[CTF] 攻防世界MISC高手区部分题目WriteUp

原创

berdb 于 2020-11-26 21:51:25 发布 1371 收藏 4

分类专栏： 笔记

本文链接：https://blog.csdn.net/qq_37969220/article/details/110193857

版权

笔记 专栏收录该内容

3 篇文章 0 订阅

订阅专栏

记录一些有意思的题目

目录索引

- 双色块
- flag_universe
- 3-11
- 互相伤害！！！
- Miscellaneous-300
- intoU
- Just-No-One
- Disk
- picture2
- MulTzor
- 攻防世界MISC高手区部分题目WriteUp（2）

双色块

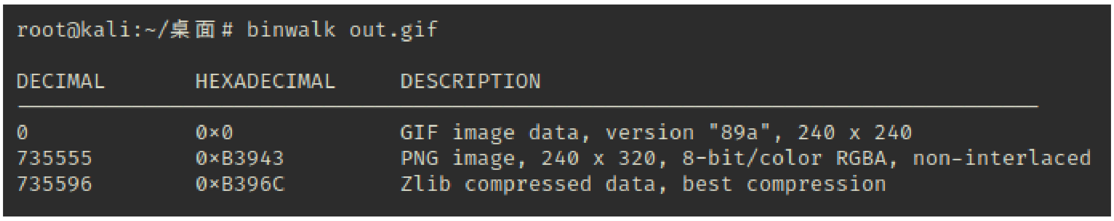
下载附件解压出来是个gif文件，仔细观察一下，发现两种色块应该分别代表01。写段Python脚本把数据提取出来

```
from PIL import Image
# 读取图片
image = Image.open('C:\\Users\\28919\\Desktop\\out.gif')
# 存放分离出的图片的路径
split = 'C:\\Users\\28919\\Desktop\\gif\\'
# 循环读取gif的每一帧
try:
 while True:
  # 当前位置
  current = image.tell()
  # 保存图片
  image.save(split + str(current) + '.png')
  # 移动到下一张
  image.seek(current + 1)
except:
 pass
# 储存提取出的字符
string = ''
# 直接把2进制转为10进制储存在line里
line = 0
i = 0
# 图片里每一个色块是10×10像素大小，总共24×24个色块
for y in range(24):
 for x in range(24):
  # 载入一张图片
  pix = Image.open(split+str(i)+'.png').convert('RGBA').load()
  # 读取色块的颜色值
  r, g, b, p = pix[x*10,y*10]
  if g == 255:
   # 0
   line = line << 1
  if r == 255 and b == 255:
   # 1
   line = (line << 1) + 1
  i += 1
  # 凑够8位就把line转为字符储存在string里
  if i%8 == 0:
   string += chr(line)
   line = 0
print(string)
```

运行后得到：

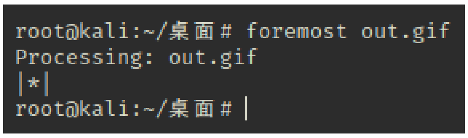
> o8DlxK+H8wsiXe/ERFpAMaBPiIcj1sHyGOMmQDkK+uXsVZgre5DSXw==hhhhhhhhhhhhhhhh

去掉后面那串无意义的hhhh就应该是个base64加密，但是直接解码出来还是乱码，应该在哪里还有别的东西。把图片丢到Kali里用binwalk命令跑一下：



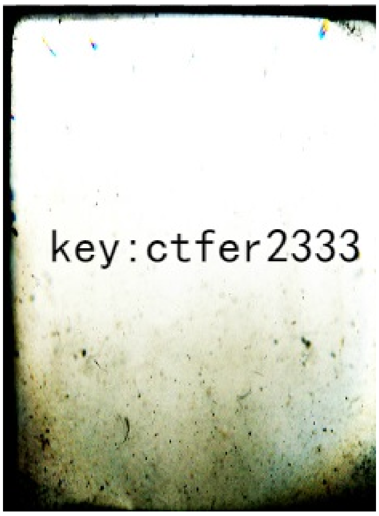
```
root@kali:~/桌面# binwalk out.gif

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0×0             GIF image data, version "89a", 240 x 240
735555        0×B3943         PNG image, 240 x 320, 8-bit/color RGBA, non-interlaced
735596        0×B396C         Zlib compressed data, best compression
```

果然还有东西，用foremost命令提取出来：



```
root@kali:~/桌面# foremost out.gif
Processing: out.gif
|*|
root@kali:~/桌面# |
```

得到一张图片：

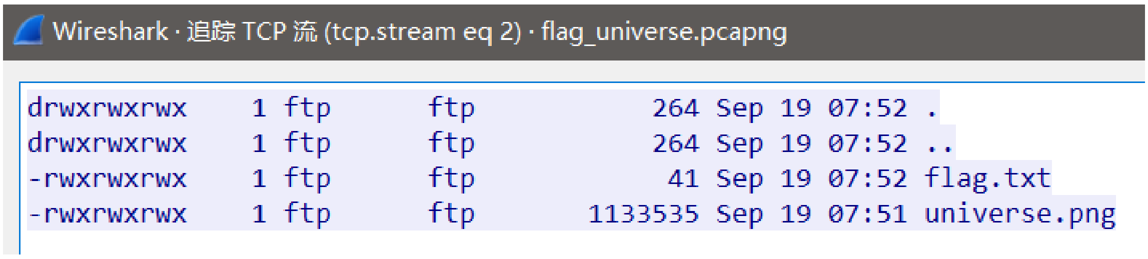


得到一个密码ctfer2333，用DES解密出来就得到flag了，解密网站：http://tool.chacuo.net/cryptdes

> flag{2ce3b416457d4380dc9a6149858f71db}

flag_universe

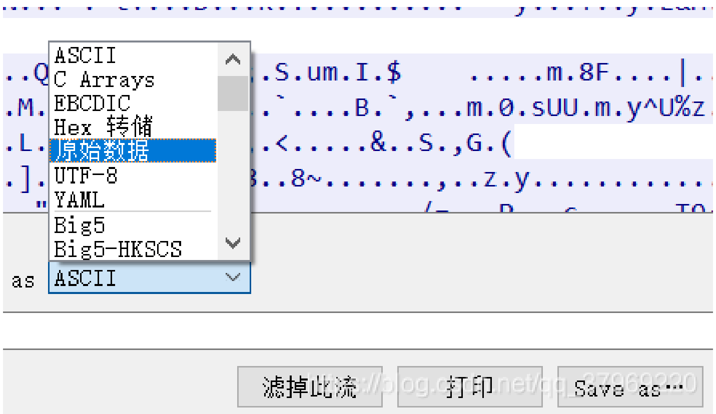
下载附件解压出来是个流量包，用wireshark打开，追踪TCP流，在第二个流里发现存在一个flag.txt文件



Wireshark · 追踪 TCP 流 (tcp.stream eq 2) · flag_universe.pcapng

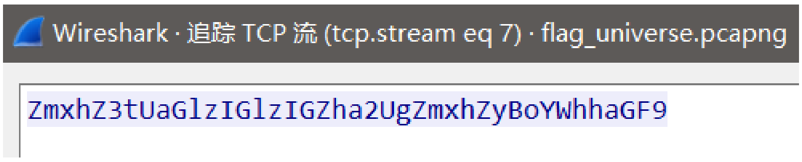
```
drwxrwxrwx    1 ftp      ftp           264 Sep 19 07:52 .
drwxrwxrwx    1 ftp      ftp           264 Sep 19 07:52 ..
-rwxrwxrwx    1 ftp      ftp            41 Sep 19 07:52 flag.txt
-rwxrwxrwx    1 ftp      ftp       1133535 Sep 19 07:51 universe.png
```

继续往后翻，在第三个流里面发现一个png文件，在下面选择原始数据，保存。



```
..Q       .S.um.I.$    .....m.8F....|.
.M.       .`....B.`,...m.0.sUU.m.y^U%z
.L.       .<.....&..S.,G.(
.].       3..8~........,..z.y..........
```

