




第一届BMZCTF公开赛-MISC-Writeup

原创

末初  于 2020-12-28 13:24:47 发布  5270  收藏 11

分类专栏: [CTF_MISC_Writeup](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/mochu7777777/article/details/111306837>

版权



[CTF_MISC_Writeup](#) 专栏收录该内容

246 篇文章 46 订阅

订阅专栏

文章目录

[签到题](#)

[你猜猜flag](#)

[Snake](#)

[Tiga](#)

[Hack K](#)

前言

首先恭喜 [白帽子社区团队](#) 成功举办第一届BMZCTF公开赛, 我是本次比赛MISC赛题 [Snake](#)、[Tiga](#) 的出题人末初

以下是我写的这次BMZCTF公开赛的MISC赛题的Writeup, 如果有什么写的不对的还请师傅们留言斧正

另外师傅们对于 [Snake](#)、[Tiga](#) 赛题有疑问的可以留言, 我会及时回复

签到题

Challenge

417 Solves

×

MISC_签到题

20

关注官方公众号:白帽子社区回复: 2020BMZCTF

Flag

Submit

<https://blog.csdn.net/mochu7777777>

2020BMZCTF



BMZCTF{W3lc0me_T0_2020BMZCTF}

BMZCTF{W3lc0me_T0_2020BMZCTF}

你猜猜flag

Challenge

116 Solves



MISC_你猜猜flag

30

flag.exe

Flag

Submit

<https://blog.csdn.net/mochu7777777>

flag.exe, binwalk 分析发现有附加zip数据, foremost 分离出来

```

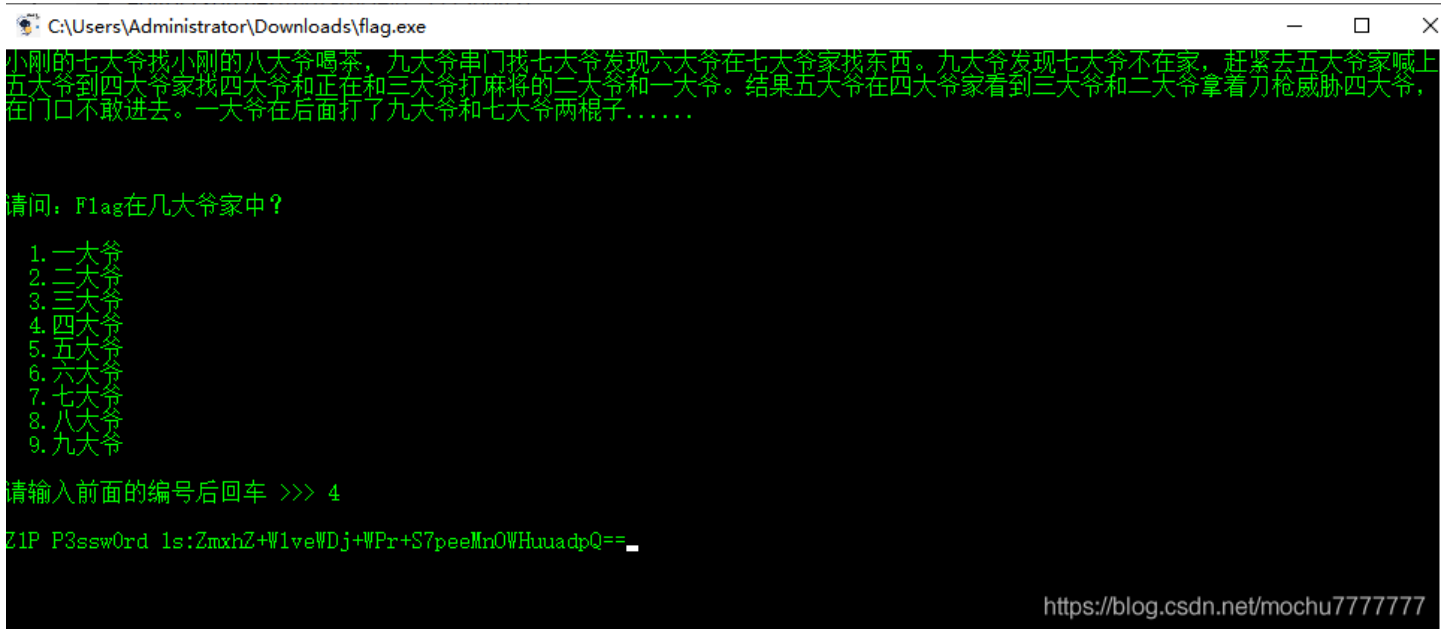
root@mochu7-pc:/mnt/c/Users/Administrator/Downloads# binwalk flag.exe
DECIMAL      HEXADECIMAL     DESCRIPTION
-----
0             0x0             Microsoft executable, portable (PE)
522808       0x7FA38        CRC32 polynomial table, little endian
525799       0x805E7        Copyright string: "Copyright 1995-1998 Jean-loup Gailly "
527011       0x80AA3        Copyright string: "Copyright 1995-1998 Mark Adler "
620392       0x97768        Copyright string: "copyright violation: edited ICC profile ignored"
687804       0xA7E8C        XML document, version: "1.0"
692271       0xA902F        Zip archive data, encrypted at least v2.0 to extract, compressed size: 779, uncompressed size: 1206, name: Flag.txt
693088       0xA9360        Zip archive data, encrypted at least v2.0 to extract, compressed size: 13787, uncompressed size: 249856, name: Misc So easy.mdb
706921       0xAC969        Zip archive data, encrypted at least v2.0 to extract, compressed size: 25, uncompressed size: 13, name: Can you guess flag.txt
707354       0xACB1A        Zip archive data, encrypted at least v2.0 to extract, compressed size: 779, uncompressed size: 1206, name: Flag.txt
708171       0xACE4B        Zip archive data, encrypted at least v2.0 to extract, compressed size: 13787, uncompressed size: 249856, name: Misc So easy.mdb
722004       0xB0454        Zip archive data, encrypted at least v2.0 to extract, compressed size: 25, uncompressed size: 13, name: Can you guess flag.txt

root@mochu7-pc:/mnt/c/Users/Administrator/Downloads# foremost flag.exe
Processing: flag.exe
|foundat=Flag.txt*)
SDHI#F#(*#~Dr#C#?#D#Dr.#5#D#w#3#A#p#w#E#D:##2#A#0#"]N#e#e#e#e#M#e#e#DmR;#D.#.#,Co#L#lFX#
#.#.#T#f#o#k# aa_#M#<#L:#$#o#g#o#O#e#e#e#e#G#
#D#e#M*L#A#>#.#f#b#B#/#B#D#61#e#k#e#Q#H#D#X^'##=D#('t#e#d#D#Z#e#D
0"#eo'2v#e#e#D#e#q6!)O#u*#D {#D#e#[[R#D\I#g'#e#D#~#e#D#H#D#E#L#e#D#b#B{#C#<#e#e#e#e#k#I#e#.#*#n#O#e#d#e#e#F#_#.#.#.#.#
foundat=Can you guess flag.txt#
#e#Yp;#.N#O#e#r/#q;#e#Y#D#W#G#P#K#D#?
foundat=Flag.txt*)
SDHI#F#(*#~Dr#C#?#D#Dr.#5#D#w#3#A#p#w#E#D:##2#A#0#"]N#e#e#e#e#M#e#e#DmR;#D.#.#,Co#L#lFX#
#.#.#T#f#o#k# aa_#M#<#L:#$#o#g#o#O#e#e#e#e#G#
#D#e#M*L#A#>#.#f#b#B#/#B#D#61#e#k#e#Q#H#D#X^'##=D#('t#e#d#D#Z#e#D
0"#eo'2v#e#e#D#e#q6!)O#u*#D {#D#e#[[R#D\I#g'#e#D#~#e#D#H#D#E#L#e#D#b#B{#C#<#e#e#e#e#k#I#e#.#*#n#O#e#d#e#e#F#_#.#.#.#.#
foundat=Can you guess flag.txt#
#e#Yp;#.N#O#e#r/#q;#e#Y#D#W#G#P#K#D#?
*)
root@mochu7-pc:/mnt/c/Users/Administrator/Downloads# tree output/
output/
├── audit.txt
├── exe
│   └── 00000000.exe
└── zip
    ├── 00001352.zip
    └── 00001381.zip

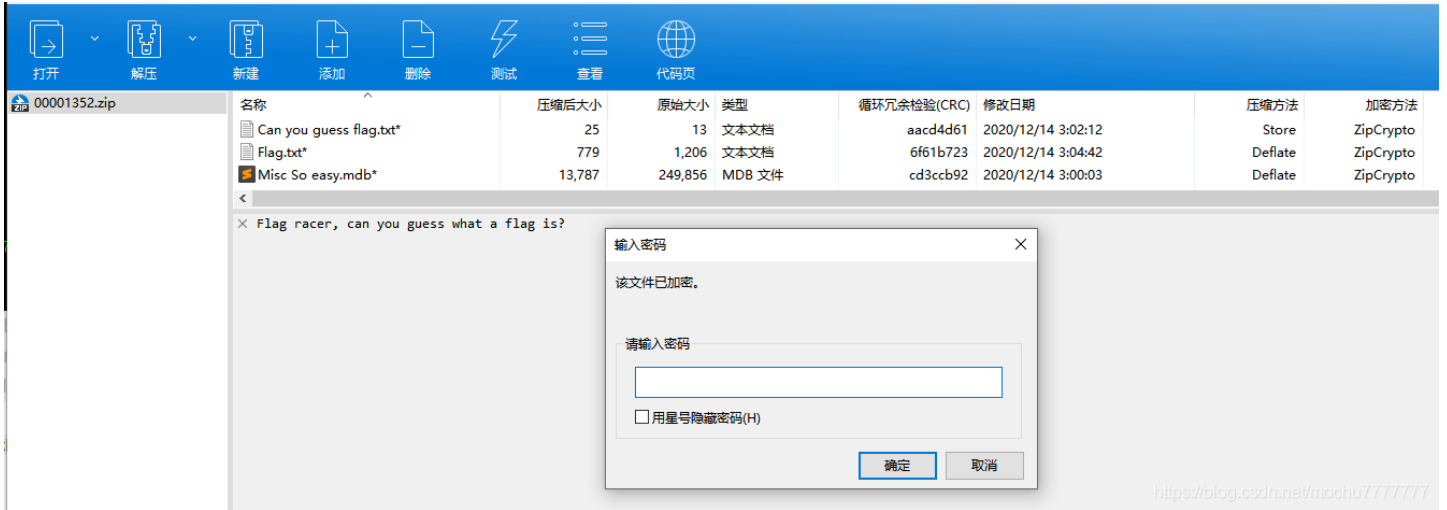
2 directories, 4 files
root@mochu7-pc:/mnt/c/Users/Administrator/Downloads#

