RSCTF一些WP（完全小白的答案）

原创

Tttttimer? 于 2019-10-31 21:33:02 发布 454 收藏 1

文章标签： CTF

本文链接：https://blog.csdn.net/Ttttttimer/article/details/102846119

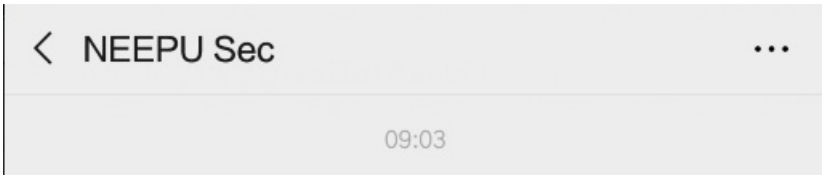
1.签到题

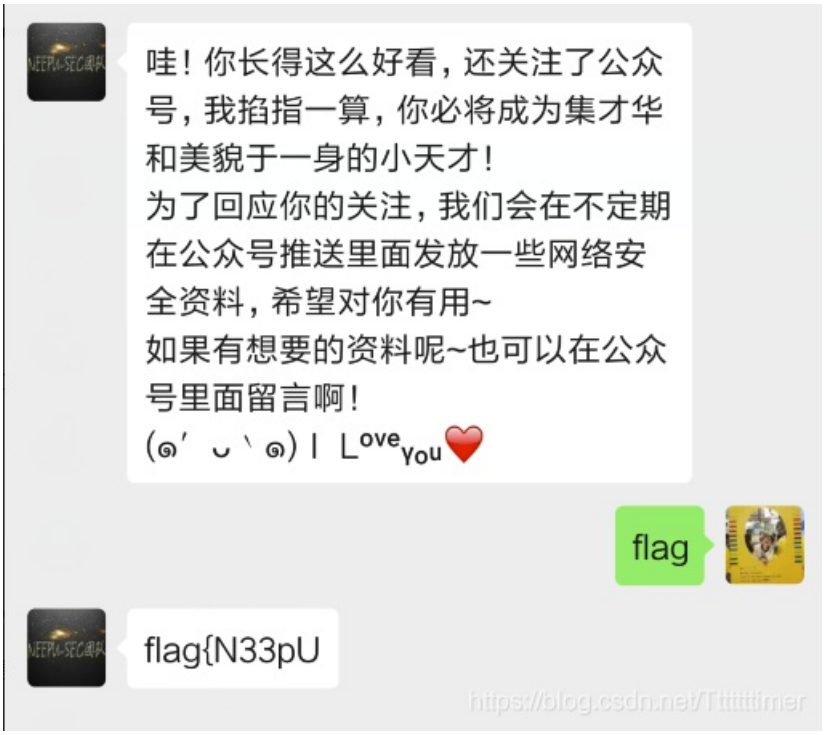
题面是这样的。很直白。

关注了七色堇安全后会得到这样的消息

可以发现有个}于是应该是后半部分。

关注了NEEPU Sec虽然没有自动回复不过问一问对方就知道了。

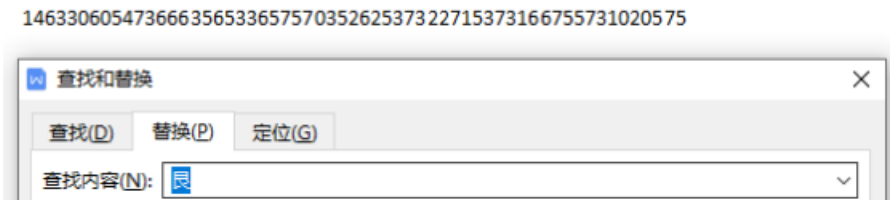




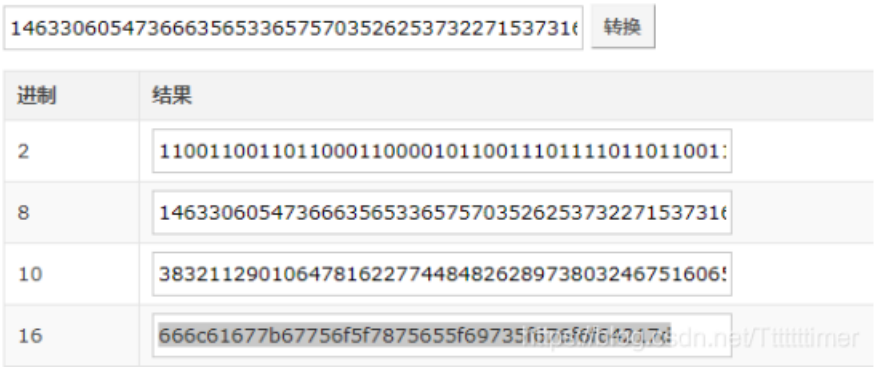
于是得到了完整的flag{N33pU@CcuT}

2.国学

题目给了一堆先天八卦，恰好我对玄学有所涉猎。八卦从乾到坤分别对应了一个数字。最大是八所以在这里应该是八进制。但是八进制并没有8，所以所有的数字都减一，变成0-7的八进制。这里给出对应关系乾0、兑1、离2、震3、巽4、坎5、艮6、坤7。
（~~一开始忘记-1子于是思路卡在这里子，耽误子好久~~）
把原题里的放到word里，挨个替换。



将得到的这串数字放到进制转换工具中进行8进制到16进制转换。



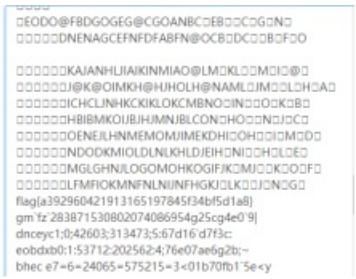
然后再把这串16进制数放到HEX里面就可以得到flag了~

3.XOR

这个xor.txt打开是JiwhJzshc3lyeXZwdHJxeXFzcXZ1cXl3eHR1JnN0liZ1JHEheD0=
最后有个=一看就是base64加密。于是先把它解一次base64。
解码后是这样的。

然后这个题目一看肯定要用到异或的。搜了一圈也没找到在线工具，只能自己写了一个。

```
s = '&,!\';!syryvptrqyqsqvuqywxtu&st\"&u$q!x='
s = list(s)
for x in range(2,100):
    result = ''
    for i in s :
        stemp = ord(i)
        temp = stemp ^ x
        result += chr(temp)
    print(result)
```

