В конце каждого раунда, за исключением последнего, субблоки А (значение до описанной выше обработки) и С меняются местами; субблоки В (значение до обработки) и D также меняются местами.

Перед первым раундом выполняется входное отбеливание — наложение опе­рацией XOR на обрабатываемые субблоки четырех фрагментов расширенно­го ключа К0...К3. Аналогично выполняется выходное отбеливание— после заключительного раунда с использованием подключей К4...К7 .

Процедура расширения ключа

Процедура расширения ключа формирует 40 32-битных подключей для ис­пользования в 16 раундах алгоритма и для выполнения операций отбеливания. Кроме того, на основе ключа шифрования вычисляются таблицы замен S0...S3.

Как и Serpent (см. разд. 3.48), алгоритм Twofish использует ключи шифрова­ния любого размера до 256 битов включительно. Однако дополнение ключа выполняется несколько иначе: исходный ключ, при необходимости, дополня­ется нулевыми битами до ближайшего стандартного размера, т. е. до 128, 192 или 256 битов. Процедура расширения ключа обрабатывает дополненный таким образом ключ.

Сначала производится предварительная обработка ключа, включающая в се­бя следующие шаги:

1. Выполняется инициализация переменных, участвующих в дальнейших расчетах:

к = N/64,

где N — размер дополненного ключа шифрования в битах, т. е. к при­нимает значение 2, 3 или 4. Ключ шифрования представляется в виде

8к байтов m0...m8\*\_| или в виде 2к 32-битных слов, обозначаемых как М Q...M 2к-\ ■

2. Формируются 3 массива, каждый из которых состоит из к 32-битных слов:

Ме — *(Mq, М*

М0=(М1,М3,~.М2к\_]У,

V=(Vk-,^-2v..V0),

где:

= 287';

|  |  |  |
| --- | --- | --- |
| 1 | 7=0 | ( т \ т8/ |
|  |  | w8i+l |
| (v Л vi, 0 |  | та+2 |
| v«.l | = м2\* | ™8»+3 |
| v/,2 |  | "г8/+4 |
| Д'.З, |  | "18/+5 |
|  |  | "г8/+6 |
|  |  | lw8/+7y |

Матрица М2 приведена в табл. 3.137.

Таблица 3.137

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 01 | А4 | 55 | 87 | 5А | 58 | DB | 9Е |
| А4 | 56 | 82 | F3 | 1Е | С6 | 68 | Е5 |
| 02 | А1 | FC | С1 | 47 | АЕ | 3D | 19 |
| А4 | 55 | 87 | 5А | 58 | DB | 9Е | 03 |

Генерация подключей к0...к39 производится на основе вычисленных на предварительном этапе массивов Ме и М0 следующим образом:

к2i = Д + 5, mod 232;  
k2i+\ - (Д + 25, mod232)«<9,

где / = 0...19,а Д и 5, —промежуточные величины, вычисляемые так:

Д = h(2ip,Me)\

В; = h((2i + 1)р, ) «< 8.

Рис. 3.218. Функция h алгоритма Twofish Константа р определена следующим образом:

р = 224 +216 + 28 +1, а функция h() заслуживает подробного описания.

Эта функция схематично представлена на рис. 3.218. Она выполняется в не­сколько шагов, состав которых зависит от размера дополненного ключа в 64- битных фрагментах, т. е. от описанного выше значения к. В качестве пара-

метров функция h() принимает 32-битное слово и массив 32-битных слов

размерностью к. Последовательность выполнения такова:

Шаг 1. Входное слово делится на 4 8-битных фрагмента, которые «прогоня­ются» через специфические операции замены <у0 и Я\ (как показано на рис. 3.218). Результат замены объединяется в 32-битное слово, ко­торое складывается с третьим словом входного массива (на рис. 3.218 в качестве входного массива показан массив Ме). Данный шаг не выполняется, если к <4 (т. е. к = 2 или к = 3). На рис. 3.218 показа­ны все возможные шаги функции h().

Шаг 2. Результат предыдущего шага или входное слово обрабатывается ана­логичным образом (с другой последовательностью применения с/0 и <7,, которая различна для каждого шага — см. рис. 3.218), но склады­вается со вторым словом входного массива. Шаг 2 не выполняется, если к = 2.

Шаги 3 и 4. Выполняются всегда и включают в себя обработку результата предыдущего шага (или входного слова — для шага 3 и к = 2), ана­логичную предыдущим шагам с использованием, соответственно, первого и нулевого слов входного массива.

Шаг 5. Как и на предыдущих шагах, применяются замены г/0 и qx, после чего выполняется следующее преобразование:

|  |  |  |
| --- | --- | --- |
| ГО |  | V |
| К | = м\* | У1 |
| К | 1 | У2 |
| Л, |  |  |

Н = \*28',

i=0

где:

* Уо—Уз — байты результата выполнения замен на шаге 5;
* матрица Мх была описана выше;
* Н — выходное значение функции h().

Операции q0 и qx представляют собой не табличные замены 8x8 битов, а вычисляют выходные значения с использованием нескольких таблиц замен 4x4 следующим образом:

a0 = |\_\*/16\_|; b0 = jt mod 16;

«1 = «o©^

b{=а0® (^о »>4 0 © 8а0 mod 16;

fl2 =to(aO'■>

Ь2 = h (Ь\);

a3=a2®b2;

b3=a2® (b2 »>4 1) © 8a2 mod 16;

a4 = ^2(^3)’

Z?4 = f3(fc,);

у = 16b4 + «4,

где:

* \* — входное значение;
* у — выходное значение;

П »>4 — операция циклического вращения вправо 4-битных величин, т. е. раздельное вращение каждого нибла обрабатываемого байта;

П tj — табличные замены, различные для qQ и qj; tt для q0 представлена в табл. 3.138.

