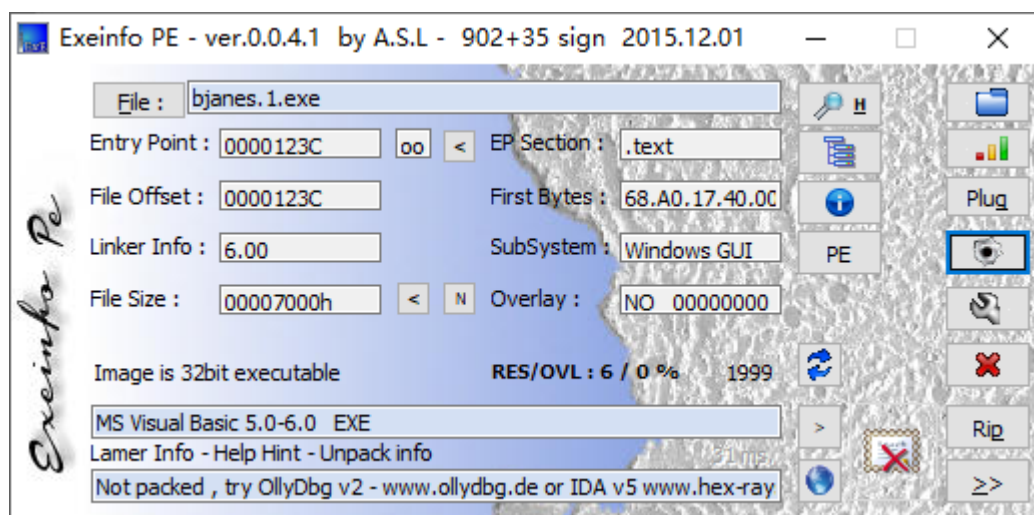


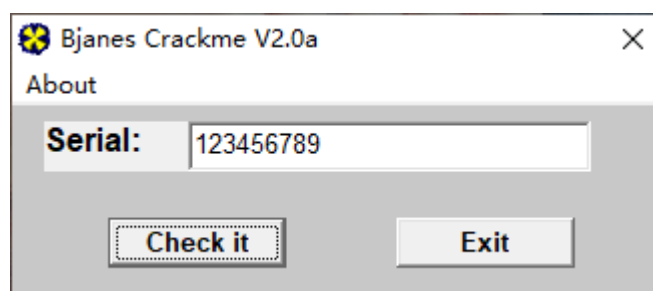
查壳
分析程序
验证结果

查壳



还是个VB写的程序，没有壳

分析程序



既然是单纯的一个序列号的保护方式 就没有必要分析算法了，直接追踪序列号就行

8945 90	mov dword ptr ss:[ebp-0x70],eax		
8945 A0	mov dword ptr ss:[ebp-0x60],eax		
C785 58FFFFFF	mov dword ptr ss:[ebp-0xA8],bjanes_1.004022F0	Wrong serial!	
89BD 50FFFFFF	mov dword ptr ss:[ebp-0xB0],edi		
FFD6	call esi	msvbvm60.__vbaStrMov	
8D95 60FFFFFF	lea edx,dword ptr ss:[ebp-0xA0]		
8D4D C0	lea ecx,dword ptr ss:[ebp-0x40]		
C785 68FFFFFF	mov dword ptr ss:[ebp-0x98],bjanes_1.004022C8	Sorry, try again!	
89BD 60FFFFFF	mov dword ptr ss:[ebp-0xA0],edi		
FFD6	call esi	msvbvm60.__vbaStrMov	
8D45 90	lea eax,dword ptr ss:[ebp-0x70]		
8D4D A0	lea ecx,dword ptr ss:[ebp-0x60]		
50	push eax		
8D55 B0	lea edx,dword ptr ss:[ebp-0x50]		
51	push ecx		
_1.004022C8 (UNICODE "Sorry, try again!")			
68j=00000051			

