**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационной безопасности**

отчет

**по лабораторной работе №4**

**по дисциплине «Криптография и защита информации»**

Тема: Изучение шифра DES

|  |  |  |
| --- | --- | --- |
| Студент гр. 9381 |  | Колованов Р.А. |
| Преподаватель |  | Племянников А.К. |

Санкт-Петербург

2022

**Цель работы.**

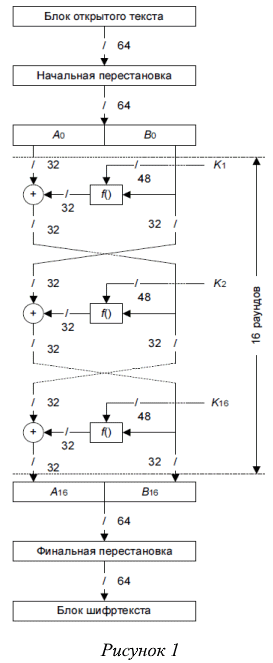
Исследовать шифры DES, 3DES, а также другие модификации шифра DES: DESX, DESL, DESXL и получить практические навыки работы с ними, в том числе с использованием приложения Cryptool 1 и 2.

**Основные теоретические положения.**

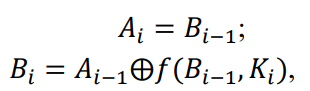
***Шифр DES.***

Стандарт шифрования данных (DES) – блочный симметричный шифр, разработанный Национальным Институтом Стандартов и Технологии (NIST – National Institute of Standards and Technology).

Шифр DES основан на сети Фейстеля. DES шифрует информацию блоками по 64 бита с помощью 64-битного ключа шифрования. Шифрование выполняется следующим образом (рис. 1):

1. Над 64-битными блоками производится начальная перестановка, задаваемая таблично;

2. После начальной перестановки блок делится на 2 субблока по 32 бита (𝐴0 и 𝐵0), над которыми производятся 16 раундов преобразований:

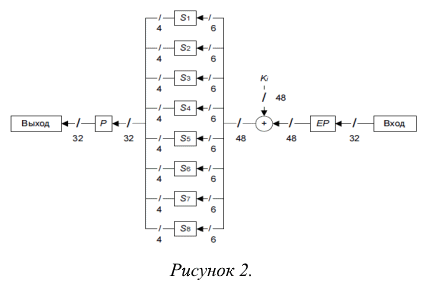


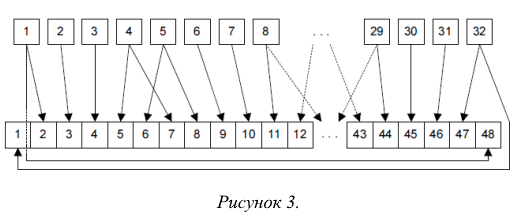
где i – номер текущего раунда, 𝐾i – ключ раунда, ⨁ – логическая операция XOR.

Схема работы функции раунда f представлена на рисунке 2. Этапы раундового преобразования:

a) Расширяющая перестановка EP, которая преобразует входные 32 бита в 48 бит (рис. 3);

b) Полученные 48 бит складываются с 𝐾i операцией XOR;





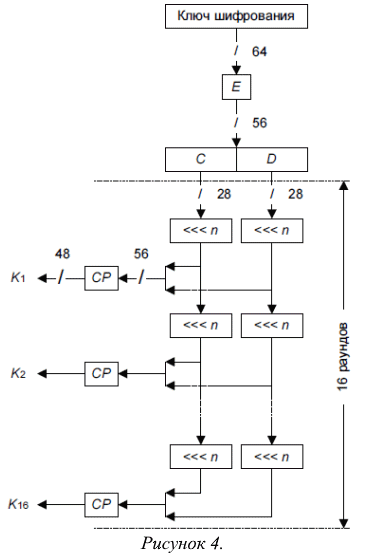
c) Результат сложения разбивается на 8 блоков по 6 битов. Каждый блок обрабатывается соответствующей таблицей замен;

d) Над полученными 32 битами, после выполнения замен, выполняется перестановка (на рисунке 2 обозначена как P);

На последнем раунде алгоритма субблоки местами не меняются.

3. Полученные в итоге субблоки 𝐴16 и 𝐵16 образуют 64-битный блок, над которым производится конечная перестановка и в итоге получается результирующий блок шифротекста.

Процедура генерации раундовых ключей представлена на рисунке 4. Из 64-битного ключа шифрования используется только 56 бит, каждый 8-й бит исключается. На рисунке 4 операция сжатия ключа и перестановка обозначена как E. После перестановки блок в 56 бит делится на два 28-битных блока (C и D). Затем выполняются 16 раундов преобразований:

1. Текущие C и D циклически сдвигаются влево на определенное количество бит;

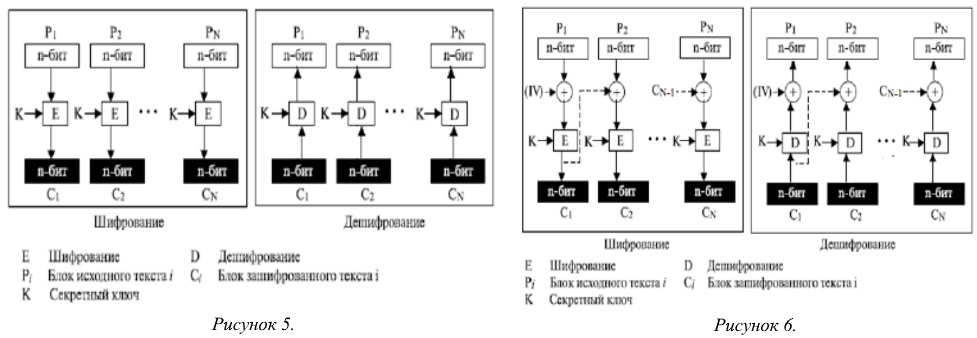
2. C и D объединяются в 56-битное значение, к которому применяется сжимающая перестановка. На выходе получаем 48-битный раундовый ключ.

Расшифровывание данных алгоритмом DES происходит при прохождении всех шагов алгоритма в обратном порядке.

***Режимы ECB и CBC шифра DES.***

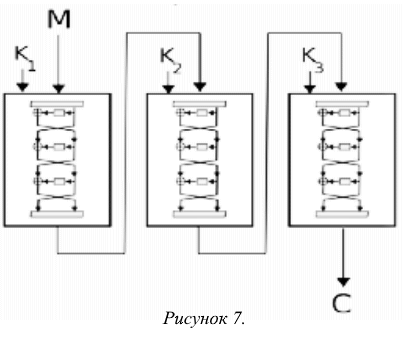
В режиме ECB шифра DES используется независимо для каждого 64- битного блока шифруемых данных. Схема использования шифра в режиме ECB представлена на рисунке 5.

В режиме CBC перед запуском DES для зашифрования каждого очередного блока открытого текста происходит побитовое XOR-сложение этого блока с блоком зашифрованного текста из предыдущего шага. Схема использования шифра в режиме CBC представлена на рисунке 6.



***Шифр 3-DES.***

Шифр 3-DES (рисунок 7) состоит в трехкратном применении обычного DES. Существует 4 основные версии данного шифра:

1. DES-EEE3 – шифрование происходит 3 раза независимыми ключами;

2. DES-EDE3 – операции шифровка-расшифровка-шифровка с тремя разными ключами;

3. DES-EEE2 – то же что и DES-EEE3, но на первом и последнем шаге одинаковый ключ;

4. DES-EDE2 – то же что и DES-EDE3, но на первом и последнем шаге используется один и тот же ключ.

На текущий момент самыми популярными версиями шифра являются DES-EDE3 и DES-EDE2.

***Модификации DESX, DESL, DESXL шифра DES.***

Алгоритм DESX использует на входе ключ длиной 184 бита, который делится на три 56-битные части. Процесс шифрования происходит по следующей схеме:



Если 𝐾1 = 𝐾2 = 0, то данный алгоритм сводится к стандартному DES.

Алгоритм DESL является облегченной версией алгоритма DES. Данный алгоритм был создан в 2006 году для RFID-меток. Алгоритм предполагает отказ от входной и выходной перестановки блока текста, т.к. они не несут криптографической сложности, а также 8 S-блоков заменяется на 1, но более стойкий чем все 8 стандартных блока DES.

Алгоритм DESXL использует те же оптимизации что и DESL, но производит шифрование по алгоритму DESX.

**Ход работы.**

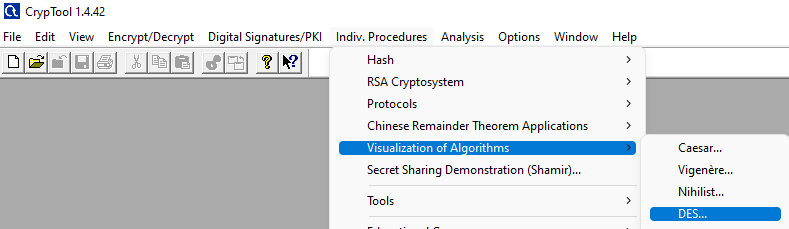
***Исследование преобразований DES.***

*Задание.*

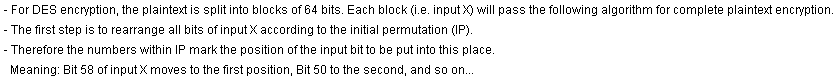
1. Изучить преобразования шифра DES с помощью демонстрационного приложения из Cryptool 1 (Indiv.Procedures -> Visualization -> DES);
2. Выполнить вручную преобразования первых двух раундов и вычисление раундовых ключей при следующих исходных данных:
   1. Открытый текст (не более 64 бит) – фамилия\_имя (транслитерация латиницей);
   2. Ключ (56 бит) – номер зачетной книжки и инициал отчества (всего 7 символов);
3. Выполнить вручную обратное преобразование зашифрованного сообщения;
4. Убедиться в совпадении результатов.

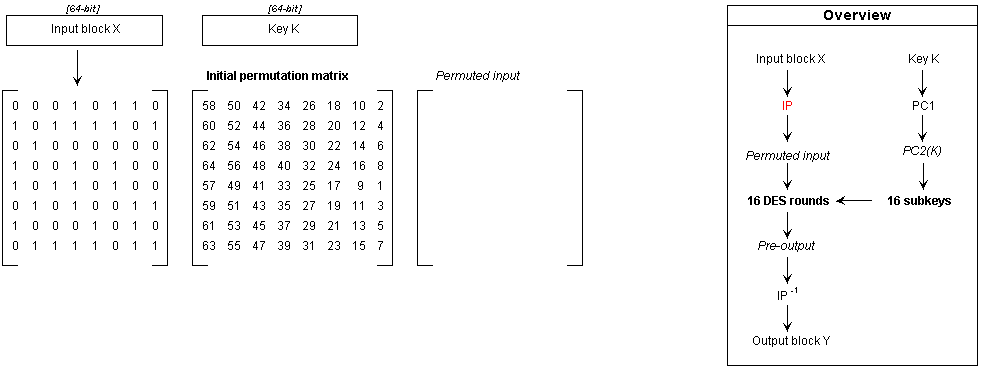
*Изучение преобразования шифра DES.*

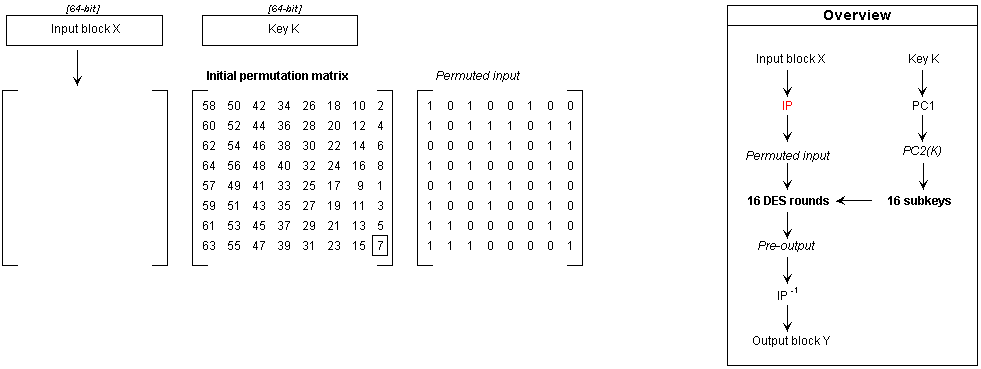
При помощи приложения CrypTool 1 (Indiv.Procedures -> Visualization -> DES) изучим процесс преобразования шифра DES:



