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 обмена ключами. Kerberos

ОТЧЁТ
ПО ДИСЦИПЛИНЕ
«КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ»

студентки 5 курса 531 группы
специальности 10.05.01 Компьютерная безопасность
факультета компьютерных наук и информационных технологий
Шуликиной Анастасии Александровны

Преподаватель Р. А. Фарахутдинов
_____
подпись, дата

Саратов 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В данной лабораторной работе поставлена задача рассмотрения протокола Kerberos, на основе изученного материала программно реализовать протокол Kerberos.

1 Цель работы и порядок её выполнения

Цель работы – изучение протокол Kerberos и реализовать его.

Порядок выполнения работы:

1. Разобрать, что такое протокол обмена ключами Kerberos.
2. Произвести программную реализацию по созданию протокола Kerberos.

2 Теория

2.1 Протокол обмена ключами Kerberos

Протокол (протокол) – описание последовательности алгоритмов, в процессе выполнения которой два или более участников последовательно исполняют эти алгоритмы и обмениваются сообщениями с целью решения некоторой поставленной перед ними задачи.

В качестве участников (субъектов, сторон) протокола могут выступать не только пользователи или абоненты, но и клиентские и серверные приложения.

Kerberos – сетевой протокол аутентификации, который предлагает механизм взаимной аутентификации клиента и сервера перед установлением связи между ними. Kerberos выполняет аутентификацию в качестве службы аутентификации доверенной третьей стороны, используя криптографический ключ, при условии, что пакеты, проходящие по незащищенной сети, могут быть перехвачены, модифицированы и использованы злоумышленником. Kerberos построен на криптографии симметричных ключей и требует наличия центра распределения ключей.

Централизованное распределение ключей симметричного шифрования подразумевает, что у каждого абонента сети есть только один основной ключ, который используется для взаимодействия с центром распределения ключей (сервером ключей). Чтобы получить ключ шифрования для защиты обмена данными с другим абонентом, пользователь обращается к серверу ключей, который назначает этому пользователю и соответствующему абоненту сеансовый симметричный ключ.

Протокол Kerberos обеспечивает распределение ключей симметричного шифрования и проверку подлинности пользователей, работающих в незащищенной сети. Реализация Kerberos - это программная система, построенная по архитектуре "клиент-сервер". Клиентская часть устанавливается на все компьютеры защищаемой сети, кроме тех, на которые устанавливаются компоненты сервера Kerberos. В роли клиентов Kerberos могут, в частности, выступать и сетевые серверы (файловые серверы, серверы печати и т.д.).

Аутентификация выполняется следующим образом:

1. Клиент посылает запрос серверу аутентификации (Authentication Server, AS) на информацию, однозначно идентифицирующую некоторый нуж-

ный клиенту сервер.

2. Сервер AS передает требуемую информацию, зашифрованную с помощью известного пользователю ключа. Переданная информация состоит из билета сервера и временного ключа, предназначенного для шифрования (часто называемого ключом сеанса).
3. Клиент пересылает серверу билет, содержащий идентификатор клиента и ключ сеанса, зашифрованные с помощью ключа, известного серверу.
4. Теперь ключ сеанса известен и клиенту, и серверу. Он может быть использован для аутентификации клиента, а также для аутентификации сервера. Ключ сеанса можно применять для шифрования передаваемой в сеансе информации или для взаимного обмена ключами подсеанса, предназначенными для шифрования последующей передаваемой информации.

Протокол Kerberos функционирует на одном или нескольких серверах аутентификации, работающих на физически защищенном хосте. Серверы аутентификации ведут базы данных партнеров по обмену информацией в сети (пользователей, серверов и т. д.) и их секретных ключей. Программный код, обеспечивающий функционирование самого протокола и шифрование данных, находится в специальных библиотеках. Для того чтобы выполнять аутентификацию Kerberos для своих транзакций, приложения должны сделать несколько обращений к библиотекам Kerberos. Процесс аутентификации состоит из обмена необходимыми сообщениями с сервером аутентификации Kerberos.

Алгоритм протокола Kerberos:

3 Программная реализация

3.1 Результаты тестирования программы

На рисунках 1-5 можно увидеть работу, реализуемой программы, по рассмотренным алгоритмам.

