Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ «Информатика и системы управления»

КАФЕДРА ИУ-7 «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

по дисциплине «Защита информации»

«Алгоритм шифрования AES»

Студент группы **ИУ7-76Б** _____ **В. М. Мансуров**
(Подпись, дата) (И.О. Фамилия)

Руководитель _____ **И. С. Чиж**
(Подпись, дата) (И.О. Фамилия)

2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Шифрование информации — занятие, которым человек занимался ещё до начала первого тысячелетия, занятие, позволяющее защитить информацию от посторонних лиц.

Шифровальная алгоритм AES — алгоритм, разработанный в 2001 году Национальным универитетом стандатов и технологий США и пришедший на смену алгоритму AES.

Целью данной работы является реализация в виде программы на языке программирования C или C++ шифровального алгоритма AES в режиме работы CFB — режима параллельного сцепления блоков шифра.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1) изучить шифроовальный алгоритм AES и его режим работы CFB;
2) реализовать шифровальный алгоритм AES в виде программы, обеспечив возможности шифрования и расшифровки файла в режиме работы CFB;
3) протестировать разработанную программу, показать, что удаётся дешфировать все файлы;
4) описать и обосновать полученные результаты в отчёте о выполненной лабораторной работе.

1 Аналитическая часть

В этом разделе будут рассмотрен шифровальный алгоритм AES в режиме шифрования CFB.

1.1 Алгоритм AES

Шифровальная алгоритм AES (англ. *Advanced Encryption Standart* — AES) — симметричный блочнный шифровальный алгоритм, разработанный в 2001 году Национальный институтом стандартов и технологий США. Он использует блочное шифрование, длина блока фиксирована и равна 128 битам, длина ключа 128, 192 либо же 256 бит. Он состоит раундов шифрования,количество которых зависит от длины ключа: 10 раундов для ключа размером 128 бит, 12 раундов для ключа размером 192 бита и 14 раундов для ключа размером 256 бит.

Прежде чем перейти к раундам шифрования, происходит генерация ключей раунда (раундовых ключей) из исходного ключа, Рассмотрим, как это происходит.

1.1.1 Получение ключей раунда

Определим фунцию g, изменяющую четырёхбайтовое слово так, как указано на рисунке 1.1.

Ключей раундов k_i необходимо на 1 больше, чем количество раундов, т.е. 11 ключей раундов для основногоключа длиной 128 бит, 13 ключей раунда для основного ключа длиной 192 бита и 15 ключей раунда для основного ключа длинной 256 бит.

