

# MISC-zip压缩包的总结

原创

Qwzf 于 2019-11-27 18:57:16 发布 5725 收藏 54

分类专栏: [CTF](#) 文章标签: [CTF](#)

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/qq\\_43625917/article/details/96148661](https://blog.csdn.net/qq_43625917/article/details/96148661)

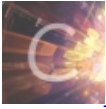
版权



[CTF](#) 同时被 3 个专栏收录

30 篇文章 6 订阅

订阅专栏



[ZIP](#)

1 篇文章 0 订阅

订阅专栏



[MISC](#)

4 篇文章 0 订阅

订阅专栏

---

## MISC-zip压缩包的总结

做了那么多的MISC压缩包的CTF题, 是时候总结一下经验了。手撕压缩包走起!

### 1、伪加密

zip中有一位是标记文件是否加密的, 如果更改一个未加密zip包的加密标记位, 那么在打开压缩包时就会提示该文件是加密的。

**压缩源文件数据区:**

50 4B 03 04: 这是头文件标记 (0x04034b50)  
 14 00: 解压文件所需 pkware 版本  
 00 00: 全局方式位标记 (有无加密)  
 08 00: 压缩方式  
 5A 7E: 最后修改文件时间  
 F7 46: 最后修改文件日期  
 16 B5 80 14: CRC-32校验 (1480B516)  
 19 00 00 00: 压缩后尺寸 (25)  
 17 00 00 00: 未压缩尺寸 (23)  
 07 00: 文件名长度  
 00 00: 扩展记录长度  
 6B65792E7478740BCECC750E71ABCE48CDC9C95728CECC2DC849AD284DAD0500

**压缩源文件目录区:**

50 4B 01 02: 目录中文件文件头标记(0x02014b50)  
 3F 00: 压缩使用的 pkware 版本  
 14 00: 解压文件所需 pkware 版本  
 00 00: 全局方式位标记 (有无加密, 这个更改这里进行伪加密, 改为09 00打开就会提示有密码了)  
 08 00: 压缩方式  
 5A 7E: 最后修改文件时间  
 F7 46: 最后修改文件日期  
 16 B5 80 14: CRC-32校验 (1480B516)  
 19 00 00 00: 压缩后尺寸 (25)  
 17 00 00 00: 未压缩尺寸 (23)  
 07 00: 文件名长度  
 24 00: 扩展字段长度  
 00 00: 文件注释长度  
 00 00: 磁盘开始号  
 00 00: 内部文件属性  
 20 00 00 00: 外部文件属性  
 00 00 00 00: 局部头部偏移量  
 6B65792E7478740A00200000000000010018006558F04A1CC5D001BDEBDD3B1CC5D001BDEBDD3B1CC5D001

**压缩源文件目录结束标志:**

50 4B 05 06: 目录结束标记  
 00 00: 当前磁盘编号  
 00 00: 目录区开始磁盘编号  
 01 00: 本磁盘上纪录总数  
 01 00: 目录区中纪录总数  
 59 00 00 00: 目录区尺寸大小  
 3E 00 00 00: 目录区对第一张磁盘的偏移量  
 00 00: ZIP 文件注释长度

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	50	4B	03	04	14	00	01	00	08	00	5A	7E	F7	46	16	B5	PK
00000010	80	14	19	00	00	00	17	00	00	00	07	00	00	00	6B	65	Z~縹
00000020	79	2E	74	78	74	0B	CE	CC	75	0E	71	AB	CE	48	CD	C9	ke
00000030	C9	57	28	CE	CC	2D	C8	49	AD	28	4D	AD	05	00	50	4B	y.txt 翁u q沔H蛄
00000040	01	02	3F	00	14	00	09	00	08	00	5A	7E	F7	46	16	B5	蒞(翁-莢?M? PK
00000050	80	14	19	00	00	00	17	00	00	00	07	00	24	00	00	00	? 縹 Z~ F μ
00000060	00	00	00	00	20	00	00	00	00	00	00	00	6B	65	79	2E	\$
00000070	74	78	74	0A	00	20	00	00	00	00	00	01	00	18	00	65	key.
00000080	58	F0	4A	1C	C5	D0	01	BD	EB	DD	3B	1C	C5	D0	01	BD	txt e
00000090	EB	DD	3B	1C	C5	D0	01	50	4B	05	06	00	00	00	00	01	X縹 判 縹? 判
000000A0	00	01	00	59	00	00	00	3E	00	00	00	00	00	00	00	00	縹; 判 PK
																	Y >

**重点部分**

把504B0304后的第3、4个byte改成0000还有  
把504B0102后的第5、6个byte改成0000即可破解伪加密。

### 识别真假加密

#### 无加密

压缩源文件数据区的全局加密应当为00 00  
且压缩源文件目录区的全局方式位标记应当为00 00

#### 假加密

压缩源文件数据区的全局加密应当为00 00  
且压缩源文件目录区的全局方式位标记应当为09 00

#### 真加密

压缩源文件数据区的全局加密应当为09 00

例：

[题目下载](#)

Challenge 24 Solves ×

## 不成熟的加密

10

奈何本人技术差，一伪加密行天下

**Blog.zip**

Key

**SUBMIT**

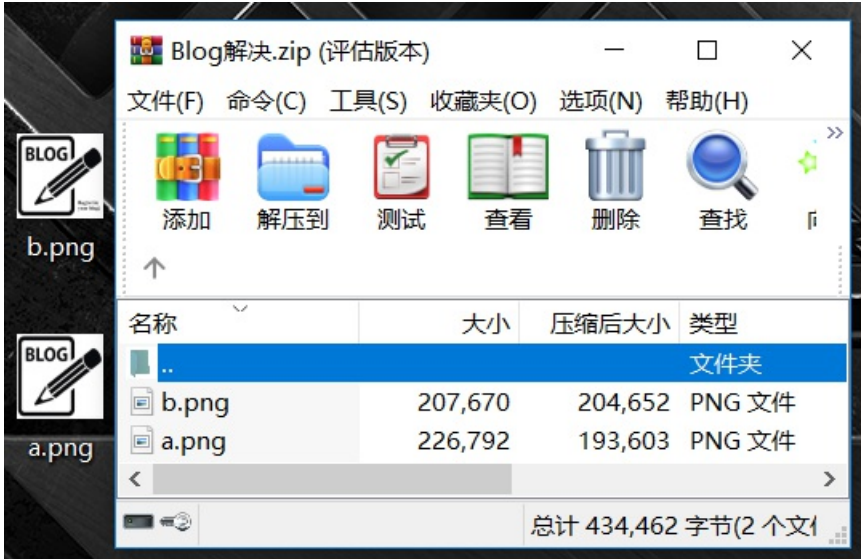
根据题目，这可能是一道伪加密题。把压缩包文件用winrar打开，看出来b.png进行了伪加密



