CS557: Cryptography

Stream Cipher

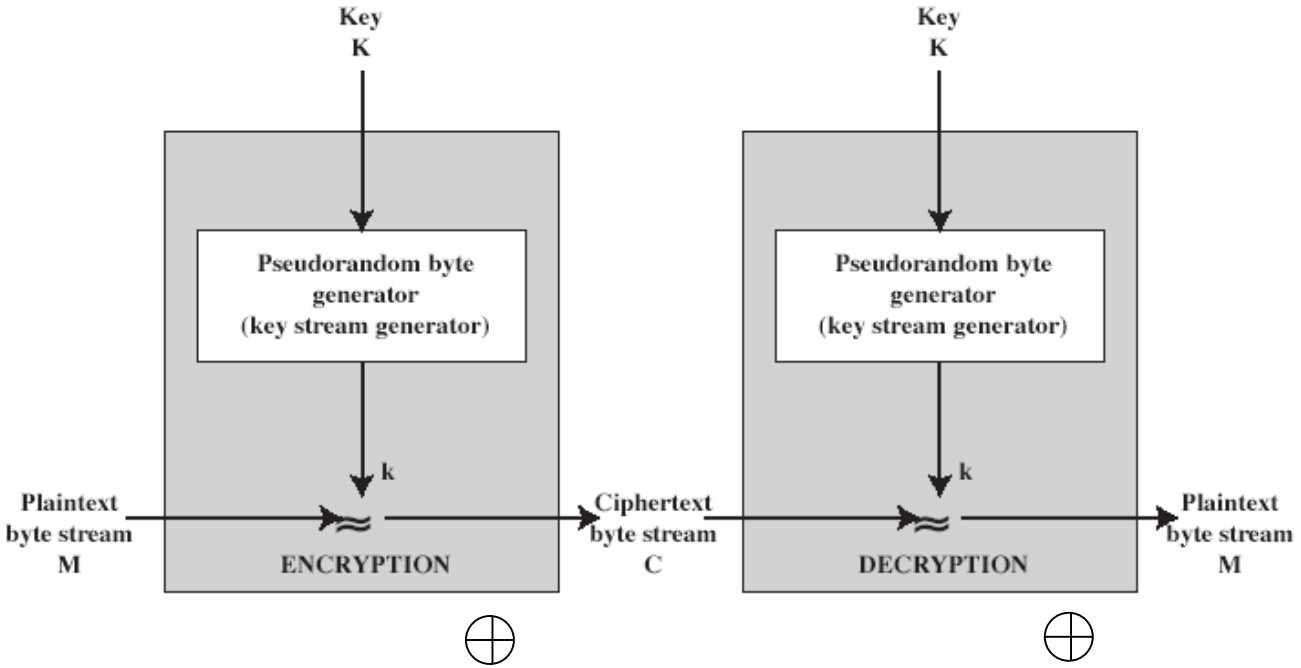
S. Tripathy
IIT Patna

Previous Classes

- Modern Cipher
  - Block cipher
    - DES &AES
      - Modes of operations
    - Cryptanalysis
      - Linear Cryptanalysis
      - Differential Cryptanalysis

Stream Ciphers

- process the message bit by bit (as a stream)
- typically have a (pseudo) random stream key
- combined (XOR) with plaintext bit by bit
- randomness of stream key completely destroys any statistically properties in the message
  - $C_i = M_i \text{ XOR } StreamKey_i$



Never reuse stream key

- Stream cipher outputs keystream, KS
  - KS produced by a function, F, that is initialized with a key, k
  - $C = E_k(P) = P \oplus KS$
  - $P = C \oplus KS$
- k can be used only once
  - $C1 = E_{k1}(P1)$; $C2 = E_{k2}(P2)$
  - $C1 \oplus C2 = P1 \oplus KS1 \oplus P2 \oplus KS2 = P1 \oplus P2$ if $KS1 = KS2$
  - Will know when P1 and P2 have identical bits
  - If know part of P1 (if packet headers, format information), then can obtain part of P2

- Period – how long is KS before it starts repeating?
  - repeating is equivalent to reusing a key

