Chapter 8

Encipherment Using Modern Symmetric-Key Ciphers

8-1 USE OF MODERN BLOCK CIPHERS

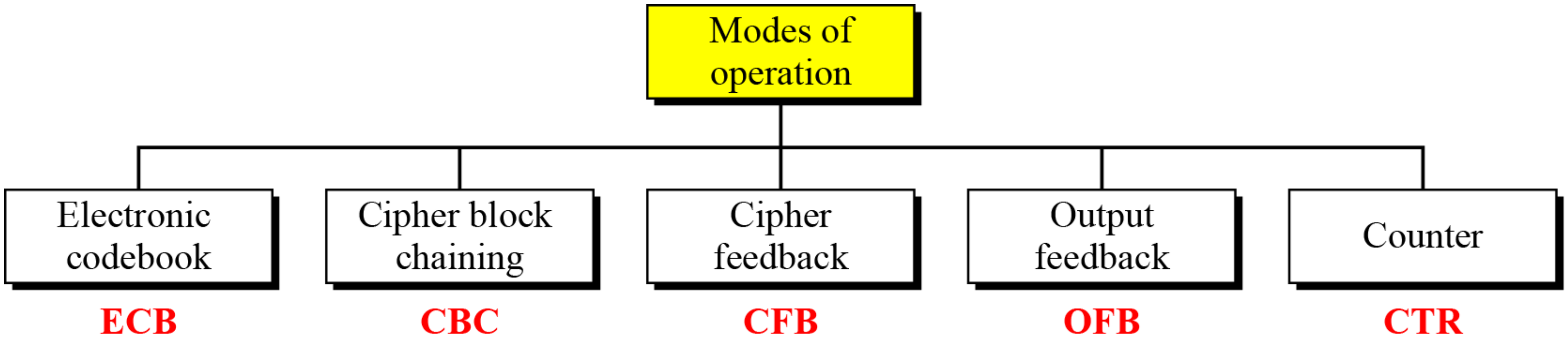
Symmetric-key encipherment can be done using modern block ciphers. Modes of operation have been devised to encipher text of any size employing either DES or AES.

Topics discussed in this section:

8.1.1 Electronic Codebook (ECB) Mode
8.1.2 Cipher Block Chaining (CBC) Mode
8.1.3 Cipher Feedback (CFB) Mode
8.1.4 Output Feedback (OFB) Mode
8.1.5 Counter (CTR) Mode

8-1 Continued

Figure 8.1 ***Modes of operation***

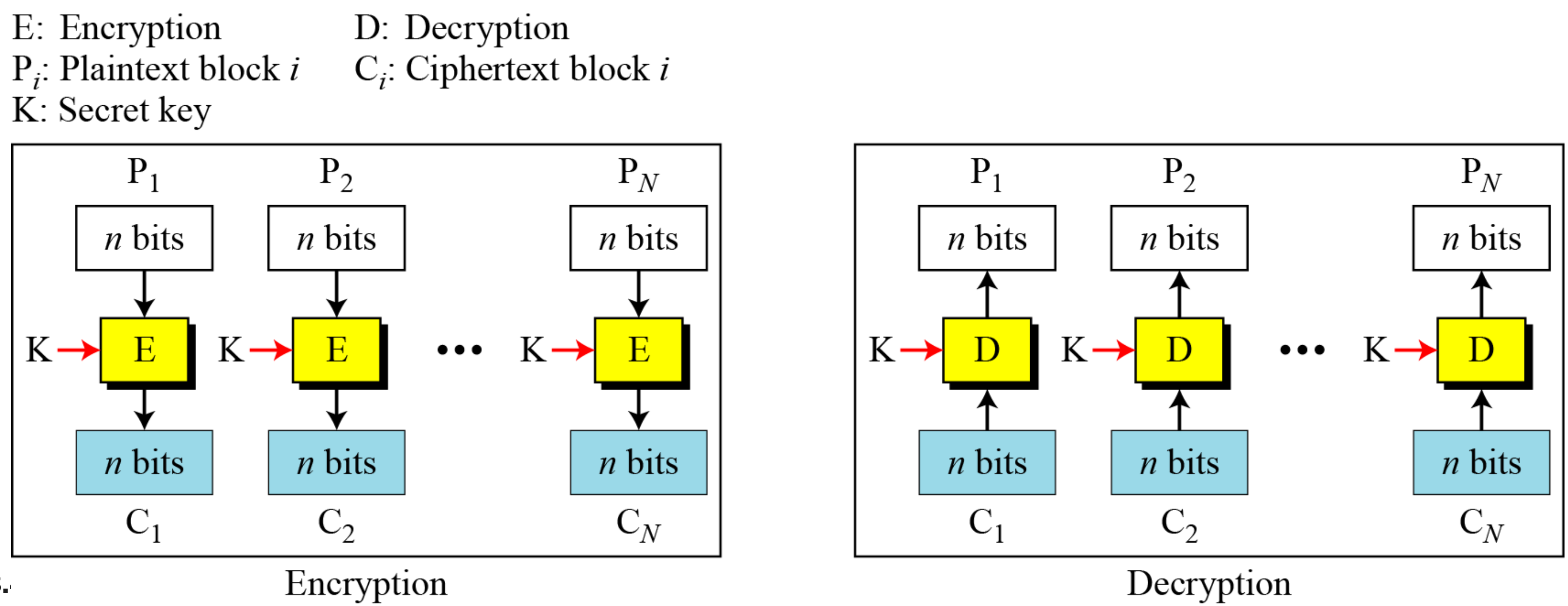


8.1.1 Electronic Codebook (ECB) Mode

The simplest mode of operation is called the electronic codebook (ECB) mode.

