CTF杂项-BUUCTF竞赛真题WriteUp(2）

原创

Tr0e 于 2021-05-05 23:44:11 发布 1448 收藏 24

分类专栏： CTF之路

本文链接：https://blog.csdn.net/weixin_39190897/article/details/116430110

CTF之路 专栏收录该内容

17 篇文章 27 订阅

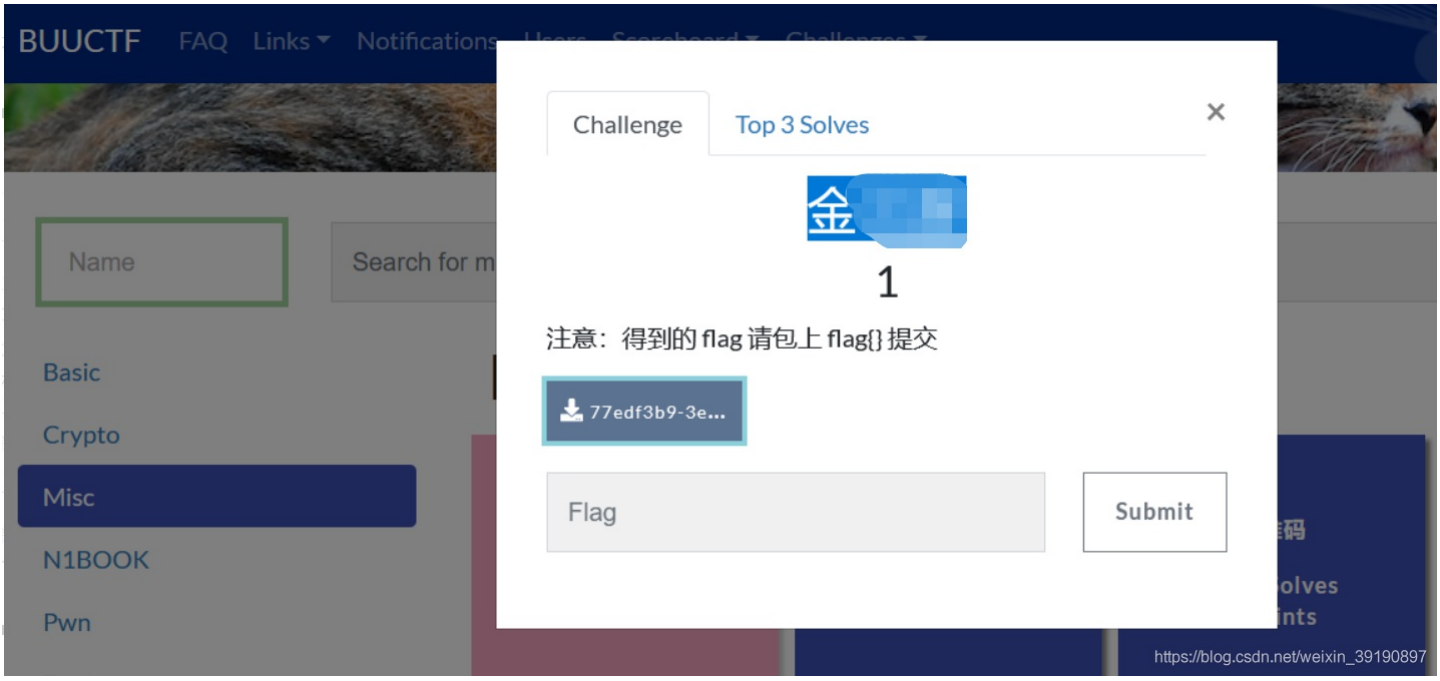
订阅专栏

文章目录

前言

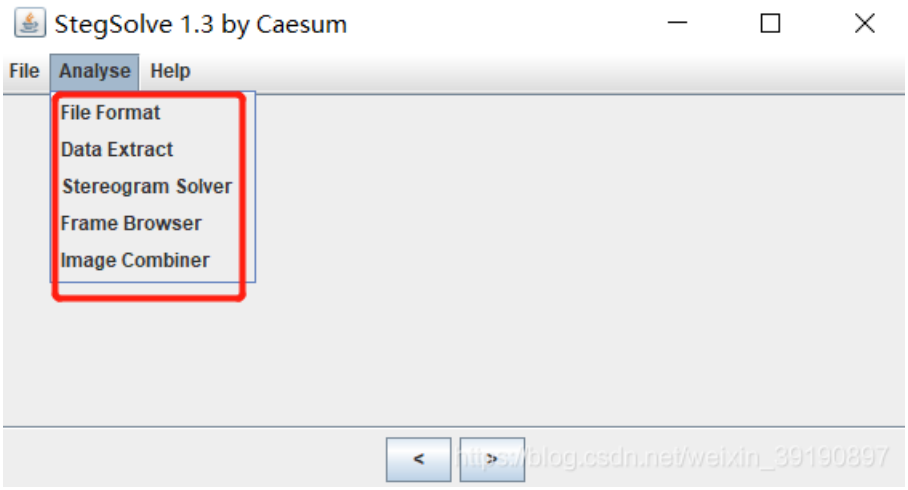
为了划水过几天的“ 红帽杯 ” 网络安全大赛，学习并记录下 BUUCTF 平台的杂项部分题目，因为去年参加“ 强网杯 ”网络安全大赛发现杂项类型的题目还是可以争取得分的，也希望过几天运气好点吧…

No.1 Git动图分解提取信息



下载附件发现是一个动图，闪烁着红色 flag 字体。该题需要用到隐写图片查看的神器------**stegsolve**，下载地址。

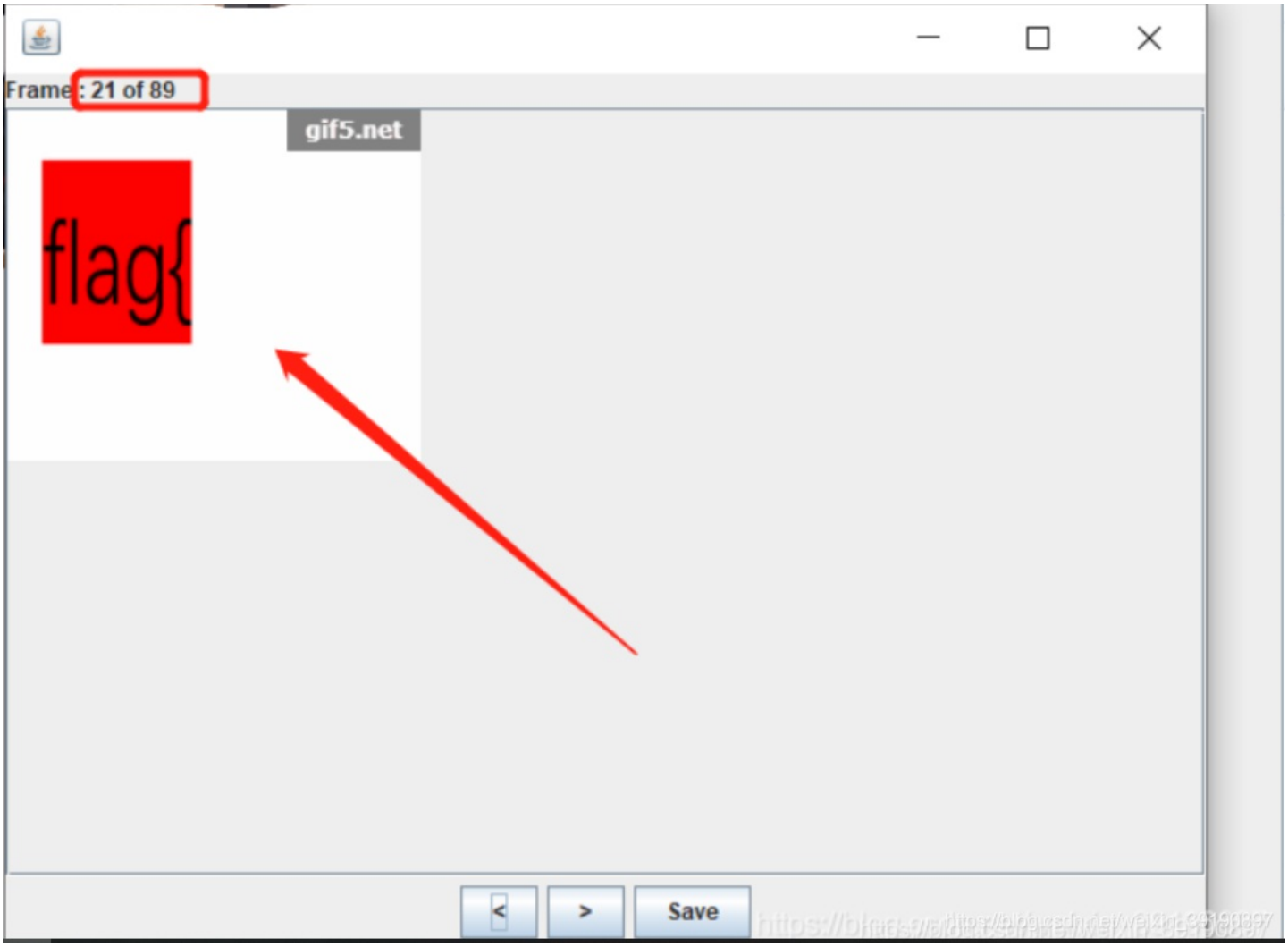
Stegsolve 功能简介：



上面是软件打开的界面，界面简单。主要供能为 analyse，下面对 Analyse 下面几个功能键作简单介绍：

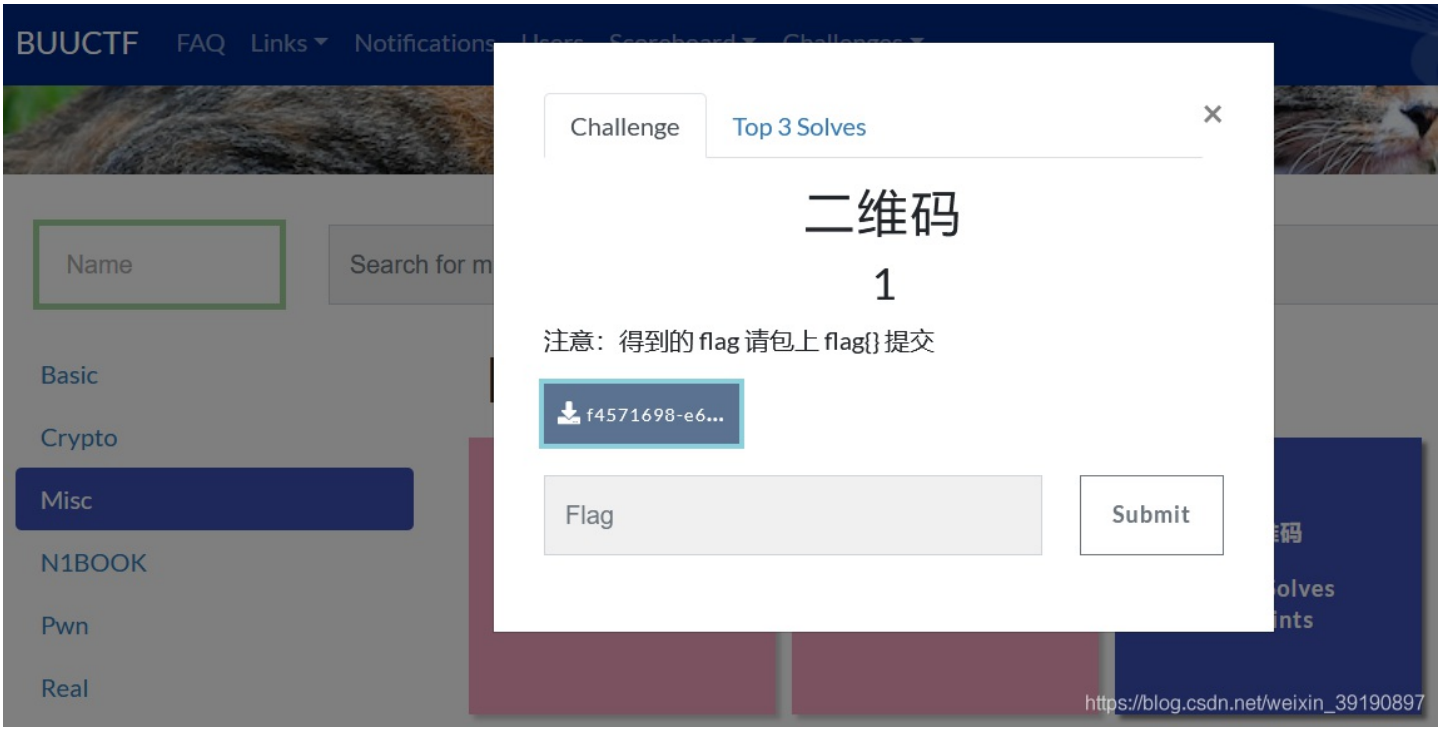
功能键	作用
File Format	文件格式，这个主要是查看图片的具体信息
Data Extract	数据抽取，图片中隐藏数据的抽取
Frame Browser	帧浏览器，主要是对GIF之类的动图进行分解，动图变成一张张图片，便于查看
Image Combiner	拼图，图片拼接

此题是动图中闪烁着 flag，故使用 Frame Browser 功能对 git 动图进行分解，获得 89 张分解后的静图，查阅获得 flag：

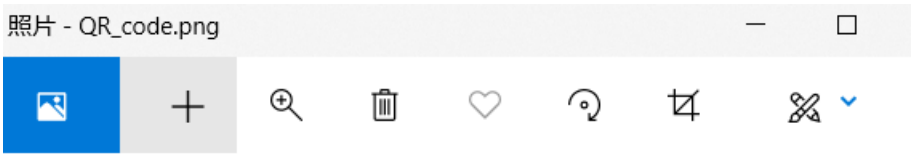


获得 flag：`flag{he11ohongke}`，本入门题完。

No.2 隐藏文件提取与爆破



1、下载完打开是个二维码，用微信扫描只能看见一句话——“secret is here”。



2、猜测二维码图片中隐藏了别的文件，可以使用 Binwalk 工具进行文件分析和分离。

1.Binwalk 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。会在目标文件的目录。

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

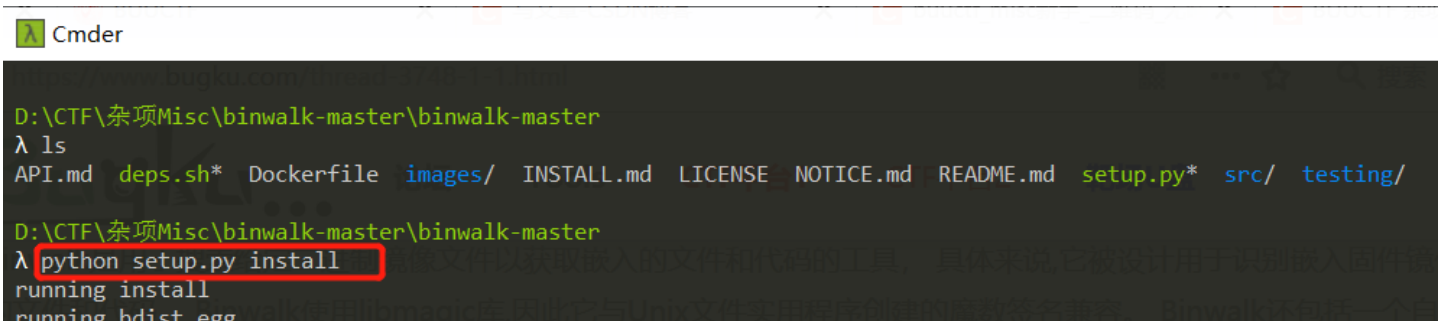
用法：

分析文件：binwalk filename

分离文件：binwalk –e filename

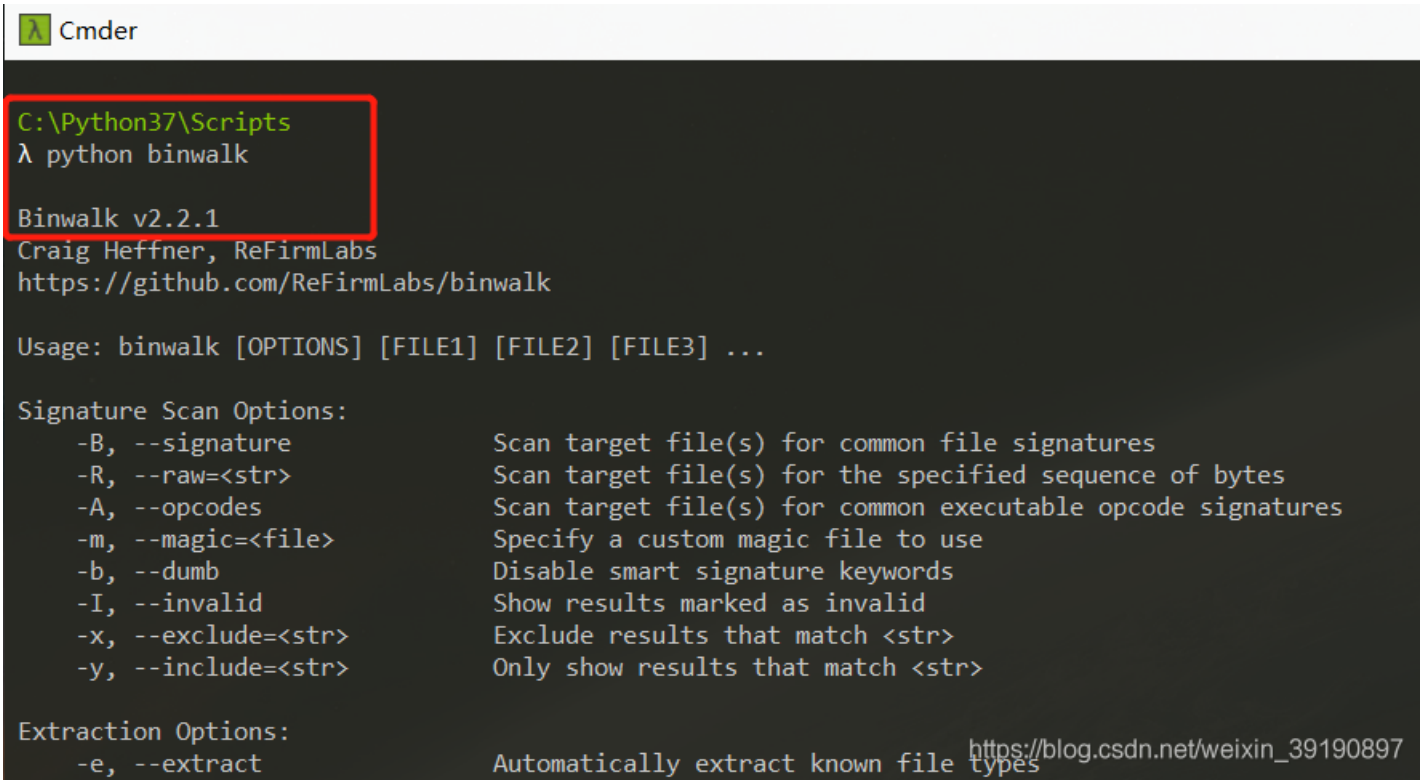
```
root@kali2:~/ctf# binwalk ans.jpg

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 Companyny"
```


Kali Linux 自带了 Binwalk 工具，但我不想开虚拟机……所以在 Windows 下安装，Github下载源码，本地安装好 python 并进入 Binwalk 所在文件夹，执行命令如下，程序将会自动安装：

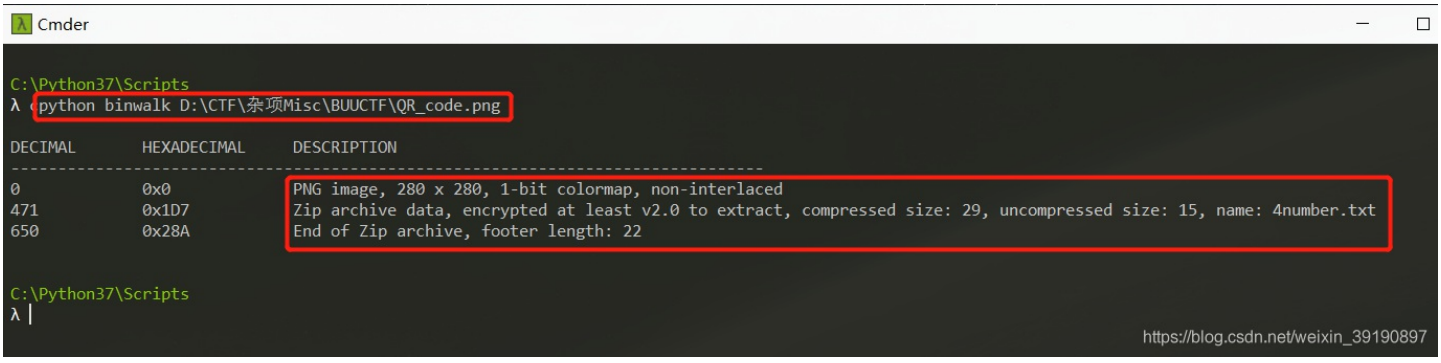


```
running egg_info
creating src\binwalk.egg-info
writing src\binwalk.egg-info\PKG-INFO
writing dependency_links to src\binwalk.egg-info\dependency_links.txt
writing top-level names to src\binwalk.egg-info\top_level.txt
writing manifest file 'src\binwalk.egg-info\SOURCES.txt'
reading manifest file 'src\binwalk.egg-info\SOURCES.txt'
```

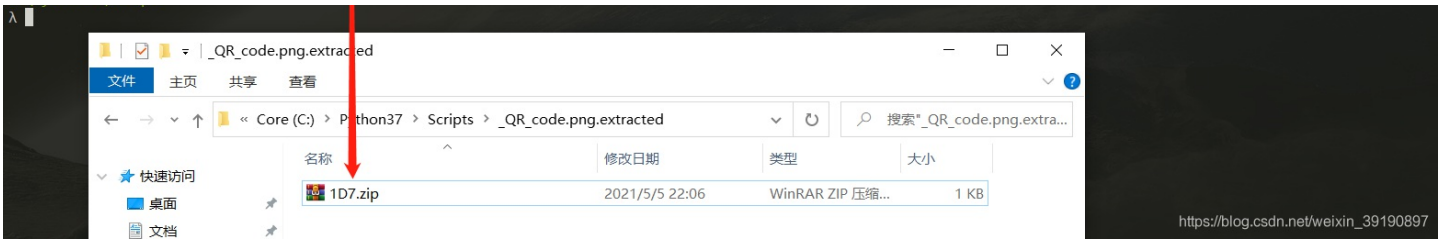

安装完成后在 python 安转目录下的 Script 目录里有个名字叫做 binwalk 的文件，然后在该文件夹启动命令行就可以使用 Binwalk 了：



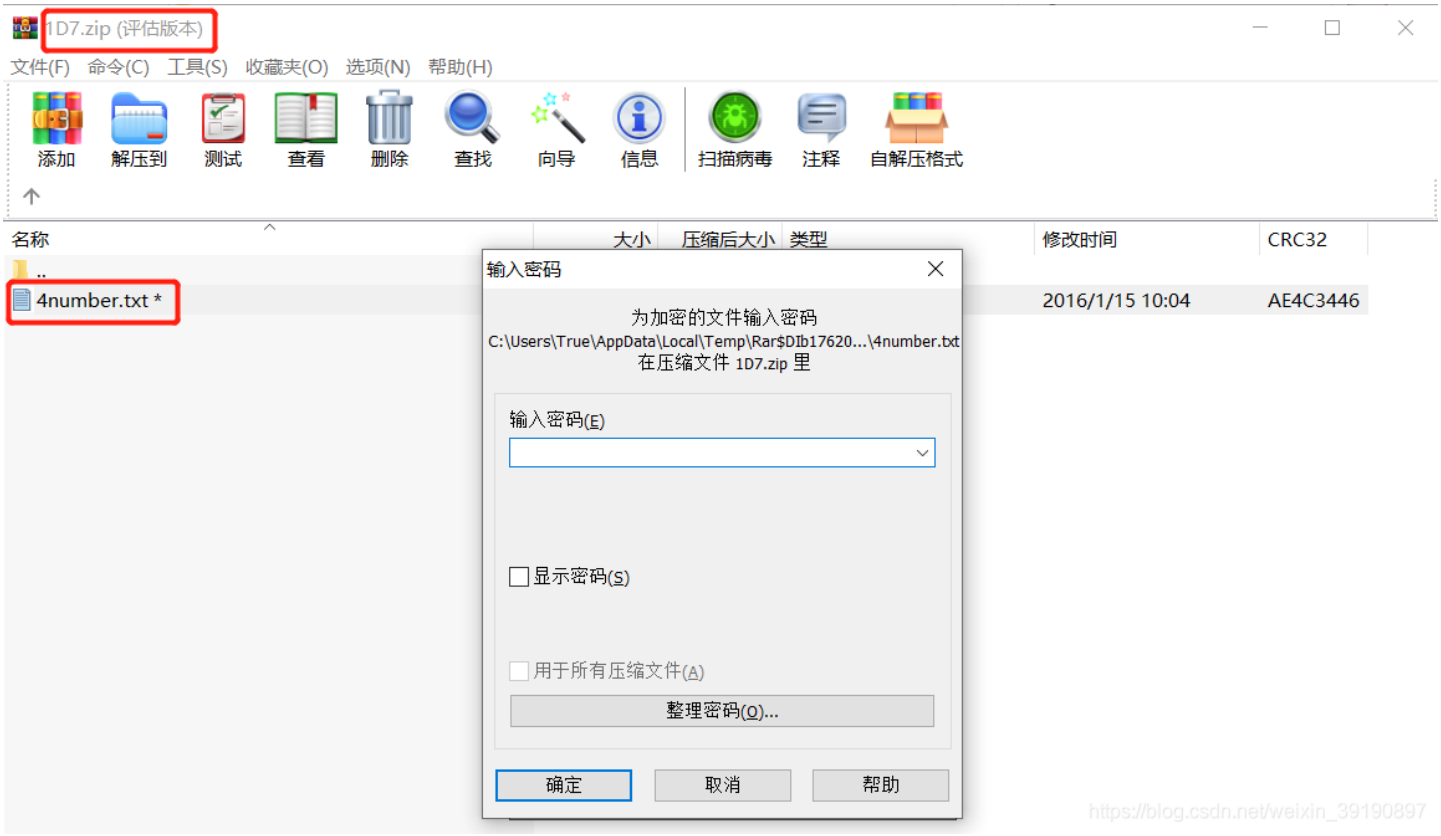
3、使用 Binwalk 分析目标图片，可以看到隐藏着几个文件：



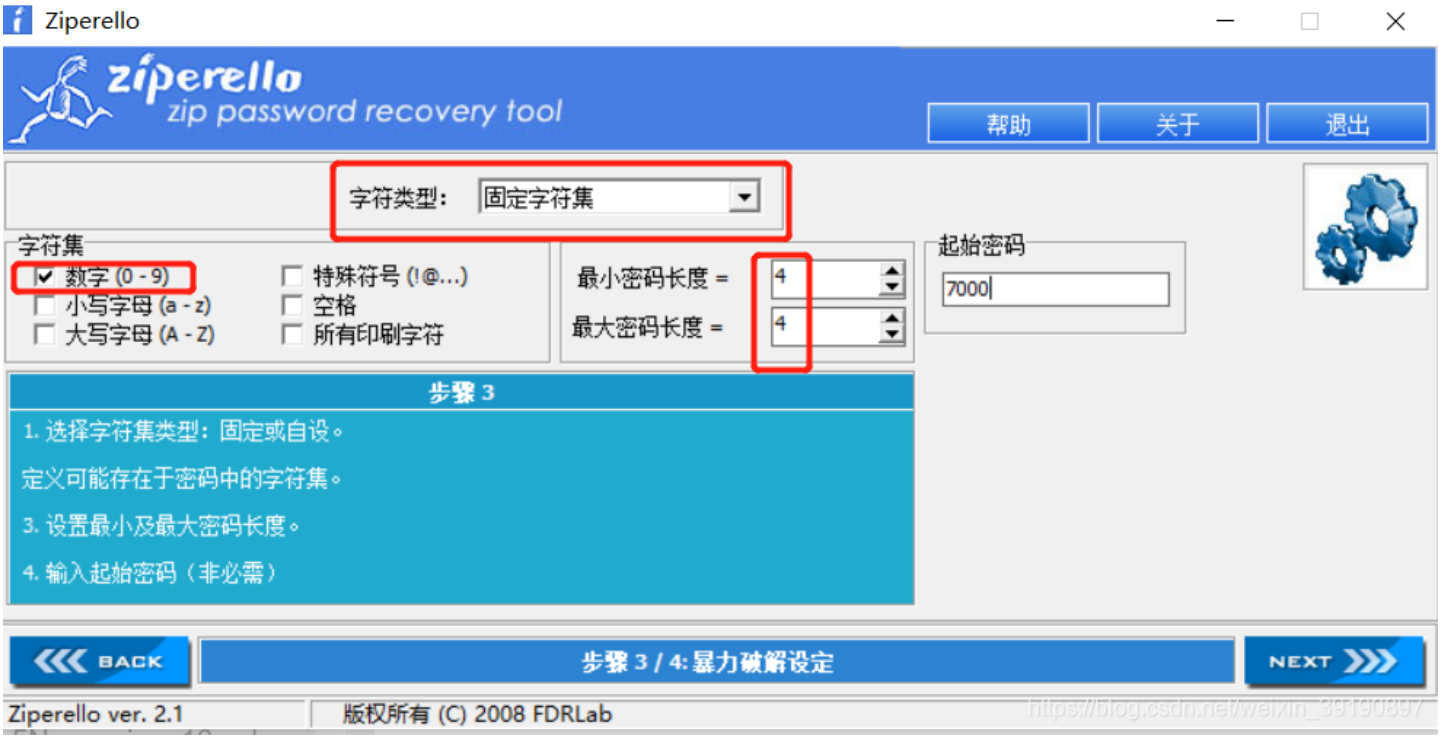
4、使用 Binwalk 进一步分离出二维码图片中隐藏的文件：



5、打开分离出来的压缩文件，提示需要进行密码才能查看文件：



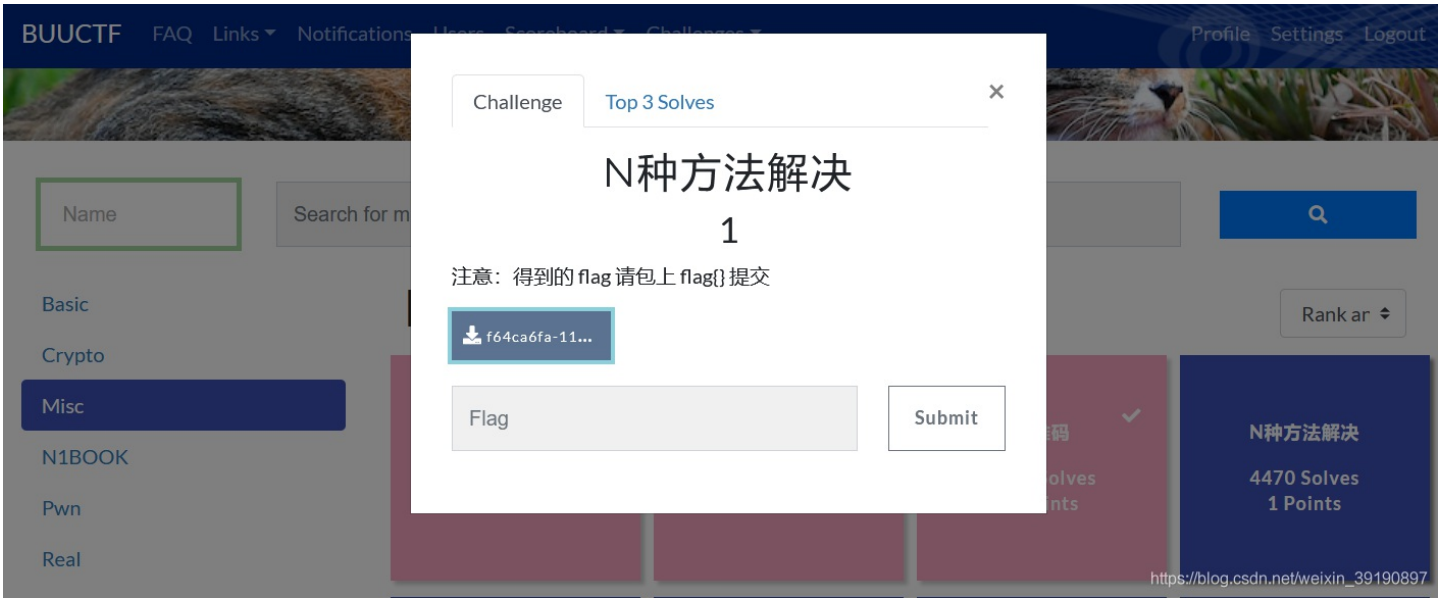
6、根据提示是 4 位数的密码，直接使用 Ziperello 工具对 ZIP 文件进行暴力破解（下载链接：https://pan.baidu.com/s/17TxQVXE8oZZA_dsyJglaqQ；提取码：2333）：


Ziperello
ziperello
zip password recovery tool
帮助
关于
退出
字符类型:
固定字符集
字符集
数字 (0 - 9)
小写字母 (a - z)
大写字母 (A - Z)
特殊符号 (!@...)
空格
所有印刷字符
最小密码长度 =
4
最大密码长度 =
4
起始密码
7000
步骤 3
1. 选择字符集类型：固定或自设。
定义可能存在于密码中的字符集。
3. 设置最小及最大密码长度。
4. 输入起始密码（非必需）
BACK
步骤 3 / 4: 暴力破解设定
NEXT
Ziperello ver. 2.1
版权所有 (C) 2008 FDRLab

