

# DDCTF-2018-writeup(5misc)

转载

[weixin\\_30340819](#) 于 2018-04-20 11:40:00 发布 166 收藏 1

文章标签: [python](#) [操作系统](#) [密码学](#)

原文链接: <http://www.cnblogs.com/kagari/p/8889412.html>

版权

打了好几天最后也只是80多名,我好菜啊.jpg

(ノ ◡ ◡) ◡ ◡ ㄣ

## 掀起了没技术的桌子

0x00 (ノ ◡ ◡) ◡ ◡ ㄣ

题目:

(ノ ◡ ◡) ◡ ◡ ㄣ

d4e8e1f4a0f7e1f3a0e6e1f3f4a1a0d4e8e5a0e6ece1e7a0e9f3baa0c4c4c3d4c6fbb9e1e6b3e3b9e4b3b7b7e2b6

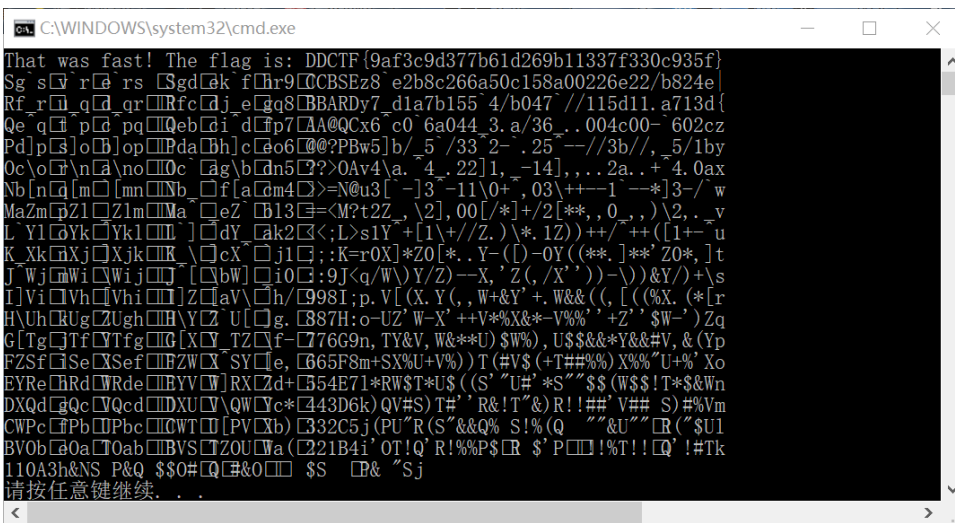
一开始被这个(ノ ◡ ◡) ◡ ◡ ㄣ误导以为是jencode, 尝试了好久没啥结果, 后面又有人说是翻转, 试了很久还是没结果。然后跑了下移位密码获取flag

贴上脚本

```

1
s='d4e8e1f4a0f7e1f3a0e6e1f3f4a1a0d4e8e5a0e6ece1e7a0e9f3baa0c4c4c3d4c6fbb9e1e6b3e3b9e4b3b7b7e2b6b1e4b2b6b9e2b
1b1b3b3b7e6b3b3b0e3b9b3b5e6fd'
2 '''
3 s1=''
4 for x in range(len(s)/2):
5     s1+=chr((int(s[x*2:x*2+2],16))%128)
6 print s1
7 '''
8
9 for j in range(20):
10    s1=''
11    for x in range(len(s)/2):
12        s1+=chr((int(s[x*2:x*2+2],16)-j)%128)
13    print s1
14

```



## 0x01 第四扩展FS

题目:D公司正在调查一起内部数据泄露事件, 锁定嫌疑人小明, 取证人员从小明手机中获取了一张图片引起了怀疑。这是一道送分题, 提示已经在题目里, 日常违规审计中频次有时候非常重要。

