CTF misc图片类总结（深育杯brige、西湖论剑YUSA的小秘密、湖湘杯leaker、wear_a_mask）

原创

Hardworking666 于 2021-11-24 13:26:38 发布 630 收藏 5

分类专栏： CTF 文章标签： 安全 unctf python

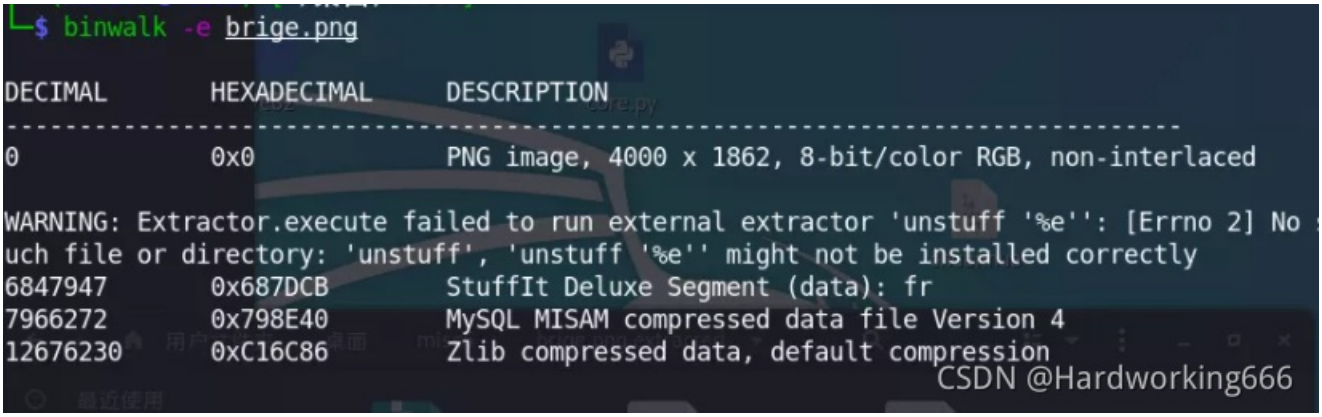
本文链接：https://blog.csdn.net/Hardworking666/article/details/121512575

文章目录

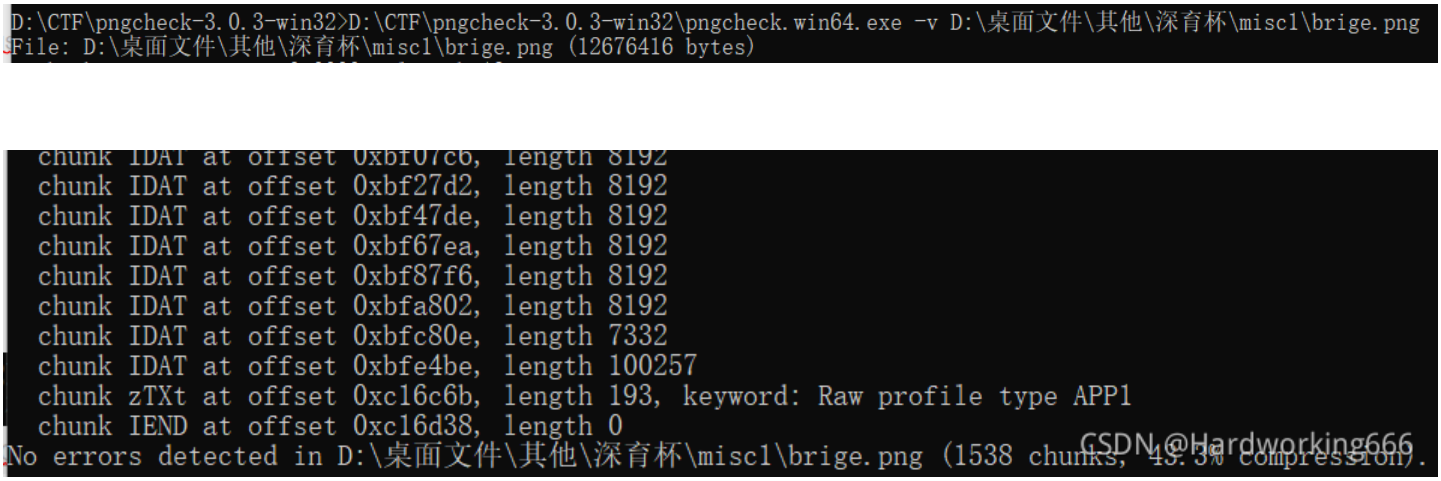
一、zlib_rar+exif+分析像素_zip

第一届深育杯misc，brige.png

使用binwalk分析出有zlib数据，但是无法使用binwalk -e或foremost分离出有效文件，在010editor中查看图片。

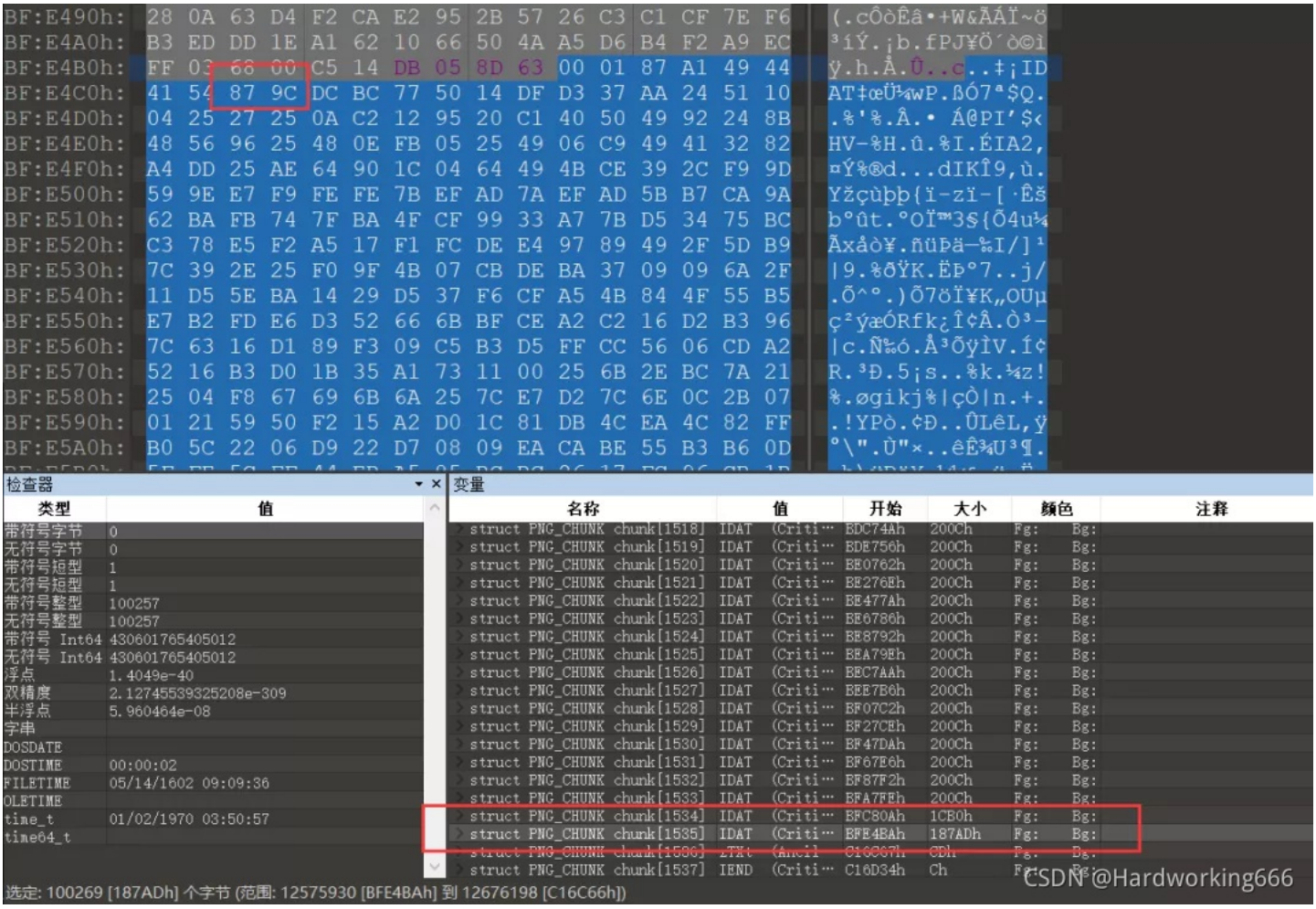


（1）运行命令 pngcheck.exe -v xx.png，可以详细查看每个数据块的情况



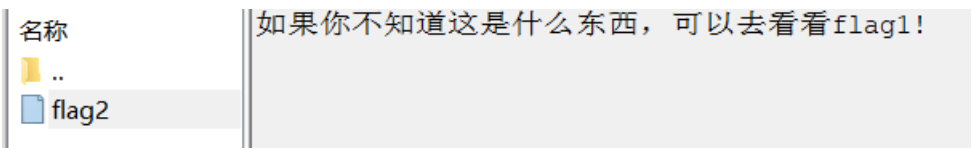
或者（2）010 editor中看到最后一个IDAT数据块长度异常，导出这段zlib数据。（编辑–>复制为十六进制文本，粘贴到txt中）观察IDAT标识后面的87 9C两个字节，恢复成zlib数据头标识78 9C，写python3脚本将此段zlib数据解压缩，可得到一个rar压缩包。

