UNIT-III

Symmetric Encryption: Symmetric encryption is a method of encryption where the same key is used for both encryption and decryption. Block ciphers are a fundamental category within symmetric encryption. They operate on fixed-size blocks of data, transforming them one block at a time.

Block Ciphers: A block cipher is a method of encrypting data in fixed-size blocks using a cryptographic key and algorithm, producing ciphertext. Unlike stream ciphers, which encrypt data one bit at a time, block ciphers process blocks simultaneously. Modern block ciphers typically handle blocks of either 64 or 128 bits.

A block cipher operates with a symmetric key and algorithm to encrypt and decrypt data. An initialization vector (IV) is crucial, enhancing security by increasing the cipher's keyspace and complicating brute force attacks. Derived from a random number generator, the IV is combined with the plaintext of the first block and the key, ensuring that subsequent ciphertext blocks are unique and do not resemble the first encryption block.

Key steps in a block cipher include dividing data into fixed-size blocks, combining the IV with the plaintext of the first block, and using the key to produce ciphertext. DES (Data Encryption Standard) and AES (Advanced Encryption Standard) are notable symmetric block ciphers. DES, designed by IBM in 1975, uses 64-bit blocks and a 56-bit key but is now insecure due to its short key size. It was replaced by AES in 1998, which uses 128-bit blocks and key sizes of 128, 192, or 256 bits, providing robust security for modern encryption needs.

Applications of Block Ciphers

1. Data Encryption:
   - o Used for encrypting private and sensitive data (e.g., passwords, credit card details) during transmission or storage.
2. File and Disk Encryption:
   - o Used by software like BitLocker and TrueCrypt to encrypt entire files and disks, protecting contents from unauthorized access.
3. Virtual Private Networks (VPN):
   - o Encrypts data transmitted between devices over the internet, ensuring data security during transmission.
4. Secure Sockets Layer (SSL) and Transport Layer Security (TLS):
   - o Encrypts data between web browsers and servers, securing sensitive information like login credentials and credit card details.
5. Digital Signatures:
   - o Provides authenticity and integrity to digital documents by generating unique signatures for verification and detecting malicious activities.

Feistel Networks:

The Feistel cipher is a symmetric structure used to construct block ciphers, named after the German cryptographer Horst Feistel. This model is the basis for many block ciphers, including the well-known Data Encryption Standard (DES). A key feature of the Feistel cipher is that the same algorithm is used for both encryption and decryption, simplifying the design and implementation.

Characteristics:

Invertible Components: The Feistel cipher can have components that are invertible, non-invertible, or self-invertible.

Round Keys: Different keys are used for each round, but the same set of keys is used for both encryption and decryption

.Feistel Cipher Algorithm

Preparation:

Convert the plaintext to ASCII and then to an 8-bit binary format

.Divide the binary plaintext string into two halves: left half (L1) and right half (R1).

Generate random binary keys (K1 and K2) for each round, with lengths equal to half the length of the plaintext.

Encryption:

First Round:

Generate function (f1) using (R1) and (K1): f1 = XOR(R1, K1)).

Compute the new left half (L2) and right half (R2):

R2 = XOR(f1, L1)

L2 = R1

Second Round:

Generate function (f2) using (R2) and (K2): f2 = XOR(R2, K2)).

Compute the new left half (L3) and right half (R3):

R3 = XOR(f2, L2)

L3 = R2

Concatenate (R3) and (L3) to form the ciphertext.

Decryption:

The same algorithm is used to retrieve the plaintext from the ciphertext.

Example:

Plaintext: 'Hello"

Ciphertext: 'E1!w("

Retrieved Plaintext: 'Hello"

Plaintext: 'Geeks"

Ciphertext: "O;Q"

Retrieved Plaintext: "Geeks"

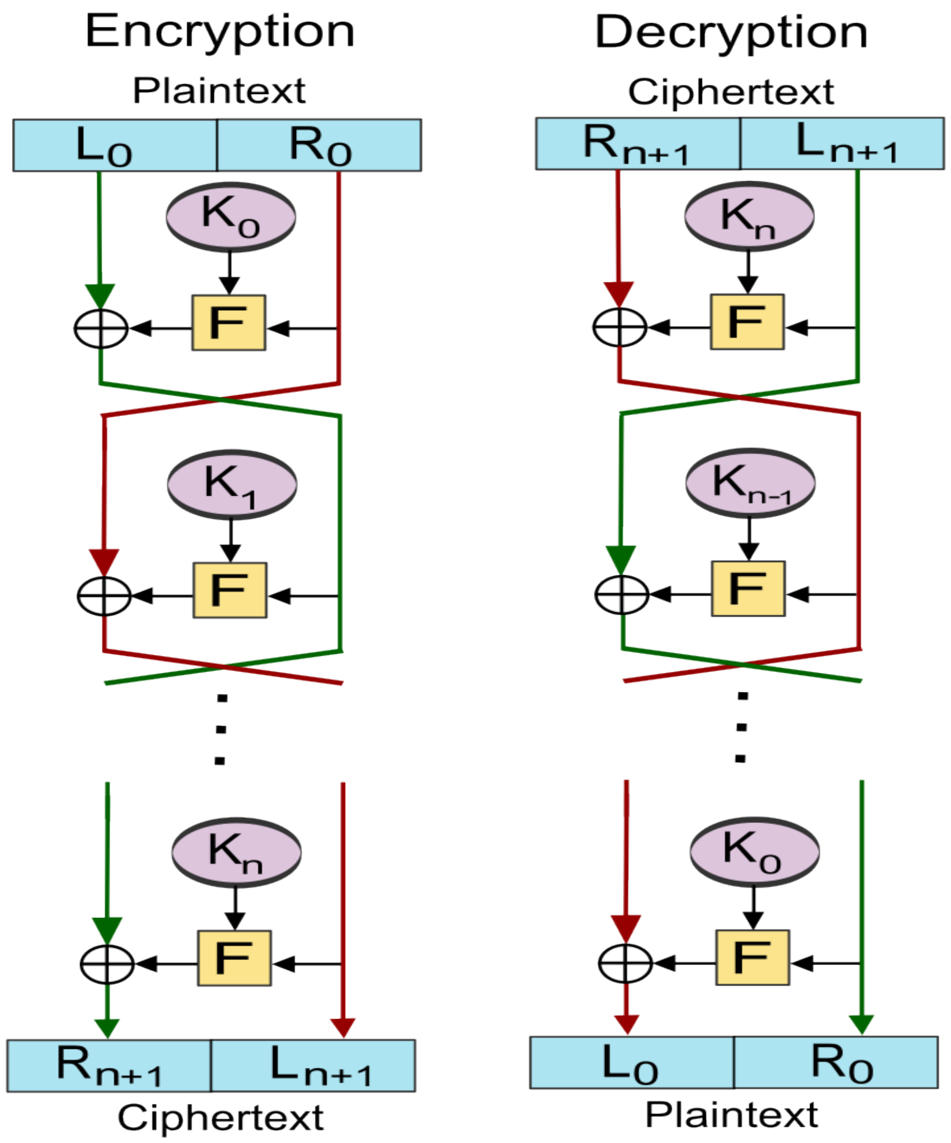
Advantages:

Symmetric key usage simplifies key management.

The same algorithm for encryption and decryption reduces complexity.

Disadvantages:

The security of a Feistel cipher depends heavily on the complexity and robustness of the function fff.

