HSC-1th WriteUP

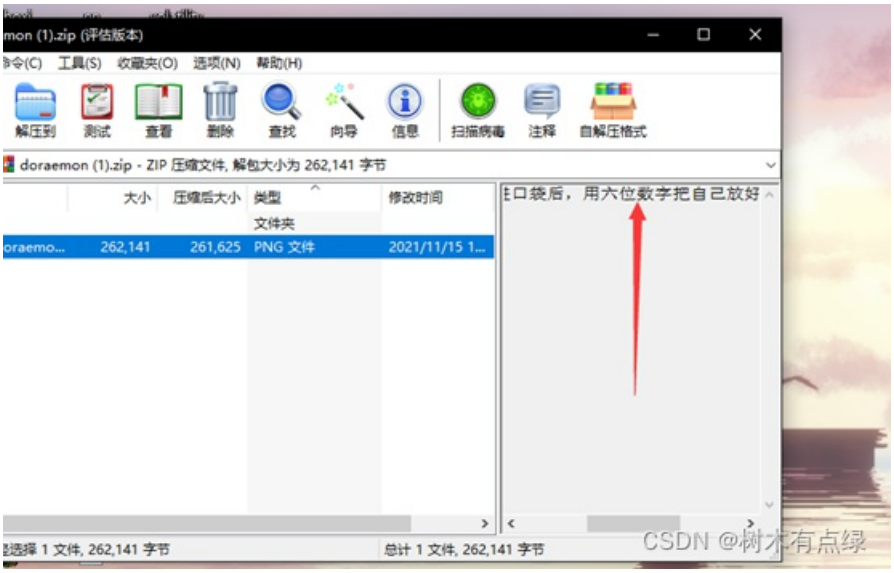
原创

树木有点绿 于 2022-03-12 09:26:11 发布 49 收藏
分类专栏： CTF 文章标签： 安全 网络安全

Sign-in

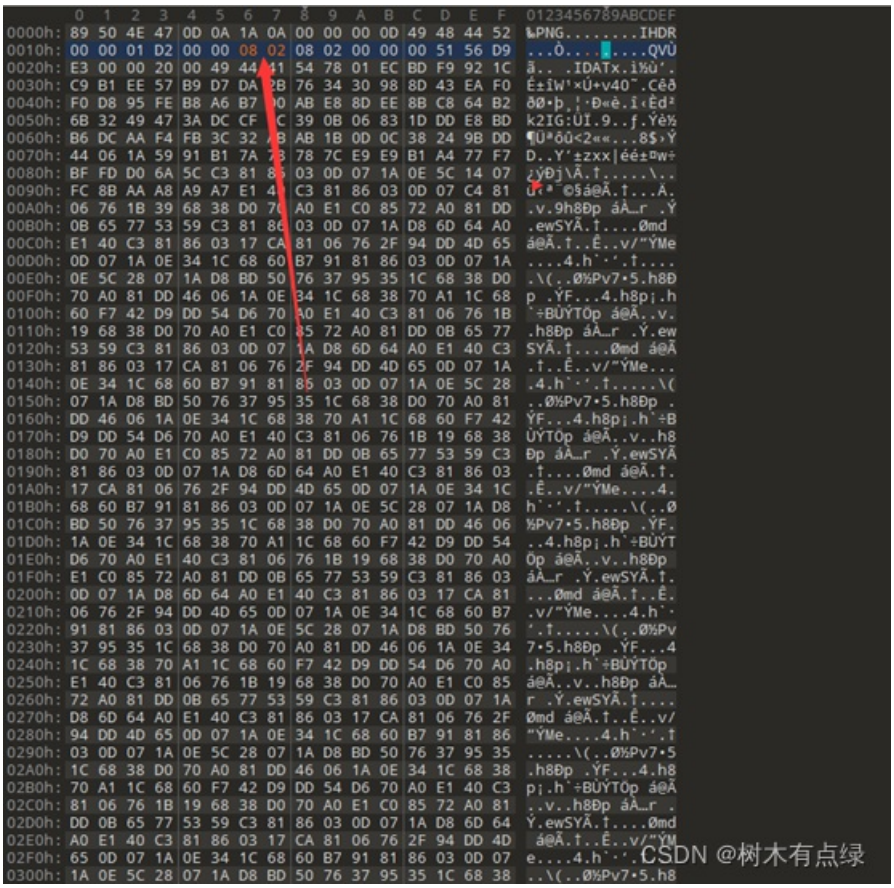
关注公众号，回复HSC2019，拿到flag

DORAEMON



开幕告诉我是6位数字，经过爆破得到密码为：376852