Encryption: $C_i = E_K(P_i)$ Decryption: $P_i = D_K(C_i)$

Figure 8.2 ***Electronic codebook (ECB) mode***



Example 8.1

It can be proved that each plaintext block at Alice's site is exactly recovered at Bob's site. Because encryption and decryption are inverses of each other,

$$\text{Encryption: } C_i = E_K(P_i) \qquad \text{Decryption: } P_i = D_K(C_i)$$

Example 8.2

This mode is called electronic codebook because one can precompile 2^K codebooks (one for each key) in which each codebook has 2^n entries in two columns. Each entry can list the plaintext and the corresponding ciphertext blocks. However, if K and n are large, the codebook would be far too large to precompile and maintain.

8.1.1 Continued

Example 8.3

How can Eve break the security?

- **If she has low payment, she can replace it with the corresponding block if high paid employee**

Error Propagation

A single bit error in transmission can create errors in several in the corresponding block. However, the error does not have any effect on the other blocks.

Algorithm 8.1 *Encryption for ECB mode*

```
ECB_Encryption (K, Plaintext blocks)
{
    for (i = 1 to N)
    {
        C_i ← E_K (P_i)
    }
    return Ciphertext blocks
}
```

8.1.1 Continued

Ciphertext Stealing

A technique called ciphertext stealing (CTS) can make it possible to use ECB mode without padding. In this technique the last two plaintext blocks, P_{N-1} and P_N, are encrypted differently and out of order, as shown below, assuming that P_{N-1} has n bits and P_N has m bits, where m $\leq n$.

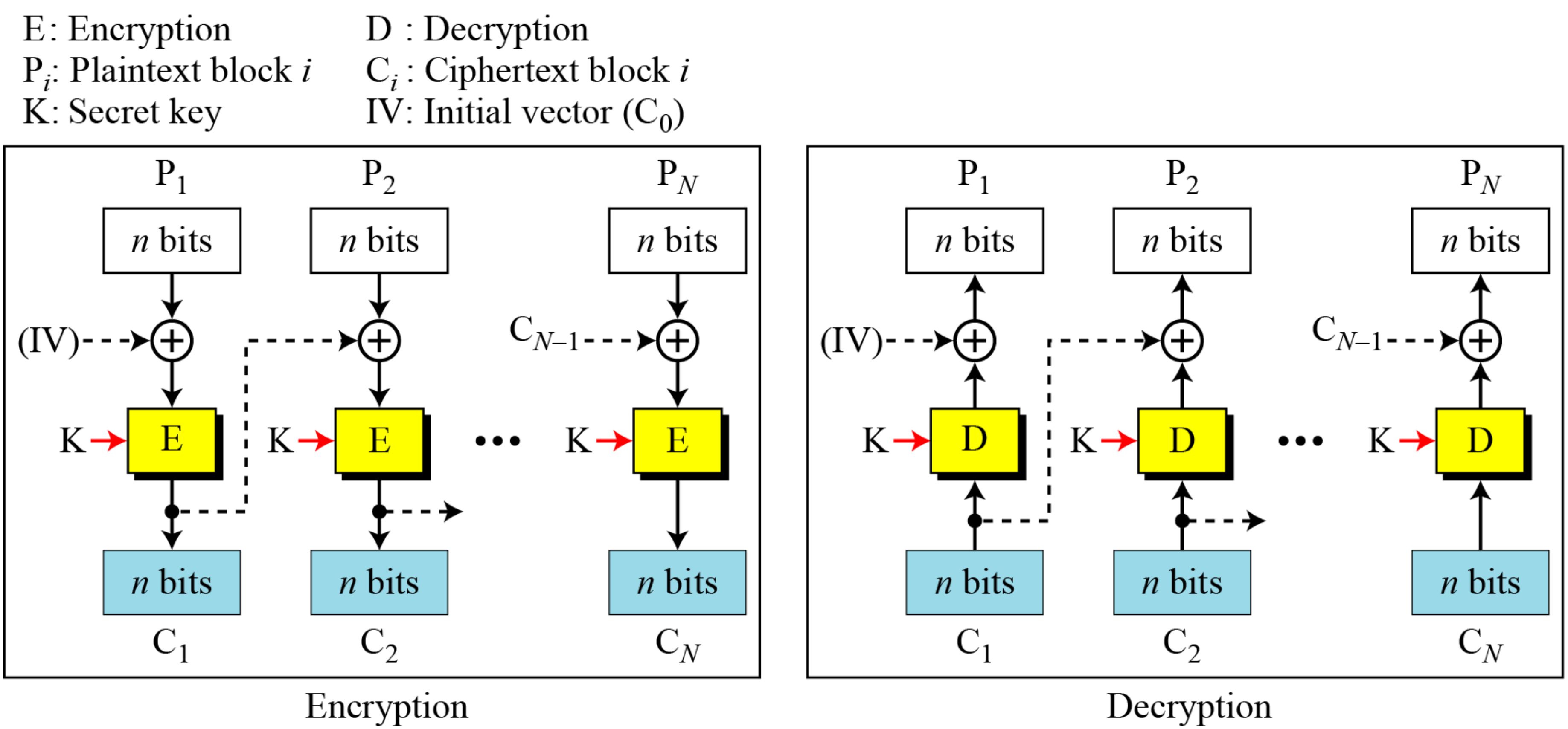
$$X = E_K(P_{N-1}) \quad \rightarrow \quad C_N = head_m(X)$$

