MISC总结——隐写术(四）

转载

weixin_34190136 于 2018-08-01 22:10:00 发布 1250 收藏 7
文章标签： python 网络 操作系统
原文链接：https://yq.aliyun.com/articles/649046

这一篇是继上一篇之后的另一篇关于隐写术在ctf比赛中常见的套路问题：

MISC总结——隐写术(一)详情请见：https://www.cnblogs.com/lxz-1263030049/p/9388511.html

MISC总结——隐写术(二)详情请见：https://www.cnblogs.com/lxz-1263030049/p/9388602.html

MISC总结——隐写术(三)详情请见：https://www.cnblogs.com/lxz-1263030049/p/9392923.html

这应该是关于隐写术问题的最后一篇文章了！！

如果有喜欢的请关注一下哦！！！！

本文参考自：先知社区：https://xz.aliyun.com/t/1875

隐写术介绍：

隐写术是关于信息隐藏，即不让计划的接收者之外的任何人知道信息的传递事件（而不只是信息的内容）的一门技巧与科学。

英文写作Steganography，而这篇内容将带大家了解一下CTF赛场上常见的图片隐写方式，以及解决方法。有必要强调的是，隐写术与密码编码是完全不同的概念。

第一部分：基于DCT域的JPG图片隐写

知识背景：

JPEG图像格式使用离散余弦变换（Discrete Cosine Transform，DCT）函数来压缩图像，而这个图像压缩方法的核心是：

通过识别每个8×8像素块中相邻像素中的重复像素来减少显示图像所需的位数，并使用近似估算法降低其冗余度。

因此，我们可以把DCT看作一个用于执行压缩的近似计算方法。因为丢失了部分数据，所以DCT是一种有损压缩（Loss Compression）技术，但一般不会影响图像的视觉效果。

在这个隐写家族中，常见的隐写方法有JSteg、JPHide、Outguess、F5等等

Stegdetect

实现JPEG图像Jphide隐写算法工具有多个，比如由Neils Provos开发通过统计分析技术评估JPEG文件的DCT频率系数的隐写工具Stegdetect，

它可以检测到通过JSteg、JPHide、OutGuess、Invisible Secrets、F5、appendX和Camouflage等这些隐写工具隐藏的信息，

并且还具有基于字典暴力破解密码方法提取通过Jphide、outguess和jsteg-shell方式嵌入的隐藏信息。

JPHS

一款JPEG图像的信息隐藏软件JPHS，它是由Allan Latham开发设计实现在Windows和Linux系统平台针对有损压缩JPEG文件进行信息加密隐藏和探测提取的工具。

软件里面主要包含了两个程序JPHIDE和JPSEEK， JPHIDE程序主要是实现将信息文件加密隐藏到JPEG图像功能，

而JPSEEK程序主要实现从用JPHIDE程序加密隐藏得到的JPEG图像探测提取信息文件，Windows版本的JPHS里的JPHSWIN程序具有图形化操作界面且具备JPHIDE和JPSEEK的功能。

