1. Confusion

- ❑ **Definition**: Confusion refers to making the relationship between the ciphertext and the encryption key as complex as possible. This ensures that even if an attacker understands how the ciphertext relates to the plaintext, they cannot easily infer the key.
- ❑ **Goal**: Confusion makes it difficult to reverse-engineer the key by obfuscating the connection between the input (plaintext/key) and output (ciphertext).

Example of Confusion (AES)

In the **AES encryption algorithm**, confusion is primarily introduced by the **SubBytes** step. Here’s how it works:

- ❑ **SubBytes** uses a non-linear substitution (S-box), where each byte of the state matrix is replaced with another byte from a precomputed table. This substitution is non-linear, meaning that the relationship between the input and output is not a simple one-to-one mapping.

2. Diffusion

- ❑ **Definition**: Diffusion refers to spreading the influence of each part of the plaintext over as much of the ciphertext as possible. In other words, changing a single bit of the plaintext should result in multiple bits of the ciphertext changing in a complex way.
- ❑ **Goal**: Diffusion ensures that small changes in the plaintext lead to significant changes in the ciphertext, thus making it hard for an attacker to find patterns in the encryption process.

Example of Diffusion (AES)

In **AES**, diffusion is primarily introduced by the **ShiftRows** and **MixColumns** steps. Here’s how diffusion works in these steps:

- ❑ **ShiftRows**: This step shifts the rows of the state matrix by a certain number of positions, ensuring that bytes from different columns get mixed together. This spreads out the effect of each byte across the state matrix.

- ❑ **MixColumns**: This step mixes each column of the state matrix using matrix multiplication. As a result, any change in one byte affects all the other bytes in the column. This spreads the influence of each byte throughout the state, enhancing **diffusion**.

AES (Advanced Encryption Standard) is a symmetric encryption algorithm widely used to secure data. It operates on fixed-size blocks of data (128-bit blocks) and supports key sizes of 128, 192, or 256 bits. The algorithm involves several rounds of transformations, where the number of rounds depends on the key length:

- 10 rounds for 128-bit keys
- 12 rounds for 192-bit keys
- 14 rounds for 256-bit keys

Each round consists of several steps: **SubBytes**, **ShiftRows**, **MixColumns** (except the last round), and **AddRoundKey**. Below is a detailed explanation of each step.

1. Key Expansion

Before the actual encryption starts, the input key is expanded into an array of round keys. Each round of AES requires its own 128-bit key, which is derived from the original key using a key scheduling algorithm. The key expansion depends on the key length, but in general, it involves XORing parts of the key and applying the Rijndael S-box (the same S-box used in the SubBytes step).

2. Initial Round (AddRoundKey)

AES starts by XORing the block of plaintext with the first round key, which is derived from the original key in the Key Expansion step.

- **AddRoundKey**: The plaintext is combined with the round key using a bitwise XOR operation. This step ensures that the initial plaintext is affected by the secret key from the beginning.

3. SubBytes

This is a non-linear substitution step where each byte in the state is replaced by its corresponding value from a precomputed substitution box, called the Rijndael S-box.

- **SubBytes**: Each byte in the 4x4 matrix of the data block is replaced using a fixed S-box. The S-box provides a non-linear transformation that improves security by making it more difficult for an attacker to predict changes in the output based on changes in the input.

4. ShiftRows

This step is a simple permutation that shifts the rows of the 4x4 matrix.

- **ShiftRows**: The rows of the state are shifted cyclically to the left:
  - Row 0 remains unchanged.
  - Row 1 is shifted left by one byte.
  - Row 2 is shifted left by two bytes.
  - Row 3 is shifted left by three bytes.

This step creates diffusion by ensuring that bytes from different columns are mixed together.

5. MixColumns

In this step, the columns of the 4x4 matrix are treated as polynomials and are mixed using matrix multiplication over a Galois Field.

- **MixColumns**: Each column of the state is multiplied by a fixed matrix. This step ensures that each byte in a column is affected by all the other bytes in the same column, which spreads the influence of each byte across the whole column, enhancing diffusion.

6. AddRoundKey

This step is the same as the initial round, where the current state is XORed with a round key derived from the Key Expansion step.

- **AddRoundKey**: The state matrix is XORed with the round key. This operation integrates the key into the current state, making it harder to reverse the transformations without knowing the key.

7. Final Round (No MixColumns)

The final round of AES is slightly different from the other rounds. It consists of SubBytes, ShiftRows, and AddRoundKey, but **MixColumns** is omitted in this last round to maintain the integrity of the cipher structure.

Decryption

The decryption process in AES is the reverse of encryption. It uses inverse transformations:

- **InvSubBytes** (inverse of SubBytes)
- **InvShiftRows** (inverse of ShiftRows)
- **InvMixColumns** (inverse of MixColumns, except in the final round)
- **AddRoundKey** remains the same since XOR is its own inverse.