

软件保护技术作业 01

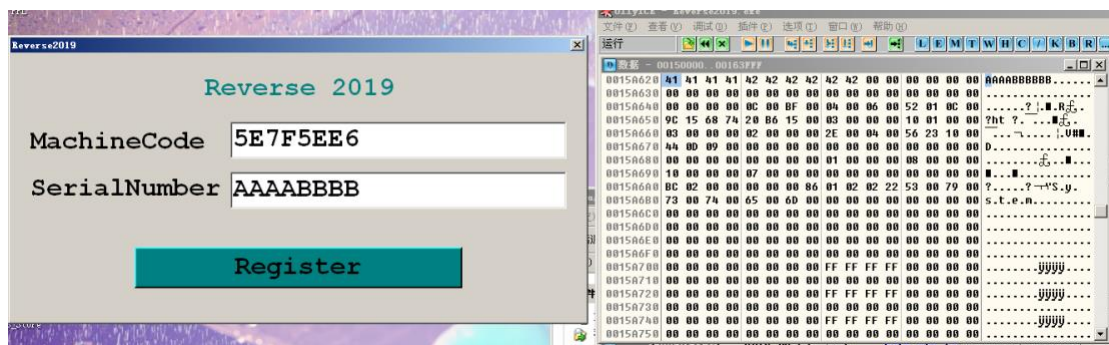
作者：刘皓 学号：3160104994

一、实验要求

1. 用 OllyDbg+IDA Pro(可以只用其中之一,也可以一起用)分析 reverse2019.exe, 找到与注册码计算相关的函数, 写出分析报告。分析报告内容包括推理过程、跟踪步骤、核心代码注释。
2. 用 VC6 编写注册机程序, 输入一个 MachineCode, 输出一个正确的注册码。该注册机程序的工程创建请按以下步骤进行:
File->New->Projects->Win32 Console Application
3. 分析报告请命名为"reverse01.pdf"; VC6 工程文件夹请命名为"reverse01"。
4. 新建一个文件夹, 命名为"学号姓名", 并把步骤 3 所涉及的文件及文件夹全部拷到"学号姓名"文件夹内, 并压缩
5. 此文件夹生成"学号姓名.zip", 上传到以下 ftp:
ftp://bhhreverse:bhhreverse@10.71.45.100/作业 01

