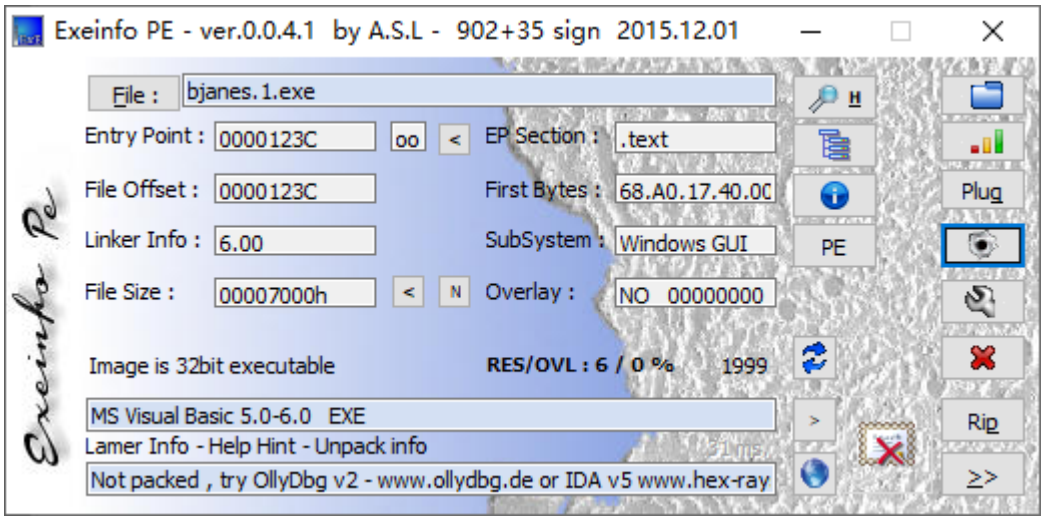
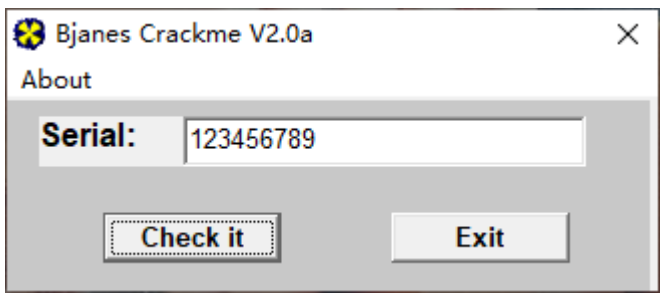
查壳



还是个VB写的程序，没有壳

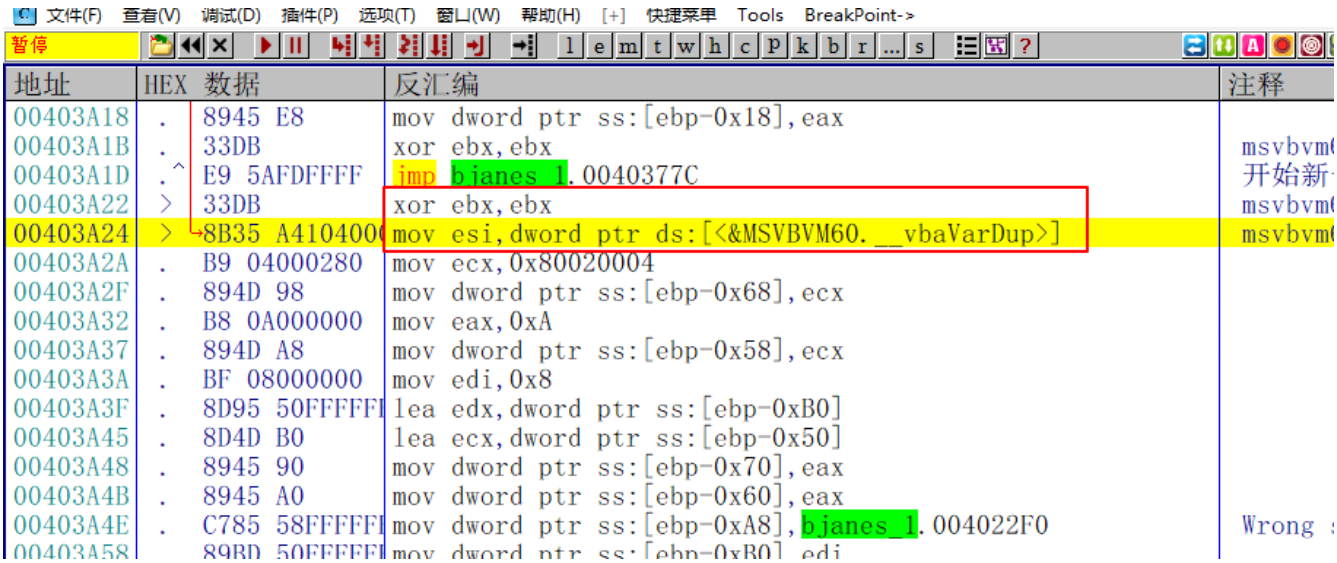
分析程序



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 58FFFFF mov dword ptr ss:[ebp-0xA8],bjanes_1.004022F0   Wrong serial!
