Chapter 12
Message Authentication Codes

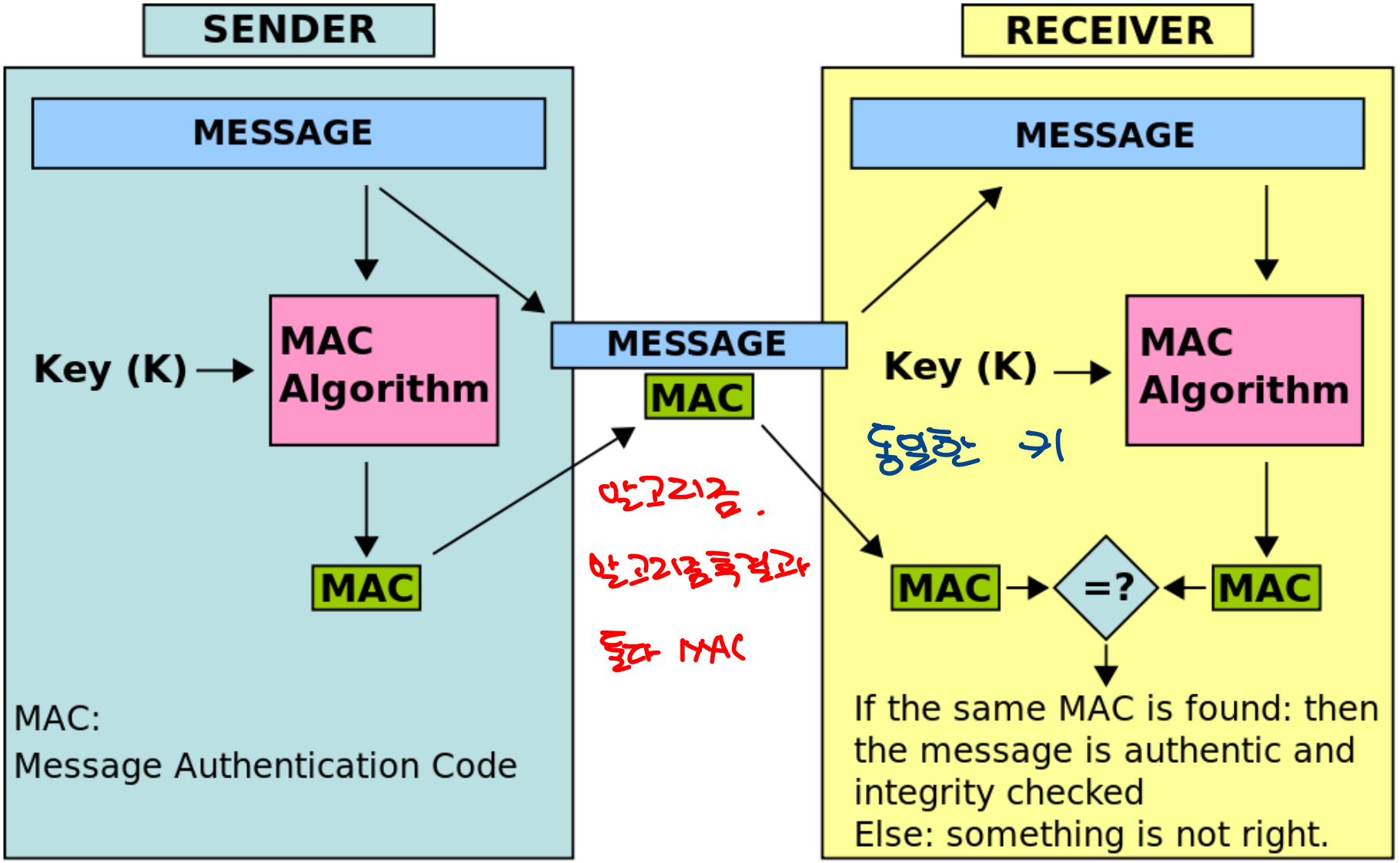
정보보안

Cryptographic Primitives

	Symmetric cryptography (Private-key cryptography)	**Asymmetric cryptography (Public-key cryptography)**
Confidentiality	Private-key encryption (e.g., AES)	Public-key encryption (e.g., RSA-OAEP)
Integrity	Message authentication code (MAC)	Digital signature (e.g., RSA-PSS)

