

# 看雪CTF.TSRC 2018 团队赛 第一题 初世纪 writeup

原创

xuenixiang 于 2018-12-03 20:22:11 发布 486 收藏 1

分类专栏: CTF 文章标签: CTF 看雪CTF.TSRC 2018 团队赛 第一题 初世纪 wr 初世纪 TSRC

版权声明: 本文为博主原创文章, 遵循CC 4.0 BY-SA 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/qq\\_41917908/article/details/84778692](https://blog.csdn.net/qq_41917908/article/details/84778692)

版权



[CTF 专栏收录该内容](#)

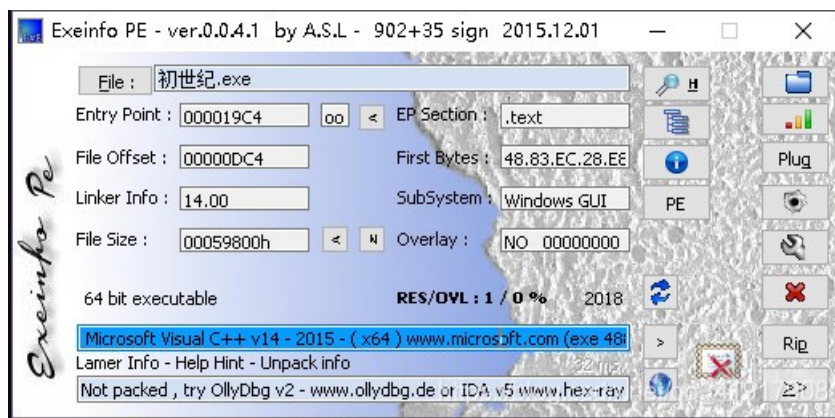
2 篇文章 0 订阅

订阅专栏

这个 writeup我花了1个多小时才写出来, 我感觉这已经不是 writeup了, 这简直就是解题图文教程!!

每个人都是从新手过来的, 几年前我初学逆向的时候, 看到大牛写的 writeup, 一头雾水, 根本看不懂, 所以我写这么详细是想让更多初学逆向的朋友能跟学会这门技术, 感受她的魅力……

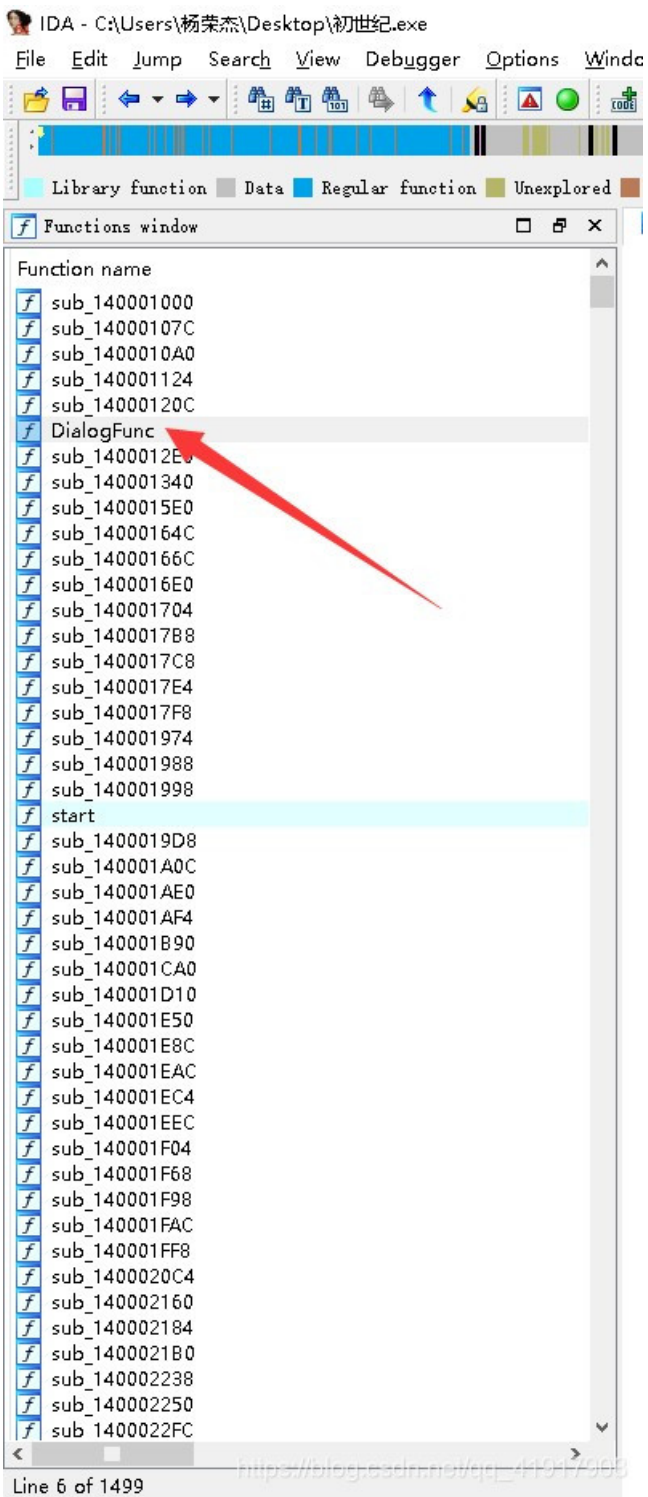
首先用PEID看一下有没有壳, 发现没有壳, 而且是个64位程序, 这时候就可以排除用OD分析的可能了(很难过, 最擅长的就是OD了),



