BUUCTF笔记之Misc系列部分WriteUp（二）

原创

KogRow 于 2021-06-18 15:16:07 发布 809 收藏

分类专栏： 杂项 CTF 文章标签： MISC CTF

杂项 同时被 2 个专栏收录

9 篇文章 0 订阅

订阅专栏

CTF

59 篇文章 4 订阅

订阅专栏

1.爱因斯坦

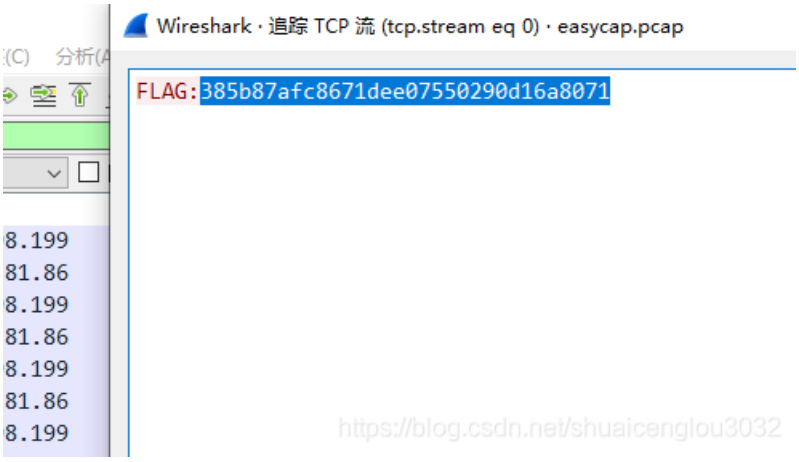
binwalk分离出一个压缩包。

查看图片备注：

解压得到flag。

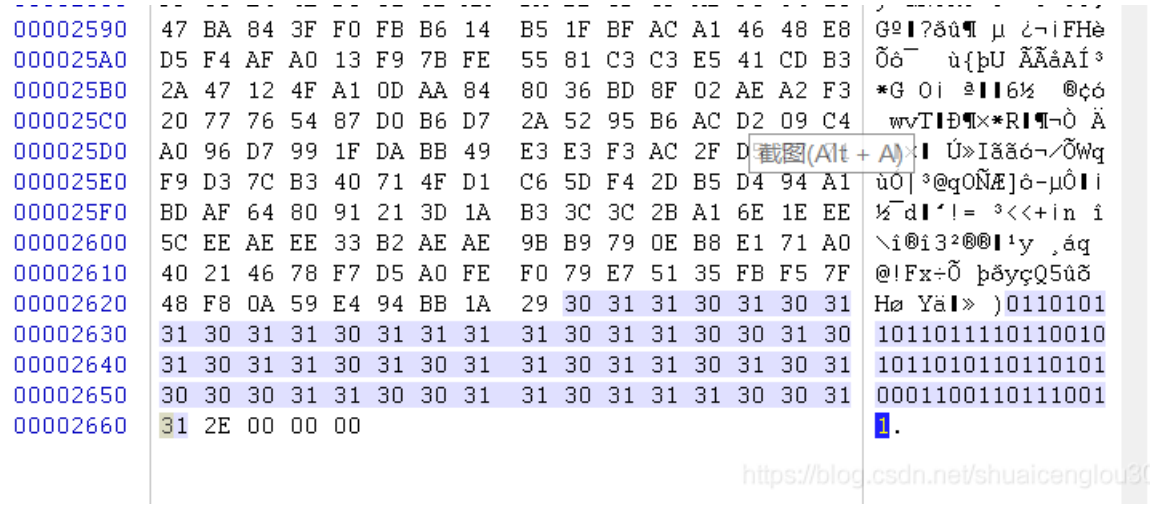
2.easycap

追踪TCP流：

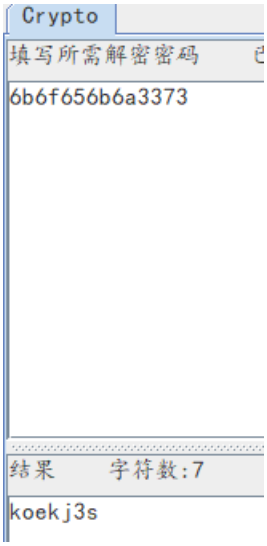


3.另外一个世界

binwalk和steg一无所获，图片备注也看了没有收获，winhex看看：

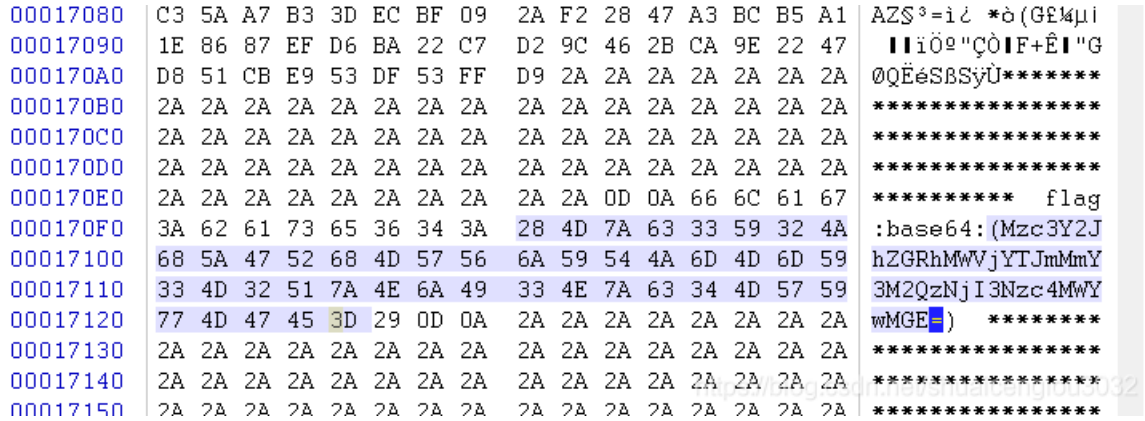


复制出来转字符串得到：



4.隐藏的钥匙

路飞一行人千辛万苦来到了伟大航道的终点，找到了传说中的One piece，但是需要钥匙才能打开One Piece大门，钥匙就隐藏在下面的图片中，聪明的你能帮路飞拿到钥匙，打开One Piece的大门吗？

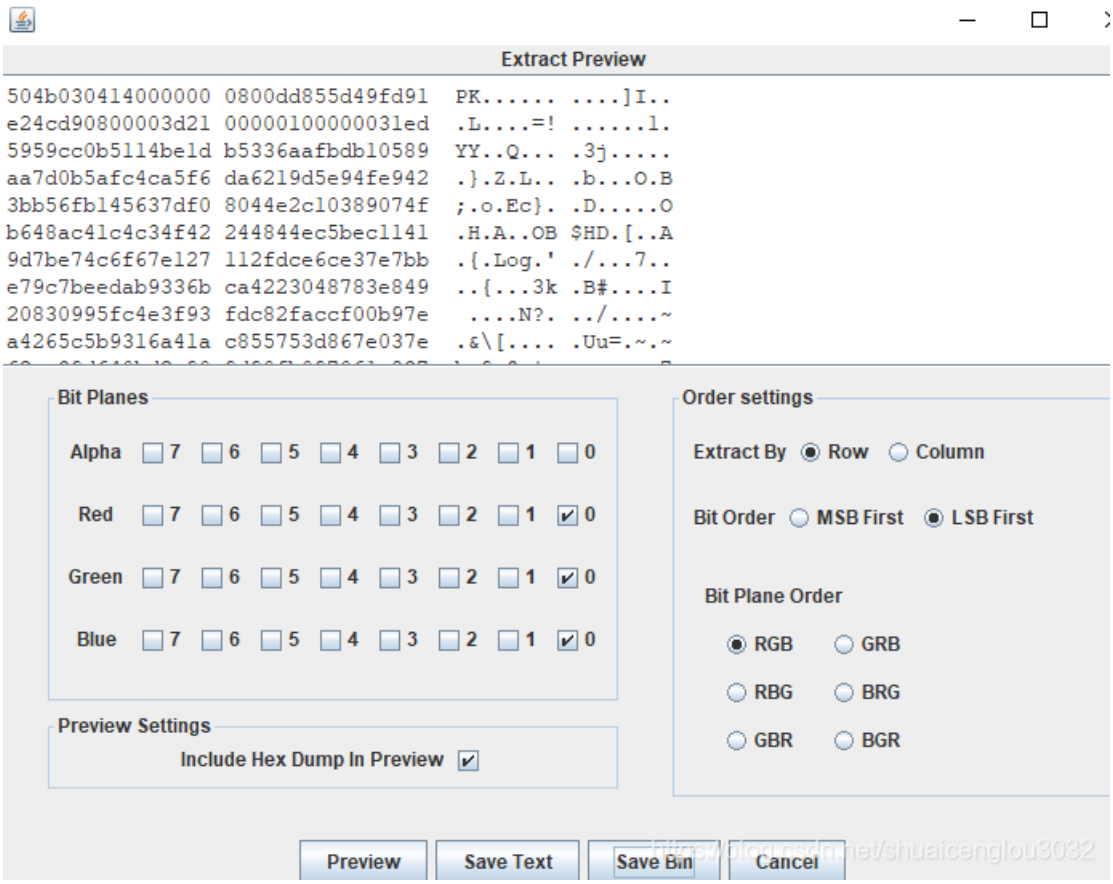


```
00017080  C3 5A A7 B3 3D EC BF 09  2A F2 28 47 A3 BC B5 A1  AZ§³=ì¿ *ò(G£¼µ¡
00017090  1E 86 87 EF D6 BA 22 C7  D2 9C 46 2B CA 9E 22 47  ▲‡‡ïÖº"ÇÒœF+Ê▲"G
000170A0  D8 51 CB E9 53 DF 53 FF  D9 2A 2A 2A 2A 2A 2A 2A  ØQËéSßSÿÙ*******
000170B0  2A 2A 2A 2A 2A 2A 2A 2A  2A 2A 2A 2A 2A 2A 2A 2A  ****************
000170C0  2A 2A 2A 2A 2A 2A 2A 2A  2A 2A 2A 2A 2A 2A 2A 2A  ****************
000170D0  2A 2A 2A 2A 2A 2A 2A 2A  2A 2A 2A 2A 2A 2A 2A 2A  ****************
000170E0  2A 2A 2A 2A 2A 2A 2A 2A  2A 2A 0D 0A 66 6C 61 67  **********  flag
000170F0  3A 62 61 73 65 36 34 3A  28 4D 7A 63 33 59 32 4A  :base64:(Mzc3Y2J
00017100  68 5A 47 52 68 4D 57 56  6A 59 54 4A 6D 4D 6D 59  hZGRhMWVjYTJmMmY
00017110  33 4D 32 51 7A 4E 6A 49  33 4E 7A 63 34 4D 57 59  3M2QzNjI3Nzc4MWY
00017120  77 4D 47 45 3D 29 0D 0A  2A 2A 2A 2A 2A 2A 2A 2A  wMGE=)  ********
00017130  2A 2A 2A 2A 2A 2A 2A 2A  2A 2A 2A 2A 2A 2A 2A 2A  ****************
00017140  2A 2A 2A 2A 2A 2A 2A 2A  2A 2A 2A 2A 2A 2A 2A 2A  ****************
00017150  2A 2A 2A 2A 2A 2A 2A 2A  2A 2A 2A 2A 2A 2A 2A 2A  ****************
```

