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

Карленок Ю.А.

Минск 2020

1. **Теоретические сведения**

В 1972 г. Национальное бюро стандартов США (ныне – Национальный институт стандартов и технологии, National Institute of Standarts & Technology – NIST) инициировал программу защиты каналов связи и компьютерных данных. Одна из целей – разработка единого стандарта криптографического шифрования. Основными критериями оценки алгоритма являлись:

* алгоритм должен обеспечить высокий уровень защиты,
* алгоритм должен быть понятен и детально описан,
* криптостойкость алгоритма должна зависеть только от ключа,
* алгоритм должен допускать адаптацию к различным применениям,
* алгоритм должен быть разрешен для экспорта.

В качестве начального варианта нового алгоритма рассматривался Lucifer – разработка компании IBM начала семидесятых годов. В основе указанного алгоритма использовались два запатентованных 1971 г. Хорстом Фейстелем (Horst Feistel) устройства, реализующие различные алгоритмы шифрования, позже получившие шифр (сеть) Фейстеля (Feistel cipher, Feistel network). В первой версии проекта Lucifer сеть Фейстеля не использовалась.

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

Блочное зашифрование (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**

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:

Li = Ri-1,

Ri = Li-1 + f (Ri-1, Ki)

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 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).

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.

**DES**

Стандарт шифрования данных DES (DATA ENCRYPTION STANDARD) – блочный шифр с симметричными ключами, разработан Национальным Институтом Стандартов и Технологии (NIST – National Institute of Standards and Technology).

Для шифрования DES принимает 64-битовый открытый текст и порождает 64-битовый зашифрованный текст и наоборот, получив 64 бита зашифрованного текста, он выдает 64 бита расшифрованного. В обоих случаях для шифрования и дешифрования применяется один и тот же 56-битовый ключ.

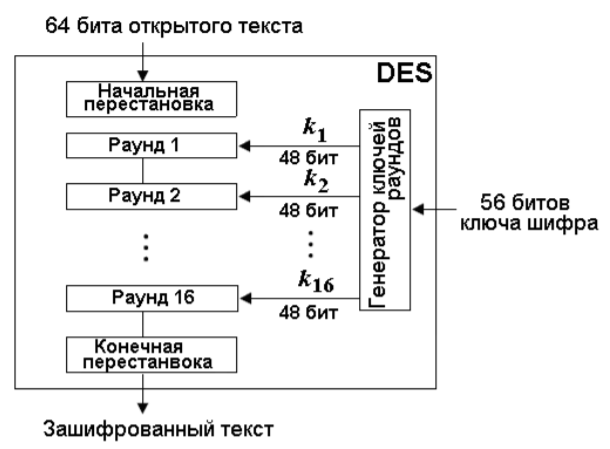


Рисунок 1.1 – Схема работы DES

Процесс шифрования состоит из двух перестановок, которые называют начальной и финальной (конечной) перестановками, и 16 раундов Фейстеля. Каждый раунд использует различные сгенерированные 48-битовые ключи.

На вход каждой из них поступает 64 бита, которые затем переставляются в соответствии с заданными таблицами. Эти перестановки взаимно обратны. Другими словами, 58-й бит на входе начальной перестановке переходит в 1-ую позицию на выходе из нее. А финальная перестановка 1-ый входной бит переведет в 58-ую позицию на выходе.



Таблица 1.2 – Начальная и конечная перестановки

DES использует 16 раундов. Каждый раунд DES применяет шифр Фейстеля, как это показано на рисунке.

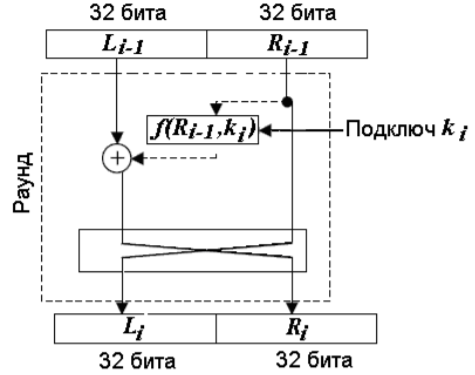


Рисунок 1.3 – Раунды DES

Раунд принимает полублоки Li−1 и Ri−1 от предыдущего раунда (или начального блока перестановки) и создает полублоки Li и Ri для входа в следующий раунд (или конечный блок перестановки). Все необратимые элементы сосредоточены в функции 1 ( , ) i i f R k − .

