Symmetric Crypto: Block Ciphers

2019. 3. 11

Contents

Block Cipher

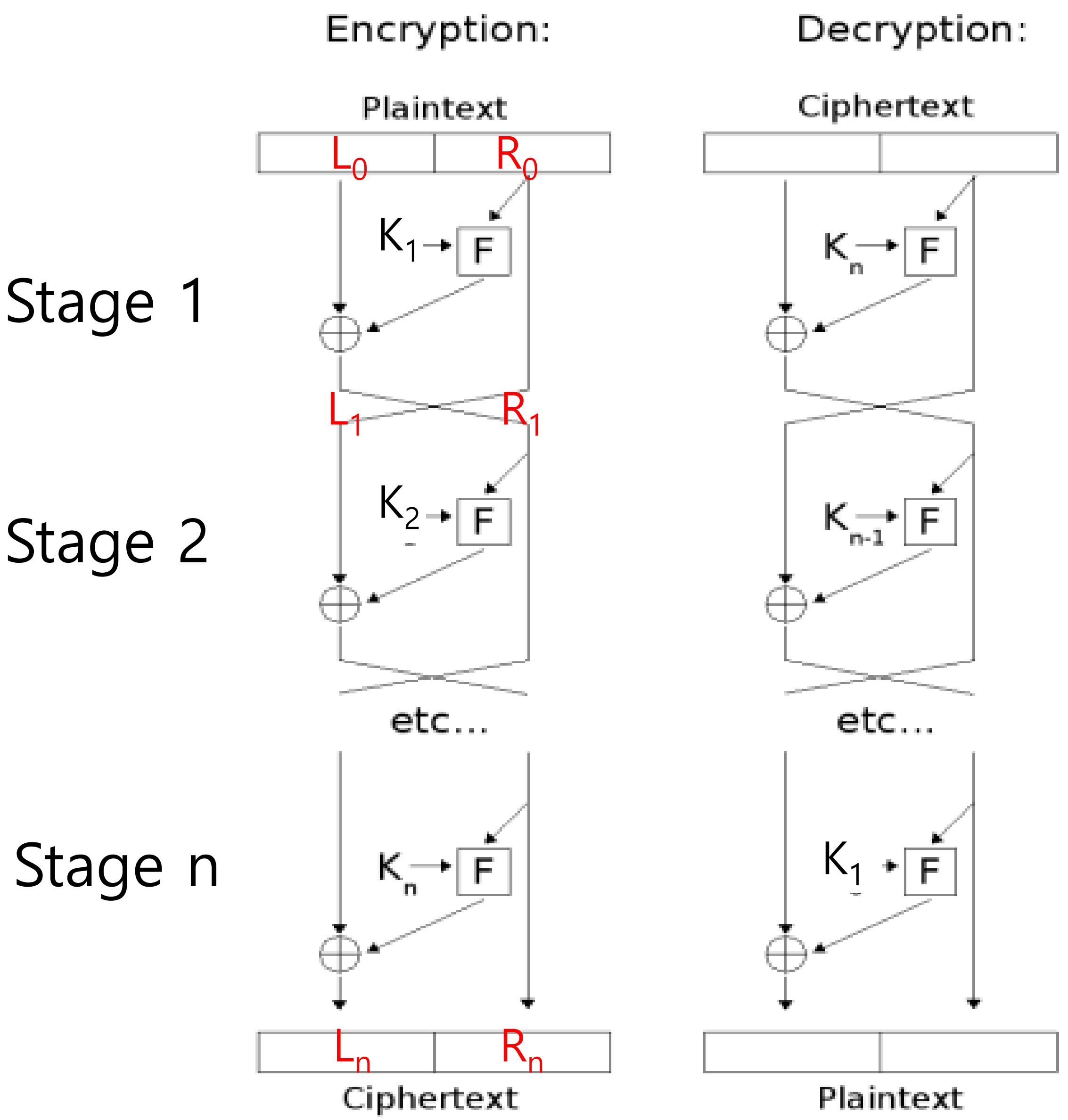
- Plaintext and ciphertext consist of fixed-sized blocks
- Ciphertext obtained from plaintext by iterating a **round function**
- Input to round function consists of ***key*** and ***output*** of previous round