把Blog的zip压缩包放进winhex,在最上面没发现伪加密标志位, 所以我在最下面发现

1B FF A4 7F 00 50 4B 01 02 3F 00 14 00 00 00 08	ÿPK ?
00 9A A8 75 4E 21 79 19 E3 43 F4 02 00 E8 75 03	š`uN!y äCò èu
00 05 00 24 00 00 00 00 00 00 00 20 00 00 00 00	§
00 00 00 61 2E 70 6E 67 0A 00 20 00 00 00 00 00	a.png
01 00 18 00 F1 EB A4 AB E6 DF D4 01 4B 2B 72 AB	ñè«æßÔ K+r«
E6 DF D4 01 7F 14 29 AB E6 DF D4 01 50 4B 01 02	æßÔ )«æßÔ PK
3F 00 14 00 09 00 08 00 15 A6 75 4E 47 F4 68 02	? !uNGôh
6C 1F 03 00 36 2B 03 00 05 00 24 00 00 00 00 00	l 6+ §
00 00 20 00 00 00 00 00 01 00 18 00 62 8D E7 68 E4	fô b.png
DF D4 01 2F 10 E2 68 E4 DF D4 01 6A 72 8A 8B E1	b çhä
DF D4 01 50 4B 05 06 00 00 00 00 02 00 02 00 AE	ßÔ / âhãßÔ jrš<á
00 00 00 F5 13 06 00 00 00	ßÔ PK @
	õ

把b.png的504B0102后第五位09改成00, 即可破解伪加密, 解压即可在b.png中得到flag!!



## 2、明文攻击

明文攻击是一种较为高效的攻击手段，大致原理是当不知道一个zip的密码，但是有zip中的一个已知文件（文件大小要大于12Byte）时，因为同一个zip压缩包里的所有文件都是使用同一个加密密钥来加密的，所以可以用已知文件来找加密密钥，利用密钥来解锁其他加密文件。

推荐一个工具:APCHPR(可进行爆破/明文/字典/掩码攻击)

例:

[题目下载](#)

Challenge 0 Solves

# 明文

10

已知明文, 言尽于此

zip

Key

SUBMIT

根据题目看出来这道题可能是明文攻击，下载之后得到zip压缩包，用winrar打开

zip (评估版本)

文件(F) 命令(C) 工具(S) 收藏夹(O) 选项(N) 帮助(H)

添加 解压到 测试 查看 删除 查找 向导 信息 扫描病毒 注释

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
blingbling.zip	9,755	9,558	WinRAR ZIP 压缩...	2019/3/20 21:...	4FB0B314
提示.txt	39	39	文本文档	2019/3/20 21:...	5C6114DB

blingbling.zip (评估版本)

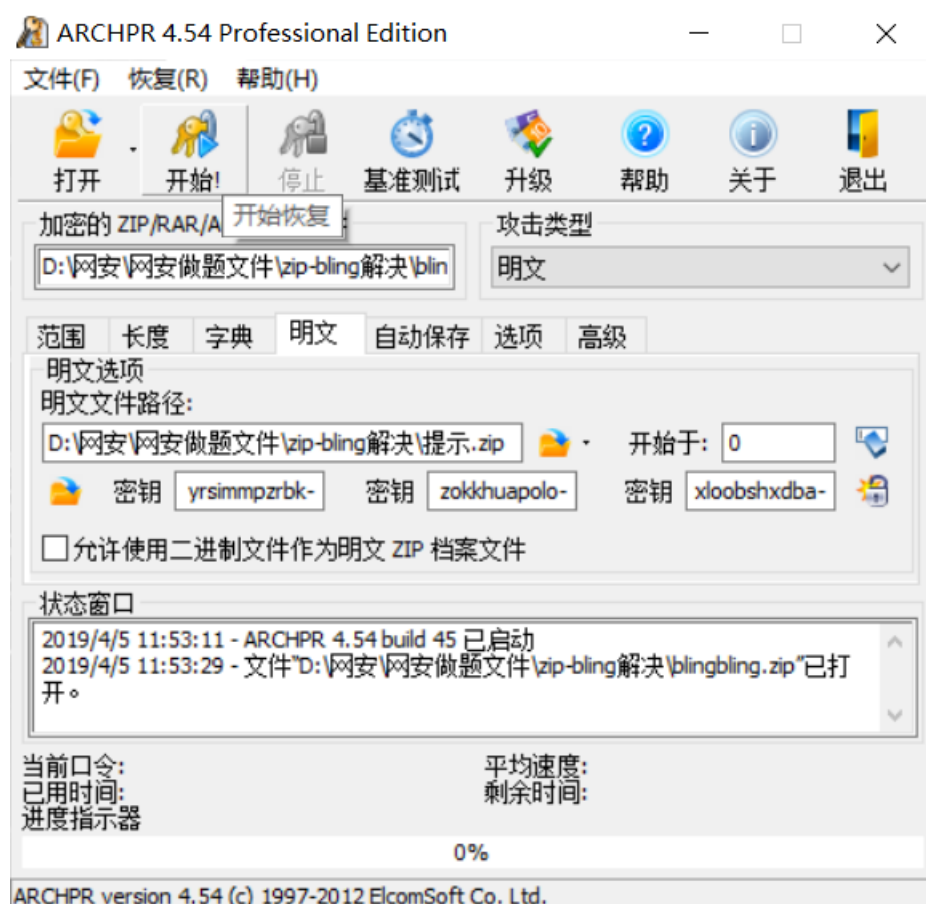
文件(F) 命令(C) 工具(S) 收藏夹(O) 选项(N) 帮助(H)

添加 解压到 测试 查看 删除 查找 向导 信息 扫描病毒 注释

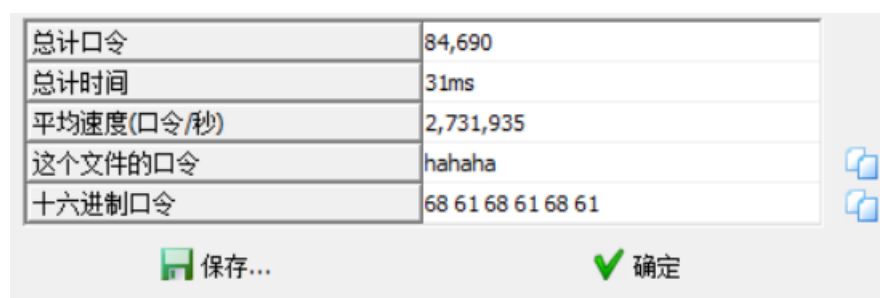
名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
提示.txt *	39	51	文本文档	2019/3/20 21:...	5C6114DB
空白文档? ? ? ...	11,945	9,226	Microsoft Word ...	2019/3/20 20:...	F6928DE8

总计 11,984 字节(2 个文件)

在zip文件里有个未加密的提示.txt文件，在blingbling.zip有个加密的提示.txt文件。所以这应该是明文攻击。把未加密的提示.txt文件压缩成zip压缩包当作明文。对blingbling.zip进行明文攻击，如下：



得到\*\*空白文档???'的口令密码:



打开空白文档???'，发现真的是空白的，想吐血。不过查了查百度发现，打开word文件，选择“选项”，然后选择“显示”



在隐藏文字和打印隐藏文件前打对勾，然后确定，然后打开空白文档???, 就能看到flag了!!

