第四届“强网”拟态防御国际精英挑战赛MISC 部分复现

原创

小蓝同学` 已于 2022-04-08 23:31:34 修改 518 收藏 1

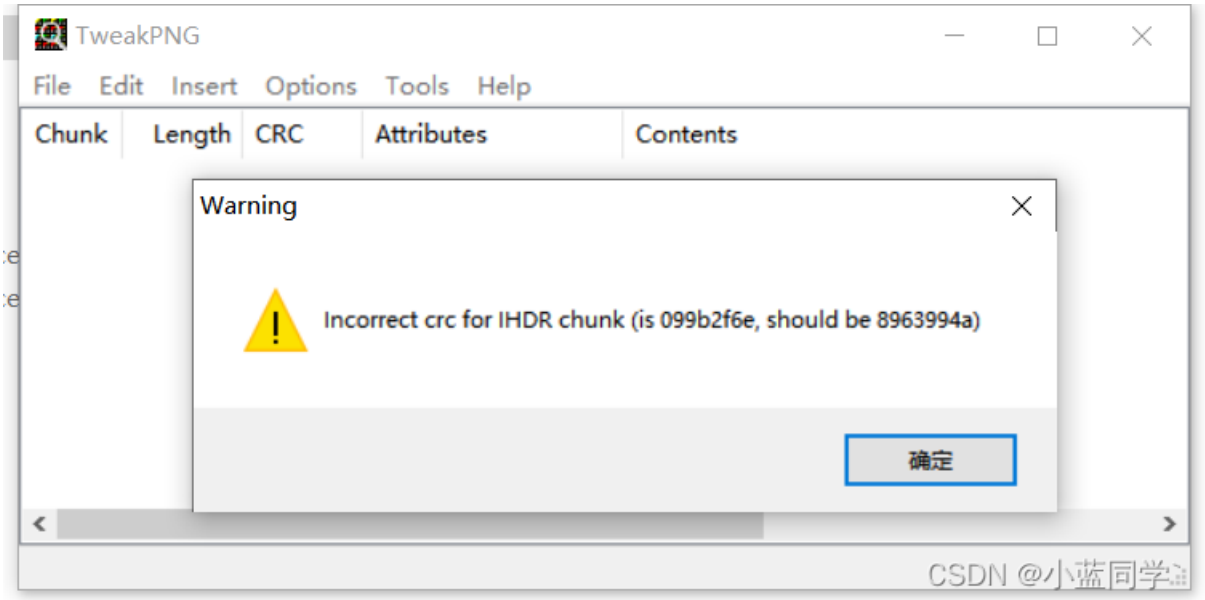
分类专栏： MISC 文章标签： 拟态强网 MISC

于 2022-04-04 17:10:36 首次发布

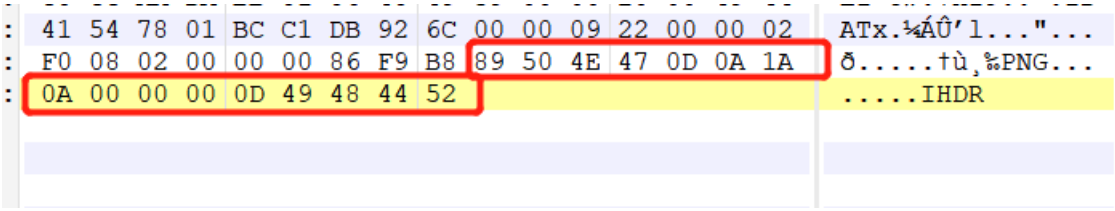
强网拟态复现 MISC

mirror

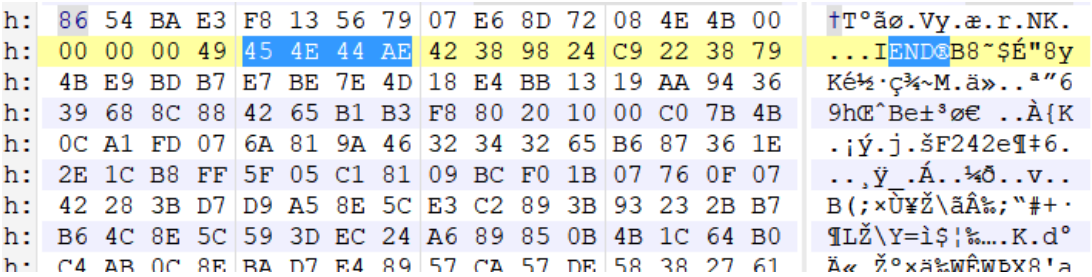
开局给一张图片，使用TweakPNG打开发现CRC32报错，估计是宽高的问题，这里等会修复。



