ctf-misc总结(一)

原创

博闻善行 于 2020-05-10 21:20:47 发布 3328 收藏 29
分类专栏： CTF

CTF 专栏收录该内容
18 篇文章 5 订阅
订阅专栏
MISC处处是细节，玩的就是套路。工欲善其事，必先利其器，先在此附上MISC常用的工具下载地址：

```
https://github.com/ctf-resources/misc
```

常用的文件头总结

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

ZIP 文件头：504B0304　　文件尾：50 4B

TIFF (tif) 文件头：49492A00

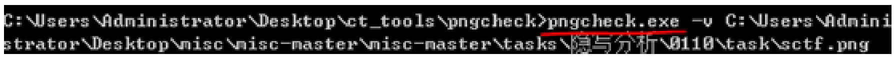
Windows Bitmap (bmp) 文件头：424D

CAD (dwg)　文件头：41433130

Photoshop (psd) 文件头：38425053

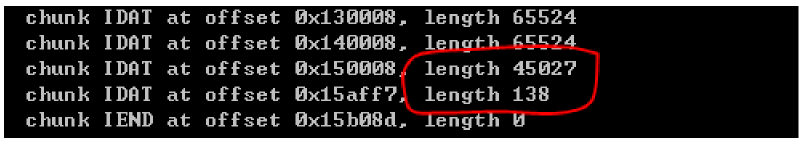
培训题目答题过程总结

1.PNG图片处理：



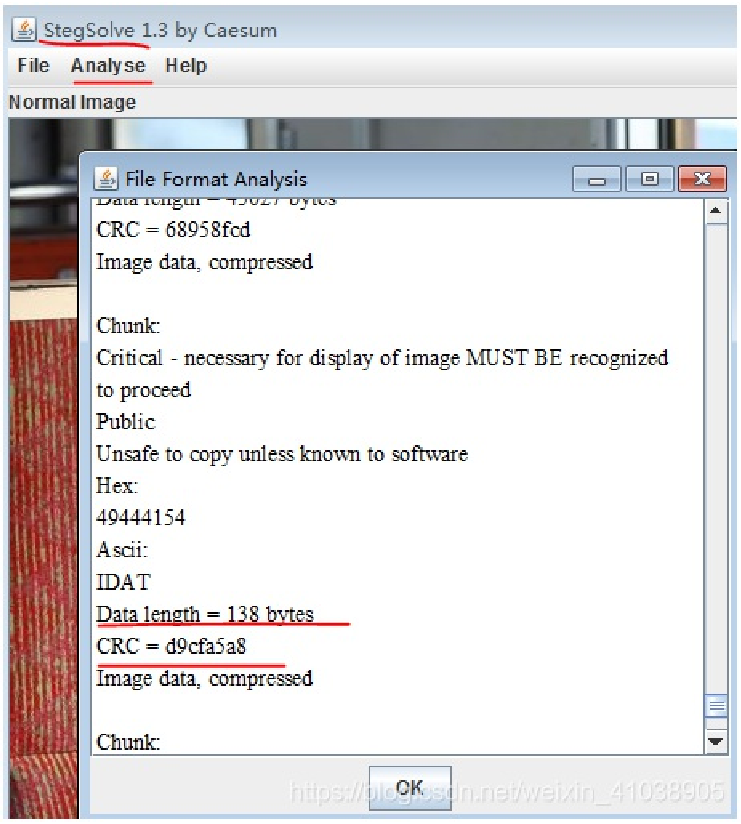
```
C:\Users\Administrator\Desktop\ct_tools\pngcheck>pngcheck.exe -v C:\Users\Admini
strator\Desktop\misc\misc-master\misc-master\tasks\隐写分析\0110\task\sctf.png
```