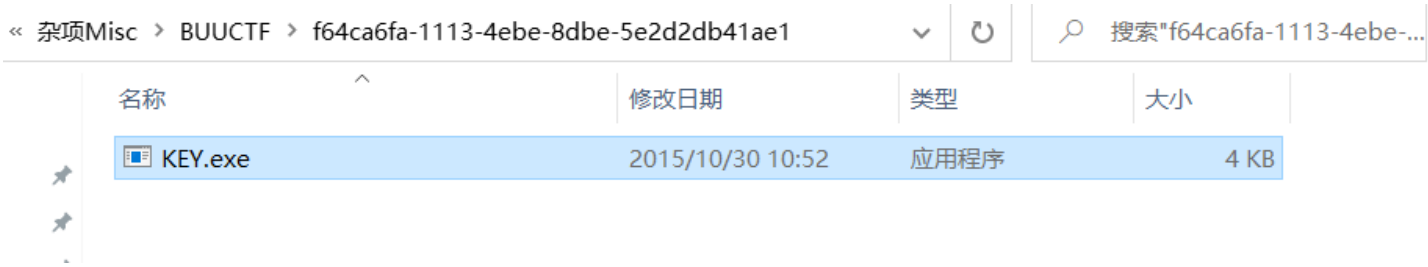
获得密码为 7639：

7、打开加密文件，获得 Flag：

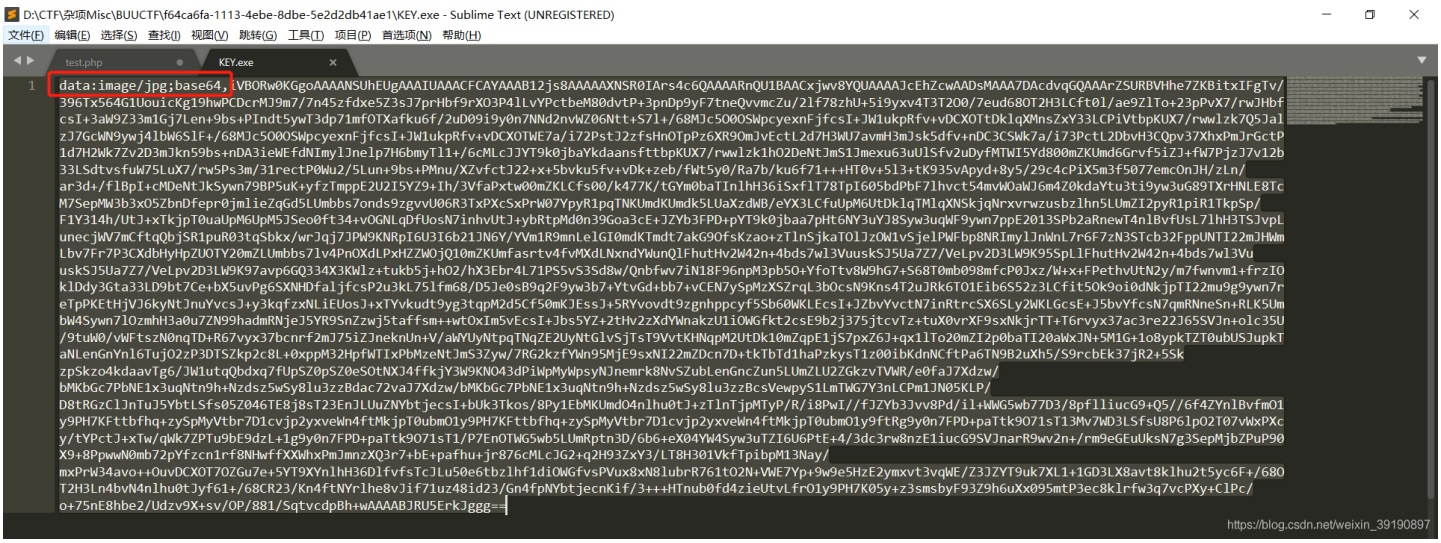
N0.3 Base64编码还原图片



1、下载后是一个 KEY.exe 文件：

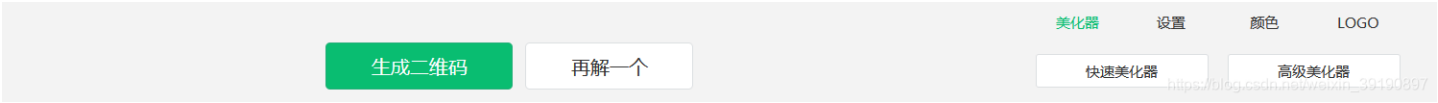


2、使用 Sublime 打开，发现是图片转换成 Baase 64编码：

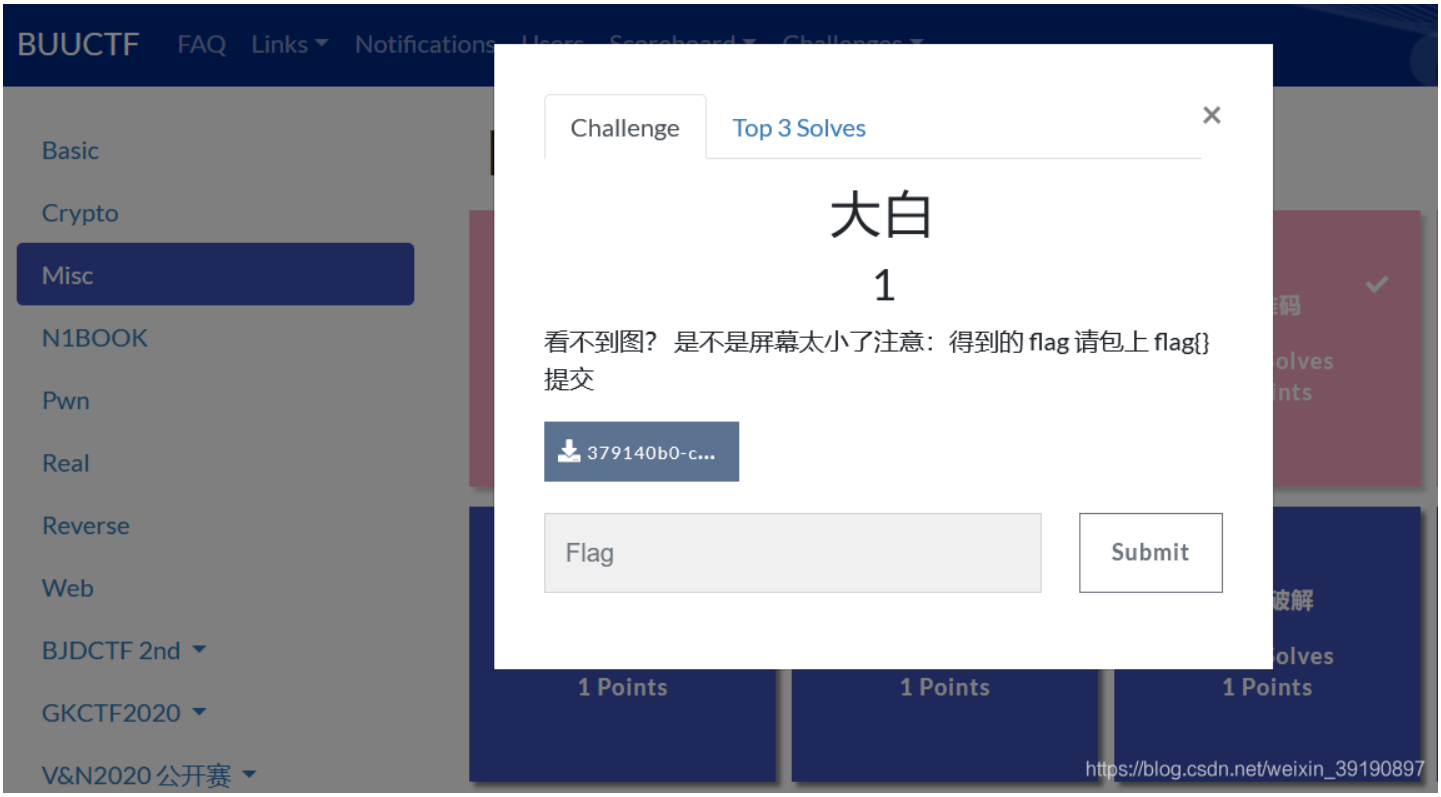


3、使用站长之家在线转换该 base 64 编码并还原成图片：

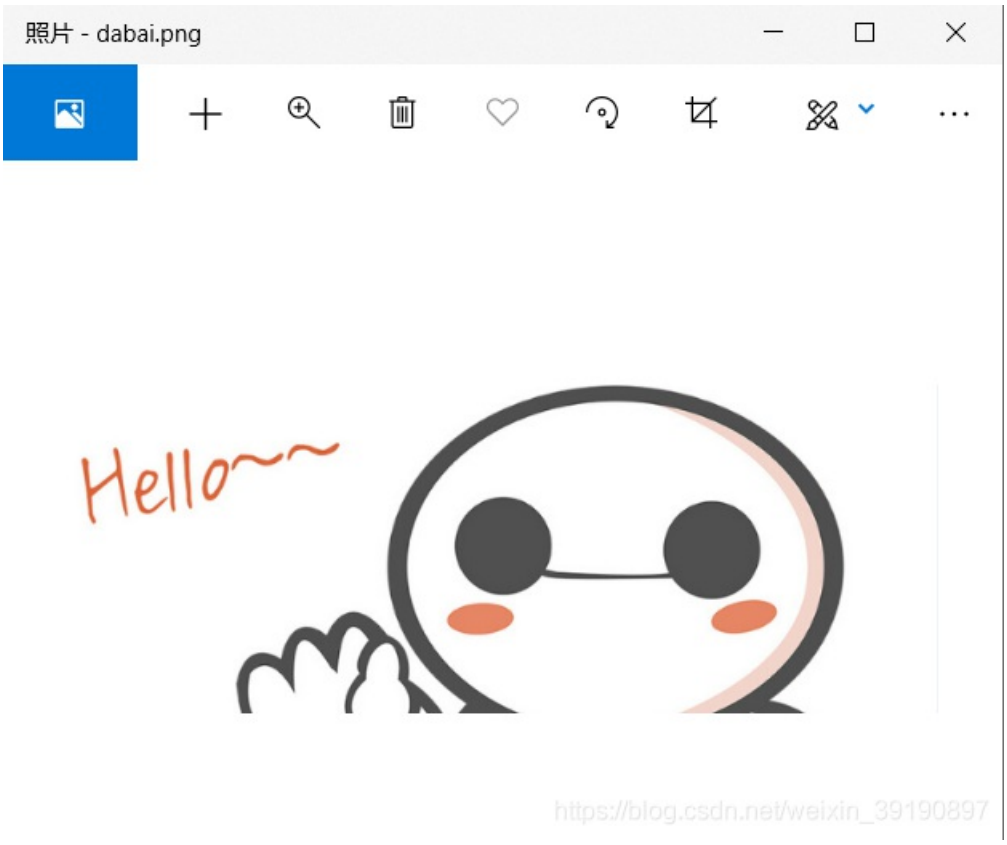
4、接着使用微信扫描二维码或者在线识别二维码图片，可获得 flag：



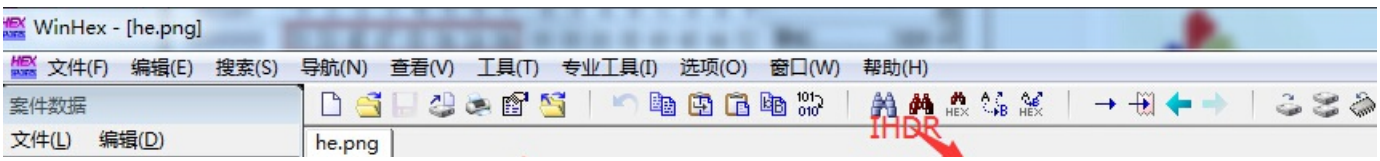
No.4 winhex修改图片大小

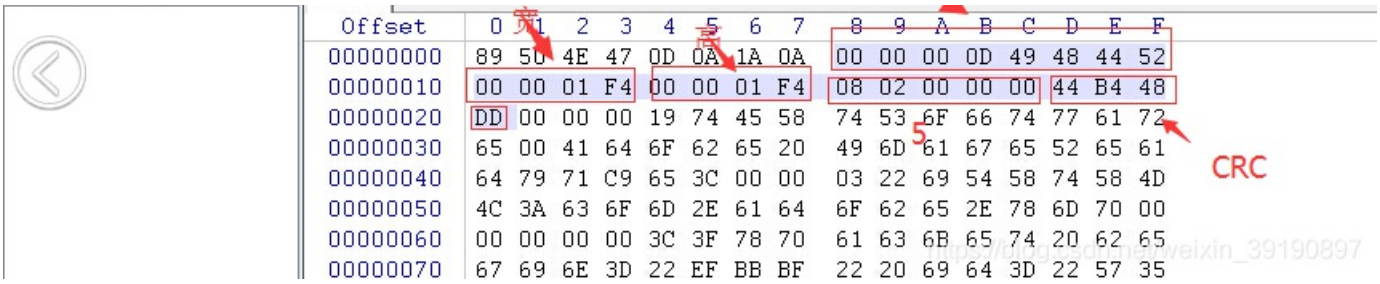


1、下载后打开图片如下：



2、根据题目提示，需要将图片变大，可使用 winhex 打开图片并修改图片的宽、高。先来介绍下相关知识：





5.2.3 分析PNG图像文件结构（2）

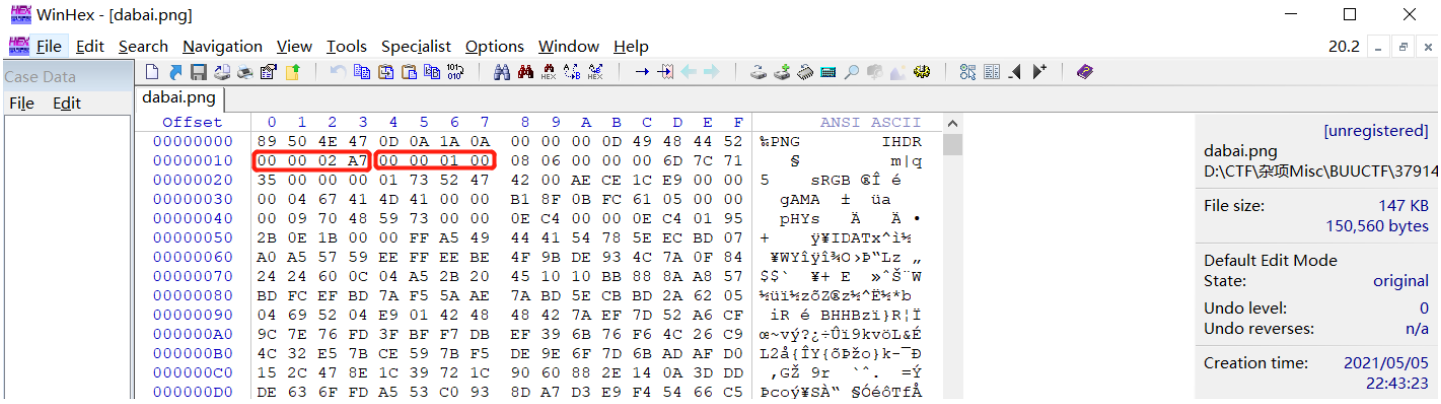
表5-15归纳了pic1.png图像文件中文件头数据块（IHDR）中各字段的含义。由于PNG文件使用Big-Endian顺序存储数据，所以不需要反转字节数据理解。

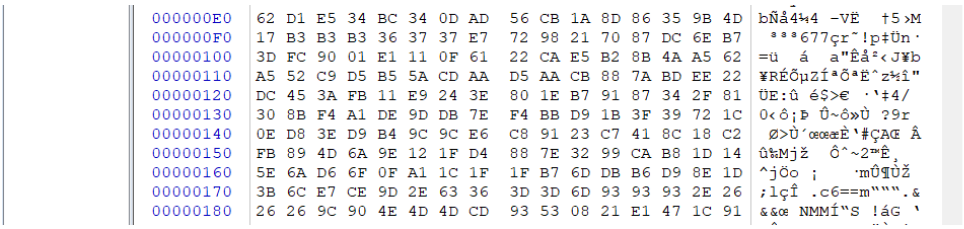
表5-15 pic1.png图像文件中文件头数据块(IHDR)的各字段含义

十六进制值	描 述
00 00 00 0D	文件头的数据长度，00 00 00 0D =13
49 48 44 52	数据块类型标志，49 48 44 52的ASCII值等于IHDR
00 00 00 C8	图像的宽度，00 00 00 C8 = 200
00 00 00 96	图像的高度，00 00 00 96 = 150
08	色深，表示2的8次幂等于256色
03	03表示索引图像
00	00表示使用Deflate压缩编码压缩图像数据
00	00表示为将来使用更好的压缩方法预留
00	00表示非隔行扫描
AC 02 37 2B	AC 02 37 2B表示CRC

从表5-14看到pic1.png文件的文件头数据块（IHDR）结构中的CRC字段的值为AC 02 37 2B，这个CRC值是按照从数据块类型标志字段到CRC字段前一字节的数据计算而来的，即使用数据49 48 44 52 00 00 00 C8 00 00 00 96 08 03 00 00 00计算，CRC的计算代码如下：

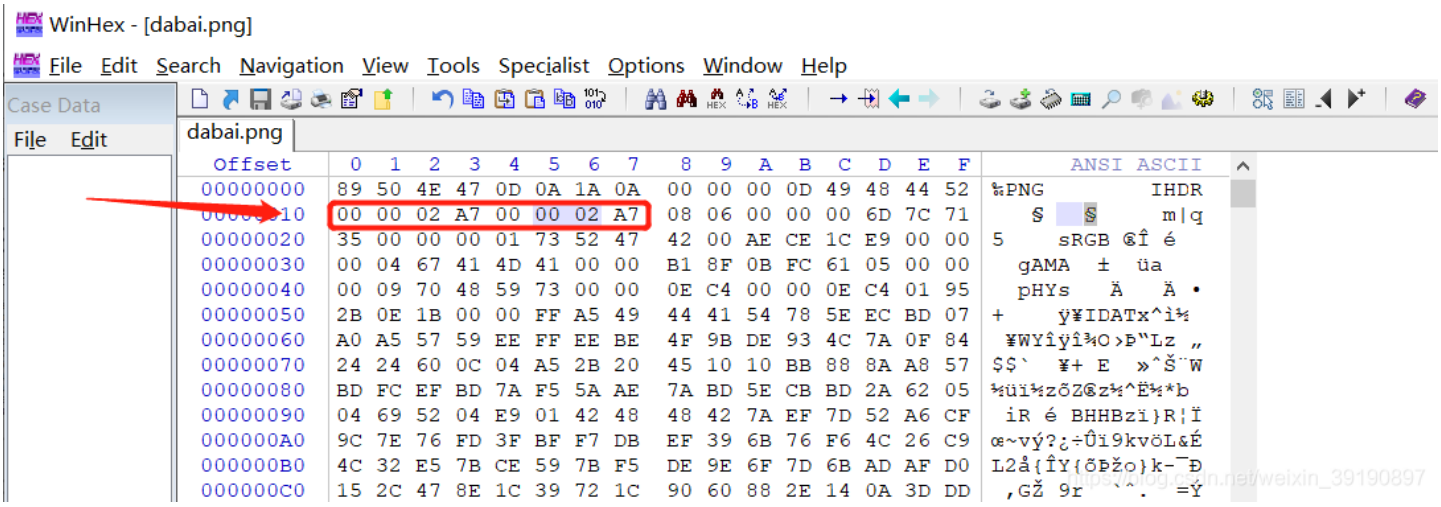
3、故使用 winhex 工具打开图片，找到宽、高位置，修改其值：



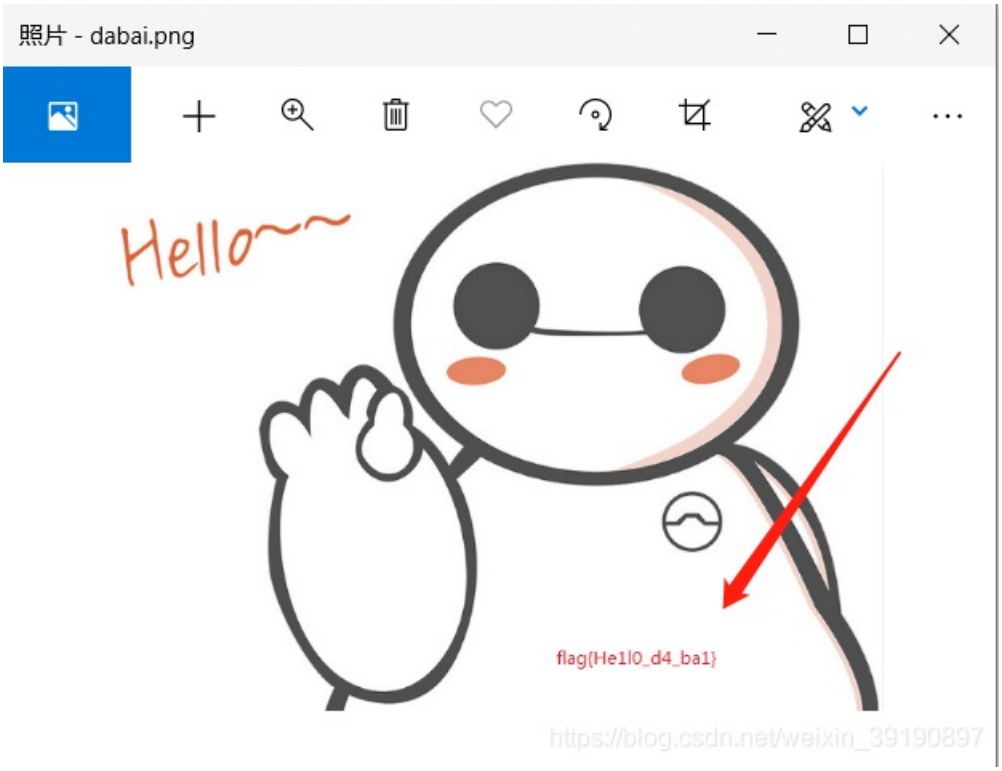


000000E0	62	D1	E5	34	BC	34	0D	AD	56	CB	1A	8D	86	35	9B	4D
000000F0	17	B3	B3	B3	36	37	37	E7	72	98	21	70	87	DC	6E	B7
00000100	3D	FC	90	01	E1	11	0F	61	22	CA	E5	B2	8B	4A	A5	62
00000110	A5	52	C9	D5	B5	5A	CD	AA	D5	AA	CB	88	7A	BD	EE	22
00000120	DC	45	3A	FB	11	E9	24	3E	80	1E	B7	91	87	34	2F	81
00000130	30	8B	F4	A1	DE	9D	DB	7E	F4	BB	D9	1B	3F	39	72	1C
00000140	0E	D8	3E	D9	B4	9C	9C	E6	C8	91	23	C7	41	8C	18	C2
00000150	FB	89	4D	6A	9E	12	1F	D4	88	7E	32	99	CA	B8	1D	14
00000160	5E	6A	D6	6F	0F	A1	1C	1F	1F	B7	6D	DB	B6	D9	8E	1D
00000170	3B	6C	E7	CE	9D	2E	63	36	3D	3D	6D	93	93	93	2E	26
00000180	26	26	9C	90	4E	4D	4D	CD	93	53	08	21	E1	47	1C	91

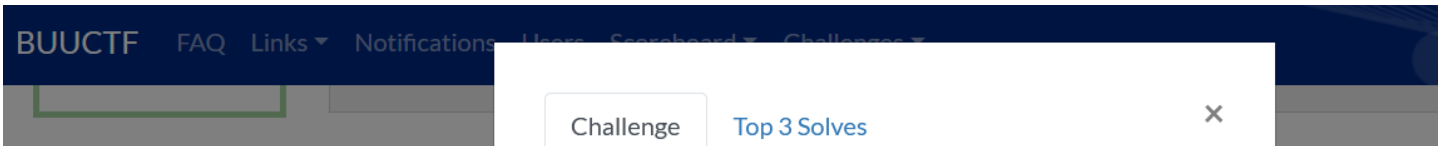
第 18、19 位是长，22、23是宽，我们把宽度设置大一点，就设置和长度一样好了：



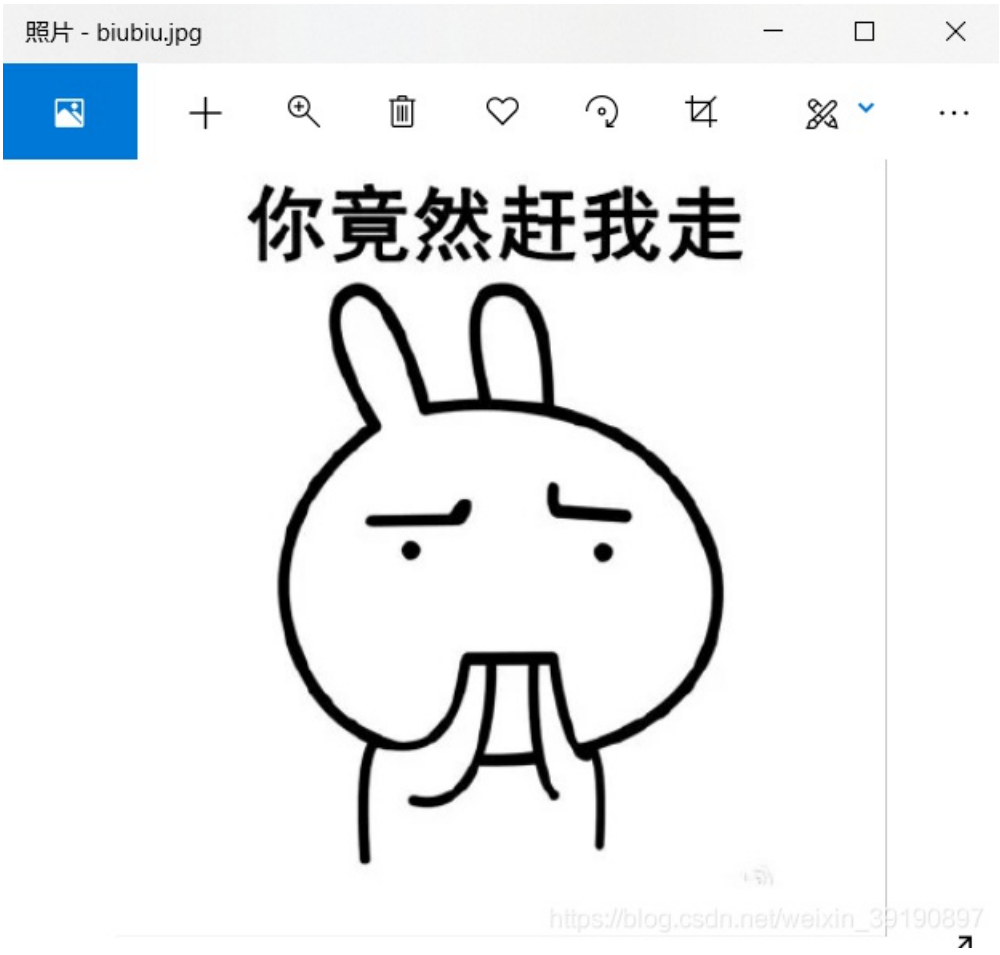
4、保存后重新打开图片，可获得 flag：



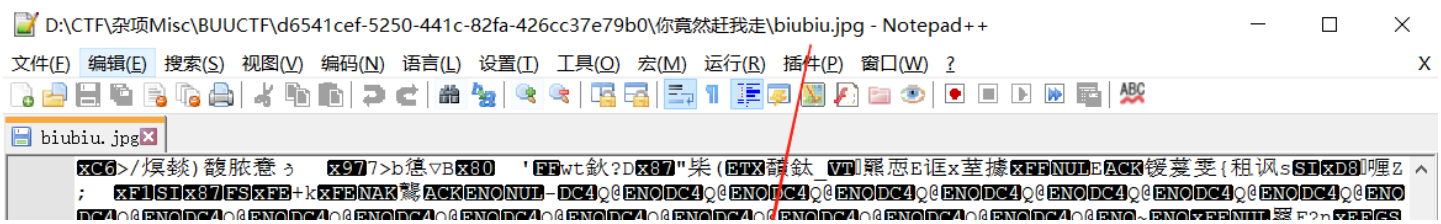
No.5 编辑器查看图片隐写

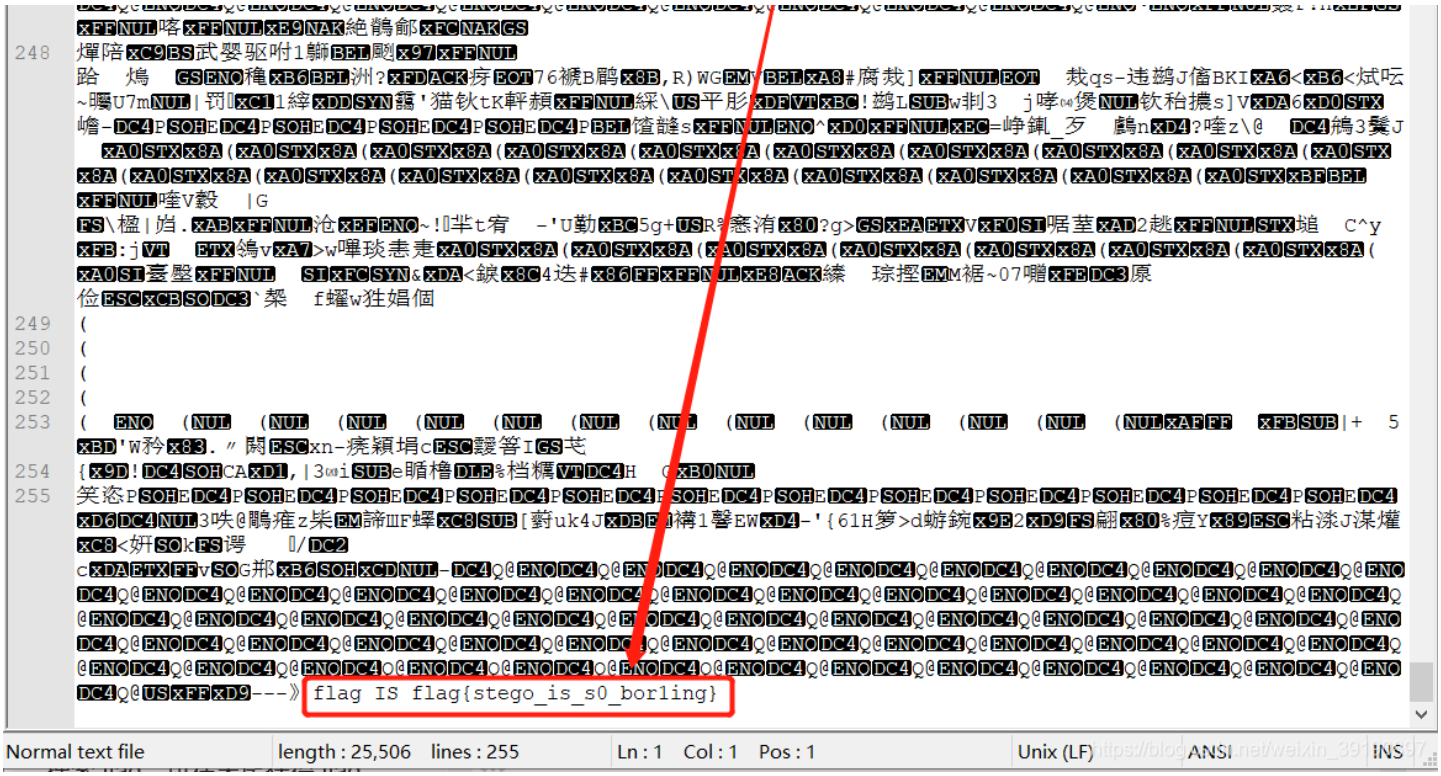


1、下载后打开是一张图片：

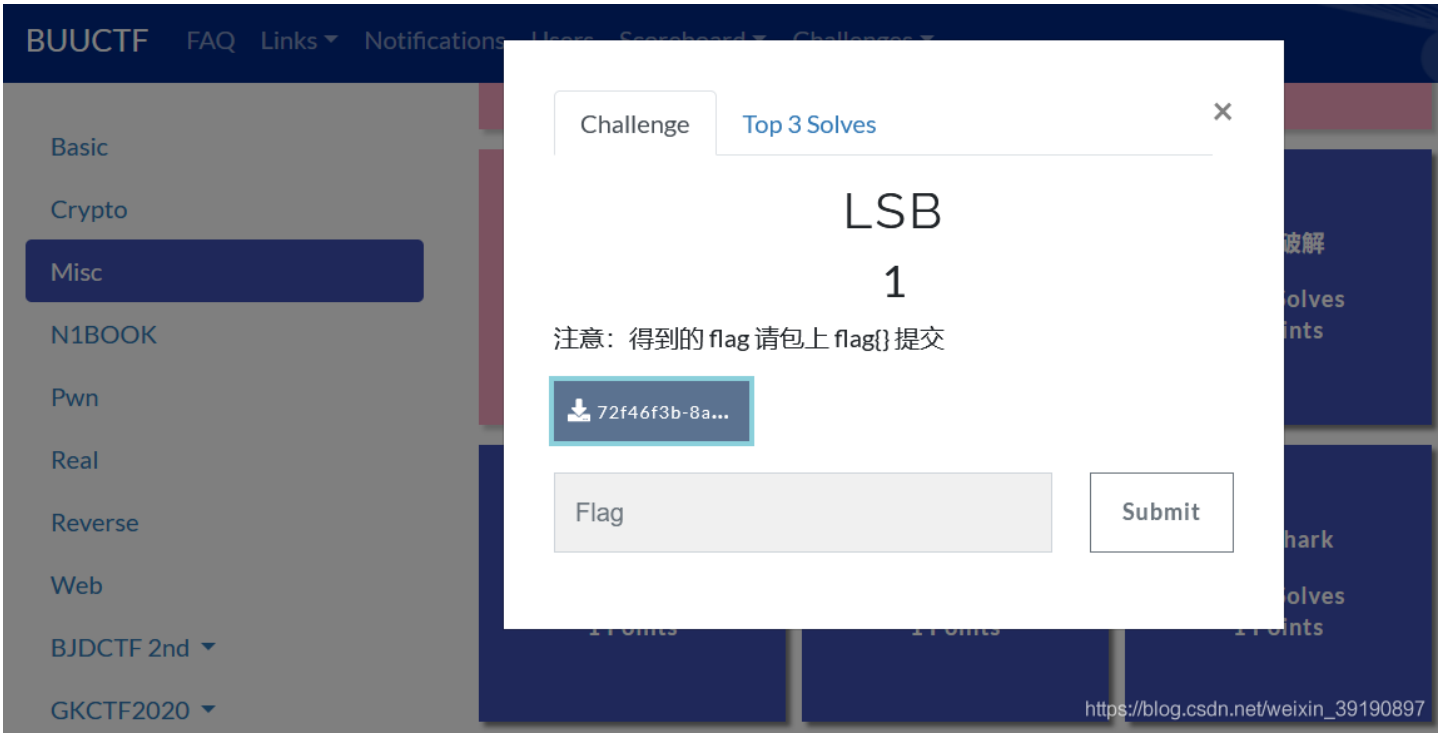


2、使用 Notepad 编辑器打开查看图片，搜索 flag，可在末尾获得 flag：

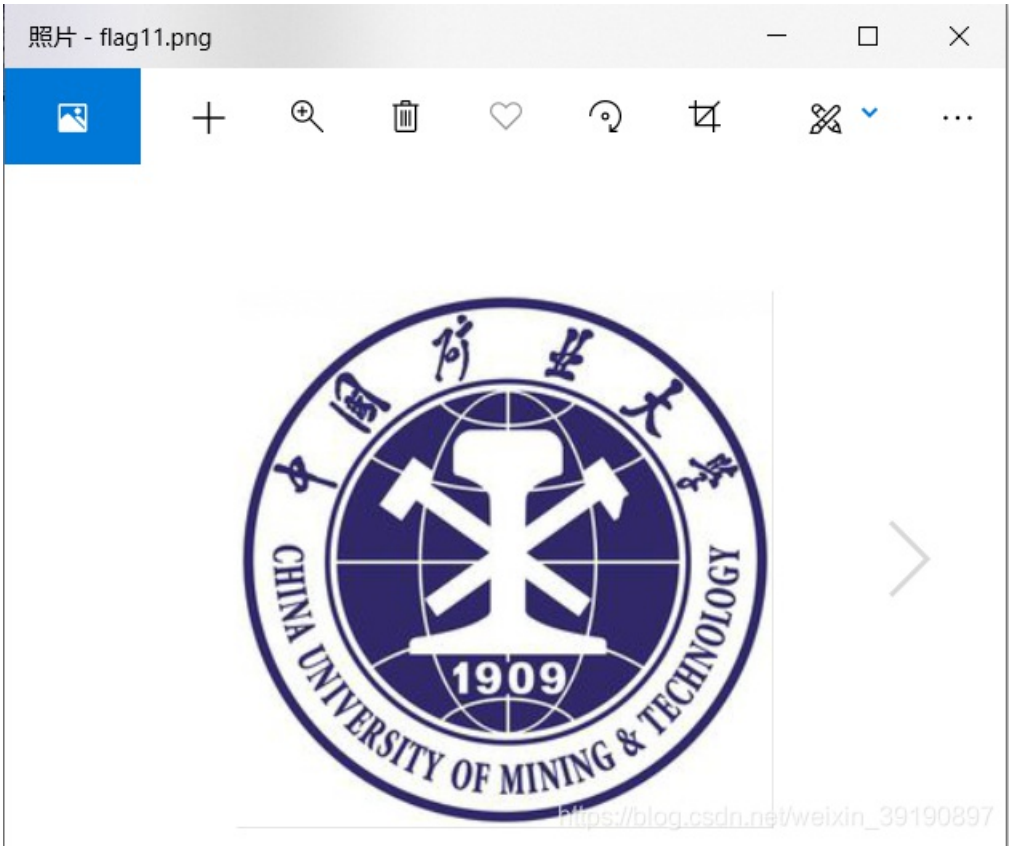




No.6 LSB隐写的信息导出



1、下载后是一张 png 图片：



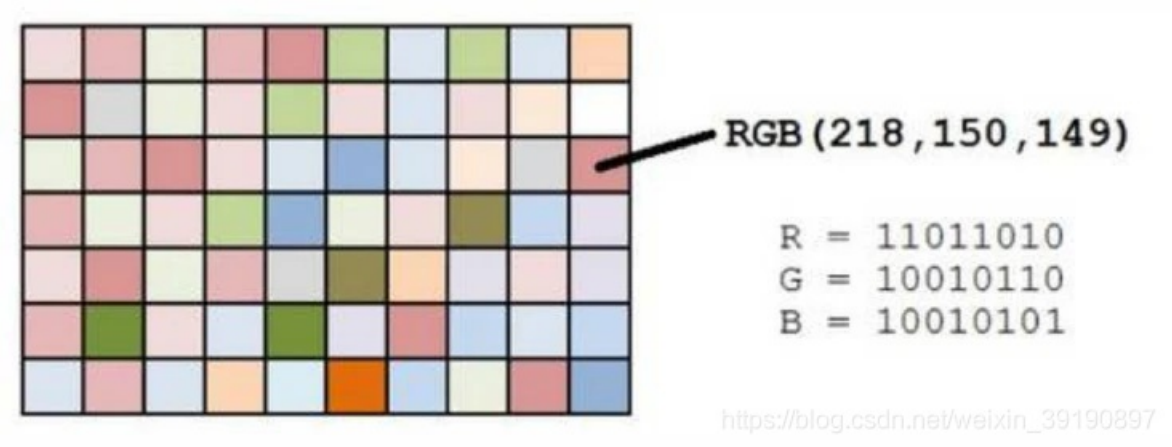
题目的提示很明显了，这是一道 LSB 隐写题。

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。lsb 隐写很实用，算法简单，能存储的信息量也大，是 CTF 比赛中的常客。

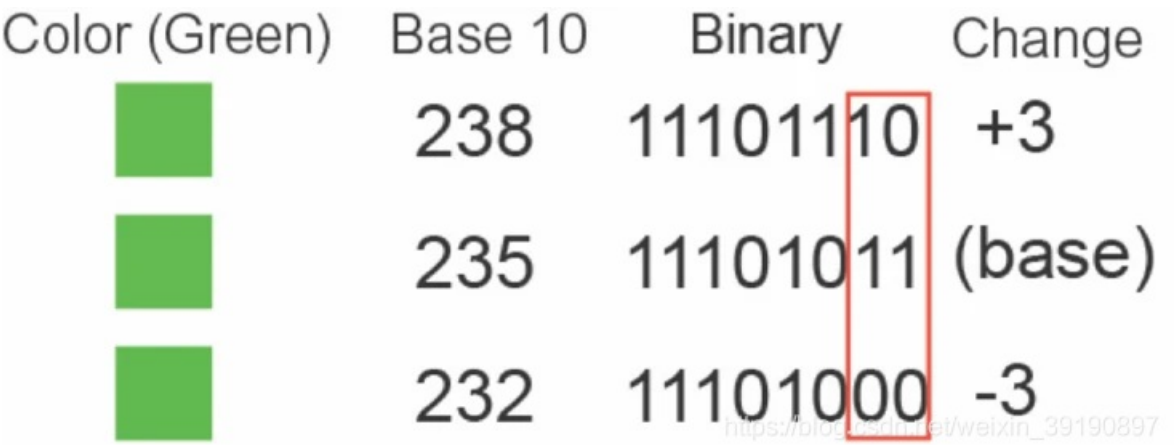
【原理介绍】

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。

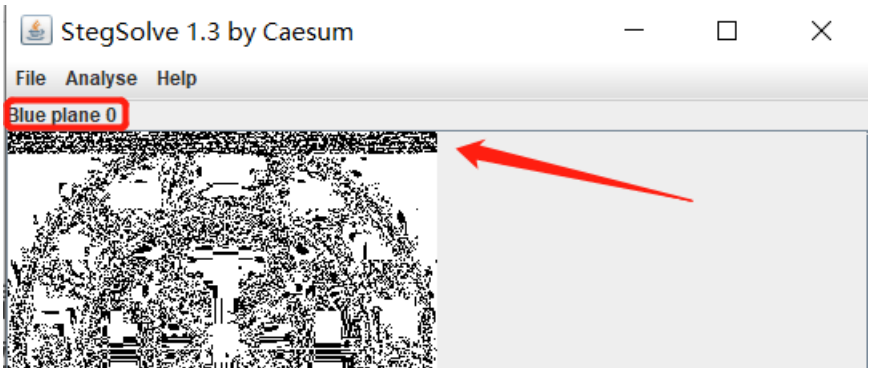


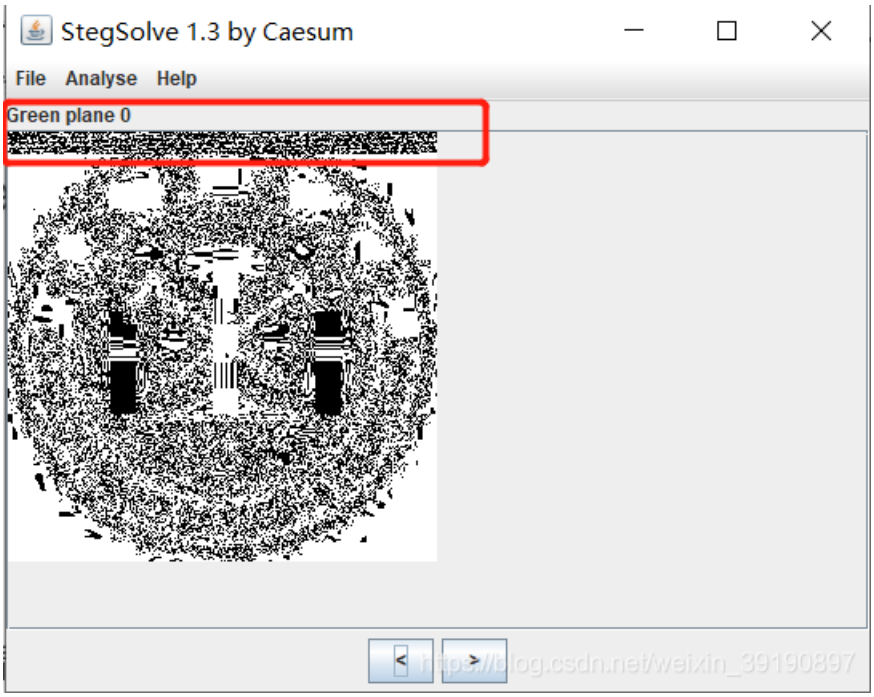
LSB 隐写就是修改 RGB 颜色分量的最低二进制位也就是最低有效位（LSB），而人类的眼睛不会注意到这前后的变化，每个像数可以携带3比特的信息。



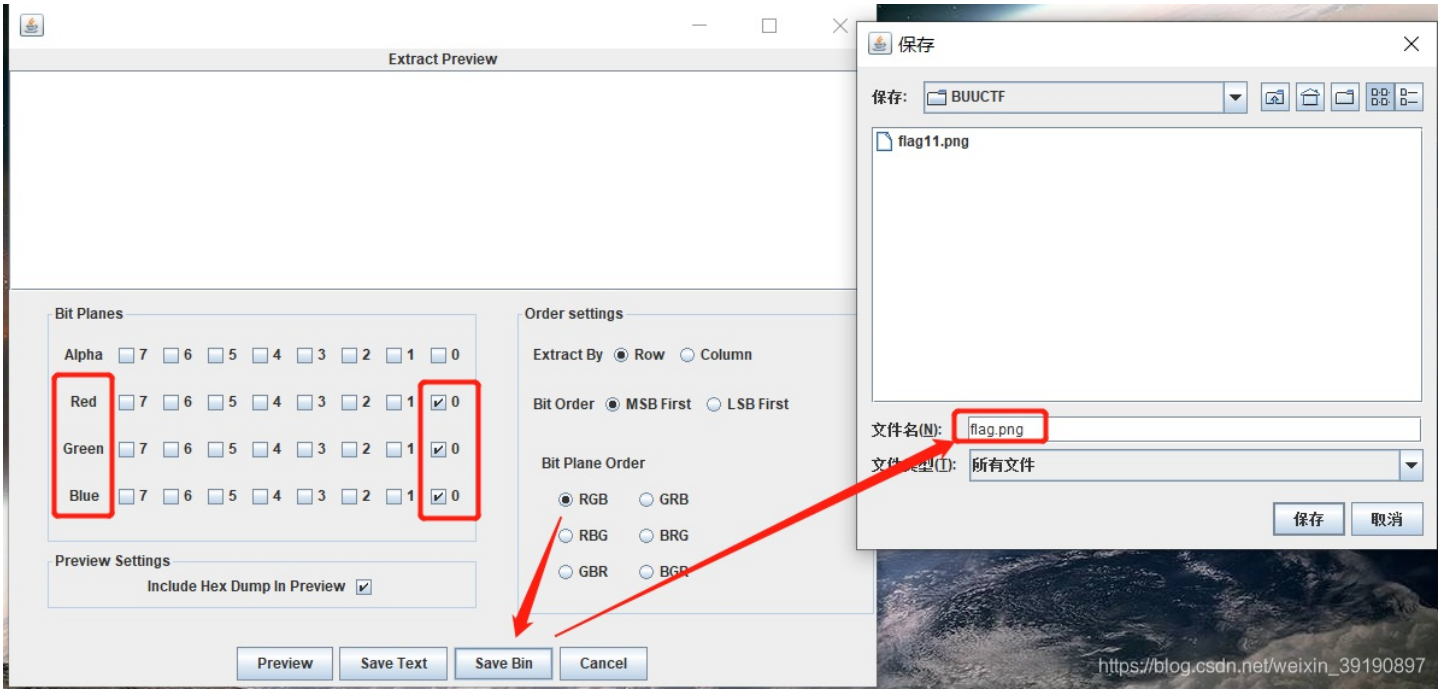
上图我们可以看到，十进制的 235 表示的是绿色，我们修改了在二进制中的最低位，但是颜色看起来依旧没有变化。我们就可以修改最低位中的信息，实现信息的隐写。

2、回到这道题目中来，针对 LSB 隐写，同样可以使用隐写图片查看的神器------**stegsolve**来分析，用 stegsolve 打开这张图片，发现 `red0, green0, blue0` 这三个通道的图片上方有异样：





3、因此用 Analyse 菜单栏里的 Data Extract 功能（数据抽取，图片中隐藏数据的抽取）查看图片，将三个位置的 0 通道勾选并导出保存为 png 格式的文件：

