**CBC-MAC, HMAC-PMAC and CW-MAC CBC-CBC (Cipher Block Chaining)**

* **Mode of operation for symmetric encryption algorithms, such as the Advanced Encryption Standard (AES).**
* **CBC is designed to provide confidentiality and protect the privacy of data by introducing an element of feedback and interdependence between the blocks of plaintext and cipher-text.**
* **Block Encryption: CBC operates on fixed-size blocks of data, typically 128 bits (16 bytes) for AES. The plaintext message is divided into blocks, and each block is encrypted independently using the chosen encryption algorithm (e.g., AES) with a secret encryption key.**
* **Initialization Vector (IV): CBC requires the use of an**•**Initialization Vector, which is a random value of the same block size as the encryption algorithm (e.g., 128 bits for AES). The IV is XOR with the first plaintext block before encryption.**
* **Chaining: For subsequent blocks, prior to encryption, CBC XORs**• **the plaintext block with the previous cipher-text block. This introduces a feedback mechanism where each cipher-text block depends on all previous plaintext blocks. The result of the XOR operation is then encrypted using the encryption algorithm.**
* **Encryption Process: The encryption process continues for all blocks of the plaintext, each depending on the previous cipher-text block through the XOR operation. The resulting cipher-text blocks are generated.**
* **Block Decryption: Each cipher-text block is decrypted**• **independently using the decryption algorithm (e.g., AES) and the same secret encryption key. Chaining: Similar to encryption, for each decrypted cipher-text**• **block (except the first one), CBC XORs it with the previous cipher-text block, which effectively reverses the XOR operation performed during encryption.**
* **Initialization Vector (IV): In CBC decryption, the IV is XOR**• **with the first decrypted cipher-text block to recover the first block of the plaintext.**
* **Decryption Process: The decryption process continues for all**• **cipher-text blocks, each block being decrypted and XOR with the previous cipher-text block to recover the original plaintext blocks. CBC mode provides several security benefits:**
* **Confidentiality: By XOR each plaintext block with the previous cipher-text block, CBC introduces a level of diffusion, making it difficult for an attacker to extract meaningful information from the cipher-text**
* **Error Propagation: Any errors or alterations in the cipher-text will propagate to subsequent blocks, making it easier to detect tampering. Random Initialization Vector (IV): A unique and random IV for each encryption operation ensures that even if the same plaintext is encrypted multiple times, the resulting cipher-text will be different.**

**MAC Message Authentication Code (MAC)**

* **MAC algorithm is a symmetric key cryptographic technique to provide message authentication. For establishing MAC process, the sender and receiver share a symmetric key K. Essentially a MAC is an encrypted checksum generated on the underlying message that is sent along with a message to ensure message authentication.**
* **Let us now try to understand the entire process:-**
* **The sender uses some publicly known MAC algorithm, inputs the message and the secret key K and produces a MAC value.**
* **Similar to hash, MAC function also compresses an arbitrary long input into a fixed length output. The major difference between hash and MAC is that MAC uses secret key during the compression.**
* **The sender forwards the message along with the MAC. Here, we assume that the message is sent in the clear, as we are concerned of providing message origin authentication, not confidentiality.**
* **If confidentiality is required then the message needs encryption. On receipt of the message and the MAC, the receiver feeds the received message and the shared secret key K into the MAC algorithm and re-computes the MAC value.**
* **The receiver now checks equality of freshly computed MAC with the MAC received from the sender. If they match, then the receiver accepts the message and assures himself that the message has been sent by the intended sender. If the computed MAC does not match the MAC sent by the sender, the receiver cannot determine whether it is the message that has been altered or it is the origin that has been falsified. As a bottom-line, a receiver safely assumes that the message is not the genuine.**
* **CBC MAC,** **short for Cipher Block Chaining Message Authentication Code, is a method used to provide message integrity and authenticity in cryptography. It is a construction that combines a symmetric encryption algorithm with a message authentication code (MAC). The CBC-MAC algorithm operates on fixed-length blocks of data, typically using a block cipher such as AES (Advanced Encryption Standard). It takes two inputs: a secret key and the data to be authenticated. The data is divided into blocks, and each block is XOR with the previous cipher-text block before encryption. Here's a step-by-step explanation of the CBC-MAC algorithm: Dividing the Data:- The data to be authenticated is divided into fixed-length blocks. Padding may be applied if the data length is not a multiple of the block size. Initialization: A random initialization vector (IV) is generated. Encryption: The first data block is XOR with the IV. The resulting value is then encrypted using the block cipher algorithm with the secret key. The cipher-text is stored. Chaining:- For each subsequent block, the previous cipher-text block is XOR with the current data block. The result is then encrypted using the block cipher with the secret key, and the cipher-text is stored again. Finalization: The last cipher-text block obtained from the chaining process is considered the MAC value. It represents the integrity and authenticity of the entire message. To verify the authenticity of a message using CBC-MAC, the receiver applies the same algorithm to the received message and compares the resulting MAC with the one transmitted alongside the message. If they match, the message is considered valid and has not been tampered with.**