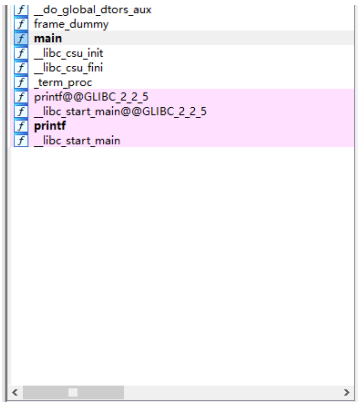
winhex打开，搜索flag，base64解码就有。

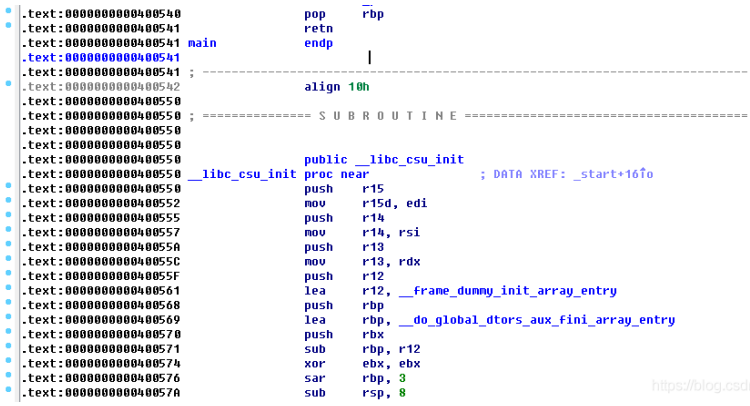
5.FLAG

steg逐位尝试在最低位得到一个zup压缩包：



保存下来之后解压修复一下得到一个EFI。
用IDA打开，定位到main函数：





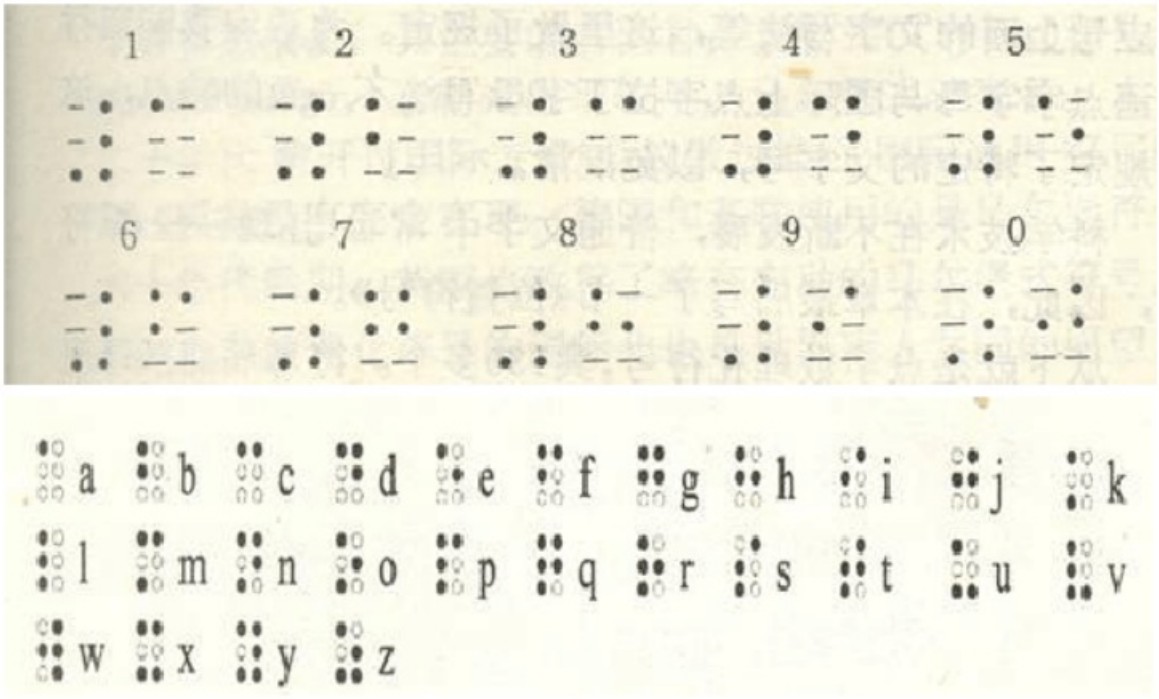
F5一波得到伪代码：

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  return printf("hctf{dd0gf4c3tok3yb0ard4g41n~~~~}", argv, envp);
}
```

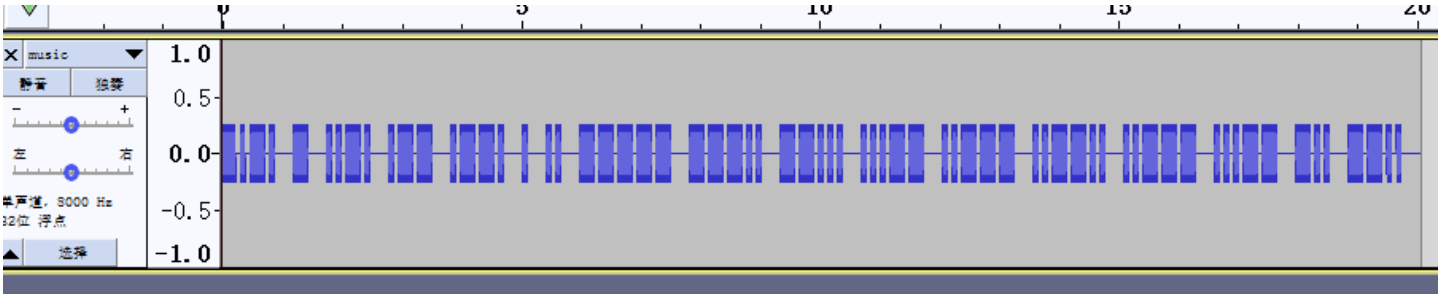
拿到flag。

6.假如给我三天光明

拿到是一张图和一个压缩包。先把图上的盲文翻译一下：
盲文对照表：



所以翻译出来是：kmdonowg。
解压得到一个wav音频。打开听是嘟嘟声，猜测是摩斯电码。

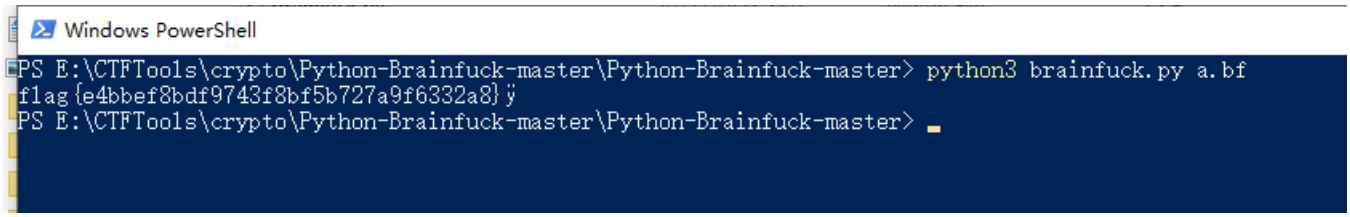


根据上面得到：
-.-. - …-. .-- .-. . … ----- —… --… …-- …— …-… …— …-- -… --…
解码：

7.神秘龙卷风

根据提示，爆破密码：

然后文件的内容是brainfuck代码。把txt重命名为.bf文件后解码得到：



8.后门查杀

发现flag。

```
if( IS_GPC ) {
    $_POST = s_array($_POST);
}
$P = $_POST;
unset($_POST);
/*====================== ******** ======================*/

//echo encode_pass('angel');exit;
// ******Ç******,***□ĵ*%****,***Ï****Ç**
$pass ='6ac45fb83b3bc355c024f5034b947dd3';//angel

//****** cookie ***÷*X******Ç**, ***¾******, ***□********, *****¼***Ï**
// cookie ǰ□
$cookiepre = '';
// cookie ******
$cookiedomain = '';
// cookie ****·**
$cookiepath = '/';
// cookie **Ч**
```

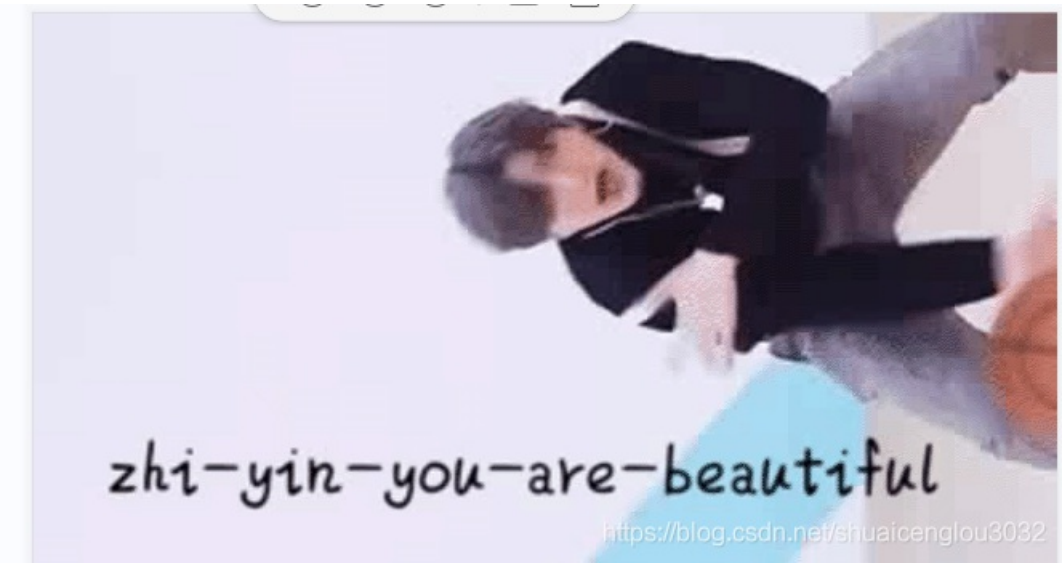

9.[BJDCTF2020]鸡你太美

使用winhex打开两张gif，副本图片缺少gif文件头
补全：

```
47 49 46 38 37 61 or
47 49 46 38 39 61
```

```
GIF87a
GIF89a
```

GIF Graphics interchange format file
Trailer: 00 3B (.;)



10.荷兰宽带数据泄露

用户名包上flag{}就行

11.webshell后门

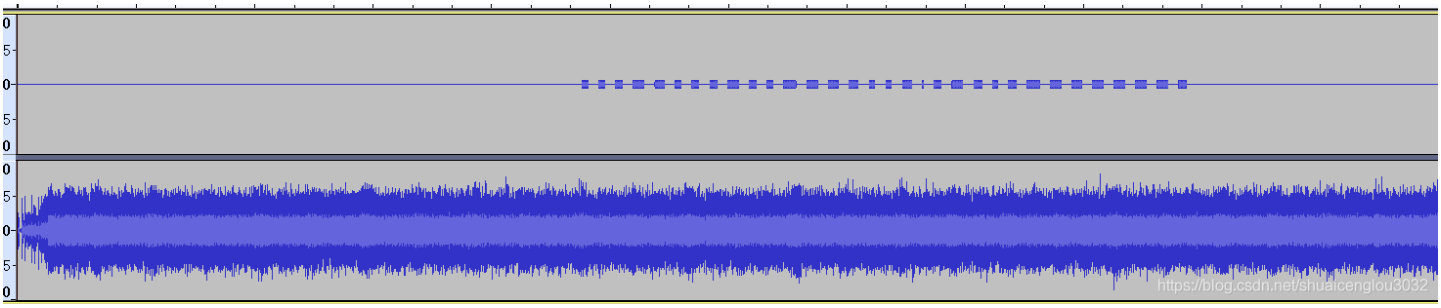

```php
ob_start();
$mtime = explode(' ', microtime());
$starttime = $mtime[1] + $mtime[0];
define('SA_ROOT', str_replace('\\', '/', dirname(__FILE__)).'/');
define('SELF', $_SERVER['PHP_SELF'] ? $_SERVER['PHP_SELF'] : $_SERVER['SCRIPT_NAME']);
define('IS_WIN', DIRECTORY_SEPARATOR == '\\');
define('IS_GPC', 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;
//angel = ba8e6c6f35a53933b871480bb9a9545c
// 如果需要密码验证,请修改登陆密码,留空为不需要验证
$pass   = 'ba8e6c6f35a53933b871480bb9a9545c'; //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
$cookiedomain = '';
// cookie 作用路径
$cookiepath = '/';
// cookie 有效期
$cookielife = 86400;
```


12.来首歌吧



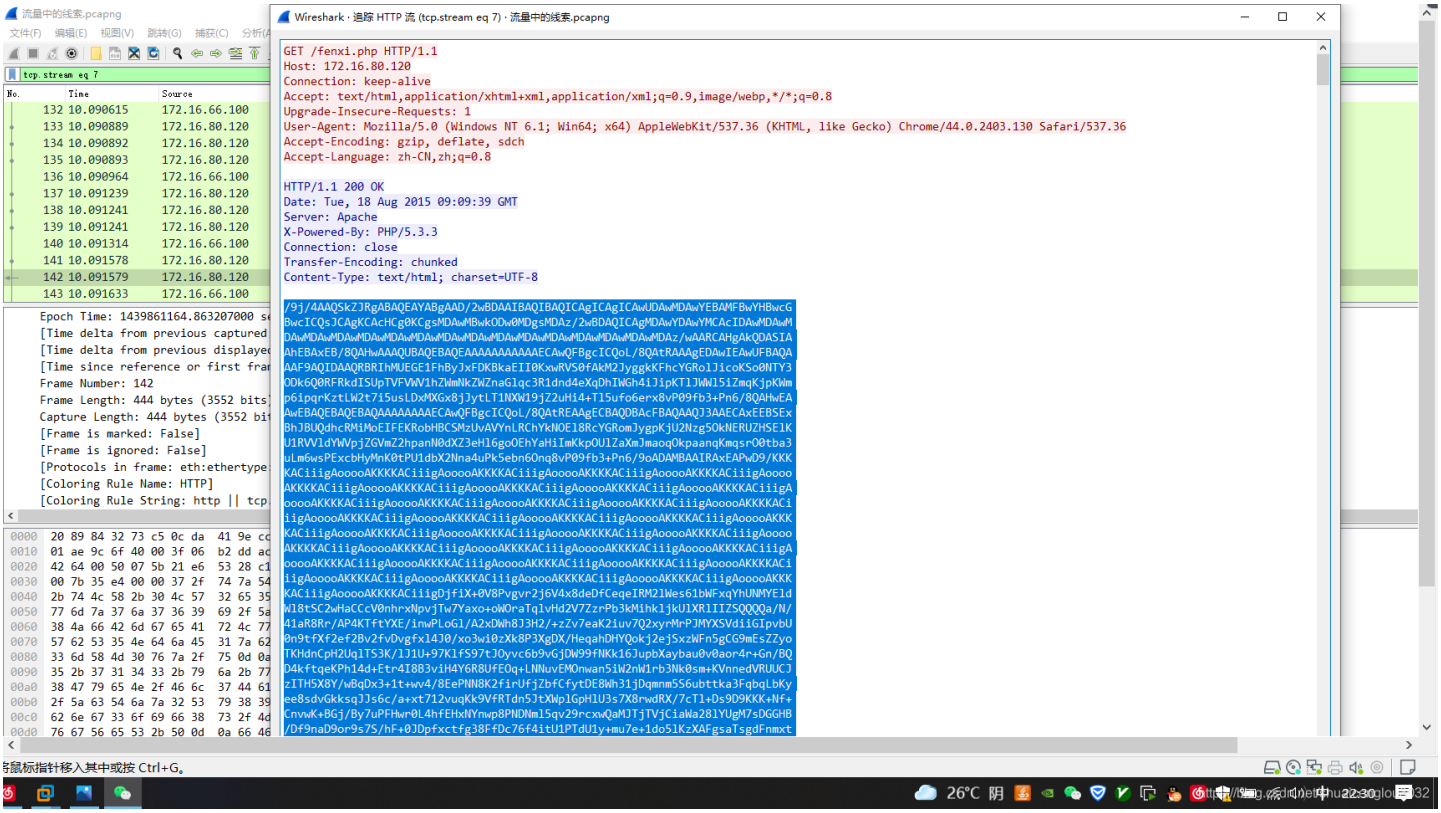
先盲猜摩斯密码

... -... -.-. ----. ...— ... -... ...- ----. -.-. -... ----- .---- —... —... ...-. ... ...— . -... .---- --... -... --... ----- ----. ...— -----. .---- ----. .---- -.-.

解码：

5BC925649CB0188F52E617D70929191C

13.数据包中的线索



目测是图片的base64编码，解码一下：

