PART 6

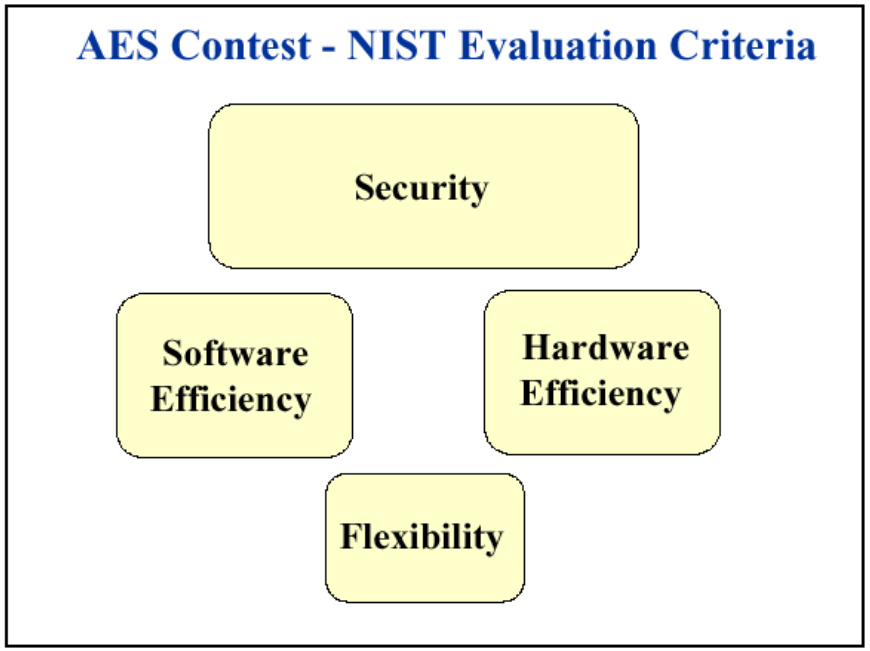
高级加密标准

目录 CONTENTS

6.1 AES简介

AES的由来

- 由于DES作为数据加密标准自1977年颁布以来，已超期服役了很长时间，1997年美国国家标准技术研究所NIST向密码学界征寻一个用于新的数据加密标准，名为高级加密标准AES候选算法的提议。



- NIST规定候选算法必须满足下面的要求：

- 1）密码必须是没有密级的，绝不能像商业秘密那样来保护它。
- 2）算法的全部描述必须公开披露。
- 3）密码必须可以在世界范围内免费使用。
- 4）密码系统支持至少128bit长的分组。
- 5）密码支持的密钥长度至少为128bit、192bit和256bit。
- 6)安全性不低于3DES。

15个候选算法

- 1998 年8 月20 日：NIST 召开了第一次候选大会，并公布了15 个候选算法。

AES: Candidate algorithms

North America (8)

Canada:
CAST-256
Deal

USA:
Mars
RC6
Twofish
Safer+
HPC

Costa Rica:
Frog

Europe (4)

Germany:
Magenta

Belgium:
Rijndael

France:
DFC

Israel, GB, Norway:
Serpent

Asia (2)

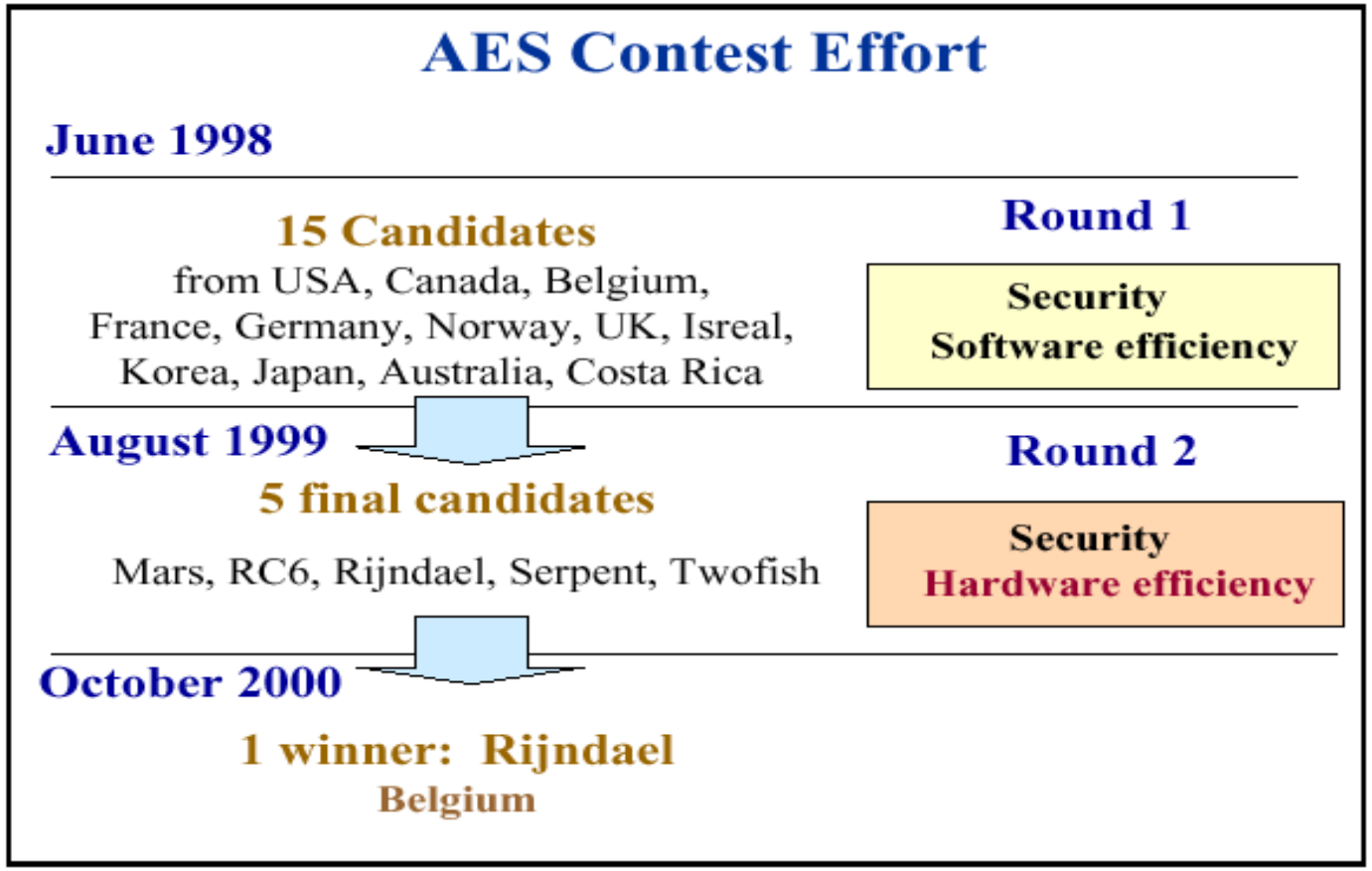
Korea:
Crypton

Japan:
E2

Australia (1)

