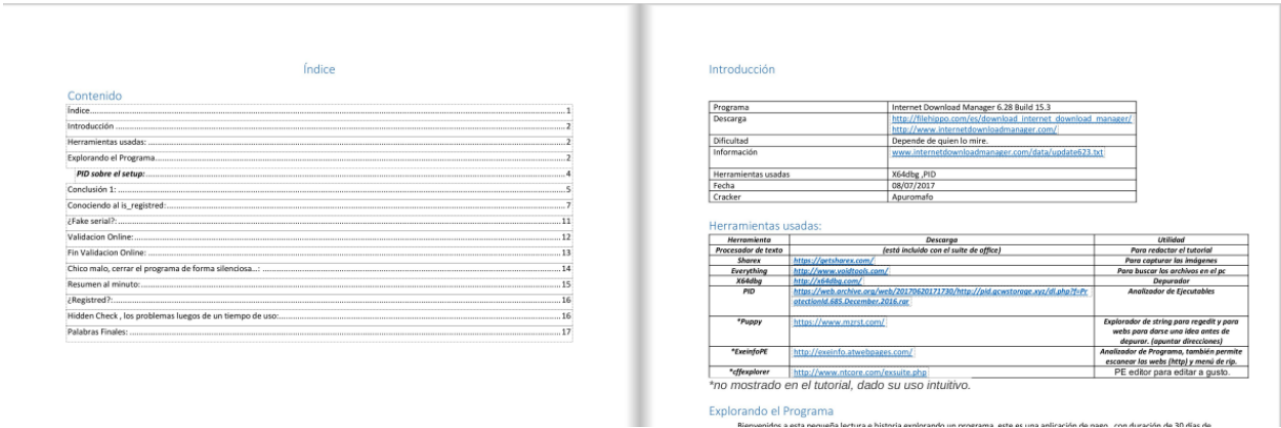
Antes de modificar



<img 61 como se ve el demo >

Después de la modificación no hay rastro de demo version

Vemos que al anular esta vez edx con 0 mata 2 limitaciones (exportar imágenes) y el que crea los flip pages (usa el mismo lugar para marcar el watermark)



<img 61 nuevo watermark anulado >

Probamos el que refiere imprimir , refiere que creara un watermark (pero al parecer también fue vencido con el primera), no hay watermark



<img 61 nuevo watermark, en pdf también anula >

Sigo probando el menú (imprimir pdf)

Mandamos a imprimir y existe el mismo watermark de imágenes si pasa a ser llamado por el call que teníamos alerta, asi que colocamos la misma estrategia y pillamos de nuevo nuestras frases demo versión en orden desordenado

0098D0B5	BA 1E 00 00 00	mov ecx, 0x1E	
0098D0C0	E8 6F 45 B5 FF	call apuromafo_tute2.511634	
0098D0C5	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D0C8	05 44 F8 FF FF	add eax, 0xFFFFF844	
0098D0CD	BA F0 D2 9B 00	mov edx, apuromafo_tute2.98D2F0	98D2F0:L"DE"
0098D0D2	E8 E9 E8 A4 FF	call apuromafo_tute2.40B9C0	
0098D0D7	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D0DA	05 E8 F9 FF FF	add eax, 0xFFFFF9E8	98D304:L"RS"
0098D0DF	BA 04 D3 9B 00	mov edx, apuromafo_tute2.98D304	
0098D0E4	E8 D7 E8 A4 FF	call apuromafo_tute2.40B9C0	
0098D0E9	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D0EC	05 44 F8 FF FF	add eax, 0xFFFFF844	98D318:L"MO"
0098D0F1	BA 18 D3 9B 00	mov edx, apuromafo_tute2.98D318	
0098D0F6	E8 55 F7 A4 FF	call apuromafo_tute2.40C850	
0098D0F8	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D0FE	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D101	05 E8 F9 FF FF	add eax, 0xFFFFF9E8	98D32C:L"IO"
0098D106	BA 2C D3 9B 00	mov edx, apuromafo_tute2.98D32C	
0098D108	E8 40 F7 A4 FF	call apuromafo_tute2.40C850	
0098D110	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D113	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D116	05 44 F8 FF FF	add eax, 0xFFFFF844	98D340:L"VE"
0098D118	BA 40 D3 9B 00	mov edx, apuromafo_tute2.98D340	
0098D120	E8 28 F7 A4 FF	call apuromafo_tute2.40C850	
0098D125	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D128	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	
0098D12B	05 E8 F9 FF FF	add eax, 0xFFFFF9E8	
0098D130	BA 54 D3 9B 00	mov edx, apuromafo_tute2.98D354	
0098D135	E8 16 F7 A4 FF	call apuromafo_tute2.40C850	
0098D13A	8B 45 08	mov eax, dword ptr ss:[ebp + 0x8]	

