# IceCTF 2016

原创

浅零半泣 于 2017-09-21 23:15:21 发布 2181 收藏 4

分类专栏： CTF

本文链接：https://blog.csdn.net/sinat_34200786/article/details/78058112

版权

CTF 专栏收录该内容

19 篇文章 0 订阅

订阅专栏

## WriteUp

1. Corrupt Transmission
2. Rotated!
3. Blue Monday
4. All your Base are belong to us
5. Thor’s a hacker now
6. Scavenger Hunt
7. R.I.P Transmission

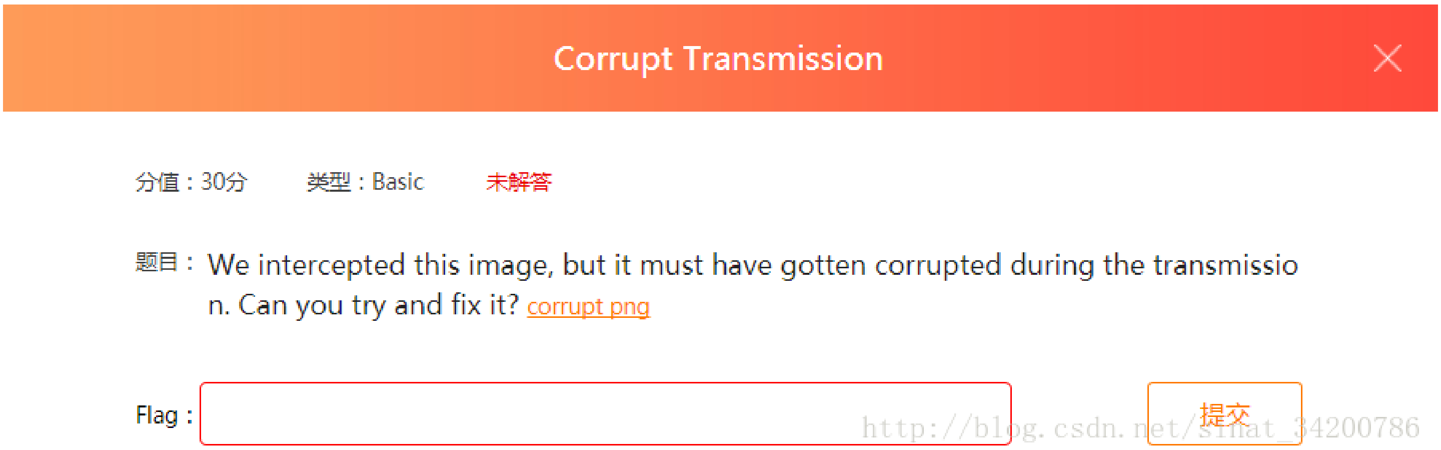
## 涨姿势

1. Audio Problems
2. Intercepted Conversations Pt.1
3. Intercepted Conversations Pt.2

---

## Corrupt Transmission

原题



Corrupt Transmission

分值：30分　　类型：Basic　　未解答

题目：We intercepted this image, but it must have gotten corrupted during the transmission. Can you try and fix it? corrupt png

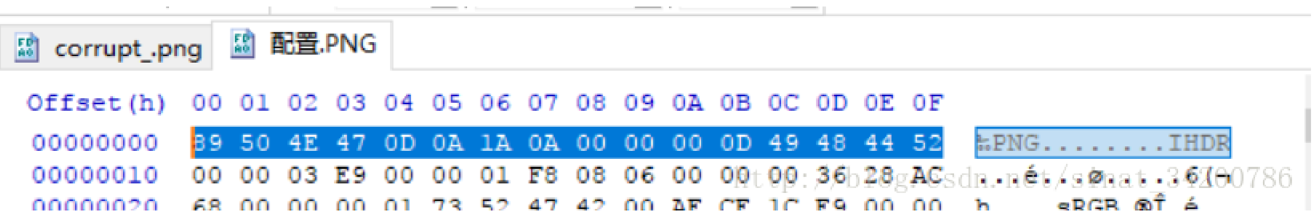
Flag：

解题思路

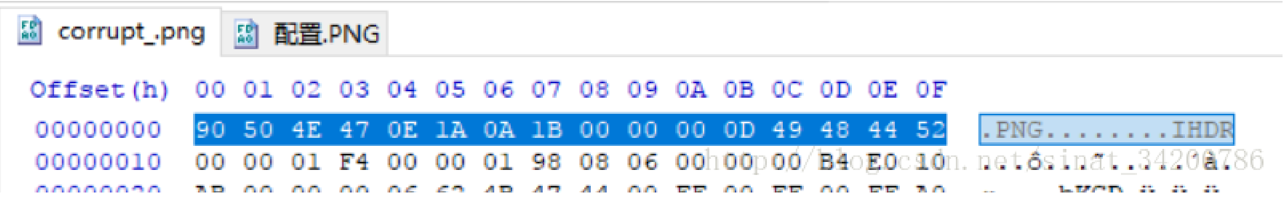
```
PNG图片格式
```

WriteUp

```
已知PNG是损坏的，拿张正常的PNG对比，一下就可以发现题目的PNG前几个字节不对
```

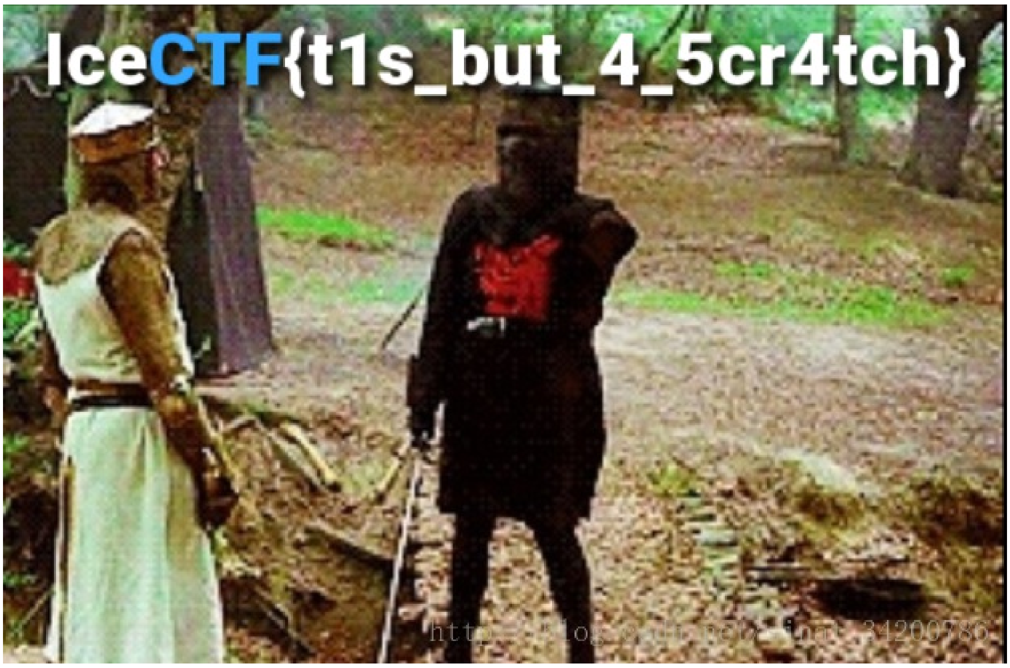


```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52  ‰PNG........IHDR
00000010  00 00 03 E9 00 00 01 F8 08 06 00 00 00 36 28 AC  ...é...ø.....6(¬
00000020  68 00 00 00 01 73 52 47 42 00 AE CE 1C E9 00 00  h....sRGB.®Î.é..
```



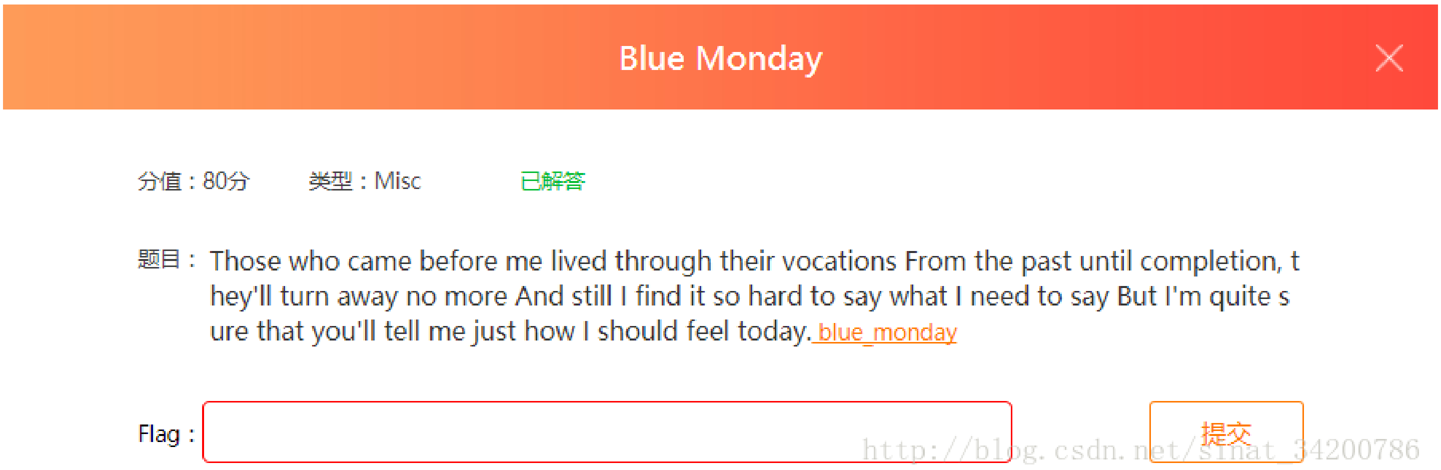
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  90 50 4E 47 0E 1A 0A 1B 00 00 00 0D 49 48 44 52  .PNG........IHDR
00000010  00 00 01 F4 00 00 01 98 08 06 00 00 00 B4 E0 10  ...ô.........´à.
```

```
将正确的字节替换错误的字节保存即可
```



## Blue Monday

原题


Blue Monday

分值：80分 类型：Misc 已解答

题目：Those who came before me lived through their vocations From the past until completion, they'll turn away no more And still I find it so hard to say what I need to say But I'm quite sure that you'll tell me just how I should feel today. blue_monday

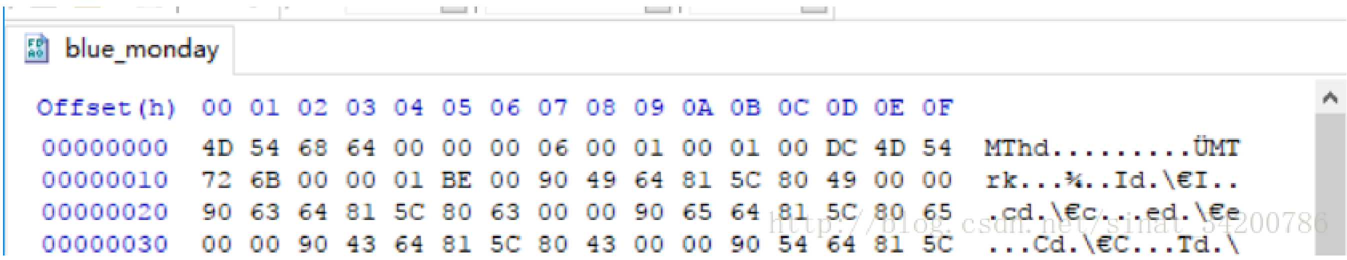
Flag：

解题思路

```
观察法
```

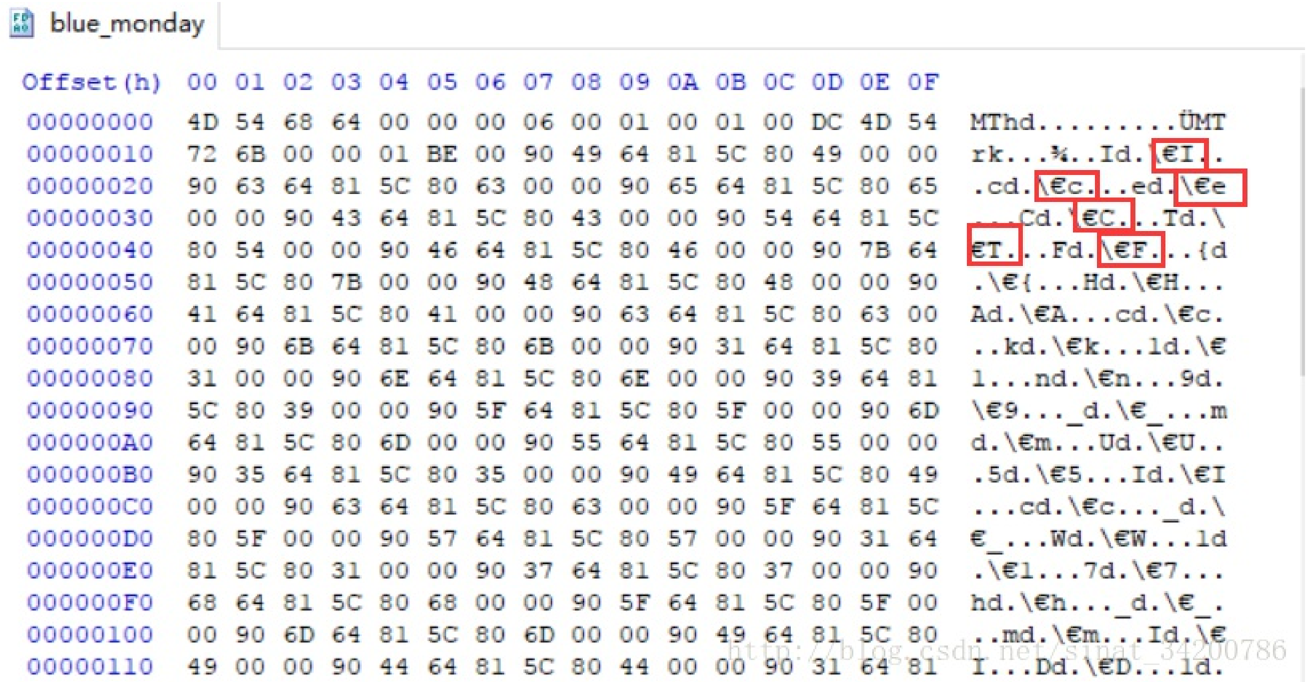
WriteUp

```
不知道是什么文件，用HxD看看再说
```



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  4D 54 68 64 00 00 00 06 00 01 00 01 00 DC 4D 54  MThd.........ÜMT
00000010  72 6B 00 00 01 BE 00 90 49 64 81 5C 80 49 00 00  rk...¾..Id.\€I..
00000020  90 63 64 81 5C 80 63 00 00 90 65 64 81 5C 80 65  .cd.\€c...ed.\€e
00000030  00 00 90 43 64 81 5C 80 43 00 00 90 54 64 81 5C  ...Cd.\€C...Td.\
```

```
可以知道这是MIDI文件，播放一遍后没发现异常，准备用Audacity看看，这时发现了关键信息
```

