

实验简介

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，而本套教程内容将

实验内容

本次图片隐写实验包括四大部分

- 一、附加式的图片隐写
- 二、基于文件结构的图片隐写
- 三、基于LSB原理的图片隐写
- 四、基于DCT域的JPG图片隐写
- 五、数字水印的隐写
- 六、图片容差的隐写

第二部分 基于文件结构的图片隐写

实验环境

- 操作机：Windows XP
- 实验工具：
 - 010Editor
 - CRC Calculator

背景知识

首先这里需要明确一下我这里所说的文件结构是什么意思。文件结构特指的是图片文件的文件结构。我们这里主要讲的是PNG图片的文件结构。

PNG，图像文件存储格式，其设计目的是试图替代GIF和TIFF文件格式，同时增加一些GIF文件格式所不具备的特性。是一种位图文件(bitmap file)存储格式，读作“ping”。PNG用来存储灰度图像时，灰度图像的深度可多到16位，存储彩色图像时，彩色图像的深度可多到48位，并且还可存储多到16位的 α 通道数据。

对于一个正常的PNG图片来讲，其文件头总是由固定的字节来表示的，以16进制表示即位 89 50 4E 47 0D 0A 1A 0A，这一部分称作文件头。标准的PNG文件结构应包括：

- PNG文件标志
- PNG数据块

PNG图片是有两种数据块的，一个是叫关键数据块，另一种是辅助数据块。正常的关键数据块，定义了4种标准数据块，个PNG文件都必须包含它们。它们分别是长度，数据块类型码，数据块数据，循环冗余检测即CRC。

我们这里重点先了解一下，png图片文件头数据块以及png图片IDAT块，这次的隐写也是以这两个地方位基础的。

png图片文件头数据块

即IHDR，这是PNG图片的第一个数据块，一张PNG图片仅有一个IHDR数据块，它包含了哪些信息呢？IHDR中，包括了图片的宽，高，图像深度，颜色类型，压缩方法。

如图中蓝色的部分即IHDR数据块。

IDAT 数据块

它存储实际的数据，在数据流中可包含多个连续顺序的图像数据块。这是一个可以存在多个数据块类型的数据块。它的作用就是存储着图像真正的数据。

因为它是可以存在多个的，所以即使我们写入一个多余的IDAT也不会多大影响肉眼对图片的观察

下面进行实验Part 2 基于文件结构的隐写

高度被修改引起的隐写

背景知识中，我们了解到，图片的高度，宽度的值存放于PNG图片的文件头数据块，那么我们就可以通过修改PNG图片的高度值，来对部分信息进行隐藏的。

- 实验：

— [REDACTED]

- ■■■■■■■■ hight.png■

[illegible]

```
- 010Editor PNG
- PNG CRC
- 
```

用010Editor打开图片，运行PNG模板

10 editor呢？因为这个16进制编辑器，有模版功能，当我们运行模版后，可以轻易的找到图片的各个数据块的位置以及内容。

10 editor这个16进制编辑器，有模版功能，当我们运行模版后，可以轻易的找到图片的各个数据块的位置以及内容。

找到PNG图片高度值所对应的位置，并修改为一个较大的值

我们找到IHDR数据块，并翻到struct IHDR Ihdr位置，修改height的值到一个较大的值，如从700修改到800。

使用CRC Calculator重新计算CRC校验值

输入参数，然后点击Calculator计算，得到CRC值

为什么要重新计算CRC校验值呢？防止图片被我们修改后，自身的CRC校验报错，导致图片不能正常打开。

修改相应的CRC校验值，为我们重新计算后数值

思考

这个实验，我们进行了PNG图片高度修改以及CRC校验值的重计算，那么请大家以下问题

1. JPG图片是否也有这样的隐写形式呢？
2. 了解JPG以及GIF等图片文件的格式。

隐写信息以IDAT块加入图片

在背景知识中，我们提到了一个重要的概念就是图片的IDAT块是可以存在多个的，这导致了我们可以将隐写西信息以IDAT块的形似加入图片。

- 实验：

```
- 
- hidden.png
- 
- pngcheck
- pngcheck
- 
```

前景知识

pngcheck可以验证PNG图片的完整性（通过检查内部CRC-32校验和&bra;比特&ket;）和解压缩图像数据；它能够转储几乎所有任选的块级别信息在该图像中的可读数据。