Symmetric key Block Ciphers

- Data Encryption Standard (DES)
  - Adapted in 1973 by NIST
  - 64-bits blocks, 56 bits key
- Triple DES
  - ANSI X9.17 in 1986
  - 168 bits key
- Advanced Encryption Standard (AES)
  - Adapted in 2001 by NIST
  - 128 bits block length, key length 128 bits(192, 256)
- International Data Encryption Algorithm (IDEA)
  - Published in 1991
  - Block size 64bits, key size 128 bits
- RC5
  - In 1994
  - Key size: variable to 2048, block size: 64bits

Feistel Cipher

- It provides a kind of framework for designing block ciphers.
- Split the input block into two parts
  - Input plaintext block = (L_0, R_0)
- At each stage (i=1,2,...,n) do the following computation
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$ where F is round function and K is subkey
- Final ciphertext = (L_n, R_n)



Feistel Cipher

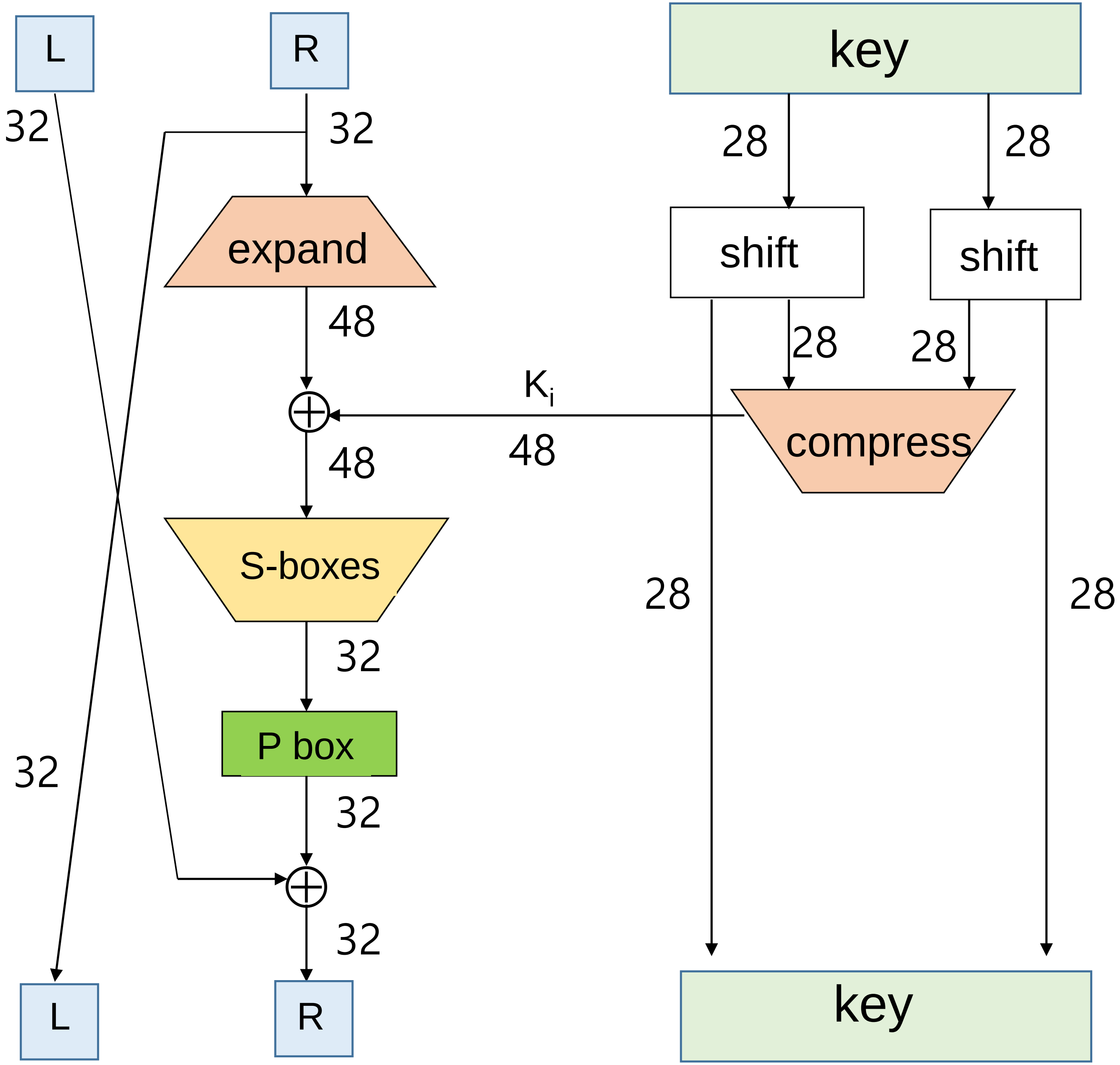
Data Encryption Standard (DES) History

- In 1972, US National Bureau of Standards(now NIST) initiated a request for proposals for a standardized cipher in the USA, which was somewhat a revolutionary act.
- In 1974 NBS received the IBM's Lucifer as a candidate.
  - Based on Feistel cipher, 64 bits of block, 128bit of key
- NSA was secretly involved in the process.
  - It caused controversy and worry since they might plant trapdoor in the cipher.
  - Key length reduced from 128 to 56 bits (by NSA's request)
  - Subtle changes to Lucifer algorithm
- In 1977, DES was published as the U.S. government standard

DES Characteristics

- DES is a Feistel cipher with
  - 64 bit block length
  - 56 bit key length
  - 16 rounds
  - 48 bits of key used each round (subkey)
- Each round is simple (for a block cipher)
- Security depends heavily on "S-boxes"
