Descripción del algoritmo DES (Data Encryption Standard)

Jorge Sánchez Arriazu

diciembre de 1999

DES

Introducción

DES (*Data Encryption Standard*, estándar de cifrado de datos) es un algoritmo desarrollado originalmente por IBM a requerimiento del NBS (National Bureau of Standards, Oficina Nacional de Estandarización, en la actualidad denominado NIST, National Institute of Standards and Technology, Instituto Nacional de Estandarización y Tecnología) de EE.UU. y posteriormente modificado y adoptado por el gobierno de EE.UU. en 1977 como estándar de cifrado de todas las informaciones sensibles no clasificadas. Posteriormente, en 1980, el NIST estandarizó los diferentes modos de operación del algoritmo. Es el más estudiado y utilizado de los algoritmos de clave simétrica.

El nombre original del algoritmo, tal como lo denominó IBM, era Lucifer. Trabajaba sobre bloques de 128 bits, teniendo la clave igual longitud. Se basaba en operaciones lógicas booleanas y podía ser implementado fácilmente, tanto en software como en hardware.

Tras las modificaciones introducidas por el NBS, consistentes básicamente en la reducción de la longitud de clave y de los bloques, DES cifra bloques de 64 bits, mediante permutación y sustitución y usando una clave de 64 bits, de los que 8 son de paridad (esto es, en realidad usa 56 bits), produciendo así 64 bits cifrados.

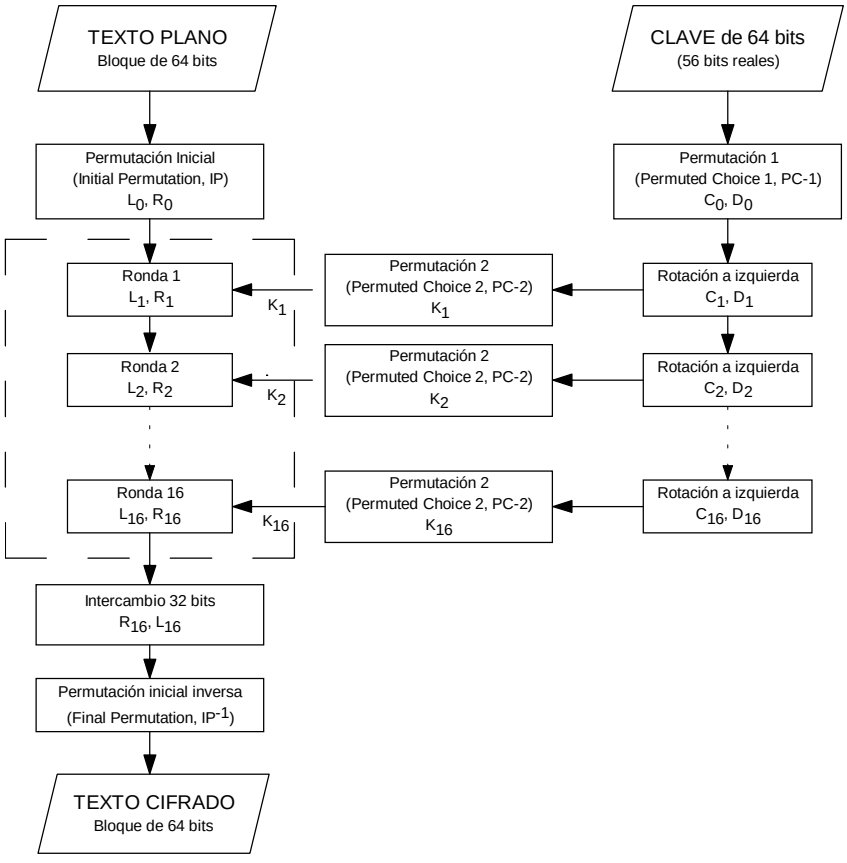

Figura 1: Esquema general del algoritmo DES

DES tiene 19 etapas diferentes.

La primera etapa es una transposición, una permutación inicial (IP) del texto plano de 64 bits, independientemente de la clave. La última etapa es otra transposición (IP-1), exactamente la inversa de la primera. La penúltima etapa intercambia los 32 bits de la izquierda y los 32 de la derecha. Las 16 etapas restantes son una Red de Feistel de 16 rondas.

