# Оглавление

# Введение

Криптография (что в переводе с греческого означает “тайнопись”) издавна использовалась при обмене самой разнообразной информацией. Самые ранние упоминания об использовании криптографии: Египет - 1900 г. до н.э., Месопотамия - 1500 г. до н.э., при написании Библии - 500 г. до н.э.

Одним из наиболее известных в древней истории деятелей, постоянно пользовавшийся тайнописью, был Юлий Цезарь. Он придумал шифр, носящий название шифр Цезаря (Caesar cipher).

Проблема скрытой передачи и хранения информации, представленной в цифровом виде, имеет особое место в современном мире. В первую очередь, это связано с быстрым развитием вычислительной техники, сети Интернет и новых каналов передачи информации. Решение этой проблемы нашли в криптографии, которая позволяет зашифровать информацию, расшифровать которую сможет только тот, кому это предназначалось. Обычно для этого используется ключ, который задействован в процессе шифрования, и так же помогает расшифровать зашифрованную им информацию.

Цель данной работы: узнать, что такое и чем занимается наука криптография; познакомиться с основными понятиями и терминами в области шифрования; исследовать алгоритм шифрования AES и режим CTR.

# 1. Изучение понятийного аппарата криптографии

## 1.1 Основные понятия

Криптография— это наука об использовании математики для зашифрования и расшифрования данных. Криптография позволяет хранить важную информацию или передавать её по ненадёжным каналам связи (таким как Интернет) так, что она не может быть прочитана никем, кроме легитимного получателя.

Криптосистема — это завершённая комплексная модель, способная производить двусторонние криптопреобразования над данными произвольного объёма и подтверждать время отправки сообщения, обладающая механизмом преобразования паролей, ключей и системой транспортного кодирования.

Зашифрование - процесс преобразования открытых данных в зашифрованные данные (шифротекст) при помощи шифра. Зашифрование представляет собой зависящее от ключа взаимно однозначное криптографическое преобразование, которое ставит в соответствие блоку открытой информации, представленной в цифровой кодировке, блок шифрованной информации, также представленной в цифровой кодировке.

Расшифрование- это процесс преобразования зашифрованных данных в открытые данные при помощи ключа

Ключ - секретная информация, используемая криптографическим алгоритмом при зашифровании/расшифровании сообщений, постановке и проверке цифровой подписи, вычислении кодов аутентичности (MAC). При использовании одного и того же алгоритма результат шифрования зависит от ключа. Для современных алгоритмов сильной криптографии утрата ключа приводит к практической невозможности расшифровать информацию.

Симметричная криптосистема – это способ кодирования, когда для кодирования и раскодирования применяются один и тот же ключ и один и тот же алгоритм кодирования, называется *симметричным.* В симметричном методе кодирования ключ *K* является секретным, закрытым

**Основные задачи криптографии:**

1. *Обеспечение конфиденциальности*. Решение проблемы защиты информации от ознакомления с ее содержанием со стороны лиц, не имеющих права доступа к ней. В зависимости от контекста вместо термина "конфиденциальная" информация могут выступать термины "секретная", "частная", "ограниченного доступа" информация. Перечень информации, подлежащей защите, определен в законодательных актах: государственная тайна, коммерческая тайна, профессиональная тайна (врачебная, тайна следствия и т.п.), персональные данные.

2. *Обеспечение целостности*. Целостности информации – неизменность ее в процессе передачи или хранения. Для гарантии целостности необходим простой и надежный критерий обнаружения любых манипуляций с данными. Манипуляции с данными включают вставку, удаление и замену.Для обеспечения целостности в информацию вносится избыточность. Как правило, это достигается добавлением к сообщению некоторой проверочной комбинации, вычисляемой с помощью специального алгоритма и играющей роль контрольной суммы для проверки целостности полученного сообщения.Главное отличие от методов теории кодирования состоит в том, что алгоритм выработки проверочной комбинации является "криптографическим", то есть зависящим от секретного ключа. Без знания секретного ключа вероятность успешного навязывания противником искаженной или ложной информации мала. Такая вероятность служит мерой имитостойкости шифра, то есть способности самого шифра противостоять активным атакам со стороны противника.

3. *Обеспечение аутентификации*. Аутентификация — установление подлинности сторон (идентификация) и самой информации в процессе информационного взаимодействия. Информация, передаваемая по каналу связи, должна быть аутентифицирована по источнику, времени создания, содержанию данных, времени пересылки и т. д.Задачи аутентификации решаются различными методами, в том числе и криптографией.

Если стороны доверяют друг другу и обладают общим секретным ключом, задача аутентификации решается автоматически. Каждое успешно декодированное получателем сообщение может быть создано только отправителем, так как только он знает их общий секретный ключ. Для не доверяющих друг другу сторон решение подобных задач с использованием общего секретного ключа становится невозможным. Поэтому при аутентификации источника данных нужен механизм цифровой подписи.

4. *Обеспечение невозможности отказа от авторства* (ренегатства). В некоторых ситуациях,например в силу изменившихся обстоятельств, отдельные лица могут отказаться от ранее принятых обязательств. В связи с этим необходим некоторый механизм, препятствующий подобным попыткам. Основным механизмом решения этой проблемы также является цифровая подпись.

5. Обеспечение неотслеживаемости информации. Невозможность получения нарушителем содержательной информации на основе наблюдения за действиями законных пользователей.

## 1.2 Режимы работы шифров

Режим электронной кодовой книги(Electronic Code Book);

Каждый блок открытого текста заменяется блоком шифротекста. В ГОСТ 28147—89 называется режимом простой замены.

$$C_i = E_k\left(P_i\right)$$

где $i$ — номера блоков, $C_i$ и $P_i$ — блоки зашифрованного и открытого соответственно, а $E_k$ — функция блочного шифрования. Расшифровка аналогична:

$$P_i = D_k\left(C_i\right)$$

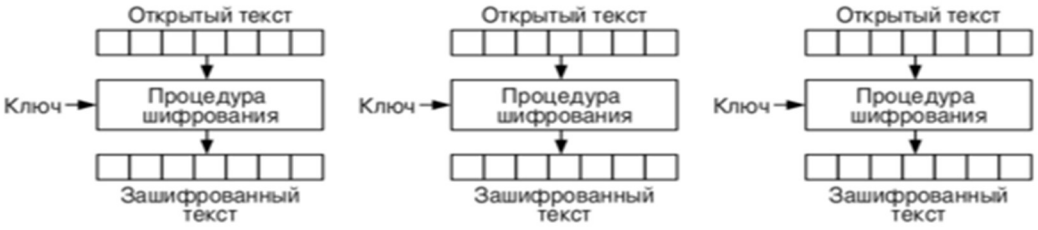



Рис.1 Шифрование в режиме ECB

Сцепление блоков шифра CBC (Cipher Block Chaning);

