CTF.show：misc入门24-49

原创

FW_ENJOEY 于 2021-04-10 13:10:58 发布 591 收藏 3

分类专栏： CTF.show CTF_MISC_Writeup

本文链接：https://blog.csdn.net/qq_46230755/article/details/115311646


CTF.show 同时被 2 个专栏收录

23 篇文章 4 订阅

订阅专栏


CTF_MISC_Writeup

24 篇文章 2 订阅

订阅专栏

上一次剩3题坐半天不会写。还是先把后面写了吧。

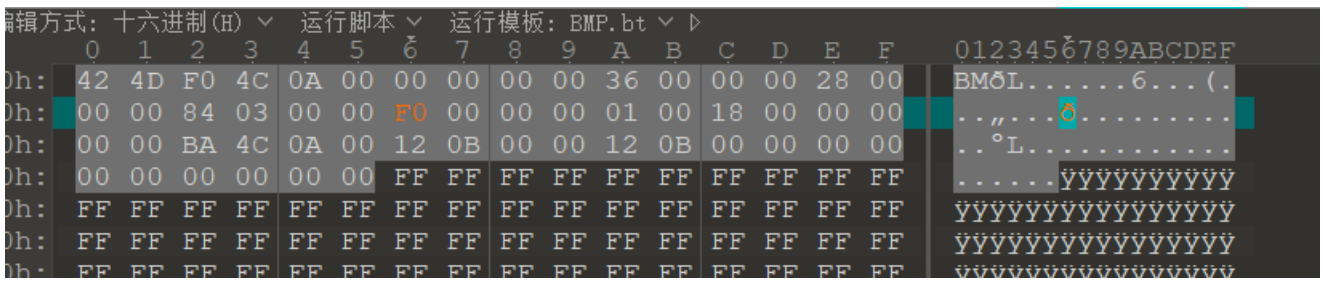
只是为了自己记录，希望师傅们多提点建议多教教我。

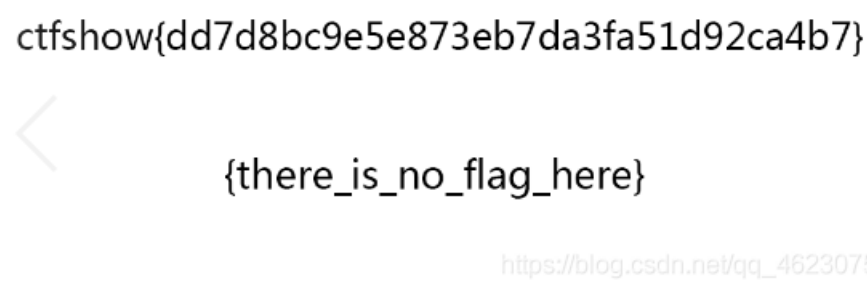
文章目录

图片篇(文件结构)

misc24

提示：`flag在图片上面。`
010里面修改bmp的高度

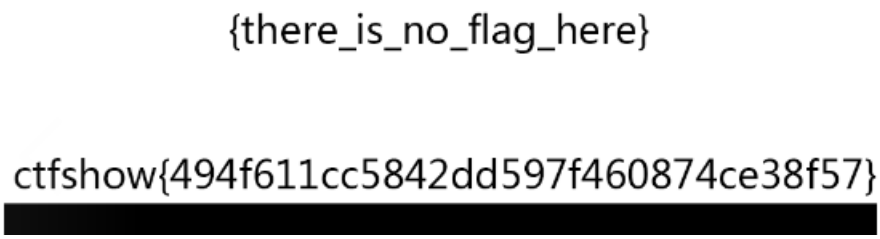





```
ctfshow{dd7d8bc9e5e873eb7da3fa51d92ca4b7}
```

misc25

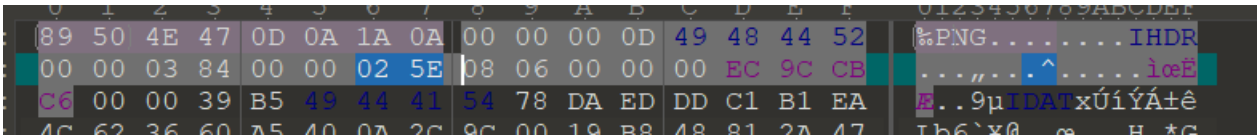
提示：`flag在图片下面。`
打开图片，调整图片高度，得到flag。
在这里插入图片描述



利用脚本进行crc爆破

```
#coding=utf-8
import zlib
import struct
#读文件
file = 'C://Users/24471/Desktop/misc26.png'  #注意，1.png图片要和脚本在同一个文件夹下哦~
fr = open(file,'rb').read()
data = bytearray(fr[12:29])
crc32key = eval(str(fr[29:33]).replace('\\x','').replace("b'",'0x').replace("'",''))
#crc32key = 0xCBD6DF8A #补上0x，copy hex value
#data = bytearray(b'\x49\x48\x44\x52\x00\x00\x01\xF4\x00\x00\x01\xF1\x08\x06\x00\x00\x00')  #hex下copy grep hex
n = 4095 #理论上0xffffffff,但考虑到屏幕实际，0x0fff就差不多了
for w in range(n):#高和宽一起爆破
    width = bytearray(struct.pack('>i', w))#q为8字节，i为4字节，h为2字节
    for h in range(n):
        height = bytearray(struct.pack('>i', h))
        for x in range(4):
            data[x+4] = width[x]
            data[x+8] = height[x]
            #print(data)
        crc32result = zlib.crc32(data)
        if crc32result == crc32key:
            print(width,height)
            #写文件
            newpic = bytearray(fr)
            for x in range(4):
                newpic[x+16] = width[x]
                newpicuo[x+20] = height[x]
            fw = open(file+'.png','wb')#保存副本
            fw.write(newpic)
            fw.close
```

得到一个新的png，然后拖到010里面去查看



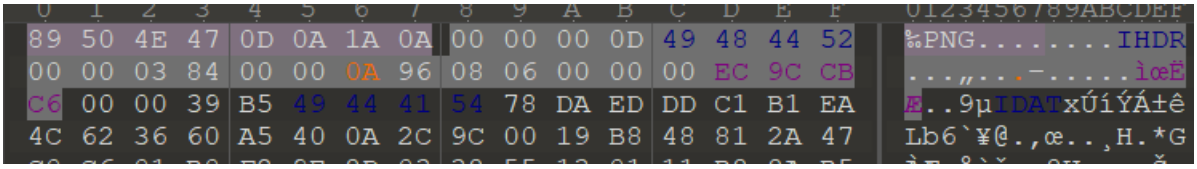
高度的hex为25e
得到flag

```
ctfshow{94aef125e087a7ccf2e28e742efd704c}
```

misc26

提示：`flag在图片下面。`

和25一样，也是改高度，但是这次改的比较多



ctfshow{94aef1
+True height(hex) of this picture+
087a7ccf2e28e742efd704c}

得到flag，不过有提示，不完整。

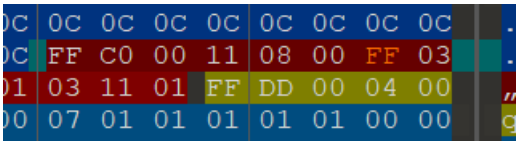
这里需要，找到图片的真实高度。

```
ctfshow{94aef0961087a7ccf2e28e742efd704c}
```

misc27

提示：`flag在图片下面`

图片是jpg，修改图片高度，把150的十六进制0096改成00FF得到flag



{there_is_no_flag_here}

ctfshow{5cc4f19eb01705b99bf41492430a1a14}

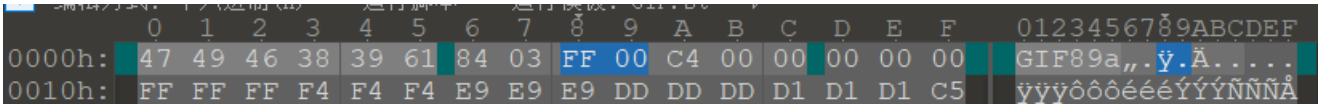

```
ctfshow{5cc4f19eb01705b99bf41492430a1a14}
```

