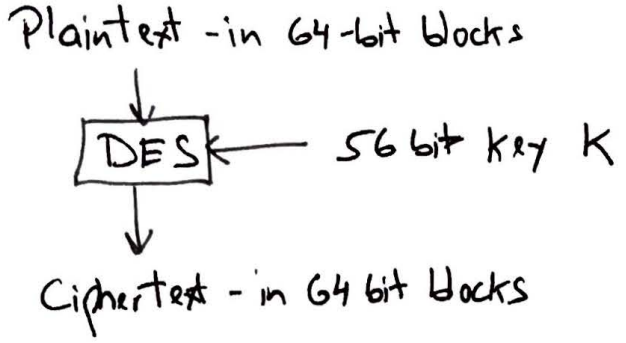
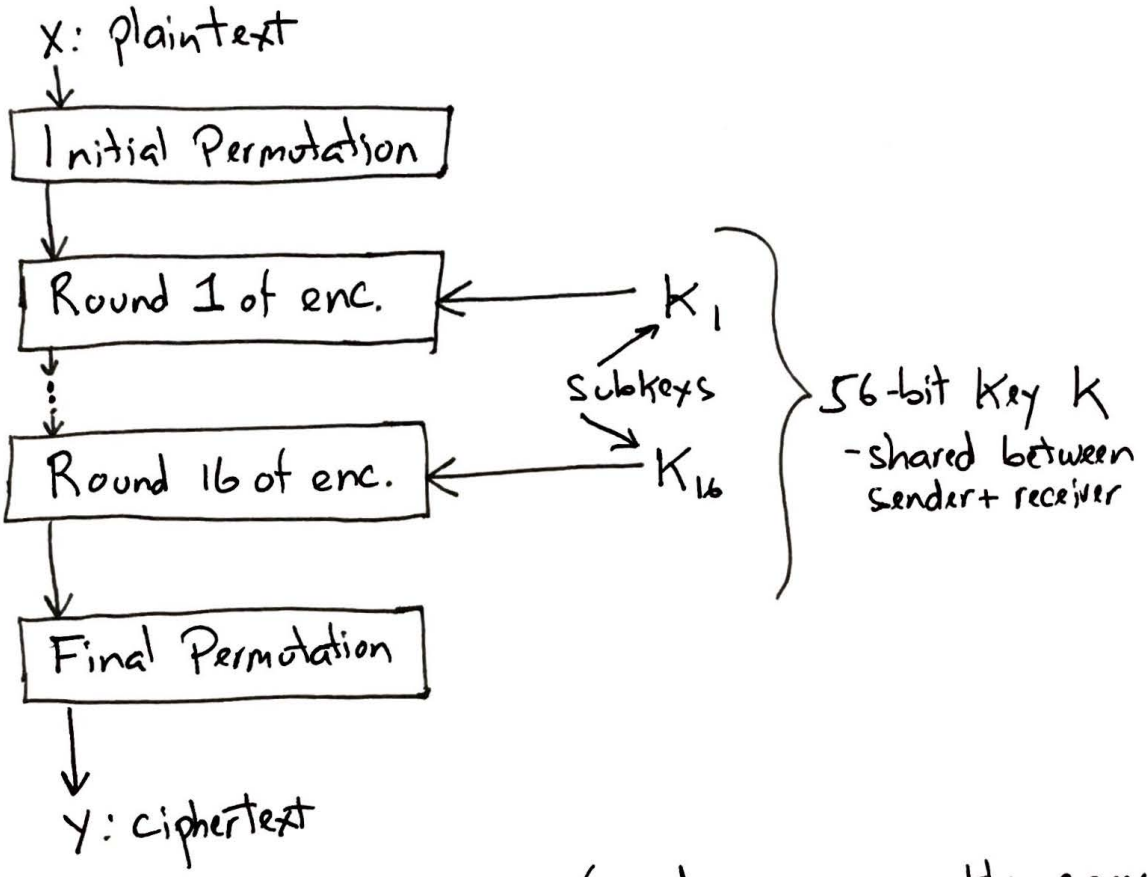
DES - High Level