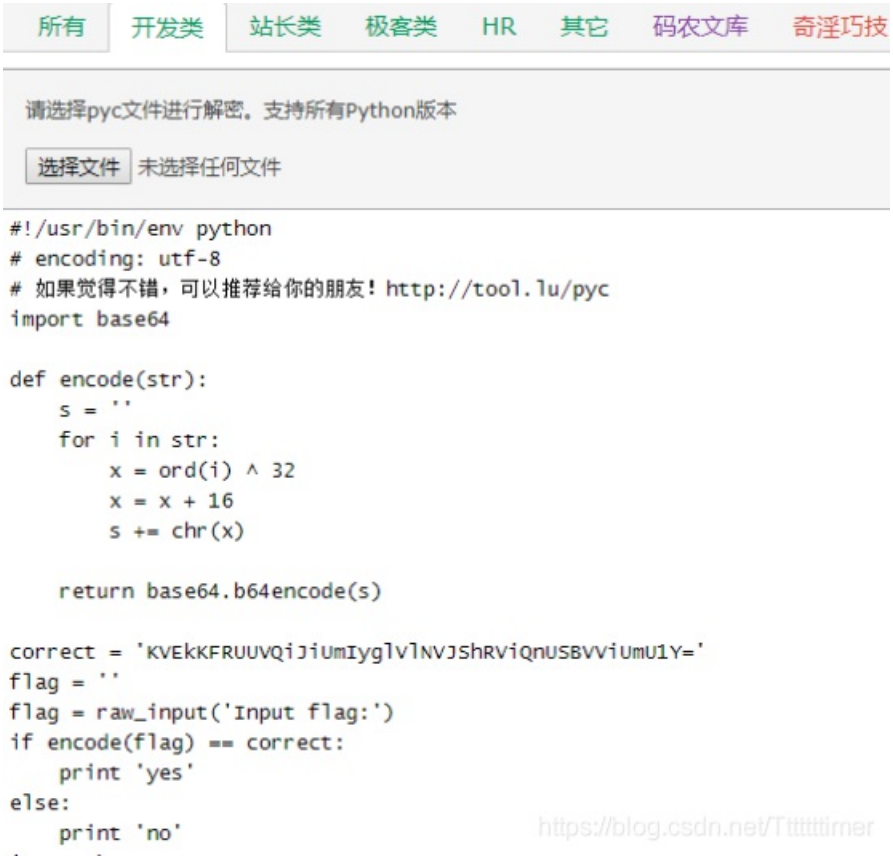


```
□EODO@F8DGOGEG@CGOANBC□E8□□C□G□N□
□□□□□DNENAGCEFNFDFABFN@OCB□DC□□B□F□O

□□□□□□KAJANHLJIAIKINMIAO@LM□KL□□M□I□@□
□□□□□□J@K@OIMKH@HJHOLH@NAML□JM□□L□H□A□
□□□□□□ICHCLJNHKCKIKLOKCMBNO□IN□□O□K□B□
□□□□□□HBIBMKOJBJHJMNJBLCON□HO□□N□J□C□
□□□□□□OENEJLHNMEMOMJIMEKDHI□OH□□I□M□D□
□□□□□□NDODKMIOLDLNILKHLDJEIH□NI□□H□L□E□
□□□□□□MGLGHNJLOGOMOHKOGIFJK□MJ□□K□O□F□
□□□□□□LFMFIOKMNFNLNIJNFHGKJ□LK□□J□N□G□
flag{a3929604219131651978451f34bf5d1a8}
gm`fz`2838715308020740869545g25cg4e0`9|
dnceyc1;0;42603;313473;5:67d16`d7f3c:
eobdxb0:1:53712:202562:4;76e07ae6g2b;~
bhec e7=6=24065=575215=3<01b70fb1`5e<y
```

