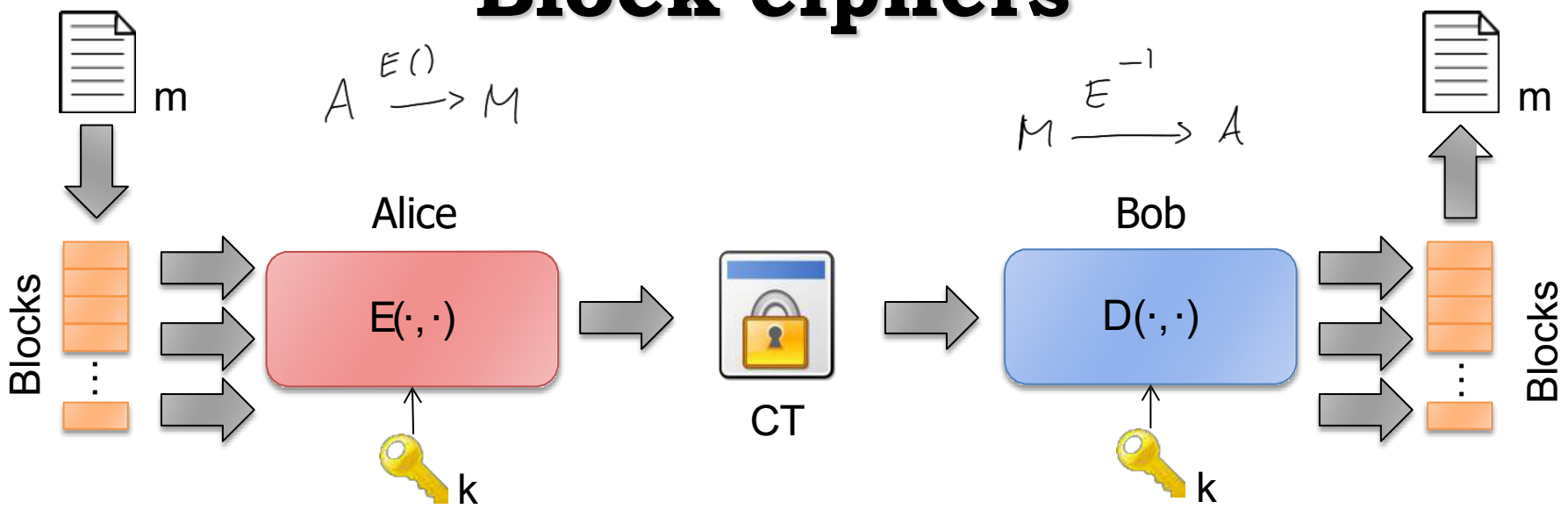
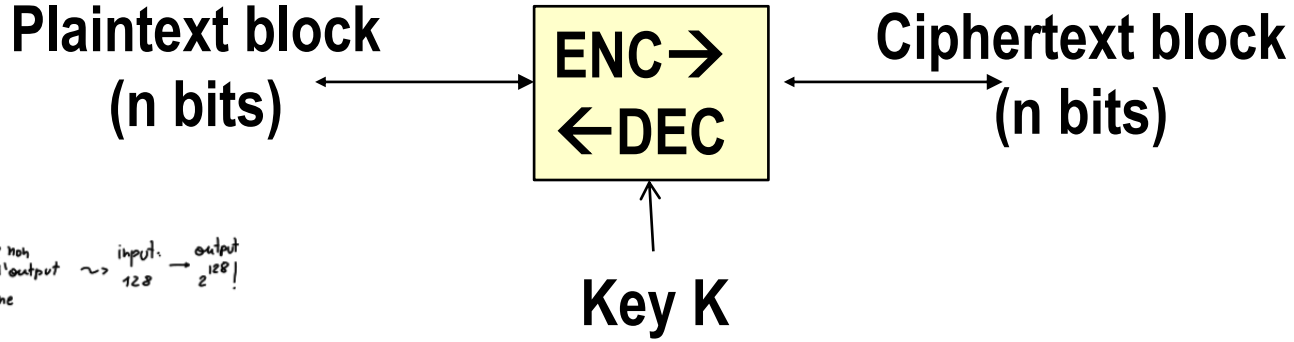
Review of Block cipher Modes of Operation

Block ciphers



➔**Goal: "generalize" substitution ciphers**



➔ **Block algorithm should implement a Pseudo Random Permutation**

⇨ in practice key will permit to select "only" among 2^(keysize) permutations

Graphics courtesy of Marco Spaziani Brunella

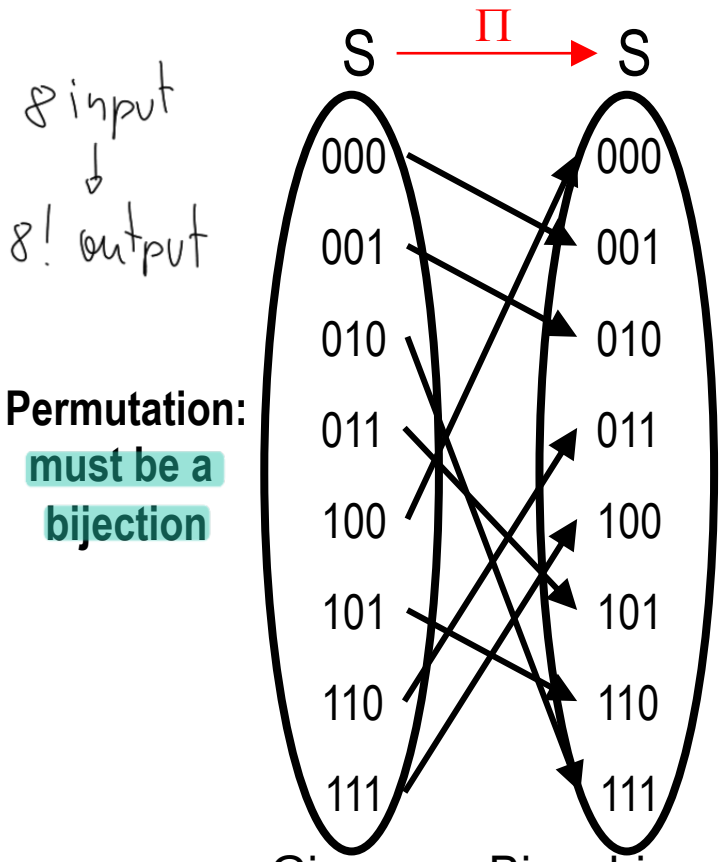
Block ciphers → PRPs

Pseudo Random Permutation:

→ S = set of all possible plaintexts

→n = 3 bits, S = {000,001,010,011,100,101,110,111}, |S| = $2^n = 8$

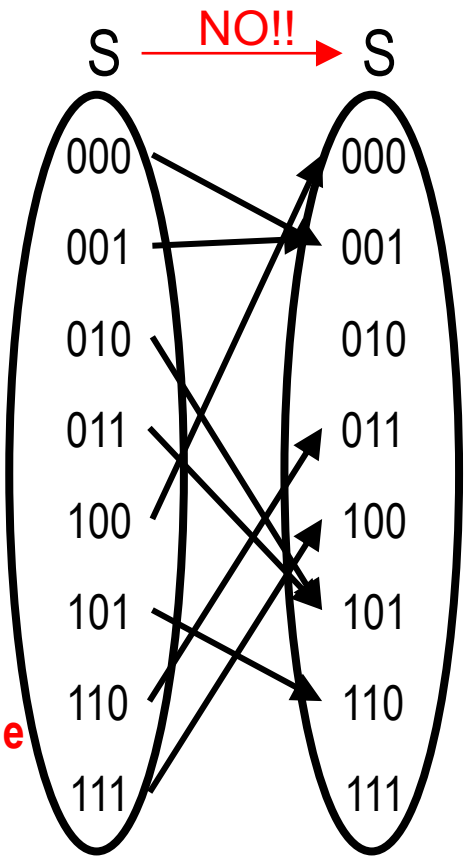
→ Permutation: bijective function Π: S→S (1-to-1 mapping)



Plaintext:
111

ENC→
←DEC

Ciphertext:
100



This is
WRONG!
Non-reversible
Mapping!