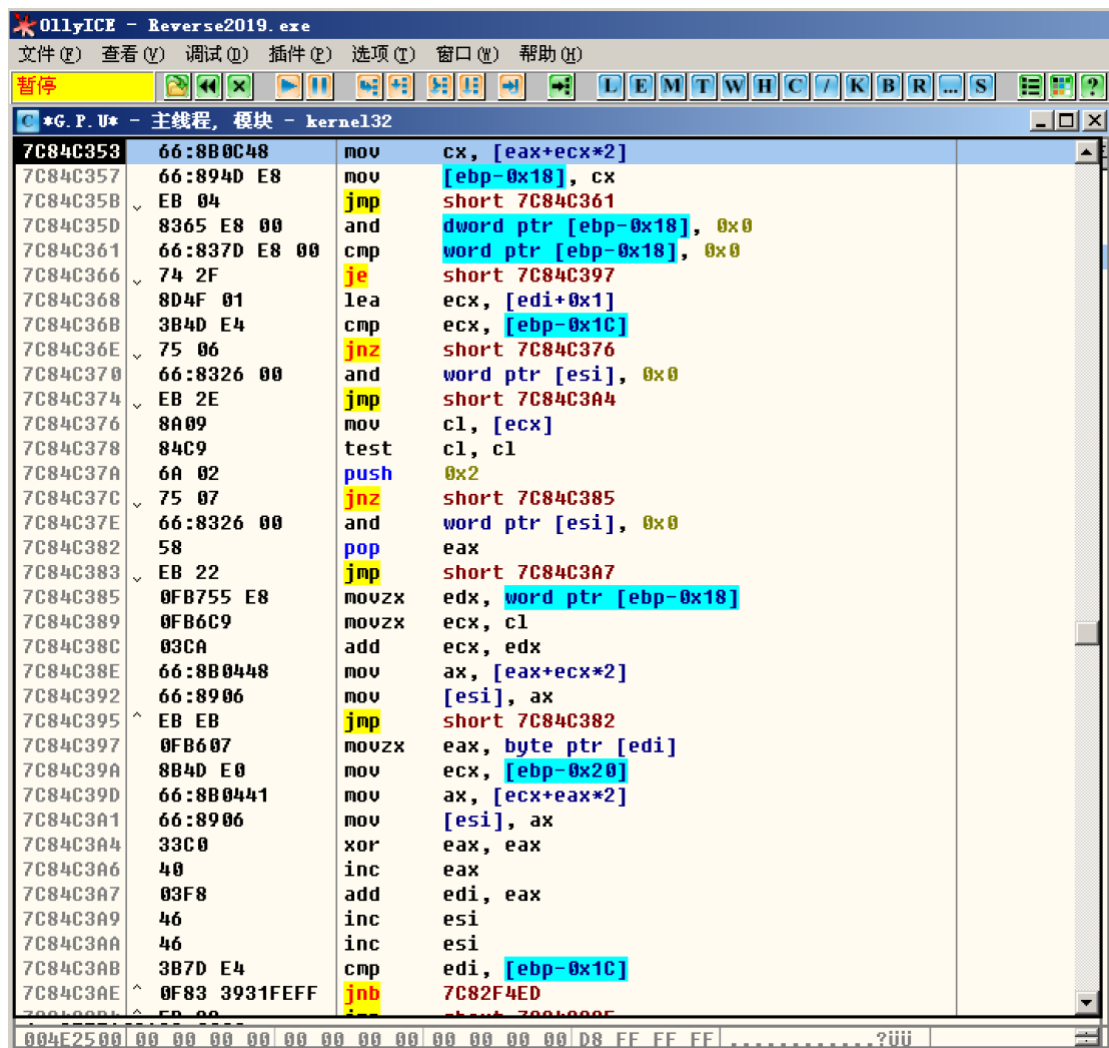
二、实验过程

首先随意输入一个注册码, 比如 AAAABBBB, 然后在内存中搜索, 于第一个处设置硬件断点。

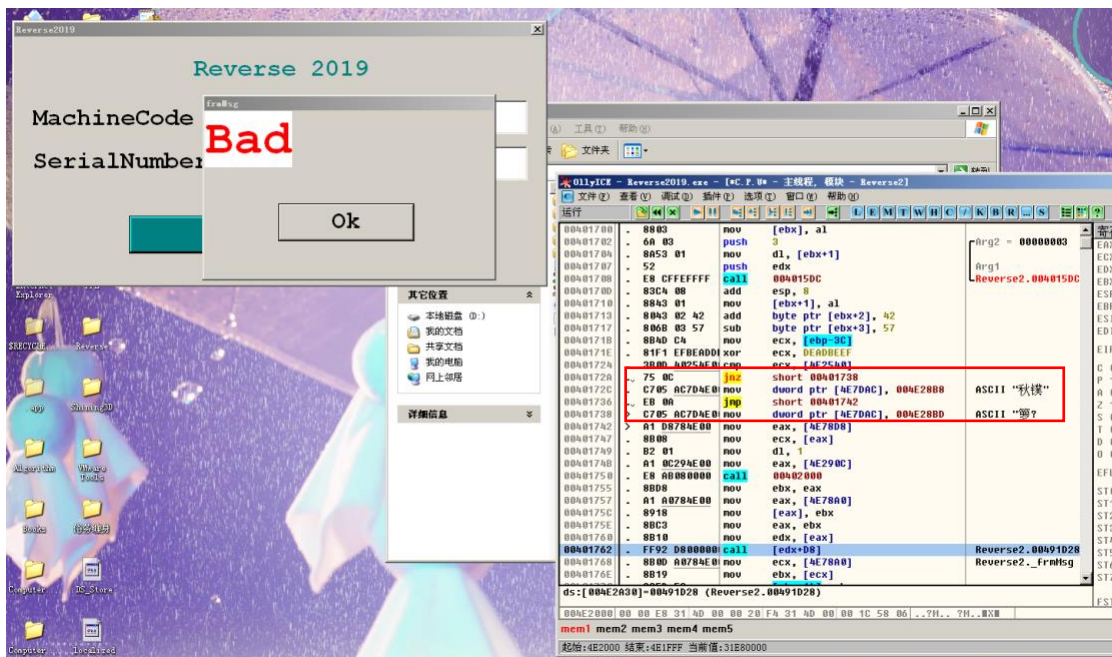




点击 Register 之后，断点生效，我们取消硬件断电，然后开始跟踪。



进入用户态之后继续跟踪，直到弹出窗口。



查看弹出窗口之前的代码，发现红色方框内比较可疑，可能与弹出窗口之前的判断有关，在这加个断点尝试一下吧。

cmp 之后的 ZF 为 0，我们改为 1 试一下，就会运行 jump short 00401742 那条指令。

00401710	88 03	mov	[ebx+0x1], al			C 0	ES	0023	32位	0 (FFFFFFFF)
00401713	88 03	mov	[ebx+0x1], al			P 1	CS	001B	32位	0 (FFFFFFFF)
00401717	80 03	add	byte ptr [ebx+0x3], 0x57	修改serial number		A 1	SS	0023	32位	0 (FFFFFFFF)
0040171B	80 03	sub	byte ptr [ebx+0x3], 0x57	修改serial number		Z 0	DS	0023	32位	0 (FFFFFFFF)
0040171E	80 03	mov	ecx, [ebp-0x3C]			S 0	FS	003B	32位	7FDF000 (FFF)
00401720	80 03	xor	ecx, 0xDEADBEEF			T 0	GS	0000	NULL	
00401724	80 03	cmp	ecx, [0x4E2540]	ecx应为machine code		D 0				
0040172A	75 0C	jnz	short 00401738	替换之后good		O 1	LastErr	ERROR_SUCCESS (00000)		
