DDCTF 2018线上赛writeup

转载

weixin_34077371 于 2018-04-24 14:35:00 发布 99 收藏
文章标签： 人工智能 python json
原文链接：http://www.cnblogs.com/semishigure/p/8930257.html

第一题：

d4e8e1f4a0f7e1f3a0e6e1f3f4a1a0d4e8e5a0e6ece1e7a0e9f3baa0c4c4c3d4c6fbb9b2b2e1e2b9b9b7b4e1b4b7

解题思路：

首先尝试各种解密，无果。

开始研究，分析字符串，首先看看有哪些字符，然后准备分析频率

```
import string

cipertext =
"d4e8e1f4a0f7e1f3a0e6e1f3f4a1a0d4e8e5a0e6ece1e7a0e9f3baa0c4c4c3d4c6fbb9b2b2e1e2b9b9b7b4e1b4b7e3e4b3b2b2e3e6b
4b3e2b5b0b6b1b0e6e1e5e1b5fd"
for i in string.lowercase:
    if i in cipertext:
        print i
for i in string.digits:
    if i in cipertext: print i
```

打印出来发现字符串由a-f 0-9组成，瞬间想到16进制格式，开始向16进制进攻

len()查看到字符长度为134，16进制一般是两个字符组成一个字节，所以两个两个拆解分开试试

```
cipertext =
"d4e8e1f4a0f7e1f3a0e6e1f3f4a1a0d4e8e5a0e6ece1e7a0e9f3baa0c4c4c3d4c6fbb9b2b2e1e2b9b9b7b4e1b4b7e3e4b3b2b2e3e6b
4b3e2b5b0b6b1b0e6e1e5e1b5fd"
i = 0
plaintext = ""
while i < 133:
    plaintext += str(int(cipertext[i:i + 2], 16)) + " "
    i += 2
print plaintext
```

得到212 232 225 244 160 247 225 243 160 230 225 243 244 161 160 212 232 229 160 230 236 225 231 160 233 243 186 160 196 196 195 212 198 251 185 178 178 225 226 185 185 183 180 225 180 183 227 228 179 178 178 227 230 180 179 226 181 176 182 177 176 230 225 229 225 181 253

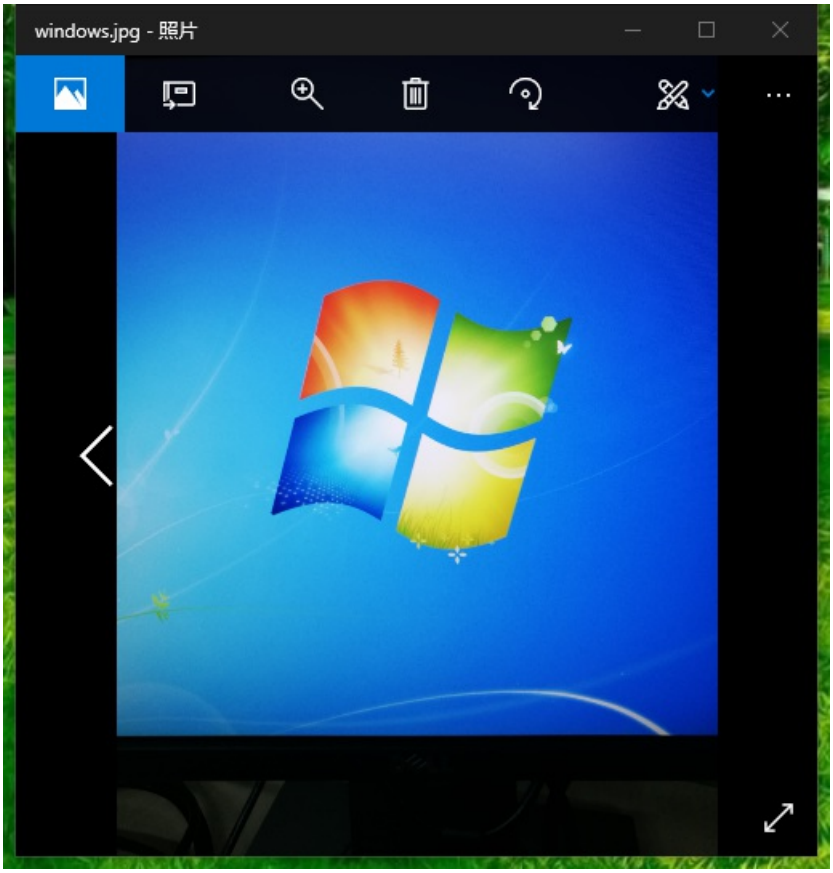
明显看到超出常规ascii范围，考虑尝试凯撒解密，由于ascii有128字符，尝试先减去128

```
cipertext =
"d4e8e1f4a0f7e1f3a0e6e1f3f4a1a0d4e8e5a0e6ece1e7a0e9f3baa0c4c4c3d4c6fbb9b2b2e1e2b9b9b7b4e1b4b7e3e4b3b2b2e3e6b
4b3e2b5b0b6b1b0e6e1e5e1b5fd"
i = 0
plaintext = ""
while i < 133:
    asciinum = int(cipertext[i:i + 2], 16)-128
    plaintext += chr(asciinum)
    i += 2
print plaintext
```