继续翻，在第七个流里面发现一段base64文本，解码后是一个假flag



Wireshark · 追踪 TCP 流 (tcp.stream eq 7) · flag_universe.pcapng

```
ZmxhZ3tUaGlzIGlzIGZha2UgZmxhZyBoYWhhaGF9
```

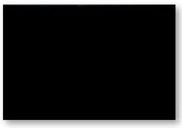
> flag{This is fake flag hahaha}

全都保存出来后得到了这么几个文件：

a.png

b.png

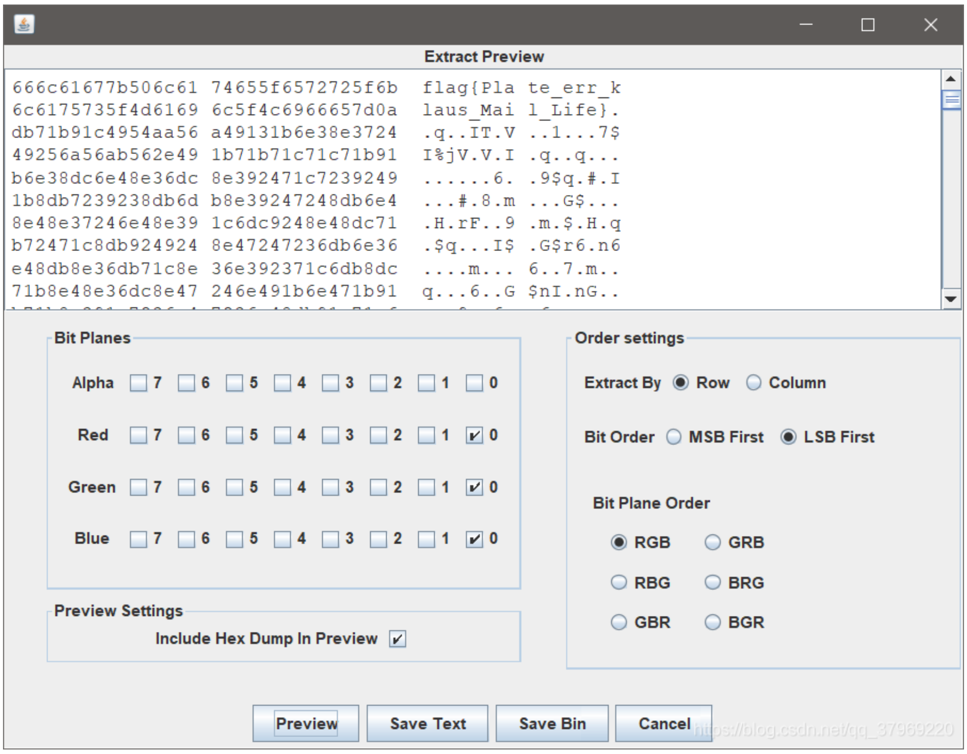
c.png


d.png

e.png

f.png

逐个检查这几张图片，发现在最后一张里存在LSB隐写，用Stegsolve打开文件，一番操作之后flag就出来了


Extract Preview

```
666c61677b506c61 74655f6572725f6b  flag{Pla te_err_k
6c6175735f4d6169 6c5f4c6966657d0a  laus_Mai l_Life}.
db71b91c4954aa56 a49131b6e38e3724  .q..IT.V ..1...7$
49256a56ab562e49 1b71b71c71c71b91  I%jV.V.I .q..q...
b6e38dc6e48e36dc 8e392471c7239249  ......6. .9$q.#.I
1b8db7239238db6d b8e39247248db6e4  ...#.8.m ...G$...
8e48e37246e48e39 1c6dc9248e48dc71  .H.rF..9 .m.$.H.q
b72471c8db924924 8e47247236db6e36  .$q...I$ .G$r6.n6
e48db8e36db71c8e 36e392371c6db8dc  ....m... 6..7.m..
71b8e48e36dc8e47 246e491b6e471b91  q...6..G $nI.nG..
```

Bit Planes

Alpha □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 □ 0

Red □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 ☑ 0

Green □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 ☑ 0

Blue □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 ☑ 0

Preview Settings

Include Hex Dump In Preview ☑

Order settings

Extract By ● Row ○ Column

Bit Order ○ MSB First ● LSB First

Bit Plane Order

● RGB ○ GRB

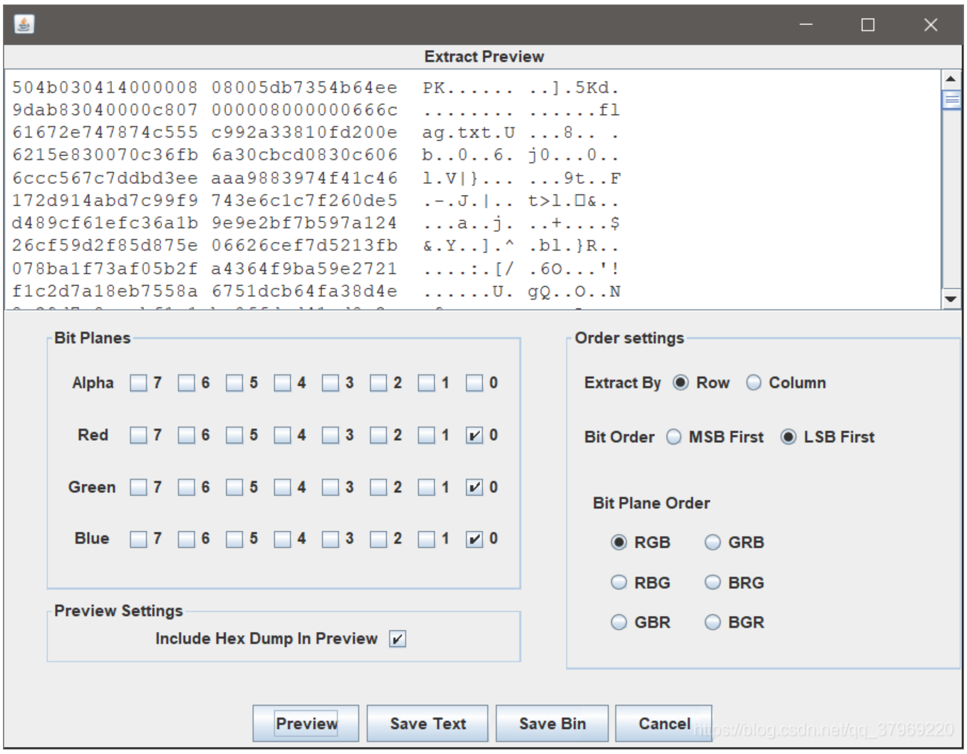
○ RBG ○ BRG

○ GBR ○ BGR

> flag{Plate_err_klaus_Mail_Life}

3-11

下载得到一张图片，用Stegsolve打开，发现存在LSB隐写，开头为zip的文件头，直接保存用压缩软件打开。



Extract Preview

```
504b030414000008 08005db7354b64ee  PK...... ..].5Kd.
9dab83040000c807 000008000000666c  ........ ......fl
61672e747874c555 c992a33810fd200e  ag.txt.U ...8.. .
6215e830070c36fb 6a30cbcd0830c606  b..0..6. j0...0..
6ccc567c7ddbd3ee aaa9883974f41c46  l.V|}... ...9t..F
172d914abd7c99f9 743e6c1c7f260de5  .-.J.|.. t>l.□&..
d489cf61efc36a1b 9e9e2bf7b597a124  ...a..j. ..+....$
26cf59d2f85d875e 06626cef7d5213fb  &.Y..].^ .bl.}R..
078ba1f73af05b2f a4364f9ba59e2721  ....:.[/ .6O...'!
f1c2d7a18eb7558a 6751dcb64fa38d4e  ......U. gQ..O..N
```

