Block Cipher Modes of Operation

February 1, 2022

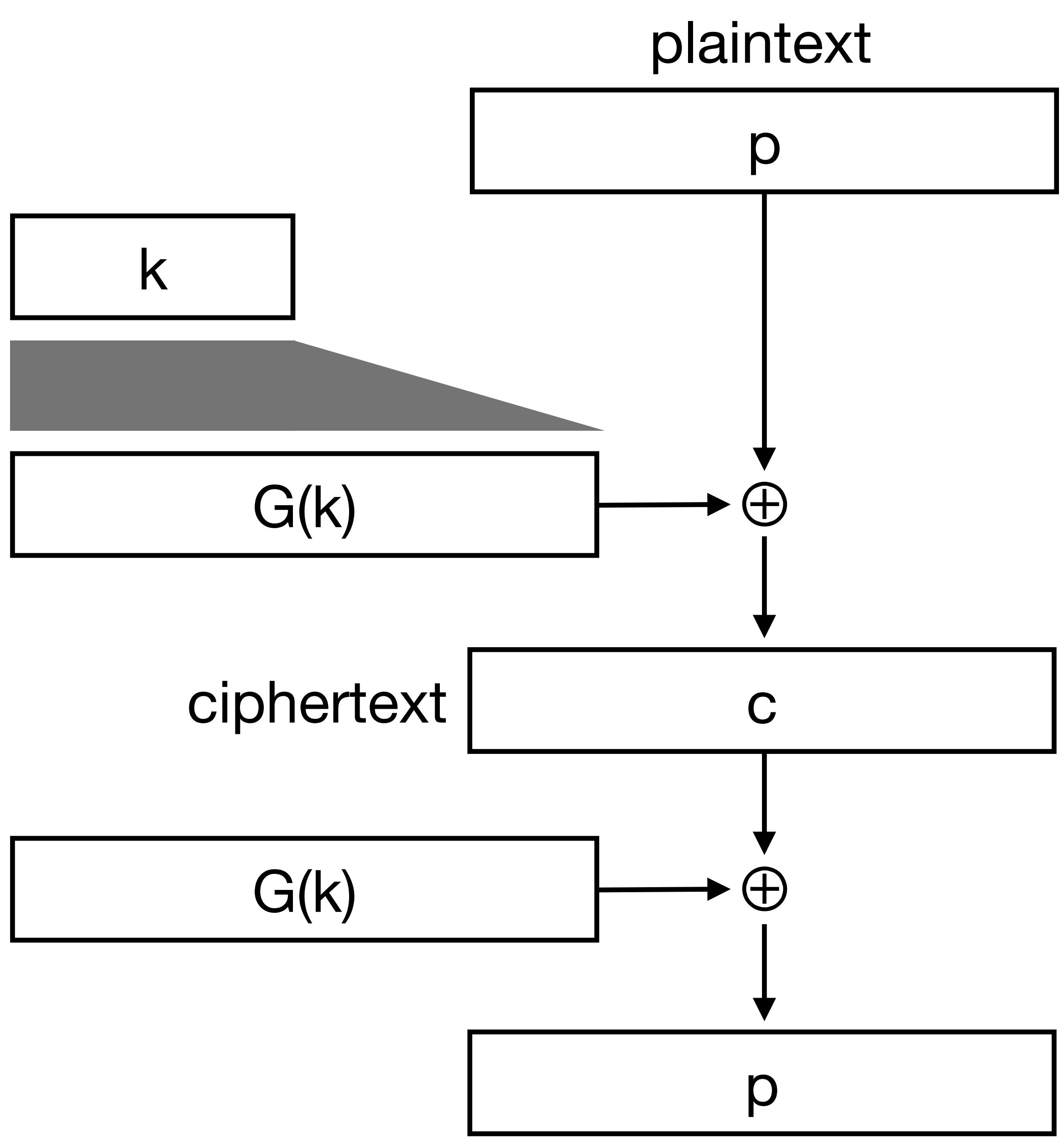
Homework 1 & Today

- Homework 1
  - will be available on **Blackboard** this week
  - based on cryptography lectures, **requires Python or Java programming**
  - due **February 20th** (Sunday) at 11:59pm
- Today:
  How to use block ciphers in practice?
  - multiple encryption
  - block cipher modes of operation:
    ECB, CBC, CFB, OFB, CTR

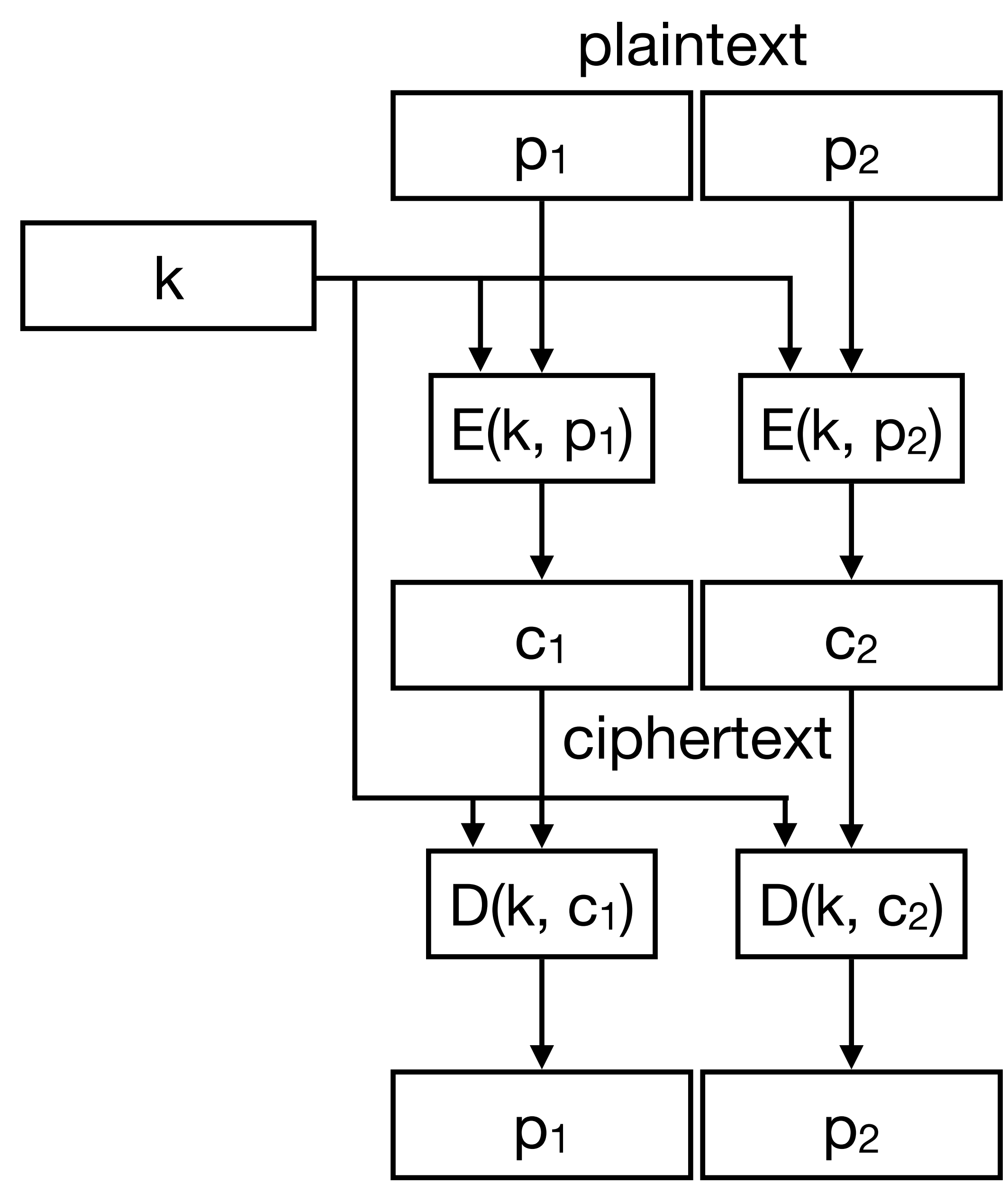
Feedback: `https://forms.gle/JGbNCmCsU69iWaTv8`

Reminder: Encryption

Stream ciphers



Block ciphers



Multiple Encryption

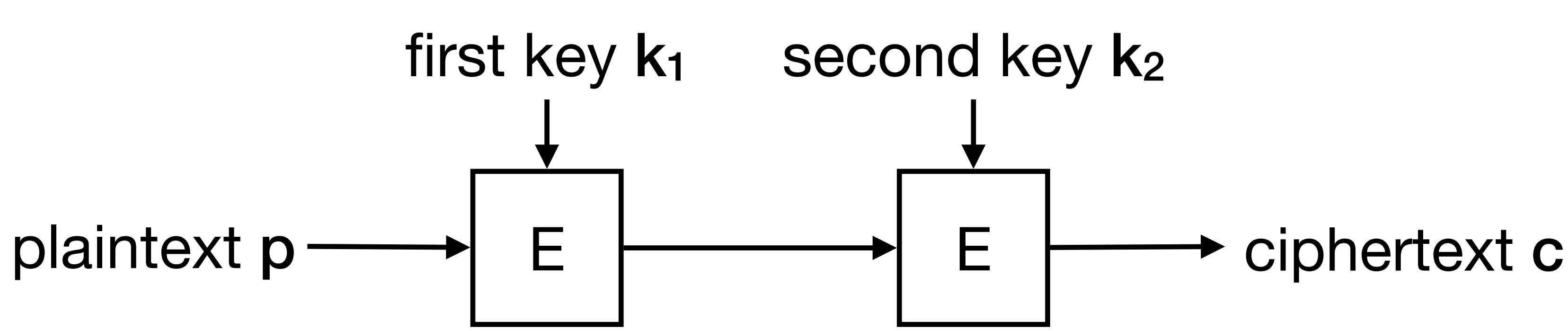
Motivation for Multiple Encryption

Why we do not like DES (anymore):
key size is **only 56 bits** → 2^{56} step brute-force attacks are feasible

Why we still like DES:

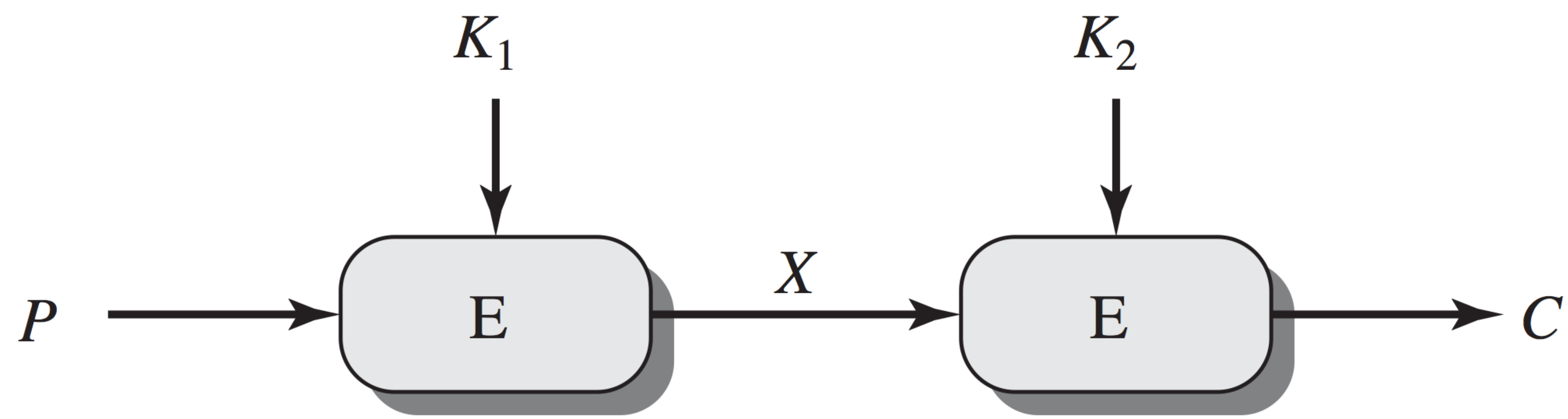
- **relatively secure against cryptanalytic attacks**
  (*best attack:* linear cryptanalysis in 2^{43} steps)
- thoroughly studied and widely supported

- **Multiple encryption**
  - use the same encryption algorithm multiple times, each time with a **different key**
  - widely used with DES, but the principle can be applied to any block cipher

