Il DES: Data Encryption Standard

Nel 1973 la IBM fece richiesta alla NBS (National Bureau of Standards) per far riconoscere un nuovo algoritmo di crittografia, LUCIFER, come uno standard nazionale. Questo venne inoltrato alla NSA che, dopo averlo analizzato e modificato, creò quello che poi sarebbe stato il DES

Il DES: Data Encryption Standard

Il DES è un algoritmo che ‘‘spezza’’ il testo da criptare in blocchi da 64 bits, criptandoli singolarmente.

Il modo in cui lo fa è chiamato Sistema di Feistel, da cui i round di Feistel che verranno mostrati a seguire.

Per capire però il suo funzionamento, bisogna introdurre un algoritmo semplificato chiamato Simplified-DES, che ha praticamente tutte le caratteristiche del DES originale

Simplified DES

Come il DES cripta singolarmente i messaggi.

In questo caso, per semplicità, consistono in un blocco solo.

Il messaggio è composto da 12 bits e le sue metà (6 bits) sono identificate come L0 ed R0

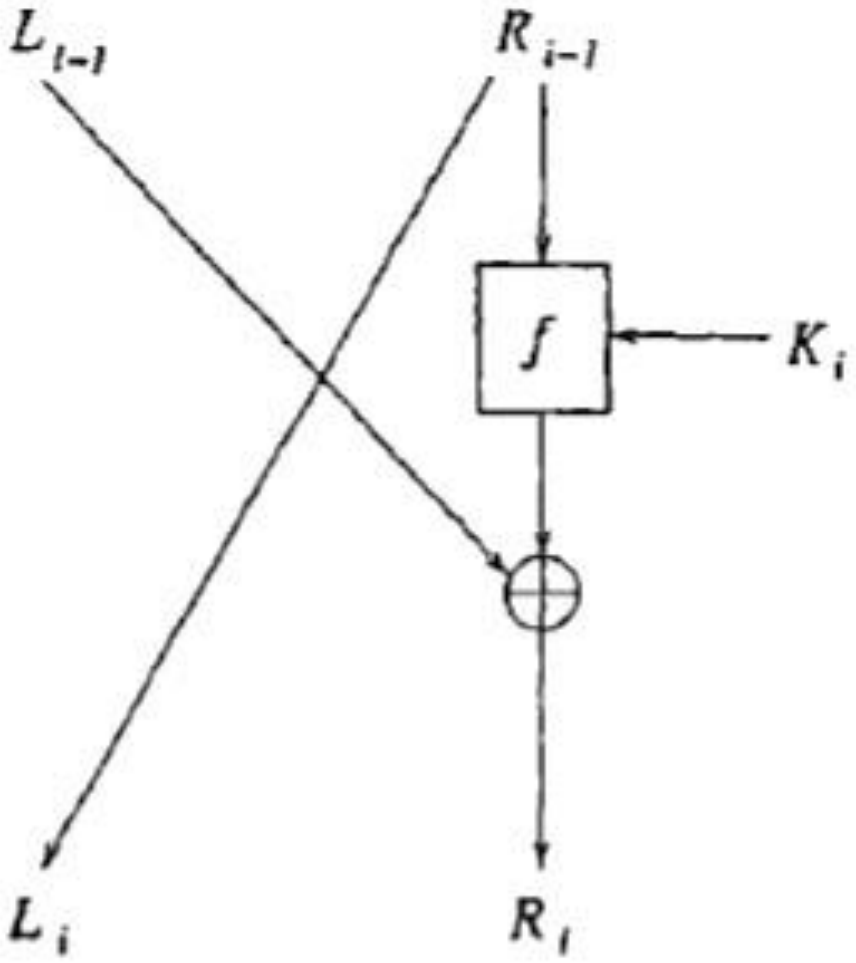
La chiave K è formata da 9 bits

Questa verrà utilizzata per creare 4 sottochiavi da 8 bits leggendo K dall'i-esimo bit

L'algoritmo agisce in 4 rounds, detti di Feistel

Simplified DES - Crittazione

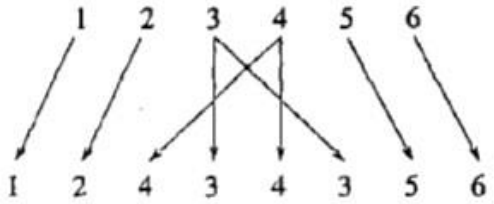
L'i-esimo round dell'algoritmo trasforma l'input Li-1Ri-1 in LiRi