发现数据块异常



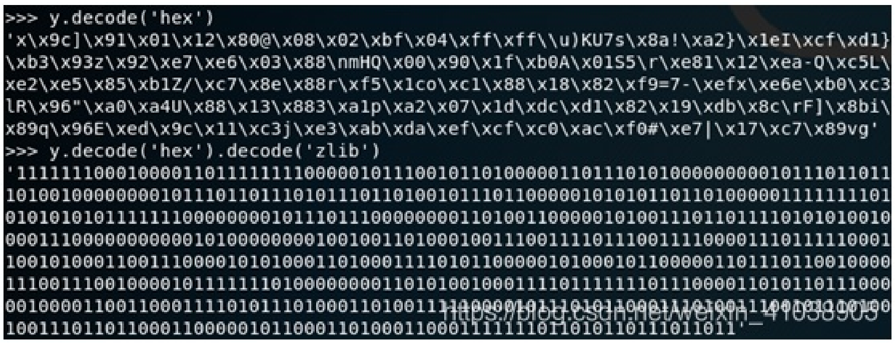
```
  chunk IDAT at offset 0x130008, length 65524
  chunk IDAT at offset 0x140008, length 65524
  chunk IDAT at offset 0x150008, length 45027
  chunk IDAT at offset 0x15aff7, length 138
  chunk IEND at offset 0x15b08d, length 0
```

使用stegsolve分析根据块大小找到块的CRC码，根据校验码在winhex中搜索



```
Data length = 45027 bytes
CRC = 68958fcd
Image data, compressed

Chunk:
Critical - necessary for display of image MUST BE recognized
to proceed
Public
Unsafe to copy unless known to software
Hex:
49444154
Ascii:
IDAT
Data length = 138 bytes
CRC = d9cfa5a8
Image data, compressed

Chunk:
```

复制出异常的数据块
789C5D91011280400802BF04FFFF5C75294B5537738A21A27D1E49CFD17DB3937A92E7E603880A6D485100901FB04
10153350DE83112EA2D51C54CE2E585B15A2FC78E8872F51C6FC1881882F93D372DEF78E665B0C36C529622A0A45
588138833A170A2071DDCD18219DB8C0D465D8B6989719645ED9C11C36AE3ABDAEFCFC0ACF023E77C17C789766
7
在Python中对数据块进行解码



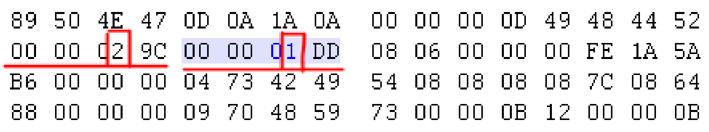
```
>>> y.decode('hex')
'x\x9c]\x91\x01\x12\x80@\x08\x02\xbf\x04\xff\xff\\u)KU7s\x8a!\xa2}\x1eI\xcf\xd1}
\xb3\x93z\x92\xe7\xe6\x03\x88\nmHQ\x00\x90\x1f\xb0A\x01S5\r\xe81\x12\xea-Q\xc5L\
xe2\xe5\x85\xb1Z/\xc7\x8e\x88r\xf5\x1co\xc1\x88\x18\x82\xf9=7-\xefx\xe6e\xb0\xc3
lR\x96"\xa0\xa4U\x88\x13\x883\xa1p\xa2\x07\x1d\xdc\xd1\x82\x19\xdb\x8c\rF]\x8bi\
x89q\x96E\xed\x9c\x11\xc3j\xe3\xab\xda\xef\xcf\xc0\xac\xf0#\xe7|\x17\xc7\x89vg'
>>> y.decode('hex').decode('zlib')
'1111111000100000110111111111000000101110010110100000011011101010000000000101110110 11
1010010000000001011101101110101110110100101110110000001010101101101000000111111101
0101010101111111000000000101110111000000000110100110000001010011101101111010101001 0
0001110000000000001010000000001001001101000100111001111011100111100000111011111000 1
1001010001100111000010101000110100011110101100000010100010110000001101110110010000
1110011100100000101111111101000000000110101001000111101111111101110000011010110111000
0010000011001100011110101110100011010011110000001011101011000111010011100101001000
1001110110110001100000101100011010001100011111111011010110111011011'
```

这个01字符串的长度刚好是625，考虑可能是二维码，编写程序如下：

```
#!/usr/bin/env python
from PIL import Image
MAX = 25
pic = Image.new("RGB",(MAX, MAX))
str = "1111111000100001101111111100000101110010110100000110111010100000000010111011011101001000000001011101101111
0101110110100101110110000010101011011010000011111110101010101011111110000000010111011100000000110100110000010101
0111011011110101010010000111000000000001010000000010010011010001001110011110111001111000011101111100011001010001
1001110000101010001101000111101011000001010001011000001101110110010000111001110010000101111111010000000011010100
1000111101111111101110000110101101110000010000110011000111101011101000110100111110000101110101100011101001110010 1
1101001001110110110001100000101100011010001100011111110110101101110111011011"
i=0
for y in range (0,MAX):
    for x in range (0,MAX):
        if(str[i] == '1'):
            pic.putpixel([x,y],(0, 0, 0))
        else:
            pic.putpixel([x,y],(255,255,255))
        i = i+1

#pic.show()
pic.save("flag.png")
```