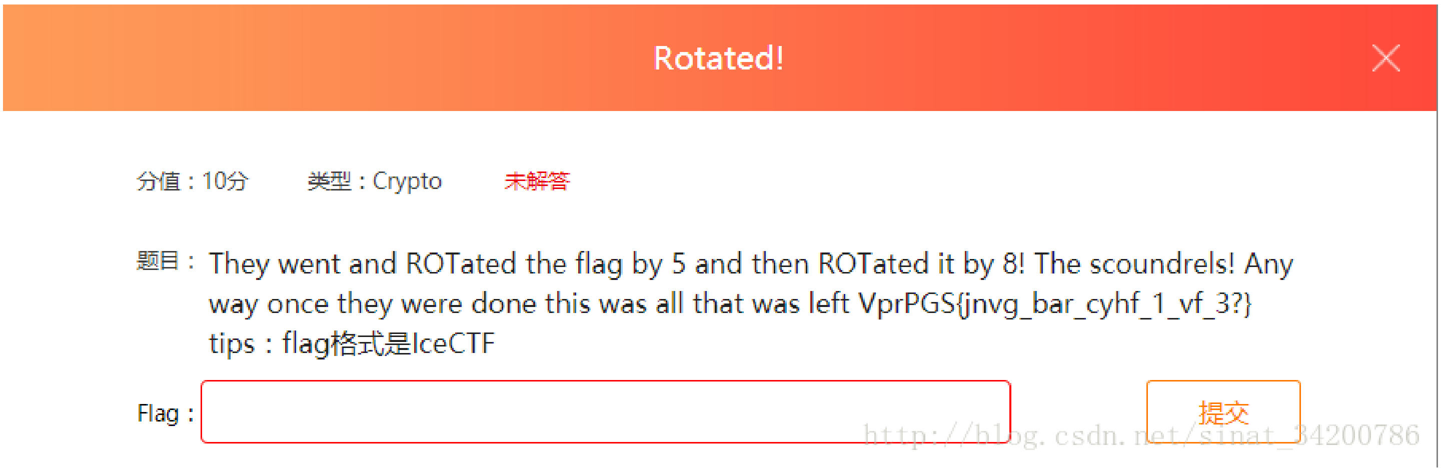


```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  4D 54 68 64 00 00 00 06 00 01 00 01 00 DC 4D 54  MThd.........ÜMT
00000010  72 6B 00 00 01 BE 00 90 49 64 81 5C 80 49 00 00  rk...¾..Id.\€I..
00000020  90 63 64 81 5C 80 63 00 00 90 65 64 81 5C 80 65  .cd.\€c...ed.\€e
00000030  00 00 90 43 64 81 5C 80 43 00 00 90 54 64 81 5C  ...Cd.\€C...Td.\
00000040  80 54 00 00 90 46 64 81 5C 80 46 00 00 90 7B 64  €T...Fd.\€F...{d
00000050  81 5C 80 7B 00 00 90 48 64 81 5C 80 48 00 00 90  .\€{...Hd.\€H...
00000060  41 64 81 5C 80 41 00 00 90 63 64 81 5C 80 63 00  Ad.\€A...cd.\€c.
00000070  00 90 6B 64 81 5C 80 6B 00 00 90 31 64 81 5C 80  ..kd.\€k...1d.\€
00000080  31 00 00 90 6E 64 81 5C 80 6E 00 00 90 39 64 81  1...nd.\€n...9d.
00000090  5C 80 39 00 00 90 5F 64 81 5C 80 5F 00 00 90 6D  \€9..._d.\€_...m
000000A0  64 81 5C 80 6D 00 00 90 55 64 81 5C 80 55 00 00  d.\€m...Ud.\€U..
000000B0  90 35 64 81 5C 80 35 00 00 90 49 64 81 5C 80 49  .5d.\€5...Id.\€I
000000C0  00 00 90 63 64 81 5C 80 63 00 00 90 5F 64 81 5C  ...cd.\€c..._d.\
000000D0  80 5F 00 00 90 57 64 81 5C 80 57 00 00 90 31 64  €_...Wd.\€W...1d
000000E0  81 5C 80 31 00 00 90 37 64 81 5C 80 37 00 00 90  .\€1...7d.\€7...
000000F0  68 64 81 5C 80 68 00 00 90 5F 64 81 5C 80 5F 00  hd.\€h..._d.\€_.
00000100  00 90 6D 64 81 5C 80 6D 00 00 90 49 64 81 5C 80  ..md.\€m...Id.\€
00000110  49 00 00 90 44 64 81 5C 80 44 00 00 90 31 64 81  I...Dd.\€D...1d.
```

```
在字节数据中发现了IceCTF，这不就是flag的开头吗？接下来还有
'{'符号和'}'符号，可以肯定flag就在其中，那么只要一个个字符抄下来即可
```

## Rotated!

原题



Rotated!

分值：10分　　类型：Crypto　　未解答

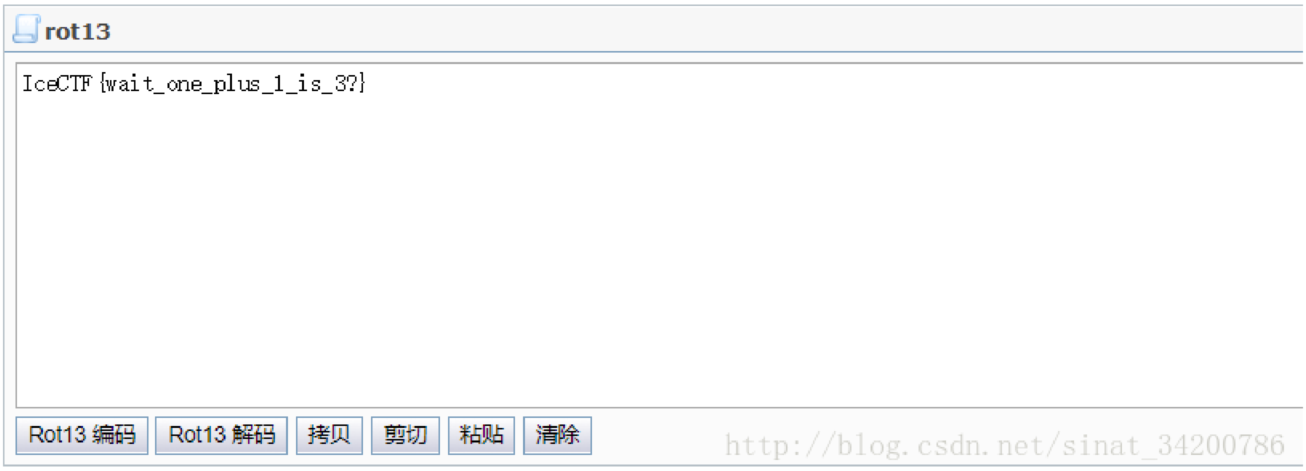
题目：They went and ROTated the flag by 5 and then ROTated it by 8! The scoundrels! Any way once they were done this was all that was left VprPGS{jnvg_bar_cyhf_1_vf_3?}
tips：flag格式是IceCTF

Flag：

解题思路

```
ROT13
```

WriteUp

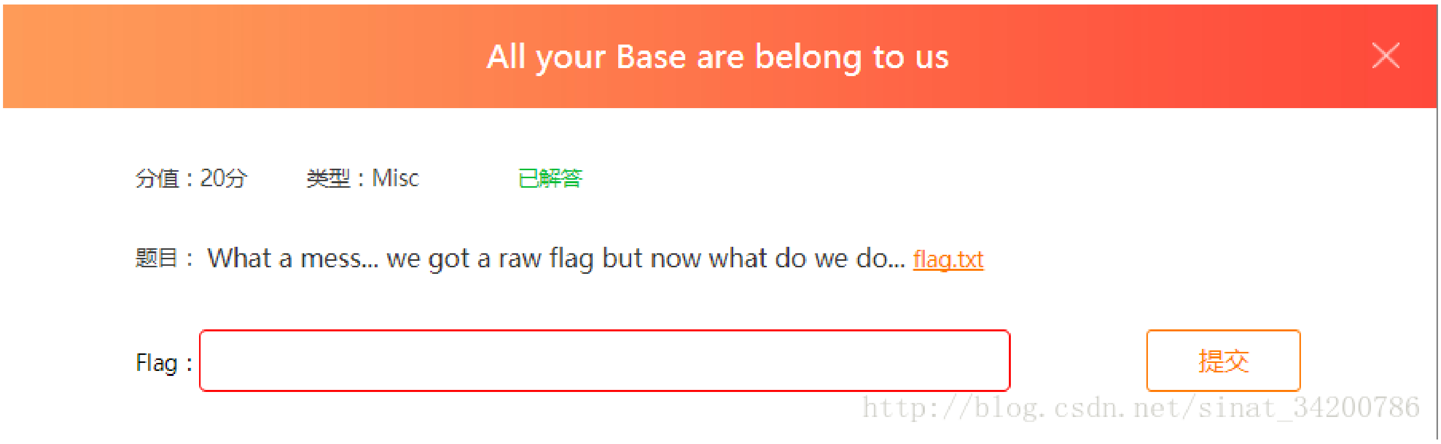


rot13

IceCTF{wait_one_plus_1_is_3?}

## All your Base are belong to us

原题



All your Base are belong to us

分值：20分　类型：Misc　已解答

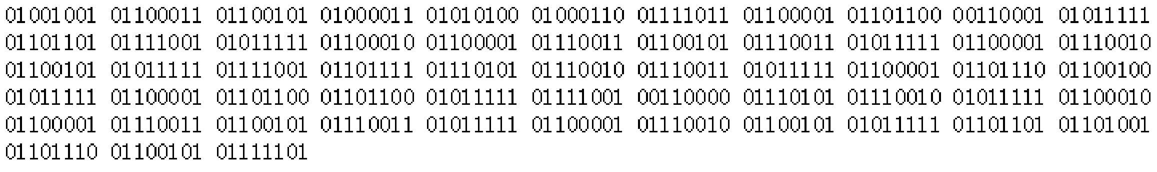
题目： What a mess... we got a raw flag but now what do we do... flag.txt

Flag :

解题思路

```
二进制转ASCII
```

WriteUp



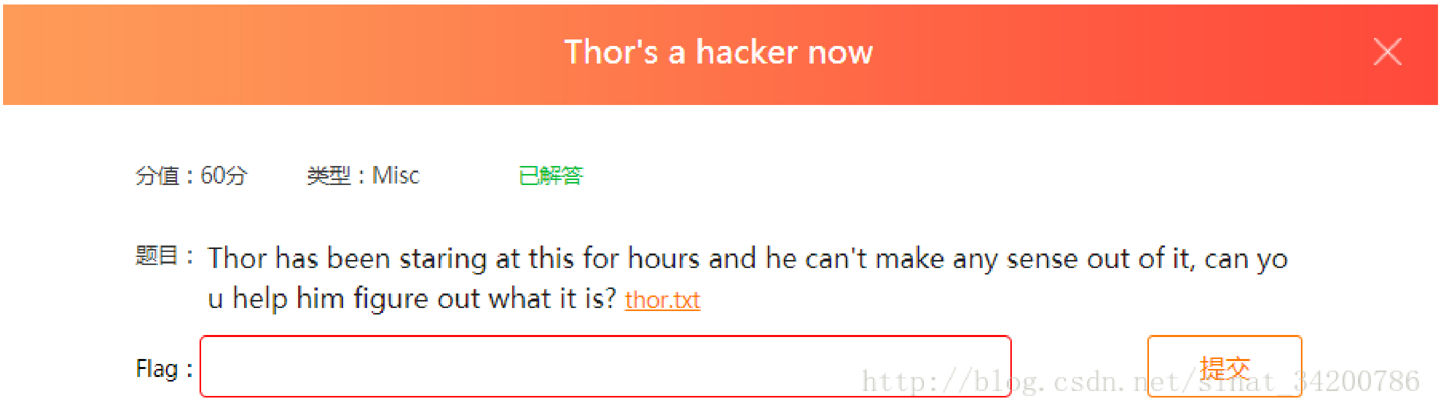
```
01001001 01100011 01100101 01000011 01010100 01000110 01111011 01100001 01101100 00110001 01011111
01101101 01111001 01011111 01100010 01100001 01110011 01100101 01110011 01011111 01100001 01110010
01100101 01011111 01111001 01101111 01110101 01110010 01110011 01011111 01100001 01101110 01100100
01011111 01100001 01101100 01101100 01011111 01111001 00110000 01110101 01110010 01011111 01100010
01100001 01110011 01100101 01110011 01011111 01100001 01110010 01100101 01011111 01101101 01101001
01101110 01100101 01111101
```

```
将前面两个转为十进制再转为ASCII对照一下就可以发现是要将所有二进制转为ASCII
```

```
f = open('01.txt','r')
numList = []

for line in f.readlines():
    numList += list(line.rstrip().split(' '))

s = ''
for n in numList:
    s += chr(int(n,2))

print(s)
```