注意解压缩的zlib数据应该是去掉IDAT-length、IDAT-type、IDAT-crc的数据部分，即只有（78 9C … 60 A5 85 A2）。




```
import zlib

data = open("zlib_hex_data.txt", 'r').read().replace(" ", "").replace("\n", "").strip()
data_dec = zlib.decompress(bytes.fromhex(data))
print(data_dec[:100])
with open("zlib_data.rar", 'wb') as wf:
    wf.write(data_dec)
# 结果：b'Rar!\x1a\x07\x01\x00J\x97,}\x0c\x01\x05\x08\x00\x07\x01\x01\x96\x9c\x87\x80\x00\xf7\xea}W\x13\x03\x02\
xbd\x00\x04\xbd\x00\x00\x90:\xd1\xdc\x80\x00\x00\x03CMT\xe5\xa6\x82\xe6\x9e\x9c\xe4\xbd\xa0\xe4\xb8\x8d\xe7\x9f\
xa5\xe9\x81\x93\xe8\xbf\x99\xe6\x98\xaf\xe4\xbb\x80\xe4\xb9\x88\xe4\xb8\x9c\xe8\xa5\xbf\xef\xbc\x8c\xe5\x8f\xaf\
xe4\xbb\xa5\xe5\x8e\xbb\xe7\x9c\x8b'
```

解压压缩包可得flag2，注意压缩包中有提示请先获取flag1。



继续找flag1，分析最开始的那张图片，实际使用zsteg和zsteg可以发现其他可以信息。

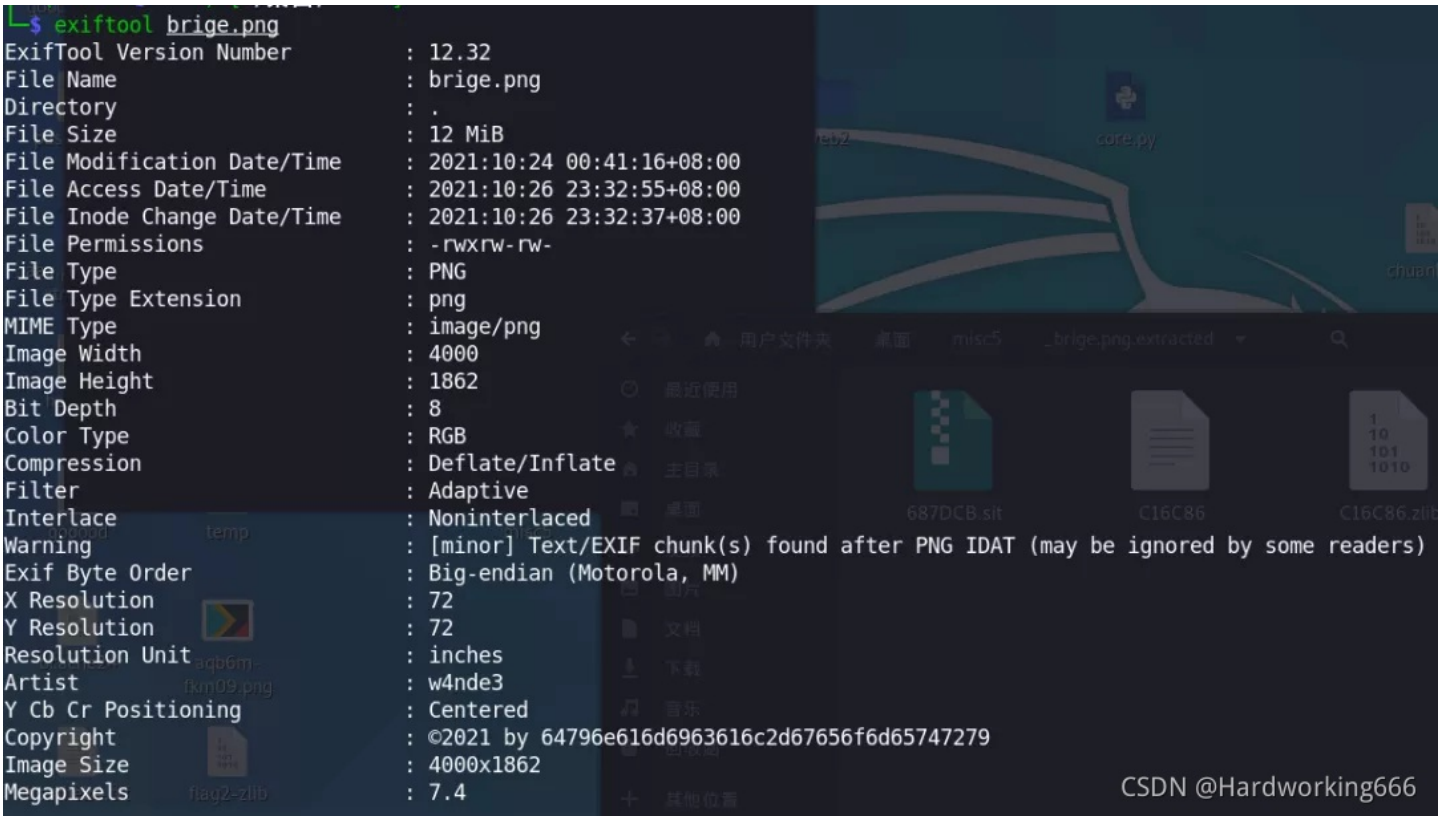
ExifTool可以提取照片exif信息。可交换图像文件格式（英语：Exchangeable image file format，官方简称Exif），是专门为数码相机的照片设定的，可以记录数码照片的属性信息和拍摄数据。

exiftool安装：

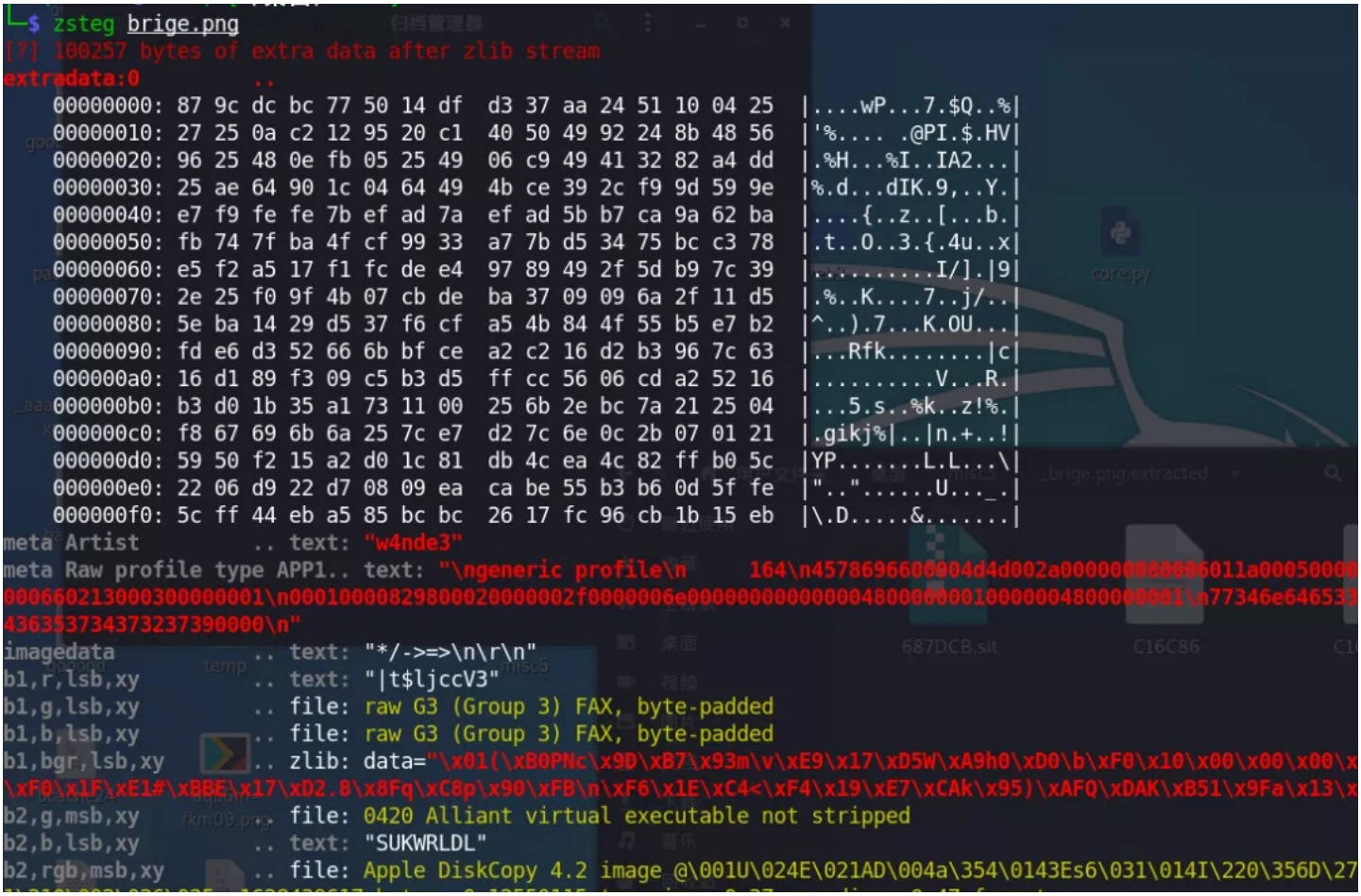
```
sudo apt install libimage-exiftool-perl
```

