4 triple-DES and AES

ch4~6 & Sec7.1 in textbook

Yanwei Yu

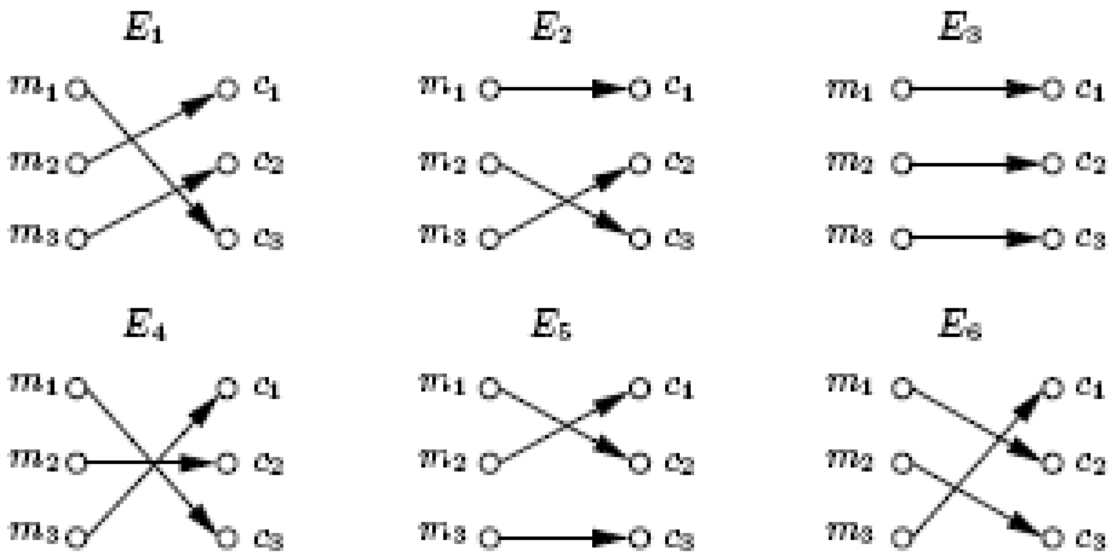
E-mail: ywyu@ustc.edu.cn

Outline

- 2DES, 3DES
- AES History
- Mathematical Basis —— GF(2^n)
- AES Principle
- AES Security Analysis
- Implementation of AES

Example of Symmetric Block Cipher

- **Assume M={m1,m2,m3}, C={c1,c2,c3}, K={1,2,3,4,5,6} and C=E_k(M), M=D_k(C)**



Number of substitution table: 3!= 6

number of key: 6

For one plaintext-ciphertext pair (mi, cj), two correct keys exist.

So, “real” correct key is determined for two pairs of plaintext-ciphertext.

- DES: $C=E_k(M)$
  - Plaintext block M: 64 bits
  - Key k: 56 bits
  - Cipher-text block C: 64 bits
- Theoretically, $(2^{64})!$ mappings exist from plaintext block to cipher-text block.
- In fact, only (2^{56}) mappings are used because key size $=2^{56}$.
- Q: For one plaintext-ciphertext pair (mi, cj), How many correct keys exist?

Outline

- 2DES, 3DES
- AES History
- Mathematical Basis —— $GF(2^n)$
- AES Principle
- AES Security Analysis
- Implementation of AES

Review of DES

- 1973: NBS issued a request for proposals for a national cipher standard
- 1977: adopted IBM's submission as the Data Encryption Standard
- 1994: use DES for another five years; NIST recommended the use of DES for applications other than the protection of classified information.
- 1999: use DES for legacy systems and use triple DES (which in essence involves repeating the DES algorithm three times on the plaintext using two or three different keys to produce the ciphertext) instead.

Multiple Encryption & DES

- **A replacement for DES was needed**
  - **potential vulnerability of DES to a brute-force attack: $O(2^{55})$**
  - **$O(2^{54})$ under the chosen plaintexts**

    (if $C = E_K(P)$, then $\overline{C} = E_{\overline{K}}(\overline{P})$)
- **Two alternatives**
  - **design a completely new algorithm: AES**
  - **use multiple encryption with DES and multiple keys to preserve the existing investment in software and equipment: Double-DES, Triple-DES**

Double-DES?

- could use two encryption stages and two keys (2DES) on each block
  - Encrypt: $C = E_{K2}(E_{K1}(P))$
  - Decrypt: $P = D_{K1}(D_{K2}(C))$
- issue of reduction to single stage
  - $E_{K2}(E_{K1}(P)) = E_{K3}(P)$?? not likely but only finally proved in 1992
- "meet-in-the-middle" attack: $O(2^{56})$
  - Assume pair (P,C), have $C = E_{K2}E_{K1}(P)$
    - $X = E_{K1}(P) = D_{K2}(C)$
    - attack by encrypting P with all keys and store
    - then decrypt C with keys and match X value
  - two blocks of known plaintext-ciphertext will succeed against double DES

Triple-DES

- **use three stages of encryption**
- **cost of the known-plaintext attack:** $O(2^{112})$
- **Two forms:**
  - **use two keys with E-D-E sequence**
  - **use Three Keys with E-D-E sequence**

Triple-DES with Two-Keys

- use two keys with E-D-E sequence
  - $C = E_{K1}(D_{K2}(E_{K1}(P)))$
  - encrypt & decrypt equivalent in security
  - if `K1==K2` then can work with single DES
- standardized in ANSI X9.17 & ISO8732
- no current known practical attacks
- Key length: 112 bits

Triple-DES with Three-Keys

- using two-key 3DES may feel some concern although are no practical attacks on two-key Triple-DES
- three-key 3DES is the preferred alternative: key length is 168bits
  - $C = E_{K3}(D_{K2}(E_{K1}(P)))$
  - if `K1==K2 or K2 ==K3` then can work with single DES
- has been adopted by some Internet applications, including PGP, S/MIME.

Triple-DES summary

- **Attractions:**
  - its key of 168-bit length overcomes the vulnerability to brute-force attack of DES.
  - Security of algorithm. the underlying encryption algorithm in 3DES is the same as in DES.
    - DES is very resistant to cryptanalysis.
    - since 1977, no effective cryptanalytic attack based on DES rather than brute force has been found after until 1999.
- **3DES would be an appropriate choice for a standardized encryption algorithm for decades to come.**

- Drawbacks
  - relatively less effective in software. The original DES was designed for mid-1970s hardware implementation and does not produce efficient software code. 3DES, which has three times as many rounds as DES, is correspondingly slower.
  - Short block size. both DES and 3DES use a 64-bit block size. For reasons of both efficiency and security, a larger block size is desirable.
- 3DES is not a reasonable candidate for long-term use

Outline

- 2DES, 3DES
- AES History
- Mathematical Basis —— GF(2^n)
- AES Principle
- AES Security Analysis
- Implementation of AES

Origins

- clear a replacement for DES was needed
  - potential vulnerability of DES to a brute-force attack: $O(2^{55})$
- can use 3DES(Triple-DES), but 3DES is slow and has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99
- Rijndael was selected as the AES in Oct-2000
- issued as FIPS PUB 197 standard in Nov-2001

AES Requirements

- private key symmetric block cipher
- 128-bit block, 128/192/256-bit keys
- stronger & faster than Triple-DES
- active life of 20-30 years (+ archival use)
- provide full specification & design details
- both C & Java implementations
- NIST have released all submissions & unclassified analyses

AES Evaluation Criteria

- initial criteria:
  - security – effort for practical cryptanalysis
  - cost – in terms of computational efficiency
  - algorithm & implementation characteristics
- final criteria:
  - general security
  - ease of software & hardware implementation
  - Attacks to implementation
    - E.g, Timing attacks.
  - flexibility (in en/decrypt, keying, other factors)

AES Shortlist(候选者)

- after testing and evaluation, shortlist in Aug-99:
  - MARS (IBM) - complex, fast, high security
  - RC6 (USA) - simple, fast, low security
  - Rijndael (Belgium) - clean, fast, good security
  - Serpent (Euro) - clean, slow, high security
  - Twofish (USA) - complex, fast, high security

Outline

- 2DES, 3DES
- AES History
- Mathematical Basis —— GF(2^n)
  - Chapter 5
- AES Principle
- AES Security Analysis
- Implementation of AES

Learning Objectives

- Distinguish among groups, rings, and fields.
- Define finite fields of the form GF(p).
- Explain the differences among three classes of polynomial(多项式) arithmetic as bellows:
  - ✓ ordinary polynomial arithmetic
  - ✓ polynomial arithmetic with coefficients in Zp
  - ✓ modular polynomial arithmetic in GF(2^n).
- Define finite fields of the form GF(2^n).
- Explain the two different uses of the mod operator.

Groups, Rings, and Fields

- A number of cryptographic algorithms rely heavily on properties of finite fields
  - the Advanced Encryption Standard (AES) :GF(2^n)
  - elliptic curve cryptography: GF(P) and GF(2^n)
- Finite field: a field with a finite number of elements
- concern operations on "numbers"
  - combine two elements of the set, perhaps in some operation , to obtain a third element of the set
- Operation:
  - Basis operation: +, ·
  - Derived operation: -, ÷

Groups, Rings, and Fields

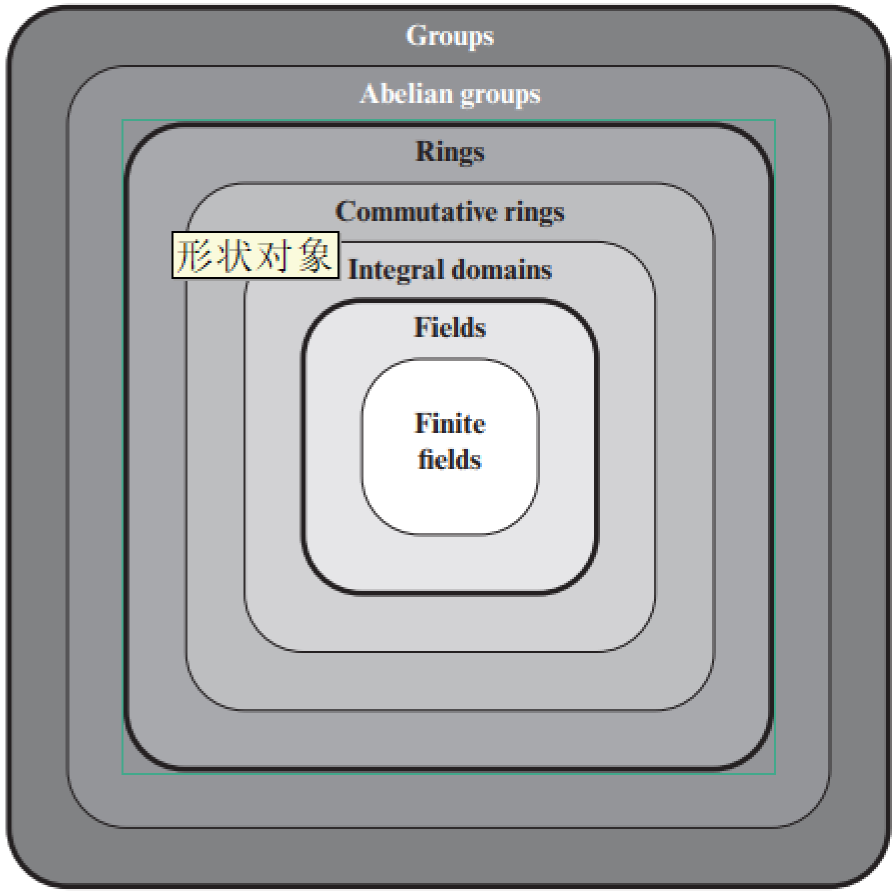
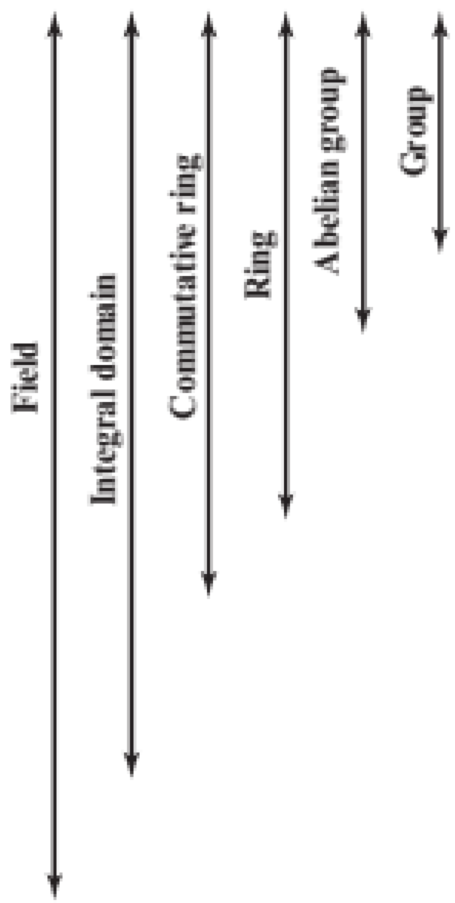