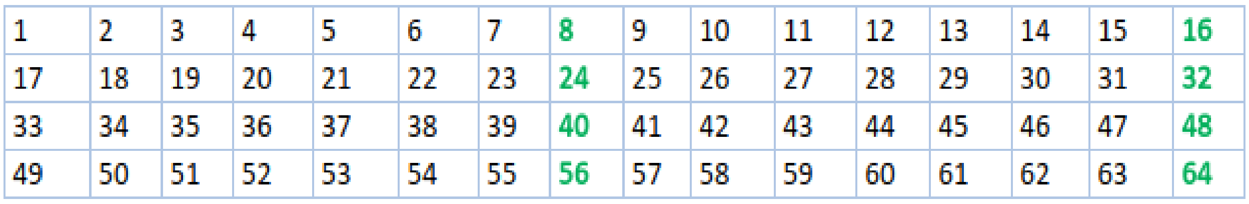


Data Encryption Standard (DES):

The Data Encryption Standard (DES) is a historic encryption algorithm that has significantly contributed to the field of data security. DES is a symmetric block cipher, meaning the same key is used for both encryption and decryption, and it operates on fixed-size blocks of data. Initially designed by IBMand adopted as a federal standard in 1977, DES has a key length of 56 bits, making it vulnerable to brute-force attacks with today's computational power. Despite its vulnerabilities, DES has paved the way for more secure algorithms like the Advanced Encryption Standard (AES).

What is DES?

DES encrypts data in 64-bit blocks, transforming each block into ciphertext using a series of well-defined operations. The process involves both substitution (confusion) and transposition (diffusion), which are fundamental principles of cryptography. Although DES uses a 56-bit key, the initial key is actually 64 bits long. However, every 8th bit is discarded, reducing it to 56 bits before the encryption process begins.



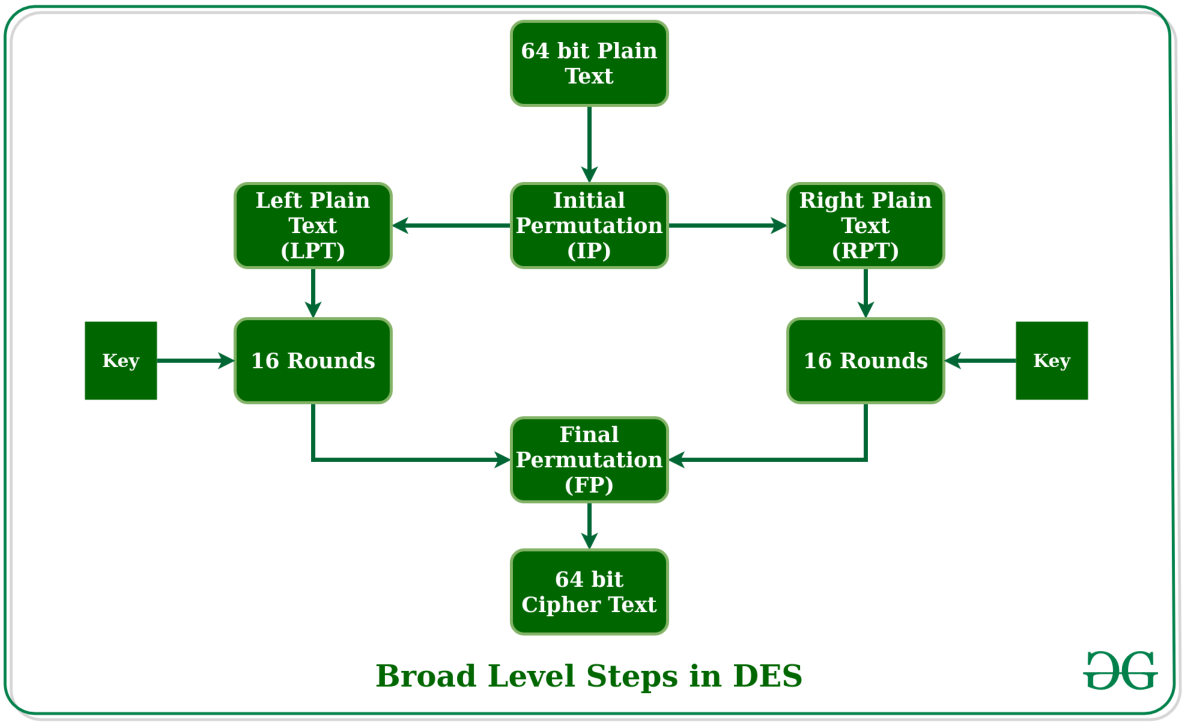
1	2	3	4	5	6	7	**8**	9	10	11	12	13	14	15	**16**
17	18	19	20	21	22	23	**24**	25	26	27	28	29	30	31	**32**
33	34	35	36	37	38	39	**40**	41	42	43	44	45	46	47	**48**
49	50	51	52	53	54	55	**56**	57	58	59	60	61	62	63	**64**

Figure - discording of every 8th bit of original key

Key Concepts of DES:

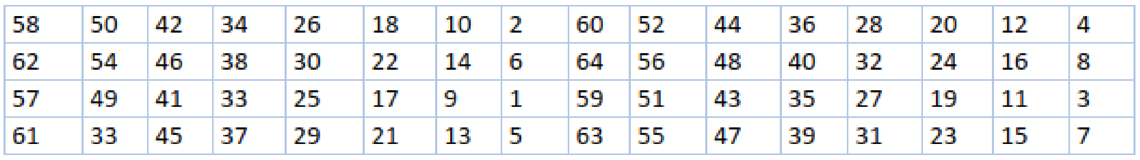
1. Initial Permutation (IP): The plaintext is subjected to an initial permutation, which rearranges the bits according to a predefined table. This is a simple bitwise permutation and does not involve any key.
2. Rounds: DES consists of 16 rounds of processing, each involving substitution and transposition. Each round uses a different 48-bit subkey derived fromthe original 56-bit key.
3. Final Permutation (FP): After 16 rounds, the two halves of the data block are combined and subjected to a final permutation to produce the ciphertext.

Detailed Steps of DES:



1. Initial Permutation (IP):

The 64-bit plaintext block is permuted according to a fixed table. For example, the first bit of the plaintext is replaced by the 58th bit, the second bit by the 50th bit, and so on. This step ensures that the bits are shuffled in a specific pattern before the actual encryption process begins.



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	33	45	37	29	21	13	5	63	55	47	39	31	23	15	7

Figure - Initial permutation table

2. Division into Halves:

The permuted block is split into two 32-bit halves: Left Plain Text (LPT) and Right Plain Text (RPT).

3. Key Transformation:

The original 64-bit key is reduced to a 56-bit key by discarding every 8th bit.

The 56-bit key is then divided into two 28-bit halves. For each round, these halves are circularly shifted left by one or two positions, depending on the round number.

After shifting, a subset of 48 bits is selected fromthe 56-bit key using a process called Compression Permutation. This subset serves as the subkey for that round.

4. Expansion Permutation:

The 32-bit RPT is expanded to 48 bits using an expansion permutation. This involves splitting the 32 bits into eight 4-bit blocks and then expanding each block to 6 bits.

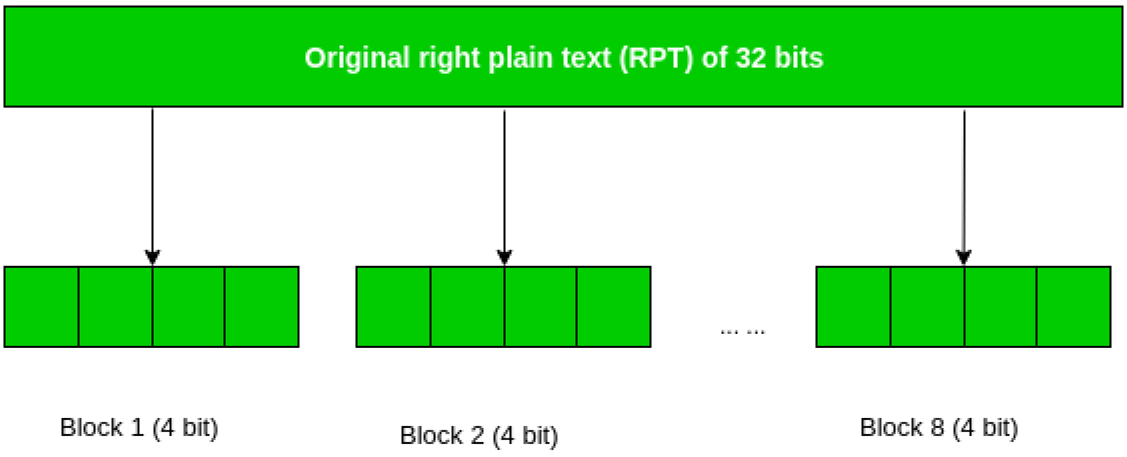


Figure - division of 32 bit RPT into 8 bit blocks

5. XOR Operation:

The 48-bit expanded RPT is XORed with the 48-bit subkey generated in the key transformation step.

