CTF杂项知识点

原创

Ka1tt 于 2020-10-03 15:16:08 发布 1996 收藏 51

分类专栏： CTF 文章标签： 运维

本文链接：https://blog.csdn.net/weixin_46555037/article/details/108910195

CTF-杂项

文件操作与隐写

1.File命令

当文件没有后缀名或者有后缀名而无法正常打开时，根据识别出的文件类型来修改后缀名即可正常打开文件

使用场景：不知道后缀名，无法打开文件

```
root@kali:~# file ctf
ctf: PNG image data, 731 x 672 //png文件
```

2.winhex

通过winhex程序中可以查看文件头类型，根据文件头类型判断出文件类型

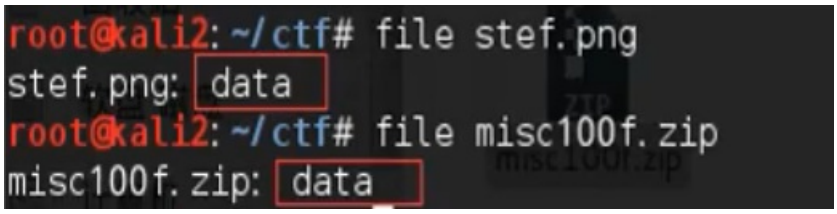
使用场景：windows下通过文件头信息判断文件类型

常见的文件头类型如图所示

文件类型	文件头
JPEG (jpg)	FFD8FFE1
PNG (png)	89504E47
GIF (gif)	47494638
TIFF (tif)	49492A00
Windows Bitmap (bmp)	424DC001
ZIP Archive (zip)	504B0304
RAR Archive (rar)	52617221
Adobe Photoshop (psd)	38425053
Rich Text Format (rtf)	7B5C727466
XML (xml)	3C3F786D6C
HTML (html)	68746D6C3E
Adobe Acrobat (pdf)	255044462D312E
Wave (wav)	57415645
pcap (pcap)	4D3C2B1A

3.文件头缺失/错误

通常文件无法正常打开有两种情况，一种是文件头缺失，一种是文件头错误。可以使用winhex打开文件，然后修改正确。



当file一个文件的时候显示data就是该文件头缺失或错误。

4.Binwalk文件分离工具

Binwalk是Linux下用来分析和分离文件的工具，可以快速分辨文件是否由多个文件合并而成，并将文件进行分离。如果分离成功会在目标文件的目录。有压缩包自动解压

同目录下生成一个形如_文件_extracted的文件目录，目录中有分离后的文件。

用法：

分析文件：binwalk filename

分离文件：binwalk -e filename

```
root@kali2:~/ctf# binwalk ans.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
8232          0x2028          TIFF image data, big-endian
19610         0x4C9A          Copyright string: " (c) 1998 Hewlett-Packard Company"
```

从0x0到0x2028(十六进制数)是一个JPG图片，从0x2028到0x4C9A是一个TIF文件。

5.foremost文件分离

如果binwalk无法正确分离文件，可以使用foremost，将目标文件复制到kali中，成功执行后，会在目标文件的文件目录下生成我们设置的目录，目录中会按文件类型分离出文件。有压缩包不会自动解压

用法：

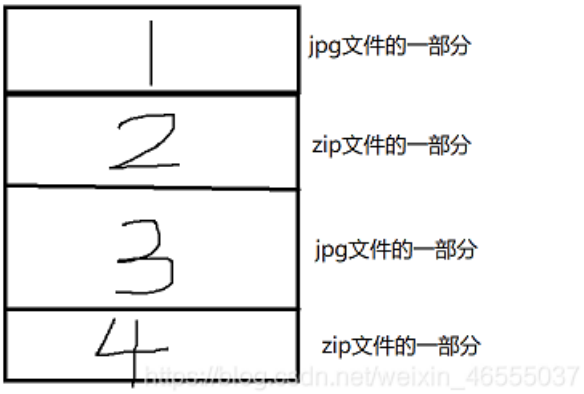
foremost 文件名 -o 输出目录名

```
root@kali2:~/ctf# foremost oddpic.jpg -o oddpic
Processing: oddpic.jpg
|*|
```

6.dd文件分离

当文件自动分离出错或者因为其他原因无法自动分离时，可以使用dd实现文件手动分析。

使用情况：



题目将jpg文件和zip文件分成两部分打乱排序。这时候使用binwalk和foremost是分离不出来的。

格式：

dd if=源文件 of=目标文件名 bs=1 skip=开始分离的字节

参数说明：

if=file #输入文件名，缺省为标准输入。

of=file #输出文件名，缺省为标准输出。

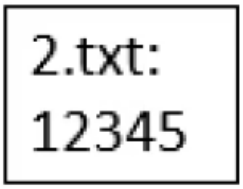
bs=bytes #同时设置读写块的大小为bytes，可代替ibs和obs。

skip=blocks #从输入文件开头跳过blocks个块后再开始复制。

命令一：

```
dd if=1.txt of=2.txt bs=5 count=1
```

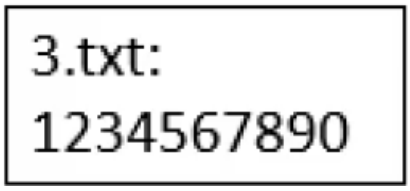
结果一：



命令二：

```
dd if=1.txt of=3.txt bs=5 count=2
```

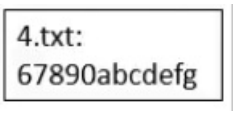
结果二：



命令三：

```
dd if=1.txt of=4.txt bs=5 count=3 skip=1
```

结果三：



一张sim.jpg已经通过binwalk分析出0-22895是jpg，22895-23046是zip使用dd将其分离

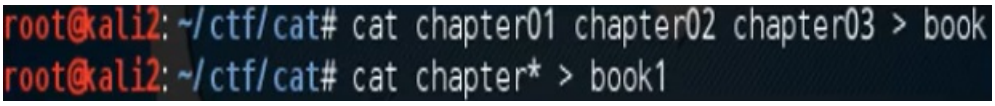
```
dd if=sim.jpg of=111111.zip bs=1 count=23046 skip=22895
```

7.文件合并

（1）Linux下的文件合并

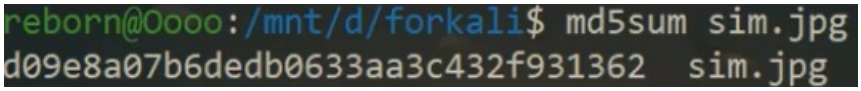
使用场景：出题人把一张图片分开好几个碎片给你。或对文件名相似的文件要进行批量合并。

