

## CIFRARI MODERNI

Oggi si usa la potenza del calcolatore.

Seppur possa sembrare il contrario, quest'ultima e' mosta amica:

Un ATTACCO DI FORZA BRUTA permette al mostro avversario di "rompere" il mostro cifrario: piu' il calcolatore e' potente piu' il mostro avversario riesce a fare un attacco esaustivo.  
E vero ... ma possiamo usare anche noi la potenza del calcolatore a nostro vantaggio!

## MACCHINA A ROTORI

E' una apparecchiatura eletro- meccanica che assomiglia ad una macchina da scrivere.

- Ognuno dei  $K$  rotori individua una SOSTITUZIONE MONOALFABETICA
  - Ogni rotore "girando" porta ad una diversa sostituzione, ripetendosi dopo  $N$  volte.
  - Ottieniamo cosi'  $N^K$  SOSTITUZIONI
  - Per ogni carattere del testo cambia la SOSTITUZIONE MONOALFABETICA utilizzata.
- 

Questa e' storia! ... che ci porta a CIFRARI MODERNI.

Quelli che analizzeremo di seguito, sono tutti esempi di cifrare un blocco di lunghezza fissa...

**CIFRARI A BLOCCHI**

## DES - Data Encryption Standard

14/03/2023

Pubblicato nel 1977;

E' stato lo standard dei CIFRARI SIMMETRICI fino al 2002.