Таблица 3.138

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| г0 | 8 | 1 | 7 | D | 6 | F | 3 | 2 | 0 | В | 5 | 9 | Е | С | А | 4 |
| h | Е | С | В | 8 | 1 | 2 | 3 | 5 | F | 4 | А | 6 | 7 | 0 | 9 | D |
| h | В | А | 5 | Е | 6 | D | 9 | 0 | С | 8 | F | 3 | 2 | 4 | 7 | 1 |
| h | D | 7 | F | 4 | 1 | 2 | 6 | Е | 9 | В | 3 | 0 | 8 | 5 | С | А |

Выходное значение таблицы берется из позиции, соответствующей входному значению; например, Г, заменяет 0 на Е, 1 на С и т. д.

Таблицы для <7, приведены в табл. 3.139.

Таблица 3.139

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| {о | 2 | 8 | В | D | F | 7 | 6 | Е | 3 | 1 | 9 | 4 | 0 | А | С | 5 |
| h | 1 | Е | 2 | В | 4 | С | 3 | 7 | 6 | D | А | 5 | F | 9 | 0 | 8 |
| h | 4 | С | 7 | 5 | 1 | 6 | 9 | А | 0 | Е | D | 8 | 2 | В | 3 | F |
| h | В | 9 | 5 | 1 | С | 3 | D | Е | 6 | 4 | 7 | F | 2 | 0 | 8 | А |

Осталось описать зависимость от ключа таблиц замен SQ...S3. Фактически в алгоритме шифрования вместо описанной выше функции g() применяется функция h{), использующая в качестве входного значения 32-битный суб­блок А или В, а в качестве входного массива — описанный ранее массив V. Таким образом, замена выполняется на основе тех же таблиц r0..которые используются в процедуре расширения ключа.

Видно, что процедура расширения ключа является сложной и занимает не­сравнимо больше времени, чем шифрование одного блока данных. Авторы алгоритма в его спецификации [347] описали несколько возможных вариан­тов оптимизации процедуры расширения ключа, один из которых можно вы­брать в конкретной реализации алгоритма, учитывая следующие параметры:

* доступный объем энергонезависимой и оперативной памяти;
* предполагаемую частоту смены ключа при шифровании данных;

П требования к скорости шифрования блоков данных.

Общий принцип оптимизации состоит в поисках требуемого компромисса между скоростью шифрования и временем выполнения процедуры расшире­ния ключа: чем быстрее нужно шифровать, тем дольше будет выполняться расширение ключа, и наоборот.

Алгоритм Twofish-FK

Помимо описанного выше «стандартного» варианта алгоритма Twofish, авто­ры предложили еще и Twofish Family Key (или Twofish-FK)— шаблон для формирования на основе Twofish вариантов алгоритма, несовместимых меж­ду собой [347] и предназначенных, таким образом, для ограниченных приме­нений, например, в рамках конкретных организаций для защиты внутреннего документооборота.

Несовместимость вариантов достигается путем применения дополнительного ключа (FK), одинакового для всех субъектов, использующих конкретный ва­риант алгоритма Twofish-FK. То есть конкретное значение используемого FK формирует некий «контур совместимости» криптосредств, реализующих Twofish-FK.

Дополнительный ключ FK «подмешивается» на этапе расширения ключа следующим образом:

1. Инициализируются переменные, используемые в дальнейших операциях (этот шаг выполняется однократно для конкретного FK):

T0 = FK-

Т\ = (Efk (0) • Efk (1)) © FK;

*T2=Efk{*2>, h = EFK(3>,

T4 = Efk( 4)mod264,

где:

* EFK(N) — результат зашифровывания алгоритмом Twofish на ключе FK 128-битного блока, первый байт которого содержит значение /V, а остальные байты обнулены;
* • — операция конкатенации.

Стоит отметить, что переменные Т0 и Г, имеют размерность 256 битов,

Т2 и Тъ — по 128 битов, а ТА — 64 бита.

2. Наложение ключа FK выполняется однократно для каждого ключа шиф­рования, используемого совместно с данным FK.

* Перед выполнением процедуры расширения ключа шифрования произ­водится наложение Т0 на ключ шифрования операцией XOR. Если ключ шифрования имеет размер, меньший 256 битов, то используется необходимое количество битов Т0.
* После выполнения процедуры расширения ключа операциями XOR Т2 накладывается на подключи К0...К3, Тъ— на подключи КЛ...К1, а Г4 — на каждую пару подключей, используемых в раундах шифро­вания.
* Перед использованием табличных замен выполняется наложение опе­рацией XOR требуемого количества битов 7j на ключ шифрования.

Криптостойкость алгоритмов, построенных на основе Twofish-FK, по словам авторов алгоритма, полностью эквивалентна криптостойкости стандартного Twofish, поскольку подобное наложение дополнительного ключа никак не влияет на основные параметры алгоритма [347].

Достоинства и недостатки алгоритма

Сравнительный анализ достоинств и недостатков алгоритмов — финалистов конкурса AES приведен в разд. 2. /.