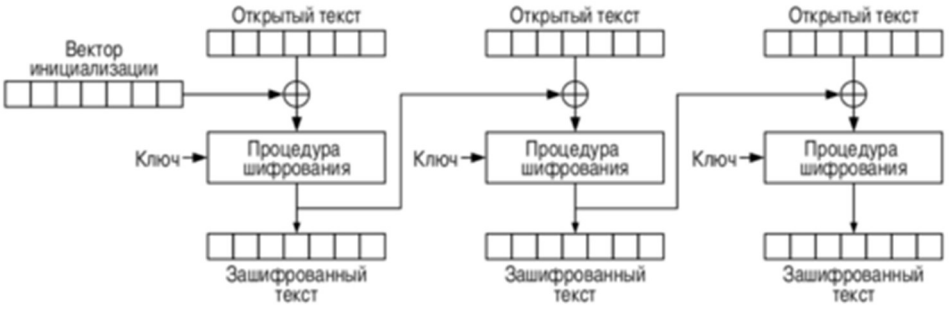
Каждый блок открытого текста (кроме первого) побитово складывается по модулю 2 (операция XOR) с предыдущим результатом шифрования. Шифрование:

$$C_0 = IV$$

$$C_i = E_k\left(P_i \oplus C_{i-1}\right),$$

где — номера блоков, $IV$ — вектор инициализации (синхропосылка), и — блоки зашифрованного и открытого текстов соответственно, а — функция блочного шифрования. Расшифровка:

$$P_i = C_{i-1} \oplus D_k\left(C_i\right)$$



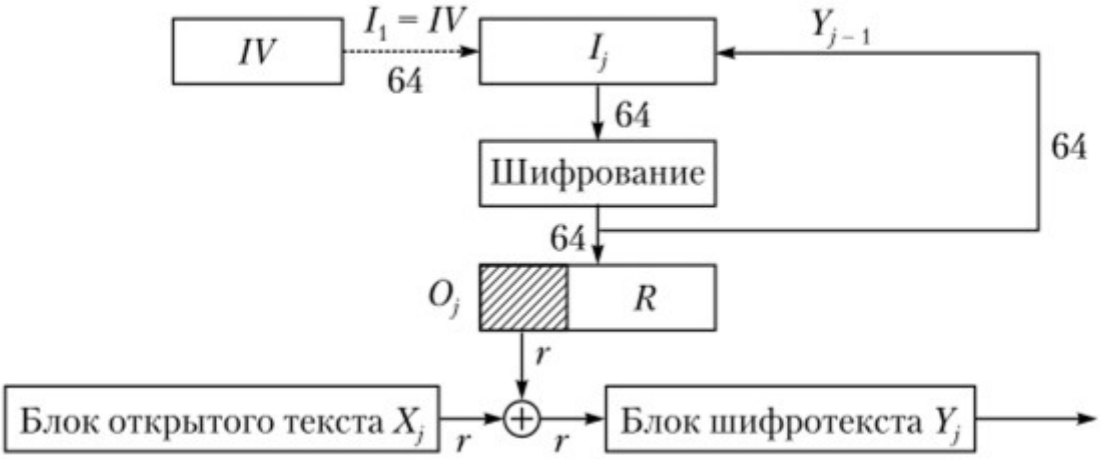

Обратная связь по шифротексту CFB (Cipher Feed Back);

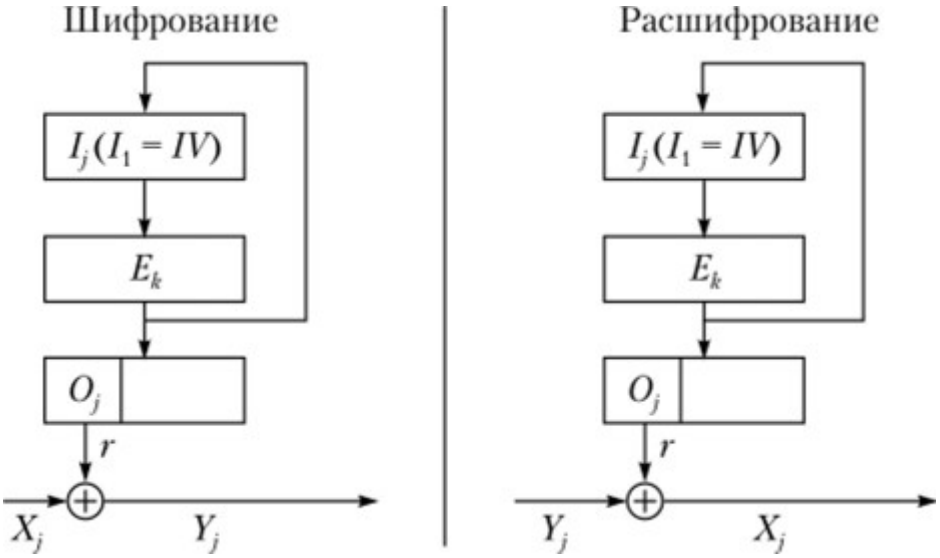
В режиме CFB алгоритм блочного шифрования можно использовать в качестве потокового шифра, установив длину г равной размеру символов потока (например, восемь бит для передачи текстов в ASCII-кодах или один бит для передачи двоичных последовательностей).

Использование поточного шифра избавляет от необходимости дополнять сообщение до целого числа блоков. Кроме того, с поточным шифром можно работать в режиме реального времени. Например, при передаче потока символов с помощью посимвольного поточного шифра каждый символ можно шифровать и сразу же передавать адресату, не дожидаясь окончания шифрования остальной части сообщения. При этом, как и в режиме CBC, шифрованный текст, соответствующий любому элементу открытого текста, будет зависеть от всех предыдущих элементов открытого текста.

Изменение одного бита шифротекста при передаче влечет неправильное расшифрование текущего /-битного блока и всех последующих /-битных блоков до тех пор, пока ошибка находится в сдвиговом регистре. Затем текст будет расшифровываться правильно. Пропуск или вставка символа в шифротексте также ведут к неправильному расшифрованию нескольких r-битных блоков (пока ошибка находится в сдвиговом регистре). Таким образом, режим CPB является самосинхронизирующимся как по отношению к ошибкам замены, так и по отношению к ошибкам иропуска/вставки.

Режим *обратной связи по выходу OEB* похож на предыдущий (CFB) с той разницей, что на каждом шаге значение сдвигового регистра заменяется результатом шифрования его предыдущего значения (/; = $E_к(1,$ ,)).


IV
$I_1 = IV$
64
$I_j$
$Y_{j-1}$
64
64
Шифрование
64
64
$O_j$
R
r
Блок открытого текста $X_j$
r
r
Блок шифротекста $Y_j$


Шифрование
$I_j(I_1 = IV)$
$E_k$
$O_j$
r
$X_j$
$Y_j$
Расшифрование
$I_j(I_1 = IV)$
$E_k$
$O_j$
r
$Y_j$
$X_j$

## 1.3 Структура блочного шифра

У большинства современных блочных шифров алгоритмы зашифрования и расшифрования устроены следующим образом.