首先，根据错误的字符串提示向上找跳转

地址	HEX	数据	反汇编	注释
00403A18	.	8945 E8	mov dword ptr ss:[ebp-0x18],eax	
00403A1B	.	33DB	xor ebx,ebx	msvbvm60
00403A1D	^	E9 5AFDFFFF	jmp bjanes_1.0040377C	开始新
00403A22	>	33DB	xor ebx,ebx	msvbvm60
00403A24	>	8B35 A4104000	mov esi,dword ptr ds:[<&MSVBVM60.__vbaVarDup>]	msvbvm60
00403A2A	.	B9 04000280	mov ecx,0x80020004	
00403A2F	.	894D 98	mov dword ptr ss:[ebp-0x68],ecx	
00403A32	.	B8 0A000000	mov eax,0xA	
00403A37	.	894D A8	mov dword ptr ss:[ebp-0x58],ecx	
00403A3A	.	BF 08000000	mov edi,0x8	
00403A3F	.	8D95 50FFFFFF	lea edx,dword ptr ss:[ebp-0xB0]	
00403A45	.	8D4D B0	lea ecx,dword ptr ss:[ebp-0x50]	
00403A48	.	8945 90	mov dword ptr ss:[ebp-0x70],eax	
00403A4B	.	8945 A0	mov dword ptr ss:[ebp-0x60],eax	
00403A4E	.	C785 58FFFFFF	mov dword ptr ss:[ebp-0xA8],bjanes_1.004022F0	Wrong :
00403A58	.	89BD 50FFFFFF	mov dword ptr ss:[ebp-0xB0],edi	

这两句一共有三个跳转，也就是说对序列号进行了三次判断

ds:[004010A4]=66106DF6 (msvbvm60.__vbaVarDup)
esi=660E6C30 (msvbvm60.__vbaStrMove)
跳转来自 00403704

首先来到403704这个位置的跳转

地址	HEX	数据	反汇编	注释
004036E3	.	33C9	xor ecx,ecx	
004036E5	.	83F8 09	cmp eax,0x9	比较长度是否为9
004036E8	.	0f95c1	setne cl	
004036EB	.	F7D9	neg ecx	
004036ED	.	8BF1	mov esi,ecx	
004036EF	.	8D4D E4	lea ecx,dword ptr ss:[ebp-0x1C]	
004036F2	.	FF15 C0104000	call dword ptr ds:[<&MSVBVM60.__vbaFreeStr>]	msvbvm60.__vbaFreeStr
004036F8	.	8D4D D4	lea ecx,dword ptr ss:[ebp-0x2C]	
004036FB	.	FF15 C4104000	call dword ptr ds:[<&MSVBVM60.__vbaFreeObj>]	msvbvm60.__vbaFreeObj
00403701	.	66:3BF3	cmp si,bx	
00403704	..	0F85 1A030000	jnz bjanes_1.00403A24	
0040370A	.	8B17	mov edx,dword ptr ds:[edi]	
0040370C	.	57	push edi	
0040370D	.	FF92 08030000	call dword ptr ds:[edx+0x308]	
00403713	.	50	push eax	
00403714	.	8D45 D4	lea eax,dword ptr ss:[ebp-0x2C]	
00403717	.	50	push eax	

这里是比较字符串的长度是否为9，不是则报错

地址	HEX 数据	反汇编	注释		
00403A18	. 8945 E8	mov dword ptr ss:[ebp-0x18], eax			
00403A1B	. 33DB	xor ebx, ebx	msvbvm60.rtcStrFromVar		
00403A1D	. ^ E9 5AFDFFFF	jmp bjanex_1.0040377C	开始新一轮循环		
00403A22	> -33DB	xor ebx, ebx	msvbvm60.rtcStrFromVar		
00403A24	> 8B35 A4104000	mov esi, dword ptr ds:[<MSVBVM60.__vbaVarDup>]	msvbvm60.__vbaVarDup		
00403A2A	. B9 04000280	mov ecx, 0x80020004			
00403A2F	. 894D 98	mov dword ptr ss:[ebp-0x68], ecx			
00403A32	. B8 0A000000	mov eax, 0xA			
00403A37	. 894D A8	mov dword ptr ss:[ebp-0x58], ecx			
00403A3A	. BF 08000000	mov edi, 0x8			
00403A3F	. 8D95 50FFFFFF	lea edx, dword ptr ss:[ebp-0xB0]			
00403A45	. 8D4D B0	lea ecx, dword ptr ss:[ebp-0x50]			
00403A48	. 8945 90	mov dword ptr ss:[ebp-0x70], eax			
00403A4B	. 8945 A0	mov dword ptr ss:[ebp-0x60], eax			
00403A4E	. C785 58FFFFFF	mov dword ptr ss:[ebp-0xA8], bjanex_1.004022F0	Wrong serial!		
00403A58	. 89BD 50FFFFFF	mov dword ptr ss:[ebp-0xB0], edi			
00403A5E	. FFD6	call esi	msvbvm60.__vbaStrMove; <MSV		
ebx=660E07D5 (msvbvm60.rtcStrFromVar)					
跳转来自 004038AD, 00403809					
地址	HEX 数据	UNICODE	地址	数值	注释
004038AD	30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00000000000000000000000000000000	0019F280	08 00 00 00	00000000

