# CTF 图像隐写Python脚本处理

转载





CTF中经常会遇到很多图片的隐写题目需要使用脚本来解题，最常用到的就是使用python中的PIL库，所以如果要更好的解出图片隐写相关处理的题目，掌握好这个库的使用是必要的。本期就来给大家介绍下这个库的基本使用和几道图片题目的解题思路。

### 0x00 PIL vs Pillow

首先介绍PIL这个库，PIL：Python Imaging Library，该库虽然是第三方库，但是俨然已经成为了图像处理的官方库。官方手册：https://pillow.readthedocs.io/en/latest/handbook/tutorial.html

由于PIL仅支持到Python 2.7，加上年久失修，于是一群志愿者在PIL的基础上创建了兼容的版本，名字叫Pillow，支持最新Python 3.x，又加入了许多新特性，因此，我们可以直接安装使用Pillow。

github: https://github.com/python-pillow/Pillow

安装：win/lin/mac

```
pip install Pillow
```

### 0x01 Image 类

用到最多的是Image 这个类，一般我们会按照下面这个格式引用，下面就介绍下这个类的一些基本用法。

```
from PIL import Image
```

**1.加载图片，使用方法open**

```
from PIL import Image
im = Image.open('Mycat.jpg')
```

im是一个Image对象，属性有format，size，mode。format是格式，size 是一个元组，表示(宽，高)，mode则指的图片的模式。

```
from PIL import Image
im = Image.open('Mycat.jpg')
print im.format,im.size,im.mode
```

console输出如下：

```
JPEG (245, 280) RGB
```

**2.呈现图片，方法show方便用来调试和测试。**

```
from PIL import Image
im = Image.open('Mycat.jpg')
im.show()
```

**3.图片的读和写**

读文件用Image.open()，保存文件用Image.save()，也可以用save方法来进行图片的格式转换。使用os模块中的os.path.splitext()方法可以讲文件名和扩展名分离开来，下面的代码能够把jpg格式的图片转为png格式。

```
infile = 'Mycat.jpg'
f,e = os.path.splitext(infile)
outfile = f + '.png'
try:
Image.open(infile).save(outfile)
except IOError:
print "cannot convert",infile
```

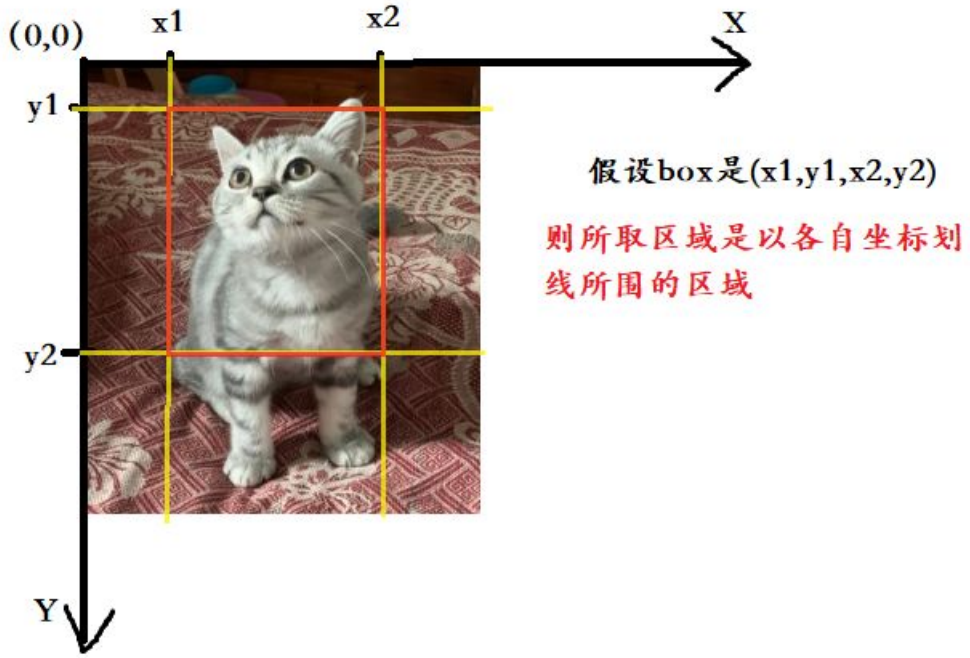
**4.图片的剪切，黏贴**

**(1) 图片剪切**

从一张图片中剪切出一块区域，比如从图片提取矩形，使用crop()方法。

```
im = Image.open('Mycat.jpg')
box = (150,150,245,280)
region = im.crop(box)
region.show()
```

