LA CRITTOGRAFIA SIMMETRICA

Sicurezza informatica

Elena Maria Dal Santo

elenamaria.dalsanto@its-ictpiemonte.it

Intervento realizzato da
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

La crittografia simmetrica

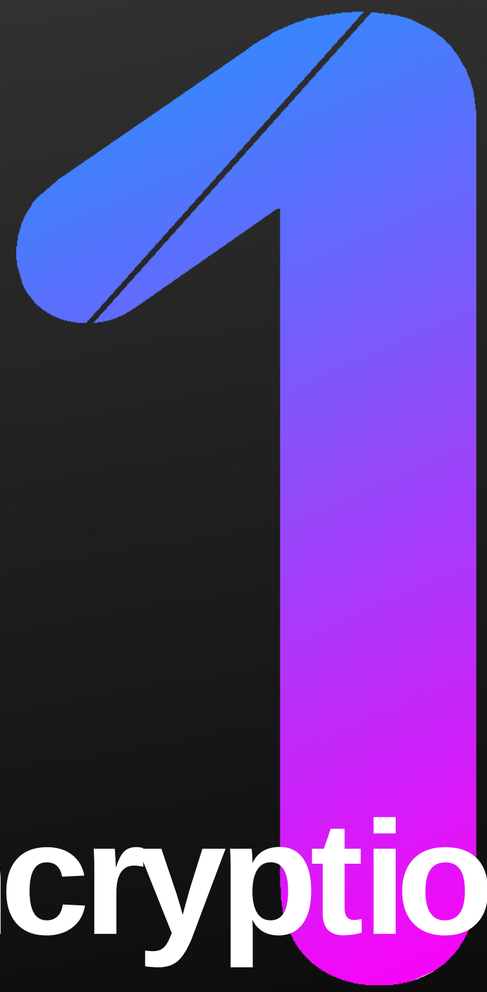
Detta anche crittografia a chiave privata

- La chiave usata per criptare il messaggio è la stessa che usiamo per decrittarlo
- La chiave deve essere protetta da eventuali nemici

La crittografia simmetrica

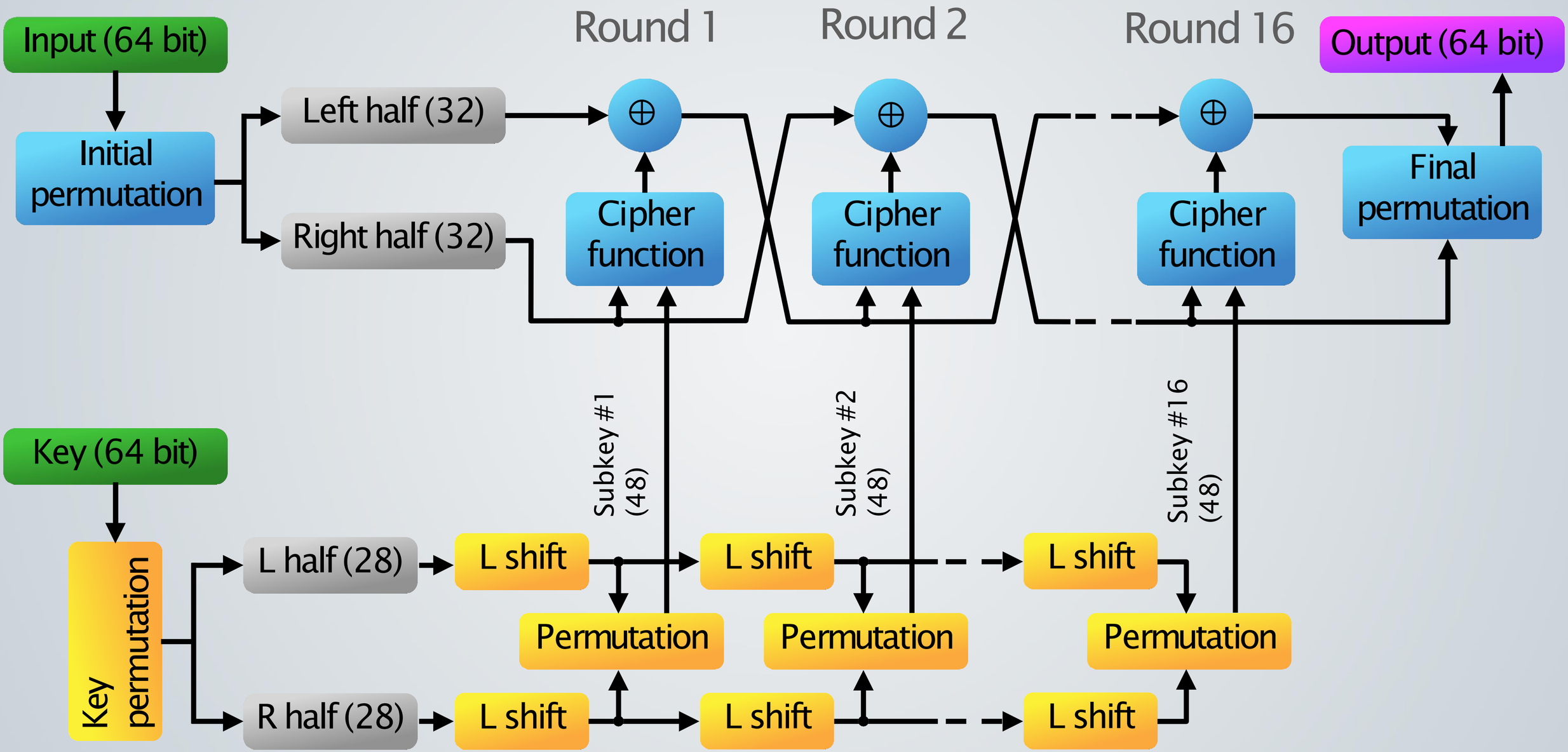
Tutto ciò di cui abbiamo parlato finora, riguardava la crittografia simmetrica.