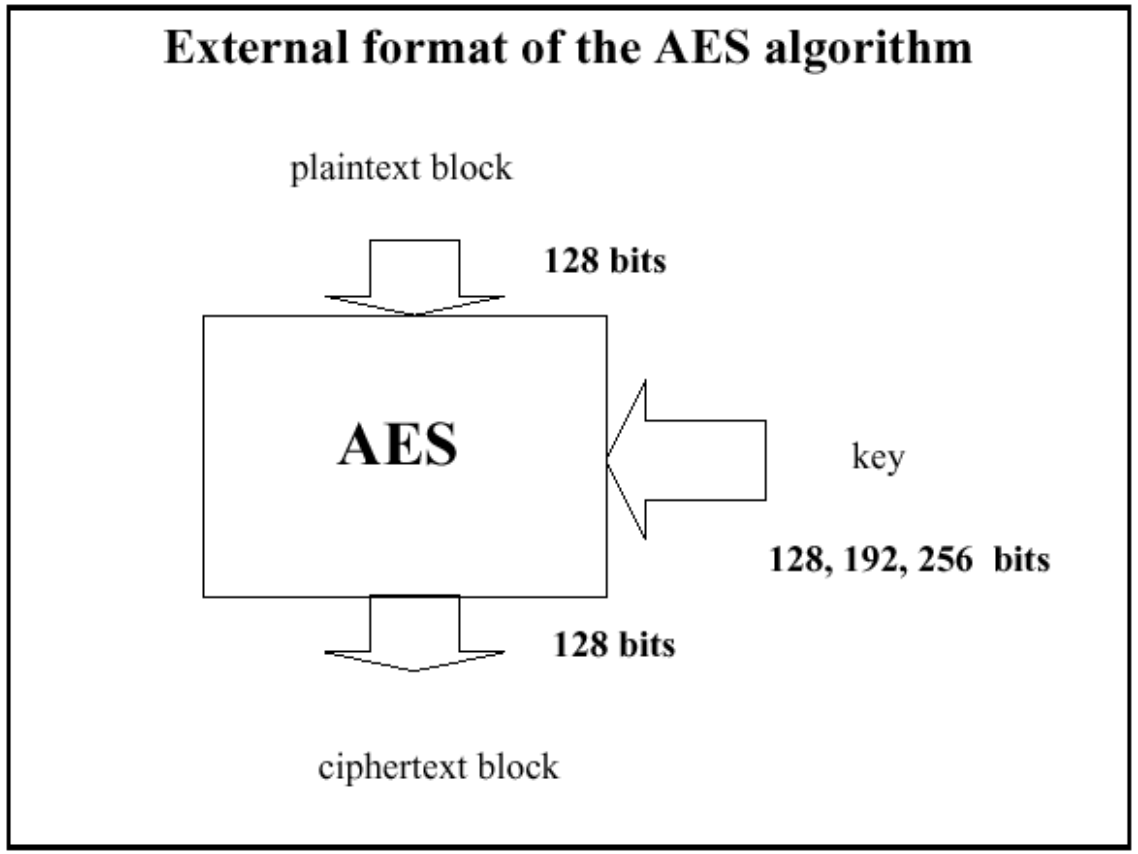
Australia:
LOKI97

- 在1999年8月9日，NIST宣布已经选出了5个候选算法进行第二轮的选择，它们是MARS、RC6、Serpent 、Twofish 、Rijndael。
- 2000年10月2日，由比利时的Joan Daeman和Vincent Rijmen设计的Rijndael数据加密算法最终获胜。

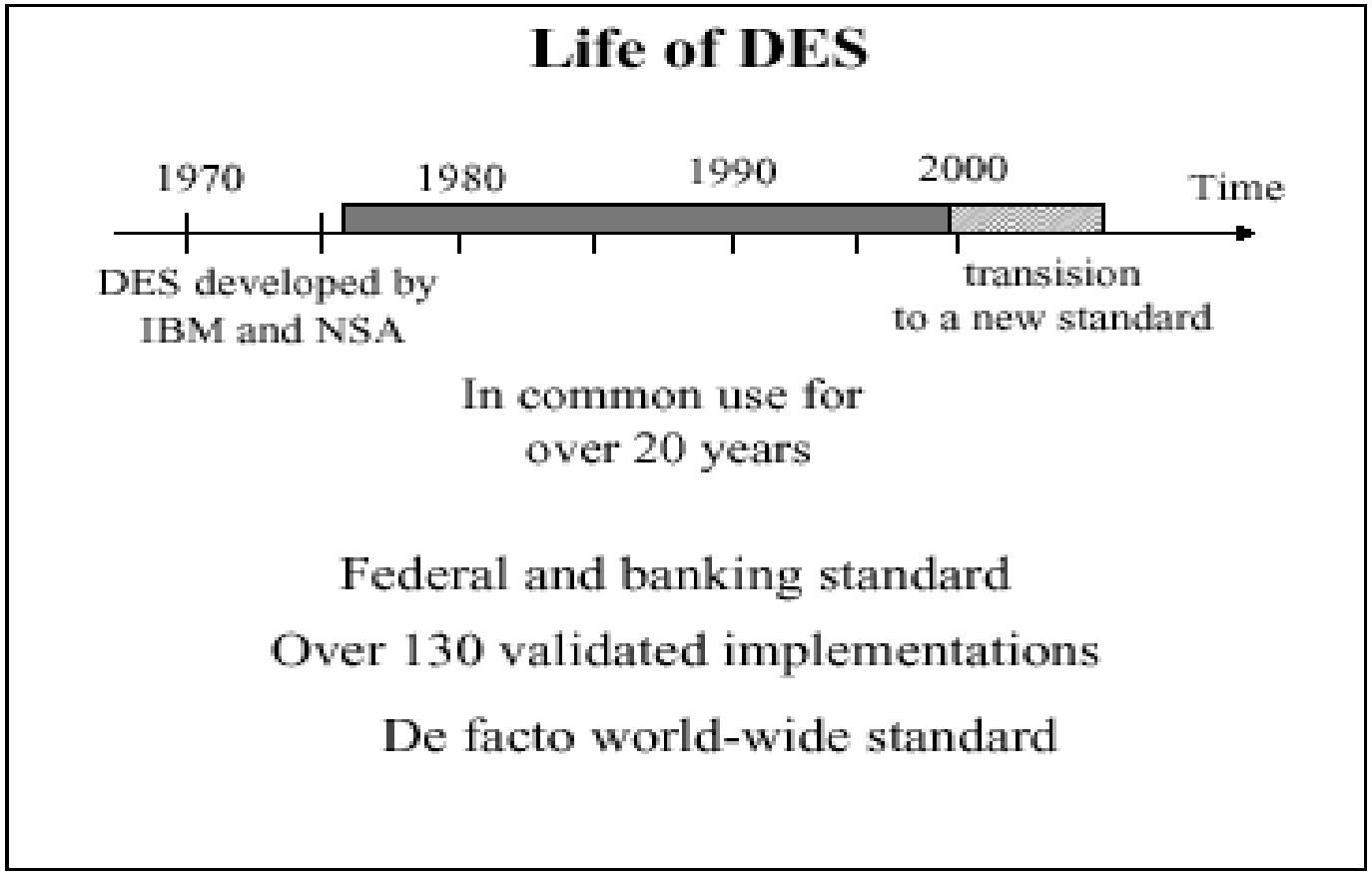


AES

- 分组长度只能是128,密钥长度均能分别指定为128位、192位、和256位。



Life of AES


Life of DES
1970
1980
1990
2000
Time
DES developed by
IBM and NSA
transision
to a new standard
In common use for
over 20 years
Federal and banking standard
Over 130 validated implementations
De facto world-wide standard

Rijndael算法&AES算法

- 在Rijndael算法中，分组长度和密钥长度均能分别指定为128位、192位、和256位，
- 在AES高级加密标准中，密钥的长度可以使用三者中的任意一种，但分组长度只能是128位，算法的许多参数与密钥长度相关：

Table 5.3 AES Parameters

Key size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

AES的基本运算

- 字节代替SubBytes
- 列混淆MixColumns
- 轮密钥加AddRoundKey
- 行移位ShiftRows

“三代替、一换位”

AES的加密和解密(以密钥为128为例)