不知道key所以穷举出来，还好找到了flag{}这种格式的。

4.Pyc

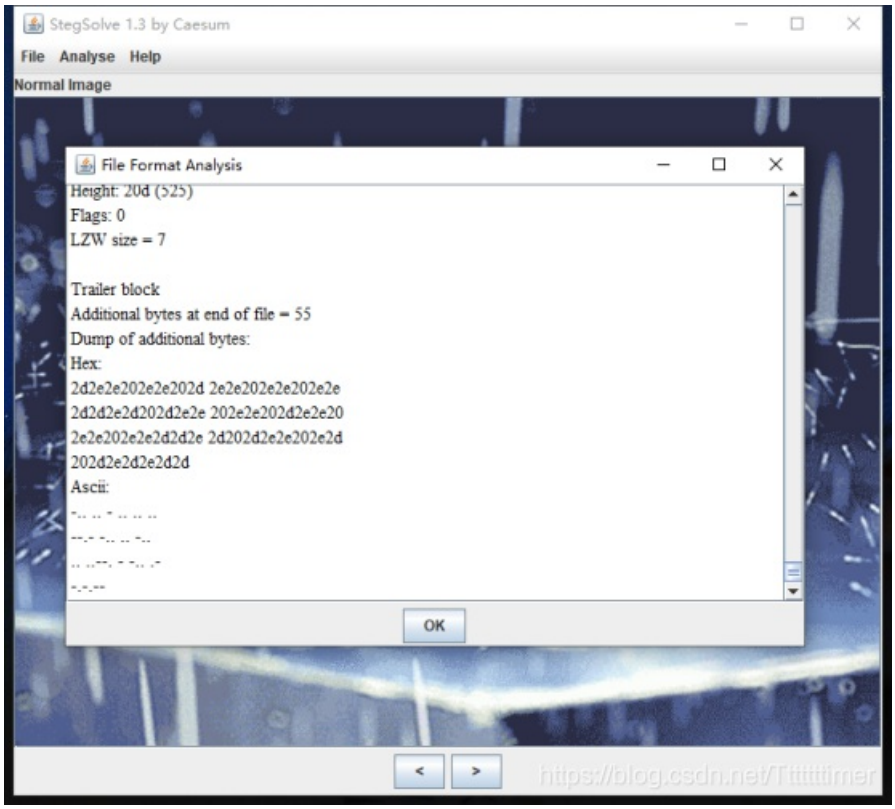


首先这个pyc需要反编译，可以得到以下内容。

可以看到这里首先每一位都与32做异或，然后在+16之后进行base64加密。
只要反着按照步骤解回来就可以了。先进行base64解码，然后-16，最后再跟32异或。
最后得到flag：

5.听

这道题浪费了我好多时间。因为是这个图片是gif的，我就没一开始就先放进hex看看。习惯性打开了Stegsolve，但是没什么特别



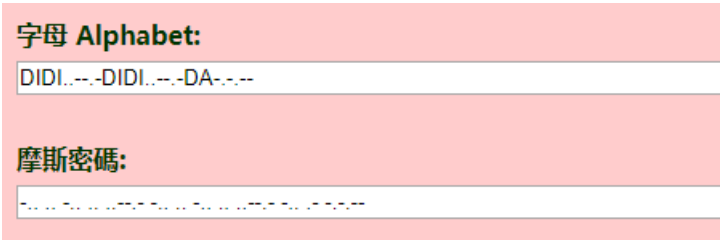
的内容。文件信息里有一些比较特殊的内容。

把这个HEX的内容放进去可以得到一串摩斯密码。

跟这个Ascii里面的是一致的。

然后放到在线工具提示需要在两个连续的之间放上空格。

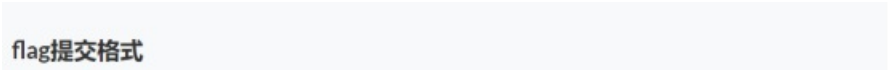
于是在这里卡了很久，换了很多个工具。幸好有个工具把不能翻译的部分完整的留下来了。



是中间那几个很长的，不是数字也不是字母，所以怀疑是符号。于是找了一个符号摩斯密码的对照表。

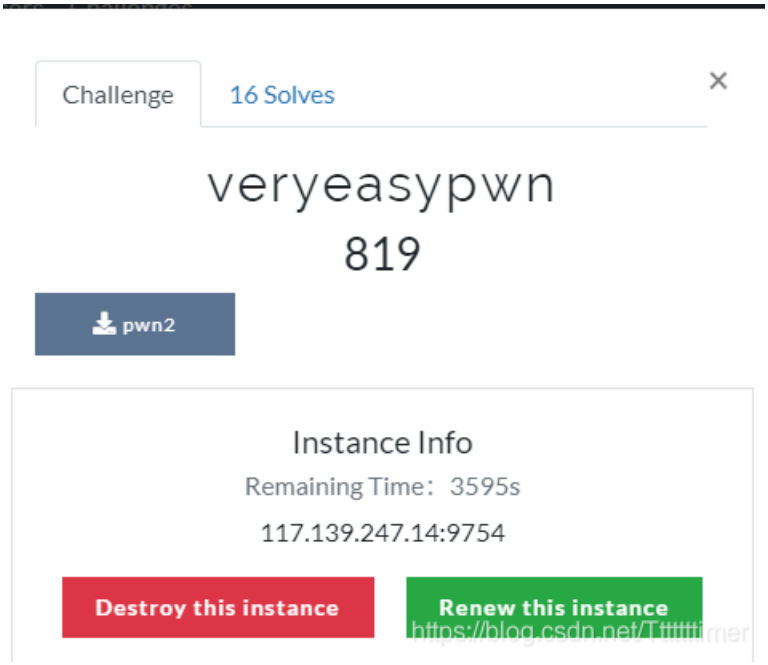
所以不能被工具翻译的内容就是下划线和感叹号了。最后把这些内容放进题目的flag格式里面，这次的比赛公告说基本都是flag{}这种格式，所以就把字母都换成小写，于是就对了。~~（我后来想试试大写对不对，但是已经不能提交了。毕竟摩斯密码是不分大小写的）~~

