Тема 2.3. Современные симметричные криптосистемы

AES

Advanced Encryption Standard (AES), также известный как Rijndael (произносится [rɛindaːl] (Рэндал)) — 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 AES (и только его) введена фирмой Intel в семейство процессоров x86 начиная с Intel Core i7-980X Extreme Edition, а затем на процессорах Sandy Bridge.

Январь 1997г. — НИСиТ США объявил о начале работ над новым алгоритмом шифрования с симметричным блочным ключом, призванным заменить DES.

20 августа 1998г. НИСиТ объявил о создании группы, состоящей из 15 алгоритмов, претендующих на роль нового стандарта AES. После первого раунда (до 15 апреля 1999г.) остались 5 финалистов — MARS, RC6, Rijndael, Serpent, Twofish. После 2-го раунда (до 15 мая 2000г.) институт получил всю необходимую информацию и 2 октября 2000г. Институт объявил победителя — Rijndael.

Описание алгоритма

128-битовый блок сообщения разбивается на 16 сегментов, каждый размеров в 1 байт:

InputBlock = m0m1m2...m15

Аналогичная операция выполняется над блоком ключа:

InputKey = k0k1k2...k15

Для внутреннего представления обычно используется матрица 4x4, куда записаны сегменты по столбцам.

И, как и в DES, над матрицей производятся вычисления, называемые раундами. Таких раундов для минимального варианта (128 сообщение и 128 ключ) выполняется 10 раундов. Назовем это функцией Round(State, RoundKey), где State — текущее состояние матрицы сообщения, а RoundKey — матрица, представляющая матрицу ключа раунда, которая создается из входного ключа с помощью расписания ключей (чуть позже). В первом раунде блок State совпадает с inputBlock. Преобразования в каждом раунде состоит из нескольких частей:

Round(State, RoundKey) {

SubBytes(State);

ShiftRows(State);

MixColumns(State);

AddRoundKey(State, RoundKey);

}

Послений раунд немного отличается от остальных и называется FinalRound(State, RoundKey). В нем не происходит функции MixColumns(State).

Все функции в раунде обратимые, чтобы можно было расшифровать сообщение, поэтому при дешифровке используются обратные функции в обратном порядке.

SubBytes

Первое преобразование, используется на стороне шифрования. Чтобы применить подстановку к байту, мы интерпретируем байт как две шестнадцатеричные цифры. Левая цифра определяет строку, а правая — колонку в таблице перестановки. На пересечении строки и колонки, обозначенных этими шестнадцатеричными цифрами, находится новый байт.

В преобразовании SubBytes состояние обрабатывается как матрица байтов 4 x 4. В один момент проводится преобразование одного байта. Содержание каждого байта изменяется, но расположение байтов в матрице остается тем же самым. В процессе преобразования каждый байт преобразуется независимо от других — это шестнадцать личных преобразований байт в байт.

Например байт 5A16 преобразуется в байт BE16.

Таблица преобразования SubBytes

ShiftRows

Эта функция применяется к каждой строке матрицы State. При этой трансформации строки состояния циклически сдвигаются на r байт по горизонтали влево, в зависимости от номера строки. Для нулевой строки r = 0, для первой строки r = 1 и т. д

MixColumns

В процедуре MixColumns, четыре байта каждой колонки State смешиваются, используя для этого обратимую линейную трансформацию. MixColumns обрабатывает состояния по колонкам, трактуя каждую из них как полином четвёртой степени. Т.е. если столбец имеет вид (s0, s1, s2, s3), то его можно представить в виде многочлена s(x) = s3x3 + s2x2 + s1x + s0. Теперь мы умножаем этот многочлен на другой фиксированный многочлен c(x) по модулю x4 + 1, где полином c(x) = c3x3 + c2x2 + c1x + c0 = 3x3 + x2 + x + 2. Как и shiftRows, функция MixColumns просто вносит диффузию в шифр.

В данном случае умножение полиномов можно выполнить перемножением матрицы на столбец матрицы State.

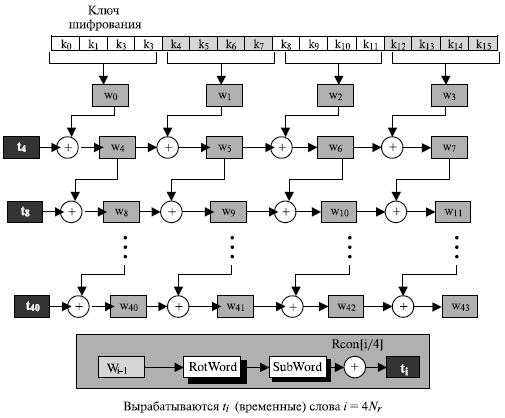


AddRoundKey

В процедуре AddRoundKey, RoundKey каждого раунда объединяется со State. Для каждого раунда Roundkey получается из InputKey используя процедуру KeyExpansion; каждый RoundKey такого же размера, что и State. Процедура производит побитовый XOR каждого байта State с каждым байтом RoundKey.

Расширение ключа

Алгоритм AES использует ключи шифрования трех фиксированных размеров: 128, 192 и 256 битов. В зависимости от размера ключа конкретный вариант алгоритма AES может обозначаться как AES-128, AES-192 и AES-256 соответственно.



Для того чтобы создать ключ для каждого раунда, AES использует процесс ключевого расширения. Если номер раунда — Nr, процедура расширения ключей создает Nr+1 ключи раунда на 128 бит от единственного 128-битового ключа шифра. Первые ключи раунда используются для преобразования перед раундом (AddRoundKey); остающиеся ключи раунда применяются для последнего преобразования (AddRoundKey) в конце каждого раунда.

Процедура расширения ключей создает слово за словом ключи раунда, где слово — массив из четырех байтов. Процедура создает 4 x (Nr + 1) слова, которые обозначаются

w0, w1, w2..., w4(Nr+1)-1

Другими словами, в версии AES-128 ( 10 раундов) имеются 44 слова; в AES-192 версии ( 12 раундов), есть 52 слова; и в AES 256 версий (с 14 раундами) есть 60 слов. Каждый ключ раунда состоит из четырех слов.

Процесс расширения ключа:

1. Первые четыре слова ( W0,W1,W2,W3 ) получены из ключа шифра. Ключ шифра представлен как массив из 16 байтов ( k0 до k15 ). Первые четыре байта ( k0 до k3 ) становятся W0 ; следующие четыре байта ( k4 до k7 ) становятся w1 ; и так далее. Другими словами, последовательное соединение (конкатенация) слов в этой группе копирует ключ шифра.
2. Остальная часть слов ( wi ) от i = 4 – 43 получается следующим образом:
   1. Если, , то это означает, что каждое слово получено из одного левого и одного верхнего.
   2. Если . Здесь t — временное слово, результат применения двух процессов, subword и rotword, со словом wi-1 и применения операции XOR c константой раунда Rcon. Другими словами, мы имеем 

**RotWord** (rotate word) — процедура, подобная преобразованию ShiftRows, но применяется только к одной строке. Процедура принимает слово как массив из четырех байт и сдвигает каждый байт влево с конвертированием.

**SubWord** (substitute word) — процедура, подобная преобразованию SubBytes, но применяется только к одной строке. Процедура принимает каждый байт в слове и заменяет его другим.

**Rcon.** Каждая константа раунда Rcon — это 4 -байтовое значение, в котором самые правые три байта являются всегда нулевыми. Таблица показывает значения для версии AES-128 (с 10 раундами).

