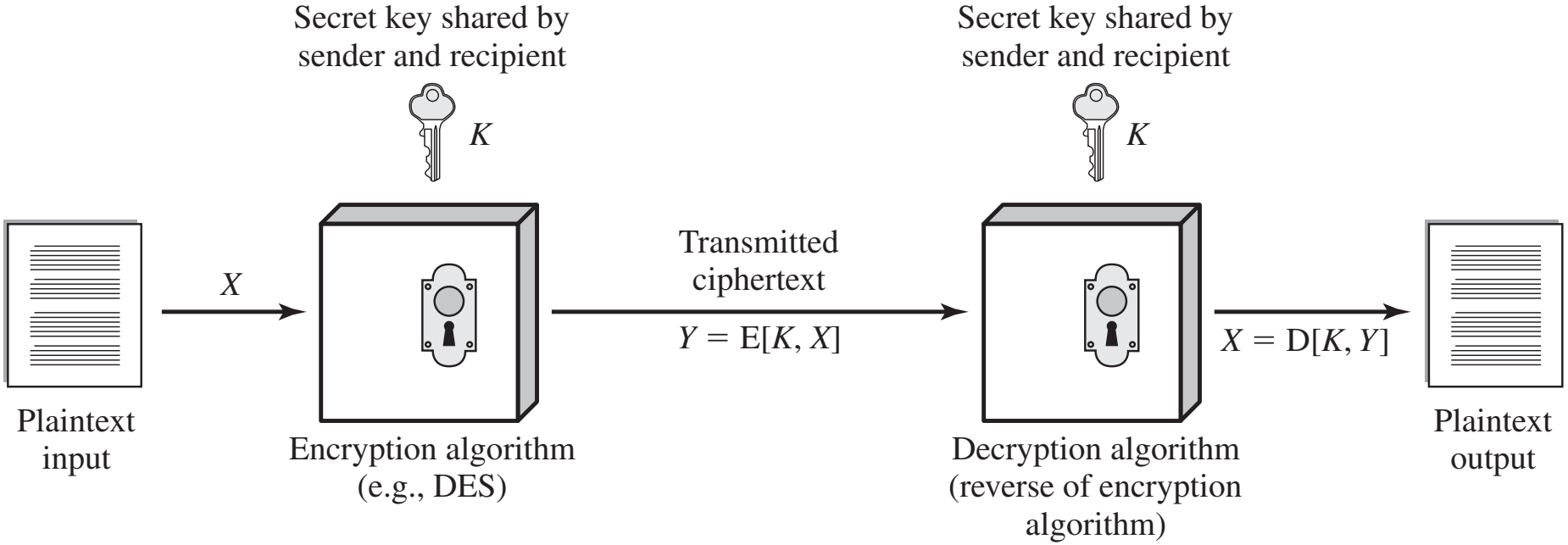
Symmetric Key Encryption

Kehuan Zhang
© All Rights Reserved

IERG4130 2022

Review: Four Elements in Crypto-System

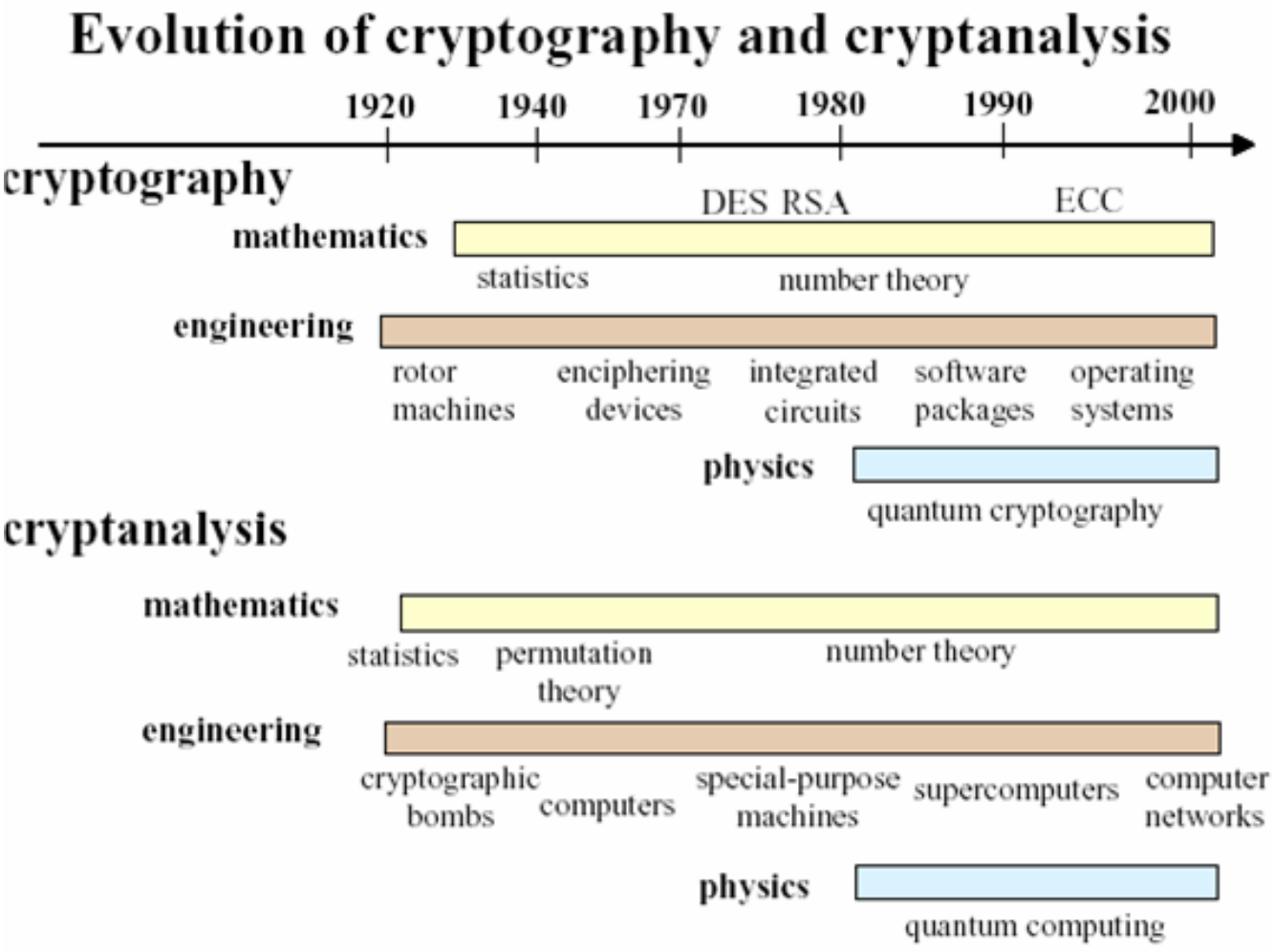
- Plaintext
- Ciphertext
- Encryption Algorithm
- Encryption Key
- Encryption: operation to transform plaintext into ciphertext
- Decryption: operation to convert ciphertext back to plaintext



Security of Encryption Algorithms

- Crypto-analysis: try to decode the ciphertext without knowing the encryption key
  - Naive and baseline: Brute-force attack
  - Often try to identify weakness in an encryption algorithm
    - ★ Compare with brute-force to measure effectiveness of crypto-analysis
- Common Type of Attacks - with increasing knowledge and power
  - Cipher text only - most challenging
  - Known Plaintext - attackers have some plaintext cipher text pairs
  - Chosen Plaintext - attackers can get the ciphertext of some plaintext generated by them
- According to open design principle, the encryption algorithm will always be known
- Three Laws of Security
  - Absolutely secure systems do not exist
  - To halve your vulnerability, you have to double your expenditure
  - Cryptography is typically bypassed, not penetrated

Evolution of cryptography and cryptanalysis



- New area: post-quantum crypto → but it is difficult
  - `https://thequantuminsider.com/2022/08/05/nist-approved-post-quantum-safe-algorithm-cracked-in-an-ho`

Classical Cryptography Techniques

Basic Techniques in Conventional Ciphers

- Permutation (transposition): rearrange bits (or bytes) in the plaintext
- Substitution: replace bits (or bytes, blocks) with new values
- Question: why do we need to learn classical cryptography algorithms as they are not used any more?
  - To understand the basic concepts
  - To understand the idea of designing an encryption algorithm
  - To understand the idea of crypto-analysis

Example of Permutation: "Rail-Fence" Cipher

```
plaintext:  DISGRANTLED EMPLOYEE
                     |
                     V
               D R L E O
permutation:   IGUTEMLYE
               S N D P E
                     |
                     V
ciphertext: DRLEOIGUTEMLYESNDPE
```

- Given the ciphertext: IESKLKTIBOIHO, what is the plaintext?

```
I E S K
LKTIBO
I H O
```

Example of Substitution: Caesar Cipher

- Mapping every input character to another one

```
Plaintext Character:      ABCDEFGHIJKLMNOPQRSTUVWXYZ
To be substituted (Key): DEFGHIJKLMNOPQRSTUVWXYZABC
```

- In mathematic form
  - Encryption:

$$c = E(k, p) = (p + k) mod 26$$

  - Decryption:

$$p = E(k, c) = (c - k) mod 26$$

  - It is a kind simple "Monoalphabetic Cipher" (shifting only)
    - ★ The substitution mapping are fixed for the whole message
- An example:
  - Plain text: `SEE ME AFTER THIS CLASS`
  - Ciphertext: `VHH PH DIWHU WKLV FODVV`
- How secure is it?
  - Using the metrics of brute force analysis complexity
  - There are only 25 possibilities to try
    - ★ Too easy to enumerate all of them
- Question: how can you improve this algorithm?

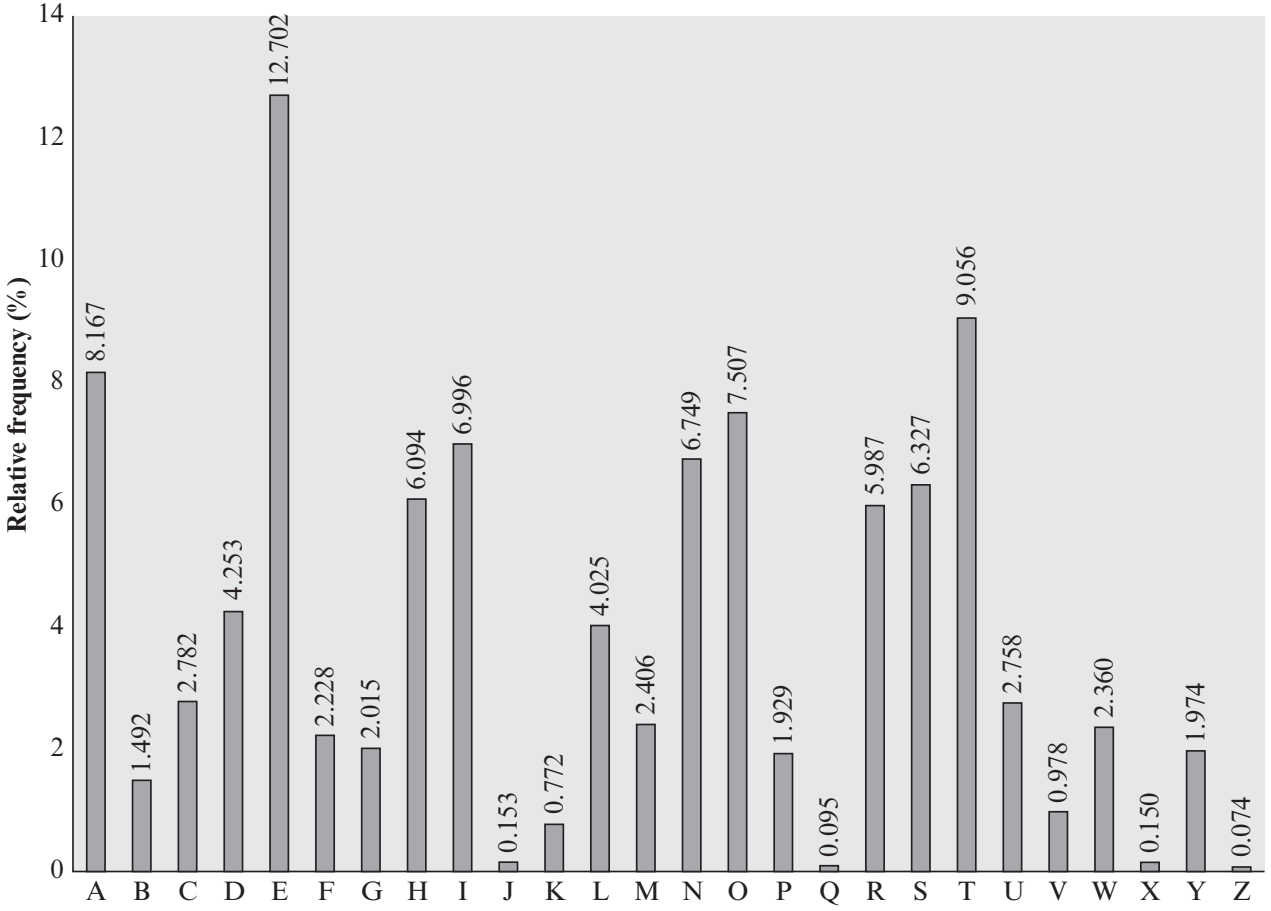
A more complex Monoalphabetic Cipher

- Not shifting, but every mapping is random. For rexample:

```
Plaintext Character:      ABCDEFGHIJKLMNOPQRSTUVWXYZ
To be substituted (Key): DWEOGUIJKLPNZEQRSTBVFXYDABC
```

- What is the complexity for brute-force attack to it?
  - 26!-1
- Is it secure enough?
  - Vulnerable to frequency analysis

Frequency based attack



- Attackers can calculate letter frequency for cipher text
  - Monoalphabetic cipher does not change frequency
  - It is still fixed and one-to-one substitution

Polyalphabetic substitution cipher

- A set of monoalphabetic substituted rules will be used
  - A particular substituted rule will be chosen according to encryption key
- An Example: Vigenere Cipher

```
Plain text :  D   I   G
Key        :  R   E   D
Cipher text:  U   M   J
```

Plain	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	x
1	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y
2	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
3	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
4	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
5	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
6	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
7	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
8	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
9	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
10	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
...																								

