隐写术之图片隐写

原创

陈京九 于 2019-04-18 10:31:26 发布 2020 收藏 26

本文链接：https://blog.csdn.net/weixin_42317232/article/details/103080976

> 欢迎大家光临我的个人博客，大家一起交流学习
> http://www.chenjingjiu.cn
> 本文大部分源于 先知社区 中M1n3所作 Misc 总结 ----隐写术之图片隐写一文，对其进行了相应的精简并增加了自己的思考以作为自己的学习笔记，如有侵删。

0x00 背景介绍

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。而密码编码是关于信息加密，即设想到信息可能会被接受者之外的第三方获取而采取的一种措施，通过通信双方预先设定的规则对信息进行加密，使第三方即使获取到信息也无法理解其含义。所以隐写术重点在于信息的隐藏，密码编码重点在于信息的加密，这两者属于完全不同的概念。

0x01 图片隐写术的分类

0x02 附加式的图片隐写

在附加式的图片隐写术中，通常用某种程序或者某种方法在载体文件中直接附加上需要被隐写的目标，然后将载体文件直接传输给接受者或者发布到网站上，然后接受者者根据方法提取出被隐写的消息，这一个过程就是附加式图片隐写。

在CTF赛事中，关于这种图片隐写的大概有两种经典方式，一是直接附加字符串，二是图种的形式出现。

A.附加字符串
这种方式是利用工具将隐藏信息直接写入到图片结束符之后，由于计算机中图片处理程序识别到图片结束符就不再继续向下识别，因此后面的信息就被隐藏起来。这种方式可以利用winhex，ghex等工具进行打开，或者notepad打开也可看到最后的附加的字符，所以虽然简单，但是隐藏效果不是很好。

windows下制作这种图片的方式也有很多，比如刚才说到的winhex直接在文件尾写入字节，或利用copy /b a.jpg+b.txt c.jpg来进行制作。其中a.jpg是一张普通图片，即将作为信息的载体，b.txt中是隐藏的信息，c.jpg是附加了隐藏信息的图片，发送时就是发送c.jpg。

B.图种
图种是一种采用特殊方式将图片文件（如jpg格式）与rar文件结合起来的文件。该文件一般保存为jpg格式，可以正常显示图片，当有人获取该图片后，可以修改文件的后缀名，将图片改为rar压缩文件，便可得到其中的数据。刚才我们说过，因为计算机中图片处理程序识别图片的过程是，从图片头开始，以图片头声明的格式所定义的编码格式对数据流进行读取，一直到图片的结束符，当图片处理程序识别到图片的结束符后，不再继续向下识别，所以我们在通常情况下只能看到它是只是一张图片。

但使用binwalk程序可以轻松对其进行识别并还原，binwalk程序在上一篇博客中有讲解。也可以利用winhex寻找到图片的结束符，然后在其后查找是否有zip或rar等文件的文件头。最后也可利用转换文件后缀对其进行读取并解压，具体过程在这篇博客中有用到。

0x03 基于文件结构的图片隐写

这里的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，其16进制表示为 89 50 4E 47 0D 0A 1A 0A，这一部分称作PNG文件头。
标准的PNG文件结构应包括：
1.PNG文件标识

2.PNG数据块PNG图片有两种数据块的，一种是关键数据块，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，分别是长度，数据块类型码，数据块数据，循环冗余检测即CRC，每个PNG文件都必须包含它们。
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为基础的。

PNG图片文件头数据块
即IHDR(Image HeaDeR)，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，它包含了图片的宽，高，图像深度，颜色类型，压缩方法等等信息。(即定义图片数据流的读取规则)

PNG图片IDAT(Image DATa)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也不会明显影响肉眼对图片的观察。
高度被修改引起的隐写

刚才我们了解到，IHDR中定义了图片的高度和宽度，可以通过修改高度值或宽度值对部分信息进行隐藏。

如果图片原本是800(宽)*600(高)，然后图片的高度从600变成500，这样下面800×100区域的信息就无法从图片中显示出来，我们可见的只有上方800*500的区域，这样就达成了图片隐写的目的。同理可知图片的宽度也可以进行类似的修改以达到隐藏信息的目的。

为了还原图片，可以利用winhex或者010Editor等编辑器打开图片。但我们推荐后者，因为它提供了不同文件的模板，通过加载png模板，我们可以直观的知道哪里是PNG的长度字段或宽度字段，它提供了hex字符串到字段名的映射，更便于我们进行修改。在修改文件后，需要利用CRC Calculator对CRC校验码进行重新计算赋值，以防图片被修改后，自身的CRC校验报错，导致图片不能正常打开。

图片中加入IDAT块以实现隐写

刚才我们提到过一个图片的IDAT块是可以存在多个的，这也导致我们可以利用添加IDAT块的方式来实现信息的隐写。

利用PNGcheck软件可以验证PNG文件的完整性，利用pngcheck -v a.jpg可以对图片的文件结构进行检测。

文件结构中可能会存在size=0的IDAT块，这说明相应的块是无法用肉眼看到的，也即隐藏的内容。可以通过脚本对隐藏内容进行提取。现在我还没有刷到类似的题，刷到了会补链接和代码。