6.veryeasypwn



> Web题目：flag在根目录带有flag关键字的文件中或web根目录下

首先我们要知道这个。



题面是这样的。
用nc连接进去发现似乎是个计算器。但是直接就报错退出了。

```
C:\WINDOWS\system32>nc 117.139.247.14 9754
ls
----------------------
--calculator
--1.add
--2.subtract
--3.multiply
--4.divide
----------------------
Enter your choice:You've made some mistakes!!
```

要想办法进入它的根目录才行，所以把下载下来的pwn2放到IDA里面可以看到它的主函数。

```
v5 = __readgsdword(0x14u);
puts("----------------------");
puts("--calculator");
puts("--1.add");
puts("--2.subtract");
puts("--3.multiply");
puts("--4.divide");
puts("----------------------");
printf("Enter your choice:");
__isoc99_scanf("%d", &v4);
if ( v4 == 3 )
{
  fun_mul();
}
else if ( v4 > 3 )
{
  if ( v4 == 4 )
  {
    fun_div();
  }
  else
  {
    if ( v4 != 233 )
      goto LABEL_14;
    fun_sys();
```

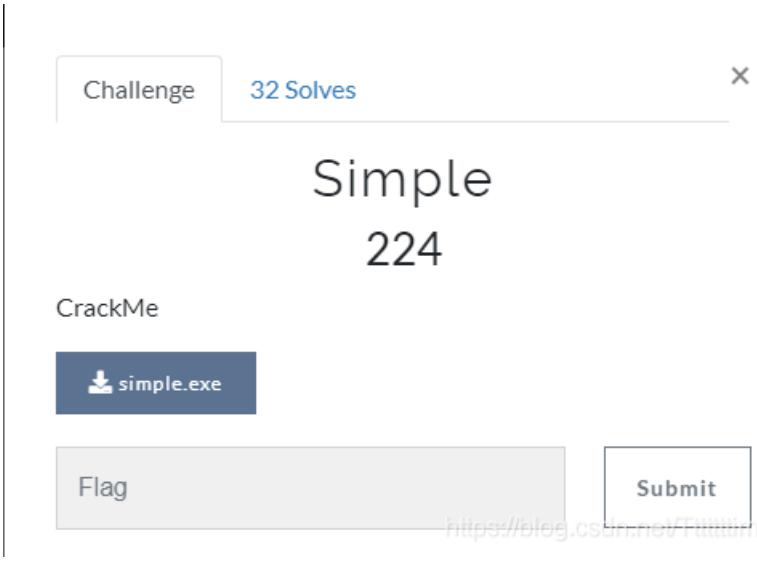
```
  }
 }
 else if ( v4 == 1 )
 {
   fun_add();
 }
 else
 {
   if ( v4 != 2 )
   {
ABEL_14:
     puts("You've made some mistakes!!");
     return 0;
   }
   fun_sub();
 }
 return 0;
```


