



# 南昌大学实验报告

学生姓名： 丁俊 学 号： 8003119100 专业班级： 信息安全 193 班  
实验类型： ☐ 验证 ☐ 综合 ☐ 设计 ☐ 创新 实验日期： 2021.11.26 实验成绩：       

## 一、实验项目名称

程序实现 Diffie-Hellman 密钥交换协议

## 二、实验目的

- 1、理解 DES 算法的概念及原理
- 2、理解 DES 算法的加密流程

## 三、实验基本原理

### 1. 算法原理

(1) 有两个全局公开的参数，一个素数  $p$  和一个整数  $a$ ， $a$  是  $p$  的一个原根（对于正整数  $\gcd(a, m) = 1$ ，如果  $a$  是模  $m$  的原根，那么  $a$  是整数模  $m$  乘法群的一个生成元）；

(2) 假设用户 A 和 B 希望交换一个密钥，用户 A 选择一个作为私有密钥的随机数  $X_A < p$ ，并计算公开密钥  $Y_A = a^{X_A} \bmod p$ ，A 对  $X_A$  的值保密存放而使  $Y_A$  能被 B 公开获得。类似地，用户 B 选择一个私有的随机数  $X_B < p$ ，并计算公开密钥  $Y_B = a^{X_B} \bmod p$ 。B 对  $X_B$  的值保密存放而使  $Y_B$  能被 A 公开获得。

(3) 用户 A 产生共享秘密密钥的计算方式是  $K = (Y_B)^{X_A} \bmod p$ 。同样，用户 B 产生共享秘密密钥的计算是  $K = (Y_A)^{X_B} \bmod p$ 。这两个计算产生相同的结果。

### 2. 关于原根的生成

在求解原根的时候采用的是暴力求解的方法，按照的规定就是  $a$  是  $p$  的原根，就是  $a^{(p-1)} = 1 \pmod p$  当且仅当指数为  $p-1$  的时候成立，所以我使用的办法就是取  $a$  在  $[2, p-1]$  之间进行取值，若是其中有满足上述式子的  $a$  则放到列表中，最后在自动生成原根的时候是人为的取得最大的那个数值作为原根。

例如当  $p = 7$  时， $a$  从 2 开始取值，因为  $2^3 = 1 \pmod 7$ ,  $3 \neq 6$ , 所以 2 不是  $p$  的原根；

当  $a = 3$  时,  $3^1 = 3 \pmod 7$ ,  $3^2 = 2 \pmod 7$ ,  $3^3 = 6 \pmod 7$ ,  $3^4 = 4 \pmod 7$ ,  $3^5 = 5 \pmod 7$ ,  $3^6 = 1 \pmod 7$