格式：cat 合并的文件 > 输出的文件



完整性检测：Linux下计算文件md5（文件的md5题目会告诉你的）：

md5sum 文件名



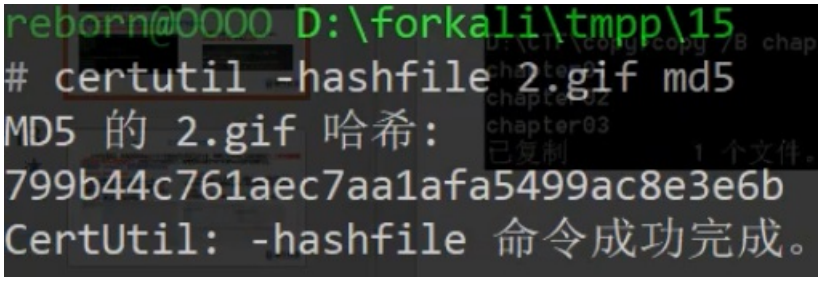
（2）Windows下的文件合并

使用场景：出题人把一张图片分开好几个碎片给你。或对文件名相似的文件要进行批量合并。

格式：copy /B 合并的文件名 输出的文件名

完整性检测：Windows下计算文件md5（文件的md5题目会告诉你的）：

certutil -hashfile 文件名 md5

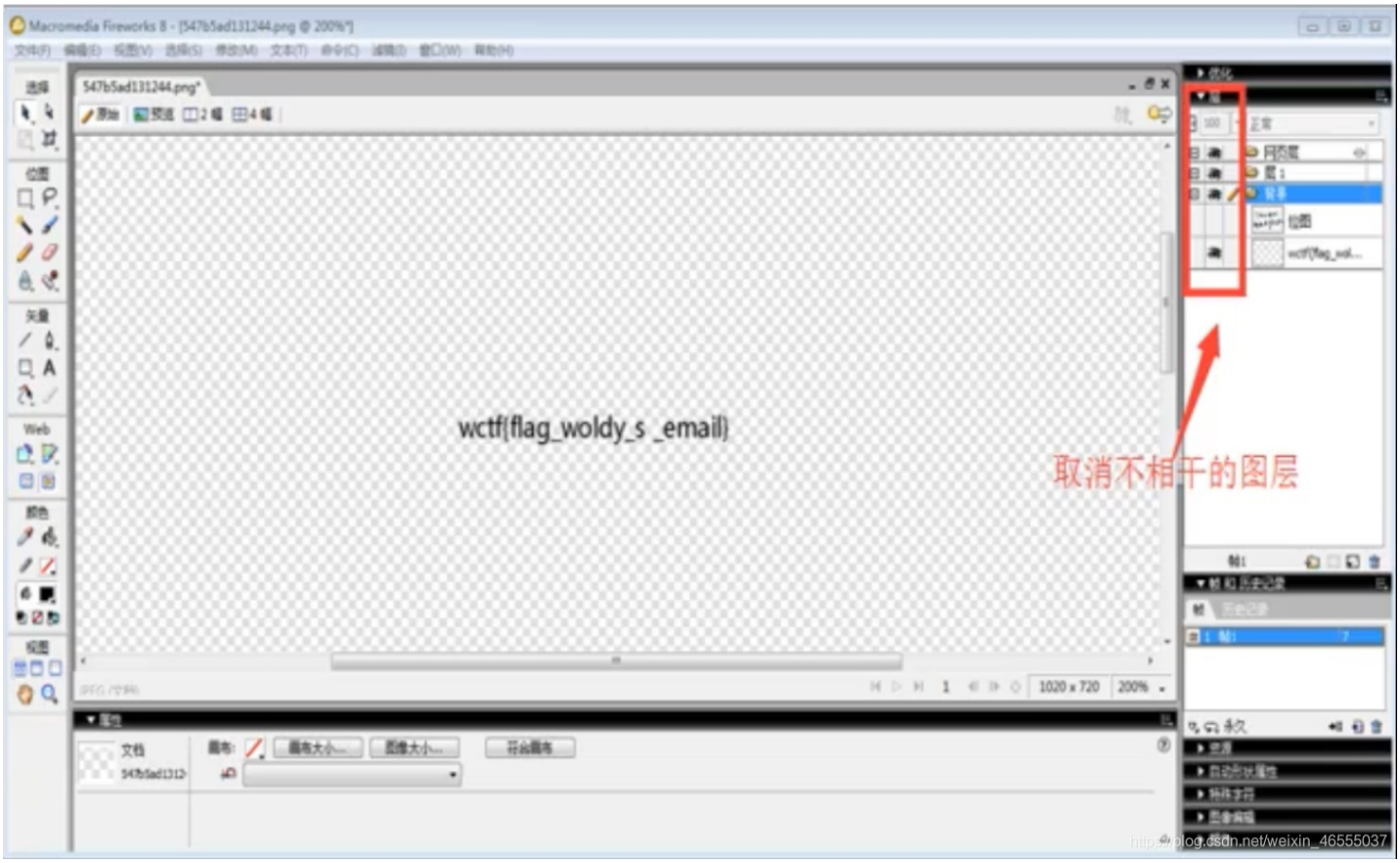


图片隐写

1.Firework

用十六进制编辑器打开文件时会看到文件头部中包含firework的标识，通过firework可以找到隐藏图片。firework可以用来查看gif动图的每一帧图片，或者一张图的每一个图层。例如flag，密码等等。

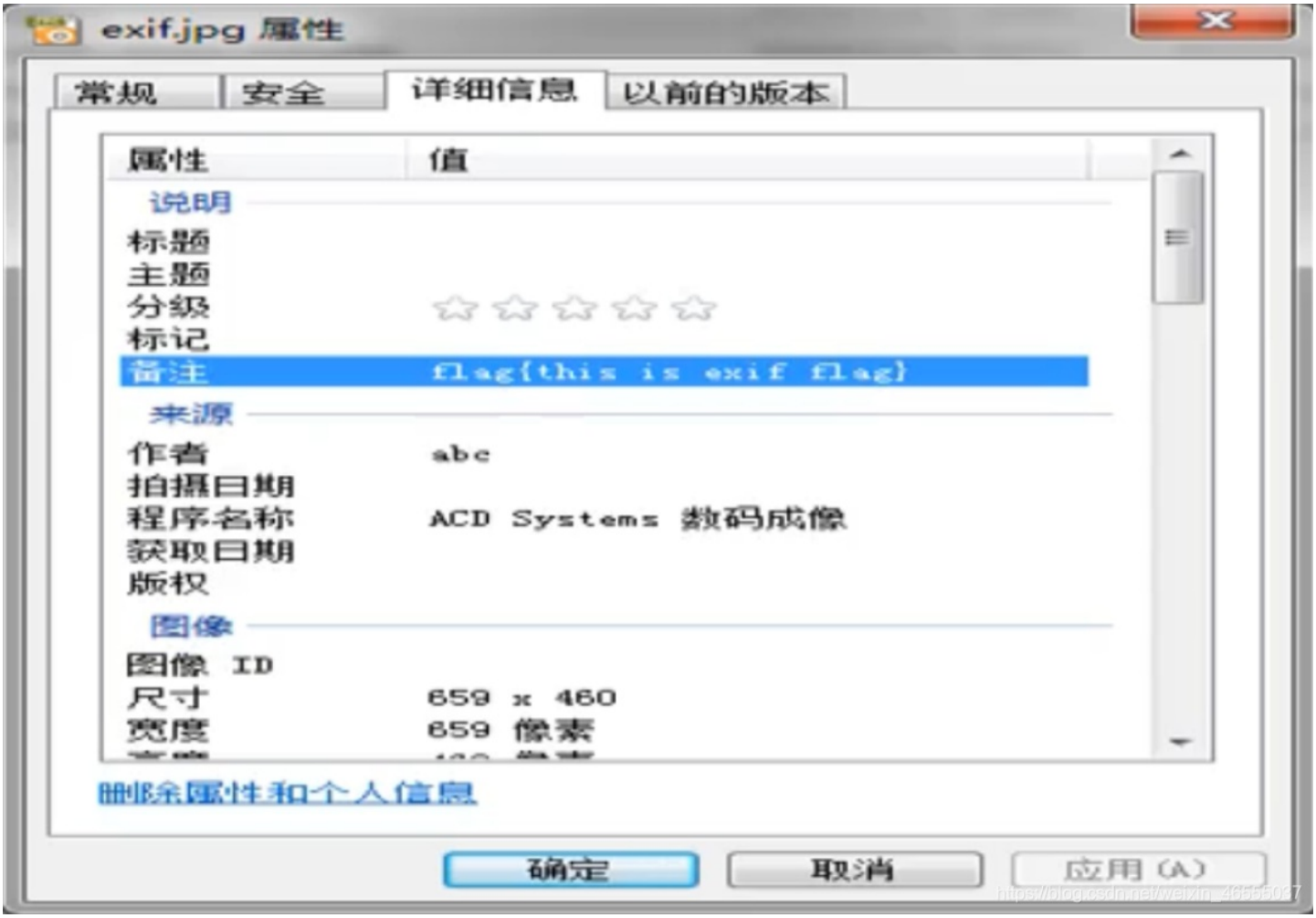
使用场景：查看隐写的图片文件



2.Exif

Exif按照JPEG的规格在JPEG中插入一些图像/数字相机的信息数据以及缩略图像.可以通过与JPEG兼容的互联网浏览器/图片浏览器/图像处理等一些软件来查看Exif格式的图像文件.就跟浏览器通常的JPEG图像文件一样.

图片右键属性，查看Exif或者查看详细信息，在相关选项卡中查找flag信息。

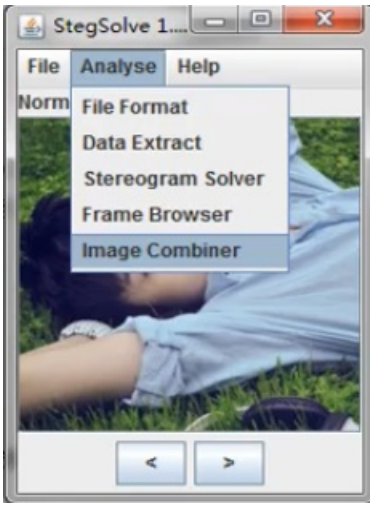


3.Stegsolve

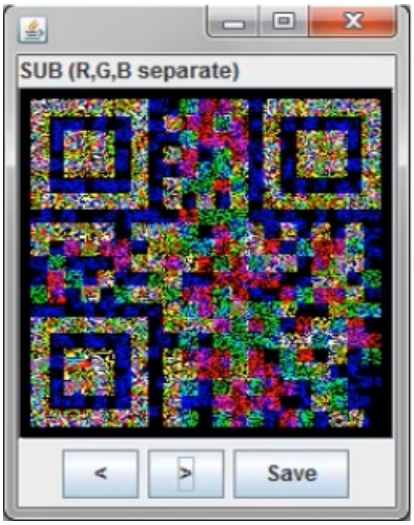
当两张jpg图片外观、大小、像素都基本相同时，可以考虑进行结合分析，即将两个文件的像素RGB值进行XOR、ADD、SUB等操作，看能否得到有用的信息，StegSolve可以方便的进行这些操作。

使用场景：题目给出两张基本一样的图片

1.打开一张图片，点击analyse–>Image combiner



2.在弹出的窗口中点击左右按钮选择处理方式，点击save保存有价值的结果。



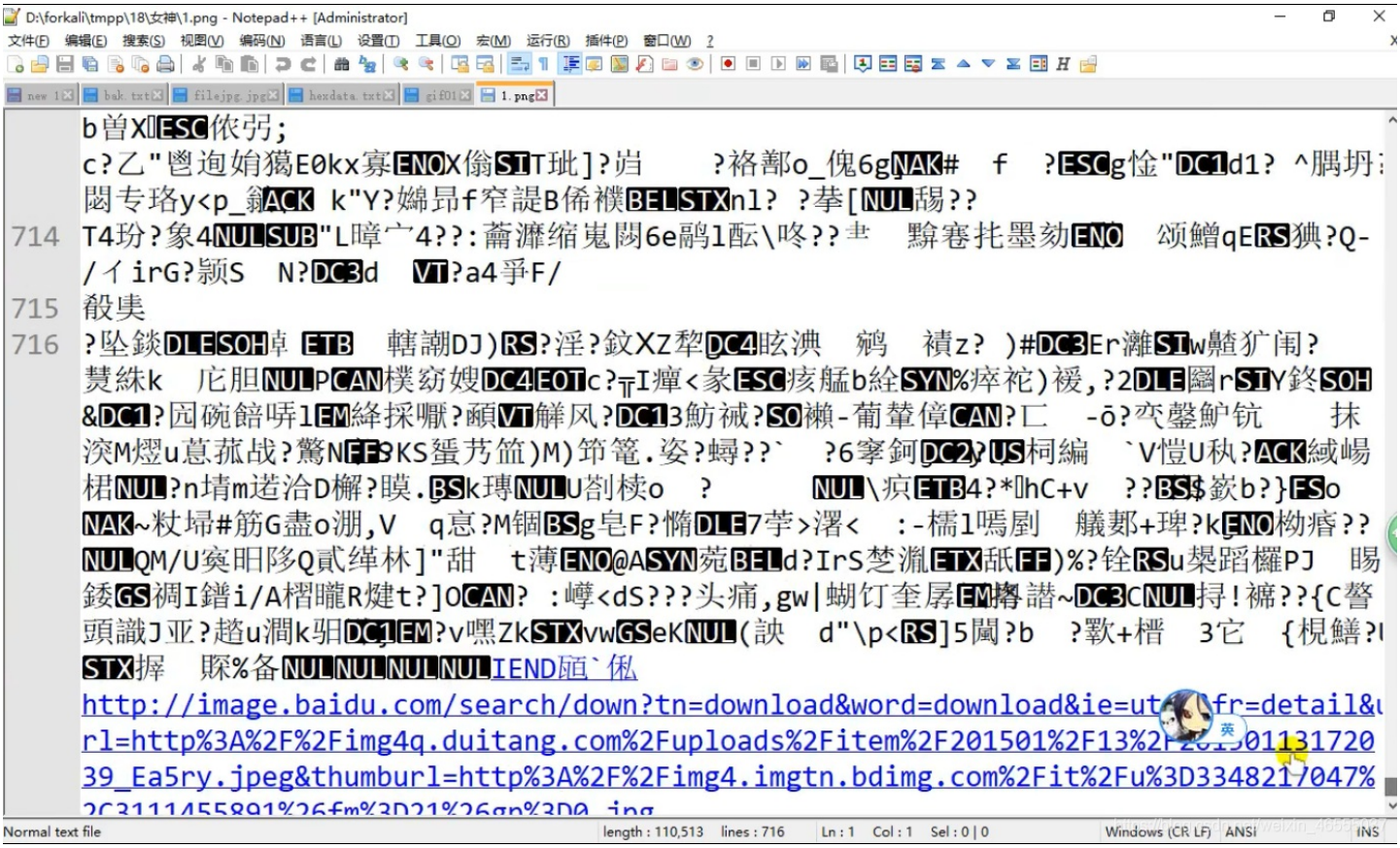
3.给出两张一样的图片的时候可以考虑使用StegSolve来将两张图片进行一个相加相减异或的操作。