可以看到除了计算器有的内容外，额外还有一个调用了fun_sys();函数的入口，入口的密码是233。于是返回连接中输入233。

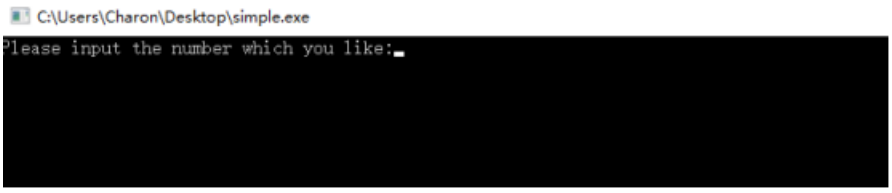
```
C:\WINDOWS\system32>nc 117.139.247.14 9754
233
ls、
/bin/sh: 1: ls、: not found
ls
bin
boot
dev
etc
flag
home
```

Cat一下这个flag就可以了。

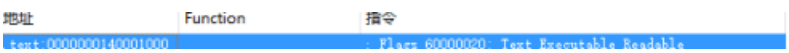
7.Simple



下载下来这个exe运行。随便输入一个数字就退出了。



放到IDA里反汇编，居然没有主函数。没关系直接搜索flag。



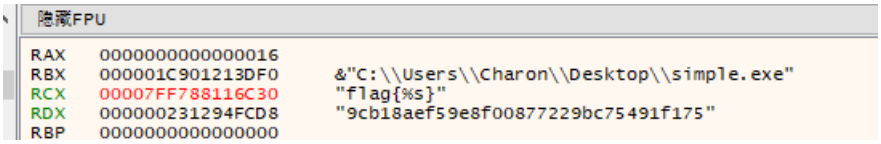
这里有两个flag{%s}是我们要的格式，挨个点进去跳转看看。选择这个有函数名字的。查看它的代码。

```
{
  int v1; // [rsp+20h] [rbp-48h]
  __int128 v2; // [rsp+28h] [rbp-40h]
  __int128 v3; // [rsp+38h] [rbp-30h]
  char v4; // [rsp+48h] [rbp-20h]

  v4 = 0;
  v2 = xmmword_140006BB0;
  v3 = xmmword_140006BC0;
  sub_1400010D2("Please input the number which you like:");
  sub_140001163("%d", &v1);
  if ( v1 == 233333 )
  {
    sub_1400010D2("Yes, I like it too!!!\n");
    sub_1400010D2("flag{%s}");
  }
  else
  {
    sub_1400010D2("OK,but I don't like that number");
  }
  return 0i64;
}
```


于是可以看到输入的数字是233333时可以看到flag{%s}，但是他会自己exit，所以需要一个断点让他停下来。
放到dbug工具里面，依旧搜索flag{%s}找到位置，为避免消失我在附近设了好多个断点。

在这些断点之间慢慢运行，可以观察到右边一些寄存器信息，于是就得到了flag。



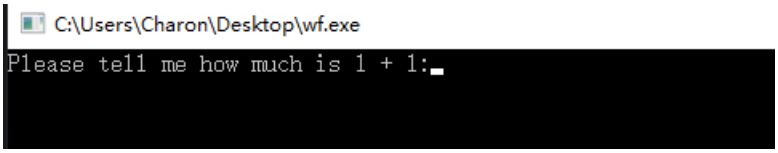
8.RSA

这道题完全不会，但是我觉得一定有迹可循，于是我去i春秋的密码题里面搜了RSA，果然让我搜到了。
https://www.ichunqiu.com/writeup/detail/581
几乎完全一样，除了我们这个需要输入一串hex来换取flag而他直接丢进hex里面就可以了。