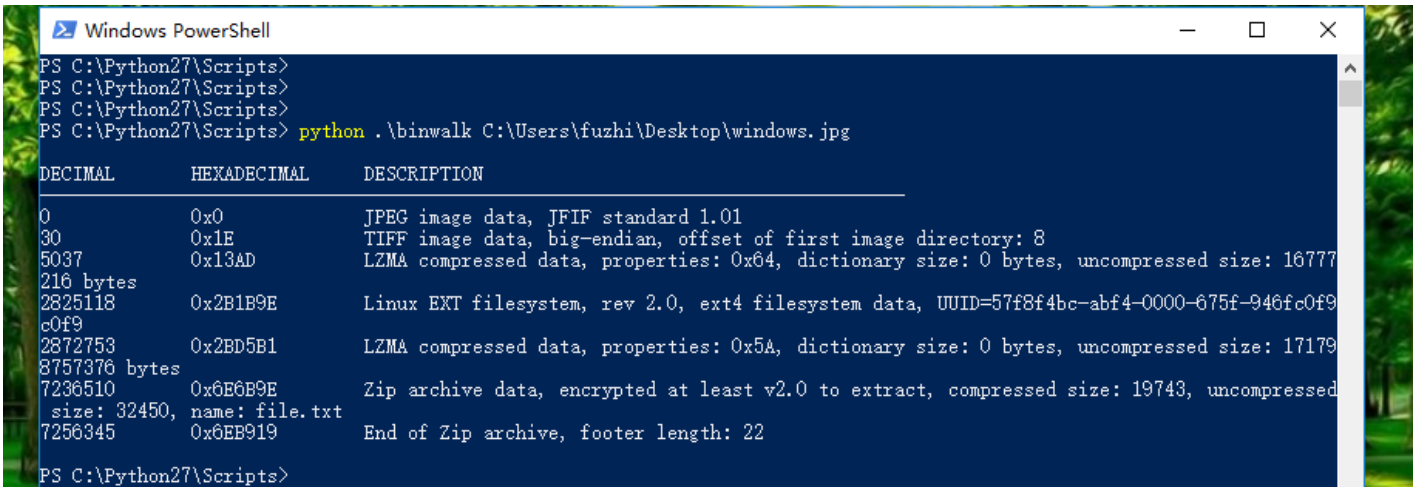
得到flag

第二题：

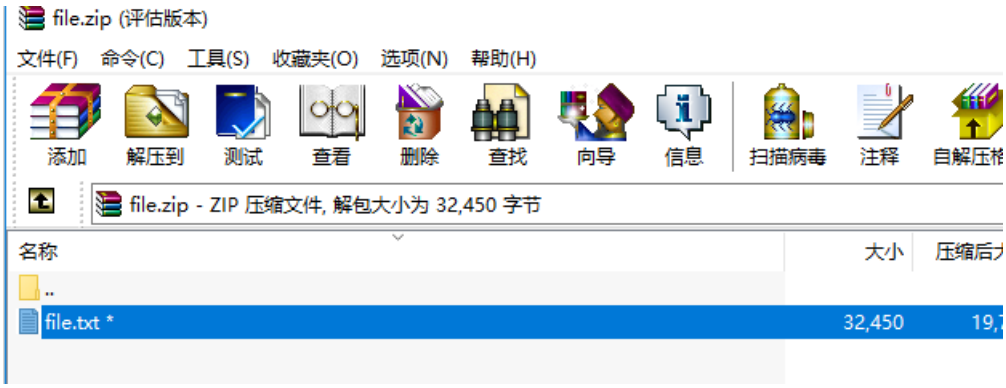
打开下载的windows.jpg



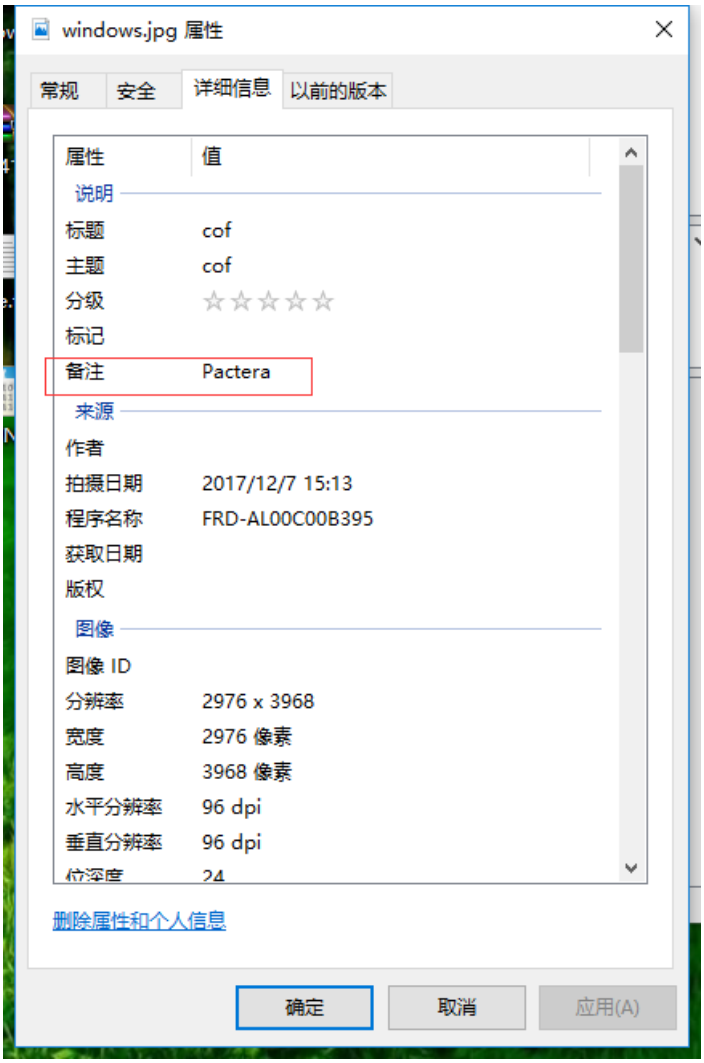
非常普通的一张图，看到图片，第一联想到隐写术，自行用binwalk看看图片有没有藏东西