例题：

题目给出一张图片

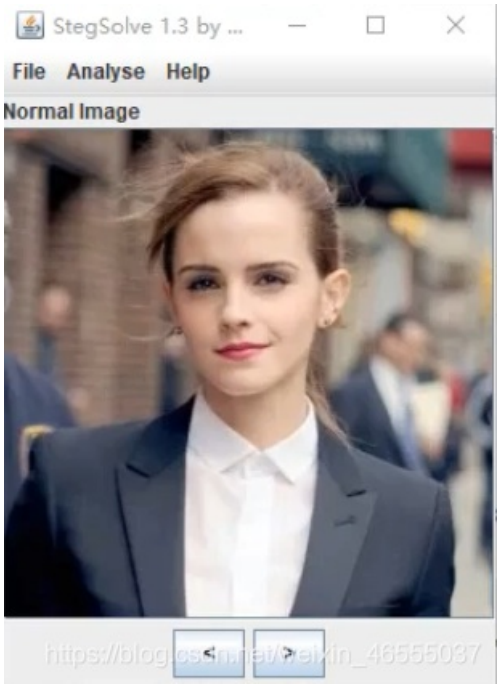

1.png

使用十六进制打开这张图片，发现一个下载链接

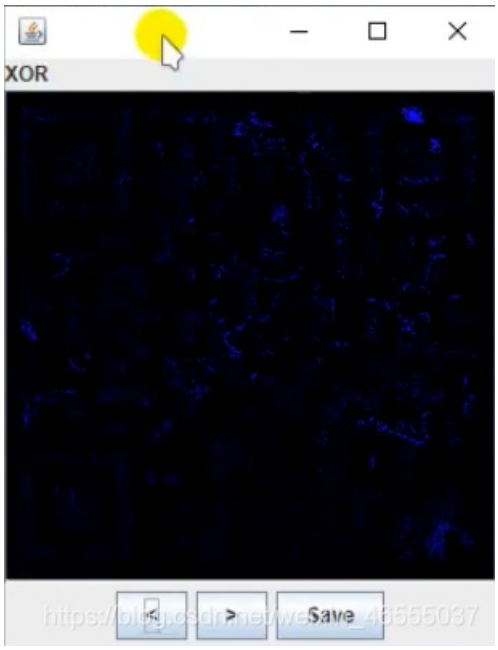


下载得到一张一模一样的图片

使用StegSolve左上角open打开下载出来的图（打开图片的顺序会影响结果，即1-2与2-1的区别）



点击Analyse–>Image Combiner打开另一张图



操作<>小箭头对两张图片进行异或加减各种操作得到一张二维码

最后对二维码进行一个取反的操作，即将黑白两色取反，最后得到一个flag。

4.LSB（最低有效位）

LSB替换隐写基本思想是用嵌入的秘密信息取代载体图像的最低比特位，原来的7个高位平面与替代秘密信息的最低位平面组合成含隐藏信息的新图形。

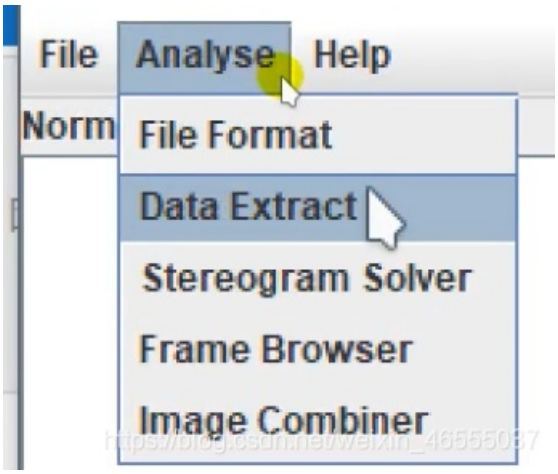
1.像素三原色（RGB）

2.通过修改像素中最低位的1bit来打到隐藏的效果

3.工具：StegSolve、zsteg、wbstego4、python脚本

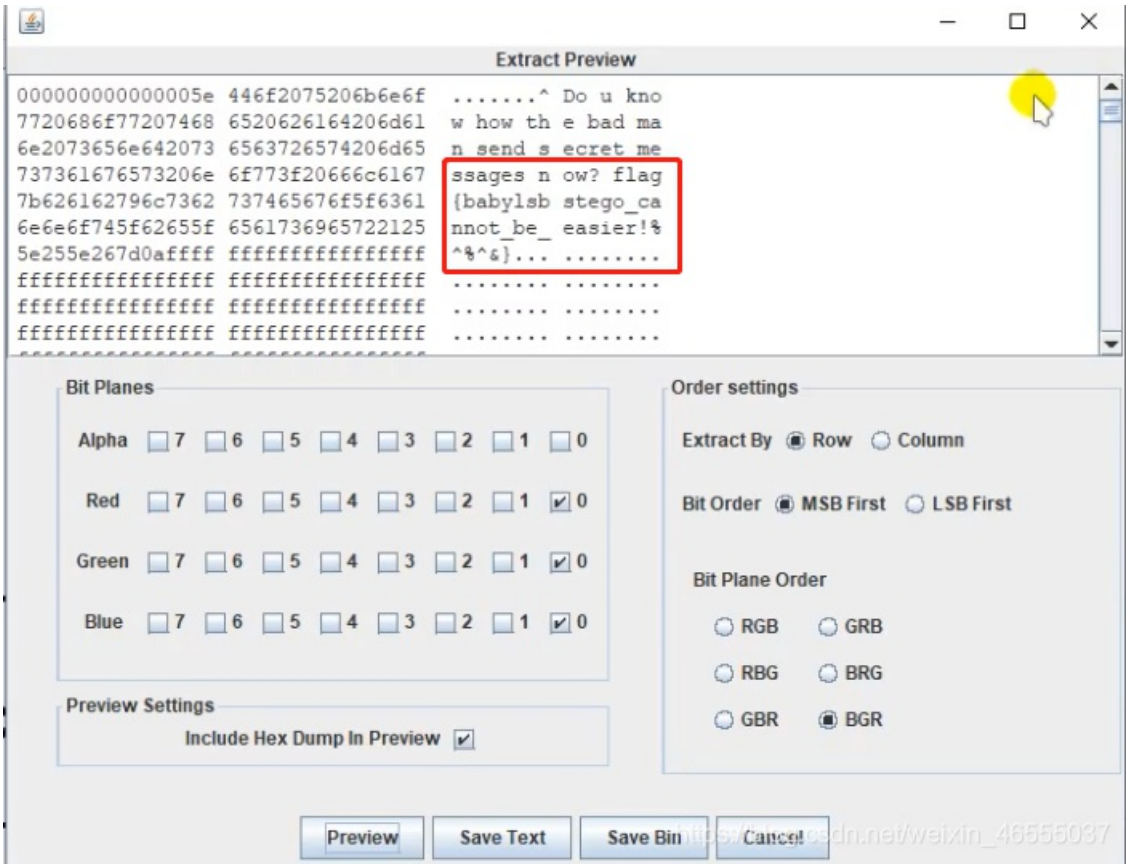
StegSolve

Analyse–>Data Extract