(nel test di Kasiski, la password EMILY è stata usata sia per criptare che per decriptare)

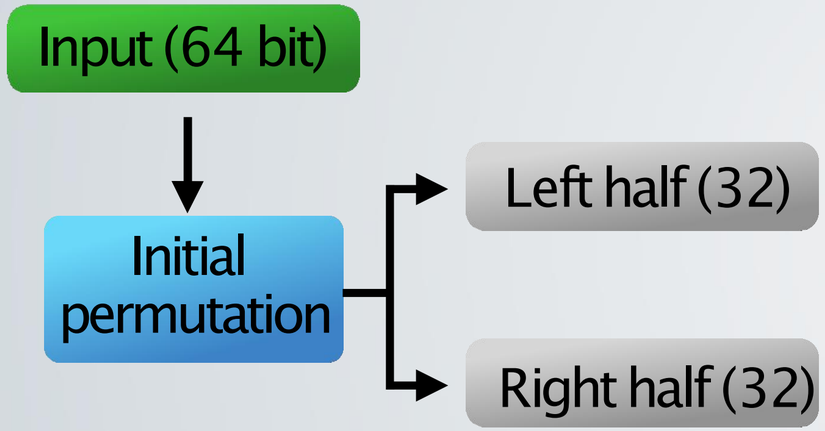


Data Encryption Standard

DES


Round 1
Round 2
Round 16
Input (64 bit)
Initial permutation
Left half (32)
Right half (32)
Cipher function
Cipher function
Cipher function
Final permutation
Output (64 bit)
Key (64 bit)
Key permutation
L half (28)
R half (28)
L shift
L shift
L shift
L shift
L shift
L shift
Permutation
Permutation
Permutation
Subkey #1 (48)
Subkey #2 (48)
Subkey #16 (48)

DES

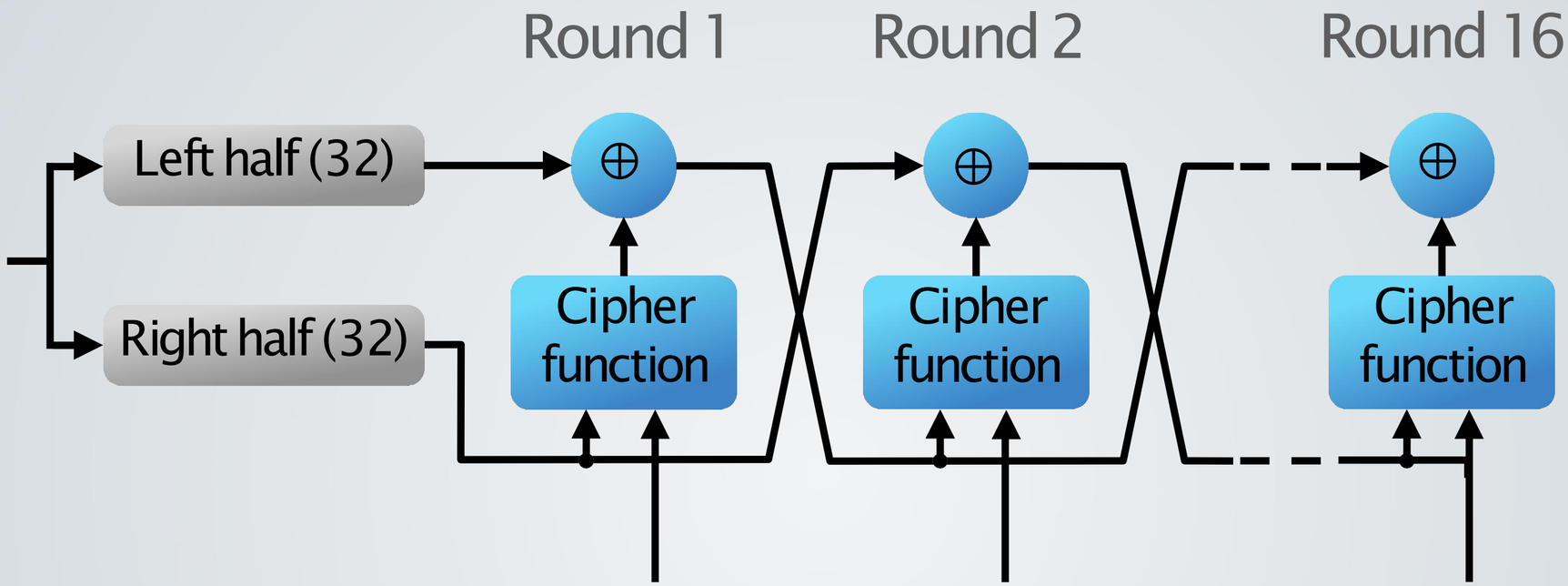


IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Il primo bit in output è preso dal 58esimo in input, il secondo in output è preso dal 50esimo, ecc…

DES



Ad ogni round

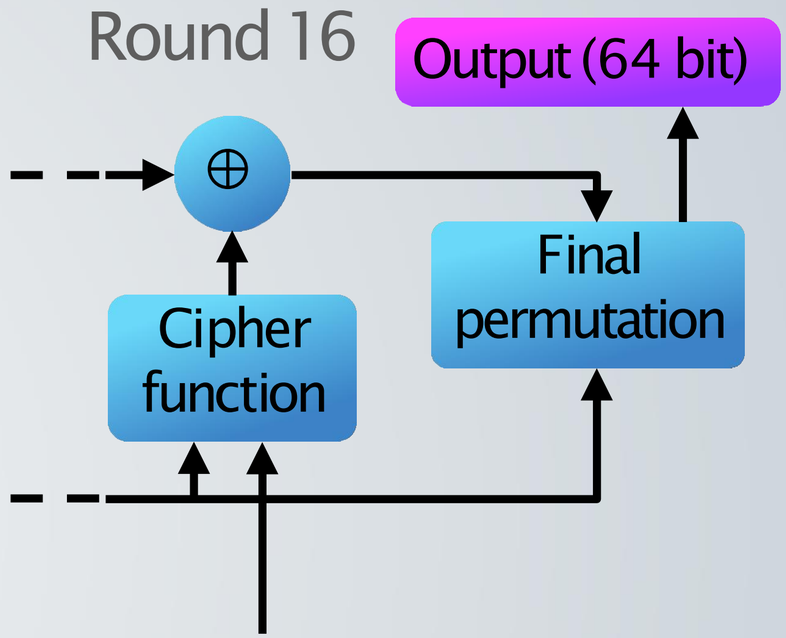
1. una delle due metà viene ‘’mescolata’’ con la chiave.
2. Le due metà vengono unite tra di loro tramite XOR
3. Le due metà vengono scambiate

