1 Define the following terms with suitable examples

a. Group b. Ring c. Field

GROUP

• A Group (G) is a set of elements with a binary operation "•" that satisfies four properties (or axioms).

{ G, •}

where, G is set of elements

"•" is a Binary operation

• A commutative group, also called an abelian group, is a group in which the operator satisfies the four properties for group plus an extra property, commutativity.

• The four properties for groups plus commutativity are defined as follows:

1. **Closure:** For all a, b ∈ G, then **a • b ∈ G**

2. **Associativity:** For all a, b, c ∈ G, then **a • (b • c) = (a • b) • c**

3. **Commutativity:** For all a, b ∈ G, then **a • b = b • a**

4. **Existence of identity:** For all element a in G, there exists an element e, identity

element, such that **e • a = a • e = a**

5. **Existence of inverse:** For each a there exists a', inverse of a, so that **a • a' = a' • a = e**

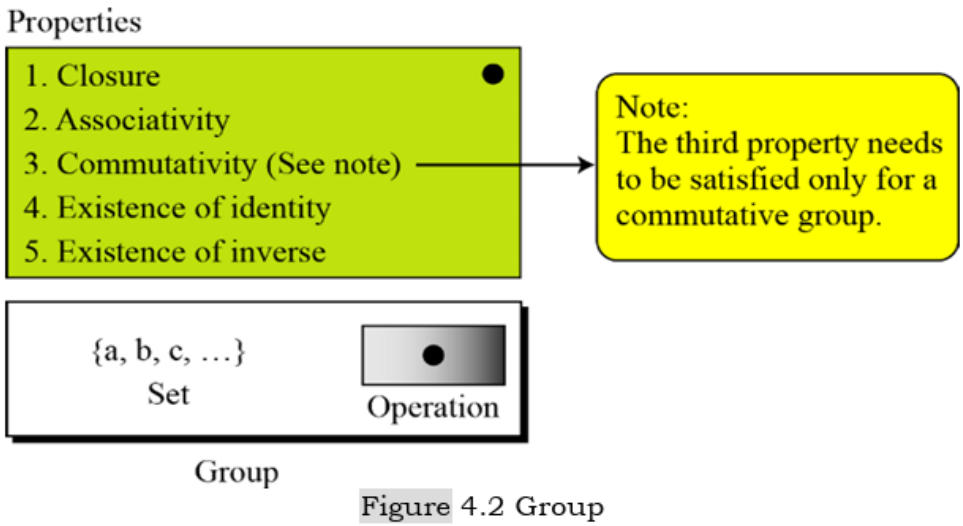


Figure 4.2 Group

Figure 4.2 shows the concept of a group and Abelian group.

EXAMPLE The set of residue integers with the addition operator, $G = \langle Z_n, + \rangle$, is a commutative group. We can perform addition and subtraction on the elements of this set without moving out of the set. Let us check the properties.

1. Closure is satisfied. The result of adding two integers in Z_n is another integer in Z_n.
2. Associativity is satisfied. The result of $4 + (3 + 2)$ is the same as $(4 + 3) + 2$.
3. Commutativity is satisfied. We have $3 + 5 = 5 + 3$.
4. The identify element is 0. We have $3 + 0 = 0 + 3 = 3$.
5. Every element has an additive inverse. The inverse of an element is its complement. For example, the inverse of 3 is -3 ($n - 3$ in Z_n) and the inverse of -3 is 3. The inverse allows us to perform subtraction on the set.

Ring

- A ring, denoted as R is an algebraic structure with two operations.

$$\{R, +, x\}$$

- The first operation must satisfy all 5 properties required for an abelian group.
- The second operation must satisfy only the first two.
- In addition, the second operation must be distributed over the first.
- **Distributivity** means that for all a, b, and c elements of R, we have

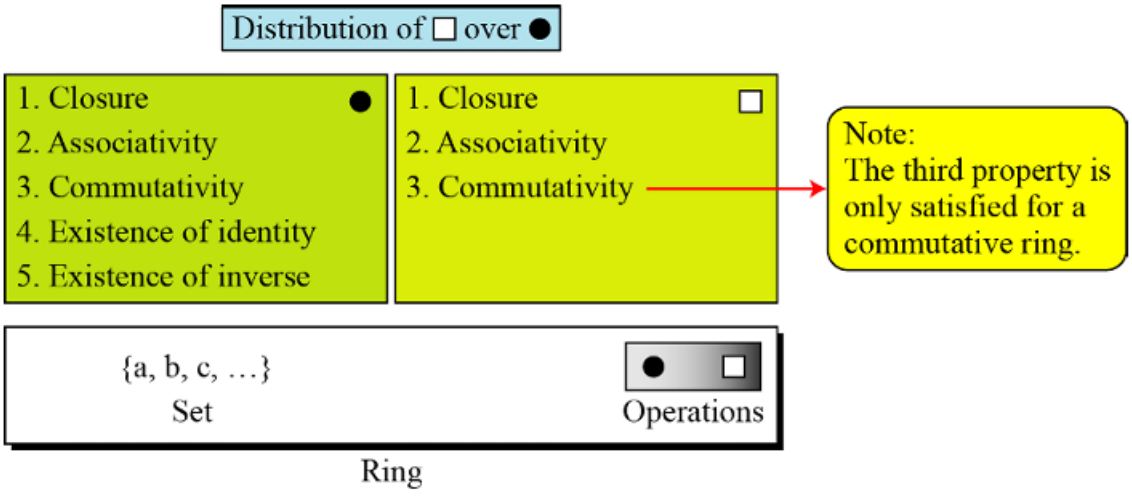
a (b + c) = (a b) +(a c)

and (a+ b) c =(a c) + (b c).

- **Multiplicative inverse**

a*1 = 1*a =a

- A commutative ring is a ring in which the commutative property is also satisfied for the second operation.
- The following figure shows a ring and a commutative ring.



Example:

- The set Z with two operations, addition and multiplication, is a commutative ring.
- We show it by R = <Z, +, ×>.
- Addition satisfies all of the five properties; multiplication satisfies only three properties.
- Multiplication also distributes over addition.

  (5 x (3 + 2) = (5 x 3) + (5 x 2)

Field

A field, denoted by F= <{……}, + , * > is a commutative ring in which the second operation satisfies all five properties defined for the first operation except that the identity of the first operation has no inverse.

