
杂项题目练习（六）

杂项题目练习（六）	1
杂项第四十一题: 远控木马	4
杂项第四十二题: Web 漏洞	4
bugku-ctf 第四十三题: 颜文字	5
杂项第四十四题: 磁盘镜像	6
杂项第四十五题: 神奇的照片	6
杂项第四十六题: 怀疑人生	8
杂项第四十七-CTF 加密篇之 ok (Ook!)	15
杂项第四十八题: 红绿灯	17
杂项第四十九题: 不简单的压缩包	22

前言：

以下是我在 bugku 练习的解题思路，编号跟我前面分享的基础是对应的，理论基础结合实践。

所有题目目录如下：

题目练习	1
杂项第一题: 签到题	3
杂项第二题: 这是一张单纯的图片	4
杂项第三题: 隐写	6
杂项第四题: telnet	8
杂项第五题: 眼见非实(ISCCCTF)	9
杂项第六题: 啊哒	12
杂项第七题: 又一张图片, 还单纯吗	14
杂项第八题: 猜	17
杂项第九题: 宽带信息泄露	19
杂项第十题: 隐写 2	20
杂项第十一题: 多种方法解决	23
杂项第十二题: 闪的好快	25
杂项第十三题: come_game	26
杂项第十四题: 白哥的鸽子	28
杂项第十五题: linux	30
杂项第十六题: 隐写 3	30
杂项第十七题: 做个游戏(08067CTF)	33
杂项第十八题: 想蹭网先解开密码	35
杂项第十九题: Linux2	39
杂项第二十题: 细心的大象	42
杂项第二十一题: 爆照(08067CTF)	47
杂项第二十二题: 猫片(安恒)	51
杂项第二十三题: 旋转跳跃	57
音频工具 MP3stego 使用 (一)	59
音频工具 MP3stego 使用 (二)	60
杂项第二十四题: 普通的二维码	61
CTF 杂项之音频及视频隐写补充	64
杂项第二十五题: 乌云邀请码	71
杂项第二十六题: CTF 之隐写术--LSB 一张图片隐藏的信息	73
杂项第二十七题: convert	76
杂项第二十八题: 听首音乐	80
杂项第二十九题: ctf 练习---摩斯密码	83
杂项第三十题: 好多数值	84
杂项第三十一题: 神秘的文件	87
杂项第三十二题: 三十 zip 明文攻击	90
杂项第三十三题: 论剑	91

杂项第三十四题: 图穷匕见.....	94 ^u
杂项第三十五题: 很普通的数独(ISCCTF).....	99 ^u
杂项第三十六题: PEN_AND_APPLE	103 ^u
NTFS 数据流及高级文件隐藏.....	105 ^u
杂项第三十七题: color	107 ^u
杂项第三十八题: 小明的密码.....	110 ^u
杂项第三十九题: 仿射加密.....	111 ^u
仿射密码解析与实例.....	113 ^u
杂项第四十题: 黑客的机密信息	117 ^u
杂项第四十一题: 远控木马.....	118 ^u
杂项第四十二题: Web 漏洞	118 ^u
bugku-ctf 第四十三题: 颜文字	120 ^u
杂项第四十四题: 磁盘镜像.....	120 ^u
杂项第四十五题: 神奇的图片	121 ^u
杂项第四十六题: 怀疑人生	122 ^u
杂项第四十七-CTF 加密篇之 ok (Ook!)	129 ^u
杂项第四十八题: 红绿灯.....	131 ^u
杂项第四十九题: 不简单的压缩包.....	136 ^u



以下是对 41-49 题的介绍

杂项第四十一题: 远控木马

某次应急响应时, 工程师发现一个远控木马的客户端程序, 请分析该远程控制木马的控制端 IP 及端口号。

在虚拟机上运行 netstat -ano 查看

```
Microsoft Windows [版本 6.1.7600]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\A>netstat -ano

活动连接

```

协议	本地地址	外部地址	状态	PID
TCP	0.0.0.0:135	0.0.0.0:0	LISTENING	764
TCP	0.0.0.0:445	0.0.0.0:0	LISTENING	4
TCP	0.0.0.0:3389	0.0.0.0:0	LISTENING	1164
TCP	0.0.0.0:49152	0.0.0.0:0	LISTENING	420
TCP	0.0.0.0:49153	0.0.0.0:0	LISTENING	868
TCP	0.0.0.0:49154	0.0.0.0:0	LISTENING	932
TCP	0.0.0.0:49155	0.0.0.0:0	LISTENING	532
TCP	0.0.0.0:49156	0.0.0.0:0	LISTENING	524
TCP	0.0.0.0:49157	0.0.0.0:0	LISTENING	1624
TCP	192.168.179.129:139	0.0.0.0:0	LISTENING	4
TCP	192.168.179.129:59189	120.92.76.82:443	TIME_WAIT	0
TCP	192.168.179.129:59190	192.168.233.222:9099	SYN_SENT	2128
TCP	192.168.179.129:59191	117.144.244.85:8080	ESTABLISHED	1684
TCP	192.168.179.129:59192	111.13.34.18:443	ESTABLISHED	1684
TCP	:::1:135	:::1:0	LISTENING	764
TCP	:::1:445	:::1:0	LISTENING	4
TCP	:::1:3389	:::1:0	LISTENING	1164

杂项第四十二题: Web 漏洞

黑客利用漏洞从 Web 系统中窃取了什么机密信息?

日志读取

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-
import urllib
import re
```

```
flag = ""
flag_list = []
```

```
with open('access.log', 'r') as file:
```

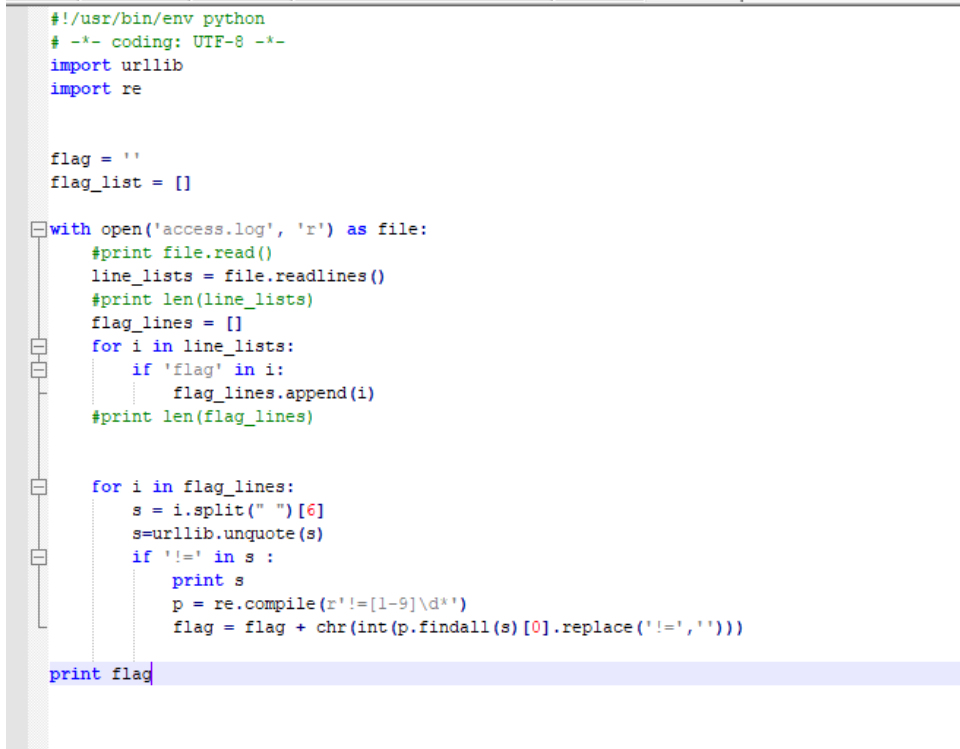
```