用密码解开压缩包，校验宽高，发现高度不高，把图片改长

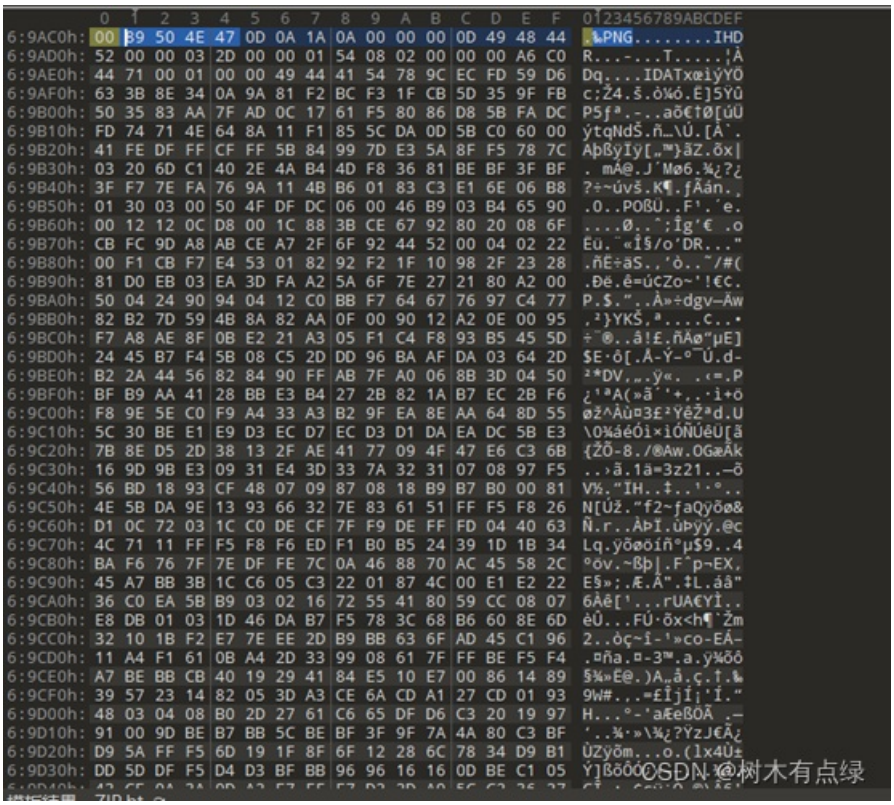


改长图片后得到

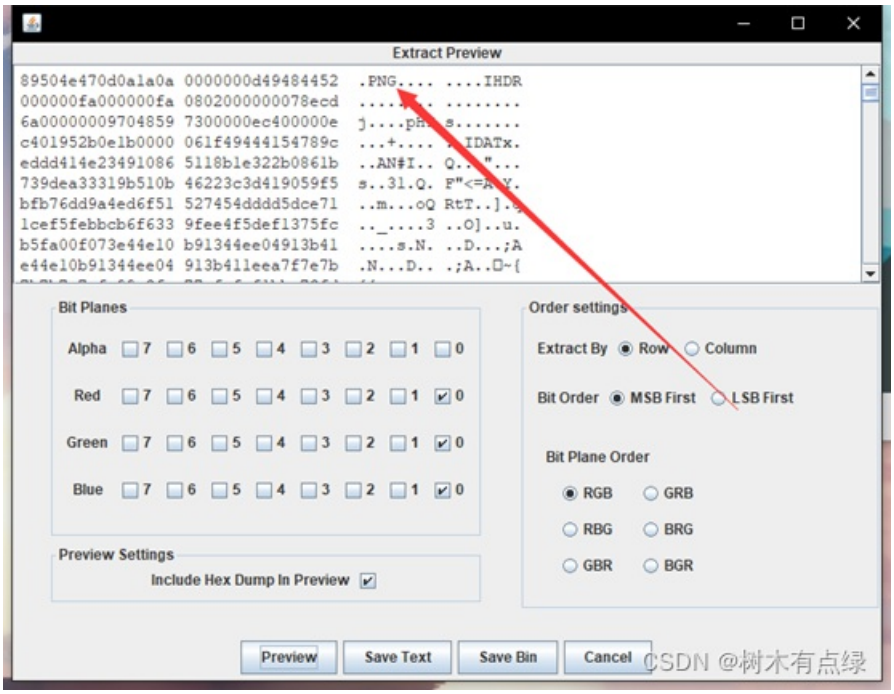
把二维码定位符补上，扫码得到flag

WIRESHARK

010打开，发现还有一个png图片



手动分离出来，得到一张小鲨鱼的图片，我们直接看看它的最低有效位



发现还存在一个png，我们把它导出，得到一个二维码

扫码得到`wrsak..iehr370`