Simplified DES - Crittazione

In un round viene eseguita la funzione f(Ri-1, Ki) che:

- espande Ri-1, che è di 6 bits, a 8 bits secondo lo schema :



- esegue lo XOR tra l'espansione e Ki, ricavando X che viene diviso in due metà XL, XR
- Queste vengono date in input a due S-Boxes, rispettivamente ad S1 ed S2
- Il primo bit identifica la riga, gli altri 3 la colonna

$$S_1 \quad \begin{bmatrix} 101 & 010 & 001 & 110 & 011 & 100 & 111 & 000 \\ 001 & 100 & 110 & 010 & 000 & 111 & 101 & 011 \end{bmatrix}$$

$$S_2 \quad \begin{bmatrix} 100 & 000 & 110 & 101 & 111 & 001 & 011 & 010 \\ 101 & 011 & 000 & 111 & 110 & 010 & 001 & 100 \end{bmatrix}$$

- Unendo i 2 outputs da 3 bits si ottiene Ri, il risultato di f all'i-esimo round

Simplified DES - Crittazione

Esempio:

Input: Li-1Ri-1=011100 100110

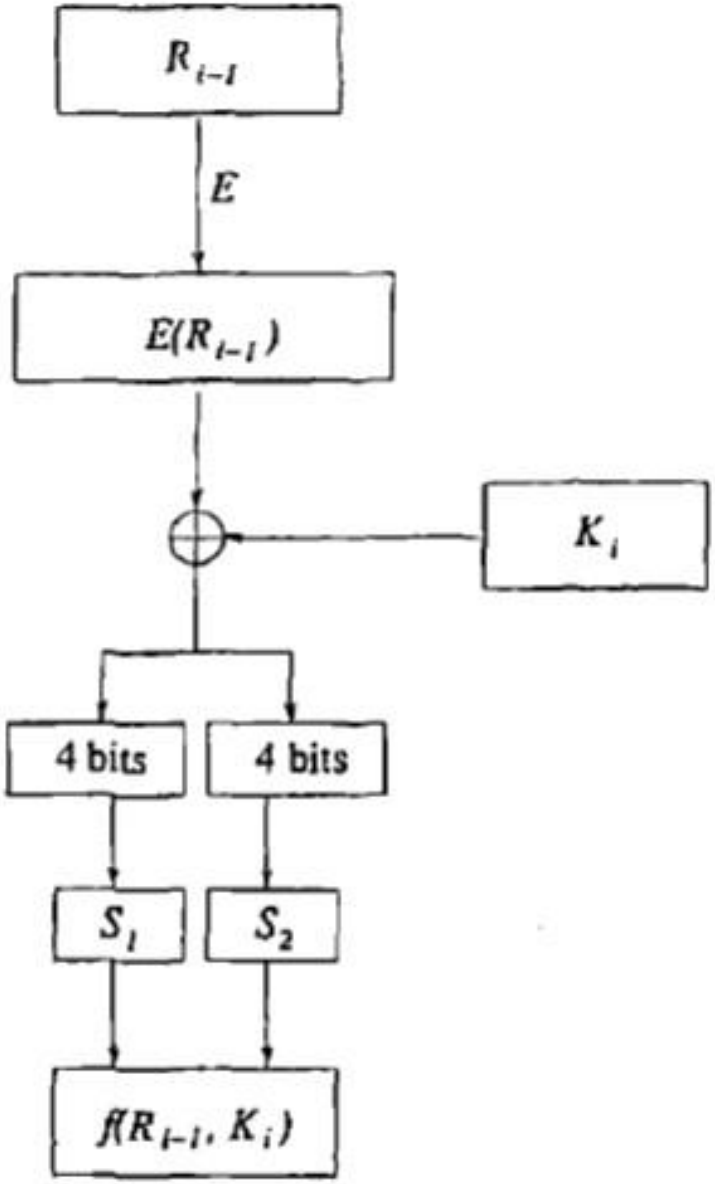
Ki=01100101

F(Ri-1, Ki) XOR Li-1=000100 XOR 011100

Ri=011000

Siccome Li=Ri-1

LiRi=100110011000



Simplified DES - Decrittazione

Il vantaggio del DES è che si può usare lo stesso algoritmo sia per la crittazione che per la decrittazione.