关于这个box，这是一个4元的坐标数组，坐标轴是左上角是(0,0)的卡迪尔坐标系。 假设box是(x1,y1,x2,y2)，则所取区域是以各自坐标划线所围的区域。




**（2）图片黏贴**

图片的黏贴就是将一张图覆盖到另一张图上面。黏贴的方法是paste()。格式为：paste(要贴的图片，要贴的图片的4元坐标组成的区域)。如下面，我们把Mycat.jpg这张图片，取区域(50,50,200,200)，将该区域旋转180度后贴到原来的位置。

```
im = Image.open('Mycat.jpg')
box = (50,50,200,200)
region = im.crop(box)
# 将图片逆序旋转180后，黏贴到原来复制的位置
region = region.transpose(Image.ROTATE_180)
im.paste(region,box)
im.show()
```

### 5.图像序列

当处理GIF这种包含多个帧的图片，称之为序列文件，PIL会自动打开序列文件的第一帧。而使用seek和tell方法可以在不同帧移动。tell是帧数，而seek是取当前帧数的图片。

使用while循环：

```
from PIL import Image
im = Image.open("laopo.gif")
im.seek(1)
im.show()

try:
while 1:
im.seek(im.tell()+1)
im.show()
except EOFError:
pass
```

如果要使用for循环，可以使用ImageSequence模块的Iterator方法。

```
from PIL import Image
from PIL import ImageSequence

im = Image.open("laopo.gif")
for frame in ImageSequence.Iterator(im):
frame.show()
```

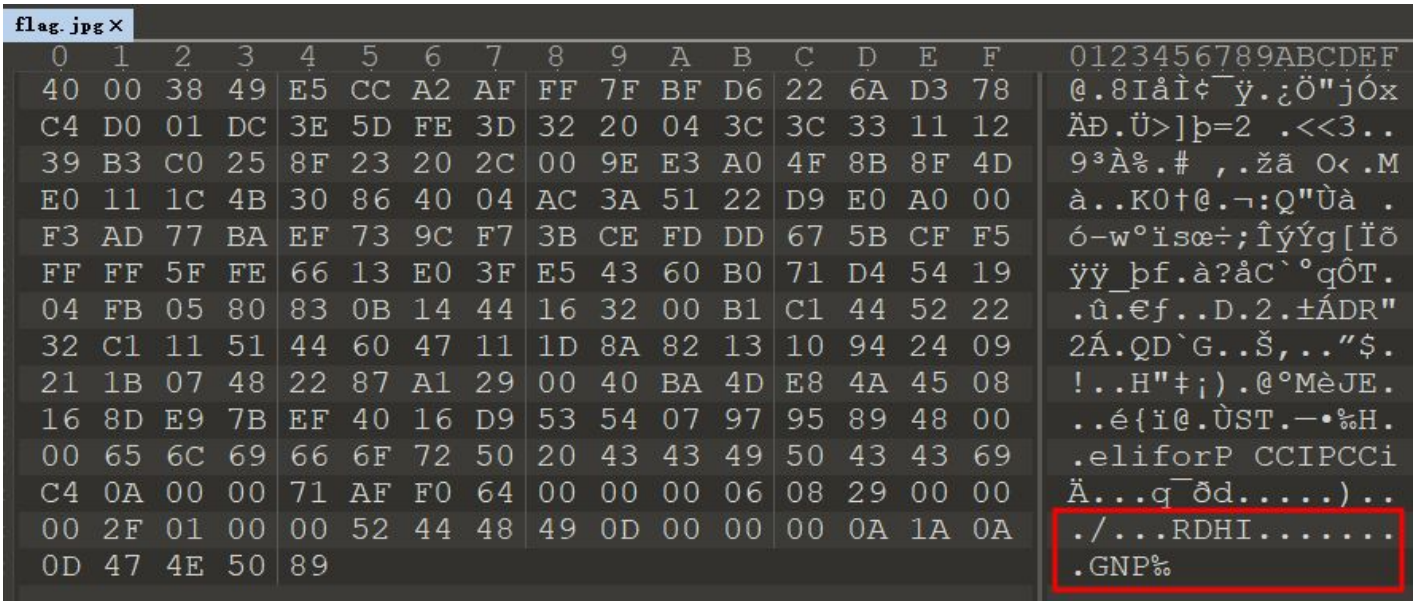
### 6.读取像素和修改像素

```
from PIL import Image
img = Image.open('Mycat.jpg')
width , height = img.size
for i in range(0,width):
for j in range(0,height):
tmp = img.getpixel((i,j))
img.putpixel((i,j),(0,0,tmp[2]))
img.show()
```