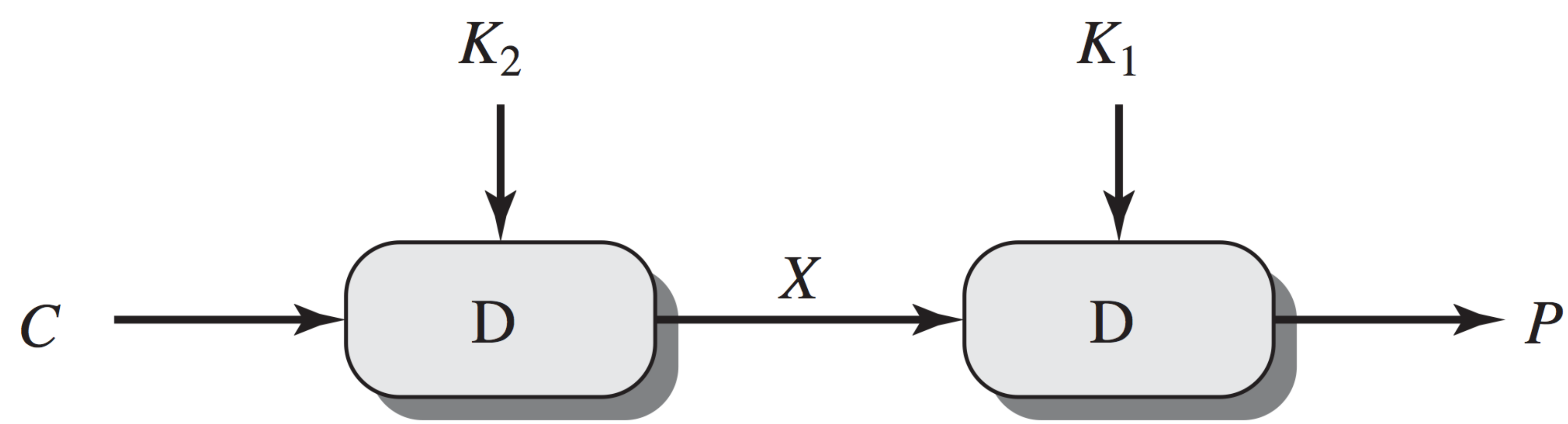


Double DES

$C = E(K_2, E(K_1, P))$



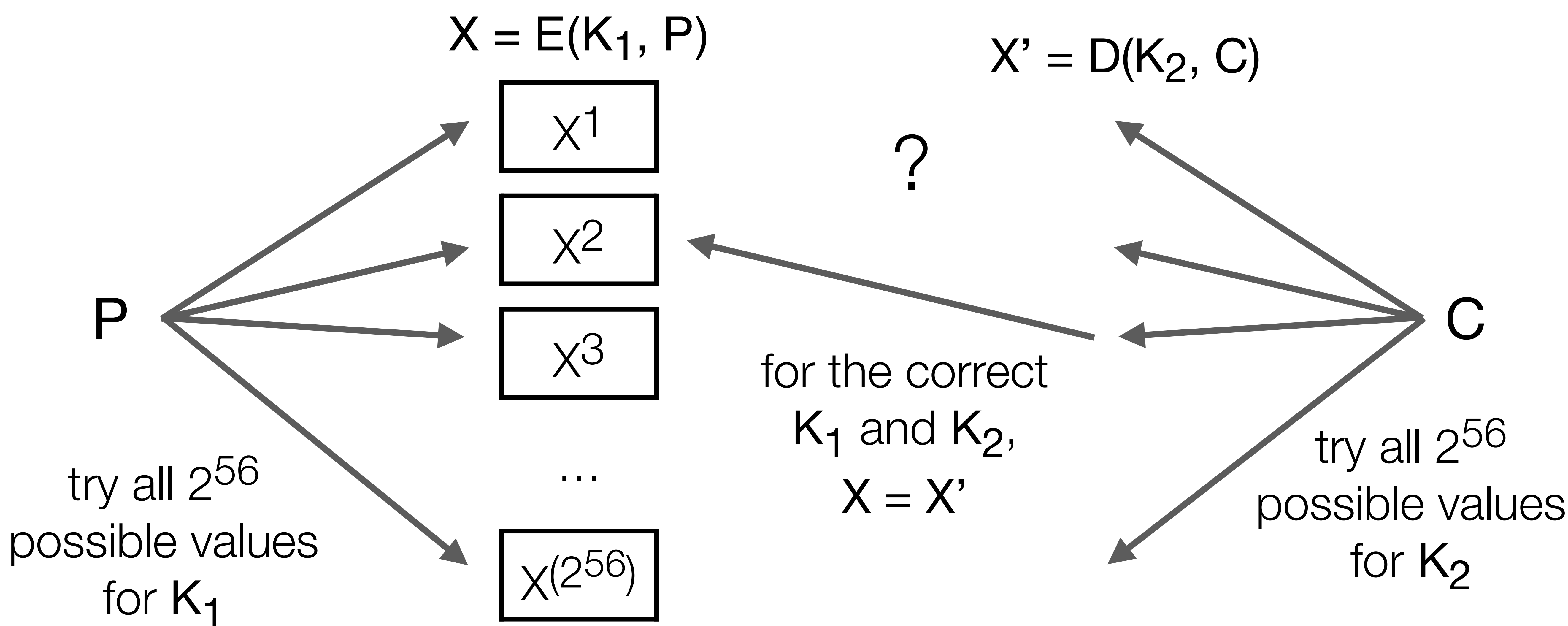
$P = D(K_1, D(K_2, C))$



key size = $2 \times 56 = 112$ bits

Meet-in-the-Middle Attack

Known-plaintext attack: suppose that attacker has a pair **P, C**

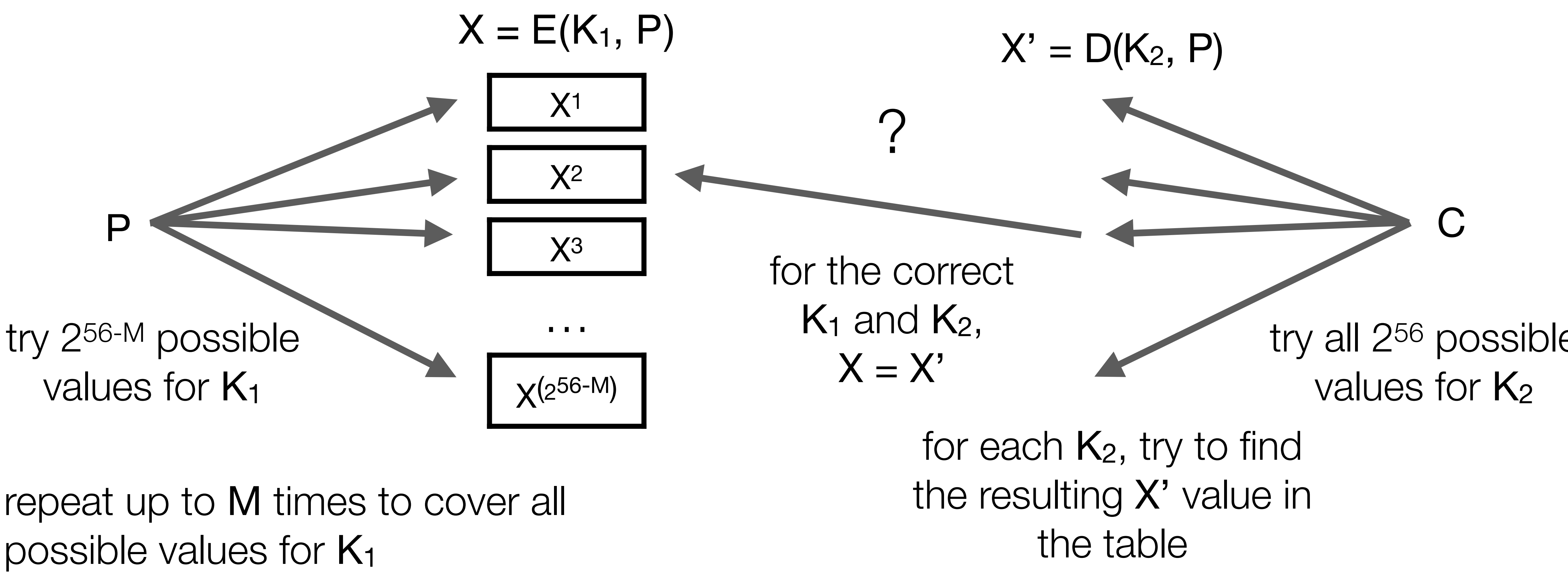


sort the resulting X^i values and store them in a table

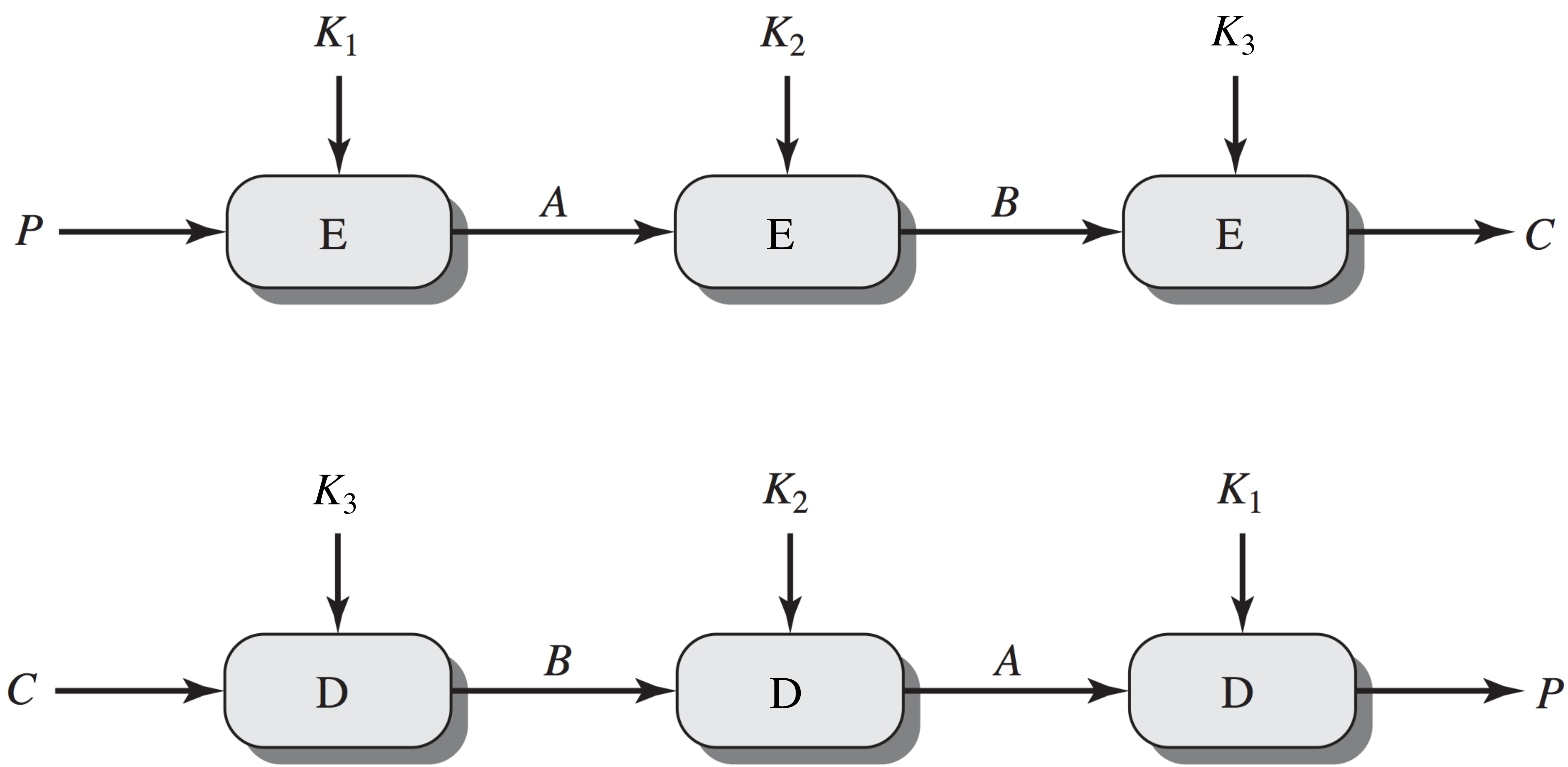
for each K_2, try to find the resulting X' value in the table

