lecture 09

Modes of operation

A mode of operation describes how to repeatedly apply a cipher's single block operation (Enc operation / DES) to securely transform amounts of data larger than a block.

1. Electronic Code Book Mode (ECB)
2. Cipher Block Chaining Mode (CBC)
3. output Feedback Mode (OFB)
4. Cipher Feedback Mode (CFB)
5. Counter Mode (CTR)
6. Galois Counter Mode (GCM)

Applications

To realize stream ciphers

To contruct hash functions

To make Message Authentication Codes (MAC)

To buid key Establishment protocols (KEP)

To make pseudo-Random Number Generator (RNG)

Benefits

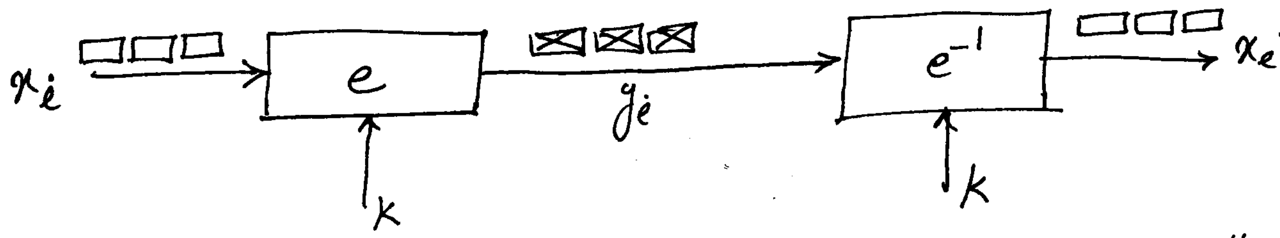
In addition to confidentiality, they provide authenticity & integrity

Is the message really coming from the original sender?

Was the ciphertext altered during transmission?

Requirement → Most modes require a unique binary sequence (Initialization vector) for each Enc operation → IV must be → non-repeating, → random } same plaintexts lead to ≠ ciphers

1. Electronic Code Book Mode (ECB)



Encryption : $y_i = e_k(x_i) \quad i \geq 1$

Decryption : $x_i = e_k^{-1}(y_i) \quad i \geq 1$

$x_1 \longrightarrow$ the 1st block
$x_2 \longrightarrow$ " 2nd "
$x_3 \longrightarrow$ " 3rd "
⋮

* Each block is encrypted seperately

Advantages $\longrightarrow$ ⊕ No block synchronization between sender & Receiver

⊕ Bit errors caused by noisy channels only affect the corresponding block but not succeeding blocks

⊕ Block cipher operating can be parallelized

Disadvantages $\longrightarrow$ ⊖ Identical plaintexts result in indentical ciphertexts

⊖ An attacker can recognize if the same message has been sent twice

⊖ plaintext blocks are encrypted <u>independenty</u> of previous blocks

Substitution Attack on ECB

- once a particular plaintext - ciphertext block mapping ($x_i \rightarrow y_i$) is known, a sequence of ciphertext blocks can be easily manipulated.