<https://pan.baidu.com/s/1DJpMFU2lajHGT00yfzTHVQ> 密码: fpp4

这道题很多坑，拿到题目看题目名字就可以知道它考的是ext4文件，先用压缩软件打开查看了一下



因为没怎么接触过ext4文件系统所以不知道journal文件是拿来干嘛的，百度后说是文件系统日志，所以考虑mount到linux上，但是怎么都不成功。

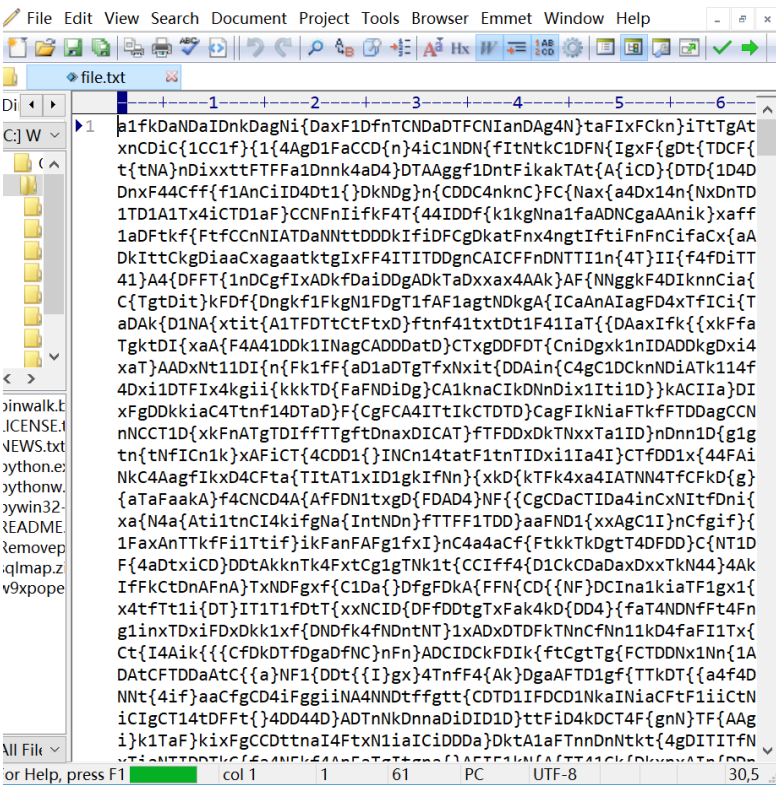
另外还有一个加密的file.txt，用binwalk扫了下文件，发觉就是这两个文件

```
PS C:\WINDOWS\system32> binwalk windows.jpg
* suggest: you'd better to input the parameters enclosed in double quotes.
* *****
DECIMAL      HEXADECIMAL  DESCRIPTION
-----
0            0x0          JPEG image data, JFIF standard 1.01
30          0x1E        TIFF image data, big-endian, offset of first image directory: 8
2825118     0x2B1B9E    Linux EXT filesystem, rev 2.0, ext4 filesystem data, UUID=57f8f4bc-abf4-0000-675f-946fc0f9c0f9
7236510     0x6E6B9E    Zip archive data, encrypted at least v2.0 to extract, compressed size: 18210, uncompressed
size: 30500, name: file.txt
7254812     0x6EB31C    End of Zip archive, footer length: 22
PS C:\WINDOWS\system32>
```

为了寻找file.txt密码，用winhex打开jpg后发现有些图片有敏感信息，然后直接查看图片属性，发现解压密码



解压打开file.txt

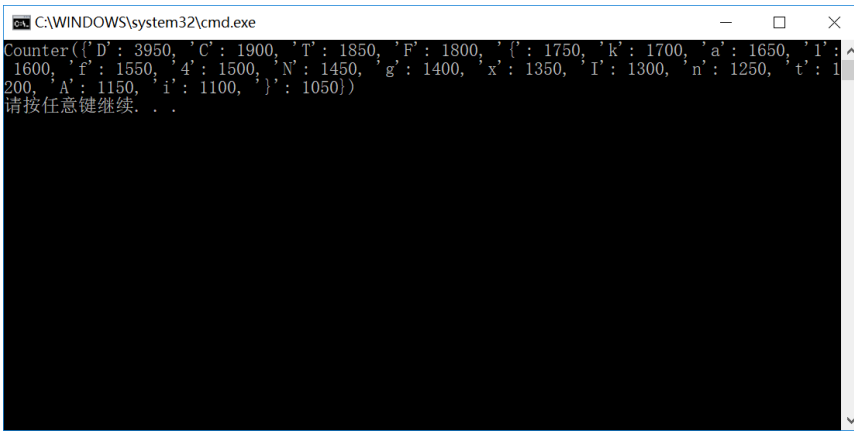


根据提示频率，统计一下字符出现频率，贴上脚本，得到flag

```

1 import collections
2
3 f=open('file.txt','r')
4 print collections.Counter(f.read())

