Алгоритм блочного симметричного шифрования Advanced Encryption Standard (AES)

Технический отчет CELLAES-01

К.С. Пан, М.Л. Цымблер

В данном документе описан алгоритм блочного симметричного шифрования Advanced Encryption Standard (AES) и режимы его применения.

1. Введение

AES [1] представляет собой алгоритм шифрования 128-битных блоков данных ключами по 128, 192 и 256 бит. AES является упрощенной версией алгоритма Rijndael [2]. Оригинальный алгоритм Rijndael отличается тем, что поддерживает более широкий набор длин блоков.

2. Терминология

Далее в изложении алгоритма нами будут использоваться следующие термины.

Байт — последовательность из 8 битов. В контексте данного алгоритма байт рассматривается как элемент поля Галуа. Операции над байтами производятся как над элементами поля Галуа GF(2^8) (конечного поля, [3]), то есть байту $\{b_7b_6b_5b_4b_3b_2b_1b_0\}$ соответствует многочлен $\sum_{i=0}^{7} b_i \cdot x^i$ в поле GF(2^8) [1].

Слово (word) — последовательность из 4 байтов.

Блок — последовательность из 16 байтов, над которой оперирует алгоритм. Блок служит входным и выходным данными алгоритма. Байты в блоке нумеруются с нуля.

Ключ — последовательность из 16, 24 или 32 байтов, используемая в качестве ключа шифрования. Байты в ключе нумеруются с нуля. Ключ, наряду с блоком, является входным данным алгоритма.

Форма (state) — двумерный массив байтов, состоящий из четырех строк. Байты в форме располагаются в порядке, изображенном в Табл. 1. В алгоритме AES форма используется для представления блока.

Табл. 1. Порядок байтов в форме

0	4	8	12	...
1	5	9	13	...
2	6	10	14	...
3	7	11	15	...

Раунд — итерация цикла преобразований над формой. В зависимости от длины ключа раундов может быть от 10 до 14, как показано в Табл. 2.

Ключ раунда (round key) — ключ, применяемый в раунде. Вычисляется для каждого раунда.

Таблица подстановок (S-box) — таблица, задающая биективное отображение байта в байт. Таблица подстановок представлена в Табл. 3.

Обратная таблица подстановок — таблица, задающая отображение, обратное задаваемому таблицей подстановок. Обратная таблица подстановок представлена в Табл. 4.

Nb — количество слов (word) в блоке.

Nk — количество слов в ключе. Nk может принимать значения 4, 6, 8.

Nr — количество раундов. Параметр Nr зависит от значений Nk. Соответствующие значения данных параметров приведены в Табл. 2.

Табл. 2. Зависимость Nr от Nk.

Nk	Nr
4	10
6	12
8	14

3. Шифрование

Для шифрования в алгоритме AES применяются следующие процедуры преобразования данных:

1. ExpandKey — Вычисление раундных ключей для всех раундов.
2. SubBytes — Подстановка байтов с помощью таблицы подстановок;
3. ShiftRows — Циклический сдвиг строк в форме на различные величины;
4. MixColumns — Смешивание данных внутри каждого столбца формы;
5. AddRoundKey — Сложение ключа раунда с формой.

Порядок выполнения процедур 2 и 3 можно поменять местами в силу опеределения этих операций.

Процедуры 4 и 5 тоже можно выполнять в разном порядке, но при этом изменяется количество их вызовов, поскольку MixColumns(AddRoundKey(A, B)) = AddRoundKey(MixColumns(A), MixColumns(B)).

Шифрование производится по алгоритму, приведенному на Рис. 1.

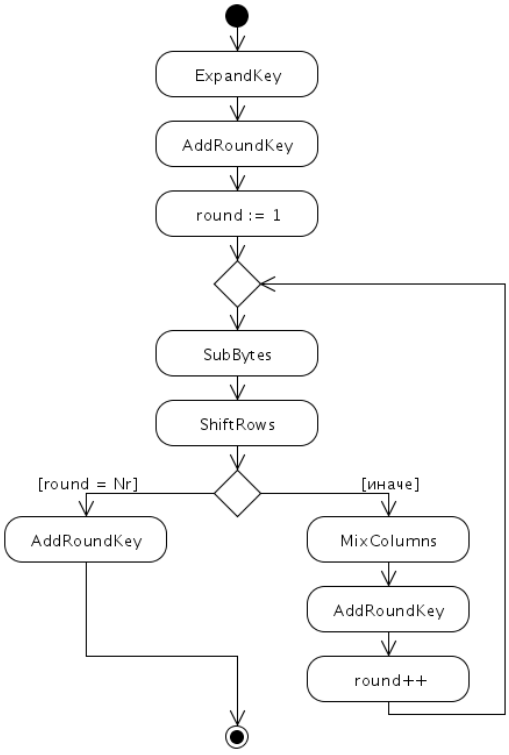

Рис. 1. Алгоритм шифрования

В Приложении 1 приведен пример работы алгоритма шифрования. Далее преобразования и их применение при шифровании рассмотрены подробнее.