<img 62 nuevo watermark, bastante difícil de encontrar v2 >

Antes de modificar en los pdf se aprecia la diferencia denuevo

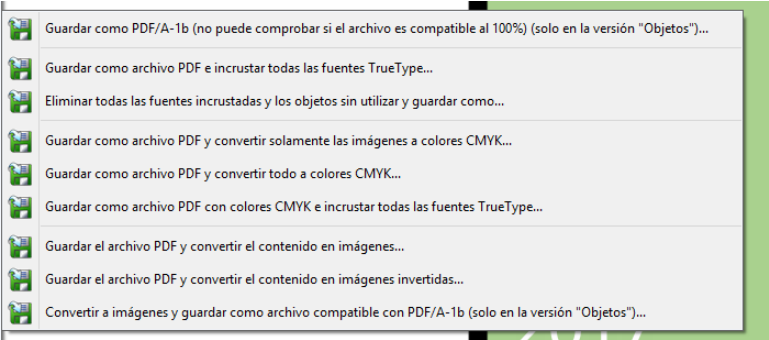
Antes	Después de modificar (todos los edx a cero)
<div></div> <div>&lt;img 63 antes &gt;</div>	<div></div> <div>&lt;img 64 despues &gt;</div>

Medio cansado de explorar el programa, ya es funcional en parte, hacemos resumen

Desde el original que hemos quitado el certificado, se ha modificado el titulo, se ha quitado el watermark para pdf en varias opciones, se ha extendido la función para txt, para Excel, para Word, Ahora falta la ultima parte estética que es seguir probando que sirve y que no, quedan toda la parte estética en manos de resource hacker.

Adicionalmente quedan algunas opciones disponibles solo en objetos (otra versión he de asumir)

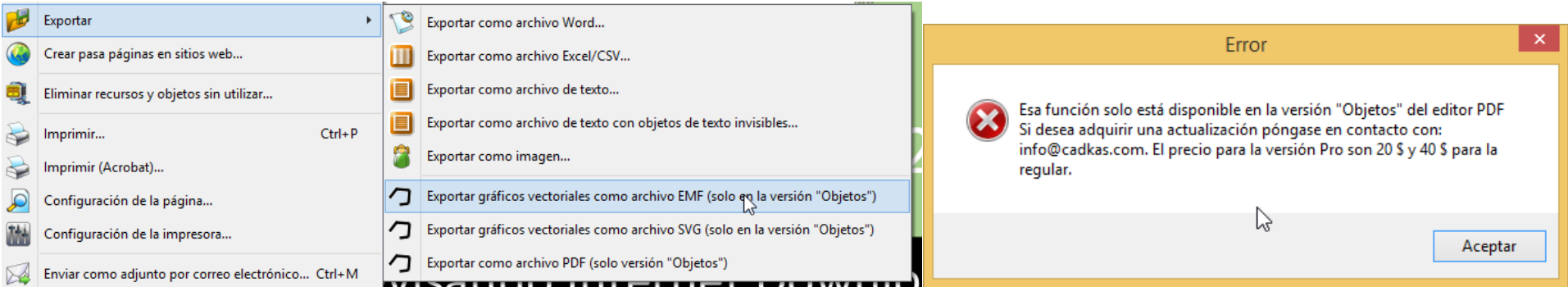
Por el lado estético guardar como (la primera y ultima funcionan) no es solo versión objetos..



