Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Исследование блочных шифров**

Студент: Гвоздовский К.В.

ФИТ 3 курс 6 группа

Преподаватель:

Нистюк Ольга Александровна

1. **Цель работы**

Изучение и приобретение практических навыков разработки и использования приложений для реализации блочных шифров.

1. **Задание**

Разработать авторское приложение в соответствии с целью лабораторной работы. При этом можно воспользоваться готовыми библиотеками либо программными кодами, реализующими некоторые блочные алгоритмы.

Приложение должно реализовывать следующие операции:

1. разделение входного потока данных на блоки требуемой длины с необходимым дополнением последнего блока;
2. выполнение требуемых преобразований ключевой информации;
3. выполнение операций зашифрования/расшифрования;
4. оценка скорости выполнения операций зашифрования/расшифрования;
5. пошаговый анализ лавинного эффекта с подсчетом количества изменяющихся символов по отношению к исходному слову.

Также, нужно проанализировать влияние слабых ключей и полуслабых ключей на конечный результат зашифрования и на лавинный эффект и оценить степень сжатия (используя любой доступный архиватор) открытого текста и соответствующего зашифрованного текста. Дать пояснения к полученному результату.

1. **Ход работы**

**Теоретическая часть**

В симметричных системах Отправитель и Получатель используют один и тот же ключ К.

К блочным шифрам относятся: DES, FEAL, RC2, Blowfish, TEA, RC5. CAST-128, XTEA, XXTEA, RC6, AES.

**Все перечисленные стандарты и алгоритмы блочных шифров (БШ) строятся на основе подстановочных и перестановочных шифров, т. е. являются комбинационными. БШ относятся также к классу симметричных.**

Блочное шифрование (расшифрование) предполагает разбиение исходного открытого (зашифрованного) текста на равные блоки, к которым применяется однотипная процедура зашифрования (расшифрования).

Указанная однотипность характеризуется прежде всего тем, что процедура зашифрования (расшифрования) состоит из совокупности повторяющихся наборов преобразований, называемых **раундами**.

Основные требования к шифрам рассматриваемого класса можно сформулировать следующим образом:

• даже незначительное изменение исходного сообщения

должно приводить к существенному изменению зашифрованного

сообщения;

• устойчивость к атакам по выбранному тексту;

• алгоритмы зашифрования/расшифрования должны быть реализуемыми на различных платформах;

• алгоритмы должны базироваться на простых операциях;

• алгоритмы должны быть простыми для написания кода, вероятность появления программных ошибок должна быть низкой;

• алгоритмы должны допускать их модификацию при переходе

на иные требования по уровню криптостойкости.

**Сеть Фейстеля**

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) длинны (как правило – 64 или128 бит).

Каждый входной блок шифруемого сообщения изначально делится на два подблока одинакового размера: левый (*L0*) и правый (*R0*).

Два блока преобразований:

– блок подстановок (S - блок)

Дешифратор преобразует n-разрядное двоичное число в одноразрядный сигнал по основанию 2^n

Шифратор преобразует сигнал из одноразрядного 2^n-ричного в n-разрядный двоичный

– блок перестановок (P - блок)

**Симметричная криптография. Алгоритм DES**

Базовые методы – подстановка и перестановка данных: 1 под. + 1 перест. – раунд (цикл)

Алгоритм состоит из 16 раундов, т.е. один блок данных длиной 64 бита обрабатывается 16 раз, в каждом из которых используется новый ключ: в каждый раз биты ключа сдвигаются, затем из 56 бит выбирается 48 бит.

Блок перестановок изменяет положение цифр т.е. является линейным устройством.

**Существуют 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.

**Практическая часть**

В соответствии с вариантом был выбран алгоритм DES-EDE3.

Для зашифрования сообщений была разработана функция Encrypt(). Она принимает в качестве параметра строку и ключ, с помощью которого будет осуществляться шифрование. Поскольку алгоритм DES-EDE3, то сначала мы будем шифровать сообщение, потом расшифровывать, потом обратно зашифровывать. И все эти три операции проводятся с разными ключами. Код функции для шифрования представлен на рисунке 3.1.

