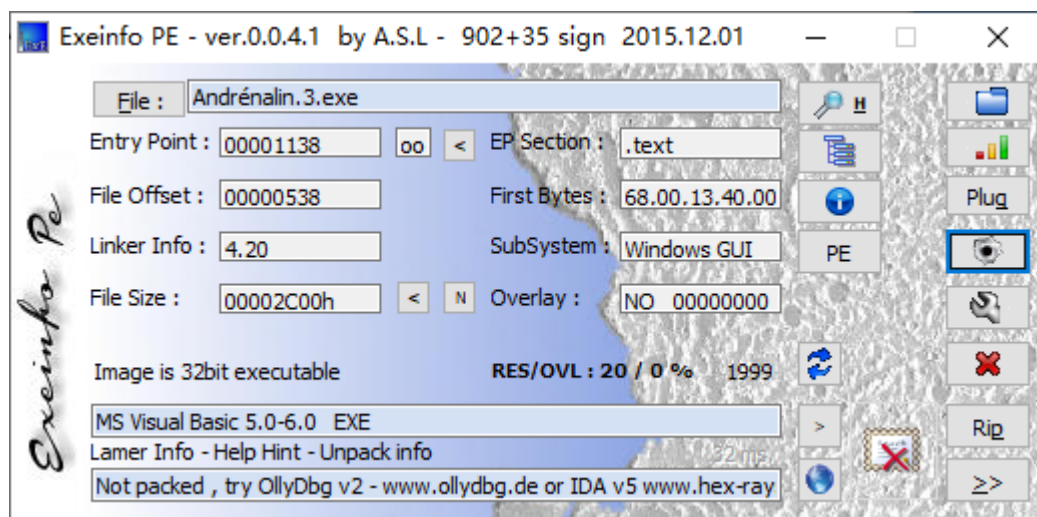


查壳
分析程序
分析算法
写出注册机
验证结果

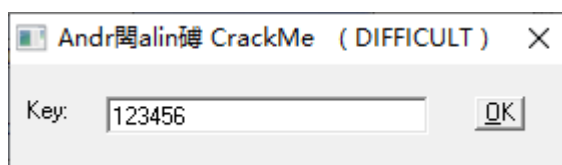
查壳



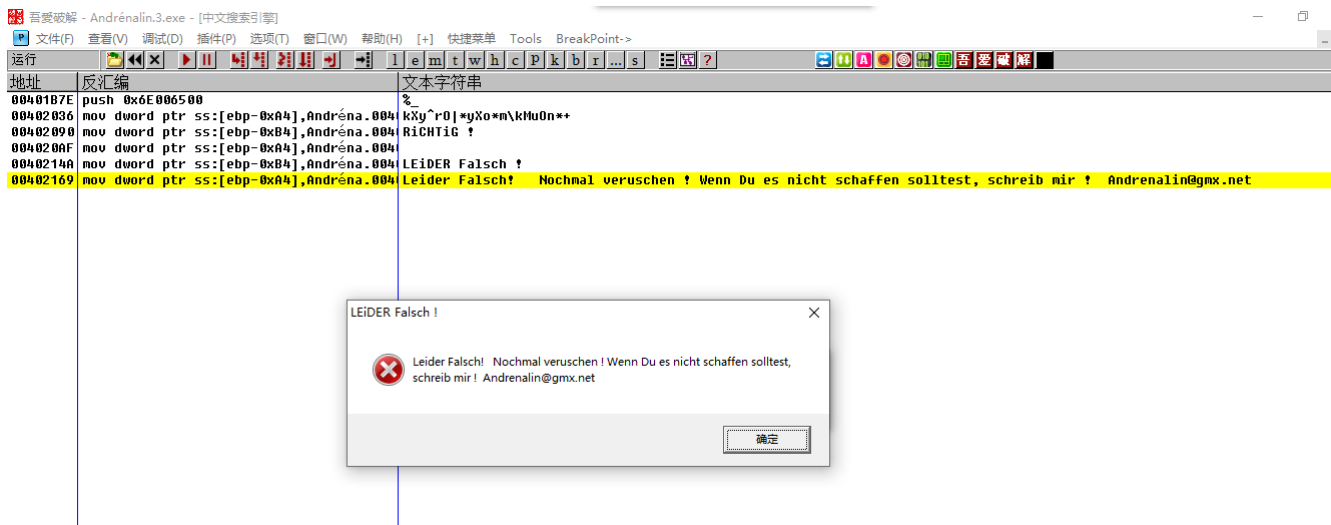
跟008和009这两个crackme一样是同一个作者，还是用VB写的。

分析程序

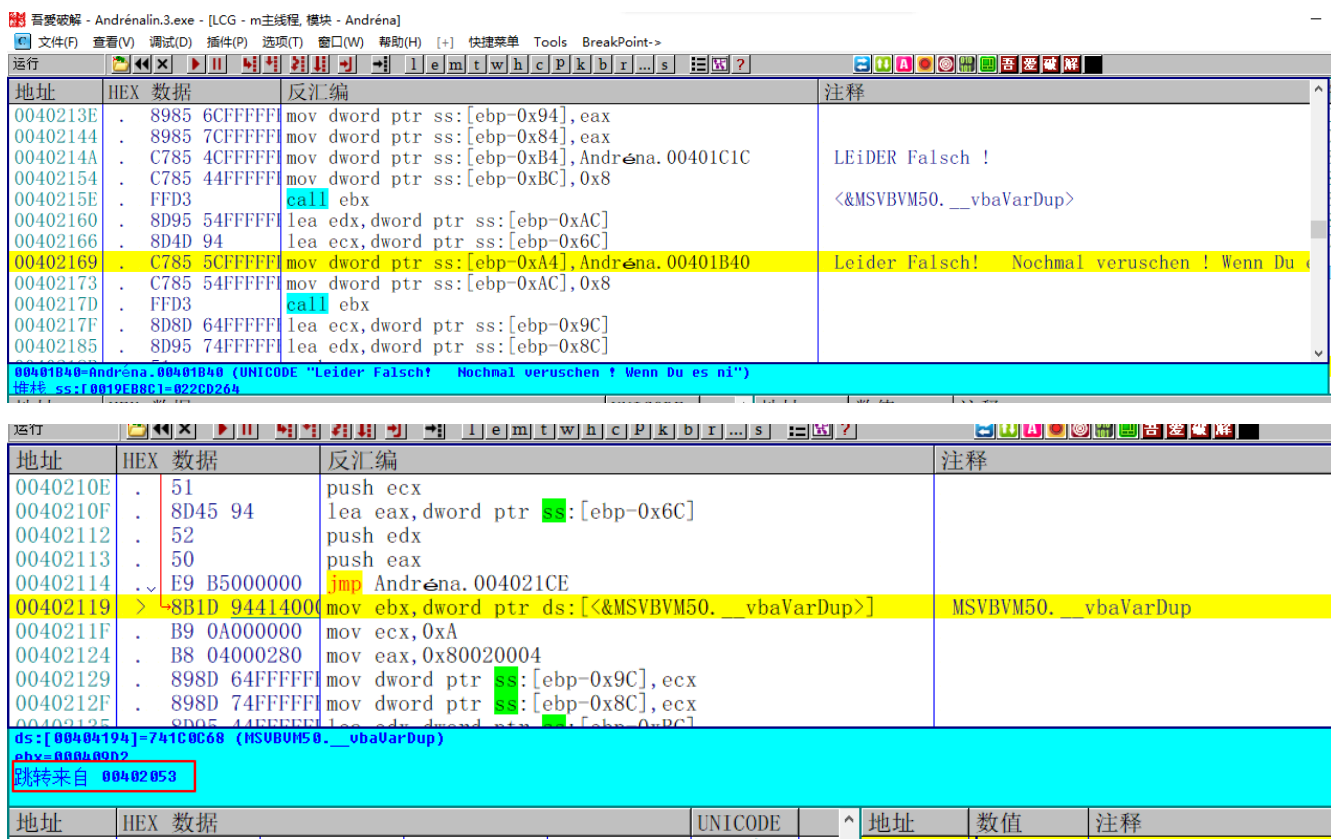
这个程序的保护方式也很简单，只有一个序列号



直接搜索字符串，



根据错误提示来到关键代码处，接着一直往上找，



接着就看到了这个地方的跳转来自0x402053这个位置，跟过去看看

地址	HEX 数据	反汇编	注释
0040202E	. 8D8D 54FFFFFF	lea ecx,dword ptr ss:[ebp-0xAC]	
00402034	. 50	push eax	
00402035	. 51	push ecx	
00402036	. C785 5CFFFFFF	mov dword ptr ss:[ebp-0xA4],Andréna.00401A8C	
00402040	. C785 54FFFFFF	mov dword ptr ss:[ebp-0xAC],0x8008	
0040204A	. FF15 40414000	call dword ptr ds:[<&MSVBVM50.__vbaVarTstEq>]	var18 = NULL var28 = NULL kXy^r0 *yXo*m\kMuOn*+
00402050	. 66:85C0	test ax,ax	__vbaVarTstEq
00402053	. 0F84 C0000000	je Andréna.00402119	
