python图片隐写

转载

搬运代码打工人 于 2019-04-18 11:17:01 发布 347 收藏 3

分类专栏： Cryptography

 Cryptography 专栏收录该内容

7 篇文章 0 订阅

订阅专栏

处理前

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from PIL import Image
import argparse

#命令行输入参数处理
parser = argparse.ArgumentParser()
parser.add_argument('file')     #输入文件
#获取参数
args = parser.parse_args()
IMG = args.file



"""
从二进制字符串转为 UTF-8 字符串
"""
def binaryToString(binary):
   index = 0
   string = []
   rec = lambda x, i: x[2:8] + (rec(x[8:], i-1) if i > 1 else '') if x else ''
   # rec = lambda x, i: x and (x[2:8] + (i > 1 and rec(x[8:], i-1) or '')) or ''
   fun = lambda x, i: x[i+1:8] + rec(x[8:], i-1)
   while index + 1 < len(binary):
       chartype = binary[index:].index('0') # 存放字符所占字节数，一个字节的字符会存为 0
       length = chartype*8 if chartype else 8
       string.append(chr(int(fun(binary[index:index+length],chartype),2)))
       index += length
   return ''.join(string)

"""
解码隐藏数据
"""
def decodeImage(image):
   pixels = list(image.getdata())  # 获得像素列表
   binary = ''.join([str(int(r>>1<<1!=r))+str(int(g>>1<<1!=g))+str(int(b>>1<<1!=b))+str(int(t>>1<<1!=t)) for (r,
g,b,t) in pixels]) # 提取图片中所有最低有效位中的数据
   # 找到数据截止处的索引
   locationDoubleNull = binary.find('0000000000000000')
   endIndex = locationDoubleNull+(8-(locationDoubleNull % 8)) if locationDoubleNull%8 != 0 else locationDoubleNu
ll
   data = binaryToString(binary[0:endIndex])
   return data
if __name__ == '__main__':
   print(decodeImage(Image.open(IMG)))
```

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from PIL import Image
import argparse

#命令行输入参数处理
parser = argparse.ArgumentParser()
parser.add_argument('file') #输入文件
parser.add_argument('text') #输入字符串
#获取参数
args = parser.parse_args()
IMG = args.file
TEXT = args.text

"""
取得一个 PIL 图像并且更改所有值为偶数（使最低有效位为 0）
"""
def makeImageEven(image):
    pixels = list(image.getdata())  # 得到一个这样的列表： [(r,g,b,t),(r,g,b,t)...]
    evenPixels = [(r>>1<<1,g>>1<<1,b>>1<<1,t>>1<<1) for [r,g,b,t] in pixels]  # 更改所有值为偶数（魔法般的移位）
    evenImage = Image.new(image.mode, image.size)  # 创建一个相同大小的图片副本
    evenImage.putdata(evenPixels)  # 把上面的像素放入到图片副本
    return evenImage

"""
内置函数 bin() 的替代，返回固定长度的二进制字符串
"""
def constLenBin(int):
    binary = "0"*(8-(len(bin(int))-2))+bin(int).replace('0b','')  # 去掉 bin() 返回的二进制字符串中的 '0b'，并在左边
补足 '0' 直到字符串长度为 8
    return binary

"""
将字符串编码到图片中
"""
def encodeDataInImage(image, data):
    evenImage = makeImageEven(image)  # 获得最低有效位为 0 的图片副本
    binary = ''.join(map(constLenBin,bytearray(data, 'utf-8'))) # 将需要被隐藏的字符串转换成二进制字符串
    if len(binary) > len(image.getdata()) * 4:  # 如果不可能编码全部数据， 抛出异常
        raise Exception("Error: Can't encode more than " + len(evenImage.getdata()) * 4 + " bits in this image.
")
    encodedPixels = [(r+int(binary[index*4+0]),g+int(binary[index*4+1]),b+int(binary[index*4+2]),t+int(binary[in
dex*4+3])) if index*4 < len(binary) else (r,g,b,t) for index,(r,g,b,t) in enumerate(list(evenImage.getdata()))]
# 将 binary 中的二进制字符串信息编码进像素里
    encodedImage = Image.new(evenImage.mode, evenImage.size)  # 创建新图片以存放编码后的像素
    encodedImage.putdata(encodedPixels)  # 添加编码后的数据
    return encodedImage


if __name__ == '__main__':
    encodeDataInImage(Image.open(IMG), TEXT).save('code.png')
```