NETWORK SECURITY AND CRYPTOGRAPHY

UNIT 02

Symmetric key cryptography:

Symmetric key cryptography is a type of encryption where the same key is used for both encrypting and decrypting information. This requires that both the sender and the receiver have access to the same secret key.

The security of this system is entirely dependent on the secrecy of the key. If the key falls into the wrong hands, the encrypted data becomes vulnerable, as anyone with access to the key can decrypt the information.

Symmetric key cryptography is generally faster than asymmetric cryptography, making it suitable for encrypting large amounts of data.

Popular symmetric key algorithms include **AES** (Advanced Encryption Standard), **DES** (Data Encryption Standard), and **Blowfish**, each offering different levels of security and performance.

Types of Symmetric Key Cryptography

1. Block Ciphers
2. Stream Ciphers

1. Block Cipher

A block cipher is a type of symmetric key cryptography that encrypts data in fixed-size blocks. The process involves taking a block of plaintext (usually 64 or 128 bits) and applying a cryptographic algorithm along with a secret key to produce a block of ciphertext of the same size. The same key is used for both encryption and

decryption, meaning the ciphertext can be transformed back into the original plaintext by applying the same algorithm with the same key.

How Block Ciphers Work:

1. **Plaintext Block:** The data to be encrypted is divided into fixed-size blocks (e.g., 128 bits).
2. **Encryption:** Each block is encrypted separately using a key and an algorithm. The result is a block of ciphertext.
3. **Decryption:** To decrypt, the same key and algorithm are applied to the ciphertext block, reversing the process to recover the original plaintext.

The most common block cipher algorithms are:

Advanced Encryption Standard (AES)

- It has support for three-length keys: 128 bits, 192 bits, or 256 bits, the most commonly used one is a 128-bit key.
- It includes secure communication, data encryption in storage devices, digital rights management (DRM), and so on.

Data Encryption Standard (DES)

- In DES, the 64-bit blocks of plaintext are encrypted using a 56-bit key.
- This weakness caused by the small key size led to the development of a more secure algorithm, called AES.

Triple Data Encryption Algorithm (Triple DES)

- The development of the Triple DES, also called Triple-DES or TDEA, was triggered by the weak security resulting from the small key size in the DES.
- Triple DES denotes a method of three times applying the DES algorithm sequentially (encrypt-decrypt-encrypt) on every plaintext block.

2. Stream Cipher

A stream cipher is a type of symmetric key cryptography that encrypts data one bit or byte at a time, rather than in blocks. It generates a stream of pseudorandom bits, called a keystream, which is then XORed with the plaintext to produce the ciphertext. The keystream is generated based on the key and the initial state of the cipher. Like block ciphers, the same key is used for both encryption and decryption.

How Stream Ciphers Work:

1. **Keystream Generation:** A keystream is generated based on the secret key. The length of the keystream matches the length of the plaintext.
2. **Encryption:** The plaintext is XORed with the keystream on a bit-by-bit or byte-by-byte basis to produce the ciphertext.
3. **Decryption:** The ciphertext is XORed with the same keystream to recover the original plaintext, as XORing twice cancels out the keystream.

The most common stream cipher algorithms are:

Rivest Cipher 4 (RC4)

- **Strengths:** The initial appeal of RC4 came from its efficient design and capability to handle variable-length data streams.
- **Current Status**: Due to these identified weaknesses, RC4 is no longer considered secure for most applications. Its use is strongly prohobited by cryptographic standards bodies.

Salsa20

- **Strengths:** It's fast and efficient, with a simple and elegant design. Most importantly, the security it offers against known attacks is robust. Apart from that, Salsa20 serves as a building block for other cryptographic protocols, exhibiting its versatility.
- **Current Status:** Salsa20 is a very widely used and well-respected stream cipher. It's used for many applications where performance and security balance.

Grain-128

- **Strengths of Grain-128 include** efficiency, lightweight implementation, and the ability to perform well with limited processing power and memory, making it ideal for radio frequency identification (RFID) tags and sensor networks. Importantly, Grain-128 still provides strong security with such simplicity.
- **Current Status:** Grain-128 is useful in some resource-limited situations where an application needs to be run with huge restrictions in the amount of data that is available for use.