In other words:

A field is a set with the two binary operations of addition and multiplication, both of which operations are commutative, associative, contain identity elements, and contain inverse elements.

The identity element for addition is 0, and the identity element for multiplication is 1.

The property: **Multiplicative Inverse:**

$a^* a^{-1} = (a^{-1})^*a = 1$

2 Write short notes on Substitution and Permutation

1. **Substitution (S-Box):**

- Substitution is the process of replacing each element in a block of plaintext with another element, typically according to a predetermined substitution table.
- An S-box can have a different number of inputs and outputs.
- An S-box is an m × n substitution unit, where m and n are not necessarily the same.
- In DES, substitution is achieved through the use of S-boxes (Substitution-boxes), which are a set of eight non-linear functions.
- Each S-box takes a 6-bit input and produces a 4-bit output. DES has eight S-boxes in total.
- The input 6 bits are divided into two parts: the first and last bits determine the row, while the middle four bits determine the column. The value found at the intersection of the determined row and column is the output.

2. **Permutation (P-Box):**

- Permutation involves rearranging the order of elements in a block of data according to a predetermined permutation table.

- In DES, permutation is achieved through the use of a fixed permutation table known as the P-box (Permutation box).
- The P-box operates on the output of the S-boxes and rearranges the bits according to its predefined permutation pattern.
- The P-box ensures that each bit of the output from the S-boxes influences many bits of the subsequent round, enhancing the diffusion property of DES.

3 Briefly explain about symmetric key cryptography.

• A symmetric-key modern block cipher encrypts an n-bit block of plaintext or decrypts an n-bit block of cipher text.

• The encryption or decryption algorithm uses a k-bit key.

• The decryption algorithm must be the inverse of the encryption algorithm, and both operations must use the same secret key so that receiver can retrieve the message sent by sender.

- Figure 5.1 shows the general idea of encryption and decryption in a modern block cipher.

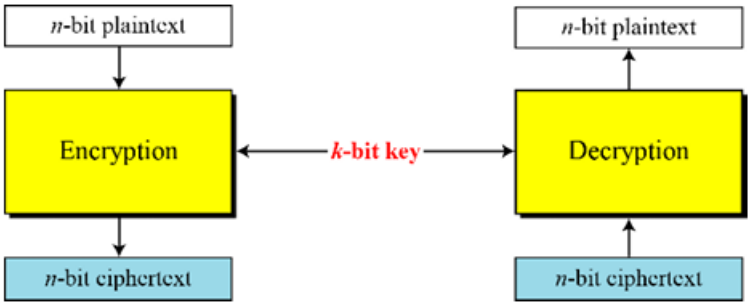


Figure 5.1 A modern block cipher

• If the message is fewer than n bits, padding must be added to make it an n-bit block; if more than n bits then it should be divided into n-bit blocks and the appropriate padding must be added.

• The common values for n are 64, 128, 256, or 512.

How it works:

1. **Key Generation**: In symmetric key cryptography, both the sender and receiver agree upon a secret key before communication begins. This key is kept confidential and is used exclusively between the communicating parties.
2. **Encryption**: To encrypt a message, the sender uses the secret key to transform the plaintext message into ciphertext. This process involves applying an encryption algorithm (such as DES, AES, or RC4) along with the secret key to the plaintext, producing the ciphertext.
3. **Decryption**: The receiver, who possesses the same secret key, uses it to decrypt the ciphertext back into plaintext. By applying the decryption algorithm (which is usually the inverse of the encryption algorithm) along with the secret key to the ciphertext, the receiver retrieves the original plaintext message.

4 Explain in detail Feistel Block Cipher structure with neat sketch. Distinguish between a Feistel and a non-Feistel block cipher.

Feistel Cipher Structure:

• Feistel designed a very intelligent and interesting cipher that has been used for decades.

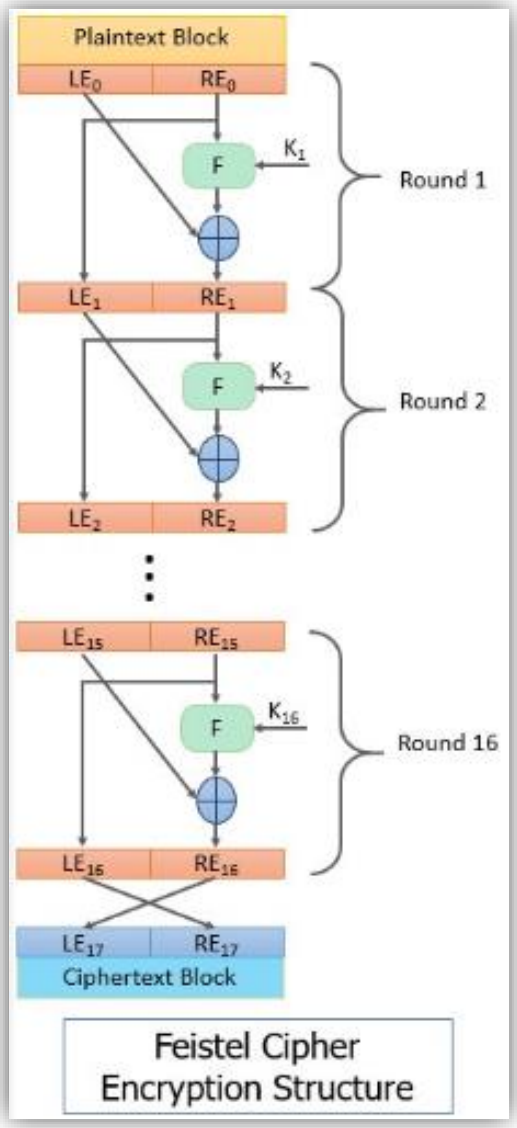
• A Feistel cipher can have three types of components: self-invertible, invertible, and noninvertible.

• A Feistel cipher combines all noninvertible elements in a unit and uses the same unit in the encryption and decryption algorithms.

• 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.



• 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.