## 0x02 CTF 图片隐写实战

### 题目1：图片内容读取

题目下载地址链接：https://pan.baidu.com/s/1cZ4GFleaG7TO4LaWOquqXQ 提取码：5lu7 。本题是一道图片内容逆序处理的题目，首先用16进制编辑器打开该图片，发现逆序该图片也是一张PNG。




这种逆序处理相对比较简单，我们把图片当成文本就行了，只是操作的时候注意模式是二进制的模式，然后用[::-1]这种写法来逆序。

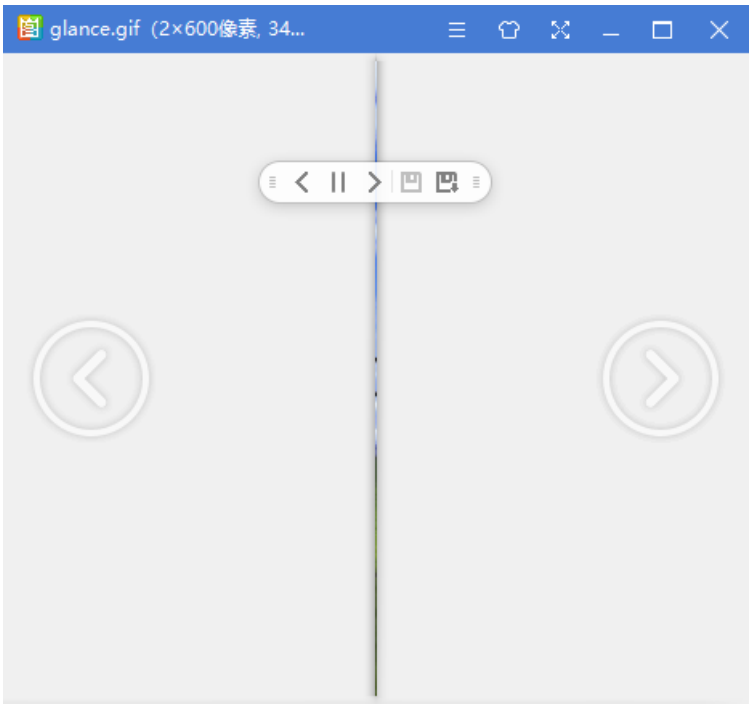
```
# coding:utf-8
# author:Reborn

f1 = open('flag.jpg','rb+')
f2 = open('fla.jpg','wb+')
f2.write(f1.read()[::-1])
f1.close()
f2.close()
```

