Information Assurance Security (IT352)

This material provides only overview of the topics covered in the class. Kindly refer text book to understand the concept in detail.

Note: Only this material is insufficient for mid-semester and end-semester examinations preparation.

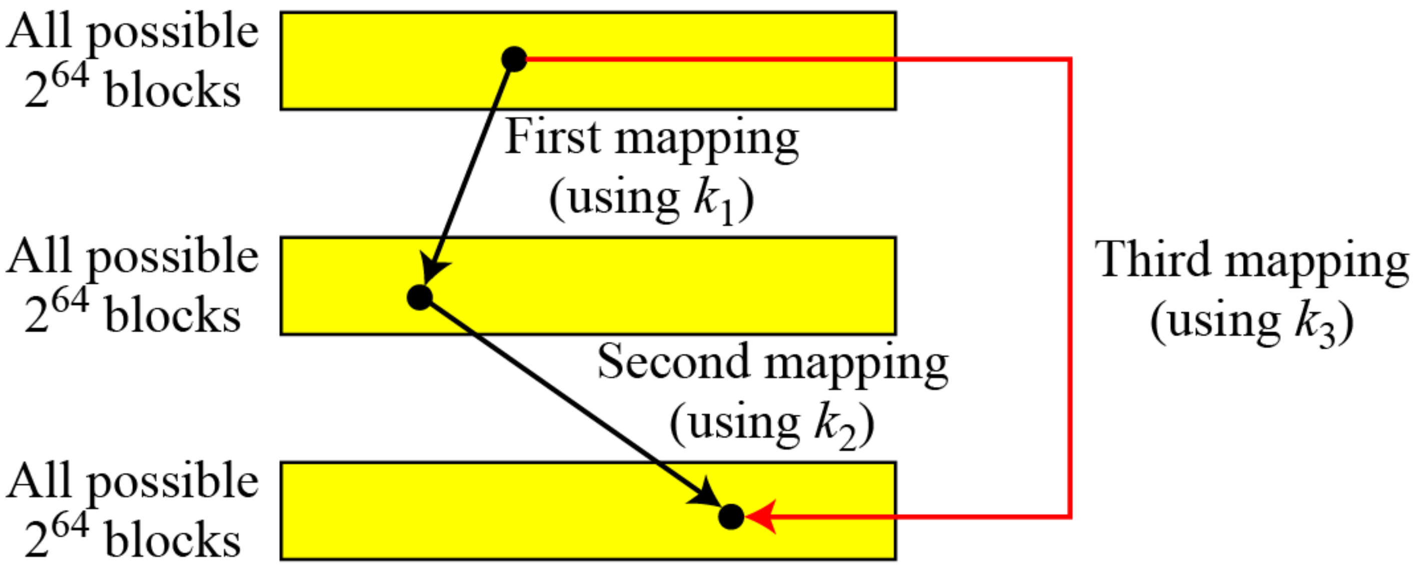
Strength of DES – Key Size

- 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values
- Brute Force search looks hard
- Recent advances have shown is possible
  - in 1997 on Internet in a few months
  - in 1998 on dedicated h/w (EFF) in a few days
  - in 1999 above combined in 22hrs!
- Still must be able to recognize plaintext
- Now considering alternatives to DES

Multiple DES

The major criticism of DES regards its key length. Fortunately DES is not a group. This means that we can use double or triple DES to increase the key size.

A substitution that maps every possible input to every possible output is a group.

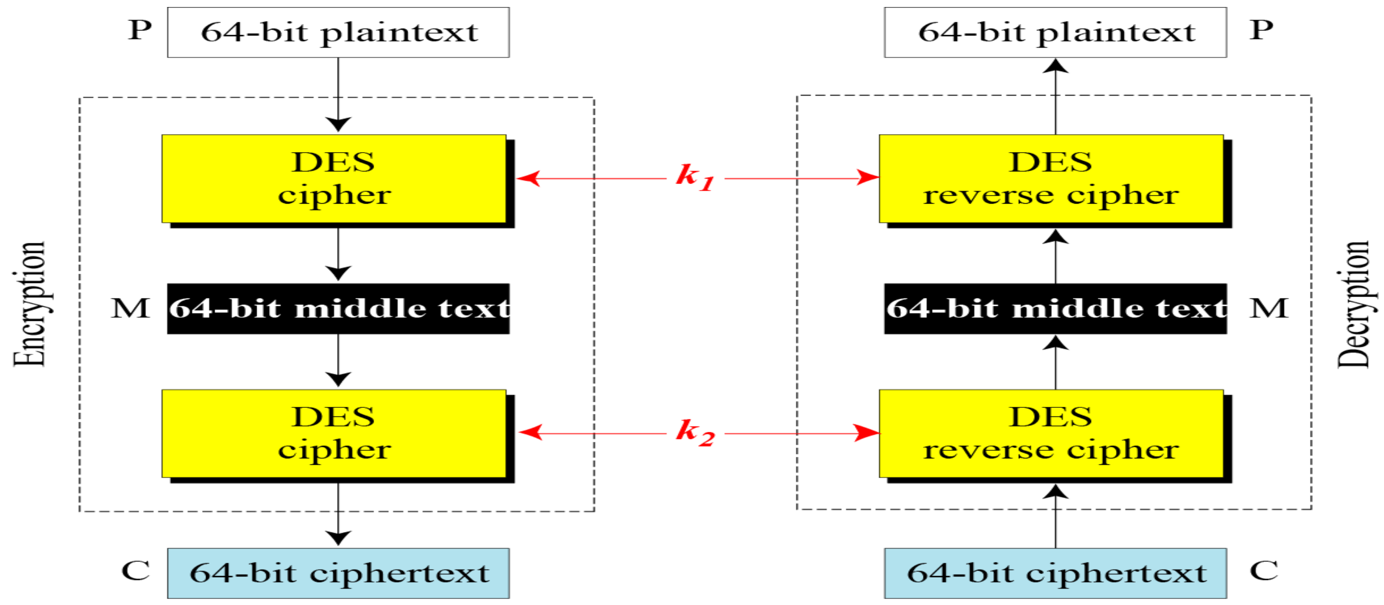

All possible 2^{64} blocks
First mapping (using k_1)
All possible 2^{64} blocks
Second mapping (using k_2)
All possible 2^{64} blocks
Third mapping (using k_3)

Double DES

The first approach is to use double DES (2DES).

Meet-in-the-Middle Attack

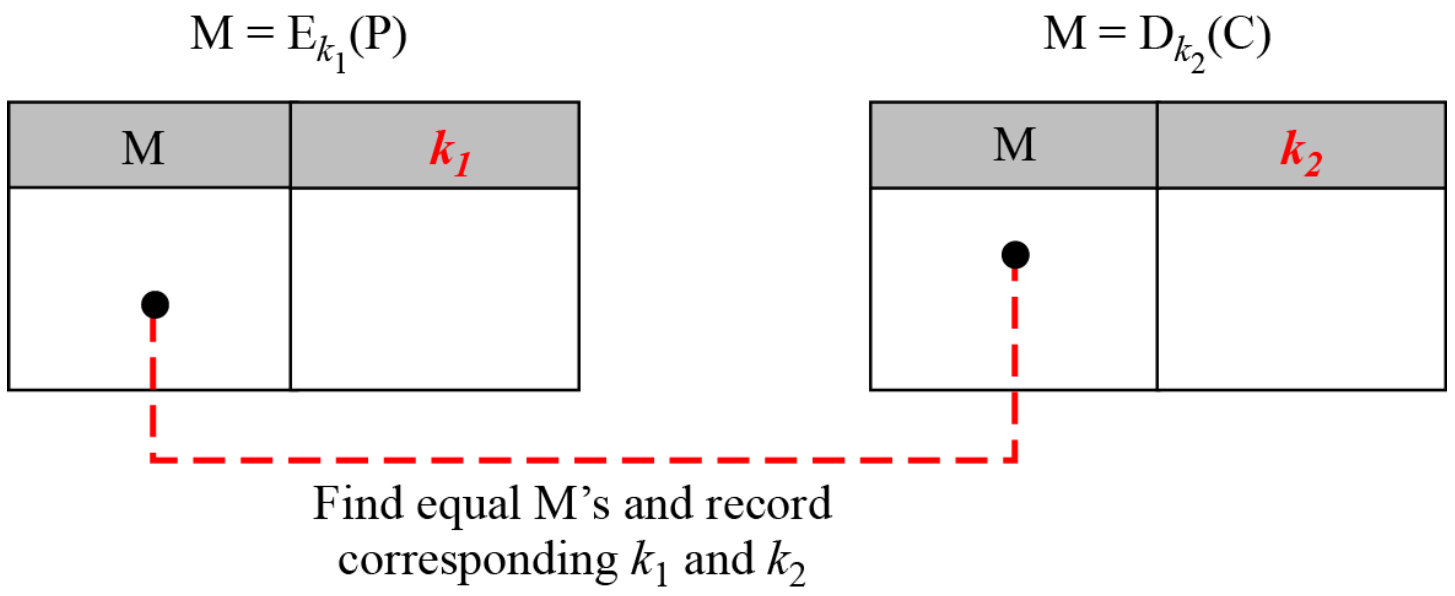
However, using a known-plaintext attack called meet-in-the-middle attack proves that double DES improves this vulnerability slightly (to 2^{57} tests), but not tremendously (to 2^{112}).



Meet-in-the-middle attack for Double DES

Double DES Contd.

Tables for meet-in-the-Middle attack



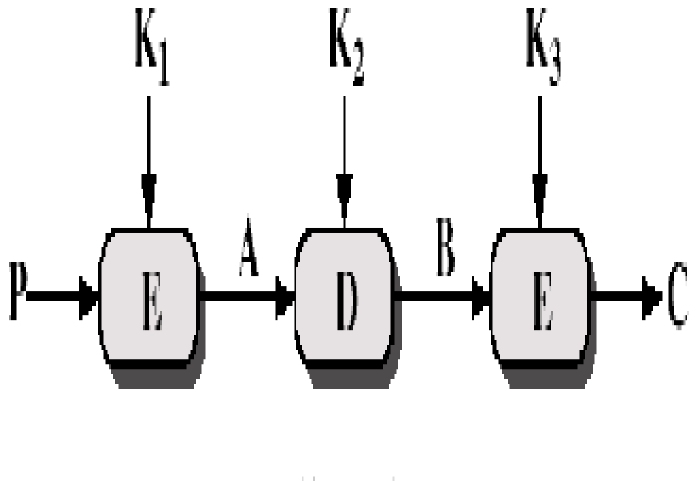
Triple DES Contd.

- Use three keys and three executions of the DES algorithm (encrypt-decrypt-encrypt)

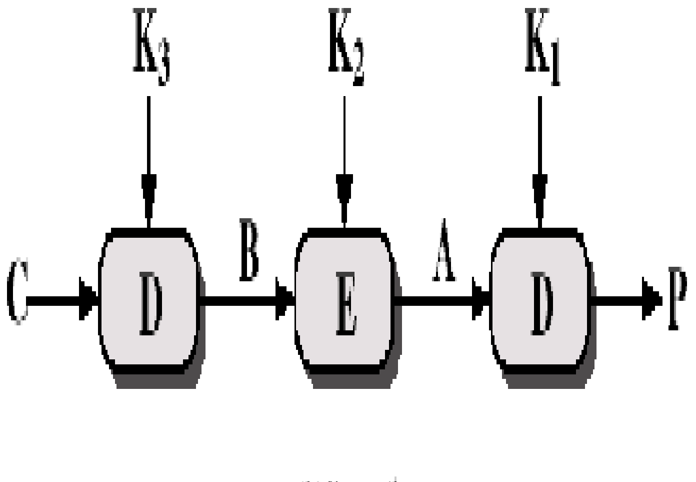
C = EK3[DK2[EK1[P]]]

  - C = Ciphertext
  - P = Plaintext
  - EK[X] = encryption of X using key K
  - DK[Y] = decryption of Y using key K