#print file.read()
line_lists = file.readlines()
#print len(line_lists)
flag_lines = []
for i in line_lists:
    if 'flag' in i:
        flag_lines.append(i)
#print len(flag_lines)

for i in flag_lines:
    s = i.split(" ")[6]
    s=urllib.unquote(s)
    if '!=' in s :
        print s
        p = re.compile(r'!=[1-9]\d*')
        flag = flag + chr(int(p.findall(s)[0].replace('!=','')))

```

print flag



```

#!/usr/bin/env python
# -*- coding: UTF-8 -*-
import urllib
import re

flag = ''
flag_list = []

with open('access.log', 'r') as file:
    #print file.read()
    line_lists = file.readlines()
    #print len(line_lists)
    flag_lines = []
    for i in line_lists:
        if 'flag' in i:
            flag_lines.append(i)
    #print len(flag_lines)

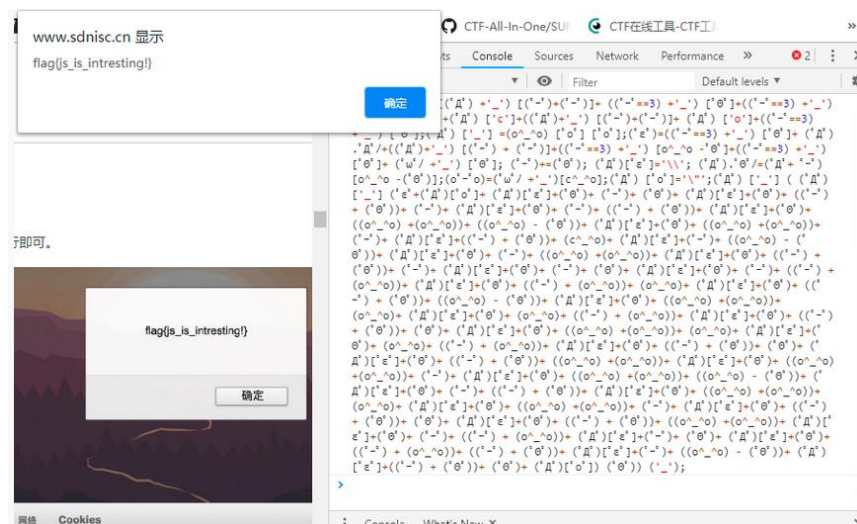
    for i in flag_lines:
        s = i.split(" ")[6]
        s=urllib.unquote(s)
        if '!=' in s :
            print s
            p = re.compile(r'!=[1-9]\d*')
            flag = flag + chr(int(p.findall(s)[0].replace('!=','')))

print flag

```

bugku-ctf 第四十三题： 颜文字

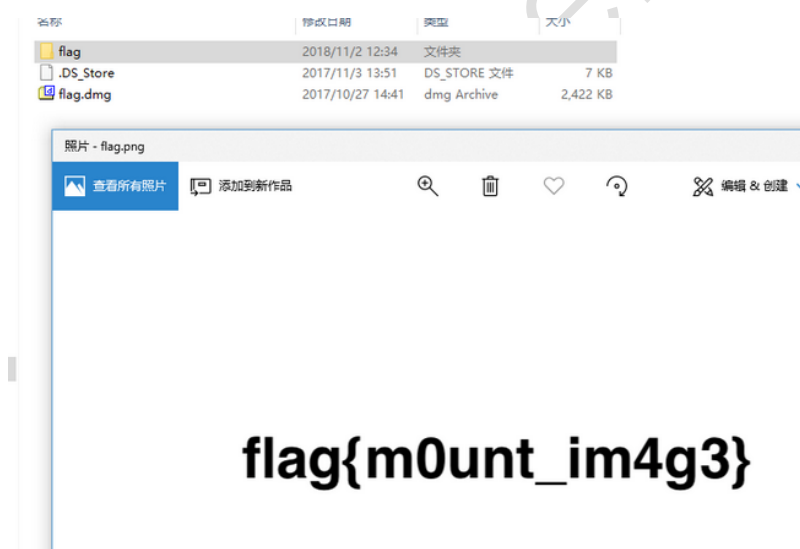
(ノ`Д)ノ



杂项第四十四题: 磁盘镜像

磁盘里藏着 flag

7z 解压, 挂载, 分离都可以



杂项第四十五题: 神奇的图片

可能是二维码也可能是图形, 要根据生成的图片进行判断。

#!/usr/bin/env python2

-*- coding: UTF-8 -*-

```

from PIL import Image

x = 150
y = 900
pic = Image.new("RGB", (x, y))
f = open("basic.txt", "r")
flag_s = []

for i in f.readlines():
    j = i.strip("\n").replace(" ", "").replace("(", "")
    flag_s.append(j)
f.close()

s = ""
k = 0
for i in range(x):
    for j in range(y):
        s = flag_s[k].split(",")
        pic.putpixel([i, j], (int(s[0]), int(s[1]), int(s[2])))
        k += 1

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

```

```

1  #!/usr/bin/env python2
2  # -*- coding: UTF-8 -*-
3  from PIL import Image
4
5  x = 150
6  y = 900
7  pic = Image.new("RGB", (x, y))
8  f = open("basic.txt", "r")
9  flag_s = []
10
11 for i in f.readlines():
12     j = i.strip("\n").replace(" ", "").replace("(", "")
13     flag_s.append(j)
14 f.close()
15
16 s = ""
17 k = 0
18 for i in range(x):
19     for j in range(y):
20         s = flag_s[k].split(",")
21         pic.putpixel([i, j], (int(s[0]), int(s[1]), int(s[2])))
22         k += 1
23
24 pic.show()
25 pic.save("flag.png")

```

更改 x, y 值可以看到里面不是二维码而是数字，继续修改到合适的位置得到的图片用 ps 翻转即可得到下面的图

flag{RGB_1s_e4sY}

杂项第四十六题：怀疑人生

Challenge 475 Solves x

怀疑人生

150

zip

Flag Submit

下载压缩包

<https://ctf.bugku.com/files/8d61ff10962c756f9b1d8cd048e9f3c6/zip>

这个链接不能直接打开，大概是因为没有注册 bugku 账户登录的话无法直接访问。

更改后缀名，解压

怀疑人生



就发现了这么三个东西 直接解压哈哈哈

第一部分，压缩包解压，发现有密码，那就爆破吧

口令已成功恢复!

Advanced Archive Password Recovery 统计信息:	
总计口令	129,040,365,507
总计时间	1h 4m 7s 145ms
平均速度(口令/秒)	33,541,851
这个文件的口令	password
十六进制口令	70 61 73 73 77 6f 72 64

得到密码解压 文本文件

ctf.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
XHU2N1x1NmNcdTYxXHU2N1x1N2JcdTY4XHU2MVx1NjNcdTZiXHU2NVx1NzI=
```