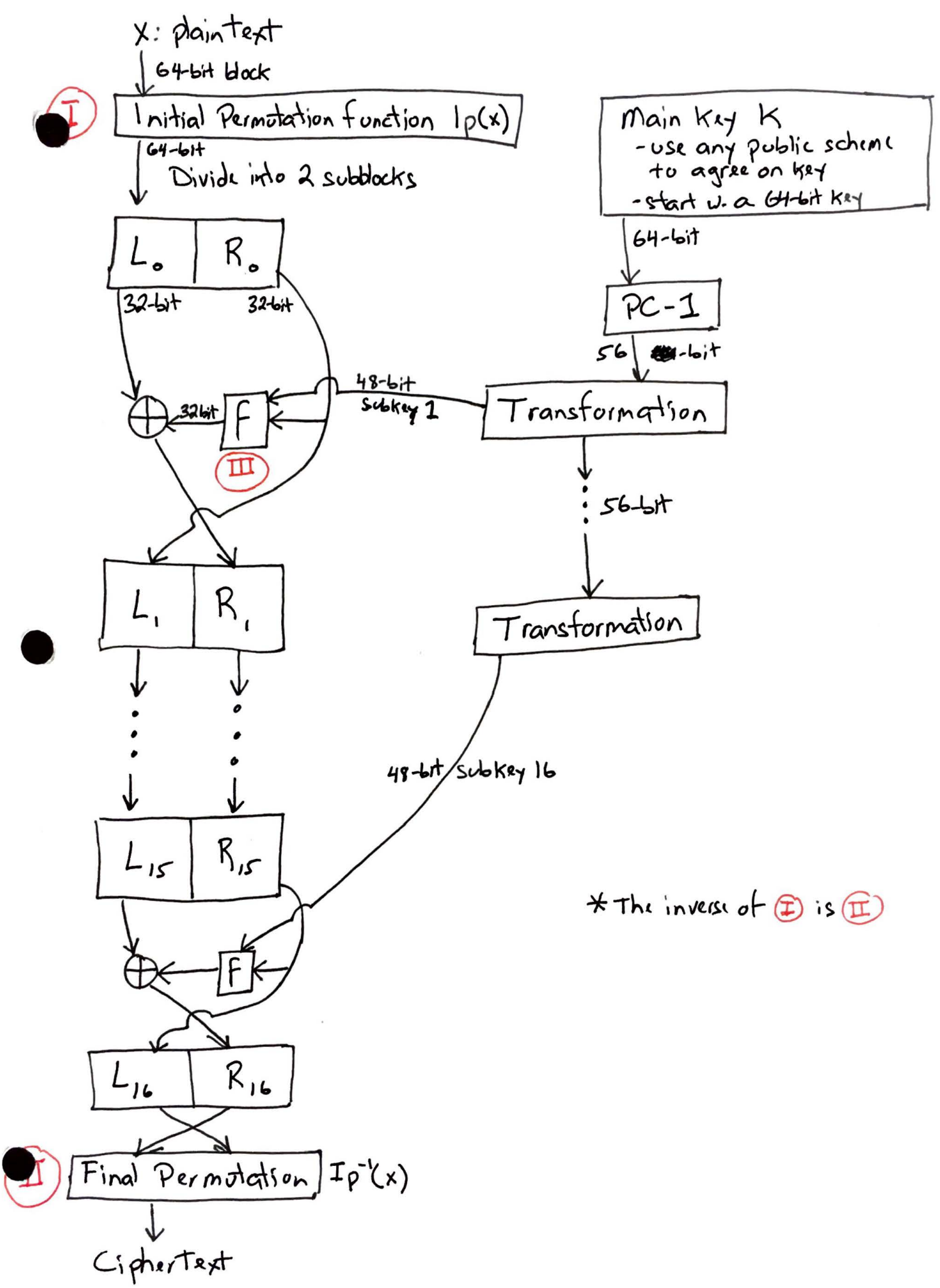


- Everything is public except the key
- Don't need to do input blocks in order



* for dec. we use the same key, but we generate subkeys in reverse order:

$K_{16}, K_{15}, K_{14}, \ldots K_1$


X: plaintext
64-bit block
I
Initial Permutation function Ip(x)
64-bit
Divide into 2 subblocks
Main key K
- use any public scheme to agree on key
- start w. a 64-bit key
64-bit
PC-1
56 -bit
L_0
R_0
32-bit
32-bit
32bit
F
III
48-bit Subkey 1
Transformation
56-bit
Transformation
L_1
R_1
48-bit Subkey 16
L_{15}
R_{15}
F
L_{16}
R_{16}
II
Final Permutation $Ip^{-1}(x)$
CipherText
* The inverse of I is II

Initial Permutation Function (I)

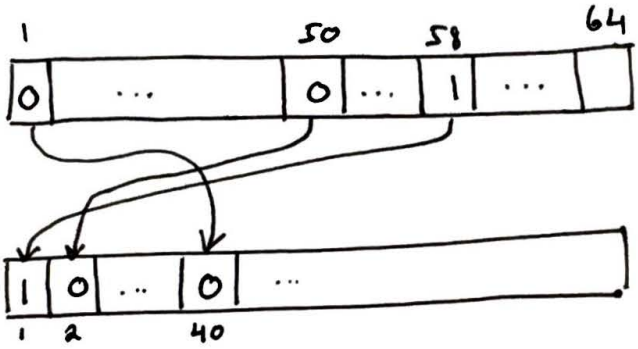
9/14 0:31:00

8x8
64 el.

	1	2	3	4	5	6	7	8
1	58	50	42	-	-	-	-	-
2	-							-
3	-							-
4	-							-
5	-							1
6	-							-
7	-							-
8	-							-

