

# Crittografia DES e AES

## Sicurezza informatica

3.1 ~ mag 2021



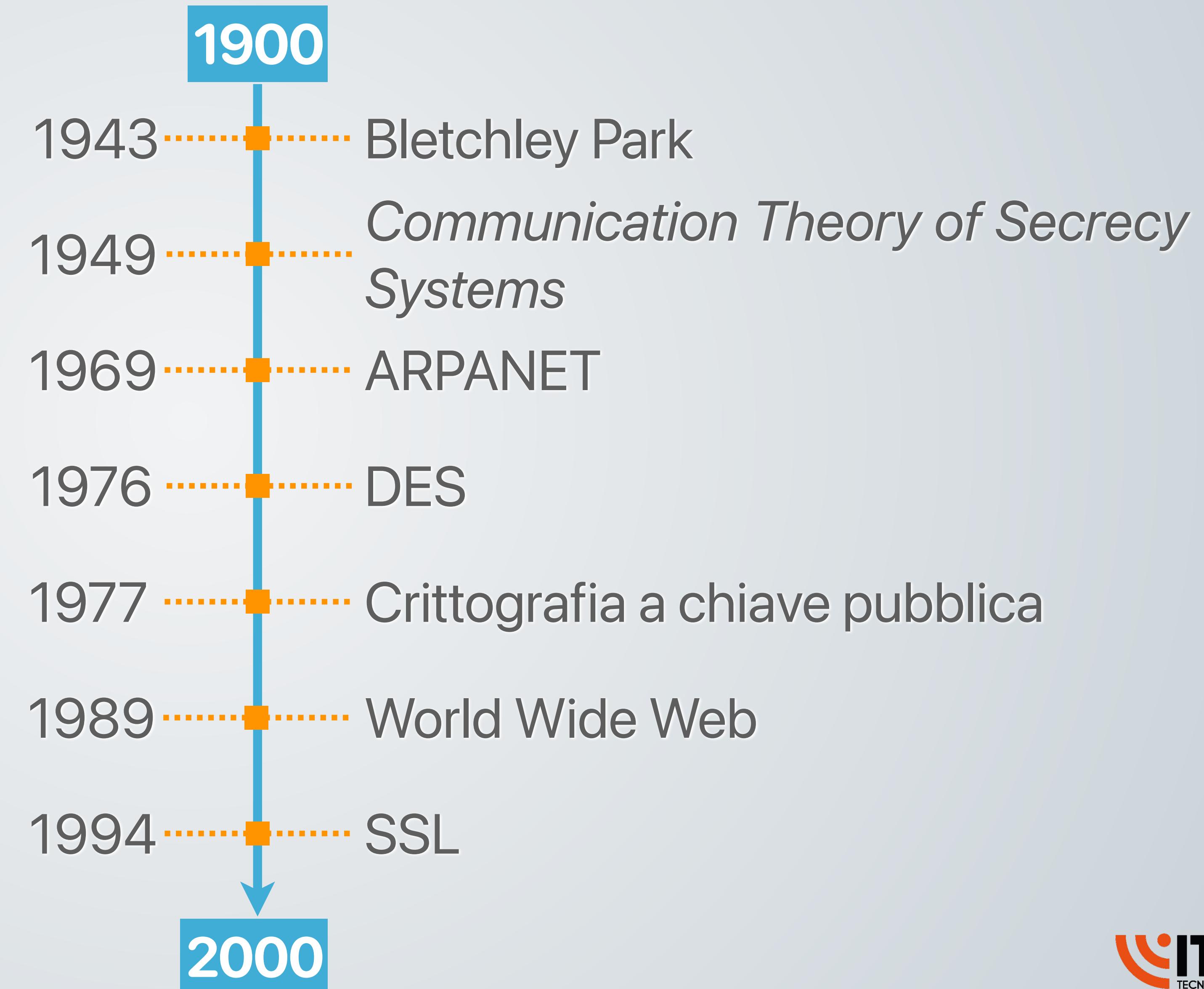
Prof. Marco Farina

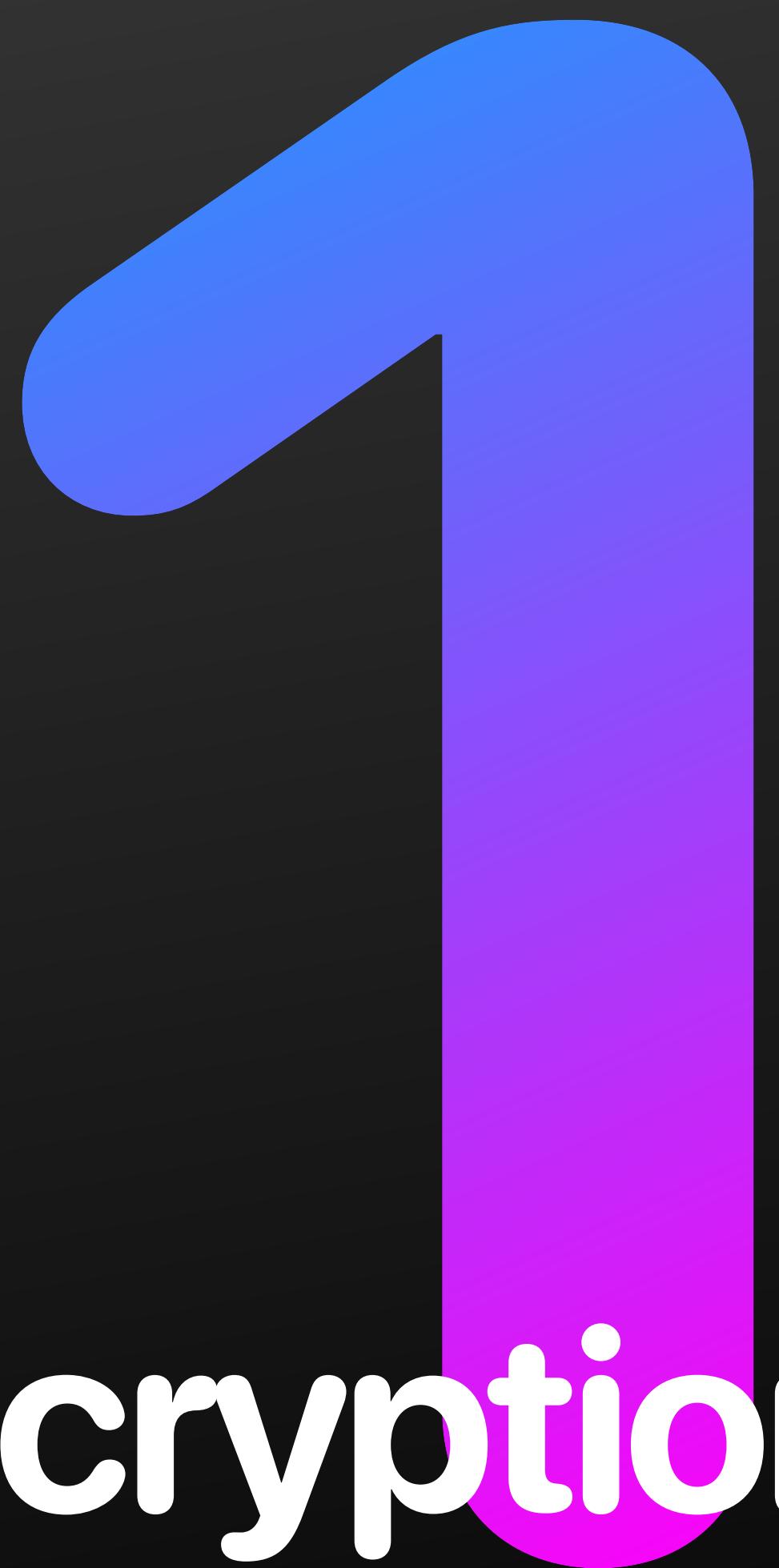
[marco.farina@its-ictpiemonte.it](mailto:marco.farina@its-ictpiemonte.it)

[t.me/marcofarina](https://t.me/marcofarina)

in collaborazione con:

