Lsb图片隐写

转载

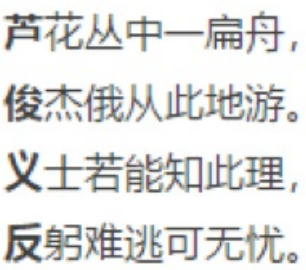
weixin_34075268 于 2018-08-31 18:06:57 发布 7045 收藏 53
文章标签： matlab 人工智能 photoshop
原文链接：https://segmentfault.com/a/1190000016223897

前言

在刚刚过去的网鼎杯第一场比赛中，做到了一道杂项题是关于lsb隐写的。LSB全称为 least significant bit，是最低有效位的意思。Lsb图片隐写是基于lsb算法的一种图片隐写术，以下统称为lsb隐写，这是一种常见的信息隐藏方法。当然关于图像的隐写的方法有很多，统称为隐写术，以后会写一篇总结这类隐写的文章。这里只把lsb隐写单独拿出来分析，因为lsb隐写很实用，算法简单，能存储的信息量也大，更何况是CTF比赛中的常客。还有一个原因是最近本人做的不少杂项题的坑都踩在了lsb隐写上（是我太菜了，大神莫笑。），所以发誓一定要把这类题搞清楚。

隐写术简介

先简单的讲讲什么是隐写。由于我们识别声音或图片的能力有限，因此稍微改动信息的某一位是不会影响我们识别声音或图片的。隐写和加密之间的相同点就是，都是需要经过特殊的处理才能获得特定的信息。它们之间的不同点简单的说就是，加密的话会是一些奇怪的字符，或数据。隐写的话，就是信息明明就在眼前，但是你却视而不见。古人的藏头诗也是隐写的一种啦。



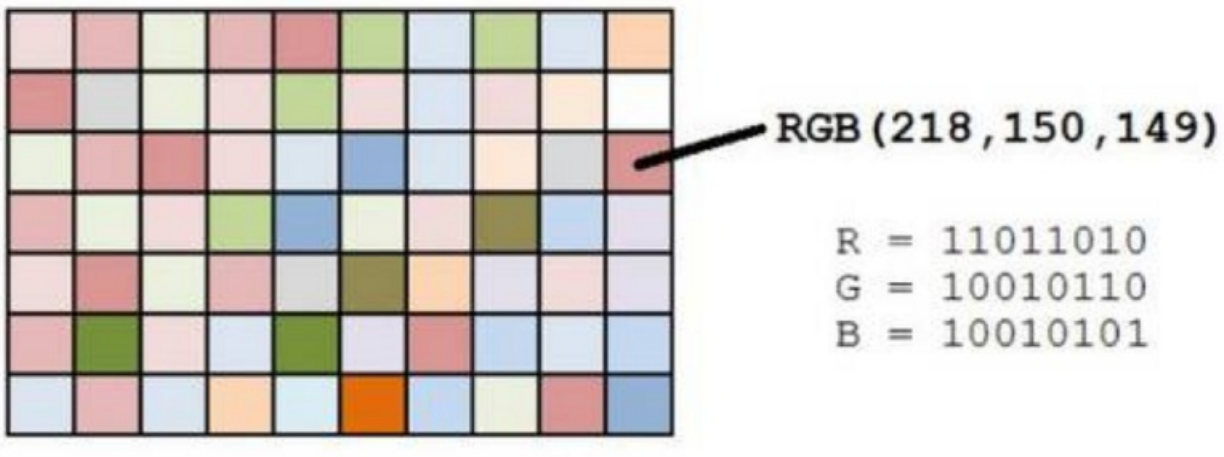
Lsb隐写

其实吧一开始我是想把一大堆算法理论和公式摆上来讲讲lsb隐写是什么，但是想想，这种做法是真的可恶啊，摆明不想让人看嘛。所以我
先来看下面一张图，这道题来自实验吧（原题链接：http://www.shiyanbar.com/ctf/1897），题目名字叫--最低位的轻吻。

