**西安电子科技大学网信院**

**信息安全基础与密码学**

**综合实验**

**实 验 报 告（四）**

**ElGamal公钥密码算法**

**班级：**

**姓名：**

**学号：**

**日期：2022年11月19日**

一、实验目的（包括实验环境、实现目标等等）

实验环境

Windows11 , Python 3.9.12

实验目标：

1.通过算法编程，熟悉大整数的基本函数操作

2.通过编程实现基于中国剩余定理的秘密共享方案，加深对于ELGamal加密体制的理解

二、方案设计

（包括背景、原理、必要的公式、图表、算法步骤等等）

实验背景

ElGamal加密算法是一个基于Diffie–Hellman密钥交换的非对称加密算法，它在1985年由塔希尔·盖莫尔提出。

EIGamal公开密钥密码体制是基于有限域中离散对数问题的难解性。它所根据的原理是：求解离散对数是困难的，而其逆运算可以应用平方乘的方法有效的计算出来。在相应的群 G中，指数函数是单向函数

实验原理

强素数

在密码学中，一个素数在满足下列条件时被称为强素数:

1. 必须是很大的数。

2.  有很大的质因数。也就是说，对于某个整数  以及大素数  ，我们有  。

3.  有很大的质因数。也就是说，对于某个整数  以及大素数  ，我们有  。

4.  有很大的质因数。也就是说，对于某个整数  以及大素数  ，我们有  。

有时，当一个素数只满足上面一部分条件的时候，我们也称它是强素数。而有的时候，我们则要求加入更多的条件。例如，我们可以要求  ，或者 。从这个角度上来说，很大的**安全素数**(形如p=2q+1)可以看作是强素数的一种。

EIGamal加密算法由三部分组成：密钥生成、加密和解密。

密钥生成的步骤如下:

Alice随机产生一个大素数及模的一个原根

Alice从  中随机选择一个。

Alice计算  。

Alice公开 的描述作为其公钥，并保留作为其私钥。私钥必须保密。

加密过程:

Bob将明文消息  加密成密文

(1) 随机选取一个整数 

(2) 计算 , 

(3) 将密文  发送给Alice。

解密过程:

Alice利用私钥对密文进行解密的算法工作方式如下:

Alice计算共享秘密 

然后计算 

本原根的生成:

算法的难题在于生成大素数p的本原根，生成本原根有多种方法，比如遍历判断等，但是这里的方法是：

随机生成一个大素数，当为素数时，则一定为素数。

生成一个随机数，若且，则为大素数的本原根。

证明:

由题意可知是素数，且

由Fermat小定理可以得到

因此





因为p是一个强素数，因此p-1的因子只有1，2，q，p-1

算法步骤

(1) 输入秘密,大素数位数n

(2) 生成一个随机n位的素数，计算

(3) 生成随机数，且

(4) 验证是否且，如果否，返回(3)