Differences between a Feistel and a non-Feistel block cipher:

Aspect	Feistel Cipher	Non-Feistel Cipher
Structure	Divides input block into two halves.	Operates on the entire block simultaneously.
Round Function	Applies a round function to each half alternately.	Typically involves substitution and permutation operations across the entire block.
Examples	DES, Triple-DES (3DES)	AES
Decryption	Uses the same algorithm as encryption.	May require a different algorithm or implementation for decryption.
Performance	Efficient, relatively simple to implement.	Can offer high performance, especially when optimized for modern hardware architectures.
Security	Generally secure, especially with an adequate number of rounds.	Can provide high security if designed with strong cryptographic primitives and a sufficient number of rounds.
Advantages	Simple implementation, efficient decryption using the same algorithm.	High performance, potential for optimization on modern hardware.
Disadvantages	May require more rounds for equivalent security compared to non-Feistel ciphers.	Might need separate algorithms or implementations for encryption and decryption.
Usage	Historical significance, still used in some applications.	Widely adopted as the standard encryption algorithm in many contexts.

5 Explain about Round Function in Data Encryption Standard.

Round Function

 DES uses 16 rounds. Each round of DES is a Feistel cipher as shown in figure 6.4.

 The Round Function is the heart of the Data Encryption Standard (DES) cipher, performing the core encryption operation in each of its 16 rounds.

 It's responsible for mixing and transforming data using a subkey derived from the main key, making the overall encryption process secure and difficult to reverse.

 The round takes LI-1 and RI-1 from previous round (or the initial permutation box) and creates LI and RI, which go to the next round (or final permutation box).

 We can assume that each round has two cipher elements (mixer and swapper). Each of these elements is invertible.

 The swapper is obviously invertible. It swaps the left half of the text with the right half. The mixer is invertible because of the XOR operation.

 All noninvertible elements are collected inside the function f (RI-1, KI).

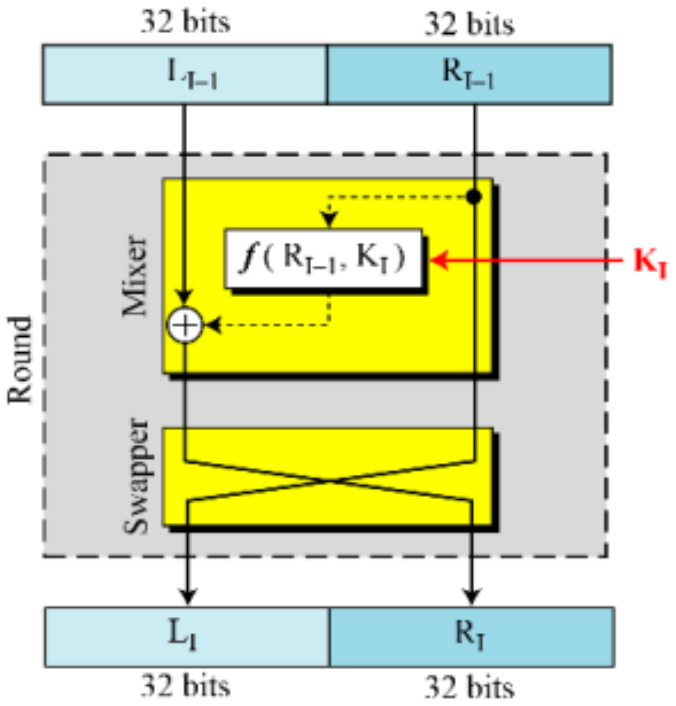


Figure 6.4 A round in DES (encryption site)

1. **Expansion Permutation (E)**:
    - The round begins with expanding the 32-bit right half of the block to 48 bits using an expansion permutation table. This permutation increases the diffusion of the input bits, providing more mixing and complexity in subsequent operations.
2. **XOR with Subkey**:

- The expanded 48-bit result is XORed with the round subkey. Each round of DES uses a unique 48-bit subkey derived from the original 56-bit key using a key scheduling algorithm. This XOR operation introduces the key material into the round, contributing to the confusion and diffusion properties of the cipher.

3. **S-box Substitution**:
    - The XOR result is divided into eight 6-bit blocks. Each of these blocks is substituted using eight S-boxes (Substitution boxes), each having a predefined substitution table. Each S-box takes 6 bits as input and outputs 4 bits, resulting in a total output of 32 bits. The S-box substitution adds non-linearity to the cipher, enhancing its resistance against differential cryptanalysis.
4. **Permutation (P)**:
    - The output of the S-box substitution is then subjected to a fixed permutation known as the P-box permutation. This permutation rearranges the bits according to a predefined permutation table. The purpose of this permutation is to further mix the output bits, providing additional confusion in the cipher.
5. **XOR with Left Half**:
    - Finally, the output of the permutation is XORed with the original left half of the block. This XOR operation completes the round function, generating the new right half for the next round.

After completing all 16 rounds of DES, the final output undergoes a final permutation to produce the ciphertext. The round function is a crucial component of DES, contributing to its confusion and diffusion properties, which are essential for achieving strong cryptographic security.

6 Explain about different transformations in Advanced Encryption Standard.

For encryption, each round consists of the following steps:

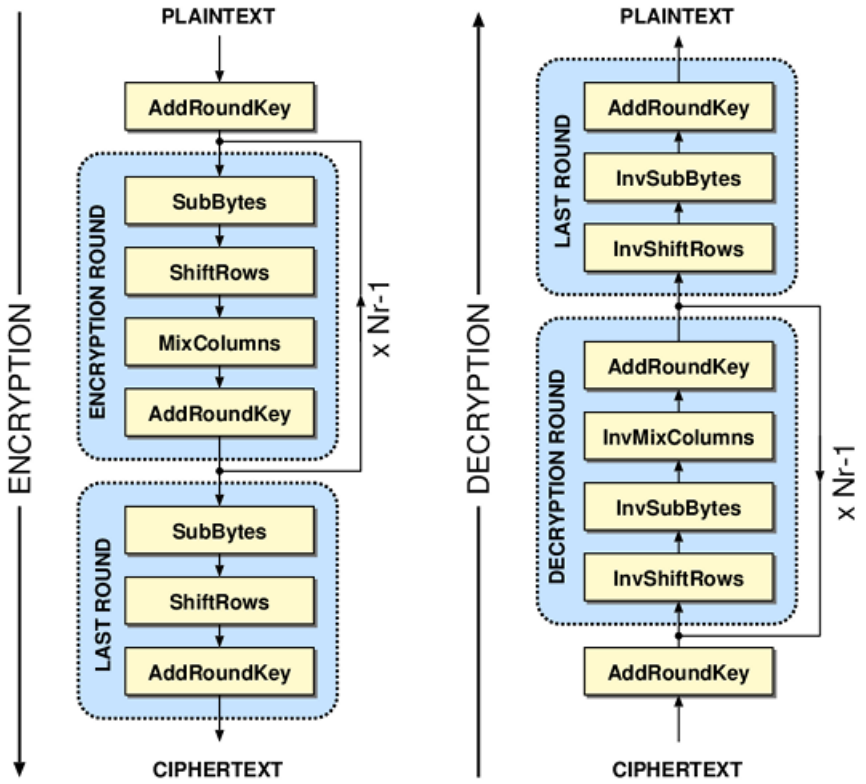
1) SubBytes

2) ShifRows

3)MixColumns

4) AddRoundKey

Above steps also called AES transformation functions.



