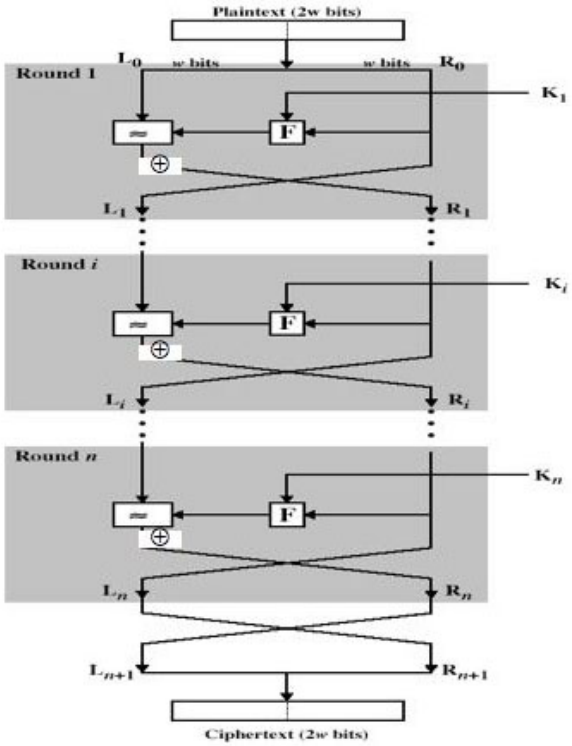
cifrario di Feistel

- 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

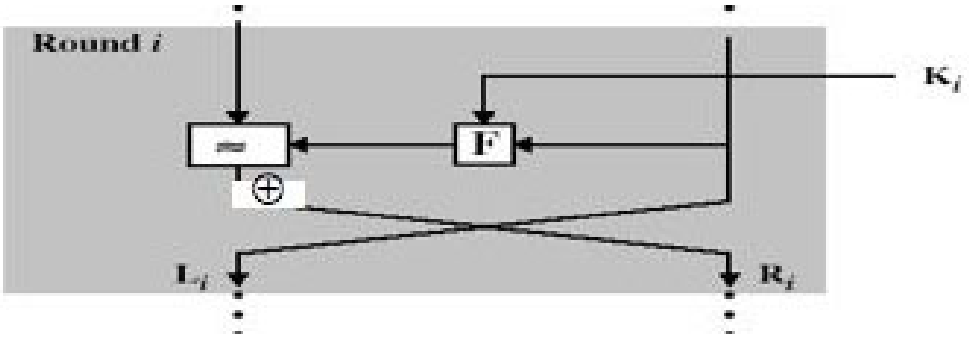
cifrario di Feistel

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 $(k_1, \ldots, k_R)$.



funzione di round

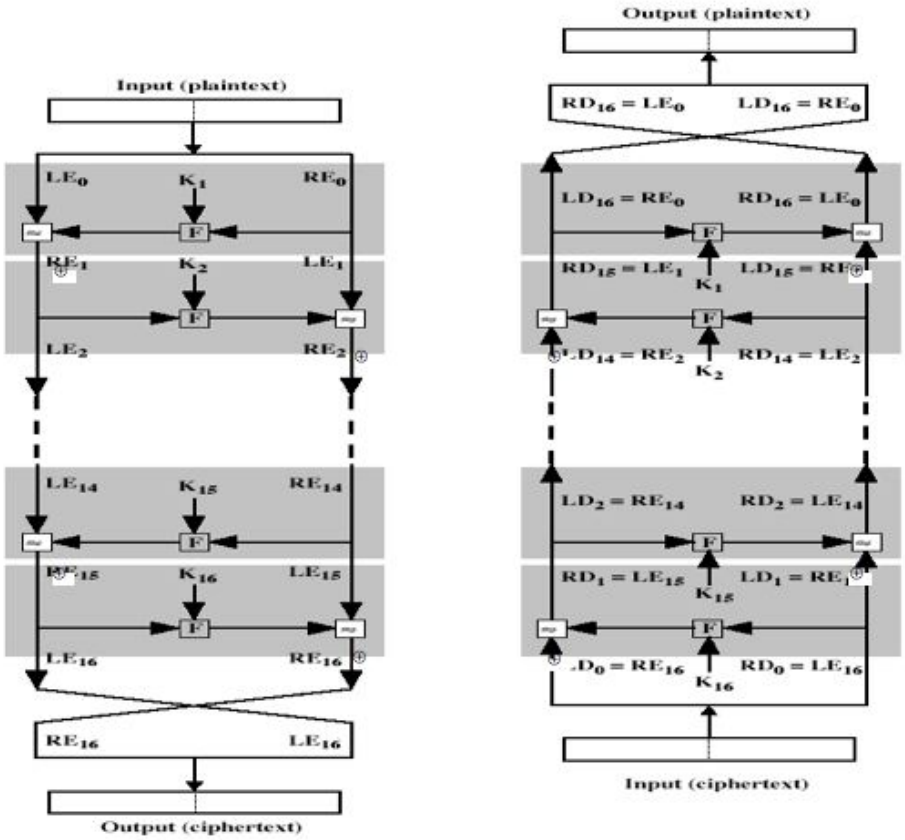
la funzione di round ha la seguente forma:
input: (L_{i-1}, R_{i-1}, k_i), output (L_i, R_i)