于是按照他的思路和脚本一步一步走下来最后就能得到结果了。

9.WF

首先把它运行一下看看，结果得到这样的运行结果。

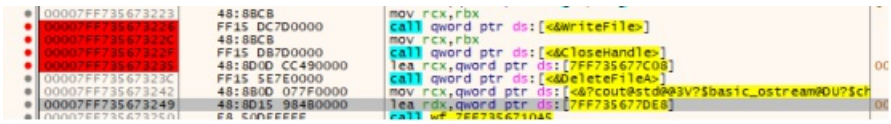


可以知道它的主函数一定可以通过搜索Please找到。放到IDA里面寻找它的主函数。然后F5看它的伪代码，得到如下内容。

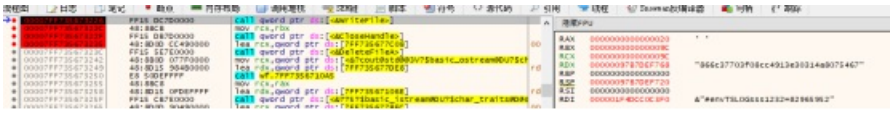
```
char v11; // [rsp+68h] [rbp-20h]

sub_1400010A5(std::cout, "Please tell me how much is 1 + 1:");
std::basic_istream<char,std::char_traits<char>>::operator>>(std::cin, &v7);
if ( v7 != 2 )
{
  v0 = sub_1400010A5(std::cout, "no no no!!!");
  std::basic_ostream<char,std::char_traits<char>>::operator<<(v0, sub_14000106E);
  exit(0);
}
sub_1400010A5(std::cout, "yes,so easy!");
v1 = CreateFileA("D:\\flag", 0xC0000000, 0, 0i64, 4u, 0x80u, 0i64);
if ( v1 == (HANDLE)-1i64 )
{
  v2 = sub_1400010A5(std::cout, "创建文件句柄失败");
  std::basic_ostream<char,std::char_traits<char>>::operator<<(v2, sub_14000106E);
}
v11 = 0;
v3 = 0i64;
v10 = xmmword_140007BC0;
NumberOfBytesWritten = 0;
v4 = _mm_load_si128((const __m128i *)&xmmword_140007C10);
Buffer = xmmword_140007B80;
do
{
  _mm_storeu_si128(
    (__m128i *)((char *)&Buffer + v3),
    _mm_add_epi8(_mm_loadu_si128((const __m128i *)((char *)&Buffer + v3)), v4));
  v3 += 16i64;
}
while ( v3 < 32 );
WriteFile(v1, &Buffer, 0x20u, &NumberOfBytesWritten, 0i64);
CloseHandle(v1);
DeleteFileA("D:\\flag");
v5 = sub_1400010A5(std::cout, "HA,did you get the flag?");
std::basic_ostream<char,std::char_traits<char>>::operator<<(v5, sub_14000[illegible]);
system("pause");
return 0i64;
```


可以看到如果输入不是数字2的话将会显示byebye然后直接退出程序。输入数字2可以往下运行。下面的内容里有一句明显的“get the flag”，也就是说这句显示之前就可以得到结果了。
而这句显示前有几个指令，一个是WF，一个是DF。写入然后直接删除。那么我们的任务就是让它停在这中间，找到写入的flag文件。
放进debug工具，查找这句话的位置，在它前面设置断点。



然后F9运行，输入数字2调试。
然后在右边寄存器里其实flag就已经出现了。



原本我以为这是一个hex十六进制，但是是866c而不是666c所以在外面套上flag{}就提交看看，结果对了。

10.Alice is loney（为啥不是~~lonely~~）

因为之前的题目知道了RSA的主要格式，所以nc进看到内容的是就知道是RSA了。这里应该是已经c、n、e求m的算法。
这道题参考了王大锤的二哥王二狗的笔记的第三部分已知 公钥（n, e） 和 密文 c 求 明文 m?
https://blog.csdn.net/vhkjhwbs/article/details/101160822
但是这道题的数给的很有趣，把n拿去因数分解居然本身就是一个质数，因数只有她自己和1。王二狗给的脚本里q和p最小要是2，所以王二狗的脚本就不能再用了。但是这次的e很有趣，是2。也就是说在该题设下，c=m的平方再对n取模。
很明显c的值远远小于n，那么在已知n有300多位的情况下，且n是一个大素数。m的平方只有两种情况，一种是$m^2=n+c$，或者$m^2<n$。
第一种情况太长了，所以先求一下第二种，当$m^2<n$时，这个mod运算相当于没做。此时的m = gmpy2.isqrt©。
将这个m输出成hex，再输入到控制台里就出了flag。（果然没有太长）