00402059	. FF15 6C414000	call dword ptr ds:[<&MSVBVM50.#rtcBeep_534>]	MSVBVM50.rtcBeep
0040205F	. 8B1D 94414000	mov ebx,dword ptr ds:[<&MSVBVM50.__vbaVarDup>]	MSVBVM50.__vbaVarDup
00402065	. B9 0A000000	mov ecx,0xA	
0040206A	. B8 04000280	mov eax,0x80020004	
0040206F	. 898D 64FFFFFF	mov dword ptr ss:[ebp-0x9C],ecx	
00402075	. 898D 74FFFFFF	mov dword ptr ss:[ebp-0x8C],ecx	
0040207B	. 8B05 44FFFFFF	mov ebx,dword ptr ds:[ebp-0x9C]	

这里有一个关键的比较，根据ax的值来提示是否正确。 kXy^r0|*yXo*m\kMuOn*+ 我一开始以为上面这个字符串就是序列号，结果发现我想多了，看来是有算法，没办法，单步跟吧。

分析算法

大致的校验过程如下

地址	HEX 数据	反汇编	注释
00401EED	. 8D4D BC	lea ecx,dword ptr ss:[ebp-0x44]	
00401EF0	. 8945 9C	mov dword ptr ss:[ebp-0x64],eax	
00401EF3	. C745 94 08000000	mov dword ptr ss:[ebp-0x6C],0x8	
00401EFA	. FFD6	call esi	MSVBVM50.__vbaVarMove; <&MSVBVM50.__vbaVarMove>