一大堆东西，尝试提取分离，得到一个zip，打开一看需要密码

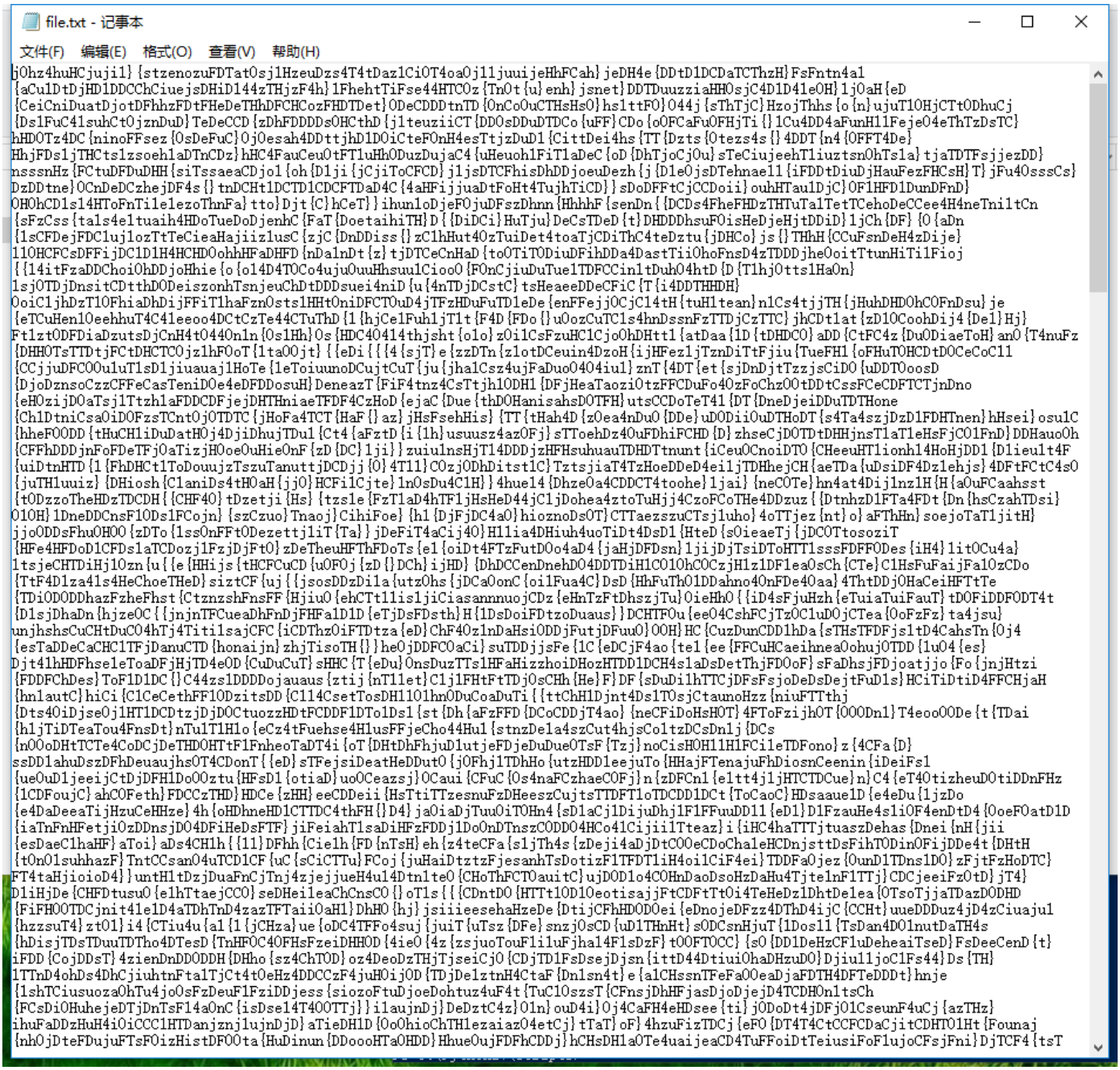


这里可以用暴力破解，不过花费时间较长，期间可以做一些其他事情，比如去看看图片的详细信息(可能藏东西)



在备注找到Pactera，尝试作为密码输入，解压成功Orz.....

