**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЁТ**

**по лабораторной работе №4**

**«Алгоритм обмена ключами Диффи–Хеллмана. Обмен ключами между двумя сторонами»**

*дисциплина: Математические основы защиты информации и информационной безопасности*

Вариант 1

Студент: Хиссен Али Уэддей

Группы: НПМмд-02-20

Ст/б: 10322009306

**МОСКВА**

**2021**

Оглавление

[Цель работы 3](#_Toc59279780)

[Описание алгоритма 3](#_Toc59279781)

[Реализация алгоритма (код программы) 6](#_Toc59279782)

[Результат работы программы 7](#_Toc59279783)

[Вывод 7](#_Toc59279784)

# **Цель работы**

Целью работы является реализация варианта криптографического алгоритма обмена ключами между сторонами.

Определила вариант с помощью формулы , где — номер студ.билета, N— количество заданий. Так как Sn=1032209306, а N=2, то номер варианта лабораторной работы 1.

# **Описание алгоритма**

Протокол Диффи — Хеллмана (англ. Diffie–Hellman, DH) — криптографический протокол, позволяющий двум и более сторонам получить общий секретный ключ, используя незащищенный от прослушивания канал связи. Полученный ключ используется для шифрования дальнейшего обмена с помощью алгоритмов симметричного шифрования. Схема открытого распределения ключей, предложенная Диффи и Хеллманом, произвела настоящую революцию в мире шифрования, так как снимала основную проблему классической криптографии — проблему распределения ключей.

В чистом виде алгоритм Диффи — Хеллмана уязвим для модификации данных в канале связи, в том числе для атаки «Man-in-the-middle (человек посередине)», поэтому схемы с его использованием применяют дополнительные методы односторонней или двусторонней аутентификации.

В 1976 г. Диффи и Хеллман опубликовали статью, которая ознаменовала собой рождении асимметричной криптографии и привела к сильному росту числа открытых исследований в области криптографии. Она содержала ошеломляющий результат: возможно построение практически стойких секретных систем, которые не требуют передачи секретного ключа. Диффи и Хеллман ввели понятие односторонней функции с потайным ходом. Под односторонней функцией f понимается функция f(x), которая легко вычислима для любого значения аргумента x из области определения, однако для данного y из области ее значений вычислительно сложно нахождение значения аргумента x, для которого f(x) = y. Применение таких функций для защиты входа в вычислительную систему путем одностороннего преобразования паролей было известно. Но как применить одностороннюю функцию в криптографических системах, когда даже законный получатель не сможет выполнить дешифрования? Для шифрования была предложена односторонняя функция с потайным ходом (секретом).

Под односторонней функцией с потайным ходом понимается семейство обратимых функций с параметром z, таких, что для данного z можно найти алгоритмы и , позволяющие легко вычислить значение для всех x из области определения, а также вычислить значение для всех y из области значений, однако практически для всех значений параметра z и практически для всех значений y из области значений нахождение вычислительно неосуществимо даже при известном .

В качестве односторонней функции Диффи и Хеллман предложили функцию дискретного возведения в степень

где – целое число, , k-битовое простое число. Причем выбирается такое число , степени которого по модулю n представляют собой упорядоченное множество чисел являющееся некоторой перестановкой чисел {1, 2, …, n-1}. (Такое число g называется первообразным корнем по модулю n.) Даже для очень больших модулей n (например, при k = 1024 бит) для данного x легко вычислить значение этой функции. Процедура вычисления этой функции называется дискретным возведением в степень. Для выполнения этой процедуры достаточно выполнение около операций умножения k-битовых чисел (или умножений и делений 2k-битовых чисел на k-битовые). Процедура дискретного возведения в степень основана на предварительном вычислений значений (по модулю n).

