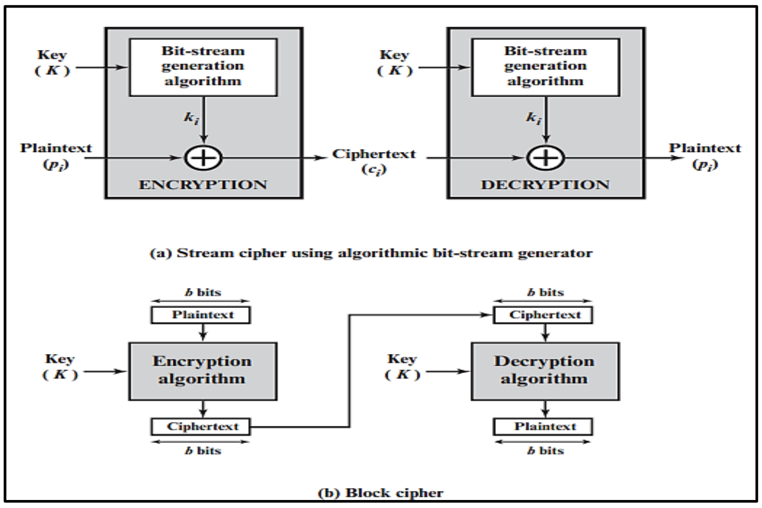
UNIT II

BLOCK CIPHERS AND ADVANCED ENCRYPTION STANDARD

01. EXPLAIN CONCEPT OF STREAM CIPHER AND BLOCK CIPHER. EXPLAIN BLOCK CIPHER PRINCIPLES.



A stream cipher and a block cipher are two ***types of symmetric encryption cipher*** that use a secret key to transform plaintext into ciphertext. The main difference between them is that a ***stream cipher encrypts and decrypts one bit or byte of plaintext at a time,*** while a block cipher ***operates on fixed-length blocks of plaintext,*** usually 64 or 128 bits.

A block cipher has several design principles that ensure its security and efficiency. Some of these principles are:

- **Number of rounds:** The number of rounds is the ***number of times the encryption algorithm is applied to each block of plaintext.*** A ***higher number of rounds*** makes the cipher more ***resistant to cryptanalysis, but also slower.***

- **Round Function:** The round function is the ***core component of the encryption algorithm that performs various substitutions and permutations*** on the plaintext block and the key. The function should be ***non-linear, complex, and have a good avalanche effect,*** which means that a small change in the input should cause a large change in the output.

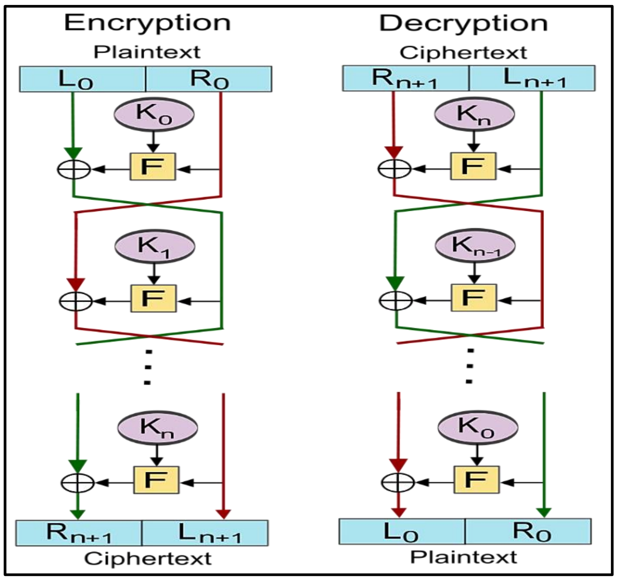
- **Key size:** The key size is the ***length of the secret key in bits.*** A larger key size means that there are more possible keys, ***making it harder for an attacker to guess the correct one by brute force. A key size of 128 bits is considered to be secure for most applications.***

- **Key schedule:** The key schedule is the ***algorithm that generates the subkeys*** used for each round of encryption ***from the main key.*** The key schedule should produce ***independent and unpredictable subkeys*** that resist attacks that exploit ***weak keys or key-dependent properties*** of the cipher.

- **Block size:** The block size is the ***length of the plaintext and ciphertext blocks in bits.*** A ***larger block size means that there are fewer repetitions of blocks in the ciphertext,*** making it harder for an attacker to analyse the ***statistical patterns*** in the plaintext. A ***block size of 128 bits*** is generally considered to be secure for most applications.