Meet-in-the-Middle Attack Requirements

- Meet-in-the-middle attack: **trading off time for storage**
  - simple brute-force attack → 2^{112} steps
  - storing 2^{56} values (see previous slide) → ~ 2^{56} steps
  - *generally*: **storing 2^{56-M} values → ~ 2^{56+M} steps**

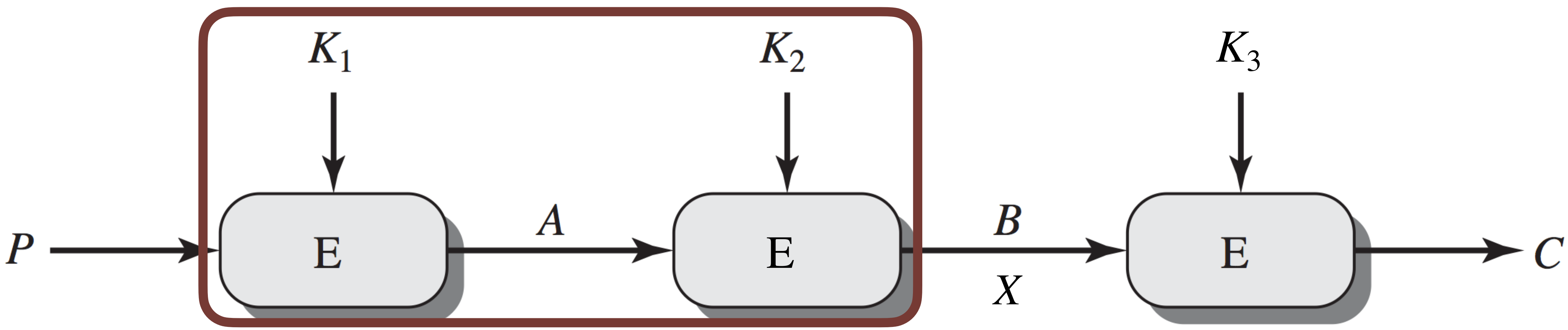


Triple DES (3DES)



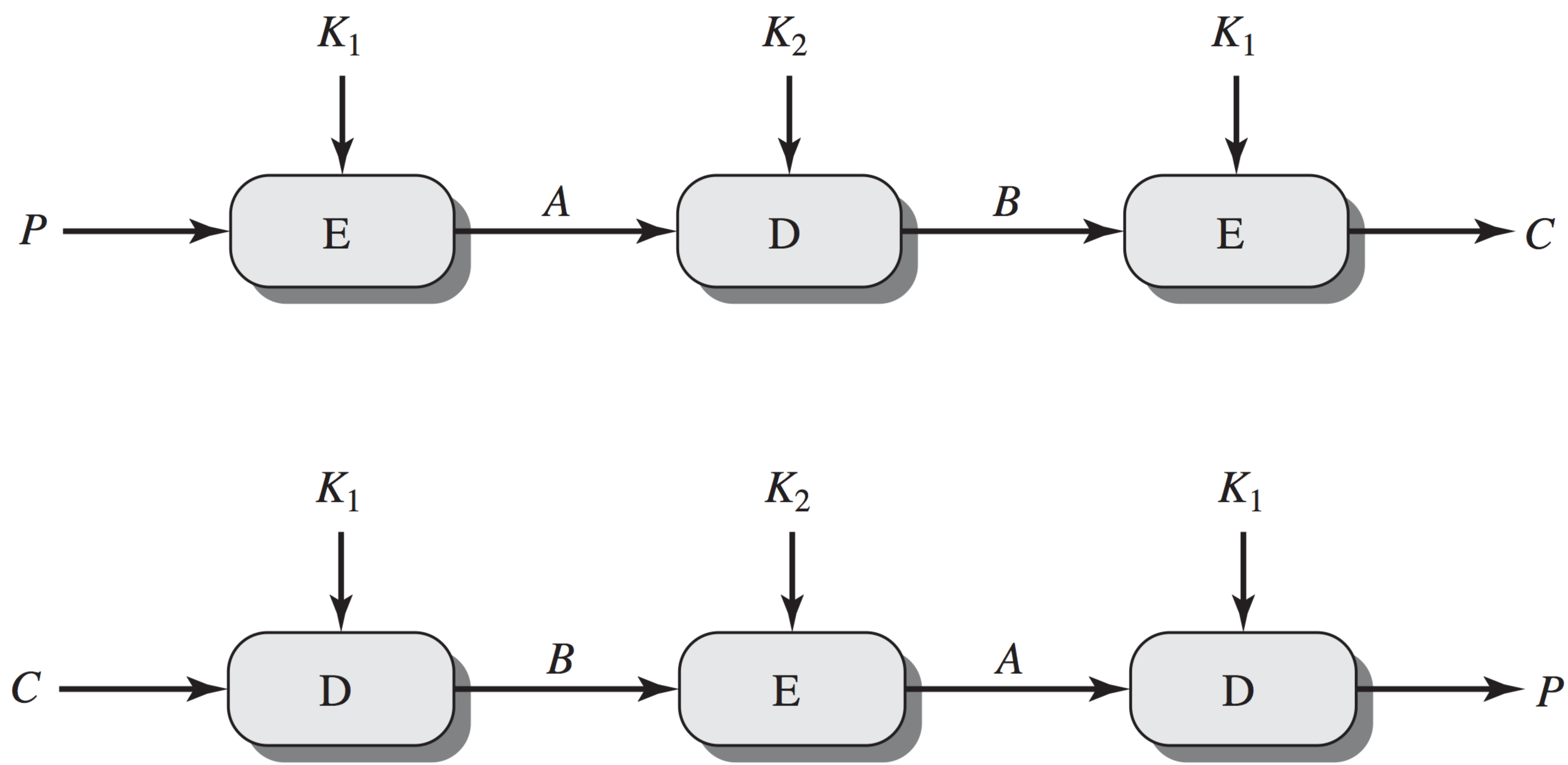
Triple DES (3DES)

- ~~Three keys ($3 \times 56 = 168$-bit key)~~
  - ~~more complex meet-in-the-middle attack → **effective security is only 112 bits**~~
    - ~~3DES can be viewed as a combination of two ciphers: one with a 56-bit key and one with a 112-bit key~~



encryption with 112-bit key (K_1 and K_2) encryption with 56-bit key (K_3)

Triple DES (3DES) with Two Keys



Triple DES (3DES)

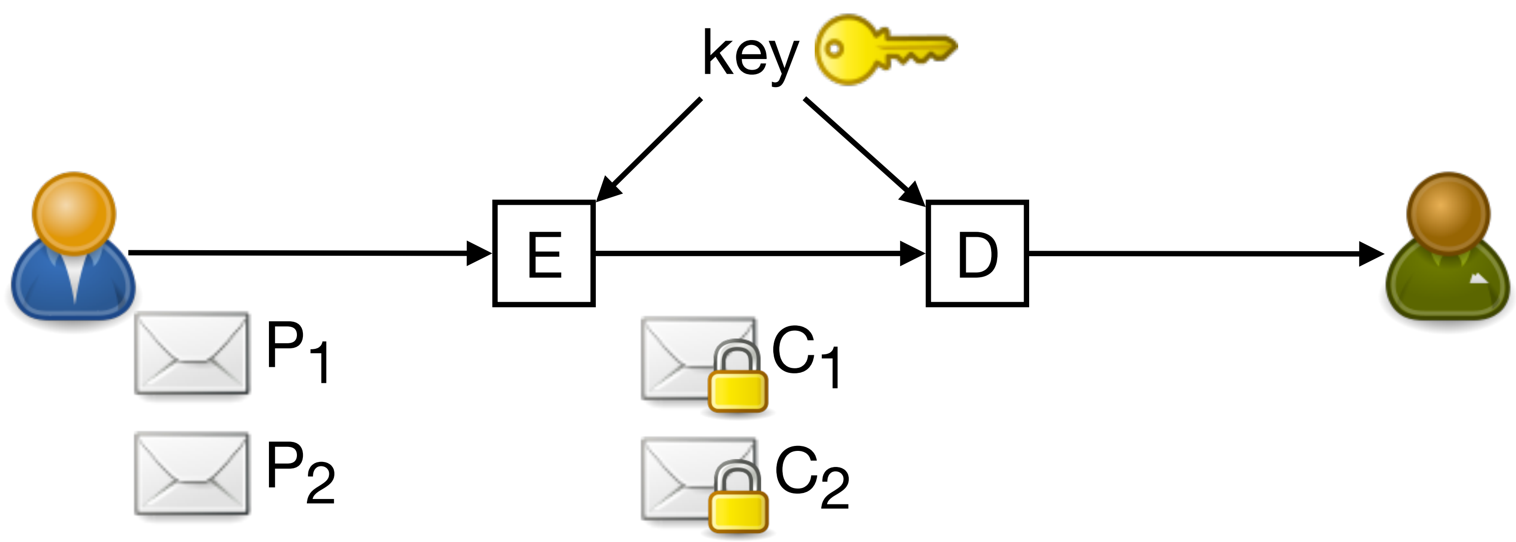
- ~~**Three keys** (3 × 56 = 168-bit key)~~
  - ~~more complex meet-in-the-middle attack → **effective security is only 112 bits**~~
    - ~~3DES can be viewed as a combination of two ciphers: one with a 56-bit key and one with a 112-bit key~~
- **Two keys** (2 × 56 = 112-bit key)
  - prevents the simple meet-in-the-middle attack presented earlier
  - however, there are other known-plaintext attacks
    → according to NIST, this approach **provides around 80 bits of security**
- **EDE** (Encryption-Decryption-Encryption) configuration
  - if $K_1 = K_2$, then 3DES is equivalent to DES → **compatibility with older systems**
- Unfortunately, 3DES is very slow and has a small block size

Block Cipher Modes of Operation

How to use block ciphers in practice?

Key Reuse

- We may have to use the **same key to encrypt multiple blocks**
  - **multiple plaintexts** (*e.g.*, sending multiple messages over an insecure channel)

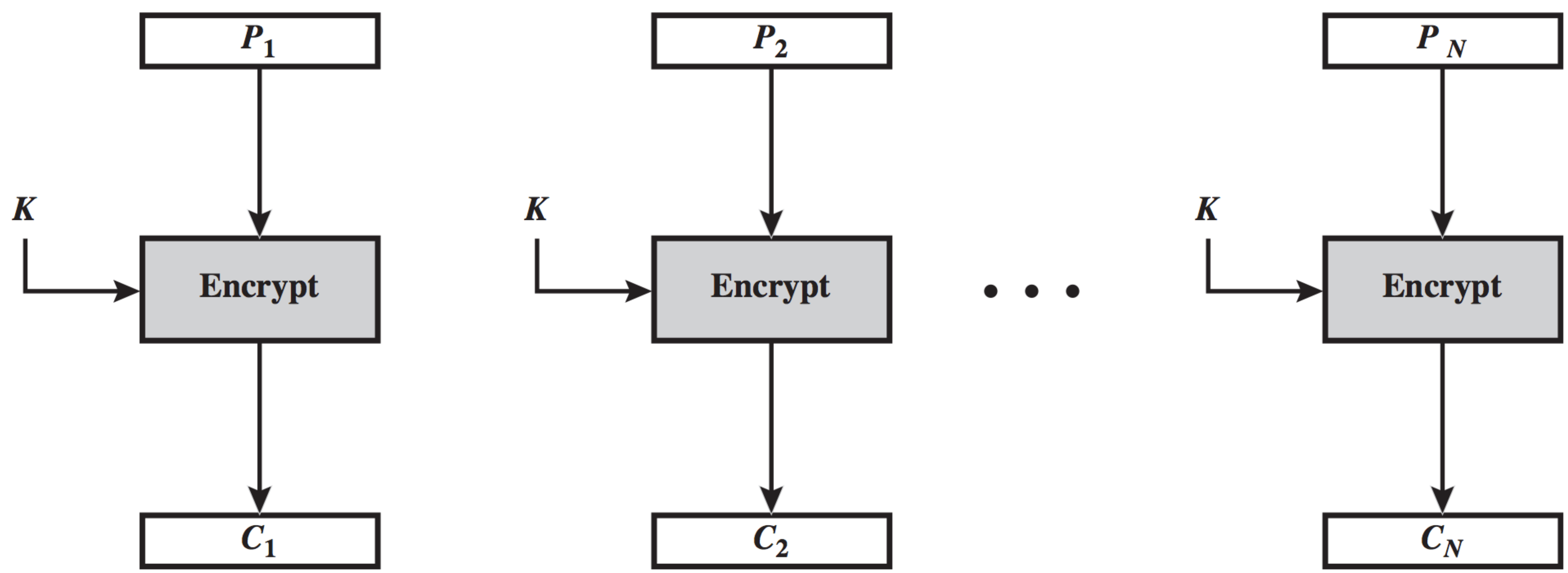


  - **long plaintext** → break up into fixed-size blocks
    P = "`The quick brown fox jumps`"
    P_1 = "`The quick bro`" P_2 = "`wn fox jumps`"