Block ciphers → PRPs

Pseudo Random Permutation:

→ S = set of all possible plaintexts

→ n = 3 bits, S = {000,001,010,011,100,101,110,111}, |S| = $2^n = 8$

→ Permutation: bijective function Π: S→S (1-to-1 mapping)

→ Pseudo-Random: block cipher should uniformly select one of the possible permutations

⇨ Permutation Π_K used by the cipher is «selected» by the secret key K

→ PROBLEM!! How many permutations? $2^n!$ (! = factorial)

⇨ For n=3: $2^3! = 8! = 40320$ → $\log_2(40320) = 15$ bit key

⇨ For n=8: $2^8! = 256! = 8.58 \times 10^{506}$ =

857817775342842654119082271681232625157781520279485619859655650377269452553147589377440291360451408450375885342
336584306157196834693696475322289288497426025679637332563368786442675207626794560187968867971521143307702077526
646451464709187326100832876325702818980773671781454170250523018608495319068138257481070252817559459476987034665
712738139286205234756808218860701203611083152093501947437109101726968262861606263662435022840944191408424615936
000000000000000000000000000000000000000000000000000000000000000000000000

⇨ In AES, n=128 …. $2^{128}!$ ~ 2^(2^{135}) → unbelievably large number!! (10^{40} digits key)

→ AES keys: 128, 192, 256 → $2^{256} << 2^{128}!$ (2^{256} just 78 digits key)

⇨ Take home: Actual AES permutations way less that ideal PRP (but still OK)

Block ciphers → PRPs

Pseudo Random Permutation:

→ S = set of all possible plaintexts

→n = 3 bits, S = {000,001,010,011,100,101,110,111}, $|S| = 2^n = 8$

How to compute large factorials?

Stirling Approximation

$$n! \approx \sqrt{2\pi n}\frac{n^n}{e^n}$$

$$2^{128}! \approx \sqrt{2\pi 2^{128}} \cdot 2^{128 \cdot 2^{128}} / e^{2^{128}} \approx$$

$$\approx \sqrt{2\pi} \cdot 2^{64+2^{135}-2^{128} \cdot \log_2 e} \approx 2^{2^{135}}$$

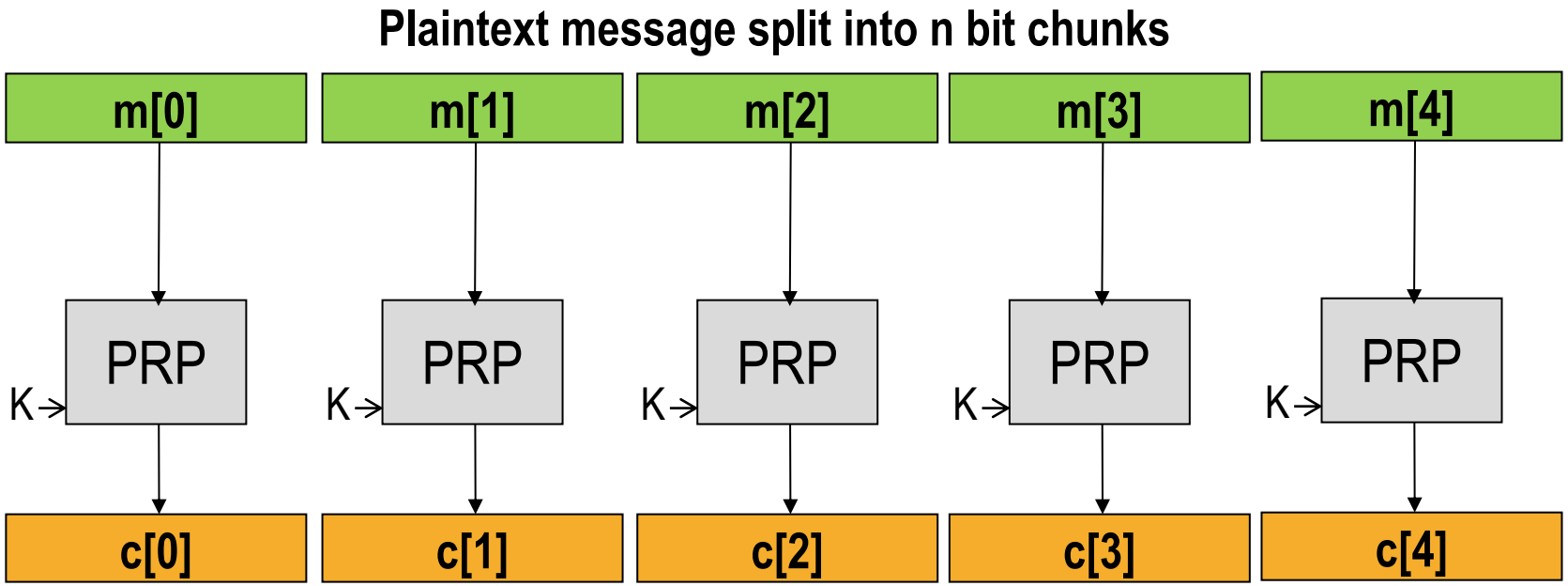
2
6
5
6

)

→AES keys: 128, 192, 256 → 2 << 2 ! (2 just 78 digits key)

⇨ Take home: Actual AES permutations way less that ideal PRP (but still OK)

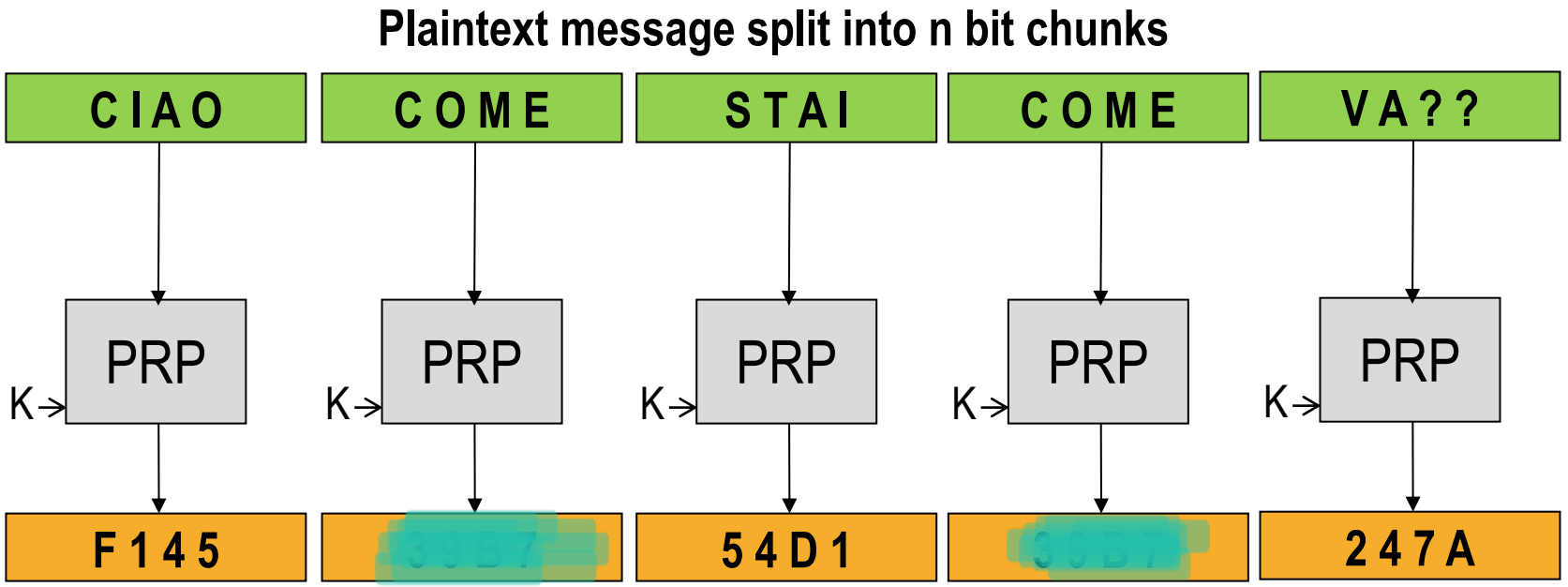
Problem 1
Plaintext longer than block size?