En cada una de las 16 iteraciones se emplea un valor, K_i, obtenido a partir de la clave de 56 bits y distinto en cada iteración

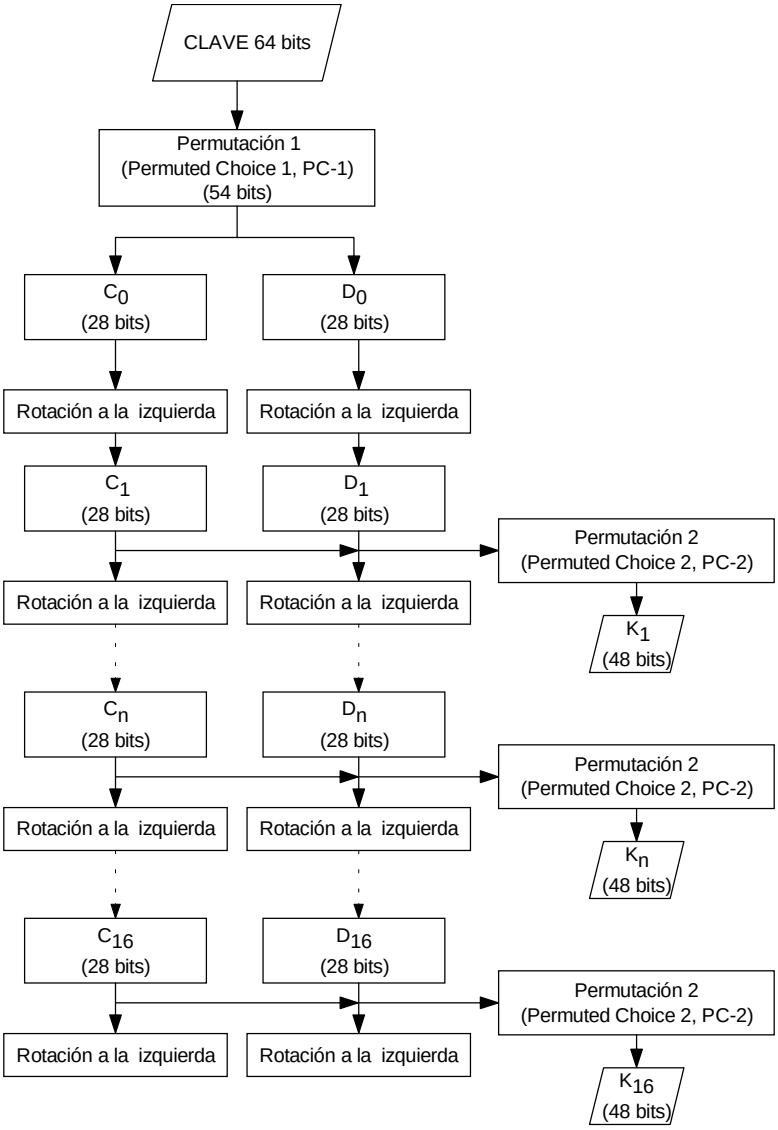

Figura 2: Cálculo de las subclaves, K_i

Se realiza una permutación inicial (PC-1) sobre la clave, y luego la clave obtenida se divide en dos mitades de 28 bits, cada una de las cuales se rota a izquierda un número de bits determinado que no siempre es el mismo. K_i se

deriva de la elección permutada (PC-2) de 48 de los 56 bits de estas dos mitades rotadas.

La función f de la red de Feistel se compone de una permutación de expansión (E), que convierte el bloque correspondiente de 32 bits en uno de 48. Después realiza una or-exclusiva con el valor K_i, también de 48 bits, aplica ocho S-Cajas de 6*4 bits, y efectúa una nueva permutación (P).

