Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Отчет к лабораторной работе**:

Исследование блочных шифров

Выполнил:

студент 3 курса 4 группы

специальности ПОИТ

Дубалеко В.В.

Минск 2020

1. **Теоретические сведения**

Само название конструкции Фейстеля (сеть) означает ее ячеистую топологию. Формально одна ячейка сети соответствует одному раунду зашифрования или расшифрования сообщения. При зашифровании сообщение разбивается на блоки одинаковой (фиксированной) длины (как правило – 64 или 128 бит). Полученные блоки называются входными. В случае, если длина входного блока меньше, чем выбранный размер, то блок удлиняется установленным способом. Каждый входной блок шифруемого сообщения изначально делится на два подблока одинакового размера: левый (L0) и правый (R0). Далее в каждом i-ом раунде выполняются преобразования в соответствии с формальным представлением ячейки сети Фейстеля:

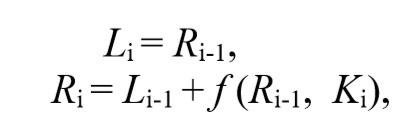


Рисунок 1.1 – Сеть Фейстеля

По какому-либо математическому правилу вычисляется раундовый ключ Ki. В приведенном выражение знак «+» соответствует поразрядному суммированию на основе «XOR». Расшифрование происходит так же, как и зашифрование, с той лишь разницей, что раундовые ключи будут использоваться в обратном порядке по отношению к зашифрованию. В своей статье [28] Х. Фейстель описывает два блока преобразований с использованием функции f (Ri-1, Ki):

• блок подстановок (S-блок, англ. S-box);

• блок перестановок (P-блок, англ. P-box).

Блок подстановок состоит из:

• дешифратора, преобразующего n-разрядное двоичное число в одноразрядное сигнал по основанию 2n;

• внутреннего коммутатора;

• шифратора, преобразующего сигнала из одноразрядного 2n-ричного в nразрядный двоичный.

В основе криптостойкости блочных шифров лежит идея К. Шеннона в представлении составного шифра таким образом, чтобы от обладал двумя важными свойствами: рассеянием и перемешиванием. Рассеивание должно скрыть отношения между зашифрованным текстом и исходным текстом. Рассеивание подразумевает, что каждый символ (символ или бит) в зашифрованном тексте зависит от одного или всех символов в исходном тексте.

Другими словами, если единственный символ в исходном тексте изменен, несколько или все символы в зашифрованном тексте будут также изменены. Идея относительно перемешивания заключается в том, что оно должно скрыть отношения между зашифрованным текстом и ключом.

Как было указано выше, в основе сети Фейстеля лежит простейшая операция суммирования 2-х (А + В) n-разрядных чисел – XOR: А + В (mod n). Помимо этой операции некоторые алгоритмы (Blowfish, IDEA, ГОСТ и др.) предусматривают выполнение операций сложения чисел по модулю более высоких порядков: XOR: А + В (mod 2n).

В DES входной блок данных, состоящий из 64 бит, преобразуется в выходной блок идентичной длины. В алгоритме широко используются рассеивания (подстановки) и перестановки битов текста, о которых мы упоминали выше. Комбинация двух указанных методов преобразования образует фундаментальный строительный блок DES, называемый раундом или циклом. Один блок данных подвергается преобразованию (и при зашифровании, и при расшифровании) в течение 16 раундов. После первоначальной перестановки и разделения 64-битного блока данных на правую (R0) и левую (L0) половины длиной по 32 бита выполняются 16 раундов одинаковых действий.