$I_p(x)$

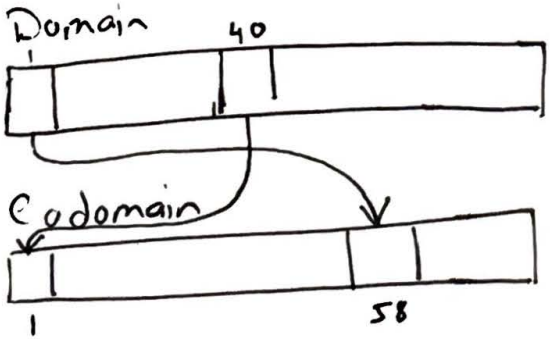
- Public table
- No repetative values



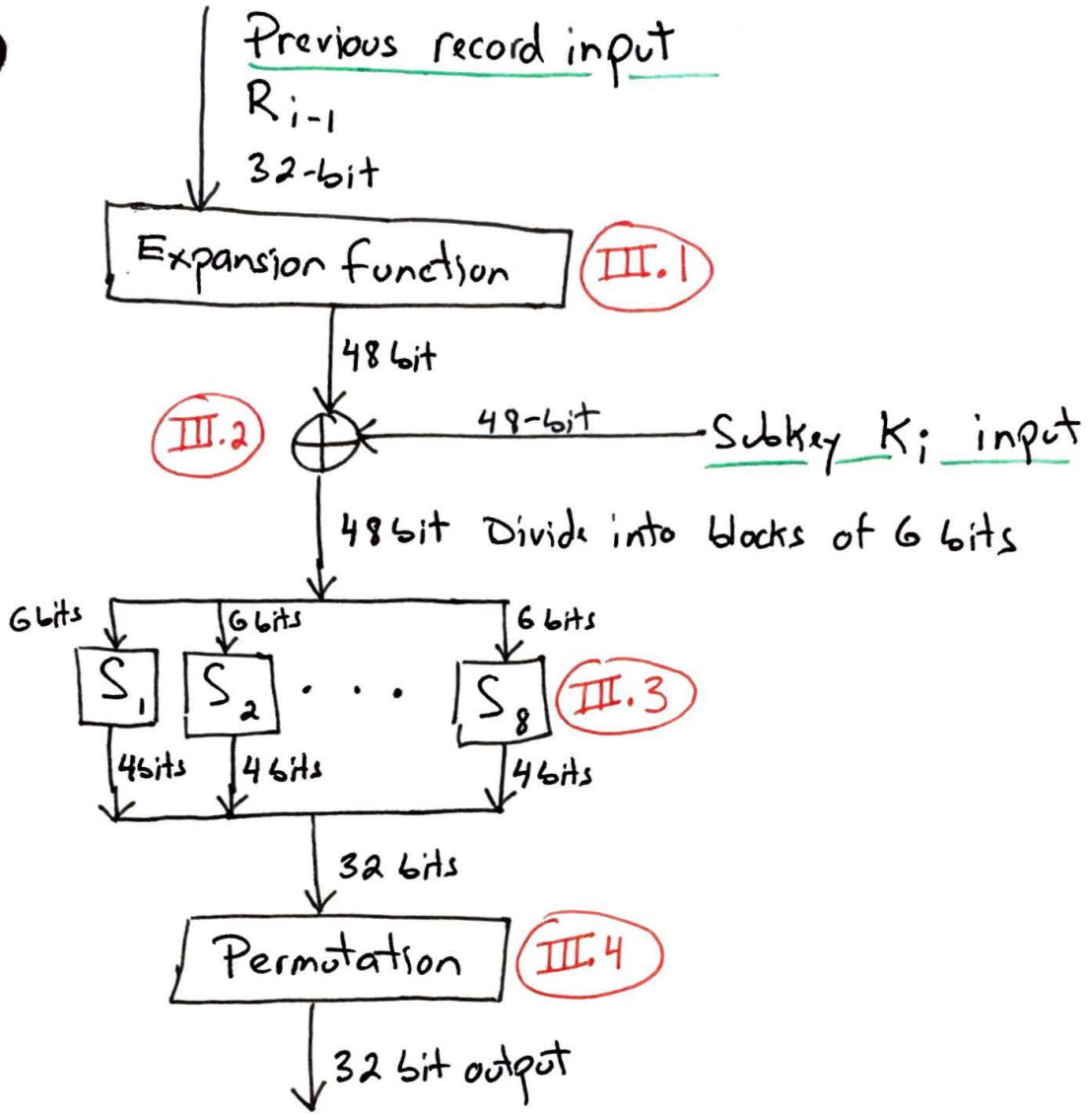
Final Permutation Function (II)

- use $I_p^{-1}(x)$ table for inverse

- Use $I_p^{-1}(x)$ table



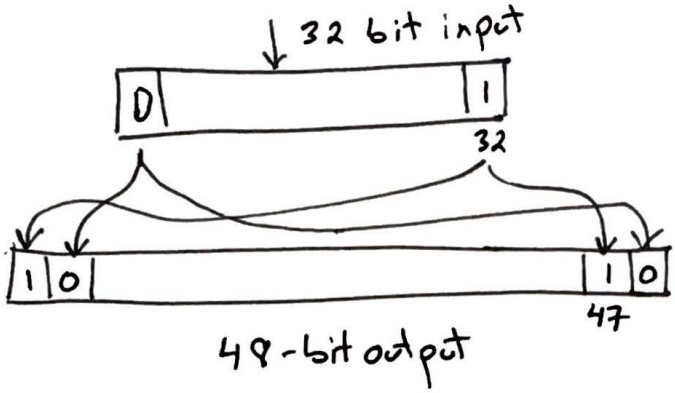
f function III



III.2

XOR	+mod 2
0 0	0
0 1	1
1 0	1
1 1	0

Expansion Function (III.1)



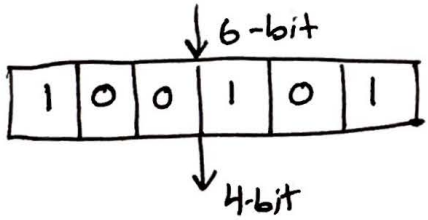
- works same as permutation function
- Expansion table
  - 8 x 6
  - Can't skip values
  - Can repeat some values

S boxes (III.3)

- Find the 8 tables online
- Table 1: ⟶
- Every row is randomely filled w. a no. from 0..15.