因为CTF的题目一般不会有病毒后门什么的, 所以直接在物理机上或者64位虚拟机里运行一下, 猜猜思路



我输入序列号, 发现会提示错误, 但是我在IDA中搜索这个字符串, 是什么都找不到的, 这说明这个字符串可能是个图片, emmm·但是我现在有2个思路了, 可能会用到GetDlgItemTextA这个函数来获取字符串, 还可能有MessageBoxA这个函数来弹窗。所以直接导入IDA来看看



看到这个函数，果断跟进去看看，这里交叉引用走到调用处



Windows Help

Instruction External symbol

IDA View-A Hex View-1 Structures Enums Imports

```

.text:000000014000145E
.text:000000014000145E loc_14000145E: ; CODE XREF: sub_140001340+DD↑j
.text:000000014000145E mov r9d, 51h ; cchMax
.text:0000000140001464 lea r8, [rbp+30h+String] ; lpString
.text:0000000140001468 mov edx, 3E8h ; nIDDlgItem
.text:000000014000146D mov rcx, rdi ; hDlg
.text:0000000140001470 call cs:GetDlgItemTextA
.text:0000000140001476 mov r9d, 65h ; cchMax
.text:000000014000147C lea r8, [rsp+130h+var_D0] ; lpString
.text:0000000140001481 mov edx, 3E8h ; nIDDlgItem
.text:0000000140001486 mov rcx, rdi ; hDlg
.text:0000000140001489 mov ebx, eax
.text:000000014000148B call cs:GetDlgItemTextA
.text:0000000140001491 cmp ebx, 6
.text:0000000140001494 jnz short loc_1400014F1
.text:0000000140001496 movzx eax, [rsp+130h+var_D0]
.text:000000014000149B sub eax, 30h
.text:000000014000149E cmp eax, ebx
.text:00000001400014A0 jnz short loc_1400014F1
.text:00000001400014A2 movzx eax, [rsp+130h+var_CF]
.text:00000001400014A7 sub eax, 40h
.text:00000001400014AA cmp eax, 5
.text:00000001400014AD jnz short loc_1400014F1
.text:00000001400014AF movzx eax, [rsp+130h+var_CE]
.text:00000001400014B4 sub eax, 70h
.text:00000001400014B7 cmp eax, 7
.text:00000001400014BA jnz short loc_1400014F1
.text:00000001400014BC movzx eax, [rsp+130h+var_CD]
.text:00000001400014C1 sub eax, 60h
.text:00000001400014C4 cmp eax, 9
.text:00000001400014C7 jnz short loc_1400014F1
.text:00000001400014C9 movzx eax, [rsp+130h+var_CC]
.text:00000001400014CE sub eax, 30h
.text:00000001400014D1 cmp eax, 9
.text:00000001400014D4 jnz short loc_1400014F1
.text:00000001400014D6 movzx eax, [rsp+130h+var_CB]
.text:00000001400014DB lea rcx, String1
.text:00000001400014E2 sub eax, 40h
.text:00000001400014E5 cmp eax, 8
.text:00000001400014E8 jnz short loc_1400014F8
.text:00000001400014EA lea rdx, [rsp+130h+var_E8]
.text:00000001400014EF jmp short loc_1400014FD
.text:00000001400014F1 ;
.text:00000001400014F1 loc_1400014F1: ; CODE XREF: sub_140001340+154↑j
.text:00000001400014F1 ; sub_140001340+160↑j ...
.text:00000001400014F1 lea rcx, String1 ; lpString1