1. SubBytes / Substitution Bytes

• AES defines a 16 X 16 matrix of byte values, called an S-box that contains a permutation of all possible 256 8-bit values.

• Each individual byte of State is mapped into a new byte in the following way:

 The left most 4 bits of the byte are used as a row value and the right most 4 bits used as a column value.

• These row and column values serve as indexes into the S-box to select a unique 8-bit output value.

• For example, the hexadecimal value 00 references row 0, column 0 of S-box which contains the value {63}, accordingly the value {12} is mapped into {C9}.

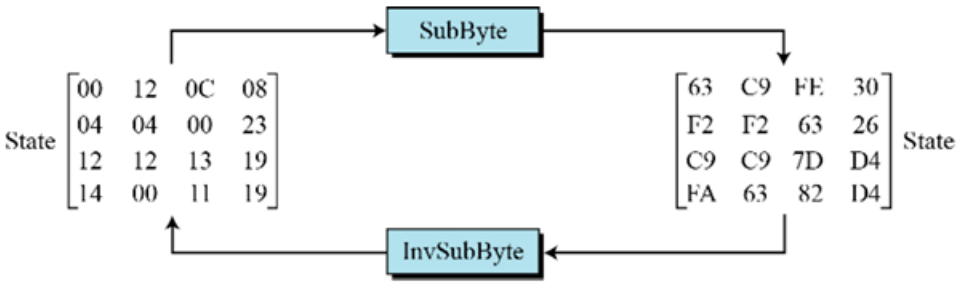


Figure: SubBytes transformation

2. ShiftRows transformation

• The shift rows transformation is called ShiftRows.

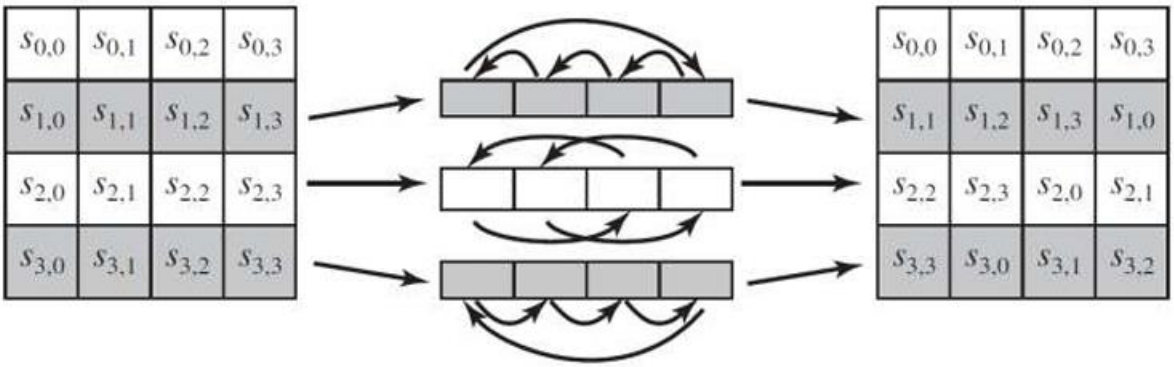
• Rules of shifting rows,

 Row 1 → now shifting

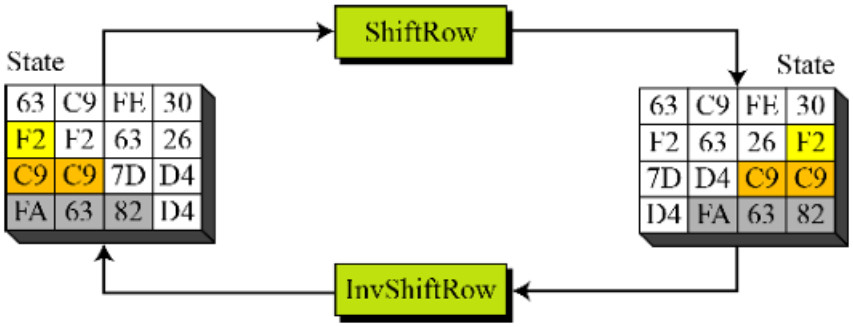
 Row 2 → 1 byte left shift

 Row 3 → 2 byte left shift

 Row 4 → 3 byte left shift



The inverse shift row transformation is called InverseShiftRows, performs the circular shift in the opposite direction for each of the last three rows, which a one-byte circular right shift for the second row and so on.



3. MixColumns

• The mix column transformation is called MixColumns, operates on each column individually.

• Each byte of a column is mapped into a new value that is a function of all four bytes in that column.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Predefine Matrix State Array New State Array

$$s'_{0,j} = (2 \bullet s_{0,j}) \oplus (3 \bullet s_{1,j}) \oplus s_{2,j} \oplus s_{3,j}$$
$$s'_{1,j} = s_{0,j} \oplus (2 \bullet s_{1,j}) \oplus (3 \bullet s_{2,j}) \oplus s_{3,j}$$
$$s'_{2,j} = s_{0,j} \oplus s_{1,j} \oplus (2 \bullet s_{2,j}) \oplus (3 \bullet s_{3,j})$$
$$s'_{3,j} = (3 \bullet s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \bullet s_{3,j})$$

Example calculation is given below.