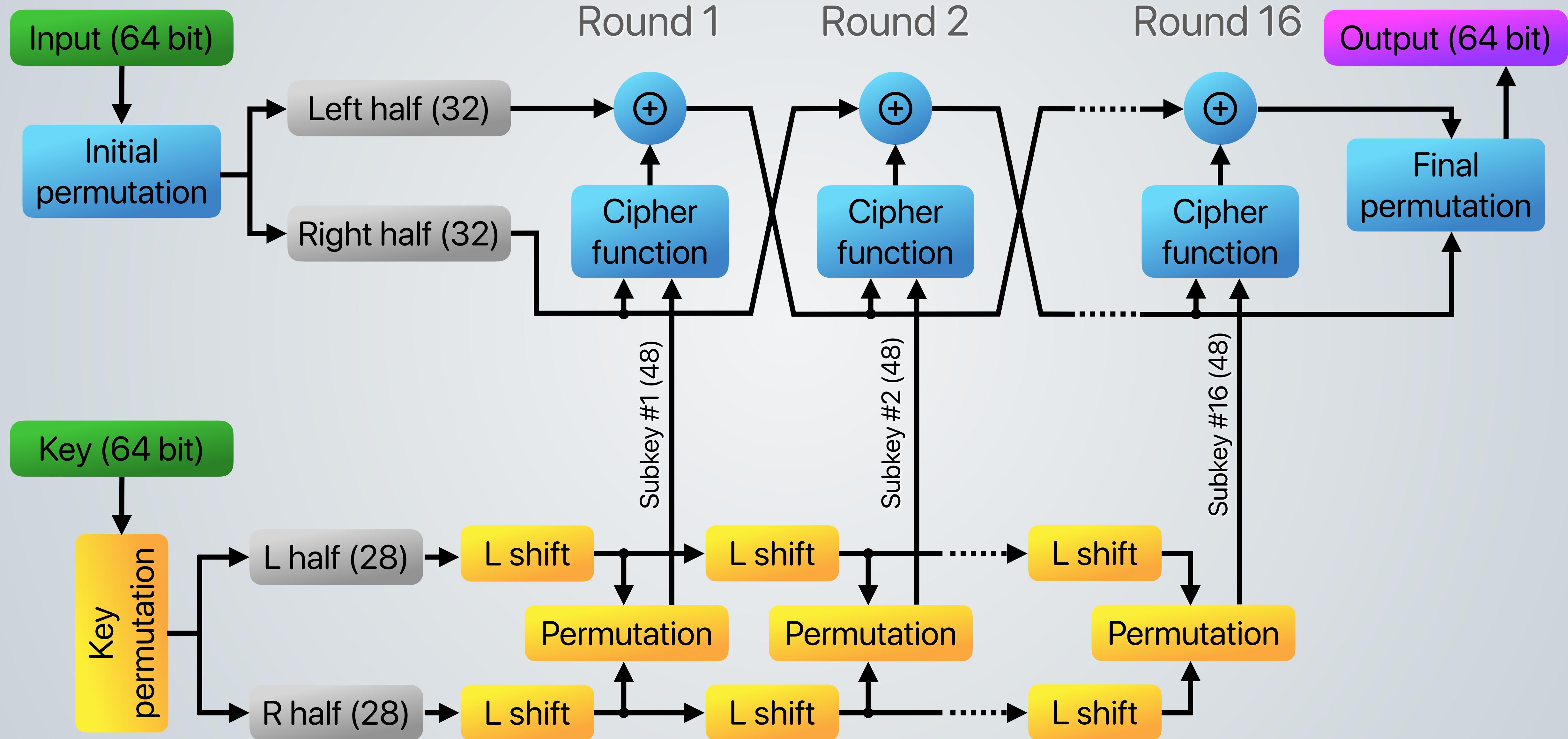
# Novecento: dalle lettere ai numeri





# Data Encryption Standard

# DES



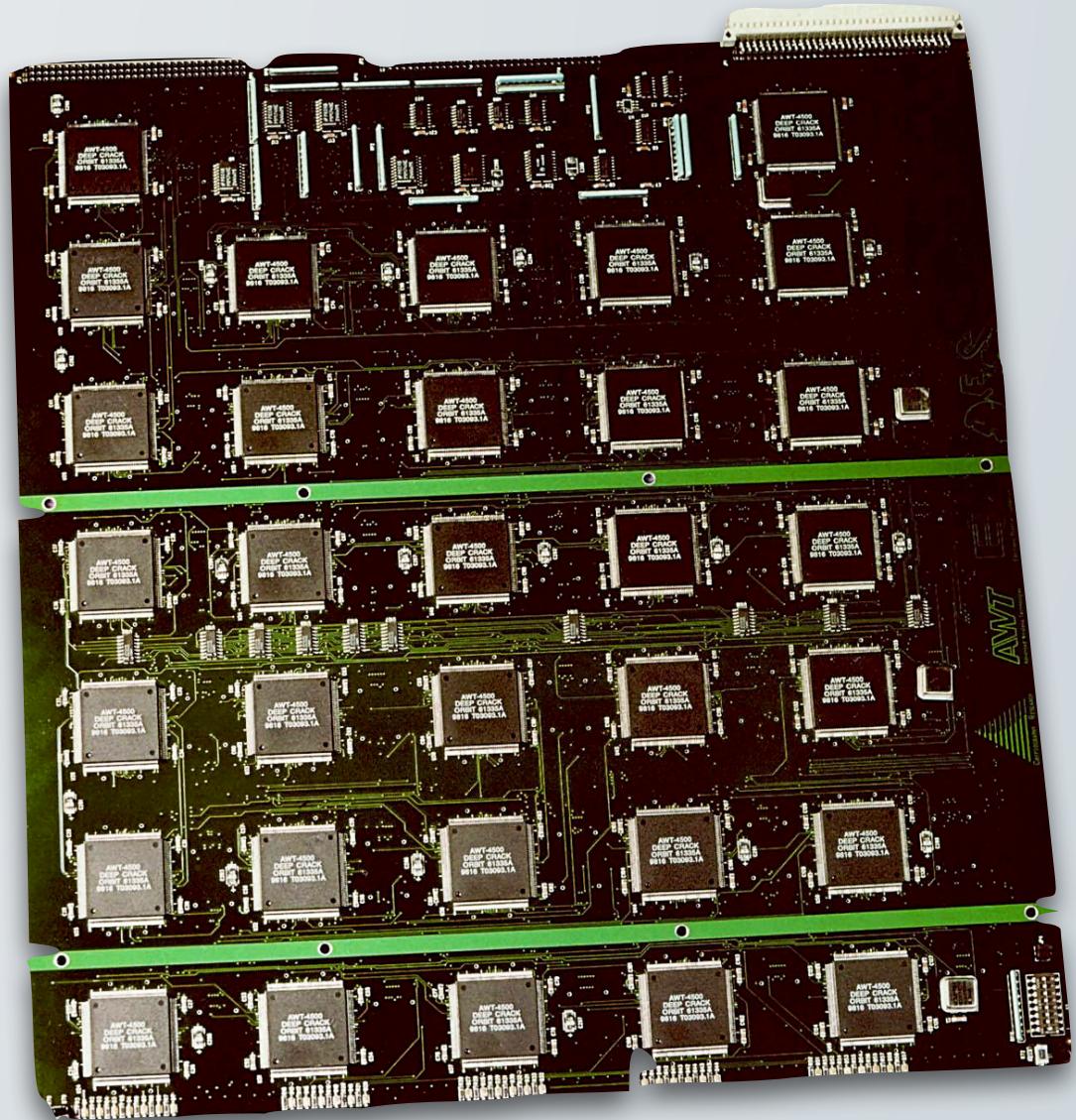
# Crittoanalisi

## Quanto è grande lo spazio delle chiavi?

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

### Deep Crack

- Chi: *distributed.net* & EFF
- Anno: 1999
- CPU: 1536
- Performance: 92 miliardi di chiavi al secondo
- Tempo: 22 ore e 15 minuti
- Costo: 200.000 \$



