Block Cipher Modes

Data and Information Management: ELEN 3015

School of Electrical and Information Engineering, University of the Witwatersrand

March 25, 2010

Overview

Motivation for Cryptographic Modes

Electronic Codebook Mode (ECB)

Cipher Block Chaining (CBC)

Cipher Feedback Mode (CFB)

Output Feedback Mode (OFB)

1. Cryptographic Modes

Problem:

With block ciphers, same plaintext block always enciphers to the same ciphertext block under the same key

1. Cryptographic Modes

Solution:

Cryptographic mode:

- block cipher
- feedback
- simple operations

Simple operations, as the security lies in the cipher.

1. Cryptographic Modes

1.1 Considerations

- The mode should not compromise security of cipher
- Mode should conceal patterns in plaintext
- Some random starting point is needed
- Difficult to manipulate the plaintext by changing ciphertext
- Requires multiple messages to be encrypted with same key
- No significant impact on efficiency of cipher
- Ciphertext same size as plaintext
- Fault tolerance - recover from errors

2. Electronic Codebook Mode

Uses the block cipher without modifications

Same plaintext block encrypts to same ciphertext under same key

Each plaintext block is encrypted independently of other plaintext blocks.

Corrupted bits only affects one block

Dropped/inserted bits cause sync errors $\longrightarrow$ all subsequent blocks decipher incorrectly

2. Electronic Codebook Mode

2.1 Advantages

ECB exhibits 'random access property' because plaintext blocks are encrypted independently

- Encryption and decryption can be done in any order
- Beneficial for databases, records can be added, deleted, modified, encrypted and deleted independently of other records

Parallel implementation

- Different blocks can simultaneously be decrypted on separate processors

Many messages can be encrypted with the same key, since each block is independent.

2. Electronic Codebook Mode

2.2 Disadvantage

An interceptor can build up a codebook of messages $\longrightarrow$ making decryption very easy

E.g. if he determines (by chosen plaintext attack say) that '5*e*081*bc*' encrypts to '7*ea*593*a*4', whenever this appears in another message, he can decrypt it.

Stereotyped beginnings and endings a problem, since many messages have standard formats, such as headers and footers.

2.3 Padding

ECB works on 64-bit blocks

When last block is smaller than 64 bits, need to use padding

Three types of padding:

- Padding sequence
- EOF-character
- Ciphertext stealing

2.3.1 Padding sequence

Pad with a sequence (zeros/ones, alternating ones and zeros, etc)

Last byte indicate the number of padding characters.

E.g. block consisting of 3 bytes (24 bits). Must add 5 bytes. Add 4 bytes of zeros and the last byte with the number 5.

After decryption delete the number of padding characters contained in the last byte of the plaintext.

Problem $\longrightarrow$ message fitting exactly into the block $\longrightarrow$ need to pad an entire block.

2. Electronic Codebook Mode

Byte								
1	1	0	0	1	0	1	1	0
2	0	1	0	0	0	1	1	1
3	1	1	1	0	1	1	0	0
4	0	0	1	0	0	1	1	0
5	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
8	0	0	0	0	0	1	0	0