```
import os, base64

with open("C:\\Users\\root\\Desktop\\1.txt", "r") as f:
    imgdata = base64.b64decode(f.read())
    file = open('C:\\Users\\root\\Desktop\\1.jpg', 'wb')
    file.write(imgdata)
    file.close()
```

得到：

14.九连环

binwalk+foremost分离得到：

修改伪加密标志位，01改成00（奇数改成偶数）解压得到：

binwalk分析图片：

```
$ binwalk 1.jpg

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

无果。
看了WP，上steg：

```
Processing triggers for kali-menu (2021.2.0) ...

┌──(fuck㉿kali)-[~]
└─$ steghide info 1.jpg
"1.jpg":
  format: jpeg
  capacity: 1.2 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "ko.txt":
    size: 48.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes

┌──(fuck㉿kali)-[~]
└─$
```

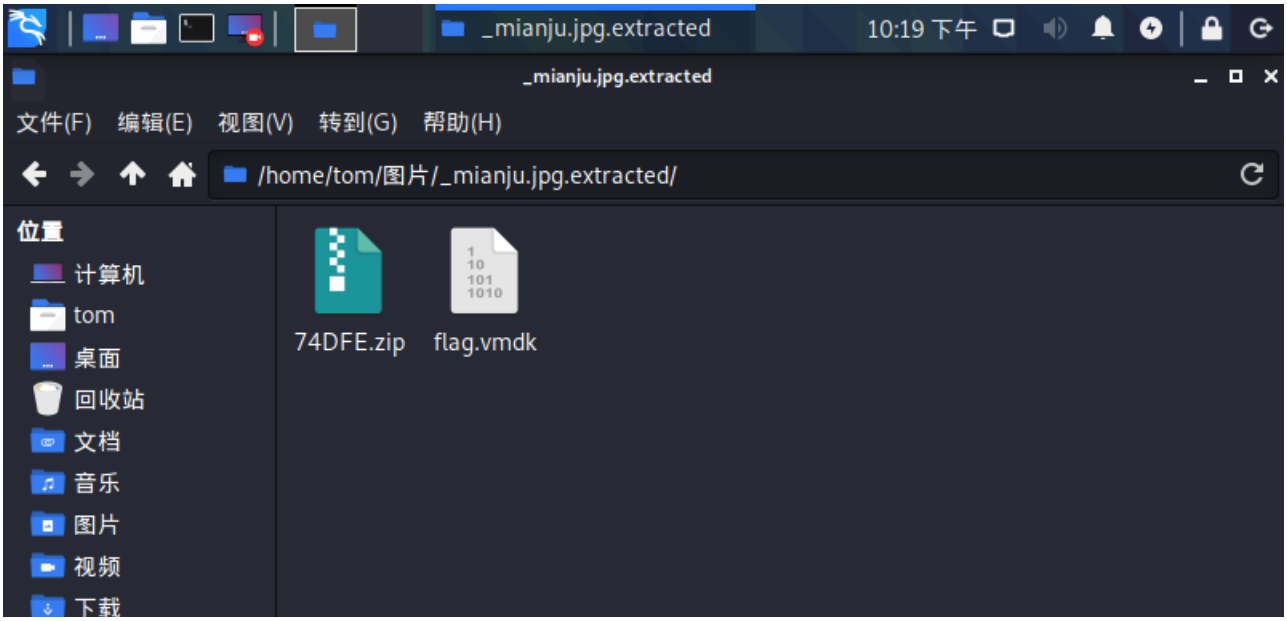

提取：

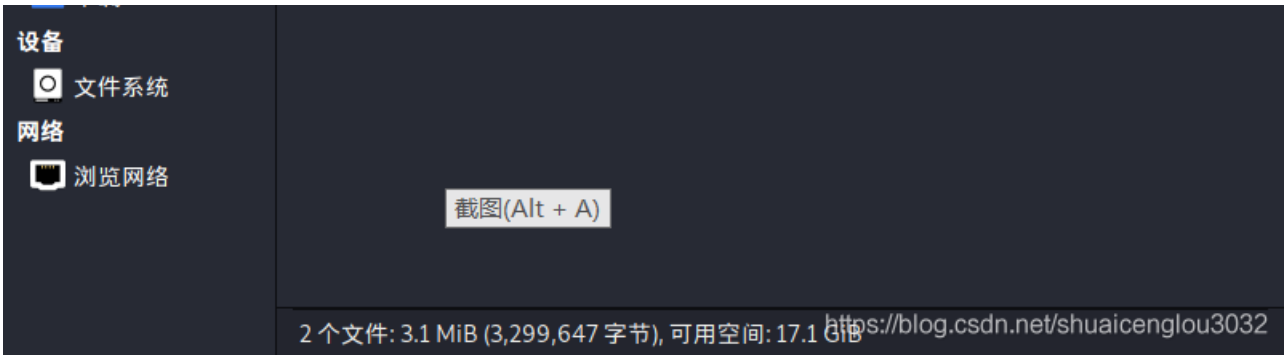
```
┌──(fuck㉿kali)-[~]
└─$ steghide extract -sf 1.jpg
Enter passphrase:
wrote extracted data to "ko.txt".
```

得到密码bV1g6t5wZDJif^J7
解压得到flag。

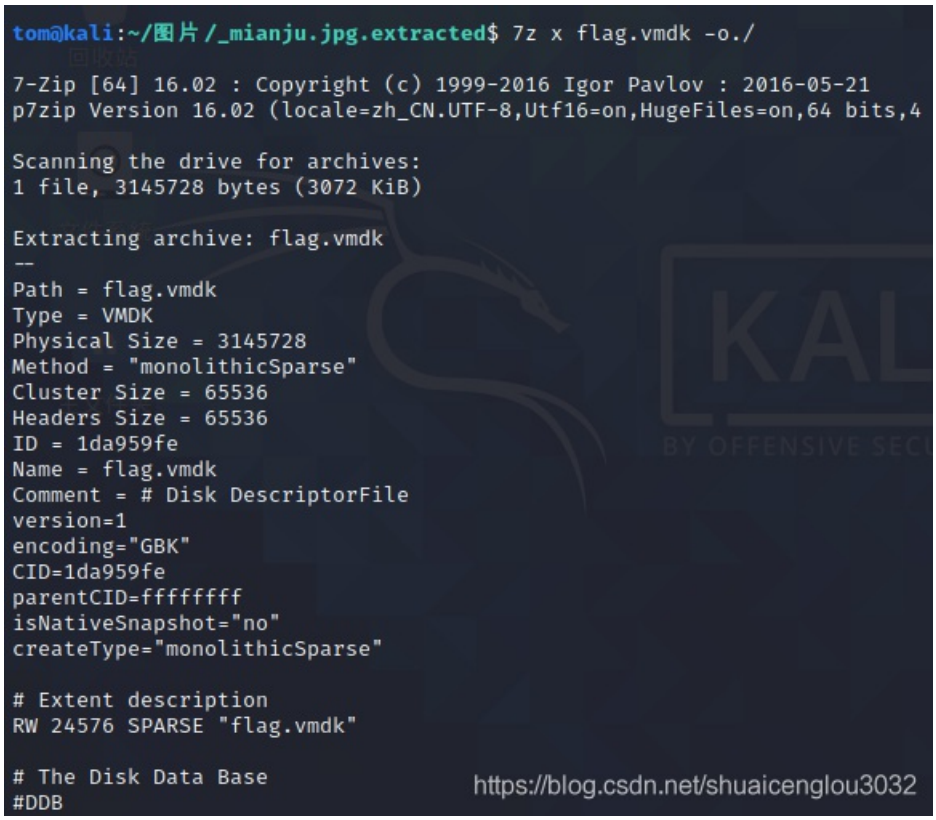
15.面具下的flag

binwalk -e mianju.jpg





压缩包改伪加密标志位之后出来的也是flag.vmdk。
右键这个.vmdk文件，映射为磁盘（我的电脑上装了VMware），打开得到：



当然，也可以在kali下使用7z命令解压：

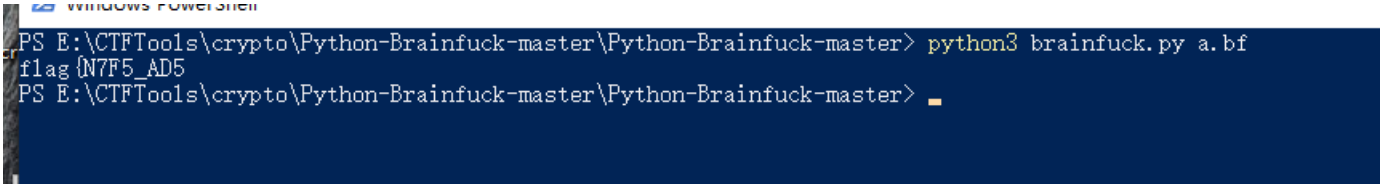
效果是一样的。（好吧，实战后发现不一样，所以还是得kali下7z命令解压）
part2：
Oops,flag_part_two_isn't_here!

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

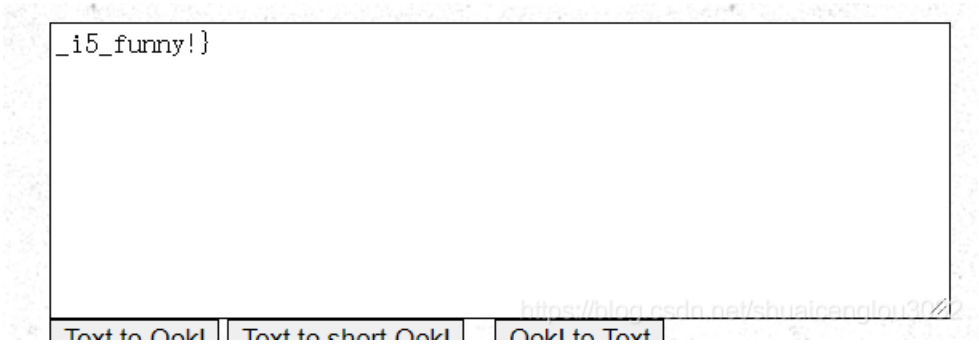
part1:

```
+++++ +++++ [->++ +++++ +++<] >++.+ +++++ .<+++ [->-- -<]>- -.+++ +++.<
++++[ ->+++ +<]>+ +++.< +++++ +[->- ----- <]>-- ----- --.<+ +++[- >----
<]>-- ----- .<+++ [->++ +<]>+ +++++ .<+++ +[->- ---<] >-.<+ +++++ [->++
++++< ]>+++ +++.< +++++ [->-- ---<] >---- -.+++ .<+++ [->-- -<]>- ----- .<
```

part1有经验的一看就知道是brainfuck。解码：



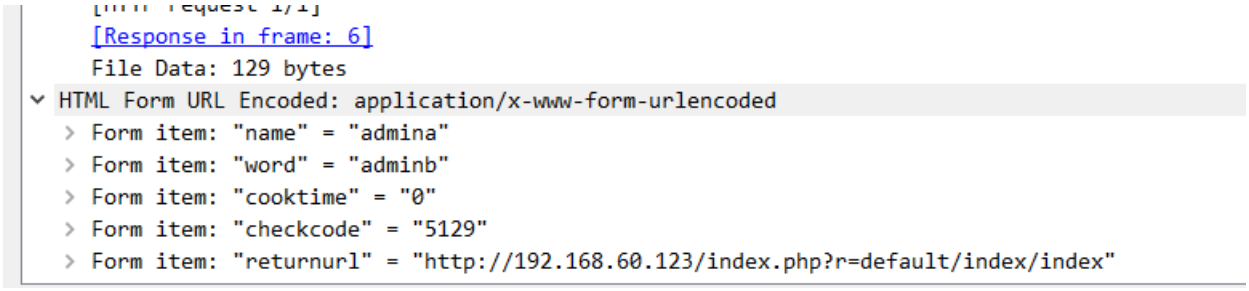
part2不知道是啥，看了WP说是OoK加密：
在线解密用这里：Ook



合起来就是flag.