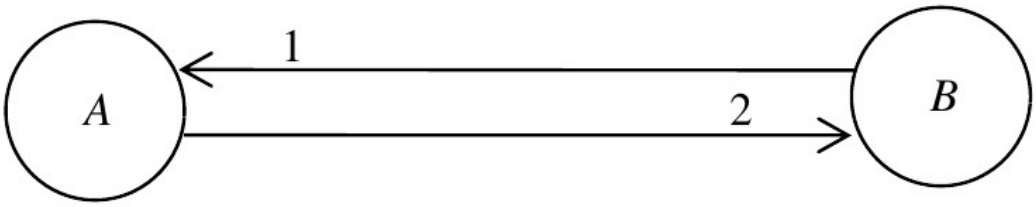

Рисунок 1 – Тест программы для отношения эквивалентности

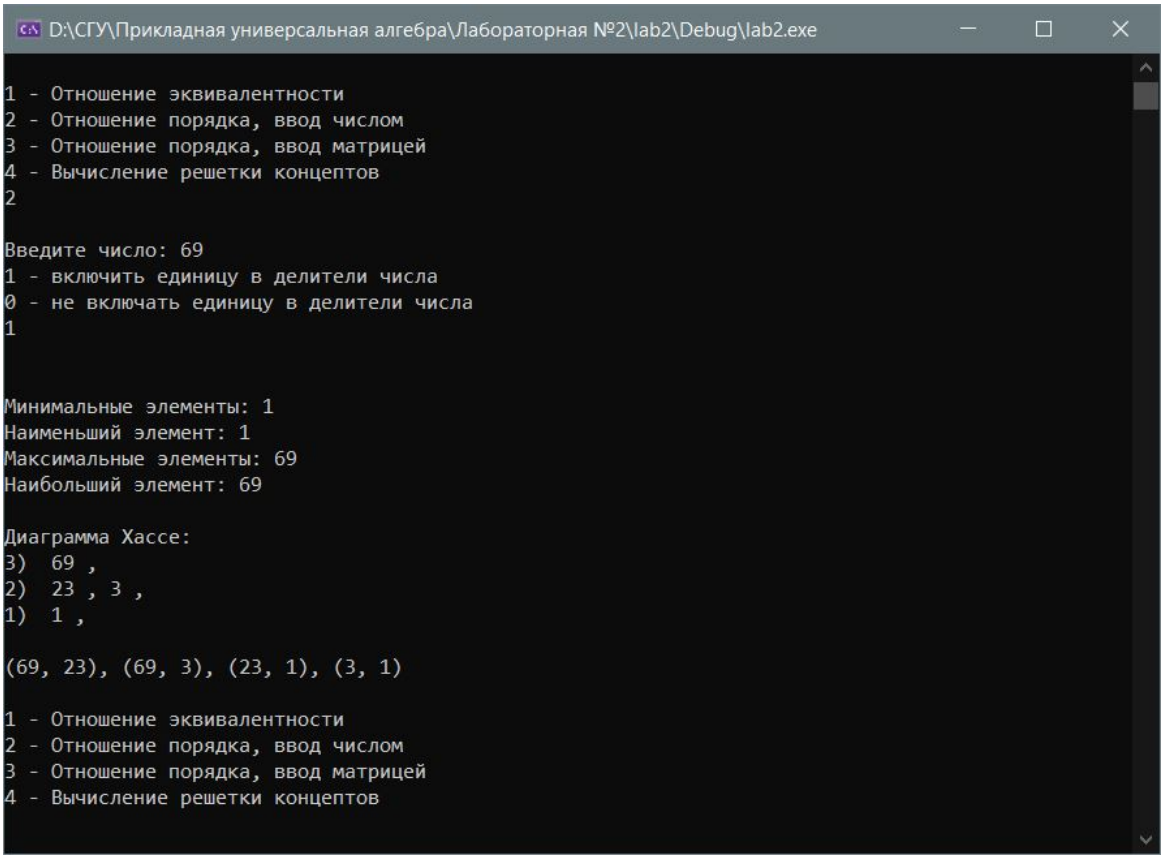

Рисунок 2 – Тест программы для отношения порядка числа(включая 1)

