Министерство образования Республики Беларусь Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Кафедра информатики

ОТЧЕТ

по лабораторной работе №1

**Симметричная криптография. Двойной и тройной DES**

Выполнил: Проверил:

студент гр. 553503 Чернявский Ю.А

Криницин А.В.

Минск 2018

**Задание:** изучить шифрование данных при помощи классического алгоритма DES и реализовать его программно.

**Введение**

**DES** (англ. ***d****ata****e****ncryption****s****tandard*) - алгоритм для симметричного шифрования, разработанный фирмой IBM и утверждённый правительством США в 1977 году как официальный стандарт (FIPS 46-3). Размер блока для DES равен 64 бита. В основе алгоритма лежит сеть Фейстеля с 16 циклами (раундами) и ключом, имеющим длину 56 бит. Алгоритм использует комбинацию нелинейных (S-блоки) и линейных (перестановки E, IP, IP-1) преобразований. Для DES рекомендовано несколько режимов:

* ECB (англ. ***e****lectronic****c****ode****b****ook*) — режим «электронной кодовой книги» (простая замена);
* CBC (англ. ***c****ipher****b****lock****c****haining*) — режим сцепления блоков;
* CFB (англ. ***c****ipher****f****eed****b****ack*) — режим обратной связи по шифротексту;
* OFB (англ. ***o****utput****f****eed****b****ack*) — режим обратной связи по выходу;
* CTR (Counter Mode) — режим счетчика.

Прямым развитием DES в настоящее время является алгоритм Triple DES (3DES). В 3DES шифрование/расшифровка выполняются путём троекратного выполнения алгоритма DES.

Входными данными для блочного шифра служат:

* блок размером n бит;
* ключ размером k бит.

На выходе (после применения шифрующих преобразований) получается зашифрованный блок размером n бит, причём незначительные различия входных данных, как правило, приводят к существенному изменению результата.

Блочные шифры реализуются путём многократного применения к блокам исходного текста некоторых базовых преобразований.

Базовые преобразования:

* сложное преобразование на одной локальной части блока;
* простое преобразование между частями блока.

Так как преобразования производятся поблочно, требуется разделение исходных данных на блоки необходимого размера. При этом формат исходных данных не имеет значения (будь то текстовые документы, изображения или другие файлы). Данные должны интерпретироваться в двоичном виде (как последовательность нулей и единиц) и только после этого должны разбиваться на блоки. Все вышеперечисленное может осуществляться как программными, так и аппаратными средствами.

**Алгоритм**

1. Исходный текст – блок 64 бит.
2. Начальная перестановка – исходный текст **T (блок 64 бита)** преобразуется с помощью начальной перестановки **IP** (Рис. 1)

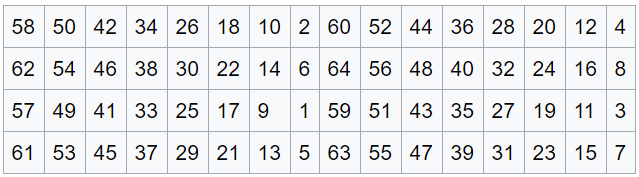


Рисунок 1 – Начальная перестановка **IP**

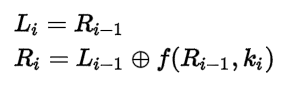
1. Picture 102Полученный после начальной перестановки 64-битовый блок IP(T) участвует в 16 циклах преобразования Фейстеля.   
   Разбить IP(T) на две части , – 32 младших и старших бита соответственно. Тогда результат i-ой операции является:  
      
    (1)
2. Функция Фестеля – Аргументами функции являются (32-битовый)  и (48-битовый) , который является результатом преобразования (56-битового) исходного ключа шифра k.  
   Для вычисления f требуется:  
    1) Функция расширения Е.  
    2) Сложение по модулю 2 с ключом .  
    3) Преобразование S, состоящее из 8 преобразований ,…,.  
    4) Перестановка   
   Функция Е расширяет (32-битовый)  до 48-битового, путём дублирования некоторых битов из . Порядок битов указан на Рис. 2.



Рисунок 2 – Функция расширения Е

Полученный после перестановок результат складывается по модулю 2 с ключом  и представляется в виде 8 последовательных блоков ,…,. Каждый  является 6-битным блоком. Далее каждый из  блоков трансформируется в 4-битовый блок  с помощью преобразований .  
Предположим, что =101111 и мы хотим найти . Первый и последний разряды  являются двоичной записью числа а, 0<=a<=3, средние 4 разряда представляют число b, 0<=b<=15. Строки таблицы S3 нумеруются от 0 до 3, столбцы таблицы S3 нумеруются от 0 до 15. Пара чисел (а, b) определяет число, находящееся в пересечении строки а и столбца b. Двоичное представление этого числа дает . В нашем случае a==3, b==3, а число, определяемое парой (3,7), равно 7. Его двоичное представление =0111.  
Значение функции Фестеля получается перестановкой P, применяемой к 32-битовому блоку ,..,.



Рисунок 3 – Перестановка Р

1. Генерация ключей  – Ключи   получаются из начального **k** ключа  (56 бит = 7 байтов или 7 символов в ASCII) следующим образом. Добавляются биты в позиции 8, 16, 24, 32, 40, 48, 56, 64 ключа **k**  таким образом, чтобы каждый байт содержал нечетное число единиц. Это используется для обнаружения ошибок при обмене и хранении ключей. Затем делают перестановку для расширенного ключа (кроме добавляемых битов 8, 16, 24, 32, 40, 48, 56, 64).

Эта перестановка определяется двумя блоками  и  по 28 бит каждый. ,  i=1,2,3…получаются из ,  одним или двумя левыми циклическими сдвигами согласно Рис. 5.



