软件保护技术作业 01

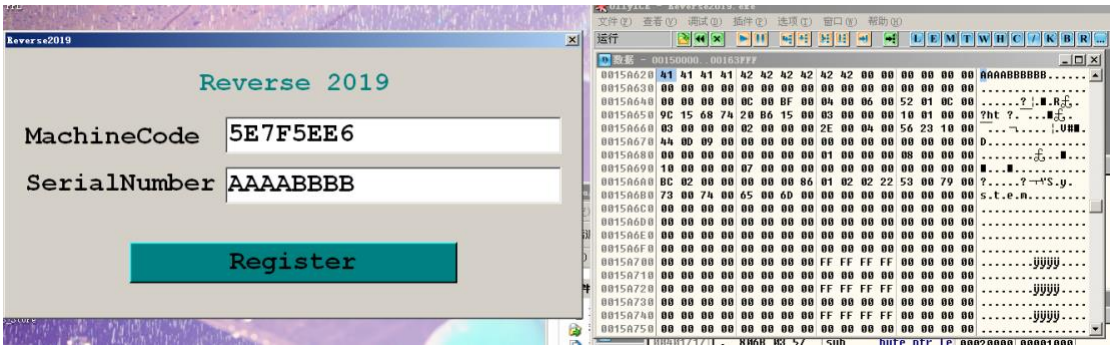
作者：刘皓 学号：3160104994

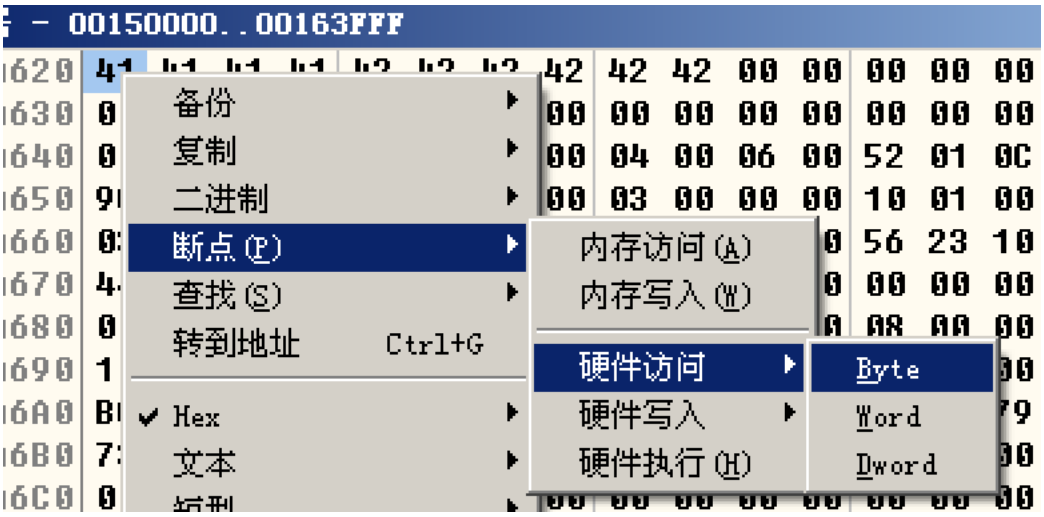
一、 实验要求

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

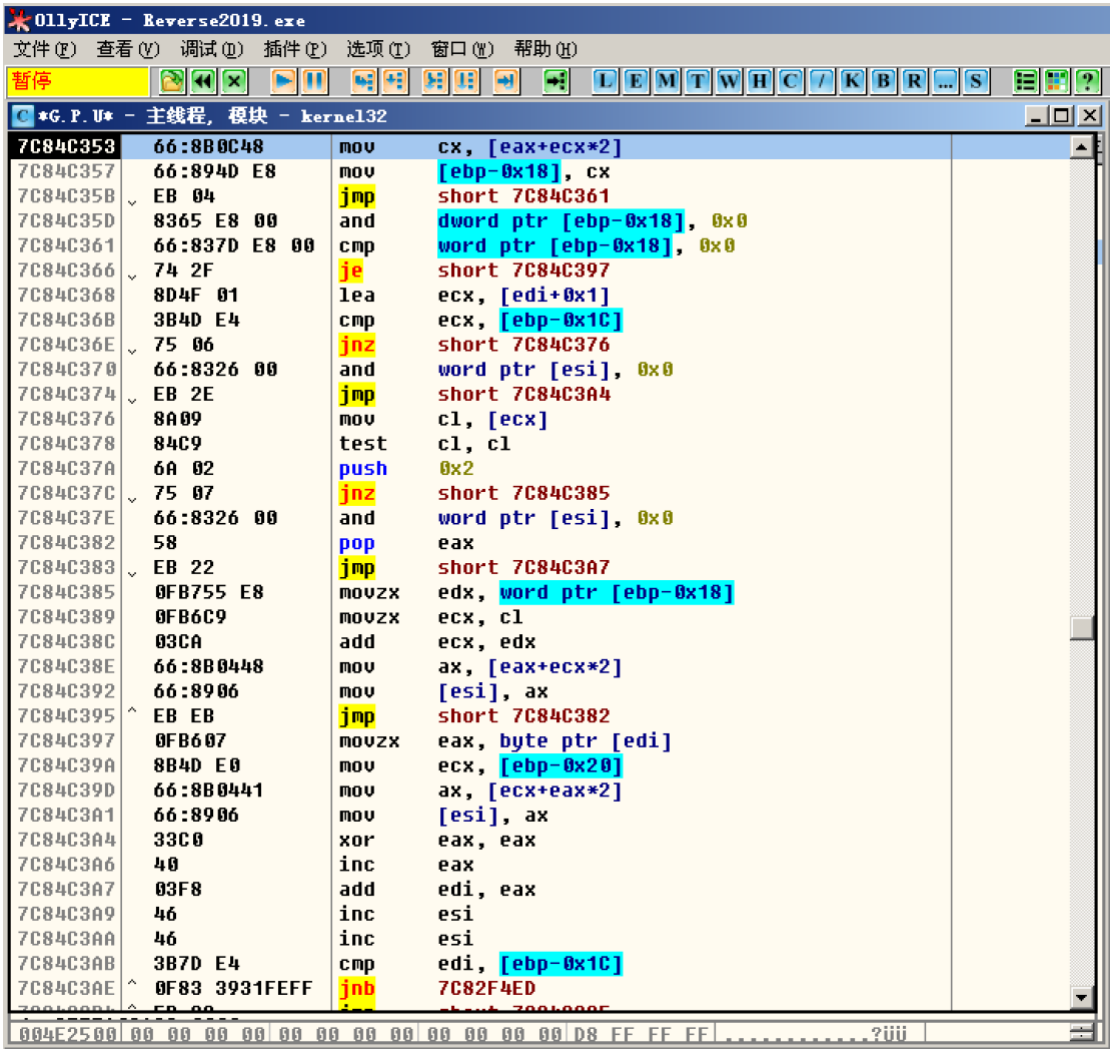