A	B	A⊻B
0	0	0
0	1	1
1	0	1
1	1	0

DES

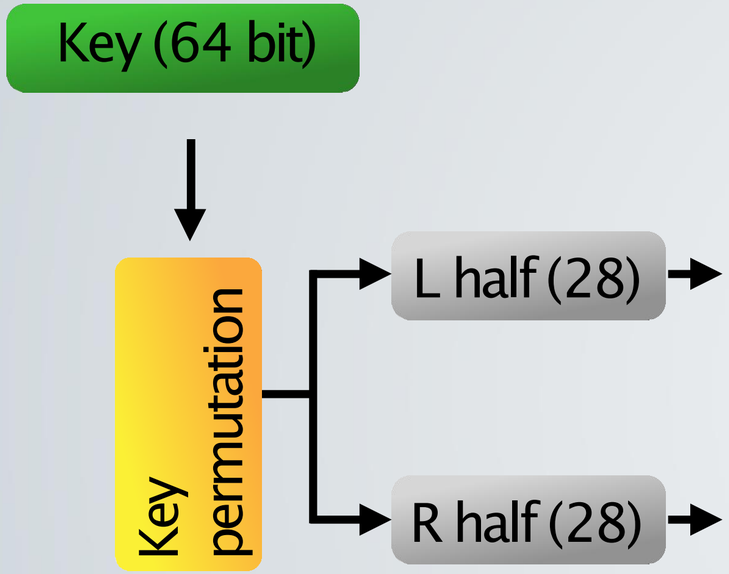
IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25



Dopo il ciclo finale, le metà non sono scambiate (per rendere cifratura e decifratura più simili)
Viene eseguita una permutazione finale (IP^{-1}), prima di restituire l'output.

DES – Gestore della chiave

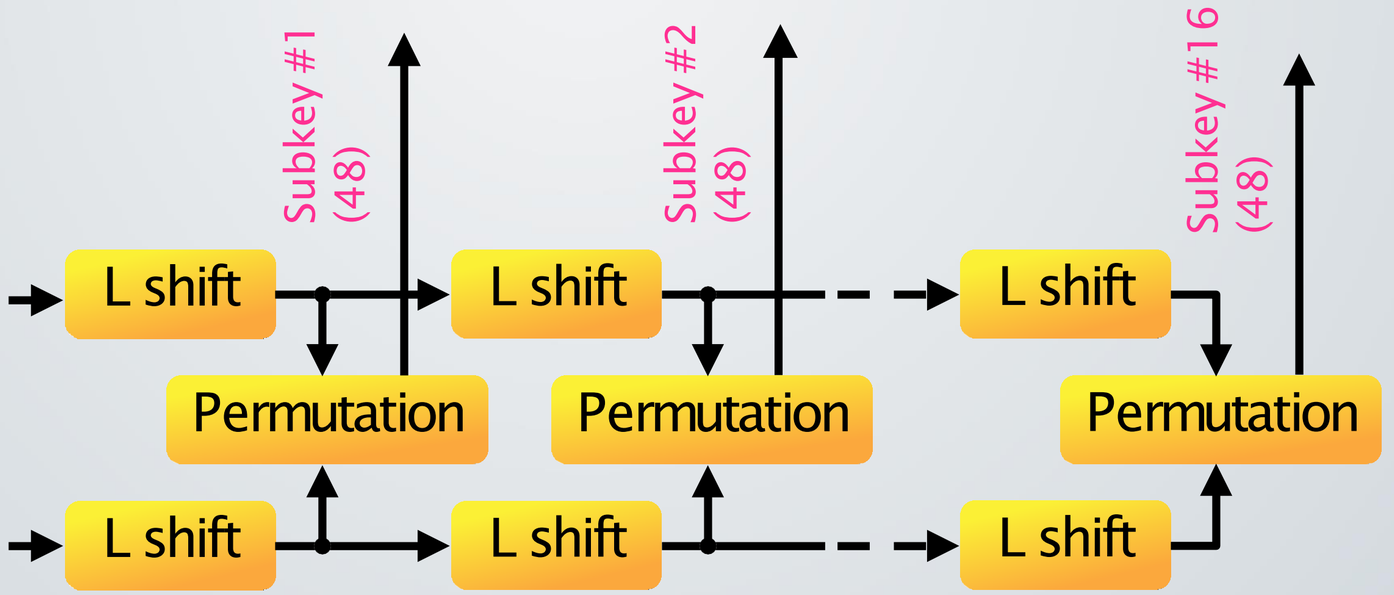


Dalla chiave in input vengono generate delle sottochiavi con questo metodo:

1. Dai 64 bit in input ne vengono selezionati 56 mediante una permutazione iniziale
2. I 56 bit vengono divisi in due parti

DES – Gestore della chiave

3. Entrambe le metà vengono fatte slittare verso sinistra di 1 o 2 bit
4. Tramite un'altra permutazione vengono scelti 48 bit da usare come sottochiave per quel ciclo



Il DES è l'archetipo della cifratura a blocchi

Algoritmo che prende in ingresso una stringa di lunghezza fissa di testo in chiaro e la trasforma, con una serie di operazioni complesse, in una stringa di testo cifrato della stessa lunghezza.
Non cifrano un bit (lettera) alla volta, cifrano un blocco di elementi contemporaneamente.

Crittoanalisi

Quanto è grande lo spazio delle chiavi?

La nostra chiave è lunga 56 bit (64, di cui 8 scartati all'inizio)

Per ognuno di questi bit abbiamo 2 possibilità: 0 oppure 1

Crittoanalisi

Quanto è grande lo spazio delle chiavi?

$$K^{\star} = |\Sigma|^{L} = 2^{56} \approx 7{,}2 \cdot 10^{16}$$

Possono sembrare tantissime, ma in realtà già a metà degli anni ‘70 esistevano computer in grado di trovare una chiave valida in un tempo ragionevole. Oggi si può forzare DES in poche ore con un attacco di forza bruta.