Techniques Used in Symmetric Key Cryptography:

Substitution and **Transposition** are two principal techniques used in symmetric-key cryptography.

➢ Substitution Techniques:

The symmetric key cryptographic method employs one secret key for the operations of encryption and decryption. Substitution techniques provide two significant approaches, wherein elements (letters, characters) from the plaintext message are replaced with new elements according to the rules based on the secret key.

- **Caesar Cipher:** Caesar cipher has since their predictability is so complete and no complexity is invested.
- **Monoalphabetic Ciphers:** This is where the ciphers use one rule of substitution throughout the message. This may involve replacing letters with numbers, symbols, or another set of letters in another order.
- **Playfair Cipher:** Implementation of repeated letters or letter pairs can expose patterns, and cryptanalysis techniques exist to exploit them.
- **Hill Cipher:** This cipher operates on blocks of letters (typically bigrams or trigrams) using a matrix multiplication approach. The Hill ciphers have a limitation on key size and susceptibility towards cryptanalysis for larger key sizes.
- **Polyalphabetic Ciphers:** This is the type of cipher where any one of the letters in the plaintext is substituted by a different letter to keep frequency analysis challenging. For example, the Vigenère cipher operates with a keyword that would determine the shift value for each letter in the plaintext.

- **One-Time Pad (OTP):** It is a theoretically impossible cipher where the key is a random string of characters that is exactly as long as the message itself. The key is used for a single encryption and then discarded.

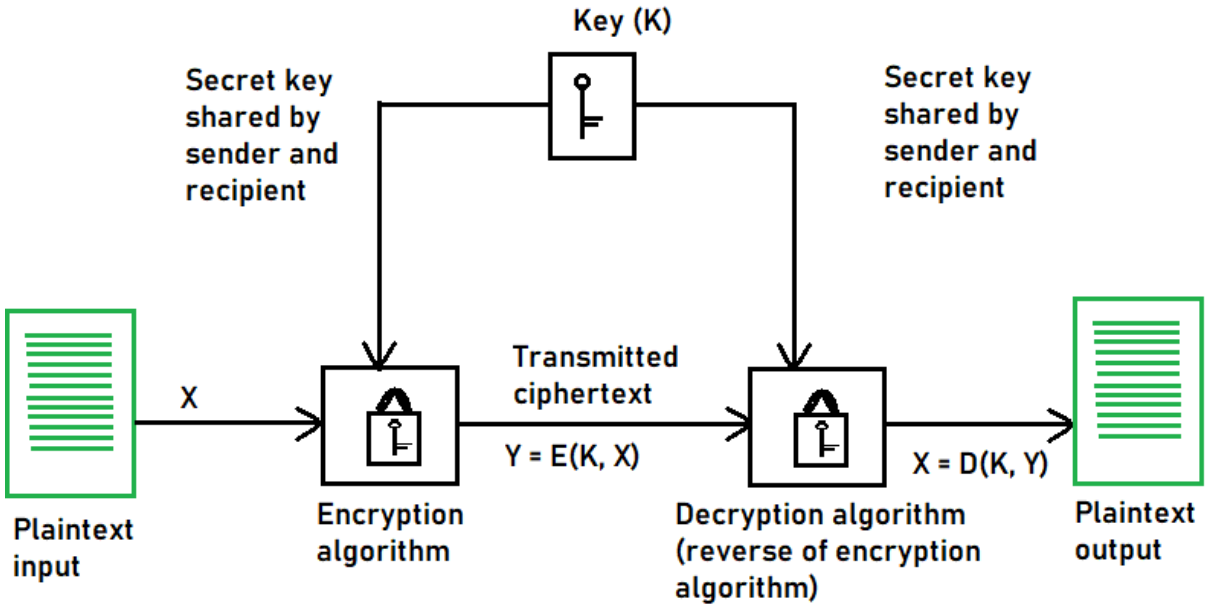


Diagram of Symmetric Encryption

➢ Transposition Techniques:

Transposition techniques rearrange the order of elements in the plaintext message without changing the elements themselves.

