2021第四届浙江省大学生网络与信息安全竞赛预赛部分Writeup

原创

塞纳河畔的春水 于 2021-10-27 14:57:50 发布 43786 收藏 9
分类专栏： CTF_Writeup 文章标签： python 信息安全 pycharm 图像处理 网络安全

本文链接：https://blog.csdn.net/qq_42815161/article/details/120992830

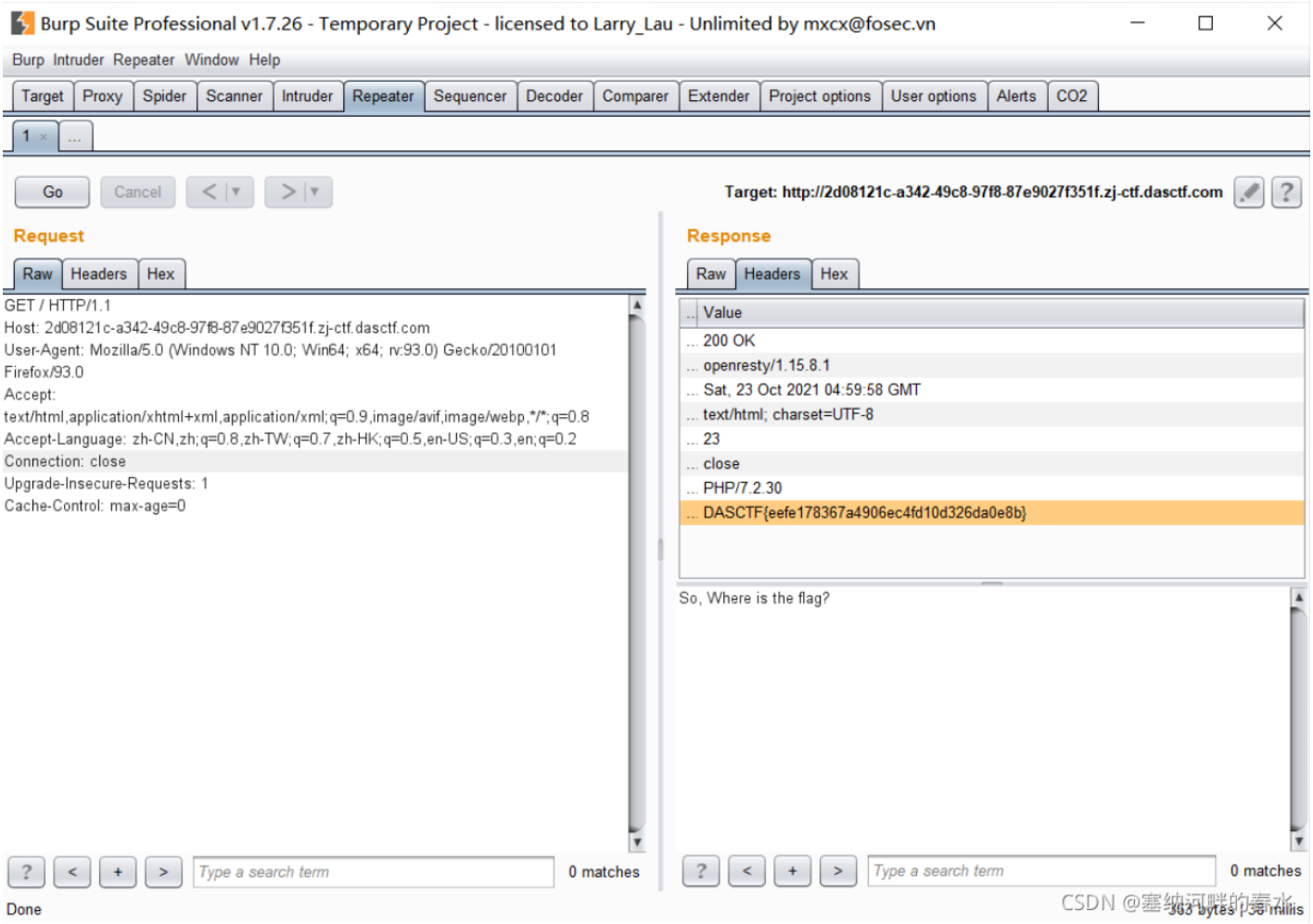
文章目录

前言：这次预赛感觉比去年难多了，难题都没解出来，还是太菜了orz。

Web

Checkin

纯签到题，题目给了一个网址，直接burpsuite抓包，在响应头上拿到flag



RE

crackPYC

题目提供python字节码，直接找到关键部分进行人工反编译，脚本如下：

```
"""
 19          174 SETUP_LOOP              48 (to 224)
             176 LOAD_NAME                11 (range)
             178 LOAD_CONST               36 (32)
             180 CALL_FUNCTION             1
             182 GET_ITER
        >>   184 FOR_ITER                36 (to 222)
             186 STORE_NAME               12 (i)

 20          188 LOAD_NAME                13 (ord)
             190 LOAD_NAME                 4 (str)
             192 LOAD_NAME                12 (i)

   st[i] = ord(str[i])

   194 BINARY_SUBSCR
             196 CALL_FUNCTION             1
             198 LOAD_NAME                10 (key)
             200 LOAD_NAME                12 (i)
   202 LOAD_NAME                6 (len)
             204 LOAD_NAME                10 (key)
             206 CALL_FUNCTION             1
   208 BINARY_MODULO
             210 BINARY_SUBSCR
```

```
            212 BINARY_XOR
            214 LOAD_NAME                9 (st)
            216 LOAD_NAME               12 (i)
            218 STORE_SUBSCR

   st[i] ^ key[i % len(key)]

  2           0 LOAD_CONST               1 (0)
              2 STORE_FAST               1 (num)

    num = 0

  3           4 SETUP_LOOP              42 (to 48)
              6 LOAD_GLOBAL              0 (range)
              8 LOAD_CONST               2 (8)
             10 CALL_FUNCTION            1
             12 GET_ITER
        >>   14 FOR_ITER                30 (to 46)
             16 STORE_FAST               2 (i)

    for i in range(0,8)

  4          18 LOAD_FAST                1 (num)
             20 LOAD_CONST               3 (7508399208111569251)
             22 BINARY_SUBTRACT
             24 LOAD_CONST               4 (4294967295)
             26 BINARY_MODULO
             28 STORE_FAST               1 (num)

    num =(num - 7508399208111569251) % 4294967295

  5          30 LOAD_FAST                0 (key)
             32 LOAD_METHOD              1 (append)
             34 LOAD_FAST                1 (num)
             36 LOAD_CONST               5 (24)
             38 BINARY_RSHIFT

             key.append(num >> 24)
"""
num = 0
key = []
for i in range(0, 8):
    num = (num - 7508399208111569251) % 4294967295
    key.append(num >> 24)
print(key)
# key = [40, 80, 121, 161, 202, 242, 27, 67]
text = [108, 17, 42, 226, 158, 180, 96, 115, 64, 24, 38, 236, 179, 173, 34, 22, 81, 113, 38, 215, 165, 135,
        97, 45, 254, 250, 172, 43, 62]
for i in range(len(text)):
    tmp = text[i] ^ key[i % 8]
    print(chr(tmp), end="")
# DASCTF{0hH_My_9Uy!_vou_D_1T_0^0}
```

