МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

ОТЧЕТ

по практической работе №2
по дисциплине «Криптографические методы защиты информации»
Тема: «Изучение шифров DES и AES»

Студент гр. 9361 _____ Кисляков Н.

Преподаватель _____ Племянников А.К.

Санкт-Петербург
2023

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 версий 1 и 2.

Исследовать характеристики шифра Rijndael и других финалистов конкурса AES, а также изучить атаку предсказанием дополнения на симметричные блочные шифры в режиме использования CBC. Получить практические навыки работы с шифрами и алгоритмом проведения атаки, в том числе с использованием приложения CrypTool 1 и 2.

1. DES

1.1 Исследование преобразований DES

1.1.1 Задание

1) Изучить преобразования шифра DES с помощью демонстрационного приложения из CrypTool 1: Indiv.Procedures –> Visualization… –> DES…

1.1.2 Основные характеристики и описание DES

Стандарт шифрования данных (DES) —блочный шифр с симметричными ключами, разработан Национальным Институтом Стандартов и Технологии (NIST –National Instituteof Standards and Technology). Шифр DES основан на сети Фейстеля. Алгоритм DES шифрует информацию блоками по 64 бита с помощью 64-битного ключа шифрования. Шифрование выполняется следующим образом:

1. Над 64-битными блоками производится начальная перестановка согласно таблице.

2. Результат предыдущей операции делится на 2 субблока по 32 бита (A_0 и B_0), над которыми производятся 16 раундов преобразований:

$$A_i = B_{i-1};$$

$$B_i = A_{i-1} \oplus f(B_{i-1}, K_i),$$

где: i – номер текущего раунда, K_i – ключ раунда, $\oplus$ – логическая операция *XOR*.

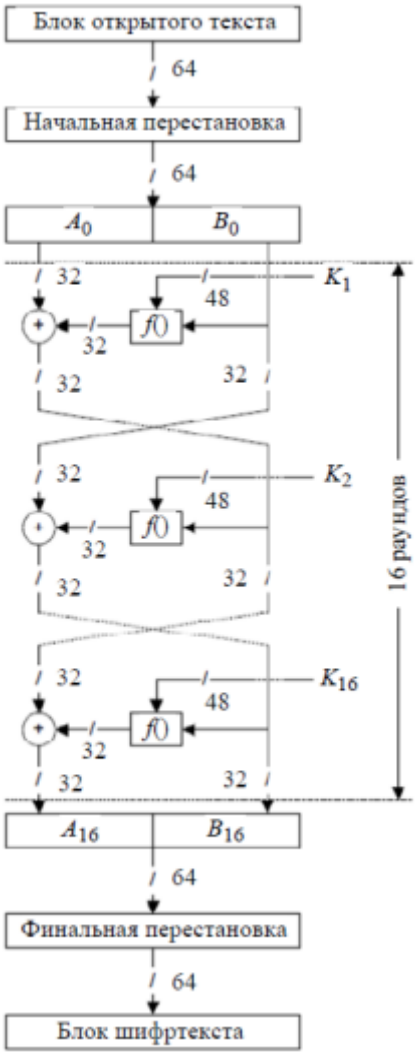


Рисунок 1 – DES

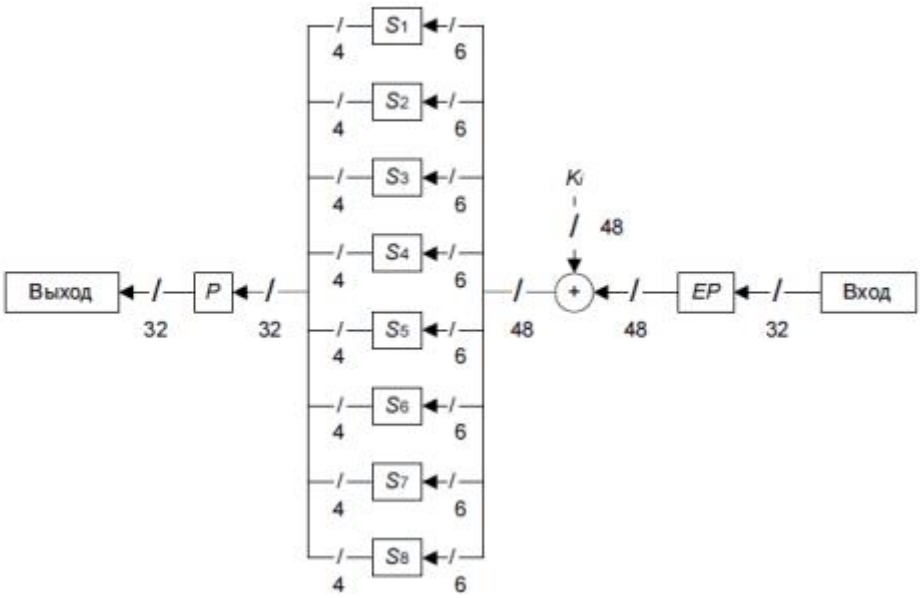


Рисунок 2 – Схема работы функции раунда

Этапы функции:

a) Расширяющая перестановка EP, которая преобразует входные 32 бита в 48 бит.

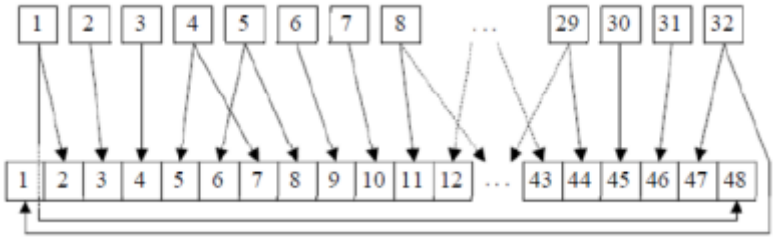


Рисунок 3 – Расширяющая перестановка EP

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 P).

e) На последнем раунде алгоритма субблоки не меняются местами.

3. Полученные субблоки A_{16} и B_{16} образуют 64-битный блок, над которым производится перестановка. Результатом перестановки является шифротекст.

Процедура генерации раундовых ключей представлена на рисунке 4.

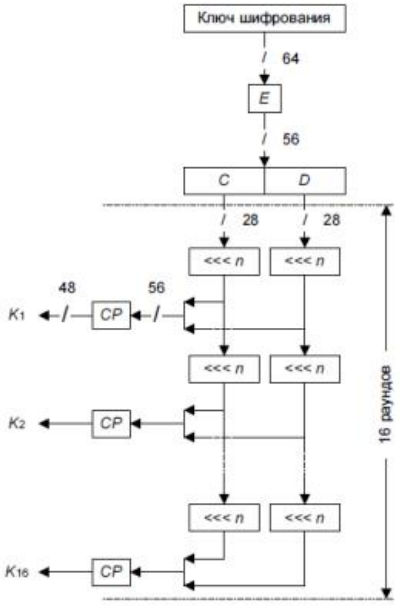


Рисунок 4 – Процедура генерации раундовых ключей

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.

1.1.3 Визуализация в CrypTool 1

В CrypTool 1 изучим преобразование шифра DES.

Для шифрования DES открытый текст разбивается на блоки по 64 бита. Каждый блок (т.е. ввод X) будет проходить следующий алгоритм для полного шифрования открытого текста.

Первым шагом является перестановка всех битов входного X в соответствии с начальной перестановкой (IP) на рисунке 5, 6.

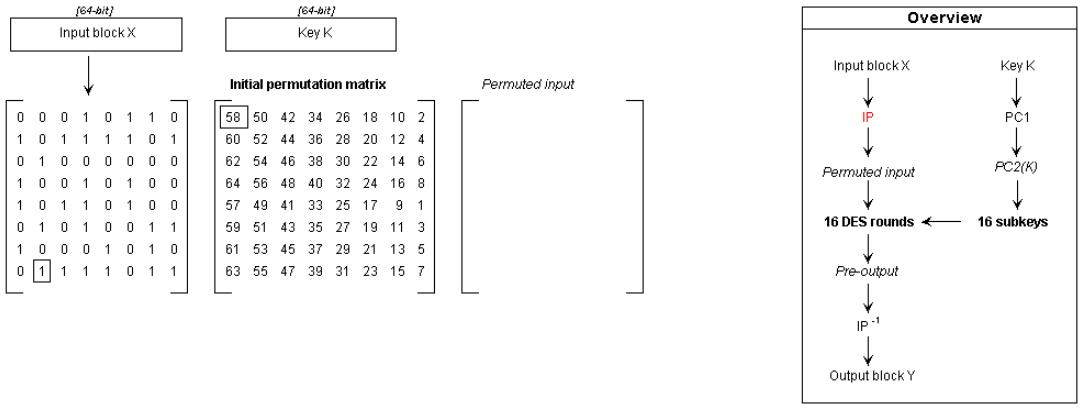


Рисунок 5 – Перестановка битов текста

Рисунок 6 – Конечный результат перестановки битов текста

Следующим шагом является перестановка 56-битного ключа. На вход идет 64-битный ключ, но каждый 8 бит удаляется, так как эти биты предназначены для проверки четности и не оказывают влияние на шифрование. Перестановка представлена на рисунках 7, 8.

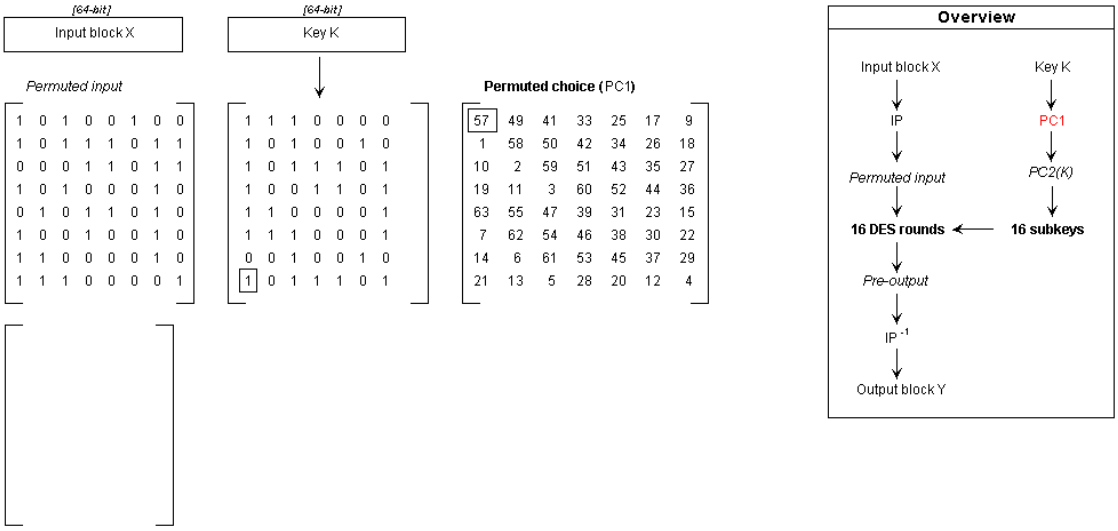


Рисунок 7 – Удаление правых битов и перестановка битов ключа

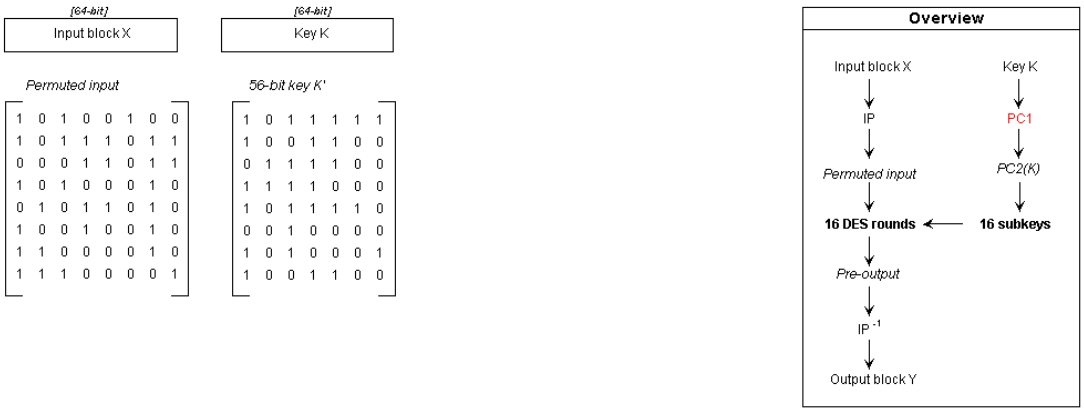
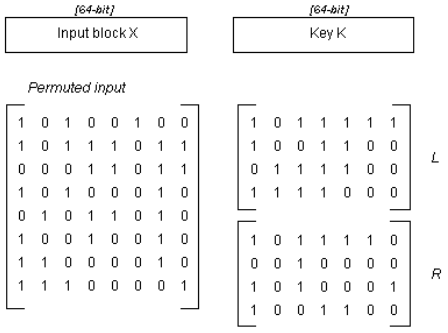


Рисунок 8 – Результат перестановки битов ключа

Далее генерируются 16 48-битных ключей из полученного на предыдущем шаге 56-битного ключа. Для этого ключ делится на две половины, как показано на рисунке 9.



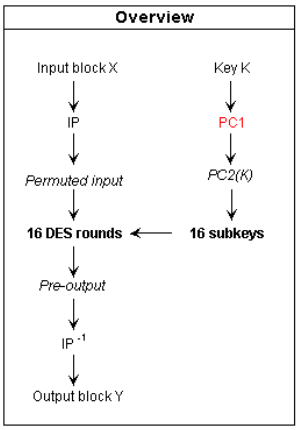


Рисунок 9 – Деление ключа на две половины

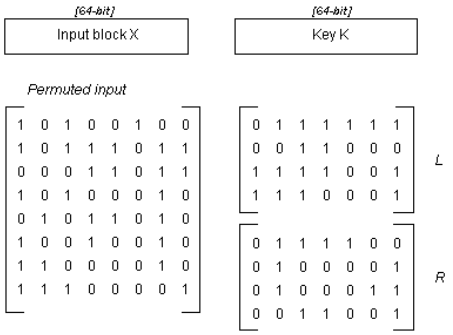
Для левой и правой части производится побитовый сдвиг на определенное количество бит, в зависимости от порядкового номера ключа и согласно схеме, приведенной на рисунке 10.

Round number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
# of bits to rotate	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Рисунок 10 – Схема для генерации раундовых ключей

Рассмотрим в качестве примера генерацию первого раундового ключа.

Сначала выполняем побитовый сдвиг влево на 1 бит, согласно схеме, приведенной выше. Результат побитового сдвига продемонстрирован на рисунке 11.



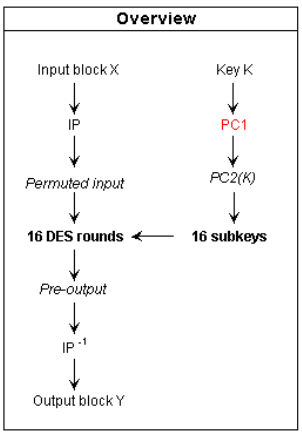


Рисунок 11 – Результат побитового сдвига

Дальше объединяем две половины вместе и выполняем сжимающую перестановку (PC2), как представлено на рисунке 12.

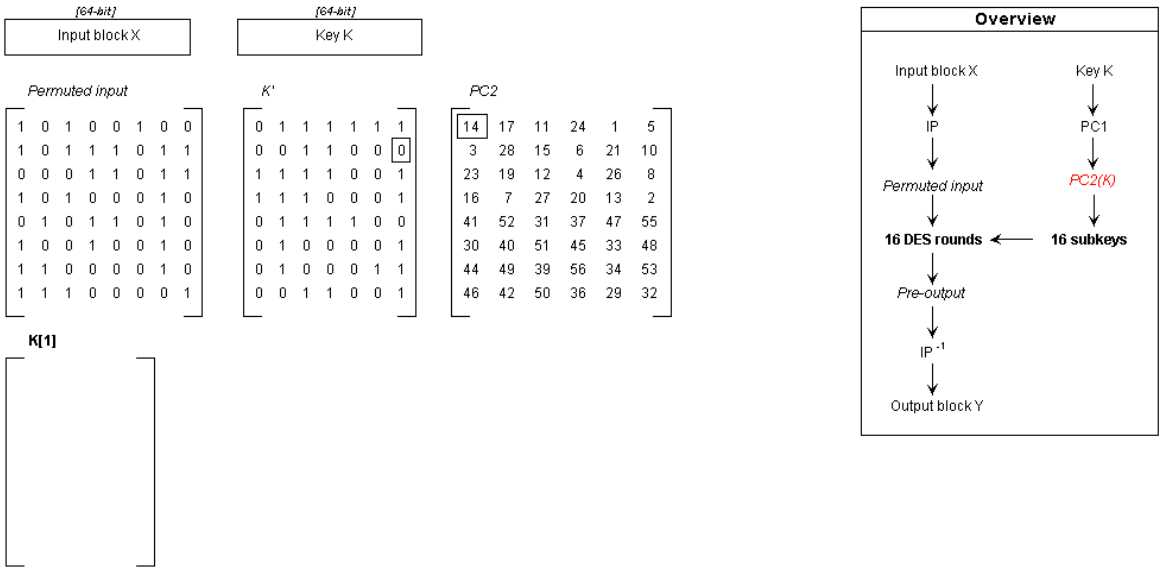


Рисунок 12 – Сжимающая перестановка

В итоге первый раундовый ключ будет выглядеть следующим образом (рисунок 13).