打开file.txt



进行一顿栅栏，编码等解密，无果。

破了很久，最后回头看题目看了的提示

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。

尝试分析字符频率,然后进行排序

```
file1 = open("file.txt", "r")
cipertext = file1.read()
dic1 = {}
for i in cipertext:
    if i not in dic1:
        dic1[i] = 1
    else: dic1[i] += 1 z = zip(dic1.values(), dic1.keys()) plaintext = "" for i in sorted(z): plaintext =
i[1] + plaintext print plaintext
```

直接得到flag

第三题：

本题差评！！！干扰项太多，不亲切，这里不再赘述踩的坑

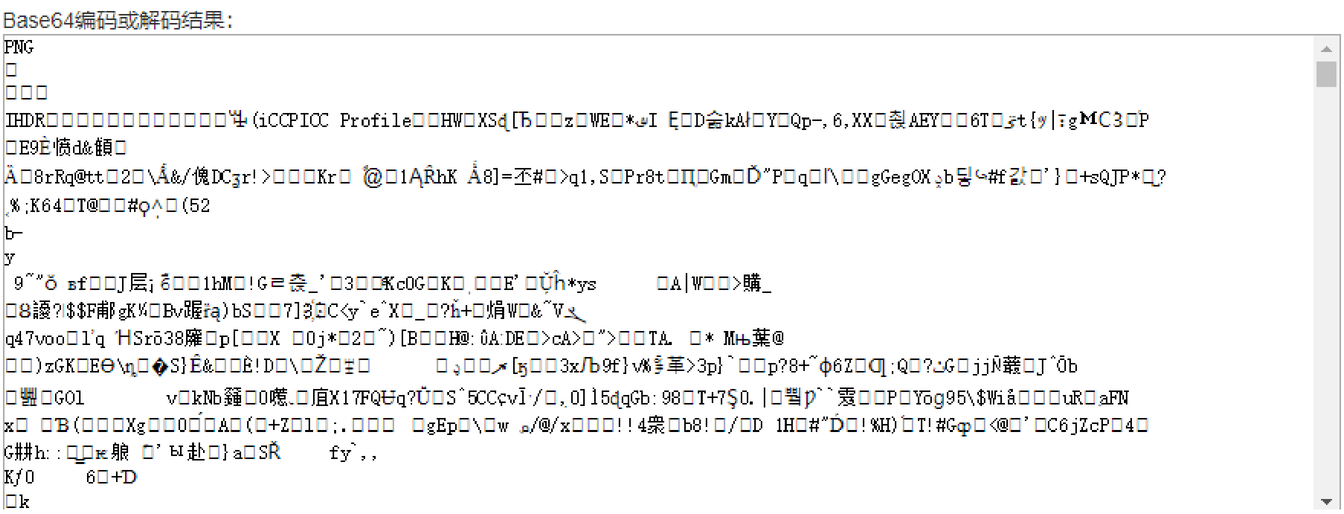
下载得到附件，是一个pcap文件，果断丢入wireshark，把解析的到的文件一次导出

只有IMF的能导出，分析查看，大概在一个文件比较大的地方，看到

--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

iVBORw0KGgoAAAANSUhEUgAABIwAAAISCAYAAACu61SmAAAMKG1DQ1BJQ0MgUHJvZmlsZQAA
SImVVwdYU8kWnluSkJDQAhGQEnoTpVepoUUQkCrYCEkgocSYEETsyKKCa0HFghVZFVFwLYAs
NixYWBTs9WFBRVkXCzZU3iQBdPV7733vfN+5979nzpz5z7kz880AoB7NEYuzUA0AskU5kpjQ
QObEpGQm6SFAAArIwBM4crhScUB0dASAMvT+p7y7Dr2hXLGXx/q5/b+KJo8v5QKAREOcypNy
syE+BADuxhVLcgAg9EC72cwcMcREyBJoSyBBiM31OF2JPeQ4VYkjFD5xMSyIUwBQoXI4knQA
lOS8mLncdBhHbRnEDiKeUARxE8S+XAGHB/FniEd1Z0+HWN0aYuvU7+Kk/yNm6nBMDid9GCtz
UYhKkFAqzuLM+j/L8b81O0s2NIYZVKpAEhYjzllet8zp4XJMhficKDUyCmItiK8KeQp/OX4i
kIXFD/p/4EpZsGaAAQBK5XGCwiE2gNhU1BUZMWj3TROGsCGGtUfjhDnsOGVflCeZHjMYH83j
S4NjhzBHohhL71Msy4wPGIy5RcBnD8VszBfEJSp5opdzhQmREKtBfFeaGRs+6PM8X8CKHPKR
yGLknOE/x0CaJCRG6YOZZ0uH8sK8BEJ25CCOyBHEhSn7Y1O5HAU3XYgz+NKJEUM8efygYGVe
WAFfFD/IHysV5wTGDPpXirOiB/2xJn5WqNxuCnGbNDd2qG9vDpxsynxxIM6Jj1Nyw7UzOOOi
lRxwWxABWCAIMIEMaiqYDjKAsK2nvgd+KVtCAAdIQDrgA/tBy1CPREWLCD5jQT74CyI+kA73
C1S08kEutH8Ztiqf9iBN0Zqr6JEJnkCcDcJBFvyWKXqJhkdLAI+hRfjT6FzINQuqvO0nG1N9
yEYMJgYRw4ghRBtcH/fFvfEI+PSH6oR74J5DvL75E54Q2gkPCdcInYRb04QFkh+YM8F40Ak5
hgxml/p9drgljOqKB+I+MD6MjTNwfWCPu8CRAnA/OLYrtH7PVTac8bdaDsYiO5BR8giyP9n6
RwZqtmquwlHklfq+FkpeqcPVYg23/JgH67v68eA7/EdPbAl2EGvBTmLnsSasHjCx4lgDlood
lePhufFYMTeGRotR8MmEcYQ/jccZHFNeNalDtUO3w+fBNpDDz8uRLxbWdPEsiTBdkMMMgLsl
n8kWcUePYjo5OMJdVL73K7eWNwzFno4wLnyzFewAwMd3YGCg6ZstvAuAg70AUO59slnDdajW
AcC5NVyZJFdpw+UPAqAAdbhS9IAR3LusYUZOwAl4A38QDMaBKBAHksBUWGcBnKcSMBPMAQtB