## Thor’s a hacker now

原题



Thor's a hacker now

分值：60分　　类型：Misc　　已解答

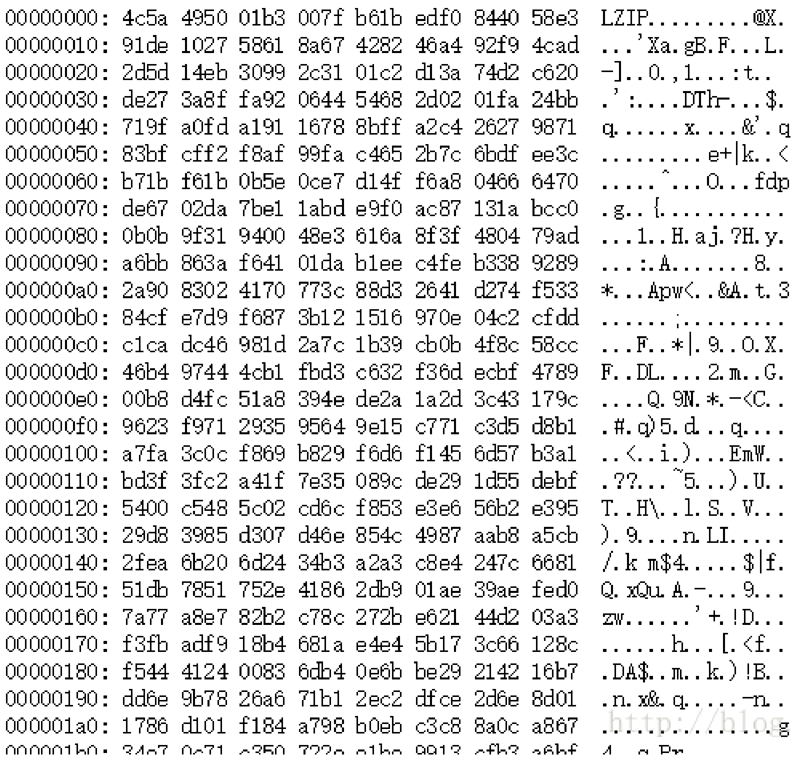
题目：Thor has been staring at this for hours and he can't make any sense out of it, can you help him figure out what it is? thor.txt

Flag：

解题思路

```
LZIP是Linux压缩文件
```

WriteUp

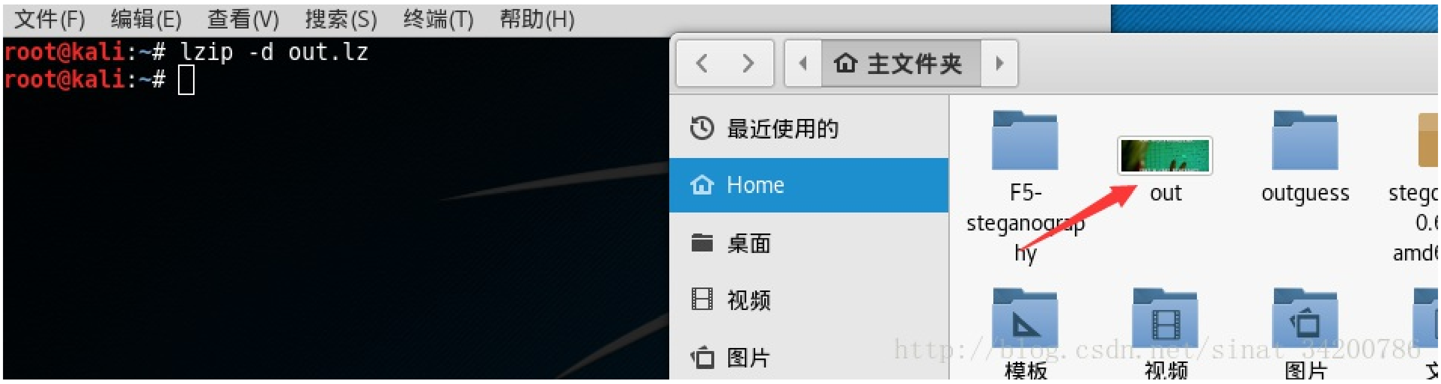


```
00000000: 4c5a 4950 01b3 007f b61b edf0 8440 58e3  LZIP.........@X.
00000010: 91de 1027 5861 8a67 4282 46a4 92f9 4cad  ...'Xa.gB.F...L.
00000020: 2d5d 14eb 3099 2c31 01c2 d13a 74d2 c620  -]..0.,1...:t..
00000030: de27 3a8f fa92 0644 5468 2d02 01fa 24bb  .':....DTh-...$.
00000040: 719f a0fd a191 1678 8bff a2c4 2627 9871  q......x....&'.q
00000050: 83bf cff2 f8af 99fa c465 2b7c 6bdf ee3c  .........e+|k..<
00000060: b71b f61b 0b5e 0ce7 d14f f6a8 0466 6470  .....^...O...fdp
00000070: de67 02da 7be1 1abd e9f0 ac87 131a bcc0  .g..{...........
00000080: 0b0b 9f31 9400 48e3 616a 8f3f 4804 79ad  ...1..H.aj.?H.y.
00000090: a6bb 863a f641 01da b1ee c4fe b338 9289  ...:.A.......8..
000000a0: 2a90 8302 4170 773c 88d3 2641 d274 f533  *...Apw<..&A.t.3
000000b0: 84cf e7d9 f687 3b12 1516 970e 04c2 cfdd  ......;.........
000000c0: c1ca dc46 981d 2a7c 1b39 cb0b 4f8c 58cc  ...F..*|.9..O.X.
000000d0: 46b4 9744 4cb1 fbd3 c632 f36d ecbf 4789  F..DL....2.m..G.
000000e0: 00b8 d4fc 51a8 394e de2a 1a2d 3c43 179c  ....Q.9N.*.-<C..
000000f0: 9623 f971 2935 9564 9e15 c771 c3d5 d8b1  .#.q)5.d...q....
00000100: a7fa 3c0c f869 b829 f6d6 f145 6d57 b3a1  ..<..i.)...EmW..
00000110: bd3f 3fc2 a41f 7e35 089c de29 1d55 debf  .??...~5...).U..
00000120: 5400 c548 5c02 cd6c f853 e3e6 56b2 e395  T..H\..l.S..V...
00000130: 29d8 3985 d307 d46e 854c 4987 aab8 a5cb  ).9....n.LI.....
00000140: 2fea 6b20 6d24 34b3 a2a3 c8e4 247c 6681  /.k m$4.....$|f.
00000150: 51db 7851 752e 4186 2db9 01ae 39ae fed0  Q.xQu.A.-...9...
00000160: 7a77 a8e7 82b2 c78c 272b e621 44d2 03a3  zw......'+.!D...
00000170: f3fb adf9 18b4 681a e4e4 5b17 3c66 128c  ......h...[.<f..
00000180: f544 4124 0083 6db4 0e6b be29 2142 16b7  .DA$..m..k.)!B..
00000190: dd6e 9b78 26a6 71b1 2ec2 dfce 2d6e 8d01  .n.x&.q.....-n..
000001a0: 1786 d101 f184 a798 b0eb c3c8 8a0c a867  ...............g
000001b0: 34e7 0c71 c350 722e e1be 9913 cfb3 a6bf  4..q.Pr
```

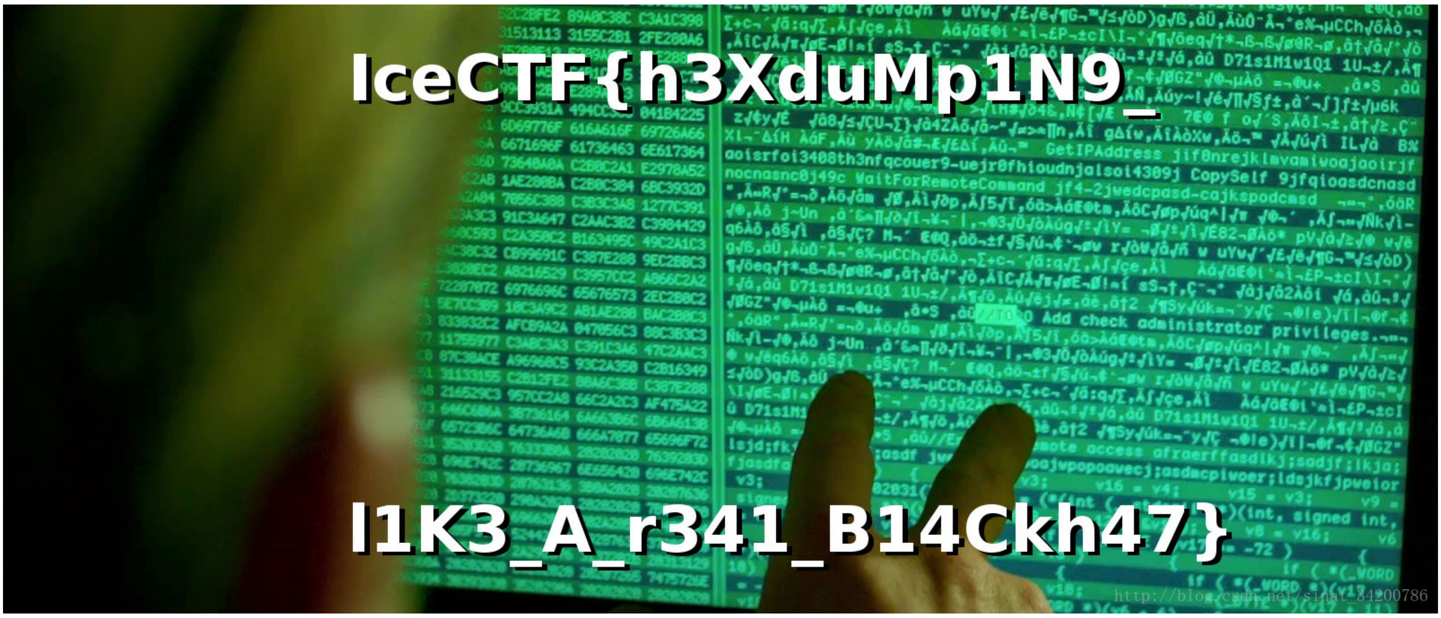
```
很明显这是某种文件在十六进制查看工具下的数据分布，开头的前几个bytes说明了文件类型：LZIP
```

```
将数据部分提取出来后粘贴到HxD中生成文件，Linux下解压即可
```

```
f = open('in.txt','r')
f2 = open('out.txt','w')

for line in f.readlines():
    line = line[10:50]+'\n'
    f2.write(line)

f.close()
f2.close()
```



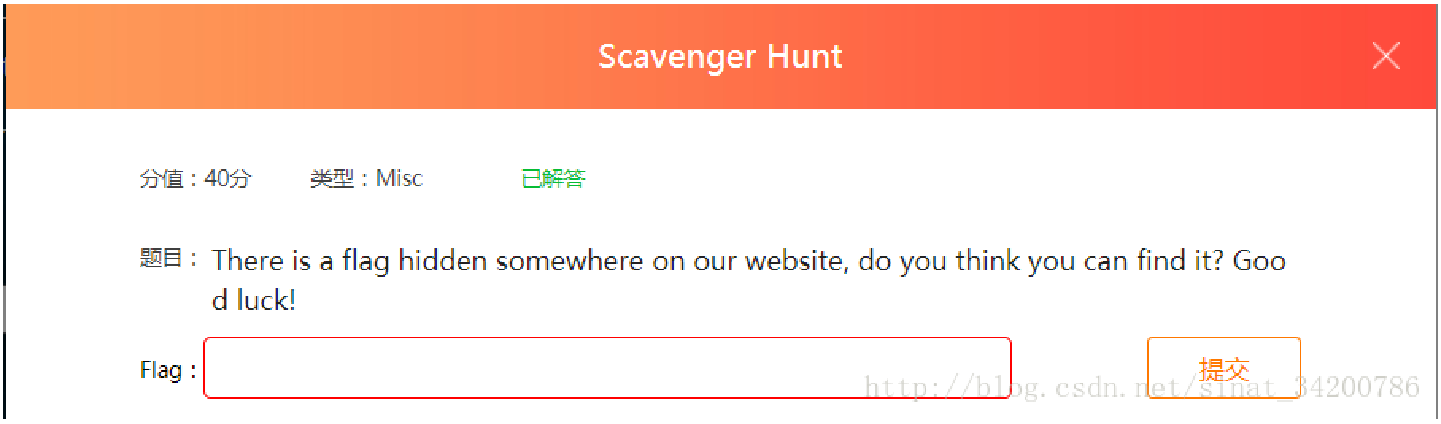
```
root@kali:~# lzip -d out.lz
root@kali:~# 
```



---

## Scavenger Hunt

原题



Scavenger Hunt

分值：40分　　类型：Misc　　已解答

题目：There is a flag hidden somewhere on our website, do you think you can find it? Good luck!