Crittoanalisi

Problema: spazio delle chiavi troppo piccolo.

Conseguenza: attacco brute force.

Soluzione? …cifriamo TRE volte. (sigh!)

Advanced Encryption Standard

Costruzione di AES

1997

Il NIST emette il bando per la costruzione di AES

1999

Richieste tre caratteristiche:
Sicurezza (chiave da 128+ bit)
Efficienza computazionale
Semplicità di implementazione

L'algoritmo scelto è stato sviluppato da due crittografi belgi, Joan Daemen e Vincent Rijmen, che lo hanno presentato con il nome di *Rijndael*, derivato dai loro nomi. *Rijndael*, in fiammingo, si pronuncia ['rɛinda:l] ("rèin-daal").

Input

State

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

Questo è un blocco di 128 bit preso dal messaggio in chiaro che deve essere cifrato.

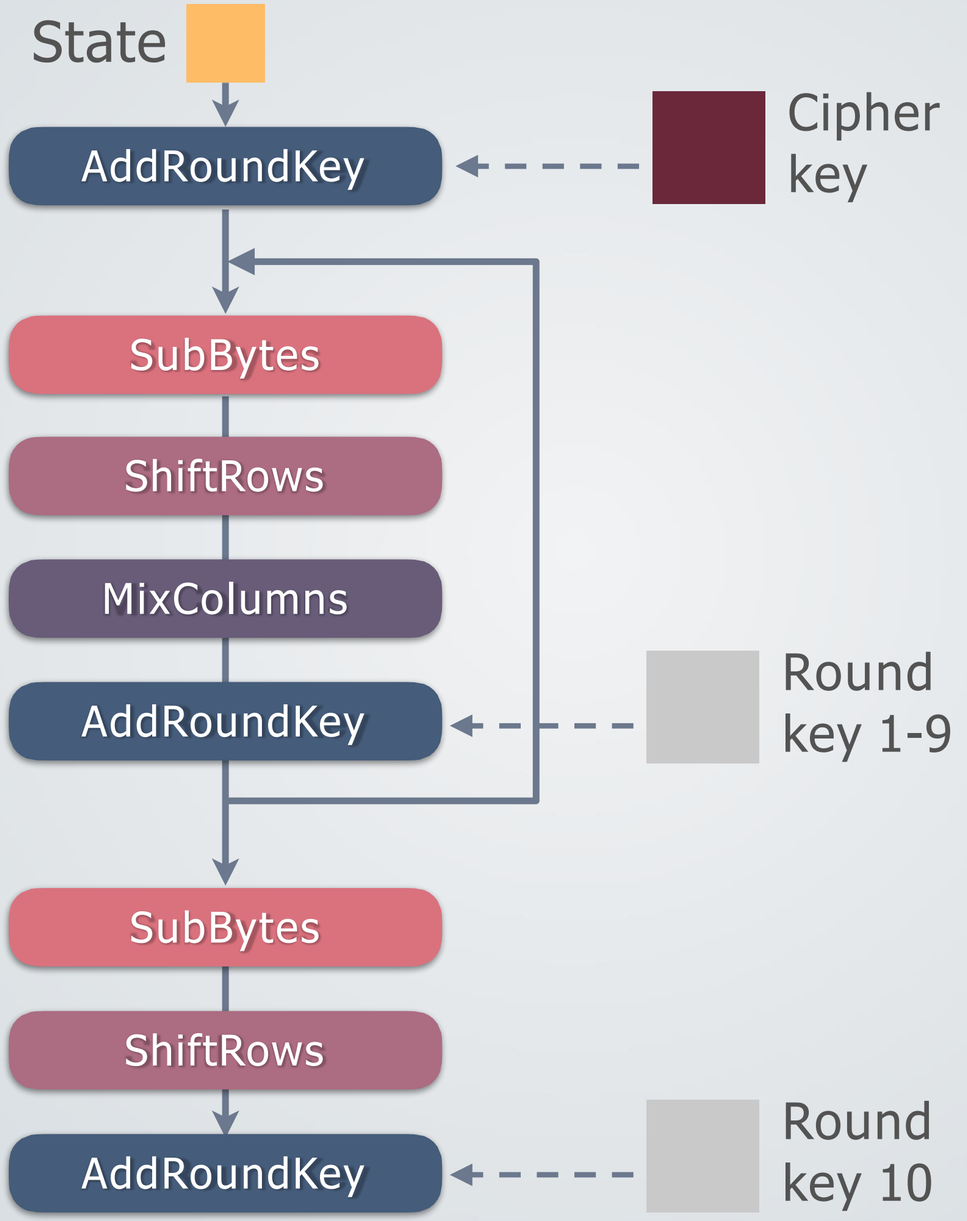
Cipher Key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

Notazione esadecimale (esempio):

32 = 0011 0010 (1 byte, 8 bit)

Processo di cifratura



Initial round

9/11/13 main rounds

final round

Processo di cifratura

Il procedimento di cifratura del blocco plaintext avviene usando 4 differenti trasformazioni

In particolare vengono usate quattro fasi, una di permutazione e tre di sostituzione:

- *Substitute bytes*: sostituzione del blocco byte per byte
- *Shift rows*: permutazione (''mescolamento'')
- *Mix columns*: una sostituzione colonna per colonna
- *Add round key*: una semplice operazione di XOR bit-a-bit del blocco corrente con una porzione della chiave di sessione

Le trasformazioni di AES

Vediamo ora le quattro trasformazioni di AES. Per ogni trasformazione vedremo come funziona intuitivamente l'algoritmo di crittografia ed infine come avviene la trasformazione della chiave.

