AES – Advanced Encryption Standard:

The Advanced Encryption Standard (AES) is a widely adopted symmetric block‐cipher algorithm. It operates on fixed 128-bit blocks of data and supports three key lengths—128, 192, or 256 bits—which correspond to 10, 12, or 14 rounds of processing, respectively. Each round applies a series of transformations—SubBytes (a non-linear byte substitution using an S-box), ShiftRows (a transposition step that cyclically shifts each row of the state), MixColumns (a linear mixing operation over each column), and AddRoundKey (an XOR with a portion of the expanded key)—to thoroughly diffuse and obscure the plaintext. The advantages of AES is a combination of strong security, high performance in both hardware and software, and straightforward, regular structure, making it the backbone of modern secure communications, disk encryption, and countless cryptographic protocols.

GF – Galois Field:

Each byte in the encryption process is treated as an element in the finite field GF . The byte is observed in a polynomial representation, such as:

, bᵢ ∈ {0,1}

Over this field 2 operation are relevant to the encryption process, addition and multiplication. Addition is a bitwise XOR. Multiplication is multiplying 2 bytes then using the modulo operation using the AES irreducible polynomial:  
 = 0x1B

Practically, Multiplication is implemented in hardware using a “Multiply by 2” base block, which involves hardware cheap operation: shift and XOR. AES uses this finite field to gain diffusion and non-linearity.

Round Key Scheduling:

For each round transformation, we use a different key each based on the previous round key. Take the key and divide it to 4 words. Such that is the MSB.

The first word, , is shifted one byte to the left, then goes through the SubBytes Block. You take the product of that and using XOR operation you can calculate the first word of the next key:

Rcon is a local LUT that contains 10 cells: {0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80,0x1B,0x36})

Each subsequent word is generated by XOR-ing the corresponding previous word with the immediately preceding newly generated word

Sub Bytes:

The Sub Bytes operation involves taking each byte of an input, and substituting it for another byte using a pre fetched 256 by 8 LUT. This process is also known as an S-Box substituting. For the decryption process you use the same logic only with an inversed table.

Shift Rows:

The Shift Rows operation takes the 128-bit block and creates a 4x4 matrix(column major), where each cell is a byte from the block.

Each row is shifted left by the row number. Row 0 stays the same, Row 1 is shifted by 1 byte, Row 2 is shifted by 2 and Row 3 is shifted by 3.

For the decryption process you shift right and not left.

Mix Columns:

The Shift Rows operation creates the same matrix as Shift Rows. Each column is multiplied from the left with a pre determined matrix ()

For the decryption process uses the same logic only with a different matrix ().

Encryption Process:

The first round(round 0) is calculated only using a XOR operation between the original key and block.

Rounds 1-9 take the newly computed block and takes it through each stage, Sub Bytes , Shift Rows and Mix Columns . finally using XOR with the current round scheduled key to generate the next block.

Round 10 performs the same operation as rounds 1-9 only without the mix columns.

At the end you get the ciphered block.

Decryption Process:

The Decryption process involves basically inversing the encryption process. Using the inverse mode for each stage, you first calculate each of the round keys and placing them in a table. For the first round you XOR the final round key with the cipher text and perform the inversed shift rows and inversed Sub Bytes. All the next round perform the inversed transformation using inversed Mix Columns, inversed Shift Rows, Inversed Sub Bytes and XOR-ing with the relevant round key.