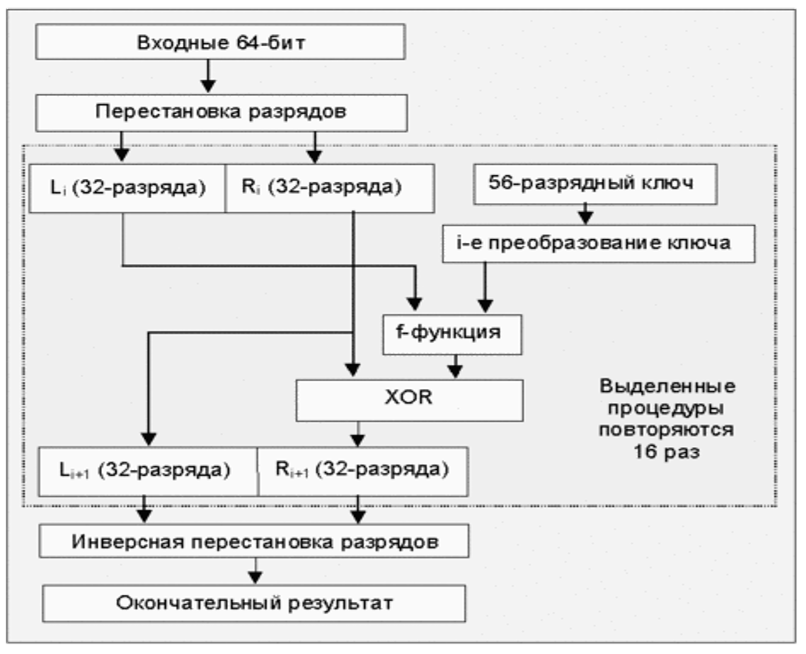


Рисунок 1.2 – Общая схема алгоритма DES

Первоначальные ключи являются слабыми. Вспомним, что первоначальное значение разделяется на две половины, каждая из которых сдвигается независимо. Если все биты каждой половины равны 0 или 1, то для всех раундов алгоритма используется один и тот же ключ. Это может произойти, если ключ состоит из одних 1, из одних 0, или если одна половина ключа состоит из одних 1, а другая – из одних 0. Коммутационная панель обеспечивает самый большой набор возможных настроек. Для первого кабеля одна сторона может иметь любое из 26 положений, а другая сторона - любое из 25 оставшихся положений (одна буква коммутируются с другой). Однако, поскольку комбинация и ее обратная сторона идентичны (AB такая же, как BA), мы должны игнорировать все двойные числа во всех возможных комбинациях для одного кабеля, предоставляя (26 x 25) / (1! х 2^1) или 325 уникальных способов коммутаций одним кабелем. Для двух кабелей: есть (26 x 25) комбинаций – для первого кабеля и, поскольку два разъема уже используются, то получается (24 x 23) комбинаций – для второго кабеля. Следуя этой простой логике, получается (26 x 25 x 24 x 23) / (2! X 2^2) = 44 850 уникальных способов коммутаций с использованием двух кабелей. Для трех кабелей – (26 x 25 x 24 x 23 x 22 x 21) / (3! х 2^3) = 3 453 450 комбинаций и так далее. Таким образом, с использованием 10 кабелей на коммутационной панели получаются 150 738 274 937 250 различных комбинаций. Формула, где n равно количеству кабелей, равна 26! / (26 - 2n)! · n! · 2n. Численно это дает: 60 x 17 576 x 676 x 150 738 274 937 250 = 107 458 687 327 250 619 360 000 или 1,07 x 10^23.

Кроме того, некоторые пары ключей при зашифровании переводят открытый текст в идентичный шифртекст. Иными словами, один из ключей пары может расшифровать сообщения, зашифрованные другим ключом пары. Это происходит из-за метода, используемого DES для генерации подключей: вместо 16 различных подключей эти ключи генерируют только два различных подключа. В алгоритме каждый из этих подключей используется восемь раз. Эти ключи, называемые полуслабыми.

Cуществуют несколько реализаций алгоритма 3DES. Вот некоторые из них:

• DES-EEE3: шифруется 3 раза с 3 разными ключами (операции шифрование-шифрование-шифрование);

• DES-EDE3: 3DES операции шифрование-расшифрование-шифрование с разными ключами:

• DES-EEE2 и DES-EDE2: как и предыдущие, однако, на первом и третьем шаге используется одинаковый ключ.