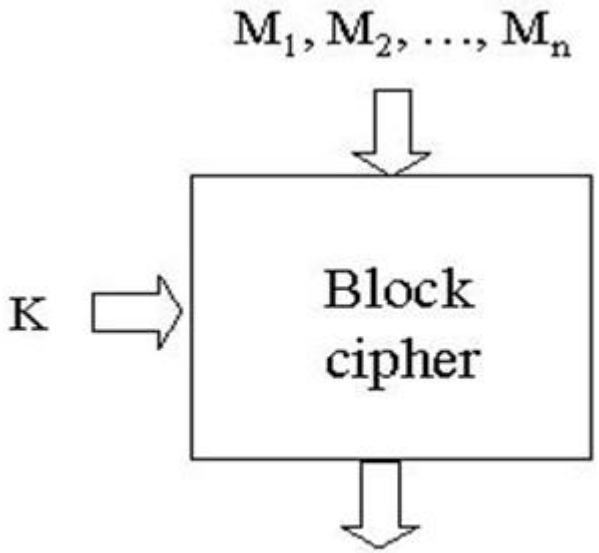
The One Time Pad (Vernam 1917)

Very fast enc/dec !!

… but long keys (as long as plaintext)

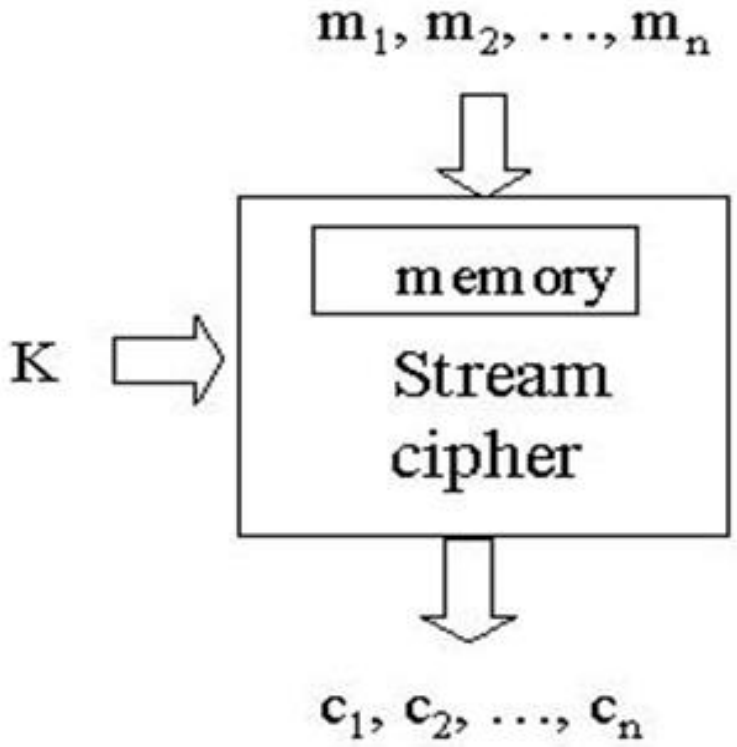
Is the OTP secure?

Block Vs Stream Cipher



$C_i = f_K(M_i)$

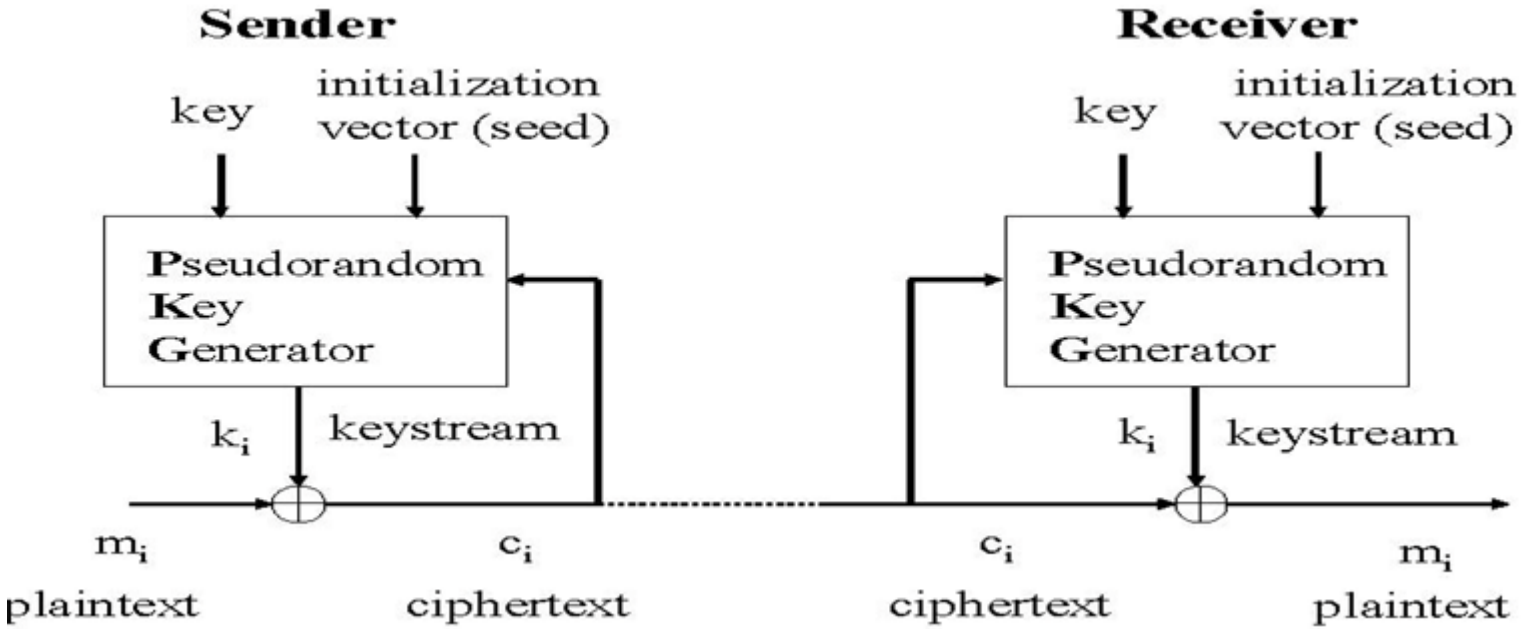
Every block of ciphertext is a function of only **one** corresponding **block** of plaintext