При зашифровании сообщение *a* е подвергается предварительному преобразованию *U,* цель которого, как правило, заключается либо в наложении ключа на шифруемое сообщение, либо в перестановке бит исходного сообщения.

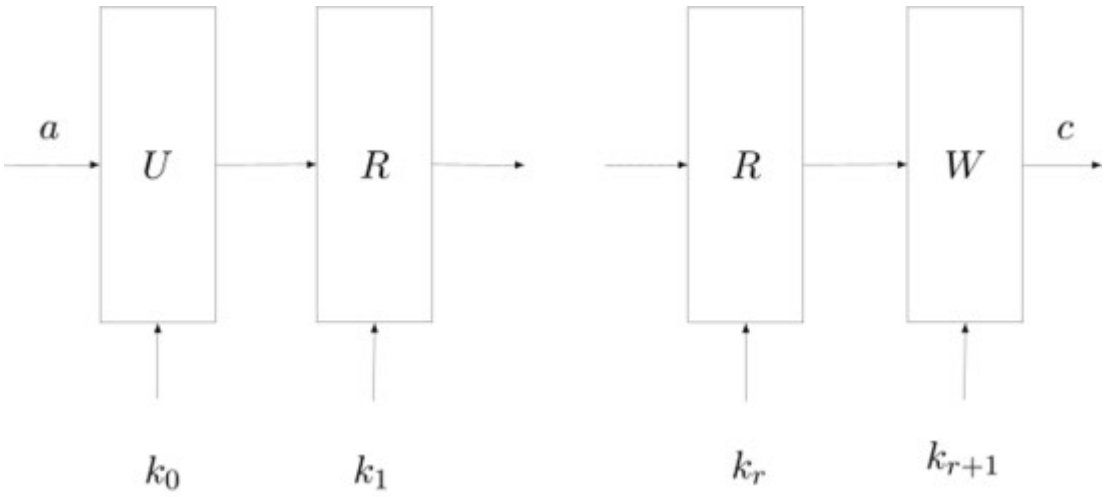
Затем к результату преобразования *U* многократно применяются однотипные преобразования *Ri,...,Rr,* называемые раундами. Параметр г определяет количество раундов алгоритма шифрования, при этом каждый из раундов зависит от своего уникального раундового ключа *к{, г* = 1,...,г. Как мы говорили выше, раундовые ключи вырабатываются из ключа *к* при помощи алгоритма развертки ключа и представляют собой двоичные вектора некоторой длины *s,* которая в общем случае может не совпадать с длиной сообщения *n.*

В завершение алгоритма шифрования применяется завершающее преобразование *W*. Преобразования *U* и *W* в зависимости от алгоритма шифрования также могут зависеть от своих уникальных ключей *ко* и, соответственно, $Ат_{г+}1$.

В общем виде алгоритм зашифрования может быть записан следующим образом

$$U, R_1, \ldots, R_r, W: \quad \mathbb{F}_2^s \times \mathbb{F}_2^n \to \mathbb{F}_2^n,$$
$$E(k, a) = W(k_{r+1}, R_r(k_r, \ldots R_1(k_1, U(k_0, a)) \ldots)).$$

Схематично мы можем представить структуру алгоритма зашифрования следующим образом.




Отметим, что во многих алгоритмах зашифрования преобразования $R,\ldots,R_r$ совпадают практически для всех индексов. Так, в алгоритме AES, см. раздел 7.5.1, отличие имеет только последний раунд, а в алгоритме ГОСТ 28147-89, см. раздел 7.4.2, все раунды одинаковы.

Мы можем записать алгоритм расшифрования в виде, аналогичном (7.3). Обозначим

$$U^{-1}, R_1^{-1}, \ldots, R_r^{-1}, W^{-1}: \quad \mathbb{F}_2^s \times \mathbb{F}_2^n \to \mathbb{F}_2^n$$

обратные преобразования, удовлетворяющие равенствам

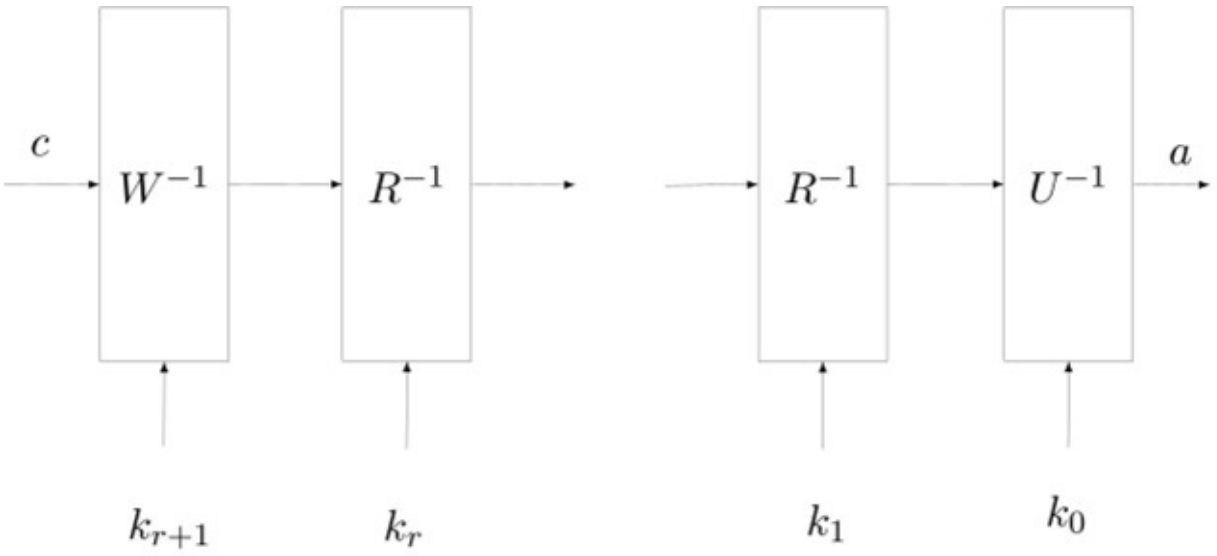
$$U^{-1}(k_0, U(k_0, a)) = a, \quad W^{-1}(k_{r+1}, W(k_{r+1}, a)) = a,$$
$$R_i^{-1}(R_i(k_i, a)) = a, \quad i = 1, \ldots, r,$$

для любого $a$ € FJ> и любого набора ключей $ко,\ldots, к_{г+}$.

Пусть $c = E(к,а)$ — шифртекст, соответствующий сообщению $а,$ тогда алгоритм расшифрования может быть записан в виде

$$D(k, c) = U^{-1}(k_0, R_1^{-1}(k_1, \ldots, R_r^{-1}(k_r, W^{-1}(k_{r+1}, c)) \ldots)).$$

Как и ранее, мы может представить структуру алгоритма расшифрования следующим образом.