1. **Практическая часть**

В данной лабораторной работе необходимо разработать пользовательское приложение, которое должно реализовывать следующие операции:

• разделение входного потока данных на блоки требуемой длины с необходимым дополнением последнего блока;

• выполнение требуемых преобразований ключевой информации;

• выполнение операций зашифрования/расшифрования;

• оценка скорости выполнения операций зашифрования/расшифрования;

• пошаговый анализ лавинного эффекта с подсчетом количества изменяющихся символов по отношению к исходному слову. С помощью разработанного приложения зашифровать сообщение в соответствии с п.1 практического задания, применив не менее 5 вариантов начальных установок роторов.

В связи с поставленными требованиями было разработано приложение реализующее алгоритм DES-EDE3 на основе встроенного классов javax.crypto.Cipher и javax.crypto.spec.SecretKeySpec в Java.

Для того что бы приступишь к шифрованию или дешифрованию файла мы сначала должны создать необходимый ключ при помощи библиотеки javax.crypto.spec.SecretKeySpec:



Рисунок 2.1 – Ключ для шифрования и дешифрования

Пример шифрации:

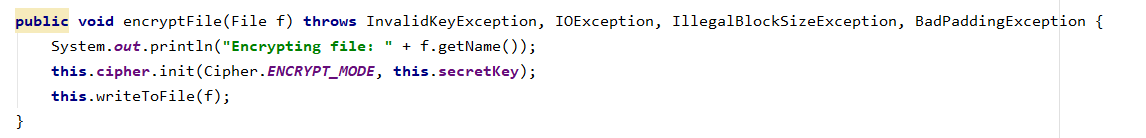


Рисунок 2.2 – Листинг шифрования

Пример дешифрации:

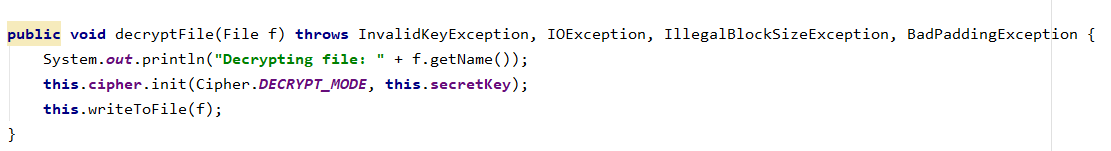


Рисунок 2.3 – Листинг дешифрования

После выполнения шифрации файла мы получим файл с нечитаемыми крякозабрами на подобии тех, что показаны на рисунке ниже.

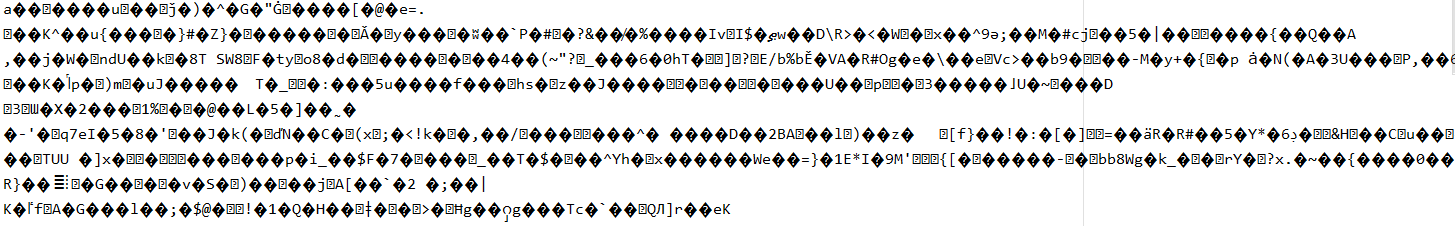


Рисунок 2.4 – Пример зашифрованного текста

**Вывод**

В данной лабораторной работе я закрепил теоретические знания по блочным шифрам и их разновидностям. А так же разработал приложение для шифрации/дешифрации по DES алгоритму.