```

000008B4 00000001400014B4: sub\_140001340+174 (Synchronized with Hex View-1)

看来我猜中了，的确是这个函数，这里可以大致的看到算法了，GetDlgItemTextA获取到文本之后，首先判断了一下我们输入的字符串的位数是不是6位，如果不是就直接报错了，不会往下判断。我们按F5转换为伪代码看

```

73 v8 = GetDlgItemTextA(v5, 1000, &String, 81);
74 GetDlgItemTextA(v5, 1000, &v23, 101);
75 if ( v8 != 6 || v23 != 54 || v24 != 69 || v25 != 119 || v26 != 105 || v27 != 57 )
76 {
77     v9 = (CHAR *)&String1;
78 }
79 else
80 {
81     v9 = (CHAR *)&String1;
82     if ( v28 == 72 )
83     {
84         v10 = (CHAR *)&v19;
85 LABEL_19:
86     IstrncpyA(v9, v10);
87     DialogBoxParamA(hInstance, (LPCSTR)0x79, v5, sub_1400012E0, 0i64);
88     return sub_1400016E0((unsigned __int64)&v14 ^ v30);
89 }
90 }
91 v10 = String2;
92 goto LABEL_19;
93 }
94 return sub_1400016E0((unsigned __int64)&v14 ^ v30);
95 }

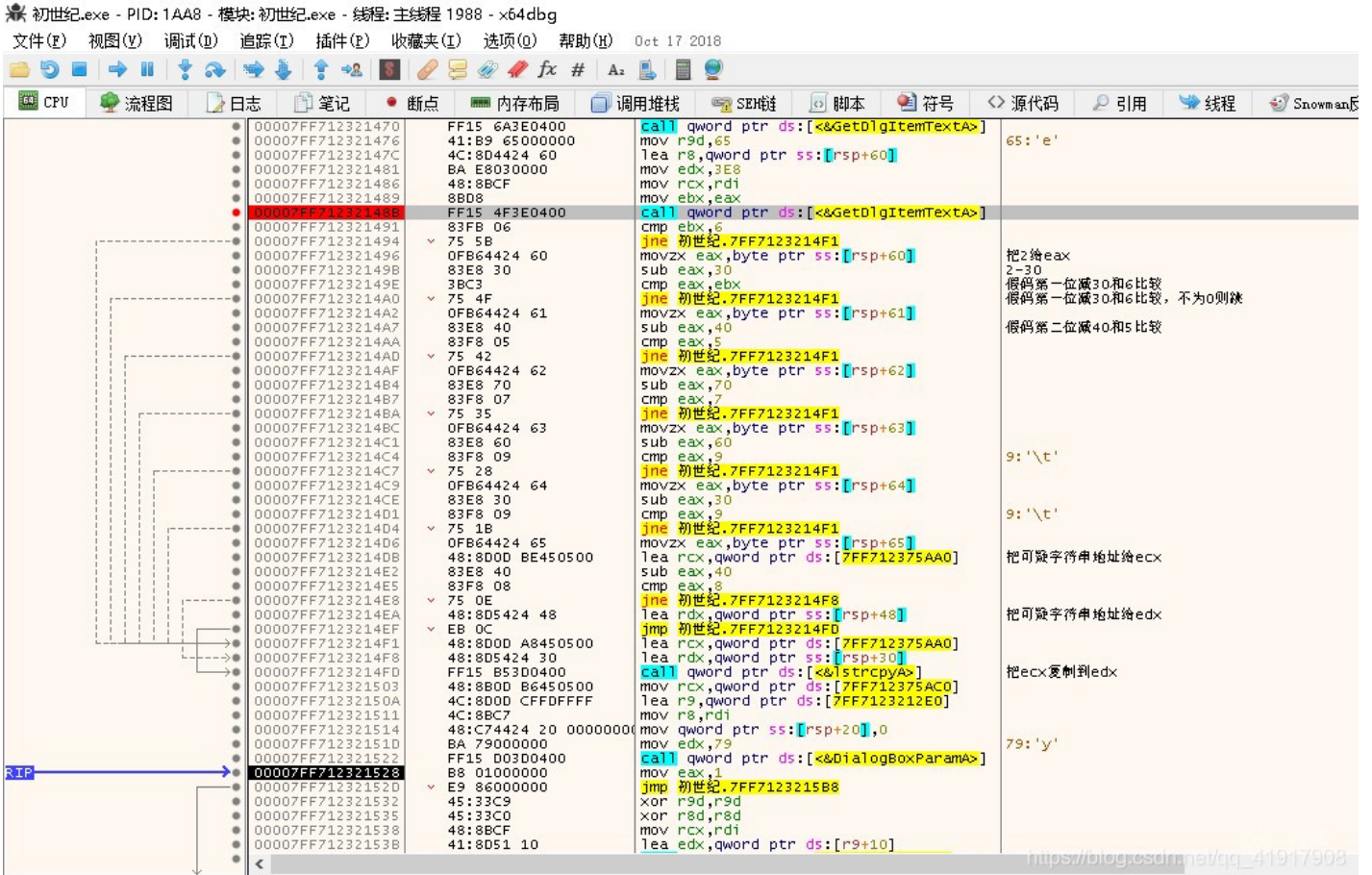
```