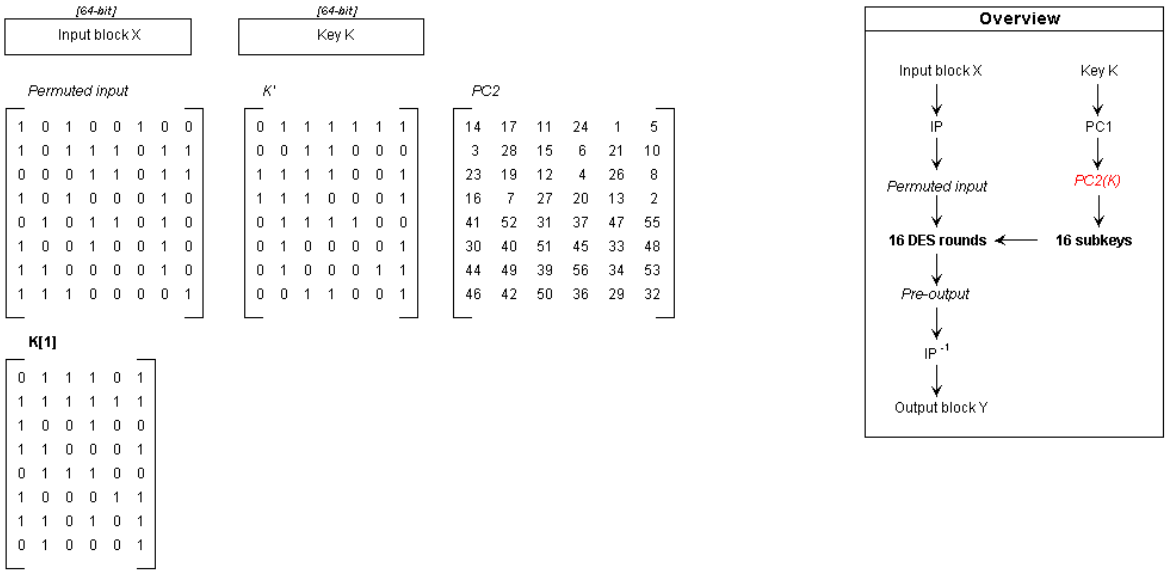


Рисунок 13 – Первый раундовый ключ

Аналогичным образом происходит генерация остальных 15 раундовых ключей. На этом все предварительные условия для основного алгоритма выполнены и можно переходить к шифрованию. Шифрование производится в 16 этапов, на каждом этапе используется соответствующий ключ (порядковый

номер этапа). Данный процесс называется сетью Фейстеля. Рассмотрим более подробно преобразования, происходящие в раундах.

На входе раунда данные делятся на две части по 32 бита каждая (левая L_0 и правая R_0). В каждом раунде выполняются следующие действия:

$$L_i = R_{i-1},$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i),$$

где $i = \overline{1,16}$, $\oplus$ - операция XOR.

Схема первых двух раундов показана на рисунке 14.

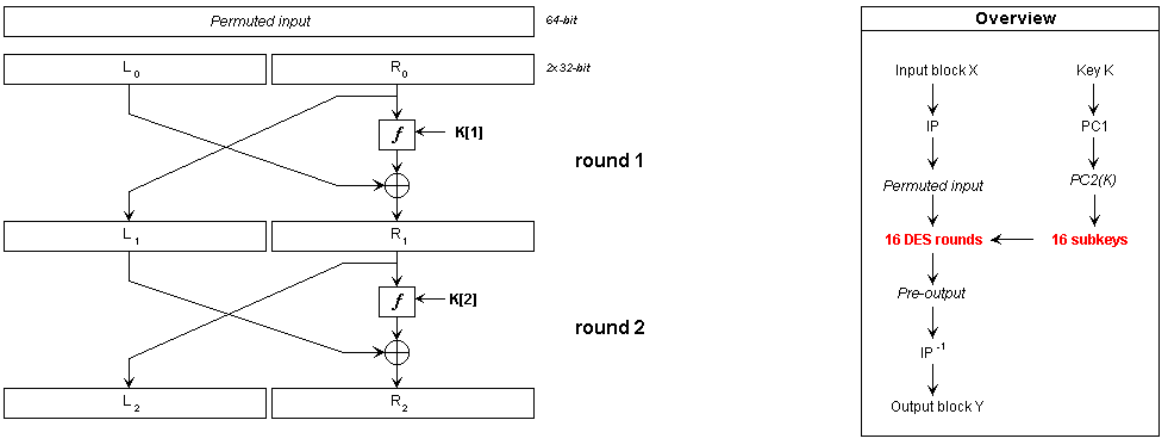


Рисунок 14 – Схема первых двух раундов

После выполнения 16 раундов получаем два блока $LL16$ и $RR16$ по 32 бита, которые меняются местами, как продемонстрировано на рисунке 15.

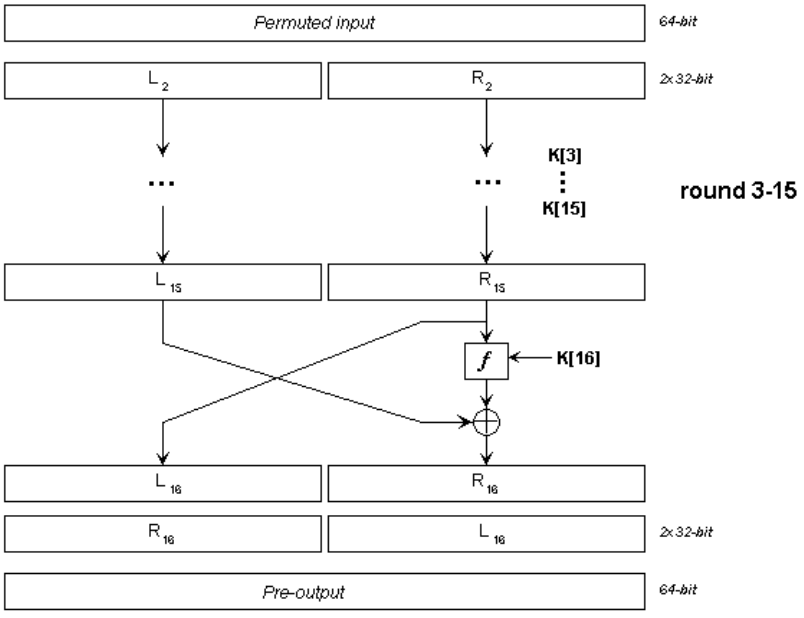


Рисунок 15 – Результат выполнения 16 раундов

Рассмотрим более подробно основную функцию DES $ff()$, которая используется в каждом раунде преобразований. Сначала производится расширяющая перестановка (E), которая преобразует правый 32-битный блок в 48-битный блок, как представлено на рисунке 16.

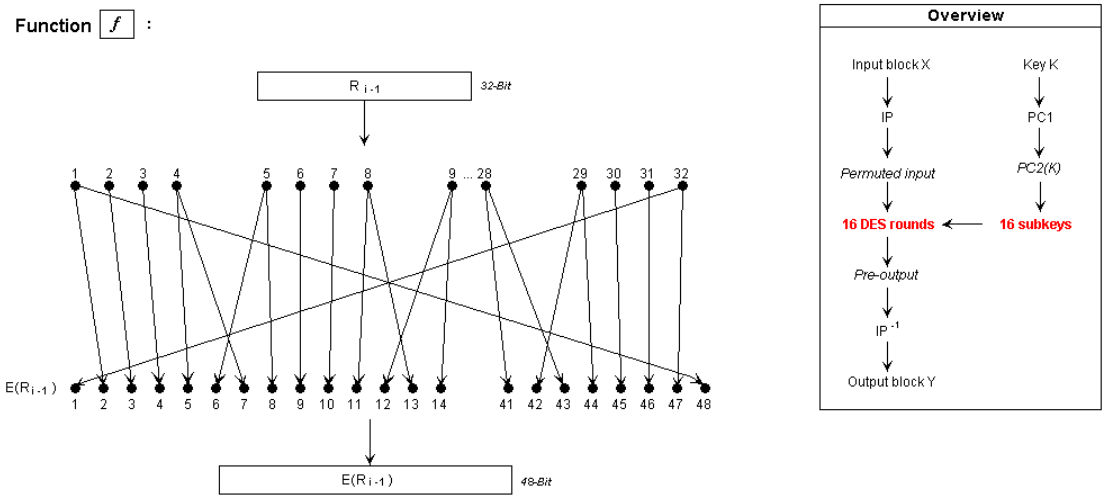


Рисунок 16 – Расширяющая перестановка

Полученный 48 битный блок складывается по модулю 2 с раундовым ключом, после чего делится на 8 блоков по 6 бит, как показана на рисунке 17.

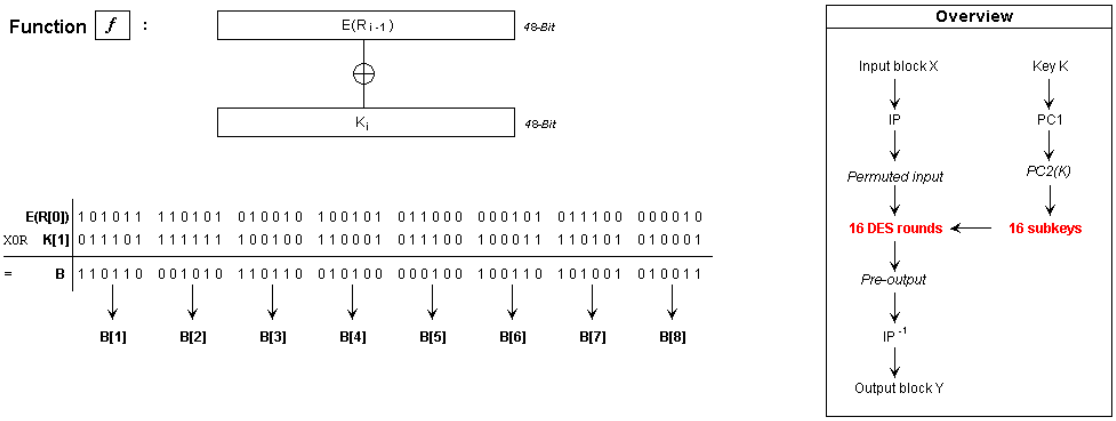


Рисунок 17 – Сложение по модулю 2 с раундовым ключом

Каждый блок $B[ii]$ обрабатывается соответствующей ему таблицей замены $S[i]$ (4 × 16). Первый и последний бит используются для определения номера строки, а оставшиеся 4 бита для определения номера столбца в таблице $S[i]$. Берем соответствующее значение, лежащее на пересечении номера строки и столбца, и переводим его в двоичный вид. Пример выполнения замены для блока $BB[1]$ продемонстрированы на рисунке 18, 19.

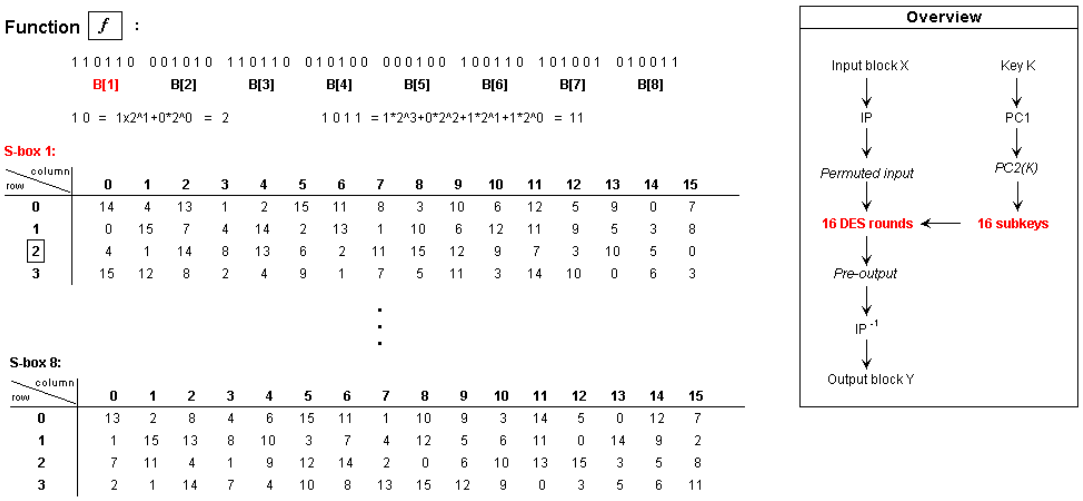


Рисунок 18 – Выполнение замены для первого блока

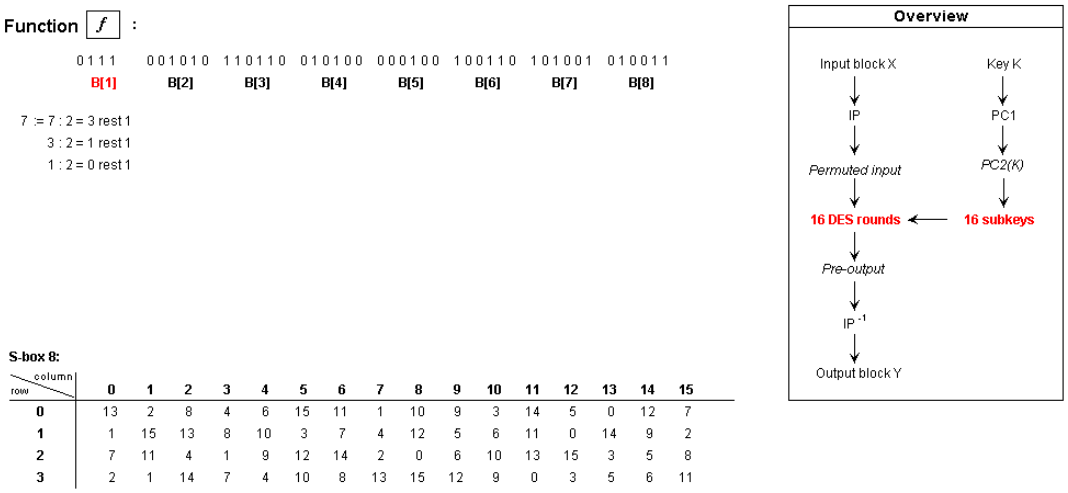


Рисунок 19 – Результат замены для первого блока

В итоге для $B[1]$заменой будет являться значение 0111_2. Таким образом 6-битные блоки заменяются на 4-битные, как представлено на рисунке 20.

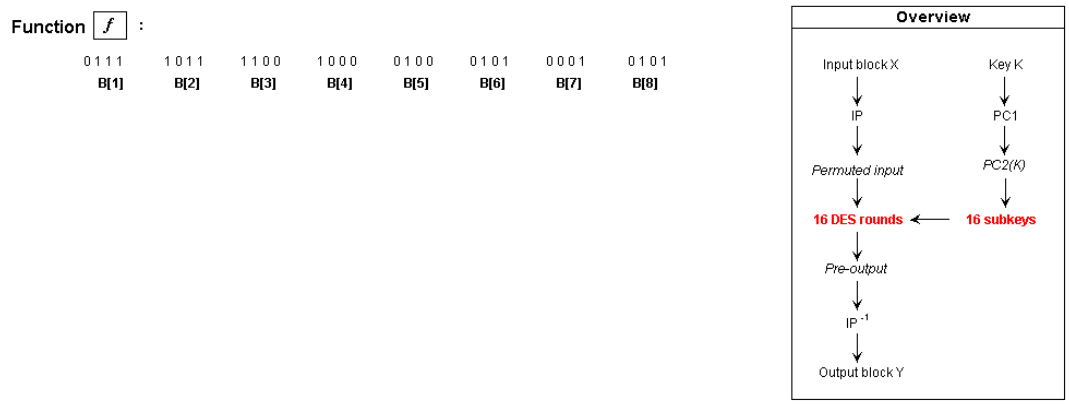


Рисунок 20 – Результат замен всех блоков

Затем объединяем полученные блоки в один 32-битный блок, для которого производим финальную перестановку (P), как показано на рисунке 21.

Рисунок 21 – Финальная перестановка

В итоге мы получаем результат преобразования функцией $f()$ блока R_0 с раундовым ключом K_1, как продемонстрировано на рисунке 22.

Рисунок 22 – Результат преобразования функции

Таким же образом происходит вычисление функции $f()$ в раундах. После прохождения всех раундов выполняется обратная начальная перестановка (IP^{-1}), как представлено на рисунке 23.