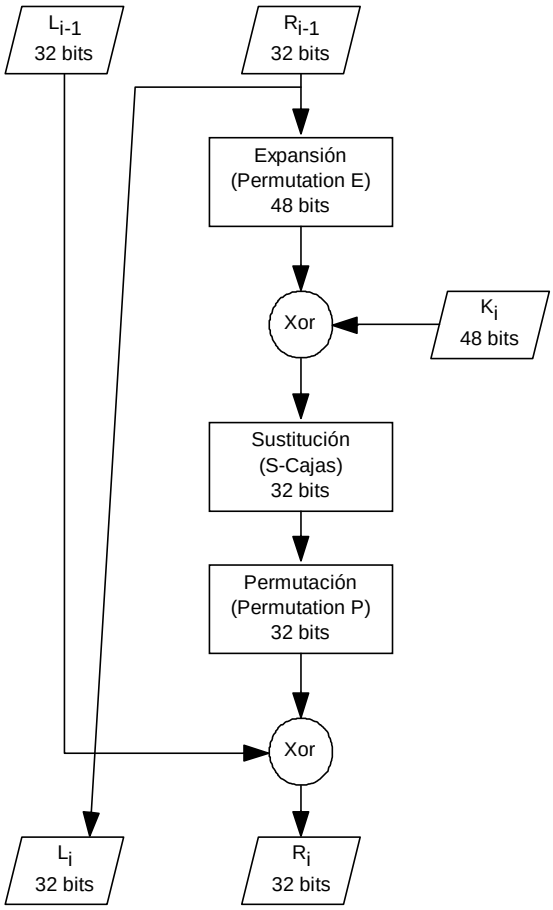

Figura 3: Ronda del algoritmo DES

Para descifrar basta con usar el mismo algoritmo empleando las K_i en orden inverso

Está descrito oficialmente en FIPS PUB 46.

Descripción paso a paso del algoritmo DES

Esta parte del documento es una traducción de un artículo que se puede encontrar en la página de internet http://www.aci.net/kalliste/des.htm y cuyo autor es Matthew Fischer (mfischer@heinous.isca.uiowa.edu).

A continuación se describen los pasos necesarios para implementar este algoritmo.

(i) Encriptación:

1.- Procesar la clave.

1.1.- Solicitar una clave de 64 bits al usuario.
La clave se puede introducir directamente o puede ser el resultado de alguna operación anterior, ya que no hay ninguna especificación al respecto.
De cada uno de los ocho bytes se elimina el octavo bit (el menos significativo)

1.2.- Calcular las subclaves.

1.2.1.- Realizar la siguiente permutación en la clave de 64 bits reduciéndose la misma a 56 bits (El bit 1, el más significativo, de la clave transformada es el bit 57 de la

Permuted Choice 1 (PC-1)

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

clave original, el bit 2 pasa a ser el bit 49, etc.).

1.2.2.- Dividir la clave permutada en dos mitades de 28 bits cada una. C(0) el bloque que contiene los 28 bits de mayor peso y D(0) los 28 bits restantes.

1.2.3.- Calcular las 16 subclaves (Empezar con i=1)

1.2.3.1.- Rotar uno o dos bits a la izquierda C(i-1) y D(i-1) para obtener C(i) y D(i), respectivamente. El número de bits de desplazamiento está dado por la tabla siguiente:

Ronda	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits despl.	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

1.2.3.2.- Concatenar C(i) y D(i) y permutar como se indica a continuación. Así se obtiene K(i), que tiene una longitud de 48 bits.

Permuted Choice 2 (PC-2)

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

1.2.3.3.- Ir a 1.2.3.1. hasta que se haya calculado K(16).

2.- Procesar el bloque de datos de 64 bits.

2.1.- Obtener un bloque de datos de 64 bits. Si el bloque contiene menos de 64 bits debe ser completado para poder continuar con el siguiente paso.

2.2.- Realizar la siguiente permutación del bloque:

Initial Permutation (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

2.3.- Dividir el bloque resultante en dos mitades de 32 bits cada una. L(0) el bloque que contiene los 32 bits de mayor peso y R(0) el resto.

2.4.- Aplicar las 16 subclaves obtenidas en el paso 1

2.4.1.- E(R(i)). Expandir R(i) de 32 a 48 bits de acuerdo con la tabla que se muestra debajo:

Expansion (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

2.4.2.- E(R(i-1)) Xor K(i). Or-exclusiva del resultado del paso 2.4.1. con K(i)

2.4.3.- B(1), B(2),..., B(8). Partir E(R(i-1)) Xor K(i) en ocho bloques de seis bits. B(1) representa a los bits 1-6, B(2) representa a los bits 7-12,..., B(8) representa a los bits 43-48.