得到flag:

```
flag{Mirr0r_R3f3ct1on_H1dd3n_f14g}
```

**题目2：GIF图片分离并合并**

题目下载地址链接：https://pan.baidu.com/s/1-11bTiu2URnJrxaz937VOw 提取码：18d2 。




本题是一个GIF题目，观察该题是一个非常细的图片在闪烁。如果直接观察的话，可以发现这个GIF其实是一张图片被分割成很多细小的长碎片，然后制作成的GIF图片。

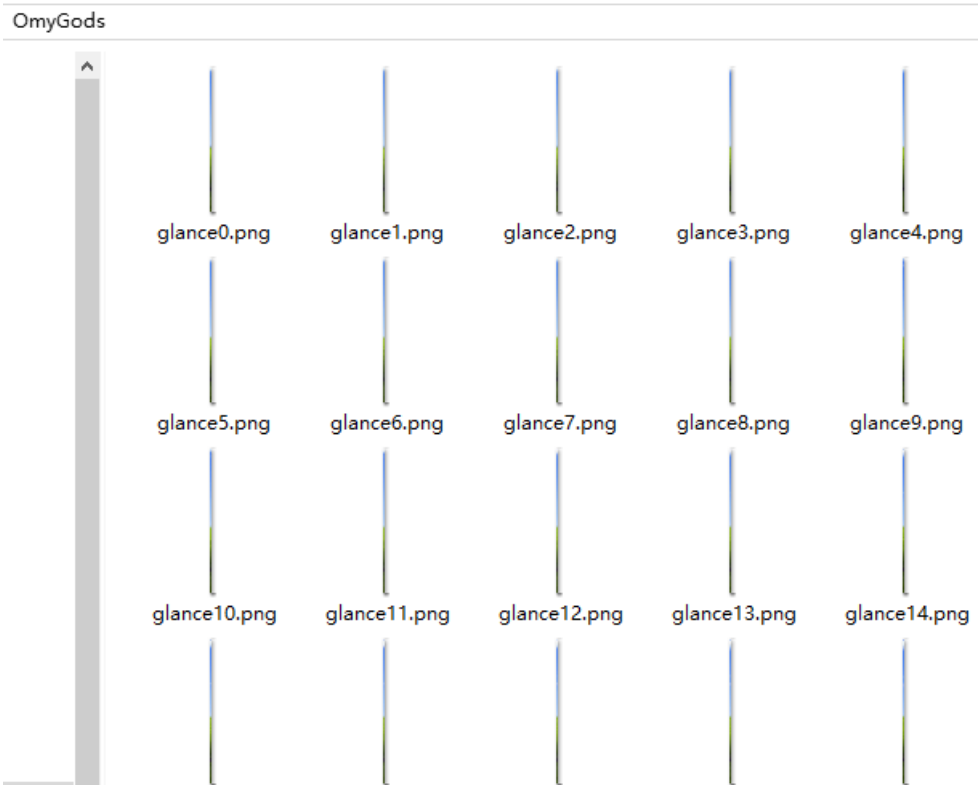
**解题思路：**

将该GIF的每一帧提取出来，然后在用黏贴的方法合并起来。

将每一帧提取出来并保存在OmyGods这个文件夹下，所以要提前建立该文件夹。

```
# coding:utf-8
# author:Reborn
from PIL import Image

savepath = "OmyGods\\"
path = '.\\'

im = Image.open('glance.gif')
try:
im.save(savepath+'glance{:d}.png'.format(im.tell()))
while True:
im.seek(im.tell()+1)
im.save(savepath+'glance{:d}.png'.format(im.tell()))
except:
pass
```