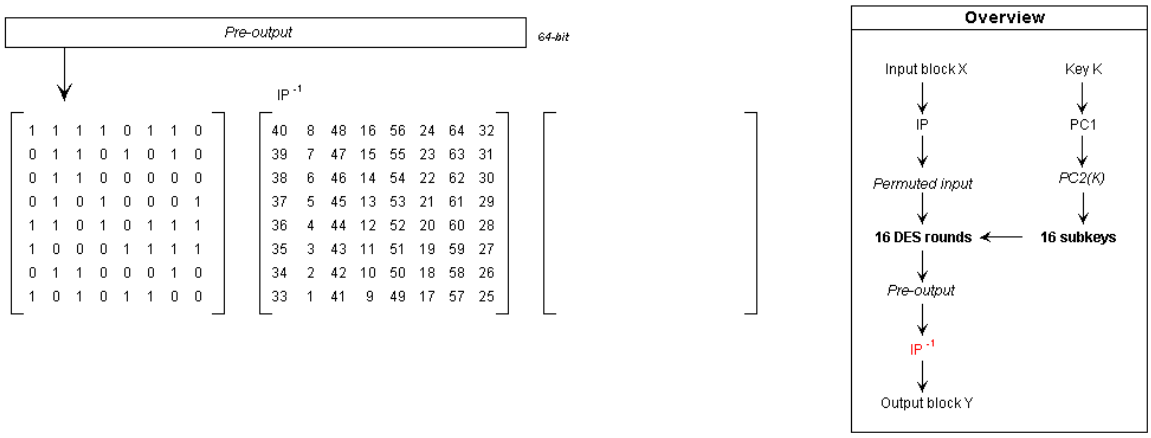


Рисунок 23 – Обратная начальная перестановка

На этом шифрование заканчивается. Мы получили зашифрованный блок, как показано на рисунке 24.

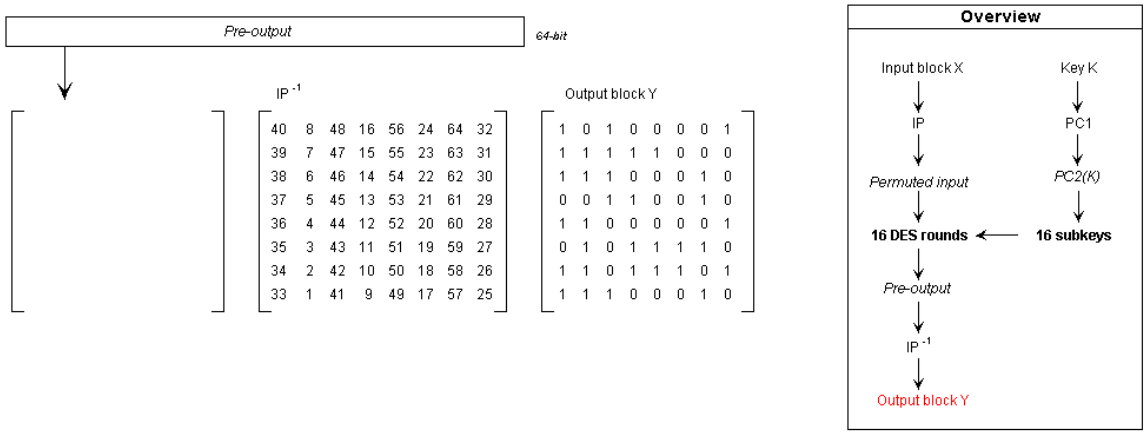


Рисунок 24 – Зашифрованный текст

1.2 Исследование DES в режимах ECB и CBC

1.2.1 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 с тем же ключом.

4) Сохранить шифровки в виде картинок для отчета.

5) Сжать исходную и две зашифрованных картинки средствами CrypTool. Зафиксировать размеры полученных файлов в таблице.

1.2.2 Схемы использования DES в режимах ECB и CBC

Режимы использования симметричных блочных шифров предназначены для зашифрования больших файлов данных. В режиме ECB шифр DES используется независимо для каждого 64-битного блока исходного файла

данных. Схема использования шифра DES в режиме ECB продемонстрирована на рисунке 25.

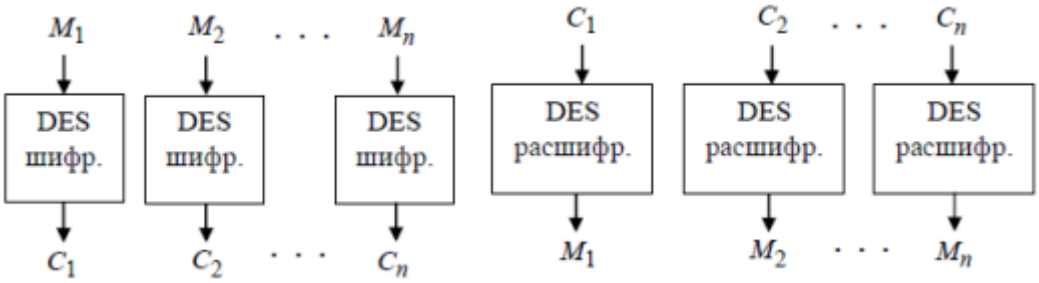


Рисунок 25 – Схема использования шифра DES в режиме ECB

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 DES в режиме CBC представлена на рисунке 26.

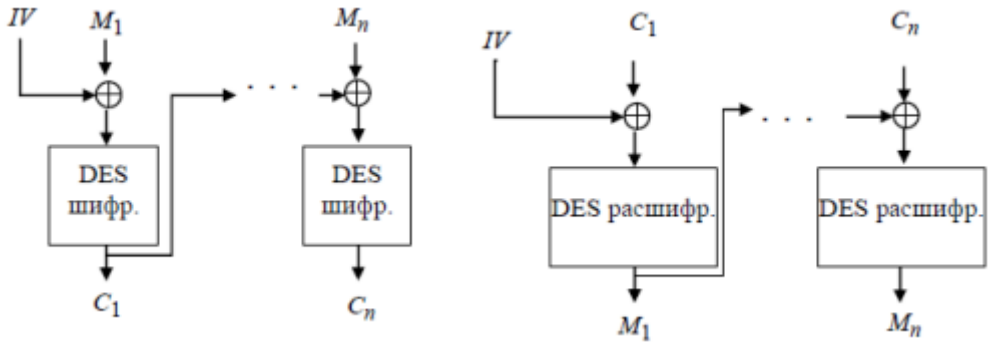


Рисунок 26 – Схема использования шифра DES в режиме CBC

1.2.3 Скриншоты исходного и шифрованных изображений в разных режимах работы шифров

Создадим картинку, на которой будет ФИО на белом фоне, в формате bmp на рисунке 27.

Кисляков Никита Алексеевич

Рисунок 27 – ФИО

Используем ключ «15 15 23 65 95 45 15 20» и зашифруем при помощи DES с режимом ECB. Результат представлен на рисунке 28.

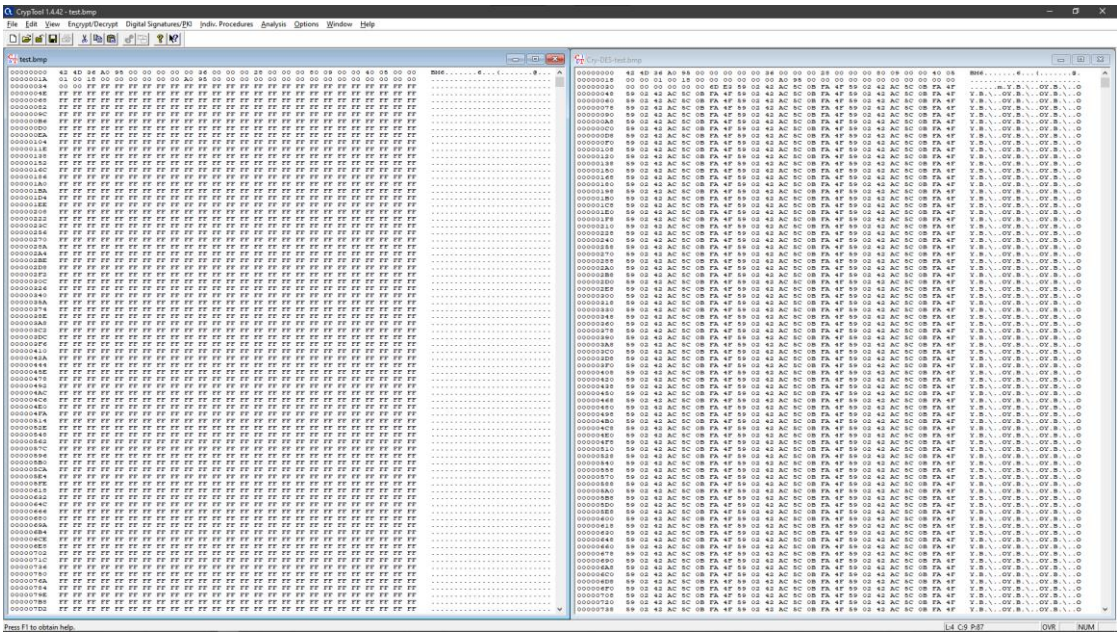


Рисунок 28 – DES с режимом ECB

Используем ключ «15 15 23 65 95 45 15 20» и зашифруем при помощи DES с режимом CBC. Результат представлен на рисунке 28.

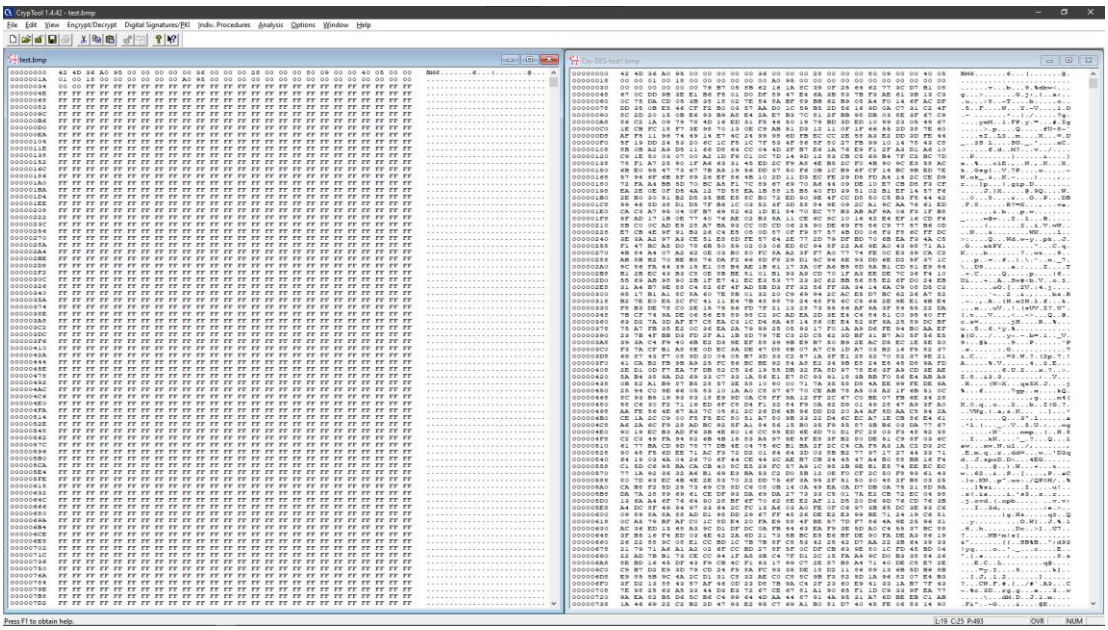


Рисунок 29 – DES с режимом CBC

В итоге мы получили следующий результат на рисунка 30,31

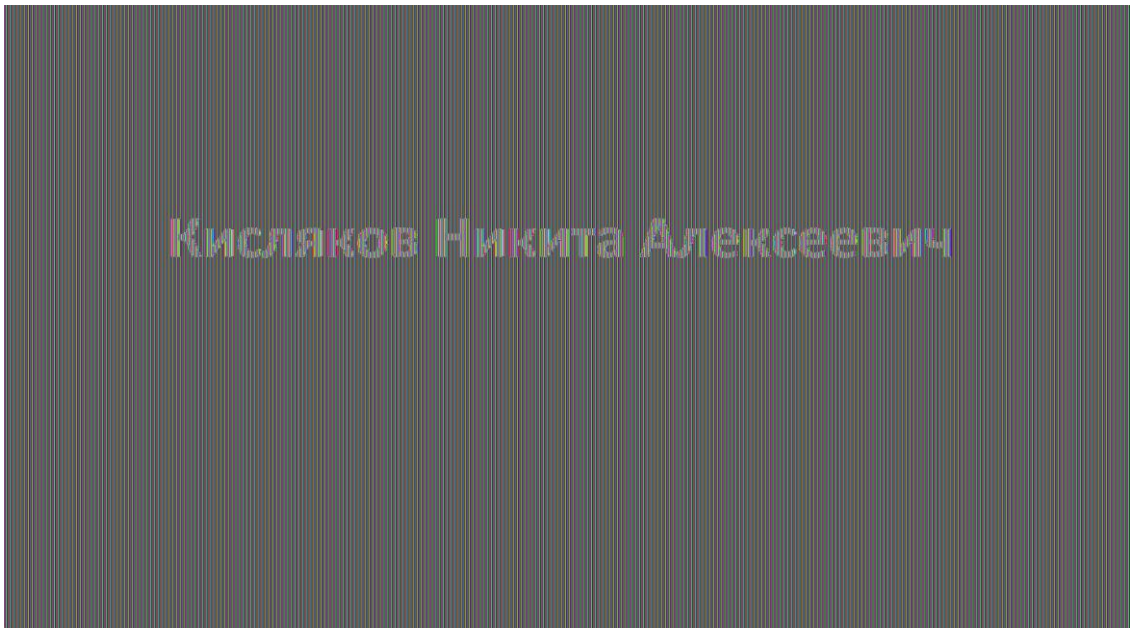


Рисунок 30 – ФИО

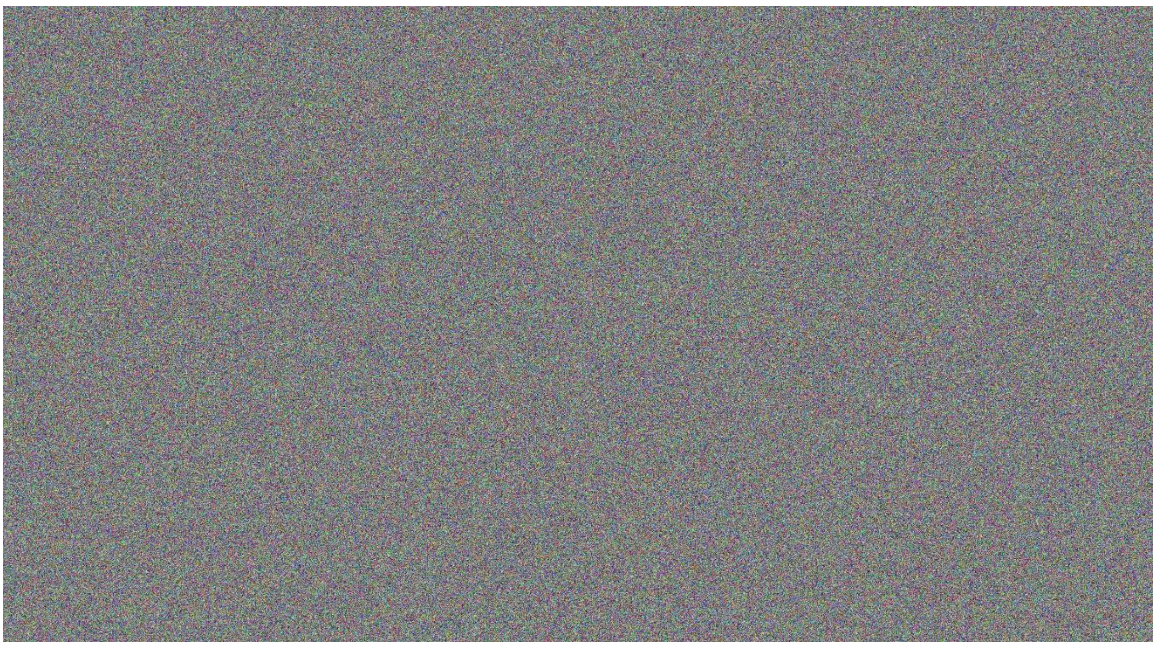


Рисунок 31 – ФИО

По полученным изображениям видно, что для шифрования картинок режим ECB не подходит, так как блоки шифротекста являются независимыми, и для одного и того же исходного блока будет получен один и тот же шифротекст. Из-за этого на картинке можно разглядеть текст.

При использовании режима CBC картинка полностью заполнилась шумом, поскольку теперь очередные блоки шифротекста будут зависеть от предыдущих блоков.

1.2.4 Таблица сравнений результатов сжатия исходного и шифрованных изображений

Следующим этапом, мы сожмем картинки, которые получили в прошлом пункте и занесем результат в таблицу 1.

Таблица №1

Картинка	Исходный размер	Размер после сжатия	Степень сжатия
Исходная	9 805 878	19 283	99%
Режим ECB	9 805 880	43 367	99%
Режим CBC	9 805 880	9 818 437	0%

Из таблицы видно, что у исходной картинки и картинки, зашифрованной DES с режимом ECB, степень сжатия выше 95%. Это можно объяснить тем,

что на картинках имеется большое количество повторяющейся информации (фон). В то время как для картинки, зашифрованной DES с режимом CBC, степень сжатия равна 0%. Это объясняется наличием сильного шума.

1.3 Исследование 3-DES

1.3.1 Задание

1) Определить экспериментальным путем, по какой схеме работает реализация 3-DES в CrypTool. Сохранить подтверждающие скриншоты.

1.3.2 Основные параметры и обобщенные схемы шифров

Шифр 3-DES состоит в трехкратном применении обычного DES. Обобщенная схема шифра 3-DES показана на рисунке 32.