1 算法的输入分组和输出，分组为128位， 轮数为10；

2 AES的结构不是个Feistel结构，而是每一轮并行地处理每一个分组；

3 密钥长度128位。扩展密钥长度为44字，每轮使用4字的轮密钥；

4 结构由4个不同的阶段组成；

字节代换:substitution byte

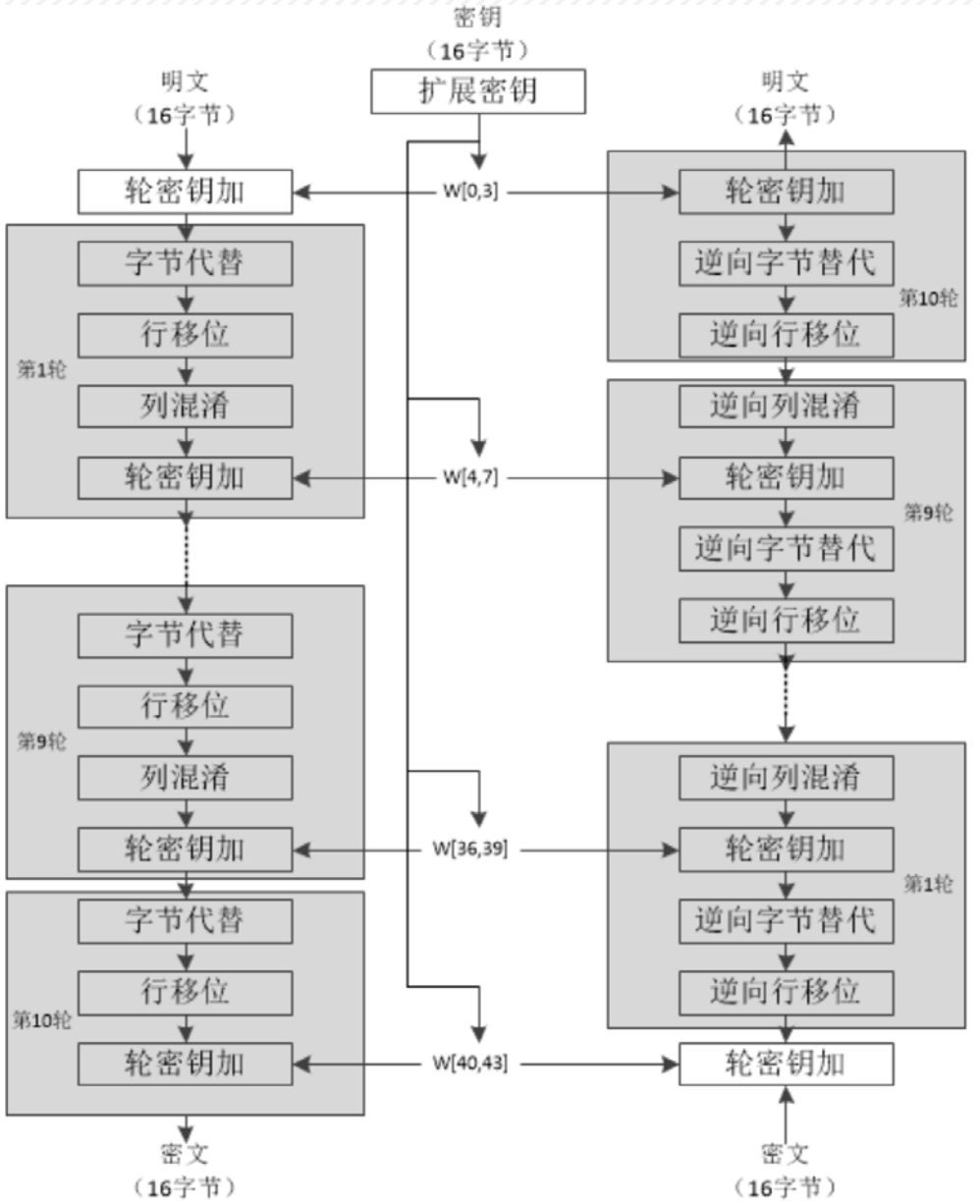
行移位；

列混合；

轮密钥加：当前分组和扩展密钥进行按位异或

5 加解密的最后一轮只包含三个阶段。

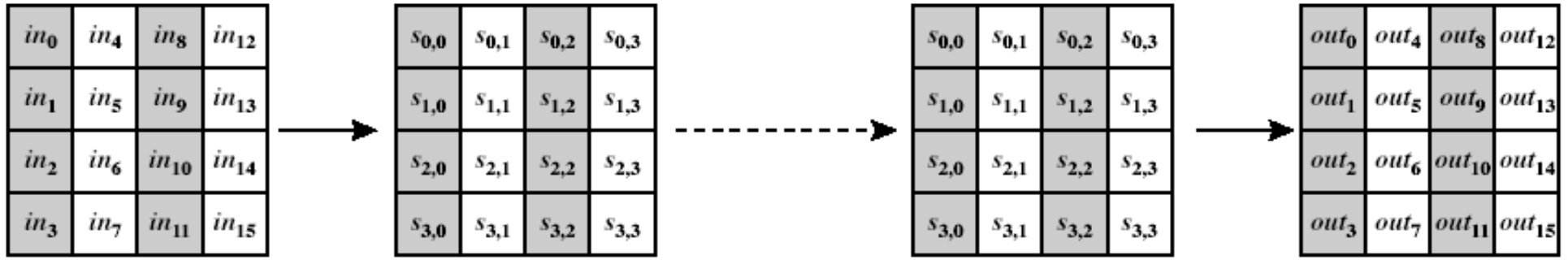
Key size (words/bytes/bits)	4/16/128
Plaintext block size (words/bytes/bits)	4/16/128
Number of rounds	10
Round key size (words/bytes/bits)	4/16/128
Expanded key size (words/bytes)	44/176



6.2 AES的结构

AES的数据结构1

- 输入分组是以字节为单位的正方形矩阵描述，该分组被复制到State数组中，这个数组在加密和解密的每个阶段都会改变。最后，State被复制到输出矩阵中作为输出的密文。