```

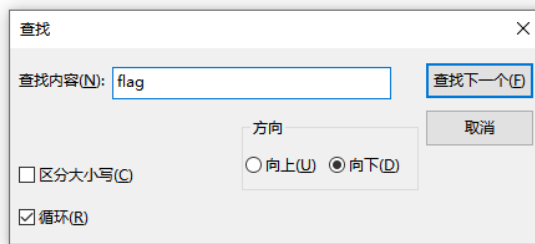
flag.exe 只有输入 4 的时候输出的zip密码与其他的号码不同



分离出来的zip压缩包密码为: `ZmxhZ+W1veWDj+WPr+S7peeMnOwHuuadpQ==`



解压，再 **mdb** 文件中找到flag



flag{D1d y0u 8u3ss?} The rose and the gun}O 7

flag{D1d y0u 8u3ss?}

Snake

Challenge

6 Solves



MISC_Snake

96

斯内克~斯内克~斯内克~斯内克~斯内克~斯内克~斯内克~

 snake.zip

Flag

Submit

<https://blog.csdn.net/mochu7777777>

难度：中下

考察知识点：Short Ook!、程序逆向还原、Steghide、Serpent加密、二进制数据转二维码


```

data_jpg = open('data.jpg', 'wb')
def file_encode():
    with open('snake.jpg', 'rb') as handle:
        i = 1
        while True:
            bytedata = handle.read(1)
            if(bytedata == b''):
                exit()
            process_data = data_encode(bytedata)
            data_write(process_data)
            i = i + 1

def data_encode(bytedata):
    data = int.from_bytes(bytedata, byteorder='big')
    if (data % 2 == 0):
        data = (data + 1) ^ 128
    else:
        data = (data - 1) ^ 128
    data = bytes([data])
    return data

def data_write(process_data):
    data_jpg.write(process_data)

if __name__ == '__main__':
    file_encode()
    data_jpg.close()

```

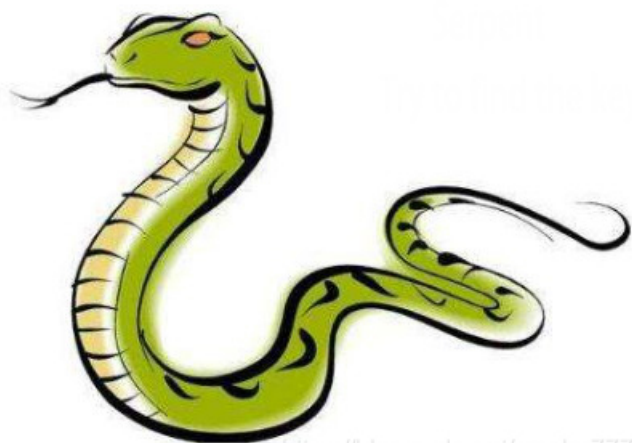
通过程序可以得知 `data.jpg` 是 `process.py` 处理 `snake.jpg` 得到的，逆向编写处理代码

```

with open('snake.jpg', 'wb') as flag:
    with open('data.jpg', 'rb') as f:
        for i in f.read():
            if (i % 2 == 0):
                i = (i + 1) ^ 128
            else:
                i = (i - 1) ^ 128
            i = bytes([i])
            flag.write(i)

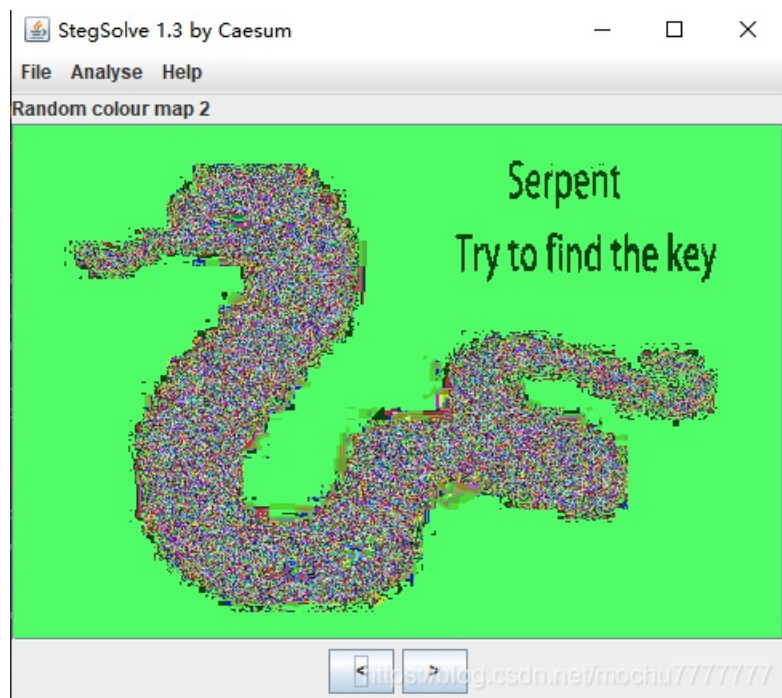
```