最后得到二维码扫描即可

2.binwalk逐张图片提取即可
3.图片高度和宽度隐写



```
89 50 4E 47 0D 0A 1A 0A  00 00 00 0D 49 48 44 52
00 00 02 9C 00 00 01 DD  08 06 00 00 00 FE 1A 5A
B6 00 00 00 04 73 42 49  54 08 08 08 08 7C 08 64
88 00 00 00 09 70 48 59  73 00 00 0B 12 00 00 0B
```

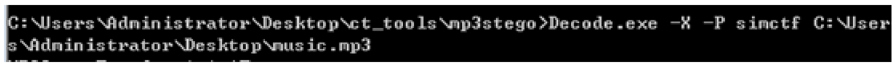
实在找不到的情况下就适当更改图片大小看看
4.打开图片看到像素点，这样的题目必然是对图片像素运算的考察

通过stegsolve逐个RGB通道观察异常，本例中R0通道为全黑，讲R,G,B,Alpha的0通道取出，两两异或后发先G0通道A0通道异或后得打flag
5.
题目信息为mirror，考虑讲图像所有字节反转后保存为图片后进行查看：

```python
data = open('../task/flag.jpg', 'rb').read()
data = data[::-1]
f = open('flag.png', 'wb')
f.write(data)
f.close()
```

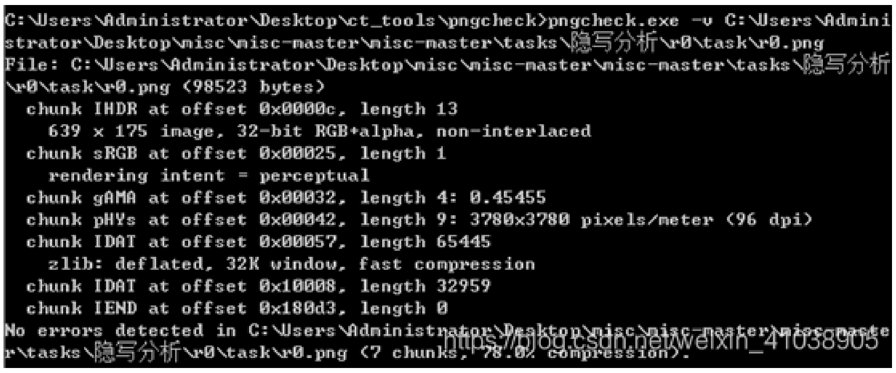
6.

发现图片名是music.jpg，直接binwalk提取以后得到两个文件，在使用mp3stego工具解码后得到flag



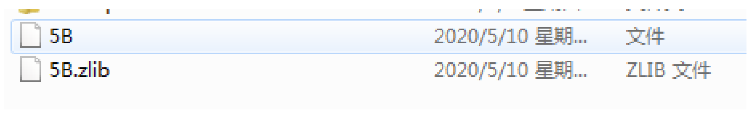
```
C:\Users\Administrator\Desktop\ct_tools\mp3stego>Decode.exe -X -P simctf C:\User
s\Administrator\Desktop\music.mp3
```

7.



```
C:\Users\Administrator\Desktop\ct_tools\pngcheck>pngcheck.exe -v C:\Users\Admini
strator\Desktop\misc\misc-master\misc-master\tasks\隐写分析\r0\task\r0.png
File: C:\Users\Administrator\Desktop\misc\misc-master\misc-master\tasks\隐写分析
\r0\task\r0.png (98523 bytes)
  chunk IHDR at offset 0x0000c, length 13
    639 x 175 image, 32-bit RGB+alpha, non-interlaced
  chunk sRGB at offset 0x00025, length 1
    rendering intent = perceptual
  chunk gAMA at offset 0x00032, length 4: 0.45455
  chunk pHYs at offset 0x00042, length 9: 3780x3780 pixels/meter (96 dpi)
  chunk IDAT at offset 0x00057, length 65445
    zlib: deflated, 32K window, fast compression
  chunk IDAT at offset 0x10008, length 32959
  chunk IEND at offset 0x180d3, length 0
No errors detected in C:\Users\Administrator\Desktop\misc\misc-master\misc-maste
r\tasks\隐写分析\r0\task\r0.png (7 chunks, 78.0% compression).
```

