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 работе №3

по курсу "Защита информации"

Тема Алгоритм работы AES

Студент Золотухин А. В.

Группа ИУ7-74Б

Преподаватели Чиж И.С.

Москва, 2023 г.

СОДЕРЖАНИЕ

Введение ... 3

1 Аналитический раздел ... 4

1.1 Алгоритм шифрования AES ... 4

2 Конструкторский раздел ... 6

2.1 Алгоритм шифрования AES ... 6

3 Технологический раздел ... 9

3.1 Средства реализации ... 9

3.2 Реализация алгоритмов ... 9

3.3 Тестирование реализации алгоритма ... 11

Заключение ... 13

Список использованных источников ... 14

ВВЕДЕНИЕ

Цель данной лабораторной работы — реализовать программу шифрования симметричным алгоритмом AES.

Информацию для разных целей пытались засекречивать с помощью шифрования на протяжении всей истории человечества. Шифр — это множество обратимых преобразований открытого текста, проводимых с целью его защиты от несанкционированного использования. Одним из таких шифров является AES.

В рамках выполнения лабораторной работы необходимо решить следующие задачи:

— описать алгоритм AES;

— реализовать алгоритм с режимом шифрования OFB.

1 Аналитический раздел

1.1 Алгоритм шифрования AES

AES, или Advanced Encryption Standard, - это алгоритм шифрования с симметричным ключом. Это одно из самых универсальных и наиболее любимых технических решений в сфере криптографии. В основе AES лежит блочный шифр, который использует 128-битный размер блока и 128, 192 или 256-битные ключи для шифрования данных. AES256 - это версия стандарта с 256-битными ключами. Этот стандарт широко считается самым безопасным стандартом цифровой криптографии, который обычно используется для наиболее надежной сквозной шифрованной связи. AES был разработан двумя бельгийскими криптографами, Джоаном Деменом и Винсентом Риджменом, и был принят в качестве официального стандарта в 2001 году Национальным институтом стандартов и технологий США. Такое достижение свидетельствует о широком признании, которое получил стандарт. Уже более 20 лет AES256 и шифрование AES в целом является одним из наиболее предпочтительных решений для разработчиков, желающих создать систему, в которой коммуникации хорошо защищены от постороннего или внешнего влияния и утечек. Вот основные шаги и логика работы AES:

1. Раунды (Rounds): Алгоритм AES использует различное количество раундов в зависимости от длины ключа. Он использует 10 раундов для 128-битного ключа, 12 раундов для 192-битного ключа и 14 раундов для 256-битного ключа. Чем больше раундов используется в процессе шифрования, тем надежнее шифрование. Каждый раунд включает в себя следующие шаги.

— Алгоритм AES использует блок подстановки (S-Box) для замены значений в процессе шифрования. S-Box — это таблица значений, которые используются для замены входных значений в процессе шифрования.

— Сдвиг строки (ShiftRow). Основная цель сдвига строки заключается в достижении разброса байтов в каждой строке, что представляет собой линейное преобразование.

— Смешивание колонок (MixColumn). Смешивание столбцов должно заменить преобразование умножением матрицы состояний и постоянной матрицы C для достижения диффузии в столбцах.

— Сложение по модулю 2 с ключом.

Основным элементом AES является ключ, который состоит из 128, 192 или 256 бит, и который используется для генерации ключей раунда. Ключ разбивается на четыре части, затем, из полученных формируется ключ раунда.

Для AES рекомендовано несколько режимов:

— ECB (англ. electronic code book) — режим «электронной кодовой книги» (простая замена);

— CBC (англ. cipher block chaining) — режим сцепления блоков;

— CFB (англ. cipher feed back) — режим обратной связи по шифротексту;

— OFB (англ. output feed back) — режим обратной связи по выходу.

Вывод

В данном разделе был рассмотрен алгоритм симметричного шифрования AES.

2 Конструкторский раздел

В данном разделе будет представлена схема алгоритма шифрования AES.

2.1 Алгоритм шифрования AES

На рисунке 2.1 изображена структурная схема шифрования AES.