得到 `snake.jpg`



<https://blog.csdn.net/mochu7777777>

使用 `stegsolve` 打开，发现提示



`Serpent` 加密，尝试寻找 `key`

Steghide 发现有隐写 key.txt，无密码

```
root@mochu7-pc: /mnt/c/Users/Administrator/Desktop/新建文件夹/新建文件夹/snake# steghide info snake.jpg
"snake.jpg":
  format: jpeg
  capacity: 1.8 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "key.txt":
    size: 15.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes
root@mochu7-pc: /mnt/c/Users/Administrator/Desktop/新建文件夹/新建文件夹/snake# steghide extract -sf snake.jpg
Enter passphrase:
wrote extracted data to "key.txt".
root@mochu7-pc: /mnt/c/Users/Administrator/Desktop/新建文件夹/新建文件夹/snake# ls
data data.jpg key.txt proces.py reconvert.py snake.jpg
root@mochu7-pc: /mnt/c/Users/Administrator/Desktop/新建文件夹/新建文件夹/snake# cat key.txt
key: VivaLaVida
root@mochu7-pc: /mnt/c/Users/Administrator/Desktop/新建文件夹/新建文件夹/snake#
```

key: VivaLaVida

知道了是 Serpent 加密，也知道了 key，那么 data 应该就是加密后的数据，直接解密

Serpent-Online-Encrypt: <http://serpent.online-domain-tools.com/>

Input type: File

File: C:\fakepath\data Browse

Function: SERPENT

Mode: ECB (electronic codebook)

Key: VivaLaVida (plain)

Plaintext Hex

> Encrypt! > Decrypt! ▶ 🔗

100% File was uploaded.

Decrypted text:

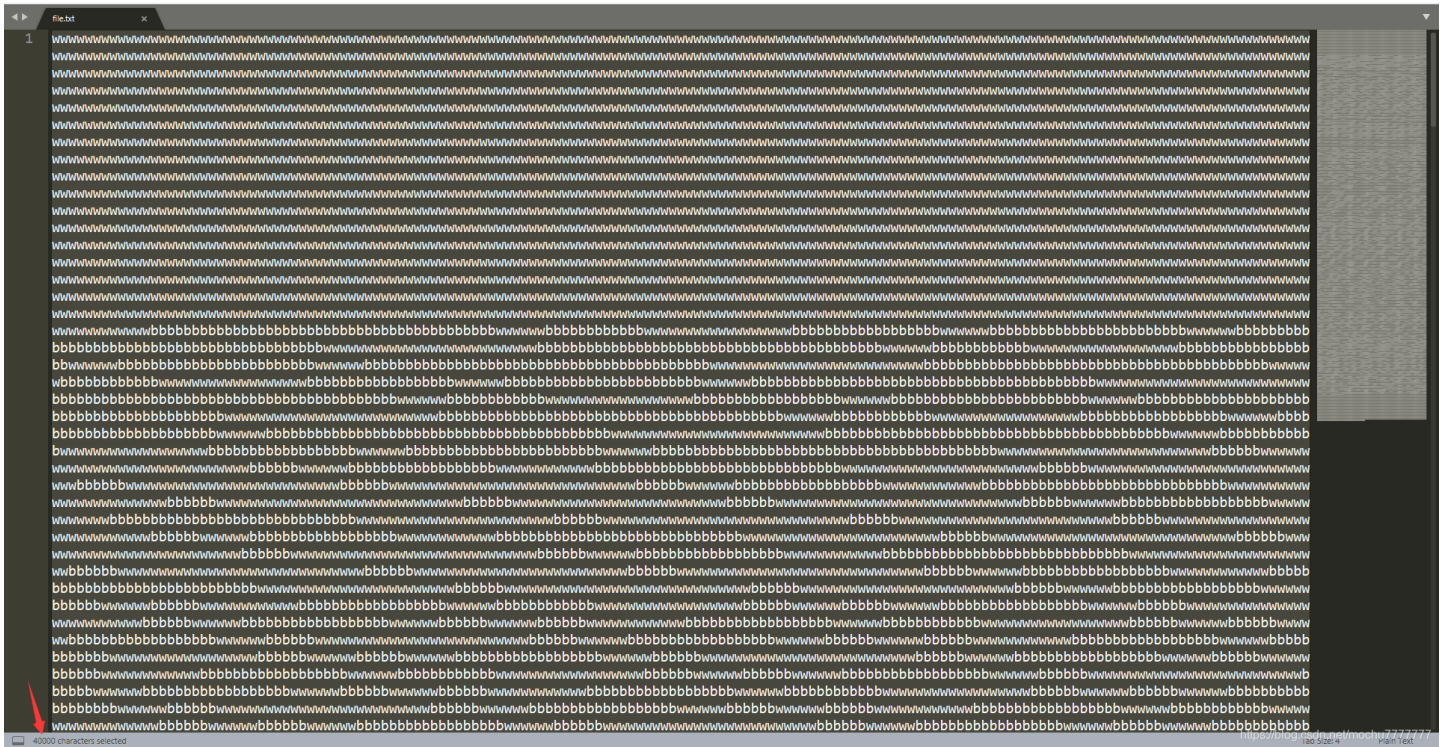
⚠ Displayed output of your task is restricted to the first 16 kB of data. You can get the full result using the *Download as a binary file link*.