$c_i = f_K(m_i, m_{i-1}, \ldots, m_2, m_1)$

Every block of ciphertext is a function of the **current and all proceeding blocks** of plaintext

Typical Stream Cipher



Main objective of a stream cipher construction is to get K as much random as possible.

Randomness Measurements

- Randomness of sequence: unpredictability property of sequence.
- To measure randomness of the sequence generated by a deterministic method
  - Take a sample output sequence and subject it to various statistical tests to determine whether the sequence posses certain kinds of attributes, a truly random sequence would be likely to exhibit.
  - This is the reason the sequence is called pseudo-random sequence instead of random sequence and the generator is called pseudo-random sequence generator (PSG) in literature.

Golomb's Randomness Postulates

R-1 : In every period, the number of 1's differ from the number of 0's by atmost 1.

R-2 : In every period, half the runs have length 1, 1/4th have length 2, 1/8th have length 3, etc., as long as the number of runs so indicted exceeds 1. Moreover, for each of the sequence lengths, there are (almost) equally many runs of 0's and of 1's

R-3: The auto correlation function $C(\tau) = \sum_{i=0}^{N-1} (-1)^{S_i + S_{i+\tau}}$ is 2-valued. Explicitly

$$C(\tau) = \begin{cases} N & if \quad \tau \equiv 0 \pmod{N} \\ T & if \quad \tau \neq 0 \pmod{N} \end{cases}$$

where T is a constant.

Example 2 :Consider the periodic sequence s of period 15 with cycles 15 =

011 001 000 111 101.

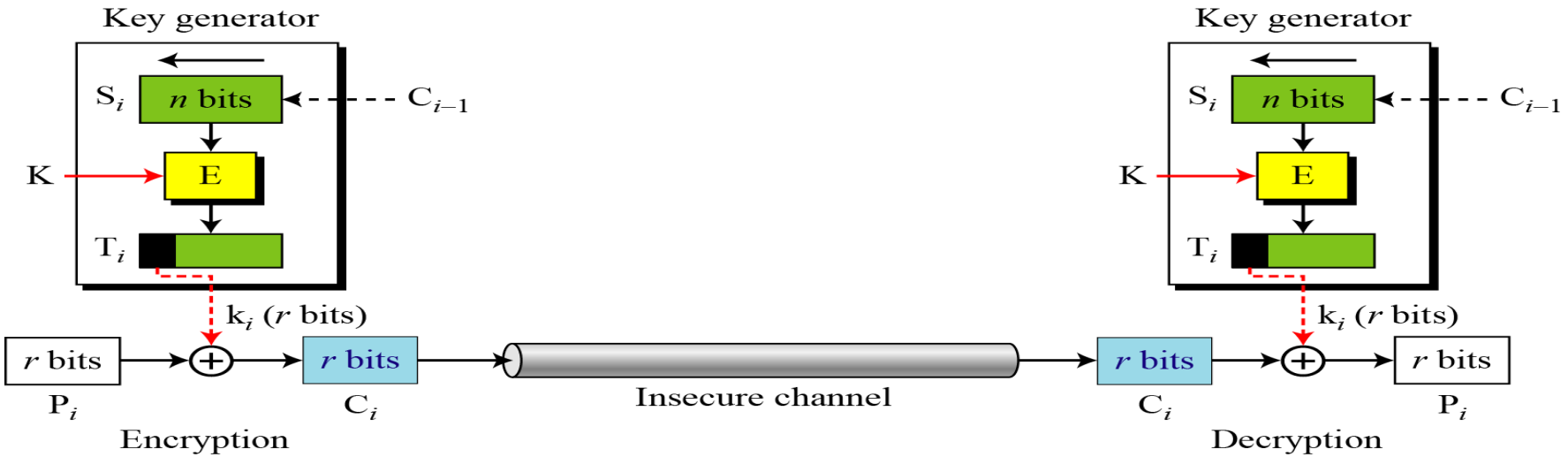
R-1: There are seven 0's and eight 1's.