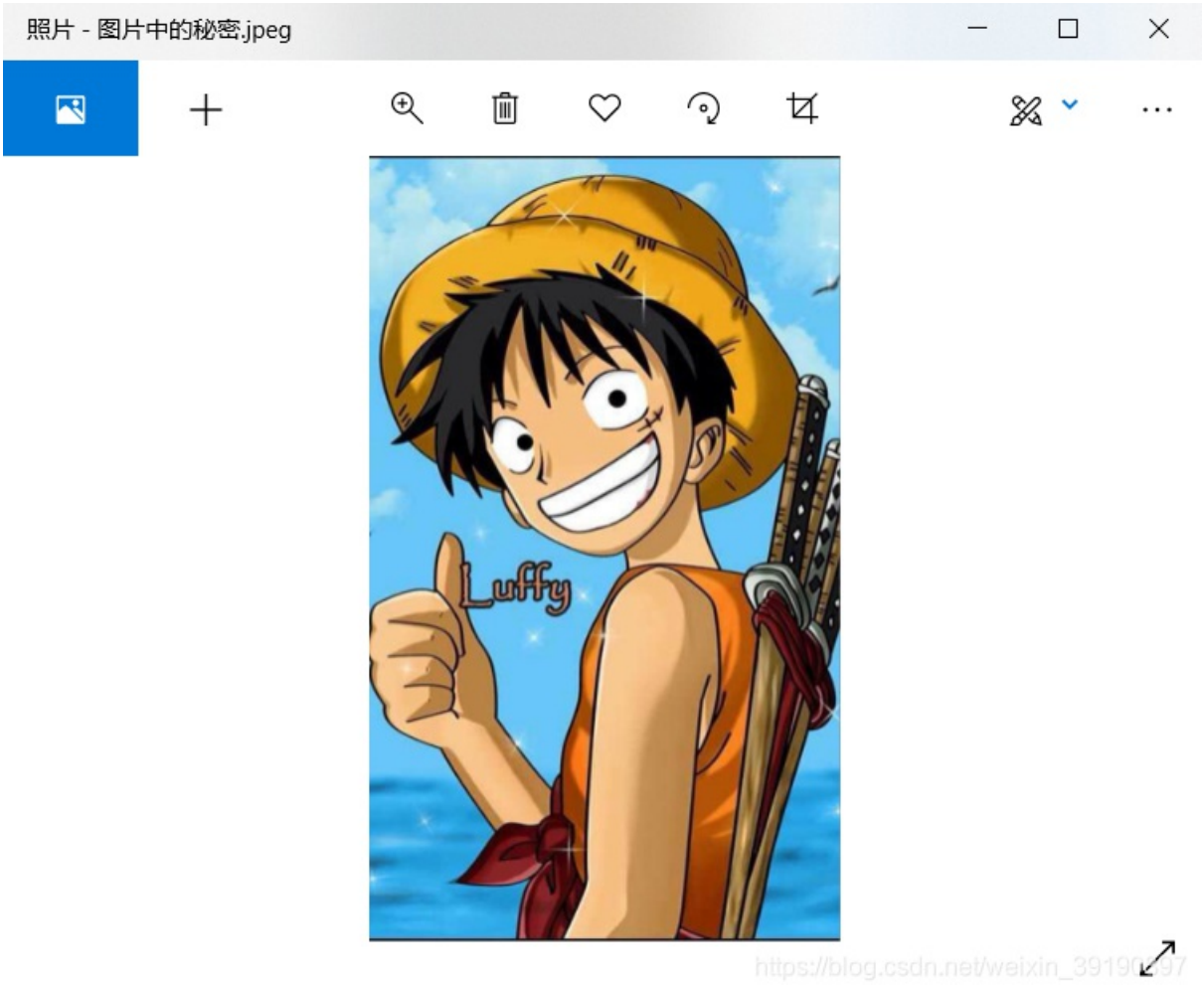


4、获得 flag.png 图片，发现是个二维码：

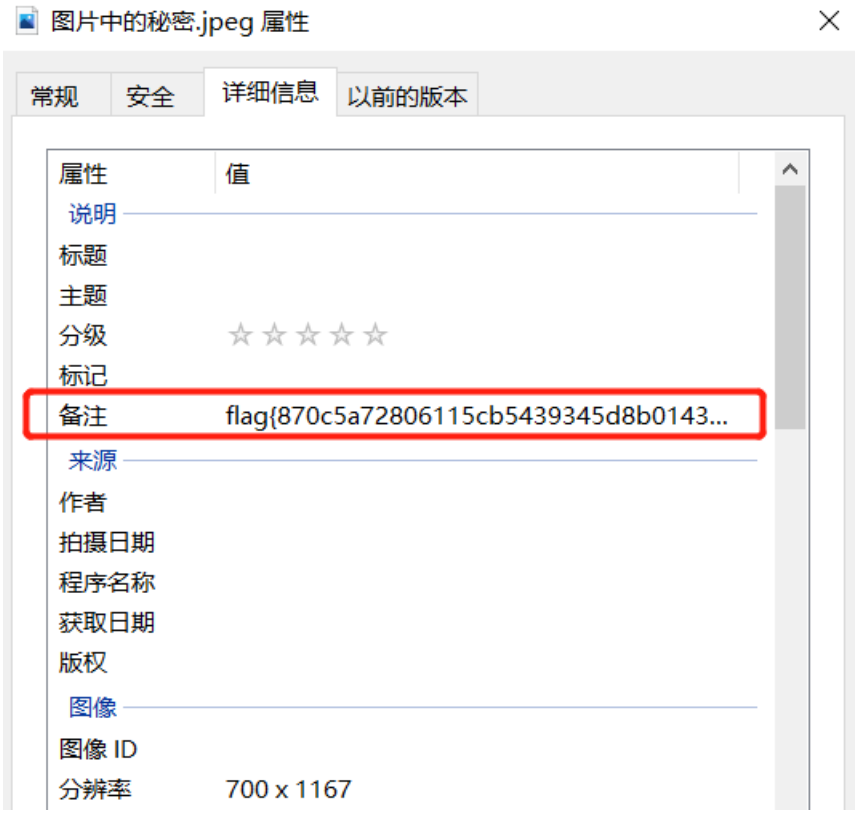
5、在线解码，获得 flag：

No.7 图片属性中隐藏信息

1、打开是一张路飞的图：



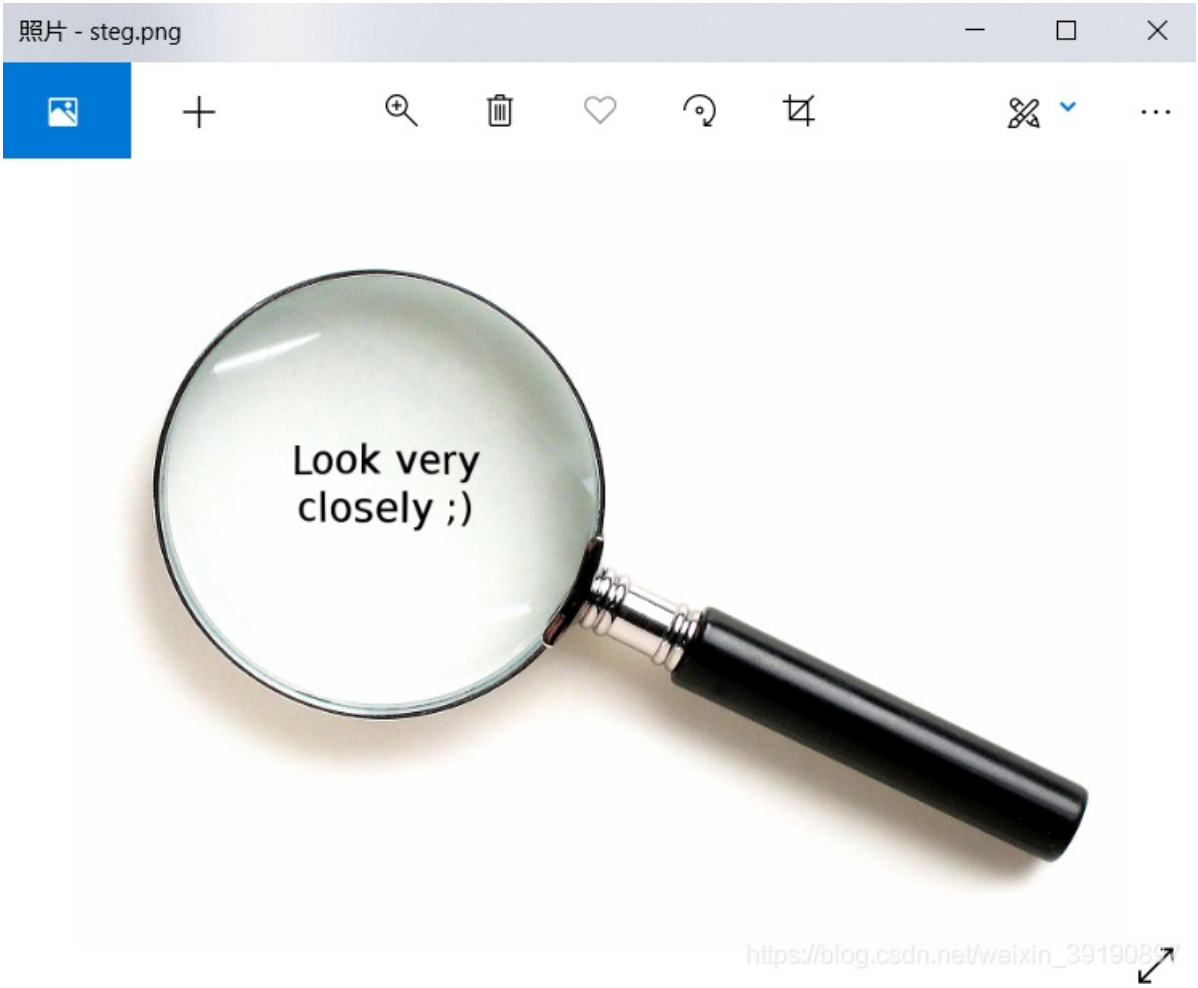
2、查看图片属性可直接获得 flag：



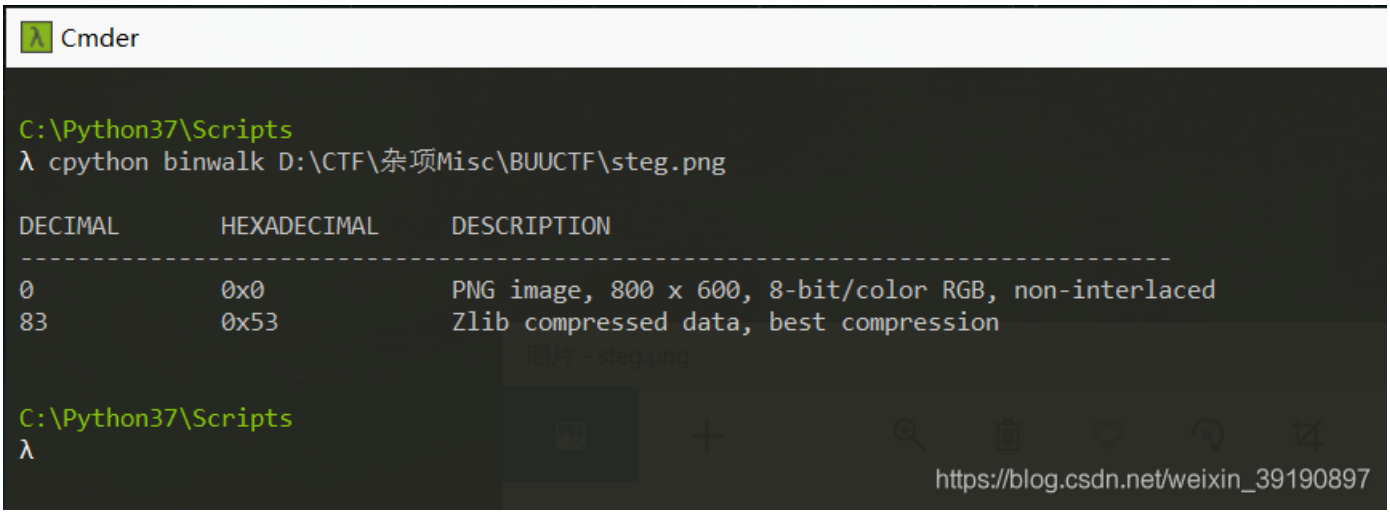
宽度	700 像素
高度	1167 像素

No.8 LSB隐写的数据抽取

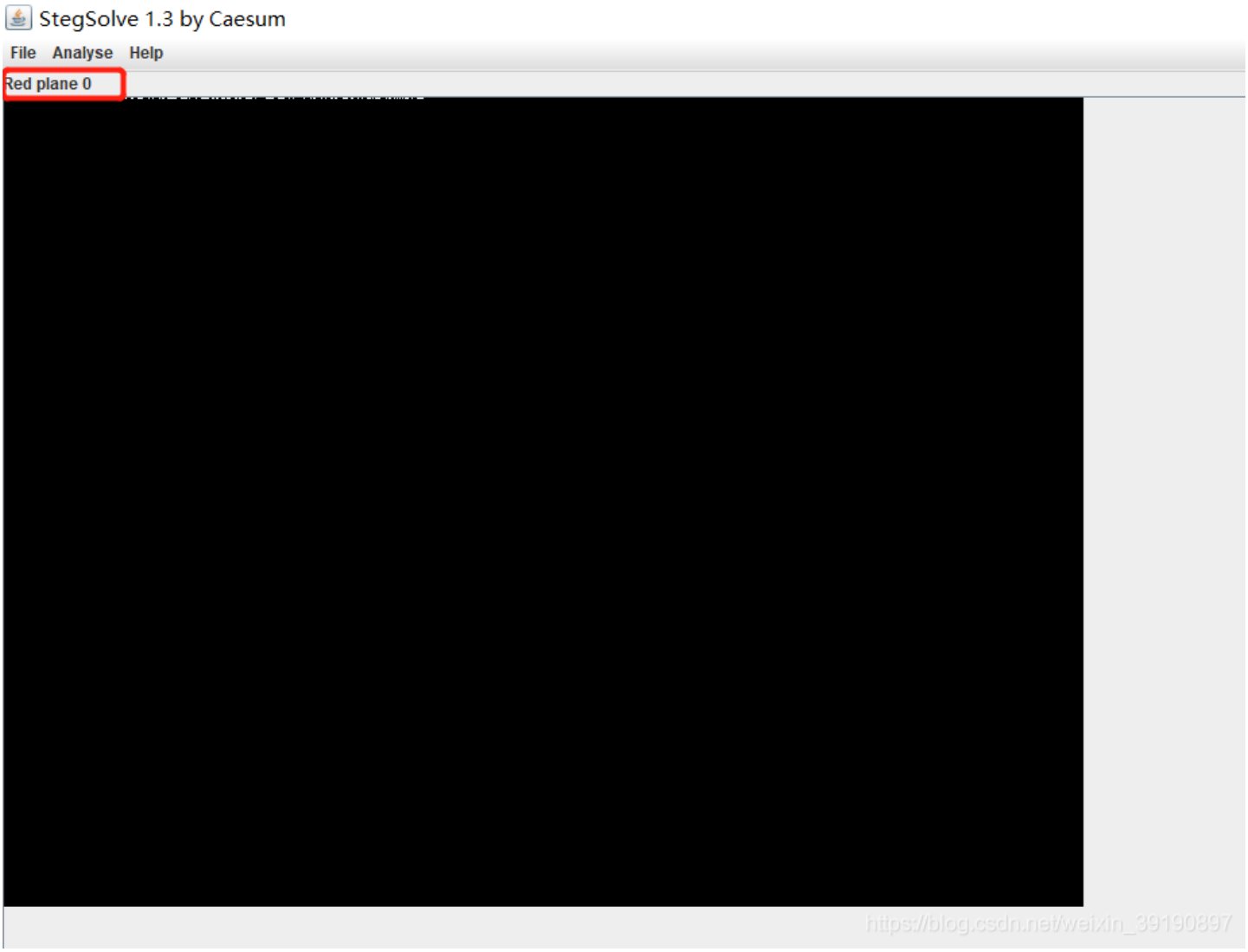
1、下载后打开是一张图片：



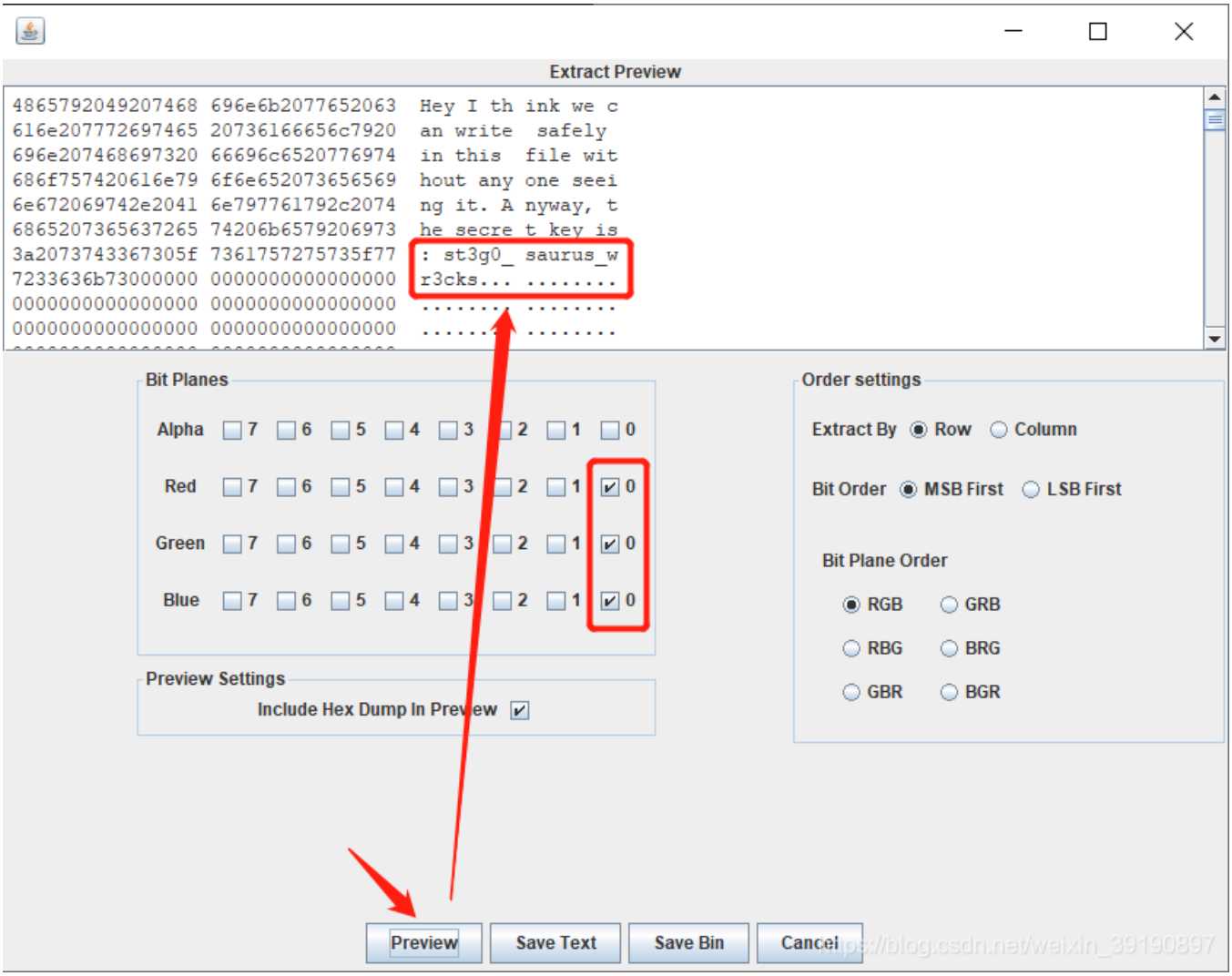
2、猜测是隐写，使用 binwalk，查看是否有隐藏文件，然而并没有：



3、使用 Stegsolve，查看是否存在 LSB 隐写，发现 `red0, green0, blue0` 这三个通道的图片均为异样的黑色：

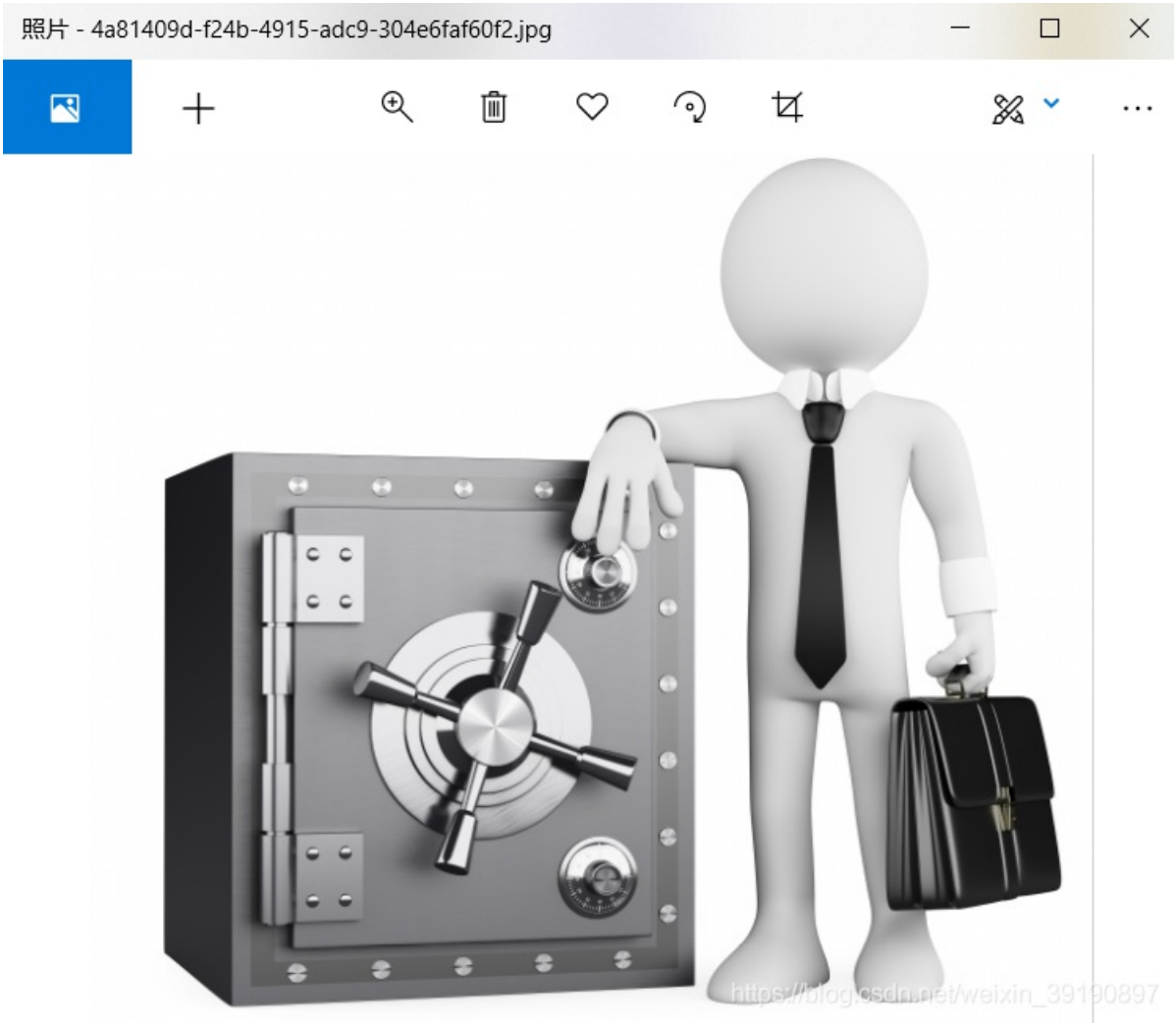


4、因此用 Analyse 菜单栏里的 Data Extract 功能（数据抽取，图片中隐藏数据的抽取）查看图片，将三个位置的 0 通道勾选并预览，发现 flag：

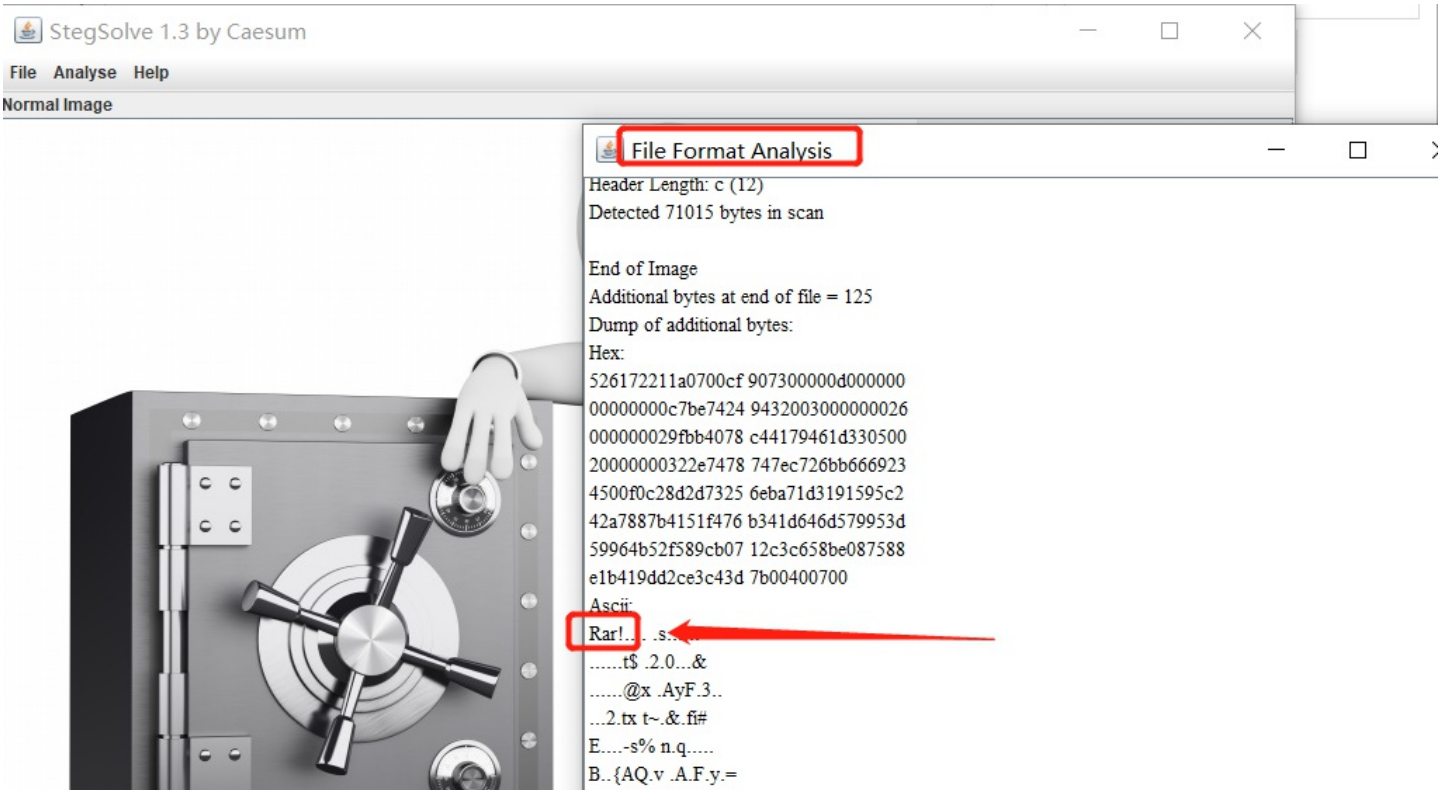


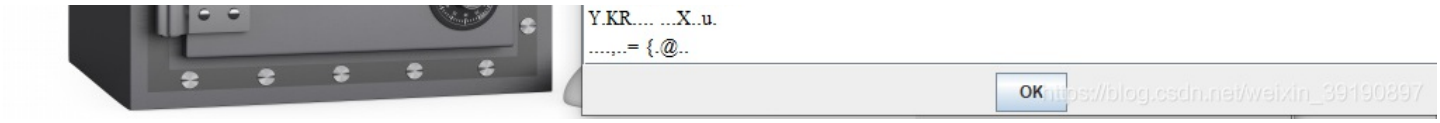
No.9 RAR文件加密的爆破

1、下载发现是一张图片：

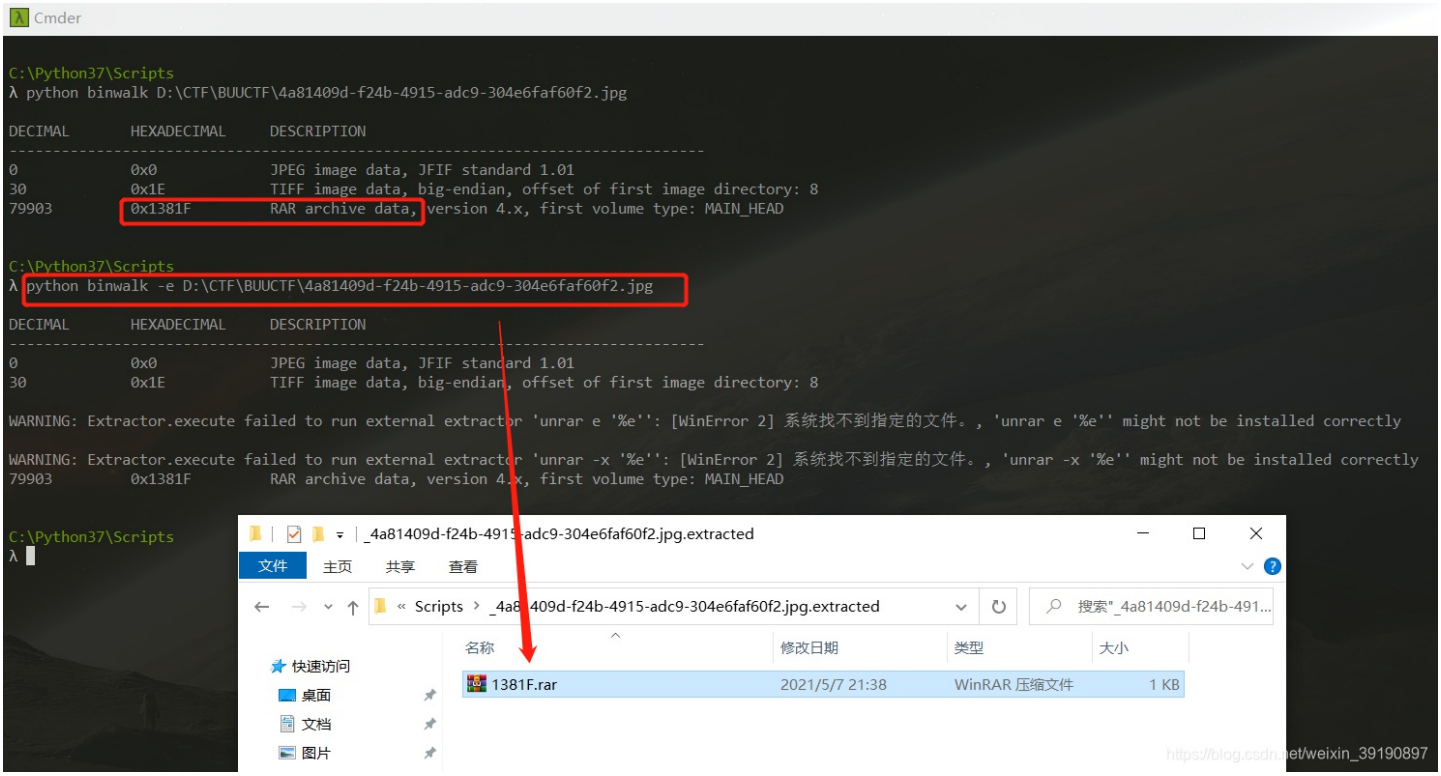


2、使用 **stegsolve** 的格式分析功能，发现 RAR 关键词：

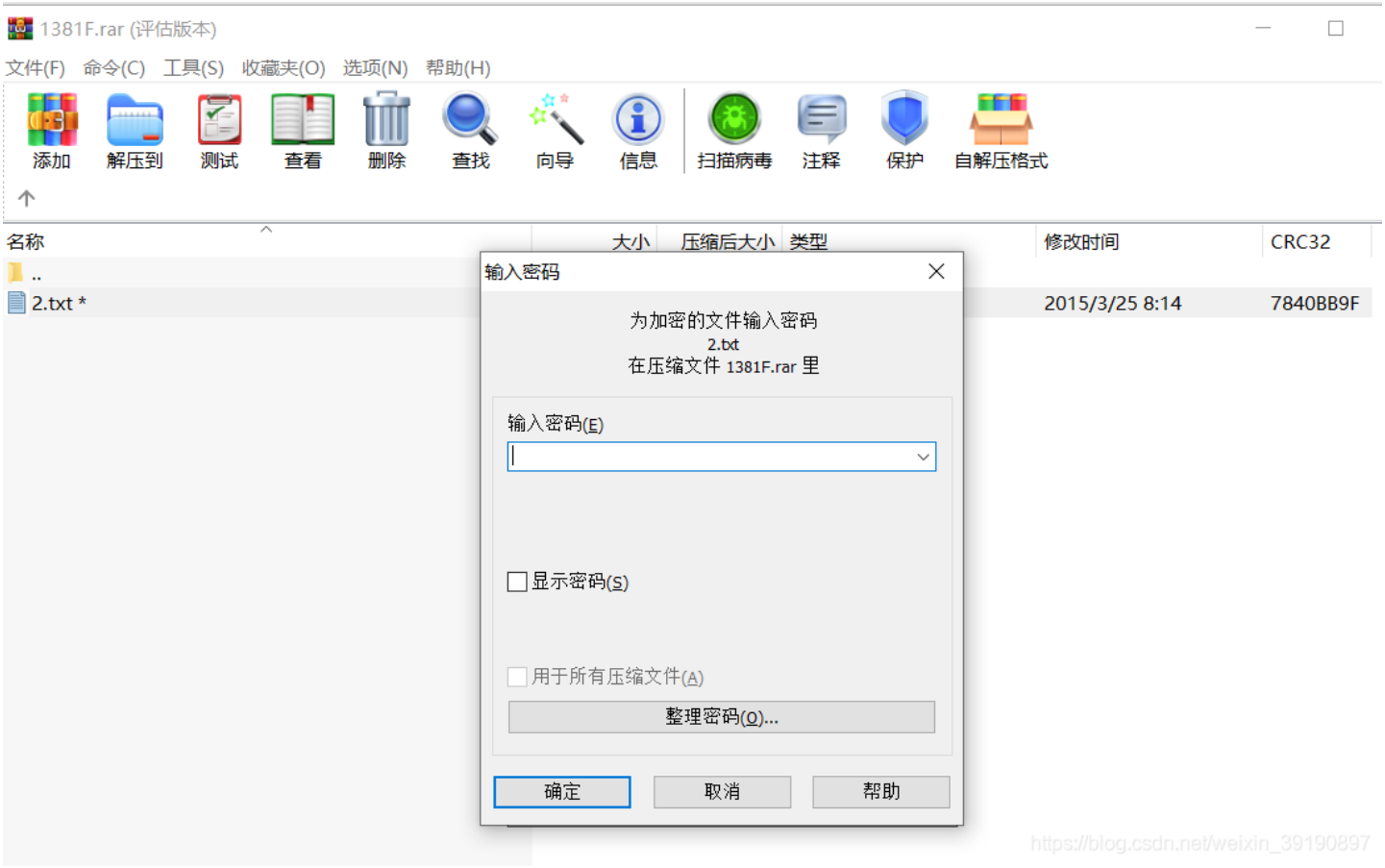




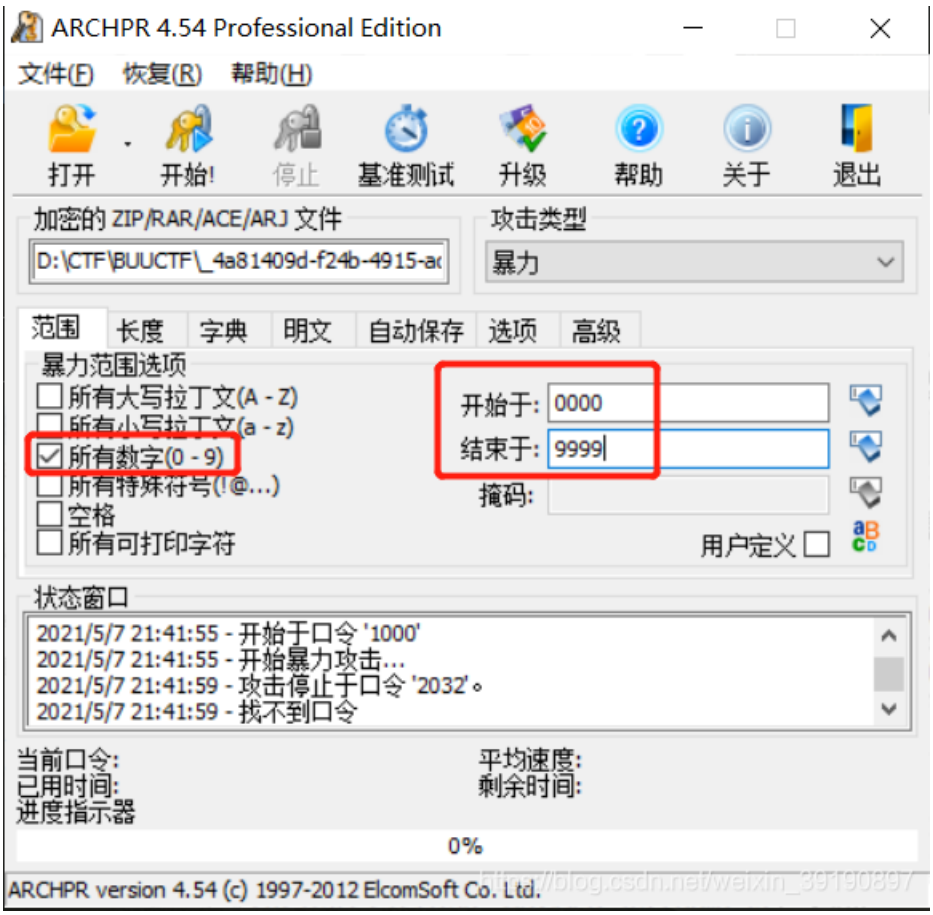
3、动用 Binwalk 工具分析，分离出 rar 压缩文件：



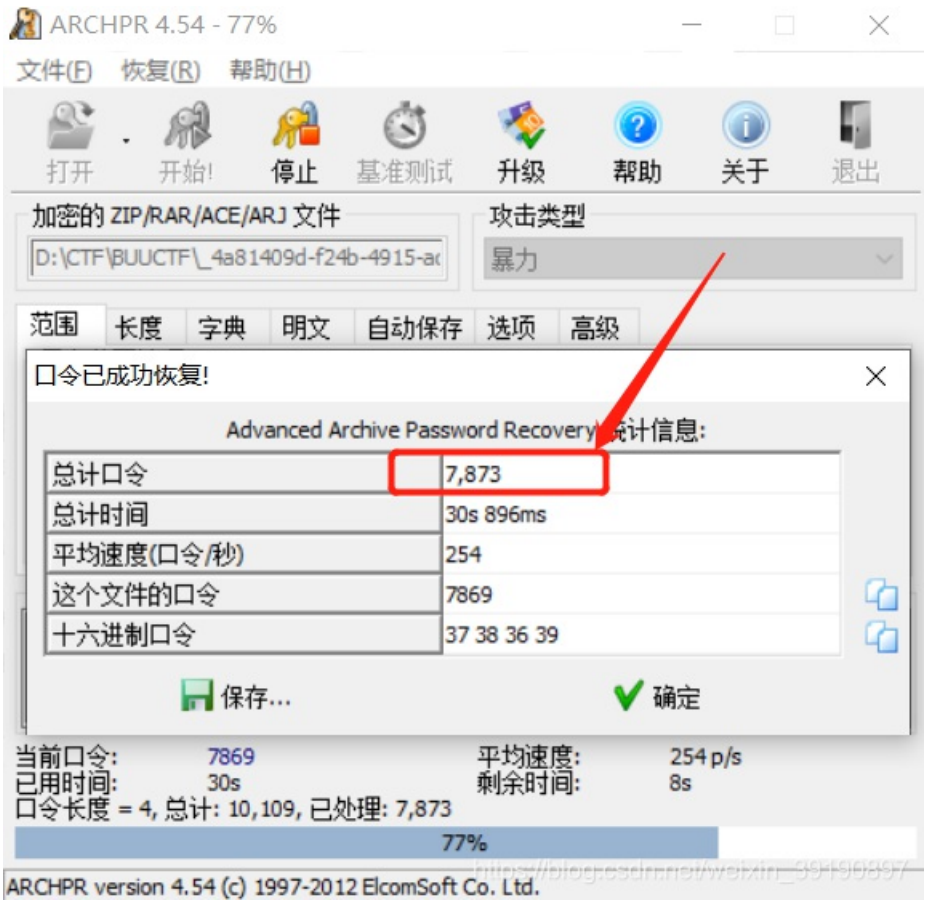
4、打开压缩文件，发现加密了：



5、使用 ARCHPR 爆破工具尝试进行 4 位数密码的破解：

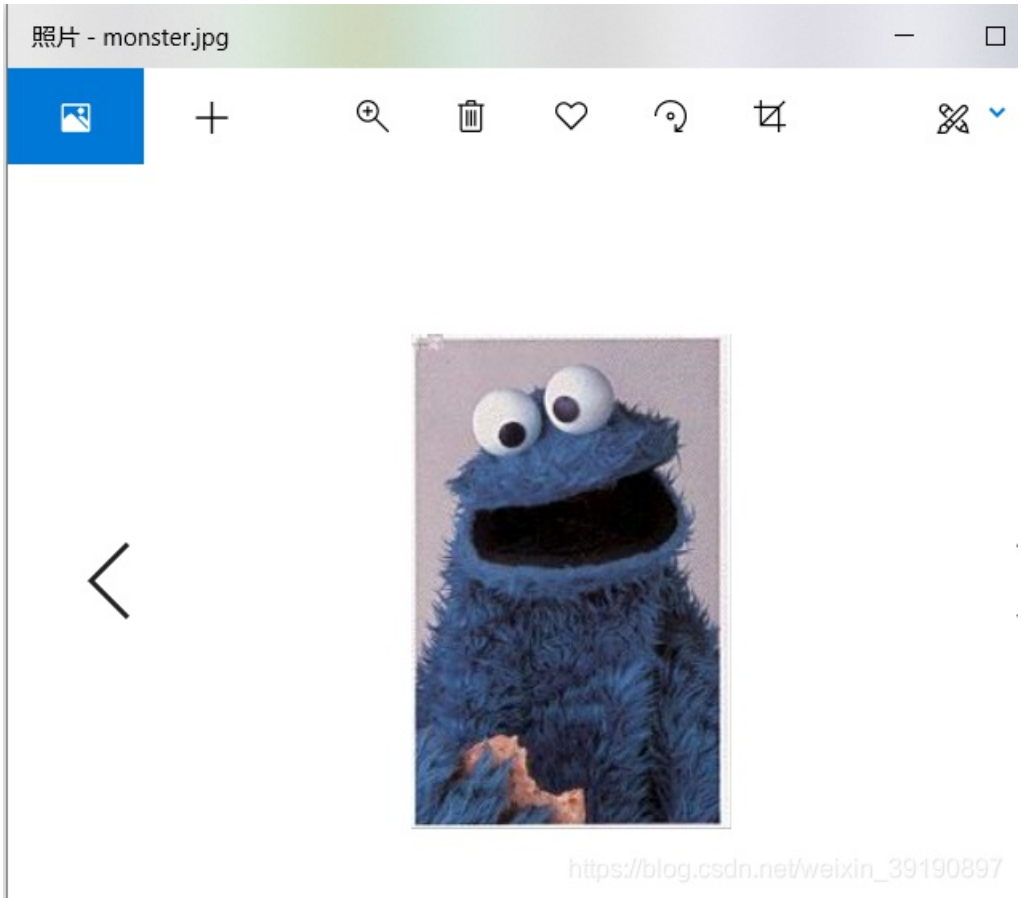


6、成功破解密码，获得 Flag：



No.10 二进制编码转AscII

1、下载后如下：



2、使用 Winhex 查看，发现末尾有二进制串：

```
D5 F4 AF A0 13 F9 7B FE  55 81 C3 C3 E5 41 CD B3
2A 47 12 4F A1 0D AA 84  80 36 BD 8F 02 AE A2 F3
20 77 76 54 87 D0 B6 D7  2A 52 95 B6 AC D2 09 C4
A0 96 D7 99 1F DA BB 49  E3 E3 F3 AC 2F D5 57 71
```