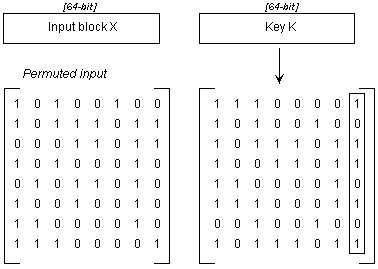
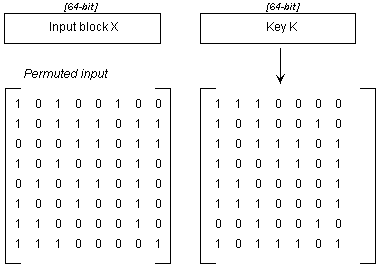
В начале выполняется первый этап, а именно начальная перестановка IP 64-битного блока, задаваемая таблично:

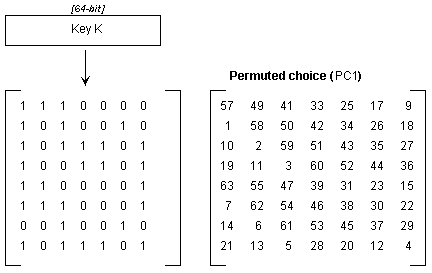
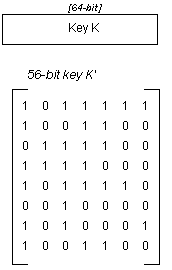


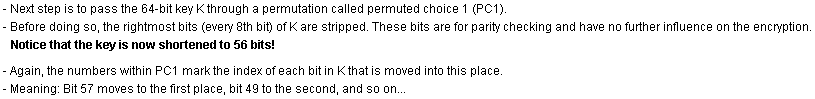




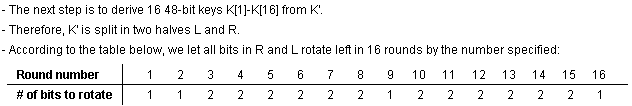
Далее производится генерация раундовых ключей. Для начала 64-битный ключ преобразуется к 56-битному ключу при помощи удаления каждого 8 бита ключа, после чего производится перестановка (PC1):

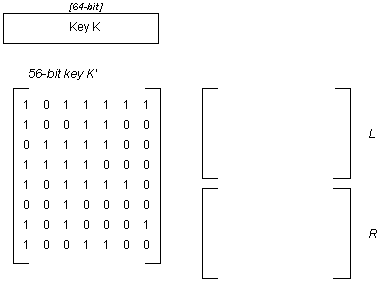
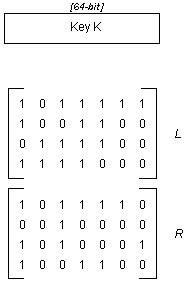
 

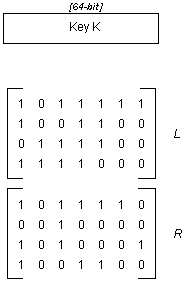
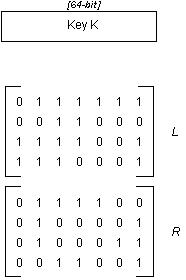
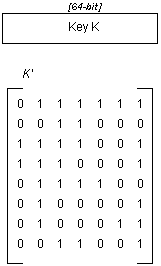


Далее производится генерация 16 раундовых 48-битных ключей из полученного 56-битного ключа K’. Для этого K’ делится на две части, левую (L) и правую (R):

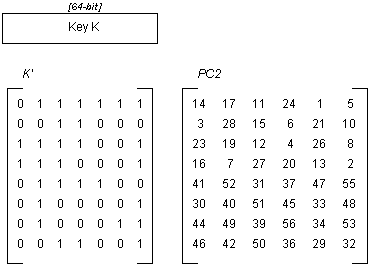
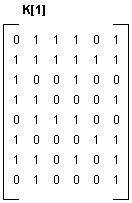


Далее для левой и правой части производится побитовый сдвиг на определенное количество бит, в зависимости от порядкового номера ключа. Например, для первого раундового ключа сдвиг будет следующий:

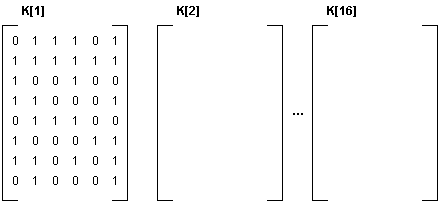
  

После выполнения сдвига выполняется сжимающая перестановка PC2. Для первого раундового ключа она будет выглядеть следующим образом:

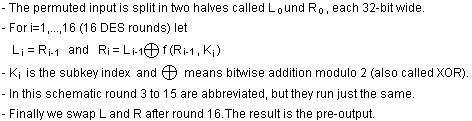
 

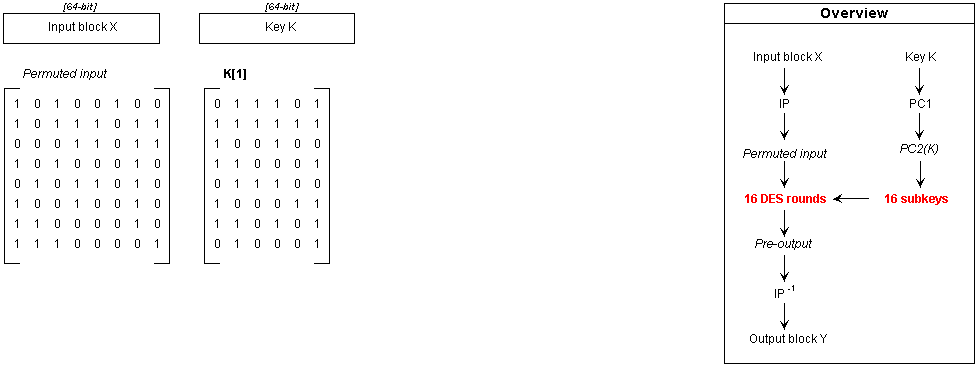
На выходе получаем итоговый 48-битный первый раундовый ключ. Аналогичным образом генерируются остальные 15 раундовых ключей:



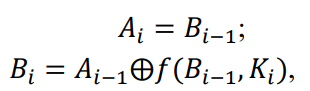


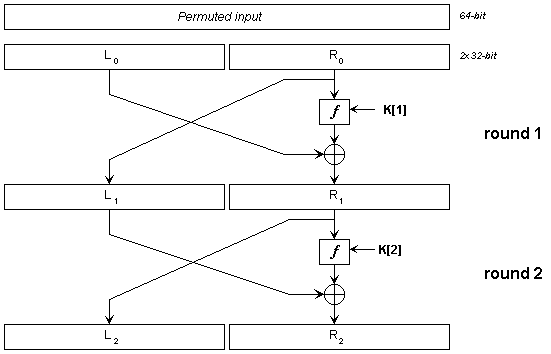
Все необходимые входные данные для основного алгоритма вычислены, теперь можно переходить к шифровке. Шифровка производится в 16 этапов, на каждом используется соответствующий ему раундовый ключ. В каждом раунде производится шифровка определенного блока, полученного из предыдущего раунда, соответствующим ключом. Данный процесс называется сетью Фейстеля. Далее рассмотрим преобразования, которые происходят в раундах:



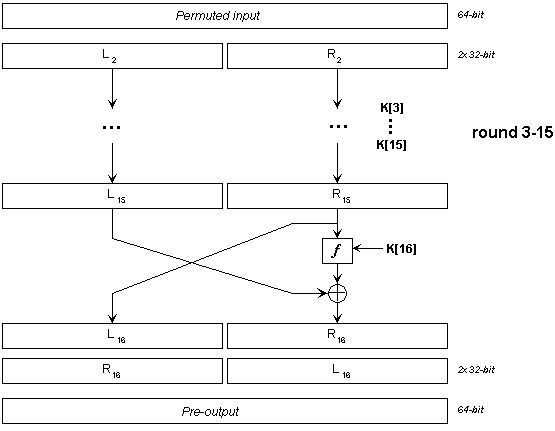


Для начала входные данные делятся на две части по 32 бита, левую (L0) и правую (R0). На каждом раунде выполняются следующие действия:



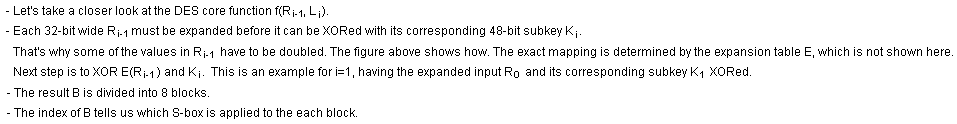


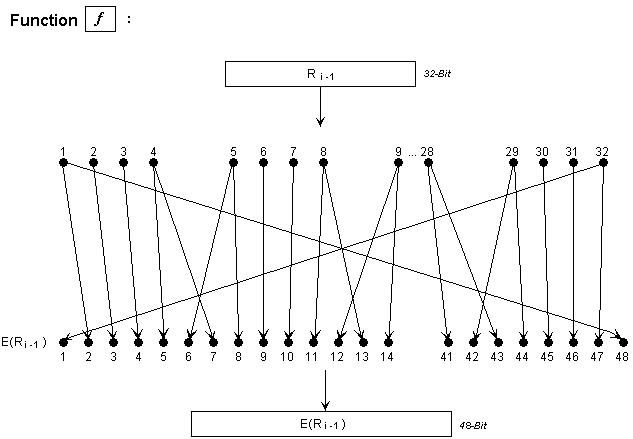
После выполнения 16 раундов на выходе получаем два блока L16 и R16 по 32 бита, которые в завершение меняются местами:



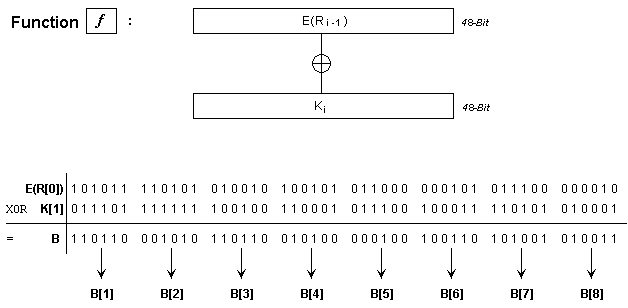
На выходе получаем Pre-Output.