exiftool看到Copyright（版权）是十六进制，转换成文本：dynamical-geometry（动态几何），这个是后面的密钥。

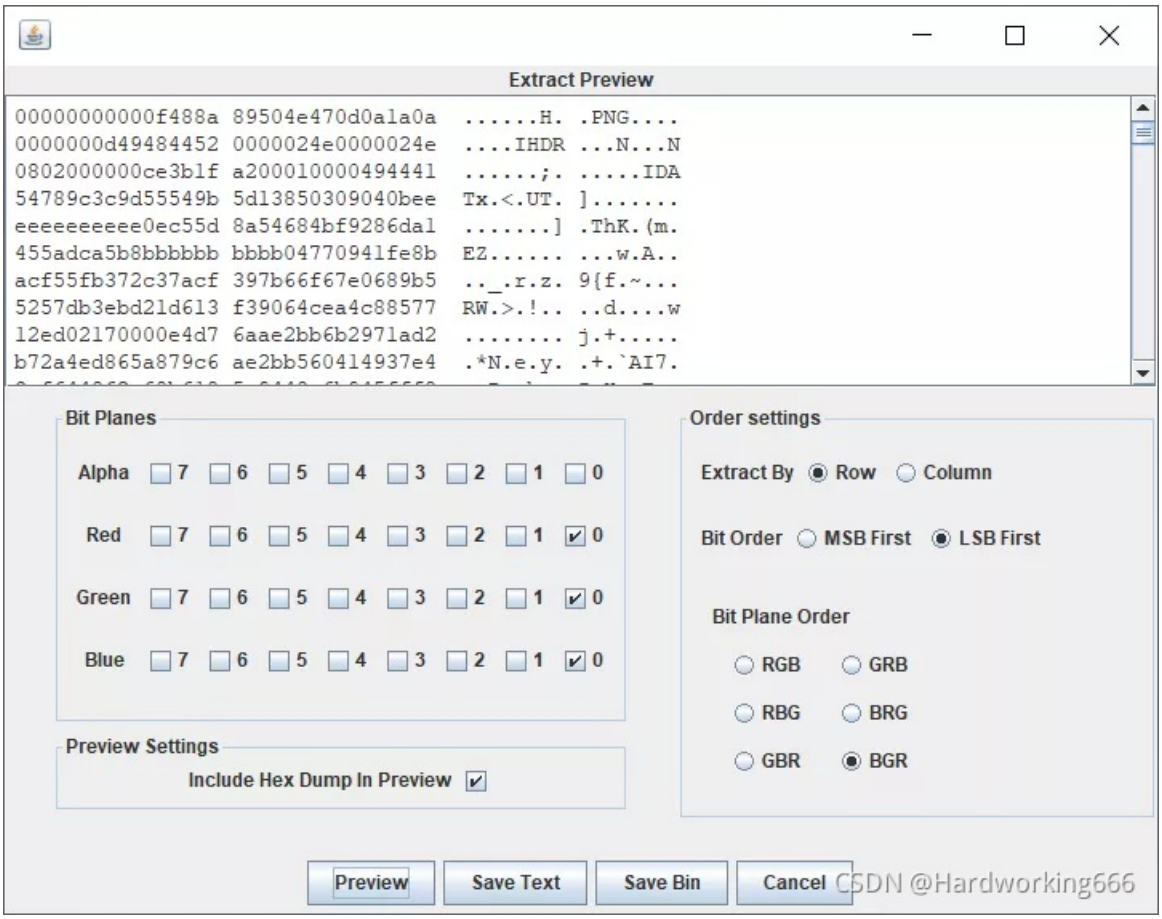
```
bytes.fromhex('64796e616d6963616c2d67656f6d65747279')
# b'dynamical-geometry'
```



zsteg发现这张图片除了存在extradata外，在中也有脏数据。



使用StegSolve检查隐写。



导出十六进制，这里不能直接打开图片，可使用foremost将PNG快速分离出来，最后得到一张590x590，大小为979KB的图片.。
注意如果仅去掉PNG字符前数据并改后缀为PNG也能正常查看图片，但会阻塞下一步分析像素操作。
到这里只有一张色彩值杂乱的PNG图片，分析其像素。

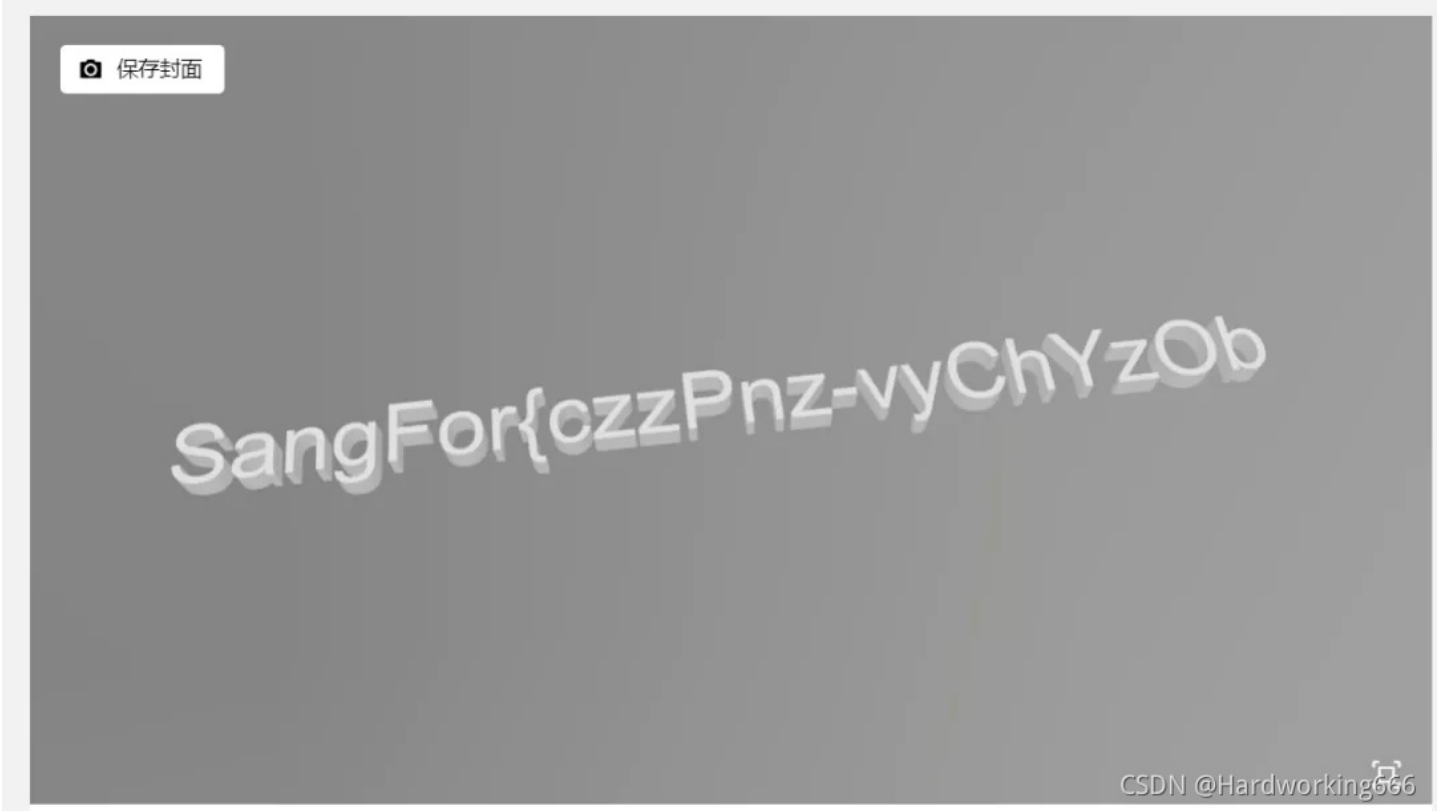
```
from PIL importImage
image = Image.open(r'bri.png')
allpixels = []
for x in range(image.width):
for y in range(image.height):
        allpixels.append(image.getpixel((x, y)))
print(len(allpixels)) # 348100
print(allpixels[:4])
# [(40, 176, 80), (37, 181, 75), (1, 253, 3), (2, 252, 4)]
#           0x50           0x4B          0x03         0x04
```