接着再来到403A04这个位置的跳转

004039AD	. 8D8D 50FFFFFF	lea ecx, dword ptr ss:[ebp-0xB0]	
004039B1	. 8D55 80	lea edx, dword ptr ss:[ebp-0x80]	
004039B4	. 51	push ecx	
004039B5	. 52	push edx	
004039B6	. FF15 A0104000	call dword ptr ds:[<MSVBVM60.__vbaVarTstNe>]	var18 = 0019F230 var28 = 0019F280 __vbaVarTstNe
004039BC	. 8BF8	mov edi, eax	
004039BE	. 8D45 D8	lea eax, dword ptr ss:[ebp-0x28]	
004039C1	. 8D4D DC	lea ecx, dword ptr ss:[ebp-0x24]	
004039C4	. 50	push eax	

这里会根据vbaVarTstNe的比较结果来判断是否进行跳转，这个函数的VB常用的比较函数，可以肯定这里就是比较正确的序列号的地方了

等待程序断下后，来观察两个参数的内容，var18固定为0，var28是个字符串

ds:[004010A0]=6610980F (msvbvm60.__vbaVarTstNe)					
地址	HEX 数据	UNICODE	地址	HEX 数据	UNICODE
0019F280	08 00 00 00 00 00 00 00 84 2C 64 00	00000000000000000000000000000000	0019F280	08 00 00 00	00000000
0019F290	08 00 00 00 05 00 00 00 14 2D 64 00	00000000000000000000000000000000	0019F290	08 00 00 00	00000000
0019F2A0	02 00 00 00 00 00 00 00 03 00 68 BF	00000000000000000000000000000000	0019F2A0	02 00 00 00	00000000
0019F2B0	02 00 00 00 72 00 10 00 31 00 00 00	00000000000000000000000000000000	0019F2B0	02 00 00 00	00000000
0019F2C0	02 00 00 00 04 00 00 00 01 00 00 00	00000000000000000000000000000000	0019F2C0	02 00 00 00	00000000
0019F2D0	00 00 00 00 1C 3B 18 02 00 00 00 00	00000000000000000000000000000000	0019F2D0	00 00 00 00	00000000
0019F2E0	14 BF 60 00 74 BE 60 00 01 00 00 00	00000000000000000000000000000000	0019F2E0	14 BF 60 00	00000000
0019F2F0	16 11 40 00 FC F1 19 00 E0 10 40 00	00000000000000000000000000000000	0019F2F0	16 11 40 00	00000000
0019F300	0C F3 19 00 33 1D 05 66 18 32 5E 00	00000000000000000000000000000000	0019F300	0C F3 19 00	00000000
0019F310	4D 1F 40 00 5C 32 5E 00 18 1E 40 00	00000000000000000000000000000000	0019F310	4D 1F 40 00	00000000
0019F320	34 20 05 66 4D 1F 40 00 D8 F3 19 00	00000000000000000000000000000000	0019F320	34 20 05 66	00000000
0019F330	00 00 00 00 18 F4 19 00 1A 21 05 66	00000000000000000000000000000000	0019F330	00 00 00 00	00000000

