# 2021年春秋杯网络安全联赛秋季赛逆向snake.exeWriteup

原创

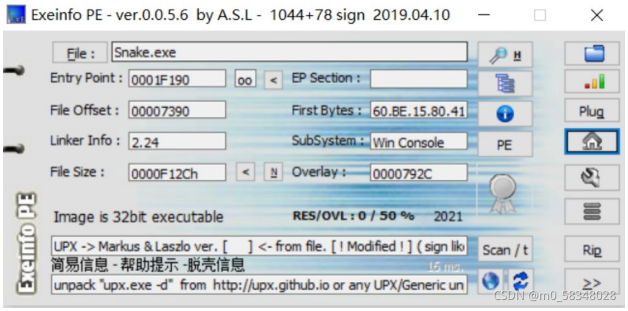
bin cat 于 2021-11-28 20:44:17 发布 9894 收藏 1
文章标签： 安全

本文链接：https://blog.csdn.net/m0_58348028/article/details/121589636

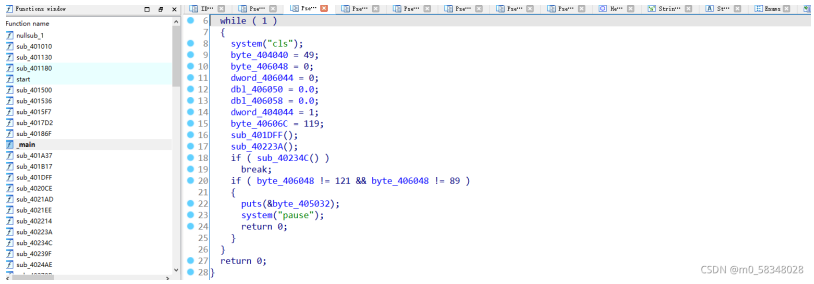


下载附件snake.exe，看题目描述是一个游戏那就打开玩玩，发现这是一个贪吃蛇游戏，游戏大概是通过玩贪吃蛇打到一定的分数之后就会有一个flag校验的窗口，所以基本可以判断出不出flag和那个贪吃蛇游戏基本没关系，可以跳过游戏直接去找flag校验的过程