- **Non-linearity:** The non-linearity of the cipher is the ***degree to which the output of the cipher depends on the input in a non-linear way.*** A ***linear cipher is vulnerable to attacks that exploit the linear properties*** of the cipher. A ***non-linear cipher should provide confusion and diffusion,*** which means that the ***ciphertext should be a complex function of the plaintext and the key.***

02. EXPLAIN THE FEISTEL CIPHER STRUCTURE



The Feistel cipher is a ***symmetric encryption structure*** developed by ***IBM*** scientist ***Horst Feistel in the 1973s,*** widely used in block cipher algorithms such as ***DES (Data Encryption Standard) and its successor AES (Advanced Encryption Standard).***
The Feistel cipher operates on a block of plaintext data and applies a series of rounds of transformations to produce the ciphertext. It is based on the concept of a ***"round function"*** that takes part of the plaintext block and a subkey as input, and produces an output. The structure of the Feistel cipher includes the following key components:

1. **Block Division**: The ***plaintext block is divided into two equal parts,*** denoted as ***L0 and R0,*** which serve as the inputs for the initial round of the Feistel cipher.
2. **Round Function**: The round function takes the ***right half (R0) of the plaintext block and a subkey as its inputs.*** The subkey is derived from the original encryption key using a ***key scheduling algorithm.*** The round function ***performs a series of substitutions and permutations*** on R0 based on the subkey, producing an output denoted as ***f(R0, subkey).***
3. **XOR Operation**: The ***output of the round function is XORed with the left half (L0) of the plaintext block.*** The result of this XOR operation becomes the ***new right half (R1)*** for the next round.
4. **Swap Operation**: The ***original right half (R0) becomes the new left half (L1)*** for the next round. In other words, the ***left and right halves are swapped after each round.***
5. **Rounds**: Steps 2-4 are ***repeated for a fixed number of rounds, typically 16 rounds in the case of DES.*** The ***output of round function and subkey are different for each round,*** and the output of each round becomes the input for the next round.

6. **Final Permutation**: After the final round, the ***left and right halves are combined*** and undergo a final permutation to produce the ciphertext block.
7. **Decryption**: Decryption using a Feistel cipher is essentially the same process as encryption, except that the ***subkeys are used in reverse order.***

The Feistel cipher has several important properties that make it suitable for encryption:

- **Confusion and Diffusion**: The ***round function provides confusion by mixing the ciphertext and subkey in a complex manner,*** while the ***multiple rounds provide diffusion*** by spreading the influence of each input bit throughout the entire block.
- **Key Mixing**: The subkey is combined with the input data in each round, ensuring that even ***small changes in the key produce significant changes in the ciphertext.***

The Feistel cipher is a ***flexible and secure encryption structure*** that can ***accommodate different block sizes and numbers of rounds,*** making it a versatile choice for building secure block ciphers. Its properties, including ***security, flexibility, and ease of implementation*** in hardware and software, have established it as a widely used and trusted encryption structure, ***forming the basis for many modern block cipher algorithms.***

03. DRAW AND EXPLAIN THE GENERAL DESCRIPTION OF DES ENCRYPTION ALGORITHM.
Data Encryption Standard, is a ***symmetric encryption algorithm*** used for ***encrypting and decrypting digital data.*** It operates on ***64-bit blocks of plaintext using a 56-bit key.*** DES ***employs a Feistel network structure,*** consisting of ***initial and final permutations, sub-key generation, round functions*** to produce ciphertext. It was widely used until the introduction of the Advanced Encryption Standard (AES).