Теперь подробнее рассмотрим функцию f шифра DES, которая используется в каждом раунде преобразований. В начале производится расширяющая перестановка EP, которая преобразует 32-битный блок в 48-битный блок:



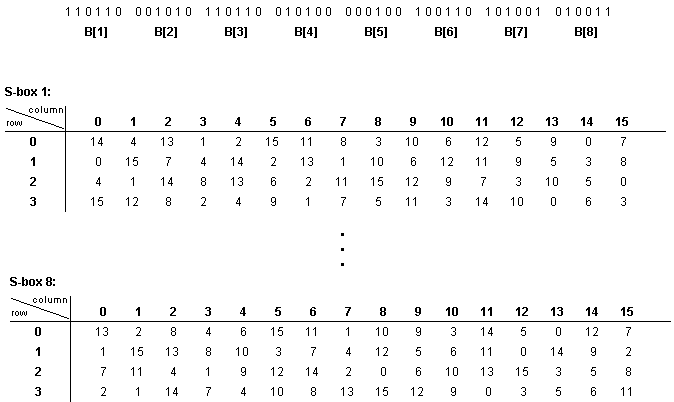


Полученный 48-битный блок складывается по модулю 2 с раундовым ключом, после чего делится на 8 блоков по 6 бит каждый:



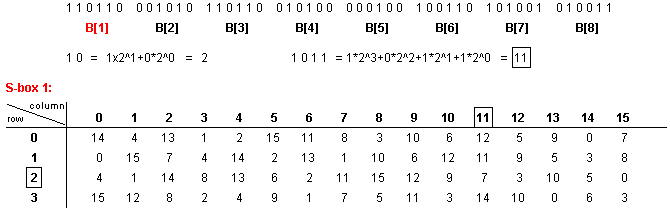
Далее каждый блок B[i] обрабатывается соответствующей ему таблицей замены S[i] (4 строки на 16 столбцов). Первый и последний бит и остальные 4 бита блока B[i] используется для определения номера строки и номера столбца значения замены в таблице S[i]:

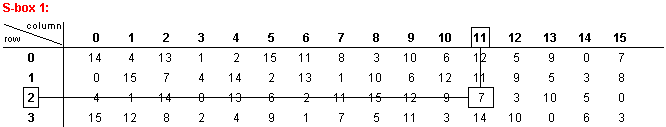






Пример выполнения замены блока B[1]:





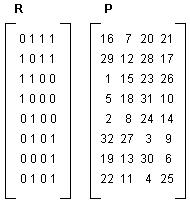
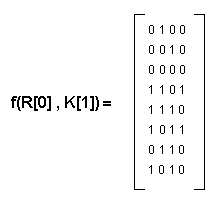
7 = 01112

В итоге для B[1] заменой будет является значение 01112. Таким образом 6-битные блоки B[i] заменяются на 4-битные блоки:

**

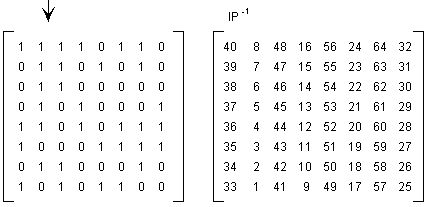
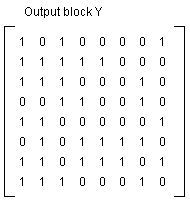
Далее объединяем полученные 4-битные блоки в 32-битный блок R, для которого осуществляем финальную перестановку P:



В итоге получаем результат преобразования функцией f блока R[0] с раундовым ключом R[1]. Таким образом происходит вычисление функции f в раундах.

Для полученного после всех раундов PreOutput в конце выполняется обратная начальная перестановка IP-1.

В итоге получаем зашифрованный блок Y. На этом шифровка входного блока X завершена.

*Ручное преобразование первых двух раундов и раундовых ключей.*

Теперь выполним ручное преобразование первых двух раундов шифра DES и вычисление раундовых ключей для следующих исходных данных:

А. Входные данные – текст «KOLOVANO»;

Б. Ключ – текст «938106A».

Для начала преобразуем исходные данные к бинарному виду:

|  |  |  |
| --- | --- | --- |
| Буква | Код ASCII | Двоичный код |
| K | 75 | 01001011 |
| O | 79 | 01001111 |
| L | 76 | 01001100 |
| O | 79 | 01001111 |
| V | 86 | 01010110 |
| A | 65 | 01000001 |
| N | 78 | 01001110 |
| O | 79 | 01001111 |

|  |  |  |
| --- | --- | --- |
| Буква | Код ASCII | Двоичный код |
| 9 | 57 | 00111001 |
| 3 | 51 | 00110011 |
| 8 | 56 | 00111000 |
| 1 | 49 | 00110001 |
| 0 | 48 | 00110000 |
| 6 | 54 | 00110110 |
| A | 65 | 01000001 |

Получаем входной 64-битный блок для шифрования и 56-битный ключ.

Теперь найдем первые два раундовых ключа. Для начала выполним перестановку PC1 для 56-битного ключа, после чего поделим его на верхнюю C0 и нижнюю D0 части по 28 бит:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 57 | 49 | 41 | 33 | 25 | 17 | 9 |
| 1 | 58 | 50 | 42 | 34 | 26 | 18 |
| 10 | 2 | 59 | 51 | 43 | 35 | 27 |
| 19 | 11 | 3 | 60 | 52 | 44 | 36 |
| 63 | 55 | 47 | 39 | 31 | 23 | 15 |
| 7 | 62 | 54 | 46 | 38 | 30 | 22 |
| 14 | 6 | 61 | 53 | 45 | 37 | 29 |
| 21 | 13 | 5 | 28 | 20 | 12 | 4 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 |

Ключ (56 бит) Перестановка (PC1) Результат

Получаем C0 и D0 соответственно:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 |

C0 D0

Исходя из таблицы побитовых сдвигов выполняем побитовый сдвиг для C0 и D0, C1 и D1 на 1 бит влево, чтобы получить блоки C и D для первого и второго раундового ключа:



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |

C1 D1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 |

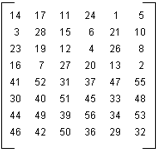
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 |

C2 D2

В завершение склеиваем блоки C1 и D1, C2 и D2, после чего применяем перестановку PC2 и получаем два итоговых раундовых ключа:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |

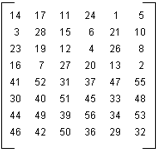
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |



К’[1] Перестановка (PC2) K[1]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 |



К’[2] Перестановка (PC2) K[2]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |

K[1] =

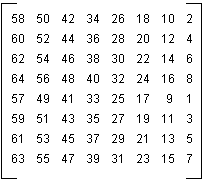
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 |

K[2] =

Первые два раундовых ключа найдены. Теперь можно приступить к шифрованию. Для начала выполним начальную перестановку IP входных данных:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |



Входные данные Перестановка (IP) Результат

Далее разделим блок на две 32-битные части L0 и R0:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

L0 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

R0 =

Далее выполним первый раунд преобразований:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

L1 = R0 =

R1 = f(R0, E[1]) XOR L0

Найдем f(R0, E[1]). Для начала выполним расширяющую перестановка EP для блока R0, которая преобразует 32-битный блок в 48-битный блок:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

R0

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 32 | 1 | 2 | 3 | 4 | 5 |
| 4 | 5 | 6 | 7 | 8 | 9 |
| 8 | 9 | 10 | 11 | 12 | 13 |
| 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 |
| 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 |
| 28 | 29 | 30 | 31 | 32 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |

Расширяющая перестановка (EP) E(R0)

Далее сложим E(R0) с K[1] по модулю 2 и разделим результат на 8 6-битных блоков B1[1], … , B1[8]:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

E(R0) K[1] Результат

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **B1[1]** | 1 | 0 | 0 | 0 | 1 | 1 |
| **B1[2]** | 0 | 1 | 0 | 1 | 0 | 0 |
| **B1[3]** | 1 | 0 | 0 | 0 | 0 | 0 |
| **B1[4]** | 0 | 0 | 1 | 0 | 1 | 0 |
| **B1[5]** | 1 | 1 | 0 | 0 | 0 | 0 |
| **B1[6]** | 0 | 0 | 0 | 0 | 0 | 1 |
| **B1[7]** | 1 | 0 | 1 | 1 | 1 | 1 |
| **B1[8]** | 1 | 1 | 1 | 1 | 1 | 1 |

Теперь каждый 6-битный блок B1[i] обработаем соответствующей таблицей замен S[i], после чего склеим полученные 8 4-битных блоков в 32-битный блок R1:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[1]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 14 | 4 | 13 | 1 | 2 | 15 | 11 | 8 | 3 | 10 | 6 | 12 | 5 | 9 | 0 | 7 |
| **1** | 0 | 15 | 7 | 4 | 14 | 2 | 13 | 1 | 10 | 6 | 12 | 11 | 9 | 5 | 3 | 8 |
| **2** | 4 | 1 | 14 | 8 | 13 | 6 | 2 | 11 | 15 | 12 | 9 | 7 | 3 | 10 | 5 | 0 |
| **3** | 15 | 12 | 8 | 2 | 4 | 9 | 1 | 7 | 5 | 11 | 3 | 14 | 10 | 0 | 6 | 13 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[2]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 15 | 1 | 8 | 14 | 6 | 11 | 3 | 4 | 9 | 7 | 2 | 13 | 12 | 0 | 5 | 10 |
| **1** | 3 | 13 | 4 | 7 | 15 | 2 | 8 | 14 | 12 | 0 | 1 | 10 | 6 | 9 | 11 | 5 |
| **2** | 0 | 14 | 7 | 11 | 10 | 4 | 13 | 1 | 5 | 8 | 12 | 6 | 9 | 3 | 2 | 15 |
| **3** | 13 | 8 | 10 | 1 | 3 | 15 | 4 | 2 | 11 | 6 | 7 | 12 | 0 | 5 | 14 | 9 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[3]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 10 | 0 | 9 | 14 | 6 | 3 | 15 | 5 | 1 | 13 | 12 | 7 | 11 | 4 | 2 | 8 |
| **1** | 13 | 7 | 0 | 9 | 3 | 4 | 6 | 10 | 2 | 8 | 5 | 14 | 12 | 11 | 15 | 1 |
| **2** | 13 | 6 | 4 | 9 | 8 | 15 | 3 | 0 | 11 | 1 | 2 | 12 | 5 | 10 | 14 | 7 |
| **3** | 1 | 10 | 13 | 0 | 6 | 9 | 8 | 7 | 4 | 15 | 14 | 3 | 11 | 5 | 2 | 12 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[4]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 7 | 13 | 14 | 3 | 0 | 6 | 9 | 10 | 1 | 2 | 8 | 5 | 11 | 12 | 4 | 15 |
| **1** | 13 | 8 | 11 | 5 | 6 | 15 | 0 | 3 | 4 | 7 | 2 | 12 | 1 | 10 | 14 | 9 |
| **2** | 10 | 6 | 9 | 0 | 12 | 11 | 7 | 13 | 15 | 1 | 3 | 14 | 5 | 2 | 8 | 4 |
| **3** | 3 | 15 | 0 | 6 | 10 | 1 | 13 | 8 | 9 | 4 | 5 | 11 | 12 | 7 | 2 | 14 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[5]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 2 | 12 | 4 | 1 | 7 | 10 | 11 | 6 | 8 | 5 | 3 | 15 | 13 | 0 | 14 | 9 |
| **1** | 14 | 11 | 2 | 12 | 4 | 7 | 13 | 1 | 5 | 0 | 15 | 10 | 3 | 9 | 8 | 6 |
| **2** | 4 | 2 | 1 | 11 | 10 | 13 | 7 | 8 | 15 | 9 | 12 | 5 | 6 | 3 | 0 | 14 |
| **3** | 11 | 8 | 12 | 7 | 1 | 14 | 2 | 13 | 6 | 15 | 0 | 9 | 10 | 4 | 5 | 3 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[6]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 12 | 1 | 10 | 15 | 9 | 2 | 6 | 8 | 0 | 13 | 3 | 4 | 14 | 7 | 5 | 11 |
| **1** | 10 | 15 | 4 | 2 | 7 | 12 | 9 | 5 | 6 | 1 | 13 | 14 | 0 | 11 | 3 | 8 |
| **2** | 9 | 14 | 15 | 5 | 2 | 8 | 12 | 3 | 7 | 0 | 4 | 10 | 1 | 13 | 11 | 6 |
| **3** | 4 | 3 | 2 | 12 | 9 | 5 | 15 | 10 | 11 | 14 | 1 | 7 | 6 | 0 | 8 | 13 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[7]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 4 | 11 | 2 | 14 | 15 | 0 | 8 | 13 | 3 | 12 | 9 | 7 | 5 | 10 | 6 | 1 |
| **1** | 13 | 0 | 11 | 7 | 4 | 9 | 1 | 10 | 14 | 3 | 5 | 12 | 2 | 15 | 8 | 6 |
| **2** | 1 | 4 | 11 | 13 | 12 | 3 | 7 | 14 | 10 | 15 | 6 | 8 | 0 | 5 | 9 | 2 |
| **3** | 6 | 11 | 13 | 8 | 1 | 4 | 10 | 7 | 9 | 5 | 0 | 15 | 14 | 2 | 3 | 12 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[8]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 13 | 2 | 8 | 4 | 6 | 15 | 11 | 1 | 10 | 9 | 3 | 14 | 5 | 0 | 12 | 7 |
| **1** | 1 | 15 | 13 | 8 | 10 | 3 | 7 | 4 | 12 | 5 | 6 | 11 | 0 | 14 | 9 | 2 |
| **2** | 7 | 11 | 4 | 1 | 9 | 12 | 14 | 2 | 0 | 6 | 10 | 13 | 15 | 3 | 5 | 8 |
| **3** | 2 | 1 | 14 | 7 | 4 | 10 | 8 | 13 | 15 | 12 | 9 | 0 | 3 | 5 | 6 | 11 |