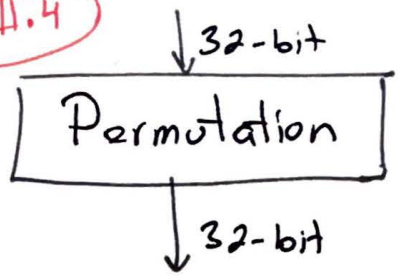
S_1	0	1	2	...	15
0	14	4	13		
1					
2					
3	15	12	8		

-

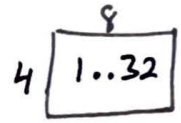


- Take 4 middle bits, 0010.
- Convert to decimal ⟶ 2
- Take 2 bits on end, 11.
- Convert to decimal ⟶ 3
- Use these against the table. [2,3] ⟶ 8
- Convert to binary ⟶ $(1000)_2$
- This is the output

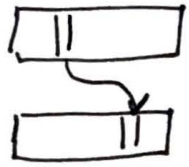
III.4



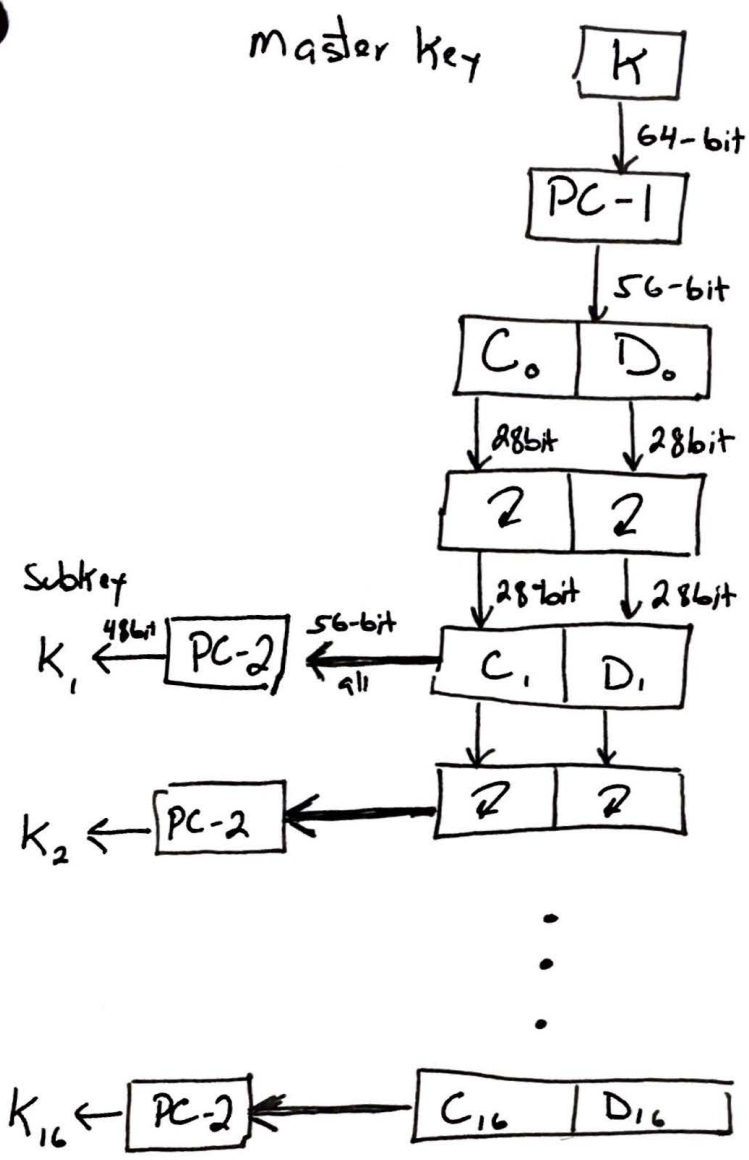
- Table is online, public
-



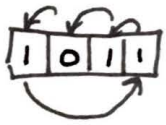
- Non repeating values
- Fill w. values 1..32
- Works similar to permutation



Key Schedule



Rotation to left

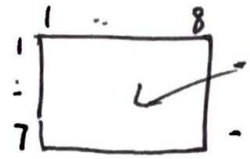


1. Rounds 1, 2, 9, 16
   - Rotate left by 1 bit
2. Other rounds,
   - rotate ~~right~~ left by 2 bits

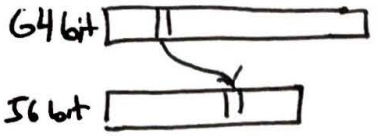
PC-1

7x8 table

8x7?



- 1..64, each no. once.
- exclude:
  8, 16, 24, 32, 48, 56, 64



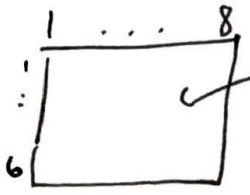
* Each byte loses the last bit

- that will convert the input from 64 → 56 bits

PC-2

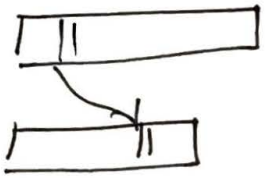
6x8

8x6?