### 3、crc32碰撞

CRC32:CRC本身是“冗余校验码”的意思，CRC32则表示会产生一个32bit（8位十六进制数）的校验值。\*\*\*\*

在产生CRC32时，源数据块的每一位都参与了运算，因此即使数据块中只有一位发生改变也会得到不同的CRC32值，利用这个原理我们可以直接爆破出加密文件的内容

例:

[题目下载](#)



# CRC是个校验码

10

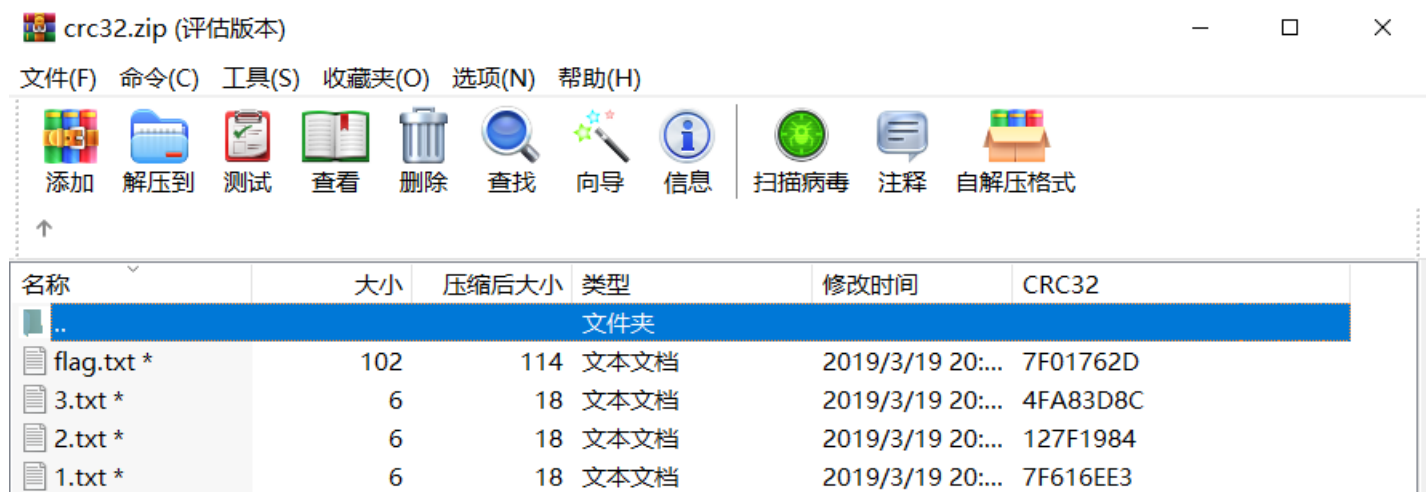
主要就是crc, 贼简单, 但别死脑筋

crc32.zip

Key

SUBMIT

根据题目可知, 这道题可能是crc32碰撞。下载该压缩包, 用winrar打开



会发现4个加密的txt文件, 有三个大小为6的, 一个存放flag大小为102的。\*\*crc32碰撞, 碰撞文件的大小一般不大于6, 大于6的一般碰撞不出来。且位于同一压缩包的文件, 文件密码相同。\*\*所以对1.txt 2.txt 3.txt进行crc32碰撞

还有一款很好用的6位的CRC32爆破

附上神器: <https://github.com/theonlypwner/crc32>

具体使用方法:

```
python crc32.py reverse 你的crc32密文
```

密文记得加上0x变成16进制, 三个txt文件碰撞结果如下

1.txt

```
$ python crc32.py reverse 0x7f616ee3
4 bytes: {0xfc, 0xf3, 0x48, 0x10}
verification checksum: 0x7f616ee3 (OK)
alternative: 06iBmA (OK)
alternative: 2GAaYT (OK)
alternative: 4BH2ir (OK)
alternative: 8LCPd9 (OK)
alternative: AGtKKP (OK)
alternative: Dbvk8f (OK)
alternative: ECiJJ3 (OK)
alternative: Hp2U49 (OK)
alternative: M9CXCK (OK)
alternative: TrCiM1 (OK)
alternative: WoYVfy (OK)
alternative: _e05jQ (OK)
alternative: aHGrpU (OK)
alternative: eLZsq6 (OK)
alternative: k3y2Hh (OK)
alternative: kCECM8 (OK)
alternative: l6lPcW (OK)
alternative: m6paxN (OK)
alternative: pymQwW (OK)
alternative: xo4nzk (OK)
alternative: you_ar (OK)
```

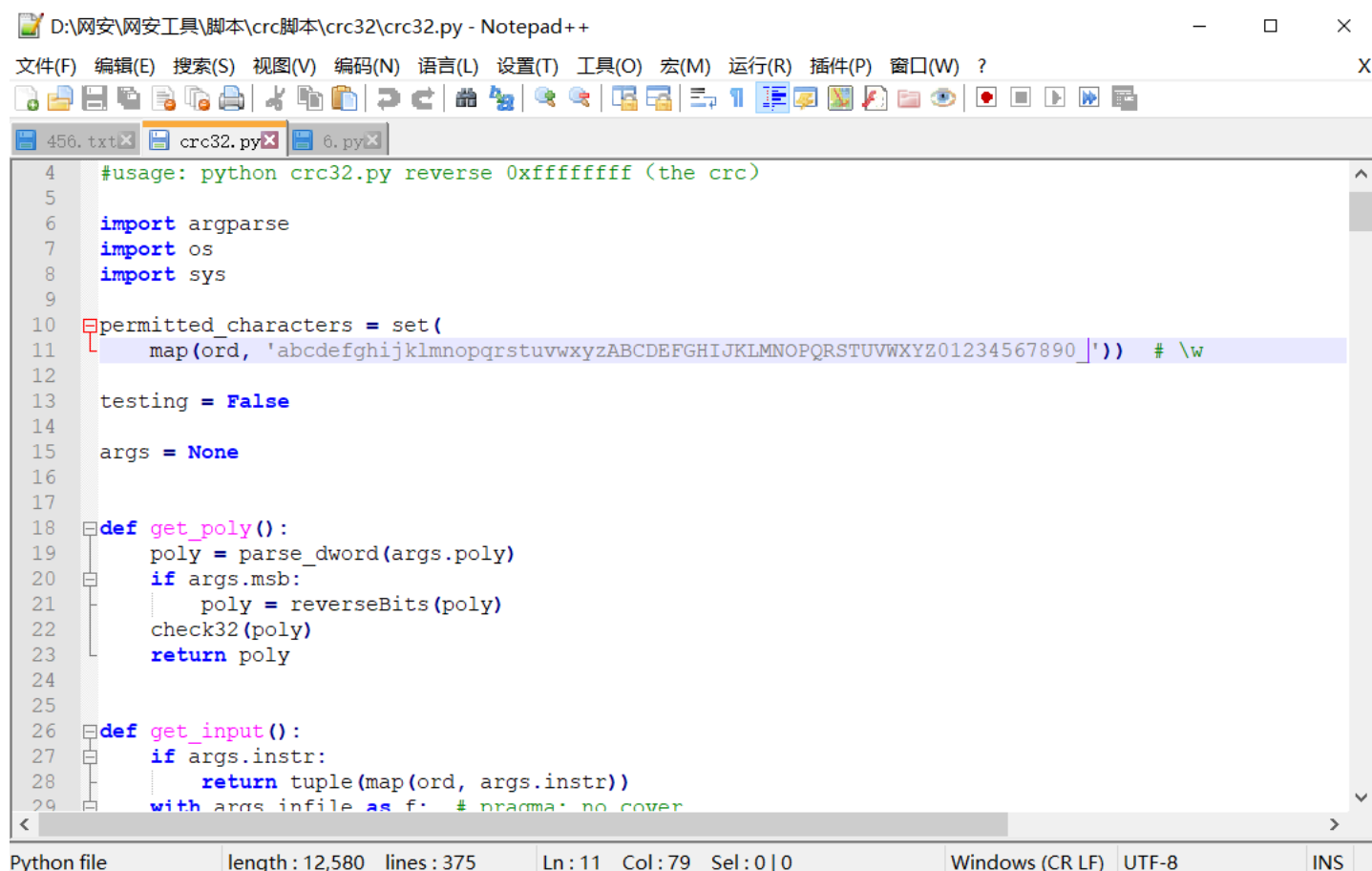
2.txt

```
$ python crc32.py reverse 0x127f1984
4 bytes: {0x0c, 0xa9, 0xe2, 0xfd}
verification checksum: 0x127f1984 (OK)
alternative: lIuEfu (OK)
alternative: 7P3JWG (OK)
alternative: 8_mKpP (OK)
alternative: ATZP_9 (OK)
alternative: K_XabT (OK)
alternative: MZQ2Rr (OK)
alternative: 076Mgs (OK)
alternative: SxjLss (OK)
alternative: TamrYX (OK)
alternative: ZnrBeV (OK)
alternative: bFsV0t (OK)
alternative: cF2gTm (OK)
alternative: e_the_ (OK)
alternative: kPkXYQ (OK)
alternative: l1lfsz (OK)
alternative: n8DEGo (OK)
alternative: oTvYX2 (OK)
alternative: swYuHv (OK)
```