Message

Padding Bits

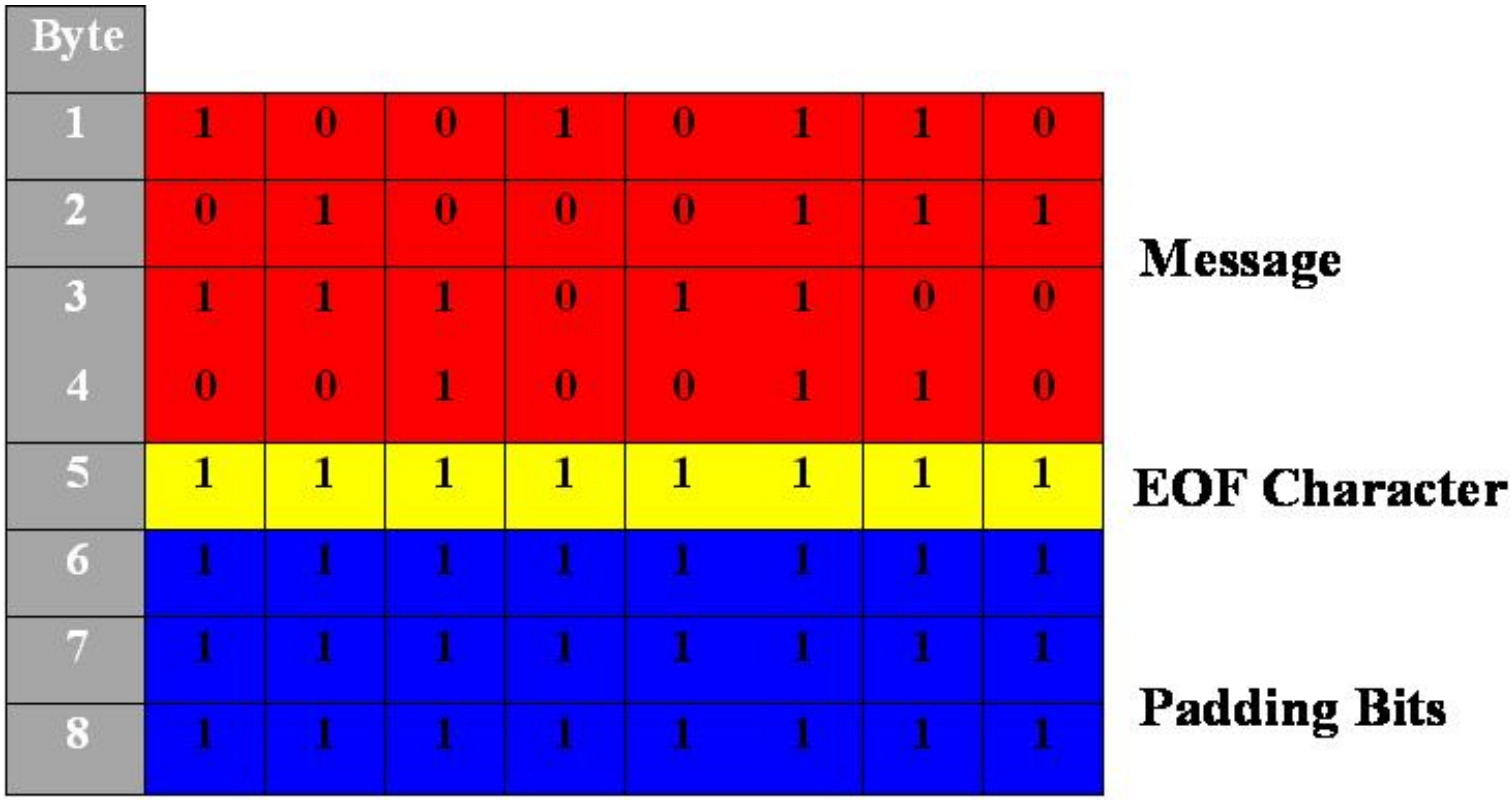
Padding Count (4 bytes)