Bit Planes

Alpha □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 □ 0

Red □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 ☑ 0

Green □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 ☑ 0

Blue □ 7 □ 6 □ 5 □ 4 □ 3 □ 2 □ 1 ☑ 0

Preview Settings

Include Hex Dump In Preview ☑

Order settings

Extract By ● Row ○ Column

Bit Order ○ MSB First ● LSB First

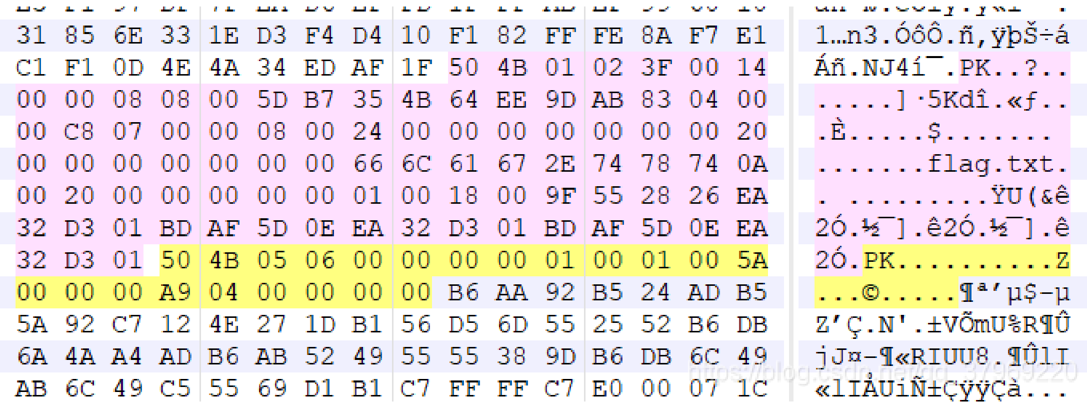
Bit Plane Order

● RGB ○ GRB

○ RBG ○ BRG

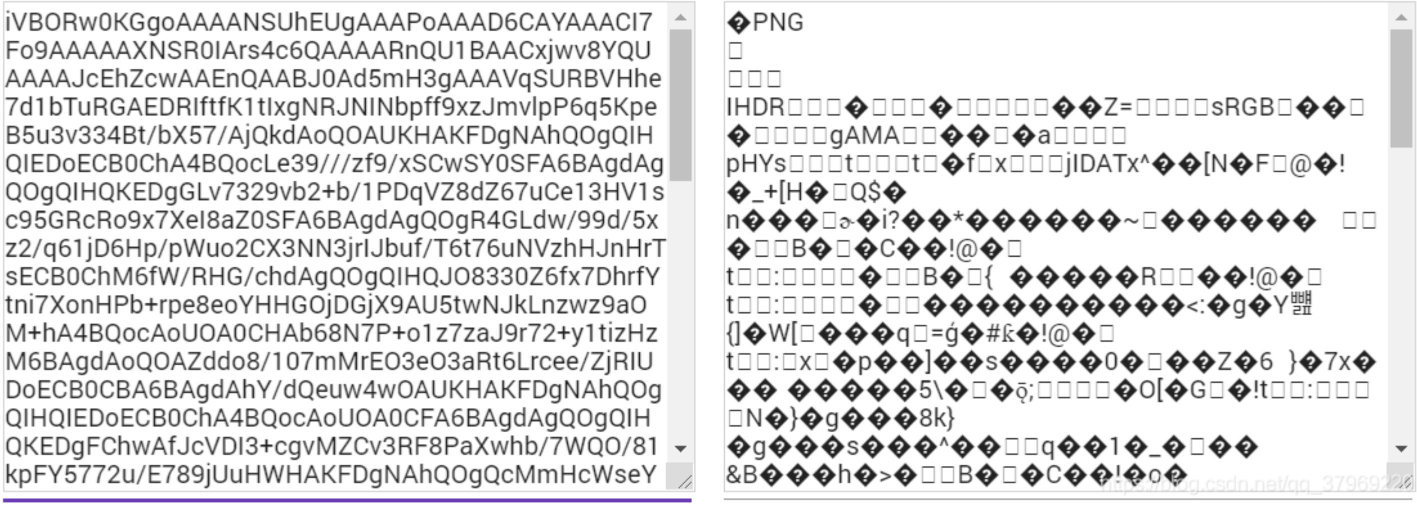
○ GBR ○ BGR

没有任何加密，但解压时提示压缩文件损坏，用010editor看一下，文件尾后面存在无用数据，直接删掉，成功解压。



```
31 85 6E 33 1E D3 F4 D4 10 F1 82 FF FE 8A F7 E1   1…n3.Óôô.ñ,ÿþŠ÷á
C1 F1 0D 4E 4A 34 ED AF 1F 50 4B 01 02 3F 00 14   Áñ.NJ4í¯.PK..?..
00 00 08 08 00 5D B7 35 4B 64 EE 9D AB 83 04 00   .....]·5Kdî.«ƒ..
00 C8 07 00 00 08 00 24 00 00 00 00 00 00 00 20   .È.....$....... 
00 00 00 00 00 00 00 66 6C 61 67 2E 74 78 74 0A   .......flag.txt.
00 20 00 00 00 00 00 01 00 18 00 9F 55 28 26 EA   . .........ŸU(&ê
32 D3 01 BD AF 5D 0E EA 32 D3 01 BD AF 5D 0E EA   2Ó.½¯].ê2Ó.½¯].ê
32 D3 01 50 4B 05 06 00 00 00 00 01 00 01 00 5A   2Ó.PK..........Z
00 00 00 A9 04 00 00 00 00 B6 AA 92 B5 24 AD B5   ...©.....¶ª’µ$µ
5A 92 C7 12 4E 27 1D B1 56 D5 6D 55 25 52 B6 DB   Z’Ç.N'.±VÕmU%R¶Û
6A 4A A4 AD B6 AB 52 49 55 55 38 9D B6 DB 6C 49   jJ¤¶«RIUU8.¶ÛlI
AB 6C 49 C5 55 69 D1 B1 C7 FF FF C7 E0 00 07 1C   «lIÅUiÑ±ÇÿÿÇà...
```

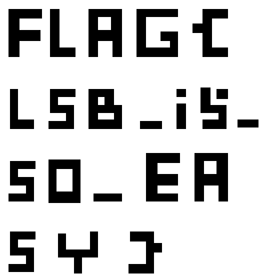
解压后的文件打开发现是base64加密，解密后能看到是一个png文件。



```
iVBORw0KGgoAAAANSUhEUgAAAPoAAAD6CAYAAACI7
Fo9AAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQU
AAAAJcEhZcwAAEnQAABJ0Ad5mH3gAAAVqSURBVHhe
7d1bTuRGAEDRIftfK1tIxgNRJNINbpff9xzJmvlpP6q5Kpe
B5u3v334Bt/bX57/AjQkdAoQOAUKHAKFDgNAhQOgQIH
QIEDoECB0ChA4BQocLe39///zf9/xSCwSY0SFA6BAgdAg
QOgQIHQKEDgGLv7329vb2+b/1PDqVZ8dZ67uCe13HV1s
c95GRcRo9x7XeI8aZ0SFA6BAgdAgQOgR4GLdw/99d/5x
z2/q61jD6Hp/pWuo2CX3NN3jrlJbuf/T6t76uNVhrfY
sECB0ChM6fW/RHG/chdAgQOgQIHQJO8330Z6fx7DhrfY
tni7XonHPb+rpe8eoYHHGOjDGjX9AU5twNJkLnzwz9aO
M+hA4BQocAoUOA0CHAb68N7P+o1z7zaJ9r72+y1tizHz
M6BAgdAoQOAZddo8/107mMrE03eO3aRt6Lrcee/ZjRIU
DoECB0CBA6BAgdAhY/dQeuw4wOAUKHAKFDgNAhQOg
QIHQIEDoECB0ChA4BQocAoUOA0CFA6BAgdAgQOgQIH
QKEDgFChwAfJcVDI3+cgvMZCv3RF8Pwhb/7WQO/81
kpFY5772u/E789jUuHWHAKFDgNAhQOgQcMmHcWseY
```

```
�PNG
�
���
IHDR����������������Z=����sRGB����
������gAMA������a����
pHYs���t���t��f�x���jIDATx^��[N�F�@��!
�_+[H��Q$�
n������i?��*�������~�������  ��
����B���C��!@���
t��:�������B����{ ������R������!@����
t��:����������������������<:�g�Y
{]�W[����q�=ģ�#ꝁ�!@���
t��:�x��p��]��s�����0����Z�6 }�7x�
�� ������5\�����;�������O[�G���!t��:���
�N�}�g���8k}
�g���s���^�����q���1�_����
&B���h�>����B���C���!�o�
```