3.1. Преобразование SubBytes

Преобразование SubBytes заключается в замене каждого байта {xy} формы (где x и y обозначают шестнадцатиричные цифры) на другой в соответствии с Табл. 3.

Табл. 3. Таблица подстановок

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Например, байт {fe} заменится на {bb}.

3.2. Преобразование ShiftRows

Преобразование ShiftRows заключается в циклическом сдвиге влево строк формы. Преобразование схематично представлено на Рис. 2. Первая строка остается неизменной. Во второй производится сдвиг на 1 байт, то есть первый байт переносится в конец. В третьей — сдвиг на 2 байта, в четвертой — на 3.

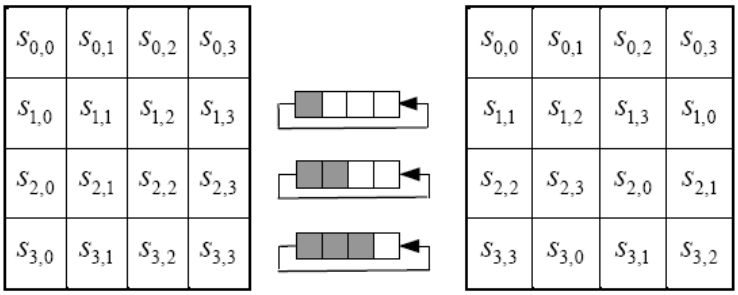

Рис. 2. Преобразование ShiftRows

3.3. Преобразование MixColumns

Преобразование MixColumns заключается в умножении квадратной матрицы 4-го порядка на каждый столбец формы:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Умножение производится в поле Галуа $GF(2^8)$.

Над каждым столбцом операция производится отдельно, как показано на Рис. 3.

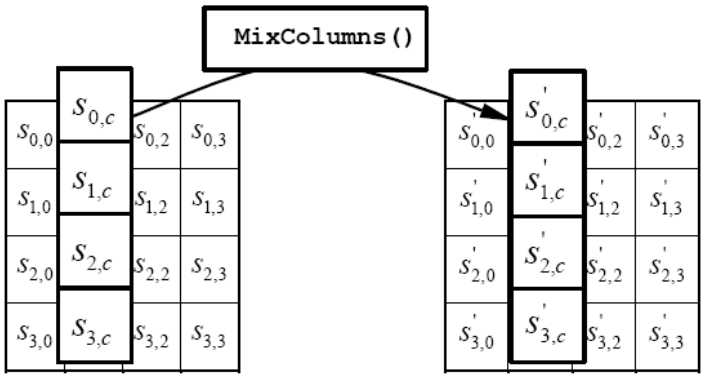

Рис. 3. Преобразование MixColumns

3.4. Преобразование AddRoundKey

В преобразовании AddRoundKey 32-битные слова раундного ключа прибавляются к столбцам формы с помощью побитовой операции XOR:

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus [w_{round*Nb+c}].$$

Здесь w_i — это столбцы ключа.

Над каждым столбцом операция производится отдельно, как показано на Рис. 4.

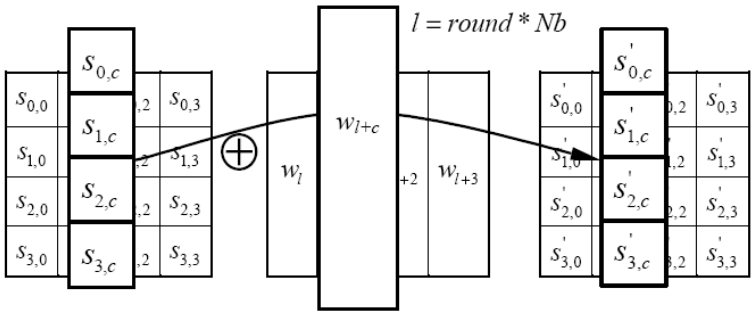

Рис. 4. Преобразование AddRoundKey

3.5. Процедура ExpandKey

В алгоритме AES генерируются раундные ключи на основе ключа шифрования с помощью процедуры ExpandKey. Процедура ExpandKey создает Nb * (Nr + 1) слов: алгоритму требуется начальный ключ размером Nb, плюс каждый из Nr раундов требует ключ из Nb слов. Ниже приведен псевдокод процедуры ExpandKey:

```
// Процедура вычисляет ключи раундов.
// key — ключ
// out — результат
// Nk — количество слов в ключе
ExpandKey(byte key[4*Nk], word out[Nb*(Nr+1)], int Nk)
begin
      i = 0

      while (i < Nk)
            out[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
            i = i + 1
      end while

      i = Nk
```

```
        while (i < Nb * (Nr+1))
              word temp = out[i-1]
              if (i mod Nk = 0)
                    temp = SubWord(RotWord(temp)) xor Rcon(i/Nk)
              else if ((Nk > 6) and (i mod Nk == 4))
                    temp = SubWord(temp)
              end if
              out[i] = out[i-Nk] xor temp
              i = i + 1
        end while
  end
```

Здесь использованы следующие функции:

SubWord осуществляет замену каждого байта в слове в соответствии с таблицей подстановок, представленной в Табл. 3.