# Crittoanalisi

## Quanto è grande lo spazio delle chiavi?

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

**Problema:** spazio delle chiavi troppo piccolo.

**Conseguenza:** attacco brute force.

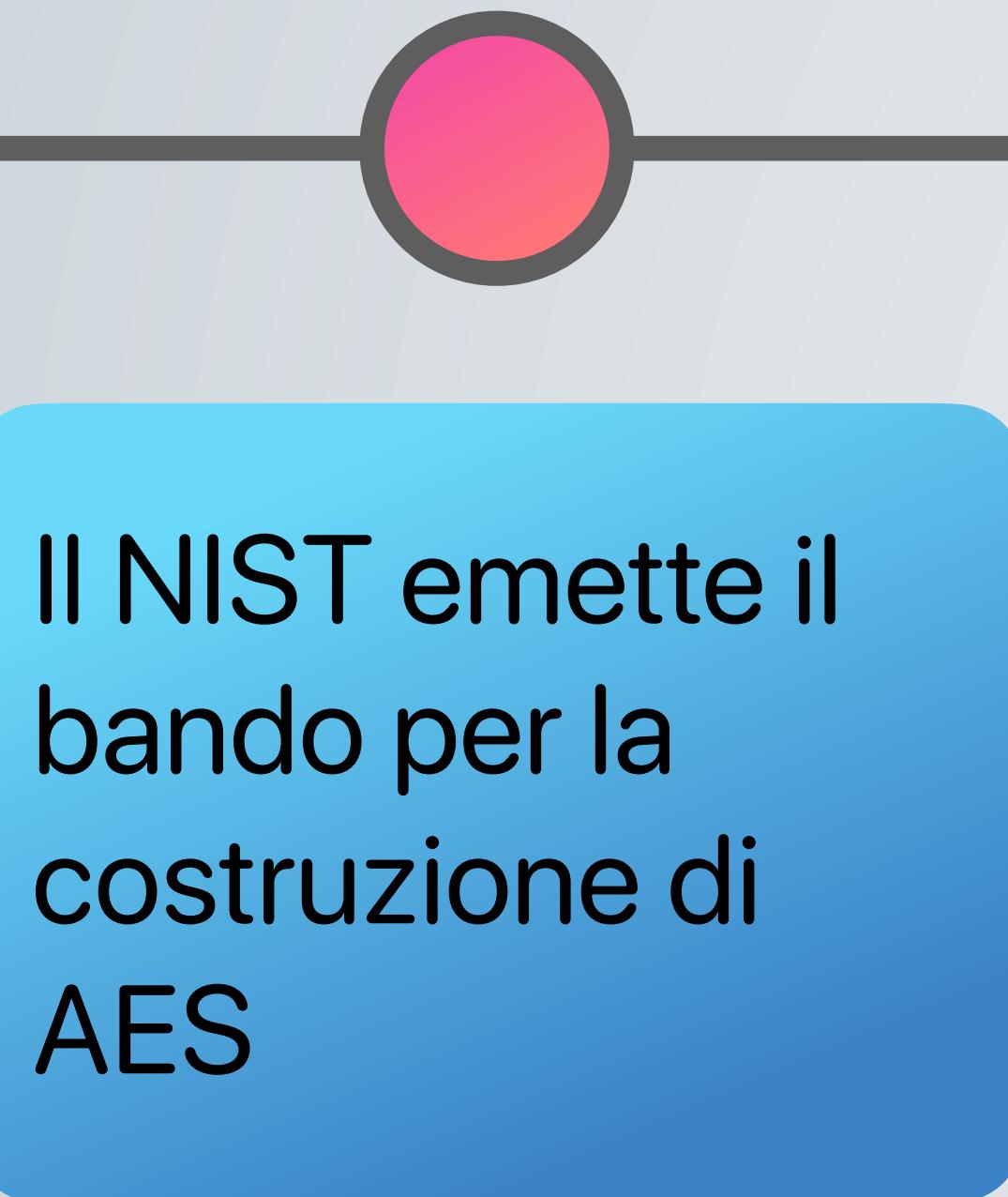
**Soluzione?** ...cifriamo TRE volte. (sigh!)



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 [rɛində:l] ("rèin-daal").

# Input

State

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

Cipher Key

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

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

Notazione esadecimale (esempio):

32 = 0011 0010 (1 byte)