然后继续将文件拉到文件末尾，发现似乎有额外的图片数据。



这里按照题目意思应该是要刚好一行的，不知道为什么我这里在了不同的一行，这里刚好16个数据，按照题目意思mirror镜像的意思，应该是藏着一段PNG的16进制数据的镜像数据。然后我们继续查看文件尾的位置。



确实符合了我们的猜想，那么我们需要提取该数据出来到新的文件当中去，并重新命名为full1.png，然后使用脚本还原，还原思路就是将所有的16进制数据逆序，然后在16个字节进行逆序即可还原到之前正确的文件。并且把上面的那一个图片也进行分离出来（命名为newfull.png）这里个思路也比较明显 不然带着冗余数据根本就做不了别的动作。

```
# 逆序的脚本
from binascii import *

with open('full1.png', 'rb') as f:
 with open('res-full.png', 'wb') as f1:
  hex_data = hexlify(f.read()).decode()[::-1]
  for i in range(0, len(hex_data), 32):
   data = hex_data[i:i+32][::-1]
   f1.write(unhexlify(data))
```

得到正确的图片，然后使用TweakPNG打开发现CRC32报错，然后这里直接使用脚本改正。（把newfull.png和res-full.png都进行改正）

```
# 修改宽高脚本
import binascii
import struct
import sys

file = 'full.png'
fr = open(file,'rb').read()
data = bytearray(fr[0x0c:0x1d])
crc32key = eval('0x'+str(binascii.b2a_hex(fr[0x1d:0x21]))[2:-1])
#原来的代码: crc32key = eval(str(fr[29:33]).replace('\\x','').replace("b'",'0x').replace("'",''))
n = 4095
for w in range(n):
    width = bytearray(struct.pack('>i', w))
    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]
        crc32result = binascii.crc32(data) & 0xffffffff
        if crc32result == crc32key:
            print(width,height)
            newpic = bytearray(fr)
            for x in range(4):
                newpic[x+16] = width[x]
                newpic[x+20] = height[x]
            fw = open(file+'.png','wb')
            fw.write(newpic)
            fw.close
            sys.exit()
```

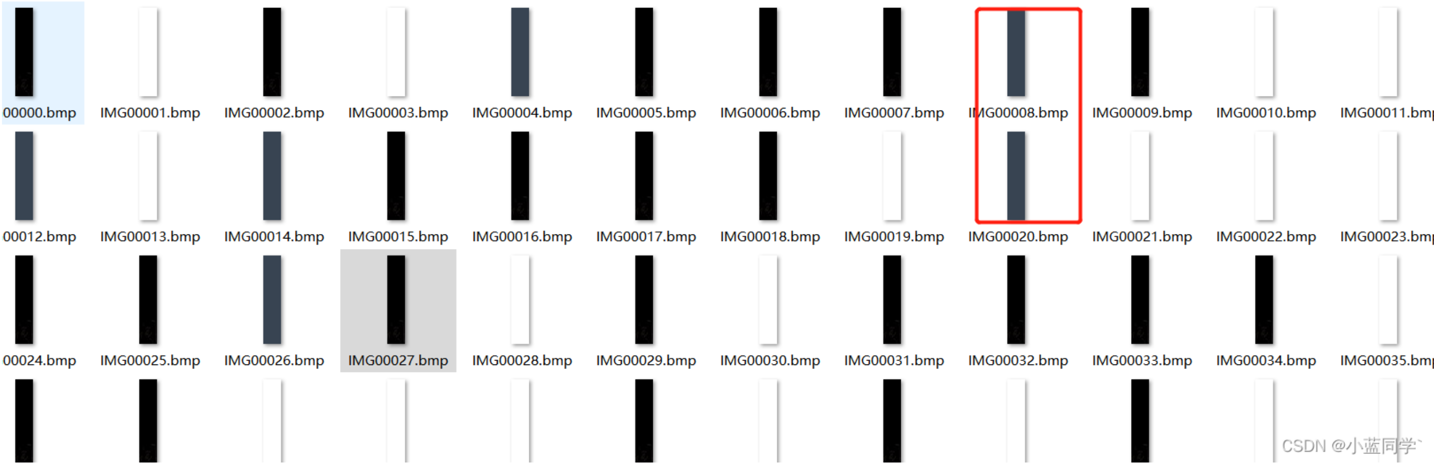
还原之后，使用stegSlove打开发现有盲水印特征，直接脚本梭哈然后是数字的排序是镜像的逆转，最后根据题目的替换得到结果。

```
flag='32effq8aa8374a02a9p1636ae8901qa0'

dict1 = {'2':'5','5':'2','e':'6','6':'e','9':'a','a':'9','p':'b','b':'p','q':'d','d':'q'}
flag_list = list(flag)

idx = 0
for char in flag_list:
 for key in dict1:
  if char == key:
   flag_list[idx] = dict1[key]
 idx += 1
res_flag = ''
for i in flag_list:
 res_flag += i
print('flag{{{}}}'.format(res_flag))
```