- Key is “RED”, Plaintext is “DIG”, Ciphertext is “UMJ”

Vigenere Cipher

- The formula for encryption and decryption (m is the key length)

$$c_i = E(p_i, k_{(i \ mod \ m)}) mod 26$$

$$p_i = E(c_i, k_{(i \ mod \ m)}) mod 26$$

- What is the complexity for a brute-force attack? $O(26^k)$ if key length k is known, otherwise $O(\frac{26(26^n-1)}{26-1})$ where n is max key length
- Can it be attacked?
  - Hints: decompose it into monoalphabetic ciphers
  - But how? It is critical to estimate key length
    - ★ Use Index of Coincidence (IC) (i.e., probability of AA, BB, ... ZZ)
    - ★ The substitution is just a uniform mapping of the original characters
    - ★ It does not affect the index of coincidence (0.067 for English)

From ciphertext calculate

$$IC = \frac{\sum_{i=1}^{c} n_i(n_i - 1)}{\frac{N(N-1)}{c}}$$

where c is the number of letter space (e.g., 26), n_i is the number of appearance of letter i in ciphertext, N is the total length of ciphertext

Ideal Cipher: one-time pad

- Rational behind:
  - Vigenere cipher has used more complexed "key" for encryption
  - The weakness is the repeated patterns due to the limited key length
  - What if the key has the same length with the plaintext?
  - To further avoid repeatings, every key can only be used one time
- So that is One-time pad (OTP) cipher
  - The Key has the same length as the plaintext
  - The Key will be used only once
  - Encryption:

$$C[i] = (P[i] + KEY[i]) mod 26$$

  - Decryption:

$$P[i] = (C[i] - KEY[i]) mod 26$$

The Security of One-Time Pad

- It can provide perfect secrecy
  - If Key is unknown, attackers cannot decrypt the ciphertext
  - The brute-force complexity will increase as plaintext (thus the key) becoming longer
- But it is hard to implement
  - Key has the same length of plaintext and never to be reused
  - The problem of securely transmitting the key actually is equivalent to securely transmitting plaintext
- One practical solution is Vernam Cipher
  - With minor security trade-offs in order to make it practical
    - Use PRNG to generate key sequence
    - Use XOR to replace Add/Minus (to make encryption/decryption really symmetric)
- So problem is reduced to the security of Key generator
  - How to generate a long KEY?
    - Use Pseudo Random Number Generate (PRNG)
    - Remember to use **cryptographically secure pseudo-random number generator (CSPRNG)**

Vernam Cipher

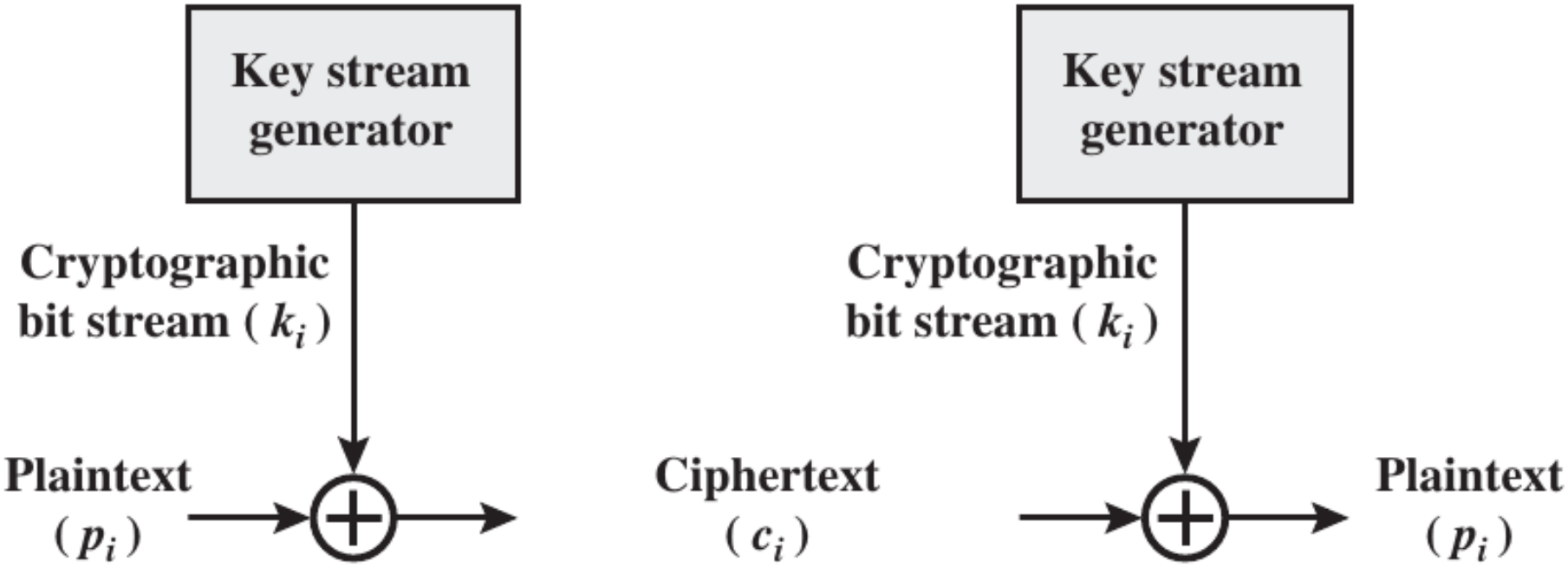
- Encryption:

$$c_i = p_i \oplus k_i$$

- Decryption:

$$p_i = c_i \oplus k_i$$

Where $\oplus$ is XOR operation



Why use XOR?

- XOR function is its own inverse
  - Make it perfect for encryption and decryption
- Given encryption:

$$c_i = p_i \oplus k_i$$

- So the decryption is:

$$c_i \oplus k_i = (p_i \oplus k_i) \oplus k_i = p_i \oplus (k_i \oplus k_i) = p_i \oplus 0 = p_i$$

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

A	A	A⊕A
0	0	0
1	1	0

A	0	A⊕0
0	0	0
1	0	1

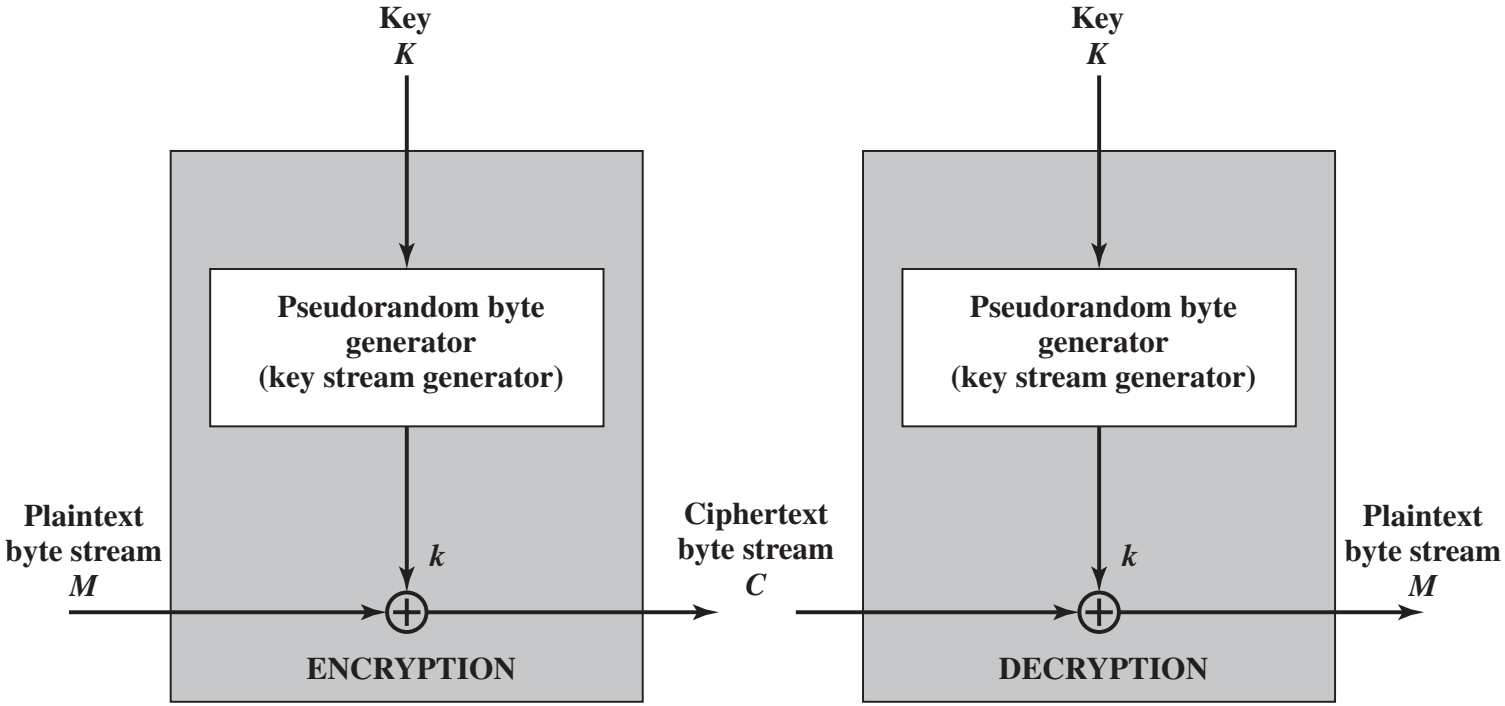
Modern Cryptographic Algorithms

Modern Cipher

- Modern Ciphers have following three features
  - Operates on binary bit sequence (thanks to XOR)
    - ★ Classic ones works on letter and digits directly
  - The encryption algorithm is known to public
    - ★ Classic ones is mainly based on "security by obscurity"
  - The security comes from the secrecy of encryption key
    - ★ Classic ones require secrecy of algorithm
- Types of Modern Cipher
  - Symmetric Encryption vs. Asymmetric Encryptionc
  - Stream Cipher vs. Block Cipher

Stream Cipher

- Normally processing message byte by byte
  - E.g., Vernam Cipher is a stream cipher
- Typically has a PRNG to generate a key stream
- Use XOR as the basic operation to combine key with plaintext (or key with ciphertext)
- The whole security is built on the security of PRNG
- Normally would be simpler and faster



A Real-World Stream Cipher - RC4 (Ron's Code #4)

- It was a proprietary cipher (trade-secret) owned by RSA
  - But was leaked to public domain in 1994 (anonymously)
- Variable key size (1 to 256 bytes), and byte-oriented stream cipher
- Was widely used in practice (Web SSL/TLS, Wireless LAN WEP)
- Key forms random permutation of all 8-bit values
- Uses that Permutations to scramble input byte-by-byte to get ciphertext
- But it started to retire due to security loops in 2013
  - Major browsers and Web sites started to disable RC4
  - For leisure reading only (will not appear in exam):
    - ★ "Attacking SSL when using RC4", published on BlackHat Asia, 2015
    - ★ "Weakness in the Key Scheduling Algorithm of RC4", published in 2001

RC4 Key Schedule

- Started with an array S of numbers: 0..255
- Use key to well and truly shuffle array S
- S forms the internal state of the cipher Given a key **k** of length **L** bytes:

```
for i = 0 to 255 do
    S[i] = i
j = 0
for i = 0 to 255 do
    j = (j + S[i] + k[i mod L]) (mod 256)
    swap (S[i ], S[j])
```

RC4 Encryption

- Encryption continues shuffling array values
- Sum of shuffled pair selects "stream key" value
- XOR with next byte of message to en/decrypt

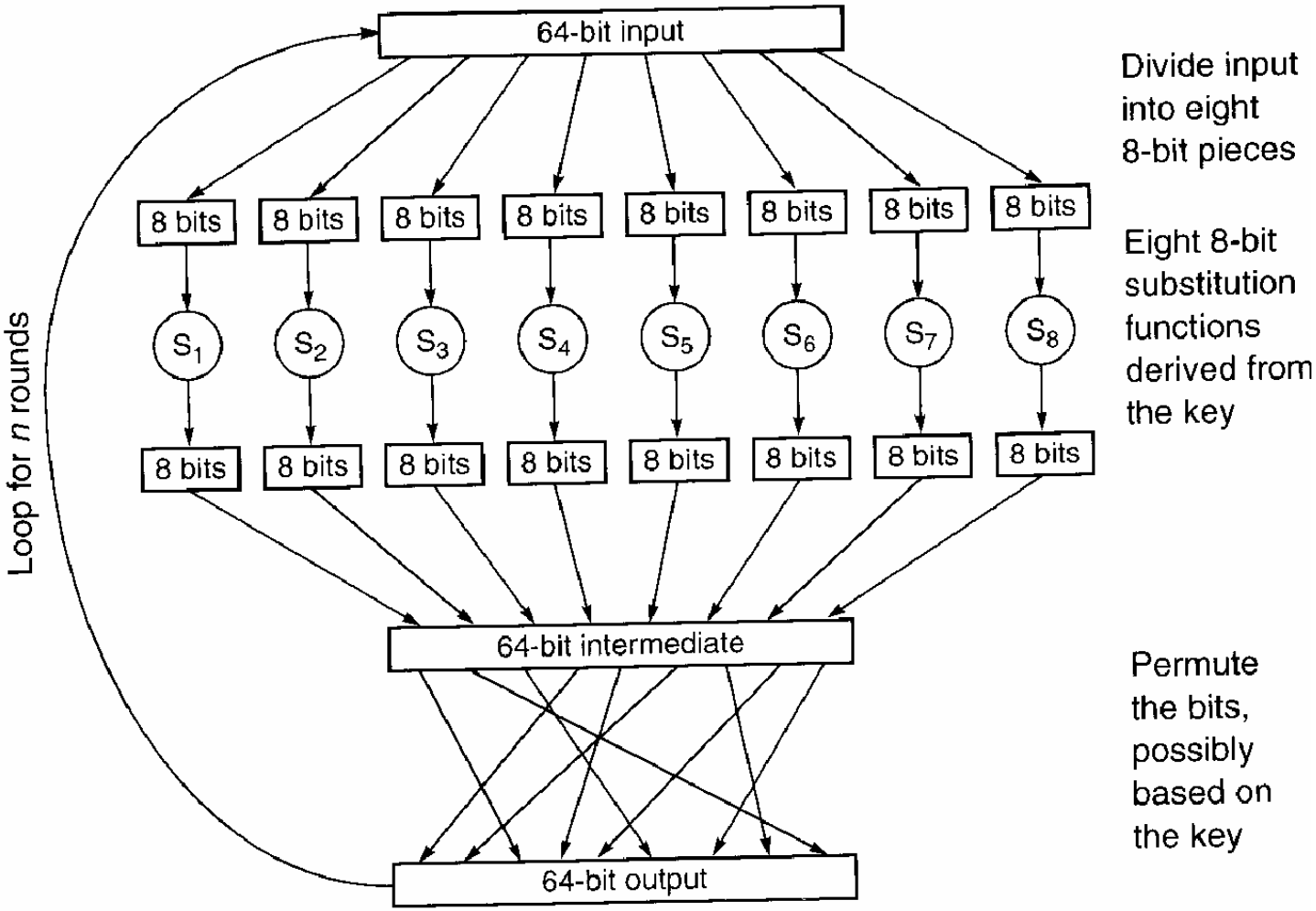
```
i = j = 0
for each message byte M[i]
    i = (i + 1) (mod 256)
    j = (j + S[i]) (mod 256)
    swap(S[i], S[j])
    t = (S[i] + S[j]) (mod 256)
    C[i] = M[i] XOR S[t]
```