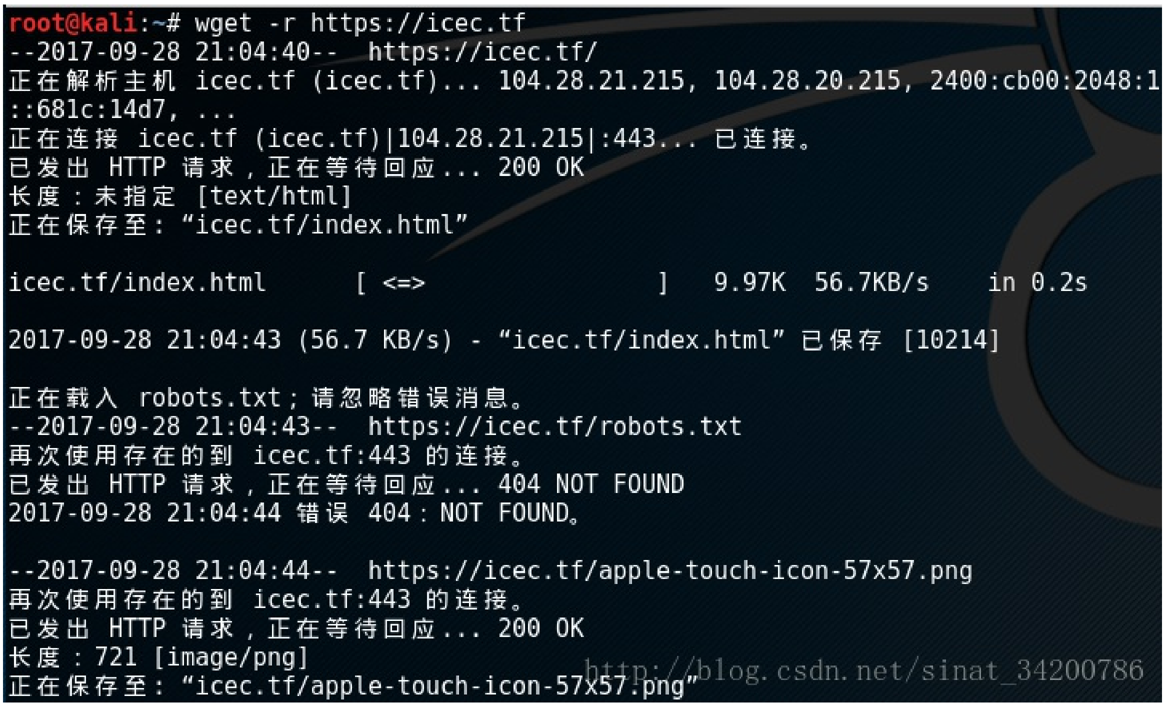
Flag：

解题思路

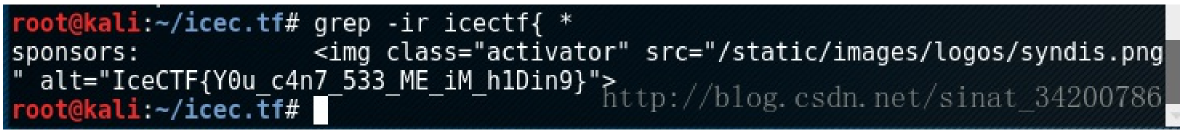
```
那就找吧
```

WriteUp

```
整个网站下载下来就容易找了
```



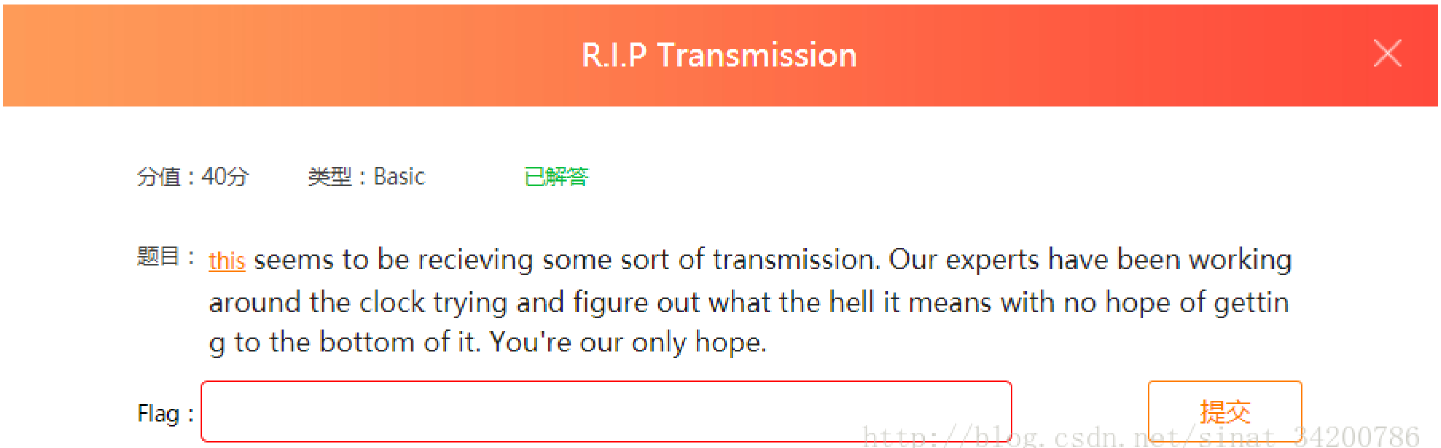
```
root@kali:~# wget -r https://icec.tf
--2017-09-28 21:04:40--  https://icec.tf/
正在解析主机 icec.tf (icec.tf)... 104.28.21.215, 104.28.20.215, 2400:cb00:2048:1
::681c:14d7, ...
正在连接 icec.tf (icec.tf)|104.28.21.215|:443... 已连接。
已发出 HTTP 请求，正在等待回应... 200 OK
长度：未指定 [text/html]
正在保存至: “icec.tf/index.html”

icec.tf/index.html      [ <=>                ]   9.97K  56.7KB/s    in 0.2s

2017-09-28 21:04:43 (56.7 KB/s) - “icec.tf/index.html” 已保存 [10214]

正在载入 robots.txt；请忽略错误消息。
--2017-09-28 21:04:43--  https://icec.tf/robots.txt
再次使用存在的到 icec.tf:443 的连接。
已发出 HTTP 请求，正在等待回应... 404 NOT FOUND
2017-09-28 21:04:44 错误 404：NOT FOUND。

--2017-09-28 21:04:44--  https://icec.tf/apple-touch-icon-57x57.png
再次使用存在的到 icec.tf:443 的连接。
已发出 HTTP 请求，正在等待回应... 200 OK
长度：721 [image/png]
正在保存至: “icec.tf/apple-touch-icon-57x57.png”
```



```
root@kali:~/icec.tf# grep -ir icectf{ *
sponsors:            <img class="activator" src="/static/images/logos/syndis.png
" alt="IceCTF{Y0u_c4n7_533_ME_iM_h1Din9}">
root@kali:~/icec.tf# 
```

## R.I.P Transmission

原题



R.I.P Transmission

分值：40分 类型：Basic 已解答

题目：this seems to be recieving some sort of transmission. Our experts have been working around the clock trying and figure out what the hell it means with no hope of getting to the bottom of it. You're our only hope.

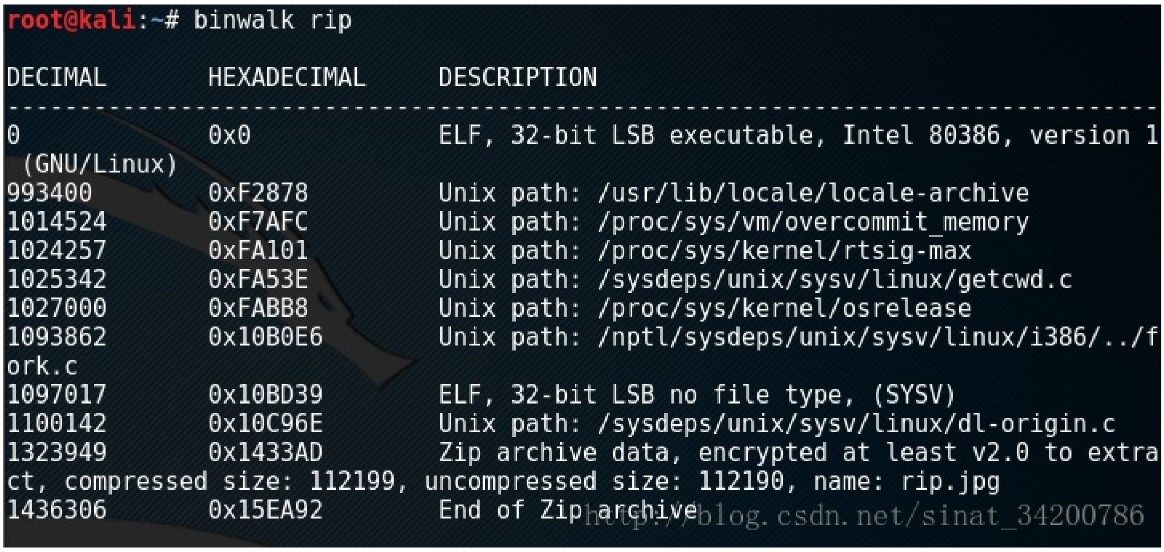
Flag：

解题思路

```
Binwalk，爆破
```

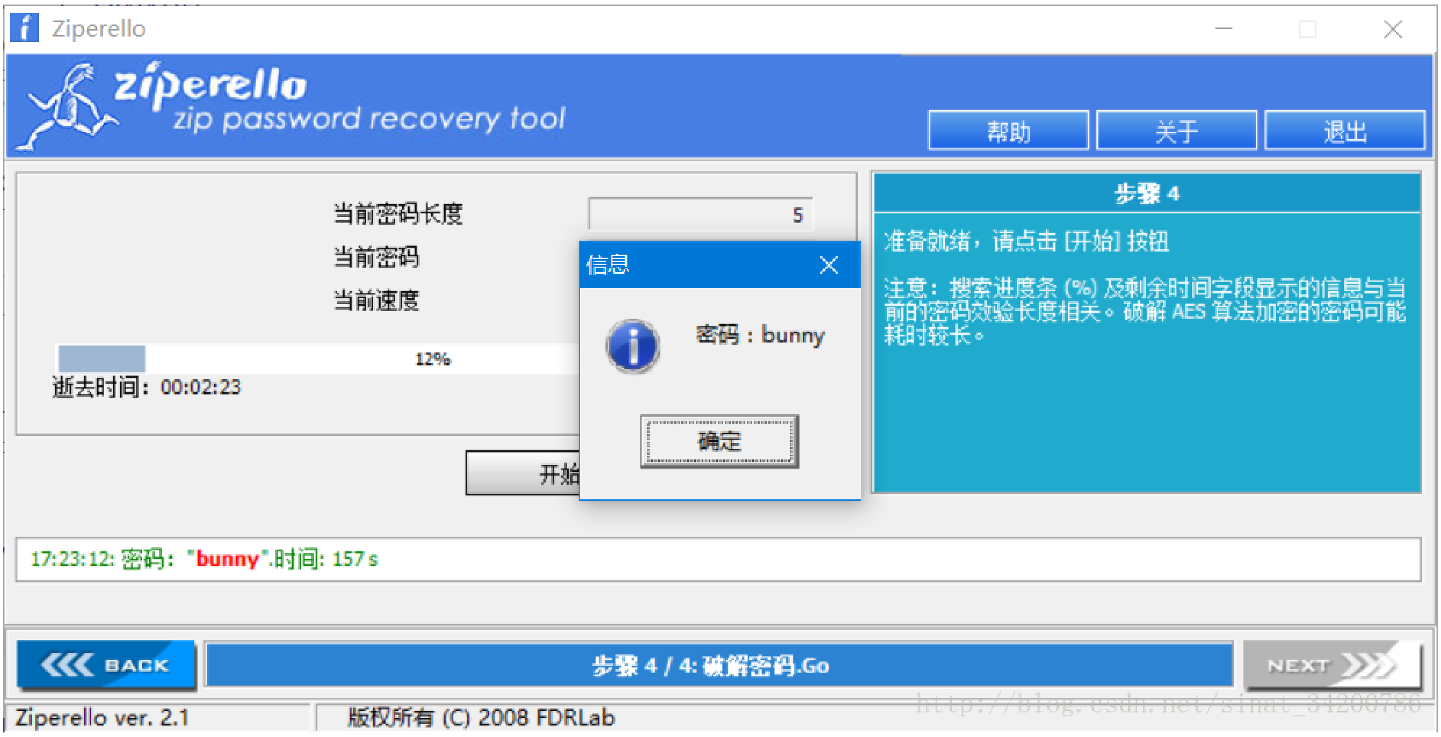
WriteUp

```
直接仍Binwalk里面可以发现隐藏有zip文件
```



```
root@kali:~# binwalk rip

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             ELF, 32-bit LSB executable, Intel 80386, version 1
 (GNU/Linux)
993400        0xF2878         Unix path: /usr/lib/locale/locale-archive
1014524       0xF7AFC         Unix path: /proc/sys/vm/overcommit_memory
1024257       0xFA101         Unix path: /proc/sys/kernel/rtsig-max
1025342       0xFA53E         Unix path: /sysdeps/unix/sysv/linux/getcwd.c
1027000       0xFABB8         Unix path: /proc/sys/kernel/osrelease
1093862       0x10B0E6        Unix path: /nptl/sysdeps/unix/sysv/linux/i386/../f
ork.c
1097017       0x10BD39        ELF, 32-bit LSB no file type, (SYSV)
1100142       0x10C96E        Unix path: /sysdeps/unix/sysv/linux/dl-origin.c
1323949       0x1433AD        Zip archive data, encrypted at least v2.0 to extra
ct, compressed size: 112199, uncompressed size: 112190, name: rip.jpg
1436306       0x15EA92        End of Zip archive
```

```
提取出来发现需要密码，爆破即可
```



步骤 4

准备就绪，请点击 [开始] 按钮

注意：搜索进度条 (%) 及剩余时间字段显示的信息与当前的密码效验长度相关。破解 AES 算法加密的密码可能耗时较长。

当前密码长度 5

当前密码

当前速度

12%

逝去时间：00:02:23

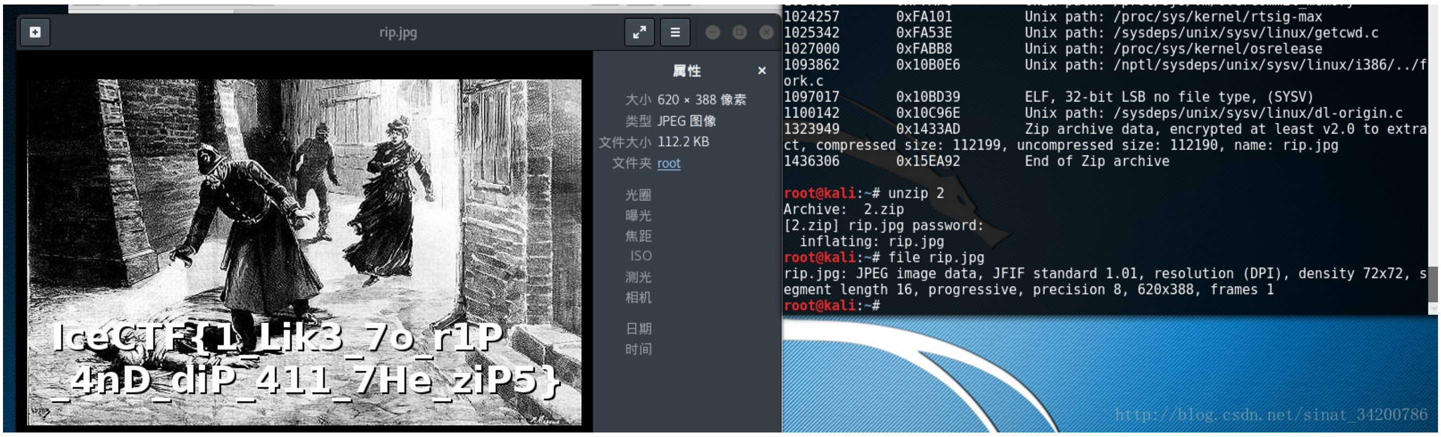
信息

密码：bunny

17:23:12: 密码："bunny".时间: 157 s

步骤 4 / 4: 破解密码.Go

Ziperello ver. 2.1　版权所有 (C) 2008 FDRLab



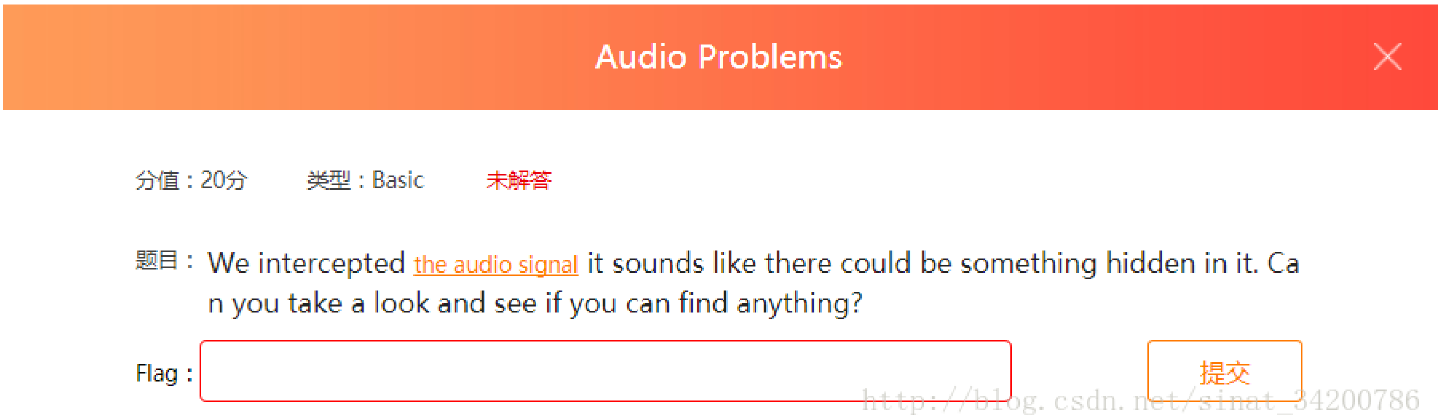
IceCTF{1_Lik3_7o_r1P_4nD_diP_411_7He_ziP5}