首地址+8才是真正的内容

地址	HEX 数据	UNICODE	地
00642C84	33 00 00 00 00 00 00 00 00 00 00 00 00 54 CA 4D F4	3.... 쫓	00
00642C94	00 3F 00 8C 20 DA 65 00 50 E6 65 00 60 EA 65 00	璉誤 e	00
00642CA4	35 00 00 00 53 CA 4A F4 00 40 00 80 74 58 2B 76	5. 쫓 溫	00
00642CB4	00 00 00 00 00 00 00 00 00 00 00 00 80 5E CA 47 F4 耀	00
00642CC4	00 41 00 80 74 58 2B 76 00 00 00 00 00 00 00 00	找耀墻	00
00642CD4	00 00 00 00 5D CA 44 F4 00 42 00 88 A8 E6 5E 00	.. 쫓 箭	00
00642CE4	E0 2C 64 00 00 00 00 00 00 00 00 80 58 CA 41 F4	pd... 耀	00
00642CF4	00 43 00 80 04 00 00 00 34 00 39 00 00 00 00 00	網耀 49	00
00642D04	00 00 00 00 67 CA 5E F4 00 44 00 88 04 00 00 00	.. 쫓 騎	00
00642D14	20 00 33 00 00 00 00 00 00 00 00 00 62 CA 5B F4	3.... 裂	00
00642D24	00 45 00 80 74 58 2B 76 00 00 00 00 00 00 00 00	矮耀墻	00
00642D34	00 00 00 80 61 CA 58 F4 00 46 00 88 A8 E6 5E 00	. 耀 蟲	00

是字符3，理论上来说这里应该是个序列号，但是只出现一个字符，也就是说这个软件比较序列号的方式是逐个字符的比较，

再来看一下判断的部分

7	FF15 0C104000	call dword ptr ds:[&MSVBVM60.__vbaFreeVarList]	msvbvm60.__vbaFreeVarList
8	83C4 18	add esp,0x18	
9	66:85FF	test di,di	比较edi
10	75 1C	jnz short bjanex_1.00403A22	不相等报错
11	8B7D 08	mov edi,dword ptr ss:[ebp+0x8]	
12	B8 01000000	mov eax,0x1	i++
13	66:0345 E8	add ax,word ptr ss:[ebp-0x18]	
14	0F80 94010000	jb bjanex_1.00403BAC	
15	8945 E8	mov dword ptr ss:[ebp-0x18],eax	
16	33DB	xor ebx,ebx	msvbvm60.rtcStrFromVar
17	E9 5AFDFFFF	jmp bjanex_1.0040377C	开始新一轮循环
18	33DB	xor ebx,ebx	msvbvm60.rtcStrFromVar
19	8B35 A4104000	mov esi,dword ptr ds:[&MSVBVM60.__vbaVarDup]	msvbvm60.__vbaVarDup
20	B9 04000280	mov ecx,0x80020004	
21	894D 98	mov dword ptr ss:[ebp-0x68],ecx	

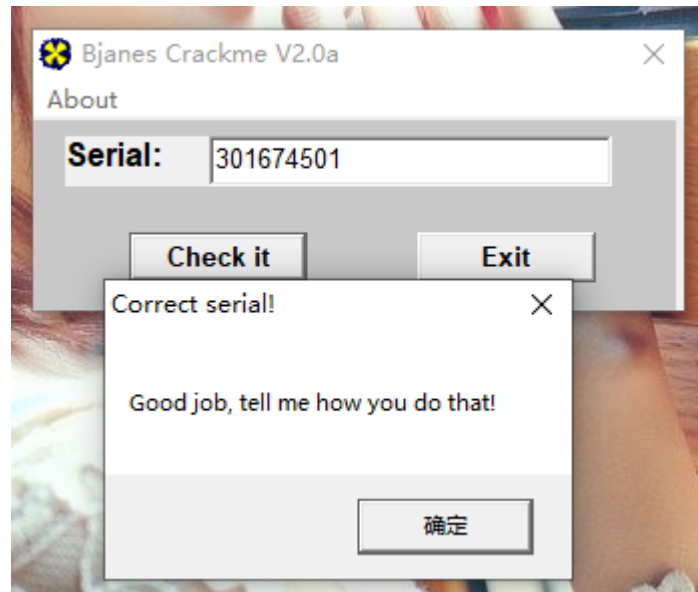
比较di之后，如果不相等则跳转到报错的地方，如果相等则继续往下走，后面两句相当于i++，然后跳转到循环开始处，

也就是说只要在vbaVarTstNe比较之后跳转之前修改ZF标志位，就能看到每个正确的序列号

正确的序列号是**301674501**

验证结果

输入得到的结果，提示正确



需要相关文件的可以到我的Github下载：<https://github.com/TonyChen56/160-Crackme>