МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теоретических основ компьютерной безопасности и криптографии

**Протокол аутентификации Шнорра**

ОТЧЁТ

ПО ДИСЦИПЛИНЕ

«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»

студента 5 курса 531 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Таранова Алексея Вадимовича

|  |  |  |
| --- | --- | --- |
| Преподаватель  аспирант | \_\_\_\_\_\_\_\_\_\_ | Р. А. Фарахутдинов |
|  | подпись, дата |  |

Саратов 2023

**1. Теоретическая часть**

Цель работы – изучение протокола аутентификации Шнорра и его программная реализация.

* Генерация общих параметров:

Для генерации общих параметров выбираются два простых числа: случайное и случайное длины бит, такое, что . Затем выбирается случайное , такое что . Числа , и являются общими параметрами системы и находятся в свободном доступе для всех участников системы.

* Генерация ключей:

Для генерации конкретной пары ключей выбирается случайное число , меньшее . Оно служит закрытым ключом Пегги. Затем вычисляется . Оно является открытым ключом Пегги.

* Шаги протокола:

1. Пегги выбирает случайное число , меньшее , и вычисляет , который отправляет Виктору.

2. Виктор посылает Пегги случайное число , из диапазона от до , где – любое целое число, большее 1.

3. Пегги вычисляет и посылает Виктору.

4. Виктор проверяет, что .

**2. Описание программы**

Для вычисления обратных чисел используется расширенный алгоритм Евклида (extended\_euclid(x1, x2)).

Главная функция main() спрашивает каждый шаг алгоритма нужно ли его выполнять или использовать ранее сохраненные значения. Рассмотрим случай, что пользователь выполняет все шаги. Тогда сначала генерируются общие параметры системы (create p), затем генерируется новый ключ для Пегги в функции create\_keys(), после чего вычисляется x и отправляется Виктору (first\_step()). Следующим шагом генерируется случайное число е и отправляется Пегги (функция second\_step()) и затем в функции third\_step() Пегги вычисляет y и посылает его Виктору. Наконец, в функции last\_step() проверяется, что . Значения вычисляемых в шагах переменных сохраняется в текстовых файлах.

**3. Пример запуска программы**

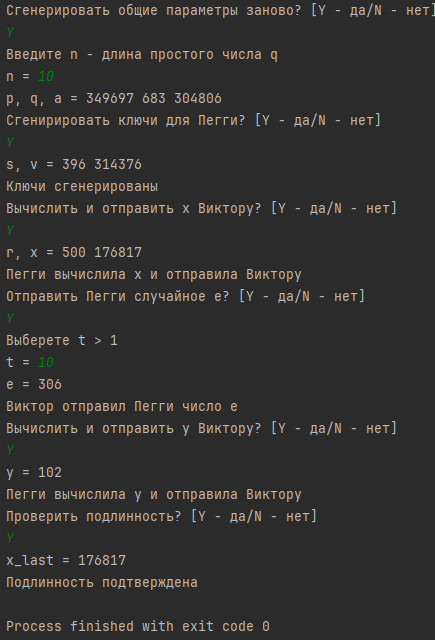
****

Рисунок 1 – Результат работы программы

**ПРИЛОЖЕНИЕ А**

Листинг 1 – main.py

import random

from sympy import isprime

def extended\_euclid(x1, x2):

o\_r, r = x1, x2

o\_s, s = 1, 0

o\_t, t = 0, 1

while r != 0:

q = o\_r // r

o\_r, r = r, o\_r - q \* r

o\_s, s = s, o\_s - q \* s

o\_t, t = t, o\_t - q \* t

return o\_r, o\_s, o\_t

def create\_p():

print("Введите n - длина простого числа q\nn = ", end='')

exp = int(input())

q = random.randint(pow(2, exp - 1), pow(2, exp))

while not isprime(q):

q = random.randint(pow(2, exp - 1), pow(2, exp))

n = 2

while not isprime(q \* (2 \*\* n) + 1):

n += 1

p = q \* (2 \*\* n) + 1

a = random.randint(2, p - 2)

a = pow(a, 2 \*\* n, p)

f = open('shared\_keys.txt', 'w')

f.write(str(p) + ' ' + str(q) + ' ' + str(a))

f.close()

print("p, q, a =", p, q, a)

return 0

def create\_keys():

f = open('shared\_keys.txt', 'r')

p, q, a = map(int, f.read().split())

f.close()

if q == 2 or q == 3:

s = 2

else:

s = random.randint(2, q - 1)

v = pow(a, s, p)