00401EFC	. 8D4D A4	lea ecx,dword ptr ss:[ebp-0x5C]	ecx=序列号
00401EFF	. FF15 AC414000	call dword ptr ds:[<&MSVBVM50.__vbaFreeObj>]	MSVBVM50.__vbaFreeObj
00401F05	. B9 02000000	mov ecx,0x2	
00401F0A	. B8 01000000	mov eax,0x1	
00401F0F	. 898D 54FFFFFF	mov dword ptr ss:[ebp-0xAC],ecx	
00401F15	. 898D 44FFFFFF	mov dword ptr ss:[ebp-0xBC],ecx	
00401F1B	. 8D8D 54FFFFFF	lea ecx,dword ptr ss:[ebp-0xAC]	
00401F21	. 8985 5CFFFFFF	mov dword ptr ss:[ebp-0xA4],eax	
00401F27	. 8985 4CFFFFFF	mov dword ptr ss:[ebp-0xB4],eax	
00401F2D	. 8D55 BC	lea edx,dword ptr ss:[ebp-0x44]	
00401F30	. 51	push ecx	Step8 = 0019F124
00401F31	. 8D45 94	lea eax,dword ptr ss:[ebp-0x6C]	
00401F34	. 52	push edx	
00401F35	. 50	push eax	var18 = 00000003 retBuffer8 = 0019F19C
00401F36	. FF15 14414000	call dword ptr ds:[<&MSVBVM50.__vbaLenVar>]	求序列号长度