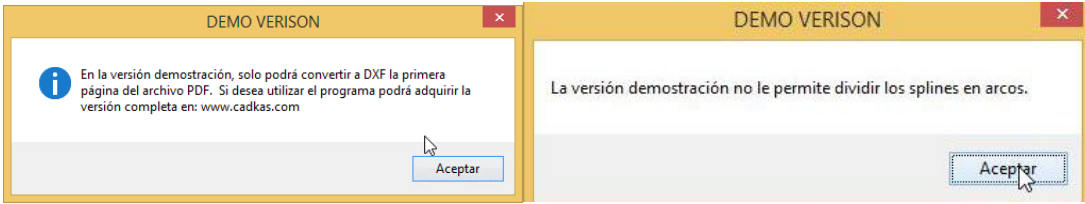
<img 63 Vision del menú que es funcional >

LADO DEMO

Por el lado de solo objetos esta capado, aunque por dentro si vemos que se puede ver que si existen las extensión svg y emf, pero se deben eliminar del menú como tal porque los deja muy mal hechos.



<img 64/65 menus no funcionales >

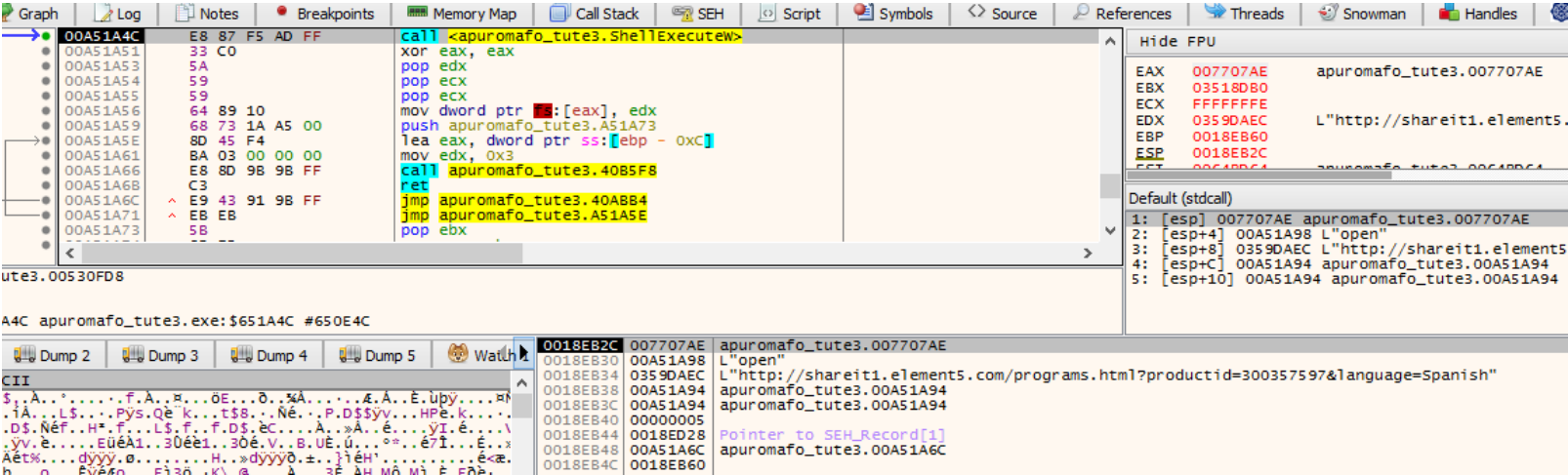


<img 66/67 menus no funcionales >

por otro lado en su web existen conversores triales para DXF(modos gráficos para software tipo autocad) <http://www.cadkas.com/ehome.php>