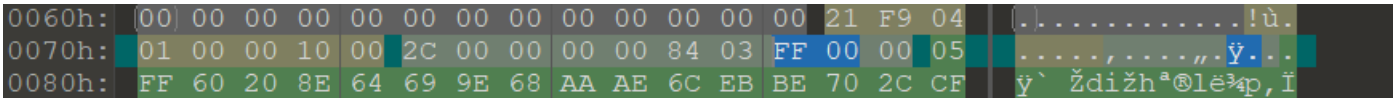
misc28

提示：`flag在图片下面。`

图片是gif，修改gif高度

一共要改两处





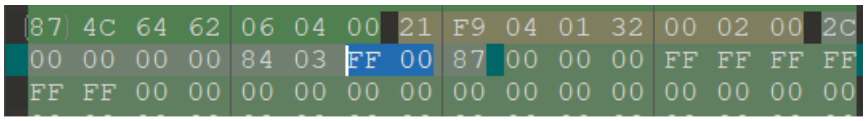
{there_is_no_flag_here}

ctfshow{59c8bc525426166b1c893fe12a387fd7}


```
ctfshow{59c8bc525426166b1c893fe12a387fd7}
```

misc29

与上一题一样。对gif全部帧进行替换。



搜索全部的`96 00`换成`FF 00`

{there_is_no_flag_here}

ctfshow{03ce5be6d60a4b3c7465ab9410801440}

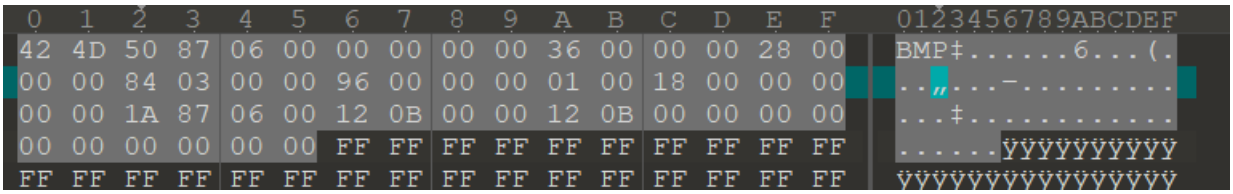
```
ctfshow{03ce5be6d60a4b3c7465ab9410801440}
```

misc30

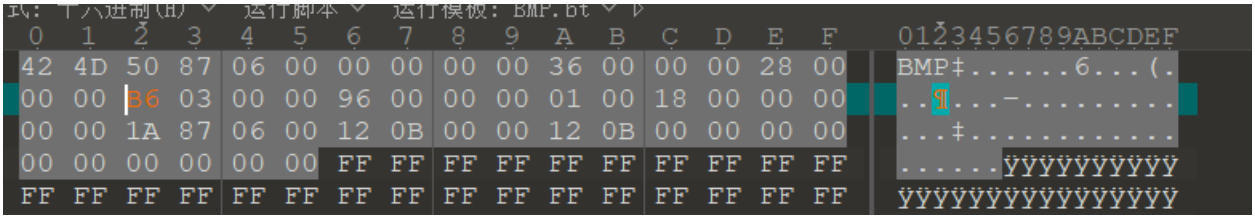
提示：正确的宽度是950。

把图片放入010，在宽度的位置进行修改成950的二进制`03 B6`，注意要倒着写。

（我一开始也不知道的，是打开来发现宽度900写的是84 03，所以才想bmp是反着的）



改完得到flag

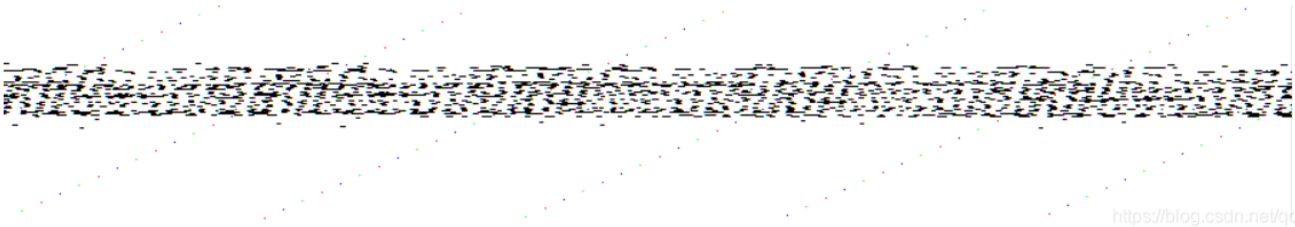


ctfshow{6db8536da312f6aeb42da2f45b5f213c}

```
ctfshow{6db8536da312f6aeb42da2f45b5f213c}
```

misc31

提示：高度是正确的，但正确的宽度是多少呢。



题目提示了改宽度

从v3师傅的博客抄的爆破bmp宽度的脚本

```
import struct
import zlib
f = open('./misc31.bmp','rb')
c = f.read()
width = c[18:22]
height = c[22:26]
# 爆破bmp宽度
for i in range(900,1100):
    f1 = open('./bpout/'+str(i)+'.bmp','wb')
    # print(struct.pack('>i',i)[::-1])
    img = c[:18]+struct.pack('>i',i)[::-1]+c[22:]
    f1.write(img)
    f1.close()
```

在1082.bmp找到正确的flag

ctfshow{fb09dcc9005fe3feeefb73646b55efd5}

```
ctfshow{fb09dcc9005fe3feeefb73646b55efd5}
```

misc32

提示：高度是正确的，但正确的宽度是多少呢

CRC爆破宽度

```
import zlib
import struct
image=open("misc32.png","rb").read()
for i in range(4096):
    for j in range(4096):
        c=image[12:16]+struct.pack('>i',i)+struct.pack('>i',j)+image[24:29]
        CRC=0xE14A4C0B
        if zlib.crc32(c)==CRC:
            print(hex(i),hex(j))
            exit(0)
```

ctfshow{685082227bcf70d17d1b39a5c1195aa9}

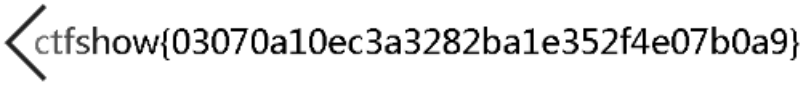
```
ctfshow{685082227bcf70d17d1b39a5c1195aa9}
```

misc33

提示：出题人丧心病狂，把高度也改了

和上题一模一样的脚本，爆破宽高

```
import zlib
import struct
image=open("misc33.png","rb").read()
for i in range(4096):
    for j in range(4096):
        c=image[12:16]+struct.pack('>i',i)+struct.pack('>i',j)+image[24:29]
        CRC=0x5255A798
        if zlib.crc32(c)==CRC:
            print(hex(i),hex(j))
            exit(0)
```