Figure 5.1 Groups, Rings, and Fields

Groups, Rings, and Fields



(A1) Closure under addition:	If a and b belong to S, then $a + b$ is also in S
(A2) Associativity of addition:	$a + (b + c) = (a + b) + c$ for all a, b, c in S
(A3) Additive identity:	There is an element 0 in R such that $a + 0 = 0 + a = a$ for all a in S
(A4) Additive inverse:	For each a in S there is an element $-a$ in S such that $a + (-a) = (-a) + a = 0$
(A5) Commutativity of addition:	$a + b = b + a$ for all a, b in S
(M1) Closure under multiplication:	If a and b belong to S, then ab is also in S
(M2) Associativity of multiplication:	$a(bc) = (ab)c$ for all a, b, c in S
(M3) Distributive laws:	$a(b + c) = ab + ac$ for all a, b, c in S; $(a + b)c = ac + bc$ for all a, b, c in S
(M4) Commutativity of multiplication:	$ab = ba$ for all a, b in S
(M5) Multiplicative identity:	There is an element 1 in S such that $a1 = 1a = a$ for all a in S
(M6) No zero divisors:	If a, b in S and $ab = 0$, then either $a = 0$ or $b = 0$
(M7) Multiplicative inverse:	If a belongs to S and $a \neq 0$, there is an element a^{-1} in S such that $aa^{-1} = a^{-1}a = 1$

Figure 5.2 Properties of Groups, Rings, and Fields

Fields

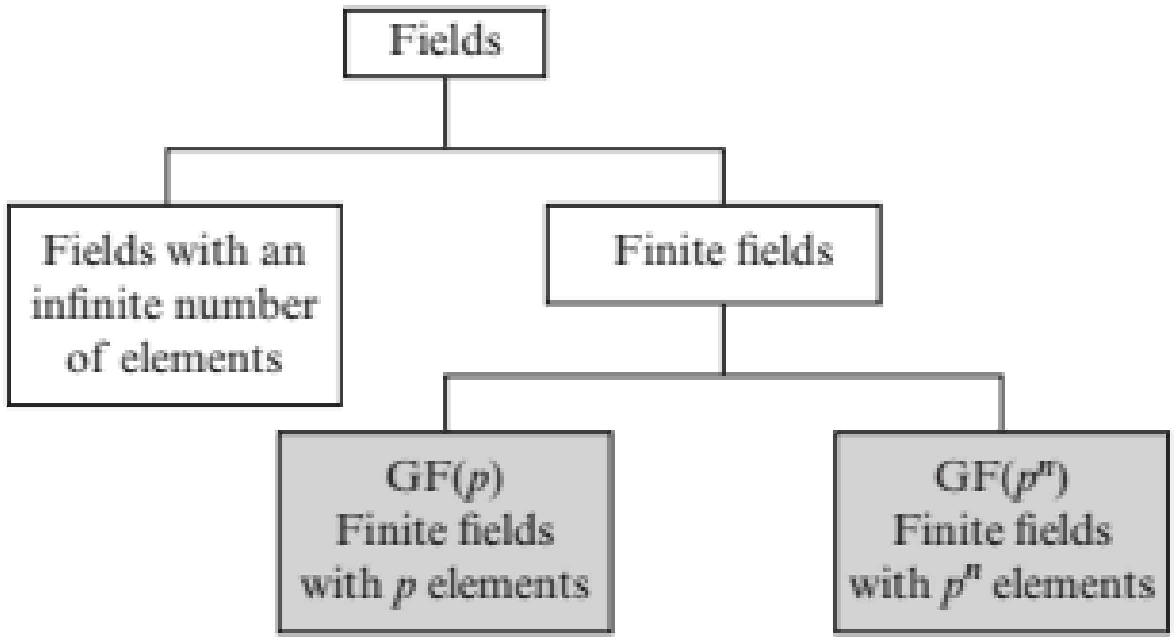


Figure 5.3 Types of Fields

Galois Fields

- finite fields play a key role in cryptography
- in particular often use the fields:
  - GF(p):
    - p is a prime
    - number of elements in a finite field is P
  - GF(2^n)
    - number of elements in a finite field must be a power of a prime, p^n
    - known as Galois fields, denoted GF(p^n)

Finite Fields of the form GF(p)

- **GF(2)**

The simplest finite field is GF(2). Its arithmetic operations are easily summarized:

+	0	1
0	0	1
1	1	0

Addition

×	0	1
0	0	0
1	0	1

Multiplication

w	$-w$	w^{-1}
0	0	–
1	1	1

Inverses

In this case, addition is equivalent to the exclusive-OR (XOR) operation, and multiplication is equivalent to the logical AND operation.

Finite Fields of the form GF(p)

- **GF(7):** using modular arithmetic modulo 7

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

(d) Addition modulo 7

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

(e) Multiplication modulo 7

w	0	1	2	3	4	5	6
$-w$	0	6	5	4	3	2	1
w^{-1}	—	1	4	5	2	3	6

(f) Additive and multiplicative inverses modulo 7

Finite Fields of the form GF(p)

- **set Z_8:** using modular arithmetic modulo 8

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

(a) Addition modulo 8

×	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

(b) Multiplication modulo 8

w	0	1	2	3	4	5	6	7
$-w$	0	7	6	5	4	3	2	1
w^{-1}	—	1	—	3	—	5	—	7

(c) Additive and multiplicative inverses modulo 8

set Z_8 is not a field!

Finite Fields of the form GF(2^3) - Motivation

- **wish to work with integers in the range 0 through 2^n-1, which fit into an n-bit word**
- **But, the set of integers modulo 2^n for n>1, is not a field**
  - **2^n is not a prime**

Example GF(2^3)

Table 4.6 Polynomial Arithmetic Modulo ($x^3 + x + 1$)

	+	000 0	001 1	010 x	011 $x+1$	100 x^2	101 x^2+1	110 x^2+x	111 x^2+x+1
000	0	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
001	1	1	0	$x+1$	x	x^2+1	x^2	x^2+x+1	x^2+x
010	x	x	$x+1$	0	1	x^2+x	x^2+x+1	x^2	x^2+1
011	$x+1$	$x+1$	x	1	0	x^2+x+1	x^2+x	x^2+1	x^2
100	x^2	x^2	x^2+1	x^2+x	x^2+x+1	0	1	x	$x+1$
101	x^2+1	x^2+1	x^2	x^2+x+1	x^2+x	1	0	$x+1$	x
110	x^2+x	x^2+x	x^2+x+1	x^2	x^2+1	x	$x+1$	0	1
111	x^2+x+1	x^2+x+1	x^2+x	x^2+1	x^2	$x+1$	x	1	0

(a) Addition

	×	000 0	001 1	010 x	011 $x+1$	100 x^2	101 x^2+1	110 x^2+x	111 x^2+x+1
000	0	0	0	0	0	0	0	0	0
001	1	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
010	x	0	x	x^2	x^2+x	$x+1$	1	x^2+x+1	x^2+1
011	$x+1$	0	$x+1$	x^2+x	x^2+1	x^2+x+1	x^2	1	x
100	x^2	0	x^2	$x+1$	x^2+x+1	x^2+x	x	x^2+1	1
101	x^2+1	0	x^2+1	1	x^2	x	x^2+x+1	$x+1$	x^2+x
110	x^2+x	0	x^2+x	x^2+x+1	1	x^2+1	$x+1$	x	x^2
111	x^2+x+1	0	x^2+x+1	x^2+1	x	1	x^2+x	x^2	$x+1$

(b) Multiplication

choose an irreducible polynomial of degree 3 : ($x^3 + x + 1$)

Table 5.1. Arithmetic Modulo 8

+	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

(a) Addition modulo 8

×	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

(b) Multiplication modulo 8

w	$-w$	w^{-1}
0	0	—
1	7	1
2	6	—
3	5	3
4	4	—
5	3	5
6	2	—
7	1	7

(c) Additive and multiplicative inverses modulo 8

GF(2^n)

- A set of integers in the range 0 through 2^n-1, which fit into an n-bit word
- Operations: +, · ; -, ÷
- attractions for cryptographic algorithms
  - provides a uniform mapping (mapping the integers evenly onto themselves)

Example of Operations in $GF(2^n)$

- Advanced Encryption Standard (AES) uses arithmetic in the finite field $GF(2^8)$, with the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$.
- Consider $f(x) = x^6 + x^4 + x^2 + x + 1$ and $g(x) = x^7 + x + 1$. Then
  - Addition:

$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1)$	$= x^7 + x^6 + x^6 + x^4 + x^2$	(polynomial notation)
$(01010111) \oplus (10000011)$	$= (11010100)$	(binary notation)

  - Multiplication:
    - $f(x) \cdot g(x) = x^{13} + x^{11} + x^9 + x^8 + x^7 + x^7 + x^5 + x^3 + x^2 + x + x^6 + x^4 + x^2 + x + 1$
      $= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$
    - $f(x) \cdot g(x) \bmod m(x) = x^7 + x^6 + 1$

$$
\begin{array}{r l}
 & x^5 + x^3 \\
x^8 + x^4 + x^3 + x + 1 \,\big)\!\!\!\! & \overline{x^{13} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + \quad + 1} \\
 & \underline{x^{13} \qquad\quad + x^9 + x^8 \qquad\quad + x^6 + x^5} \\
 & \qquad x^{11} \qquad\qquad\qquad\qquad\qquad\quad + x^4 + x^3 \\
 & \qquad \underline{x^{11} \qquad\qquad + x^7 + x^6 \qquad + x^4 + x^3} \\
 & \qquad\qquad\qquad\quad x^7 + x^6 \qquad\qquad\qquad\quad + 1
\end{array}
$$

In an earlier example, we showed that for $f(x) = x^6 + x^4 + x^2 + x + 1$, $g(x) = x^7 + x + 1$, and $m(x) = x^8 + x^4 + x^3 + x + 1$, $f(x) \times g(x) \bmod m(x) = x^7 + x^6 + 1$. Redoing this in binary arithmetic, we need to compute (01010111) x (10000011). First, we determine the results of multiplication by powers of x:

x * f(x)= (01010111) x (00000001) = (10101110)

x^2 * f(x)= (01010111) x (00000100) = (01011100) $\oplus$ (00011011) = (01000111)

x^3 * f(x)= (01010111) x (00001000) = (10001110)

x^4 * f(x)= (01010111) x (00010000) = (00011100) $\oplus$ (00011011) = (00000111)

x^5 * f(x)= (01010111) x (00100000) = (00001110)

x^6 * f(x)= (01010111) x (01000000) = (00011100)

x^7 * f(x)= (01010111) x (10000000) = (00111000)

So,

(01010111) x (10000011) = (01010111) x [(00000001) x (00000010) x (10000000)]

= (01010111) $\oplus$ (10101110) $\oplus$ (00111000) = (11000001)

which is equivalent to $x^7 + x^6 + 1$.