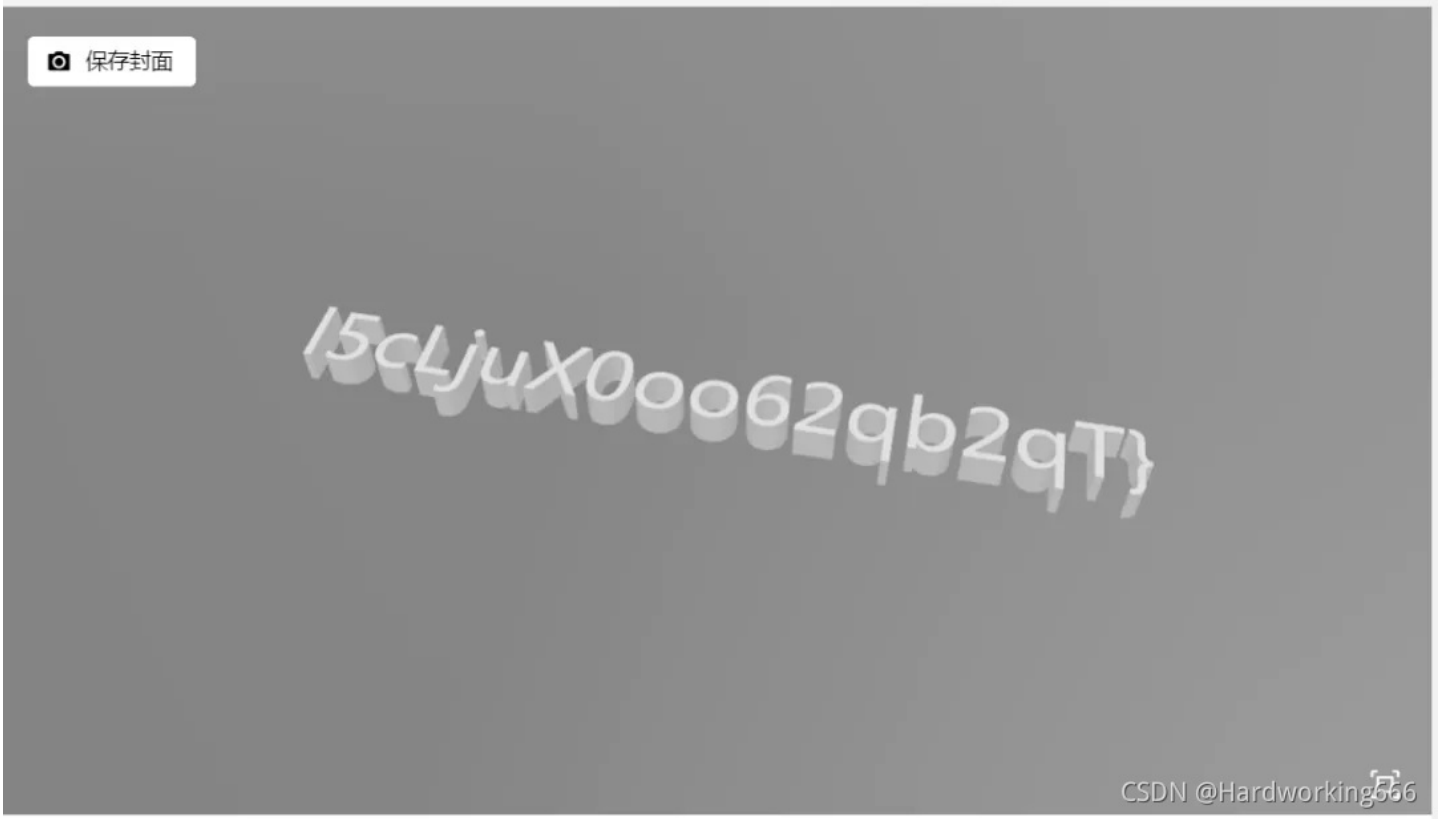
取前四个字节即可看出，像素第三列隐藏着压缩包十六进制，批量提取并保存成zip压缩包，使用前面得到的密码：dynamical-geometry解密，得到flag1文件。

```
from PIL importImage
image = Image.open(r'bri.png')
allpixels = []
for x in range(image.width):
for y in range(image.height):
if image.getpixel((x, y)) == (0, 0, 0):
continue
        allpixels.append(image.getpixel((x, y))[2])
hex_datalist = [str(hex(i))[2:].zfill(2) for i in allpixels]
print("".join(hex_datalist)[:100])
# 504b0304140009006300caa05753d904fdb22a4b0500dce856000f000b00666c6167312d61736369692e73746c0199070001


with open("outpur.txt", 'w') as wf:
    wf.write("".join(hex_datalist))
```

记事本打开文件后，是3D打印模型中的STL格式文件，STL格式分为ascii、binary格式，使用在线工具查看模型即可。
使用在线网站查看stl，根据flag1的STL格式，将flag2也尝试用STL预览器查看：

```
https://www.viewstl.com/#!
```





2021深育杯线上初赛官方WriteUp
深育杯misc

二、YCrCb通道隐写

2021“西湖论剑”网络安全大赛YUSA的小秘密

这题采用的YCrCb通道。通过 cv2.cvtColor(img, cv2.COLOR_BGR2YCrCb)对 img 图片数据进行色彩空间转换，即可得到三个通道的数据，然后对三个通道的数据分别根据奇偶做二值化处理并保存为图片。

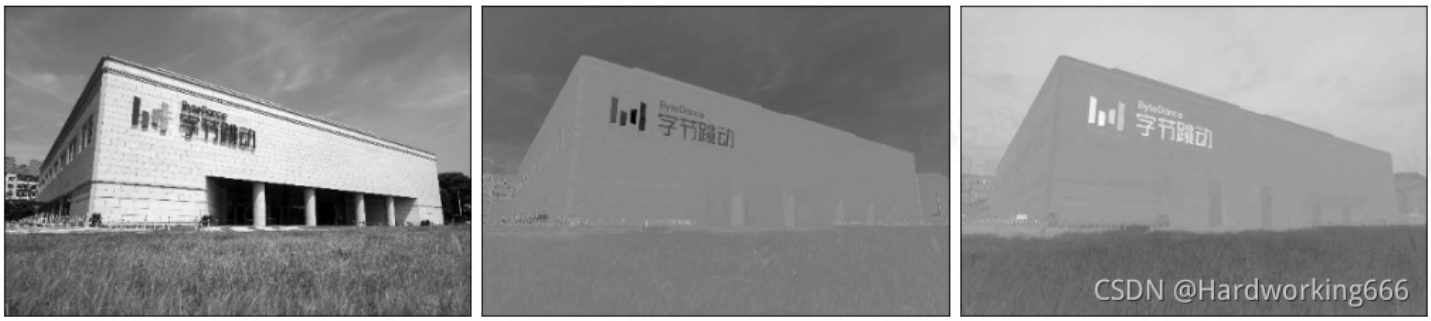
```
from cv2 import *
import cv2 as cv
img=cv2.imread('C:\\Users\\XXX\\Desktop\\yusa\\yusa.png')
src_value=cv2.cvtColor(img, cv2.COLOR_BGR2YCrCb)
a, b, c = cv.split(src_value)   #使用cv.split分离通道
cv.imwrite('a.png', (a % 2) * 255)   #对三个通道中的数据分别根据奇偶做二值化处理，并分别保存为图片
cv.imwrite('b.png', (b % 2) * 255)
cv.imwrite('c.png', (c % 2) * 255)
```

运行后会得到三个通道的图片，在其中a.png即可清晰看到flag

2021“西湖论剑”其他wp

2020ByteCTF Hardcore Watermark 01

图片中每个像素可以通过三个值(通道)来表示，常见的是 R(red)G(green)B(blue) 模式。而本题用到的通道是 YCrCb。通过 cv2.cvtColor(img, cv2.COLOR_BGR2YCrCb)对 img 图片数据进行色彩空间转换，即可得到三个通道的数据：



对三个通道中的数据根据奇偶做二值化处理，也即判断数据的最低位：

```
dst_value = (src_value % 2) * 255
```

二值化后的数据分别代表二维码的黑和白，并且每个通道可得到部分二维码图片。最后只需将三个通道数据结合到一幅图中即可复现二维码。
如果不对三通道进行拆分的话，直接提取RGB通道会造成非常多的噪点，无法识别。但是可以搜到原图进行了diff，这样能减少大部分的噪点。再结合QRazyBox工具对二维码进行还原，也可以拿到flag的内容。
QRazyBox二维码拼接网站
Linux diff 命令

