Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**Отчёт**

**по лабораторной работе №13**

**Дисциплина: КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ**

**Тема: «симметричные Алгоритмы шифрования. Режимы работы алгоритмов»**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ К. А. Корнилов

Направление подготовки 02.03.02 Фундаментальная информатика и

информационные технологии

Направленность (профиль) Математическое и программное обеспечение

компьютерных технологий

Преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А. А. Крамаренко

**Тема:** Симметричные алгоритмы шифрования. Режимы работы алгоритмов

**Цель:** Изучение симметричных алгоритмов шифрования и их режимов работы

**Ход работы:**

1. Реализовать шифрование и расшифрование DES в режиме CBC, для реализации DES можно пользоваться методами библиотек.

Для реализации поставленной задачи была написана программа, реализующая работу алгоритма DES в режиме CBC.

Для реализации работы алгоритма DES был написан отдельный класс с необходимыми методами и полями.

Внутри класса в качестве констант содержатся перестановка для сжатия ключа, перестановка для удаления проверочных битов в ключе, перестановки на входе и выходе из алгоритма, перестановка для расширения при выполнении очередного раунда, перестановка для s-блоков сжатия, перестановка для вычисления функции f при очередном раунде.

При инициализации экземпляра класса происходит генерация ключей каждого раунда. Для этого ключ делится пополам. Затем на каждой итерации генерации ключа раунда происходит сдвиг ключа влево на нужное число битов. После этого происходит сжатие полученных половинок и занесение нового ключа в список ключей раунда.

При шифровании сообщения вызывается соответствующий метод, который запускает цикл на количество итераций равное количество раундов алгоритма. На каждой итерации вызывается функция вычисления итерации с аргументами номером раунда, входящим сообщением. На каждой итерации сообщение делится на 2 половины по 32 бита. Затем от правой половины вычисляется функция F. Для этого сначала применяется перестановка расширения, затем вычисляется xor с ключом раунда и применяются 8 подстановок s-блоков. После вычисления F происходит xor между левой половиной входного сообщения и результатом вычисления F и возврат результата с первой половиной в виде изначальной правой части и второй половиной в виде вычисления последнего xor. После завершения всех раундов происходит применение подстановки выхода. Аналогичным образом работает дешифровка, но с раундами, идущими в обратном порядке.

Для реализации работы алгоритма в режиме CBC был написан отдельный класс. В данном классе сначала происходит шифрование сообщения в кодировки utf-8, затем оно расширяется до размера кратного 64 битам и делится на части по 8 байт. Для каждой части отдельно запускается блок шифрования. Данный блок сначала берет результат шифрования с прошлого блока (или рандомный вектор в 64 бита, если блок первый), проводит xor между ним и текущим сообщением, а затем зашифровывает DES полученный результат. Аналогичным образом работает и дешифровка, но только вместо вызова шифрования DES происходит вызов функции расшифровки.

1. Реализовать шифрование и расшифрование AES в режиме CTR, для реализации AES можно пользоваться методами библиотек.

Для реализации поставленной задачи была написана программа, реализующая работу алгоритма AES.

Алгоритм AES работает с блоками данных размером 128 бит и поддерживает ключи длиной 128, 192 или 256 бит. В зависимости от длины ключа, количество раундов шифрования варьируется: 10 раундов для 128-битного ключа, 12 раундов для 192-битного и 14 раундов для 256-битного ключа.

Для реализации работы алгоритма AES был написан отдельный класс, содержащий необходимые методы и поля. Внутри класса в качестве констант содержатся таблицы подстановки (S-box и обратный S-box), константы для генерации раундовых ключей (r\_con), а также матрицы для операций MixColumns и InvMixColumns.

При инициализации экземпляра класса происходит генерация раундовых ключей. Для этого исходный ключ расширяется с использованием алгоритма расширения ключа. Ключ разбивается на слова (по 32 бита), и на каждой итерации генерируется новое слово с использованием предыдущих слов, таблицы подстановки (S-box) и констант r\_con. Полученные раундовые ключи сохраняются для использования в процессе шифрования и расшифрования.

При шифровании сообщения вызывается соответствующий метод, который выполняет следующие шаги. Входные данные разбиваются на блоки по 128 бит и представляются в виде матрицы 4x4, где каждый элемент — это один байт. Выполняется начальный раунд, который состоит из операции AddRoundKey — побитового сложения (XOR) матрицы состояния с первым раундовым ключом.

Затем выполняются основные раунды. Каждый раунд (кроме последнего) состоит из четырех операций.

* + SubBytes: Замена каждого байта в матрице состояния с использованием таблицы подстановки (S-box).
  + ShiftRows: Циклический сдвиг строк матрицы состояния. Первая строка не сдвигается, вторая сдвигается на 1 байт влево, третья — на 2 байта, четвертая — на 3 байта.
  + MixColumns: Линейное преобразование столбцов матрицы состояния. Каждый столбец умножается на фиксированную матрицу в поле Галуа GF(28).
  + AddRoundKey: Побитовое сложение (XOR) матрицы состояния с раундовым ключом.

В финальном раунде операция MixColumns не выполняется. Раунд состоит из SubBytes, ShiftRows и AddRoundKey.

После завершения всех раундов матрица состояния преобразуется в зашифрованный блок данных.

Для расшифрования сообщения вызывается соответствующий метод, который выполняет обратные операции в обратном порядке.

Зашифрованные данные разбиваются на блоки по 128 бит и представляются в виде матрицы 4x4. Выполняется начальный раунд, который состоит из операции AddRoundKey с последним раундовым ключом.

Затем выполняются основные раунды. Каждый раунд (кроме первого) состоит из четырех операций.

* + InvShiftRows: Обратный сдвиг строк матрицы состояния.
  + InvSubBytes: Обратная замена байтов с использованием обратной таблицы подстановки (Inv S-box).
  + AddRoundKey: Побитовое сложение (XOR) матрицы состояния с раундовым ключом.
  + InvMixColumns: Обратное преобразование столбцов матрицы состояния.

В финальном раунде операция InvMixColumns не выполняется. Раунд состоит из InvShiftRows, InvSubBytes и AddRoundKey.

После завершения всех раундов матрица состояния преобразуется в исходный блок данных.

Для реализации поставленной задачи был написан класс CTR, который реализует режим шифрования CTR.

В конструкторе класса инициализируется объект алгоритма шифрования (по умолчанию используется DES). Также генерируется начальное значение счетчика с использованием криптографически безопасного генератора случайных чисел.

Метод шифрования принимает на вход строку, которая кодируется в байты с использованием кодировки UTF-8. Данные дополняются до размера, кратного длине блока алгоритма.

Для каждого блока данных выполняется операция шифрования в режиме CTR. Текущее значение счетчика шифруется с использованием алгоритма. Результат шифрования складывается по модулю 2 с текущим блоком данных. Затем счетчик увеличивается на 1. После обработки всех блоков зашифрованные данные возвращаются в виде бинарной строки.

Метод расшифровки принимает на вход зашифрованную бинарную строку. Для каждого блока данных выполняется операция расшифрования в режиме CTR. Текущее значение счетчика шифруется с использованием алгоритма. Результат шифрования складывается по модулю 2 с текущим блоком зашифрованных данных. Счетчик увеличивается на 1 с помощью метода. После обработки всех блоков данные преобразуются в байты, и удаляется дополнение. Расшифрованные данные декодируются из байтов в строку с использованием кодировки UTF-8 и возвращаются.

**Вывод:** Были изучены симметричные алгоритмы шифрования и режимы их работы