

# 实验吧——安全杂项（四）

原创

FunkyPants 于 2017-11-05 12:34:10 发布 2092 收藏 1

分类专栏: [Python CTF writeup](#) 文章标签: [安全](#) [解密](#) [脚本](#)

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/FunkyPants/article/details/78448725>

版权



[Python](#) 同时被 2 个专栏收录

5 篇文章 0 订阅

订阅专栏



[CTF writeup](#)

13 篇文章 0 订阅

订阅专栏

## 0.说明

每五个题目写作一篇writeup, 第一行对应解题笔记(一).....

无人能解 我已解开 注: 第1-3名解开题目额外加分20、10、5

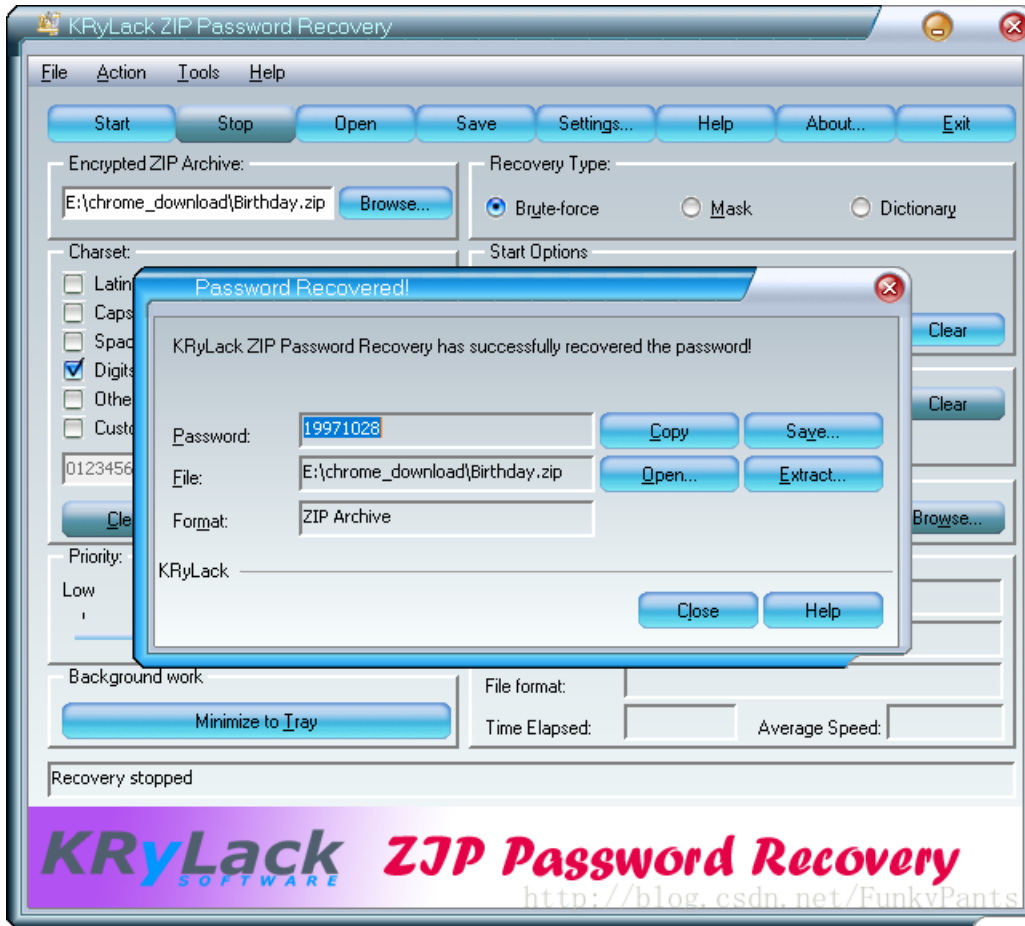
损坏的U盘镜像	藏在图片中的秘密	spaceport-map	pilot-logic	deeeeeeeeeaaaadb...
这就是一个坑	紧急报文	CheatEngine	flag.xls	图片里的动漫
Canon	ROT-13变身了	解码磁带	功夫秘籍	WTF?
社交网络	有趣的文件	Paint&Scan	64格	异性相吸
A记录	Snake	SOS	绕	理查德