png图片经过pngcheck后无块异常
继续通过binwalk提取以后得到如下两个文件：

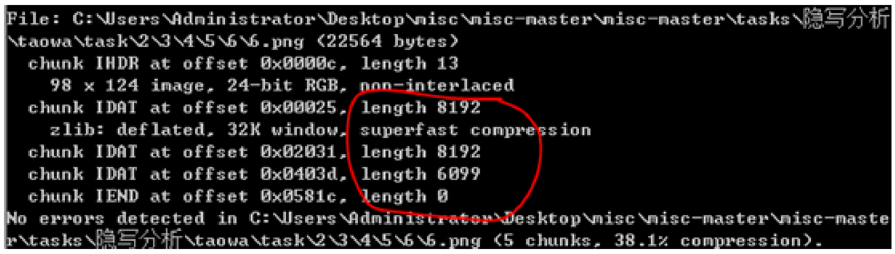


5B	2020/5/10 星期...	文件
5B.zlib	2020/5/10 星期...	ZLIB 文件

其实都没有什么意义，直接stepsolve打开后查看逐个通道，在通道处发现flag

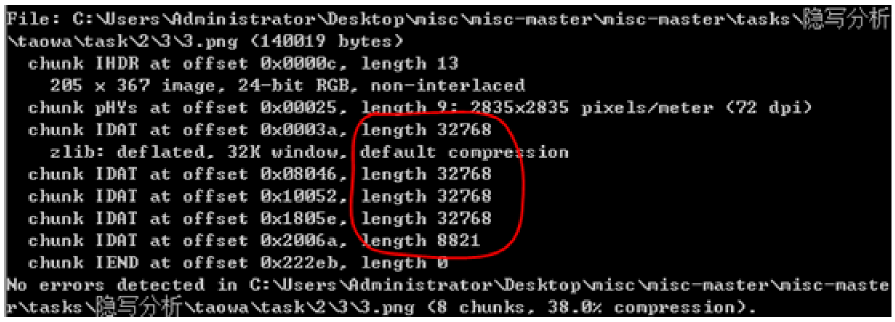
8.wav文件直接Audacity打开，观察波形图和频谱图是否有flag信息

9.通过pngcheck查看块数据发现图片6的IDAT数据块不满



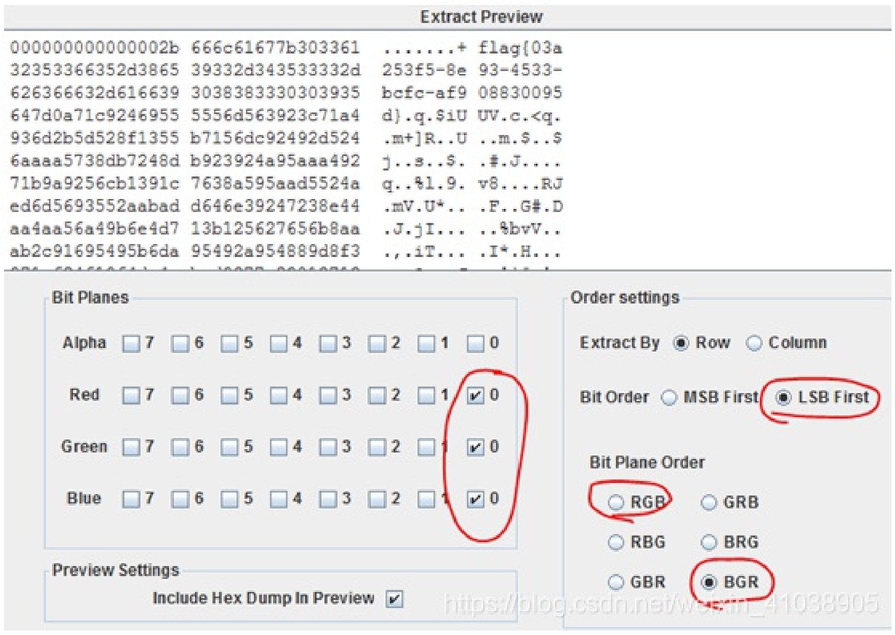
```
File: C:\Users\Administrator\Desktop\misc\misc-master\misc-master\tasks\隐写分析
\taowa\task\2\3\4\5\6\6.png (22564 bytes)
  chunk IHDR at offset 0x0000c, length 13
    98 x 124 image, 24-bit RGB, non-interlaced
  chunk IDAT at offset 0x00025, length 8192
    zlib: deflated, 32K window, superfast compression
  chunk IDAT at offset 0x02031, length 8192
  chunk IDAT at offset 0x0403d, length 6099
  chunk IEND at offset 0x0581c, length 0
No errors detected in C:\Users\Administrator\Desktop\misc\misc-master\misc-maste
r\tasks\隐写分析\taowa\task\2\3\4\5\6\6.png (5 chunks, 38.1% compression).
```