(5) 生成随机整数，计算，传递公钥

(6) 生成随机整数k,计算 ,传递加密信息

(7) 计算 计算的逆

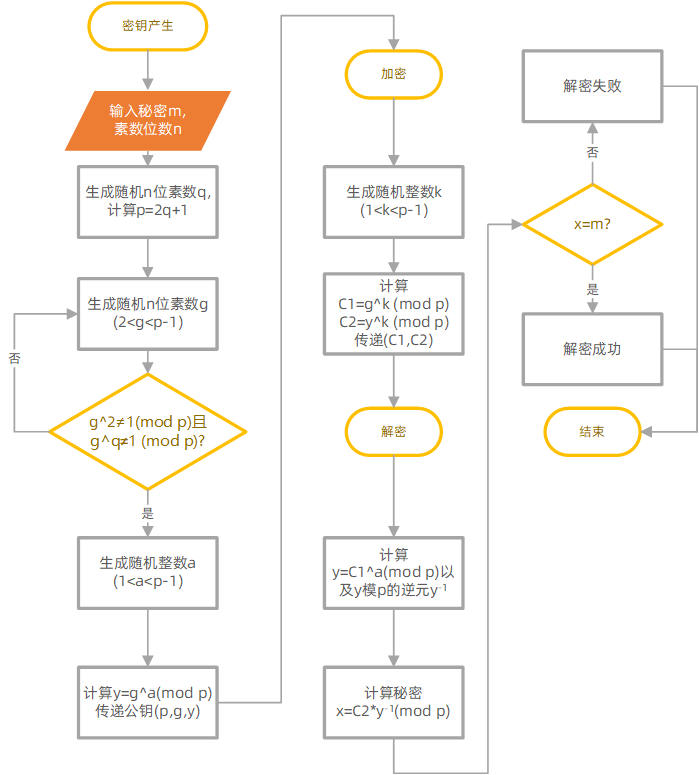
(8) 计算秘密

(7) 比较恢复的秘密是否与原秘密相同

三、方案实现

（包括算法流程图、主要函数的介绍、算法实现的主要代码等等）

算法流程图



主要函数介绍

get\_safe\_prime()生成安全素数q以及强素数p

get\_generator()寻找p的原根

def keygen() 生成私钥a以及公钥

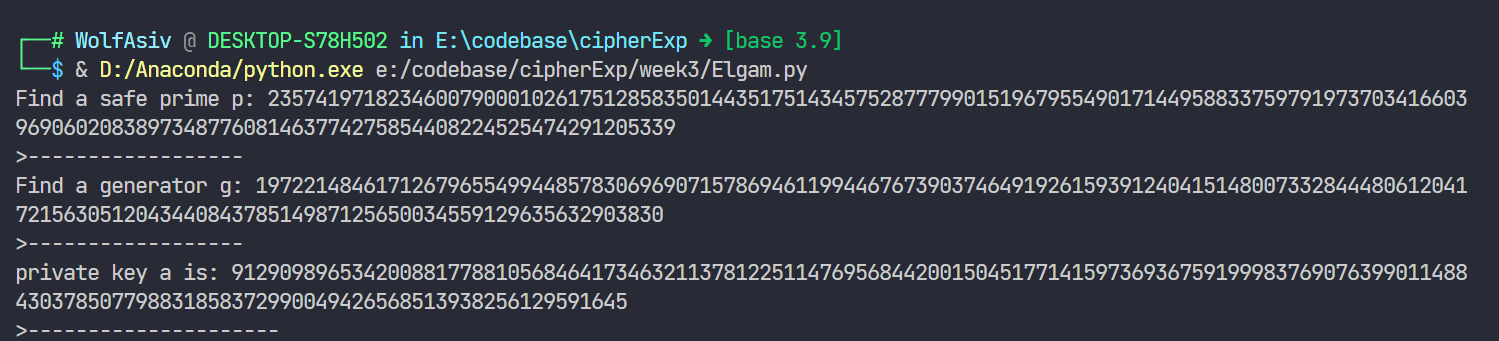
def encrypt()生成临时秘钥x，以及公钥加密

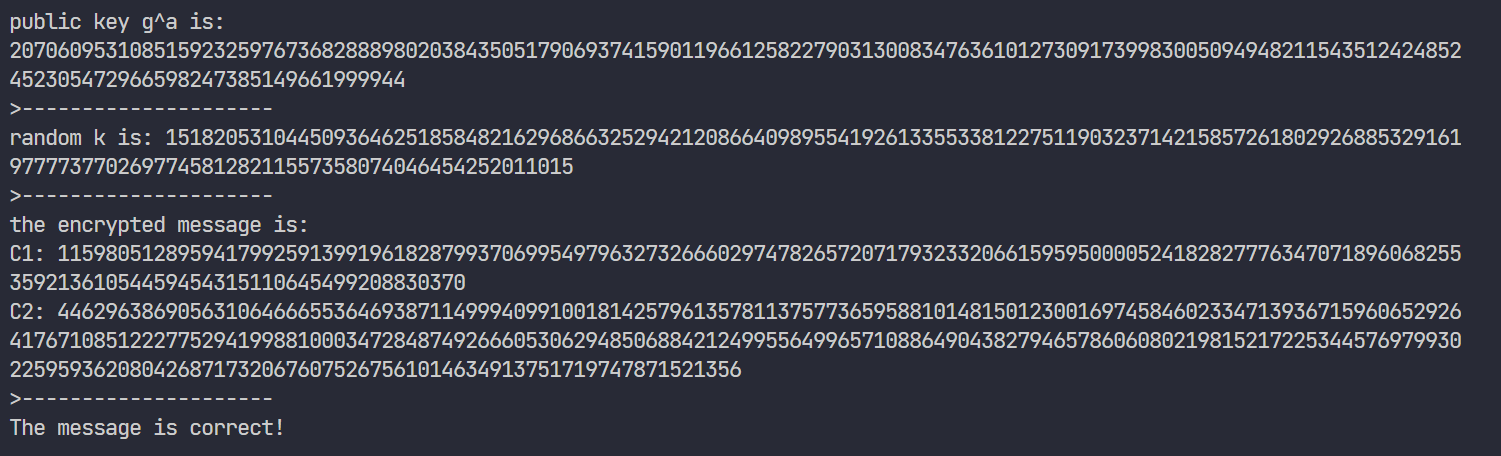
def decrypt()解密

实现代码

1. *#ElGamal公钥密码算法*
2. from Crypto.Util.number import \*
3. import random
4. import gmpy2
5. def get\_safe\_prime(bits=1024):
6. *#生成随机大素数q*
7. while True:
8. q=getPrime(bits)
9. *#生成p=2q+1*
10. p=2\*q+1
11. *#判断p是否为素数*
12. if isPrime(p):
13. return p,q
14. def get\_generator(p,q):
15. while True:
16. g=random.randint(2,p-1)
17. *#因为p是一个强素数，因此只有1，2，q，p-1.因此只需要验证2和q即可*
18. if gmpy2.powmod(g,2,p)!=1 and gmpy2.powmod(g,q,p)!=1:
19. return g
20. *#秘钥生成*
21. def keygen(nbits):
22. *#生成强素数p*
23. *# In this context p is a strong prime if p-1 and p+1 have at least one large prime factor.*
24. *# N should be a multiple of 128 and > 512.*
25. p,q=get\_safe\_prime(nbits)
26. print("Find a safe prime p:",p)
27. print(">------------------")
28. *#求原根*
29. g=get\_generator(p,q)
30. print("Find a generator g:",g)
31. print(">------------------")
32. x=random.randint(1,p-2)
33. print("private key a is:",x)
34. print(">---------------------")
35. *#计算y=g^x(mod p)*
36. y=gmpy2.powmod(g,x,p)
37. print("public key g^a is:")
38. print(y)
39. print(">---------------------")
