CS 413
Information Security

Course Instructor

Naushad Siddiqui

Head of IT Audit, Sui Southern Gas Company

Visiting Faculty, NED University

Director & Faculty Member, ISACA Karachi

CS413 Information Security

Fall 2020

Week 07

Agenda

- Modern Block Ciphers
- Feistel Cipher Structure
- Data Encryption Standard (DES)
- Simplified DES (S-DES)

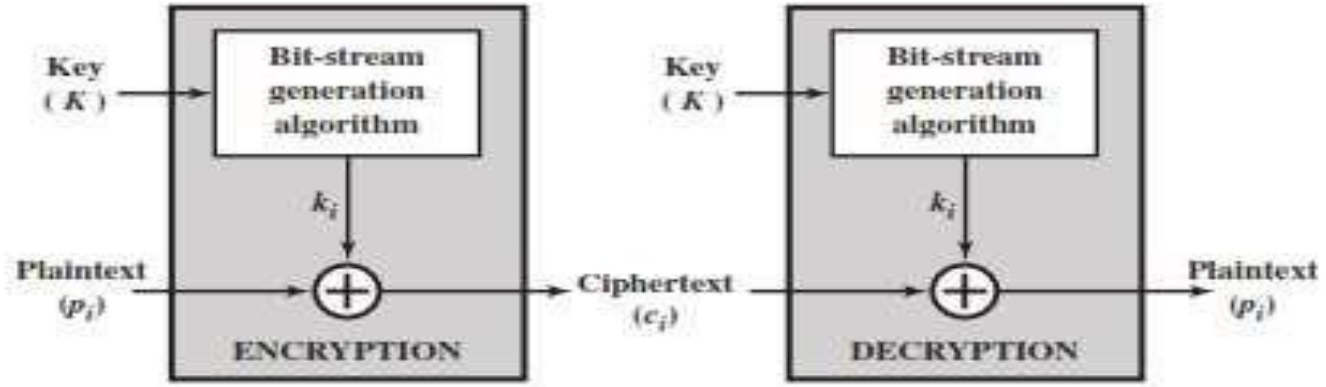
Introduction

- A block cipher is an encryption/decryption scheme in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length.
- Many block ciphers have a Feistel structure. Such a structure consists of a number of identical rounds of processing.
- In each round, a substitution is performed on one half of the data being processed, followed by a permutation that interchanges the two halves.
- The original key is expanded so that a different key is used for each round.
- The Data Encryption Standard (DES) has been the most widely used encryption algorithm until recently. It exhibits the classic Feistel structure.
- DES uses a 64-bit block and a 56-bit key.

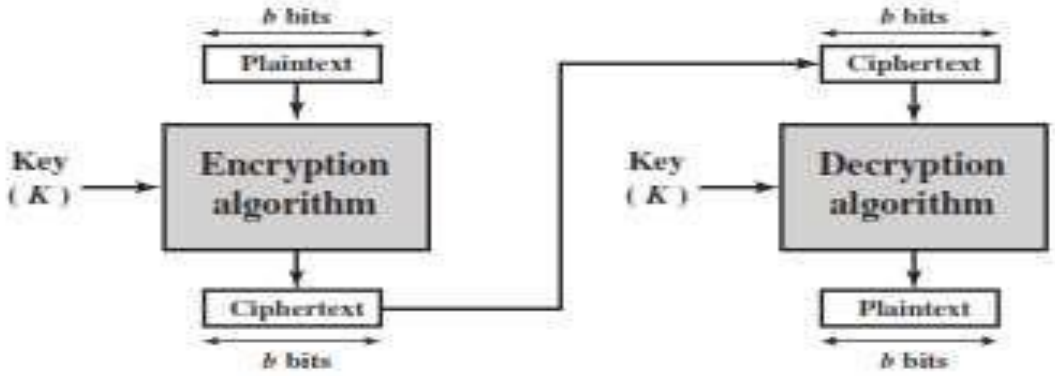
Stream Ciphers and Block Ciphers

- A **stream cipher is one that encrypts a digital data stream one bit or one byte at a time.**
- **Examples of** classical stream ciphers are the autokeyed Vigenère cipher and the Vernam cipher.
- A **block cipher is** one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length.
- Typically, a block size of 64 or 128 bits is used.
- As with a stream cipher, the two users share a **symmetric encryption key**

Stream Ciphers and Block Ciphers



(a) Stream cipher using algorithmic bit-stream generator



(b) Block cipher

Feistel Cipher Structure

- Feistel proposed that we can approximate the ideal block cipher by utilizing the concept of a product cipher, which is the execution of two or more simple ciphers in sequence in such a way that the final result or product is cryptographically stronger than any of the component ciphers.
- The essence of the approach is to develop a block cipher with a key length of *k bits and a block length of n bits, allowing a total of* 2^k possible transformations, rather than the $2^n!$ transformations available with the ideal block cipher.

Feistel Cipher Structure

- In particular, Feistel proposed the use of a cipher that alternates substitutions and permutations, where these terms are defined as follows:
- **Substitution: Each plaintext element or group of elements is uniquely replaced** by a corresponding ciphertext element or group of elements.
- **Permutation: A sequence of plaintext elements is replaced by a permutation** of that sequence. That is, no elements are added or deleted or replaced in the sequence, rather the order in which the elements appear in the sequence is changed.

Feistel Cipher Structure

Diffusion and Confusion

- Confusion means that each binary digit (bit) of the ciphertext should depend on several parts of the key.
- Diffusion means that if we change a single bit of the plaintext, then (statistically) half of the bits in the ciphertext should change, and similarly, if we change one bit of the ciphertext, then approximately one half of the plaintext bits should change.