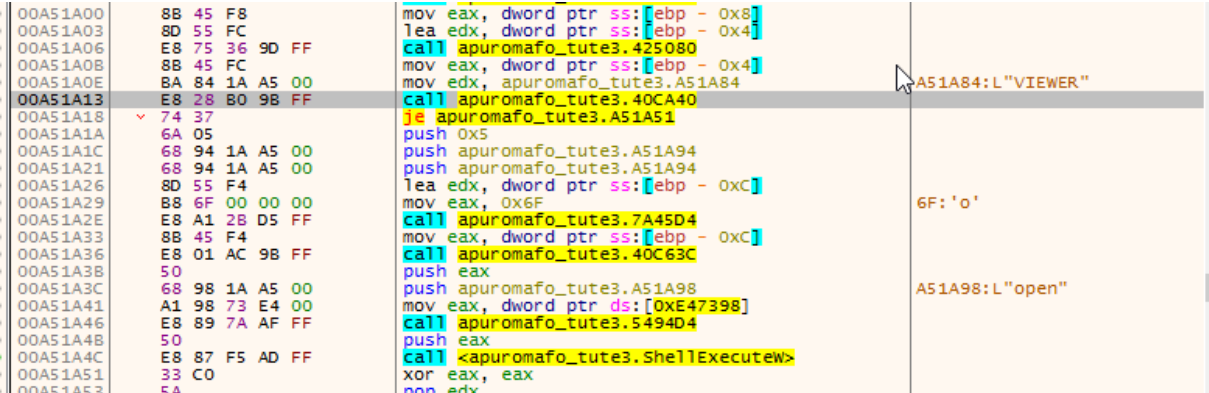
ULTIMA LIMITACIÓN, EJECUCIÓN DE WEB SIN NUESTRO CONSENTIMIENTO.

Ahora sigamos cuando cerramos la aplicación va a una pagina con BP en el call a ShellExecuteW



<img 68 ShellExecuteW para la web del programa>

Proviene de aquí:



<img 69 rutina del ShellExecuteW para la web del programa>

Para anular es fácil, o se fuerza el salto de je a jmp o se anula el call de A51A4c a otra instrucción ejemplo NOP/jmp A51A51 esto se parece mucho al comienzo cuando colocaba en el titulo demo.

El remate final para los valientes. **Aún con todo parchado en sus limitaciones aún quedan todos los nags** una idea dada por Jhon (que me ayudó a revisar la ortografía ) , sería que cada vez que llame a demo, se anulara el string , pero bueno, son temas de cada quien pueda y para rematar el descubrimiento si bajamos la versión actual al dia de hoy 8/8/17 vemos que ha incrementado un poco el ejecutable(setup), así que ha cambiado el programa, por lo que mis direcciones no deberían ser iguales a las suyas, pero los principios seguirían siendo el mismo. Y además existe otra versión pdfedit!.exe osea quien sabe cuantos programas demo crean estos autores.

spdfedit!.exe	08-08-2017 17:11	Aplicación	7.899 KB
spdfedit-.exe	22-07-2017 22:01	Aplicación	6.559 KB

<img 70 Programa ha sido modificado en algunas semanas>



Palabras Finales:

Tengo un programa demo funcional sin watermark, para quien obviamente lo necesite, ahora bien si alguien puede dar soporte es ideal que compren los que pueden realmente, aquí queda plasmada una experiencia en un programa demo (capado) que si es posible quitarle el watermark y extender muy poquito su funcionalidad, agradecer a AbelJM por sugerir el programa, Jhon por revisar conmigo la redacción, y a ti por leer una porción de mis recuerdos.

***Si te complica la ortografía, corrígela con este mismo software, que permite editar pdf ☺***

Dedicado a los lectores que suelen practicar y/o aprender reversing o simplemente una lectura amena, está más que decir que si te ha gustado el software y si tienes la posibilidad de comprarlo no dejes de apoyar al soporte del programa.

saludos   Apuromafo



Apuromafo TSKh