RotWord осуществляет циклический сдвиг байтов в слове влево, как показано на Рис. 5.

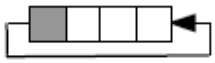

Рис. 5. Процедура RotWord

Rcon(i) формирует слово $[02^{i-1}, 00, 00, 00]$.

4. Дешифрование

При дешифровании все преобразования производятся в обратном порядке. Используются следующие обратные преобразования вместо соответствующих шифрующих:

- InvSubBytes — Подстановка байтов с помощью обратной таблицы подстановок;
- InvShiftRows — Циклический сдвиг строк в форме на различные величины;
- InvMixColumns — Смешивание данных внутри каждого столбца формы;

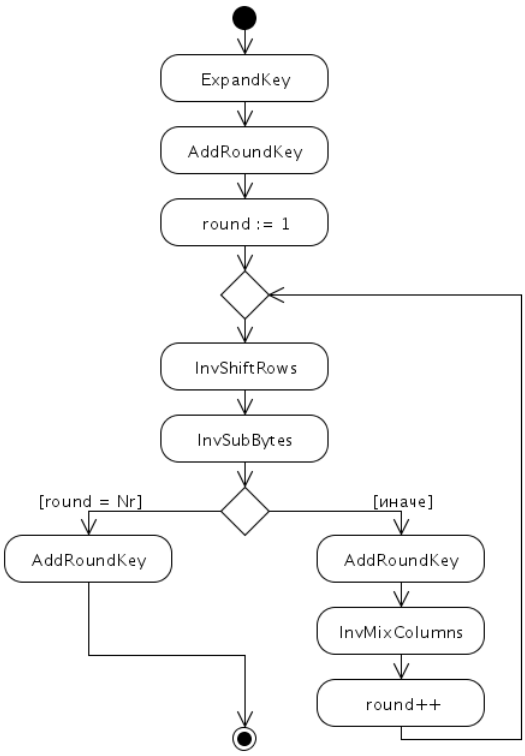

Рис. 6. Алгоритм дешифрования

Процедуры ExpandKey и AddRoundKey остаются неизменными. Ключи раунда используются в обратном порядке.

Алгоритм дешифрования представлен на Рис. 6. В Приложении 2 приведен пример работы алгоритма дешифрования.

4.1. Преобразование InvShiftRows

Это преобразование обратно преобразованию ShiftRows. Первая строка формы остается неизменной. Вторая строка циклически сдвигается вправо на 1 байт. Третья — на 2, четвертая — на 3. Схематично преобразование показано на Рис. 7.

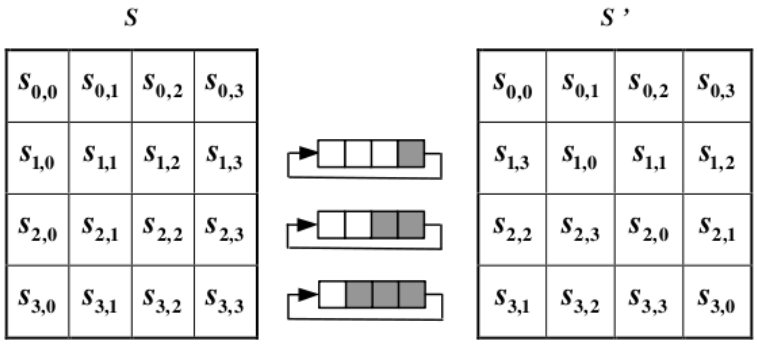

Рис. 7. Процедура InvShiftRows

4.2. Преобразование InvSubBytes

Это преобразование обратно преобразованию SubBytes. Подстановка байтов происходит аналогично с помощью обратной таблицы подстановок, представленной в Табл. 4.

Табл. 4. Обратная таблица подстановок

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

4.3. Преобразование InvMixColumns

Это преобразование обратно преобразованию MixColumns. InvMixColumns преобразует в форме каждый столбец отдельно. Преобразование происходит по следующией формуле:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Здесь умножение также производится в поле Галуа $GF(2^8)$.

5. Режимы применения

Далее описаны 5 режимов работы алгоритма, которые предусмотрены для алгоритмов блочного симметричного шифрования [4].

5.1. Режим ECB (Electronic Codebook)

В режиме ECB каждый блок шифруется независимо от других, как показано на Рис. 8. Таким образом, одинаковые блоки открытого текста преобразуются в одинаковые блоки зашифрованного текста.

