AES

Оглавление

[Шифровка 2](#_Toc1575004)

[SubBytes 4](#_Toc1575005)

[ShiftRows 5](#_Toc1575006)

[MixColumns 6](#_Toc1575007)

[AddRoundKey 7](#_Toc1575008)

[KeyExpansion 8](#_Toc1575009)

[Расшифровка 9](#_Toc1575010)

[InvSubBytes 11](#_Toc1575011)

[InvShiftRows 12](#_Toc1575012)

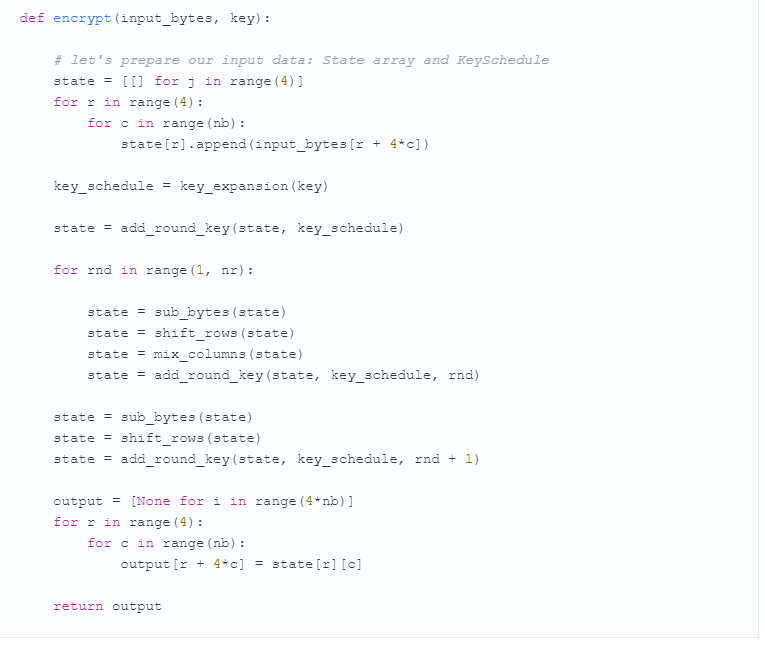
[InvMixColumns 13](#_Toc1575013)

[AddRoundKey 14](#_Toc1575014)

[Заключение 15](#_Toc1575015)

[Конечное поле или поле Галуа 16](#_Toc1575016)

# Шифровка

Advanced Encryption Standard – блочный, симметричный алгоритм шифрования.

Ключи: (128, 192, 256) бит.

State – промежуточный результат шифрования. Представляется в форма матрицы, имеющей 4 строки и Nb колонок. Каждая ячейка State содержит значение размером в 1 байт.

Nb – число столбцов (32-х битных слов), составляющих State. Регламентировано Nb = 4

Nk – длина ключа в 32-х битных словах. Для AES Nk = 4,6,8. Используем Nk = 4.

Nr – количество раундов шифрования. В зависимости от длины ключи Nr=10, 12, 14.

Последовательность действий:

Инициализация (AddRoundKey())

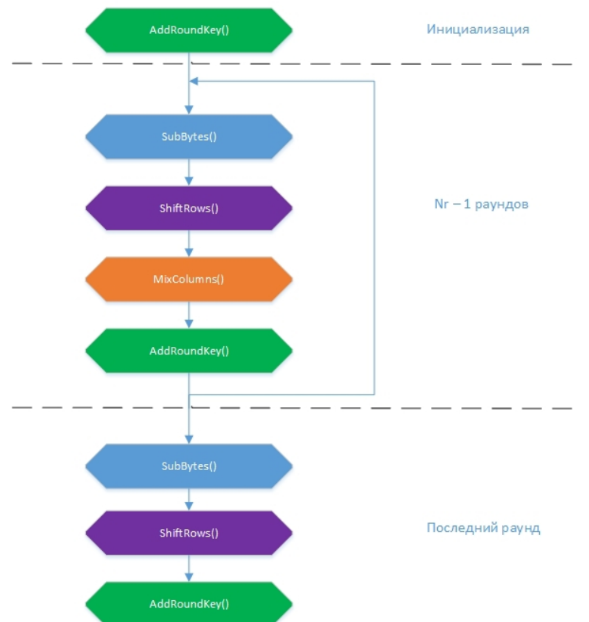
Nr – 1 раундов ((SubBytes(), ShiftRows(), MixColumns(), AddRoundKey())

