第四章 AES加密算法

- 4.1 AES算法简介
- 4.2 AES轮函数
- 4.3 AES密钥编排

4.1.1 AES提出的背景

- 随着对称密码的发展,DES数据加密标准算法由于密钥长度较小(56位),难以抵抗现有的攻击，因此不再作为加密标准。
- 1997年1月2日NIST宣布希望选择一个新的加密标准（AES），作为DES的继任者，但NIST仅要求有关各方就如何选择继任者提供方案。
- 1997年9月12日，NIST确定AES将是一种新的分组密码算法，支持128位的分组大小，支持128、192和256位的密钥长度，并公开征集新的数据加密标准。
- 共有15个算法被提交，经过三轮筛选,2000年8月2日，NIST宣布拟将比利时人Joan Daeman和VincentRijmen提交的Rijndael算法作为建议的AES标准。

4.1.1 AES提出的背景

- 2001年2月28日，Rijndael算法作为FIPS草案征求意见，并于当年11月批准为FIPS PUB 197。
- 2002年5月26日,此算法FIPS PUB 197成为有效的标准。
- Rijndael算法已成为对称加密中最流行的算法之一而被广泛应用在各个领域中。

4.1.1 AES提出的背景

- 严格的说，AES和Rijndael算法并不完全一样。
  - AES的分组长度固定位128位，密钥长度可以是128、192或256位
  - Rijndael算法的分组长度和密钥长度可以是128、160、192、224或256位。

4.1.2 AES算法框架和参数说明

- 设计思想

（1） Rijndael算法使用的是代换-置换网络，并没有采用Feistel结构，在软件和硬件上都能快速地加解密，相对来说易于实现，且只需要很少的寄存器；

（2）在多个平台上速度快，编码紧凑；

（3）抵抗所有已知的攻击；

（4）轮函数由3个不同的可逆均匀变换构成，称为3个层。

4.1.2 AES算法框架和参数说明

- 轮函数的3层

（1）线性混合层：确保多轮之上的高度扩散；

（2）非线性层：将具有最优的“最坏情况非线性特性”的S盒并行使用；

（3）密钥加层：单轮子密钥简单的异或到中间状态上，实现一次性的掩盖。

4.1.2 AES算法框架和参数说明

- 算法说明

（1）明文分组可变，128、192、256比特。

（2）密钥长度可变，各自可独立指定为128、192、256比特。

（3）状态。算法中间的结果也需要分组，称之为状态，状态可用以字节为元素的矩阵阵列表示，该阵列有4行，列数N_b为分组长度除32。

（4）种子密钥。以字节为元素的矩阵阵列描述，阵列为4行，列数N_k为密钥长度除32.

（5）算法的输入、输出和种子密钥可看成字节组成的一维数组。

（6）下标范围。输入输出：0—$4N_b$-1；种子密钥0—$4N_k$-1。

4.1.2 AES算法框架和参数说明

- N_b=6和N_k=4的状态密钥阵列

a_{00}	a_{01}	a_{02}	a_{03}	a_{04}	a_{05}
a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
a_{20}	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}
a_{30}	a_{31}	a_{32}	a_{33}	a_{34}	a_{35}

按此顺序放入和读出

k_{00}	k_{01}	k_{02}	k_{03}
k_{10}	k_{11}	k_{12}	k_{13}
k_{20}	k_{21}	k_{22}	k_{23}
k_{30}	k_{31}	k_{32}	k_{33}