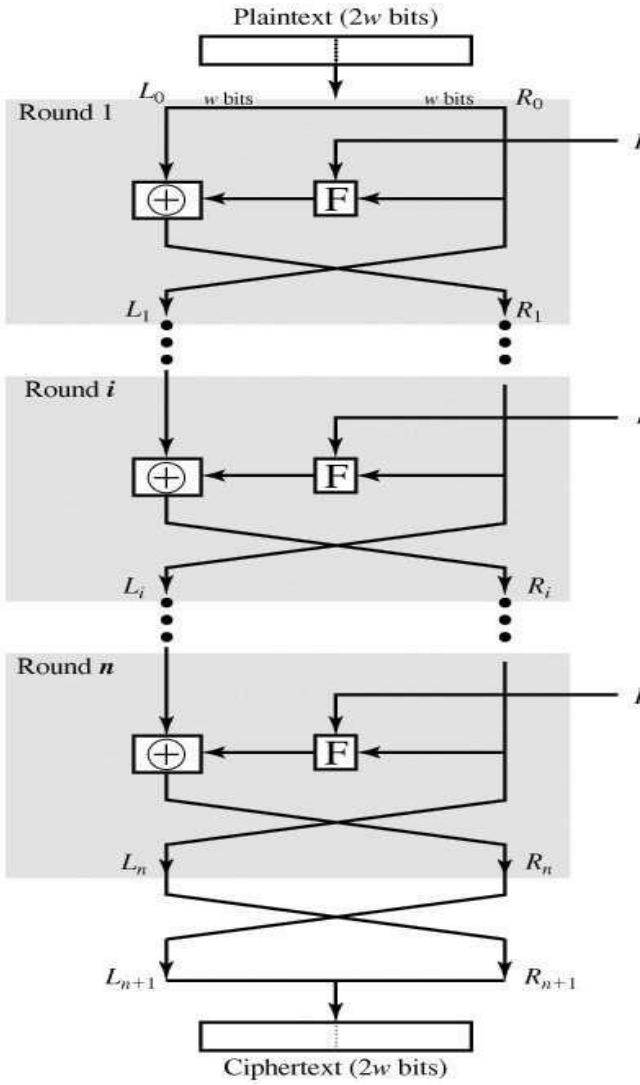
Feistel Cipher Structure

Diffusion and Confusion

- The terms diffusion and confusion were introduced by Claude Shannon to capture the two basic building blocks for any cryptographic system.
- Shannon's concern was to thwart cryptanalysis based on statistical analysis.
- Every block cipher involves a transformation of a block of plaintext into a block of ciphertext, where transformation depends on the key.
- Diffusion seeks to make the statistical relationship between the plaintext and ciphertext as complex as possible in order to thwart attempts to deduce the key.
- Confusion seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible, again to thwart attempts to discover the key.
- So successful are diffusion and confusion in capturing the essence of the desired attributes of a block cipher that they have become the cornerstone of modern block cipher design.

Feistel Cipher Structure

- The inputs to the encryption algorithm are a plaintext block of length 2w bits and a key K.
- The plaintext block is divided into two halves, L_0 and R_0.
- The two halves of the data pass through 'n' rounds of processing and then combine to produce the ciphertext block.
- Each round i has as inputs L_{i-1} and R_{i-1}, derived from the previous round, as well as a subkey K_i, derived from the overall K.
- In general, the subkeys K_i are different from K and from each other.



Feistel Cipher Structure

- All rounds have the same structure.
- A **substitution** is performed on the left half of the data. This is done by applying a *round function* F to the right half of the data and then taking the exclusive-OR of the output of that function and the left half of the data.
- The round function has the same general structure for each round but is parameterized by the round subkey K_i.
- Following this substitution, a **permutation** is performed that consists of the interchange of the two halves of the data.

Feistel Cipher Structure

The exact realization of a Feistel network depends on the choice of the following parameters and design features:

- Block size: Larger block sizes mean greater security but reduced encryption/decryption speed for a given algorithm. The greater security is achieved by greater diffusion. Traditionally, a block size of 64 bits has been considered a reasonable tradeoff and was nearly universal in block cipher design. However, the new AES uses a 128-bit block size.
- Number of rounds: The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security. A typical size is 16 rounds.
- Subkey generation algorithm: Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis.
- Round function: Again, greater complexity generally means greater resistance to cryptanalysis.

Feistel Cipher Structure

There are two other considerations in the design of a Feistel cipher:

- Fast software encryption/decryption: The speed of execution of the algorithm becomes a concern.
- Ease of analysis: Although we would like to make our algorithm as difficult as possible to cryptanalyze, there is great benefit in making the algorithm easy to analyze.
- That is, if the algorithm can be concisely and clearly explained, it is easier to analyze that algorithm for cryptanalytic vulnerabilities and therefore develop a higher level of assurance as to its strength. DES, for example, does not have an easily analyzed functionality.