Block Cipher

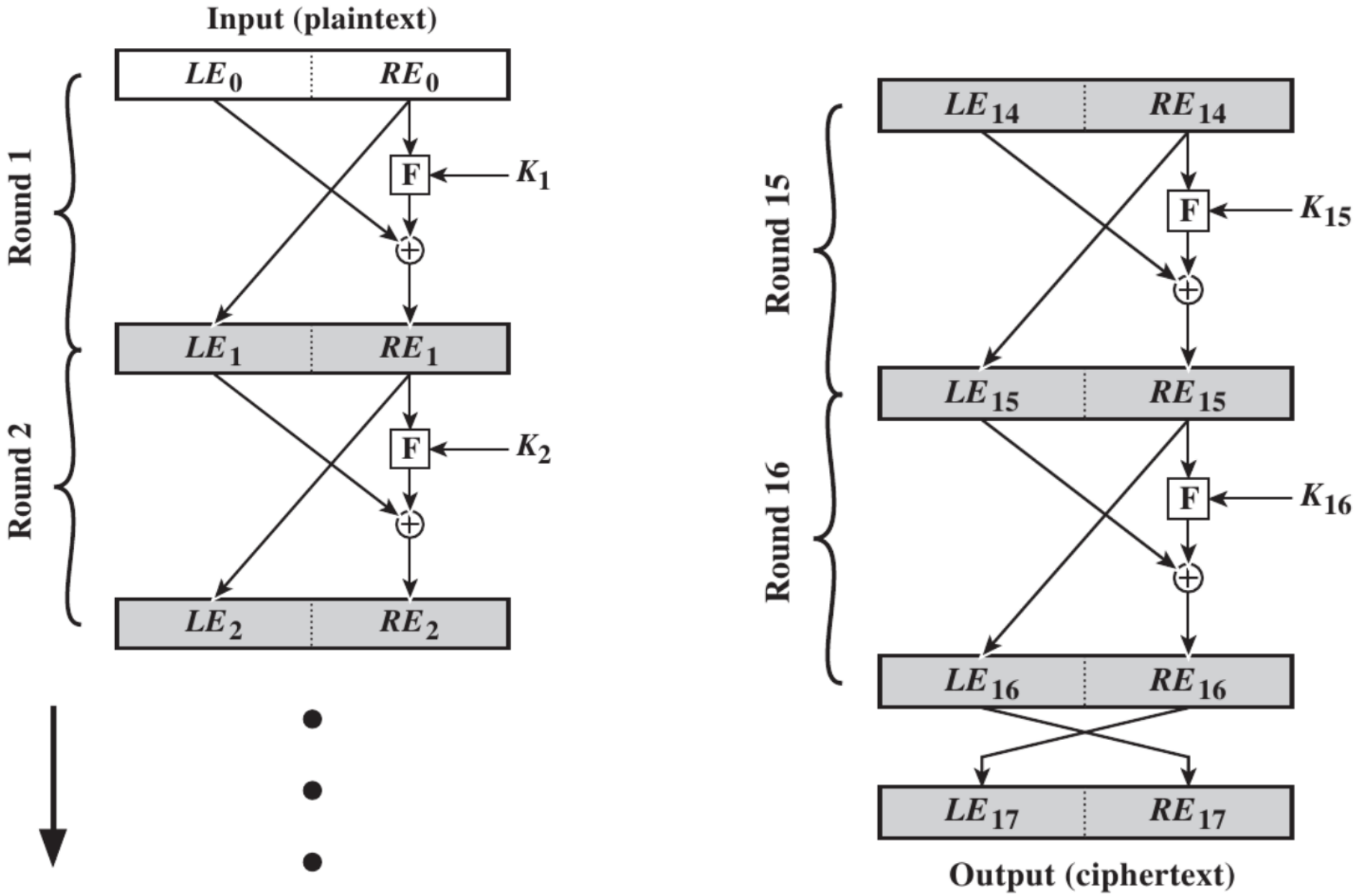
- Processing message block by block (with fixed block size)
  - Typical example: DES (Data Encryption Standard) / AES (Advanced Encryption Standard)
- Each block will go through multipel rounds of Permutations and Substitutions
- Multiple operations, like rotation, shifting
- Key dependent substitution (S-Boxes)
- Has dedicated key scheduling process (to generate keys for each round)
- Given a stream cipher, why do we still need a block cipher?
  - Weakness of Stream Cipher?
    - ★ Its security relies on the PRNG
    - ★ No protection on integrity
  - But, stream cipher has its own advantages
    - ★ Fast
    - ★ Can operate on dynamic data (e.g., data length is unknown)
    - ★ No need padding (which may introduce security loopholes)

Typical Block Cipher Structure

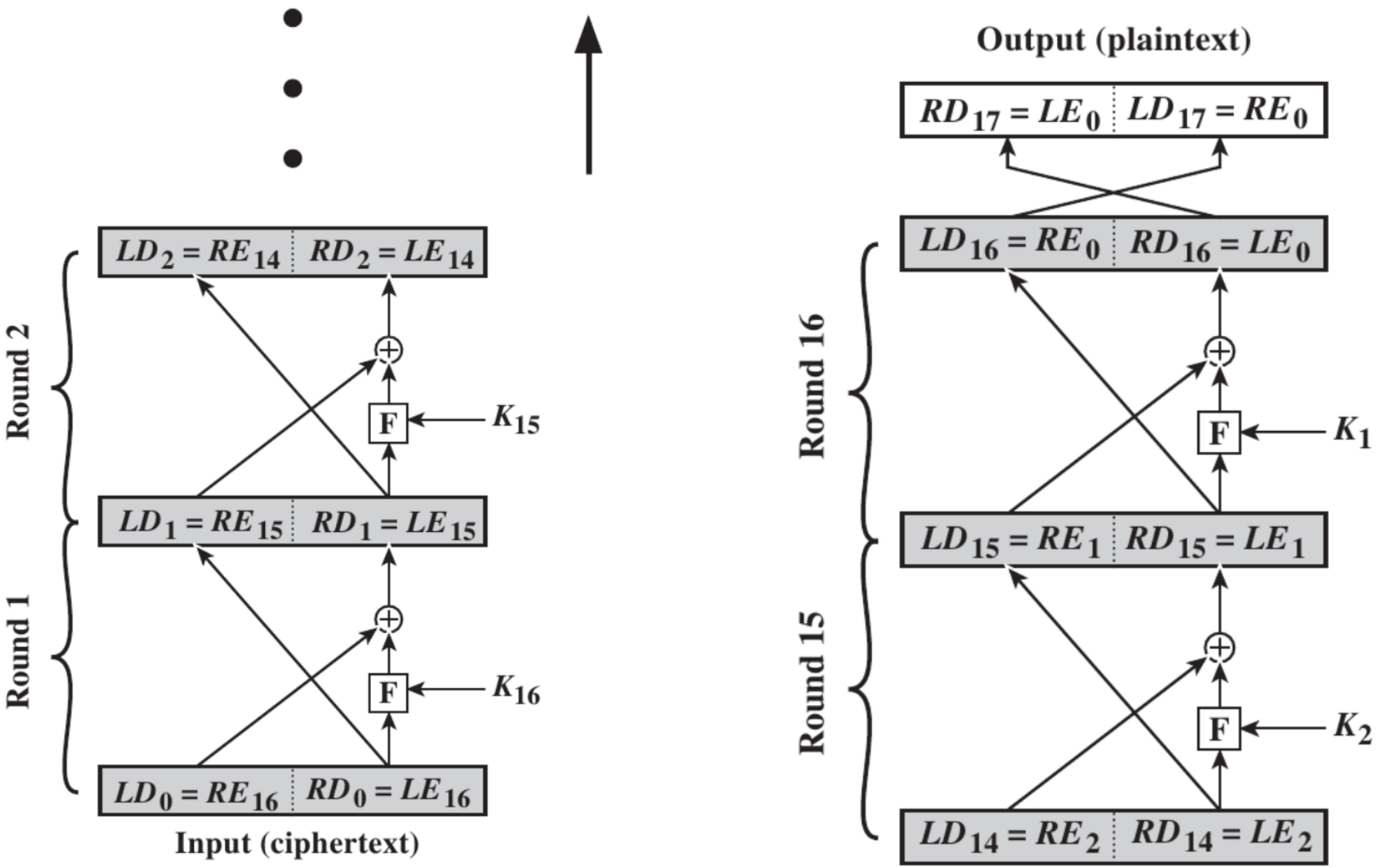
- Block Cipher generally have many rounds
  - Each round will take the output of previous rounds as input
  - also has key scheduling algorithm to generate different keys for each round



The Building Block - Feistel Network



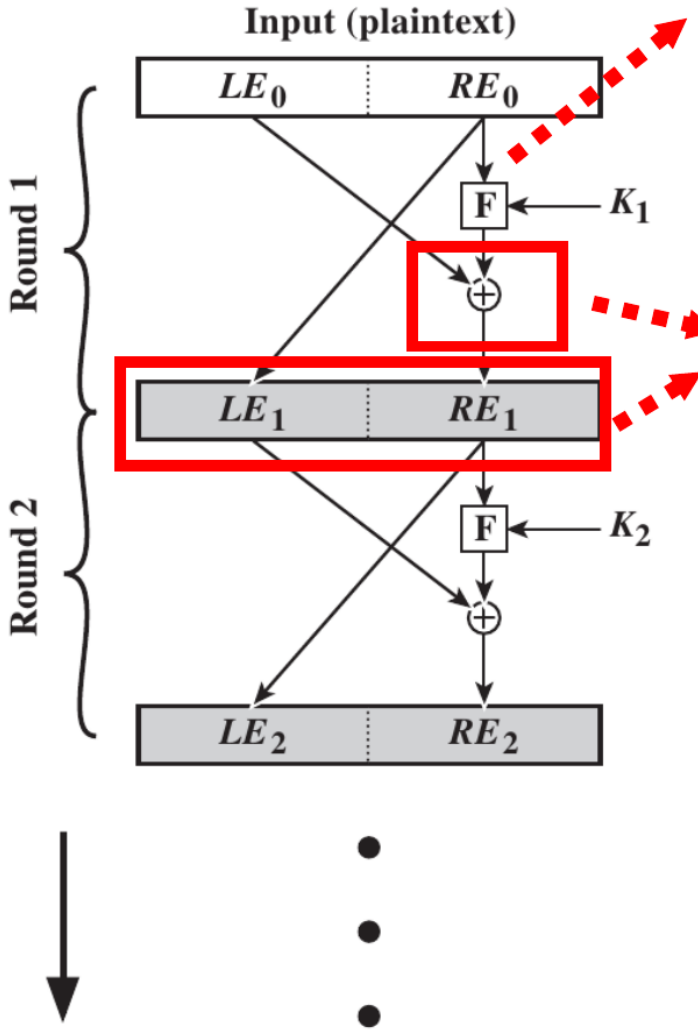
The Feistel Network for Decryption

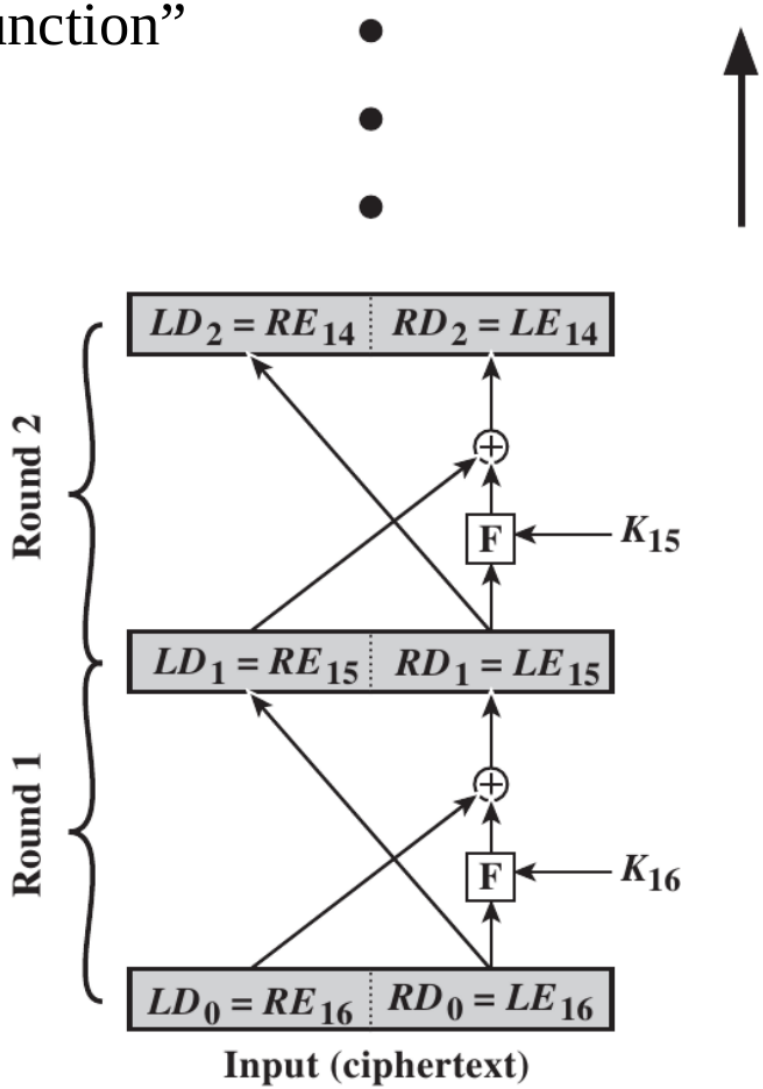


The Feistel Network: Encryption vs. Decryption

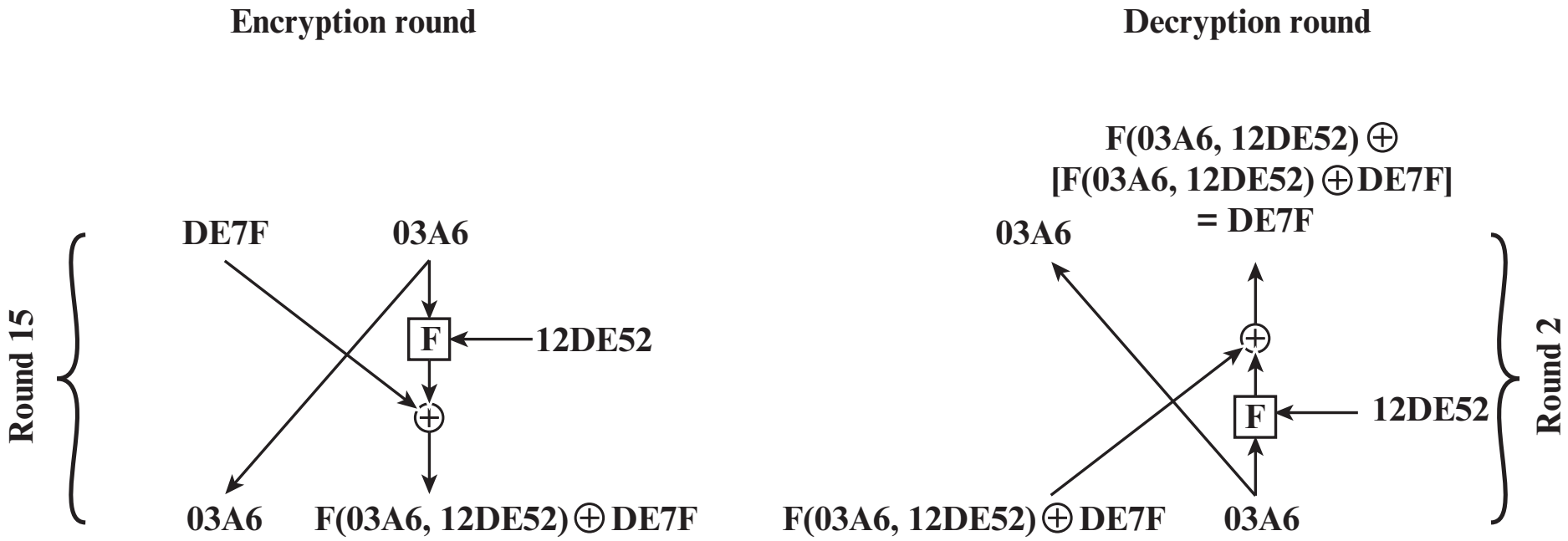
F is “round function”

What operation here?