按此顺序放入

4.1.2 AES算法框架和参数说明

- 分组和阵列中元素对应关系

（1）分组下标n

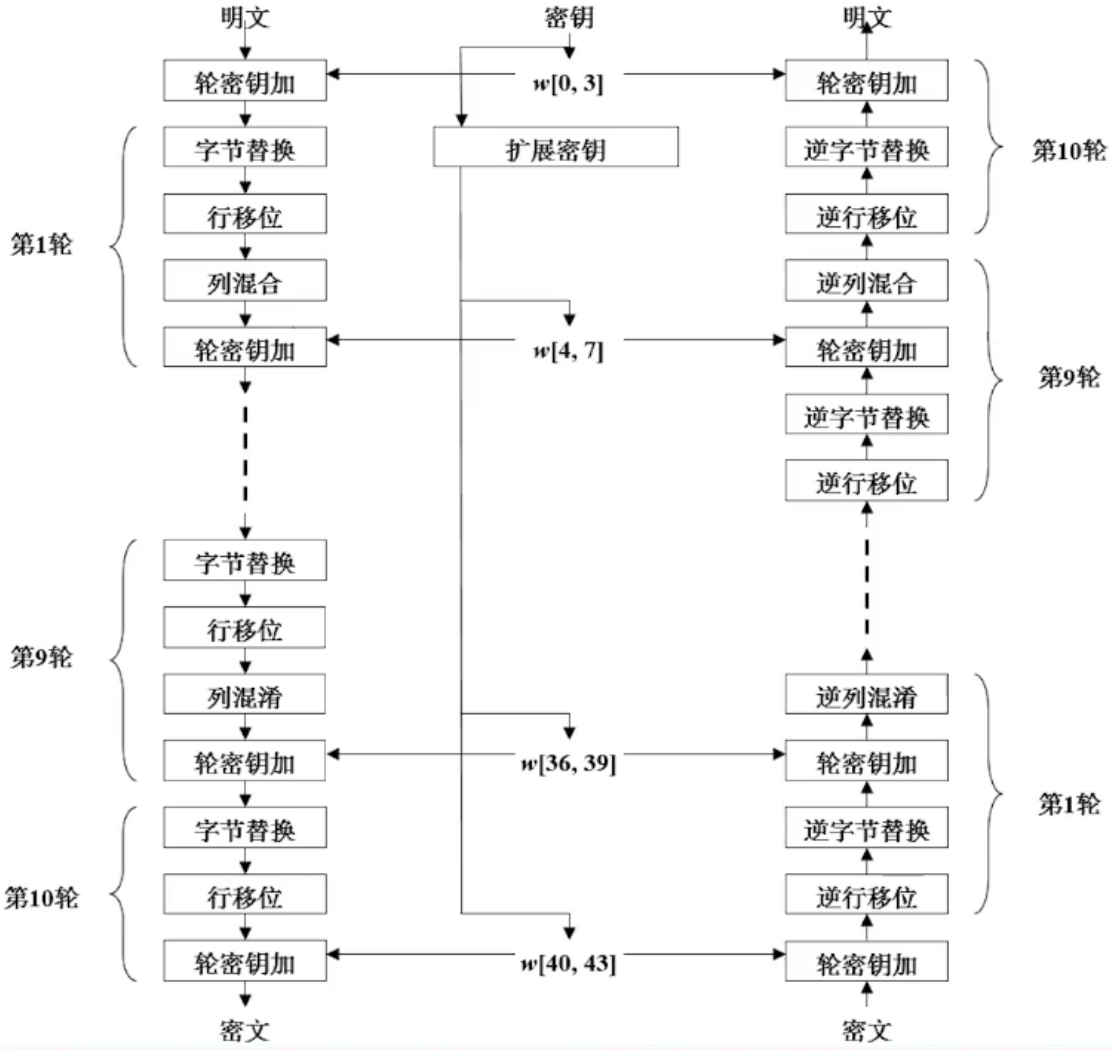
（2）阵列位置(i, j)。 i = n mod 4, j=[n/4]; n=i+4j

（3）轮数N_r与N_b和N_k对应关系如下表所示：

	N_b=4	N_b=6	N_b=8
N_k=4	10	12	14
N_k=6	12	12	14
N_k=8	14	14	14

4.1.2 AES算法框架和参数说明

- 10轮AES算法加密解密过程



4.2.1 AES字节代换

- 非线性代换，独立地对状态的每个字节进行，并且代换表（S盒）可逆，记为ByteSub(State)，分两步

（1）将输入的字节作为GF(2⁸)上的元素映射到自己的逆元（生成多项式：$x^8+x^4+x^3+x+1$）

（2）将该逆元字节做GF(2)上的仿射变换

即 $y=Ax^{-1}+B$ ，其中A是GF(2)上的一个8×8的可逆矩阵，B是GF(2)上的一个8位列向量

4.2.1 AES字节代换

- 字节代换

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

4.2.1 AES字节代换

- AES的S盒的使用：输入8a，输出7e，即7e=S(8a)