我们使用pngcheck -v hidden.png 如此的命令对图片进行检测

使用pngcheck对图片进行检测

我们使用命令：

```
pngcheck -v hidden.png
```

对图片的文件结构进行检测。

发现异常，并判断异常的原因

我们会发现，图片的的数据块形式是如下的

Type: IHDR

Size: 13

CRC : 5412913F

Pos : 33

Type: IDAT

Size: 10980

CRC : 98F96EEB

Pos : 11025

Type: IEND

Size: 0

CRC : AE426082

我们会惊讶的发现pos为11025的size居然为0，这是一块有问题的地方，我们可以怀疑，这一块是隐写的信息。

编写脚本并提取内容

```

#!/usr/bin/python

from struct import unpack
from binascii import hexlify, unhexlify
import sys, zlib

# Returns [Position, Chunk Size, Chunk Type, Chunk Data, Chunk CRC]
def getChunk(buf, pos):
    a = []
    a.append(pos)
    size = unpack('!I', buf[pos:pos+4])[0]
    # Chunk Size
    a.append(buf[pos:pos+4])
    # Chunk Type
    a.append(buf[pos+4:pos+8])
    # Chunk Data
    a.append(buf[pos+8:pos+8+size])
    # Chunk CRC
    a.append(buf[pos+8+size:pos+12+size])
    return a

def printChunk(buf, pos):
    print 'Pos : '+str(pos)+' '
    print 'Type: ' + str(buf[pos+4:pos+8])
    size = unpack('!I', buf[pos:pos+4])[0]
    print 'Size: ' + str(size)
    #print 'Cont: ' + str(hexlify(buf[pos+8:pos+8+size]))
    print 'CRC : ' + str(hexlify(buf[pos+size+8:pos+size+12]).upper())
    print

if len(sys.argv)!=2:
    print 'Usage: ./this Stegano_PNG'
    sys.exit(2)

buf = open(sys.argv[1]).read()
pos=0

print "PNG Signature: " + str(unpack('cccccccc', buf[pos:pos+8]))
pos+=8

chunks = []
for i in range(3):
    chunks.append(getChunk(buf, pos))
    printChunk(buf, pos)
    pos+=unpack('!I', chunks[i][1])[0]+12

decompressed = zlib.decompress(chunks[1][3])
# Decompressed data length = height x (width * 3 + 1)
print "Data length in PNG file : ", len(chunks[1][3])
print "Decompressed data length: ", len(decompressed)

height = unpack('!I', (chunks[0][3][4:8]))[0]
width = unpack('!I', (chunks[0][3][:4]))[0]
blocksize = width * 3 + 1
filterbits = ''
for i in range(0, len(decompressed), blocksize):
    bit = unpack('2401c', decompressed[i:i+blocksize])[0]
    if bit == '\x00': filterbits+='0'
    elif bit == '\x01': filterbits+='1'
    else:
        print 'Bit is not 0 or 1... Default is 0 - MAGIC!'
        sys.exit(3)

s = filterbits
endianess_filterbits = [filterbits[i:i+8][::-1] for i in xrange(0, len(filterbits), 8)]

flag = ''
for x in endianess_filterbits:

```

```
if x=='00000000': break
flag += unhexlify('%x' % int('0b'+str(x), 2))

print 'Flag: ' + flag
```

脚本如上，flag DrgnS{WhenYouGazeIntoThePNGThePNGAlsoGazezIntoYou}.

思考

1. 我们是否可以将一张二维码以IDAT块的形式写入图片呢？
2. 试着将信息以IDAT块的形式写入图片

保存后，重新打开图片，我们就能看到被隐藏的内容

归档.zip (0.414 MB) [下载附件](#)

[点击收藏](#) | 1 关注 | 2

[上一篇：Misc 总结 ----隐写术之图...](#) [下一篇：CISSP考试一次通过指南（文末附福利）](#)

1. 18 条回复



[176****6583](#) 2017-12-24 12:10:00

学习了学习了

0 回复Ta



[152****5136](#) 2017-12-25 20:23:40

路过

0 回复Ta



[wahaha_a](#) 2017-12-26 14:58:17

学习学习！

0 回复Ta



[threst](#) 2018-01-18 13:18:41

学习了感谢分享

0 回复Ta



[老雄](#) 2018-01-23 15:28:27

支持

0 回复Ta



[bendawang](#) 2018-01-29 16:37:57

路过

0 回复Ta



[1815837370479554](#) 2018-05-29 13:56:33

路过 路过

0 回复Ta



[wuq****@126.com](#) 2018-07-21 08:27:52

好用

0 回复Ta



[Str3am](#) 2018-08-20 14:12:30

有一个问题，隐写信息以IDAT块加入图片那最后一个块为IEND，它的size为0是正常的啊

0 回复Ta



[189****5586](#) 2018-09-14 09:52:43

不做伸手党，回复看附件

0 回复Ta



[189****5586](#) 2018-09-14 10:05:11

IEND数据段的size标志就是0，根据这个不能判断IDAT中存在隐写数据

0 回复Ta



[Catcher](#) 2018-11-26 15:06:12

学习学习~

0 回复Ta



[相视一笑似路人](#) 2018-12-26 15:08:29

pngcheck是如何判断出最后一段是有问题的？
控制台已经打印了No error

0 回复Ta



[我先让你三掌](#) 2019-03-13 17:41:44

学习学习！

0 回复Ta



[紧茶来了快跑](#) 2019-03-28 23:21:25

学习学习~

0 回复Ta



[川上白111](#) 2019-09-26 10:39:31

学习学习，入门一下隐写。

0 回复Ta



[川上白111](#) 2019-09-26 13:00:11

模板功能的那个图挂了。

0 回复Ta



[M1n3](#) 2019-11-05 21:17:21

[@川上日111](#) 先知的图挂了 - - 我也没办法救了

0 回复Ta

[登录](#) 后跟帖

[先知社区](#)

[现在登录](#)

[热门节点](#)

[技术文章](#)

[社区小黑板](#)

[目录](#)

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)