Feistel Decryption Algorithm

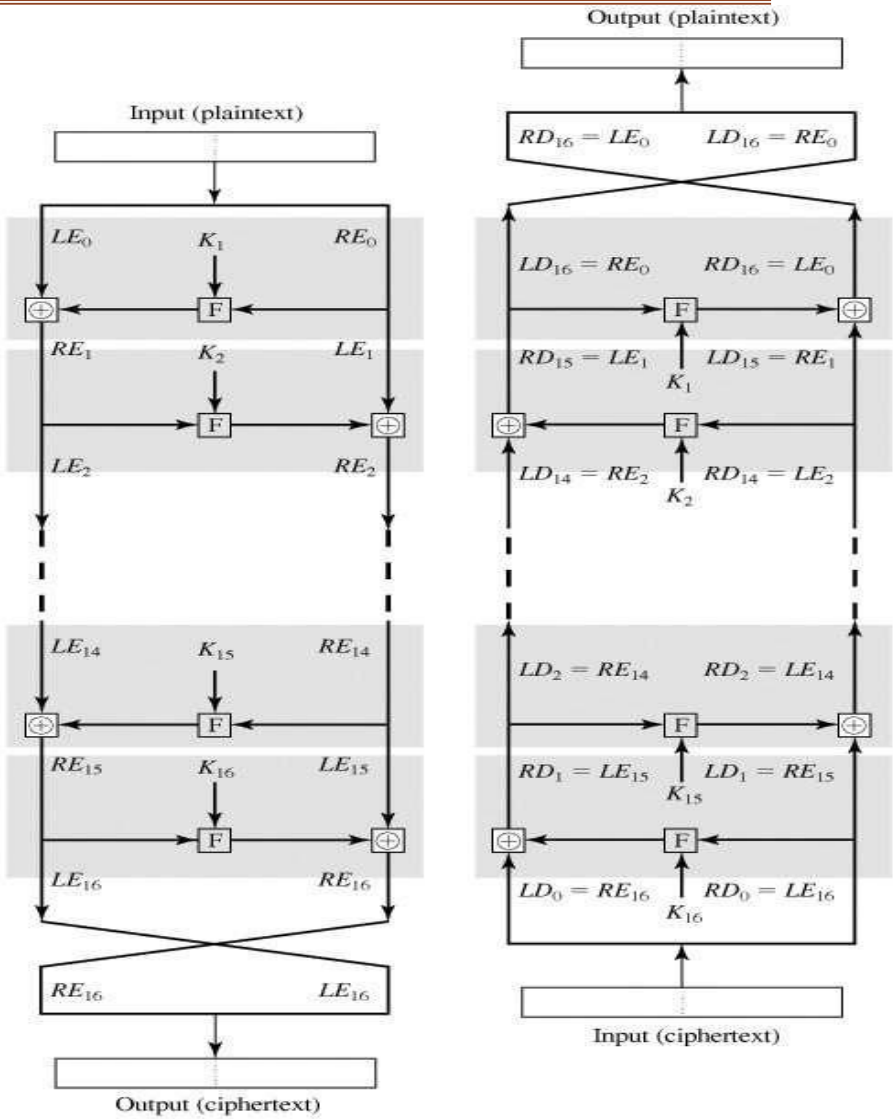
The process of decryption with a Feistel cipher is essentially the same as the encryption process.

The rule is as follows:

- Use the ciphertext as input to the algorithm, but use the subkeys K in reverse order.
- That is, use K_n in the first round, K_{n-1} in the second round, and so on until K is used in the last round. This is a nice feature because it means we need not implement two different algorithms, one for encryption and one for decryption.

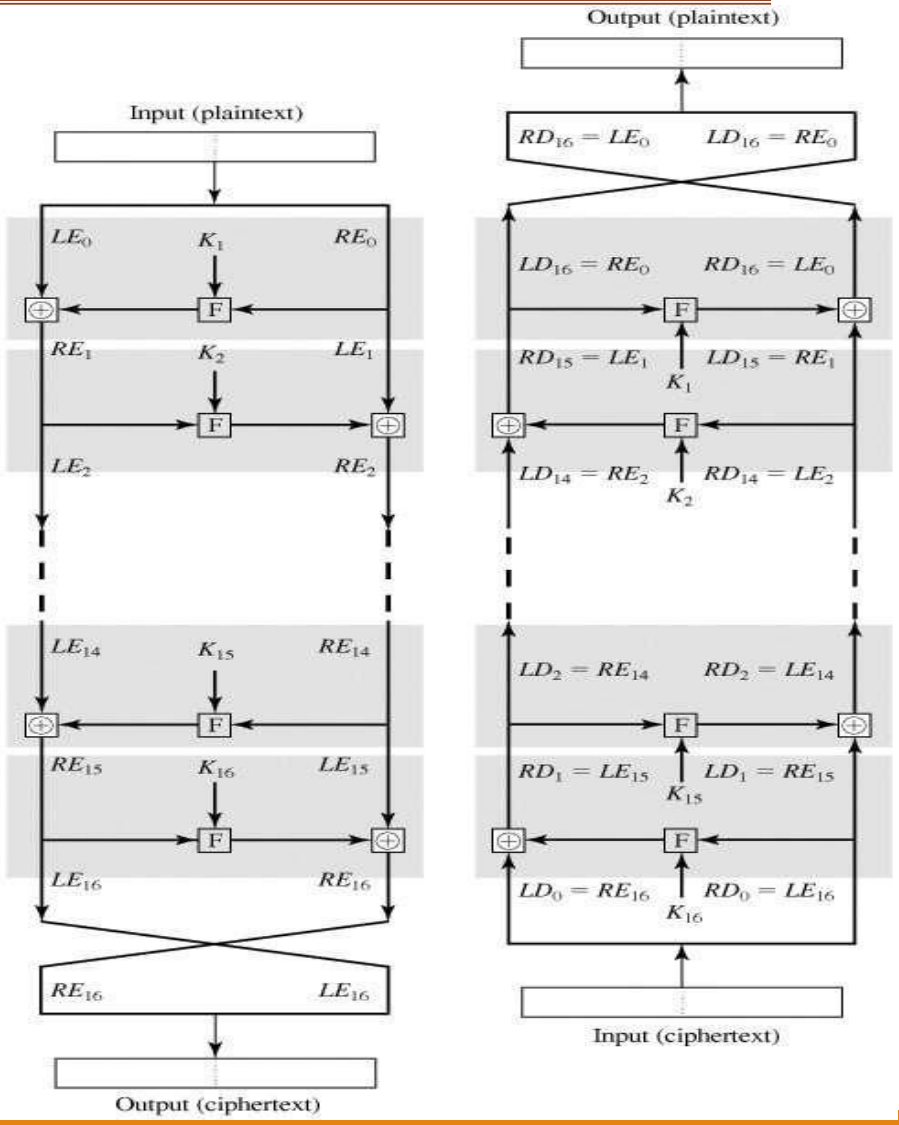
Feistel Decryption Algorithm

- Same algorithm with a reversed key order produces the correct result, which shows the encryption process going down the left-hand side and the decryption process going up the right-hand side for a 16-round algorithm.
- For clarity, we use the notation LE_i and RE_i for data traveling through the encryption algorithm and LD_i and RD_i for data traveling through the decryption algorithm.



