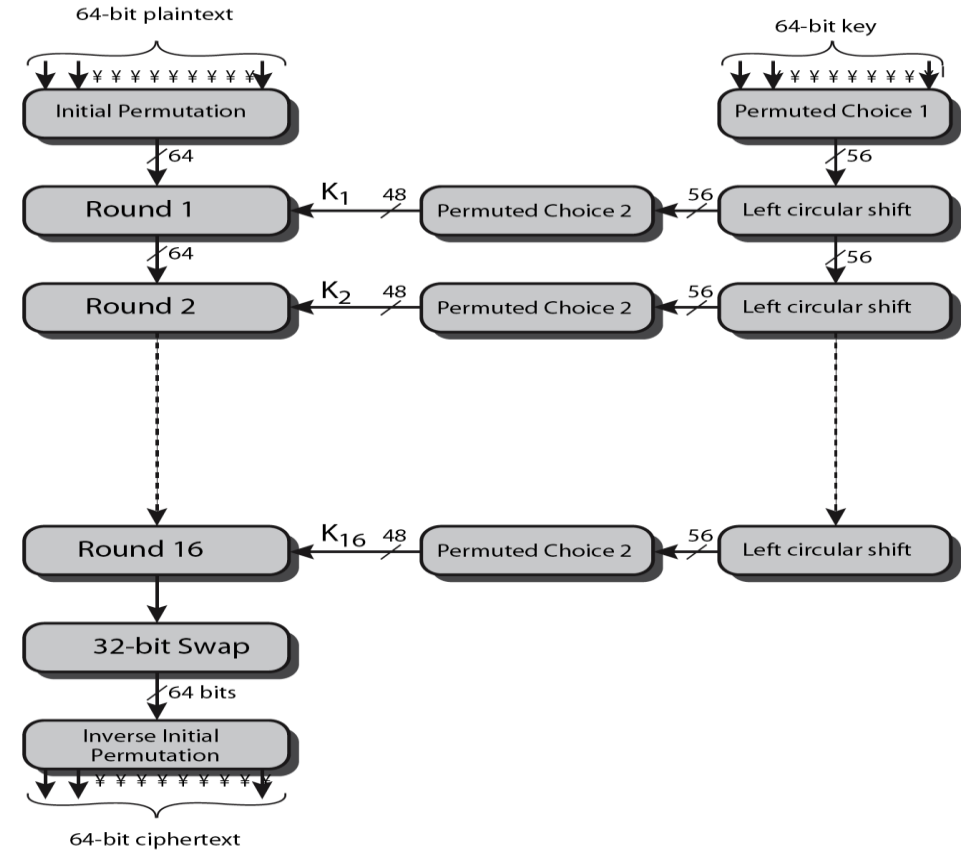
DATA ENCRYPTION STANDARD (DES)

DES (and most of the other major symmetric ciphers) is based on a cipher known as the Feistel block cipher. This was a block cipher developed by the IBM cryptography researcher Horst Feistel in the early 70's. It consists of a number of rounds where each round contains bit-shuffling, non-linear substitutions (S-boxes) and exclusive OR operations. Most symmetric encryption schemes today are based on this structure (known as a feistel network). As with most encryption schemes, DES expects two inputs - the plaintext to be encrypted and the secret key. The manner in which the plaintext is accepted, and the key arrangement used for encryption and decryption, both determine the type of cipher it is. DES is therefore a symmetric, 64 bit block cipher as it uses the same key for both encryption and decryption and only operates on 64 bit blocks of data at a time5 (be they plaintext or ciphertext). The key size used is 56 bits, however a 64 bit (or eight-byte) key is actually input. The least significant bit of each byte is either used for parity (odd for DES) or set arbitrarily and does not increase the security in any way. All blocks are numbered from left to right which makes the eight bit of each byte the parity bit. Once a plain-text message is received to be encrypted, it is arranged into 64 bit blocks required for input. If the number of bits in the message is not evenly divisible by 64, then the last block will be padded. Multiple permutations and substitutions are incorporated throughout in order to increase the difficulty of performing a cryptanalysis on the cipher. However, it is generally accepted that the initial and final permutations offer little or no contribution to the security of DES and in fact some software implementations omit them (although strictly speaking these are not DES as they do not adhere to Overall structure Figure 2.2 shows the sequence of events that occur during an encryption operation. DES performs an initial permutation on the entire 64 bit block of data. It is then split into 2, 32 bit sub-blocks, Li and Ri which are then passed into what is known as a round (see figure 2.3), of which there are 16 (the subscript i in Li and Ri indicates the current round). Each of the rounds are identical and the effects of increasing their number is twofold - the algorithms security is increased and its temporal efficiency decreased. Clearly these are two conflicting outcomes and a compromise must be made. For DES the number chosen was 16, probably to guarantee the elimination of any correlation between the ciphertext and either the plaintext or key. At the end of the 16th round, the 32 bit Li and Ri output quantities are swapped to create what is known as the pre-output. This [R16, L16] concatenation is permuted using a function which is the exact inverse of the initial permutation. The output of this final permutation is the 64 bit ciphertext.



