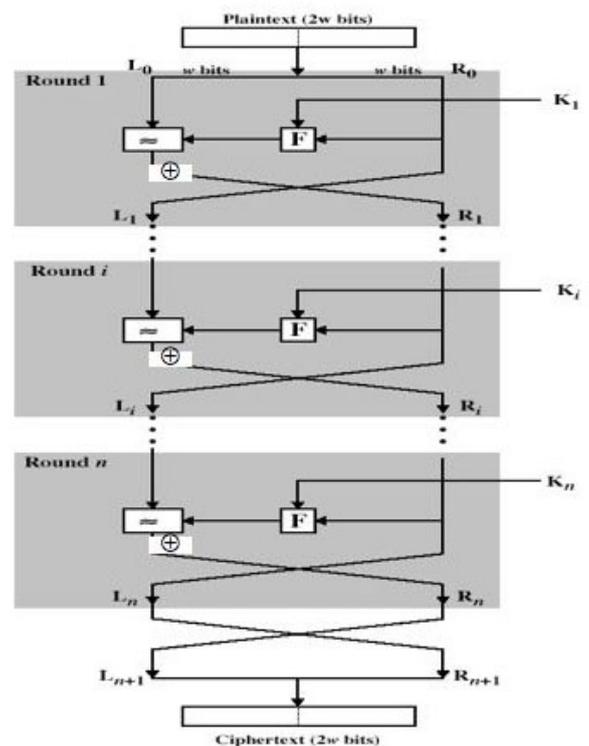


## cifrario di Feistel

- Horst Feistel (IBM) ha introdotto uno schema di cifrario nei primi anni 70.
- il DES è un cifrario di Feistel
- molti cifrari a blocchi sono cifrari di Feistel - per esempio i due finalisti AES MARS e twofish
- la cifratura e la decifratura funzionano essenzialmente allo stesso modo

In un cifrario di Feistel, i blocchi hanno lunghezza  $m = 2w$ : a ogni iterazione, ogni stato è diviso in due blocchi di lung.  $w$ ,  $(L_i, R_i)$ . Servono  $R$  chiavi di round  $(k_1, \dots, k_R)$ .

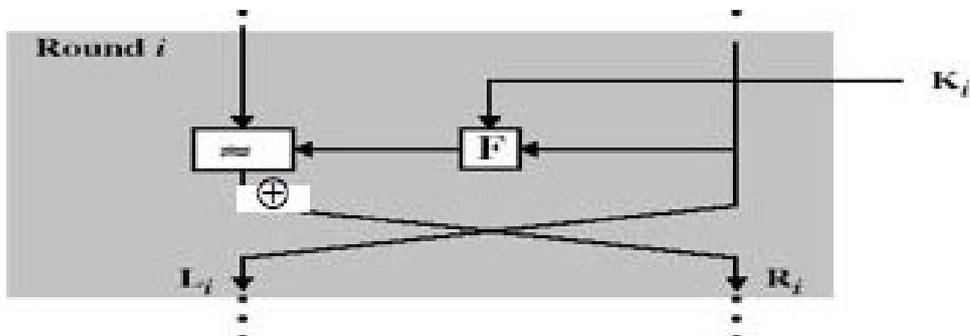
## cifrario di Feistel



## funzione di round

la funzione di round ha la seguente forma:

input:  $(L_{i-1}, R_{i-1}, k_i)$ , output  $(L_i, R_i)$



dove  $L_i = R_{i-1}$ ,

$R_i = L_{i-1} \oplus F(R_{i-1}, k_i)$ .

## cifrario di Feistel - decifratura

Notiamo che la  $F$  non deve necessariamente essere iniettiva: la funzione di round è comunque invertibile.

