**Лабораторная работа №2 по теме «Протокол Диффи-Хеллмана»**

Выполнил студент гр.Б8117-02.03.01

Михайлов Денис

**Постановка задачи:**

Реализовать протокол Диффи-Хеллмана.

**Описание метода:**

**Протокол Диффи-Хеллмана:**

Протокол Диффи-Хеллмана – это криптографический протокол, позволяющий двум и более сторонам получить общий секретный ключ, используя незащищённый от прослушивания канал связи. Полученный ключ используется для шифрования дальнейшего обмена с помощью алгоритмов симметричного шифрования.

**Алгоритм**

Предположим, существуют два абонента: Алиса и Боб. Обоим абонентам известны некоторые два числа и , которые не являются секретными и могут быть известны также другим заинтересованным лицам. Для того, чтобы создать неизвестный более никому секретный ключ, оба абонента генерируют большие случайные числа: Алиса – число , Боб – число .

Далее, Алиса вычисляет число по следующей формуле:

и пересылает его Бобу, а Боб вычисляет число :

и передаёт его Алисе.

На следующем этапе Алиса на основе имеющегося у неё числа и полученного по сети числа вычисляет следующее значение:

Боб, соответственно, вычисляет следующее значение:

У Алисы и Боба получилось одно и то же число

Его они и могут использовать в качестве секретного ключа.

В практических реализациях для и используются числа порядка и порядка . Число не обязано быть большим и обычно имеет значение в пределах первого десятка.

Для нахождения просто числа будем использовать метод Миллера-Рабина.

**Проведение тестов.**

Простое число:  239

Свободный ключ А:  10

Свободный ключ B:  100

Секретный ключ А:  10

Секретный ключ B:  10

Как можно заметить из теста, секретные ключи совпадают, что говорит о правильности метода.

**Код программы.**

# -\*- coding: utf-8 -\*-

from random import randint

def Prime\_numb(k=300):              #Calculated random Prime number

    string = '1'

    for \_ in range(k-2, -1, -1):

        string += str(randint(0, 1))

    p = int(string, 2)

    if p % 2 != 0:

        p += 1

    p += 1

    while(True):

        if MR(p, k) and MR(2\*p+1, k):

            return 2\*p+1

        p += 2

def MR(numb, k=300):                #The test of Miller-Rabin on simplicity

    d, s = factor(numb-1)

    if numb <= 1:

        return False

    if numb <= 3:

        return True

    if numb % 2 == 0:

        return False

    for \_ in range(k):

        a = randint(2, numb-2 )

        x = Expon(a, d, numb)

        if not (x == 1 or x == numb - 1):

            for j in range(s-1):

                x = Expon(a, 2\*\*(j+1) \* d, numb)

                if x == 1: return False

                if x == numb-1: break

        else: return True

    return True

def factor(numb):

    n = numb

    count = 0

    while(True):

        if n % 2 == 1: return int(n), count

        n /= 2

        count += 1

def euler(n):                       #The function of Euler

    r = n

    i = 2

    while i\*i <= n:

        if n % i == 0:

            while n % i == 0:

                n //= i

            r -= r//i

        else:

            i += 1

    if n > 1:

        r -= r//n

    return r

def isQ(mod):                       #Calculating the primitive root modulus p

    count = 100

    k = euler(mod)

    while(True):

        if Expon(count, k, mod) == 1 and count != mod:

            return count

        count += 1

def Expon(value, degree, mod):      #Exponentiation modulus

    c = 1

    for \_ in range(degree):

        c = c \* value % mod

    return c

a = randint(10\*\*5-1, 10\*\*6-1)   #User X's Private key

b = randint(10\*\*5-1, 10\*\*6-1)   #User Y's Private key

p = Prime\_numb(7)    #A Prime number that (p-1)/2 is also a Prime number (The Number Of Sophie Germain)

print('Простое число: ', p, '\n')

g = isQ(p)     #Primitive root modulus p (Calculated using the "isQ" function)

A = Expon(g, a, p)      #User X's Free key

B = Expon(g, b, p)      #User Y's Free key

print('Свободный ключ юзера А: ', A, '\n')

print('Свободный ключ юзера B: ', B, '\n')

# User X and user Y do not know each other's Private keys

# User X gets a Free key from User Y

# User Y gets a Free key from User X

K1 = Expon(B, a, p)     #User X calculated general secret key

K2 = Expon(A, b, p)     #User Y calculated general secret key

print('Секретный ключ юзера А: ', K1, '\n')

print('Секретный ключ юзера B: ', K2, '\n')

if K1 == K2:            #User X's and User Y's general secret keys're identical

    print(K1)