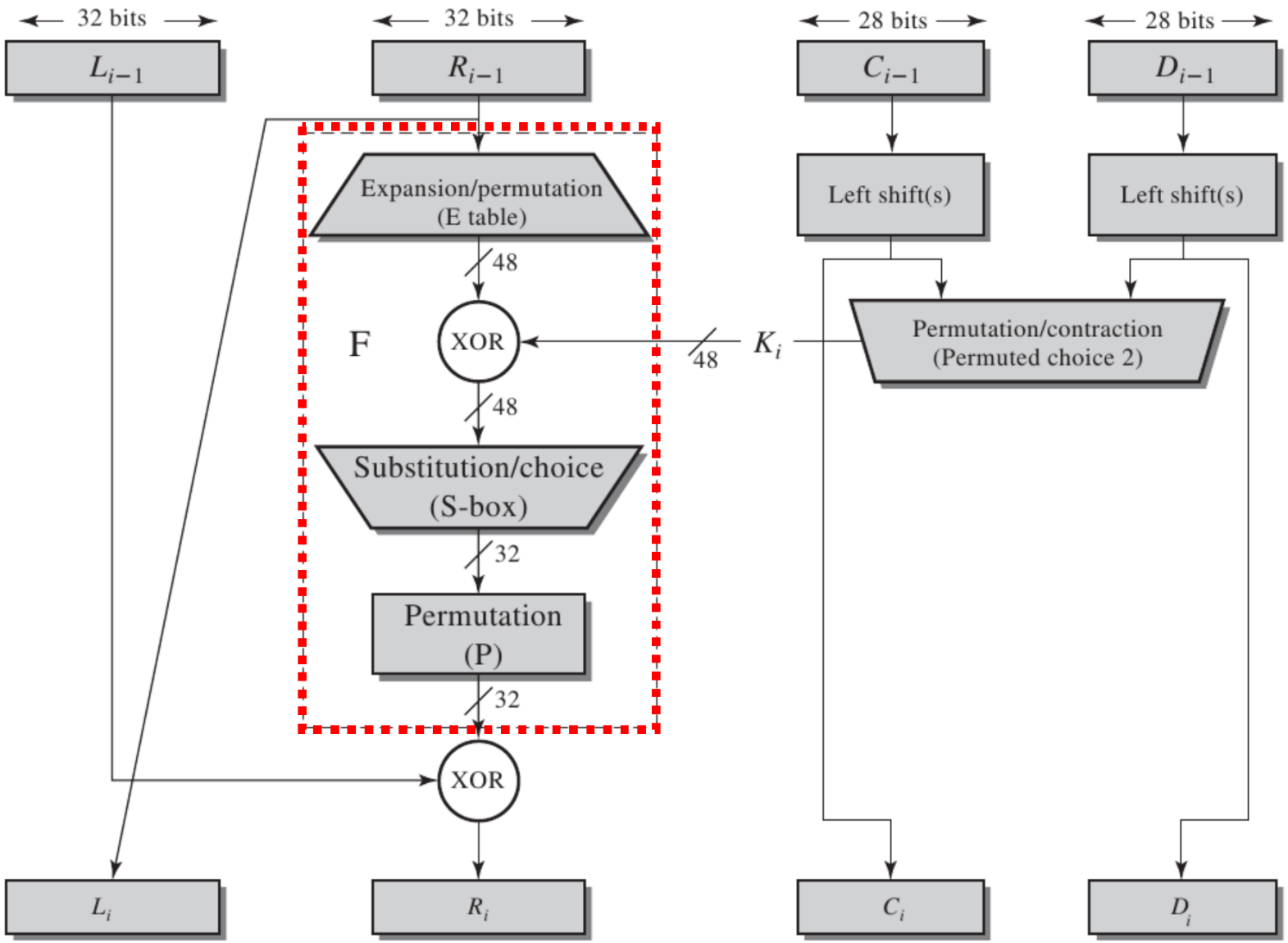




An Example

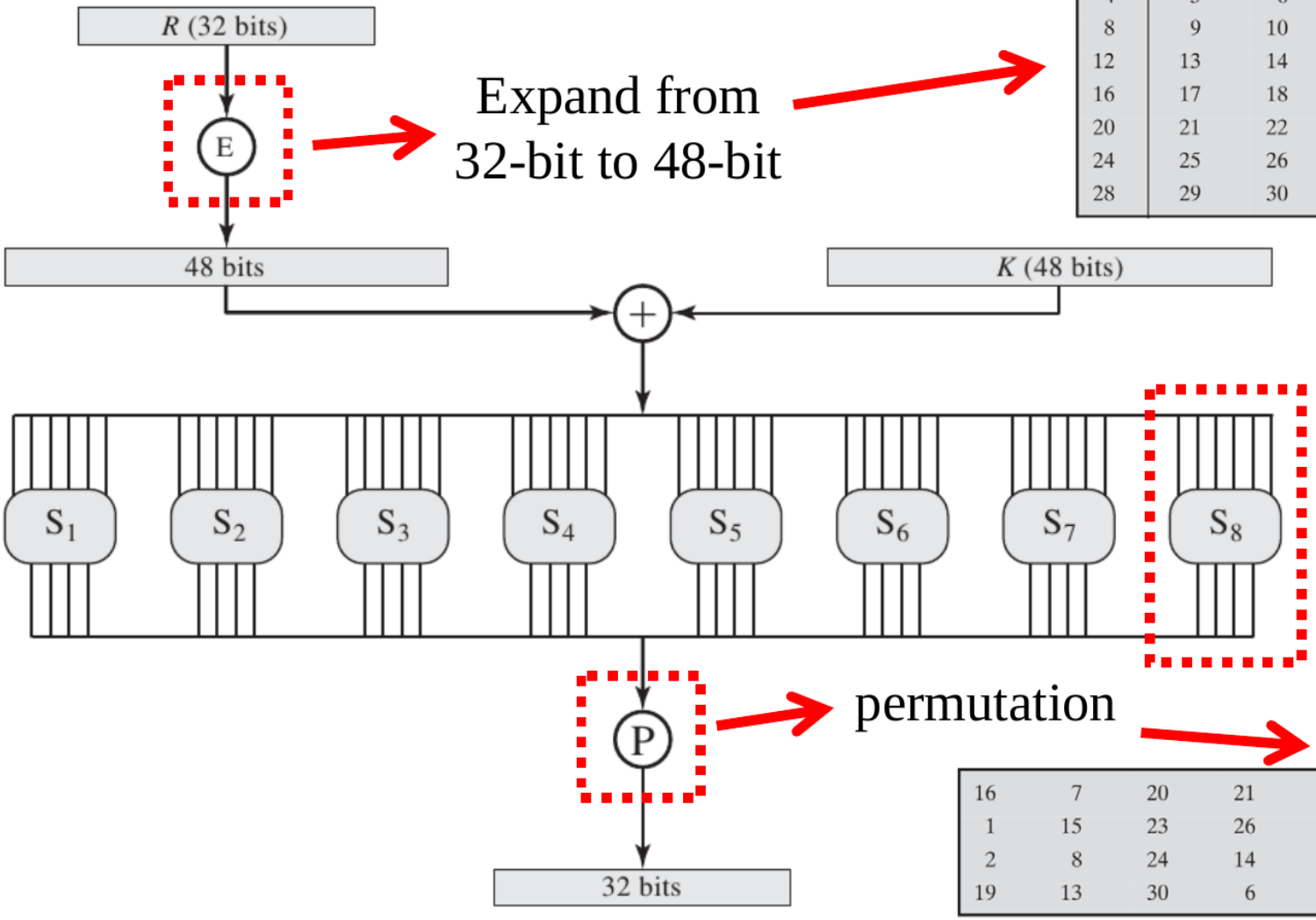

Encryption round
Decryption round
Round 15
DE7F
03A6
F
12DE52
03A6
F(03A6, 12DE52) ⊕ DE7F
F(03A6, 12DE52) ⊕
[F(03A6, 12DE52) ⊕ DE7F]
= DE7F
03A6
F
12DE52
F(03A6, 12DE52) ⊕ DE7F
03A6
Round 2

Details of Round Function F & Key Updates



More Details about F

What's *E/P/S1…S8?*



Expand from 32-bit to 48-bit

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Predefined substitution box

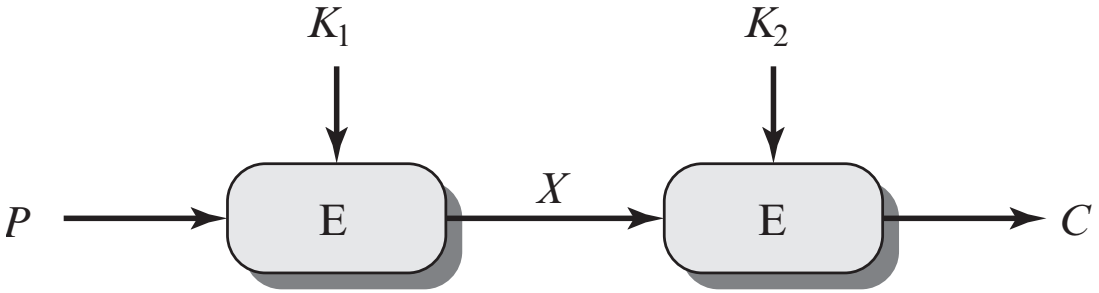
permutation

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

How Secure is DES?

- It is insecure: key is too short under today's situation
  - DES key is only 56 bits (which is sufficient in 1976 when DES was designed)
  - A DES Key can be cracked within 4.5 days in 1998
- Other Issues behind DES
  - NSA of United States had got involved into the design of DES
    - E.g., S-boxes were all chosen by NSA but no explanations on the rational behind
  - People worried that NSA might already know vulnerabilities of DES
    - Or even put a "backdoor" within the algorithm
- Extension of DES: perform DES for multiple round
  - 2DES?
  - 3DES?

About 2DES



- Encryption:

$$C = E_{k_2}(E_{k_1}(P))$$

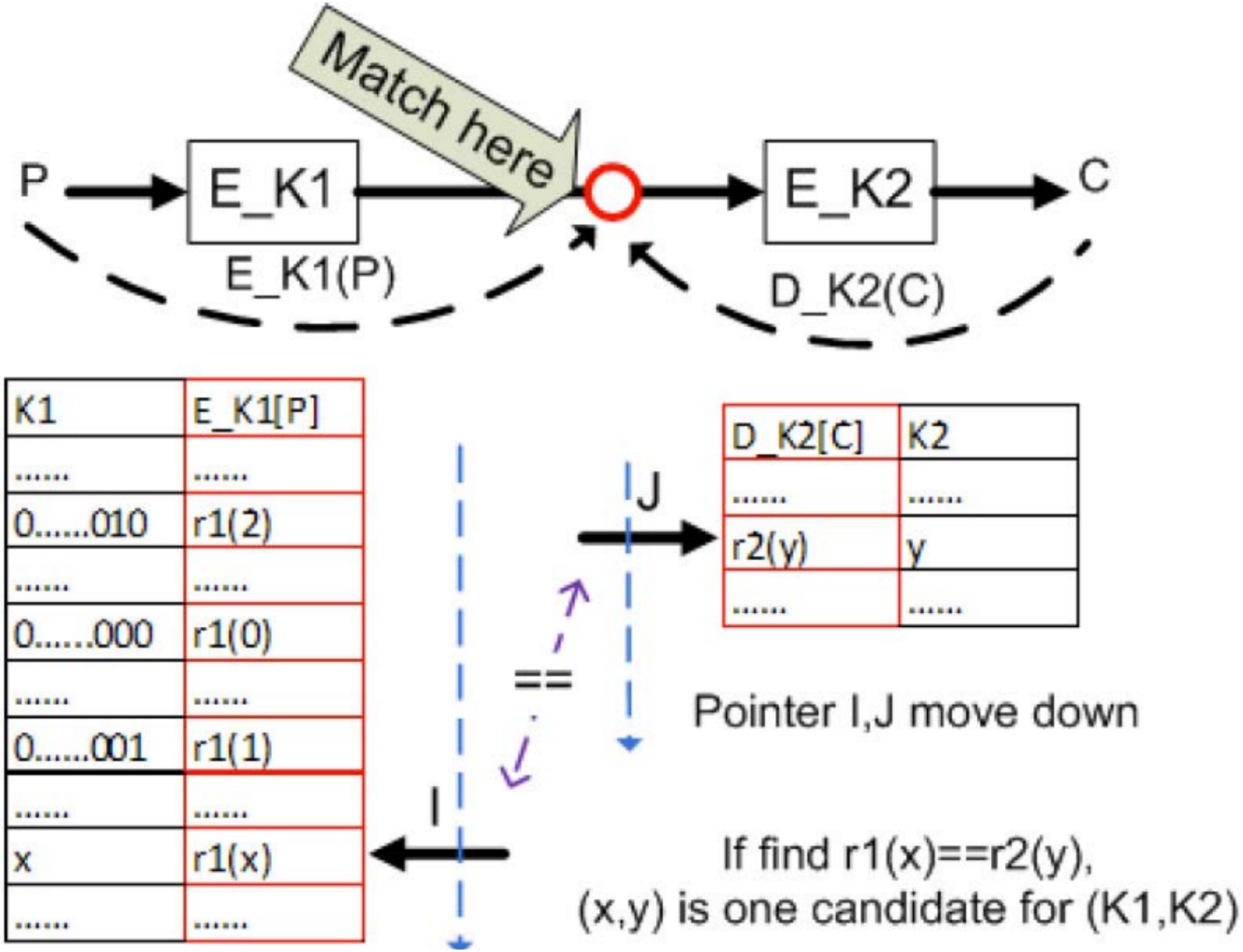
- Decryption:

$$P = D_{k_1}(D_{k_2}(C))$$

- Is 2DES secure? NO, it is vulnerable to Meet-in-the-Middle Attack
- Cryptoanalysis
  - Assume attack was able to get a few <plaintext, ciphertext> pairs
    - like: <p1, c1>, . . . , <pi, ci>
  - An important Observation:

$$D_{k_2}[c_i] = D_{k_2}(E_{k_2}(E_{k_1}(p_i))) = E_{k_1}(p_i)$$

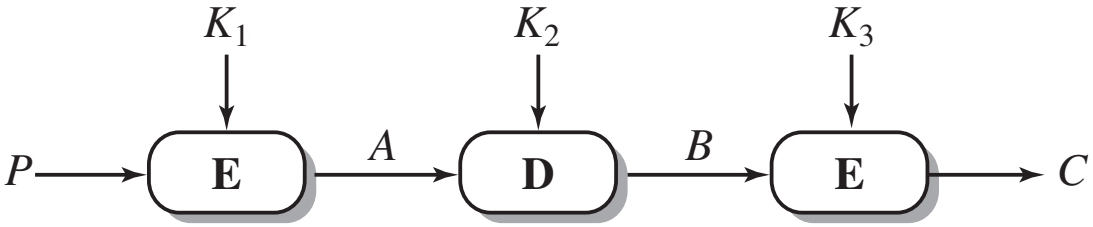
Meet-in-the-Middle Attack to 2DES



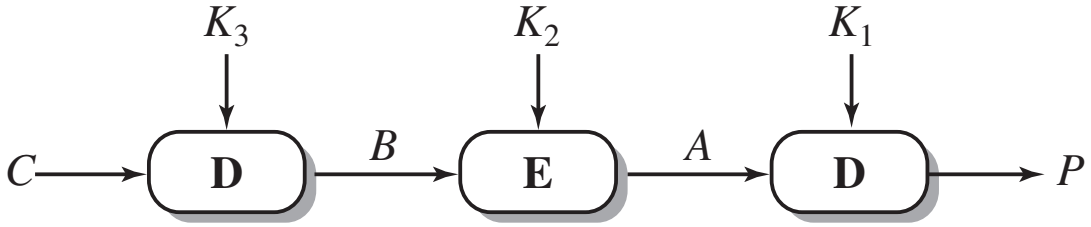
- Is the effective key length still 112? or 57 bits?