Последний раунд (AddRoundKey())

Алгоритм оперирует с байтами, считая их элементами конечного поля .

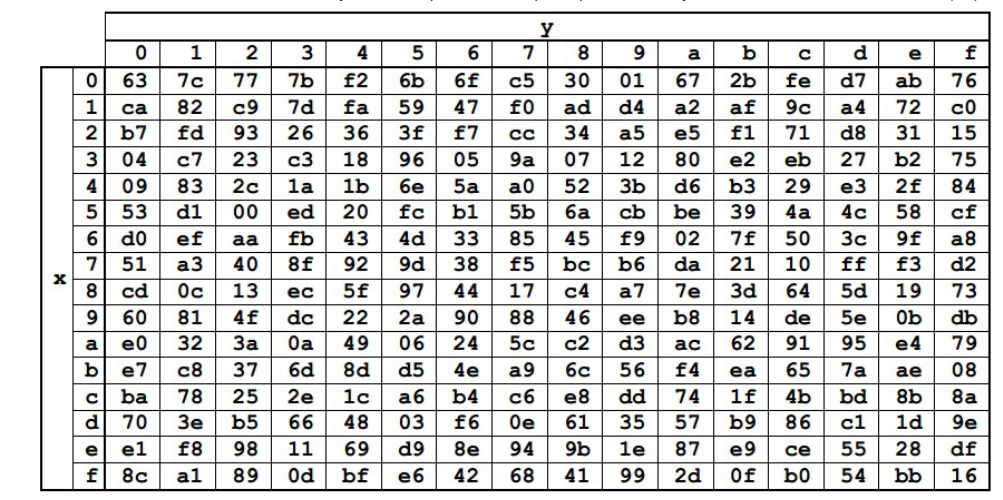
Число в скобках – мощность или количество элементов поля.

Элементами поля являются многочлены степени не более 7, которые могут быть заданы строкой своих коэффициентов. Байт очень легко представить в виде многочлена.



## SubBytes

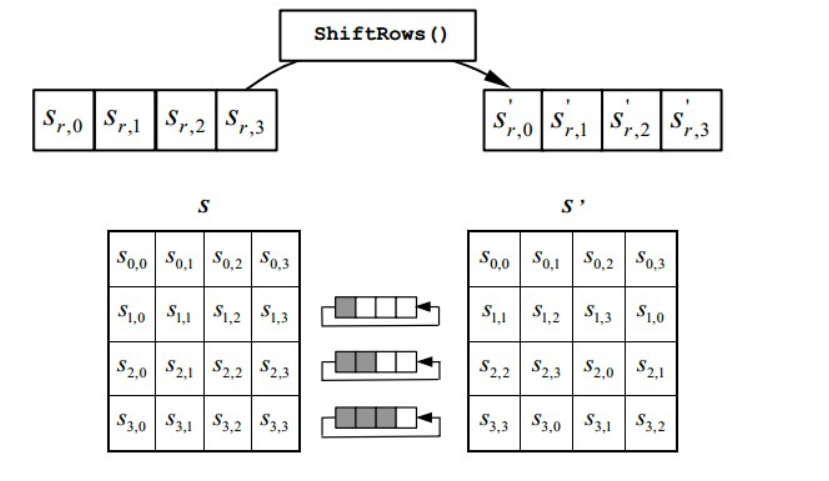
Преобразование представляет собой замену каждого байта из State на соответствующий ему из константной таблицы Sbox. Значения элементов Sbox представлены в шестнадцатеричной системе счисления. Сама же таблица получена посредством преобразований уже известного нам поля .



