جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

كلية العلوم

قسم علوم الامن السيبراني

Cyber Security Department

Subject: Data Encryption Standard
Class: 2nd
Lecturer: Asst.Lect. Mustafa Ameer Awadh

Lecture: (2)

Study Year: 2024-2025

Data Encryption Standard DES

The origins of DES go back to the early 1970s. In 1972, after concluding a study on the US government's computer security needs, the US standards body NBS (National Bureau of Standards) now named NIST (National Institute of Standards and Technology) identified a need for a government- wide standard for encrypting unclassified, sensitive information. Accordingly, on 15 May 1973, after consulting with the NSA, NBS solicited proposals for a cipher that would meet rigorous design criteria. None of the submissions, however, turned out to be suitable. A second request was issued on 27 August 1974. This time, IBM submitted a candidate which was deemed acceptable a cipher developed during the period 1973–1974 based on an earlier algorithm, Horst Feistel's Lucifer cipher. The team at IBM involved in cipher design and analysis included Feistel, Walter Tuchman, Don Coppersmith, Alan Konheim, Carl Meyer, Mike Matyas, Roy Adler, Edna Grossman, Bill Notz, Lynn Smith, and Bryant Tuckerman.

Overall Structure

The algorithm's overall structure is shown in Figure (2-1) there are 16 identical stages of processing, termed *rounds*. There is also an initial and final permutation, termed *IP* and *FP*, which are inverses (IP "undoes" the action of FP, and vice versa). IP and FP have almost no cryptographic significance but were apparently included in order to facilitate loading blocks in and out of mid- 1970s hardware, as well as to make DES run slower in software. Before the main rounds, the block is divided into two 32-bit halves and processed alternately; this crisscrossing is known as the Feistel scheme. The Feistel structure ensures that decryption and encryption are very similar processes the only difference is that the subkeys are applied in the reverse order when decrypting. The rest of the algorithm is identical. This greatly simplifies implementation, particularly in hardware, as there is no need for separate encryption and decryption algorithms. The red symbol denotes the exclusive-OR (XOR) operation. The *F-function* scrambles half a block together with some of the key. The output from the F-function is then combined with the other half of the block, and the halves are swapped before the next round. After the final round, the halves are not

swapped; this is a feature of the Feistel structure, which makes encryption and decryption similar processes.