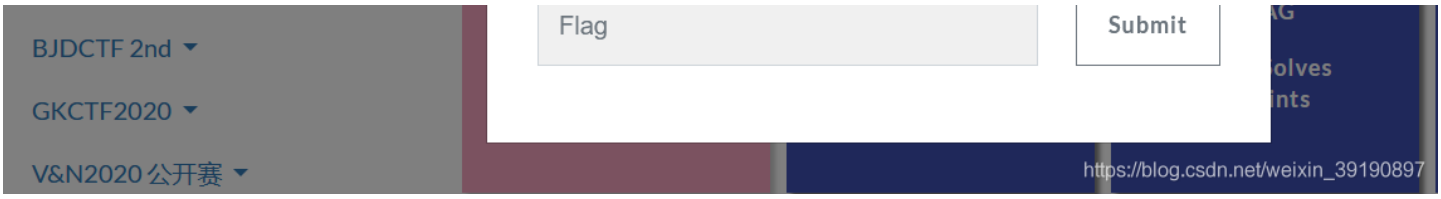
```
F9 D3 7C B3 40 71 4F D1  C6 5D F4 2D B5 D4 94 A1  ùÓ|³@qOÑÆ]ô-µÔ"¡
BD AF 64 80 91 21 3D 1A  B3 3C 3C 2B A1 6E 1E EE  ¾¯d€'!= ³<<+¡n î
5C EE AE EE 33 B2 AE AE  9B B9 79 0E B8 E1 71 A0  \î®î3²®®›¹y ¸áq
40 21 46 78 F7 D5 A0 FE  F0 79 E7 51 35 FB F5 7F  @!Fx÷Õ þðyçQ5ûõ
48 F8 0A 59 E4 94 BB 1A  29 30 31 31 30 31 30 31  Hø Yä"» )0110101
31 30 31 31 30 31 31 31  31 30 31 31 30 30 31 30  1011011110110010
31 30 31 31 30 31 30 31  31 30 31 31 30 31 30 31  1011010110110101
30 30 30 31 31 30 30 31  31 30 31 31 31 30 30 31  0001100110111001
31 2E 00 00 00                                    1.
```

3、猜测可用 8 位二进制一组，转换成 ASCII 编码，脚本如下：

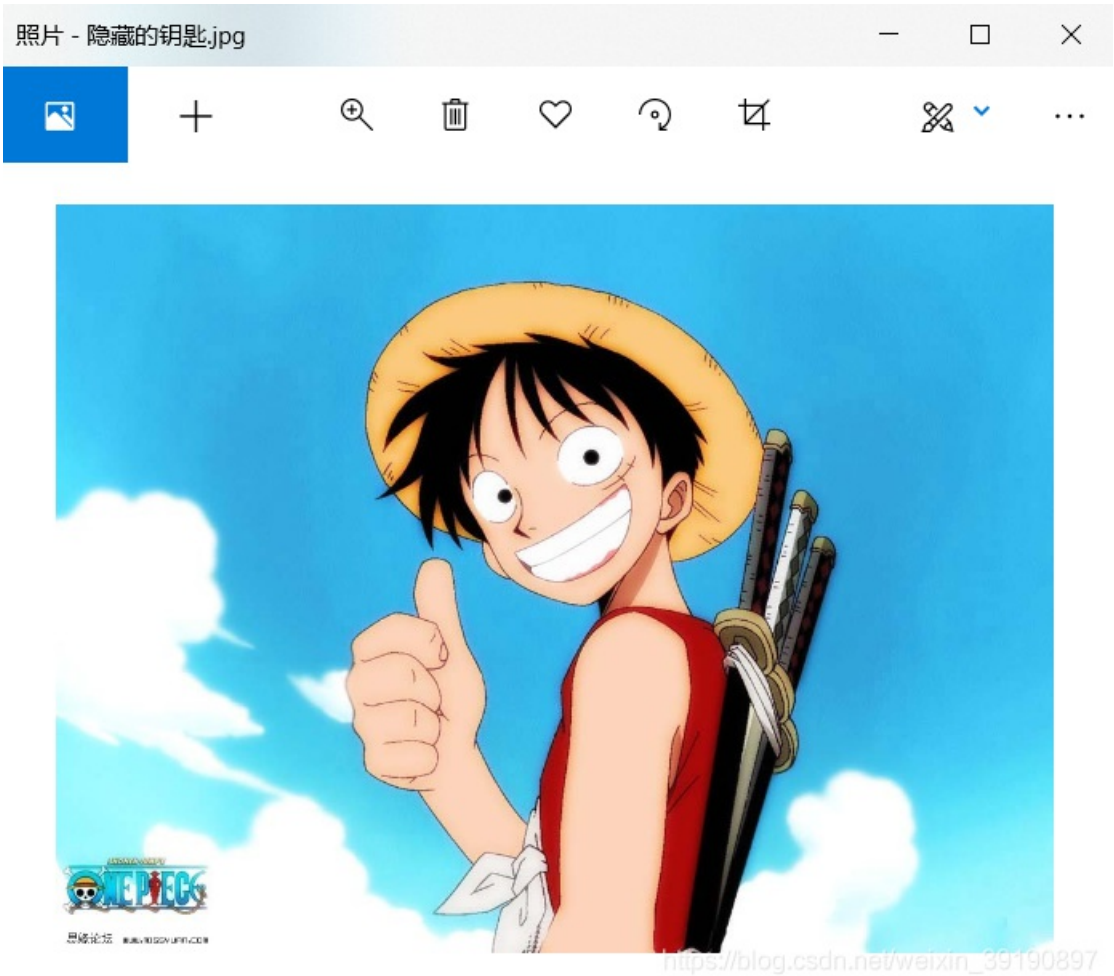
```
import binascii

a = '''01101011
01101111
01100101
01101011
01101010
00110011
01110011'''
b = ''
for ii in a.split('\n'):
 b += chr(int(ii,2))
print(b)
```

运行脚本，获得 flag：

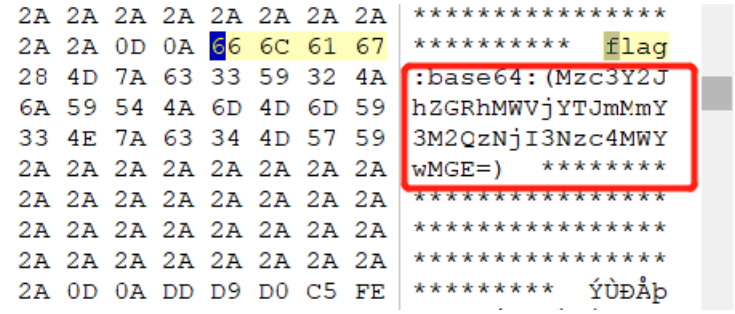
No.11 Winhex搜索获得flag



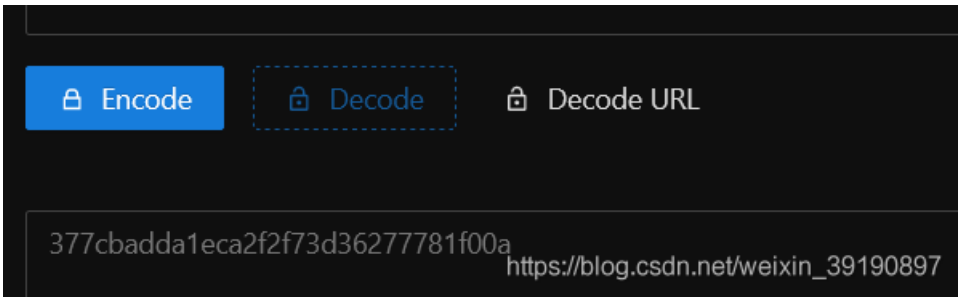
1、下载后打开如下：



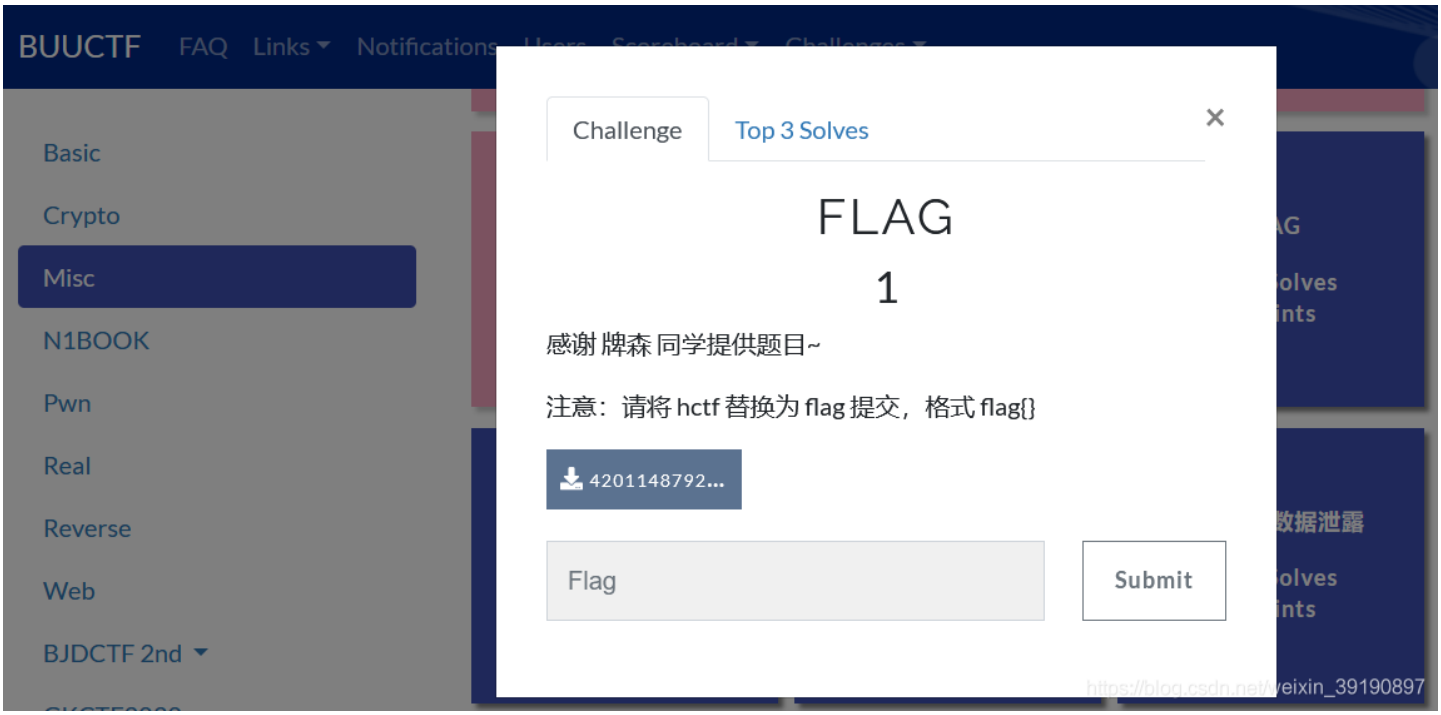
2、使用 winhex 打开图片并搜索 flag，看到 base64 编码字符串：



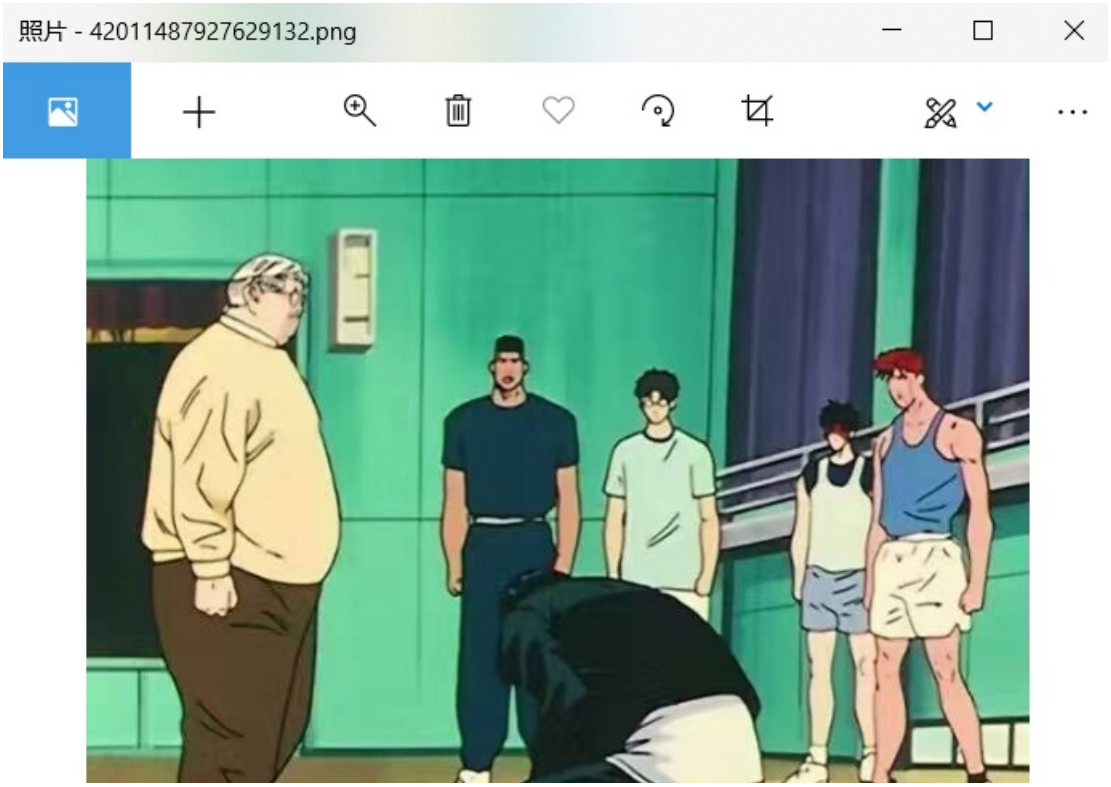
3、base64 解码获得 flag：



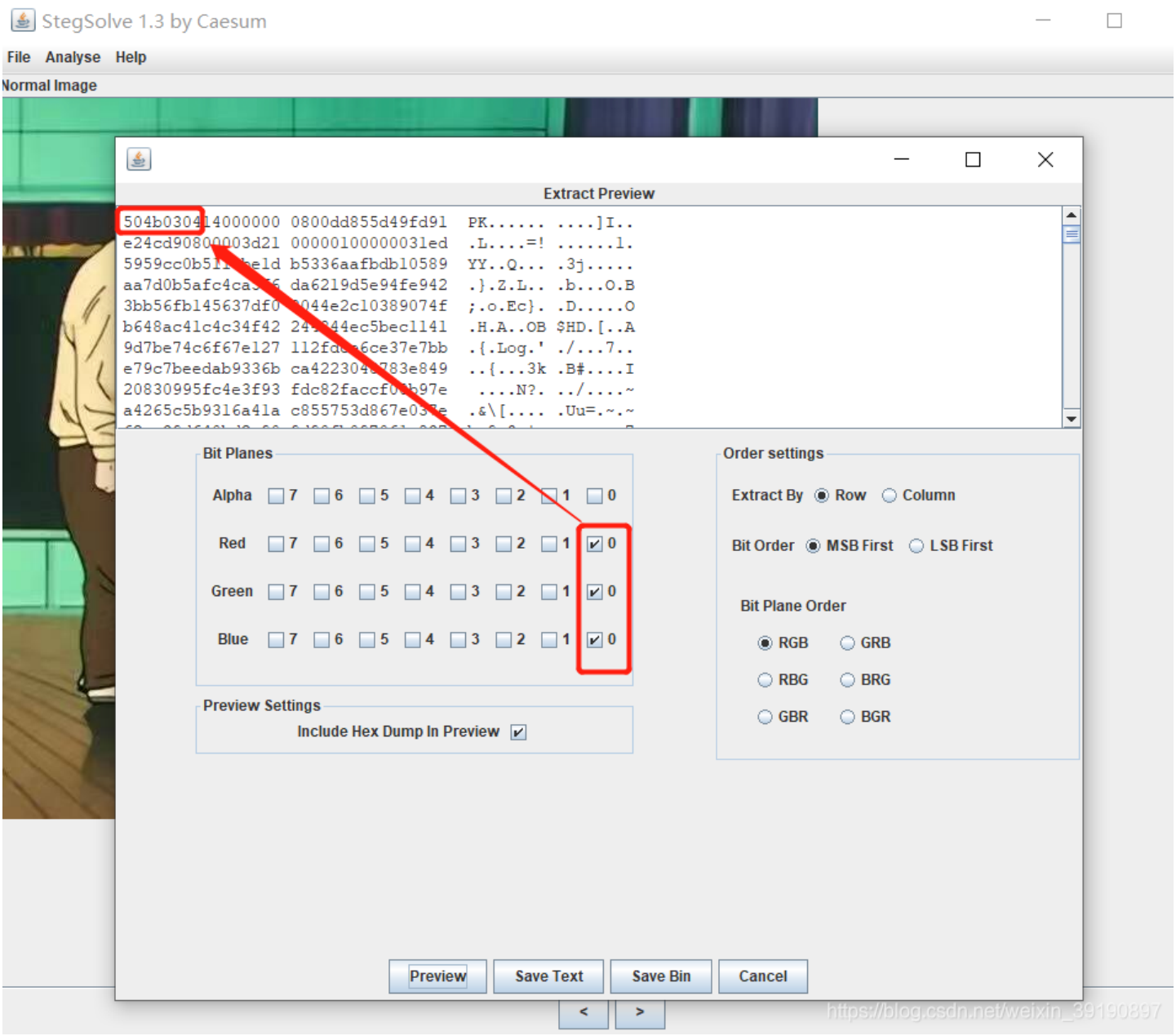
No.12 ZIP的16进制文件头



下载后是一张图片：



1、使用 stegsolve 打开图片，发现在 RGB 模式下 0 通道预览是个 ZIP 文件（文件头 `504b0304`），存在 LSB 隐写，Save Bin 保存为 flag.zip：

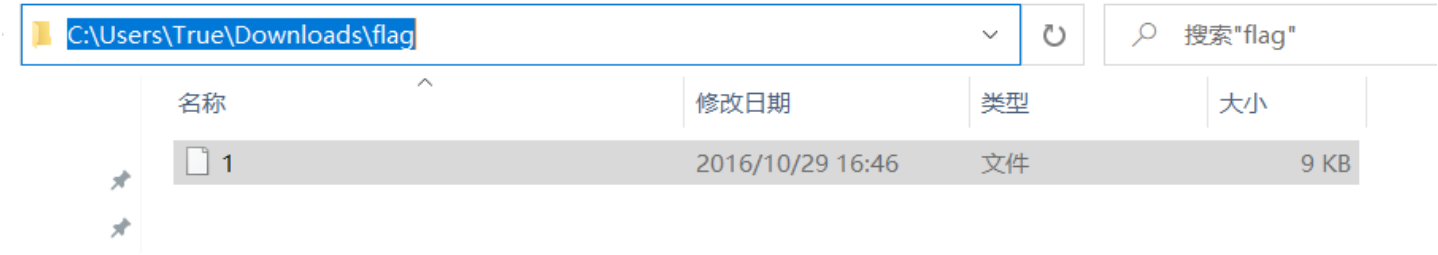


【注意】一定要对常见的文件头高度敏感：

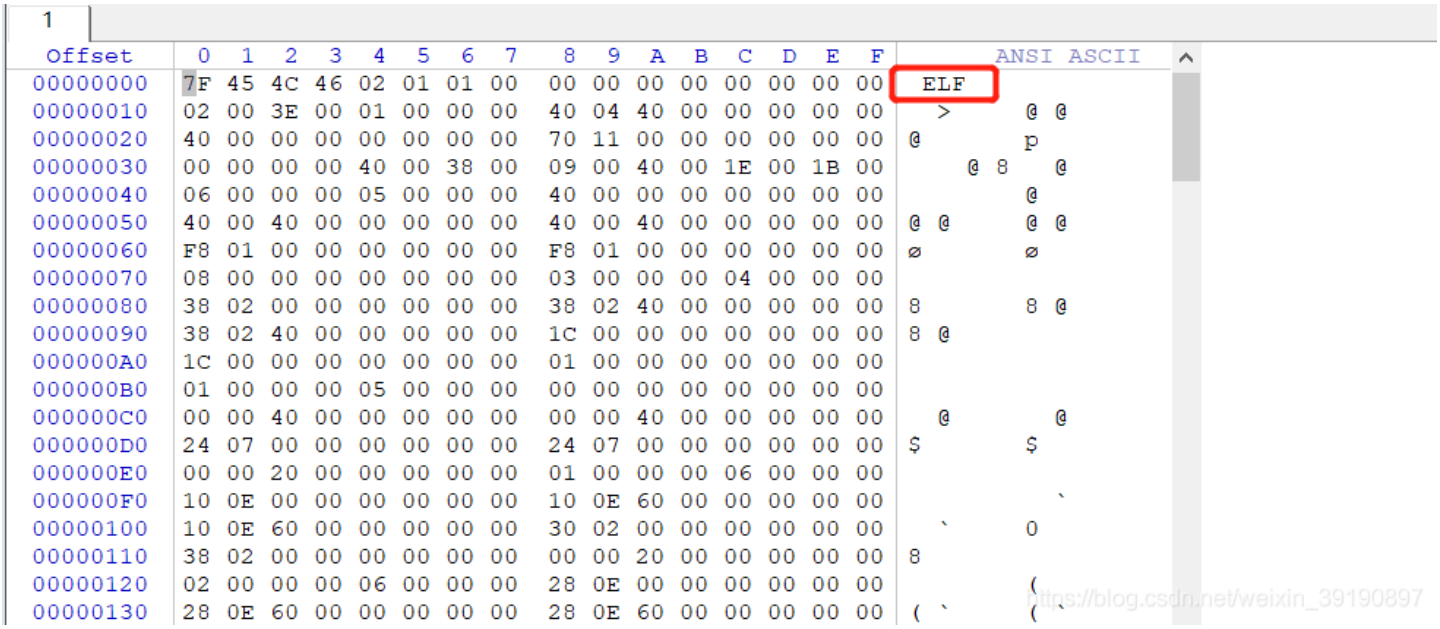
JPEG (jpg),	文件头：FFD8FF	文件尾：FF D9
PNG (png),	文件头：89504E47	文件尾：AE 42 60 82
GIF (gif),	文件头：47494638	文件尾：00 3B
Archive (zip),	文件头：504B0304	文件尾：50 4B
TIFF (tif),	文件头：49492A00	文件尾：
Windows Bitmap (bmp),	文件头：424D	文件尾：

CAD (dwg),	文件头：41433130	文件尾：
Adobe Photoshop (psd),	文件头：38425053	文件尾：
Rich Text Format (rtf),	文件头：7B5C727466	文件尾：
XML (xml),	文件头：3C3F786D6C	文件尾：
HTML (html),	文件头：68746D6C3E	
Email [thorough only] (eml),	文件头：44656C69766572792D646174653A	
Outlook Express (dbx),	文件头：CFAD12FEC5FD746F	
Outlook (pst),	文件头：2142444E	
MS Word/Excel (xls.or.doc),	文件头：D0CF11E0	
MS Access (mdb),	文件头：5374616E64617264204A	
WordPerfect (wpd),	文件头：FF575043	
Adobe Acrobat (pdf),	文件头：255044462D312E	
Quicken (qdf),	文件头：AC9EBD8F	
Windows Password (pwl),	文件头：E3828596	
RAR Archive (rar),	文件头：52 61 72 21 1A 07 00	文件尾：3D 7B 00 40 07 00
Wave (wav),	文件头：57415645	
AVI (avi),	文件头：41564920	
Real Audio (ram),	文件头：2E7261FD	
Real Media (rm),	文件头：2E524D46	
MPEG (mpg),	文件头：000001BA	
MPEG (mpg),	文件头：000001B3	
Quicktime (mov),	文件头：6D6F6F76	
Windows Media (asf),	文件头：3026B2758E66CF11	
MIDI (mid),	文件头：4D546864	

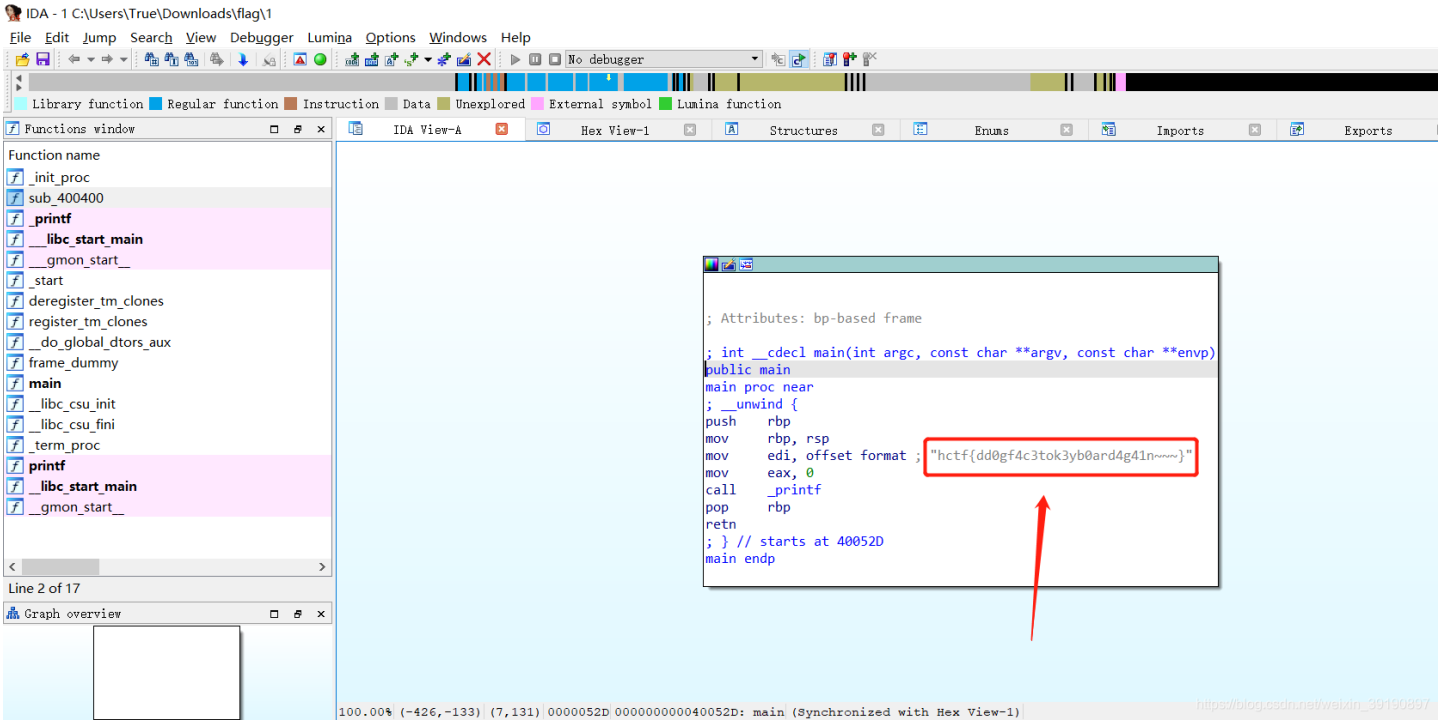
2、360解压缩导出来的 flag.zip 文件，获得一个没有后缀的文件：



3、编辑器或者 Winhex 打开发现是 ELF 文件：

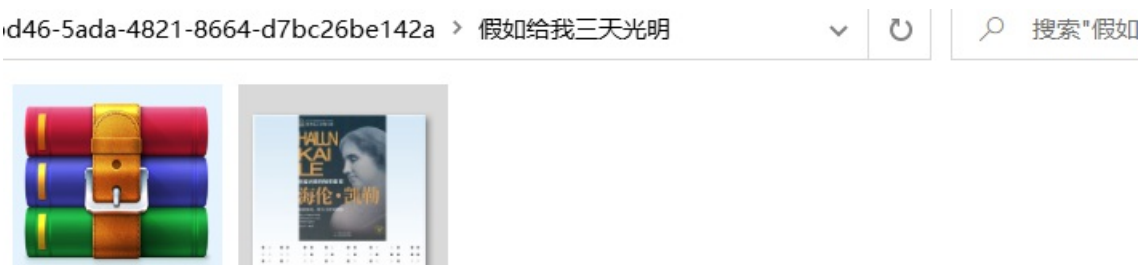


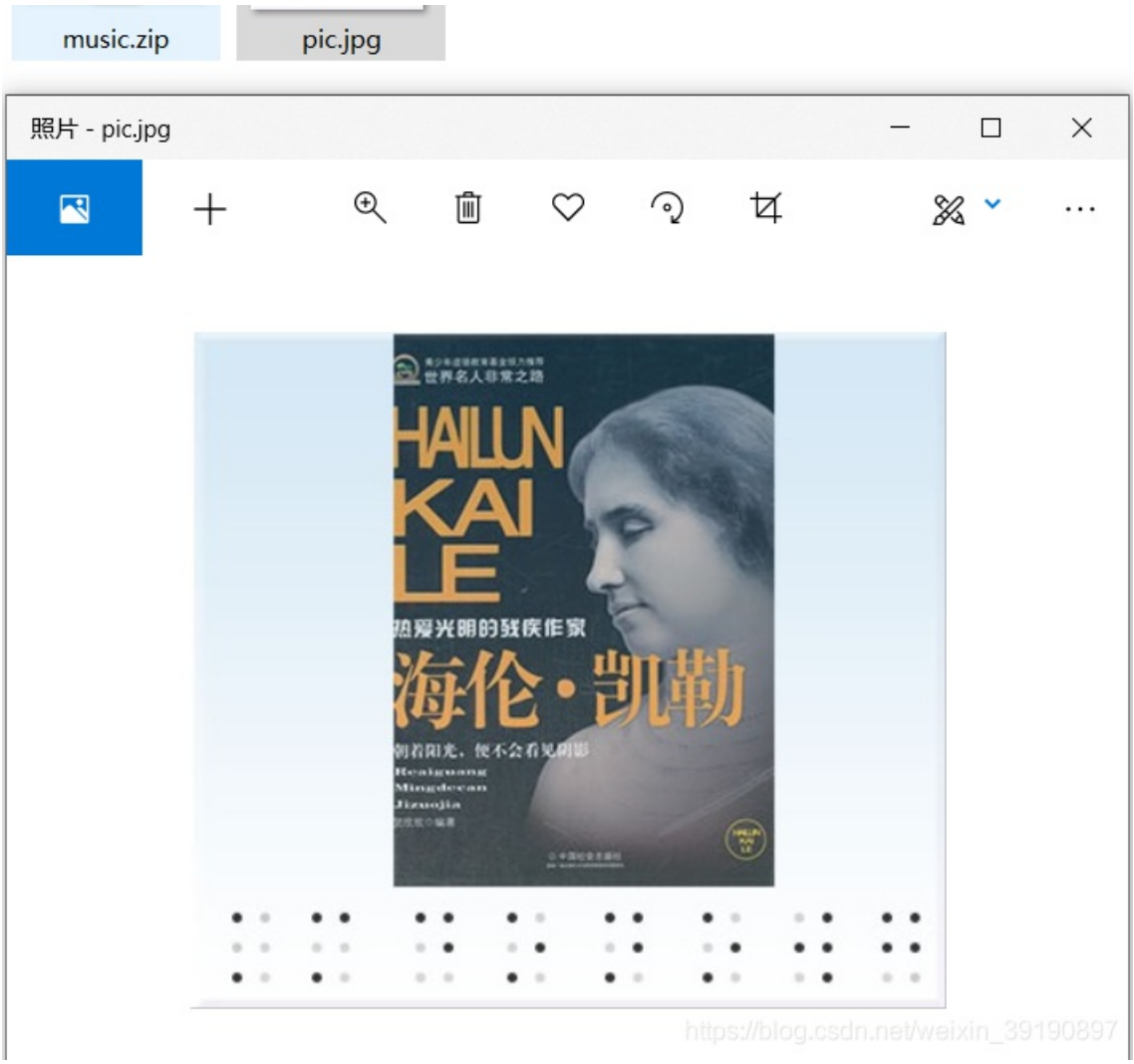
4、使用 IDA 打开可以获得 Flag：



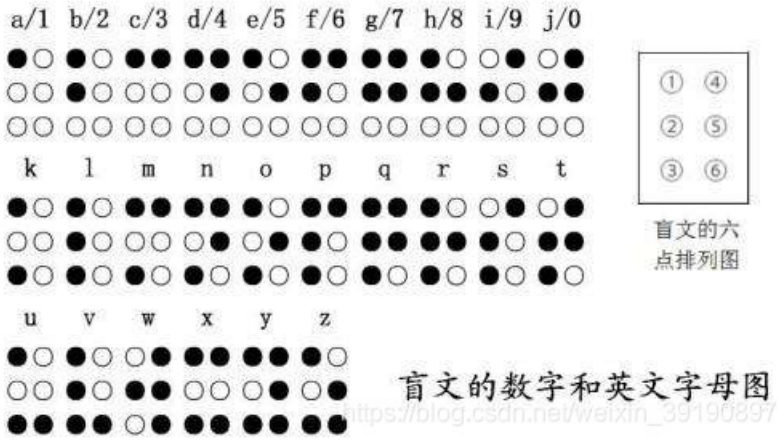
No.13 盲文与摩斯密码读取

1、下载后是一张图片和压缩文件：



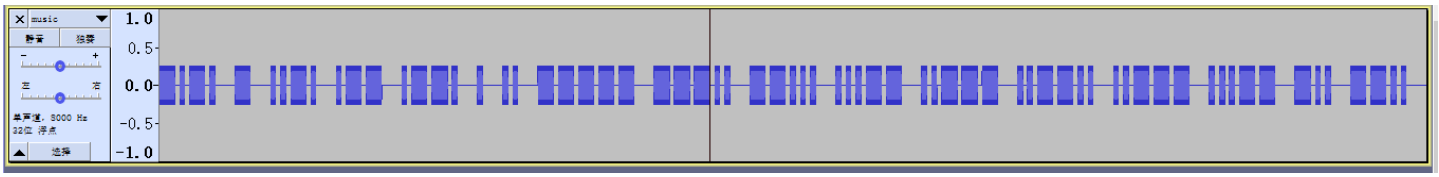


2、对照盲文解密图片中隐藏的字符串 `kmdonowg`：



3、解压缩 music.zip 文件发现需要密码，输入上面解密的字符串正好可以解开：

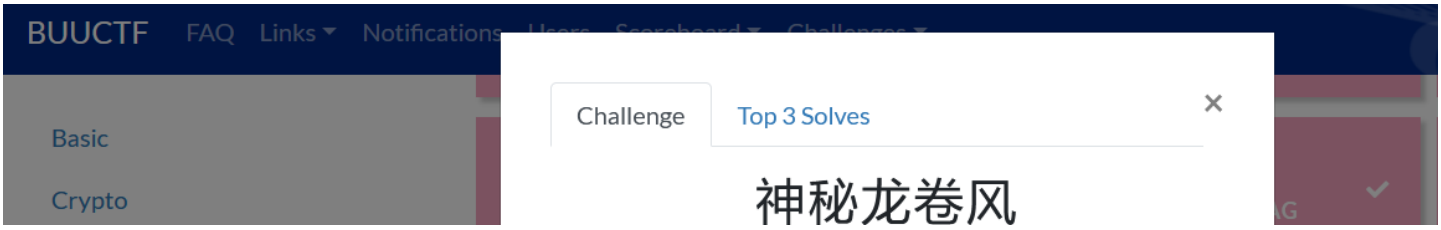
4、解压后得到一个 wav 音频文件，用 Audacity 查看其波形，如下图所示：

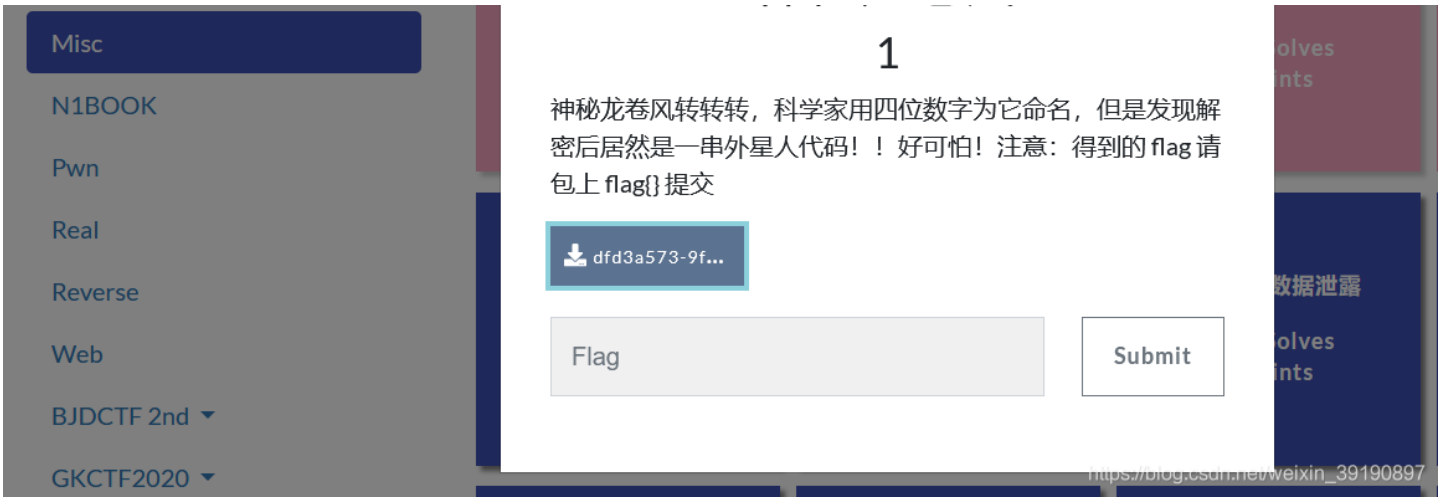


较长的蓝色即为“`-`”，较短的即为“`.`”，每一段由`-`和`.`组成的即为一个字符。可以按顺序将其莫斯解密，得到：

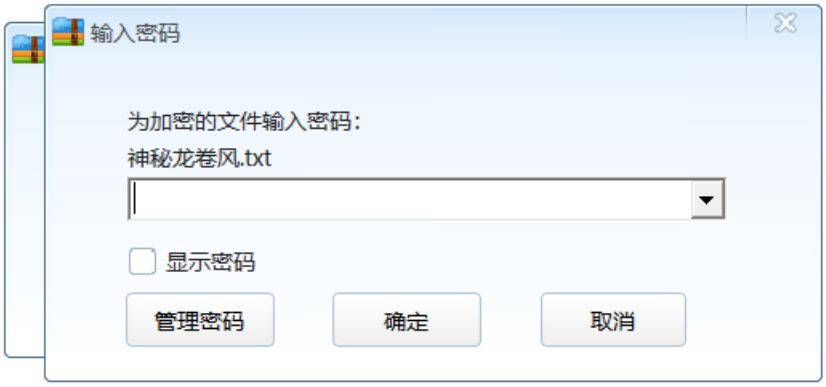
最终 flag：`flag{wpei08732?23dz}`。

No.14 Brainfuck编码与解码

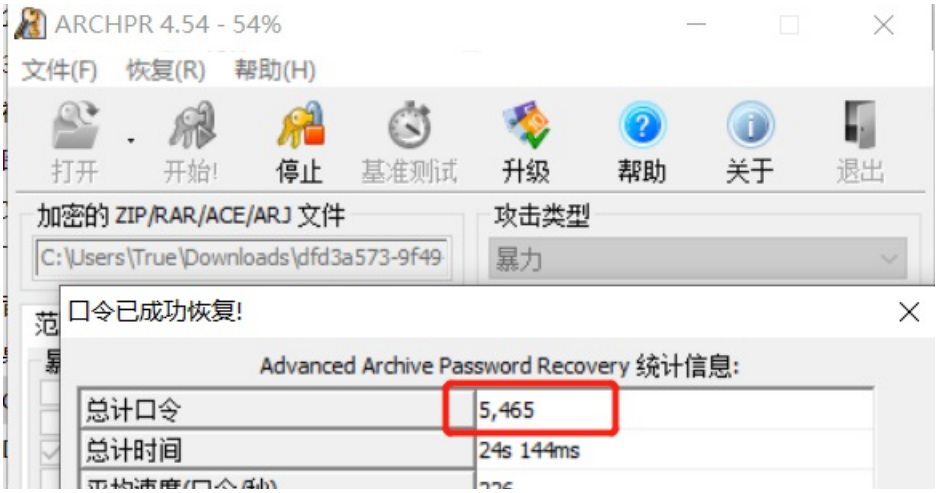




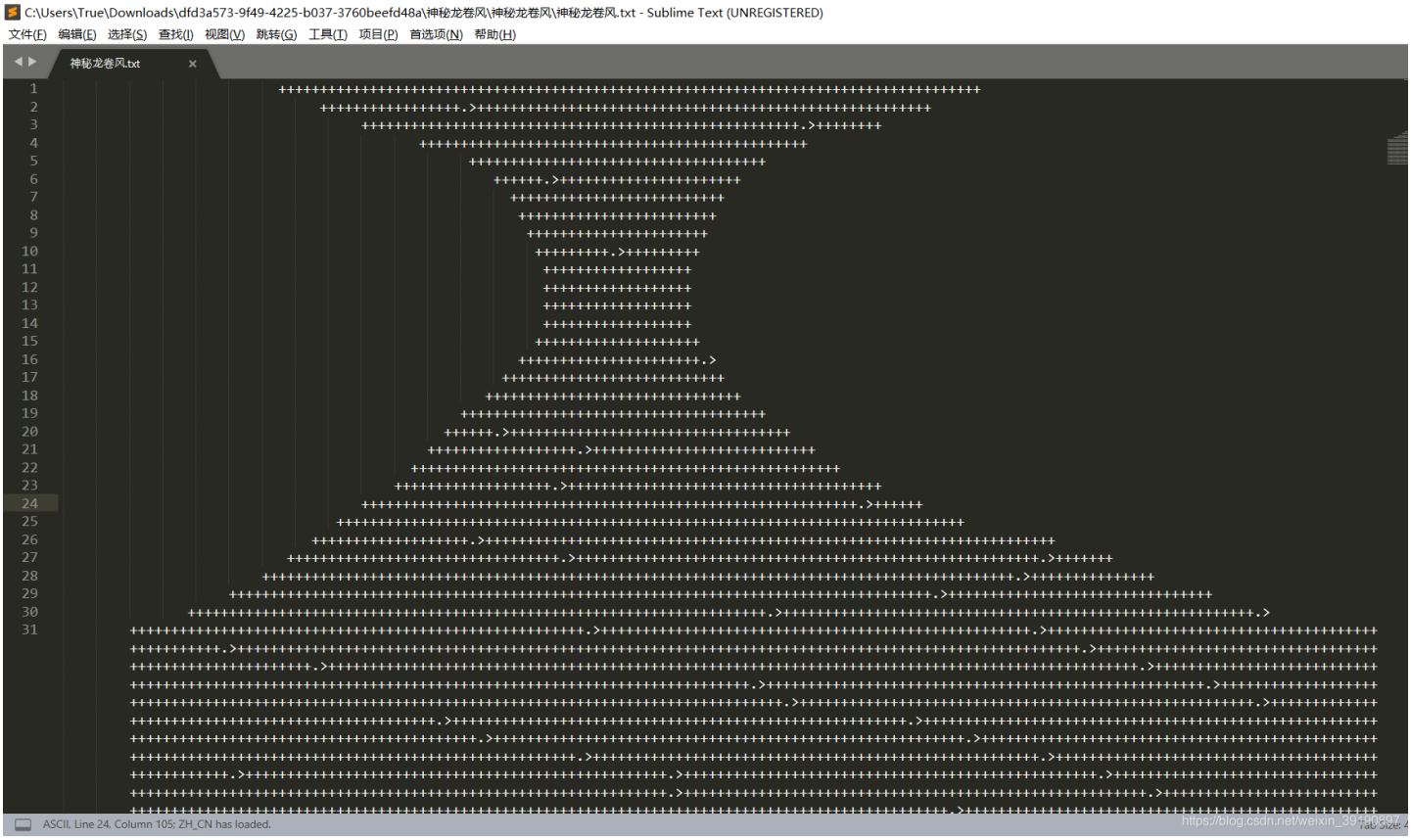
1、下载后解压需要密码：



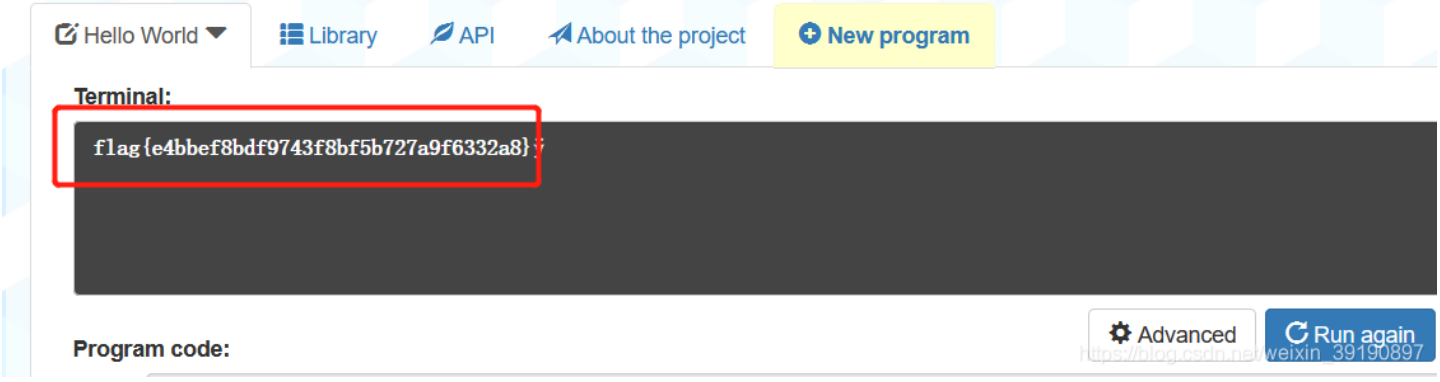
2、尝试使用 ARCHPR 进行四位数字的爆破：



3、解压后打开文件：



4、发现是 brainfuck 代码，使用 在线执行网站 运行即可得到 flag：



科普下 brainfuck 代码：

Müller的目标是建立一种简单的、可以用最小的编译器来实现的、符合图灵完全思想的编程语言。这种语言由八种状态构成，为Amiga机器编写的编译器（第二版）只有240个字节大小！