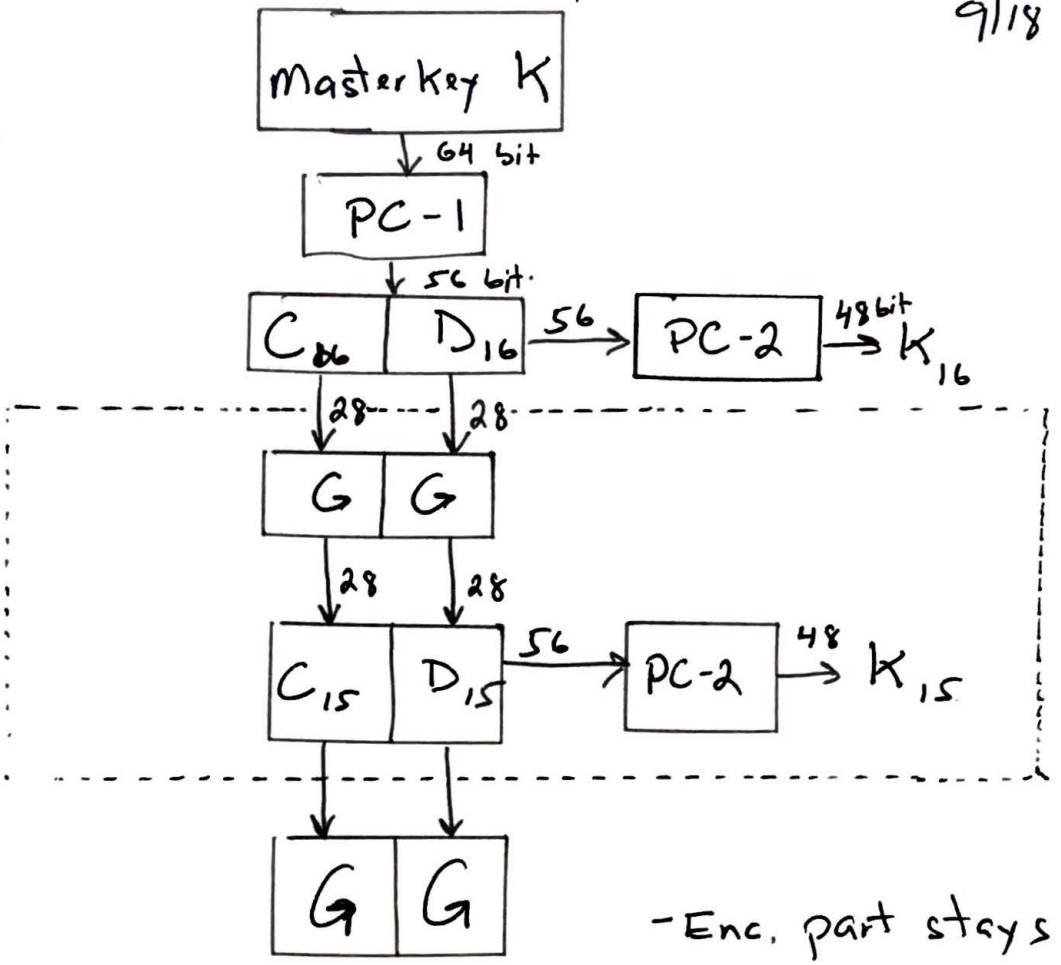
1..56

exclude 8 no.



DES Decryption

9/18 0:0:0



- Enc. part stays the same
- No rotation on 1st round
- 1 bit rotation to the right rounds 2, 9, 16.
- 2 bits rotation to the right in all other rounds

Modes of Operation 9/18

- A mode of operation describes how to repeatedly apply a cipher scheme's single block operation (enc. op. / DES / 3DES) to securely transform amounts of data larger than a single block.

1. ECB - Electronic Code Book mode
2. CBC - Cipher Block Chaining mode
3. OFB - Output Feedback mode
4. CFB - Cipher Feedback mode
5. CTR - Counter mode
6. GCM - Galois Counter mode

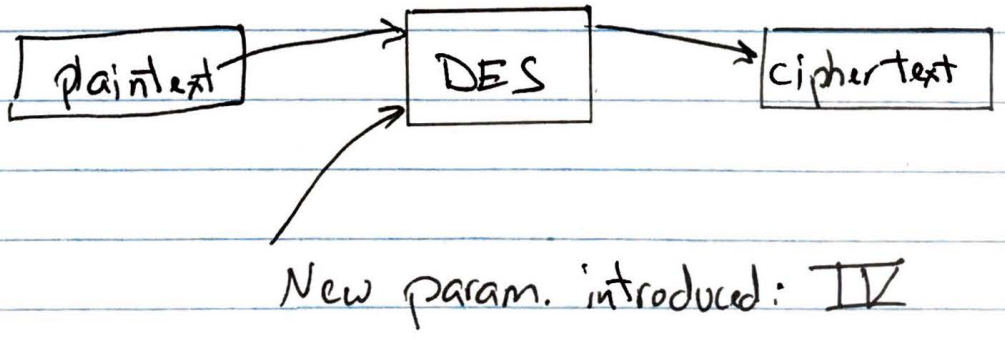
- Six different strategies used to treat your incoming blocks for enc.