(a) Input, state array, and output

- 注意：矩阵中字节排列顺序是从上到下从左到右排列的，128位分组的前4个字节被按顺序放在in矩阵的第一列，接着的4个字节放在第2列。

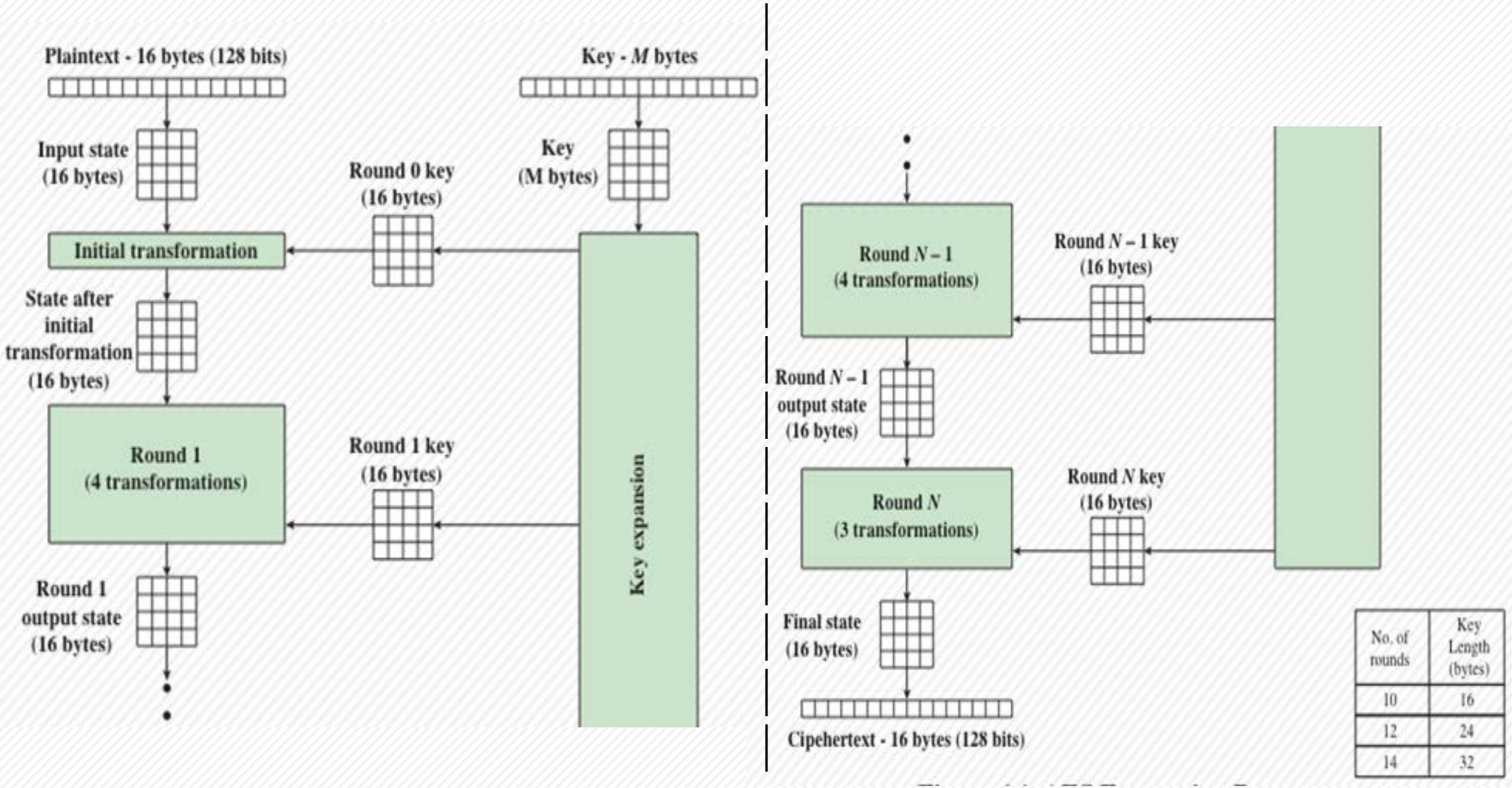
AES的数据结构2

- 128位的密钥也是以字节为单位的矩阵描述的，这个密钥采用扩展算法被扩展到一个以字为单位的密钥序列中，最终扩展成为44字的序列。

(b) Key and expanded key

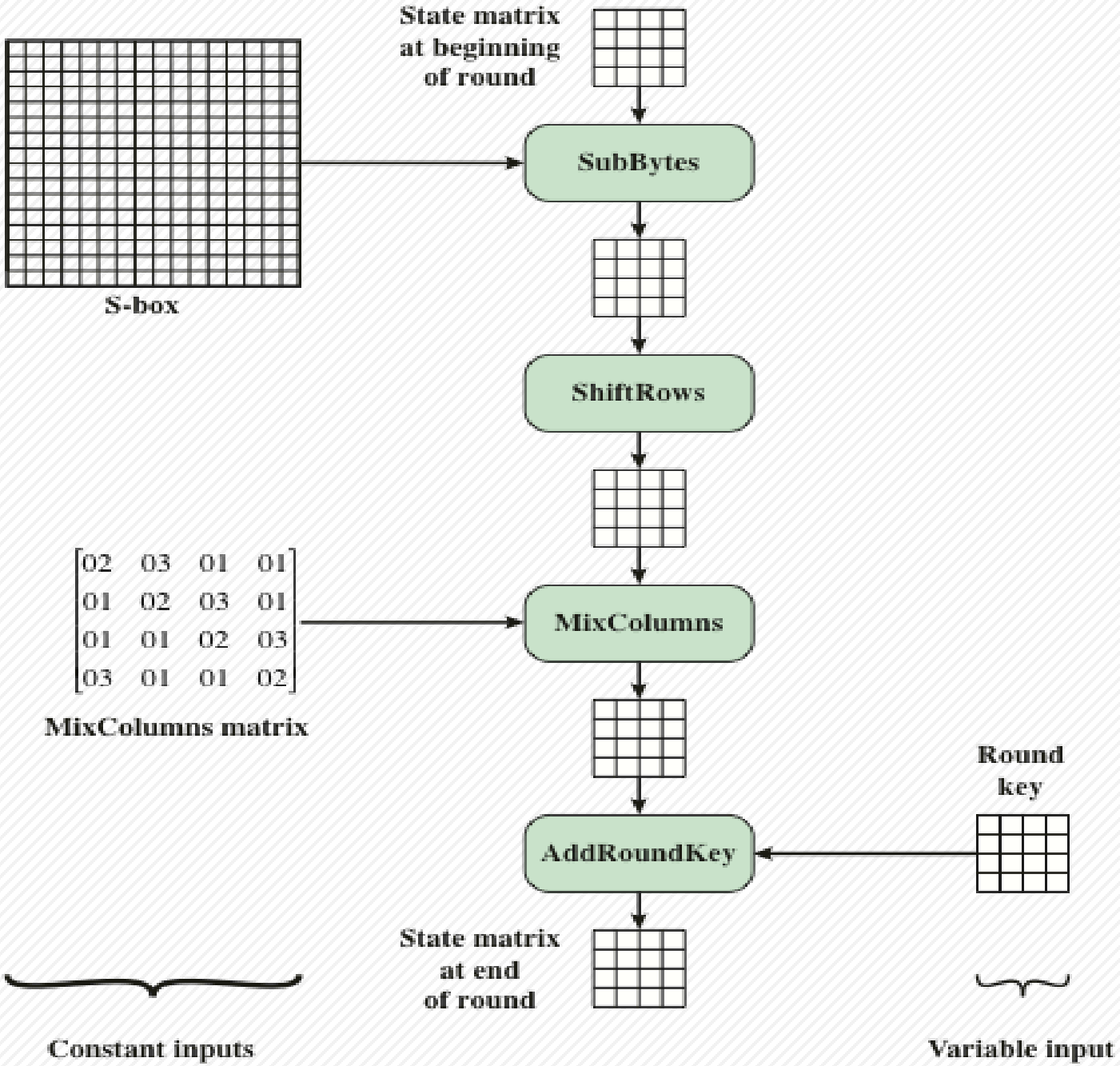
6.3 AES的变换函数

AES的加密过程



No. of rounds	Key Length (bytes)
10	16
12	24
14	32

AES的一轮加密过程


State matrix at beginning of round
S-box
SubBytes
ShiftRows
MixColumns
02 03 01 01
01 02 03 01
01 01 02 03
03 01 01 02
MixColumns matrix
Round key
AddRoundKey
State matrix at end of round
Constant inputs
Variable input