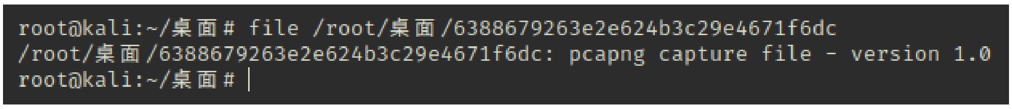
直接用base64转图片得到flag：



```
FLAG{LSB_i5_SO_EASY}
```

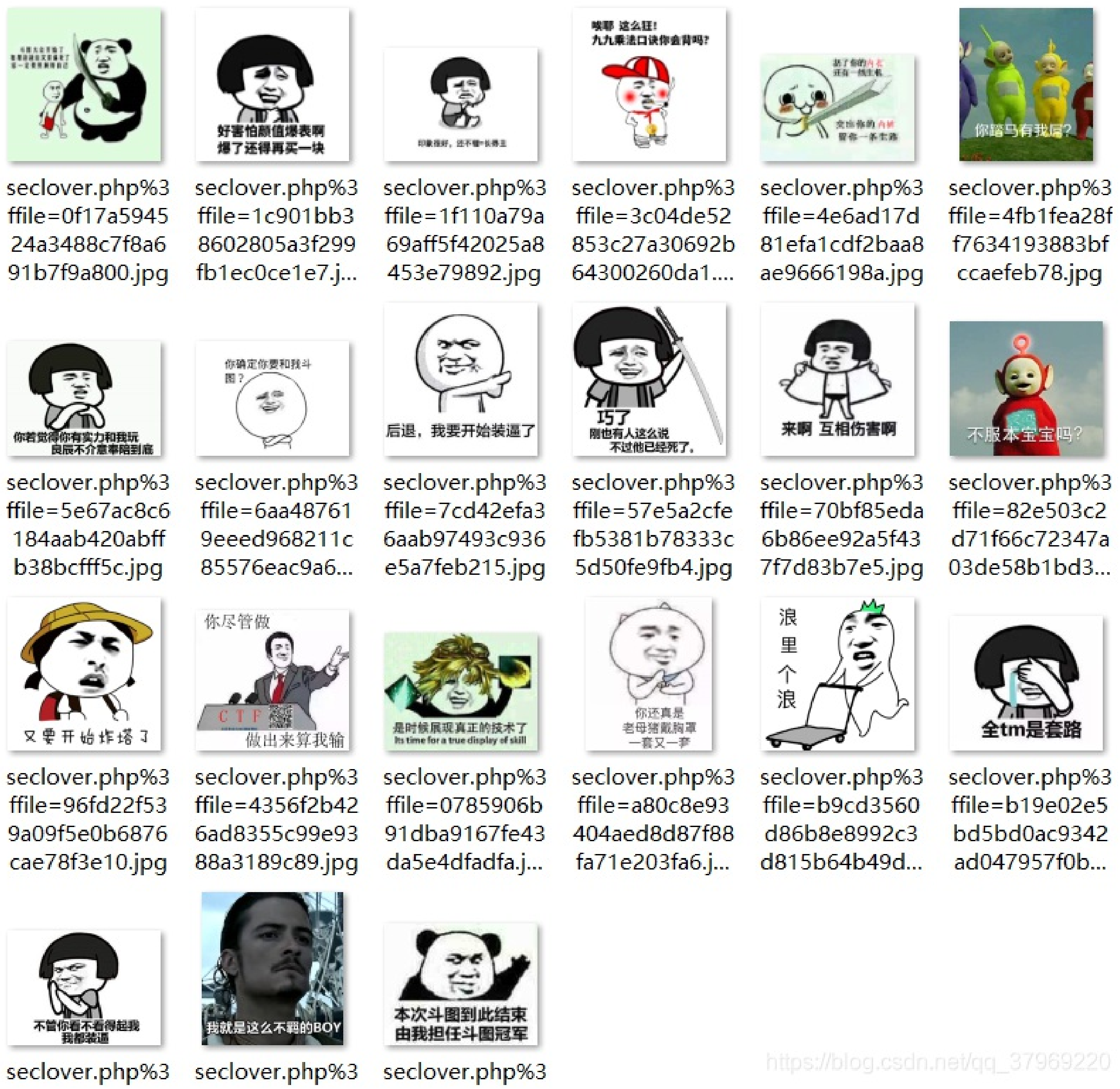
互相伤害！！！

下载解压得到一个位置类型的文件，丢到Kali里用file命令跑一下



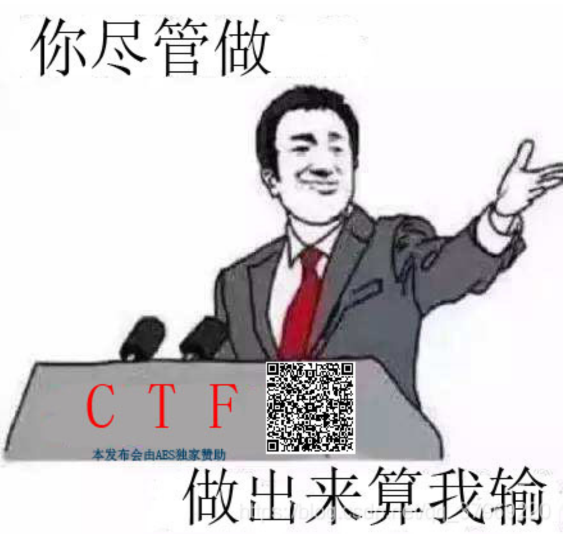
```
root@kali:~/桌面# file /root/桌面/6388679263e2e624b3c29e4671f6dc
/root/桌面/6388679263e2e624b3c29e4671f6dc: pcapng capture file - version 1.0
root@kali:~/桌面# 
```

得知为一个流量包，用Wireshark打开，右键追踪流发现全部是HTTP请求，内容全部是jpg图片，左上角文档里选择导出对象将所有HTTP请求里的东西全部导出来：