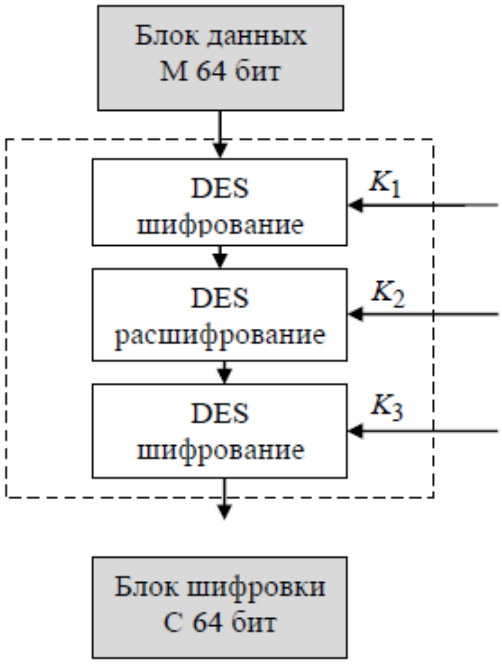


Рисунок 32 – Схема шифра 3-DES

Существуют 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 наиболее популярны версии шифра DES-EDE3 и DES-EDE2.

1.3.3 Схема реализации в 3-DES в CrypTool 1 и подтверждающие скриншоты

В CrypTool 1 используется ключ размером 128 бит, то делаем вывод что используется DES-EEE2 или DES-EDE2 на рисунке 33.

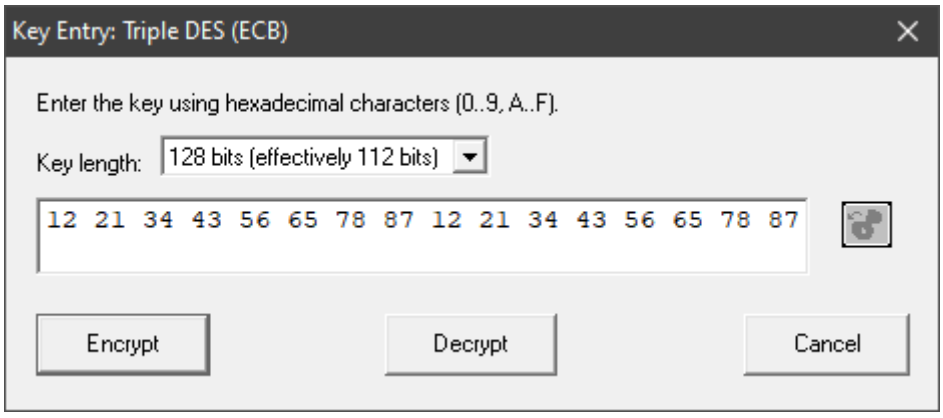


Рисунок 33 – Ключ размером 128 бит

Будем использовать следящий ключ «12 21 34 43 56 65 78 87 12 21 34 43 56 65 78 87». Для начала зашифруем файл с помощью 3-DES на рисунке 34.

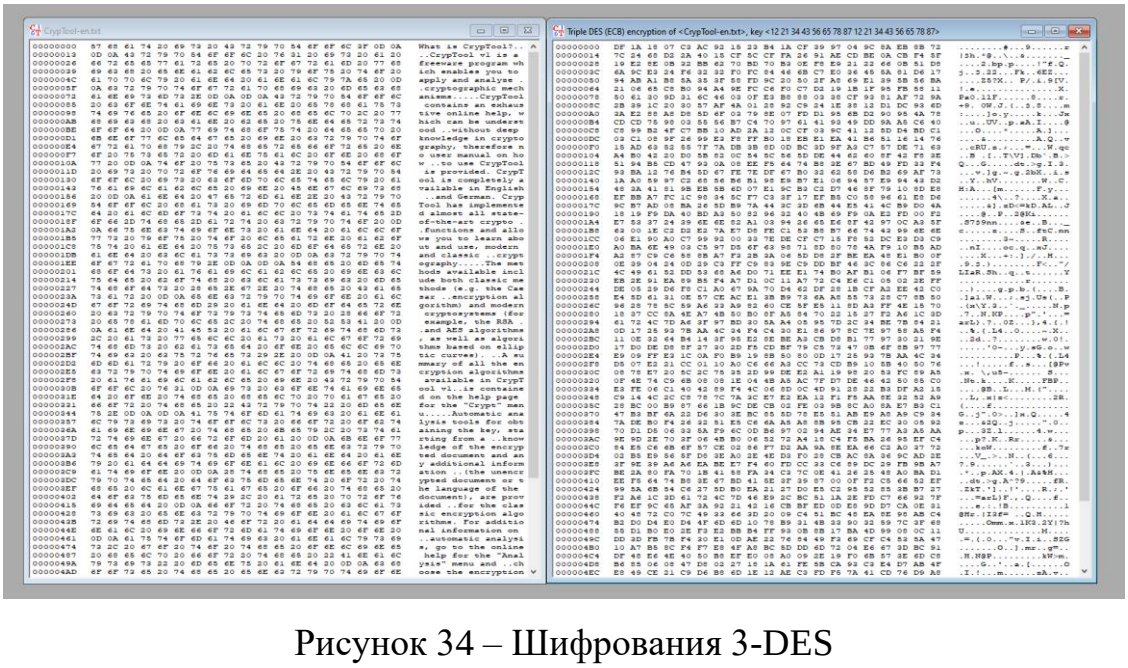


Рисунок 34 – Шифрования 3-DES

Будем использовать следящий ключ «12 21 34 43 56 65 78 87» и зашифруем файл с помощью DES.

Если сравнить с шифрованием 3-DES, то видно, что шифротексты не совпадают, следовательно версия DES-EEE2 не подходит, а значит CrypTool 1 использует версию DES-EDE2 на рисунке 35.

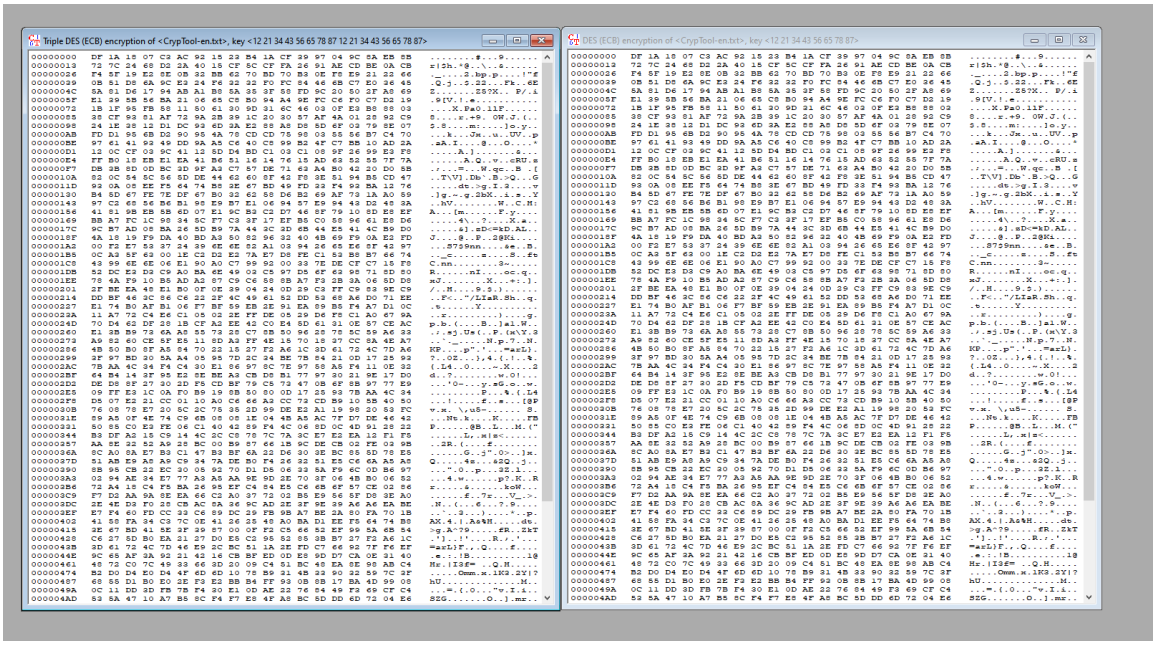


Рисунок 35 – Шифрования DES

2. Шифр AES

2.1 Исследование преобразований AES

2.1.1 Задание

1) Изучить преобразование шифра AES с помощью демонстрационного приложения из CrypTool 1: Indiv.Procedures -> Visualization… -> AES -> Rijndael Animation.

2) Провести наблюдение в потоковой модели шифра AES с помощью демонстрационного приложения из CrypTool 1 для 0-текста и 0-ключа: Indiv.Procedures -> Visualization… -> AES -> Rijndael Flow Visualisation.

2.1.2 Основные характеристики и описание AES с примерами скриншотов из демо-приложения

Шифр AES (в прежнем Rijndael) работает на основе перестановочно-подстановочной сети (SP-сеть). Обобщенная схема работы алгоритма продемонстрирована на рисунке 36.

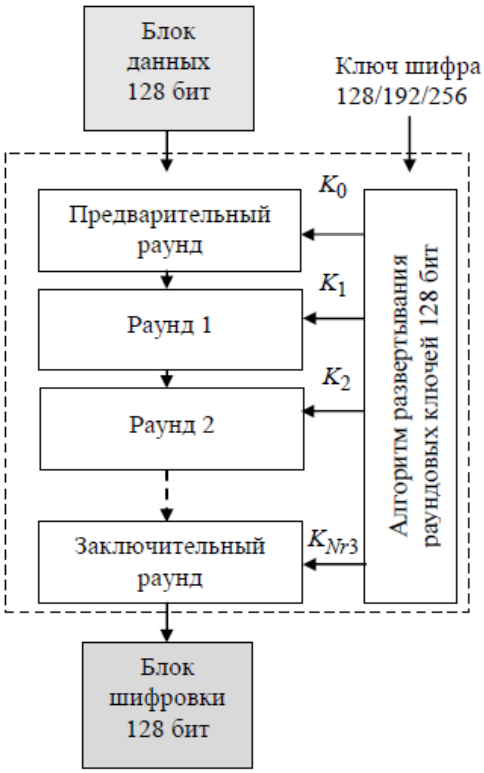


Рисунок 36 – Обобщенная схема работы алгоритма

Шифр принимает на вход блок данных 128 бит и ключ с вариантами длиной 128, 192 и 256 бит, выполняя раундовое преобразование 10, 12 и 14 раз соответственно.

В версии с наименьшей длиной ключа алгоритм AES получает на вход блок открытого текста размером 128 бит (16 байт) и ключ того же размера. Значения блока записываются в столбцы матрицы состояний размером 4 × 4 байт.

Процедура расширения ключей ExpandKey создает последовательно (слово за словом) 128-битные раундовые ключи от единственного входного ключа шифра.

После того как сформированы раундовые ключи, начинается раундовая обработка матрицы состояний. В каждом раунде алгоритма выполняются следующие преобразования, показанные на рисунке 37:

1. Столбцы матрицы состояний складываются с ключом шифра операцией XOR.

2. Полученная матрица состояний проходит через преобразование подстановки SubBytes.

3. Циклический сдвиг влево всех строк матрицы состояний выполняется преобразованием ShiftRows.

4. Смешивание столбцов матрицы состояний путем ее умножения на матрицу констант в конечном поле $GF(2^8)$ выполняется преобразованием MixColumn.

5. Сложение полученных столбцов матрицы состояний с раундовым ключом операцией XOR – преобразование AddRoundKey.

6. Действия 2–5 повторяются в каждом раунде, за исключением последнего.

7. Последний раунд не включает в себя смешивание столбцов.

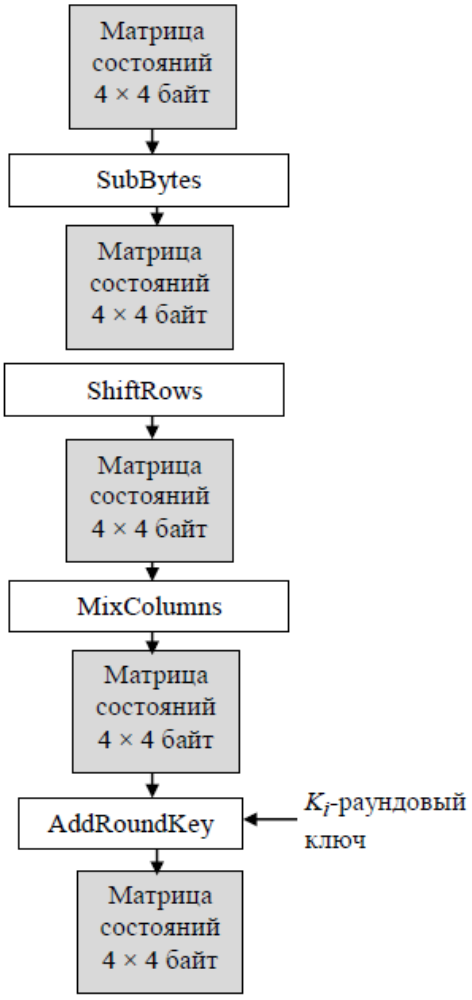


Рисунок 37 – Раундовые преобразования

Расшифровывание выполняется применением обратных операций и подстановкой раундовых ключей в обратной последовательности.

Теперь изучим преобразования AES с помощью демонстрационного приложения из CrypTool 1. В демонстрационном варианте используется AES-128, то есть с 128-битным ключом и 10 раундами. Обобщенная схема шифрования AES представлена на рисунке 38.

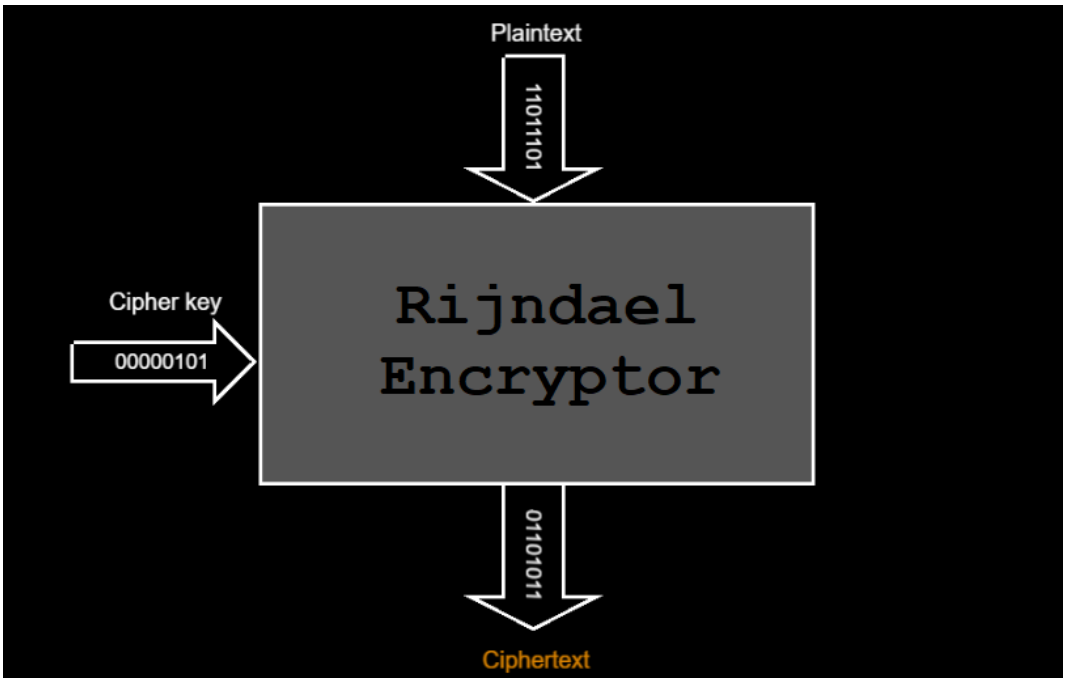


Рисунок 38 – Обобщенная схема шифрования

На вход шифру поступают 128-битный блок данных и 128-битный ключ, которые представлены в виде матрицы 4 × 4, где в каждой ячейке содержится 1 байт, как продемонстрировано на рисунке 39.