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	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

4.2.1 AES字节代换

- 逆字节替代变换是字节替代变换的逆变换，在状态的每个字节上应用逆S盒。

这是通过应用字节替代变换中的仿射变换的逆变换，再对所得结果应用有限域的乘法逆运算得到的，即 $y = A^{-1}(x - B)$

4.2.1 AES字节代换

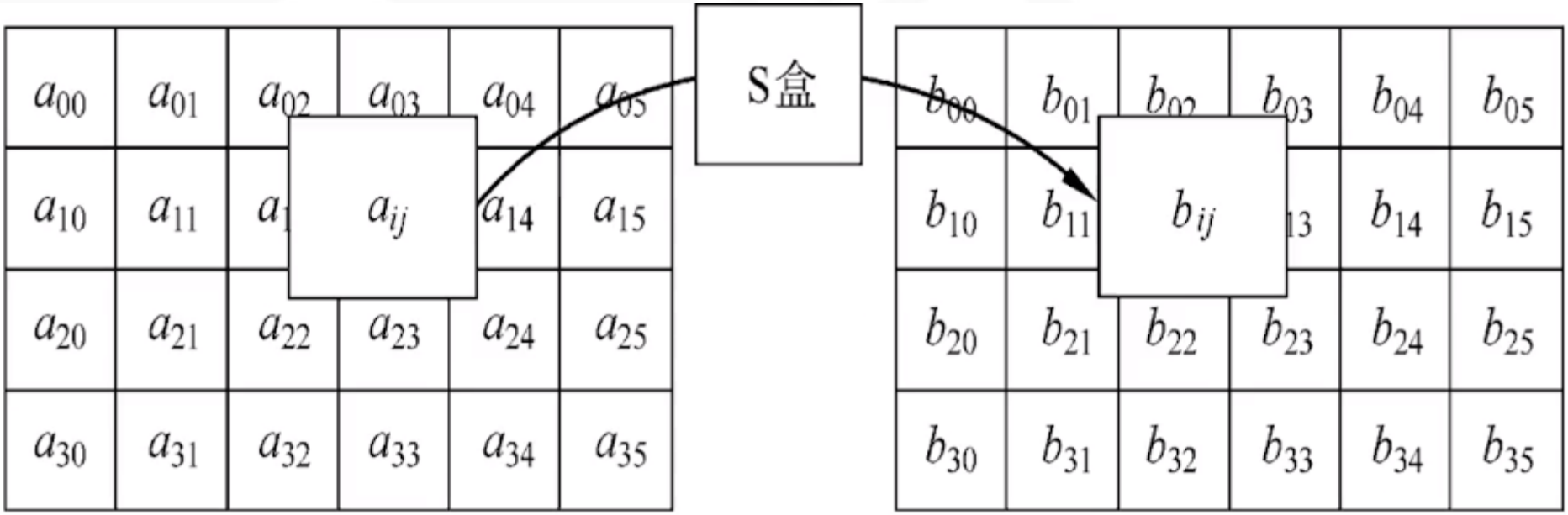
- AES的逆S盒的使用：输入7e，输出8a

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	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

4.2.1 AES字节代换

- 字节代换示意图

上述S一盒对状态的所有字节所做的变换记为ByteSub(State)