- Effective key length of 168 bits



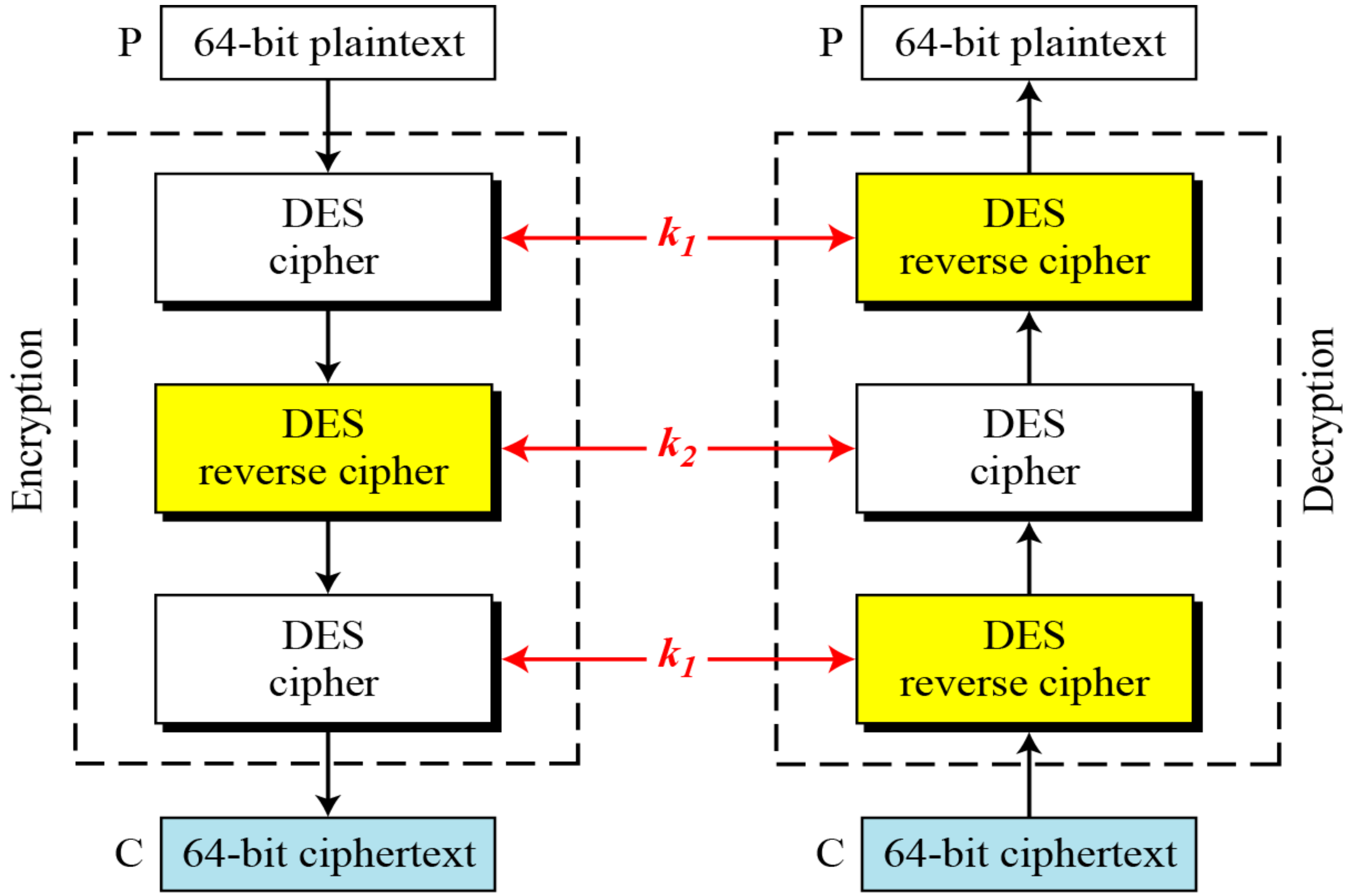
(a) Encryption



(b) Decryption

Triple DES

Triple DES with two keys



Triple DES

- Clear a replacement for DES was needed
  - theoretical attacks that can break it
  - demonstrated exhaustive key search attacks
- AES is a new cipher alternative
- Prior to this alternative was to use multiple encryption with DES implementations
- Triple-DES is the chosen form

Why Triple-DES?

- Why not Double-DES?
  - NOT same as some other single-DES use, but have
- Meet-in-the-Middle attack
  - works whenever use a cipher twice
  - since $X = E_{K1}[P] = D_{K2}[C]$
    - attack by encrypting P with all keys and store
  - then decrypt C with keys and match X value
  - can show takes $O(2^{56})$ steps

Triple-DES with Two-Keys

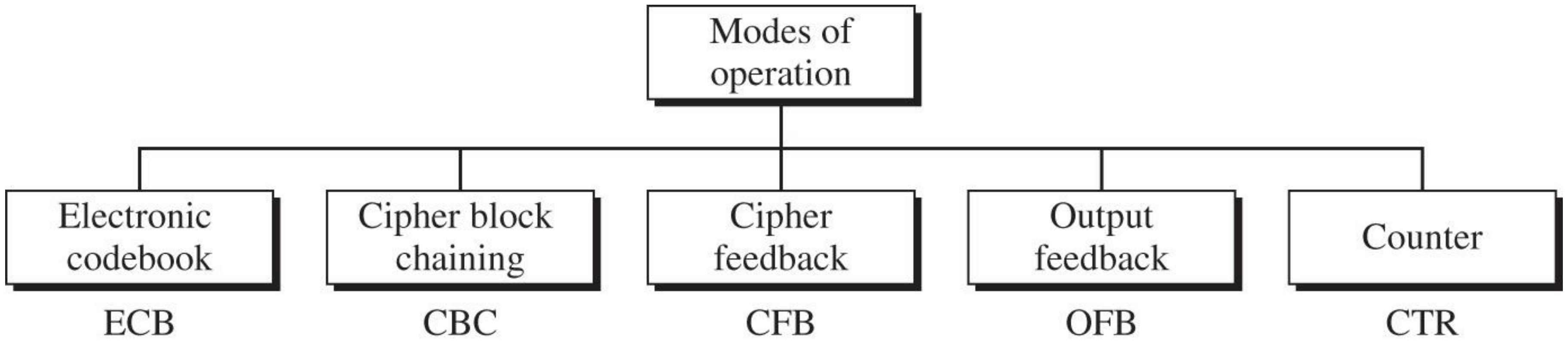
- Hence must use 3 encryptions
  - would seem to need 3 distinct keys
- but can use 2 keys with E-D-E sequence
  - $C = E_{K1}[D_{K2}[E_{K1}[P]]]$
  - nb encrypt & decrypt equivalent in security
  - if K1=K2 then can work with single DES
- Standardized in ANSI X9.17 & ISO8732
- No current known practical attacks

Triple-DES with Three-Keys

- Although are no practical attacks on two-key Triple-DES have some indications
- Can use Triple-DES with Three-Keys to avoid even these
  - $C = E_{K3}[D_{K2}[E_{K1}[P]]]$
- has been adopted by some Internet applications, eg PGP, S/MIME

Cipher Block Modes

- Different ways to transmit data
- Ciphertext depend on something else (besides key) which is different each time
- Some designed to generate ciphertext one byte at a time
- Can be used with any block cipher (DES, AES…)

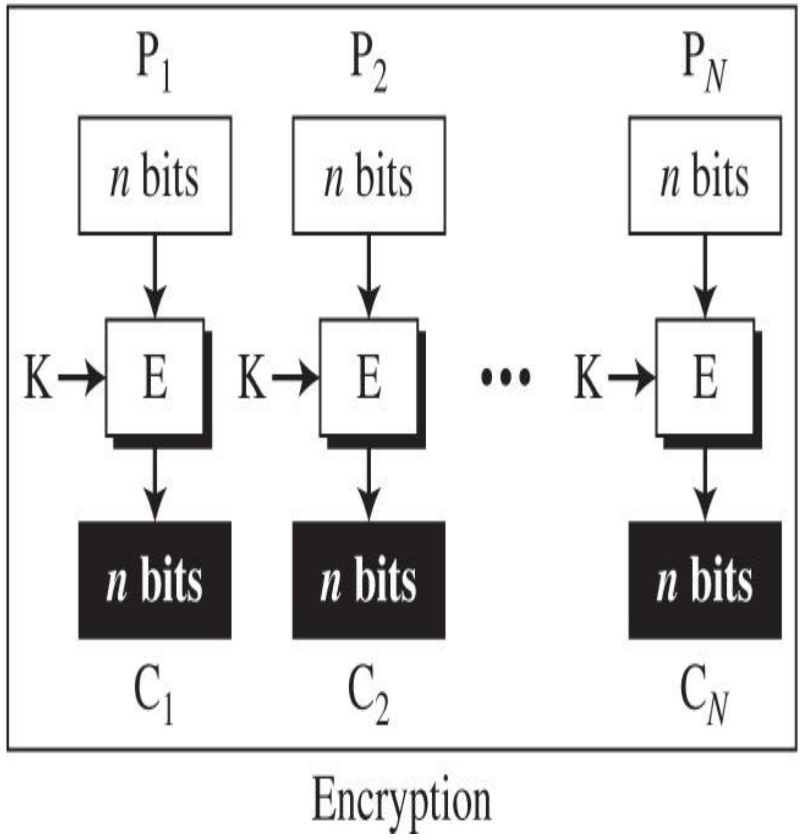


Electronic Codebook Book (ECB)

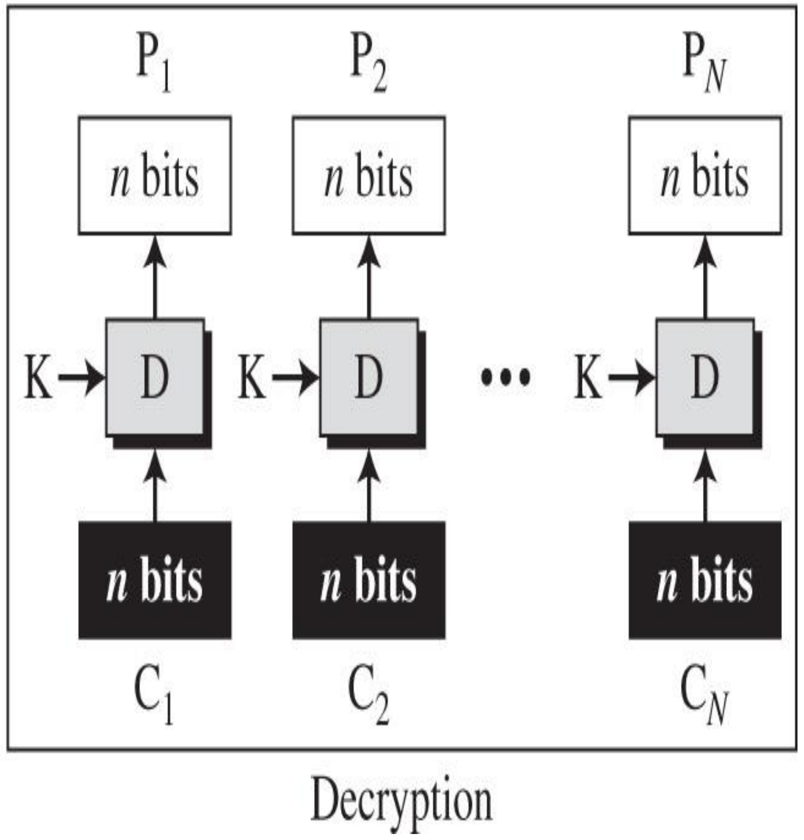
E: Encryption D: Decryption

P_i: Plaintext block i C_i: Ciphertext block i

K: Secret key



Encryption



Decryption

Electronic Codebook Book (ECB) Contd.

- It is the simplest mode of operation. Here, the incoming plain text message is divided into **‘N’** blocks of **‘n’** bits each. Each such block is then encrypted independently of the other blocks. For all blocks in a message, the same key is used for encryption.

- At the receiver’s end, the incoming data is divided into **‘N’** blocks of **‘n’** bits each, and by using the same key as was used for encryption, each block is decrypted to produce the corresponding plain text block.

- In ECB, since a single key is used for encrypting all the blocks of message, if a plain text block repeats in the original message, the corresponding cipher text block will also repeat in the encrypted message. Therefore ECB is suitable only for encrypting small message where the scope for repeating the same plain text blocks is quite less.

Electronic Codebook Mode (ECB) Contd.