3.txt

```
$ python crc32.py reverse 0x4fa83d8c
4 bytes: {0x7e, 0xfa, 0xeb, 0x0a}
verification checksum: 0x4fa83d8c (OK)
alternative: 0KjFzu (OK)
alternative: 3ka59e (OK)
alternative: AwZr1l (OK)
alternative: CK_lhq (OK)
alternative: DRXRbZ (OK)
alternative: LXN1Nr (OK)
alternative: PFpQ6n (OK)
alternative: Rzu0os (OK)
alternative: UcrqEX (OK)
alternative: a5Dvga (OK)
alternative: bXb8Iy (OK)
alternative: cDlUSt (OK)
alternative: lK2Ttc (OK)
alternative: mKseoz (OK)
alternative: nViZD2 (OK)
alternative: ruFvTv (OK)
```

在碰撞的内容中，找有意义的字符。1.txt中"you\_ar" 2.txt中"e\_the\_" 3.txt中未发现有意义的字符。做到这一步，再次想吐血。还好我用notepad++打开碰撞脚本，发现一组特殊之处



```
D:\网安\网安工具\脚本\crc脚本\crc32\crc32.py - Notepad++
文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P) 窗口(W) ?
456.txt x  crc32.py x  6.py x
4  #usage: python crc32.py reverse 0xffffffff (the crc)
5
6  import argparse
7  import os
8  import sys
9
10 permitted_characters = set(
11     map(ord, 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ01234567890|')) # \w
12
13 testing = False
14
15 args = None
16
17
18 def get_poly():
19     poly = parse_dword(args.poly)
20     if args.msb:
21         poly = reverseBits(poly)
22     check32(poly)
23     return poly
24
25
26 def get_input():
27     if args.instr:
28         return tuple(map(ord, args.instr))
29     with args.infile as f: # pragma: no cover
Python file | length: 12,580 | lines: 375 | Ln: 11 | Col: 79 | Sel: 0 | 0 | Windows (CR LF) | UTF-8 | INS
```

想着是不是因为脚本里缺少特殊字符，而3.txt里有特殊字符。所以碰撞不出3.txt的内容。加上特殊字符后，碰撞结果如下

```
$ python crc32.py reverse 0x4fa83d8c
4 bytes: {0x7e, 0xfa, 0xeb, 0x0a}
verification checksum: 0x4fa83d8c (OK)
alternative: (1XkkR (OK)
alternative: ,hEjj1 (OK)
alternative: .83Y7h (OK)
alternative: /8rh, q (OK)
alternative: /u_UAy (OK)
alternative: 0KjFzu (OK)
alternative: 3ka59e (OK)
alternative: 3w.i8q (OK)
alternative: 6R, IKG (OK)
alternative: ;, ZkXE (OK)
alternative: ;awV5M (OK)
alternative: ?ejW4. (OK)
alternative: AwZr1l (OK)
alternative: CK_lhq (OK)
alternative: DRXRbZ (OK)
alternative: ENV?XW (OK)
alternative: H, oAJA (OK)
alternative: I, .pQX (OK)
alternative: I0a, PL (OK)
alternative: LXN1Nr (OK)
alternative: M(3qP; (OK)
alternative: N5}N{s (OK)
alternative: PFpQ6n (OK)
alternative: QgopD; (OK)
alternative: Rzu0os (OK)
alternative: U. _L(P (OK)
alternative: UcrqEX (OK)
alternative: XP)n;R (OK)
alternative: Zl, pb0 (OK)
alternative: a5Dvga (OK)
alternative: bXb8Iy (OK)
alternative: best!! (OK)
alternative: cD1USt (OK)
alternative: f, CHMJ (OK)
alternative: f}!)!V (OK)
alternative: lK2Ttc (OK)
alternative: mKseoz (OK)
alternative: nViZD2 (OK)
alternative: owv{6g (OK)
alternative: ruFvTv (OK)
alternative: z. 2t4B (OK)
alternative: z2} (5V (OK)
```

找到3.txt中的有意义的字符了，好开心!!! 3.txt中的有意义字符“best!!”，结合1.txt的“you\_ar” 2.txt的“e\_the\_”得到flag.txt的密码：“you\_are\_the\_best!!”，输入密码打开后发现

flag.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

你在找这个吗？好像被加密了呢，先解密一下吧。

ZmxhZyU3QndIMWNvbWUIMjB0byUyMHNoYWxvdSUyMGFucXVhbiU3RA==

很明显是Base64加密的，所以Base64解密，得到

加密/解密   AES加密/解密   DES加密/解密   RC4加密/解密   Rabbit加密/解密   TripleDes加密/解密   MD5加密/解密   Base64加密/解密   Hash加密/解密   JS 加密   JS 解密

flag%7Bwe1come%20to%20shalou%20anquan%7D   ZmxhZyU3QndIMWNvbWUIMjB0byUyMHNhYm9keSUYMGFucXVhbiU3RA==