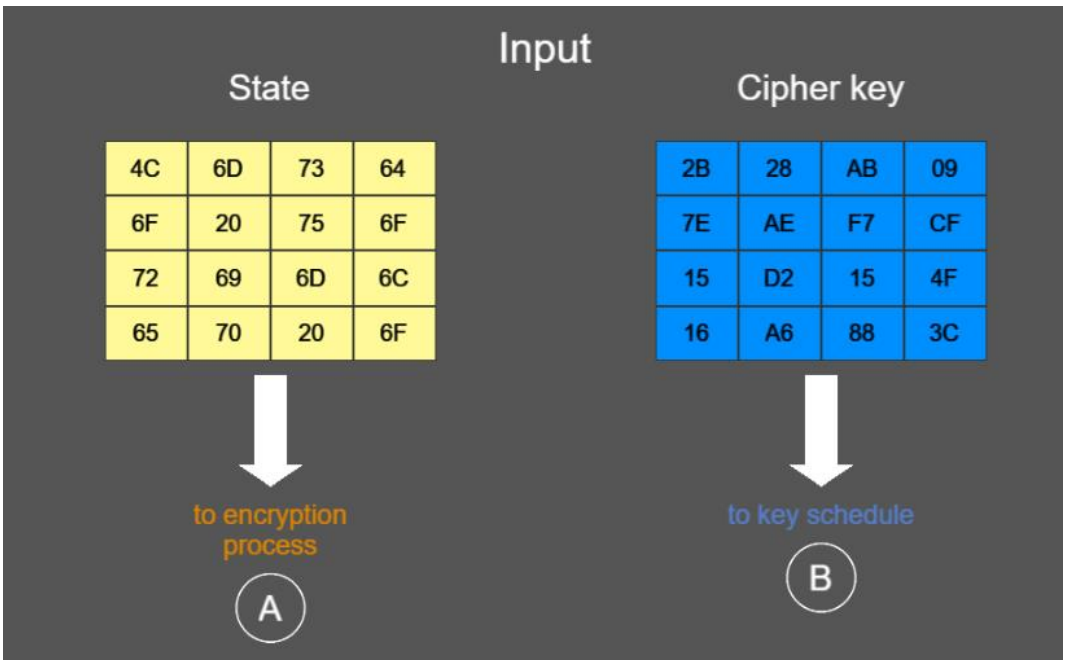


Рисунок 39 – Входные данные шифра

Матрица состояний подается на шифрование, а ключ – на процедуру генерации раундовых ключей. В самом начале выполняется преобразование AddRoundKey с исходным ключом и матрицей состояний. Следующие десять раундов используют раундовые ключи. Процесс шифрования состоит из 10 раундов. Каждый раунд, кроме последнего, состоит из четырех последовательных преобразований: SubBytes, ShiftRows, MixColumns, AddRoundKey. Последний раунд состоит только из трех последовательных преобразований: SubBytes, ShiftRows, AddRoundKey. Описанная структура показана на рисунке 40.

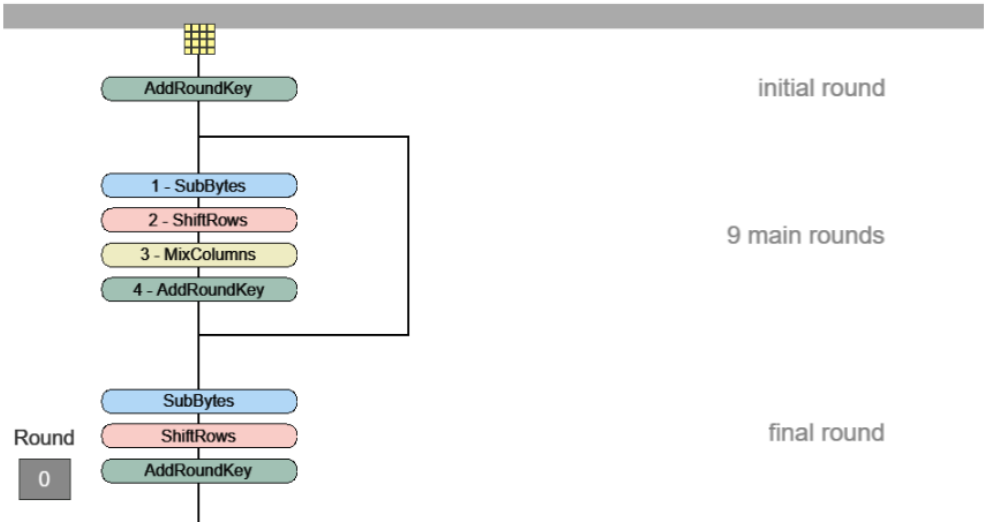


Рисунок 40 – Структура шифрования

Теперь рассмотрим более подробно раундовые преобразования. Первое преобразование – SubBytes. Здесь осуществляется замена байта матрицы состояний при помощи таблицы замен, как представлено на рисунке 41.

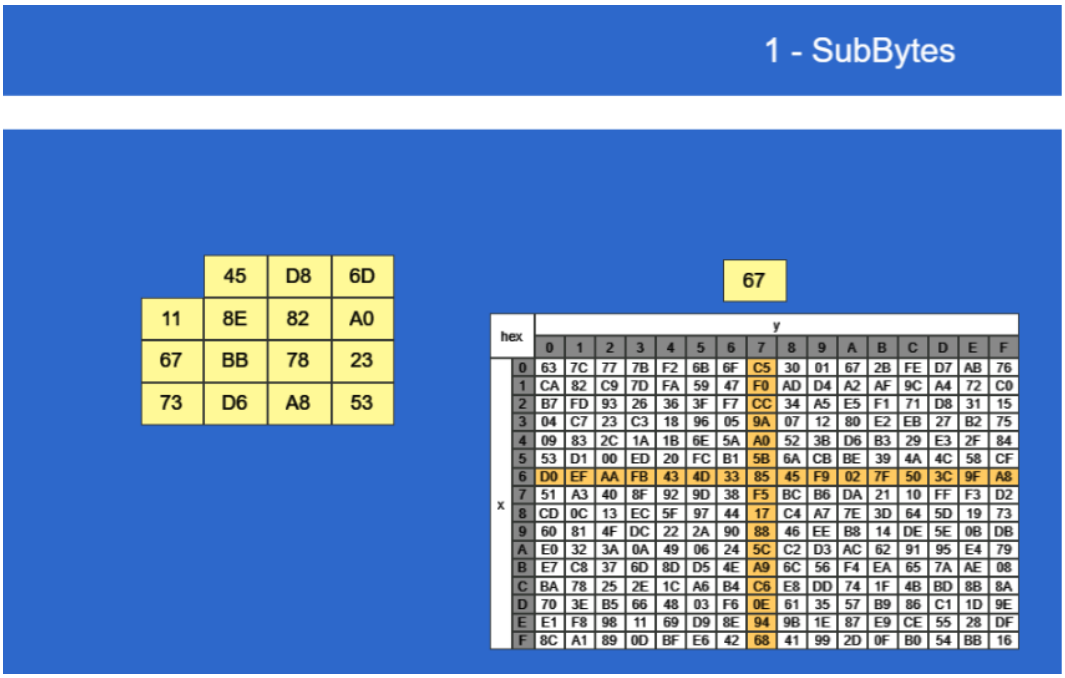


Рисунок 41 – Преобразование SubBytes

Далее идет преобразование – ShiftRows. Здесь происходит побайтовый сдвиг строк матрицы состояний. Размер сдвига определяется индексом строки, как продемонстрировано на рисунках 42, 43, 44, 45.

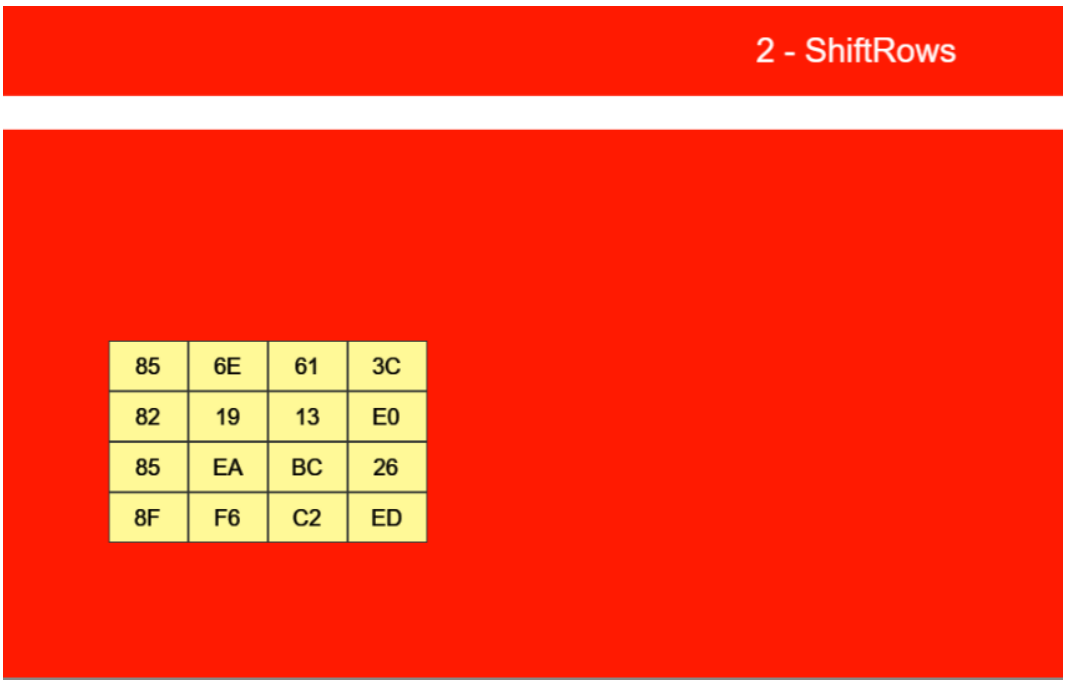


Рисунок 42 – Результат побайтового сдвига первой строки

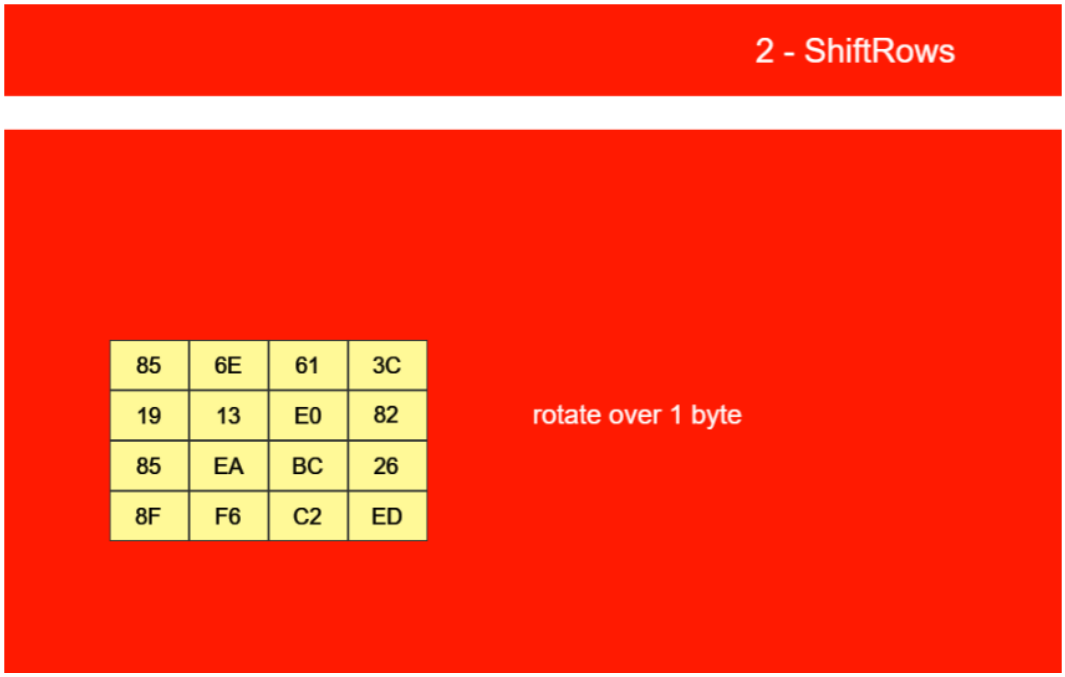


Рисунок 43 – Результат побайтового сдвига второй строки

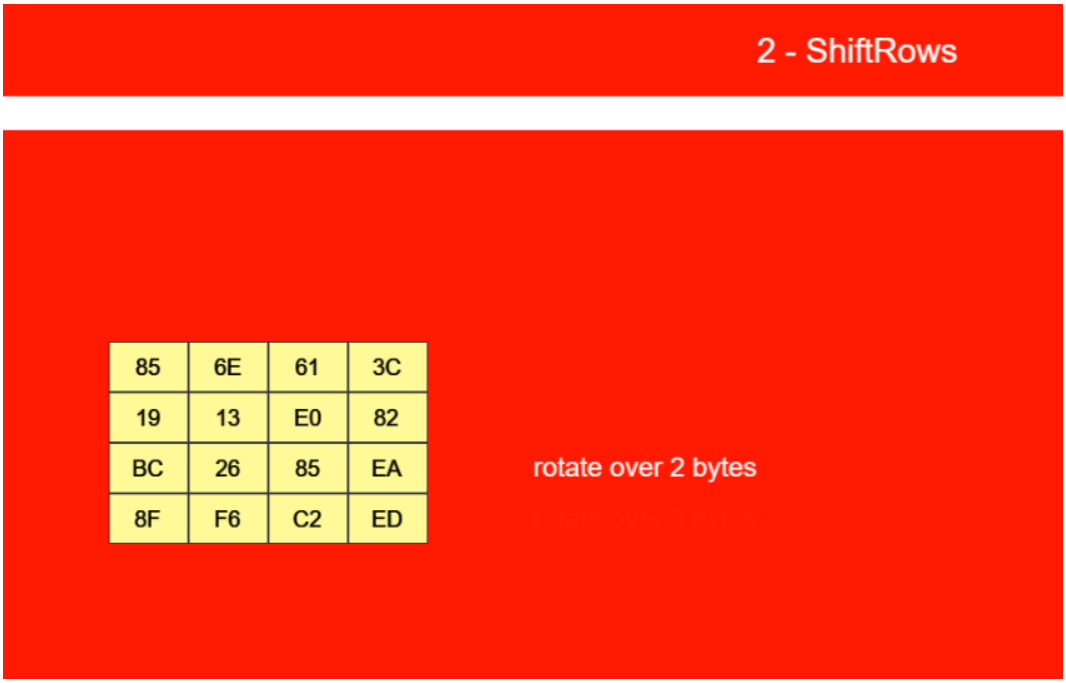


Рисунок 44 – Результат побайтового сдвига третьей строки

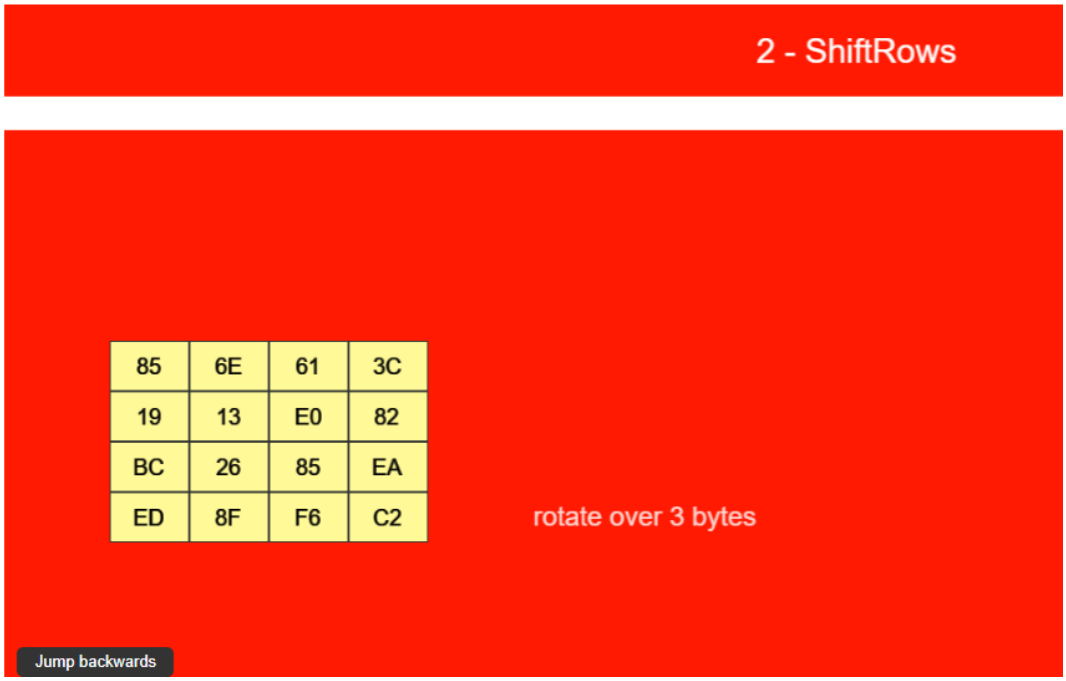


Рисунок 45 – Результат побайтового сдвига четвертой строки

Третье преобразование – MixColumns. Здесь происходит смешивание столбцов матрицы состояний путем умножения столбцов на матрицу констант в конченом поле $GF(2^8)$, как показано на рисунке 46.