2.3.2 EOF Character

Use a designated end-of-file character

EOF character indicates the last plaintext byte, padding follows

2. Electronic Codebook Mode



2.3.3 Ciphertext Stealing

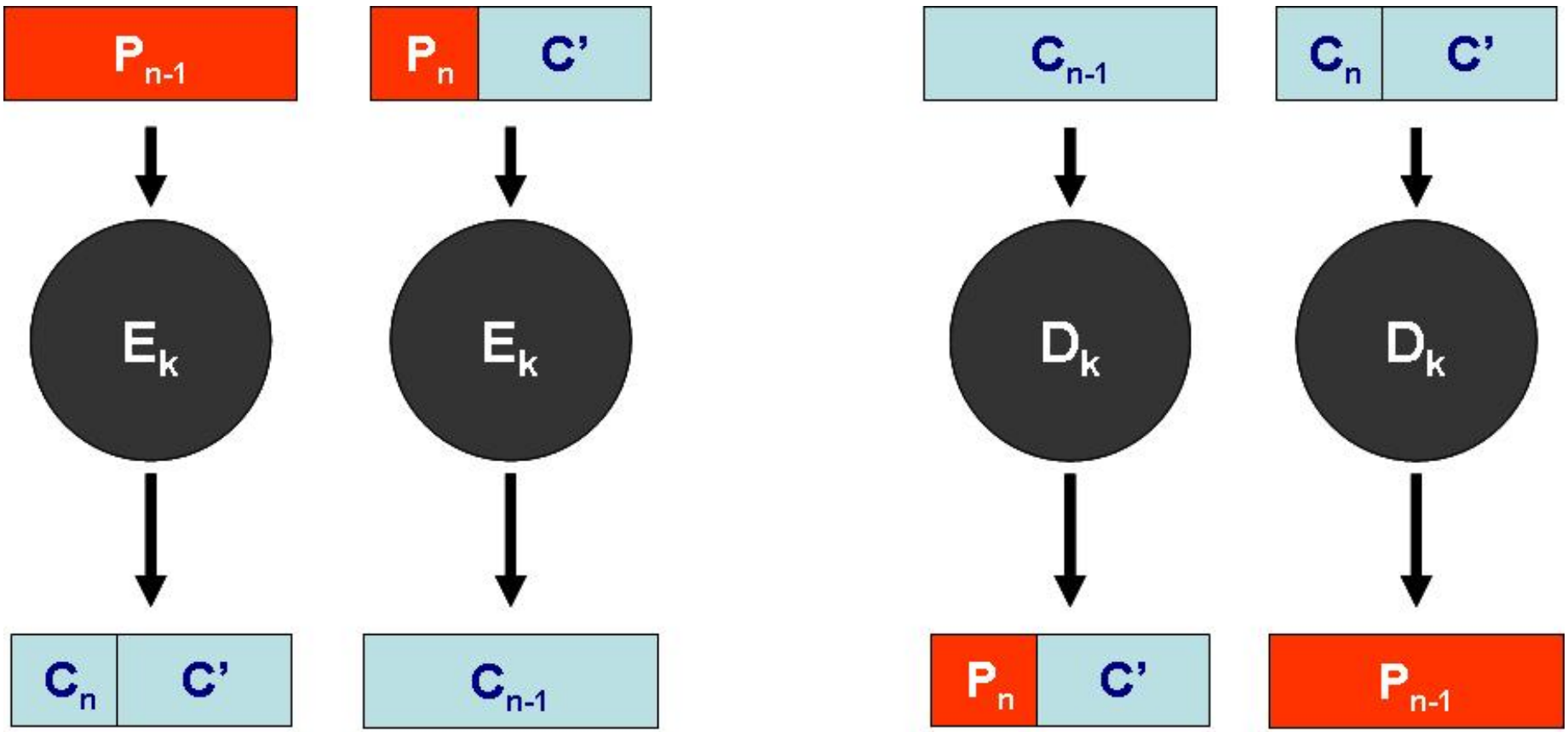
When **length**(Ciphertext) must equal **length**(Plaintext)

1. $E_k(P_{n-1}) = [C_n : C']$; **length**$(C_n)$ = **length**(P_n)
2. $E_k([P_n : C']) = C_{n-1}$
3. Transmits C_1, C_2, ..., C_{n-1}, C_n

NB: C' doesn't need to be transmitted, can be recovered from C_{n-1}.

All plaintext is enciphered, while **length**(ciphertext) = **length**(plaintext)