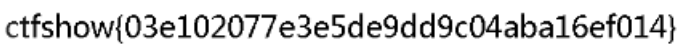

```
ctfshow{03070a10ec3a3282ba1e352f4e07b0a9}
```

misc34

提示：出题人狗急跳墙，把IHDR块的CRC也改了，但我们知道正确宽度肯定大于900

由于CRC也被修改了，所以我们对宽度进行爆破，然后自己找图片

```
import zlib
import struct
image=open("misc34.png","rb").read()
width=image[16:20]
height=image[20:24]
for i in range(900,1200):
   image1=open(str(i-900)+'.png','wb')
   change=image[:16]+struct.pack('>i',i)+image[20:]
   image1.write(change)
   image1.close()
```




```
ctfshow{03e102077e3e5de9dd9c04aba16ef014}
```

misc35

提示：出题人负隅顽抗，但我们知道正确宽度肯定大于900

先改高度为\x02\x96，

```
import zlib
import struct
image=open("misc35.png","rb").read()
for i in range(900,1200):
   image1=open(str(i-900)+'.png','wb')
   change=image[:image.index(b'\x02\x96\x03\x84')+2]+struct.pack('>h',i)+image[image.index(b'\x02\x96\x03\x84')+
4:]
   image1.write(change)
   image1.close()
```

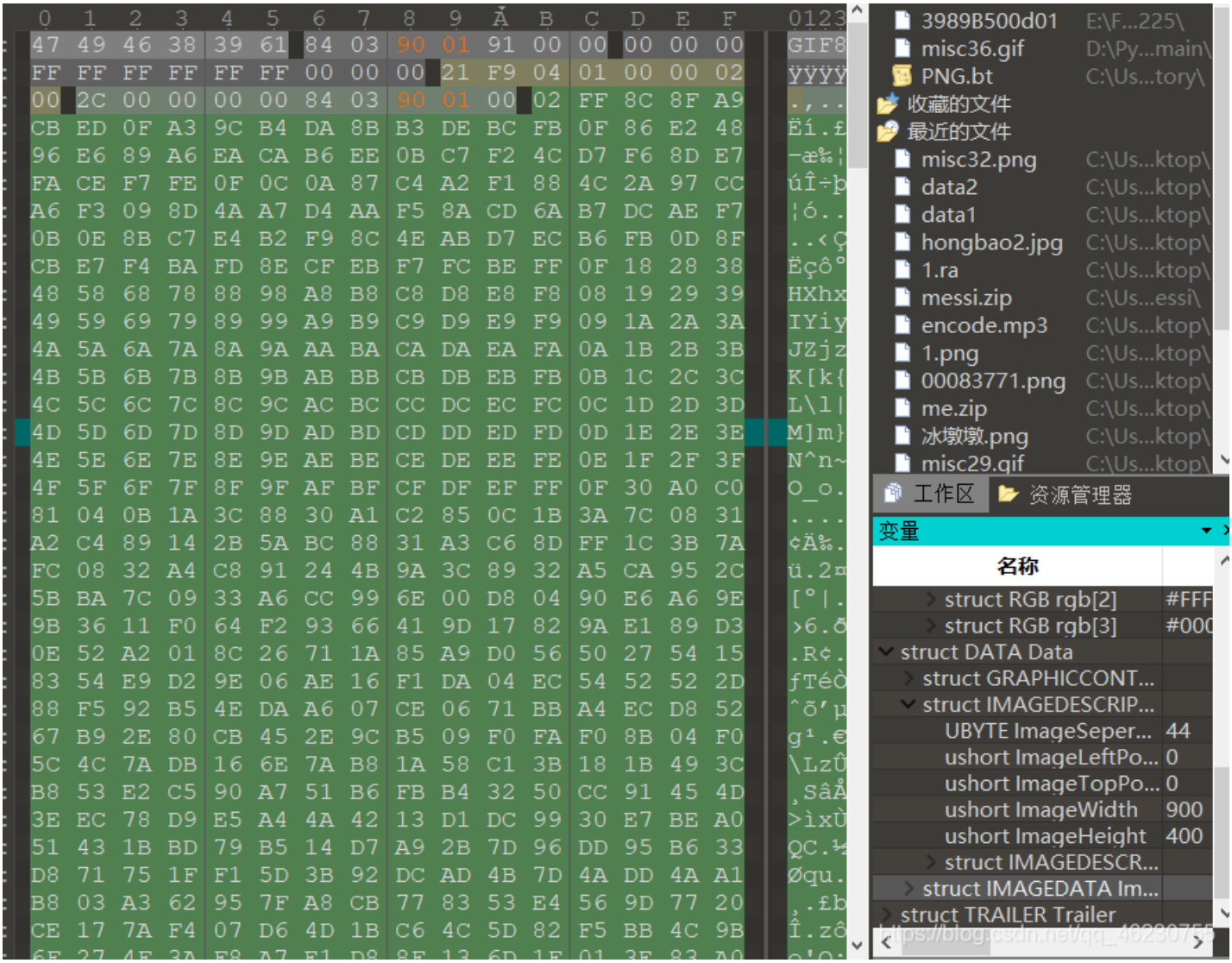
ctfshow{ca35201ca9ed607e5a68f44ef573fbc3}

```
ctfshow{ca35201ca9ed607e5a68f44ef573fbc3}
```

misc36

提示：出题人坦白从宽，正确的宽度在920-950之间

先把gif的高度改了，记得要改两处。



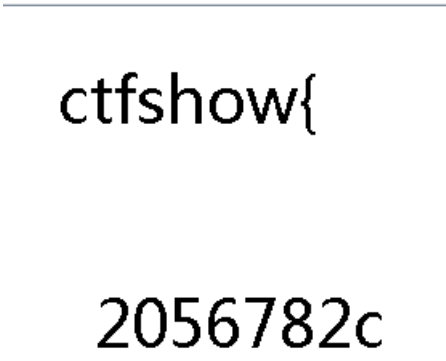
```
import zlib
import struct
image=open("misc36.gif","rb").read()
for i in range(920,950):
   image1=open(str(i-900)+'.gif','wb')
   change=image[:image.index(b'\x00\x00\x84\x03\x90\x01')+2]+struct.pack('>h',i)[::-1]+image[image.index(b'\x00\
x00\x84\x03\x90\x01')+4:]
   image1.write(change)
   image1.close()
```

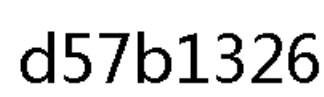

```
ctfshow{1ebf739f832906d60f57436b8179166f}
```

misc37

提示：`flag在图片里`

flag存在其中的几帧，直接看就可以。

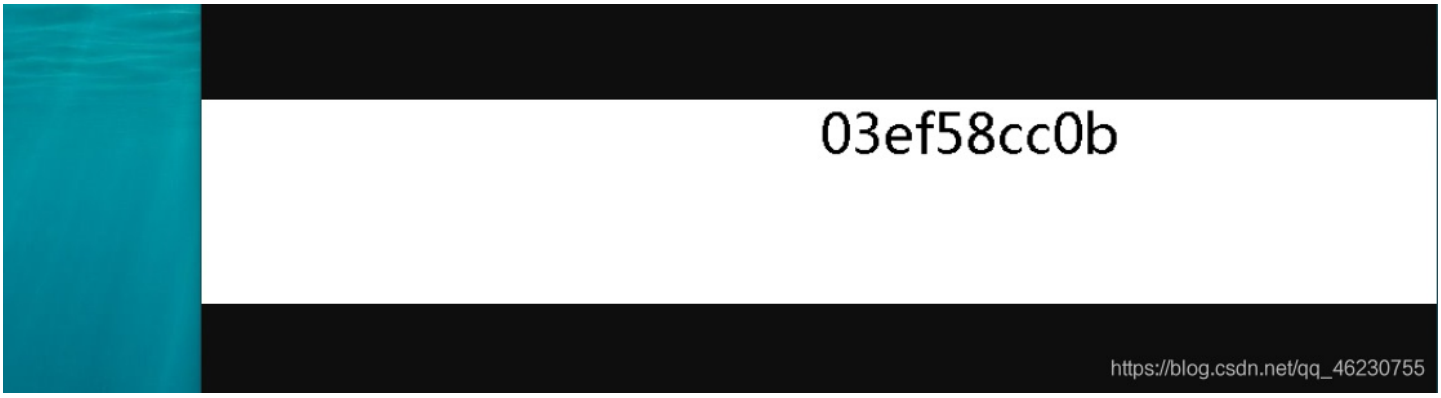





```
ctfshow{2056782cd57b13261dcbbe3d6eecda17}
```