如果是逆向大佬，到这里就基本能看出flag了，但是作为小白，还是需要动态调试验证一下自己的思路的

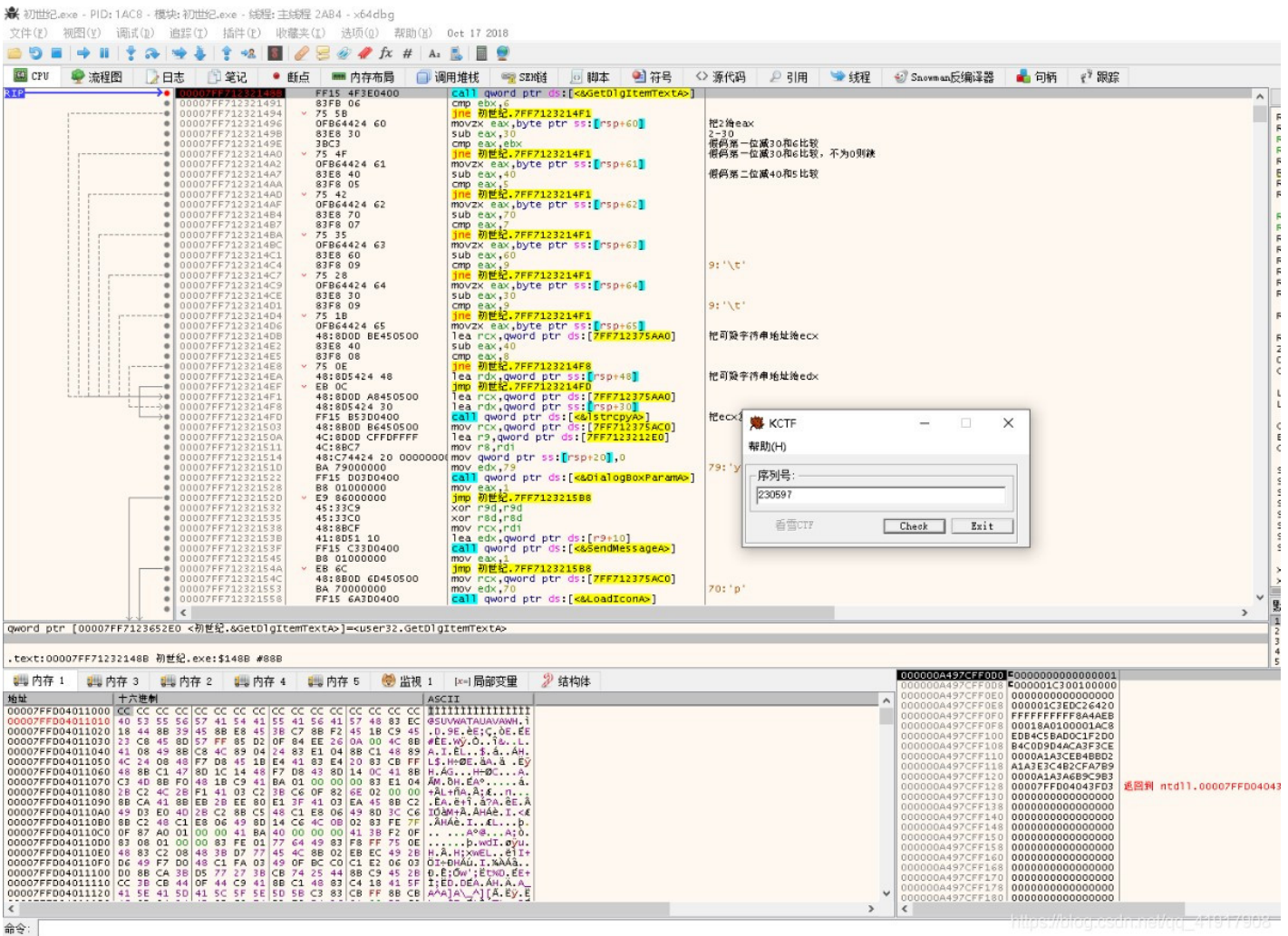
因为OD不能用，所以还剩X64Debug和MDebug，MDebug太高端……我不太会用，但是 X64Debug和OD还是长得挺像的，新手入门也快，这里就选 X64Debug了