$$Y = P_N \mid tail_{n-m}(X) \quad \rightarrow \quad C_{N-1} = E_K(Y)$$

8.1.2 Cipher Block Chaining (CBC) Mode

In CBC mode, each plaintext block is exclusive-ored with the previous ciphertext block before being encrypted.

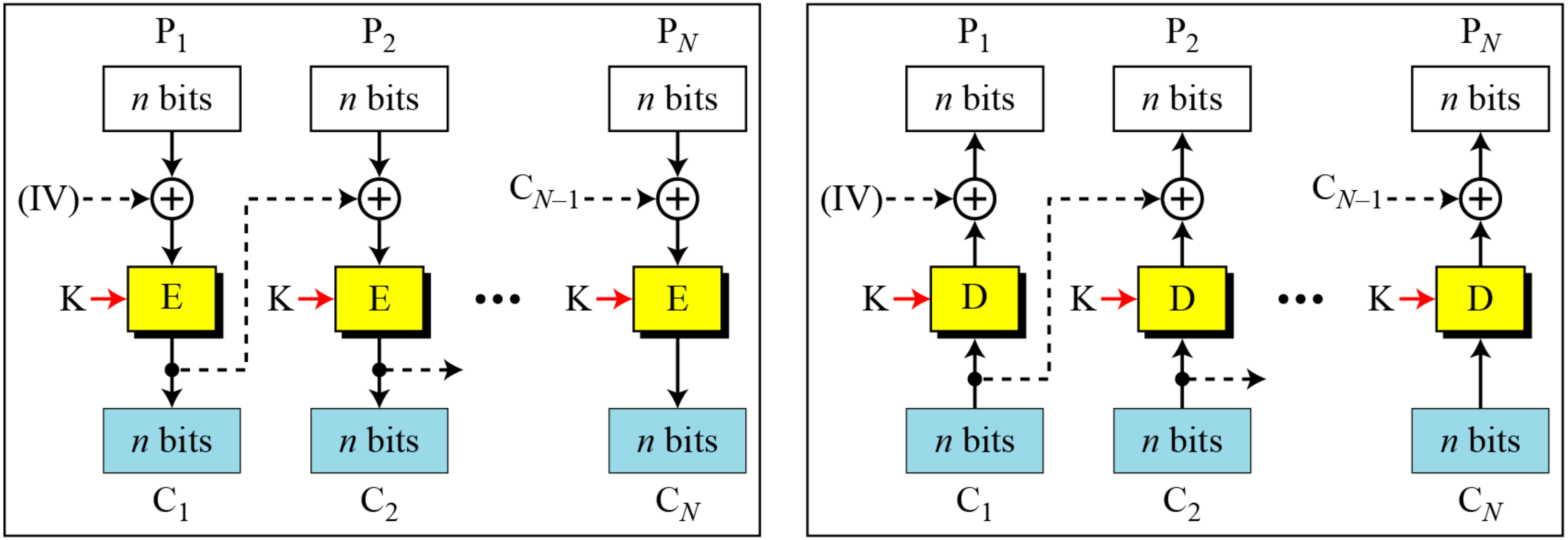
Figure 8.3 ***Cipher block chaining (CBC) mode***



8.1.2 Continued

Figure 8.3 ***Cipher block chaining (CBC) mode***

E: Encryption — D : Decryption
P_i: Plaintext block i — C_i : Ciphertext block i
K: Secret key — IV: Initial vector (C_0)



Encryption — Decryption

Encryption:	Decryption:
$C_0 = IV$	$C_0 = IV$
$C_i = E_K(P_i \oplus C_{i-1})$	$P_i = D_K(C_i) \oplus C_{i-1}$

8.1.2 Continued

Example 8.4

It can be proved that each plaintext block at Alice's site is recovered exactly at Bob's site. Because encryption and decryption are inverses of each other,

$$P_i = D_K(C_i) \oplus C_{i-1} = D_K(E_K(P_i \oplus C_{i-1})) \oplus C_{i-1} = P_i \oplus C_{i-1} \oplus C_{i-1} = P_i$$

Initialization Vector (IV)

The initialization vector (IV) should be known by the sender and the receiver.

Error Propagation

In CBC mode, a single bit error in ciphertext block C_j during transmission may create error in most bits in plaintext block P_j during decryption.