Advantages

- Each block can be encrypted/decrypted in parallel
- Noise in one block affects no other block
- Simple

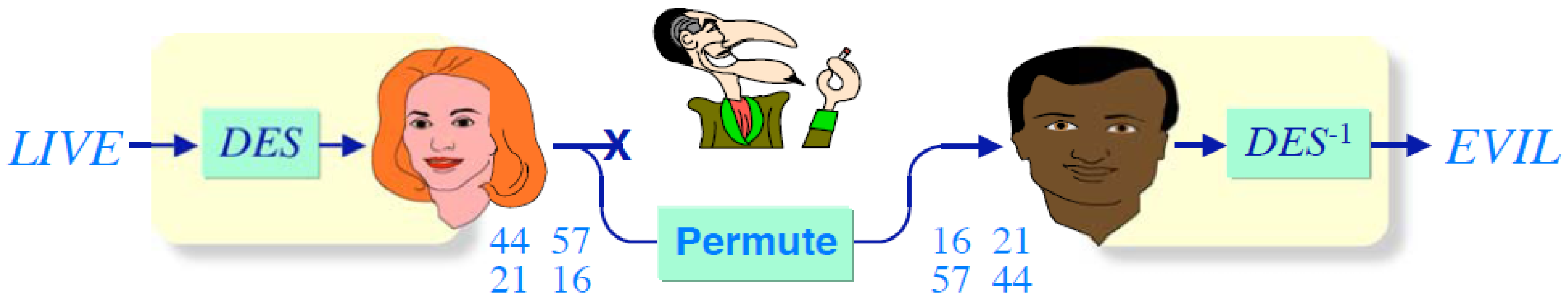
Disadvantage: vulnerable to cryptanalysis

- Repetitive information contained in the plaintext may show in the ciphertext, if aligned with blocks.
- If the same message (e.g., an SSN) is encrypted (with the same key)
- Typical application: secure transmission of short pieces of information (e.g. a temporary encryption key)

Electronic Codebook Mode (ECB) Contd.

Electronic Codebook (ECB) Mode

Misordered blocks attacks



- Alice sends Bob a secret message
  - » Message is LIVE (11 08 21 04)
  - » Message is enciphered as: 44 57 21 16
- Stanley intercepts the message and rearranges the blocks
  - » Now the enciphered message is: 16 21 57 44
  - » (Stanley could also have deleted or replayed blocks)
- Bob receives the message, deciphers it as "EVIL"
  - » How can Bob know if the message is correct?

Cipher Block Chaining (CBC)

- Cipher Block Chaining mode ensures that even if a block of plain text repeats in the input, these two or more identical plain text blocks yield totally different cipher text blocks in the output. For this, a feedback mechanism is used.

- First step receives two inputs: the first block of plain text and a random block of text called as Initialization Vector (IV)

- Remember that the IV is used only in the first plain text block. However, the same key is for encryption of all plain text blocks.

Cipher Block Chaining (CBC) Contd.

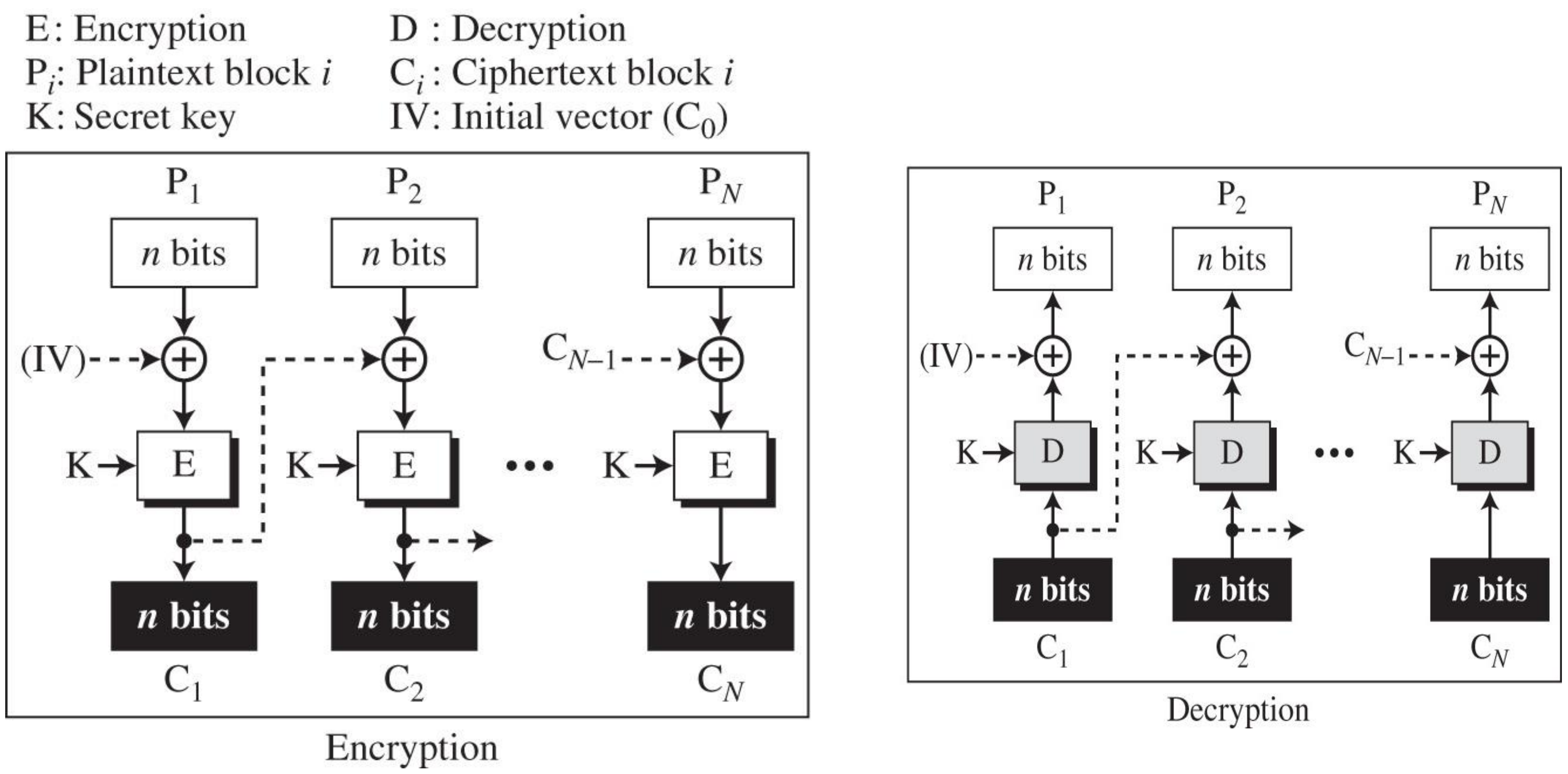
- The plaintext is broken into blocks: $P_1,\ P_2,\ P_3,\ \ldots$
- Each plaintext block is XORed $\left(\text{chained}\right)$ with the previous ciphertext block before encryption (hence the name):

$$C_i \ = \ \mathrm{E}_K\left(C_{i-1} \oplus \ P_i\right)$$

$$C_0 \ = \ \mathrm{IV}$$

- Use an Initial Vector $\left(\mathrm{IV}\right)$ to start the process.
- Decryption: $P_i \ = \ C_{i-1} \ \oplus \ \mathrm{D}_K(C_i)$
- Application: general block-oriented transmission.

Cipher Block Chaining (CBC)



Encryption

$C0 = IV$

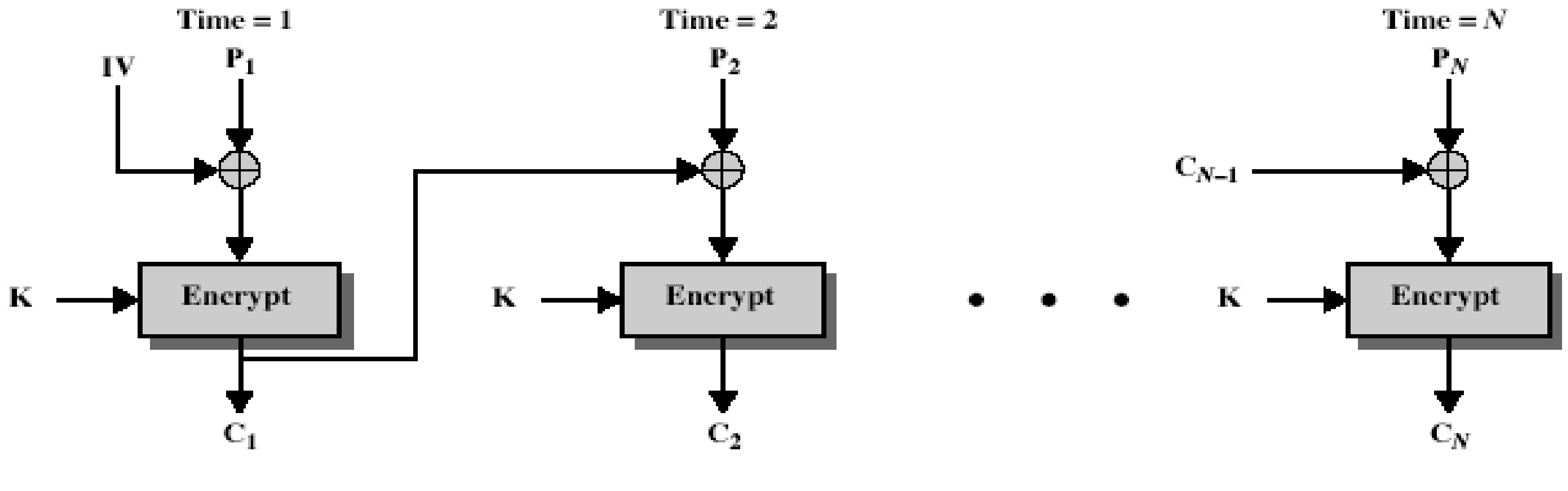
$Ci = E(K, Pi \oplus Ci\text{-}1)$

Decryption

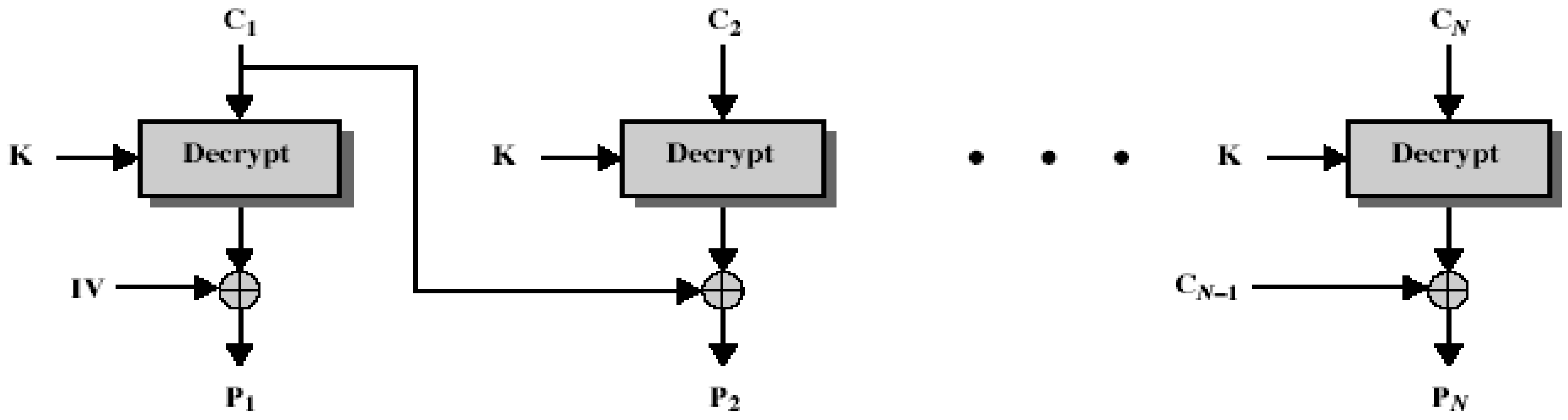
$P0 = D(K, C0) \oplus IV$

$Pi = D(K, Ci) \oplus Ci\text{-}1$

Cipher Block Chaining (CBC)



(a) Encryption



(b) Decryption

Advantages and Limitations of CBC

- So, repeated plaintext blocks are encrypted differently.
- A ciphertext block depends on **all** blocks before it

Need **Initialization Vector** (IV).

- Which must be known to Sender & Receiver.

- If sent in clear, attacker can change bits of first block, and change IV to compensate.

- Hence IV must either be a fixed value OR must be sent encrypted in ECB mode before rest of message.

Problem No. 1

Consider a sensor X that periodically sends a 64-octet measurement to a receiver Y. One day the administrator decides that X should encrypt the measurement data using DES in CBC mode. How many octets does X now send for each measurement?

Solution to Problem No. 1

- DES takes a 8-octet (64-bit) plaintext block and yields a 8-octet cipherblock.
- CBC requires a 8-octet initialization vector (IV) to be sent along with the cipherblocks.
- So X now sends 64 octets of cipherblocks plus 8 octets of IV, for a total of 72 octets.

Problem No. 2

Recall that a **DES encryption operation takes a 64-bit plaintext block and a 56-bit key and produces a 64-bit ciphertext block.** Recall also that each DES encryption operation itself consists of a number of iterations, which we shall refer to as **basic iterations.**

For the DES encryption in CBC mode of a plaintext message of N 64-bit blocks, obtain the following (in terms of N)

a. Total number of DES encryption operations.
b. Size of the output. Explain briefly.
c. Total number of basic iterations. Explain briefly.