首先载入创世纪.exe，我们有很多种方法来找到算法处，第一个可以考虑用消息断点来到达关键处，但是我们前面用IDA分析出了GetDlgItemTextA这个函数，所以就直接ctrl+G到达定位到下图位置就可以了。

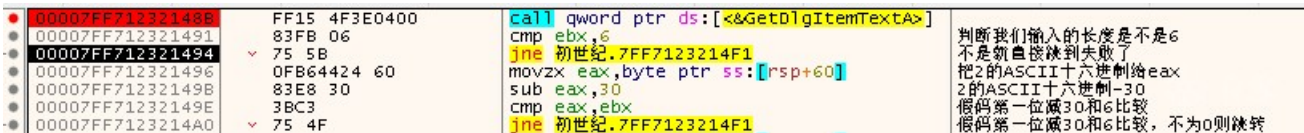


下面正式开始动态分析

我们在 GetDlgItemTextA函数处下断，运行程序，随便输入字符串230597，点击check，断在这里



然后继续向下F8单步走，先判断我们输入的是不是6位，然后把第一位数字2的ASCII提出来减去30（这里的30是十六进制，下面也一样）和6进行比较，不一样就调到失败，这里我们就可以推出flag的第一位是30+6=36（十六进制），因为比较结果不为0，jne是会跳转的，我们把ZF标志位改为1，让它不跳