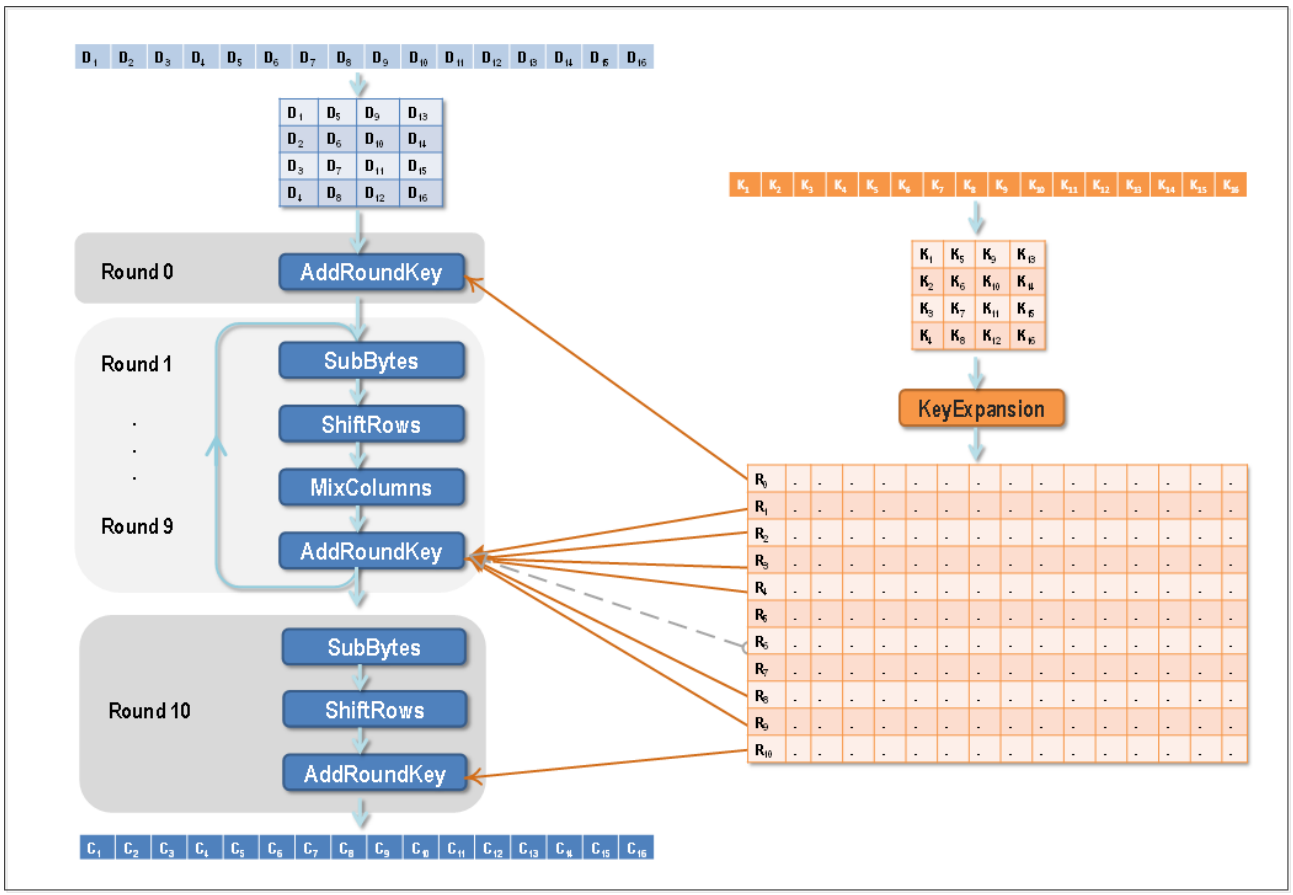

Рисунок 2.1 — Структурная схема шифрования AES

На рисунке 2.2 изображена схема блока SubBytes.