执行改脚本，得到每一帧。



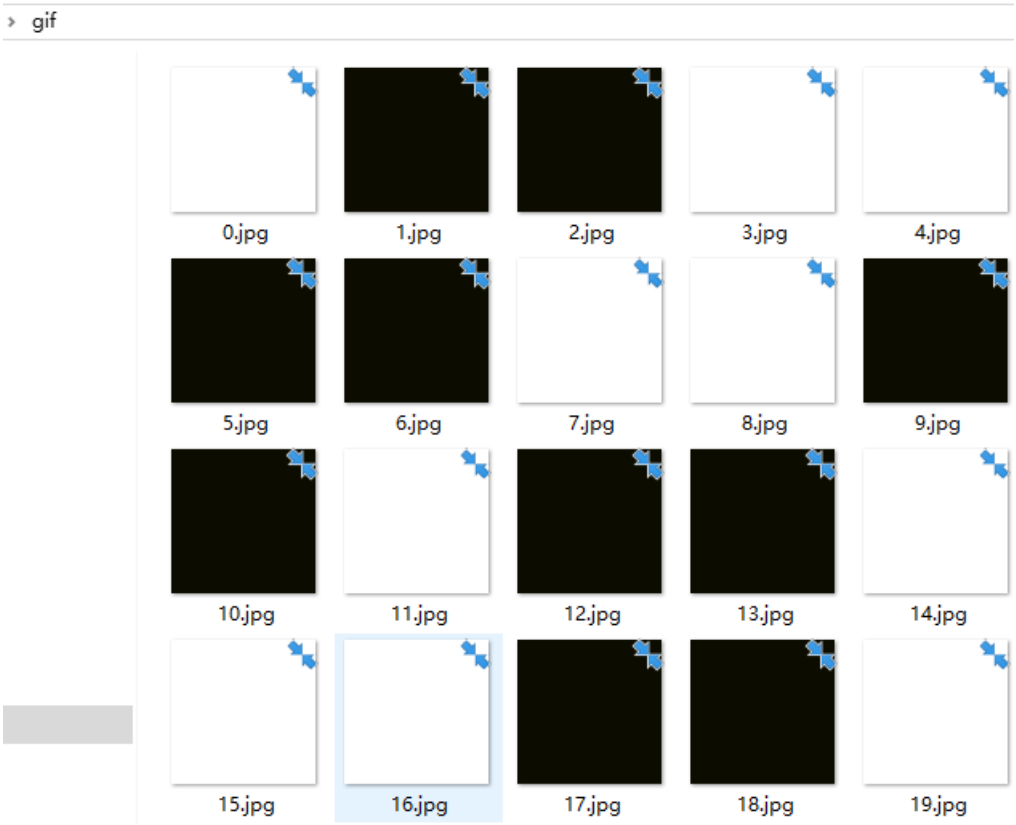

而接下来我的思路就是，创建一张大图，然后将每一帧依次黏贴上去。 如下代码，im就是创建好的大图，然后使用im.paste(image,(width,0,2+width,600))，的方法，注意这个4元坐标的变化，高度不变，宽度每次加2，因为每一帧的宽度是2。

```
# coding:utf-8
# author:Reborn
from PIL import Image
path = "OmyGods\\"
save_path = '.\\'

im = Image.new('RGBA',(2*201,600))

imagefile = []
width = 0
for i in range(201):
imagefile.append(Image.open(path+'Frame'+str(i)+'.png'))

for image in imagefile:
im.paste(image,(width,0,2+width,600))
width = width +2
im.save(save_path+'OmyGod.png')
im.show()
```

运行该脚本，得到flag的图片。


**题目3：判断图片颜色**

题目下载地址链接：https://pan.baidu.com/s/1JpYakfO7Dk9ecq_kpw5EvQ 提取码：6uqw。




本题得到的一堆的黑白图片，这种题目的一种考察方式是，黑和白表示2中信息，可以组成01010101...的信息流。

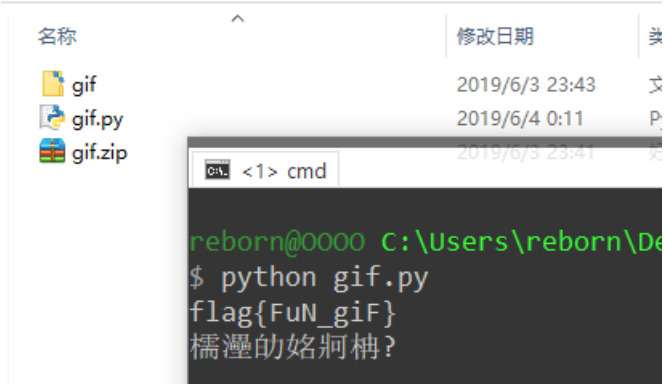
一共有104张黑白图片，我们可以用getcolors()这个方法来获取图片的信息。 一般会返回一个元组(count, (r,g,b))。 该元组第一个元素count 代表该颜色像素出现的次数，第二个元素表示(rgb)。

```
0.jpg:(白色)
(46656, (255, 255, 255))
1.jpg:(黑色)
(46656, (12, 12, 0))
2.jpg:(黑色)
(46656, (12, 12, 0))
3.Jpg:(白色)
(46656, (255, 255, 255))
```

通过比较得出，白色图片为(46656, (255, 255, 255)),黑色图片为(46656, (12, 12, 0))，通过该规律编写脚本：

```
# coding:utf-8
# author:Reborn
from PIL import Image
path = "gif\\"
save_path = '.\\'

sumDo = '0b'
sumNo = '0b'
imagefile=[]
for i in range(104):
imagefile.append(Image.open(path+str(i)+'.jpg'))
for image in imagefile:
if image.getcolors()[0][1][0] == 12:
sumDo +='1'
sumNo +='0'
else:
sumDo +='0'
sumNo +='1'

print (hex(eval(sumDo))[2:-1]).decode('hex')
print (hex(eval(sumNo))[2:-1]).decode('hex')
```