- SubBytes
- ShiftRows
- MixColumns
- AddRoundKey

SubBytes

La trasformazione *Substitute byte* è una semplice ricerca su una tabella. AES definisce una matrice di dimensione 16x16 chiamata S-Box che contiene una regola di permutazione del blocco byte per byte

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

SubBytes

19

	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	[D4]	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD		A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34		E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

SubBytes

ShiftRows

MixColumns

AddRoundKey

ShiftRows

Questo è ciò che abbiamo ottenuto dalla matrice S-box

D4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

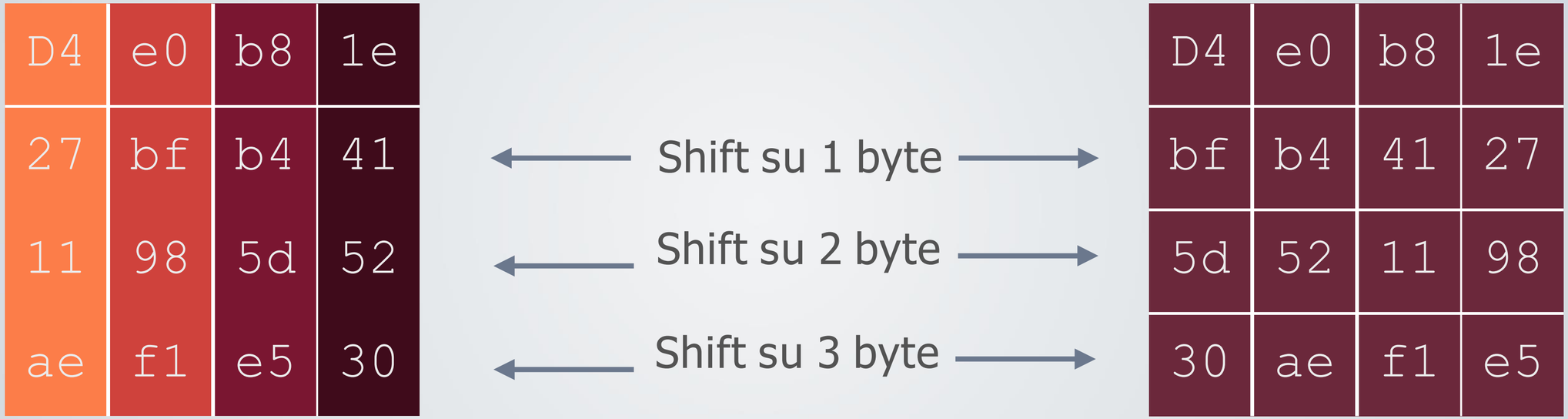
Shift su 1 byte

Shift su 2 byte

Shift su 3 byte

La prima riga resta invariata, la seconda si sposta verso sinistra di 1 posto, la terza di 2 posti, ecc

ShiftRows


D4 e0 b8 1e
27 bf b4 41
11 98 5d 52
ae f1 e5 30
Shift su 1 byte
Shift su 2 byte
Shift su 3 byte
D4 e0 b8 1e
bf b4 41 27
5d 52 11 98
30 ae f1 e5

SubBytes

ShiftRows

MixColumns

AddRoundKey

MixColumns

D4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

×

Questa trasformazione opera su una singola colonna. Ciascun byte di una colonna viene mappato in un nuovo valore che è una funzione dei quattro byte presenti nella colonna.

MixColumns

e0	b8	1e
b4	41	27
52	11	98
ae	f1	e5

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

×

D4
bf
5d
30

=

04
66
81
e5

(Su questo avete due opzioni:

1. Fidarvi
2. Studiare l'aritmetica dei campi di Galois

… io mi fiderei)

MixColumns

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

Fidandoci, questo è quello che otteniamo

SubBytes

ShiftRows

MixColumns

AddRoundKey

AddRoundKey

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

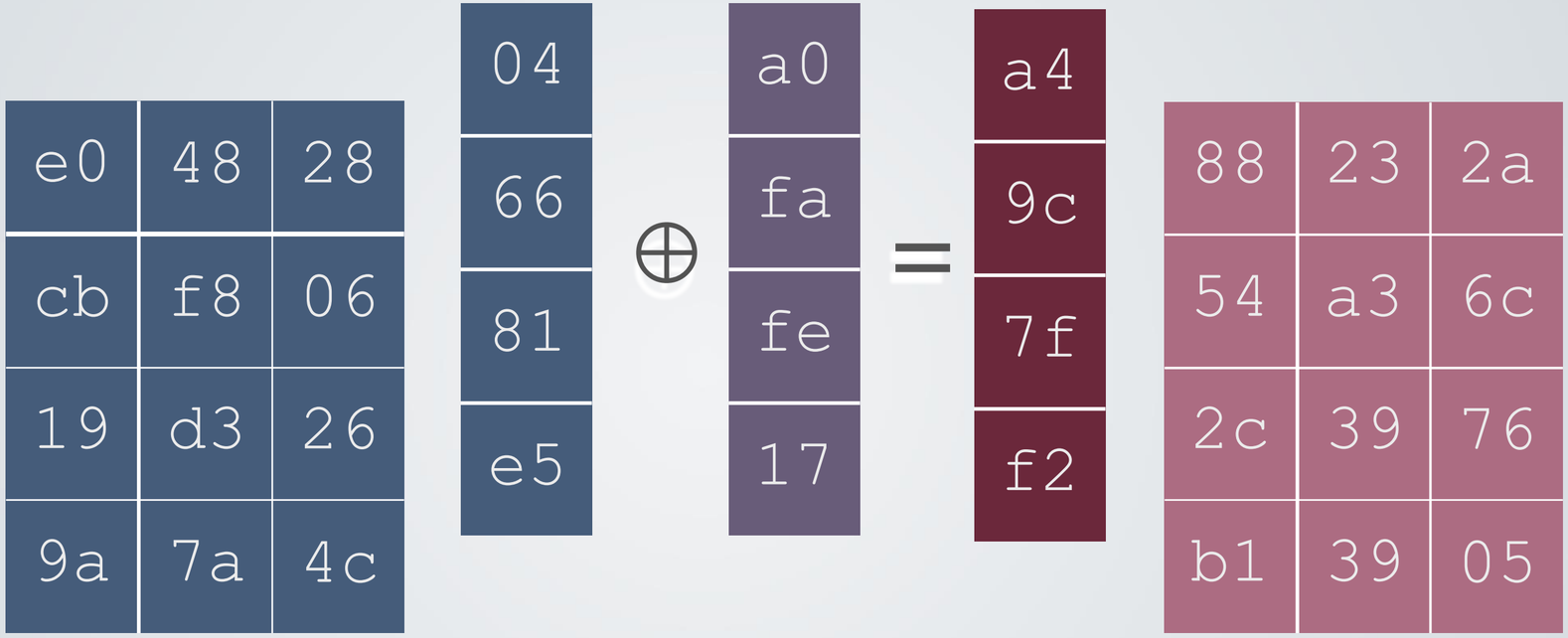
⊕

a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

Round key

Prodotta nel primo passo di generazione delle chiavi

AddRoundKey



AddRoundKey

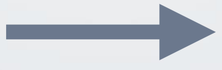
a4	68	6b	02
9c	9f	5b	6a
7f	35	ea	50
f2	2b	43	49