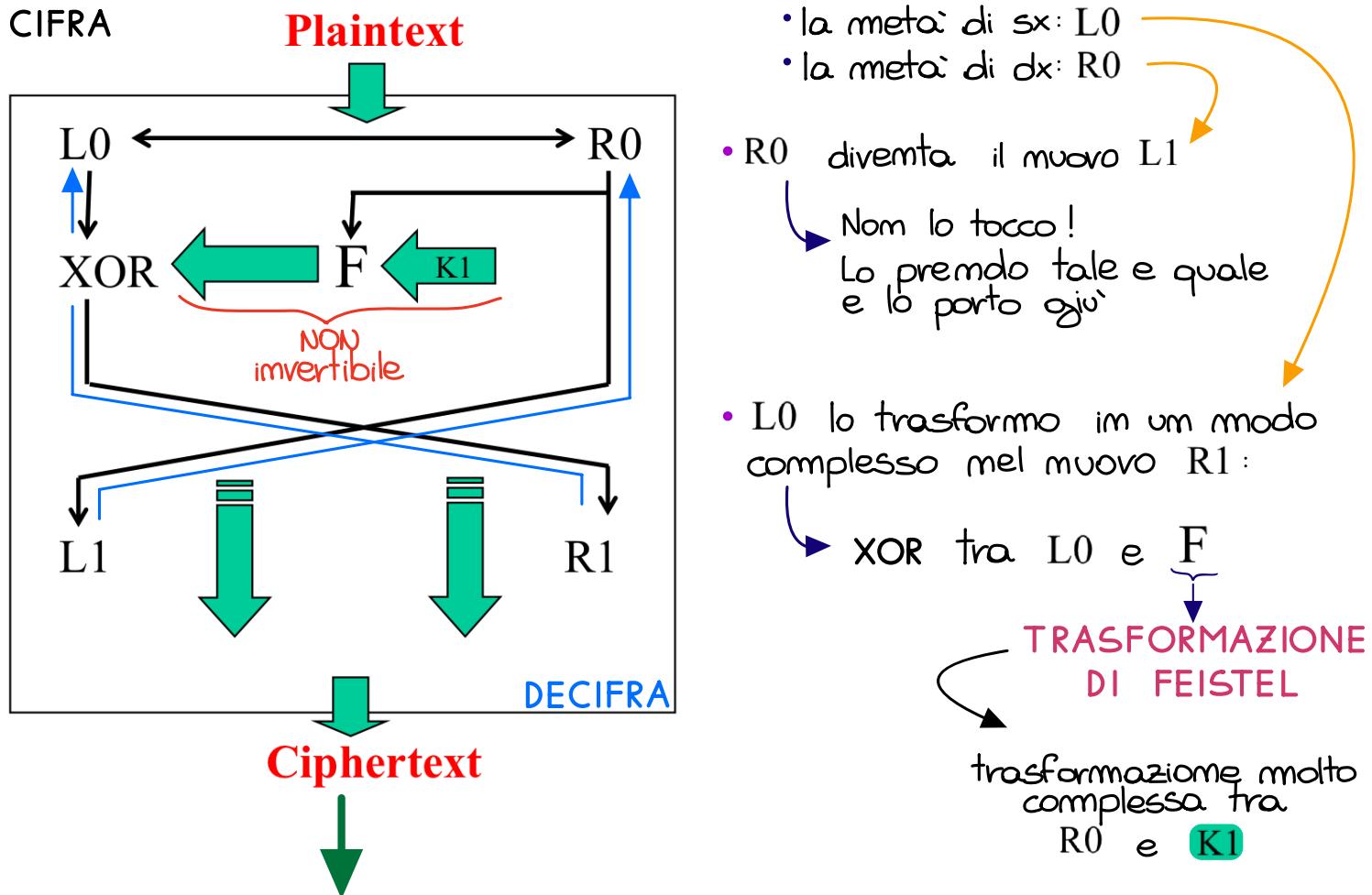
Sostituito da **3-DES** e **AES**

Si basa sul concetto di "diffusione" e "confusione" dell'informazione.

Il **DES** e' un caso di **FEISTEL CIPHER**

# FEISTEL CIPHER

CIFRA



Se mi fermassi qui, mom potrei minimamente ritemermi soddisfatto poiché avrei cifrato solo metà testo (l'altra metà è ancora in chiaro!)

Lo ripeto! Almeno 2 volte.

In realtà **DES** usa: blocco di 64 bit  
chiave **K** di 56 bit  
16 ROUD

**Chiave DES** di 56 bit  
(solitamente estesa a 64 bit con bit di parità)



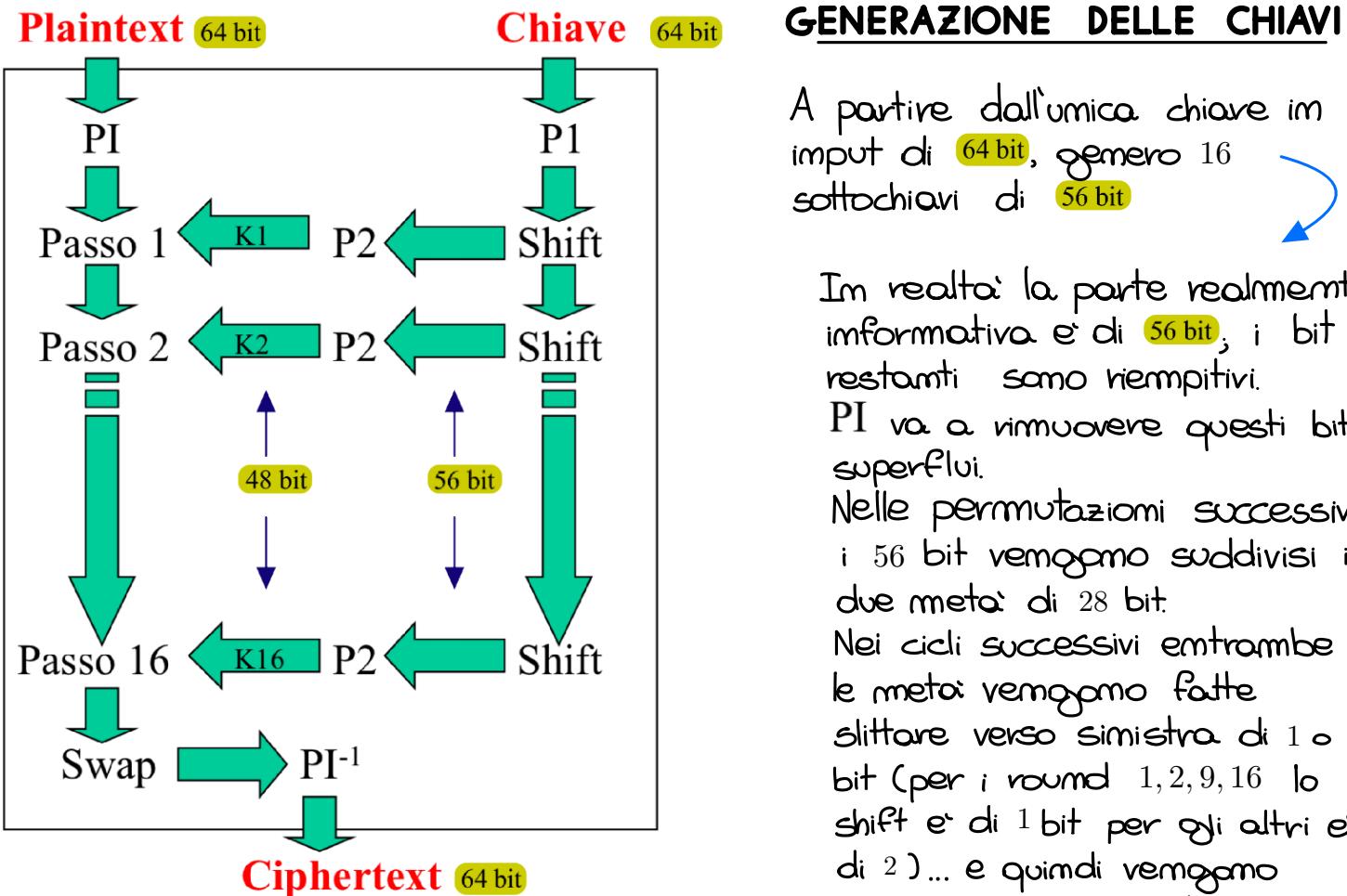
Possiamo già anticipare che, questo, è un cifrario molto efficiente poiché consiste in poche istruzioni macchina e sicuro