所以  $p = 7$  的一个原根就是 3，同样的道理可以求出  $p$  的所有的原根。

### 3、算法实现流程

(1) 提示用户输入一个素数  $p$ ，判断用户输入是否为素数，若是则生成其生成元  $a$ ，若不是则提示用户继续输入，直到输入符合要求为止；

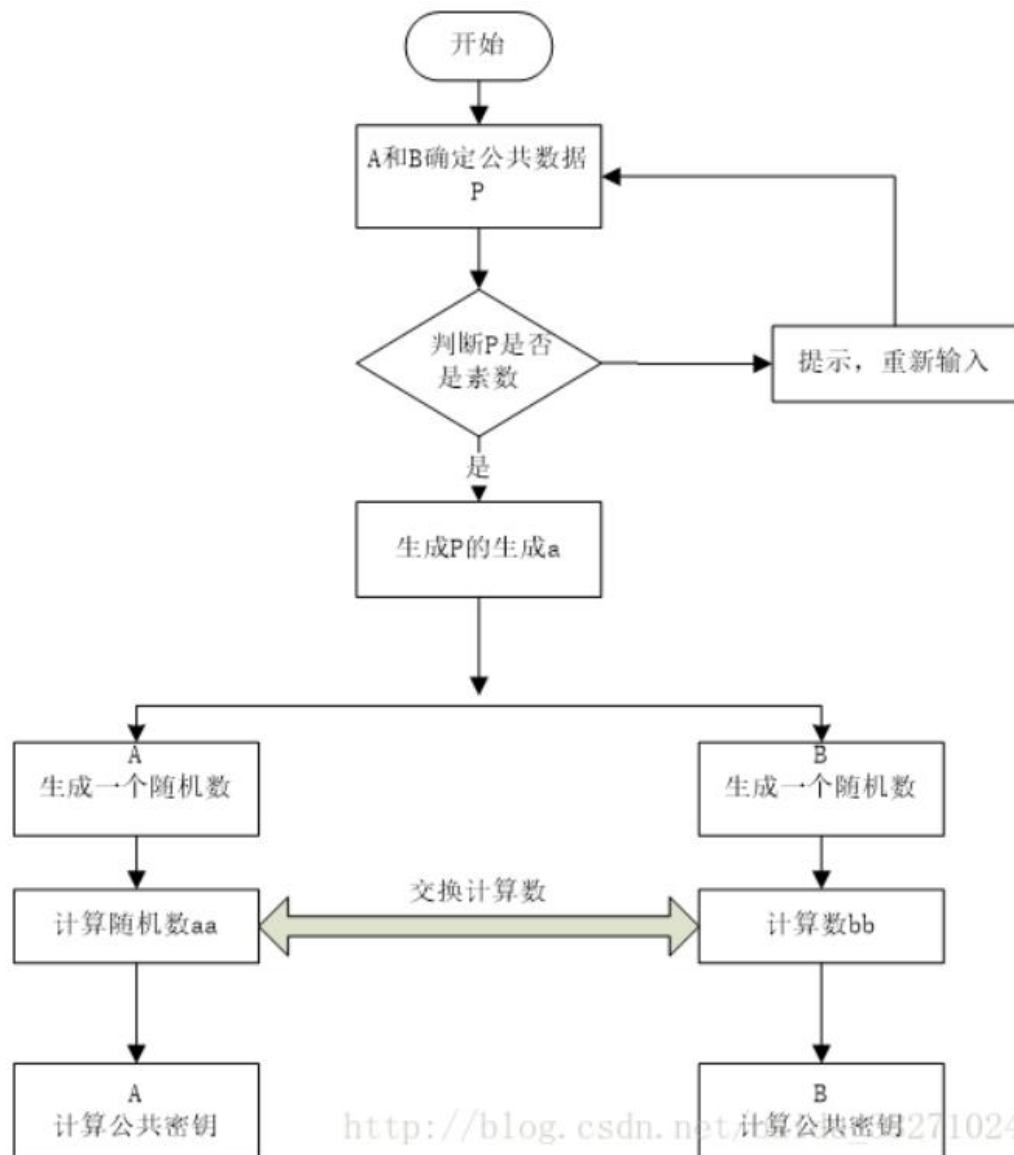
(2) 定义函数 `judge_prime` 判断用户输入是否为素数；

(3) 定义函数 `get_generator` 来得到  $p$  的生成元  $a$ ；

(4) A,B 各自选择小于  $p$  随机数作为自己的私钥  $X_A, X_B$ ，并根据  $Y_a = a^{X_a} \bmod p$ ,  $Y_b = a^{X_b} \bmod p$ ，得到各自的计算数；