Solution to Problem No. 2

Let the plaintext message be $[m_1, m_2, \ldots, m_N]$.
Its CBC encryption is given by C_j = DES_Encrypt(C_{j-1} XOR m_j) for $j = 1, \ldots, N$, where $C_0 = IV$.

a. The DES-CBC Encryption involves N DES encryption operations.

b. The output is [IV, C1, …, CN], which is (N+1) 64-bit blocks.

c. Each DES encryption operation has

- 16 iterations to transform the plaintext block into the ciphertext block. **So there are 16N of these iterations.**

- 16 iterations to produce the 16 48-bit keys from the 56-bit key. But these 16 iterations need be done only once for the entire message. So the answer is **16N + 16 iterations**. **16N and 32N are also acceptable.**

Problem No. 3

Sensor X periodically sends a 32-octet measurement to a receiver Y (1 octet = 8 bits). One day the administrator decides that X should protect the measurement data by adding a MAC obtained using DES in CBC mode (in the standard way). How many octets does X now send for each measurement?

Solution to Problem No. 3

DES operates on 8-octet (64-bit) data blocks.

CBC requires an IV of the encryption block size, so this too is 8 octets. The MAC consists of the IV and the residue (last cipher block) of DES-CBC encryption.

So X now sends 32-octet measurement plus 8-octet IV plus 8-octet residue: **48 octets total**

Cipher Feed Back (CFB)

- Not all applications can work with blocks of data. Security is also required in applications that are character-oriented. For instance, an operator can be typing keystrokes at terminal, which need to be immediately transmitted across the communications link in a secure manner. In such situations, stream cipher must be used.

- The Cipher Feedback(CFB) mode is useful in such cases. In this mode, data is encrypted in units that are smaller (e.g. they could be of size 8 bits, i.e. the size of a character typed by an operator) than a defined block size (which is usually 64 bits).

Cipher Feedback Mode Contd.

- Let us understand how CFB mode works, assuming that we are dealing with ‘J’ bits at a time. Since CFB is slightly more complicated as compared to the first two Cryptography Modes, we shall study CFB in a step-by-step fashion.

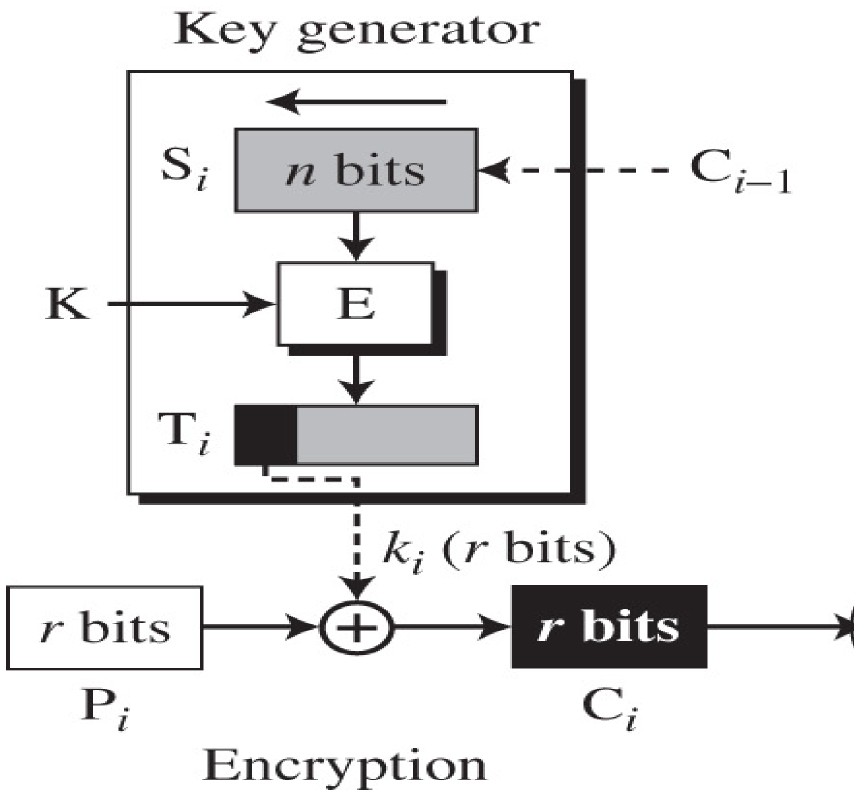
Step 1

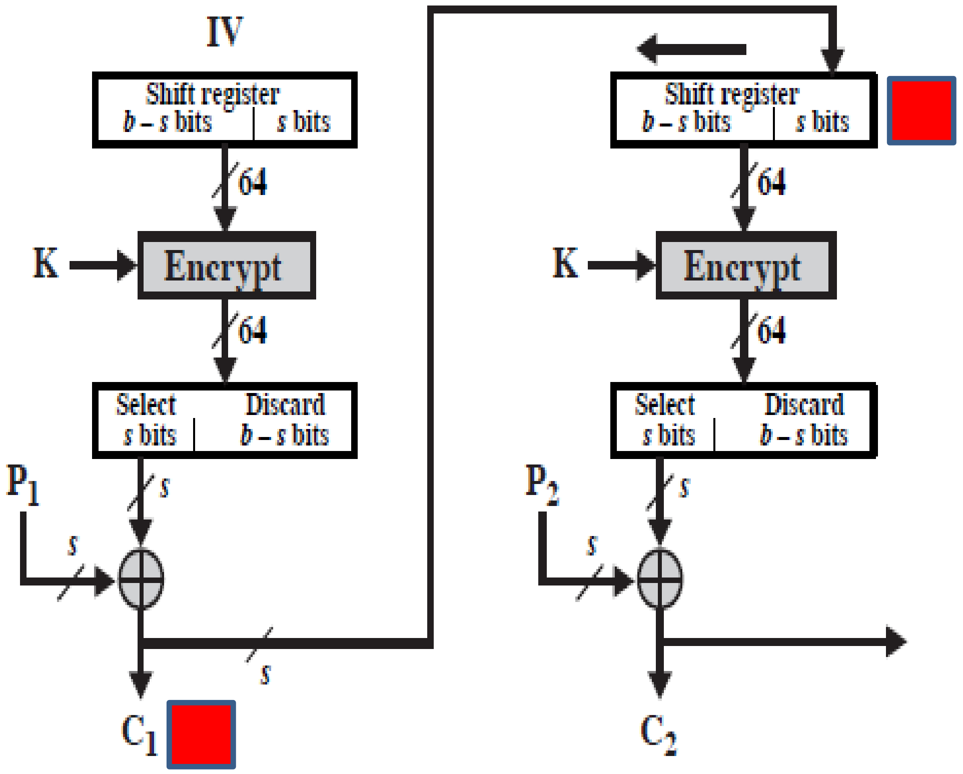
- Like CBC, a 64 bit Initialization Vector (IV) is used in the case of CFB mode. The IV is kept in a shift register. It is encrypted in the first step to produce a corresponding 64-bit IV cipher text.

Cipher Feedback Mode Contd.

- The plaintext is a sequence of segments of s bits (where $s \leq$ block-size): $P_1, P_2, P_3, P_4, \ldots$
- Encryption is used to generate a sequence of keys, each of s bits: $K_1, K_2, K_3, K_4, \ldots$
- The ciphertext is $C_1, C_2, C_3, C_4, \ldots$, where $C_i = P_i \oplus K_i$
- How to generate the key stream?

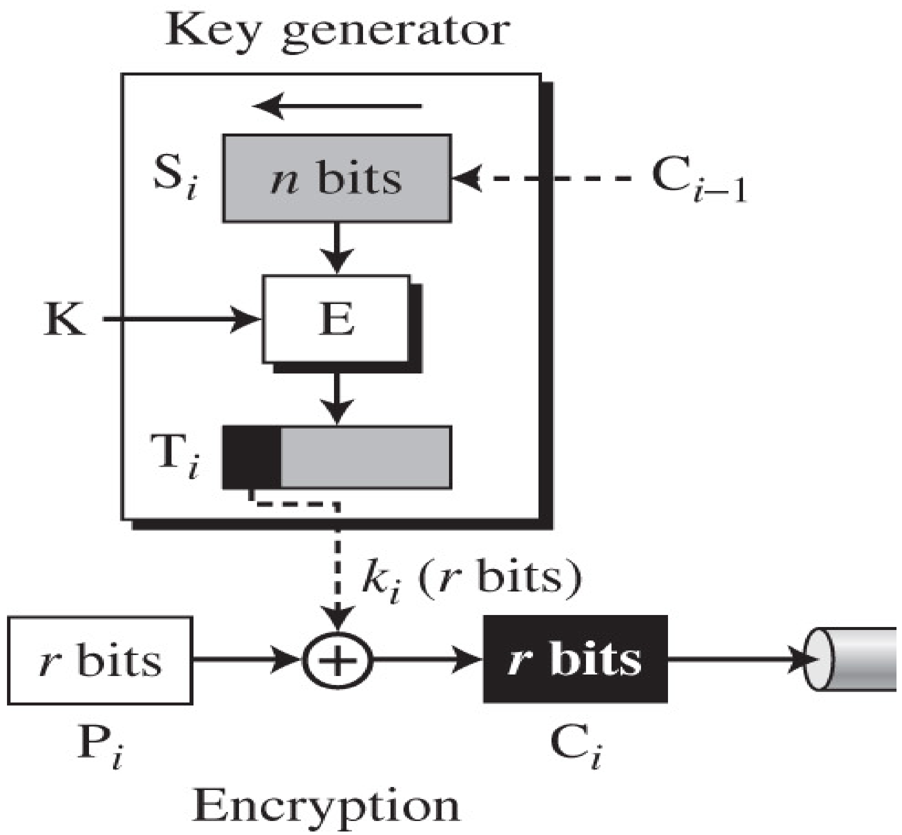
Cipher Feedback Mode Contd.





- Previous ciphertexts used to create shift register **S**
- Shift register contents encrypted with key
- Results placed in “temporary register” **T**

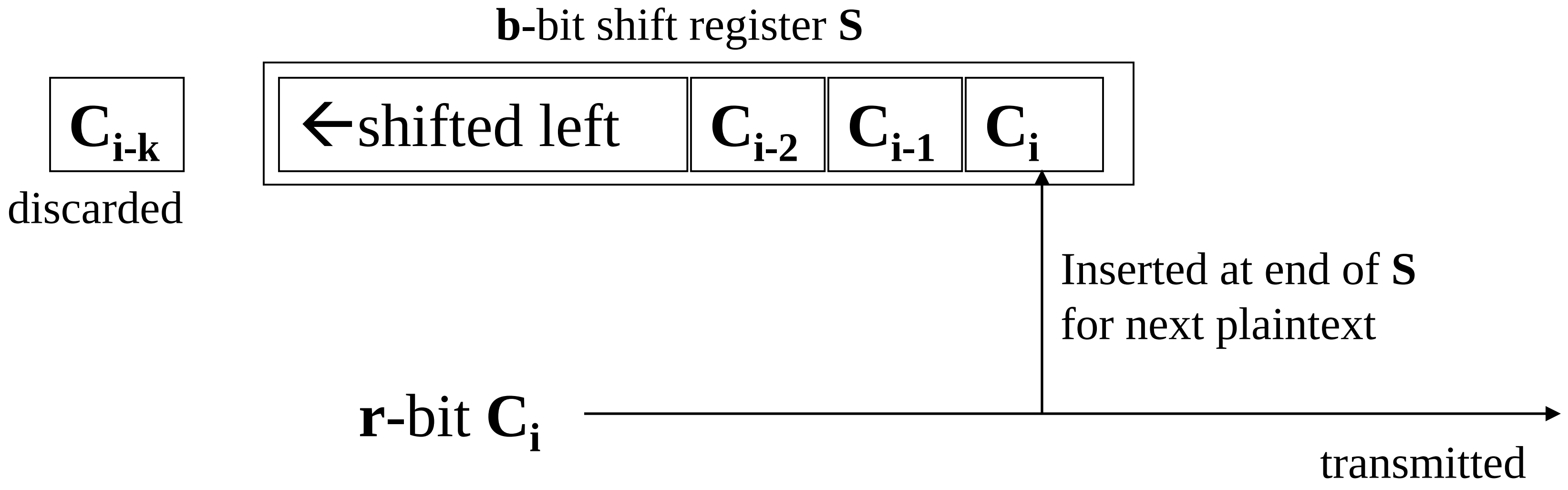
Cipher Feedback Mode Contd.