- *Reminder*: key reuse issue with stream ciphers (and one-time pad)
  - same key → same pseudorandom sequence → $C_1 \oplus C_2 = P_1 \oplus P_2$

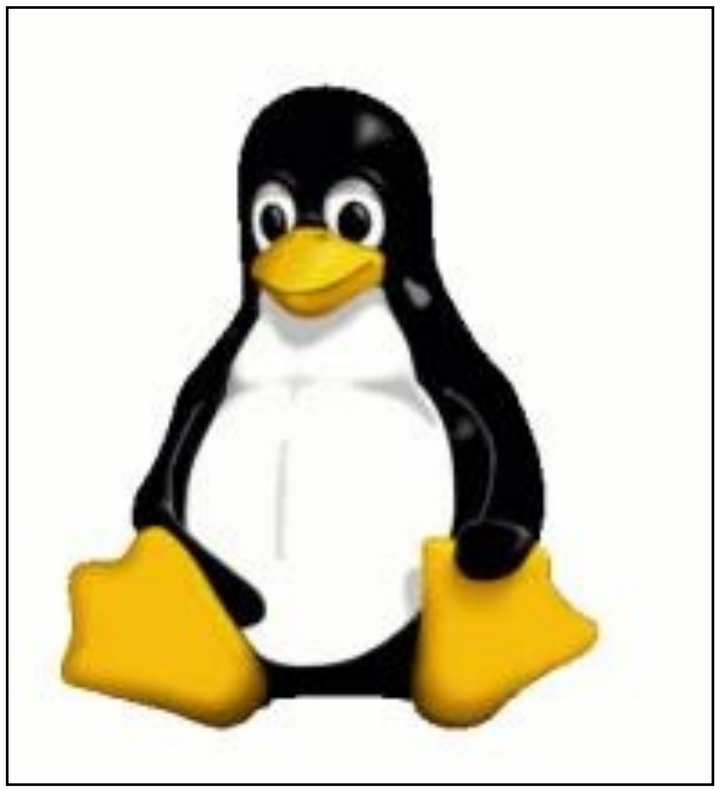
Encrypting Multiple Blocks

- Simplest approach: **encrypt each block independently**

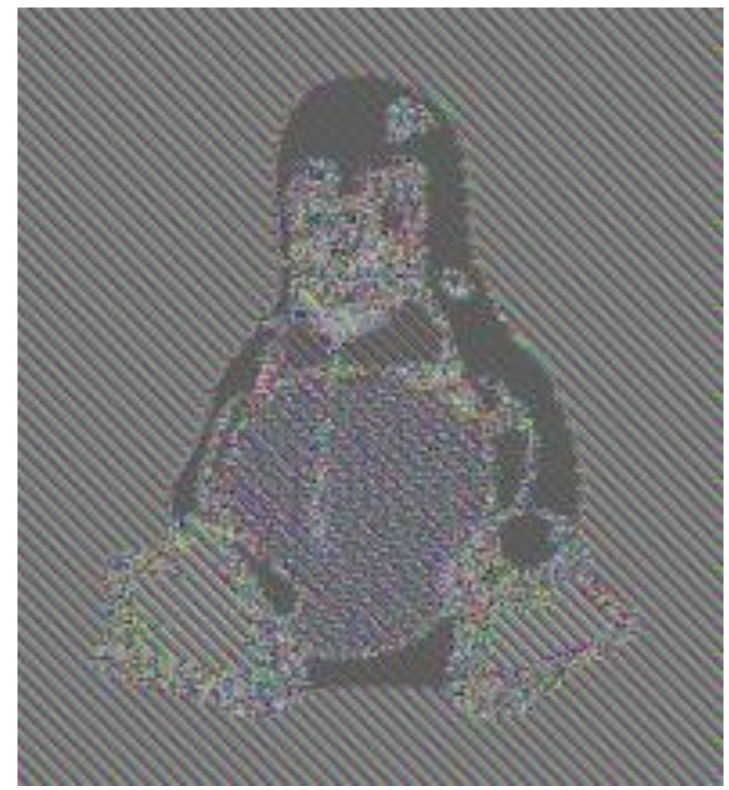


- secure encryption is indistinguishable from random permutation to the attacker
  → if $\mathbf{P_1 \neq P_2}$, then $\mathbf{C_1}$ and $\mathbf{C_2}$ look like unrelated random blocks
- encryption is invertible
  → if $\mathbf{P_1 = P_2}$, then $\mathbf{C_1 = C_2}$

Repeating Blocks



Plaintext
(bitmap)



Ciphertext

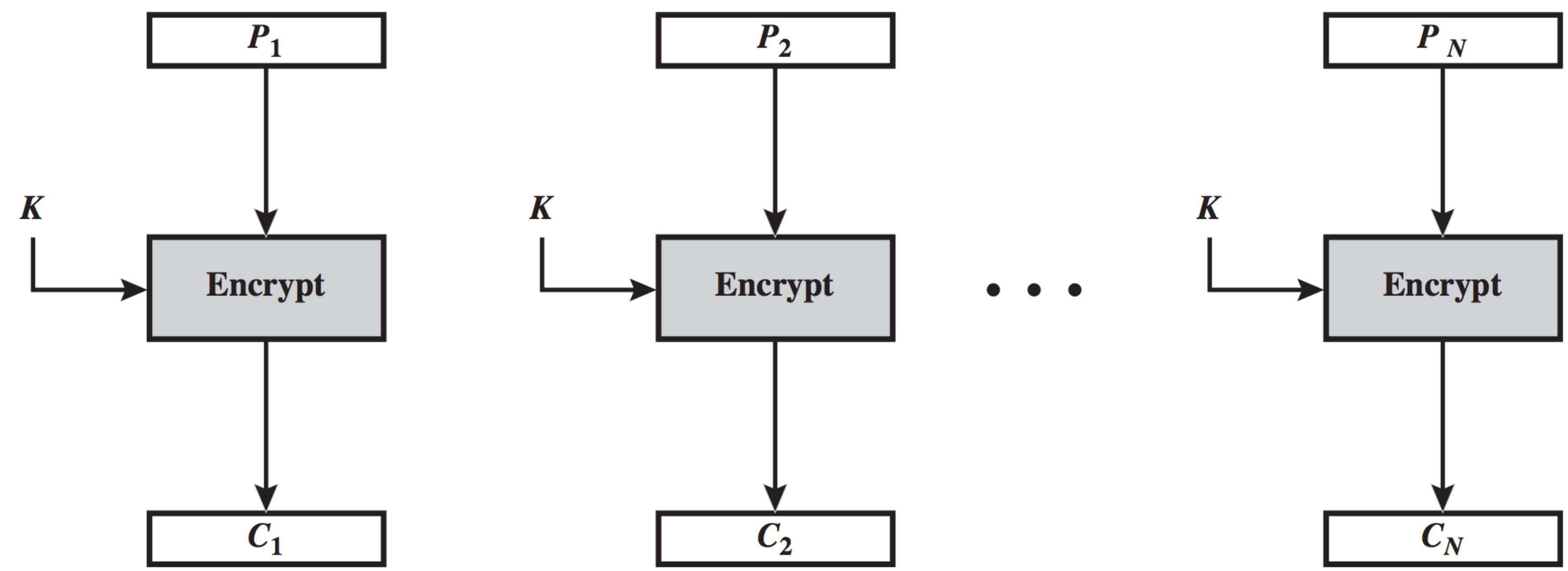
- In practice, many protocols / file formats have predefined headers and elements → repeating blocks

Block Cipher Modes of Operation

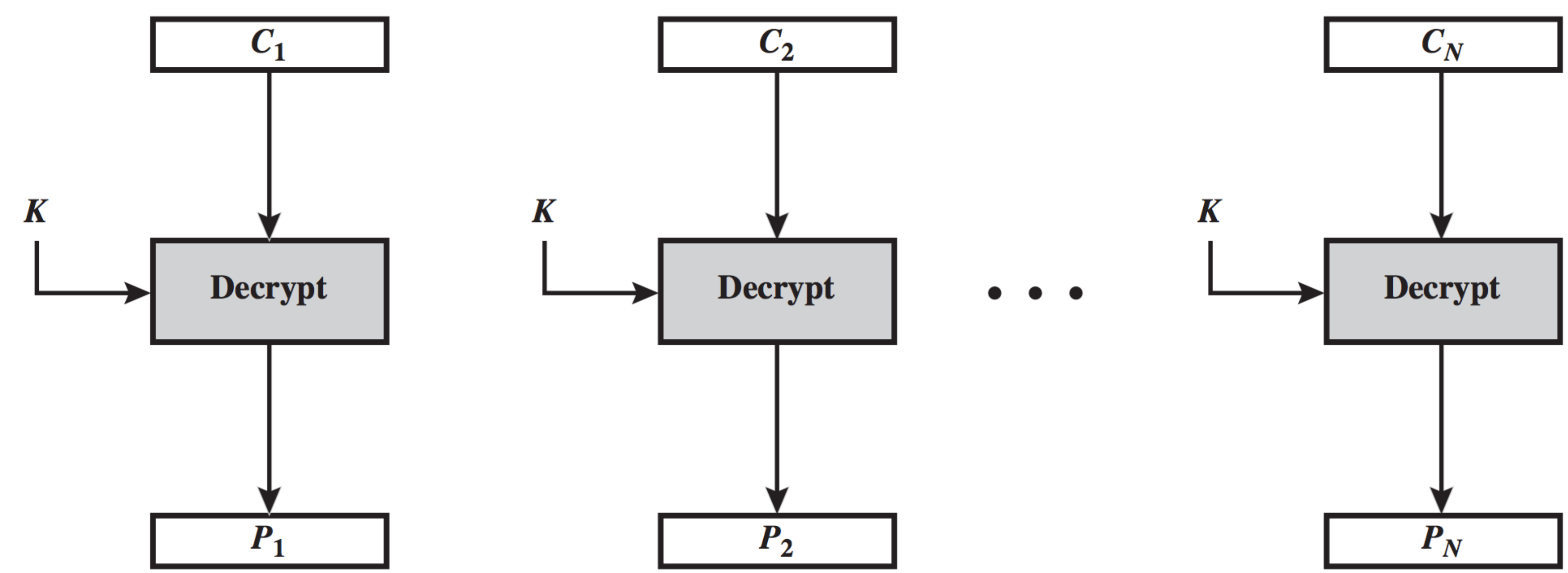
- Mode of operation:
  a technique for **enhancing the effect of a cryptographic algorithm or adapting the algorithm for an application** (*e.g.*, applying a block cipher to a sequence of blocks)
- Five **standard modes of operation** (NIST Special Publication 800-38A)
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Output Feedback (OFB)
  - Cipher Feedback (CFB)
  - Counter Mode (CTR)
- These modes can be used with any block cipher (*e.g.*, DES, AES)
- Criteria: **security**, **efficiency**, **integrity** (error recovery/propagation)