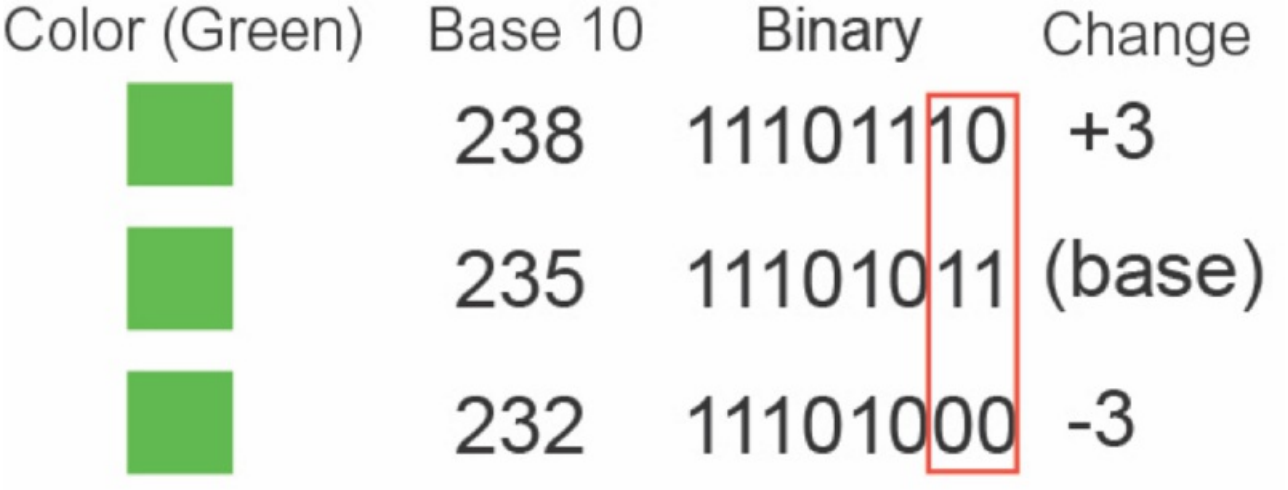
打开看是一张很著名的图片-----胜利之吻。图片是bmp格式。看到题目中有最低位几个字，如果是对隐写类的题目熟悉的话就能一下子想到是lsb隐写。因为lsb隐写就是利用的图像中的最低有效位。最低有效位这个词在前文中出现很多次了，但是到底是什么意思呢？

首先来讲png图片，png图片是一种无损压缩的位图片形格式，也只有在无损压缩或者无压缩的图片（BMP）上实现lsb隐写。如果图像是jpg图片的话，就没法使用lsb隐写了，原因是jpg图片对像数进行了有损压缩，我们修改的信息就可能会在压缩的过程中被破坏。而png图片虽然也有压缩，但却是无损压缩，这样我们修改的信息也就能得到正确的表达，不至于丢失。BMP的图片也是一样的，是没有经过压缩的。BMP图片一般是特别的大的，因为BMP把所有的像数都按原样储存，没有进行压缩。

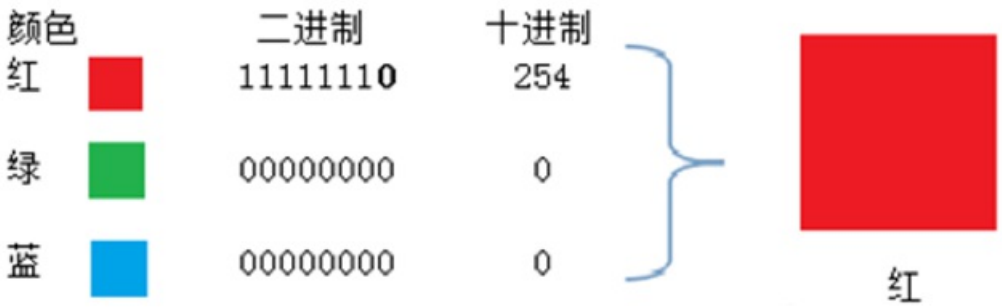
png图片中的图像像数一般是由RGB三原色（红绿蓝）组成，每一种颜色占用8位，取值范围为0x00~0xFF，即有256种颜色，一共包含了256的3次方的颜色，即16777216种颜色。而人类的眼睛可以区分约1000万种不同的颜色，这就意味着人类的眼睛无法区分余下的颜色大约有6777216种。