6. Substitution (S-Box):

The result of the XOR operation is divided into eight 6-bit blocks. Each block is then substituted using a predefined substitution box (S-Box), which maps the 6-bit input to a 4-bit output. DES uses eight different S-Boxes, one for each block.

7. Permutation (P-Box):

The 32-bit output fromthe S-Box substitution is permuted again using a fixed permutation table. This step ensures that the bits are shuffled according to a specific pattern.

8. XOR with LPT:

The permuted output is XORed with the original 32-bit LPT to produce the new RPT for the next round. The original RPT becomes the new LPT.

9. Repetition:

Steps 4 to 8 are repeated for 16 rounds, with each round using a different subkey.

10. Final Permutation (FP):

After 16 rounds, the two halves are recombined and subjected to a final permutation, which is the inverse of the initial permutation. This step produces the 64-bit ciphertext.

Example of DES Encryption

Let's walk through an example of encrypting a 64-bit block of plaintext using DES:

Plaintext: "Hello"

- Convert the plaintext to its ASCII values and then to binary.
- Apply the initial permutation to the 64-bit binary data.
- Split the permuted block into two 32-bit halves.
- Perform the key transformation to generate 16 subkeys.
- For each round:
  - o Expand the 32-bit RPT to 48 bits.
  - o XOR the expanded RPT with the subkey.
  - o Substitute using the S-Boxes.
  - o Permute the substituted output.
  - o XOR the permuted output with the LPT.
  - o Swap the halves for the next round.
- Combine the two halves after 16 rounds and apply the final permutation.

Key Characteristics of DES:

1. Substitution (Confusion): The S-Box substitution step ensures that the relationship between the key and the ciphertext is complex.
2. Transposition (Diffusion): The initial and final permutations, along with the P-Box permutation, ensure that the influence of each plaintext bit is spread out over many ciphertext bits.

Security of DES

DES was once considered secure, but advances in computational power have made it vulnerable to brute-force attacks. The 56-bit key length is too short by modern standards, allowing attackers to try all possible keys within a reasonable time frame. As a result, DES has been largely replaced by more secure algorithms like AES (Advanced Encryption Standard).

Applications of DES

Despite its vulnerabilities, DES has historical significance and has been used in various applications:

- Encryption of Financial Data: DES was widely used for encrypting banking and financial transactions.
- Secure Communications: It was employed in securing communications, especially in military and government applications.
- Legacy Systems: Some older systems and protocols still use DES for compatibility reasons.

Conclusion

The Data Encryption Standard (DES) has played a crucial role in the history of cryptography. Although its security is no longer adequate for most modern applications, understanding DES provides valuable insights into the development of encryption algorithms. The principles of substitution and transposition used in DES continue to influence the design of contemporary cryptographic systems. As we move towards more secure algorithms like AES, the legacy of DES remains an important part of the cryptographic landscape.

Triple-DES (3DES):

Triple DES: An Overview

Triple DES (3DES) is an encryption algorithm developed to enhance the security of the original Data Encryption Standard (DES). While DES became vulnerable to brute-force attacks due to its 56-bit key length, Triple DES provides a more secure alternative by applying the DES algorithm three times to each data block. This method significantly increases the encryption strength, making it harder to break compared to the original DES.

What is Triple DES?

Triple DES, also known as 3DES or TDES, is a symmetric key block cipher. It operates on 64-bit blocks of data and uses a series of three DES encryption and decryption operations, which enhances the security compared to DES. Despite being less efficient and slower than the Advanced Encryption Standard (AES), Triple DES is still used in many legacy systems due to its backward compatibility and enhanced security features.

Features of Triple DES

1. Triple Encryption: Triple DES applies the DES algorithm three times to each data block. This triple-layer encryption increases the security of the data significantly.
2. Variable Key Sizes: It supports key sizes ranging from 128 bits to 192 bits, depending on the variant used.
3. Symmetric Key Encryption: The same key is used for both encryption and decryption, simplifying the key management process.
4. Block Cipher: It encrypts data in 64-bit blocks, processing each block individually.
5. Legacy System Compatibility: Triple DES is suitable for systems that require secure encryption but still rely on DES.

Encryption Process of Triple DES

1. Key Generation

In the first step of Triple DES, three unique keys are generated. These keys are used in the subsequent encryption and decryption processes. The key generation process ensures that the keys are unique and suitable for providing strong encryption.

2. Initial Permutation (IP)

After key generation, the plaintext undergoes an initial permutation. This step rearranges the bits of the plaintext according to a predefined table, preparing the data for the encryption rounds.

3. Three Rounds of Encryption

The core of the Triple DES encryption process involves three rounds of DES encryption:

First Round: The plaintext is encrypted using the first key (K1).

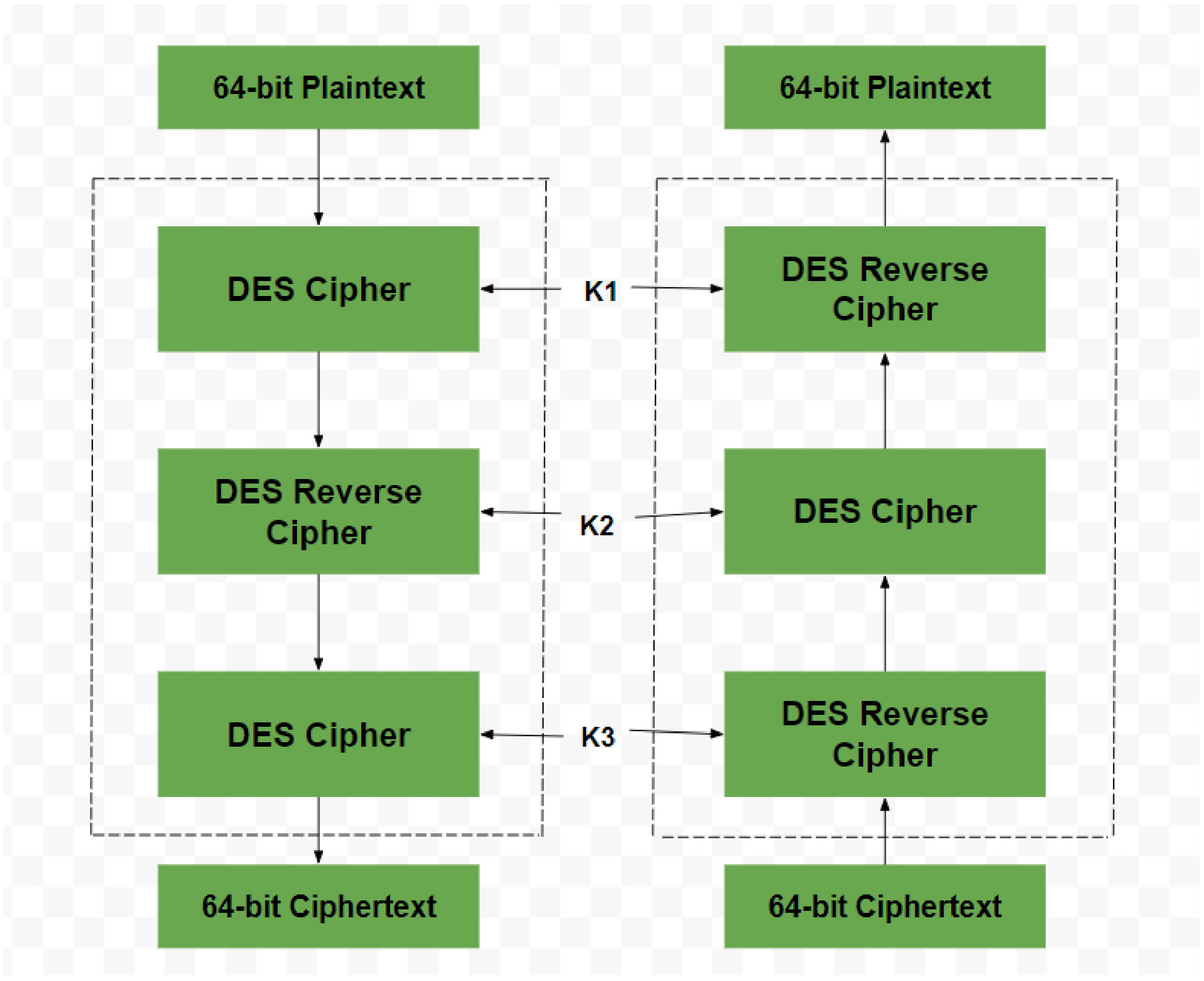
Second Round: The resulting ciphertext is decrypted using the second key (K2).