Rounds

I passi visti fin ora vengono applicati alla parola in input altre 9 volte (per un totale di 10 round). Il round finale non include la trasformazione MixColumns.

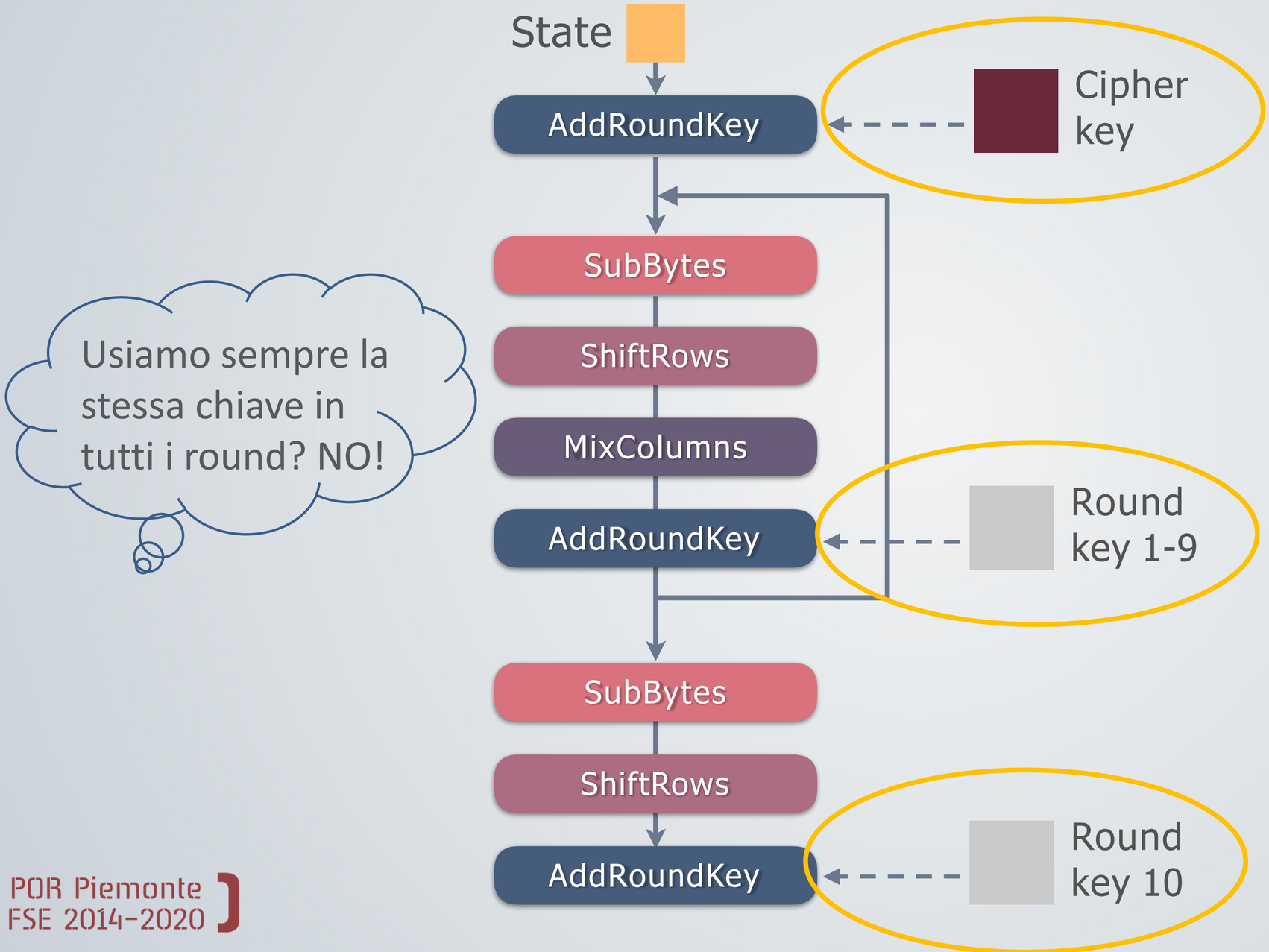
Nel nostro esempio, il plaintext produce il seguente ciphertext:

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34



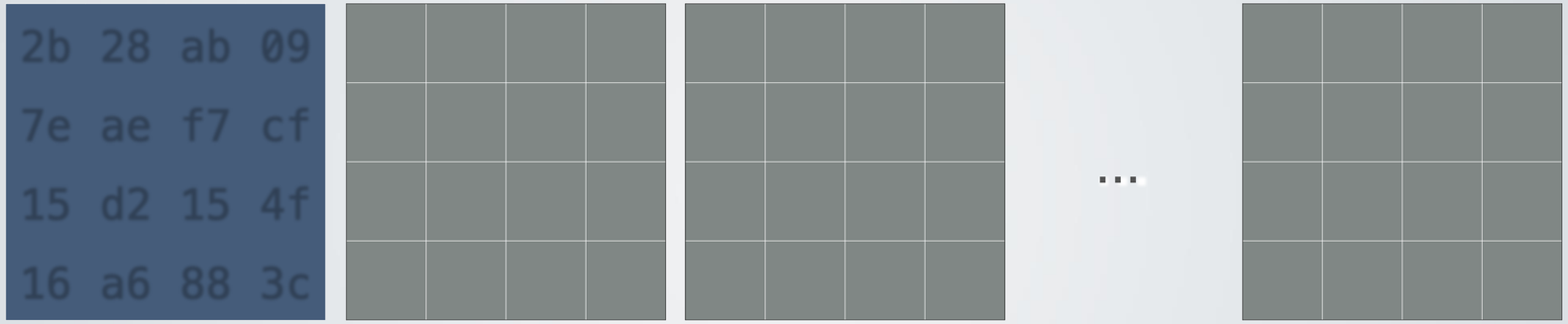
39	02	dc	19
25	dc	11	6a
84	09	85	0b
1d	fb	97	32