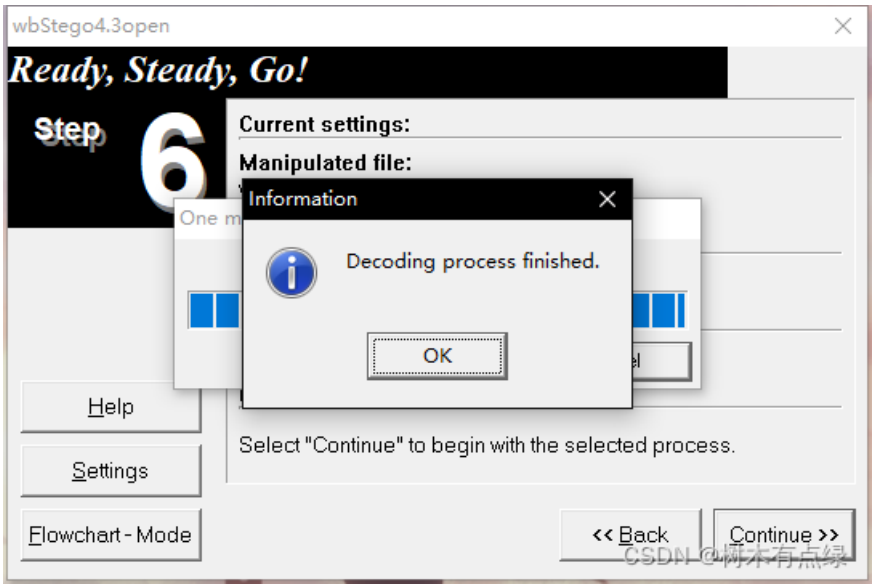
Decode Succeeded

Raw text	wrsak..iehr370
Raw bytes	71 a4 0e 77 72 73 61 6b 2e 2e 69 65 68 72 33 37 30 00 ec 11 ec 11 ec 11 ec 11 ec 11
Barcode format	QR_CODE
Parsed Result Type	TEXT
Parsed Result	wrsak..iehr370