0040172C	C7 05	mov	dwrd ptr [0x4E7DAC], 004E28B8	good时候就mov这个		EFL	00000016	{0,NB,NE,A,NS,PE,L,L		
00401730	EB 0A	jmp	short 00401742							
00401738	80 03	mov	ecx, [ebp-0x3C]	修改serial number		A 1	SS	0023	32位	0 (FFFFFFFF)
0040173B	80 03	sub	byte ptr [ebx+0x3], 0x57	修改serial number		Z 1	DS	0023	32位	0 (FFFFFFFF)
0040173E	80 03	mov	ecx, [ebp-0x3C]			S 0	FS	003B	32位	7FDF000 (FFF)
00401740	80 03	xor	ecx, 0xDEADBEEF			T 0	GS	0000	NULL	
00401744	80 03	cmp	ecx, [0x4E2540]	ecx应为machine code		D 0				
0040174A	75 0C	jnz	short 00401738	替换之后good		O 1	LastErr	ERROR_SUCCESS (00000)		
0040174C	C7 05	mov	dwrd ptr [0x4E7DAC], 004E28B8	good时候就mov这个		EFL	00000056	{0,NB,E,BE,NS,PE,L,L		
00401750	EB 0A	jmp	short 00401742							
00401758	C7 05	mov	dwrd ptr [0x4E7DAC], 004E28B8	ASCII "梦?"		MM0	-2.349076e-09, -8.458504e-39			

继续运行发现会弹出 Good 窗口，那么我们如果将 jnz short 00401738 改为 nop 并保存，就使用爆破法破解了这个程序，但为完成本次作业要求，我们继续探究，找出注册码的算法。