Computational Example

- in $GF(2^3)$ have： (x^2+1) is $(101)_2$ & $(x+1)$ is $(011)_2$
- in the finite field $GF(2^3)$, with the irreducible polynomial $m(x) = x^3 + x + 1$.
- so addition is
  - $(x^2+1) + (x+1) = x^2 + x$
  - $(101)_2$ XOR $(011)_2 = 110_2$
- and multiplication is
  - $(x+1)\cdot(x^2+1) = x\cdot +1) + 1\cdot (x^2+1)$
    $= x^3+x+x^2+1 = x^3+x^2+x+1$
  - $(011)_2\cdot(101)_2 = (101)_2$<<1 XOR $(101)_2$<<0 =
    $(1010)_2$ XOR $(101)_2 = (1111)_2$
- polynomial modulo reduction (get q(x) & r(x)) is
  - (x^3+x^2+x+1) mod $(x^3+x+1) = 1\cdot (x^3+x+1) + (x^2) = x^2$
  - $(1111)_2$ mod $(1011)_2 = (1111)_2$ XOR $(1011)_2 = (0100)_2$

Summary: Operations of GF(2^n)

- Any element of GF(2^n) is represented as a polynomial
  - E.g. 1100: $x^3 + x^2$
- +(addition)
  - polynomial notation: adding corresponding coefficients based on modulo 2
  - binary notation: a bitwise XOR operation.
- ·(multiplication)
  - polynomial notation: perform the ordinary rules of polynomial arithmetic and Arithmetic on the coefficients is performed modulo 2, then modulo some irreducible polynomial m(x) of degree
  - binary notation: multiplication is shift & XOR

Outline

- 2DES, 3DES
- AES History
- Mathematical Basis —— GF(2^n)
- AES Principle
- AES Security Analysis
- Implementation of AES

The AES Cipher - Rijndael

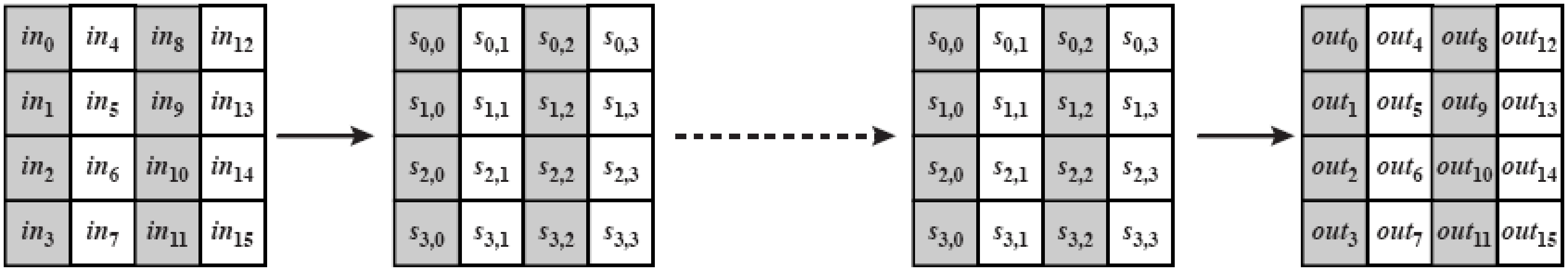
- designed by Rijmen-Daemen in Belgium
- In Rijndael proposal for AES, block data and key length can have independently 128, 192, or 256 bits.
- AES has 128/192/256 bit keys, 128 bit block data

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

Assume a **key length of 128 bits** in this section.

Parameters of AES



(a) Input, state array, and output

(b) Key and expanded key

- **Input, State, Output of each block have 4 columns of 4 bytes(16*8=128bits)**
- **128-bit (16-byte) key and expands into array of 44 32-bit words**
- **Note the ordering of bytes within a matrix is by column**

An AES Example

-

Plaintext:	0123456789abcdeffedcba9876543210
Key:	0f1571c947d9e8590cb7add6af7f6798
Ciphertext:	ff0b844a0853bf7c6934ab4364148fb9

- **Input:**

01	89	fe	76
23	ab	dc	54
45	cd	ba	32
67	ef	98	10

Note the ordering of bytes within a matrix is by column

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

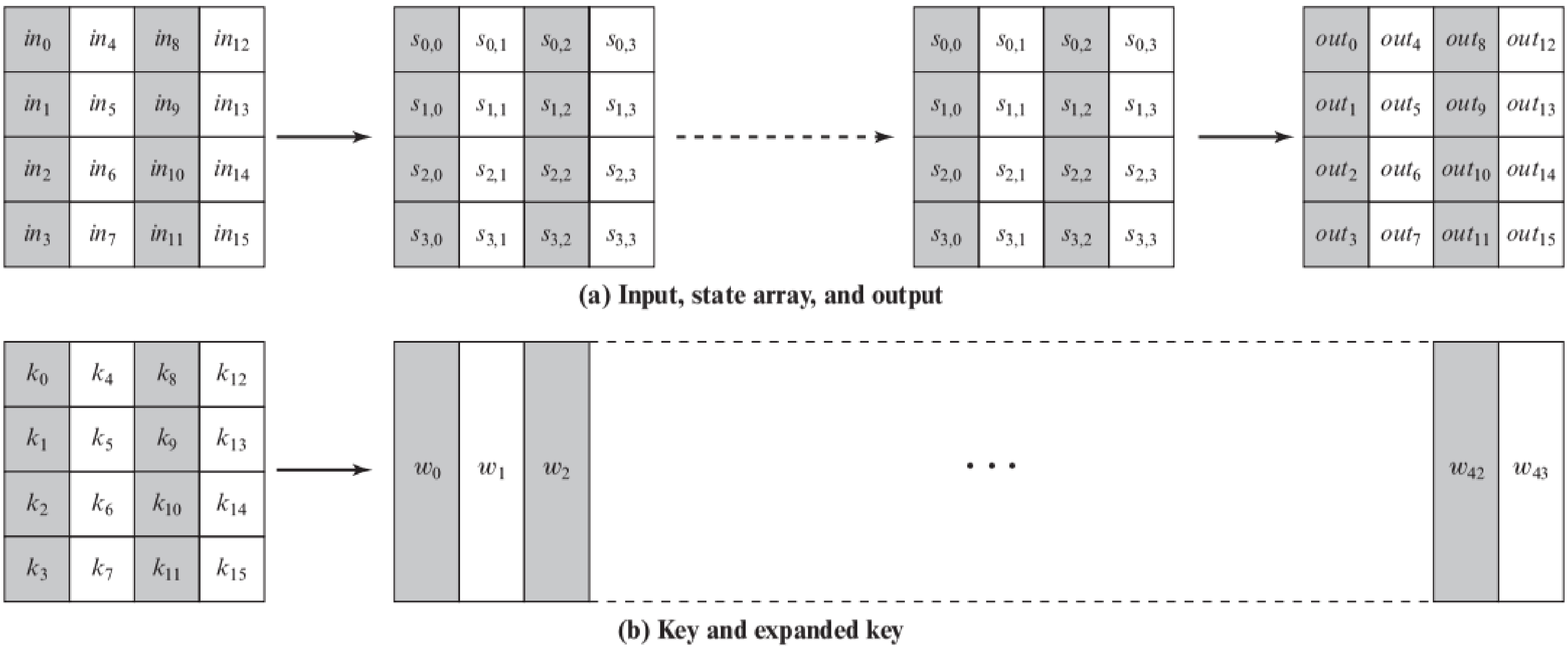
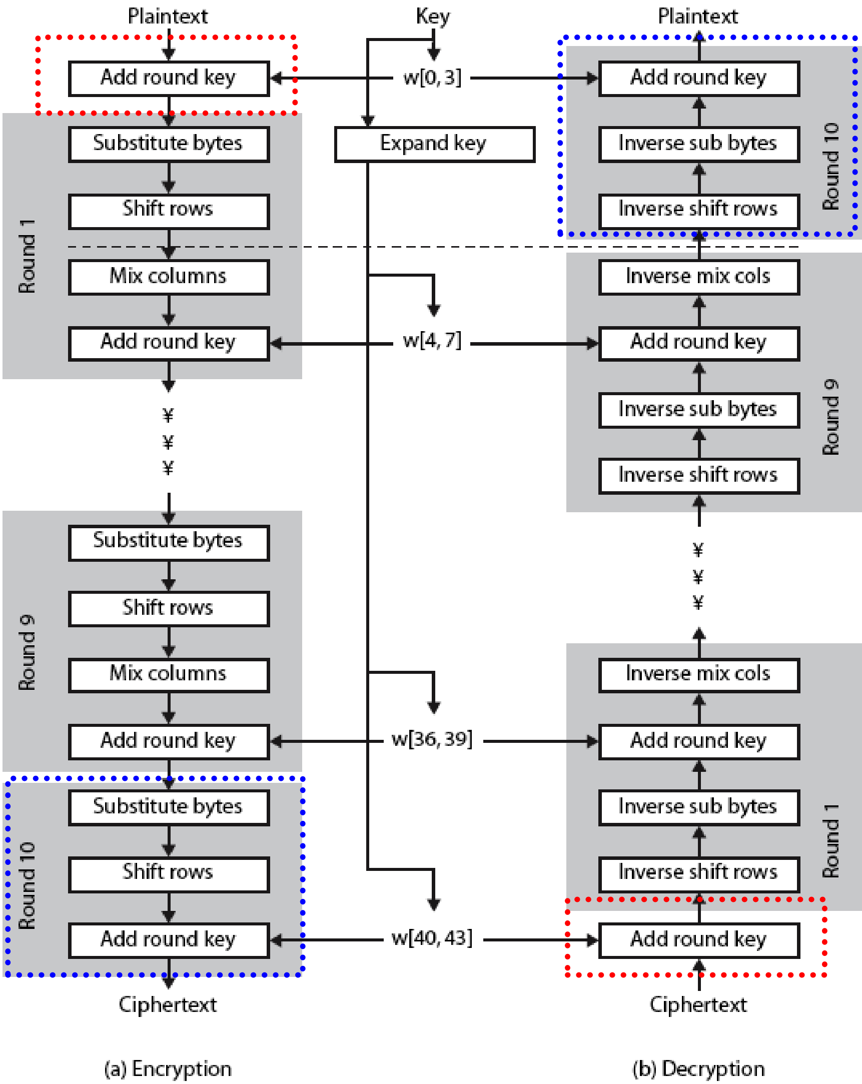