把它进行一次栅栏密码枚举，得到真正的压缩包密码：wireshark3.7.0

解密得到一个wireshark文件，查看16进制，发现很像pdf文件

补上PDF文件头，直接查看最低有效位



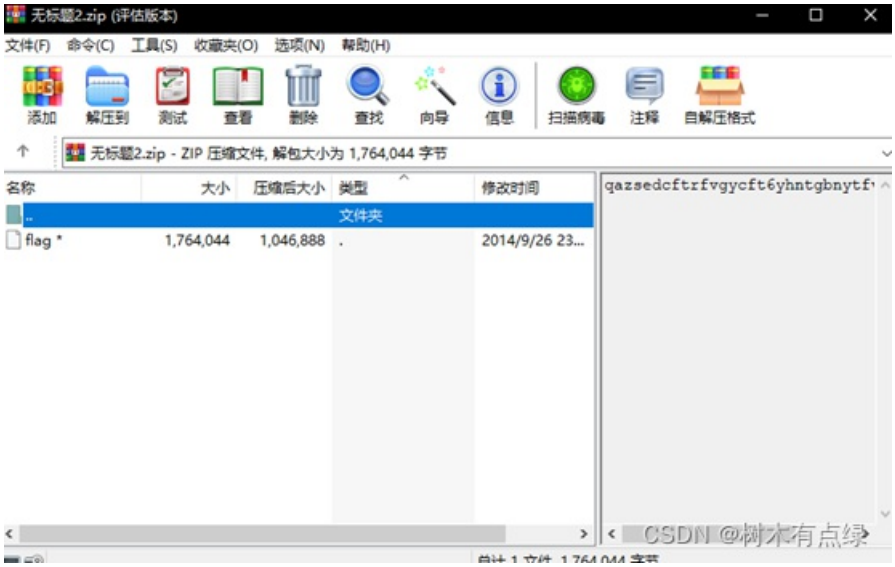
成功get到了flag

汝闻，人言否

下载解压得到一张png图片，直接16进制打开，搜索PNG文件尾（AE 42 60 82）

发现这里有一个错位的压缩包，我们把关键位置的4B 50改为50 4B（不止一处）

随后我们手动分离压缩包



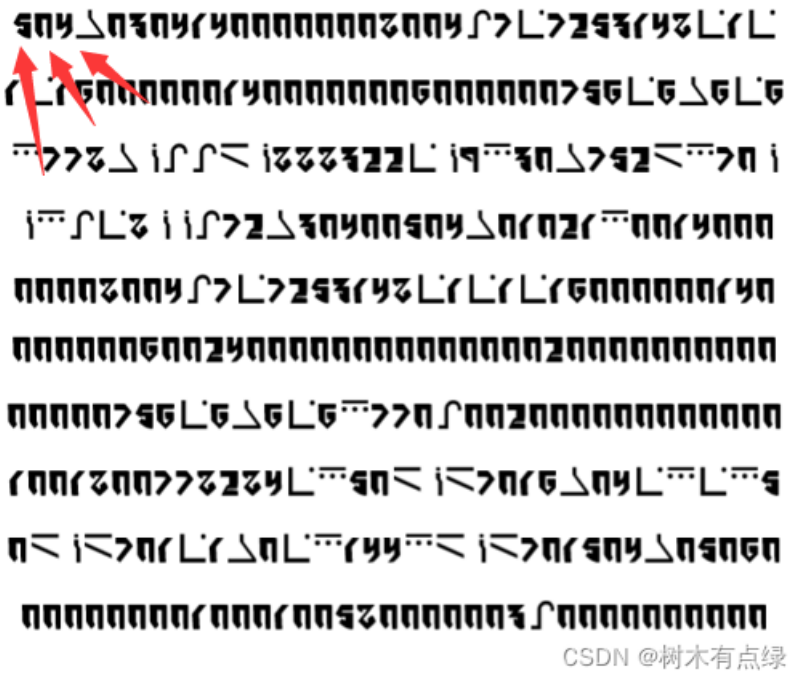
看右面的字符串，我们用键盘同时按下几个键，可以看到很像字符，保留大写，得到压缩包密码：WVALOU