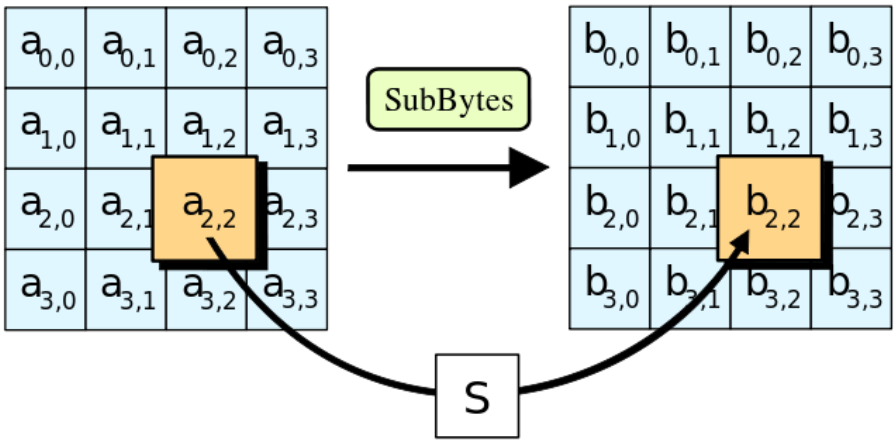

Рисунок 2.2 — Схема блока SubBytes

На рисунке 2.3 изображена схема блока ShiftRows.

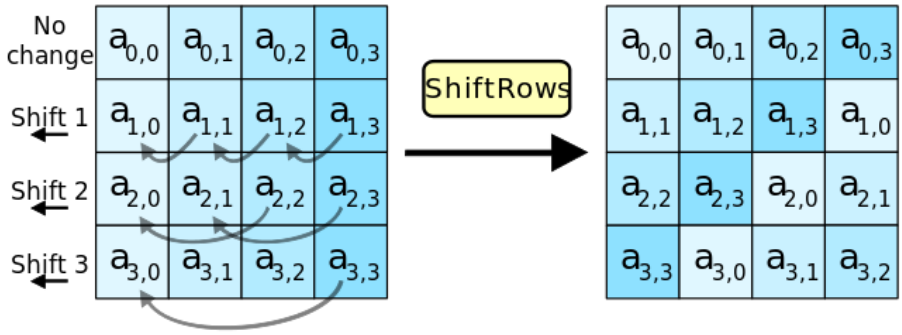

Рисунок 2.3 — Схема блока ShiftRows

На рисунке 2.4 изображена схема блока MixColumns.

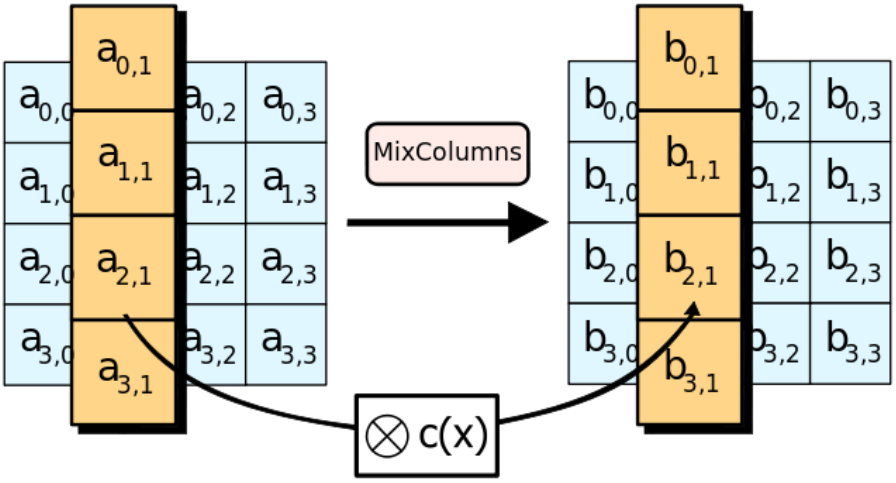

Рисунок 2.4 — Схема блока MixColumns

На рисунке 2.5 изображена схема блока AddRoundKey.