```



## 0x02 流量分析

题目:提示一: 若感觉在中间某个容易出错的步骤, 若有需要检验是否正确时, 可以比较MD5: 90c490781f9c320cd1ba671fcb112d1c

提示二: 注意补齐私钥格式

-----BEGIN RSA PRIVATE KEY-----

XXXXXXX

-----END RSA PRIVATE KEY-----

这题给的数据包很大, 而且有很多坑, 数据包前面有两个用ftp传输加密压缩文件fl-g.zip, sqlmap-dev.zip, 中间有九封邮件, 只到看到最后才会发现本题关键ssl加密通信。

5937	430.593315	172.17.0.4	172.17.0.2	FTP	74	Request: TYPE I
5938	430.593350	172.17.0.2	172.17.0.4	FTP	97	Response: 200 Switching to Binary mode.
5939	430.593384	172.17.0.4	172.17.0.2	FTP	72	Request: PASV
5940	430.593452	172.17.0.2	172.17.0.4	FTP	115	Response: 227 Entering Passive Mode (172,17,0,2)
5941	430.593493	172.17.0.4	172.17.0.2	TCP	74	51223 → 29847 [SYN] Seq=0 Win=29200 Len=0
5942	430.593506	172.17.0.2	172.17.0.4	TCP	74	29847 → 51223 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0
5943	430.593520	172.17.0.4	172.17.0.2	TCP	66	51223 → 29847 [ACK] Seq=1 Ack=1 Win=29200 Len=0
5944	430.593537	172.17.0.4	172.17.0.2	FTP	81	Request: RETR fl-g.zip
5945	430.593624	172.17.0.2	172.17.0.4	FTP	138	Response: 150 Opening BINARY mode data transfer
5946	430.593775	172.17.0.2	172.17.0.4	FTP-DATA	14546	FTP Data: 14480 bytes
5947	430.593803	172.17.0.4	172.17.0.2	TCP	66	51223 → 29847 [ACK] Seq=1 Ack=1 Win=29200 Len=0
6989	443.057849	172.17.0.2	172.17.0.4	FTP	114	Response: 227 Entering Passive Mode (172,17,0,2)
6990	443.057910	172.17.0.4	172.17.0.2	TCP	74	37977 → 57897 [SYN] Seq=0 Win=29200 Len=0
6991	443.057924	172.17.0.2	172.17.0.4	TCP	74	57897 → 37977 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0
6992	443.057939	172.17.0.4	172.17.0.2	TCP	66	37977 → 57897 [ACK] Seq=1 Ack=1 Win=29312 Len=0
6993	443.057969	172.17.0.4	172.17.0.2	FTP	83	Request: RETR sqlmap.zip
6994	443.058068	172.17.0.2	172.17.0.4	FTP	140	Response: 150 Opening BINARY mode data connection
6995	443.058199	172.17.0.2	172.17.0.4	FTP-DATA	14546	FTP Data: 14480 bytes

前面的两个压缩包可能是本题彩蛋, 不过我是真的解不出来。。。所以跳过, 直接导出邮件。

分组	主机名	内容类型	大小	文件名
8314	tools@didichuxing.com	EML file	4844 bytes	anti-spider   fix the hdfs monitor bug (1166).eml
8314	MAILER-DAEMON@7e040dccc799e.didi-ctf.com (Mail Delivery System)	EML file	7044 bytes	Undelivered Mail Returned to Sender.eml
8521	hao@memeect.com	EML file	90 kB	=?UTF-8?B?QUkxMDB5py6SZmo5a2m5Lmg5pel5oqjDlwMtctMTHf
8557		EML file	6028 bytes	.eml
8606		EML file	84 kB	.eml
8697		EML file	6107 bytes	.eml
8767		EML file	6850 bytes	.eml
8846		EML file	5174 bytes	.eml
9285		EML file	756 kB	.eml

将导出的邮件导入qq邮箱中查看, 可以发现这个

????