Algorithm 8.2 *Encryption algorithm for ECB mode*

```
CBC_Encryption (IV, K, Plaintext blocks)
{
      C_0 ← IV
      for (i = 1 to N)
      {
            Temp ← P_i ⊕ C_{i−1}
            C_i ← E_K (Temp)
      }
      return Ciphertext blocks
}
```

Ciphertext Stealing

The ciphertext stealing technique described for ECB mode can also be applied to CBC mode, as shown below.

$$
\begin{array}{llllll}
U = P_{N-1} \oplus C_{N-2} & \rightarrow & X = E_K(U) & \rightarrow & C_N & = head_m(X) \\
V = P_N \mid pad_{n-m}(0) & \rightarrow & Y = X \oplus V & \rightarrow & C_{N-1} & = E_K(Y)
\end{array}
$$

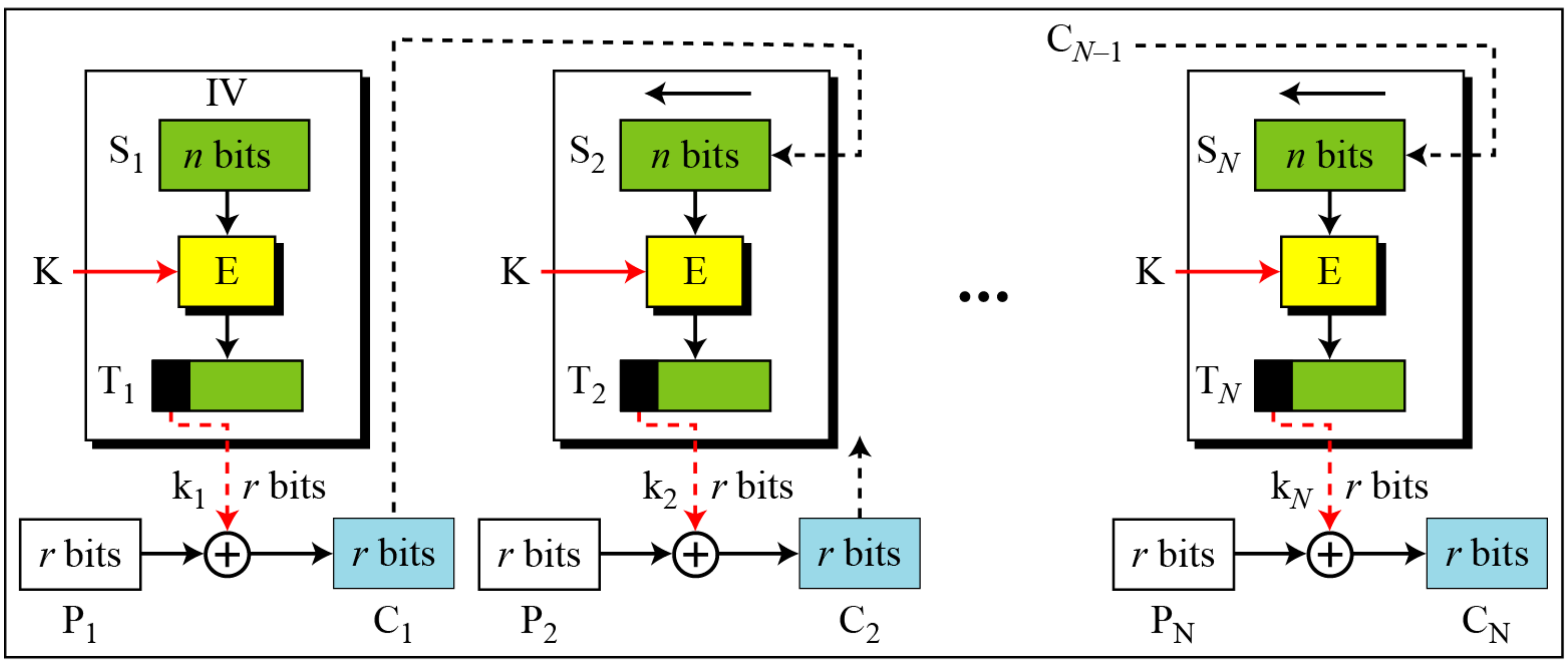
The head function is the same as described in ECB mode; the pad function inserts 0's.

8.1.3 Cipher Feedback (CFB) Mode

In some situations, we need to use DES or AES as secure ciphers, but the plaintext or ciphertext block sizes are to be smaller.

Figure 8.4 ***Encryption in cipher feedback (CFB) mode***

E : Encryption
D : Decryption
S_i: Shift register
P_i: Plaintext block i
C_i: Ciphertext block i
T_i: Temporary register
K: Secret key
IV: Initial vector (S_1)



Encryption

8.1.3 Continued

Note

In CFB mode, encipherment and decipherment use the encryption function of the underlying block cipher.