2. Electronic Codebook Mode



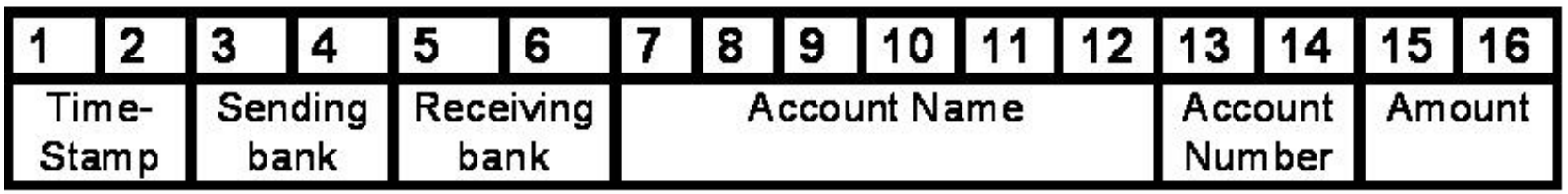
2. Electronic Codebook Mode

2.4 ECB Attacks

- Message replay
- Block replay

2. Electronic Codebook Mode

2.4 ECB Attacks



NB: $1, 2, 3, \cdots 16$ indicates encryption blocks

Block $\longrightarrow$ 8-byte encryption block

2. Electronic Codebook Mode

2.4.1 Message replay attack

Transmits two identical messages

Intercept these messages

Replay the messages

NB: Does not need to know contents of ciphertext or cipher key

2. Electronic Codebook Mode

2.4.1 Block replay attack

If timestamp added, no two messages will be identical

Identify blocks unique to Mallory's (attacker) account

Intercepts arbitrary messages, replaces account block with his

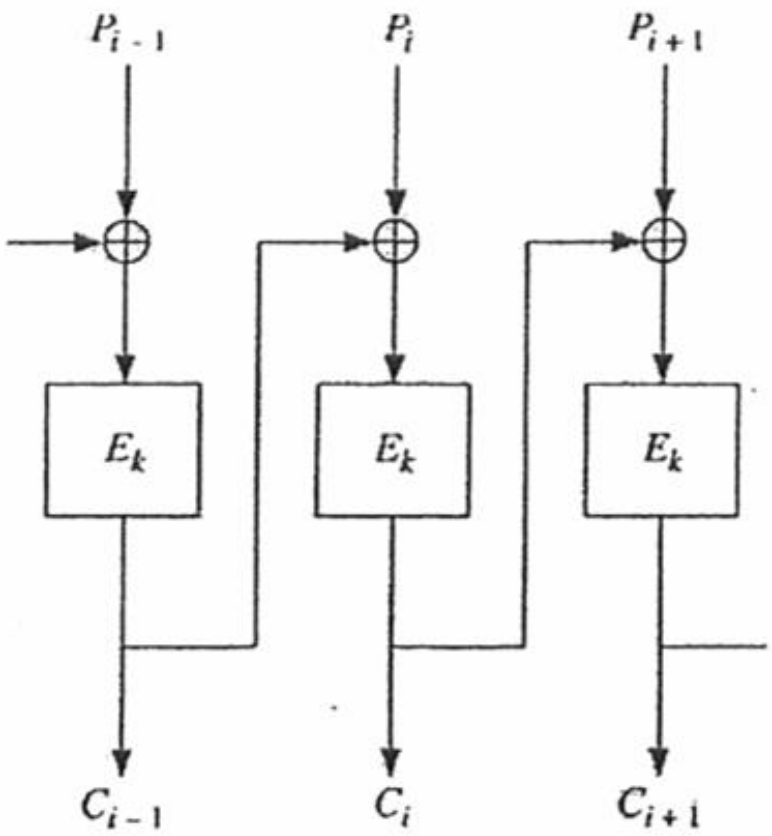
3. Cipher Block Chaining

Adds a feedback mechanism to block ciphering process

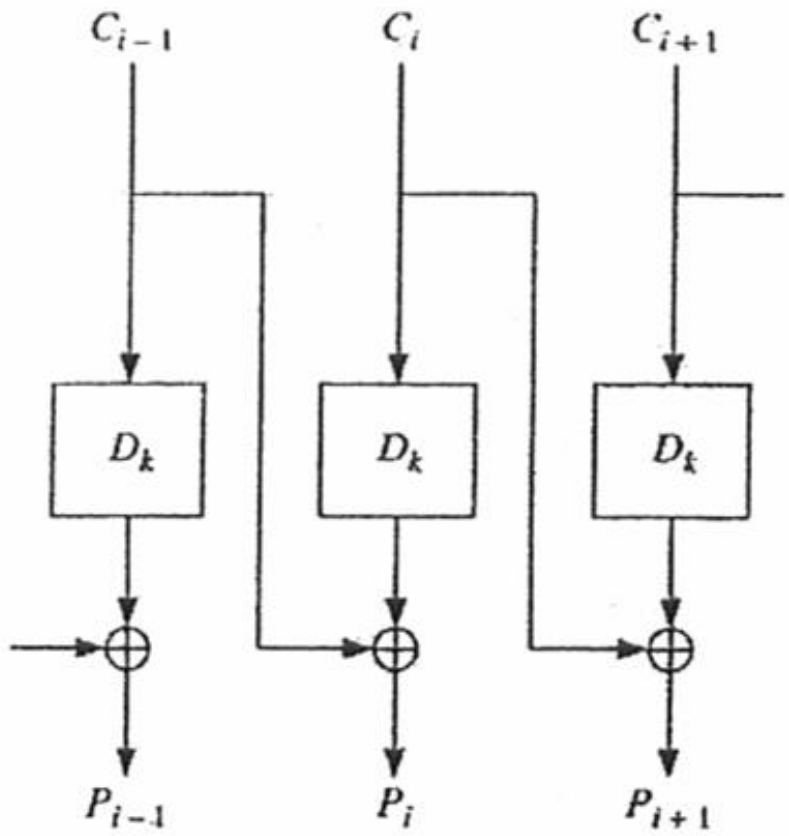
XOR current plaintext block with previous ciphertext block before encryption

Each ciphertext block depends on current plaintext block, as well as all previous plaintext blocks

3. Cipher Block Chaining



(a) CBC Encryption



(b) CBC Decryption

3. Cipher Block Chaining

3.1 CBC Initialisation

Problem:

- Messages will still encrypt the same up to the first difference (i.e. Letterheads will encrypt the same)
- Same messages will encrypt the same (message replay)

Solution:

3. Cipher Block Chaining

3.1 CBC Initialisation

Problem:

- Messages will still encrypt the same up to the first difference (i.e. Letterheads will encrypt the same)
- Same messages will encrypt the same (message replay)

Solution:

Use an Initialisation Vector (IV) of random bits.