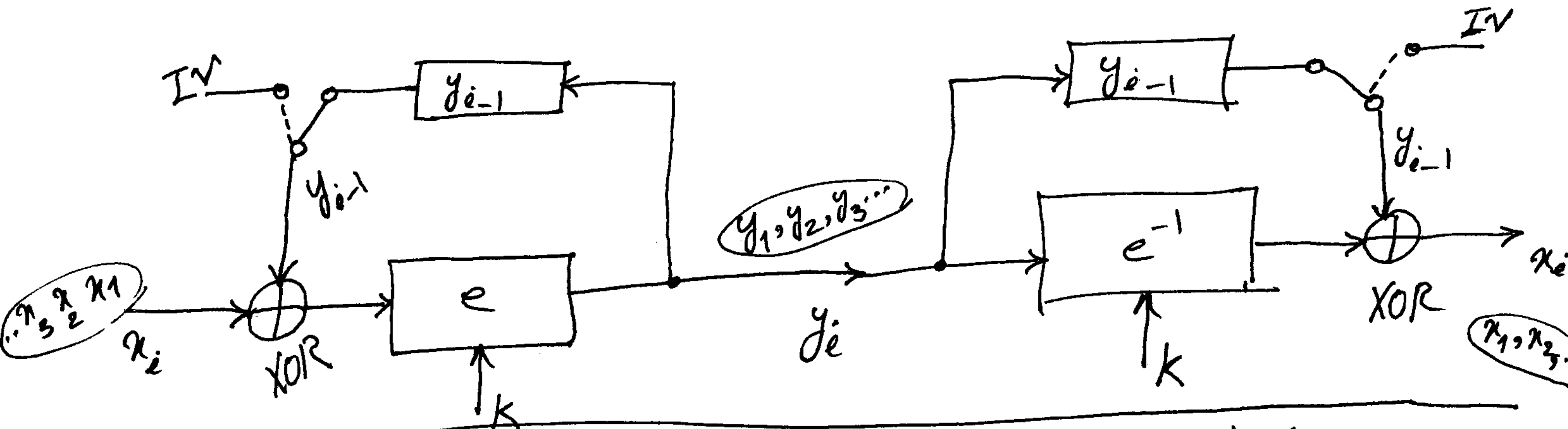
-

1	2	3	4	5
sending Bank A	sending acc #	Receiving Bank B	receiving acc #	Amount in USD $

assumption
- The enc key between two banks will not changed frequently
- The attacker sends $500 transfers from his acc at Bank A to his acc at Bank B repeatedly.
- He now simply replaces block 4 of other transfers with the block 4 that he stored before

2. Cipher Block chaining Mode (CBC)

- The enc of all blocks are "chained" together ciphertext y_i depends not only on block x_i but on all previous plaintext blocks
- The encryption is randomized by using an initialization vector



Encryption (first block): $y_1 = e_k(x_1 \oplus IV)$ $\quad i = 1$

Encryption (general block): $y_i = e_k(x_i \oplus y_{i-1})$ $\quad i \geq 2$

$y_1 = e_k(x_1 \oplus IV) \longrightarrow$ depends on x_1 & IV

$y_2 = e_k(x_2 \oplus y_1) \longrightarrow$ " " x_2, x_1 & IV

⋮

Decryption (first block): $x_1 = e_k^{-1}(y_1) \oplus IV$ $\quad i = 1$

Decryption (general block): $x_i = e_k^{-1}(y_i) \oplus y_{i-1}$ $\quad i \geq 2$

$x_1 = e_k^{-1}(y_1) \oplus IV$

$x_2 = e_k^{-1}(y_2) \oplus y_1$

⋮

it's not secret

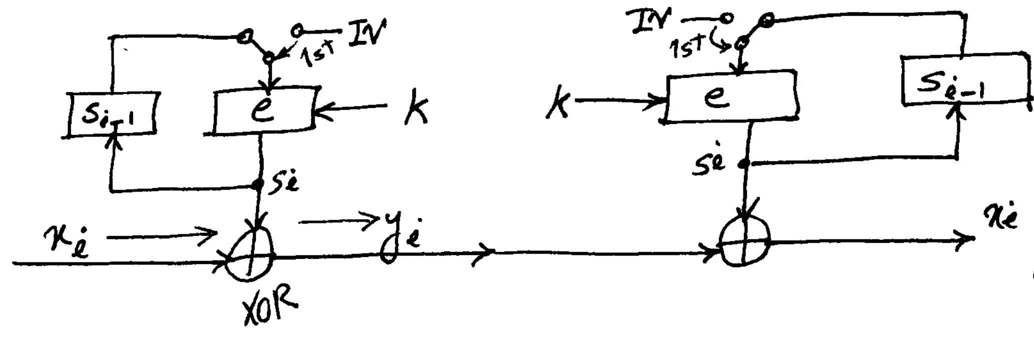
benefit → If we choose a new IV every time we encrypt, the CBC mode becomes a probabilistic Enc scheme, i.e., two ciphertexts of the same exact plaintexts look entirely different.

(3.) Output Feedback Mode (OFB)

It is used to build <u>synchronous stream cipher</u> from a block cipher

The key stream is not generated bitwise but instead in a blockwise fashion

similar to one-time pad



ciphertext plaintext

Encryption (first block): $s_1 = e_k(IV)$ and $y_1 = s_1 \oplus x_1$

Encryption (general block): $s_i = e_k(s_{i-1})$ " $y_i = s_i \oplus x_i$ $i \geq 2$