就象它的名字所暗示的，brainfuck程序很难读懂。尽管如此，brainfuck图灵机一样可以完成任何计算任务。虽然brainfuck的计算方式如此与众不同，但它确实能够正确运行。

这种语言基于一个简单的机器模型，除了指令，这个机器还包括：一个以字节为单位、被初始化为零的数组、一个指向该数组的指针（初始时指向数组的第一个字节）、以及用于输入输出的两个字节流。

这种语言，是一种按照“Turing complete（图灵完备）”思想设计的语言，它的主要设计思路是：用最小的概念实现一种“简单”的语言，BrainF**k 语言只有八种符号，所有的操作都由这八种符号的组合来完成。

字符标识

编辑

下面是这八种状态的描述，其中每个状态由一个字符标识：

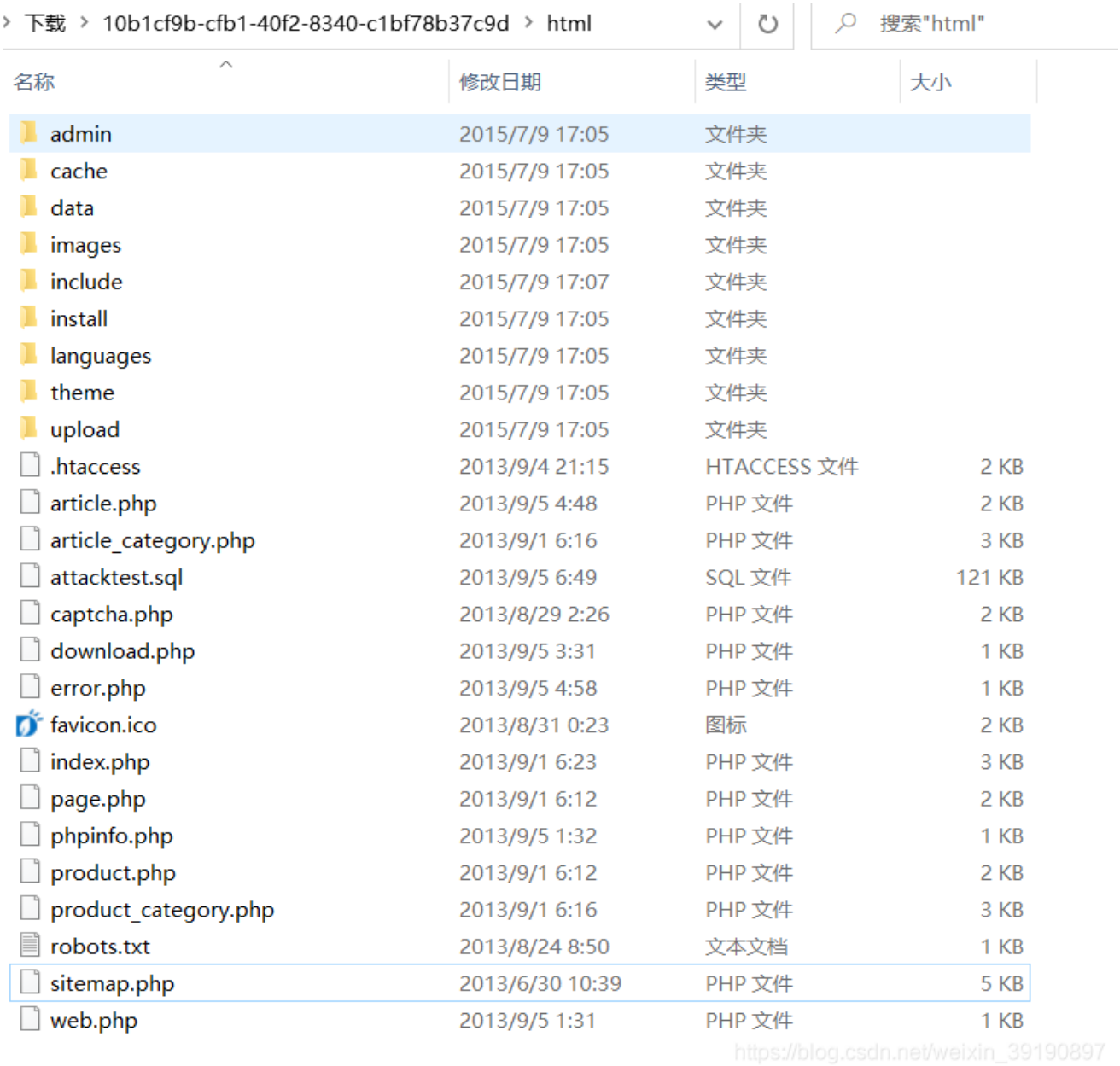
字符	含义
>	指针加一
<	指针减一
+	指针指向的字节的值加一
-	指针指向的字节的值减一
.	输出指针指向的单元内容（ASCⅡ码）
,	输入内容到指针指向的单元（ASCⅡ码）
[	如果指针指向的单元值为零，向后跳转到对应的]指令的次一指令处
]	如果指针指向的单元值不为零，向前跳转到对应的[指令的次一指令处

（按照更节省时间的简单说法，"]"也可以说成“向后跳转到对应的"["状态”。这两解释是一样的。）

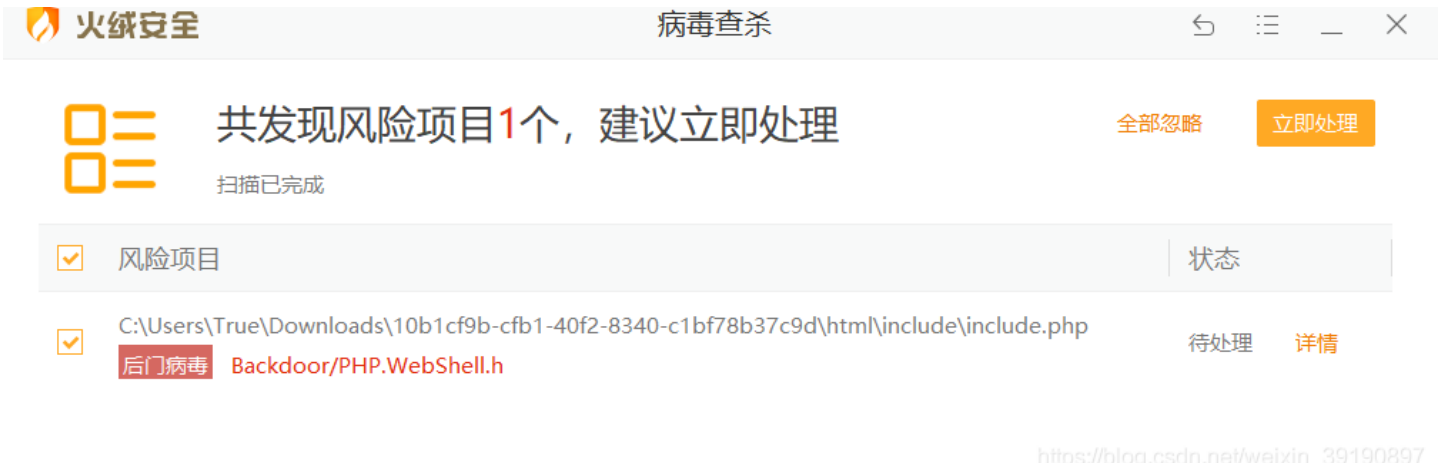
No.15 后门查杀之后门识别


BUUCTF
FAQ
Links
Notifications
Basic
Crypto
Misc
N1BOOK
Pwn
Real
Reverse
Web
BJDCTF 2nd
GKCTF2020
V&N2020 公开赛
VNCTF2021
Challenge
Top 3 Solves
后门查杀
1
小白的网站被小黑攻击了，并且上传了Webshell，你能帮小白找到这个后门么？(Webshell中的密码(md5)即为答案)。注意：得到的 flag 请包上 flag{} 提交
10b1cf9b-cf...
Flag
Submit
数据包中的线索
九连环
面具下的flag
https://blog.csdn.net/weixin_39190897

1、下载后解压缩如下：



2、找 Webshell，直接上火绒扫描该文件夹：

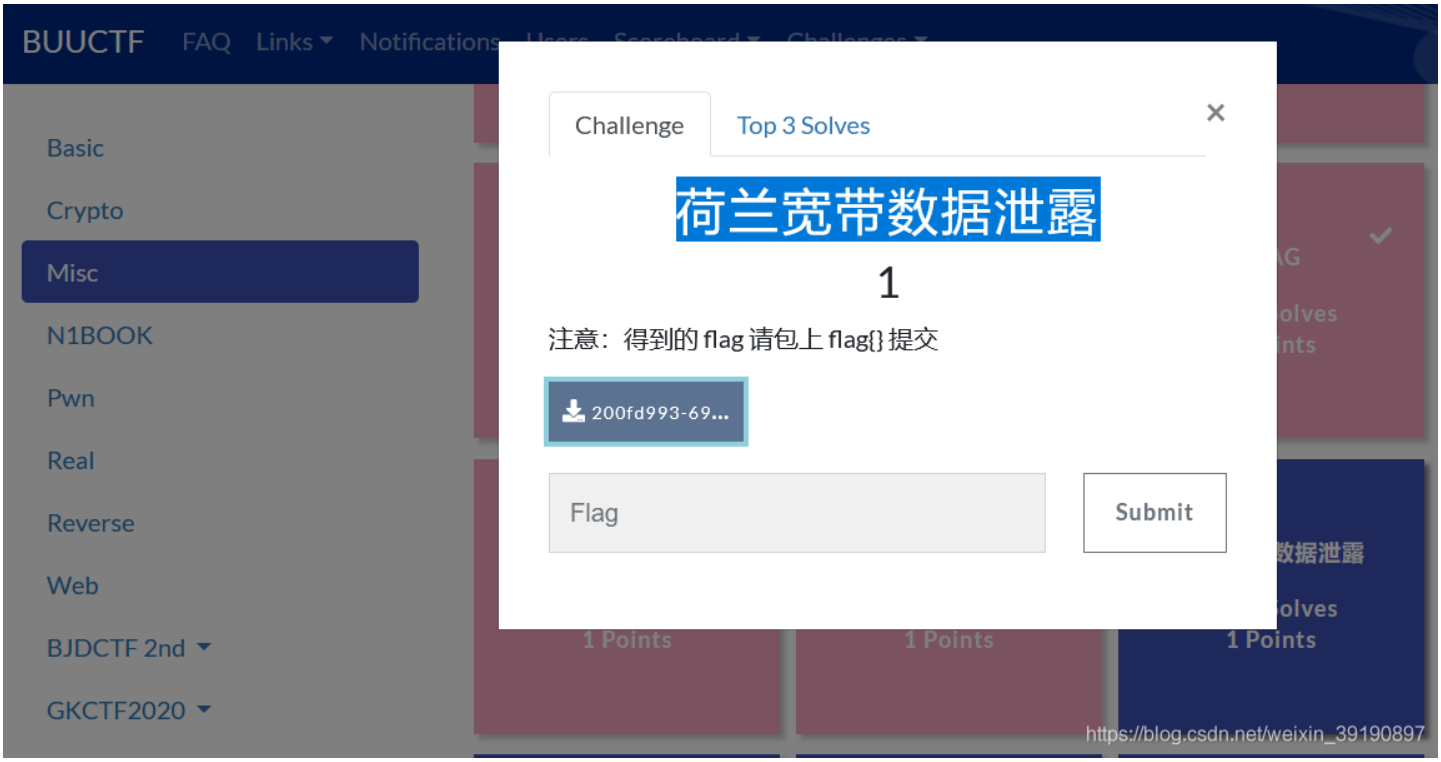


3、打开该文件，发现 flag：`flag{6ac45fb83b3bc355c024f5034b947dd3}`：

include.php

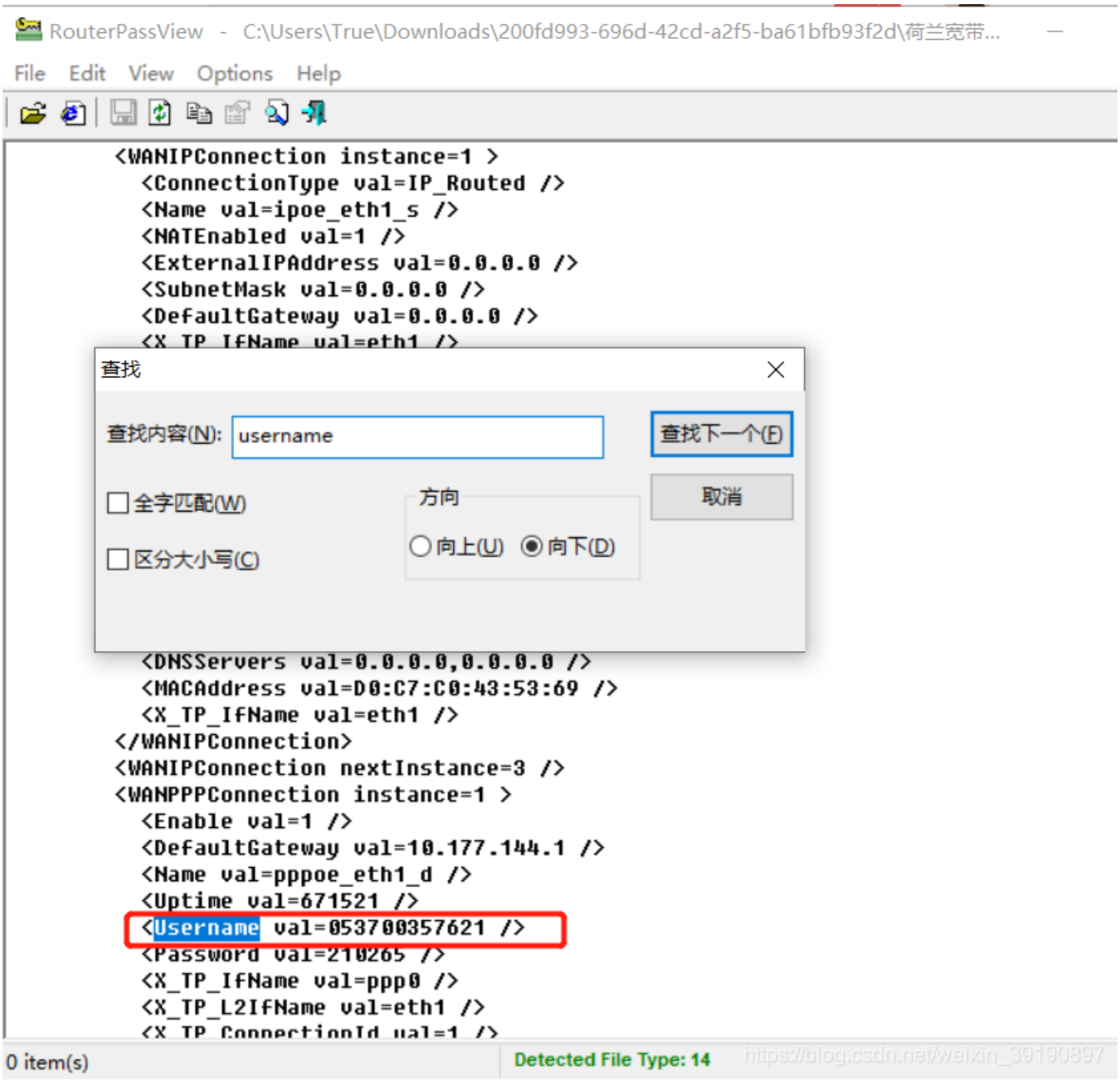
```php
<?php
//ini_set('display_errors',1);
@error_reporting(7);
@session_start();
@set_time_limit(0);
@set_magic_quotes_runtime(0);
if( strpos( strtolower( $_SERVER['HTTP_USER_AGENT'] ), 'bot' ) !== false ) {
    header('HTTP/1.0 404 Not Found');
    exit;
}
ob_start();
$mtime = explode(' ', microtime());
$starttime = $mtime[1] + $mtime[0];
define('SA_ROOT', str_replace('\\', '/', dirname(__FILE__)).'/');
define('SELF', $_SERVER['PHP_SELF'] ? $_SERVER['PHP_SELF'] : $_SERVER['SCRIPT_NAME']);
define('IS_WIN', DIRECTORY_SEPARATOR == '\\');
define('IS_GPC', get_magic_quotes_gpc());
$dis_func = get_cfg_var('disable_functions');
define('IS_PHPINFO', (!eregi("phpinfo",$dis_func)) ? 1 : 0 );

if( IS_GPC ) {
    $_POST = s_array($_POST);
}
$P = $_POST;
unset($_POST);
/*==================== 程序配置 ====================*/

//echo encode_pass('angel');exit;
// 如果需要密码验证,请修改登陆密码,留空为不需要验证
$pass  = '6ac45fb83b3bc355c024f5034b947dd3'; //angel

//如您对 cookie 作用范围有特殊要求，或登录不正常，请修改下面变量，否则请保持默认
// cookie 前缀
$cookiepre = '';
// cookie 作用域
```


No.16 路由器信息数据查看



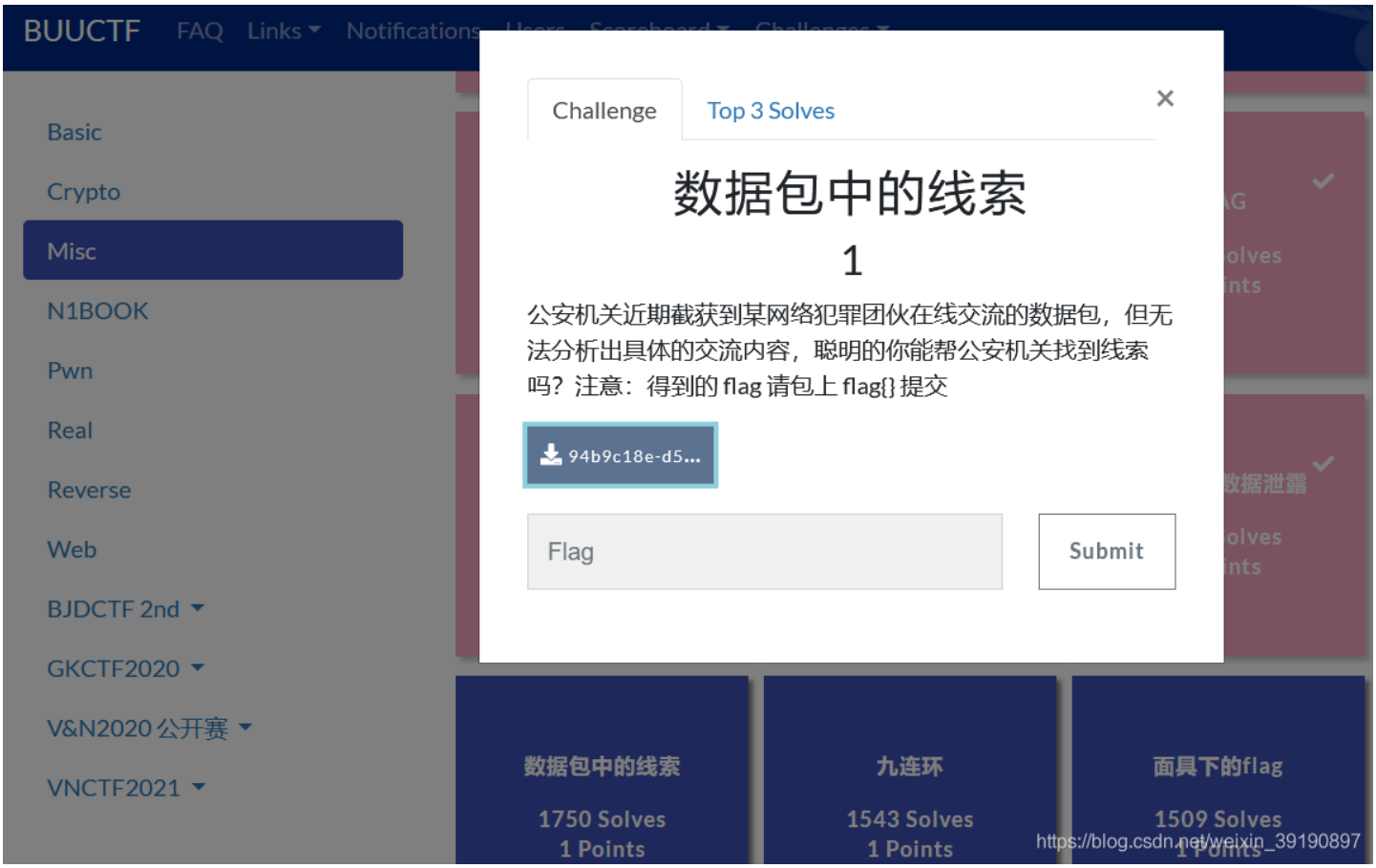
1、下载后解压缩是个 conf.bin 文件，路由器信息数据，一般包含账号密码，题目并没有提示 flag 是什么，猜测是账号或者密码加上格式为最终 flag：

2、无法使用编辑器直接打开阅读该类文件，需要用 `RouterPassView` 工具查看，打开后搜索 username 或者 password：

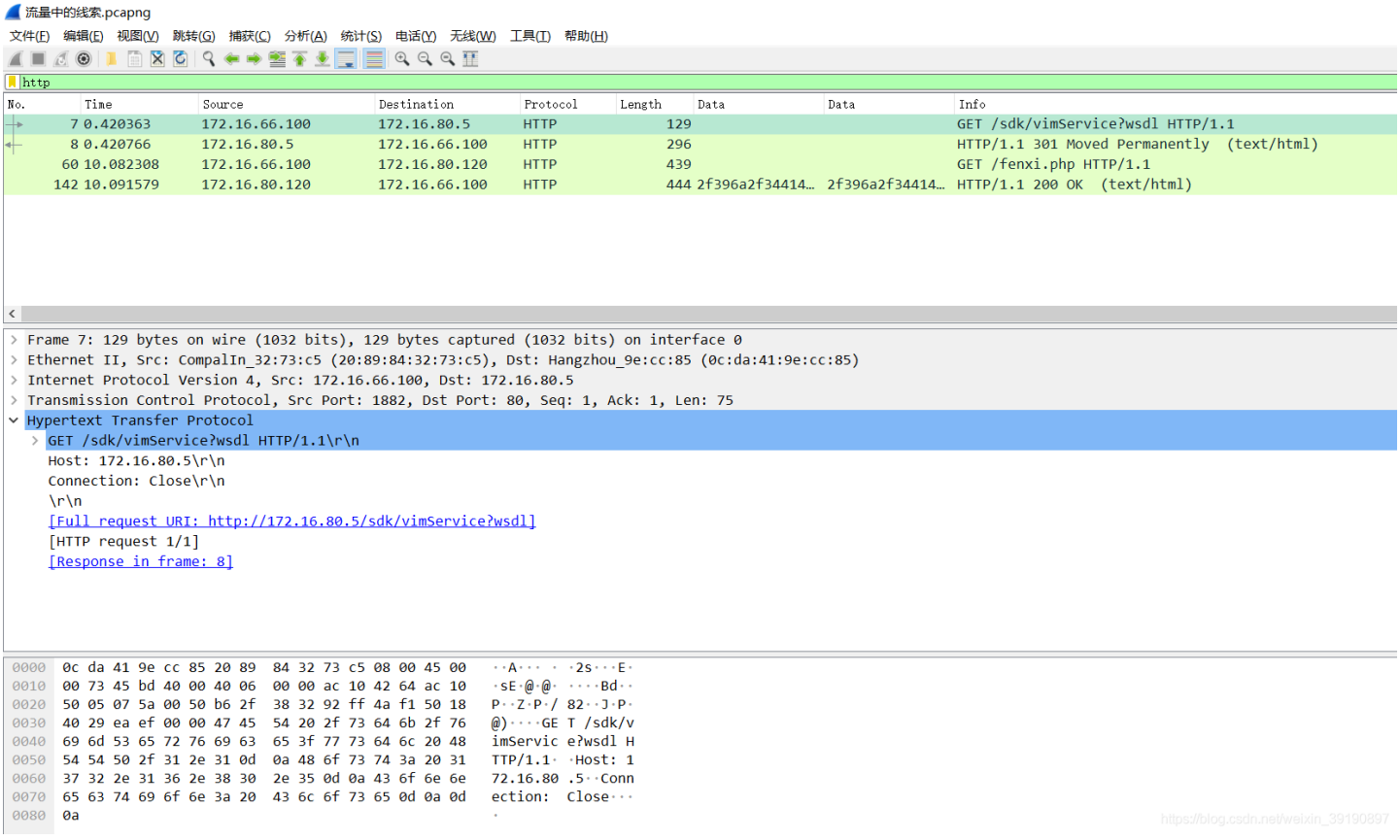


最后发现是用户名为 flag，加上格式提交即可 `flag{053700357621}`。

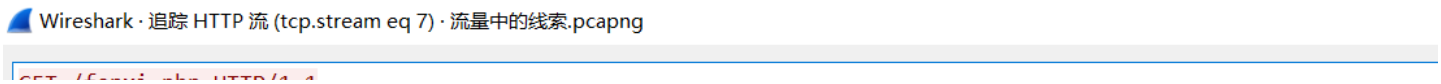
No.17 base64流量导出图片



1、下载后用 wireshark 打开，大部分都是 TCP 的包，直接过滤出 http 的包：



2、追踪 http 流：



```
GET /fenxi.php HTTP/1.1
Host: 172.16.80.120
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.
537.36
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8

HTTP/1.1 200 OK
Date: Tue, 18 Aug 2015 09:09:39 GMT
Server: Apache
X-Powered-By: PHP/5.3.3
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8
```

/9j/4AAQSkZJRgABAQEAYABgAAD/2wBDAAIBAQIBAQICAgICAgICAwUDAwMDAwYEBAMFBwYHBwcG
BwcICQsJCAgKCAcHCg0KCgsMDAwMBwkODw0MDgsMDAz/2wBDAQICAgMDAwYDAwYMCAcIDAwMDAwM
DAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAz/wAARCAHgAkQDASIA
AhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAAAAECAwQFBgcICQoL/8QAtRAAAgEDAwIEAwUFBAQA
AAF9AQIDAAQRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkKFhcYGRolJicoKSo0NTY3
ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqDhIWGh4iJipKTlJWWl5iZmqKjpKWm
p6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uHi4+Tl5ufo6erx8vP09fb3+Pn6/8QAHwEA
AwEBAQEBAQEBAQAAAAAAAAECAwQFBgcICQoL/8QAtREAAgECBAQDBAcFBAQAAQJ3AAECAxEEBSEx
BhJBUQdhcRMiMoEIFEKRobHBCSMzUvAVYnLRChYkNOEl8RcYGRomJygpKjU2Nzg5OkNERUZHSElK
U1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6goOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3
uLm6wsPExcbHyMnK0tPU1dbX2Nna4uPk5ebn6Onq8vP09fb3+Pn6/9oADAMBAAIRAxEAPwD9/KKK
KACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAoooo
AKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigA
ooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACi
iigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKK
KACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAoooo
AKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigA
ooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACi
iigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKK
KACiiigAooooAKKKKACiiigDjfiX+0V8Pvgvr2j6V4x8deDfCeqeIRM2lWes61bWFxqYhUNMYEld
Wl8tSC2wHaCCcV0nhrxNpvjTw7Yaxo+oWOraTqlvHd2V7ZzrPb3kMihkljkUlXRlIIZSQQQQa/N/

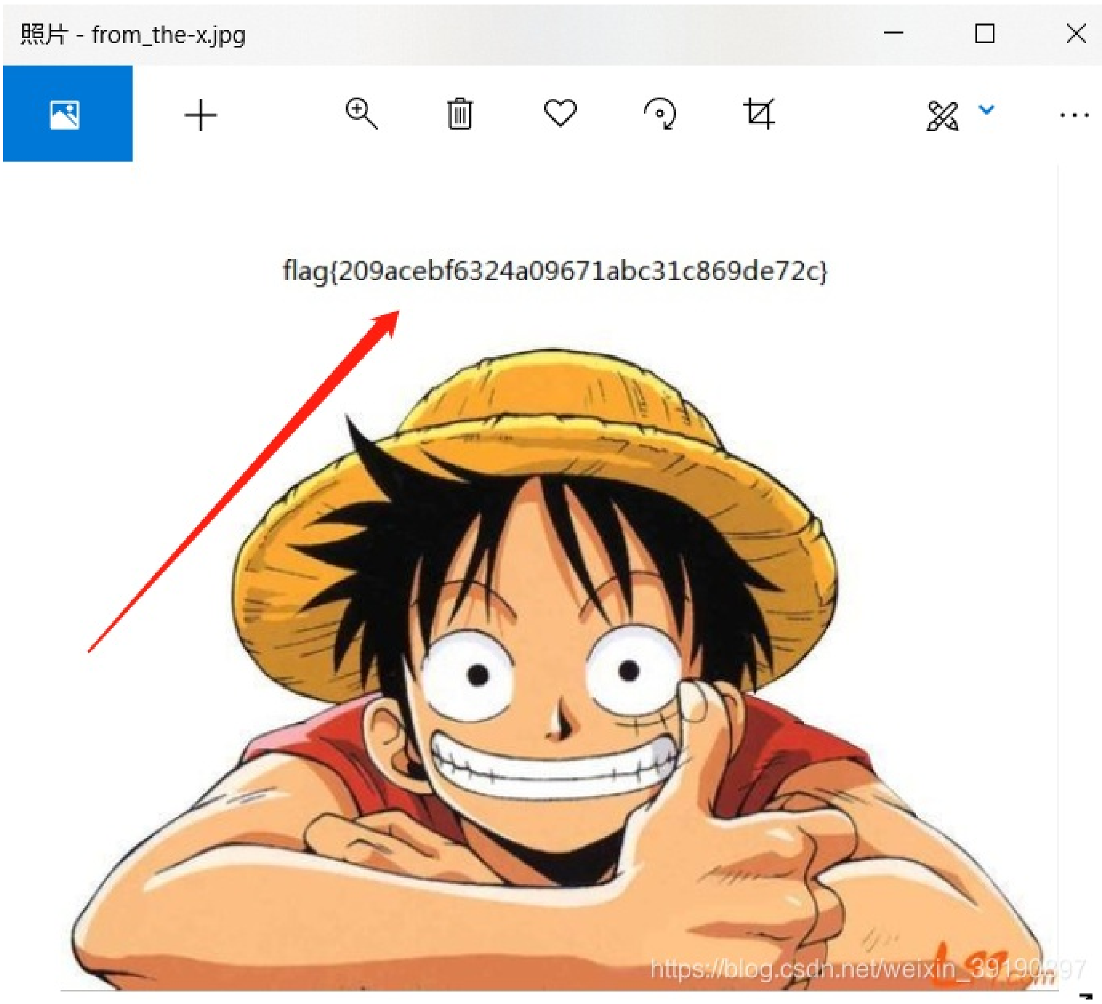
分组 60。1 客户端 分组，1 服务器 分组，1 turn(s).点击选择。

整个对话（86 kB）

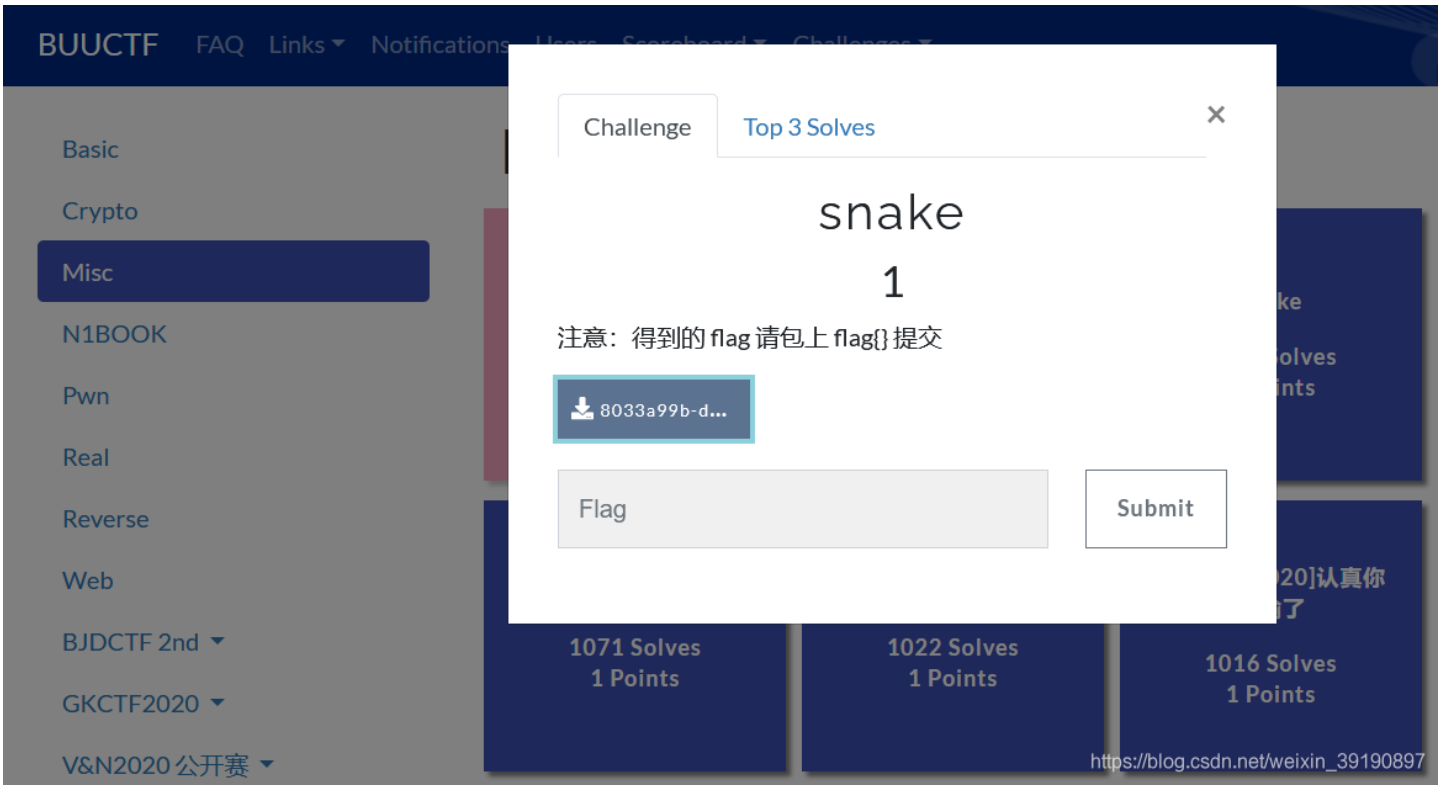
显示和保

3、传输内容好像是 base64 密文，使用 在线网站 解码 base64：

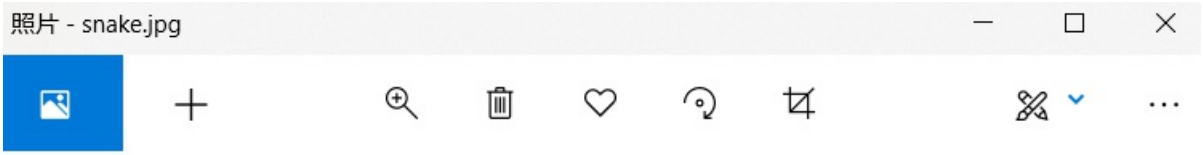
4、发现解码出来的内容是 jpg 图片格式并以自动转为图片，下载即可得到 flag：