3. Cipher Block Chaining

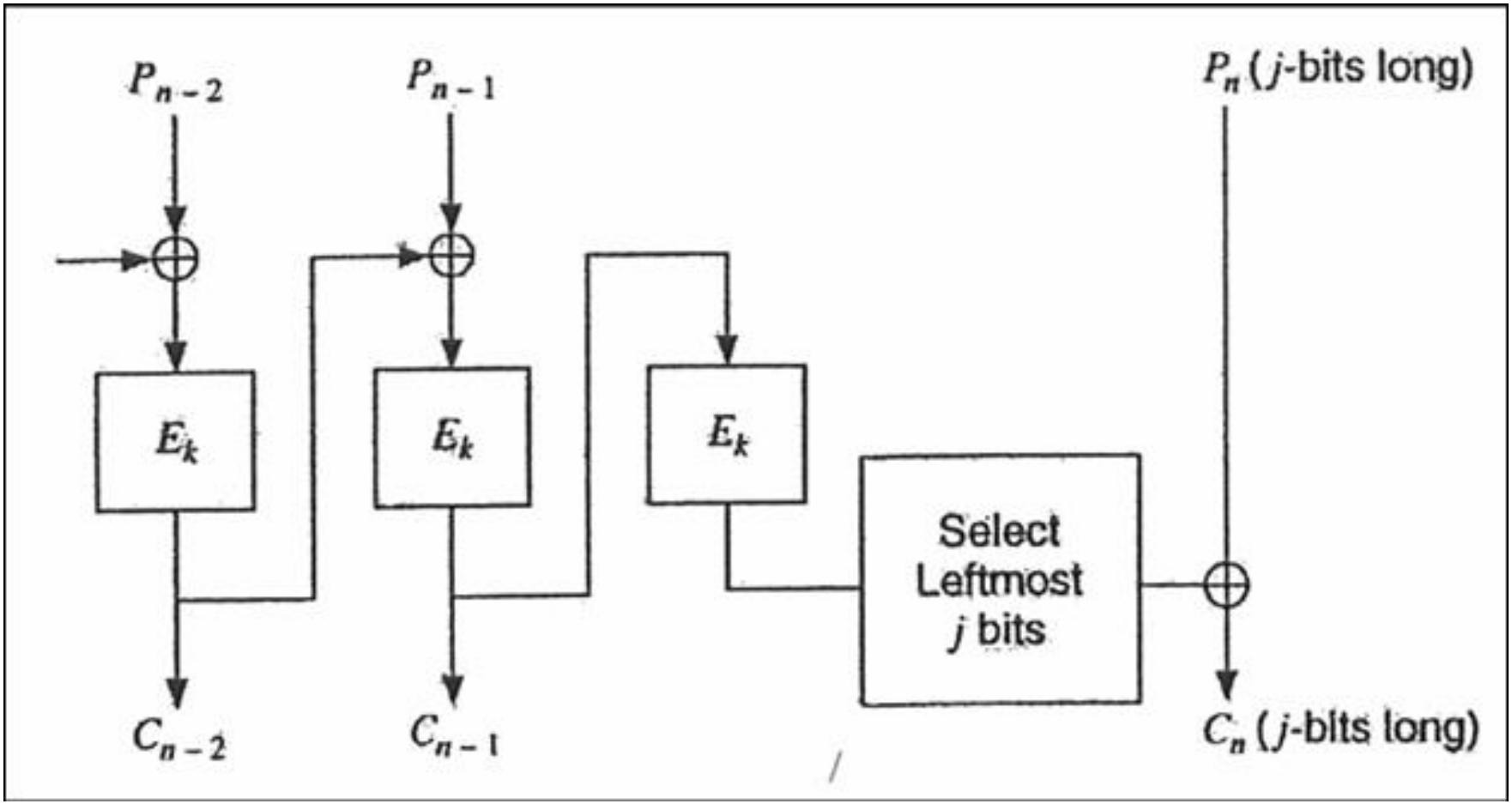
3.2 Padding

Padding is added exactly as for ECB mode (EOF and sequence)