По способу строения раундовых преобразований все блочные шифры могут быть сгруппированы в два больших класса:

• «сеть Фейстеля», впервые примененную Хорстом Фейстелем при разработке алгоритма DES,

• «подстановочно-перестановочную сеть», или коротко — SP- сеть[1], основанную на последовательном применении некоторого числа подстановок и перестановок.

Сеть Фе́йстеля, или конструкция Фейстеля (англ. *Feistel network, Feistel cipher*), — один из методов построения блочных шифров. Сеть состоит из ячеек, называемых ячейками Фейстеля. На вход каждой ячейки поступают данные и ключ. На выходе каждой ячейки получают изменённые данные и изменённый ключ. Все ячейки однотипны, и говорят, что сеть представляет собой определённую многократно повторяющуюся (итерированную) структуру

SP сеть

Выше мы рассматривали алгоритмы блочного шифрования, построенные по принципам сети Фейстеля. Другим способом построения блочных шифров является «подстановочно-перестановочная сеть», основанная на многократном применении однотипных раун- довых преобразований, состоящих из подстановок, перестановок и операций наложения ключа. Одним из первых шифров, построенных по такому принципу, был алгоритм «Люцифер», легший в основу алгоритма DES. Среди других алгоритмов, относимых к классу SP-сетей, можно выделить алгоритмы IDEA, AES и Serpent, а также российский алгоритм «Кузнечик».

## 2. Описание выбранного алгоритма и режима шифрования

AES (англ. *Advanced Encryption Standard*; также *Rijndael*, [rɛinda:l] — *рейндал*) — симметричный алгоритм блочного шифрования (размер блока 128 бит, ключ 128/192/256 бит), принятый в качестве стандарта шифрования правительством США по результатам конкурса AES. Национальный институт стандартов и технологий США (англ. *National Institute of Standards and Technology*, NIST) опубликовал спецификацию AES 26 ноября 2001 года после пятилетнего периода, в ходе которого были созданы и оценены 15 кандидатур. 26 мая 2002 года AES был объявлен стандартом шифрования. AES – симметричный итеративный блоковый алгоритм.

Общие характеристики AES

- AES зашифровывает и расшифровывает 128-битовые блоки данных.

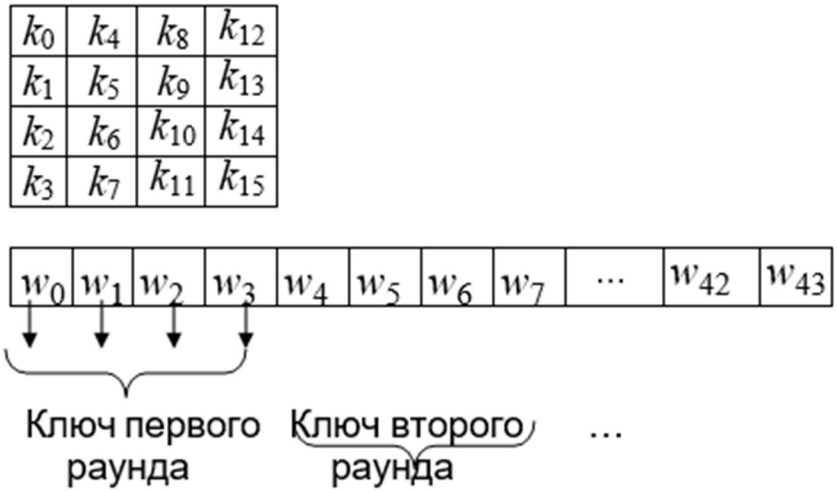
- AES позволяет использовать три различных ключа длиной 128, 192 или 256 бит (в зависимости от длины ключа версии шифра обозначают AES-128, AES-192 или AES-256).

- От размера ключа зависит число раундов шифрования: длина 128 бит – 10 раундов; длина 192 бита – 12 раундов; длина 256 бит – 14 раундов.

- Все раунды, кроме последнего, идентичны.

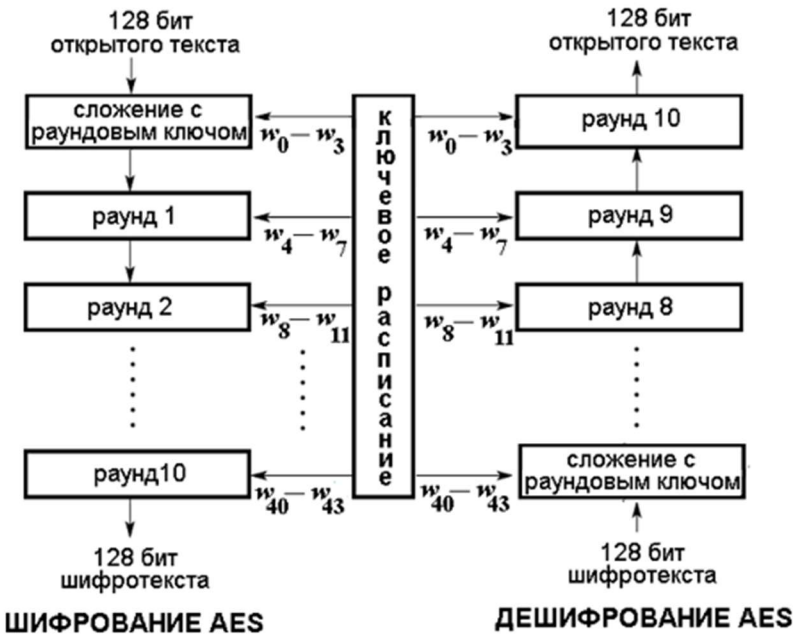
## 2.1 Общая структура AES

Как отмечалось, в рассматриваемой версии алгоритма AES-128ключ шифра состоит из 128 битов, поделенных на16 байтов $k_0, k_1,..., k_{15}$, и записывается в столбцы матрицы Input*Key*. Каждый столбец матрицы Input*Key* образует слово, т.е. фактически ключ шифра – это четыре слова $w_0, w_1, w_2, w_3$, где $w_0 = k_0k_1k_2k_3$, $w_1 = k_4k_5k_6k_7$ и т.д.




Из этих слов с помощью специального алгоритма (о нем позже) образуется последовательность из 44 слов: w0 , w1, w2 ,..., w43 (каждое слово по 32 бита). На каждый раунд шифрования подаются по четыре слова этой последовательности. Они и будут играть роль раундового ключа.

Схема преобразования данных показана на рисунке.



Перед первым раундом выполняется операция AddRoundKey (суммирование по модулю 2 с начальным ключом шифра). Преобразования, выполненные в одном раунде, обозначают

Round (State, RoundKey),

где переменная State – матрица, описывающая данные на входе раунда и на его выходе после шифрования; переменная RoundKey – матрица, содержащая раундовый ключ.