下一页

<http://blog.csdn.net/FunkyPants>

## 1.社交网络

感觉自己的脑洞还是太小, 一开始没有想到Birthday是什么意思, 后来想到生日应该是8位数字, 于是重新爆破。在这个过程中大胆猜测社交网络上的人生日应该在1900年之后, 于是从这个数字开始爆破。



```

00000850 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000860 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000870 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000880 | 00 00 00 00 00 00 00 00 | 00 00 00 00 53 4F 4E 59 |
00000890 | 00 00 32 30 31 35 3A 31 | 30 3A 32 38 20 31 36 3A |
000008A0 | 30 39 3A 32 36 00 66 6C | 61 67 7B 68 69 5F 6D 79 |
000008B0 | 5F 66 72 69 33 6E 64 7A | 7D 00 00 05 90 03 00 02 |
000008C0 | 00 00 00 14 00 00 10 A6 | 90 04 00 02 00 00 00 14 |
000008D0 | 00 00 10 92 92 91 00 02 | 00 00 00 03 36 32 00 00 |
000008E0 | 92 92 00 02 00 00 00 03 | 36 32 00 00 EA 1C 00 07 |
000008F0 | 00 00 07 B4 00 00 08 DE | 00 00 00 00 1C EA 00 00 |
00000900 | 00 08 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000910 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000920 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000930 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000940 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |
00000950 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 |

```

```

SONY
2015:10:28 16:
09:26 lag{hi_my
_fr13ndz}
|
' ' ' 62
'' 62 é
' P é

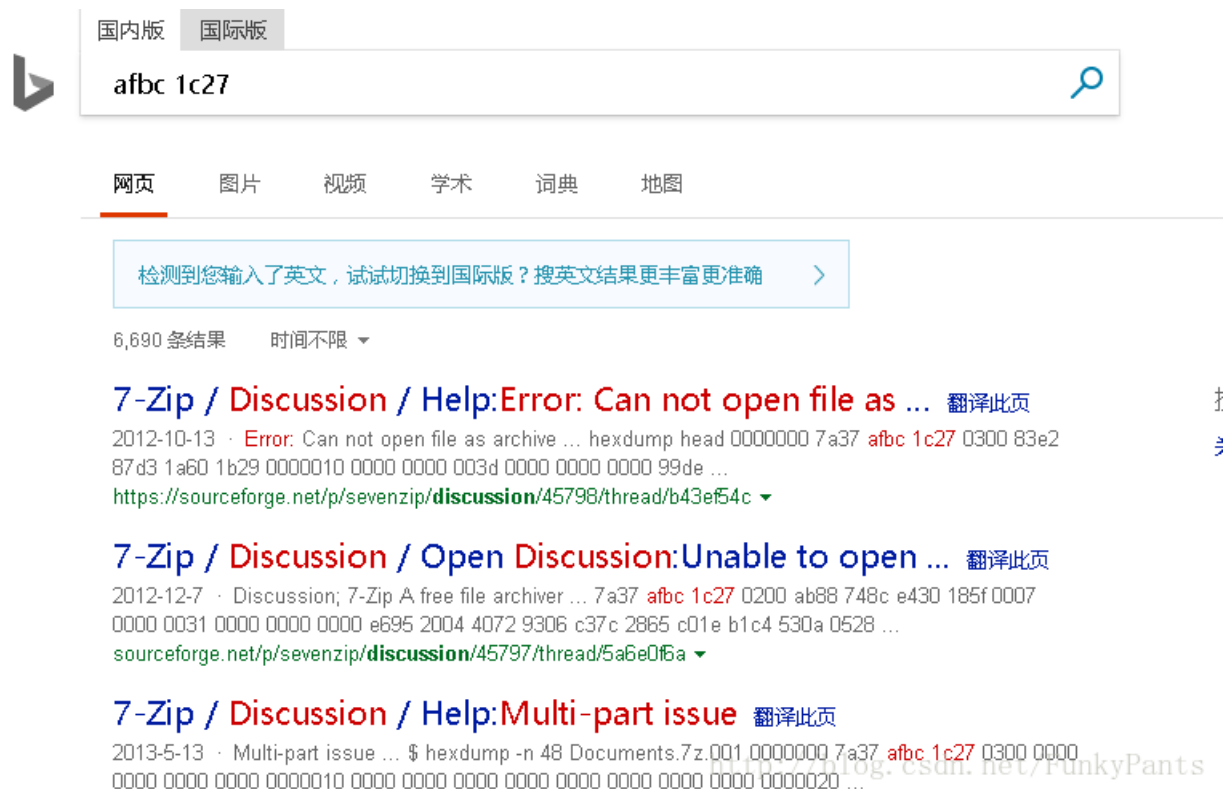
```

<http://blog.csdn.net/FunkyPants>

## 2.有趣的文件

看了大佬的writeup之后有了思路，于是自己手动写了一个脚本解密，但是其中走了很多弯路.....