< 解密   加密 >

看起来解密结果符合url编码，所以url解码，得到最终flag如下：

Native/Unicode   Native/UTF-8   Native/ASCII   URL转码

Url: flag{we1come to shalou anquan}

编码结果: flag%7Bwe1come%20to%20shalou%20anquan%7D

encodeURI  
 encodeURIComponent

URL编码 >

< URL解码

## 4、爆破/字典/掩码攻击

把这三种归位一类是因为这三种方法在本质上都是逐个尝试，只不过待选密码的集合不同

爆破：顾名思义，逐个尝试选定集合中可以组成的所有密码，知道遇到正确密码

字典：字典攻击的效率比爆破稍高，因为字典中存储了常用的密码，因此就避免了爆破时把时间浪费在脸滚键盘类的密码上

掩码攻击：如果已知密码的某几位，如已知6位密码的第3位是a，那么可以构造 ??a??? 进行掩码攻击，掩码攻击的原理相当于构造了第3位为a的字典，因此掩码攻击的效率也比爆破高出不少

例：

[题目下载](#)

Challenge

0 Solves



## 听说你们喜欢暴力破解

10

怼吧，拿出吃奶的劲怼

flag.zip

Key

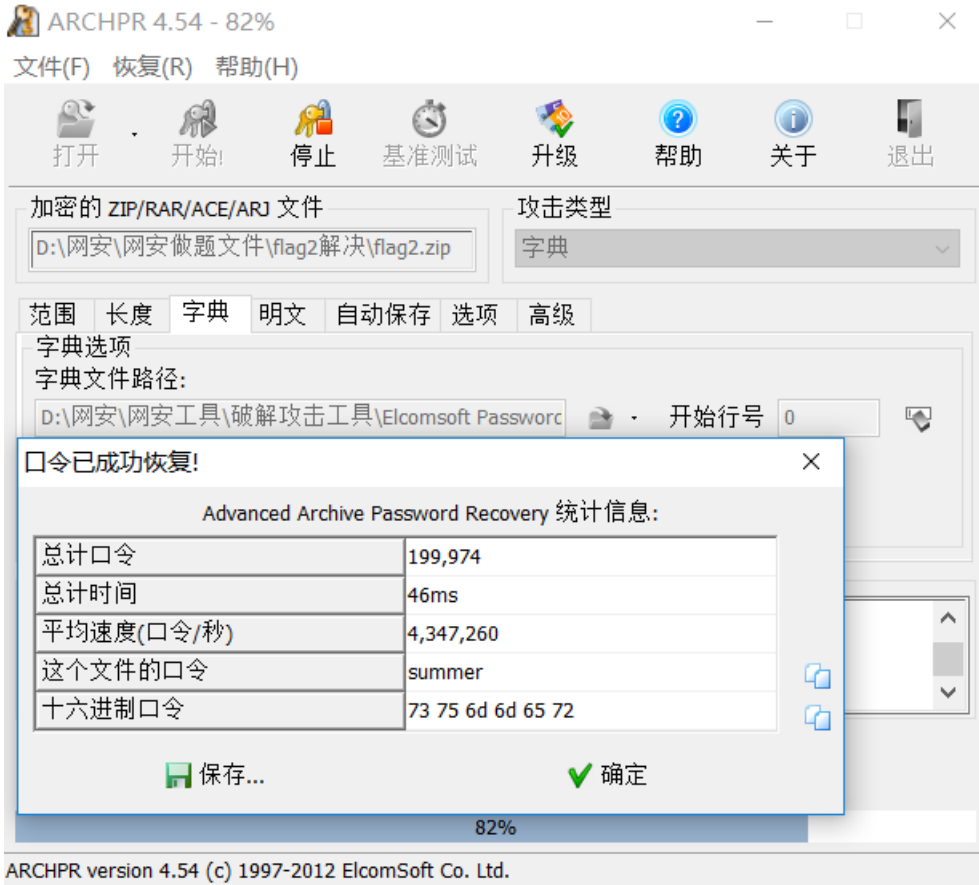
SUBMIT

用winrar打开发现一个加密文件



所以应该要暴力破解，把下载的压缩包放进ARCHPR(爆破工具)里，选择字典攻击(字典攻击比暴力攻击用时少)，

得出文件密码口令



输入密码，解压文件，得到一个文件夹。然后文件夹里有好几个文件夹，所以开启找可疑之处的历程  
最终，我发现有个地方比较可疑

```
function checknum(num) {
  if(num.length=="")
  {
    document.getElementById("usernum").innerHTML="<font color = 'red' >请输入账号</font>"
    myform.num.focus();<!--666c61677b73756d6d6572217d-->
    return false;
  }
  else{
```

看着比较像Base16加密，所以进行Base16解密，得到flag

## Base16编码解码

[666c61677b73756d6d6572217d](#)

编码 解码

flag(summer!)

## 5、杂类

例1:

[题目下载](#)