Outguess

Outgusee算法是Niels Provos针对Jsteg算法的缺陷提出的一种方法：

- 嵌入过程不修改ECT系数值为0，1的DCT系数，利用为随机数发生器产生间隔以决定下一个要嵌入的DCT系数的位置
- 纠正过程消除对效应的出现

对应的，也有针对该算法的隐写工具，名字也叫Outguess。

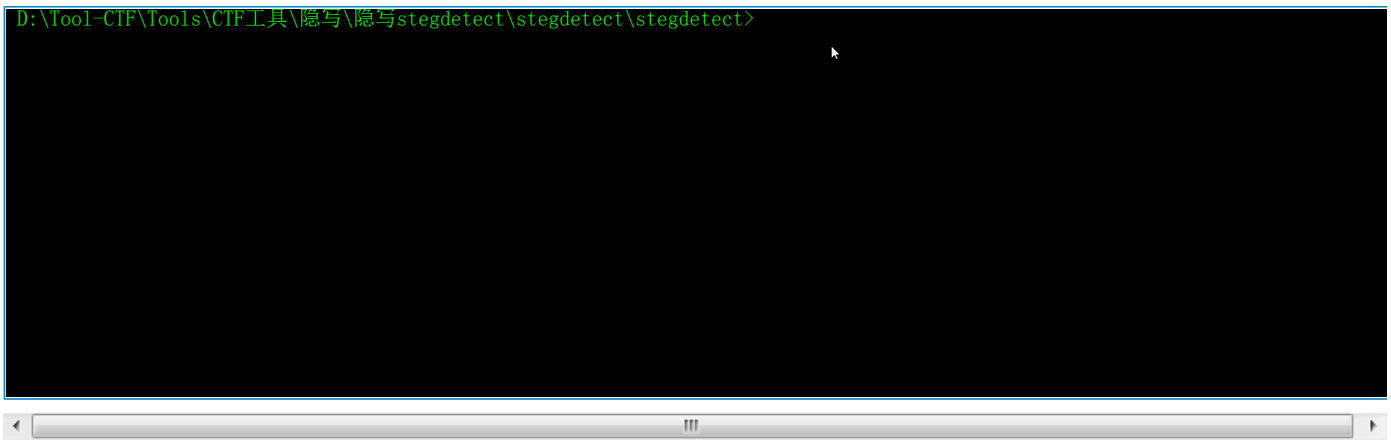
JPHS

```
在实验机中找到隐写术目录，打开图片隐写，打开图片隐写第四部分文件夹
在该文件夹找到JPhide.jpg
双击打开图片，我们先确认一下图片内容并没有什么异常
使用Stegdetect对图片的隐写方式进行检测
从结果中得知是使用JPHide的隐写，使用JPHS工具对隐写信息进行提取
最后打开保存的文件，flag是flag{Good_you_got_it}
```

Stegdetect的指令介绍

-q 仅显示可能包含隐藏内容的图像。
-n 启用检查JPEG文件头功能，以降低误报率。如果启用，所有带有批注区域的文件将被视为没有被嵌入信息。如果JPEG文件的JFIF标识符中的版本号不是1.1，则禁用OutGuess检测。
-s 修改检测算法的敏感度，该值的默认值为1。检测结果的匹配度与检测算法的敏感度成正比，算法敏感度的值越大，检测出的可疑文件包含敏感信息的可能性越大。
-d 打印带行号的调试信息。
-t 设置要检测哪些隐写工具（默认检测jopi），可设置的选项如下：
j 检测图像中的信息是否是用jsteg嵌入的。
o 检测图像中的信息是否是用outguess嵌入的。
p 检测图像中的信息是否是用jphide嵌入的。
i 检测图像中的信息是否是用invisible secrets嵌入的。
-V 显示软件版本号。
如果检测结果显示该文件可能包含隐藏信息，那么Stegdetect会在检测结果后面使用1～3颗星来标识隐藏信息存在的可能性大小，3颗星表示隐藏信息存在的可能性最大。