发件人: (杨睿信息安全部)<zhangsan@didi-ctf2.com>  
时间: 2018年1月11日(星期四) 下午5:56  
收件人: yangruiyangrui<zhangsan@didi-ctf2.com>

小张你好:



你好, 请将密钥安装到服务器上, 谢谢

--B\_3598538194\_2172270964--

--B\_3598538194\_700708737

Content-type: image/png; name="image001.png";  
x-mac-creator="4F50494D";  
x-mac-type="504E4766"  
Content-ID: <image001.png@01D38B05.8079CE40>  
Content-disposition: inline;  
filename="image001.png"  
Content-transfer-encoding: base64

```
iVBORw0KGgoAAAANSUkEUGAABIAAAASCAYAAACu61SmAAAMK61DQ1B1JQ0MgUHHJvZm1sZQAASmVWvdYU8kwnluSkJDQAhGQEnoTpVepoUUQkCrYCEkgocSYEETsyKKCa0HFghVZVFwLYASNiXyWBTs9WfBRVkcZzU3iQBdPV7733vfn+5979nzzp5z7kz880Aob7NEyuzUA0AskU5kpjQqObEpQm6SFAAArIwBM4crhScUB0dASAMvT+p7y7Dr2hXLGXx/q5/b+kJo8v5QKARE0cypNySyE+BADuxhVLCgAg9EC72cwcMcREyB3oSyBBiM310F2JPeQ4VYkjFD5xMSyIUwBQoXI4knQA10S8mLncdBhHbRnEDIKeUARxE8S+XAGHB/FniEd1Z0+HMN0aYuvU7+KK/yNm6nBMDid9GctzUYhKkFAqzuLm+j/L8b8100s2NIYZVKpAEHyjz11et8zp4XJMhfcKDUyCmItiK8KeQp/OX4iKIXFD/p/4EpZsGaaAQBK5XGcwiE2GNhU1BUZMw3TR0GscGGTufjhDnsOGvf1CeZHjMYH83jS4NjhzBHohhL71Msy4wPGIy5RcBnD8VsZBFJSp5opdzhQmREKtBFFeaGRs+6PM8X8CKHPKRyGLknoE/x0CaJRCG6YOZZ0uH8sK8BEJ25CCOyBHEhSn7Y105HAU3XYgz+NKJEUm8efygyVgVeWAFfFD/IHysV5wTGDPpXir0iB/2xJn5WqNxcnGbnDd2qG9vDpxsynxxIM6Jj1Nyw7Uz000i1RxwWxABWCAIMIEmaiQYDjKAsK2nvgd+KVtCAAdIQDrgA/tBy1CPREWLCD5jQT74CyI+kA73C1S08kEutH8Ztiqf9iBN0Zqr6JEJnkCcDcJBFvyWKXqJhkdLAI+hRfjT6FzINQuqv00nG1N9yEYMJgYRw4ghRBtch/fFvfEI+PSH6oR74J5DvL75E54Q2gkPCdcInYRb04QFkh+YM8F40Ak5hgxm1/p9drg1jQk8+I+MD6MjTNwfwCPU8CRANa/OLYrtH7PVTac8bdaDsYi05BR8giyP9n6RwZqtmqw1Hk1fq+fkpeqcPVYg23/JgH67v68eA7/EdPbA12EGvBTmLnsSasHjCx41gd1ood1ePhufFYMTeGRotR8MmEcYQ/jccZHFNeNa1DtUO3w+fBNpDDz8uRLxbwdPEsiTbdkMMMGls1
```

分组 9189, 37 客户端 分组, 8 服务器 分组, 15 turn(s), 点击查看.