Challenge 0 Solves

### 简单zip

15

贝斯说：他们家很讲究规定  
提交格式SL{}

zip.zip

Key

SUBMIT

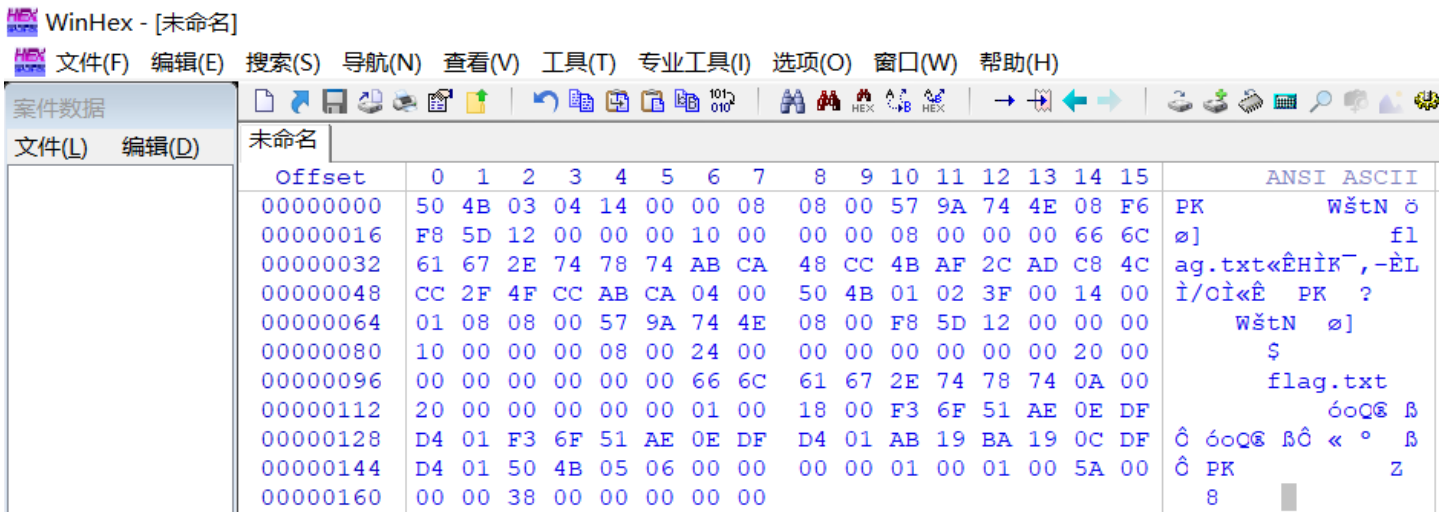
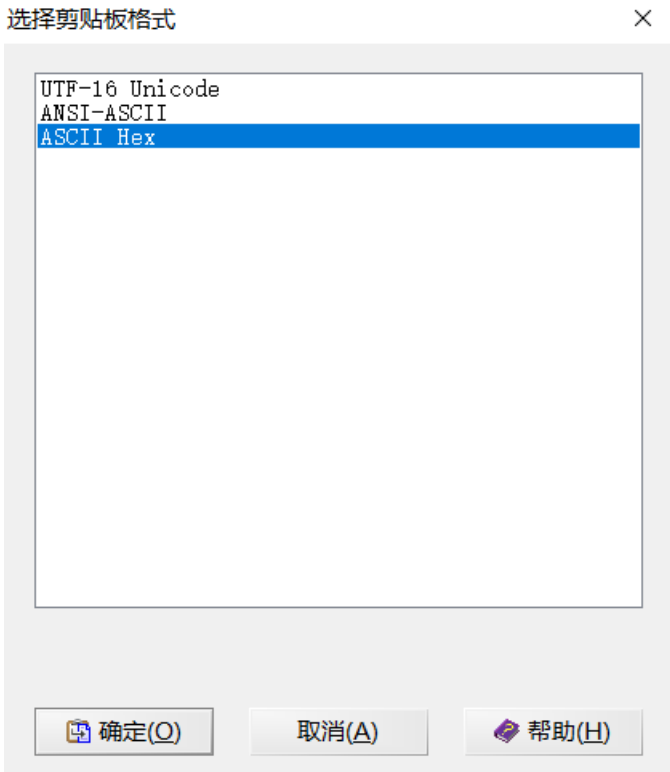
很明显，这个题并没有给出有效提示(只给了提交格式)。下载后，用winrar打开发现一个zip.txt文件，打开后

```
zip.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
504B0304140000080800579A744E08F6F85D120000001000000008000000666C61672E747874ABCA48CC4BAF2CAD84CC
C2F4FCCABCA0400
5:4B01023F00140001080800579A744E08\
\F85D120000001000000008002400000000000000200000000000000666C61672E7478740A0020000000000001001800F3
6F51AE0EDFD401F36F51AE0EDFD401AB19BA190CDFD401504B05060000000010001005A000000380000000000
```

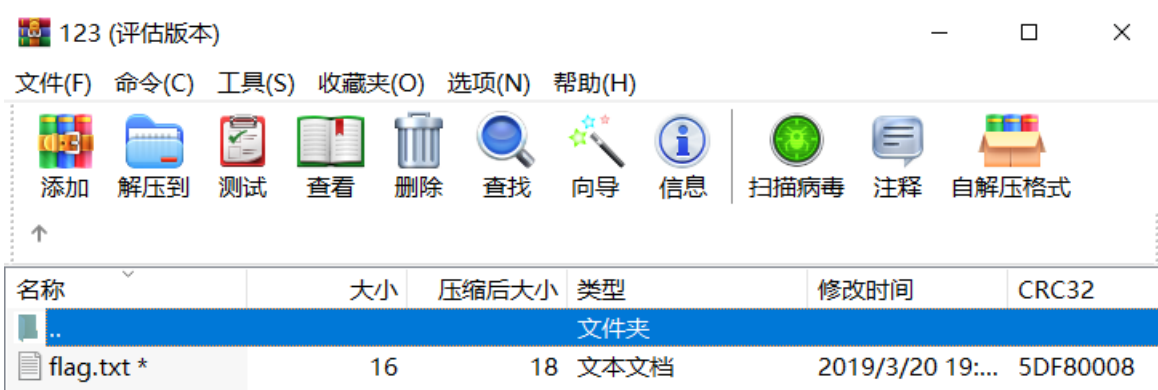
根据504B0304可以看出这应该是一个压缩包的16进制编码，然后查询一下压缩包16进制编码格式，把不符合16进制编码的改成对应的16进制编码。改完后，把16进制编码粘贴在winhex里进行对压缩包的恢复