- Applications:
  - To realize the stream cipher
  - To construct hash functions
  - To make msg. authentication codes (MAC)
  - To build key establishment protocols
  - To make pseudorandom no. generators (RNG)

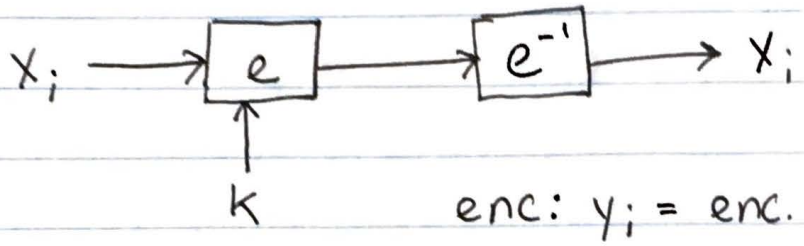
- Benefits
  - In addition to confidentiality, they provide authenticity and integrity.
    - is the msg. coming from the original sender?
    - Was the cipher text altered during transmission?

- Requirements
  - Most modes of operation require a unique binary sequence called Initialization Vector for each enc. op.
    - IV must be non-repeating and random.
    - Imagine that you want to send the same plaintext more than once. Injecting a new param. changes the ciphertext.

e.g.



① ECB - Electronic Code Book mode



enc: y_i = enc. function using master key over each block $i \geq 1$

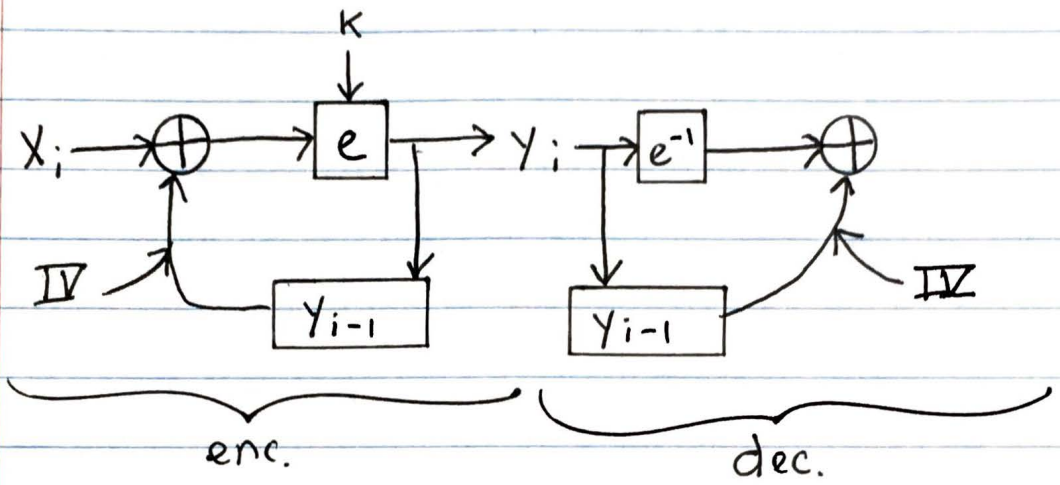
$$y_i = e_K(x_i)$$

$$\text{dec: } x_i = e_K^{-1}(y_i) \quad i \geq 1$$

- Advantages
  - ⊕ Does not use an IV
  - ⊕ No block synchronization between the sender and receiver. Don't need to coordinate anything with the receiver.
  - ⊕ Bit errors caused by noisy channels only affect the corresponding block, but not succeeding blocks. Two ways info can be changed:
    1. Attaker in the middle changes something.
    2. Noise in environment changes something.
    - error damage does not go to other blocks.
  - ⊕ Block cipher op. can be parallelized.
- Disadvantages
  - ⊖ plaintext blocks are enc. independently of prev. blocks.
  - ⊖ identical plaintext result in identical cipher text.

② CBC - Cipher Block Chaining mode

- The enc. of all blocks are "chained" together and as a result, cipher text y depends not only on block x, but on <u>all</u> prev. plain text blocks.
- This enc. is randomized by using an IV
- IV is shared between sender / receiver



- IV is used in the first round only, after that, y_{i-1} (prev y) will have a value.

Benefit:

If we choose a new or random IV everytime we enc. the CBC mode becomes probabolistic enc. scheme i.e. that is, two ciphertexts of the same exact plaintext look entirely different.

2 cont.

Enc. Formalized:

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

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

e.g. $y_1 = e_k(x_1 \oplus IV)$ → depends on x_1, IV
$y_2 = e_k(x_2 \oplus y_1)$ → depends on x_2, x_1, IV
$y_3 = e_k(x_3 \oplus y_2)$ → depends on x_3, x_2, x_1, IV

Dec. Formalized:

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

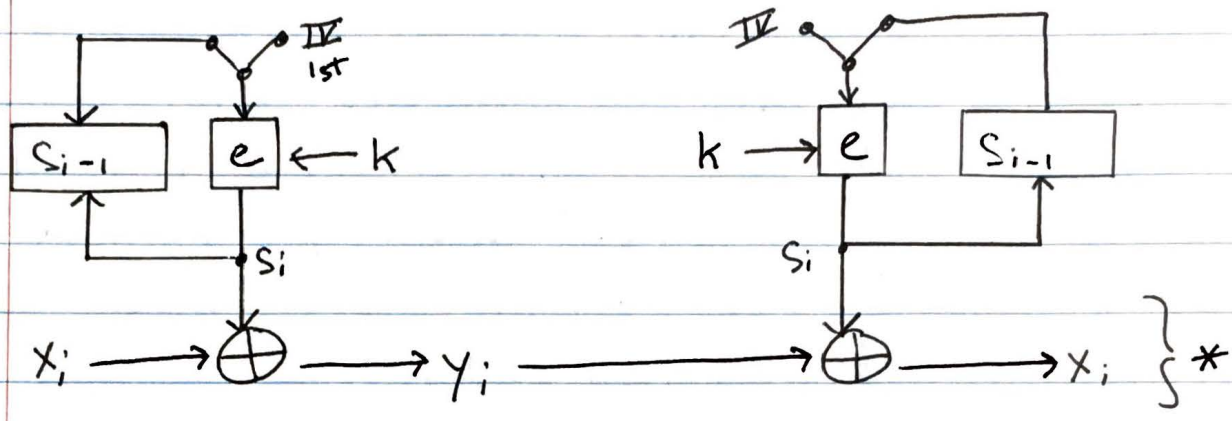
then: $x_i = e_k^{-1}(y_i) \oplus y_{i-1}$ $i \geq 2$