Electronic Code Book (ECB)

$$C_i = E(K, P_i)$$

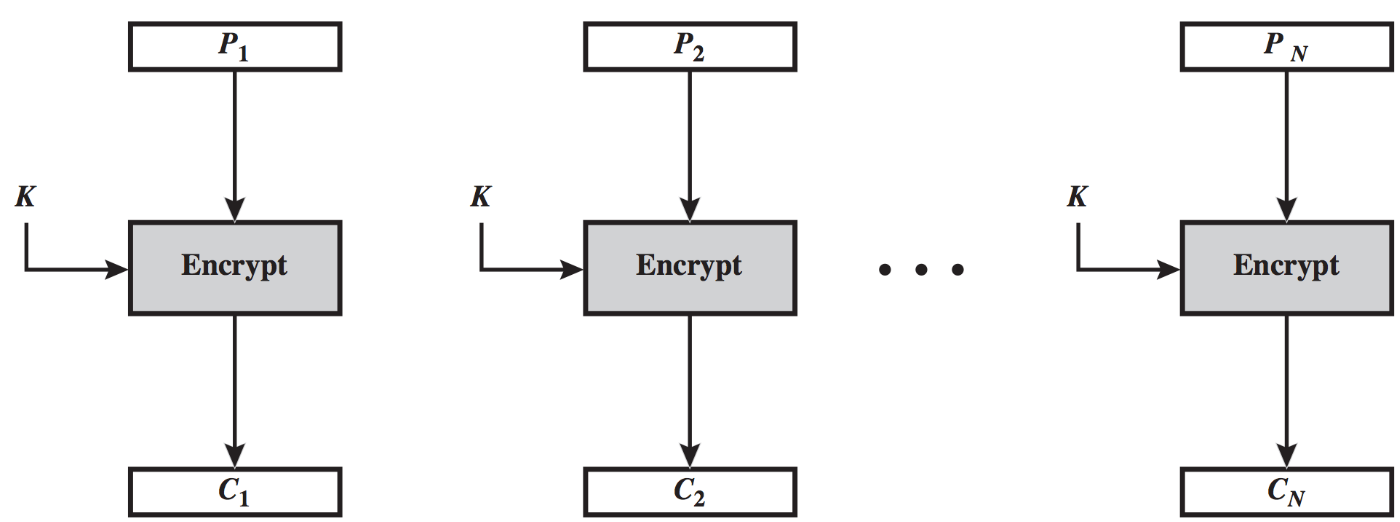


$$P_i = D(K, C_i)$$



Electronic Code Book (ECB)
Details

- Identical plaintext blocks result in identical ciphertext blocks
- Blocks can be encrypted or decrypted in parallel
  - we can start decryption with any block
- Bit error in the ciphertext
  → corresponding plaintext block becomes random
- Attacker can rearrange or remove blocks from the ciphertext
  - additional integrity protection is necessary



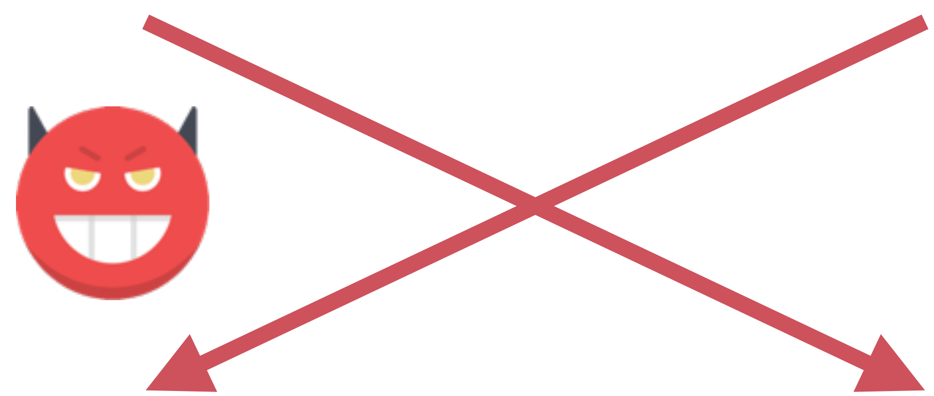
Electronic Code Book (ECB)
Reordering Blocks

Plaintext

Transfer one	million USD to	John Smith's	account from	John Doe's	account.

Ciphertext

dgyACJVKcERNl	z9iIcfkeBEYE2	sp1uELybLi3wm	fq6aSDNIa6wn6	5YRnb75iDRSFx	wFR0yVk1UrIx0



Modified ciphertext

dgyACJVKcERNl	z9iIcfkeBEYE2	5YRnb75iDRSFx	fq6aSDNIa6wn6	sp1uELybLi3wm	wFR0yVk1UrIx0

Modified plaintext

Transfer one	million USD to	John Doe's	account from	John Smith's	account.

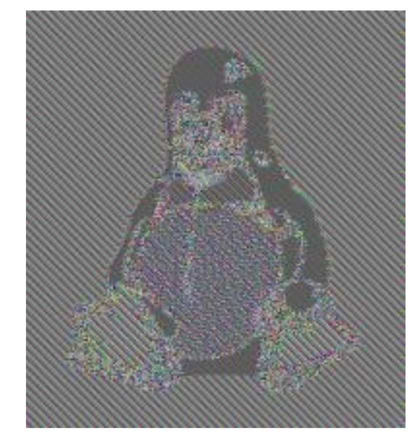
Electronic Code Book (ECB)
Summary

Advantages

- blocks can be encrypted or decrypted in parallel
(i.e., multiple blocks can be encrypted or decrypted at the same time)

Disadvantages

- identical plaintext blocks result in identical ciphertext blocks



- attacker can rearrange or remove blocks from the ciphertext

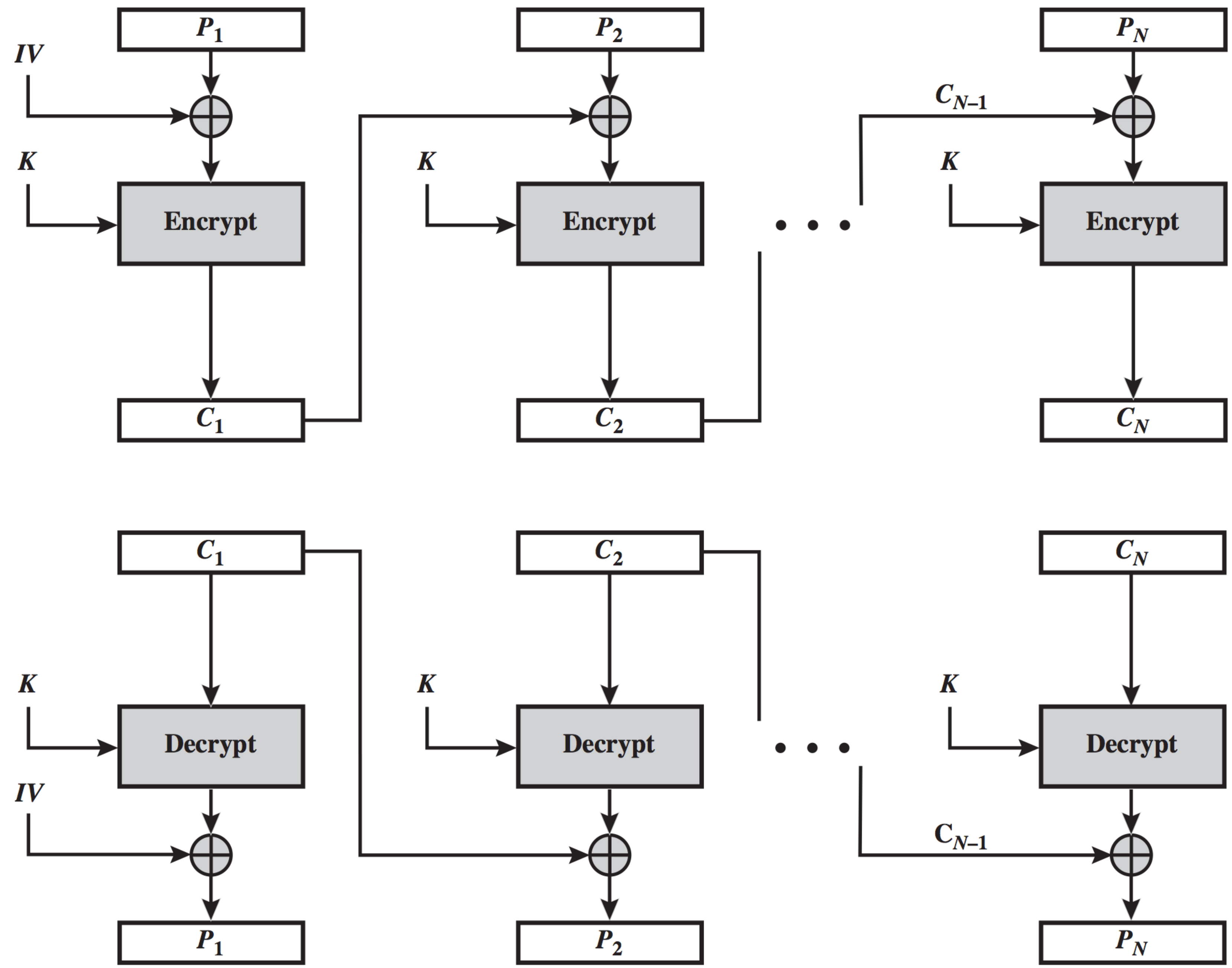
Application: secure transmission of a single block

Cipher Block Chaining (CBC)

initialization vector

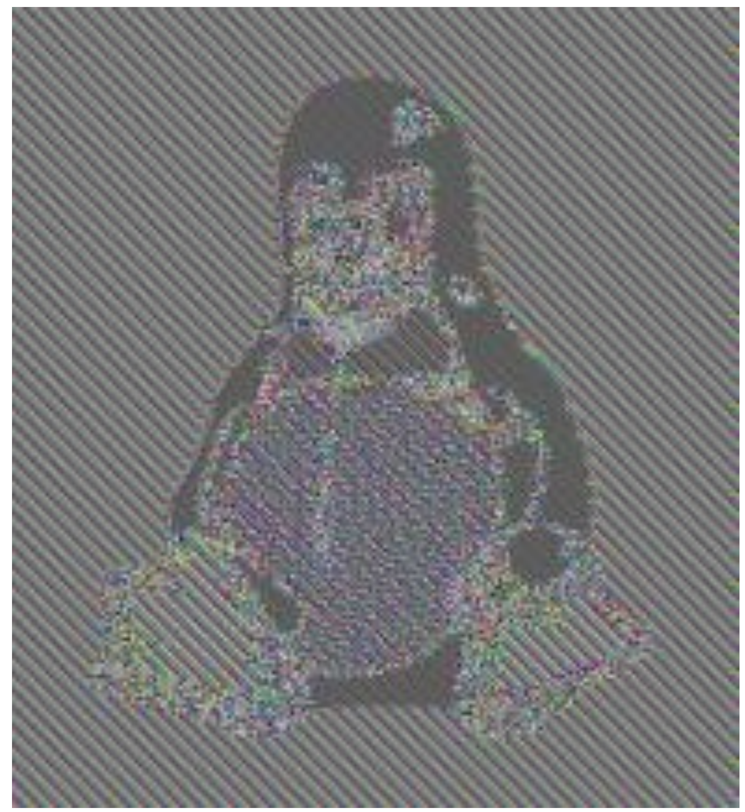
$C_i = E(K, (P_i \oplus C_{i-1}))$

$P_i = D(K, C_i) \oplus C_{i-1}$



Cipher Block Chaining (CBC)
Repetitive Plaintext

Plaintext



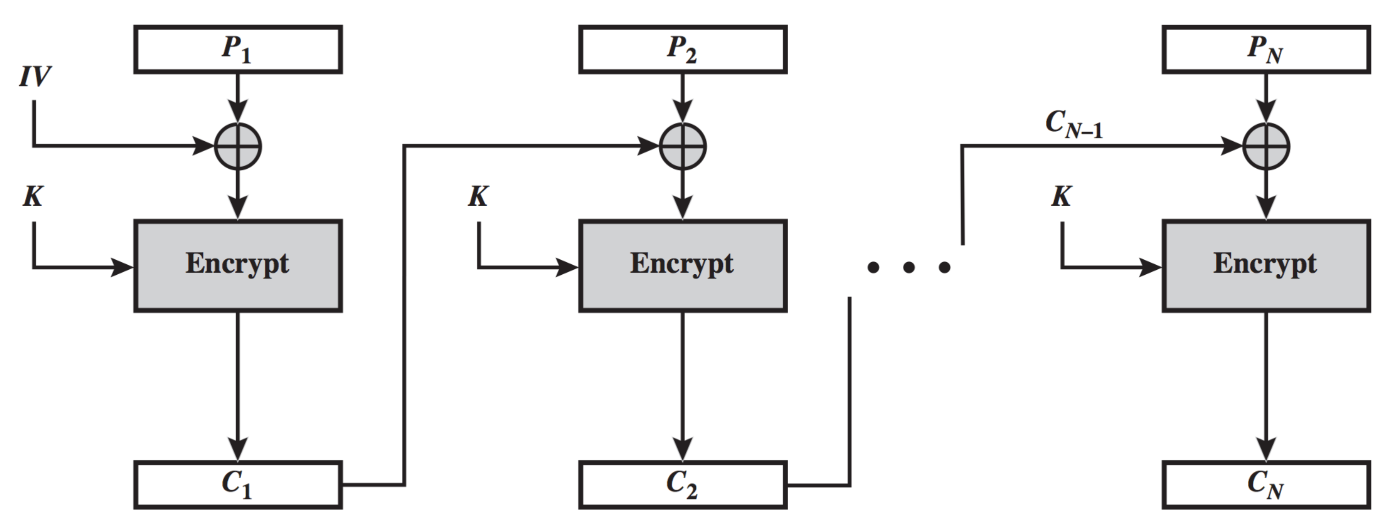
ECB

CBC

Ciphertext

Cipher Block Chaining (CBC)
Details

- Blocks can be decrypted in parallel, but cannot be encrypted in parallel
- Bit error in the ciphertext → corresponding plaintext block becomes random, bit error in the next plaintext block
    - attacker may flip some bits in a plaintext block (but preceding block becomes random)
- Initialization vector (IV) does not have to be secret, but it must be protected
  → if the attacker can change some bits in the IV,
  then the corresponding bits in the first plaintext block change
- Rearranging or removing blocks from the ciphertext may still work