Каждый байт из State можно представить как {xy} в шестнадцатеричной системе счисления. Тогда следует заменять его на элемент, стоящий на пересечении строки x и столбца y. Например, {6е} заменится на {9f}, а {15} на {59}.

## ShiftRows

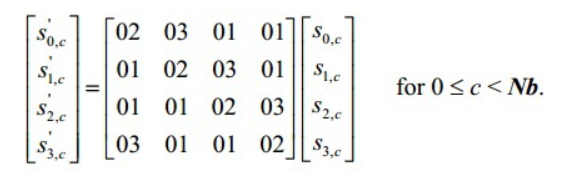
Простая трансформация. Она выполняет циклический сдвиг влево на 1 элемент для первой строки, на 2 для второй и на 3 для третьей. Нулевая строка не сдвигается.



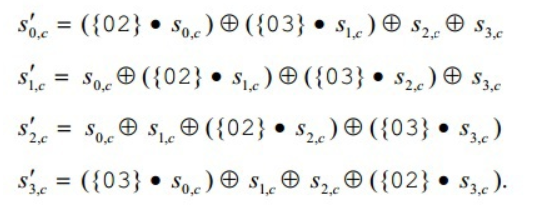
## MixColumns

В рамках этой трансформации каждая колонка в State представляется в виде многочлена и перемножается в поле по модулю с фиксированным многочленом .

В матричной форме:



При умножении матриц, значение аij получается как сумма произведений соответствующих элементов i-ой строки первой матрицы и j-ого столбца второй, т. е.



Новые правила:

* Сложение в поле эквивалентно операции XOR
* Умножение на {01} не меняет умножаемое
* Умножение на {02} производится по правилу: если умножаемое значение меньше {80}, оно сдвигается влево на 1 бит. Если же умножаемое значение больше или равно {80}, оно сначала сдвигается влево на 1 бит, а затем к результату сдвига применяется операция XOR со значением {1b}. Результат может перескочить за значение {ff}, то есть за границы одного байта. В этом случае нужно вернуть остаток от деления результата на {100}.
* Умножение на другие константы можно выразить через предыдущие

## AddRoundKey

Трансформация производит побитовый XOR каждого элемента из State с соответствующим элементом из RoundKey. RoundKey — массив такого же размера, как и State, который строится для каждого раунда на основе секретного ключа функцией KeyExpansion(), которую и рассмотрим далее.

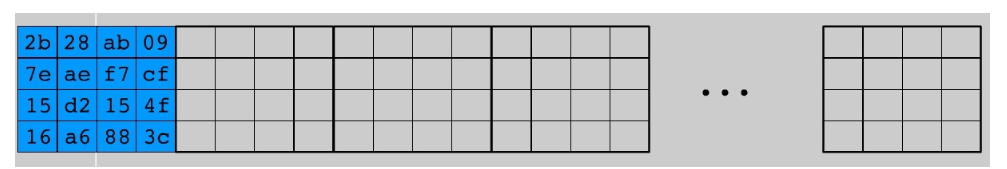
## KeyExpansion

Эта вспомогательная трансформация формирует набор раундовых ключей — KeySchedule.

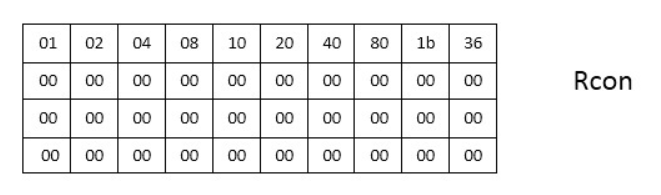
KeySchedule представляет собой длинную таблицу, состоящую из столбцов или блоков, каждый из которых равен по размеру State.