Feistel Decryption Algorithm

- The diagram indicates that, at every round, the intermediate value of the decryption process is equal to the corresponding value of the encryption process with the two halves of the value swapped.
- After the last iteration of the encryption process, the two halves of the output are swapped, so that the ciphertext is $RE_{16}||LE_{16}$.
- The output of that round is the ciphertext. Now take that ciphertext and use it as input to the same algorithm. The input to the first round is $RE_{16}||LE_{16}$, which is equal to the 32-bit swap of the output of the sixteenth round of the encryption process.



Feistel Decryption Algorithm

- If you clearly observe that the output of the first round of the decryption process is equal to a 32-bit swap of the input to the sixteenth round of the encryption process. First, consider the encryption process.

$$LE_{16} = RE_{15}$$
$$RE_{16} = LE_{15} \text{ x } F(RE_{15}, K_{16})$$

On the decryption side, $LD_1 =$

$RD_0 = LE_{16} = RE_{15}$ $RD_1 = LD_0$ x

$F(RD_0, K_{16})$

$= RE_{16}$ x $F(RE_{15}, K_{16})$

$= [LE_{15}$ x $F(RE_{15}, K_{16})]$ x $F(RE_{15}, K_{16})$

Data Encryption Standard (DES)

- The most widely used encryption scheme is based on the Data Encryption Standard
- (DES) adopted in 1977 by the National Institute of Standards and Technology (NIST).
- The algorithm itself is referred to as the Data Encryption Algorithm (DEA).
- For DES, data are encrypted in 64-bit blocks using a 56-bit key. The algorithm transforms 64-bit input in a series of steps into a 64-bit output
- The same steps, with the same key, are used to reverse the encryption.

DES History

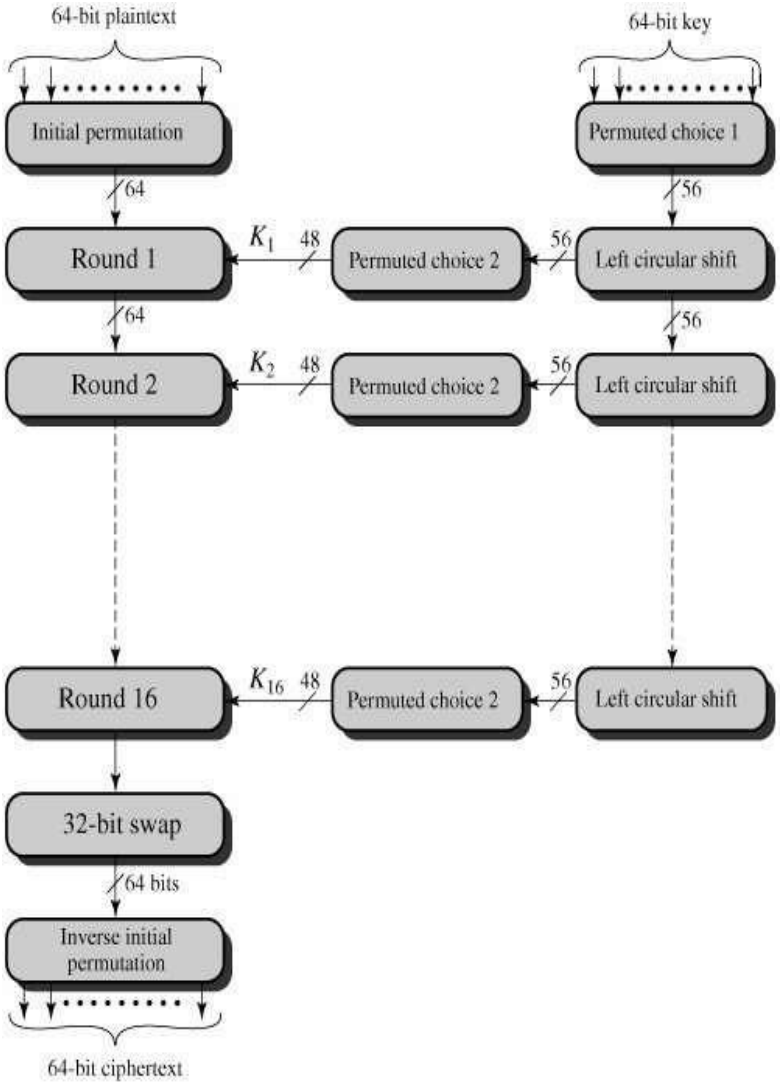
- In the late 1960s, IBM set up a research project in computer cryptography led by Horst Feistel.
- The project concluded in 1971 with the development of the LUCIFER algorithm. LUCIFER is a Feistel block cipher that operates on blocks of 64 bits, using a key size of 128 bits.
- IBM embarked on an effort, headed by Walter Tuchman and Carl Meyer, to develop a marketable commercial encryption product that ideally could be implemented on a single chip.
- It involved not only IBM researchers but also outside consultants and technical advice from NSA.
- The outcome of this effort was a refined version of LUCIFER that was more resistant to cryptanalysis but that had a reduced key size of 56 bits, to fit on a single chip.
- In 1973, the National Bureau of Standards (NBS) issued a request for proposals for a national cipher standard.
- IBM submitted the modified LUCIFER. It was by far the best algorithm proposed and was adopted in 1977 as the Data Encryption Standard.