Crypto

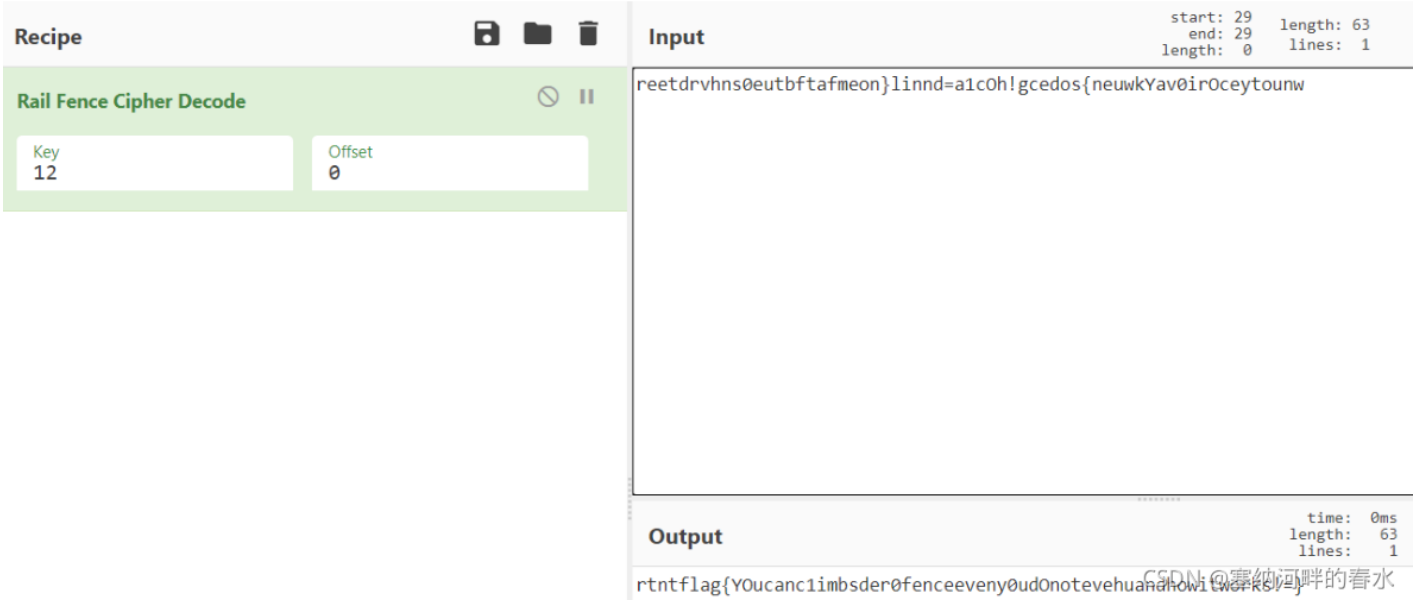
Easy Railfence

题目：

```
reetdrvhns0eutbftafmeon}linnd=a1cOh!gcedos{neuwkYav0irOceytounw
```

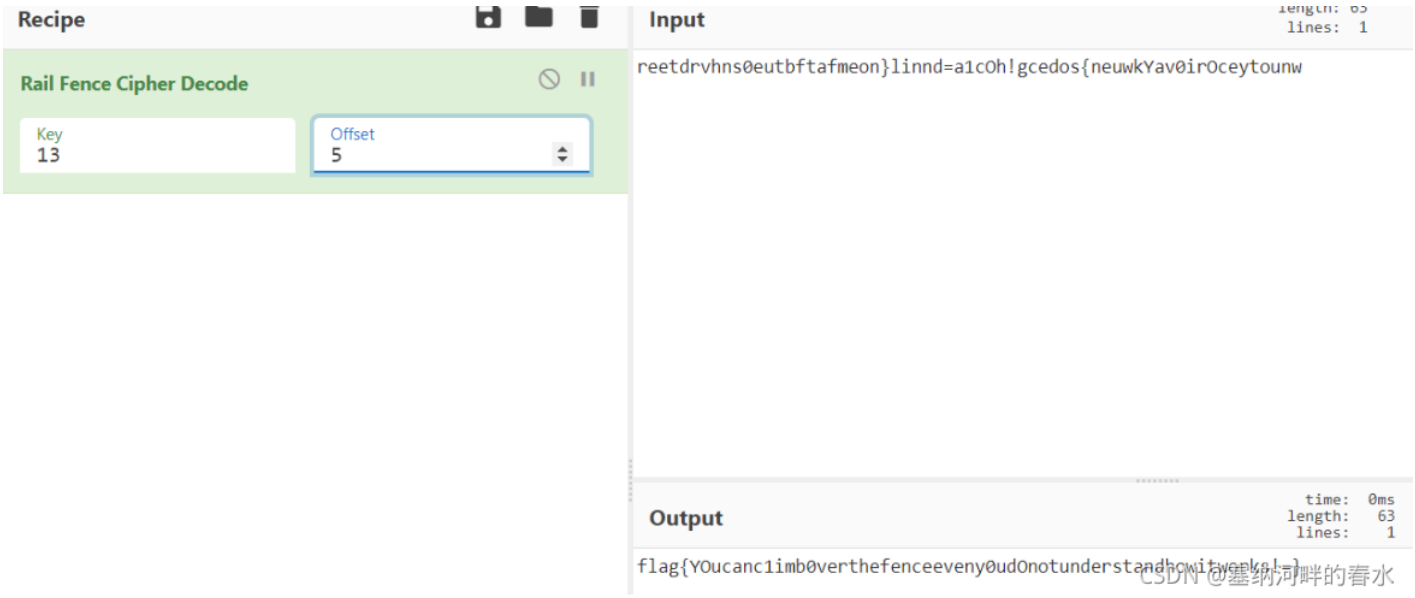
题目名提示栅栏，直接尝试栅栏编码解密，在12栏发现flag字样

```
