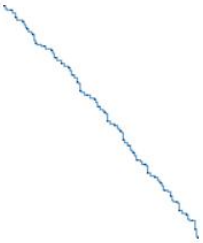
小塔WEEK1的WP

crypto

Dancing Line

题目给出了一张图片



猜测黑色点位置为分隔符，蓝色的横竖线以及长度代表信息。

进一步推测横竖代表1和0 的二进制 所含信息是以二进制表示的ASCII码

将‘ h ’的二进制表示代入第一段发现符合 假设成立

于是对照表格手动翻译出flag【擦汗】

Easy RSA

好耶！是p q c e 都给出的RSA
将原数据改为嵌套列表然后写一个弱智代码（别骂了呜呜呜）

```
import libnum
from Crypto.Util.number import long_to_bytes
list=[[12433, 149, 197, 104], [8147, 131, 167, 6633].....]#省略了数据
for i in range(0,38):

    c = list[i][3]
    e = list[i][0]
    q = list[i][1]
    p = list[i][2]
    n = p*q
    d = libnum.invmod(e, (p - 1) * (q - 1))
    m = pow(c, d, n)
    string = m
    print(string)
```

解出来得到flag

Matryoshka

俄罗斯套娃诚不欺我（hint真的帮大忙了）

除了第二步比较坑 其他步骤的标志都比较明确

不过第二步也真的是学到了——

就简单列一下解码顺序吧：

Braille -> 整个倒置（？） -> Morse -> ASCII -> Vigenère -> Base64 -> RailFence -> Caeser -> flag

Braille -> 整个倒置（？） -> Morse -> ASCII -> Vigenère -> Base64 -> RailFence -> Caeser -> flag

English Novel

附件给了一个enc文件

扔进010 Editor里面得到被加密的flag

根据题目给的加密算法

可以得到信息：

1.key是根据位置一一对应的（非字母的那一位会被跳过）

2.和偏移量有关

也就是说思路是利用被加密的小说和原小说进行配对，找出每一位的偏移量或者key，再解出flag

但是被加密后的txt文档顺序被打乱

写一个简单代码来寻找配对的txt

```
import os
path = '' #省略文件夹目录
files= os.listdir(path)

for file in files:
    f = open('path'+'/'+file ,'r')
    a = f.read()
    if      :  #省略根据原文得到的查找条件
        print(file)
```

接下来的做法有两种：

1.小改一下上面这个代码 从加密txt里直接找到**对应位置的值**和enc文件中**相同**的文档，再次用上述代码找到匹配的原txt**对应位置的值**（事实证明这个是可行的 就是速度有点慢）

2.随便找几对匹配的txt文件（要求能凑出每个位置都有字母的一段明文和一段密文），计算出偏移量或者直接写代码找出key

misc

这个压缩包有点麻烦

附件是一个压缩包

打开发现有加密，有注释：

```
Pure numeric passwords within 6 digits are not safe!
```

使用ARCHRP进行暴力攻击：口令长度为6，范围所有数字

成功得到口令。

打开README得到提示：

> I don't know if it's a good idea to write down all the passwords.

再次使用ARCHRP，把附件password-note放入进行字典攻击

成功得到口令。

打开README得到提示：

> If you don't like to spend time compressing files, just stores them.

“store”这个词用的很巧妙，因为加密的压缩包里也有一个README.txt,考虑使用明文攻击

把外面这个README.txt文件用“存储”的方式压缩，发现CRC32一致，证实为明文攻击

又又又使用ARCHRP，把两个压缩包放入进行明文攻击

5分钟后停止，看到窗口显示"加密密钥已成功恢复！"

保存文件，得到一张图片：

图片上没有什么信息，于是放进010 Editor发现里面有zip文件

```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
-
0             0x0             JPEG image data, JFIF standard 1.01
20421         0x4FC5          Zip archive data, encrypted at least v2.0 to extr
act, compressed size: 12410, uncompressed size: 13251, name: flag.jpg
32959         0x80BF          End of Zip archive, footer length: 22
```