大小 620 × 388 像素
类型 JPEG 图像
文件大小 112.2 KB
文件夹 root

```
1024257       0xFA101         Unix path: /proc/sys/kernel/rtsig-max
1025342       0xFA53E         Unix path: /sysdeps/unix/sysv/linux/getcwd.c
1027000       0xFABB8         Unix path: /proc/sys/kernel/osrelease
1093862       0x10B0E6        Unix path: /nptl/sysdeps/unix/sysv/linux/i386/../f
ork.c
1097017       0x10BD39        ELF, 32-bit LSB no file type, (SYSV)
1100142       0x10C96E        Unix path: /sysdeps/unix/sysv/linux/dl-origin.c
1323949       0x1433AD        Zip archive data, encrypted at least v2.0 to extra
ct, compressed size: 112199, uncompressed size: 112190, name: rip.jpg
1436306       0x15EA92        End of Zip archive

root@kali:~# unzip 2
Archive:  2.zip
[2.zip] rip.jpg password:
  inflating: rip.jpg
root@kali:~# file rip.jpg
rip.jpg: JPEG image data, JFIF standard 1.01, resolution (DPI), density 72x72, s
egment length 16, progressive, precision 8, 620x388, frames 1
root@kali:~#
```

## Audio Problems

原题



Audio Problems

分值：20分　　类型：Basic　　未解答

题目：We intercepted the audio signal it sounds like there could be something hidden in it. Can you take a look and see if you can find anything?

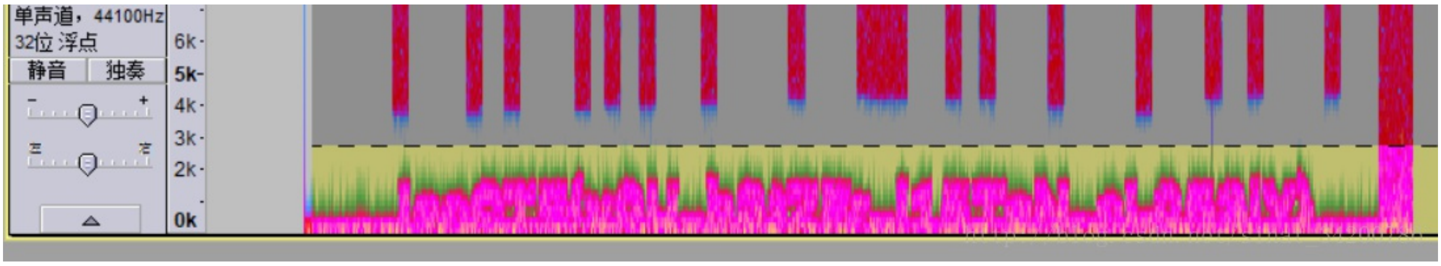
Flag：

解题思路

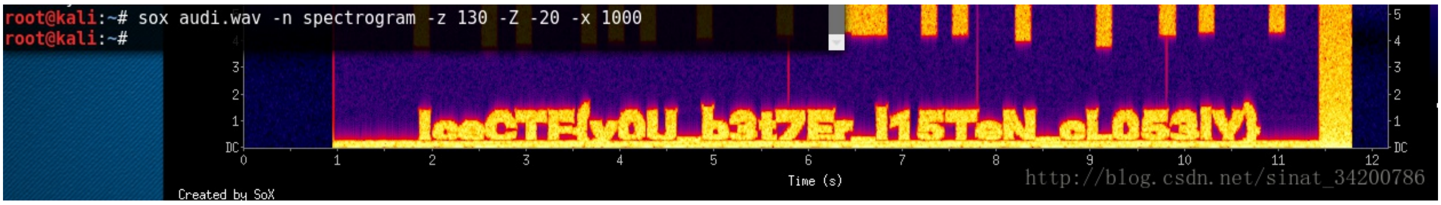
```
Audacity? 还是要sox
```

WriteUp

```
惯例先上Audacity，频谱图有发现，不过好像太模糊了
```



```
怎么看都看不清，怎么办，sox试试啊
```



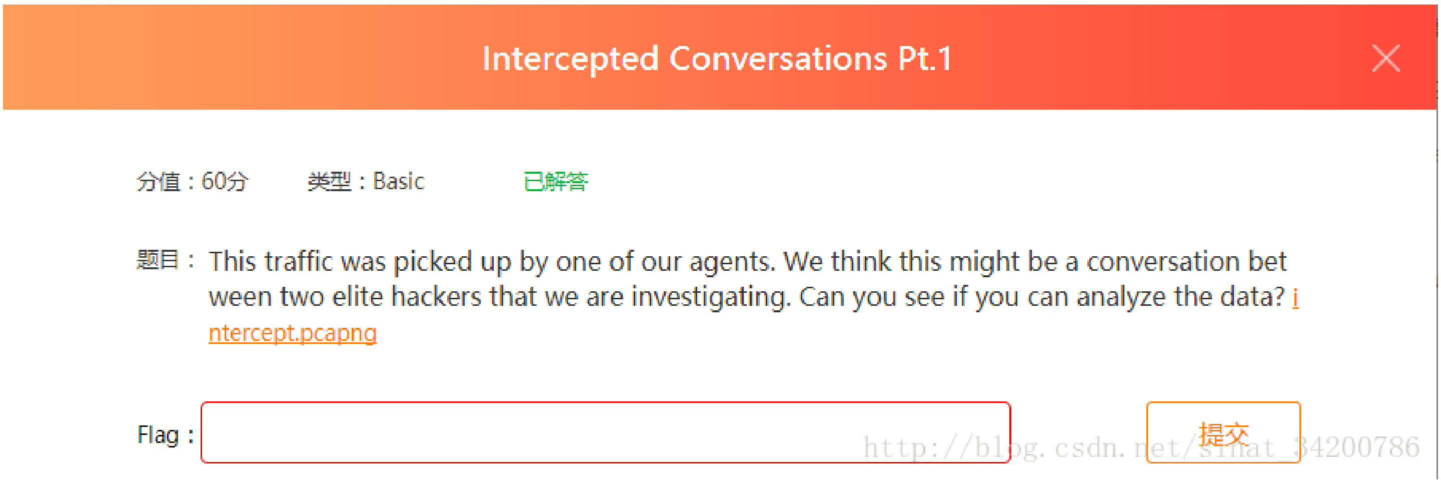
```
root@kali:~# sox audi.wav -n spectrogram -z 130 -Z -20 -x 1000
root@kali:~#
```

涨姿势点

```
sox, the Swiss Army knife of sound processing programs
掌握基本用法
```

## Intercepted Conversations Pt.1

原题



Intercepted Conversations Pt.1

分值：60分　类型：Basic　已解答

题目： This traffic was picked up by one of our agents. We think this might be a conversation between two elite hackers that we are investigating. Can you see if you can analyze the data? intercept.pcapng

Flag :

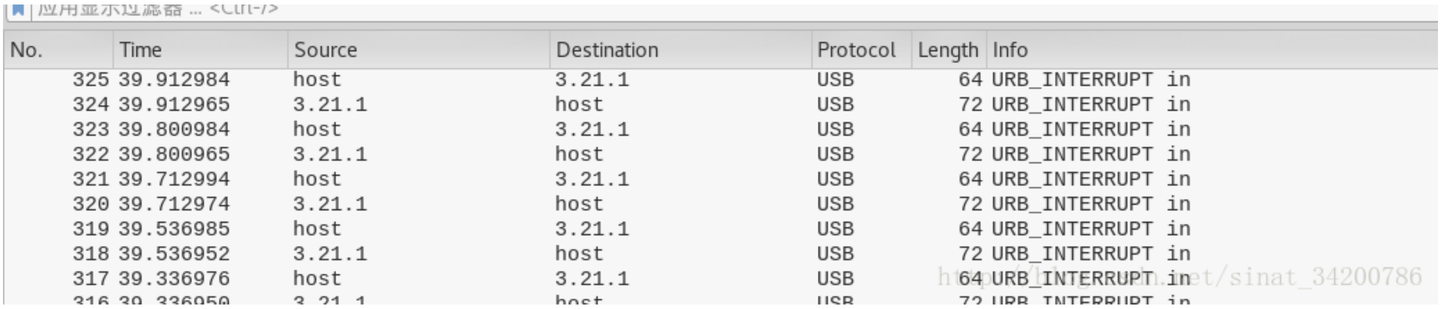
解题思路

```
键盘数据包，tshark提取数据，Dvorak键盘
```

WriteUp

## 参考资料

```
从后缀 '.pcapng' 为截取的数据包，Wireshark查看证实是usb通信的数据包
```

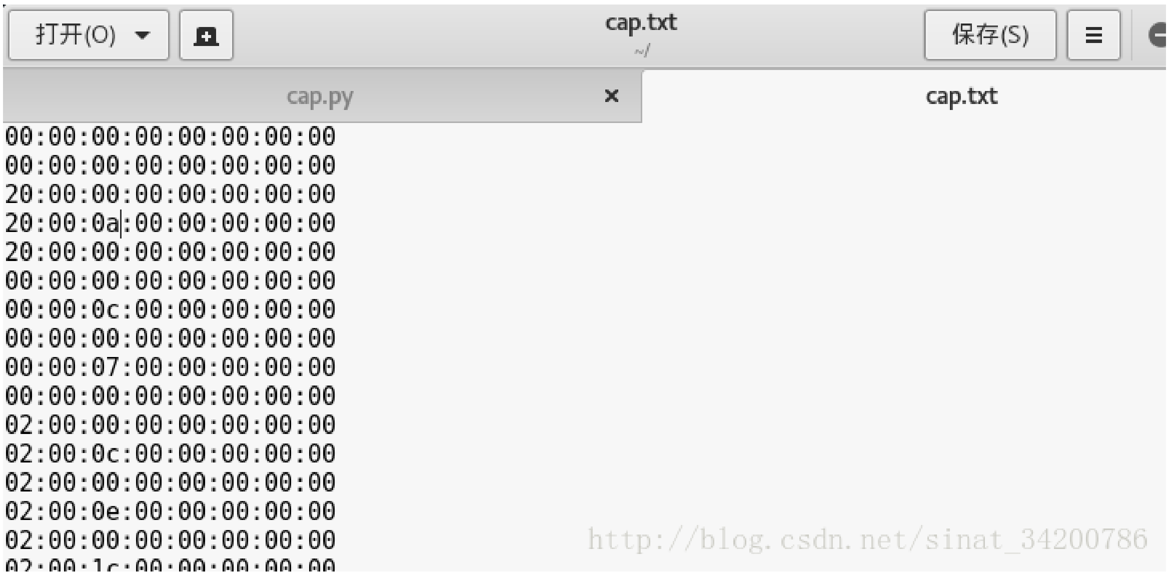


| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 325 | 39.912984 | host | 3.21.1 | USB | 64 | URB_INTERRUPT in |
| 324 | 39.912965 | 3.21.1 | host | USB | 72 | URB_INTERRUPT in |
| 323 | 39.800984 | host | 3.21.1 | USB | 64 | URB_INTERRUPT in |
| 322 | 39.800965 | 3.21.1 | host | USB | 72 | URB_INTERRUPT in |
| 321 | 39.712994 | host | 3.21.1 | USB | 64 | URB_INTERRUPT in |
| 320 | 39.712974 | 3.21.1 | host | USB | 72 | URB_INTERRUPT in |
| 319 | 39.536985 | host | 3.21.1 | USB | 64 | URB_INTERRUPT in |
| 318 | 39.536952 | 3.21.1 | host | USB | 72 | URB_INTERRUPT in |
| 317 | 39.336976 | host | 3.21.1 | USB | 64 | URB_INTERRUPT in |
| 316 | 39.336950 | 3.21.1 | host | USB | 72 | URB_INTERRUPT in |

```
那么应该就是要从里面提取键盘输入的数据，用tshark完成
```

```
tshark -r interceptpt1.pcapng -T fields -e usb.capdata -Y 'usb.capdata && usb.transfer_type == 0x01 && frame.len == 72' >cap.txt
```

```
提取的数据如下
02 和 20 开头的数据表示输入的是大写
02 = 20 = shift
```