Cipher Block Chaining (CBC)
Cutting and Pasting

Plaintext

`https://www.e`	`xample.com/i`	`ndex.html?pa`	`ssword=secret`

Ciphertext

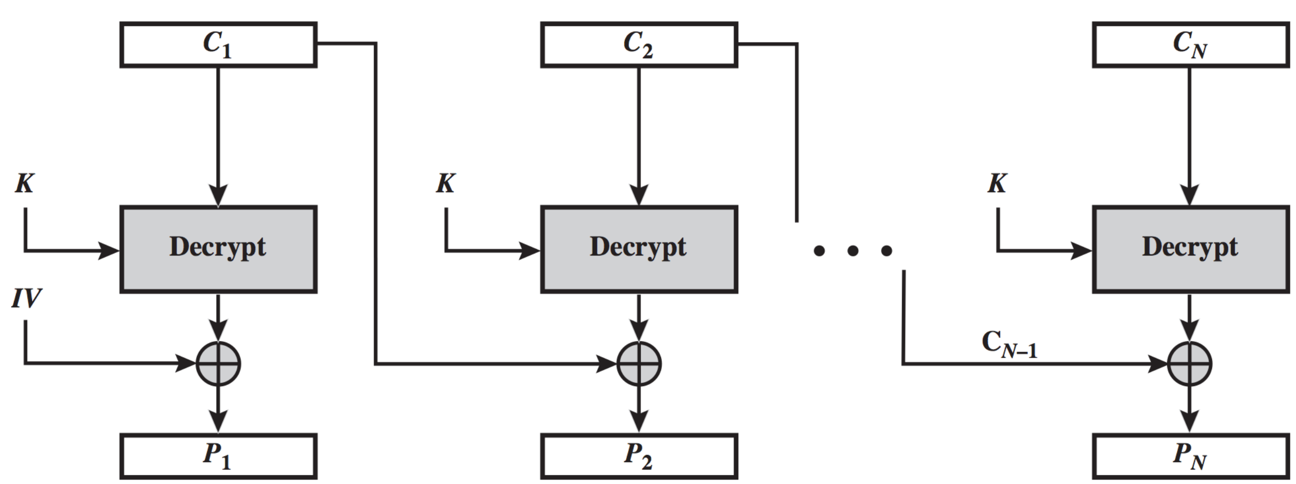
`dgyACJVKcERNl`	`z9iIcfkeBEYE2`	`sp1uELybLi3wm`	`fq6aSDNIa6wn6`

Modified ciphertext

`dgyACJVKcERNl`	`sp1uELybLi3wm`	`fq6aSDNIa6wn6`	`dgyACJVKcERNl`	`z9iIcfkeBEYE2`	`sp1uELybLi3wm`

Modified plaintext

`https://www.e`	`wFR0yVk1UrIx0`	`ssword=secret`	`5YRnb75iDRSFx`	`xample.com/i`	`ndex.htm?pa`



Cipher Block Chaining (CBC)
Summary

Advantages

- hides patterns in the plaintext
- blocks can be decrypted in parallel

Disadvantages

- blocks cannot be encrypted in parallel
- attacker might be able to rearrange or remove blocks from the ciphertext
- IV needs integrity protection
- attacker might be able to tamper with the bits of the plaintext

Application: general-purpose block-oriented transmission

Using Block Ciphers as Stream Ciphers

- Short plaintext (*e.g.*, one bit)
  - if we use the previous two modes (ECB or CBC), we need to **send an entire block** (64 bits for DES and 128 for AES)
  - with stream ciphers, the ciphertext is only as long as the plaintext (*e.g.*, one bit)
- Converting a block cipher into a stream cipher
  - Output Feedback (OFB)
  - Cipher Feedback (CFB)
  - Counter Mode (CTR)
- Stream ciphers always need integrity protection to detect tampering

Stream Ciphers
Changing Bits

Original plaintext:	Y	E	S
binary representation:	01011001	01000101	01010011
Pseudorandom sequence: (example)	11010010	00100000	11110101
Original ciphertext:	10001011	01100101	10100110

Modified ciphertext:	10011100	01101111	11010100

Pseudorandom sequence:	11010010	00100000	11110101
	01001110	01001111	00100001
Modified plaintext:	N	O	!

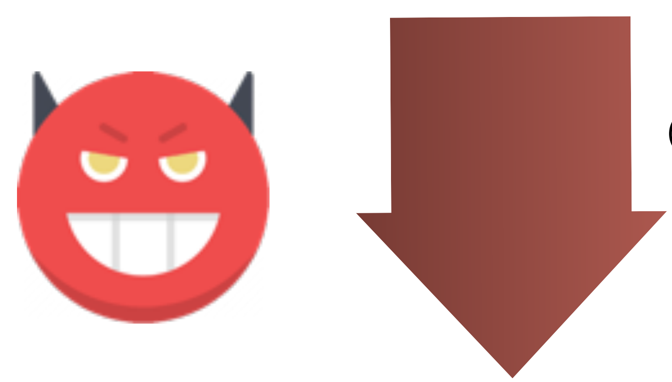
Stream Ciphers
Changing Bits

Plaintext

`Transfer one million dollars to Mr. John Smith's account.`

Ciphertext (example)

`1lDE8aAs7gzUovteKIy6G7yttaacP5pFcGPW3m54Nr4Hepdl7kAjr4kfs`



$\oplus$ ("Smith's" $\oplus$ "Doe's   ")

Modified ciphertext

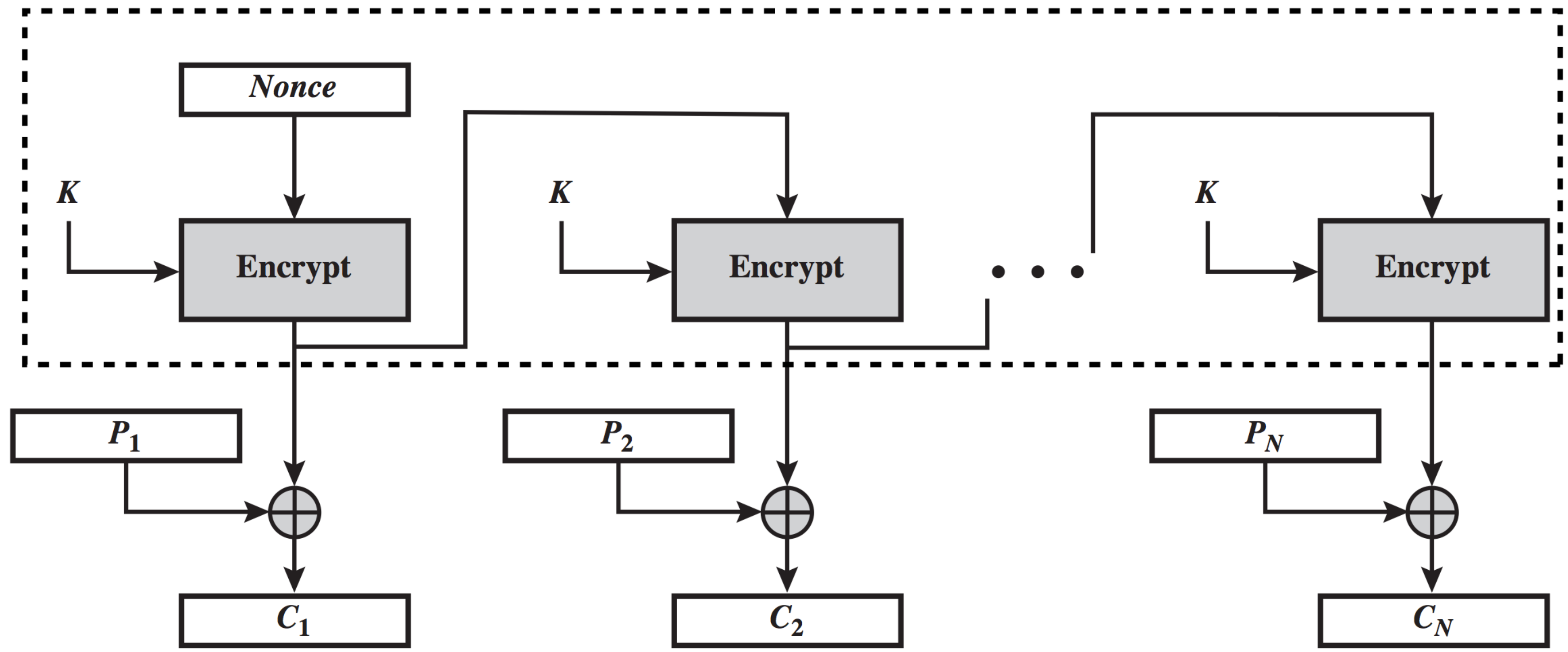
`1lDE8aAs7gzUovteKIy6G7yttaacP5pFcGPW3m54Nypj9xhJ7kAjr4kfs`

Modified plaintext

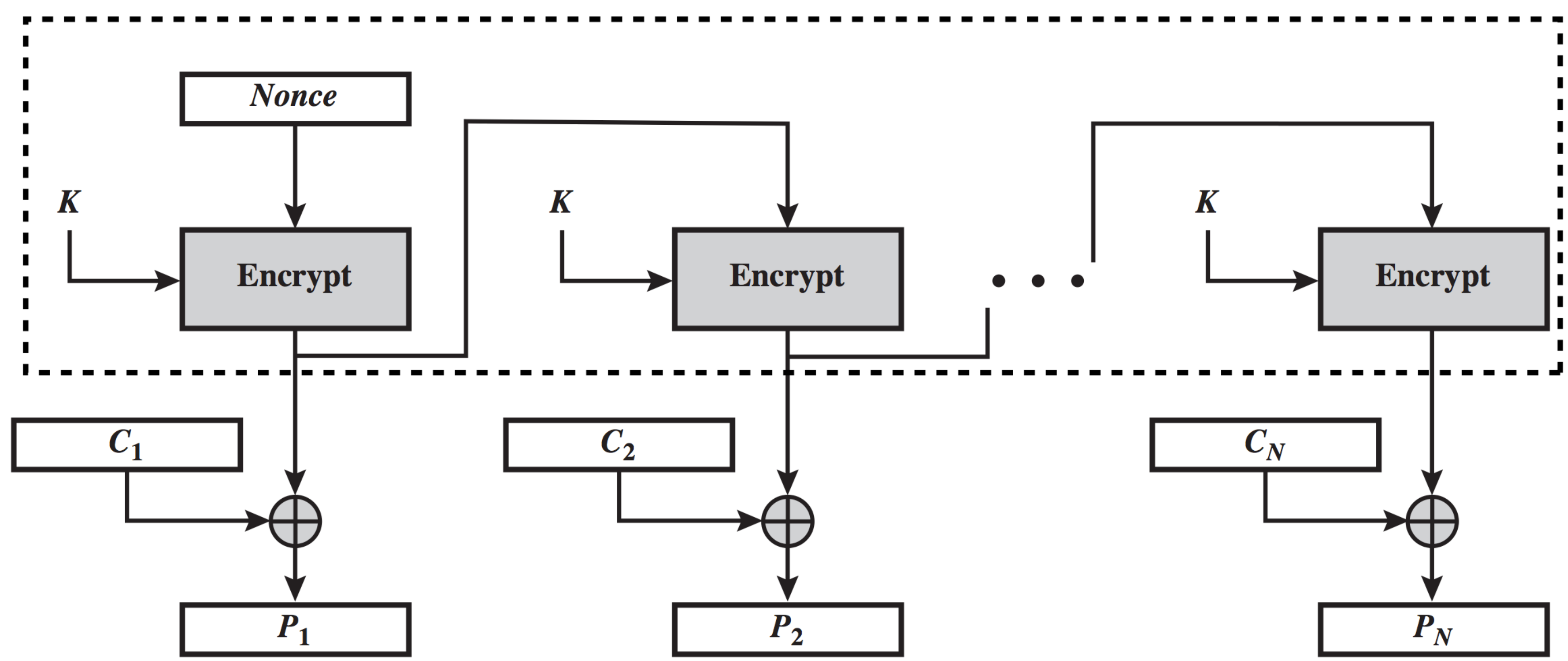
`Transfer one million dollars to Mr. John Doe's   account.`