因为有两种可能，所以将两种可能都打印出来。




**题目4：LSB处理**

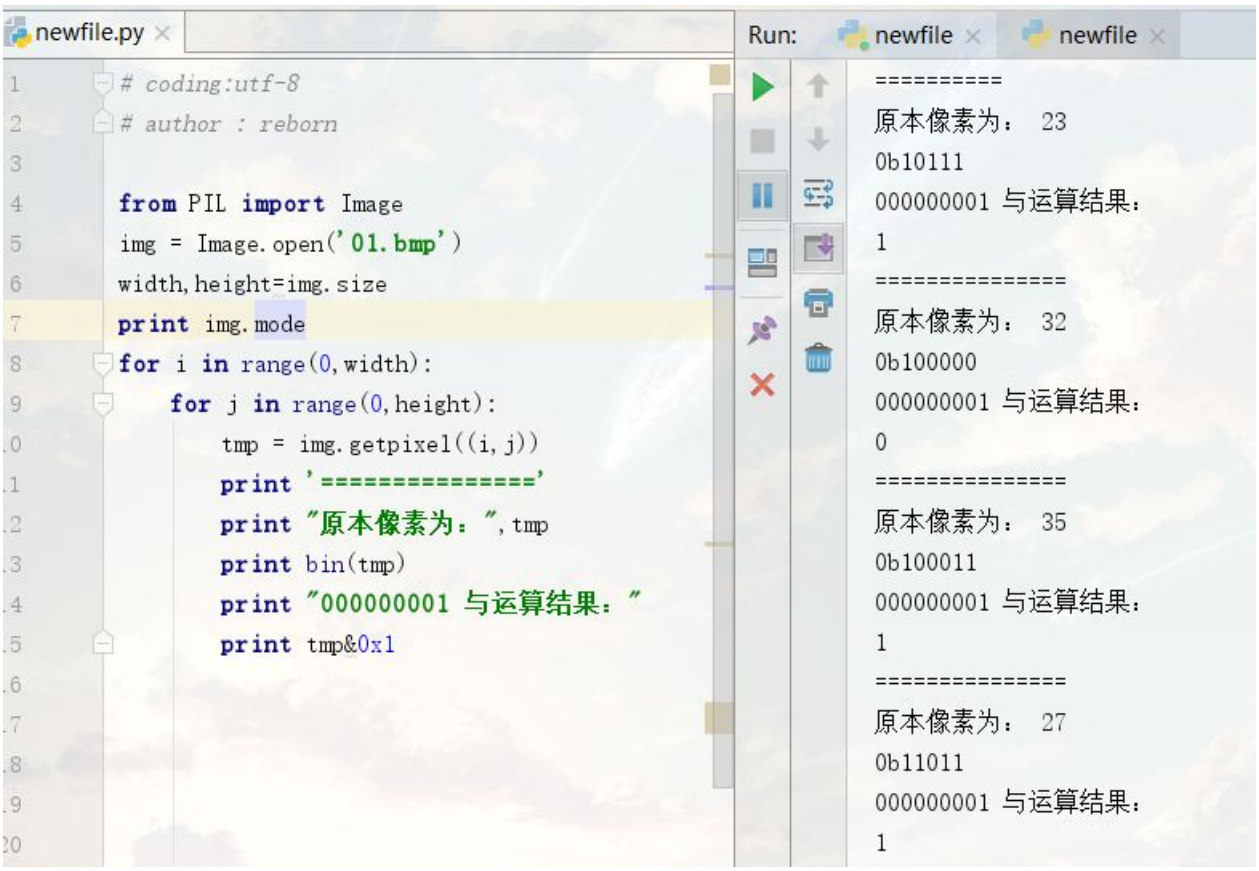
题目下载地址链接：https://pan.baidu.com/s/1rgg-qEVj7dT_TKXpHvAjag 提取码：m7tk 。