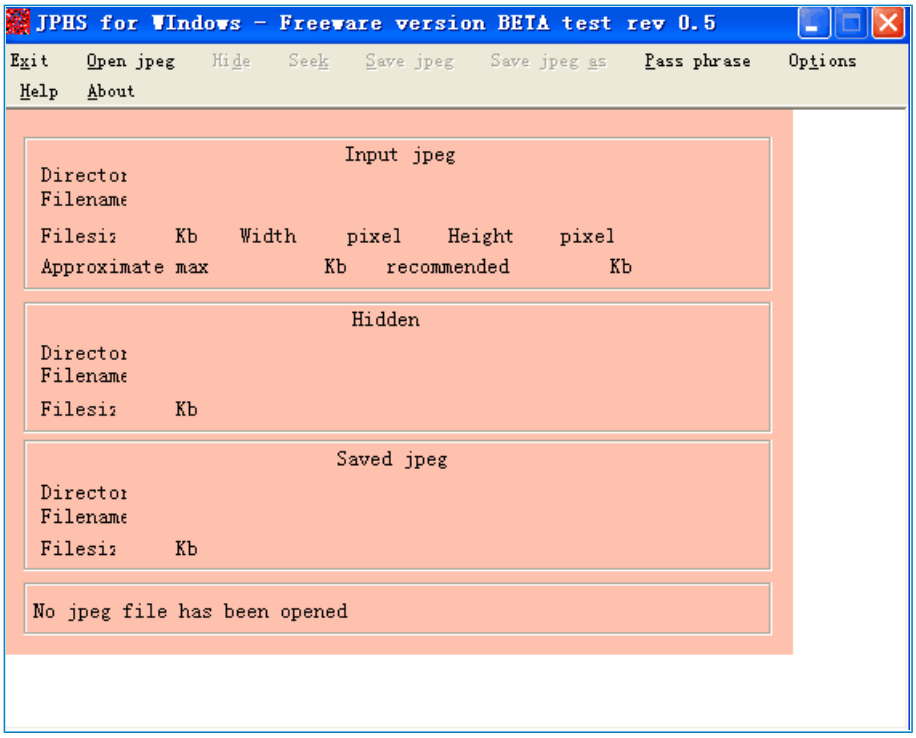
首先，在cmd中运行，Stegdetect，对目标图片进行检测



发现结果显示是jphide隐写的可能性很大。

接着，我们使用工具JPHS提取信息

这是一款针对Jphide算法的隐写工具，正如对症下药，我们也得用对工具，才能更好的解决问题。



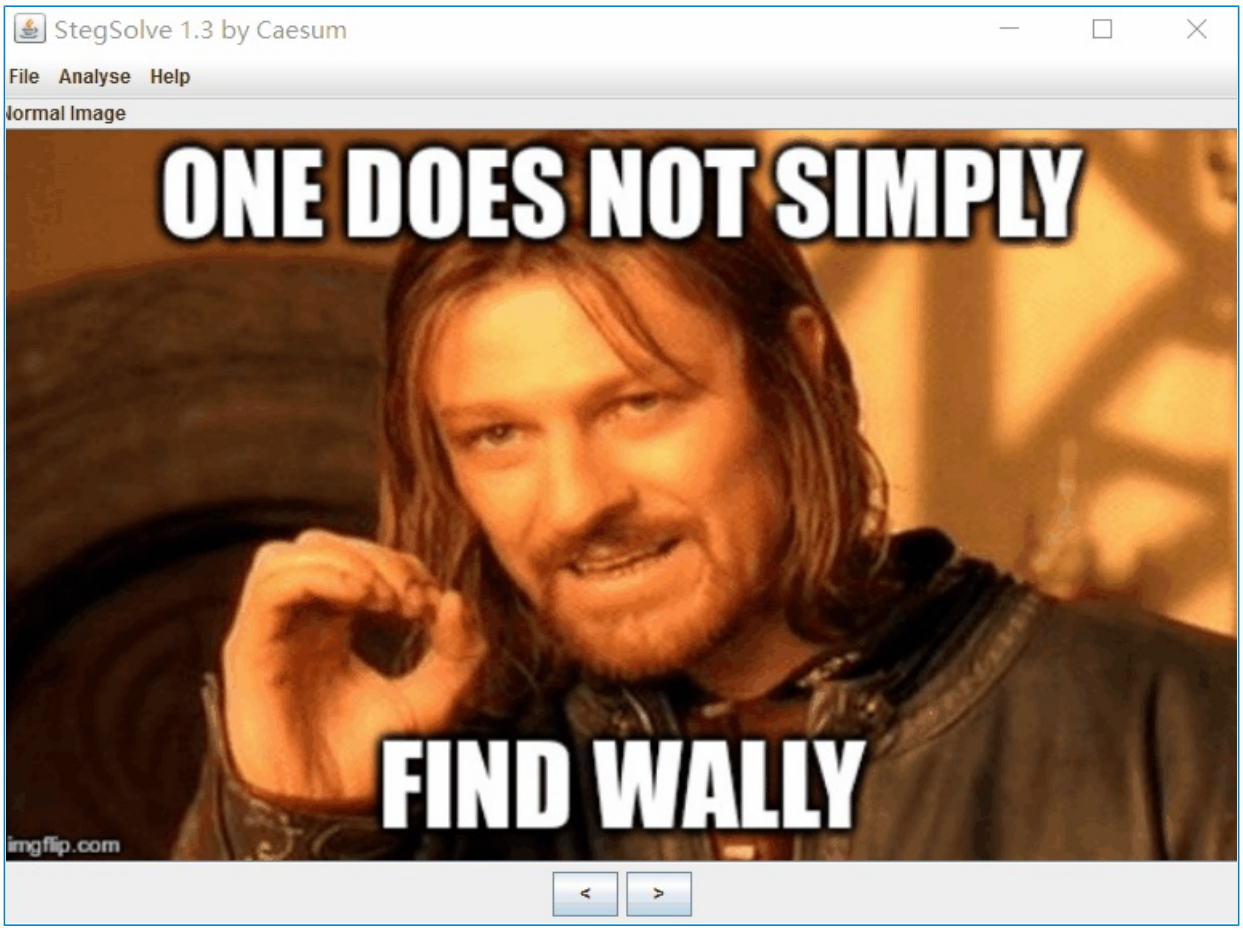
首先，在使用JPHS打开图片，点击Seek功能，紧接着会弹出一个密码的输入框，我们这里默认为空口令，直接点击OK，将提取出来的信息保存为flag.txt。

