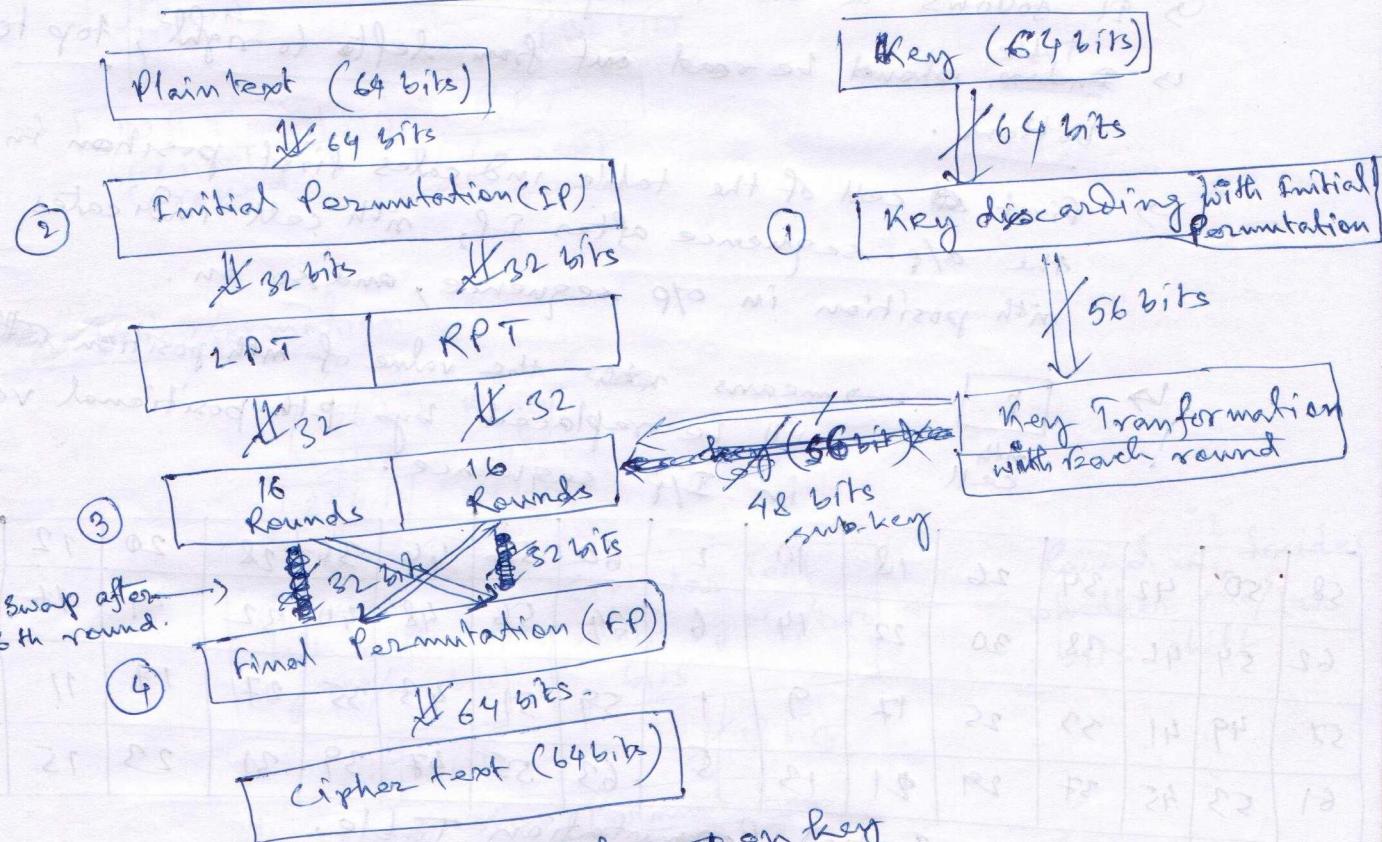


Data Encryption Standard (DES)

Data Encryption Algo (DEA)

- DES ~~was~~ has been a landmark in cryptographic Algorithms.
 - DES has been used as standard for over two decades.
 - DES has been found & vulnerable against very powerful attacks.