\_, v, \_ = extended\_euclid(v, p)

if v < 0:

v += p

f = open('close\_keys.txt', 'w')

f.write(str(s))

f.close()

f = open('open\_keys.txt', 'w')

f.write(str(v))

f.close()

print("s, v =", s, v)

return 0

def first\_step():

f = open('shared\_keys.txt', 'r')

p, q, a = map(int, f.read().split())

f.close()

r = random.randint(2, q - 1)

f = open('r.txt', 'w')

f.write(str(r))

f.close()

x = pow(a, r, p)

f = open('x.txt', 'w')

f.write(str(x))

f.close()

print("r, x =", r, x)

return 0

def second\_step():

print("Выберете t > 1\nt = ", end='')

while True:

t = int(input())

if t > 1:

break

print("t введено неверно")

e = random.randint(0, pow(2, t - 1))

f = open('e.txt', 'w')

f.write(str(e))

f.close()

print("e =", e)

return 0

def third\_step():

f = open('close\_keys.txt', 'r')

s = int(f.read())

f.close()

f = open('shared\_keys.txt', 'r')

\_, q, \_ = map(int, f.read().split())

f.close()

f = open('r.txt', 'r')

r = int(f.read())

f.close()

f = open('e.txt', 'r')

e = int(f.read())

f.close()

y = (r + s \* e) % q

f = open('y.txt', 'w')

f.write(str(y))

f.close()

print("y =", y)

return 0

def last\_step():

f = open('x.txt', 'r')

x = int(f.read())

f.close()

f = open('shared\_keys.txt', 'r')

p, \_, a = map(int, f.read().split())

f.close()

f = open('open\_keys.txt', 'r')

v = int(f.read())

f.close()

f = open('e.txt', 'r')

e = int(f.read())

f.close()

f = open('y.txt', 'r')

y = int(f.read())

f.close()

x\_last = (pow(a, y, p) \* pow(v, e, p)) % p

print("x\_last =", x\_last)

return x == x\_last

def main():

print("Сгенерировать общие параметры заново? [Y - да/N - нет]")

while True:

f = input()

if f == 'Y':

create\_p()

break

elif f == 'N':

break

else:

print("Некорректный ввод")

print("Сгенирировать ключи для Пегги? [Y - да/N - нет]")

while True:

f = input()

if f == 'Y':

create\_keys()

print("Ключи сгенерированы")

break

elif f == 'N':

return 0

else:

print("Некорректный ввод")

print("Вычислить и отправить х Виктору? [Y - да/N - нет]")

while True:

f = input()

if f == 'Y':

first\_step()

print("Пегги вычислила x и отправила Виктору")

break

elif f == 'N':

return 0

else:

print("Некорректный ввод")

print("Отправить Пегги случайное е? [Y - да/N - нет]")

while True:

f = input()

if f == 'Y':

second\_step()

print("Виктор отправил Пегги число e")

break

elif f == 'N':

return 0

else:

print("Некорректный ввод")

print("Вычислить и отправить у Виктору? [Y - да/N - нет]")

while True:

f = input()

if f == 'Y':

third\_step()

print("Пегги вычислила у и отправила Виктору")

break

elif f == 'N':

return 0

else:

print("Некорректный ввод")

print("Проверить подлинность? [Y - да/N - нет]")

while True:

f = input()

if f == 'Y':

if last\_step():

print("Подлинность подтверждена")

else:

print("Подлинность не подтверждена")

break

elif f == 'N':

return 0

else:

print("Некорректный ввод")

return 0

main()