直接复制前面8个十六进制字符去搜索的话，会发现这很像7z压缩包的文件头，但是每两个字节之间需要换下位置（我之前并不知道7z的格式是什么样的.....）。图一是错误的格式，图二是正确的格式



The screenshot shows a search engine interface with the search term "afbc 1c27". The results list three entries related to 7-Zip discussions on SourceForge:

- 7-Zip / Discussion / Help:Error: Can not open file as ...** (2012-10-13) - Error: Can not open file as archive ... hexdump head 0000000 7a37 afbc 1c27 0300 83e2 87d3 1a60 1b29 0000010 0000 0000 003d 0000 0000 0000 99de ...  
<https://sourceforge.net/p/sevenzip/discussion/45798/thread/b43ef54c>
- 7-Zip / Discussion / Open Discussion:Unable to open ...** (2012-12-7) - Discussion; 7-Zip A free file archiver ... 7a37 afbc 1c27 0200 ab88 748c e430 185f 0007 0000 0031 0000 0000 0000 e695 2004 4072 9306 c37c 2865 c01e b1c4 530a 0528 ...  
[sourceforge.net/p/sevenzip/discussion/45797/thread/5a6e0f6a](https://sourceforge.net/p/sevenzip/discussion/45797/thread/5a6e0f6a)
- 7-Zip / Discussion / Help:Multi-part issue** (2013-5-13) - Multi-part issue ... \$ hexdump -n 48 Documents.7z.001.0000000 7a37 afbc 1c27 0300 0000 0000 0000 0000 0000010 0000 0000 0000 0000 0000 0000 0000 00000020 ...  
<https://blog.csdn.net/FunkyPants>



The screenshot shows a search engine interface with the search term "377a bcaf 271c". The results list two entries:

- 7-Zip / Discussion / Help:Cobian 11 problem** (2015-4-21) - Cobian 11 problem ... 377a bcaf 271c 0003 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 81c0 b634 1568 92c3 c185 9e75 c7b5 ...  
[sourceforge.net/p/sevenzip/discussion/45798/thread/c0838d18](https://sourceforge.net/p/sevenzip/discussion/45798/thread/c0838d18)
- 377a bcaf 271c 0004 9b97 92e0 0a2f 0000 0000 0000 ...**  
377a bcaf 271c 0004 9b97 92e0 0a2f 0000. 0000 0000 5a00 0000 0000 0000 4624 bf06. e08e b42f 025d 003d e18c d2e2 bb89 d288. f2b7 e69c 654f 5a88 945a 1b67 b56e 18e0. <https://blog.csdn.net/FunkyPants>  
<https://pastebin.com/1BUJqZQv>

于是我写了一个脚本，去除行号，去除换行符，并且在4个字符中1号2号位置的字符和3号4号位置字符交换位置。最终得到了正确的十六进制数据。

```

with open('flag.txt', 'wb') as f_7z:
    f_7z.write('377a '.encode('utf-8'))
    with open('funfile.txt', 'r') as f:
        while True:
            line = f.readline()
            if not line:
                break
            #去除行号, 去除换行符
            line_strip = line.replace(line.split(' ')[0]+' ', '')
            line_reverse = ''
            for i in line_strip.split(' '):
                line_reverse += i[2:4] + i[0:2] + ' '
            if line != '0018af0':
                f_7z.write(line_reverse.encode('utf-8'))