(5) 交换彼此的计算数；

(6) 对对方交换的数进行计算，若是得到的结果相同则是公共密钥。



## 四、主要仪器设备及耗材

Windows 操作系统, DevCpp

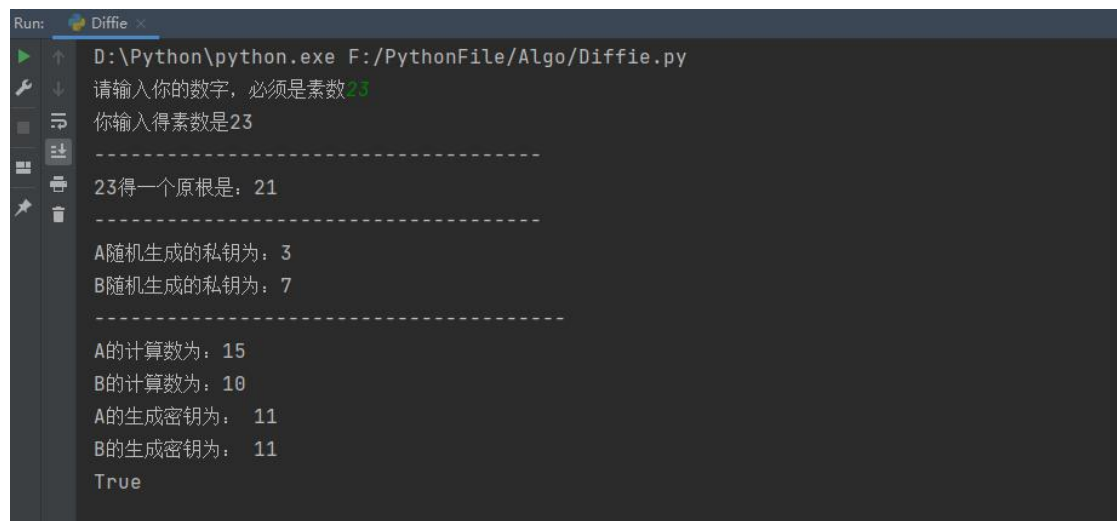
## 五、实验步骤

```
1. # 判断是否是素数
2. def is_prime(p):
3.     if p <= 1:
4.         return False
5.     i = 2
6.     while i < math.sqrt(i):
7.         if p % i == 0:
8.             return False
9.         i += 1
10.    return True
11.
12.
13. # 获得所有原根
14. def get_generator(p):
15.     a = 2
16.     gist = []
17.     while a < p:
18.         flag = 1
19.         while flag != p:
20.             if (a ** flag) % p == 1:
21.                 break
22.             flag += 1
23.             if flag == (p - 1):
24.                 gist.append(a)
25.             a += 1
26.     return gist
27.
28.
29. def calculation(a, x, p):
30.     return (a ** x) % p
31.
32.
33. def get_key(X, Y, p):
34.     return (Y ** X) % p
35.
36.
37. # Diffie 密钥交换算法
38. def Diffie_main():
```

```
39.     # 输入得数要是素数
40.     flag = False
41.     while flag == False:
42.         print('请输入你的数字，必须是素数', end='')
43.         p = input()
44.         p = int(p)
45.         flag = is_prime(p)
46.     print('你输入得素数是' + str(p))
47.
48.     ge_list = get_generator(p)
49.     print('-----')
50.     print(str(p) + '得一个原根是: ', end='')
51.     print(ge_list[-1])
52.     ge = ge_list[-1] # 得到生成元
53.     print('-----')
54.
55.     # 得到 A 得私钥
56.     Xa = random.randint(0, p - 1)
57.     print('A 随机生成的私钥为: %d' % Xa)
58.
59.     # 得到 B 的私钥
60.     Xb = random.randint(0, p - 1)
61.     print('B 随机生成的私钥为: %d' % Xb)
62.     print('-----')
63.
64.     Ya = calculation(ge, Xa, p)
65.     print('A 的计算数为: %d' % Ya)
66.
67.     Yb = calculation(ge, Xb, p)
68.     print('B 的计算数为: %d' % Yb)
69.
70.     # 交换后 A 的密钥
71.     Key_A = get_key(Xa, Yb, p)
72.     print('A 的生成密钥为: %d' % Key_A)
73.
74.     # 交换后 B 的密钥
75.     Key_B = get_key(Xb, Ya, p)
76.     print('B 的生成密钥为: %d' % Key_B)
77.
78.     print(Key_A == Key_B)
79. if __name__ == '__main__':
80.     Diffie_main()
```

## 六、实验数据及处理结果

如图 1，用户输入一个素数后，A 和 B 自动生成各自的私钥，并交换各自的计算数，最后利用双方自己的私钥计算协商密钥。



```
Run: Diffie <
D:\Python\python.exe F:/PythonFile/Algo/Diffie.py
请输入你的数字，必须是素数23
你输入得素数是23
-----
23得一个原根是：21
-----
A随机生成的私钥为：3
B随机生成的私钥为：7
-----
A的计算数为：15
B的计算数为：10
A的生成密钥为： 11
B的生成密钥为： 11
True
```

图 1

## 七、思考讨论题或体会或对改进实验的建议

无

## 八、参考资料

现代密码学第 4 版