So in total the processing of the plaintext proceeds in three phases as can be seen from the left hand side of figure 2.2: 1. Initial permutation (IP - defined in table 2.1) rearranging the bits to form the "permuted input". 2. Followed by 16 iterations of the same function (substitution and permutation). The output of the last iteration consists of 64 bits which is a function of the plaintext and key. The left and right halves are swapped to produce the preoutput. 3. Finally, the preoutput is passed through a permutation (IP−1 - defined in table 2.1) which is simply the inverse of the initial permutation (IP). The output of IP−1 is the 64-bit ciphertext.

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

E/P

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

Inverse Initial 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

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

(a) Input Key

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

(b) Permuted Choice One (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

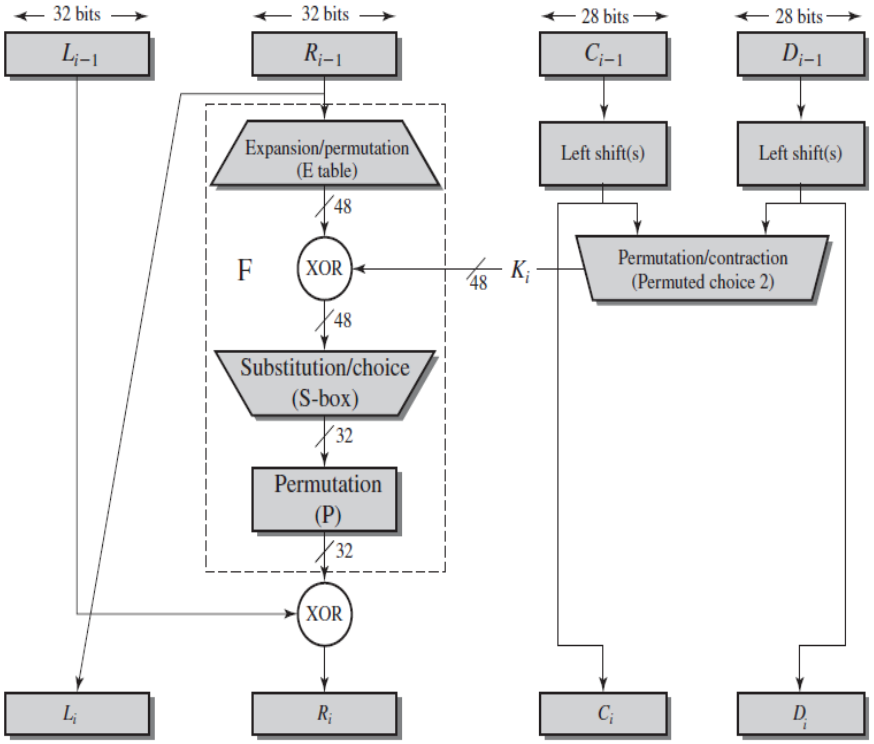
(c) Permuted Choice Two (PC-2)

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

(d) Schedule of Left Shifts

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

As figure 2.2 shows, the inputs to each round consist of the Li , Ri pair and a 48 bit subkey which is a shifted and contracted version of the original 56 bit key. The use of the key can be seen in the right hand portion of figure 2.2: • Initially the key is passed through a permutation function (PC1 - defined in table 2.2) • For each of the 16 iterations, a subkey (Ki) is produced by a combination of a left circular shift and a permutation (PC2 - defined in table 2.2) which is the same for each iteration. However, the resulting subkey is different for each iteration because of repeated shifts.