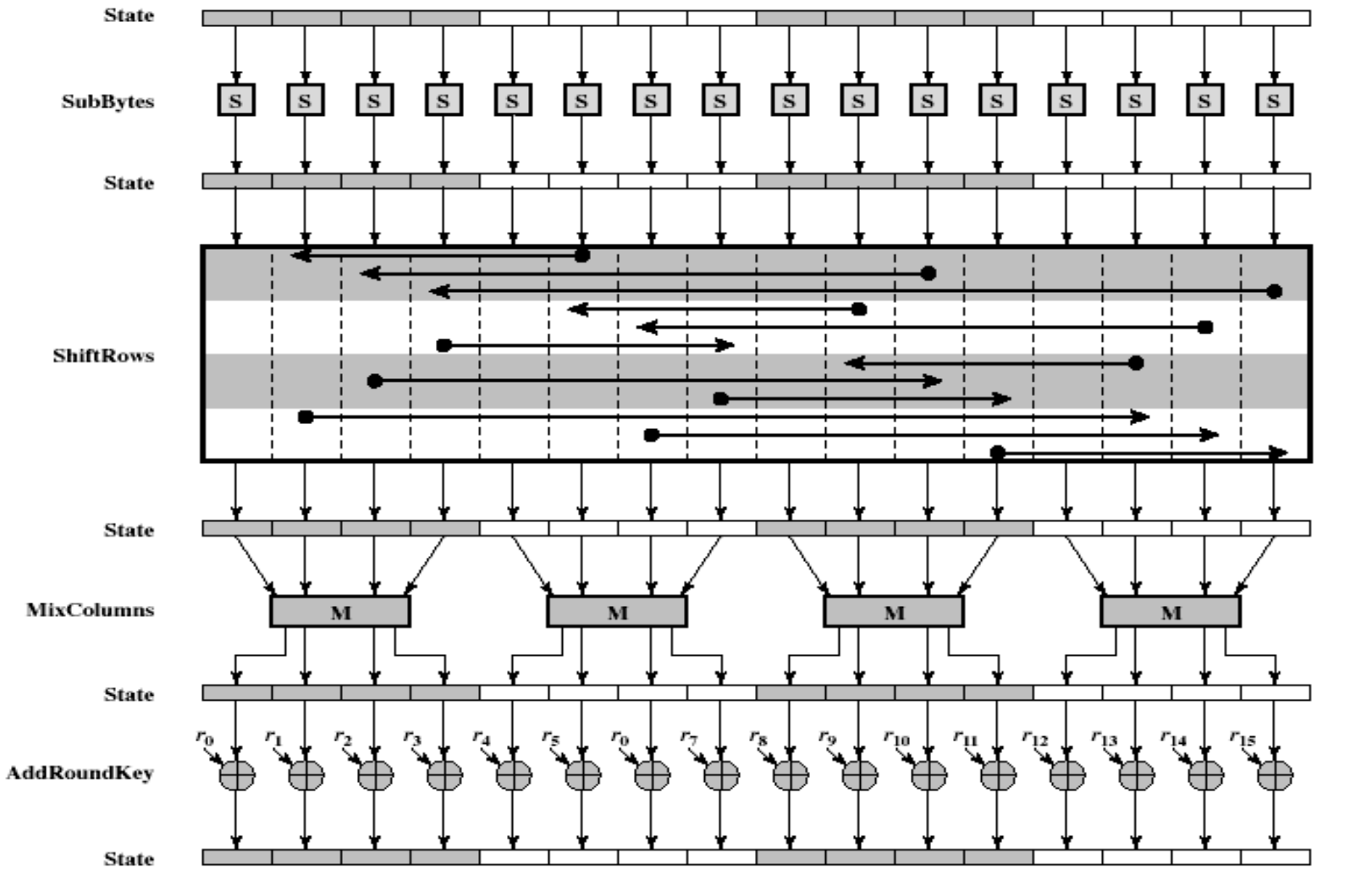
AES的一轮加密过程

字节代换

行移位

列混合

轮密钥加



一共进行10轮

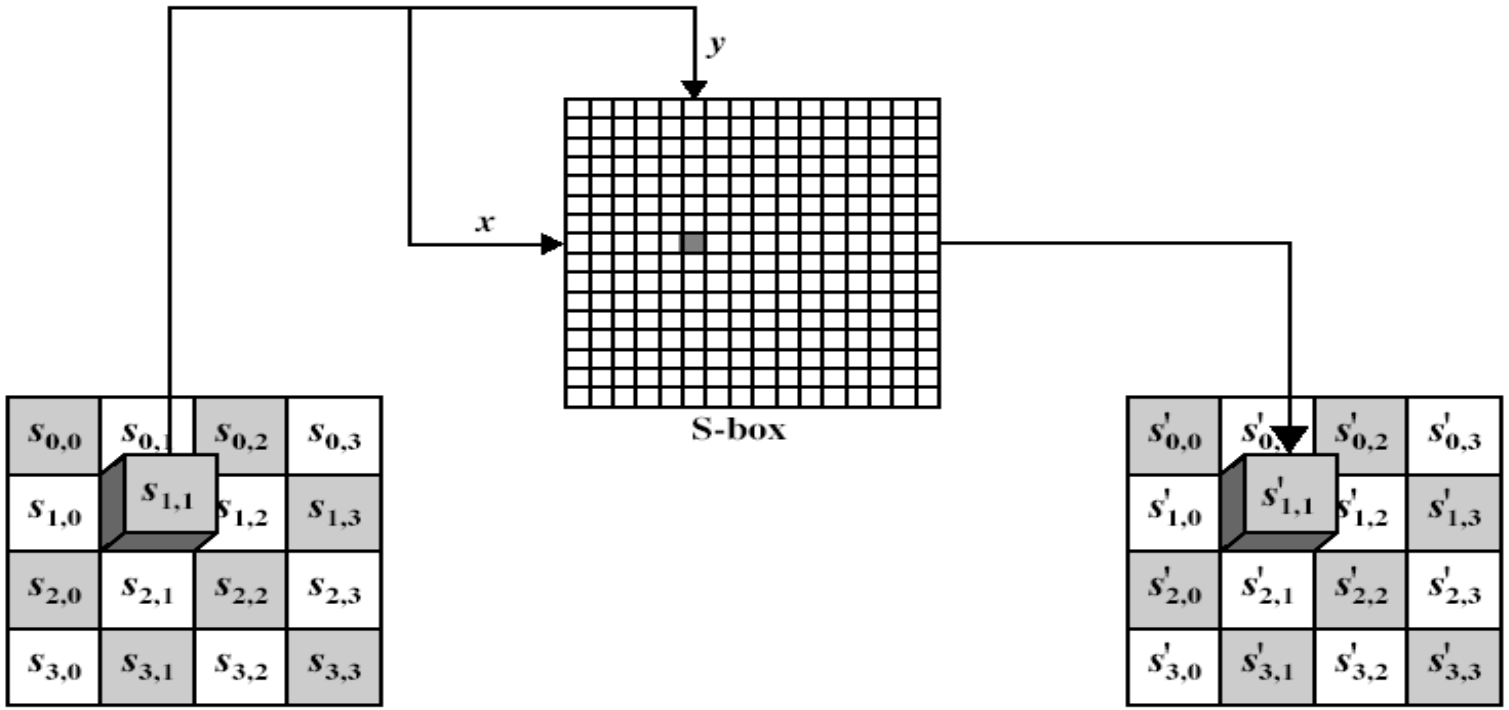
变换1：字节代换SubByte

- **字节代换可以通过查表操作获得，AES定义了一个S盒，它是一个16*16个字节组成的矩阵，包含了8位值所能表达的256种可能的变换。**

(a) S-box

		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

- State中的每一个字节按照如下方式映射为一个新的字节：该字节的高4位为S盒的行号，低4位作为列值，然后取出S盒中对应行列的元素作为输出。



(a) Substitute byte transformation

SubByte变换包含的运算

- **SubByte变换的实质——实际上是有限域GF（2^8）上的运算，由两个步骤组成：**

1、 求GF（2^8）上的乘法逆元：把S盒中的每一个字节映射为它在有限域GF（2^8）中的逆；{00}被映射为它自己。

定义：在GF（2^8）上的二进制多项式b(x)的乘法逆为满足下列方程式的二进制多项式a(x)，记为$b^{-1}(x)$

$$a(x)b(x) \bmod m(x)=1$$

2、仿射变换：把S盒中的每个字节记为$x_7x_6x_5x_4x_3x_2x_1x_0$，作如下变换：

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

- 以”95”作为输入，“95”在GF（2^8）中的乘法逆为“8a”，用二进制表示为10001010.
- 代入

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

- 结果为00101010,即2a,与前面查表结果相同

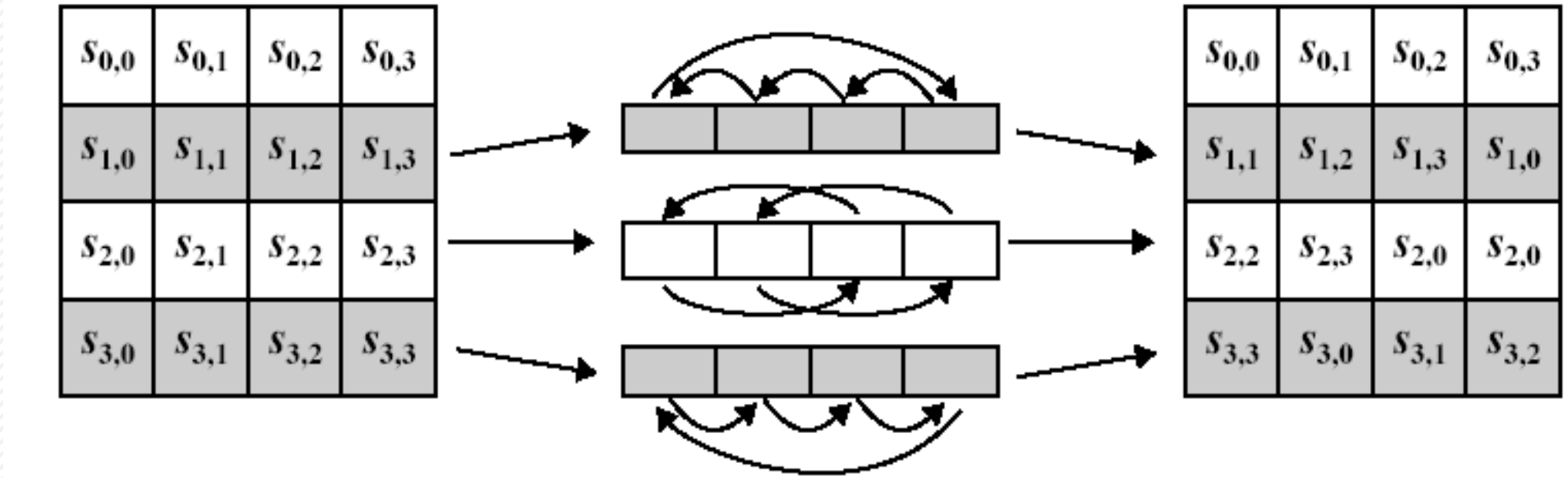
逆字节变换（逆S盒）

(b) Inverse S-box

		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

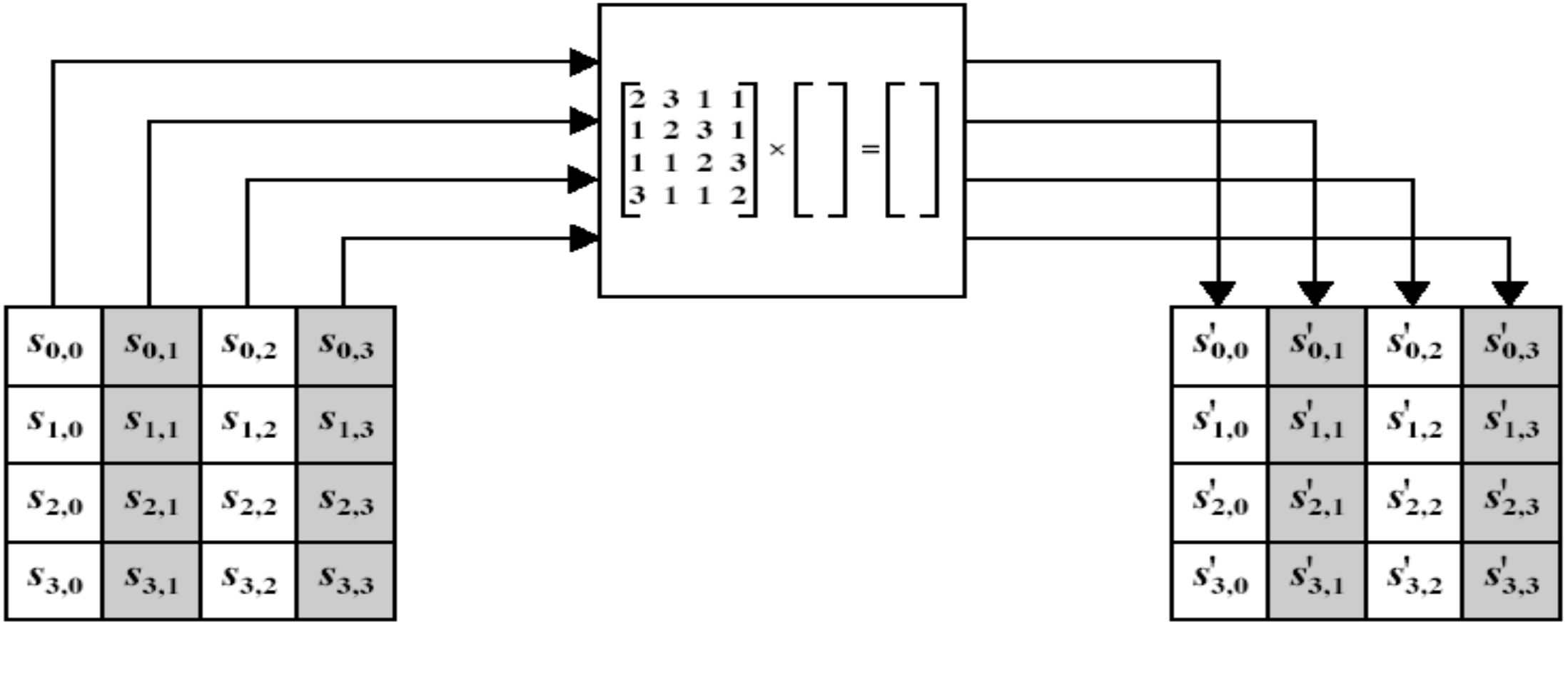
变换2：行移位变换