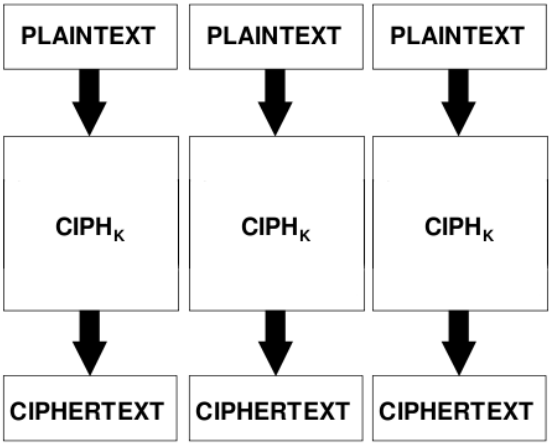

Рис. 8. Режим ECB

Дешифрование происходит по аналогичной схеме.

В режиме ECB можно производить шифрование и дешифрование нескольких блоков параллельно.

5.2. Режим CBC (Cipher Block Chaining)

В режиме CBC к каждому блоку открытого текста перед шифрованием прибавляется результат шифрования предыдущего блока с помощью побитовой операции XOR. К первому блоку открытого текста прибавляется Initialization Vector, который генерируется случайным образом и обычно передается вместе с шифрованными данными, чтобы их можно было дешифровать. Шифрование и дешифрование в режиме CBC показаны на Рис. 9 и Рис. 10.

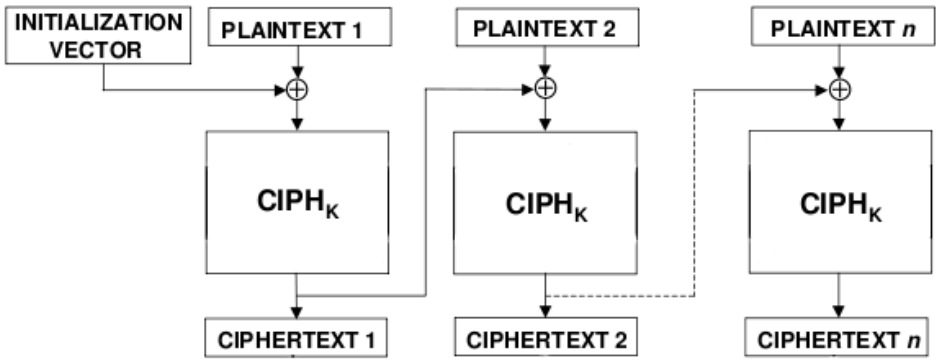

Рис. 9. Шифрование в режиме CBC

В режиме CBC для шифрования каждого следующего блока нужно иметь результат шифрования предыдущего блока, поэтому шифровать несколько блоков одновременно нельзя. Но можно производить дешифрование нескольких блоков параллельно, поскольку для дешифрования каждого блока нужно иметь только этот блок и предыдущий.

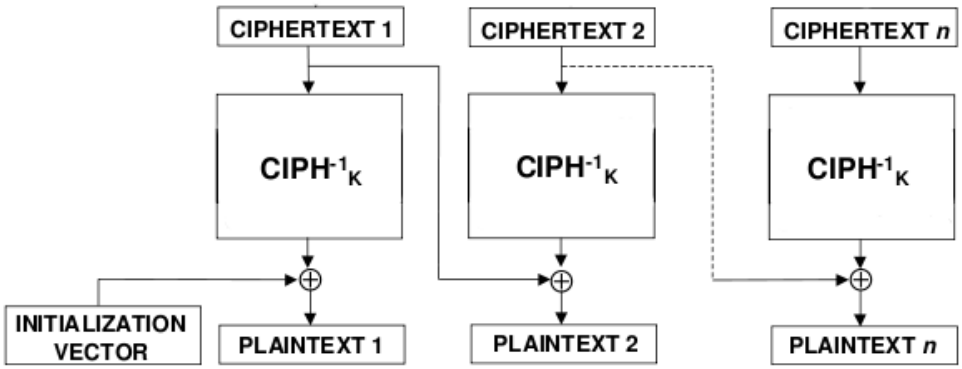

Рис. 10. Дешифрование в режиме CBC

5.3. Режим CFB (Cipher Feedback)

В режиме CFB при шифровании каждого блока используется только первые s битов результата. К ним прибавляются s битов открытого (или зашифрованного, при дешифрации) текста с помощью побитовой операции XOR.

В качестве первого входного блока используется Initialization Vector. Каждый следующий входной блок получается из предыдущего путем его побитового сдвига влево на s битов и прибавления первых s битов результата шифрования предыдущего блока. При дешифровании входные блоки генерируются точно так же.

Шифрование и дешифрование в режиме CFB представлены на Рис. 11.