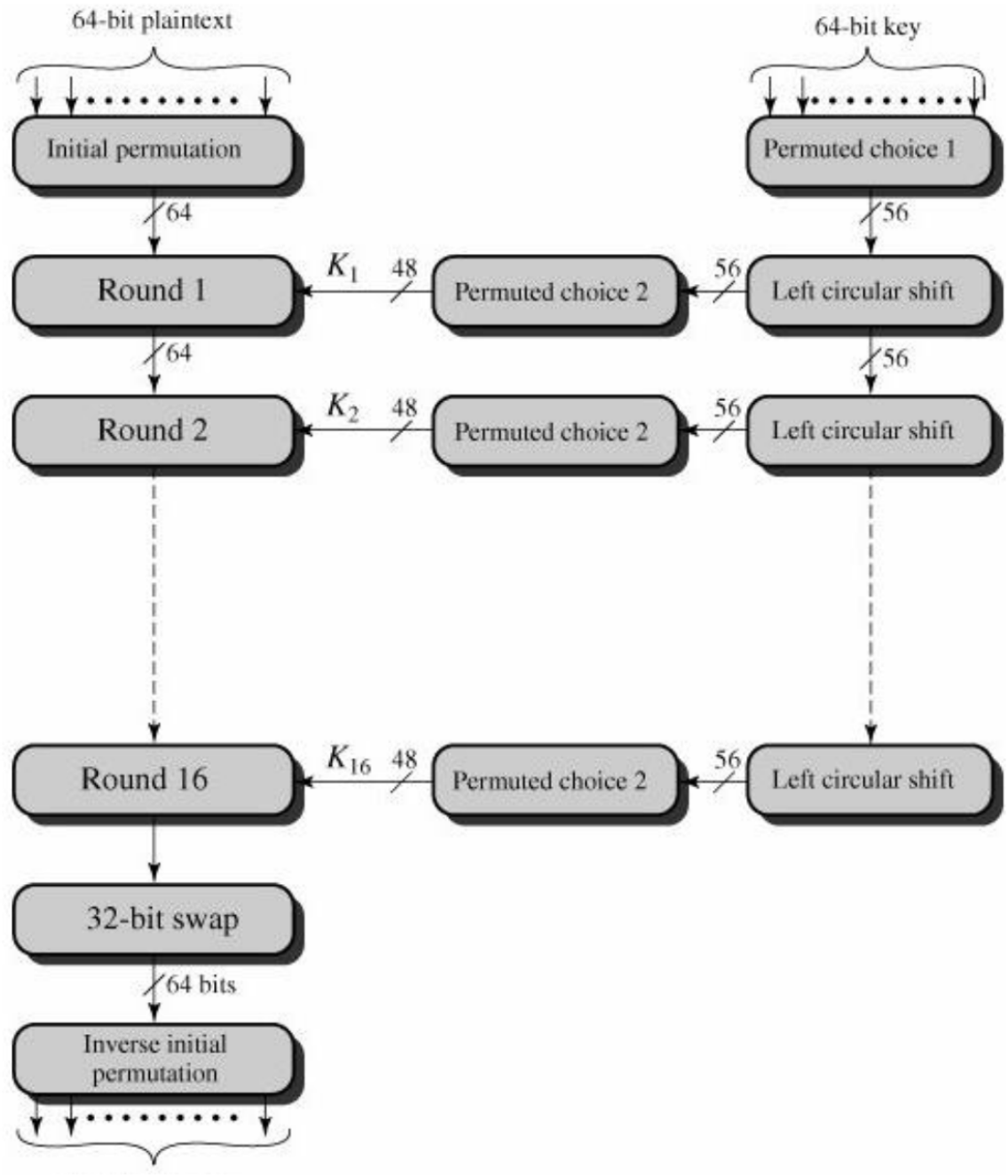

Fig.1 DES technique structure.