Independent encryption of each block = Electronic Code Book (ECB Mode)

se input: ABCD ho 4! permutazioni, di cui circa $\frac{4!}{e}$ non sostituiscono un valore con se stesso! (ABCD → BADC OK; → ACDB NO)

Problem 1
Plaintext longer than block size?



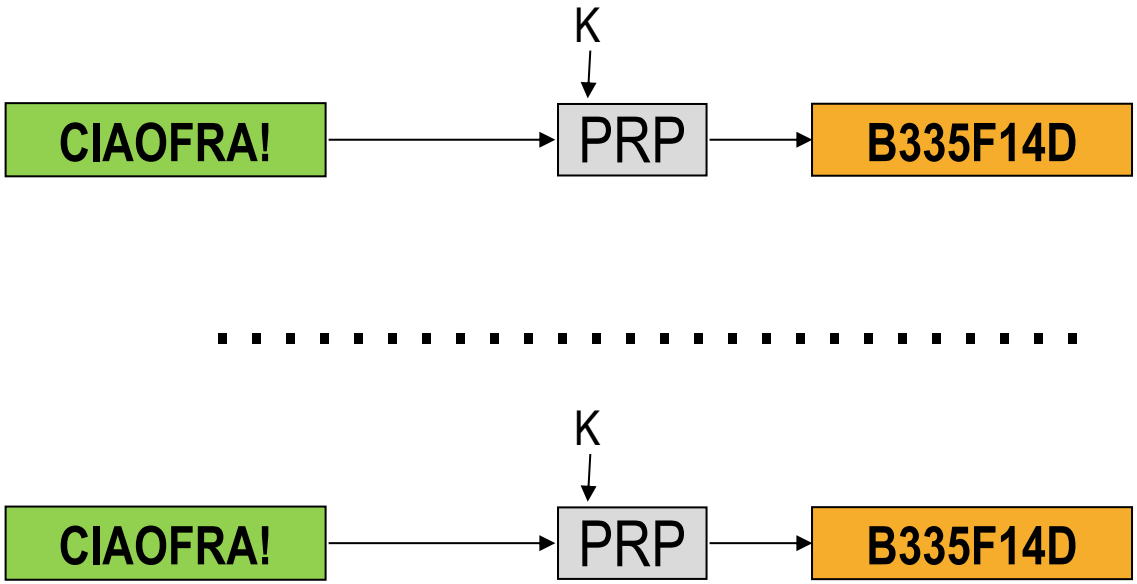
Independent encryption of each block = Electronic Code Book (ECB Mode)

Same plaintext block → Same ciphertext block!!!

No semantic security → Trivial Cryptoanalysis

REMEMBER: never, NEVER use ECB!!!

Problem 2
Encrypt same message twice?

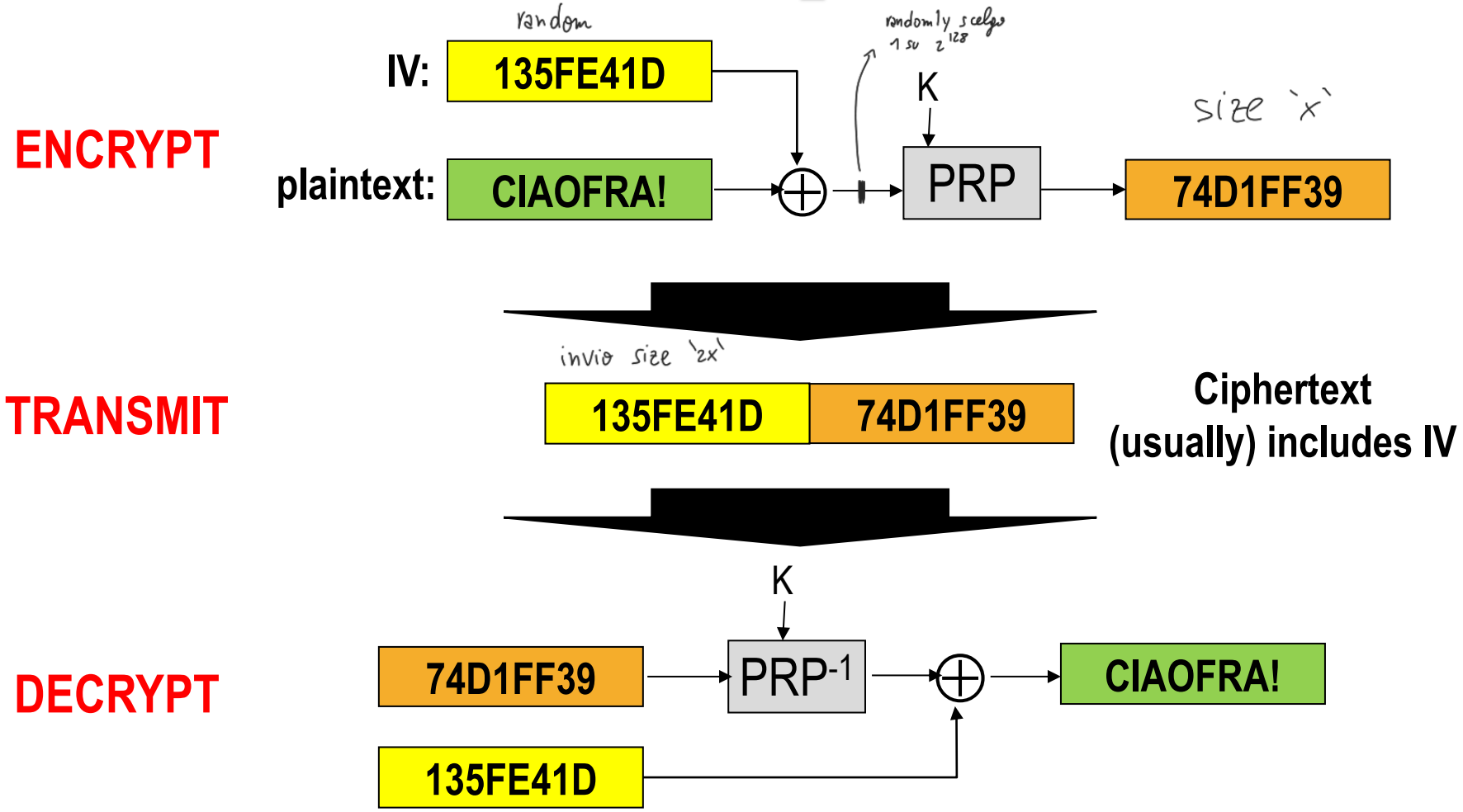


CT = ENC(PT, key) → same key, same PT = same CT!

