# HCTF 2017 bin Level1 Evr_Q Writeup

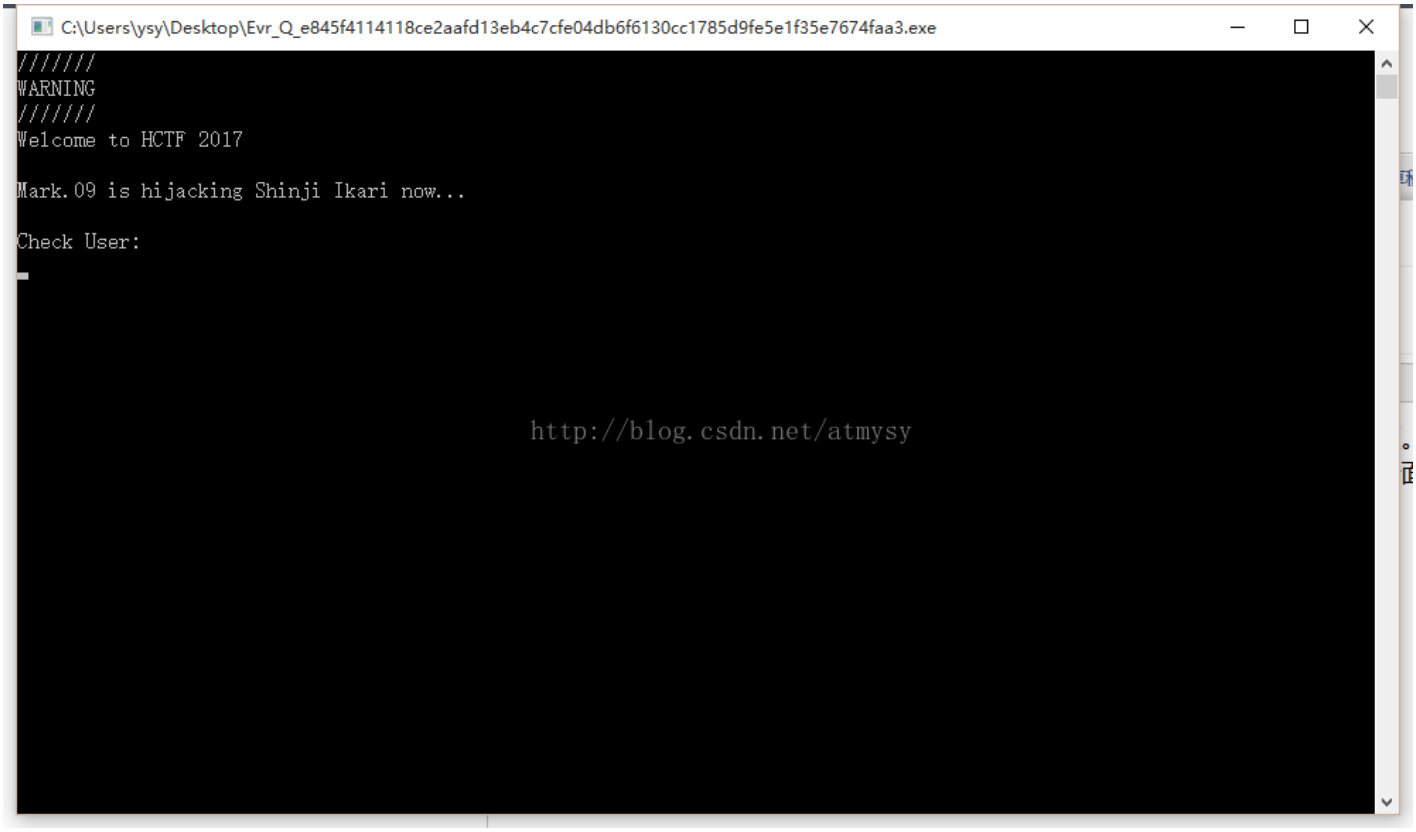
原创

inori jam 于 2017-11-13 21:57:04 发布 605 收藏
分类专栏： Writeup 文章标签： CTF 逆向工程 动态调试 网络安全

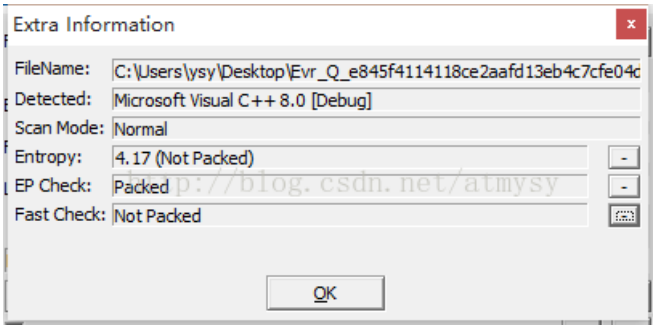


HCTF 2017完结撒花，萌新领教到了赛棍们的强大，但也有了一些自己的收获。我想将其中一些题的思路记录下来，一是记录，二是与有需要的人分享，当然本篇WP和大佬们的比还是很菜。话不多说，进入正题。