- State的第1行不变，第2行循环左移一个字节，第3行循环左移2个字节，第4行循环左移3个字节。



变换3：列混合变换

- **列混合变换对每一列独立进行，同样涉及GF（2^8）上的加，乘运算。**



$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} C9 \\ 6E \\ 46 \\ A6 \end{bmatrix} = \begin{bmatrix} DB \\ 37 \\ 94 \\ ED \end{bmatrix}$$

$$S'_{0,0} = (02 \bullet C9) \oplus (03 \bullet 6E) \oplus (01 \bullet 46) \oplus (01 \bullet A6)$$

其中：

$$02 \bullet C9 = 02 \bullet 11001001_B = 10010010_B \oplus 00011011_B = 10001001_B$$

$$03 \bullet 6E = (01 \oplus 02) \bullet 6E = 01101110_B \oplus 11011100_B = 10110010_B$$

$$01 \bullet 46 = 01000110_B$$

$$01 \bullet A6 = 10100110_B$$

则：

$$S'_{0,0} = 10001001_B \oplus 10110010_B \oplus 01000110_B \oplus 10100110_B$$

$$= 11011011_B = DB$$

变换4：轮密钥加

- **128位的State按位与128位的轮密钥异或。**

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

$\approx$

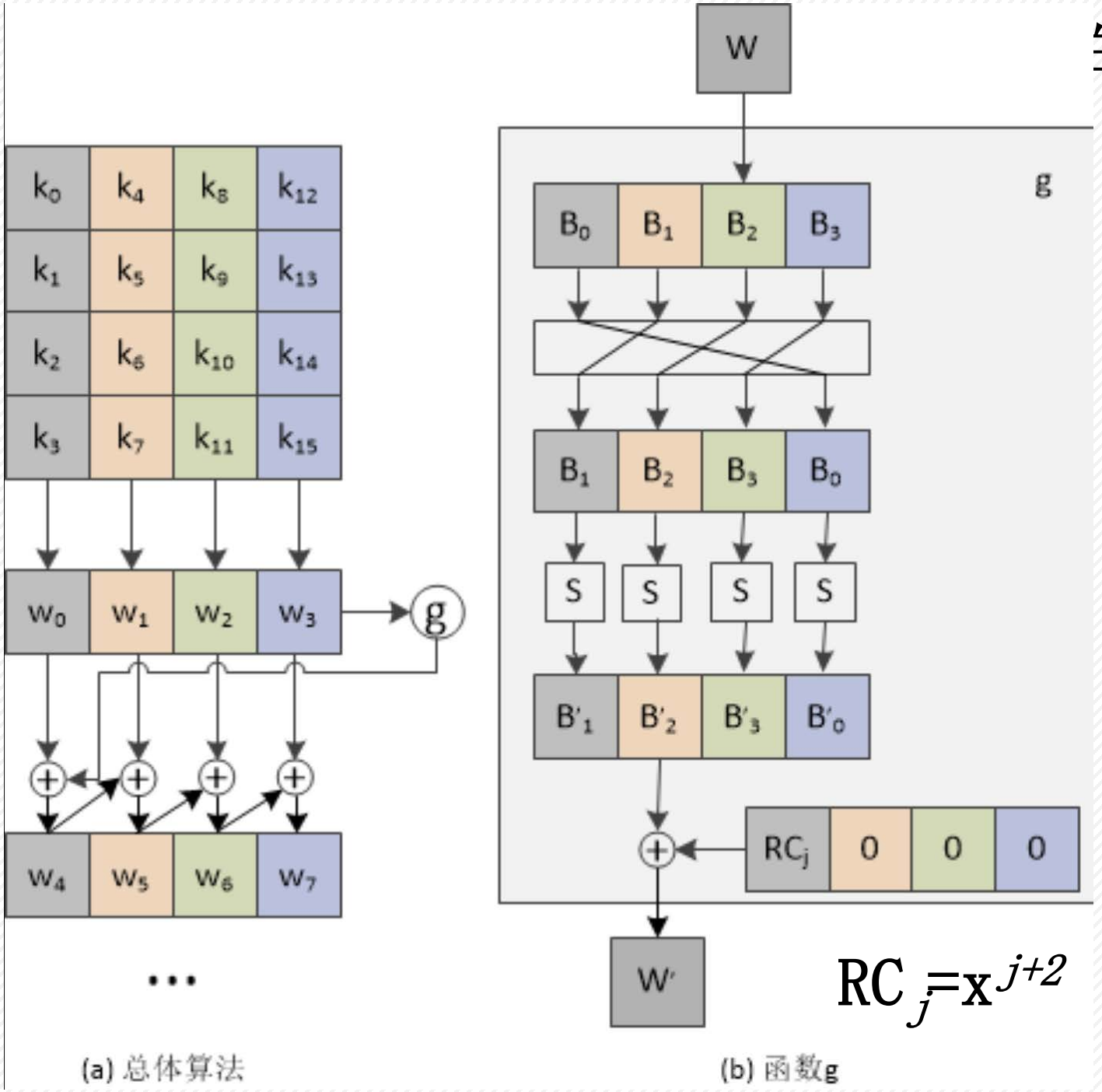
w_i	w_{i+1}	w_{i+2}	w_{i+3}

$=$

$s'_{0,0}$	$s'_{0,1}$	$s'_{0,2}$	$s'_{0,3}$
$s'_{1,0}$	$s'_{1,1}$	$s'_{1,2}$	$s'_{1,3}$
$s'_{2,0}$	$s'_{2,1}$	$s'_{2,2}$	$s'_{2,3}$
$s'_{3,0}$	$s'_{3,1}$	$s'_{3,2}$	$s'_{3,3}$