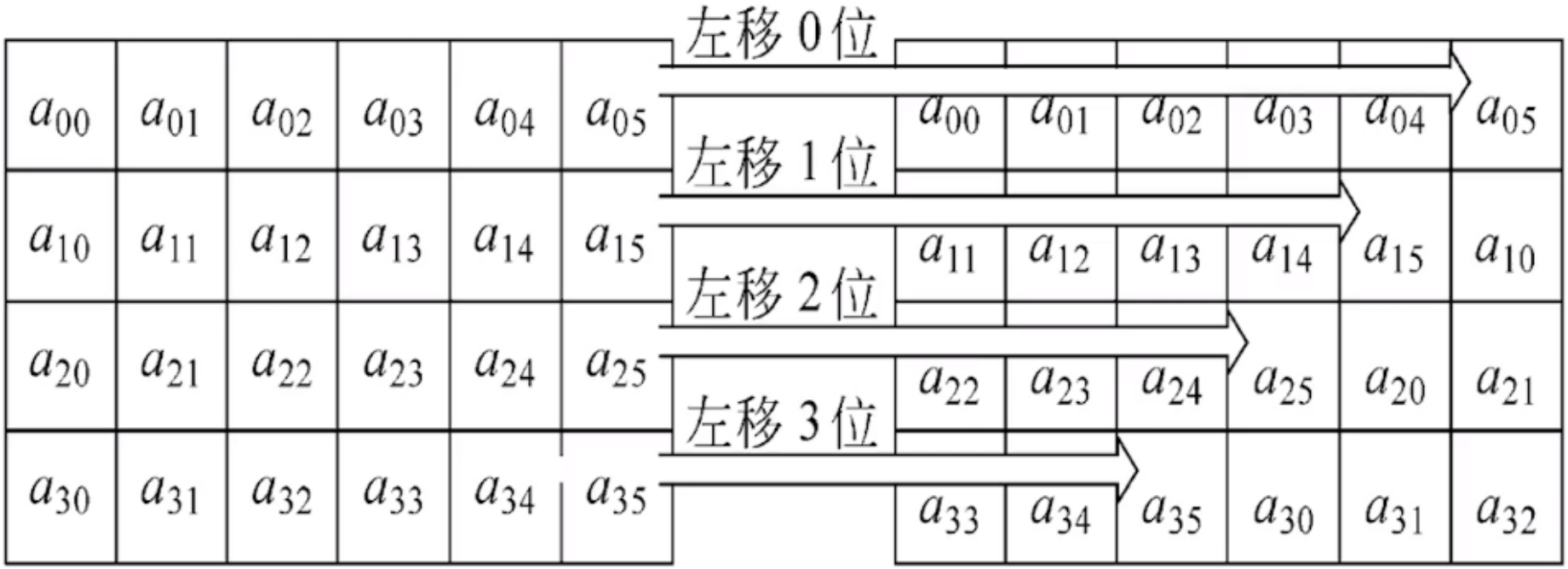
4.2.2 AES行移位

- 将状态阵列的各行进行循环移位，不同的行的移位量不同
- 0行：不动
- 1行：循环左移C_1字节
- 2行：循环左移C_2字节
- 3行：循环左移C_3字节
- 记为：ShiftRow(State)

N_b	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

4.2.2 AES行移位

- 行移位示意图（分组长度为192bit时）



4.2.3 AES列混淆

- 将每列视为GF(2^8)上的多项式，与固定的多项式c(x)进行模x^4+1乘法，记为⊗，要求c(x)模x^4+1可逆。
- 表示为MixColumn(State)