При работе алгоритма каждая сторона:

1. Генерирует случайное [натуральное число](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D1%82%D1%83%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE) a — закрытый ключ
2. Совместно с удалённой стороной устанавливает открытые параметры p и g (обычно значения p и g генерируются на одной стороне и передаются другой), где p является случайным простым числом также должно быть случайным простым числом (для повышения безопасности), g является первообразным корнем по модулю *p* (также простое число).
3. Вычисляет открытый ключ А, используя преобразования над закрытым ключом
4. Обмениваются открытыми ключами с удаленной стороной
5. Вычисляет общий секретный ключ *K,* используя открытый ключ удаленной стороны *B* и свой закрытый ключ *a*

*K* получается равным с обеих сторон, так как

В практических реализациях для a и b используются числа порядка 10100 и p порядка 10300. Число g не обязано быть большим и обычно имеет значение в пределах первого десятка.

Сам же обмен ключами между пользователями позволяет использовать криптографические алгоритмы. Если отправитель и получатель желают обменяться зашифрованными сообщениями, каждый должен иметь соответствующее программное обеспечение для шифрования отсылаемых сообщений и расшифровки полученных.

Особенности необходимого программного обеспечения зависят от используемой технологии шифрования. Если они используют коды, оба должны иметь копии одной книги кодов. Если используется шифр, требуются подходящие ключи. Если шифр симметричный, обоим требуются копии одного и того же ключа. Если асимметричный, им требуются открытые ключи друг друга.

Проблема обмена ключами заключается в том, что обмен ключами или другой информацией следует проводить так, чтобы никто другой не мог получить копию. Как правило, это требует доверенного посыльного или иного безопасного канала.

# **Реализация алгоритма (код программы)**

g и p - открытое простое число.

a - секретный ключ Алисы.

b - секретный ключ Боба.

Их вычисления общего закрытого ключа (s) должны совпадать

import random

a = random.randint(1,50)

b = random.randint(1,50)

p=37

g=59

alicekey = (g\*\*a)%p

bobkey = (g\*\*b)%p

print ("Alice secret key: ", alicekey)

print ("Bob secret key: ", bobkey)

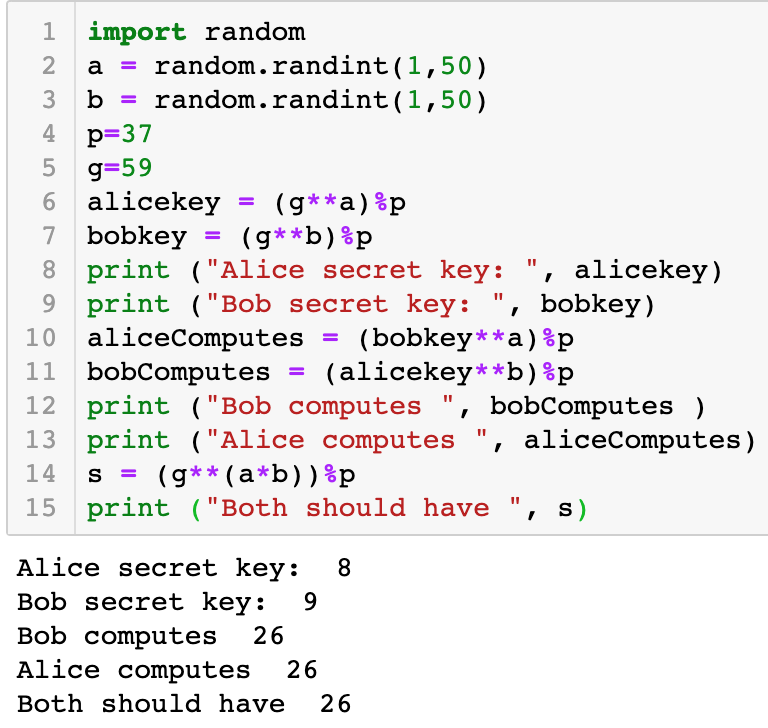
aliceComputes = (bobkey\*\*a)%p

bobComputes = (alicekey\*\*b)%p

print ("Bob computes ", bobComputes) print ("Alice computes ", aliceComputes) s = (g\*\*(a\*b))%p

print ("Both should have ", s)

# **Результат работы программы**



# **Вывод**

В ходе выполнения лабораторной работы была выполнена реализация варианта криптографического алгоритма обмена ключами между сторонами