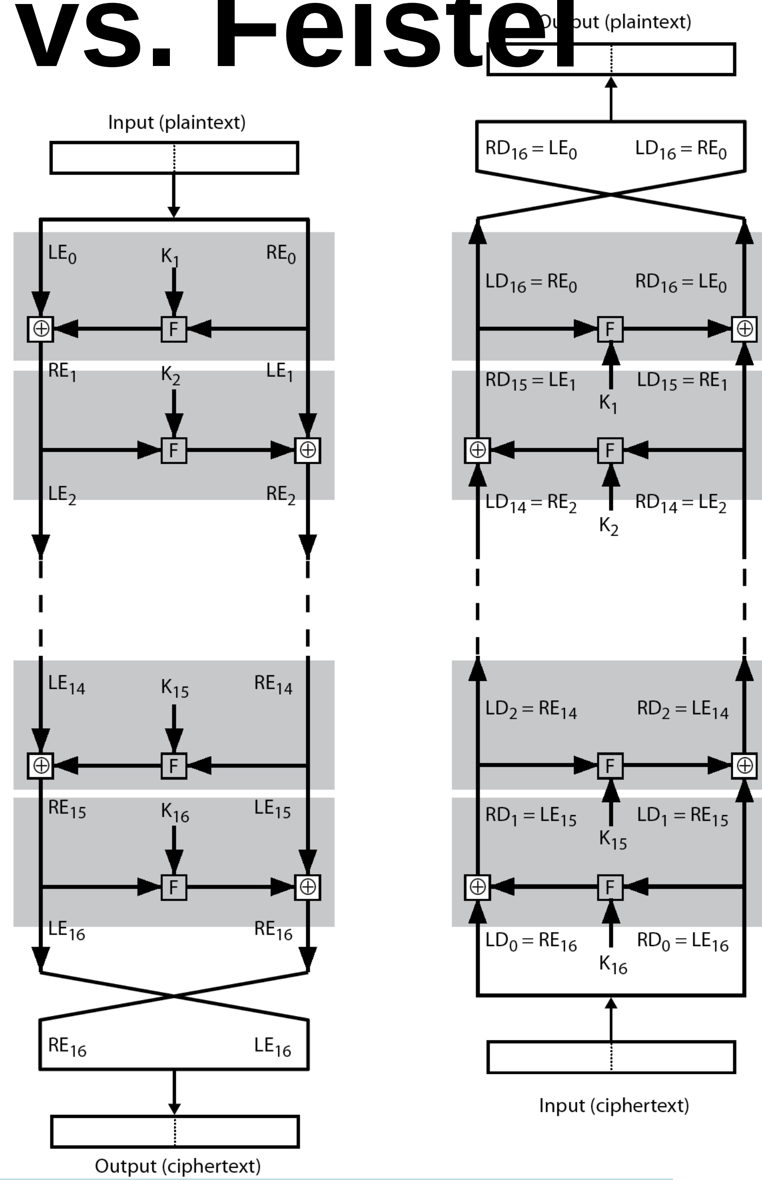
Figure 6.2 AES Data Structures

Structure of AES vs. Feistel



(a) Encryption

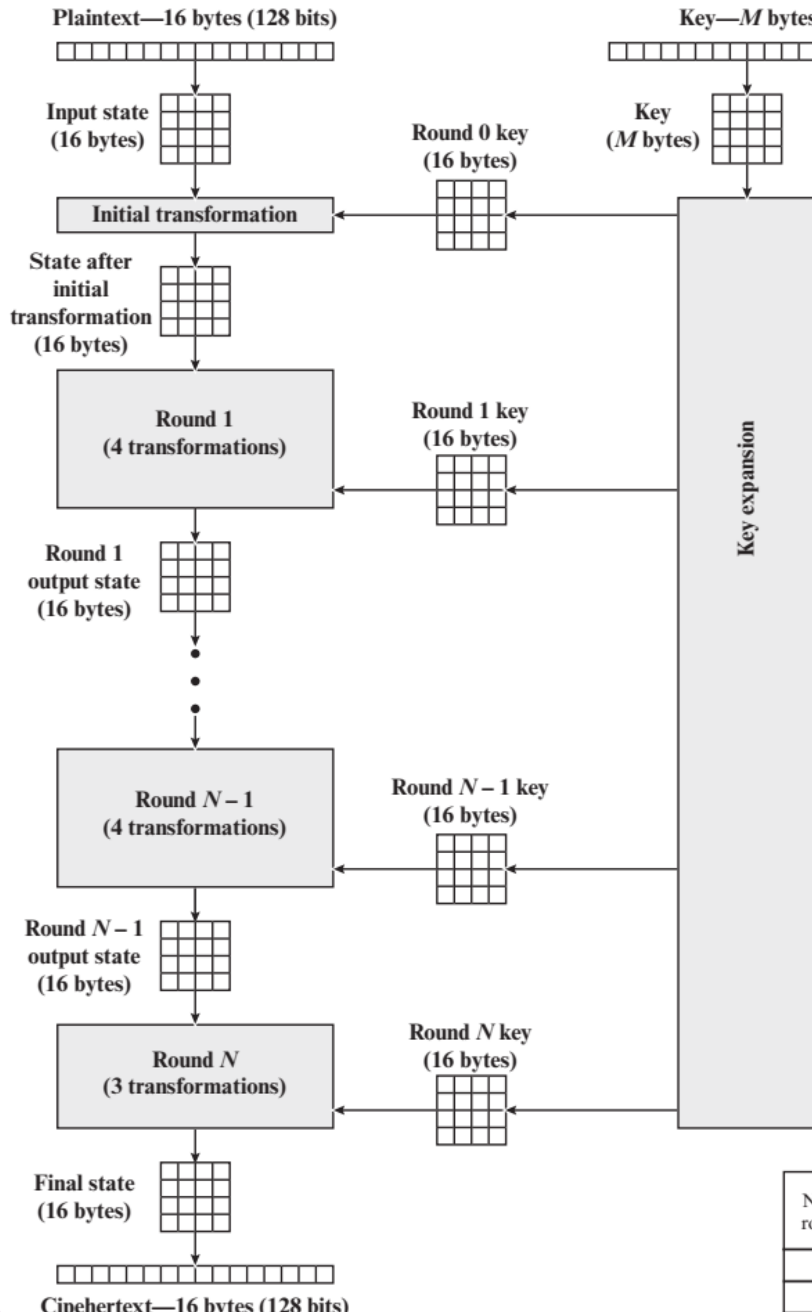
(b) Decryption



AES is an iterative rather than feistel cipher

Rijndael

- has 10 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multipy of groups)
  - add round key (XOR state with key material)
- Each round can be viewed as alternating(交替) XOR key & scramble data bytes
- Each step is invertible
- initial XOR key material & incomplete last round
- with fast XOR & table lookup implementation



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

Figure 6.1 AES Encryption Process

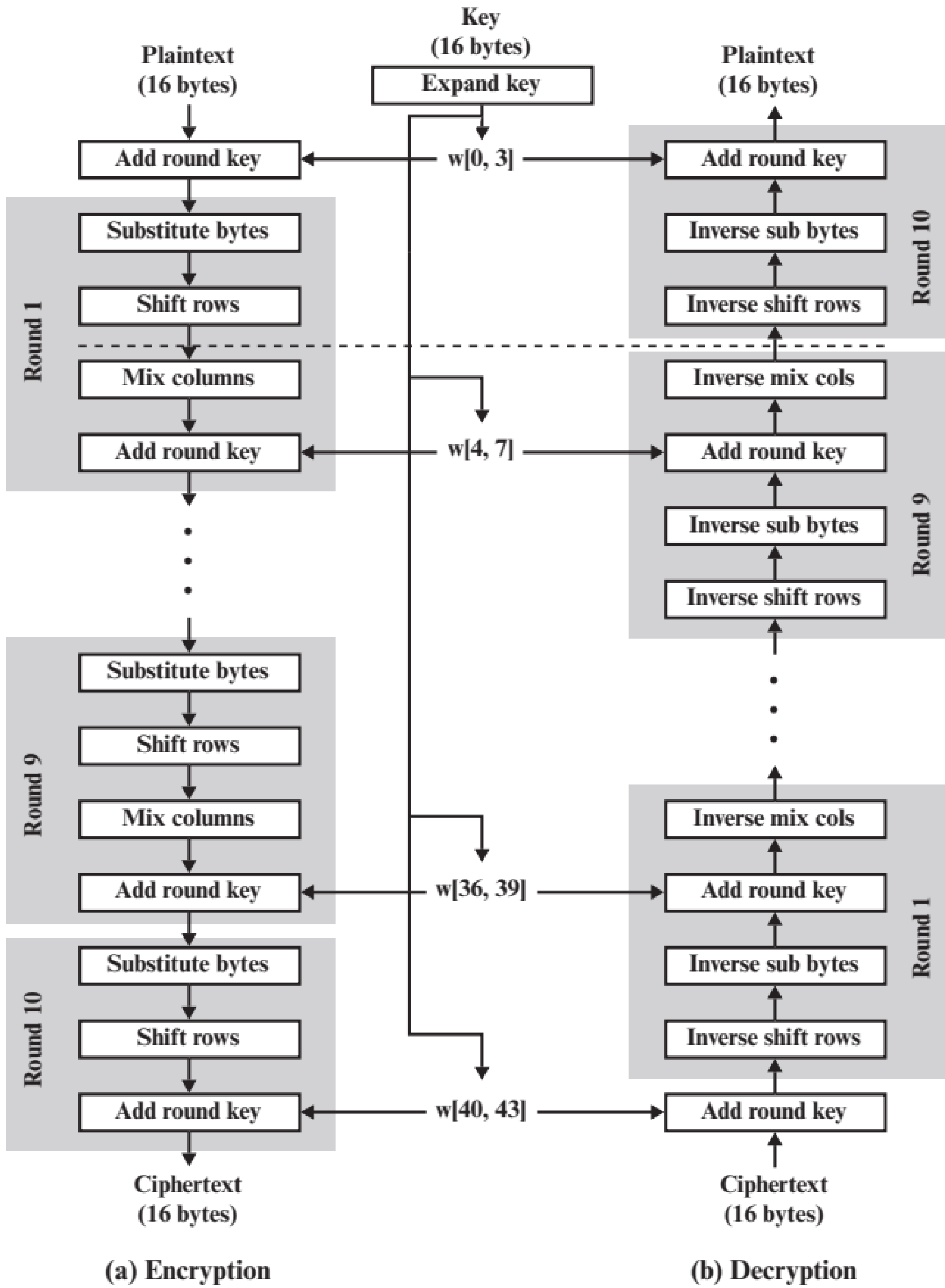
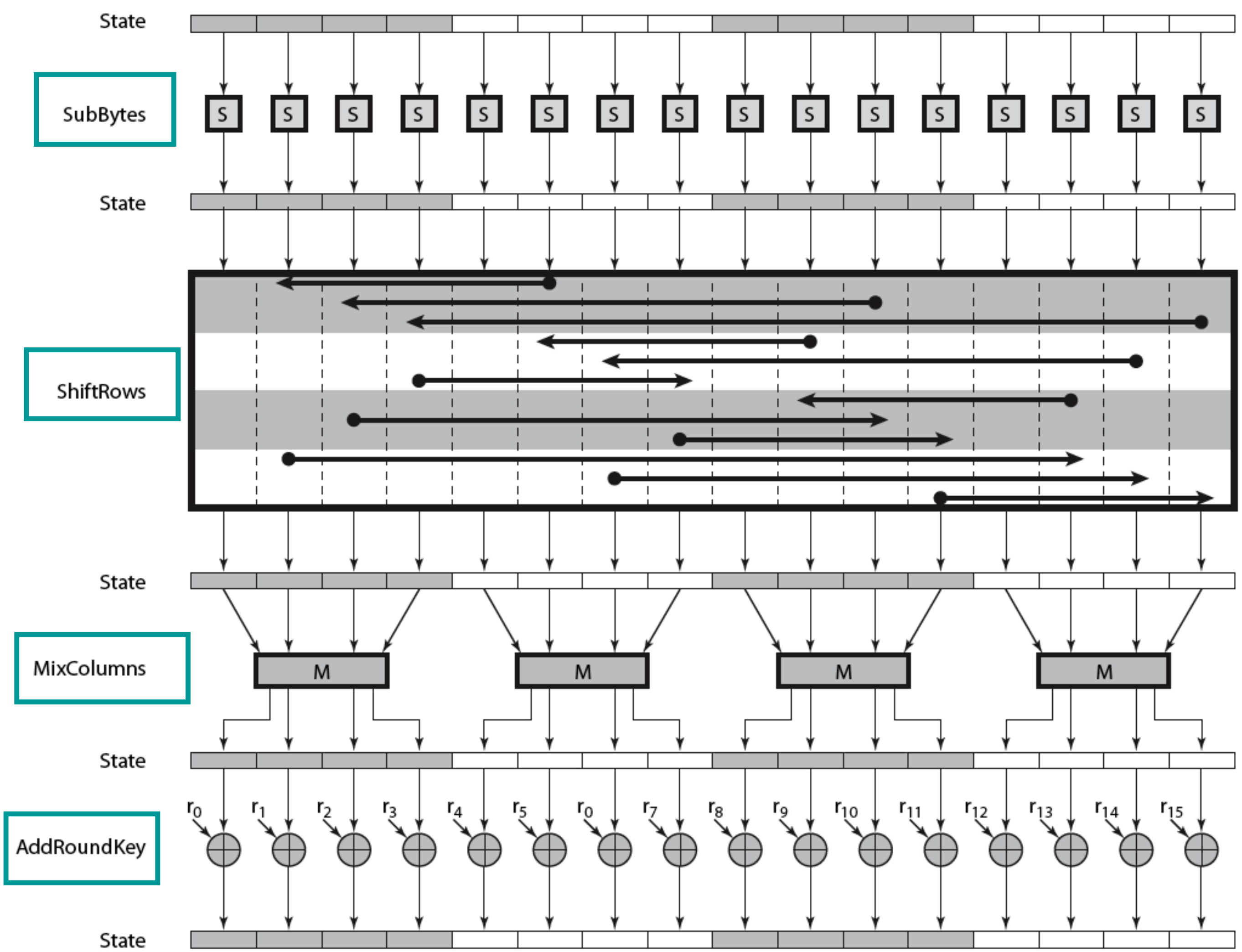