Third Round: The output fromthe second round is encrypted again using the third key (K3).

Each round involves the typical DES processes of substitution and permutation, significantly enhancing the security of the data.

4. Final Permutation (FP)

The final permutation is the last step in the Triple DES encryption process. This step rearranges the bits of the resulting ciphertext block to their original order. The final permutation is the inverse of the initial permutation, ensuring the data is correctly processed.



Advantages of Triple DES

1. Enhanced Security: The three-layer encryption technique provides a high level of

security, making it much more difficult for attackers to break the encryption compared to single DES.

2. Backward Compatibility: Triple DES can work with systems designed for DES, making it a suitable upgrade for legacy systems.
3. Variable Key Sizes: The support for variable key sizes allows for flexibility in the level of security.
4. Wide Usage: Triple DES is a widely recognized and used encryption standard, making it compatible with many encryption protocols and standards.

Applications of Triple DES

1. Financial Transactions: Triple DES is commonly used to secure financial transactions, such as online banking and credit card payments, ensuring the security of sensitive financial information.
2. Data Protection: It is used to protect sensitive data stored on computers, servers, and other electronic devices, across various sectors including healthcare and government.
3. Virtual Private Networks (VPNs): Triple DES secures communications between remote locations by encrypting data transmitted over VPNs.
4. Authentication and Digital Signatures: Triple DES, combined with cryptographic hash functions, is used to generate digital signatures and verify the authenticity of digital documents and messages.

Frequently Asked Questions (FAQs)

1. What is the key size of DES?

The key size used in DES is 56-bit.

2. What type of algorithm is 3DES?

The type of algorithm used in Triple DES is the Triple Data Encryption Algorithm, a symmetric key block cipher.

3. How many rounds does 3DES perform when encrypting data?

Triple DES encryption typically performs 48 rounds in total, comprising three sets of 16 rounds each, depending on the variant and desired level of security.

Conclusion

Triple DES is a robust encryption algorithm designed to improve upon the security limitations of the original DES. While it is slower and less efficient than modern algorithms like AES, its backward compatibility and enhanced security features make it a valuable encryption standard, especially for legacy systems and specific applications requiring strong encryption. Understanding the workings and benefits of Triple DES helps in appreciating its role in the evolution of cryptographic algorithms and data security.

Advanced Encryption Standard (AES):

The Advanced Encryption Standard (AES) is a widely trusted encryption algorithm established by the U.S. National Institute of Standards and Technology (NIST) in 2001. AES is a symmetric key block cipher, meaning it uses the same key for both encryption and decryption, and it encrypts data in fixed-size blocks of 128 bits. The algorithm can use key sizes of 128, 192, or 256 bits, offering varying levels of security. Known for its efficiency and robust security, AES has become a cornerstone of modern cryptography, securing data in numerous applications from internet communications to file encryption.

What is Advanced Encryption Standard (AES)?

AES is a highly trusted encryption algorithm used to convert data into an unreadable format without the proper key. It was developed to replace the older Data Encryption Standard (DES) due to DES's vulnerability to brute-force attacks. AES operates on blocks of data and applies multiple rounds of transformation, involving substitution and permutation, to secure the data. It is recognized globally for its strength in protecting information from unauthorized access.

Key Features of AES

1. Block Cipher: AES encrypts data in blocks of 128 bits each.
2. Variable Key Sizes: AES supports key sizes of 128, 192, and 256 bits, allowing for varying levels of security.
3. Substitution-Permutation Network: AES uses a combination of substitution and permutation operations to transform the input data.
4. Symmetric Key Encryption: The same key is used for both encryption and decryption.
5. Efficiency: Despite being complex, AES is designed to be efficient and fast in both hardware and software implementations.

Working of AES

1. Data Blocks and Byte Processing

AES operates on bytes rather than bits, processing data in 128-bit blocks (16 bytes). The number of rounds of transformation depends on the key length:

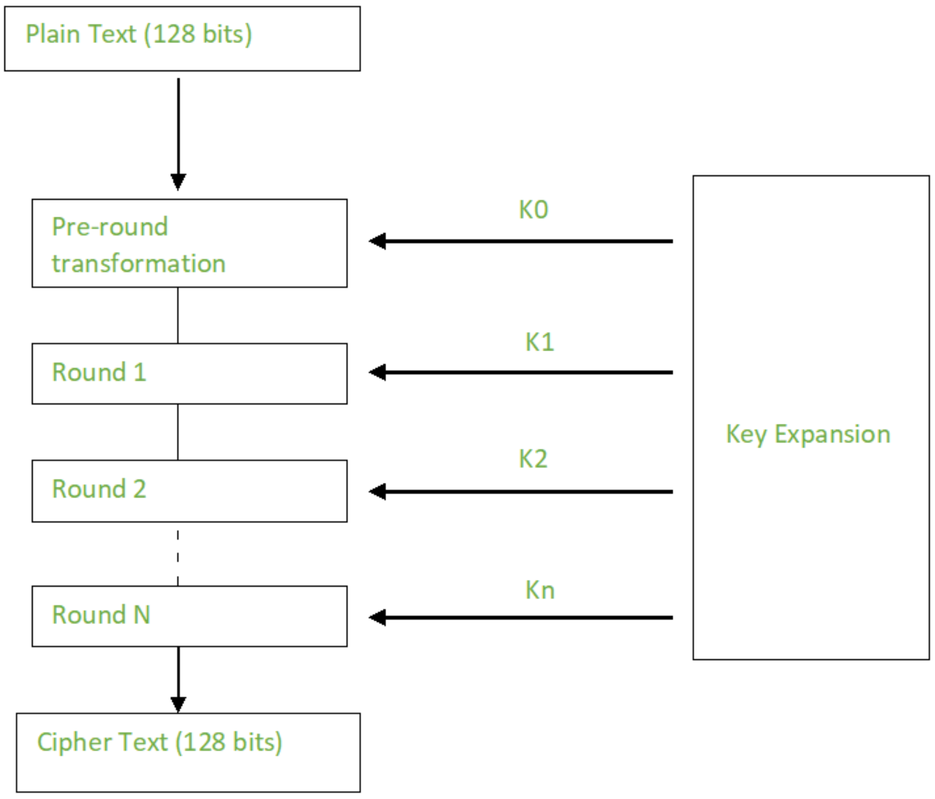
128-bit key: 10 rounds

192-bit key: 12 rounds

256-bit key: 14 rounds

2. Creation of Round Keys

A Key Schedule algorithm generates a series of round keys from the initial key. Each round key is used in the corresponding round of encryption or decryption. The key schedule algorithm ensures that each round key is unique and derived from the initial key in a secure manner.