- First **r** bits of **T** used to create **byte key k_i**
- Byte key_XORed with next **r** bits of plaintext to produces next **r** bits of ciphertext for transmission

Cipher Feedback Mode Contd.

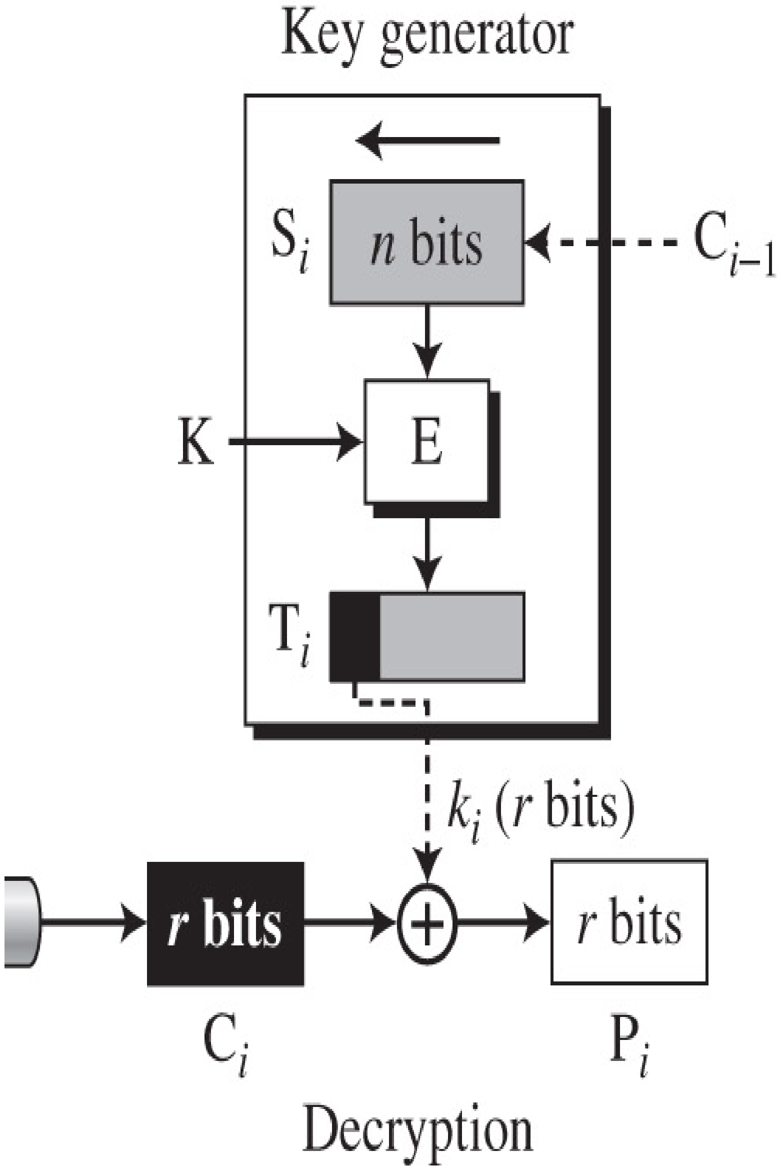
- Previous **r** bits of ciphertext added to end of shift register **S**
  - All other bits in **S** shifted left
  - First **r** bits discarded



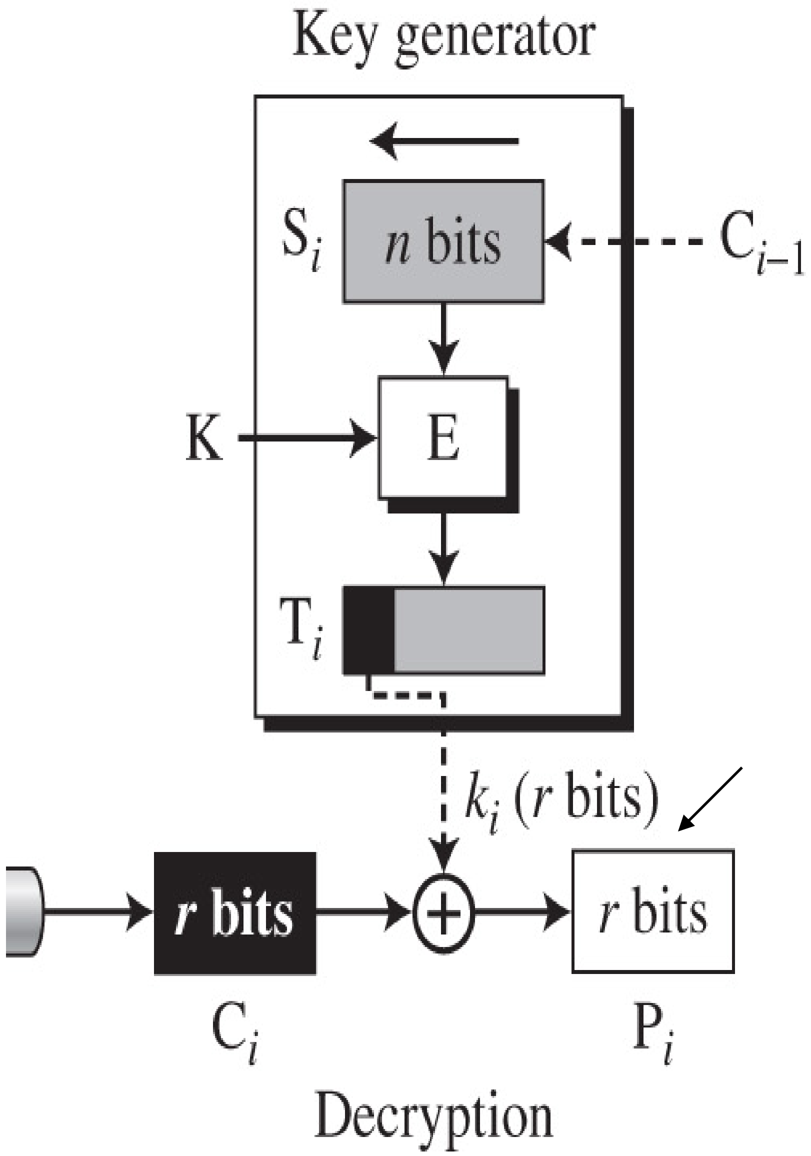
Cipher Feedback Mode Contd.

Decryption:

- Recipient uses previous ciphertext to create same shift register **S**
  - Encrypted with key
  - First **r** bits taken to create byte key $\mathbf{k_i}$
  - XORed with next **r** bits of ciphertext received to get next **r** bits of plaintext.



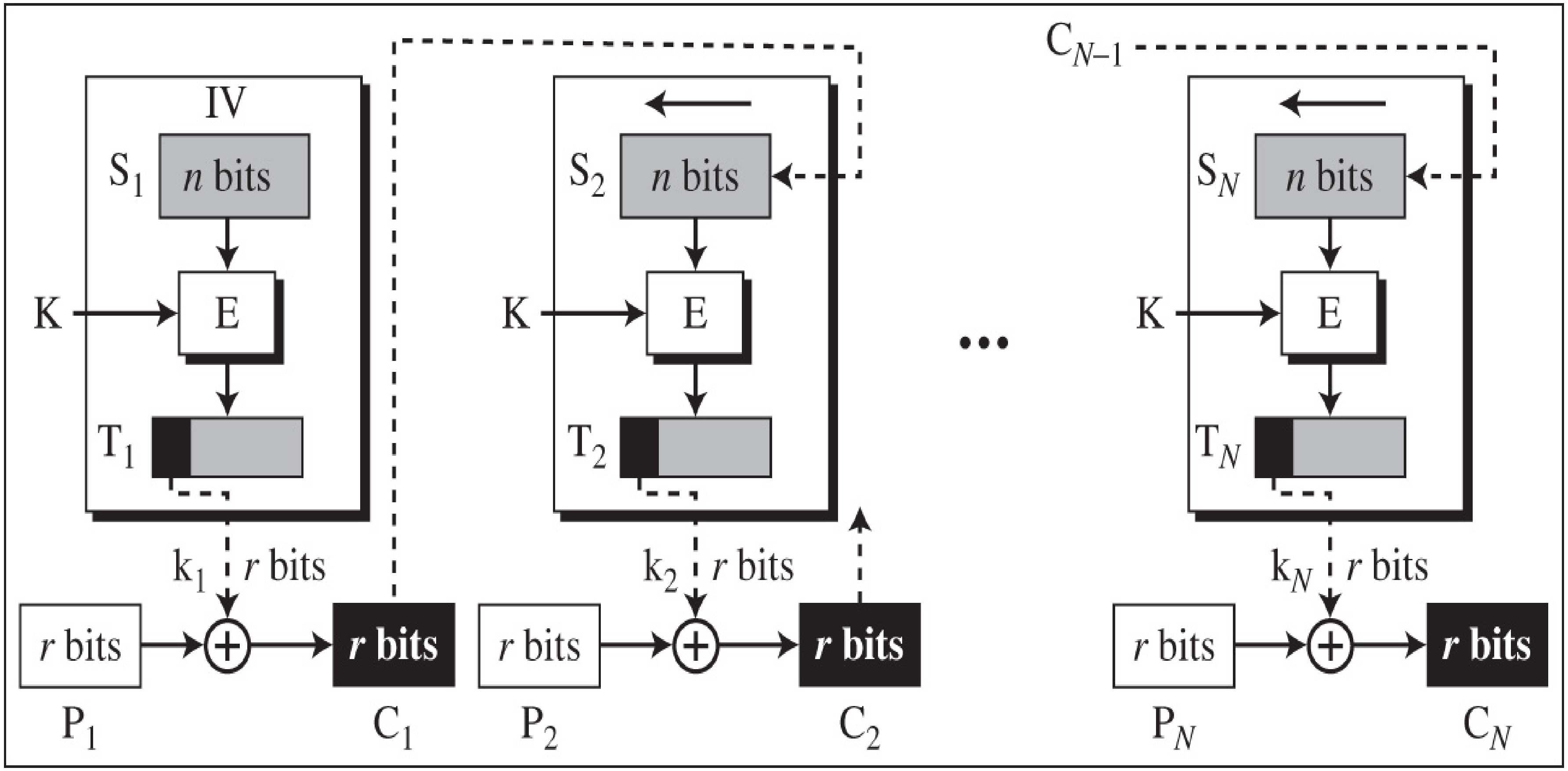
Cipher Feedback Mode Contd.



- Transmissions can be corrupted by noise
- In CFB one error corrupts many decrypted bytes (until error leaves shift register)
- Generally not a problem in modern networks which do error checking

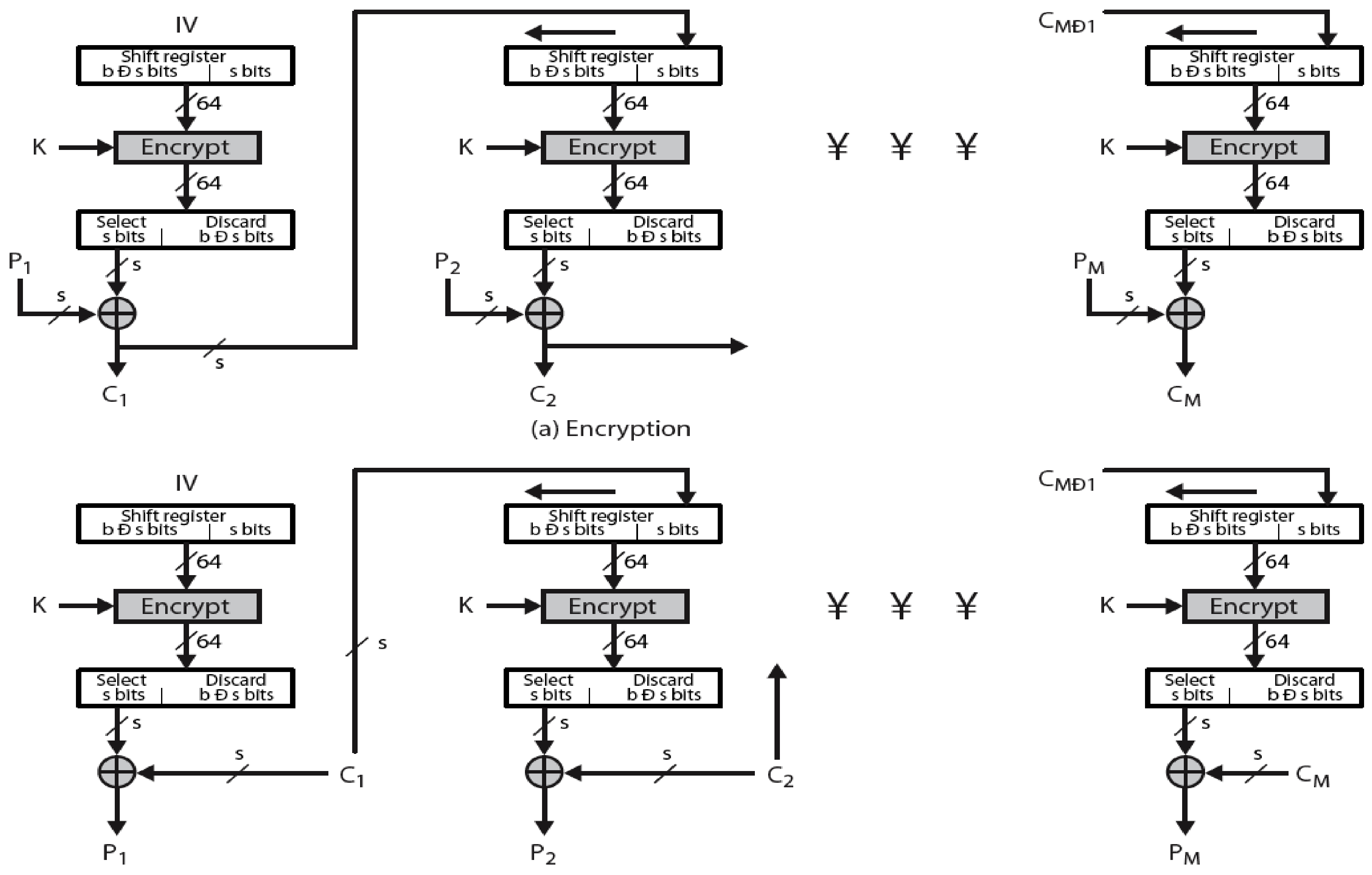
Cipher Feedback Mode Contd.

