SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

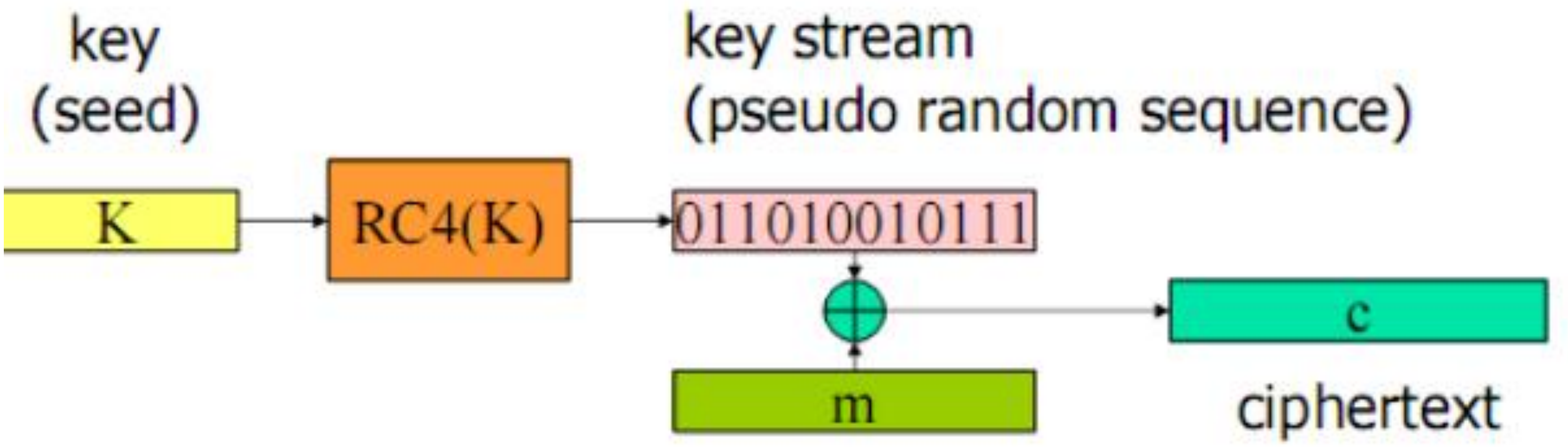
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE NAME : 19CS503 Cryptography and Network Security

III YEAR /V SEMESTER

Unit 2- SYMMETRIC KEY CRYPTOGRAPHY

Topic : RC4 – Key distribution


key
(seed)
key stream
(pseudo random sequence)
K
RC4(K)
01101001011 1
c
m
ciphertext

One-Time Pad

- Developed by Gilbert Vernam in 1918, another name: ***Vernam Cipher***
- The key
  - a truly random sequence of 0’s and 1’s
  - the same length as the message
  - use one time only
- The encryption
  - adding the key to the message modulo 2, bit by bit.

Encryption $c_i = m_i \oplus k_i \qquad i = 1,2,3,\ldots$

Decryption $m_i = c_i \oplus k_i \qquad i = 1,2,3,\ldots$

m_i : plain-text bits.
k_i : key (key-stream) bits
c_i : cipher-text bits.

Example

- **Encryption:**
- 1001001 1000110   plaintext
- 1010110 0110001   key
- 0011111 1110110   ciphertext

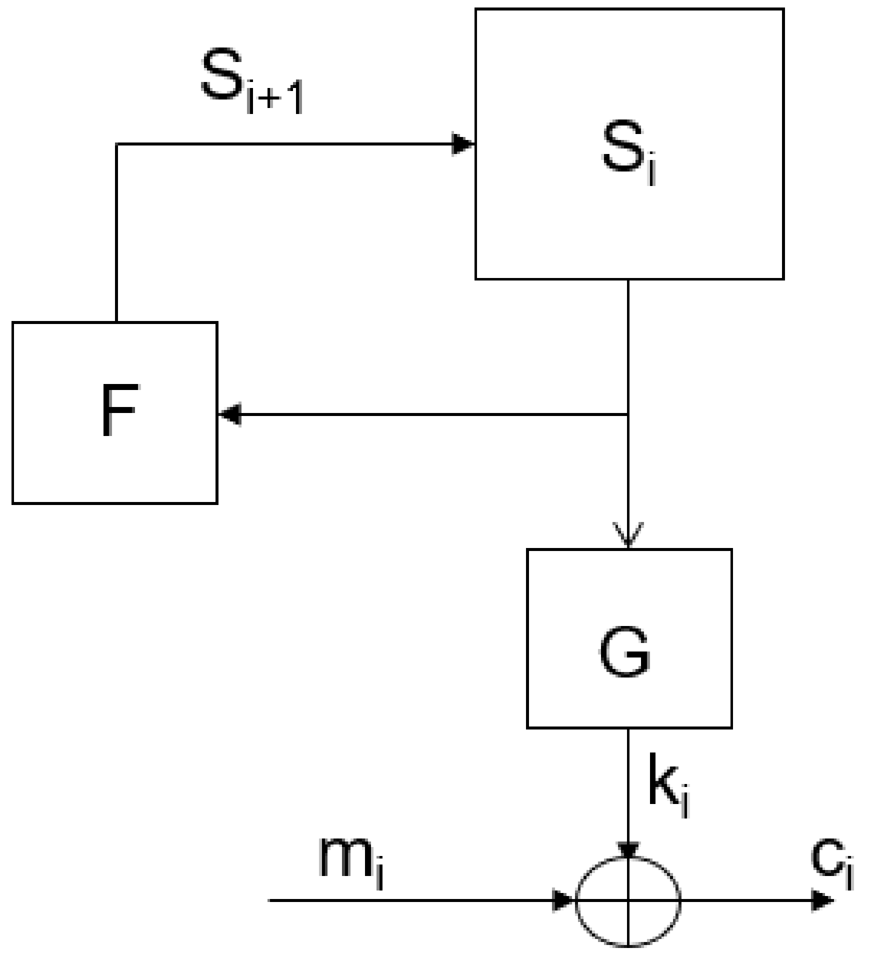
- **Decryption:**
- 0011111 1110110   ciphertext
- 1010110 0110001   key
- 1001001 1000110   plaintext