  - Each S-boxes maps 6 bits to 4 bits

One Round of DES



Expansion Permutation

- Input 32 bits

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

- Output 48 bits

31	0	1	2	3	4	3	4	5	6	7	8
7	8	9	10	11	12	11	12	13	14	15	16
15	16	17	18	19	20	19	20	21	22	23	24
23	24	25	26	27	28	27	28	29	30	31	0

S-box

- 8 “substitution boxes” or S-boxes
- Each S-box maps 6 bits to 4 bits
- The first S-box

input bits (1,2,3,4)

input bits (0,5)

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	0001	1110	1000	1101	0110	0010	1011	1111	1100	1001	0111	0011	1010	0101	0000
11	1111	1100	1000	0010	0100	1001	0001	0111	0101	1011	0011	1110	1010	0000	0110	1101

11

P-box

- Input 32 bits

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

- Output 32 bits

15	6	19	20	28	11	27	16	0	14	22	25	4	17	30	9
1	7	23	13	31	26	2	8	18	12	29	5	21	10	3	24

Subkey(1)

- 56 bit DES key, numbered 0,1,2,…,55

- Left half key bits: LK

49	42	35	28	21	14	7
0	50	43	36	29	22	15
8	1	51	44	37	30	23
16	9	2	52	45	38	31

- Right half key bits: RK

55	48	41	34	27	20	13
6	54	47	40	33	26	19
12	5	53	46	39	32	25
18	11	4	24	17	10	3

Subkey(2)

- For rounds i=1,2,...,16
  - Let LK = (LK circular shift left by r_i)
  - Let RK = (RK circular shift left by r_i)
  - Left half of subkey K_i is of LK bits

13	16	10	23	0	4	2	27	14	5	20	9
22	18	11	3	25	7	15	6	26	19	12	1

  - Right half of subkey K_i is RK bits

12	23	2	8	18	26	1	11	22	16	4	19
15	20	10	27	5	24	17	13	21	7	0	3

Subkey(3)

- For rounds 1, 2, 9 and 16 the shift r_i is 1, and in all other rounds r_i is 2
- Bits 8,17,21,24 of LK omitted each round
- Bits 6,9,14,25 of RK omitted each round
- **Compression permutation** yields 48 bit subkey K_i from 56 bits of LK and RK
- **Key schedule** generates subkey