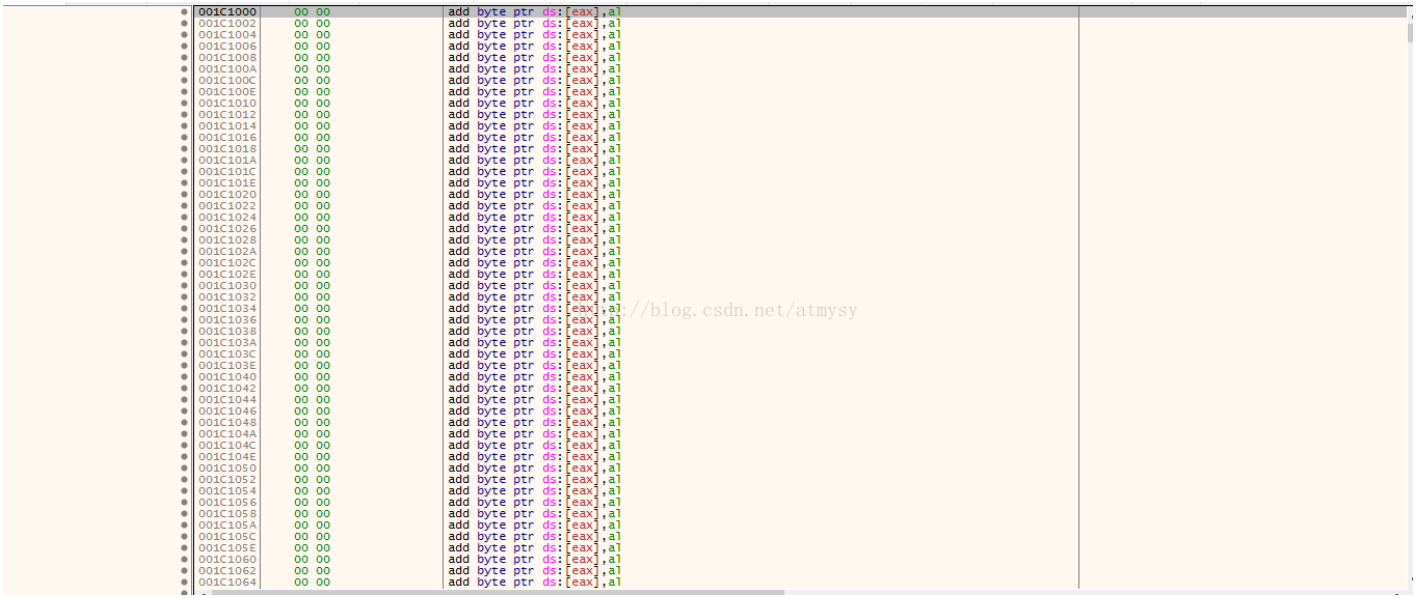
首先我们拿到一个看起来很吊的程序，如下图：



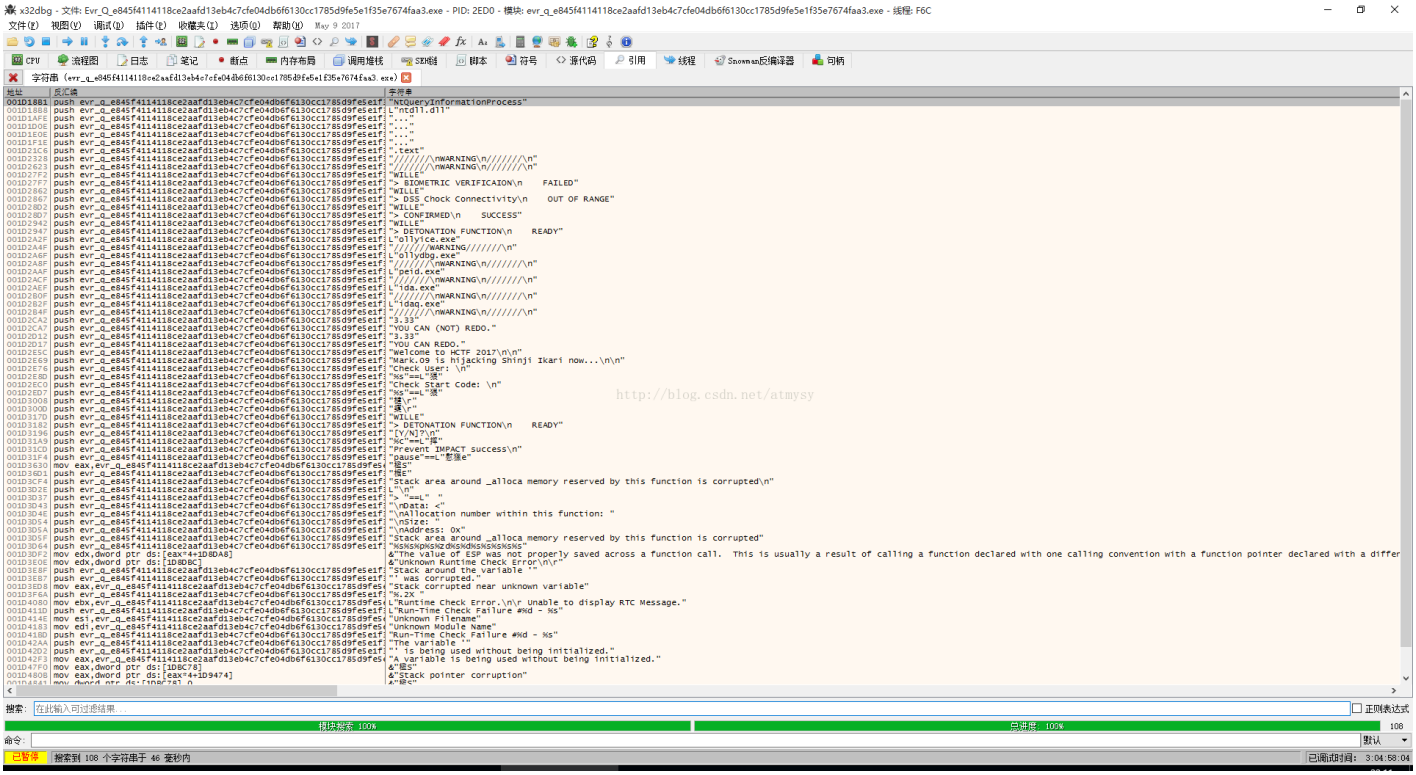

例行公事，PEID查壳，查到EP段有加壳，但是不影响我们Crack




直接上神器X32-DBG，考虑到壳的关系，附加到程序进程，来到主程序模块发现什么都没有

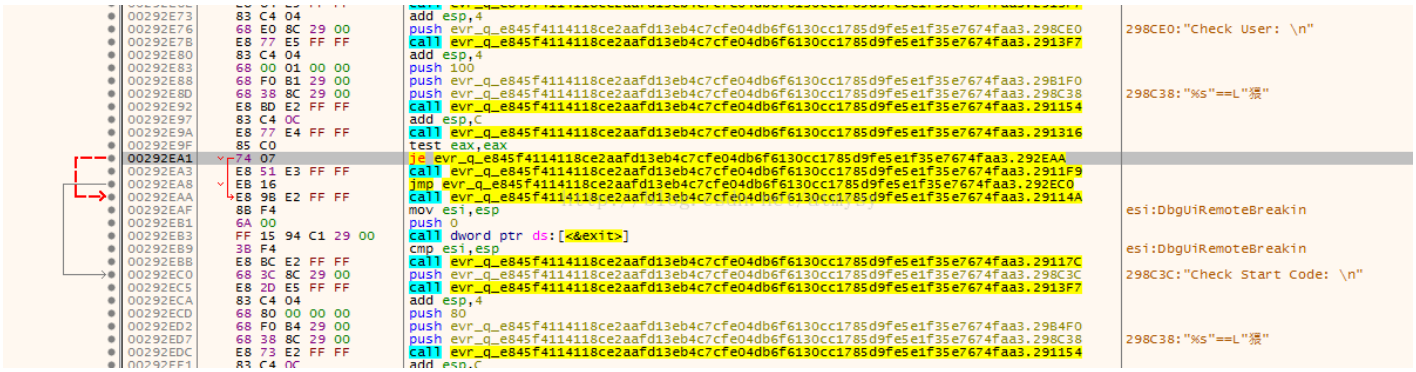



搜一下字符串发现有这么多

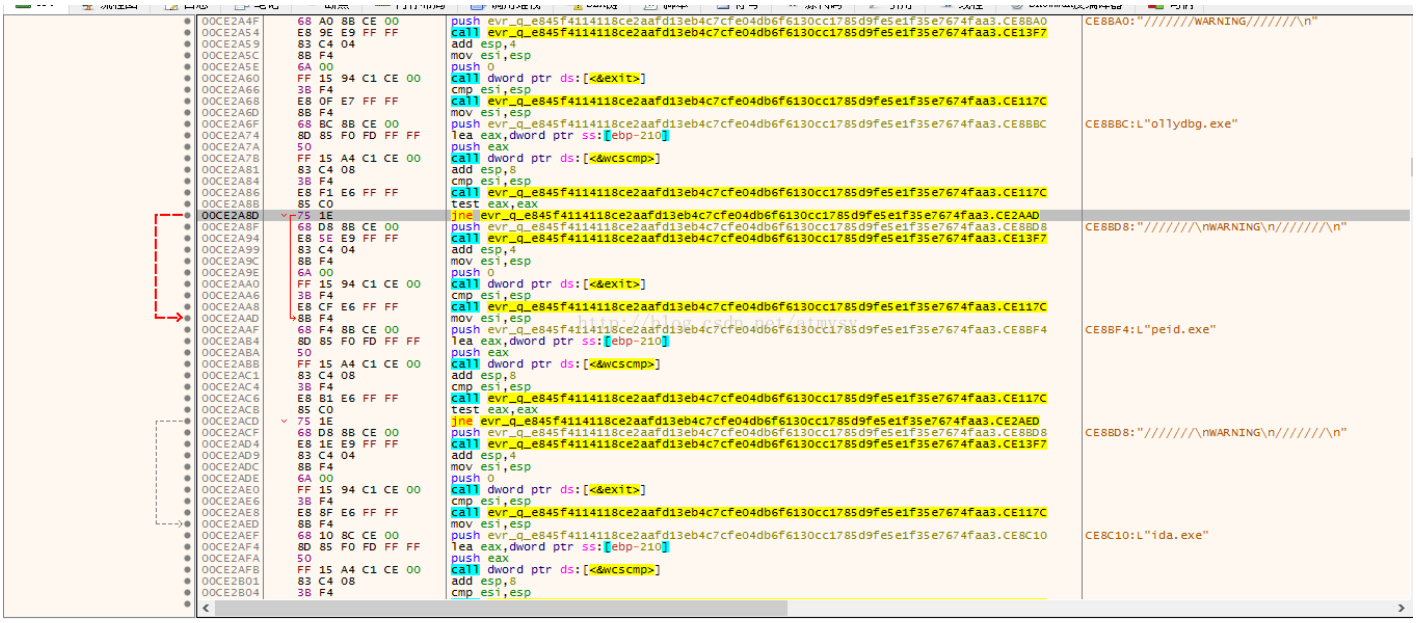



找到看起来比较关键的字符串点进去，我们的老朋友：汇编代码就回来了

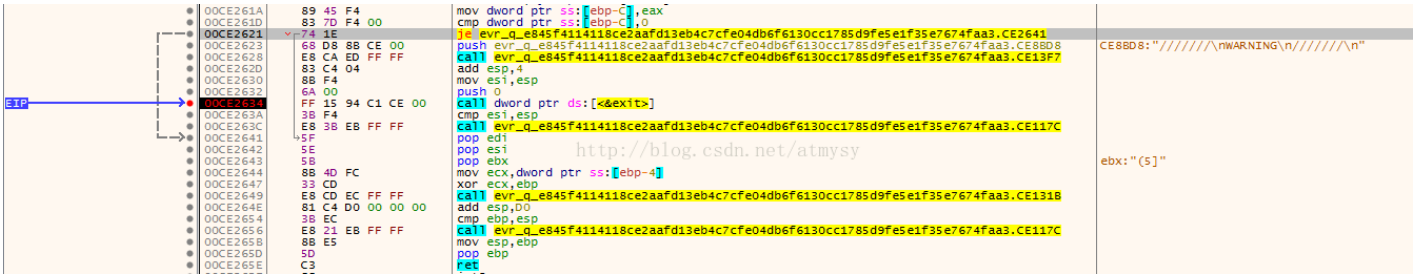