One-Time pad practical Problem

- Key-stream should be as long as plain-text
- Difficult in Key distribution & Management
- **Solution :**
  - Stream Ciphers
  - Key-stream is generated in pseudo-random fashion form Relatively short secret key

Stream Cipher Model

- **Output function** appears random



S_i : state of the cipher at time t = i.
F : state function.
G : output function.

Initial state, output and state functions are controlled by the secret key.

Random Numbers

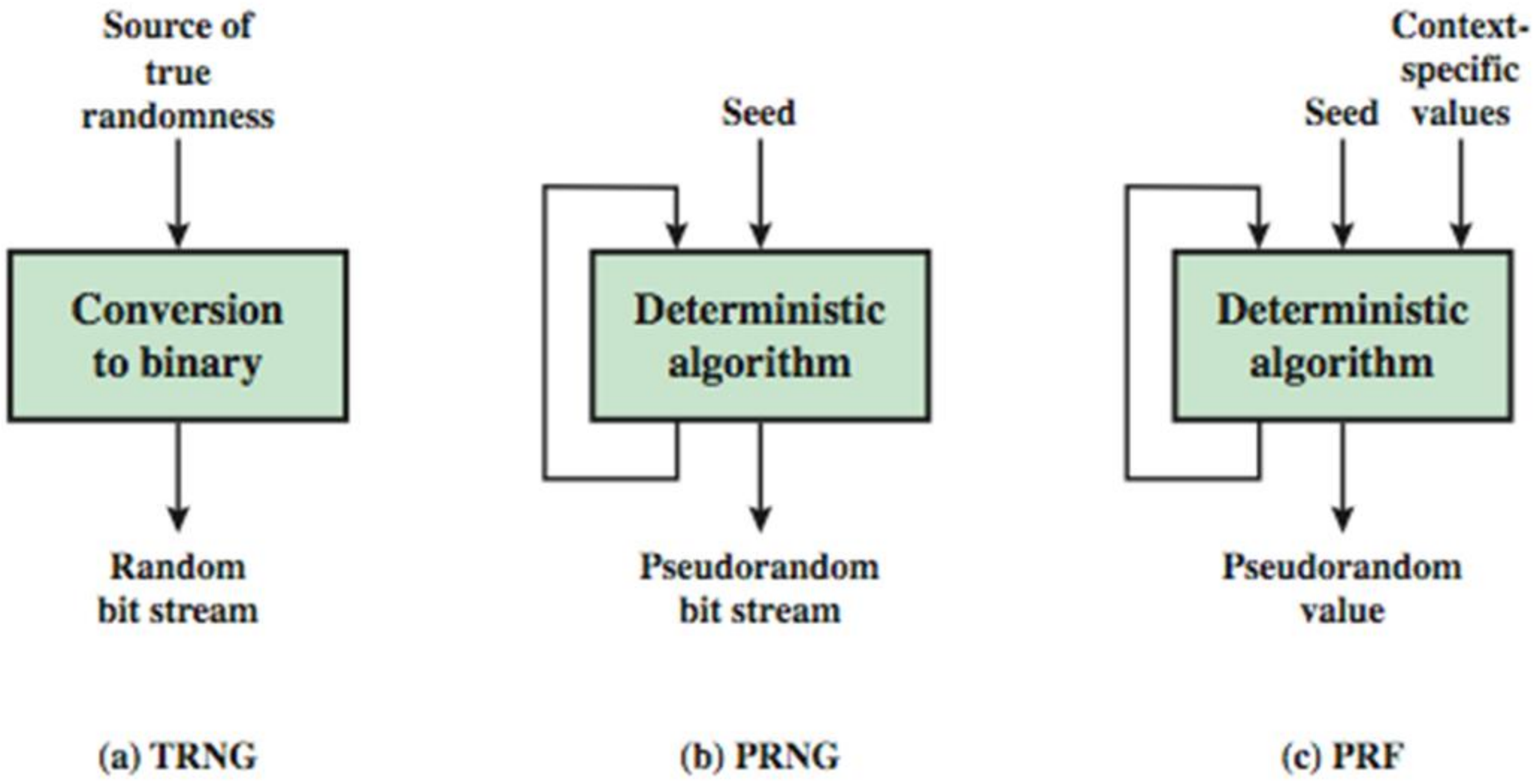
- Many uses of **random numbers** in cryptography
  - Nonce as Initialize Vector
  - Session keys
  - Public key generation
  - Keystream for a one-time pad
- In all cases its critical that these values be
  - statistically random, uniform distribution, independent
  - unpredictability of future values from previous values
- Care needed with generated random numbers