- Initial contents of shift register **S** is **Initialization Vector** (**IV)**
- Rest of ciphertext depends on previous ciphertext

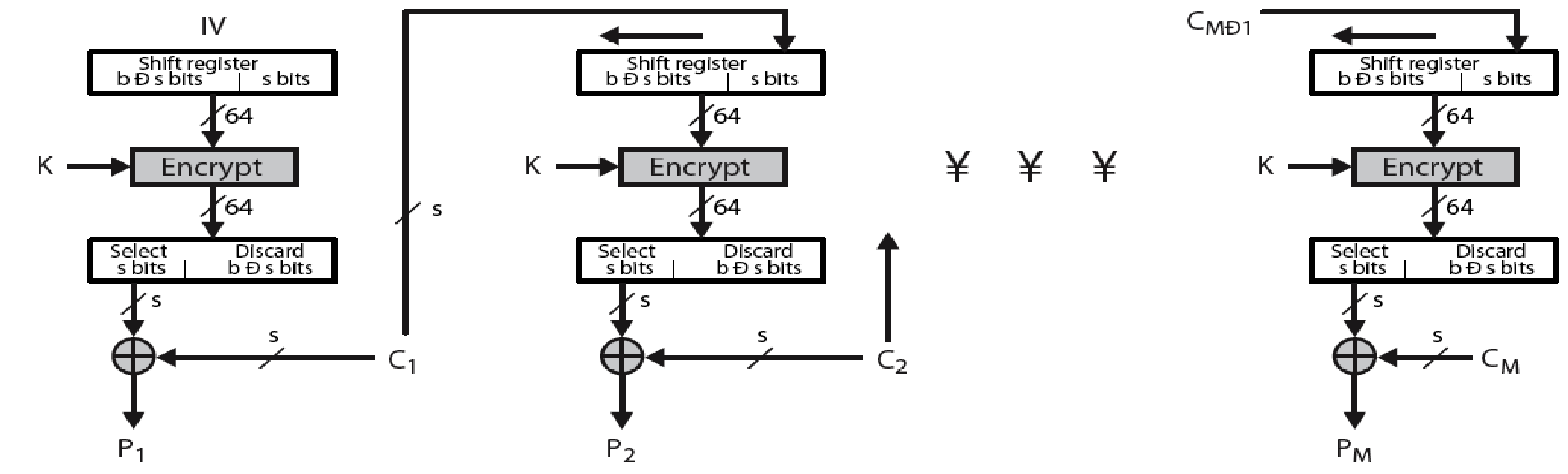


Encryption

Cipher Feedback Mode Contd.



(a) Encryption



(b) Decryption

Advantages and Limitations of CFB

- Appropriate when data arrives in bits/bytes
- Limitation is need to stall while do block encryption after every n-bits
- CFB is a stream cipher as opposed to block cipher since it uses only 8-bit blocks
- CFB works like CBC by chaining all the preceding plaintexts
- CFB uses an initialization vector of size 64 bits
- CFB uses a shift register of 8 bits
- Note that the block cipher is used in **encryption** mode at **both** ends

Cipher Feedback Mode (CFB)

Problem

- CFB inherently sequential
  - Each block depends on previous block(s)
  - Cannot take advantage of parallel hardware to speed up encryption/decryption
  - Cannot generate byte keys in advance while waiting for rest of message

Solutions:

- Output Feedback Mode (OFB)
- Counter Mode (CTR)

Problem 1

- With the ECB mode of DES, if there is an erro in a block of the transmitted cipher text only the corresponding plaintext block is affected. However, in the CBC mode, this error propagates. For example, an error in the transmitted C_1 obiviously corrupts in P_1 and P2
  - Are any blocks beyond P2 affected
  - Suppose that there is a bit error in the source version of $P_{1.}$ Through how many Ciphertext blocks is this error propagated? What is the effect at the receiver

- If a Bit error occurs in transmission of a cipher text character in 8bit CFB mode how far does the error propagates.

Solutions

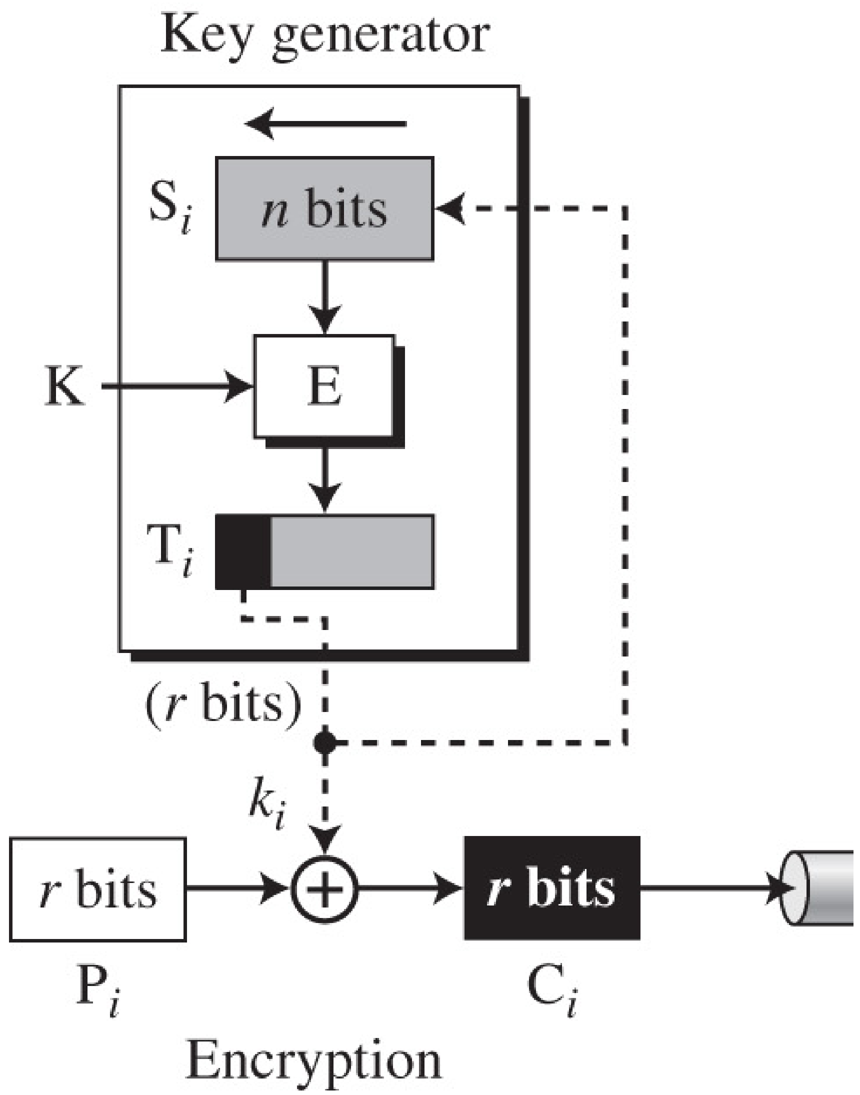
A. No. For example, suppose C1 is corrupted. The output block P3 depends only on the input blocks C2 and C3.

B. An error in P1 affects C1. But since C1 is input to the calculation of C2, C2 is affected. This effect carries through indefinitely, so that all ciphertext blocks are affected. However, at the receiving end, the decryption algorithm restores the correct plaintext for blocks except the one in error. You can show this by writing out the equations for the decryption. Therefore, the error only effects the corresponding decrypted plaintext block.

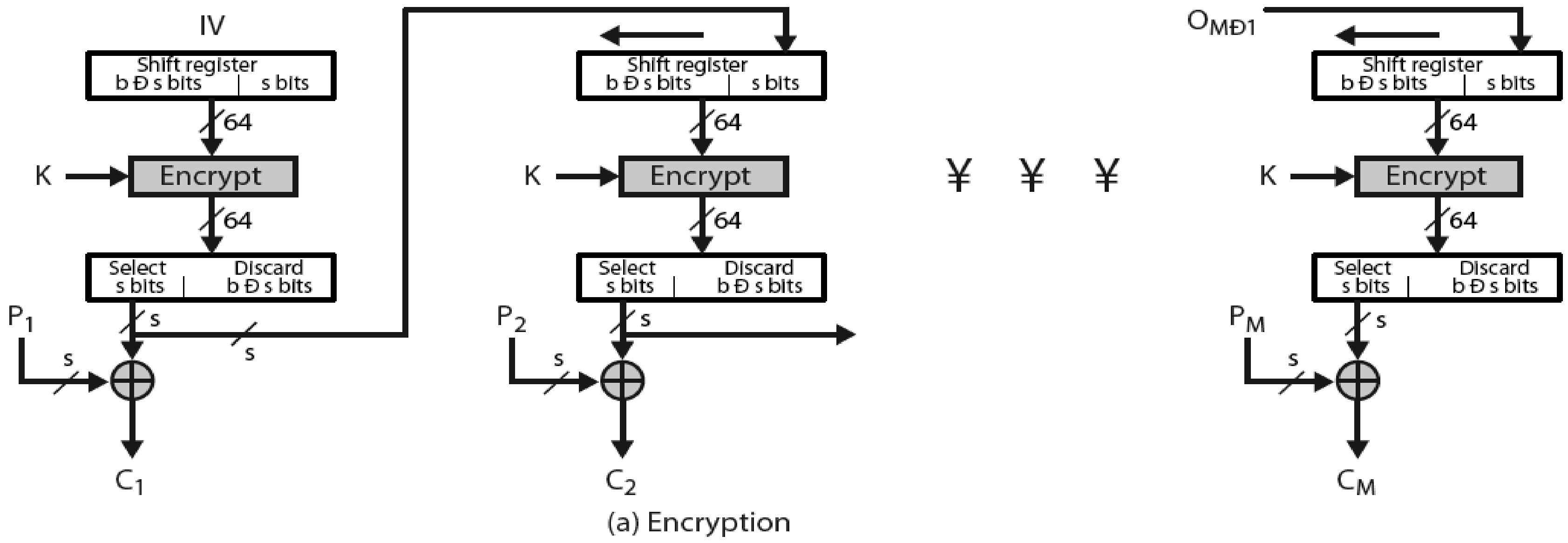
2. Nine plaintext characters are affected. The plaintext character corresponding to the ciphertext character is obviously altered. In addition, the altered ciphertext character enters the shift register and is not removed until the next eight characters are processed.