oggi non più poiché non è difficile provare  $2^{56}$  chiavi

Ad ogni passo viene usata una sotto-chiave (Key schedule) prodotte dalla chiave principale.

→ Non è un simonimo di sicurezza: si pensi ad un drink ammaccato; la quantità di alcool è sempre la stessa.

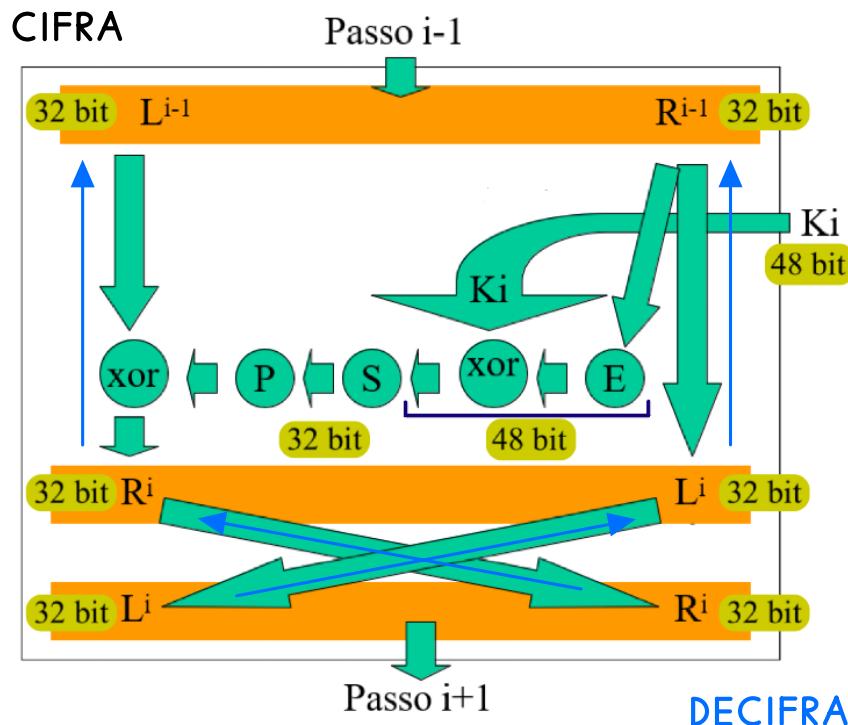
Analizziamo più nel dettaglio l'implementazione del DES:



$PI$  e  $PI^{-1}$  sono rispettivamente la permutazione iniziale e finale, che sono tra di loro inverse:  $PI$  "disfa" l'azione di  $PI^{-1}$  e viceversa.