注意选择16进制粘贴格式

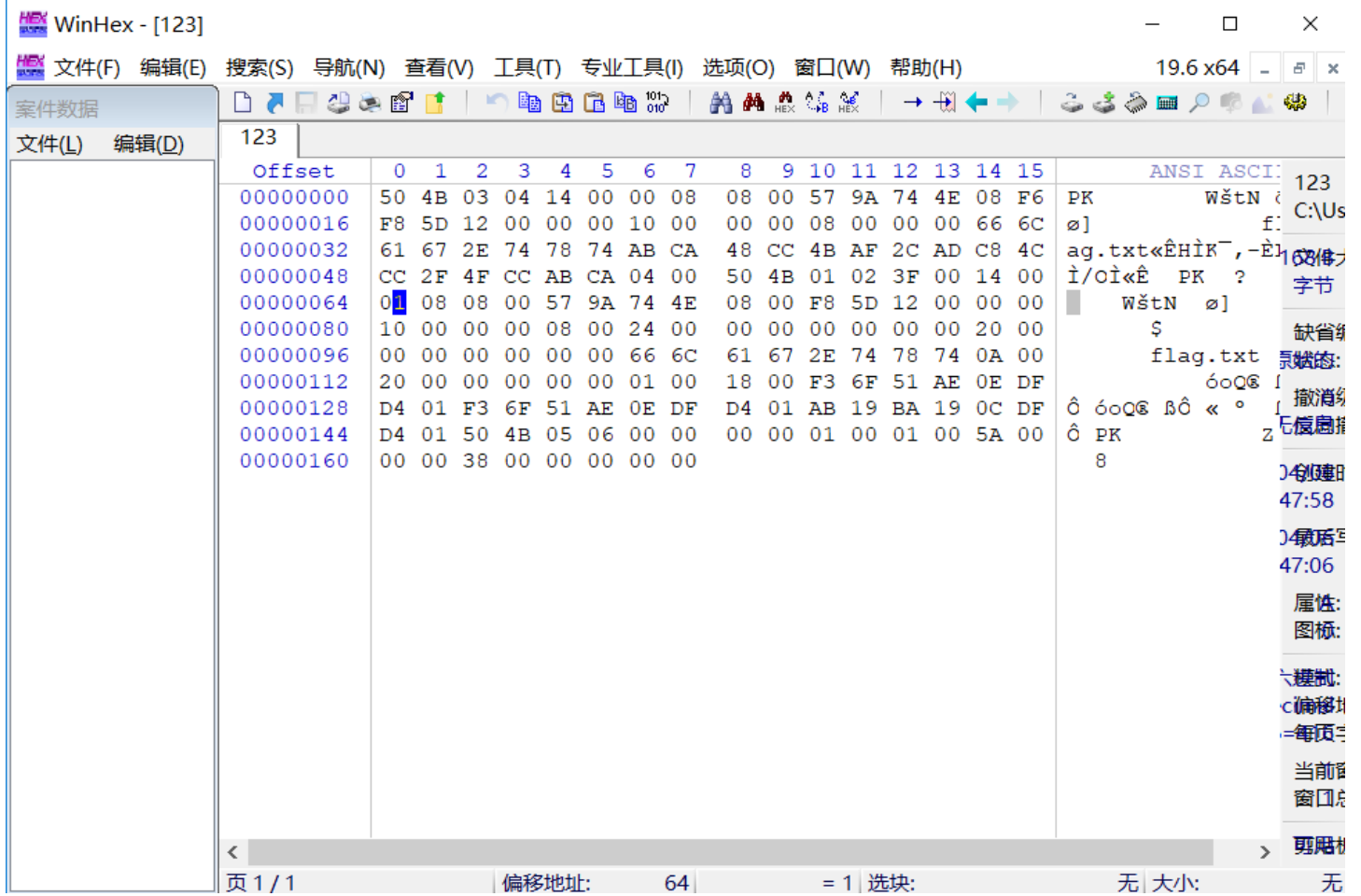




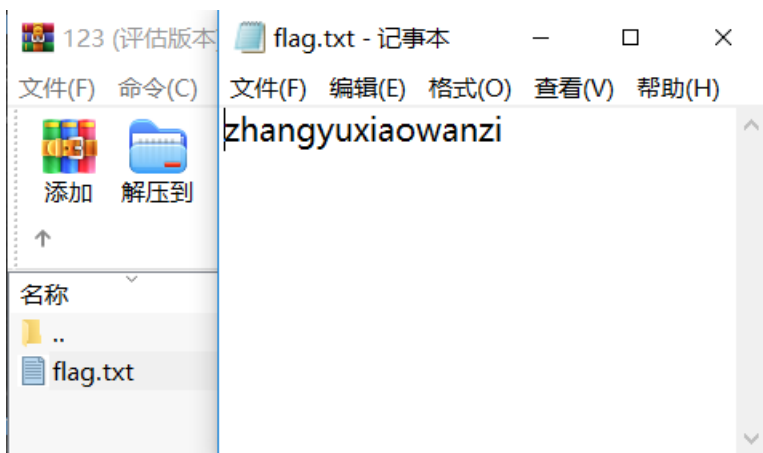
然后保存，用winrar打开恢复后的压缩包，结果发现一个flag.txt文件竟然是加密的，再次想吐血！



也没给什么提示，想着是不是伪加密，所以我用winhex打开恢复后的压缩包，发现果然是伪加密



把01改成00后保存，再次用winrar打开，发现flag.txt变成了未加密，开心！找到flag了！！



例2:

[题目下载](#)

Challenge

0 Solves



## 有人要求我在来一道

15

Give you color see see?真皮沙发了解一下?

了解过凯撒就行

$f(x)=(x-n)\%26+97$

zip

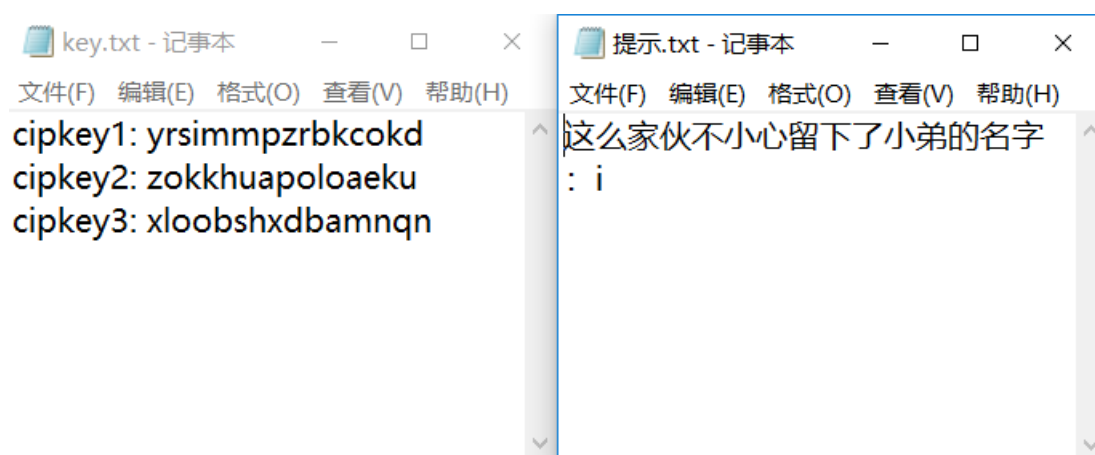
Key

SUBMIT

这个题给了提示，一个公式 $f(x)=(x-n)\%26+97$ ，看着有点像凯撒公式。下载后，用winrar打开压缩包，进入“皮”文件夹发现两个txt文件

[外链图片转存失败(img-BfScdxNr-1563265524323)(<http://i1.fuimg.com/690374/5a1c93aa0a869757.png>)]

打开两个txt文件



key.txt应该就是凯撒加密的密文了，而提示.txt可能是密钥。然后，我开始理解题目提示的凯撒公式 $f(x)=(x-n)\%26+97$ ，我理解这是个加密公式， $f(x)$ 是密钥， $x$ 是明文， $n$ 是密文。然后我写出对应的解密公式 $str=((c1-97)+(key-97))\%26+97$ ，\*\*\*str是明文，c1是密文，key是密钥。\*\*\*由于写这篇blog时，我还不会写python脚本，于是我写了个c语言的

```

#include <stdio.h>
#include <stdlib.h>

int main()
{
    int j;
    char key[20];
    char c1[20]="yrsimppzrbkcokd",c2[20]="zokkhuapoloaeku",c3[20]="x1oobshxdbamqn",str1[20],str2[20],str3[20];
    for(j=1;;j++)
    {
        printf("请输入密钥:");scanf("%s",key);
        printf("密文1:%s\n密文2:%s\n密文3:%s\n",c1,c2,c3);
        //printf("密文1:%s",c1);//scanf("%s",c1);
        //printf("密文2:%s",c2);//scanf("%s",c2);
        // printf("密文3:%s",c3);//scanf("%s",c3);
        printf("明文1:");
        for(int i=0;i<strlen(key);i++)
        {
            str1[i]=(c1[i]-97+key[i]-97)%26+97;
            printf("%c",str1[i]);
        }
        printf("\n");
        printf("明文2:");
        for(int i=0;i<strlen(key);i++)
        {
            str2[i]=(c2[i]-97+key[i]-97)%26+97;
            printf("%c",str2[i]);
        }
        printf("\n");
        printf("明文3:");
        for(int i=0;i<strlen(key);i++)
        {
            str3[i]=(c3[i]-97+key[i]-97)%26+97;
            printf("%c",str3[i]);
        }
        printf("\n\n");
        printf("a b c d e f g h i j k l m n o p q r s t u v w x y z");
        printf("\n");
        printf("1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26");
        printf("\n");
        printf("26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1");
        printf("\n\n");
    }
    return 0;
}