Trivial things

- An initial permutation before round 1
- Halves are swapped after last round
- A final permutation (inverse of initial perm) applied to (R_{16}, L_{16})
- None of this serves security purpose

DES Security

- Security depends heavily on S-boxes
  - Everything else in DES is linear
- Thirty+ years of intense analysis has revealed no “back door”
- No attacks have been known possible except exhaustive key search.
- It was robust against any mathematical cryptanalysis attack.
- **Inescapable conclusions**
  - Designers of DES knew what they were doing
  - Designers of DES were way ahead of their time

Destiny of DES

- For over 30 years DES had been challenged for its security.
- In 1998, the EFF(Electronic Frontier Foundation) built the computer, Deep Crack, which did brute-force attack against DES in 56 hours and was built for less than $250,000.
- A key size of 56 bits is too short to encrypt text, so it is no more useful for confidential data.

Triple DES

- Today, 56 bit DES key is too small
  - Exhaustive key search is feasible
- But DES is everywhere, so what to do?
- **Triple DES** or **3DES** (112 bit key)
  - $C = E(D(E(P,K_1),K_2),K_1)$
  - $P = D(E(D(C,K_1),K_2),K_1)$
- Why Encrypt-Decrypt-Encrypt with 2 keys?
  - Backward compatible: E(D(E(P,K),K),K) = E(P,K)
  - And 112 bits is enough

Advanced Encryption Standard (AES)

AES History

- In 1999, NIST recommended to use 3DES, but it had drawbacks:
  - Not efficient with software implementation. DES S/W was common, then 3DES made it 3 times slower.
  - Block size of 64 bits was too small.
  - They were worried about future quantum computers.
- In 1997, NIST called for new proposals for a new Advanced Encryption Standard (AES).
  - Unlike DES, the whole process was open.
  - NSA openly involved

AES

- The requirements for AES candidates
  - Block cipher with 128 bits block size
  - 3 key lengths must be supported: 128, 192, and 256 bits
  - Security relative to other submitted algorithm
  - Efficiency in software and hardware
- In 2001, NIST declared the Rijndael(pronounced like “Rain Doll” or “Rhine Doll”) as the new AES and published it as the standard.
- Iterative stages (like DES)
- Not a Feistel cipher (unlike DES)

AES Characteristics(1)

- **Block size:** 128 bits (128, 192, 256 bits in Rijndael)
- **Key length:** 128, 192 or 256 bits (independent of block size)
- Variable rounds (depends on key length)
  - 10 if K = 128 bits
  - 12 if K = 192 bits
  - 14 if K = 256 bits
- Each round uses 128 bits round key.
  - Nr+1 round keys for Nr rounds

AES Characteristics(2)

- State: 4X4 array of bytes = 16 bytes = 128 bits
- Each round uses 4 functions (3 “layers”)
  - ByteSub (nonlinear layer)
  - ShiftRow (linear mixing layer)
  - MixColumn (nonlinear layer)
  - AddRoundKey (key addition layer)