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

cifrario di Feistel - invertibiltà

- la funzione di round è una funzione $f : \{0,1\}^{2w} \to \{0,1\}^{2w}$
- la F è una funzione arbitraria $F : \{0,1\}^{w} \to \{0,1\}^{w}$
- comunque sia scelta la F (invertibile o non invertibile) la funzione di round f è comunque invertibile
- da
  $L_i = R_{i-1}$,
  $R_i = L_{i-1} \oplus F(R_{i-1}, k_i)$,
- si ha
  $L_{i-1} = R_i \oplus F(L_i, k_i)$
  $R_{i-1} = L_i$
- 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'algortimo

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

risultati teorici

- se la funzione $F : k \times \{0,1\}^w \to \{0,1\}^w$ è una funzione pseudocasuale sicura
- allora bastano 3 round alla Feistel
- $f : k^3 \times \{0,1\}^{2w} \to \{0,1\}^{2w}$ è una permutazione pseudocasuale sicura
- Luby - Rackoff ca 1985

Data Encryption Standard

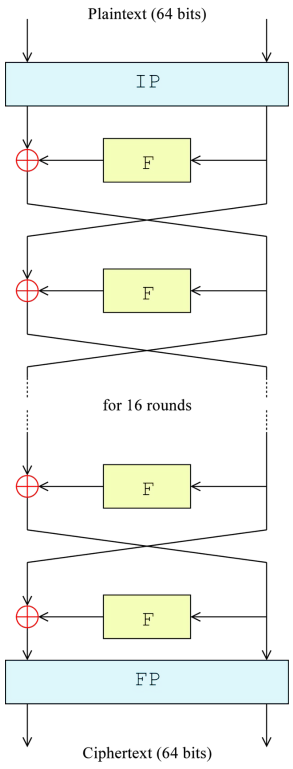
Storia:

- 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'IMB (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

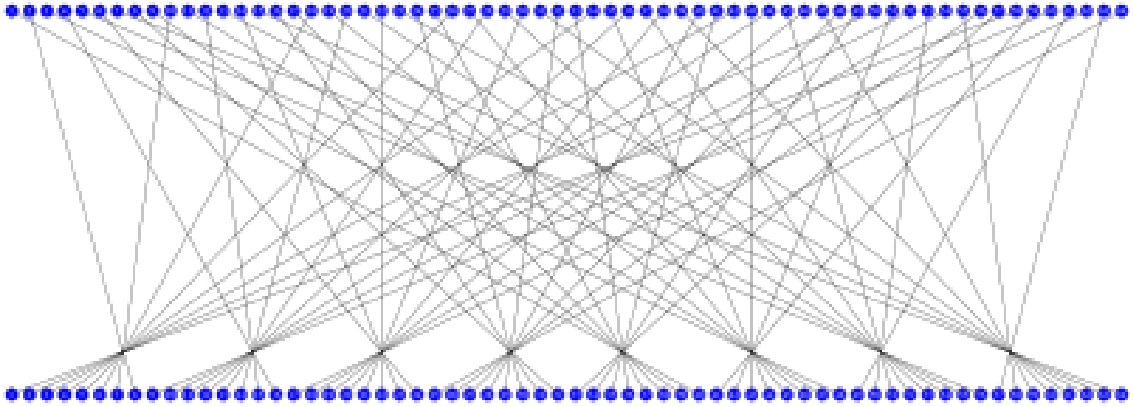
Le due permutazioni, iniziale e finale, sono l'unica cosa che rende il DES diverso da un cifrario di Feistel.
Non servono a aumentare la sicurezza, ma sono legate al processo di caricamento del plaintext nei chip.