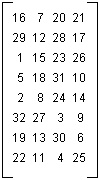
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Блок | Значение блока | Номер строки | Номер столбца | Замена |
| B1[1] | 100011 | 11 = 3 | 0001 = 1 | 12 = 1100 |
| B1[2] | 010100 | 00 = 0 | 1010 = 10 | 2 = 0010 |
| B1[3] | 100000 | 10 = 2 | 0000 = 0 | 13 = 1101 |
| B1[4] | 001010 | 00 = 0 | 0101 = 5 | 6 = 0110 |
| B1[5] | 110000 | 10 = 2 | 1000 = 8 | 15 = 1111 |
| B1[6] | 000001 | 01 = 1 | 0000 = 0 | 10 = 1010 |
| B1[7] | 101111 | 11 = 3 | 0111 = 7 | 7 = 0111 |
| B1[8] | 111111 | 11 = 3 | 1111 = 15 | 11 = 1011 |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |

R1 =

В конце для R1 выполним перестановку P:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 |



Перестановка P Результат

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 |

f(R0, E[1]) =

Теперь сложим f(R0, E[1]) с L0 по модулю 2:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

L0 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

R1 = f(R0, E[1]) XOR L0 =

Итого после первого раунда получаем:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

L1 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

R1 =

Теперь выполним второй раунд преобразований:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

L2 = R1 =

R2 = f(R1, E[2]) XOR L1

Найдем f(R1, E[2]). Для начала выполним расширяющую перестановка EP для блока R1, которая преобразует 32-битный блок в 48-битный блок:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

R1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 32 | 1 | 2 | 3 | 4 | 5 |
| 4 | 5 | 6 | 7 | 8 | 9 |
| 8 | 9 | 10 | 11 | 12 | 13 |
| 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 |
| 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 |
| 28 | 29 | 30 | 31 | 32 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 |

Расширяющая перестановка (EP) E(R1)

Далее сложим E(R1) с K[2] по модулю 2 и разделим результат на 8 6-битных блоков B2[1], … , B2[8]:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |

E(R1) K[2] Результат

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **B2[1]** | 1 | 1 | 0 | 1 | 1 | 1 |
| **B2[2]** | 0 | 1 | 0 | 0 | 0 | 1 |
| **B2[3]** | 0 | 1 | 0 | 0 | 1 | 1 |
| **B2[4]** | 0 | 0 | 0 | 1 | 1 | 0 |
| **B2[5]** | 1 | 0 | 0 | 0 | 0 | 1 |
| **B2[6]** | 1 | 1 | 0 | 1 | 1 | 1 |
| **B2[7]** | 0 | 1 | 1 | 1 | 0 | 0 |
| **B2[8]** | 0 | 1 | 1 | 1 | 1 | 0 |

Теперь каждый 6-битный блок B2[i] обработаем соответствующей таблицей замен S[i], после чего склеим полученные 8 4-битных блоков в 32-битный блок R2:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[1]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 14 | 4 | 13 | 1 | 2 | 15 | 11 | 8 | 3 | 10 | 6 | 12 | 5 | 9 | 0 | 7 |
| **1** | 0 | 15 | 7 | 4 | 14 | 2 | 13 | 1 | 10 | 6 | 12 | 11 | 9 | 5 | 3 | 8 |
| **2** | 4 | 1 | 14 | 8 | 13 | 6 | 2 | 11 | 15 | 12 | 9 | 7 | 3 | 10 | 5 | 0 |
| **3** | 15 | 12 | 8 | 2 | 4 | 9 | 1 | 7 | 5 | 11 | 3 | 14 | 10 | 0 | 6 | 13 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[2]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 15 | 1 | 8 | 14 | 6 | 11 | 3 | 4 | 9 | 7 | 2 | 13 | 12 | 0 | 5 | 10 |
| **1** | 3 | 13 | 4 | 7 | 15 | 2 | 8 | 14 | 12 | 0 | 1 | 10 | 6 | 9 | 11 | 5 |
| **2** | 0 | 14 | 7 | 11 | 10 | 4 | 13 | 1 | 5 | 8 | 12 | 6 | 9 | 3 | 2 | 15 |
| **3** | 13 | 8 | 10 | 1 | 3 | 15 | 4 | 2 | 11 | 6 | 7 | 12 | 0 | 5 | 14 | 9 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[3]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 10 | 0 | 9 | 14 | 6 | 3 | 15 | 5 | 1 | 13 | 12 | 7 | 11 | 4 | 2 | 8 |
| **1** | 13 | 7 | 0 | 9 | 3 | 4 | 6 | 10 | 2 | 8 | 5 | 14 | 12 | 11 | 15 | 1 |
| **2** | 13 | 6 | 4 | 9 | 8 | 15 | 3 | 0 | 11 | 1 | 2 | 12 | 5 | 10 | 14 | 7 |
| **3** | 1 | 10 | 13 | 0 | 6 | 9 | 8 | 7 | 4 | 15 | 14 | 3 | 11 | 5 | 2 | 12 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[4]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 7 | 13 | 14 | 3 | 0 | 6 | 9 | 10 | 1 | 2 | 8 | 5 | 11 | 12 | 4 | 15 |
| **1** | 13 | 8 | 11 | 5 | 6 | 15 | 0 | 3 | 4 | 7 | 2 | 12 | 1 | 10 | 14 | 9 |
| **2** | 10 | 6 | 9 | 0 | 12 | 11 | 7 | 13 | 15 | 1 | 3 | 14 | 5 | 2 | 8 | 4 |
| **3** | 3 | 15 | 0 | 6 | 10 | 1 | 13 | 8 | 9 | 4 | 5 | 11 | 12 | 7 | 2 | 14 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[5]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 2 | 12 | 4 | 1 | 7 | 10 | 11 | 6 | 8 | 5 | 3 | 15 | 13 | 0 | 14 | 9 |
| **1** | 14 | 11 | 2 | 12 | 4 | 7 | 13 | 1 | 5 | 0 | 15 | 10 | 3 | 9 | 8 | 6 |
| **2** | 4 | 2 | 1 | 11 | 10 | 13 | 7 | 8 | 15 | 9 | 12 | 5 | 6 | 3 | 0 | 14 |
| **3** | 11 | 8 | 12 | 7 | 1 | 14 | 2 | 13 | 6 | 15 | 0 | 9 | 10 | 4 | 5 | 3 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[6]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 12 | 1 | 10 | 15 | 9 | 2 | 6 | 8 | 0 | 13 | 3 | 4 | 14 | 7 | 5 | 11 |
| **1** | 10 | 15 | 4 | 2 | 7 | 12 | 9 | 5 | 6 | 1 | 13 | 14 | 0 | 11 | 3 | 8 |
| **2** | 9 | 14 | 15 | 5 | 2 | 8 | 12 | 3 | 7 | 0 | 4 | 10 | 1 | 13 | 11 | 6 |
| **3** | 4 | 3 | 2 | 12 | 9 | 5 | 15 | 10 | 11 | 14 | 1 | 7 | 6 | 0 | 8 | 13 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[7]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 4 | 11 | 2 | 14 | 15 | 0 | 8 | 13 | 3 | 12 | 9 | 7 | 5 | 10 | 6 | 1 |
| **1** | 13 | 0 | 11 | 7 | 4 | 9 | 1 | 10 | 14 | 3 | 5 | 12 | 2 | 15 | 8 | 6 |
| **2** | 1 | 4 | 11 | 13 | 12 | 3 | 7 | 14 | 10 | 15 | 6 | 8 | 0 | 5 | 9 | 2 |
| **3** | 6 | 11 | 13 | 8 | 1 | 4 | 10 | 7 | 9 | 5 | 0 | 15 | 14 | 2 | 3 | 12 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S[8]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 13 | 2 | 8 | 4 | 6 | 15 | 11 | 1 | 10 | 9 | 3 | 14 | 5 | 0 | 12 | 7 |
| **1** | 1 | 15 | 13 | 8 | 10 | 3 | 7 | 4 | 12 | 5 | 6 | 11 | 0 | 14 | 9 | 2 |
| **2** | 7 | 11 | 4 | 1 | 9 | 12 | 14 | 2 | 0 | 6 | 10 | 13 | 15 | 3 | 5 | 8 |
| **3** | 2 | 1 | 14 | 7 | 4 | 10 | 8 | 13 | 15 | 12 | 9 | 0 | 3 | 5 | 6 | 11 |

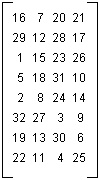
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Блок | Значение блока | Номер строки | Номер столбца | Замена |
| B2[1] | 110111 | 11 = 3 | 1011 = 11 | 14 = 1110 |
| B2[2] | 010001 | 01 = 1 | 1000 = 8 | 12 = 1100 |
| B2[3] | 010011 | 01 = 1 | 1001 = 9 | 8 = 1000 |
| B2[4] | 000110 | 00 = 0 | 0011 = 3 | 3 = 0011 |
| B2[5] | 100001 | 11 = 3 | 0000 = 0 | 11 = 1011 |
| B2[6] | 110111 | 11 = 3 | 1011 = 11 | 7 = 0111 |
| B2[7] | 011100 | 00 = 0 | 1110 = 14 | 6 = 0110 |
| B2[8] | 011110 | 00 = 0 | 1111 = 15 | 7 = 0111 |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 |

R2 =

В конце для R2 выполним перестановку P:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 |



Перестановка P Результат

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 |

f(R1, E[2]) =

Теперь сложим f(R1, E[2]) с L1 по модулю 2:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

L1 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

R2 = f(R1, E[2]) XOR L1 =

Итого после первого раунда получаем:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

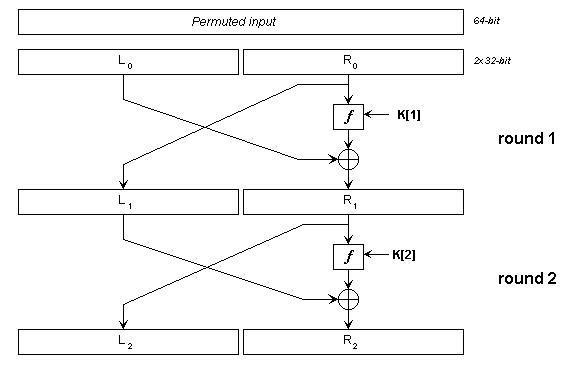
L2 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

R2 =

*Ручное обратное преобразование шифротекста.*

Теперь выполним обратное преобразование L2 и R2 к исходным данным. Поскольку нам известен R1 (он равен L2) и R2, мы можем найти f(R1, E[2]), после чего зная f(R1, E[2]) и R2 найти L1. Аналогичный принцип можно применить для нахождения L0 и R0.



Найдем L1 и R1. Из предыдущих пунктов:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

R2 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

L2 = R1 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |

f(L2, E[2]) =

Отсюда находим L1:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

L1 =

Найдем L0 и R0. Из предыдущих пунктов:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

R1 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

L1 = R0 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

f(L1, E[1]) =

Отсюда находим L0:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

L0 =

Осталось объединить блоки L0 и R0 в один 64-битный блок и выполнить обратную начальную перестановку IP-1:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |



L0 и R0 Перестановка (IP-1) Результат

Как видно из результата, расшифрованные данные совпадают с начальными данными. Если перевести результат в текст, то получим сообщение «KOLOVANO».