The relation between plaintext and ciphertext blocks is shown below:

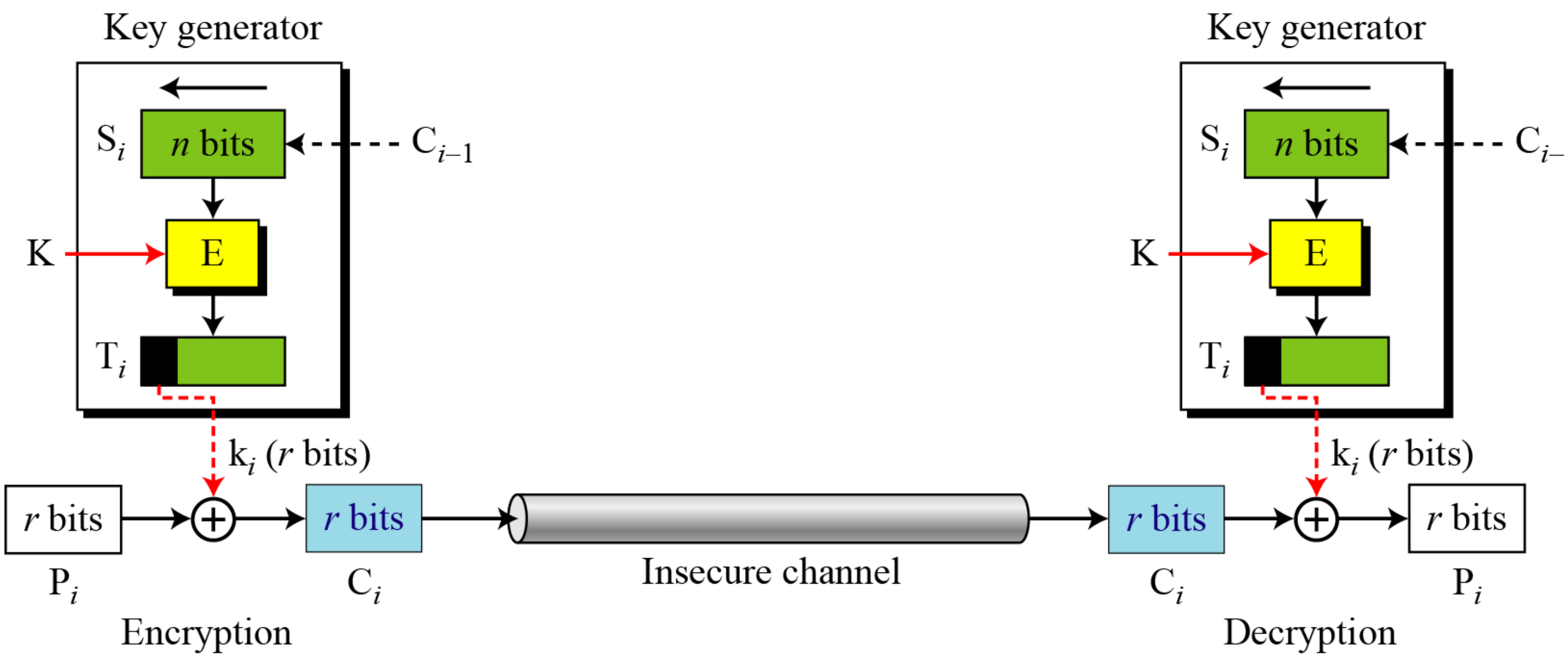
Encryption: $C_i = P_i \oplus \text{SelectLeft}_r\{E_K[\text{ShiftLeft}_r(S_{i-1}) \mid C_{i-1})]\}$

Decryption: $P_i = C_i \oplus \text{SelectLeft}_r\{E_K[\text{ShiftLeft}_r(S_{i-1}) \mid C_{i-1})]\}$

