16/02/2021

Lecture No. 12

Feistel Cipher & Data Encryption Standard (DES)

Feistel Cipher is not a specific scheme of block cipher. It is a design model from which many different block ciphers are derived. DES is just one example of a Feistel Cipher. A cryptographic system based on Feistel cipher structure uses the same algorithm for both encryption and decryption.

Encryption Process:

The encryption process uses the Feistel structure consisting multiple rounds of processing of the plaintext, each round consisting of a "substitution" step followed by a permutation step.

The input block to each round is divided into two halves that can be denoted as L and R for the left half and the right half.

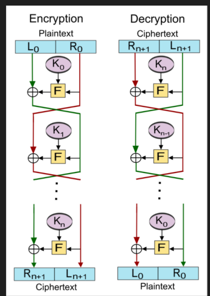
In each round, the right half of the block, R, goes through unchanged. But the left half, L, goes through an operation that depends on R and the encryption key. First, we apply an encrypting function 'f' that takes two input – the key K and R. The function produces the output f(R,K). Then, we XOR the output of the mathematical function with L.

In real implementation of the Feistel Cipher, such as DES, instead of using the whole encryption key during each round, a round-dependent key (a subkey) is derived from the encryption key. This means that each round uses a different key, although all these subkeys are related to the original key.

The permutation step at the end of each round swaps the modified L and unmodified R. Therefore, the L for the next round would be R of the current round. And R for the next round be the output L of the current round.

Above substitution and permutation steps form a 'round'. The number of rounds are specified by the algorithm design.

Once the last round is completed then the two sub blocks, 'R' and 'L' are concatenated in this order to form the ciphertext block.



Decryption Process:

The process of decryption in Feistel cipher is almost similar. Instead of starting with a block of plaintext, the ciphertext block is fed into the start of the Feistel structure and then the process thereafter is exactly the same as described in the given illustration.

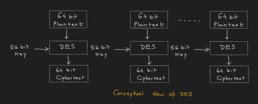
The process is said to be almost similar and not exactly same. In the case of decryption, the only difference is that the subkeys used in encryption are used in the reverse order.

The final swapping of 'L' and 'R' in last step of the Feistel Cipher is essential. If these are not swapped then the resulting ciphertext could not be decrypted using the same algorithm.

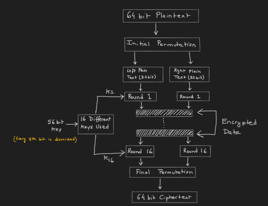
Number of Rounds:

The number of rounds used in a Feistel Cipher depends on desired security from the system. More number of rounds provide more secure system. But at the same time, more rounds mean the inefficient slow encryption and decryption processes. Number of rounds in the systems thus depend upon efficiency-security tradeoff.

Data Encryption Standard

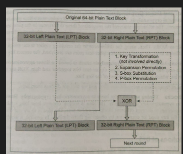


Conceptual View of DES



Detail Steps in DES

1. Divide Plaintext message into block of size 64-bits each.
2. Subject it to Initial Permutation
3. Divide the plaintext block into 2 equal halves
4. The LPT & RPT goes through 16 rounds of encryption process along with 16 different keys for each rounds.
5. After 16 rounds, left plaintext & right plaintext gets combined & final permutation is performed on these combined blocks.
6. The result of final permutation produces 64-bit of ciphertext.



Detailed Explanation of DES

1. Initial Permutation :

The process of rearranging or shuffling each bit of original plaintext block with any other random bit of same plaintext message block.

48	57	59	63	8	9	43	64
47	42	10	6	4	11	13	18
20	5	16	19	32	1	25	27
2	3	23	29	30	31	34	7

After initial permutation, the 64 bit plaintext block get divided into two halves LPT & RPT.

2. Internal Rounds

Step 1: Key Transformation:

For each round, a 56 bit key is available. From this 56-bit key, a different 48 bit sub key is generated during each round using a process called as Key Transformation

* For this, the 56 bit key is divided into 2 halves each of 28 bits each.
* These halves are circularly shifted LEFT by one or two positions, depending on the round.

Round No	Shift
1, 2, 9, 16	1 position
All others	2 positions

* After an appropriate shift, 48 of the 56 bits are selected using the following table

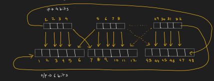
14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

* Because of this compression permutation, different subset of key bits are used in each round. That makes DES difficult to crack.

Step 2: Expansion Permutation

Recall, that after Initial permutation, we had both LPT & RPT of size 32 bits. So we need to expand the RPT to 48 bits for further operations.

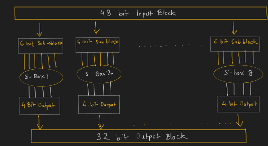
* Besides, we also permute the positions of the bits, hence Expansion Permutation
* The 32 bit RPT is divided into 8 blocks, with each block consisting of 4 bits.
* Each 4-bit block of the above step is then expanded to a corresponding 6-bit block as follows.



* Now, this 48 bit RPT is XOR'ed with the 48 bit key.
* The resultant output is given to the next step.

Step 3: S-box Substitution

* This step makes sure that we get the result in 32 bits or else we won't be able to XOR it with the LPT.
* We convert the output (48 bits) of the previous step & convert that into 32 bits after a series of substitutions.
* The substitution is performed by 8 S-boxes. Each of these 8 S-boxes has a 6-bit input & 4-bit output. The 48 bit i/p is divided into 8 sub-blocks, and each sub-block is given to a S-box. The S-box transforms it into a 4 bit o/p.



* We can think of every S-Box as a table that has 4 rows and 16 columns. We have 8 such S-Boxes.
* At the intersection of every row & column, a 4-bit no. is present.
* Bit No. 1 & 6 → 2 bit Row No.
  Bit No 2 3 4 5 → 4 bit Column No.

For example, Consider the following 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

* Suppose, the bits 101101 are inputs to this S-box 1, then bits 11 will be used to select the Row No.
* Similarly bits 0110 will be used to select the Column Number.
* Hence Row ⇒ 3
  Column ⇒ 6
* Hence I/p ⇒ 101101 i.e 45 will be replaced by 1.
* Similarly, the output of each S-Box is then combined to form a 32-bit block.

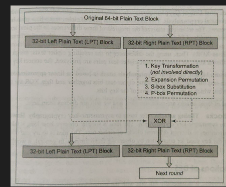
Step 4: P-Box Permutation

This is a straight forward permutation mechanism involving simple permutation of each bit with another bit, according to a P-Box Table.

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

Step 5: XOR & Swap

* All the operations that we have done so far was made on the 32-bit RPT. The LPT was left untouched so far.
* Now, the 32 bit LPT will be XOR'ed with the result of step 4, which now becomes the new RPT for the next round.



XOR & Swap.

3. Final Permutation

At the end of the 16 rounds, the final permutation is performed (only once). This is a simple transposition. The output of this step yields the 64-bit Ciphertext.

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