DES Design Controversies

- Before its adoption as a standard, the proposed DES was subjected to intense & continuing criticism over the size of its key & the classified design criteria.
- Recent analysis has shown despite this controversy, that DES is well designed.
- DES is theoretically broken using Differential or Linear Cryptanalysis but in practice is unlikely to be a problem yet.
- Also rapid advances in computing speed though have rendered the 56 bit key susceptible to exhaustive key search, as predicted by Diffie & Hellman.
- DES has flourished and is widely used, especially in financial applications.
- It is still standardized for legacy systems, with either AES or triple DES for new applications.
- 3DES still strong (112 bit key)

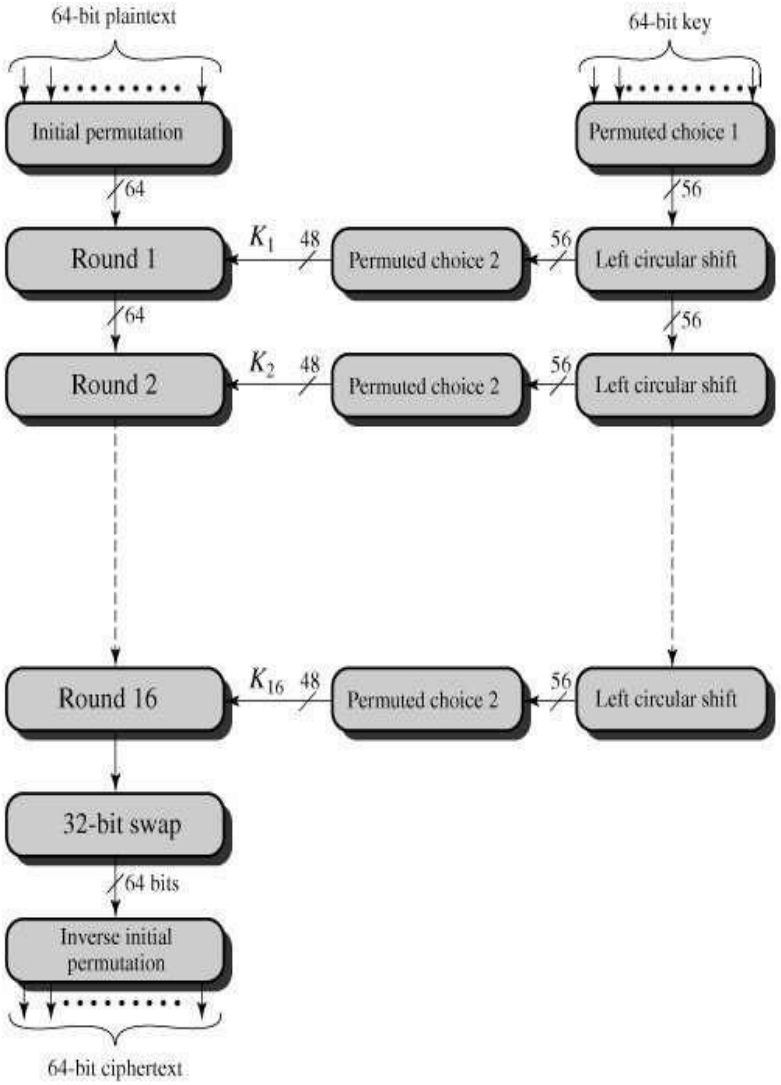
DES Encryption

- As with any encryption scheme, there are two inputs to the encryption function: the plaintext to be encrypted and the key. In this case, the plaintext must be 64 bits in length and the key is 56 bits in length.
- Actually, the function expects a 64-bit key as input. However, only 56 of these bits are ever used; the other 8 bits can be used as parity bits or simply set arbitrarily.
- We can see that the processing of the plaintext proceeds in three phases.
  - First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the ***permuted input***.