8.1.3 Continued

CFB as a Stream Cipher

Figure 8.5 *Cipher feedback (CFB) mode as a stream cipher*



8.1.3 Continued

Algorithm 8.3 *Encryption algorithm for CFB*

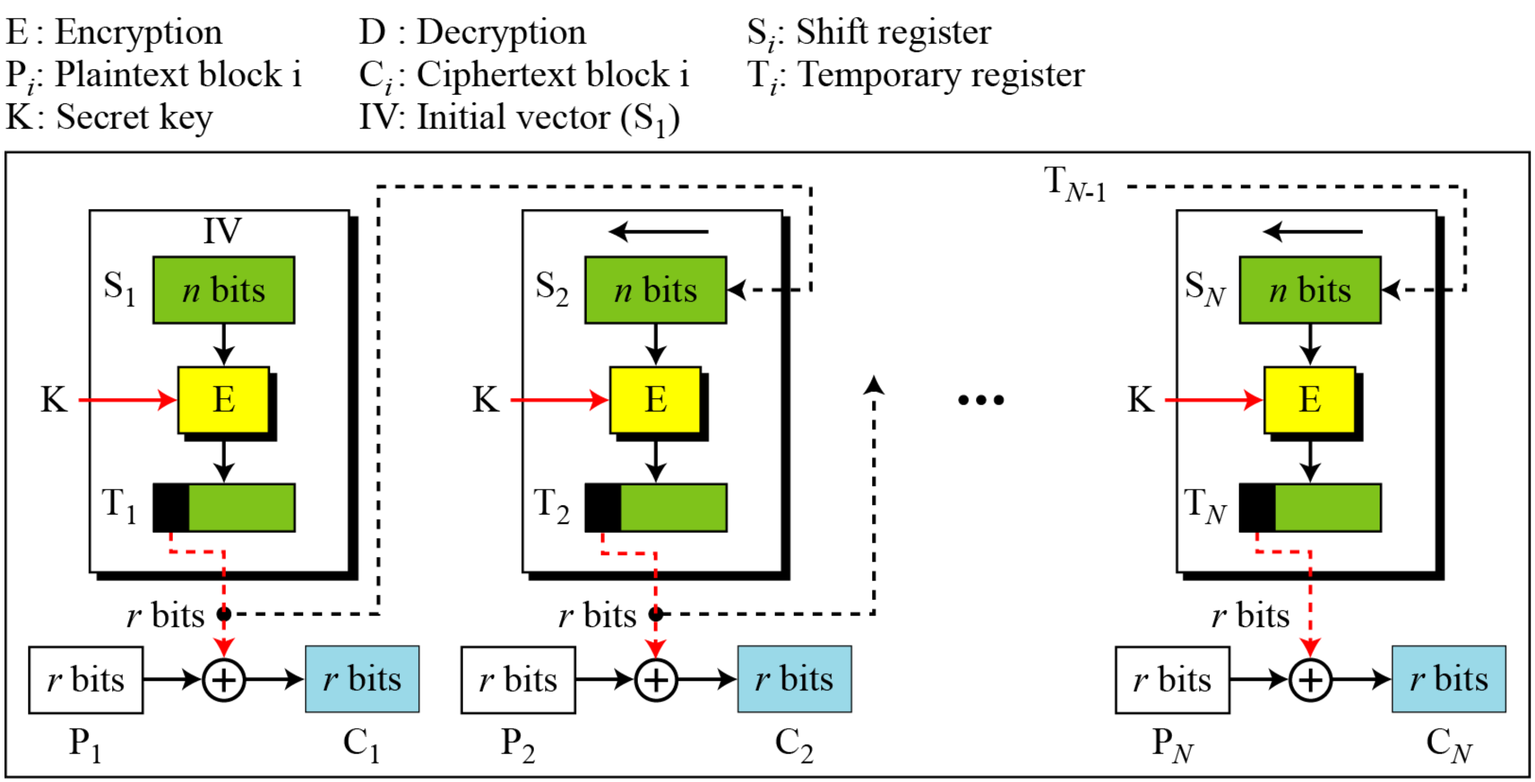
CFB_Encryption (IV, K, r)
{
  $i \leftarrow 1$
  while (more blocks to encrypt)
  {
  input (P_i)
  if ($i = 1$)
    $S \leftarrow IV$
  else
    {
    Temp $\leftarrow$ **shiftLeft**$_r$ (S)
    $S \leftarrow$ **concatenate** (Temp, C_{i-1})
    }
  $T \leftarrow E_K(S)$
  $k_i \leftarrow$ **selectLeft**$_r$ (T)
  $C_i \leftarrow P_i \oplus k_i$
  output (C_i)
  $i \leftarrow i + 1$
  }
}

18.1.4 Output Feedback (OFB) Mode

In this mode each bit in the ciphertext is independent of the previous bit or bits. This avoids error propagation.