Функция DES с помощью 48-битового ключа зашифровывает 32 самых правых бит Ri−1 , чтобы получить на выходе 32-битовый результат. Эта функция содержит, как это показано на 4 составляющие: операция XOR, P-бокс расширения, группу S -боксов и прямой бокс.

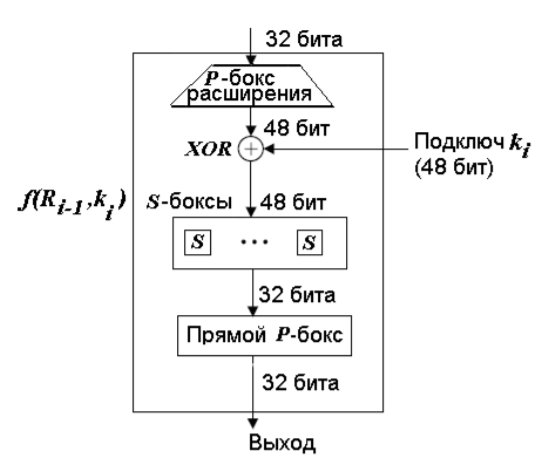


Рисунок 1.4 – Перестановочные боксы

P-бокс расширения служит для расширения 32-битового блока Ri−1 до 48 битов, чтобы согласовать его размеров с размерами подключа раунда. Блок Ri−1 делится на 8 секций по 4 бита. Каждая секция расширяется до 6 бит. (Для секции значения входных битов в позициях 1, 2, 3 и 4 присваиваются битам в позициях 2, 3, 4 и 5 соответственно на выходе. 1-ый выходной бит равен входному 4-му биту предыдущей секции; 6-ой бит выхода равен 1-му биту следующей секции. Если секции 1 и 8 рассматривать как соседние секции, то те же самые правила применяются к битам 1 и 32).

Хотя отношения между входом и выходом могут быть определены математически, P-бокс задают таблицей.



Таблица 1.5 – P-бокс расширения в i-том раунде

После расширения DES использует операцию XOR над расширенной частью правого полублока Ri−1 и ключом раунда i k . После суммирования с битами ключа блок из 48 битов делится на 8 последовательных 6-битових векторов 1 2 8 b ,b ,...,b , каждый из которых заменяется на 4-битовий вектор j b′ с помощью S -боксов.

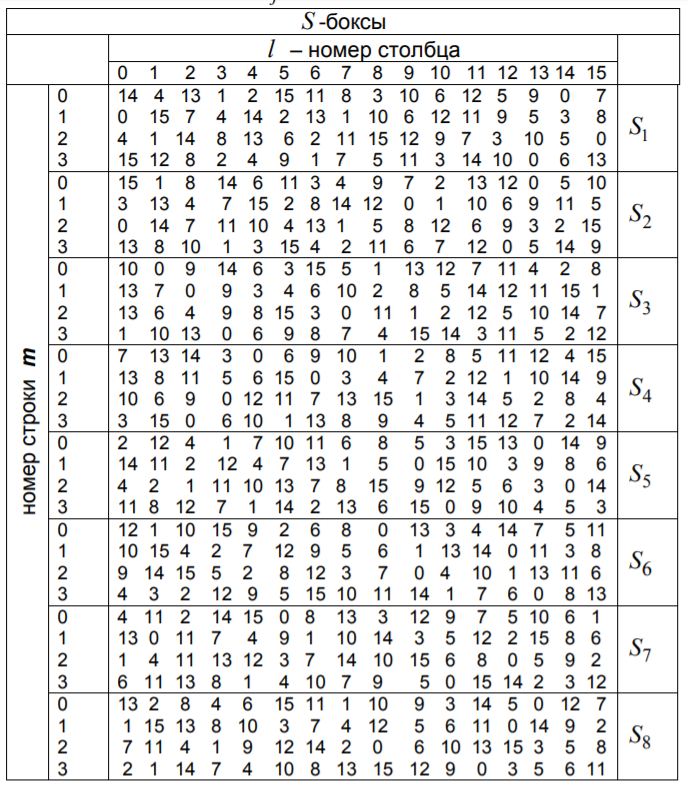


Таблица 1.6 – Подстановочный S-бокс

Таким образом, после S-боксов мы получаем 8 4-битовых векторов 1 2 8 b b b ′ ′ ′ , ,..., , которые опять объединяют в 32-битовый блок. Далее биты блока перетасовываются в прямом P-боксе на основе заданной таблицы (правила пользования таблицей перестановки старые: например, 7-ой бит входа станет 2-ым битом выхода).