Pseudorandom Number Generators (PRNGs)

- Often use deterministic algorithmic techniques to create “random numbers”
  - although are not truly random
  - can pass many tests of “randomness”

- Known as “Pseudorandom Numbers”
- Created by “Pseudorandom Number Generators (PRNGs)”

Random & Pseudorandom Number Generators



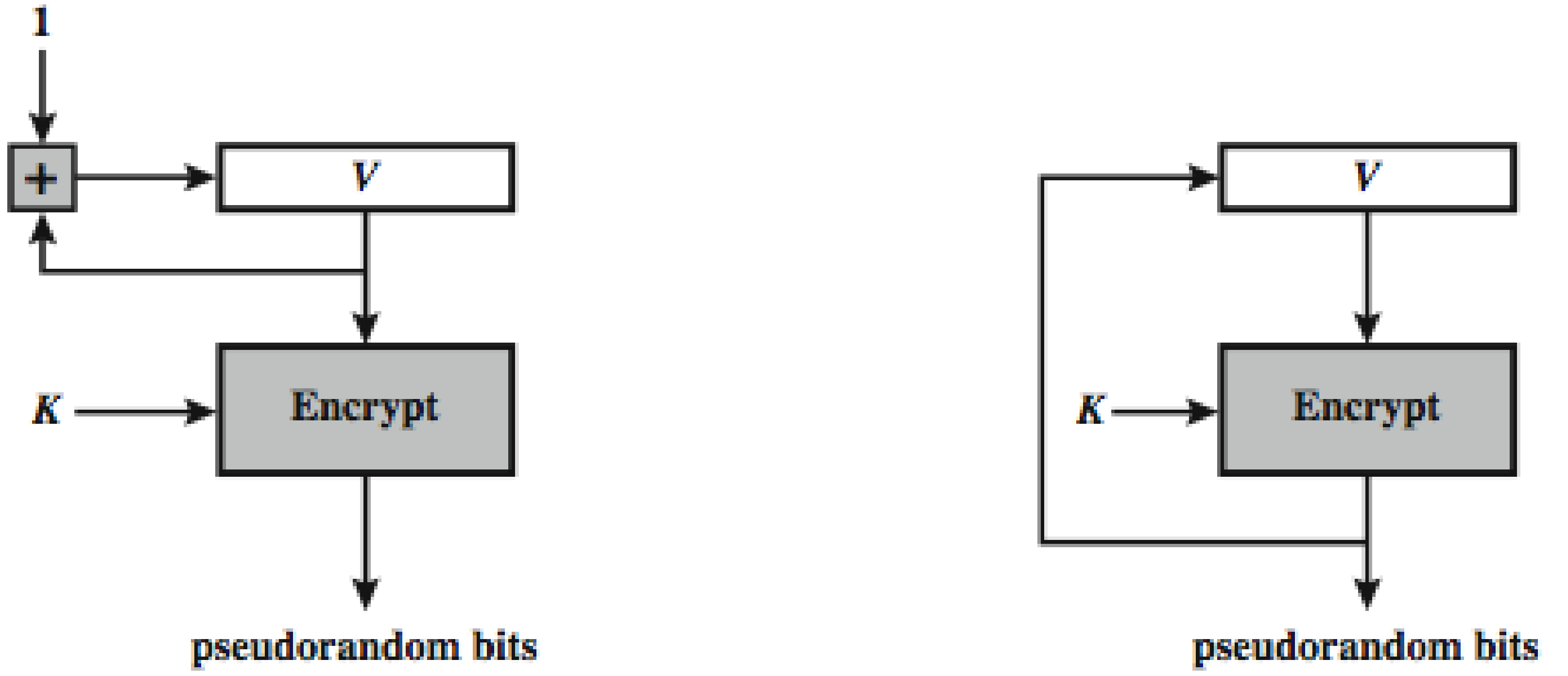
PRNG Requirements

- Randomness
  - uniformity, scalability, consistency
- Unpredictability
  - forward & backward Unpredictability
  - use same tests to check
- Characteristics of the seed
  - Secure
  - if known adversary can determine output
  - so must be random or pseudorandom number