本题题目考察的是LSB隐写，所以图片是一张模式为L的二值图，所以像素没有r,g,b这么多，只有1个。

因为该题LSB是将原本的像素转8位2进制，将8位2进制的左后一位置0或者置1来隐写数据，所以我们可以枚举所有像素，当该位像素最后一位不为0时，置为255的黑点。

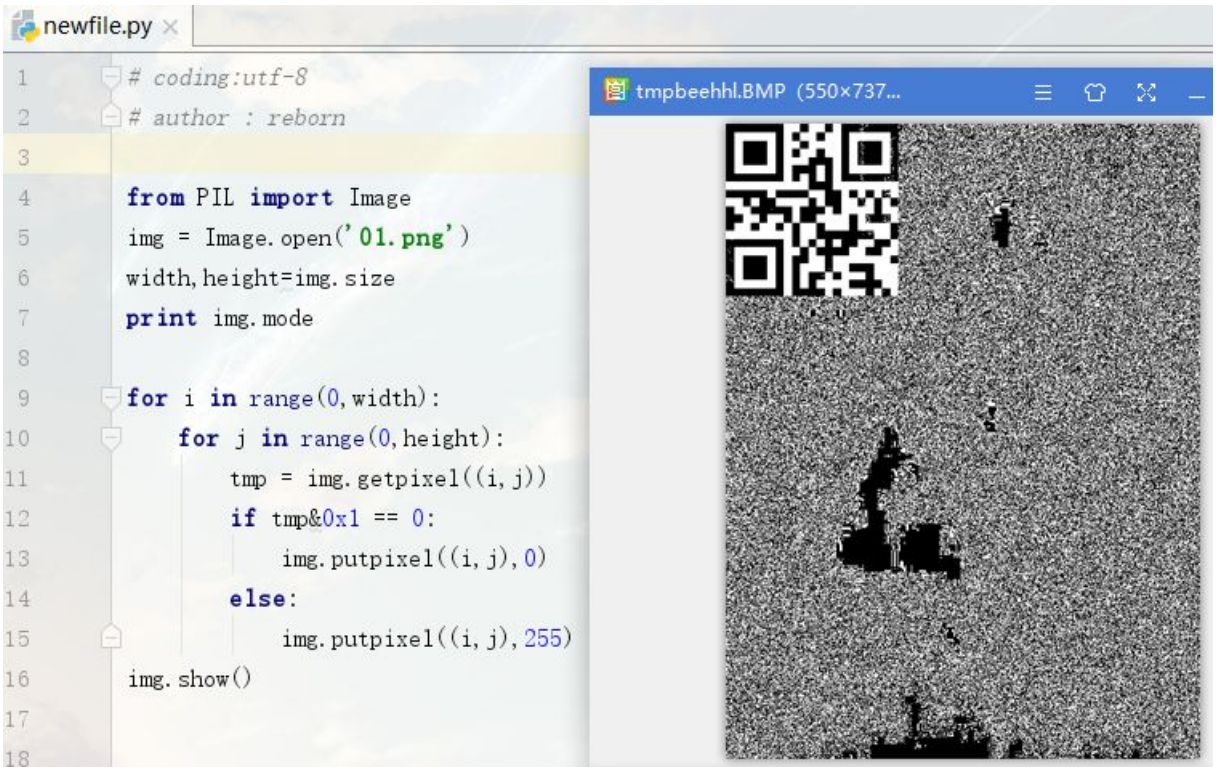
判断方式我们可以取像素置和0x1进行与运算，脚本如下。

```
# coding:utf-8
# author : reborn

from PIL import Image
img = Image.open('01.bmp')
width,height=img.size
print img.mode
for i in range(0,width):
for j in range(0,height):
tmp = img.getpixel((i,j))
print '==============='
print "原本像素为：",tmp
print bin(tmp)
print "000000001 与运算结果："
print tmp&0x1
```




然后通过运算，如果隐写值为0则整个像素置0（白），如果隐写值为1则整个像素置为255（黑）。

```
# coding:utf-8
# author : reborn

from PIL import Image
img = Image.open('01.bmp')
width,height=img.size
for i in range(0,width):
for j in range(0,height):
tmp = img.getpixel((i,j))
if tmp&0x1 == 0:
img.putpixel((i,j),0)
else:
img.putpixel((i,j),255)
img.show()
```




最终获得一张二维码，扫描该二维码可得flag。

**0x03 最后**

本期图像隐写Python脚本处理就介绍到这里，下期将会带来更多的图片隐写技巧和案例，我们下期见。