ByteCTF 2020 部分题目 官方Writeup

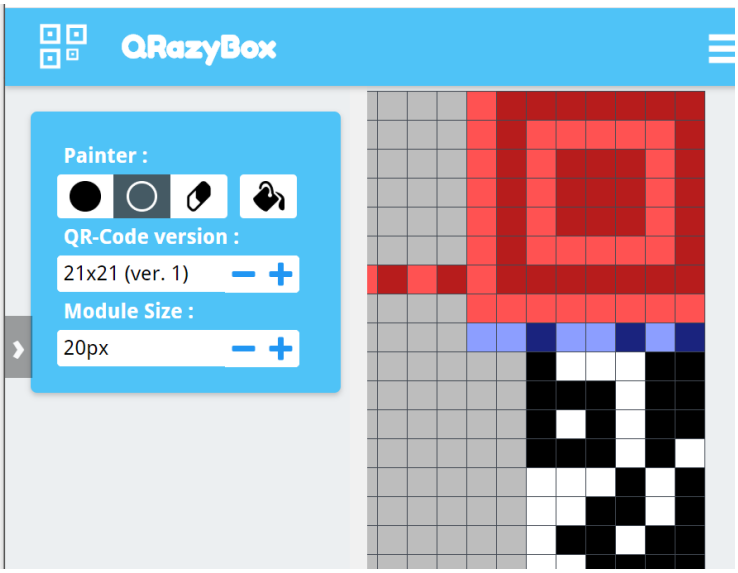
三、gif二维码

安恒月赛 DASCTF 7月赛QrJoker
题目是一个gif，都是只有一点点的二维码

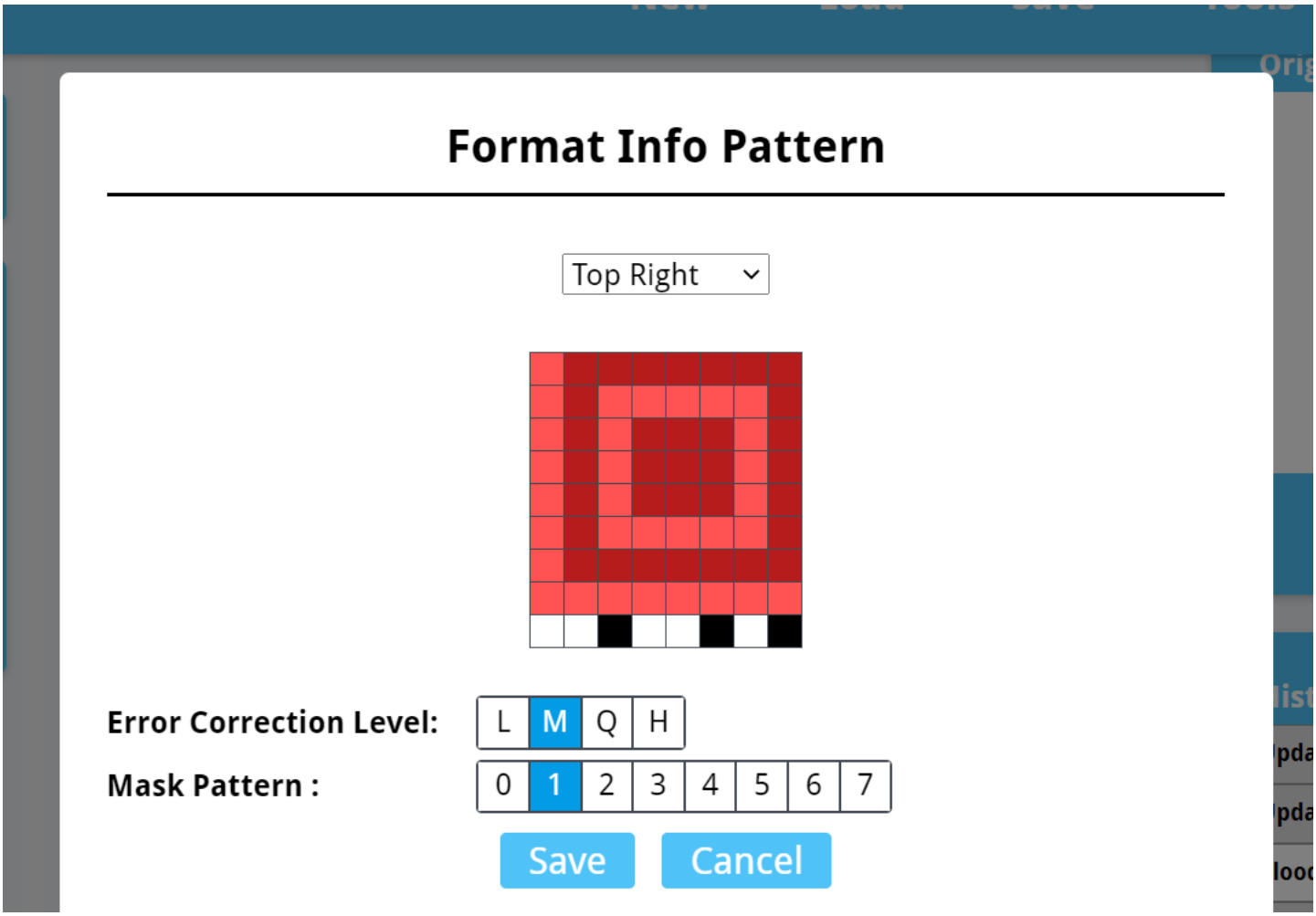
使用脚本切割一下：

```
from PIL import Image
import os
gifFileName = 'QrJoker.gif'
#使用Image模块的open()方法打开gif动态图像时，默认是第一帧
im = Image.open(gifFileName)
pngDir = gifFileName[:-4]
#创建存放每帧图片的文件夹
os.mkdir(pngDir)
try:
 while True:
  #保存当前帧图片
  current = im.tell()
  im.save(pngDir+'/'+str(current)+'.png')
  #获取下一帧图片
  im.seek(current+1)
except EOFError:
  pass
```

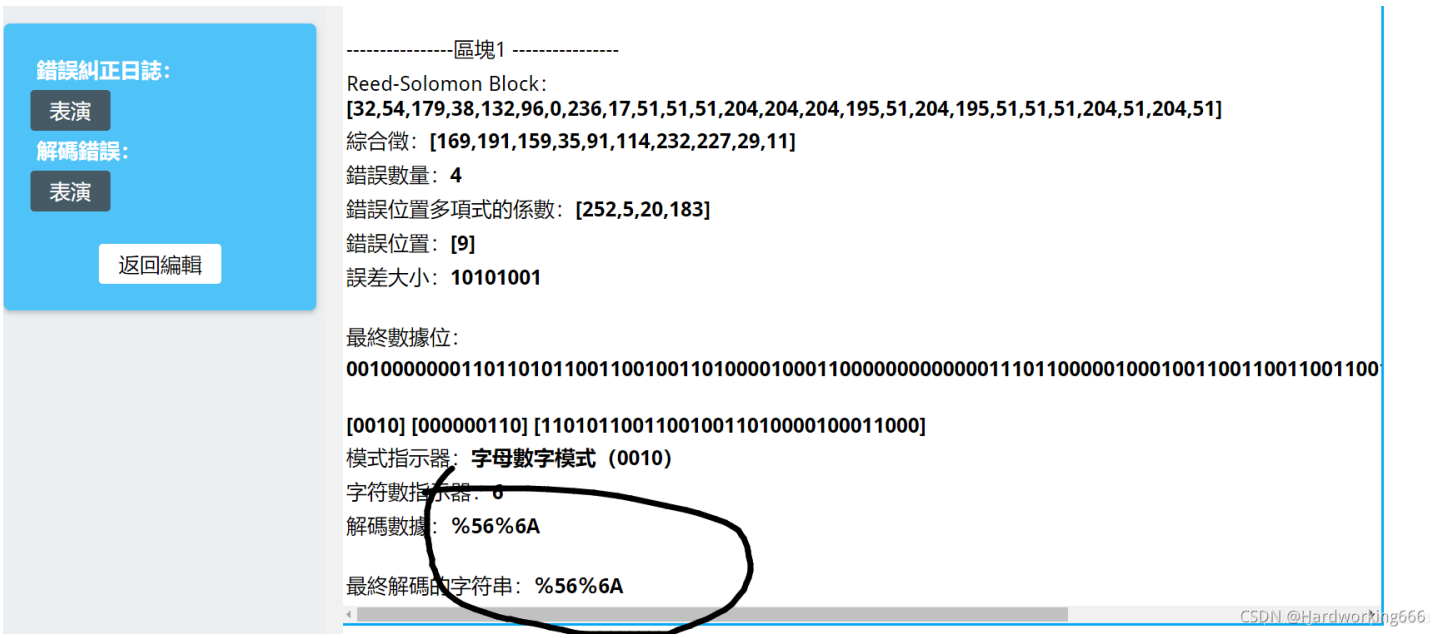
https://merricx.github.io/qrazybox/
这个高是420 一个格子20 420/20=21
得出是21*21格子然后开始操作



在当中选择这个模式，就可以填充了




QRazyBox
新
加載
保存
工具
類
救命
關於
編輯器模式
畫家：
QR碼版本：
21x21 (ver. 1)
模組尺寸
20px
工具清單
提取QR信息
強制解碼並儘可能獲取有關當前QR碼的信息
里德-所羅門解碼器
通過解碼Reed-Solomon塊進行糾錯和糾刪
蠻力格式信息模式
解碼時嘗試所有格式信息模式的可能性
數據屏蔽
使用Mask模式模擬數據屏蔽（XOR）
填充位恢復
通過放置終止符和填充位來恢復丟失的位
數據序列分析（實驗）
分析QR碼的數據序列
關
原始樣本：
加載樣本
歷史：
更新格式信息模式
洪水填充
洪水填充
更新格式信息模式
更新格式信息模式