00000000	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000010	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000020	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000030	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000040	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000050	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000060	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000070	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000080	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w
00000090	77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	w w w w w w w w w w w w w w w w

[\[Download as a binary file\]](#) [\[Show more\]](#) [\[Show all\]](#) [\[?\]](#)

<https://blog.csdn.net/mochu777777> Inactive

下载解密后的数据，发现内容如下：



然后从以下几点看出这是个二维码的二进制数据：

- 只有 **w** 和 **b** 两种字符，二进制
- **w** 对应 **white**，**b** 对应 **black**
- 总字符 $200 \times 200 = 40000$ 正方形，可能是张二维码
- 做过二进制转二维码题目的同学从内容的首尾都是 **w** 字符，就应该看得出这是个二维码数据

使用Python将这些转换为二维码

```
import PIL
from PIL import Image

width=height=200
img = Image.new("RGB",(width,height))
i = 0
char = "这里填数据"
for w in range(width):
    for h in range(height):
        if (char[i]=='w'):
            img.putpixel([w,h],(255,255,255))
        else:
            img.putpixel([w,h],(0,0,0))
        i = i + 1
img.save('flag.png')
```

扫描即可得到flag



flag{67bd09fc-e252-4c21-858f-2a7d698d555f}

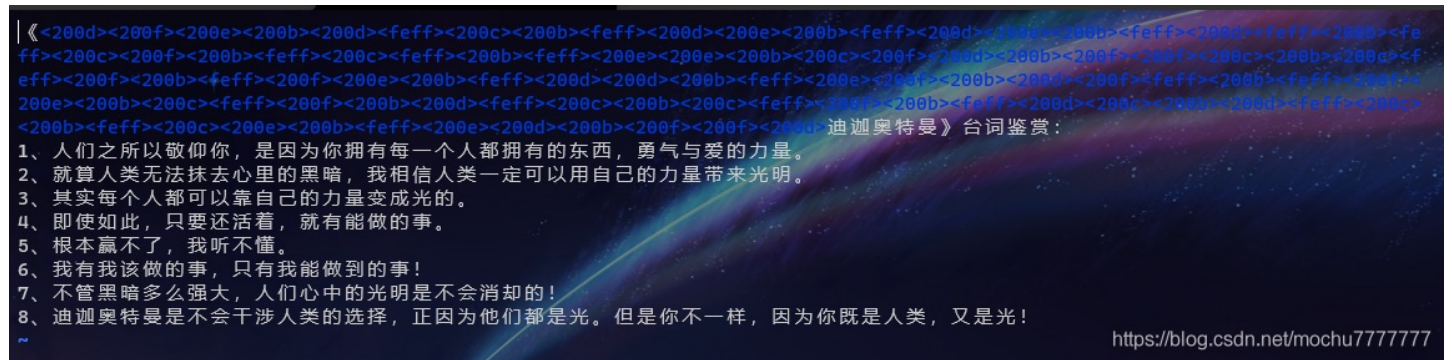
Tiga



难度：中下

考察知识点：零宽度字符隐写、CRC爆破、zip密码爆破、zip密码明文攻击、字节流数据还原、Base16/32/64/85混淆解码

vim查看 `tiga.txt` 发现存在零宽度字符

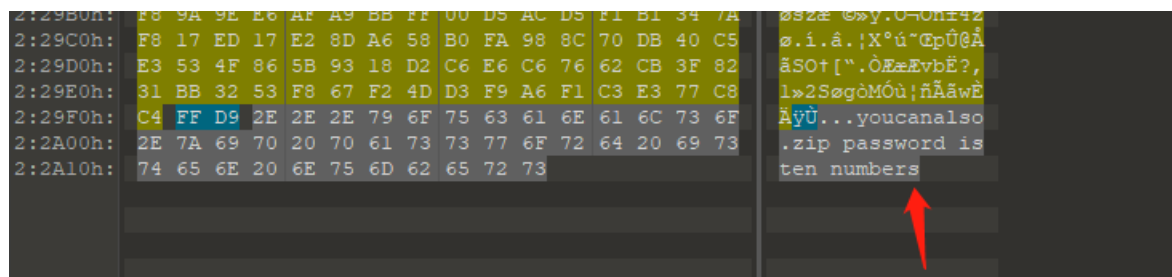


零宽度字符隐写在网站：<https://yuanfux.github.io/zero-width-web/>



得到 `misc.zip` 密码：`GiveTiGaGuang!`

`迪迦.jpg` 文件尾附加了 `youcanalso.zip` 的密码信息



`file.zip` 的密码很明显需要通过爆破这些密码片段文件的CRC得到内容

名称	压缩后大小	原始大小	类型	循环冗余检验(CRC)	修改日期
password					
password1.txt*	15	3	TXT 文件	14433530	2020/12/13 13:57:41
password2.txt*	15	3	TXT 文件	af251007	2020/12/13 13:58:04
password3.txt*	15	3	TXT 文件	d554e7b6	2020/12/13 13:58:18
password4.txt*	15	3	TXT 文件	0ebb3156	2020/12/13 13:58:32
password5.txt*	15	3	TXT 文件	bb474d49	2020/12/13 13:58:48
password6.txt*	15	3	TXT 文件	2cb8a39b	2020/12/13 13:59:02
password7.txt*	15	3	TXT 文件	75fe76f0	2020/12/13 13:59:26

爆破脚本如下：

```
import binascii
import string

def crack_crc():
    print('-----Start Crack CRC-----')
    crc_list = [0x14433530, 0xaf251007, 0xd554e7b6, 0xebb3156, 0xbb474d49, 0x2cb8a39b, 0x75fe76f0]
    comment = ''
    chars = string.printable
    for crc_value in crc_list:
        for char1 in chars:
            for char2 in chars:
                for char3 in chars:
                    res_char = char1 + char2 + char3
                    char_crc = binascii.crc32(res_char.encode())
                    calc_crc = char_crc & 0xffffffff
                    if calc_crc == crc_value:
                        print('[+] {}: {}'.format(hex(crc_value), res_char))
                        comment += res_char
    print('-----CRC Crack Completed-----')
    print('Result: {}'.format(comment))

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



```

1  import binascii
2  import string
3
4  def crack_crc():
5      print('-----Start Crack CRC-----')
6      crc_list = [0x14433530, 0xaf251007, 0xd554e7b6, 0xebb3156, 0xbb474d49, 0x2cb8a39b, 0x75fe76f0]
7      comment = ''
8      chars = string.printable
9      for crc_value in crc_list:
10         for char1 in chars:
11             for char2 in chars:
12                 for char3 in chars:
13                     res_char = char1 + char2 + char3
14                     char_crc = binascii.crc32(res_char.encode())
15                     calc_crc = char_crc & 0xffffffff
16                     if calc_crc == crc_value:
17                         print('[+] {}: {}'.format(hex(crc_value), res_char))
18                         comment += res_char
19     print('-----CRC Crack Completed-----')
20     print('Result: {}'.format(comment))
21
22 if __name__ == '__main__':
23     crack_crc()
24

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

[Running] D:/Python/Python3/python.exe "c:\Users\Administrator\Desktop\新建文件夹\tiga\misc\code.py"