misc38

提示：`flag在图片里`
是一个动态的png



用工具`APNG Disassembler`分离（下载地址）

然后找就行

```
ctfshow{48b722b570c603ef58cc0b83bbf7680d}
```

misc39

提示：`flag就像水，忽快忽慢地流`

看了v3□ 的博客发现是我没见过的gif帧数间隔隐写，kali里用命令`identify -format "%T " misc39.gif > misc39.txt`提取帧数

```
37 37 36 36 36 37 37 37 37 37 36 37 36 36 37 37 36 36 37 37 36 37 37 37 36 36 37 37 37 37 36 37 36 36 36 37 37 36
37 37 37 37 37 37 37 36 37 37 37 37 37 37 37 36 37 37 36 37 37 36 37 36 37 36 37 37 36 36 37 36 36 36 37 37 37 36 36
36 36 37 37 36 36 36 37 36 37 37 36 36 37 36 37 37 36 36 37 37 36 37 37 36 36 37 37 36 36 37 37 37 36 36 37 36 37
37 37 36 36 37 36 37 37 36 37 36 37 37 37 36 36 37 37 36 37 37 36 36 36 37 36 36 37 37 36 37 37 37 37 37 36 36 36
37 36 37 37 36 36 37 36 37 36 37 37 36 36 37 36 36 37 37 36 37 37 36 36 37 37 37 36 36 36 37 37 36 36 37 36 36 36
37 37 37 36 36 37 36 37 37 36 37 37 36 36 37 37 36 36 37 37 37 37 36 36 36 37 36 37 37 37 36 36 37 37 37 37 36 36
37 36 37 37 37 36 36 36 37 36 37 37 36 36 36 37 37 37 37 37 36 36 36 37 36 37 37 36 36 36 36 36 37 37 36 37 36 36
36 37 37 36 37 36 37 36 37 37 37 36 36 37 37 37 37 37 37 37 36 37
```

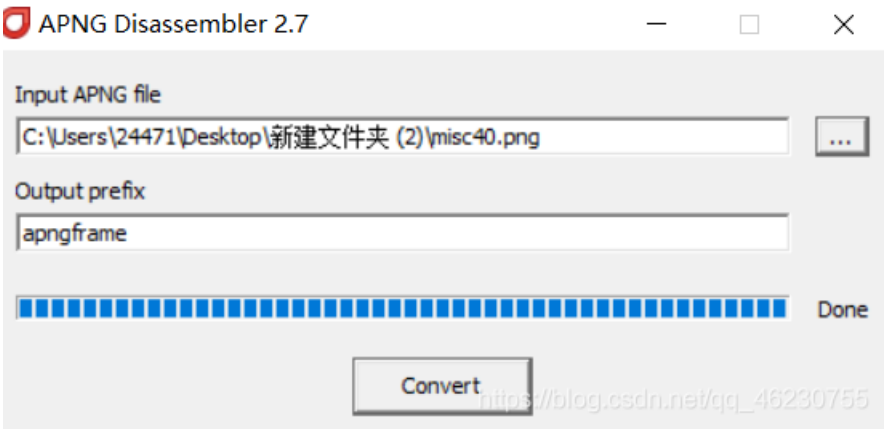
脚本每七位转01在转字符。为什么7位应该是对应最大128把

```
str = '37 37 36 36 36 37 37 37 37 37 36 37 36 36 37 37 36 36 37 37 36 37 37 37 36 36 37 37 37 37 36 37 36 36 36
37 37 36 37 37 37 37 37 37 37 36 37 37 37 37 37 37 37 36 37 37 36 37 37 36 37 36 37 36 37 37 36 36 37 36 36 37 3
7 37 36 36 36 36 37 37 36 36 36 37 36 37 37 36 36 37 36 37 37 36 36 37 37 36 37 37 36 36 37 37 36 36 37 37 37 36
 36 37 36 37 37 37 36 36 37 36 37 37 36 37 36 37 37 37 36 36 37 37 36 37 37 36 36 36 37 36 36 37 37 36 37 37 37
37 37 36 36 36 37 36 37 37 36 36 37 36 37 36 37 37 36 36 37 36 36 37 37 36 37 37 36 36 37 37 37 36 36 36 37 37 3
6 36 37 36 36 36 36 37 37 37 36 36 37 36 37 37 36 37 37 36 36 37 37 36 36 37 37 37 37 36 36 36 36 37 36 37 37 37 36
 36 37 37 37 36 36 37 36 37 37 37 36 36 36 37 36 37 37 36 36 36 37 37 37 37 36 36 36 36 37 36 37 37 36 36 36 36
36 37 37 36 37 36 36 36 37 37 36 37 36 37 36 37 37 37 36 36 37 37 37 37 37 37 36 37'
str=str.replace('37','1').replace('36','0').replace(' ','')
for i in range(len(str)//7):
    try:
        print(chr(int(str[i*7:(1+i)*7],2)),end='')
    except:
        pass
```


```
ctfshow{52812ff995fb7be268d963a9ebca0459}
```

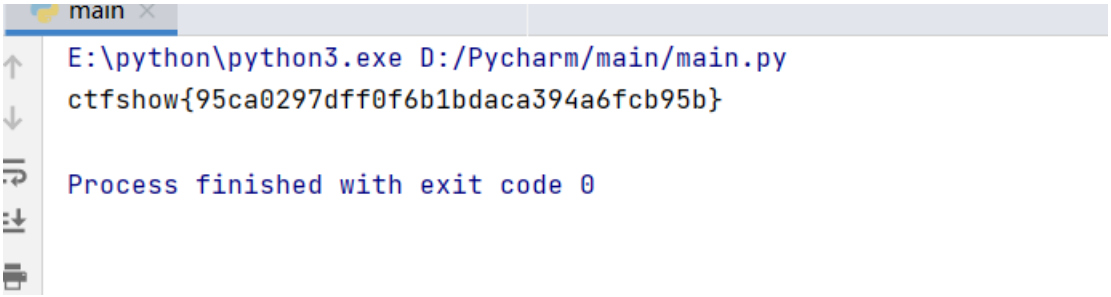
misc40

提示：`flag就像歌，有长有短仿佛岁月悠悠`
一样用到上题的软件



读取每一个txt文件然后去掉干扰从28开始。

```
flag=''
for i in range(28,69):
    f= open('C:/Users/24471/Desktop/新建文件夹 (2)/apngframe'+str(i)+'.txt')
    j=f.read()
    flag+=chr(int(j.split('/')[0].split('=')[1]))
print(flag)
```