二、 实验过程

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。

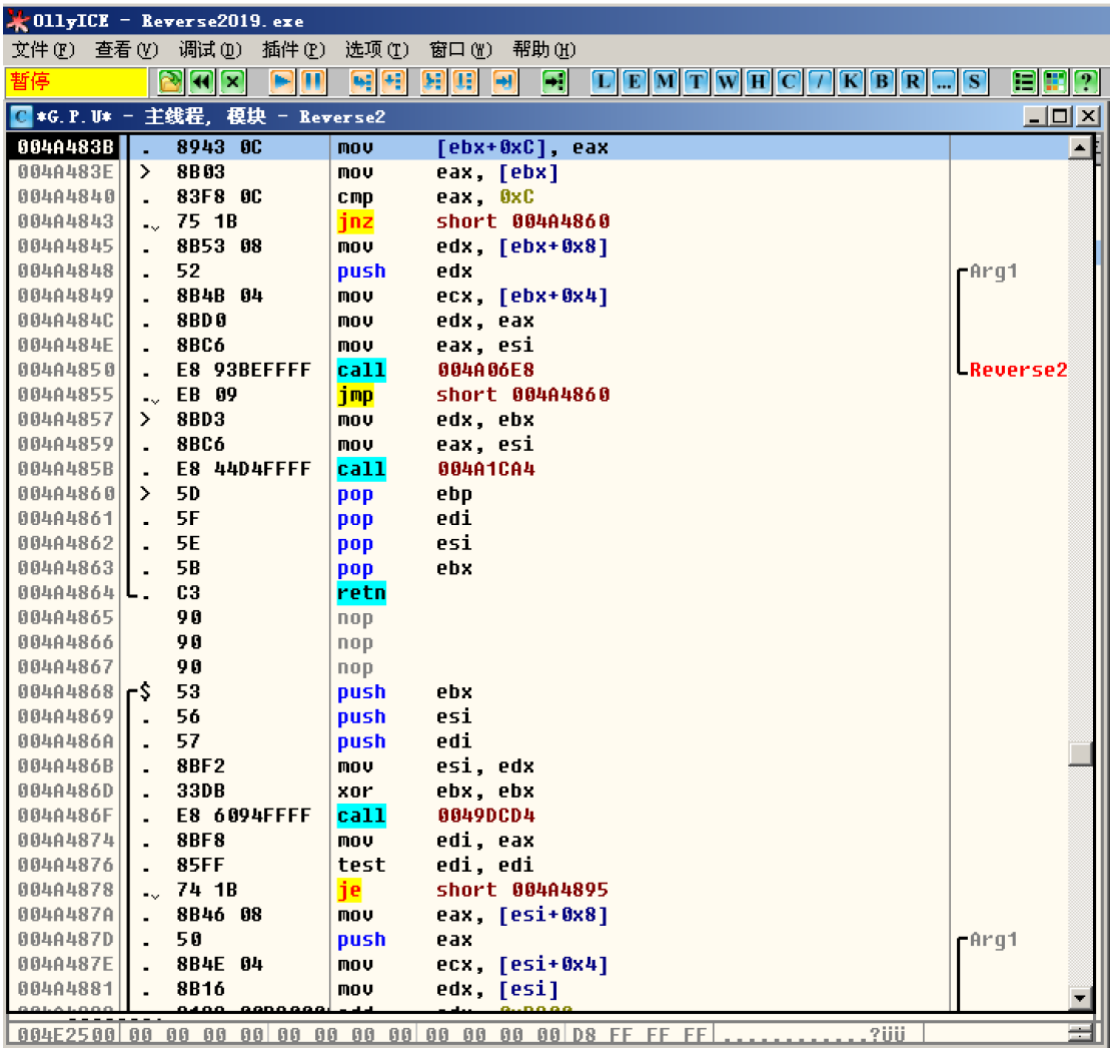




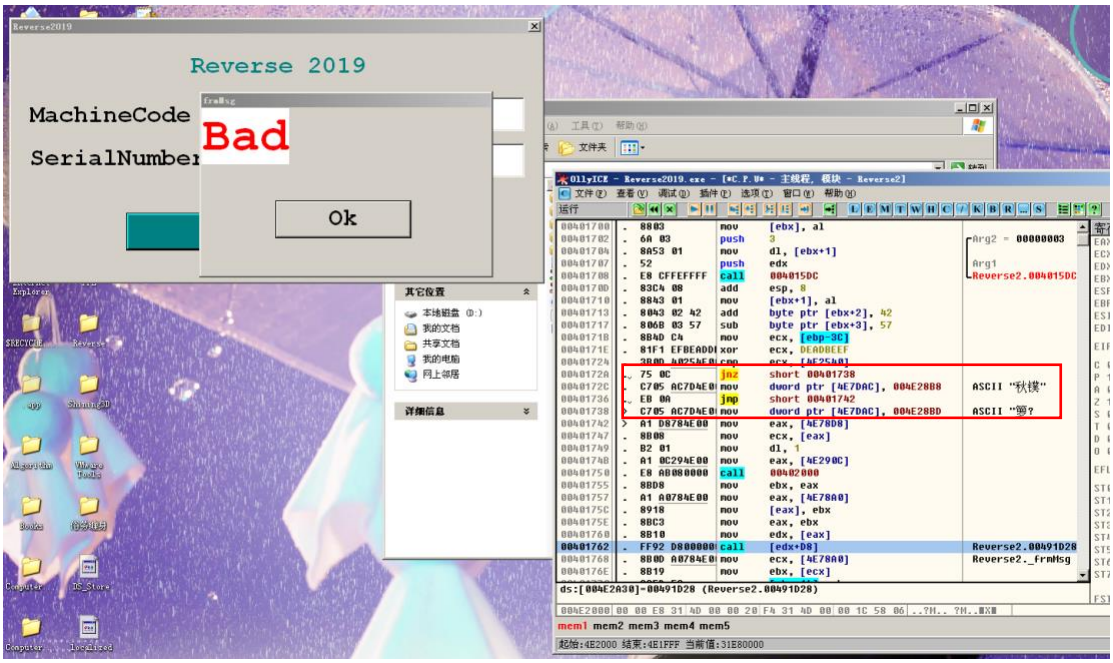
点击 Register 之后，断点生效，我们取消硬件断电，然后开始跟踪。



进入用户态之后继续跟踪，直到弹出窗口。



我们发现在这一步会弹出 Bad 窗口，如下图所示。

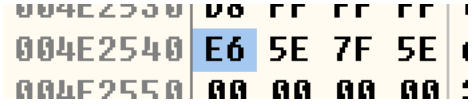


查看弹出窗口之前的代码，发现红色方框内比较可疑，可能与弹出窗口之前的判断有关，在这加个断点尝试一下吧。

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。

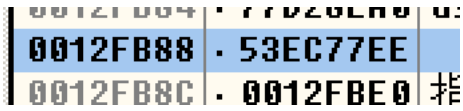
继续运行发现会弹出 Good 窗口，那么我们如果将 jnz short 00401738 改为 nop 并保存，就使用爆破法破解了这个程序，但为完成本次作业要求，我们继续探究，找出注册码的算法。

从这个判断开始逆向思考是比较自然的做法，注意到 Good 的条件等价于 ecx 和[0x4E2540]的值是相同的，从数据窗口发现那里正保存着 Machine Code。