Using Block Ciphers as PRNGs

- For cryptographic applications, can use a block cipher to generate random numbers
- Often for creating session keys from master key

- **CTR**
  - $X_i = E_K[V_i]$
- **OFB**
  - $X_i = E_K[X_i\text{-}1]$



(a) CTR Mode

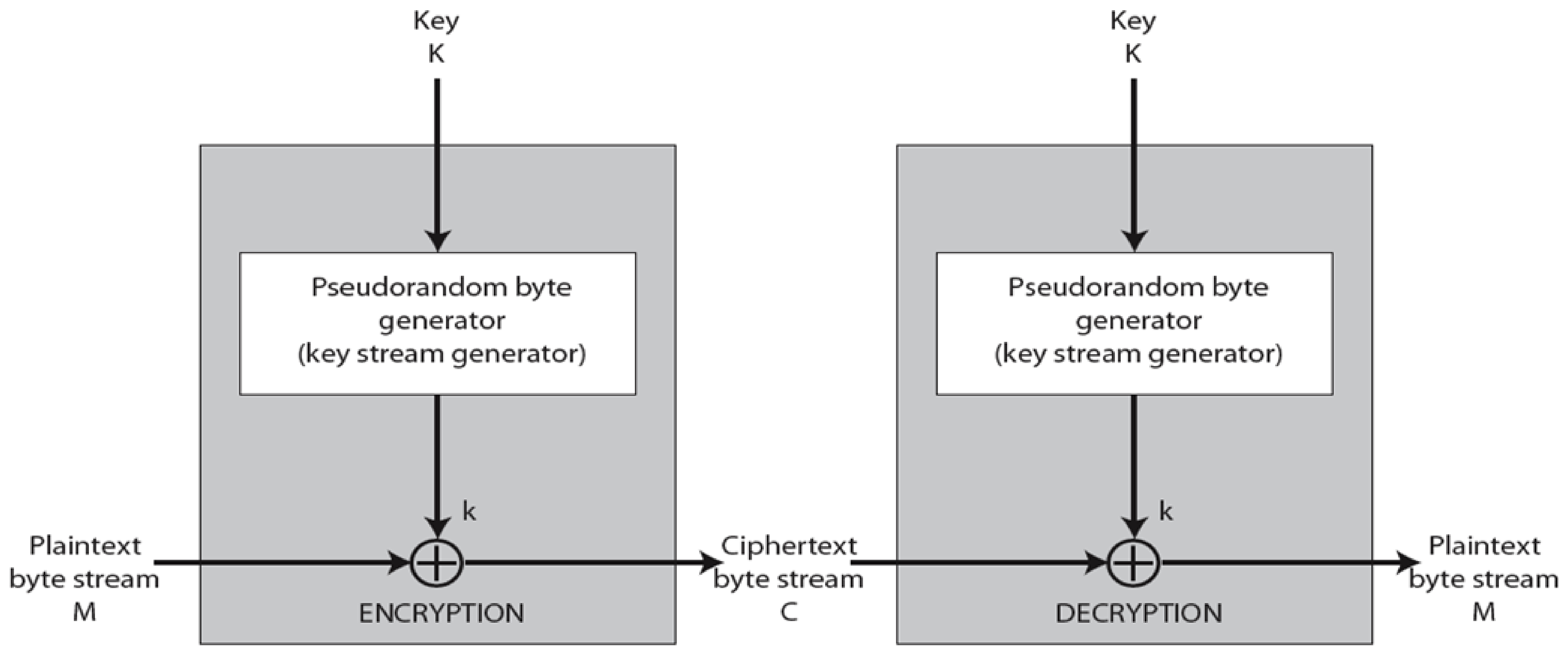
(b) OFB Mode

Stream Ciphers

- Generalization of **one-time pad**
- Stream cipher is initialized with short **key**
- Key is “stretched” into long **keystream**
  - have a pseudo random property
- Keystream is used like a one-time pad
  - XOR to encrypt or decrypt

Stream Cipher Structure

- Randomness of stream key completely destroys statistically properties in message
- Must **never reuse** stream key
  - otherwise can recover messages



Stream Cipher Properties

- Some design considerations are:
  - long period with no repetitions
  - statistically random
  - depends on large enough key
  - large linear complexity
- Properly designed, can be as secure as a block cipher with same size key
- Benefit : usually ***simpler*** & ***faster***