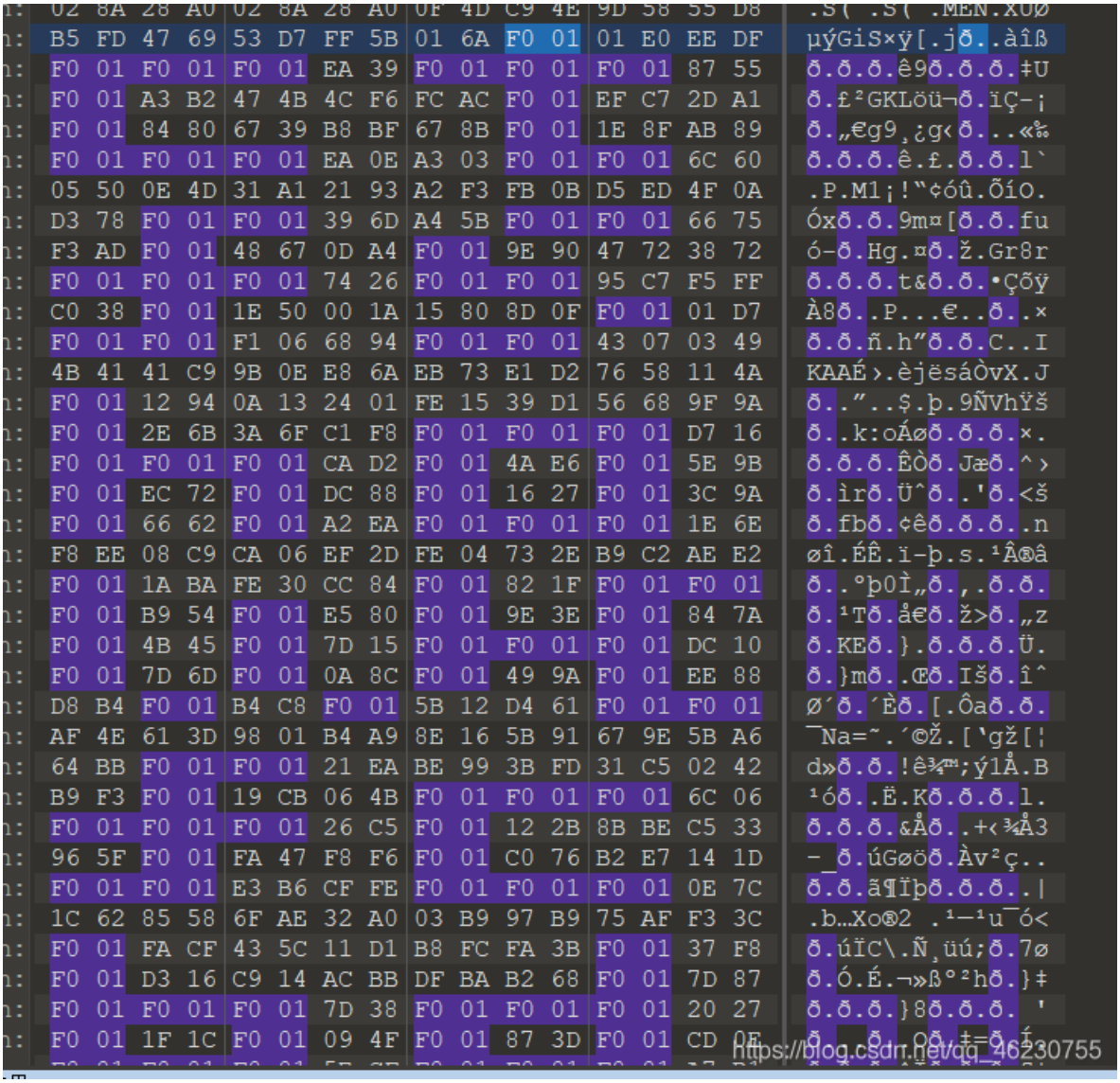



```
ctfshow{95ca0297dff0f6b1bdaca394a6fcb95b}
```

misc41

提示：`H4ppy Apr1l F001's D4y! 愚人节到了，一群笨蛋往南飞，一会儿排成S字，一会儿排成B字。`

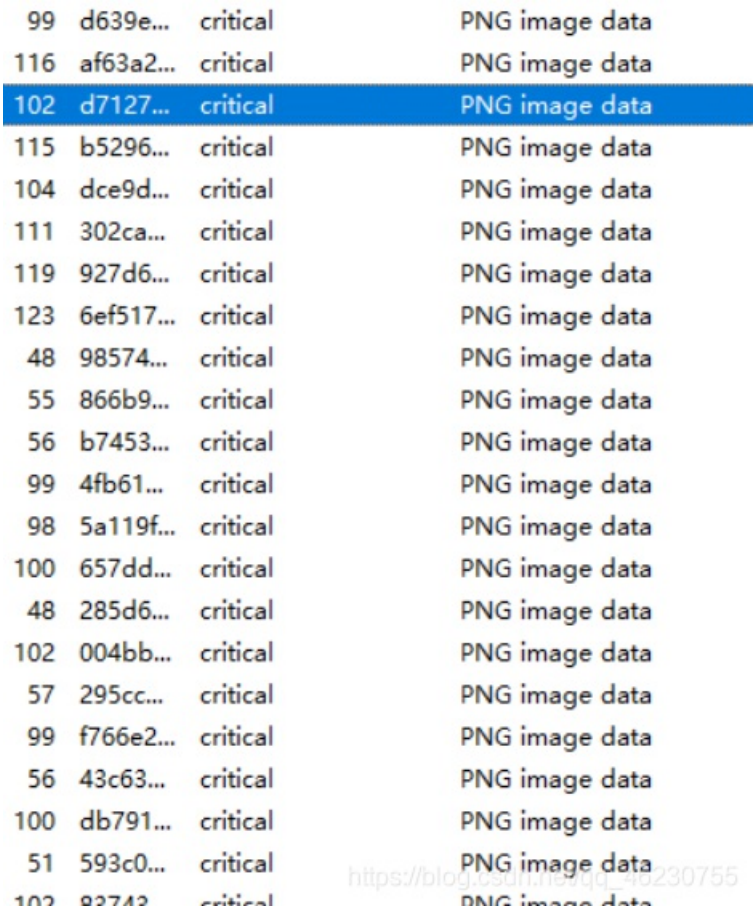
在010里面搜索F0 01 全部高亮




```
ctfshow{fcbd427caf4a52f1147ab44346cd1cdd}
```

misc42

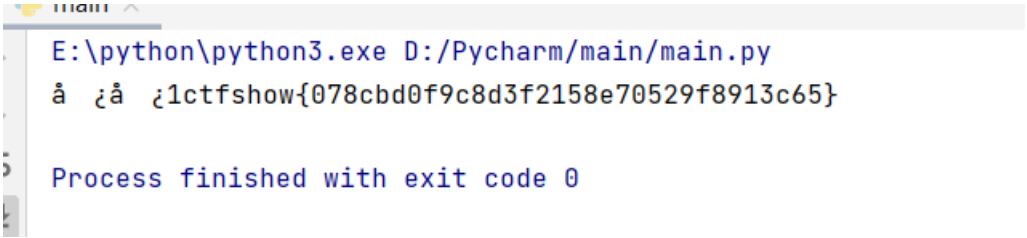
提示：`flag有多长？2cm……不好意思打错了，41位`
我猜这题在黑套宝。



99	d639e...	critical	PNG image data
116	af63a2...	critical	PNG image data
102	d7127...	critical	PNG image data
115	b5296...	critical	PNG image data
104	dce9d...	critical	PNG image data
111	302ca...	critical	PNG image data
119	927d6...	critical	PNG image data
123	6ef517...	critical	PNG image data
48	98574...	critical	PNG image data
55	866b9...	critical	PNG image data
56	b7453...	critical	PNG image data
99	4fb61...	critical	PNG image data
98	5a119f...	critical	PNG image data
100	657dd...	critical	PNG image data
48	285d6...	critical	PNG image data
102	004bb...	critical	PNG image data
57	295cc...	critical	PNG image data
99	f766e2...	critical	PNG image data
56	43c63...	critical	PNG image data
100	db791...	critical	PNG image data
51	593c0...	critical	PNG image data