Рисунок 5

Ключ , i=1,…16 состоит из 48 бит, выбранных из битов вектора  (56 бит) согласно Рис. 6.



Рисунок 6

1. Конечная перестановка – Конечная перестановка  действует на  и является обратной к первоначальной перестановке. Конечная перестановка определяется Рис. 7.



Рисунок 7 – Конечная перестановка 

**Практическая часть**

Практическая часть данной лабораторной работы построена таким образом, чтобы пользователь имел доступ к только методам шифрования и расшифрования, таким образом данный алгоритм легко можно было комбинировать и преобразовывать, а внутренняя реализация была независимой. Также такой подход позволяет применять данный код на языке C# в любом другом языке .Net Framework, так как открытые методы не содержат нагрузки, которой может не быть в других языках, например, VB.

1. Шифрование и расшифрование исходного сообщения (рис. 8):

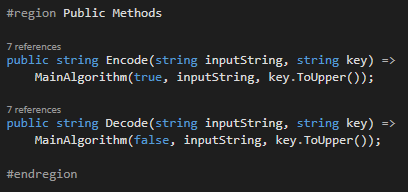


Рисунок 8

1. Главный алгоритм (рис. 9):

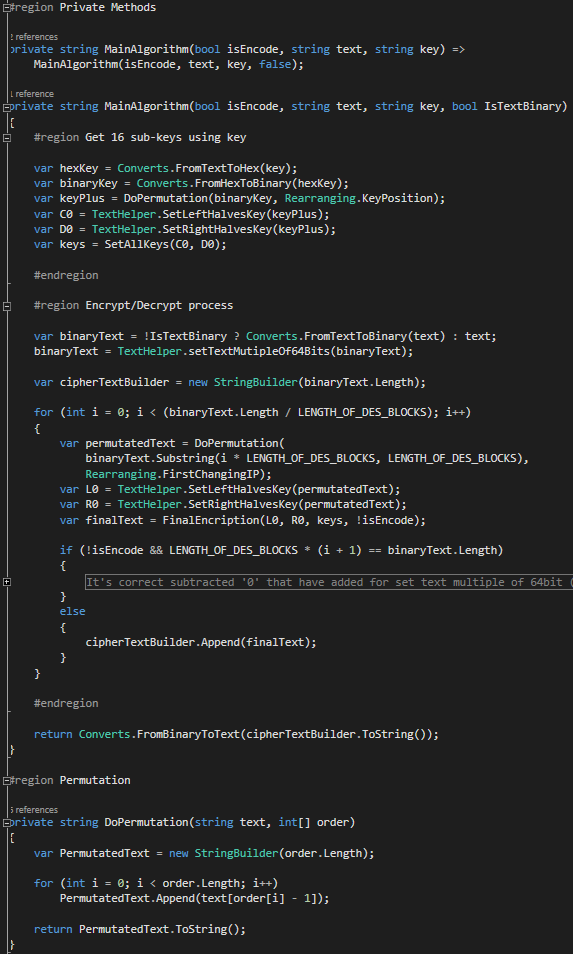


Рисунок 9

**TripleDes 1 и 2 ключа:**

1. Шифрование и расшифрование исходного сообщения. (рис. 10):

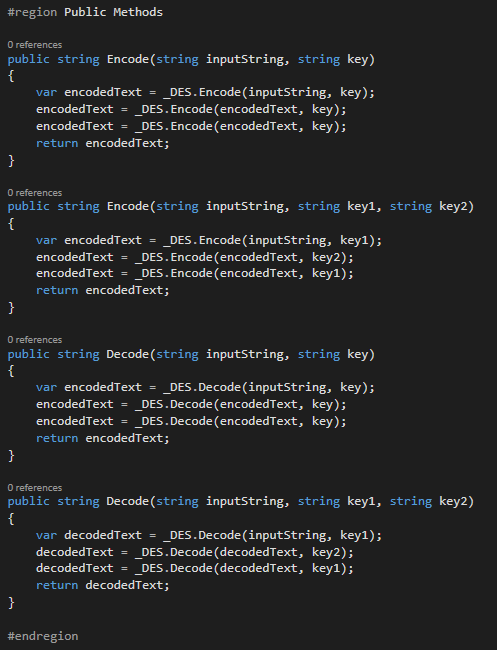


Рисунок 10

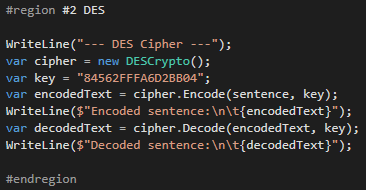
2. Пример вызова DES (рис. 12):

Рисунок 12

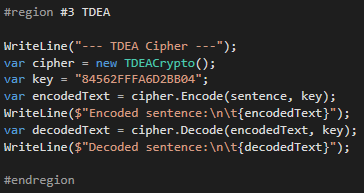
3. Пример вызова TDES (TDEA) (рис. 13):

Рисунок 13

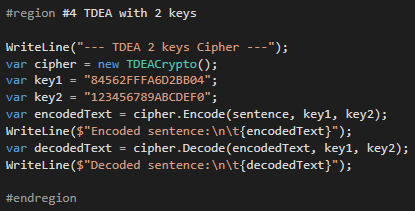
4. Пример вызова TDES с 2 ключами (рис. 14):

Рисунок 14

**Скриншоты выполнения**

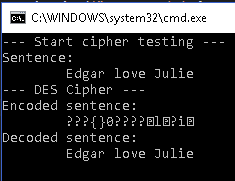
На рисунке 14 представлен результат выполнения программы DES. Остальные будут также аналогично выдавать ответ

Рисунок 14