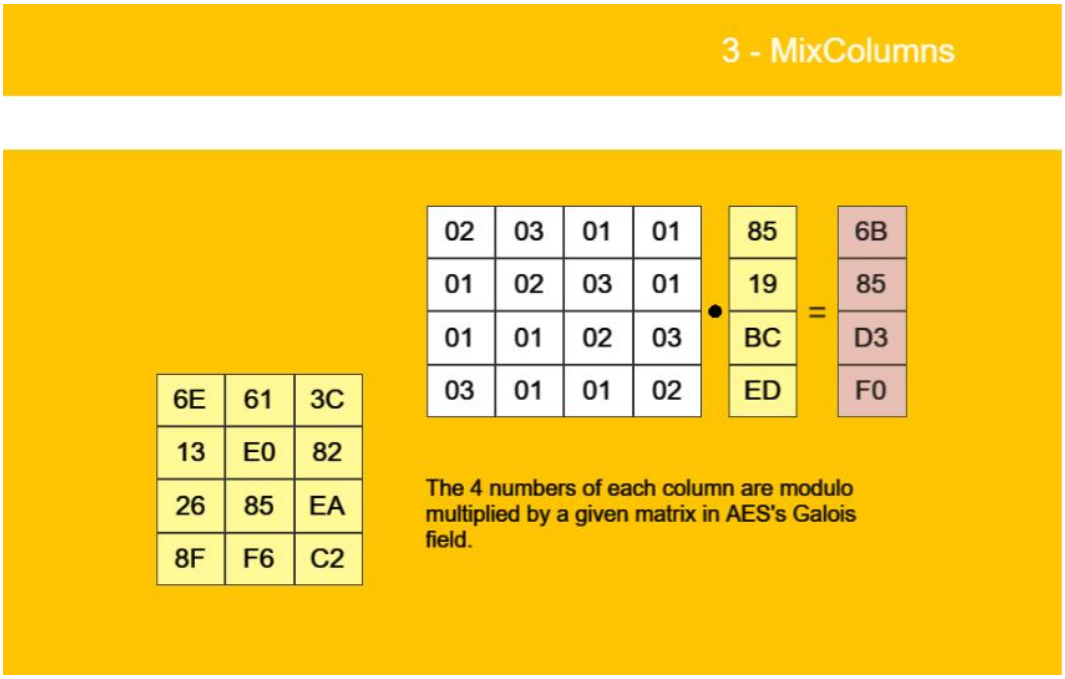


Рисунок 46 – Умножения столбца на матрицу констант

Конечный результат преобразования представлен на рисунке 47.

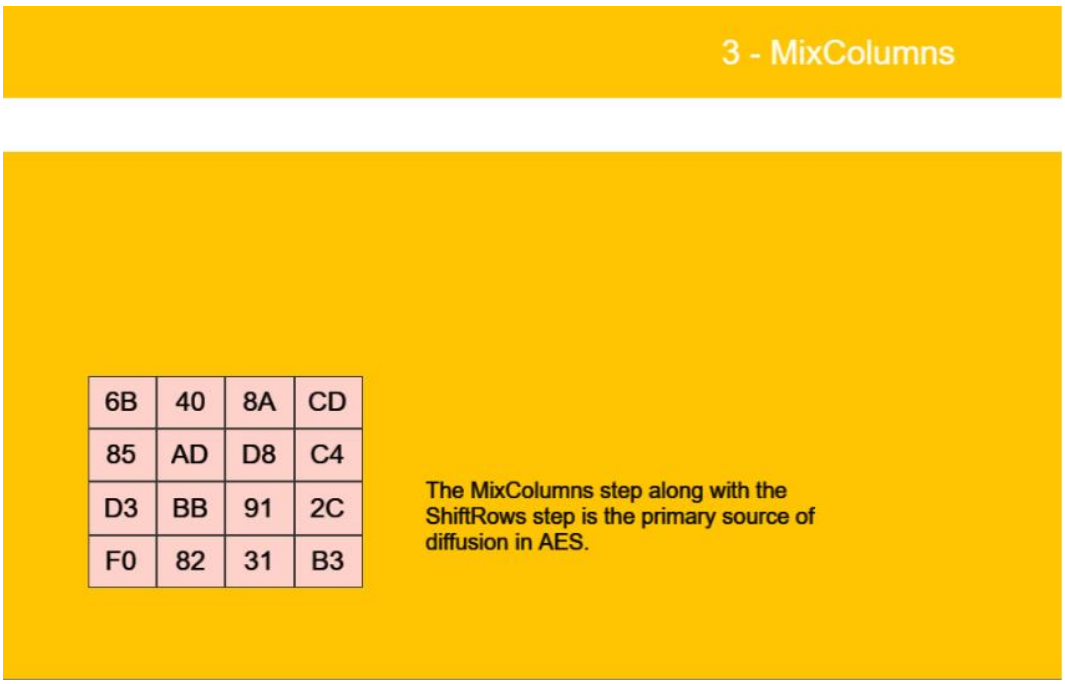


Рисунок 47 – Конечный результат преобразования MixColumns

Последним преобразованием является AddRoundKey. Здесь происходит сложение по модулю 2 столбцов матрицы состояний и раундового ключа, как продемонстрировано на рисунке 48.

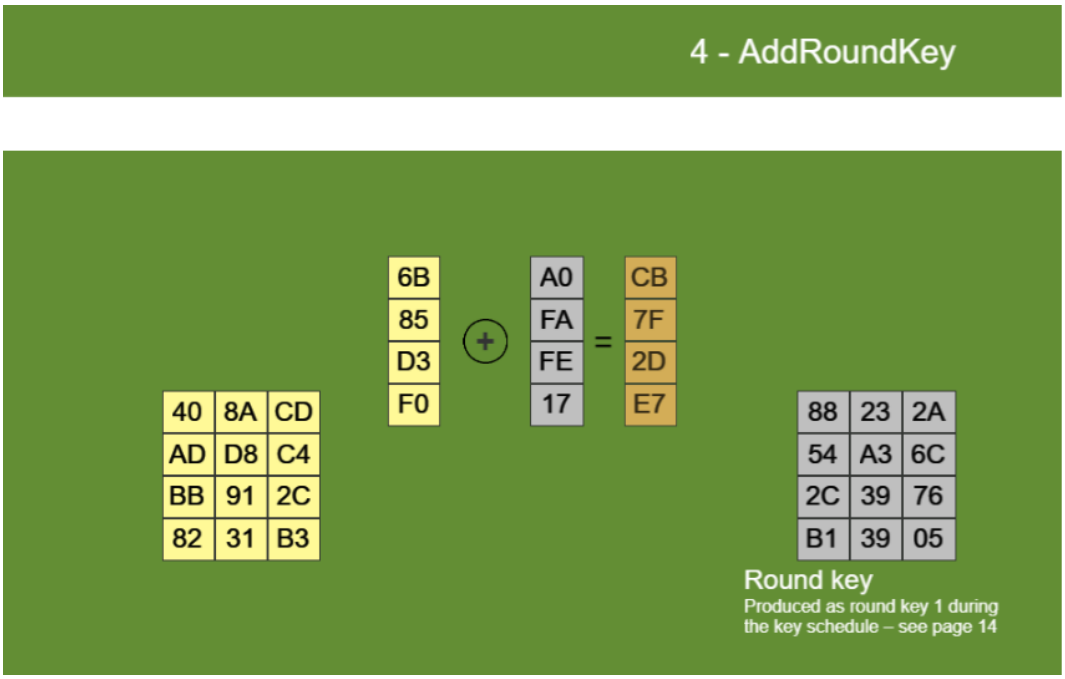


Рисунок 48 – Сложение по модулю 2 столбцов

Конечный результат преобразования показан на рисунке 49.

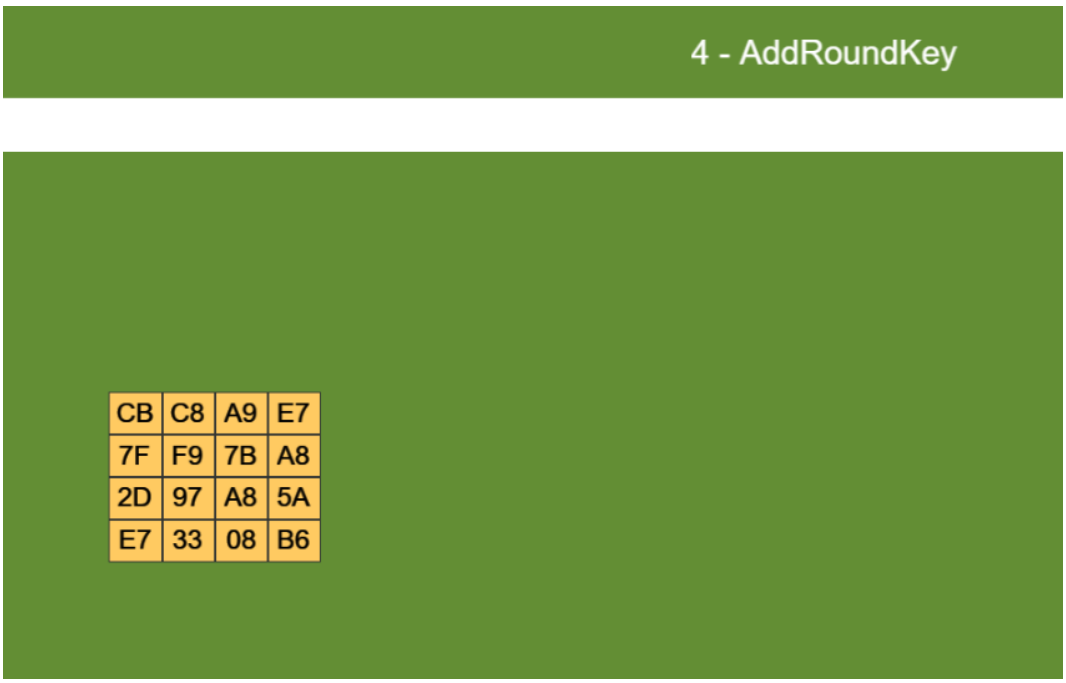


Рисунок 49 – Конечный результат преобразования AddRoundKey

Результаты раундовых преобразований представлены на рисунках 50, 51.

Рисунок 50 – Результаты первых пяти раундов

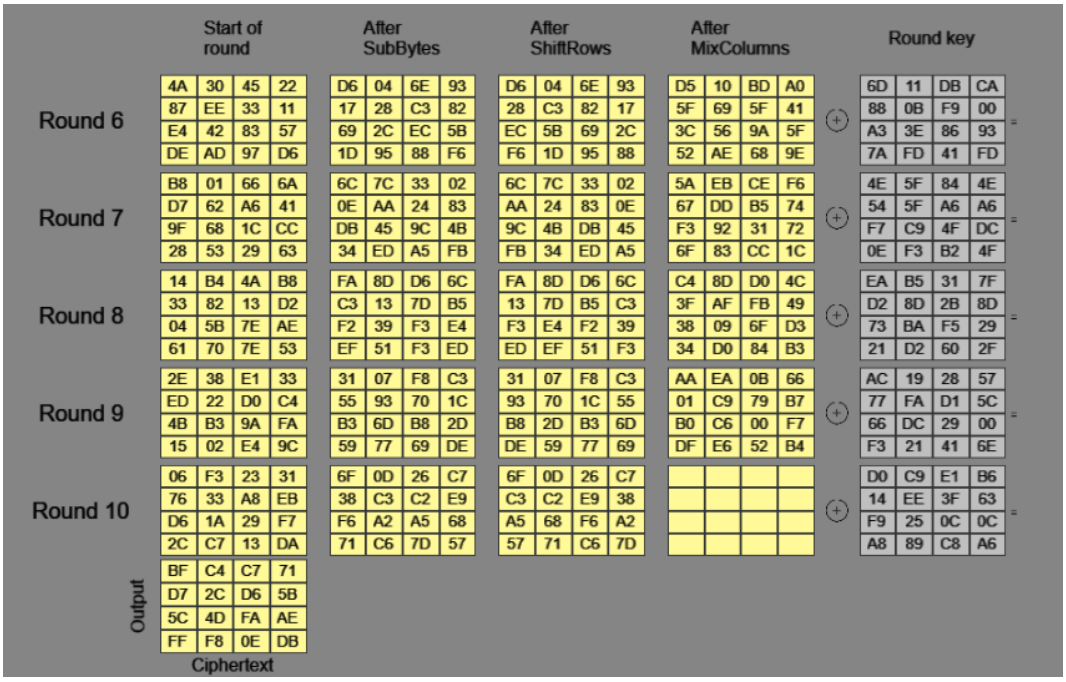


Рисунок 51 – Результаты оставшихся раундов

Далее рассматривается генерация 11 раундовых ключей (начальный раунд, 9 основных раундов, финальный раунд). Для генерации ключа берется последний столбец предыдущего ключа, к нему применяются преобразования RotWord (циклический сдвиг вверх) и SubBytes (замена байтов по таблице замены), после чего результат складывается по модулю 2 со столбцом, стоящим на 4 позиции позади, и соответствующим столбцом матрицы Rcon, как продемонстрировано на рисунках 52, 53, 54.

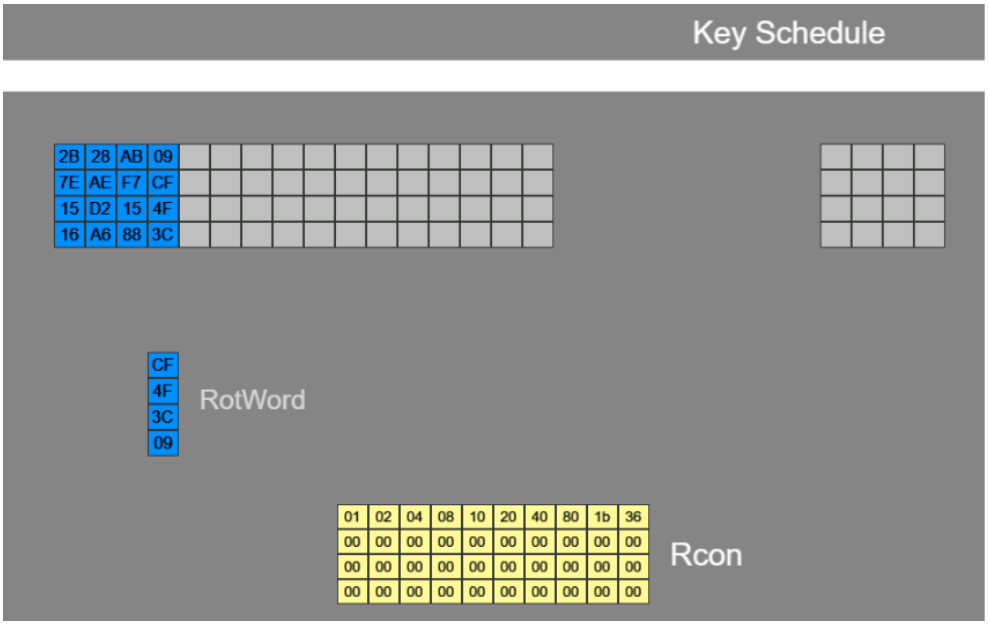


Рисунок 52 – Преобразование RotWord

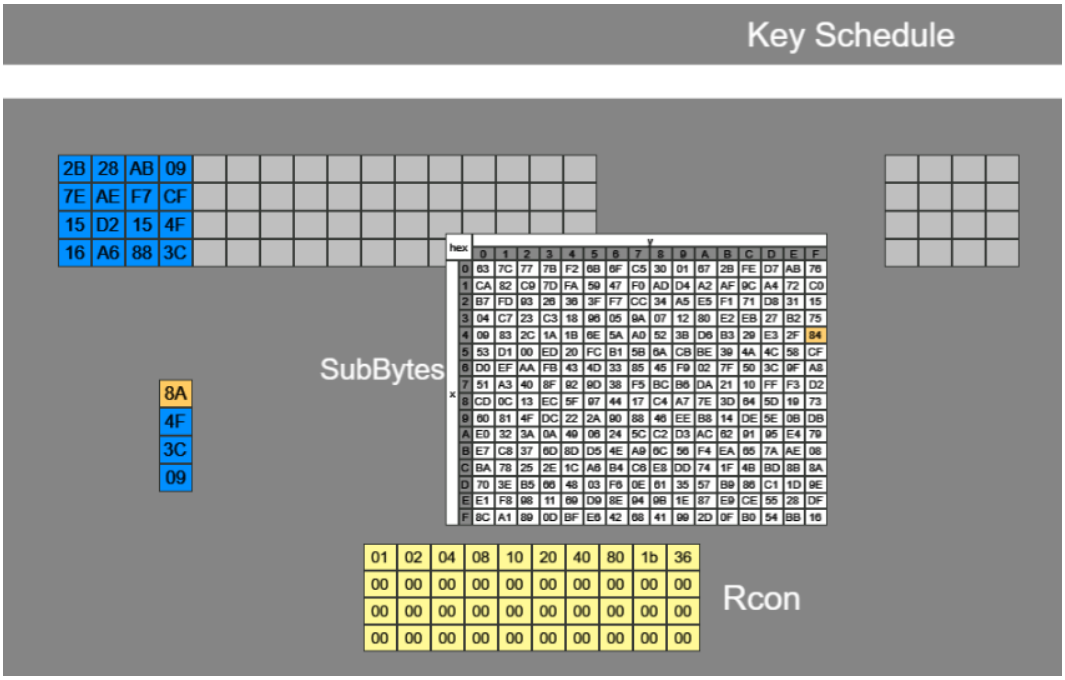


Рисунок 53 – Преобразование SubBytes с ключом

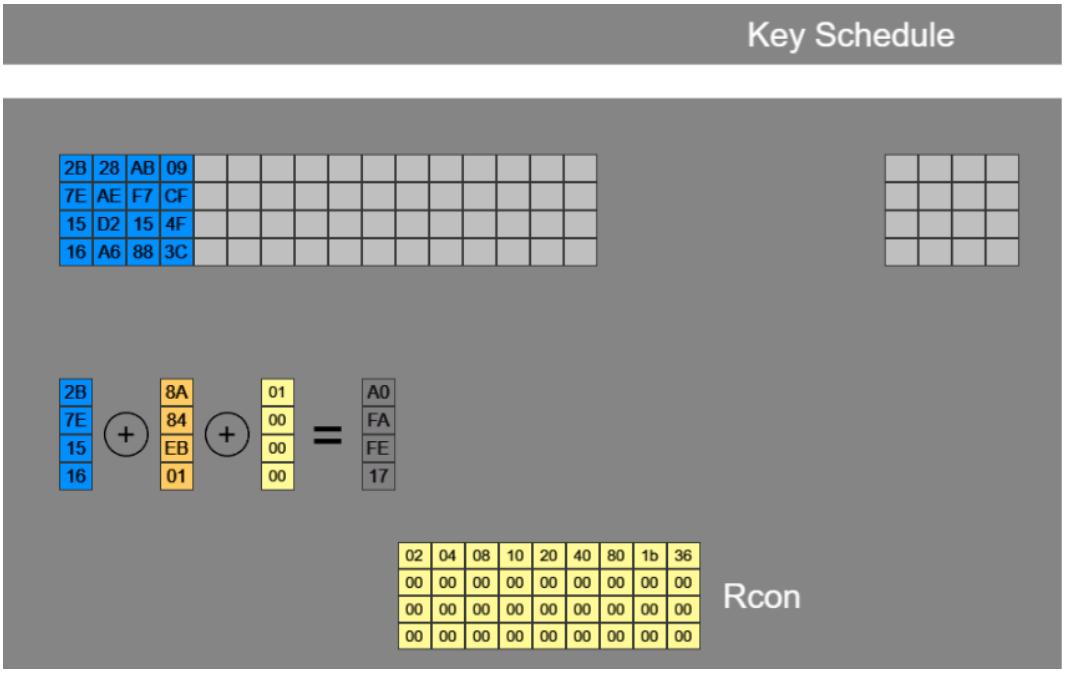


Рисунок 54 – Полученный столбец ключа

Оставшиеся три столбца ключа получаются сложением по модулю 2 с последними тремя столбцами предыдущего ключа, как показано на рисунке 55.

