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 «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по лабораторной работа №3

по курсу «Защита информации»

на тему: «Алгоритм AES»

Вариант № 3

Студент ИУ7-73Б (Группа) _____ (Подпись, дата) Лысцев Н. Д. (И. О. Фамилия)

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

2024 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью данной лабораторной работы является реализация программы шифрования симметричным алгоритмом AES с применением режима CFB.

Для достижения поставленной цели необходимо решить следующие задачи:

1) описать алгоритм AES с режимом CFB;

2) выбрать средства программной реализации;

3) реализовать данный алгоритм.

1 Аналитический раздел

В этом разделе будет 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.

Функция g:

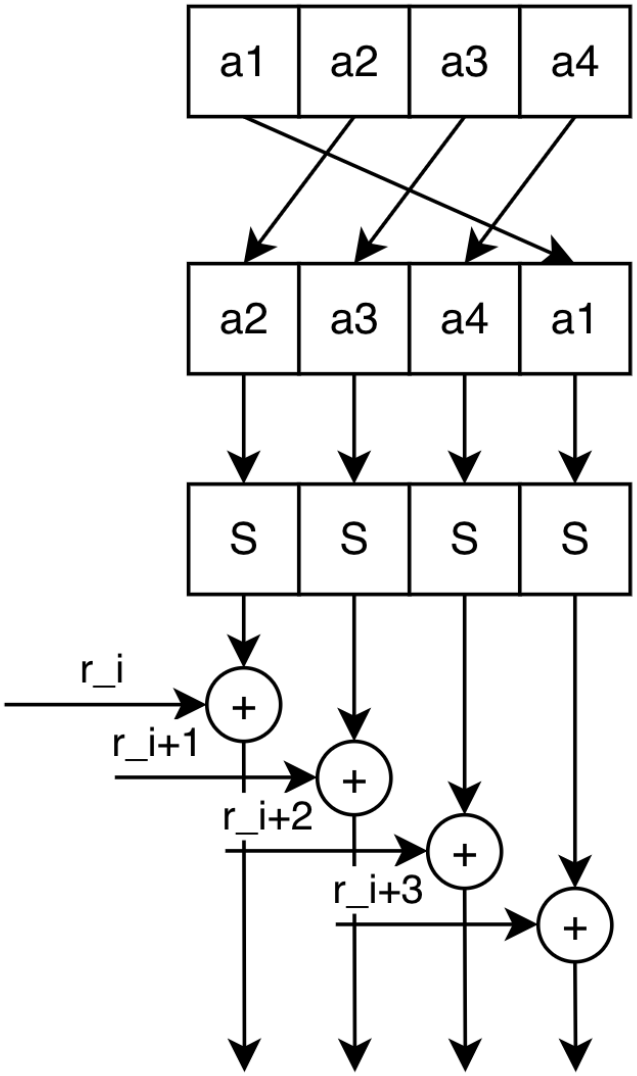

Рисунок 1.1 – Схема функции g

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

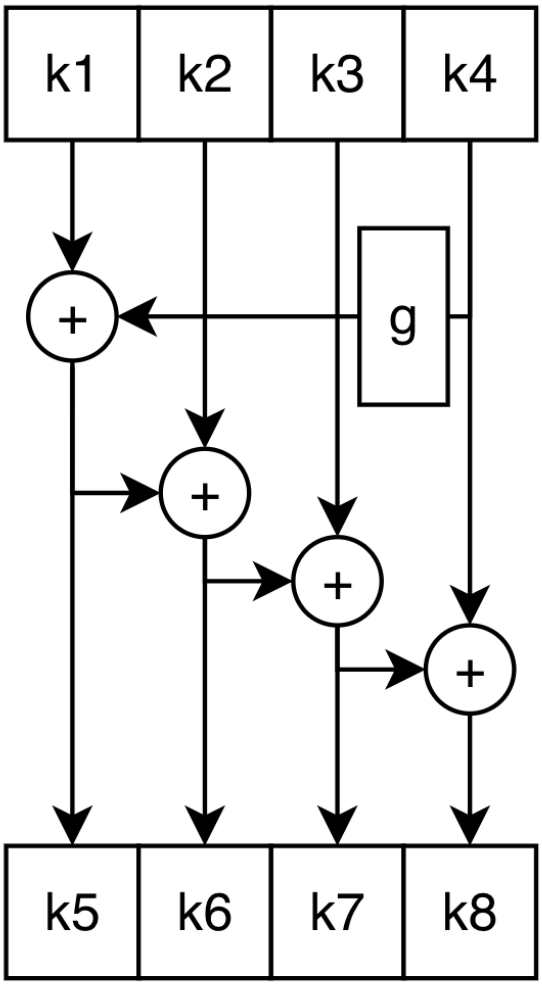

Рисунок 1.2 – Схема алгоритма получения ключей раунда

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, обрабатывая каждый байт состояния, производя нелинейную замену байт с использованием таблицы замен.