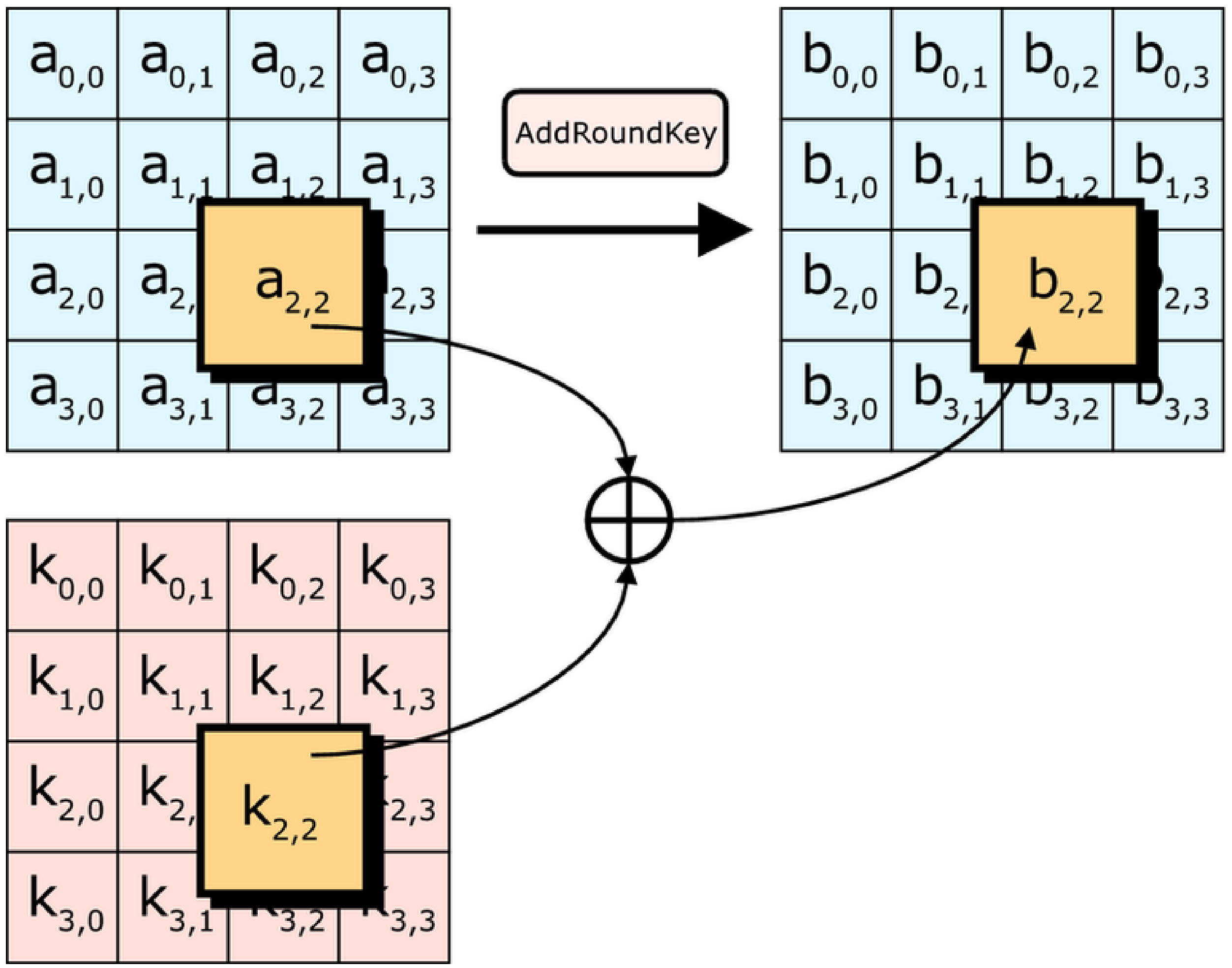
- Permutation
  - ShiftRow
- Substitution
  - ByteSub (State, S-box)
  - MixColumn (State)
  - AddRoundKey (State, KeyNr)

AES High-level description

```
State = X
AddRoundKey(State, Key0)                    (op1)
for r = 1 to Nr - 1
        SubBytes(State, S-box)              (op2)
        ShiftRows(State)                    (op3)
        MixColumns(State)                   (op4)
        AddRoundKey(State, KeyNr)
endfor
SubBytes(State, S-box)
ShiftRows(State)
AddRoundKey(State, KeyNr)
Y = State
```