16.被劫持的神秘礼物

流量分析入门题。



adminaadminb的md5就是flag.

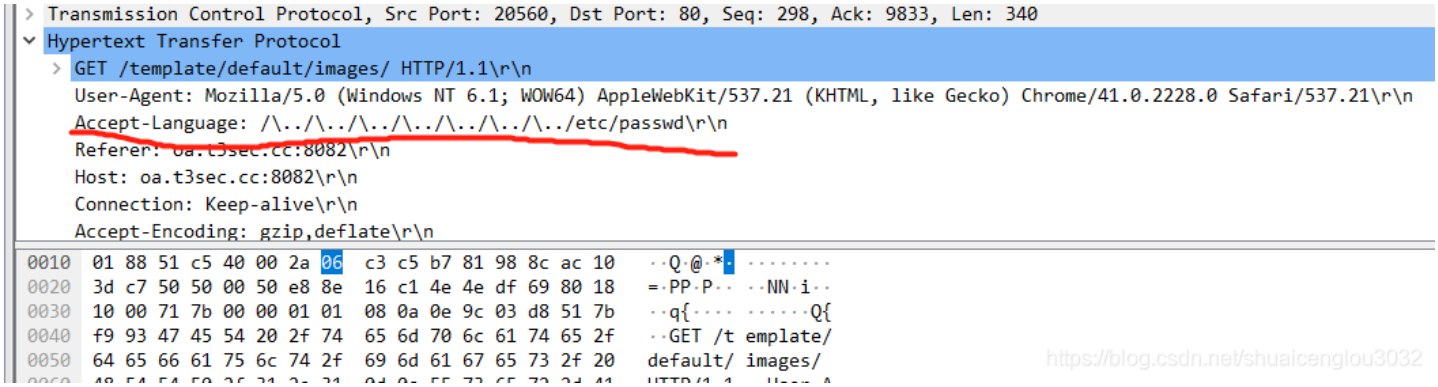
17.大流量分析（一）

这种题很有实战意义。

某黑客对A公司发动了攻击，以下是一段时间内我们获取到的流量包，那黑客的攻击ip是多少？
下载下来是非常巨大的流量包，但是有时间段，我们从最早的20160809_164452看起：

名称	修改日期	类型	大小
数据采集D_eth0_NS_20160809_164452.pcap	2021/6/27 11:24	Wireshark captu...	488,282 KB
数据采集D_eth0_NS_20160809_170106.pcap	2021/6/27 11:24	Wireshark captu...	488,283 KB
数据采集D_eth0_NS_20160809_170427.pcap	2021/6/27 11:24	Wireshark captu...	488,283 KB
数据采集D_eth0_NS_20160809_171230.pcap	2021/6/27 11:24	Wireshark captu...	488,283 KB
数据采集D_eth0_NS_20160809_172831.pcap	2021/6/27 11:26	Wireshark captu...	488,282 KB
数据采集H_eth0_NS_20160809_164535.pcap	2021/6/27 11:26	Wireshark captu...	488,283 KB
数据采集H_eth0_NS_20160809_170103.pcap	2021/6/27 11:26	Wireshark captu...	488,283 KB
数据采集H_eth0_NS_20160809_170424.pcap	2021/6/27 11:26	Wireshark captu...	488,283 KB
数据采集H_eth0_NS_20160809_170930.pcap	2021/6/27 11:26	Wireshark captu...	488,283 KB
数据采集H_eth0_NS_20160809_172819.pcap	2021/6/27 11:26	Wireshark captu...	488,283 KB

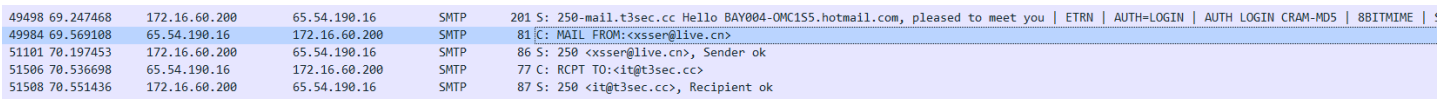
重点看响应状态码为200的HTTP包：



发现问题了，Accept-Language内容是一个敏感文件读取的操作，这里我们猜测可能是黑客发起的请求，因为正常请求不长这样。
看IP部分得到flag{183.129.152.140}

18.大流量分析（二）

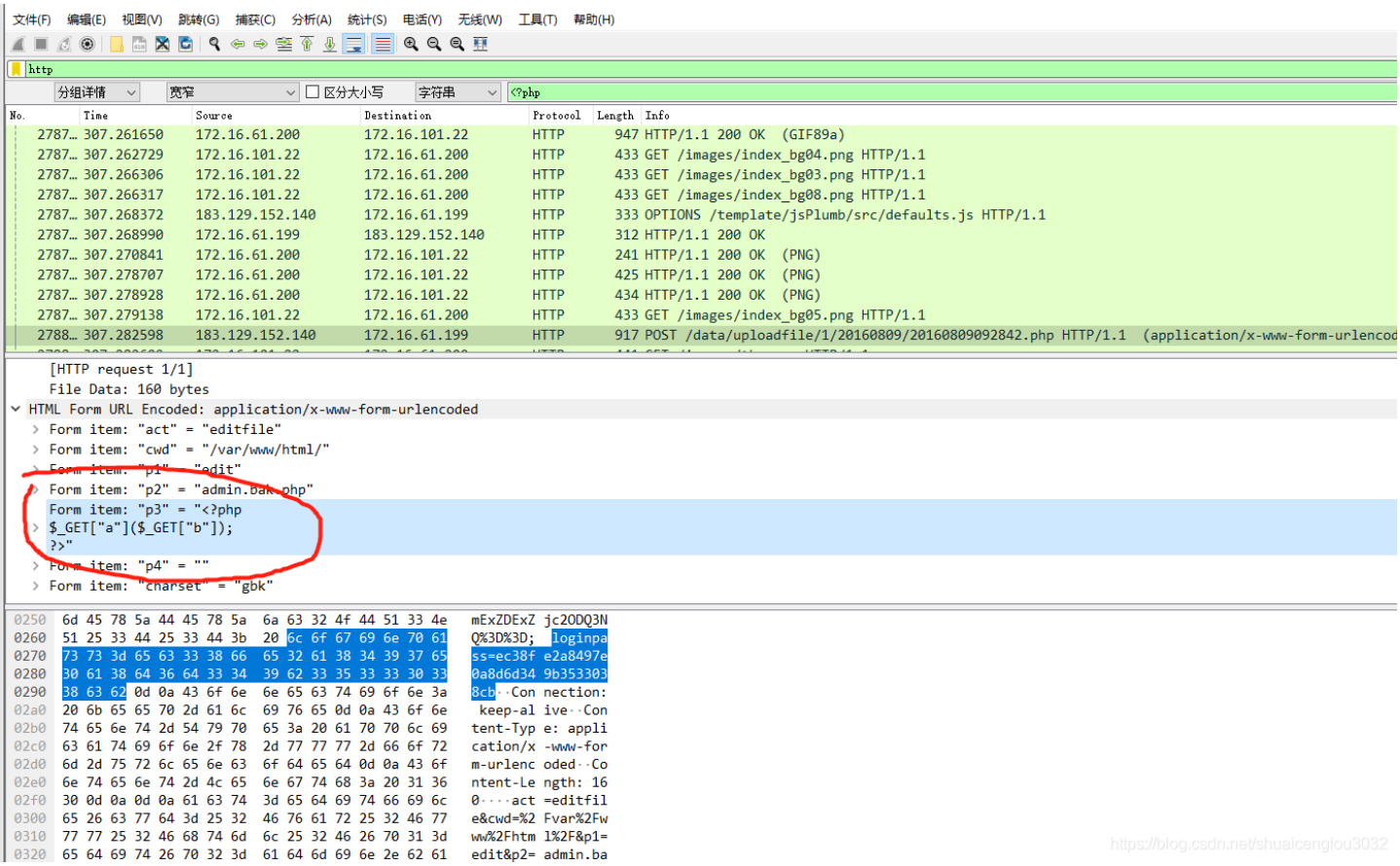
flag{xsser@live,[,换成.]cn}
黑客对A公司发动了攻击，以下是一段时间内获取到的流量包，那黑客使用了哪个邮箱给员工发送了钓鱼邮件？
这题询问邮箱。
过滤一下SMTP的协议，看到这条：



看到了邮箱`xsser@live.cn`

19.大流量分析（三）

某黑客对A公司发动了攻击，以下是一段时间内我们获取到的流量包，那黑客预留的后门的文件名是什么？
172819这个包里面过滤http协议，然后查找一下关键词：<?php



这就很诡异了，动态函数。
开始我以为后门文件是201608092842.php，但提交了发现不对
于是试试这个包里面的admin.bak.php
成功。flag{admin.bak.php}

20.[ACTF新生赛2020]base64隐写

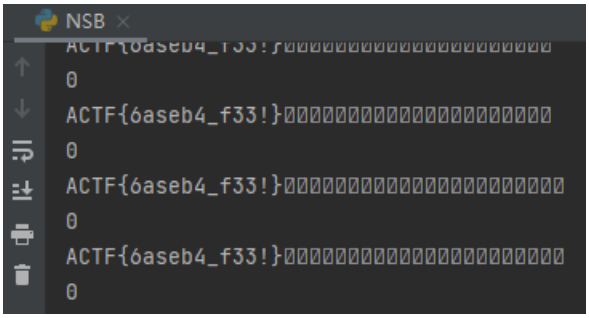
base64隐写。上python2脚本：

```python
def get_base64_diff_value(s1, s2):
    base64chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
    res = 0
    for i in xrange(len(s2)):
        if s1[i] != s2[i]:
            return abs(base64chars.index(s1[i]) - base64chars.index(s2[i]))
    return res


def solve_stego():
    with open('E:/1.txt', 'rb') as f:
        file_lines = f.readlines()
        bin_str = ''
        for line in file_lines:
            steg_line = line.replace('\n', '')
            norm_line = line.replace('\n', '').decode('base64').encode('base64').replace('\n', '')
            diff = get_base64_diff_value(steg_line, norm_line)
            print diff
            pads_num = steg_line.count('=')
            if diff:
                bin_str += bin(diff)[2:].zfill(pads_num * 2)
            else:
                bin_str += '0' * pads_num * 2
            print goflag(bin_str)


def goflag(bin_str):
    res_str = ''
    for i in xrange(0, len(bin_str), 8):
        res_str += chr(int(bin_str[i:i + 8], 2))
    return res_str


if __name__ == '__main__':
    solve_stego()
```

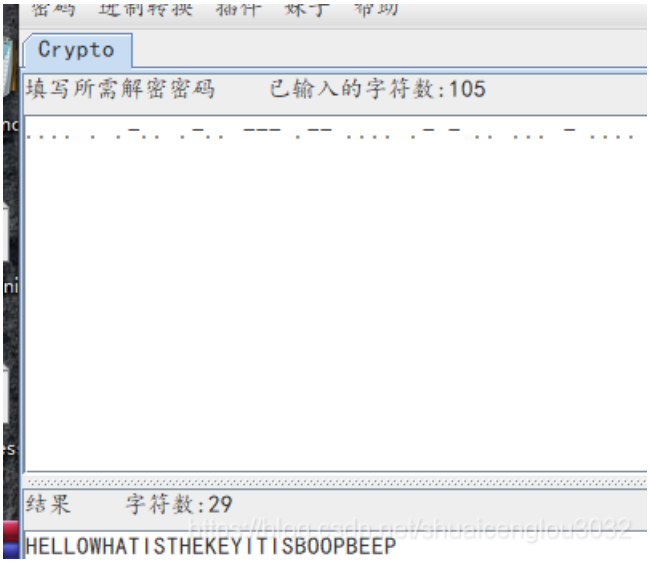


21.robomunication

点击打开，目测是摩斯密码。
整理得：

... . .-... .-... — .-- ... .- - ... ... - ... . -.- . -.-- ... - ... ... -... — --- .-. -... . . .-.