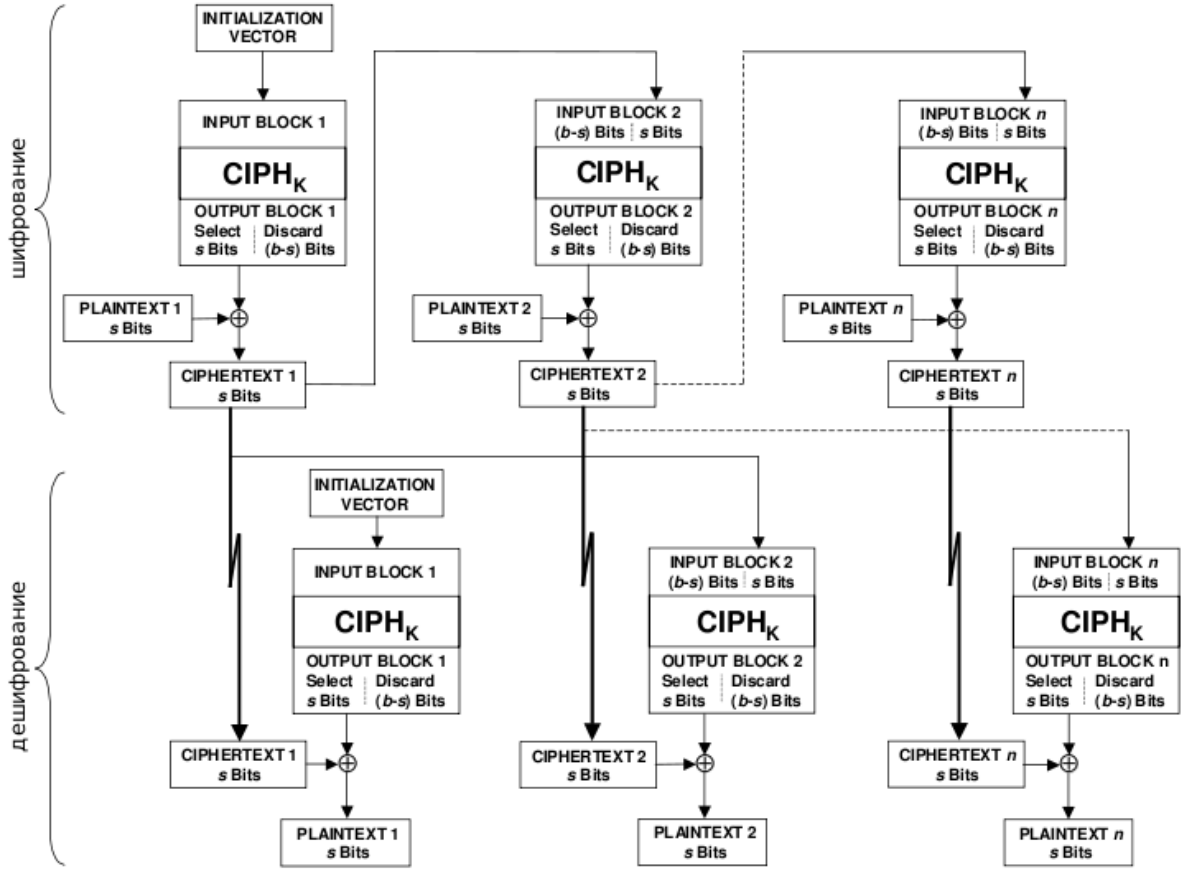

Рис. 11. Шифрование и дешифрование в режиме CFB

При шифровании в режиме CFB для применения $CIPH_K$ к каждому блоку требуются результаты шифрования предыдущего блока. Поэтому шифрование нескольких блоков

одновременно невозможно. При дешифровании все данные для получения открытого текста каждого блока уже есть — это зашифрованный текст данного и нескольких предыдущих блоков. Таким образом, возможна дешифрация нескольких блоков одновременно.

5.4. Режим OFB (Output Feedback)

В режиме OFB входным блоком служит результат применения $CIPH_K$ к предыдущему входному блоку. Первым входным блоком служит Initialization Vector.

Шифрование и дешифрование в режиме OFB представлены на Рис. 12 и Рис. 13.