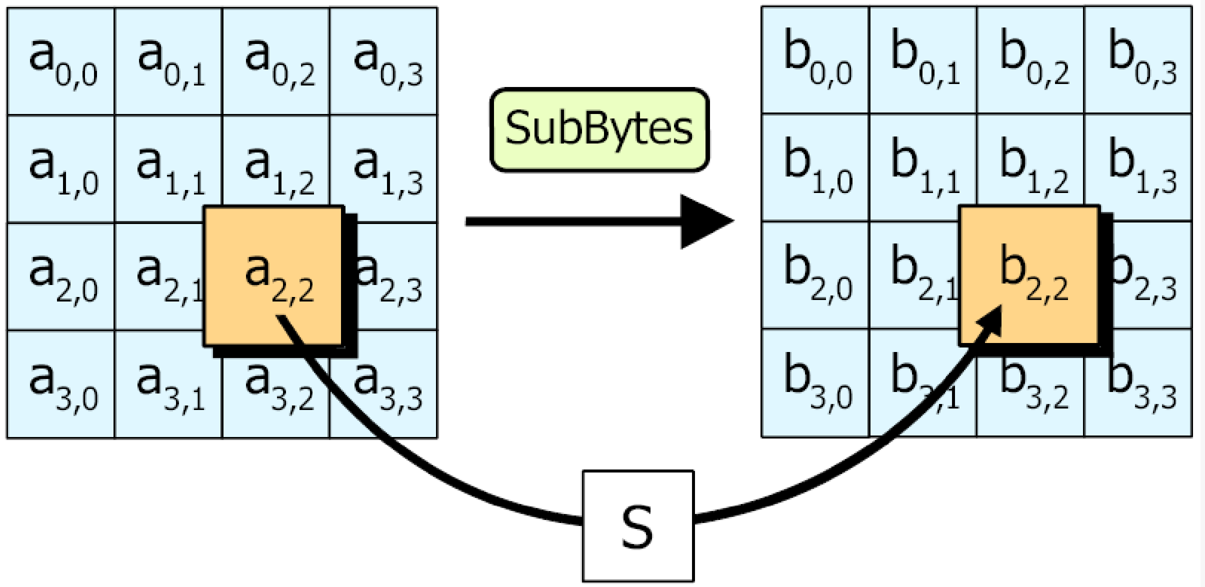
AddRoundKey

- XOR subkey and block
- Subkey(round key) is determined by the key schedule algorithm.



ByteSub

- Treat 128 bit block as 4x4 byte array



- ByteSub is AES's "S-box"
  - Byte substitution
- Can be viewed as nonlinear (but invertible) composition of two math operations

AES “S-box”