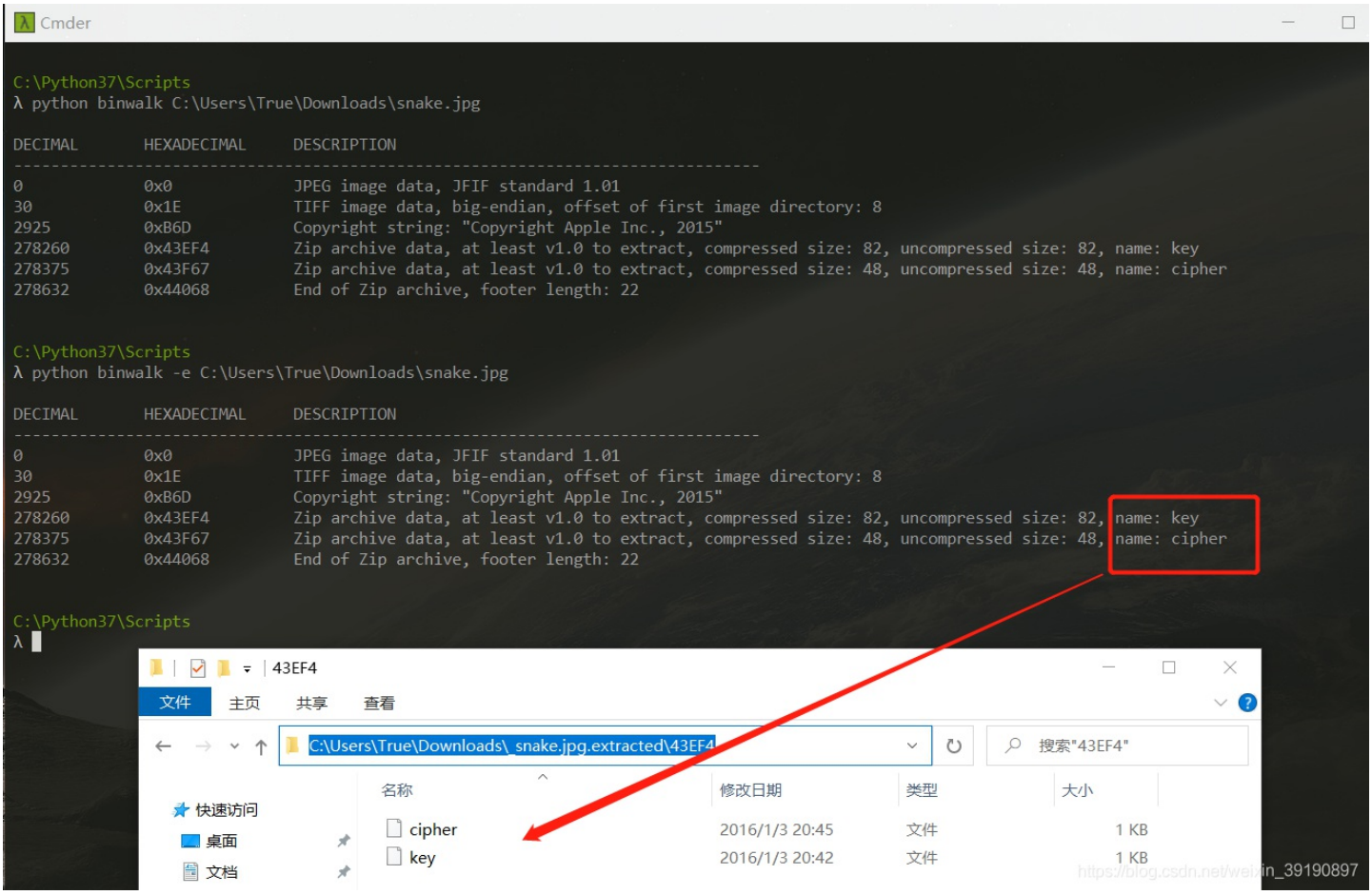
No.18 脑洞大开歌名猜密钥



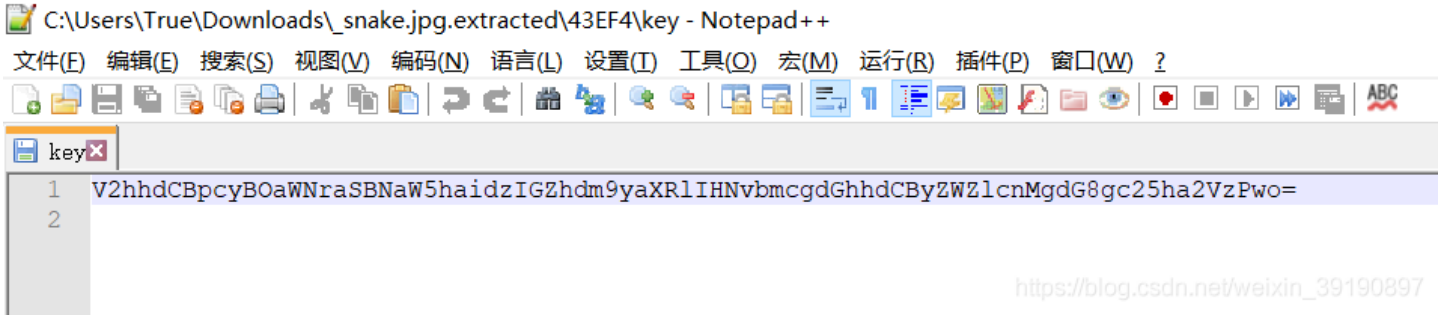
下载后是一张蛇的照片：



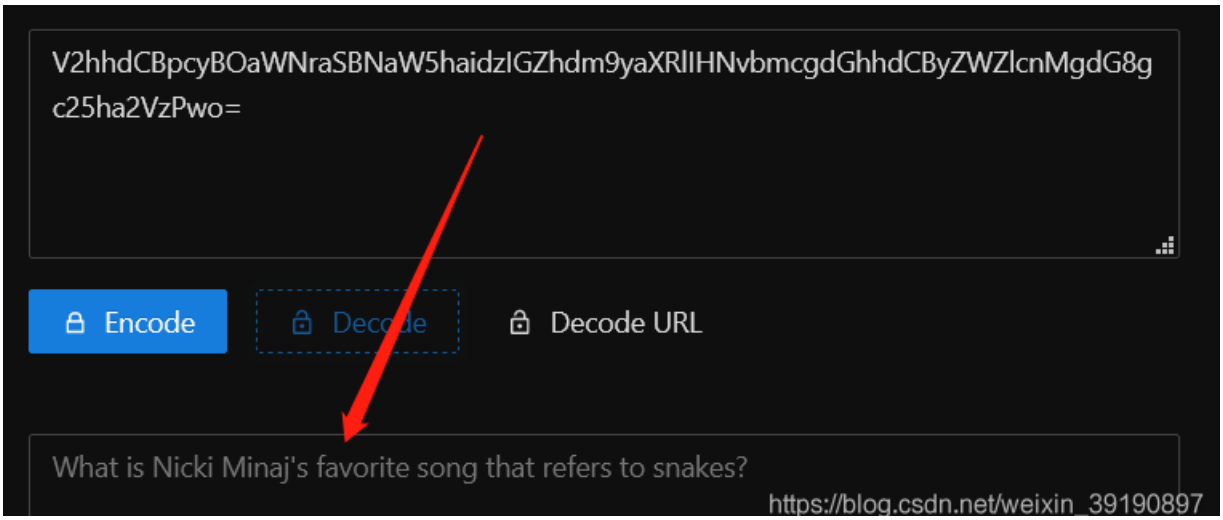
1、使用 Binwalk 分离提取出两个隐藏文件：



2、打开 Key 文件发现 Base64 编码：



3、解码如下：

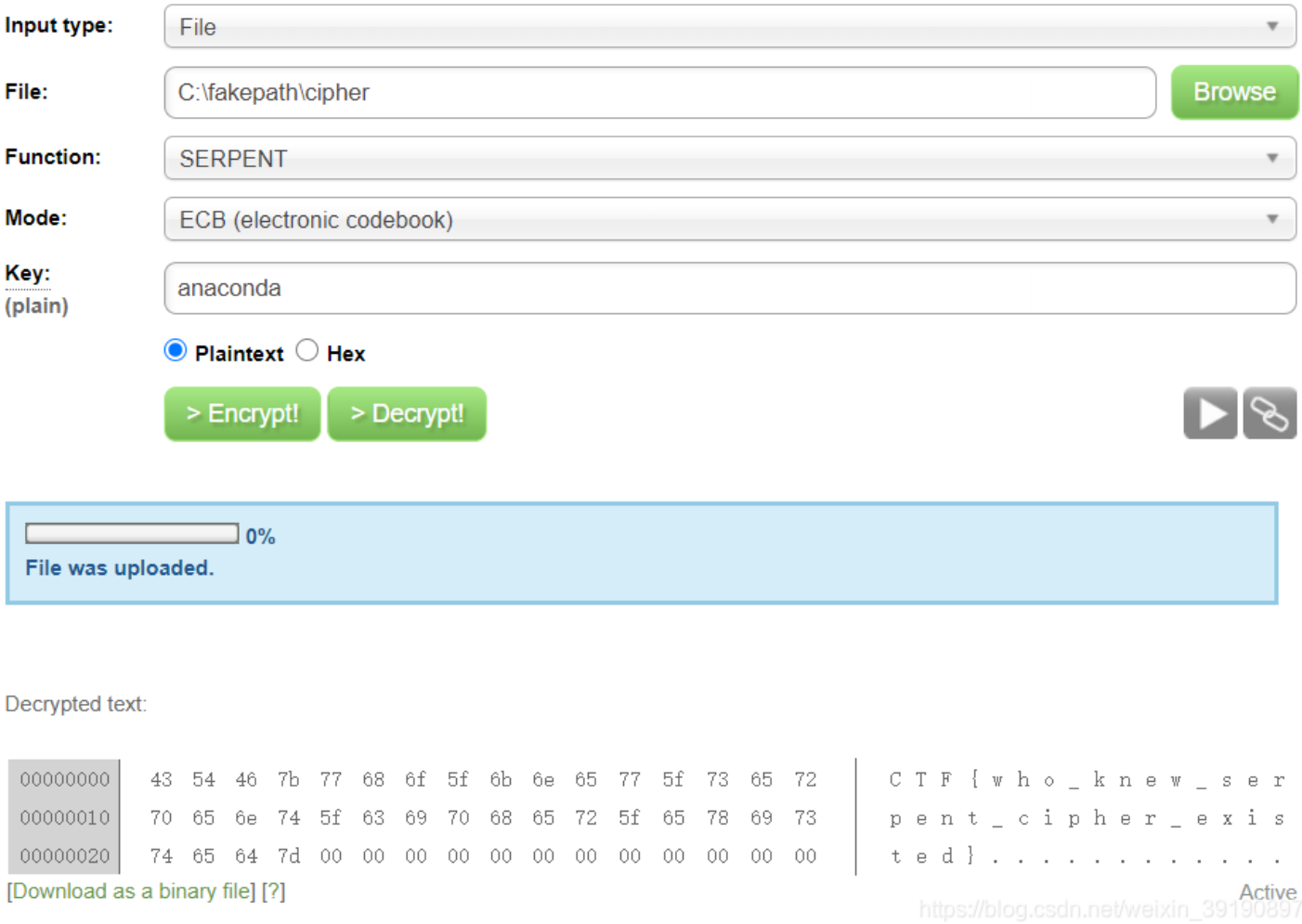


What is Nicki Minaj's favorite song that refers to snakes?

跟蛇有关，于是我们百度 Nicki Minaj 然后有她的百度百科，观看一下，发现：

Anaconda 是关于蛇的作品，这是不是我们的 key 呢？

于是我们用 Serpent 工具解密，Serpent 是一个加密算法（为何猜测是这种加密算法……不知道……），http://serpent.online-domain-tools.com/，我们用这个然后key是 Anaconda：



获得 flag：`flag{who_knew_serpent_cipher_existed}`。

No.19 Steghide隐写工具的使用

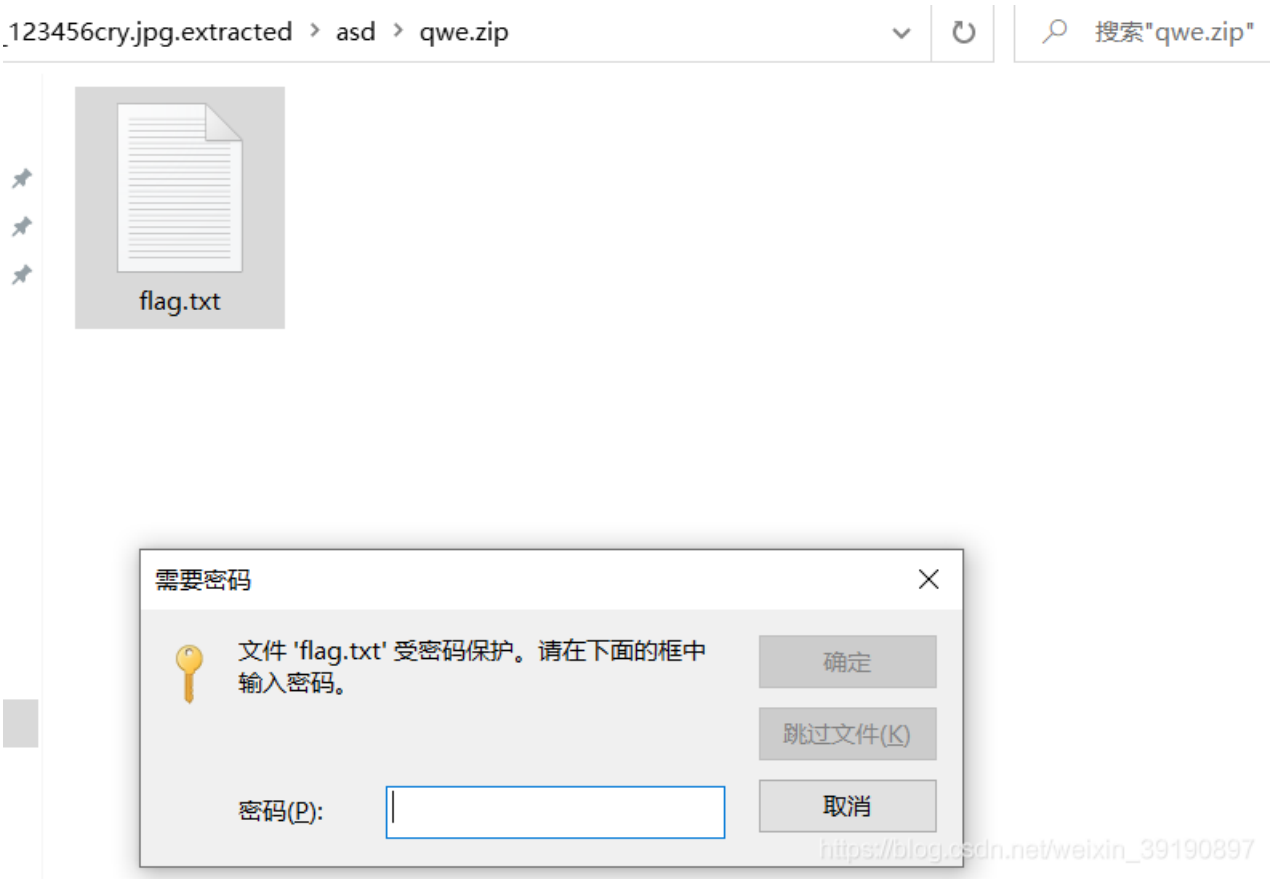
下载后是一张图片：

1、Binwalk 查看发先隐藏了一个 ZIP 文件，进行提取：

```
Cmder

C:\Python37\Scripts
λ python binwalk C:\Users\True\Downloads\389a0c11-d0df-4180-829a-b529e6b0a1bc\123456cry.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
19560         0x4C68          Zip archive data, at least v1.0 to extract, name: asd/
48454         0xBD46          Zip archive data, at least v1.0 to extract, compressed size: 184, uncompressed size: 184, name: asd/qwe.zip
48657         0xBE11          End of Zip archive, footer length: 22
48962         0xBF42          End of Zip archive, footer length: 22


C:\Python37\Scripts
λ python binwalk -e C:\Users\True\Downloads\389a0c11-d0df-4180-829a-b529e6b0a1bc\123456cry.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
19560         0x4C68          Zip archive data, at least v1.0 to extract, name: asd/
48454         0xBD46          Zip archive data, at least v1.0 to extract, compressed size: 184, uncompressed size: 184, name: asd/qwe.zip
48657         0xBE11          End of Zip archive, footer length: 22
48962         0xBF42          End of Zip archive, footer length: 22


C:\Python37\Scripts
λ
```


2、提取后发现两个文件：

3、直接打开 qwe.zip 里面的文件发现需要密码，那大概率就是回到另一个图片`good-已合并.jpg`进行信息提取了：



4、使用 Binwalk 查看 `good-已合并.jpg` 是否由多个文件组合而成，无果：

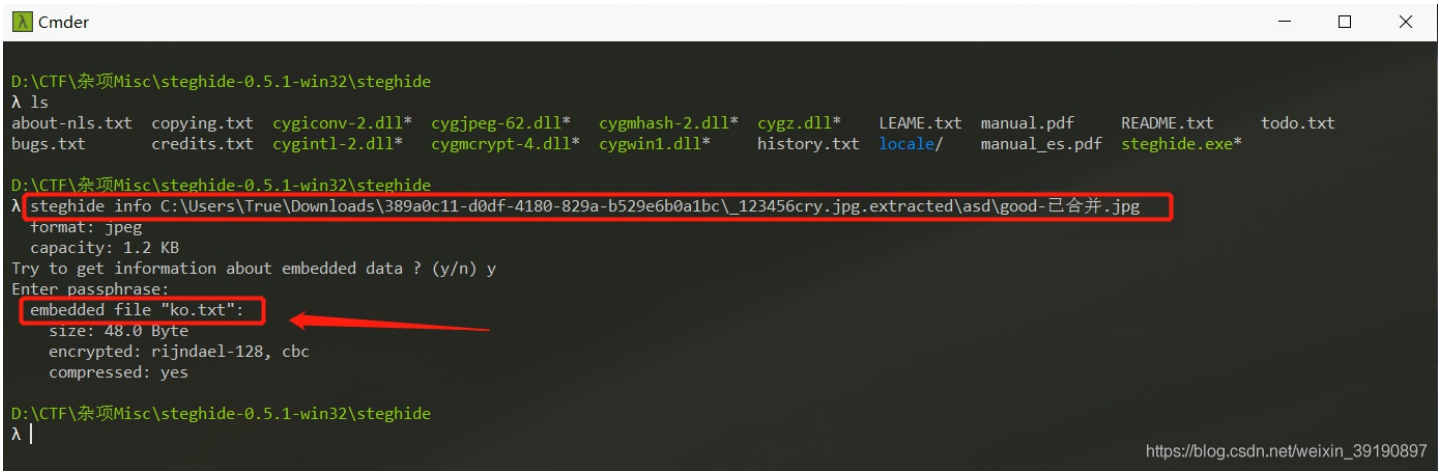
```
Cmder

C:\Python37\Scripts
λ python binwalk C:\Users\True\Downloads\389a0c11-d0df-4180-829a-b529e6b0a1bc\_123456cry.jpg.extracted\asd\good-已合并.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01


C:\Python37\Scripts
λ
```

5、此时只能用尝试用隐写工具 **Steghide**（官方下载地址）进行信息提取。这是一个可以将文件隐藏到图片或音频中的工具，其使用如下：

目的	命令
Liunx安装	apt-get install steghide
隐藏文件	steghide embed -cf 1.jpg（图片文件载体） -ef 1.txt（待隐藏文件）-p 123456（-p参数用于添加密码，非必选）

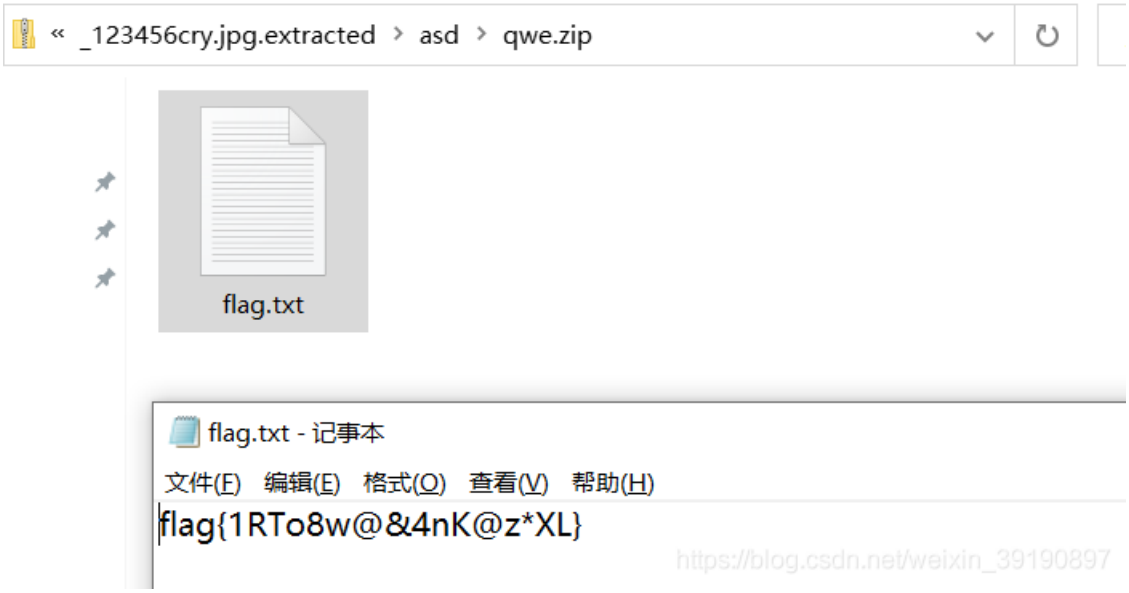
目的	命令
查看图片中嵌入的文件信息	steghide info 1.jpg
提取图片中隐藏的文件	steghide extract -sf 1.jpg -p 123456

我下载的是 Windows 版本（懒得打开虚拟机…），其使用方法为命令行 cd 到 `steghide.exe` 所在的文件夹后，与 Linux 版本的参数和命令结构都差不多，也是无密码直接回车就好：

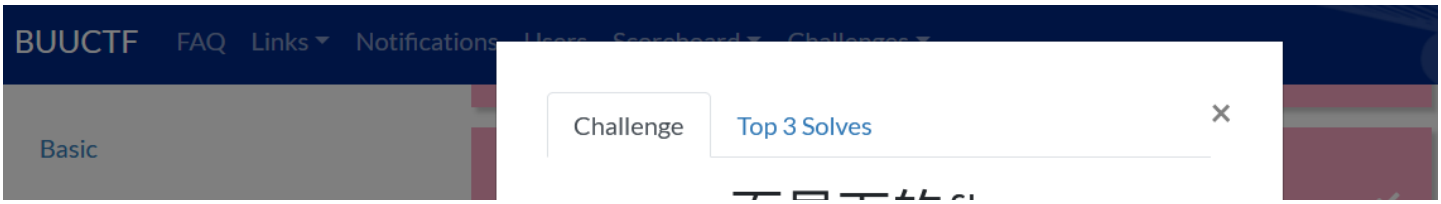


6、上面已经发现图片中隐藏了 ko.txt 文件了，进一步进行信息提取：

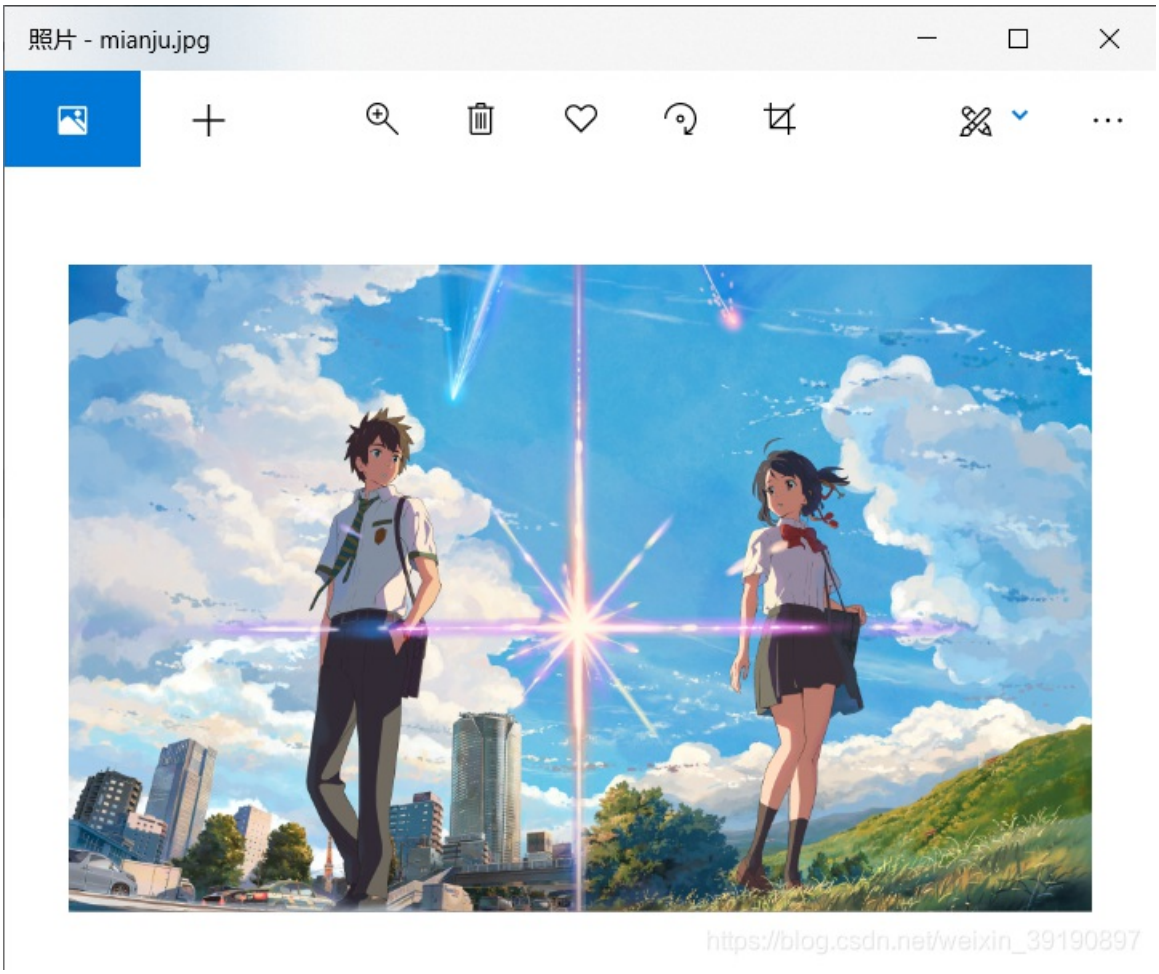
7、拿着上述压缩包密码解密 flag.txt，获得 flag：



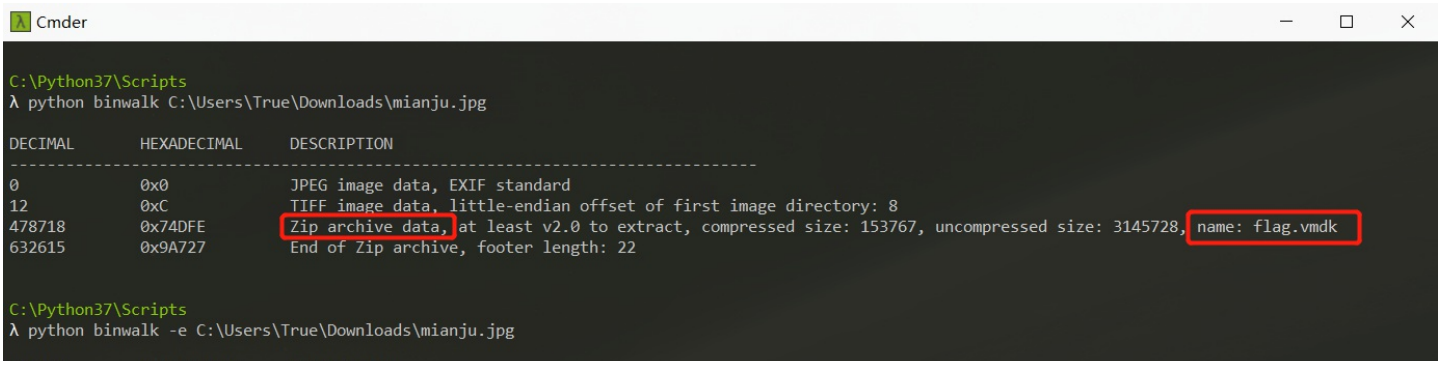
No.20 ZIP伪加密与多重编码转换



下载后是一张图片：

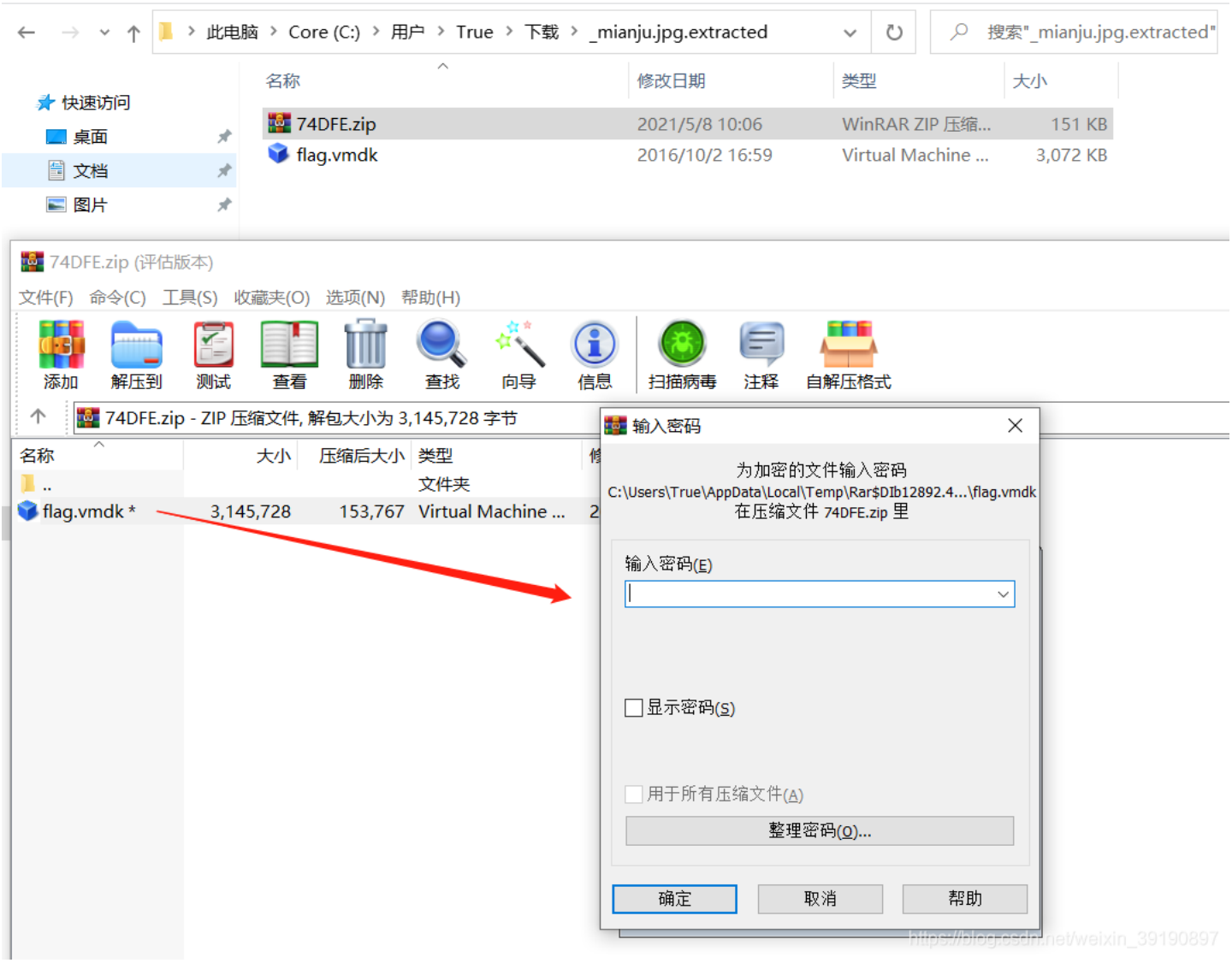


1、使用 Binwalk 发现隐藏着文件，将其分离出来：

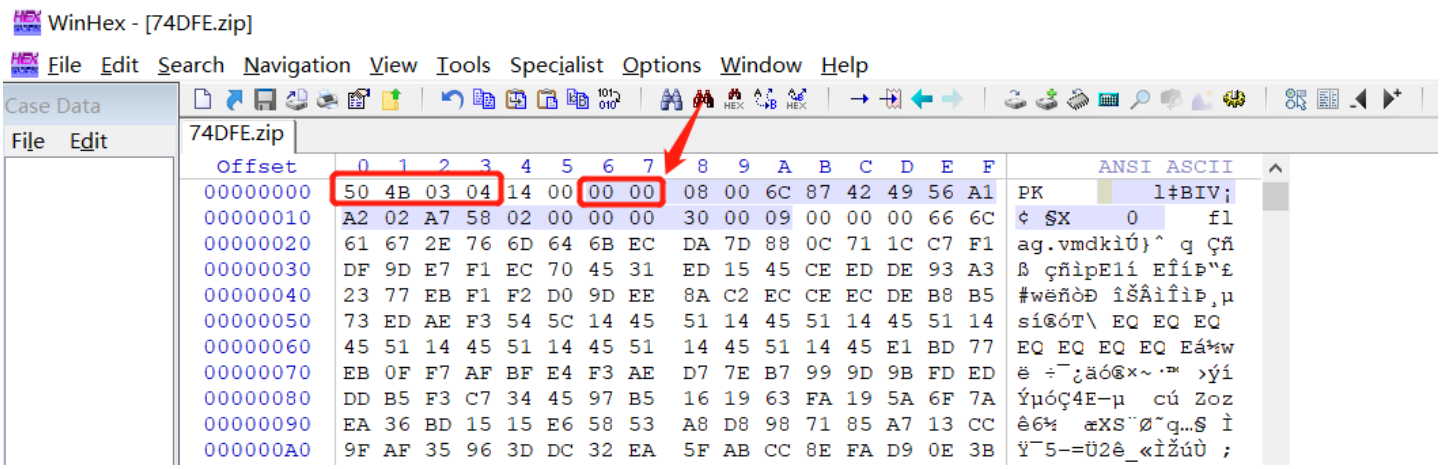


```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, EXIF standard
12            0xC             TIFF image data, little-endian offset of first image directory: 8
478718        0x74DFE         Zip archive data, at least v2.0 to extract, compressed size: 153767, uncompressed size: 3145728, name: flag.vmdk
632615        0x9A727         End of Zip archive, footer length: 22


C:\Python37\Scripts
λ
```


2、分离出来的 ZIP 文件需要密码：



3、使用 Winhex 打开 ZIP 压缩文件，发现是 ZIP 伪加密（CTF杂项-ZIP伪加密与Base64隐写实战），如图可见压缩源文件数据区的全局加密为 `00 00`：

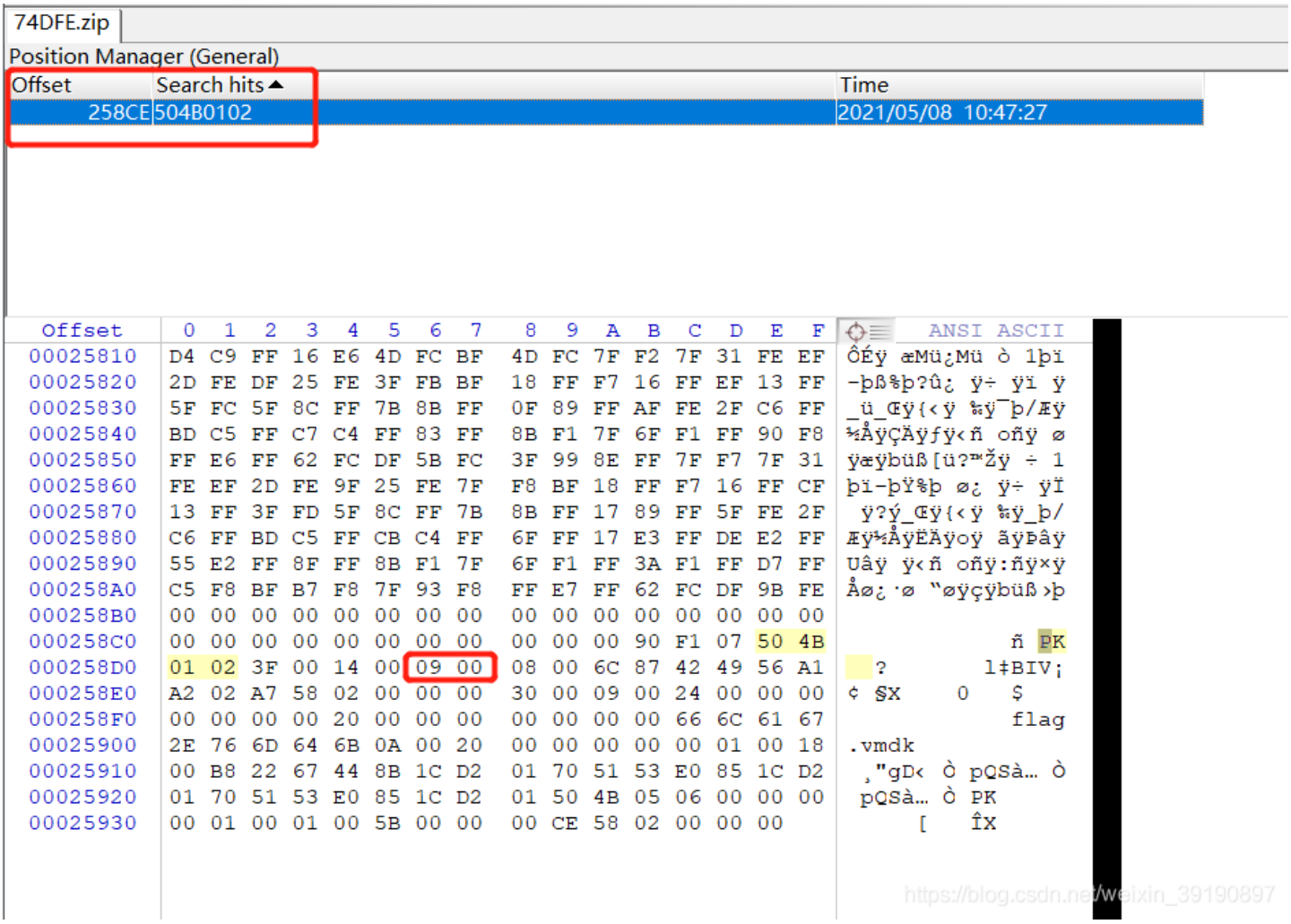


```
000000B0  EA 65 E3 19 BF 33 17 64  16 FA 29 CF EA F2 32 59  êeã ¿3 d ú)Ïêò2Y
000000C0  3F 48 D7 87 2D 2F 1D 0F  5C 3F 9D AC 0F 2D 6A 68  ?H×‡-/  \? ¬ -jh
000000D0  0A 59 8D 4B A2 F5 61 D7  A9 AB AE 4B 78 56 A7 93   Y K¢õa×©«®KxV§"
000000E0  F1 D2 B9 FC A6 44 21 CB  CF 2E 77 72 7E 97 D7 92  ñÒ¹ü¦D!ËÏ.wr~—×'
```

真假加密识别

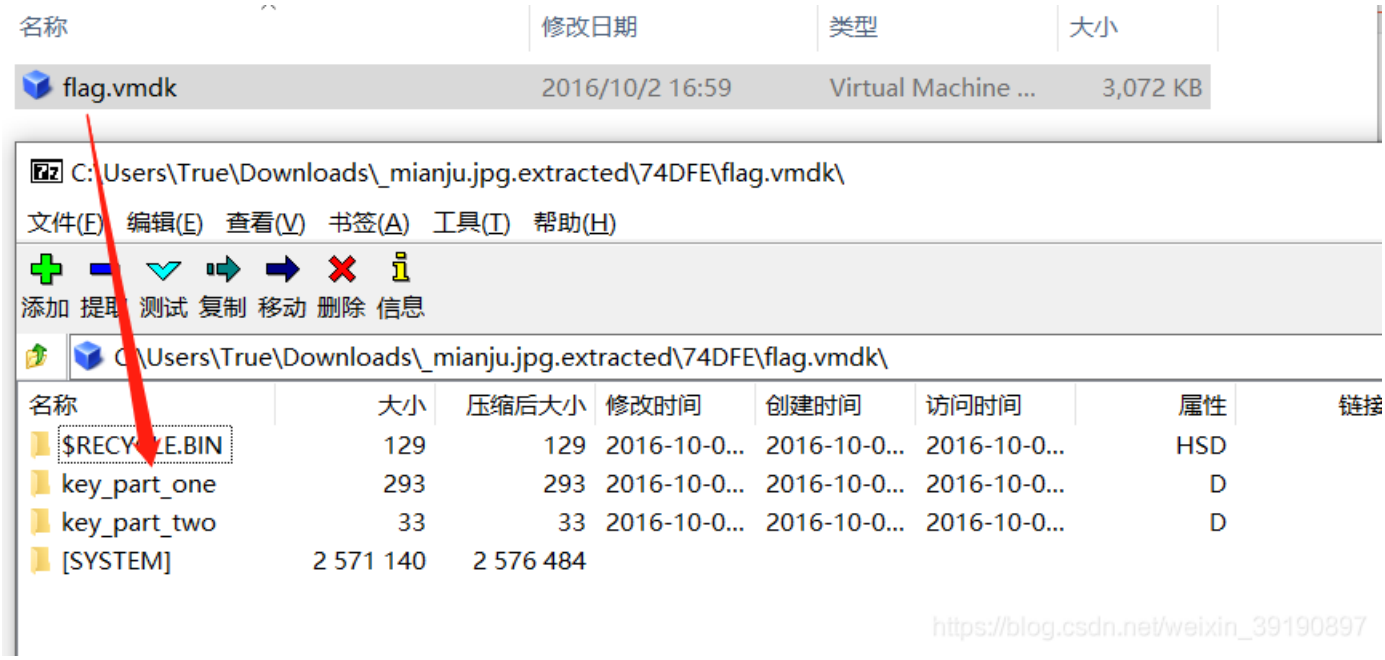
解密情况	识别方法
无加密	压缩源文件数据区的全局加密应当为`00 00`；且压缩源文件目录区的全局方式位标记应当为`00 00`
假加密	压缩源文件数据区的全局加密应当为`00 00`；且压缩源文件目录区的全局方式位标记应当为`09 00`
真加密	压缩源文件数据区的全局加密应当为`09 00`；且压缩源文件目录区的全局方式位标记应当为`09 00`