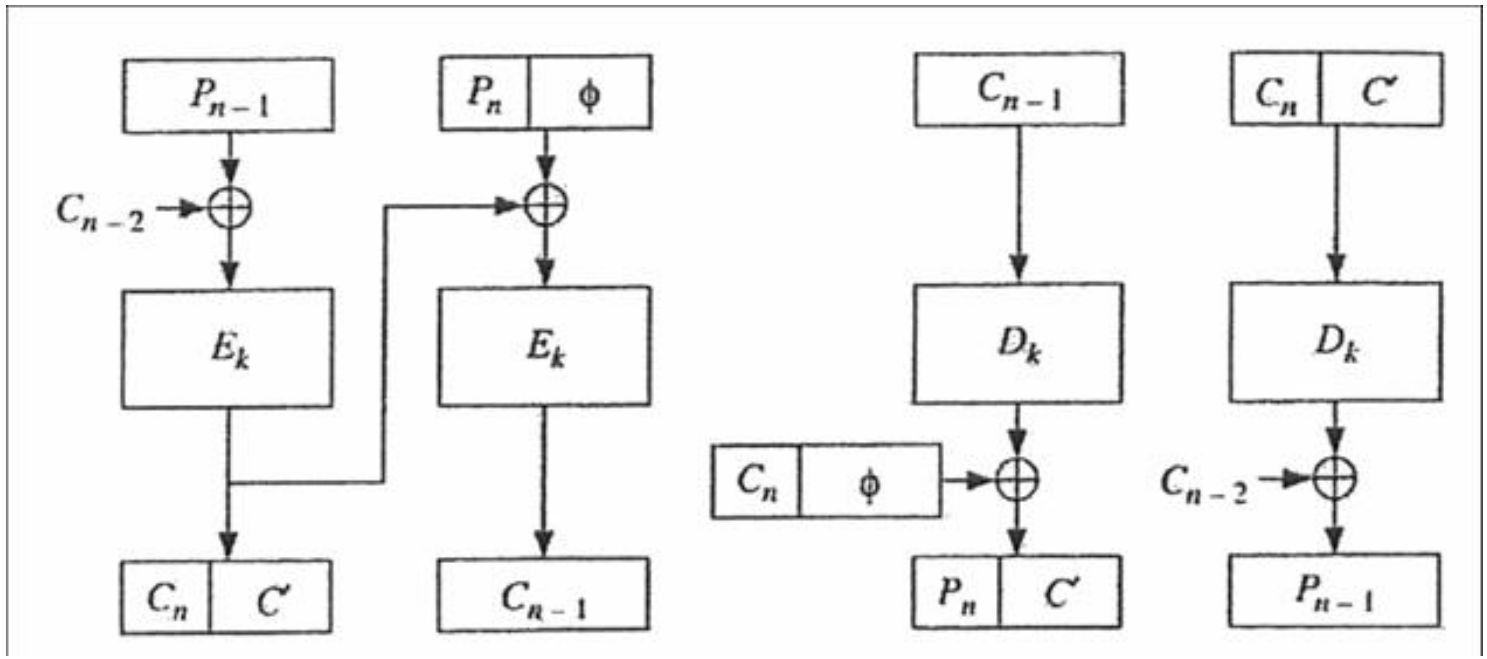
When **length**(ciphertext) = **length**(plaintext):

- Re-encoding of block C_{n-1}
- Ciphertext stealing

3. Cipher Block Chaining



3. Cipher Block Chaining



3. Cipher Block Chaining

3.3 Error Propagation

A single bit error in the plaintext block will affect that ciphertext block and all subsequent ciphertext blocks. This is not a problem, since decryption will reverse this effect and the plaintext will have the same single bit error.

Bit errors in ciphertext are more common. Due possibly to noisy communications line or malfunction in storage media.

- Single bit error in the ciphertext garbles entire block and one bit of the recovered plaintext.
- CBC is self-recovering $\rightarrow$ blocks after the 2nd are not affected.

Synchronization errors are not recoverable at all in CBC mode.

5. Cipher Feedback Mode

Implements a block cipher as a stream cipher

Useful for applications where each character must be encrypted immediately (e.g. from a terminal to a server)

5. Cipher Feedback Mode

5.1. Operation

Size of shift register = size of internal block cipher (8 bytes for DES)

Contents of shift register are encrypted

Left-most byte is XORed with the plaintext byte $\longrightarrow$ ciphertext byte

Copy of ciphertext byte is left-shifted into the shift register (from the right hand end)