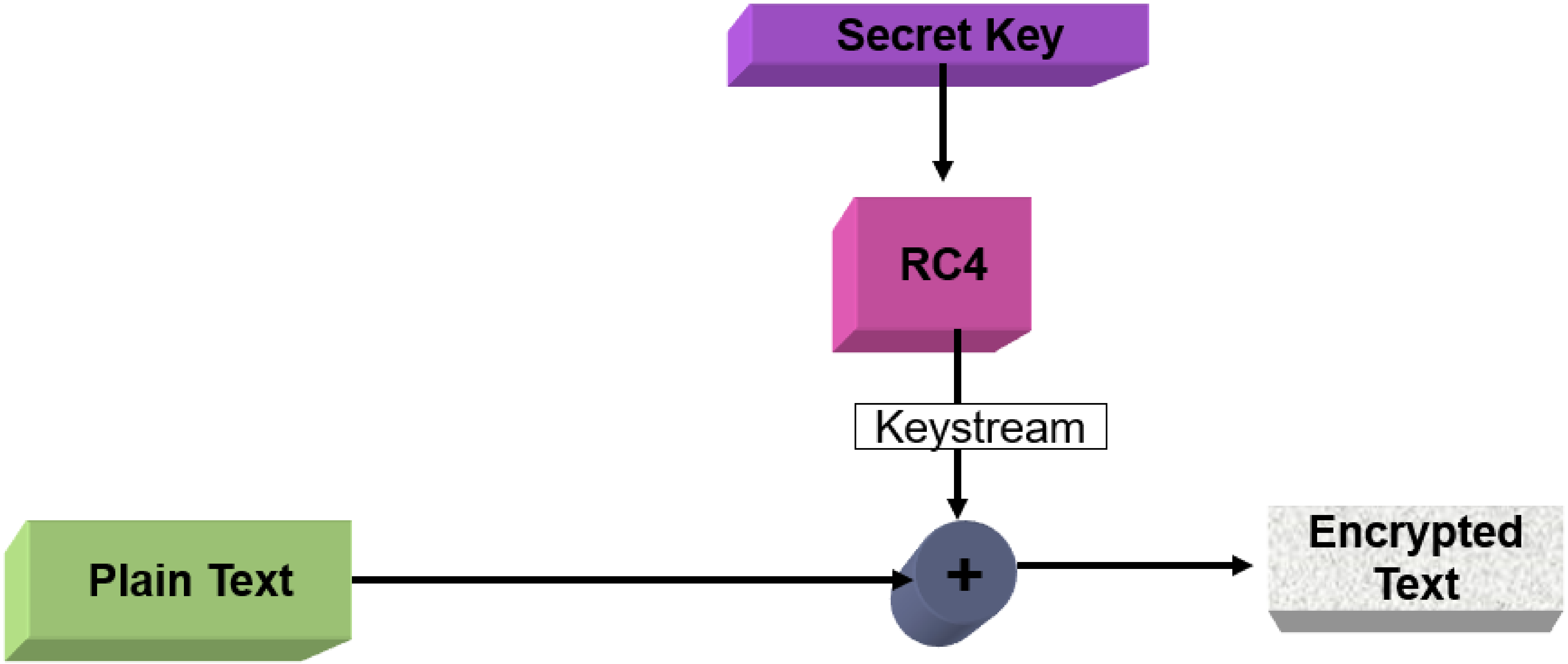
RC4 Basics

- A symmetric key encryption algorithm invented by Ron Rivest
  - A proprietary cipher owned by RSA, kept secret
  - Code released anonymously in Cyberpunks mailing list in 1994
  - Later posted sci.crypt newsgroup
- **Variable key size**, **byte-oriented** stream cipher
  - Normally uses 64 bit and 128 bit key sizes.
- Used in
  - SSL/TLS (Secure socket, transport layer security) between web browsers and servers,
  - IEEE 802.11 wirelss LAN std: WEP (Wired Equivalent Privacy), WPA (WiFi Protocol Access) protocol

RC4-based Usage

- WEP
- WPA default
- Bit Torrent Protocol Encryption
- Microsoft Point-to-Point Encryption
- SSL (optionally)
- SSH (optionally)
- Remote Desktop Protocol
- Kerberos (optionally)

RC4 Block Diagram



Cryptographically very strong and easy to implement

RC4 ...Inside

- Consists of 2 parts:
  - Key Scheduling Algorithm (KSA)
  - Pseudo-Random Generation Algorithm (PRGA)

- KSA
  - Generate State array
- PRGA on the KSA
  - Generate keystream
  - XOR keystream with the data to generated encrypted stream

The KSA

- Use the secret key to initialize and **permutation** of state vector **S**, done in two steps
- Use 8-bit index pointers **i** and **j**