- **Rail Fence Cipher:** This is a simple cipher that rearranges the elements by writing the plaintext message in a zigzag pattern, with the different components written in rows (rails) of an imaginary fence and then reading through the columns in a standard order. The key to this is the number of rails used.
- **Columnar Transposition:** In the case of a plaintext message written in columns and then the columns rearranged according to a permutation determined by the key, this cipher is known as columnar transposition. Although it is still vulnerable to cryptanalysis techniques that exploit the statistical properties of the language.

Attacks on Symmetric Key Cryptography:

Several types of attacks can target symmetric key cryptography, exploiting weaknesses in the algorithms, key management, or implementation. Here are some of the most common attacks:

1. **Brute-Force Attack**

- **Description:** An attacker systematically tries every possible key until the correct one is found. The effectiveness of this attack depends on the key length; shorter keys are more vulnerable.
- **Countermeasure:** Use long keys (e.g., 128-bit or 256-bit keys) to make brute-force attacks impractical due to the sheer number of possible keys.

2. **Cryptanalysis Attack**

- **Description:** Involves analyzing the ciphertext to find patterns or weaknesses in the encryption algorithm that can be exploited to determine the key or decrypt the data.
- **Types:**
  - **Linear Cryptanalysis:** Explores linear approximations of the encryption algorithm.
  - **Differential Cryptanalysis:** Studies how differences in plaintext affect differences in ciphertext.
- **Countermeasure:** Use algorithms like AES, which are resistant to known cryptanalytic attacks.

3. **Key Guessing Attack**

- **Description:** Similar to brute-force but relies on guessing the key based on common patterns, weak key choices, or predictable key generation processes.
- **Countermeasure:** Use strong, random key generation processes and avoid weak or predictable keys.

Advantages of Symmetric Key Cryptography:

1. **Fast and Efficient:** Ideal for encrypting large amounts of data quickly.
2. **Simple Implementation:** Uses the same key for both encryption and decryption, making it easy to implement.
3. **Low Resource Requirements:** Suitable for devices with limited computational power.
4. **Widely Supported:** Compatible across various platforms and systems.
5. **Great for Bulk Data:** Well-suited for securing large datasets.

Disadvantages of Symmetric Key Cryptography:

1. **Key Distribution Challenges:** Securely sharing the key between parties is difficult.
2. **Scalability Issues:** Managing keys for many users becomes complex.
3. **Single Point of Failure:** If the key is compromised, all encrypted data is at risk.
4. **No Built-in Authentication:** It doesn't verify the identity of the sender.
5. **Lack of Non-repudiation:** Cannot prove who encrypted the data, limiting its use in scenarios requiring legal proof.

Popular symmetric key algorithms include

- Data Encryption Standard(DES)
- International Data Encryption Algorithm (IDEA)
- Advanced Encryption Standard(AES)
- RC4
- Blowfish

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). It was adopted in 1977 for government agencies to protect sensitive data and was officially retired in 2005.

Data Encryption Standard (DES) is a block cipher with a 56-bit key length that has played a significant role in data security**.** Data encryption standard (DES) has been found vulnerable to very powerful attacks therefore, the popularity of DES has been found slightly on the decline. DES is a block cipher and encrypts data in blocks of size of **64 bits** each, which means 64 bits of plain text go as the input to DES, which produces 64 bits of ciphertext. The same algorithm and key are used for encryption and decryption, with minor differences. The key length is **56 bits**, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only).