复制出来，看到是base64，解码，得到一串字符



这个看到应该是一张图片，所以这里保存成图片看一看。。

```
#!/usr/bin/env python
# coding=utf-8
import base64

file1 = open("test", "r")
cipertext = file1.read()
file1.close()

plaintext = base64.b64decode(cipertext)  # 读取file1进行base64解码
file2 = open("testpng.png", "wb")  # 以.png写入保存
file2.write(plaintext)
file2.close()
```

打开我们刚才保存的图片

```
MIICXAIBAAKBgQDCm6vZmclJrVH1AAyGuCuSSZ8O+mIQiOUQCvN0HYbj8153JfSQ
LsJIhbRYS7+zZ1oXvPemWQDv/u/tzegt58q4ciNmcVnq1uKiygc6QOtvT7oiSTyO
vMX/q5iE2iClYUIHZEKX3BjjNDxrYvLQzPyGD1EY2DZIO6T45FNKYC2VDwIDAQAB
AoGAbtWUKUkx37lLfRq7B5sqjZVKdpBZe4tL0jg6cX5Djd3Uhk1inR9UXVNw4/y4
QGfzYqOn8+Cq7QSoBysHOeXSiPztW2cL09ktPgSlfTQyN6ELNGuiUOYnaTWYZpp/
QbRcZ/eHBulVQLlk5M6RVs9BLI9X08RAl7EcwumiRfWas6kCQQDvqC0dxl2wIjwN
czILcoWLig2c2u71Nev9DrWjWHU8eHDuzCJWvOUAHIrkexddWEK2VHd+F13GBCOQ
ZCM4prBjAkEAz+ENahsEjBE4+7H1HdIaw0+goe/45d6A2ewO/lYH6dDZTAzTW9z9
kzV8uz+Mmo5163/JtvwYQcKF39DJGGtqZQJBAKa18XR16fQ9TFL64EQwTQ+tYBzN
+04eTWQCmH3haeQ/0Cd9XyHBUveJ42Be8/jeDcIx7dGLxZKajHbEAfBFnAsCQGq1
AnbJ4Z6opJCGu+UP2c8SC8m0bhZJDelPRC8IKE28eB6SotgP61ZqaVmQ+HLJ1/wH
/5pfc3AmEyRdfyx6zwUCQCAH4SLJv/kprRz1a1gx8FR5tj4NeHEFFNEgq1gmiwmH
2STT5qZWzQFz8NRe+/otNOHBR2Xk4e8IS+ehIJ3TvyE=
```

这里就要使用图像识别了，ocr了解一下

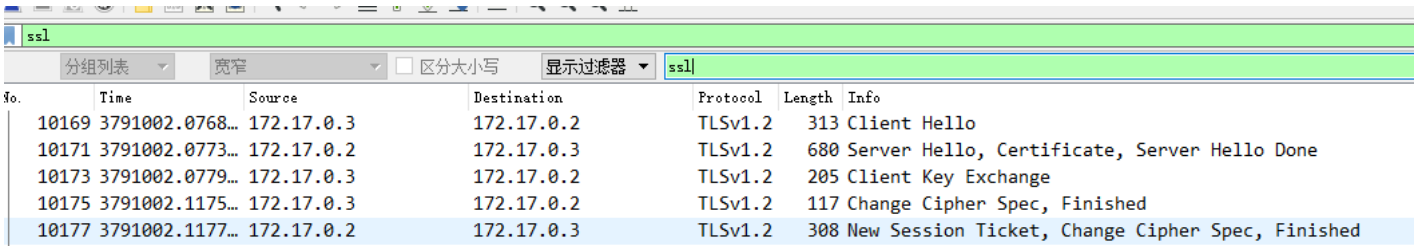
```
#!/usr/bin/env python
# coding=utf-8

import base64
import pytesseract
from PIL import Image


def getcode(imgurl):
    """识别图片"""
    image = Image.open(imgurl) vcode = pytesseract.image_to_string(image) code =
base64.b64decode(vcode.encode("utf-8")) return code temp_imgurl = 'testpng.png' code = getcode(temp_imgurl)
print code
```