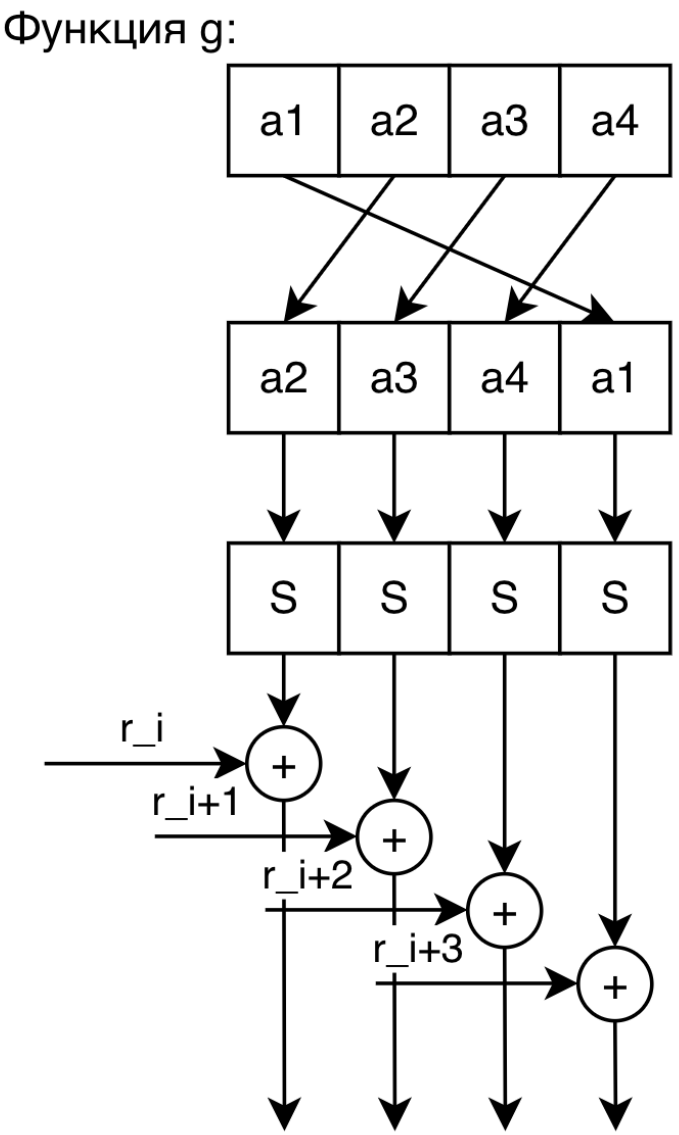

Рисунок 1.1 – Схема функции g

Алгоритм получения ключа раунда из исходного ключа преставлен в виде схемы алгоритма на рисунке 1.2.

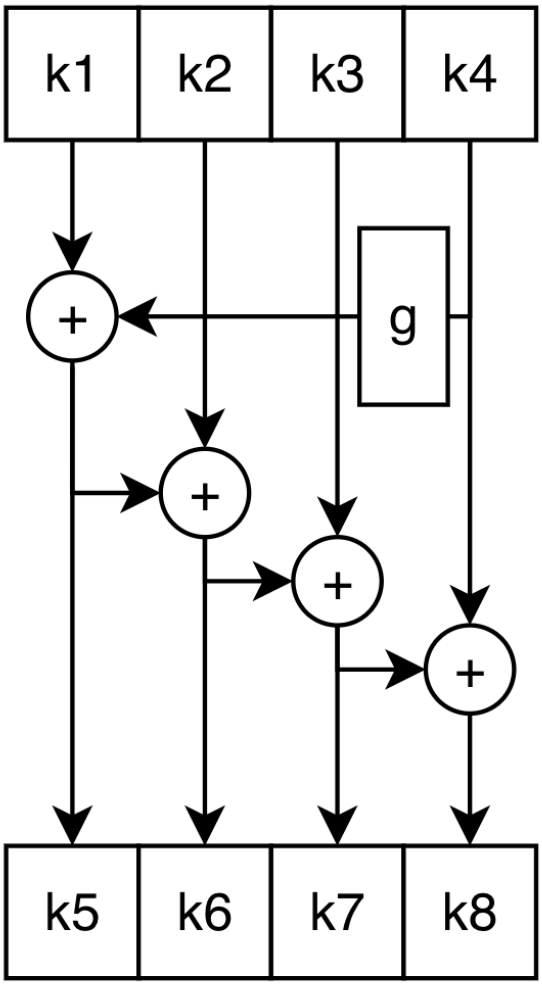

Рисунок 1.2 – Схема функции g

1.1.2 Раунд шифрования

Раунд шифрования состоит из 5 следующих этапов

1) замена (англ. *confussion*);
2) процедура перестановки строк (англ. *row-row mix procedure* — RR);
3) процедура перестановки столбцов (англ. *row-columns mix* — RC);
4) смешивание ключа (англ. *key mixing* — KM).

Замена обеспечивает нелинейность алгоритма шифрования, обрабатываая каждый байт состояния, производя нелинейную замену байт с использованием таблицы замен.

Процедура перестановки строк представляет из себя циклический сдвиг строки ссостояний на количество байт, зависящее от номера строки.

Процедура перестановки столбцов 4 байта каждого столбца смешиваются с использовоанием обратимой линейной трансформации. На последнем раунду эта процедура не выполняется.

Смешивание ключа представляет из себя операцию XOR с ключом раунда, полученным заранее.

1.2 Режимы работы алгоритма AES

Режим шифрования — метод применения блочного шифра, позволяющий преобразовать последовательность блоков открытых данных в последовательность блоков зашифрованных данных.