89BD 50FFFFF mov dword ptr ss:[ebp-0xB0],edi
FFD6         call esi                                        msvbvm60.__vbaStrMov
8D95 60FFFFF lea edx,dword ptr ss:[ebp-0xA0]
8D4D C0      lea ecx,dword ptr ss:[ebp-0x40]
C785 68FFFFF mov dword ptr ss:[ebp-0x98],bjanes_1.004022C8   Sorry, try again!
89BD 60FFFFF 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!")
68]=00000051
```

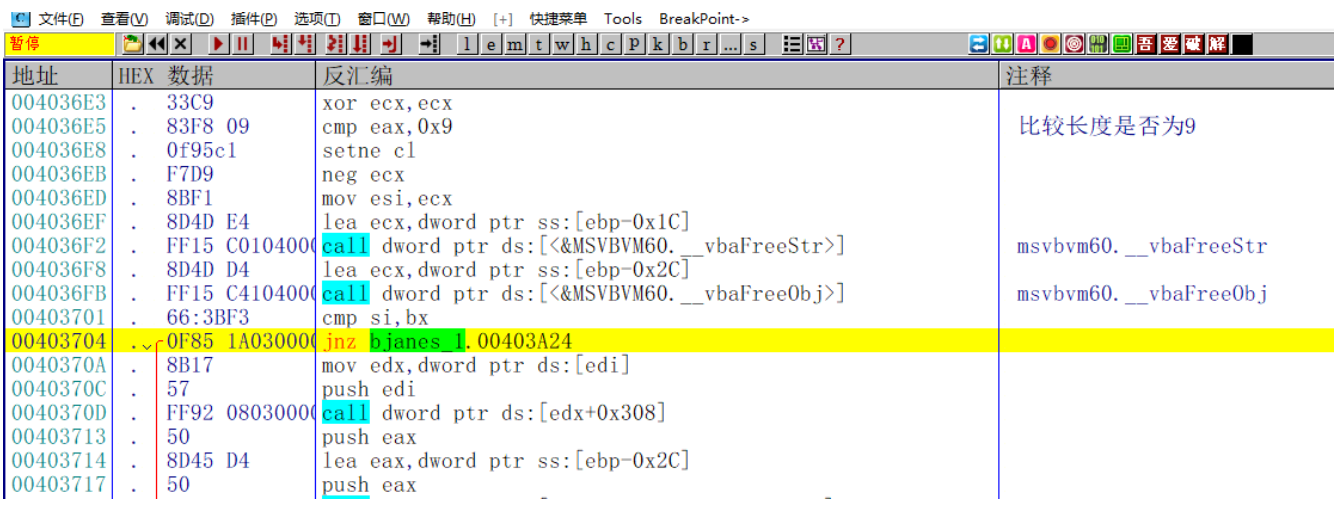
首先，根据错误的字符串提示向上找跳转



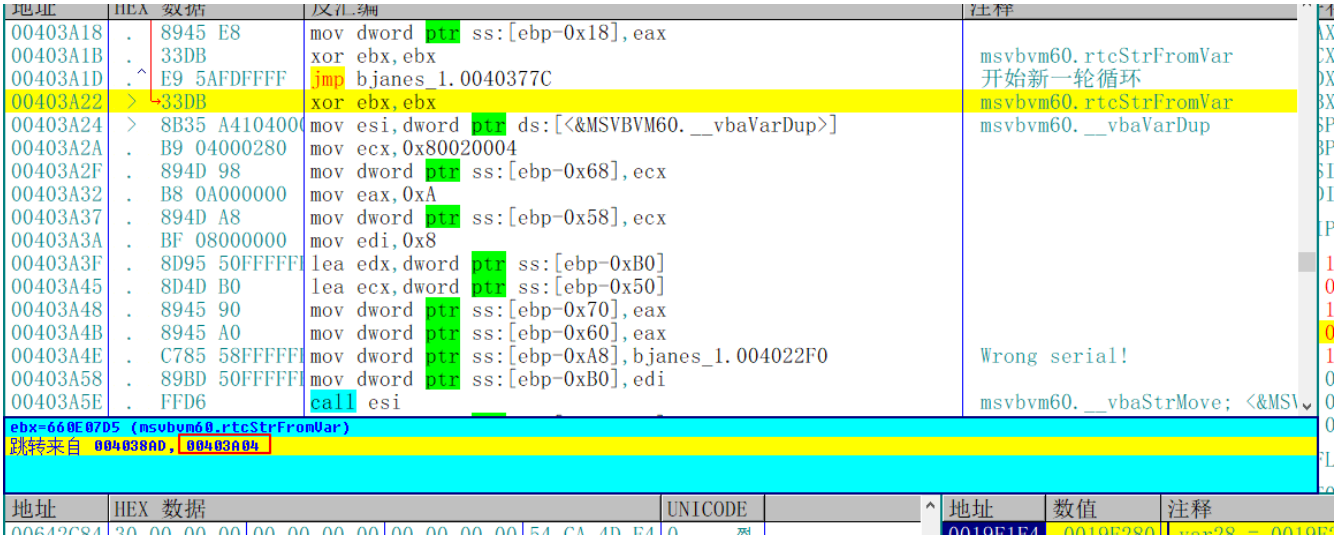
这两句一共有三个跳转，也就是说对序列号进行了三次判断

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

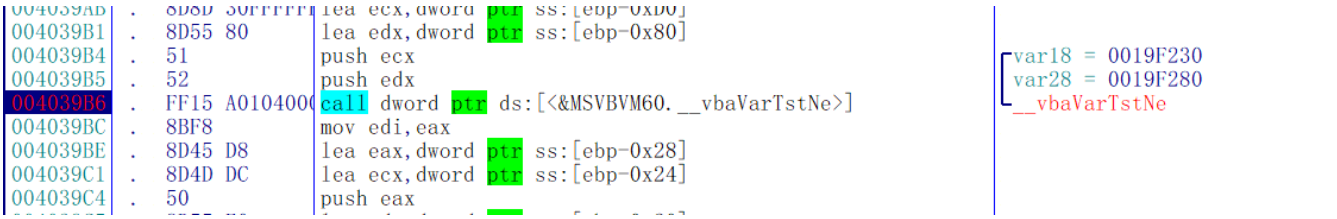
首先来到403704这个位置的跳转



这里是比较字符串的长度是否为9，不是则报错