Процедура перестановки строк представляет из себя циклический сдвиг строки состояний 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 параллельноого 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).

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 C_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 C_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

Вывод

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

2 Конструкторский раздел

На рисунках 2.1 представлены схемы алгоритма AES, раунда AES.

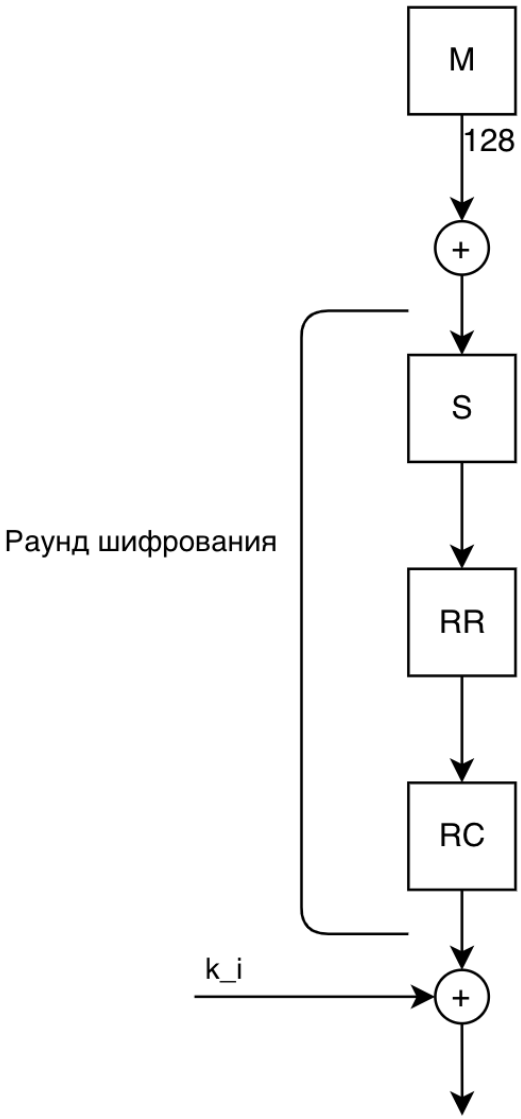

Рисунок 2.1 – Схема шифровального алгоритма AES

Вывод

В этом разделе была представлена схема алгоритма шифрования AES.

3 Технологический раздел

В данном разделе будут перечислены требования к программному обеспечению, средства реализации и листинги кода.

3.1 Требования к программному обеспечению

К программе предъявляется ряд требований:

– программа должна предоставлять возможность шифрования и расшифровки произвольного файла, а также текстового сообщения;

– программа должна корректно работать с пустым однобайтовым файлом;

– программа должна уметь обрабатывать файл архива.

3.2 Средства реализации

В качестве языка программирования для этой лабораторной работы был выбран $C++$ [1].

3.3 Сведения о модулях программы

Программа состоит из трех модулей:

– `main.cpp` — файл, содержащий точку входа в программу;

– `AES.cpp` — модуль, реализующий алгоритм AES;

– `CFB.cpp` — модуль, реализующий шифрование в режиме CFB.

3.4 Тестирование

Таблица 3.1 – Функциональные тесты

Входная строка	**Шифрованная строка**
Hello	gùÄv
gùÄv	Hello
B	hí'(Ñ!
hí'(Ñ!	B

Все тесты были успешно пройдены. Тесты с файлами были также успешно пройдены.

3.5 Реализации алгоритмов

На листинге 3.1 представлена функция, реализующая алгоритм DES.

На листинге 3.2 представлена функция, реализующая генерацию ключей для каждого раунда шифрования.

На листинге 3.3 представлена функция, реализующая добавление ключа раунда к state.

На листинге 3.4 представлена функция, реализующая нелинейную замену байт в state с помощью sbox.

На листинге 3.5 представлена функция, реализующая циклический сдвиг строк state влево в зависимости от номера строк.