0x04 基于LSB原理的图片隐写

LSB，最低有效位，英文是Least Significant Bit 。我们知道图像像素一般是由RGB三原色（即红绿蓝）组成的，每一种颜色占用8位，0x00~0xFF，即一共有256种颜色，一共包含了256的3次方的颜色，颜色太多，而人的肉眼能区分的只有其中一小部分，这导致了当我们修改RGB颜色分量中最低的二进制位的时候，我们的肉眼是区分不出来的。

简单的LSB隐写
这种隐写仅对于某一通道值进行改写。将要隐写的信息图片直接覆盖该通道的相应值，即可实现信息的隐写。这种方式利用Stegsolve软件变换图层即可实现还原，相比而言隐秘性较低。

有一点难度的LSB隐写
最简单的隐写我们只需要通过工具Stegsolve切换到不同通道，可以直接看到隐写内容了，那么更复杂一点就不是这么直接了，而是只能通过工具来查看LSB的隐写痕迹，再通过工具或者脚本的方式提取隐写信息。

今年阿里春招的第一题就是类似的题目，问的是图片和视频的隐写方式。当时还没有接触过LSB的我写出了如下的回答：
操作图片中的某些像素点，通过调整该像素点的RGB值，或其中的某一个值，达到隐藏信息的作用。可以将要嵌入的信息进行编码，变换为01字串，然后顺序加到每一个像素点的RGB值上去。因为RGB阈值是0-255，若是当前像素点的RGB中某一个值已经是255，则将该值减一，保证不会超出其阈值，又不容易被发现。
传输时同时传送两张图片，一张是原来的图片，一张是嵌入信息后的图片。两方商量好加解密的规则，传送后，对两张图片中每一个像素的RGB信息作差取绝对值，还原01字串。对于RGB值的轻微改动一般无法用肉眼发现，可以保障该方法的隐秘性。(当时直接保存下来的，没有进行任何的修改)

现在看来这种方式有点像一种LSB，也有点像后面要提到的容差隐写，只不过当时不知道其名词而已。
这种方式的解密，可以利用工具将每个LSB的值导出为一个流，然后转换为字符串进行读取。这种方式的隐蔽性相对来讲较高，一般难以发现。

0x05 基于DCT域的JPG图片隐写

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。(有点CNN的影子)
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：

1.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

2.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。
对于这种隐写方法的还原，首先利用Stegdetect对图片进行检测，判断该图片利用的是何种隐写方式。然后针对不同的隐写方式，选择对应的工具进行信息的还原。比如利用JPHS还原jphide隐写，利用Outguess还原outguess隐写等等。

0x06 数字水印隐写

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

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。
具体盲水印的提取等遇到了再回来补。

0x07 容差比较的隐写

容差，指的是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。

在隐写术方面，可以根据容差进行信息的隐藏。若是有两张图片，则对两张图片的每一个像素点进行对比，设置一个容差的阈值α，超出这个阈值的像素点RGB值设置为(255,255,255),若是没超过阈值，则设置该像素点的RGB值为(0,0,0)。因此，通过调整不同的α值，可以使对比生成的图片呈现不同的画面。比如两张图完全一样，设置阈值α为任何值，最后得到的对比图都只会是全黑。若两张图每一个像素点都不同，阈值α设置为1，则对比图将是全白。如果将隐藏信息附加到某些像素点上，这时调整阈值α即可看到隐藏信息。

如果是一张图片，则根据每一像素点周围像素的值进行判断，同样设置一个阈值，若当前像素点超过周围像素点的均值，或者其它的某种规则，则将该像素点RGB值置为(255,255,255)，反之则不进行处理，或者设置为全0.这样也可以获得隐藏的信息。

其实这种隐写方式也有点像我在阿里笔试中提到的隐写方式，那就是需要两张图片的对比获取其差值，然后再通过差值得到隐藏信息。只不过这个是通过图片的形式获得了信息，而我所想的是通过对比获得01bit流，然后bit流根据双方协定的规则进行还原再得到隐藏信息。我所构想的这种方式可以实现多种隐写术的嵌套应用，比如01字节流还原后是一个jpg格式的文件，这个文件就又可以进行图片隐写术。也可以通过编码规则的不同实现最底层的数据加密，当然这些在实际中也不一定有多少应用场景吧，有这个功夫直接加密就好了，何必用这么复杂的隐写术呢。

0x08 总结

本文总结了目前较为流行的几种图片隐写方法，对它们进行了分类并阐述了相关原理。具体实例我遇到过的都给了博客链接，没有遇到的以后会再补。通过这一篇博文的撰写得到了很多收获，印证了我想到的隐写术是确实存在的，这对于我的信心也是一个鼓舞。未来还是要多看，多学，才能在实践中更加游刃有余。

路漫漫其修远兮，吾将上下而求索