Message Authentication Code

- MAC
  - In cryptography, a message authentication code (MAC) is a short piece of information used to authenticate a message and to provide integrity and authenticity assurances on the message.
  - Integrity assurances detect accidental and intentional message changes, while authenticity assurances affirm the message's origin.
- MAC functionality
  - A MAC algorithm, sometimes called a keyed (cryptographic) hash function (however, cryptographic hash function is only one of the possible ways to generate MACs), accepts as input a secret key and an arbitrary-length message to be authenticated, and outputs a MAC (sometimes known as a tag).
  - The MAC value protects both a message's data integrity as well as its authenticity, by allowing verifiers (who also possess the secret key) to detect any changes to the message content.

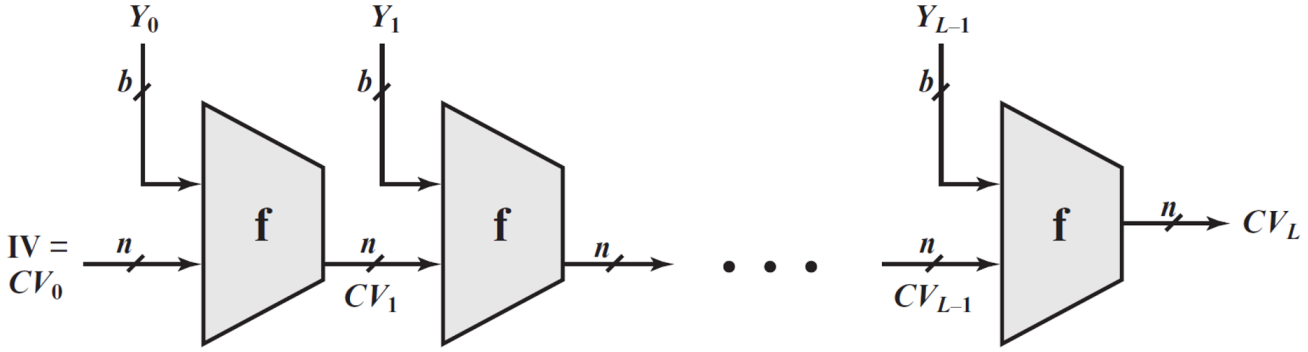

SENDER
RECEIVER
MESSAGE
MESSAGE
Key (K)
MAC Algorithm
MAC
MESSAGE
MAC
Key (K)
MAC Algorithm
알고리즘.
알고리즘결과
또다 MAC
동일한 키
MAC
=?
MAC
MAC:
Message Authentication Code
If the same MAC is found: then the message is authentic and integrity checked
Else: something is not right.

Message Authentication Code

- Algorithms
  - Key generation algorithm
  - Tag-generation algorithm: T = Mac(K, M)
  - Verification algorithm: Vrfy(K, M, T) = 1/0
- Security
  - To be considered secure, a MAC function must resist existential forgery under chosen-plaintext attacks. This means that even if an attacker has access to an oracle which possesses the secret key and generates MACs for messages of the attacker's choosing, the attacker cannot guess the MAC for other messages (which were not used to query the oracle) without performing infeasible amounts of computation.
- Implementation
  - MAC algorithms can be constructed from other cryptographic primitives, such as cryptographic hash functions (as in the case of HMAC) or from block cipher algorithms (as in the case of CMAC).

Length Extension Attack

- In cryptography and computer security, a length extension attack is a type of attack where an attacker can use Hash($message_1$) and the length of $message_1$ to calculate Hash($message_1 \parallel message_2$) for an attacker-controlled $message_2$, without needing to know the content of $message_1$.
- Algorithms like MD5, SHA-1, and SHA-2 that are based on the Merkle–Damgård construction are susceptible to this kind of attack.



- The SHA-3 algorithm is not susceptible.