首先获取到序列号的长度

地址	HEX 数据	反汇编	注释
00401F30	. 51	push ecx	Step8 = 0019F124
00401F31	. 8D45 94	lea eax,dword ptr ss:[ebp-0x6C]	
00401F34	. 52	push edx	
00401F35	. 50	push eax	var18 = 00000003 retBuffer8 = 0019F19C
00401F36	. FF15 14414000	call dword ptr ds:[<&MSVBVM50.__vbaLenVar>]	求序列号长度
00401F3C	. 8D8D 44FFFFFF	lea ecx,dword ptr ss:[ebp-0xBC]	
00401F42	. 50	push eax	End8 = 0019F19C
00401F43	. 8D95 ECFEFFFF	lea edx,dword ptr ss:[ebp-0x114]	
00401F49	. 51	push ecx	Start8 = 0019F124
00401F4A	. 8D85 FCFEFFFF	lea eax,dword ptr ss:[ebp-0x104]	
00401F50	. 52	push edx	TMPend8 = 00000003
00401F51	. 8D4D DC	lea ecx,dword ptr ss:[ebp-0x24]	
00401F54	. 50	push eax	
00401F55	. 51	push ecx	TMPstep8 = 0019F19C Counter8 = 0019F124
00401F56	. FF15 1C414000	call dword ptr ds:[<&MSVBVM50.__vbaVarForInit>]	__vbaVarForInit
00401F5C	. 8B1D 68414000	mov ebx,dword ptr ds:[<&MSVBVM50.__vbaVarCat>]	MSVBVM50.__vbaVarCat
00401F62	. 8B3D 00414000	mov edi,dword ptr ds:[<&MSVBVM50.__vbaFreeVarList>]	MSVBVM50.__vbaFreeVarList
00401F68	. 85C0	test eax,eax	
00401F6A	. 0F84 BB000000	je Andréna.0040202B	

然后将序列号的长度作为循环的次数

地址	HEX 数据	反汇编	注释
00401F94	. 52	push edx	RetBUFFER = 00000003
00401F95	. FF15 34414000	call dword ptr ds:[<&MSVBVM50. #rtcMidCharVar_6]	rtcMidCharVar
00401F9B	. 8D45 84	lea eax, dword ptr ss:[ebp-0x7C]	把序列号转成ASCII
00401F9E	. 8D4D A8	lea ecx, dword ptr ss:[ebp-0x58]	
00401FA1	. 50	push eax	String8 = 0019F19C
00401FA2	. 51	push ecx	ARG2 = 0019F124
00401FA3	. FF15 64414000	call dword ptr ds:[<&MSVBVM50. __vbaStrVarVal>]	取出序列号的ASCII每一位
00401FA9	. 50	push eax	String = "□"
00401FAA	. FF15 08414000	call dword ptr ds:[<&MSVBVM50. #rtcAnsiValueBst]	取出序列号的ASCII值
00401FB0	. 66:05 0A00	add ax, 0xA	序列号的ASCII值+0xA
00401FB4	. 0F80 B0020000	jo Andrena.0040226A	溢出则报错
00401FBA	. 0FBFD0	movsx edx, ax	edx=序列号的ASCII值+0xA
00401FBD	. 52	push edx	
00401FBE	. FF15 70414000	call dword ptr ds:[<&MSVBVM50. #rtcBstrFromAnsi]	把相加后的数值转为字符串
00401FC4	. 8985 7CFFFFFF	mov dword ptr ss:[ebp-0x84], eax	把字符串保存到[ebp-0x84]
00401FCA	. 8D45 CC	lea eax, dword ptr ss:[ebp-0x34]	
00401FCD	. 8D8D 74FFFFFF	lea ecx, dword ptr ss:[ebp-0x8C]	ecx=序列号ASCII+0xA的值
00401FD3	. 50	push eax	
00401FD4	. 8D95 64FFFFFF	lea edx, dword ptr ss:[ebp-0x9C]	
ds:[00404164]=7411A825 (MSUBVM50. __vbaStrUarVal)			