***Исследование DES в режимах ECB и CBC.***

*Задание.*

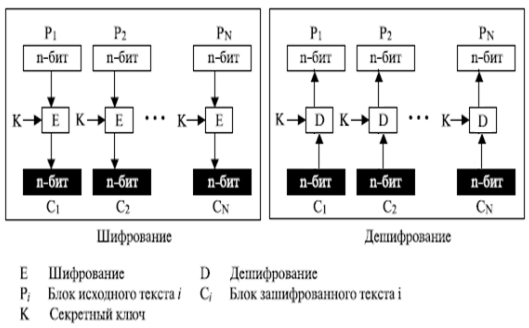
1. Создать картинку со своими ФИО (формат bmp);
2. Зашифровать картинку шифром DES в режиме ECB;
3. Зашифровать картинку шифром DES в режиме CBC c тем же ключом;
4. Сохранить скриншоты картинок для отчета;
5. Сжать исходную и 2 зашифрованных картинки средствами CrypTool. Зафиксировать размеры полученных файлов в таблице;
6. Выбрать случайный текст на английском языке (не менее 1000 знаков) и зашифровать его DES в режиме ECB;
7. Для одного и того же шифротекста оценить время проведения атаки «грубой силы» в случаях, когда известно n-4, n-6, n-8,.., 2 байт секретного ключа. Зафиксировать результаты измерений в таблице;
8. Повторить подобные измерения для DES в режиме CBC.

*Основные параметры и обобщенная схема шифров.*

Существуют следующие режимы работы блочных шифров:

* Электронная кодовая книга (ECB);
* Сцепление шифрованных блоков (CBC);
* Режим обратной связи по шифру (CFB);
* Режим внешней обратной связи (OFB);
* Счетчик (CTR).

Структура режима ECB:



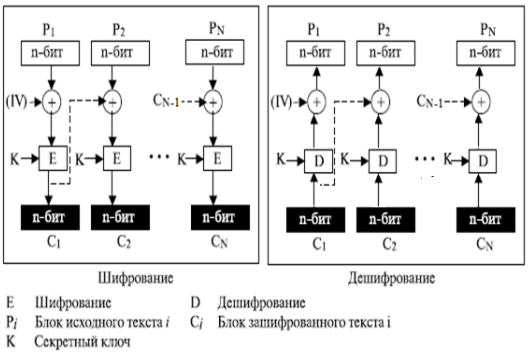
Достоинства режима ECB:

* Шифрование может быть параллельным;
* Ошибка в передаче блока не имеет никакого влияния на другие блоки.

Недостатки режима ECB:

* Одинаковые блоки открытого текста будут преобразовываться в одинаковые блоки шифротекста;
* Независимость блоков создает возможность для замены некоторых блоков зашифрованного текста без знания ключа.

Структура режима CBC:



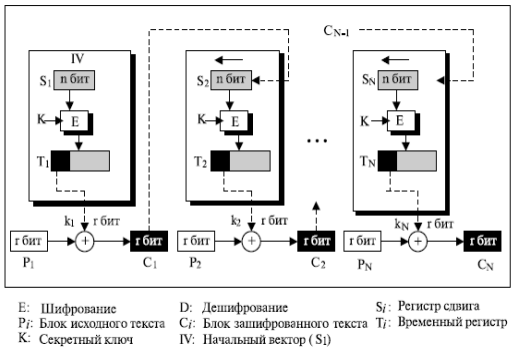
Достоинства режима CBC:

* Одинаковые блоки открытого текста будут преобразовываться в различные блоки шифротекста;
* Если при передаче произойдет изменение одного бита шифротекста, то данная ошибка распространяется на следующие блоки, но при расшифровке произойдет самовосстановление;
* Последний блок шифротекста зависит от всех бит открытого текста сообщения и может использоваться для контроля целостности сообщения.

Недостатки режима CBC:

* Шифрование сообщения не поддается распараллеливанию.

Структура режима CFB:



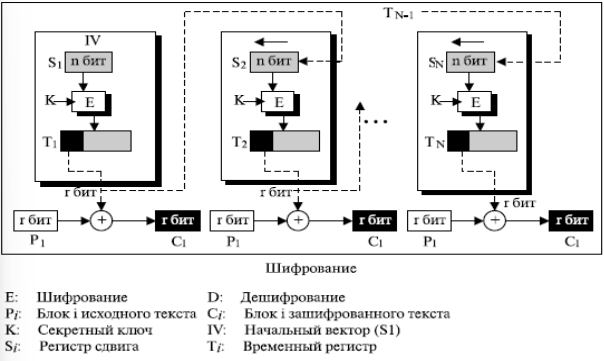
Достоинства режима CFB:

* Это шифр потока, в котором ключевой поток зависит от зашифрованного текста;
* В этом режиме не требуется дополнение блоков.

Недостатки режима CFB:

* Ошибка в единственном бите шифротекста создает ошибку в следующих блоках до тех пор, пока ошибка находится в регистре сдвига.

Структура режима OFB:



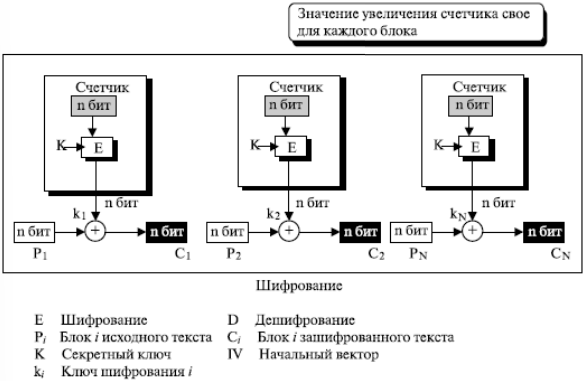
Достоинства режима OFB:

* Фактически, это шифр потока, в котором ключевой поток не зависит от исходного и зашифрованного текста;
* Каждый бит в зашифрованном тексте независим от предыдущего бита или битов. Это позволяет избежать распространения ошибок.

Недостатки режима OFB:

* Чтобы одним и тем же ключом зашифровать больше, чем одно сообщение, значение IV должно быть изменено для каждого сообщения.

Структура режима CTR:

**

Достоинства режима CTR:

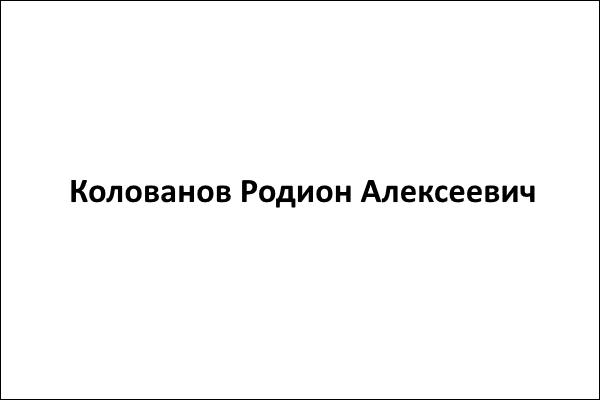
* Создает n-битовый зашифрованный текст, блоки которого независимы друг от друга – они зависят только от значений счетчика. Фактически, это шифр потока;
* Режим, подобно режиму ECB, может использоваться, чтобы зашифровывать и расшифровывать файлы произвольного доступа, и значение счетчика может быть связано номером записи в файле.

Недостатки режима CTR:

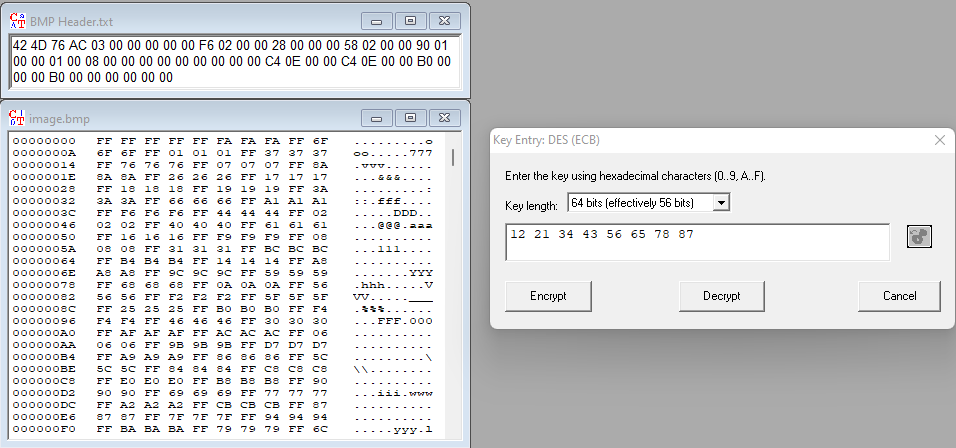
* Если значения счетчиков совпадает, то шифрование производится на одном ключе.

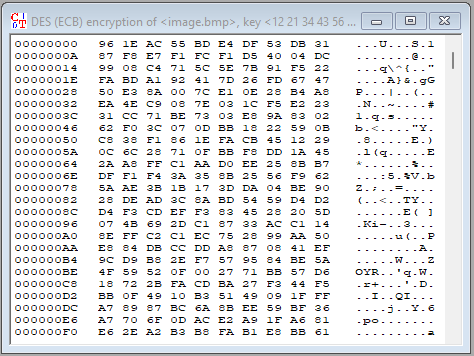
*Шифрование картинки шифром DES в режимах ECB и CBC.*

Для начала была создана исходная картинка с текстом «Колованов Родион Алексеевич» на белом фоне:

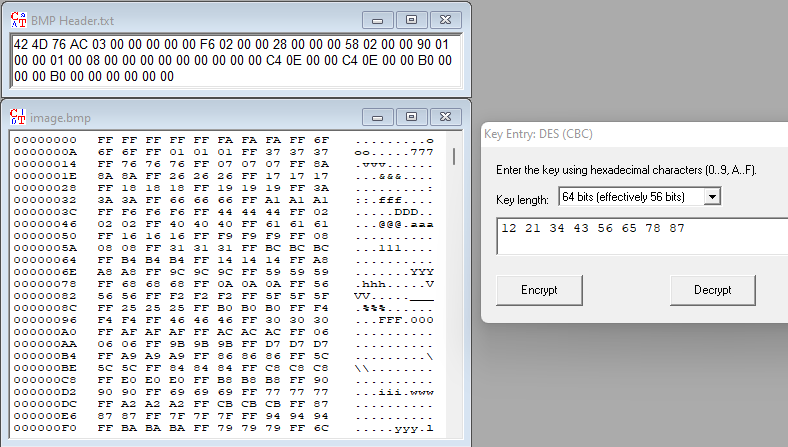


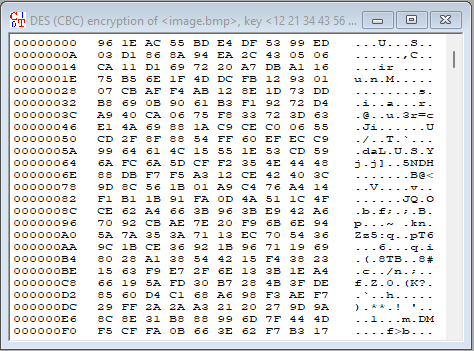
Далее исходная картинка (без заголовка BMP) была зашифрована при помощи DES с режимом ECB, используя ключ «12 21 34 43 56 65 78 87»:





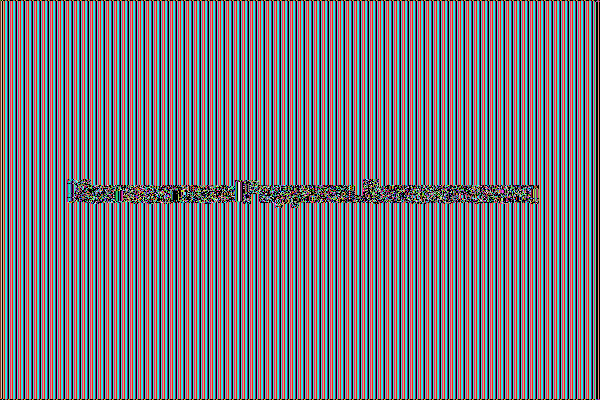
Далее исходная картинка (без заголовка BMP) была зашифрована при помощи DES с режимом CBC, используя ключ «12 21 34 43 56 65 78 87»:





Далее к полученным зашифрованным картинкам в начало был добавлен незашифрованный исходный BMP-заголовок. Были получены следующие картинки:

Режим ECB:



Режим CBC:



По полученным изображениям, что для шифрования картинок ECB не очень подходит, поскольку в этом режиме блоки шифротекста являются независимыми, и для одного и того же исходного блока дает один и тот же шифротекст. Поэтому в картинке можно частично различить контуры текста и даже прочитать некоторые буквы. А при использовании режима CBC картинка полностью заполняется шумом, поскольку теперь очередные блоки шифротекста будут зависеть от предыдущих блоков шифротекста.