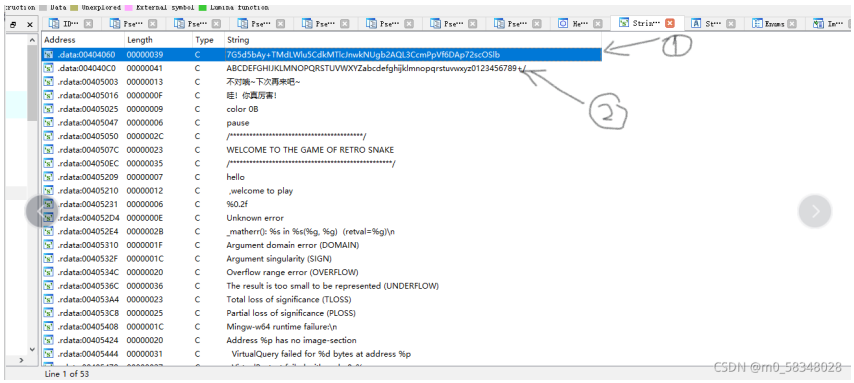
下面动手，先查壳



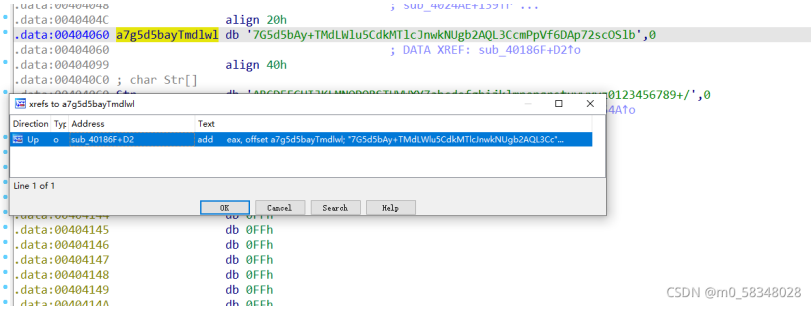

一个32位的程序，有一个UPX壳，把壳脱掉扔进ida pro找到main函数



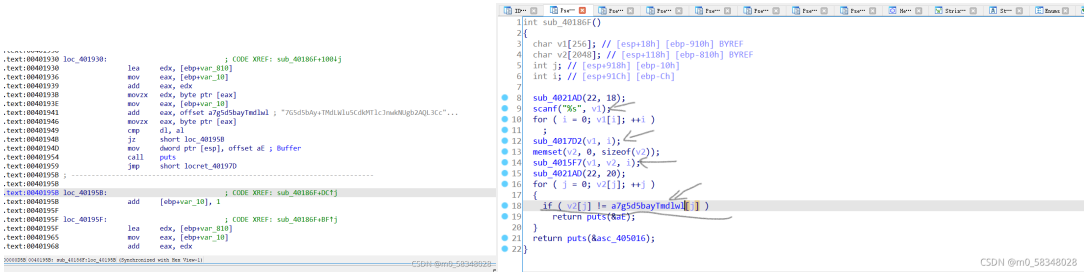

猜测这里是一个对贪吃蛇游戏的描述，我们可以跳过这个去找找字符串窗口看看能不能找到什么




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

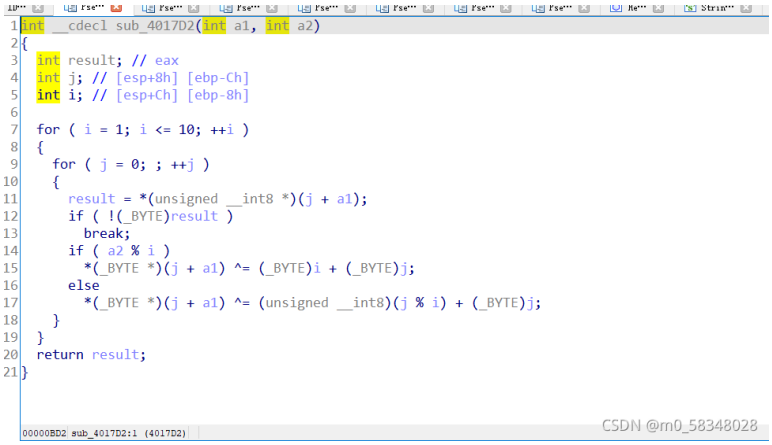



找到引用它的函数继续跟函数




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

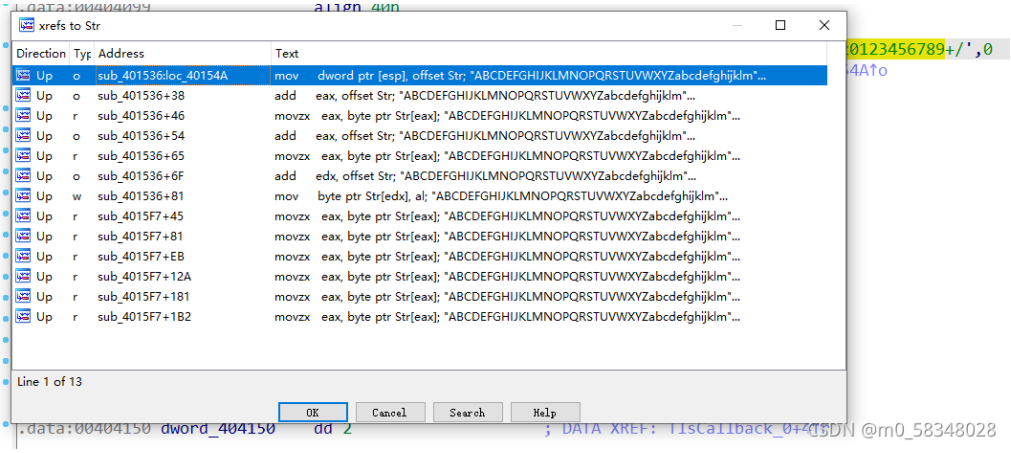



这是第二次

```
  v9 = (16 * *(_BYTE *)(v14 + a1)) & 0x30;
  if ( v14 + 1 >= a3 )
  {
    *(_BYTE *)(a2 + v12) = Str[v9];
    *(_BYTE *)(v12 + 1 + a2) = 61;
    v4 = v12 + 2;
    v11 = v12 + 3;
    *(_BYTE *)(v4 + a2) = 61;
    break;
  }
  v5 = v12;
  v13 = v12 + 1;
  *(_BYTE *)(a2 + v5) = Str[((int)*(unsigned __int8 *)(v14 + 1 + a1) >> 4) & 0xF | v9];
  v10 = (4 * *(_BYTE *)(v14 + 1 + a1)) & 0x3C;
  if ( v14 + 2 >= a3 )
  {
    *(_BYTE *)(a2 + v13) = Str[v10];
    v6 = v13 + 1;
    v11 = v13 + 2;
    *(_BYTE *)(v6 + a2) = 61;
    break;
  }
  *(_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






发现base64表有一个写操作，跟进去就发现表被改了

```
int sub_401536()
{
  int result; // eax
  char v1; // [esp+13h] [ebp-15h]
  signed int v2; // [esp+14h] [ebp-14h]
  int j; // [esp+18h] [ebp-10h]
  int i; // [esp+1Ch] [ebp-Ch]

  result = dword_406060;
  if ( !dword_406060 )
  {
    v2 = strlen(Str);
    for ( i = 0; v2 / 2 > i; ++i )
    {
      for ( j = 0; v2 - i - 1 > j; ++j )
      {
        if ( Str[j] > Str[j + 1] )
        {
          v1 = Str[j];
          Str[j] = Str[j + 1];
          Str[j + 1] = v1;
        }
      }
```



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ABCDEFGHIJKLMNOPQRST0123456789+/UVWXYZabcdefghijklmnopqrstuvwxyz

后面只需要直接写代码逆就ok了

```
import base64
xx = "7G5d5bAy+TMdLWlu5CdkMTlcJnwkNUgb2AQL3CcmPpVf6DAp72scOSlb"
string1 = "ABCDEFGHIJKLMNOPQRST0123456789+/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}