da

$$L_i = R_{i-1},$$

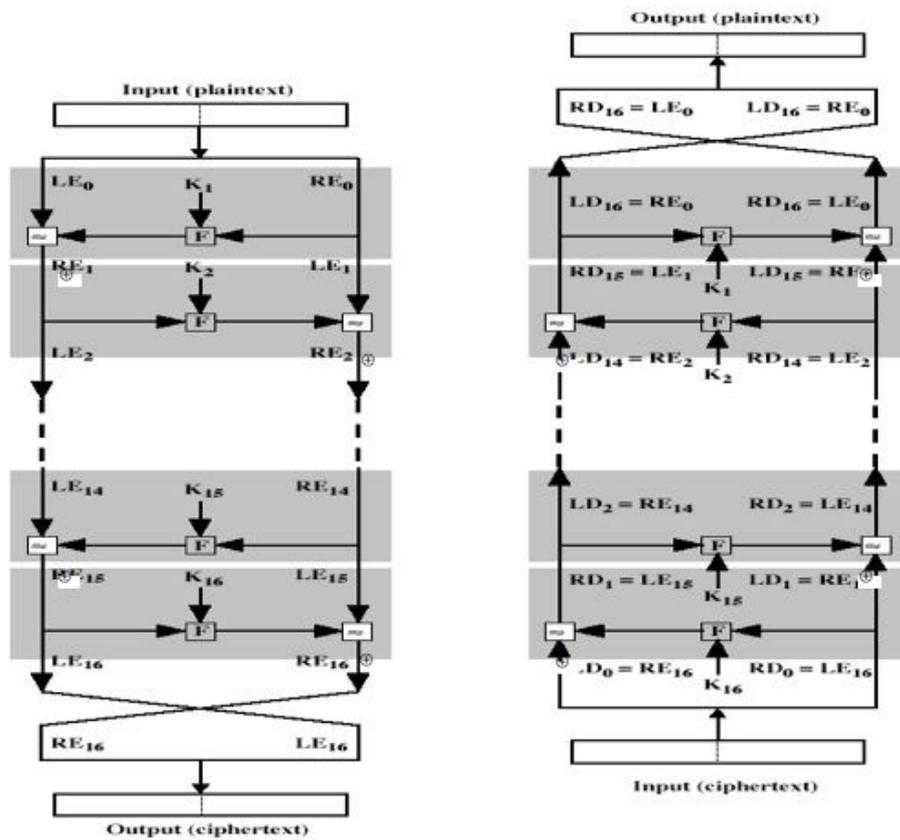
$$R_i = L_{i-1} \oplus F(R_{i-1}, k_i),$$

si ha

$$L_{i-1} = R_i \oplus F(L_i, k_i)$$

$$R_{i-1} = L_i$$

$$\text{infatti } R_i \oplus F(L_i, k_i) = L_{i-1} \oplus F(R_{i-1}, k_i) \oplus F(L_i, k_i) = L_{i-1}$$



**Dimensione del blocco** L' aumento della grandezza del blocco aumenta la sicurezza ma rallenta la velocità di cifratura e decifratura

**Dimensione della chiave** L' aumento della lunghezza della chiave aumenta la sicurezza e rende la ricerca esaustiva più difficile ma rallenta l' algoritmo

**Numero di round** L' aumento del numero di round aumenta la sicurezza ma rallenta l' algoritmo

**Funzione F** L' aumento di complessità rende l' analisi più difficile ma rallenta l' algoritmo

# Data Encryption Standard

Storia:

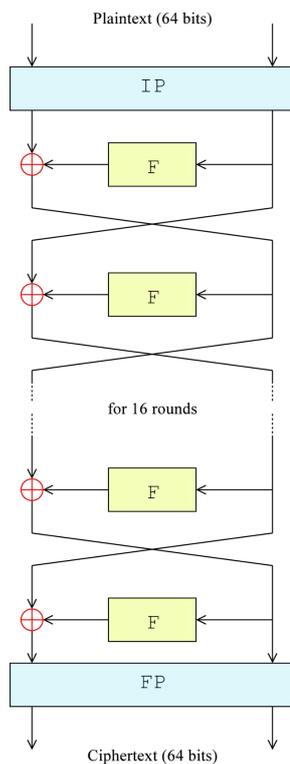
- pubblicato nel 1977 dal National Bureau of Standards (ora NIST National Institute of Standards and Technology) per uso commerciale e applicazioni del governo degli USA
- 1973-75 sviluppo del DES, dal LUCIFER (70) dell'IBM creato da Feistel e altri
- il DES sviluppato dall'IBM (Feistel, Coopersmith, Tuchman e altri)
- testato (e modificato) dall'NSA
- 75-77 il DES va al vaglio dei crittografi
- nel 1999 il NIST ha pubblicato una nuova versione chiamata triple DES (3DES) o TDEA – comunque il DES è stato molto più longevo del previsto (si credeva 10-15 anni)

Molte controversie sulla sicurezza dell'algoritmo:

si è creduto che l'NSA abbia interferito con la progettazione.

La chiave è stata portata dai 128 bit del LUCIFER a 56 bit.

Si crede(va) che ci fosse una trapdoor nella progettazione delle S-box.