接着会取出序列号ASCII的每一位，然后将序列号的ASCII值+0xA之后，再将这个值转为字符串。

00401FC4	. 8985 7CFFFFFF	mov dword ptr ss:[ebp-0x84], eax	把字符串保存到[ebp-0x84]
00401FCA	. 8D45 CC	lea eax, dword ptr ss:[ebp-0x34]	
00401FCD	. 8D8D 74FFFFFF	lea ecx, dword ptr ss:[ebp-0x8C]	ecx=序列号ASCII+0xA的值
00401FD3	. 50	push eax	
00401FD4	. 8D95 64FFFFFF	lea edx, dword ptr ss:[ebp-0x9C]	
00401FDA	. 51	push ecx	
00401FDB	. 52	push edx	
00401FDC	. C785 74FFFFFF	mov dword ptr ss:[ebp-0x8C], 0x8	
00401FE6	. FFD3	call ebx	拼接转换后的字符串
00401FE8	. 8BD0	mov edx, eax	
00401FEA	. 8D4D CC	lea ecx, dword ptr ss:[ebp-0x34]	
00401FED	. FFD6	call esi	把结果移动到[ebp-0x34]
00401FEF	. 8D4D A8	lea ecx, dword ptr ss:[ebp-0x58]	
00401FF2	. FF15 B0414000	call dword ptr ds:[<&MSVBVM50. __vbaFreeStr>]	MSVBVM50. __vbaFreeStr
00401FF8	. 8D85 74FFFFFF	lea eax, dword ptr ss:[ebp-0x8C]	
00401FFE	. 8D4D 84	lea ecx, dword ptr ss:[ebp-0x7C]	
00402001	. 50	push eax	
00402002	. 8D55 94	lea edx, dword ptr ss:[ebp-0x6C]	
00402005	. 51	push ecx	

然后又是国际惯例了，会通过一个函数将结果保存到[ebp-0x34]这个位置。(VB的程序每次都是这样，这不是给逆向人员开绿色通道吗？)那么这个循环我们只要跟了一次，然后直接看[ebp-0x34]的结果就可以了。

00402020	. FF15 A4414000	call dword ptr ds:[<&MSVBVM50. __vbaVarForNext>]	__vbaVarForNext
00402026	. E9 3DFFFFFF	jmp Andrena.00401F68	
0040202B	. 8D45 CC	lea eax, dword ptr ss:[ebp-0x34]	
0040202E	. 8D8D 54FFFFFF	lea ecx, dword ptr ss:[ebp-0xAC]	
00402034	. 50	push eax	var18 = 0019F19C
00402035	. 51	push ecx	var28 = 0019F124
00402036	. C785 5CFFFFFF	mov dword ptr ss:[ebp-0xA4], Andrena.00401A8C	kXy r0 *yXo*m\kMuOn*+
00402040	. C785 54FFFFFF	mov dword ptr ss:[ebp-0xAC], 0x8008	
0040204A	. FF15 40414000	call dword ptr ds:[<&MSVBVM50. __vbaVarTstEq>]	__vbaVarTstEq
00402050	. 66:85C0	test ax, ax	
00402053	. 0F84 C0000000	je Andrena.00402119	
00402059	. FF15 6C414000	call dword ptr ds:[<&MSVBVM50. #rtcBeep_534>]	MSVBVM50. rtcBeep
0040205F	. 8B1D 94414000	mov ebx, dword ptr ds:[<&MSVBVM50. __vbaVarDup>]	MSVBVM50. __vbaVarDup
ds:[00404140]=741CB99E (MSUBVM50. __vbaVarTstEq)			

地址	HEX 数据	Unicode	地址	数值	注释
00682974	3B 00 3C 00	3D 00 3E 00	3F 00 40 00	41 00 42 00	:<=>2@AB
00682984	43 00 00 00	00 00 59 2A	5F 5C 77 8D	E7 9E 62 C5	C. . . 肩起
00682994	00 0B 00 88	12 00 00 00	31 00 32 00	33 00 34 00	□ 蛸 □. 12
006829A4	35 00 36 00	37 00 38 00	39 00 00 00	79 00 00 00	56789. y.
006829B4	B4 F6 19 00	E2 9E 1F C5	00 0C 00 8C	3C 13 14 6E	□ 戮矣
006829C4	00 00 00 00	02 00 00 00	80 6F 66 00	00 00 00 00	. . . 灌 f.
006829D4	00 00 00 00	03 00 00 00	70 00 00 00	E9 9E 18 C5	. . . □. 點
006829E4	00 0D 00 88	60 1C AC 76	00 10 AC 76	01 00 00 00	□ 蛸 □ 確 □ 確
006829F4	30 B5 64 00	5A DD 13 17	E7 68 5B 4A	9A F6 59 2A	□ d □ 棧 靴
00682A04	59 5C 77 8D	94 9E 15 C5	00 0E 00 88	B4 DE 4B 6E	肩起 腐 岩 □
0019F098	0019F124				var28 = 0019F124
0019F09C	0019F19C				var18 = 0019F19C
0019F0A0	0019F1DC				
0019F0A4	0019F2B8				ASCII " 蒞 "
0019F0A8	022CD194				
0019F0AC	0004B604				Unicode "-security-logon-11-1-1"
0019F0B0	005B0FC0				
0019F0B4	00000000				
0019F0B8	FFB6171D				
0019F0BC	00000003				