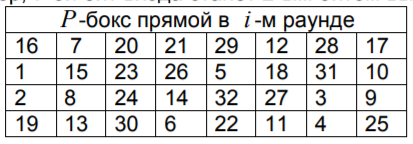


Таблица 1.7 – P-бокс прямой в i-том раунде

Посля 16-го раунда DES правый и левый блоки уже не меняются местами, а объединяются в блок R16L16 и подвергаються финальной перестановке −1 IP.

Генерация ключей.

DES создает 16 раундовых ключей i k по 48 битов из ключа k шифра на 56 битов. Однако, чтобы задать ключ шифра надо среди 56 битов ключа дополнительно вписать 8 битов в позиции 8,16,...,64 для проверки четности таким образом, чтобы каждый байт содержал нечетное число единиц. С помощью этой операции выявляют ошибки при обмене и хранении ключей.

Ключевое расписание состоит из этапов:

1). Перестановка сжатия для удаления битов проверки – из 64- битового ключа удаляют биты 8,16,24, 32,…,64 и переставляет остальные биты согласно таблице (в ходе перестановки сохраняется нумерация битов расширенного ключа).



Таблица 1.8 – Удаление проверочных битов ключа и перестановка

2) После перестановки 56 битов ключа делятся на два блока C0 и D0 по 28 бит каждый. Дале для генерации раундовых ключей из блоков C0 и D0 с помощью операции циклического сдвига влево на 1-2 бита строятся блоки Ci и Di , i =1,2,...,16 . В раундах 1,2,9 и 16 смещение – на 1 бит, в других раундах — на 2 бита. После определения блоков Ci и Di биты этих блоков объединяются в один ключ на 56 битов.

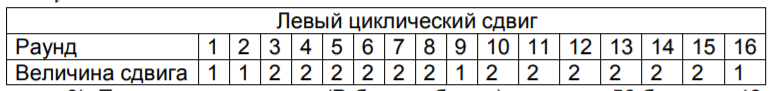


Таблица 1.9 – Сдвиги блоков С0 и D0

3). Перестановка сжатия (P-бокс, таблица) изменяет 56 битов на 48 битов, которые образуют раундовый ключ.



Таблица 1.10 – Перестановка сжатия ключа

При расшифровании – раундовые ключи теже, что и при зашифровании, но теперь они используются в обратном порядке.

Лавинный эффект – проявление зависимости всех выходных битов шифротекста от каждого входного бита открытого текста (в криптографии такой анализ проводят для блочных шифров и хэшфункций). Лавинный эффект проявляется в зависимости всех выходных битов от каждого входного бита. Термин введен Фейстелем, хотя концептуальное понятие использовалось еще Шенноном. Если криптографический алгоритм не обладает лавинным эффектом в достаточной степени, противник может сделать предположение о входной информации, основываясь на выходной информации. Таким образом, достижение лавинного эффекта является важной целью при разработке криптографического алгоритма.

Криптоалгоритм удовлетворяет лавинному критерию, если при изменении одного бита на входе алгоритма изменяется в среднем половина битов на выходе алгоритма. Если же при изменении одного бита на входе каждый бит на выходе изменяется с вероятностью ½, то криптоалгоритм удовлетворяет строгому лавинному критерию.

В DES лавинный эффект проявляется уже на 4-5 раунде. Так, если зашифровать на одном ключе с помощью DES, 2 блока открытого текста, отличающиеся одним битом, то блоки шифротекстов будут отличаться на 29 бит, т.е. изменение открытого текста на 1,5% вызывает 45% изменений шифротекста.

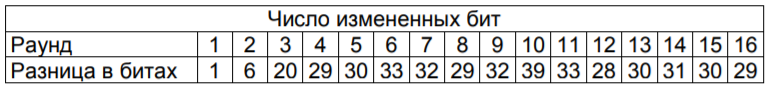


Таблица 1.11 – Изменение битов по раундам

Проблемы ключей.