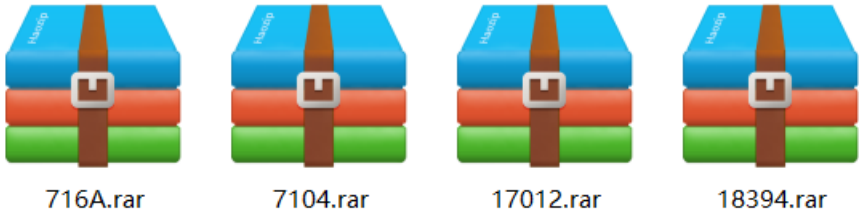
解码：



flag{BOOPBEEP}

22.[SWPU2019]神奇的二维码

steg查看无果。
binwalk分析有RAR.
foremost提取失败，binwalk -e 提取成功。

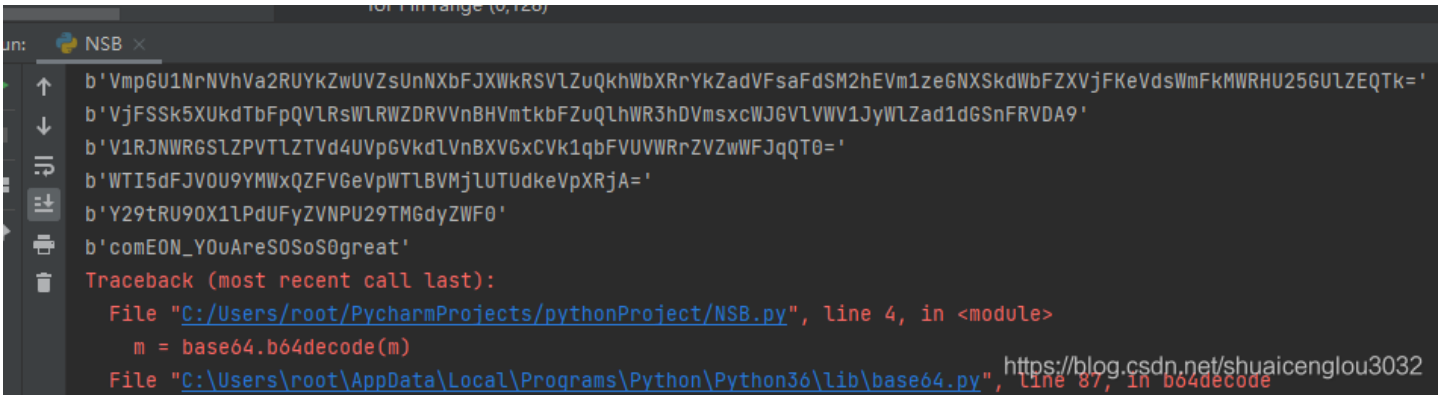


716A毫无价值。
7104解码之后是716A的里面的压缩包的密码，也没有价值。
17012得到一大段base64.
试了几次一直是base64。写脚本循环跑看看：

```
import base64
m = "Vm0wd2QyUXlVWGxWV0d4V1YwZDRWMVl3WkRSV01WbDNXa1JTVjAxV2JETlhhMUpUVmpBeFYySkVUbGhoTVVwVVZtcEJlRll5U2tWVWJHaG9
UVlZ3VlZadGNFSmxSbGw1VTJ0V1ZXSkhhRzlVVmxaM1ZsWmFjVkZ0UmxSTmJFcEpWbTEwYTFkSFNrZGpSVGxhVmpOU1IxcFZXbUZrUjA1R1UyMTR
VMkpIZHpGV1ZFb3dWakZhV0ZOcmFHaFNlbXhXVm1wT1QwMHhjRlpYYlVaclVqQTFSMWRyV25kV01ERkZVbFJHVjFaRmIzZFdha1poVjBaT2NtRkh
hRk5sYlhoWFZtMXdUMVF3TUhoalJscFlZbFZhY1ZadGRHRk5SbFowWlVaT1ZXSlZXVEpWYkZKSFZqRmFSbUl6WkZkaGExcG9WakJhVDJOdFJraGh
SazVzWWxob1dGWnRNWGRVTVZGM1RVaG9hbEpzY0ZsWmJGWmhZMnhXY1ZGVVJsTk5XRUpIVmpKNFQxWlhTa2RqUmxwWFlsaFNNMVpxU2t0V1ZrcFp
Xa1prYUdFeGNGbFhhMVpoVkRKTmVGcElUbWhTTW5oVVdWUk9RMWRzV1hoWGJYUk9VakZHTlZaWE5VOWhiRXAwVld4c1dtSkdXbWhaTW5oWFl6Rld
jbHBHWkdsU2JrSmFWMnhXWVZReFdsaFRiRnBZVmtWd1YxbHJXa3RUUmxweFUydGFiRlpzV2xwWGExcHJZVWRGZUdOR2JGaGhNVnBvVmtSS1QyTXl
Ua1phUjJoVFRXNW9WVlpHWTNoaU1rbDRWMWhvWVZKRlNtRldiWGh6VFRGU1ZtRkhPV2hpUlhCNldUQmFjMWR0U2tkWGJXaGFUVzVvV0ZreFdrZFd
Wa3B6VkdzMVYySkdhM2hXYTFwaFZURlZlRmR1U2s1WFJYQnhWVzB4YjFZeFVsaE9WazVPVFZad2VGVXlkREJXTVZweVkwWndXR0V4Y0ROV2FrWkx
WakpPU1dKR1pGZFNWWEJ2Vm10U1MxUXlUWGxVYTFwb1VqTkNWRmxZY0ZkWFZscFlZMFU1YVUxcmJEUldNV2h2V1ZaS1IxTnNaRlZXYkZwNlZHeGF
ZVmRGTlZaUFZtaFRUVWhDU2xac1pEUmpNV1IwVTJ0a1dHSlhhR0ZVVnpWdlYwWnJlRmRyWkZkV2EzQjZWa2R6TVZZd01WWmlla1pYWWxoQ1MxUld
aRVpsUm1SWldrVTFWMVpzY0ZWWFYzUnJWVEZzVjFWc1dsaGlWVnBQVkZaYWQyVkdWWGxrUkVKWFRWWndlVmt3V25kWFIwVjRZMFJPV21FeVVrZGF
```

WM2hIWTIxS1IxCEhIRmMTVlhCS1ZtMTBVMU14V1hOWFdHaFI2DXhhVmxSc1pHOWp3DHB4VTIwNWJHSkhVDGxhV1dNMVIwVXhjbUpPVWxkTmFsW1V
Wa2Q0YTFOR1ZuTlhiRlpYVFRGS05sWkhlR0ZXTWxKSVZXdG9hMUl5YUhCVmJHaENaREZhYzFwRVVtcE5WMUl3VlRKMGExZEhTbGhoUjBaVlZucFd
kbFl3V25OT2JFcHpXa2R3YVZORlNrbFdNblJyWXpGVmVWTnVTbFJpVlZwWVZGYzFiMWRHYkhGVGExcHNVbTFTV2xkclZURldNVnB6WTBaV1dGWnN
Ta2hhUkVaYVpVZEtTVk5zYUdoTk1VcFdWbGN4TkdReVZrZFdXR3hyVWpCYWNGVnRlSGRsYkZsNVpVaGtXRkl3VmpSWk1GSlBWMjFGZVZWclpHRld
NMmhJV1RJeFMxSXlSa2RoUmxKVFZsaENTMVp0TVRCVk1VMTRWbGhvV0ZkSGFGbFpiWGhoVm14c2NscEhPV3BTYkhCNFZrY3dOVll4V25OalJXaFl
Wa1UxZGxsV1ZYaGpNVTUxWTBaa1RtRnNXbFZXYTJRMFlURk9SMVp1VGxoaVJscFlWRlJHUzA1c1draGxSMFpYVFd4S1NWWlhkRzloTVVwMFlVWlN
WVlpYYUVSVk1uaGhZekZ3UlZWdGNFNVdNVWwzVmxSS01HRXhaRWhUYkdob1VqQmFWbFp0ZUhkTk1XeFdWMjVrVTJKSVFraFdSM2hUVlRKRmVsRnF
WbGRTTTJob1ZrUktSMWRHU2xsYVIzQlRWak5vV1ZkWGVHOVJNVkpIVlc1S1dHSkZjSE5WYlRGVFpXeHNWbGRzVG1oU1ZFWjZWVEkxYjFZeFdqWlJ
hbEpWWVRKU1NGVnFSbUZYVm5CSVlVWk9WMVpHV2xaV2JHTjRUa2RSZVZaclpGZFhSM2h5VldwQ1lXTkdWblJsU0dSc1lrWldOVnBWYUd0WFIwcEh
ZMFpvV2sxSGFFeFdha1poVW14a2NtVkdaRTVXYmtKSlYxUkplRk14U1hoalJXUmhVbFJXVDFWc2FFTlRNVnAwVFZSQ1ZrMVZNVFJXYkdodlYwWmt
TR0ZHYkZwaVdHaG9WbXBHYzJNeGNFaFBWbVJUWWxob1lWZFVRbUZoTWtwSVUydG9WbUpIZUdoV2JHUk9UVlpzVjFaWWFGaFNiRnA1V1ZWYWExUnR
Sbk5YYkZaWFlUSlJNRlpFUms5VFJrcHlXa1pLYVZKdVFuZFdiWFJYVm0xUmVGZHVVbXBTVjFKWFZGWmFkMDFHVm5Sa1J6bFdVbXh3TUZsVldsTld
WbHBZWVVWU1ZXSkdjR2hWTUdSWFUwWktkR05GTlZkTlZXd3pWbXhTUzAxSFJYaGFSV2hVWWtkb2IxVnFRbUZXYkZwMVkwWmthMkpHYkROV01qVlB
WREpLUm1OSWNGaGhNbEl6V1ZaYVQxSnNUbkppUm1ST1lteEtlVmRZY0VkV2JWWlhWRzVXVkdKR1NuQldiRnAzVjFaYVIxbDZSbWxOVjFKSVdXdG9
SMVpIUlhoalNFNVdZbFJHVkZZeWVHdGpiRnBWVW14a1RsWnVRalpYVkVKaFZqRmtSMWRZY0ZaaWEzQllWbXRXWVdWc1duRlNiR1JxVFZkU2VsbFZ
aSE5oVmxweVkwUmFWMDFYVVhkWFZtUlNaVlphY2xwR1pHbGlSWEJRVm0xNGExVXhXbk5WYkdoclUwZFNWRlJXV25OT1ZuQldZVWQwV0ZJd2NFaFp
NRnB2VjJzeFNGVnVXbGROYWtaSFdsWmFWMk5zY0VoU2JHUk9UVzFvU2xZeFVrcGxSazE0VTFob2FsSlhVbWhWYkZKWFZERldjMkZGVGxSTlZuQXd
WRlpTUTFack1WWk5WRkpYWWtkb2RsWXdXbXRUUjBaSFlrWndhVmRIYUc5V2JURTBZekpPYzJORmFGQldNMEpVV1d0b1EwNUdXbkpaTTJSUFZqQld
OVlV5ZEd0aGJFcFlZVVpvV21KR1NrTlVWbHBoVjBkTmVtRkdhRk5pUm05NFYxUkNZV0V4VW5SU2JrNVlZWHBHV0ZsWGRFdGpiRlkyVW1zNVUwMVd
jREZXVjNoUFlWWkplRk51V2xoV2JFcElXa1JHVDFZeFpISmhSM1JUVFVad2FGWnRNSGhWTVU1WFYyeG9hMUo2Ykc5VVZsWnpUbFpzVm1GRlRsZGl
WWEJKV1ZWV1QxbFdTa1pYYldoYVpXdGFNMVZzV2xka1IwNUdUbFprVGxaWGQzcFdiWGhUVXpBeFNGTlliRk5oTWxKVldXMXpNVlpXYkhKYVJ6bFh
Za1p3TUZwVmFHdFVhekZYWWtST1YwMXFWa3haYTFwTFpFWldkV0pIUmxOV01VbDZWMVphWVZsWFRuUlVhMXBwVW0xU2NGVnFSa1prTVdSWFZXdDB
VMDFXYkRSV1J6VlhWbTFLUmxOc2FGWmlSa3BZVmpGYVlWSXhiRFpTYld4T1ZqRktTVmRYZEc5U01WcElVbGhvYWxORk5WZFpiRkpIVmtaWmVXVkh
kR3BpUm5CV1ZXMTRhMVJzV25WUmFscFlWa1ZLYUZacVJtdFNNV1JaWTBaYWFXRXpRbGxXYlhSWFdWZFdjMWR1UmxOaVIxSnhWRlprVTJWc2JGWmF
TRTVvVm14d2VWa3dVbUZXTURGWVZWaGtXRlp0VWxOYVZscGhZMnh3UjFwSGJHbFNXRUpSVm0weE5HRXhWWGhYYms1V1lrZG9jbFV3WkZOV1JsSlh
WMjVPVDFadVFsZFpWV1F3VjBaSmQyTkZhRnBOUm5CMlZqSnplRk5IUmtabFJtUm9ZVEZ3YjFaWE1UUmhNazUwVm10a1lWSlVWbGhaYlhSTFUyeFp
lRlZyY0d4U2F6RTBWVEZvYjJGc1NsaFZiV2hXWWtaS1dGWkVSbGRqTWtaR1ZHeFNUbFp1UVhkV1JscFRVVEZhY2sxV1drNVdSa3BZV1d0a2IyUnN
XWGRYYlhSVVVqQmFTRmxyV25kaFZtUklZVWM1VjJKVVJUQlpla3BQWXpKT1JtRkdRbGROTUVwVlYxZDBiMUV3TlVkWGJrcGFUVEpTVUZadGVITk9
SbGw1VGxaT1YySlZjRWxaVlZwdlZqSkdjazVWT1ZWV2JIQm9WakJrVG1WdFJrZGhSazVwVW01Qk1WWXhXbGRaVjBWNFZXNVNVMkpyTlZsWmExcGh
WMFpzVlZOc1NrNVNiVkpZVmpKME1HRnJNVmRUYWtaWFVucEdkbFpVU2t0U01rNUhZa1prVTJKRmNFMVhWM0JIVkRGWmVGcElTbWhTTTJoVVZGVmF
kMkZHV25STlNHaFdUVlZzTkZaWE5VOVhSMHBXVjJ4a1ZtSllhRE5VVlZwaFYwZE9ObFpzYUdsU2JrSklWa2Q0VjFVeFdsaFRhMlJxVWpKb1YxUlZ
aRk5YUmxWM1YydDBhMUl3TlVkVWJGcHJWR3hhV0dRemNGZGlXR2hVVlhwQmVGTkdTbGxoUjBaVFZqSm9WbGRYZEd0aU1rbDRWbTVHVW1KVldsaFp
hMXAzVFZacmQxZHRkR2hOYTNCSVdXdFNUMVl3TVhGV2JFSlhVa1Z3VEZWdE1VOVRWMDVIWVVkb1RtSkZiRFpXTVZwaFdWWlJlRk51VGxWaWEzQnd
WVzB4YjFkR2JISlhibVJzVm0xU1dsa3dWbXRXTWtwWFVtcE9WVlpzV25wWlZWcExZMnMxV0U5V2NHaE5iV2hGVm1wR1lXRXhXWGhqUlZaU1lsaFN
jRlp0ZEZwTlJscHhVMnBTVjAxV1ZqVlZNblJyWVd4T1JrNVdaRnBpUmtwSVZtdGFVMVl4WkhOWGJYaFhUVVJSZVZaWE1UUmhNVkp6VjI1V1VtRnN
jRmxXYTFaTFlVWmFjVkp0ZEZOTlYxSXhWa2Q0VTJGRk1YUmhSemxYVmpOU1dGZFdaRTlqTVZwMVVteFNhRTB4U2xaV2JURTBVekF4UjJKR1dsaGh
lbXh3VldwQ2QxZHNiRlpYYTJSWFRXdFdORmt3Wkc5WFJscDBWV3hPWVZac2NHaFpNbmgzVWpGd1NHSkhiRk5YUlVwU1ZtMHdlRTVIUlhoV1dHaFl
WMGRvVjFsclpHOWpiRlYzV2taT1dGSnNTbGhYYTFKVFlrWmFjMk5HYkZWV2JGcHlWakJhUzJOdFNrVlViR1JwVjBWS1ZWWnFTbnBsUmtsNFZHNU9
VbUpIVW05WlZFNURVMVprVlZOcVVtaE5helV3Vm0xMGExbFdTWGxoUnpsVlZrVktURlpYZUdGak1WWnlWRzFvVGxaWGR6QldWRVp2WXpGVmVWSll
iR2hUUlVwWFdXeG9UbVZHYTNkWGJrNVhWbXRhTVZkclZURmhWa3AxVVZoa1YxSnNjRlJWVkVaaFkyc3hWMWRyTlZkU2EzQlpWbTB3ZUdJeVZuTlh
iazVZWWxoU1ZWVnFSbUZUUmxsNVpVaGtWMDFWY0ZoWmFrNTNWMFphYzFkdGFGZGhhM0JRVm1wR1UyUldWbk5SYkdScFZtdHdWbFl4WkRCV01sRjR
Xa2hPV0dFeWFITlZhazVEVlVaYWRHVklUazlTYkd3MVZHeFZOV0ZIU2taalJteGFWbFp3ZWxacVNrWmxSbHBaWVVkR1UwMHlhRFpXYlhCSFdWWmt
XRkpyWkdoU2F6VndWVzB3TlU1R1dYaFZhMDVhVmpCV05GWlhOVTlYUm1SSVpVYzVWbUV4V2pOV01GcHpWMGRTUm1SSGNHbFNiR3Q1Vmxjd2VFMUd
XWGROVm1ScVVrVmFWMVJYTlc5U1JscHhVMnQwVkZacldqRlhhMXByWVZaa1IxTnNiRmRpVkVJMFZsY3hWMUl4Y0VsV2JFNXBVbGhDZDFadGVHRmt
Na1pIVjI1U1RsTkhhRmRVVmxVeFYwWlplV1ZIT1doTlZXOHlXV3RqTlZaV1duTlhibkJWWWtad2VsWnRNVWRTYkZKeldrZHNWMWRGU2t0V01WcFh
WakZSZUdKR1pGUmhNWEJaV1d4a2IxbFdjRmhrUjBaT1RWWmFlbFl5ZUd0aE1VbDNUbFZrVldKR2NISldSM2hoVjBkUmVtTkdaR2xYUjJoVlZsaHd
RbVZHVGtkVWJHeHBVbXMxYjFSWGVFdFdiR1JZVFZod1RsWnNjRmhaYTJoUFZqSktWbGRzYUZwaE1YQXpWRlZhZDFadFJraGtSbFpvWld0YVdWZFV
RbTlqTVZsM1RWaEdVMkV5YUdGV2FrNXZZVVpyZVUxVk9WTldhM0I2V1ZWa2IxUnNaRVpUYkVwWFlsaFNjbFJyV25OWFJsSjFWV3hXYUUxV2NGbFh
WM1JyVlRGYVIxVnNWbFJpVkd4d1ZGWmFkMlZXV2xoa1JFSldUVVJHV0ZsclVsTldNVW8yVm14b1YySlVSa3hWYlhoUFl6SkdSMWR0YkZOaE0wSmF
WbTF3UjFsWFJYaFhXR2hYWW10d2FGVnFUbE5VTVd4VlVtNWtWRlpzY0hoVk1uQlRWakF4VjFacVZsWmlSMmgyV1ZkNFQxSnJOVmRhUm5CcFVtdHd
TVlp0ZEdGa01XUklWbXRzVldKSFVuQlZha1pMVG14YWNsa3phR2xOVmxZMFZqSjBZVmRIUm5OalJtaFhZVEZhZVZwVlduTldWa3B6WTBkNFUySld
TbUZYVkVKaFdWZEdWMWRZYkdoU2JXaFpXV3RrVW1ReFpGZFhiazVYVFdzMVNGWXllRzloVmtsNFUyNW9WMUp0VVhkWFZscEtaVVpXV1dGR2FHbFh
Sa3AyVmxkd1IxbFhWbk5YV0d4cVVqQmFXRlJWVWtkWFZscFhZVWQwV0ZKc2NEQldWM2hQV1ZaYWMyTkhhRnBOYm1nelZXcEdkMU5IU2toaVJrNVl
VbFZ3VTFadE1IZGxSVFZJVWxob1YxZEhhRmxXTUdSdlZqRnNjbHBIT1dwaVJsWXpWMnRhVDFkR1NuTlRiR2hYVFdwV2NsWkhlRXRrUjFKRlZHeG9
hRTFXY0hsV2FrSmhVMjFSZVZScldtaFNia0pQVlcwMVEwMXNXbkZUYm5Cc1VtczFTRlp0TlZkWFIwcElWV3M1V21KVVJuWlpha1poWTFaR2RGSnN
aRTVoZWxZMlYxUkNWMkl4VlhsVGEyaFdZa2RvVmxadGVHRk5NVnBZWlVkR2FrMVlRa3BYYTFwVFZHeGFWVkpVUWxkV1JWcDJXWHBHVm1WV1NsbGl
SbHBwVmtkNFdGZFhlRzlpTVZKSFYyNUtXR0pWV25GVVYzUmhVakZhU0dWR1RsVmlSbkF4VlZkd1UxWXhXalpSYWxKV1lXdGFhRmt5YzNoV01XUnl
UbFprVTJFelFscFdiVEIzWlVkSmVWWnVUbGhpYXpWWldXeG9VMVpXVm5GUmJVWlVWbTE0VjFZeU1VZFdWMHBHWTBSR1ZsWjZRVEZXYWtwTFZsWkt
WVkZzY0d4aE0wSlJWMWh3UjJReFRsZFVibEpyVW1zMVQxUlZWbmRXYkZsNFdrUkNXbFl4UmpOVWJGWnJWMGRLU0dGRk9WZGhNVnBNVmtSR1YyUkh
Wa2xVYXpsVFlrZDNNVlpIZUZaT1YwWklVMnRhYWxKdGVHRldiRnAzWkd4YWNWTnJaR3BoZWxaWVZsZDRhMVV4V25WUmFscFhZbGhvYUZWcVJtdFh