这里无敌坑，ocr识别准确率不是很高，code得到后还要和图片进行比对检查。

根据提示，补全RSA

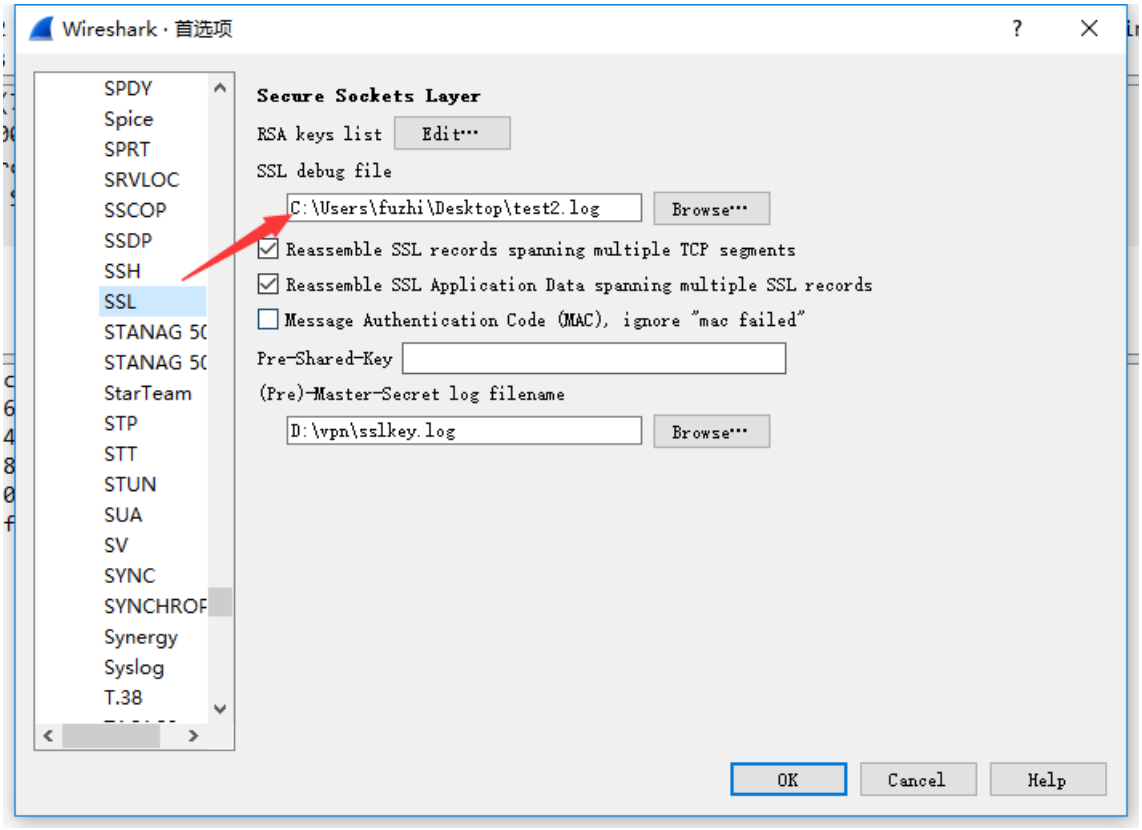
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQDCm6vZmclJrVH1AAyGuCuSSZ8O+mIQiOUQCvN0HYbj8153JfSQ
LsJIhbRYS7+zZ1oXvPemWQDv/u/tzegt58q4ciNmcVnq1uKiygc6QOtvT7oiSTyO
vMX/q5iE2iClYUIHZEKX3BjjNDxrYvLQzPyGD1EY2DZIO6T45FNKYC2VDwIDAQAB
AoGAbtWUKUkx37lLfRq7B5sqjZVKdpBZe4tL0jg6cX5Djd3Uhk1inR9UXVNw4/y4
QGfzYqOn8+Cq7QSoBysHOeXSiPztW2cL09ktPgSlfTQyN6ELNGuiUOYnaTWYZpp/
QbRcZ/eHBulVQLlk5M6RVs9BLI9X08RAl7EcwumiRfWas6kCQQDvqC0dxl2wIjwN
czILcoWLig2c2u71Nev9DrWjWHU8eHDuzCJWvOUAHIrkexddWEK2VHd+F13GBCOQ
ZCM4prBjAkEAz+ENahsEjBE4+7H1HdIaw0+goe/45d6A2ewO/lYH6dDZTAzTW9z9
kzV8uz+Mmo5163/JtvwYQcKF39DJGGtqZQJBAKa18XR16fQ9TFL64EQwTQ+tYBzN
+04eTWQCmH3haeQ/0Cd9XyHBUveJ42Be8/jeDcIx7dGLxZKajHbEAfBFnAsCQGq1
AnbJ4Z6opJCGu+UP2c8SC8m0bhZJDelPRC8IKE28eB6SotgP61ZqaVmQ+HLJ1/wH
/5pfc3AmEyRdfyx6zwUCQCAH4SLJv/kprRz1a1gx8FR5tj4NeHEFFNEgq1gmiwmH
2STT5qZWzQFz8NRe+/otNOHBR2Xk4e8IS+ehIJ3TvyE=
-----END RSA PRIVATE KEY-----