Ключевое расписание ключей для раундов допускает слабые ключи. При генерировании раундовых ключей 56-битный ключ шифра делится на две половины и каждая из них сдвигается независимо. Если ключ шифра состоит только из 0 или 1 или, если одна его половина из 0, а другая – из 1, то в этом случае раундовые ключи оказываются попарно одинаковыми, т.е. k1 = k16, k2 = k15 и т.д., и процедуры шифрования/дешифрования оказываются идентичными. Формально слабым ключом DES называется такой 56-битный ключ K , при котором Ek(Ek(X)) = X, где X – 64-битный блок открытого текста. Среди всех 2^56 возможных ключей DES имеется 4 слабых.

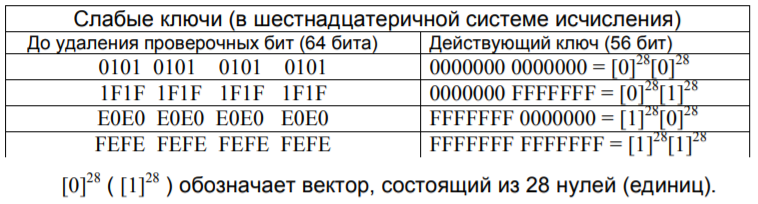


Таблица 1.12 – Слабые ключи

Для алгоритма DES кроме слабых ключей существуют еще полуслабые ключи – это пара 56-битных ключей k1, k2, при которой DESk1(DESk2(X)) = X. Полуслабые ключи дают одинаковый результат при шифровании текстов. Это тоже связано с генерацией раундовых ключей – вместо 16 различных ключей генерируются только 2 различных, которые затем используются 8 раз в алгоритмах. Таких ключей у DES 6 пар.

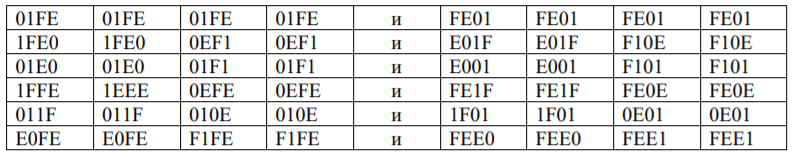


Таблица 1.13 – Полуслабые ключи

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.

В связи с поставленными требованиями было разработано пользовательское приложение, которое в точности повторяет алгоритм DES (без использования библиотек). Приложение поддерживает стандарт шифрования DES. Ключ автоматически обрезается до 64-бит (в последствии в алгоритме отсекутся 8-битов, отвечающих за контрольную сумму). Про переполнении исходного текста количество бит расширяется до 64-бит (также с выводом).

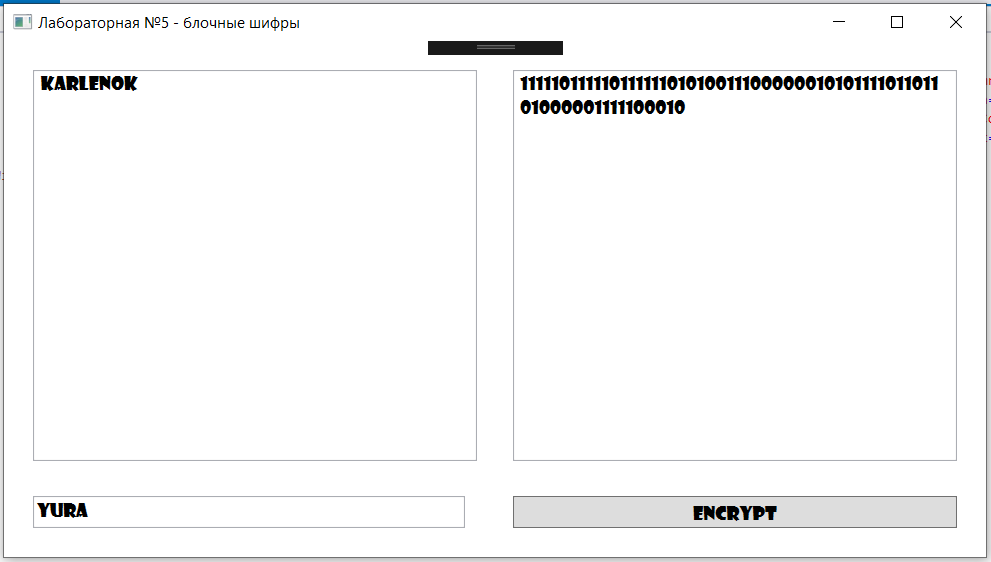


Рисунок 2.1 – Пользовательское приложение-шифратор

Также было разработано приложение-дешифровщик, которое позволяет по введенному ключу и шифротексту получить исходный текст (могут быть переполнения в связи с необходимостью расширить).