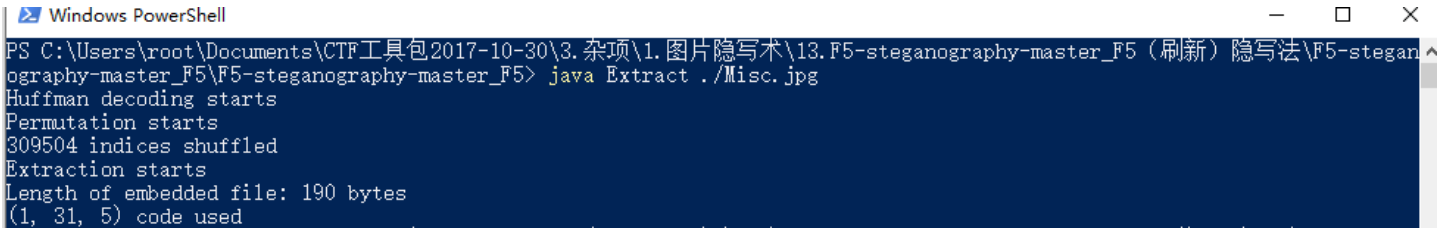
```
Sa3B5V2taV2FWSXhTblpXUmxKRFUyc3hjMWR1VW10U00xSlFWVzB4TkZkR1duTmhTRTVYVWpCV05Ga3dXbk5XTURGSVlVVlNWMDFHY0doWmVrWnJaRlp3UjFSck5WZGhNV3QzVm0xd1MwMUdVWGhYYmxKVVlURndWVmxyV25kV2JGcHpWMnRrVGsxV1draFZiRkp6VlZaV1ZVMUVhejA9"
for i in range (0,128):
    m = base64.b64decode(m)
    print(m)
```



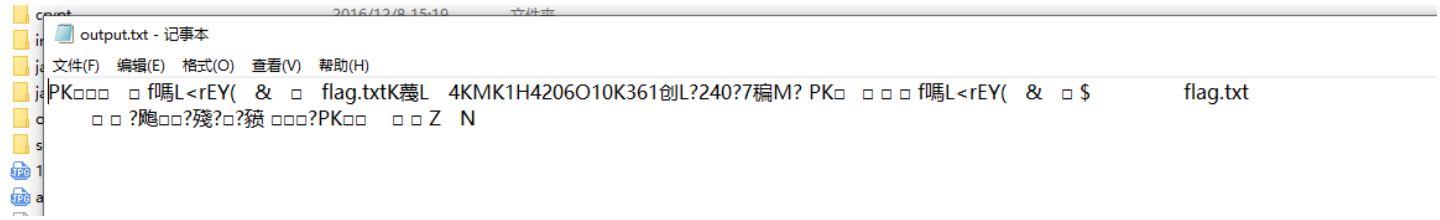
根据报错知道得到了正确的原文：comEON_YOuAreSOSoS0great。
这个字符串可以用来解码18394.
解压出来是一个mp3。打开听是莫斯密码。-- — .-. … . … … …- . .-. -.-- …- . .-. -.-- . .- … -.–
解码得到flag{morseisveryveryeasy}

23.刷新过的图片

这题提示了是F5隐写。
直接F5-steganography一把梭：

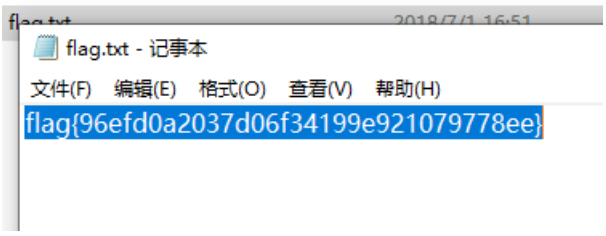


得到一个output.txt:



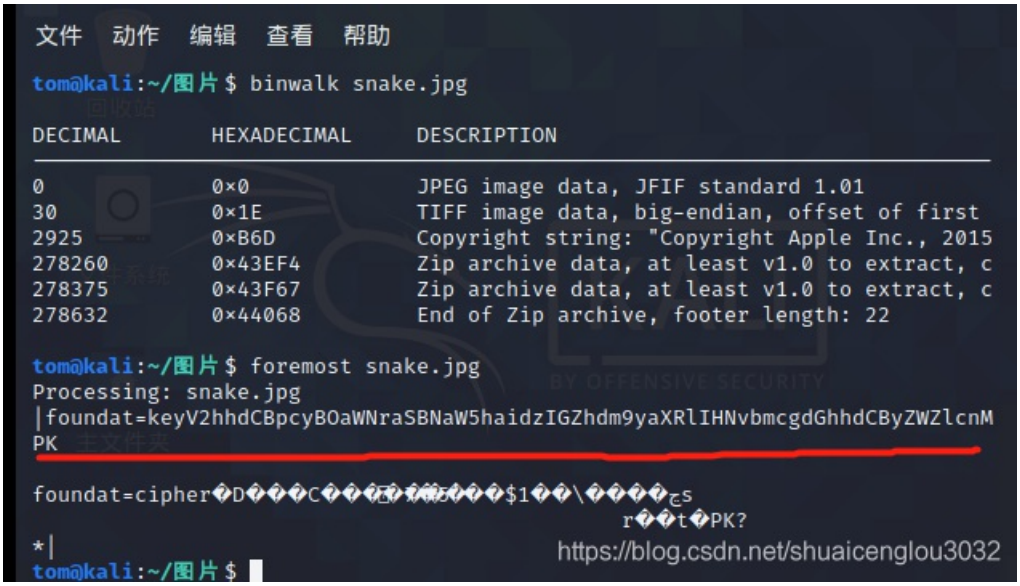
看到文件头的PK，改成zip，解压需要密码：

winhex打开看到50 4B 01 02 1F 00 14 00 01 00
猜测是伪加密，这里尝试把01改成00。
成功解压：

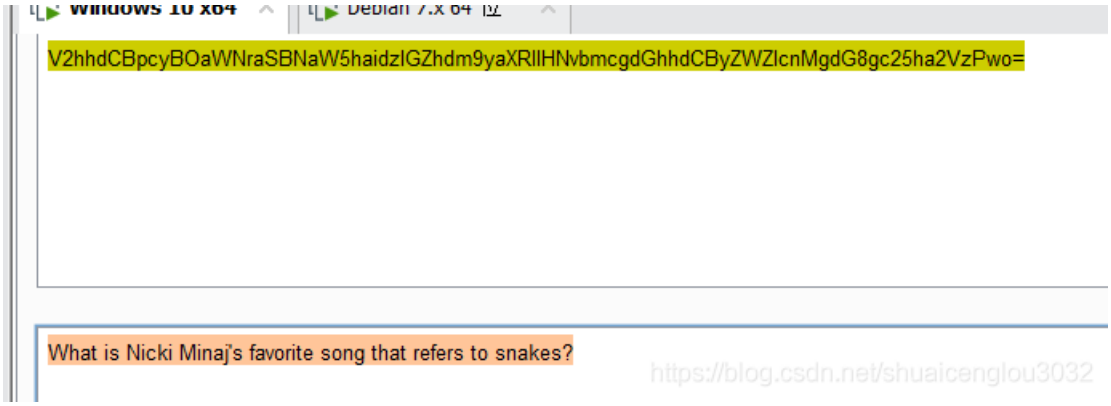


24.snake

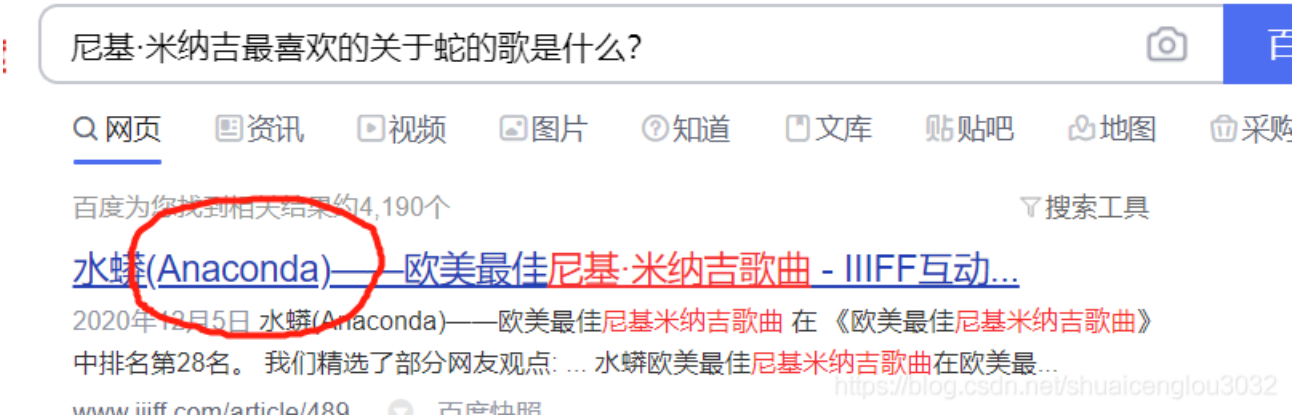
先上stegsolver无果。
binwalk分析：



base64解密一下：



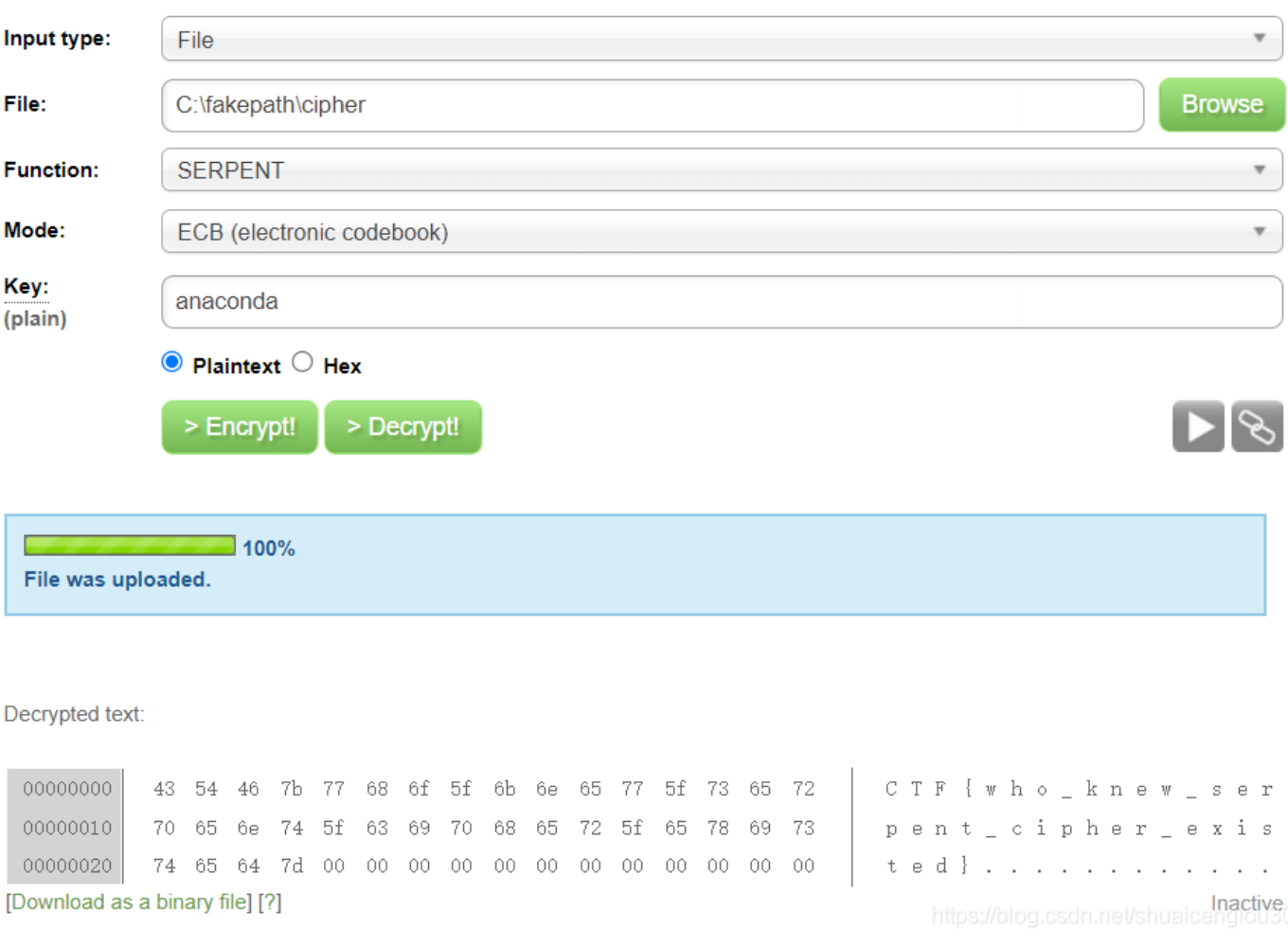
百度：



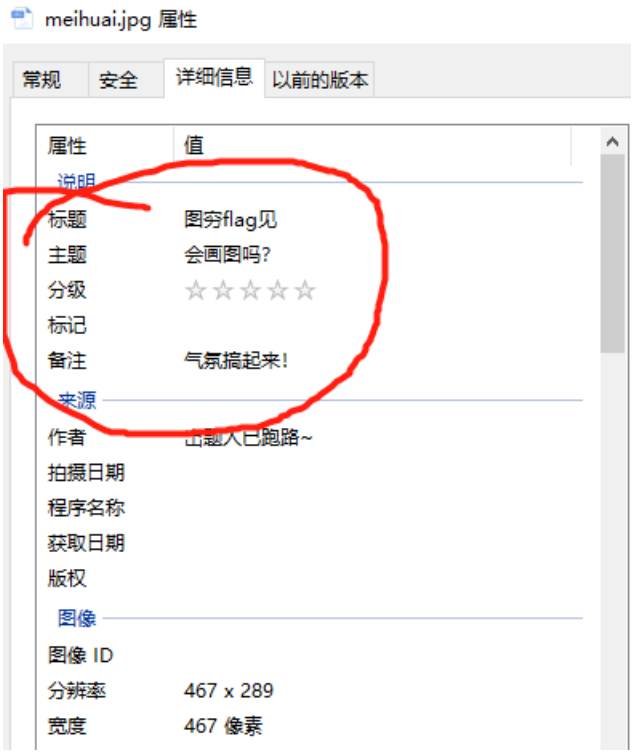
这里我们得到了一个key:anaconda
但是有什么用呢。提取出来的压缩包解压得到一个key和一个cipher。