3. Encryption Process

```
[ b0 | b4 | b8 | b12 |
| b1| b5 | b9 | b13 |
| b2 | b6 | b10| b14 |
| b3 | b7 | b11| b15 ]
```

AES encryption involves several rounds, each consisting of four main steps:

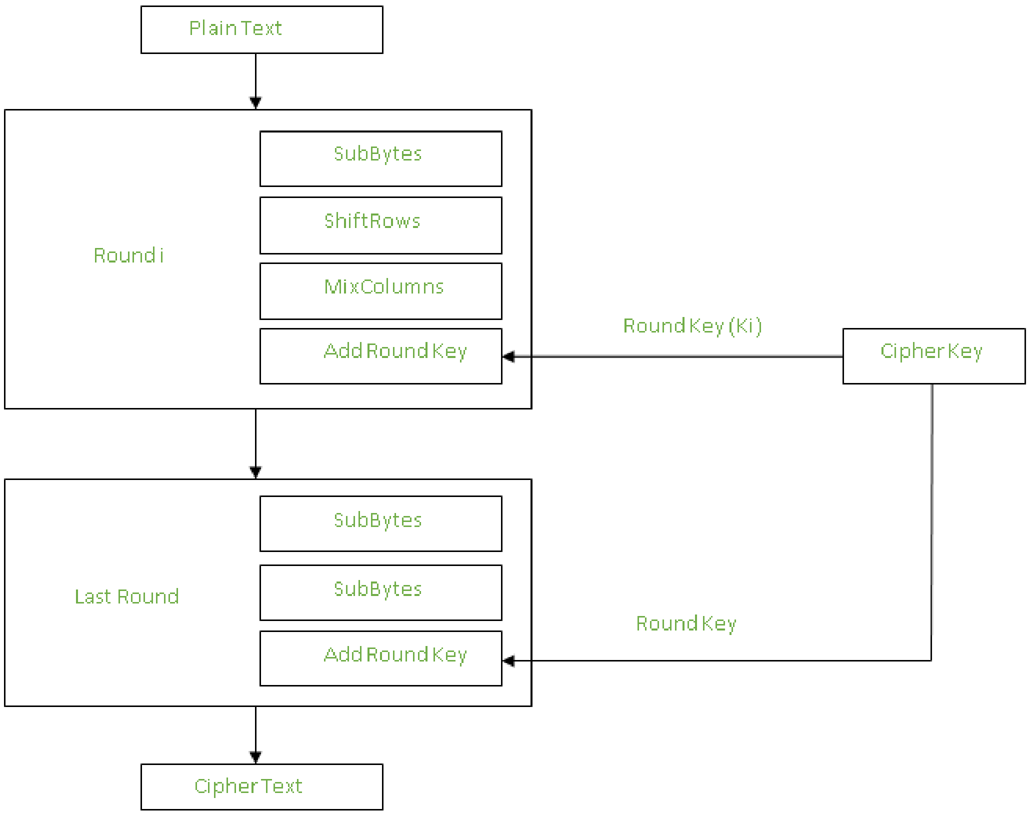
1. SubBytes: This step substitutes each byte of the block using a lookup table called the S-box. The substitution is designed to prevent bytes frombeing substituted by themselves or their complements.
2. ShiftRows: This step shifts the rows of the block matrix cyclically to the left. The number of positions each row is shifted depends on the row index:
   - o First row. No shift
   - o Second row. Shifted left by one position
   - o Third row. Shifted left by two positions
   - o Fourth row. Shifted left by three positions

```
[ b0  | b1 | b2  | b3  ]           [ b0  | b1 | b2  | b3  ]
| b4  | b5  | b6  | b7  |    ->   | b5  | b6  | b7  | b4  |
| b8  | b9  | b10 | b11 |          | b10 | b11 | b8  | b9  |
[ b12 | b13 | b14 | b15 ]          [ b15 | b12 | b13 | b14 ]
```

3. MixColumns: This step mixes the bytes in each column by multiplying them with a fixed matrix. This operation changes the positions of bytes and creates diffusion in the block.

```
[ c0 ]       [ 2  3  1  1]  [ b0 ]
| c1 |  =    | 1  2  3  1|  | b1 |
| c2 |       | 1  1  2  3|  | b2 |
[ c3 ]       [ 3  1  1  2]  [ b3 ]
```

4. AddRoundKey: The block is XOR-ed with the round key derived from the initial key. This step combines the block data with the round key, ensuring that each round has a unique transformation.



In the final round, the MixColumns step is omitted, and the block undergoes only SubBytes, ShiftRows, and AddRoundKey transformations.

4. Decryption Process:

The decryption process reverses the encryption steps. The stages of each round of decryption are as follows:

Add Round Key

Inverse MixColumns: Uses a different matrix for multiplication compared to the encryption MixColumns step.

Inverse ShiftRows: Shifts the rows cyclically to the right.

Inverse SubBytes: Uses an inverse S-box for byte substitution.

Each block of data is decrypted by applying these steps in reverse order, using the round keys generated during the key schedule phase.

Applications of AES

AES is used in various applications requiring secure data storage and transmission, including:

1. Wireless Security: AES secures Wi-Fi networks by encrypting data to prevent unauthorized access.
2. Database Encryption: AES encrypts sensitive data stored in databases, protecting personal information and financial records.
3. Secure Communications: AES is used in protocols such as SSL/TLS for secure internet communications, email, instant messaging, and voice/video calls.
4. Data Storage: AES encrypts data stored on hard drives, USB drives, and other storage media, protecting it fromunauthorized access.
5. Virtual Private Networks (VPNs): AES secures data transmitted between a user’s device and a remote server, ensuring privacy and security.
6. Password Storage: AES encrypts passwords before storage, adding an extra layer of security to user credentials.
7. File and Disk Encryption: AES encrypts files and folders on computers and external storage devices, protecting sensitive data during transfer and storage.

Conclusion

The Advanced Encryption Standard (AES) is a robust encryption algorithmthat has remained secure for over two decades. Integrated into CPUs to improve speed and security, AES offers strong protection against cyber threats and is widely implemented in various applications. While no vulnerabilities have been found in the algorithmitself, the implementation of AES must be done correctly to ensure its effectiveness.

Frequently Asked Questions on Advanced Encryption Standard (AES)

1. What is the AES Advanced Encryption Standard?

AES is a symmetric key encryption algorithmthat encrypts and decrypts data using the same key. It applies multiple rounds of transformation to ensure data security.

2. What is the AES encryption code?

AES, also known as the Rijndael algorithm, is a symmetric block cipher that works with 128-bit data blocks and uses keys of 128, 192, or 256 bits.

3. Where is AES used?

AES is used in wireless security, protecting processors, encrypting files, securing internet connections with SSL/TLS, and in various applications by government agencies, businesses, and other organizations to keep sensitive data safe.

Stream Ciphers:

Stream ciphers are a type of symmetric key cipher where plaintext digits are combined with a pseudorandom cipher digit stream (keystream). Unlike block ciphers, which process data in fixed-size blocks, stream ciphers encrypt data one bit or byte at a time. This makes them particularly efficient for certain applications, especially those requiring real-time encryption, such as video streaming and online gaming.

How Stream Ciphers Work

1. Initialization:
   - o A key (k) is provided as input to a pseudorandom bit generator.
   - o The generator produces a keystream of random bits or bytes.
2. Encryption:
   - o Each bit or byte of plaintext is XOR-ed with the corresponding bit or byte of the keystream.
   - o The result is the ciphertext.
3. Decryption:
   - o The same keystream is used to XOR with the ciphertext.
   - o This operation reverses the encryption process, revealing the original plaintext.

Example

Encryption:

Plaintext: 10011001

Keystream: 11000011

Ciphertext: 01011010 (Result of XOR operation between plaintext and keystream)

Decryption:

Ciphertext: 01011010

Keystream: 11000011

Plaintext: 10011001 (Result of XOR operation between ciphertext and keystream)

Key Points of Stream Cipher

1. Pseudorandom Number Stream:
   - o Stream ciphers rely on a pseudorandom number stream to generate the keystream.

2. Cryptanalysis Resistance:

   o The keystream must be long and complex to resist cryptanalysis.