这应该是 base64 把 解密看看

米斯特安全团队 CTFCrackingTools pro v2.1 Beta

解码方式 进制转换 插件 妹子

Crypto Image UnZip

填写所需检测的密码: (已输入字符数统计: 60)

```
XHU2N1x1NmNcdTYxXHU2N1x1N2JcdTY4XHU2MVx1NjNcdTZiXHU2NVx1NzI=
```

结果: (字符数统计: 44)

```
\u66\u6c\u61\u67\u7b\u68\u61\u63\u6b\u65\u72
```

得到 Unicode 编码

```
\u66\u6c\u61\u67\u7b\u68\u61\u63\u6b\u65\u72
```

之后在解码

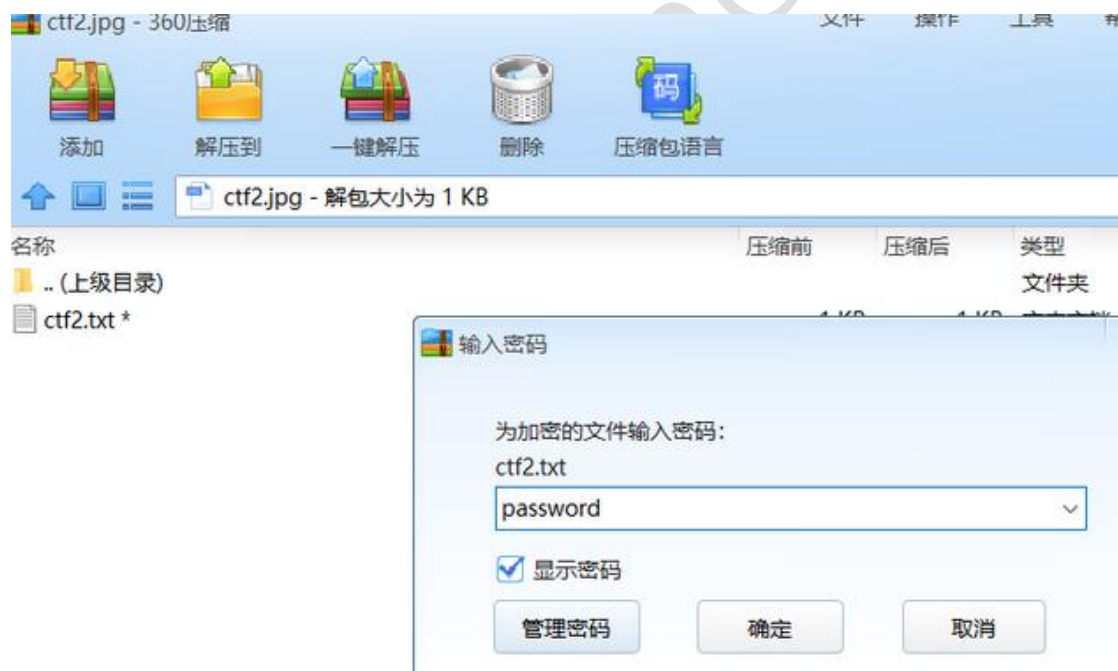
填写所需检测的密码: (已输入字符数统计: 44)

\u66\u6c\u61\u67\u7b\u68\u61\u63\u6b\u65\u72

结果: (字符数统计:

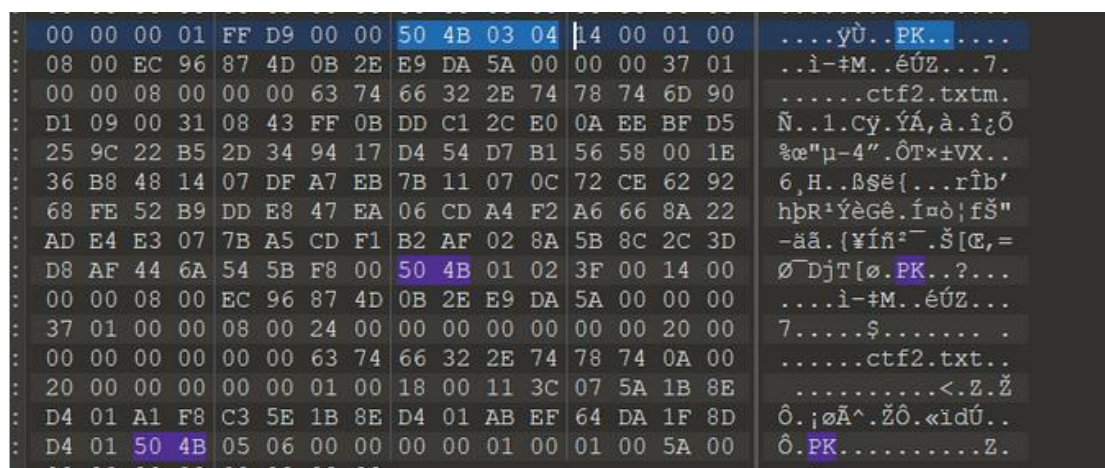
flag{hacker

接着看第二个图片 ctf2 我还是用 360 压缩打发现有东西 但是需要密码



用之前的密码是错误的

那么在十六进制编译器里打开看看



发现有 .zip 的文件头和文件尾：

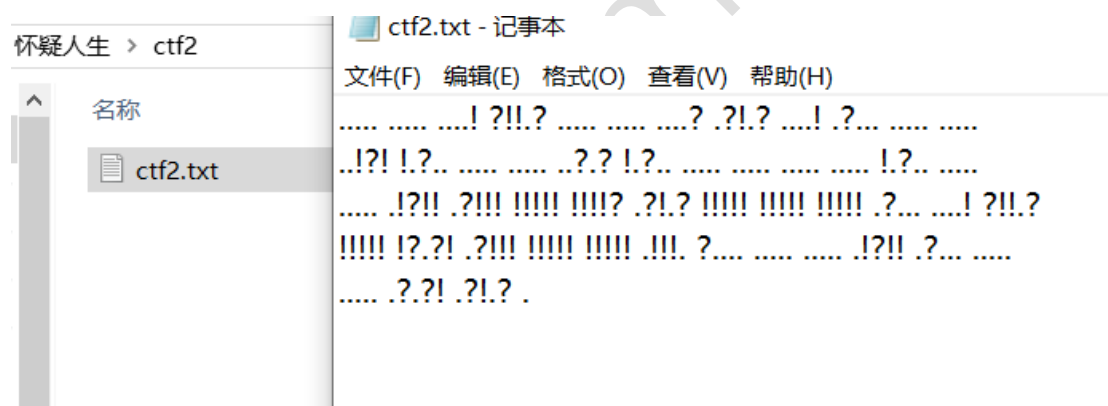
(https://blog.csdn.net/qg_42777804/article/details/98876791)