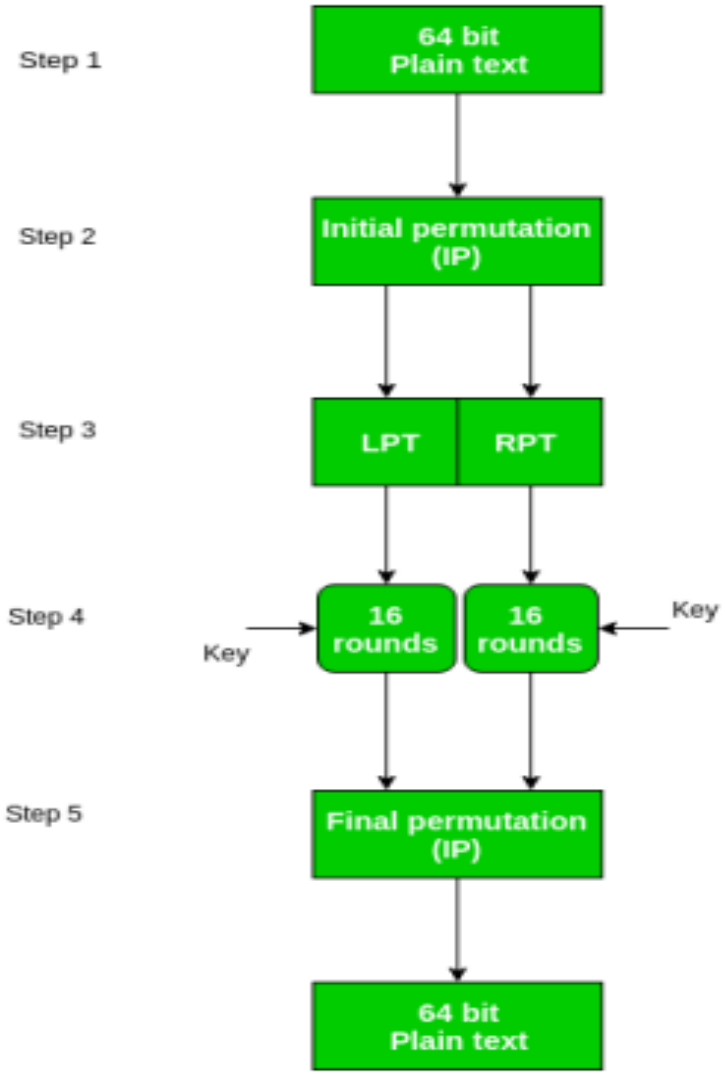
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64

Figure - discording of every 8th bit of original key

Working of data encryption standard:

Broad-level steps in DES:

1. In the first step, the 64 bit plain text block is handed over to an initial Permutation (IP) function.
2. The initial permutation performed on plain text.
3. Next the initial permutation (IP) produces two halves of the permuted block; says Left Plain Text (LPT) and Right Plain Text (RPT).
4. Now each LPT and RPT to go through 16 rounds of encryption process.
5. In the end, LPT and RPT are rejoined and a Final Permutation (FP) is performed on the combined block.
6. The result of this process produces 64 bit cipher text.



Working of DES in detail:

1. Initial Permutation (IP):

❖ **Purpose:** This is the first step in the DES algorithm where the plaintext block is permuted.

❖ **Process:**

- The 64-bit plaintext is divided into smaller 64-bit chunks if the data is larger.

- The initial permutation rearranges these bits according to a predefined table. For example, the 58th bit of the original data moves to the 1st position, the 50th to the 2nd, and so on.
- After the permutation, the 64-bit block is split into two 32-bit halves: Left Plain Text (LPT) and Right Plain Text (RPT).

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. Key Transformation

- **Purpose:** To derive the subkeys used in each round of encryption.
- **Process:**
  - DES originally uses a 64-bit key, but only 56 bits are used in the actual process, with the 8th bit of each byte serving as a parity check and being discarded.
  - The 56-bit key is split into two 28-bit halves.
  - These halves are shifted left circularly by one or two bits depending on the round number.
  - After shifting, the 56-bit key is compressed into a 48-bit key through a compression permutation. This 48-bit key is used in the following steps.

For example, if the round number 1, 2, 9 or 16 the shift is done by only position for other rounds, the circular shift is done by two positions. The number of key bits shifted per round is show in figure.

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
#key bits shifted	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Figure - number of key bits shifted per round

After an appropriate shift, 48 of the 56 bit are selected. for selecting 48 of the 56 bits the table show in figure given below. For instance, after the shift, bit number 14 moves on the first position, bit number 17 moves on the second position and so on. If we observe the table carefully, we will realize that it contains only 48 bit

positions. Bit number 18 is discarded (we will not find it in the table), like 7 others, to reduce a 56-bit key to a 48-bit key. Since the key transformation process involves permutation as well as selection of a 48-bit sub set of the original 56-bit key it is called Compression Permutation.

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

Figure - compression permutation

Because of this compression permutation technique, a different subset of key bits is used in each round. That's make DES not easy to crack.