Outguess

实验

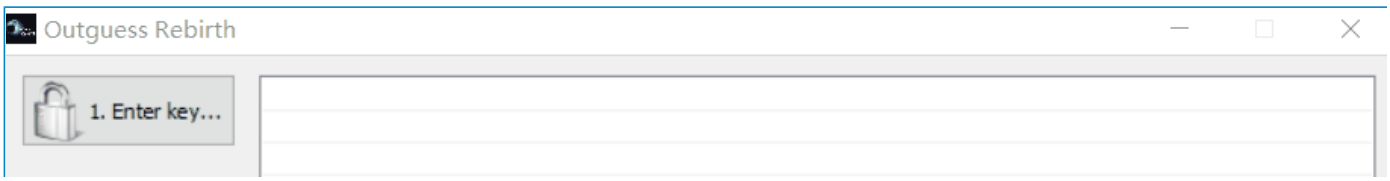
- 在实验机中找到隐写术目录，打开图片隐写，打开图片隐写第四部分文件夹
- 在该文件夹找到Outguess.jpg
- 双击打开图片，我们先确认一下图片内容并没有什么异常
- 使用Stegdetect对图片的隐写方式进行检测
- 从结果中得知是使用JPHide的隐写，使用JPHS工具对隐写信息进行提取
- 最后打开保存的问就，flag是flag{Good_you_got_it}

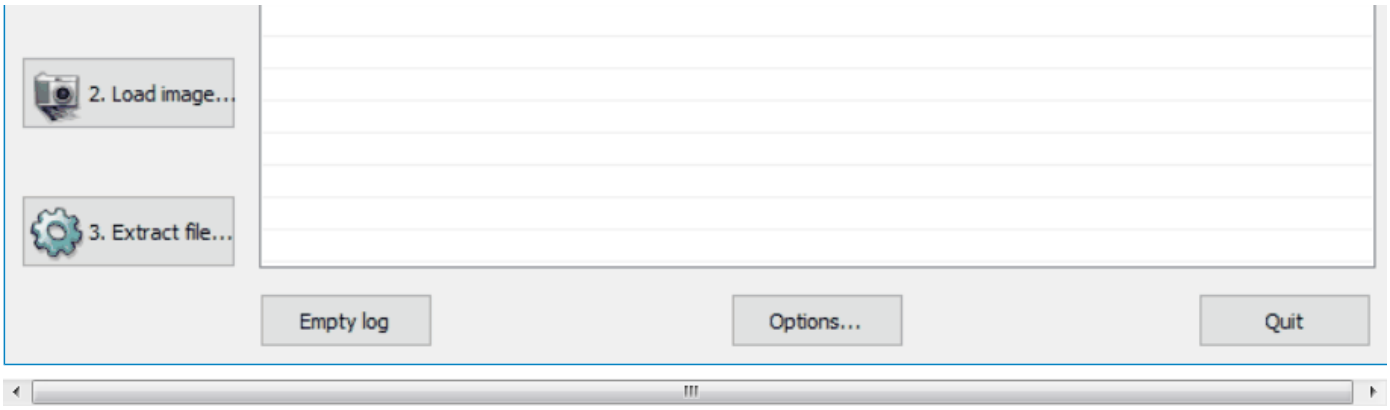
第一步，在cmd中运行，Stegdetect，对目标图片进行检测



从结果可以得知，这里的隐写方式是，Outguess。

第二步，使用Outguess工具提取隐写信息





点击Enter key功能，输入密码，这里我隐写的时候，使用的密码是123456
点击Load image 加载目标图片
点击Extract file功能，将提取出来的信息保存成flag.txt文件。
打开保存后的文件，flag是flag{Outguess}

小结：
从上面的实验来看，JPG图片常用的隐写方式一般也是DCT域的隐写了，不过一般在CTF赛场上，这种题目是可以直接用Stegdetect所检测出来的。

第二部分：数字水印隐写

知识背景：

数字水印
数字水印（digital watermark）技术，是指在数字化的数据内容中嵌入不明显的记号。
特征是，被嵌入的记号通常是不可见或不可察的，但是可以通过计算操作检测或者提取。
盲水印与傅里叶变换