Same problem as in stream ciphers → same idea!

A fresh Initialization Vector (IV) for every new encryption!

Initialization Vectors in block ciphers



IV NEVER REPEATS? YES BUT not enough! IV must now ALSO be UNpredictable (why? more later)

Block ciphers: modes of operation

➔ **Plain ECB is only OK if**

- ⇨ Message smaller than one block
  - →AES = 128 bits = 16 bytes, very rare as short
- ⇨ Message will never repeat

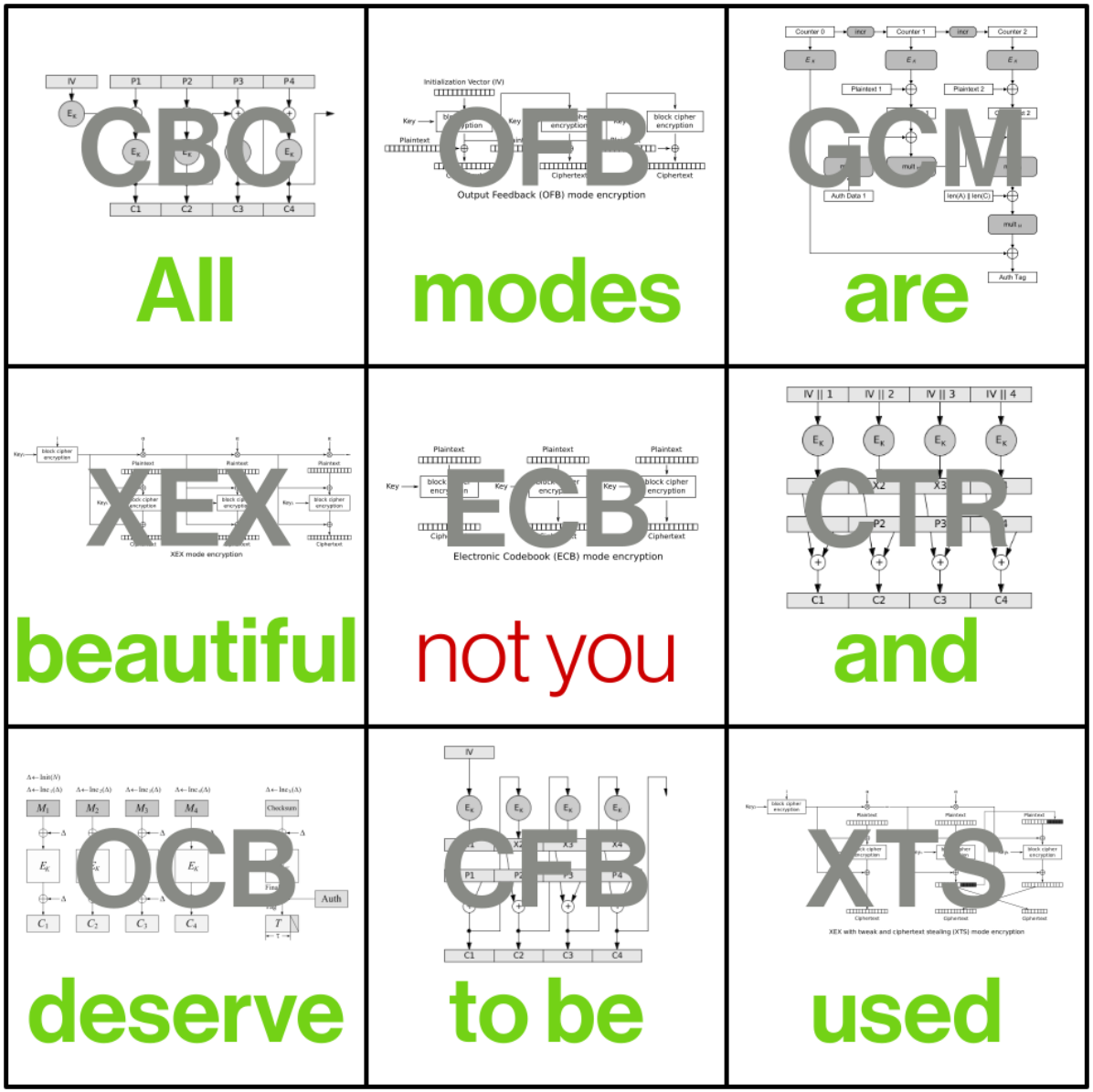
➔ **For repeated messages:**

- ⇨ Use initialization vector
  - →Random value, same size as block

➔ **For longer messages:**

- ⇨ **Modes of operation**: "secure" way to combine blocks
- ⇨ Permits to encrypt a text of arbitrary length, **avoiding that a same plaintext block is encrypted twice as same ciphertext**
- ⇨ "secure" = semantic security, IND-CPA