这里是本题关键, 在数据包中这里是很大一串base64, 解密后得到rsa私钥



MIICXAIBAAKBgQDCm6vZmc1JrVH1AAyGuCuSSZ80+mIQiOUQCvN0HYbj8153JfSQ  
 LsJIhbRYS7+zZ1oXvPemWQDv/u/tzegt58q4ciNmcVnq1uKiygc6Q0tvT7oiSTy0  
 vMX/q5iE2iClYUIHZEKX3BjjNDxrYvLQzPyGD1EY2DZIO6T45FNKYC2VDwIDAQAB  
 AoGAbtWUKUkx371LFRq7B5sqjZVKdpBZe4tL0jg6cX5Djd3Uhk1inR9UXVNw4/y4  
 QGfzYqOn8+Cq7QSoBysH0eXSiPztW2cL09ktPgSlfTQyN6ELNGuiUOYnaTWYzpp/  
 QbRcZ/eHBu1VQLlk5M6RVs9BLI9X08RA17EcwumiRfWas6kCQQDvqC0dx12wIjwN  
 czILcoWLiG2c2u71Nev9DrWjWHU8eHDuzCJWvOUAHIrkexddWEK2VHd+F13GBCOQ  
 ZCM4prBjAkEAz+ENahsEjBE4+7H1HdIaw0+goe/45d6A2ew0/1YH6dDZTAzTW9z9  
 kzV8uz+Mmo5163/JtvwYQcKF39DJGGtqZQJBAKa18XR16fQ9TFL64EQwTQ+tYBzN  
 +04eTWQcmH3haeQ/0Cd9XyHBUveJ42Be8/jeDcIx7dGLxZKajHbEAfBFnAsCQGq1  
 AnbJ4Z6opJCGu+UP2c8SC8m0bhZJDe1PRC8IKE28eB6SotgP61ZqaVmQ+HLJ1/wH  
 /5pfc3AmEyRdfyx6zwUCQCAH4SLJv/kprRz1a1gx8FR5tj4NeHEFFNEgq1gmiwMH  
 2STT5qZWzQFz8NRe+/otNOHBR2Xk4e8IS+ehIJ3TvyE=

o.	Time	Source	Destination	Protocol	Length	Info
10166	3791071.882991	172.17.0.3	172.17.0.2	TCP	74	50620 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=
10167	3791071.883018	172.17.0.2	172.17.0.3	TCP	74	443 → 50620 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_P
10168	3791071.883037	172.17.0.3	172.17.0.2	TCP	66	50620 → 443 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=725461496 TSec
10169	3791071.939763	172.17.0.3	172.17.0.2	TLSv1.2	313	Client Hello
10170	3791071.939842	172.17.0.2	172.17.0.3	TCP	66	443 → 50620 [ACK] Seq=1 Ack=248 Win=30080 Len=0 TSval=725461553 TS
10171	3791071.940104	172.17.0.2	172.17.0.3	TLSv1.2	680	Server Hello, Certificate, Server Hello Done
10172	3791071.940146	172.17.0.3	172.17.0.2	TCP	66	50620 → 443 [ACK] Seq=248 Ack=615 Win=30464 Len=0 TSval=725461553
10173	3791071.941154	172.17.0.3	172.17.0.2	TLSv1.2	205	Client Key Exchange
10174	3791071.980496	172.17.0.2	172.17.0.3	TCP	66	443 → 50620 [ACK] Seq=615 Ack=387 Win=31104 Len=0 TSval=725461594
10175	3791071.980539	172.17.0.3	172.17.0.2	TLSv1.2	117	Change Cipher Spec, Finished
10176	3791071.980545	172.17.0.2	172.17.0.3	TCP	66	443 → 50620 [ACK] Seq=615 Ack=438 Win=31104 Len=0 TSval=725461594
10177	3791071.980721	172.17.0.2	172.17.0.3	TLSv1.2	308	New Session Ticket, Change Cipher Spec, Finished
10178	3791071.981130	172.17.0.3	172.17.0.2	HTTP	169	GET / HTTP/1.1
10179	3791071.981222	172.17.0.2	172.17.0.3	HTTP	995	HTTP/1.1 200 OK (text/html)
10180	3791071.981517	172.17.0.3	172.17.0.2	TLSv1.2	97	Alert (Level: Warning, Description: Close Notifv)

Frame 10175: 117 bytes on wire (936 bits), 117 bytes captured (936 bits)