R-2: Total runs is 8. 4 runs of length 1 (2 for each 0's and 1's), 2 runs of length 2 (1 for each 0's and 1's), 1 run of 0's of length 3 and 1 run of 1's of length 4.

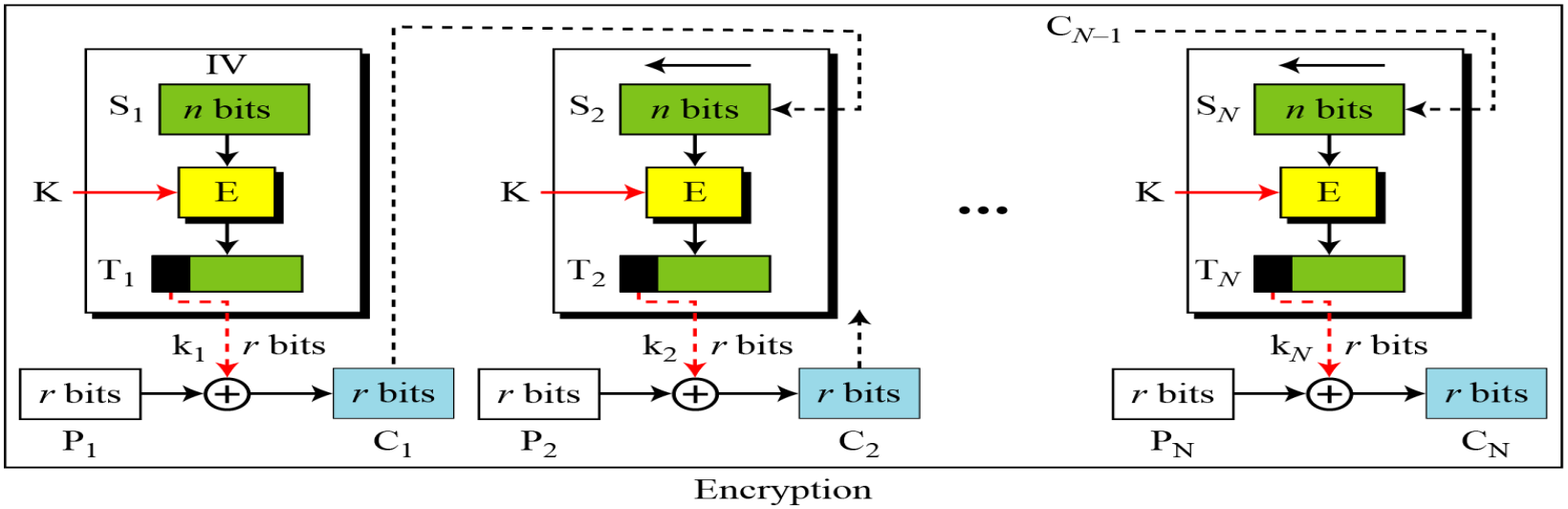
R-3: The function $C(\tau)$ takes only two values: $C(0) = 15$ and $C(\tau) = -1$ for $1 \leq \tau \leq 14$.

CFB as a Stream cipher

CFB: $C_1 = M_1$ xor Ss[E(K,IV)]