6.4 AES的密钥扩展

的密钥扩展算法



输入密钥直接被复制到扩展密钥数组的前4个字，然后每次用4个字填充扩展密钥数组余下的部分。

在扩展密钥数组中，w[i]的值依赖于w[i-1]和w[i-4](即w[4]依赖于w[3]和w[0])。

对W数组中下标为4的倍数的元素，采用一个更复杂的g函数来计算。

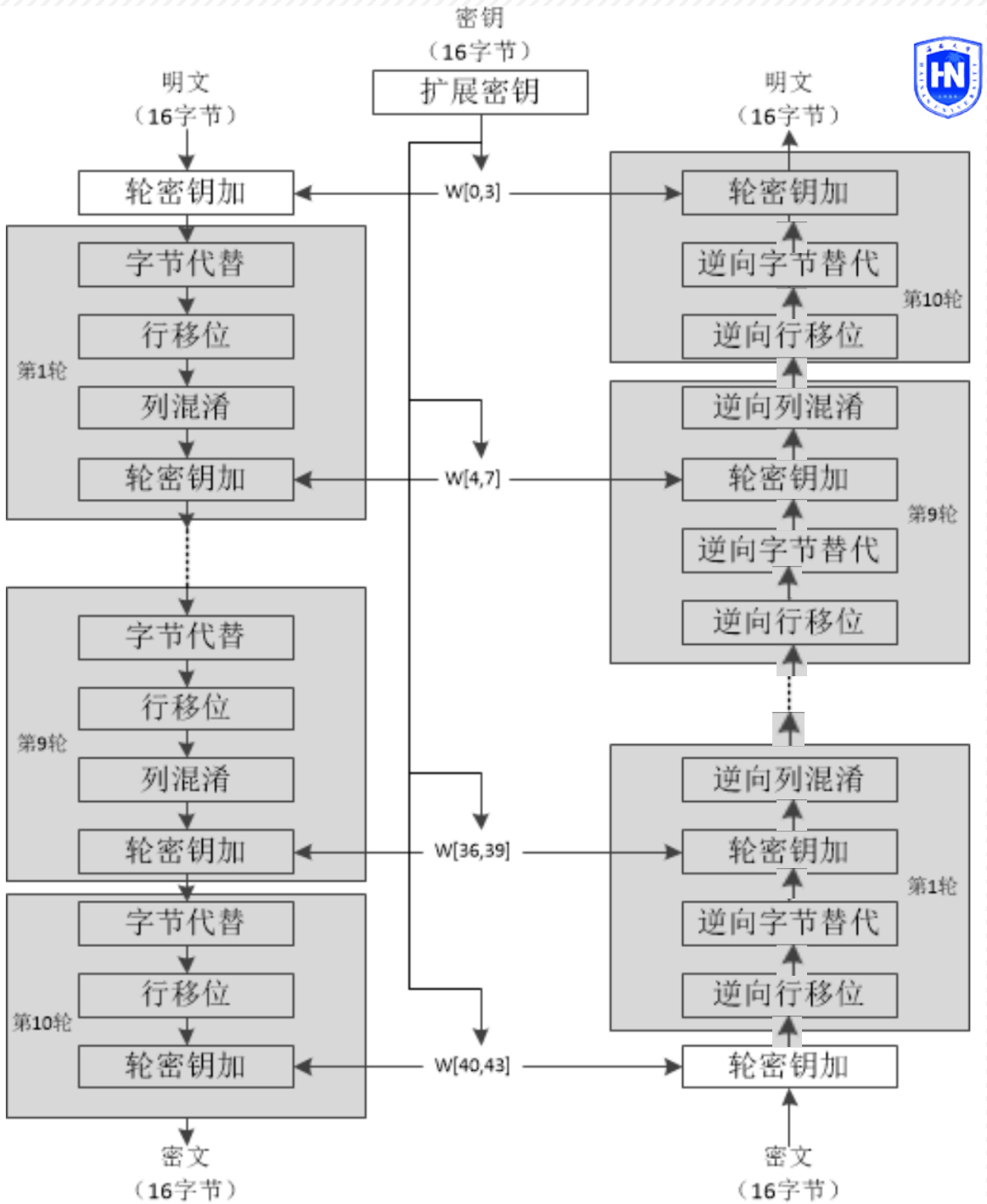
6.5 AES的解密

AES的解密

标准解密流程与标准加密流程并不完全一致

加密每一轮的流程是：字节代替->行移位->列混淆->轮密加。

解密每一轮的流程是：逆向行移位->逆向字节代替->轮密加->逆向列混淆。



AES的解密

可以对解密构成进行改进，使得解密流程与加密流程等效。

1、**交换逆向行移位和逆向字节代替**：

由于逆向行移位并不影响state数组中字节的内容，而逆向字节代替也不会影响state数组中字节的位置，因而两者可以交换顺序而不影响解密。

2、**交换轮密钥加和逆向列混淆**：

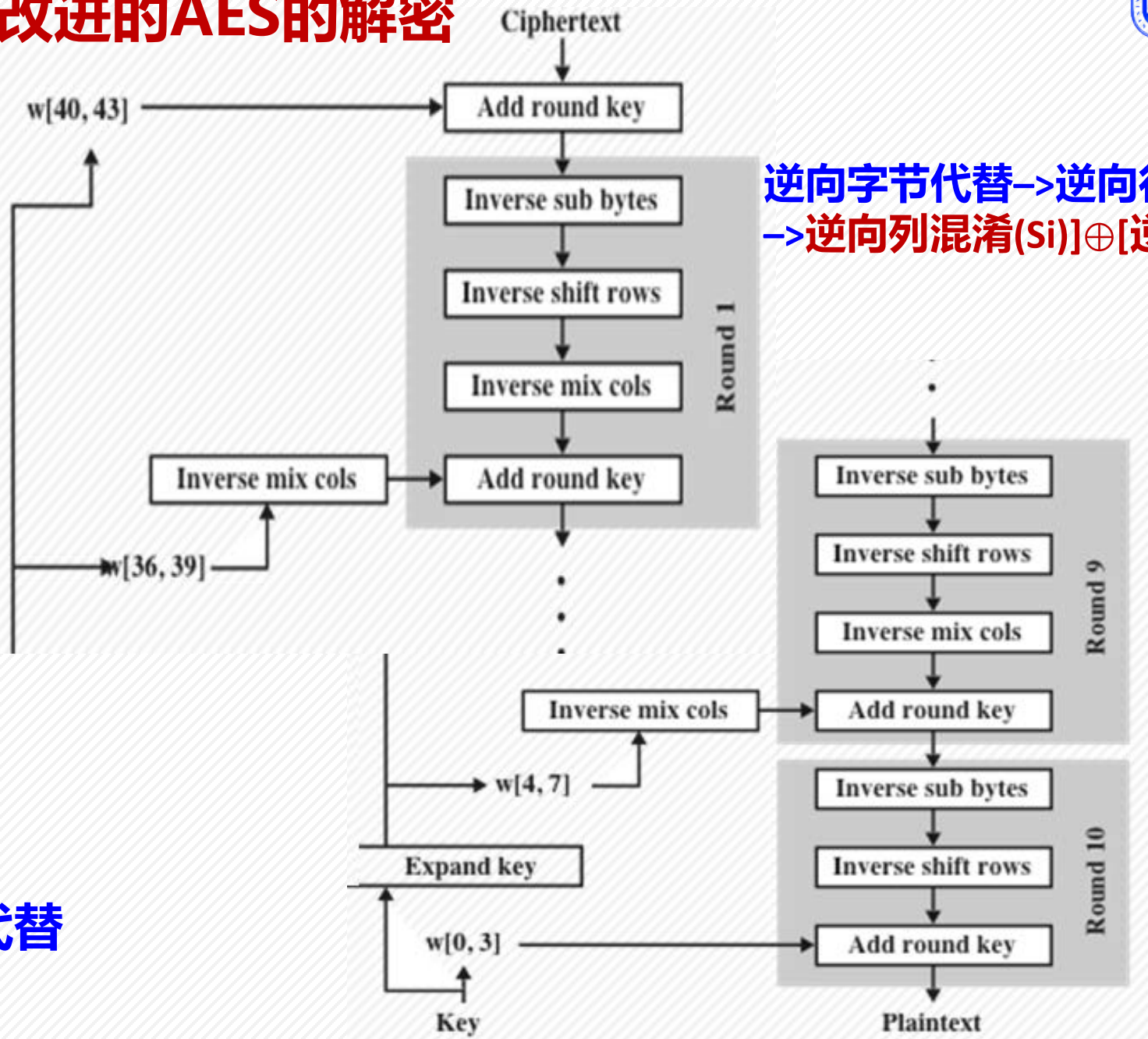
这两种操作均不会改变state中字节的顺序，给定状态Si和给定轮密钥wi，可证明