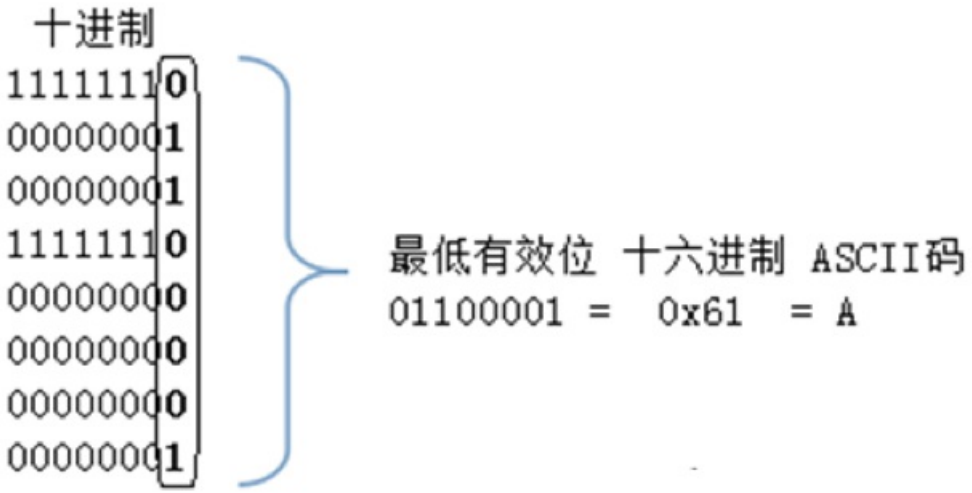


LSB隐写就是修改RGB颜色分量的最低二进制位也就是最低有效位（LSB），而人类的眼睛不会注意到这前后的变化，每个像数可以携带3比特的信息。



上图我们可以看到，十进制的235表示的是绿色，我们修改了在二进制中的最低位，但是颜色看起来依旧没有变化。我们就可以修改最低位中的信息，实现信息的隐写。我修改最低有效位的信息的算法就叫做lsb加密算法，提取最低有效位信息的算法叫做lsb解密算法。
再放两张图加深下理解：





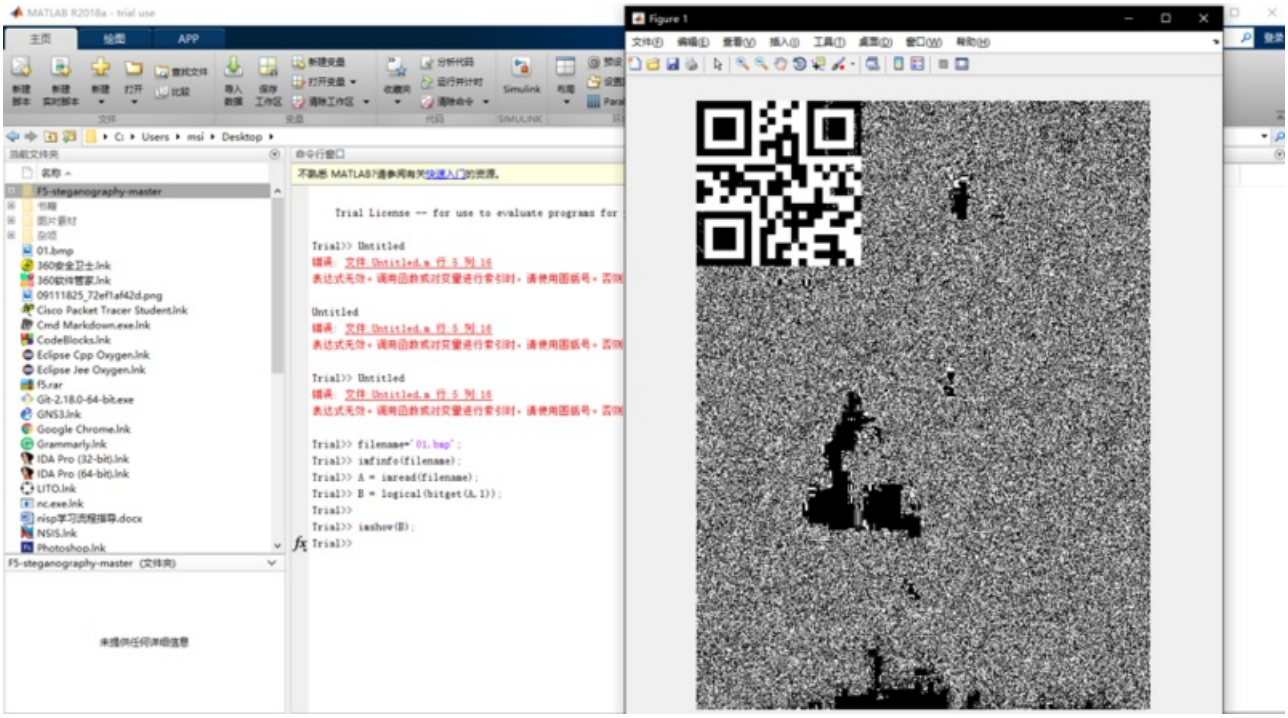
回到题目上来，这里我们使用一款功能很强大的lsb隐写分析工具---StegSolve图片通道查看器（下载地址：http://www.caesum.com/handboo...）。
使用过photoshop的朋友应该对图片通道有些概念，一幅完整的图像，红色绿色蓝色三个通道缺一不可。一幅图像，如果关闭了红色通道，那么图像就偏青色。如果关闭了绿色通道，那么图像就偏洋红色。如果关闭了蓝色通道，那么图像就偏黄色。当然还有个Alpha通道，是一个8位的灰度通道，也可以理解为透明度（粗糙的理解）。关于图像通道详细的讲解可以自行百度，这里不再详细说明。
使用stegsolve打开图片，按右方向键查看各通道显示的图像。一般有些题目会在某一个图像通道中直接显示出flag，但是显然这题不行，看来还需要绕些弯，要获取最低位的图片信息。
所以这道题的思路就是将图片转换成0,1 像素点（图像处理问题），这里可以直接使用MATLAB（MATLAB特别适合图像处理，而且语法特别特别简单）：
在命令行窗口输入以下命令：

```
filename='01.bmp';
imfinfo(filename);
A = imread(filename);
B = logical(bitget(A,1));
imshow(B);
```