打开之后选择RGB最低有效位

在右边Bit Plane Order选项栏中选择三基色的顺序，然后点击Preview得到flag



zsteg

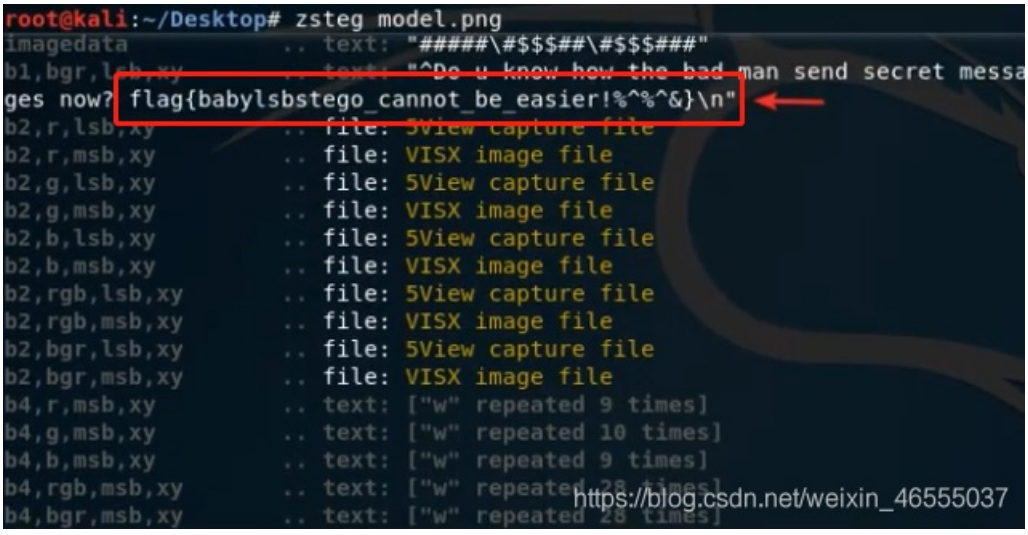
这个工具是在linux命令行上使用的

安装命令：

gem install zsteg

检测LSB隐写

zsteg xxx.png



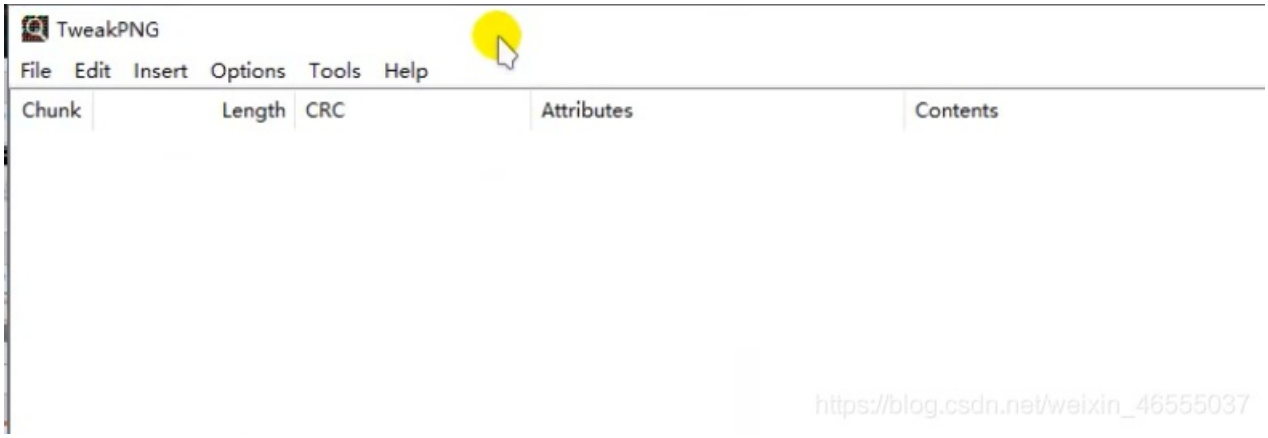
5.TweakPNG

TweakPNG是一款简单易用的PNG图像检查工具，它允许查看和修改一些PNG图像文件的元信息存储。

CRC值是根据文件头，图片长宽等数据来计算出来的一个检验值。

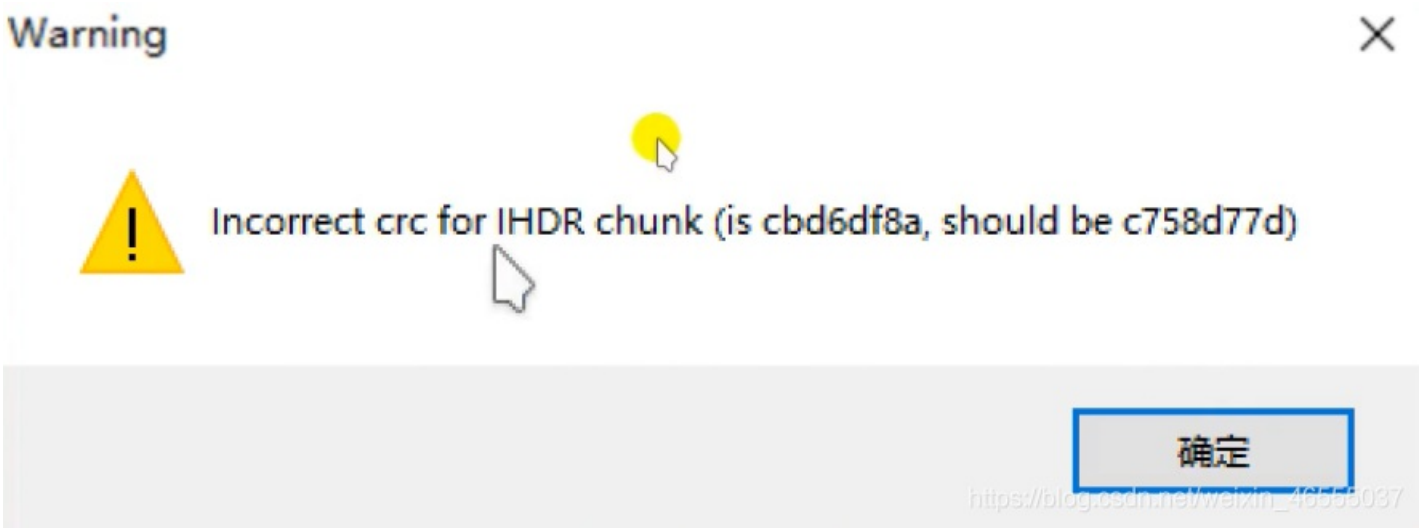
使用场景：文件头正常却无法打开文件，利用TweakPNG查看CRC值有没有错

打开TweakPNG



File–>open打开你要检查的图片

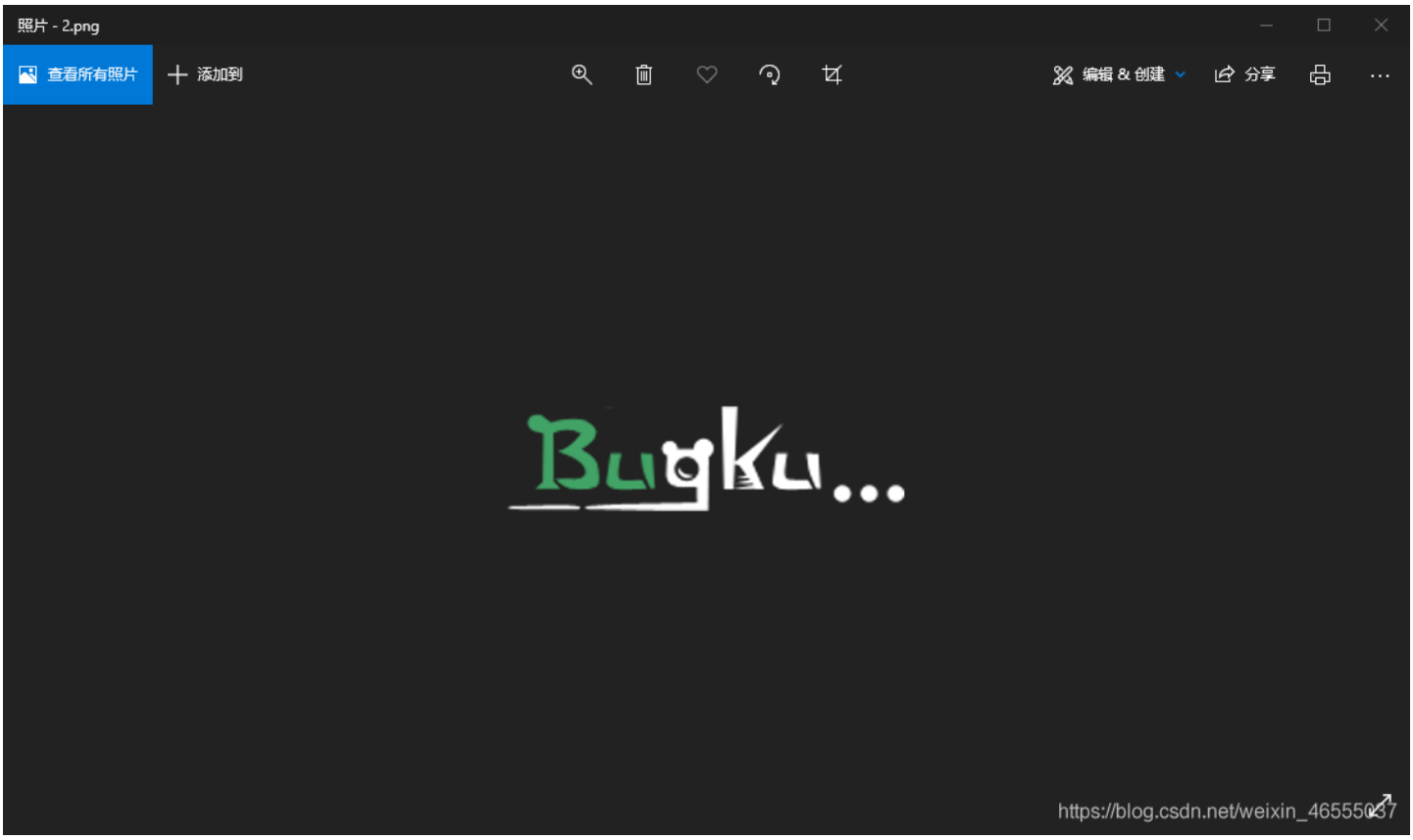
如果错误则会弹窗



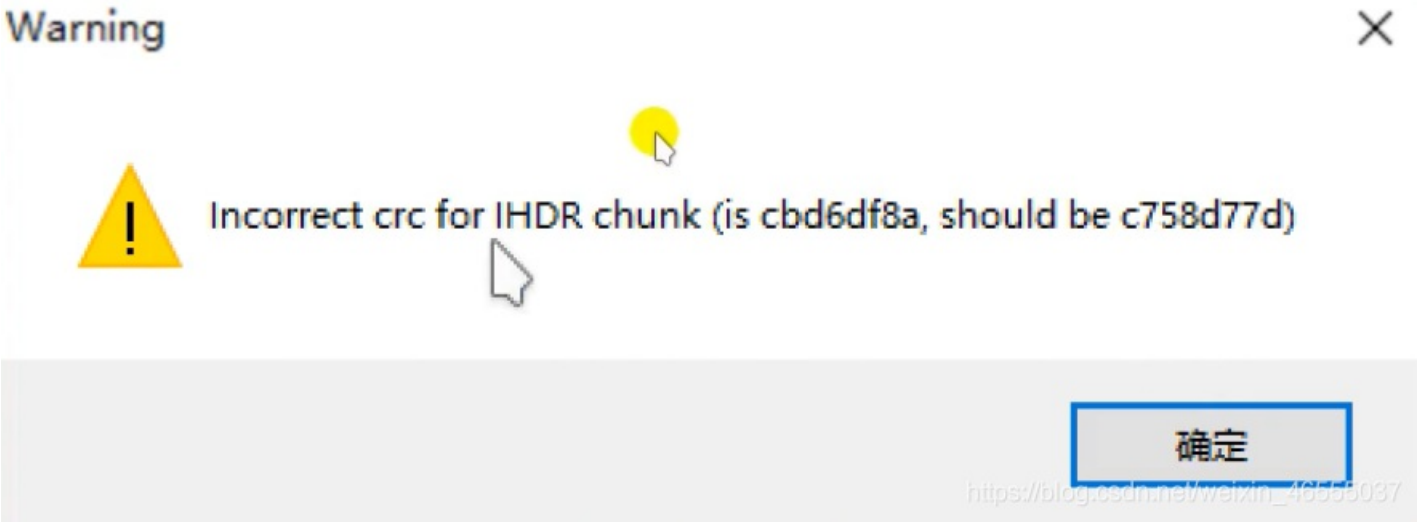
知道CRC错误了之后用十六进制编译器打开这张图片，Ctrl+f找到提示的错误CRC然后把错误的值改好就可以打开图片了。