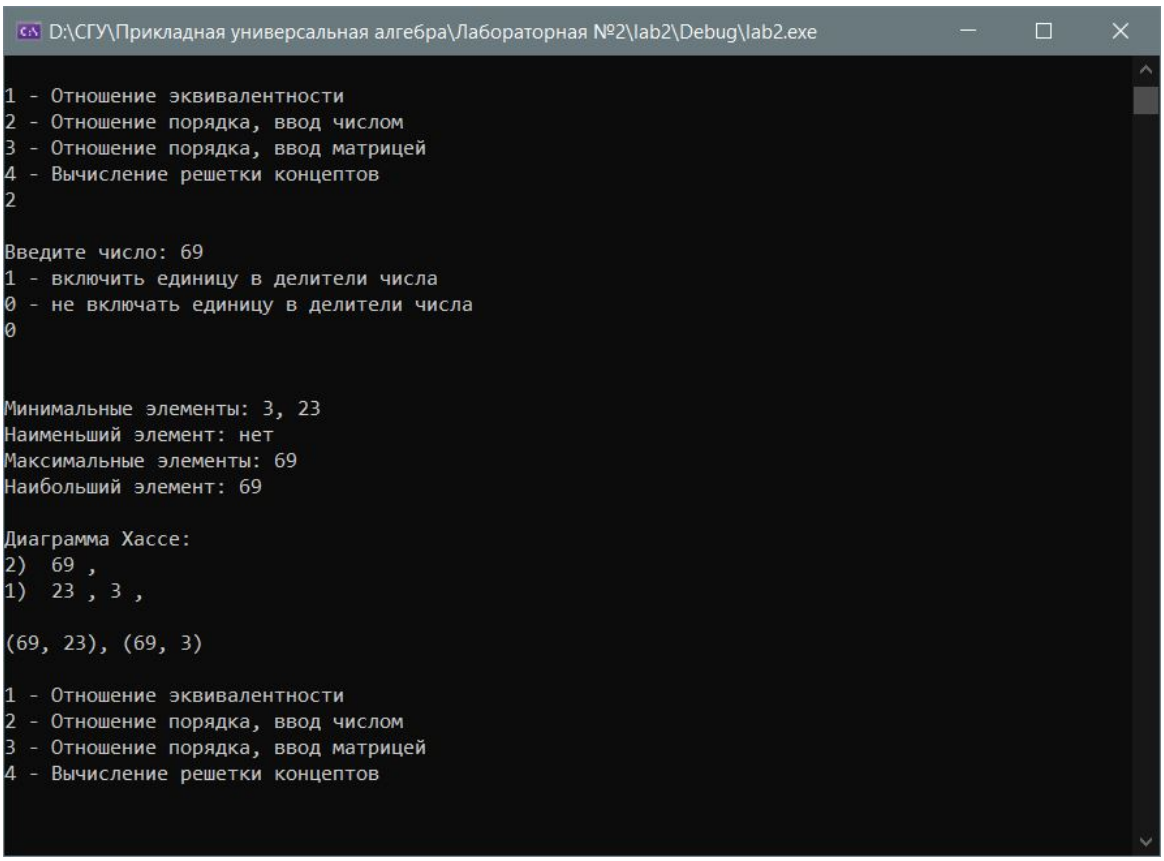

Рисунок 3 – Тест программы для отношения порядка числа(не включая 1)