-----Start Crack CRC-----

[+] 0x14433530: T&h

[+] 0xaf251007: g%W

[+] 0xd554e7b6: L0^

[+] 0xebb3156: rm@

[+] 0xbb474d49: c!V

[+] 0x2cb8a39b: K\$x

[+] 0x75fe76f0: Et~

-----CRC Crack Completed-----

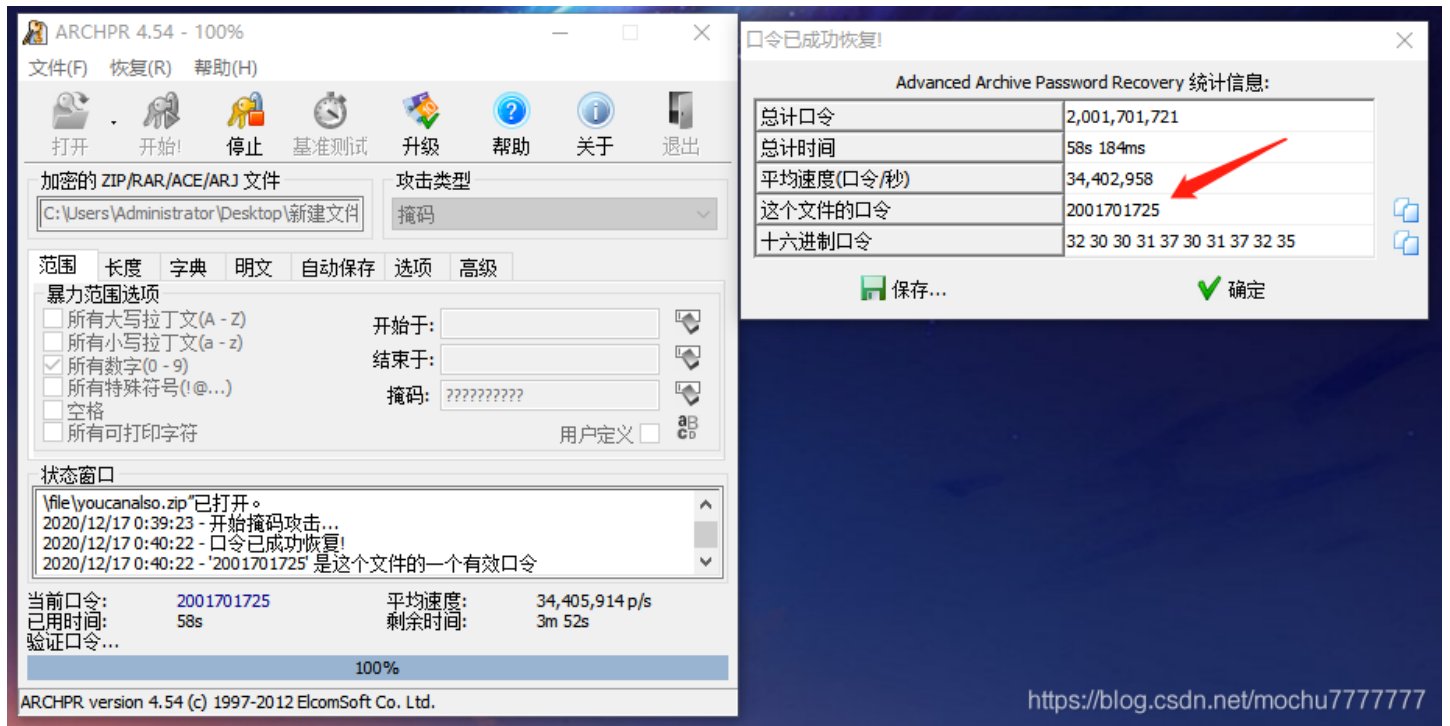
Result: T&hg%WL0^rm@c!VK\$xEt~

[Done] exited with code=0 in 2.703 seconds

<https://blog.csdn.net/mochu7777777>

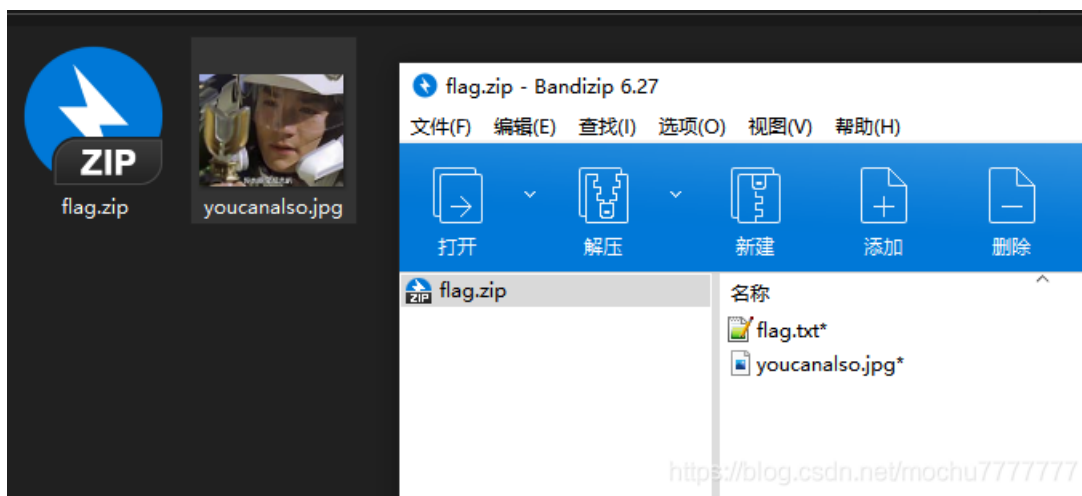
得到 file.zip 密码: T&hg%WL0^rm@c!VK\$xEt~

解压得到 `youcanalso.zip` 根据之前的得到的信息，使用ARCHPR掩码爆破十位数字



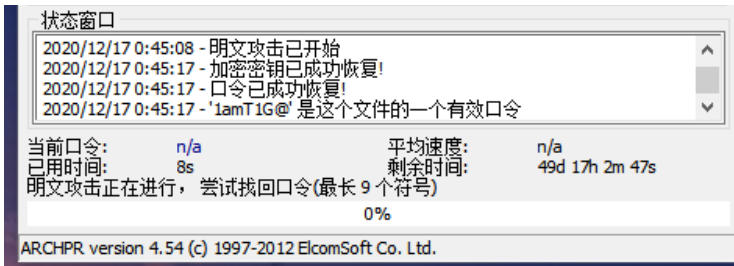
得到密码: `2001701725`

解压得到 `flag.zip` 和 `youcanalso.zip`

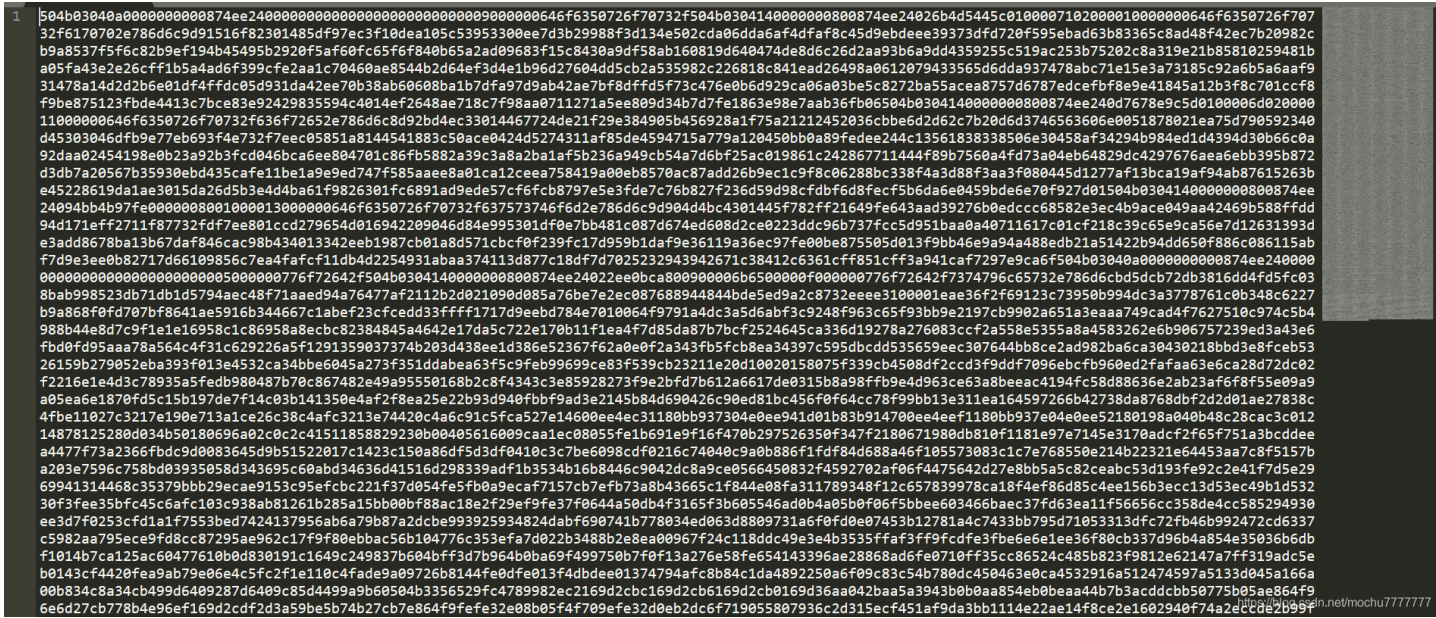


这里很明显可以进行明文攻击





得到 flag.zip 密码: 1amT1G@ 得到 flag.txt, 打开发现通过观察头和尾的内容是 zip 文件的字节流数据



将字节流写成 zip 文件, 脚本如下:

```
import struct

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

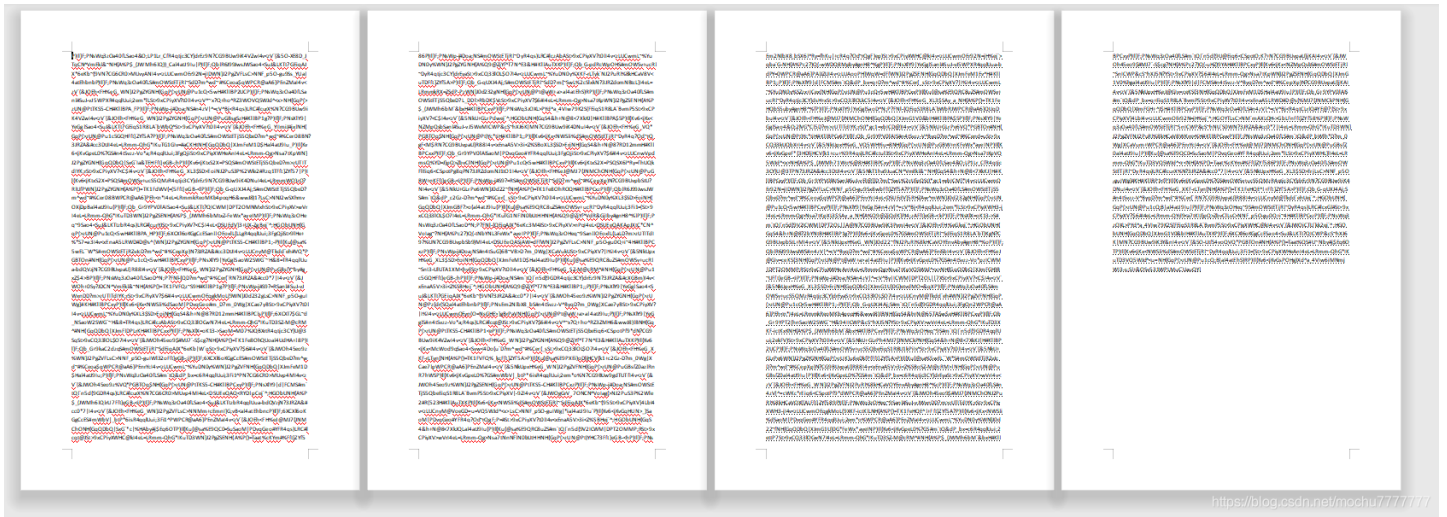
with open("data.zip", "wb") as f:
    for i in res:
        s = struct.pack('B', int(i, 16))
        f.write(s)
```

得到 data.zip 发现是 doc 文件的内容

名称	压缩后大小	原始大小	类型	循环冗余检验(CRC)	修改日期
._rels	253	737			2012/7/2 9:52:14
customXml	566	881			2012/7/2 9:52:14
docProps	951	1,630			2012/7/2 9:52:14
word	11,362	54,089			2012/7/2 9:52:14
[Content_Types].xml	363	1,432	XML 文档	daffb6d4	2012/7/2 9:52:14

修改后缀为得到 data.docx, 打开发现是 base85 数据

注意：后两页是隐藏文字，需要开启隐藏文字查看选项才能发现



将这个 **base5** 数据解了一下发现是：base16/32/64/85的随机套娃编码

这里可以通过写个识别脚本通过识别base16/32/64/85进行逐步解码，也可以手工，因为这里设置的编码次数不多，就15次

PS：本来是想设置的编码次数多一点的，不过我比较懒，不想写识别脚本2333

不过推荐使用 **basecrack** 这个工具直接解码最快：<https://github.com/mufeedvh/basecrack>

```
python .\basecrack.py --magic
```