Il DES consiste di:

- Una permutazione iniziale  $IP$  dei 64 bit
- 16 round identici “alla Feistel”
- Una permutazione finale, che è l’inversa di quella iniziale:  $FP = IP^{-1}$

## permutazioni

Le due permutazioni, iniziale e finale, sono l’unica cosa che rende il DES diverso da un cifrario di Feistel.

Non servono a aumentare la sicurezza, ma sono legate al processo di caricamento del plaintext nei chip.

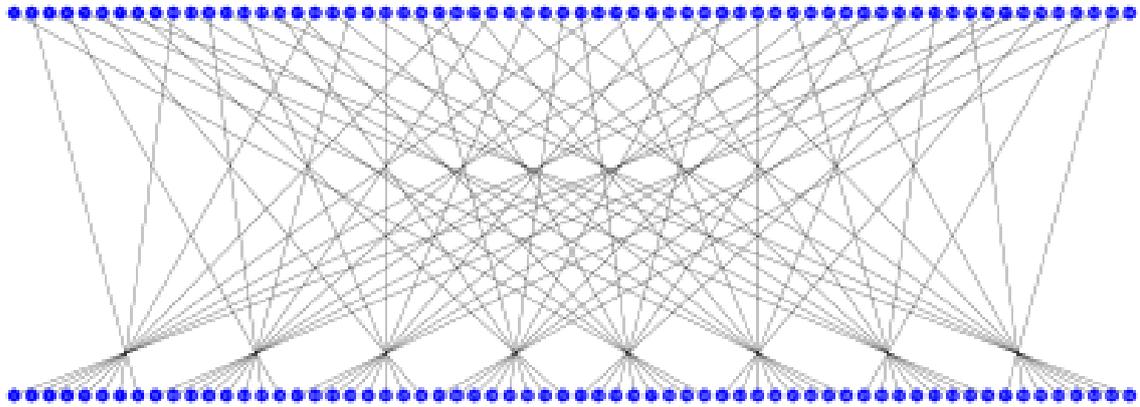
IP

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

cioè:

il primo bit in output è il 58simo dell’input;  
 il secondo bit in output è il 50simo dell’input. . .

$58 \rightarrow 1, 50 \rightarrow 2 \dots$

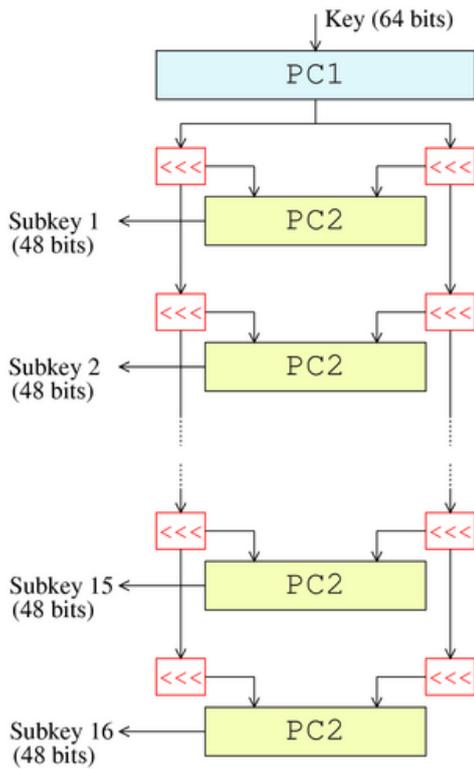


## chiave

La chiave: 64 bit, di cui 8 sono un controllo di parità. La chiave ha solo 56 bit.

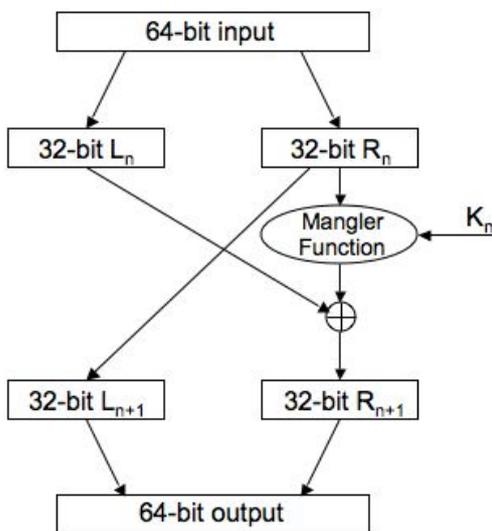
Dalla chiave  $k$  ottengo 16 chiavi di round  $(k_1, \dots, k_{16})$  di 48 bit ciascuna.