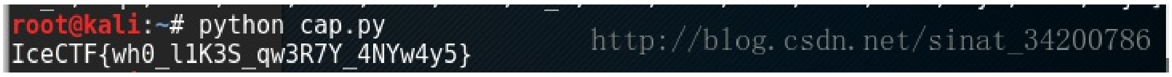


```
00:00:00:00:00:00:00:00
00:00:00:00:00:00:00:00
20:00:00:00:00:00:00:00
20:00:0a:00:00:00:00:00
20:00:00:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:0c:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:07:00:00:00:00:00
00:00:00:00:00:00:00:00
02:00:00:00:00:00:00:00
02:00:0c:00:00:00:00:00
02:00:00:00:00:00:00:00
02:00:0e:00:00:00:00:00
02:00:00:00:00:00:00:00
02:00:1c:00:00:00:00:00
```

```
学过汇编应该都知道系统在读取键盘输入时在缓冲区存储的是字母的代码，例如
G的代码是0x0a，前面的02或者20控制大写，那么我们就要把代码转换为字母
```

```
可是转换后会发现根本就不是flag，这个时候就要用到另一种键盘，Dvorak键盘，它的字母排列和我们常用的键盘不同，只要在根据
```

```
#代码修改自 @Jhonathan Davi
hids_codes = {"0x04":"a","0x05":"b","0x06":"c","0x07":"d","0x08":"e","0x09":"f","0x0A":"g","0x0B":"h","0x0C":"i"
,"0x0D":"j","0x0E":"k","0x0F":"l","0x10":"m","0x11":"n","0x12":"o","0x13":"p","0x14":"q","0x15":"r","0x16":"s","
0x17":"t","0x18":"u","0x19":"v","0x1A":"w","0x1B":"x","0x1C":"y","0x1D":"z","0x1E":"1","0x1F":"2","0x20":"3","0x
21":"4","0x22":"5","0x23":"6","0x24":"7","0x25":"8","0x26":"9","0x27":"0","0x36":",","0x33":":","0x28":"\n","0x2
C":" ","0x2D":"_","0x2E":"=","0x2F":"{","0x30":"}"}

layout_dvorak = { 'q':"'", 'w':',', 'e':'.', 'r':'p', 't':'y', 'y':'f', 'u':'g', 'i':'c', 'o':'r', 'p':'l', '_':
'_', ':':'S','[':'/', '{':'{', '}':'}' ,']':'=','a':'a', 's':'o', 'd':'e', 'f':'u', 'g':'i', 'h':'d', 'j':'h', '
k':'t', 'l':'n', ';':'s', "'":'-','z':';', 'x':'q', 'c':'j', 'v':'k', 'b':'x', 'n':'b', 'm':'m', ',':'w', '.':'v
', '.':'z',' ':' ','Q':"'", 'W':',', 'E':'.', 'R':'P', 'T':'Y', 'Y':'F', 'U':'G', 'I':'C', 'O':'R', 'P':'L','A':
'A', 'S':'O', 'D':'E', 'F':'U', 'G':'I', 'H':'D', 'J':'H', 'K':'T', 'L':'N', ';':'S', "'":'-','Z':';', 'X':'Q',
'C':'J', 'V':'K', 'B':'X', 'N':'B', 'M':'M','0':'0','1':'1','2':'2','3':'3','4':'4','5':'5','6':'6','7':'7','7':
'7','8':'8','9':'9'}

flag = ''

file = open('cap.txt','r')
for line in file.readlines():
    spli = line.split(':')
    conv = '0x'+spli[2].upper()
    if conv in hids_codes:
        if spli[0] == '00':
            flag += layout_dvorak[hids_codes[conv]]
        else:
            flag += layout_dvorak[hids_codes[conv].upper()]

print(flag)
```



```
root@kali:~# python cap.py
IceCTF{wh0_l1K3S_qw3R7Y_4NYw4y5}
```

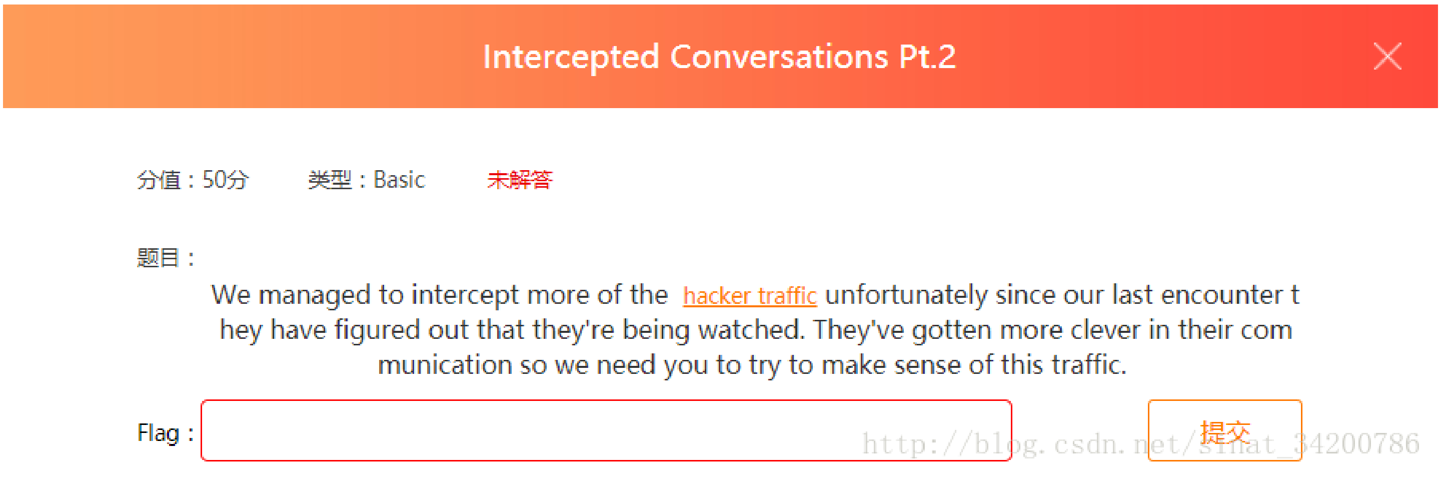
涨姿势点

```
    Wireshark能抓取不同端口的数据包
    tshark的基本用法
    Dvorak键盘的排列不同于普通键盘
```

---

## Intercepted Conversations Pt.1

原题



Intercepted Conversations Pt.2

分值：50分　类型：Basic　未解答

题目：

We managed to intercept more of the hacker traffic unfortunately since our last encounter they have figured out that they're being watched. They've gotten more clever in their communication so we need you to try to make sense of this traffic.

Flag：

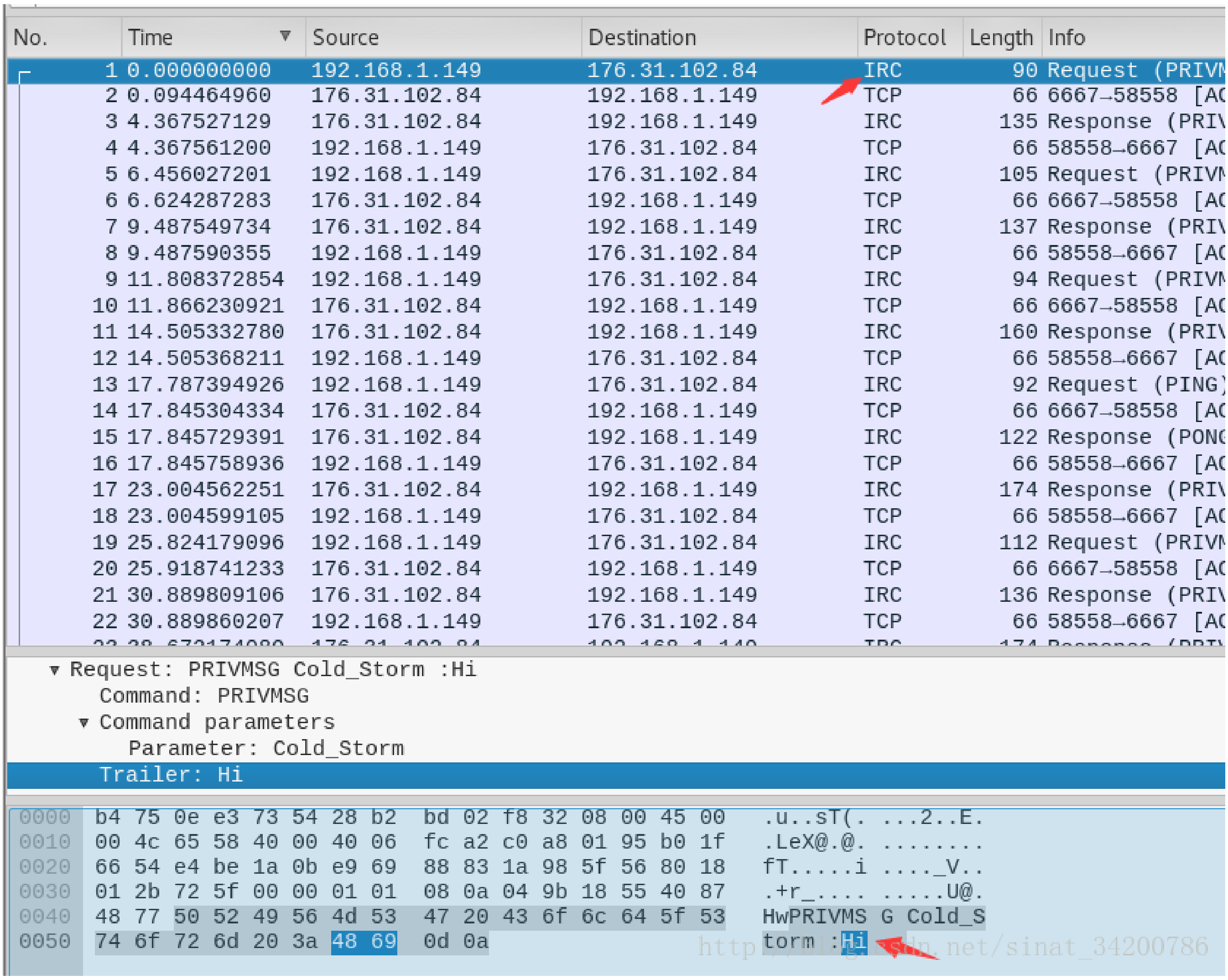
## 解题思路

```
Wireshark发现关键信息及提取文件，反编译pyc，算法逆向
```

## WriteUp

## 参考资料

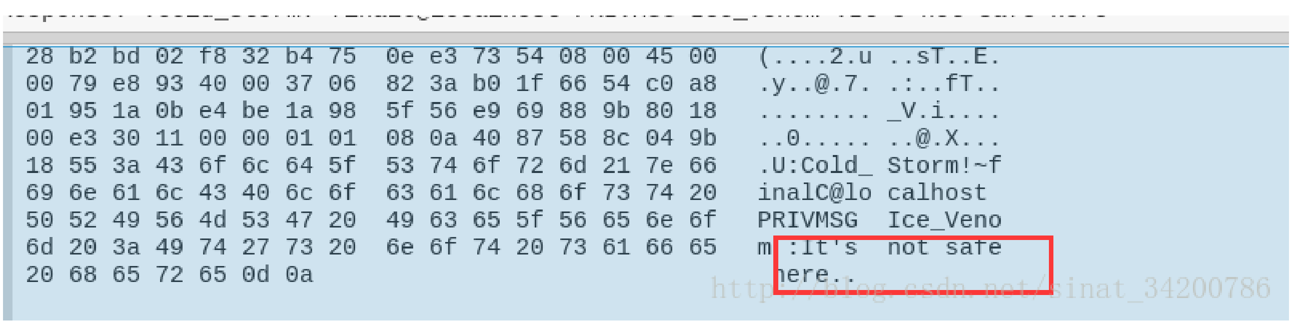
```
Wireshark分析数据包可以发现有个IRC网络的，解析几个数据包可以发现有人正在通信
```



```
No.     Time            Source           Destination      Protocol  Length Info
  1 0.000000000    192.168.1.149    176.31.102.84    IRC          90 Request (PRIVM
  2 0.094464960    176.31.102.84    192.168.1.149    TCP          66 6667→58558 [AC
  3 4.367527129    176.31.102.84    192.168.1.149    IRC         135 Response (PRIV
  4 4.367561200    192.168.1.149    176.31.102.84    TCP          66 58558→6667 [AC
  5 6.456027201    192.168.1.149    176.31.102.84    IRC         105 Request (PRIVM
  6 6.624287283    176.31.102.84    192.168.1.149    TCP          66 6667→58558 [AC
  7 9.487549734    176.31.102.84    192.168.1.149    IRC         137 Response (PRIV
  8 9.487590355    192.168.1.149    176.31.102.84    TCP          66 58558→6667 [AC
  9 11.808372854   192.168.1.149    176.31.102.84    IRC          94 Request (PRIVM
 10 11.866230921   176.31.102.84    192.168.1.149    TCP          66 6667→58558 [AC
 11 14.505332780   176.31.102.84    192.168.1.149    IRC         160 Response (PRIV
 12 14.505368211   192.168.1.149    176.31.102.84    TCP          66 58558→6667 [AC
 13 17.787394926   192.168.1.149    176.31.102.84    IRC          92 Request (PING)
 14 17.845304334   176.31.102.84    192.168.1.149    TCP          66 6667→58558 [AC
 15 17.845729391   176.31.102.84    192.168.1.149    IRC         122 Response (PONG
 16 17.845758936   192.168.1.149    176.31.102.84    TCP          66 58558→6667 [AC
 17 23.004562251   176.31.102.84    192.168.1.149    IRC         174 Response (PRIV
 18 23.004599105   192.168.1.149    176.31.102.84    TCP          66 58558→6667 [AC
 19 25.824179096   192.168.1.149    176.31.102.84    IRC         112 Request (PRIVM
 20 25.918741233   176.31.102.84    192.168.1.149    TCP          66 6667→58558 [AC
 21 30.889809106   176.31.102.84    192.168.1.149    IRC         136 Response (PRIV
 22 30.889860207   192.168.1.149    176.31.102.84    TCP          66 58558→6667 [AC
 23 38.673174080   176.31.102.84    192.168.1.149    IRC         174 Response (PRIV
```