Figure 6.3 AES Encryption and Decryption

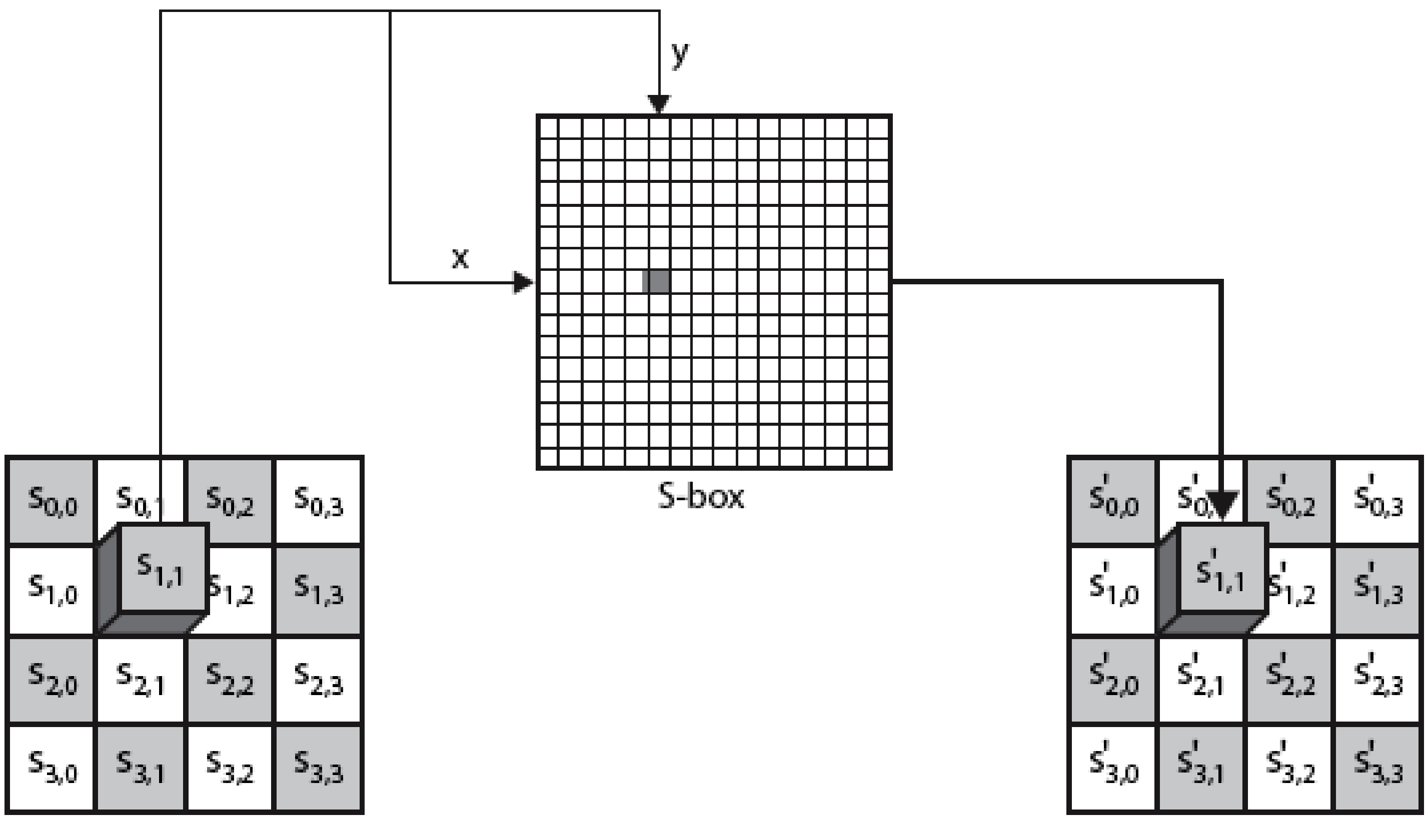
AES Round—— 4 transformations



(1)Byte Substitution

- a simple substitution of each byte
- uses one table of 16x16 bytes containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte indexed by row (left 4-bits) & column (right 4-bits)
  - eg. byte {95} is replaced by byte in row 9 column 5
  - which has value {2A}
- S-box must be invertible and can be constructed using defined transformation of values in $GF(2^8)$
- designed to be resistant to all known attacks
  - has a low correlation between input bits and output bits

Byte Substitution



(a) S-box

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

an example of the SubBytes transformation:

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

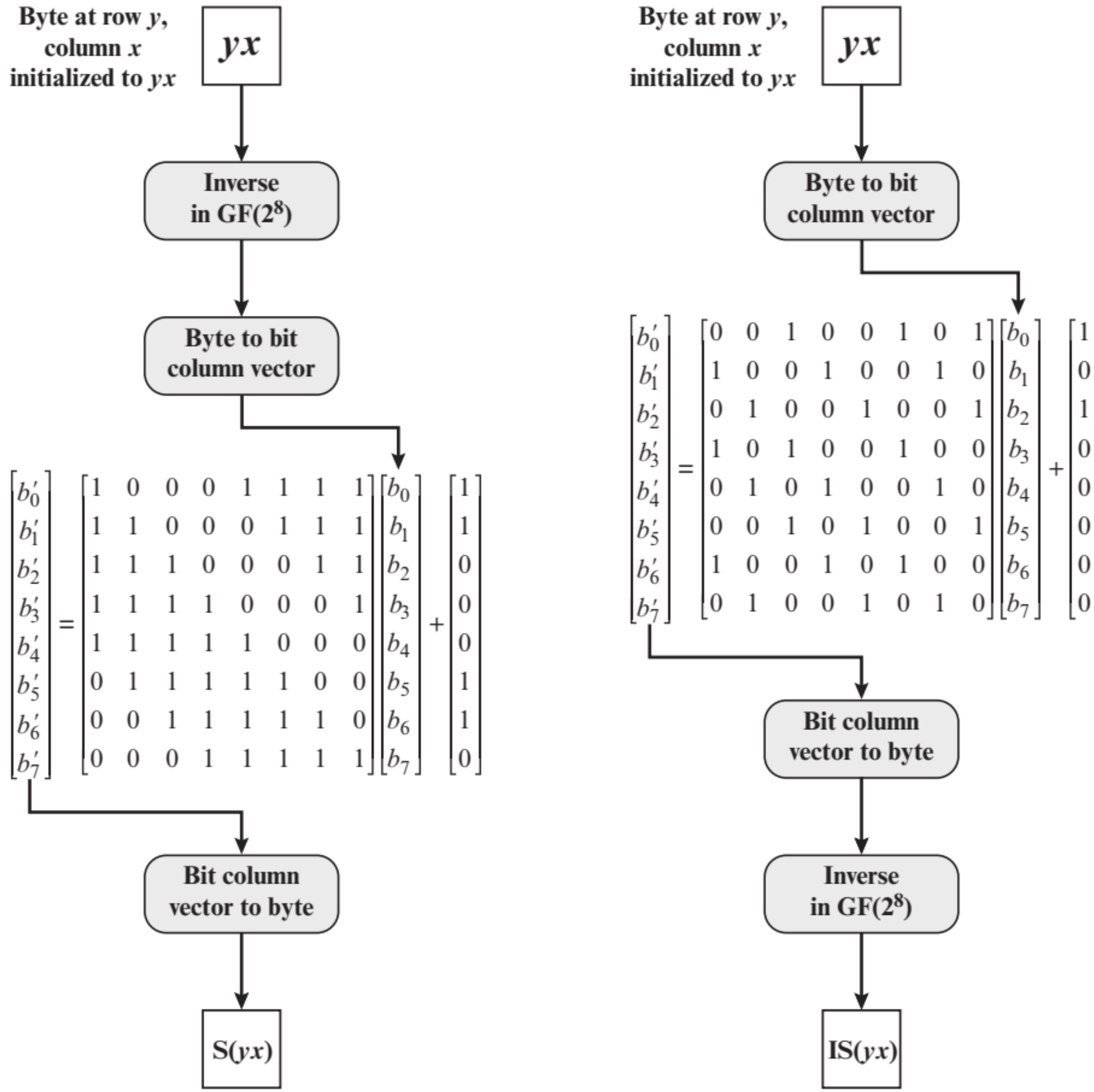
→

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

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

{4C} -> {5D}



$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1&0&0&0&1&1&1&1 \\ 1&1&0&0&0&1&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&1&0&0&0&1 \\ 1&1&1&1&1&0&0&0 \\ 0&1&1&1&1&1&0&0 \\ 0&0&1&1&1&1&1&0 \\ 0&0&0&1&1&1&1&1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 0&0&1&0&0&1&0&1 \\ 1&0&0&1&0&0&1&0 \\ 0&1&0&0&1&0&0&1 \\ 1&0&1&0&0&1&0&0 \\ 0&1&0&1&0&0&1&0 \\ 0&0&1&0&1&0&0&1 \\ 1&0&0&1&0&1&0&0 \\ 0&1&0&0&1&0&1&0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

(a) Calculation of byte at row y, column x of S-box

(a) Calculation of byte at row y, column x of IS-box

Figure 6.6 Constuction of S-Box and IS-Box

How to Construct S-box ?

For {xy}, compute its result {mn} after byte substitution transformation

- **Step1: For {xy}, compute its multiplicative inverse {ab} in the finite field GF(2^8); the value {00} is mapped to itself.**
- **Step2:**
  - **Assume {ab} == $(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)_2$, then {mn} =$(b'_7, b'_6, b'_5, b'_4, b'_3, b'_2, b'_1, b'_0)_2$**

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Example

- **consider the input value {95}. The result is {2A}, which should appear in row {09} column {05} of the S-box**
- **The multiplicative inverse in GF(2^8) is $\{95\}^{-1} = \{8A\} = (10001010)_2$**
- $$\begin{bmatrix} 1&0&0&0&1&1&1&1\\ 1&1&0&0&0&1&1&1\\ 1&1&1&0&0&0&1&1\\ 1&1&1&1&0&0&0&1\\ 1&1&1&1&1&0&0&0\\ 0&1&1&1&1&1&0&0\\ 0&0&1&1&1&1&1&0\\ 0&0&0&1&1&1&1&1 \end{bmatrix} \underbrace{\begin{bmatrix} 0\\1\\0\\1\\0\\0\\0\\1 \end{bmatrix}}_{\{8A\}} \oplus \begin{bmatrix} 1\\1\\0\\0\\0\\1\\1\\0 \end{bmatrix} = \begin{bmatrix} 1\\0\\0\\1\\0\\0\\1\\0 \end{bmatrix} \oplus \begin{bmatrix} 1\\1\\0\\0\\0\\1\\1\\0 \end{bmatrix} = \underbrace{\begin{bmatrix} 0\\1\\0\\1\\0\\1\\0\\0 \end{bmatrix}}_{\{2A\}}$$

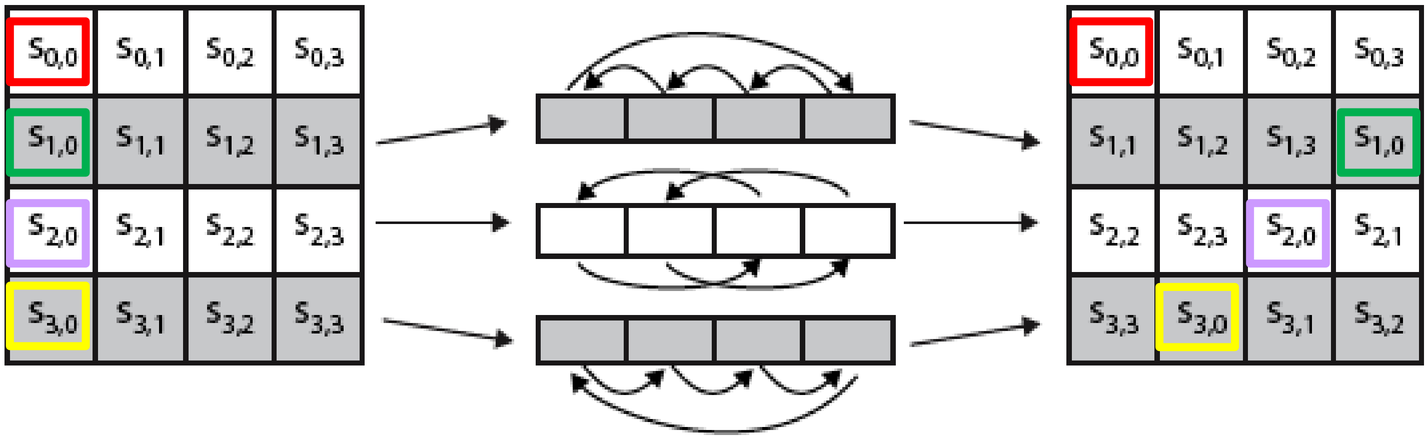
RATIONALE for S-Box design

- **resistant to known cryptanalytic attacks.**
- **has a low correlation between input and output**
- **the output is not a linear mathematical function of the input**
  - **due to the use of the multiplicative inverse.**
- **has no fixed points** $[\text{S-box}(a) = a]$ **and no "opposite fixed points"** $[\text{S-box}(a) = \overline{a}]$, where $\overline{a}$ **is the bitwise complement of a.**
- **must be invertible, but does not self-inverse**
  - **IS-box[S-box(a)]=a.   S-box(a)≠IS-box(a).**
  - **For example, S-box({95})={2A}, but S-box({95})={AD}.**

(2) Shift Rows

- a circular byte shift in each each
  - 1st row is unchanged
  - 2nd row does 1 byte circular shift to left
  - 3rd row does 2 byte circular shift to left
  - 4th row does 3 byte circular shift to left
- decrypt inverts using shifts to right
- since state is processed by columns, this step permutes bytes between the columns

Shift Rows



- a row shift moves an individual byte from one column to another, which is a linear distance of a multiple of 4 bytes.