2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

*

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$$(\{02\} \cdot \{87\}) \oplus (\{03\} \cdot \{6E\}) \oplus \{46\} \oplus \{A6\} = \{47\}$$
$$\{87\} \oplus (\{02\} \cdot \{6E\}) \oplus (\{03\} \cdot \{46\}) \oplus \{A6\} = \{37\}$$
$$\{87\} \oplus \{6E\} \oplus (\{02\} \cdot \{46\}) \oplus (\{03\} \cdot \{A6\}) = \{94\}$$
$$(\{03\} \cdot \{87\}) \oplus \{6E\} \oplus \{46\} \oplus (\{02\} \cdot \{A6\}) = \{ED\}$$

4. AddRoundKey

• In the forward add round key transformation, AddRoundKey, the 128-bits of State are bitwise XORed with the 128-bits of the round key.

• As shown in figure, the operation is viewed as a column wise operation between 4 bytes of a state column and one word of the round key; it can also be viewed as a byte-level XOR operation.

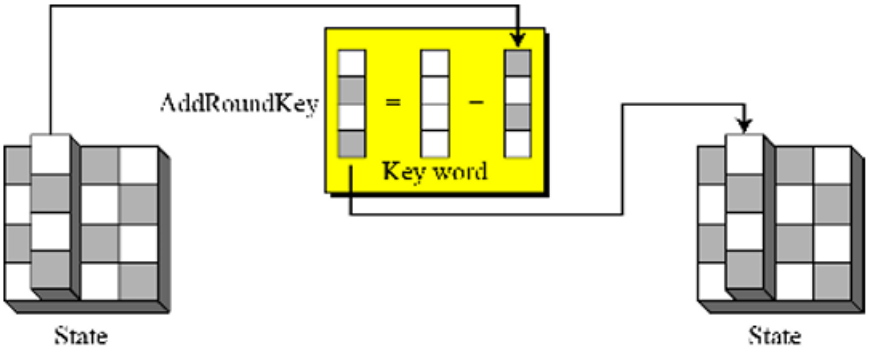


Figure: AddRoundKey transformation

7 Explain DES cryptography in detail.

DATA ENCRYPTION STANDARD (DES) The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST)

The **DES algorithm** takes the plain text of 64-bit as input & produces a ciphertext of 64-bit using a key of 56 bits. Initially, a 64-bit key length is used but an 8-bit is discarded.

- **Key length:** 56 bits (effective length)
- **Block size:** 64 bits
- **Rounds:** 16
- **Structure:** Feistel structure

DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration

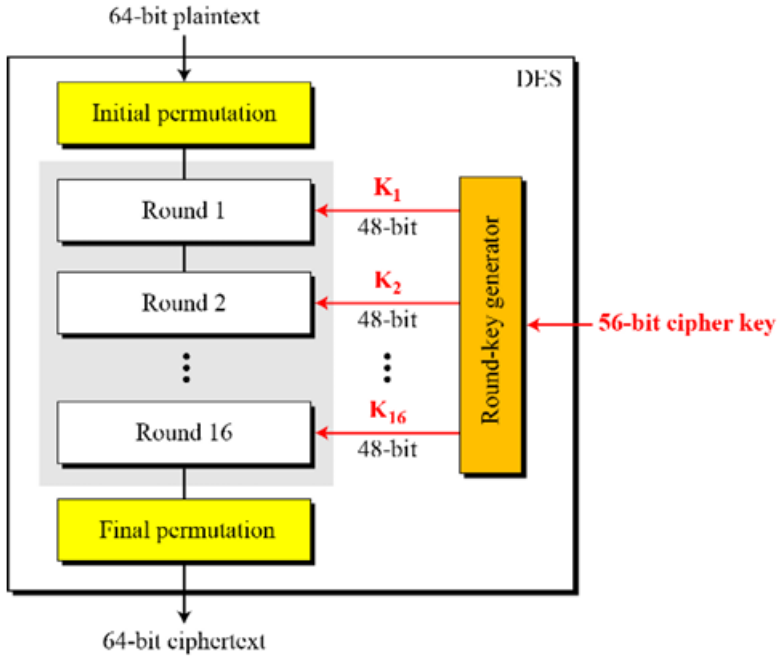


Figure 6.2 General structure of DES

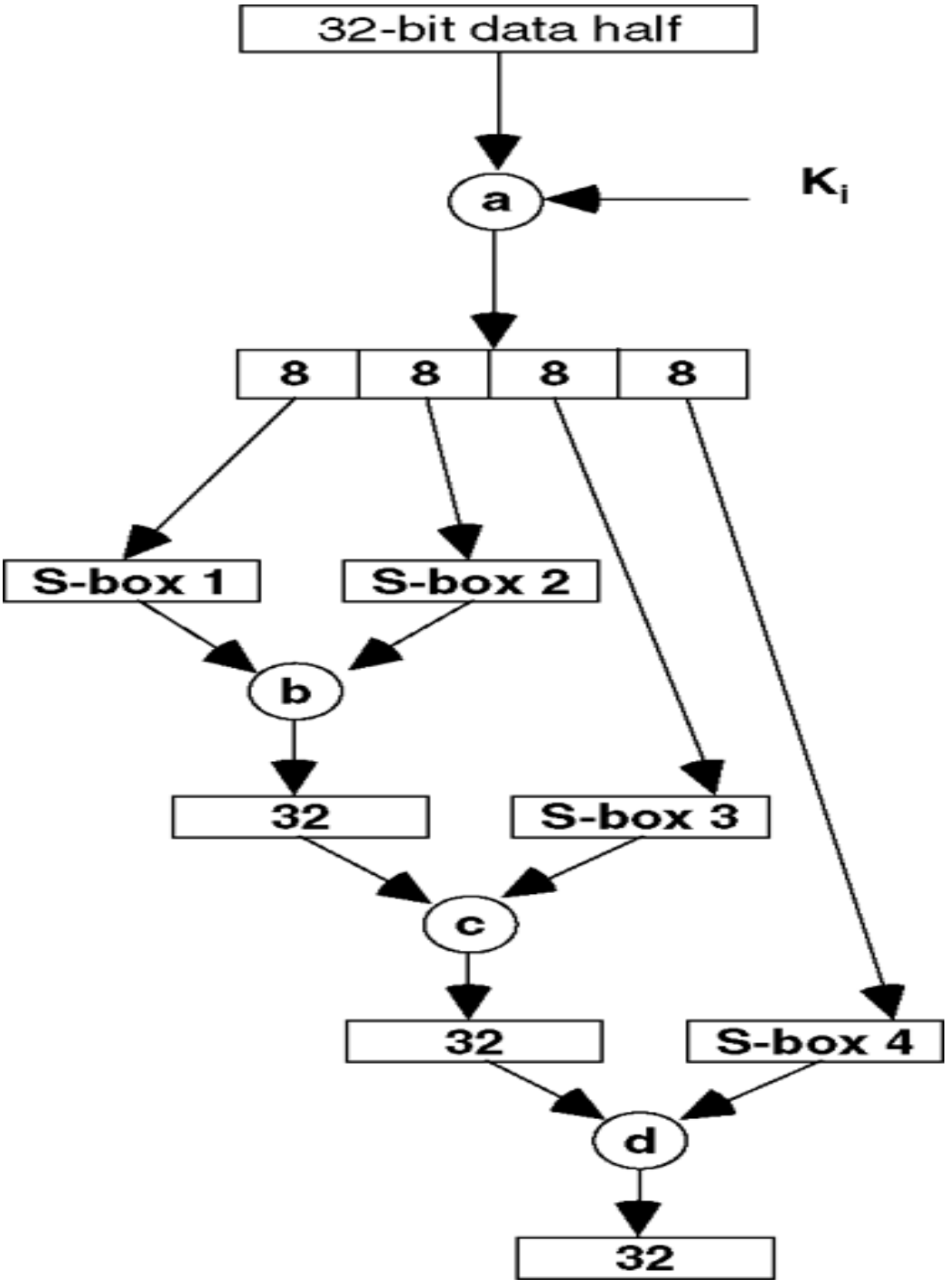
8 Briefly explain about CAST algorithm.