第二种情况就是CRC值没有出现错误，但是图片的高度或者宽度发生了错误，也会打不开图片，或者图片显示不完全，这是就需要通过CRC计算出正确的高度或者宽度。

例：BUGKU里的一道题目，首先下载到一张图片

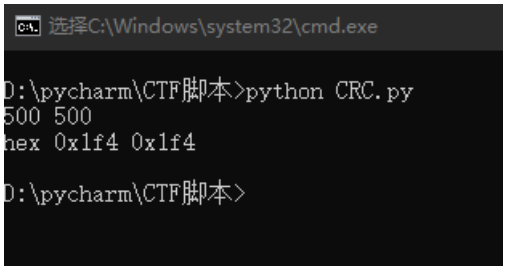


用TweakPNG检查发现这张图片的CRC值错误了

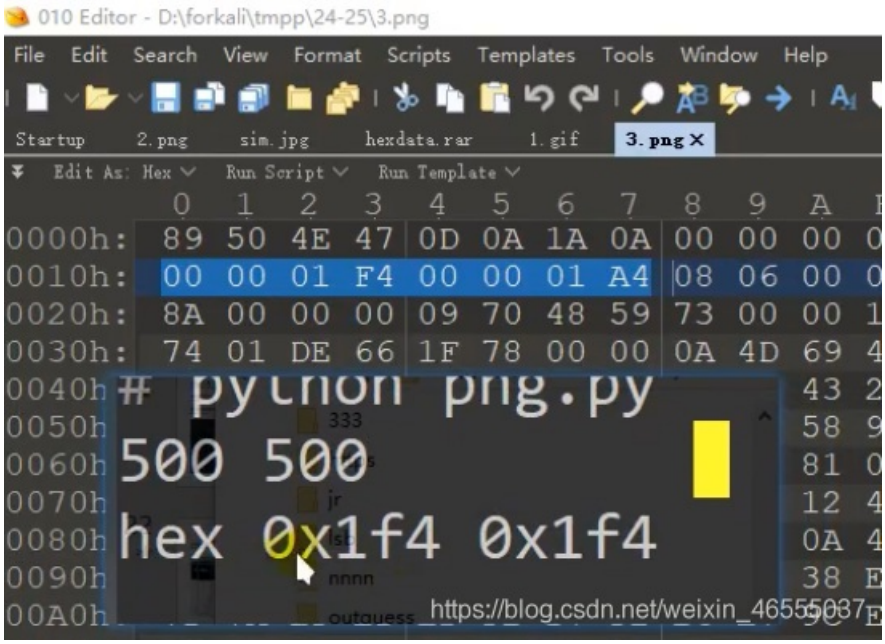


通过notepad++把CRC值改正之后再打开发现并没有得到Flag，说明了是这张图片的宽高设置有问题。

用python脚本通过CRC值计算出正确的宽高。



再打开图片对比查看



发现高的那段有错误 00 00 01 A4 —> 00 00 01 f4

修改正确之后保存文件，打开图片发现底部有flag

6.Bftools

Bftools用于解密图片信息。

使用场景：在windows的cmd下，对加密过的图片文件进行解密

格式：

Bftools.exe decode braincopter 要解密的图片名称 -o 输出文件名

Bftools.exe run 上一步写的输出的文件名

```
D:\CTF\bftools\bftools>bftools.exe decode braincopter zzzzzzyu.png --output 123.
png

D:\CTF\bftools\bftools>bftools.exe run 123.png
XDCTF{ji910-dad9jq0-iopuno}
D:\CTF\bftools\bftools>
```

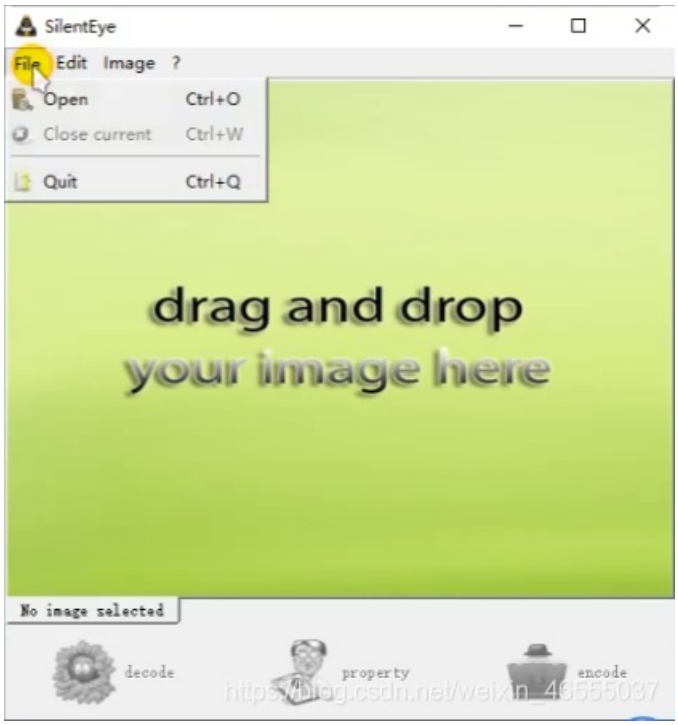
7.SilentEye

SilentEye是一款可以将文字或者文件隐藏到图片的解密工具。

使用场景：windows下打开SilentEye工具，对加密的图片进行解密

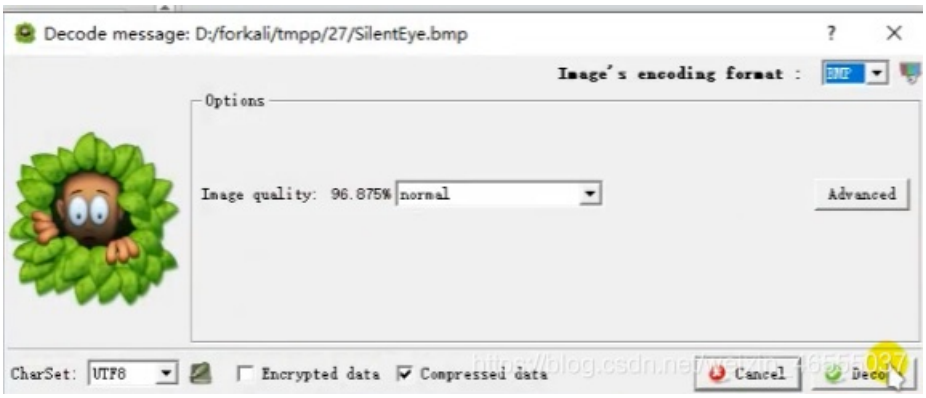
例：

打开SilentEye—>open 打开你想解密的图片

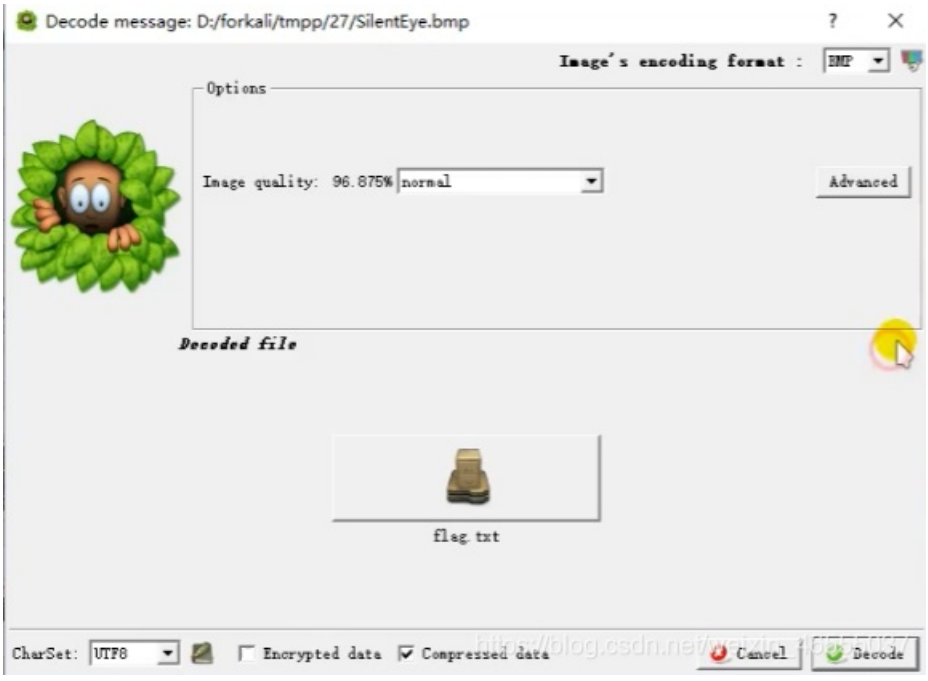


打开图片之后选择左下角decode解密

点击Decode

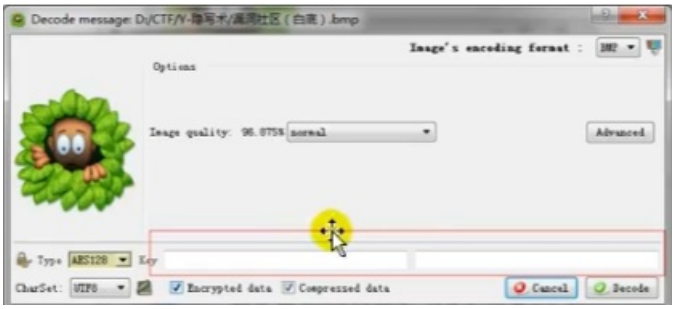


把生成的flag.txt文件保存在你想保存的地方



打开flag.txt得到解密后的flag值

如果解密的时候需要密码，勾选encrypted data，输入密码和确认密码，再点击decode解密



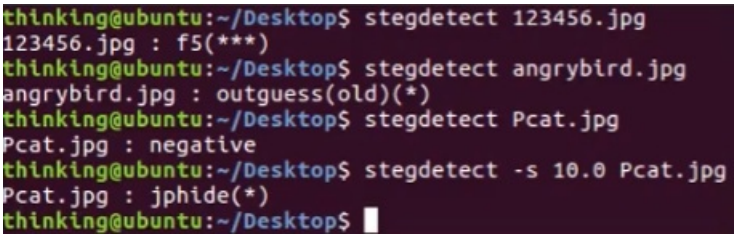
8.Stegdetect

Stegdetect程序专门用于分析JPEG文件，如果确定了这个文件肯定使用了jpg的加密方式就可以用Stegdetect。因此用Stegdetect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。