- Also note that the transformation ensures that the 4 bytes of one column are spread out to four different columns

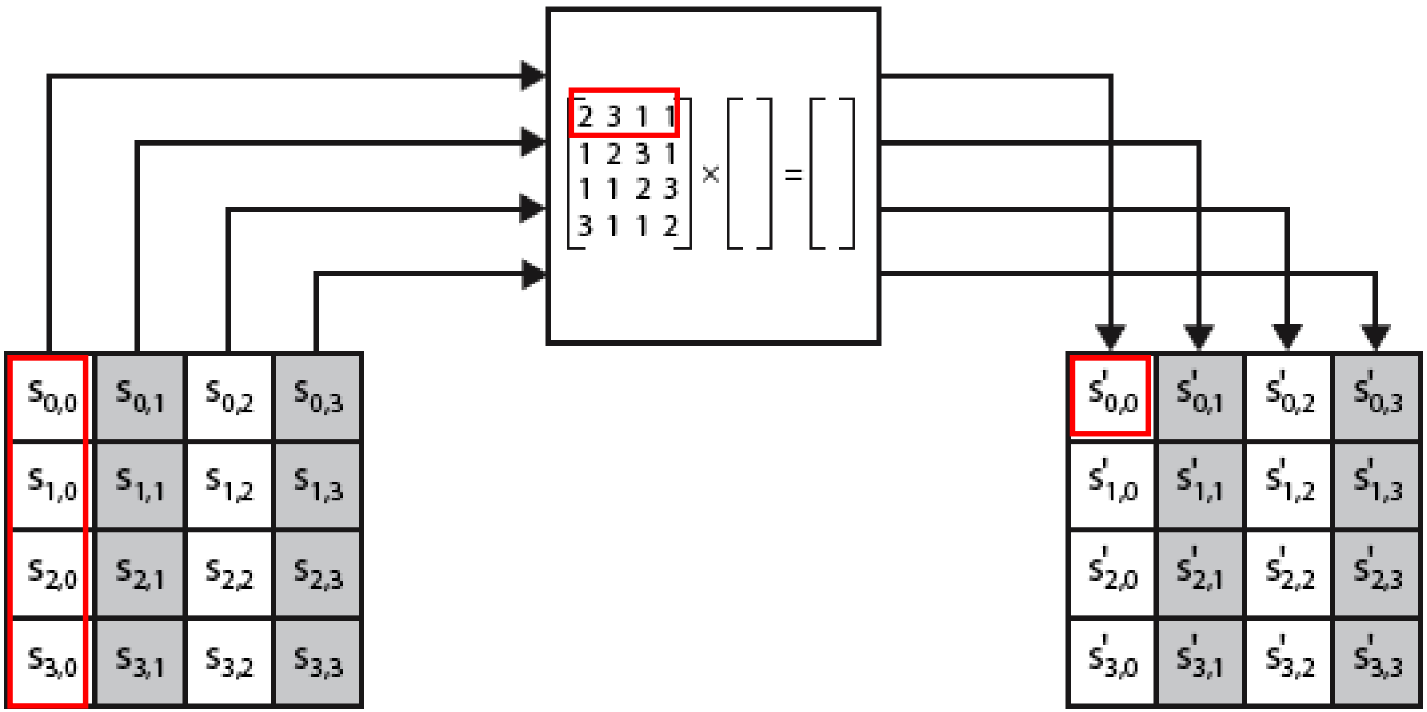
(3) Mix Columns

- **each column is processed separately**
- **each byte is replaced by a value dependent on all 4 bytes in the column**
- **effectively a matrix multiplication in $GF(2^8)$ using prime poly $m(x) = x^8+x^4+x^3+x+1$**

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

Mix Columns

each byte is replaced by a value dependent on all 4 bytes in the column.



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

Mix Columns

- **can express each col as 4 equations**
  - **to derive each new byte in col**

$$
\begin{aligned}
s'_{0,j} &= (2 \cdot s_{0,j}) \oplus (3 \cdot s_{1,j}) \oplus s_{2,j} \oplus s_{3,j} \\
s'_{1,j} &= s_{0,j} \oplus (2 \cdot s_{1,j}) \oplus (3 \cdot s_{2,j}) \oplus s_{3,j} \\
s'_{2,j} &= s_{0,j} \oplus s_{1,j} \oplus (2 \cdot s_{2,j}) \oplus (3 \cdot s_{3,j}) \\
s'_{3,j} &= (3 \cdot s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \cdot s_{3,j})
\end{aligned}
$$

An Example of MixColumns

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$$
\begin{aligned}
(\{02\}\cdot\{87\}) &\oplus (\{03\}\cdot\{6E\}) \oplus \{46\} \oplus \{A6\} = \{47\}\\
\{87\} &\oplus (\{02\}\cdot\{6E\}) \oplus (\{03\}\cdot\{46\}) \oplus \{A6\} = \{37\}\\
\{87\} &\oplus \{6E\} \oplus (\{02\}\cdot\{46\}) \oplus (\{03\}\cdot\{A6\}) = \{94\}\\
(\{03\}\cdot\{87\}) &\oplus \{6E\} \oplus \{46\} \oplus (\{02\}\cdot\{A6\}) = \{ED\}
\end{aligned}
$$

For the first equation, we have $\{02\}\cdot\{87\} = (0000\ 1110) \oplus \boxed{(0001\ 1011)} = (0001\ 0101)$ and $\{03\}\cdot\{6E\} = \{6E\} \oplus (\{02\}\cdot\{6E\}) = (0110\ 1110) \oplus (1101\ 1100) = (1011\ 0010)$. Then,

$$
\begin{aligned}
\{02\}\cdot\{87\} &= 0001\ 0101\\
\{03\}\cdot\{6E\} &= 1011\ 0010\\
\{46\} &= 0100\ 0110\\
\{A6\} &= \underline{1010\ 0110}\\
& \ 0100\ 0111 = \{47\}
\end{aligned}
$$

prime poly m(x)
$=x^8+x^4+x^3+x+1$

Mix Columns

- **decryption requires use of inverse matrix**
  - **with larger coefficients, hence a little harder**

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

- **Can prove:**

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{0,3} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

which is equivalent to showing

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Choice of coefficients in MixColumns

- based on a linear code with maximal distance between code words, which ensures a good mixing among the bytes of each column.
- influenced by implementation considerations
  - All are {01}, {02},or {03}.
  - multi-plication by these coefficients involves at most a shift and an XOR.

Coefficients in InvMixColumns

- **But, the coefficients in InvMixColumns are more formidable to implement.**
  - **encryption was deemed more important than decryption for two reasons:**
    - ✓ **1. For the CFB and OFB cipher modes (Figures 7.5 and 7.6; described in Chapter 7), only encryption is used.**
    - ✓ **2. As with any block cipher, AES can be used to construct a message authentication code (Chapter 13), and for this, only encryption is used.**

Another way of characterizing the MixColumns transformation

- have an alternate characterization, see Appendix 5A
  - each column a 4-term polynomial
  - with coefficients in $GF(2^8)$
  - and polynomials multiplied modulo (x^4+1)

(4) Add Round Key

- XOR state with 128-bits of the round key
- again processed by column (though effectively a series of byte operations)
- the only step which makes use of the key and obscures the result, hence MUST be used at start and end of each round
- inverse for decryption identical
  - since XOR own inverse, with reversed keys

Add Round Key

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \oplus \begin{bmatrix} w_i & w_{i+1} & w_{i+2} & w_{i+3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

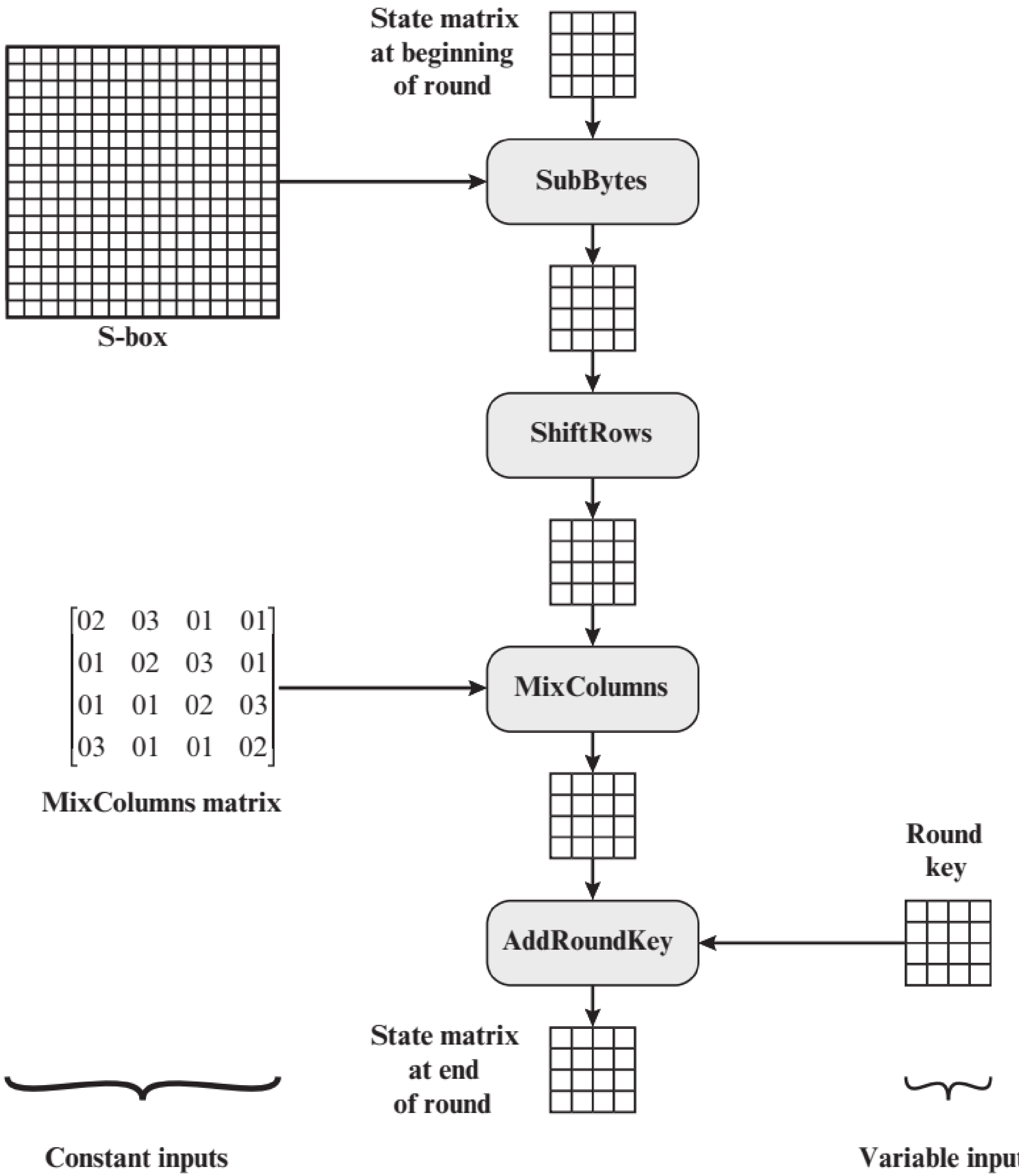
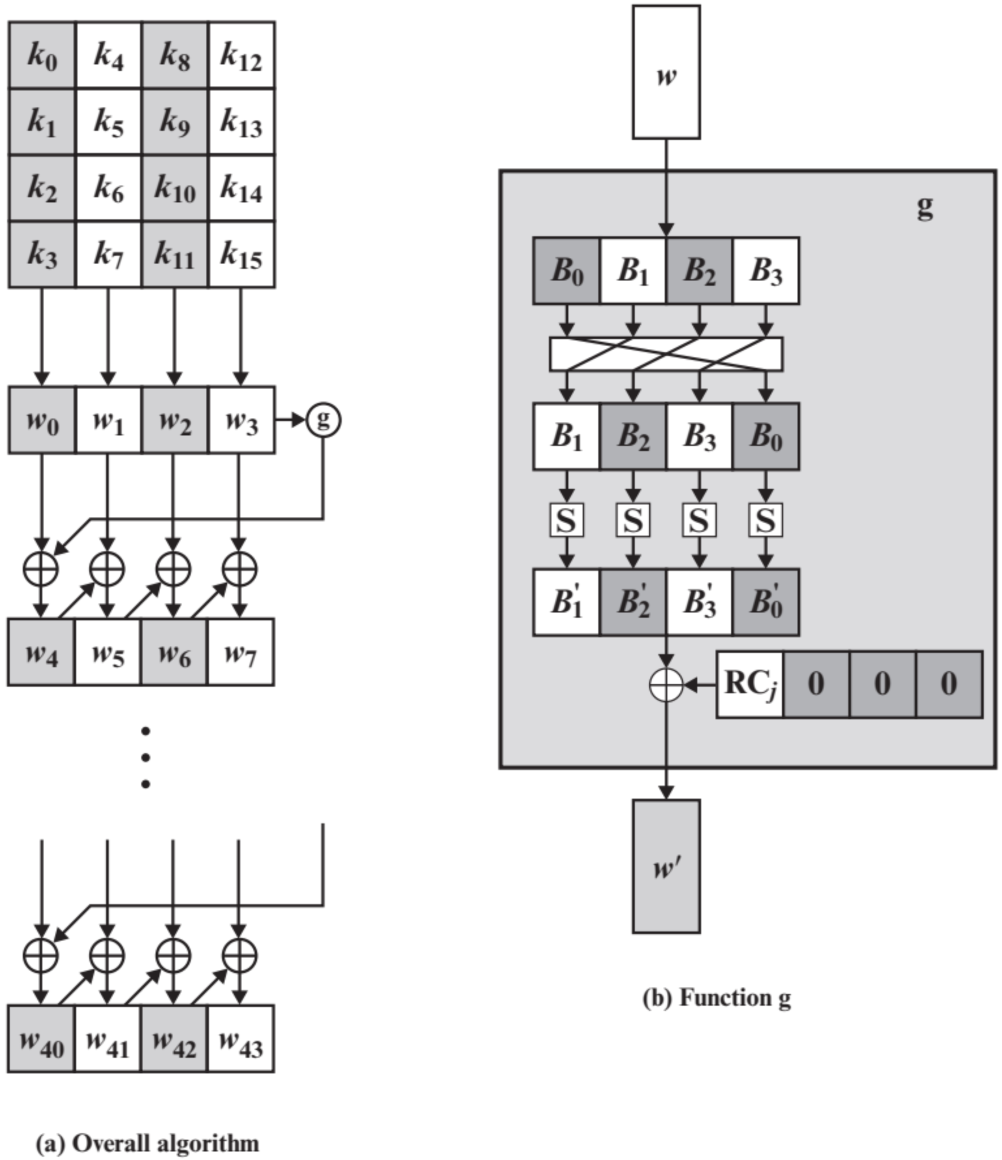