ZIP Archive (zip),

文件头：504B0304

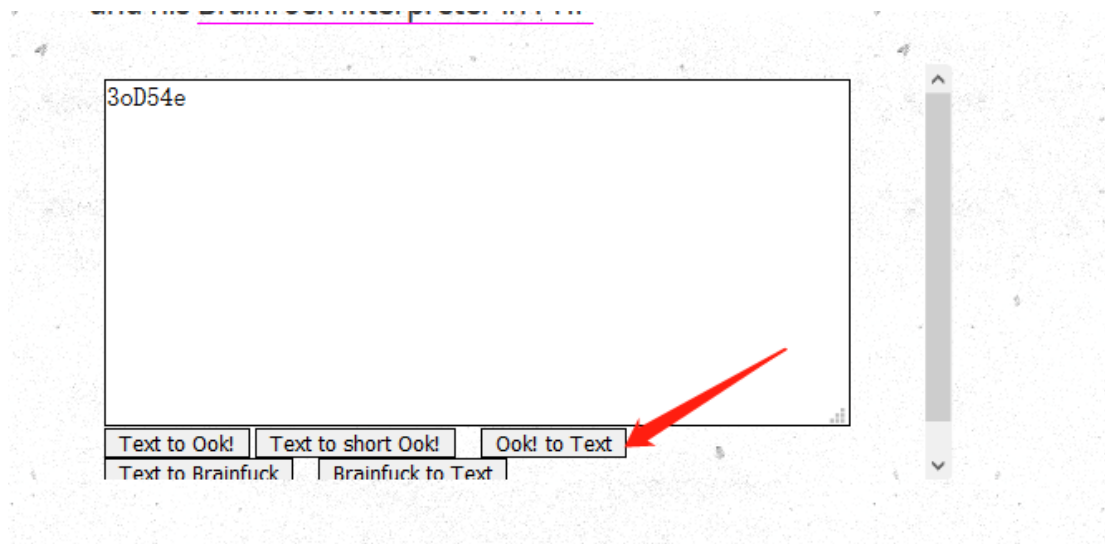
文件尾：50 4B

那么我们直接将他后缀改为 .zip 发现可以解压
解压后得到文本文件



发现这个是 ook 编码

那么在线 解码 <https://www.splitbrain.org/services/ook>



得到 3oD54e

查了好久这个居然是 base58

还是 base58 在线解码 <https://www.jisuan.mobi/pbHzbBHbzHB6uSJx.html>

字符串

3oD54e

计算

解码结果

misc

复制

第三张图片是一张二维码 扫描 得到：
我用 qq 直接扫描出来了!!!!!!

解码结果:

12580}

生成二维码

美化二维码

最后部分为 12580}

flag{hackermisc12580}

但是我看其他大佬都是用这个软件扫出来的

公众号 LemonSec



纠错等级

H(30%)

掩码

Auto

版本

Auto

☐ Auto

尺寸

4

已解码数据 1:

位置: (16.0,16.0)-(724.0,16.0)-(16.0,724.0)-(724.0,724.0)

颜色正常, 正像

版本: 40

纠错等级: H, 掩码: 0

内容:

12580}

杂项第四十七-CTF 加密篇之 ok (Ook!)

Challenge 5040 Solves x

ok
30

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

ok

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

本题要点: Ook! 编码

首先看到题目~

哇!

好长的 Ook! 啊~~~

这个就只能依靠日常积累和万能的搜索引擎了~~~
有一个非常好用的在线网站~

<https://www.splitbrain.org/services/ook>

来吧~ 复制粘贴

languages.

All the hard work (like actually understanding how those languages work) was done by Dai and his [Brainfuck interpreter in PHP](#)

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

Text to Ook! Text to short Ook! Ook! to Text
Text to Brainfuck Brainfuck to Text

在线解码
bingo~

```
flag{ok-ctf-1234-admin}
```

Text to Ook! Text to short Ook! Ook! to Text
Text to Brainfuck Brainfuck to Text

Ook. Ook.

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

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

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

Flag

Submit

Correct

杂项第四十八题：红绿灯

Challenge 271 Solves ×

红绿灯
150

Traffic_Light.gif

Flag

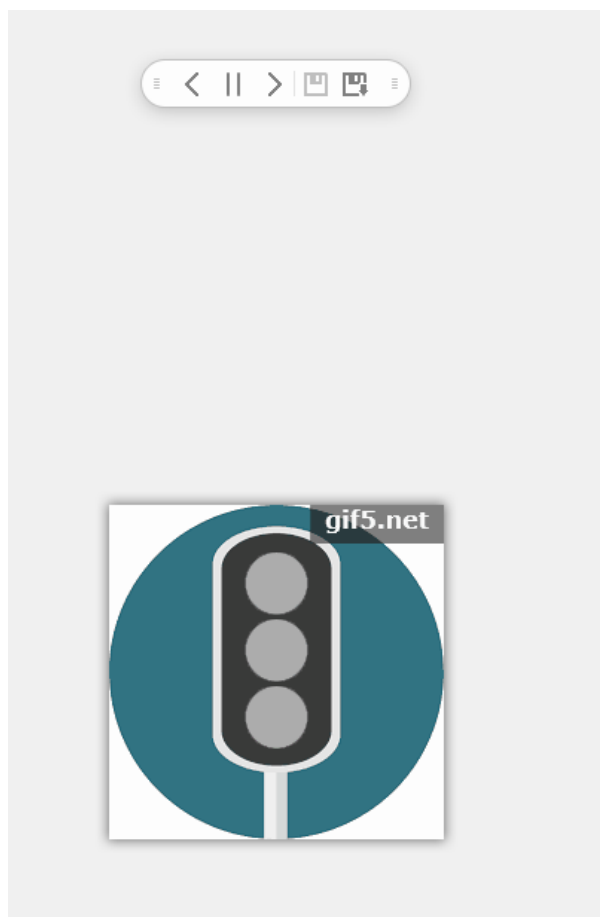
Submit

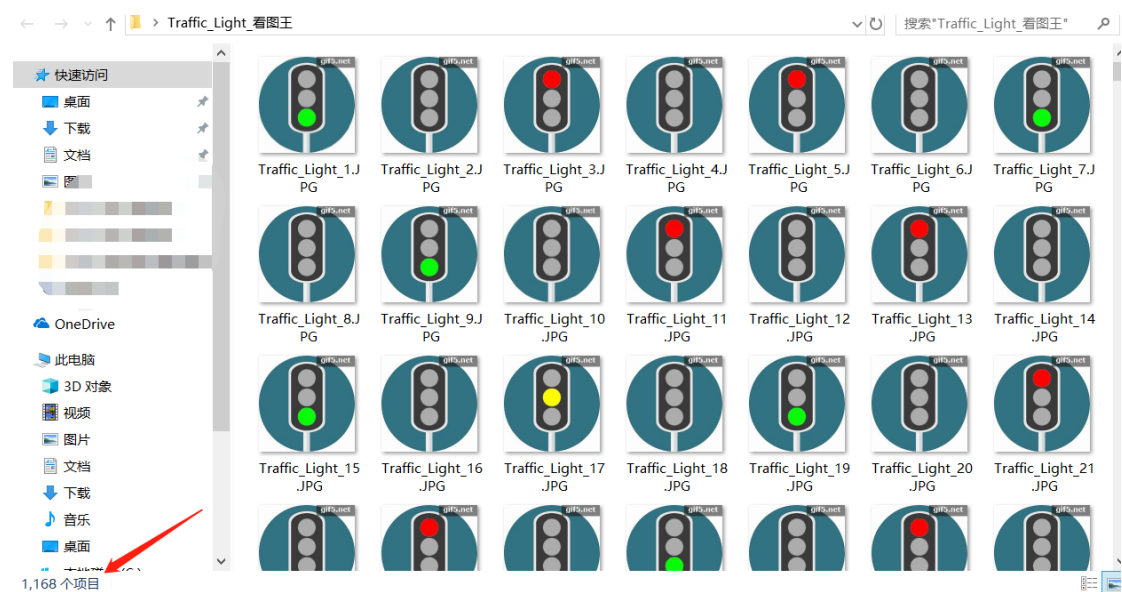
https://ctf.bugku.com/files/65beb9db8419bd99c0d97068959d2b3e/Traffic_Light.gif