3. Brute Force Attack Resistance:

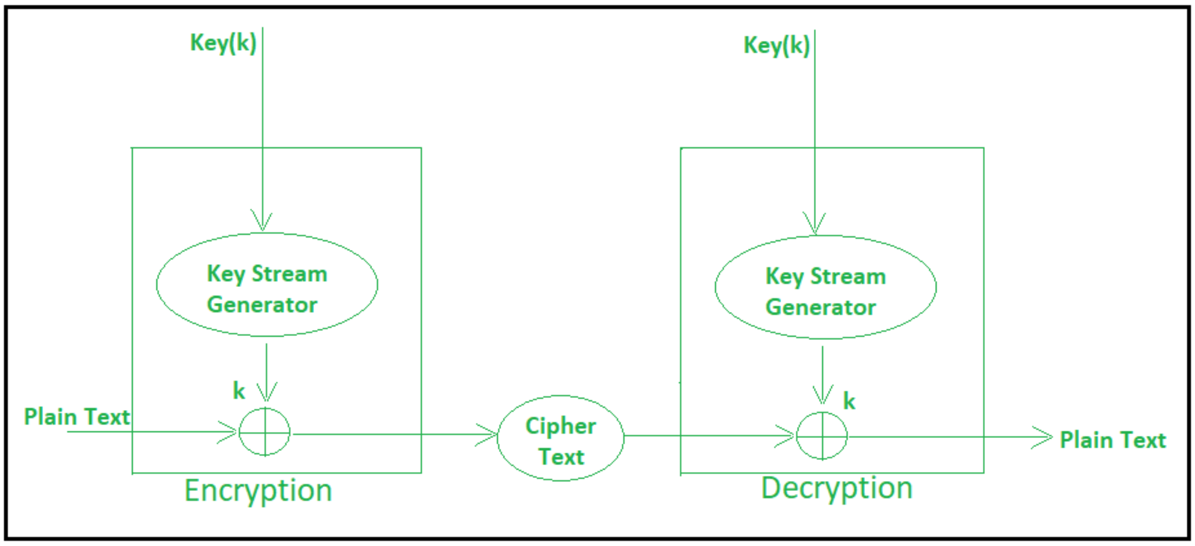
   o Longer keys provide stronger security, making brute force attacks more difficult.

4. Efficient Keystream Design:

   o An effective keystream contains a balanced mix of 1s and 0s to increase cryptanalysis difficulty.

5. Simplicity:

   o Stream ciphers require fewer lines of code compared to block ciphers, making them easier to implement.



RC4:

RC4 is one of the most widely used stream ciphers.

Operates by initializing a permutation of all 256 possible bytes and then generating a pseudorandom output byte for each input byte.

There are many other stream ciphers, each with varying costs, speeds, and complexity. A list of 25 distinct kinds of stream ciphers is available on Wikipedia.

Advantages of Stream Ciphers

1. Speed:

   o Generally faster than block ciphers because they process data bit-by-bit or byte-by-byte.

2. Low Complexity:

   o Easier to implement with less sophisticated hardware requirements.

3. Sequential Nature:

   o Ideal for real-time applications where data is processed and transmitted in a continuous stream.

4. Accessibility:
   - o Symmetric encryption avoids the need for public and private key management, simplifying the encryption process.

Disadvantages of Stream Ciphers

1. Error Propagation:
   - o Errors during transmission can corrupt subsequent bits, affecting the entire message.
2. Key Management:
   - o Maintaining and distributing keys can be challenging, especially in large systems or networks.
3. Predictability:
   - o If the keystream is not properly designed, it may be predictable or vulnerable to attacks, compromising security.

Stream Cipher Examples and Use Cases

1. Real-Time Communication:

Stream ciphers are ideal for real-time applications such as video streaming and online gaming because they can encrypt and decrypt data as it is transmitted.

2. Wireless Communication:

Used in wireless communication protocols to ensure secure data transmission.

3. Secure Sockets Layer (SSL) and Transport Layer Security (TLS):

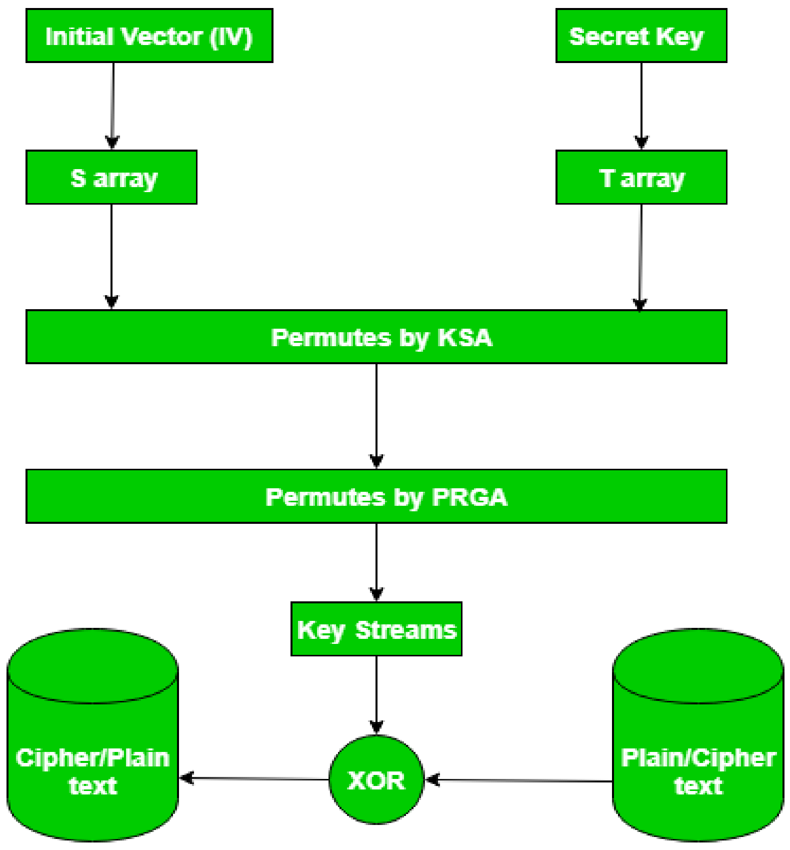
Stream ciphers are employed in SSL/TLS protocols to secure internet communications.

Conclusion

Stream ciphers offer a balance of speed, simplicity, and efficiency, making them suitable for many applications requiring real-time data encryption. However, their use requires careful key management and keystream design to avoid vulnerabilities and ensure robust security. Despite their advantages, the inherent risk of error propagation and key management complexities must be addressed to maintain data integrity and security.

RC4:

RC4 is a widely used stream cipher and variable-length key algorithm that encrypts data one byte at a time. It was designed by Ron Rivest in 1987 for RSA Security. Despite its popularity, RC4 is no longer recommended for use due to several vulnerabilities that have been discovered over time.



How RC4 Works

1. Initialization:
   - o A key (k) is input into a pseudorandom bit generator, producing a stream of 8-bit numbers (keystream).
   - o This keystream is combined one byte at a time with the plaintext using the XOR operation to produce the ciphertext.
2. Encryption:
   - o Plaintext byte and keystream byte undergo XOR operation to produce the ciphertext.
   - o Example:

     Plaintext: 1001000

     Keystream: 0101000

     Ciphertext: 1100100 (Result of XOR operation)
3. Decryption:
   - o Ciphertext byte and keystream byte undergo XOR operation to recover the plaintext.
   - o Example:

     Ciphertext: 1100100

     Keystream: 0101000

Plaintext: 10011000 (Result of XOR operation)

Key-Generation Algorithm

A variable-length key from 1 to 256 bytes initializes a 256-byte state vector SSS, with elements S[0]S[0]S[0] to S[255]S[255]S[255]. For encryption and decryption, a byte kkk is generated from SSS by selecting one of the 255 entries in a systematic fashion, followed by permuting the entries in SSS again.

Key-Scheduling Algorithm: Initialization

The entries of SSS are set equal to the values from 0 to 255 in ascending order.

A temporary vector TTT is created. If the length of the key kkk is 256 bytes, then kkk is assigned to TTT. Otherwise, for a key with length kkk-len bytes, the first kkk-len elements of TTT are copied from kkk, and then kkk is repeated as many times as necessary to fill TTT.

Permutation of SSS

Starting with S[0]S[0]S[0] to S[255]S[255]S[255], for each S[i]S[i]S[i] algorithm swaps it with another byte in SSS according to a scheme dictated by T[i]T[i]T[i].