The CAST Block Cipher

- The CAST Block Cipher is an improvement of the DES block cipher, invented in Canada by Carlisle Adams and Stafford Tavares.
- The name of the cipher seems to be after the initials of the inventors.
- The CAST algorithm has 64 bit block size and has a key of size 64 bits.

- **Key length:** 64 bits
- **Block size:** 64 bits
- **Rounds:** 8
- **Structure:** Feistel structure

- CAST is based on the Feistel structure to implement the substitution permutation network. The authors state that they use the Feistel structure, as it is well studied and free of basic structural weaknesses.
- The plaintext block is divided into a left half and a right half. The algorithm has 8 rounds. Each round is essentially a Feistel structure.
- In each round the right half is combined with the round key using a function f and then XOR-ed with the left half. The new left half after the round is the same as the right half before the round.
- After 8 iterations of the rounds, the left and the right half are concatenated to form the ciphertext.



9 Explain about general structure of AES algorithm.

AES is cryptographic algorithm published by NIST (National Institute of Standards and Technology).

• The algorithm was proposed by Rijndael, it is also known as Rijndael algorithm.

• AES works on block cipher technique. Size of plaintext and ciphertext is same.

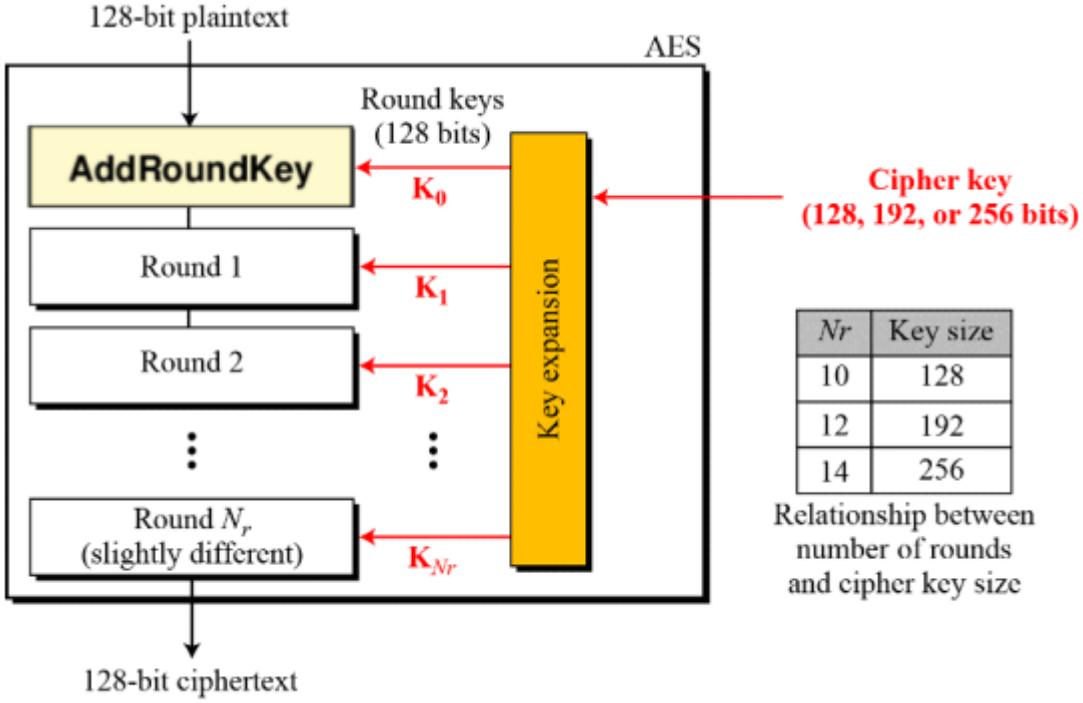
Key length:	128 bits -> 10 rounds
	192 bits -> 12 rounds
	256 bits -> 14 rounds
Block size:	128 bits
Structure:	Non-Feistel structure

• The AES states that the algorithm can only accept a block size of 128 bits and a choice of three keys - 128, 192, 256 bits.

• Depending on which version is used, the name of the standard is modified to AES-128, AES-192 or AES- 256 respectively.

• In the following figure, Nr defines the number of rounds. The figure also shows the relationship between the number of rounds and the key size.

• However, the rounds keys, which are created by the key-expansion algorithm are always 128-bits, the same size as the plaintext or ciphertext block.



Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Figure: General design of AES encryption cipher.

• The number of keys generated by the key-expansion algorithm is always one more than the number of rounds.