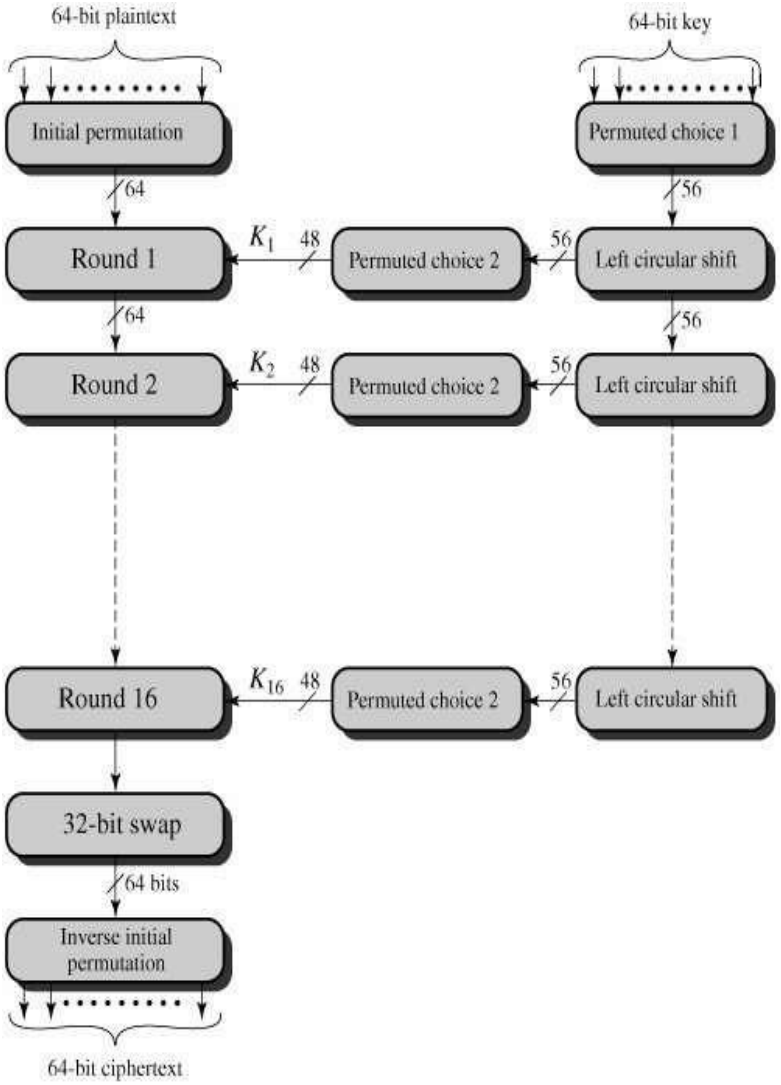
DES Encryption

- This is followed by a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions.
- The output of the last (sixteenth) round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the **preoutput.**
- Finally, the preoutput is passed through a permutation (IP^{-1}) that is the inverse of the initial permutation function, to produce the 64-bit ciphertext. With the exception of the initial and final permutations, DES has the exact structure of a Feistel cipher.



DES Encryption

- The right-hand portion of Figure shows the way in which the 56- bit key is used.
- Initially, the key is passed through a permutation function.
- Then, for each of the 16 rounds, a *subkey (K)* is produced by the combination of a left circular shift and a permutation.
- The permutation function is the same for each round, but a different subkey is produced because of the repeated shifts of the key bits.



DES Decryption

- As with any Feistel cipher, decryption uses the same algorithm as encryption, except that the application of the subkeys is reversed.
- Additionally, the initial and final permutations are reversed.

Block Cipher Principles

- Most symmetric block encryption algorithms in current use are based on a structure referred to as a Feistel block cipher.
- A block cipher operates on a plaintext block of ‘n’ bits to produce a ciphertext block of ‘n’ bits.
- From an implementation and performance point of view, however, an arbitrary reversible substitution cipher for a large block size is not practical.
- In general, for an n-bit general substitution block cipher, the size of the key is $n \times 2^n$.
- For a 64-bit block, which is a desirable length to thwart statistical attacks, the key size is $64 \times 2^{64} = 2^{70} = 10^{21}$
- In considering these difficulties, Feistel points out that what is needed is an approximation to the ideal block cipher system for large ‘n’, built up out of components that are easily realizable.