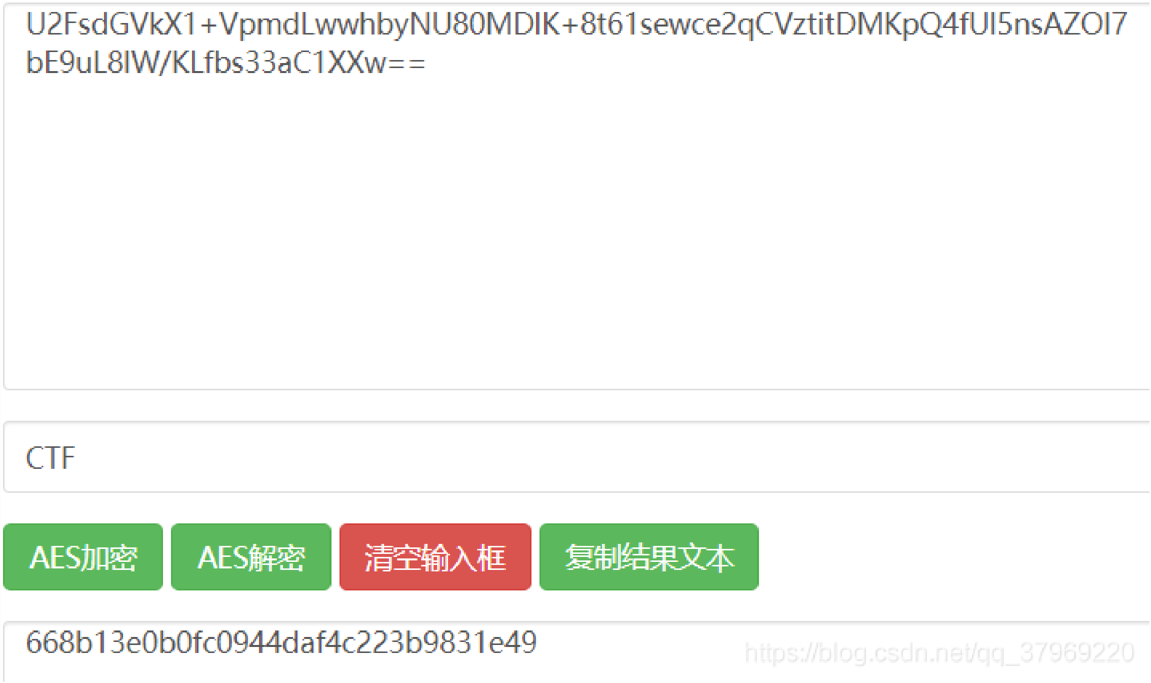


seclover.php%3ffile=0f17a594524a3488c7f8a691b7f9a800.jpg
seclover.php%3ffile=1c901bb38602805a3f299fb1ec0ce1e7.j...
seclover.php%3ffile=1f110a79a69aff5f42025a8453e79892.jpg
seclover.php%3ffile=3c04de52853c27a30692b64300260da1....
seclover.php%3ffile=4e6ad17d81efa1cdf2baa8ae9666198a.jpg
seclover.php%3ffile=4fb1fea28ff7634193883bfccaefeb78.jpg
seclover.php%3ffile=5e67ac8c6184aab420abffb38bcfff5c.jpg
seclover.php%3ffile=6aa487619eeed968211c85576eac9a6...
seclover.php%3ffile=7cd42efa36aab97493c936e5a7feb215.jpg
seclover.php%3ffile=57e5a2cfefb5381b78333c5d50fe9fb4.jpg
seclover.php%3ffile=70bf85eda6b86ee92a5f437f7d83b7e5.jpg
seclover.php%3ffile=82e503c2d71f66c72347a03de58b1bd3...
seclover.php%3ffile=96fd22f539a09f5e0b6876cae78f3e10.jpg
seclover.php%3ffile=4356f2b426ad8355c99e9388a3189c89.jpg
seclover.php%3ffile=0785906b91dba9167fe43da5e4dfadfa.j...
seclover.php%3ffile=a80c8e93404aed8d87f88fa71e203fa6.j...
seclover.php%3ffile=b9cd3560d86b8e8992c3d815b64b49d...
seclover.php%3ffile=b19e02e5bd5bd0ac9342ad047957f0b...
seclover.php%3
seclover.php%3
seclover.php%3

发现这张图里有一个二维码



直接解码解不出来，但根据图片上的信息可知为AES加密，密码为CTF，解码网站：http://www.jsons.cn/aesencrypt

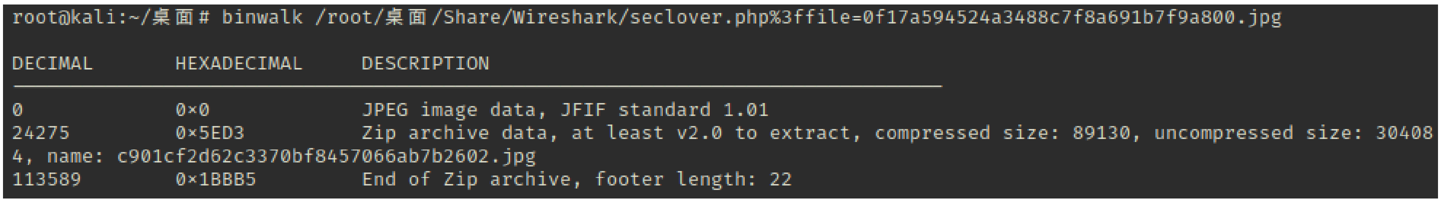


U2FsdGVkX1+VpmdLwwhbyNU80MDIK+8t61sewce2qCVztitDMKpQ4fUI5nsAZOI7
bE9uL8IW/KLfbs33aC1XXw==

CTF

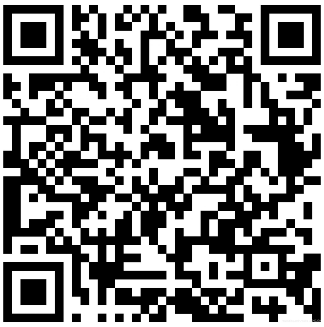
668b13e0b0fc0944daf4c223b9831e49

解码后是一串数字，但这并不是flag，继续用binwalk分析图片：



```
root@kali:~/桌面# binwalk /root/桌面/Share/Wireshark/seclover.php%3ffile=0f17a594524a3488c7f8a691b7f9a800.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
24275         0x5ED3          Zip archive data, at least v2.0 to extract, compressed size: 89130, uncompressed size: 30408
4, name: c901cf2d62c3370bf8457066ab7b2602.jpg
113589        0x1BBB5         End of Zip archive, footer length: 22
```

发现藏有一个zip文件，foremost分离出来解压得到一张清晰的二维码：



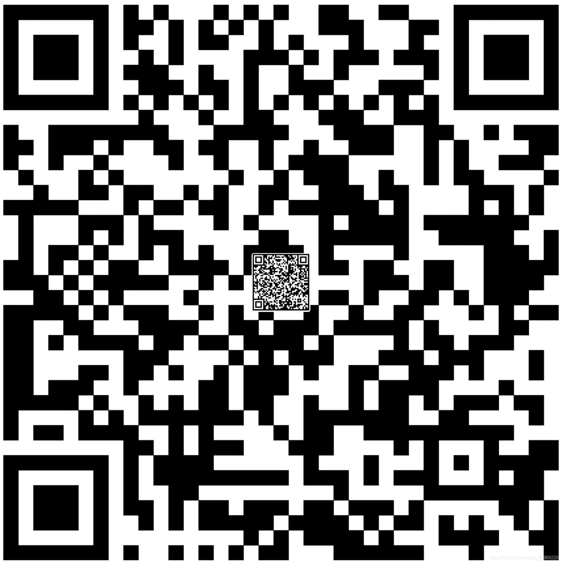
扫码得一句话：

> 扔下内衣真有一线生机？？？？
> 交出内裤才有活路！！！！

那么我们的目标就锁定到了这张图上：

然而这张图foremost分离出的压缩包里的二维码和之前那一张一模一样，最后我们发现这一张图片分离出的压缩包存在密码（真就照应题目名呗）：

这时我们想到之前得到的那一串数字，输入后成功解压，得到一张不一样的二维码：



外面大的二维码扫描结果和前面的一样，里面那个小二维码扫描后得到：

> flag{97d1-0867-2dc1-8926-144c-bc8a-4d4a-3758}

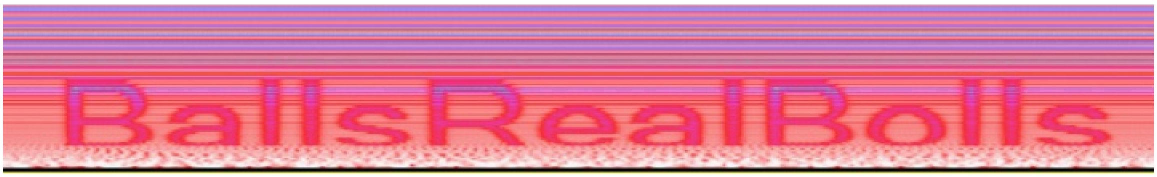
题目里说要提交flag{xxx}内的xxx内容，所以最终的答案就是括号里面的内容（又一个坑）

Miscellaneous-300

下载附件得到一个有密码的压缩包，暴力破解之得密码46783，刚好是被加密的压缩包的名字，解压出来的压缩包还是有密码，那么这题就应该是一个压缩包套娃题，题目里让我们检查73168.zip这个文件，直接写Python脚本解之：

```
import zipfile
name = r"C:\Users\28919\Desktop\f932f55b83fa493ab024390071020088.zip"
while True:
    if name == r"C:\Users\28919\Desktop\zip\73168.zip":
     print("find")
 ts1 = zipfile.ZipFile(name,'r')
 # 获取压缩包里第一个文件名并截取前5个字符
 passwd = ts1.namelist()[0][0:5]
 print(passwd)
 ts1.extractall("C:\\Users\\28919\\Desktop\\zip\\",pwd=passwd.encode())
 name = "C:\\Users\\28919\\Desktop\\zip\\"+ts1.namelist()[0]
```