錯誤糾正日誌：
表演
解碼錯誤：
表演
返回編輯
----------區塊1 ----------
Reed-Solomon Block:
[32,54,179,38,132,96,0,236,17,51,51,51,204,204,204,195,51,204,195,51,51,51,204,51,204,51]
綜合徵：[169,191,159,35,91,114,232,227,29,11]
錯誤數量：4
錯誤位置多項式的係數：[252,5,20,183]
錯誤位置：[9]
誤差大小：10101001
最終數據位:
[0010] [000000110] [11010110011001001101000010000]
模式指示器：字母數字模式（0010）
字符數指示器：6
解碼數據：%56%6A
最終解碼的字符串：%56%6A

```
%56%6A%49%77%65%45%35%48%52%6B%64%69%4D%33%42%72%55%6A%4E%53%55%46%6C%58%4D%54%52%6A%4D%57%52%79%57%6B%5A%61%54%
31%5A%55%52%6C%5A%5A%56%57%51%30%56%32%31%57%63%6B%31%45%52%6C%56%4E%56%6B%70%78%56%47%78%56%4E%56%4A%58%53%6B%6
8%68%52%54%6C%58%54%56%5A%56%64%31%59%79%4D%58%64%69%4D%6B%5A%79%54%6C%52%61%55%6C%64%49%51%6D%46%57%61%32%52%50
%54%6C%5A%52%65%46%6F%7A%5A%44%46%56%56%44%41%35
解码一下
VjIweE5HRkdiM3BrUjNSUFlXMTRjMWRyWkZaT1ZURlZZVWQ0V21Wck1ERlVNVkpxVGxVNVJXSkhhRTlXTVZVd1YyMXdiMkZyTlRaUldIQmFWa2RP
TlZReFozZFFVVDA5
base64
V20xNGFGb3pkR3RPYW14c1drZFZOVTFVYUd4WmVrMDFUMVJqTlU5RWJHaE9WMVUwV21wb2FrNTZRWHBaVkdONVQxZ3dQUT09
base64
Wm14aFozdGtOamxsWkdVNU1UaGxZek01T1RjNU9EbGhOV1U0Wmpoak56QXpZVGN5T1gwPQ==
base64
ZmxhZ3tkNjllZGU5MThlYzM5OTc5ODlhNWU4ZjhjNzAzYTcyOX0=
base64
flag{d69ede918ec3997989a5e8f8c703a729}
```

一个神仙的脚本

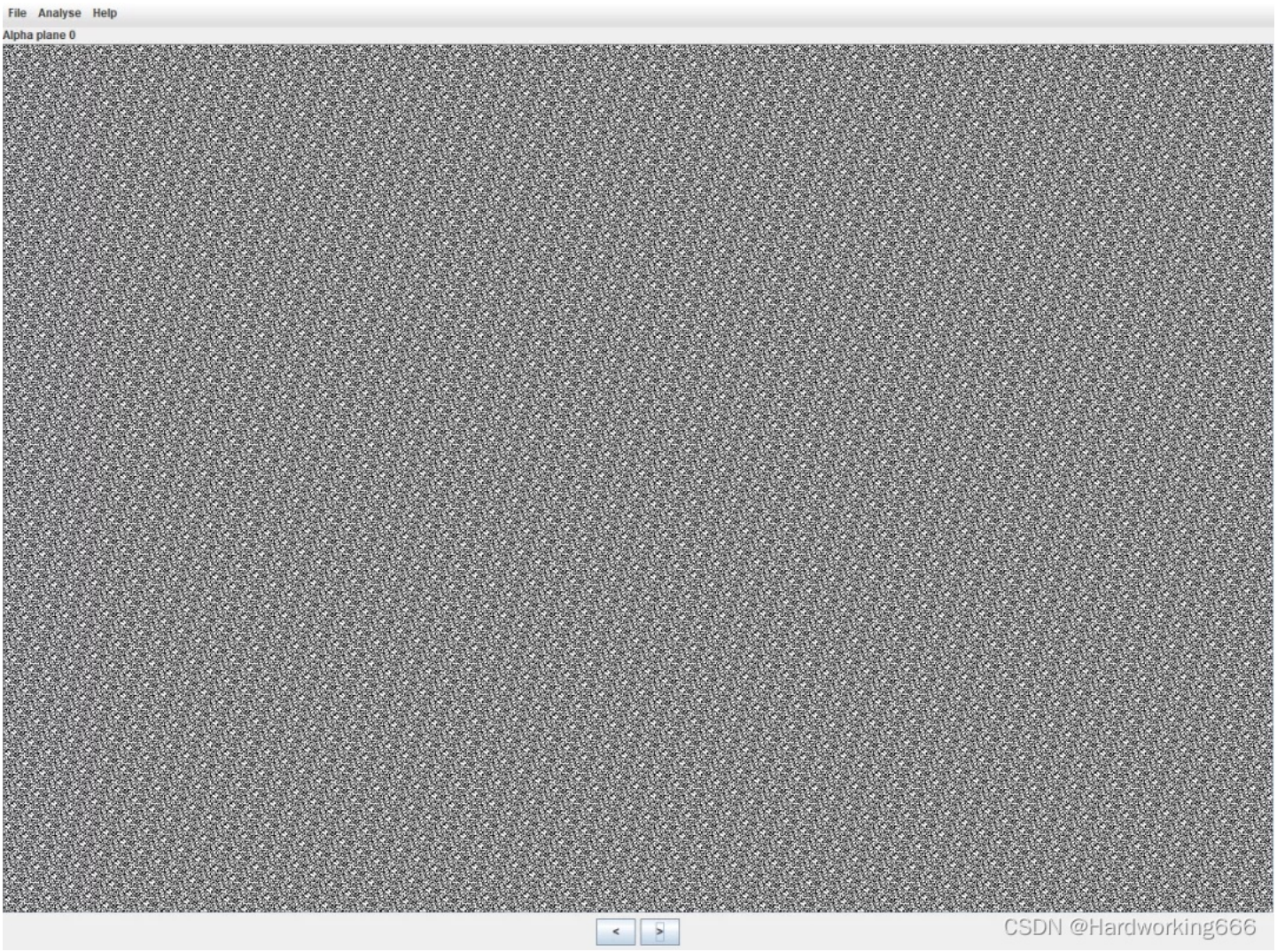
```
from PIL import Image
from Crypto.Util import number
import base64

dic = {0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:'9',10:'A',11:'B',12:'C',13:'D',14:'E',15:'F',38:
'%'}
res = ''
for num in range(64):
    p = Image.open('frame'+str(num+1)+'.bmp')
    a,b = p.size
    # print(a)
    data = []
    for y in range(100,b-10,10):
        d = []
        for x in range(410,470,10):
            if (y//10)%2 == 0:
                if p.getpixel((x,y)) >= 200:
                    d.append('1')
                else:
                    d.append('0')
            else:
                if p.getpixel((x,y)) >= 200:
                    d.append('0')
                else:
                    d.append('1')
        data.append(d)
    mode = data[11][5]+data[11][4]+data[10][5]+data[10][4]
    # print(mode)
    length = data[9][5]+data[9][4]+data[8][5]+data[8][4]+data[7][5]+data[7][4]+data[6][5]+data[6][4]+data[5][5]
    # print(length)
    d1 = data[5][4]+data[4][5]+data[4][4]+data[3][5]+data[3][4]+data[2][5]+data[2][4]
    d2 = data[1][5]+data[1][4]+data[0][5]+data[0][4]+data[0][3]+data[0][2]+data[1][3]+data[1][2]
    d3 = data[2][3]+data[2][2]+data[3][3]+data[3][2]+data[4][3]+data[4][2]+data[5][3]+data[5][2]
    d4 = data[6][3]+data[6][2]+data[7][3]+data[7][2]+data[8][3]+data[8][2]+data[9][3]+data[9][2]
    d5 = data[10][3]+data[10][2]+data[11][3]+data[11][2]+data[11][1]+data[11][0]+data[10][1]+data[10][0]
    d6 = data[9][1]+data[9][0]+data[8][1]+data[8][0]+data[7][1]+data[7][0]+data[6][1]+data[6][0]+data[5][1]
    d = d1+d2+d3+d4+d5+d6
    fin = d[:33]
    for i in range(0,len(fin),11):
        y = int(fin[i:i+11],2)%45
        x = int(fin[i:i+11],2)//45
        res += dic[x] + dic[y]
b64_data = number.long_to_bytes(int(res.replace('%',''),16))
while 1:
    b64_data = base64.b64decode(b64_data)
    if b'flag' in b64_data:
        print(b64_data)
        break
```

四、水印_矩阵代表一个字符的ASCII码

第七届“湖湘杯” leaker

解压题目附件，得到一张截图，用 Stegsolve 读取图片，发现图片是 RGBA 模式，而 Alpha 通道只有 255 和 254 两种取值，说明最低位有问题。

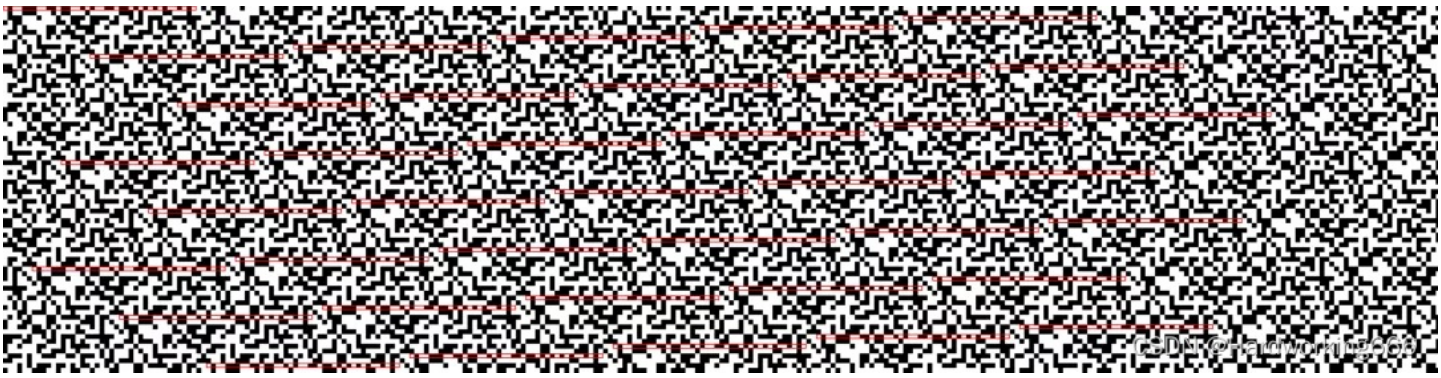
注：阿尔法通道（α Channel或Alpha Channel）是指一张图片的透明和半透明度。可以表示256级的半透明度，因为阿尔法通道有8个比特可以有256种不同的数据表示可能性。当Alpha值为0时，该像素是完全透明的，而当Alpha值为255时，则该像素是完全不透明。