继续追踪，发现程序对该序列号上做了一些计算，先对最低 1 个字节作为 rol2bits 的处理。

004016F9	. 8D5D C4	lea	ebx, [ebp-0x3C]	Arg2 = 00000002	S 0 FS 003B 32 07
004016FB	. 6A 02	push	0x2	Arg1	T 0 GS 0000 NULL
004016F5	. 8A03	mov	al, [ebx]	rol 2	D 0
004016F7	. 50	push	eax		0 0 LastErrr ERROR
004016F8	. E8 CFEFFFFFFF	call	004015CC		EFL 00000206 (NO, NB
004016FD	. 83C4 08	add	esp, 0x8		
00401700	. 8803	mov	[ebx], al		
00401702	. 6A 03	push	0x3	Arg2 = 00000003	MM0 1.744600e-39,
00401704	. 8A53 01	mov	d1, [ebx+0x1]	Arg1	MM1 +NaN 7FFDF700,
00401707	. 52	push	edx	ror 3	MM2 0.0,
00401708	. E8 CFEFFFFFFF	call	004015DC		MM3 -2.348910e-09,
0040170D	. 83C4 08	add	esp, 0x8		MM4 -5.614212e-33,
00401710	. 8843 01	mov	[ebx+0x1], al		MM5 -1.000350,
00401713	. 8043 02 42	add	byte ptr [ebx+0x2], 0x42	修改serial number	MM6 -3.031649e-13,
00401717	. 806B 03 57	sub	byte ptr [ebx+0x3], 0x57	修改serial number	MM7 -1.136868e-13,
0040171B	. 884D C4	mov	ecx, [ebp-0x3C]		
0040171E	. 81F1 EFBADDDI	xor	ecx, 0xDEADBEEF		
00401724	. 3B00 40254E0	cmp	ecx, [0x4E2540]	ecx应为machine code	

004E2520	05 04 00 00 FC FF FF FF	00 00 00 00 00 00 00 00	Y...?jij	0012FB70	. 0012FBC4	
004E2530	D8 FF FF FF 00 00 05 00	00 00 00 00 1C 25 4E 00	?jij...?N	0012FB74	. 77D4048F	user32.77D4048F
004E2540	E6 5E 7F 5E 63 3A 5C 00	25 30 38 58 00 00 25 58	...c:\%00X.%X	0012FB78	. 77D18830	user32.77D18830
004E2550	00 00 00 00 34 18 40 00	05 04 00 00 FC FF FF FF	...4000.Y...?jij	0012FB7C	. FFFFFFFF	
004E2560	00 00 00 00 00 00 00 00	D8 FF FF FF 00 00 05 00	...?jij..Y	0012FB80	. 77D1882A	user32.77D1882A
004E2570	00 00 00 00 54 25 4E 00	74 14 40 00 04 00 00 00	...T%N.t000. ...	0012FB84	. 77D28EA0	user32.77D28EA0
004E2580	FC FF FF FF 00 00 00 00	00 00 00 00 D8 FF FF FF	?jij...?jij	0012FB88	. AAAA8BEE	