这里介绍的盲水印是以知乎某答主的频域添加盲水印的文章为基础，在2016HCTF的也出了一个隐写题目，也是以频域为背景的。
盲水印，是指人感知不到的水印，包括看不到或听不见（没错，数字盲水印也能够用于音频）。

其主要应用于音像作品、数字图书等，目的是，在不破坏原始作品的情况下，实现版权的防护与追踪。

对图像进行傅里叶变换，起始是一个二维离散傅里叶变换，图像的频率是指图像灰度变换的强烈程度，将二维图像由空间域变为频域后，

图像上的每个点的值都变成了复数，也就是所谓的复频域，通过复数的实部和虚部，可以计算出幅值和相位，计算幅值即对复数取模值，

将取模值后的矩阵显示出来，即为其频谱图。但是问题来了，复数取模后，数字有可能变的很大，远大于255，如果数据超过255，

则在显示图像的时候会都当做255来处理，图像就成了全白色。因此，一般会对模值再取对数，在在0~255的范围内进行归一化，

这样才能够准确的反映到图像上，发现数据之间的差别，区分高频和低频分量，这也是进行傅里叶变换的意义

频域盲水印隐写

在实验找到隐写术目录，打开图片隐写，打开图片隐写第五部分文件夹
在该文件夹找到ori.jpg，res,png
双击打开图片，我们先确认一下图片内容并没有什么异常
运行我提供的脚本，提取水印

> 有必要提以下的是，如果用mathlab生成的盲水印隐写是不需要原图的，这里我能力有限，只能做到需要原图才能提取水印。

```
python decode.py --original <original image file> --image <image file> --result <result file>
```

```
# coding=utf-8
import cv2
import numpy as np
import random
import os
from argparse import ArgumentParser
ALPHA = 5


def build_parser():
    parser = ArgumentParser()
    parser.add_argument('--original', dest='ori', required=True)
    parser.add_argument('--image', dest='img', required=True)
    parser.add_argument('--result', dest='res', required=True)
    parser.add_argument('--alpha', dest='alpha', default=ALPHA)
    return parser


def main():
    parser = build_parser()
    options = parser.parse_args()
    ori = options.ori
    img = options.img
    res = options.res
    alpha = options.alpha
    if not os.path.isfile(ori):
        parser.error("original image %s does not exist." % ori)
    if not os.path.isfile(img):
        parser.error("image %s does not exist." % img)
    decode(ori, img, res, alpha)


def decode(ori_path, img_path, res_path, alpha):
    ori = cv2.imread(ori_path)
    img = cv2.imread(img_path)
    ori_f = np.fft.fft2(ori)
    img_f = np.fft.fft2(img)
    height, width = ori.shape[0], ori.shape[1]
    watermark = (ori_f - img_f) / alpha
    watermark = np.real(watermark)
    res = np.zeros(watermark.shape)
    random.seed(height + width)
```

```
    random.seed(height + width)
    x = range(height / 2)
    y = range(width)
    random.shuffle(x)
    random.shuffle(y)
    for i in range(height / 2):
        for j in range(width):
            res[x[i]][y[j]] = watermark[i][j]
    cv2.imwrite(res_path, res, [int(cv2.IMWRITE_JPEG_QUALITY), 100])


if __name__ == '__main__':
    main()
```

提取盲水印

original 是输入原图， image是之后跟的是加入了水印的图， result是保存水印图片。

```
C:\Users\[redacted]\Desktop\blind-watermark-master>python decode.py --original ori.png --image res.png --result deco.png
```

结果

如果是像HCTF那样的隐写题，只需要有mathlab这个强大的工具，再运用提取盲水印的代码，是不需要原图的,代码如下。

```
imageA = imread('3.bmp','bmp');
fftA = fft2(imageA);
imshow(fftshift(fftA))
imshow(fft(rgb2gray(imread('shimakaze.bmp'))), [1,2]);
```