```

但是因为我不知道怎么把十六进制数据写入文件并让它也存储为十六进制数据，所以在得到十六进制的数据后，使用winhex来帮我完成接下来的工作。

新建一个txt文档，直接拖入winhex中准备写入数据。

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI	ASCII
00000000																		

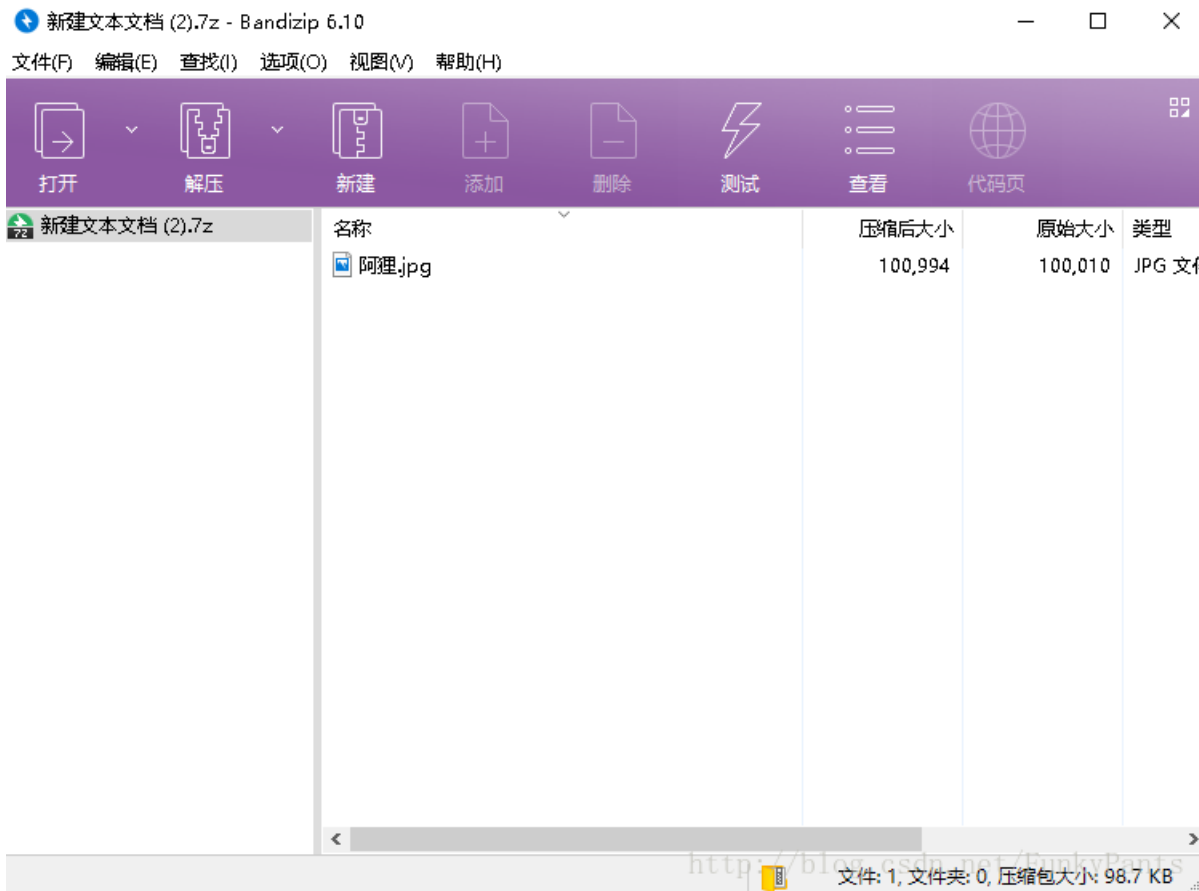
<http://blog.csdn.net/FunkyPants>

复制上个程序中得到的十六进制数据



粘贴到winhex中，选择ASCII hex模式粘贴（这里的图没截到.....）

之后保存文件，手动将文件后缀改为7z，可以打开压缩包，将里面的阿狸图片再拖到winhex中可以看到base64编码的flag。



### 3.Paint&Scan

因为对于图片隐写有一定的基础，看到点的坐标时就知道这个题目需要通过点的坐标画图还原FLAG，一开始还以为可以直接得到FLAG字符串的，结果还需要扫个二维码.....