HMAC

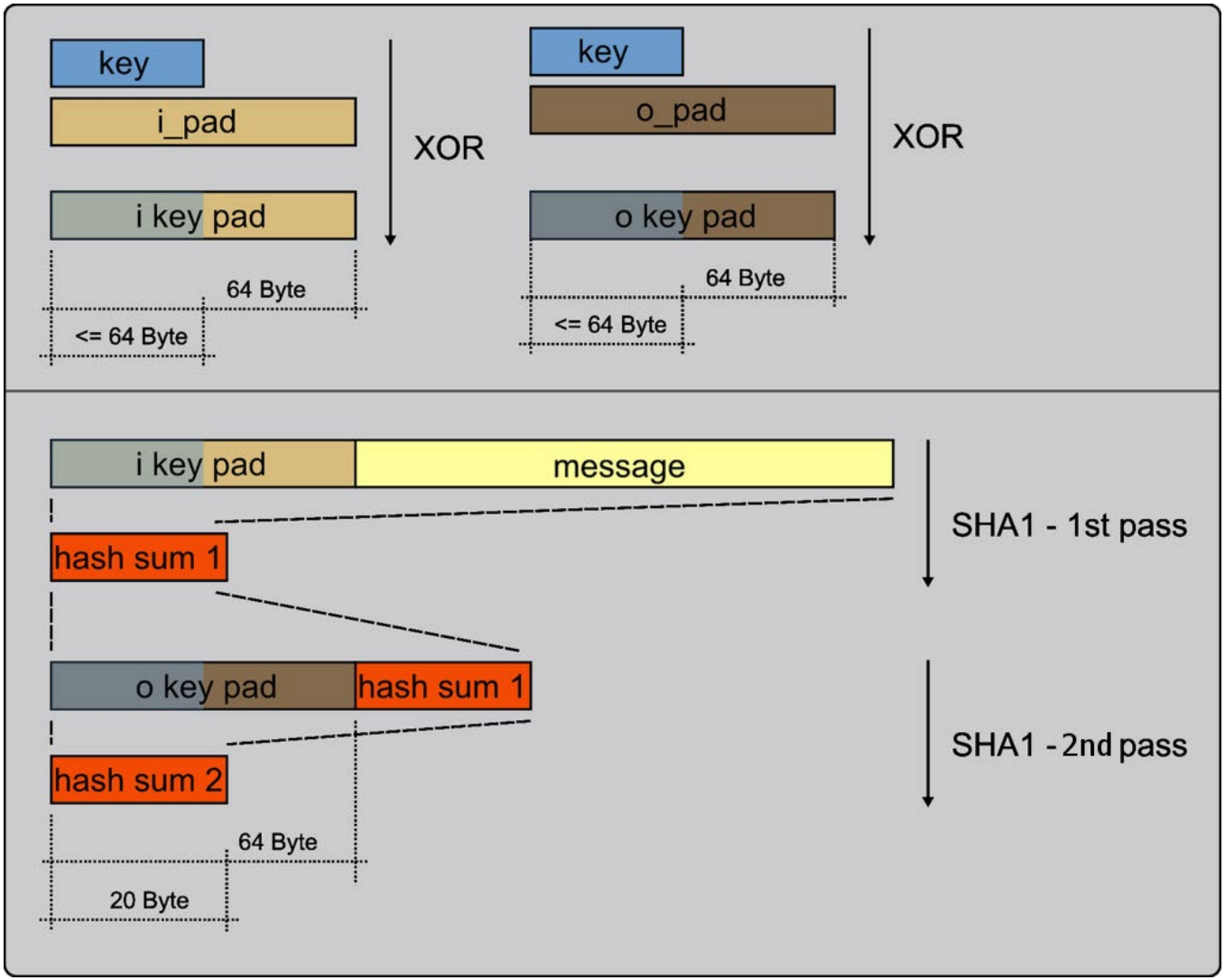
- HMAC (keyed-hash message authentication code or hash-based message authentication code)
  - Any cryptographic hash function, such as SHA-2 or SHA-3, may be used in the calculation of an HMAC; the resulting MAC algorithm is termed HMAC-X, where X is the hash function used (e.g. HMAC-SHA256 or HMAC-SHA3-256).
  - HMAC uses two passes of hash computation. The secret key is first used to derive two keys – inner and outer. The first pass of the algorithm produces an internal hash derived from the message and the inner key. The second pass produces the final HMAC code derived from the inner hash result and the outer key. Thus the algorithm provides better immunity against length extension attacks.