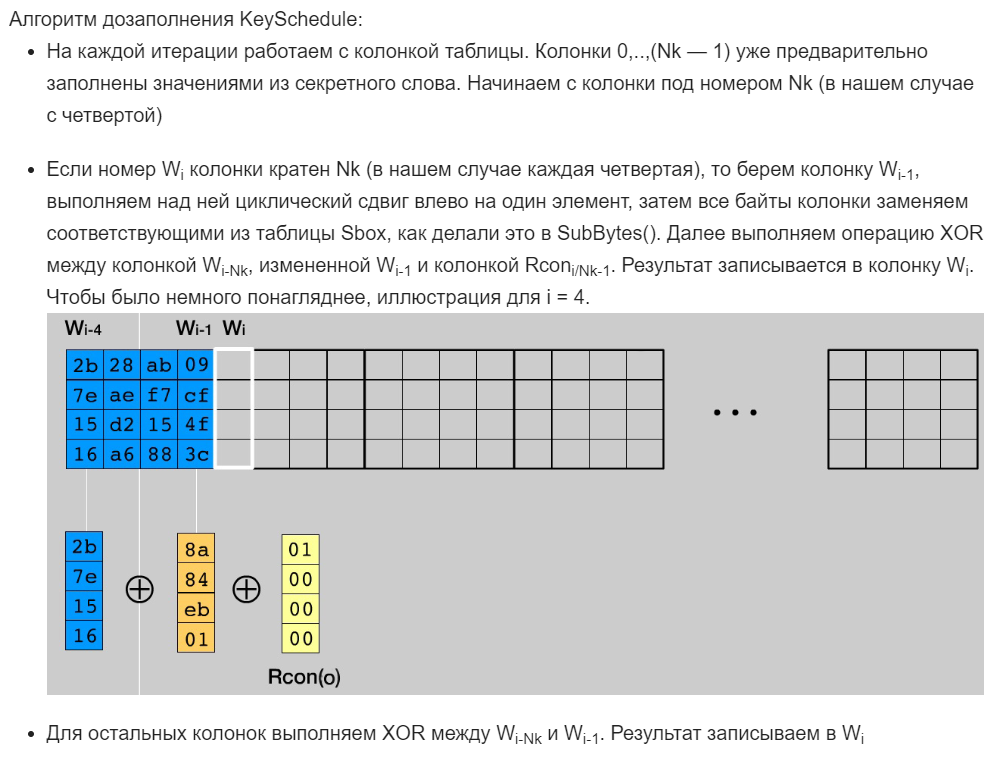
Первый раундовый ключ заполняется на основе секретного ключа, который вы придумаете, по формуле:

На рисунке изображен макет KeySchedule для AES-128: 11 блоков по 4 колонки. Для других вариаций алгоритма будет соответственно (Nr + 1) блоков по Nb колонок.



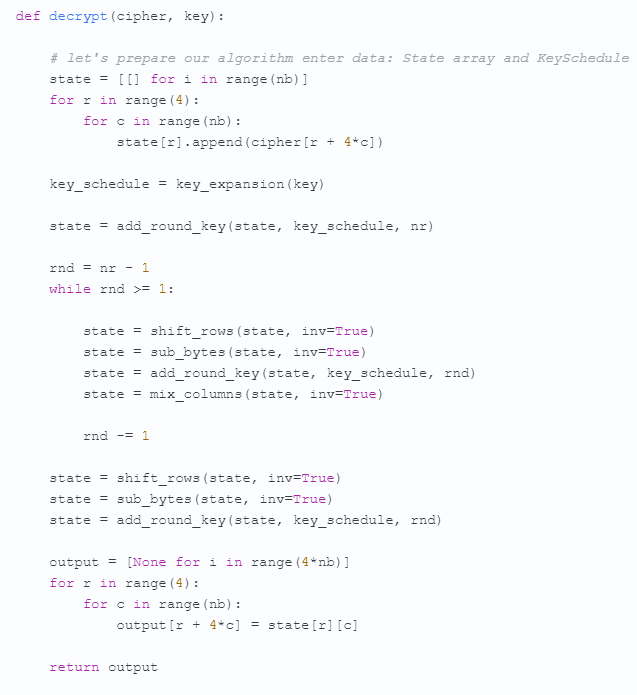
Теперь нам предстоит заполнить пустые места. Для преобразований опять определена константная таблица — Rcon — значения которой в шестнадцатеричной системе счисления.