`flag{356ffd89983749059ab1e3e968a01d90}`

Bar

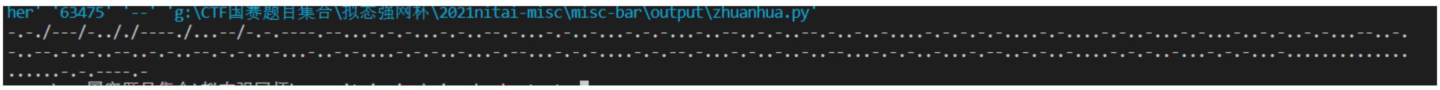
下载来发现是一个动图，使用gif分离工具进行动态图的分离，分离出334张图片，仔细查看发现有黑白灰三种图片，猜测黑为"-“，白为”."，灰色为‘/’



所以先转化为莫斯码看一下。

```
from PIL import Image
import os

for i in os.listdir():
    if(i.split(".")[1] == "png"):
        p = Image.open(i).convert("L")
        a,b=p.size
        for x in range(a):
            for y in range(b):
                if p.getpixel((x,y)) == 0:
                    print("-",end="")
                    break
                elif p.getpixel((x,y)) == 255:
                    print(".",end="")
                    break
                else:
                    print(" ",end="")
                    break
            break
```

得结果：




```
-.-./---/-../ ./----./...--/-.-.----.--...-.-.-...-.-..--.-..--.-...-.-..-...-.-.-...-..--..-.-..--.-..-..-....-.-.-.-.
-....-.-....-.-..-...-.-...-..-.-..-.-...--..-.-..--.-..-...-..-..--..-..-..--.-.-...-...-..-.-....-.-.-..-...-.--...-.-
.-....-.-.--.-...-.-..-..-..--...-.-.-..-...-.--..-.-..-...-..-.....-.-.-..-...-.-.-...-.................-.-.----.-
```

将前几张带有分隔符的进行莫斯解码，得到code93

那么可以猜测后面的图片应该就是code93相关的编码，查看相关文档code93为9位的0和1组成一组对应一个字符，所以我们这里进行代码编写转化一下。

```
from PIL import Image
import os


num = 0
for i in os.listdir():
    if(i.split(".")[1] == "bmp" and int(i.split(".")[0][3:] )>= 27):
        num = num +1
        p = Image.open(i).convert("L")
        a,b=p.size
        str = ''
        for x in range(a):
            for y in range(b):
                # print(p.getpixel((x,y)))
                if p.getpixel((x,y)) == 0:
                    print("1",end="")
                    break
                elif p.getpixel((x,y)) == 255:
                    print("0",end="")
                    break
            break
        if num%9 == 0:
            print(" ")
```