解压出来得到一个没有后缀的文件，查看hex发现是WAV文件，使用AU打开，得到flag

PERFORMANCE-ART

这个题目是真的很艺术，看到图片后首先想到了星际密码，下面给出对照表

对照着对照着发现，有一些字符对照表里没有



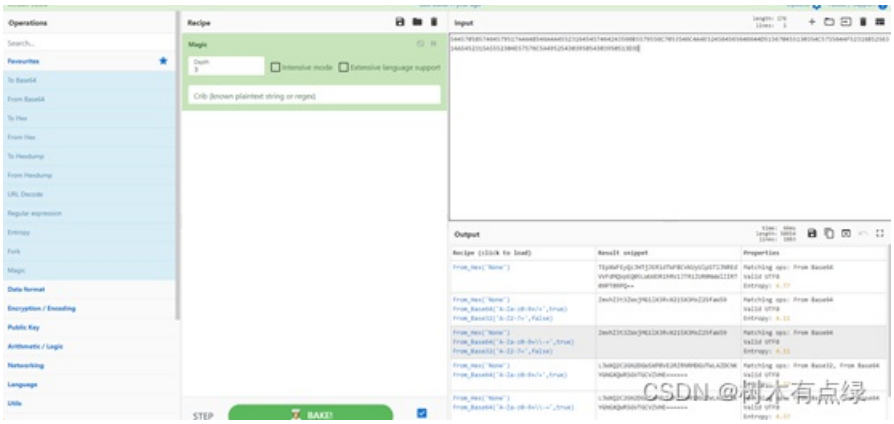
根据我们的经验，发现这个很像504B03041400，很容易联想到压缩包

接下来就是20多分钟的摸索了，慢慢摸索出来了这些对照表没有的字符代表什么，得到zip

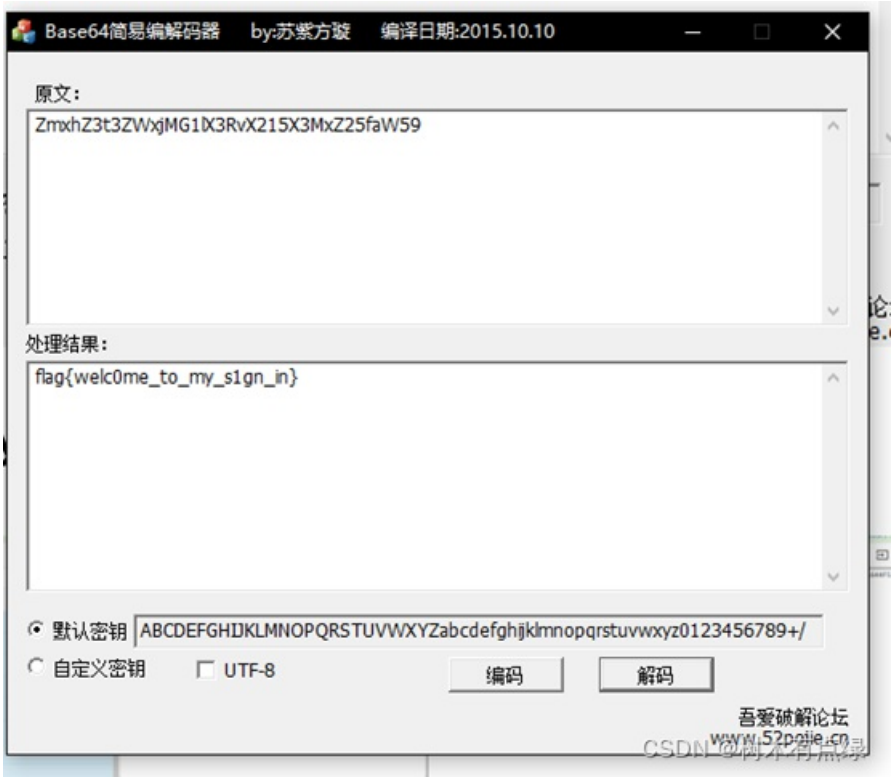
```
50 4B 03 04 14 00 00 00 08 00 4A 7E 72 53 14 8E

1E 1E 16 00 00 00 14 00 00 00 06 00 00 00 75 6E

6B 6E 6F 77 8B CA AD C8 88 32 2E C9 F3 0B 75 2D

F7 0C CF AE 8C CA 72 B3 04 00 50 4B 01 02 1F 00

14 00 00 00 08 00 4A 7E 72 53 14 8E 1E 1E 16 00

00 00 14 00 00 00 06 00 24 00 00 00 00 00 00 00

20 00 00 00 00 00 00 00 75 6E 6B 6E 6F 77 0F 00

20 00 00 00 00 00 01 00 18 00 77 82 84 EF 50 DC

D7 01 6B 04 EF EF 60 DC D7 01 E1 B0 EF 14 4F DC

D7 01 50 4B 05 06 00 00 00 00 01 00 01 00 58 00

00 00 3A 00 00 00 00 00
```