Broad Level Steps in DSS



① Key Discarding ~~with Initial Permutation on key
(from 64 bits → to 56 bits)~~

- Initial key consists of 64 bits
 - Then, every 8th bit of the key is discarded to produce 56 bit key. So, 8, 16, 24, 32, 40, 48, 56, 64th bits will be lost and rest of the bits will remain as it is. Then it will

56 bit key. So, 8, 16, 24, 32, 40, 48, 56, 64 will be removed. Rest of the bits will remain as it is. Then it will follow permutation using the below table to get 56 bit key.

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

↳ DES is based on two fundamental attributes:

- ① Substitution (also called confusion)
↳ non-linear substitution (e.g. round function in Feistel Cipher)
- ② Transposition (also called diffusion)
↳ or, permutation repeatedly.

↳ DES consists of 16 rounds in the middle.

↳ Each round performs the steps of substitution & transposition.

↳ It also uses initial & final permutations.

② Initial Permutation (IP)

↳ It runs only once, before any round.

↳ It follows a ~~not~~ transposition table

↳ ~~Table~~ should be read out from left to right, top to bottom.

↳ first ~~cell~~ cell of the table indicates first position in the o/p sequence after IP, ~~nth~~ cell indicates ~~nth~~ position in o/p sequence, and so on.

↳ \boxed{p} ~~means~~ the value of n th position ~~in o/p~~ will be replaced by p th positional value in IP sequence.

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

Initial Permutation Table.

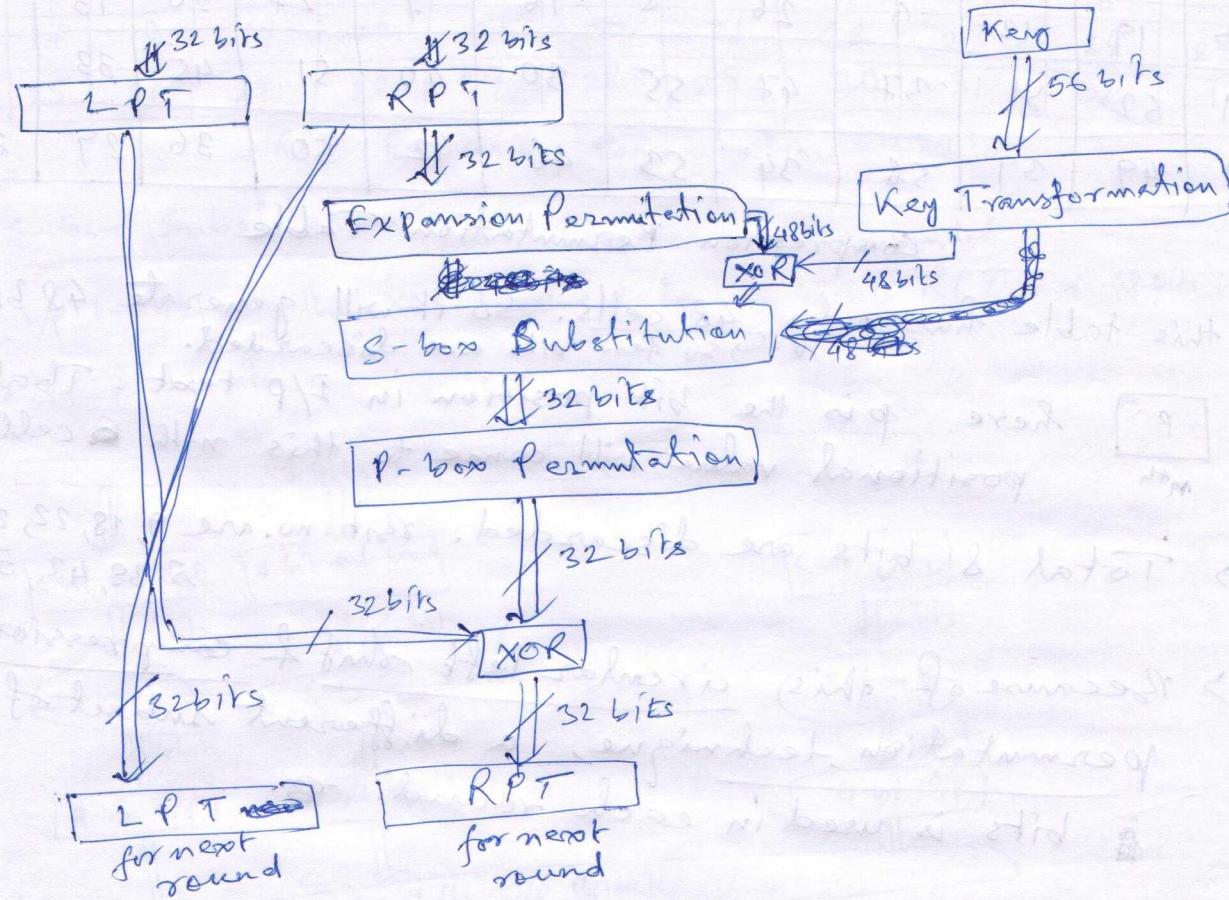
↳ After IP, the resulting 64-bit permuted text block is divided into two half blocks, each contains 32 bits