Python的PIL库简单易用(在Python3中对应库为Pillow)，在网上随便找了篇画图教程之后就知道了其原理，个人觉得稍难一点的地方是还需要调用这个库去新建一个图片。可以参考官方文档：<http://www.effbot.org/imagingbook/image.htm>

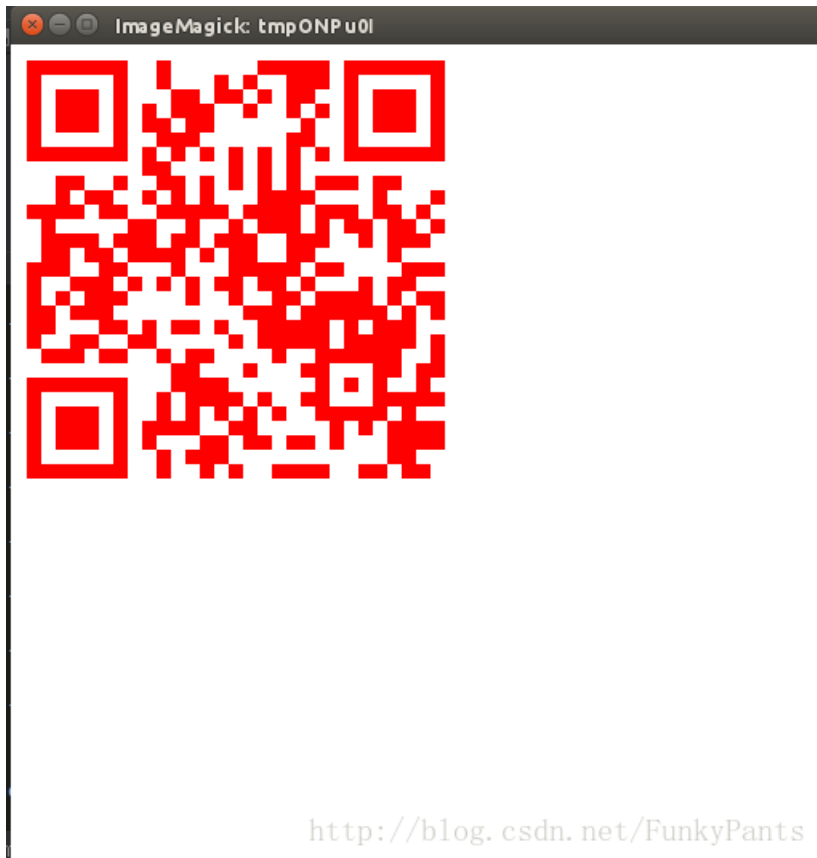
最终的代码如下：

```
from PIL import Image, ImageDraw

#fp = open('flag.jpg', 'rb')
im = Image.new("RGB", (512, 512), "white")
#im.save('flag.jpg', 'jpg')
draw = ImageDraw.Draw(im)
with open('Paint&Scan.txt', 'r') as f:
    while True:
        line = f.readline().strip()
        print line
        if not line:
            break
        #draw.point()
        exec 'draw.point(' + line + ', fill = (255, 0, 0))'
        print 'draw.point(' + line + ', fill = (255, 0, 0))'

im.show()
```

得到结果如下图所示，扫码可得FLAG。



## 4.64格

因为这个题目做完之后没及时写Writeup，所以忘了截图。

打开所给GIF图片后不能显示，放到winhex里面发现文件头错误，修改文件头为GIF89a，可以看到一张19帧的活蹦乱跳的小黄人GIF，放入逐帧分解的软件中（我用的PS），可以看到每一帧中小黄人的位置，将小黄人对应位置用六十四卦幻方配数图对应为相应数字，再用Base64索引表解码得到base64字符串Q1RGe2FiY19kZWZfZ30，解码得flag（这两种编码方法是结合题目名称64在网上搜索到的）。

## 5.异性相吸

下载后得到一个明文和一个密文，并且都是32位的，同时题目难度为“易”，猜测可能是进行了一些简单的位运算，又因为题目名称为“异性相吸”，猜测可能是进行了异或，使用以下Python脚本解密

```
f_cry = open('密文.txt', 'r')
f_txt = open('明文.txt', 'r')

line_cry = f_cry.read()
line_txt = f_txt.read()

for i in range(0, len(line_txt)):
    print(chr(ord(line_txt[i]) ^ ord(line_cry[i])), end='')
```

这个程序打印出来的结果提交后并不正确，展开脑洞把后面都写成biubiubiu后成功.....