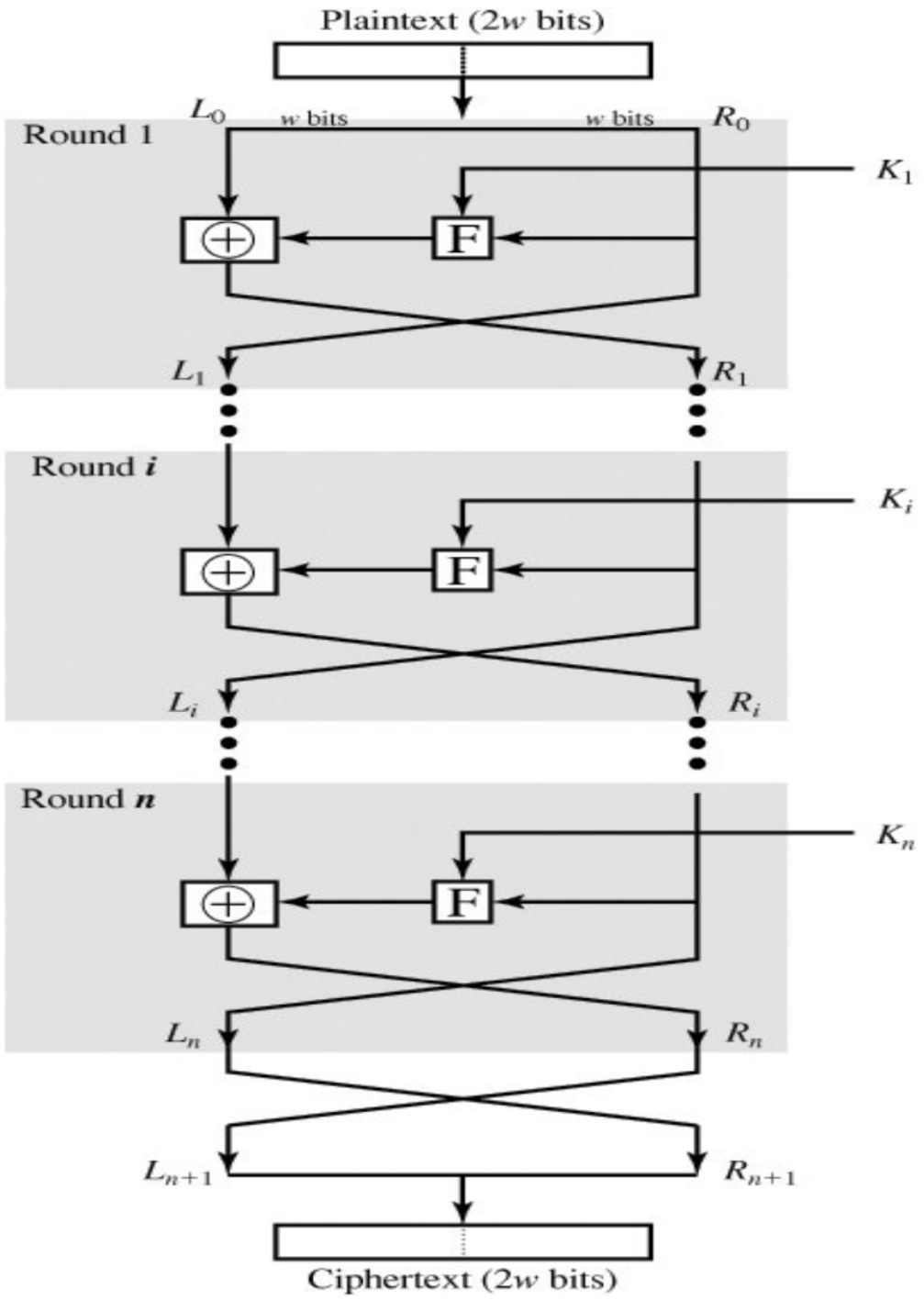

Fig.2 Structure of DES Block Cipher.

The Feistel (F) Function

The F-function, depicted in Figure (3), operates on half a block (32 bits) at a time and consists of four stages:

1. *Expansion* the 32-bit half-block is expanded to 48 bits using the *expansion permutation*, denoted *E* in the diagram, by duplicating some of the bits.

2. *Key mixing* the result is combined with a *subkey* using an XOR operation. Sixteen 48-bit subkeys one for each round are derived from the main key using the *key schedule* (described below).

3. *Substitution* after mixing in the subkey, the block is divided into eight 6-bit pieces before processing by the *S-boxes*, or *substitution boxes*. Each of the eight S-boxes replaces its six input bits with four output bits according to a **non-linear transformation**, provided in the form of a look up table. The S- boxes provide the core of the security of DES without them; the cipher would be linear, and trivially breakable.

4. *Permutation* finally, the 32 outputs from the S-boxes are rearranged according to a fixed permutation, the *P-box*. The alternation of substitution from the S-boxes, and permutation of bits from the P-box and Expansion provides so-called "confusion and diffusion" respectively, a concept identified by Claude Shannon in the 1940s as a necessary condition for a secure yet practical cipher.

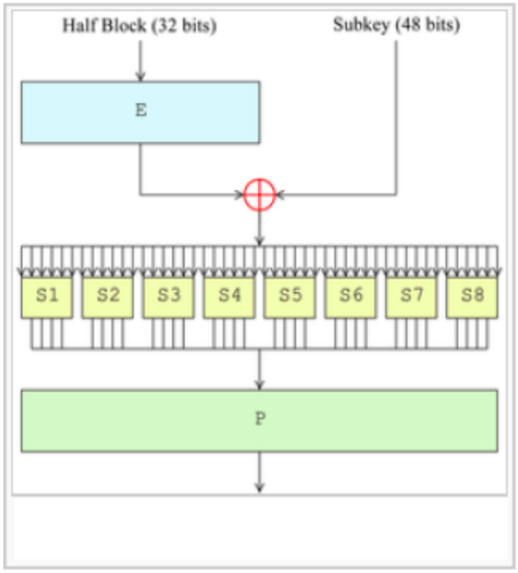

Fig.3 The Feistel Function (F-function) of DES.

Key Schedule

Figure (4) illustrates the *key schedule* for encrypting the algorithm which generates the subkeys. Initially, 56 bits of the key are selected from the initial 64 by *Permuted Choice 1* (*PC-1*) the remaining eight bits are either discarded or used as parity check bits. The 56 bits are then divided into two 28-bit halves; each half is thereafter treated separately. In successive rounds, either halves are rotated left by one or two bits (specified for each round), and then 48 subkey bits are selected by *Permuted Choice 2* (*PC-2*) 24 bits from the left half, and 24 from the right. The rotations (denoted by "<<<" in the diagram) mean that a different set of bits is used in each subkey, the key schedule for decryption is similar the subkeys are in reverse order compared to encryption. Apart from that change, the process is the same as for encryption.