Decryption (first block): $s_1 = e_k(IV)$ and $x_1 = s_1 \oplus y_1$

plaintext

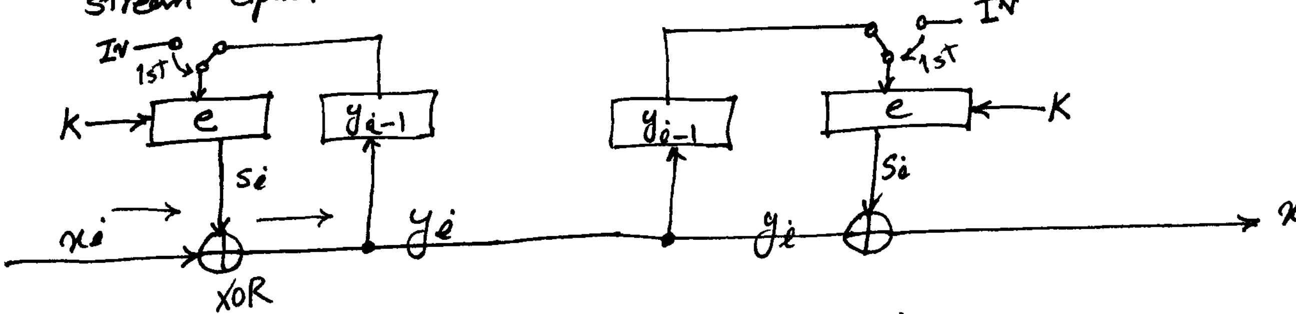
Decryption (general block) $s_i = e_k(s_{i-1})$ and $x_i = s_i \oplus y_i$ $i \geq 2$

key stream

(4.) cipher Feedback Mode (CFB)

It uses a block cipher as a buiding block for an asynchronous stream cipher

similar to one-time pad



Encryption (first block): $y_1 = e_k(IV) \oplus x_1$

Encryption (general block): $y_i = e_k(y_{i-1}) \oplus x_i$ $i \geq 2$

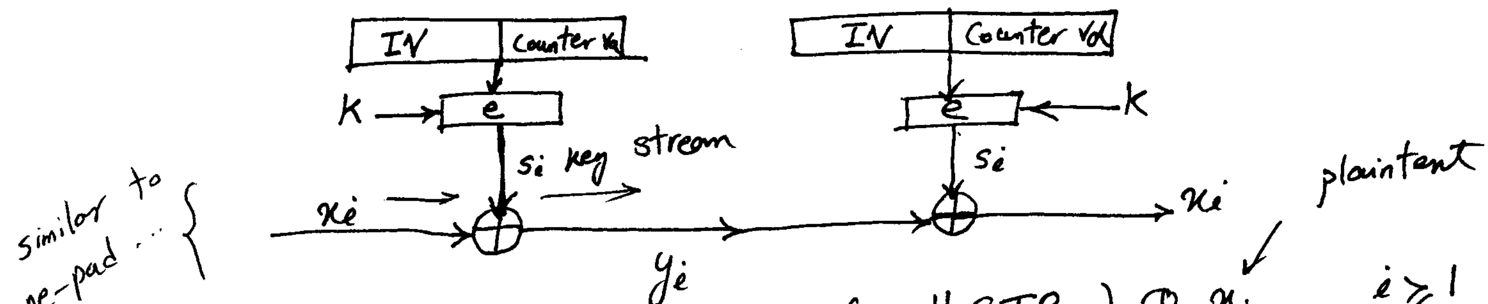
Decryption (first block): $x_1 = e_k(IV) \oplus y_1$