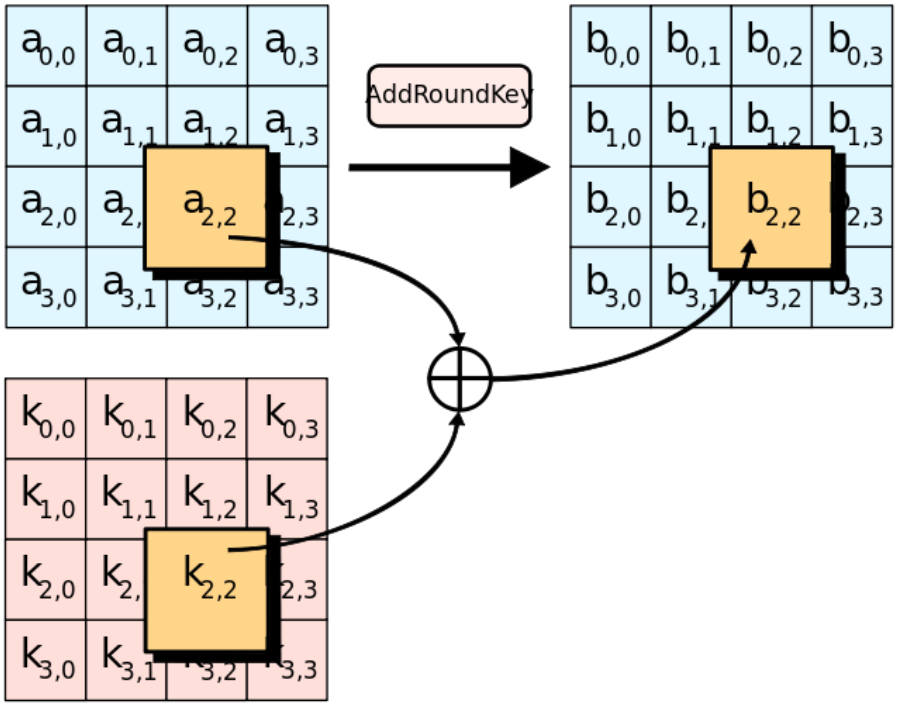

Рисунок 2.5 — Схема блока AddRoundKey

На рисунке 2.6 изображена структурная схема блока KeyExpansion.

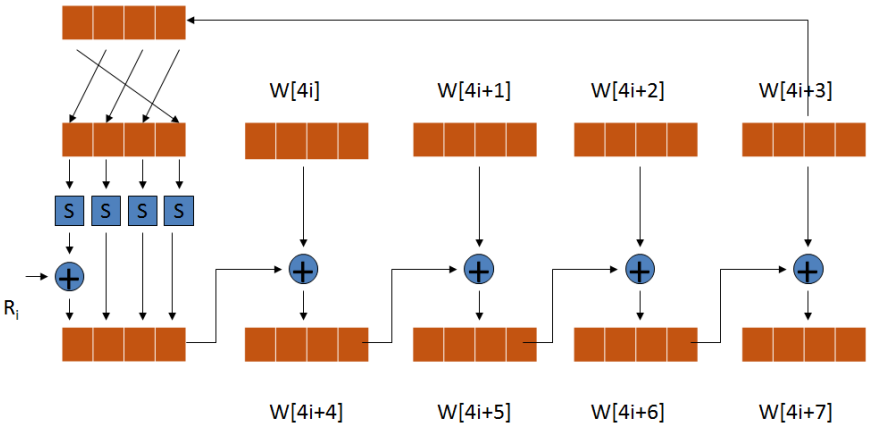

Рисунок 2.6 — Структурная схема блока KeyExpansion

Вывод

В данном разделе были приведены структурные схемы шифрования AES.

3 Технологический раздел

3.1 Средства реализации

В качестве языка программирования для реализации данной лабораторной работы использовался язык программирования C++ [1], так как он позволяет работать с файлами и массивами. В качестве среды разработки использовалась Visual Studio Code [2].

3.2 Реализация алгоритмов

В листингах 3.1–3.3 представлена реализация алгоритма шифрования AES.

Листинг 3.1 — Функция шифрования файла

```
1  int FileEncryption::cipher(string input, string output)
2  {
3          ifstream ifile;
4      ofstream ofile;
5      char inbuffer[16], outbuffer[16];
6
7      if(input.length() < 1)
8      ifile = ifstream(stdin);
9      else
10     ifile.open(input, ios::binary | ios::in | ios::ate);
11
12     if(output.length() < 1)
13     ofile = ofstream(stdout);
14     else
15     ofile.open(output, ios::binary | ios::out);
16     size_t size = ifile.tellg();
17     ifile.seekg(0, ios::beg);
18
19
20     size_t block = size / 16;
21
22     for(size_t i = 0; i < block; i++)
23     {
24         ifile.read(inbuffer, 16);
25         aes.Encrypt(inbuffer, 16, outbuffer);
26         ofile.write(outbuffer, 16);
27     }
28     if (size % 16 != 0)
29     {
30         int padding = 16 - (size % 16);
31
32         memset(inbuffer, 0, 16);
33         ifile.read(inbuffer, 16 - padding);
```

```
34          aes.Encrypt(inbuffer, 16, outbuffer);
35          ofile.write(outbuffer, 16);
36      }
37      ifile.close();
38      ofile.close();
39      return 0;
40  }
```