Для AES рекомендованы следующие режими работы:

1) режим электронной кодовой книги (англ. *Electronic Code Bloc* — ECB);
2) режим сцепления блоков (англ. *Cipher Block Chaining* – CBC);
3) режим параллельнооого сцепления блоков (англ. *Parallel Cipher Block Chaining* – PCBC);
4) режим обратной связи по шифротексту (англ. *Cipher Feed Back* — CFB);
5) режим обратной связи по выходу (англ. *Output Feed Back* — OFB).

В данной работе будет CFB.

1.3 Режимы работы алгоритма CFB

Алгоритм CFB схематично представлен на рисунке 1.3. Суть алгоритма заключается в том, что изначально берется блок из 128 битов $_0$, который называется синхропосылкой. Вектор инициализации (или результаты прошлого XOR) шифруется алгоритмом AES или DES (в нашем случае AES). Затем результат суммируется по модулю 2 с блоком $_0$ и при этом результат

используется для шифрования следующего блока. Таким образом каждый блок суммируется с результатом шифрования предыдущего блока.

Особенностью данного режима является распространение ошибки на всесь последующий текст. Применяется как правило для шифрования потков информации видео и аудио.

Рисунок 1.3 – Обобщенная схема алгоритма режима шифрования CFB

2 Конструкторская часть

В этом разделе будут представлены описания модулей программы, а также схема алгоритма шифроваания AES.

2.1 Разработка алгоритмов

На рисунках 2.1 представлены схемы алгоритма AES, раунда AES.

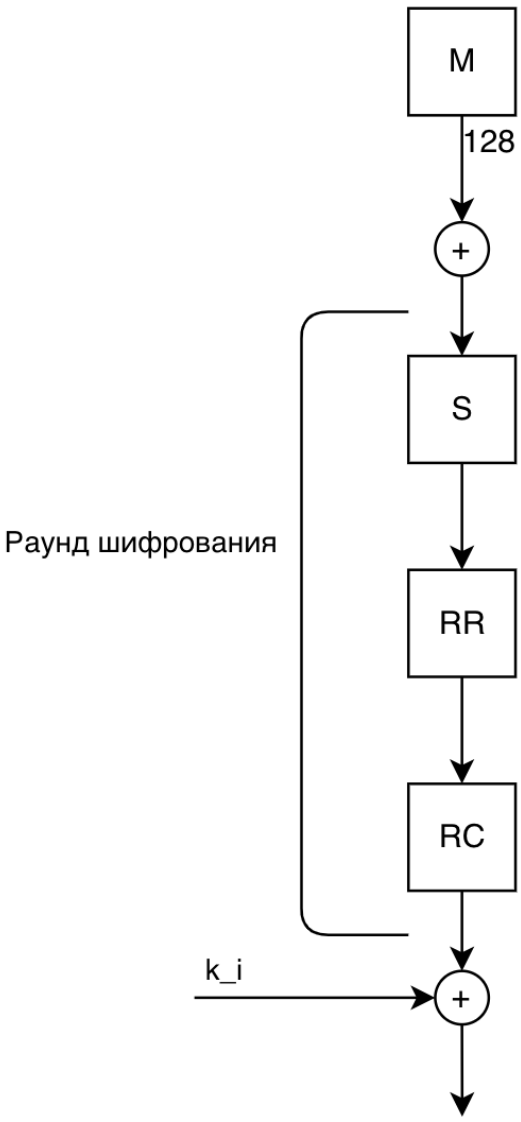

Рисунок 2.1 – Схема шифровального алгоритма AES

3 Технологическая часть

3.1 Средства реализации

Для программной реализации шифровальной машины был выбран язык C++ [2]. В данном языке есть все требующиеся инструменты для данной лабораторной работы. В качестве среды разработки была выбрана среда CLion [3].