# Input

State

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

Cipher Key

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



al processo di  
cifratura



all'espansione  
della chiave

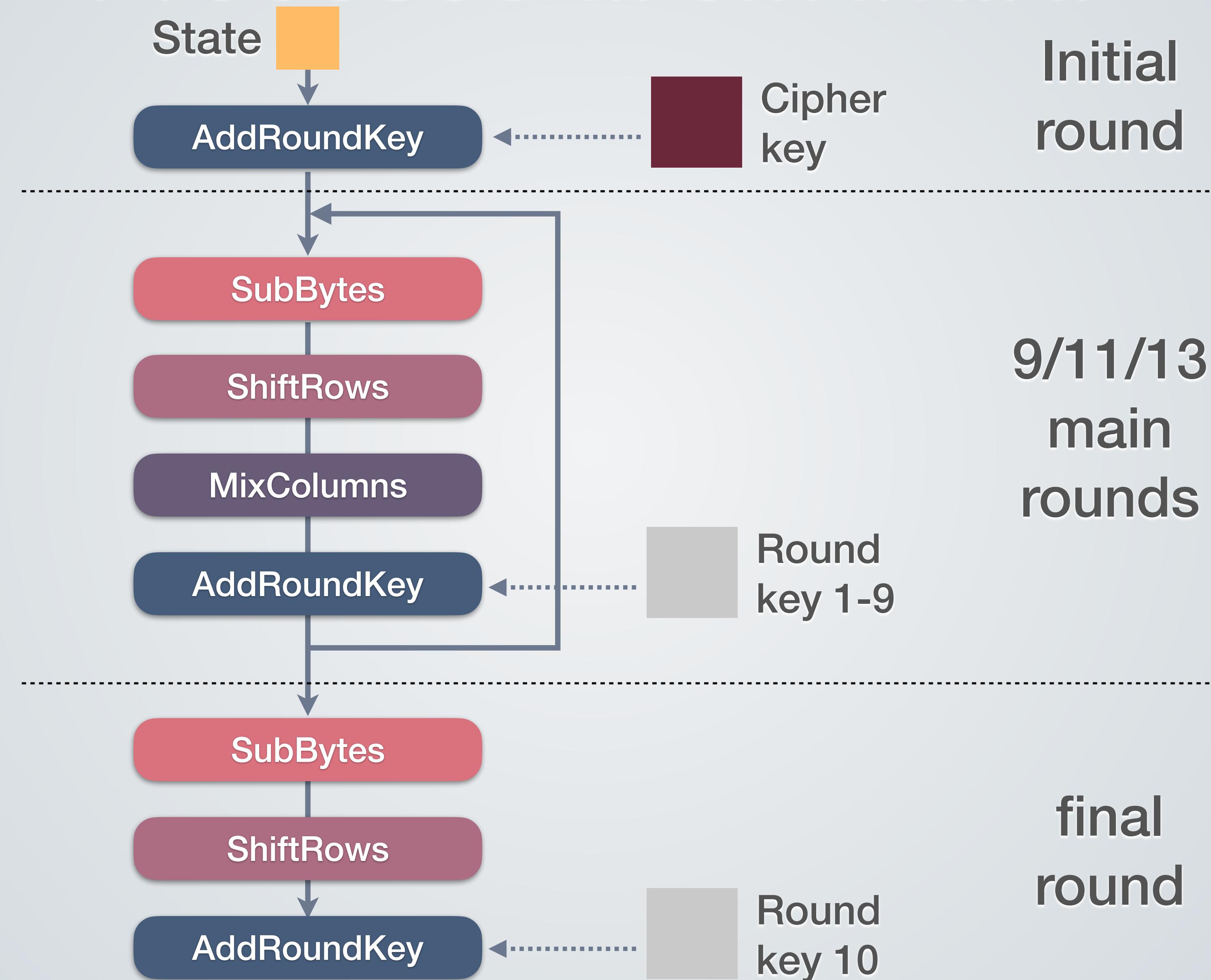
# Processo di cifratura

Il procedimento di cifratura del blocco plaintext avviene usando 4 differenti trasformazioni nel **round iniziale**, nei **9 round principali** e nel **round finale**.

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

- ***Substitute bytes***: usa una S-Box per svolgere una sostituzione del blocco byte per byte.
- ***Shift rows***: una semplice permutazione.
- ***Mix columns***: una sostituzione che utilizza l'aritmetica sui campi di Galois  $GF(2^8)$ .
- ***Add round key***: una semplice operazione di XOR bit-a-bit del blocco corrente con una porzione della chiave espansa.

# Processo di cifratura



# 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 permutazione di tutti i 256 valori a 8 bit.

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	97	2B	FE	D7	AB	76	
1	CA	82	C9	7D	FA	59	47	FO	A0	12	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	5	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

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

	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

ShiftRows

MixColumns

AddRoundKey

# 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

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

x

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. La trasformazione può essere definita dalla moltiplicazione in figura.

# MixColumns

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

$$\begin{array}{cc|cc} 02 & 03 & 01 & 01 \\ \hline 01 & 02 & 03 & 01 \\ \hline 01 & 01 & 02 & 03 \\ \hline 03 & 01 & 01 & 02 \end{array} \times \begin{array}{c} D4 \\ bf \\ 5d \\ 30 \end{array} = \begin{array}{c} 04 \\ 66 \\ 81 \\ e5 \end{array}$$

Nello standard la trasformazione viene definita considerando ciascuna colonna come un polinomio di quattro termini con coefficienti nel campo finito  $GF(2^8)$ . Ciascuna colonna viene moltiplicata modulo  $(x^4+1)$  per il polinomio fisso  $a(x)$  dato da:

$$a(x) = 3x^3 + x^2 + x + 2$$

# MixColumns

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

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

$$\begin{array}{|c|c|c|} \hline e0 & 48 & 28 \\ \hline cb & f8 & 06 \\ \hline 19 & d3 & 26 \\ \hline 9a & 7a & 4c \\ \hline \end{array} \quad \begin{array}{|c|} \hline 04 \\ \hline 66 \\ \hline 81 \\ \hline e5 \\ \hline \end{array} \oplus \begin{array}{|c|} \hline a0 \\ \hline fa \\ \hline fe \\ \hline 17 \\ \hline \end{array} = \begin{array}{|c|} \hline a4 \\ \hline 9c \\ \hline 7f \\ \hline f2 \\ \hline \end{array} \quad \begin{array}{|c|c|c|} \hline 88 & 23 & 2a \\ \hline 54 & a3 & 6c \\ \hline 2c & 39 & 76 \\ \hline b1 & 39 & 05 \\ \hline \end{array}$$

Round key

Prodotta nel primo  
passo di generazione  
delle chiavi

# 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. Il round finale non include la trasformazione MixColumns.

Nel nostro esempio, il plaintext produce il seguente ciphertext:

The diagram illustrates the transformation of a 4x4 plaintext matrix into a 4x4 ciphertext matrix. A large blue arrow points from the plaintext matrix on the left to the ciphertext matrix on the right.