Block ciphers: modes of operation

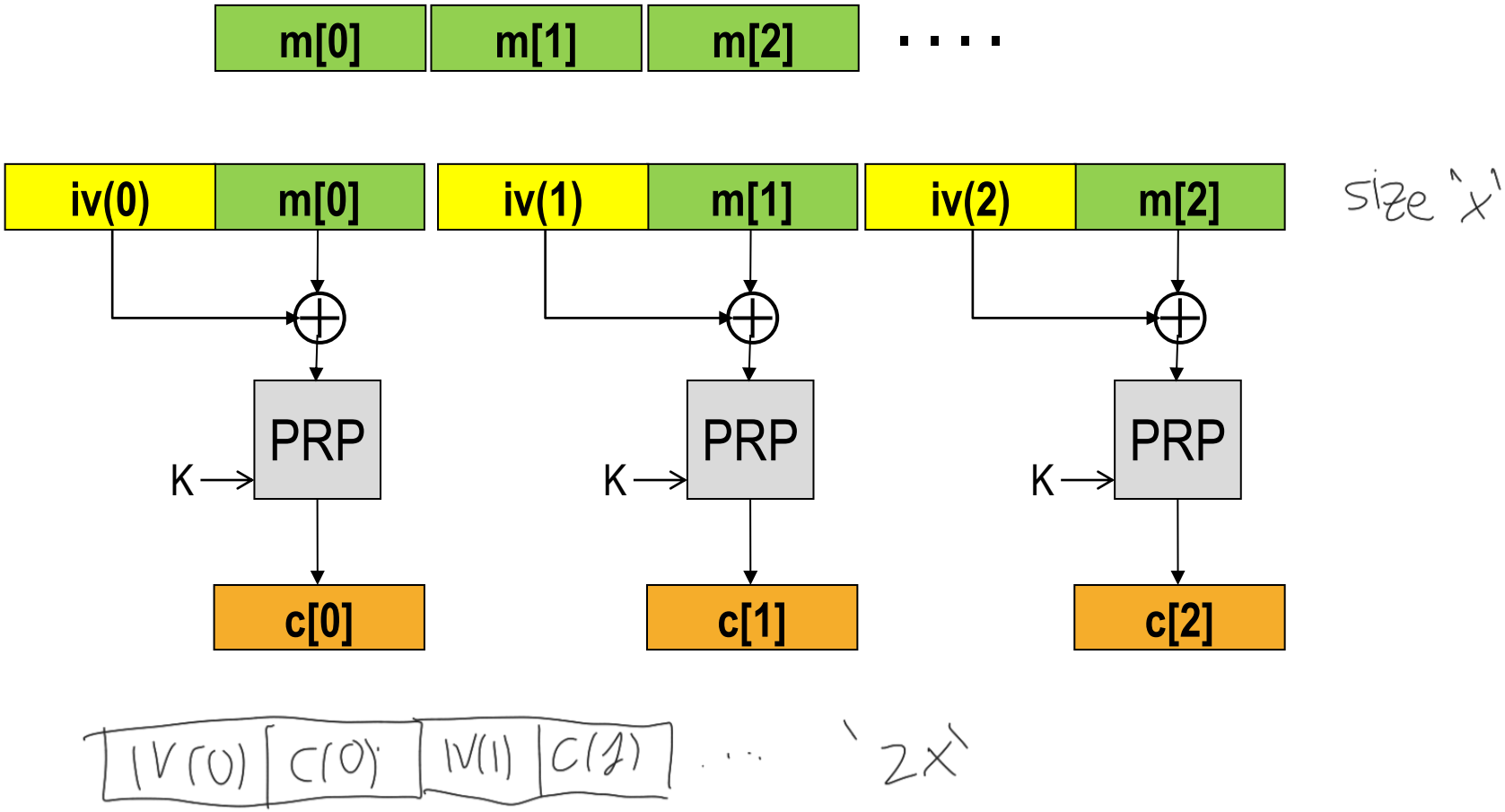


Most used:
Cipher Block Chaining (CBC)
Counter Mode (CTR)

NIST recommended (in 2001 document) but not much used in practice – despite quite interesting properties
Cipher Feedback Mode (CFB)
Output Feedback Mode (OFB)

More advanced properties (authenticated encryption)
Galois Counter Mode (GCM)
Offset Codebook Mode (OCB)
(more later)

Semantic security: how to? (in principle)



If all iv(x) random → semantic security. But LOTS of overhead! (double size)

Cipher Block Chaining - CBC

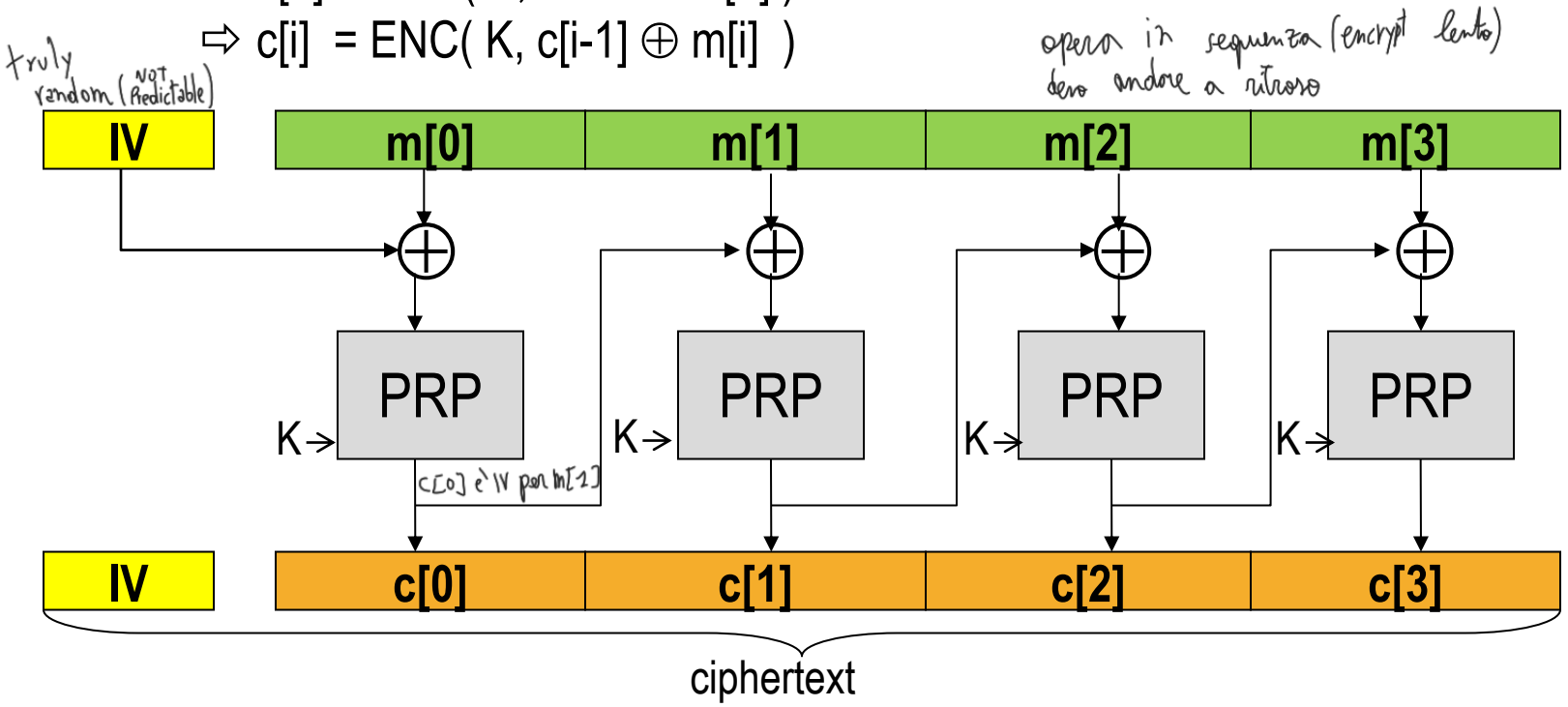
Pseudo Random Permutation

➔ **Good PRP → pseudorandom output → c[i] «looks» as random**

➔ **CBC idea: use previous ciphertext block as IV for next block!!**

⇨ c[0] = ENC(K, IV ⊕ m[0])

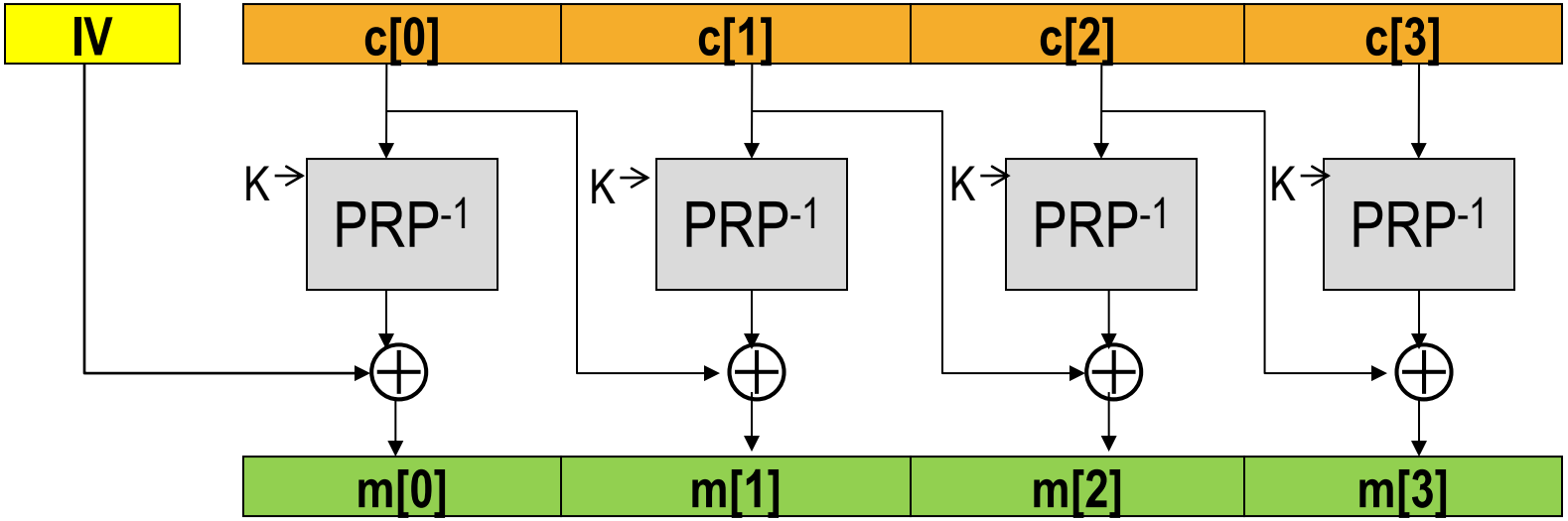
⇨ c[i] = ENC(K, c[i-1] ⊕ m[i])



Ciphertext overhead: just the initial IV → marginal!