About 3DES

- Question 1: mode configuration (i.e., EEE or EDE or EED or . . .)
- Question 2: how many keys to use (1 or 2 or 3 keys)?
- The 3DES standard has used EDE mode with 2 keys
  - EDE mode can be reduced back to DES (if use same key three times)
  - Let k3 == k1 then the equivalent key length is 112 bits



(a) Encryption



(b) Decryption

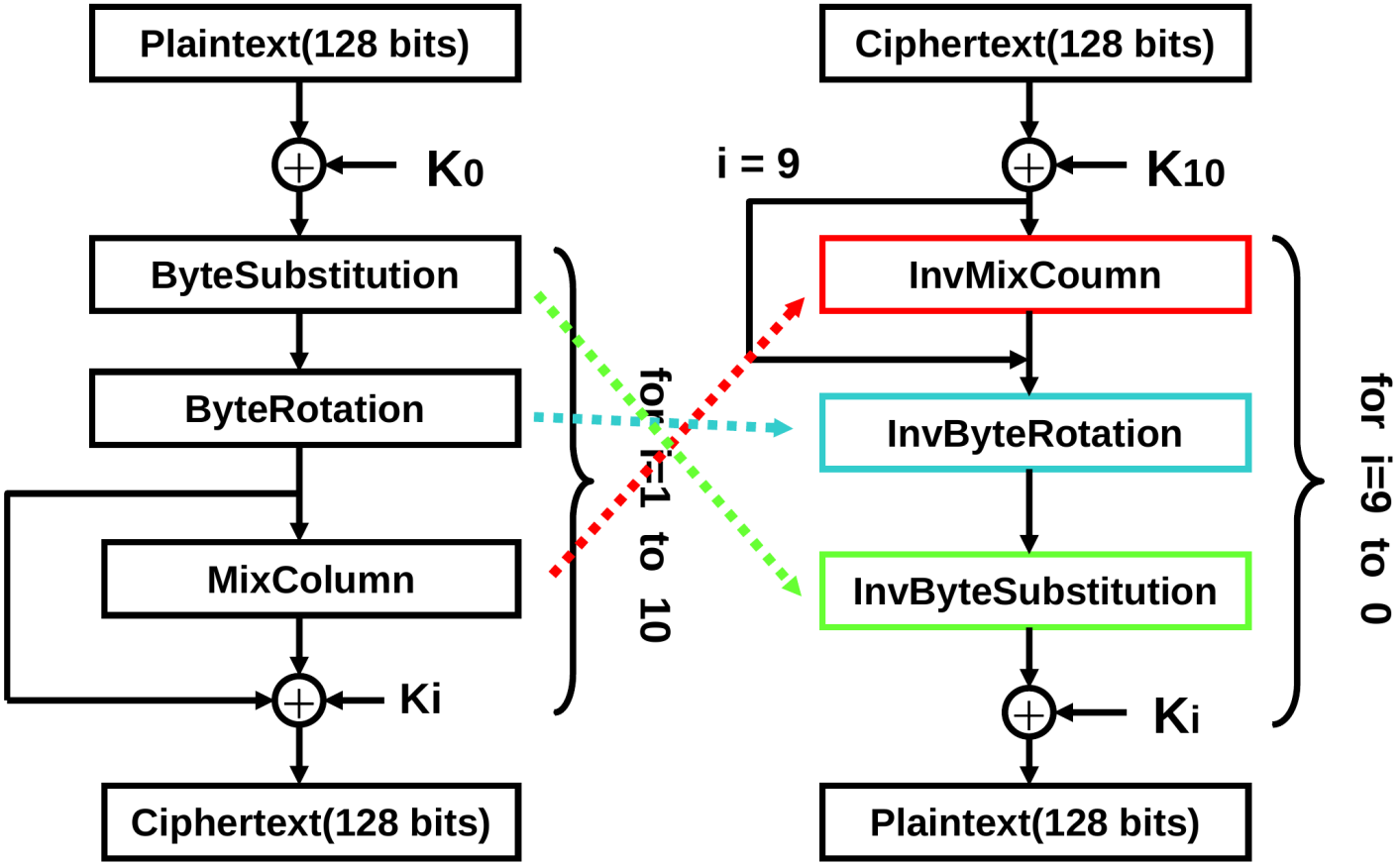
- But 3DES is only a temporary solution, so here comes AES

AES (Advanced Encryption Standard)

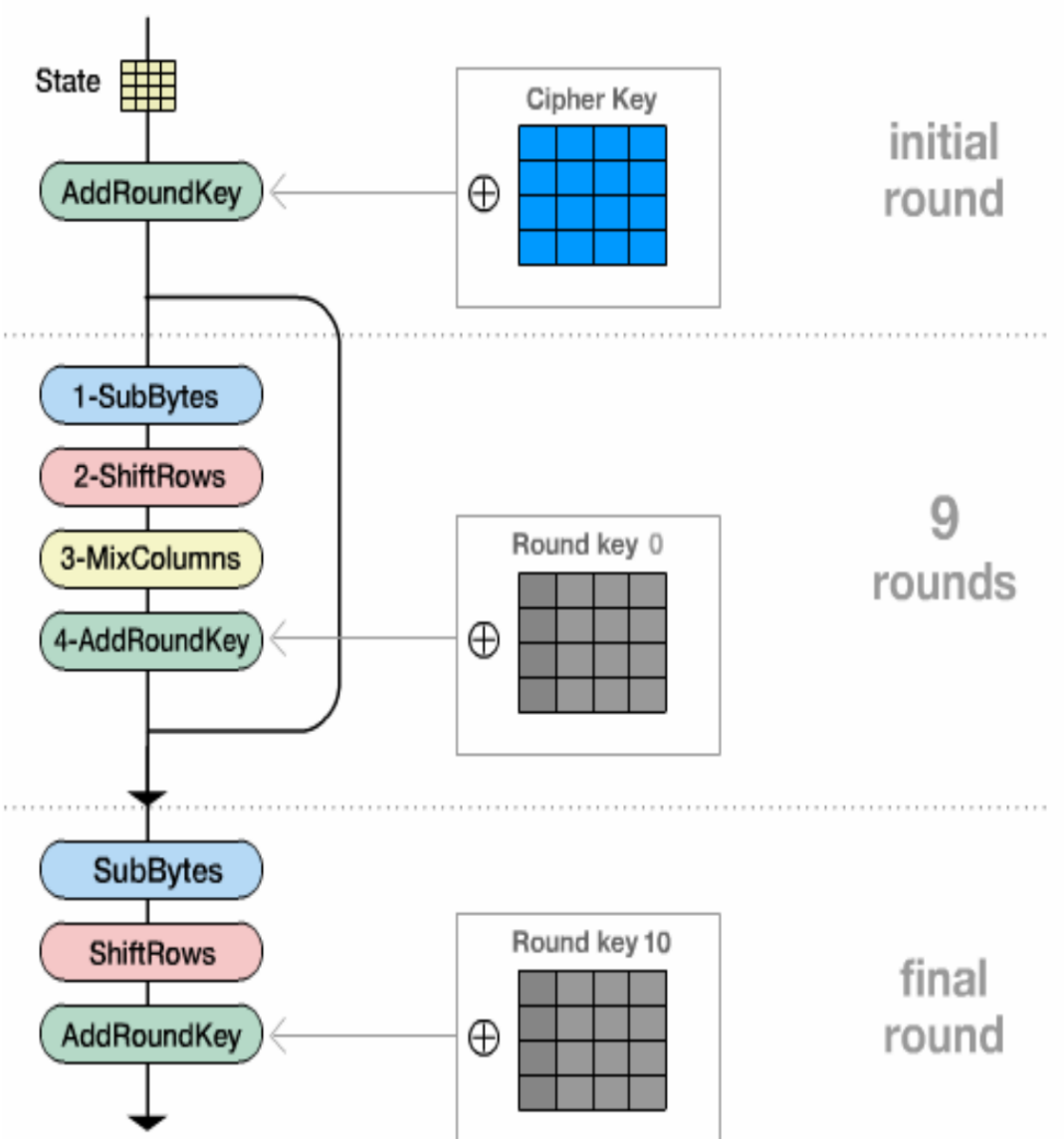
- NIST had an open call for proposlas in 1997 for AES standard
- Rijindael ['reinda:l] algorithm was the winner 2002 and standardized as AES in 2002
  - Created by two Belgium cryptographers: Joan Daeman and Vicent Rijmen
- AES is expected to replace DES and 3DES
- There are actually total five final condidates (out of 21 submissions)
  - MARS (IBM) - complex, fast, high security margin
  - RC6 (USA) - v. simple, v. fast, low security margin
  - **Rijndael** (Belgium) - clean, fast, good security margin
  - Serpent (Euro) - slow, clean, v. high security margin
  - Twofish (USA) - complex, v. fast, high security margin

Overview of AES

- Data block size is 128-bit
- Encryption can be different lengths (128, 192, 256)
- The number of encryption round will depend on key length
- An animation to show how AES works (Recommended) https://www.youtube.com/watch?v=gP4PqVGudtg