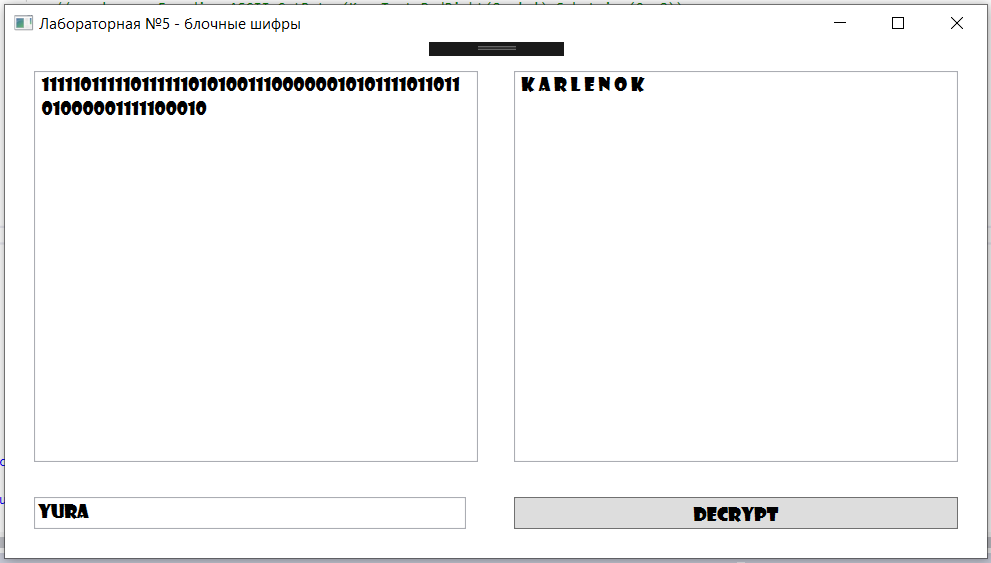


Рисунок 2.2 – Пользовательское приложение-дешифратор

Также на готовом примере были детально разобраны все фазы шифрования стандартом DES:

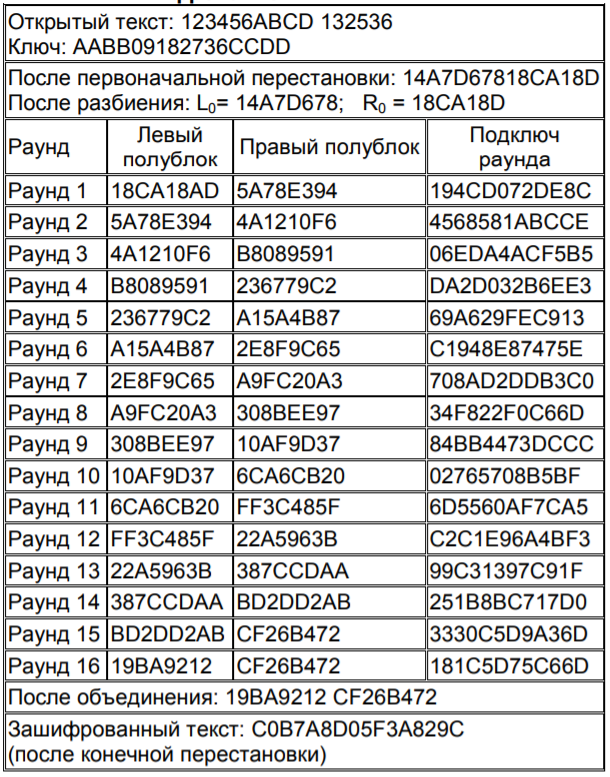


Таблица 2.3 – Пошаговый разбор фаз DES

**Вывод**

В данной лабораторной работе я закрепил теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости блочных шифров. Разработал приложение для реализации указанных преподавателем методов блочного зашифрования/расшифрования. Выполнил анализ криптостойкости блочных шифров. А также оценил скорость заширфования/расшифрования реализованных шифров.