搜索十六进制数值`504B0102`（压缩源文件目录区头标志），发现全局方式位标记为`09 00`，断定为伪加密：

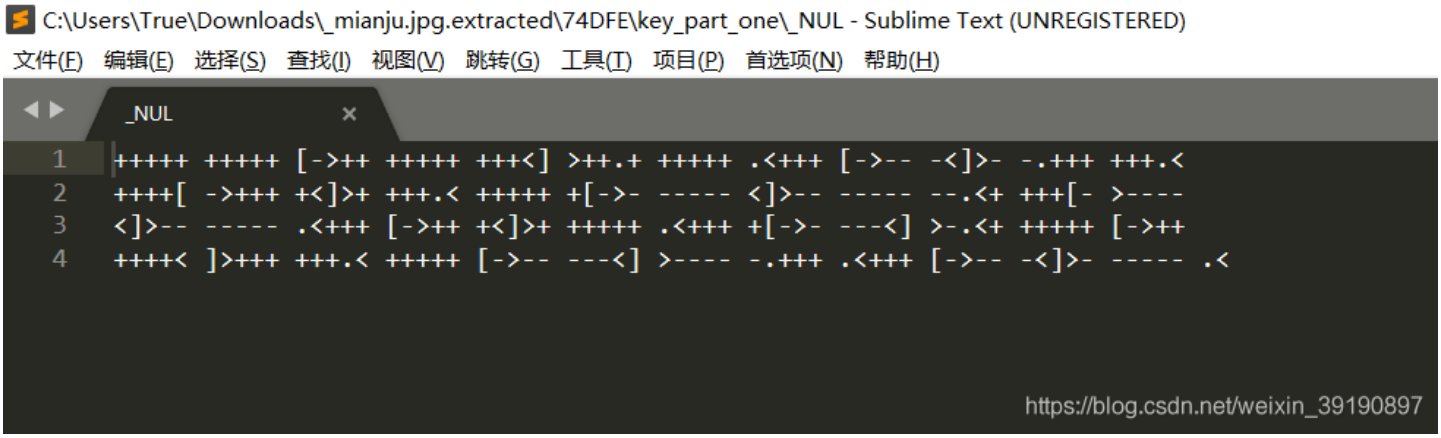


4、修改`09 00`为`00 00`后保存，即可解压缩该伪加密的 ZIP 文件，获得如下文件：

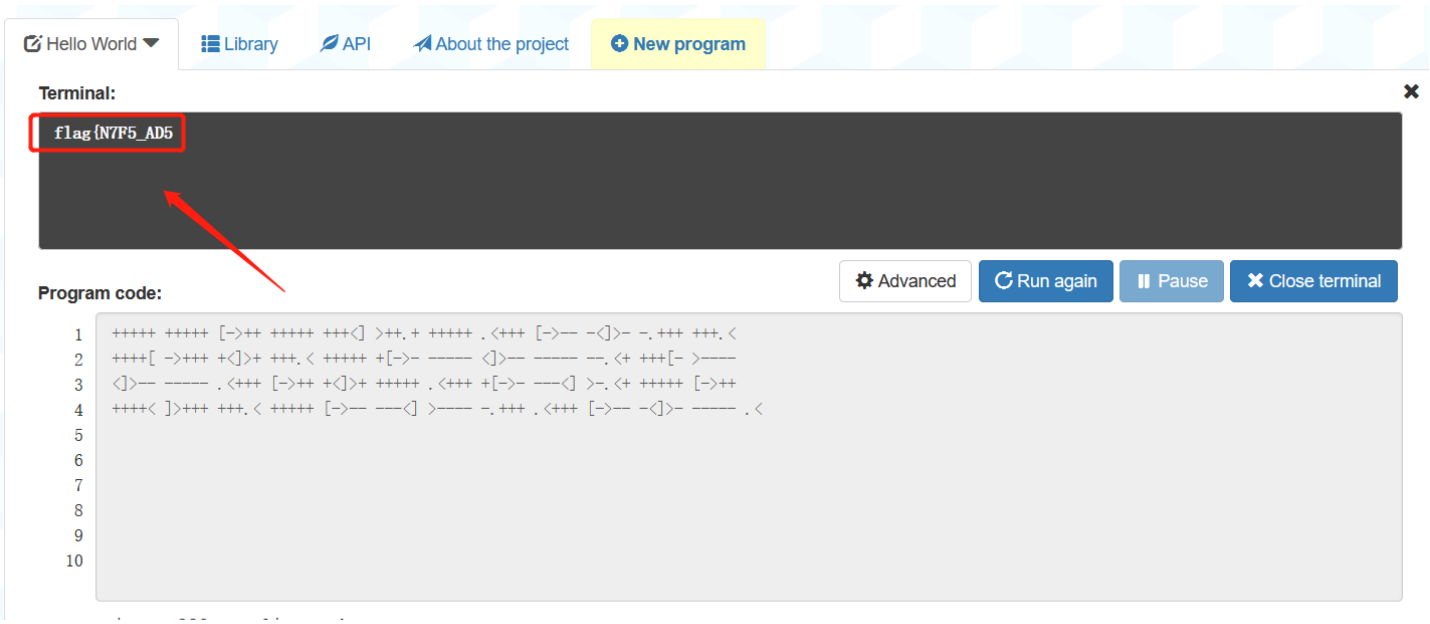
5、对于 vmdk 文件，需要使用 `7-ZIP` 工具解压缩（官网下载地址），安装后打开后压缩包，如下：



6、打开 Key_part_one 文件夹里面的文件：



`Brainfuck` 编码，Brainfuck在线解密：



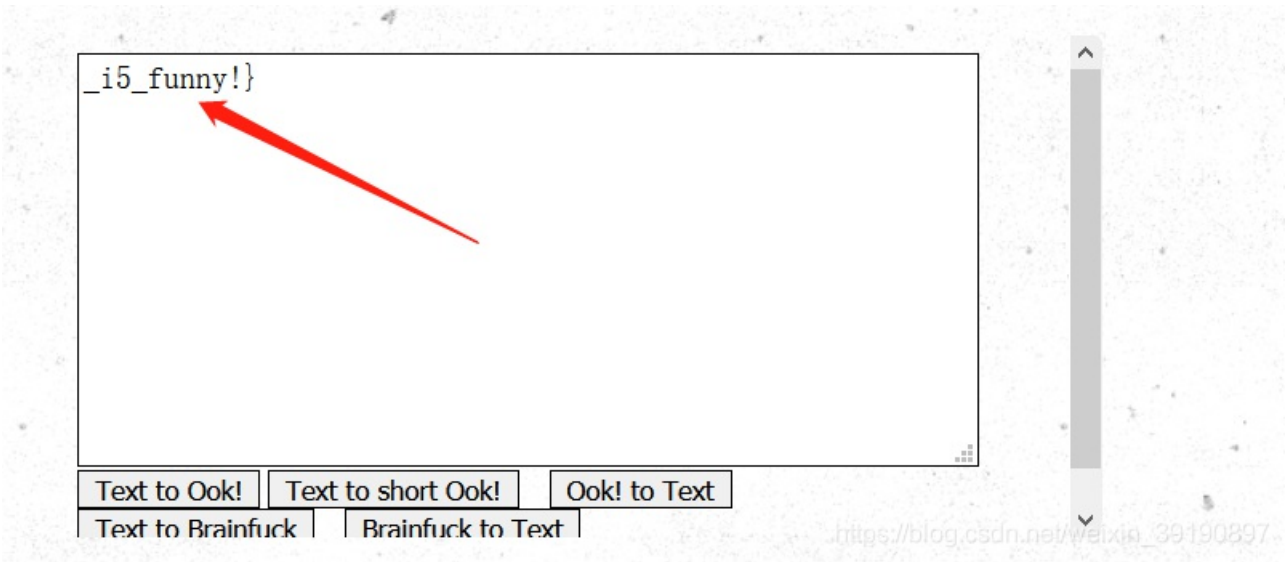
size: 290 lines: 4

More Run program

7、打开 Key_part_one 文件夹里面的文件：

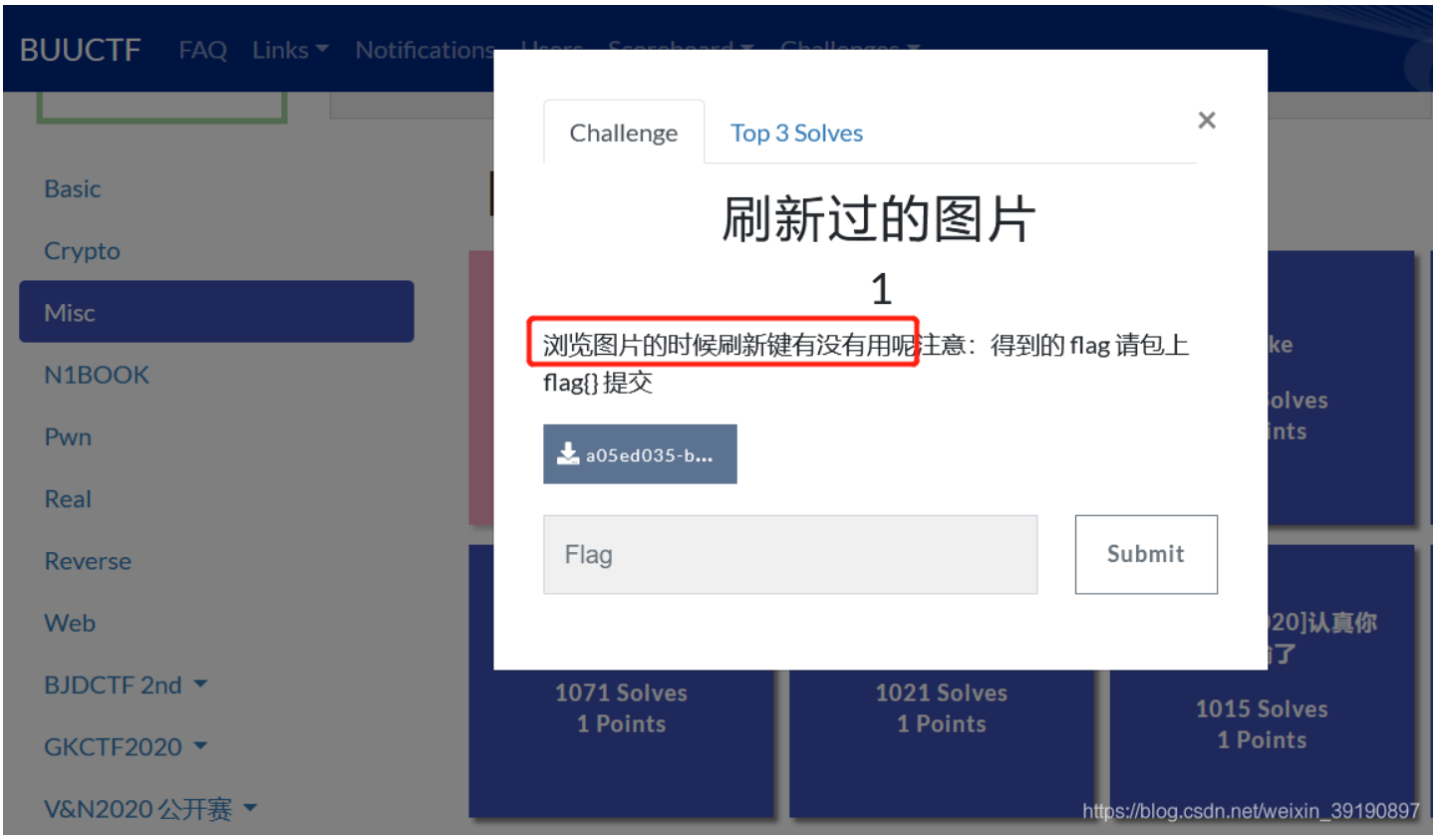
```
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook?
Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook!
Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook.
Ook. Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook! Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook? Ook. Ook? Ook! Ook. Ook?
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook.
Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook!
Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook?
Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook! Ook. Ook?
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook.
Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook!
Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
Ook? Ook. Ook? Ook! Ook. Ook? Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook!
Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook! Ook. Ook? Ook.
```

`Ook!` 编码，在线解密：

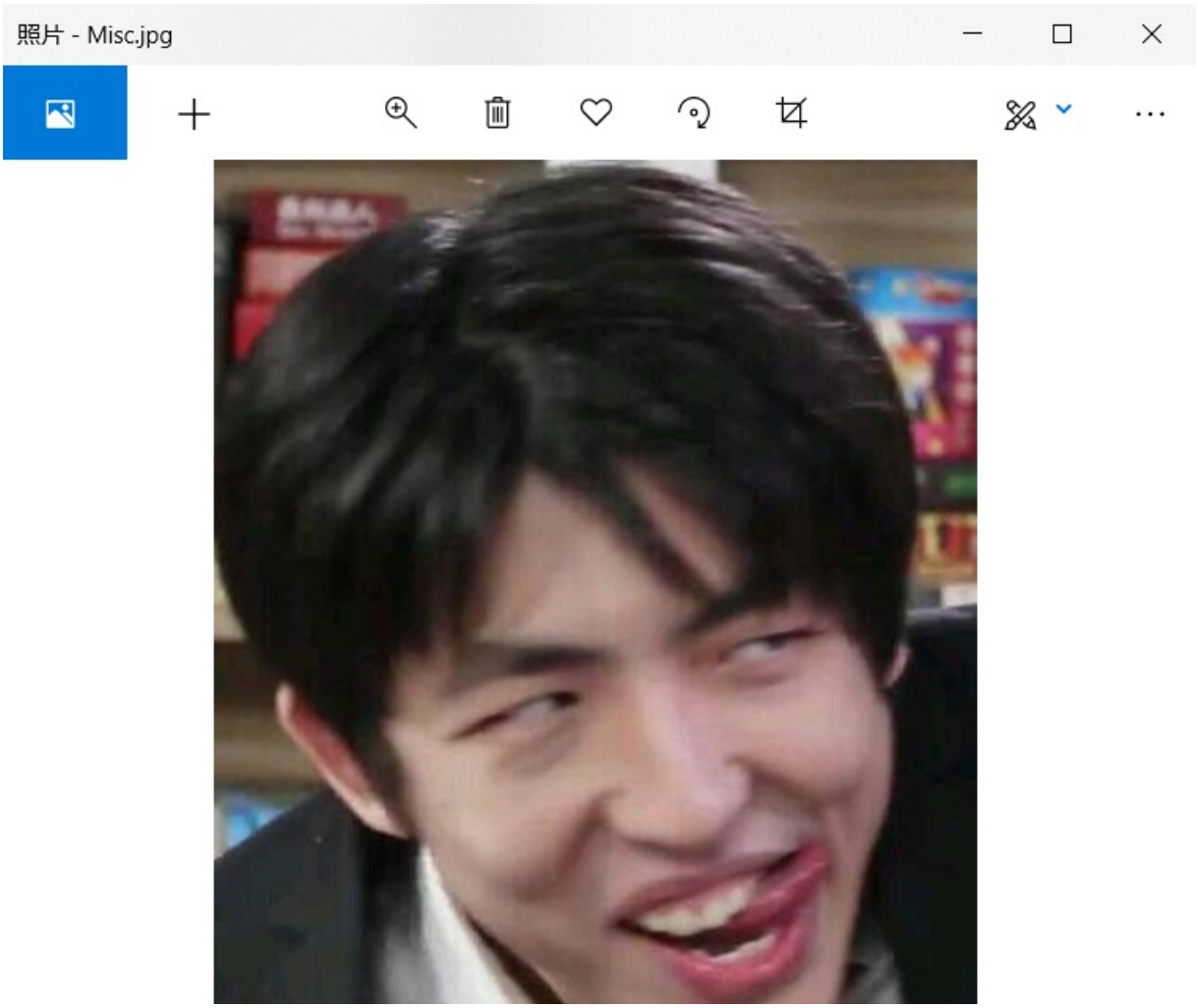


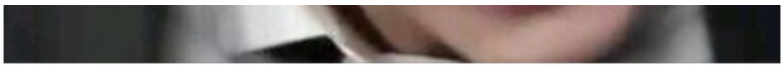
拼接获得 flag：`flag{N7F5_AD5_i5_funny!}`。

No.21 F5隐写与ZIP文件头的识别

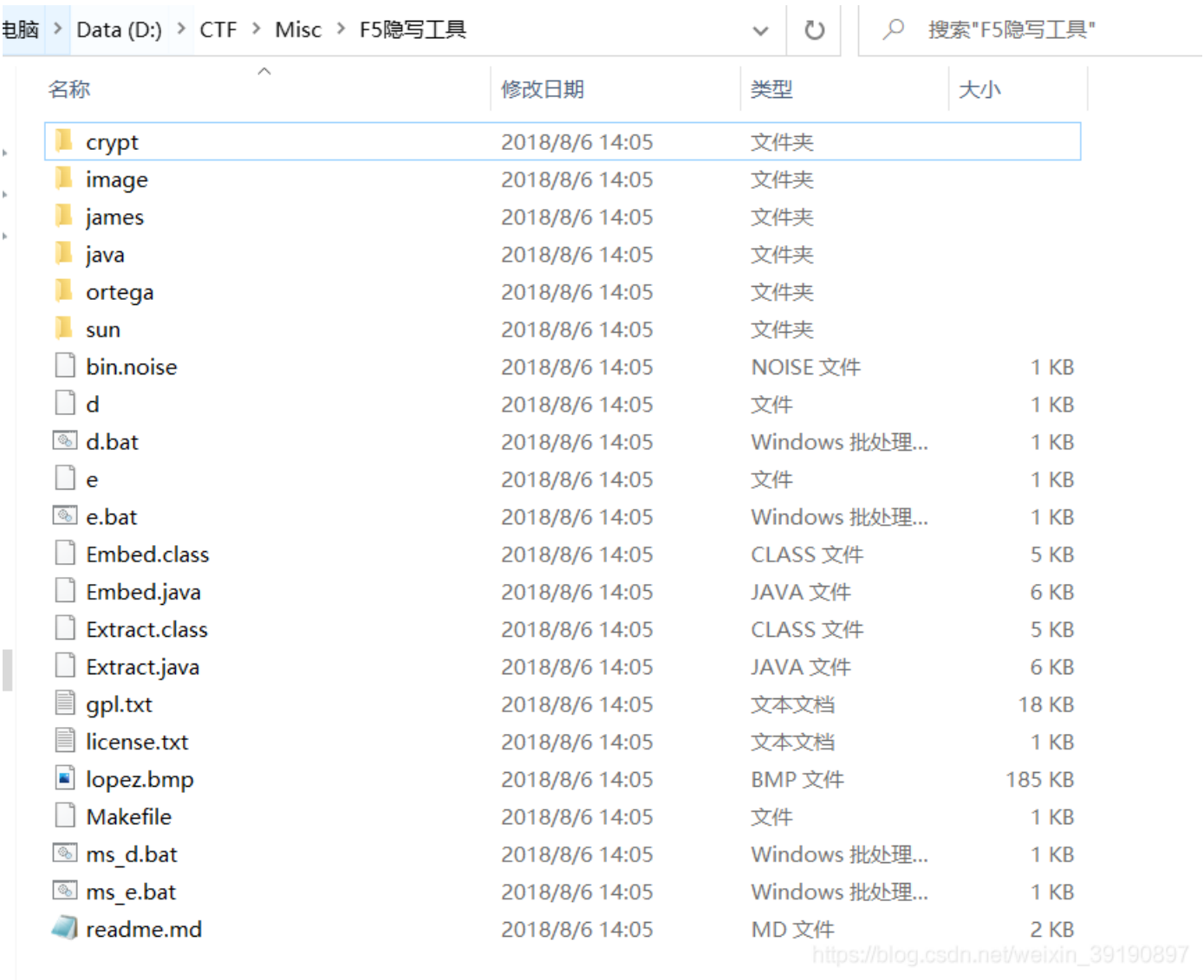


下载后是一张图片：

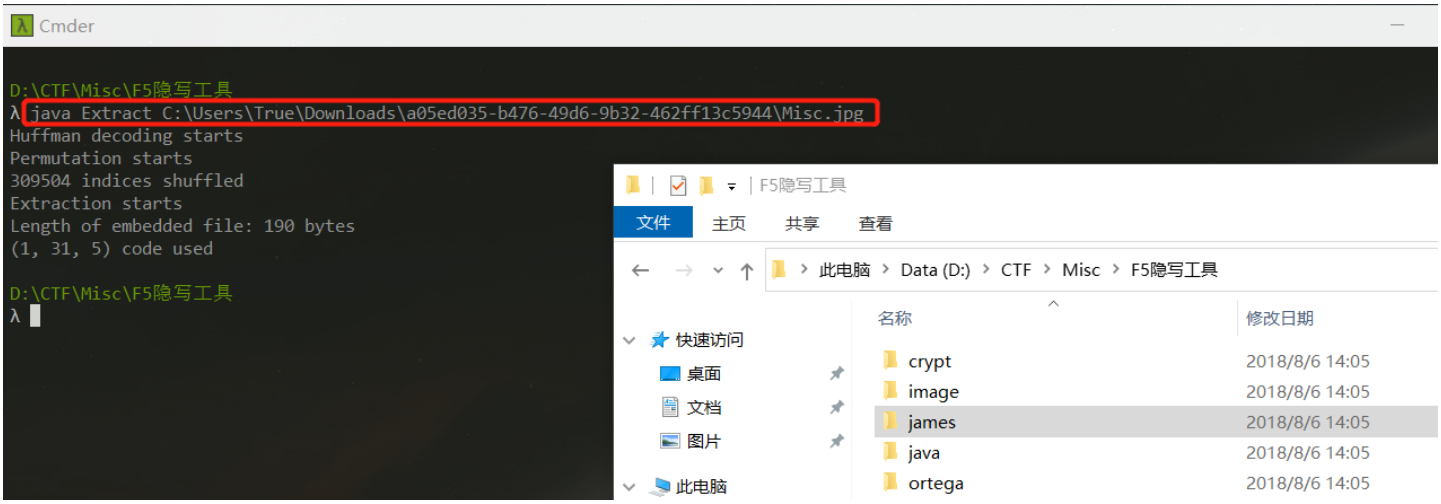


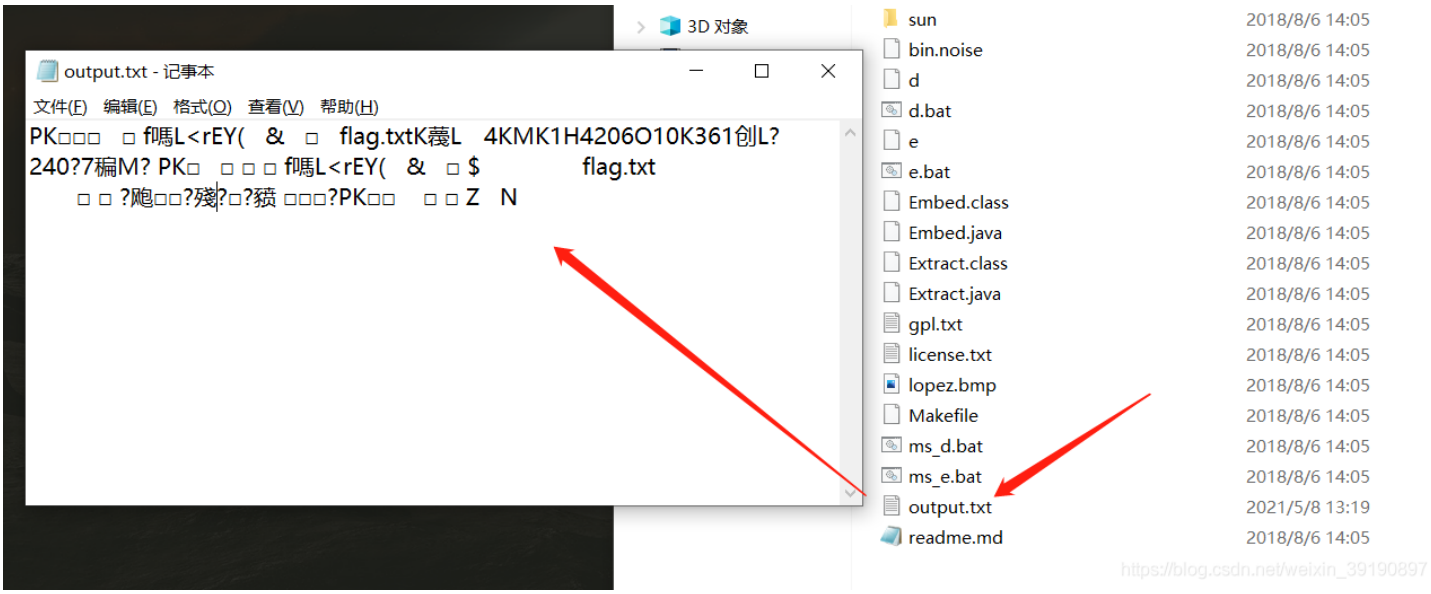


使用各种解密软件（Stegsolve、steghide、binwalk）都没有找到问题所在，回到题目提示：“浏览图片的时候刷新键有没有用呢”，刷新，F5？？？在百度的帮助下了解了 F5 隐写（小白啊，真的蒙了）……还真有！F5 隐写的原理可参见： F5隐写算法及其隐写分析研究，此处直接上 F5 隐写利用工具（F5-steganography的Github地址）：

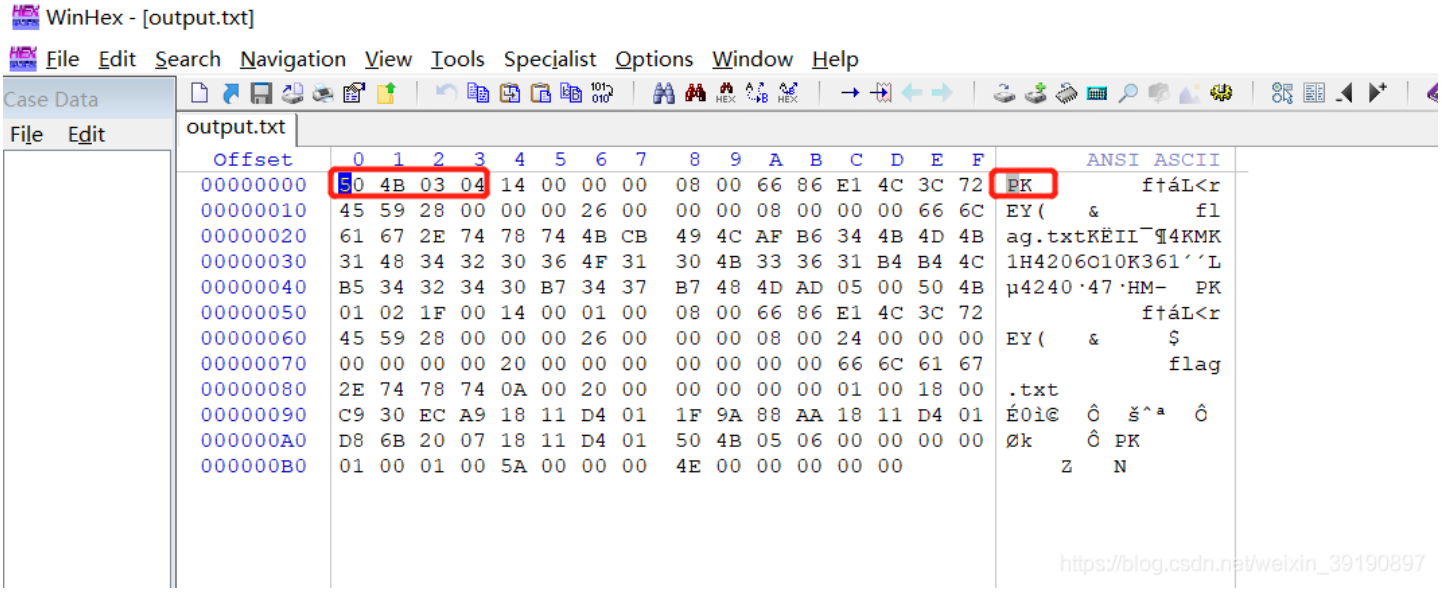


1、在该工具的文件夹路径下执行命令 `java Extract 图片路径` 即可在输出 `output.txt` 文件记录了图片隐写的信息：

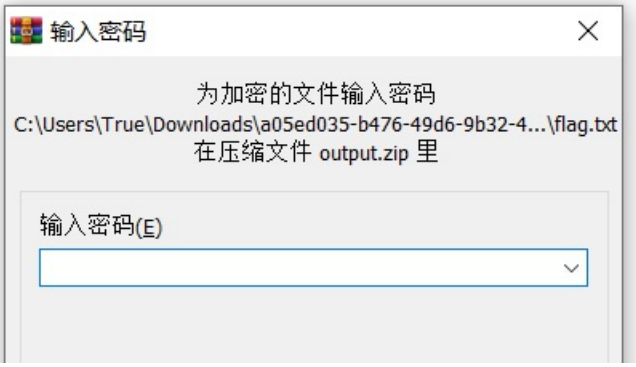


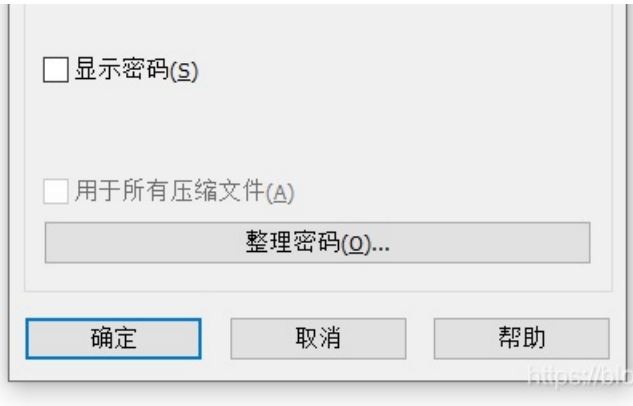


2、使用 Winhex 打开 output.txt 发现 ZIP 文件的文件头（实际上根据打开的 txt 文件的 `PK` 首部字符也可猜测是 ZIP 文件：

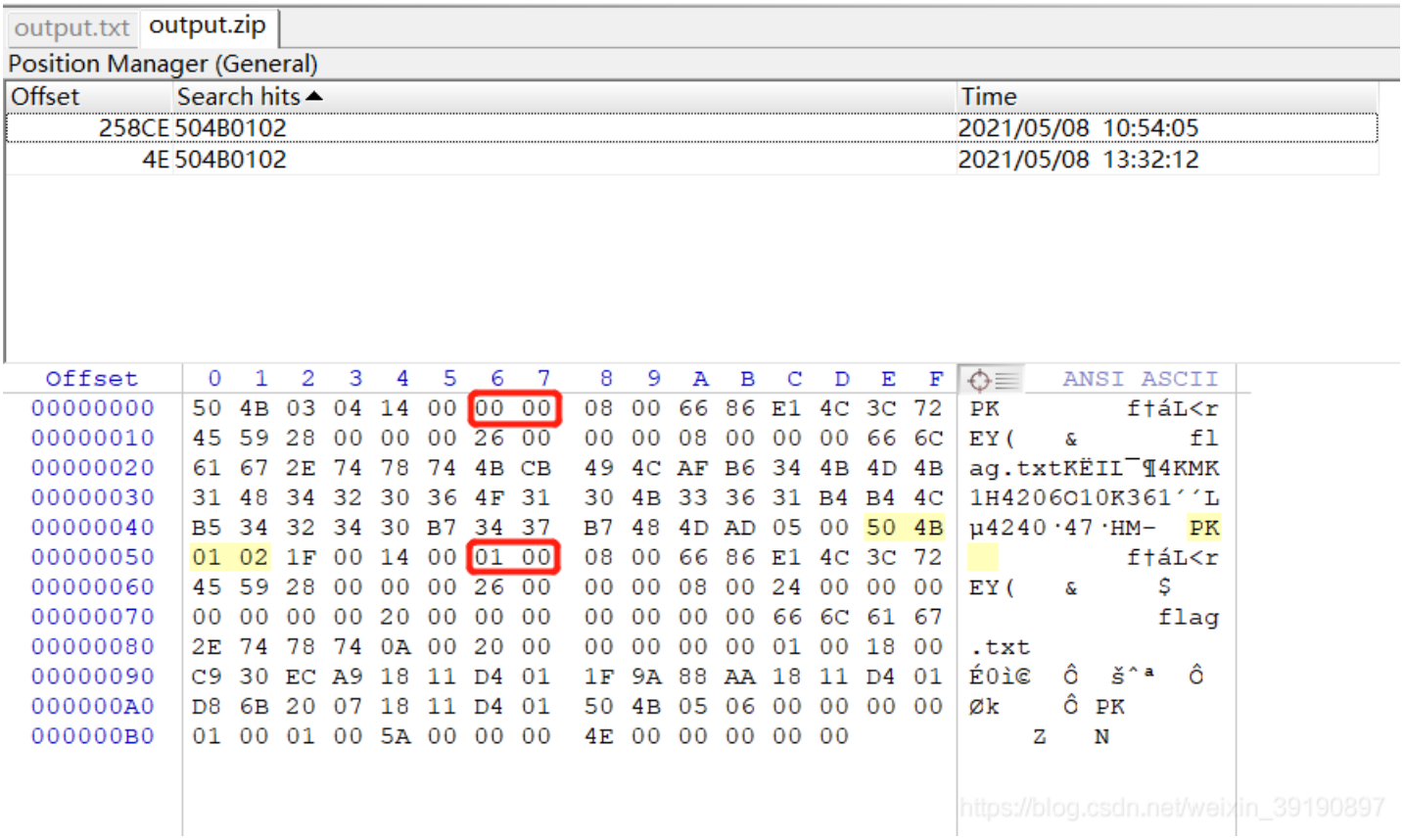


3、故修改 output.txt 后缀获得 output.zip 压缩文件，解压发现加密：



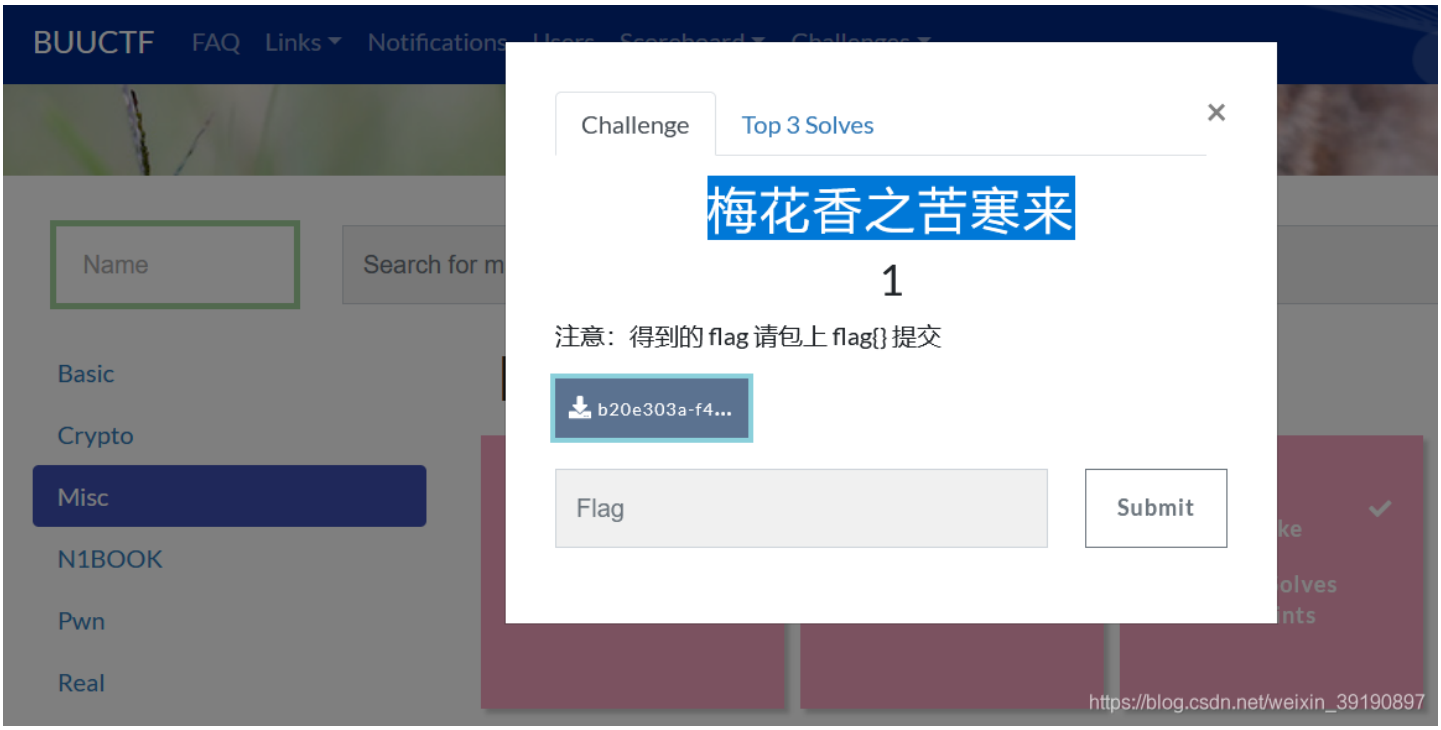


4、伪加密，将 `01 00` 改为 `00 00` 后保存，即可解压：



5、解压缩文件后获得 flag：

No.22 Py脚本16进制坐标绘二维码



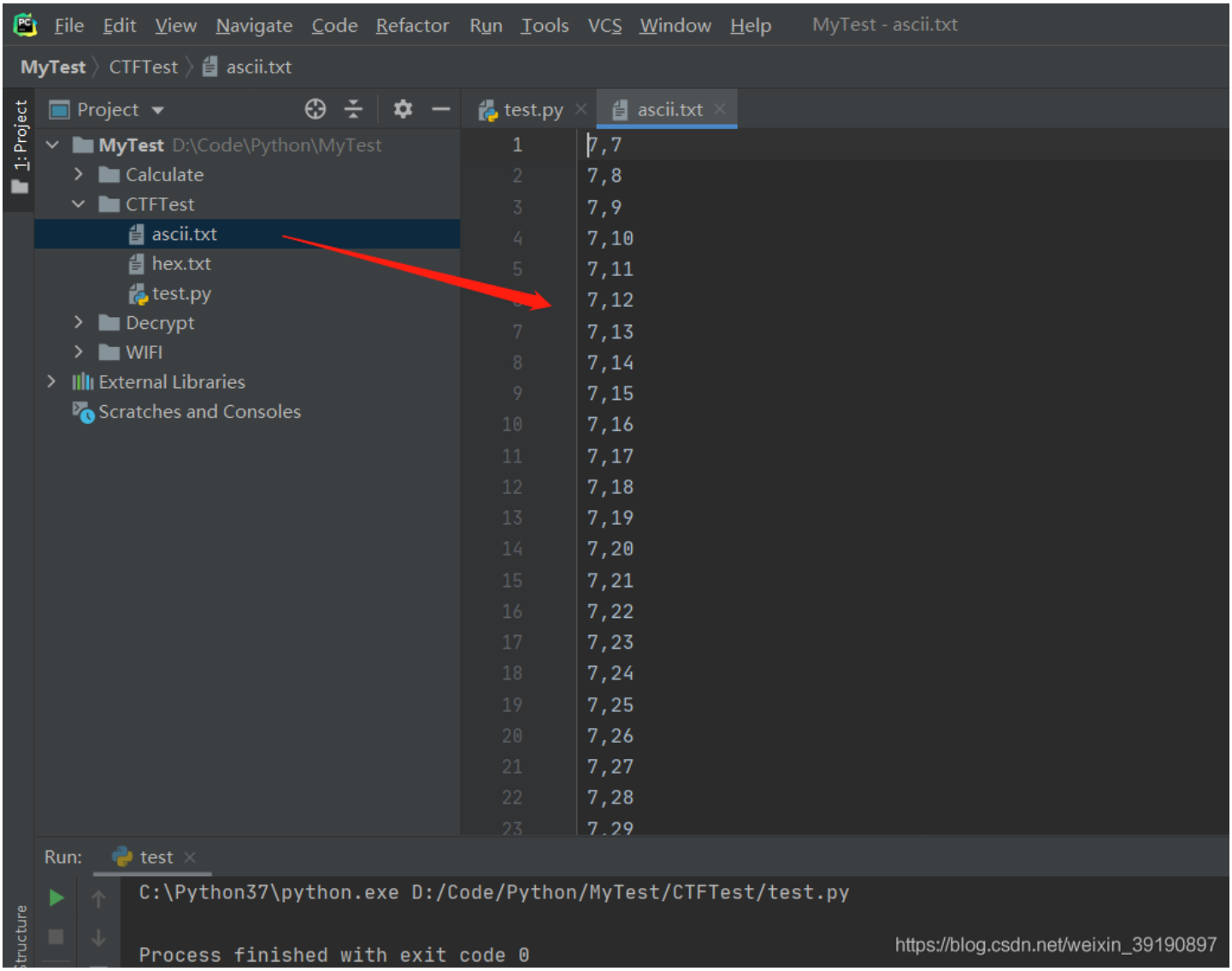
下载后是一张图片：

1、编辑器或者 winhex 打开图片，发现末尾存在大量十六进制：

2、将 16 进制数据复制到 hex.txt 利用脚本转化为 ascii 获取坐标点：

```
with open('hex.txt', 'r') as h:
    h = h.read()
with open('./ascii.txt', 'a') as a:
    for i in range(0, len(h), 2):
        tmp = '0x'+h[i]+h[i+1]
        tmp = int(tmp, base=16)
        if chr(tmp) != '(' and chr(tmp) != ')':
            a.write(chr(tmp))
```