$$c(x)='03'x^3+'01'x^2+'01'x+'02'$$

4.2.3 AES列混淆

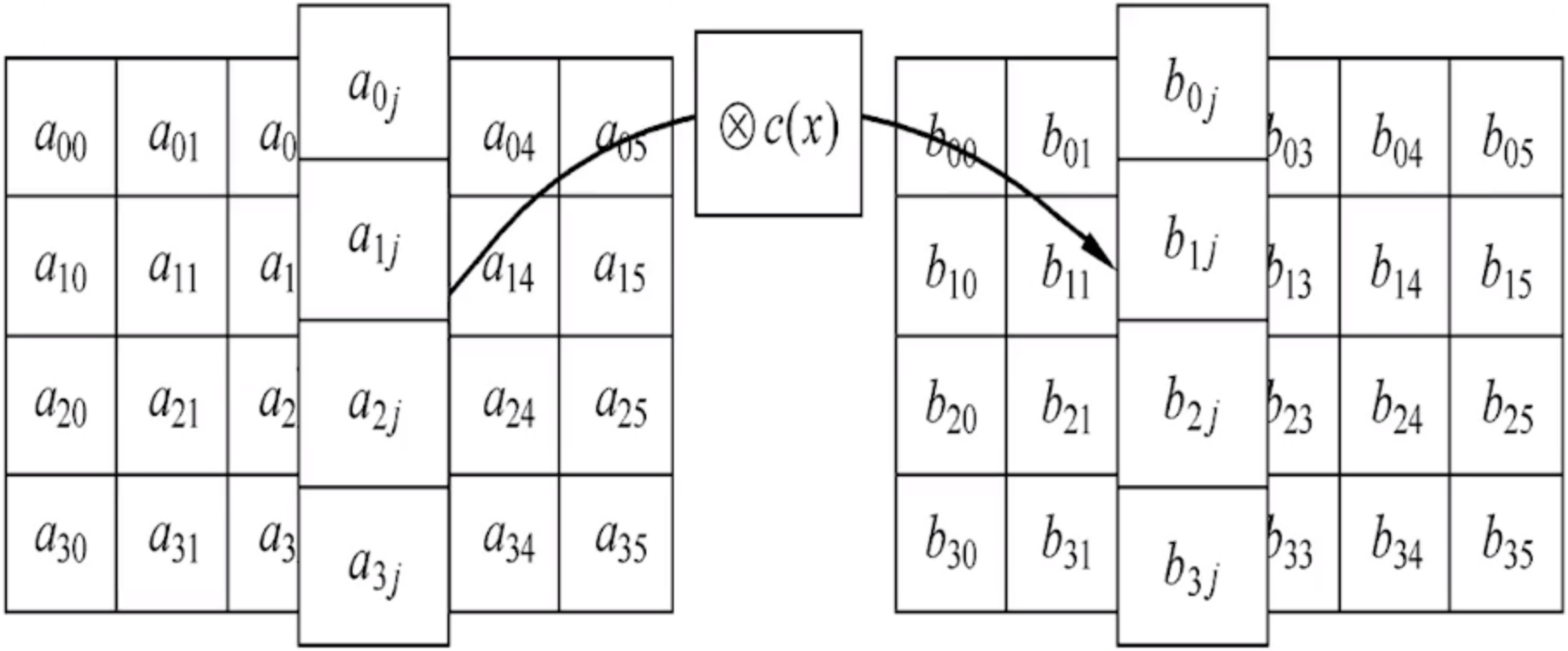
- 列混淆的矩阵表示

  列混淆运算也可写为矩阵乘法。设 $b(x)=c(x)\otimes a(x)$，则

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

4.2.3 AES列混淆

- 列混淆示意图



4.2.3 AES列混淆

- 逆列混淆变换是列混淆变换的逆
- 它将状态矩阵中的每一列视为系数在GF(2⁸)上的次数小于4的多项式与同一个固定的多项式d(x)相乘。d(x)满足

$$('03'x^3+'01'x^2+'01'x+'02')\otimes d(x)='01'$$

由此可得

$$d(x)='0B'x^3+'0D'x^2+'09'x+'0E'$$

4.2.3 AES列混淆

- 逆列混淆的矩阵表示

  逆列混淆运算也可写为矩阵乘法。

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_2 \end{pmatrix} = \begin{pmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

4.2.4 AES轮密钥加

- 轮密钥与状态进行逐比特异或。
- 轮密钥由种子密钥通过密钥编排算法得到。
- 轮密钥长度与分组长度相同。
- 表示为AddRoundKey(State,RoundKey)