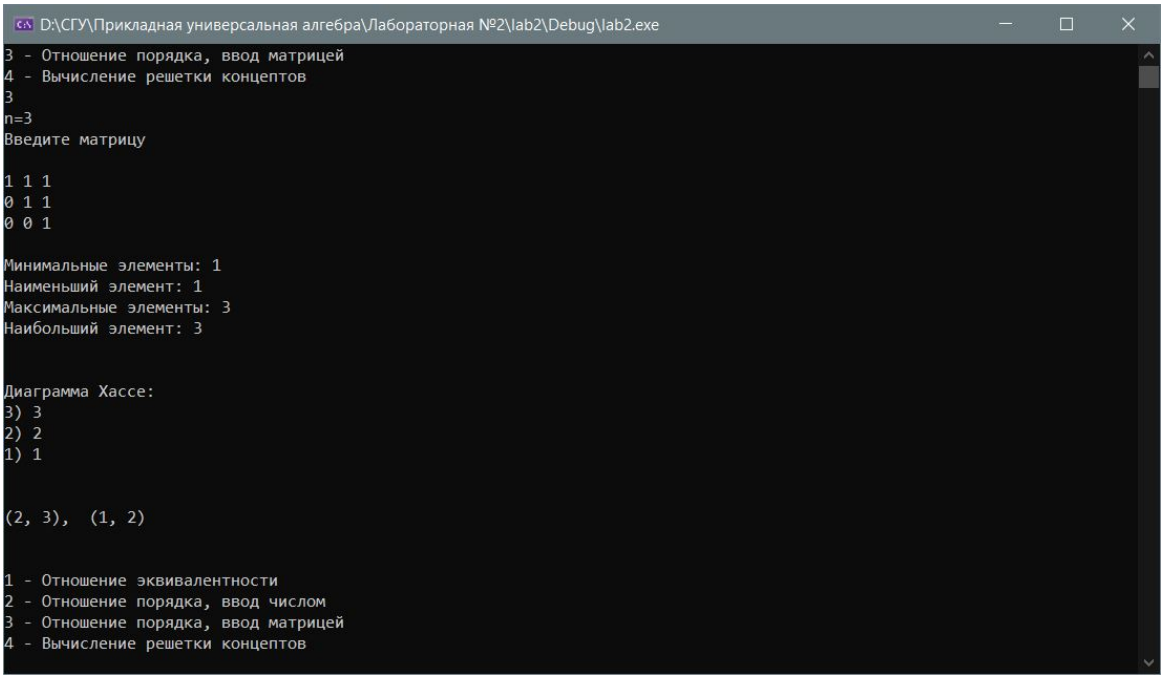

Рисунок 4 – Тест программы отношения порядка матрицы

3.2 Код программы, на основе рассмотренных алгоритмов, на языке C++

```
#include <iostream>
```