2.4.4.- S(1)(B(1)), S(2)(B(2)),..., S(8)(B(8)). Sustituir todos los B(j) por los valores correspondientes de las S-Cajas o tablas de sustitución (Substitution Boxes, S-Boxes) de 6*4 bits, según se indica en los subapartados que siguen. Todos los valores de las S-Cajas se consideran de 4 bits de longitud. (Ver S-cajas del algoritmo DES, página siguiente)

2.4.4.1.- Tomar los bits 1º y 6º de B(j) y formar un número de 2 bits que llamaremos m. Este valor nos indicará la fila en la tabla de sustitución correspondiente S(j). Obsérvese que m=0 representa la 1ª fila y m=3 la última.

2.4.4.2.- Con los bits 2º a 5º de B(j) formar otro número, n, de cuatro bits que indicará la columna de S(j) en la que buscar el valor de sustitución. En esta ocasión n=0 representa la 1ª columna y n=15 la última columna.

2.4.4.3.- Reemplazar B(j) con S(j)(m,n), m fila y n columna.

2.4.4.4.- Ejemplo. Sea B(3)=42, en binario B(3)=101010. Buscaremos el nuevo valor de B(3) en S(3). Fila m y columna n, según lo expuesto anteriormente m=10, n=0101, y en decimal m=2 y n=5. Por tanto, B(3) será S(3)(2,5)=15

2.4.4.5.- Volver a 2.4.4.1. hasta que todos los bloques B(j) hayan sido reemplazados por el valor de S(j) adecuado.

2.4.5.- P[S(1)(B(1))... S(2)(B(8))]. Concatenar los bloques B(1) a B(8) y permutar los 32 bits (cuatro bits cada B(j)) en función de esta tabla:

Permutation 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

2.4.5.1.- Volver a 2.4.4.1. hasta que todos los bloques B(j) hayan sido reemplazados por el valor de S(j) adecuado.

Fila	Columna																S-Caja
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	**S_1**
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	**S_2**
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	**S_3**
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	**S_4**
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	**S_5**
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	**S_6**
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	**S_7**
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	**S_8**
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

S-cajas del algoritmo DES

2.4.6.- P[S(1)(B(1))... S(2)(B(8))]. Concatenar los bloques B(1) a B(8) y permutar los 32 bits (cuatro bits cada B(j)) en función de esta tabla:

Permutation 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

2.4.7.- R(i). Realizar una or-exclusiva entre el valor resultante y L(i-1). Este valor será R(i). Por tanto, R(i)=L(i-1) Xor P[S(1)(B(1))... S(2)(B(8))]

2.4.8.- L(i). L(i)=R(i-1)

2.4.9.- Repetir desde 2.4.1. hasta que se hayan aplicado las 16 subclaves.

2.5.- Hacer la siguiente permutación del bloque R(16)L(16). Obsérvese que esta vez R(16) precede a L(16)

Final Permutation (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

(ii) Desencriptación:

Usar el mismo proceso descrito con anterioridad pero empleando las subclaves en orden inverso, esto es, en lugar de aplicar K(1) para la primera iteración aplicar K(16), K(15) para la segunda y así hasta K(1).

Vectores de prueba para DES

Estos son algunos de los valores que se pueden emplear para comprobar si la implementación que se ha llevado a cabo del algoritmo DES es correcta.

Ejemplo 1:

Bloque a cifrar en hexadecimal:

0123456789abcdef

Clave:

133457799BBCDFF1

Cifrado:

85E813540F0AB405

Ejemplo 2:

Bloque a cifrar en hexadecimal:

8787878787878787

Clave:

0E329232EA6D0D73

Cifrado:

0000000000000000

Ejemplo 3:

Texto a cifrar:

"Your lips are smoother than vaseline"

(36 caracteres+Chr(13)+Chr(10))

En hexadecimal:

596f7572206c6970 732061726520736d 6f6f746865722074
68616e2076617365 6c696e650d0a0000

Relleno

Clave:

0E329232EA6D0D73

Cifrado:

C0999FDDE378D7ED 727DA00BCA5A84EE 47F269A4D6438190
D9D52F78F5358499 828AC9B453E0E653

Ejemplo 4:

Texto a cifrar (24 Caracteres):

"Now is the time for all "

(24 Caracteres -hay un espacio en blanco del final-)

En hexadecimal:

4e6f772069732074 68652074696d6520 666f7220616c6c20

Clave:

0123456789ABCDEF

Cifrado:

3FA40E8A984D4815 6A271787AB8883F9 893D51EC4B563B53