12栏：rtntflag{YOucanc1imbsder0fenceeveny0udOnotevehuandhowitworks!=}
```



尝试遍历offset无果

改到13栏，遍历offset，最终在offset=5处拿到flag

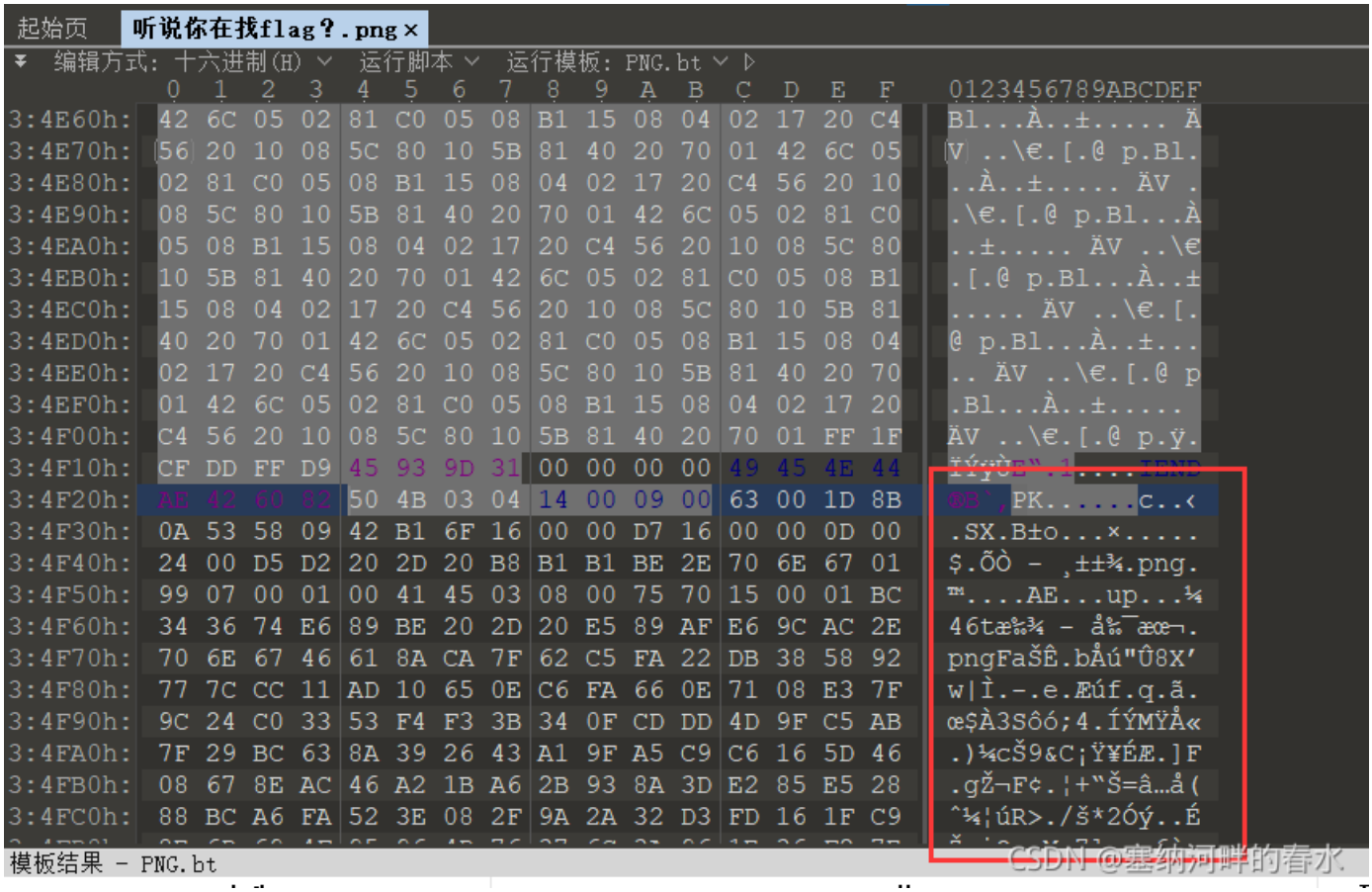



```
flag{YOucanc1imb0verthefenceeveny0udOnotunderstandhowitworks!=}
```