## PASSI

Ogni passo ha un input una chiave diversa e l'output del passo precedente:



Consiste di quattro passi:

- **ESPANSIONE** E
- **MISCELAZIONE** XOR
- **SOSTITUZIONE** S
- **PERMUTAZIONE** P

**ESPANSIONE** (Ammacquamento) → Il mezzo blocco di 32 bit è espanso fino a 48 bit, duplicando alcuni bit.

**MISCELAZIONE** (con la chiave) → XOR tra la sottochiave  $i$ -esima ed il risultato dell'operazione precedente.

**SOSTITUZIONE** (S-box) → Il blocco viene diviso in 8 parti di 6 bit che vengono processate con le **S-BOX**

Substitution Box  
scatola di sostituzione

$2^6 = 64 \rightarrow 2^4 = 16$  possibilità  
se scelte male, il DES  
può diventare vulnerabile

sostituisce 6 bit in input con 4 bit in output, mediante una trasformazione non lineare effettuata attraverso una tabella.  
Rappresentiamo il vero cuore del DES

senza di esse, la cifratura  
sarebbe lineare e quindi  
facilmente violabile

**PERMUTAZIONE** (P-box) → I 32 bit risultanti dalle S-BOX sono riordinati in base alle permutazioni fisse della **P-BOX**

Permutation Box  
scatola di permutazione

Tutto ciò fornisce la così detta confusione e diffusione: un concetto identificato come condizione necessaria per rendere pratica e sicura la cifratura.

Come abbiamo già accennato il DES è:

**Efficiente!** poche istruzioni macchine che rendono il cifrario molto veloce

**Sicuro!** NON sono molte vulnerabilità di matrice algoritmica

**Tuttavia** l'unica vulnerabilità reale è relativa alla lunghezza della chiave

→ sono stati sviluppati software e hardware in grado di violare il DES (DES challenge)

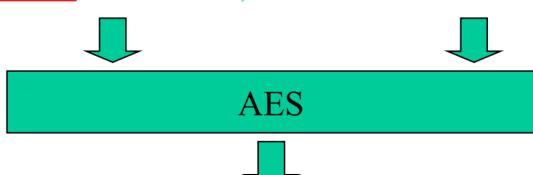


Sostituito da  
**3-DES** e **AES**

L'AES usa un blocco di 128 bit e, a seconda della versione:  
 una chiave  $K$  di 128 bit oppure di 192 bit oppure di 256 bit  
 10 ROUD oppure 12 ROUD oppure 14 ROUD

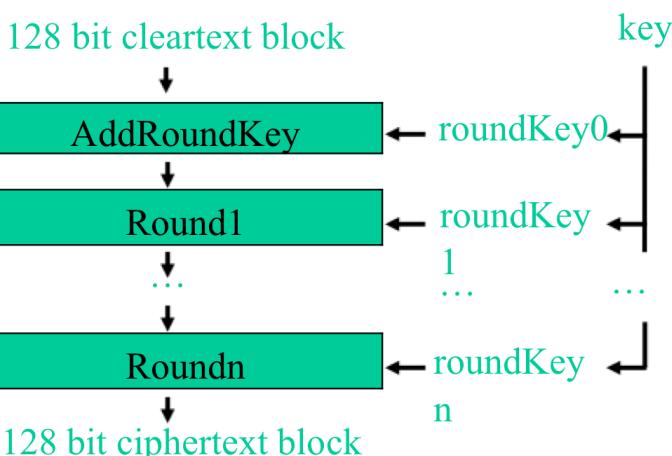
Testo in chiaro di 128 bit  
(Plaintext o Cleartext)

Chiave AES di 128 bit  
 oppure di 192 bit oppure  
 di 256 bit



E' stato progettato per macchine a 64 bit.

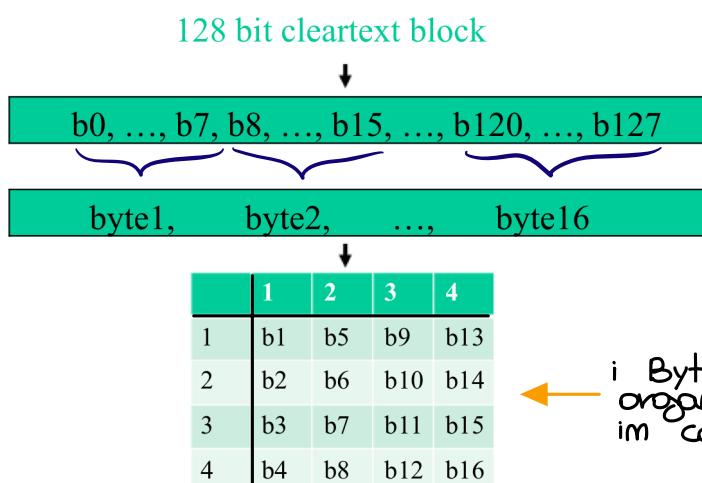
E' robusto rispetto ad attacchi lineari e differenziali.