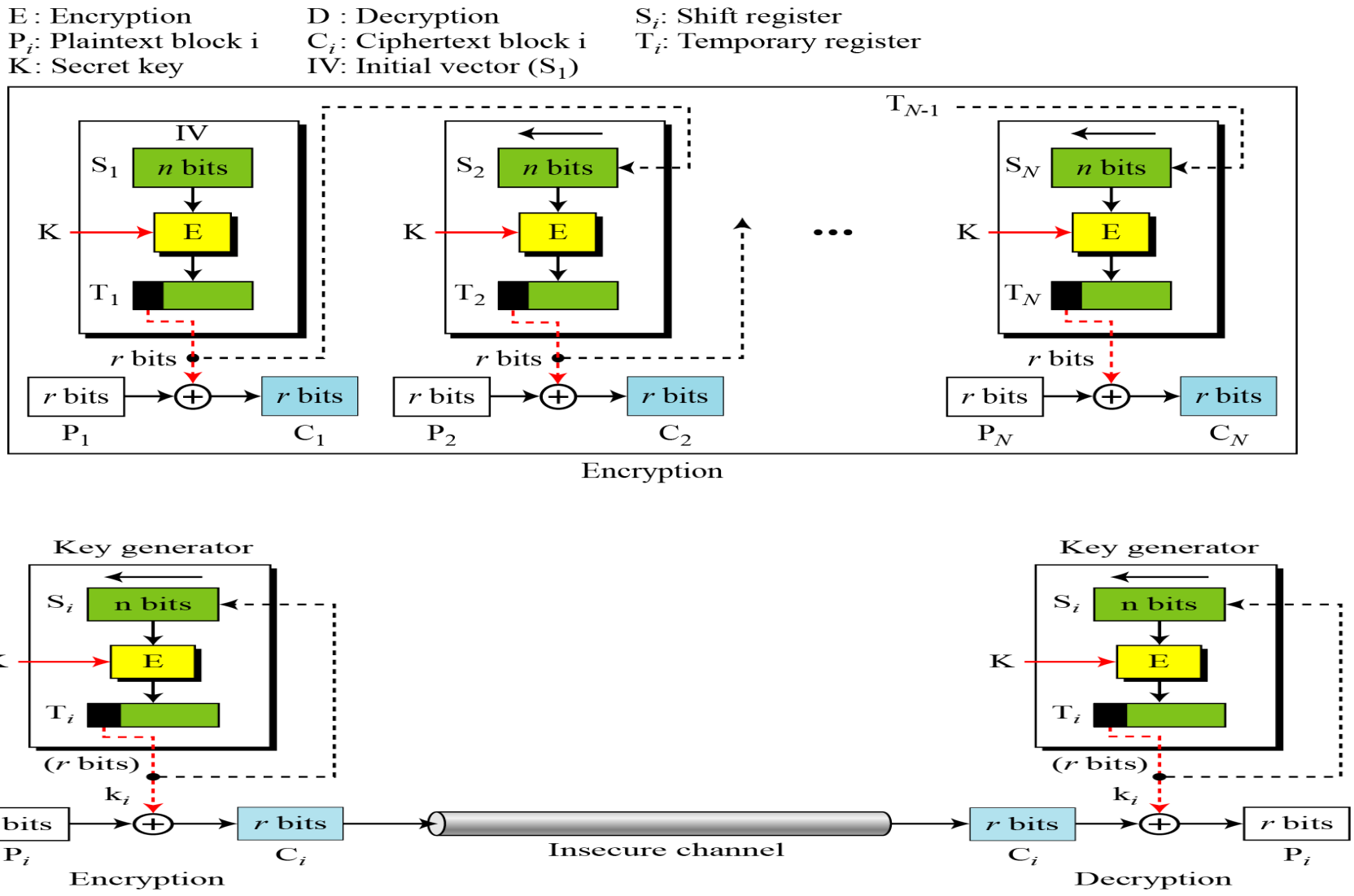


E : Encryption
D : Decryption
S_i: Shift register
P_i: Plaintext block i
C_i: Ciphertext block i
T_i: Temporary register
K: Secret key
IV: Initial vector (S_1)



OFB mode as Stream cipher

OFB: $C_1 = M_1$ xor Ss[E(K,IV)]