放入tweakpng然后进行长度的提取，批量转

```
str=[229,152,191,229,152,191,49,99,116,102,115,104,111,119,123,48,55,56,99,98,100,48,102,57,99,56,100,51,102,50,
49,53,56,101,55,48,53,50,57,102,56,57,49,51,99,54,53,125]
flag=''
for i in str:
    flag+=chr(i)
print(flag)
```



```
E:\python\python3.exe D:/Pycharm/main/main.py
å ¿å ¿1ctfshow{078cbd0f9c8d3f2158e70529f8913c65}

Process finished with exit code 0
```

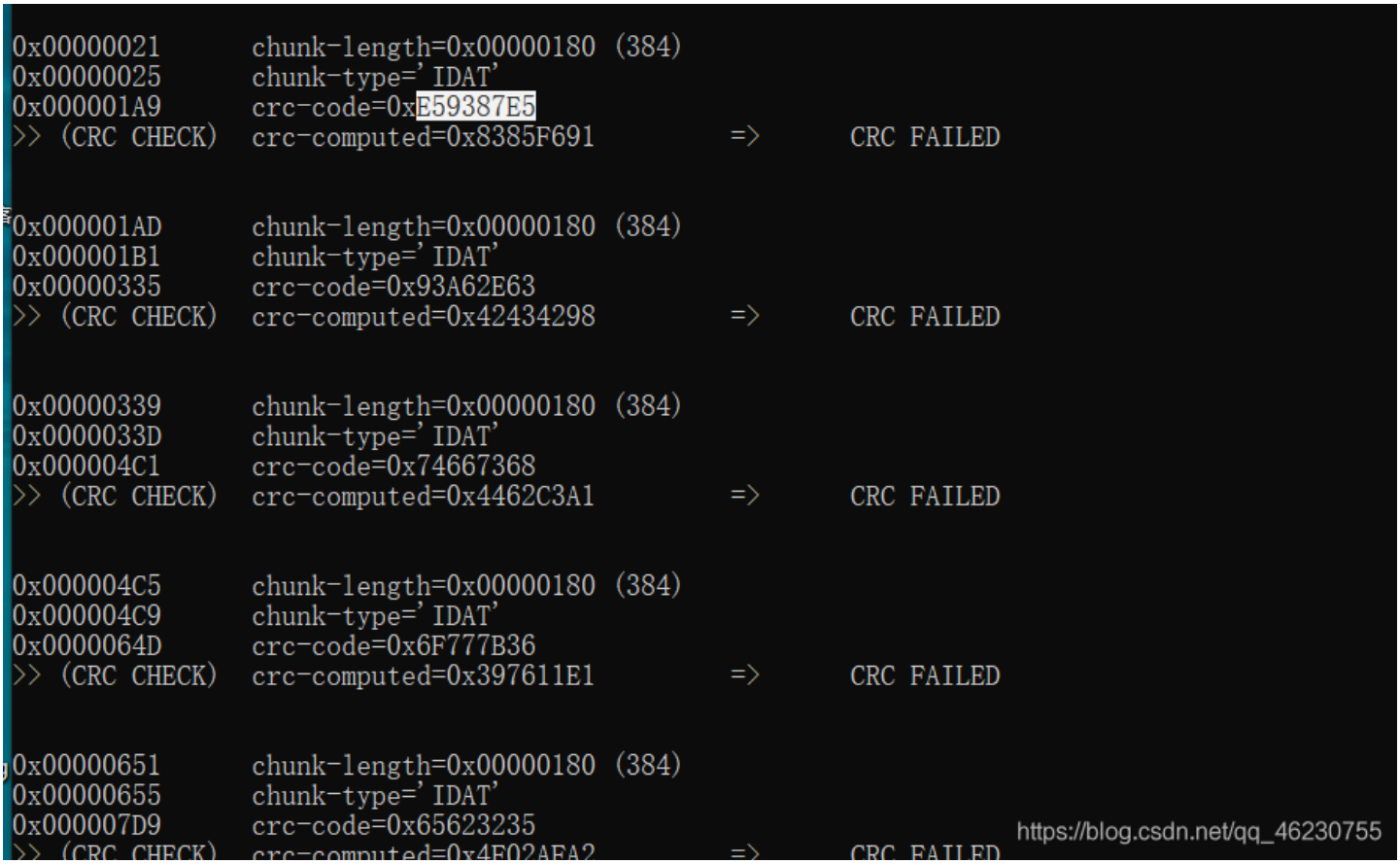
```
ctfshow{078cbd0f9c8d3f2158e70529f8913c65}
```

misc43

提示：错误中隐藏着通往正确答案的道路

猜测与crc错误有关。

利用pngdebuger提取



把错误的code提出来用hex转字符

```
import binascii
str='E59387E593A62E63746673686F777B3665623235383966666666635653339306665366238373530346462633038393270'
print(binascii.a2b_hex(str))
```

```
E:\python\python3.exe D:/Pycharm/main/main.py
b'\xe5\x93\x87\xe5\x93\xa6.ctfshow{6eb2589ffff5e390fe6b87504dbc0892}'
```

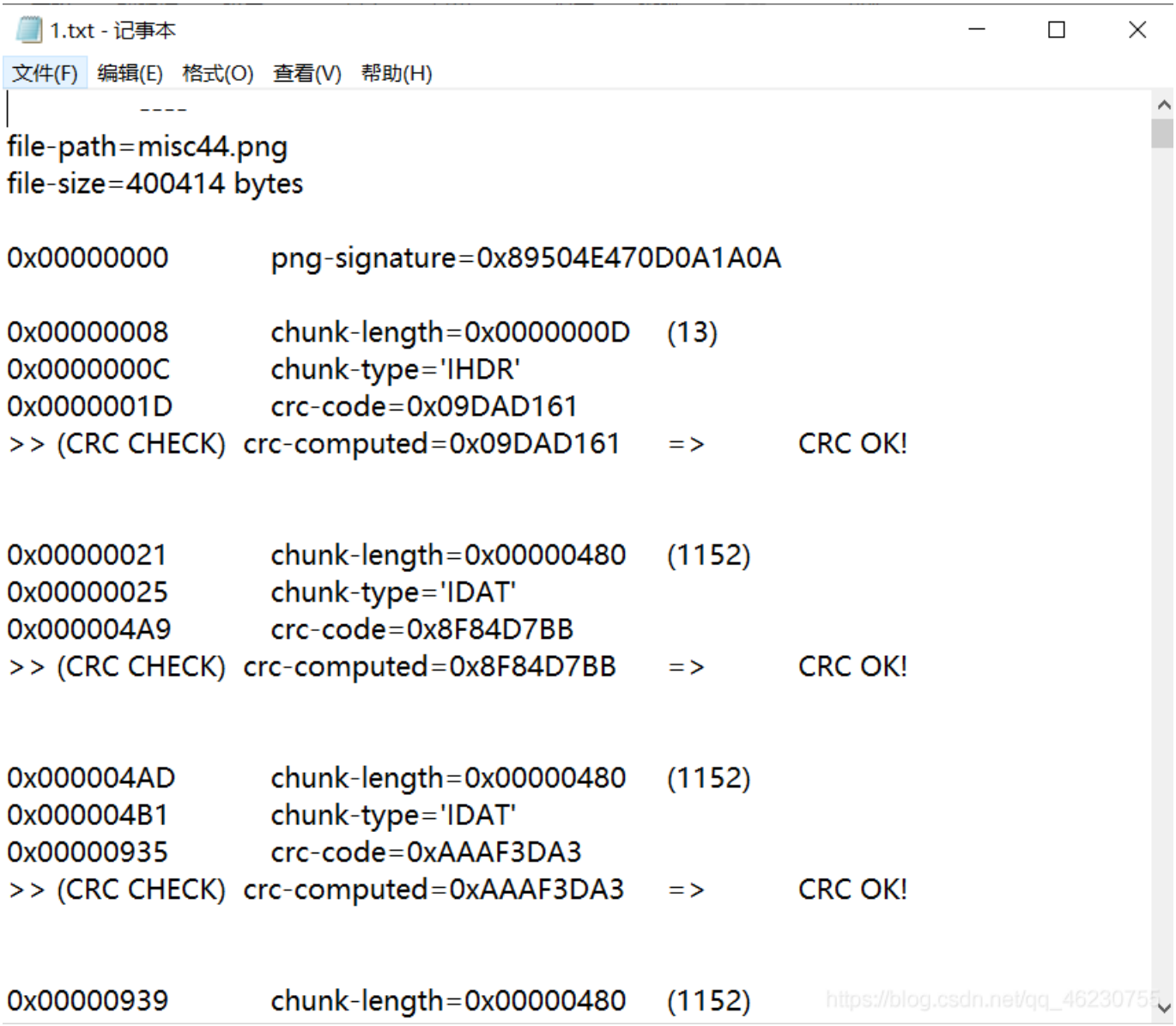
```
ctfshow{6eb2589ffff5e390fe6b87504dbc0892}
```