循环结束之后，会把最后的结果跟硬编码的一个字符串作比较，根据比较的结果提示正确或者错误。那么我们很容易就能写出这个crackme的注册机了

写出注册机

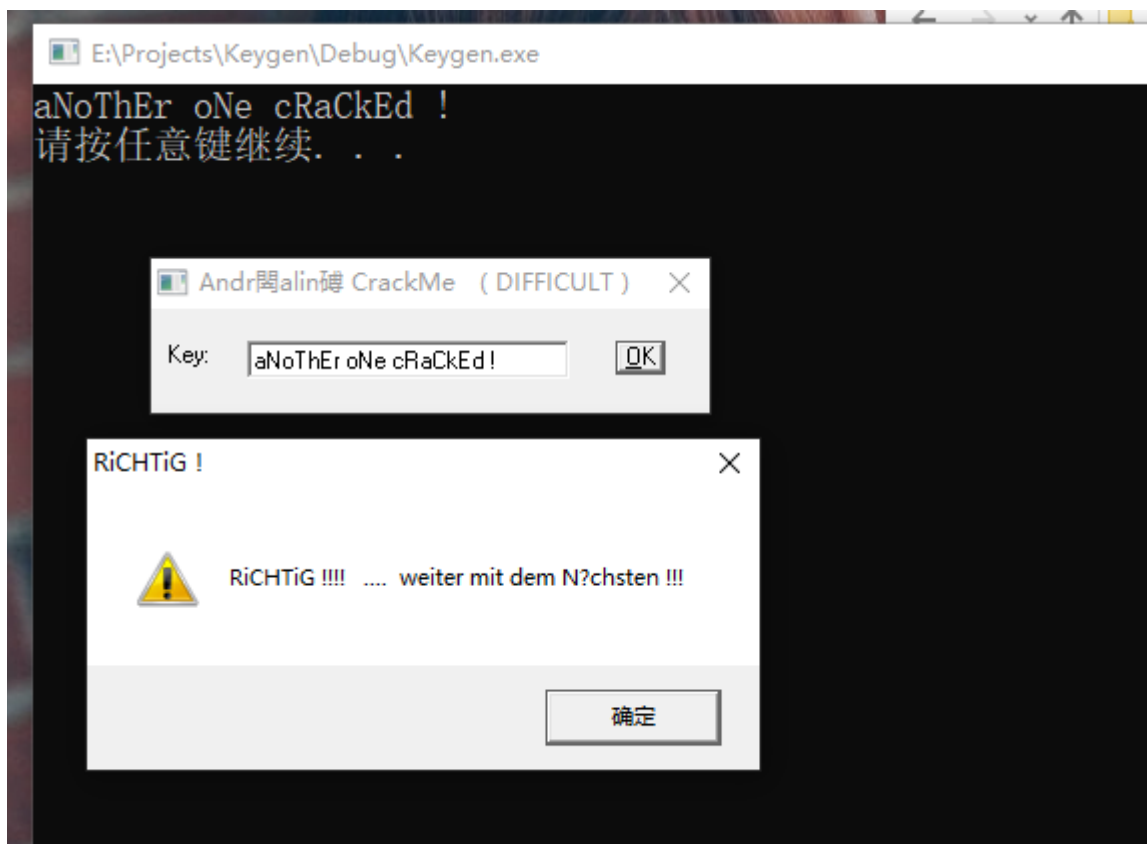
```
int CalcKey()
{
    char result[MAX_PATH] = { 0 };
    char key[MAX_PATH] = { "kXy^rO|*yXo*m\\kMuOn*+" };
    int keyLen = strlen(key);

    for (int i = 0; i < keyLen; i++)
    {
        result[i] = key[i] - 0xA;
    }

    printf("%s\n", result);

    return 0;
}
```

验证结果



把注册机的结果全部复制下来，显示正确，那么这个crackme就完成了

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