- *Note that the size of the plaintext is r , which usually less than n*

Figure 8.6 *Encryption in output feedback (OFB) mode*

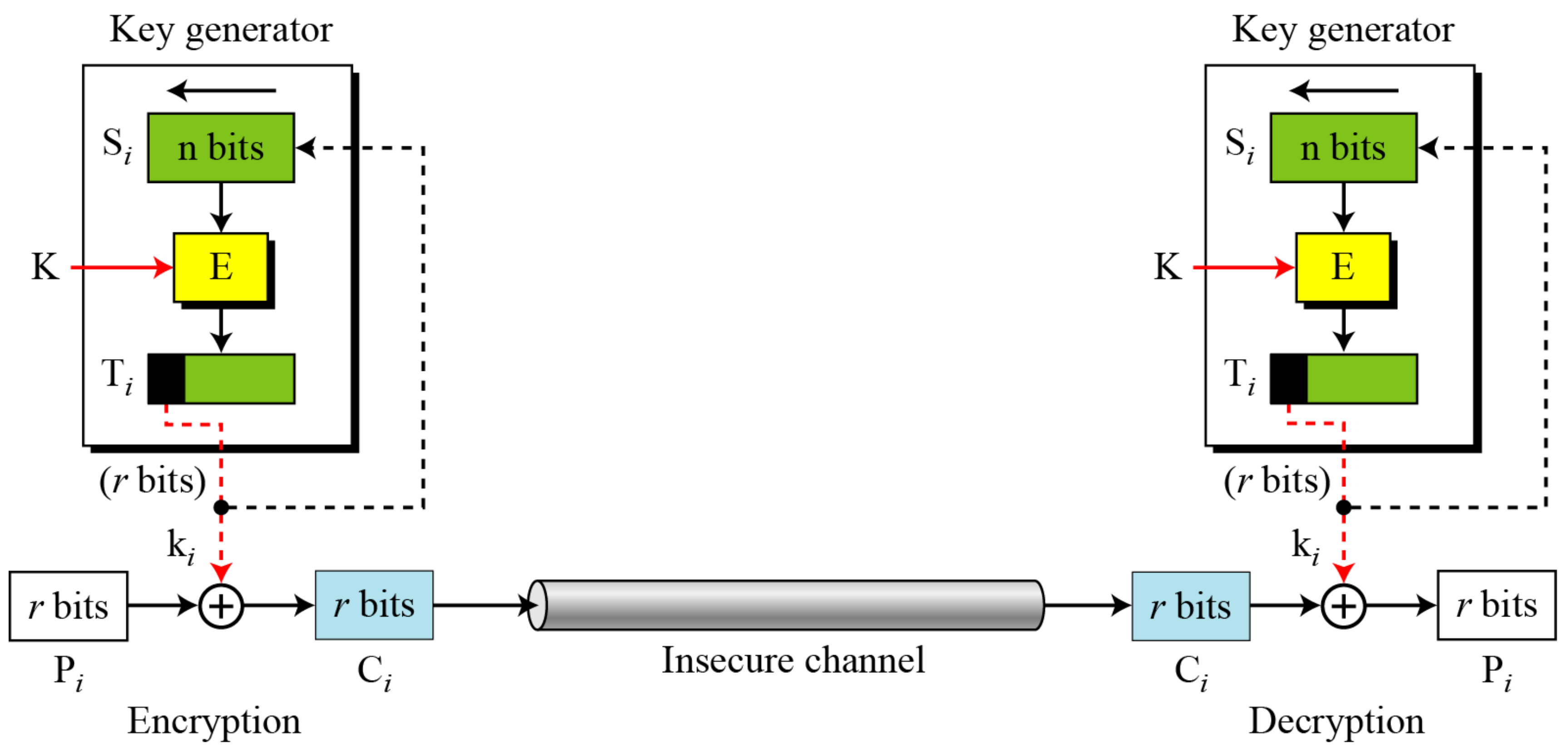


Encryption

8.1.4 Continued

OFB as a Stream Cipher

Figure 8.7 ***Output feedback (OFB) mode as a stream cipher***



8.1.4 Continued

Algorithm 8.4 *Encryption algorithm for OFB*

OFB_Encryption (IV, K, r)
{
  $i \leftarrow 1$
  while (more blocks to encrypt)
  {
    input (P_i)
    if ($i = 1$) $S \leftarrow IV$
    else
    {
      Temp $\leftarrow$ **shiftLeft**$_r$ (S)
      $S \leftarrow$ **concatenate** (Temp, k_{i-1})
    }
    $T \leftarrow E_K(S)$
    $k_i \leftarrow$ **selectLeft**$_r$ (T)
    $C_i \leftarrow P_i \oplus k_i$
    output (C_i)
    $i \leftarrow i + 1$
  }
}

8.1.5 Counter (CTR) Mode

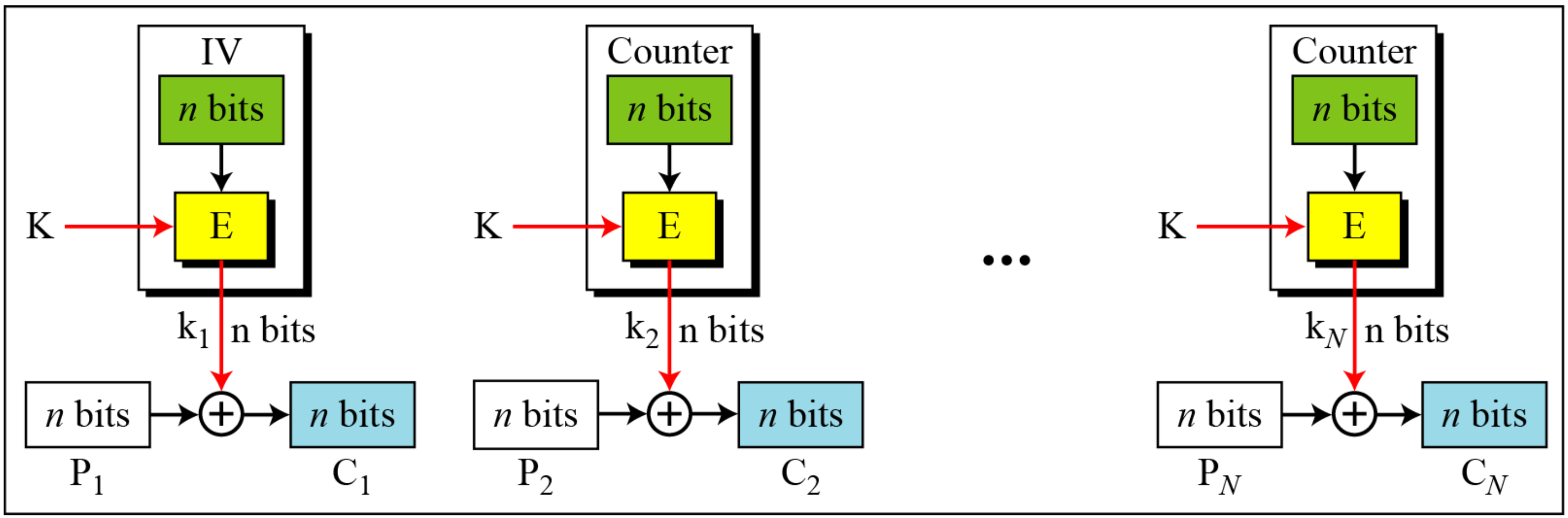
In the counter (CTR) mode, there is no feedback. The pseudorandomness in the key stream is achieved using a counter.