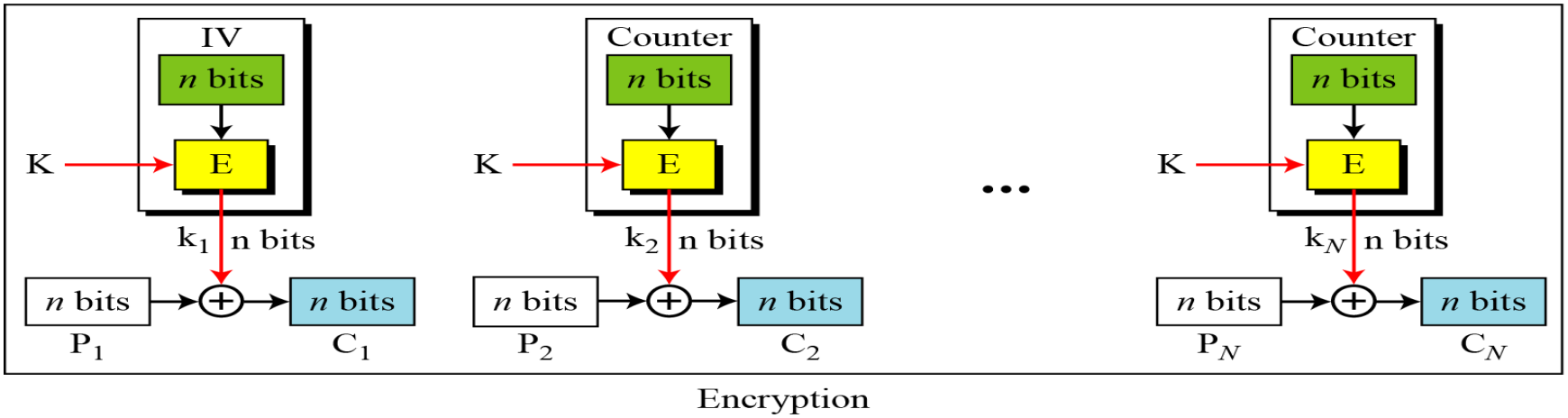


CTR mode as Stream Cipher

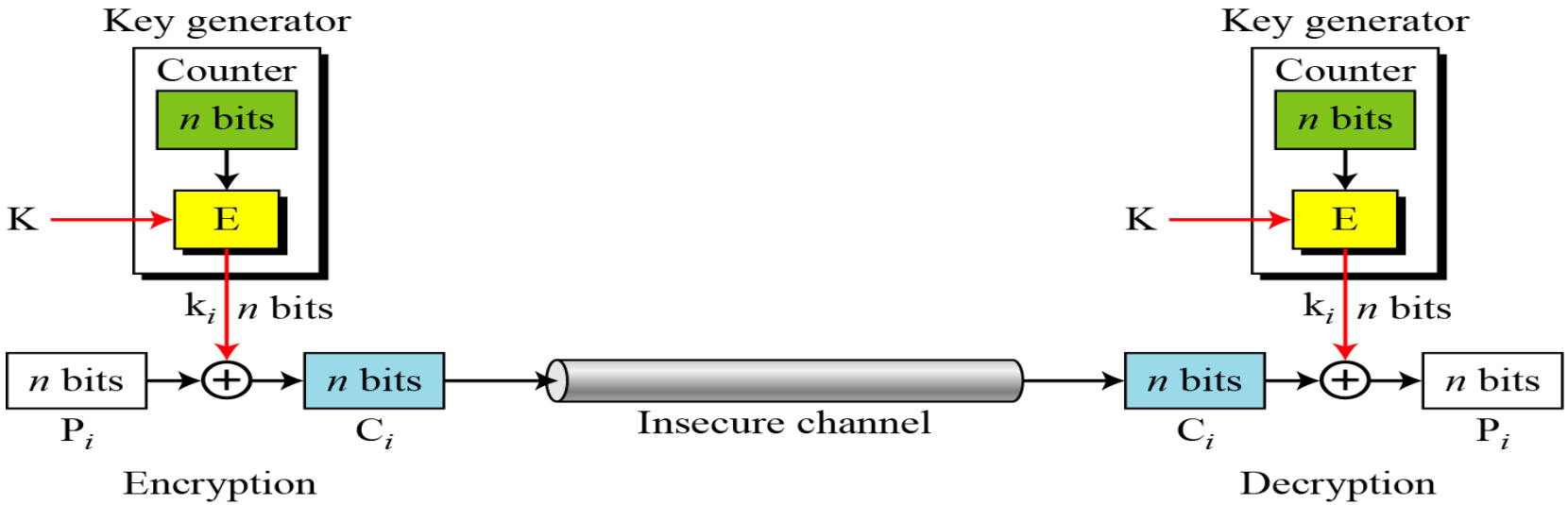
E : Encryption
P_i : Plaintext block *i*
K : Secret key

IV: Initialization vector
C_i : Ciphertext block *i*
k_i : Encryption key *i*

The counter is incremented for each block.