Figure 6.8 Inputs for Single AES Round

AES Key Expansion

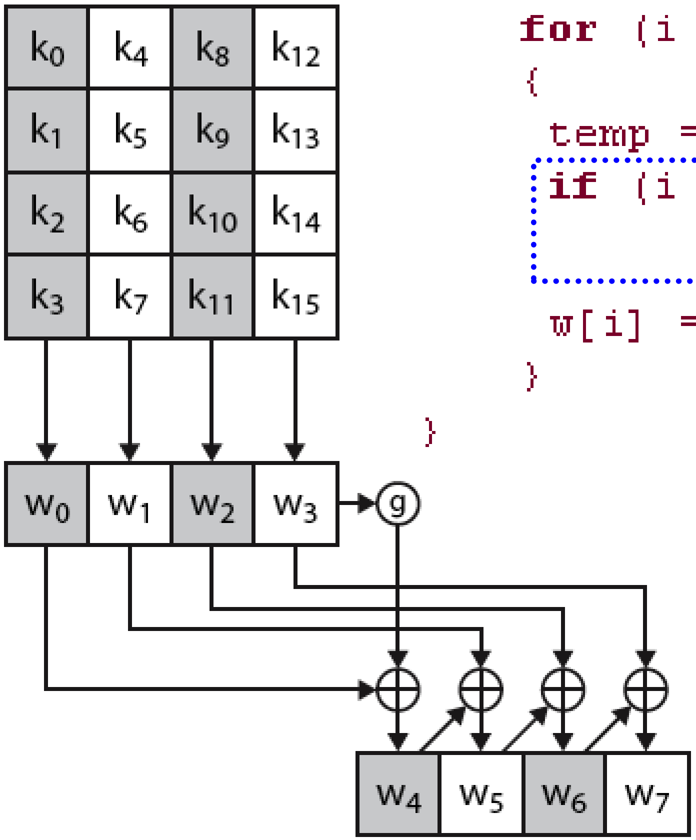
- takes 128-bit (16-byte) key and expands into array of 44 32-bit words
- start by copying key into first 4 words
- then loop creating words that depend on values in previous & 4 places back
  - in 3 of 4 cases just XOR these together
  - 1st word in 4 has rotate + S-box + XOR round constant on previous, before XOR 4th back

AES Key Expansion



AES Key Expansion

```
KeyExpansion (byte key[16], word w[44])
{
     word temp
     for (i = 0; i < 4; i++) w[i] = (key[4*i],
    key[4*i+1], key[4*i+2], key[4*i+3]);
     for (i = 4; i < 44; i++)
     {
      temp = w[i - 1];
      if (i mod 4 = 0) temp = SubWord (RotWord (temp))
                                ⊕ Rcon[i/4];
      w[i] = w[i - 4] ⊕ temp
     }
}
```



g

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36

Plaintext:	0123456789abcdeffedcba9876543210
Key:	0f1571c947d9e8590cb7add6af7f6798
Ciphertext:	ff0b844a0853bf7c6934ab4364148fb9

Table 6.3 Key Expansion for AES Example

Key Words	Auxiliary Function
w0 = 0f 15 71 c9 w1 = 47 d9 e8 59 w2 = 0c b7 ad d6 w3 = af 7f 67 98	RotWord (w3) = 7f 67 98 af = x1 SubWord (x1) = d2 85 46 79 = y1 Rcon (1) = 01 00 00 00 y1 ⊕ Rcon (1) = d3 85 46 79 = z1
w4 = w0 ⊕ z1 = dc 90 37 b0 w5 = w4 ⊕ w1 = 9b 49 df e9 w6 = w5 ⊕ w2 = 97 fe 72 3f w7 = w6 ⊕ w3 = 38 81 15 a7	RotWord (w7) = 81 15 a7 38 = x2 SubWord (x2) = 0c 59 5c 07 = y2 Rcon (2) = 02 00 00 00 y2 ⊕ Rcon (2) = 0e 59 5c 07 = z2
w8 = w4 ⊕ z2 = d2 c9 6b b7 w9 = w8 ⊕ w5 = 49 80 b4 5e w10 = w9 ⊕ w6 = de 7e c6 61 w11 = w10 ⊕ w7 = e6 ff d3 c6	RotWord (w11) = ff d3 c6 e6 = x3 SubWord (x3) = 16 66 b4 83 = y3 Rcon (3) = 04 00 00 00 y3 ⊕ Rcon (3) = 12 66 b4 8e = z3
w12 = w8 ⊕ z3 = c0 af df 39 w13 = w12 ⊕ w9 = 89 2f 6b 67 w14 = w13 ⊕ w10 = 57 51 ad 06 w15 = w14 ⊕ w11 = b1 ae 7e c0	RotWord (w15) = ae 7e c0 b1 = x4 SubWord (x4) = e4 f3 ba c8 = y4 Rcon (4) = 08 00 00 00 y4 ⊕ Rcon (4) = ec f3 ba c8 = 4

(Continued)

Table 6.3 Continued

Key Words	Auxiliary Function
w16 = w12 ⊕ z4 = 2c 5c 65 f1 w17 = w16 ⊕ w13 = a5 73 0e 96 w18 = w17 ⊕ w14 = f2 22 a3 90 w19 = w18 ⊕ w15 = 43 8c dd 50	RotWord (w19) = 8c dd 50 43 = x5 SubWord (x5) = 64 c1 53 1a = y5 Rcon(5) = 10 00 00 00 y5 ⊕ Rcon (5) = 74 c1 53 1a = z5
w20 = w16 ⊕ z5 = 58 9d 36 eb w21 = w20 ⊕ w17 = fd ee 38 7d w22 = w21 ⊕ w18 = 0f cc 9b ed w23 = w22 ⊕ w19 = 4c 40 46 bd	RotWord (w23) = 40 46 bd 4c = x6 SubWord (x6) = 09 5a 7a 29 = y6 Rcon(6) = 20 00 00 00 y6 ⊕ Rcon(6) = 29 5a 7a 29 = z6
w24 = w20 ⊕ z6 = 71 c7 4c c2 w25 = w24 ⊕ w21 = 8c 29 74 bf w26 = w25 ⊕ w22 = 83 e5 ef 52 w27 = w26 ⊕ w23 = cf a5 a9 ef	RotWord (w27) = a5 a9 ef cf = x7 SubWord (x7) = 06 d3 bf 8a = y7 Rcon (7) = 40 00 00 00 y7 ⊕ Rcon(7) = 46 d3 df 8a = z7
w28 = w24 ⊕ z7 = 37 14 93 48 w29 = w28 ⊕ w25 = bb 3d e7 f7 w30 = w29 ⊕ w26 = 38 d8 08 a5 w31 = w30 ⊕ w27 = f7 7d a1 4a	RotWord (w31) = 7d a1 4a f7 = x8 SubWord (x8) = ff 32 d6 68 = y8 Rcon (8) = 80 00 00 00 y8 ⊕ Rcon(8) = 7f 32 d6 68 = z8
w32 = w28 ⊕ z8 = 48 26 45 20 w33 = w32 ⊕ w29 = f3 1b a2 d7 w34 = w33 ⊕ w30 = cb c3 aa 72 w35 = w34 ⊕ w32 = 3c be 0b 3	RotWord (w35) = be 0b 38 3c = x9 SubWord (x9) = ae 2b 07 eb = y9 Rcon (9) = 1B 00 00 00 y9 ⊕ Rcon (9) = b5 2b 07 eb = z9
w36 = w32 ⊕ z9 = fd 0d 42 cb w37 = w36 ⊕ w33 = 0e 16 e0 1c w38 = w37 ⊕ w34 = c5 d5 4a 6e w39 = w38 ⊕ w35 = f9 6b 41 56	RotWord (w39) = 6b 41 56 f9 = x10 SubWord (x10) = 7f 83 b1 99 = y10 Rcon (10) = 36 00 00 00 y10 ⊕ Rcon (10) = 49 83 b1 99 = z10
w40 = w36 ⊕ z10 = b4 8e f3 52 w41 = w40 ⊕ w37 = ba 98 13 4e w42 = w41 ⊕ w38 = 7f 4d 59 20 w43 = w42 ⊕ w39 = 86 26 18 76	