## key schedule

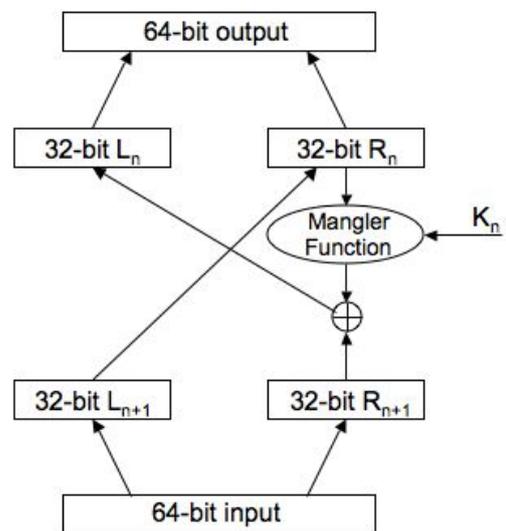


- produce le 16 chiavi di round (a 48 bit)
1. permutazione iniziale  $PC_1$  della chiave
  2. divisione dei 56 bit in 2 metà a 28 bit
  3. a ogni round
    - 3.1. Left shift ogni metà (28bit) separatamente di 1 o 2 posti (questo sarà l'input del prossimo round)
    - 3.2. riunione delle due metà, e
- permutazione/contrazione:  $PC_2$  ha 56 bit in input, output a 48 bit

## DES round



**Encryption**



**Decryption**

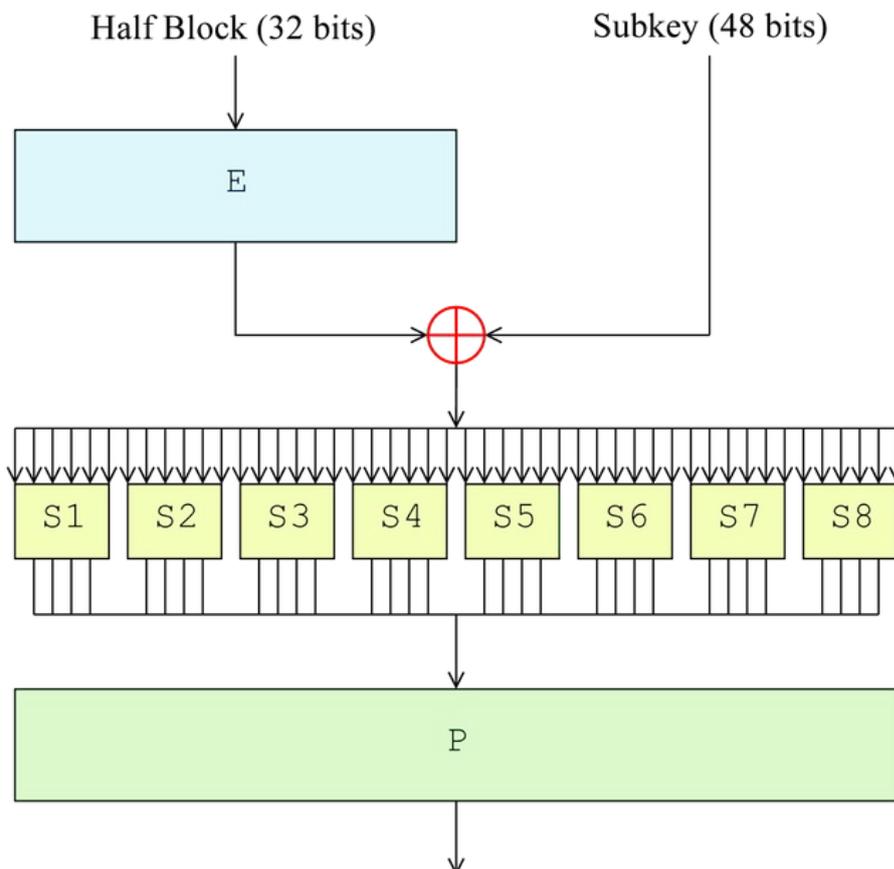
## funzione $F$ , o mangler

Dettagli della funzione  $F$ :

al round  $i$

$$F(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))$$

- 1 Prende i 48 bit della chiave  $k_i$ .
- 2 Espande i 32 bit di destra,  $R_{i-1}$ , a 48 bit
- 3 Esegue lo XOR della chiave e dei 48 bit e invia il risultato in input alle S-Box
- 4 Le S-box sono tabelle di sostituzione e contrazione predefinite. In output danno 32 bit
- 5 Questi 32 bit vengono permutati