，将私钥识别补齐格式进行ssl解密得到

### SSL Decrypt

IP address	Port	Protocol	Key File	Password
172.2.0.2	443	http	/1234/private.pem	

Protocol	Length	Info
TLSv1.2	313	Client Hello
TLSv1.2	680	Server Hello, Certificate, Server Hello Done
TLSv1.2	205	Client Key Exchange
TLSv1.2	117	Change Cipher Spec, Finished
TLSv1.2	308	New Session Ticket, Change Cipher Spec, Finished
HTTP	169	GET / HTTP/1.1
HTTP	995	HTTP/1.1 200 OK (text/html)
TLSv1.2	97	Alert (Level: Warning, Description: Close Notify)



```
Etag: "5a912a2b-28d"
Accept-Ranges: bytes

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx. DDCTF{efd2a79ae9ae5008694a3561fb55470e}</em></p>
</body>
</html>
```

## 0x03 安全通信

题目：请通过nc XXXX.XXXX.XXXX.XXXX XXXX答题，mission key是b9ba15b341c847c8beba85273f9b7f90，agent id随意填就可以

```
1 #!/usr/bin/env python
2 import sys
3 import json
4 from Crypto.Cipher import AES
5 from Crypto import Random
6
7
8 def get_padding(rawstr):
9     remainder = len(rawstr) % 16
10    if remainder != 0:
11        return '\x00' * (16 - remainder)
12    return ''
13
14
15 def aes_encrypt(key, plaintext):
16    plaintext += get_padding(plaintext)
17    aes = AES.new(key, AES.MODE_ECB)
18    cipher_text = aes.encrypt(plaintext).encode('hex')
19    return cipher_text
20
21
22 def generate_hello(key, name, flag):
23    message = "Connection for mission: {}, your mission's flag is: {}".format(name, flag)
24    return aes_encrypt(key, message)
25
26
27 def get_input():
28    return raw_input()
29
30
31 def print_output(message):
32    print(message)
33    sys.stdout.flush()
34
35
36 def handle():
37    print_output("Please enter mission key:")
38    mission_key = get_input().rstrip()
39
40    print_output("Please enter your Agent ID to secure communications:")
41    agentid = get_input().rstrip()
42    rnd = Random.new()
43    session_key = rnd.read(16)
44
45    flag = '<secret>'
46    print_output(generate_hello(session_key, agentid, flag))
47    while True:
48        print_output("Please send some messages to be encrypted, 'quit' to exit:")
49        msg = get_input().rstrip()
50        if msg == 'quit':
51            print_output("Bye!")
52            break
53        enc = aes_encrypt(session_key, msg)
54        print_output(enc)
55
56
57 if __name__ == "__main__":
58    handle()
```