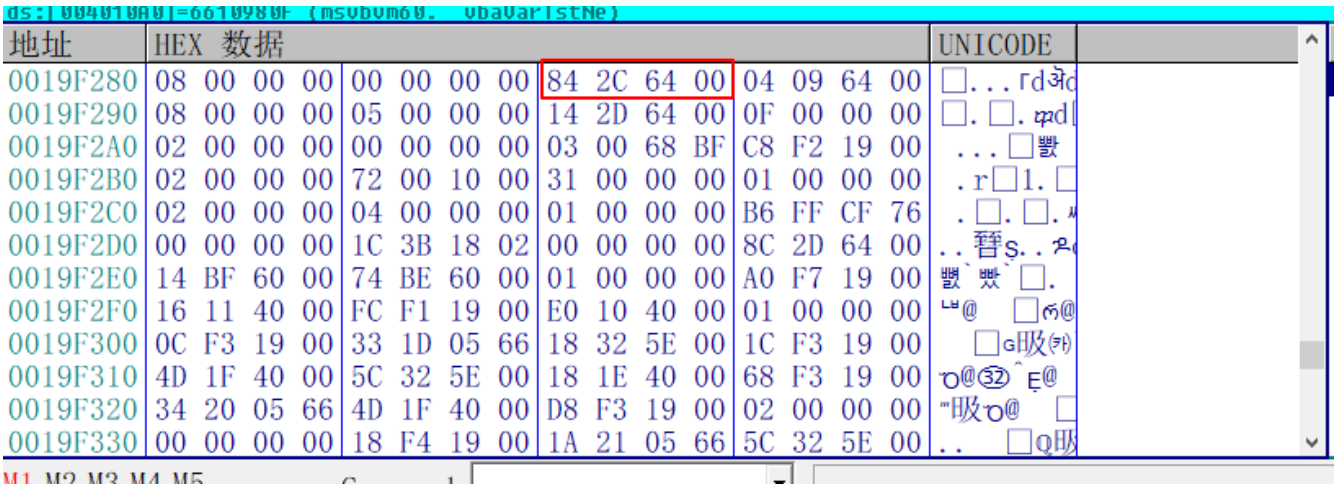


接着再来到403A04这个位置的跳转

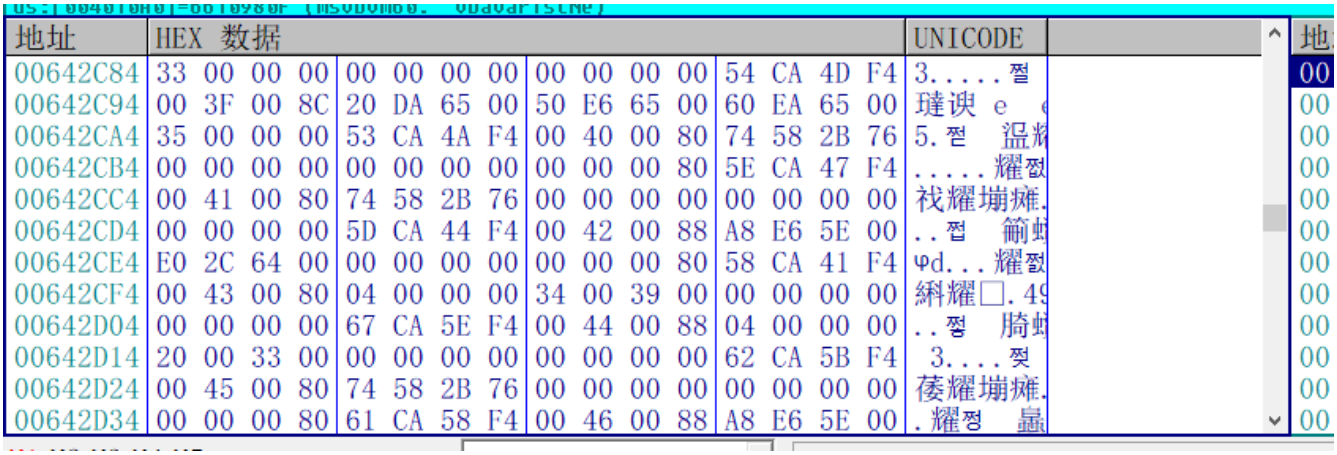


这里会根据vbaVarTstNe的比较结果来判断是否进行跳转，这个函数的VB常用的比较函数，可以肯定这里就是比较正确的序列号的地方了

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



首地址+8才是真正的内容



地址	HEX 数据												UNICODE
00642C84	33	00	00	00	00	00	00	00	00	00	00	00	3.....