容易发现像素分布存在一定规律，先单独提取出来得到 01 矩阵。（把上面这个图保存后在MATLAB中打开看01值）

对于水印题，我们要做的事情就是找规律。一般来说，为了做到随机截取图片任意一块还能完整读取水印包含的信息，水印会将想要隐藏的信息重复填写，达到抗修改的效果，所以可以尝试先寻找数据的规律。

首先可以发现，行存在重复出现的规律，周期是 76 行，也就是说第 1 行的数据和第 77 行的数据是一样的。我们取出第 1 行所代表的 01 序列的前缀，大概取前 40 个 01 数据就好，然后在图中查找，发现这串 01 序列前缀同样出现在了第 3,5,7… 行中。通过分析我们可以发现数据隔两行就会重复一次，因此我们可以将分析数据的范围缩小成两行。



然后我们再尝试查找这两行有没有什么重复的模式，发现这两行内出现了很多次 2x4 的 0 矩阵，而每两个 0 矩阵之间的信息都是一样的，这让我们又可以将范围缩小。到这里我们就得到了完整的一份信息经过加密后的结果。

接着我们发现第一行每隔 4 位都为 0，可以猜测是 ASCII 码的最高位，于是猜测每 2x4 个矩阵代表一个字符的 ASCII 码。
通过分析 ASCII 码的 01 分布规律，我们发现第二行第二列的格子上的 01 分布是不均匀的，因此我们可以猜测该地方为 ASCII 的第 4 位， 再根据第二行第一列以及第一行第二列的 01 分布情况，结合 [0-9a-zA-Z] 的 ASCII 码在各个二进制位上的 01 分布情况，进行一一对应，最后推测出解读方法：
1 3 5 7
2 4 6 8
解读数据，得到一串 Base64 编码 ZmxhZ3tlZjNkZTlzYS02MTRhLTQ4NjYtYjlzYi0yNDk2MjBiYTk1ZWR9，解码得到 flag 。
注：**Zmxh** Base64解码为**fla**

```
from PIL import Image
import numpy as np
import random
import base64

im = Image.open('leaker.png')
im = np.asarray(im)
x, y, z = im.shape

print(im.shape)

c = []
for j in range(y):
    c.append(1 - im[0, j, 3] // 255)
    c.append(1 - im[1, j, 3] // 255)

c = ''.join([str(x) for x in c])
c = c[:c.find('00000000')]
flag = ''.join([chr(int(c[i:i+8], 2)) for i in range(0, len(c), 8)])

print(flag)
print(base64.b64decode(flag.encode()))
```

本题附件：
https://github.com/chunqiugame/cqb_writeups/raw/master/2021hxb/leaker_4a03eb590cb2db880820e52b475e3def.zip

五、python 0 1转化为二维码

第七届“湖湘杯” wear_a_mask|

首先通过 Stegsolve 查看各个颜色通道各个二进制位上的情况，发现 Blue 通道的最低位有问题。

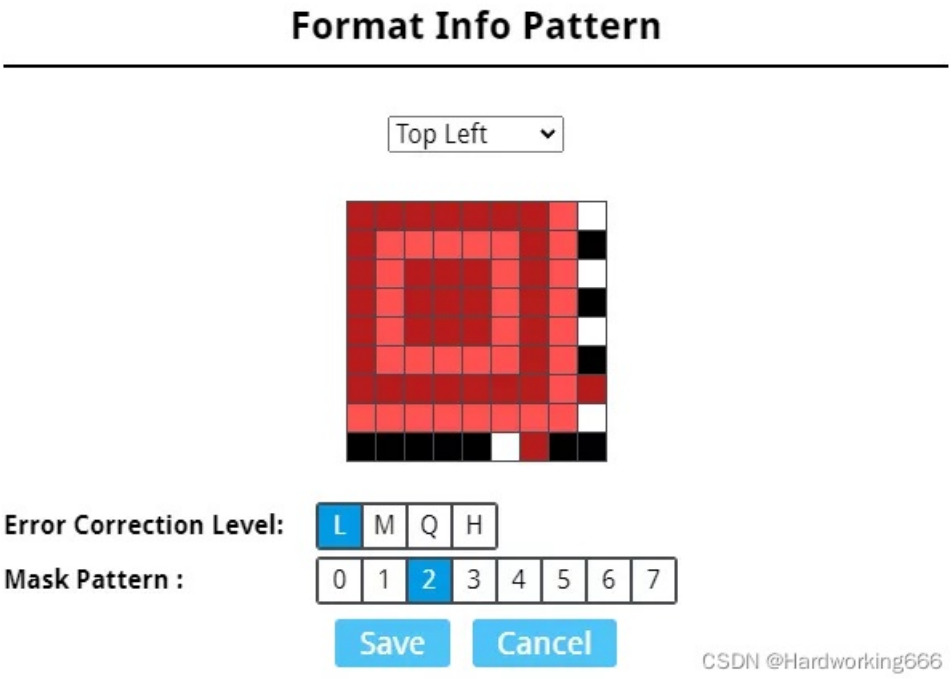
推测数据隐藏在非全白的像素中，并被写入到了 Blue 通道的最低位。

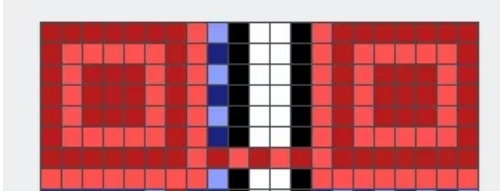
通过观察可以发现在口罩的下面有一段奇怪的花纹，呈现一个有规律的变动。

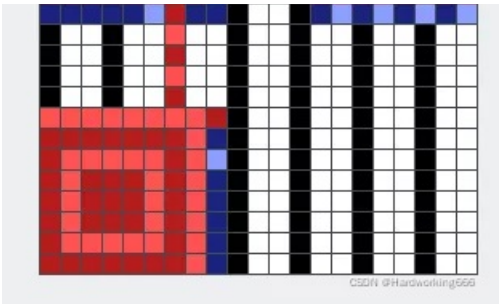
推测数据存在重复模式，通过研究花纹的重复规律可以发现，如果从上往下从左往右阅读数据，每 841 位就会重复一次。

接着问题就是如何解读重复数据。将 841 质因数分解可以得到 29*29，所以尝试将其转成正方形的形状，可以得到一个二维码。

但是这个二维码并不能扫，因为此时的二维码是一个没有与模板（Mask）异或的二维码，所以扫描不出来。结合题目名称「wear_a_mask」，加上双关语「mask」的提示，根据 QRCode 格式所标出来的 Mask Mode 进行 XOR 变换。

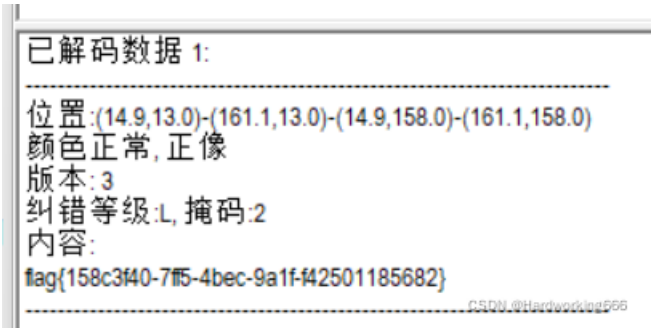






得到最后的 QRCode。

扫描得到 Flag。

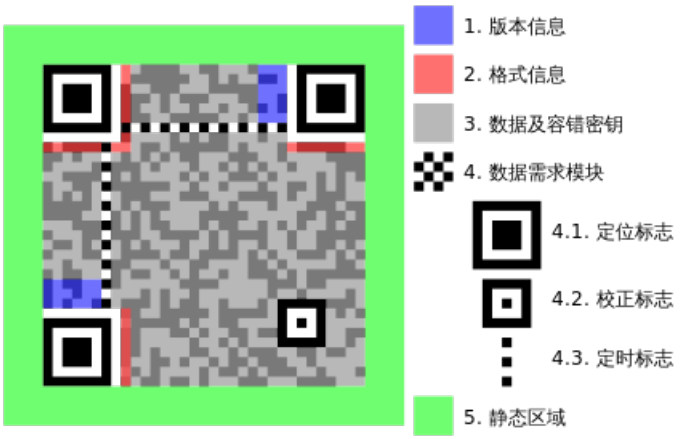


本题附件：https://github.com/chunqiugame/cqb_writeups/raw/master/2021hxb/mask_a2c5c7556dc66ca474a412e4f90af3b9.zip