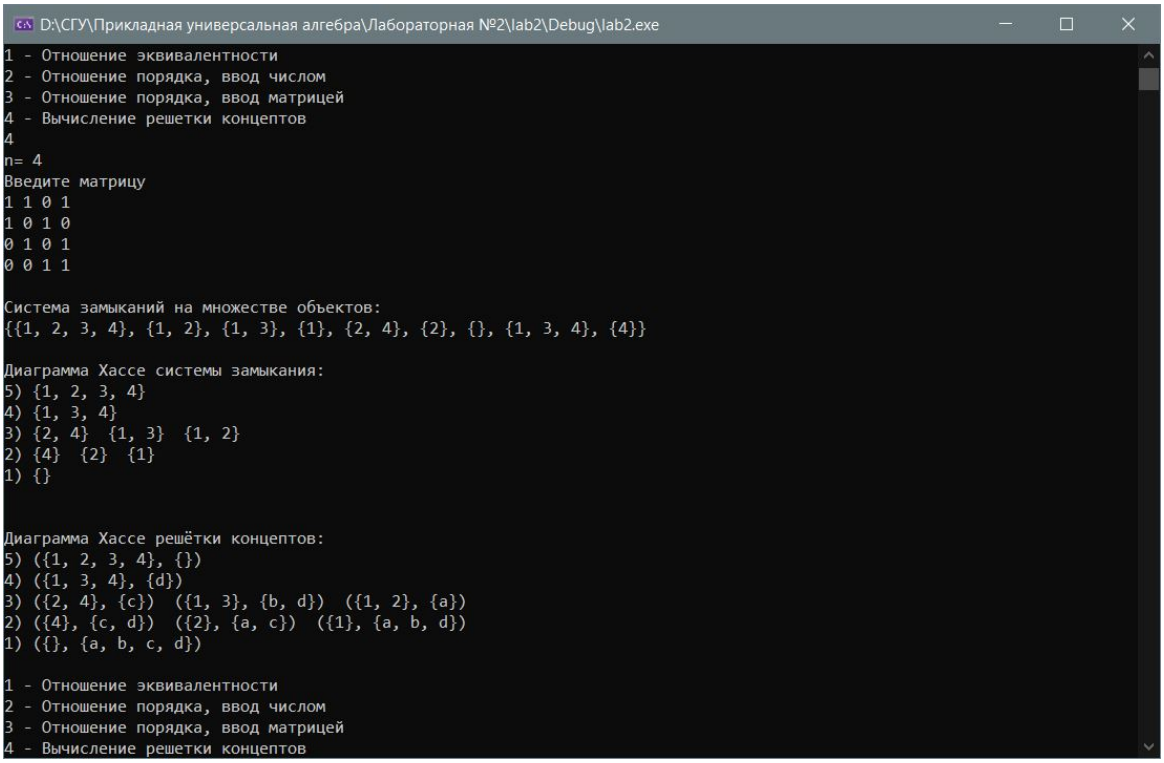

Рисунок 5 – Тест программы вычисления решетки концептов

```
#include <cstdlib>
#include <ctime>

using namespace std;

struct KeyPair {
        int publicKey;
        int privateKey;
};

bool isPrime(int n) {
        if (n <= 1) return false;
        if (n <= 3) return true;
        if (n % 2 == 0 || n % 3 == 0) return false;
        for (int i = 5; i * i <= n; i += 6) {
                if (n % i == 0 || n % (i + 2) == 0)
                return false;
        }
        return true;
```

```
}

int modPow(int x, int y, int p) {
        int result = 1;
        x = x % p;
        while (y > 0) {
                if (y % 2 == 1) {
                        result = (result * x) % p;
                }
                y = y / 2;
                x = (x * x) % p;
        }
        return result;
}

int keyExchange(int privateKey, int publicKey, int prime) {
        return modPow(publicKey, privateKey, prime);
}

bool isPrimitiveRoot(int g, int p) {
        int phi = p - 1;
        for (int i = 2; i <= phi; i++) {
                int result = 1;
                for (int j = 0; j < i; j++) {
                        result = (result * g) % p;
                }
                if (result == 1) {
                        return false;
                }
        }

        return true;
}
```

```
int generateRandomPrime(int min, int max) {
        int p;
        do {
                p = rand() % (max - min + 1) + min;
        } while (!isPrime(p));
        return p;
}

int main() {
        setlocale(LC_ALL, "Rus");

        srand(time(nullptr));

        int sharedPrime = generateRandomPrime(100, 1000);

        //int primitiveRoot = 2
        //while (!isPrimitiveRoot(primitiveRoot,
        sharedPrime)) {
                //     primitiveRoot++;
                //}

        KeyPair aliceKey;
        KeyPair bobKey;

        aliceKey.privateKey = rand() %
        (sharedPrime - 2) + 1;
        bobKey.privateKey = rand() %
        (sharedPrime - 2) + 1;

        aliceKey.publicKey = modPow(2, aliceKey.privateKey,
        sharedPrime);
        bobKey.publicKey = modPow(2, bobKey.privateKey,
        sharedPrime);
```

```
    int sharedSecretAlice =
    keyExchange(aliceKey.privateKey,
    bobKey.publicKey, sharedPrime);
    int sharedSecretBob =
    keyExchange(bobKey.privateKey,
    aliceKey.publicKey, sharedPrime);

    cout << "p: " << sharedPrime << endl;
    cout << "Alice's private key: " <<
    aliceKey.privateKey << endl;
    cout << "Bob's private key: " <<
    bobKey.privateKey << endl;
    cout << "Alice's public key: " <<
    aliceKey.publicKey << endl;
    cout << "Bob's public key: " <<
    bobKey.publicKey << endl;
    cout << "k (Alice): " <<
    sharedSecretAlice << endl;
    cout << "k' (Bob): " <<
    sharedSecretBob << endl;

    return 0;
}
```

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы были рассмотрены теоретические сведения о протоколах открытого распределения ключей и подробно рассмотрен алгоритм Хьюза. Также, на основе рассмотренных материалов, была реализована программа.