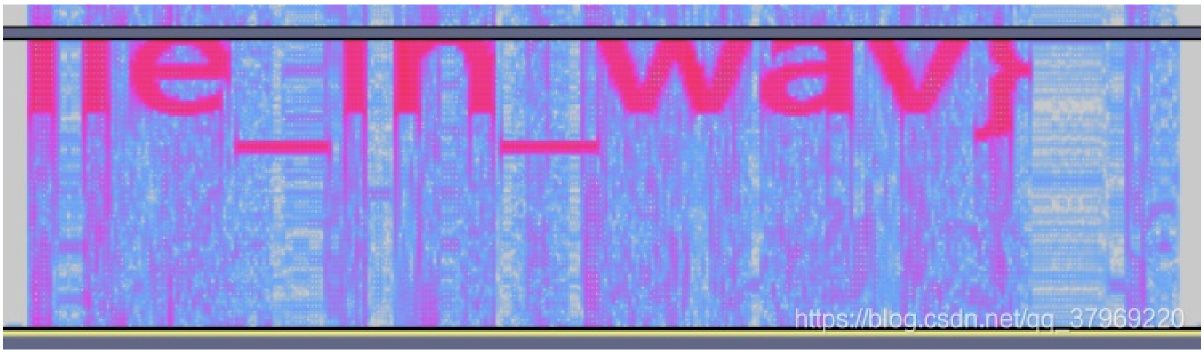
全部解压完有1509个压缩包，然而没有一个是73168，但是我们看到最后解压出的压缩包里是一个wav文件，暴力破解得密码：**b0yzz**，用Audacity打开切换到频谱图看到flag：



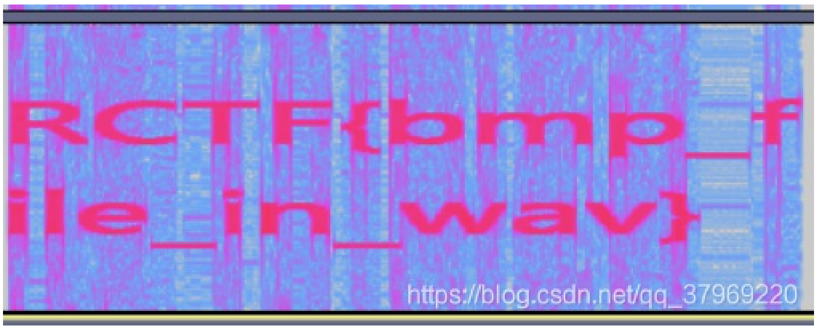
> BallsRealBolls

intoU

下载解压得到一个wav文件，直接播放发现最后有一段杂音，用Audacity打开切换到频谱图看到有一段flag，然而并不完整



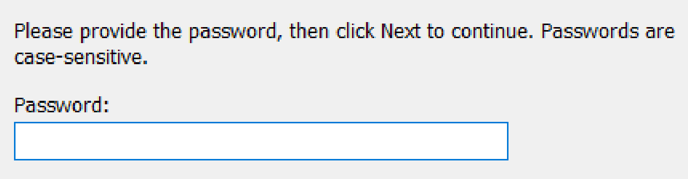
这道题目做法有很多，主要是Audacity的使用，左侧可以在菜单中把采样率改小，或者直接右击左侧频率栏选择缩小，一番操作完成后flag就出来了



> RCTF{bmp_file_in_wav}

Just-No-One

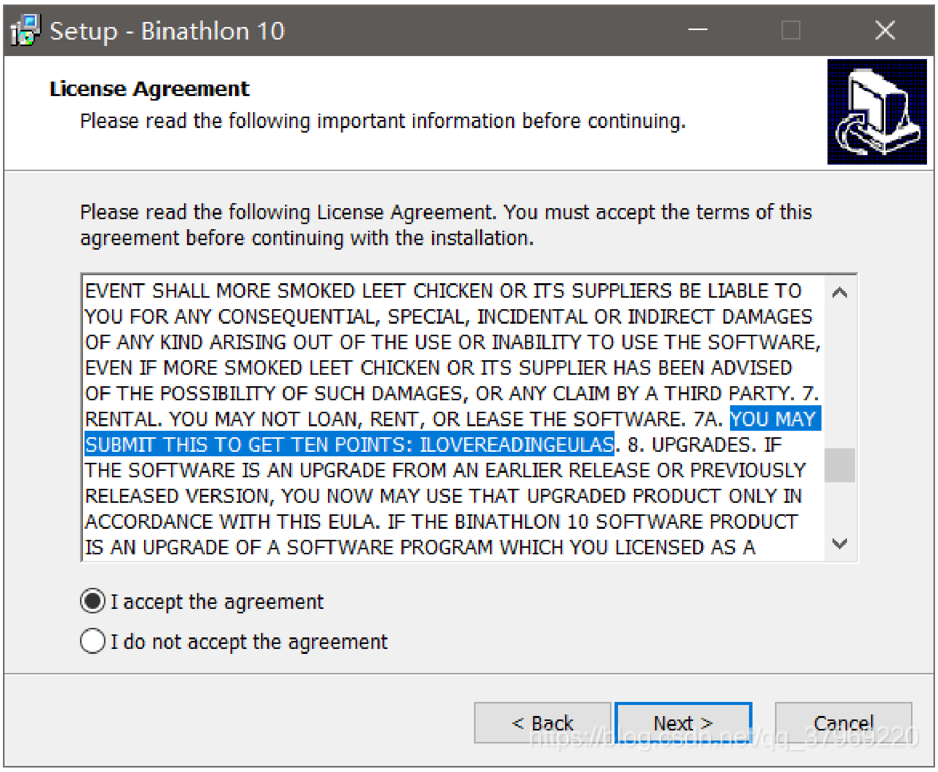
下载附件得一个exe文件，运行，发现是一个安装程序，继续，发现需要输入密码



Please provide the password, then click Next to continue. Passwords are case-sensitive.

Password:

由于这是道MISC的题，应该不是逆向分析，我们回到上一步看这个永远没有人会看的许可协议，里面有一句话：



License Agreement

Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

EVENT SHALL MORE SMOKED LEET CHICKEN OR ITS SUPPLIERS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, SPECIAL, INCIDENTAL OR INDIRECT DAMAGES OF ANY KIND ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE, EVEN IF MORE SMOKED LEET CHICKEN OR ITS SUPPLIER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR ANY CLAIM BY A THIRD PARTY. 7. RENTAL. YOU MAY NOT LOAN, RENT, OR LEASE THE SOFTWARE. 7A. YOU MAY SUBMIT THIS TO GET TEN POINTS: ILOVEREADINGEULAS. 8. UPGRADES. IF THE SOFTWARE IS AN UPGRADE FROM AN EARLIER RELEASE OR PREVIOUSLY RELEASED VERSION, YOU NOW MAY USE THAT UPGRADED PRODUCT ONLY IN ACCORDANCE WITH THIS EULA. IF THE BINATHLON 10 SOFTWARE PRODUCT IS AN UPGRADE OF A SOFTWARE PROGRAM WHICH YOU LICENSED AS A