Листинг 3.2 — Функция шифратора

```
1  void AES::EncryptBlock(const unsigned char in[], unsigned char out[],
2  unsigned char key[]) {
3      unsigned char state[4][Nb];
4      unsigned int i, j, round;
5      unsigned char* roundKeys = new unsigned char[4 * Nb * (Nr + 1)];
6
7      KeyExpansion(key, roundKeys);
8
9      for (i = 0; i < 4; i++) {
10         for (j = 0; j < Nb; j++) {
11             state[i][j] = in[i + 4 * j];
12         }
13     }
14
15     AddRoundKey(state, roundKeys);
16
17     for (round = 1; round <= Nr - 1; round++) {
18         SubBytes(state);
19         ShiftRows(state);
20         MixColumns(state);
21         AddRoundKey(state, roundKeys + round * 4 * Nb);
22     }
23
24     SubBytes(state);
25     ShiftRows(state);
26     AddRoundKey(state, roundKeys + Nr * 4 * Nb);
27
28     for (i = 0; i < 4; i++) {
29         for (j = 0; j < Nb; j++) {
30             out[i + 4 * j] = state[i][j];
31         }
32     }
33     delete[] roundKeys;
34 }
```

Листинг 3.3 — Функция расширения ключей

```
1  void AES::KeyExpansion(const unsigned char key[], unsigned char w[]) {
```

```
2       unsigned char temp[4];
3       unsigned char rcon[4];
4
5       unsigned int i = 0;
6       while (i < 4 * Nk) {
7           w[i] = key[i];
8           i++;
9       }
10
11      i = 4 * Nk;
12      while (i < 4 * Nb * (Nr + 1)) {
13          temp[0] = w[i - 4 + 0];
14          temp[1] = w[i - 4 + 1];
15          temp[2] = w[i - 4 + 2];
16          temp[3] = w[i - 4 + 3];
17
18          if (i / 4 % Nk == 0) {
19              RotWord(temp);
20              SubWord(temp);
21              Rcon(rcon, i / (Nk * 4));
22              XorWords(temp, rcon, temp);
23          }
24          else if (Nk > 6 && i / 4 % Nk == 4) {
25              SubWord(temp);
26          }
27
28          w[i + 0] = w[i - 4 * Nk] ^ temp[0];
29          w[i + 1] = w[i + 1 - 4 * Nk] ^ temp[1];
30          w[i + 2] = w[i + 2 - 4 * Nk] ^ temp[2];
31          w[i + 3] = w[i + 3 - 4 * Nk] ^ temp[3];
32          i += 4;
33      }
34  }
```

3.3 Тестирование реализации алгоритма

Было проведено тестирование на следующих входных данных:

1. Входящая последовательность байтов:

00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff

Зашифрованный текст:

69, c4, e0, d8, 6a, 7b, 04, 30, d8, cd, b7, 80, 70, b4, c5, 5a

Расшифрованная последовательность байтов:

00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff

2. Входящая последовательность байтов:

00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f

Зашифрованная последовательность байтов:

69, c4, e0, d8, 6a, 7b, 04, 30, d8, cd, b7, 80, 70, b4, c5, 5a, 07, fe, ef, 74, e1, d5, 03, 6e, 90, 0e, ee, 11, 8e, 94, 92, 93

Расшифрованная последовательность байтов:

00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f

Все тесты пройдены успешно.

Вывод

В данном разделе были перечислены средства разработки, с помощью которых был реализованы алгоритм шифрования AES, приведена реализация алгоритма.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной лабораторной работы была достигнута цель работы: реализована программа шифрования симметричным алгоритмом AES.

Были решены все задачи — описан и реализован алгоритм шифрования AES с режимом шифрования OFB.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация языка C++ [Электронный ресурс]. — Режим доступа: `https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2013/n3690.pdf` (дата обращения: 13.11.2022).

2. Visual Studio Code [Электронный ресурс]. — Режим доступа: `https://code.visualstudio.com/docs` (дата обращения: 20.09.2022).