In other words, we have

Number of round keys = number of rounds + 1

We refer round keys as K0, K1, K2, ……., KNr

10 Explain about Blowfish algorithm.

BLOWFISH

• Blowfish is a 64-bit block cipher invented by Bruce Schneier.

• Blowfish was designed for fast ciphering on 32-bit microprocessors.

- **Block size:** 64 bits
- **Key size:** 32 to 448 bits
- **Rounds:** 16
- **Structure:** Feistel structure

• Blowfish is also compact and has a variable key length which can be increased to 448 bits.

• Blowfish is suitable for applications where the key does not change frequently like communication links or file encryptors.

• However for applications like packet switching or as a one-way hash function, it is unsuitable.

• Blowfish is not ideal for smart cards, which requires even more compact ciphers.

• Blowfish is faster than DES when implemented on 32-bit microprocessors.

Round Structure

• The algorithm is based on the Feistel structure and has two important parts: The round structure and the key expansion function.

• Each round in Blowfish consists of the following steps, processing a 64-bit data block and a subkey derived from the main key:

1. Splitting: The data block is divided into two 32-bit halves: left (L) and right (R).

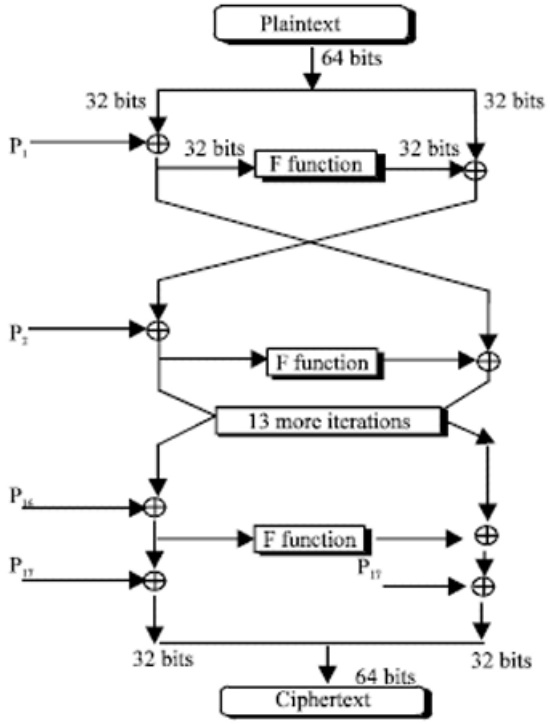
2. Right Half Processing: * **Expansion:** The right half (R) is expanded to 48 bits using a fixed E-box (expansion table). * **Mixing with Subkey:** The expanded R is XORed with a 48-bit subkey derived from the main key using the key schedule.

3. S-Boxes: The mixed data (48 bits) is divided into eight 6-bit sub-blocks, each processed by a different S-box (substitution box). Each S-box has a unique table that substitutes the 6-bit input with a 4-bit output, introducing non-linearity and making the cipher resistant to linear cryptanalysis.

4. Combining with Left Half: The outputs of the S-boxes (32 bits) are XORed with the left half (L) from the previous round.

5. Swapping: The halves are swapped, with the new left half (L) becoming the right half (R) for the next round.

6. Subkey Update: The subkey index increments for the next round.



Key Scheduling Algorithm

The subkeys are computed using the following method:

1. The P-array and then the four S-Boxes are initialized with a fixed string. The string is the hexadecimal digits of π.
2. P1 is XOR-ed with 32 bits of the key, P2 is XOR-ed with the next 32 bits of the key, and so on for all the bits of the key. If needed the key bits are cycled to ensure that all the P- array elements are XOR-ed.
3. An all-zero string is encrypted with the Blowfish algorithm, with the subkeys P1 to P18 obtained so far in steps 1 and 2.
4. P1 and P2 are replaced by the 64 bit output of step 3.
5. The output of step 3 is now encrypted with the updated subkeys to replace P3 and P4 with the ciphertext of step 4.
6. This process is continued to replace all the P-arrays and the S-Boxes in order.

• This complex key-scheduling implies that for faster operations the subkeys should be precomputed and stored in the cache for faster access.

• Security analysis by Serge Vaudenay shows that for a Blowfish algorithm implemented with known S-Boxes (note that in the original cipher the S-Boxes are generated during the encryption process) and with r-rounds, a differential attack can recover the P-array with 28r+1 chosen plaintexts.

11 Explain IDEA algorithm.

IDEA Algorithm:

• IDEA is another block cipher.

• It operates on 64 bit data blocks and the key is 128 bit long.

- **Block size:** 64 bits
- **Key size:** 128 bits
- **Rounds:** 8.5 rounds

• It was invented by Xuejia Lai and James Massey, and named IDEA (International Data Encryption Algorithm) in 1990, after modifying and improving the initial proposal of the cipher based on the seminal work on Differential cryptanalysis by Biham and Shamir.

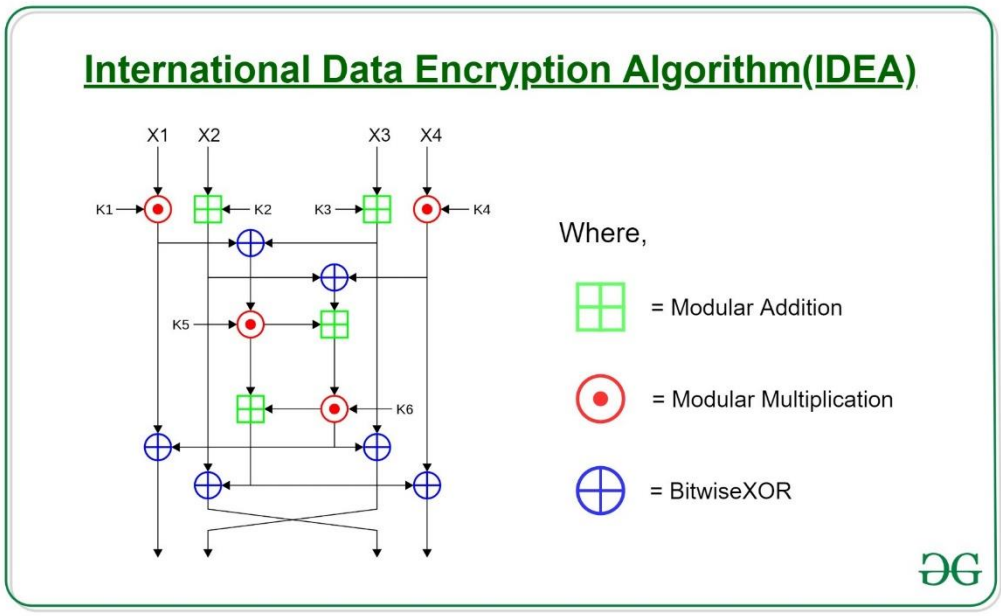
• The design principle behind IDEA is the "mixing of arithmetical operations from different algebraic groups".