3.2 Реализация алгоритма

Листинг 3.1 – Класс реализации режима CFB

```
class cfb {
    public:
    explicit cfb (const vector<uint8_t>& vi);

    string crypt(string message, bool decrypt = false);

    vector<block> divideBlocks(const vector<uint8_t> &message);
    vector<uint8_t> mergeBlocks(const vector<block> &blocks);

    vector<uint8_t> encrypt(vector<uint8_t>& block128, const
       vector<uint8_t>& key);
    vector<uint8_t> decrypt(vector<uint8_t>& block128, const
       vector<uint8_t>& key);

    void setVI(const vector<uint8_t>& vi);

    void print_bloks(const vector<block> &blocks);
    void print_message(const vector<uint8_t> message);
};
```

Листинг 3.2 – Класс шифрования и дешифрования AES

```
1  class aes: public IEncoder{
2      public:
3      block CryptBlock(const block& block128, const key& key128,
           bool decrypter = false);
4      block EncryptBlock(const block& block128, const key& key128)
           override;
5      block DecryptBlock(const block& block128, const key& key128)
           override;
6
7      vector<key> GetKeys128(key key128, bool decrypter = false);
8      void AddRoundKey(mtx& block, const key& roundKey);
9      void SubBytes(mtx& block);
10     void InvSubBytes(mtx& block);
11     void ShiftRows(mtx& block);
12     void InvShiftRows(mtx& block);
13     void MixColumns(mtx& block);
14     void InvMixColumns(mtx& block);
15
16     mtx ArrayToMrx4x4(const block& block);
17     block Mrx4x4ToArray(const mtx &mtx);
18     void AddPadding(vector<uint8_t>& data);
19     void RemovePadding(vector<uint8_t>& data);
20     private:
21     // Уожение поля Галуа
22     uint8_t GMul(uint8_t x, uint8_t y);
23
24     AES_MODE mode = AES128;
25 }
```

Листинг 3.3 – Реализация алгоритма AES

```
block aes::EncryptBlock(const block &block128, const key &key128)
    {
     // Расширение ключа – KeyExpansion
     auto keys = GetKeys128(key128, false);

     // для удобства конвертируем в матрицу
     auto state = ArrayToMrx4x4(block128);

     // Начальный раунд – сложение с основным ключом;
     AddRoundKey(state, keys[0]);

     // 9 раундов шифрования
     for (int i = 1; i < 10; i++)
     {
          SubBytes(state);
          ShiftRows(state);
          MixColumns(state);
          AddRoundKey(state, keys[i]);
     }

     // Финальный раунд
     SubBytes(state);
     ShiftRows(state);
     AddRoundKey(state, keys[10]);

     return Mrx4x4ToArray(state);
}
```

3.3 Тестирование

Тестирование разработанной программы производилось следующим образом: выбирались случайные значения ключа и вектора IV, а также получа-

лась случайная последовательность блоков для шифрования длиной n. Она зашифровывалась и расшифровывалась, проверялось совпадение полученного результата с начальными данными. Данная процедура повторялась n раз для значений n от 1 до 100.

Таблица 3.1 – Функциональные тесты

Длина, байты	Шифруемое значение	Результат работы
8	12345678	ЃЯЄ70ЌЗ
16	1234567812345678	ЃЯЄ70ЌЗРх ¬6@,
32	1234567812345678 1234567812345678	ЃЯЄ70ЌЗРх ¬6@, vќ F°qђЎsђGZд/$

Вывод

В данном разделе были рассмотрены средства реализации, а также представлены листинги реализации шифровального алгоритма AES и режима работы CFB, произведено тестирование.

Заключение

В результате лабораторной работы был реализован в виде программы шифровальный алгоритма AES в режиме работы PCBC

Был и выполнены следующие задачи:

1) изучен шифроовальный алгоритм AES и его режим работы CFB;
2) реализован шифровальный алгоритм AES в виде программы, обеспечена возможность шифрования и расшифровки файла в режиме работы CFB ;
3) протестирована разработанная программа;
4) описаны и обоснованы полученные результаты в отчёте о выполненной лабораторной работе.

Список использованных источников

1. И.М. Шолин. Алгоритм переносной шифровальной машины энигма. — Кубанский государственный технологический университет.

2. Язык программирования C++. `https://learn.microsoft.com/en-us/cpp/cpp/cpp-language-reference?view=msvc-170`. дата обращения: 15.10.2023.

3. CLion. `jetbrains.com`. дата обращения: 15.10.2023.