Output Feedback Mode (OFB)

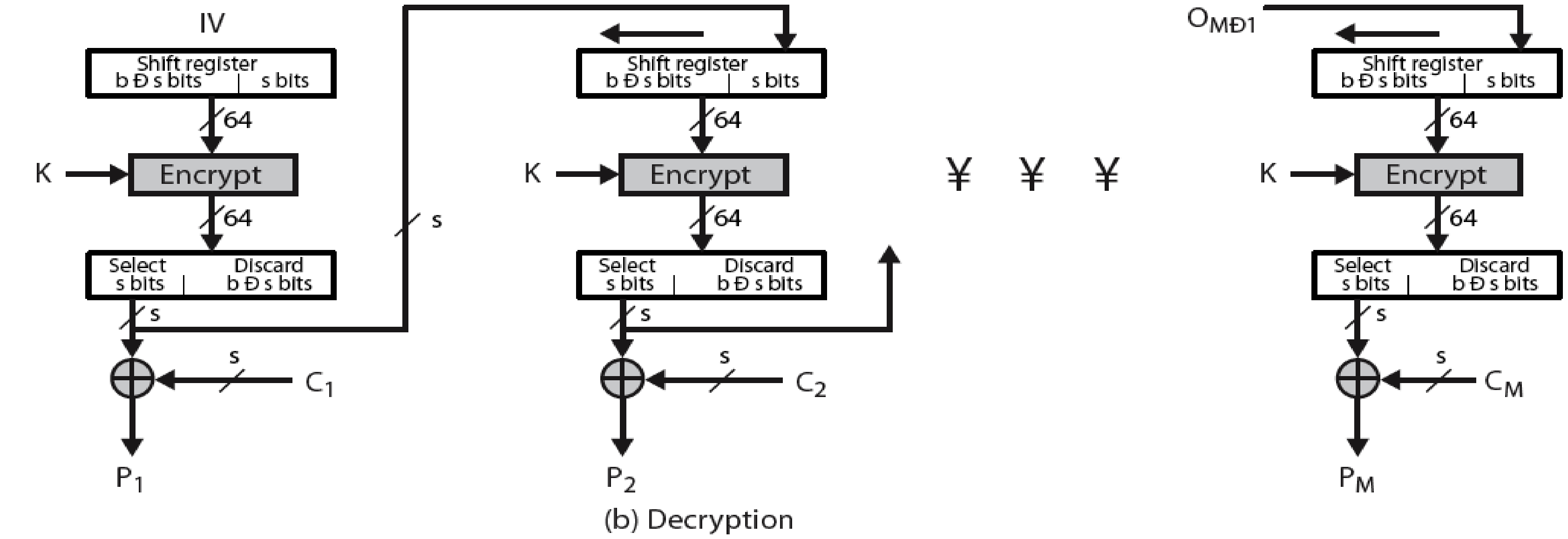
- Contents added to shift register taken directly from **T**
- Not dependent on the plaintext
- Could theoretically generate all byte keys in advance



Output Feedback Mode



(a) Encryption



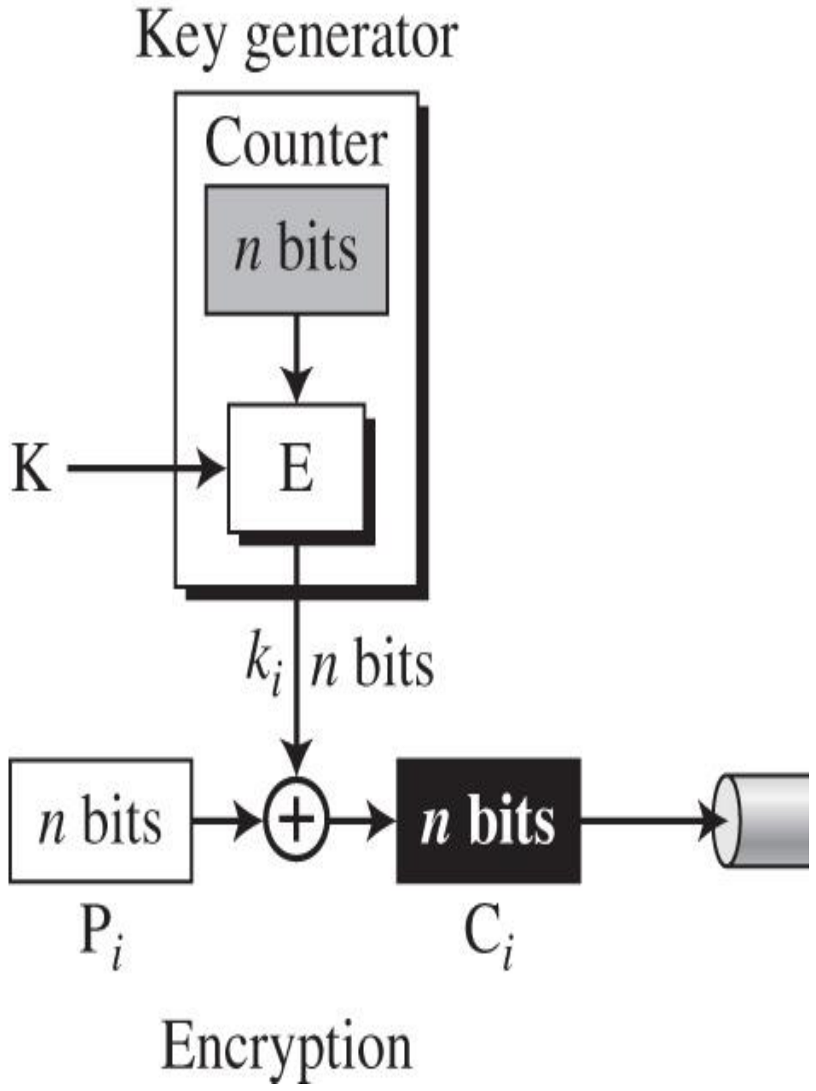
(b) Decryption

Counter Mode (CTR)

- Use a simple counter to generate next bytes of ciphertext
  - Counter increments each time →
    different ciphertext generated
  - Know all counter values in advance →
    Know all byte keys $\mathbf{k_i}$ in advance →
    Can encrypt/decrypt in parallel

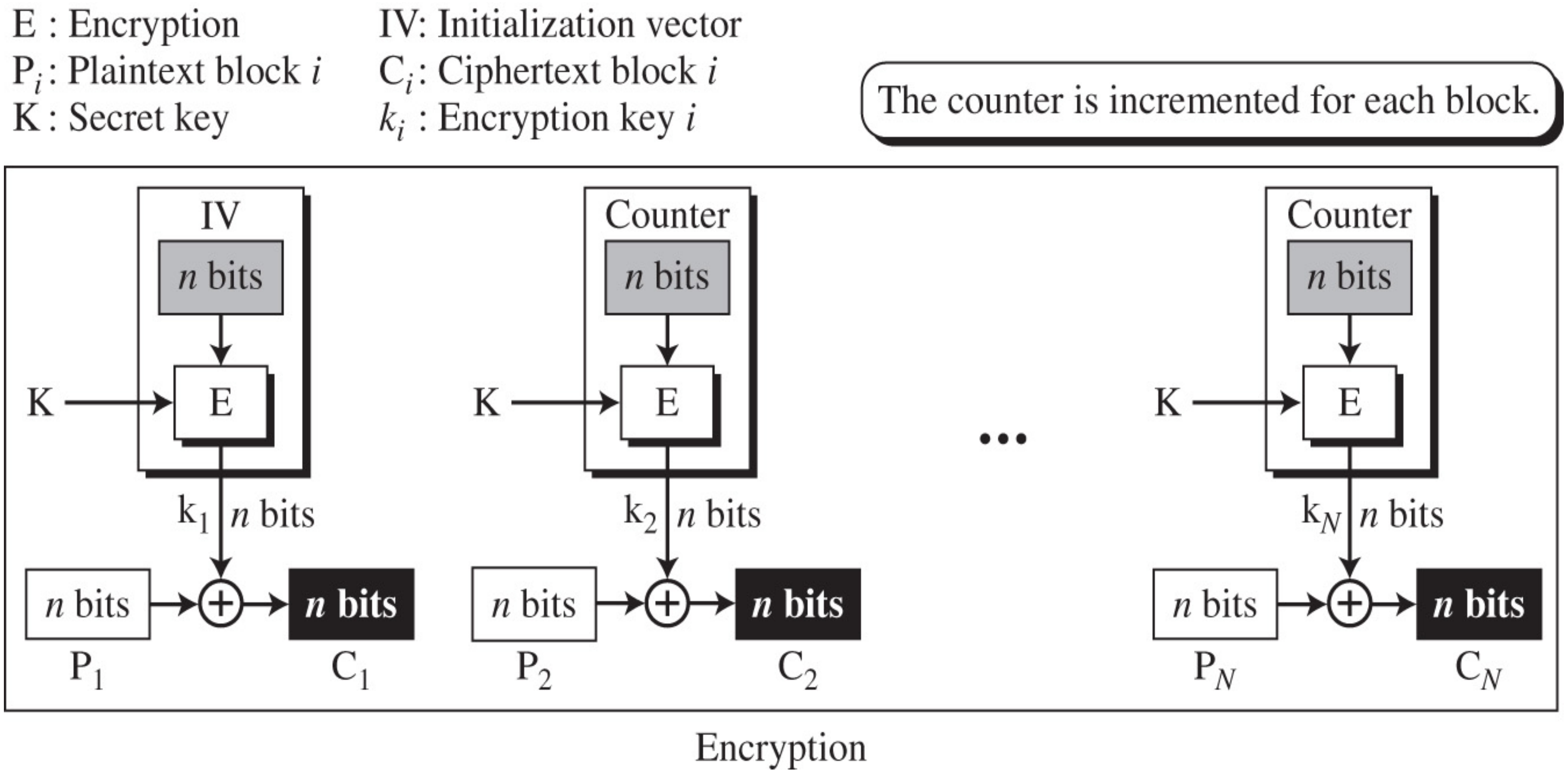
Counter Mode (CTR)

- Counter generates next *n* bits used in key generator
    - Encrypted with key
    - XORed with plaintext
    - Can select first **r** bits of result for stream transmission



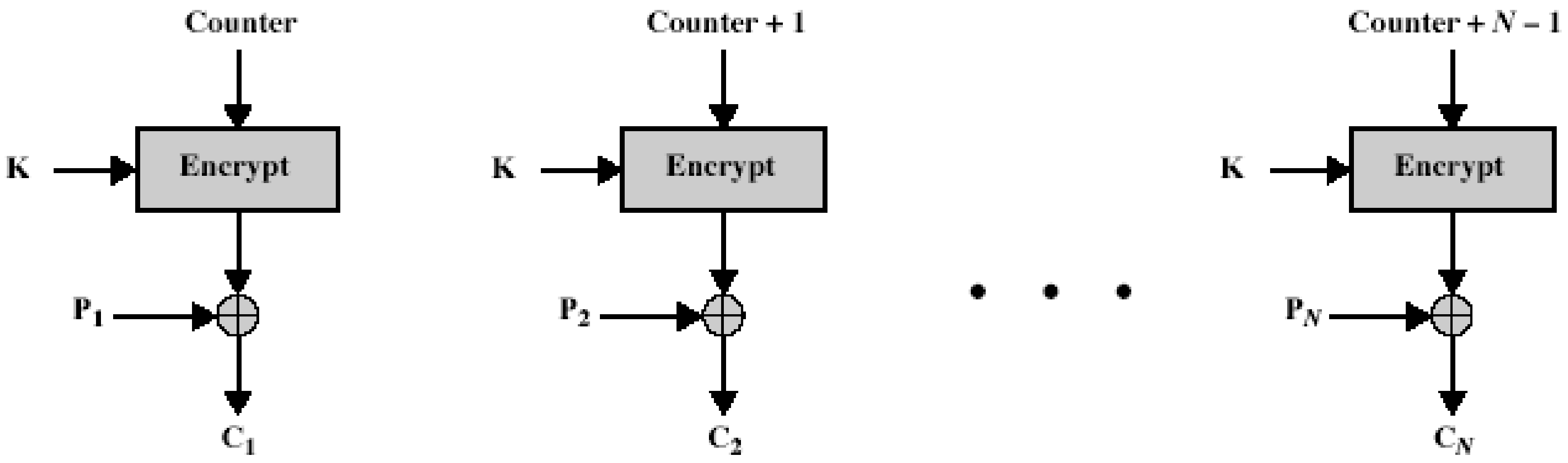
Counter Mode (CTR)

- Sender and recipient must know <u>initial</u> counter value **IV**
  - Can be transmitted via ECB mode