CBC decryption

(più veloce)



➔CBC decryption: parallelizable!

⇨Unlike CBC encryption, which has to be sequential

➔not suitable for high speed HW implementation

CBC discussion

➔ **widely used (most common) & secure**

⇨ But secure only if Nonce is random and non predictable

➔Predictable IV ➔ CPA attack ➔ exploited in TLS (2011, BEAST)

➔ **Slow during encryption**

⇨ Non-parallelizable encryption ➔ not appropriate for very high performance scenarios

➔ **Encryption and decryption: require two different circuits** , come in (CBC, ECB) (block cipher)

⇨ Encryption = PRP, decryption = PRP^{-1} ➔ more circuitry

⇨ Other modes (CTR, CFB, OFB) use the PRP encryption block also in reception (direct transformation, usare block cipher come stream cipher)

➔ **Plaintext must be multiple of block size**

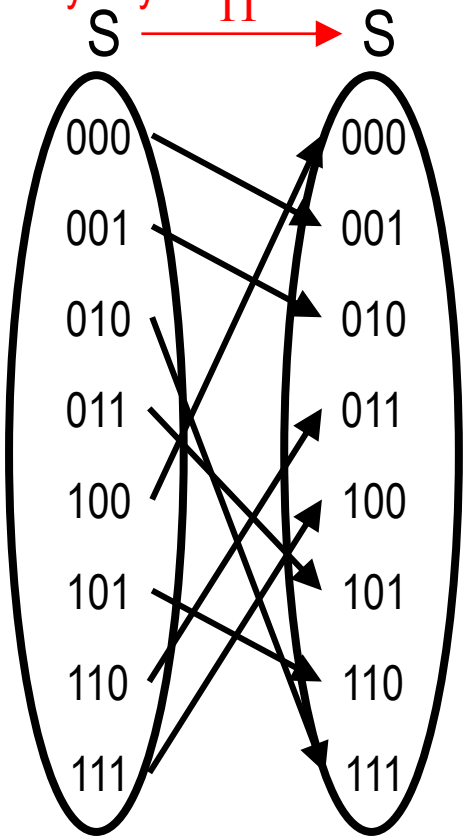
riempire i bit mancanti

⇨ Padding used to «fill» space ➔ standards do exist (e.g. PKCS#7)

⇨ Padding necessary also in (some) other modes, e.g. ECB

Test your understanding

3-bit blocks;
Specific PRP selected by key K:

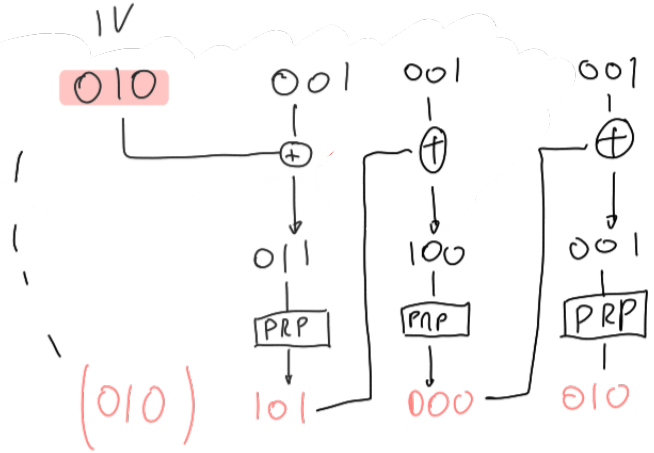


1) Encrypt M = 001 001 001 using ECB

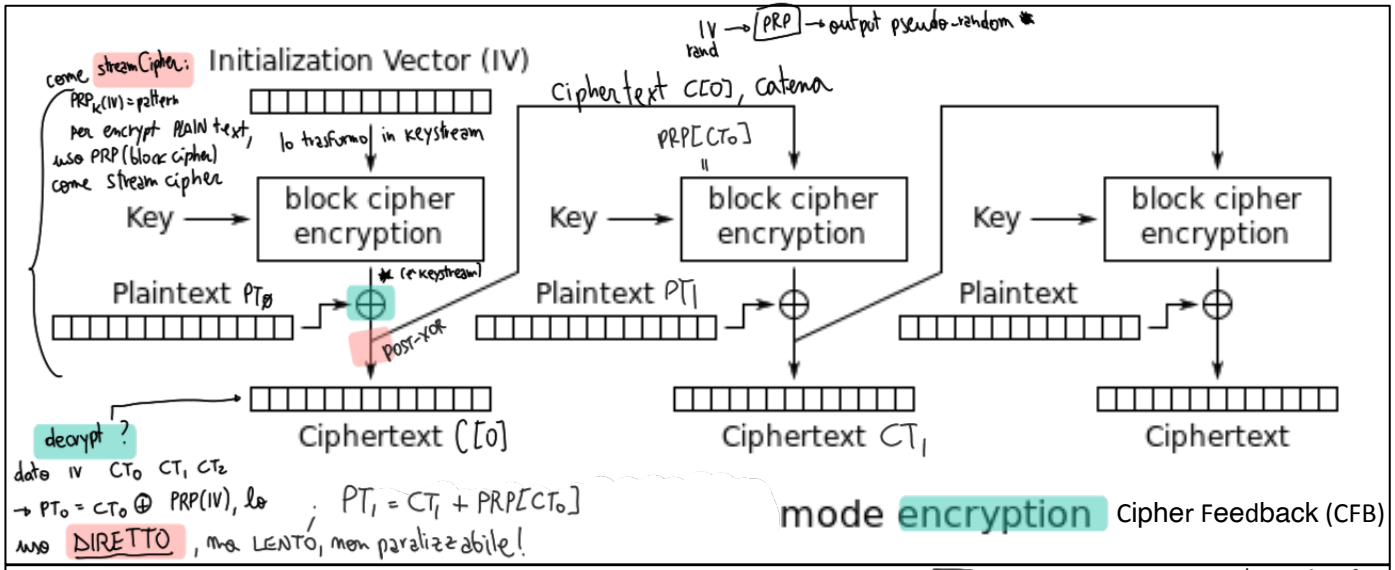
(uso $S \xrightarrow{\Pi} S$)