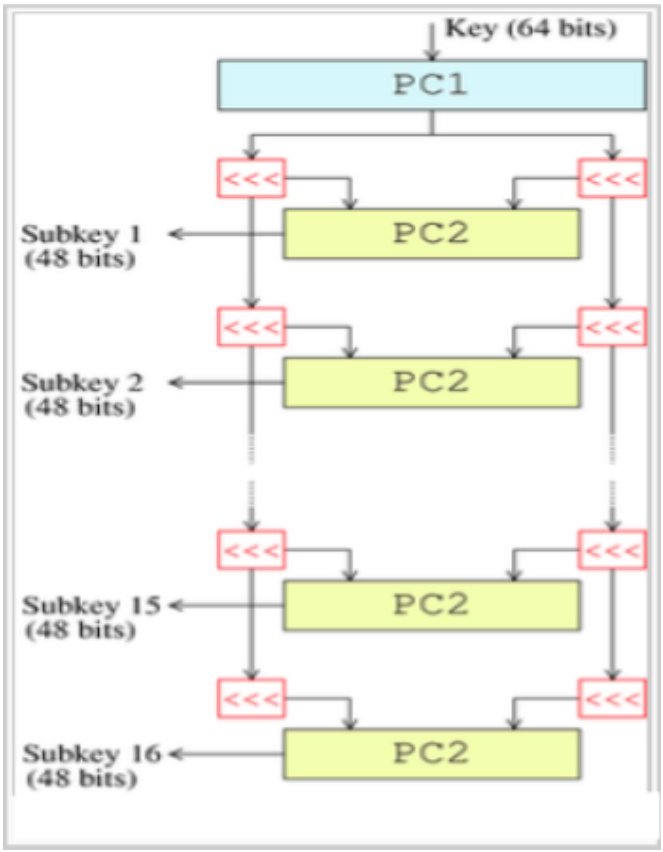

Fig.4 Key Schedule

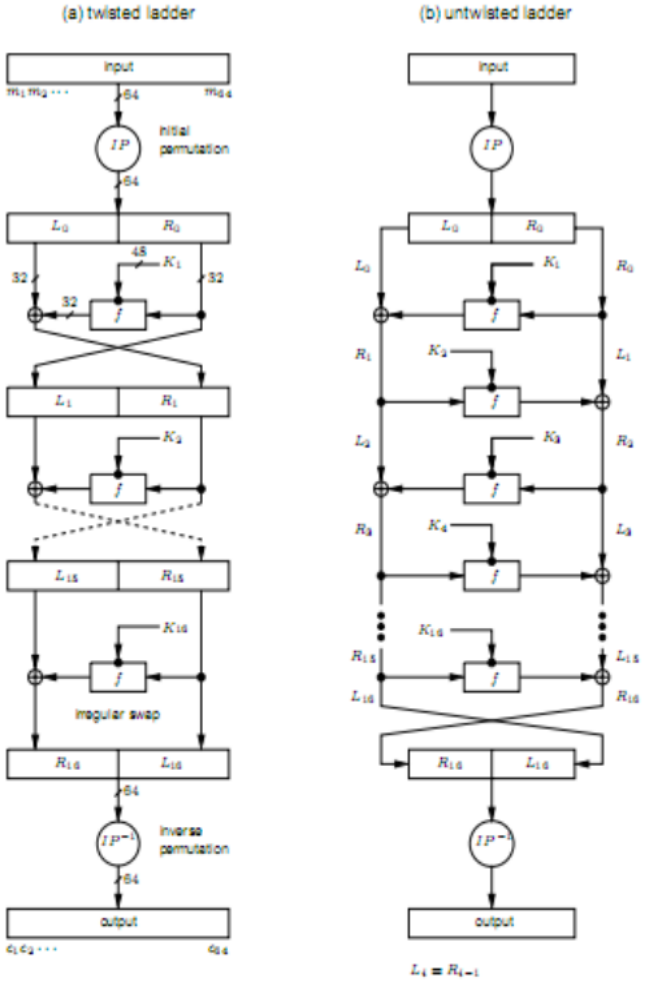

Fig.5 Twisted ladder And Untwisted

Decrypting DES

After all the substitutions, permutations, XORs, and shifting around, you might think that the decryption algorithm is completely different and just as confusing as the encryption algorithm. On the contrary, the various operations were chosen to produce a very useful property: The same algorithm works for both encryption and decryption. With DES it is possible to use the same function to encrypt **or decrypt a block.** The only difference is that the keys must be used in the reverse order. That is, if the encryption keys for each round are K1 K2 K3,..., K16 then the decryption

keys are K16 K15 K14,..., K1. The algorithm that generates the key used for each round is circular as well. The key shift is a right shift, and the number of positions shifted is 0,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1.