На листинге 3.6 представлена функция, реализующая процедуру MixColumns.

Листинг 3.1 – Функция, реализующая алгоритм AES

```
vector<uint8_t> AES::encryptBlock(const vector<uint8_t> &input,
   const vector<uint8_t> &key) {
    vector<vector<vector<uint8_t>>> roundKeys =
       this->_keyExpansion(key);

    AES::_addRoundKey(state, roundKeys[0]);

    for (int i = 1; i < this->_nr; i++) {
        this->_subBytes(state);
        AES::_shiftRows(state);
        AES::_mixColumns(state);
        AES::_addRoundKey(state, roundKeys[i]);
    }

    this->_subBytes(state);
    AES::_shiftRows(state);
    AES::_addRoundKey(state, roundKeys[roundKeys.size() - 1]);

    return this->_stateToOutput(state);
}
```

Листинг 3.2 – Функция, реализующая генерацию ключей раундов

```
vector<vector<vector<uint8_t>>> AES::_keyExpansion(const
   vector<uint8_t> &key) {
    vector<vector<uint8_t>> keyAsMtrWords =
       this->_keyToMtrWords(key);

    vector<vector<uint8_t>> resultWords(this->_nb * (this->_nr +
       1), vector<uint8_t>(this->_nb));
    for (int i = 0; i < keyAsMtrWords.size(); i++) {
        resultWords[i] = keyAsMtrWords[i];
    }

    for (int i = this->_nk; i < this->_nb * (this->_nr + 1);
       i++) {
        vector<uint8_t> tmpWord = resultWords[i - 1];

        if (i % this->_nk == 0) {
            auto tmp = this->_subWord(AES::_rotWord(tmpWord));
            tmpWord = AES::_xorWordAndRcon(tmp, this->_rcon[i /
               this->_nk]);
        } else if (this->_nk > 6 and i % this->_nk == 4) {
            tmpWord = this->_subWord(tmpWord);
        }

        resultWords[i] = AES::_xorTwoWords(resultWords[i -
           this->_nk], tmpWord);
    }

    return this->_convertVecWordsToVecRoundKeys(resultWords);
}
```

Листинг 3.3 – Функция, реализующая добавление ключа раунда к state

```
void AES::_addRoundKey(vector<vector<uint8_t>> &state, const
   vector<vector<uint8_t>> &roundKey) {
    for (int i = 0; i < state.size(); ++i) {
        for (int j = 0; j < state[0].size(); ++j) {
            state[i][j] ^= roundKey[i][j];
        }
    }
}
```

Листинг 3.4 – Функция, реализующая нелинейную замену байт в state с помощью sbox

```
void AES::_subBytes(vector<vector<uint8_t>> &state) {
    for (int i = 0; i < state.size(); ++i) {
        for (int j = 0; j < state[0].size(); ++j) {
            state[i][j] = this->_sbox[state[i][j]];
        }
    }
}
```

Листинг 3.5 – Функция, реализующая циклический сдвиг строк state влево в зависимости от номера строки

```
void AES::_shiftRows(vector<vector<uint8_t>> &state) {
    for (int i = 1; i < state.size(); ++i) {
        std::rotate(state[i].begin(), state[i].begin() + i,
           state[i].end());
    }
}
```

Листинг 3.6 – Функция, реализующая процедуру MixColumns

```
void AES::_mixColumns(vector<vector<uint8_t>> &state) {
    for (size_t i = 0; i < 4; ++i) {
        uint8_t s0 = state[0][i];
        uint8_t s1 = state[1][i];
        uint8_t s2 = state[2][i];
        uint8_t s3 = state[3][i];

        state[0][i] = AES::GMul(s0, 0x02) ^ AES::GMul(s1, 0x03)
           ^ s2 ^ s3;
        state[1][i] = s0 ^ AES::GMul(s1, 0x02) ^ AES::GMul(s2,
           0x03) ^ s3;
        state[2][i] = s0 ^ s1 ^ AES::GMul(s2, 0x02) ^
           AES::GMul(s3, 0x03);
        state[3][i] = AES::GMul(s0, 0x03) ^ s1 ^ s2 ^
           AES::GMul(s3, 0x02);
    }
}
```

Вывод

В данном разделе были перечислены требования к программному обеспечению, средства реализации и листинги кода.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Справочник по языку C++ [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/ru-ru/cpp/cpp/cpp-language-reference?view=msvc-170 (дата обращения: 28.09.2022).