CTF python 0 1转化为二维码：

```
from PIL import Image
from zlib import *

MAX = 36 # 数字的长度为一个整数的平方（如36^2=1296)
pic = Image.new("RGB",(MAX,MAX))
str ="111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
111111000000011110010000000011000000011111011111111100110000100111111111011111010000011111110010010011000001011
110101111111010101100000011111010111110101000111010100100111010001010111110101000110110101010111110001010111110
010001011001011101111100010101111111111111010101010000110111111111111111111111111010101100110001001111111100100
111001010011111110110110111111100000101011110101011000111110011111101010111101001000001100001111111111011100101
001100110011000000111111111111111111111001110111110000111111100000000000000000011011011101011111000101101101111
101001111000010111111100010101111010010001100101001011111101010101011001000110001101110001111101111010011101110
100001111000011111001010011110000000100010110100011111001001000011011000010000100111011111000101100011010111101
011011100111111100100011101100100010111111111111111111111101000101000001011111111111110101000111000011111111100
010101111101010001101111111111111000101011111010100010101001111111111000101011110101111101111111111111111111101
011111010000011010011111111111000001011111011111111111111111111111111111011111000000011001111111111110000000111
111111111111111111111111111111111111111111111111111111111111111111111111"

i=0
for y in range(0,MAX):
    for x in range(0,MAX):
        if(str[i] == '1'):
            pic.putpixel([x,y],(0,0,0))
        else:pic.putpixel([x,y],(255,255,255))
        i = i+1
pic.show()
pic.save("flag.png")
```

记事本打开ctrl+H，0替换□，1替换■



六、stegpy隐写爆破

第四届2021美团网络安全高校挑战赛MT-CTF线上初赛Misc: Boom部分wp:

通过stegsolve可以确定是颜色通道最低两位的隐写，而stegpy则是这种隐写方式。

解密需要密码，可以写脚本交互爆破，同时根据图片名boom，也联想到爆破。

```python
#!/usr/bin/env python3
# Module for processing images, audios and the least significant bits.

import numpy
from PIL import Image
from . import crypt

MAGIC_NUMBER = b'stegv3'

class HostElement:
    """ This class holds information about a host element. """
    def __init__(self, filename):
        self.filename = filename
        self.format = filename[-3:]
        self.header, self.data = get_file(filename)

    def save(self):
        self.filename = '_' + self.filename
        if self.format.lower() == 'wav':
            sound = numpy.concatenate((self.header, self.data))
            sound.tofile(self.filename)
        elif self.format.lower() == 'gif':
            gif = []
            for frame, palette in zip(self.data, self.header[0]):
                image = Image.fromarray(frame)
                image.putpalette(palette)
                gif.append(image)
            gif[0].save(self.filename, save_all=True, append_images = gif[1:], loop=0, duration=self.header[1])
        else:
            if not self.filename.lower().endswith(('png', 'bmp', 'webp')):
                print("Host has a lossy format and will be converted to PNG.")
                self.filename = self.filename[:-3] + 'png'
            image = Image.fromarray(self.data)
            image.save(self.filename, lossless=True, minimize_size=True, optimize=True)
        print("Information encoded in {}.".format(self.filename))

    def insert_message(self, message, bits=2, parasite_filename=None, password=None):
        raw_message_len = len(message).to_bytes(4, 'big')
        formatted_message = format_message(message, raw_message_len, parasite_filename)
        if password:
            formatted_message = crypt.encrypt_info(password, formatted_message)
        self.data = encode_message(self.data, formatted_message, bits)

    def read_message(self, password=None):
        msg = decode_message(self.data)
```

```
        if password:
            try:
                salt = bytes(msg[:16])
                msg = crypt.decrypt_info(password, bytes(msg[16:]), salt)
            except:
                return("Wrong password.")

        check_magic_number(msg)
        msg_len = int.from_bytes(bytes(msg[6:10]), 'big')
        filename_len = int.from_bytes(bytes(msg[10:11]), 'big')

        start = filename_len + 11
        end = start + msg_len
        end_filename = filename_len + 11
        if(filename_len > 0):
            filename = '_' + bytes(msg[11:end_filename]).decode('utf-8')

        else:
            text = bytes(msg[start:end]).decode('utf-8')
            print(text)
            return

        with open(filename, 'wb') as f:
            f.write(bytes(msg[start:end]))

        print('File {} succesfully extracted from {}'.format(filename, self.filename))

    def free_space(self, bits=2):
        shape = self.data.shape
        self.data.shape = -1
        free = self.data.size * bits // 8
        self.data.shape = shape
        self.free = free
        return free

    def print_free_space(self, bits=2):
        free = self.free_space(bits)
        print('File: {}, free: (bytes) {:,}, encoding: 4 bit'.format(self.filename, free, bits))

def get_file(filename):
    ''' Returns data from file in a list with the header and raw data. '''
    if filename.lower().endswith('wav'):
        content = numpy.fromfile(filename, dtype=numpy.uint8)
        content = content[:10000], content[10000:]
    elif filename.lower().endswith('gif'):
        image = Image.open(filename)
        frames = []
        palettes = []
        try:
            while True:
                frames.append(numpy.array(image))
                palettes.append(image.getpalette())
                image.seek(image.tell()+1)
        except EOFError:
            pass
        content = [palettes, image.info['duration']], numpy.asarray(frames)
    else:
        image = Image.open(filename)
        if image.mode != 'RGB':
            image = image.convert('RGB')
```

```
            image = image.convert('RGB')
        content = None, numpy.array(image)
    return content

def format_message(message, msg_len, filename=None):
    if not filename: # text
        message = MAGIC_NUMBER + msg_len + (0).to_bytes(1, 'big') + message
    else:
        filename = filename.encode('utf-8')
        filename_len = len(filename).to_bytes(1, 'big')
        message = MAGIC_NUMBER + msg_len + filename_len + filename + message
    return message;

def encode_message(host_data, message, bits):
    ''' Encodes the byte array in the image numpy array. '''
    shape = host_data.shape
    host_data.shape = -1, # convert to 1D
    uneven = 0
    divisor = 8 // bits

    print("Host dimension: {:,} bytes".format(host_data.size))
    print("Message size: {:,} bytes".format(len(message)))
    print("Maximum size: {:,} bytes".format(host_data.size // divisor))

    check_message_space(host_data.size // divisor, len(message))

    if(host_data.size % divisor != 0): # Hacky way to deal with pixel arrays that cannot be divided evenly
        uneven = 1
        original_size = host_data.size
        host_data = numpy.resize(host_data, host_data.size + (divisor - host_data.size % divisor))

    msg = numpy.zeros(len(host_data) // divisor, dtype=numpy.uint8)

    msg[:len(message)] = list(message)

    host_data[:divisor*len(message)] &= 256 - 2 ** bits # clear last bit(s)
    for i in range(divisor):
        host_data[i::divisor] |= msg >> bits*i & (2 ** bits - 1) # copy bits to host_data

    operand = (0 if (bits == 1) else (16 if (bits == 2) else 32))
    host_data[0] = (host_data[0] & 207) | operand # 5th and 6th bits = log_2(bits)

    if uneven:
        host_data = numpy.resize(host_data, original_size)

    host_data.shape = shape # restore the 3D shape

    return host_data

def check_message_space(max_message_len, message_len):
    ''' Checks if there's enough space to write the message. '''
    if(max_message_len < message_len):
        print('You have too few colors to store that message. Aborting.')
        exit(-1)
    else:
        print('Ok.')

def decode_message(host_data):
    ''' Decodes the image numpy array into a byte array. '''
    host_data.shape = -1, # convert to 1D
```

```
    bits = 2 ** ((host_data[0] & 48) >> 4) # bits = 2 ^ (5th and 6th bits)
    divisor = 8 // bits

    if(host_data.size % divisor != 0):
        host_data = numpy.resize(host_data, host_data.size + (divisor - host_data.size % divisor))

    msg = numpy.zeros(len(host_data) // divisor, dtype=numpy.uint8)

    for i in range(divisor):
        msg |= (host_data[i::divisor] & (2 ** bits - 1)) << bits*i

    return msg

def check_magic_number(msg):
    if bytes(msg[0:6]) != MAGIC_NUMBER:
        print(bytes(msg[:6]))
        print('ERROR! No encoded info found!')
        exit(-1)

if __name__ == '__main__':
    message = 'hello'.encode('utf-8')
    host = HostElement('gif.gif')
    host.insert_message(message, bits=4)
    host.save()
```

此脚本为原python库 stegpy脚本修改，主要改动读取隐写信息函数read_message一段，命名为lsb2.py，放在stegpy库目录下/usr/local/lib/python3.6/dist-packages/stegpy
网上找常见的弱口令字典进行爆破

```
#coding=utf-8
from stegpy import lsb2

host = lsb2.HostElement('flag.png')
dic = open('password1.txt').readlines()

for i in range(len(dic)):
    tmp = host.read_message(dic[i][:-1])
    if tmp != "Wrong password.":
        break
    else:
        print(i, dic[i][:-1])
```

爆破得到密码是123123@@@
隐写内容是783d793c313030
解开压缩包得到一张图片，无法正常观看。很明显是crc校验错误，爆破图片的正确宽高。
第四届2021美团网络安全高校挑战赛MT-CTF线上初赛官方wp