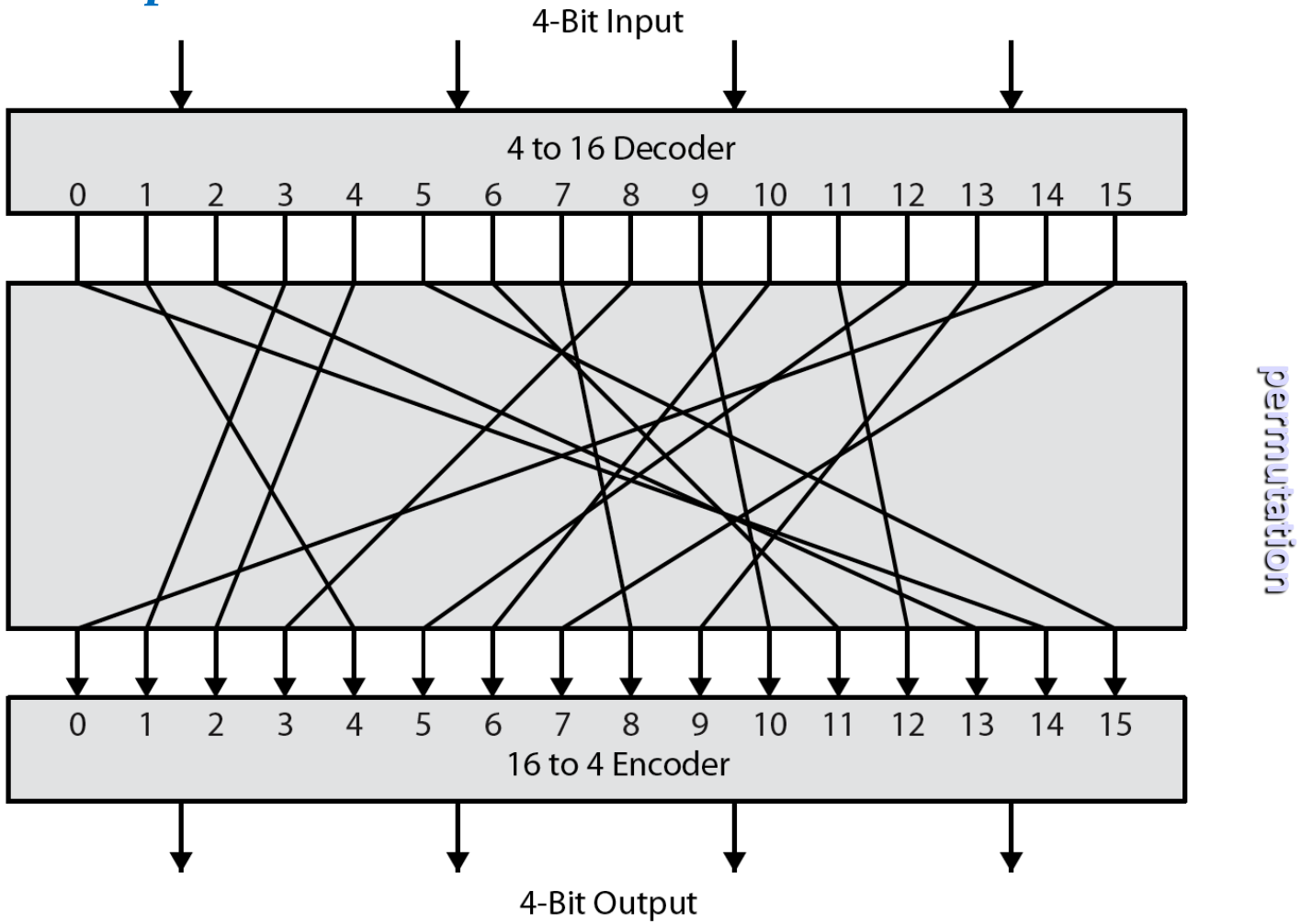
Block Cipher Principles

Ideal Block Cipher

- Feistel refers to an n-bit general substitution as an ideal block cipher, because it allows for the maximum number of possible encryption mappings from the plaintext to ciphertext block.
- A 4-bit input produces one of 16 possible input states, which is mapped by the substitution cipher into a unique one of 16 possible output states, each of which is represented by 4 ciphertext bits.
- The encryption and decryption mappings can be defined by a tabulation
- It illustrates a tiny 4-bit substitution to show that each possible input can be arbitrarily mapped to any output - which is why its complexity grows so rapidly.

Block Cipher Principles

Ideal Block Cipher



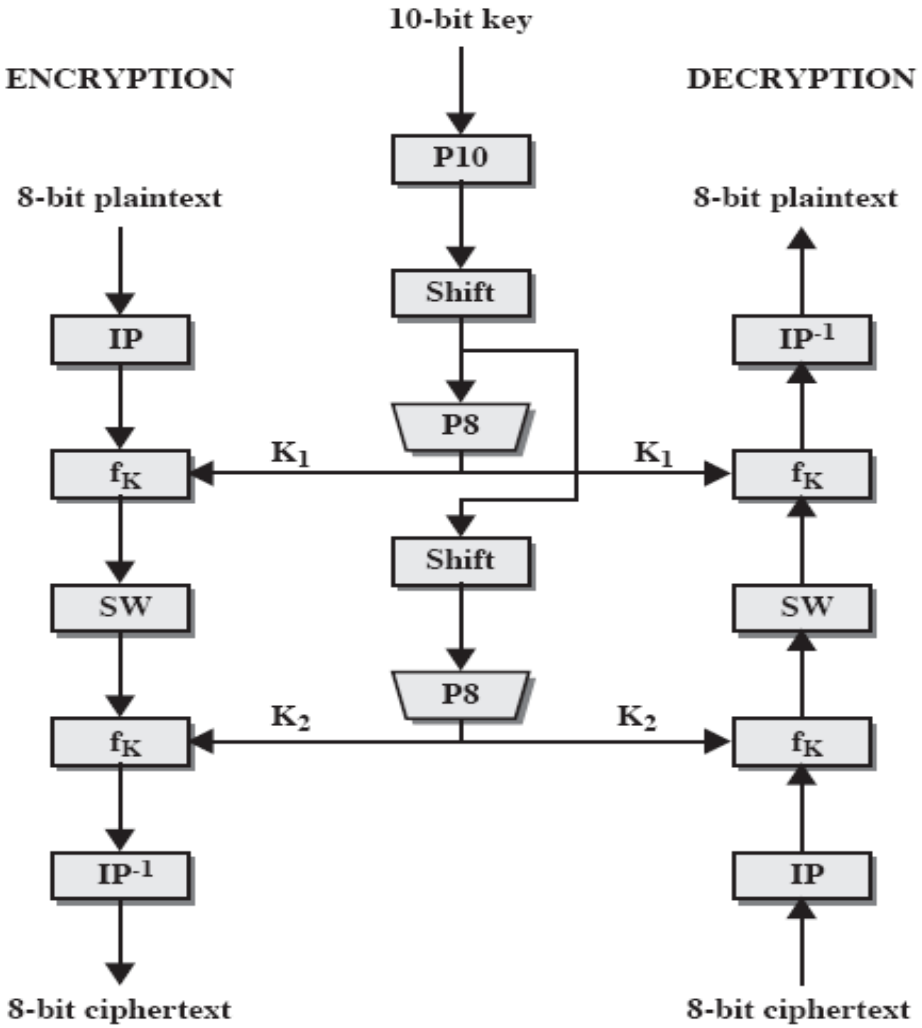
Claude Shannon and Substitution-Permutation Ciphers

- Claude Shannon introduced idea of substitution-permutation (S-P) networks in 1949 paper
- It formed the basis of modern block ciphers
- S-P nets are based on the two primitive cryptographic operations seen before:
  - substitution (S-box)
  - permutation (P-box)
- Provides confusion & diffusion of message & key

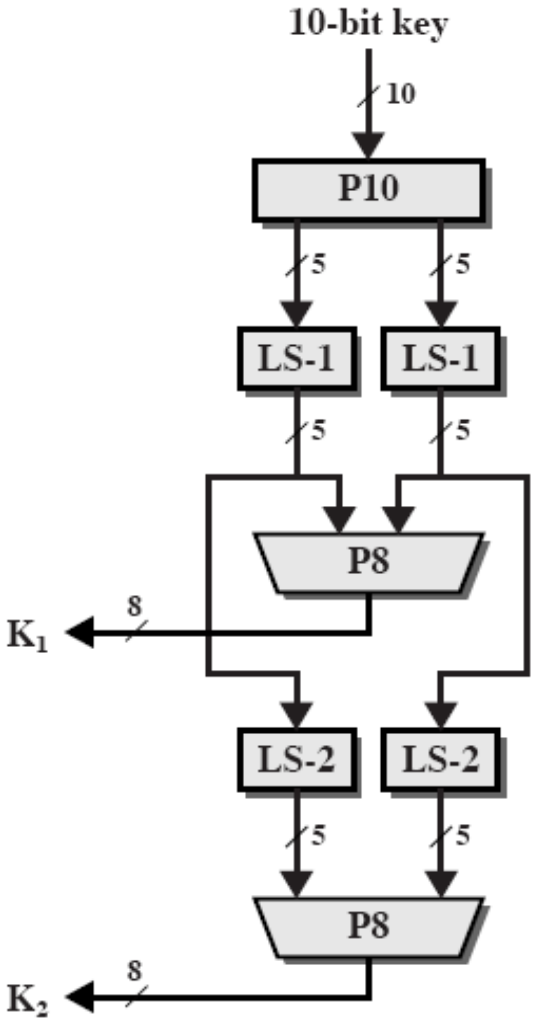
Simplified DES (S-DES)