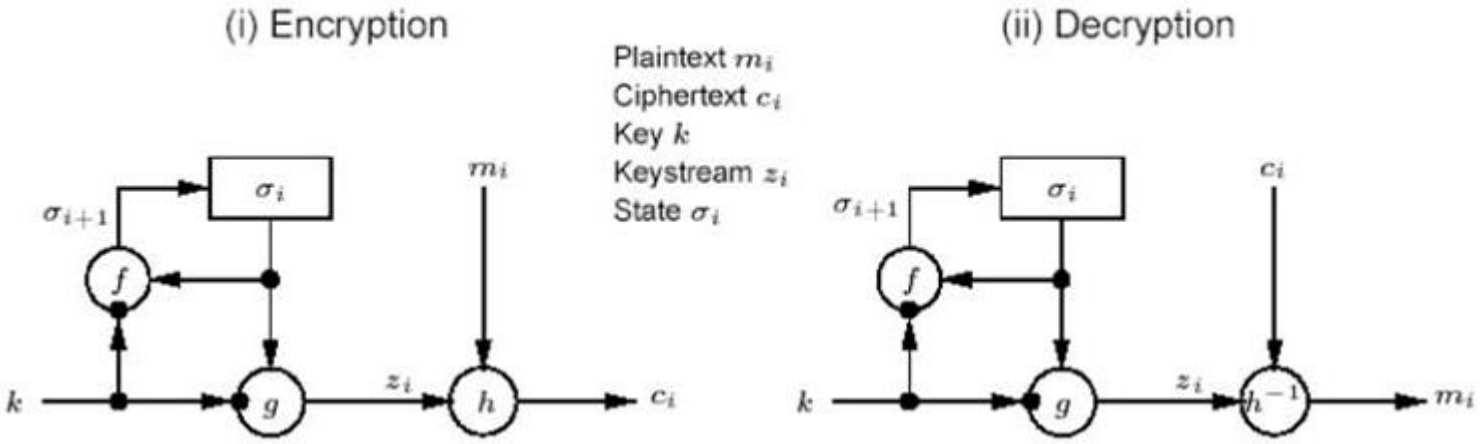
Encryption



Types of Stream Ciphers

- Synchronous Stream Cipher
  - Keystream is independent of plaintext/ciphertext
  - No error propagation
  - Sender and Receiver must be synchronized using the same key and operating at the same state within that key
  - Synchronization is a problem if ciphertext lost
- 'Asynchronous Stream Cipher
  - Keystream depends on plaintext/ciphertext
  - Error propagation
- Self-synchronizing Stream Cipher
  - Keystream depends on finite ciphertext and key
  - Synchronization `automatic

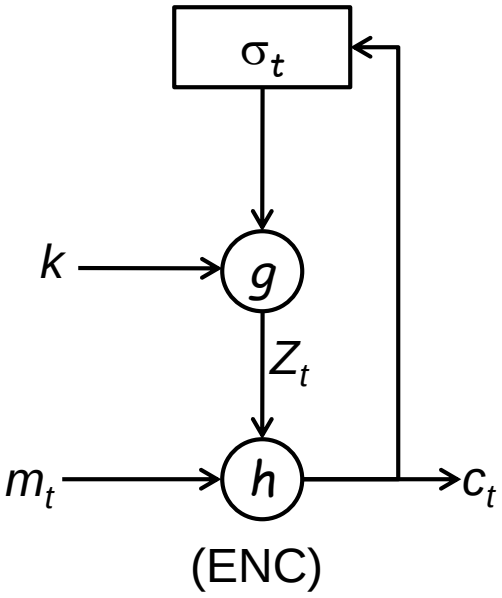
General Model: Synchronous Stream Cipher



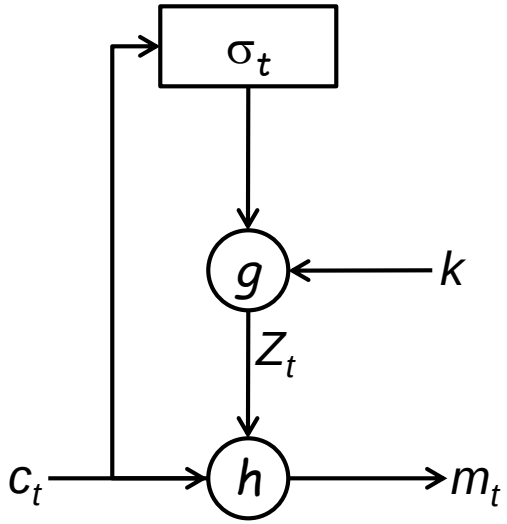
Next State Function
Key Stream Function
Output Function

$$\begin{aligned} \sigma_{i+1} &= f(\sigma_i, k), \\ z_i &= g(\sigma_i, k), \\ c_i &= h(z_i, m_i). \end{aligned}$$

Self Synchronizing Stream Cipher



(ENC)