Pseudo-random Generation Algorithm (Stream Generation)

Once the vector SSS is initialized, the input key is no longer used. For each S[i]S[i]S[i], the algorithm swaps it with another byte in SSS according to the current configuration of SSS. After reaching S[255]S[255]S[255], the process continues from S[0]S[0]S[0] again.

Features of RC4 Encryption Algorithm:

Symmetric Key Algorithm:

- o RC4 uses the same key for both encryption and decryption.

Stream Cipher Algorithm:

- o RC4 encrypts and decrypts data one byte at a time.

Variable Key Size:

- o RC4 supports variable key sizes from 40 bits to 2048 bits, offering flexibility for different security requirements.

Fast and Efficient:

- o Suitable for low-power devices and high-speed data transmission applications.

Widely Used:

- o Historically used in various applications, including wireless networks, SSL, VPNs, and file encryption.

Advantages of RC4

1. Speed and Efficiency:
   - o RC4 is fast and efficient, making it suitable for applications where speed is

critical.

2. Simplicity:
    o RC4 is relatively simple to implement in both software and hardware.
3. Variable Key Size:
    o Supports variable key sizes, providing flexibility for different security needs.
4. Historical Usage:
    o Widely used in various applications due to its simplicity and efficiency.

Disadvantages of RC4

1. Vulnerabilities:
    o RC4 has several known vulnerabilities, such as a bias in the first few bytes of the keystream, which can be exploited to recover the key.
2. Security Weaknesses:
    o Inherent weaknesses in its design make RC4 less secure compared to other encryption algorithms like AES or ChaCha20.
3. Limited Key Length:
    o The maximum key length of 2048 bits may not be sufficient for some applications requiring stronger encryption.
4. Not Recommended for New Applications:
    o Due to its vulnerabilities and weaknesses, RC4 is no longer recommended for use in new applications. More secure stream cipher algorithms, such as AES-CTR or ChaCha20, should be used instead.

Conclusion

While RC4 has played a significant role in encryption over the years, its vulnerabilities have made it obsolete for modern applications. Current best practices recommend using more secure alternatives like AES-CTR or ChaCha20 to ensure robust data protection.

MODES OF OPERATION:

Encryption algorithms are divided into two categories based on the input type: block ciphers and stream ciphers. A block cipher is an encryption algorithm that processes fixed-size blocks of input data (say bbb bits) and produces ciphertext of the same size. If the input exceeds bbb bits, it is divided into multiple blocks for encryption. Block ciphers are commonly used with various modes of operation to meet different application needs.

Modes of Operation

1. Electronic Code Book (ECB) Mode:

Description: ECB is the simplest mode of block cipher operation. It directly encrypts each block of plaintext independently, resulting in blocks of encrypted ciphertext. If the message is larger than bbb bits, it is divided into blocks, and the process is repeated for each block.Each plaintext block is encrypted separately.

Advantages:

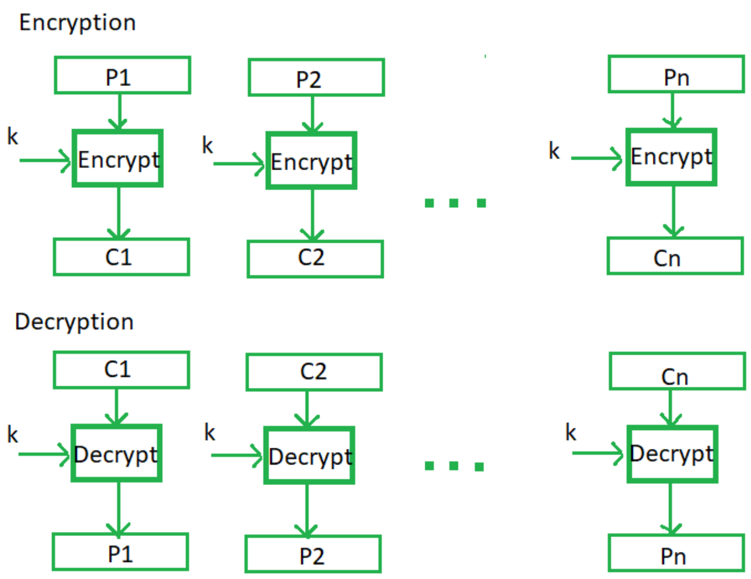
Allows parallel encryption of blocks, making it faster.

Simple and straightforward to implement.

Disadvantages:

Prone to cryptanalysis due to the direct relationship between plaintext and ciphertext blocks.

Identical plaintext blocks result in identical ciphertext blocks, revealing patterns.



2. Cipher Block Chaining (CBC) Mode:

Description: CBC improves upon ECB by enhancing security. In CBC, the previous ciphertext block is combined with the current plaintext block using an XOR operation before encryption. An initialization vector (IV) is used for the first block to ensure each encryption is unique.

Advantages:

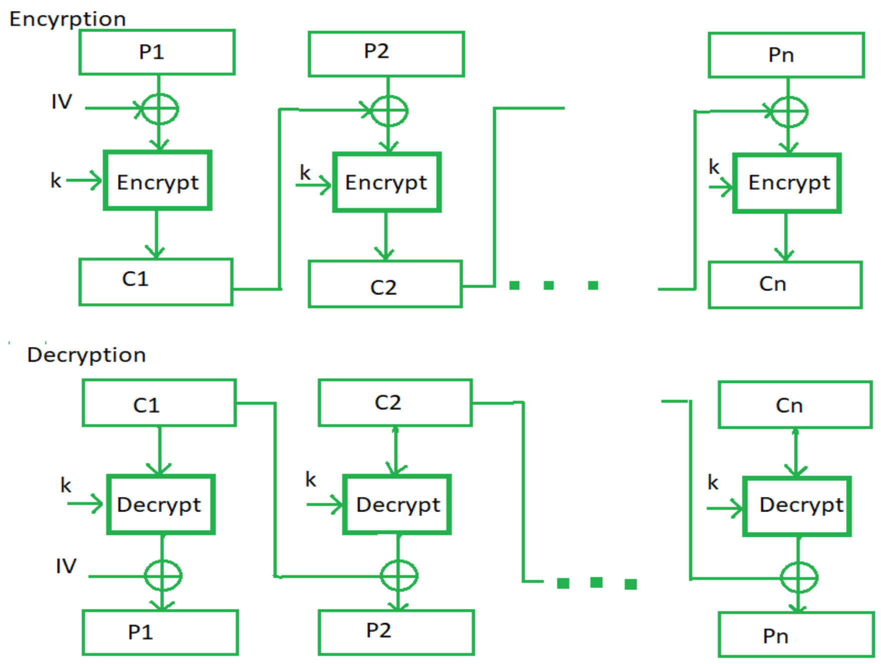
More secure than ECB, providing better resistance to cryptanalysis.

Suitable for encrypting data larger than bbb bits.

Provides a good authentication mechanism

Disadvantages:

Encryption cannot be parallelized because each block depends on the previous one.



3. Cipher Feedback (CFB) Mode:

Description: CFB mode treats the block cipher as a streamcipher. It uses an IV for the first encryption and divides output bits into sets. The leftmost sss bits are selected and combined with plaintext bits via XOR. The result is input to a shift register, and the process continues.The shift register updates for the next block.