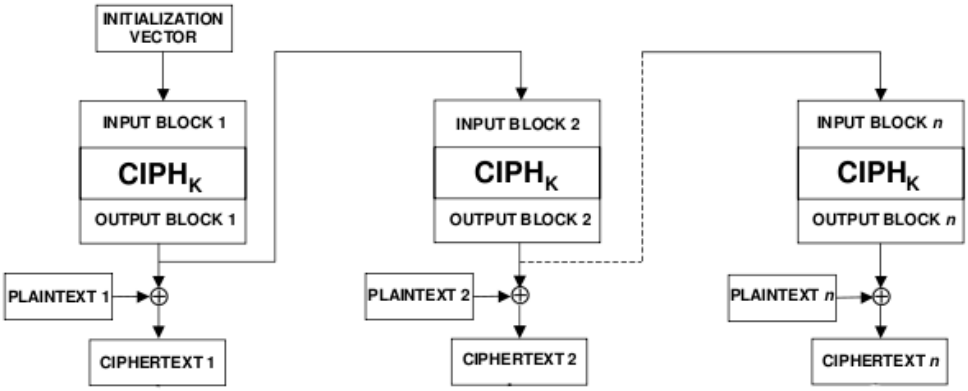

Рис. 12. Шифрование в режиме OFB

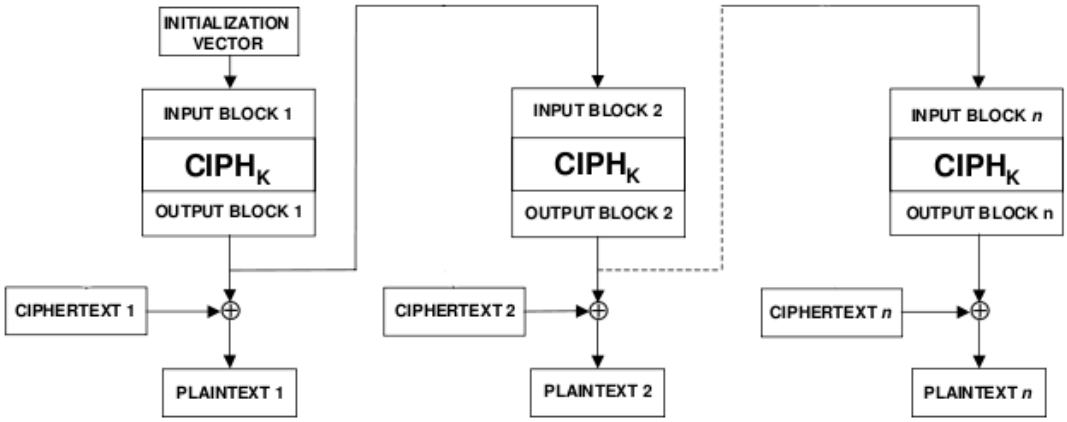

Рис. 13. Дешифрование в режиме OFB

Шифрование и дешифрование нескольких блоков одновременно произвести не получится, поскольку в обоих случаях для применения $CIPH_K$ к какому-либо блоку нужно применить $CIPH_K$ и ко всем предыдущим блокам.

5.5. Режим CTR (Counter)

В режиме CTR входными блоками являются значения некоторой функции T(i), называемой счетчиком, где i — номер блока.

Шифрование и дешифрование в режиме CTR представлены на Рис. 14 и Рис. 15.

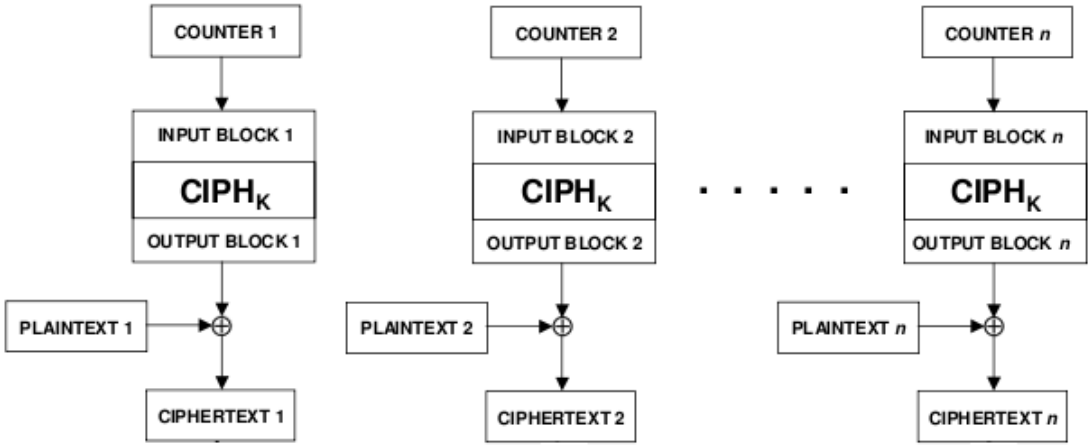

Рис. 14. Шифрование в режиме CTR

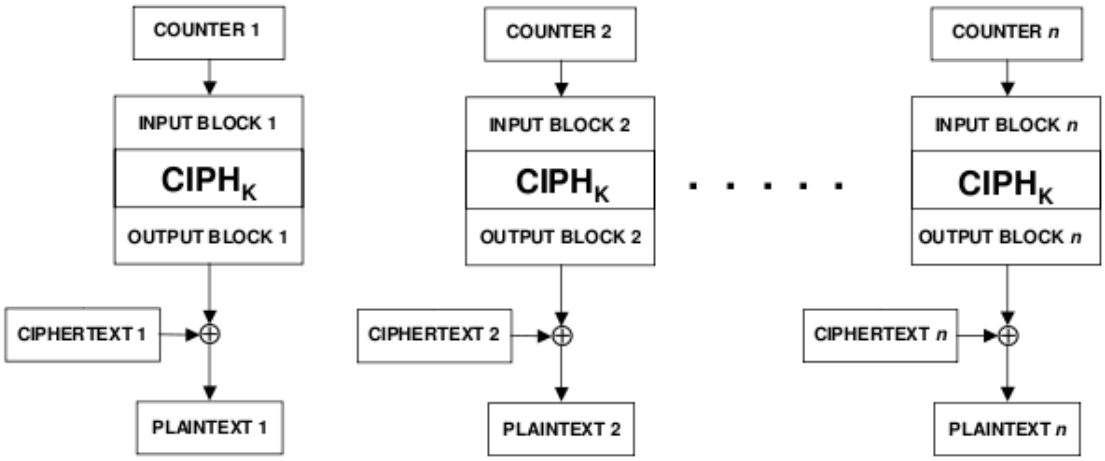

Рис. 15. Дешифрование в режиме CTR

Режим CTR допускает параллельное шифрование (и дешифрование) нескольких блоков.