e.g. $x_1 = e_k^{-1}(y_1) \oplus IV$
$x_2 = e_k^{-1}(y_2) \oplus y_1$

9/25
8:54

(3) Output Feedback Mode OFB

- used to build synchronous stream cipher from the block cipher.
- The key stream is not generated bitwise, but instead in a blockwise fasion.



* Similar to one-time-pad, but in order to have a sequence of bits from the master key, we add this block.
- Use IV for first round. IV is public. IV is enc. 1st round.
- Round 1: K and IV is enc., result is $\oplus$ w. y_1
- Round 2: Prev. S_i is enc. w. k, then $\oplus$ w. y_2

enc. (first block): $S_1 = e_k(IV)$ and $y_1 = S_1 \oplus x_1$

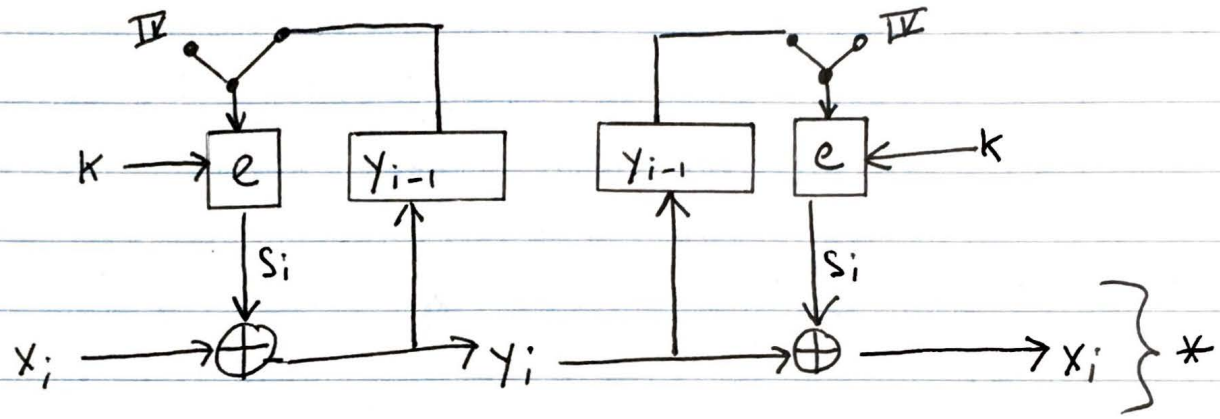
enc. (general block): $S_i = e_k(S_{i-1})$ and $y_i = S_i \oplus x_i$ $i \geq 2$

dec. (first block): $S_1 = e_k(IV)$ and $x_1 = S_1 \oplus y_1$

dec. (general block): $S_i = e_k(S_{i-1})$ and $x_i = S_i \oplus y_i$ $i \geq 2$

(4) Cipher Feedback Mode CFB

- gets feedback from cipher
- uses a block cipher as a building block for an asynchronous stream cipher.



*similar to one-time-pass

enc.(first block): $y_1 = e_K(IV) \oplus x_1$

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

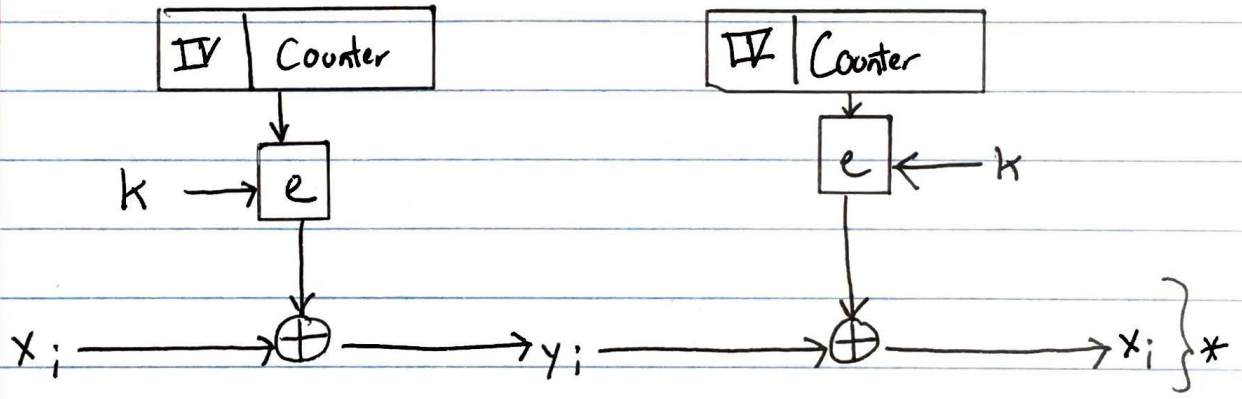
dec.(first block): $x_1 = e_K(IV) \oplus y_1$

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

(5) Counter Mode CTR

- very similar to stream cipher
- the key stream is computed in a block wise fasion
- Counter value will be changed for each key stream. Whatever you do on the sender side, you have to do on the receiver side. So if you increment one-by-one on enc. side, then also inc one-by-one on dec. side.
- Useful for high speed implementation

⊕ benefit: CTR mode can be parallelized since the 2nd enc. can begin before the first one.