Left-most byte is shifted out and discarded

Process is repeated

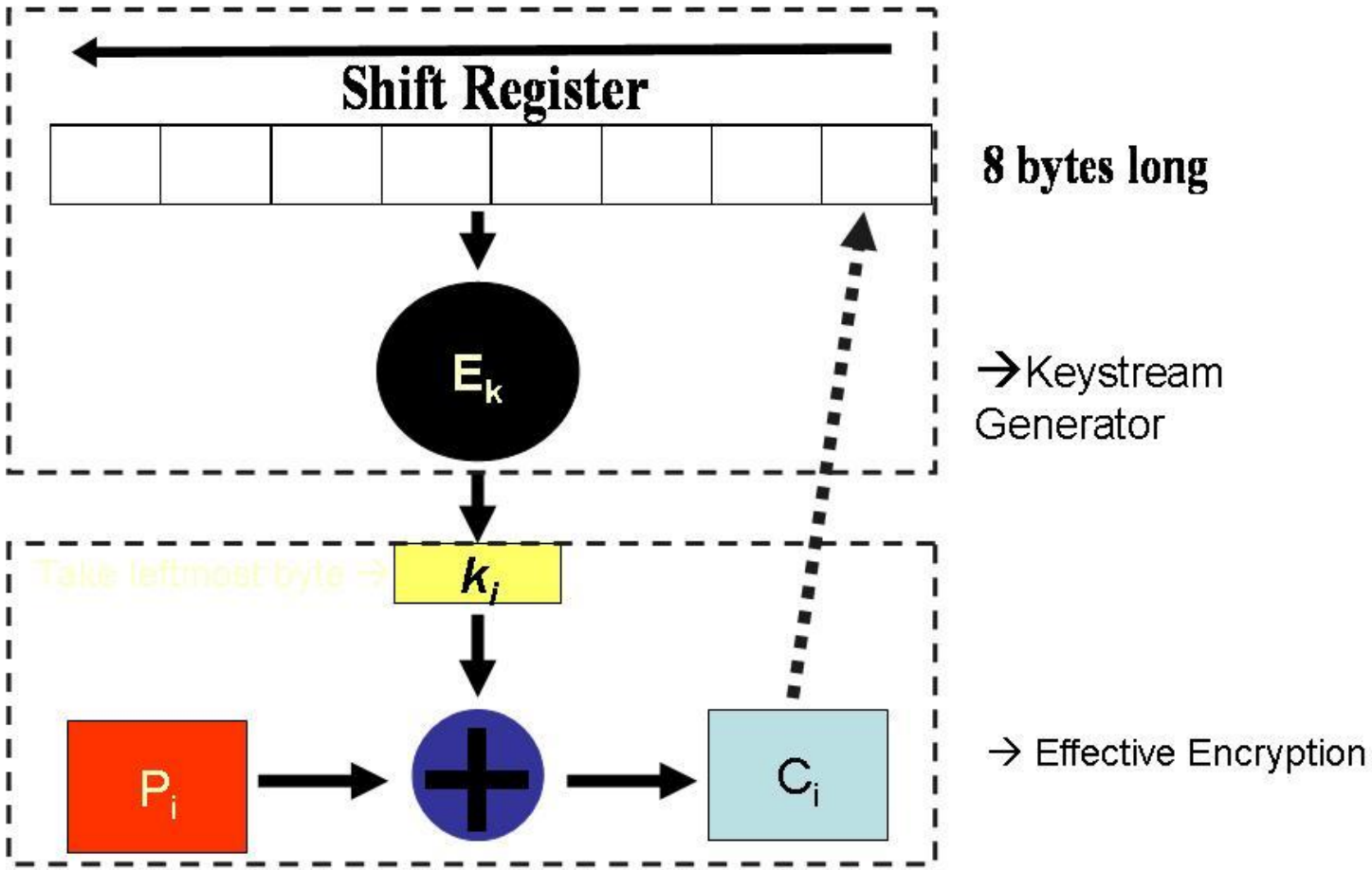
5. Cipher Feedback Mode

5.1. Operation

Decryption is the reverse of encryption

Important: Both enciphering and deciphering algorithms uses block cipher in encryption mode.

5. Cipher Feedback Mode



5. Cipher Feedback Mode

5.2. CFB Initialisation

Also need an Initialisation Vector for the shift register

IV must be unique for each message sent under the same key

5. Cipher Feedback Mode

5.2. Error Propagation

An error in the plaintext affects all subsequent ciphertexts, but is reversible.

Bit Errors in Ciphertext:

- Single bit errors in the ciphertext results in a single bit error in the plaintext. Thereafter the bit enters the shift register and garbles output until the erroneous bit gets shifted out of the register. Single bit results in 9 bytes of plaintext being garbled.
- After that the system recovers and all subsequent ciphertext is decrypted correctly.

5. Cipher Feedback Mode

5.2. Error Propagation

Synchronization Errors are recoverable.