其他图片的IDAT数据块大小为32768，上面的8192明显小与这个数据。考虑是隐写



```
File: C:\Users\Administrator\Desktop\misc\misc-master\misc-master\tasks\隐写分析
\taowa\task\2\3\3.png (140019 bytes)
  chunk IHDR at offset 0x0000c, length 13
    205 x 367 image, 24-bit RGB, non-interlaced
  chunk pHYs at offset 0x00025, length 9: 2835x2835 pixels/meter (72 dpi)
  chunk IDAT at offset 0x0003a, length 32768
    zlib: deflated, 32K window, default compression
  chunk IDAT at offset 0x08046, length 32768
  chunk IDAT at offset 0x10052, length 32768
  chunk IDAT at offset 0x1805e, length 32768
  chunk IDAT at offset 0x2006a, length 8821
  chunk IEND at offset 0x222eb, length 0
No errors detected in C:\Users\Administrator\Desktop\misc\misc-master\misc-maste
r\tasks\隐写分析\taowa\task\2\3\3.png (8 chunks, 38.0% compression).
```

考虑使用stegsolve进行数据提取，常用的配置说明如图所示：



Extract Preview

```
000000000000002b 666c61677b303361  .......+ flag{03a
32353366352d3865 39332d343533332d  253f5-8e 93-4533-
626366632d616639 3038383330303935  bcfc-af9 08830095
647d0a71c9246955 5556d563923c71a4  d}.q.$iU UV.c.<q.
936d2b5d528f1355 b7156dc92492d524  .m+]R..U ..m.$..$
6aaaa5738db7248d b923924a95aaa492  j..s..$. .#.J....
71b9a9256cb1391c 7638a595aad5524a  q..%l.9. v8....RJ
ed6d5693552aabad d646e39247238e44  .mV.U*.. .F..G#.D
aa4aa56a49b6e4d7 13b125627656b8aa  .J.jI... ..%bvV..
ab2c91695495b6da 95492a954889d8f3  .,.iT... .I*.H...
```

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

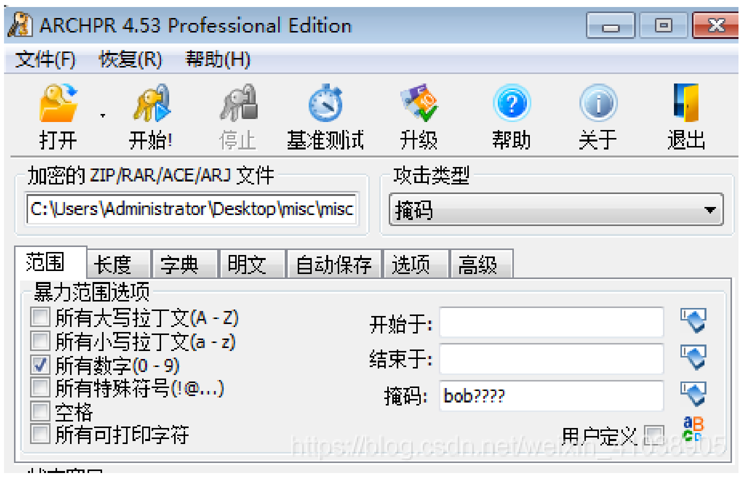
RGB GRB RBG BRG GBR [✓ BGR]

9.缺少图片头，补上图片头即可
GIF文件目前有两种文件头GIF89和GIF79

```python
f = open('../task/xx.gif', 'rb').read()
new_f = open('flag.gif', 'wb')
new_f.write('GIF8'+f)
new_f.close()
```