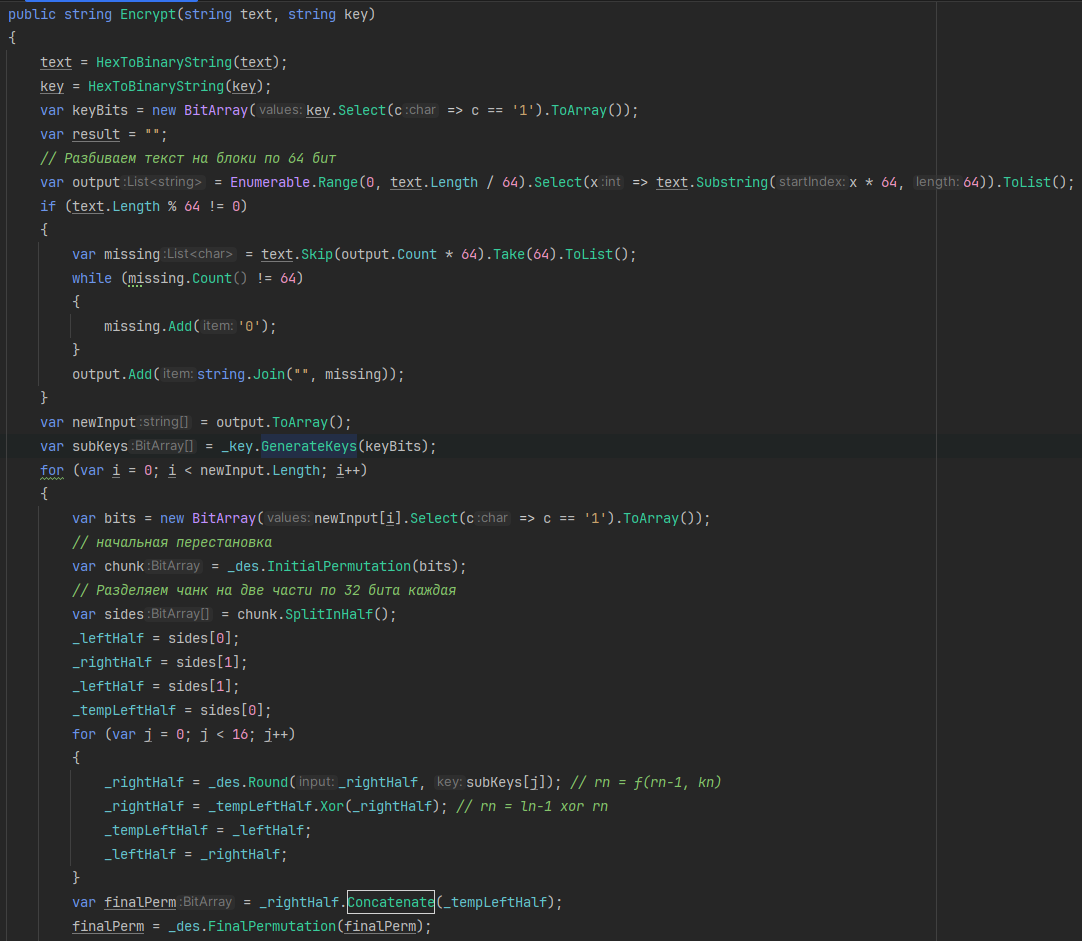


Рисунок 3.1 – Функция Encrypt(), реализующая алгоритм 3DES

Для расшифрования сообщения была разработана функция Decrypt(). Стоить заметить, что для расшифрования сообщения нужно использовать функции в обратном порядке: расшифрование – зашифрование – расшифрование. Код функции для расшифрования представлен на рисунке 3.2.