Example:

Key init(5b5a5767, 6a56676e)
PC1(Key) C=00ffd820, D=ffec9370
KeyRnd01 C1=01ffb040, D1=ffd926f0, PC2(C, D) = (38 09 1b 26 2f 3a 27 0f)
KeyRnd02 C2=03ff6080, D2=ffb24df0, PC2(C, D) = (28 09 19 32 1d 32 1f 2f)
KeyRnd03 C3=0ffd8200, D3=fec937f0, PC2(C, D) = (39 05 29 32 3f 2b 27 0b)
KeyRnd04 C4=3ff60800, D4=fb24dff0, PC2(C, D) = (29 2f 0d 10 19 2f 1d 3f)
KeyRnd05 C5=ffd82000, D5=ec937ff0, PC2(C, D) = (03 25 1d 13 1f 3b 37 2a)
KeyRnd06 C6=ff608030, D6=b24dfff0, PC2(C, D) = (1b 35 05 19 3b 0d 35 3b)
KeyRnd07 C7=fd8200f0, D7=c937ffe0, PC2(C, D) = (03 3c 07 09 13 3f 39 3e)
KeyRnd08 C8=f60803f0, D8=24dfffb0, PC2(C, D) = (06 34 26 1b 3f 1d 37 38)
KeyRnd09 C9=ec1007f0, D9=49bfff60, PC2(C, D) = (07 34 2a 09 37 3f 38 3c)
KeyRnd10 C10=b0401ff0, D10=26fffd90, PC2(C, D) = (06 33 26 0c 3e 15 3f 38)
KeyRnd11 C11=c1007fe0, D11=9bfff640, PC2(C, D) = (06 02 33 0d 26 1f 28 3f)
KeyRnd12 C12=0401ffb0, D12=6fffd920, PC2(C, D) = (14 16 30 2c 3d 37 3a 34)
KeyRnd13 C13=1007fec0, D13=bfff6490, PC2(C, D) = (30 0a 36 24 2e 12 2f 3f)
KeyRnd14 C14=401ffb00, D14=fffd9260, PC2(C, D) = (34 0a 38 27 2d 3f 2a 17)
KeyRnd15 C15=007fec10, D15=fff649b0, PC2(C, D) = (38 1b 18 22 1d 32 1f 37)
KeyRnd14 C14=401ffb00, D14=fffd9260, PC2(C, D) = (34 0a 38 27 2d 3f 2a 17)

Table 1

Initial Permutation

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.

Table 2

Key Permutation (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.

Table 3

Number of Key Bits Shifted per Round

Round 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Number 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1

Table 4
Compression Permutation (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.

Table 5
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.

The S-Box Substitution

After the compressed key is XORed with the expanded block, the 48-bit result moves to a substitution operation. The substitutions are performed by eight substitution boxes, or S

Table 6

S-Boxes

S-box 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-box 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-box 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-box 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,

S-box 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,
41, 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-box 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-box 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-box 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

The P-Box Permutation

The 32-bit output of the S-box substitution is permuted according to a P-box. This permutation maps each input bit to an output position; no bits are used twice, and no bits are ignored. Table 7 shows the position to which each bit moves. For example, bit 21 moves to bit 4. while bit 4 moves to bit 3 1.

Table 7

P-Box Permutation

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

Finally, the result of the P-box permutation is XORed with the left half of the initial 64-bit block. Then the left and right halves are switched, and another round begins.

The Final Permutation

The final permutation is the inverse of the initial permutation and is described. Note that the left and right halves are not exchanged after the last round of DES; instead, the concatenated block R16L16 is used as the input to the final permutation. There's nothing going on here; exchanging the halves and shifting around the permutation would yield the same result. This is so that the algorithm can be used to both encrypt and decrypt.