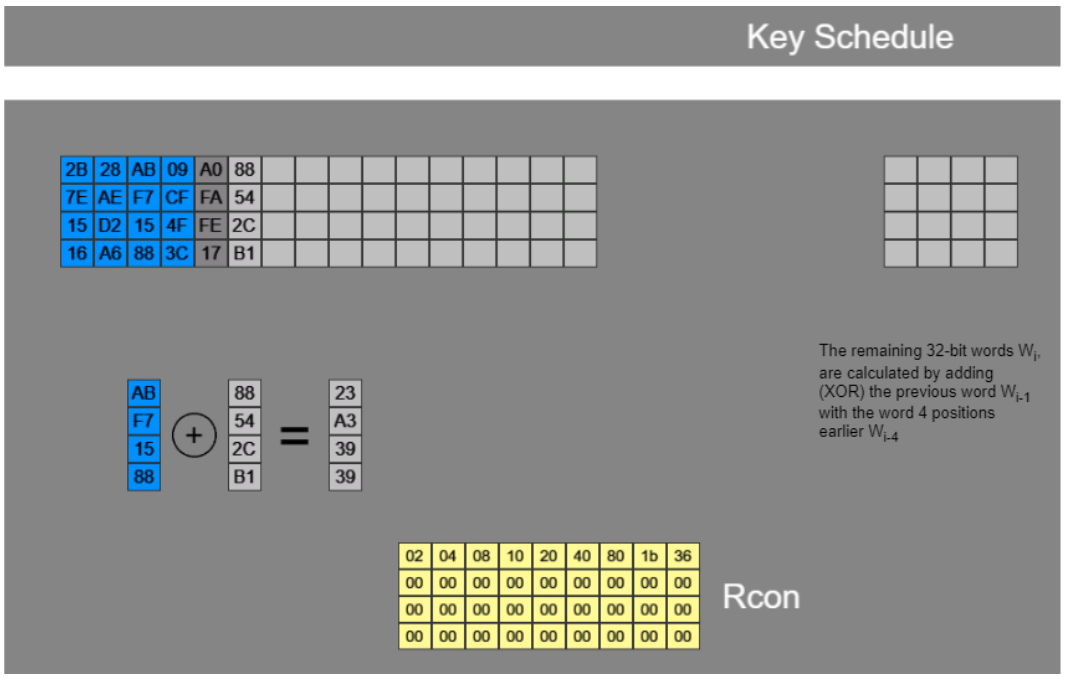


Рисунок 55 – Генерация оставшихся трех столбцов ключа

Остальные ключи генерируются по такому же принципу. На этом этапе демонстрация заканчивается. Результат генерации ключей представлен на рисунке 56.

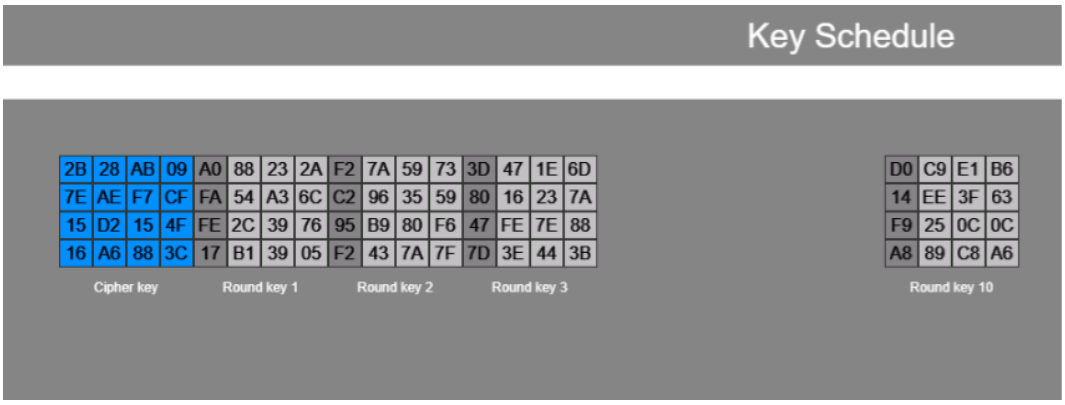


Рисунок 56 – Результат генерации ключей

2.1.3 Скриншоты наблюдений потоковой модели шифра и сопутствующие выводы

Проведем наблюдения в потоковой модели шифра AES при помощи демонстрационного приложения из CrypTool 1 для 0-текста и 0-ключа. Результаты наблюдений продемонстрированы на рисунках 57, 58, 59, 60, 61.

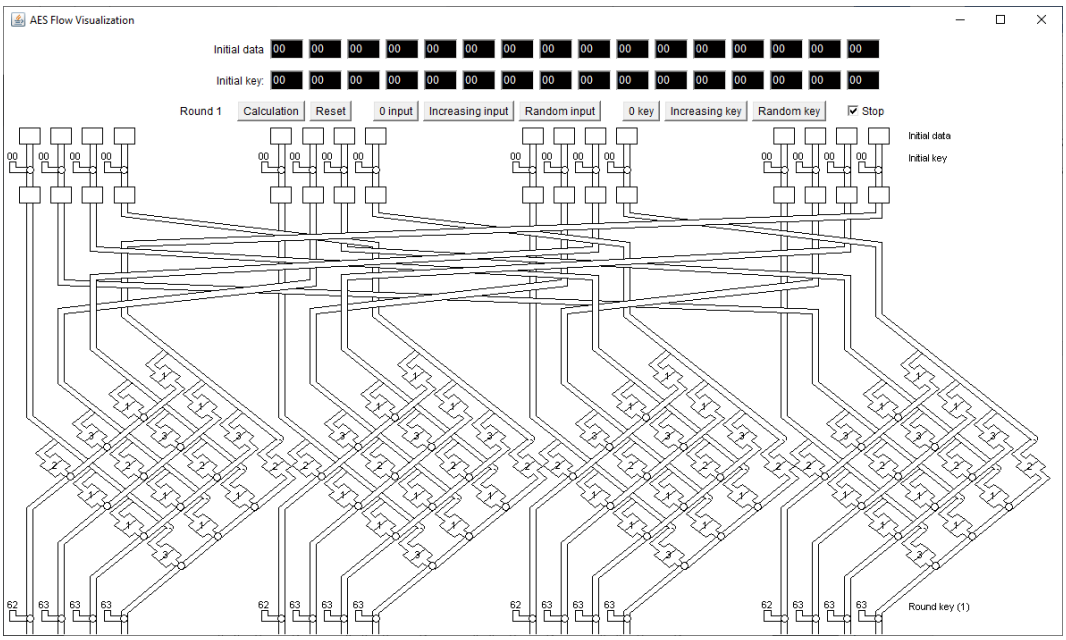


Рисунок 57 – Установка входных параметров

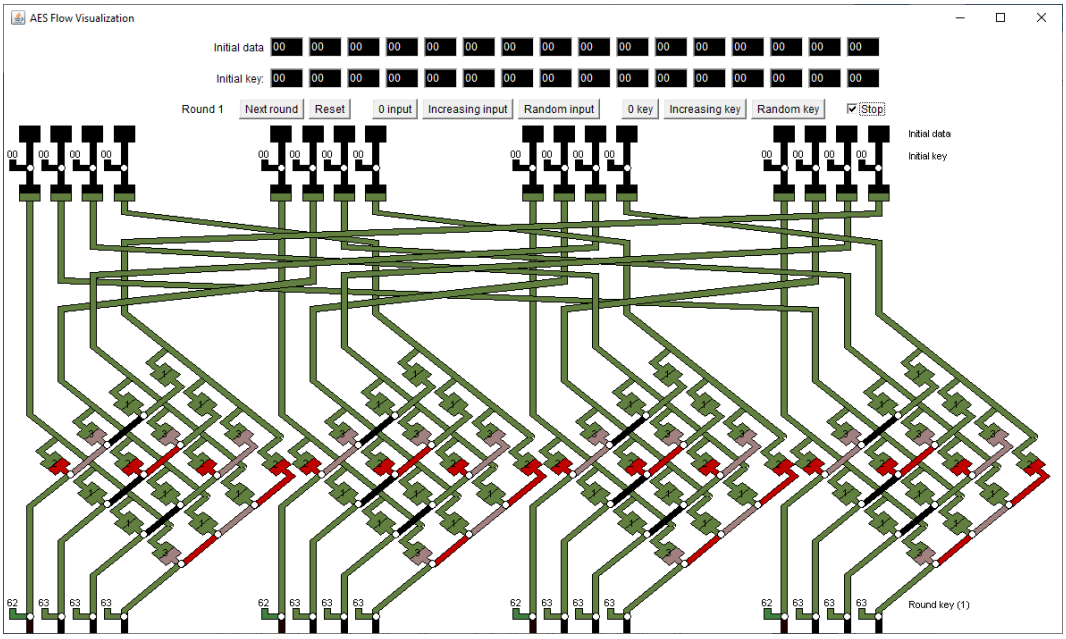


Рисунок 58 – Первый раунд

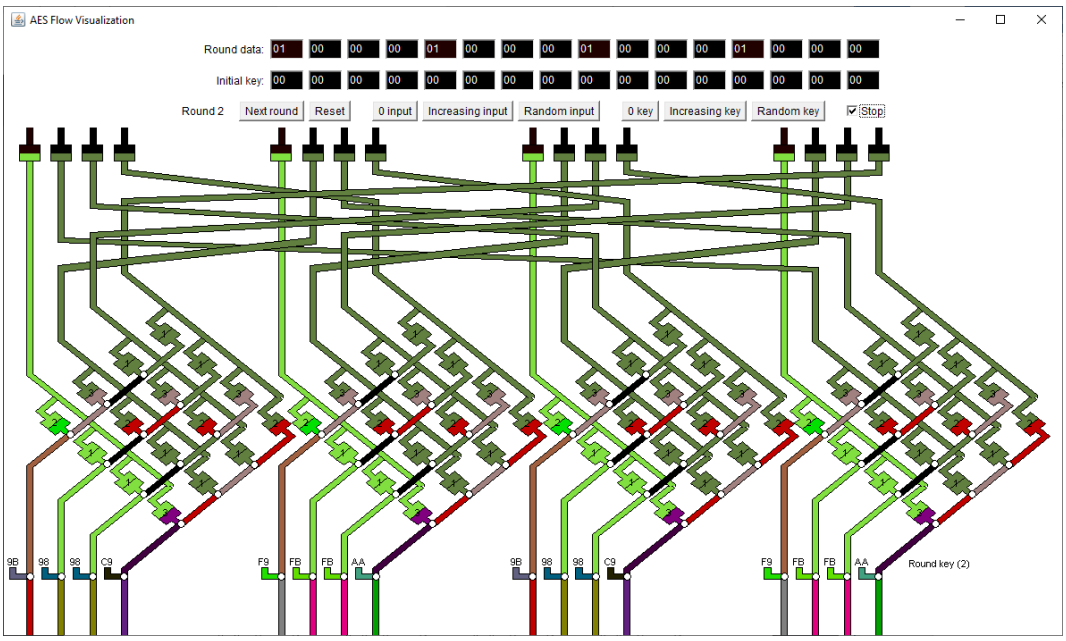


Рисунок 59 – Второй раунд

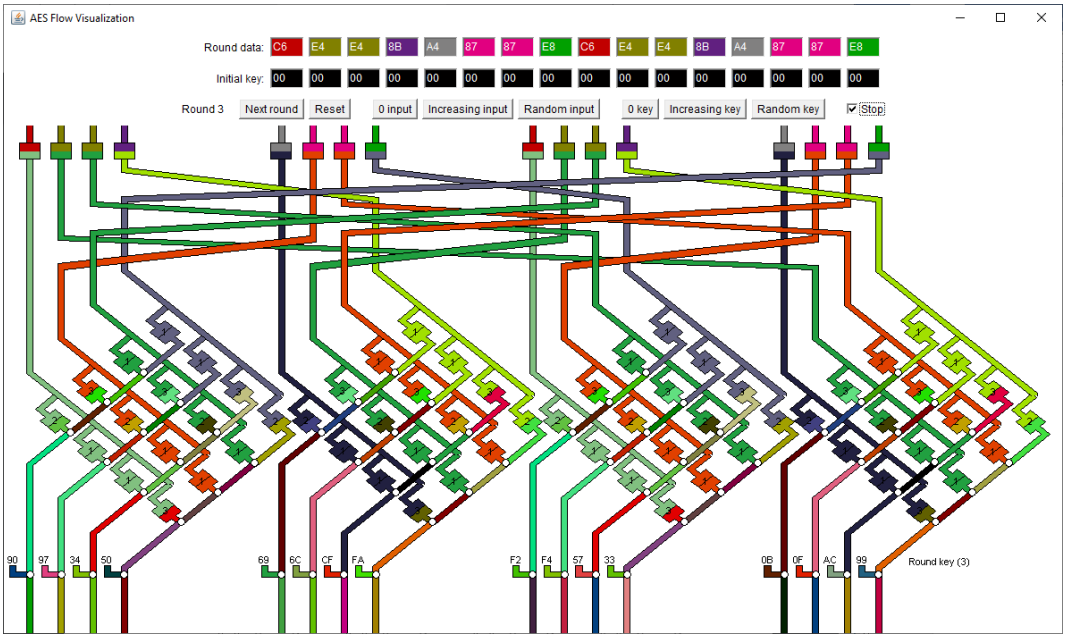


Рисунок 60 – Третий раунд

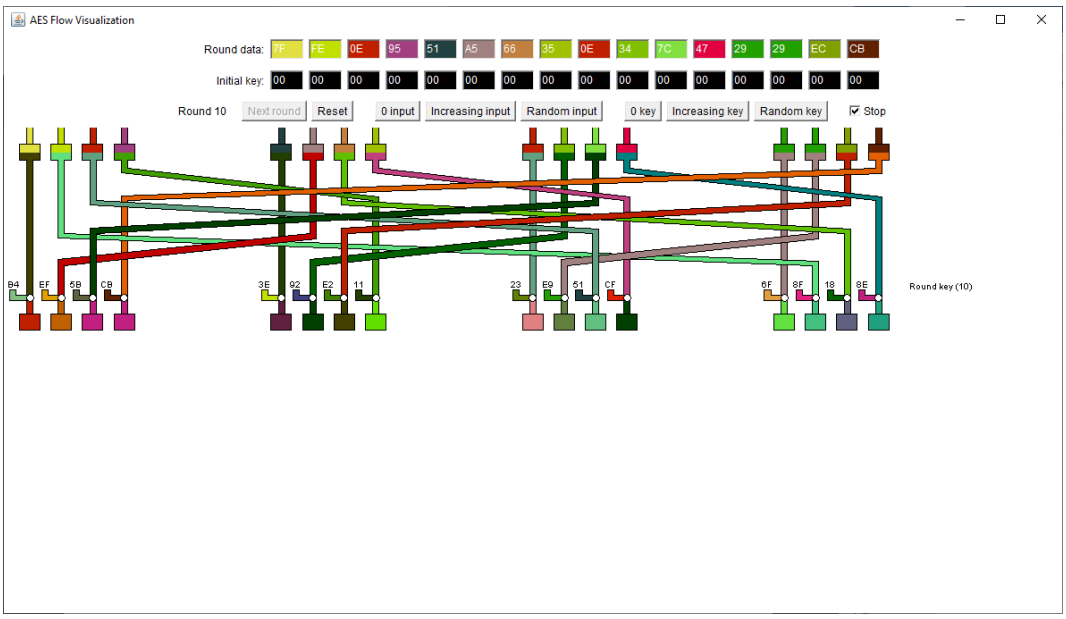


Рисунок 61 – Финальный раунд

Видно, что зашифрованный текст сильно поменялся при открытом 0-тексте.