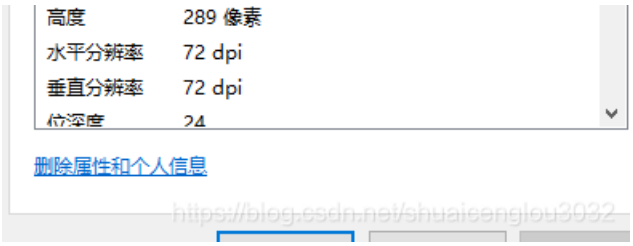
key就是之前得到的base64解码结果。

百（看）度（W）了§知道对于蛇这个名词，在英语中还有一个翻译：Serpent。Serpent是一个加密算法，于是对 cipher文件进行 Serpent解密：



24.梅花香之苦寒来

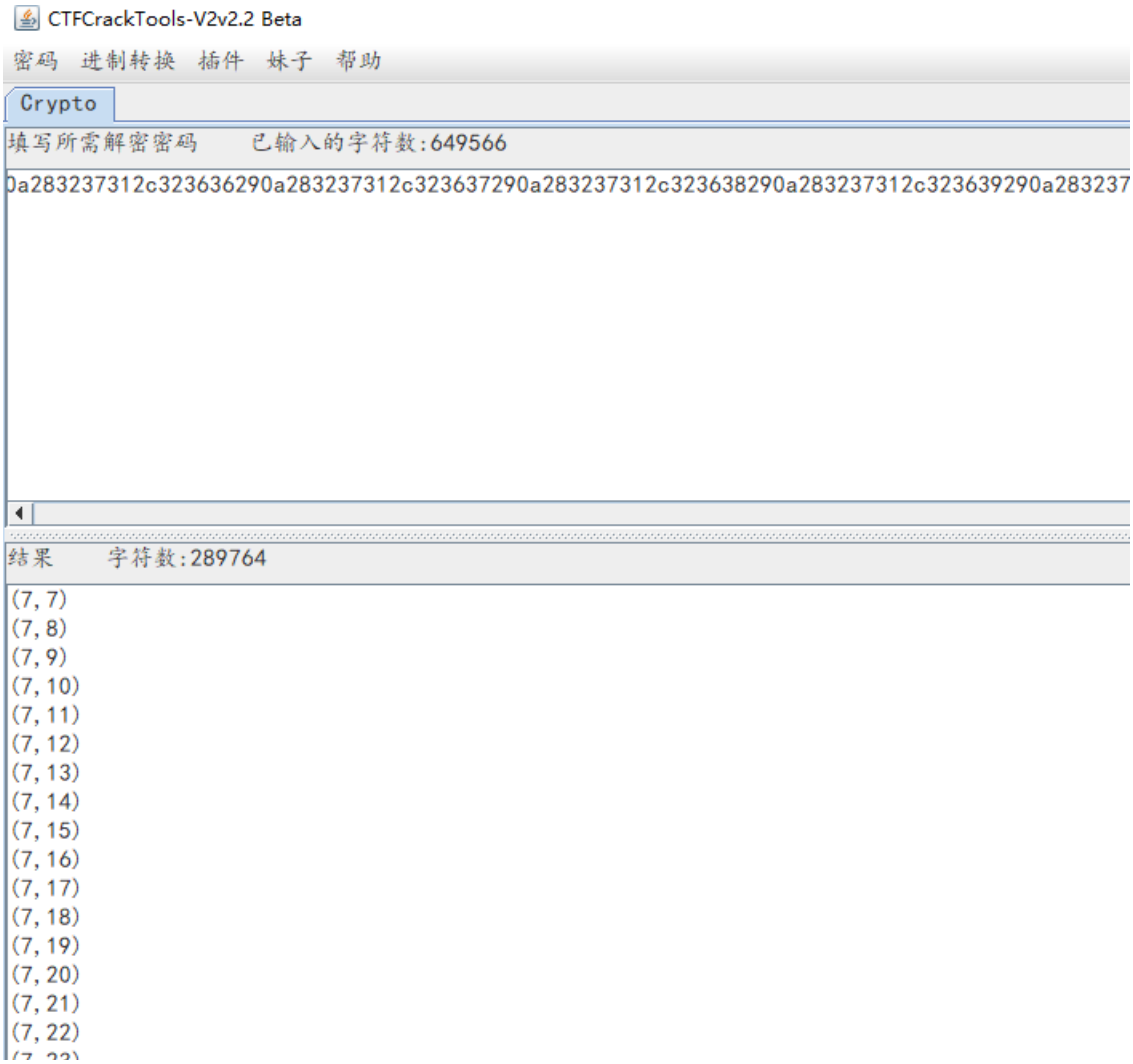




winhex看看：

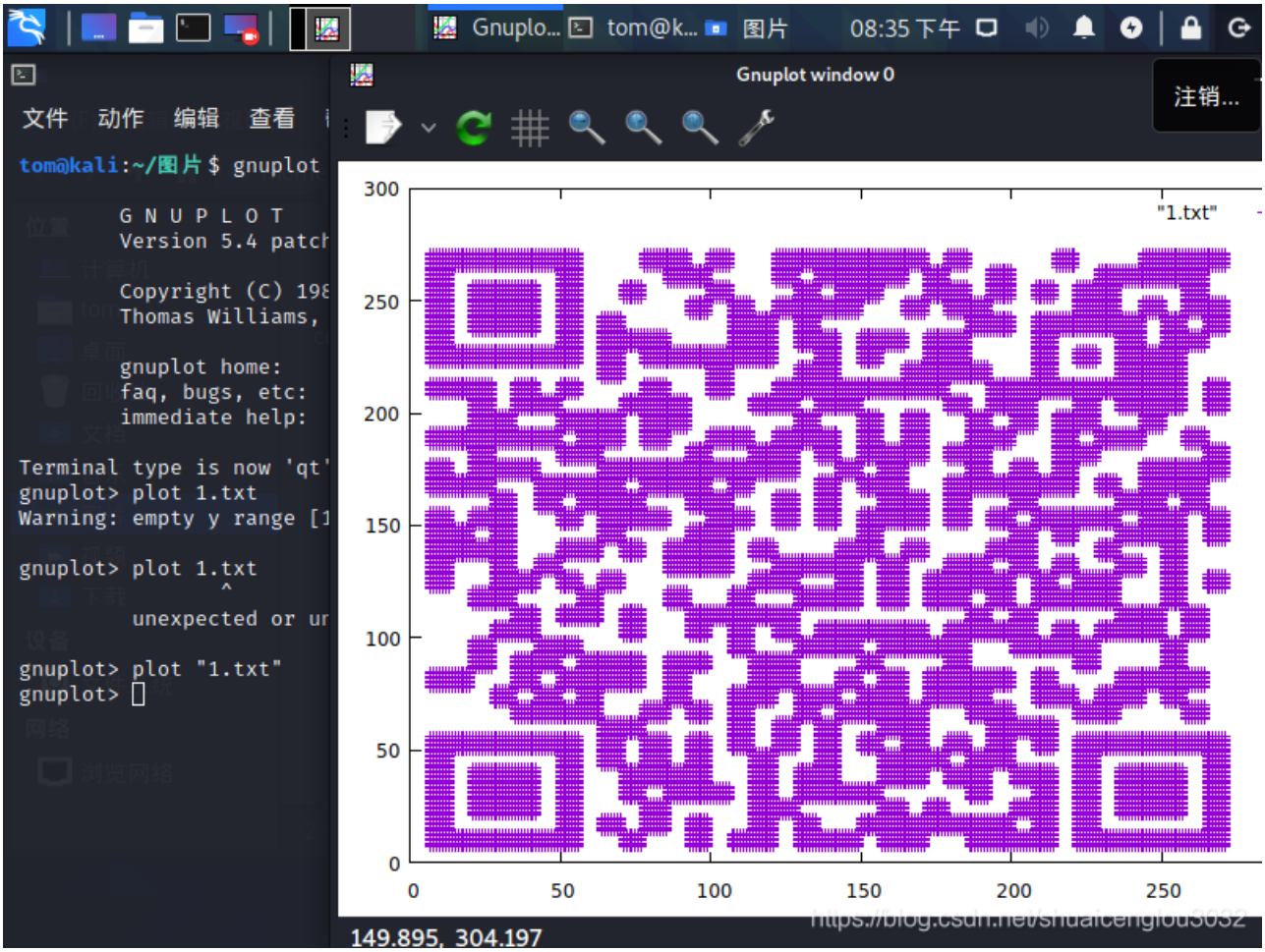
```
8 42 12 A8 3F 2A  63 92 10 A4 0F 92 A6 A1  R§˜B ¨?*c' ¤ '¦¡
1 C9 14 21 52 10  DC AC F9 A1 0B 30 6E 84   Þ É !R Ü¬ù¡ 0n„
2 93 B9 42 11 54  36 49 08 44 07 E6 F6 50  !i ‰¹B T6I D æöP
8 42 95 60 3C 95  0D 8A 10 80 6E E8 76 E8  î^¨B•`<• Š €nèvè
3 98 52 EE 68 42  80 3F 2A 63 72 84 20 A6  B~ ˜RîhB€?*cr„ ¦
1 0B 47 E9 1D D5  72 42 13 F4 09 0E 48 42  ì‡! Gé ÕrB ô  HB
7 2F 54 21 11 68  42 15 02 10 85 00 84 21   ¢§/T! hB   … „!
1 50 24 84 28 1A  10 84 02 10 85 40 84 21   „!P$„(  „  …@„!
1 07 FF D9 32 38  33 37 32 63 33 37 32 39   „! ÿÙ28372c3729
2 38 33 37 32 63  33 38 32 39 30 61 32 38  0a28372c38290a28
2 63 33 39 32 39  30 61 32 38 33 37 32 63  372c39290a28372c
3 30 32 39 30 61  32 38 33 37 32 63 33 31  3130290a28372c31
2 39 30 61 32 38  33 37 32 63 33 31 33 32  31290a28372c3132
0 61 32 38 33 37  32 63 33 31 33 33 32 39  290a28372c313329
2 38 33 37 32 63  33 31 33 34 32 39 30 61  0a28372c3134290a
3 37 32 63 33 31  33 35 32 39 30 61 32 38  28372c3135290a28
2 63 33 31 33 36  32 39 30 61 32 38 33 37  372c3136290a2837
3 31 33 37 32 39  30 61 32 38 33 37 32 63  2c3137290a28372c
3 38 32 39 30 61  32 38 33 37 32 63 33 31  3138290a28372c31
2 39 30 61 32 38  33 37 32 63 33 32 33 30  39290a28372c3230
```

文件结束之后是一大坨看似十六进制的东西，复制出来转字符串。



```
(7, 23)
(7, 24)
(7, 25)
(7, 26)
(7, 27)
(7, 28)
```


得到类似坐标的东西。
先把()去除，逗号换成空格。这样gnuplot就能识别。
gnuplot绘图：



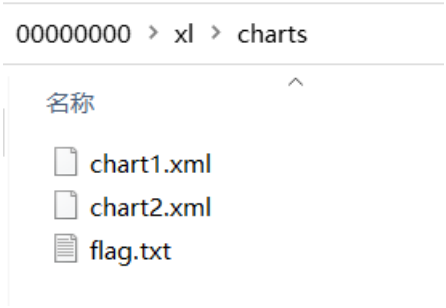
扫码得到flag。

25.[BJDCTF2020]认真你就输了

binwalk:

```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             Zip archive data, at least v2.0 to extract, compressed size: 19
uncompressed size: 17, name: xl/charts/flag.txt
67            0x43            Zip archive data, at least v1.0 to extract, name: docProps/
106           0x6A            Zip archive data, at least v2.0 to extract, compressed size: 40
 uncompressed size: 872, name: docProps/app.xml
556           0x22C           Zip archive data, at least v2.0 to extract, compressed size: 32
 uncompressed size: 608, name: docProps/core.xml
925           0x39D           Zip archive data, at least v1.0 to extract, name: xl/
958           0x3BE           Zip archive data, at least v2.0 to extract, compressed size: 14
, uncompressed size: 9895, name: xl/calcChain.xml
2445          0x98D           Zip archive data, at least v1.0 to extract, name: xl/charts/
2485          0x9B5           Zip archive data, at least v2.0 to extract, compressed size: 10
, uncompressed size: 3398, name: xl/charts/chart1.xml
3599          0xE0F           Zip archive data, at least v2.0 to extract, compressed size: 99
 uncompressed size: 3001, name: xl/charts/chart2.xml
4645          0x1225          Zip archive data, at least v1.0 to extract, name: xl/drawings/
4687          0x124F          Zip archive data, at least v2.0 to extract, compressed size: 39
 uncompressed size: 1013, name: xl/drawings/drawing1.xml
5138          0x1412          Zip archive data, at least v2.0 to extract, compressed size: 39
 uncompressed size: 1014, name: xl/drawings/drawing2.xml
5589          0x15D5          Zip archive data, at least v1.0 to extract, name: xl/drawings/_
ls/
5637          0x1605          Zip archive data, at least v2.0 to extract, compressed size: 17
 uncompressed size: 293, name: xl/drawings/_rels/drawing1.xml.rels
5881          0x16F9          Zip archive data, at least v2.0 to extract, compressed size: 17
 uncompressed size: 293, name: xl/drawings/_rels/drawing2.xml.rels
6125          0x17ED          Zip archive data, at least v2.0 to extract, compressed size: 27
```


分离，找一下文件得到佛莱格：



26.[GXYCTF2019]佛系青年

zip包是伪加密，修改标志位之后txt里面是与佛论禅：

27.[ACTF新生赛2020]outguess

压缩包是伪加密，直接改标志位解压。
解压出来查看图片详情得到

得到密码abc,然后outguess一把梭：

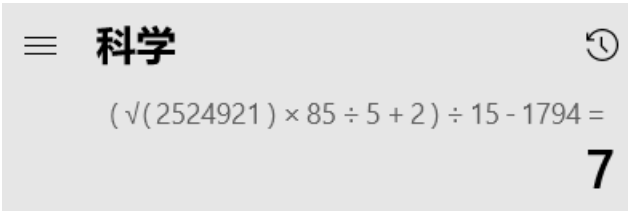
```
root@kali:/home/tom/视频/outguess-master# ./outguess  -k "abc" -r mmm.jpg out.txt
Reading mmm.jpg....
Extracting usable bits:   17550 bits
Steg retrieve: seed: 93, len: 23
root@kali:/home/tom/视频/outguess-master#
```

```
ACTF{gue33_Gu3Ss!2020}
```

28.[安洵杯 2019]easy misc

这题综合性比较强。
下载下来是三个莫名其妙的文件：

计算一下这个算式：



flag in 7+NNULLULL,
直接用这个当密码失败，大（kan）胆(le)猜(write)测(up)7是前面掩码数量。爆破一下：

解压看下：

```
a = dIW
b = sSD
c = adE
d = jVf
e = QW8
f = SA=
g = iBt
```

```
h = 5RE
i = tRQ
j = SPA
k = 8DS
l = XiE
m = S8S
n = MkF
o = T9p
p = PS5
q = E/S
r = -sd
s = SQW
t = obW
u = /WS
v = SD9
w = cw=
x = ASD
y = FTa
z = AE7
```

盲猜这个是字频隐写。

再分析下那张图片：

binwalk发现包含了2张图片，foremost分离出来：

这里是盲水印,使用工具提取一下（这里需要加一个参数–oldseed，不然提取出来的没法看）：

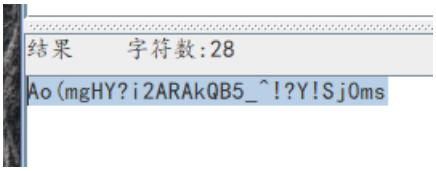
```
PS E:\CTFTools\misc\盲水印检测\BlindWaterMark-master> python3 bwmforpy3.py decode 00000000.png 00000232.png fuck.png
image<00000000.png> + image(encoded)<00000232.png> -> watermark<fuck.png>
PS E:\CTFTools\misc\盲水印检测\BlindWaterMark-master> python3 bwmforpy3.py decode 00000000.png 00000232.png fuck.png --o
ldseed
image<00000000.png> + image(encoded)<00000232.png> -> watermark<fuck.png>
PS E:\CTFTools\misc\盲水印检测\BlindWaterMark-master>
```

此外hint.txt还提示hint:取前16个字符。

统计11.txt里的字符出现频率得到：

```
('ps1: ', ('e', 39628))
('ps2: ', ('t', 27993))
('ps3: ', ('a', 25887))
('ps4: ', ('o', 25809))
('ps5: ', ('n', 21337))
('ps6: ', ('r', 20990))
('ps7: ', ('h', 19535))
('ps8: ', ('i', 19422))
('ps9: ', ('s', 18870))
('ps10: ', ('d', 15932))
('ps11: ', ('l', 14385))
('ps12: ', ('u', 9562))
('ps13: ', ('y', 8293))
('ps14: ', ('g', 8127))
('ps15: ', ('w', 7744))
('ps16: ', ('m', 6729))
('ps17: ', ('f', 6431))
('ps18: ', ('c', 6403))
('ps19: ', ('b', 4980))
('ps20: ', ('p', 4909))
('ps21: ', ('k', 3930))
```

取前16个字符得到etaonrhisdluygw.
再对照之前解压出来的码表得到：
QW8obWdlWT9pMkFSQWtRQjVfXiE/WSFTajBtcw==
base64看下：



base85解码：
flag{have_a_good_day1}

29.[UTCTF2020]zero

这题是零宽度字符隐写，把题目给的文件放进winhex就能看见汉字中间夹杂了很多不可见字符。