打开图片然后另存到桌面
是个红绿灯的动图



题目内容为一个 gif 的文件，然后使用 2345 看图王打开，并将每一帧图片都保存下来





1168 个图

其中有大概一半左右的图片为没有颜色的红绿灯，将其全部删除。



删除完之后，在文件夹里面调节大小的时候，突然发现最左侧和最右侧的一列颜色都是一样的，推测最右侧的一列为空格，因为 ascii128 位，最左侧的为 0，得出绿色为 0，红色为 1，留下中间的 7 列。

然后把图片的大小调节为合适的大小，7 个一行。

一共 455 张图片，65 组 7 位二进制数，红灯为 1，绿灯为 0，将图片信息记录下来

```
110011011011001100001110011111110111010000110110001100110110100111001101100
111011111111100000110100111100110111101101001110100111010001100111101110111
0100011000101100001101110101111111010001100001011111110100111001001101001
10011011001100110001110001110111111110011011010011001100111110100111100
1101111111101111101000011001111011101011111111001011000011101011011110110
100111001001100111011111011000011101011110100111001101100011100100011001111
11101
```

[illegible]

```
1 def fun1():#二进制字符串转换字符串
2     #需要转换的字符串
3     f = '1100110110110011000011100111111101110100001101100011001101101001110
4     b = ''
5     i = 0
6     j = 7
7     while j <= len(f):
8         a = '0' + f[i:j]
9         b += chr(int(a,2))
10        i = j
11        j += 7
12    print(b)
13 def fun2():#字符串转换二进制字符串
14     #需要转换的字符串
15     f = ' '
16     b = ''
17     c = ''
18     for i in f:
19         a = str(bin(ord(i)))
20         b = a[2:].zfill(7)
21         c += b
22    print(c)
23 fun1()
24 #fun2()
```