然后对倒数第 2 个字节 ror3bits。

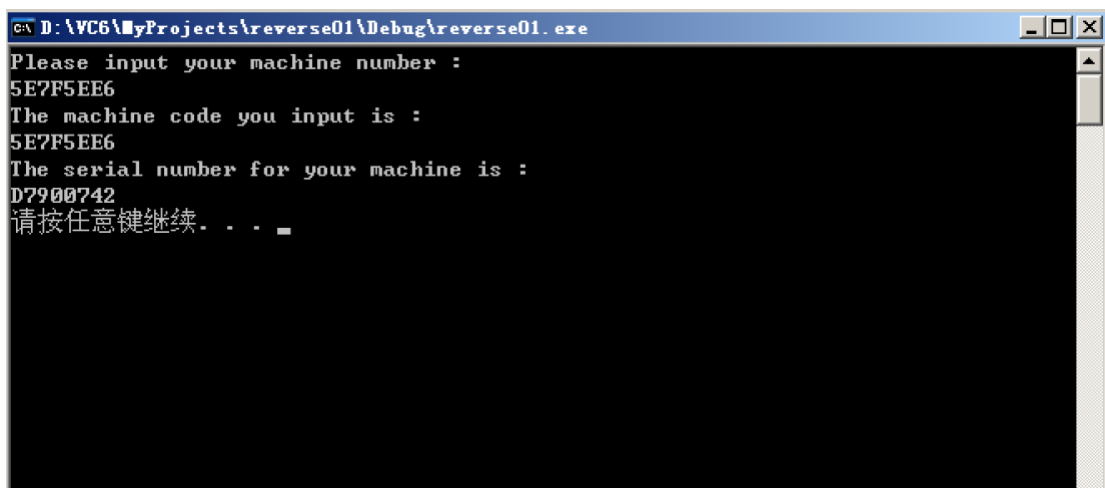
004016FD	. 83C4 08	add	esp, 0x8		
00401700	. 8803	mov	[ebx], al		
00401702	. 6A 03	push	0x3	Arg2 = 00000003	MM0 1.744600e-39,
00401704	. 8A53 01	mov	d1, [ebx+0x1]	Arg1	MM1 +NaN 7FFDF700,
00401707	. 52	push	edx	ror 3	MM2 0.0,
00401708	. E8 CFEFFFFFFF	call	004015DC		MM3 -2.348910e-09,
0040170D	. 83C4 08	add	esp, 0x8		MM4 -5.614212e-33,
00401710	. 8843 01	mov	[ebx+0x1], al		MM5 -1.000350,
00401713	. 8043 02 42	add	byte ptr [ebx+0x2], 0x42	修改serial number	MM6 -3.031649e-13,
00401717	. 806B 03 57	sub	byte ptr [ebx+0x3], 0x57	修改serial number	MM7 -1.136868e-13,
0040171B	. 884D C4	mov	ecx, [ebp-0x3C]		
0040171E	. 81F1 EFBADDDI	xor	ecx, 0xDEADBEEF		
00401724	. 3B00 40254E0	cmp	ecx, [0x4E2540]	ecx应为machine code	

004E2520	05 04 00 00 FC FF FF FF	00 00 00 00 00 00 00 00	Y...?jij	0012FB70	. 0012FBC4	
004E2530	D8 FF FF FF 00 00 05 00	00 00 00 00 1C 25 4E 00	?jij...?N	0012FB74	. 77D4048F	user32.77D4048F
004E2540	E6 5E 7F 5E 63 3A 5C 00	25 30 38 58 00 00 25 58	...c:\%00X.%X	0012FB78	. 77D18830	user32.77D18830
004E2550	00 00 00 00 34 18 40 00	05 04 00 00 FC FF FF FF	...4000.Y...?jij	0012FB7C	. FFFFFFFF	
004E2560	00 00 00 00 00 00 00 00	D8 FF FF FF 00 00 05 00	...?jij..Y	0012FB80	. 77D1882A	user32.77D1882A
004E2570	00 00 00 00 54 25 4E 00	74 14 40 00 04 00 00 00	...T%N.t000. ...	0012FB84	. 77D28EA0	user32.77D28EA0
004E2580	FC FF FF FF 00 00 00 00	00 00 00 00 D8 FF FF FF	?jij...?jij	0012FB88	. AAAA77EE	

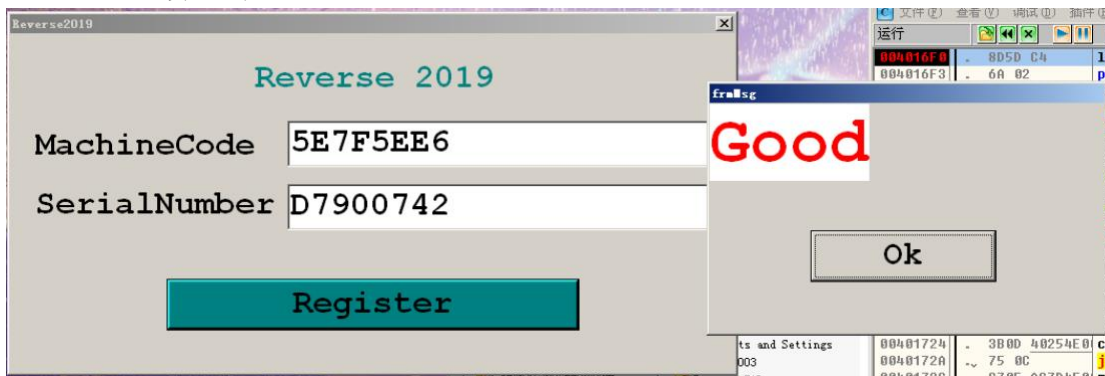
接下来对第 1byte 分别+0x42 和第 1byte-0x57，然后和那个 magic number 作 xor，就来到了我们之前说的判断处。