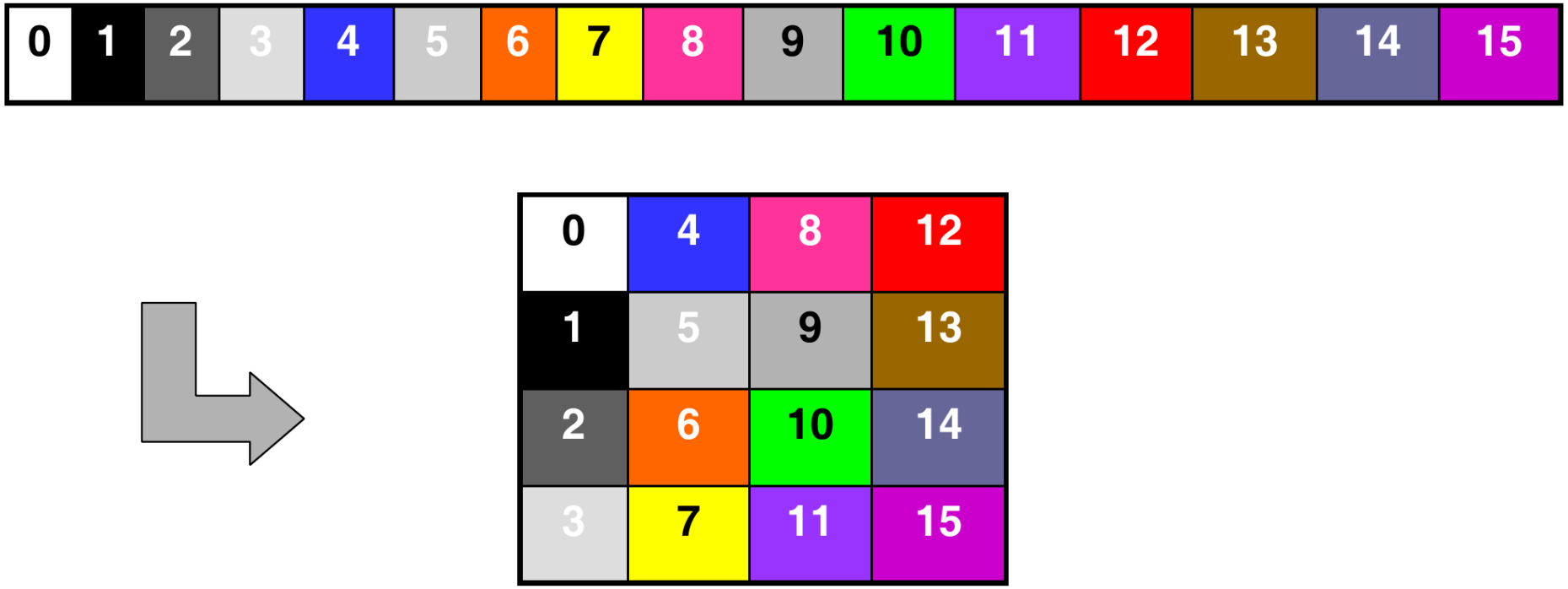


Single Round of AES



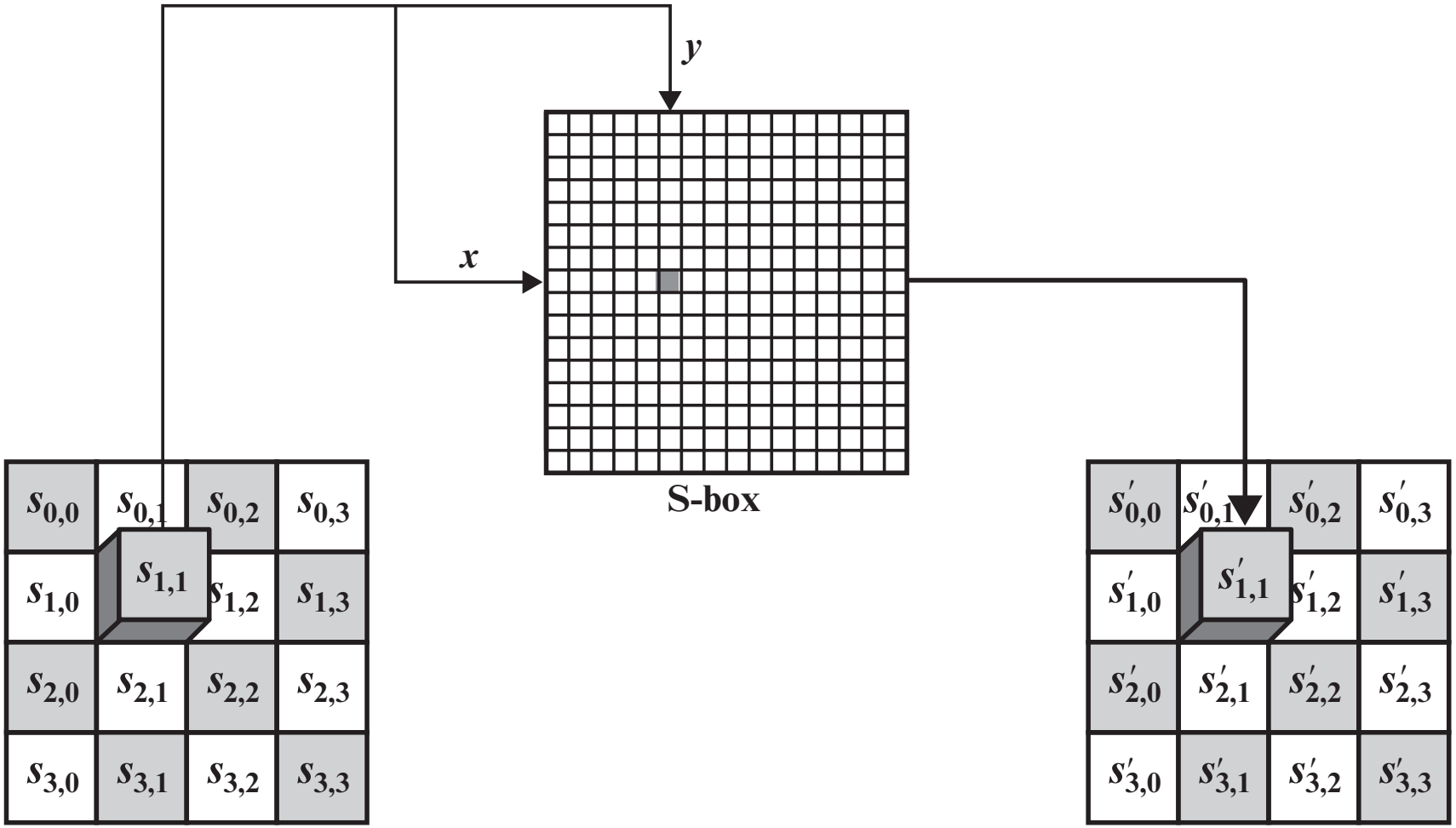
AES State Organization

- Each state has the same size as data block size (128 bits)
- 128-bit –> 16 bytes –> 4x4 matrix



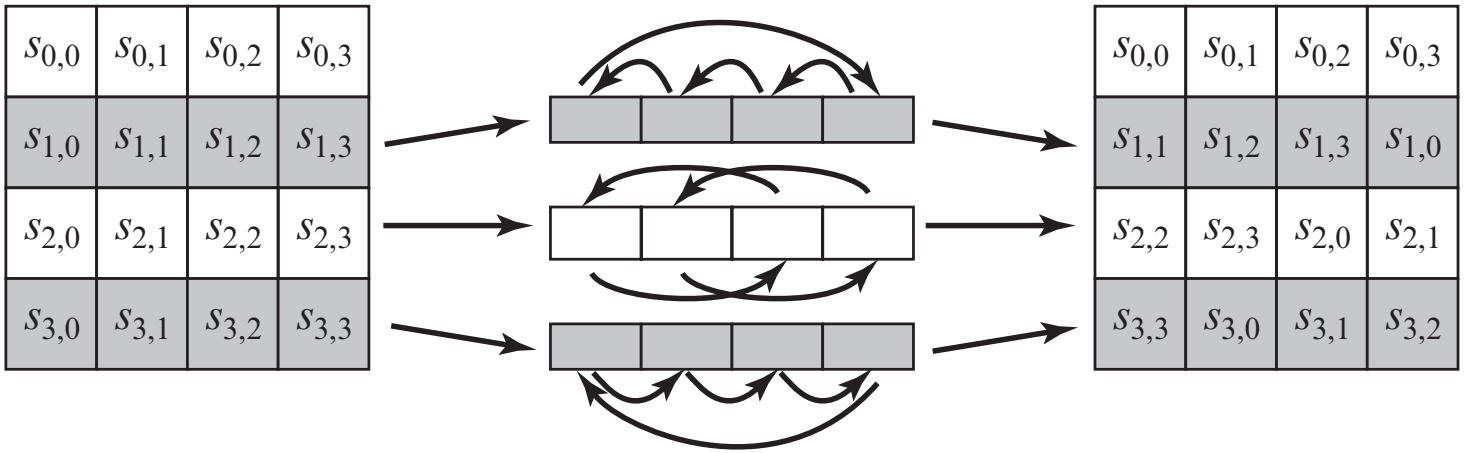
Byte Substitution and S-Box in AES

- S-Box is generated with some math process (vs. black-box in DES)



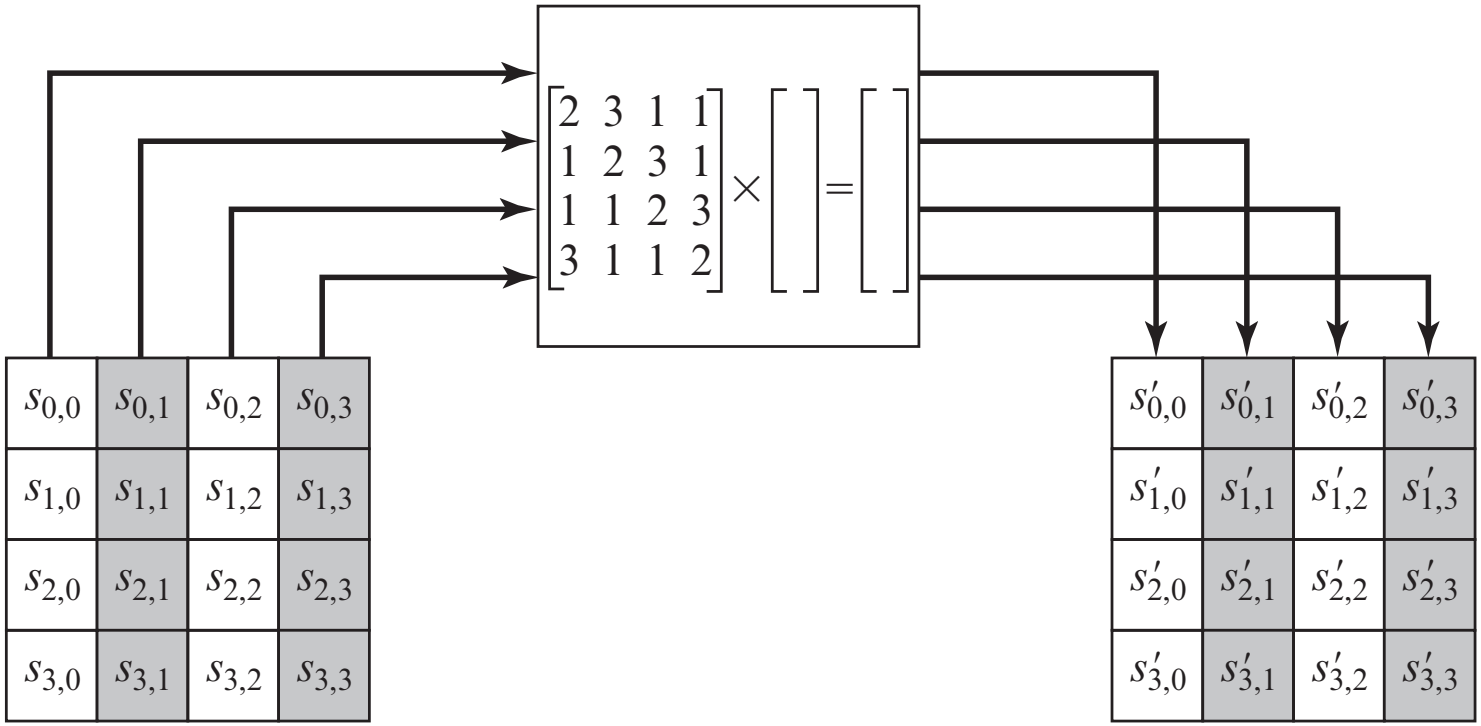
Row Shifting (Rotaion) in AES

- Circular Left Shifting *n* positions for the *n*-th row (n=0,1,2,3)
  - No shifting for the first row (i.e., row 0)



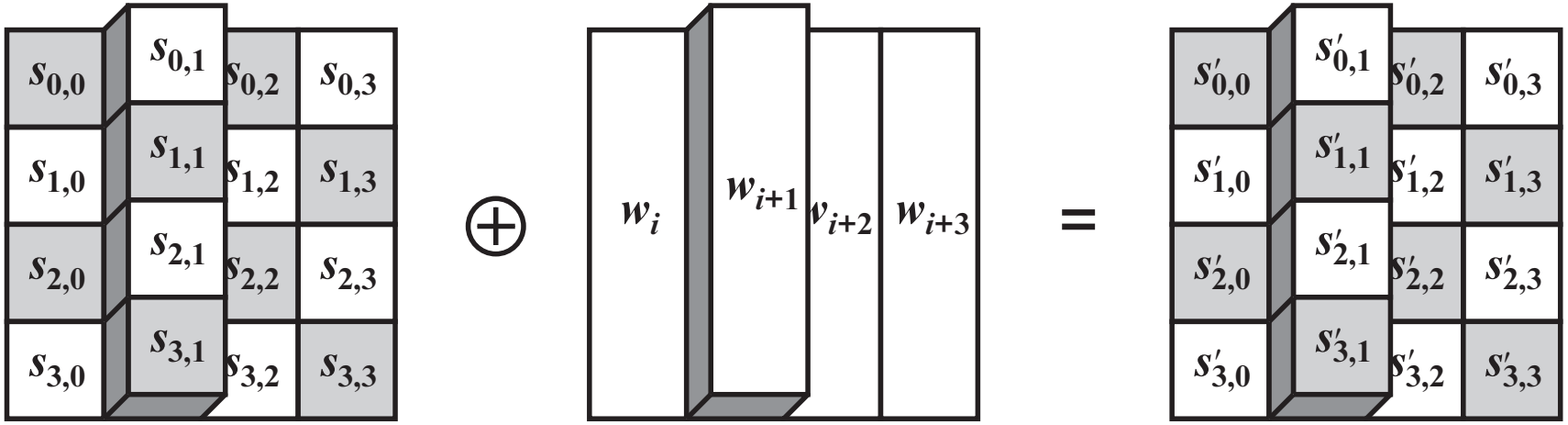
Column Mixing in AES

- The constant matrix is result of some math theory



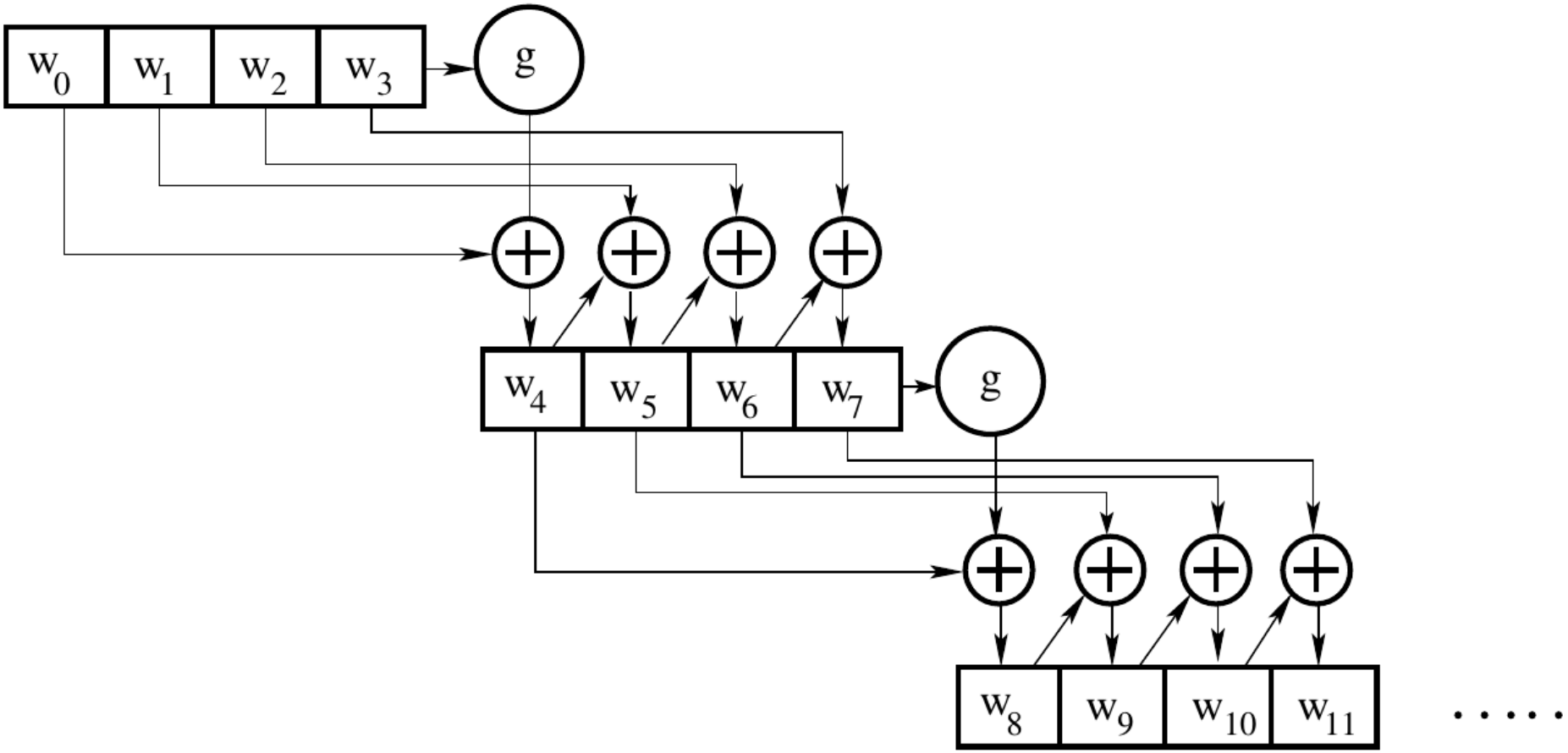
Add Round Key

- Key adding is done column by column
  - Because Key generation is also done column by column (next page)



