Crypto in CTF入门指北-MoeCTF2025

> 遇到文档内容阅读和学习上的困难点这里

什么是Cryptography?

🗣️ 引言：在“hacker”面前保护信息

"Cryptography"其实就是一种对抗监听和篡改的艺术和科学。想想看：你在给朋友发消息，但怕被别人（CTF 里的 "Eve"）看到或修改，就得用密码学保护你的信息安全。

📌 你的日常：密码学无处不在

当你通过微信聊天、用支付宝支付，甚至登录学校网站时，背后都在走“加密→传输→解密”的流程。
如果有人尝试窃听，即使他知道你用的是 AES、RSA 或其他算法，也因为没有“密钥”而读不出信息。

> 就像开锁一样，你知道面前这个锁是月牙儿形锁，但是没有钥匙，就打不开门。CTF题目就相当于给你一个简化版的锁，或者半截钥匙，或者锁的内部结构之类的，让你去开锁。

📚 密码学的三个核心目标

- 机密性(Confidentiality):确保只有合法接收者能读懂消息。
- 完整性 & 验证(Integrity & Authentication):你能确认消息没被修改，并判断是谁发的。
- 不可否认(Non-repudiation):数字签名能证明某人确实发过这条消息，事后无法否认。

🔍 在 CTF 中你会遇到这些

- 对称加密（如 AES、XOR）：解密需要知道同一把钥匙。
- 非对称加密（如 RSA、ElGamal）：用公钥加密，用私钥解密。
- 哈希函数（如 SHA256）：做信息摘要、完整性校验。
- 数字签名（如 RSA 签名、ECDSA）：验证身份，是不是你发的。

🎯 密码学试题怎么出?

在 CTF 的 Crypto 板块，常见的是“打穿错误用法”而不是推断出整个算法：

例：有人重用计算随机数、不安全地生成密钥、用错加密模式，导致逻辑被攻破。

> 正确使用算法的情况下，就连我也无法破解（

你只需掌握常见算法原理 + 编写脚本快速验证，就能在题目里“一帆风顺”。

What is Crypto in CTF?

CTF比赛中的Crypto方向，一般来说注重于现代密码学中的Cryptoanalysis（密码分析）。在比赛中，出题人往往会将加密算法公开出来，然后再将密文等信息发送给参赛选手，参赛选手需要分析加密算法中的缺点、漏洞，进而找到相应的利用方法。

Some basic knowledge for Crypto in CTF

> 关于基础知识，边做题便搜索即可。不懂的先搜一下，再问问AI。

基本技能：

1. 了解python和c语言的基础语法。

2. 会编辑和运行代码。

基础知识:

1. 初等数论
2. 线性代数
3. 抽象代数

基本学习方法:

1. 学会使用搜索引擎去寻找有价值的学习资源。

   推介的学习网站：https://ctf-wiki.org/

   > 来点入门网站cryptohack.org

2. 多读书，多看文档，勤于思考。
3. 读大佬的博客文章进行学习。

   诸如

   传奇密码手 d33b4t0：https://d33b4t0.com/ （适合进阶观看）

   糖醋小鸡块：https://tangcuxiaojikuai.xyz/

   Van1sh：https://jayxv.github.io/

4. Get your hands dirty! 在实践中学习，切忌纸上谈兵！在学习知识的基础之上，多去训练场练习（buuctf，攻防世界，等），多参加比赛。（Moe CTF）

对于初学者而言，最为关键的是熟练运用浏览器和大语言模型，只要能够持续学习，入门并不困难。

相关tools

目前CTF种的Crypto主要使用Python

- 如何配置python环境：Python3 环境搭建

> 附件解压后遇见.py文件不要直接双击（运行）。
> 正确打开方式：选中，右键点击.py文件，点击打开方式，选择配置好的环境，就可以查看代码了。

笔者陈列几个建议安装的Python Package:

1.gmpy2

gmpy2是Python的一个扩展库，是对GMP（GNU高精度算术运算库）的封装，其前身为gmpy。
install:

```
pip install gmpy2
```

2.cryptography

几乎必安的一个库，内含多个密码学必要工具。
install:

```
pip install cryptography
```

3.pwntools

CTF Crypto交互题好帮手
install:

```
pip install pwntools
```

4.sage

在环境中安装sage，需要安装sagemath。安装指北Installation Guide。

> Sagemath是一款快速成长且开源的数学软件，提供了非常丰富的代数与数论方面的功能。对于密码人来说，Sagemath的使用体验还是相当不错的。在以后的学习中，你会慢慢感受到它的强大之处的。

笔者建议在 WSL（Windows Subsystem for Linux）里安装 SageMath。初学者或安装有困难者可以暂时跳过。

如何解决困难（diffculties）？

- 尝试利用搜索引擎或LLM（大语言模型）解决，尝试各种问法。（如：怎样提问更能获取有效答案）
- 在充分阅读《提问的智慧》并领会其主要精神后，再去尝试求助群里的Crypto管理员。
- 尝试改变你的求解策略，切忌钻牛角尖。“山重水复疑无路，柳暗花明又一村”

Attention

需要注意：

1. 不要作弊，作弊可耻！
2. 比赛期间，各位师傅请不要公开发布你的Writeup！你的无心之举很有可能将会夺走他人的一次学习机会。

Get your hands dirty!

> 阅读、理解并求解下面的例子！你可以获得第一个flag！

```
#!/usr/bin/env python3
from Crypto.PublicKey import ElGamal
from Crypto.Random import get_random_bytes, random
from Crypto.Util.number import *
from random import *
from secret import flag
def generate_elgamal_keypair(bits=512):
    p = getPrime(bits)

    for _ in range(1000):
        g = getRandomRange(2, 5)
        if pow(g, (p - 1) // 2, p) != 1:
            break

    x = randrange(2, p - 1)
```

```
    y = pow(g, x, p)

    return p, g, y, x
key=generate_elgamal_keypair(bits=512)
p, g, y ,x= key

print("=== 公钥 (p, g, y) ===")
print("p =", p)
print("g =", g)
print("y =", y)
print()

k = randrange(1, p - 2)
m = bytes_to_long(flag)
c1 = pow(g, k, p)
c2 = (m * pow(y, k, p)) % p

print("=== 密文 (c1, c2) ===")
print("c1 =", c1)
print("c2 =", c2)
#不小心把x输出了()
print("x =", x)

"""
=== 公钥 (p, g, y) ===
p =
11540963715962144951763578255357417528966715904849014985547597657698304891044841099894993117258279094910424033273299863589407477091830213468539451196239863
g = 2
y =
83134247833660112870146235827735215953332852913805406894670732122129316484155800652070814497841358357112053241866624825263578340420134007654219252742718 53

=== 密文 (c1, c2) ===
c1 =
66520535530556453582753622595548565259769318413182511529404645431751085601329496109160124908379708511912041447574093350118118748960564301052925342447328 63
c2 =
23149135680815264282479817191009523314449388523990318266354759719474846634183625333635914412165705974177891204707035488433421705670393998303774592282979 83
x =
80109570780865542840209596641247844796109135965600350119511432695597612291140277387914409618641502257980491205825409518749562551158845393339664290210042 14
"""
```

> 注:此处的secret模块可以理解为存储了明文flag的python文件，属于未知信息，而非公开的可用模块。你需要尝试从密文去恢复原始的flag，而不是去下载secret模块。secrets则是一个用于生成管理密码的安全随机数的python标准库。
> bytes_to_long函数用于将flag编码为整数。

Hint：什么是`Elgamal`

> 还是附个链接吧Elgamal