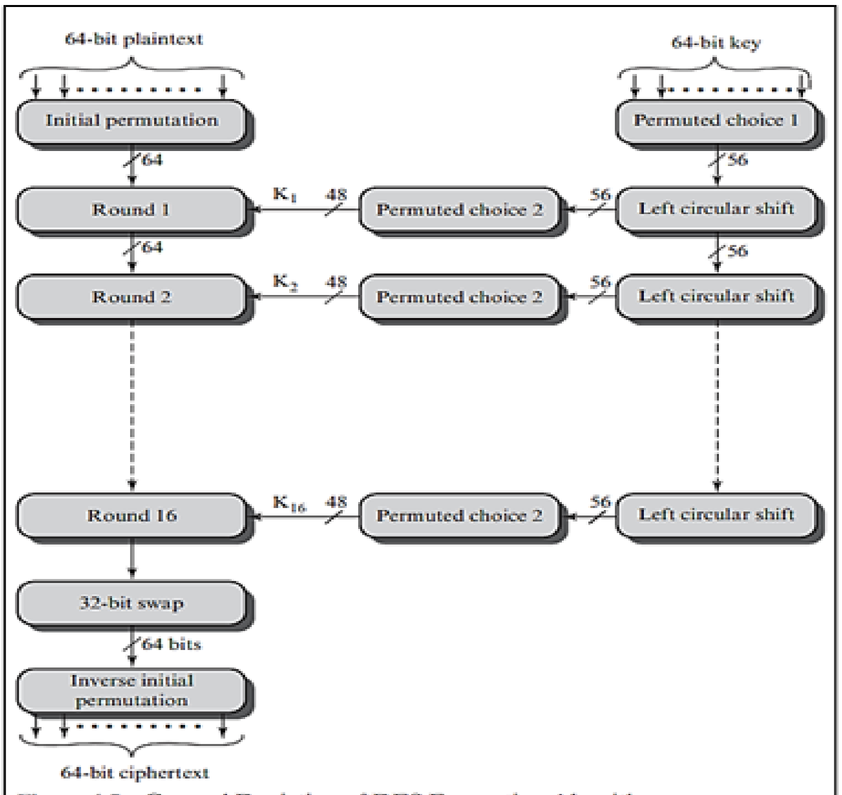


Figure 4.5 General Depiction of DES Encryption Algorithm

Here's a general description of the DES encryption algorithm:

1. **Initial Permutation (IP)**:
   - The 64-bit plaintext undergoes an initial permutation, where the ***positions of the bits are rearranged according to a fixed permutation table.***
   - This rearrangement ***doesn't affect the value of the bits*** but determines their order for subsequent processing.
2. **Key Processing**:
   - The 64-bit encryption key undergoes an initial permutation, known as ***Permuted Choice 1 (PC-1).***
   - The ***PC-1 selects 56 bits from the original key, discarding 8 parity bits.***
   - Subsequently, the ***key is divided into two 28-bit halves,*** often referred to as ***C0 and D0.***
3. **Sub-Key Generation**:
   - For each of the 16 rounds of DES, a subkey (also known as round key) is derived from the main key.
   - This process involves ***left circular shifts of the C and D halves*** and another permutation, known as ***Permuted Choice 2 (PC-2).***
   - Each round generates a ***unique 48-bit subkey.***
4. **Round Function**:
   - The core of DES encryption involves ***16 rounds of the same function applied to the plaintext.***
   - Each round consists of ***expansion, permutation, substitution*** **(S-box substitution).** ***DES's round function implements Feistel network structure.***
   - The function uses both the ***plaintext and the sub-key as inputs.***
5. **Final Permutation (IP^-1)**:
   - After the 16th round, the 64-bit output undergoes a final permutation, which is the ***inverse of the initial permutation.***
   - This ***rearranges the bits back to their original order.***
6. **Ciphertext**:
   - The final output after the inverse initial permutation is the 64-bit ciphertext.

In summary, the DES encryption algorithm involves initial and final permutations, key processing to generate round keys, and a series of rounds where a combination of ***expansion, permutation, substitution*** are applied to the plaintext using the round function. The ***output is the ciphertext, which is 64 bits in length.***

04. EXPLAIN IN DETAIL THE S-DES ENCRYPTION SCHEME.

S-DES encryption scheme is a simplified version of the DES algorithm, which is a symmetric key cipher that encrypts and decrypts 8-bit blocks of data using a 10-bit key.

S-DES consists of the following steps:

- **Key generation:** The 10-bit key is permuted and shifted to produce two 8-bit subkeys, K1 and K2, using the functions P10, P8, and LS-1.

- **Initial permutation:** The 8-bit plaintext is permuted using the function IP.

- **Round 1:** The permuted plaintext is divided into two 4-bit halves, L0 and R0. The right half, R0, is expanded to 8 bits using the function E/P and XORed with the first subkey, K1. The result is then passed through the function S-box, which produces a 4-bit output. The output is then permuted using the function P4 and XORed with the left half, L0. The result is the new left half, L1. The new right half, R1, is the same as R0.

- **Switch function:** The two halves, L1 and R1, are swapped to produce the input for the next round.

- **Round 2:** The same process as round 1 is repeated, except that the second subkey, K2, is used instead of K1

- **Inverse permutation:** The final output of round 2 is permuted using the function IP-1, which is the inverse of IP.

The output of the inverse permutation is the 8-bit ciphertext. Decryption is the reverse process of encryption, using the same key and functions.

05. WHAT IS AVALANCHE EFFECT? LIST AND EXPLAIN THE STRENGTHS OF DES.

The avalanche effect is a desirable ***property of encryption algorithms where a small change in either the plaintext or the key results in a significant change in the ciphertext.*** In other words, even a minor alteration in the input should cause a drastic alteration in the output. This characteristic is ***essential for ensuring the security of the encryption algorithm*** because it makes difficult for an attacker to discern any patterns or relationships between the plaintext and the ciphertext.