Run: 转换_二进制_字符串 ×

flag{Pl34s3_p4y_4tt3nt10n_t0_tr4ff1c_s4f3ty_wh3n_y0u_4r3_0uts1d3}

Process finished with exit code 0

flag{Pl34s3_p4y_4tt3nt10n_t0_tr4ff1c_s4f3ty_wh3n_y0u_4r3_0uts1d3}

杂项第四十九题：不简单的压缩包

Challenge

184 Solves

×

不简单的压缩包

150

zip

Flag

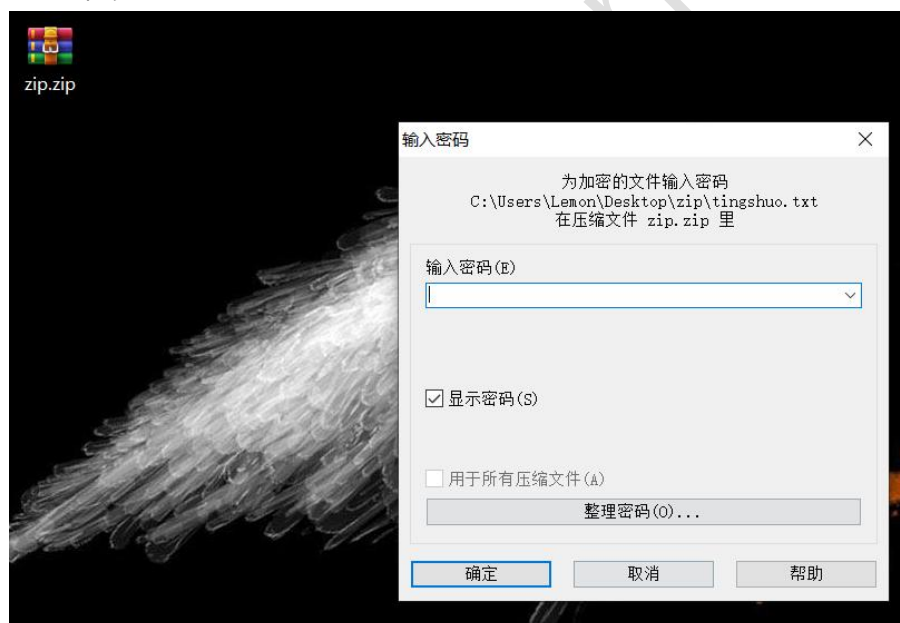
Submit

下载压缩包

<https://ctf.bugku.com/files/e5a937a3985f5264a723bcbd0e062b0f/zip>

更改后缀名后，解压

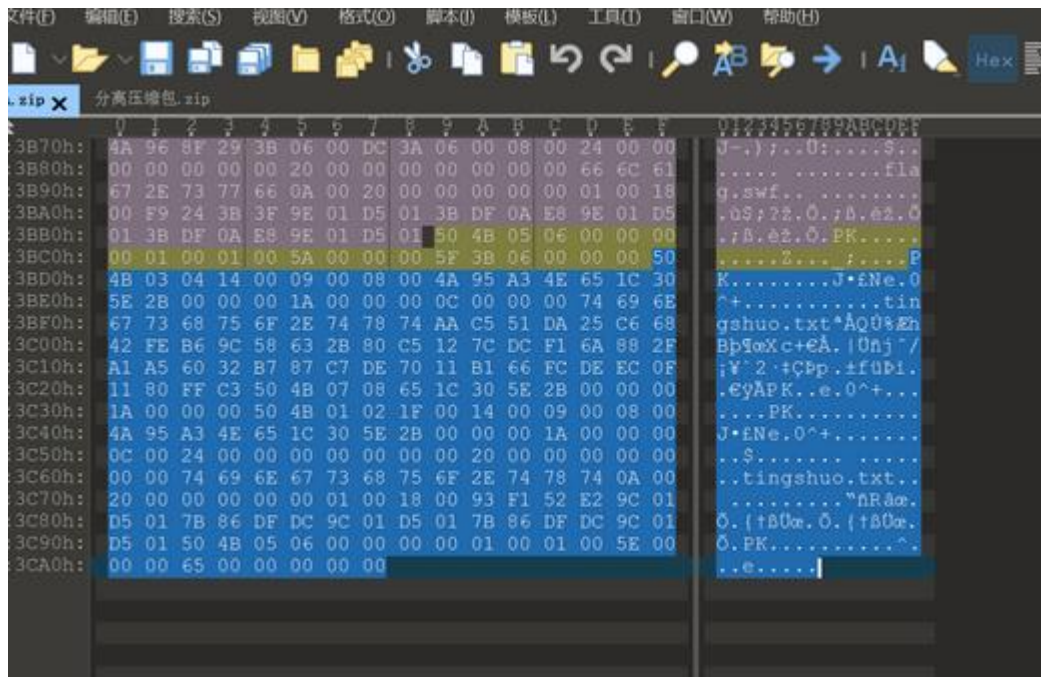
需要密码



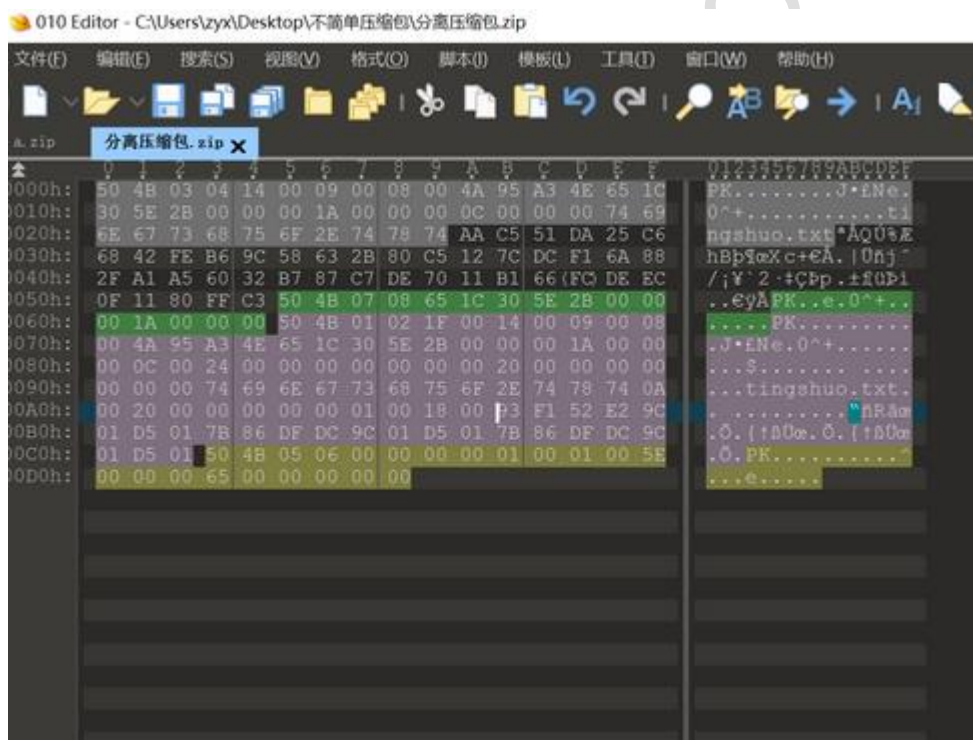
打开 010

在末尾发现第二个压缩包

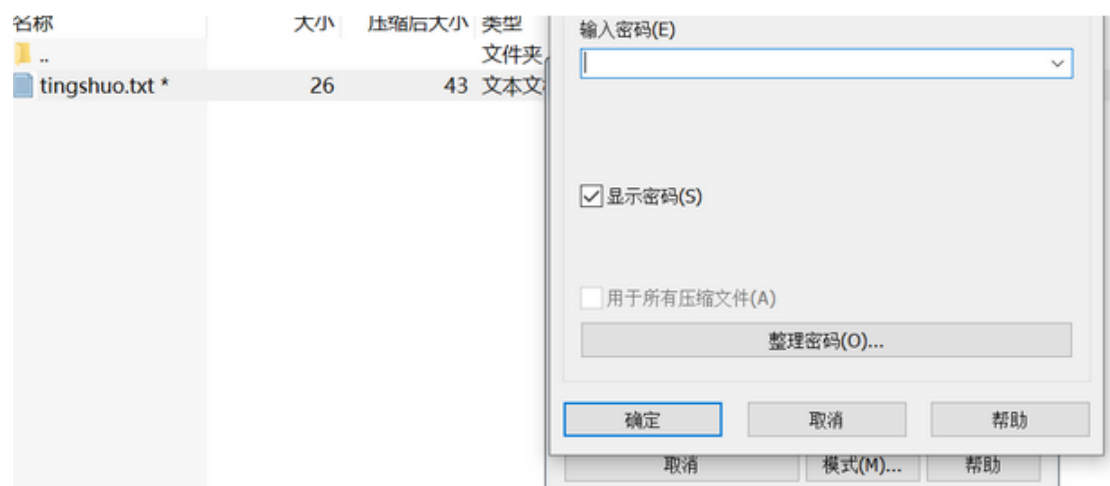
zip 文件头是 504B0304，文件尾 504B



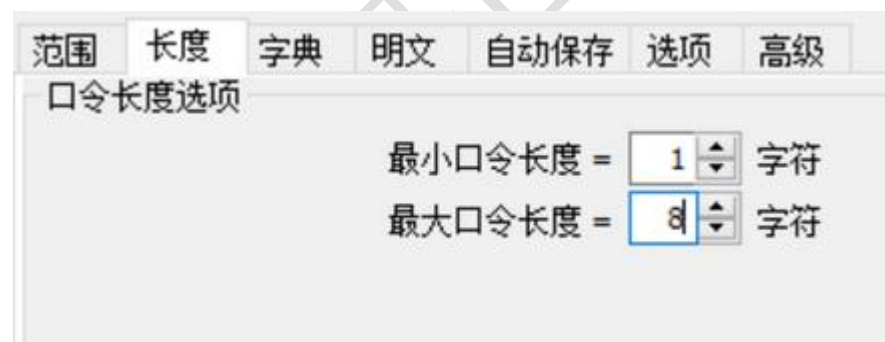
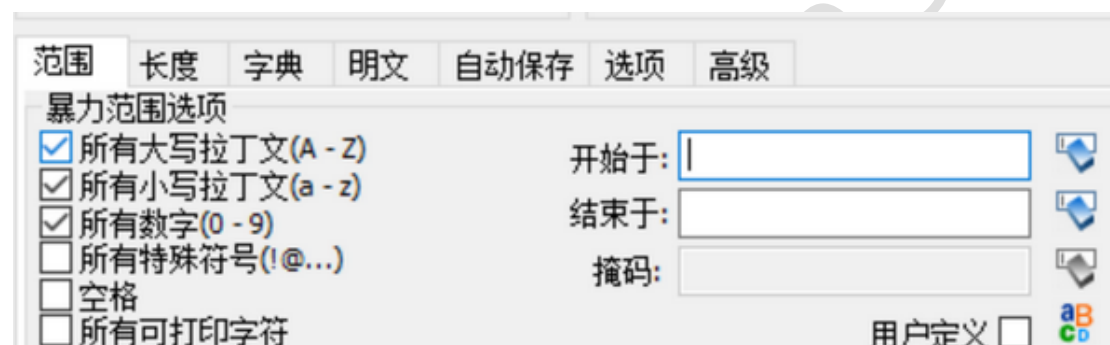
手动分离出去或者使用 binwalk 分离