看到这个rsa秘钥，自然联想到ssl，搜索一下（ssl了解一下）

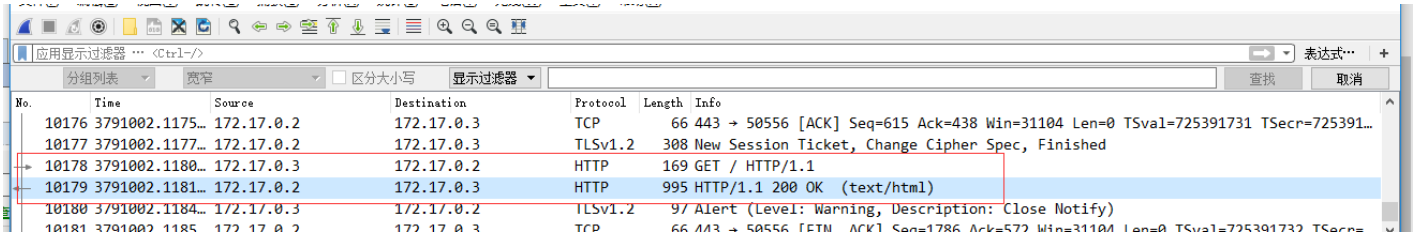


看到了ssl的通信

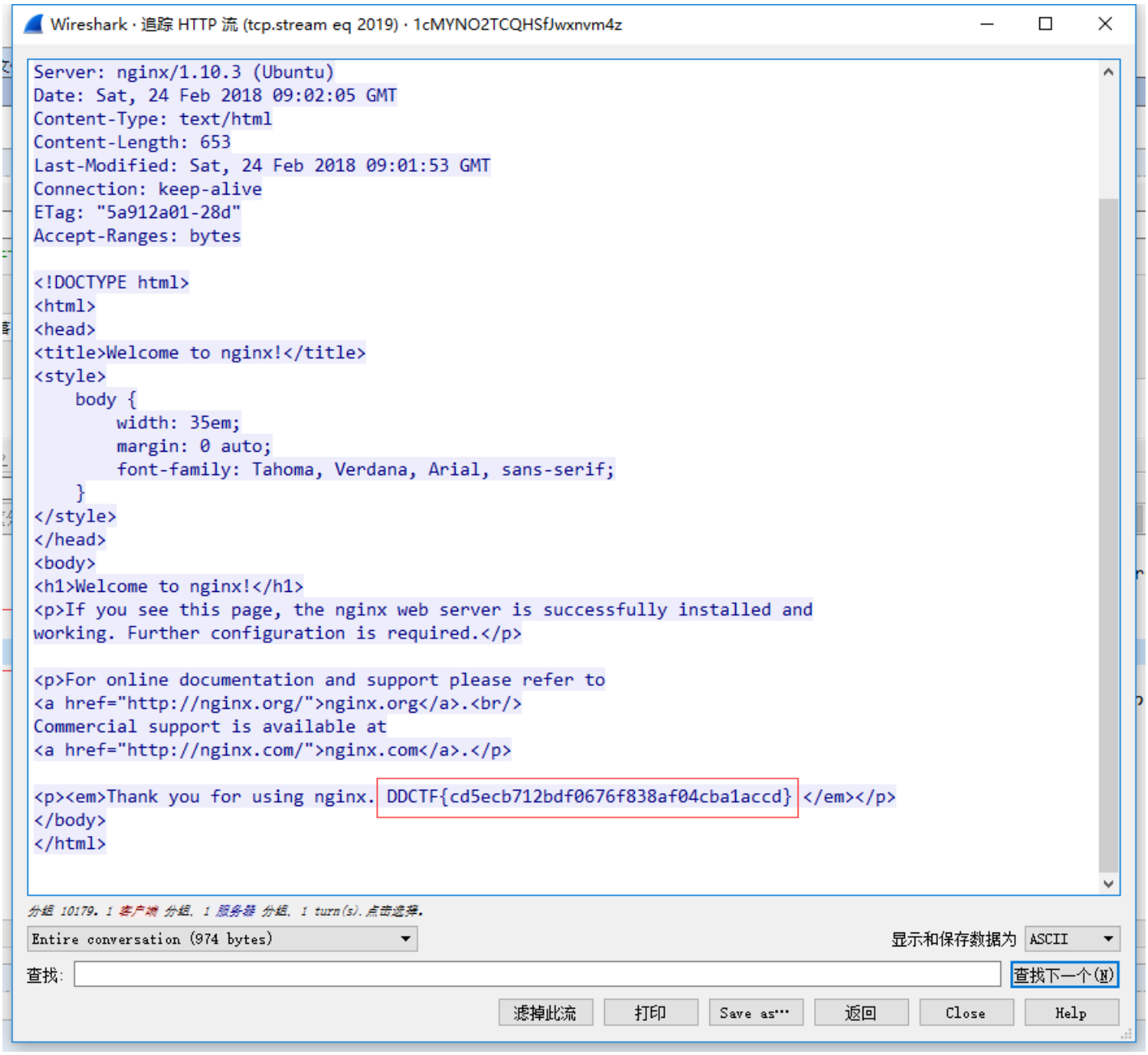
把秘钥导入



重启wireshark，惊奇地发现在最后多了http



追踪http流


Wireshark · 追踪 HTTP 流 (tcp.stream eq 2019) · 1cMYNO2TCQHSfJwxnvm4z
Server: nginx/1.10.3 (Ubuntu)
Date: Sat, 24 Feb 2018 09:02:05 GMT
Content-Type: text/html
Content-Length: 653
Last-Modified: Sat, 24 Feb 2018 09:01:53 GMT
Connection: keep-alive
ETag: "5a912a01-28d"
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>
If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.
For online documentation and support please refer to
nginx.org.
Commercial support is available at
nginx.com.
Thank you for using nginx. DDCTF{cd5ecb712bdf0676f838af04cba1accd}
</body>
</html>
分组 10179. 1 客户端 分组, 1 服务器 分组, 1 turn(s). 点击选择.
Entire conversation (974 bytes)
显示和保存数据为 ASCII
查找:
查找下一个(N)
滤掉此流
打印
Save as...
返回
Close
Help

第四题


提交答案
已解决66
安全通信
250
请通过nc 116.85.48.103 5002答题, mission key是
f49348cf84d390da52498077ae7137d5, agent id随意填就可以

```python
#!/usr/bin/env python
import sys
import json
from Crypto.Cipher import AES
from Crypto import Random


def get_padding(rawstr):
    remainder = len(rawstr) % 16
    if remainder != 0:
        return '\x00' * (16 - remainder)
    return ''


def aes_encrypt(key, plaintext):
    plaintext += get_padding(plaintext)
    aes = AES.new(key, AES.MODE_ECB)
    cipher_text = aes.encrypt(plaintext).encode('hex')
    return cipher_text


def generate_hello(key, name, flag):
    message = "Connection for mission: {}, your mission's flag is: {}".format(name, flag)
    return aes_encrypt(key, message)


def get_input():
    return raw_input()


def print_output(message):
    print(message)
    sys.stdout.flush()


def handle():
    print_output("Please enter mission key:")
    mission_key = get_input().rstrip()

    print_output("Please enter your Agent ID to secure communications:")
    agentid = get_input().rstrip()
    rnd = Random.new()
    session_key = rnd.read(16)

    flag = '<secret>'