用binwalk分离出来

又是一个加密的zip文件，可是没有关于密码的提示

猜测是伪加密，用010 Editor修改对应位置09 00为00 00

成功打开里面的图片 得到flag

好康的流量

附件为一个pcapng文件’

用Wireshark打开 发现全是TCP和SMTP流

统计会话 追踪TCP流 发现有png文件

将文件解密保存，得到图片

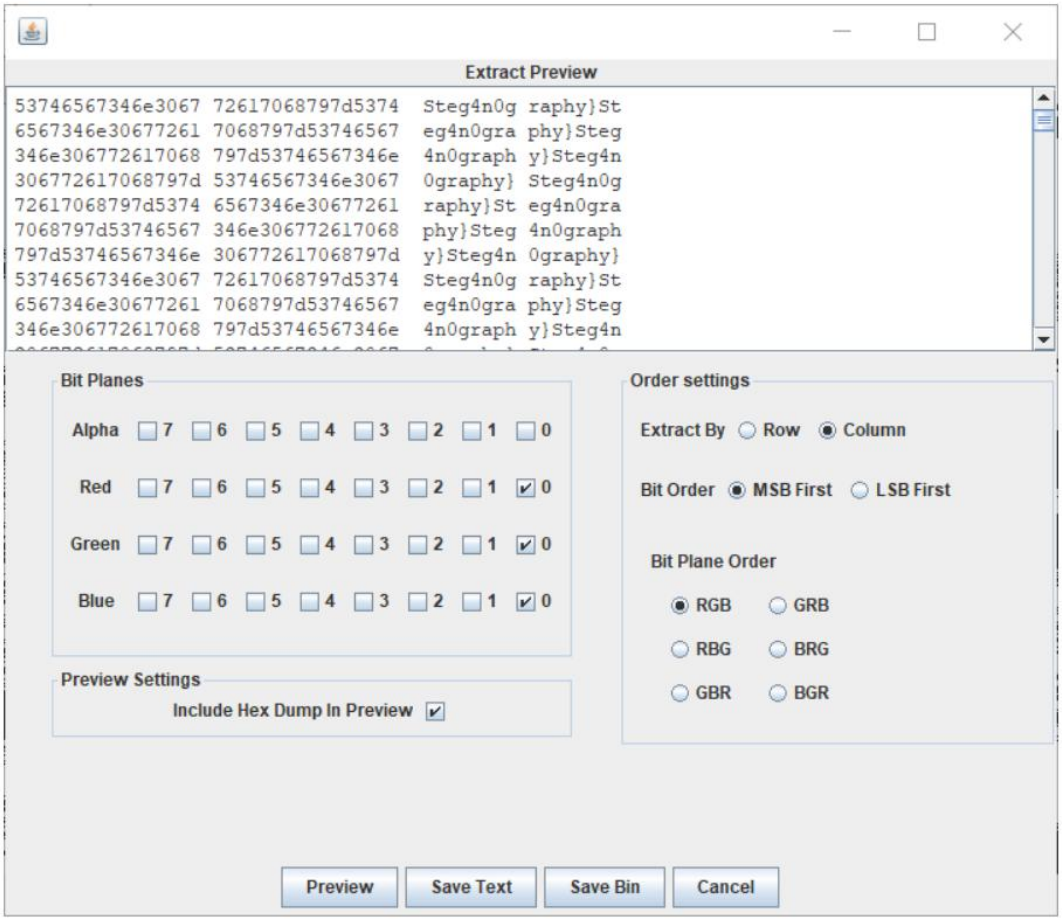
把图片放进Stegsolve，先把所有图层检查一下

在Green plane 2发现条形码：

扫码得到一半的flag

（中间省略好一大部分走了一天的弯路QAQ）

对图片进行LSB隐写分析【各种姿势】：



终于发现另一半flag

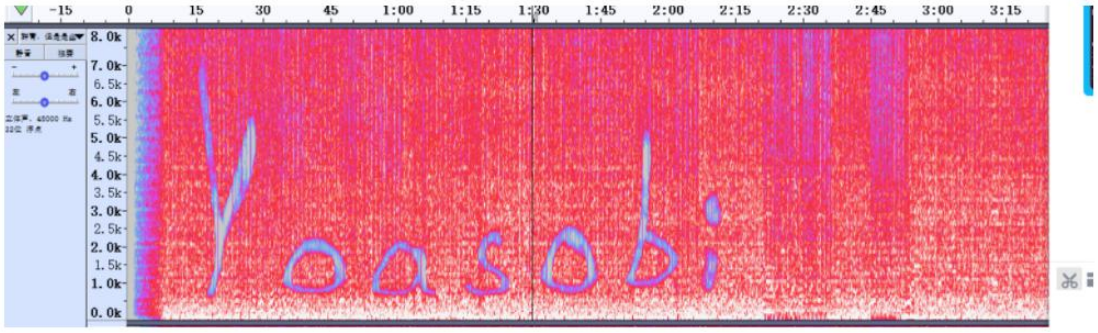
群青（其实是幽灵东京）

附件是一个音频文件

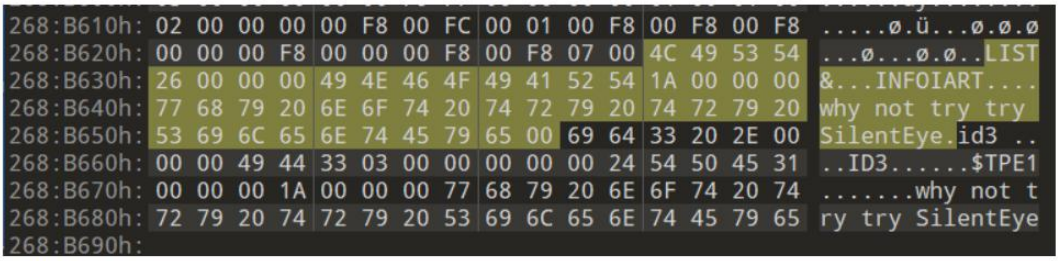
打开听听 ~~幽灵东京真好听忍不住跟着唱~~ 没有发现明显的异常

放进Audacity

在频谱图发现信息：



再放进010 Editor看一下，发现提示：



```
268:B610h: 02 00 00 00 00 F8 00 FC 00 01 00 F8 00 F8 00 F8  .....ø.ü...ø.ø.ø
268:B620h: 00 00 00 F8 00 00 00 F8 00 F8 07 00 4C 49 53 54  ...ø...ø.ø..LIST
268:B630h: 26 00 00 00 49 4E 46 4F 49 41 52 54 1A 00 00 00  &...INFOIART....
268:B640h: 77 68 79 20 6E 6F 74 20 74 72 79 20 74 72 79 20  why not try try
268:B650h: 53 69 6C 65 6E 74 45 79 65 00 69 64 33 20 2E 00  SilentEye.id3 ..
268:B660h: 00 00 49 44 33 03 00 00 00 00 00 24 54 50 45 31  ..ID3......$TPE1
268:B670h: 00 00 00 1A 00 00 00 77 68 79 20 6E 6F 74 20 74  .......why not t
268:B680h: 72 79 20 74 72 79 20 53 69 6C 65 6E 74 45 79 65  ry try SilentEye
268:B690h:
```

火速下载SilentEye 打开 猜测频谱图的信息是解密使用的密钥

解密成功再次得到一个音频文件

打开听听 ~~好怪哦什么东西再听一遍~~ 品不出摩斯密码

放进Audacity也分析不出

注意到音频文件的名字 S_S_T_V

搜搜看 发现是一种用音频传输图像的方式

使用Robot36得到图片：

扫描二维码 得到flag~