Come il DES, anche lui utilizza delle sotto-chiavi (ROUD-KEY)

NON e' un caso di FEISTEL

I round non sono tutti uguali



Divido il testo in chiaro (128 bit) in (16) Byte e li sistemo in una matrice: **MATRICE DI STATO**  
 Quest'ultima rappresenta l'input (e l'output) di ciascun round

i Byte sono organizzati in colonna

Vedo questi Byte come elementi di un **CAMPO FINITO GF(2<sup>8</sup>)**, questo al fine di effettuare delle operazioni più complesse su questi Byte.  
 Ad esempio se io sommo, o peggio, moltiplico due Byte rischio di avere un overflow.

L'idea e' definire la SOMMA e il PRODOTTO in modo diverso.  
Obiettivo: produrre un output che sia ancora un Byte.  
Ai nostri scopi ci basta sapere che c'e' dietro una teoria basata sull'aritmetica dei polinomi:

**SOMMA** definita come XOR

**PRODOTTO** definito in modo che ci garantisce due cose:

- moltiplicando due Byte otteniamo ancora un Byte
- esiste sempre un Inverso Moltiplicativo tale che

$$b \cdot b^{-1} = 00000001 = (1)_{10}$$

Avendo definito tali operazioni, posso anche fare somma e prodotto tra matrici

## PRODOTTO

Sia  $b = b_7b_6b_5b_4b_3b_2b_1b_0$ .

Moltipicare per 2 in aritmetica binaria, significa shiftare a sx di 1 bit ed aggiungere uno 0 a dx.

Ottieniamo cosi'  $y = b_6b_5b_4b_3b_2b_1b_00$

Dopodiche' abbiamo due casi:

$b_7 = 0 \rightarrow$  Output:  $y$  NO problem!  
poiche' non c'e' riporto

$b_7 = 1 \rightarrow$  Output:  $y \oplus 00011011$  C'e' un problema.  
poiche' c'e' riporto!

↑  
funge da  
modulo

Questa e' la moltiplicazione per 2.

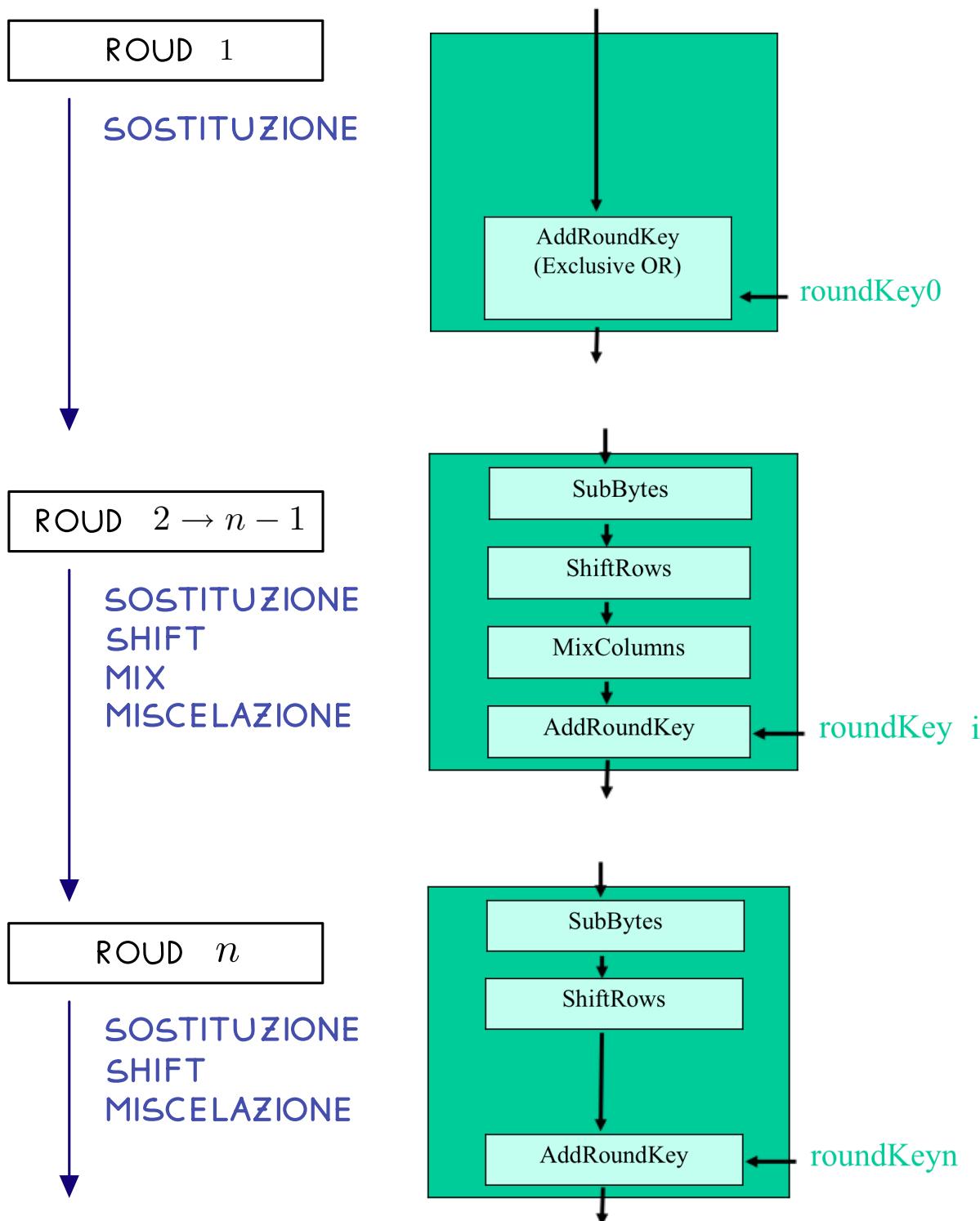
Per tutti gli altri valori ( $2^n$ ) analogamente, basta ripetere  $n$  volte la moltiplicazione per 2