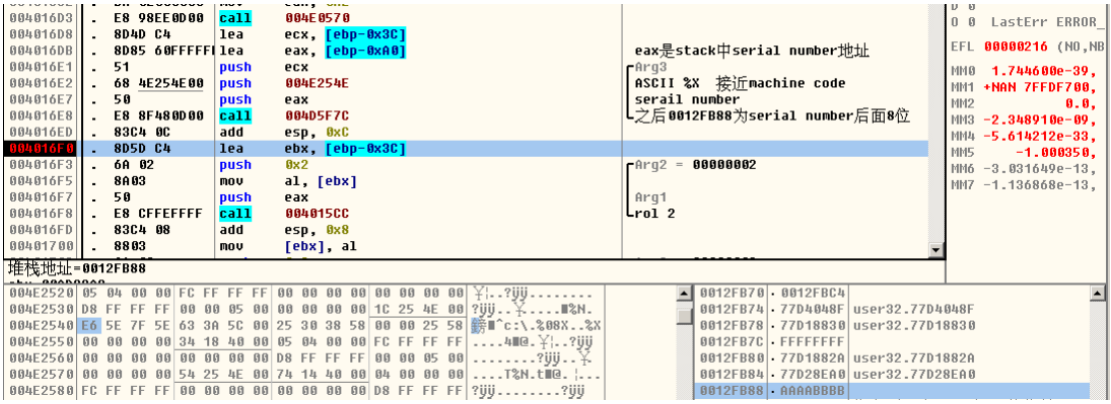


而之前一步是 xor ecx, 0xDEADBEEF，也就是说 ecx 的值在 xor 后要等于 Machine Code。

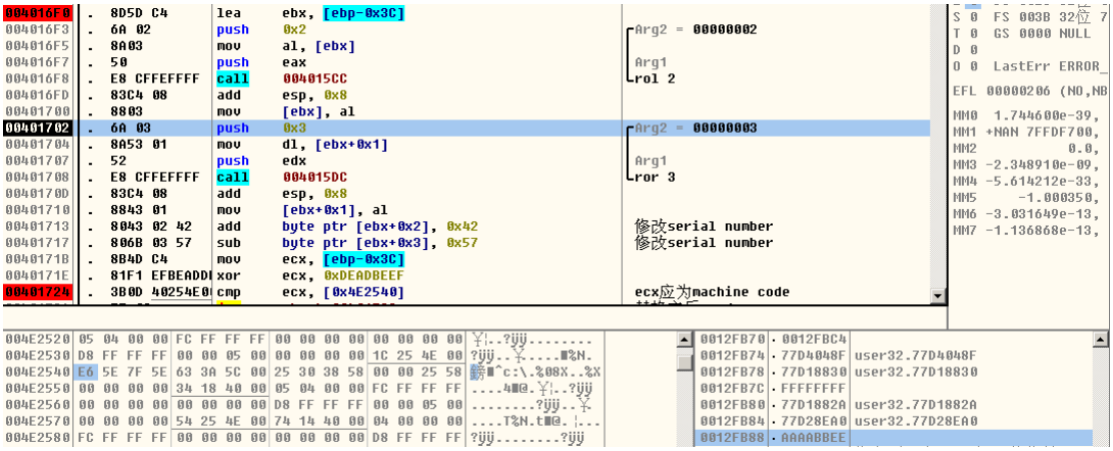
而 ecx 来自[ebp-0x3C],再重新追踪一次，发现这个地址为 0012FB88，但其中的值意义还不明，我们就重点关注这个地址即可。



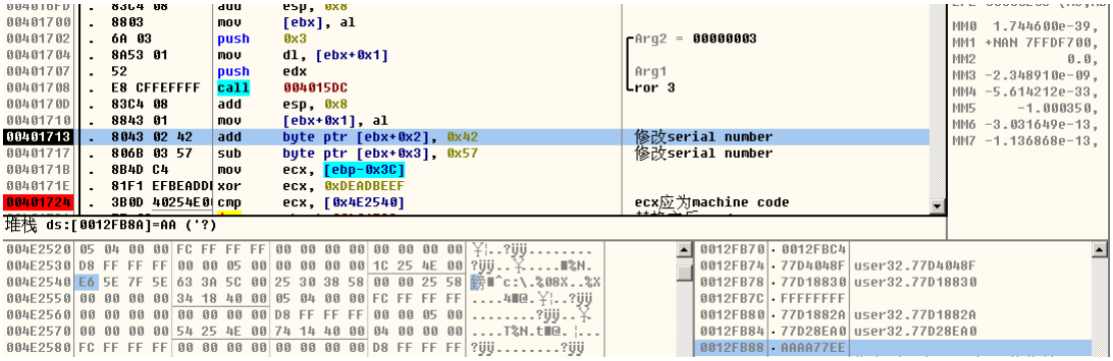
再次追踪，这次在更早的地方断住，发现在此之后该地址的值为 AAAABBBB，即我们输入的序列号。