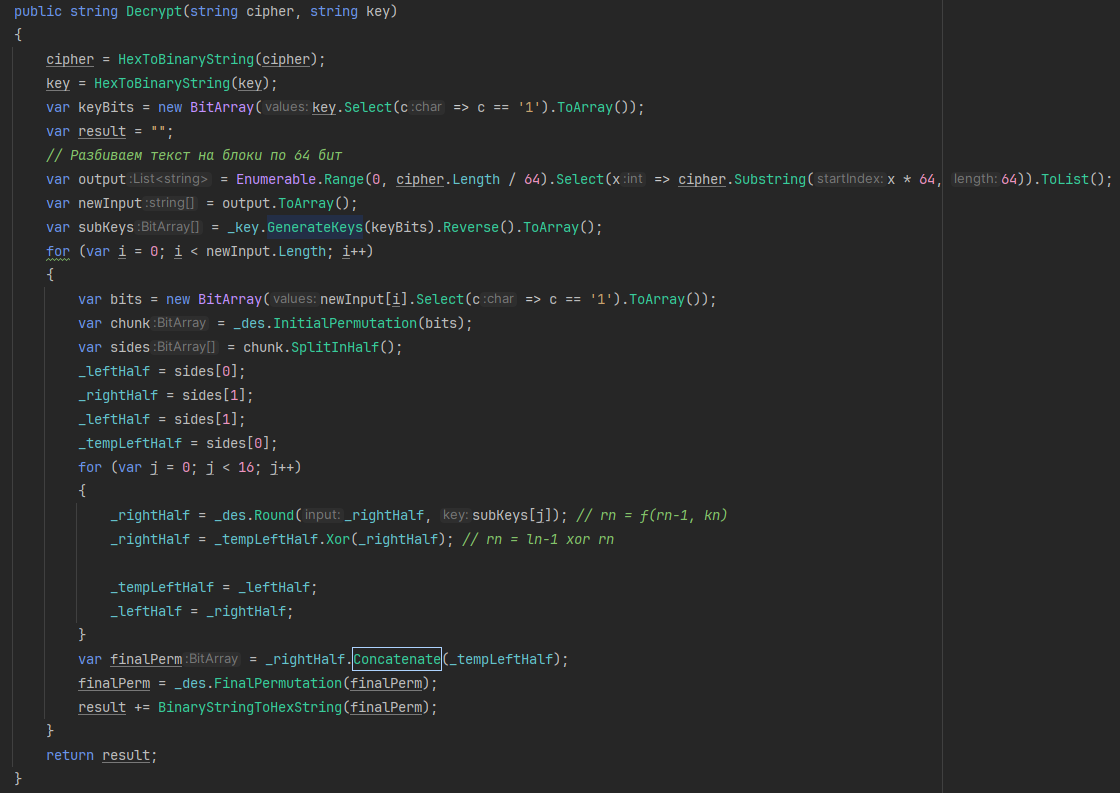


Рисунок 3.2 – Функция Decrypt(), реализующая алгоритм 3DES

Далее рассмотрим пример использования этих функций. Он представлен на рисунке 3.3.

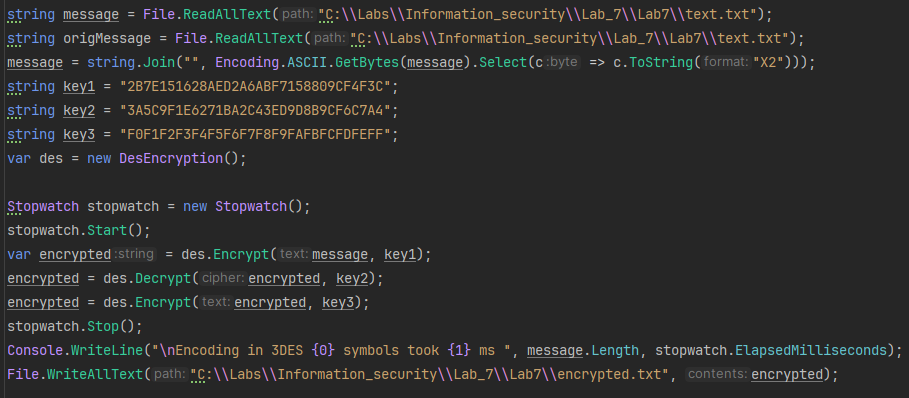


Рисунок 3.3 – Использование функций Encrypt() и Decrypt()

Результаты выполнения функций Encrypt() и Decrypt() представлены на рисунках 3.4 и 3.5.