(DEC)

$$\sigma_i = (C_{i-t}, C_{i-t+1} \ldots\ldots C_{i-1})$$

$$Z_i = g(\sigma_i, k)$$

$$C_i = h(Z_i, m_i)$$

$$\sigma_o = (C_{-t}, \ldots\ldots C_{-1})$$

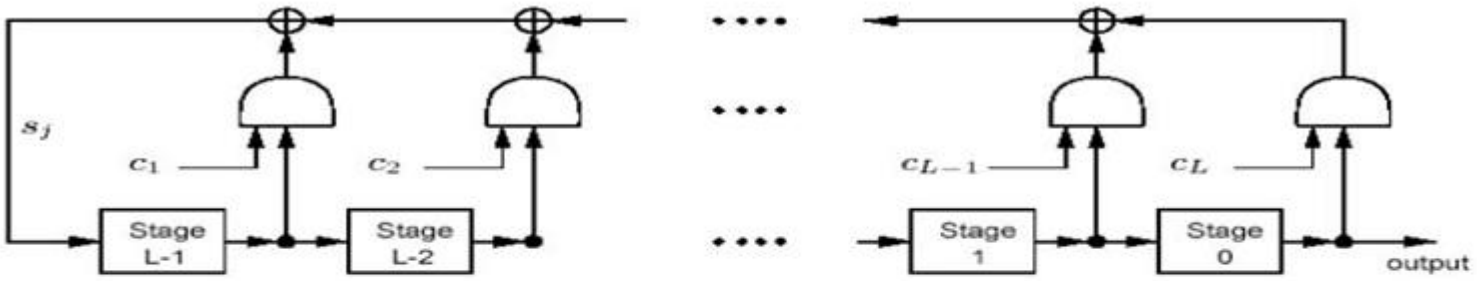
Linear feedback shift registers (LFSRs)

- LFSRs are used in many of the keystream generators
  - Well-suited to hardware implementation;
  - can produce sequences of large period
  - Can produce sequences with good statistical properties
  - can be readily analyzed using algebraic techniques.

LFSR

- An LFSR of length L consists of L stages numbered 0,1, . . . , L-1 each storing one bit and having one input and one output; together with a clock which controls the movement of data.
- During each unit of time the following operations are performed
  I. The content of stage 0 is output and form the part of the output sequence
  II. The content of stage i is moved to stage i-1; and
  III. The new content of stage L-1 is the feedback bit which is calculated by adding together modulo 2 the previous content of a fixed subset of stages 0,1,. . . , L -1

Linear Feedback Shift Register (LFSR)

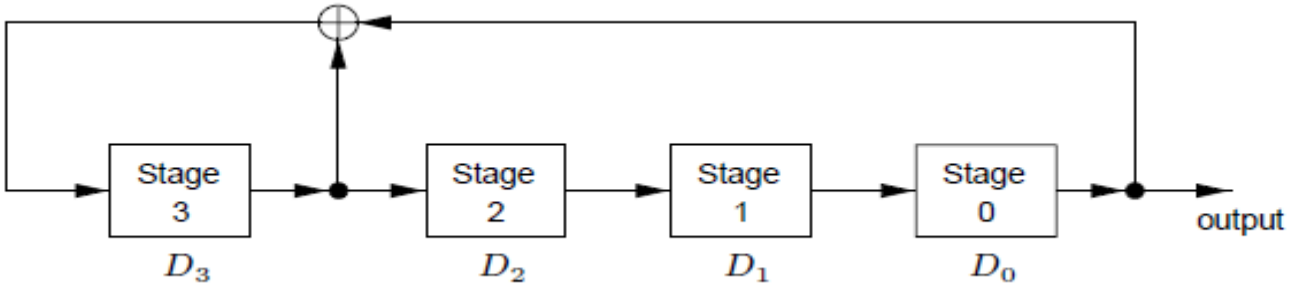


$\langle L, C(D) \rangle$

Length **Connection polynomial, C(D)**

$$C(D) = 1 + c_1 D + c_2 D^2 + \ldots + c_L D^L$$

EX.: $LFSR\langle 4, 1 + D + D^4 \rangle$



LFSR (4,1+D+D^4)

t	D_3	D_2	D_1	D_0
0	0	1	1	0
1	0	0	1	1
2	1	0	0	1
3	0	1	0	0
4	0	0	1	0
5	0	0	0	1
6	1	0	0	0
7	1	1	0	0

t	D_3	D_2	D_1	D_0
8	1	1	1	0
9	1	1	1	1
10	0	1	1	1
11	1	0	1	1
12	0	1	0	1
13	1	0	1	0
14	1	1	0	1
15	0	1	1	0

$$s = 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, \ldots,$$

Verify Random Property using Golomb's Postulates

- Thanks