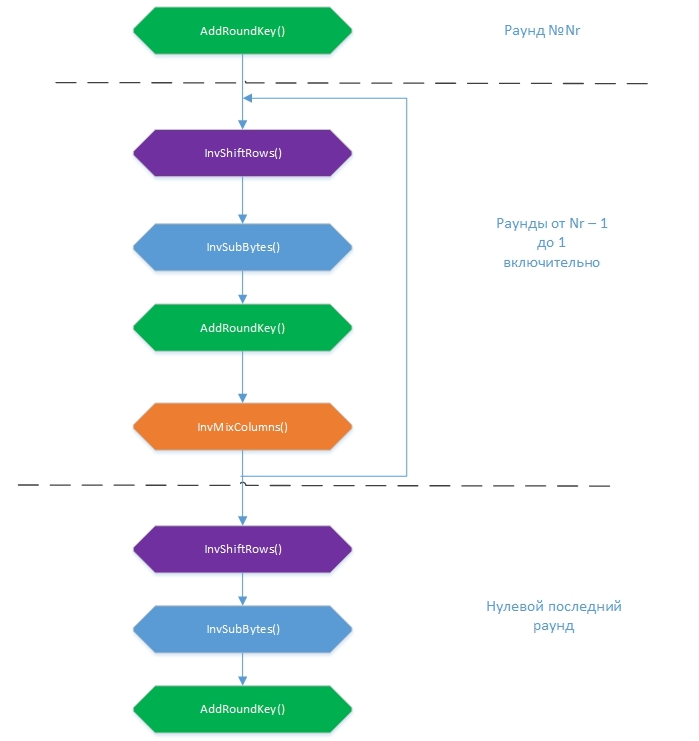
Выходной массив зашифрованных байтов составляется из State по формуле

# Расшифровка



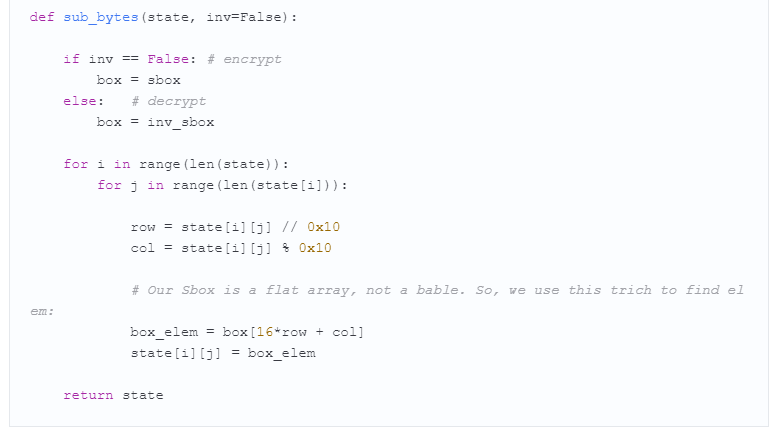
Идея здесь проста: если с тем же ключевым словом выполнить последовательность трансформаций, инверсных трансформациям шифрования, то получится исходное сообщение. Такими инверсными трансформациями являются *InvSubBytes(), InvShiftRows(), InvMixColumns()* и *AddRoundKey()*.