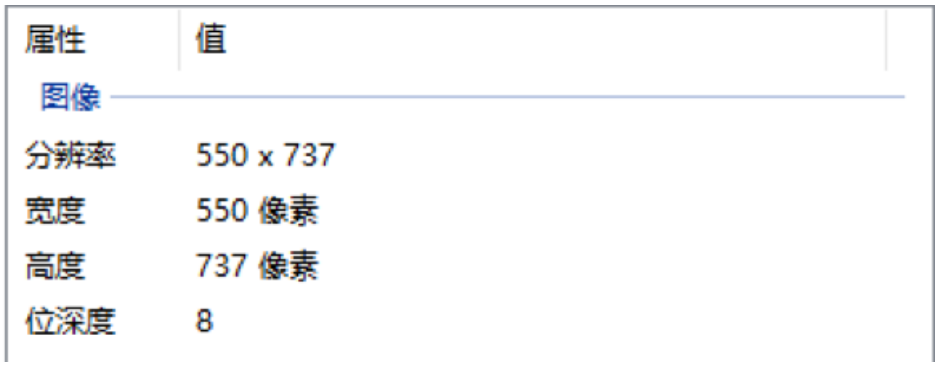
指令详解：

```
>> filename='01.bmp';          %输入要打开的文件名
>> imfinfo(filename);          %调用imfinfo函数获取图片文件信息
>> A = imread(filename);       %调用imread函数读入图片文件信息
>> B=logical(bitget(A,1));     %首先调用bitget函数：获取A信息所对应的二进制中
>>                             %从右往左第一位的值，即最低有效位LSB；再调用
>>                             %logical函数：把数值变成逻辑值
>> imshow(B);                  %输出由逻辑值0（黑），1（白）组成的像素点
```

运行得结果如下。扫描这个二维码，就能直接得到flag，有兴趣的朋友可以自己动手扫一下。



这里还可以使用其他的编程算法来解，原理都是一样的，但如果不会matlab语法，该怎么办呢。其实这题还有一种解法，因为只是简单的获取最后一位然后画图，但是为啥stegsolve获取不到呢。
我们先来看下图像信息：