00642C94	00	3F	00	8C	20	DA	65	00	50	E6	65	00	?.. ?e.P?e.
00642CA4	35	00	00	00	53	CA	4A	F4	00	40	00	80	5.....@.?
00642CB4	00	00	00	00	00	00	00	00	00	00	00	80	.....
00642CC4	00	41	00	80	74	58	2B	76	00	00	00	00	.A.?tX+v....
00642CD4	00	00	00	00	5D	CA	44	F4	00	42	00	88	....]?D?.B.?
00642CE4	E0	2C	64	00	00	00	00	00	00	00	00	80	?d.....
00642CF4	00	43	00	80	04	00	00	00	34	00	39	00	.C.?....4.9.
00642D04	00	00	00	00	67	CA	5E	F4	00	44	00	88	....g?^?.D.?
00642D14	20	00	33	00	00	00	00	00	00	00	00	00	3.....
00642D24	00	45	00	80	74	58	2B	76	00	00	00	00	.E.?tX+v....
00642D34	00	00	00	80	61	CA	58	F4	00	46	00	88	...?a?X?.F.?

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

```
.  FF15 0C104000  call dword ptr ds:[<&MSVBVM60.__vbaFreeVarList>]   msvbvm60.__vbaFreeVarList
.  83C4 18        add esp,0x18
.  66:85FF        test di,di                                         比较edi
.  75 1C          jnz short bjanes_1.00403A22                        不相等报错
.  8B7D 08        mov edi,dword ptr ss:[ebp+0x8]
.  B8 01000000    mov eax,0x1                                        i++
.  66:0345 E8     add ax,word ptr ss:[ebp-0x18]
.  0F80 94010000  jo bjanes_1.00403BAC
.  8945 E8        mov dword ptr ss:[ebp-0x18],eax
.  33DB           xor ebx,ebx                                        msvbvm60.rtcStrFromVar
.  E9 5AFDFFFF    jmp bjanes_1.0040377C                              开始新一轮循环
>  33DB           xor ebx,ebx                                        msvbvm60.rtcStrFromVar
>  8B35 A4104000  mov esi,dword ptr ds:[<&MSVBVM60.__vbaVarDup>]     msvbvm60.__vbaVarDup
.  B9 04000280    mov ecx,0x80020004
.  894D 98        mov dword ptr ss:[ebp-0x68],ecx
```

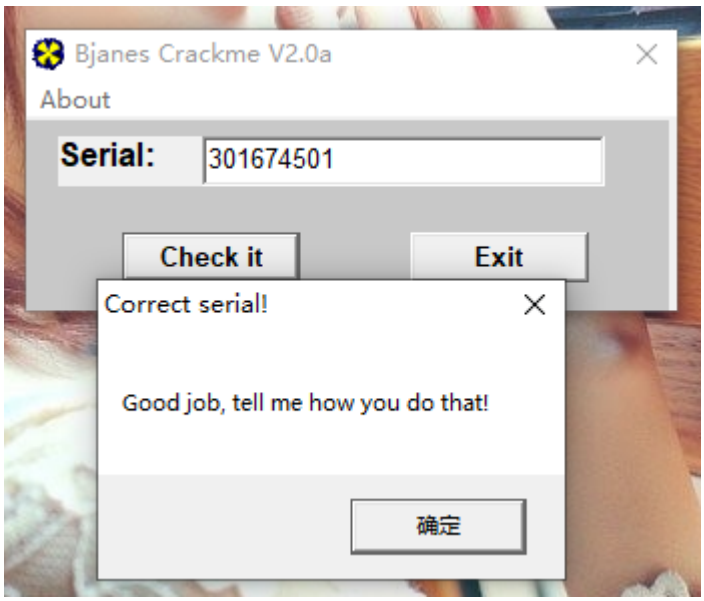
比较di之后，如果不相等则跳转到报错的地方，如果相等则继续往下走，后面两句相当于i++，然后跳转到循环开始处，

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