格式：

stegdetect xxx.jpg

stegdetect -s 敏感度 xxx.jpgexi



压缩文件处理

1.伪加密

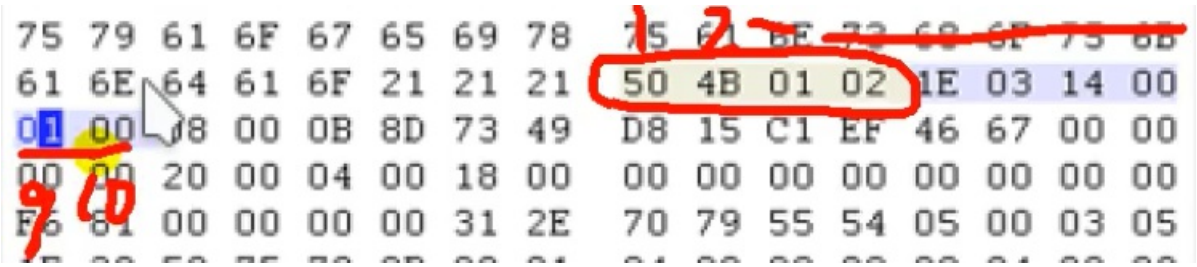
如果压缩文件是加密的，或文件头正常但解压错误，首先尝试文件是否伪加密。zip文件是否加密是通过标识符来显示的，在每个文件的文件目录字段有一位专门标识了文件是否加密，将其设置成00表示该文件未加密，如果成功解压则表示文件为伪加密，如果解压出错说明文件为真加密。

使用场景：伪加密文件

操作方法：使用十六进制编译器打开压缩文件，找到从文件头开始数的第九第十个字符，将其修改为0000.

（1）Zip

1.十六进制编译器打开文件Ctrl+f搜索16进制的504B0102，可以看到每个加密文件的文件头字段。



第九第十位为00 00的话是无加密，如果是00 01或者01 00的话是伪加密，将00 01改成00 00后解除伪加密状态。

（2）RAR

RAR文件由于有头部检验，使用伪加密时打开文件会出现报错，使用十六进制编译器修改标志位后如报错消失且正常解压缩，说明是伪加密。使用十六进制编译器打开RAR文件，找到第24个字节，该字节位数为4表示加密，0表示无加密，将尾数改为0即可破解伪加密。

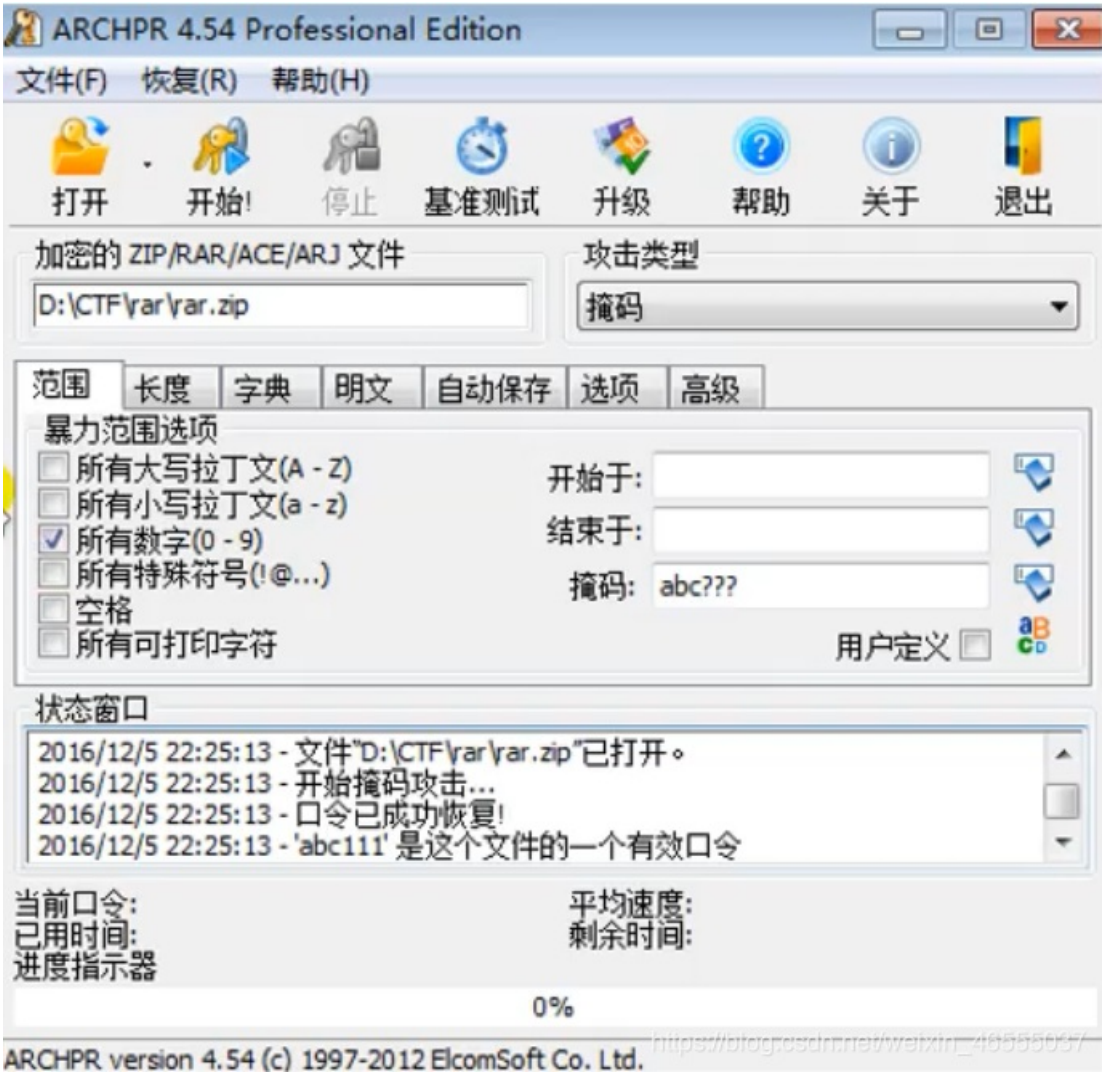
2.暴力破解

通常我们可以使用ARCHRP.exe工具来破解zip文件

windows下加密过的zip文件

1.攻击类型选择暴力破解，在范围位置根据提示选择暴力破解范围选项设置暴力破解包含的类型、开始于和结束于选项具体范围，如果没有定义则全范围暴力破解。点击打开选择要破解的文件，点击开始进行破解。建议使用1~9位的数字密码，以及系统自带的英文字典作为密码字典。

2.攻击类型选择掩码可以进行复杂的暴力破解，比如知道密码前3位是abc，后三位是数字，则在攻击类型选择掩码，在掩码出输入abc???，暴力范围选择所有数字，打开要破解的文件，点击破解。此时？？？的部分会被我们选择的暴力破解范围中的字符代替。



破解成功后会提示（口令就是密码）

3.明文攻击

明文攻击指的是，加密的ZIP文件中你已经知道了这个ZIP加密文件中的其中一个或多个文件，李永忠和谐内容推测出密钥并解密ZIP文件的攻击方法，相比于暴力破解，这种方法在破解密码较为复杂的压缩包时效率更高。例如：假设一个加密的压缩包中有两个文件readme.txt和flag.txt，其中flag.txt的内容是我们希望知道的内容，而我们拥有readme.txt这个明文文件，这时候就可以使用明文攻击了。

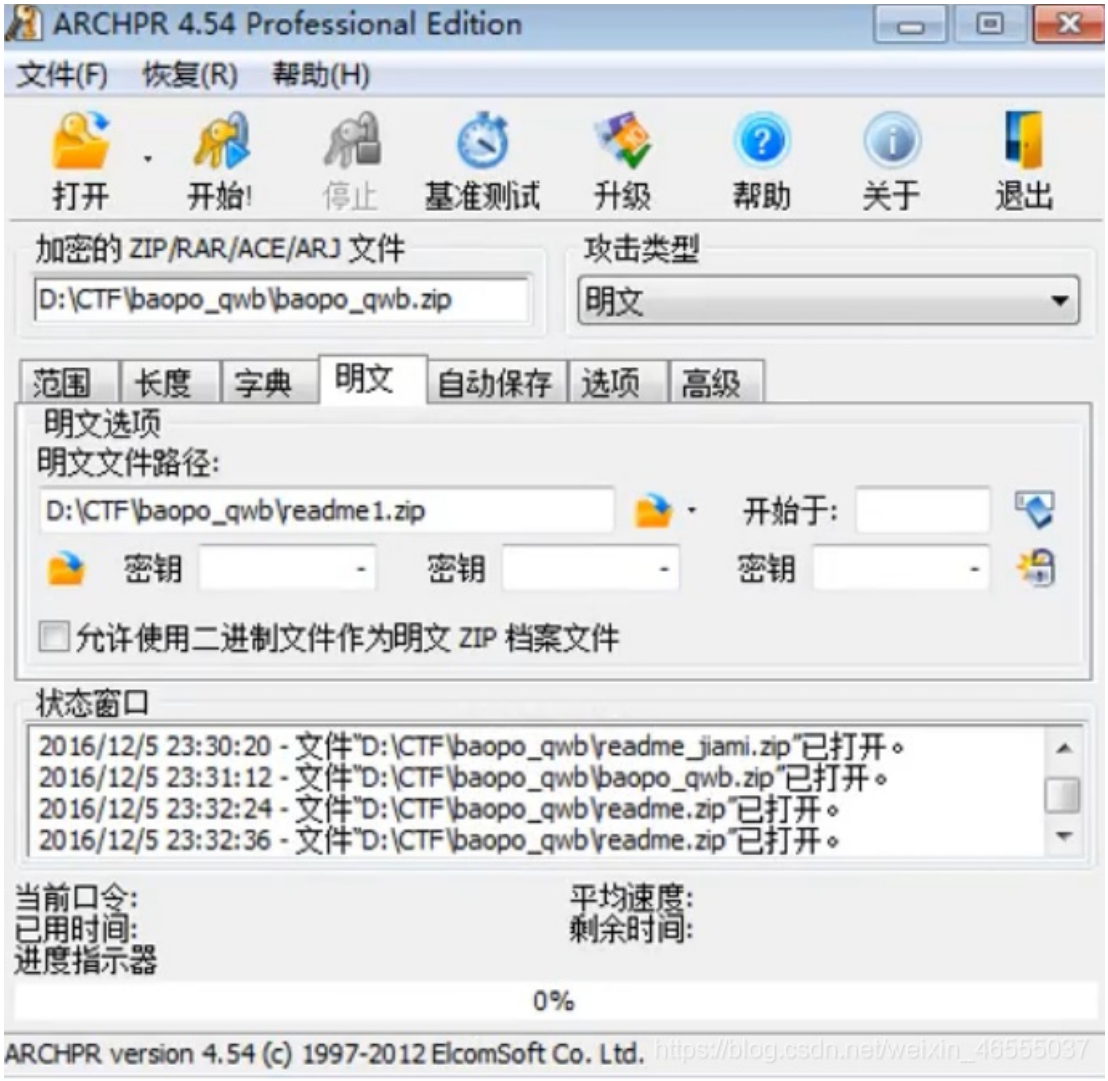
操作:

1、将readme.txt的明文文件进行压缩，变成readme1.zip[#明文文件的压缩算法一定要和加密的压缩文件的算法一样！！不然会出错！

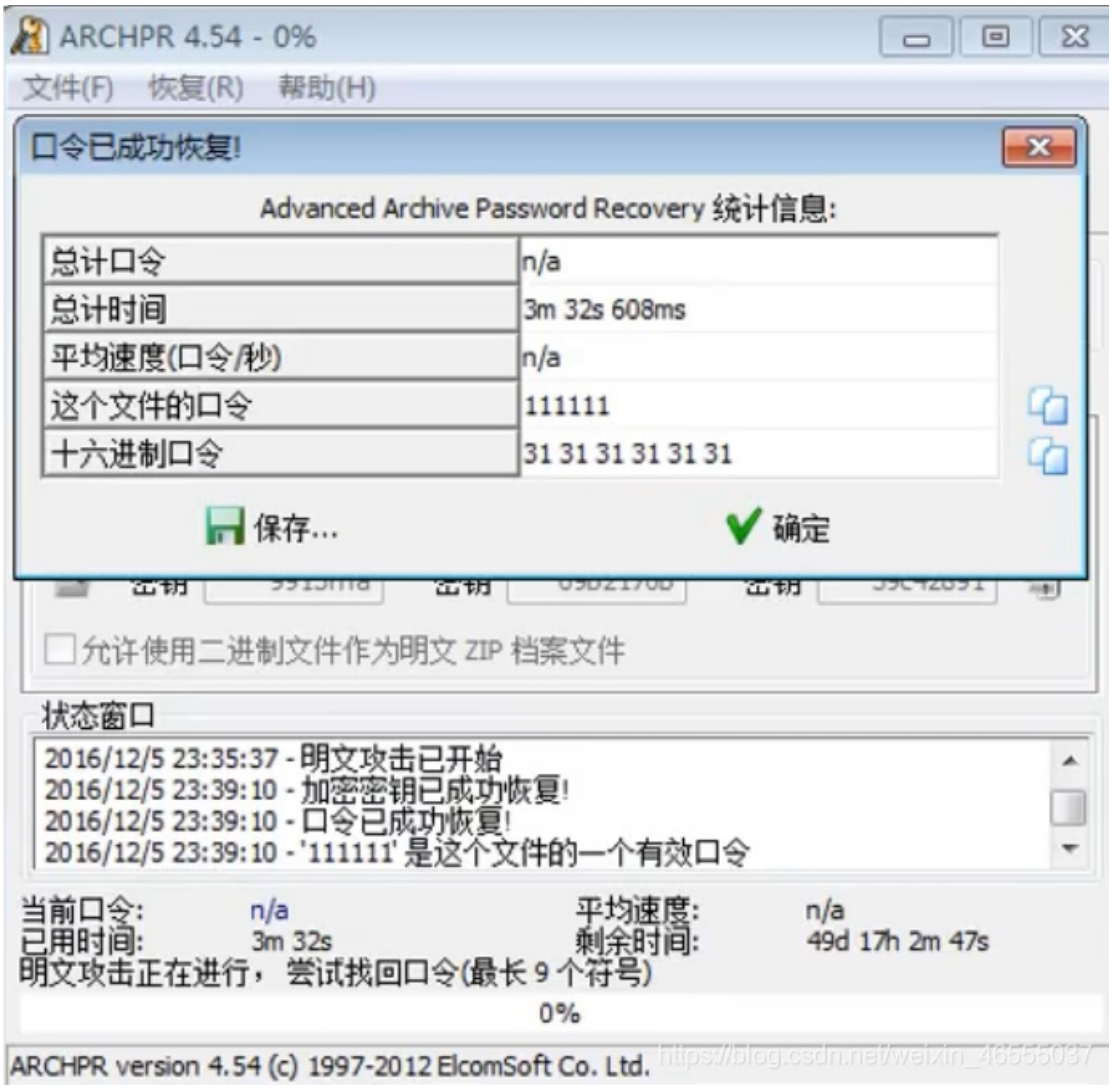
2、打开ARCHPR，攻击类型选择明文，明文文件路径选择readme1.zip即将明文文件不加密压缩后的文件）

3、选择要破解的文件，点击开始，破解成功后会获得密码。

图一

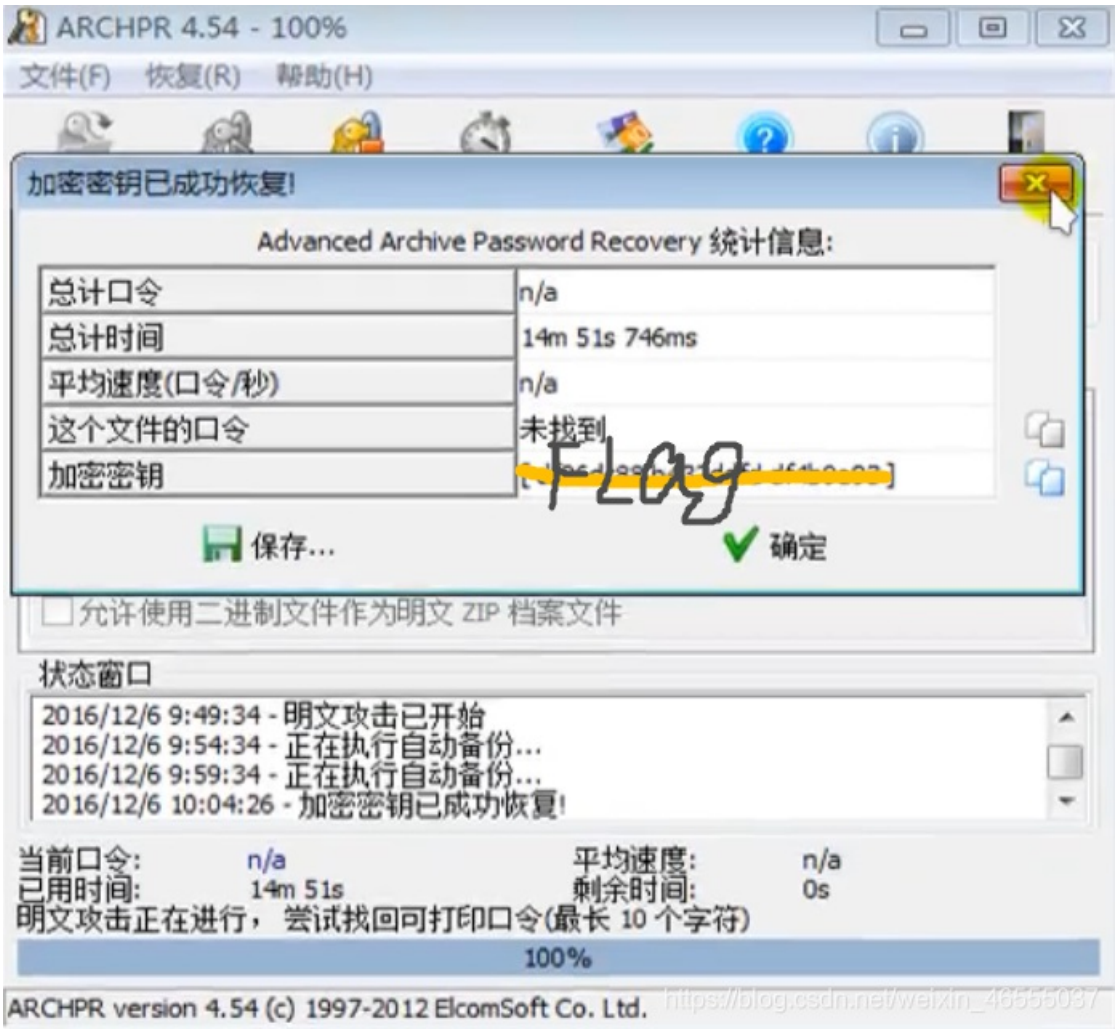


图二



有时候不一定能破解出文件口令，但是能够找到加密密钥等信息，可以直接将文件解密，点击确定保存解密后的文件即可。

还有一种情况就是加密的文件无论如何都是解密不了的，通过破解软件可以得到它的加密密钥，Flag就在加密密钥上



流量取证技术

流量取证题目通常都是给出一个pcac文件，需要用wireshark对这个pcac文件进行一个流量检测。

过滤ip地址

如源IP或者目标X.X.X.X

ip.src eq x.x.x.x or ip.dst eq x.x.x.x 或者 ip.addr eq x.x.x.x

过滤端口

tcp.port eq 80 or udp.port eq 80

tcp.dstport == 80 只显示tcp协议的目标端口为80

tcp.srcport == 80 只显示tcp协议的源端口为80

tcp.port >=1 and tcp.port <= 80

过滤协议

直接tcp/udp/arp/icmp/http/ftp/dns/ip…

过滤MAC

eth.dst == A0:00:00:04:C5:84

包长度过滤

udp.length == 26 # 这个长度是指udp本身固定长度8加上udp下面那块数据包之和

tcp.len >= 7 # 指的是ip数据包（tcp下面那块数据），不包括tcp本身

ip.len == 94 # 除了以太网头固定长度14，其他都算是ip.len，即从ip本身到最后

frame.len == 119 # 整个数据包长度，从eth开始到最后

http模式过滤

http.request.method == “GET” # 筛选http协议里面的GET请求方法

http.request.method == “POST” # 筛选http协议里面的POST请求方法

http.request.uri == “/img.logo-edu.gif” #

http contains “GET” # 筛选http协议里面包含GET字段的包

http contains “HTTP/1.” # 筛选http协议里面包含HTTP/1字段的包

http.request.method == “GET” && http contains “User-Agent:”