*Определение размера сжатых исходной и зашифрованных картинок.*

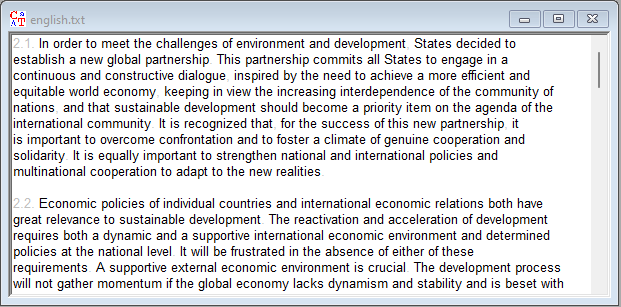
Теперь выполним сжатие трех картинок, полученных в предыдущем пункте, и определим степень сжатия данных изображений.

|  |  |  |  |
| --- | --- | --- | --- |
| Картинка | Исходный размер | Размер после сжатия | Степень сжатия |
| Исходная | 241 664 байт | 8 192 байт | 97% |
| ECB | 241 664 байт | 12 288 байт | 95% |
| CBC | 241 664 байт | 241 664 байт | 0% |

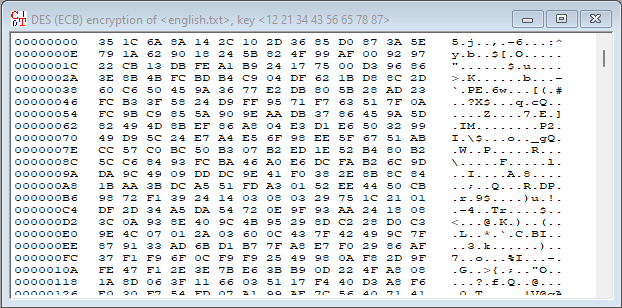
Как видно, для исходной картинки и картинки, зашифрованной с режимом ECB, степень сжатия выше 90%, что можно объяснить большим количество повторяющейся информации (повторяющийся фон). А для картинки, зашифрованной с режимом CBC, степень сжатия равна 0%, что можно объяснить наличием сильного шума.

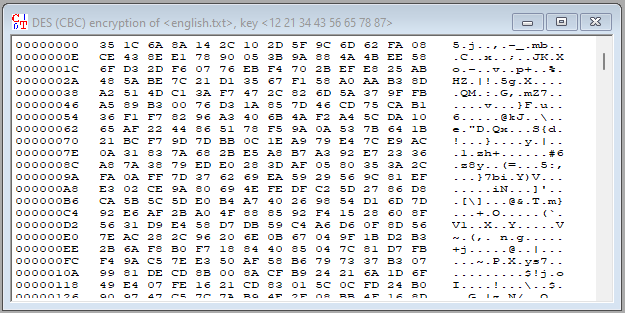
*Оценка времени проведения атаки грубой силы в зависимости от количества известных байт ключа для режимов ECB и CBC.*

Был выбран ключ «12 21 34 43 56 65 78 87» и следующий исходный текст (количество символов больше 1000, текст взят из файла «CrypTool/reference/english.txt»):



Далее исходный текст был зашифрован шифром DES с режимами работы ECB и CBC:



******

Далее оценим время проведения атаки «грубой силы» в случаях, когда известно некоторое количество байт ключа:

|  |  |  |
| --- | --- | --- |
| Режим | Количество известных байт ключа | Время проведения атаки |
| ECB | 6 байт | ~ 0.5 секунд |
| CBC | ~ 0.5 секунд |
| ECB | 5 байт | ~ 15 секунд |
| CBC | ~ 25 секунд |
| ECB | 4 байта | ~ 33 минуты |
| CBC | ~ 53 минуты |
| ECB | 3 байта | ~ 2.9 дней |
| CBC | ~ 4.8 дней |
| ECB | 2 байта | ~ 363 дней |
| CBC | ~ 1.6 лет |
| ECB | 1 байт | ~ 120 лет |
| CBC | ~ 210 лет |

Исходя из полученных результатов, можно сделать вывод, что шифр DES с режимом CBC более криптостойкий по отношению к атаке «грубой силы», чем шифр DES с режимом ECB.

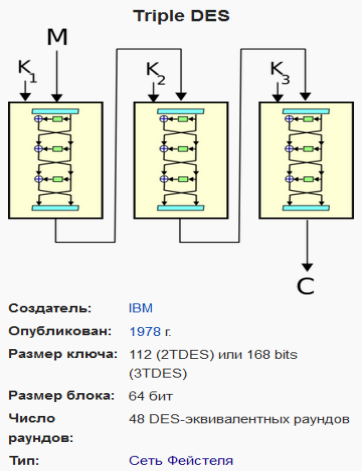
***Исследование 3-DES.***

*Задание.*

1. Выбрать случайный текст на английском языке (не менее 1000 знаков);
2. Создать бинарный файл с этим текстом, зашифровав и расшифровав его DES на 0-м ключе;
3. Снять и сохранить частотную и автокорреляционную характеристику этого файла;
4. Зашифровать бинарный файл шифром 3-DES в режиме ECB;
5. Снять и сохранить частотную и автокорреляционную характеристику файла с шифровкой;
6. Зашифровать исходный бинарный файл 3-DES в режиме CBC c тем же ключом;
7. Снять и сохранить частотную и автокорреляционную характеристику файла с шифровкой;
8. Определить экспериментальным путем по какой схеме работает реализация 3-DES в CrypTool. Сохранить подтверждающие скриншоты.

*Основные параметры и обобщенная схема шифра.*

Шифр 3-DES (рисунок 7) состоит в трехкратном применении обычного DES. Существует 4 основные версии данного шифра:

1. DES-EEE3 – шифрование происходит 3 раза независимыми ключами;

2. DES-EDE3 – операции шифровка-расшифровка-шифровка с тремя разными ключами;

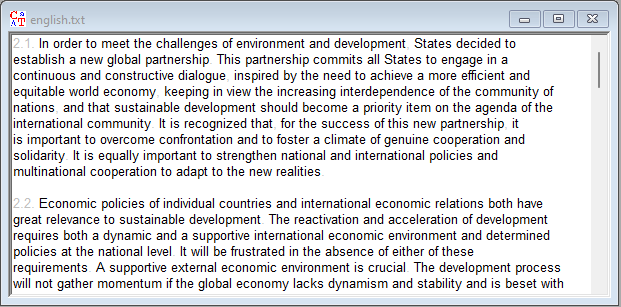
3. DES-EEE2 – то же что и DES-EEE3, но на первом и последнем шаге одинаковый ключ;

4. DES-EDE2 – то же что и DES-EDE3, но на первом и последнем шаге используется один и тот же ключ.

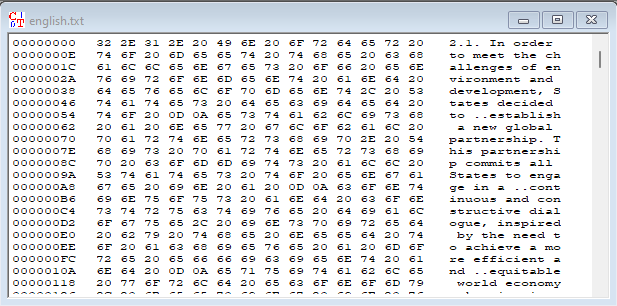
Самая популярная разновидность это DES-EDE3 и DES-EDE2. Реализован во многих приложениях, ориентированных на работу в сети Интернет, в том числе в PGP и S/mime.

*Частотная и автокорреляционная характеристика шифротекста шифра DES и шифра 3-DES с режимами ECB и CBC.*

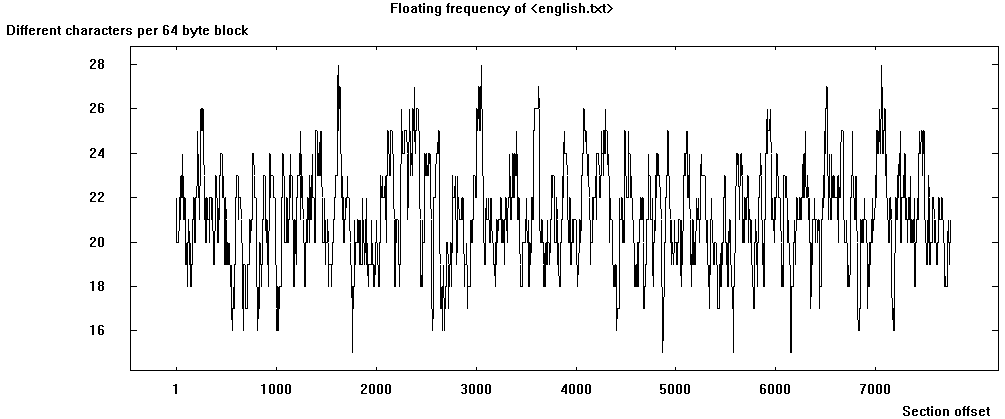
В качестве исходного текста был выбран следующий текст (количество символов больше 1000, текст взят из файла «CrypTool/reference/english.txt»):

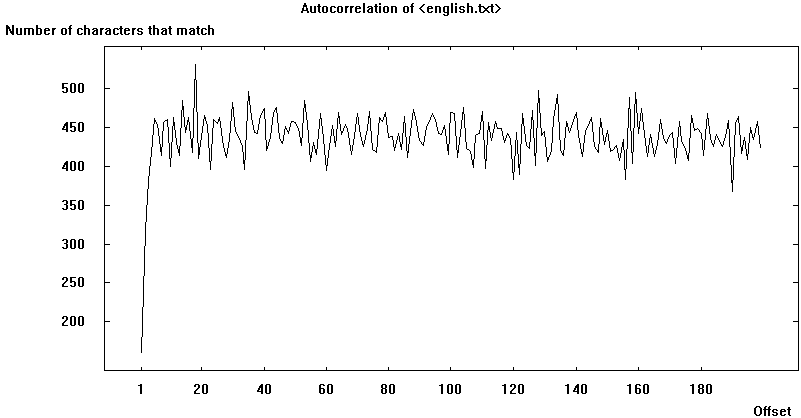


Для представления исходного текста в виде бинарного файла была выбрана опция «Show as HexDump»:

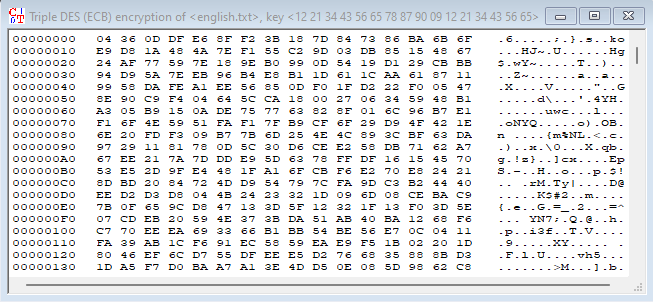


Снимем частотную и автокорреляционную характеристику данного файла:

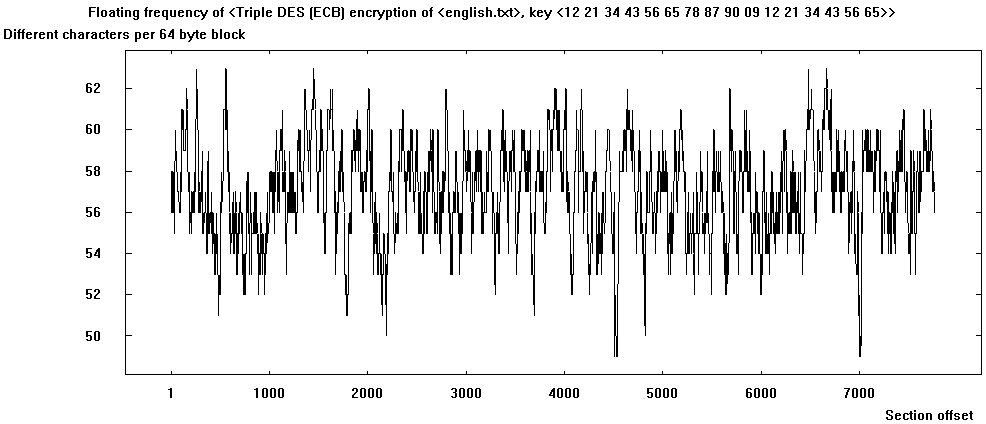


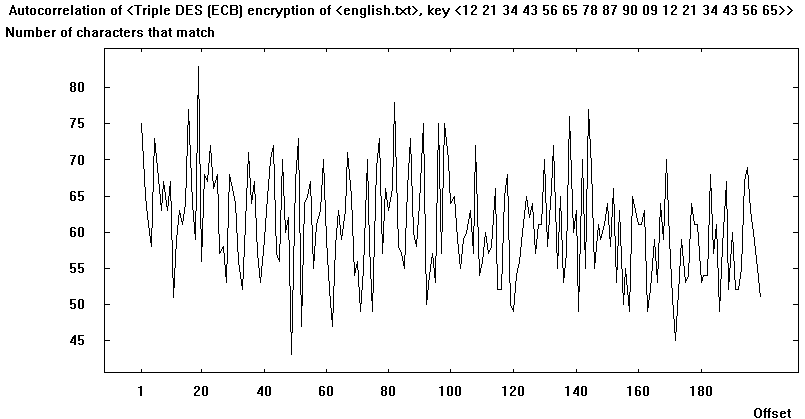


Теперь зашифруем исходный текст шифром 3-DES с использованием режима ECB (используя ключ «12 21 34 43 56 65 78 87 90 09 12 21 34 43 56 65»):

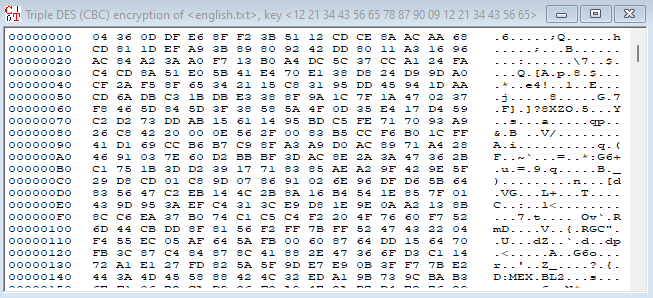


Снимем частотную и автокорреляционную характеристику полученного файла с шифровкой:

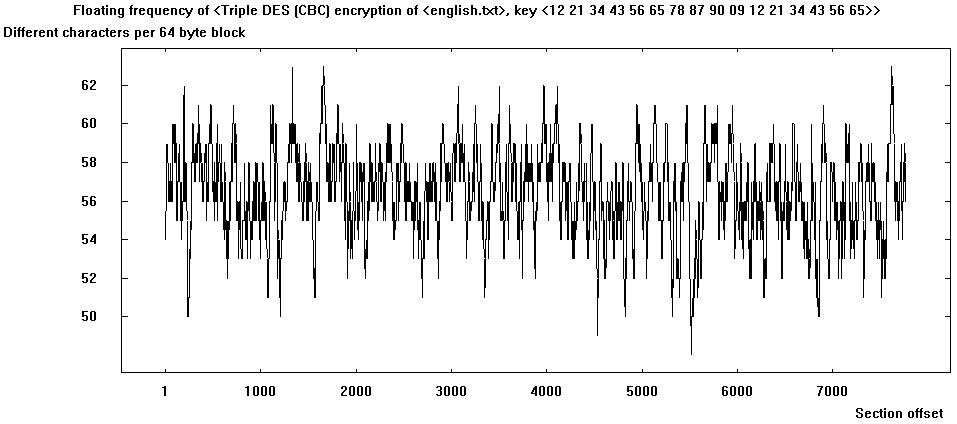


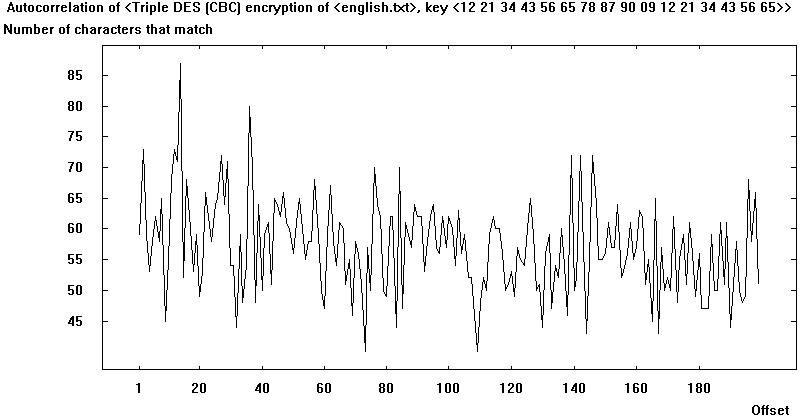


Теперь зашифруем исходный текст шифром 3-DES с использованием режима CBC (используя ключ «12 21 34 43 56 65 78 87 90 09 12 21 34 43 56 65»):

****

Снимем частотную и автокорреляционную характеристику полученного файла с шифровкой:

****

****

Как видно по характеристикам, в исходном тексте символы в блоках по 64 байта зачастую совпадают (значение количества различных символов не превышает 28) и в целом при посимвольном сдвиге текста значительная часть (более 400) символов совпадают. А в зашифрованном шифром 3-DES с режимами ECB и CBC файле – символы в блоках по 64 байта зачастую различаются (значение количества различных символов зачастую превышает 50) и в целом при посимвольном сдвиге текста только незначительная часть символов (40-80) совпадают.

*Оценка времени проведения атаки грубой силы в зависимости от количества известных байт ключа для шифра 3-DES с режимами ECB и CBC.*

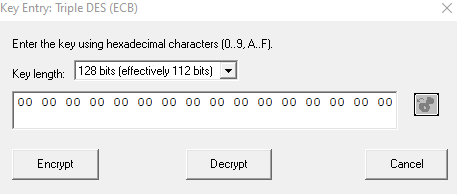
Оценим время проведения атаки «грубой силы» в случаях, когда известно некоторое количество байт ключа:

|  |  |  |
| --- | --- | --- |
| Режим | Количество известных байт ключа | Время проведения атаки |
| ECB | 14 байт | ~ 0.1 секунд |
| CBC | ~ 0.5 секунд |
| ECB | 12 байт | ~ 1 час 3 минуты |
| CBC | ~ 1 час 13 минут |
| ECB | 10 байта | ~ 1.9 лет |
| CBC | ~ 2.3 года |
| ECB | 8 байта | ~ 31000 лет |
| CBC | ~ 36000 лет |
| ECB | 6 байта | ~ 510000000 лет |
| CBC | ~ 590000000 лет |
| ECB | 4 байт | ~ 7900000000000 лет |
| CBC | ~ 9300000000000 лет |

Исходя из полученных результатов, можно сделать вывод, что шифр 3-DES с режимом CBC более криптостойкий по отношению к атаке «грубой силы», чем шифр 3-DES с режимом ECB. Шифр 3-DES значительно более криптостойкий по отношению к атаке «грубой силы», чем шифр DES. Если нам известна половина ключа, то шифр DES можно взломать примерно за полчаса, а шифр 3-DES – примерно за 2 года.

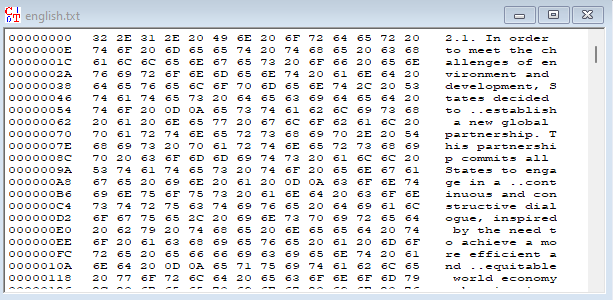
*Схема реализации шифра 3-DES в CrypTool.*

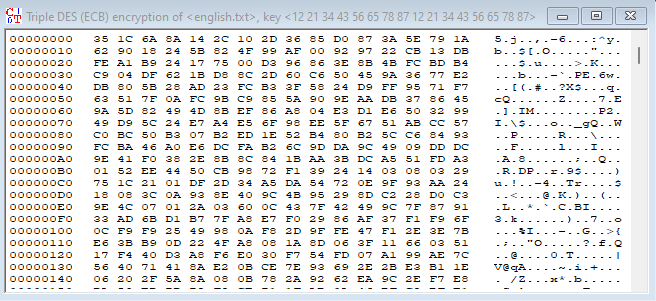
Поскольку шифр 3-DES в CrypTool использует ключ размера 16 байт, то можно сделать вывод, что шифр 3-DES в CrypTool использует схему DES-EEE2 или DES-EDE2.



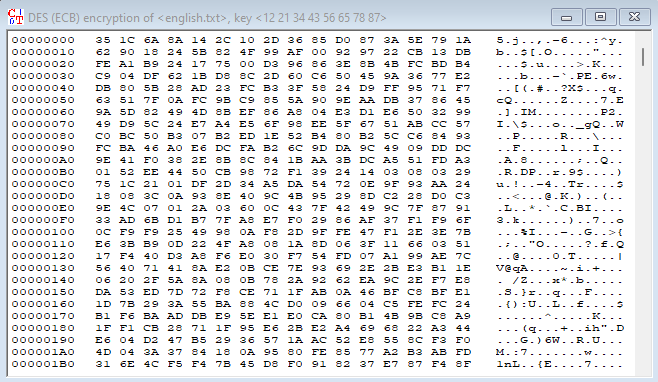
Поскольку DES-EEE2 и DES-EDE2 для первого и последнего шифрования использует один и тот же ключ, можно попробовать зашифровать текст вручную при помощи трехкратного применения шифрования или дешифрования DES. В зависимости от комбинации шифрования или дешифрования, можно определить схему шифра.

Зашифруем исходный текст при помощи 3-DES с ключом «12 21 34 43 56 65 78 87 12 21 34 43 56 65 78 87»:





Теперь подряд зашифруем исходный текст при помощи DES с ключом «12 21 34 43 56 65 78 87»:



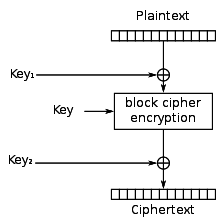
Как видно, шифротекст совпадает с шифротекстом 3-DES. Поскольку для шифра 3-DES мы задали ключ, левая и правая половины которых совпадает, можно предположить, что шифр 3-DES зашифровал текст ключом «12 21 34 43 56 65 78 87», после чего расшифровал его этим же ключом, получив исходный текст, после чего заново зашифровал его ключом «12 21 34 43 56 65 78 87». Отсюда можно сделать вывод, что CrypTool 1 использует схему DES-EDE2.

***Исследование модификаций DESX, DESL, DESXL шифра DES.***

*Задание.*

1. Выбрать случайный текст на английском языке (не менее 1000 знаков);
2. Создать бинарный файл с этим текстом, зашифровав и расшифровав его DES на 0-м ключе;
3. С помощью CrypTool зашифровать текст с использованием шифров DESX, DESL, DESXL;
4. Средствами CrypTool вычислить энтропию исходного текста и шифротекстов, полученных в итоге. Зафиксировать результаты измерений в таблице;
5. Средствами CrypTool оцените время проведения атаки «грубой силы» при полном отсутствии информации о секретном ключе.