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

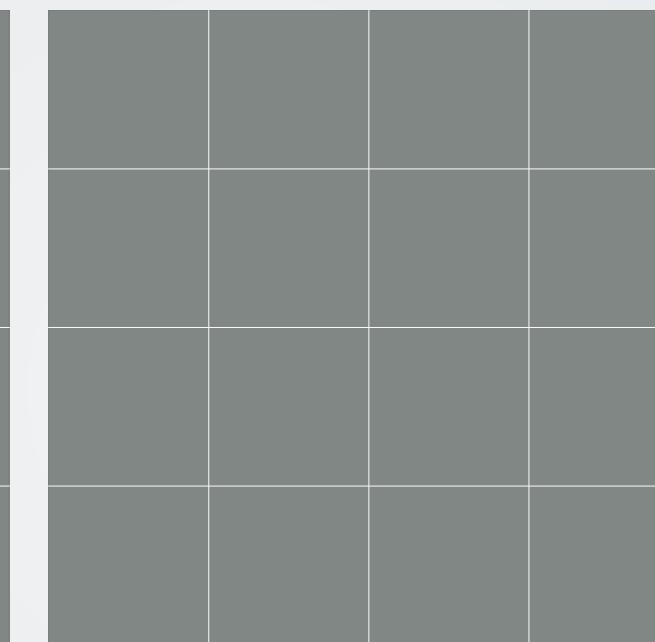
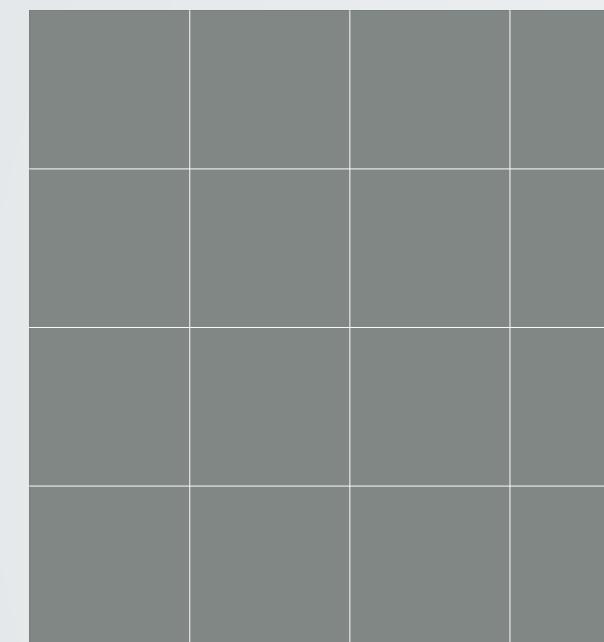
→

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

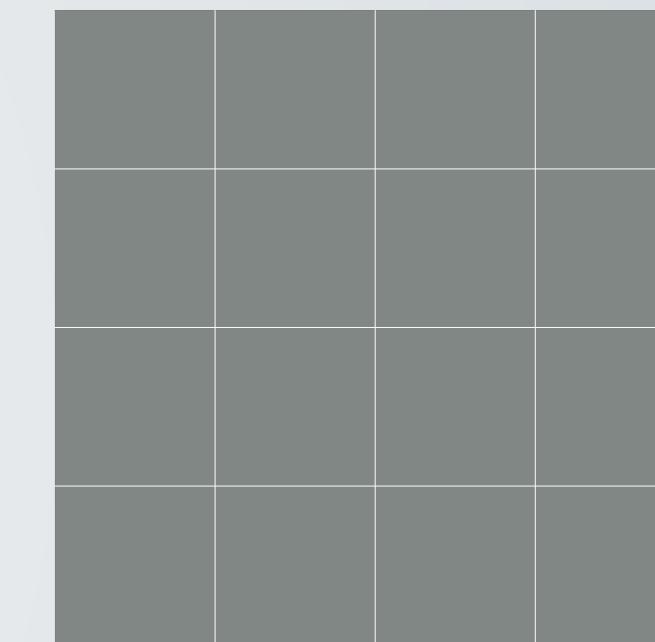
# Key schedule

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.