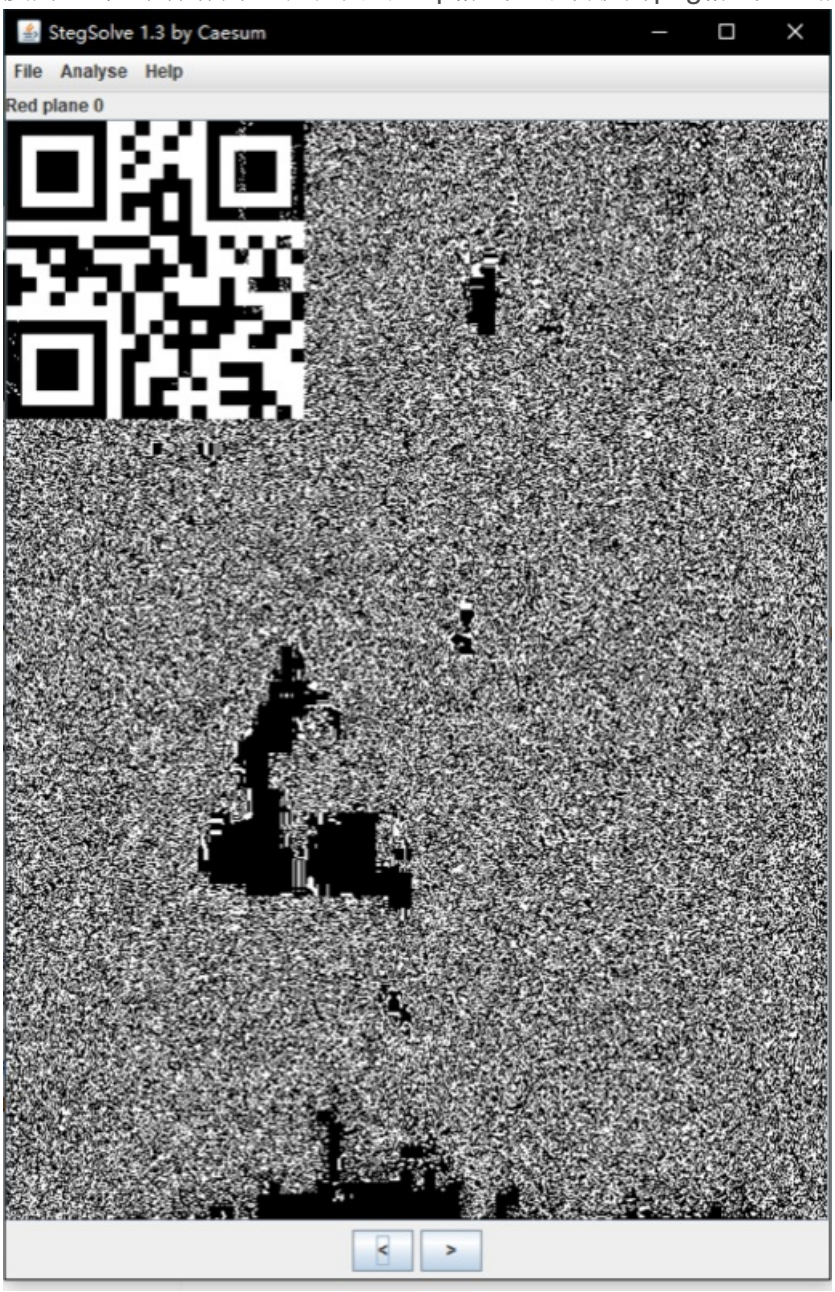


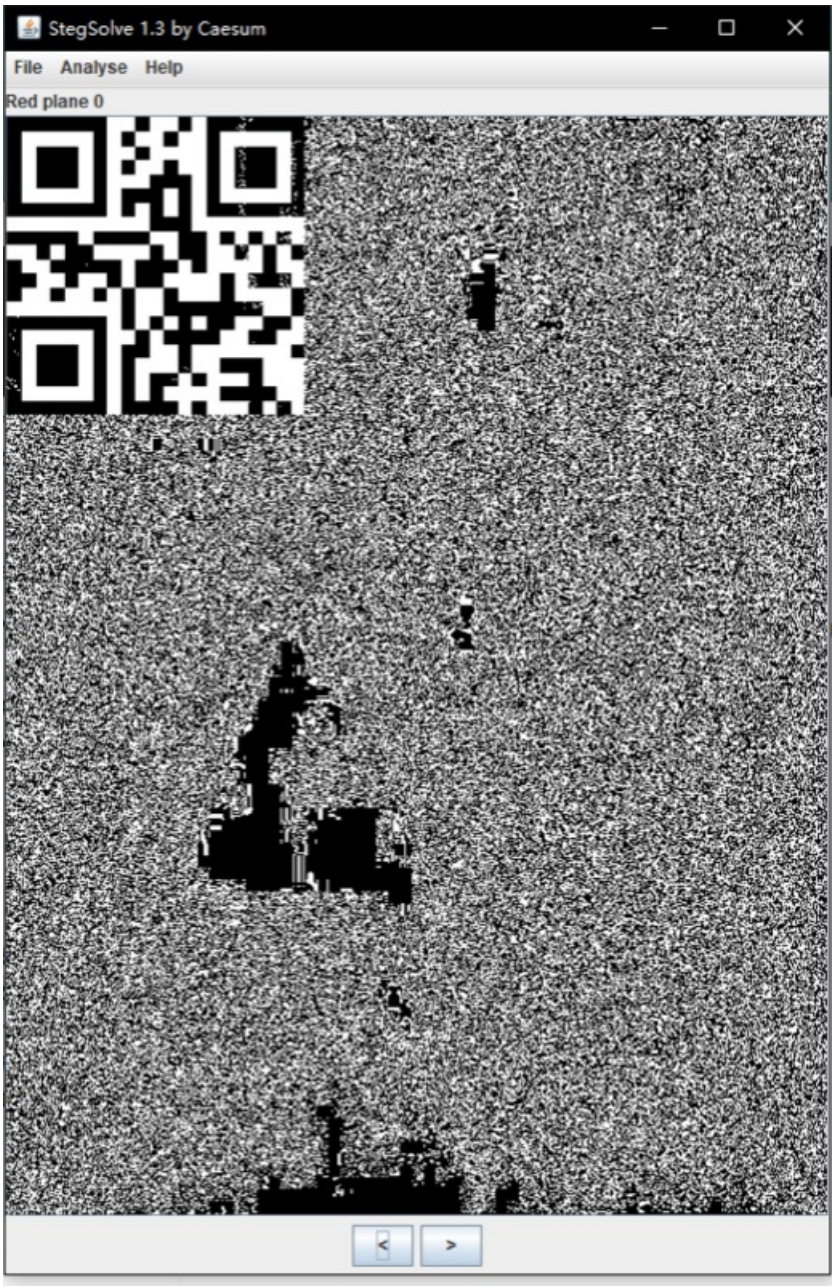
发现是bmp的8位灰度图。猜测是StegSolve解析8位的BMP存在问题？

常见的8位通道RGB图像，3个通道共24位，即一张24位RGB图像里可表现大约1670万种颜色；而16位通道RGB图像，3个通道共48位，2的48次方是多少种颜色。32位深度CMYK（8位×4通道）。
这里的8位、16位、32位指颜色深度（Color Depth）用来度量图像中有多少颜色信息可用于显示或打印像素，其单位是“位（Bit）”，所以颜色深度有时也称为位深度。
常用的颜色深度是1位、8位、24位和32位。1位有两个可能的数值：0或1。较大的颜色深度（每像素信息的位数更多）意味着数字图像具有较多的可用颜色和较精确的颜色表示。

试试转换成png格式呢再看看呢？用画图另存为png格式（不能直接改后缀）。此时发现文件变大，图像信息如下图：

用StegSolve打开后，在RGB的最后一位看到二维码。（原图在保存的时候显示的类型是256色位图，就是位深度为8，如果保存为24位位图的bmp格式，不转换为png格式，也能用StegSolve找到如上图的二维码。）