至此全部的计算就完成了，我们的目的就使得输入的注册码在 xor 后与 Machine Code 一致，那只要一步一步倒回去就可以了。

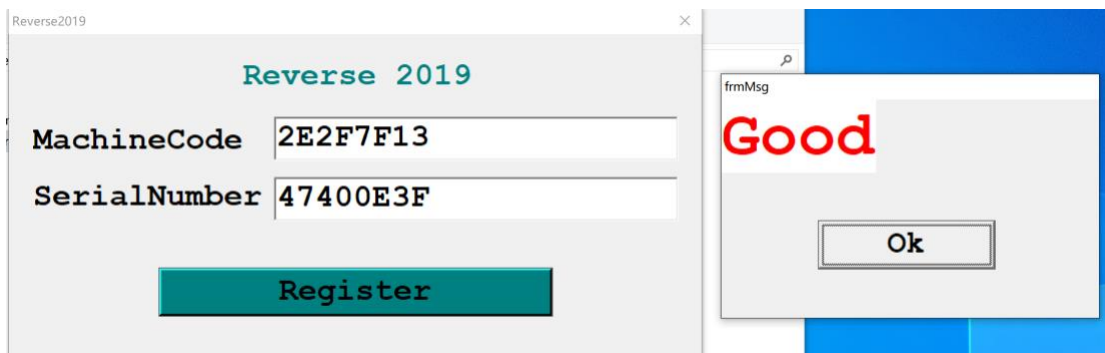
对 Machine Code 先与 magic number xor，然后第 1byte-0x42，第 2byte+0x57，第 3byte rol 3bits，第 4byte ror 2bits，即可得到注册码，编写注册机程序，然后验证。



发现由此计算出来的注册码是正确的。



最终在另一台电脑（Windows 10）上进行验证，我们的注册机仍然是可行的。



三、注册机代码

```
1 reverse01.cpp : Defines the entry point for the console
application
2
3
4 #include <stdio.h>
```

```

5
6 using namespace std;
7
8 unsigned char ROL(unsigned char val, int n)
9 {
10     return (val << n) | (val >> (8 - n));
11 }
12
13 unsigned char ROR(unsigned char val, int n)
14 {
15     return (val >> n) | (val << (8 - n));
16 }
17
18 int main() {
19     unsigned int magicNumber = 0xDEADBEEF;
20     union Mcode{
21         unsigned char byte[4];
22         unsigned int Code;
23     } MachineCode;
24     printf("Please input your machine number :\n");
25     scanf("%x", &MachineCode.Code);
26     MachineCode.Code = 0x5E7F5EE6;
27     printf("The machine code you input is :\n%X\n",
MachineCode.Code);
28     MachineCode.Code ^= magicNumber;
29     //printf("%X\n", MachineCode.Code);
30     MachineCode.byte[0] = ROR(MachineCode.byte[0], 2);
31     //printf("%X\n", MachineCode.Code);
32     MachineCode.byte[1] = ROL(MachineCode.byte[1], 3);
33     //printf("%X\n", MachineCode.Code);
34     MachineCode.byte[2] -= 0x42;
35     //printf("%X\n", MachineCode.Code);
36     MachineCode.byte[3] += 0x57;
37     printf("The serial number for your machine is :\n%X\n",
MachineCode.Code);

```



```
38     system("pause");  
39     return 0;  
40 }
```