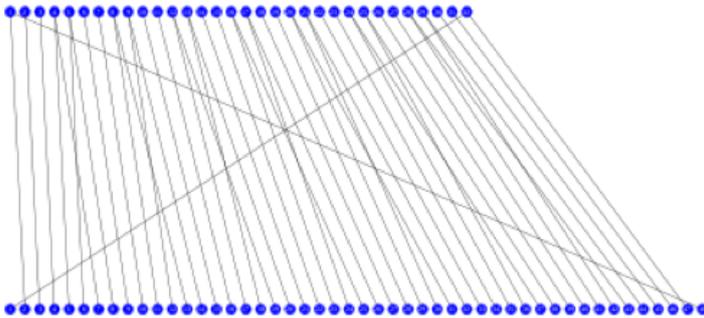


# espansione E

E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

1 → 2, 1 → 32  
2 → 3 ...



$S_1$

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

$S_2$

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

$S_3$

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

$S_4$

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

$S_5$

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

$S_6$

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

$S_7$

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

$S_8$

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

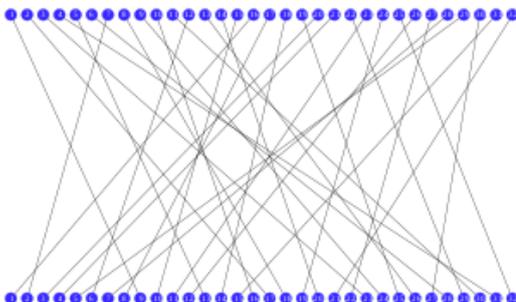
S <sub>5</sub>															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

Dato un input di 6 bit, i 4 bit di output si trovano scegliendo la riga usando i due bit esterni, la colonna usando i 4 bit interni.  
 Esempio: in 011011, i bit esterni sono 01 e quelli interni sono 1101.  
 Righe e colonne sono numerate partendo da 0; l'output corrisponde al valore nella seconda riga, 14sima colonna, cioè 9.  
 Quindi 011011 → 1001

## permutazione P

P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

16 → 1, 7 → 2  
 diffusione



## progettazione delle S-box

- ① 6 bit in entrata, 4 in uscita (il massimo possibile per i chip del 74)
- ② gli output NON sono una combinazione lineare degli input (né approssimabili da una funzione lineare)
- ③ le righe sono permutazioni di  $\{0, 1, \dots, 15\}$
- ④ cambiando un bit nell'input, cambiano almeno due bit in output
- ⑤ ogni bit dell'output dipende da tutti i bit in input

Il DES ha un forte *effetto valanga*: una proprietà desiderabile per un algoritmo di cifratura è che un cambiamento di un solo bit nell'input o nella chiave comporti una modifica di almeno metà dei bit di uscita. Tentare di indovinare la chiave diventa impossibile.

Il DES può essere efficientemente implementato in hardware  
Relativamente lento se implementato in software  
non era questa l'intenzione!

## crittoanalisi del DES

Di fatto, il modo più efficiente di violare il DES è un attacco a forza bruta: provare ogni possibile chiave, una dopo l'altra.

Ci sono due attacchi "a livello teorico"

**crittoanalisi differenziale:** "scoperta" da Biham e Shamir (ca 89).

Il DES resiste molto bene alla CD: la tecnica era nota all'IBM e/o all'NSA al momento della progettazione, ed è stata tenuta segreta.

Per violare il DES con la CD, servono  $2^{47}$  chosen plaintext.

**crittoanalisi lineare:** scoperta da Matsui nel 93, migliorata via via.

Servono non meno di  $2^{40}$  chosen plaintext. Non ci sono prove che il DES sia stato sviluppato per resistere a questo tipo di attacco.

## attacchi a forza bruta

Lo spazio delle chiavi del DES ha grandezza  $2^{56}$ . Già al momento dell'introduzione era considerato inadeguato.

Proposte per una macchina DES-cracker

1977: Diffie-Hellmann propongono una macchina che può trovare la chiave in un giorno. Costo stimato \$ 20.000.000

1993: Wiener, 7 ore, \$ 1.000.000

(Probabilmente, mai realizzate)

Nel 97 , la RSA Security ha sponsorizzato una gara, offrendo \$ 10.000 ai primi a decrittare un messaggio cifrato con il DES. La gara è stata vinta dal progetto DESCHALL, usando il calcolo distribuito.

Nel 98, la EFF (Electronic Frontier Foundation) ha costruito una macchina al costo di circa \$ 250.000. La macchina ha trovato una chiave in 2 giorni circa.