另存为第二个 zip



打开，有密码，以为是伪加密，把 140009 改成 140000，进压缩包，提示解压文件损坏。。那行吧，不是伪加密，只能放 ARCHPR 暴力破解，说不定破解几个小时就出来了。。然鹅。。。。事情并不是这样

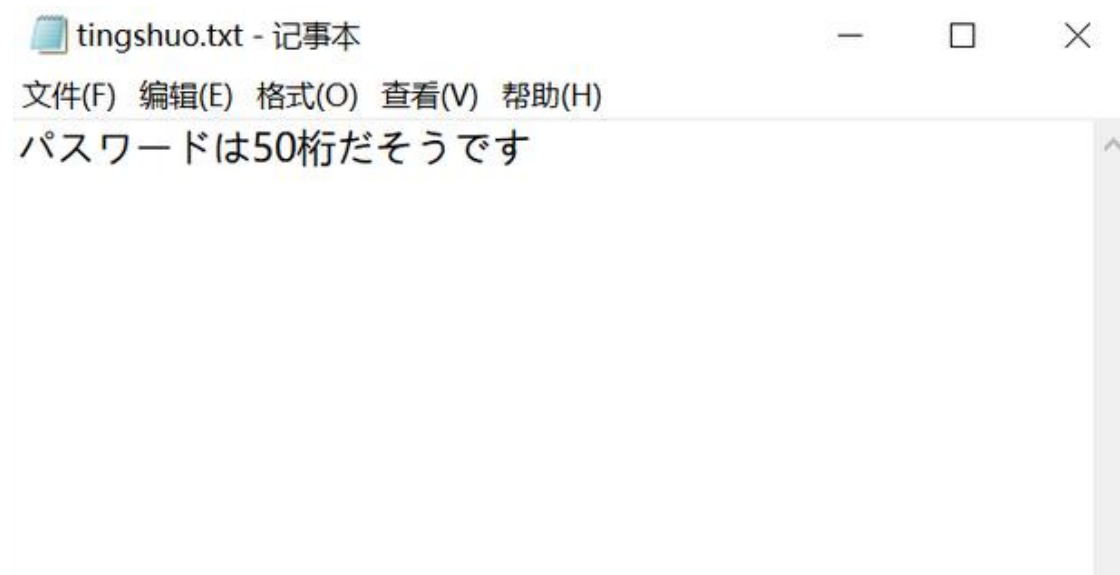


口令已成功恢复!

Advanced Archive Password Recovery 统计信息:	
总计口令	1
总计时间	9ms
平均速度(口令/秒)	111
这个文件的口令	0
十六进制口令	30

居然一运行就跑出了密码，密码是 0。。。。

接下来，你们懂的，解压出 tingshuo.txt，发现一段谜之日语



翻译一下



密码 50 位，对应的应该是 zip.zip 的，但是密码它有 50 位？？？暴力破解能解到地老天荒，密码会不会在属性里？看了一下属性好像没什么问题（之前还试过流隐写破解，还是没什么发现）

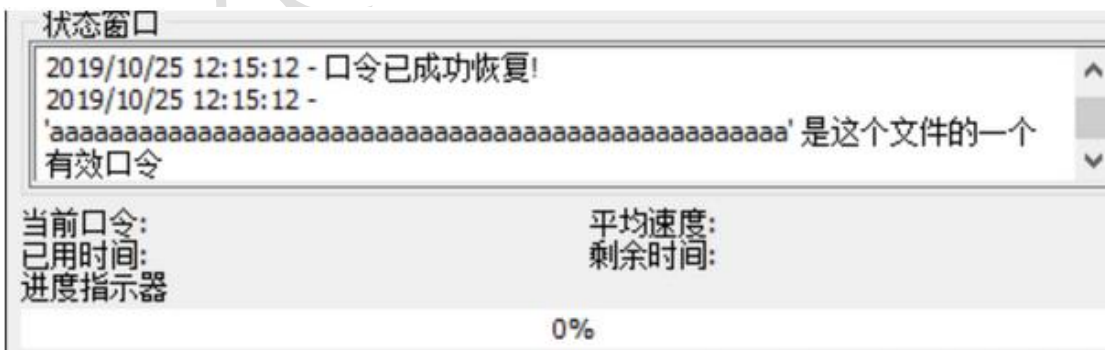
那，猜猜密码？？50 位密码，如果混合组合，肯定没法解开，要不然试试相同符号密码？于是我做了一个这样的字典。

[illegible]

同字符

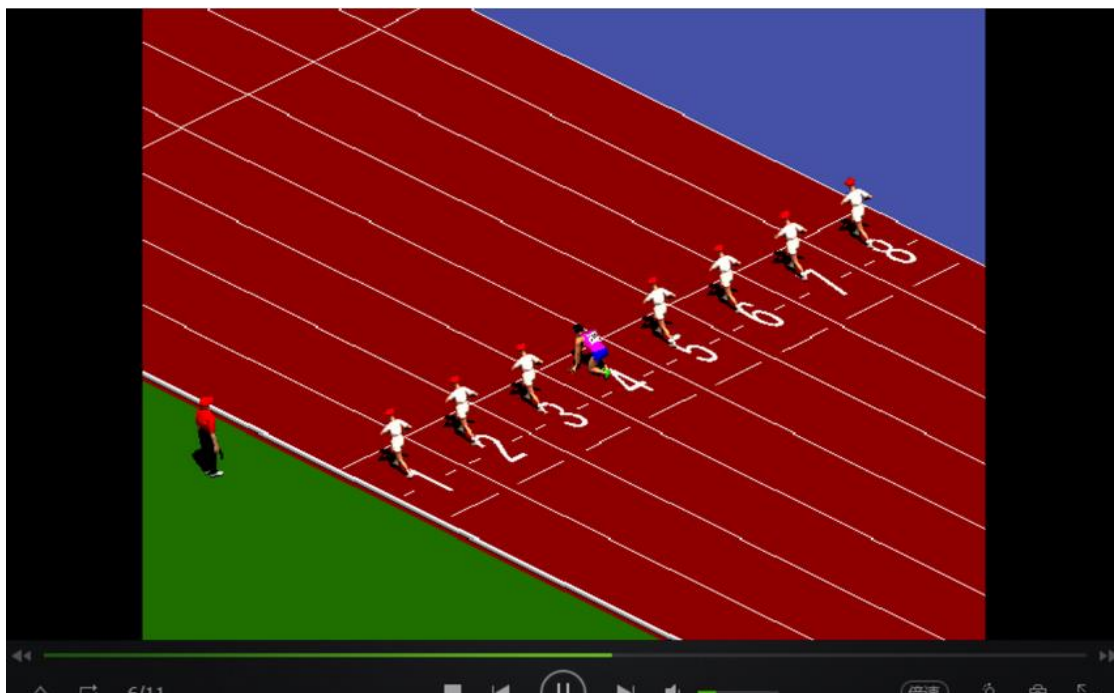
跑一下字典

跑一下字典



50 ↑ a


好了，打开 zip.zip，输入密码，解压出 flag.swf 文件



一个谜之游戏，我试了好几次，通不了关，可能是我太菜了吧
我想着。。。是一个游戏，通关之后可能会有 flag，试试改存档？

但是并没有存档文件出现在文件夹里。。。

调整一下思路
在百度搜索.swf

 [进入词条](#)