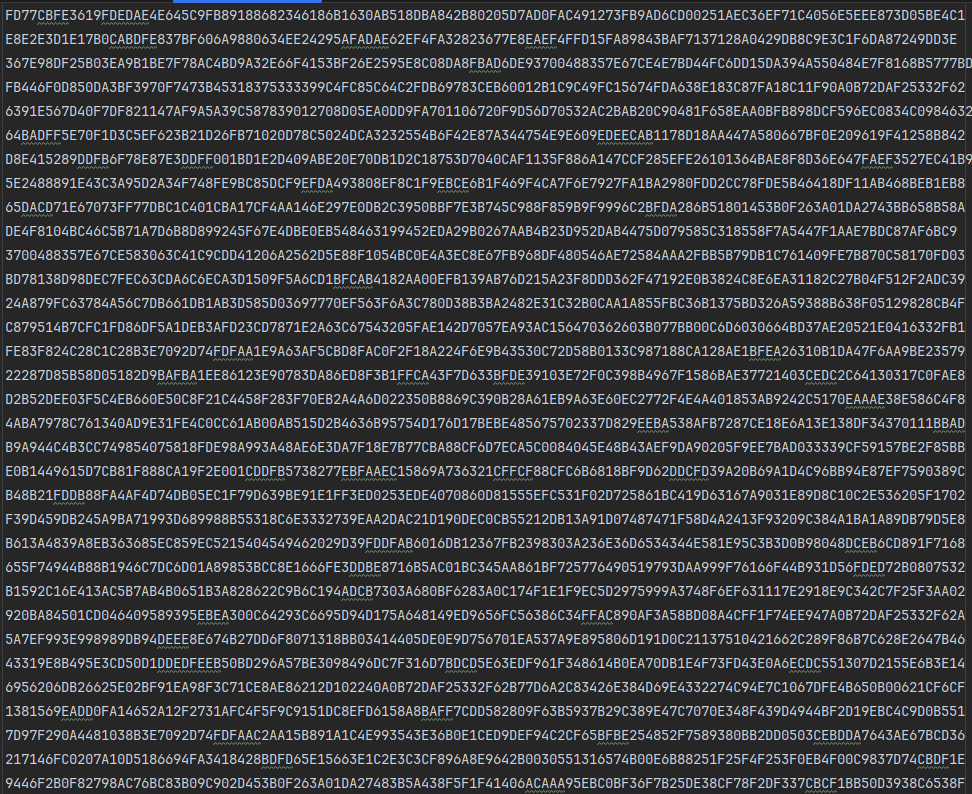


Рисунок 3.4 – Результат зашифрования сообщения

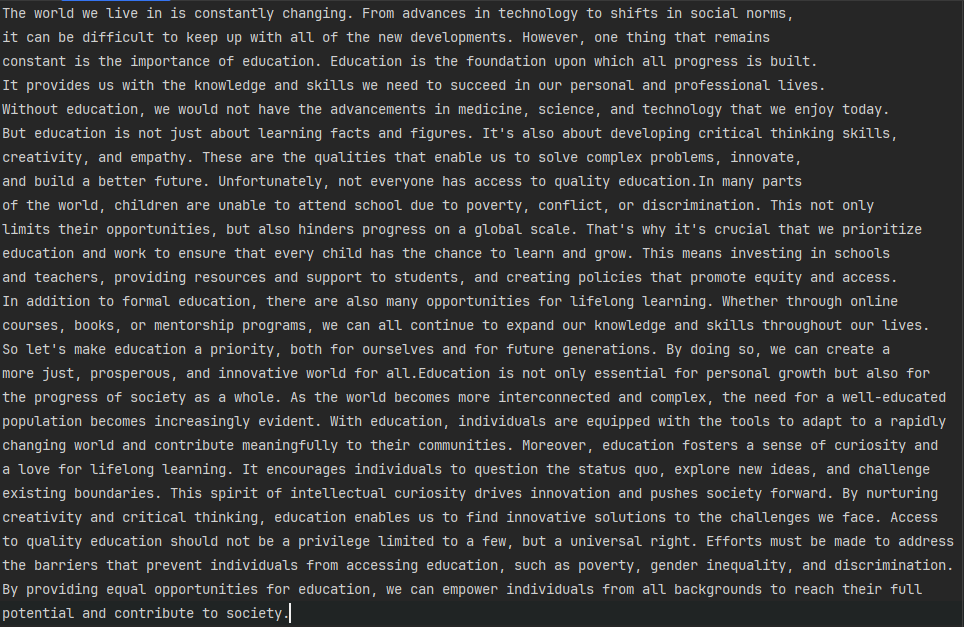


Рисунок 3.5 – Результат расшифрования сообщения

Также оценим скорость выполнения данных операций. Результаты представлены на рисунках 3.6 – 3.7.

Рисунок 3.6 – Скорость выполнения операции зашифрования

Рисунок 3.7 – Скорость выполнения операции расшифрования

Следующим шагом нужно было пошагово проанализировать лавинный эффект с подсчетом количества изменяющихся символов по отношению к исходному слову.

Лавинный эффект в криптографии означает, что небольшое изменение входного сообщения должно приводить к значительному изменению зашифрованного сообщения. Для проверки лавинного эффекта можно изменить один символ в исходном сообщении, затем зашифровать исходное сообщение и измененное сообщение и сравнить количество измененных символов в зашифрованных сообщениях.

Следующий код, представленный на рисунке 3.8 зашифрует исходное сообщение, изменит один символ в исходном сообщении и зашифрует его, затем посчитает количество измененных символов в зашифрованных сообщениях. Кроме того, он расшифрует исходное зашифрованное сообщение и выведет его на экран для проверки.