Output Feedback (OFB)

$O_i = E(K, O_{i-1})$
$C_i = P_i \oplus O_i$

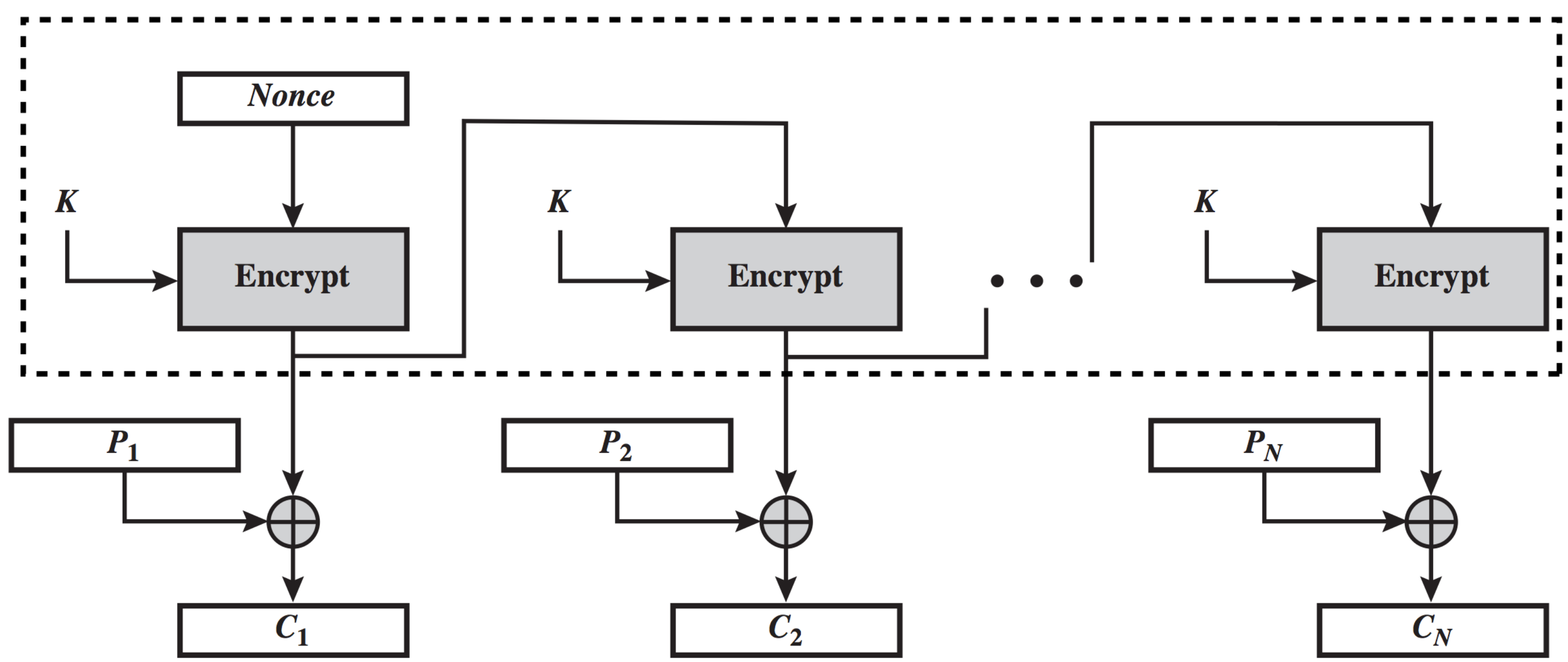


$O_i = E(K, O_{i-1})$
$P_i = C_i \oplus O_i$



Output Feedback (OFB)
Details

- Blocks can be neither encrypted nor decrypted in parallel
  - however, the sequence can be pre-computed
- No “seeking” to arbitrary position in the sequence
- Bit error in the ciphertext → bit error in the corresponding plaintext block
  - attacker can flip bits in a plaintext by flipping the corresponding bits in the ciphertext (without introducing any unwanted changes)



Output Feedback (OFB)
Summary

Advantages

- bit errors do not propagate
- pre-computation is possible

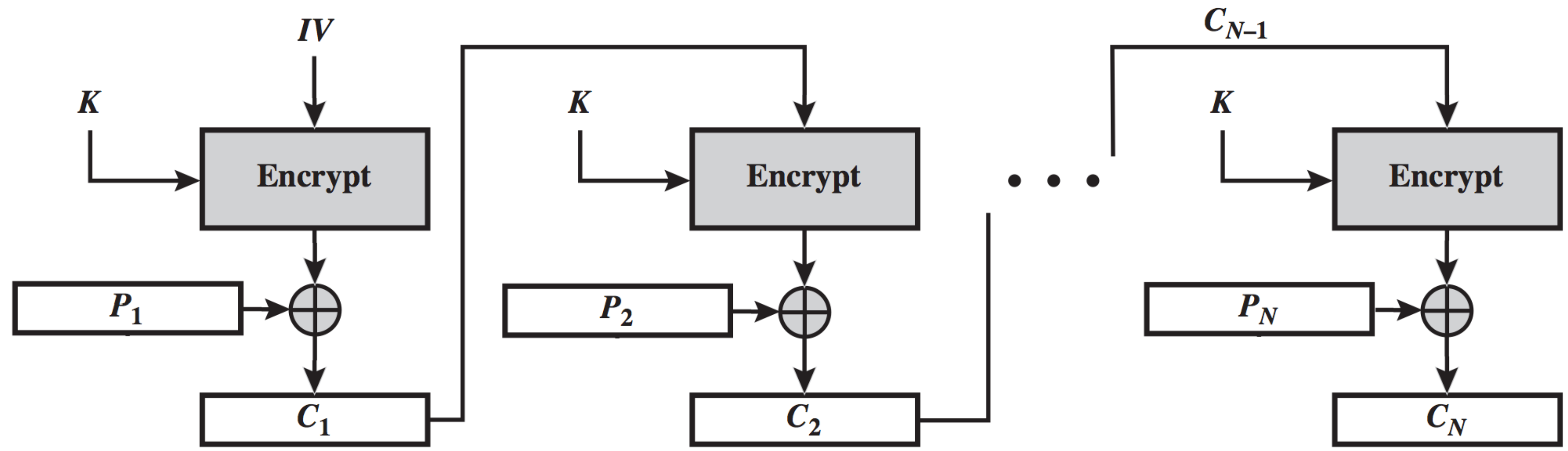
Disadvantages

- blocks cannot be encrypted or decrypted in parallel (unless the sequence is precomputed)
- attacker can tamper with the bits of the plaintext

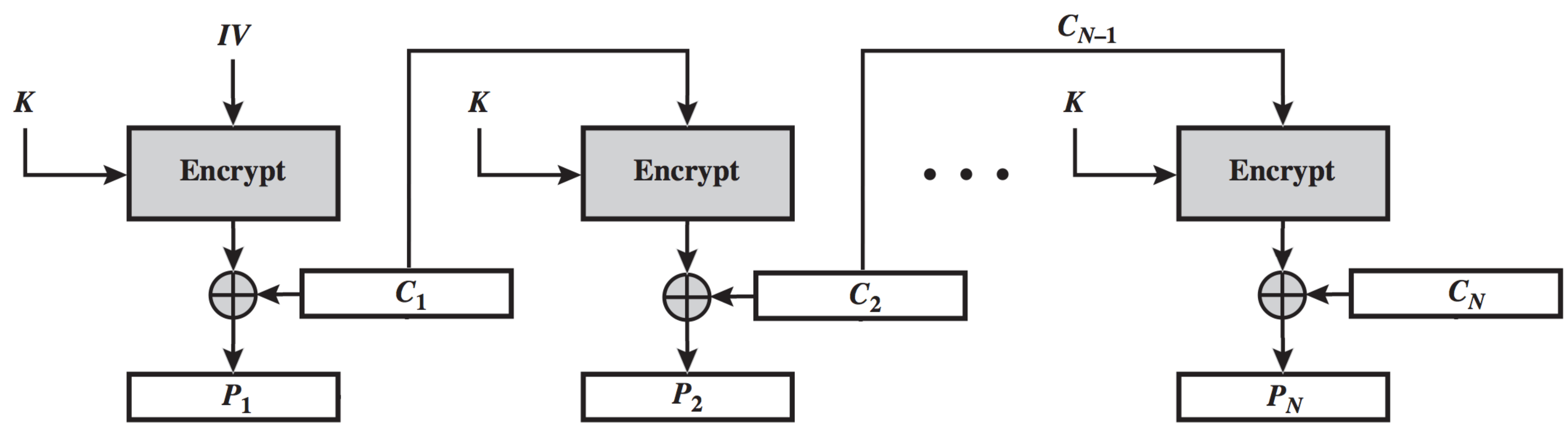
Application: stream-oriented transmission over noisy channel

(Simplified) Cipher Feedback (CFB)

$C_i = P_i \oplus E(K, C_{i-1})$



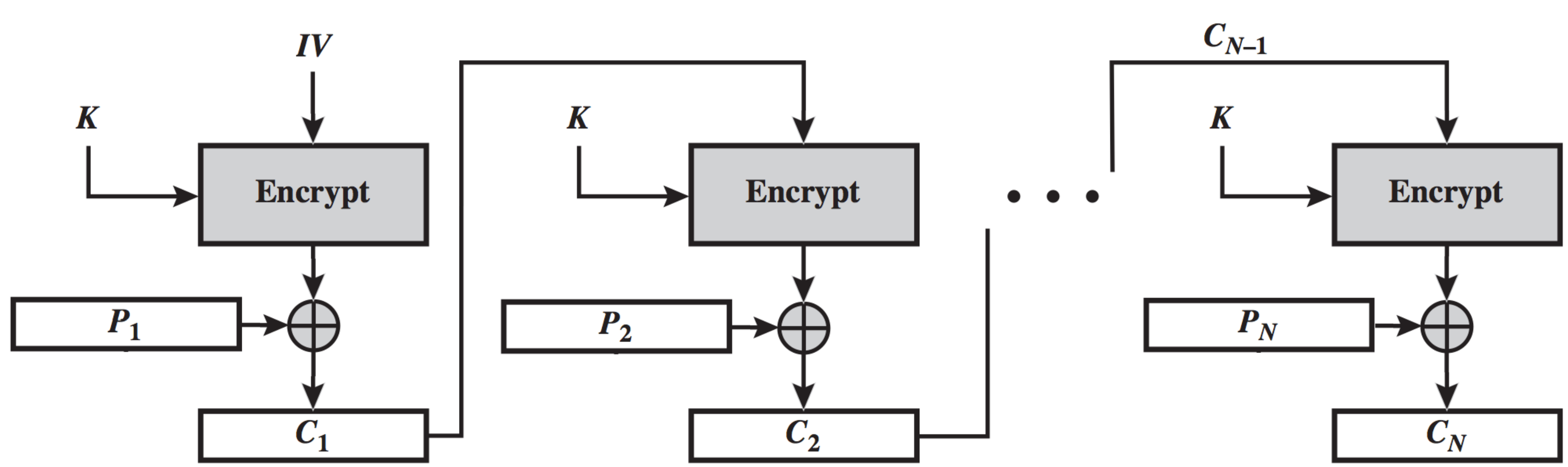
$P_i = E(K, C_{i-1}) \oplus C_i$



For comparison, CBC was: $C_i = E(K, (P_i \oplus C_{i-1}))$, $P_i = D(K, C_i) \oplus C_{i-1}$