Strengths of DES (Data Encryption Standard):

1. **56-Bit Key Size:** DES uses a 56-bit key, resulting in a ***key space of approximately 7.2 * 10^16 possible keys.*** While this key size was considered secure when DES was first standardized, it's now considered relatively small and susceptible to brute-force attacks.
2. **Resistance Against Cryptanalysis:** Some people think DES is not secure enough, especially because of its ***S-boxes and some possible flaws.*** But ***no one has been able to crack DES yet.*** Even though some people have tried to find problems in the algorithm, DES can still protect the data if the key is good.
3. **Widespread Adoption and Implementation:** DES has been ***widely studied, tested, and implemented in various systems over the years.*** Its widespread use has contributed to its ***reliability and trustworthiness*** in many applications, particularly in legacy systems.
4. **Efficiency:** ***DES was designed to be computationally efficient,*** allowing for ***fast encryption and decryption operations even on older hardware.*** This efficiency made it suitable for a wide range of applications, including those with limited computational resources.
5. **Versatility:** DES can be easily implemented in both hardware and software, making it adaptable to different environments and platforms. ***Its versatility contributed to its widespread adoption*** and continued use in various applications for many years.

Despite these strengths, DES is now considered obsolete for modern cryptographic applications due to its small key size and vulnerability to brute-force attacks. ***It has been replaced by more secure algorithms such as AES (Advanced Encryption Standard) and Triple DES,*** which offer larger key sizes and stronger security guarantees.

Let's illustrate the avalanche effect with a simple example using a ***fictional encryption algorithm*** that operates on 4-bit blocks.
Suppose we have a plaintext block of 4 bits: **1010**, and we're using a key of 4 bits: **1101**. Our encryption algorithm takes each bit of the plaintext and performs an XOR operation with the corresponding bit of the key.

Here's how the encryption process would work:

1. Plaintext: **1010**
2. Key: **1101**
3. Encrypted Ciphertext (after XOR operation):
    - **1 XOR 1 = 0**
    - **0 XOR 1 = 1**
    - **1 XOR 0 = 1**
    - **0 XOR 1 = 1**

So, the resulting ciphertext would be **0111**.
Now, let's ***change one bit in either the plaintext or the key*** and see how it affects the ciphertext.
Let's change the first bit of the plaintext from **1** to **0**:

- Original Plaintext: **1010**
- Changed Plaintext: **0010**

When we encrypt the changed plaintext with the same key, we get:

1. Changed Plaintext: **0010**
2. Key: **1101**
3. Encrypted Ciphertext (after XOR operation):
    - **0 XOR 1 = 1**
    - **0 XOR 1 = 1**
    - **1 XOR 0 = 1**
    - **0 XOR 1 = 1**

So, the resulting ciphertext would be **1111**.
By changing just one bit in the plaintext, we've caused a significant change in the ciphertext. This ***exemplifies the avalanche effect.***

06. EXPLAIN BLOCK CIPHER MODES OF OPERATIONS.

Block cipher modes of operation are techniques used in cryptography to apply block ciphers to messages of arbitrary length. Block ciphers operate on fixed-size blocks of data, typically 64 or 128 bits, and do not directly support encryption or decryption of longer messages. Modes of operation allow block ciphers to be used to encrypt and decrypt messages of varying lengths while providing additional security features. Here are some common modes of operation:

1. **Electronic Codebook (ECB)**:

- In ECB mode, each block of plaintext is encrypted independently with the same key.
- This can lead to security vulnerabilities when the same plaintext block results in the same ciphertext block, making patterns in the data visible.
- ECB mode is not recommended for secure communications but may be suitable for deterministic encryption in certain situations.

2. **Cipher Block Chaining (CBC)**:
    - In CBC mode, each plaintext block is XORed with the previous ciphertext block before encryption.
    - It requires an initialization vector (IV) to XOR with the first plaintext block.
    - CBC mode provides confidentiality and some level of integrity protection.
    - Parallel encryption and decryption are not feasible due to the dependency on the previous ciphertext block.
3. **Cipher Feedback (CFB)**:
    - CFB mode turns a block cipher into a self-synchronizing stream cipher.
    - It encrypts plaintext blocks by XORing them with the output of the block cipher applied to the previous ciphertext block.
    - CFB mode allows for encryption of arbitrary amounts of data and supports error recovery.
4. **Output Feedback (OFB)**:
    - OFB mode is similar to CFB but operates on the output of the block cipher instead of the ciphertext.
    - It generates a keystream independent of the plaintext, which is then XORed with the plaintext to produce the ciphertext.
    - OFB mode allows for parallel encryption and decryption and supports error recovery.
5. **Counter (CTR)**:
    - CTR mode turns a block cipher into a stream cipher by using a counter to generate a unique keystream for each plaintext block.
    - The counter value is encrypted with the block cipher to produce the keystream.
    - CTR mode allows for parallel encryption and decryption and supports random access to encrypted data.
    - It does not require padding and is not susceptible to padding oracle attacks.

Each mode of operation has its own strengths and weaknesses, and the choice of mode depends on the specific requirements of the cryptographic application, such as security, performance, and data integrity.