misc44

提示：错误中还隐藏着坑

利用pngdebugger进行解析

```
PNGDebugger.exe misc44.png >1.txt
```



提取正确的CRC OK!为1，错误的CRC FAILED为0，删除无用的前10行和后四行。

```
f=open("1.txt")
str=''
while 1:
    s=f.readline()
    if s:
        if 'OK!' in s:
            str+='1'
        elif 'FAILED' in s:
            str+='0'
    else:
        break
print(str)
print(len((str)))
for i in range(len(str)//8):
    print(chr(int(str[i*8:(i+1)*8],2)),end="")
```

```
E:\python\python.exe D:/Pycharm/main/main.py
1111111111111111011000110111010001100110011100110110100001101111011101110113
44
ÿÿctfshow{cc1af32bf96308fc1263231be783f69e}
进程已结束，退出代码为 0
```

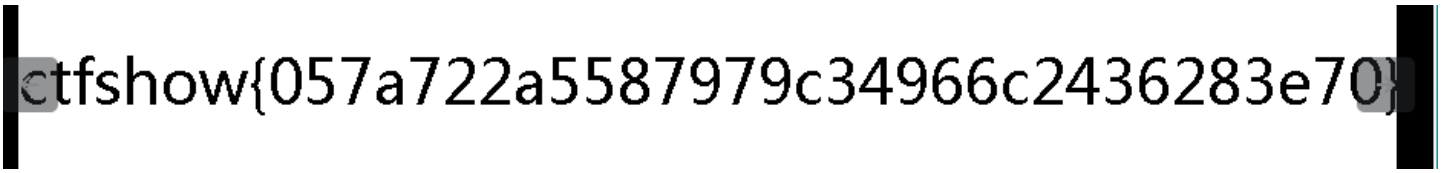
```
ctfshow{cc1af32bf96308fc1263231be783f69e}
```

misc45

提示：有时候也需要换一换思维格式

因为bmp与png的读取像素的方式不一样，所以我们可以将其先进行转换格式

然后`binwalk misc45 -e`




```
ctfshow{057a722a5587979c34966c2436283e70}
```

misc46

提示：你见过扶乩吗

利用identify进行提取

```
identify misc46.gif > 1.txt
```

再利用一个画图的模块画图。

根据的是每一帧的偏移坐标

以下脚本参考V3师傅博客：

https://blog.csdn.net/xczzhf/article/details/115434194?spm=1001.2014.3001.5501

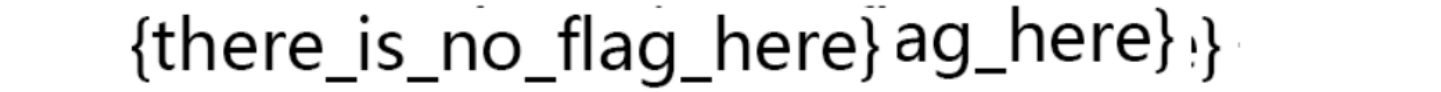
```
from PIL import Image
import matplotlib.pyplot as plt
f=open('1.txt')
pp=[]
while 1:
    c=f.readline()
    if c:
        s=eval(c.split('+')[1]+','+c.split('+')[2][:2])
        pp.append(s)
    else:
        break
img =Image.new('RGB',(400,70),(255,255,255))
for i in pp:
    new = Image.new('RGB',(1,1),(0,0,0))
    img.paste(new,i)
img.save('1.png')
```

ctfshow{05906b3be8742a13a93898186bc5802f}

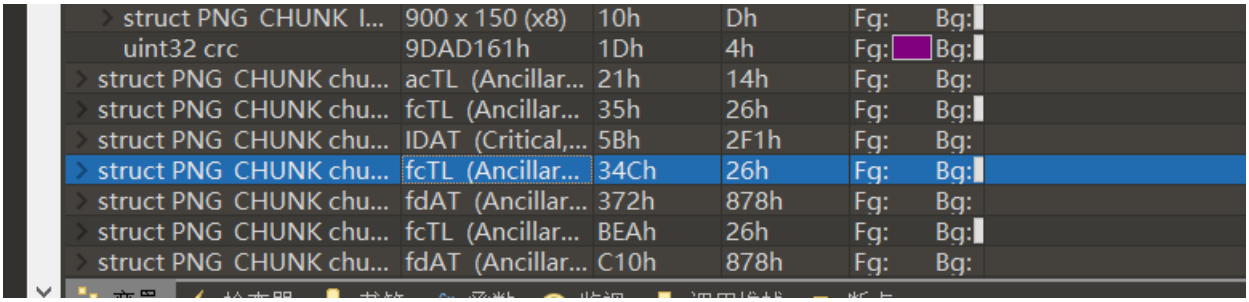
```
ctfshow{05906b3be8742a13a93898186bc5802f}
```

misc47

提示：没见过扶乩，那你知道笔仙吗
是apng文件，利用浏览器打开



看别的师傅博客发现需要提取其偏移位。
然后编写脚本进行提取。



struct PNG_CHUNK_I...	900 x 150 (x8)	10h	Dh	Fg: Bg:
uint32 crc	9DAD161h	1Dh	4h	Fg: Bg:
struct PNG_CHUNK chu...	acTL (Ancillar...	21h	14h	Fg: Bg:
struct PNG_CHUNK chu...	fcTL (Ancillar...	35h	26h	Fg: Bg:
struct PNG_CHUNK chu...	IDAT (Critical,...	5Bh	2F1h	Fg: Bg:
struct PNG_CHUNK chu...	fcTL (Ancillar...	34Ch	26h	Fg: Bg:
struct PNG_CHUNK chu...	fdAT (Ancillar...	372h	878h	Fg: Bg:
struct PNG_CHUNK chu...	fcTL (Ancillar...	BEAh	26h	Fg: Bg:
struct PNG_CHUNK chu...	fdAT (Ancillar...	C10h	878h	Fg: Bg:

v3师傅博客

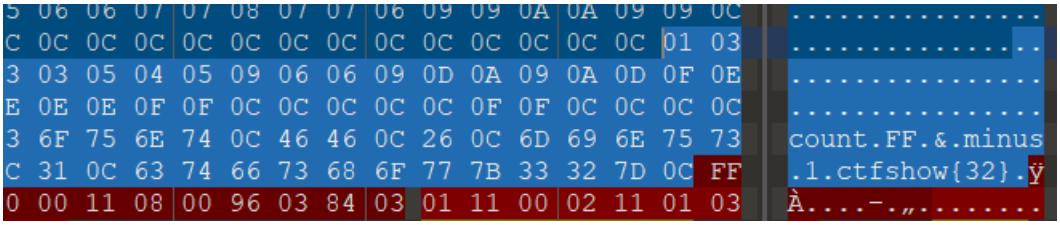
```
import struct
from PIL import Image
import matplotlib.pyplot as plt
f=open('misc47.png','rb')
c=f.read()
c=c[c.index(bytes.fromhex('6663544C00000001')):]
pp=[]
for i in range(1,1124,2):
    start=c.index(bytes.fromhex('6663544C0000')+struct.pack('>h',i))
    fc=c[start:start+30]
    pp.append(struct.unpack('>h',fc[18:20])+struct.unpack('>h',fc[22:24]))

img =Image.new('RGB',(400,70),(255,255,255))
for i in pp:
    new = Image.new('RGB',(1,1),(0,0,0))
    img.paste(new,i)
img.save('1.png')
```

ctfshow{6d51f85b45a0061754a2776a32cf26c4}

```
ctfshow{6d51f85b45a0061754a2776a32cf26c4}
```