1

```
for i = 0 to 255 do
    S[i] = i;
    T[i] = K[i mod(|K|)]);
```

[S], S is set equal to the values from 0 to 255
S[0]=0, S[1]=1,..., S[255]=255
[T], A temporary vector
[K], Array of bytes of secret key
|K| = Keylen, Length of (K)

2

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

- Use T to produce initial permutation of S
- The only operation on S is a swap; S still contains number from 0 to 255

After KSA, the input key and the temporary vector T will be no longer used

The PRGA

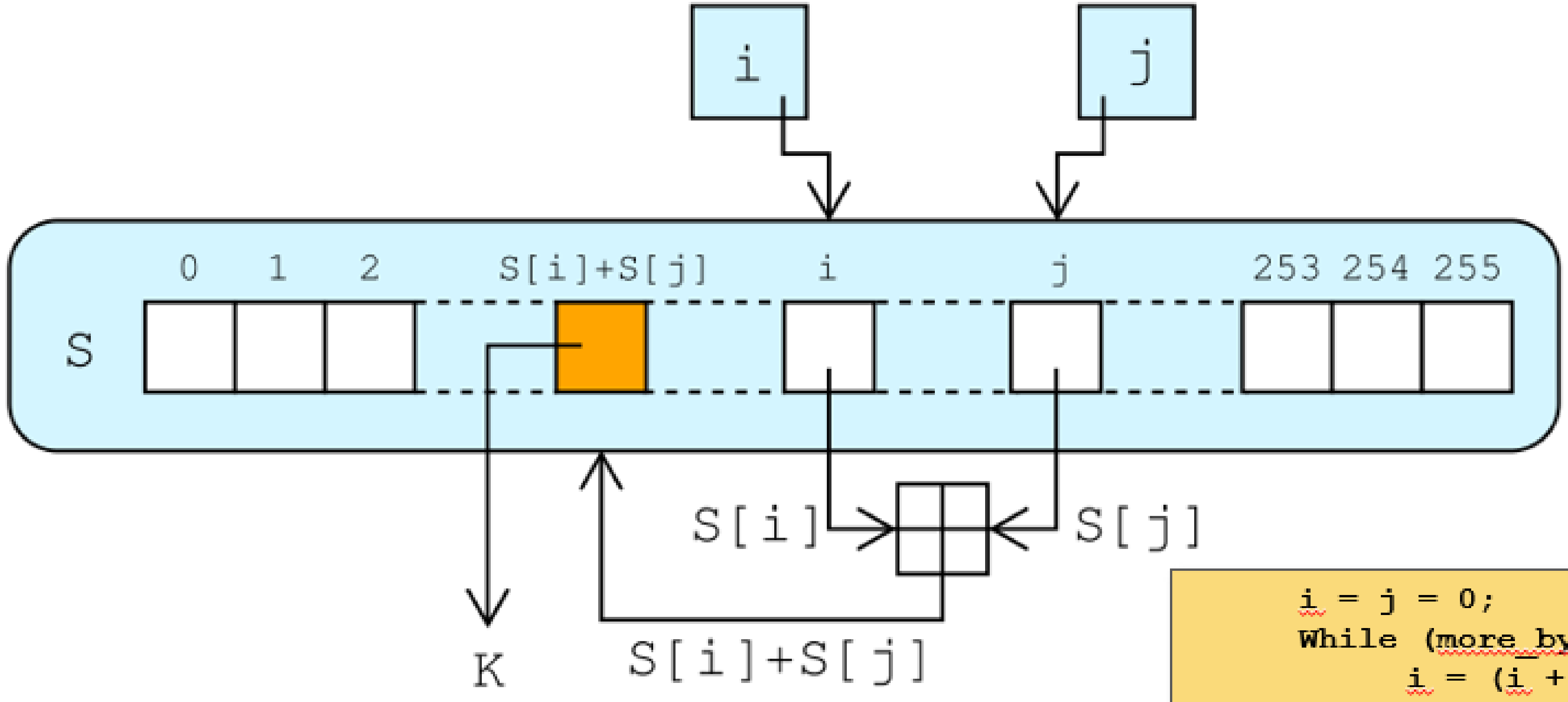
- Generate key stream k , one by one
- XOR S[k] with next byte of message to encrypt/decrypt

```
i = j = 0;
While (more_byte_to_encrypt)
    i = (i + 1) (mod 256);
    j = (j + S[i]) (mod 256);
    swap(S[i], S[j]);
    k = (S[i] + S[j]) (mod 256);
    Ci = Mi XOR S[k];
```