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...
$58 \to 1, 50 \to 2 \ldots$

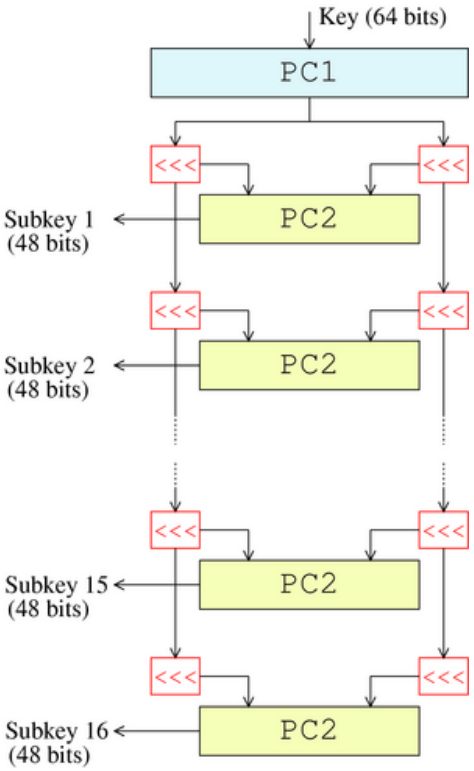
IP



chiave

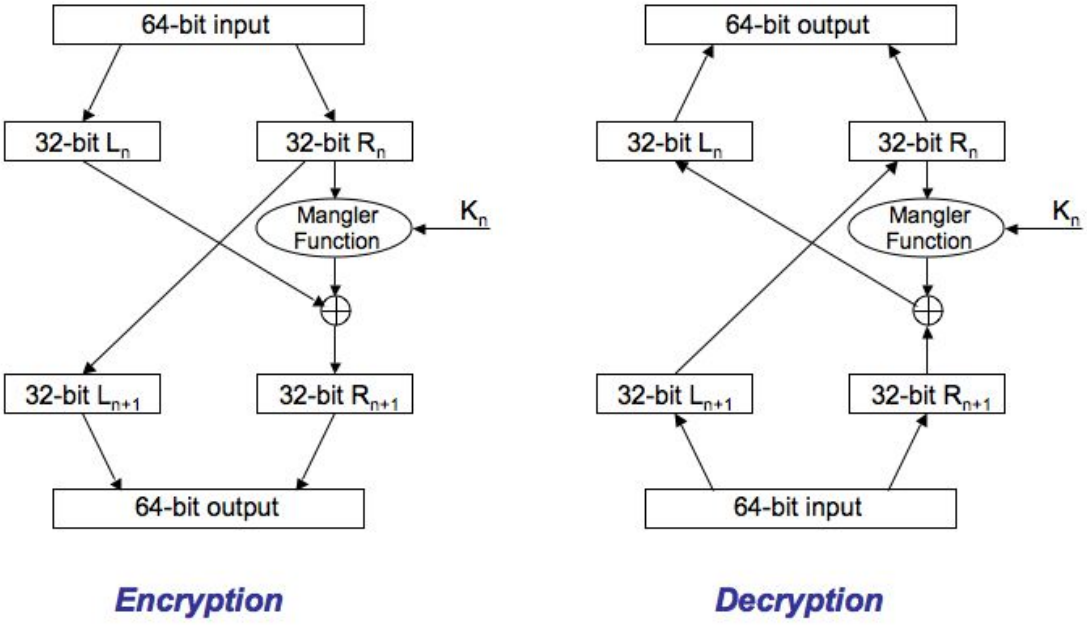
La chiave: 64 bit, di cui 8 sono un controllo di parità. La chiave ha solo 56 bit.
Dalla chiave k ottengo 16 chiavi di round $(k_1, \ldots, k_{16})$ di 48 bit ciascuna.

