

# 2021年春秋杯网络安全联赛秋季赛逆向snake.exe Writeup

原创

bin cat 于 2021-11-28 20:44:17 发布 9894 收藏 1

文章标签: 安全

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) 版权协议, 转载请附上原文出处链接和本声明。

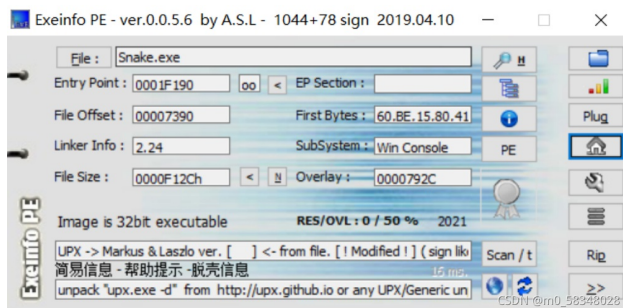
本文链接: [https://blog.csdn.net/m0\\_58348028/article/details/121589636](https://blog.csdn.net/m0_58348028/article/details/121589636)

版权

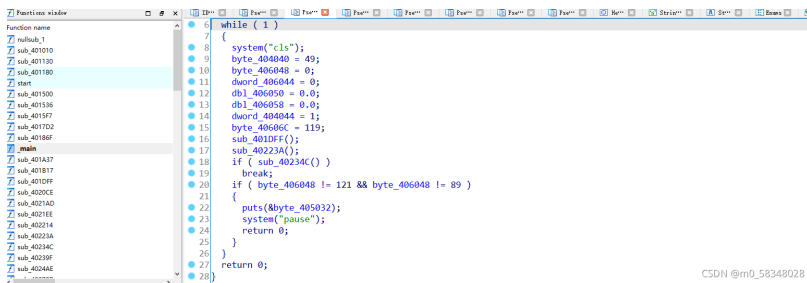


下载附件snake.exe, 看题目描述是一个游戏那就打开玩玩, 发现这是一个贪吃蛇游戏, 游戏大概是通过玩贪吃蛇打到一定的分数之后就会有一个flag校验的窗口, 所以基本可以判断出不出flag和那个贪吃蛇游戏基本没关系, 可以跳过游戏直接去找flag校验的过程

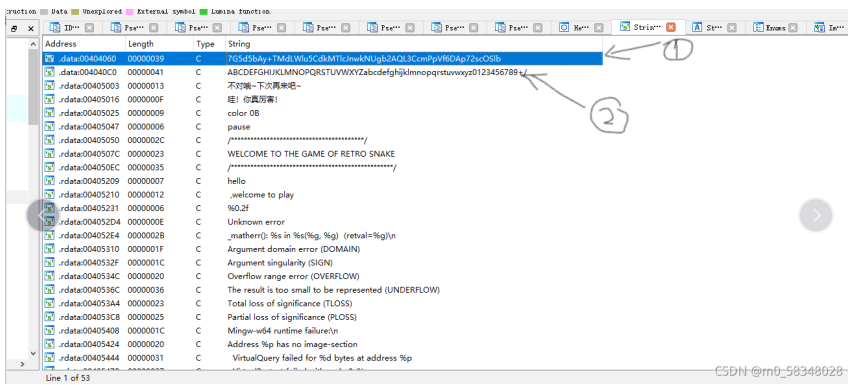
下面动手, 先查壳



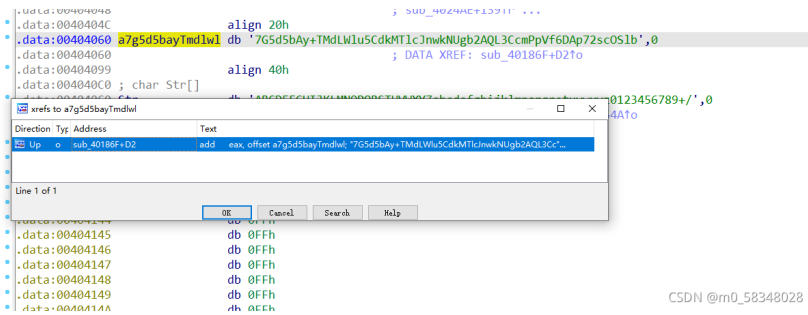
一个32位的程序, 有一个UPX壳, 把壳脱掉扔进ida pro找到main函数



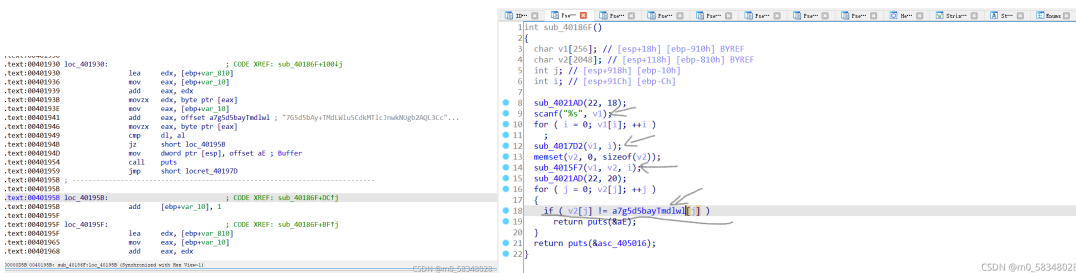
猜测这里是一个对贪吃蛇游戏的描述, 我们可以跳过这个去找找字符串窗口看看能不能找到什么