Sum of shuffled pair selects "stream key" value from permutation

RC4 Lookup Stage

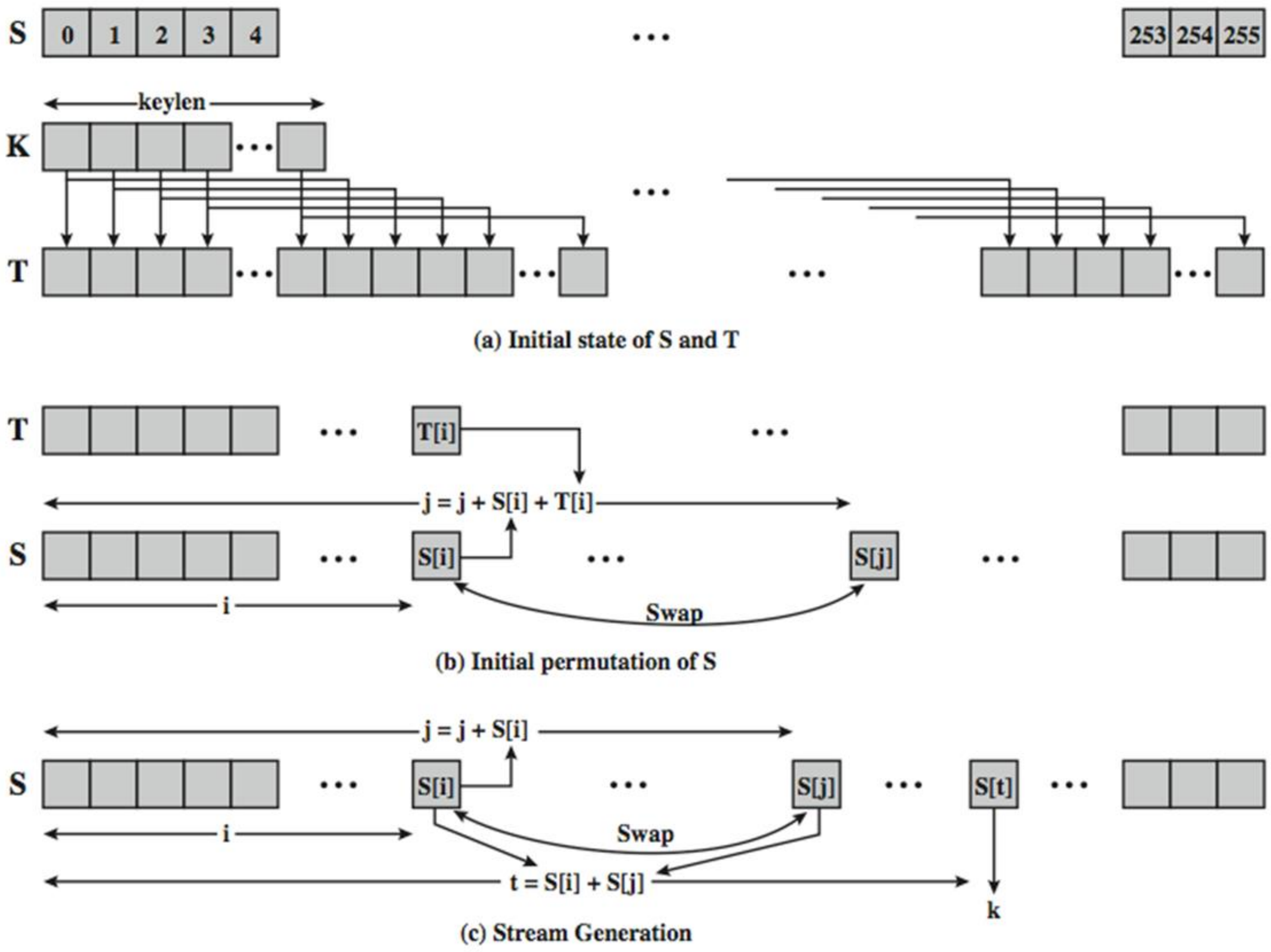
- The output byte is selected by looking up the values of S[i] and S[j], adding them together modulo 256, and then looking up the sum in S
- S [S[i] + S[j]] is used as a byte of the key stream, *K*