解压出来解密base64，得到flag

Easy SignIn



直接用军刀一把梭，得到ZmxhZ3t3ZWxjMG1lX3RvX215X3MxZ25faW59，进行base64解密，得到flag



LINE-GENERATION-TEST

观察描述，发现是在暗示希尔，也就是希尔密码

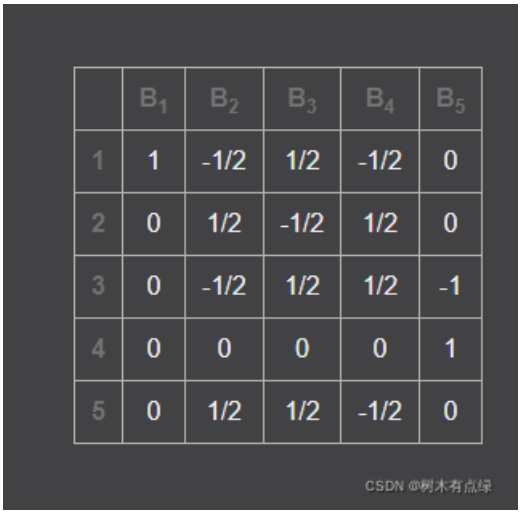
Key_encrypt:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Enc:

$$\begin{pmatrix} 9 \\ 23 \\ 0 \\ 13 \\ 19 \end{pmatrix}$$

我们直接求出它的矩阵的逆



	B_1	B_2	B_3	B_4	B_5
1	1	-1/2	1/2	-1/2	0
2	0	1/2	-1/2	1/2	0
3	0	-1/2	1/2	1/2	-1
4	0	0	0	0	1
5	0	1/2	1/2	-1/2	0

学习了一下矩形乘法的运算方式（左边矩阵每行乘以右边的列向量），得到明文对应的数字，拿到关键的“RSCTF”，对它进行md5加密再加上flag{}，得到最终flag

RSA

脚本如下:

```
n=124689085077258164778068312042204623310499608479147230303784397390856552161216990480107601962337145795119

t=10



import gmpy2

for k in range(-1000000,1000000):

    x=gmpy2.iroot(k**2+4*t*n,2)




    if x[1]:



            p=(-k+x[0])//(2*t)

            q=t*p+k

            break



import gmpy2

from Crypto.Util.number import long_to_bytes,bytes_to_long

phi=(p-1)*(q-1)

e=57742

c=57089349656454488535971268237112640808678921972499308620061475860564979797594115551952530069277022452969 3



t=gmpy2.gcd(e,phi)

d=gmpy2.invert(e//t,phi)

m=pow(c,d,n)

msg=gmpy2.iroot(m,t)

if msg[1]:

    print(long_to_bytes(msg[0]))
```

AFFINE

研究研究脚本，发现是仿射密码

```python
# -*- coding: utf-8 -*-
import string
import hashlib

letter=string.ascii_letters+string.digits

def encrypt(m, c, a, b):
    for i in range(len(m)):
        ch=m[i]
        t=(letter.index(ch) * a + b) % 62
        c.append(letter[t])
    d = ''.join(c)
    print(d)

m =
c = []
a =
b =

assert ("flag" in m)

print("加密后的密文为：")
Cipher = encrypt(m, c, a, b)
flag = hashlib.md5("".join(str(m)).encode("utf8")).hexdigest()
#print(flag)
"""
加密后的密文为：
xGJ13kkRK9QDfGRQomFOf9NIa9LKVIvGqVIaVO9NOkorv
"""
```