4.3 AES密钥编排

- 密钥编排指从种子密钥得到轮密钥的过程，它由密钥扩展和轮密钥选取两部分组成。其基本原则如下：

（1）轮密钥的比特数等于分组长度乘以轮数加1；例如将128比特的明文经过10轮的加密，则总共需要（10+1）*128=1408比特的密钥；

（2）种子密钥被扩展成为扩展密钥；

（3）轮密钥从扩展密钥中取，其中第1轮轮密钥取扩展密钥的前N_b个字，第2轮轮密钥取接下来的N_b个字，如此下去。

4.3 AES密钥编排

- 密钥扩展

  扩展密钥是以4字节字为元素的一维阵列，表示为$W[N_b*(N_r+1)]$，其中前N_k个字取为种子密钥，以后每个字按递归方式定义。扩展算法根据$N_k \le 6$和$N_k > 6$有所不同。

w_0	w_1	w_2	w_3	w_4	w_5	……	
k_{00}	k_{01}	k_{02}	k_{03}	k_{04}	k_{05}		
k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}		
k_{20}	k_{21}	k_{22}	k_{23}	k_{24}	k_{25}		
k_{30}	k_{31}	k_{32}	k_{33}	k_{34}	k_{35}		

……

4.3 AES密钥编排

- N_k≤6的情况下的算法(RotByte为字循环移位，SubByte为S盒变换 Rcon为轮常数)

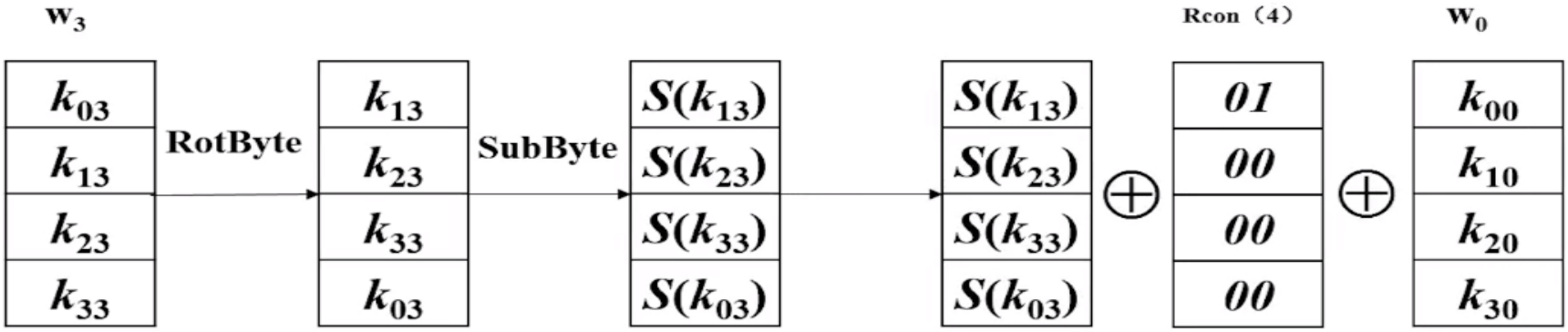
```
KeyExpansion (byteKey[4*Nk] , W[Nb*(Nr+1)])
{
   for (i =0; i < Nk; i ++)
                W[i]=(Key[4* i],Key[4* i +1],Key[4* i +2],Key[4* i +3] );
   for (i =Nk; i <Nb*(Nr+1); i ++)
   {
                temp=W[i-1];
                if (i % Nk= =0)
                        temp=SubByte (RotByte (temp))^Rcon[i /Nk];
                W[i]=W[i-Nk]^ temp;
   }
}
```

4.3 AES密钥编排

- $N_k \leq 6$的情况下的算法举例（$N_k=4$）

w_0	w_1	w_2	w_3	w_4	w_5	.....	
k_{00}	k_{01}	k_{02}	k_{03}	k_{04}	k_{05}		
k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}		
k_{20}	k_{21}	k_{22}	k_{23}	k_{24}	k_{25}		
k_{30}	k_{31}	k_{32}	k_{33}	k_{34}	k_{35}		

.....