- Developed by Prof. Edward Schaefer of Santa Clara University 1996.
- Takes 8 bit block of plain text and 10 bit key as input and produce an 8 bit block cipher text output.
- The encryption algorithm involves 5 functions:
  - initial permutation (IP)
  - a complex function f_k which involves substitution and permutation depends on the key
  - simple permutation function (switch) SW
  - the function f_k again
  - final inverse of the initial permutation(IP^{-1})

Simplified DES Scheme



Key Generation for S-DES



Key Generation for S-DES

First permute the key in the following way:

P_{10}									
3	5	2	7	4	10	1	9	8	6

Ex: (1010000010) is permuted to (1000001100)

Perform a circular left shift to each bits of the key:

Ex: (10000 | 01100)⇒(00001 | 11000)

Next apply P_8

P_8							
6	3	7	4	8	5	10	9

This yields K_1=(10100100)

Key Generation for S-DES

Then perform again 2 bit circular shift left on each of the five bits:

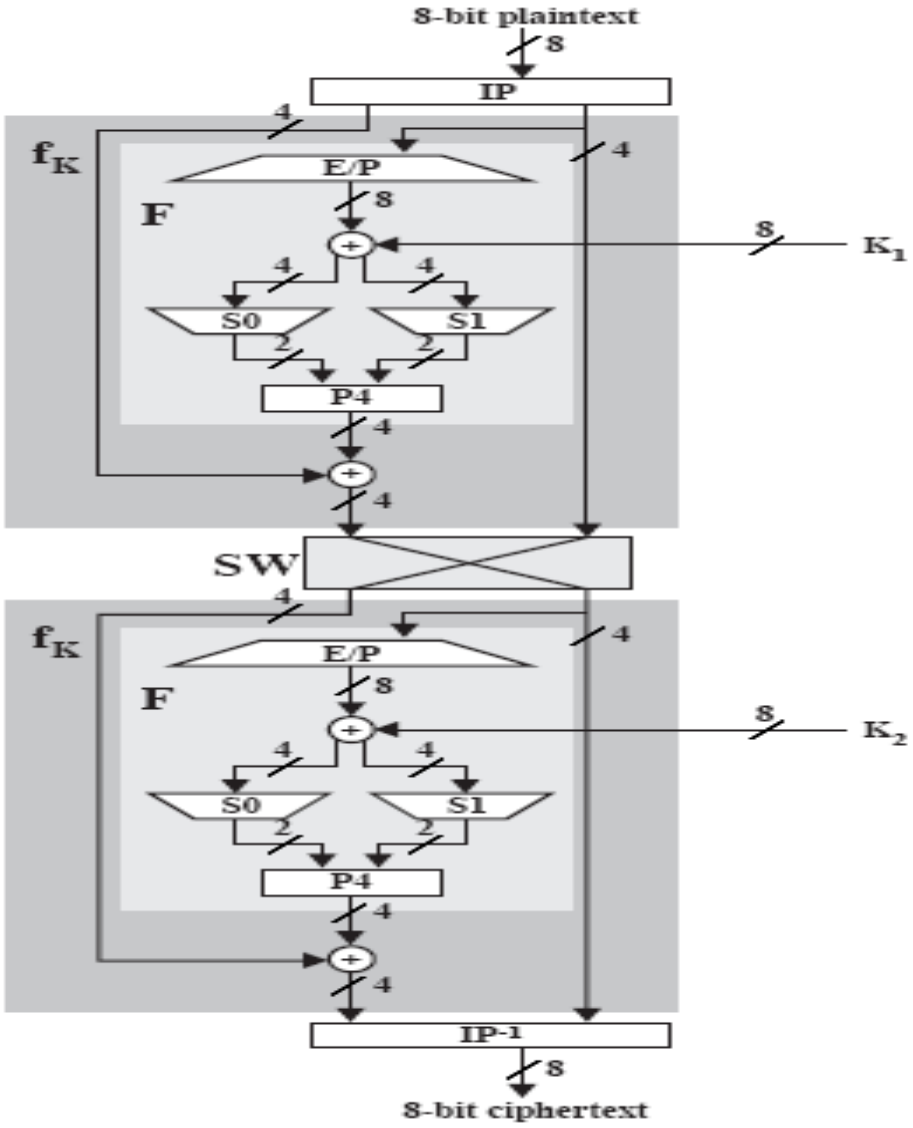
(00001)(11000)⇒(00100)(00011)

Finally apply again P_8:

P_8							
6	3	7	4	8	5	10	9

Then K_2=(01000011)

S-DES Encryption



S-DES Encryption

The IP 8-bit block plaintext is first permuted using the IP function:

IP							
2	6	3	1	4	8	5	7

At the end of the algorithm the inverse permutation is used :

IP^{-1}							
4	1	3	5	7	2	8	6

$IP^{-1}(IP(X)) = X$;

Ex: IP{(10110101)}=(01111100)

IP^{-1} {01111100}=(10110101)

End of Week 07