```

输入第一个密钥，运行结果如下

D:\网安\网安做题文件\zip2凯撒解决\c语言代码\bin\Debug\123.exe

```
请输入密钥:i
密文1:yrsimmpzrbkcokd
密文2:zokkhuapoloaeku
密文3:xloobshxdbamnqn
明文1:g
明文2:h
明文3:f

a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

请输入密钥:_
```

然后想到明文3前四个字母应该就是f、l、a、g，而密文3前四个字母是x、l、o、o。所以第二个密钥应该是 $f(x)=(l-l)\%26+97=a$ ;

第三个密钥应该是 $f(x)=(a-o)\%26+97=m$ ;第四个密钥应该是 $f(x)=(g-o)\%26+97=s$ ;

```
请输入密钥:iams
密文1:yrsimmpzrbkcokd
密文2:zokkhuapoloaeku
密文3:xloobshxdbamnqn
明文1:gree
明文2:howc
明文3:flag

a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```

然后明文1第五个字母应该是t，密文1第五个字母是m。第五个密钥应该是 $f(x)=(t-m)\%26+97=g+1=h$ ;

注意:

明文-密文=正值，从前往后数，密钥=正值(即英文字母序号)+1;

明文-密文=负值，从后往前数，密钥=负值的绝对值(即26-英文字母序号);

找到密钥序号对应的字母

明文-密文=零，密钥=a;

```
请输入密钥:iamsh
密文1:yrsimmpzrbkcokd
密文2:zokkhuapoloaeku
密文3:xloobshxdbamnqn
明文1:great
明文2:howco
明文3:flagi

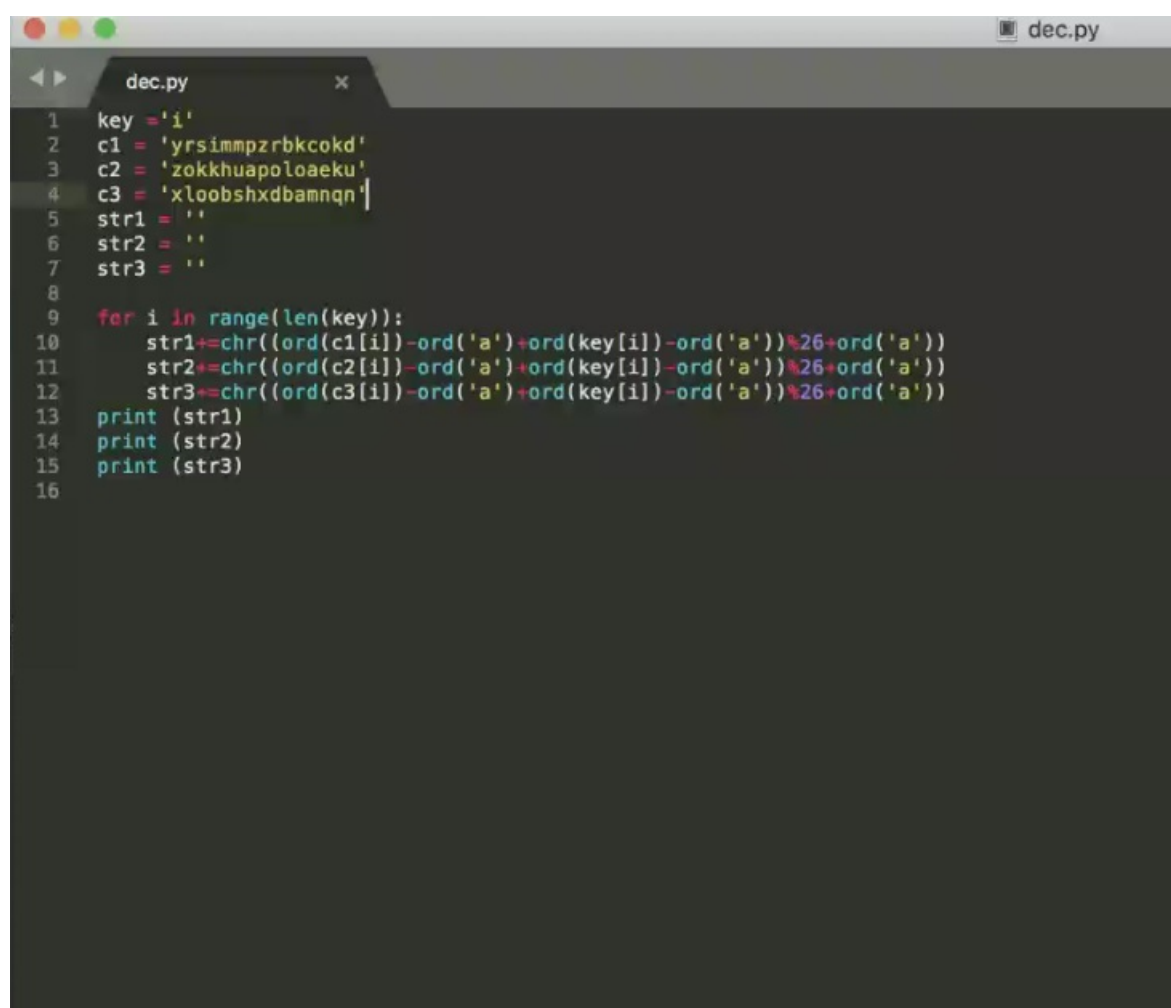
a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```

不断找寻密钥、明文、密文间的规则，最后得出全部密钥和最终flag

```
请输入密钥:iamshalouhacker
密文1:yrsimmpzrbkcokd
密文2:zokkhuapoloaeku
密文3:xloobshxdbamnqn
明文1:greatmanlikeyou
明文2:howcouldisocool
明文3:flagisslxiaoxue

a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```

下面也有个别人写好得python脚本，果然写python脚本才更简洁、更容易。我要努力学python了！！



```
dec.py
1 key = 'i'
2 c1 = 'yrsimmpzrbkcokd'
3 c2 = 'zokkhuapoloaeku'
4 c3 = 'xloobshxdbamnqn'
5 str1 = ''
6 str2 = ''
7 str3 = ''
8
9 for i in range(len(key)):
10     str1+=chr((ord(c1[i])-ord('a')+ord(key[i])-ord('a'))%26+ord('a'))
11     str2+=chr((ord(c2[i])-ord('a')+ord(key[i])-ord('a'))%26+ord('a'))
12     str3+=chr((ord(c3[i])-ord('a')+ord(key[i])-ord('a'))%26+ord('a'))
13 print (str1)
14 print (str2)
15 print (str3)
16
```

以上便是我做CTF压缩包类型题的总结，总结许多收获也许多。小白进阶ing!!!