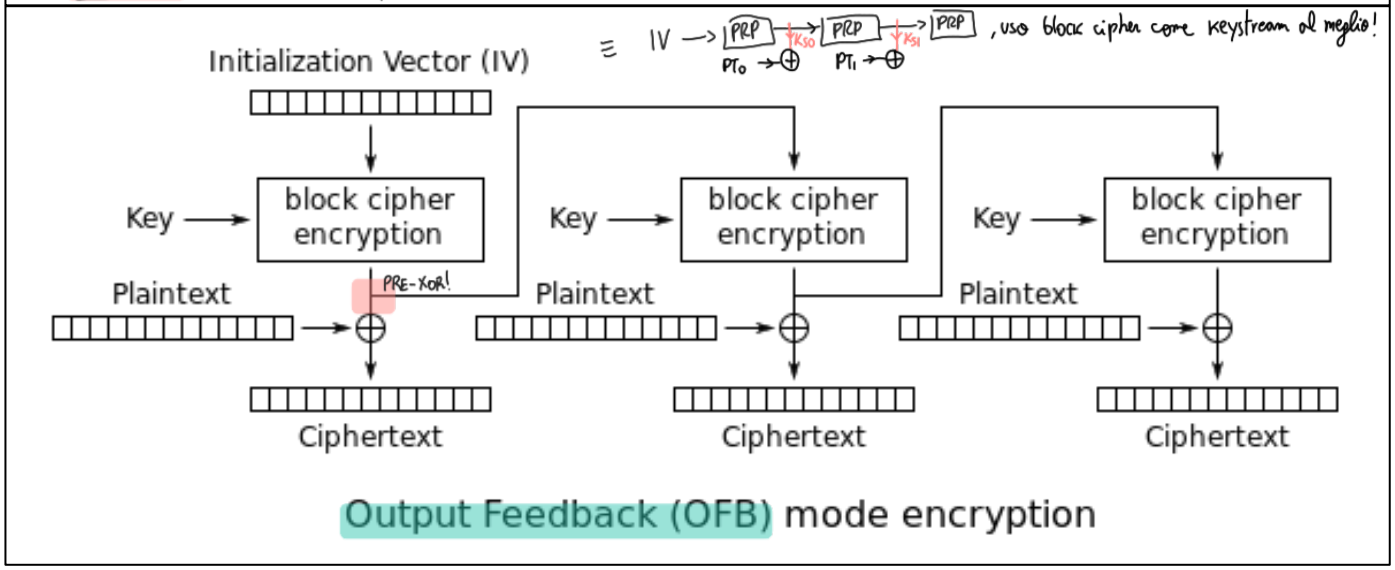
CT = 010 010 010

2) Encrypt M = 001 001 001 using CBC with IV=010 (prima ⊕, poi trasformo)



Other modes: CFB, OFB





Block cipher used in «stream» manner

- PT encrypted with XOR
- No need for padding

Non parallelizable in encryption

- But OFB advantage: permits preprocessing!

Reason: CFB, like CBC, depends on previous ciphertexts

- $c[0] = m[0] \oplus ENC_K(IV)$
- $c[i] = m[0] \oplus ENC_K(c[i-1])$

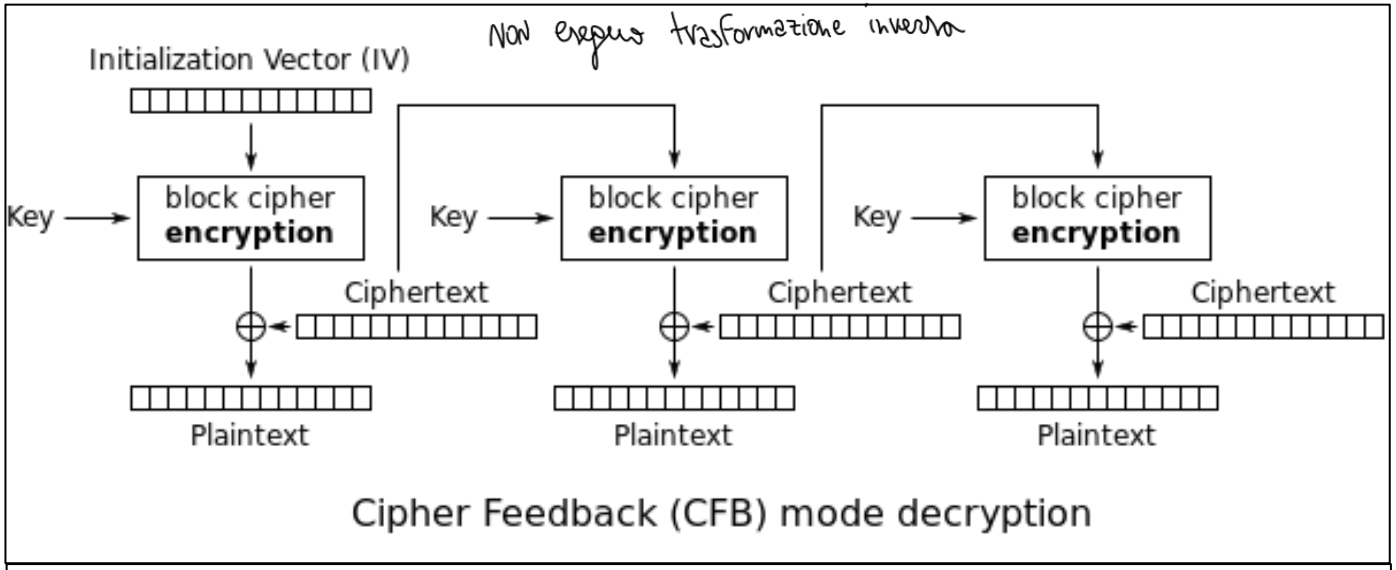
OFB «keystream» depends only on IV!

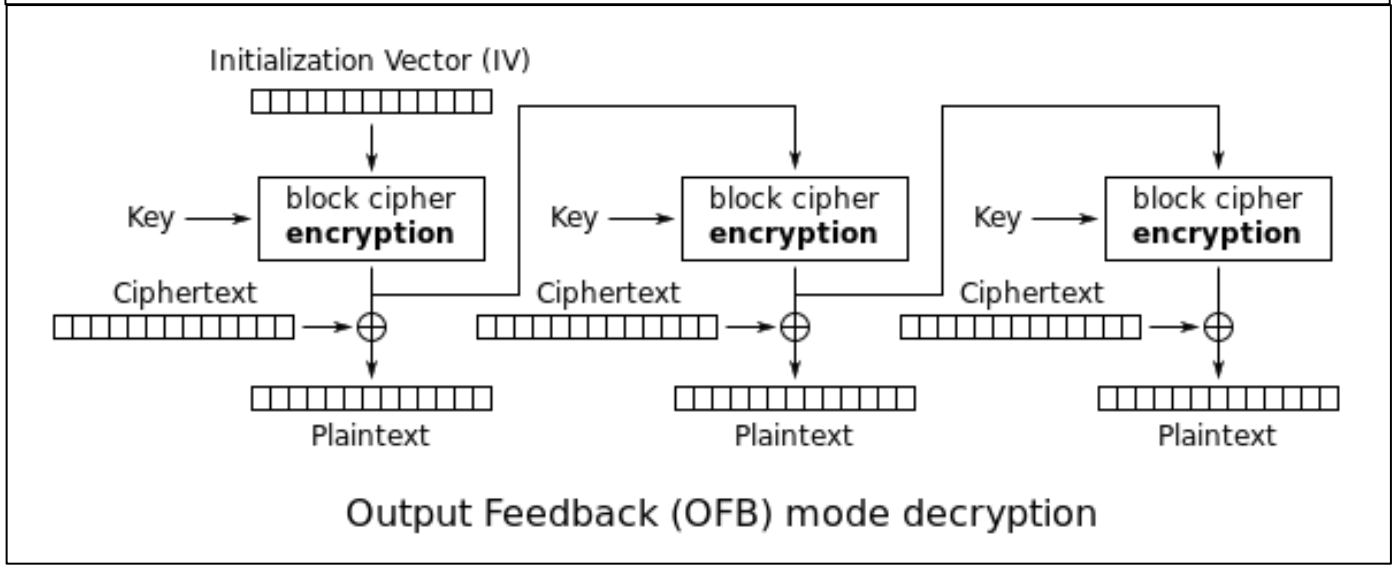
- **Robust to errors**

Looks like a classical stream cipher!