http contains “flag” # 筛选http协议里面包含flag字段的包

http contains “key” # 筛选http协议里面包含key字段的包

TCP类的也可以这样：

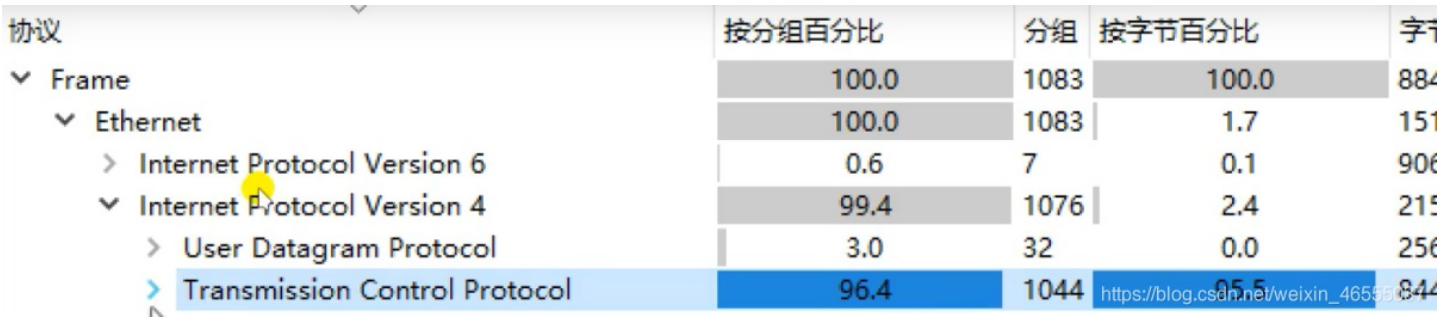
tcp contains “flag” # 筛选TCP协议中有包含flag字段的包

WireShark协议分析

解析：在出题人的角度看，出题人在制作这个流量包的时候肯定是先写好过滤代码—>点击开始抓包—>发送图片—>停止抓包。所以做出来的流量包绝大部分应该是与flag相关的，只有少部分是与flag无关的没来得及剔除的无关流量，所以使用一个协议分级能很好的找出与flag相关的包。

1.点击统计—>协议分析

2.分析哪些包占比大

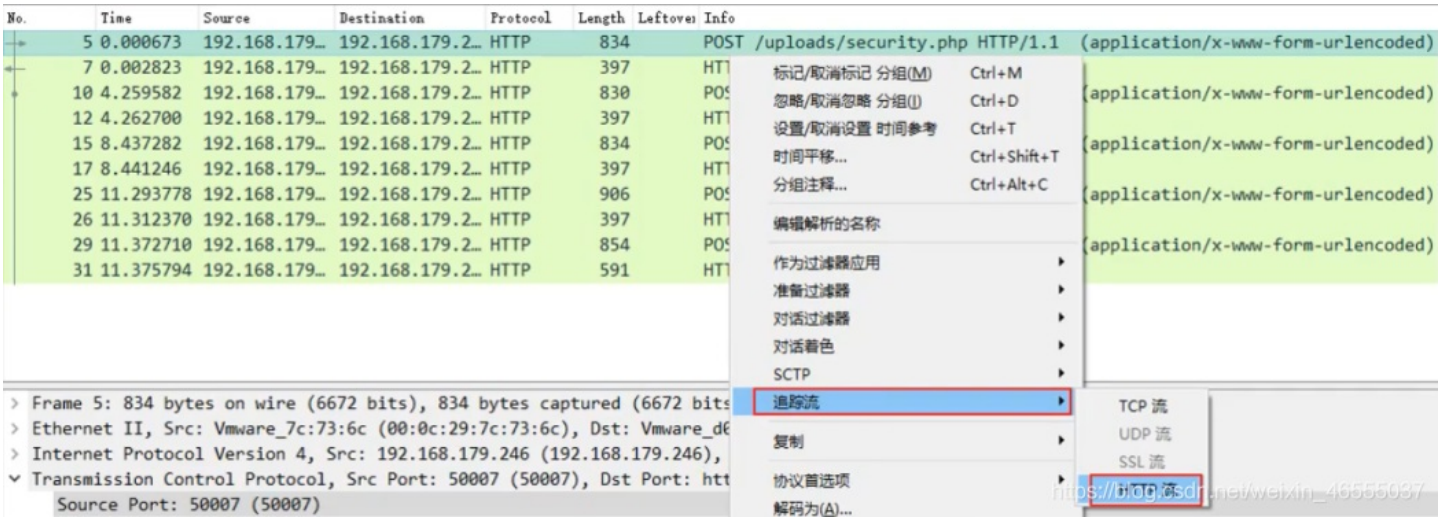


比如这个 IPv6很少 IPv4很多，在IPv4里面又有UDP和TCP，TCP占大多数。

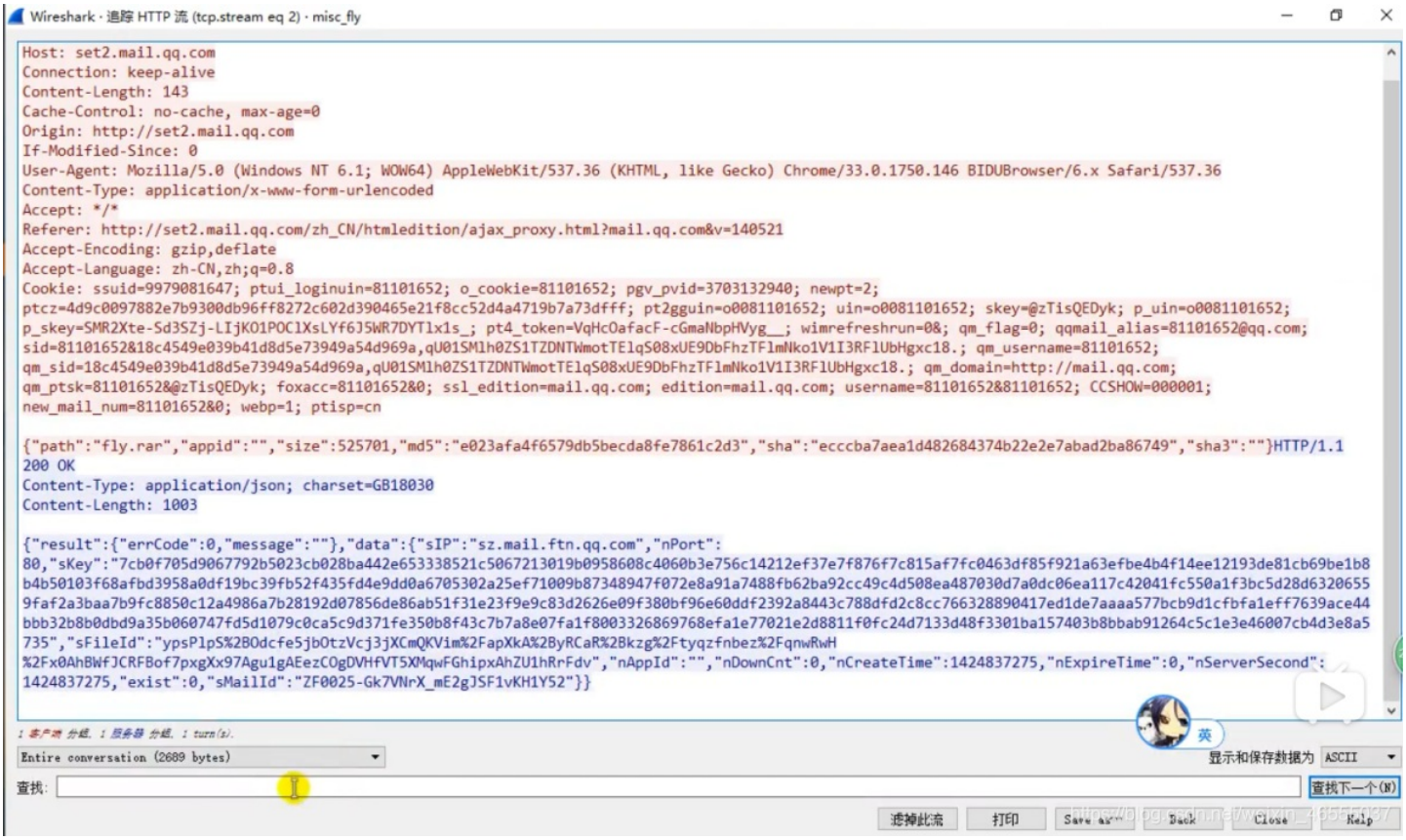
WireShark流汇聚

再按关注的http数据包或tcp数据包中选择流汇聚，可以将HTTP流或TCP流汇聚或还原成数据，在弹出的框中可以看到数据内容。

HTTP流：



选择完HTTP流之后会弹出一个窗口



红色的是文件头，蓝色的正文内容，通常flag都藏在正文内容中。

常见的HTTP流关键内容：

1.HTML中直接包含重要信息。

2.上传或下载文件内容，通常包含文件名、hash值等关键信息，常用POST请求上传。

3.一句话木马，POST请求，内容包含eval，内容使用base64加密。