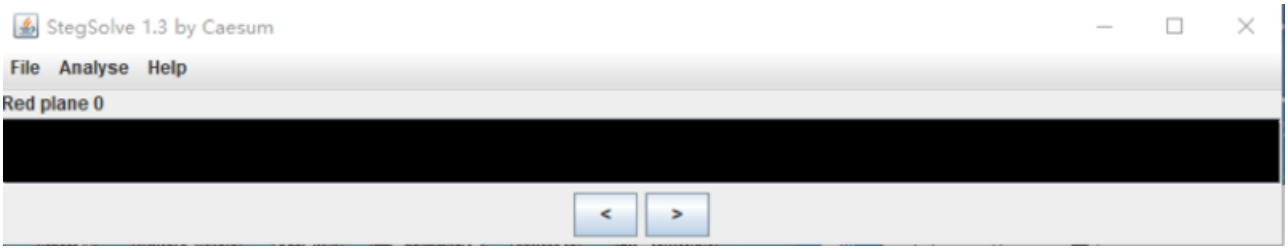


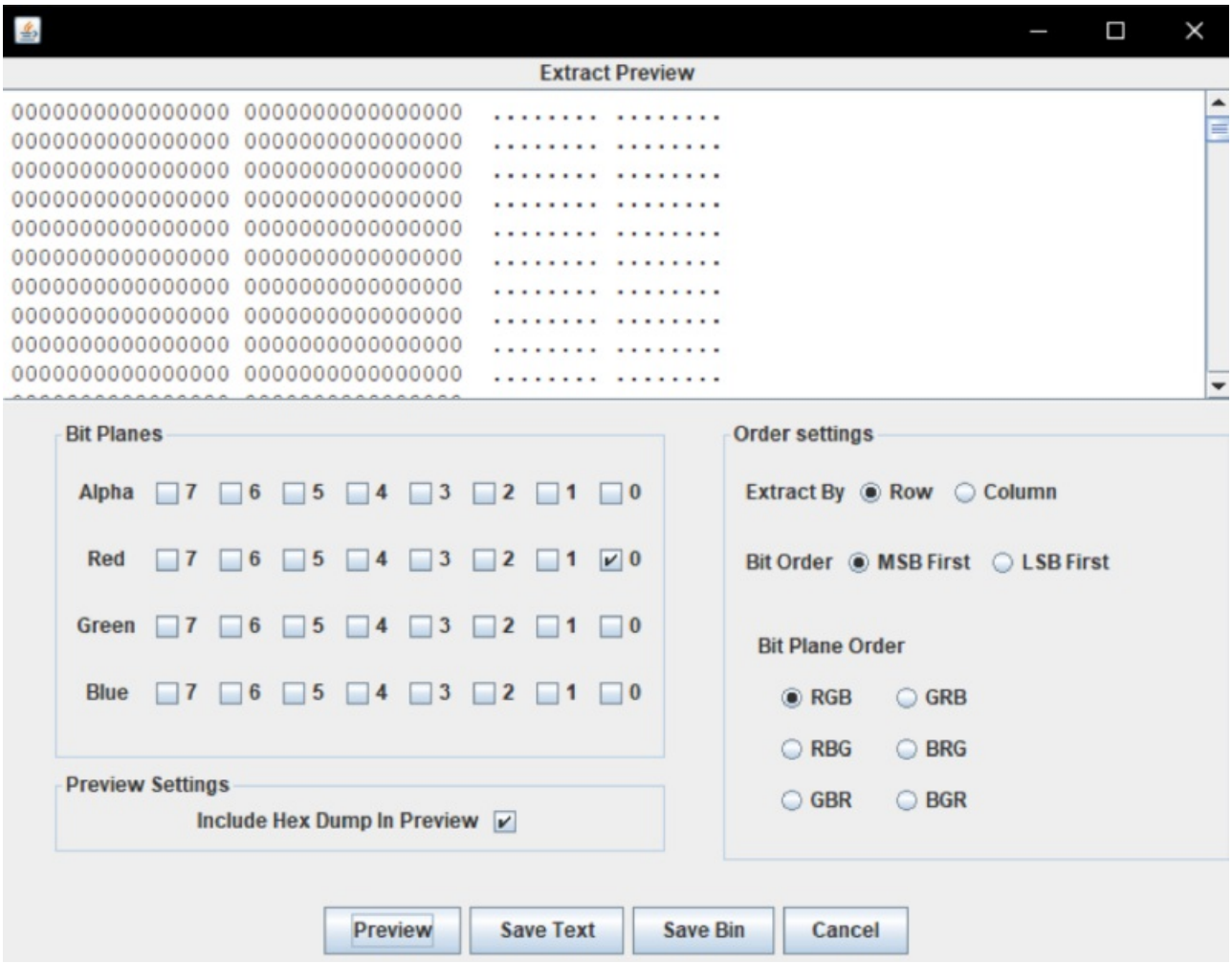
所以我们改变图片位深度，就能得到其中的二维码信息。

Lsb隐写的变形题

经过上面的例子，我们基本上了解了什么是lsb隐写。一般的lsb隐写我们都能使用工具或者编写程序提取到图片的最低有效位信息，从而得到其中的内容。但是出题人的脑洞不局限于此，lsb隐写还有各种扩展的使用方法。下面是网鼎杯第一场中的一道杂项题，也是一道lsb隐写题。不同的是，光提取最低有效位是不能进行解答的。
首先给我们的是一张花花的图是png格式。题目名字是minify，使变小的意思，那应该就是指lsb隐写。

