CTF 密码学入门指北

让我们先来看看什么是密码学(Cryptography):

与平时所说的“账号密码”不同，这里提到的密码学研究的是“加密”，而常说的“账号密码”对应的说法应该是“口令”（password）。一个明文经过密钥加密后，就会变为一段“让人看不懂”的密文，而经过密钥解密后，就会变回有意义的明文。然而，有些人没有密钥也想获取密文中的部分或者全部内容，甚至是直接利用密文去达到自己的要求，这就是CTFer在比赛中需要达成的目的。在比赛中，通常会把加密算法公开出来，然后再将密文等信息发送给参赛者，参赛者需要分析加密算法的缺点，以找到利用密文的方法。

在密码学中还有许多有意思的问题，例如，如何在没有事先商定好密钥的情况下，在一个公开的网络中完成信息的秘密传输？如何确认这个信息是由朋友本人传输的而不是他人伪造？是否存在理论上不可攻破的密码体系……诸多问题，将在以后的学习中得到解答。

密码学所需知识（Knowledge）：

- 基本python编程知识（只需要看懂基本的代码即可，可以边学边精进编程能力）
  https://www.runoob.com/python3/python3-tutorial.html

数学知识:

（千万不要被繁杂困难的数学知识劝退，更多的是边做题边学习，慢慢来）
入门:　初等数论
进阶: 线性代数
　　抽象代数

入门书可以看群文件

算法知识（更重要的是理解算法思想）：

- 搜索的时间复杂度，空间复杂度的估计与平衡
- 二分思想
- 二进制技巧优化 快速幂
- bsgs思想：打哈希表&分块
- etc.

相关工具（Tools）：

python

- gmpy2
- install：pip install gmpy2
  doc： https://gmpy2.readthedocs.io/en/latest/intro.html
- Crypto（pyCryptodome）
  install：pip install pycryptodome
  doc： https://www.pycryptodome.org/
- pwntools 交互题好帮手
  install：pip install pwntools

doc： https://docs.pwntools.com/en/stable/

在线网站

- CTF在线工具
- 大整数在线分解网站
- 在线sage
- etc.（剩余的就由聪明的你慢慢寻找收集啦😁

软件

- sagemath
- 强大的代数功能
- 覆盖许多数学功能的应用软件，包括代数、组合数学、图论、计算数学、数论、微积分和统计。
- 你会慢慢领悟它的强大的😆

sagemath安装：sagemath对windows支持并不好，推荐将其装在wsl或者其他虚拟机里。
https://doc.sagemath.org/html/en/installation/index.html

如果你刚入门，并且安装它有困难，可以先跳过。

学习方法（How):

- 搜索引擎google、bing
- 看书（群文件->密码相关）
- 在ctf-wiki-Crypto分区
- 在网上找大佬的博客文章进行学习
  例如： https://d33b4t0.com/

如何解决困难（Difficulty)

- 尝试利用搜索引擎解决，尝试各种问法
- 阅读《**提问的智慧**》，然后求助群里的crypto管理员~~

在Crypto的世界，你不仅可以见到各式各样的古典密码，也可以看到严密的现代密码体系，只要你能坚持下来，保持求知热情，相信你一定能收获攻破一个个有缺陷密码系统的乐趣的！

Well Done！现在开始实战吧！

阅读，**运行**并**理解**下面的例子！你可以得到第一个flag！

例题

```
from random import randint
from Crypto.Util.number import bytes_to_long,GCD
from secret import flag1#这是我们要求的
assert flag1[:7]==b'moectf{'#如何利用这个条件？

a=randint(0,256)
assert GCD(256,a)==1
b=randint(0,256)
c=[]

#enc
```

```
for i in range(len(flag1)):
    c.append((a*flag1[i]%256+b)%256)#这一步做了什么？
print(bytes_to_long(bytes(c)))

'''
23904604480218951222924468885892706253385766083586197703800132687872601727899557
8546238319758863654721 22
'''
```

参考答案1

```
# solution1 for ex1

from Crypto.Util.number import long_to_bytes,inverse

def dec(c):#a和b范围都不大，一对对(a，b)尝试或许能行
    for a in range(256):
        for b in range(256):
            if((ord('m')*a+b)%256==c[0] and (ord('o')*a+b)%256==c[1]):
                print("a=",a,"b=",b)
                #什么是乘法逆元？inverse函数具体是如何实现的？
                print(bytes(list(map(lambda x:(x-b)*inverse(a,256)%256,c))))
                return

c=long_to_bytes(23904604480218951222924468885892706253385766083586197703800132687872601727899557854623831975886365472122)

dec(c)
```

参考答案2

```
# solution1 for ex1

from Crypto.Util.number import long_to_bytes,inverse

def dec(c):#不用搜索！解一个二元方程组即可
    a=inverse((ord('c')-ord('t')),256)*(c[3]-c[4])%256
    b=(c[0]-ord('m')*a)%256
    print("a=",a,"b=",b)
    print(bytes(list(map(lambda x:(x-b)*inverse(a,256)%256,c))))
    return

c=long_to_bytes(23904604480218951222924468885892706253385766083586197703800132687872601727899557854623831975886365472122)

dec(c)
```