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。



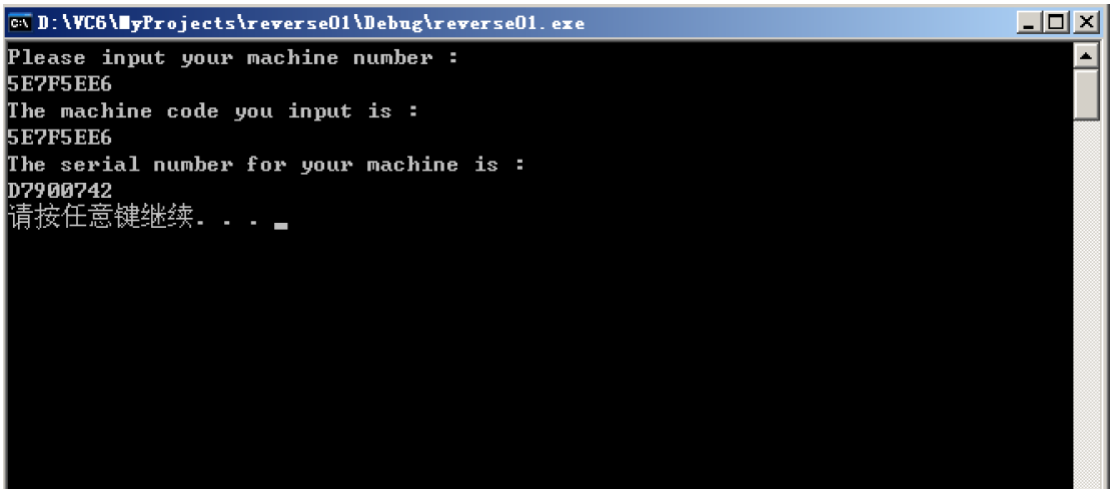
然后对倒数第 2 个字节 ror3bits。



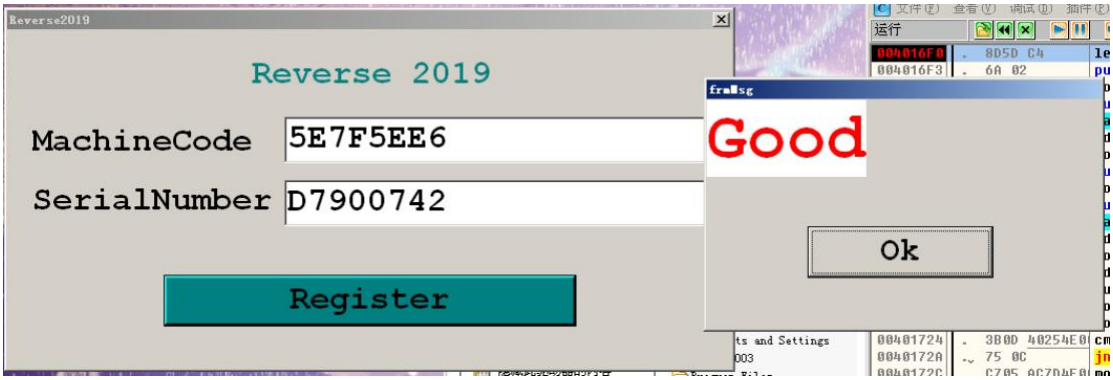
接下来对第 1byte 分别+0x42 和第 1byte-0x57，然后和那个 magic number 作 xor，就来到了我们之前说的判断处。

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。

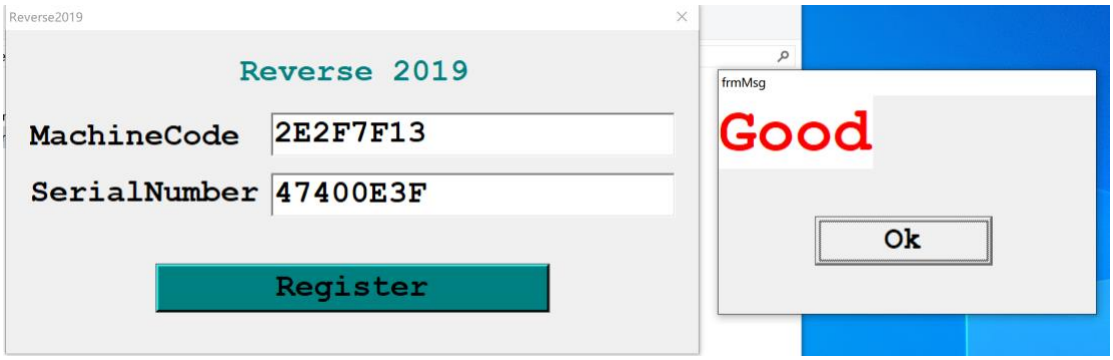
对 Machine Code 先与 magic number xor，然后第 1byte-0x42，第 2byte+0x57，第 3byte rol 3bits，第 4byte ror 2bits，即可得到注册码，编写注册机程序，然后验证。



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 }
```