HMAC

- Definition (RFC 2104)
  - HMAC(K, M) = H ((K′ ⊕ opad) || H ((K′ ⊕ ipad) || M))
    - H is a cryptographic hash function.
    - $K' = \begin{cases} K & : \text{padded with extra zeros} \\ H(K) & : \text{if longer than block size} \end{cases}$

      K′ is a secret key padded to the right with extra zeros to the input block size of the hash function, or the hash of the original key if it's longer than block size.
    - M is the message to be authenticated.
    - || denotes concatenation.
    - ⊕ denotes exclusive or (XOR).
    - opad is the outer padding (0x5c5c5c…5c5c, one-block-long hexadecimal constant).
    - ipad is the inner padding (0x363636…3636, one-block-long hexadecimal constant).
    - Note: For example, the block size is 64 bytes (512 bits) when using one of the following hash functions: MD5, SHA-1, SHA-224, SHA-256

HMAC-SHA1

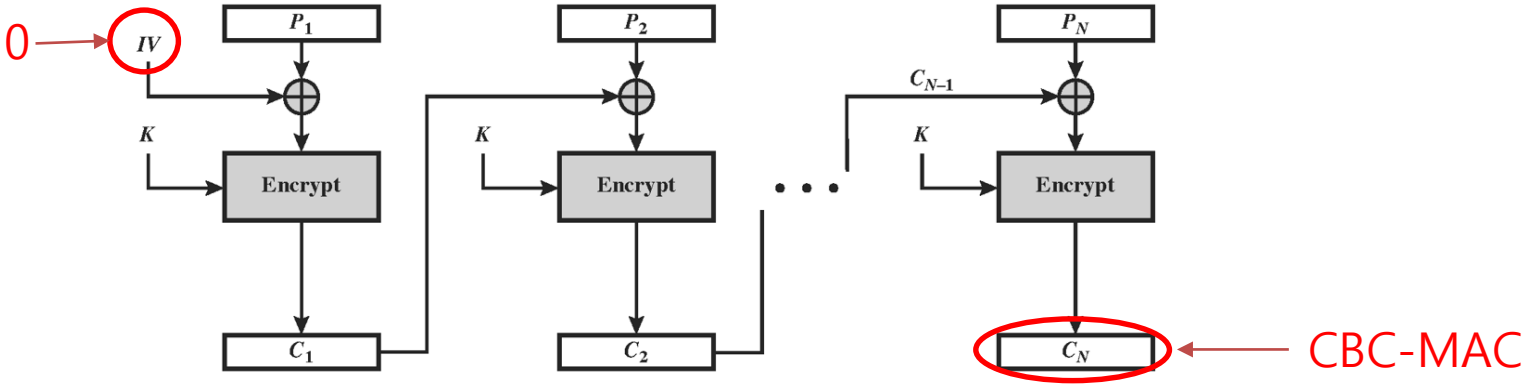

key
i_pad
XOR
i key pad
64 Byte
<= 64 Byte
key
o_pad
XOR
o key pad
64 Byte
<= 64 Byte
i key pad
message
SHA1 - 1st pass
hash sum 1
o key pad
hash sum 1
SHA1 - 2nd pass
hash sum 2
64 Byte
20 Byte

SHA-3 vs. HMAC

- The Keccak hash function, that was selected by NIST as the SHA-3 competition winner, doesn't need this nested approach and can be used to generate a MAC by simply prepending the key to the message.