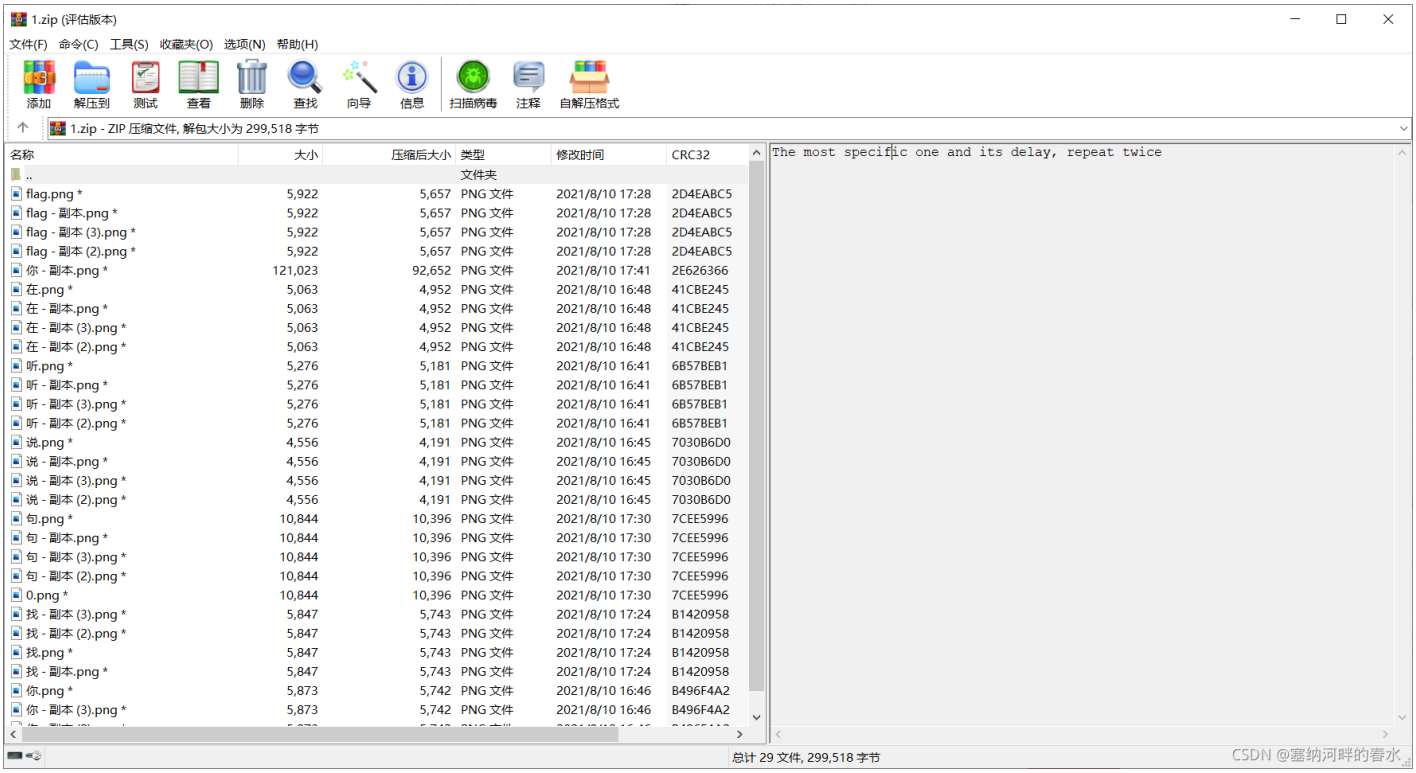
MISC

你能找到flag吗？

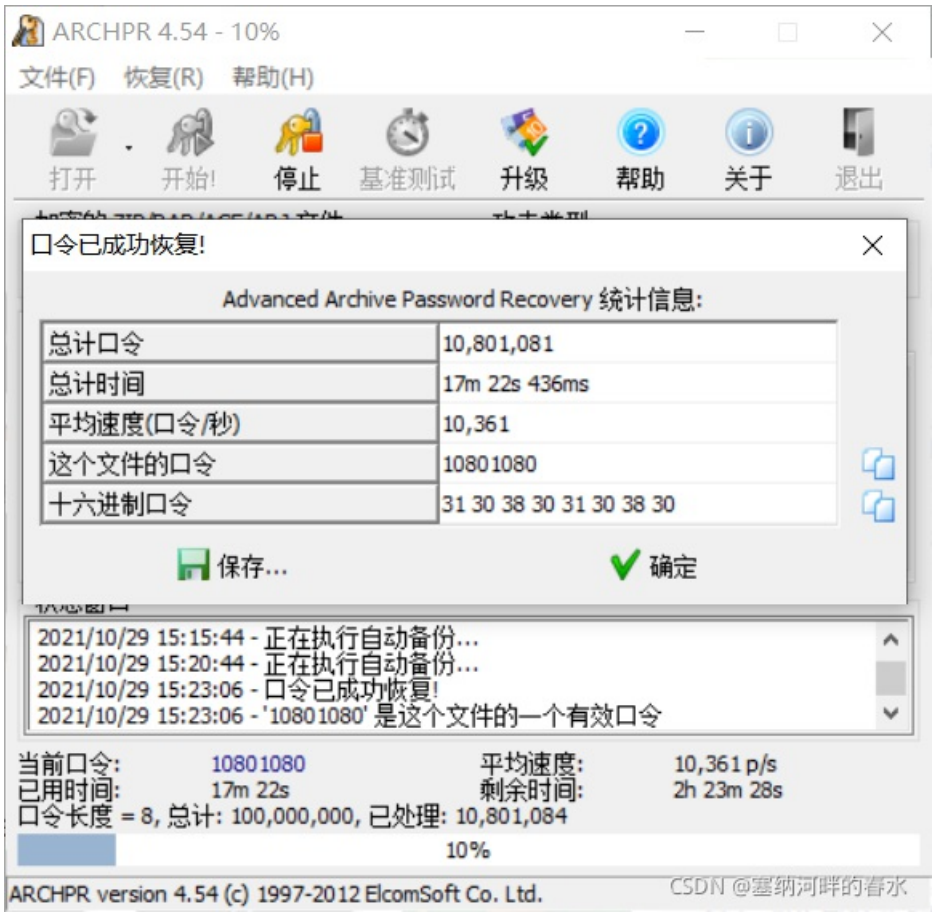
图片给了一张png图片，010Editor打开发现末尾存在压缩包，将压缩包分离。