● I accept the agreement

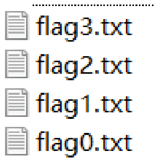
○ I do not accept the agreement

嗯，没错这就是flag

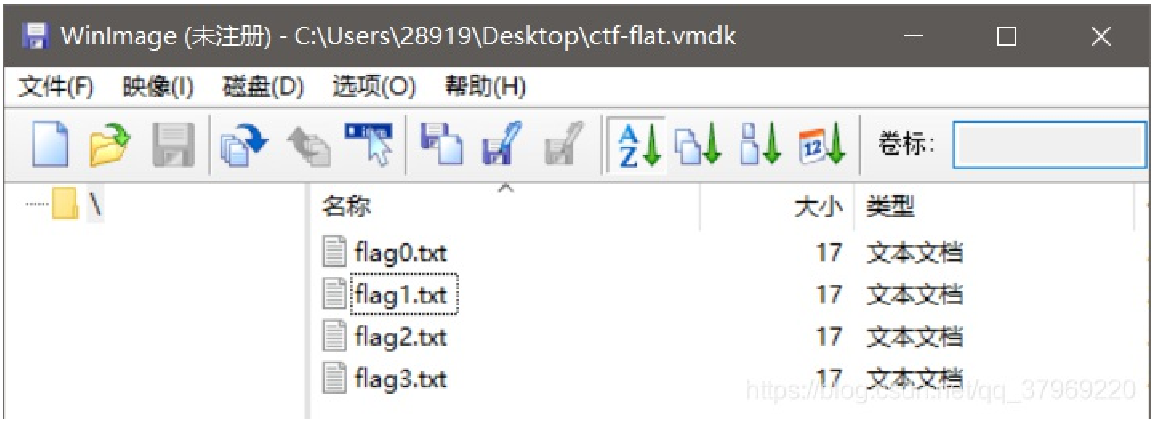
> ILOVEREADINGEULAS

Disk

下载解压得到一个vmdk文件，解压看到里面有4个txt文件（WinRar可能会报错压缩包损坏）

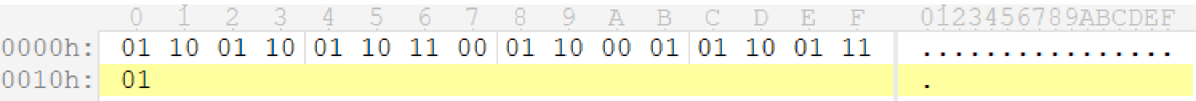

flag3.txt
flag2.txt
flag1.txt
flag0.txt

打开里面只有一句话：**flag is not here.**
用WinImage打开，右键将4个文件提取出来


WinImage (未注册) - C:\Users\28919\Desktop\ctf-flat.vmdk

名称	大小	类型
flag0.txt	17	文本文档
flag1.txt	17	文本文档
flag2.txt	17	文本文档
flag3.txt	17	文本文档

再用010Editor打开提取出来的文件，看到全部都是是二进制格式，将数据整合到一起转为字符串就得到flag

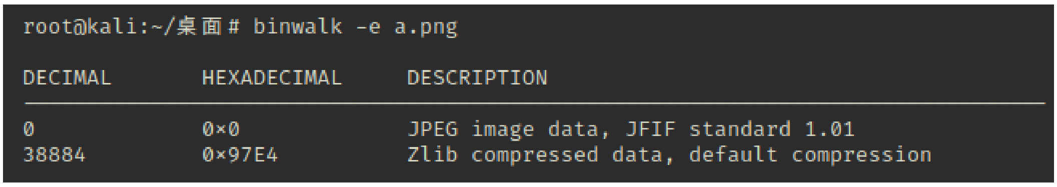

```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
0000h: 01 10 01 10 01 10 11 00 01 10 00 01 01 10 01 11   ................
0010h: 01                                                .
```

```python
# 二进制转换为文本
s = '''01100110011011000110000101100111011110110011010\
001000100010100110101111100110001011011100101111101000\
1000011000101110011011010110111101'''
flag = ''
for i in range(0,len(s),8):
    flag += chr(int(s[i:i+8],2))
print(flag)
```

> flag{4DS_1n_D1sk}

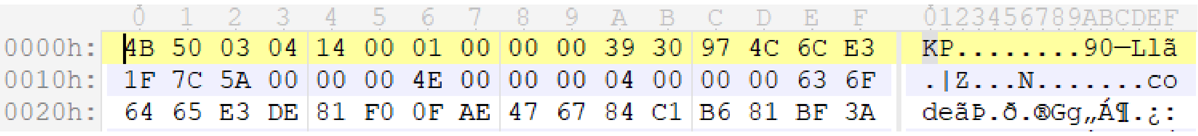
picture2

下载得到一张png图片，但是实际格式是jpg，用binwalk命令跑一下发现里面有一个zlib文件，再用binwalk分离出来（binwalk会自动将zlib文件解压）



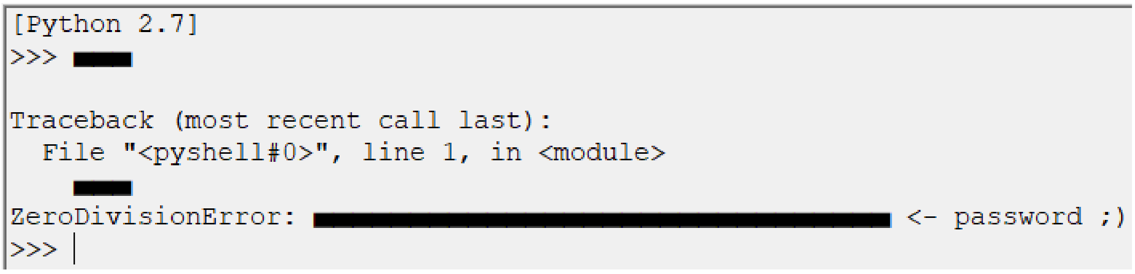
```
root@kali:~/桌面# binwalk -e a.png

DECIMAL         HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0               0x0             JPEG image data, JFIF standard 1.01
38884           0x97E4          Zlib compressed data, default compression
```

得到的文件是一段Base64，解码后根据文件头可以看出是一个zip文件，但是文件头前两个字符颠倒了



```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
0000h: 4B 50 03 04 14 00 01 00 00 00 39 30 97 4C 6C E3   KP........90—Llã
0010h: 1F 7C 5A 00 00 00 4E 00 00 00 04 00 00 00 63 6F   .|Z...N.......co
0020h: 64 65 E3 DE 81 F0 0F AE 47 67 84 C1 B6 81 BF 3A   deãÞ.ð.®Gg„Á¶.¿:
```

改正后解压发现需要密码，压缩包里的注释告诉我们密码是ZeroDivisionError这个报错信息后面的一句话，用Python写个1/0运行得到后面的文本为：**integer division or modulo by zero**



```
[Python 2.7]
>>> [redacted]

Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    [redacted]
ZeroDivisionError: [redacted] <- password ;)
>>> 
```

解压后对里面的这段文本用 uuencode 解码即可得到flag

```
G0TE30TY[,C,X.$%&,C@Y,T5".#5%0C%"-#,Y04)&1C8Q-S,Q.49]
```

```
CISCN{2388AF2893EB85EB1B439ABFF617319F}
```