3. Expansion Permutation

- **Purpose:** To expand the 32-bit RPT to 48 bits to match the size of the key for the XOR operation.
- **Process:**
  - The 32-bit RPT is expanded by dividing it into eight 4-bit blocks.
  - Each of these 4-bit blocks is expanded to 6 bits by duplicating some bits.
  - The expansion involves permuting the bits so that the input 32 bits become 48 bits.
  - This expanded RPT is then XORed with the 48-bit key derived in the previous step.

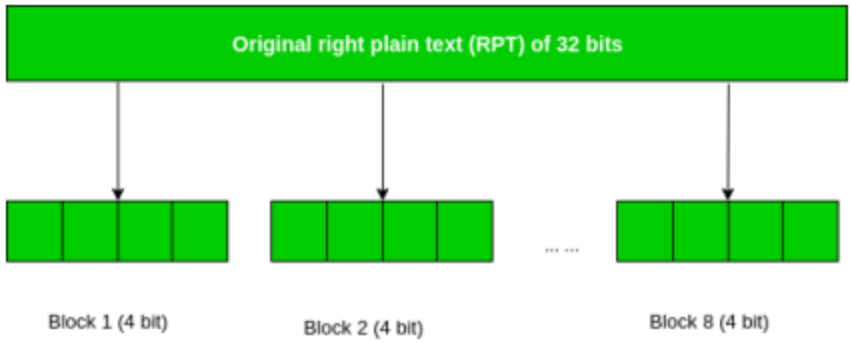


Figure - division of 32 bit RPT into 8 bit blocks

This process results into expansion as well as permutation of the input bit while creating output. Key transformation process compresses the 56-bit key to 48 bits. Then the expansion permutation process expands the 32-bit RPT to 48-bits. Now the 48-bit key is XOR with 48-bit RPT and resulting output is given to the next step, which is the S-Box substitution.

International Data Encryption Algorithm:

IDEA stands for International Data Encryption Algorithm. IDEA is a symmetric key block cipher encryption algorithm, which means it uses the same key for both encryption and decryption. It was developed by James Massey and Xuejia Lai and published in 1991. It was developed to provide a high level of security and efficiency, making it one of the notable encryption algorithms of its time. It has a 128-bit key length and works with 64-bit blocks.

IDEA encryption operates on 64-bit blocks and uses a 128-bit key. However, the blocks get split into four internal blocks of 16 bits. Thus, the encryption occurs in a series of eight rounds.
The algorithm uses six keys for every round (each key is 16 bits). Since IDEA encryption completes eight rounds, it uses 48 subkeys in total. Additionally, its key schedule generates four extra keys for transforming the output in the last round. Other procedures relevant to this process are bitwise XOR, multiplication modulo, and addition modulo.

Working of International Data Encryption Algorithm:

The **International Data Encryption Algorithm (IDEA)** is a **symmetric key block** cipher that:

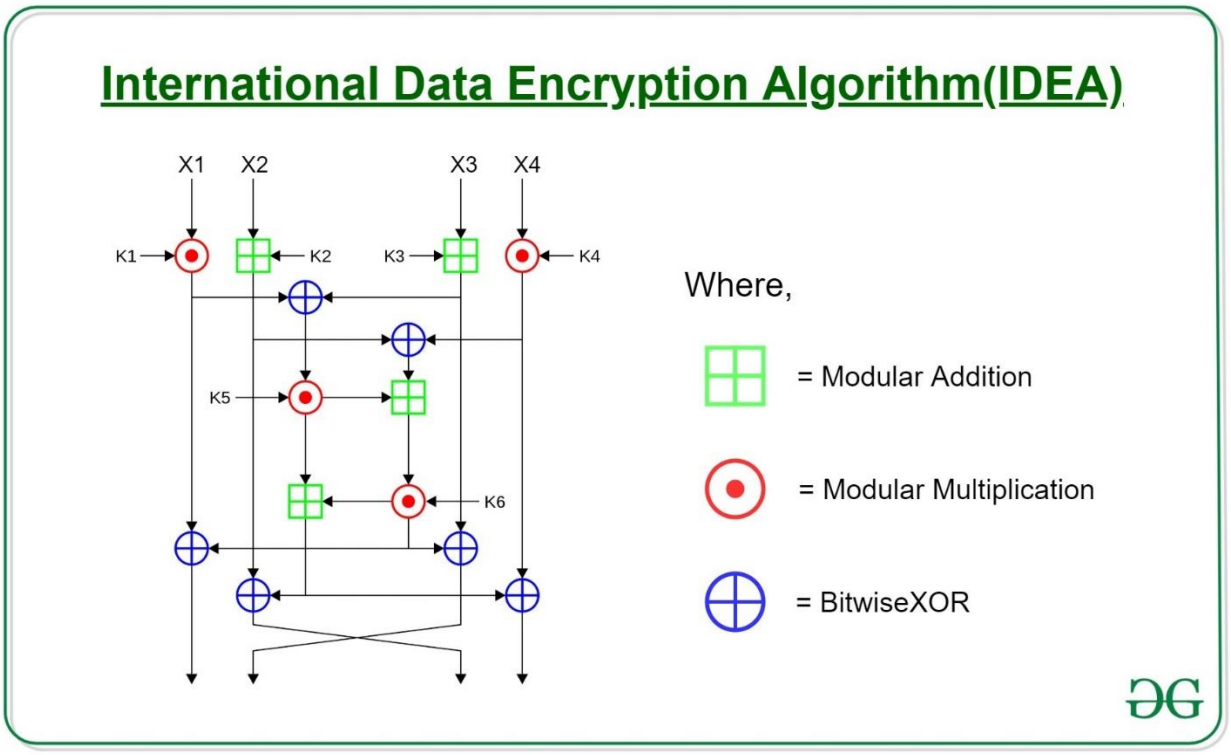
- uses a fixed-length plaintext of **16 bits** and
- encrypts them in **4 chunks of 4 bits** each
- to produce **16 bits ciphertext**.
- The length of the key used is **32 bits**.
- The key is also divided into 8 blocks of 4 bits each.

This algorithm involves a series of 4 identical complete rounds and 1 half-round. Each complete round involves a series of 14 steps that includes operations like:

- modular addition
- modular multiplication
- bitwise exclusive-OR (XOR)

After 4 complete rounds, the final "half-round" consists of only the first 4 out of the 14 steps previously used in the full rounds. To perform these rounds, each binary notation must be converted to its equivalent decimal notation, perform the operation and the result obtained should be converted back to the binary representation for the final result of that particular step.

Key Schedule: 6 subkeys of 4 bits out of the 8 subkeys are used in each complete round, while 4 are used in the half-round. So, 4.5 rounds require 28 subkeys. The given key, 'K', directly gives the first 8 subkeys. By rotating the main key left by 6 bits between each group of 8, further groups of 8 subkeys are created, implying less than one rotation per round for the key (3 rotations).



	K1	K2	K3	K4	K5	K6
Round 1	1101	1100	0110	1111	0011	1111
Round 2	0101	1001*	0001	1011	1100	1111
Round 3	1101	0110	0111	0111*	1111	0011
Round 4	1111	0101	1001	1101	1100	0110*
Round 4.5	1111	1101	0110	0111		

NOTE: * denotes a shift of bits

The 16-bit plaintext can be represented as **X1 || X2 || X3 || X4**, each of size 4 bits. The 32-bit key is broken into 8 subkeys denoted as K1 || K2 || K3 || K4 || K5 || K6 || K7 || K8, again of size 4 bits each. Each round of 14 steps uses the three

algebraic operation-Addition modulo (2^4), Multiplication modulo (2^4)+1 and Bitwise XOR. The steps involved are as follows:

1. X1 * K1
2. X2 + K2
3. X3 + K3
4. X4 * K4
5. Step 1 ^ Step 3 I,e Bitwise XOR the results of steps 1 and 3
6. Step 2 ^ Step 4 I,e Bitwise XOR the results of steps 2 and 4.
7. Step 5 * K5
8. Step 6 + Step 7
9. Step 8 * K6
10. Step 7 + Step 9
11. Step 1 ^ Step 9
12. Step 3 ^ Step 9
13. Step 2 ^ Step 10
14. Step 4 ^ Step 10

The input to the next round is Step 11 || Step 13 || Step 12 || Step 14, which becomes X1 || X2 || X3 || X4. This swap between 12 and 13 takes place after each complete round, except the last complete round (4th round), where the input to the final half round is Step 11 || Step 12 || Step 13 || Step 14.