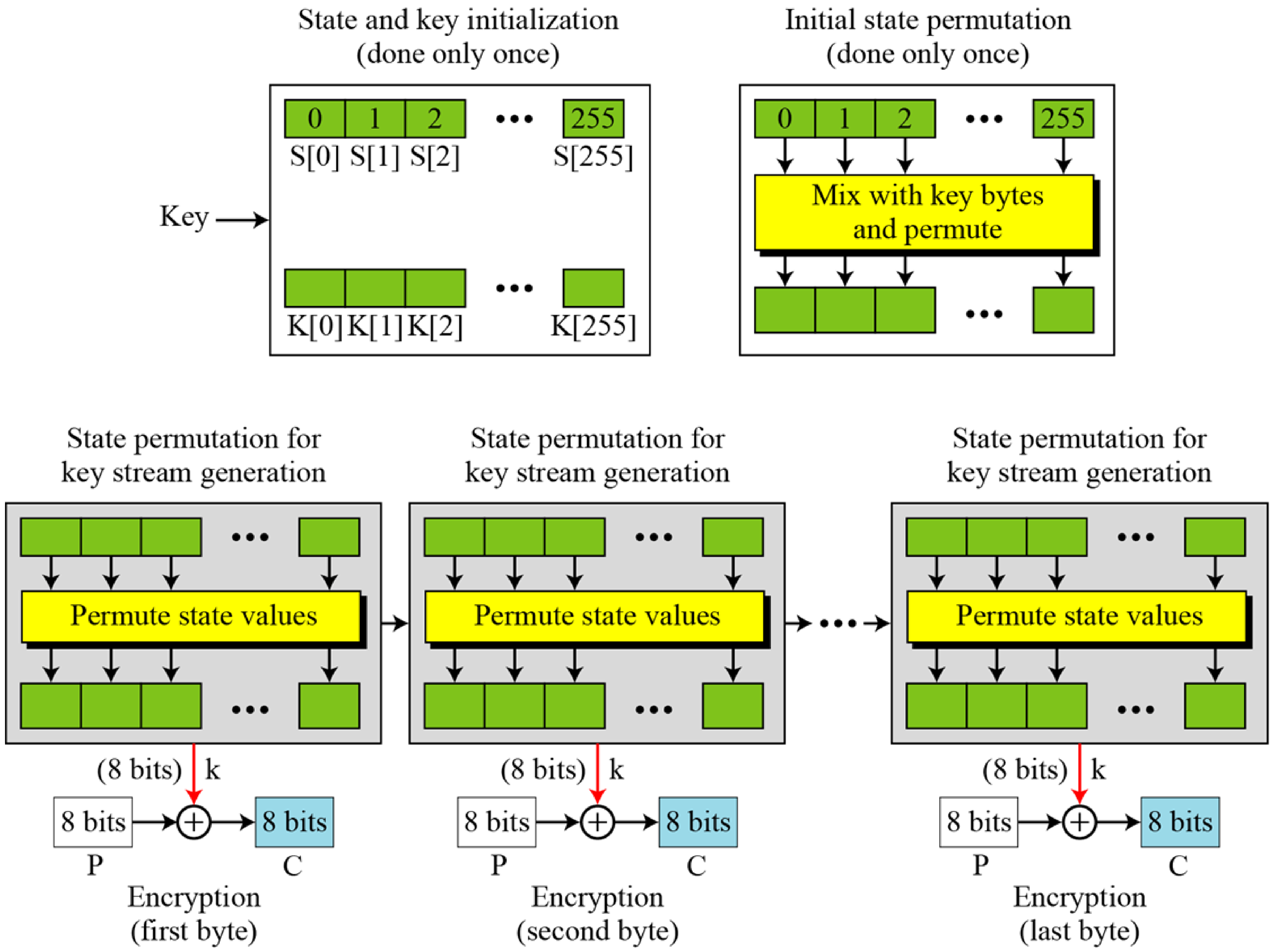
http://en.wikipedia.org/wiki/File:RC4.svg

```
i = j = 0;
While (more_byte_to_encrypt)
    i = (i + 1) (mod 256);
    j = (j + S[i]) (mod 256);
    swap(S[i], S[j]);
    k = (S[i] + S[j]) (mod 256);
    Ci = Mi XOR S[k];
```

Detailed Diagram



Overall Operation of RC4



Decryption using RC4

- Use the same secret key as during the encryption phase.
- Generate keystream by running the KSA and PRGA.
- XOR keystream with the encrypted text to generate the plain text.
- Logic is simple :

(A xor B) xor B = A

A = Plain Text or Data
B = KeyStream

RC4 and WEP

- WEP is a protocol using RC4 to encrypt packets for transmission over IEEE 802.11 wireless LAN.
- WEP requires each packet to be encrypted with a separate RC4 key.
- The RC4 key for each packet is a concatenation of a 24-bit IV (initialization vector) and a 40 or 104-bit long-term key.

RC4 key: IV (24) Long-term key (40 or 104 bits)

RC4 and WEP

- WEP is a protocol using RC4 to encrypt packets for transmission over IEEE 802.11 wireless LAN.
- WEP requires each packet to be encrypted with a separate RC4 key.
- The RC4 key for each packet is a concatenation of a 24-bit IV (initialization vector) and a 40 or 104-bit long-term key.

RC4 key: IV (24) Long-term key (40 or 104 bits)

Assessment 1

1. What are the allowable values of word size in bit for RC4 algorithm?
   a) 16, 32
   b) 16, 32, 64
   c) 8, 16, 32
   d) 16, 32, 48

2. The number of rounds in RC4 can range from 0 to ___________
   a) 127
   b) 63
   c) 255
   d) 31

REFERENCES

1. William Stallings, Cryptography and Network Security, 6 th Edition, Pearson Education, March 2013.

THANK YOU