Cipher Feedback (CFB)
Details

- Blocks can be decrypted in parallel, but cannot be encrypted in parallel
- Bit error in the ciphertext
  → bit error in the corresponding plaintext block, next plaintext block becomes random
  - attacker may flip some bits in a plaintext block (but the next block becomes random)
- Self-synchronizing: decryption requires only the value of the previous ciphertext block, but not its position in the ciphertext



Cipher Feedback (CFB)
Summary

Advantages

- blocks can be decrypted in parallel
- **self-synchronizing** stream cipher

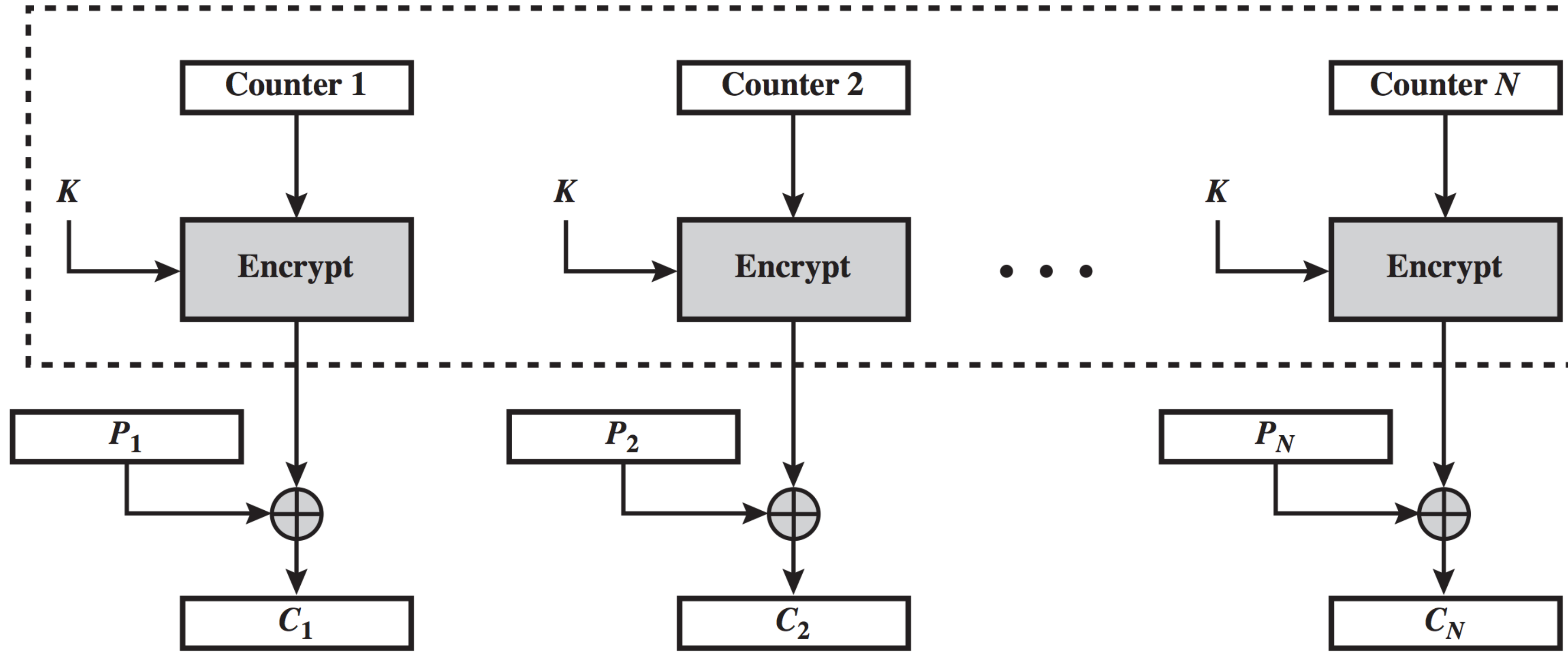
Disadvantages

- blocks cannot be encrypted in parallel
- attacker might be able to tamper with the bits of the plaintext
- attacker might be able to rearrange or remove blocks

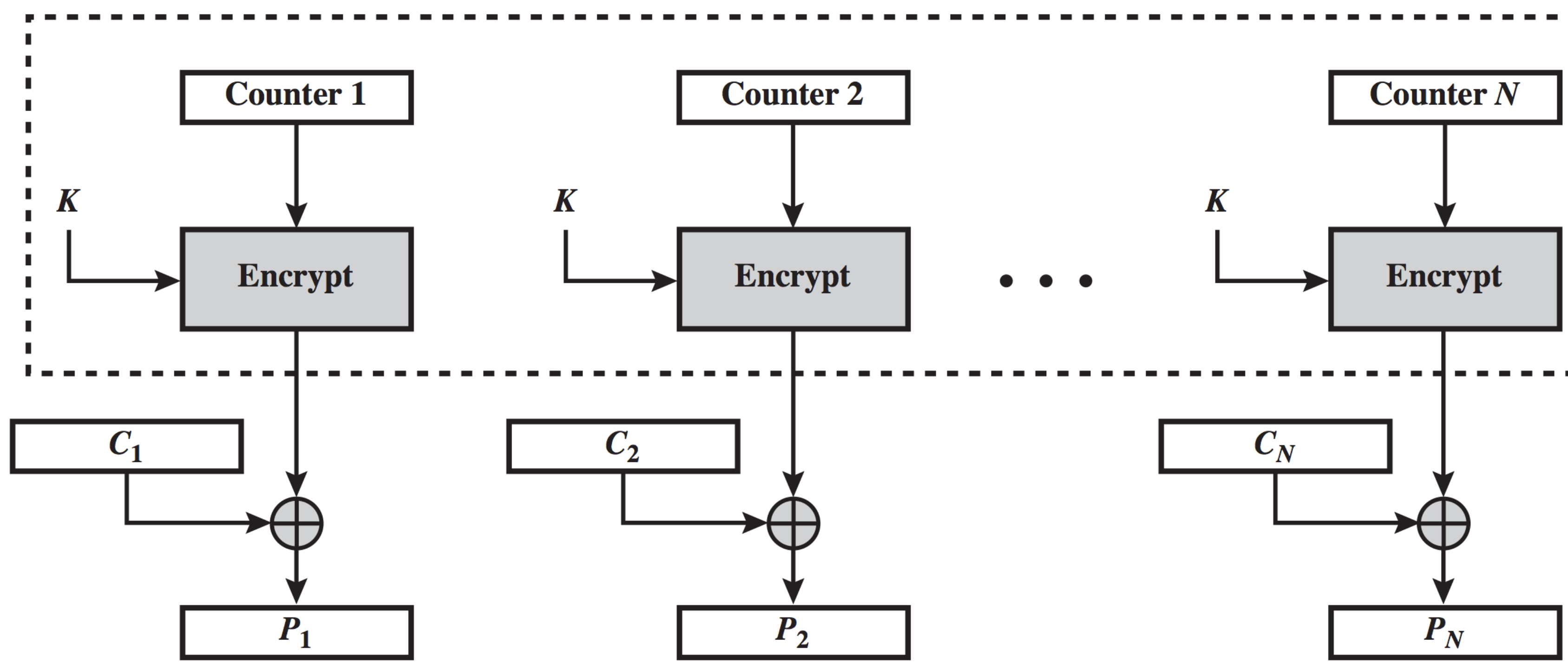
Application: general-purpose stream-oriented transmission

Counter (CTR)

$$C_i = P_i \oplus E(K, T_i)$$

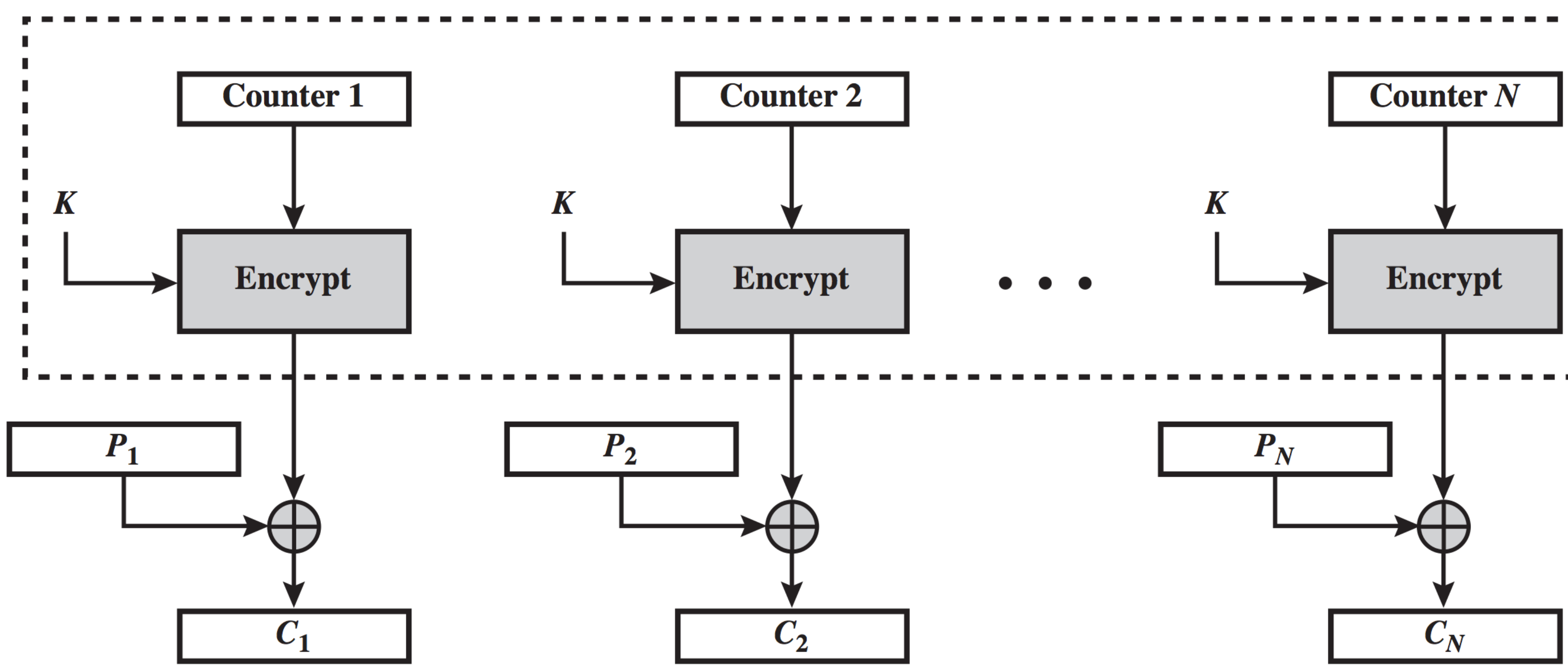


$$P_i = C_i \oplus E(K, T_i)$$



Counter (CTR)
Details

- Counter value must be increased after each block
  - otherwise, we run into the key-reuse problem for stream ciphers
- Blocks can be both encrypted and decrypted in parallel
  - further, the sequence can be precomputed
- Bit error in the ciphertext → bit error in the corresponding plaintext block
  - attacker can flip bits in a plaintext by flipping the corresponding bits in the ciphertext (without introducing any unwanted changes)



Counter (CTR)
Summary

Advantages

- blocks can be encrypted and decrypted in parallel
- bit errors do not propagate
- pre-computation is possible

Disadvantages

- attacker can tamper with the bits of the plaintext

Application: general-purpose transmission

Summary of Standard Block Cipher Modes

- Block-oriented
  - **Electronic Code Book** (ECB): simplest, use only for transmitting a single block
  - **Cipher Block Chaining** (CBC): commonly used
- Stream-oriented
  - Output Feedback (OFB): no random access
  - Cipher Feedback (CFB): self-synchronizing stream cipher
  - **Counter** (CTR): very efficient, very commonly used
- None of these modes provide full integrity protection
  - **authenticated encryption modes:**
    providing confidentiality and integrity protection simultaneously

Next lecture:

Public-Key Encryption