MulTzor

题目有点小问题，最后应该少了一位，这里贴上正确的原文

```
38708d2a29ff535d9e3f20f85b40df3c3fab465b9a731ce55b54923279e85b4397362be25c54df2020f8465692733ce5535193363dab465
b9a732eee41479a2137ab735f933a3cf8125a91730ee4405f9b730eea4013b61a79ff5d138d3638ef12408a312aff535d8b3a38e7125292
3c2ce54640df3c3fab7f5c8d203ca6515c9b363dab40529b3a36ab515c923e2ce55b509e2730e45c40df3c3fab465b9a7318f35b40df233
6fc57418c732de35347df3b38ef12519a3637ab575d9c3a29e357419a3779fe415a913479ce5c5a983e38ab5f529c3b30e55740d1730de
35b40df2a30ee5e579a3779e65b5f962738f94b13963d2dee5e5f96343ce55156df2431e2515bd37338e75d5d98732ee2465bdf2731ea4
613992136e6125c8b3b3cf912579a302bf242479a3779ca4a5a8c732bea565a907338e556138b3635ee4241963d2dee40138b2138e54
15e96202ae25d5d8c7f79fc5340df3430fd575ddf2731ee125090373ce5535e9a730ce746419e7d79df5a5a8c732eea41139c3c37f85b57
9a213cef125186732eee41479a2137ab61468f213ce65713be3f35e25757df1036e65f5291373cf91277883a3ee34613bb7d79ce5b409a
3d31e445568d732de4125b9e253cab50569a3d79a956569c3a2ae24456dd732de41247973679ca5e5f96363dab445a9c2736f94b1df55
90de35713ba3d30ec5f52df3e38e85a5a91362aab45568d3679ea12559e3e30e74b13903579fb5d418b323be757139c3a29e35741df3e
38e85a5a91362aab455a8b3b79f95d47902179f851419e3e3be757418c7d79cc5d5c9b7336fb57419e2730e555138f2136e857578a213
cf81e138f2136fb5741932a79ee5c5590213aee561fdf2436fe5e57df3b38fd571392323dee1247973679fb5e46983136ea4057df1637e25
55e9e7334ea515b963d3cab475d9d213cea59529d3f3ca5127b90243cfd5741d37334e44147df3c3fab465b9a731eee405e9e3d79e65b
5f962738f94b13993c2be85740d3732aee51419a2779f85741893a3aee41139e3d3dab515a893a35e2535ddf323eee5c5096362aab465
b9e2779fe41569b731ce55b54923279ee5f43933c20ee56138f3c36f9125c8f362bea465a913479fb405c9c363dfe40568c7f79ea5c57df3
a2dab45528c732de357409a7329e45d41df232be451569b262bee41138b3b38ff1252933f36fc5757df2731ee1276913a3ee6531392323
ae35b5d9a2079ff5d139d3679f957459a212aee1f56913430e557568d363dab535d9b732de357139c3a29e357418c732de412519a732b
ee5357d15953df5a56df143cf95f52917329e747549d3c38f9561e9a222ce242439a3779ce5c5a983e38ab50569c3234ee127d9e2930ab
75568d3e38e54b148c7329f95b5d9c3a29ea5e139c2120fb465cd22020f84656927d79c2461388322aab504190383ce5125186732de35
713af3c35e2415bdf143ce557419e3f79d8465299357ef81270962331ee4013bd262bee5346df3a37ab76569c3634e95741df6260b8001
fdf2430ff5a138b3b3cab535a9b7336ed12758d3637e85a1e8c2629fb5e5a9a3779e25c479a3f35e2555691303cab5f528b362be2535fdf3
c3bff535a91363dab5441903e79ea12749a2134ea5c138c2320a51272df3e36e5465bdf313ced5d419a732de3571390262de940569e38
79e45413a83c2be75613a8322bab7b7ad37338ff1252df3036e554568d3637e85713973635ef125d9a322bab65528d2038fc1e138b3b3
cab625c933a2ae31270962331ee4013bd262bee5346df2031ea40569b7330ff4113ba3d30ec5f52d2312bee5358963d3eab46569c3b37
e243469a2079ea5c57df273ce85a5d903f36ec4b13883a2de31247973679cd4056913031ab535d9b731bf95b47962031a512778a2130e
555138b3b3cab75568d3e38e5125a912538f85b5c917336ed1263903f38e5561fdf3036f95713af3c35e2415bdf1030fb5a568d731bfe40
569e2679fb57418c3c37e5575fdf243cf957139a2538e847528b363da71245963279d95d5e9e3d30ea1e138b3c79cd405291303cab455
b9a213cab465b9a2a79ee41479e3135e2415b9a3779ff5a56df031aab70418a3d36ab415a983d38e74113963d2dee5e5f96343ce55156
df202dea465a903d79fc5b4797731ff9575d9c3b79ed5350963f30ff5b568c732afe424390212da512608a303aee4140992635ab515c902
33cf95347963c37ab535e903d3eab465b9a7309e45e568c7f79ff5a56df152bee5c50977f79ea5c57df2731ee12718d3a2de2415bdf322d
ab705f9a273ae35e56867309ea4058df3036e5465a91263cef1246912730e712798a3d3cab030acb6375ab455b9a3d79cd405291303ca
b41468d213ce556568d363dab465cdf2731ee12749a2134ea5c40d15953cd405c92732de35b40df313cec5b5d913a37ec1e138b3b3cab
7041962730f85a13b83c2fee405d923637ff127090373cab535d9b731af2425b9a2179d8515b903c35ab1a74bc751ad81b139e2779c95e
568b3031e7574adf0338f959139d2630e746138a2379ea5c139a2b2dee5c4096253cab514186232dea5c52932a2de251139c3229ea50
5a933a2df21c13b63d30ff5b52933f20a71247973679ef57508d2a29ff5b5c91732eea4113923230e55e4adf3c3fab7e4699272eea54559
a7371cc5741923237ab535a8d733fe440509a7a79ea5c57df3279ed5744df1b3cee4013d7143cf95f52917338f95f4ad67334ee41409e34
3cf81e139e2079ff5a56df182be257548c3e38f95b5d9a7371cc5741923237ab5c52892a70ab575e8f3f36f25757df3e2ce85a13923c2bee
12409a302cf957138f2136e857578a213cf81255902179fe415a913479ce5c5a983e38a51272933237ab66468d3a37ec1e139e731aea5f
518d3a3dec5713aa3d30fd57418c3a2df2125e9e2731ee5f528b3a3ae2535ddf3237ef125f903430e85b52917f79fb405c893a3dee56139
2263ae3125c99732de35713902130ec5b5d9e3f79ff5a5a913830e555138b3b38ff125f9a3779ff5d138b3b3cab56568c3a3ee5125c9973
2de357139c2120fb4652913235f2465a9c3235ab505c92313cab5f529c3b30e55740df2731ea461388362bee125a91202df9475e9a3d2d
ea5e13963d79ee445691272cea5e5f86733bf95752943a37ec1247973679e553459e3f79ce5c5a983e38a5127b90243cfd5741d3732de
35713b42130ee554092322be25c56df3a37ff405c9b263aee56139e3d79ce5c5a983e38ab44568d2030e45c13883a2de31252df3536fe4
04797732be4465c8d733fe4401396272aab671e9d3c38ff411fdf213cf8475f8b3a37ec125a917338ab4241903f36e555569b7329ee405a
903779fc5a5691732de357409a7334ee41409e343cf81250902635ef125d902779e957139b363af94b438b363da51264962731ab465b9
a733aea42478a213cab5d55df213ce757459e3d2dab515a8f3b3cf912589a2a2aab535d9b732de357138a203cab5d55df3e2ce85a1399
322aff5741df060aab7c52892a79e95d5e9d362aa712419a342ce75341d3732bea425a9b732bee5357963d3eab5d55df0674e95d528b7
334ee41409e343cf812419a202ce65757d15953df5a56df3535ea5513962063ab7677bc071ff002579c3638b806069d326dbd040bcf316
9e90101cc3761ea0a02cf656db8570a82
```

用Python把16进制字符转为字符串：

```python
f = open('input.txt','r')
s = f.read()
s = s.decode('hex')
open("output.txt","w").write(s)
```

这道题用的是异或加密，用xortool尝试找出密钥：

```
root@kali:~/桌面# xortool -c 20 /root/桌面/output.txt
The most probable key lengths:
 3:  11.9%
 6:  19.7%
 9:   9.4%
12:  14.5%
15:   7.1%
18:  11.2%
21:   5.3%
24:   8.4%
30:   6.8%
36:   5.7%
Key-length can be 3*n
1 possible key(s) of length 6:
w3\xffSY\x8b
Found 1 plaintexts with 95%+ valid characters
See files filename-key.csv, filename-char_used-perc_valid.csv
```

输出的文件最后可以看到明显的flag，但并不正确，猜测应该是密钥还是存在错误.

> T-e fla" is: DCTF{udcea3q5ba46s80b0bv23d8a}10643 9}Y

密钥共6位，而从文件头开始每6个字符里第一个都是错误的，推测出密钥的第一位错误。再看上面那句话，T-e应该为The，对 ‘-’ 和 ‘w’ 进行异或得原文字符为 ‘Z’，再对 ‘Z’ 和 ‘h’ 异或得密钥为 ‘2’。至此我们就得到了异或加密的密钥：**23\xffSY\x8b**
最后用Python进行异或解密即可得flag：

```
f = open('input.txt','r')
s = f.read()
s = s.decode('hex')
flag = ''
key = '23\xffSY\x8b'
length = len(key)
for i in range(len(s)):
 flag += chr(ord(s[i]) ^ ord(key[i%length]))
print flag
```

> DDCTF{0dcea345ba46680b0b323d8a810643e9}

攻防世界MISC高手区部分题目WriteUp（2）

后续