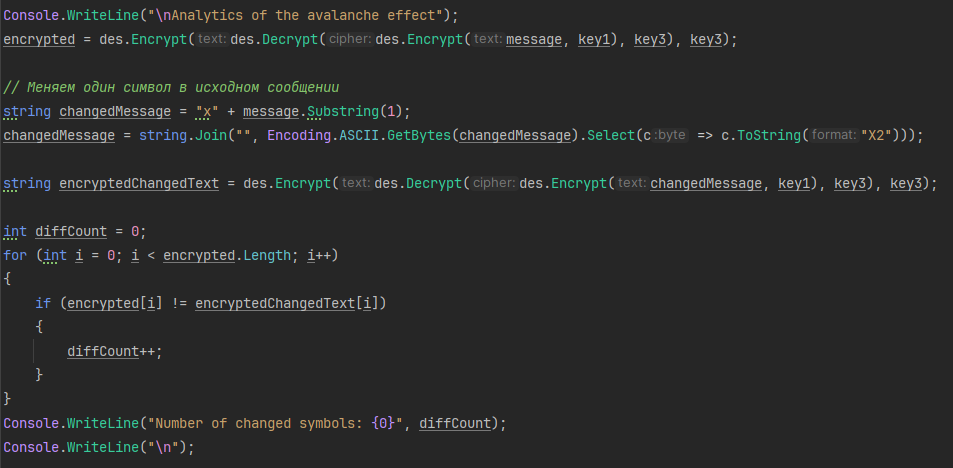


Рисунок 3.8 – Код для анализа лавинного эффекта

На рисунке 3.9 представлен результат работы данного кода для исходного сообщения.

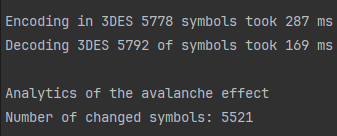


Рисунок 3.9 – Демонстрация лавинного эффекта

Из рисунка мы видим, что после изменения одного символа в сообщении у нас изменилось 5521 символов.

Далее, нужно было проанализировать влияние слабых ключей и полуслабых ключей на конечный результат зашифрования и на лавинный эффект. Ниже, на рисунке 3.9, приведён результат использования слабых и полуслабых ключей и то, как они влияют на количество изменённых символов в сравнении с исходным сообщением, а также на лавинный эффект.

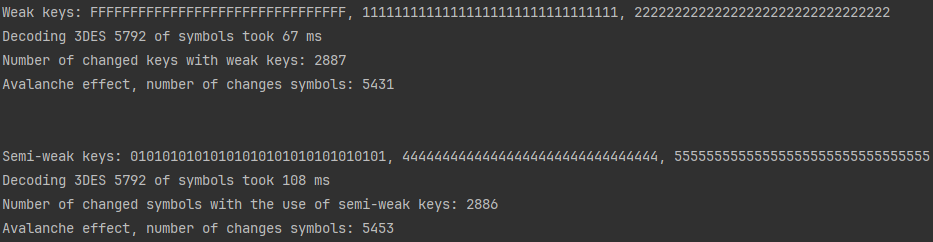


Рисунок 3.10 – Влияние слабых и полуслабых ключей на результат шифрования и на лавинный эффект

Из рисунка мы видим, что изменилось 5431 символов при изменении одного символа и использовании слабых ключей. При использовании полуслабых ключей изменилось 5453 символа.

На основе данных значений мы можем сделать вывод влияния ключа на лавинный эффект. Чем слабее ключ, тем меньше будет проявляться лавинный эффект, то есть будет меньше изменяться символов при изменении одного.

Количество измененных символов в зашифрованном сообщении, особенно в таком большом объеме, является плохим с точки зрения безопасности и целостности данных. Когда большое количество символов изменяется в процессе шифрования, это может привести к таким проблемам, как отслеживаемость, нарушение целостности данных, уязвимость к атакам. В целом, количество измененных символов в зашифрованном сообщении должно быть минимальным, чтобы обеспечить надежность и безопасность шифрования.

Кроме того, нужно было оценить степень сжатия открытого и зашифрованного текста. Для этого посмотрим на рисунки 3.10 – 3.11.



Рисунок 3.11 – Сжатие исходного текстового файла



Рисунок 3.12 – Сжатие зашифрованного текстового файла

Таким образом, можно увидеть закономерность, что зашифрованные файлы меньше или же вовсе не сжимаются.

**Вывод**

В ходе лабораторной работы были изучены и приобретены практические навыки разработки и использования приложений для реализации блочных шифров.

Также было разработано авторское приложение в соответствии с целью лабораторной работы.