10.压缩包爆破

掩码爆破,bob是掩码吗，四个？？？？表示后面四位密码不知道是啥



加密的 ZIP/RAR/ACE/ARJ 文件: C:\Users\Administrator\Desktop\misc\misc

攻击类型: 掩码

暴力范围选项
- [ ] 所有大写拉丁文(A - Z)
- [ ] 所有小写拉丁文(a - z)
- [✓] 所有数字(0 - 9)
- [ ] 所有特殊符号(!@...)
- [ ] 空格
- [ ] 所有可打印字符

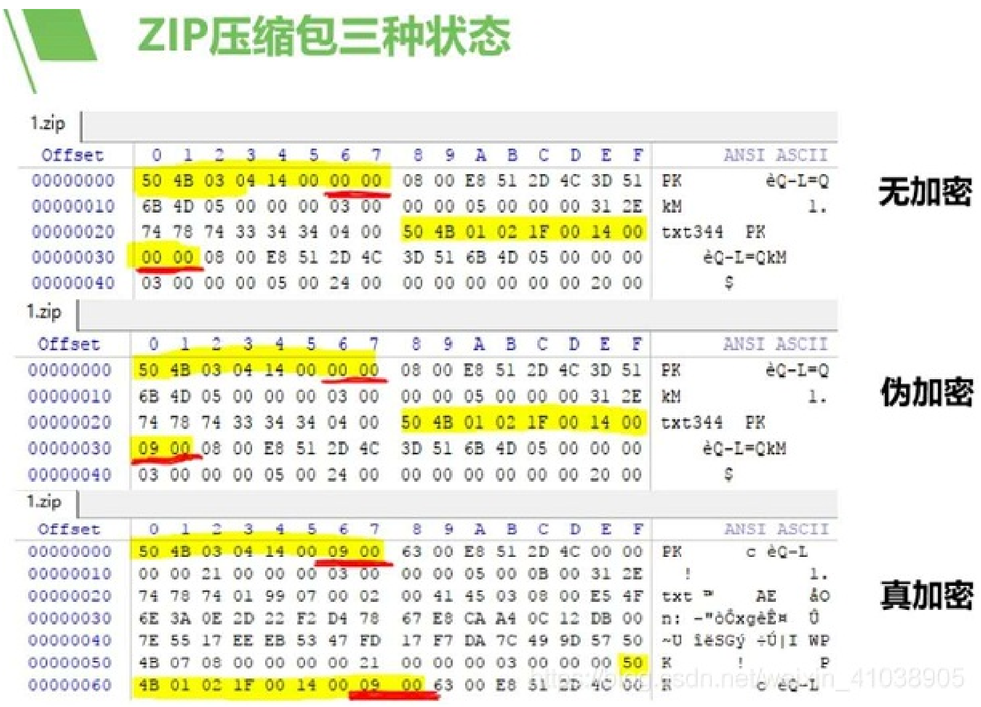
开始于:

结束于:

掩码: bob????

用户定义 [ ]

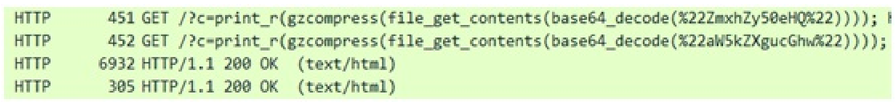
11.伪加密



```
1.zip
Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F   ANSI ASCII
00000000 50 4B 03 04 14 00 00 00  08 00 E8 51 2D 4C 3D 51   PK        èQ-L=Q
00000010 6B 4D 05 00 00 00 03 00  00 00 05 00 00 00 31 2E   kM             1.
00000020 74 78 74 33 34 34 04 00  50 4B 01 02 1F 00 14 00   txt344  PK
00000030 00 00 08 00 E8 51 2D 4C  3D 51 6B 4D 05 00 00 00       èQ-L=QkM
00000040 03 00 00 00 05 00 24 00  00 00 00 00 00 00 20 00         $
1.zip
Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F   ANSI ASCII
00000000 50 4B 03 04 14 00 00 00  08 00 E8 51 2D 4C 3D 51   PK        èQ-L=Q
00000010 6B 4D 05 00 00 00 03 00  00 00 05 00 00 00 31 2E   kM             1.
00000020 74 78 74 33 34 34 04 00  50 4B 01 02 1F 00 14 00   txt344  PK
00000030 09 00 08 00 E8 51 2D 4C  3D 51 6B 4D 05 00 00 00       èQ-L=QkM
00000040 03 00 00 00 05 00 24 00  00 00 00 00 00 00 20 00         $
1.zip
Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F   ANSI ASCII
00000000 50 4B 03 04 14 00 09 00  63 00 E8 51 2D 4C 00 00   PK      c èQ-L
00000010 00 00 21 00 00 00 03 00  00 00 05 00 0B 00 31 2E     !            1.
00000020 74 78 74 01 99 07 00 02  00 41 45 03 08 00 E5 4F   txt ™    AE    åO
00000030 6E 3A 0E 2D 22 F2 D4 78  67 E8 CA A4 0C 12 DB 00   n: -"òÔxgèÊ¤   Û
00000040 7E 55 17 EE EB 53 47 FD  17 F7 DA 7C 49 9D 57 50   ~U îëSGý ÷Ú|I WP
00000050 4B 07 08 00 00 00 00 21  00 00 00 03 00 00 00 50   K      !        P
00000060 4B 01 02 1F 00 14 00 09  00 63 00 E8 51 2D 4C 00   K          c èQ-L
```