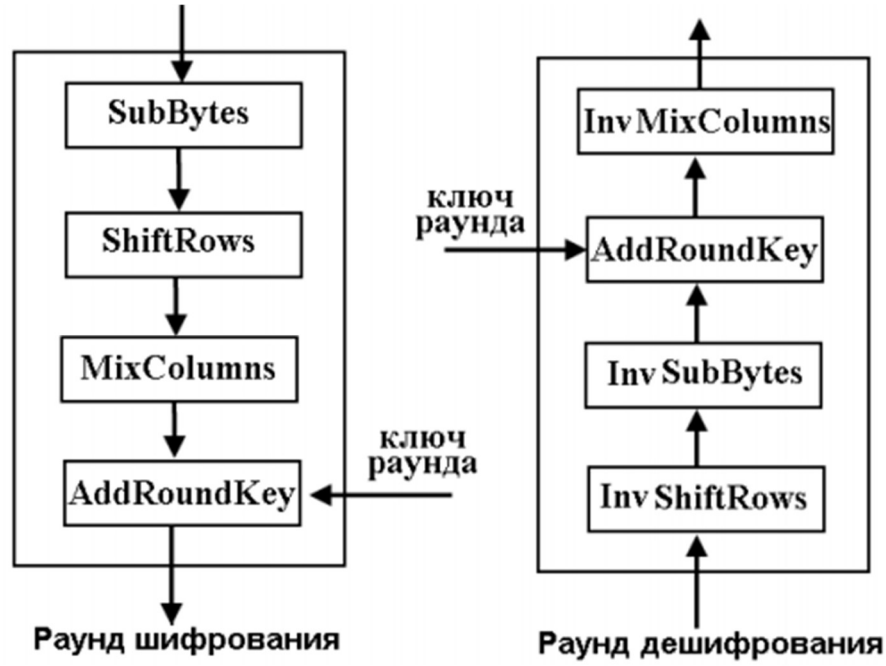
Раунд состоит из 4 различных преобразований:

♣ SubBytes – побайтовая подстановка в S-боксе с фиксированной таблицей замен;

♣ ShiftRows – побайтовый сдвиг строк матрицы State на различное количество байт;

♣ MixColumns – перемешивание байт в столбцах;

♣ AddRoundKey – сложение с раундовым ключом (операция XOR). Последний раунд несколько отличается от предыдущих тем, что не задействует функцию MixColumns.




При дешифровании в каждом раунде выполняются обратные операции: InvShiftRows, InvSubBytes, AddRoundKey и InvMixColumns (в обозначениях перед названием функции появляется приставка Inv). Порядок выполнения операций при шифровании и дешифровании различен, причины чего будут ясны после детального рассмотрения каждого преобразования.

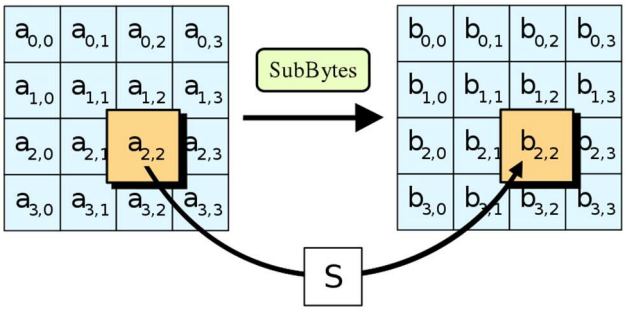
## 2.2 Раундовое преобразование AES

Рассмотрим подробнее преобразования раунда шифрования.

**Операция SubBytes:**

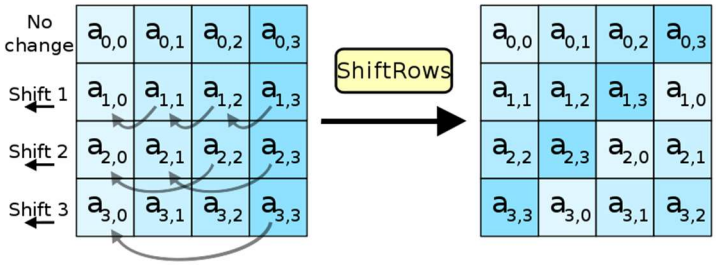
Процедура SubBytes() обрабатывает каждый байт состояния, независимо производя нелинейную замену байтов, используя таблицу замен (S-box). Такая операция обеспечивает нелинейность алгоритма шифрования. Построение S-box состоит из двух шагов. Во-первых, производится взятие обратного числа в GF($2^8$). Во-вторых, к каждому байту b, из которых состоит S-box, применяется следующая операция:

$$b'_i = b_i \oplus b_{(i+4) \bmod 8} \oplus b_{(i+5) \bmod 8} \oplus b_{(i+6) \bmod 8} \oplus b_{(i+7) \bmod 8} \oplus c_i$$




**Операция ShiftRows:**

ShiftRows работает со строками State. При этой трансформации строки состояния циклически сдвигаются на r байт по горизонтали в зависимости от номера строки. Для нулевой строки r = 0, для первой строки r = 1 и т. д. Таким образом, каждая колонка выходного состояния после применения процедуры ShiftRows состоит из байтов из каждой колонки начального состояния.

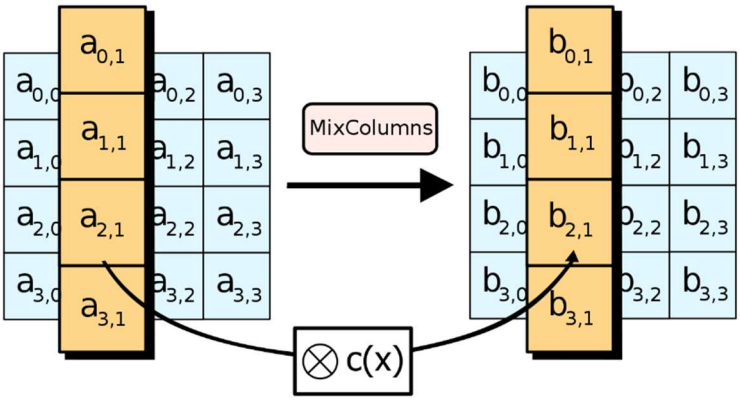



**Операция MixColomns:**

С помощью этой операции выполняется перемешивание байтов в столбцах матрицы State. Каждый столбец этой матрицы принимается за многочлен над полем GF($2^8$ ) и умножается на фиксированный многочлен

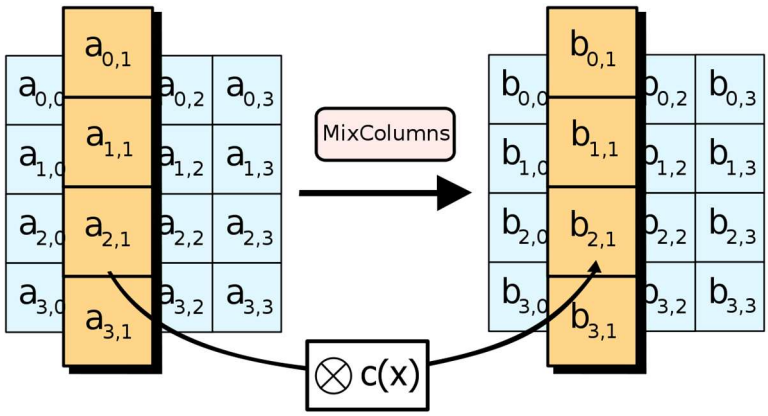
$$c(x) = c_3x^3 + c_2x^2 + cx + c_0 = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$$

по модулю многочлена $x^4 + 1$ (напомним, все коэффициенты многочленов над полем $GF(2^8)$ – байты).