• These arithmetical operations are easily implemented both in hardware and software.

• The underlying operations are XOR, addition modulo 216, multiplication modulo 210+1.

• The cipher obtains the much needed non-linearity from the later two arithmetical operations and does not use an explicit S-Box.

• Round Transformation of IDEA The 64-bit data is divided into four 16 bit blocks: X1, X2, X3, X4.



• These four blocks are processed through eight rounds and transformed by the above arithmetical operations among each other and with six 16 bit subkeys. In each round the sequence of operations is as follows:

1. Multiply X1 and the first subkey.
2. Add X2 and the second subkey.
3. Add X3 and the third subkey.
4. Multiply X4 and the fourth subkey.
5. XOR the results of step 1 and 3.
6. XOR the results of step 2 and 4.
7. Multiply the results of steps 5 with the fifth subkey.
8. Add the results of steps 6 and 7.
9. Multiply the results of steps 8 with the sixth subkey.
10. Add the results of steps 7 and 9.
11. XOR the results of steps 1 and 9.
12. XOR the results of steps 3 and 9.

13. XOR the results of steps 2 and 10.
14. XOR the results of steps 4 and 10.

• The outputs of steps 11, 12, 13 and 14 are stored in four words of 16 bits each, namely Y1, Y2, Y3 and Y4.
• The blocks Y2 and Y3 are swapped, and the resultant four blocks are the output of a round of IDEA.

• It may be noted that the last round of IDEA does not have the swap step.

• Instead the last round has the following additional transformations:

1. Multiply Y1 and the first subkey.
2. Add Y2 and the second subkey.
3. Add Y3 and the third subkey.
4. Multiply Y4 and the fourth subkey.

Finally, the ciphertext is the concatenation of the blocks Y1, Y2, Y3 and Y4.

12 Explain about Design Criteria and Properties of DES.

• Critics have used a strong magnifier to analyze DES.

• Tests have been done to measure the strength of some desired properties in a block cipher.

• The elements of DES have gone through scrutinies to see if they have met the established criteria.

Properties

Two desired properties of a block cipher are the avalanche effect and the completeness.

1. Avalanche Effect:

• Avalanche effect means a small change in the plaintext (or key) should create a significant change in the ciphertext.

• DES has been proved to be strong with regard to this property.

2. Completeness Effect:

• It means that each bit of the ciphertext needs to depend on many bits on the plaintext.

• The diffusion and confusion produced by D-boxes and S-boxes in DES, show a very strong completeness effect.

Design Criteria

• The design of DES was revealed by IBM in 1994.

• Many tests on DES have proved that it satisfies some of the required criteria as claimed.

Some of these design issues are:

S-Boxes:

• The design provides confusion and diffusion of bits from each round to the next.

• According to this revelation and some research, we can mention several properties of S-boxes.

1. The entries of each row are permutations of values between 0 and 15.
2. S-boxes are nonlinear. In other words, the output is not an affine transformation of the input.
3. If we change a single bit in the input, two or more bits will be changed in the output.

4. If two inputs to an S-box differ only in two middle bits (bits 3 and 4), the output must differ in at least two bits. In other words, S(x) and S(x $\oplus$ 001100) must differ in at least two bits where x is the input and S(x) is the output.
5. If two inputs to an S-box differ in the first two bits (bits 1 and 2) and are the same in the last two bits (5 and 6), the two outputs must be different. In other words, we need to have the following relation S(x) $\neq$ S(x $\oplus$ 11bc00), in which b and c are arbitrary bits.
6. There are only 32 6-bit input-word pairs (xi and xj), in which xi $\oplus$ xj $\neq$ (000000)2. These 32 input pairs create 32 4-bit output-word pairs. If we create the difference between the 32 output pairs, d = yi $\oplus$ yj, no more than 8 of these d's should be the same.
7. A criterion similar to # 6 is applied to three S-boxes.
8. In any S-box, if a single input bit is held constant (0 or 1) and the other bits are changed randomly, the differences between the number of 0s and 1s are minimized.

D-Boxes:

• Between two rows of S-boxes (in two subsequent rounds), there are one straight D-box (32 to 32) and one expansion D-box (32 to 48).

• These two D-boxes together provide diffusion of bits.

• The following criteria were implemented in the design of D-boxes to achieve this goal:

1. Each S-box input comes from the output of a different S-box (in the previous round).
2. No input to a given S-box comes from the output from the same box (in the previous round).
3. The four outputs from each S-box go to six different S-boxes (in the next round).
4. No two output bits from an S-box go to the same S-box (in the next round).
5. If we number the eight S-boxes, S1 , S2 , …, S8,
    a. An output of Sj-2 goes to one of the first two bits of Sj (in the next round).
    b. An output bit from Sj -1 goes to one of the last two bits of Sj (in the next round).
    c. An output of Sj +1 goes to one of the two middle bits of Sj (in the next round).
6. For each S-box, the two output bits go to the first or last two bits of an S-box in the next round. The other two output bits go to the middle bits of an S-box in the next round.
7. If an output bit from Sj goes to one of the middle bits in Sk (in the next round), then an output bit from Sk cannot go to the middle bit of Sj. If we let j = k, this implies that none of the middle bits of an S-box can go to one of the middle bits of the same S-box in the next round.

Number of Rounds:

• DES uses sixteen rounds of Feistel ciphers.

• It has been proved that after eight rounds, each ciphertext is a function of every plaintext bit and every key bit; the ciphertext is thoroughly a random function of plaintext and ciphertext.

• Therefore, it looks like eight rounds should be enough. However, experiments have found that DES versions with less than sixteen rounds are even more vulnerable to known- plaintext attacks than brute-force attack, which justifies the use of sixteen rounds by the designers of DES.