- *Note that the size of the plaintext is n , similar to the IV*

Figure 8.8 *Encryption in counter (CTR) mode*

E : Encryption
P_i : Plaintext block i
K : Secret key
IV: Initialization vector
C_i : Ciphertext block i
k_i : Encryption key i

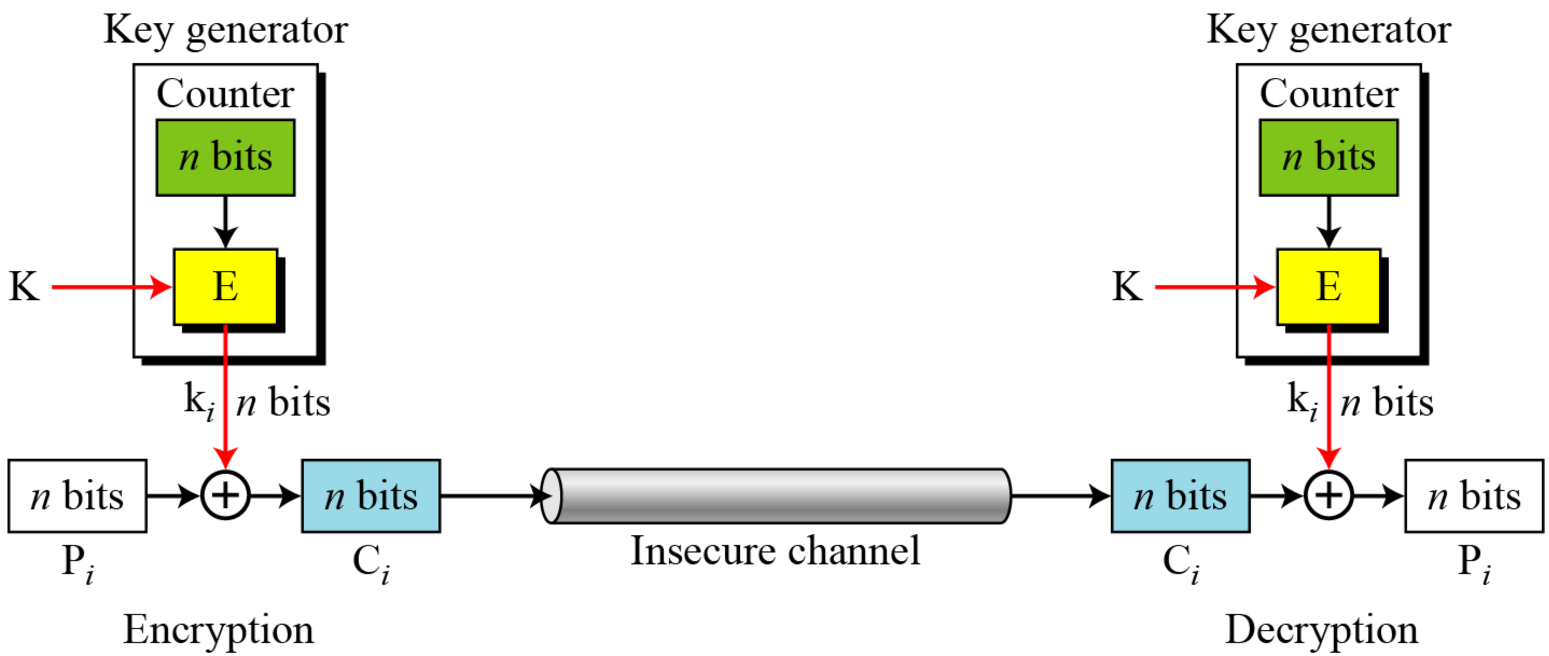
The counter is incremented for each block.



Encryption

8.1.5 Continued

Figure 8.9 *Counter (CTR) mode as a stream cipher*



8.1.5 Continued

Algorithm 8.5 *Encryption algorithm for CTR*

```
CTR_Encryption (IV, K, Plaintext blocks)
{
    Counter ← IV
    for (i = 1 to N)
    {
        Counter ← (Counter + i − 1) mod 2^N
        k_i ← E_K (Counter)
        C_i ← P_i ⊕ k_i
    }
    return Ciphertext blocks
}
```

8.1.5 Continued

Comparison of Different Modes

Table 8.1 *Summary of operation modes*

Operation Mode	*Description*	*Type of Result*	*Data Unit Size*
ECB	Each n-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 ciphertext.	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	$r \leq n$
OFB	Same as CFB, but the shift register is updated by the previous r-bit key.	Stream cipher	$r \leq n$
CTR	Same as OFB, but a counter is used instead of a shift register.	Stream cipher	n