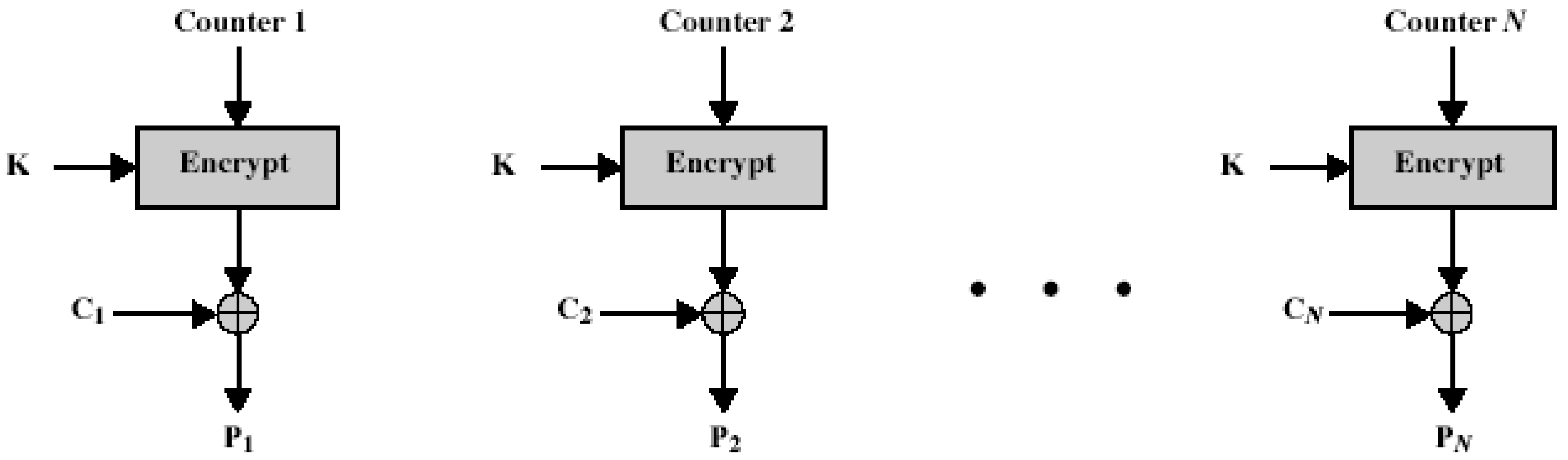


E : Encryption
P_i : Plaintext block i
K : Secret key
IV: Initialization vector
C_i : Ciphertext block i
k_i : Encryption key i
The counter is incremented for each block.
IV
Counter
Counter
n bits
K
E
k_1 *n* bits
k_2 *n* bits
k_N *n* bits
P_1
C_1
P_2
C_2
P_N
C_N
Encryption

Counter Mode

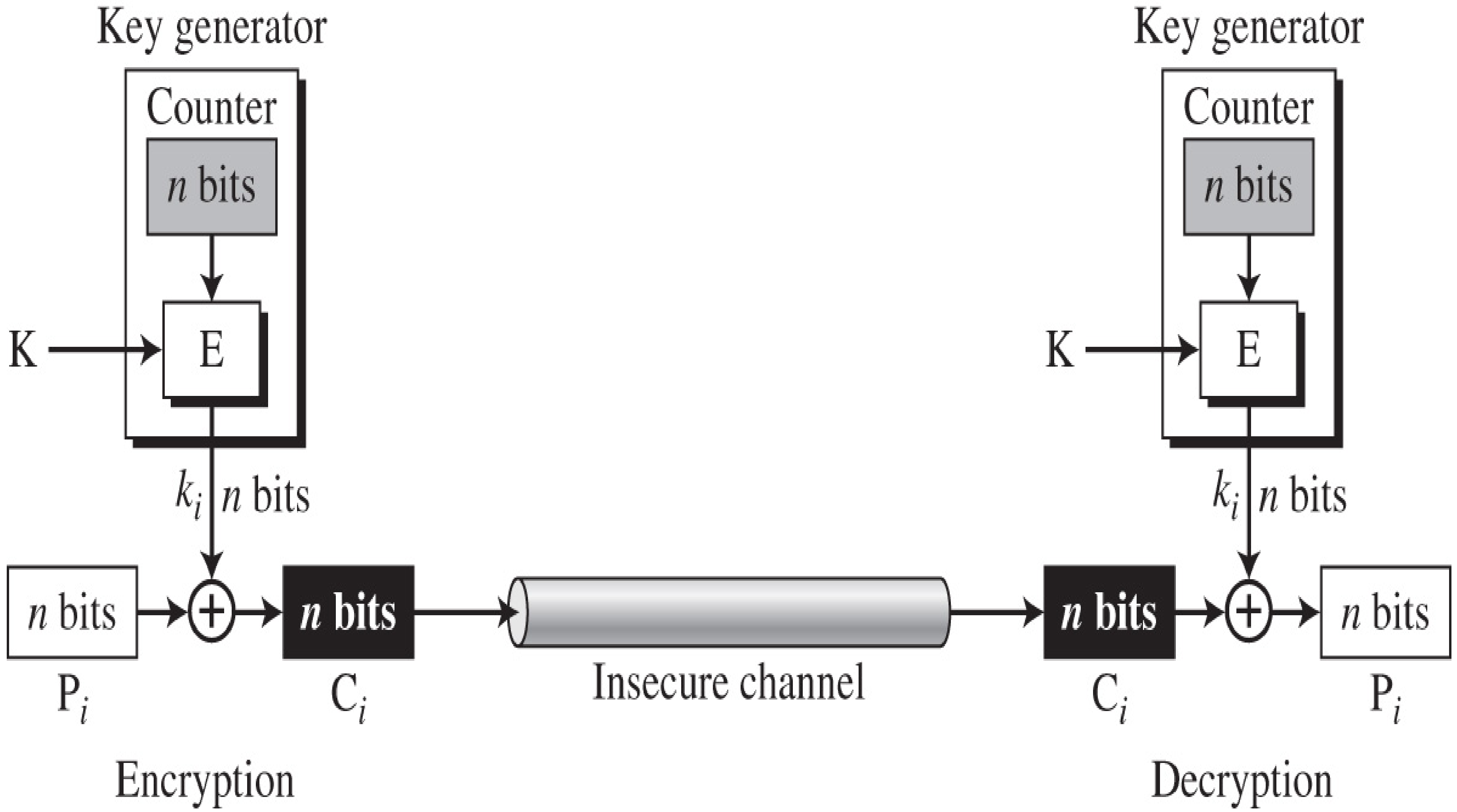


(a) Encryption



(b) Decryption

Counter Mode (CTR)



Comparison of Different Modes

Operation Mode	Description	Type of Results	Data Unit Size
ECB	Each b-bit block is encrypted independently with the same Cipher Key.	Block Cipher	n
CBC	Same as ECB, but each block is first exclusive ORed with the previous Cipher Text.	Block Cipher	n
CFB	Each r-bit block is exclusive-ORed with an r-bit key, which is part of previous cipher text.	Stream Cipher	r<=n
OFB	Same as CFB, but the shift register is updated by the previous r-bit key.	Stream Cipher	r<=n
CTR	Same OFB, but a counter is used instead of a shift register.	Stream Cipher	n

Output FeedBack (OFB)

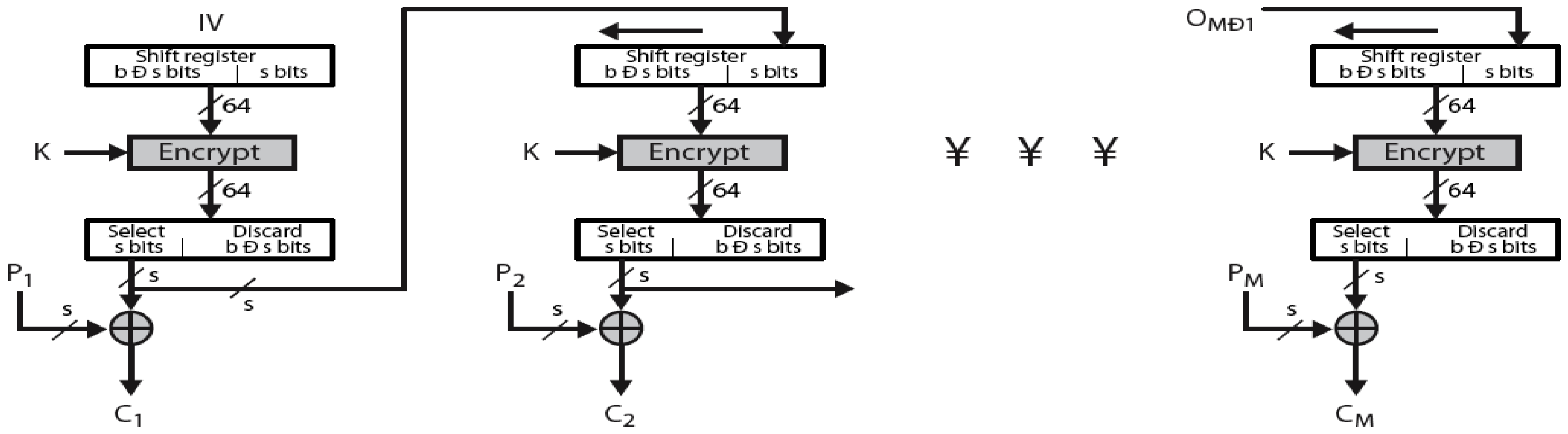
- message is treated as a stream of bits
- output of cipher is added to message
- output is then feed back (hence name)
- feedback is independent of message
- can be computed in advance

  $C_i = P_i \text{ XOR } O_i$

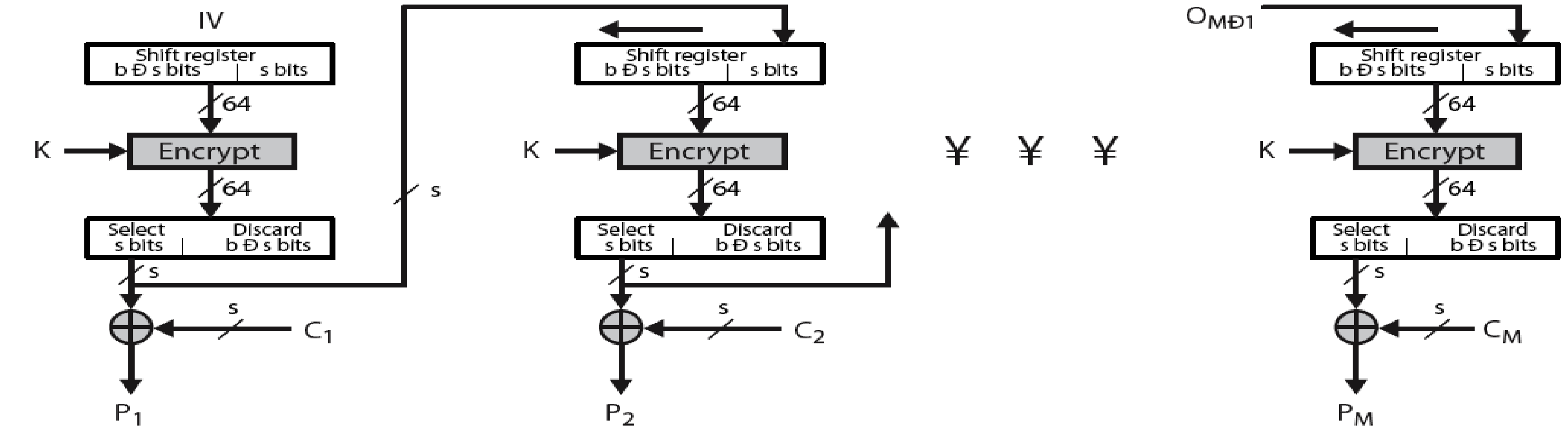
  $O_i = DES_{K1}(O_{i-1})$

  $O_{-1} = IV$
- uses: stream encryption on noisy channels

Output FeedBack (OFB)



(a) Encryption



(b) Decryption

Advantages and Limitations of OFB

- bit errors do not propagate
- more vulnerable to message stream modification
- a variation of a Vernam cipher
  - hence must **never** reuse the same sequence (key+IV)
- sender & receiver must remain in sync
- originally specified with m-bit feedback
- subsequent research has shown that only **full block feedback** (ie CFB-64 or CFB-128) should ever be used

Counter (CTR)

- a “new” mode, though proposed early on
- similar to OFB but encrypts counter value rather than any feedback value
- must have a different key & counter value for every plaintext block (never reused)

  C_i = P_i XOR O_i

  $O_i = DES_{K1}(i)$
- uses: high-speed network encryptions

Advantages and Limitations of CTR

- Efficiency
  - can do parallel encryptions in h/w or s/w
  - can preprocess in advance of need
  - good for bursty high speed links
- Random access to encrypted data blocks
- Provable security (good as other modes)
- But must ensure never reuse key/counter values, otherwise could break (cf OFB)