40. return p,g,y,x
41. *#加密*
42. def encrypt(m,p,g,y):
43. *#生成随机大素数k，满足1≤k≤p-2 k在每次加密中都不同*
44. k=random.randint(1,p-2)
45. print("random k is:",k)
46. print(">---------------------")
47. *#计算a=g^k(mod p)*
48. y1=gmpy2.powmod(g,k,p)
49. *#计算b=m\*y^k(mod p)*
50. y2=m\*gmpy2.powmod(y,k,p)
51. return y1,y2
52. *#解密*
53. def decrypt(y1,y2,p,x):
54. *#计算y1^x(mod p)*
55. y=gmpy2.powmod(y1,x,p)
56. *#计算y^-1(mod p)*
57. y=gmpy2.invert(y,p)
58. *#计算m=y2\*y(mod p)*
59. m=(y2\*y)%p
60. return m
61. if \_\_name\_\_ == '\_\_main\_\_':
62. *# m=int(input("input the message:"))*
63. m=9327260388393076415930260479153046010064951650867096323260782111903507989221223155043829435161867334962529353992935468294479465522637777146290777
64. n=512
65. p,g,y,x=keygen(n)
66. y1,y2=encrypt(m,p,g,y)
67. print("the encrypted message is:")
68. print("C1:",y1)
69. print("C2:",y2)
70. print(">---------------------")
71. decrypted\_m=decrypt(y1,y2,p,x)
72. if decrypted\_m==m:
73. print("The message is correct!")
74. else:
75. print("The message is wrong!")

四、数据分析(包括算法测试数据的分析，运行结果截图等等)





五、思考与总结

1. 请简述什么是本原根，给定素数P，如何求其本原根？。

答:在  时，定义对模的阶 为使 成立的最小的正整数。若 , 则称 是模的原根。

求本原根:

求任何一个质数x的任何一个原根，一般就是枚举2到x-1，并检验。

有一个方便的方法就是，求出x-1所有不同的质因子p1,p2...pm，对于任何,判定a是否为x的原根，只需要检验这m个数中，是否存在一个数mod x为1，若存在，a不是x的原根，否则就是x的原根。

1. 如果𝑘与𝑝−1不互素，可能会发生什么情况？

答：会有机会暴露原文,由费马小定理可得



如果p-1和k不互素，可以得到



因而Bob在加密过程时

1. 实验过程中还遇到了什么问题，如何解决的？通过该实验有何收获？

答:从线性寻找原根，效率非常低效，因此去指定2q+1的安全素数，且只去检验2和q不满足和