ex, byte $53_h \rightarrow ed_h$

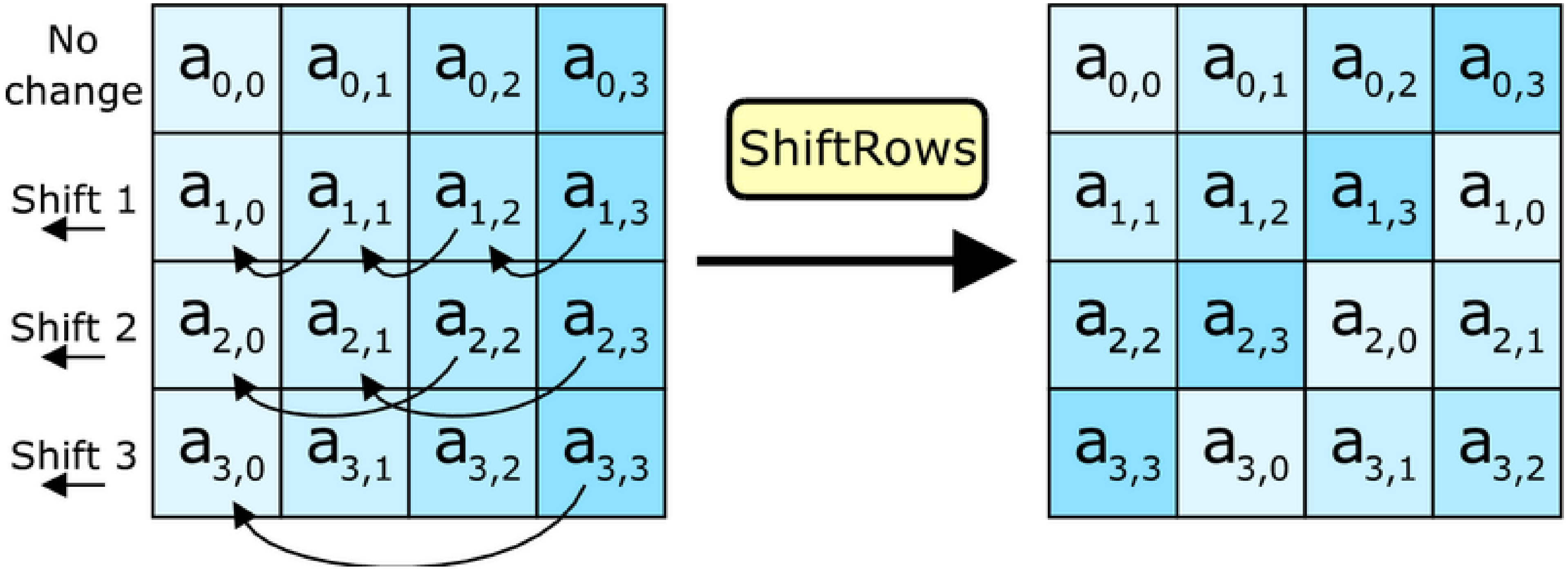
Last 4 bits of input byte

First 4 bits of Input byte

	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

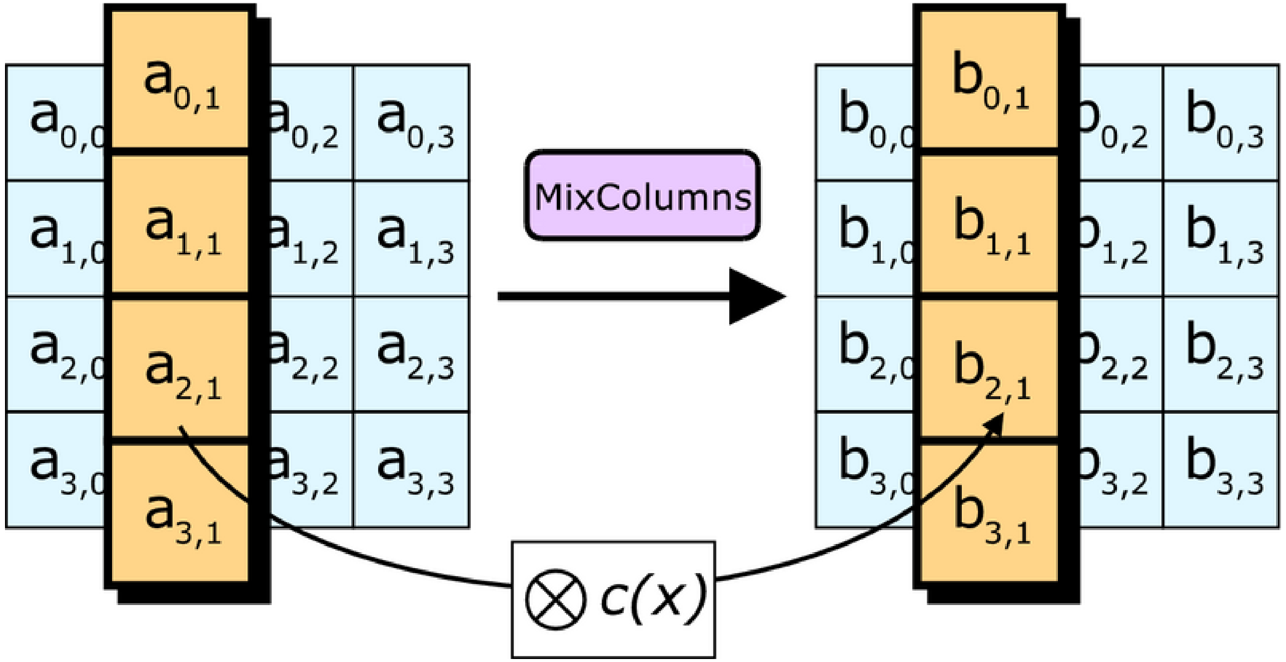
ShiftRow

- Cyclic shift rows



MixColumn

- Invertible, linear operation applied to each column



- Implemented as a (big) lookup table

Decryption

- To decrypt, process must be invertible
- Inverse of MixAddRoundKey is easy, since “$\oplus$” is its own inverse
- MixColumn is invertible (inverse is also implemented as a lookup table)
- Inverse of ShiftRow is easy (cyclic shift the other direction)
- ByteSub is invertible (inverse is also implemented as a lookup table)