Comincia da un messaggio criptato LnRn facendone lo swap, RnLn

Applica quindi gli stessi passaggi della crittazione ma con le chiavi invertite,

Quindi in f si applicherà prima Kn per poi arrivare a K1 nell'ultimo round, restituendo L0R0

Questo perché: Ln=Rn-1 e Rn=Ln-1 XOR f(Rn-1, Kn)

[Ln] [Rn XOR f(Ln, Kn)]= [Rn-1] [Ln-1 XOR f(Rn-1, Kn) XOR f(Rn-1, Kn]=……L0R0

S-DES – Crittoanalisi Differenziale

Per attaccare l'algoritmo, oltre al classico metodo della forza bruta che prova ogni possibile chiave ed è molto esoso in termini di tempo e risorse, si usa la **crittoanalisi differenziale**.

Il concetto è quello di comparare le differenze di 2 messaggi scelti e dedurne la chiave.

Siccome la chiave è introdotta facendo lo XOR con l'espansione di Ri-1, guardando agli XOR degli input ci permette di ridurre le chiavi da analizzare

S-DES – Crittoanalisi Differenziale

Cominciamo con l'analisi per tre round. Si comincia con L1R1 e K1.

Supponiamo di avere accesso ad una macchina per crittare e ne vogliamo scoprire la chiave.

Usiamo diversi input L1R1 e otteniamo diversi output L4R4

L'unico vincolo è che R1=R1* per ogni input

Ri'=Ri XOR Ri* Li'=Li XOR Li*

Si definisce differenza di LiRi e Li*Ri* il valore Li'Ri'

S-DES – Crittoanalisi Differenziale

Considerando il funzionamento del DES:

L3=R2=L1 XOR f(R1, K2)

R4=L3 XOR f(R3, K4)=L1 XOR f(R1, K2) XOR f(R3, K4)

Avendo un altro messaggio L1*R1* si ha che:

R4’=R4 XOR R4*= L1’ XOR f(R3, K4) XOR f(R3*, K4)=

=R4 XOR L1’= f(R3, K4) XOR f(R3*, K4)

Siccome R3=L4 e R3*=L4* si ha

R4 XOR L1’= f(L4, K4) XOR f(L4*, K4)

Quindi se sappiamo L1’R1’ e gli outputs L4R4 e L4*R4*, sappiamo tutto tranne K4

S-DES – Crittoanalisi Differenziale

Analizzando gli inputs delle S-Boxes date da E(L4) XOR K4 e da E(L4*) XOR K4,

Il loro XOR sarà uguale a E(L4'), semplificando K4

Quindi sappiamo che:

- Gli XOR degli input sono uguali a E(L4'), prima metà S1, seconda S2
- Gli XOR degli output sono uguali a R4' XOR L1' , prima metà S1, seconda S2

Rimane da trovare ogni coppia data da (xor di input XOR 0-16, 0-16) che abbia come xor di output R4' XOR L1', le prime metà rappresentano S1 e le seconde S2

S-DES – Crittoanalisi Differenziale

Una volta ottenute le coppie si trovano delle corrispondenze tra i primi e gli ultimi bits trovati.

Questi rappresentano K4.

Una volta trovata K4, basta fare un backshift per ricondursi a K. Manca però il terzo bit della chiave finale, quindi si avrà una chiave del tipo 00(b3)000111, per trovarlo basta provare con 1 o 0 e vedere se viene prodotto lo stesso messaggio cifrato della macchina a cui si ha accesso.

S-DES – Crittoanalisi Differenziale 4

Per eseguirla in 4 rounds vale lo stesso procedimento per i 3 rounds ma bisogna applicare delle tecniche probabilistiche.

Vi è una debolezza in S1 ed S2:

- In S1 di 16 coppie che hanno xor di input=0011, 12 hanno xor di output=011
- In S2 di 16 coppie che hanno xor di input=1100, 8 hanno xor di output=010

Quindi lo xor di output delle S-Boxes ha una probabilità 3/8 di essere 011010

Supponiamo R0’=001100 e L0’=011010

Quindi R1’=011010 XOR L0’=000000

S-DES – Crittoanalisi Differenziale 4

Quindi applicando dei messaggi casuali L0'R0' ma con xor=011010001100, C'è una probabilità di 3/8 che L1'R1'=001100000000

La strategia è quindi quella di provare diversi L0'R0' e di esaminarne gli outputs.