Processo di cifratura



Key schedule

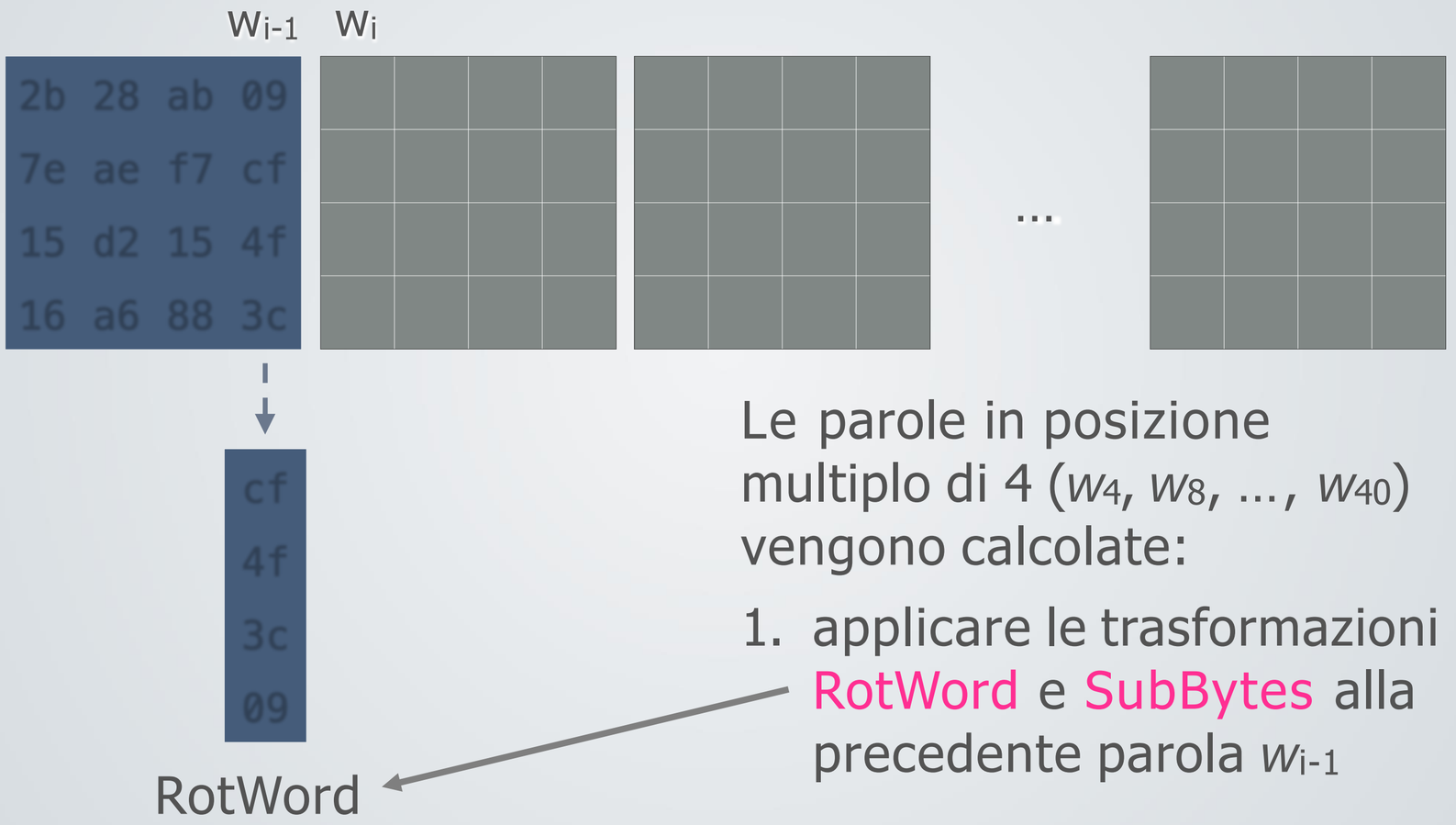
La chiave di cifratura subisce un procedimento di espansione per generare 11 chiavi parziali, usate nel'*initial round*, nei 9 *main round* e nel *final round*.



Cipher key

La chiave espansa può essere vista come un array di colonne (parole) di 32 bit, numerate da 0 a 43. La prima colonna è riempita con la chiave crittografica.