求解仿射密码需要两组明文与对应的密文来求出系量和增量，题目中我们已知flag是明文的一部分，我们直接爆破获取密钥然后解出密码

```
letter= 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'

meg='xGJ13kkRK9QDfORQomFOf9NZs9LKVZvGqVIsVO9NOkorv'

known='flag'



for i in range(90):

    for j in range(90):

        c=[]

        for len in range(4):

            ch = known[len]

            t=(letter.index(ch) * i + j) % 62

            c.append(letter[t])

        d = ''.join(c)

        if  d in meg:

            print("i=",i)

            print("j=",j)

flag = ""

for c in meg:

    for m in letter:

        if letter[(11*letter.index(m)+17)%62] == c:

            print(m,end='')
```

运行脚本后得到flag，进行md5加密后得出正确flag

CLICK

查看源码，分析main.js

发现有一串base64，我们解密一下得到flag

Web-sign in

3080/robots.txt

我当然知道robots协议了，我们在后面加上/rotbots.txt

```
User-agent: *
Disallow:
Disallow: fiag_ls_h3re.php
```

我们访问这个PHP，Ctrl+U拿到flag

hiahia o(*^▽^*)┛

直接IDA打开，F5看一下伪代码

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  __int64 v4; // [rsp+20h] [rbp-40h]
  __int64 v5; // [rsp+28h] [rbp-38h]
  int v6; // [rsp+30h] [rbp-30h]
  char v7[28]; // [rsp+40h] [rbp-20h]
  unsigned int i; // [rsp+5Ch] [rbp-4h]

  _main();
  v4 = 788729552762125706 5i64;
  v5 = 2682778951892353141i64;
  v6 = 1882989628;
  printf("please input your flag:");
  scanf("%s", v7);
  for ( i = 0; (signed int)i <= 19; ++i )
  {
    *((_BYTE *)&v4 + (signed int)i) = flag((unsigned int)*((char *)&v4 + (signed int)i), i);
    if ( v7[i] != *((_BYTE *)&v4 + (signed int)i) )
    {
      printf("Aha, Well done!");
      return 0;
    }
  }
  printf("Aha!");
  return 0;
}
```


看看flag函数

```
  v3 = a1;
  if ( a2 > 9 )
  {
    if ( a2 > 9 )
    {
      if ( !(a2 & 1) )
        v3 = a1 - 11;
      if ( a2 % 2 == 1 )
        v3 += 13;
    }
  }
  else
  {
    if ( !(a2 & 1) )
      v3 = a1 - 3;
    if ( a2 % 2 == 1 )
      v3 += 5;
  }
  return v3;
}
```


哇哦，虽然看起来很复杂，但其实就是简单的ASCII加减运算，直接使用python还原一波

```
def get_flag(a1,a2):

    v3 = ord(a1)

    if (a2>9):

        if a2%2 ==1:

            v3+=13

        if a2&1 == 0:

            v3 = ord(a1) -11

    else:

        if a2%2 == 1:

            v3+=5

        if a2&1 == 0:

            v3 = ord(a1)-3

    return chr(v3)

v4 = "igdb~Mumu@p&>%;%<$<p"

for i in range(20):

    flag = get_flag(v4[i],i)

    print(flag,end='')
```

运行得到flag

ANDROID

直接jeb反编译源码

反编译到了有兴趣的内容，就是个java代码，给了两个数组，分析运算的过程

算法分析可知：当i是个偶数的时候，flag对应的字符可以由列表1对应下表的数字和i异或得到

```
a1 = [102, 13, 99, 28, 127, 55, 99, 19, 109, 1, 121, 58, 83, 30, 79, 0, 64, 42]

a2 = [42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42]

a3 = []



for i in range(17):

    if i%2 == 0:

        a3.append(a1[i] ^ i)

print(a3)
```

得到了

```
[102, 97, 123, 101, 101, 115, 95, 65, 80]
```

利用上面求出的一半flag的ASCII码可以用类似的异或操作求出下一半

```
[108, 103, 82, 118, 114, 101, 95, 80]
```

对应ASCII码得到最终flag