Литература

1. Specification for the Advanced Encryption Standard (AES) [http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf], 26.11.2001.
2. *Daemen J., Rijmen V.* AES Proposal: Rijndael: [http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf], 09.03.1999
3. *Лидл Р., Нидеррайтер Г.* Конечные поля. В 2-х томах. -Москва: Мир, 1988-. с. 820.
4. *Dworkin M.* NIST Special Publication 800-38A, 2001 Edition. Recommendation for Block Cipher Modes of Operation. Methods and Techniques. [http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf] — National Institute of Standards and Technology. — Gaithersburg, 2001.

Приложение 1. Пример шифрования

В Табл. 1 приведен пример шифрования 128-битным ключом. Входной блок {00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff} шифруется ключом {00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f}.

Табл. 1. Пример шифрования

Раунд	Процедура	Результат
	входной блок	00112233445566778899aabbccddeeff
	ключ	000102030405060708090a0b0c0d0e0f
	AddRoundKey	00102030405060708090a0b0c0d0e0f0
1	SubBytes	63cab7040953d051cd60e0e7ba70e18c
	ShiftRows	6353e08c0960e104cd70b751bacad0e7
	MixColumns	5f72641557f5bc92f7be3b291db9f91a
	ключ раунда	d6aa74fdd2af72fadaa678f1d6ab76fe
	AddRoundKey	89d810e8855ace682d1843d8cb128fe4
2	SubBytes	a761ca9b97be8b45d8ad1a611fc97369
	ShiftRows	a7be1a6997ad739bd8c9ca451f618b61
	MixColumns	ff87968431d86a51645151fa773ad009
	ключ раунда	b692cf0b643dbdf1be9bc5006830b3fe
	AddRoundKey	4915598f55e5d7a0daca94fa1f0a63f7
3	SubBytes	3b59cb73fcd90ee05774222dc067fb68
	ShiftRows	3bd92268fc74fb735767cbe0c0590e2d
	MixColumns	4c9c1e66f771f0762c3f868e534df256
	ключ раунда	b6ff744ed2c2c9bf6c590cbf0469bf41
	AddRoundKey	fa636a2825b339c940668a3157244d17
4	SubBytes	2dfb02343f6d12dd09337ec75b36e3f0
	ShiftRows	2d6d7ef03f33e334093602dd5bfb12c7
	MixColumns	6385b79ffc538df997be478e7547d691
	ключ раунда	47f7f7bc95353e03f96c32bcfd058dfd
	AddRoundKey	247240236966b3fa6ed2753288425b6c
5	SubBytes	36400926f9336d2d9fb59d23c42c3950
	ShiftRows	36339d50f9b539269f2c092dc4406d23
	MixColumns	f4bcd45432e554d075f1d6c51dd03b3c
	ключ раунда	3caaa3e8a99f9deb50f3af57adf622aa
	AddRoundKey	c81677bc9b7ac93b25027992b0261996
6	SubBytes	e847f56514dadde23f77b64fe7f7d490
	ShiftRows	e8dab6901477d4653ff7f5e2e747dd4f
	MixColumns	9816ee7400f87f556b2c049c8e5ad036

Раунд	Процедура	Результат
	ключ раунда	5e390f7df7a69296a7553dc10aa31f6b
	AddRoundKey	c62fe109f75eedc3cc79395d84f9cf5d
7	SubBytes	b415f8016858552e4bb6124c5f998a4c
	ShiftRows	b458124c68b68a014b99f82e5f15554c
	MixColumns	c57e1c159a9bd286f05f4be098c63439
	ключ раунда	14f9701ae35fe28c440adf4d4ea9c026
	AddRoundKey	d1876c0f79c4300ab45594add66ff41f
8	SubBytes	3e175076b61c04678dfc2295f6a8bfc0
	ShiftRows	3e1c22c0b6fcbf768da85067f6170495
	MixColumns	baa03de7a1f9b56ed5512cba5f414d23
	ключ раунда	47438735a41c65b9e016baf4aebf7ad2
	AddRoundKey	fde3bad205e5d0d73547964ef1fe37f1
9	SubBytes	5411f4b56bd9700e96a0902fa1bb9aa1
	ShiftRows	54d990a16ba09ab596bbf40ea111702f
	MixColumns	e9f74eec023020f61bf2ccf2353c21c7
	ключ раунда	549932d1f08557681093ed9cbe2c974e
	AddRoundKey	bd6e7c3df2b5779e0b61216e8b10b689
10	SubBytes	7a9f102789d5f50b2beffd9f3dca4ea7
	ShiftRows	7ad5fda789ef4e272bca100b3d9ff59f
	ключ раунда	13111d7fe3944a17f307a78b4d2b30c5
	AddRoundKey	69c4e0d86a7b0430d8cdb78070b4c55a
	выходной блок	69c4e0d86a7b0430d8cdb78070b4c55a