接下来就是最后一部分了

第三部分：图片容差隐写

知识背景：

容差

容差，在选取颜色时所设置的选取范围，容差越大，选取的范围也越大，其数值是在0-255之间。

容差比较的隐写

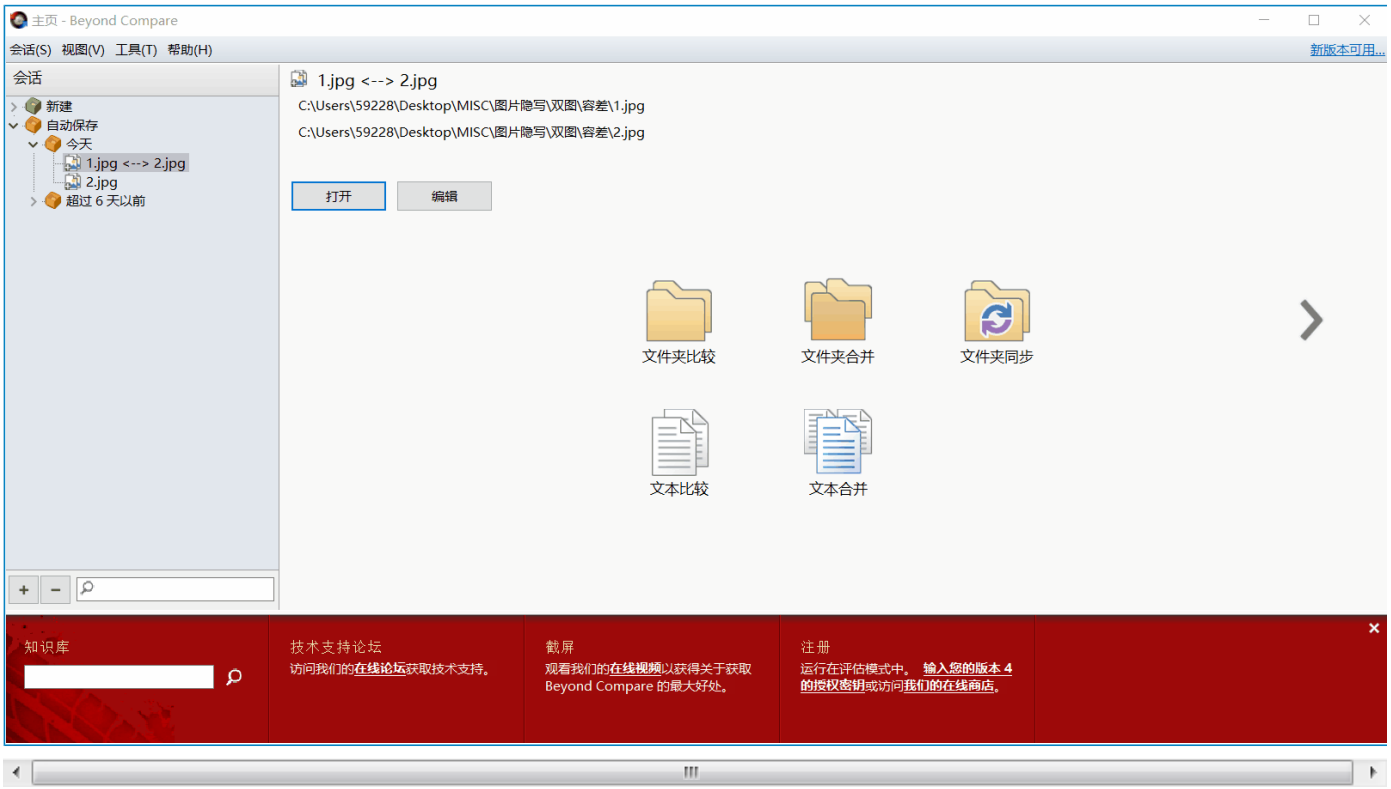
beyond compare

beyond compare是一款很适合用来对图片进行比较的工具，就图片而言，它支持容差、范围、混合等模式。

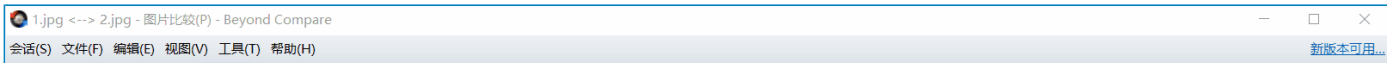
- 实验：

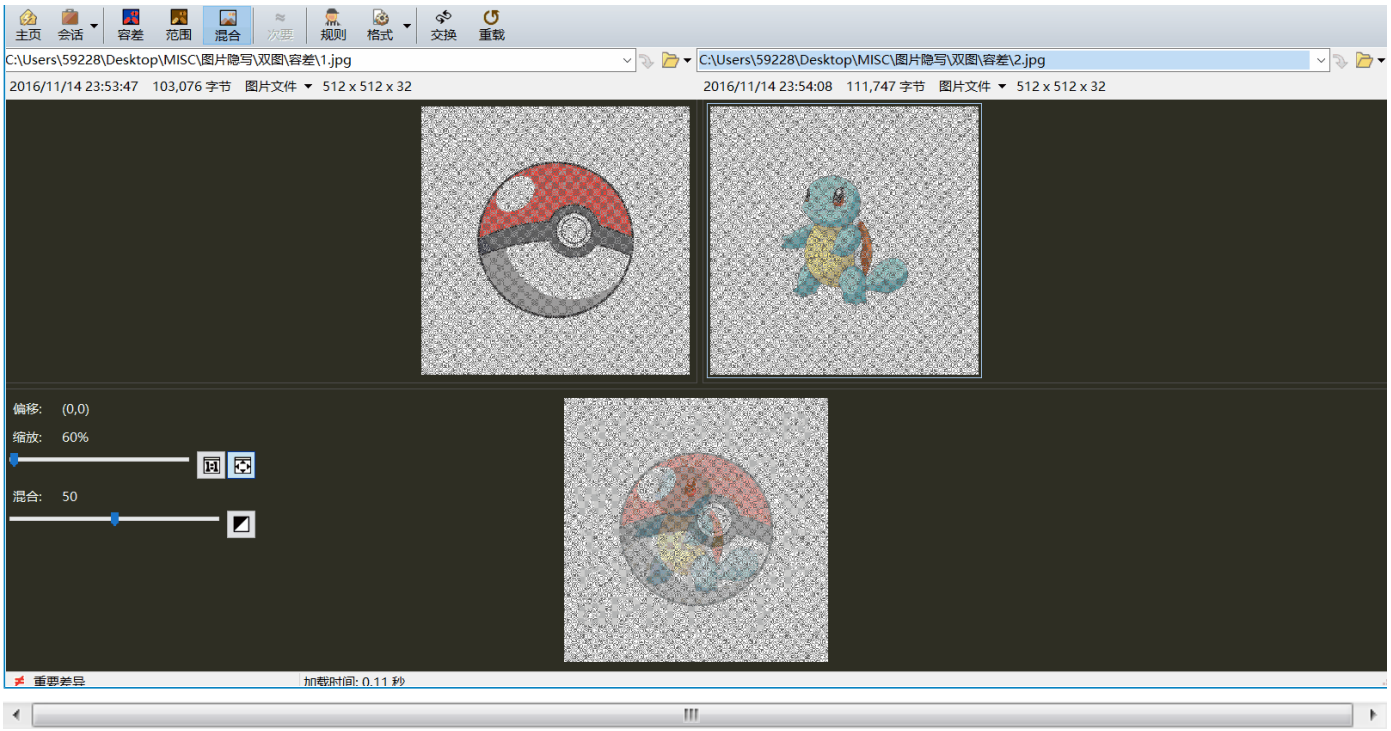
在实验中找到隐写术目录，打开图片隐写，打开图片隐写第六部分文件夹
在该文件夹找到 pokemon_1.jpg，pokemon_2.jpg
打开工具Beyond Compare，选择图片比较，导入两张图片
在左下角慢慢修改容差，
在容差的修改过程中得到了flag

打开工具，选择图片比较，导入pokemon_1.jpg，pokemon_2.jpg



选择容差模式，并调整容差大小





小结：

如果在CTF赛场中，就隐写这一部分，出题人给于两张或者多张图片，一般都是需要对图片的内容进行比较的。

思考：

1. Stegsolve 也有图片的比较的功能，是否能完成这个隐写？如果不可以为什么？

DCT.zip (0.074 MB) 下载附件

您可以考虑给博主来个小小的打赏以资鼓励，您的肯定将是我最大的动力。

作者：
落花四月

出处：
https://www.cnblogs.com/lxz-1263030049/

关于作者：潜心于网络安全学习。如有问题或建议，请多多赐教！

特此声明：所有评论和私信都会在第一时间回复。也欢迎园子的大大们指正错误，共同进步。或者直接私信我

声援博主：如果您觉得文章对您有帮助，可以点击文章右下角【推荐】一下。您的鼓励是作者坚持原创和持续写作的最大动力！