CBC-MAC

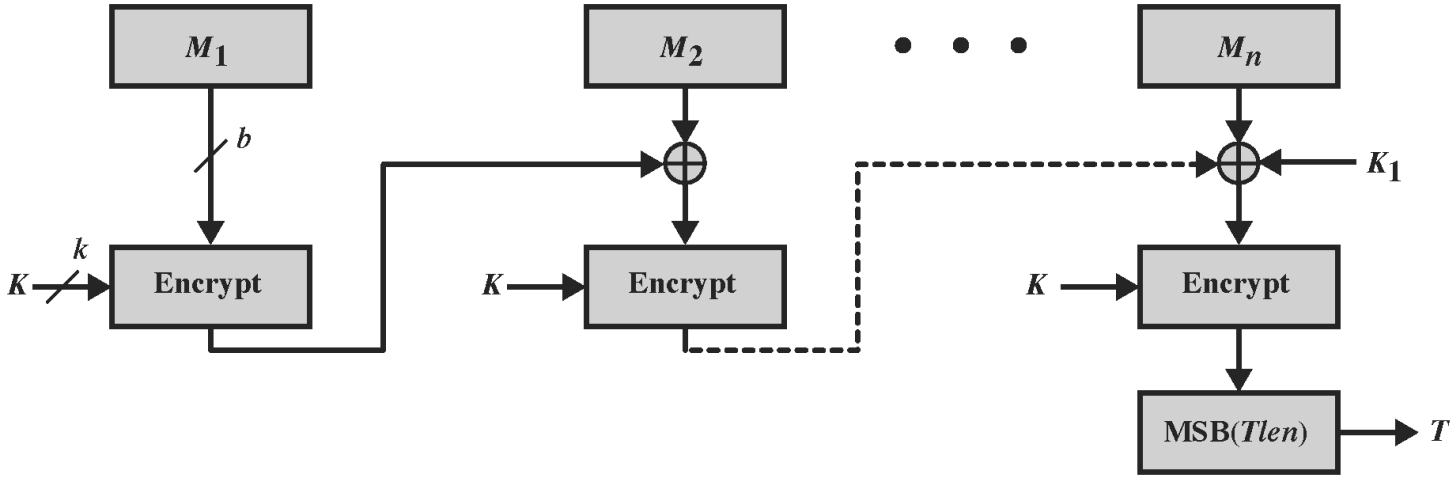
- CBC-MAC (Cipher Block Chaining MAC)
  - To calculate the CBC-MAC of message M, one encrypts M in CBC mode with zero IV.



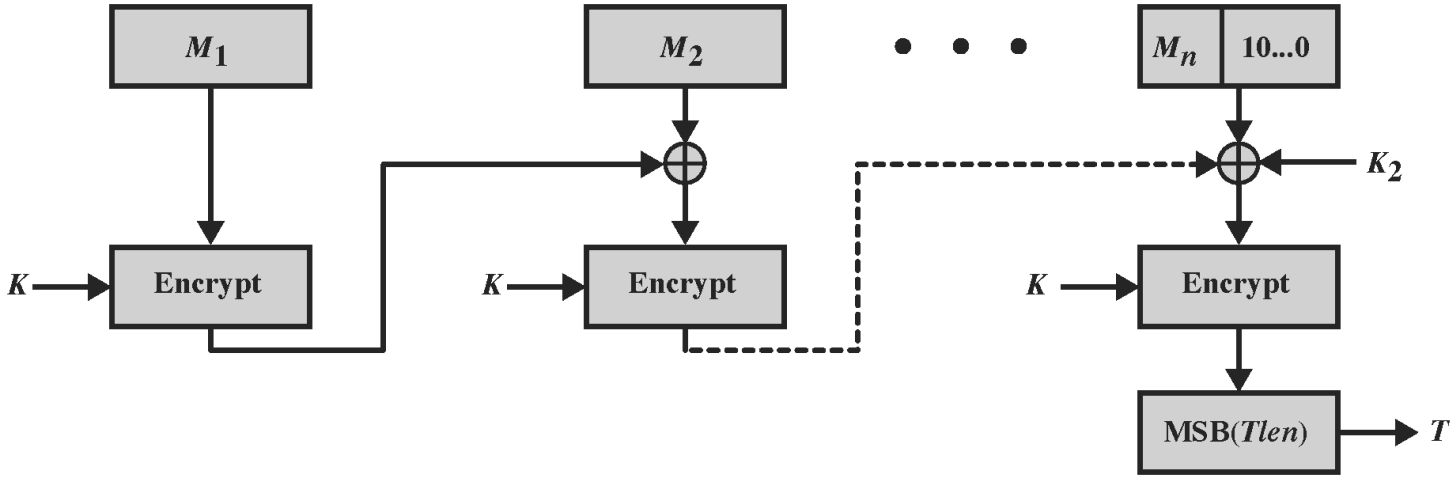
  - If the block cipher used is secure (meaning that it is a pseudorandom permutation), then CBC-MAC is secure for fixed-length messages. However, it is not secure for variable-length messages.