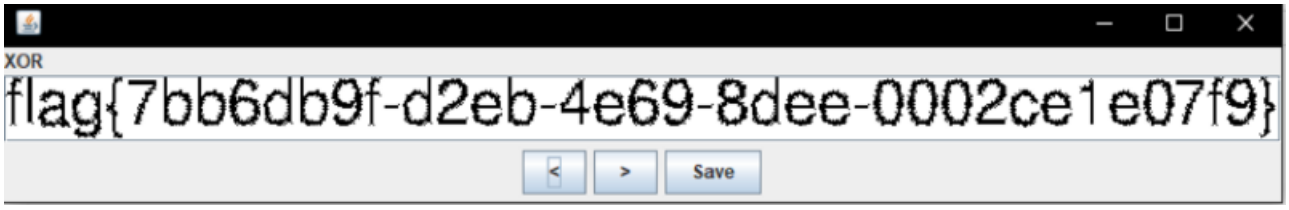
光凭肉眼是真的什么也看不出，我们借助于工具StegSolve进行分析。
一路翻下去，除了花花的还是花花的，唯一值得怀疑的地方就在red plane 0 这里是纯黑，



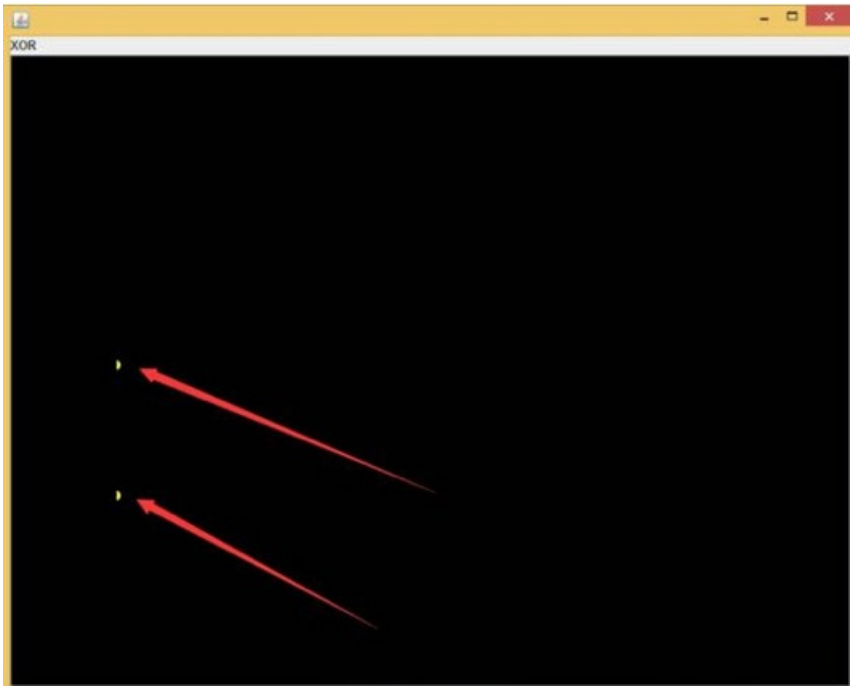


说明这里什么也没有，正常的图片都不会是这样子的，其它通道也都显示正常。所以这个异常给我们什么启示呢？从这里可以真正确定是lsb隐写了。

那我们要从其他的通道比如：blue、alpha、green中找些到信息。整张图片看起来是毫无规则的像素点，那一定想把真正的信息隐藏起来，再用一些毫无规则的像素点干扰我们。我们如果想得到其中的信息，就要去掉这些干扰点。但是到底去掉哪些呢。经过前面的步骤我们知道了信息可能隐藏在plane 0中，所以我们要先把各个通道的plane 0提取出来。Red plane 0因为是空信息，可以不用提取了。我们提取出（File->Save as）Green plane 0、Alpha plane 0、Blue plane 0，把他们各另存为一张图。然后各个图进行比对（Analyse->image Combiner），最后发现Alpha plane 0 和Green plane 0 异或运算下的图出现了flag



异或对比常常是为了检查两张图片之间的差异，能发现我们肉眼看不到的及细微的差异。



上图是异或（XOR）对比出一张图片中的区别，所以用这个方法，也能把我们的信息隐藏在其中，但我觉得这并不是一种实用的方法。

后话

CTF中有关隐写术的这类题目真的是考验脑洞了，扩展开讲，还有能有很多很多可以举例的。Lsb隐写还能结合其他的隐写术，不只是能隐藏一些图片，还能把文件写入其中。这类题目这就需要用其他的工具进行文件提取分离，比如binwalk和foremost。当然，万变不离其宗，这里我们只要把原理搞清楚就足够了，剩下的就是解题思路，这就需要开开脑洞了。网上也存在着很多优秀的lsb隐写算法，能够实现更多复杂的操作，有兴趣的朋友可以进一步的去了解。

参考文章：https://www.tuicool.com/artic...