11.饿了么

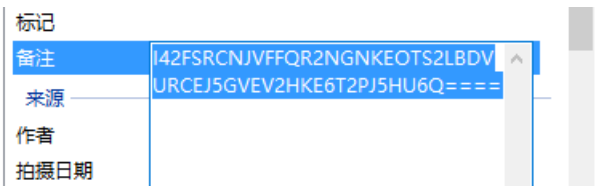
首先拿到一个普通的压缩包，里面只有个叫base的图片，没什么特别的。但是这个名字base却不叫肯德基或者KFC首先想到是

base64加密。然后就没什么特别的内容了。

把图片解压出来放到HEX里面查看，发现冗杂的信息很多，而且我注意到了zip的文件头。可以看到里面隐藏了一个file.txt

```
1C0  08 49 80 24 84 28 92 22 50 84 24 30 42 10 90 09  .I€$„('"P„$0B...
1D0  05 08 48 03 A2 10 84 C6 08 42 10 00 84 21 30 04  ..H.¢.„Æ.B..„!0.
1E0  D0 84 00 90 84 20 01 08 42 04 08 42 10 30 42 10  Ð„..„ ..B..B.0B.
1F0  80 3F FF D9 50 4B 03 04 14 00 09 00 08 00 EC BC  €?ÿÙPK.......ì¼
200  35 4F 1F 3B 0E EA 1E 47 00 00 24 77 00 00 08 00  5O.;.ê.G..$w....
210  00 00 66 69 6C 65 2E 74 78 74 4B 57 BA 2C 99 53  ..file.txtKW°,™S
220  F7 17 8C 88 A4 AC AC B4 D5 83 2B AC FB 93 3C 16  ÷.Œˆ¤¬¬´Õƒ+¬û"<.
```

于是把base.jpg变成base.zip得到一个加密的压缩包。



密码找了好久都没找到。。连CRC碰撞都快尝试了。迫不得已又看了一下图片详情~~（其实我刚拿到图片的时候就看了一次的，可能当时没仔细看）~~
果然发现备注里有以====结尾的一串字符。
虽然一开始想到用base64，但是真的拿去解码的时候是一串乱码不能再继续了。而且这也是我第一次见到四个等号结束的base加密。就回到了开始的疑问为什么文件名是base而不是base64呢。于是我猜base除了base64可能还有别的方式。

于是找到了一个博客。这里参考。
https://www.cnblogs.com/superking-sukai/p/8491349.html
发现三个以上等号结尾应该是base32。于是去解码。这里需要解码两次，会得到16进制串。

丢进HEX得到password（就是password，直白的密码）。
解压缩得到file.txt，里面的字符也太多了。同样参考上面的博文，拿去做词频统计。~~（他的工具没用明白）~~
用了一个简单的词频统计。

可以看到除了固定格式flag{}之外按顺序是6432bgHieAsFck。只能我自己拼起来可读内容了。（另外这个我后来问了一下同学，只有我自己的词频分析是这样的，他们的词频分析出来就已经是正确的flag了）
一眼望去有bAse6432，结合图片还有kFc，最后剩下一个Hi。
所以是flag{Hibase6432kFc}

剩下我自己没有做出来的WP可以参考dalao的内容

> https://mp.weixin.qq.com/s/FG5AM0PGftQuQzCl5-Es-w