*Основные параметры и обобщенные схемы шифров.*

Алгоритм DESX использует на входе ключ длиной 184 бита, который делится на одну 56-битную часть и две 64-битные части. Процесс шифрования происходит по следующей схеме:



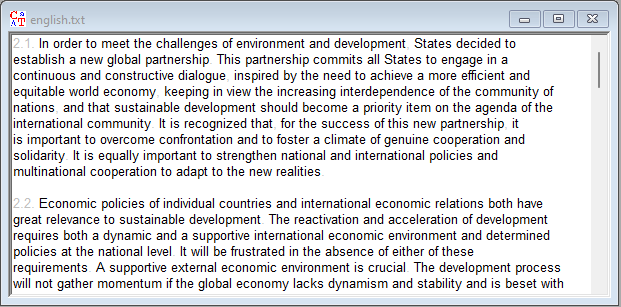
Если 𝐾1 = 𝐾2 = 0, то данный алгоритм сводится к стандартному DES.

Алгоритм DESL является облегченной версией алгоритма DES. Алгоритм предполагает отказ от входной и выходной перестановки блока текста, т.к. они не несут криптографической сложности, а также 8 S-блоков заменяется на 1, но более стойкий чем все 8 стандартных блока DES.

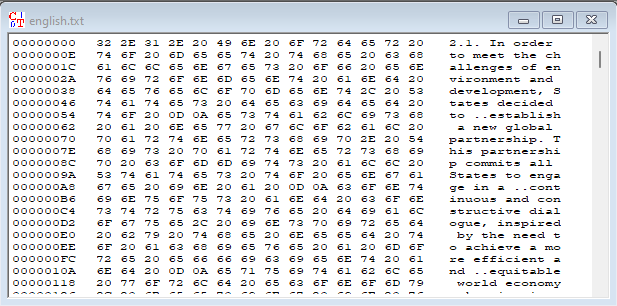
Алгоритм DESXL использует те же оптимизации что и DESL, но производит шифрование по алгоритму DESX.

*Зависимость энтропии шифротекста от используемого шифра.*

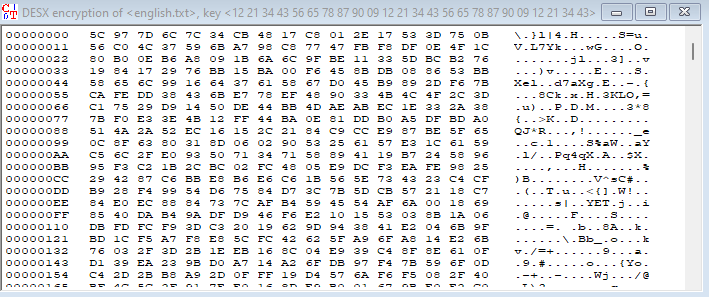
В качестве исходного текста был выбран следующий текст (количество символов больше 1000, текст взят из файла «CrypTool/reference/english.txt»):



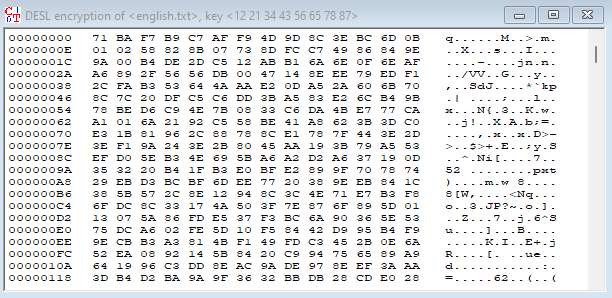
Для представления исходного текста в виде бинарного файла была выбрана опция «Show as HexDump»:



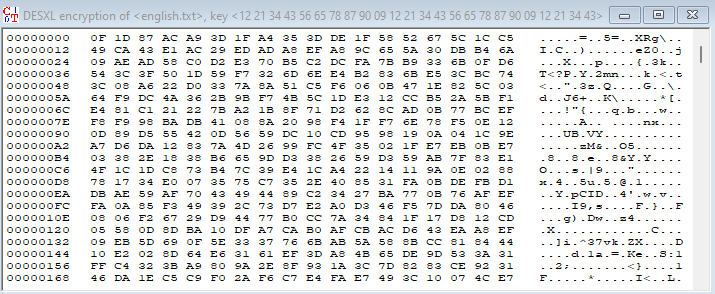
Далее зашифруем исходный текст при помощи шифра DESX с использованием ключа «12 21 34 43 56 65 78 87 90 09 12 21 34 43 56 65 78 87 90 09 12 21 34 43»:



Далее зашифруем исходный текст при помощи шифра DESL с использованием ключа «12 21 34 43 56 65 78 87»:



Далее зашифруем исходный текст при помощи шифра DESXL с использованием ключа «12 21 34 43 56 65 78 87 90 09 12 21 34 43 56 65 78 87 90 09 12 21 34 43»:



Далее вычислим энтропию исходного текста и полученных шифротекстов:

|  |  |
| --- | --- |
| Текст | Значение энтропии |
| Исходный текст |  |
| Шифротекст DESX |  |
| Шифротекст DESL |  |
| Шифротекст DESXL |  |

Как видно из результатов, для исходного текста значение энтропии далеко от максимума, а используемые значения байт не охватывают весь диапазон возможных значений. Для шифротекстов DESX, DESL, DESXL значение энтропии почти равно максимальному значению энтропии, а используемые значения байт охватывают весь диапазон возможных значений.

*Оценка времени проведения атаки грубой силы в зависимости от количества известных байт ключа для шифров.*

Оценим время проведения атаки «грубой силы» в случаях, когда известно некоторое количество байт ключа:

|  |  |  |
| --- | --- | --- |
| Шифр | Количество известных байт ключа | Время проведения атаки |
| DESX | 22 байт | ~ 0.5 секунд |
| DESXL | ~ 0.5 секунд |
| DESX | 20 байта | ~ 5 часов 53 минуты |
| DESXL | ~ 4 часа 55 минут |
| DESX | 16 байта | ~ 2.9 \* 106 лет |
| DESXL | ~ 2.4 \* 106 лет |
| DESX | 12 байта | ~ 1.3 \* 1016 лет |
| DESXL | ~ 1.1 \* 1016 лет |
| DESX | 8 байт | ~ 5.4 \* 1025 лет |
| DESL | ~ 0 секунд |
| DESXL | ~ 4.5 \* 1025 лет |
| DESX | 4 байт | ~ 1.4 \* 1034 лет |
| DESL | ~ 17 минут |
| DESXL | ~ 1.2 \* 1034 лет |
| DESX | 0 байт | ~ 4 \* 1042 лет |
| DESL | ~ 8600 лет |
| DESXL | ~ 3.2 \* 1042 лет |

Как видно из результатов, DESL менее криптостойкий по отношению к атаке «грубой силы», чем шифр DES. Шифры DESX и DESXL значительно более криптостойки по отношению к атаке «грубой силы», чем DES и 3-DES. Шифр DESX более криптостойкий по отношению к атаке «грубой силы», чем шифр DESXL.

**Выводы.**

В ходе выполнения данной лабораторной работы были рассмотрены следующие шифры:

* Шифр DES с режимами EBC и CBC;
* Шифр 3-DES с режимами EBC и CBC (модификация шифра DES);
* Шифры DESX, DESL, DESXL (модификации шифра DES).

1. Для шифра DES:
   1. Был изучен демонстрационный пример выполняемых преобразований при шифровании блока данных в CrypTool 1;
   2. Было определено, что шифр DES – симметричный блочный шифр, основанный на сети Фейстеля. Размер блока составляет 8 байт. Размер ключа составляет 7 байт. Процесс шифрования основан на выполнении 16 раундов преобразований, в каждом из которых используется свой 48-битный раундовый ключ. Расшифрование производится при помощи прохождения всех шагов шифра в обратном порядке;
   3. Были изучены два режима работы ECB и CBC;
   4. Для режима ECB определено: в данном режиме каждый блок шифруется независимо от других блоков одним и тем же способом. Исходя из этого, есть возможность распараллеливания процесса шифрования/дешифрования. Но есть недостаток – одинаковые исходные блоки при шифровании дают одинаковые блоки шифротекста;
   5. Для режима CBC определено: в данном режиме каждый блок шифруется с использованием шифротекста предыдущего блока (перед шифрованием производится сложение по модулю 2 исходного текста с шифротекстом предыдущего блока). Исходя из этого, возможность распараллеливания процесса шифрования/дешифрования отсутствует. Но теперь при шифровании одинаковых исходных блоков получаются разные блоки шифротекста. При шифровании режимом CBC шифротекст получается более криптостойкий по отношению к атаке «грубой силы», чем при шифровании режимом CBC;
   6. Было определено, что при шифровании изображения шифром DES с режимом ECB зашифрованное изображение в некоторой степени сохраняет контуры деталей, поскольку для одинаковых исходных блоков будут вычисляться одинаковые блоки шифротекста. Сжатие шифротекста при этом достигает 95% на уровне сжатия исходного изображения, равного 97%;
   7. Было определено, что при шифровании изображения шифром DES с режимом CBC зашифрованное изображение полностью теряет детали и превращается в шум, поскольку для одинаковых исходных блоков теперь будут вычисляться разные блоки шифротекста. Сжатие шифротекста при этом равно 0%.
2. Для шифра 3-DES:
   1. Было определено, что шифр 3-DES основан на последовательном использовании DES с тремя или двумя независимыми ключами. Размер блока составляет 8 байт. Размер ключа составляет 14 или 21 байт. Последовательно применяются 3 операции шифрования или дешифрования шифром DES;
   2. Были рассмотрены 4 основные схемы работы: DES-EEE3, DES-EEE2, DES-EDE3 и DES-EDE2. Три буквы после DES означают порядок использования типов операции (шифрование или дешифрования), а цифра – количество используемых ключей. В случае использования двух ключей по 7 байт для первого и последнего преобразования используется один и тот же ключ;
   3. Были изучены два режима работы ECB и CBC;
   4. Было определено, что шифр 3-DES значительно более криптостойкий по отношению к атаке «грубой силы», чем шифр DES;
   5. Для режима CBC было определено, что он является более криптостойким по отношению к атаке «грубой силы», чем шифр 3-DES с режимом ECB;
   6. Было определено, что приложение CrypTool 1 использует схему DES-EDE2.
3. Для шифра DESX:
   1. Было определено, что шифр DESX выполняет шифрование по следующей формуле: DESX(M) = K2 ⨁ DES(M ⨁ K1, K). Размер блока составляет 8 байт. Размер ключа составляет 23 байта;
   2. Было определено, что шифротекст обладает высокой энтропией;
   3. Было определено, что шифр является более криптостойким по отношению к атаке «грубой силы», чем шифры DESXL, DESL, DES и 3-DES;
   4. Были определено, что при отсутствии знаний о ключе, атака «грубой силы» будет осуществляться порядка 1042 лет.
4. Для шифра DESL:
   1. Было определено, что шифр DESL является упрощенной версией шифра DES, в котором отказались от начальной входной и выходной перестановки блока, а также в котором используется один, но более криптостойкий, S-блок вместо восьми S-блоков;
   2. Было определено, что шифротекст обладает высокой энтропией;
   3. Было определено, что шифр является менее криптостойким по отношению к атаке «грубой силы», чем шифры DESXL, DESX, DES и 3-DES;
   4. Были определено, что при отсутствии знаний о ключе, атака «грубой силы» будет осуществляться порядка 103 лет.
5. Для шифра DESXL:
   1. Было определено, что шифр DESXL производит шифрование по алгоритму DESX, но с использованием оптимизаций из шифра DESL;
   2. Было определено, что шифротекст обладает высокой энтропией;
   3. Было определено, что шифр является более криптостойким по отношению к атаке «грубой силы», чем шифры DESL, DES и 3-DES;
   4. Были определено, что при отсутствии знаний о ключе, атака «грубой силы» будет осуществляться порядка 1042 лет.

Были получены практические навыки работы с рассматриваемыми шифрами с использованием приложения CrypTool 1.