目录

- 1 基本信息
- 2 文件结构
- 3 SWF填充
- 4 格式转换

基本信息

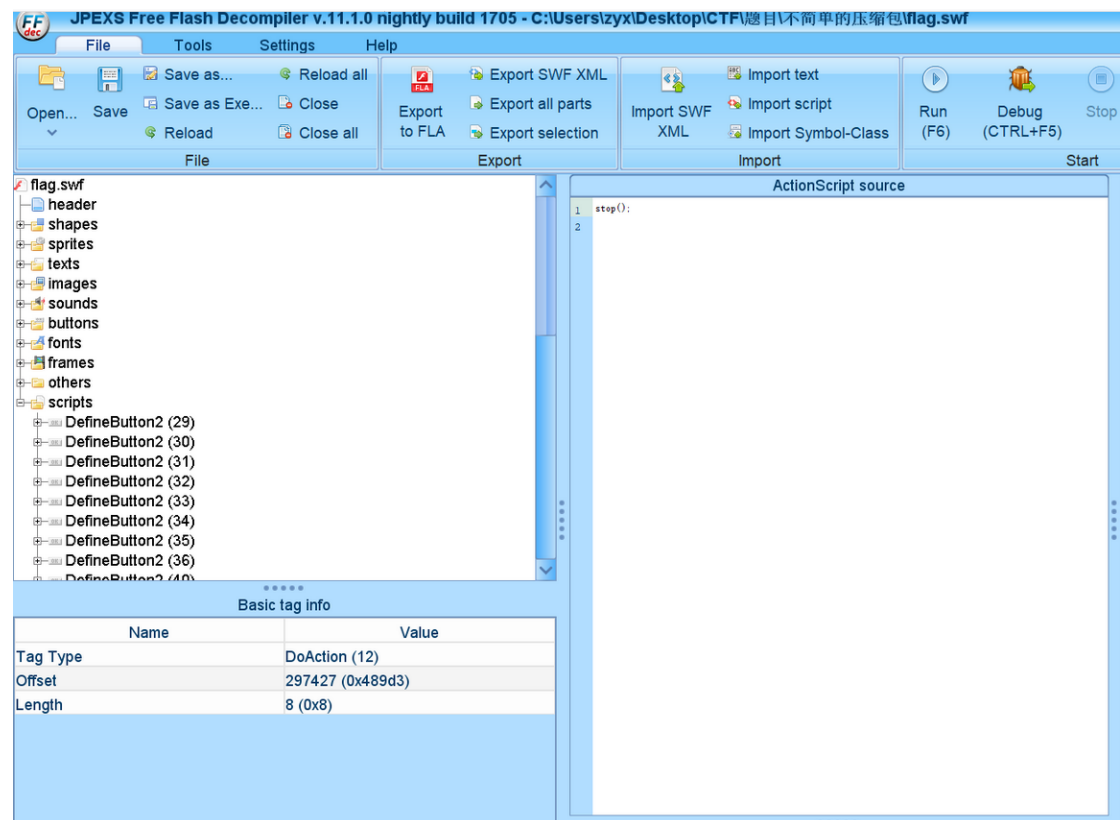
SWF是一种基于矢量的Flash动画文件，一般用FLASH软件创作并生成SWF文件格式，也可以通过相应软件将PDF等类!换为SWF格式。SWF格式文件广泛用于创建吸引人的应用程序，它们包含丰富的视频、声音、图形和动画。可以在Flash中创建原始内容或者从其它Adobe应用程序（如Photoshop或Illustrator）导入它们，快速设计简单的动画，以及使用Adobe Animate 3.0开发高级的交互式项目。设计人员和开发人员可使用它来创建演示文稿、应用程序和其它允许用户交互的内容。Flash可以包含简单的动画、视频内容、复杂演示文稿和应用程序以及介于它们之间的任何内容。通常，使用Flash创作的各个内容单元称应用程序，即使它们可能只是很简单的动画。您也可以通过添加图片、声音、视频和特殊效果，构建包含丰富媒体的Flash应用程序。

如何播放SWF：

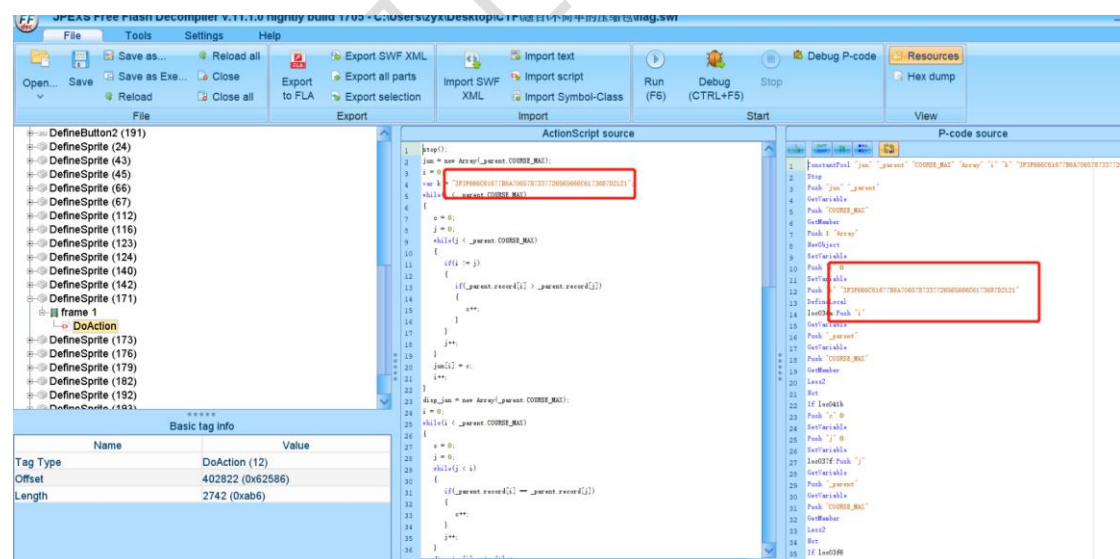
可以利用FLASH控件实现播放FLASH的SWF文件，常用的第三方软件（如：实用Flash播放器、超级Flash播放器 ^[1]、SWF Flash Player）可以直接在主流下载站下载后并安装，即可使用。

既然是应用程序那就试试反编译看源码喽，也许修改能直接通关吧。

然后，我在网上找到一个名叫 JPEXS Free Flash Decompiler 的软件。
用它打开 flag.swf



一个一个看源码，在 scripts 中有一个名叫 DefineSprite (171)的项目，点开之后发现一段奇怪的字符串。



Push "k" "3F3F666C61677B6A7065787337726566666C6173687D2121"

看上去像是 16 进制，转换一下试试，



密码居然在这里

填进 flag，居然通过了!!!!

flag{jpeks7reeflash}

看了大佬的解题思路，脑洞太多了，自己解不出来。。。

整理了下最近做题常用到的工具，自行考虑下载，毕竟这些工具也是来源于网络！