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



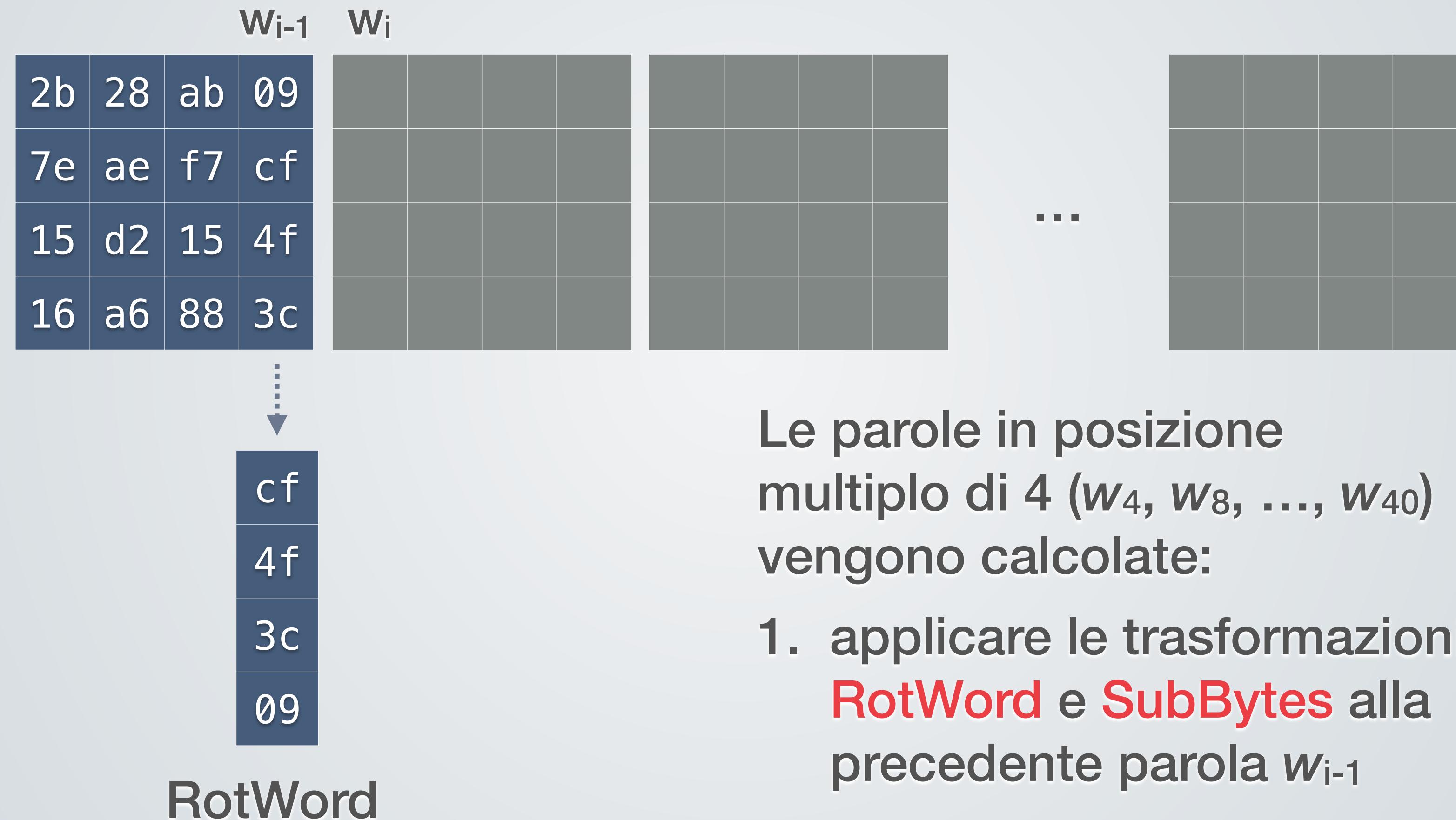
...



Cipher key

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

# Key schedule



# Key schedule

$w_{i-4}$	$w_{i-1}$	$w_i$	...
2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

$$\begin{array}{c} \downarrow \\ \begin{array}{ccccc} 2b & & 8a & & 01 \\ 7e & \oplus & 84 & \oplus & 00 \\ 15 & & eb & & 00 \\ 16 & & 01 & & 00 \end{array} = \begin{array}{c} a0 \\ fa \\ fe \\ 17 \end{array} \end{array}$$

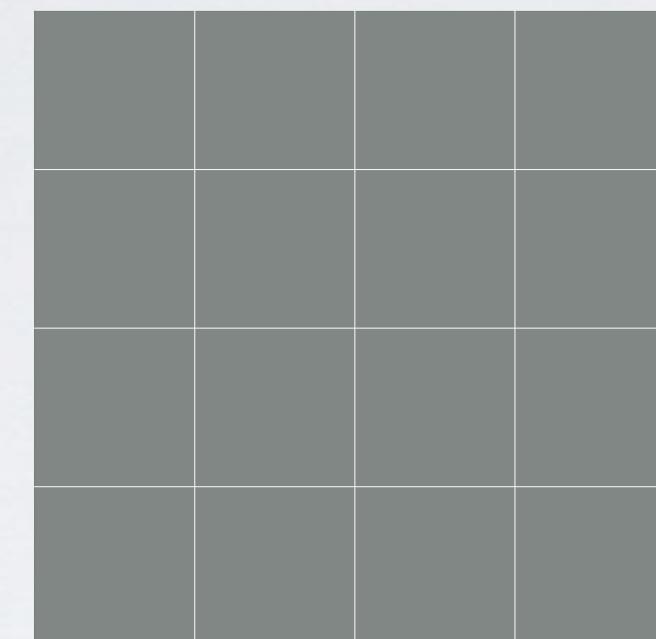
SubByte  
con S-Box

Le parole in posizione  
multiplo di 4 ( $w_4, w_8, \dots, w_{40}$ )  
vengono calcolate:  
2. Si somma (XOR) il risultato  
con la parola di 4 posizioni  
prima  $w_{i-4}$  più una  
costante **Rcon**