接下来可以查看check user后几个关键call，发现程序将用户名倒序，再对用户名中每个ASCII字符对应的十六进制作XOR，AND等一系列运算，最后与密文比较，根据这个，我们直接就能用十六进制计算器反算出正确的用户名原文为：M.KATSURAGI。实际上直接改跳转也能使验证成功，且不会对我们get flag造成任何影响，但稳妥起见，还是按流程走吧。



输入正确的用户名，程序弹框提示验证成功，但由于程序有反调试，所以在弹框后可能会自动退出，我们能轻易找到反调试的代码，在所有exit调用函数处下断,我们再运行



发现罪魁祸首原来不在刚才那里，而在此处被断下来，修改前面的je跳转,弹框后便不再退出



接下来进入重头戏，程序要求我们输入Start Code，按照常识，这应该就是我们要找的flag无疑，所以我们在check start code字符串后的getchar函数处下断点，然后随意输入一个start code，单步调试一遍后，知道了各个call的功能并做好注释



我们现在知道程序先检查输入长度是否为35，然后将输入的start code进行XOR运算，然后取第8位到第14位，第15位到21位，第22位到28位三组7位字符串，这三组字符串分别经过方法相同但参数不同的运算，用结果替换掉原输入对应位置的字符串，最后与密文比较。到这里，我们可以根据程序中所给参数建立方程，方程的解即为正确的原文，但方程比较复杂，这里采用穷举来解方程，我们都知道，二位十六进制数范围为00到FF，转换为十进制即0到255，于是写出脚本如下：

```
def solve(idx1,idx2,idx3,idx4):
    for x in range(0,255):
        if(((x ^ idx1 )& idx2 )>> idx3) | (((x ^ idx1) << idx3 )& idx2) == idx4:
            print hex(x^0x76)
key1 = [0x6f,0xdd,0xdd,0x48,0x64,0x63,0xd7]
for ele in key1:
    solve(0xAD,0xAA,0x1,ele)
key2 = [0x2e,0x2c,0xfe,0x6a,0x6d,0x2a,0xf2]
for ele in key2:
    solve(0xbe,0xcc,0x2,ele)
key3=[0x6f,0x9a,0x4d,0x8b,0x4b,0xfa,0x1a]
for ele in key3:
    solve(0xef,0xf0,0x4,ele)
```

穷举出3组7位原文，再把35位原文经XOR得到的前七位和后七位与之前XOR运算的参数再作一次XOR运算（我们都知道，XOR运算是可逆的），最后将所有结果拼接起来放到在线16进制转ASCII码的网站，转换为ASCII码，get flag！为：hctf{>>D55_CH0CK3R_B0o0M!-68271c0a}

今年hctf对bin选手不友好是真的，ORZ。总之还得多跟大佬们学学，嗯，就这样。