    print_output(generate_hello(session_key, agentid, flag))
    while True:
        print_output("Please send some messages to be encrypted, 'quit' to exit:")
        msg = get_input().rstrip()
        if msg == 'quit':
            print_output("Bye!")
            break
        enc = aes_encrypt(session_key, msg)
        print_output(enc)


if __name__ == "__main__":
    handle()
```

这题算是纯crypto，采用的是MODE_ECB的AES加密。

题目大概的意思就是明文由agentid和flag组成，随意输入 agentid后明文会被随机产生的key进行AES加密，由于每一次远程到116.85.48.103时key都是随机产生的，所以我们只能通过后面的加密尝试来猜测出明文

首先AES了解一下

ECB加密是分组进行加密的，解密也是分组解密。分组与分组之间的明文产生的密文互相独立，且由于算法的缘故，相同的明文分组在相同的密钥加密下会产生相同的密文
加解密流程如下图所示

而我们要做的事是通过这些个分组且明文加密固定密文的特性猜出flag的每一位来
题中以16字节为一组，我们举例也拿16字节为一组举例
首先我们假设xxxx是我们可控的输入，一般情况下的加密会是这样的

但是如果我们控制xxx为十五个固定的字符如十五个A
则加密过程会变成这样：

现在我们记录下此时的HEX_1，在与密文进行比较，如果一样就记下此时F的值，如此一个一个循环推测

以下是解题代码

```python
#!/usr/bin/env python
# coding=utf-8
import time
import socket
import string

agentid = ""
message = ""
flag = ""

for i in range(45):
    agentid += "1"


def returnmsg(data):
    """发送与接收"""
    s.send(data)
    time.sleep(0.3)
    msg = s.recv(1024)
    return msg


while True:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(("116.85.48.103", 5002))

    print returnmsg("2acba569d223cf7d6e48dee88378288a\n")  # 发送题目的mission key
    cipertext = returnmsg(agentid + "\n")  # 得到密文
    print cipertext

    for i in string.lowercase + string.digits + string.uppercase + "{}":
        message = "Connection for mission: {}, your mission's flag is: {}".format(agentid, flag + i)
        info = returnmsg(message + "\n")
        if info.split("\n")[0] in cipertext: # 如果尝试加密的内容和第一次密文相同,则记下i
            print info
            flag += i
            print "find key code: " + i
            break

    print message
    s.shutdown(2)
    s.close()
    if agentid == "": # 每一次猜到keycode后,agentid都要减一位
        break
    else:
        agentid = agentid[:-1]
    if i == "}":
        break
```

转载于:https://www.cnblogs.com/semishigure/p/8930257.html