↳ LPT & RPT

↳ Now, ~~16~~ 16 rounds are performed on these two blocks independently.

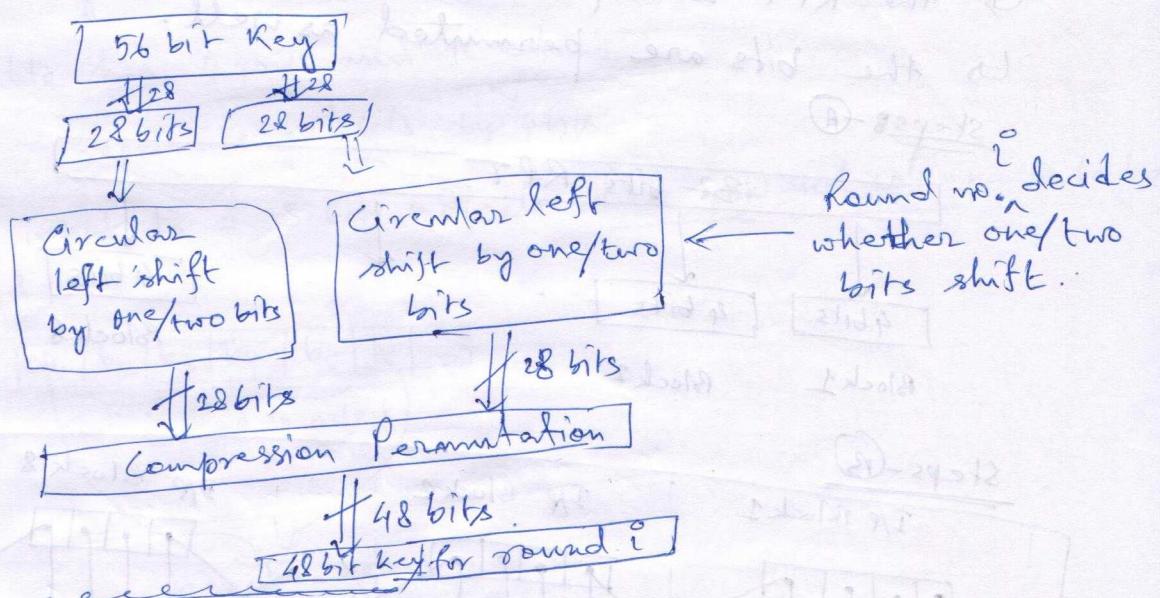
(3)

Details of one Round



3.1

Key Transformation



Number of key bits shifted per round:

Round No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
No. of bits shifted	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

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	58	34	53	46	42	50	36	29	32

Compression Permutation Table

↳ this table has only 48 cells. So it will generate 48 bits key
↳ so few bits are discarded.

↳ here, p is the bit position in I/P text. That m^{th} positional value will come to this n^{th} cell in O/P

↳ Total 8 bits are discarded. Bit no. are 9, 18, 22, 25, 35, 38, 43, 54

→ Because of this circular left shift & compression
permutation technique, a different subset of key
bits is used in each round.

(3-2) Expansion Permutation

→ the RPT is expanded from 32 bits to 48 bits.

↳ the bits are permitted as well.