Decryption (general block): $x_i = e_k(y_{i-1}) \oplus y_i$ $i \geq 2$

5. Counter Mode (CTR)

\# very much similar to stream cipher

\# key stream is computed in a blockwise fashion



Encryption : $y_i = e_K(IV \,||\, CTR_i) \oplus x_i$ $\quad i \geq 1$

Decryption : $x_i = e_K(IV \,||\, CTR_i) \oplus y_i$ $\quad i \geq 1$

Counter value will be change for each key stream

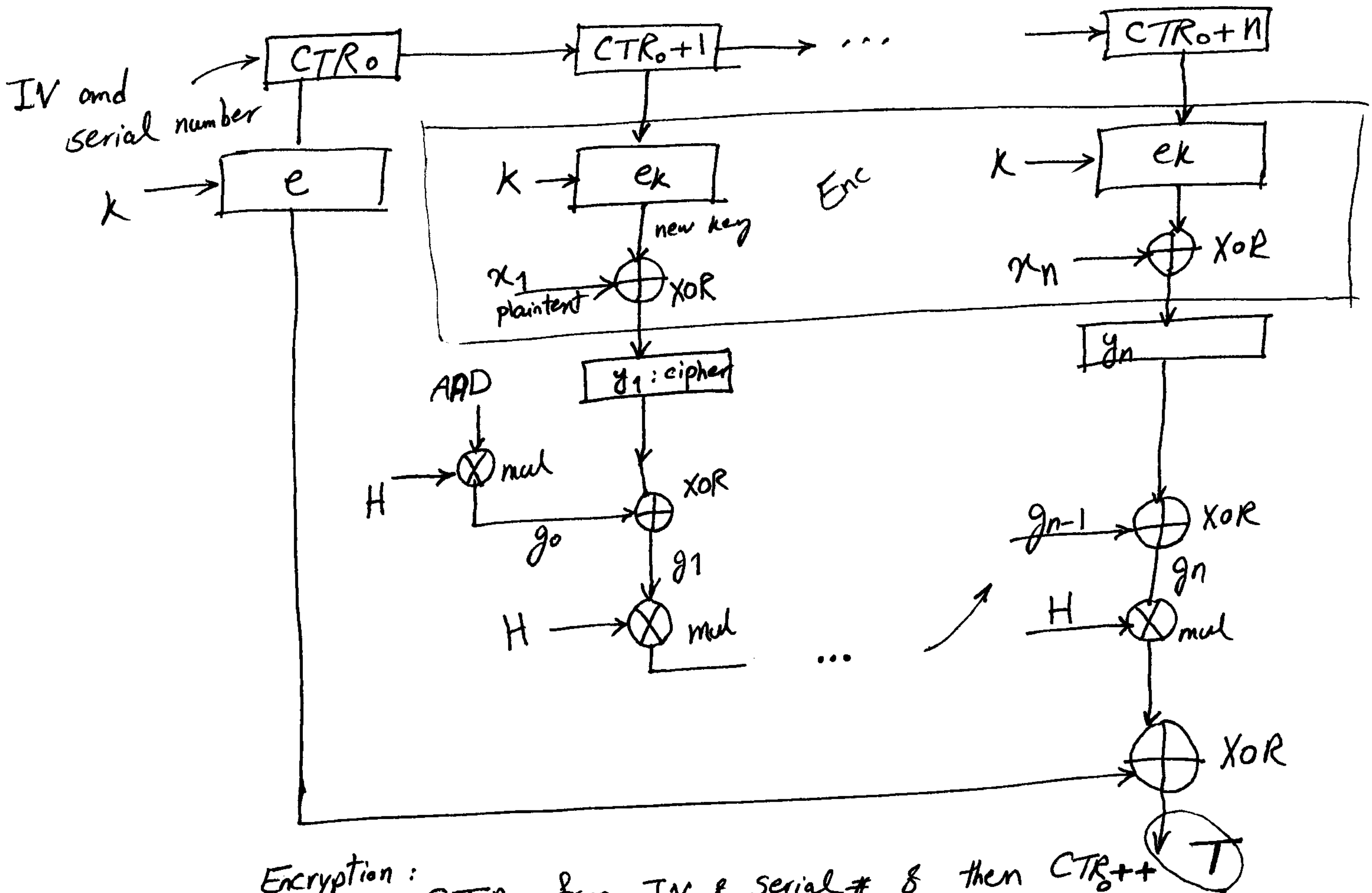
benefit # CTR mode can be parallelized since the 2nd encryption can begin before the 1st one has finished ...

\# Useful for high-speed implementation

6. Galois Counter Mode (GCM)

\# It also computes a message authentication code (MAC) i.e., a cryptographic checksum is computed for a message.

- message Authentication
- message Integrity



Encryption:

- \# CTR_0 from IV & serial # & then CTR_0++
- \# $y_i = e_K(CTR_i) \oplus x_i \quad i \geq 1$

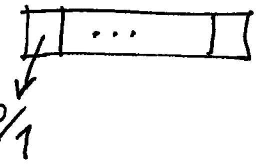
$\boxed{\text{Authentication:}}$

- $H = e_K(0)$
- ADD: Additional Authentication Data
- $g_0 = ADD \times H$ (Galois field multiplication)
- $g_i = (g_{i-1} \oplus y_i) \times H \quad 1 \leq i \leq n$ (Galois field mul)
- Tag: $T = (g_n \times H) \oplus e_K(CTR_0)$ (Galois field mul)

* Exhaustive key search

key → 56 bit → 2^{56}

56 bit



$2 \times 2 \ldots \times 2 = 2^{56}$

a pair of (cipher – plaintexts)

if you can map, then you can find the actual key from 2^{56} possible keys

plaintext space

2^{56} key

ciphertext space

↓

what the key had been

64 bits
↓
2^{64} possible message

key
80 bits
2^{80} keys

64 bits
↓
2^{64} possible ciphers

one pair of (p–c) is not enough, and you need a couple of (p–c) pairs to avoid false positive