然后输入编码数据，得到flag

```
Windows PowerShell

[-] Decoding as Base16: KJ4HMTJJKA6GERD6EY7XZZZFRN2WWWBZRBDEQ232RFUDQULEEM2S2TLKFCU66R2KJQVE4LLKA6G1IJ3KIST4JL6J5DWWST3
KZHU2QRIRKAVHC6CPKN2XGTCXJ4CSZTEKJBE84CDKZFVSSCK
{{(=====)}}

[-] Iteration: 12

[-] Heuristic Found Encoding To Be: Base32

[-] Decoding as Base32: Rxya)P<eD~V7]g%$ukX9PFHkzQh9QdS5-MkQE0yZRaRqkP<d1;Rk%K^OGkJ{VOMB(P*qx0S0uLVRx)rdRBBpCVKTHJ
{{(=====)}}

[-] Iteration: 13

[-] Heuristic Found Encoding To Be: Base85

[-] Decoding as Base85: V143PytkcEtyY0dpNWtxRyVZU9HQkdWUVctPytkVkstEGxfawhwFUZmF1BSV212TjZHa3A=
{{(=====)}}

[-] Iteration: 14

[-] Heuristic Found Encoding To Be: Base64

[-] Decoding as Base64: W^77+dpkKrcG15kqGNYOGBGVQW-?(dVK-(1Eihp=FrcPRWivN6Gkp
{{(=====)}}

[-] Iteration: 15

[-] Heuristic Found Encoding To Be: Base85

[-] Decoding as Base85: flag{8fa3e8c4-0121-4f2a-a7f0-0a60032e3763}
{{(=====)}}

[-] Total Iterations: 15

[-] Encoding Pattern: Base85 -> Base85 -> Base85 -> Base32 -> Base85 -> Base16 -> Base32 -> Base32 -> Base64 -> Base32 ->
Base16 -> Base32 -> Base85 -> Base64 -> Base85

[-] Magic Decode Finished With Result: flag{8fa3e8c4-0121-4f2a-a7f0-0a60032e3763}

[-] Finished in 4.1051 seconds https://blog.csdn.net/mochu777777
```

```
flag{8fa3e8c4-0121-4f2a-a7f0-0a60032e3763}
```

Hack K

Challenge

0 Solves

×

MISC_Hack K

100

Instance Info

Remaining Time: 3232s

<http://www.bmzclub.cn:20351>

Destroy this instance

Renew this instance

Flag

Submit

<https://blog.csdn.net/mochu7777777>

这题其实我也和出题的师傅商量过

flag最后的存放文件名 `flag.php` 太过平常(应该改复杂点), 用平常做web的思路, 目录扫描一扫就出

域名:

线程: (条 CPU核心 * 5最佳) DIR: 1156 ASFX: 822 探测200
 ASP: 1854 PHP: 1067 探测403
 超时: (秒 超时的页面被丢弃) MDB: 419 JSP: 631 探测3XX

扫描信息: 扫描完成... 扫描线程: 0 扫描速度: 0/秒

ID	地址	HTTP响应
1	http://www.bmzclub.cn:20351/flag.php	200
2	http://www.bmzclub.cn:20351/index.html	200

<https://blog.csdn.net/mochu777777>

访问即可得到flag



姓名: K

身份: 黑客 | 强项: 电子取证、无线安全测试、逆向工程等

个性签名: 永恒的面貌是自己的, 我们可以尝试伪装。

他人评价: 神出鬼没, 莫名其妙。



GIVE YOU FLAG:

Flag: flag{Zme12a9rqsK123S14RH}

2020 Copyrights - All Rights Reserved

M - CTF

Encryption Encoding SQL XSS Other

Lead URL

Split URL

Execute Post data Referer User Agent Cookies

<https://blog.csdn.net/mochu777777>

首先观察了这个网站, 唯一可以想到出题方向的就是这几张图片了, 下载下来发现其中一张图片容量非常大

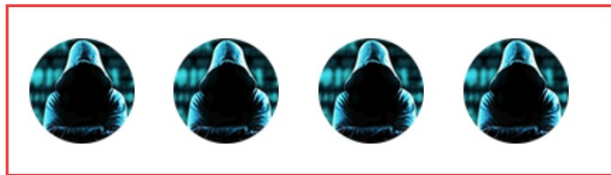


姓名: K

身份: 黑客 | 强项: 电子取证、无线安全测试、逆向工程等

个性签名: 永恒的面貌是自己的, 我们可以尝试伪装。

他人评价: 神出鬼没, 莫名其妙。



他的经历

父亲: 乔詹姆斯

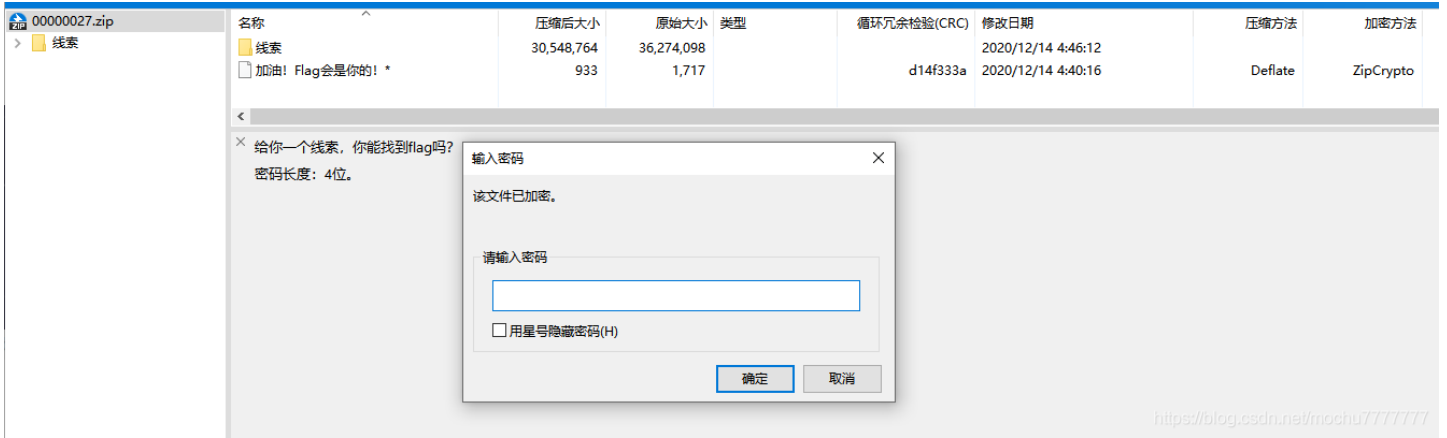
<https://blog.csdn.net/mochu7777777>

binwalk 分析图片有附加别的数据

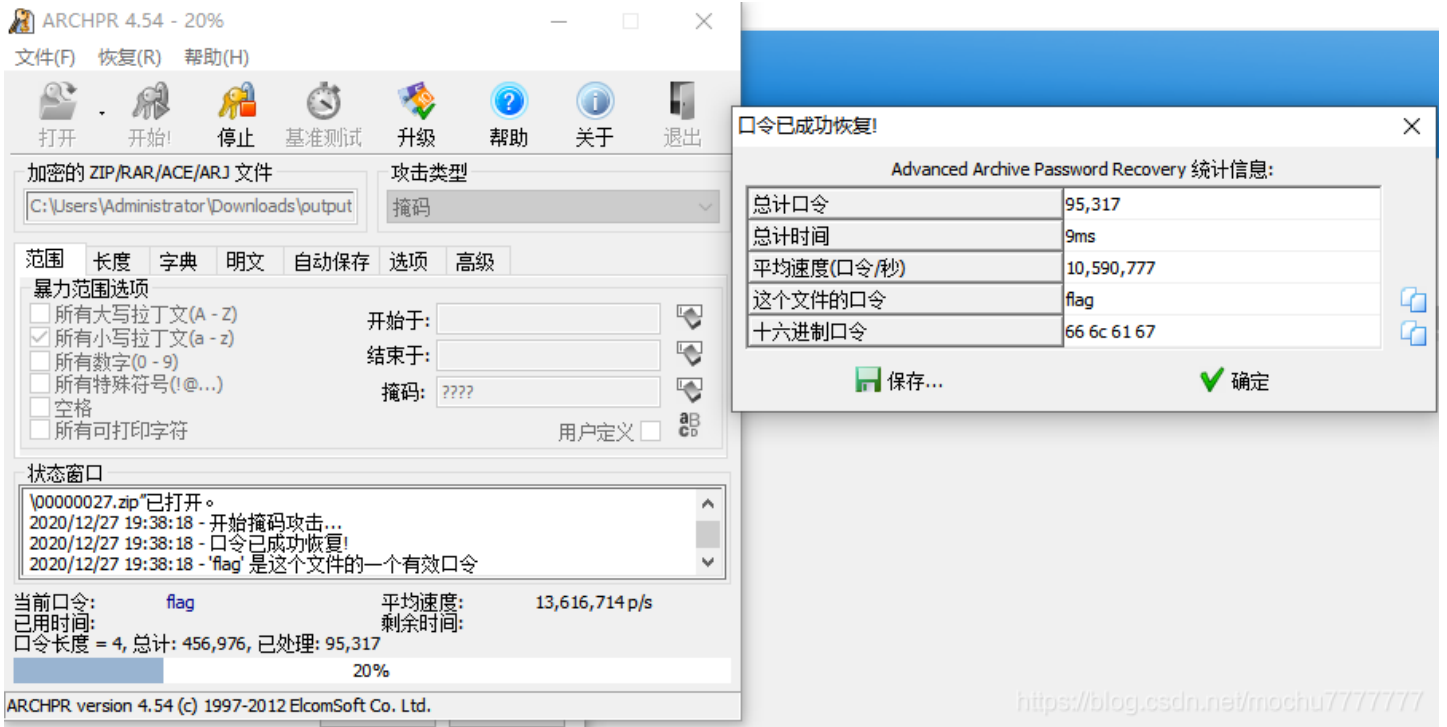
```
root@mochu7-pc:/mnt/c/Users/Administrator/Downloads# binwalk social.png
DECIMAL      HEXADECIMAL  DESCRIPTION
-----
0            0x0          PNG image, 74 x 74, 8-bit/color RGBA, non-interlaced
1594        0x63A       Zlib compressed data, default compression
14823291    0xE22F7B    COBALT boot rom data (Flat boot rom or file system)

root@mochu7-pc:/mnt/c/Users/Administrator/Downloads# |
```