2.2.2 Details of individual rounds Details of an individual round can be seen in figure 2.3. The main operations on the data are encompassed into what is referred to as the cipher function and is labeled F. This function accepts two different length inputs of 32 bits and 48 bits and outputs a single 32 bit number. Both the data and key are operated on in parallel, however the operations are quite different. The 56 bit key is split into two 28 bit halves Ci and Di (C and D being chosen so as not to be confused with L and R). The value of the key used in any round is simply a left cyclic shift and a permuted contraction of that used in the previous round. Mathematically, this can be written as Ci = Lcsi(Ci−1), Di = Lcsi(Di−1) (2.1) Ki = P C2(Ci , Di) where Lcsi is the left cyclic

shift for round i, Ci and Di are the outputs after the shifts, P C2(.) is a function which permutes and compresses a 56 bit number into a 48 bit number and Ki is the actual key used in round i. The number of shifts is either one or two and is determined by the round number i. For i = {1, 2, 9, 16} the number of shifts is one and for every other round it is two (table 2.2). The common formulas used to describe the relationships between the input to one round and its output (or the input to the next round) are: Li = Ri−1 (2.3) Ri = Li−1 ⊕ F(Ri−1, Ki) (2.4) where L and R have their usual meaning and F(.) is the cipher function. This function F is the main part of every round and consists of four separate stages (see figure 2.4): 1. The E-box expansion permutation - here the 32-bit input data from Ri−1 is expanded and permuted to give the 48 bits necessary for combination with the 48 bit key (defined in table 2.1). The E-box expansion permutation delivers a larger output by splitting its input into 8, 4-bit blocks and copying every first and fourth bit in each block into the output in a defined manner. The security offered by this operation comes from one bit affecting two substitutions in the S-boxes.

This causes the dependency of the output bits on the input bits to spread faster, and is known as the avalanche affect. 2. The bit by bit addition modulo 2 (or exclusive OR) of the E-box output and 48 bit subkey Ki . 3. The S-box substitution - this is a highly important substitution which accepts a 48-bit input and outputs a 32-bit number (defined in table 2.3). The S-boxes are the only non-linear operation in DES and are therefore the most important part of its security. They were very carefully designed although the conditions they were designed under has been under intense scrutiny since DES was released. The reason was because IBM had already designed a set of S-boxes which were completely changed by the NSA with no explanation why7 . The input to the S-boxes is 48 bits long arranged into 8, 6 bit blocks (b1, b2, . . . , b6). There are 8 S-boxes (S1, S2, . . . , S8) each of which accepts one of the 6 bit blocks. The output of each S-box is a four bit number. Each of the S-boxes can be thought of as a 4×16 matrix. Each cell of the matrix is identified by a coordinate pair (i, j), where $0 \leq i \leq 3$ and $0 \leq j \leq 15$. The value of i is taken as the decimal representation of the first and last bits of the input to each S-box, i.e. Dec(b1b6) = i and the value of j is take from the decimal representation of the inner four bits that remain, i.e. Dec(b2b3b4b5) = j. Each cell within the S-box matrices contains a 4-bit number which is output once that particular cell is selected by the input. 4. The P-box permutation - This simply permutes the output of the S-box without changing the size of the data (defined in table 2.1). It is simply a permutation and

nothing else. It has a one to one mapping of its input to its output giving a 32 bit output from a 32 bit input

DES decryption

The decryption process with DES is essentially the same as the encryption process and is as follows: • Use the ciphertext as the input to the DES algorithm but use the keys Ki in reverse order. That is, use K16 on the first iteration, K15 on the second until K1 which is used on the 16th and last iteration. **Avalanche effect** A desirable property of any encryption algorithm is that a small change in either plaintext or key should produce significant changes in the ciphertext.