Стоит отметить, что последовательность добавления раундовых ключей в AddRoundKey() должна быть обратной: от Nr + 1 до 0. Изначально, как и при шифровании, из массива входных байтов формируется таблица State. Затем над ней в каждом раунде производятся преобразования, в конце которых должно получиться расшифрованный файл. Порядок трансформаций немного изменился. Что будет первым, InvSubBytes() или InvShiftRows(), на самом деле не важно, потому что одна из них работает со значениями байтов, а вторая переставляет байты, этих самых значений не меняя.

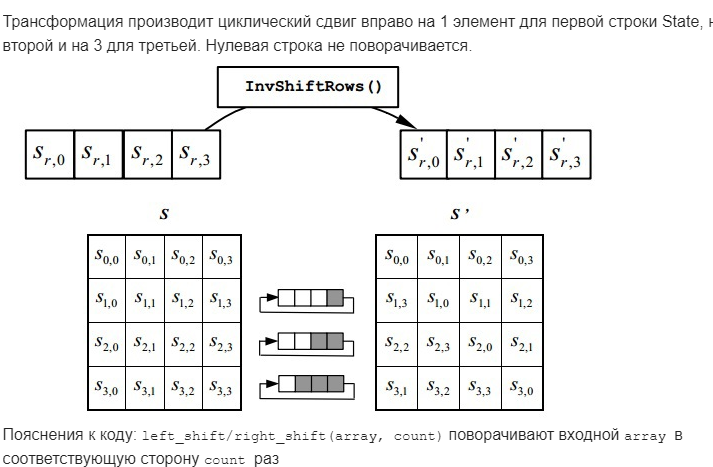


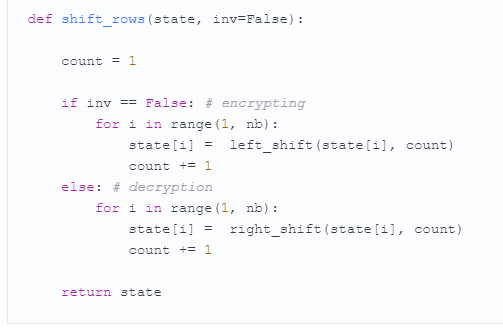
## InvSubBytes



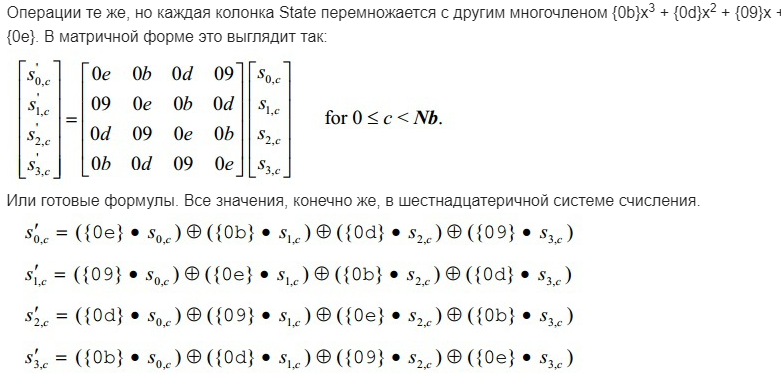


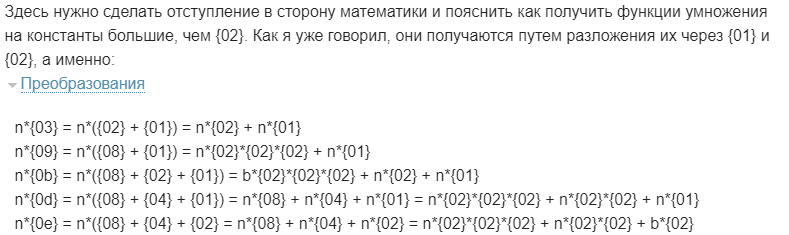
## InvShiftRows

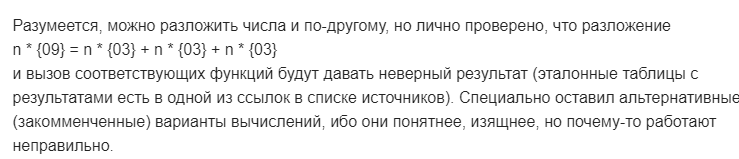




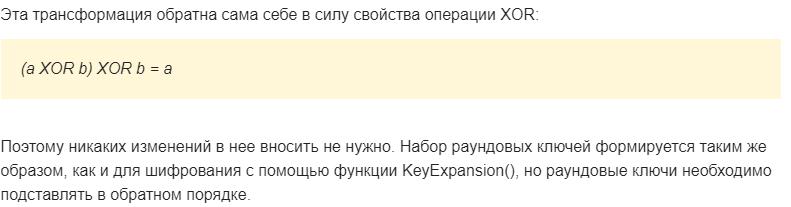
## InvMixColumns

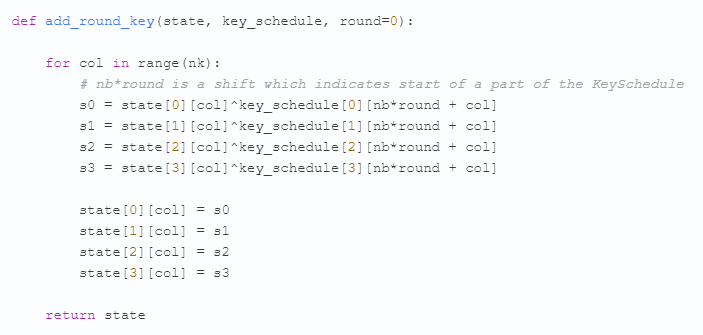




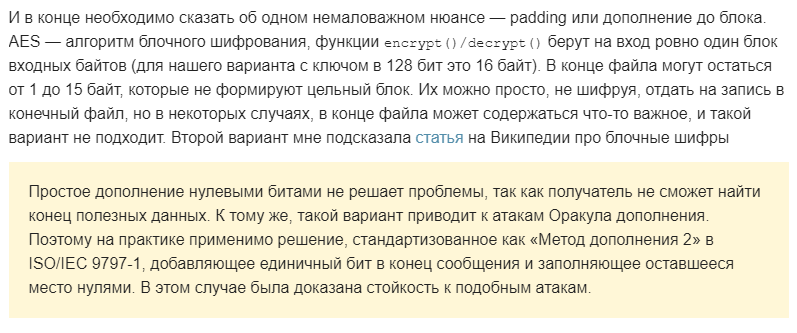


## AddRoundKey





# Заключение



1. Как бороться с незаполненным до конца блоком?

Ответ: до конца блока заполнить 0

1. Спрашивает про поле Галуа

Ответ: числа в поле Галуа представляются как наборы коэффициентов перед многочленами. В нем переопределены операции сложения (xor) и умножения (Умножение на 1 - тождественно. умножение на 2 - это умножение на x, остальное выводится из этих двух.). Используем для того, чтобы не выйти за пределы 1-байтных чисел.

1. Что будет если подать 1 байт?

Ответ: Дополняем 15 байт нулей

# Конечное поле или поле Галуа

***Конечное поле*** – поле, состоящее из конечного числа элементов. Число этих элементов называется ***порядком*** поля.

Обозначается или (Galois Field).

Где -число элементов поля.

С точностью до ***изоморфизма*** конечное поле полностью определяется ***порядком***, который является степенью какого-нибудь простого числа.

Где – простое число, -любое натуральное число.

При этом p является его ***характеристикой***.

Опр. ***Конечным полем*** называется конечное множество, на котором определены произвольные операции, называемые сложением, умножением, вычитанием и делением (кроме деления на 0) в соответствии с аксиомами поля.