字符串窗口找到1号可疑字符，2号字符猜测是一个base64表后期有没有用目前未知，先跟1号进去

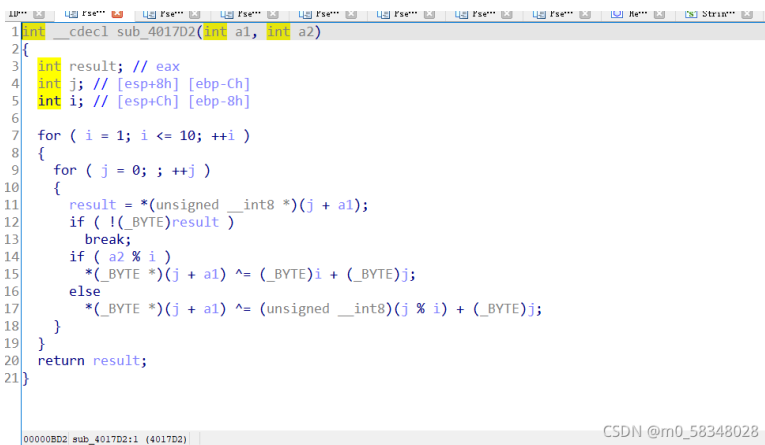


找到引用它的函数继续跟函数



这逻辑就出来了先是输入v1在经过两次处理之后和我们这个可疑字符串进行了对比，显而易见v1即为我们要找的flag，所以我们只要逆推就可以了找到两次处理

这是第一次



这是第二次

```

12 v9 = (16 * *(_BYTE *)(v14 + a1)) & 0x30;
13 if ( v14 + 1 >= a3 )
14 {
15     *(_BYTE *)(a2 + v12) = Str[v9];
16     *(_BYTE *)(v12 + 1 + a2) = 61;
17     v4 = v12 + 2;
18     v11 = v12 + 3;
19     *(_BYTE *)(v4 + a2) = 61;
20     break;
21 }
22 v5 = v12;
23 v13 = v12 + 1;
24 *(_BYTE *)(a2 + v5) = Str[((int)* (unsigned __int8 *) (v14 + 1 + a1) >> 4) & 0xF | v9];
25 v10 = (4 * *(_BYTE *)(v14 + 1 + a1)) & 0x3C;
26 if ( v14 + 2 >= a3 )
27 {
28     *(_BYTE *)(a2 + v13) = Str[v10];
29     v6 = v13 + 1;
30     v11 = v13 + 2;
31     *(_BYTE *)(v6 + a2) = 61;
32     break;
33 }
34 *(_BYTE *)(a2 + v13) = Str[((int)* (unsigned __int8 *) (v14 + 2 + a1) >> 6) & 3 | v10];

```

观察第二次处理方式不难发现这是一个base64加密对应上了我们一开始在字符串窗口看到的那个base64表，返回字符串窗口看看base64表有没有被改

Direction	Typ	Address	Text
Up	o	sub_401536:loc_40154A	mov dword ptr [esp], offset Str;"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	o	sub_401536+38	add eax, offset Str;"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_401536+46	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	o	sub_401536+54	add eax, offset Str;"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_401536+65	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	o	sub_401536+6F	add edx, offset Str;"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	w	sub_401536+81	mov byte ptr Str[edx], al; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_4015F7+45	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_4015F7+81	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_4015F7+EB	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_4015F7+12A	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_4015F7+181	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0
Up	r	sub_4015F7+182	movzx eax, byte ptr Str[eax]; "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'",0

发现base64表有一个写操作，跟进去就发现表被改了

```

1 int sub_401536()
2 {
3     int result; // eax
4     char v1; // [esp+13h] [ebp-15h]
5     signed int v2; // [esp+14h] [ebp-14h]
6     int j; // [esp+18h] [ebp-10h]
7     int i; // [esp+1Ch] [ebp-Ch]
8
9     result = dword_406060;
10    if ( !dword_406060 )
11    {
12        v2 = strlen(Str);
13        for ( i = 0; v2 / 2 > i; ++i )
14        {
15            for ( j = 0; v2 - i - 1 > j; ++j )
16            {
17                if ( Str[j] > Str[j + 1] )
18                {
19                    v1 = Str[j];
20                    Str[j] = Str[j + 1];
21                    Str[j + 1] = v1;
22                }
23            }
24        }
25    }

```

外面的都不重要，红色圈才是真正对base64原表下手，按照他的逻辑看看他把表变成什么样了

```

a='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
a=list(a)
ss=len(a)
i=0

while(ss / 2 > i):
    j=0
    while(ss - i - 1>j):
        if (a[j] > a[j + 1]):
            v1 = a[j]
            a[j] = a[j + 1]
            a[j + 1] = v1
        j+=1
    i+=1
aa=''
for xxxx in a:
    aa+=xxxx

print(aa)

```

发现表变成了这样ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789+/UVWXYZabcdefghijklmnopqrstuvwxyz

后面只需要直接写代码逆就ok了

```

import base64
xx = "7G5d5bAy+TMdLWlu5CdkMT1cJnwkNUgb2AQL3CcmPpVf6DAp72scOS1b"
string1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789+/UVWXYZabcdefghijklmnopqrstuvwxyz"
string2 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
print (base64.b64decode(xx.translate(str.maketrans(string1,string2))))
xxx=list(base64.b64decode(xx.translate(str.maketrans(string1,string2))))
for a in range(1, 11):
    for j in range(42):
        if len(xxx) % a:
            xxx[j] ^= (a + j)
        else:
            xxx[j] ^= ((j % a) + j)
        j += 1

for i in xxx:
    print(chr(i), end="")

```

出flag

flag{5e2200bc-f21a-5421-a90b-57dec19fe196}