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

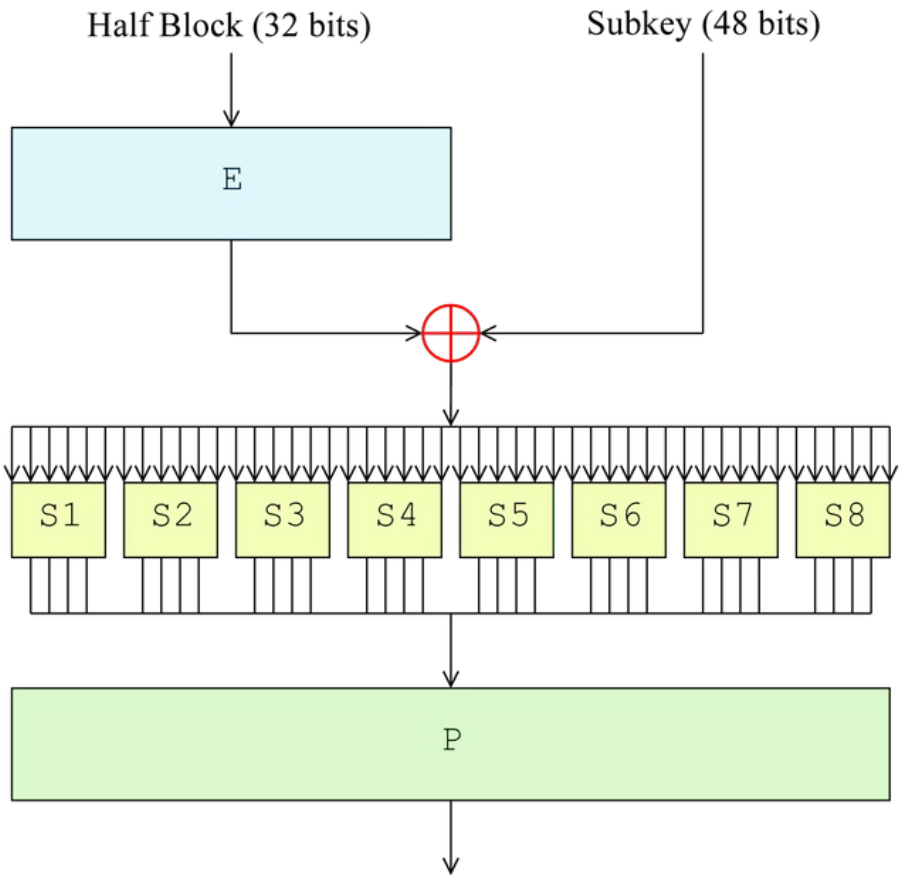
DES round



funzione F, o mangler

Dettagli della funzione F:
al round i
$F(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))$

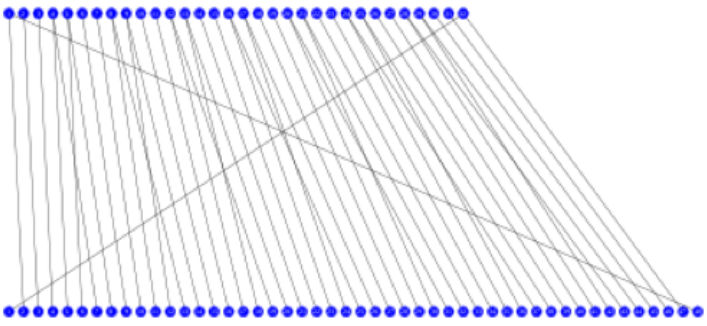
1. Prende i 48 bit della chiave k_i.
2. Espande i 32 bit di destra, R_{i-1}, a 48 bit
3. Esegue lo XOR della chiave e dei 48 bit e invia il risultato in input alle S-Box
4. Le S-box sono tabelle di sostituzione e contrazione predefinite. In output danno 32 bit
5. Questi 32 bit vengono permutati