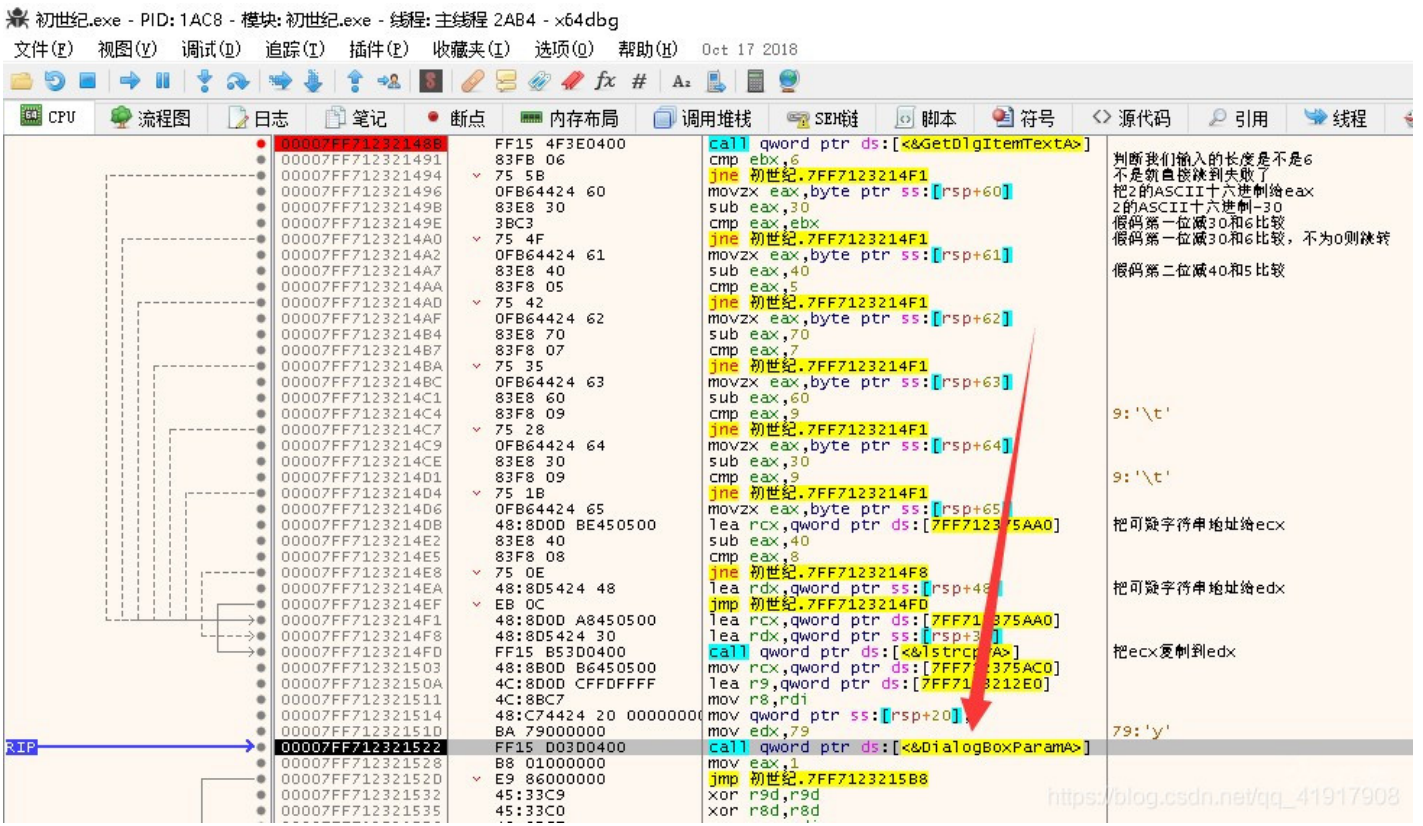


继续向下分析，可以看出依次把第二、三、四、五、六位进相似的判断，我们依次逆向相加，推出第2-6位的ASCII十六进制依次为45 77 69 39 48

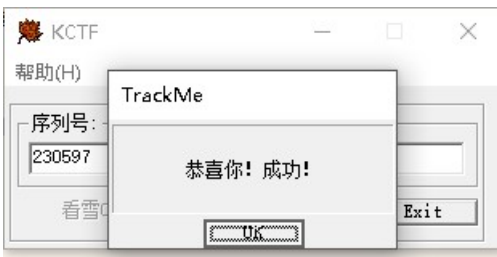


判断完成后，程序对提示成功的字符串缓冲区地址进行复制等操作，最后当做参数传给这DialogBoxParamA这个函数

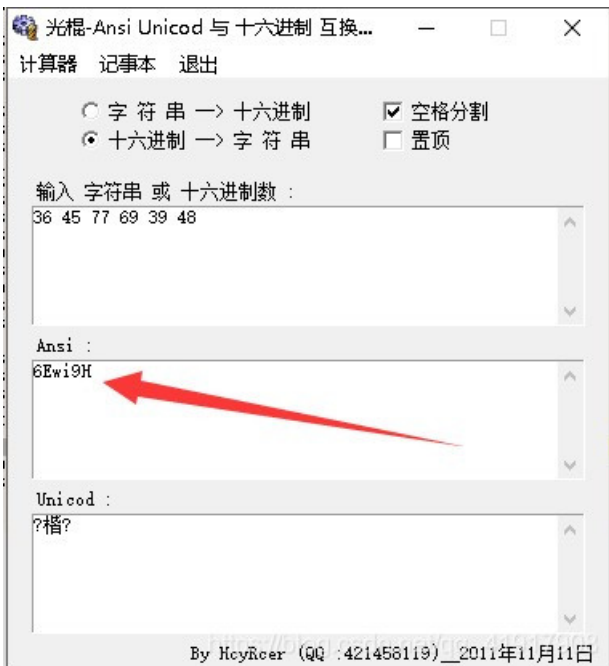




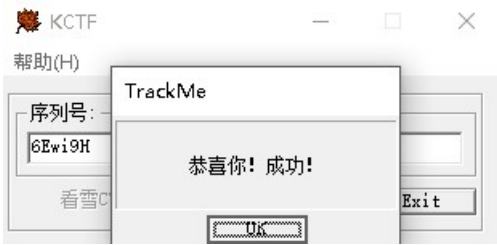
最后执行到这个call，CM弹出正确窗口



这时候我们就可以验证刚才的逆推结果了，我们把36 45 77 69 39 48转换为ASCII就是flag了



完全正确



小白的 writeup, 若有错误欢迎指正!!!