**Операция AddRoundKey:**

Функция AddRoundKey(State, RoundKey) побитово складывает элементы переменной RoundKey и элементы переменной State по принципу: i-й столбец данных (i = 0,1,2,3) складывается с определенным 4-байтовым фрагментом расширенного ключа W [4r + 1], где r – номер поточного раунда алгоритма. При шифровании первое сложение ключа раунда происходит до первого выполнения операции SubBytes.



## 2.3 Ключевое расписание AES

Раундовые ключи вырабатываются из ключа шифра K с помощью процедуры расширения ключа, в результате чего формируется массив раундовых ключей, из которого затем непосредственно выбирается необходимый раундовый ключ.

Каждый раундовый ключ имеет длину 128 бит (или 4 четырехбайтовых слова $w_i, w_{i+1}, w_{i+2}, w_{i+3}$ , а длина в битах всех раундовых ключей равна 128 бит·(10 раундов+1) = 1408 бит (или 44 четырехбайтовых слова $w_0$ , $w_1$, $w_2$ ,..., $w_{42}$ , $w_{43}$). Первые четыре слова $w_0$ , $w_1$, $w_2$ , $w_3$ в ключевом массиве заполнены ключом шифра, из остальных выработанных 40 слов выбираются по 4 слова для ключа раунда. Выбор слов прост: первые четыре слова (они совпадают с ключом шифра) являются ключом с номером 0, следующие четыре слова $w_4, w_5, w_6, w_7$ – раундовым ключом для первого полного раунда и т.д.

Новые слова $w_{i+4}$ , $w_{i+5}$ , $w_{i+6}$ , $w_{i+7}$ следующего раундового ключа определяются из слов $w_i$ , $w_{i+1}$, $w_{i+2}$ , $w_{i+3}$ предыдущего ключа на основе уравнений:

$$w_{i+5} = w_{i+4} \oplus w_{i+1};$$

$$w_{i+6} = w_{i+5} \oplus w_{i+2};$$

$$w_{i+7} = w_{i+6} \oplus w_{i+3}.$$

Первое слово $w_{i+4}$ в каждом раундовом ключе изменяется по-другому:

$$w_{i+4} = w_i \oplus g(w_{i+3}),$$

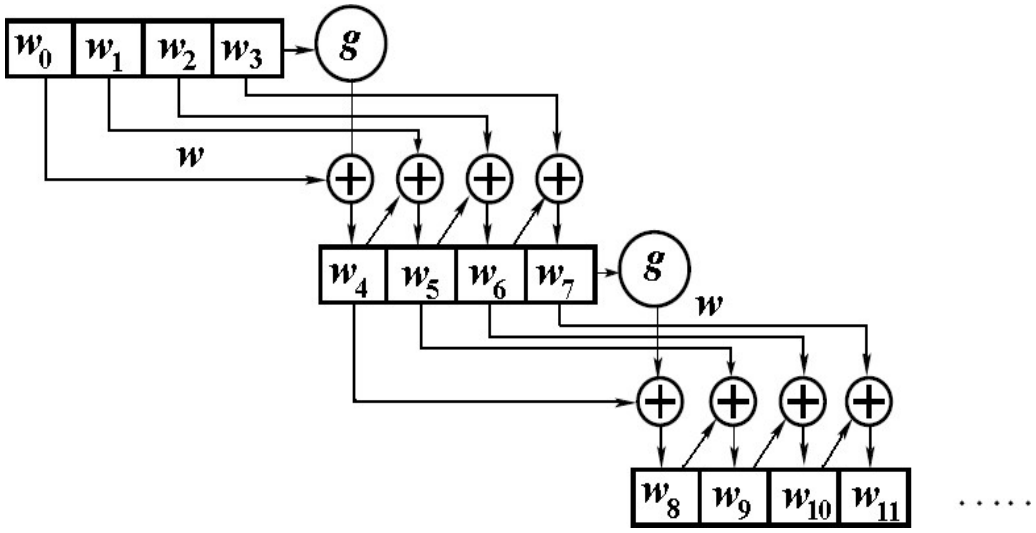
Здесь действие функции $g$ сводится к последовательному выполнению трех шагов, отображающих слово в слово:

$1^0$ циклический сдвиг четырехбайтового слова влево на один байт (операция RotWord);

$2^0$ замена каждого байта слова, полученного на шаге $1^0$, всоответствии с таблицей SubBytes , используемой при шифровании (операция SubWord);

$3^0$ суммирование по mod 2 байтов, полученных на шаге 20, с раундовой постоянной Rcon[i] = (RC[i],0, 0, 0), несекретной и уникальной для каждого раундового ключа Ki . Три самые правые байты этой константы

– нулевые, а ненулевой левый Байт меняется по известному закону рекурсии: $RC[1]=1,\ RC[i]=2\cdot RC[i-1],\ i=1,2,...10$.