Si assume quindi che L1'R1'=001100000000 e si applica la crittoanalisi a 3 rounds. Quindi si avranno 3/8 volte la chiave giusta.
Siccome le altre saranno casuali, la chiave che si ripete per più volte è ragionevolmente quella giusta

DES

L’algoritmo originale, come detto, usa blocchi di 64 bits.

Anche la chiave è di 64 bits ma ne vengono usati solo 56. I bits multipli di 8 sono bits di parità, usati in modo tale da far avere un numero dispari di 1 in ogni blocco di 8 bits.

L’algoritmo si divide in 3 fasi:

- Il messaggio di 64 bits è inizialmente permutato in maniera fissa (IP(m))
- Esegue per 16 rounds Li=Ri-1 ed Ri=Li-1 XOR f(Ri-1, Ki) dove Ki e composta da 48 bits ottenuti da K
- Inverte R16L16 ed applica l’inversa della permutazione iniziale per aver il testo cifrato

DES - Crittazione

La funzione f originale si divide in 4 step:

- R viene espansa con la seguente tabella:

Expansion Permutation											
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

- Calcola E(R) XOR Ki di 48 bits, suddividendolo in gruppi da 6 bits
- Ci sono 8 S-Boxes che hanno come input i gruppi da 6 bits. I primi 2 bits determinano la riga, gli altri 4 la colonna

- Infine il risultato viene permutato con la seguente tabella:

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

DES - Chiave

La chiave è ottenuta in 3 passi:

- Vengono scartati i bit di parità ed i rimanenti si permutano con la tabella:

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

ottenendo C0D0 da 56 bits

- Nei 16 rounds si ha che Ci=LSi(Ci-1) e Di=LSi(Di-1) dove Lsi indica uno shift di 1 o 2 posizioni a sinistra:

Number of Key Bits Shifted per Round																
Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Shift	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

- Vengono scelti 48 bits secondo la tabella:

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

DES - Decrittazione

Similmente all'algoritmo semplificato si usa la stessa procedura della crittazione applicando le chiavi in maniera inversa (K16….K1)

Qui però lo switch finale L1R1 è già incluso nel passo 3

La permutazione iniziale, che non ha una funzione dal punto di vista crittografico, è stata introdotta per far girare meglio l'algoritmo nei pc di allora

DES non è un gruppo

Per aumentare la sicurezza dell'algoritmo si può pensare di effettuare un doppio DES con due chiavi EK2(EK1(P))

Questo però aumenta leggermente la protezione dato che gli attacchi Meet in the Middle possono romperlo facilmente

Se il DES è un gruppo equivale a dire che esiste una EK3=(EK1)(EK2)

EKEK=EK2 EKEKEK=EK3 EKnEK=Ekn (Ek)^m-n=P

Sia T=qn+r il numero minimo positivo tale che (EK)^j=Id

n= lunghezza ciclica di P

P=(EK)^T(P)=(EK)^r((EK)^n(EK)^n…(EK)^n(P))=(EK)^r(P)

E1E0=EK (E1E0)^nPi(Pi) gcd(nP1…nP33)>2^56 impossibile perché T<2^56

DES – Operation modes

Molto probabilmente il messaggio m sarà di lunghezza diversa dai 64 bits dell'algoritmo, quindi si utilizzano delle modalità di funzionamento per adattarne il comportamento:

- **Electronic Code Book:** spezza m in vari blocchi da 64 bits
- **Cipher Block Chaining:** migliora ECB. La crittazione di un blocco dipende da quello di prima. Altrimenti posso scoprire porzioni di chiave e modificare il messaggio cifrato
- **Cipher FeedBack:** ECB e CBC funzionano solo con blocchi da 64 bits. Genera bits pseudo casuali per poter lavorare su messaggi corti
- **Output FeedBack:** evita la propagazione di errori di CFB
- **Counter:** simile a CFB e OFB ma l'output non è collegato a quelli precedenti