代码结果：

```
101011110
110001010
100010100
110100010
100100010
101000100
110010100
110100100
100001010
101010000
101000010
100100010
100010010
100101000
110010100
110100100
110010100
110101000
100010010
100001010
100100010
110001010
100001010
110100010
100100100
110001010
100100010
110010100
100001010
100100010
101000100
000000000
000000000
101011110
1
```

得到结果后转化为对应的字符：参考转化来链接:条码规范—Code 93

```
from enum import Flag
from PIL import Image

string = ['100010100', '101001000','101000100', '101000010', '100101000',
'100100100', '100100010', '101010000','100010010', '100001010',
'110101000', '110100100', '110100010', '110010100','110010010',
'110001010', '101101000', '101100100', '101100010', '100110100',
'100011010', '101011000', '101001100', '101000110', '100101100',
'100010110','110110100', '110110010', '110101100', '110100110',
'110010110', '110011010','101101100', '101100110', '100110110',
'100111010', '100101110']

number = ['0', '1', '2', '3', '4', '5','6', '7', '8', '9',
'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L',
'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X',
 'Y', 'Z', '-']
Flag = ''
data = "101011110 110001010 100010100 110100010 100100010 101000100 110010100 110100100 100001010 101010000 1010
00010 100100010 100010010 100101000 110010100 110100100 110010100 110101000 100010010 100001010 100100010 110001
010 100001010 110100010 100100100 110001010 100100010 110010100 100001010 100100010 101000100 000000000 00000000
0 101011110 1"
print(len(data.split(" ")))
for i in data.split(" "):
    if i in string:
        flag = number[string.index(i)]
        print(flag.lower(),end="")
        Flag+=flag
    else:
        pass
print("\n",len(Flag))
```

得到结果：

```
35
f0c62db973684dbda896f9c5f6d962
 30
```

```
f0c62db973684dbda896f9c5f6d962
```

这个还不是完整的flag，因为CODE93在转化为数字的过程中会提前两位结束，所以上面的结果有两位是空的，所以要得到完整的flag 还需转化为类似条形码的图案，再进行编程转化为完整的flag。
在线网站：CODE93输出

使用脚本读取：

```
from PIL import Image
import os
p = Image.open('cnaidc.png').convert("L")
a,b=p.size

print(a,b)

num = 0
for x in range(0,a,2):
    num = num +1
    for y in range(b):
        if p.getpixel((x,y)) == 0:
            print("1",end="")
        elif p.getpixel((x,y)) == 255:
            print("0",end="")
        # print(p.getpixel((x,y)))
        break
    if num%9 == 0:
        print("")
```

得到数据：

```
101011110
100110010
110001010
100010100
100110010
110100010
100100010
101000100
100110010
110010100
100110010
110100100
100001010
101010000
101000010
100100010
100010010
100101000
100110010
110010100
100110010
110100100
100110010
110010100
100110010
110101000
100010010
100001010
100100010
100110010
110001010
100001010
100110010
110100010
100100100
100110010
110001010
100100010
100110010
110010100
100001010
100100010
101000100
110010110
101001100
101011110
1
```

最后那两个`110010110`、`101001100`为um，所以最终的flag:
`flag{f0c62db973684dbda896f9c5f6d962um}`

BlueWhale

流量中里找到个th1sIsThEpassw0rD，发现压缩包里也有个password.txt,长度还一样，直接明文攻击了，很快就解出来了，连密码也整出来了!2b$3&Ec 打开压缩包，图片lsb简简单单zsteg一下`flag{F1nallY_y0uve_f0unD_1t}`

WeirdPhoto

这道题算是比较简单的题目了，想法和思路都比较简单。使用TweakPNG打开图片后发现报错，使用脚本修改图片宽高，出现一行字符。直接脑洞栅栏密码得到一个类似密码的东西。

TIEWOFTHSAEOUI

ITNRBCOSHSTSAN

接着就纯靠猜不断尝试最后确认是栅栏密码（key=4）得到压缩包密码：`THISISTHEANSWERTOOBSFUCATION`
然后对rar文件解密，得到一个out，看出是pdf文件格式，只是缺少了头部，将前面四个00改为：25504446

pdf隐写，使用wbStego，密码还是解压缩包的密码，得到flag：
`flag{th1s_ls_thE_f1n4l_F14g_y0u_want}`