## 2.3 Режим шифрования CTR

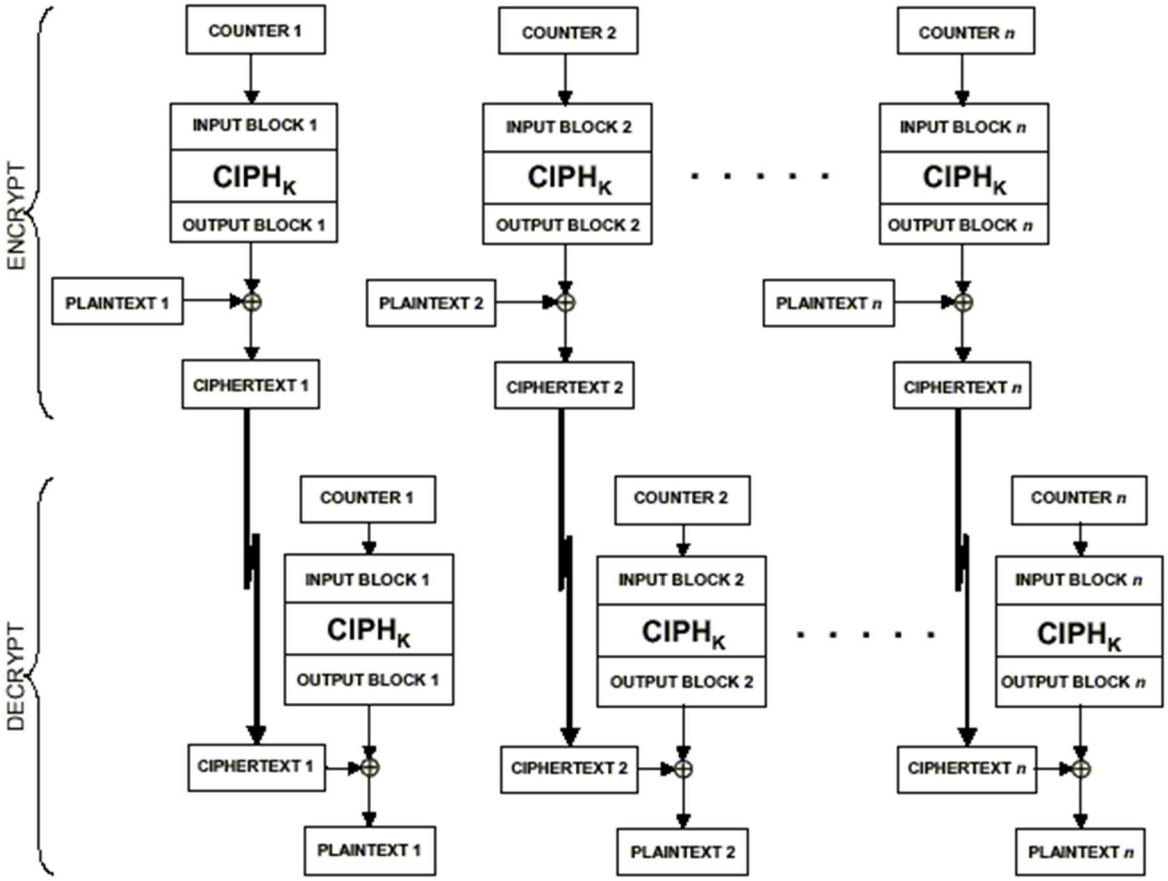
Режим счётчика (counter mode, CTR) предполагает возврат на вход соответствующего алгоритма блочного шифрования значения некоторого счётчика, накопленного с момента старта. Режим делает из блочного шифра потоковый, то есть генерирует последовательность, к которой применяется операция XOR с текстом сообщения. Исходный текст и блок зашифрованного текста имеют один и тот же размер блока, как и основной шифр (например, AES) Режим CTR предусматривает следующие операции.

Шифрование в режиме CTR

$$C_i = P_i \oplus E_k(Ctr_i); i = 1, 2, \ldots, m$$

Расшифровка в режиме CTR

$$P_i = C_i \oplus E_k(Ctr_i); i = 1, 2, \ldots, m$$



## 2.4 Режим шифрования CBC

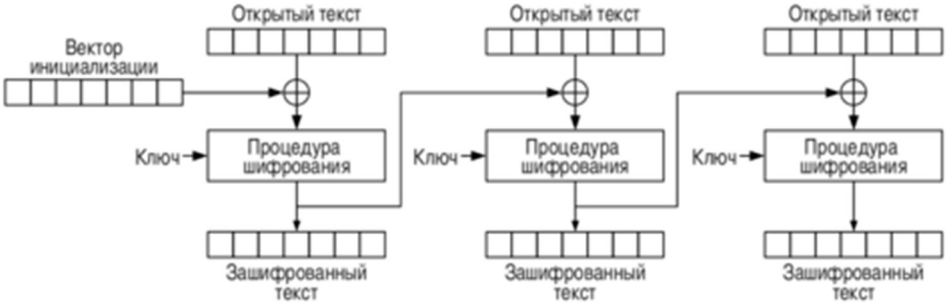
Каждый блок открытого текста (кроме первого) побитово складывается по модулю 2 (операция XOR) с предыдущим результатом шифрования. Шифрование:

$$C_0 = IV$$

$$C_i = E_k\left(P_i \oplus C_{i-1}\right),$$

где — номера блоков, $IV$ — вектор инициализации (синхропосылка), и — блоки зашифрованного и открытого текстов соответственно, а — функция блочного шифрования. Расшифровка:

$$P_i = C_{i-1} \oplus D_k\left(C_i\right)$$




## 2.5 Режим шифрования OFB

Режим (OFB) обратной связи вывода превращает блочный шифр в синхронный шифр потока: он генерирует ключевые блоки, которые являются результатом сложения с блоками открытого текста, чтобы получить зашифрованный текст. Так же, как с другими шифрами потока, зеркальное отражение в зашифрованном тексте производит зеркально отражённый бит в открытом тексте в том же самом местоположении. Это свойство позволяет многим кодам с исправлением ошибок функционировать как обычно, даже когда исправление ошибок применено перед кодированием.

Из-за симметрии операции сложения, шифрование и расшифрование похожи:

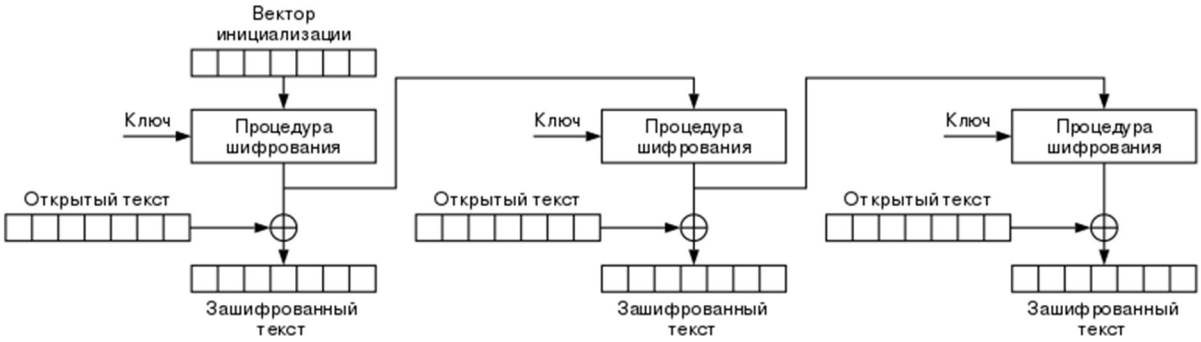
$$C_i = P_i \oplus O_i$$

$$P_i = C_i \oplus O_i$$

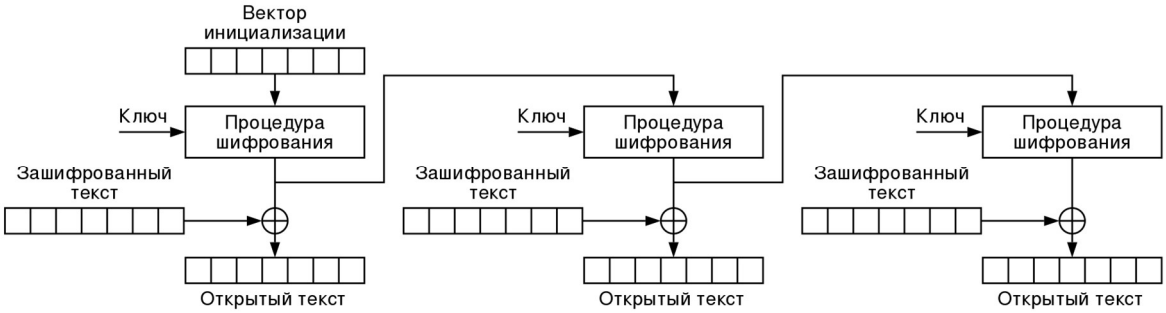
$$O_i = E_k(O_{i-1})$$

$$O_0 = IV$$

Зашифрование в режиме OFB:




Расшифрование в режиме OFB:




Каждая операция блочного шифра обратной связи вывода зависит от всех предыдущих и поэтому не может быть выполнена параллельно. Однако, из-за того, что открытый текст или зашифрованный текст используются только для конечного сложения, операции блочного шифра могут быть выполнены заранее, позволяя выполнить заключительное шифрование параллельно с открытым текстом.

Обратная связь по выходу на k разрядов не рекомендуется из соображений криптостойкости. Режим OFB имеет следующее преимущество по сравнению с режимом CFB: ошибки, возникающие в результате передачи по каналу с шумом, при дешифровании не «размазываются» по всему шифротексту, а локализуются в пределах одного блока. Однако открытый текст может быть изменён путём определённых манипуляций с блоками шифротекста. Несмотря на то, что OFB-шифрование не поддаётся распараллеливанию, эффективность процедуры может быть повышена за счёт предварительной генерации независимой последовательности блоков

Данный метод называется также «режим обратной связи по выходу».

# 3. Практическая часть

Задача состояла в реализации алгоритма шифрования и режимов шифрования к нему.

В качестве алгоритма выл выбран AES128, а в качестве режима шифрования был выбран CTR, CBC и OFB.

## 3.1 Реализация

В качестве языка реализации был выбран C. Процесс шифрования блока в AES определяется следующей функцией:

*aes_cipher (inputblock, outputblock, expanded key),*

где *aes_cipher* – функция шифрования, *inputblock* – входящий блок информации (16 байтный массив), *outputblock* – зашифрованный блок информации (16 байтный массив), *expanded key* – расширенный ключ (44 байтный массив), полученный из ключа (16 байтный массив), при помощи функции *aes_key_expansion*.

Расширение ключа происходило при помощи функции *aes_key_expansion*:

*aes_key_expansion* (*key, expanded key*),

где *key* – входящий ключ (16 байтный массив), *expanded key* – расширенный ключ (44 байтный массив).

Входящая информация, в виде стоки (*string*), преобразовывается в блоки по 16 байт при помощи функции *make_blocks_form_string,* которая возвращает указатель на массив указателей на входящие блоки информации(*input_blocks*), а также передает информацию о количестве блоков(*numberofblocks*):

*make_blocks_form_string (string, & numberofblocks).*

Далее полученные блоки (*inputblocks*) подаются в функцию шифрования (расшифрования) в режиме CTR *encryption_decryptionCTR* возвращающую указатель на массив указателей на массивы с зашифрованными данными:

*encryption_decryptionCTR* (*inputblocks, numberofblocks, seed, key*),

где *seed* – это аргумент генерации счетчика для режима CTR.

Генерация счетчиков для CTR происходит при помощи функции *make_counters,* возвращающую указатель на массив указателей на счетчики для каждого последующего раунда шифрования (расшифрования):

*make_counters (amount, seed),*

где *amount* – количество нужных блоков для шифрования (расшифрования).

Рассмотрим шифрование на режиме CBC. Оно реализовано с помощью функции *encryption_CBC*:

*encryption_CBC (inputblocks, numberofblocks, IV, key),*

где *inputblocks* – массив из 16 байтов входящей информации, *numberofblocks* – количество этих блоков*, IV* – вектор инициализации (массив из 16 байтов), *key* – ключ шифрования (массив из 16 байтов).

На выход функция возвращает указатель на массив указателей на массивы с зашифрованными данными.

Расшифрование происходит с помощью функции *decryption_CBC*:

*decryption_CBC (inputblocks, numberofblocks, IV, key).*

Результатом ее работы является указатель на массив указателей на массивы с расшифрованными данными.

Далее рассмотрим шифрование на режиме OFB, которое происходит с помощью функции шифрования (расшифрования) *encryption_decryptionOFB*:

*encryption_decryptionOFB (inputblocks, numberofblocks, IV, key).*

Результатом ее работы является указатель на массив указателей на массивы с расшифрованными (зашифрованными) данными.

Блоки, полученные при расшифровке, заново преобразуются в строку при помощи функции *make_string_from_block*:

*make_string_from_block (blockaddr, ammountofblocks)*,

где *blockaddr* - указатель на массив указателей на массивы с данными, *ammountofblocks* количество массивов с данными.

С полным кодом можно ознакомиться в приложенных файлах.

# Заключение

Значение криптографии в современном обществе очень велико. Новая информационная инфраструктура создает новые опасности для информации. Криптография в нашем мире стала очень востребованной наукой, поэтому ее развитие не стоит на месте, и наряду с новыми угрозами находятся новые способы защиты. Появляется заметный спрос на специалистов, знакомых с принципами криптографической защиты информации.

В рамках данной работы мы подробно рассмотрели понятийный аппарат криптографии и основные ее задачи, рассмотрели структуру блочных шифров, а также реализовали алгоритм шифрования AES и режимы CTR, CBC и OFB. Данный проект помог нам расширить знания в области криптографии, что может понадобиться при передаче ценной информации во избежание ее утери.

# Список литературы


1) «AES (стандарт шифрования)» – Википедия (электронный ресурс). Доступ по ссылке: https://ru.wikipedia.org/wiki/AES_(%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82_%D1%88%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) (дата обращения: 08.05.2021).
2) «Режимы шифрования» – Википедия (электронный ресурс). Доступ по ссылке: https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B6%D0%B8%D0%BC_%D1%88%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F (дата обращения: 15.05.2021).
3) А.Б. Лось, А.Ю. Нестеренко, М.И. Рожков «Криптографические методы защиты информации» (дата обращения: 20.04.2021)
4) ГОСТ Р 34.13-2015 «Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров» (дата обращения: 22.04.2021).

# Приложение 1

С кодом можно ознакомиться по ссылке:

https://github.com/moonmeander47/aes_CTR_CBC_OFB