Graphics: source = Wikipedia

Other modes: CFB, OFB



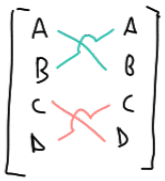


DECRYPTION advantage:
Reuse encryption block!
No need for «inverse» PRP implementation

Further CFB advantage:
Parallel decryption
OFB still serial decrypt

in CBC ho PRP,
in CFB ho 1° blocco in cui uso PRP (block cipher) come stream cipher.
Posso usare PRP come trasformazione diretta e NON indiretta.

voglio tutti diversi!



neanche questo è ok

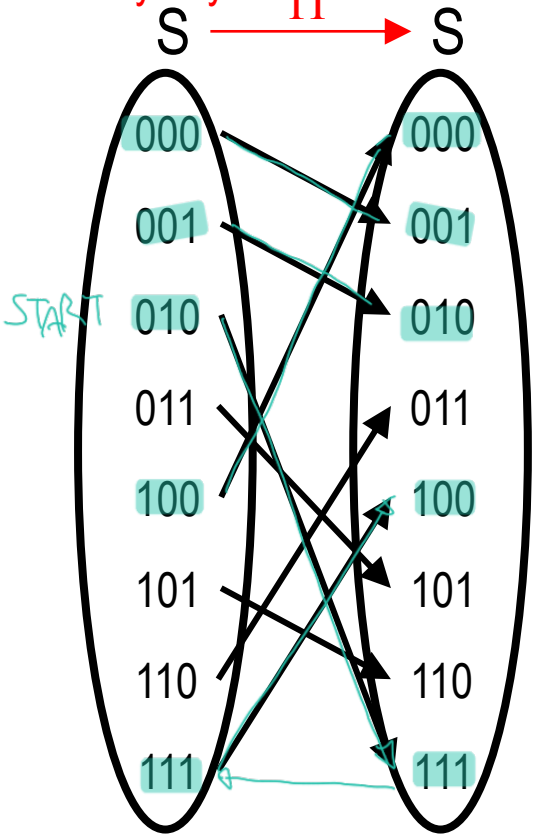
Chaining blocks:

HA SENSO BLOCK CIPHER PER KEYSTREAM? dovremmo usare algoritmi adatti! esistono IV 'deboli', generano sequenze brevi.

short cycle problem!

perché (hash chain non OK) (alcune permutazioni ne soffrono)

Example: 3-bit blocks; Specific PRP selected by key K: Π



1) OFB keystream, with IV=010?

010 111 100 000 001 010 111 …

cycle = 5 partendo da 'start', vado in $f(\pi) = 111$. Ritorno nel dominio partendo da 111 e trovo $f(\pi_{111})$ ….

2) OFB keystream, with IV=011?

011 101 110 011 101 110 011 …

cycle = 3 – shorter! Cycle may depend on IV…

3) Same problem in CBC when encrypting a «regular» plaintext.

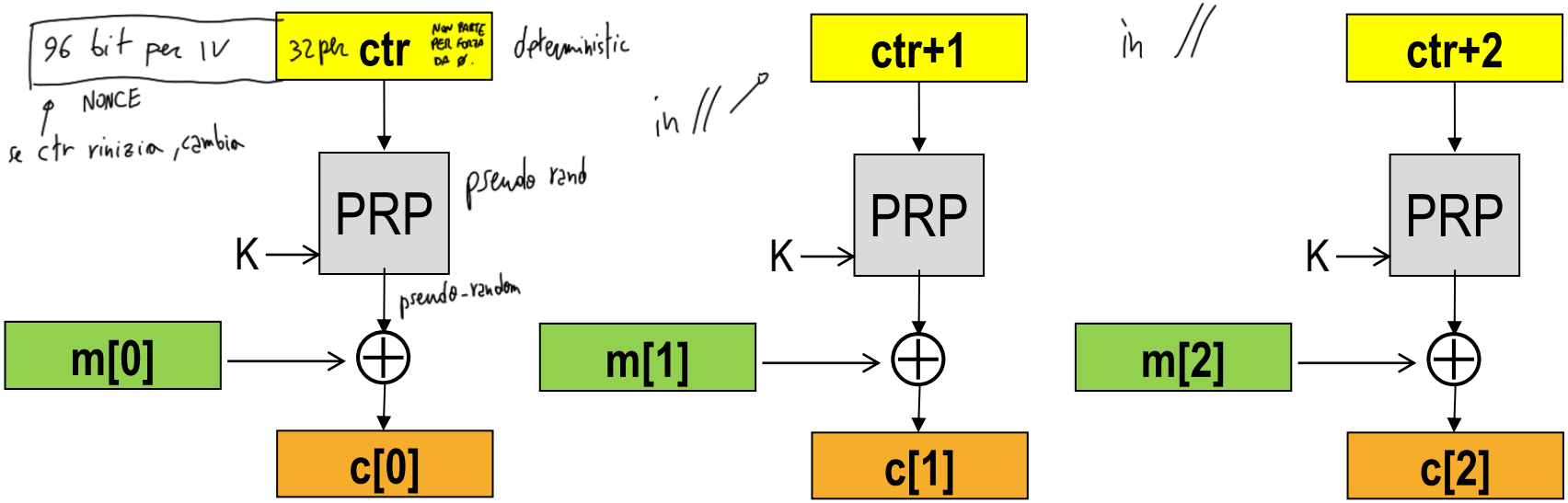
Example: encrypt 011 011 011 with IV=010

CT = (010) 010 010 010 (visto prima)

regular input, regular output

Counter Mode (CTR)

- **Very simple:**
  - Initialize counter ctr, and increase it at every new block
  - «encrypt» counter with block cipher (independent of plaintext, can be precomputed)
  - XOR output with plaintext block → looks like a keystream in stream cipher
- **In pratice: builds a PRNG keystream out of a block PRP!**
  - Provably secure: if PRP is secure then PRNG is secure as well



AES-CTR detail on Initial Counter Block (ICB), see e.g. RFC3686:
first 96 bits = IV, next 32 bits = counter, starting from 1

encryption, No integrity

Counter Mode (CTR)

CBC veniva più usata per 'abitudine'! oggi si cerca authenticated encryption!

- **Mode with most of the advantages!**
  - ⇨ Turns block cipher into a stream one
  - ⇨ Combines all the advantages of CFB and OFB!
- **Implementation efficiency (SW/HW)**
  - →Parallel encryption AND Parallel decryption
- **Requires implementation of encryption block only**
  - ⇨ Unlike ECB and CBC which require also inverse block
- **Random access**
  - ⇨ Decryption of i^{th} block does not depend on previous blocks
    - →Unlike CBC or CFB, where you need to decrypt all previous blocks
- **Secure**
  - ⇨ Counters (if properly used) do not repeat! Predictability NOT a concern!
  - ⇨ Guaranteed NO short cycle problems!
    - →Since block is permutation, same output when same input repeats
      - » periodicity = 2^n

esempio CTR

Dato 001 001 001 e ICB = 100 (Block counter)

⊕ ⊕ ⊕

PRP[100] PRP[100+1] PRP[100+2]

|| || ||

001 111 010

risultati considerando PRP nello schema $S \xrightarrow{\pi} S$