Steps - A

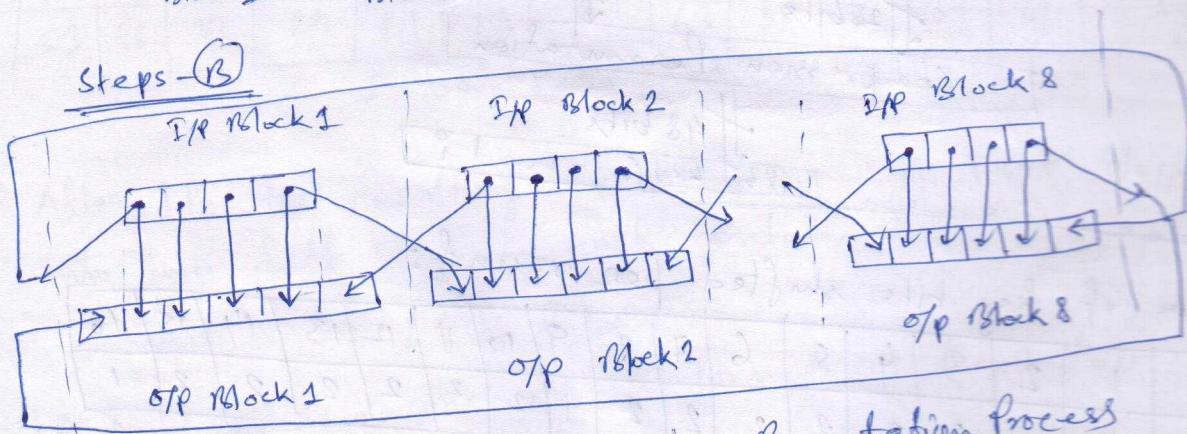
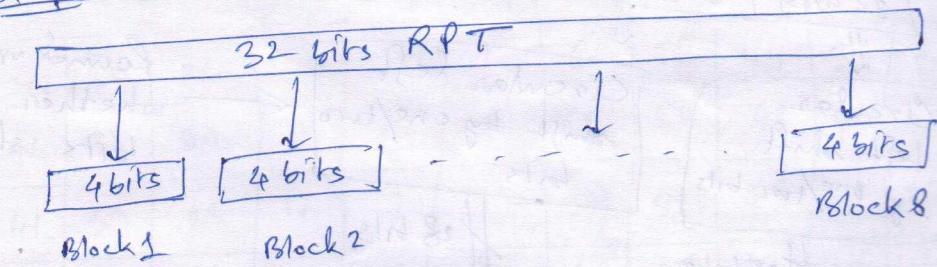


Fig: RPT Expansion Permutation Process

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

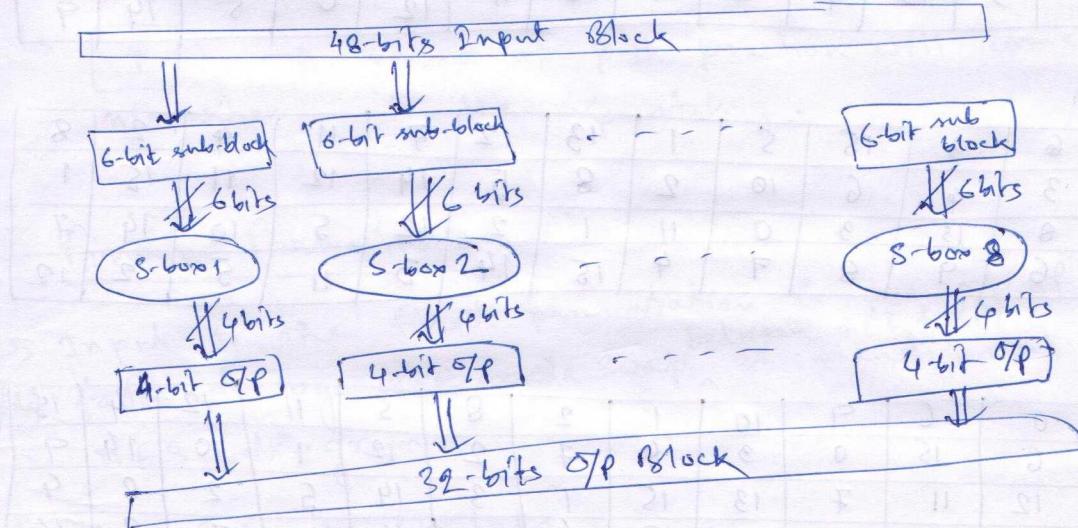
RPT Expansion Permutation Table.

(3.3) S-box Substitution

expanded

↳ S/P to this is the XORed value between 48-bits RPT and 48-bits Key.

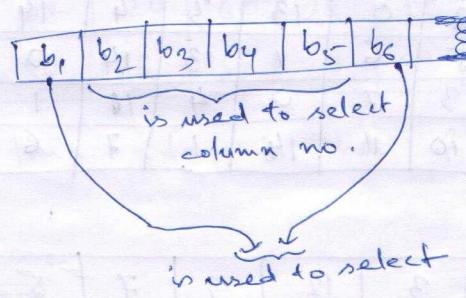
↳ O/P is 32-bits



↳ Corresponding to each S-box, we have one substitution table.