# ROUD

Ogni round ha im input una MATRICE DI STATO ed una ROUD-KEY di 128 bit (oppure di 192 bit oppure di 256 bit)

→ a differenza del DES, la sottochiave ha la stessa lunghezza di quella primaria.

Ogni round produce im output una MATRICE DI STATO.  
I round non sono tutti uguali e sono così definiti:



Analizziamo più nel dettaglio le operazioni:

## SOSTITUZIONE

SubBytes

Sostituzione di Byte tramite S-box.

Definito nello standard in cui il Byte espresso in esadecimale viene convertito incrociando la rispettiva riga e colonna.

Ad esempio:  $(33)_{16} = Sbox(3, 3) = (C3)_{16}$

E' possibile trovare anche la sostituzione inversa:

$$(00)_{16} \rightarrow (63)_{16}$$

$\nwarrow$

$(52)_{16}$

dir	0	1	2	3	4	5	6	7	...	F
0	63	7C	77	7B	F2	6B	6F	C5		
1	CA	82	C9	7D	FA	59	47	F0		
2	B7	FD	93	26	36	3F	F7	CC		
3	4	C7	23	C3	18	96	5	9A		
4	9	83	2C	1A	1B	6E	5A	A0		
5	53	D1	0	ED	20	FC	B1	5B		
6	D0	EF	AA	FB	43	4D	33	85		
	⋮								⋮	
									...	
										F

S-Box creata a partire dagli inversi moltiplicativi di  $GF(2^8)$  a cui viene moltiplicato una matrice costante ed aggiunto un vettore costante.

## SHIFT

ShiftRows

Shift sulle righe di un certo numero di posizioni. Detto in altre parole, faccio delle PERMUTAZIONI.

- 1<sup>a</sup> riga → shift di 0 Byte
- 2<sup>a</sup> riga → shift di 1 Byte
- 3<sup>a</sup> riga → shift di 2 Byte
- 4<sup>a</sup> riga → shift di 3 Byte