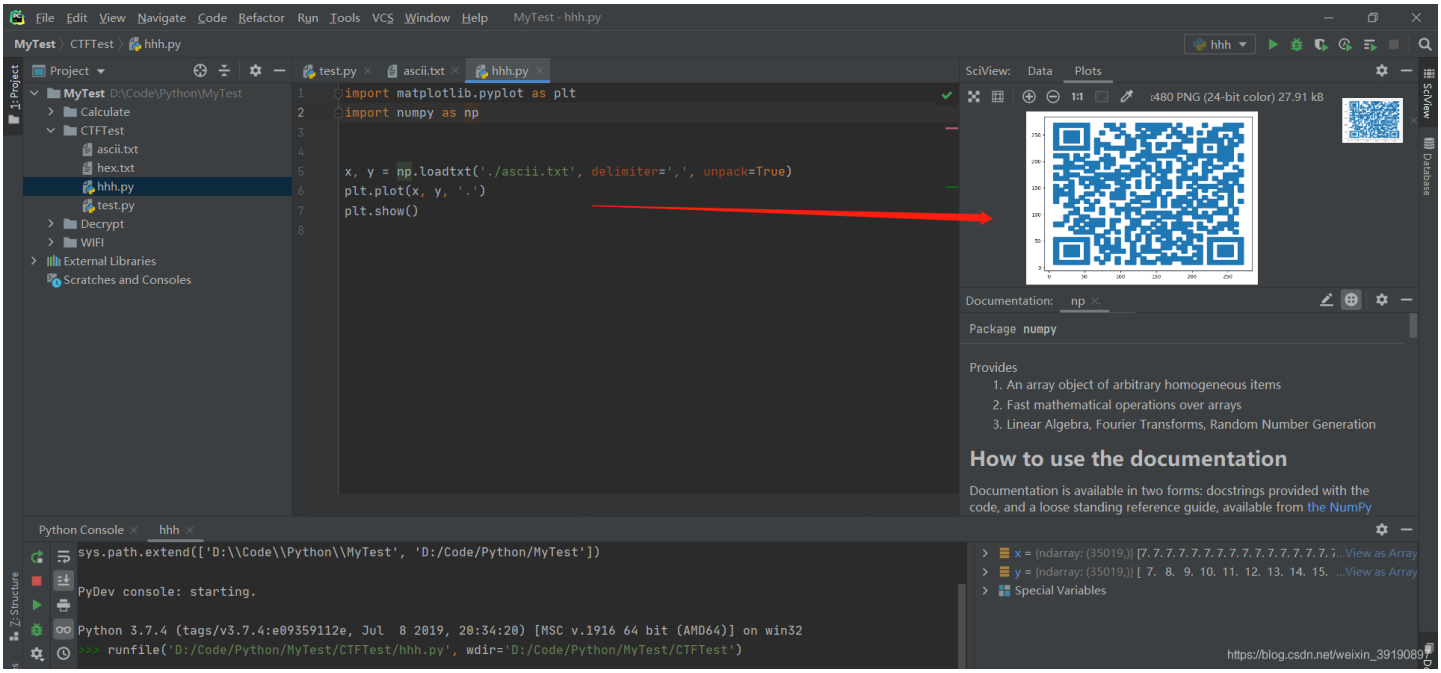
执行脚本：



3、用 matplotlib 绘图得到二维码：

```
import matplotlib.pyplot as plt
import numpy as np

x, y = np.loadtxt('./ascii.txt', delimiter=',', unpack=True)
plt.plot(x, y, '.')
plt.show()
```

生成二维码：

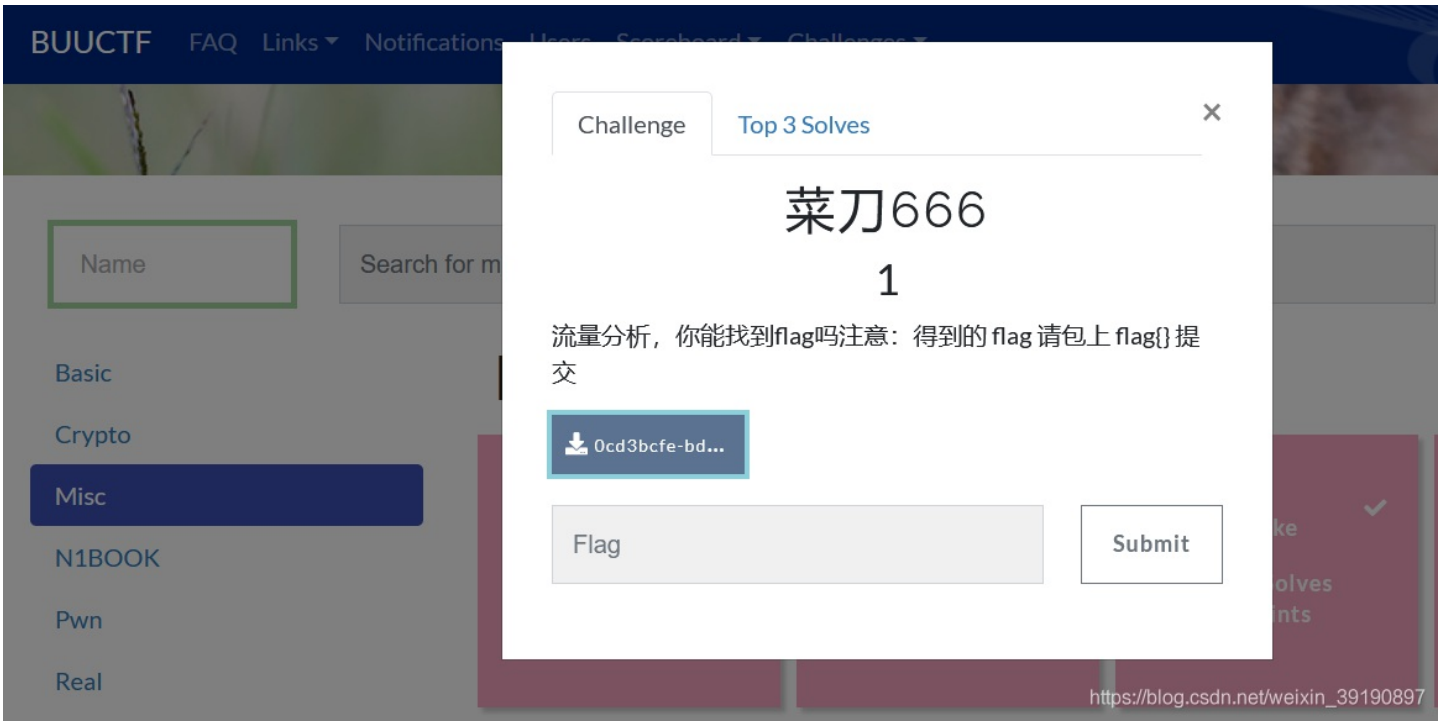


扫描二维码获得 flag：

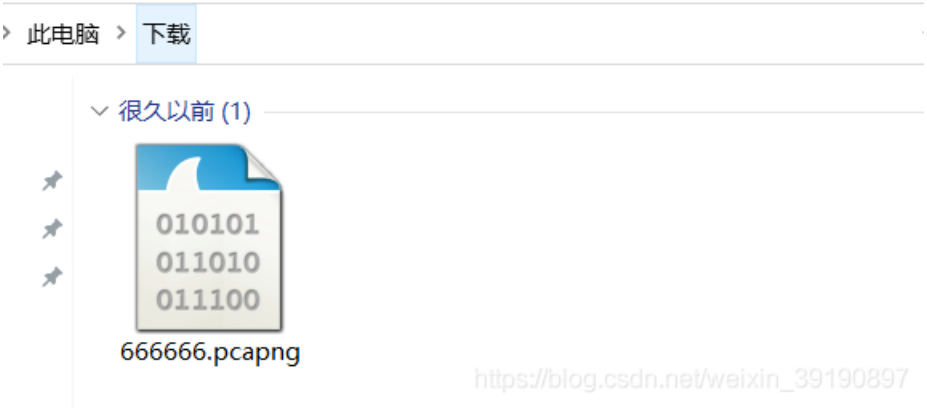
最终脚本：

```
import matplotlib.pyplot as plt
import numpy as np

with open('hex.txt', 'r') as h:
    h = h.read()
with open('./ascii.txt', 'a') as a:
    for i in range(0, len(h), 2):
        tmp = '0x'+h[i]+h[i+1]
        tmp = int(tmp, base=16)
        if chr(tmp) != '(' and chr(tmp) != ')':
            a.write(chr(tmp))


x, y = np.loadtxt('./ascii.txt', delimiter=',', unpack=True)
plt.plot(x, y, '.')
plt.show()
```

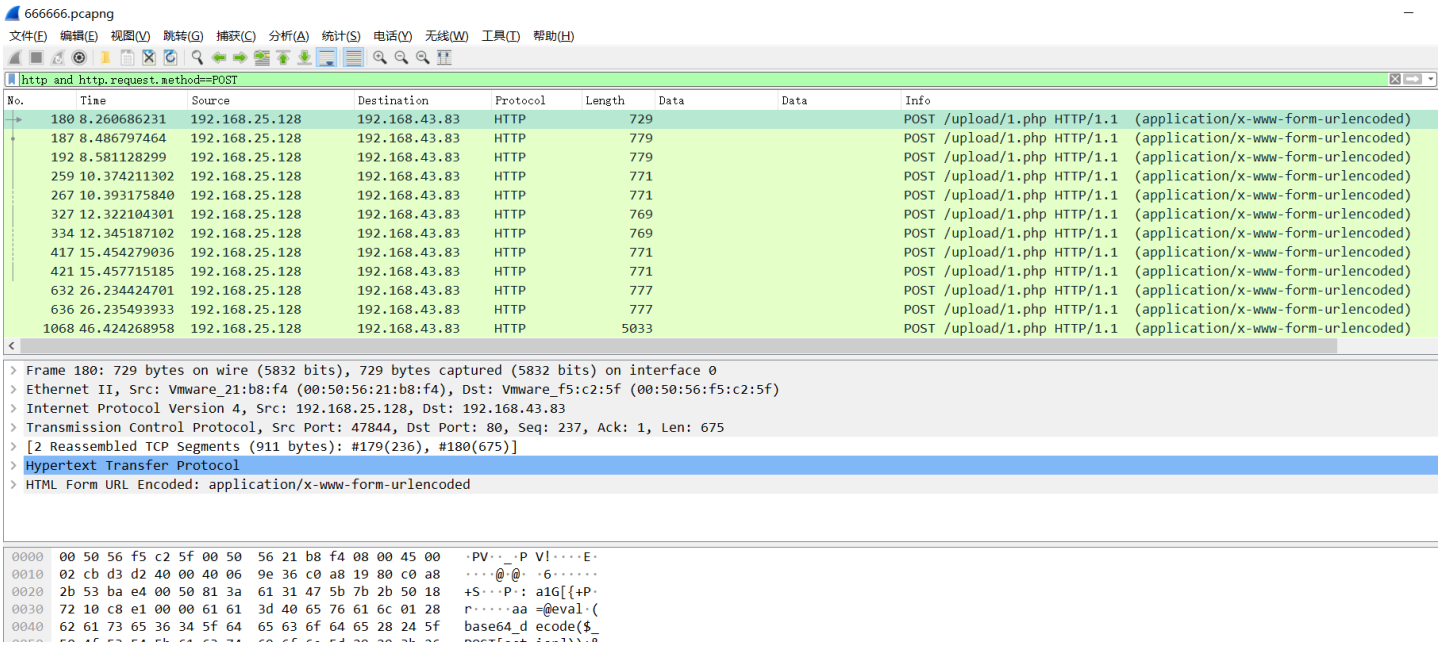
No.22 HTTP流量分析与文件转换

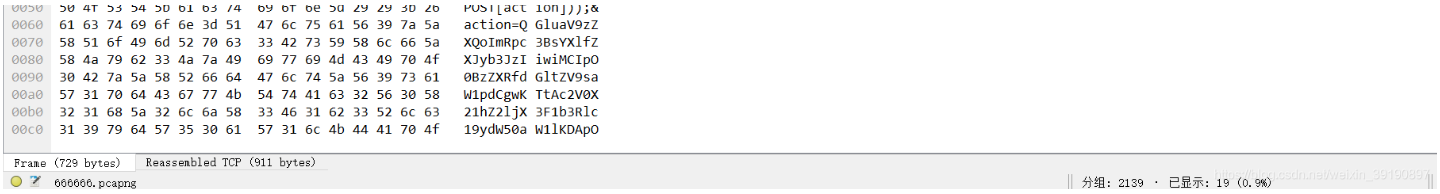


下载后是一个流量文件：

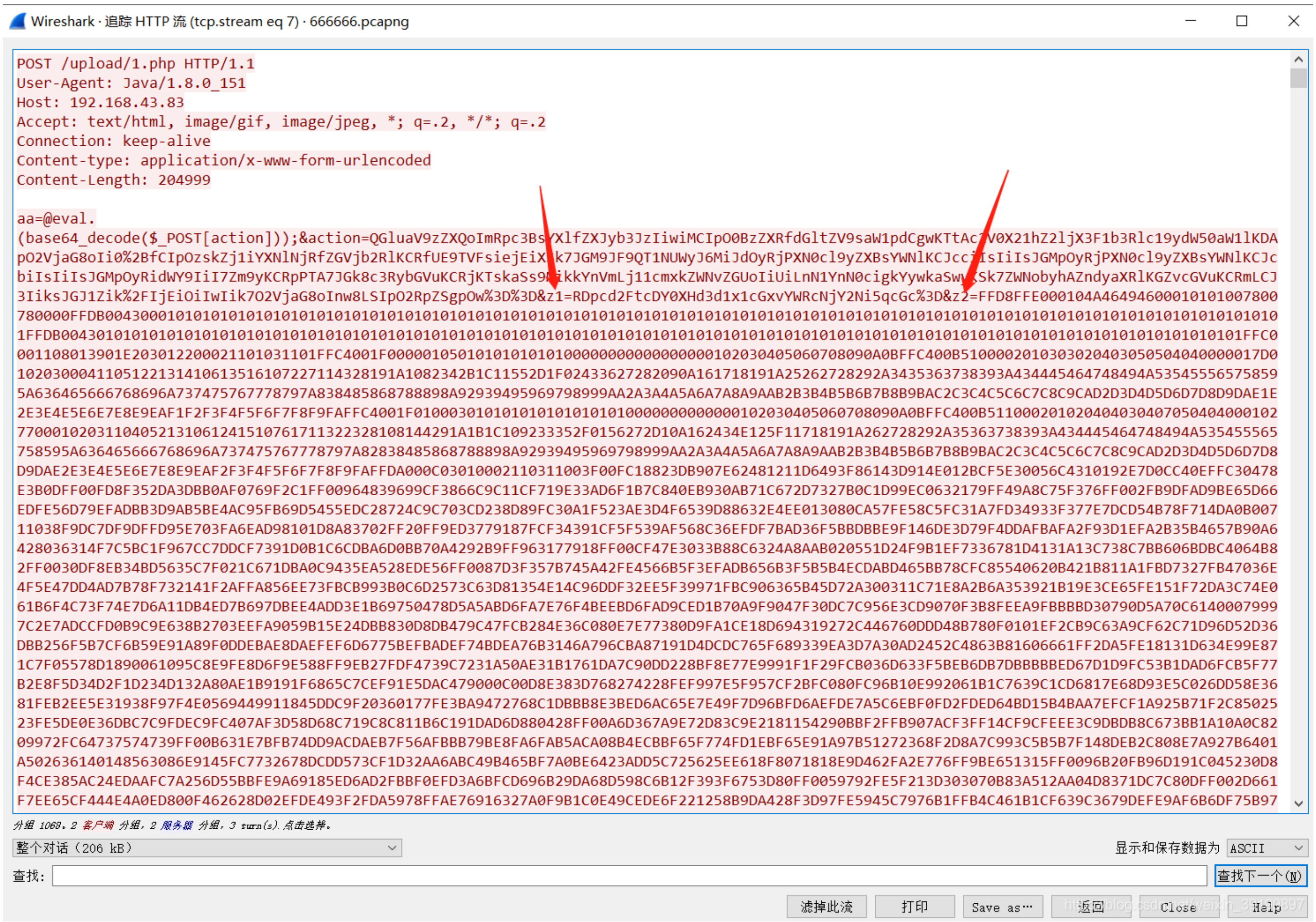


1、直接过滤出 HTTP 流量，题目因为是菜刀吗？一般都是 post 连接，于是我们过滤 post 数据 `http.request.method==POST`，然后分析流量：

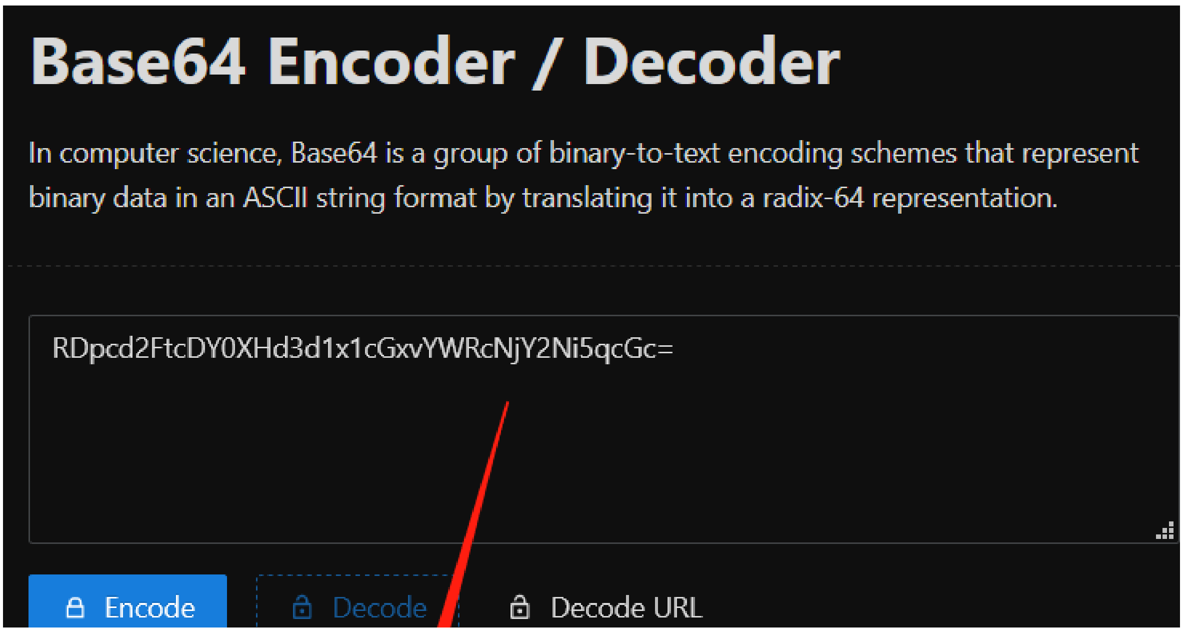


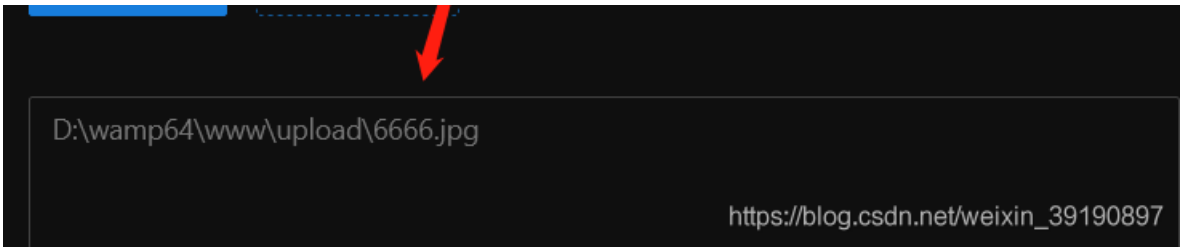


2、当我们分析到流7时，发现了 base64 编码，解码一看是上传的图片的地址：



z1 的值拿去解码得到：

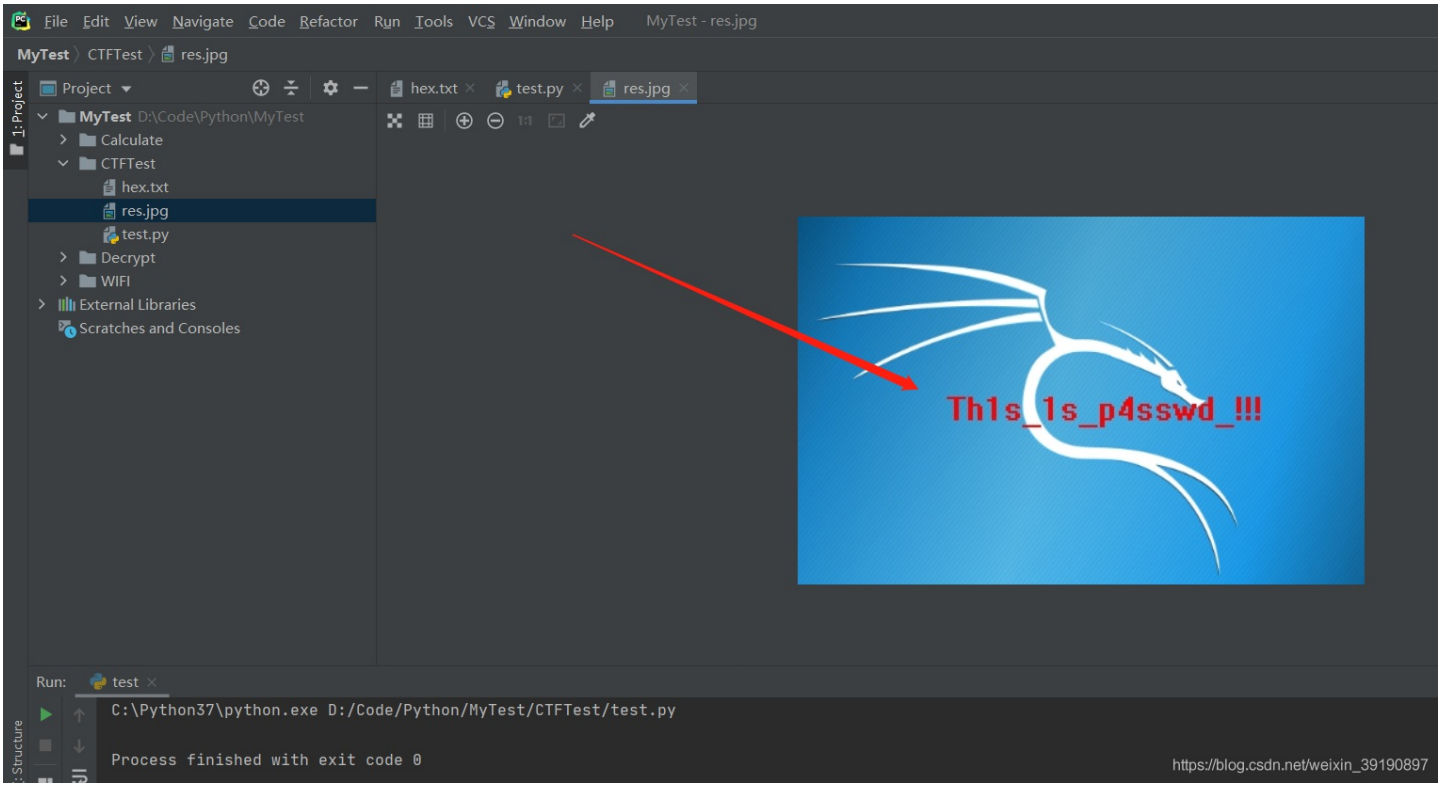




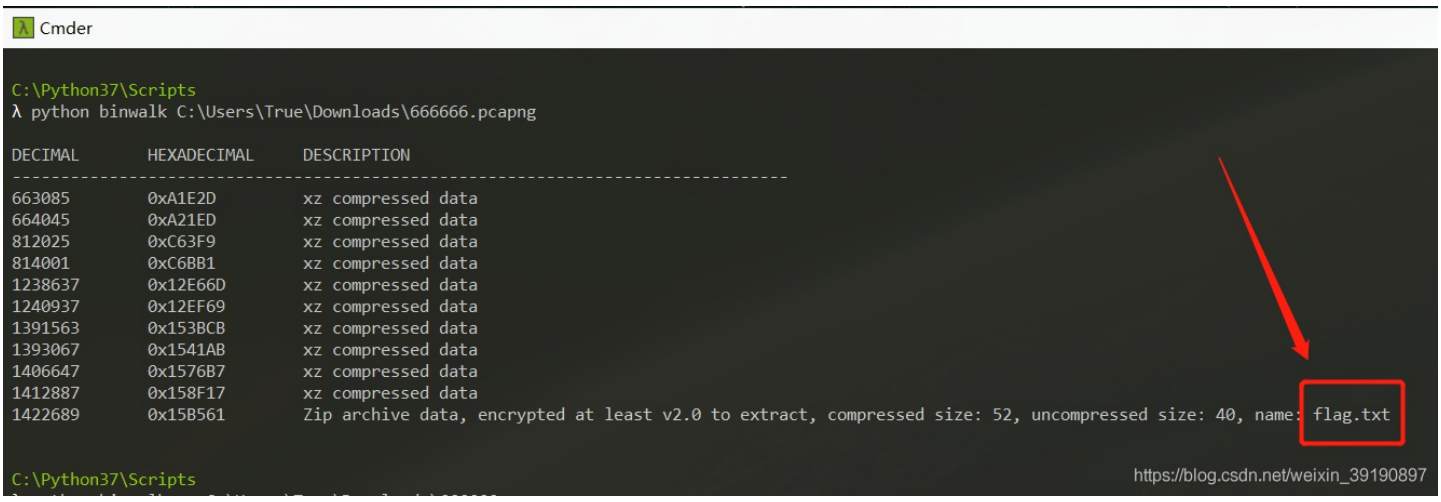
后面跟了个 z2 参数`FF D8`开头`FF D9`结尾，判断为 jpg 图片，将这些十六进制复制出来，以原始文件流写入文件：

```
import struct

a = open("hex.txt","r")#十六进制数据文件
lines = a.read()
res = [lines[i:i+2] for i in range(0,len(lines),2)]

with open("res.jpg","wb") as f:
 for i in res:
  s = struct.pack('B',int(i,16))
  f.write(s)
```

执行脚本获得图片：



2、使用 Binwalk 分析 `666666.pcapng` 文件，发现隐藏着一个 flag.txt，提取：

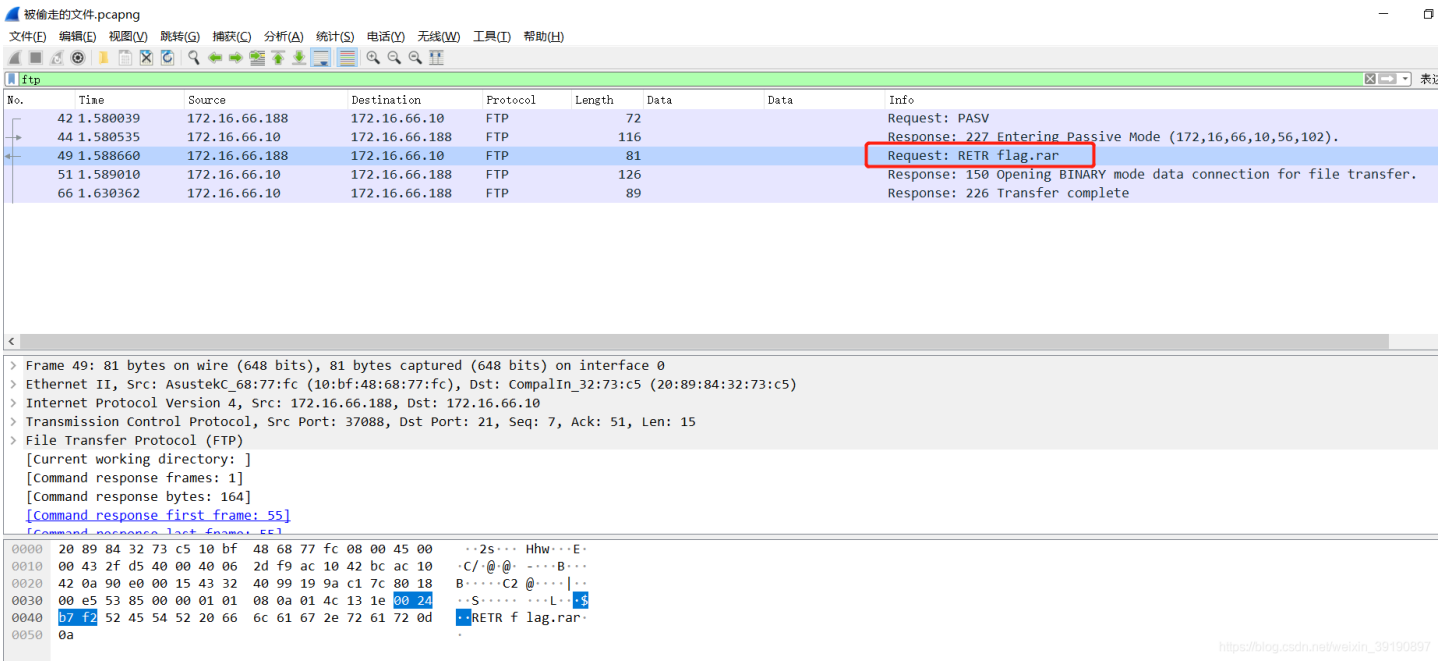


分离出来一个压缩包，需要密码，难道这就是压缩包的密码，然后输入密码，打开了文件，然后获得了flag：`flag{3OpWdJ-JP6FzK-koCMAK-VkfWBq-75Un2z}`。

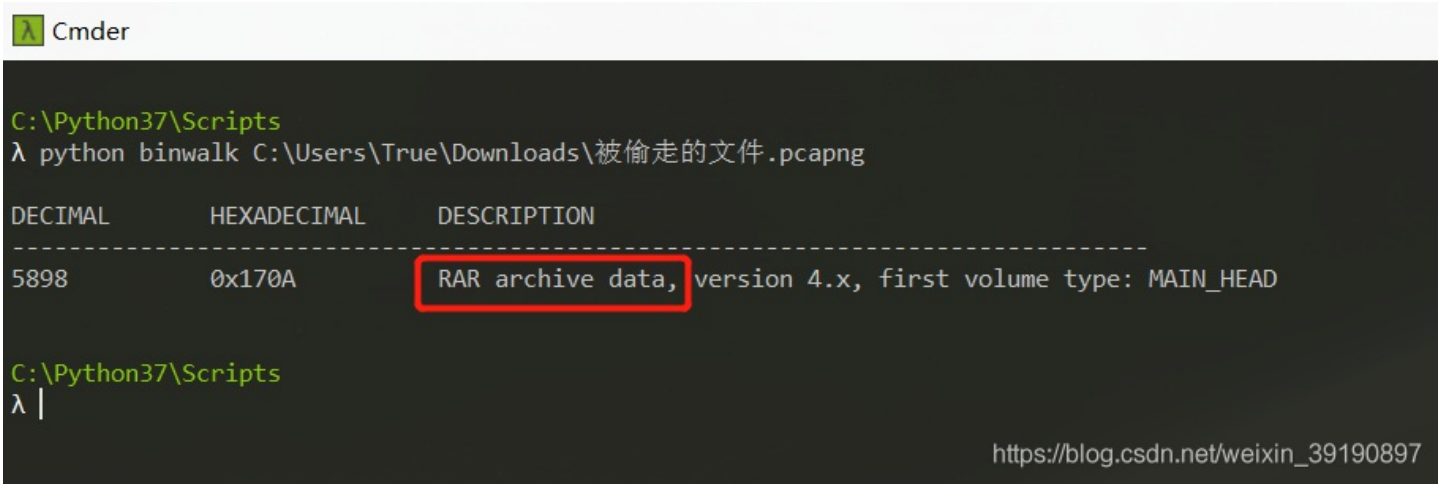
No.23 HTTP流量分析与文件提取

下载后如下：

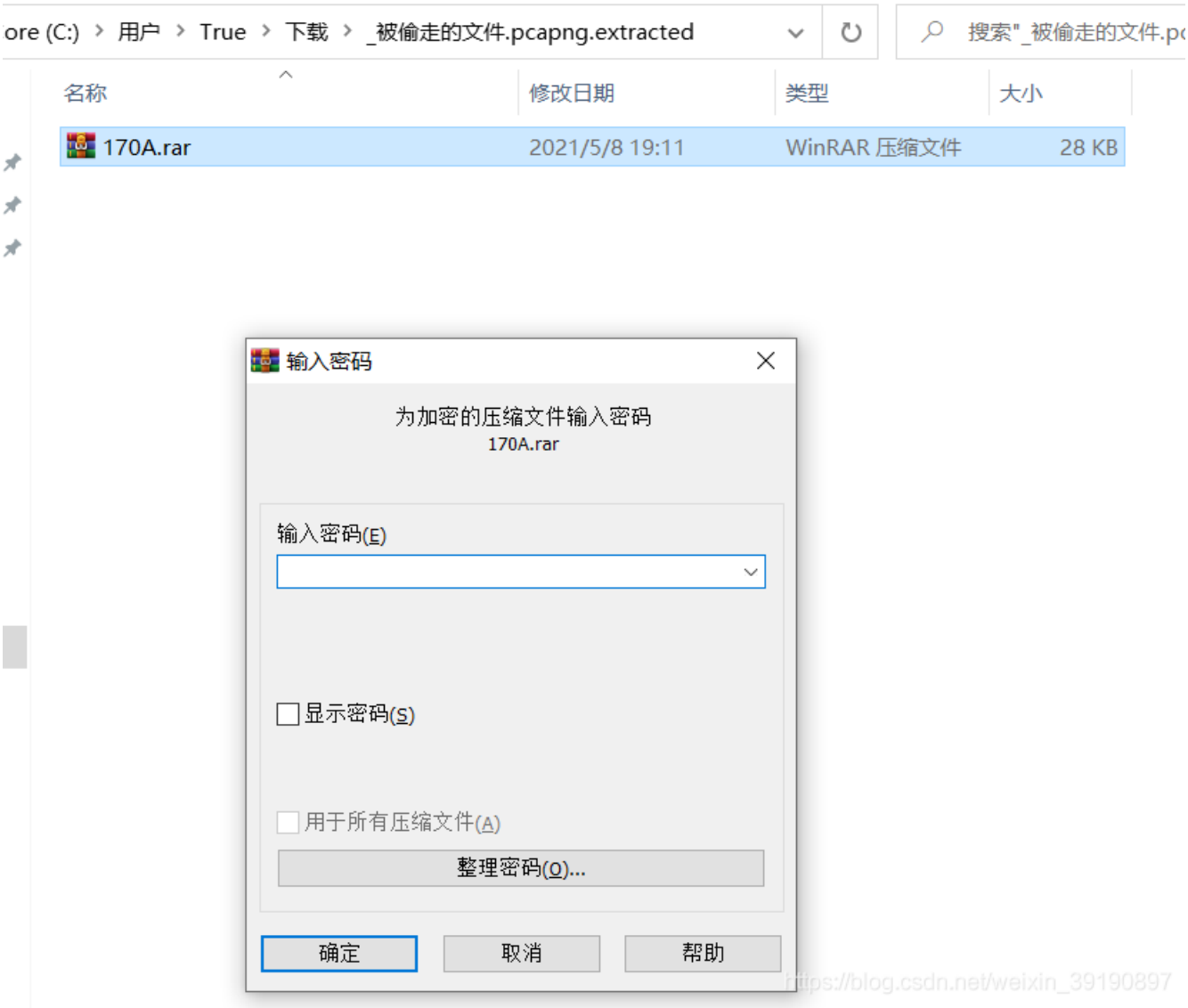
1、传输文件看到了 ftp 协议，过滤一下，发现 flag.rar 文件：



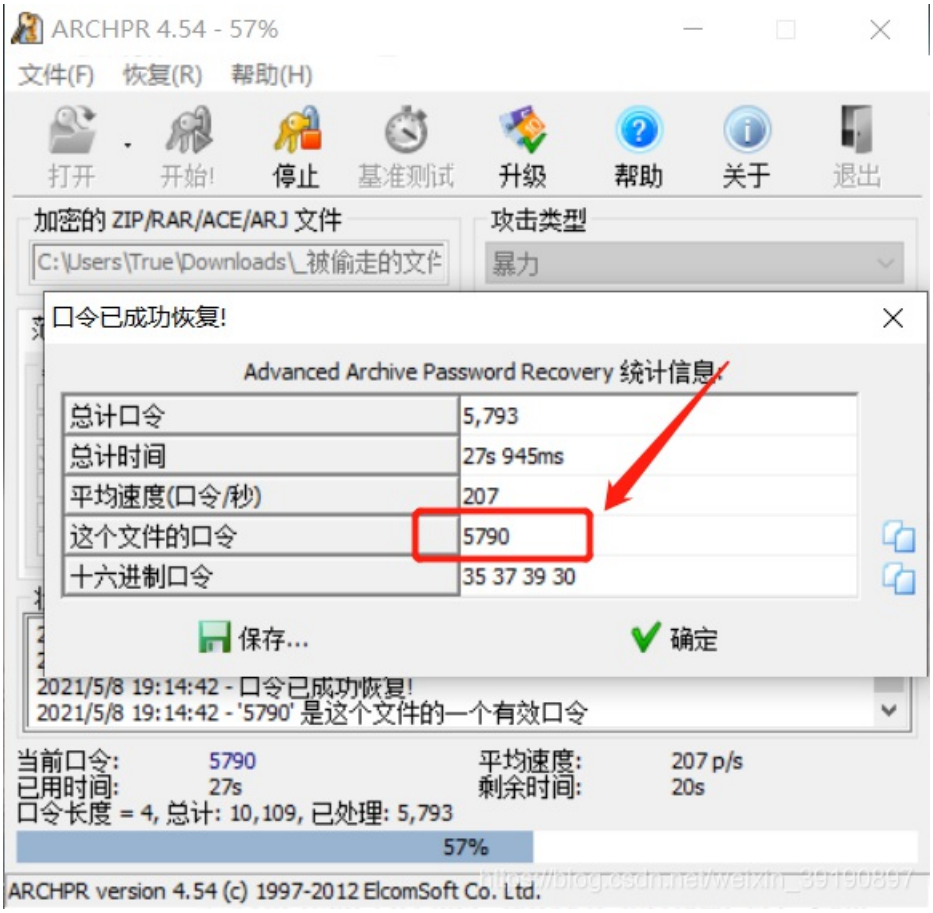
2、使用 Binwalk 分析流量文件`被偷走的文件.pcapng`，发现隐藏的文件，进行提取：



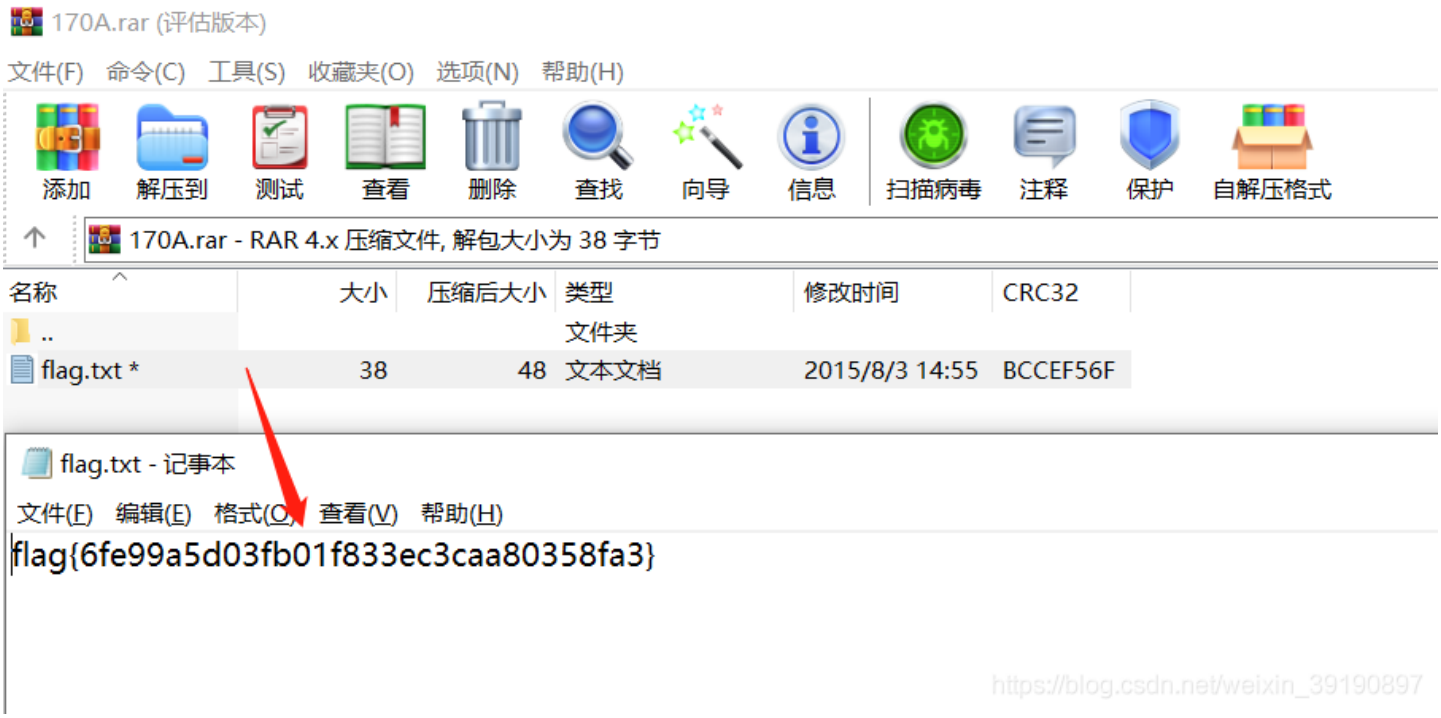
3、获得 170A.rar 文件，需要密码：



4、尝试 ARCHPR 四位数字密码爆破：



5、输入密码获得 flag：



总结

本文是 BUUCTF 较为简单的杂项题目的汇集，总结下本文所涉及的一些杂项题目解题思路，主要还是图片隐写：

1. 拿到文件可先查看属性是否有特殊信息；
2. 可使用 Winhex 或者 Notepad++ 编辑器查看文件是否有特殊信息，同时关注是否隐藏常见的文件头；
3. 必要时可使用 Winhex 修改图片的长、宽数值获得 flag；
4. 隐写图片查看的神器------stegsolve，可以帮助查看文件信息、分离动图、处理 LSB 隐写；
5. Binwalk 工具可快速分辨文件是否由多个文件合并而成，并可将文件进行分离提取；
6. 对于压缩文件的加密，可看看是不是 ZIP 伪加密或者尝试使用 ARCHPR 工具进行四位数字的爆破；
7. 在 Binwalk 无法分析出存在合并的隐藏文件的情况下，可以考虑使用 Steghide 隐写工具查看是否隐写了字符串。

CTF 杂项题目还很多知识和套路不懂，路漫漫其修远兮……