4.3 AES密钥编排

- $N_k \leq 6$的情况下的算法举例（$N_k=4$）

w_0	w_1	w_2	w_3	w_4	w_5	……	
k_{00}	k_{01}	k_{02}	k_{03}	k_{04}	k_{05}		
k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}		
k_{20}	k_{21}	k_{22}	k_{23}	k_{24}	k_{25}		
k_{30}	k_{31}	k_{32}	k_{33}	k_{34}	k_{35}		

……

w_1		w_4		w_5
k_{01}		k_{04}		k_{05}
k_{11}	$\oplus$	k_{14}	$=$	k_{15}
k_{21}		k_{24}		k_{25}
k_{31}		k_{34}		k_{35}

4.3 AES密钥编排

- N_k>6的情况下的算法(RotByte为字循环移位，SubByte为S盒变换 Rcon为轮常数)

```
KeyExpansion (byte  Key[4*Nk] , W[Nb*(Nr+1)])
{
        for (i=0; i < Nk; i ++)
                W[i]=(Key[4* i], Key[4* i +1], Key[4* i +2], Key[4* i +3] );
for (i =Nk; i <Nb*(Nr+1); i ++)
  {
                temp=W[i -1];
                if (i % Nk= =0)
                   temp=SubByte (RotByte (temp))^Rcon[i /Nk];
    else if (i % Nk==4)
       temp=SubByte (temp);
W[i]=W[i - Nk]^ temp;
  }
}
```

N_k>6与N_k≤6的密钥扩展算法的区别在于：当i-4为N_k的整数倍时，须先将前一个字W[i-1]经过SubByte变换

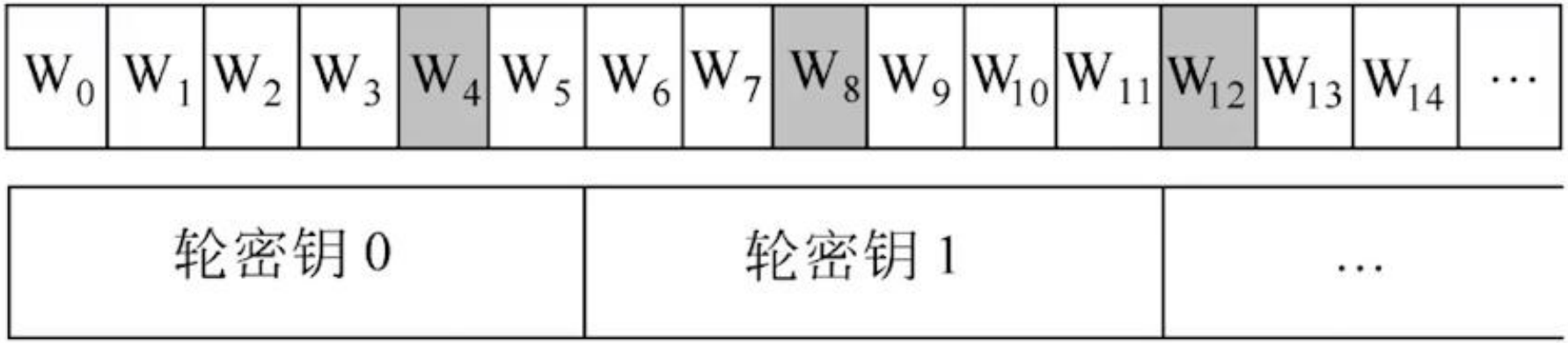
4.3 AES密钥编排

- 以上两个算法中，Rcon[i/N_k]为轮常数，其值与N_k无关，定义为（字节用十六进制表示，同时理解为GF(2^8)上的元素）：
- Rcon[i] = (RC[i],'00','00','00')
- 其中RC[i]是GF(2^8)中值为x^{i-1}的元素，因此
- RC[1]=1（即'01'）
- RC[i]=x*RC[i-1]=x^{i-1}

4.3 AES密钥编排

- 轮密钥选取

  轮密钥i（即第i个轮密钥）由轮密钥缓冲字$W(N_b*i)$到$W(N_b*(i+1))$给出，如下图所示。



N_b=6且N_k=4时的密钥扩展与轮密钥选取

问题：实现AES算法。