2.2 Атака предсказанием дополнения на шифр AES в режиме CBC (Padding Oracle Attack)

2.2.1 Задание

1) Найти и запустить шаблон атаки в CrypTool 2: Padding Oracle Attack on AES.

2) Подготовиться к атаке теоретически, т.е. изучить: комментарии к шаблону; действия атакующего злоумышленника.

3) Внедрить во второй блок исходного текста коды символов своего имени.

4) Выполнить 3 фазы атаки и сохранить итоговые скриншоты по окончании каждой фазы.

5) Убедиться, что атака удалась.

2.2.2 Исходные данные для экспериментов

Используем исходный ключ «NIKITA», а также секретный ключ «1234567812345678».

2.2.3 Шаблон атаки «Padding Oracle Attack» из CrypTool 2

На рисунке 62 представлен шаблон атаки «Padding Oracle Attack» из CrypTool 2.

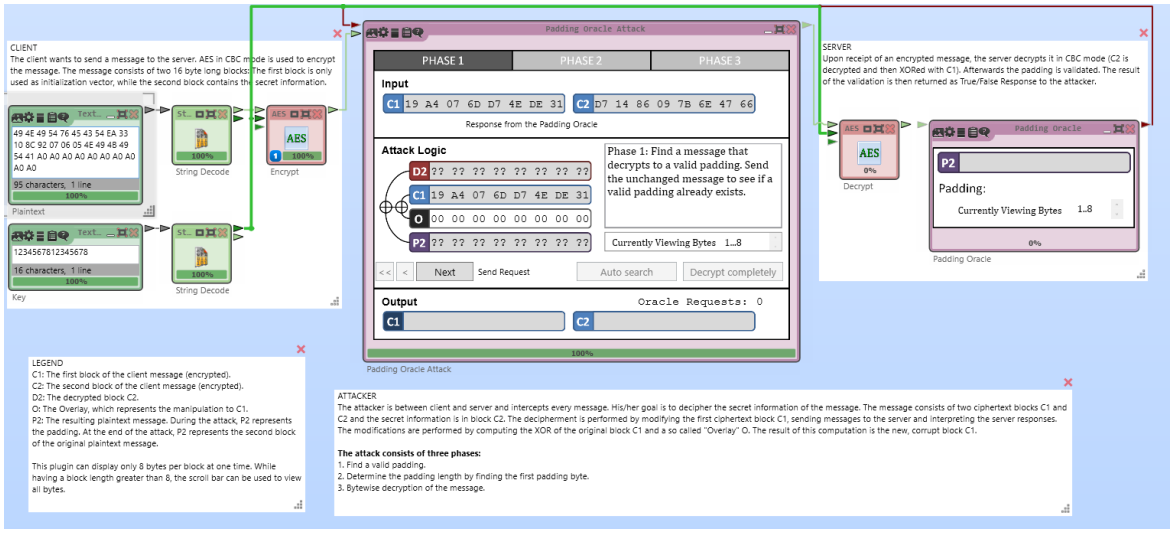


Рисунок 62 – Шаблон атаки

2.2.4 Описание атаки «Padding Oracle Attack»

Цель атаки – дешифровка блоков сообщения без знания ключа. При проведении этой атаки предполагается, что нарушитель может модифицировать и отправлять блоки зашифрованного сообщения серверу для расшифровки, а также распознавать ответы сервера о корректности дополнения последнего блока. Расшифровка сообщения нарушителем начинается с последнего блока шифротекста.

Рассмотрим расшифровку блока C_{i+1}.

1. Формируем R: все биты, кроме последнего, - случайные значения. Перебираем байт R_n от 0x00 до 0xFF, каждый раз посылая на сервер $[R \parallel C_{i+1}]$. Если при некотором R_n сервер «одобряет», то $T_n = 01$, $S_n = R_n \oplus 0x01$, $p_n = S_n \oplus c_n$. Схема первого этапа представлена на рисунке 63, где P_i – блок открытого текста; C_i – блок шифротекста; I_i – промежуточное состояние; K – ключ; D_K – функция расшифровки; T_i – формируемое дополнение.

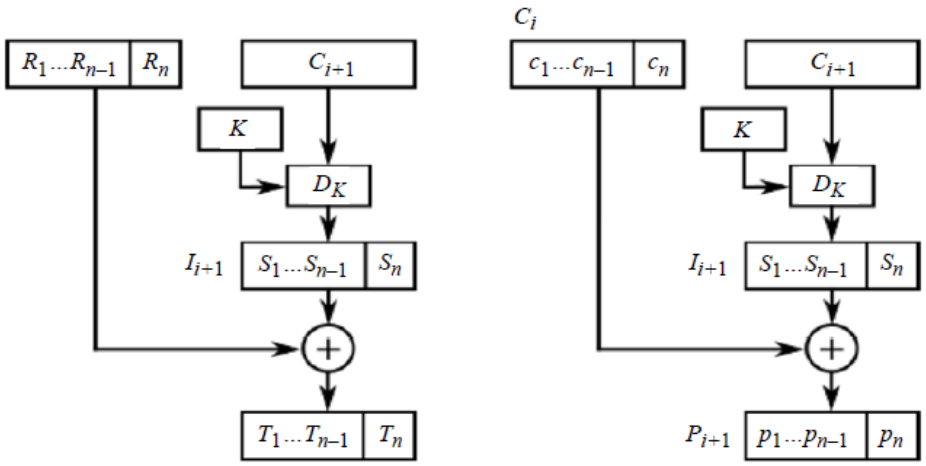


Рисунок 63 – Схема первого этапа атаки

2. Формируем R: все биты, кроме двух последних, - случайные значения. $R_n = S_n \oplus 0x02$, чтобы $T_n = 02$. Перебираем байт R_{n-1} от 0x00 до 0xFF, каждый раз посылая на сервер $[R \parallel C_{i+1}]$. Если при некотором R_{n-2} сервер «одобряет», то $T_{n-1} = 02$, $S_n = R_{n-1} \oplus 0x02$, $p_{n-1} = S_{n-1} \oplus c_{n-1}$.

На третьем шаге пытаемся получить дополнение 030303, на четвертом 04040404. После N шагов получаем блок p_{i+1} полностью.

В CrypTool 2 атака предсказанием дополнения реализована в три фазы:

1. Нахождение длины дополнения.

2. Подбор дополнения.

3. Расшифровка текста.

2.2.5 Результаты 3 фаз атак в виде итоговых скриншотов ПО

Мы используем шаблон Padding Oracle Attack on AES» из приложения CrypTool 2. На рисунке 64, мы внесли наши исходные данные.

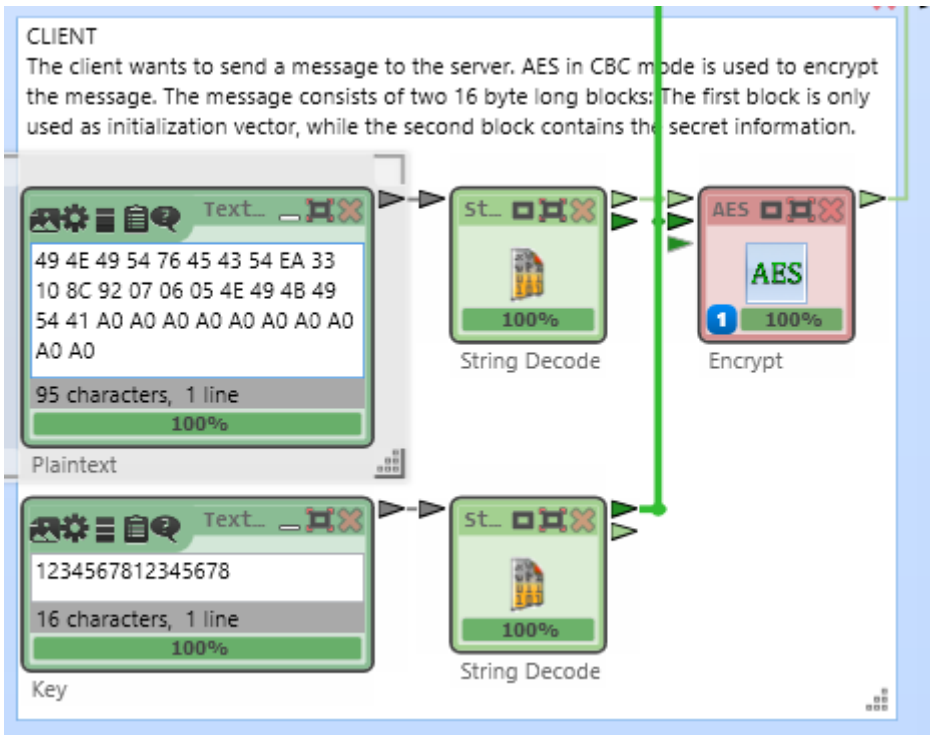


Рисунок 64 – Открытый текст

Выполним первую фазу атаки. Поскольку в изначальном тексте установлен корректный padding, то передавая шифротекст сервер сообщит о том, что padding исходного сообщения корректен. Результат выполнения показан на рисунке 65.

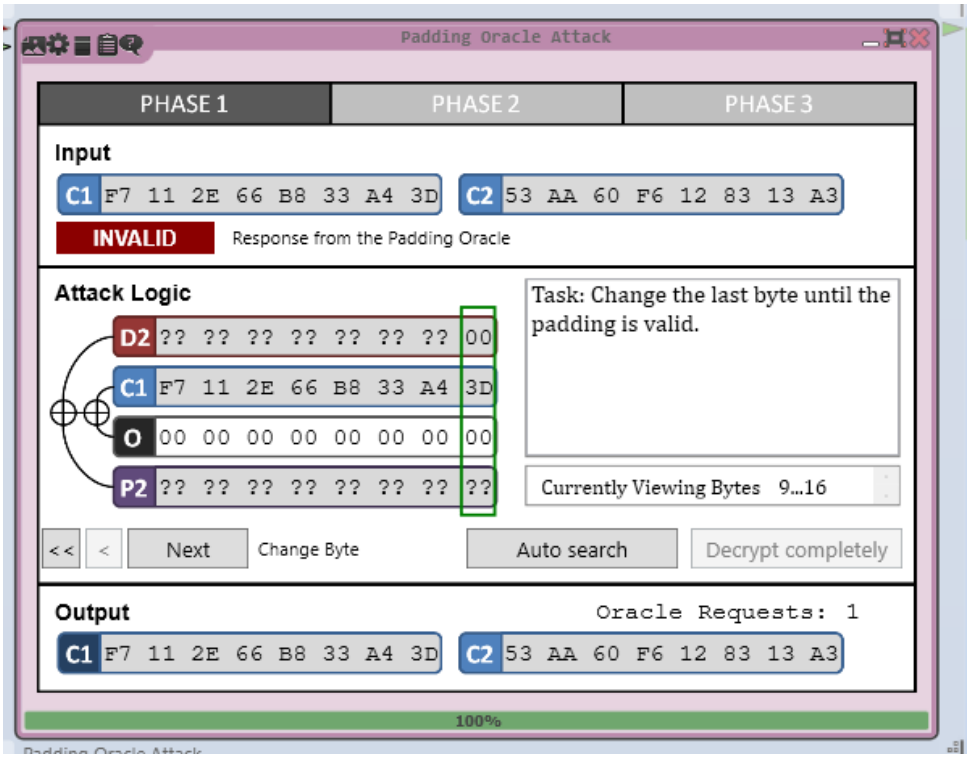


Рисунок 65 – Результат выполнения первой фазы атаки

На следующем этапе мы определили размер дополнения. Размер дополнения был определен верно – 10, как представлено на рисунке 66.

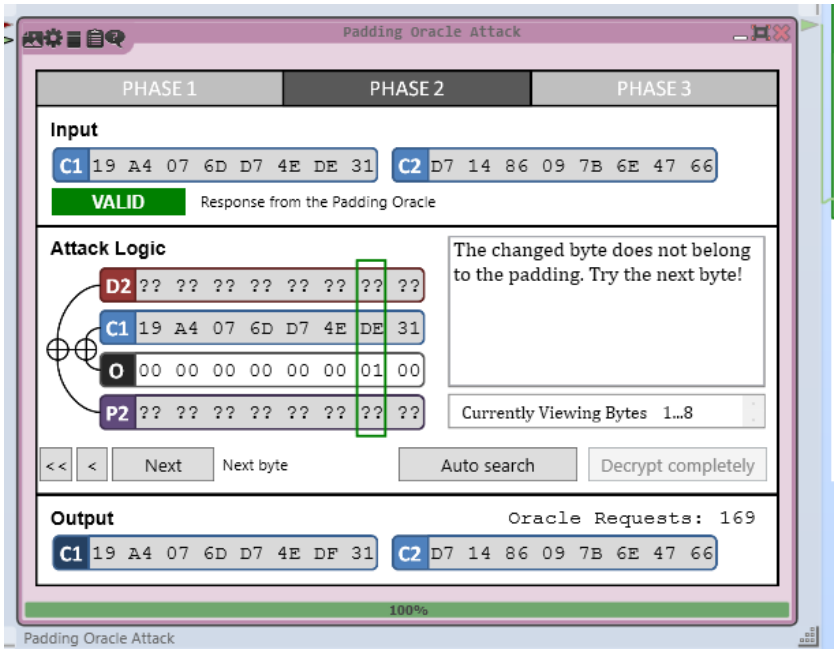


Рисунок 66 – Размер дополнения

Выполним третью фазу. Зная дополнение, мы можем найти байты блока D2 по информации о корректности дополнения, перебирая значения для байтов C1. После того, как мы найдем D2, можно легко найти P2 (P2 = C1 XOR D2). Результат выполнения третьей фазы продемонстрирован на рисунке 67.

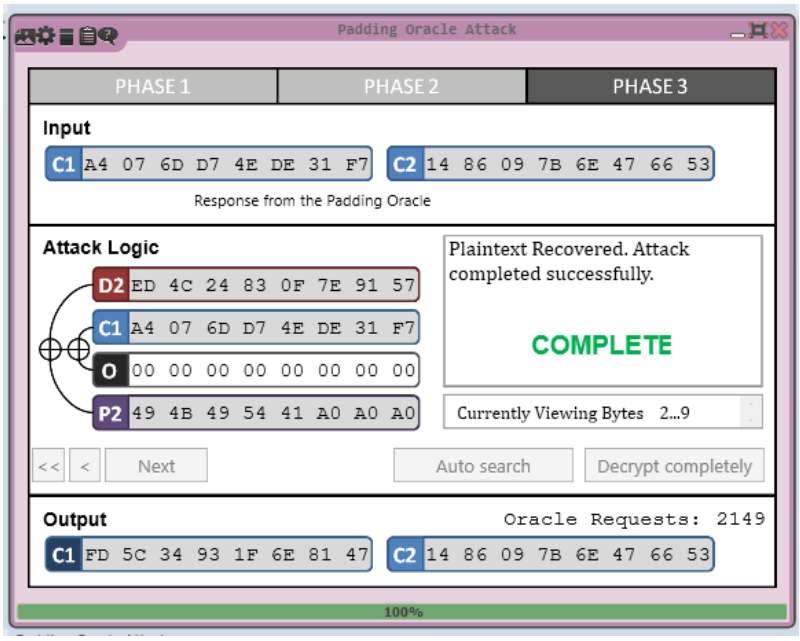


Рисунок 66 – Результат выполнения третьей фазы атаки

Выводы

1. DES

В результате выполнения данной лабораторной работы исследовал шифры DES, 3DES и получил практические навыки работы с ними.

1) Шифрование DES – шифрование блоками по 64 бита. Размер ключа 56 бит. Одним и тем же ключом шифруется каждый блок. Процесс шифрования состоит из 16 раундов, в каждом из которых используется свой раундовый ключ. Также в начале и в конце используются перестановки.

Режим ECB шифра DES работает независимо с каждым 64-битным блоком шифруемых данных. Но существует недостаток – одинаковые исходные блоки при шифровании дают одинаковые блоки шифротекста. При шифровании изображения шифром DES в режиме ECB зашифрованное изображение сохраняет детали. Сжатие зашифрованного изображения при этом достигает 99%, что находится на уровне сжатия исходного – 99%.

Режим CBC шифра DES перед запуском шифрования каждого очередного блока складывает его с предыдущим операцией XOR. Сжатие зашифрованного изображения при этом достигает 0%.

2) Шифр 3-DES основан на последовательном использовании DES с тремя или двумя независимыми ключами. Размер блока составляет 64 бит. Размер ключа составляет 112 или 168 бит.

Приложение CrypTool 1 использует следующую схему работы шифра 3-DES – DES-EDE2.

2. AES

AES – симметричный блочный шифр, использующий структуру «квадрат» и SP-сеть. Размер блока составляет 128 бит. Размер ключа может составлять 128, 196, 256 бит. В зависимости от размера ключа, количество раундов может составлять, соответственно, 10, 12 и 14. Каждый раунд состоит из 4 различных обратимых преобразований (подстановка, перемешивания строк, перемешивания столбцов, рандомизация). Для расшифрования операции производятся в обратном порядке.

Была изучена и проведена «Padding Oracle Attack» – атака предсказанием дополнения на шифр AES. В основе атаки лежит возможность перехвата и изменения блоков шифротекста, а также получение информации о корректности дополнения в последнем блоке шифротекста.