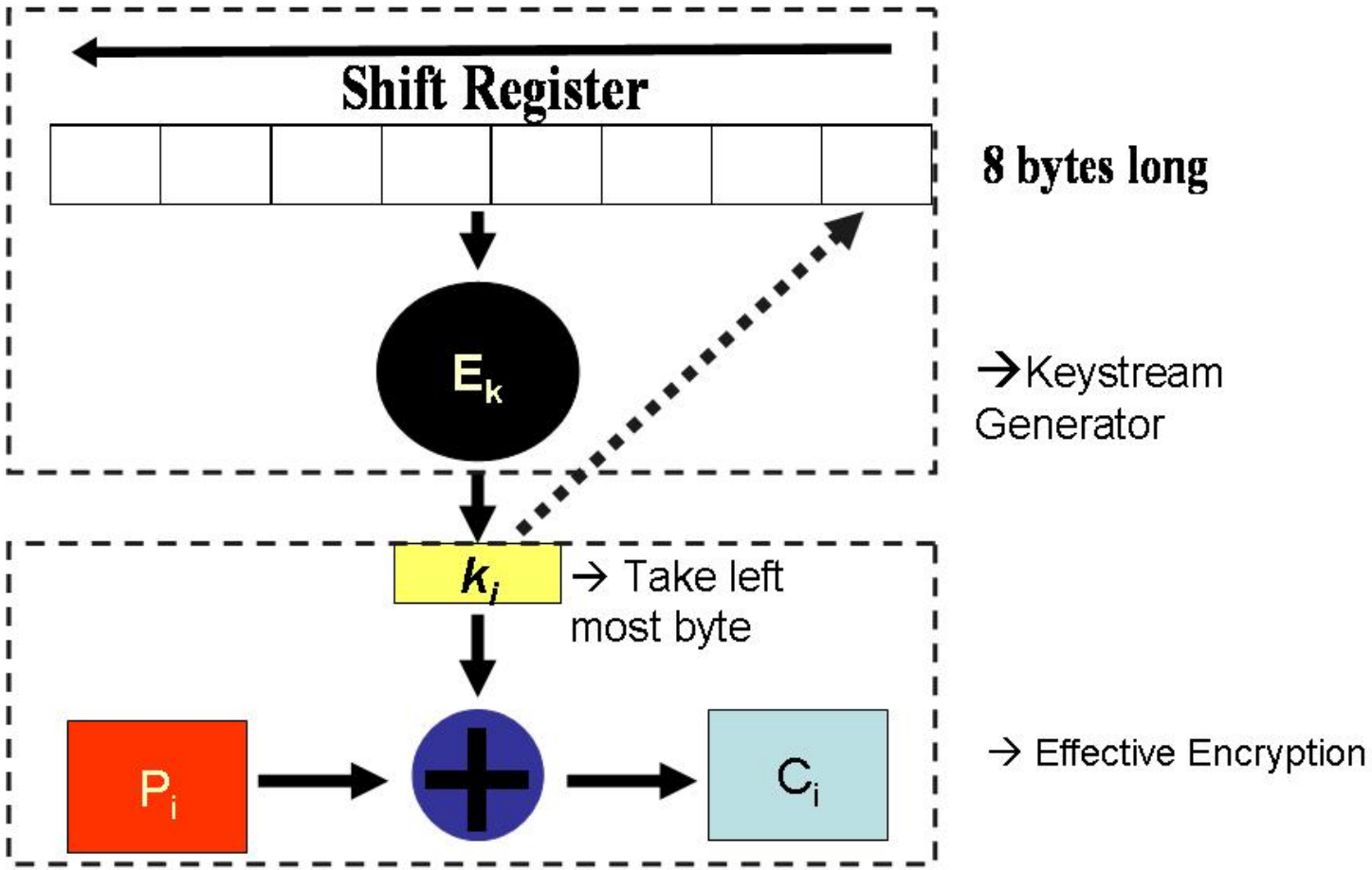
- The error enters the shift register and garbles 8 bytes of data until it falls off the register.
- Example of a block cipher being used as a self-synchronizing stream cipher.

6. Output Feedback Mode

Cipher stream generation similar to CFB mode

Key stream is not dependent on the data which is being encrypted

6. Output Feedback Mode



6. Output Feedback Mode

6.2. OFB Initialisation

Also need an Initialisation Vector for the shift register

IV must be unique

6. Output Feedback Mode

6.3. OFB Advantages

Entire keystream can be prepared before encryption starts $\longrightarrow$ Can make it very quick

Keystream is not dependent on the encrypted data $\longrightarrow$ Random access is possible (once keystream has been generated)

No error extension $\longrightarrow$ one bit ciphertext error causes one bit error in plaintext

6. Output Feedback Mode

6.3. OFB Issues

Loss of sync is unrecoverable (e.g. shift registers lose sync on encryption and decryption side)

Keystream will eventually repeat $\longrightarrow$ must change key before this