After last complete round, the half-round is as follows:

1. X1 * K1
2. X2 + K2
3. X3 + K3
4. X4 * K4

After completing all rounds and the final half-round, the four 16-bit blocks are concatenated to produce the final 64-bit ciphertext.

Decryption

Decryption in IDEA is the reverse of the encryption process. However, it uses the same set of subkeys in reverse order, with the operations (addition, multiplication, XOR) being reversed accordingly.

- The multiplication in the encryption process becomes a multiplicative inverse in the decryption.
- The addition becomes subtraction (modulo 2^{16}).
- XOR remains the same since it is its own inverse.

What is RC4?

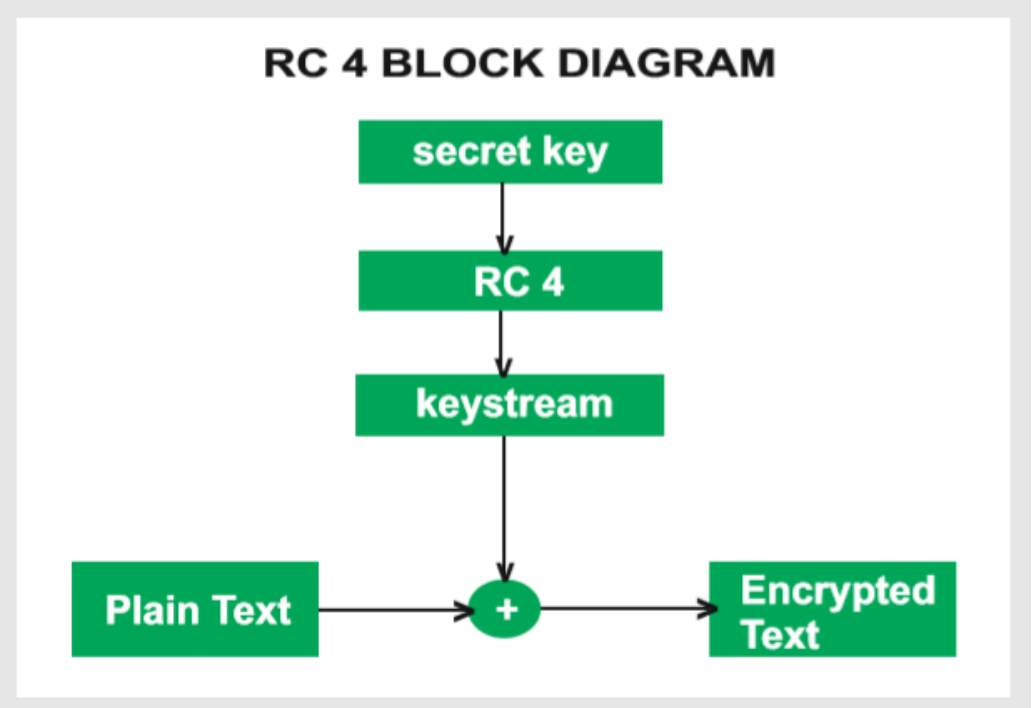
RC4 stands for Rivest Cipher 4. Ron Rivest invented RC4 in 1987, and it is a stream cipher. Because RC4 is a stream cipher, it encrypts data byte by byte.

Of all the stream ciphers, RC4 is the widely used stream cipher due to its speed of operations and simplicity.

While RC4 is known for its ease of use and speed in software, it has been found to have several weaknesses, making it insecure. When the beginning of the output keystream isn't destroyed, or when non-random or linked keys are utilized, it's highly vulnerable. The usage of RC4, in particular, has resulted in relatively insecure protocols such as WEP.

Working of RC4:

RC4 makes use of Key Scheduling Algorithm (KSA) and Pseudo-Random Generation Algorithm (PRGA) Algorithms.



Encryption Procedure:

- The user inputs a plain text file and a secret key.
- For the secret key entered, the encryption engine then generates the keystream by using KSA and PRGA Algorithm.
- This keystream is now XOR with the plain text, this XORing is done byte by byte to produce the encrypted text.
- The encrypted text is then sent to the intended receiver, the intended receiver will then decrypted the text and after decryption, the receiver will get the original plain text.