逆向列混淆(Si⊕wi)= [逆向列混淆(Si)]⊕[逆向列混淆(wi)]

逆向行移位->逆向字节代替->轮密加->逆向列混淆

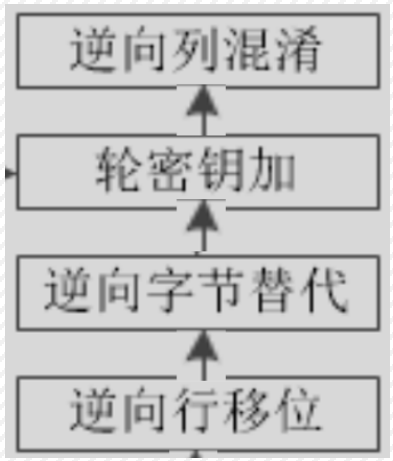
逆向字节代替->逆向行移位->逆向列混淆(Si)]⊕[逆向列混淆(wi)]

改进的AES的解密



逆向字节代替–>逆向行移位
–>逆向列混淆(Si)]⊕[逆向列混淆(wi)]

标准AES的解密



逆向行移位–>逆向字节代替
–>轮密加–>逆向列混淆

逆S盒

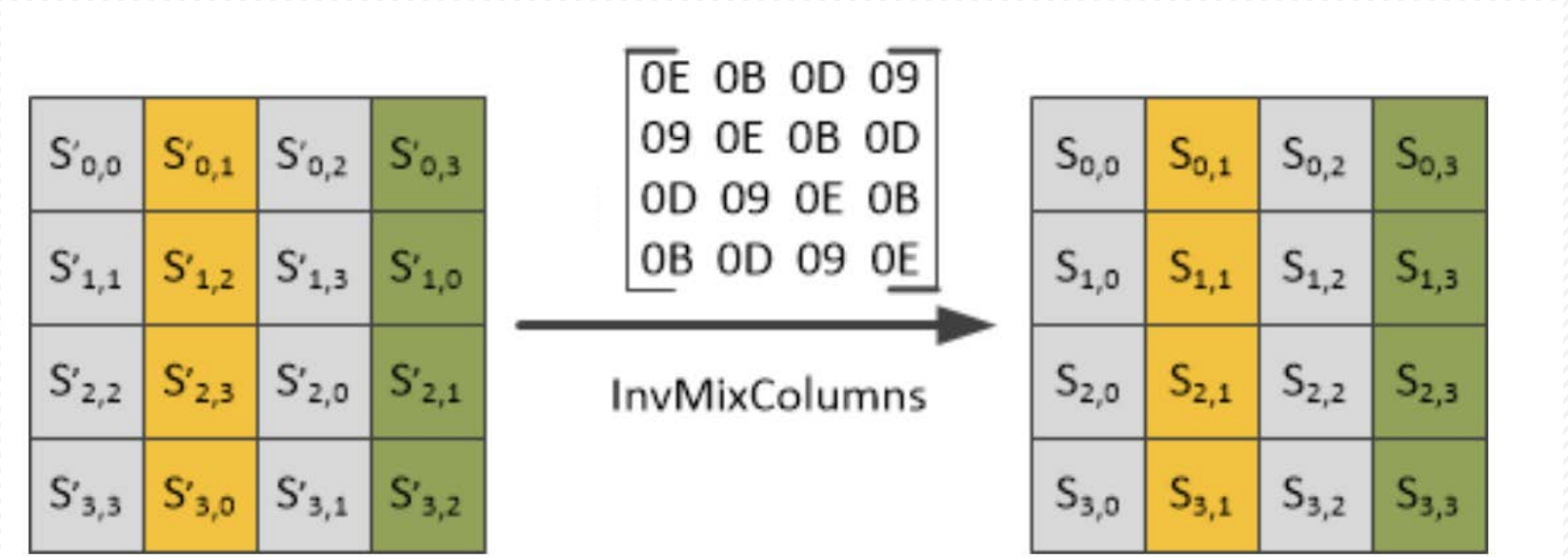
$$\begin{pmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

逆S盒：

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

(b) Inverse S-box

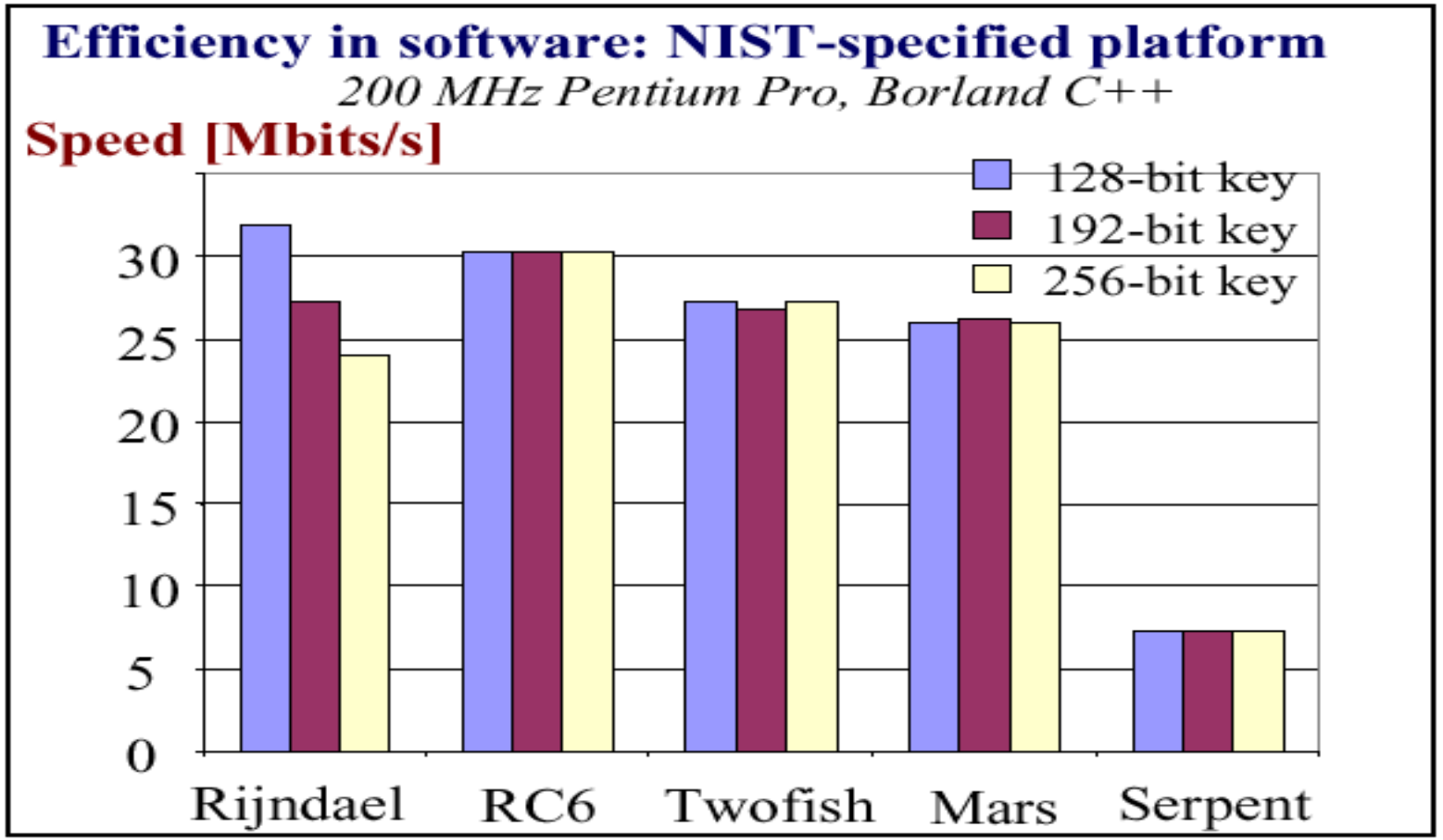
逆向列混淆的原理图如下：



$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} = \begin{bmatrix} 01 & 00 & 00 & 00 \\ 00 & 01 & 00 & 00 \\ 00 & 00 & 01 & 00 \\ 00 & 00 & 00 & 01 \end{bmatrix}$$

6.6 AES性能

- AES算法很快而且所需的内存不多，这个算法非常可靠
- AES汇聚了安全性能、效率、可实现性和灵活性等优点




海南大学
Hainan university