↳ This table has 4 rows and 16 columns.
represented by 2 bits by 4 bits.

↳ 6 bits input to a S-box indicate which row and column to be selected.



↳ Now, the value in the selected cell is p. Here $0 \leq p \leq 15$

↳ So, the 4-bits output is the binary equivalent of p.

Note, ~~these~~ these tables are substitution table, but not the permutation table where the value indicates position.

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

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
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

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

3.4

P-box Permutation

↳ from S-box, total 32-bits comes as I_{fp} to P-box for simple permutation following the P-box permutation table.

↳ No expansion or compression are done here.

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

↳ \boxed{p} → value of p-th position will come to the n-th position in the o_{fp} string.

④ Final Permutation

↳ Input to the final permutation is the o_{fp} of final round (i.e. 16 bits) followed by ~~reverse swap~~ as mentioned in the initial rounds) followed by ~~reverse swap~~ as mentioned in the initial DES diagram.

↳ So, I_{fp} is 64 bits

↳ Final permutation is done only once. Sometimes, it is also called Inverse Initial Permutation (I_{fp}⁻¹).

↳ It is just a simple transposition of bits following the final permutation table.

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

↳ o_{fp} of this step is the final 64-bit ciphertext.

DES Decryption

↳ The same algo used for ~~DES~~ encryption also works for decryption.

↳ Only difference is:

↳ for encryption, K is divided into K₁, K₂, K₃...K₁₆ for the 16 rounds.

↳ for decryption, same sub-keys will be used in reverse order, i.e. the sequence is K₁₆, K₁₅, ..., K₁ for 16 rounds.

Analysis About DES

① Key length

↳ DES uses 56-bit keys.

↳ Thus, 2^{56} possible keys $\approx 7.2 \times 10^{16}$ key need to check.

↳ Thus, it seems that brute-force attack is impractical.

Some estimate

↳ A single computer performing one DES encryption per microsecond would require more than 1000 years to break DES.

↳ Another study: A single PC with multicore processor can perform 10^9 (≈ 1 billion) key combinations per second.

↳ Needs ≈ 1.125 years to break DES.

↳ A contemporary supercomputer can do 10^{13} encryptions per second.

↳ Needs ≈ 1 hour to break DES.

② Avalanche Effect

↳ If we make minor changes to the plaintext or key, then if we get significant changes in ciphertext, it is called avalanche effect.

↳ DES has strong Avalanche effect.

③ Completeness Effect

↳ means, each ciphertext bit should depend on more than one plaintext bits.

↳ DES has high completeness effect because of S-box & P-box.