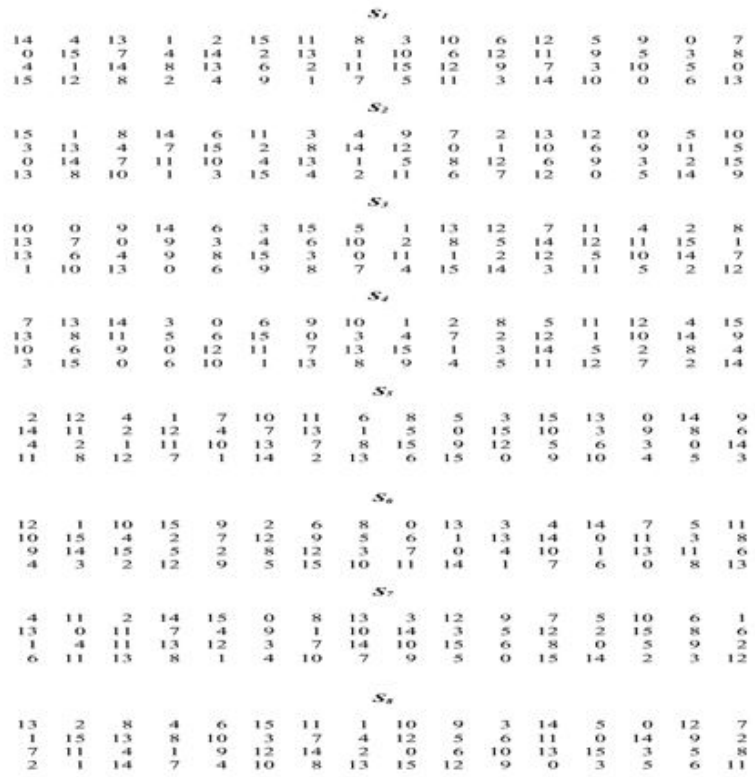
espansione E

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



$1 \to 2, \quad 1 \to 32$
$2 \to 3 \ldots$



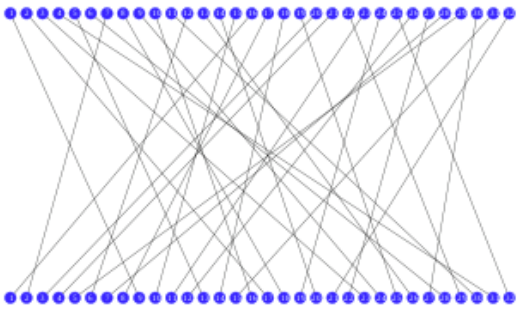
S_5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

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 $\rightarrow$ 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



$16 \to 1, \quad 7 \to 2$
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 $\{0, 1, \ldots, 15\}$
4. cambiando un bit nell'input, cambiano almeno due bit in output
5. $S(x)$ e $S(x + 001100)$ differiscono in almeno 2 bit

il ruolo di P, E, e PC2 è di spargere gli output delle S-box in modo tale che ogni bit in output dipenda da tutti i bit in input in pochi round

Il DES ha un forte *effetto valanga*: una propietà deisderabile 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. Già presente dopo poche iterazioni

Table 3.6 Avalanche Effect in DES: Change in Plaintext

Round		δ
	02468aceeca86420 12468aceeca86420	1
1	3cf03c0fbad22845 3cf03c0fbad32845	1
2	bad2284599e9b723 bad3284539a9b7a3	5
3	99e9b7230bae3b9e 39a9b7a3171cb8b3	18
4	0bae3b9e42415649 171cb8b3ccaca55e	34
5	4241564918b3fa41 ccaca55ed16c3653	37
6	18b3fa419616fe23 d16c3653cf402c68	33
7	9616fe2367117cf2 cf402c682b2cefbc	32
8	67117cf2c11bfc09 2b2cefbc99f91153	33
9	c11bfc09887fbc6c 99f911532eed7d94	32
10	887fbc6c600f7e8b 2eed7d94d0f23094	34
11	600f7e8bf596506e d0f23094455da9c4	37
12	f596506e738538b8 455da9c47f6e3cf3	31
13	738538b8c6a62c4e 7f6e3cf34bc1a8d9	29
14	c6a62c4e56b0bd75 4bc1a8d91e07d409	33
15	56b0bd7575e8fd8f 1e07d4091ce2e6dc	31
16	75e8fd8f25896490 1ce2e6dc365e5f59	32
IP^{-1}	da02ce3a89ecac3b 057cde97d7683f2a	32

Table 3.7 Avalanche Effect in DES: Change in Key

Round		δ
	02468aceeca86420 02468aceeca86420	0
1	3cf03c0fbad22845 3cf03c0f9ad628c5	3
2	bad2284599e9b723 9ad628c59939136b	11
3	99e9b7230bae3b9e 9939136b768067b7	25
4	0bae3b9e42415649 768067b75a8807c5	29
5	4241564918b3fa41 5a8807c5488dbe94	26
6	18b3fa419616fe23 488dbe94aba7fe53	26
7	9616fe2367117cf2 aba7fe53177d21e4	27
8	67117cf2c11bfc09 177d21e4548f1de4	32

Round		δ
9	c11bfc09887fbc6c 548f1de471f64dfd	34
10	887fbc6c600f7e8b 71f64dfd4279876c	36
11	600f7e8bf596506e 4279876c399fdc0d	32
12	f596506e738538b8 399fdc0d6d208dbb	28
13	738538b8c6a62c4e 6d208dbbb9bdeeaa	33
14	c6a62c4e56b0bd75 b9bdeeaad2c3a56f	30
15	56b0bd7575e8fd8f d2c3a56f2765c1fb	33
16	75e8fd8f25896490 2765c1fb01263dc4	30
IP^{-1}	da02ce3a89ecac3b ee92b50606b62b0b	30