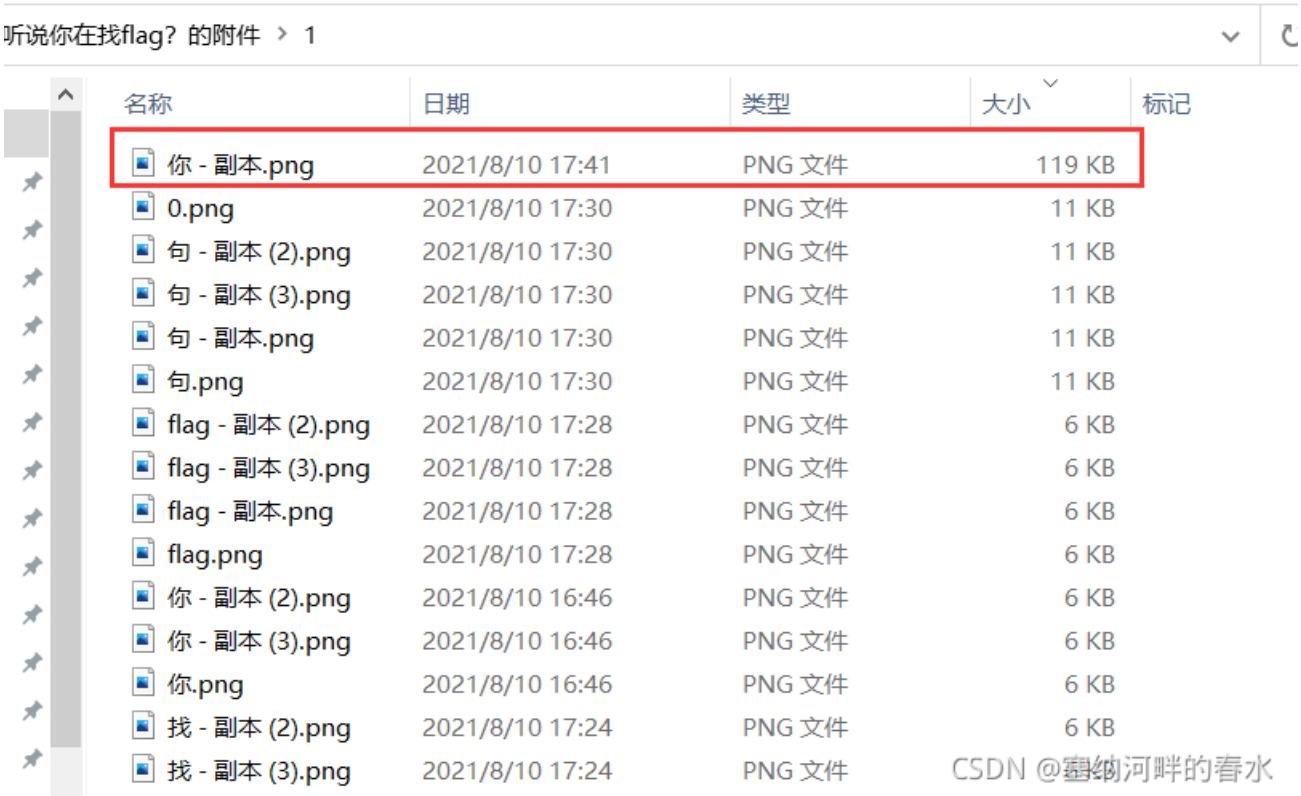
得到一个加密压缩包，里面含有大量图片。



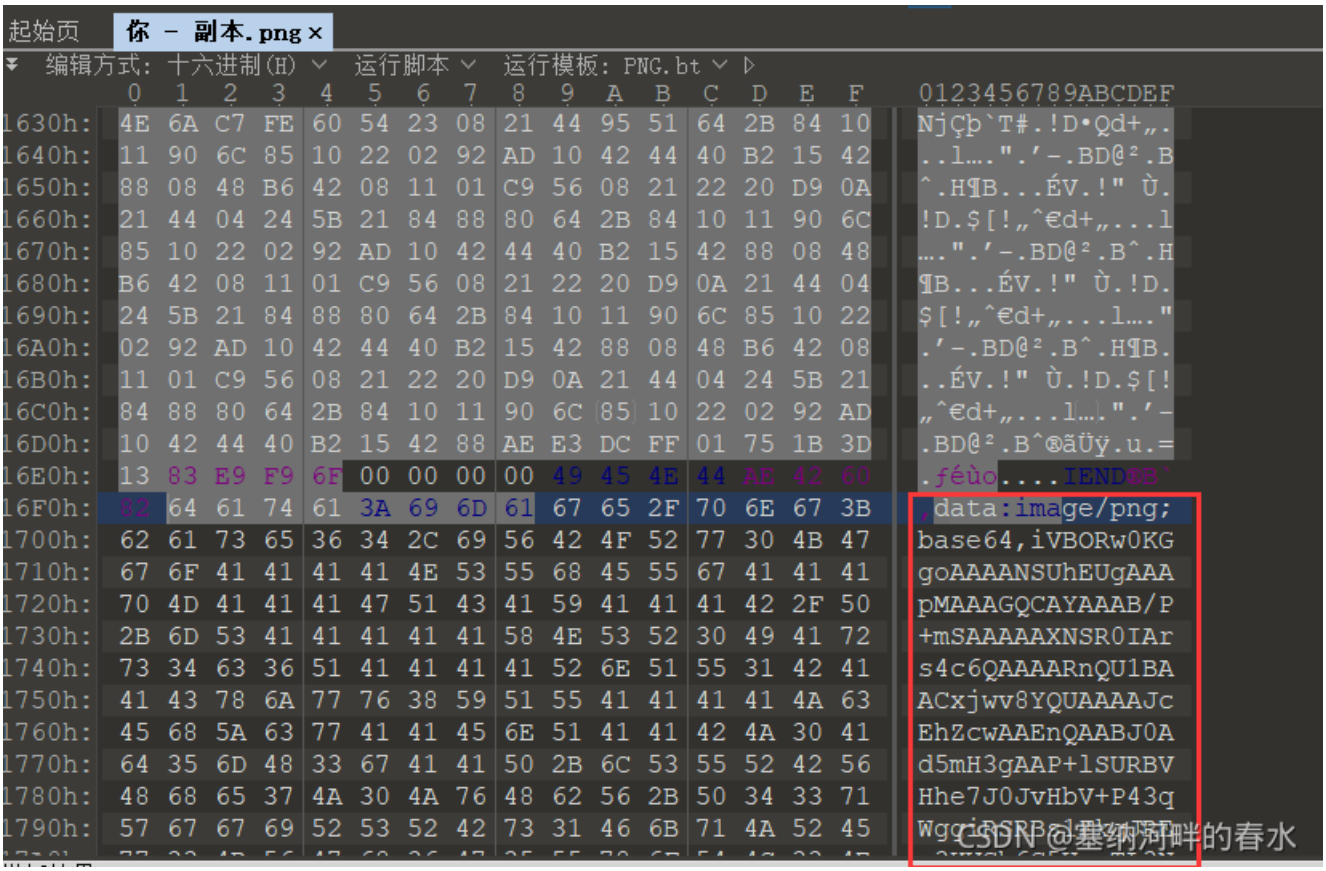