分析代码得知，该题为aes ecb加密，这几天密码学课刚好讲到这，ecb是一种非常不安全的加密，wiki上有很好的举例。脚本放后面了。

1.本题通过输入agent\_id,构造形如

Connection for mission: {agent\_id}, your mission's flag is: DDCTF{32位}的字符串，

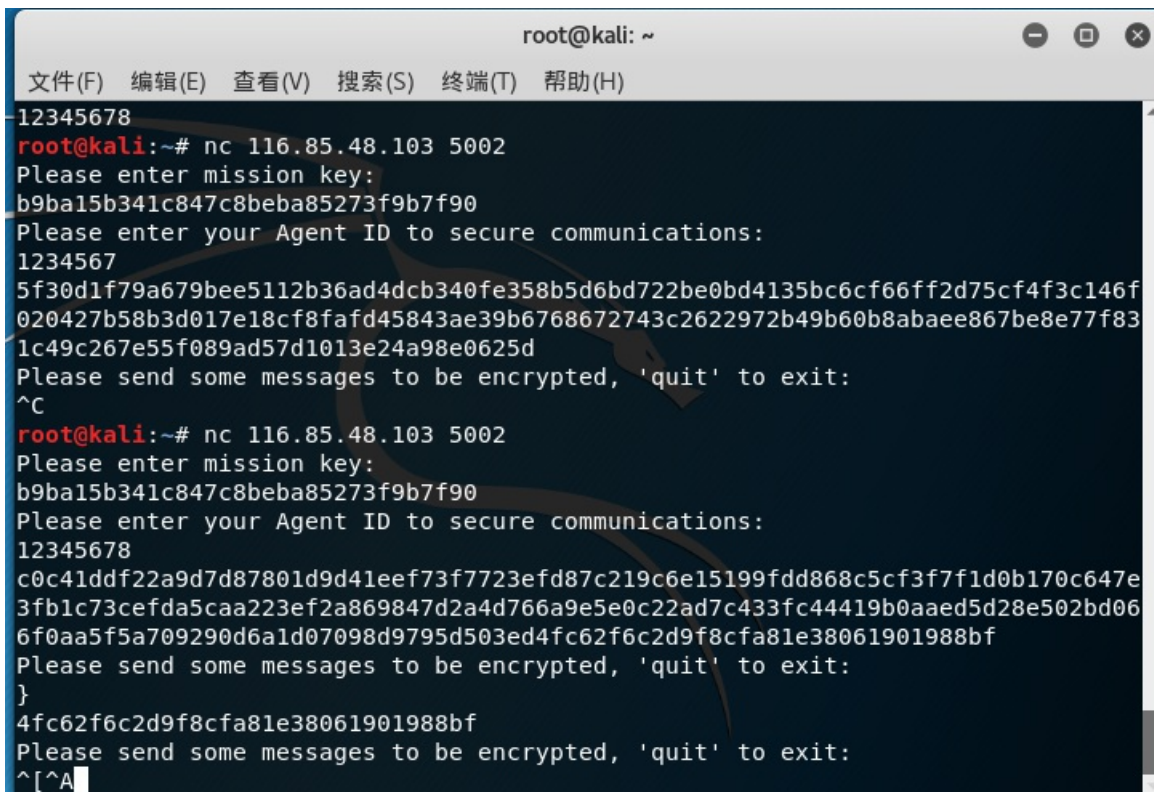
用随机生成的16位密钥进行加密，在一次连接中密钥不会变换

2.因为aes的特性通过改变输入agent\_id长度，可以确定flag长度为32位，如下图

3.因为aes特性将字符串16位一组分组，当输入12345678时，字符串多出一位}，分析脚本得知会用0补齐。

```
Connection for mission: 1234567, your mission's flag is: Edctf{89012345678901234561234567890123}
Connection for mission: 12345678, your mission's flag is: Edctf{89012345678901234561234567890123}
```

4.aes\_ecb的特性，最后一组的}和单独输入}加密后一样，得知可以进行爆破，通过改变agent\_id长度进行爆破。得到flag

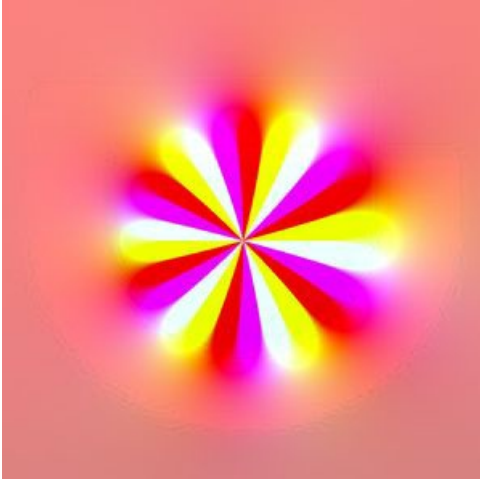


```
root@kali: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
12345678
root@kali:~# nc 116.85.48.103 5002
Please enter mission key:
b9ba15b341c847c8beba85273f9b7f90
Please enter your Agent ID to secure communications:
1234567
5f30d1f79a679bee5112b36ad4dcb340fe358b5d6bd722be0bd4135bc6cf66ff2d75cf4f3c146f
020427b58b3d017e18cf8fafd45843ae39b6768672743c2622972b49b60b8abaee867be8e77f83
1c49c267e55f089ad57d1013e24a98e0625d
Please send some messages to be encrypted, 'quit' to exit:
^C
root@kali:~# nc 116.85.48.103 5002
Please enter mission key:
b9ba15b341c847c8beba85273f9b7f90
Please enter your Agent ID to secure communications:
12345678
c0c41ddf22a9d7d87801d9d41eef73f7723efd87c219c6e15199fdd868c5cf3f7f1d0b170c647e
3fb1c73cefd5caa223ef2a869847d2a4d766a9e5e0c22ad7c433fc44419b0aaed5d28e502bd06
6f0aa5f5a709290d6a1d07098d9795d503ed4fc62f6c2d9f8cfa81e38061901988bf
Please send some messages to be encrypted, 'quit' to exit:
}
4fc62f6c2d9f8cfa81e38061901988bf
Please send some messages to be encrypted, 'quit' to exit:
^[^A
```