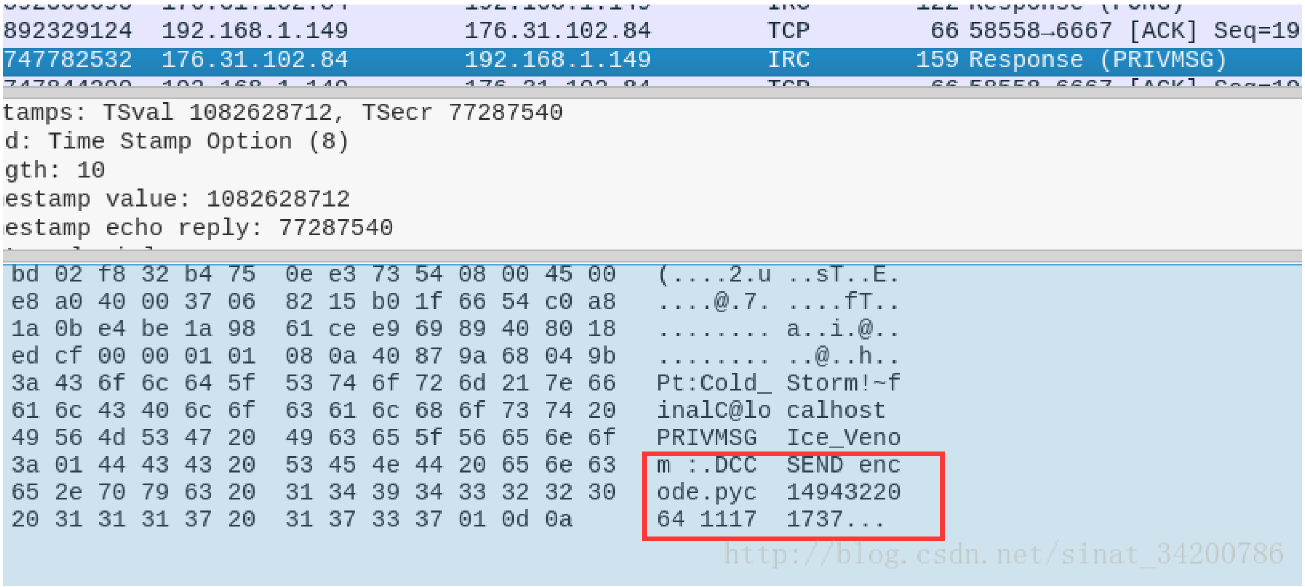
```
  ▾ Request: PRIVMSG Cold_Storm :Hi
      Command: PRIVMSG
    ▾ Command parameters
        Parameter: Cold_Storm
      Trailer: Hi
```

```
0000  b4 75 0e e3 73 54 28 b2  bd 02 f8 32 08 00 45 00   .u..sT(. ...2..E.
0010  00 4c 65 58 40 00 40 06  fc a2 c0 a8 01 95 b0 1f   .LeX@.@. ........
0020  66 54 e4 be 1a 0b e9 69  88 83 1a 98 5f 56 80 18   fT.....i ...._V..
0030  01 2b 72 5f 00 00 01 01  08 0a 04 9b 18 55 40 87   .+r_.... .....U@.
0040  48 77 50 52 49 56 4d 53  47 20 43 6f 6c 64 5f 53   HwPRIVMS G Cold_S
0050  74 6f 72 6d 20 3a 48 69  0d 0a                     torm :Hi
```



```
esponse: :Cold_Storm!~finalC@localhost PRIVMSG Ice_Venom :It's not safe here
```

```
 28 b2 bd 02 f8 32 b4 75  0e e3 73 54 08 00 45 00   (....2.u ..sT..E.
 00 79 e8 93 40 00 37 06  82 3a b0 1f 66 54 c0 a8   .y..@.7. .:..fT..
 01 95 1a 0b e4 be 1a 98  5f 56 e9 69 88 9b 80 18   ........ _V.i....
 00 e3 30 11 00 00 01 01  08 0a 40 87 58 8c 04 9b   ..0..... ..@.X...
 18 55 3a 43 6f 6c 64 5f  53 74 6f 72 6d 21 7e 66   .U:Cold_ Storm!~f
 69 6e 61 6c 43 40 6c 6f  63 61 6c 68 6f 73 74 20   inalC@lo calhost
 50 52 49 56 4d 53 47 20  49 63 65 5f 56 65 6e 6f   PRIVMSG  Ice_Veno
 6d 20 3a 49 74 27 73 20  6e 6f 74 20 73 61 66 65   m :It's  not safe
 20 68 65 72 65 0d 0a                                here..
```

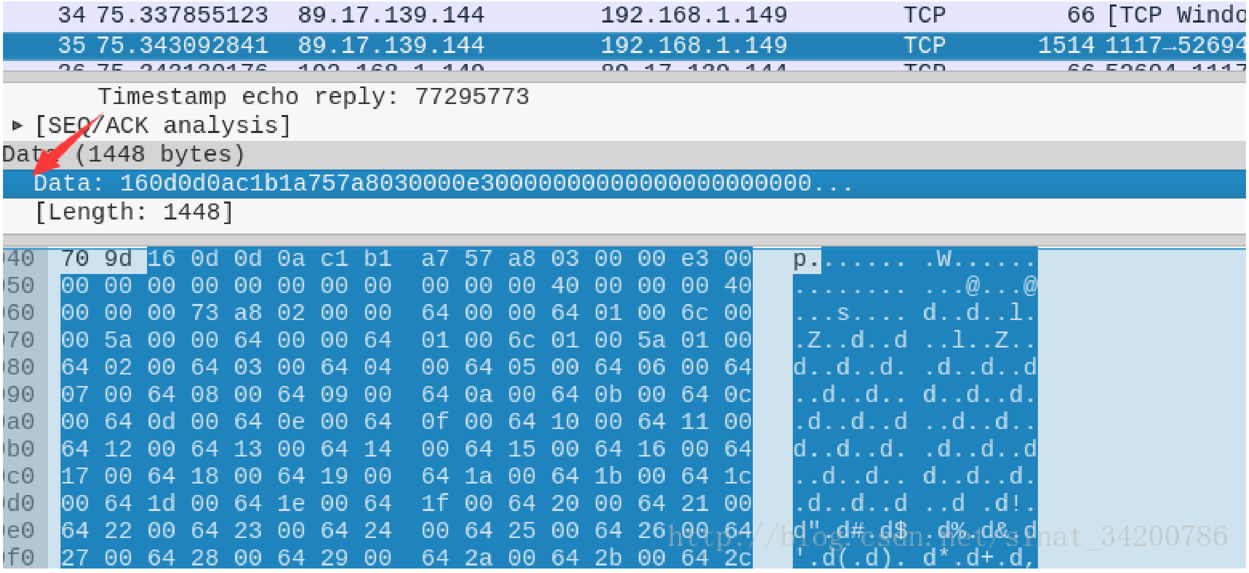


```
892329124  192.168.1.149         176.31.102.84         TCP          66 58558→6667 [ACK] Seq=19
747782532  176.31.102.84         192.168.1.149         IRC         159 Response (PRIVMSG)
```

```
tamps: TSval 1082628712, TSecr 77287540
d: Time Stamp Option (8)
gth: 10
estamp value: 1082628712
estamp echo reply: 77287540
```

```
bd 02 f8 32 b4 75  0e e3 73 54 08 00 45 00   (....2.u ..sT..E.
e8 a0 40 00 37 06  82 15 b0 1f 66 54 c0 a8   ....@.7. ....fT..
1a 0b e4 be 1a 98  61 ce e9 69 89 40 80 18   ........ a..i.@..
ed cf 00 00 01 01  08 0a 40 87 9a 68 04 9b   ........ ..@..h..
3a 43 6f 6c 64 5f  53 74 6f 72 6d 21 7e 66   Pt:Cold_ Storm!~f
61 6c 43 40 6c 6f  63 61 6c 68 6f 73 74 20   inalC@lo calhost 
49 56 4d 53 47 20  49 63 65 5f 56 65 6e 6f   PRIVMSG  Ice_Veno
3a 01 44 43 43 20  53 45 4e 44 20 65 6e 63   m :.DCC  SEND enc
65 2e 70 79 63 20  31 34 39 34 33 32 32 30   ode.pyc  14943220
20 31 31 31 37 20  31 37 33 37 01 0d 0a      64 1117  1737...
```

```
从截取的对话可以发现通信方发送了一个名为 encode.pyc 的文件，我们可以在接下来的几个TCP协议的数据包的data段获取该文件
```

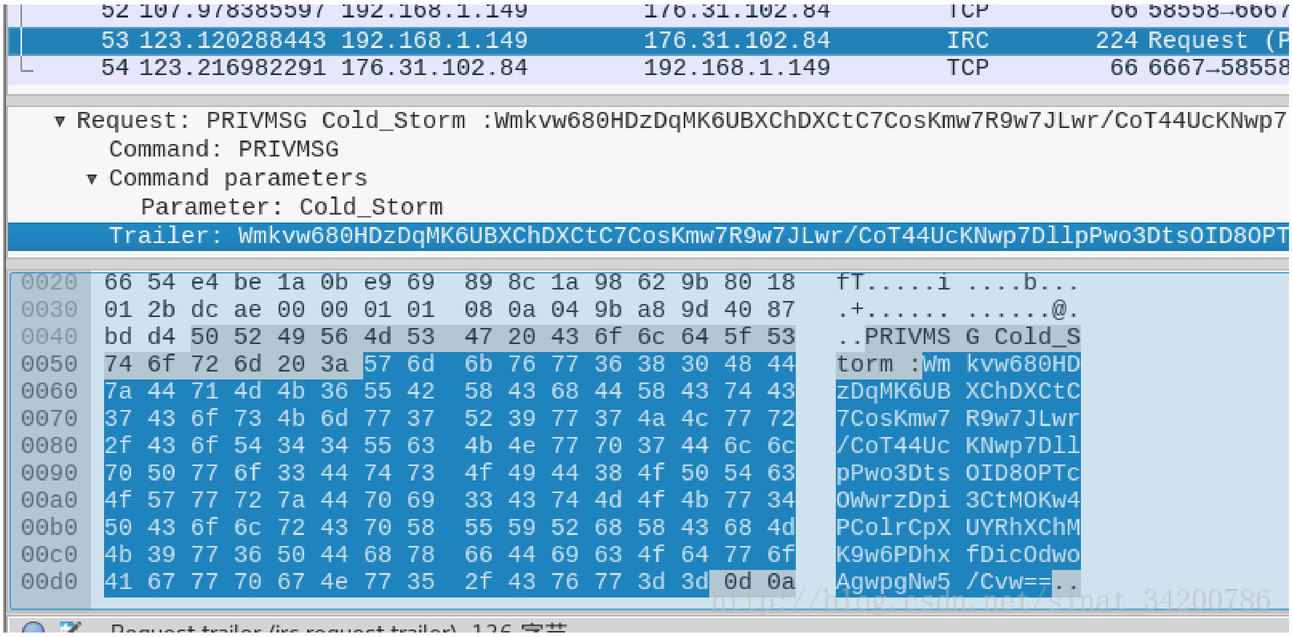


```
 34 75.337855123  89.17.139.144         192.168.1.149         TCP           66 [TCP Windo
 35 75.343092841  89.17.139.144         192.168.1.149         TCP         1514 1117→52694
```

```
      Timestamp echo reply: 77295773
  ▸ [SEQ/ACK analysis]
Data (1448 bytes)
   Data: 160d0d0ac1b1a757a8030000e3000000000000000000000000...
   [Length: 1448]
```

```
40  70 9d 16 0d 0d 0a c1 b1  a7 57 a8 03 00 00 e3 00   p....... .W......
50  00 00 00 00 00 00 00 00  00 00 00 40 00 00 00 40   ........ ...@...@
60  00 00 00 73 a8 02 00 00  64 00 00 64 01 00 6c 00   ...s.... d..d..l.
70  00 5a 00 00 64 00 00 64  01 00 6c 01 00 5a 01 00   .Z..d..d ..l..Z..
80  64 02 00 64 03 00 64 04  00 64 05 00 64 06 00 64   d..d..d. .d..d..d
90  07 00 64 08 00 64 09 00  64 0a 00 64 0b 00 64 0c   ..d..d.. d..d..d.
a0  00 64 0d 00 64 0e 00 64  0f 00 64 10 00 64 11 00   .d..d..d ..d..d..
b0  64 12 00 64 13 00 64 14  00 64 15 00 64 16 00 64   d..d..d. .d..d..d
c0  17 00 64 18 00 64 19 00  64 1a 00 64 1b 00 64 1c   ..d..d.. d..d..d.
d0  00 64 1d 00 64 1e 00 64  1f 00 64 20 00 64 21 00   .d..d..d ..d .d!.
e0  64 22 00 64 23 00 64 24  00 64 25 00 64 26 00 64   d".d#.d$ .d%.d&.d
f0  27 00 64 28 00 64 29 00  64 2a 00 64 2b 00 64 2c   '.d(.d). d*.d+.d,
```