找密码，fuzz一下没什么发现尝试纯数字爆破，十几分钟后，密码10801080



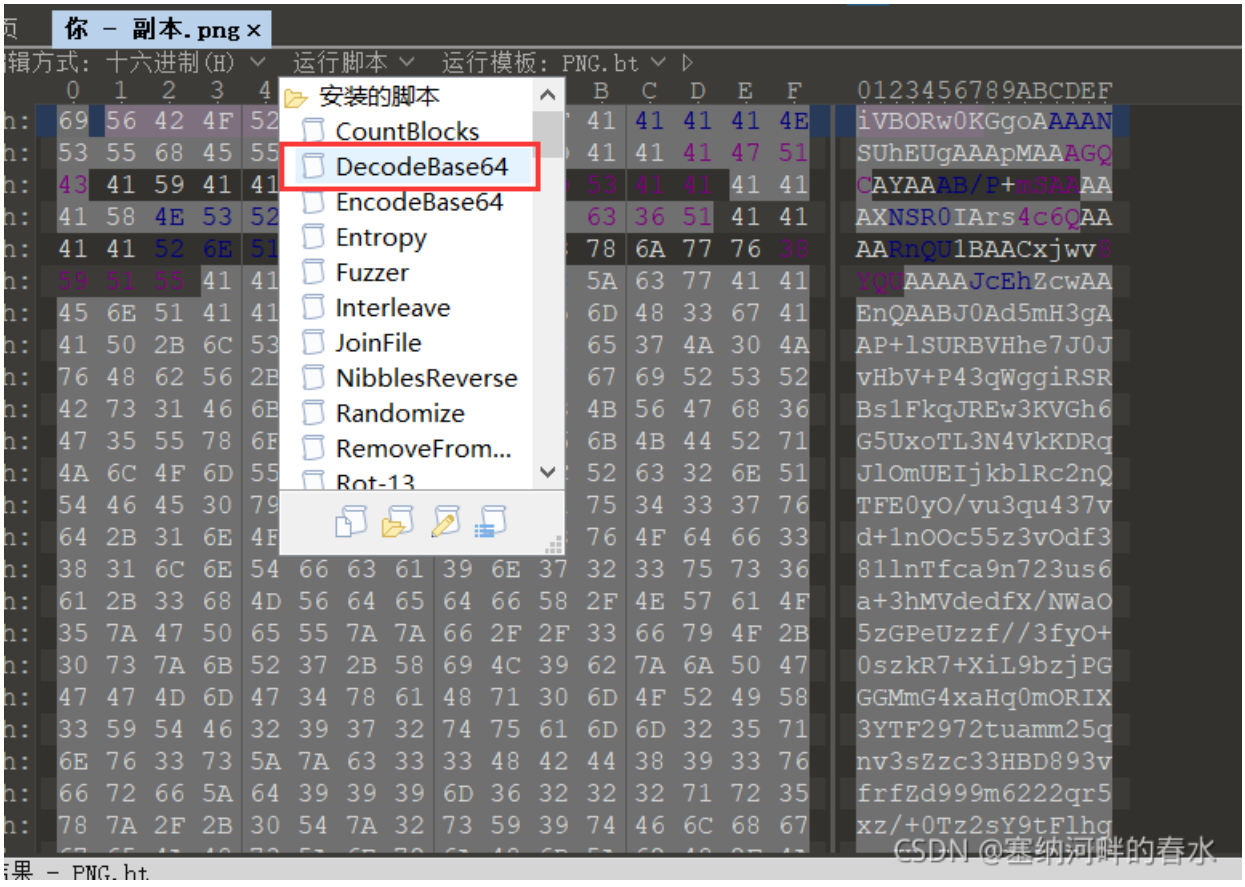
有密码后直接解压，发现有张图片特别大（没发现特别大，题目其实也给了hint，后面再说）。