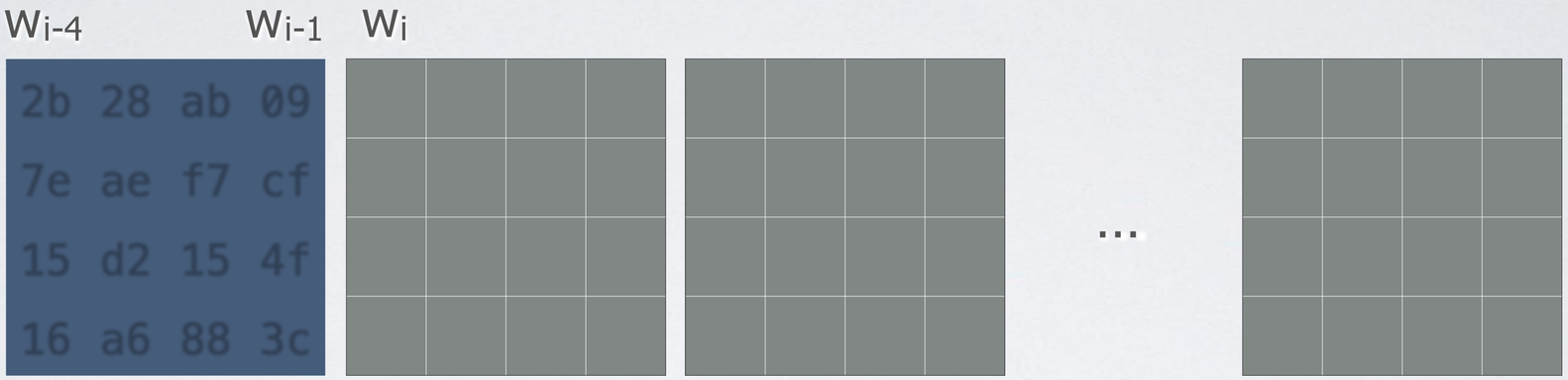
Key schedule

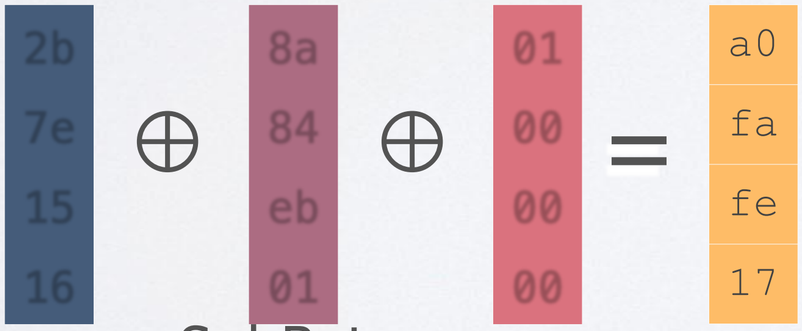


Le parole in posizione multiplo di 4 (w_4, w_8, …, w_{40}) vengono calcolate:

1. applicare le trasformazioni RotWord e SubBytes alla precedente parola w_{i-1}

Key schedule



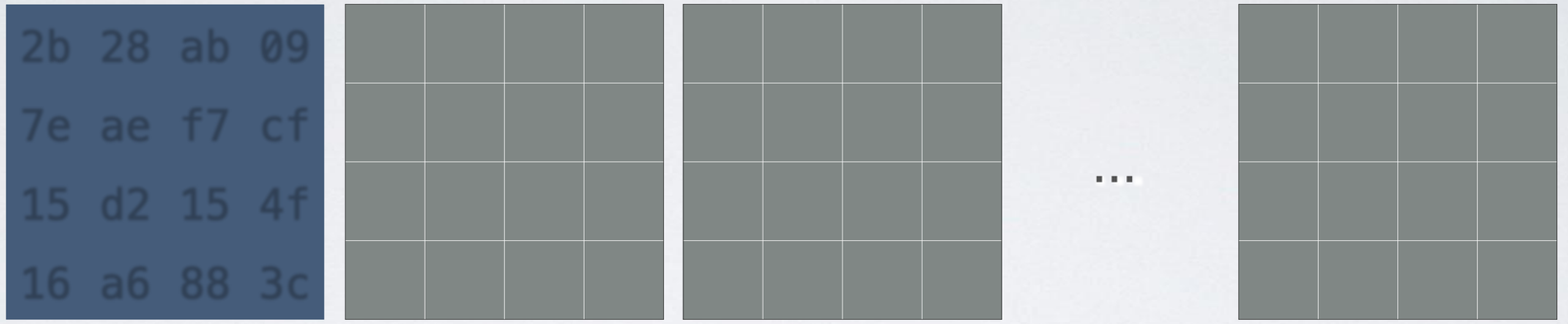


SubByte
con S-Box

Le parole in posizione multiplo di 4 (w_4, w_8, ..., w_{40}) vengono calcolate:

2. Si somma il risultato con la parola di 4 posizioni prima w_{i-4} più una costante Rcon

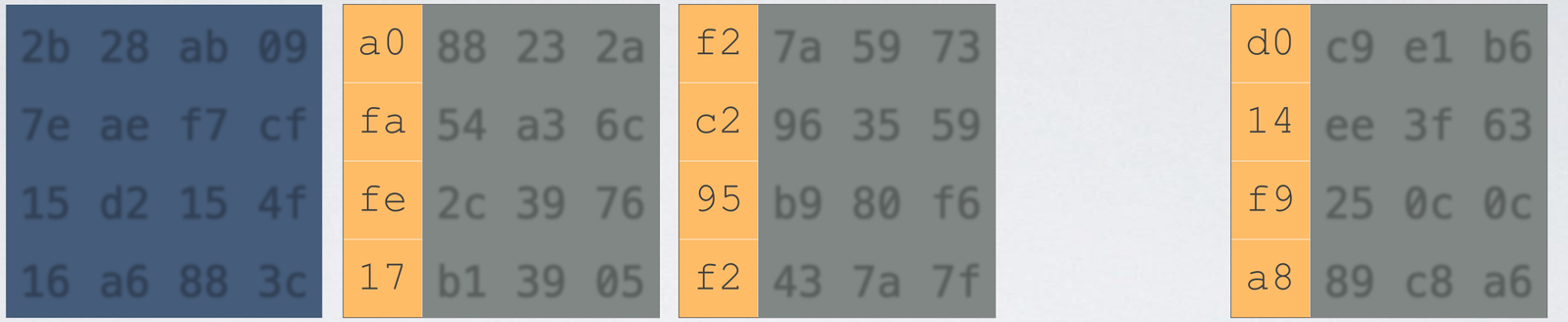
Key schedule



Cipher key Round key 1

a0
fa
fe
17

Key schedule



Il procedimento viene ripetuto fino a ottenere tutte le 11 chiavi che vengono usate nell'algoritmo

Sicurezza di AES

AES per garantire la sua sicurezza effettua:

- 10 round per la chiave a 128 bit,
- 12 round per la chiave a 192 bit,
- 14 round per la chiave a 256 bit.

I migliori attacchi sono riusciti a forzare l'AES con:

- 7 round e chiave di 128 bit,
- 8 round e chiave di 192 bit,
- 9 round e chiave di 256 bit.