Generating Round Key

- 128-bit key –> 4x4 matrix –> 4 columns
- The first column of round key (e.g., w4) is a little bit special

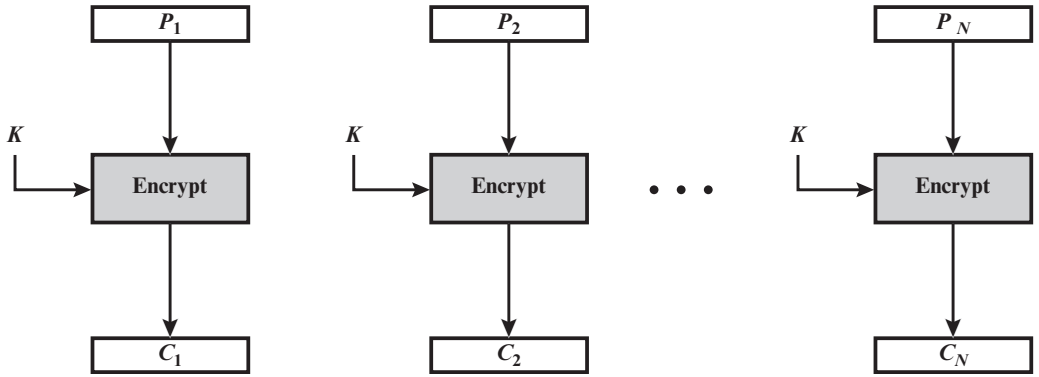


Operation Mode for Block Cipher

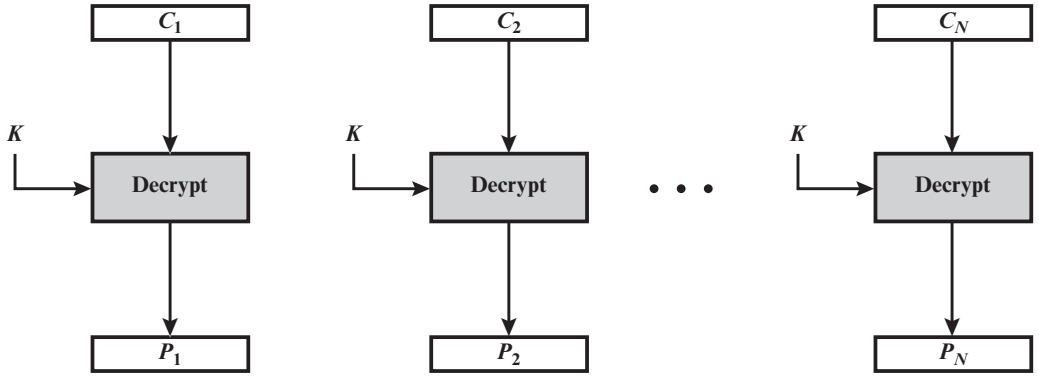
- Divide plaintext into blocks with size of 64-bit or 128-bit,
  - But how will these blocks be correlated to each during encryption?
- There several different modes
  - ECB: Electronic Code Book Mode
  - CBC: Cipher Block Chaining Mode
  - CFB: Cipher Feed Back Mode
  - OFB: Output Feed Back Mode
  - CTR: Counter Mode

ECB: Electronic Code Book Mode

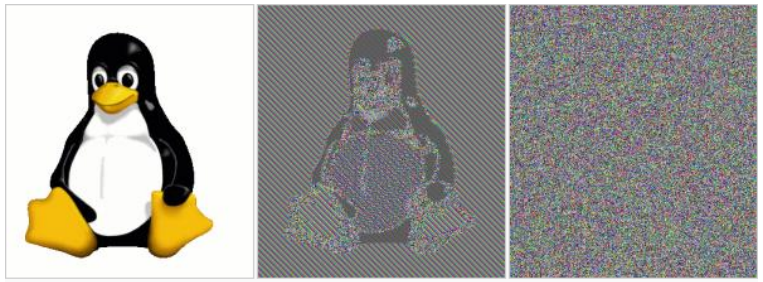
- Blocks are independent to each other and encrypted with the same Key
  - Not recommended due to security limitations:
    - ★ Repeated plaintext will lead to the same cipher text
    - ★ Blockwise swapping of ciphertext may still proceduce meaningful output upon decryption



(a) Encryption



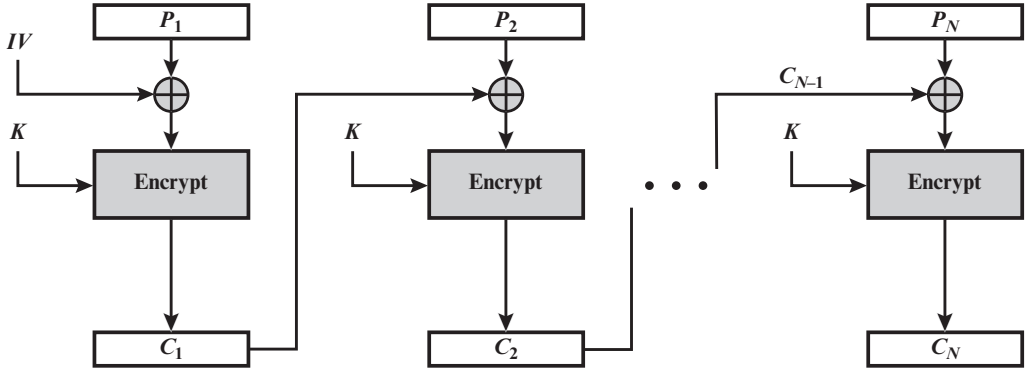
(b) Decryption



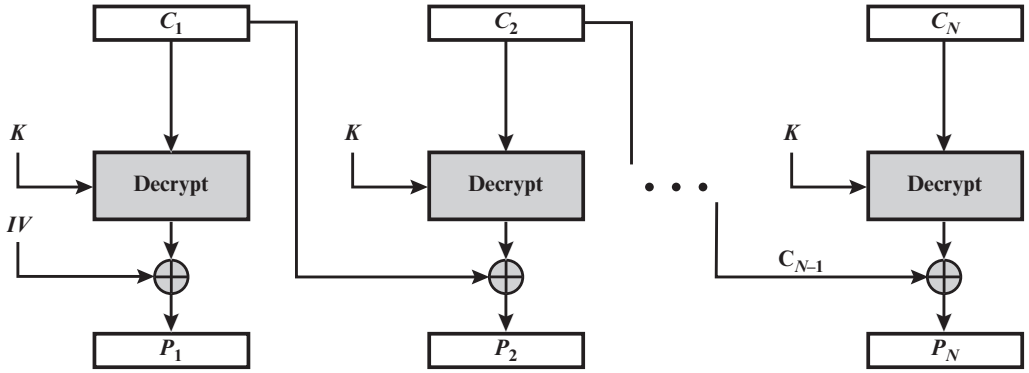
Original image | Encrypted using ECB mode | Modes other than ECB result in

CBC: Cipher Block Chaining Mode

- Previous block's output will be XOR-ed with input of current block
  - For the 1st block, an random *Initial Vector* will be used as XOR input
  - Repeated patterns will not be exposed
  - Sensitive to noise communication channel (only current and next block)



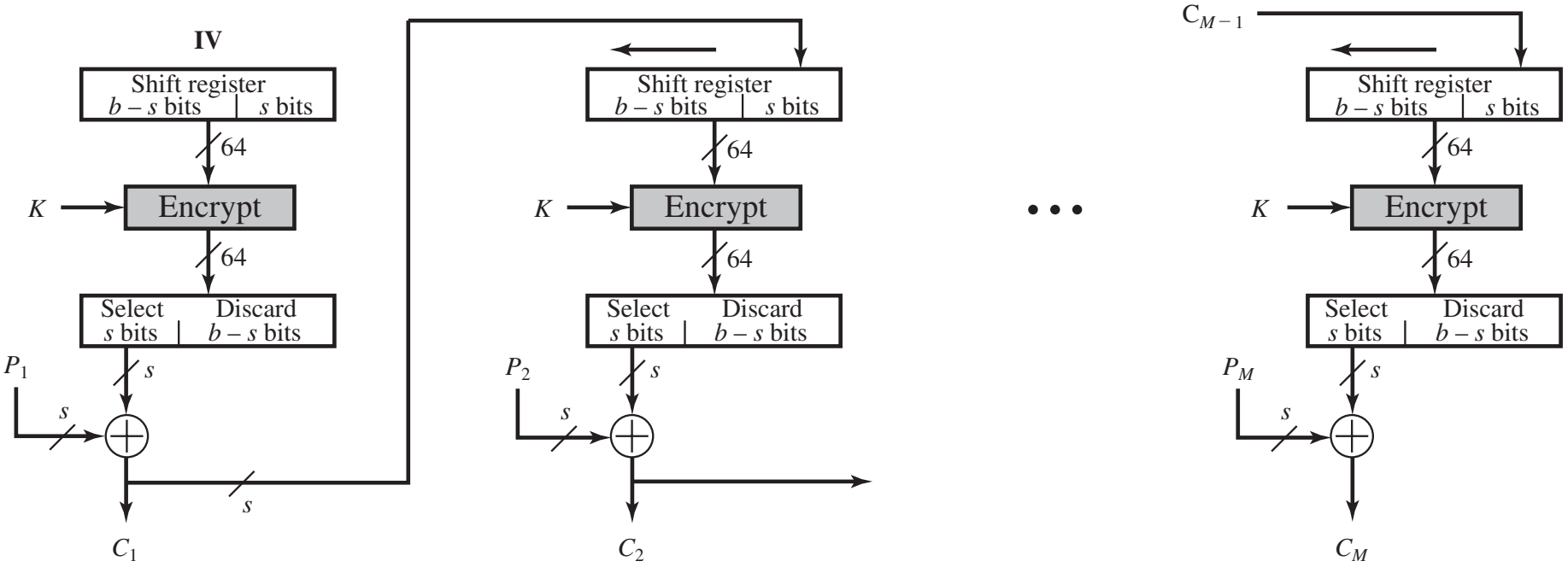
(a) Encryption



(b) Decryption

CFB: Cipher Feed Back Mode

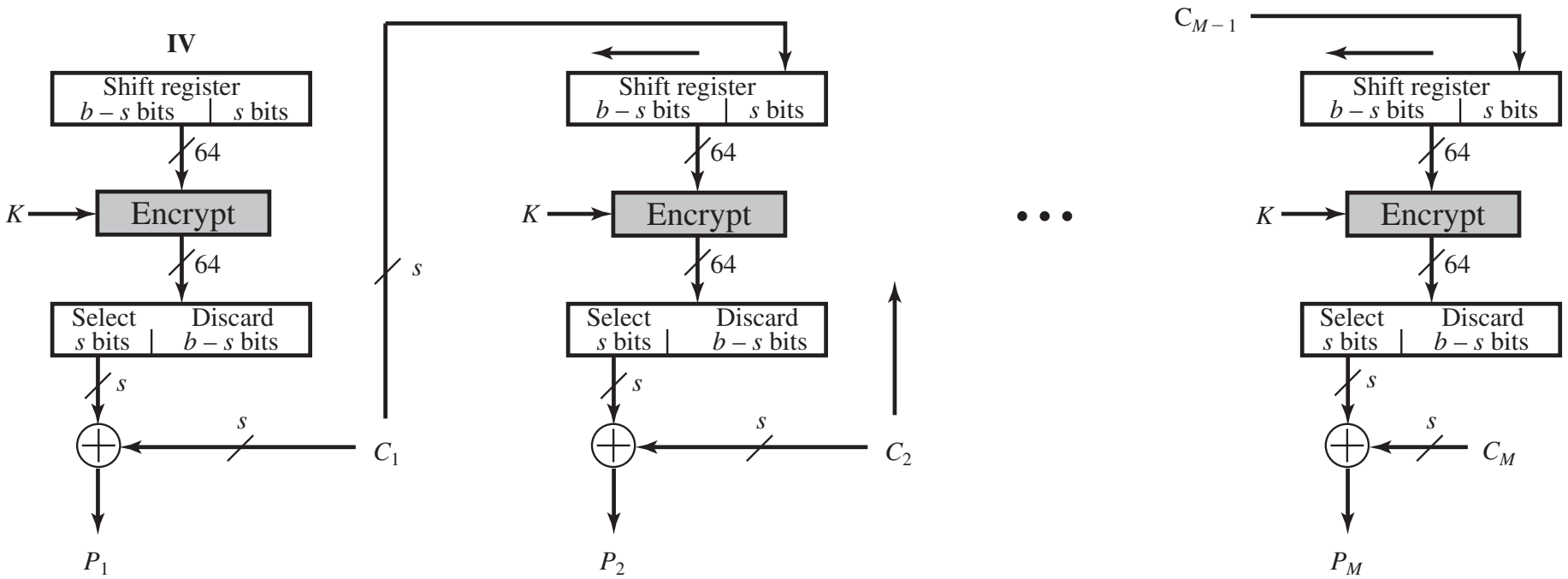
- Convert block cipher into a kind of stream cipher
  - Using Block cipher and a shifting initial vector (with s-bit feedbacks) to encryption the KEY (CFB-1, CFB-8, CFB-64, etc.)
  - Plaintext is XOR-ed with output of above block cipher
  - part of the XOR-ed results will be feedback to the shifting register
  - But with performance drawbacks (due to nature of block operation)
  - Sensitive to noise communication channel (could affect current, next and more following blocks, depends on relationship between b and s)



(a) Encryption

CFB: Cipher Feed Back Mode (cont.)