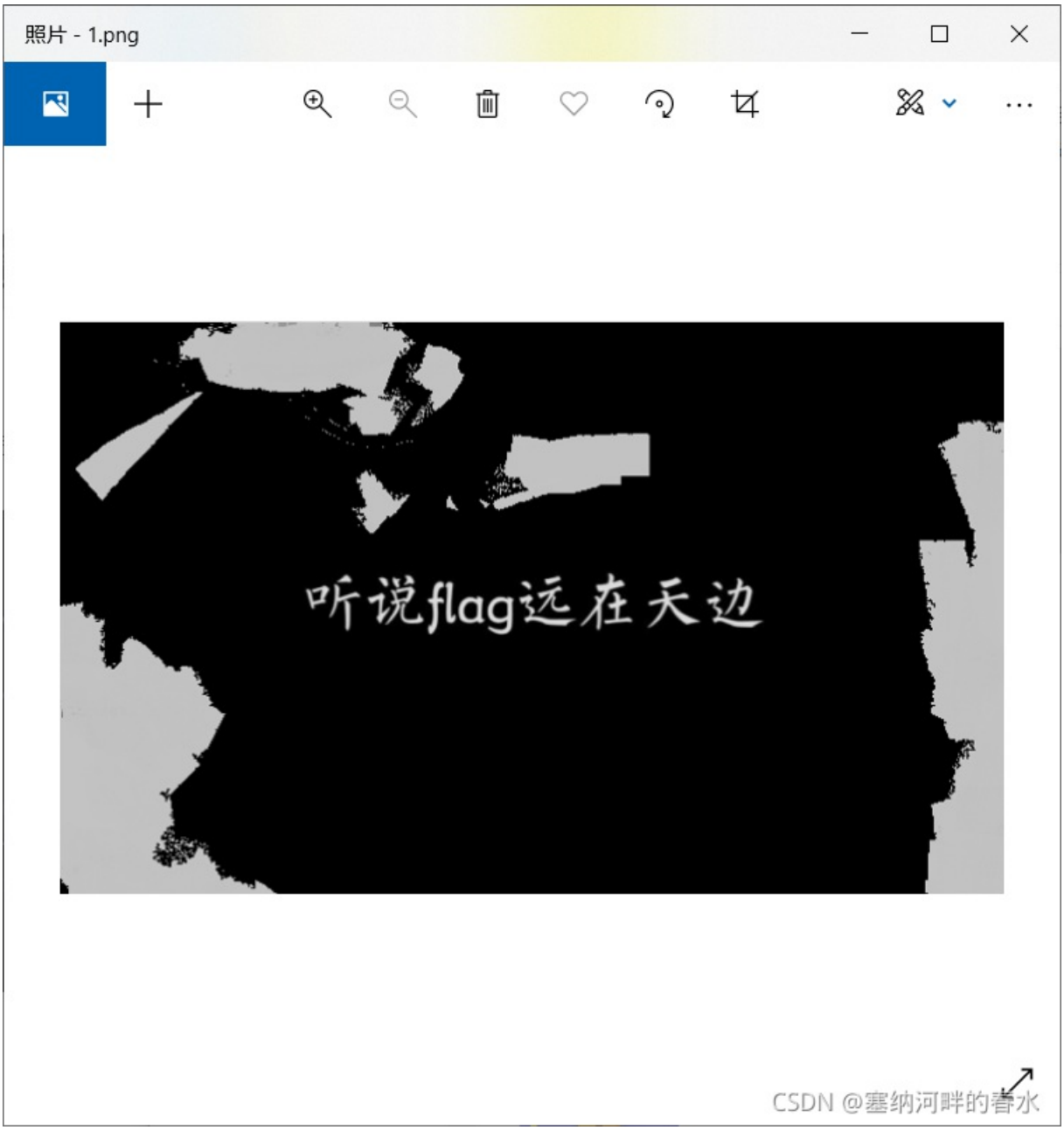


010Editor打开，在最后发现一串base64图片编码。



截取出base64后运行010Editor内置脚本编码。





得到一张图片，fuzz一下宽高不对劲，尝试爆破。

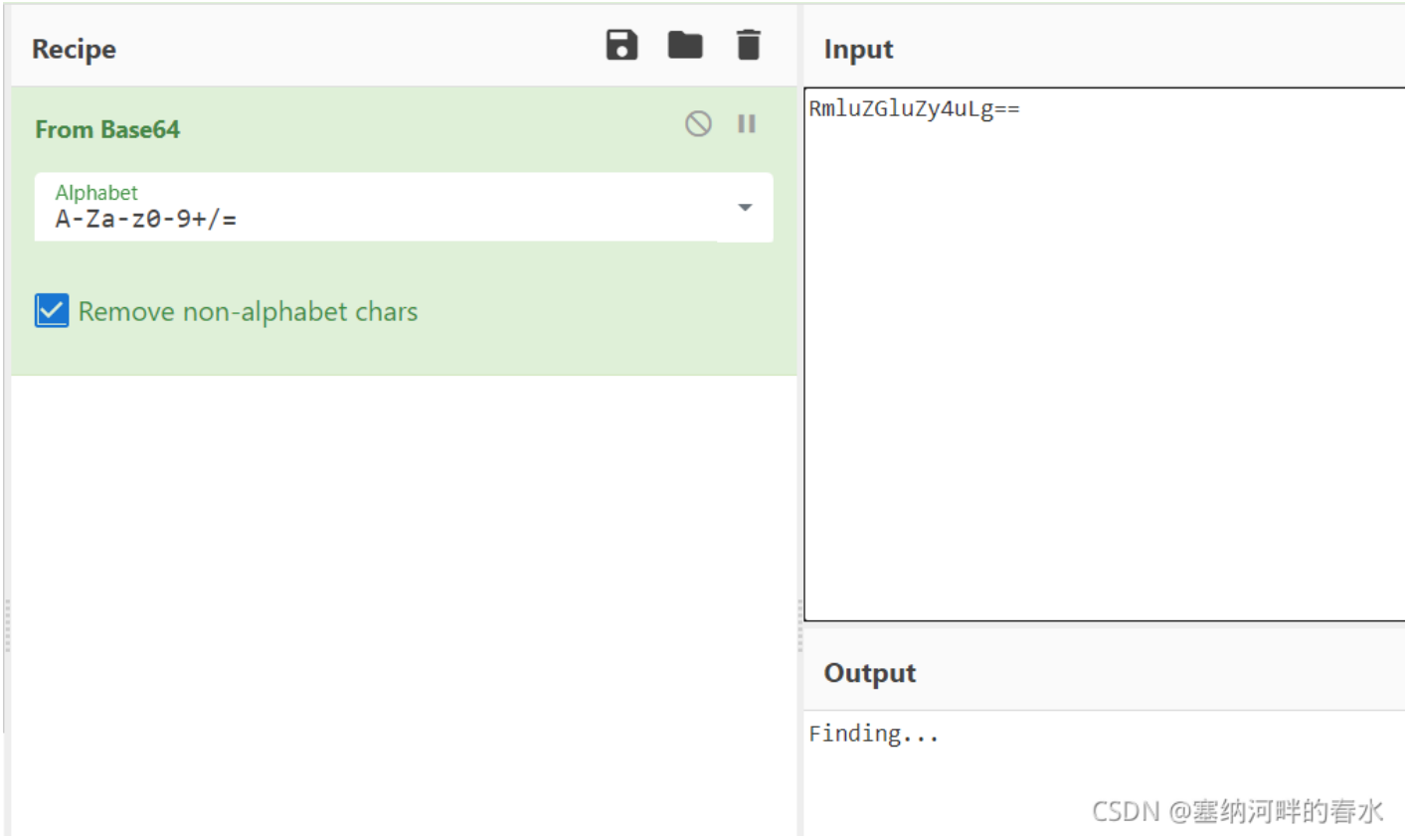
```
import struct
import binascii
from Crypto.Util.number import bytes_to_long

png_input = open('1.png', 'rb').read()

for i in range(0xFFFF):
    stream = png_input[12:20] + struct.pack('>i', i) + png_input[24:29]
    crc = binascii.crc32(stream)
    if crc == bytes_to_long(png_input[29:33]):
        print(i)
        # 800
```

修改png图片高度为800，出现一串编码，形似镜像后的base64。

解码拿到字符串 Finding...



用密码去解压flag.rar，拿到最终flag。

```
DASCTF{flag_1s_close_at_your_hand}
```

qrimg

下载附件，是个gif文件，将其一帧一帧分离，共计312帧，尝试将其中的图片拖进StegSolve工具查看，发现在Blue plane0时会出现二维码，如下图所示：

312张图片每一张的Blue plane0通道都有二维码，写脚本提取：

```
from PIL import Image
import pyzbar.pyzbar as pyzbar
import os, base64

def qrcode_parse_content(img):
    barcodes = pyzbar.decode(img)

    result = []
    for barcode in barcodes:
        barcode_content = barcode.data.decode('utf-8')
        result.append(barcode_content)

    return result


def getB0(filepath):
    list1 = []
    im1 = Image.open(filepath)
    for h in range(height):
        for w in range(width):