Variations of DES

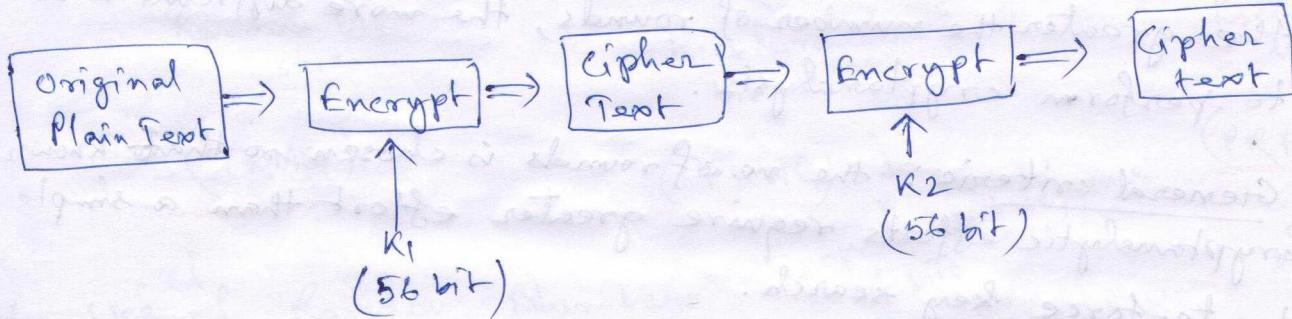
① Double DES

② Triple DES

with two keys

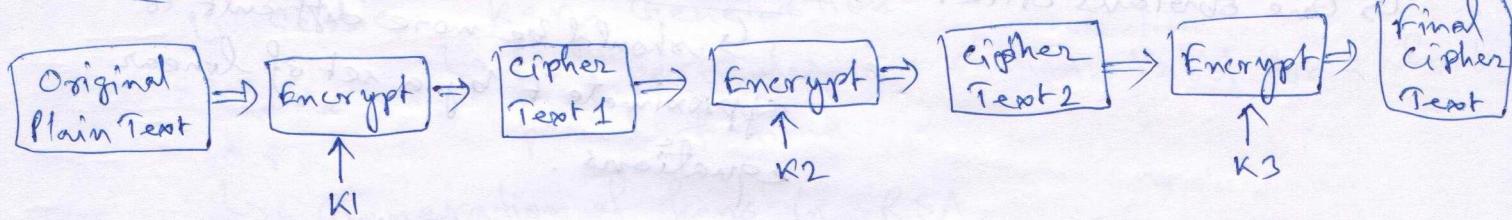
with three keys.

Double DES Encryption



↳ In decryption, keys are used in reverse order. i.e., K_2, K_1

Triple DES with three keys

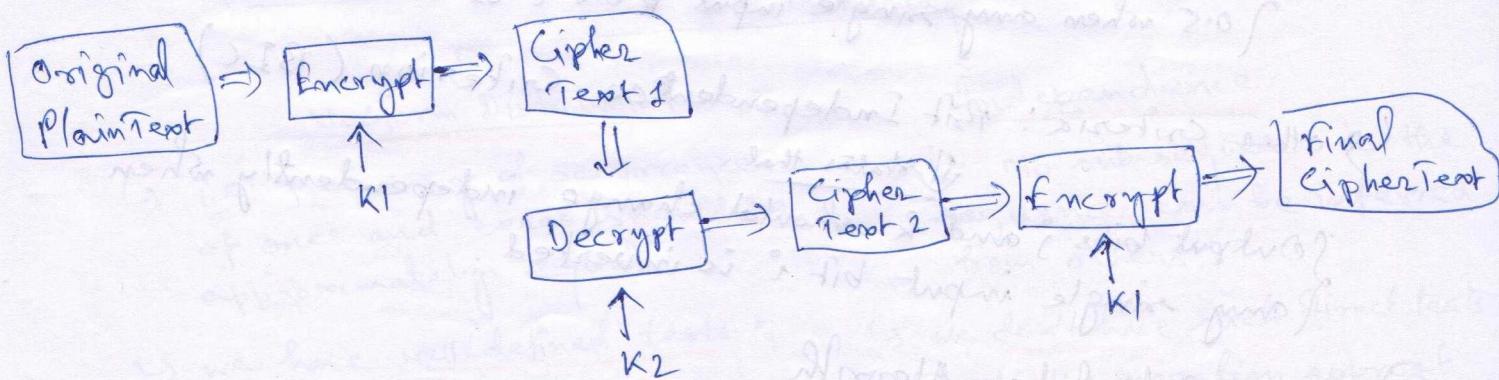


↳ Decryption: $P = D_{K_1}(D_{K_2}(D_{K_3}(c)))$

↳ Dis-advantage: requiring $56 \times 3 = 168$ bits for key

↳ Many Internet-based application adopted this 3DES.

Triple DES with two keys e.g. PGP, S/MIME



↳ Decryption: $P = D_{K_1}(E_{K_2}(D_{K_1}(c)))$

Block Cipher Design Principles

↳ Three critical aspects of block cipher design

① → the number of rounds

② → design of the function F

③ → key scheduling

① Number of Rounds

↳ the greater the number of rounds, the more difficult it is to perform cryptanalysis.

↳ General criteria: the no. of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search.

② Design of F

↳ The heart of a Feistel Block cipher is this F

↳ One obvious criterion: F be non-linear

↳ should be more difficult to approximate F by a set of linear equations.

↳ other criteria: Good Avalanche properties.

↳ More strict version: Strict Avalanche Criterion (SAC)

↳ states that

{ Any output bit j of an S-box should change with probability 0.5 when any single input bit i is inverted.

↳ other criteria: Bit Independence Criterion (BIC)

↳ states that

{ Output bits j and k should change independently when any single input bit i is inverted.

③ Key Schedule Algorithm

↳ No general criteria yet.

↳ one suggestion: Key schedule should guarantee SAC and BIC.