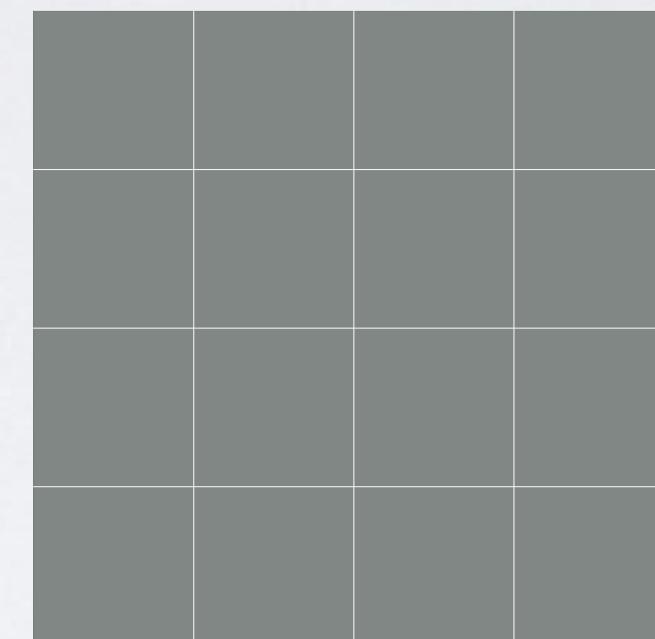
# Key schedule

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

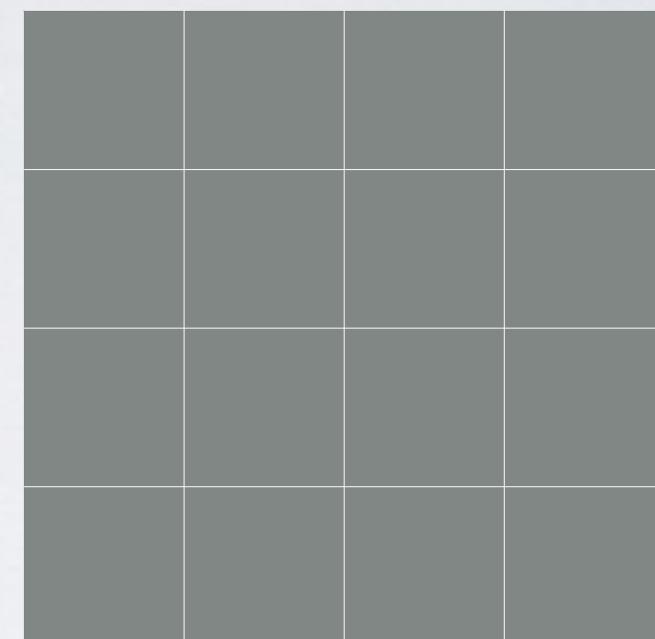
Cipher key



Round key 1



...



a0  
fa  
fe  
17

# Key schedule

2b	28	ab	09	a0	88	23	2a	f2	7a	59	73	d0	c9	e1	b6	
7e	ae	f7	cf	fa	54	a3	6c	c2	96	35	59	14	ee	3f	63	
15	d2	15	4f	fe	2c	39	76	95	b9	80	f6	...	f9	25	0c	0c
16	a6	88	3c	17	b1	39	05	f2	43	7a	7f	a8	89	c8	a6	

Cipher key      Round key 1      Round key 2      ...      Round key 10

Riassumendo, il procedimento di *key schedule* è:

- l'input è una parola di 32 bit e un numero di iterazione  $i$ . L'output è una parola di 32 bit;
- la parola in input viene copiata nell'output;
- si applica la funzione **Rotate** definita precedentemente per ruotare l'output di otto bit verso sinistra;
- si applica la **S-box di Rijndael** su tutti e quattro i singoli byte della parola di output;
- solo sul primo byte della parola di output, si applica l'or esclusivo tra il byte e **Rcon( $i$ )**.

# Sicurezza di AES

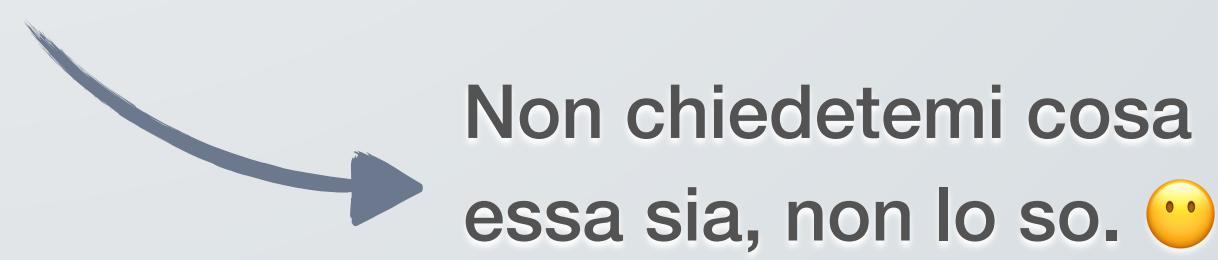
AES per garantire la sua sicurezza effettua:

- 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.

Attualmente, secondo i documenti segreti trafiguti da Edward Snowden, l'NSA sta facendo ricerche per capire se un attacco basato sulla **statistica tau** possa aiutare a rompere AES.



# Prestazioni

AES funziona su una grande varietà di dispositivi: dalle smart card a 8 bit fino ai super-computer.

Attualmente, sui processori Intel Core i3/i5/i7, AMD APU e le cpu che supportano il set di istruzioni AES-NI, la cifratura AES arriva ad avere un throughput di **700 MB/s per thread**.



# Learn by doing



La libreria openssl fornisce un'implementazione sicura dei principali cifrari. È bene provare a familiarizzare con alcuni dei comandi possibili.

Manuale eBook gratuito: <https://www.feistyduck.com/books/openssl-cookbook/>

Provate a cifrare dei messaggi scritti in un file txt con 3DES e con AES.