            b0 = bin(im1.getpixel((w, h))[2])[-1]  # 取b通道的低0位
            list1.append(b0)
    return list1
```

```
def genpic(list1):
    im1 = Image.new("RGB", (width, height), 'white')
    i = 0
    for y in range(height):
        for x in range(width):
            if list1[i] == '0':
                im1.putpixel([x, y], (0, 0, 0))
            else:
                im1.putpixel([x, y], (255, 255, 255))
            i = i + 1
    return im1


if __name__ == "__main__":
    tmppath = './tmp/'
    if not os.path.exists(tmppath):
        os.mkdir(tmppath)
    print(tmppath)

    tmp = []
    for i in range(312):
        filepath = './qrimg gif/' + 'IMG' + str(i).zfill(5) + '.bmp'
        width, height = Image.open(filepath).size  # 得到宽高
        img = genpic(getB0(filepath))
        img.save(tmppath + str(i).zfill(5) + '.png')
        print(qrcode_parse_content(img))
        tmp += qrcode_parse_content(img)
    str1 = ''.join(tmp)
    print(str1)
    while True:
        try:
            str1 = base64.b64decode(str1).decode("utf-8")
            print(150 * '*')
            print(str1)
        except:
            print(str1)
            break
```

最后连成字符串为base64编码，多次解码之后拿到flag

```
flag{32bb3b8cec39e43a06038a9f96921906}
```