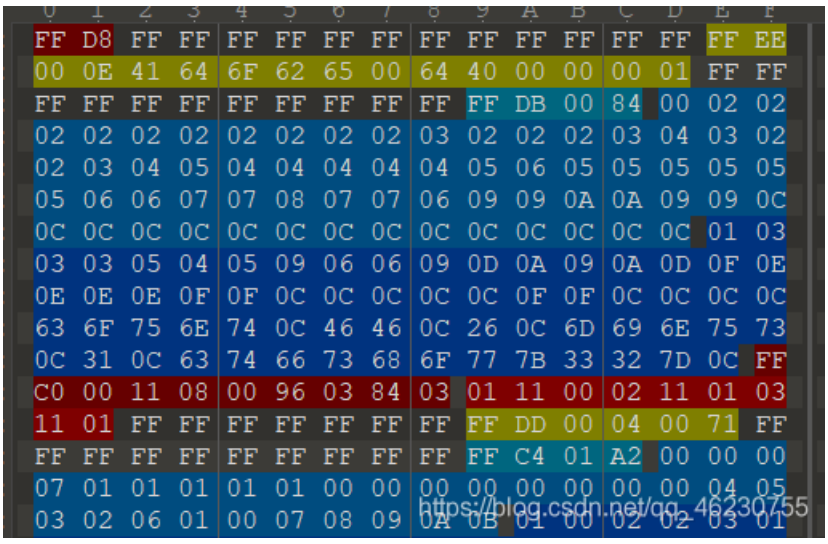
misc48

提示：`附件的第（Di）七（Qi）题（Ti）中有提示。本题略脑洞，可跳过`



提示我们计算FF的数量。

> 八神师傅说是统计每两个有意义块之间的FF的数量再减一，因为每两个有意义块之间插入数据好像是不太影响的，至于要减一是因为这段中的最后一个FF是下一个有意义块的开头，取前32个段即可，会发现每个段长度都不超过16，所以直接转为16进制就是flag



手动算一下得到

```
0 12 11 0 7 10 13 13 9 0 9 13 0 13 6 0 10 9 2 1 0 1 10 8 11 5 12 7 2 2 3 10
```

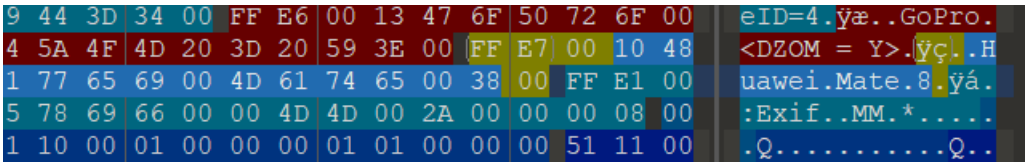
```
s=[0,12,11,0,7,10,13,13,9,0,9,13,0,13,6,0,10,9,2,1,0,1,10,8,11,5,12,7,2,2,3,10]
f='0123456789abcdef'
flag='ctfshow{'
for i in range(len(s)):
    flag+=f[s[i]]
flag+='}'
print(flag)
```

```
ctfshow{0cb07add909d0d60a92101a8b5c7223a}
```

misc49

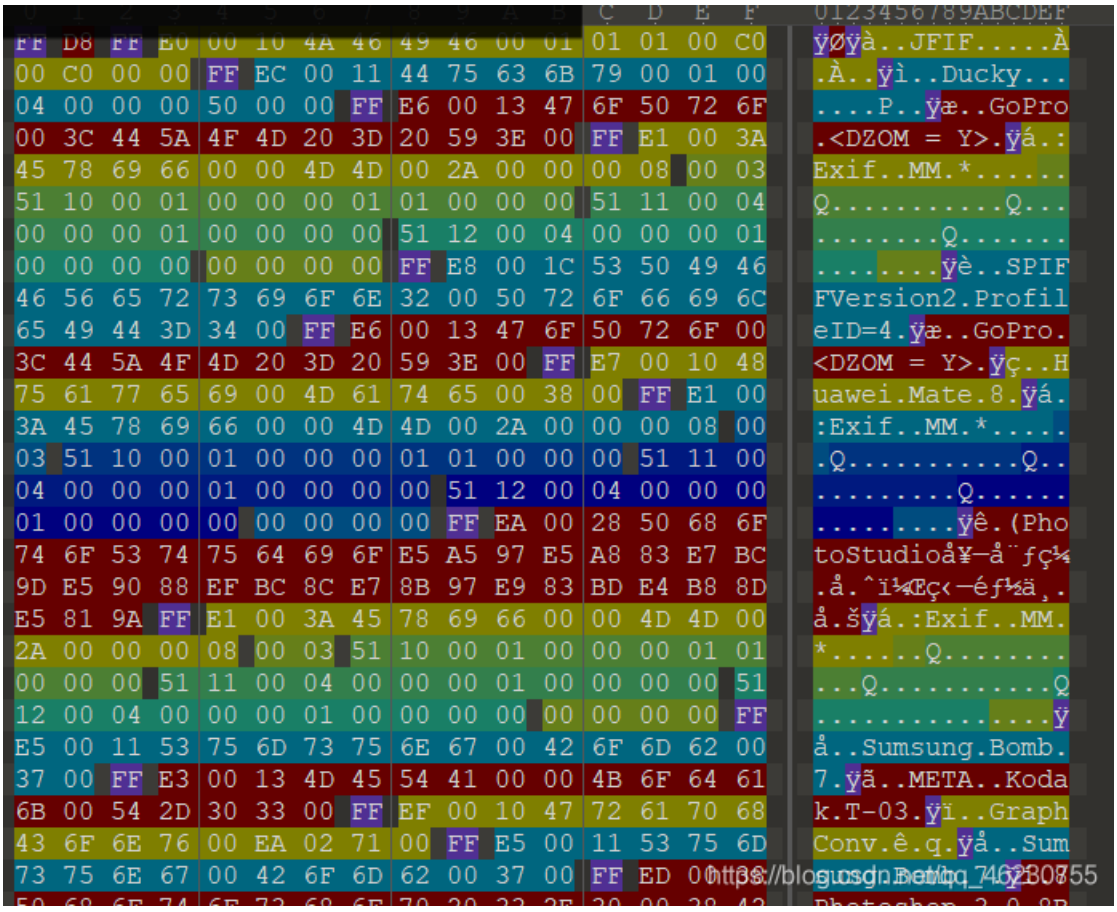
提示：它们一来就是十六种。本题略脑洞，可跳过

丢入010



看到一堆的设备

> 问了下八神师傅，原来E后面的那位就是flag，最后按照这些块的顺序把E后面那位合起来就行了，最后如下




```
ctfshow{0c618671a153f5da3948fdb2a2238e44}
```

参考大量的V3□博客

v3□