	1	2	3	4
1	b1	b5	b9	b13
2	b2	b6	b10	b14
3	b3	b7	b11	b15
4	b4	b8	b12	b16



	1	2	3	4
1	b1	b5	b9	b13
2	b6	b10	b14	b2
3	b11	b15	b3	b7
4	b16	b4	b8	b12

L'inverso di tale operazione è lo shift a destra di  $m$  posizioni

**MIX**

MixColumns

Moltiplico la matrice per una mat. costante

scelgo numeri così piccoli per un fatto di efficienza

02, 03, 01, 01  
01, 02, 03, 01  
01, 01, 02, 03  
03, 01, 01, 02

di cui esiste anche la matrice inversa

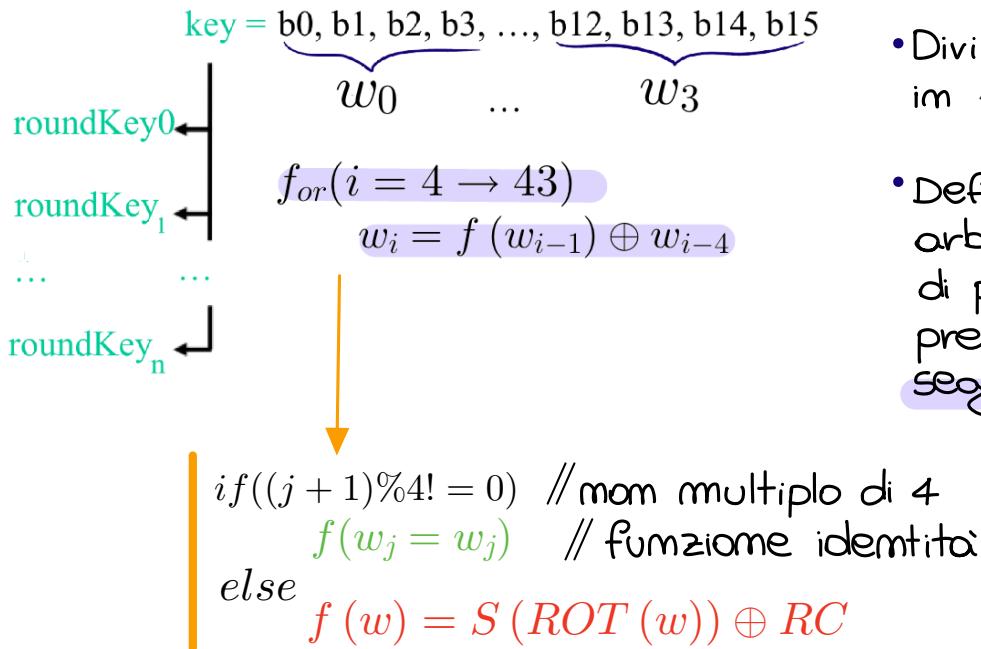
## MISCELAZIONE (con la chiave)

AddRoundKey

XOR tra la sottochiave  $i$ -esima ed il risultato dell'operazione precedente.

## GENERAZIONE DELLE CHIAVI

La generazione delle ROUND-KEY avviene in maniera algebrica:

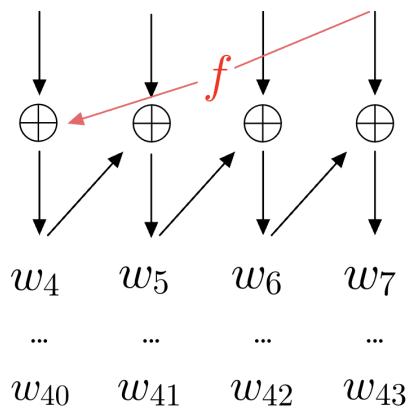


- Divido la chiave principale in 4 parole di 4 Byte.
- Definisco una sequenza arbitrariamente lunga (<sup>es.</sup> 43) di parole, a partire dalle precedenti secondo la seguente formula

dove:  $S = S - box$ , sostituzione di 1 Byte

$w_0 \quad w_1 \quad w_2 \quad w_3 \quad ROT = \text{Rotazione, shift a sx di 1 Byte}$

$RC = \text{Costante così definita.}$



$[C_{ROUND}, 0, 0, 0]$

ultimi 3 Byte pari a 0

1° Byte cambia con il cambiare del round C:

$C_{ROUND} = (01, 02, 04, 08, 10, 0, 40, 80, 1B, 36)$

1° round

10° round

Tutto ciò complica fortemente la vita al nostro avversario poiché si introduce un forte elemento di non linearità nel calcolo.

Ad oggi infatti, l'AES è un cifrario sicuro ed efficiente, ampiamente utilizzato.

## EXTRA

### STEGANOGRAFIA

Nella crittografia noi nascondiamo una informazione. In alcuni contesti però, ad esempio militari, può essere necessario cifrare senza che l'avversario capisca che stiamo cifrando.

Un esempio può essere leggere la prima lettera di ogni parola.