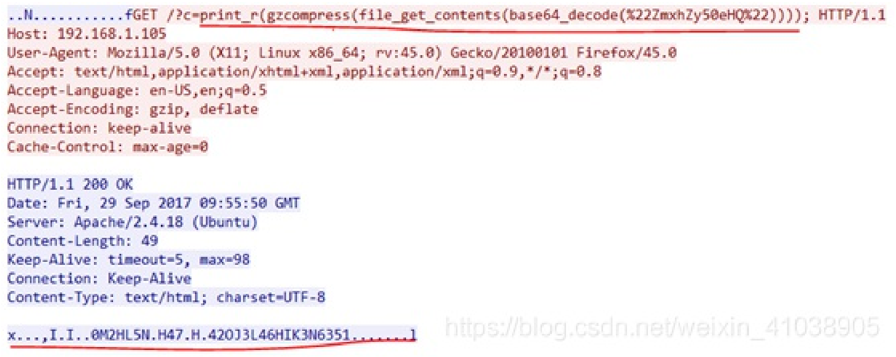
12.wireshark流量分析

对于http流量较多的pcap包，可直接尝试搜索字符串flag或php



```
HTTP       451 GET /?c=print_r(gzcompress(file_get_contents(base64_decode(%22ZmxhZy50eHQ%22)))); H
HTTP       452 GET /?c=print_r(gzcompress(file_get_contents(base64_decode(%22aW5kZXgucGhw%22))));
HTTP      6932 HTTP/1.1 200 OK  (text/html)
HTTP       305 HTTP/1.1 200 OK  (text/html)
```

追踪Http流发现如下关键信息



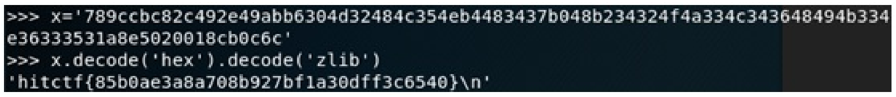
```
..N...........fGET /?c=print_r(gzcompress(file_get_contents(base64_decode(%22ZmxhZy50eHQ%22)))); HTTP/1.1
Host: 192.168.1.105
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Cache-Control: max-age=0

HTTP/1.1 200 OK
Date: Fri, 29 Sep 2017 09:55:50 GMT
Server: Apache/2.4.18 (Ubuntu)
Content-Length: 49
Keep-Alive: timeout=5, max=98
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

x...,I.I..0M2HL5N.H47.H.42OJ3L46HIK3N6351.......l
```

对上面的base64字符串拼接解密

```
�f��FW��-�
�flag.txt
```

```
22aW5kZXgucGhw22ZmxhZy50eHQ=
```

说明http请求回复的内容是flag.txt的内容，我们将原始数据复制出来进行下图解码操作就可以获取flag了



```
>>> x='789ccbc82c492e49abb6304d32484c354eb4483437b048b234324f4a334c343648494b334
e36333531a8e5020018cb0c6c'
>>> x.decode('hex').decode('zlib')
'hitctf{85b0ae3a8a708b927bf1a30dff3c6540}\n'
```

备注：

从通信方式的角度看，后门可以分为http/https型、irc型、dns型、icmp型等，对存在这些协议的流量包进行分析，最后在icmp协议中发现每个包最后的字符可以拼接成flag。