Приложение 2. Пример дешифрования

В Табл. 1 приведен пример дешифрования 128-битным ключом. Блок {69 c4 e0 d8 6a 7b 04 30 d8 cd b7 80 70 b4 c5 5a} дешифруется ключом {00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f}.

Табл. 1. Пример дешифрования

Раунд	Процедура	Результат
	входной блок	69c4e0d86a7b0430d8cdb78070b4c55a
	ключ раунда	13111d7fe3944a17f307a78b4d2b30c5
	AddRoundKey	7ad5fda789ef4e272bca100b3d9ff59f
1	InvShiftRows	7a9f102789d5f50b2beffd9f3dca4ea7
	InvSubBytes	bd6e7c3df2b5779e0b61216e8b10b689
	ключ раунда	549932d1f08557681093ed9cbe2c974e
	AddRoundKey	e9f74eec023020f61bf2ccf2353c21c7
2	InvMixColumns	54d990a16ba09ab596bbf40ea111702f
	InvShiftRows	5411f4b56bd9700e96a0902fa1bb9aa1
	InvSubBytes	fde3bad205e5d0d73547964ef1fe37f1
	ключ раунда	47438735a41c65b9e016baf4aebf7ad2
	AddRoundKey	baa03de7a1f9b56ed5512cba5f414d23
3	InvMixColumns	3e1c22c0b6fcbf768da85067f6170495
	InvShiftRows	3e175076b61c04678dfc2295f6a8bfc0
	InvSubBytes	d1876c0f79c4300ab45594add66ff41f
	ключ раунда	14f9701ae35fe28c440adf4d4ea9c026
	ключ раунда	c57e1c159a9bd286f05f4be098c63439
4	InvMixColumns	b458124c68b68a014b99f82e5f15554c
	InvShiftRows	b415f8016858552e4bb6124c5f998a4c
	InvSubBytes	c62fe109f75eedc3cc79395d84f9cf5d
	ключ раунда	5e390f7df7a69296a7553dc10aa31f6b
	AddRoundKey	9816ee7400f87f556b2c049c8e5ad036
5	InvMixColumns	e8dab6901477d4653ff7f5e2e747dd4f
	InvShiftRows	e847f56514dadde23f77b64fe7f7d490
	InvSubBytes	c81677bc9b7ac93b25027992b0261996
	ключ раунда	3caaa3e8a99f9deb50f3af57adf622aa
	AddRoundKey	f4bcd45432e554d075f1d6c51dd03b3c
6	InvMixColumns	36339d50f9b539269f2c092dc4406d23
	InvShiftRows	36400926f9336d2d9fb59d23c42c3950
	InvSubBytes	247240236966b3fa6ed2753288425b6c
	ключ раунда	47f7f7bc95353e03f96c32bcfd058dfd

Раунд	Процедура	Результат
	AddRoundKey	6385b79ffc538df997be478e7547d691
7	InvMixColumns	2d6d7ef03f33e334093602dd5bfb12c7
	InvShiftRows	2dfb02343f6d12dd09337ec75b36e3f0
	InvSubBytes	fa636a2825b339c940668a3157244d17
	ключ раунда	b6ff744ed2c2c9bf6c590cbf0469bf41
	AddRoundKey	4c9c1e66f771f0762c3f868e534df256
8	InvMixColumns	3bd92268fc74fb735767cbe0c0590e2d
	InvShiftRows	3b59cb73fcd90ee05774222dc067fb68
	InvSubBytes	4915598f55e5d7a0daca94fa1f0a63f7
	ключ раунда	b692cf0b643dbdf1be9bc5006830b3fe
	AddRoundKey	ff87968431d86a51645151fa773ad009
9	InvMixColumns	a7be1a6997ad739bd8c9ca451f618b61
	InvShiftRows	a761ca9b97be8b45d8ad1a611fc97369
	InvSubBytes	89d810e8855ace682d1843d8cb128fe4
	ключ раунда	d6aa74fdd2af72fadaa678f1d6ab76fe
	AddRoundKey	5f72641557f5bc92f7be3b291db9f91a
10	InvMixColumns	6353e08c0960e104cd70b751bacad0e7
	InvShiftRows	63cab7040953d051cd60e0e7ba70e18c
	InvSubBytes	00102030405060708090a0b0c0d0e0f0
	ключ	000102030405060708090a0b0c0d0e0f
	AddRoundKey	00112233445566778899aabbccddeeff
	выходной блок	00112233445566778899aabbccddeeff