Example:

Plain Text: 10011001

Keystream: 11000011

Cipher Text: 01011010

Decryption Procedure:

Decryption is achieved by doing the same byte-wise X-OR operation on the Ciphertext.

Example:

Cipher Text: 01011010

Keystream: 11000011

Plain Text: 10011001

Advantages of RC4

1. RC4 is simple to use.
2. Speed of operation is fast as compared to other cipher suites.
3. RC4 cipher is easy to implement.
4. RC4 does not consume more memory.
5. For large streams of data, RC4 is the preferred choice.

Disadvantages of RC4

1. If a strong MAC is not used, RC4 is vulnerable to a bit-flipping attack.
2. RC4 does not support authentication.
3. RC4 is not feasible to be implemented on small streams of data.

Blowfish:

Blowfish is a symmetric, 64-bit block cipher with changeable length. As a "general-purpose algorithm," it was created by Bruce Schneier in 1993 as a quick, cost-free replacement for the venerable Data Encryption Standard (DES) and International Data Encryption Algorithm (IDEA) encryption techniques.

Blowfish is unpatented, substantially quicker than DES and IDEA, and freely accessible for all purposes. However, because of its short block size, which is seen as unsafe, it couldn't totally replace DES.

Key features:

1. **blockSize**: 64-bits
2. **keySize**: 32-bits to 448-bits variable size
3. **number of subkeys**: 18 [P-array]
4. **number of rounds**: 16
5. **number of substitution boxes**: 4 [each having 512 entries of 32-bits each]

How Blowfish work:

The Blowfish algorithm consists of two major parts:

How block ciphers like Blowfish work

1. **Data Encryption**

Blowfish encrypts data using a 16-round Feistel network, which is a structure used in many symmetric block ciphers. Each round consists of two main operations:

- **Key-dependent permutation**: The data is shuffled in a manner dependent on the key.
- **Key- and data-dependent substitution**: Specific portions of the data are substituted based on both the key and the data itself.

Large, key-dependent S-boxes (substitution boxes) are used during substitution, which are integral to the encryption process. The operations used in Blowfish encryption include XOR (a bitwise exclusive OR operation) and addition on 32-bit words.

2. **Key Expansion and Subkeys**

Before data encryption or decryption can occur, Blowfish expands the key into multiple subkeys that are used during the encryption rounds. Blowfish can use a key size of up to 448 bits. The key expansion process involves generating subkey arrays totaling 4,168 bytes, including a P-array and S-boxes.

- **P-array**: Contains 18 32-bit subkeys.
- **S-boxes**: Consist of four 32-bit S-boxes, each with 256 entries.

Subkey Generation Process

1. **Initialization**: The P-array and S-boxes are initialized with a fixed string derived from the hexadecimal digits of pi (π).
2. **Key Integration**: The key is XORed with the elements of the P-array. For example, the first 32 bits of the key are XORed with the first element of the P-array (P1), the next 32 bits with P2, and so on.
3. **Subkey Refinement**:
    - Encrypt an all-zero string using the partially initialized subkeys.
    - Replace P1 and P2 with the output from this encryption.
    - Encrypt the modified data again and replace P3 and P4, continuing this process until all the P-array and S-box entries are refined.

In total, Blowfish executes 521 iterations during the key expansion phase, processing around 4 KB of data to generate all necessary subkeys.

Advantages of Blowfish:

- Much faster and more efficient than DES and IDEA algorithms;
- Unpatented and can be freely used by anyone even without a license;
- Despite the complex initialization phase before encryption, the data encryption process is efficient on large microprocessors;
- Provides extensive security for software and applications developed in Java;
- Provides secure access for backup tools; and
- Supports secure user authentication for remote access.

Disadvantages of Blowfish:

- **Key Setup Time**: The key expansion process is slow, making it less ideal for applications requiring frequent key changes.
- **Block Size**: Uses a 64-bit block size, which is less secure against modern attacks compared to ciphers with 128-bit blocks.
- **Outdated**: Newer algorithms like AES offer better performance and security for most applications.

- Each new key requires preprocessing equivalent to 4 KB of text, which affects its speed, making it unusable for some applications.

These notes are prepared by **Suhail Abass Hurrah** (S.P College Srinagar, batch 2019)
For any queries, you can email me at Hurrahsuhail1@gmail.com