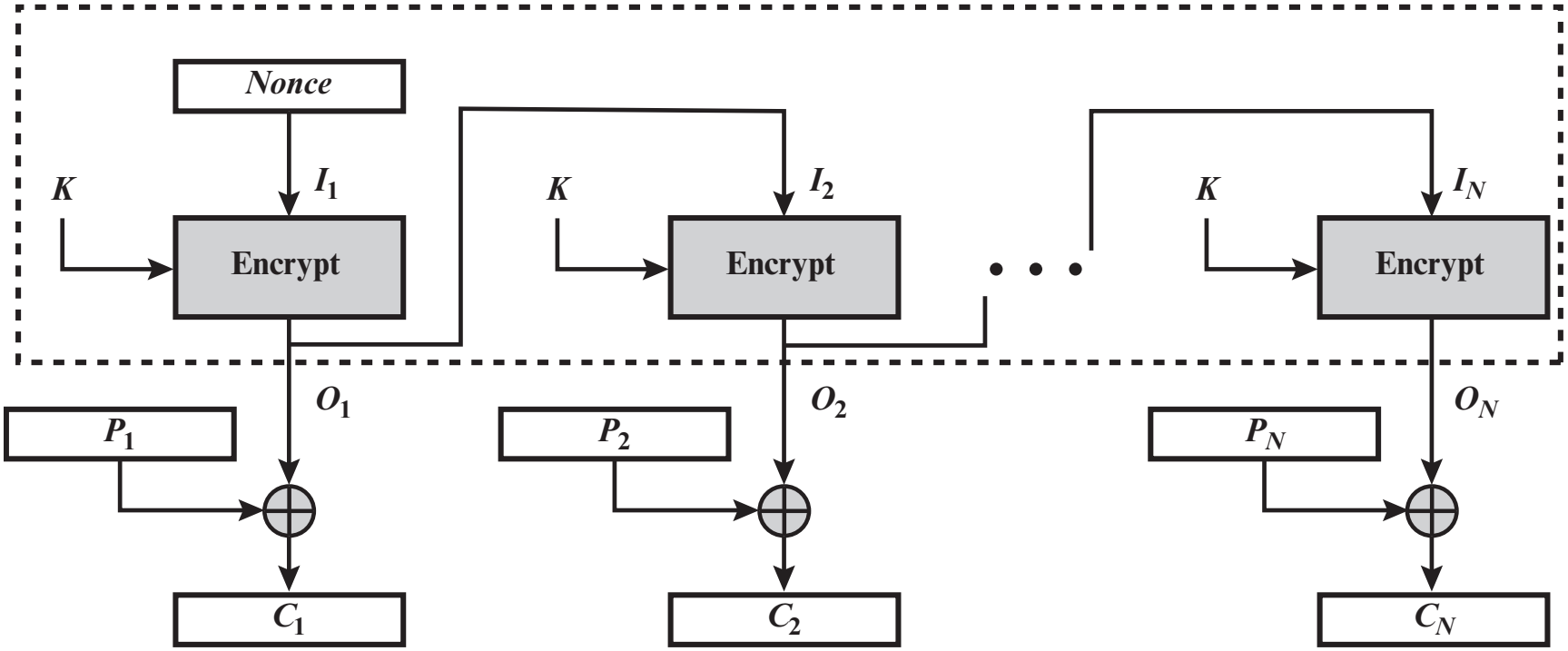
- The concept of **confusion** and **diffusion**
  - Confusion means that each single bit of ciphertext should depend on several parts of the key
  - Diffusion means any single bit change in plaintext will lead to half of the bits (statistically) changes in ciphertext, and vice verse
  - Both of them are supposed to make connections among plaintext, key and ciphertext more complex and thus cryptoanalysis is more difficult



(b) Decryption

OFB: Output Feed Back Mode

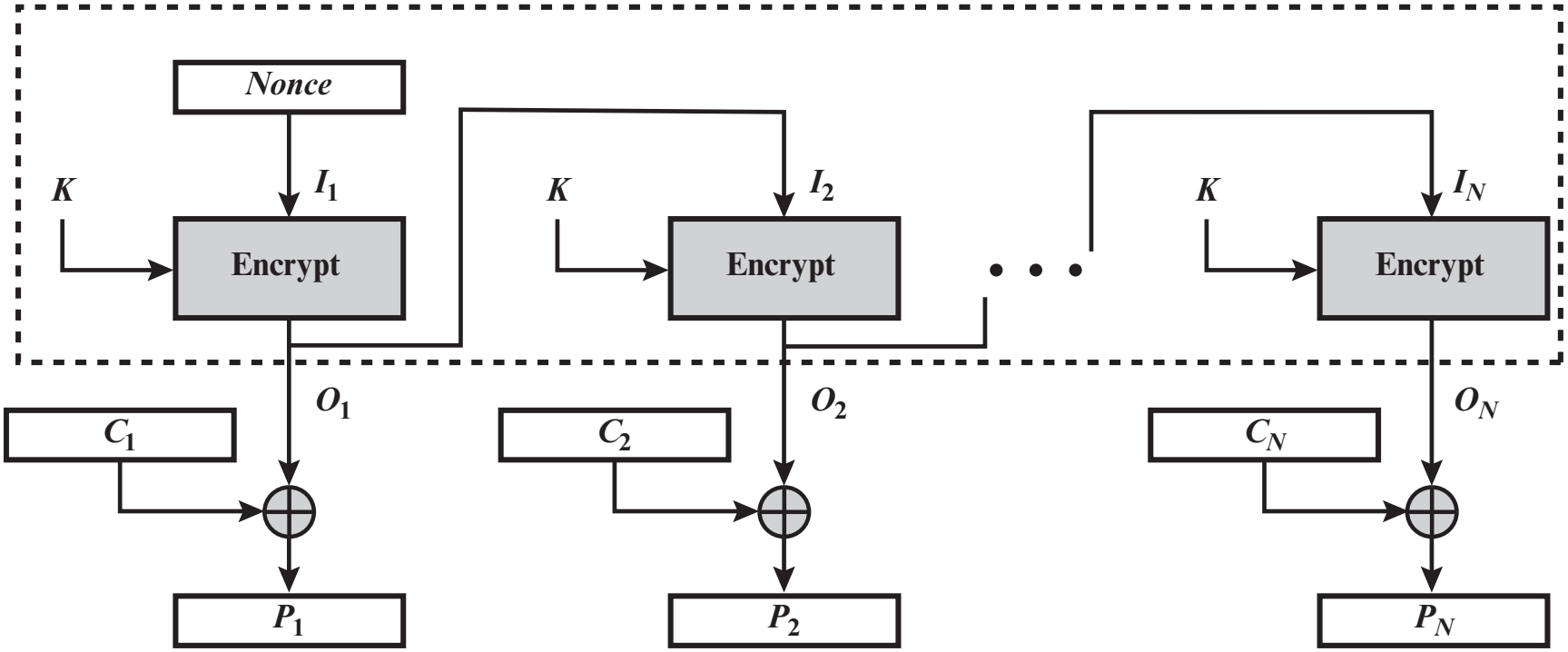
- Similar to CFB
- But the feedback is derived only from keys
  - Even more robust to communication channel noises
  - The feedback can be calculated in advance
    - ★ Since it is independent to plaintext and ciphertext



(a) Encryption

OFB: Output Feed Back Mode (Cont.)

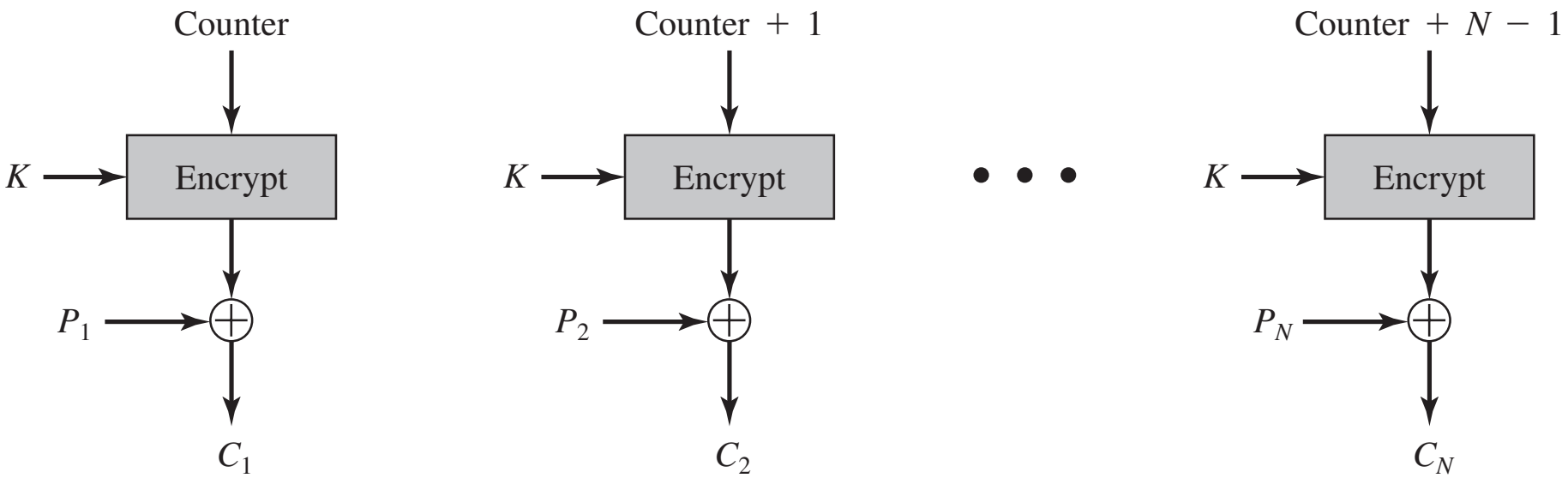
- Possible limitations
  - Never reuse the Key and the Nounce (i.e., Initial Vector)
    - ★ Why? vulnerable to known- or chosen-plaintext attack
  - Block sequence synchronization becomes more important
    - ★ Packet lost or out-of-order will cause decryption failure, even every single bit is received correctly



(b) Decryption

CTR: Counter Mode

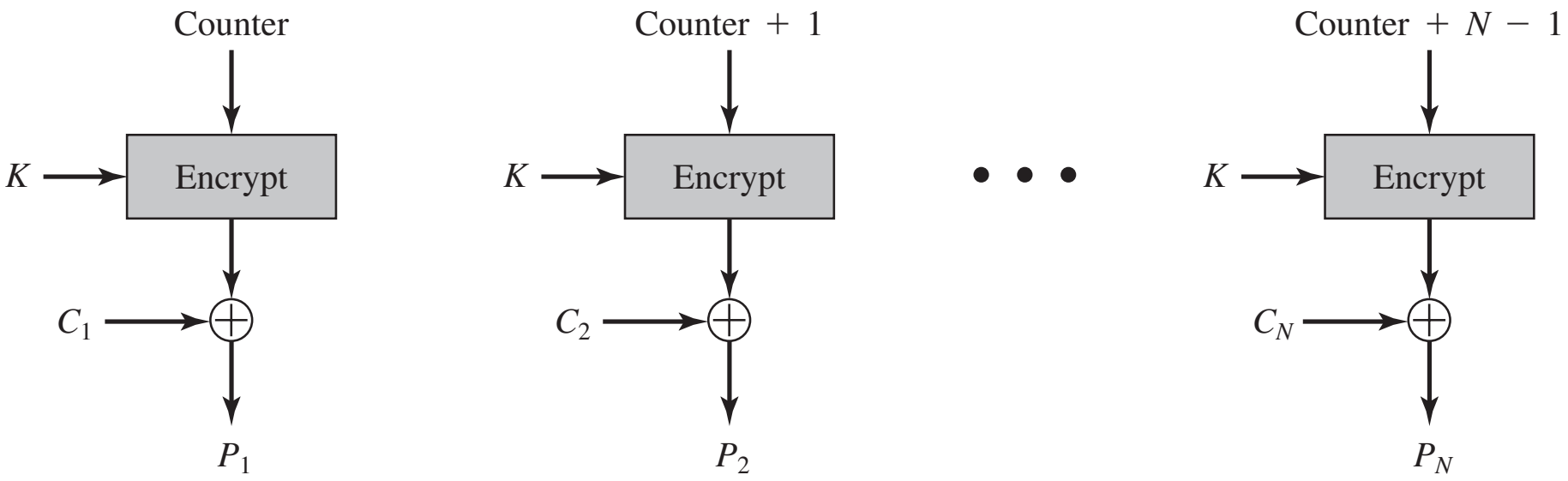
- No feedback any more
  - Feedback is replaced with a couting number
- What are the benefits?
  - Parallel operation: no feedback → no inter-dependence
  - Random access to encrypted data blocks (no inter-dependence)
  - Robust to noisy channel
  - Some data can be pre-calculated → good for burst communication at high-speed



(a) Encryption

CTR: Counter Mode (Cont.)

- Is it secure?
  - It is secure as long as the key and counter values are not reused
    - ★ The initial Counter values could be random (i.e., no need to start from 0 everytime, as long as synchronized between two parties)

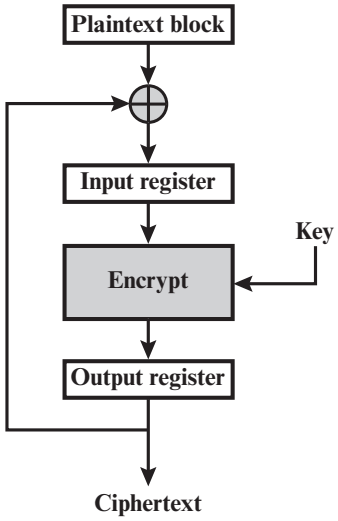


(b) Decryption

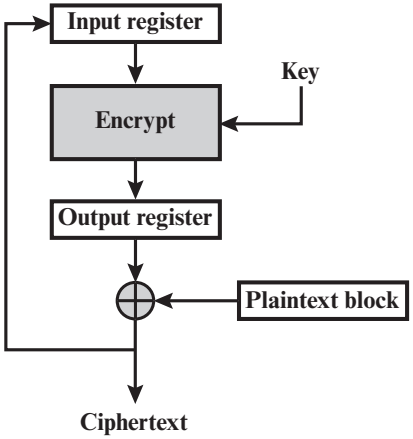
Comparision

- Key Points to Differentiate them
  - When will plaintext will be used
    - Some follow block cipher, others may follow stream cipher
  - Will the plaintext affect the feedback?
    - In some modes, feedback is derived from plaintext, while others don't
  - How will the feedback be generated?
    - Some use full or partial ciphertext directly
    - Some does not use ciphertext to generate feedback
    - CTR mode use indirect feedback ($counter_{i+1} = counter_i + 1$)

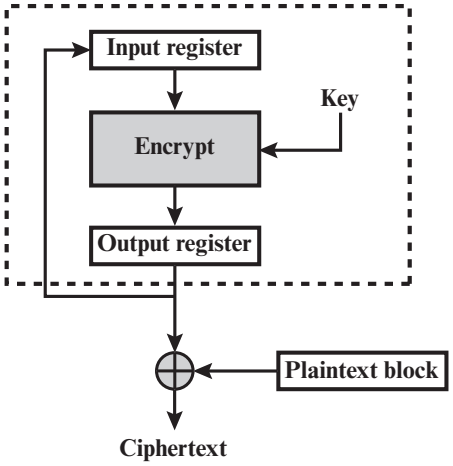
Comparision (Cont.)



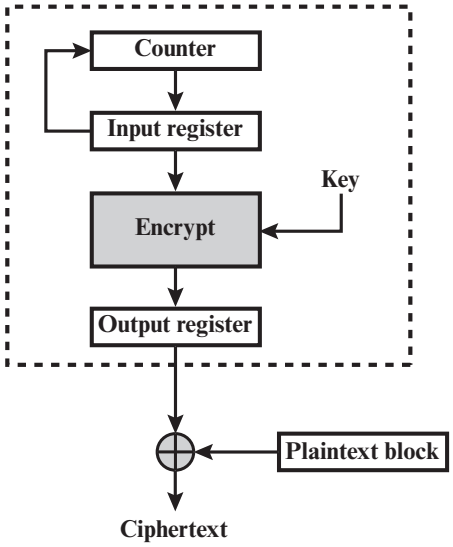
(a) Cipher block chaining (CBC) mode



(b) Cipher feedback (CFB) mode



(c) Output feedback (OFB) mode



(d) Counter (CTR) mode