foremost 直接分离, 得到一个压缩包, 有密码



根据压缩包注释提示使用 **ARCHPR** 爆破，得到密码 **flag**

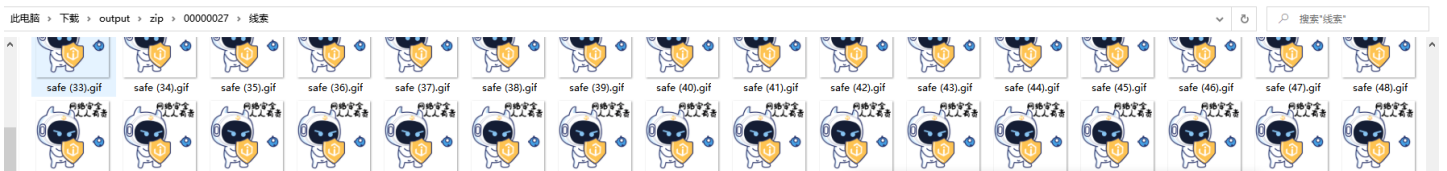


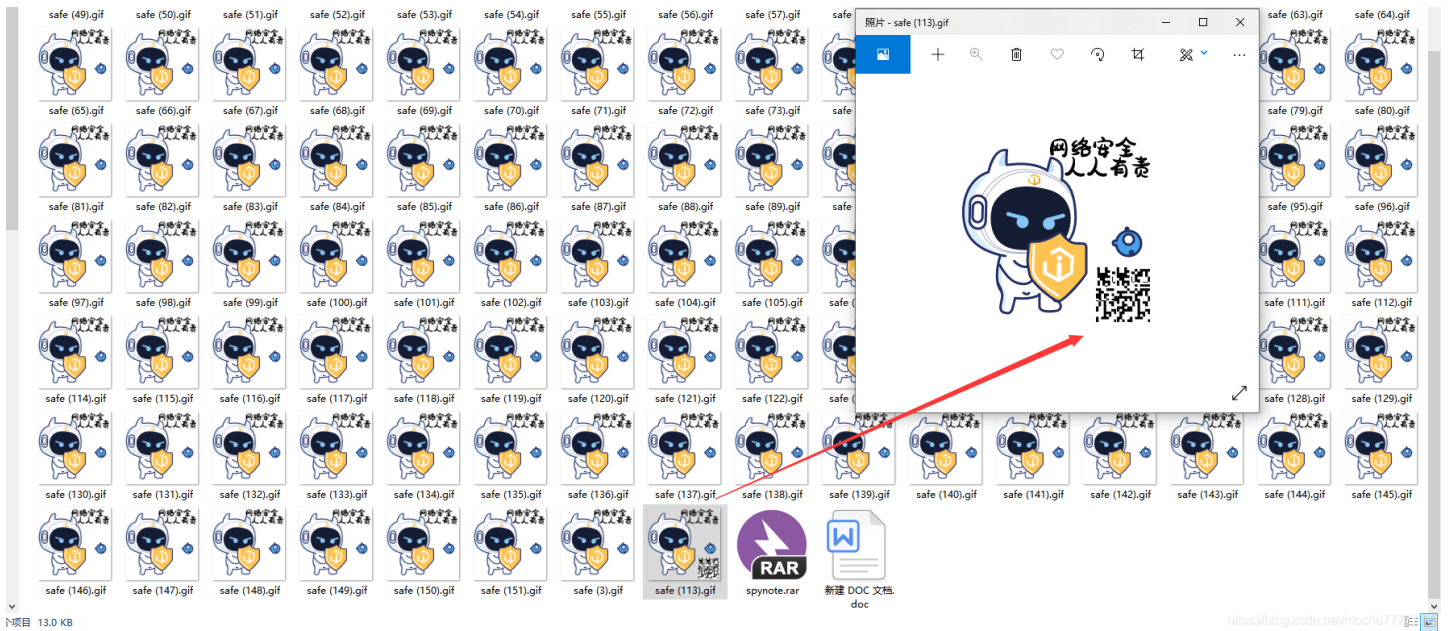
得到一个好像是二维码定位符的png图片

此电脑 > 下载 > output > zip > 00000027



以及一大堆gif图片和文件，其中很容易发现 **safe (113).gif** 中有张好像没有定位符的二维码

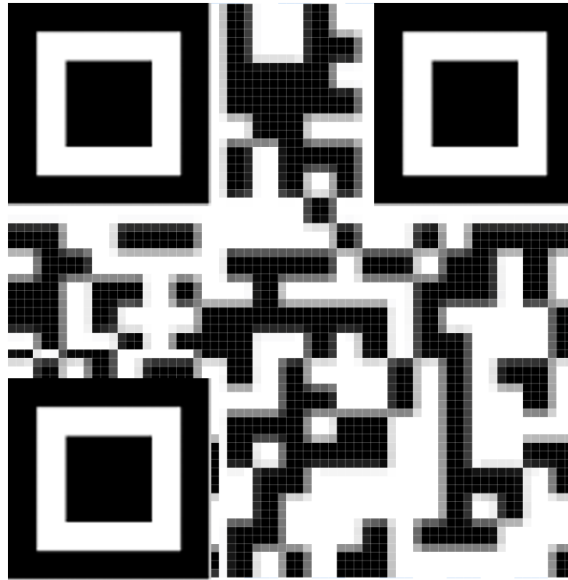




二维码取出来



使用 PS 修补一下



访问 `flag.php` 得到flag



姓名: K

身份: 黑客 | 强项: 电子取证、无线安全测试、逆向工程等

个性签名: 永恒的面貌是自己的, 我们可以尝试伪装。

他人评价: 神出鬼没, 莫名其妙。

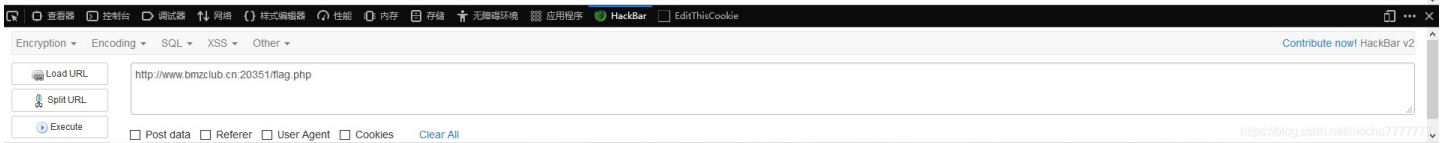


GIVE YOU FLAG:

Flag: flag{Zme12a9rqsK123S14RH}

2020 Copyrights - All Rights Reserved

M - CTF



flag{Zme12a9rqsK123S14RH}



[创作打卡挑战赛](#)

[赢取流量/现金/CSDN周边激励大奖](#)