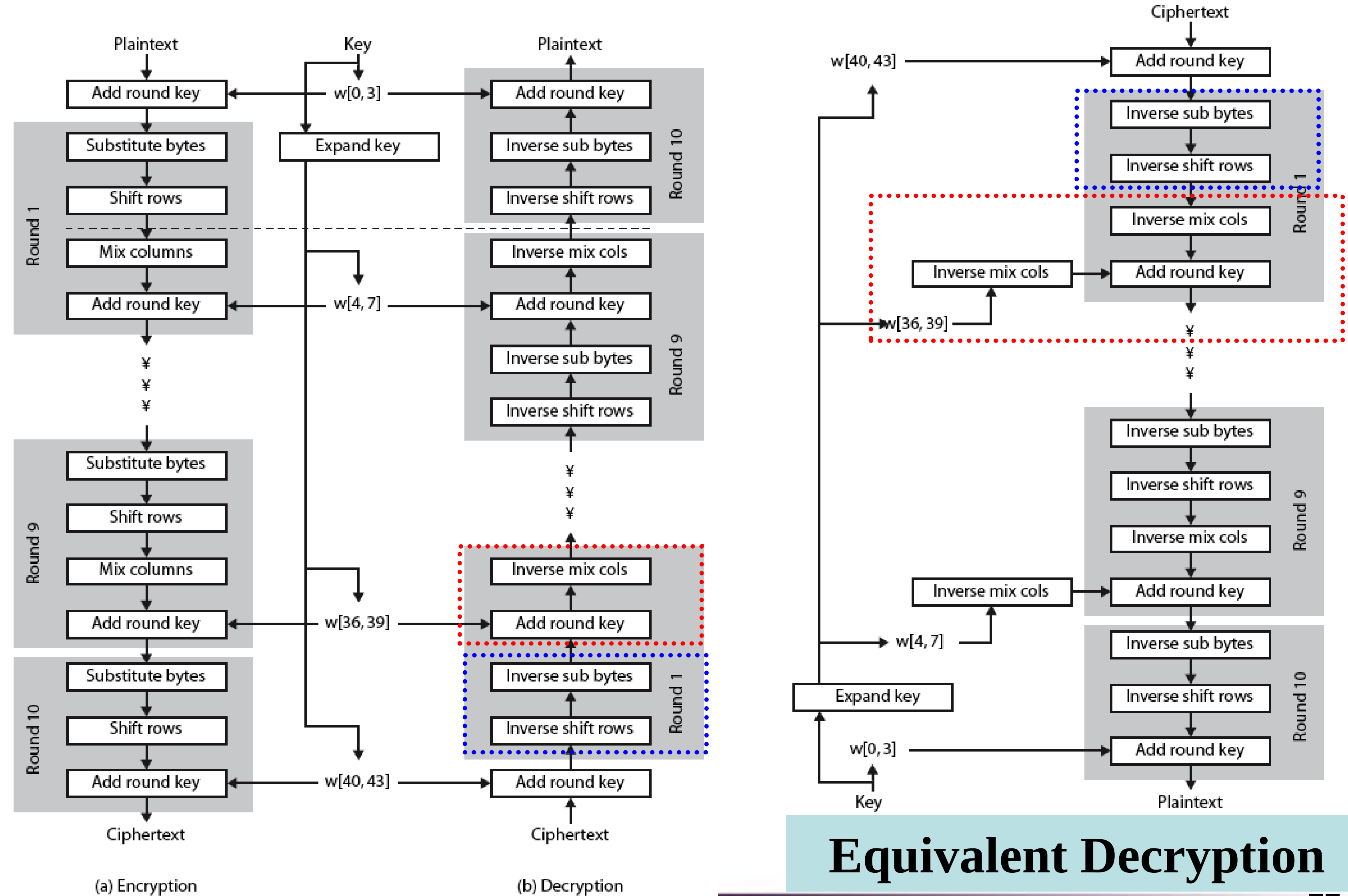
Key Expansion Rationale

- designed to resist known attacks
- design criteria included
  - knowing part key insufficient to find many more
  - invertible transformation
  - fast on wide range of CPU's
  - use round constants to break symmetry
  - diffuse key bits into round keys
  - enough non-linearity to hinder analysis
  - simplicity of description

AES Decryption

- AES decryption is not identical to encryption since steps done in reverse
- but can define an equivalent inverse cipher with steps as for encryption
  - but using inverses of each step
  - with a different key schedule
- works since result is unchanged when
  - swap byte substitution & shift rows
  - swap mix columns & add (tweaked) round key

$$\mathrm{InvMixColumns}(s_i \oplus w_j) = [\mathrm{InvMixColumns}(s_i)] \oplus [\mathrm{InvMixColumns}(w_j)]$$



Equivalent Decryption

Outline

- 2DES, 3DES
- AES History
- Mathematical Basis —— GF(2^n)
- AES Principle
- AES Security Analysis
- Implementation of AES

AES Security Analysis

- AES is efficient and highly secure it is believed.
- AES are a series of XOR with key then scramble/permute block repeated
- Add Round Key stage is the only step which makes use of the key and obscures the result, hence MUST be used at start and end of each round, since otherwise could undo effect of other steps.
- But the other steps provide confusion/diffusion/non-linearity.
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multiply of groups)
  - Key Expansion

Plaintext:	0123456789abcdeffedcba9876543210
Key:	0f1571c947d9e8590cb7add6af7f6798
Ciphertext:	ff0b844a0853bf7c6934ab4364148fb9

Table 6.5 Avalanche Effect in AES: Change in Plaintext

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0023456789abcdeffedcba9876543210	1
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c4a9ad090fc7ff3fc0e8e8ca4dd02a9c	20
2	5c7bb49a6b72349b05a2317ff46d1294 fe2ae569f7ee8bb8c1f5a2bb37ef53d5	58
3	7115262448dc747e5cdac7227da9bd9c ec093dfb7c45343d689017507d485e62	59
4	f867aee8b437a5210c24c1974cffeabc 43efdb697244df808e8d9364ee0ae6f5	61
5	721eb200ba06206dcbd4bce704fa654e 7b28a5d5ed643287e006c099bb375302	68
6	0ad9d85689f9f77bc1c5f71185e5fb14 3bc2d8b6798d8ac4fe36a1d891ac181a	64
7	db18a8ffa16d30d5f88b08d777ba4eaa 9fb8b5452023c70280e5c4bb9e555a4b	67
8	f91b4fbfe934c9bf8f2f85812b084989 20264e1126b219aef7feb3f9b2d6de40	65
9	cca104a13e678500ff59025f3bafaa34 b56a0341b2290ba7dfdfbddcd8578205	61
10	ff0b844a0853bf7c6934ab4364148fb9 612b89398d0600cde116227ce72433f0	58

key is 0e1571c947d9e8590cb7add6af7f6798

Table 6.6 Avalanche Effect in AES: Change in Key

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0123456789abcdeffedcba9876543210	0
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c5a9ad090ec7ff3fc1e8e8ca4cd02a9c	22
2	5c7bb49a6b72349b05a2317ff46d1294 90905fa9563356d15f3760f3b8259985	58
3	7115262448dc747e5cdac7227da9bd9c 18aeb7aa794b3b66629448d575c7cebf	67
4	f867aee8b437a5210c24c1974cffeabc f81015f993c978a876ae017cb49e7eec	63
5	721eb200ba06206dcbd4bce704fa654e 5955c91b4e769f3cb4a94768e98d5267	81
6	0ad9d85689f9f77bc1c5f71185e5fb14 dc60a24d137662181e45b8d3726b2920	70
7	db18a8ffa16d30d5f88b08d777ba4eaa fe8343b8f88bef66cab7e977d005a03c	74
8	f91b4fbfe934c9bf8f2f85812b084989 da7dad581d1725c5b72fa0f9d9d1366a	67
9	cca104a13e678500ff59025f3bafaa34 0ccb4c66bbfd912f4b511d72996345e0	59
10	ff0b844a0853bf7c6934ab4364148fb9 fc8923ee501a7d207ab670686839996b	53

Outline

- 2DES, 3DES
- AES History
- Mathematical Basis —— GF(2^n)
- AES Principle
- AES Security Analysis
- Implementation of AES

Implementation Aspects on 8-bit CPU

- can efficiently implement on 8-bit CPU
  - byte substitution works on bytes using a table of 256 entries
  - shift rows is simple byte shift
  - add round key works on byte XOR's
  - mix columns requires matrix multiply in GF(2^8) which works on byte values, can be simplified to use table lookups & byte XOR's

Simplified Implementation for mix columns

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

$$s'_{0,j} = (2 \bullet s_{0,j}) \oplus (3 \bullet s_{1,j}) \oplus s_{2,j} \oplus s_{3,j}$$
$$s'_{1,j} = s_{0,j} \oplus (2 \bullet s_{1,j}) \oplus (3 \bullet s_{2,j}) \oplus s_{3,j}$$
$$s'_{2,j} = s_{0,j} \oplus s_{1,j} \oplus (2 \bullet s_{2,j}) \oplus (3 \bullet s_{3,j})$$
$$s'_{3,j} = (3 \bullet s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \bullet s_{3,j})$$

$$Tmp = s_{0,j} \oplus s_{1,j} \oplus s_{2,j} \oplus s_{3,j}$$
$$s'_{0,j} = s_{0,j} \oplus Tmp \oplus [2 \bullet (s_{0,j} \oplus s_{1,j})]$$
$$s'_{1,j} = s_{1,j} \oplus Tmp \oplus [2 \bullet (s_{1,j} \oplus s_{2,j})]$$
$$s'_{2,j} = s_{2,j} \oplus Tmp \oplus [2 \bullet (s_{2,j} \oplus s_{3,j})]$$
$$s'_{3,j} = s_{3,j} \oplus Tmp \oplus [2 \bullet (s_{3,j} \oplus s_{0,j})]$$

Define a 256-byte table X2, where X2[i]={02}•i. Then,

$$Tmp = s_{0}, j \oplus s_{1}, j \oplus s2, j \oplus s3, j$$
$$s'_{0}, j = s_{0}, j \oplus \mathrm{Tmp} \oplus X2[s_{0}, j \oplus s_{1}, j]$$
$$s'_{1}, c = s_{1}, j \oplus \mathrm{Tmp} \oplus X2[s_{1}, j \oplus s_{2}, j]$$
$$s'_{2}, c = s_{2}, j \oplus \mathrm{Tmp} \oplus X2[s_{2}, j \oplus s_{3}, j]$$
$$s'_{3}, j = s_{3}, j \oplus \mathrm{Tmp} \oplus X2[s_{3}, j \oplus s_{0}, j]$$

Implementation Aspects on on 32-bit CPU

- **For some round: Input state a, Output state e.**

SubBytes	$b_{i,j} = S[\underline{a_{i,j}}]$
ShiftRows	$\begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix} = \begin{bmatrix} b_{0,j} \\ b_{1,j-1} \\ b_{2,j-2} \\ b_{3,j-3} \end{bmatrix}$
MixColumns	$\begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix}$
AddRoundKey	$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ \underline{e_{3,j}} \end{bmatrix} = \begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} \oplus \boxed{\begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}}$

Round Key

$$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S[a_{0,j}] \\ S[a_{1,j-1}] \\ S[a_{2,j-2}] \\ S[a_{3,j-3}] \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

a -> e

$$= \left(\begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \bullet S[a_{0,j}] \right) \oplus \left(\begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \bullet S[a_{1,j-1}] \right) \oplus \left(\begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \bullet S[a_{2,j-2}] \right)$$

$$\oplus \left(\begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \bullet S[a_{3,j-3}] \right) \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

$T_0[x] = \left(\begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \bullet S[x] \right)$	$T_1[x] = \left(\begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \bullet S[x] \right)$	$T_2[x] = \left(\begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \bullet S[x] \right)$	$T_3[x] = \left(\begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \bullet S[x] \right)$

$$\begin{bmatrix} s'_{0,j} \\ s'_{1,j} \\ s'_{2,j} \\ s'_{3,j} \end{bmatrix} = T_0[s_{0,j}] \oplus T_1[s_{1,j-1}] \oplus T_2[s_{2,j-2}] \oplus T_3[s_{3,j-3}] \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

s -> s'

Implementation Aspects on on 32-bit CPU

- can efficiently implement on 32-bit CPU
  - redefine steps to use 32-bit words
  - can precompute 4 tables of 256-words
  - then each column in each round can be computed using 4 table lookups + 4 XORs
  - at a cost of 4KB to store tables
- Rijndael designers believe this very efficient implementation was a key factor in its selection as the AES cipher

Summary

- have considered:
  - 3DES
  - the AES selection process
  - AES mathematical basis - GF(2^n)
  - the details of Rijndael – the AES cipher
  - looked at the steps in each round
  - the key expansion
  - implementation aspects

Key Terms

Advanced Encryption Standard (AES) avalanche effect field	finite field irreducible polynomial key expansion	National Institute of Standards and Technology (NIST) Rijndael S-box

- **Review Questions**
  - 6.3 What is the difference between Rijndael and AES?
  - Why is Double DES not secure?

Thanks!