CMAC

- CMAC (Cipher-based MAC)
  - The core of the CMAC algorithm is a variation of CBC-MAC that Black and Rogaway proposed and analyzed under the name XCBC and submitted to NIST.
  - This mode of operation fixes security deficiencies of CBC-MAC (CBC-MAC is secure only for fixed-length messages).



(a) Message length is integer multiple of block size



(b) Message length is not integer multiple of block size

Figure 12.8 Cipher-Based Message Authentication Code (CMAC)

CMAC

$$
\begin{aligned}
C_1 &= \mathrm{E}(K, M_1) \\
C_2 &= \mathrm{E}(K, [M_2 \oplus C_1]) \\
C_3 &= \mathrm{E}(K, [M_3 \oplus C_2]) \\
&\bullet \\
&\bullet \\
&\bullet \\
C_n &= \mathrm{E}(K, [M_n \oplus C_{n-1} \oplus K_1]) \\
T &= \mathrm{MSB}_{Tlen}(C_n)
\end{aligned}
$$

where

T = message authetication code, also referred to as the tag
$Tlen$ = bit length of T
$\mathrm{MSB}_s(X)$ = the s leftmost bits of the bit string X

If the message is not an integer multiple of the cipher block length, then the final block is padded to the right (least significant bits) with a 1 and as many 0s as necessary so that the final block is also of length b. The CMAC operation then proceeds as before, except that a different n-bit key K_2 is used instead of K_1.

CMAC

The two n-bit keys are derived from the k-bit encryption key as follows.

$$
\begin{aligned}
L &= \mathrm{E}(K, 0^n) \\
K_1 &= L \bullet x \\
K_2 &= L \bullet x^2 = (L \bullet x) \bullet x
\end{aligned}
$$

where multiplication ($\bullet$) is done in the finite field GF(2^n) and x and x^2 are first- and second-order polynomials that are elements of GF(2^n). Thus, the binary representation of x consists of $n - 2$ zeros followed by 10; the binary representation of x^2 consists of $n - 3$ zeros followed by 100. The finite field is defined with respect to an irreducible polynomial that is lexicographically first among all such polynomials with the minimum possible number of nonzero terms. For the two approved block sizes, the polynomials are $x^{64} + x^4 + x^3 + x + 1$ and $x^{128} + x^7 + x^2 + x + 1$.

Authenticated Encryption

- Authenticated encryption
  - Encryption systems that simultaneously protect confidentiality and authenticity (integrity) of communications.