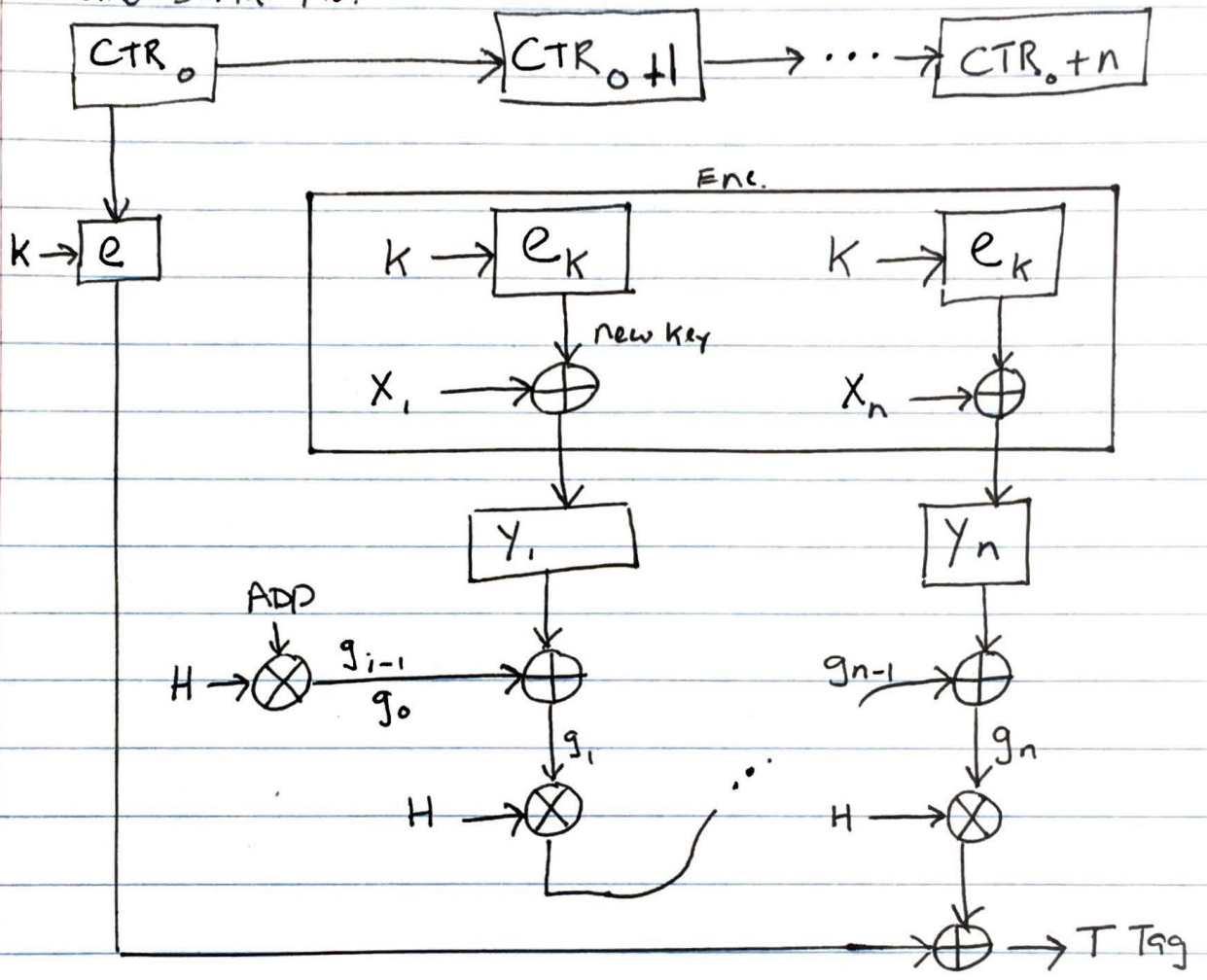
* similar to one-time-pad

enc. $y_i = e_k(IV \,||\, CTR_i) \oplus x_i \qquad i \geq 1$

dec. $x_i = e_k(IV \,||\, CTR_i) \oplus y_i \qquad i \geq 1$

⑥ Galois Counter Mode 9/25 0:37

IV and serial no.



⊗: Like modular multiplication. Do mult., then modular reduction.

enc. CTR_0 will be generated from IV and serial no. and CTR_0 will be incremented

$Y_i = e_K(CTR_i) \oplus X_i$ $i \geq 1$

H: $e_K(0)$

ADD: Additional authentication data.

6. cont.

$g_0 = \text{ADD} \times H \rightarrow$ Galois field mult

- Tag is sent w. ciphertext for verification.

$$g_i = (g_{i-1} \oplus y_i) \times H$$

$$\text{Tag: } T = (g_n \times H) \oplus e_k(CTR_0)$$