Advantages:

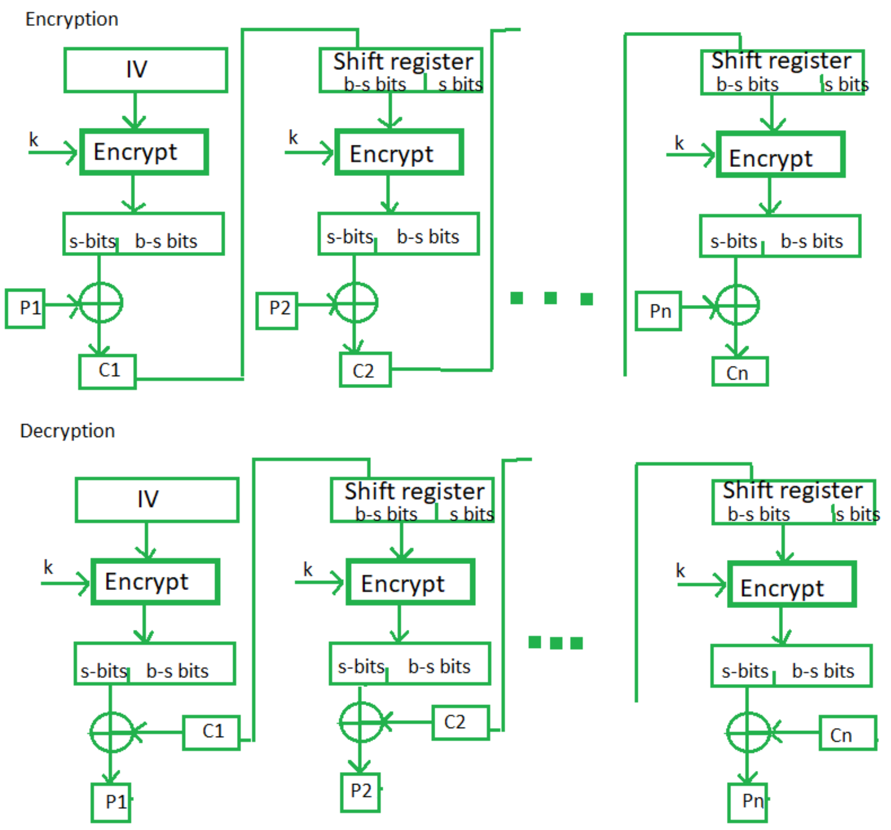
Provides resistance to cryptanalysis by introducing data loss through the shift register.

Decryption is parallelizable and loss-tolerant.

Disadvantages:

Encryption is not parallelizable.

Propagates errors in the ciphertext.



4. Output Feedback (OFB) Mode:

Description: OFB mode is similar to CFB, but instead of feeding the actual ciphertext back, it feeds the encrypted output. This mode avoids error propagation and reduces the dependency of the ciphertext on the plaintext.

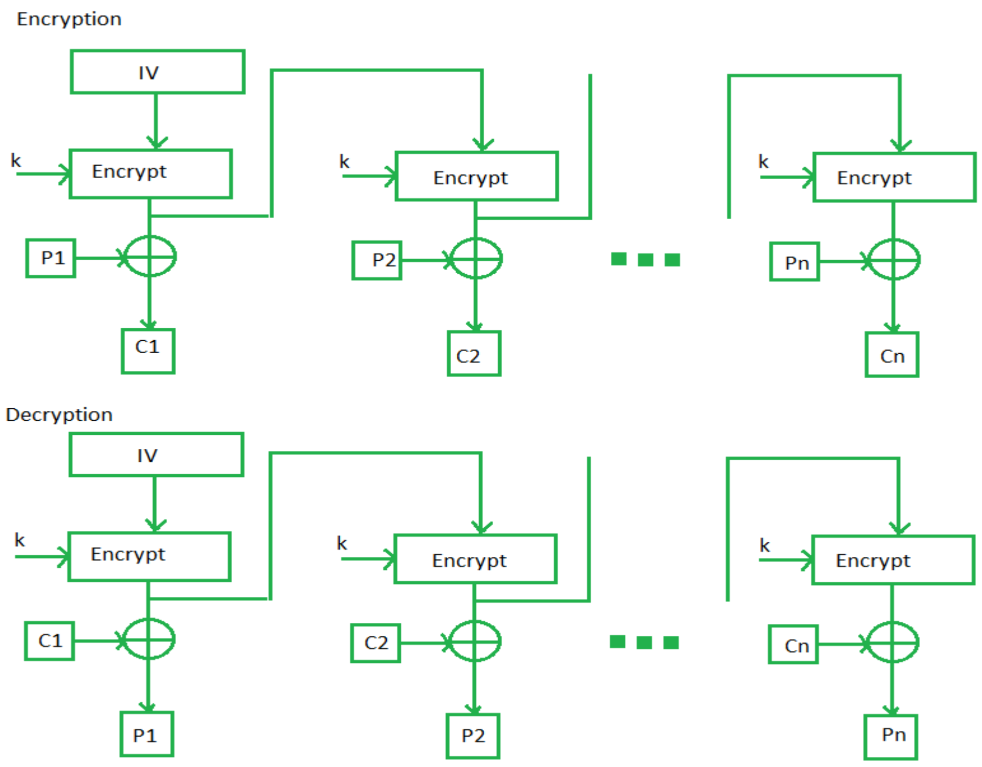
Advantages:

Free from bit errors in plaintext blocks.

Does not propagate single-bit errors to subsequent blocks.

Disadvantages:

More susceptible to message stream modification attacks compared to CFB.



5. Counter (CTR) Mode:

Description: CTR mode uses a counter-based approach. Each counter-initiated value is encrypted and combined with the plaintext block using XOR to produce the ciphertext. It allows for parallel processing.

Advantages:

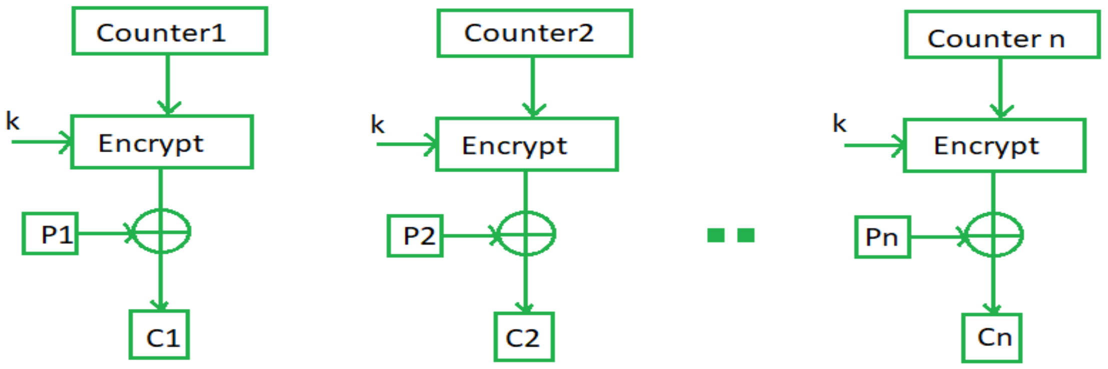
Avoids direct plaintext-ciphertext relationships.

Supports parallel encryption since blocks are independent.

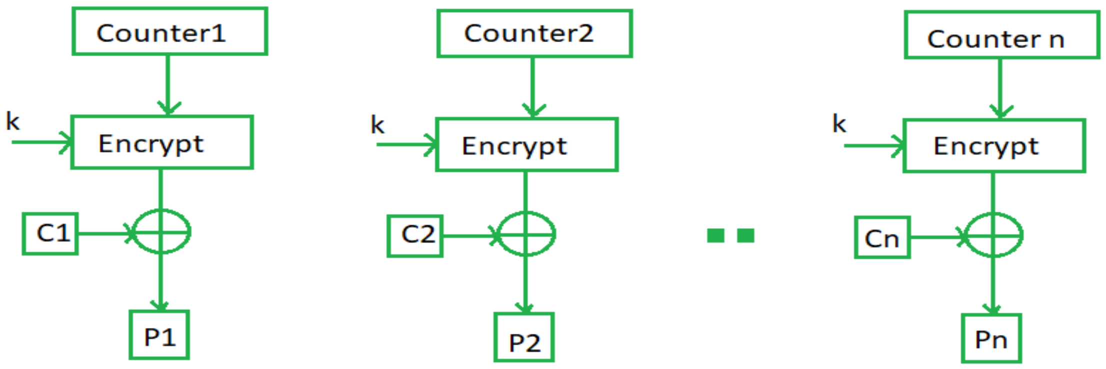
Disadvantages:

Requires synchronized counters at both ends. Loss of synchronization leads to erroneous decryption.

Encryption



Decryption


Counter1
k
Encrypt
C1
P1
Counter2
k
Encrypt
C2
P2
Counter n
k
Encrypt
Cn
Pn