**AES**

AES (англ. Advanced Encryption Standard; также Rijndael, [rɛindaːl] — 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. Этот алгоритм хорошо проанализирован и сейчас широко используется, как это было с его предшественником DES. 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. По состоянию на 2009 год AES является одним из самых распространённых алгоритмов симметричного шифрования. Поддержка ускорения AES была введена фирмой Intel в семейство процессоров x86 начиная с Arrandale в 2010 году, а затем на процессорах Sandy Bridge; фирмой AMD — в Bulldozer с 2011 года.

**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.

Основным элементом, которым оперирует алгоритм AES, является байт – последовательность 8 бит, обрабатываемых как единое целое. Для формирования байтов 128 битов блока открытого текста, выходного блока шифротекста и ключа шифра делятся на группы из 8- ми рядом стоящих бит так, чтобы в целом получился массив байт. Ниже представлена принятая нумерация бит в пределах каждого байта:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № бита на  входе | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | … |
| № байта | 0 | | | | | | | | 1 | | | | | | | | 2 | | | | | | | | … |
| № бита в  байте | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | … |

Задавать значение байта удобно в шестнадцатеричной системе исчисления. Для этого байт делится на две группы из 4-х бит: группа старших бит в байте представляется первым шестнадцатеричным символом, а группа младших бит – вторым. Например, для байта 10101100 получим

Обозначим

# 10101100 = 1010 1100 = *АС*.

*in*0, *in*1,…, *in*15 – 16 байт блока открытого текста;

*k*0, *k*1,…, *k*15 – 16 байт ключа шифра;

– 16 байт блока шифротекста.

Входными данными для операций шифрования есть массив из 16 байт *in*0, *in*1,…, *in*15. Перед началом шифрования байты этого массива размещаются последовательно в столбцы матрицы Inpu*tBlock* (сверху вниз). Внутри алгоритма операции выполняются над матрицей байт, называемой ***матрицей состояний*** *State* или просто ***состоянием***. Конечное значение матрицы состояния Outpu*tBlock* является выходом алгоритма и преобразуется в последовательность байтов шифротекста *оut*0, *out*1,…, *out*15. Аналогично в столбцы матрицы Input*Key* попадают и

16 байтов

*k*0 , *k*1,..., *k*15 ключа шифра. Размерность всех матриц *–* 4 x 4.

Схематически такое представление данных выглядит так:

Входные байты Выходные байты

Inpu*tBlock State* Outpu*tBlock*

# → =

|  |  |  |  |
| --- | --- | --- | --- |
| *in*0 | *in*4 | *in*0 | *in*0 |
| *in*1 | *in*5 | *in*9 | *in*13 |
| *in*2 | *in*6 | *in*10 | *in*14 |
| *in*3 | *in*7 | *in*11 | *in*15 |

|  |  |  |  |
| --- | --- | --- | --- |
| *s*0,0 | *s*0,1 | *s*0,2 | *s*0,3 |
| *s*1,0 | *s*1,1 | *s*1,2 | *s*1,3 |
| *s*2,0 | *s*2,1 | *s*2,2 | *s*2,3 |
| *s*3,0 | *s*3,1 | *s*3,2 | *s*3,3 |

|  |  |  |  |
| --- | --- | --- | --- |
| *out*0 | *out*4 | *out*0 | *out*0 |
| *out*1 | *out*5 | *out*9 | *out*13 |
| *out*2 | *out*6 | *out*10 | *out*14 |
| *out*3 | *out*7 | *out*11 | *out*15 |

Байты ключа

Inpu*tKey*

|  |  |  |  |
| --- | --- | --- | --- |
| *k*0 | *k*4 | *k*0 | *k*0 |
| *k*1 | *k*5 | *k*9 | *k*13 |
| *k*2 | *k*6 | *k*10 | *k*14 |
| *k*3 | *k*7 | *k*11 | *k*15 |

Четыре байта в каждом столбце матрицы состояний или ключа можно рассматривать как одно 32-х битовое ***слово***. Поэтому матрица

состояний – это массив из 4 слов *w*0 , *w*1, *w*2 , *w*3 , где

*w*0 = *s*0,0 *s*1,0 *s*2,0 *s*3,0 ;

*w*1 = *s*0,1 *s*1,1 *s*2,1 *s*3,1;

*w*2 = *s*0,2 *s*1,2 *s*2,2 *s*3,2 ;

*w*3 = *s*0,3 *s*1,3 *s*2,3 *s*3,3.

Матрица, поступающая на вход каждого раунда называется

матрицей Inpu*tState*, а на выходе раунда образуется матрица Outpu*tState*. Очевидно, на входе первого раунда Inpu*tState=* Input*Block* , а на выходе последнего раунда Outpu*tState=* Outpu*tBlock*.

**Общая структура AES**

Как отмечалось, в рассматриваемой версии алгоритма AES-128 ключ шифра состоит из 128 битов, поделенных на 16 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*0*k*1*k*2*k*3 ,

*w*1 = *k*4*k*5*k*6*k*7

и т.д.

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

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *w*0 | *w*1 | *w*2 | *w*3 | *w*4 | *w*5 | *w*6 | *w*7 | … | *w*42 | *w*43 |

Ключ первого раунда

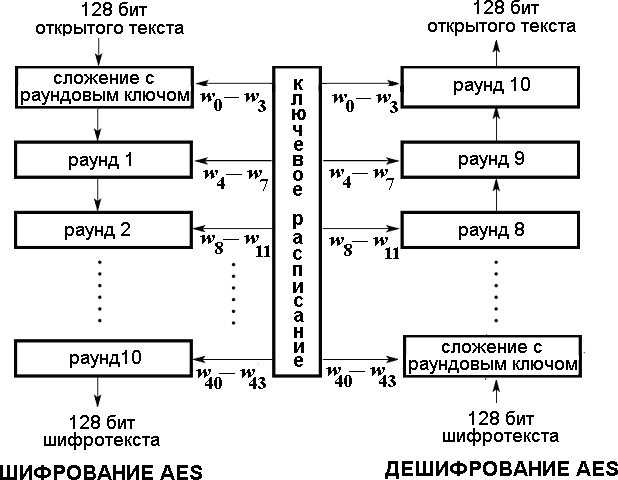


Ключ второго … раунда

Из этих слов с помощью специального алгоритма (о нем позже)

образуется последовательность из 44 слов: *w*0 , *w*1, *w*2 ,..., *w*43  (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 AddRound*Key* (суммирование по модулю 2 с начальным ключом шифра). Преобразования, выполненные в одном раунде, обозначают

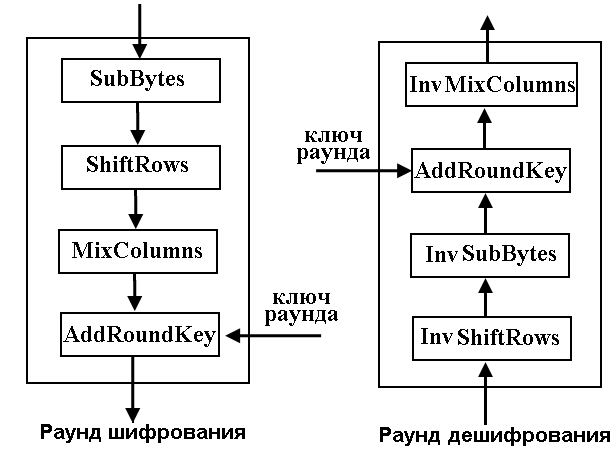
Round (*State*, *RoundKey*),

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.