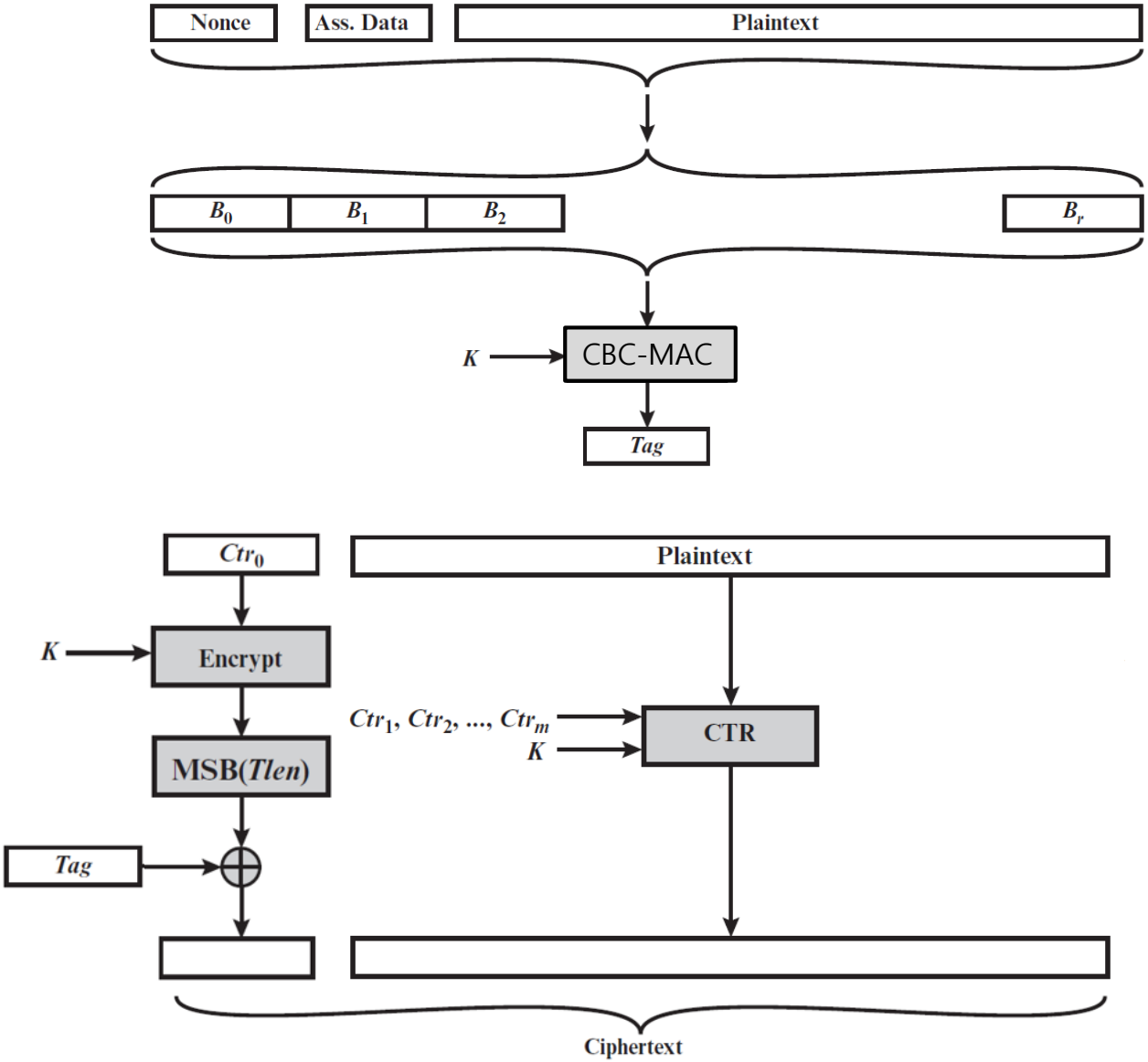
- Mode of operation
  - CCM and GCM

CCM/GCM

- CCM (Counter with CBC-MAC)
  - CCM mode combines CBC-MAC with the counter mode of encryption.
  - CBC-MAC is first computed on the message to obtain a tag and then the message and the tag are encrypted using counter mode.
  - CCM mode is used in the IEEE 802.11i (as CCMP, an encryption algorithm for WPA2), IPsec, Bluetooth Low Energy, and TLS.

- GCM (Galois/Counter Mode)
  - GCM can take full advantage of parallel processing.
  - GCM combines the counter mode of encryption with the new Galois mode of authentication.
  - GCM mode is used in the IEEE 802.1AE (MACsec) Ethernet security, IEEE 802.11ad (also dubbed WiGig), ANSI (INCITS) Fibre Channel Security Protocols (FC-SP), IEEE P1619.1 tape storage, IETF IPsec standards, SSH, and TLS 1.2.

CCM


Nonce
Ass. Data
Plaintext
B_0
B_1
B_2
B_r
K
CBC-MAC
Tag
Ctr_0
Plaintext
K
Encrypt
$Ctr_1, Ctr_2, ..., Ctr_m$
K
CTR
MSB(Tlen)
Tag
Ciphertext