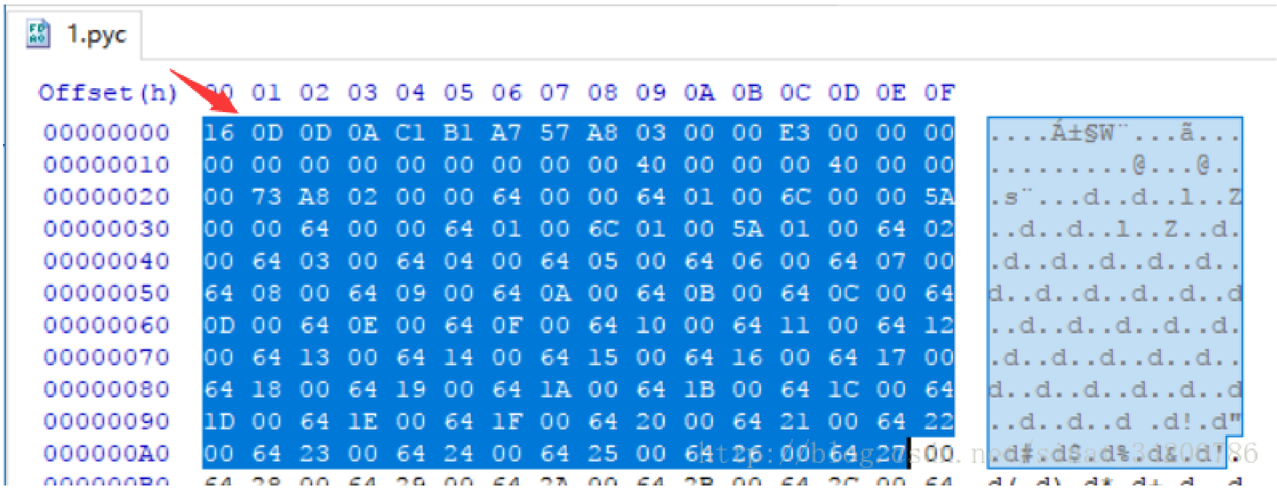
```
然后我们在通信的最后可以发现通信方发送了一个字符串
'Wmkvw680HDzDqMK6UBXChDXCtC7CosKmw7R9w7JLwr/CoT44UcKNwp7DllpPwo3DtsOID8OPTcOWwrzDpi3CtMOKw4PColrCpXUYRh
```



```
52 107.978385597 192.168.1.149      176.31.102.84     TCP         66 58558→6667
53 123.120288443 192.168.1.149      176.31.102.84     IRC        224 Request (P
54 123.216982291 176.31.102.84      192.168.1.149     TCP         66 6667→58558

▾ Request: PRIVMSG Cold_Storm :Wmkvw680HDzDqMK6UBXChDXCtC7CosKmw7R9w7JLwr/CoT44UcKNwp7
    Command: PRIVMSG
  ▾ Command parameters
      Parameter: Cold_Storm
    Trailer: Wmkvw680HDzDqMK6UBXChDXCtC7CosKmw7R9w7JLwr/CoT44UcKNwp7DllpPwo3DtsOID8OPT

0020  66 54 e4 be 1a 0b e9 69  89 8c 1a 98 62 9b 80 18   fT.....i ....b...
0030  01 2b dc ae 00 00 01 01  08 0a 04 9b a8 9d 40 87   .+...... ......@.
0040  bd d4 50 52 49 56 4d 53  47 20 43 6f 6c 64 5f 53   ..PRIVMS G Cold_S
0050  74 6f 72 6d 20 3a 57 6d  6b 76 77 36 38 30 48 44   torm :Wm kvw680HD
0060  7a 44 71 4d 4b 36 55 42  58 43 68 44 58 43 74 43   zDqMK6UB XChDXCtC
0070  37 43 6f 73 4b 6d 77 37  52 39 77 37 4a 4c 77 72   7CosKmw7 R9w7JLwr
0080  2f 43 6f 54 34 34 55 63  4b 4e 77 70 37 44 6c 6c   /CoT44Uc KNwp7Dll
0090  70 50 77 6f 33 44 74 73  4f 49 44 38 4f 50 54 63   pPwo3Dts OID8OPTc
00a0  4f 57 77 72 7a 44 70 69  33 43 74 4d 4f 4b 77 34   OWwrzDpi 3CtMOKw4
00b0  50 43 6f 6c 72 43 70 58  55 59 52 68 58 43 68 4d   PColrCpX UYRhXChM
00c0  4b 39 77 36 50 44 68 78  66 44 69 63 4f 64 77 6f   K9w6PDhx fDicOdwo
00d0  41 67 77 70 67 4e 77 35  2f 43 76 77 3d 3d 0d 0a   AgwpgNw5 /Cvw==..

Request trailer (irc.request.trailer), 126 字节
```

Google之后知道 .pyc文件是python文件编译后的文件，那么我们反编译后就可以获得 .py 文件，推测和最后一个字符串的加密有

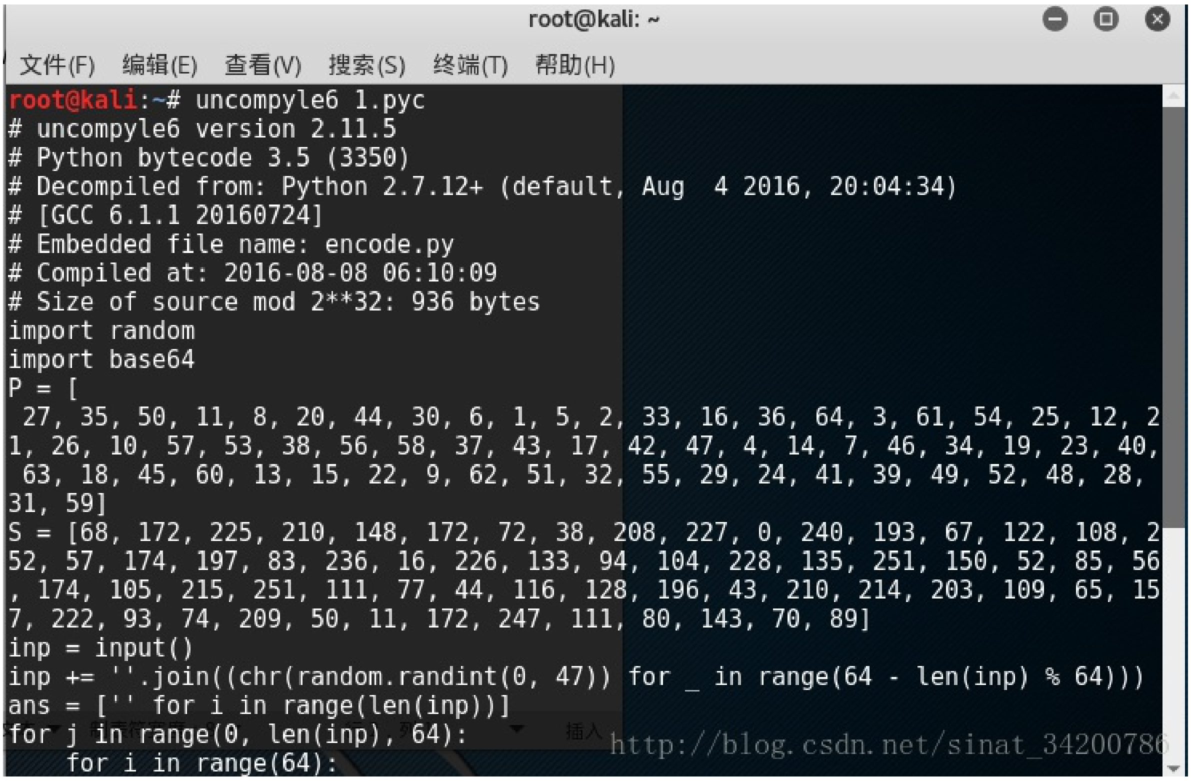
把获取到的文件数据粘贴进HxD中生成 .pyc 文件



```
1.pyc

Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  16 0D 0D 0A C1 B1 A7 57 A8 03 00 00 E3 00 00 00  ....Á±§W¨...ã...
00000010  00 00 00 00 00 00 00 00 00 40 00 00 00 40 00 00  .........@...@..
00000020  00 73 A8 02 00 00 64 00 00 64 01 00 6C 00 00 5A  .s¨...d..d..l..Z
00000030  00 00 64 00 00 64 01 00 6C 01 00 5A 01 00 64 02  ..d..d..l..Z..d.
00000040  00 64 03 00 64 04 00 64 05 00 64 06 00 64 07 00  .d..d..d..d..d..
00000050  64 08 00 64 09 00 64 0A 00 64 0B 00 64 0C 00 64  d..d..d..d..d..d
00000060  0D 00 64 0E 00 64 0F 00 64 10 00 64 11 00 64 12  ..d..d..d..d..d.
00000070  00 64 13 00 64 14 00 64 15 00 64 16 00 64 17 00  .d..d..d..d..d..
00000080  64 18 00 64 19 00 64 1A 00 64 1B 00 64 1C 00 64  d..d..d..d..d..d
00000090  1D 00 64 1E 00 64 1F 00 64 20 00 64 21 00 64 22  ..d..d..d .d!.d"
000000A0  00 64 23 00 64 24 00 64 25 00 64 26 00 64 27 00  .d#.d$.d%.d&.d'.
000000B0  64 28 00 64 29 00 64 2A 00 64 2B 00 64 2C 00 64  d(.d).d*.d+.d,.d
```

接下来进行反编译，网上有在线反编译的，不过对于这个文件来说效果不好，那么就自己反编译，这里用到python的'uncompyle6',

这里顺便提一下文件开头的 16 0D的含义，这是magic number，不同版本编译出来的 .pyc 文件的magic number是不同的，具体
python 3.5b2 这在uncompyle6进行反编译时也会有提示


root@kali: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
http://blog.csdn.net/sinat_34200786

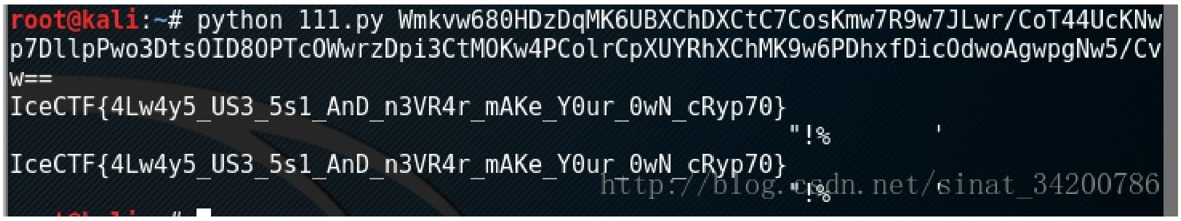
```
root@kali:~# uncompyle6 1.pyc
# uncompyle6 version 2.11.5
# Python bytecode 3.5 (3350)
# Decompiled from: Python 2.7.12+ (default, Aug  4 2016, 20:04:34)
# [GCC 6.1.1 20160724]
# Embedded file name: encode.py
# Compiled at: 2016-08-08 06:10:09
# Size of source mod 2**32: 936 bytes
import random
import base64
P = [
 27, 35, 50, 11, 8, 20, 44, 30, 6, 1, 5, 2, 33, 16, 36, 64, 3, 61, 54, 25, 12, 2
1, 26, 10, 57, 53, 38, 56, 58, 37, 43, 17, 42, 47, 4, 14, 7, 46, 34, 19, 23, 40,
 63, 18, 45, 60, 13, 15, 22, 9, 62, 51, 32, 55, 29, 24, 41, 39, 49, 52, 48, 28,
31, 59]
S = [68, 172, 225, 210, 148, 172, 72, 38, 208, 227, 0, 240, 193, 67, 122, 108, 2
52, 57, 174, 197, 83, 236, 16, 226, 133, 94, 104, 228, 135, 251, 150, 52, 85, 56
, 174, 105, 215, 251, 111, 77, 44, 116, 128, 196, 43, 210, 214, 203, 109, 65, 15
7, 222, 93, 74, 209, 50, 11, 172, 247, 111, 80, 143, 70, 89]
inp = input()
inp += ''.join((chr(random.randint(0, 47)) for _ in range(64 - len(inp) % 64)))
ans = ['' for i in range(len(inp))]
for j in range(0, len(inp), 64):
    for i in range(64):
```

然后我们就得到了加密的 .py文件

```
import random              #加密 .py
import base64

P = [
 27, 35, 50, 11, 8, 20, 44, 30, 6, 1, 5, 2, 33, 16, 36, 64, 3, 61, 54, 25, 12, 21, 26, 10, 57, 53, 38, 56, 58, 3
7, 43, 17, 42, 47, 4, 14, 7, 46, 34, 19, 23, 40, 63, 18, 45, 60, 13, 15, 22, 9, 62, 51, 32, 55, 29, 24, 41, 39,
49, 52, 48, 28, 31, 59]

inp = input()
inp += ''.join((chr(random.randint(0, 47)) for _ in range(64 - len(inp) % 64)))
ans = ['' for i in range(len(inp))]

for j in range(0, len(inp), 64):
    for i in range(64):
        ans[j + P[i] - 1] = chr((ord(inp[j + i]) + S[i]) % 256)

ans = ''.join(ans)
print(base64.b64encode(ans.encode('utf8')).decode('utf8'))
```

分析加密代码后写出解密代码，解密最后一个字符串即可

```
# Offsec Research CTF Team

import random, base64, string, sys

P = [27, 35, 50, 11, 8, 20, 44, 30, 6, 1, 5, 2, 33, 16, 36, 64, 3, 61, 54, 25, 12, 21, 26, 10, 57, 53, 38, 56, 58, 37,
 43, 17, 42, 47, 4, 14, 7, 46, 34, 19, 23, 40, 63, 18, 45, 60, 13, 15, 22, 9, 62, 51, 32, 55, 29, 24, 41, 39, 49,
 52, 48, 28, 31, 59]

S = [68, 172, 225, 210, 148, 172, 72, 38, 208, 227, 0, 240, 193, 67, 122, 108, 252, 57, 174, 197, 83, 236, 16, 226, 133,
 94, 104, 228, 135, 251, 150, 52, 85, 56, 174, 105, 215, 251, 111, 77, 44, 116, 128, 196, 43, 210, 214, 203, 109,
 65, 157, 222, 93, 74, 209, 50, 11, 172, 247, 111, 80, 143, 70, 89]

# comment these lines if not running under python2
reload(sys)
sys.setdefaultencoding('utf8')

# Get the encoded flag and do the conversions in reverse order
ans = ((base64.b64decode(sys.argv[1])).encode('utf8')).decode('utf8')

# Create a list with length of character in ans (encoded flag)
ans_list = list(ans)

# Create empty inp list
inp = ['' for i in range(len(ans))]

for j in range(0, len(ans), 64):
    for i in range(64):
        # Try every printable ascii character and if the equation is satisfied, we've found one character of the initial input
        for c in string.printable:
            if (ans_list[j + P[i] - 1] == unichr(((ord(c) + S[i]) % 256))):
                inp[j + i] = c

inp = ''.join(inp)
print(inp)
```



```
root@kali:~# python 111.py Wmkvw680HDzDqMK6UBXChDXCtC7CosKmw7R9w7JLwr/CoT44UcKNw
p7DllpPwo3DtsOID8OPTcOWwrzDpi3CtMOKw4PColrCpXUYRhXChMK9w6PDhxfDicOdwoAgwpgNw5/Cv
w==
IceCTF{4Lw4y5_US3_5s1_AnD_n3VR4r_mAKe_Y0ur_0wN_cRyp70}
                                                              "!%        '
IceCTF{4Lw4y5_US3_5s1_AnD_n3VR4r_mAKe_Y0ur_0wN_cRyp70}
                                                              "!%
```

涨姿势点

```
       IRC网络
       Wireshark从数据流中提取文件的方式
       不同版本的python编译后的 .pyc文件的magic number不同
       uncomple6反编译 .pyc文件，也有在线反编译的网站
```

备注

```
虽然文件是python 3.5的版本编译的，不过用uncompyle6反编译时的python版本为 2.7
```

---