警方正在利用某黑客使用过的设备，来追查其行踪，并发现了下面附件中的文件。

该黑客利用附件中的encrypt.py，将其密钥转换成encrypted.bmp保存。警方在测试时，发现利用encrypt.py字符串“DDCTF{”能被转换成下图的美丽图样。



你能帮助警方从encrypted.bmp恢复出该黑客的密钥么？（为便于修正误差，本题flag后面重复出现了一次flag本体）

题解:根据两张图片和一些我自定义的字符串进行加密，得到字符串长7位时每种颜色出现6次，8位时出现7次，数出加密图片颜色次数位37次可以确定加密的字符串长38位。根据题意flag本体重复了一次所以加密字符串格式为DDCTF{15位} 15位，还有一位一直不知道是啥，现在知道是空格了

直接引用出题人的解答吧，毕竟我连傅里叶变换都没学过。只知道解题脚本是用那个微分方程写出来的。

[https://github.com/garzon/DDCTF\\_2018/tree/master/complex\\_stego](https://github.com/garzon/DDCTF_2018/tree/master/complex_stego)

根据函数名z\_encrypt、题目名complex、提示（傅里叶变换视频）、搜索引擎或者数学功底，了解到这是一个z变换，encrypted.png为其函数图象，参照[https://en.wikipedia.org/wiki/Z-transform#Inverse\\_Z-transform](https://en.wikipedia.org/wiki/Z-transform#Inverse_Z-transform) 中的路径积分公式

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{+\pi} X(e^{j\omega}) e^{j\omega n} d\omega.$$

计算即可。在单位圆上面选取1000个点计算数值积分即可。

贴一下出题人的解题脚本

```

1 from PIL import Image
2 import cmath, math, random, string
3
4 maxLen = 45
5 img_size = 800
6 abs_cor_size = 1.05
7 maxValue = 2200.0
8
9 def linear_map(v, old_dbound, old_ubound, new_dbound, new_ubound):
10     return (v-old_dbound)*1.0/(old_ubound-old_dbound)*(new_ubound-new_dbound) + new_dbound
11
12 def read_from_img(img):
13     def mapping(r, i):
14         x = int(round(linear_map(r, -abs_cor_size, abs_cor_size, 0, img_size)))
15         y = int(round(linear_map(i, -abs_cor_size, abs_cor_size, img_size, 0)))
16         return (x, y)
17     def X(z):
18         pix_pos = mapping(z.real, z.imag)
19         t = img.getpixel(pix_pos)
20         return complex(linear_map(t[1], 0, 255, -maxValue, maxValue), linear_map(t[2], 0, 255, -maxValue,
maxValue))
21     return X
22
23 def z_decrypt(X):
24     real_max_val = 0.0
25     decrypted = []
26     for n in xrange(maxLen):
27         res = 0.0
28         dw = 0.001
29         w = -math.pi
30         while w < math.pi:
31             v = X(pow(math.e, 1j * w))
32             if v.real > real_max_val: real_max_val = v.real
33             if v.imag > real_max_val: real_max_val = v.imag
34             res += v * pow(math.e, 1j * w * n) * dw
35             w += dw
36         decrypted.append(res/math.pi/2)
37     print 'realmax', real_max_val
38     err = [math.fabs(j.imag) for j in decrypted]
39     res = ''.join([string.printable[j-1] if j <= len(string.printable) and j >= 1 else '?' for i in
decrypted for j in [int(round(i.real))]]).strip('?')
40     return res
41
42 img = Image.open('encrypted.bmp')
43 X = read_from_img(img)
44 decrypted = z_decrypt(X)
45 print len(decrypted), decrypted
46 print '----- decrypted! -----'

```

```

C:\WINDOWS\system32\cmd.exe
realmax 1475.29411765
38 DDCTF {ZtR@n5F0rm_be7a} ZtR@n5F0rm_be7a
----- decrypted! -----
请按任意键继续. . .

```