M2L7

Hashing Techniques: HMAC & CMAC

Message Authentication Codes

Each of the messages, like each one he had ever read of Stern's commands, began with a number and ended with a number or row of numbers. No efforts on the part of Mungo or any of his experts had been able to break Stern's code, nor was there any clue as to what the preliminary number and those ultimate numbers signified.

***—Talking to Strange Men,* Ruth Rendell**

General Security Requirements
(resistance against the following attacks)

- disclosure
- traffic analysis
- masquerade
- content modification
- sequence modification
- timing modification
- source repudiation
- destination repudiation

Message Authentication Functions

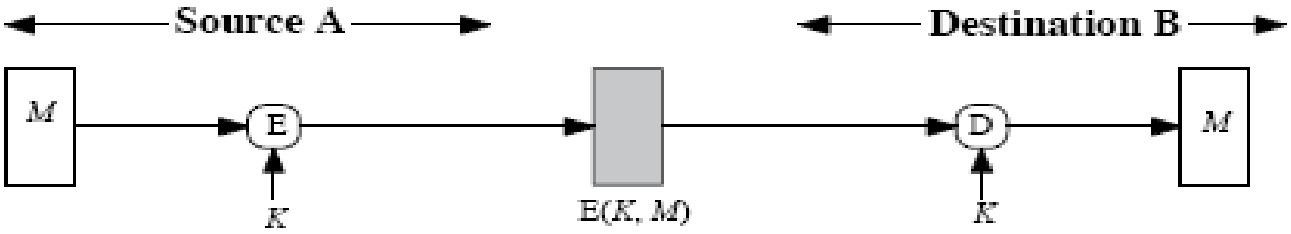
- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)
- will consider the security requirements
- then three alternative functions used:
  - hash function
  - message encryption
  - message authentication code (MAC)

Message Encryption

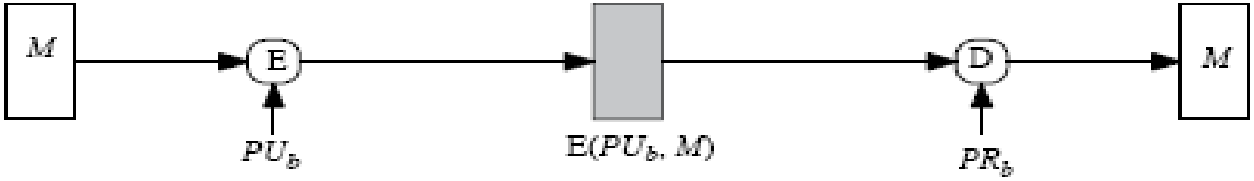
- message encryption by itself also provides a measure of authentication
- if symmetric encryption is used then:
  - receiver know sender must have created it
  - since only sender and receiver know key used
  - content can not be altered
  - message has suitable structure, redundancy or a checksum to detect any changes

Message Encryption

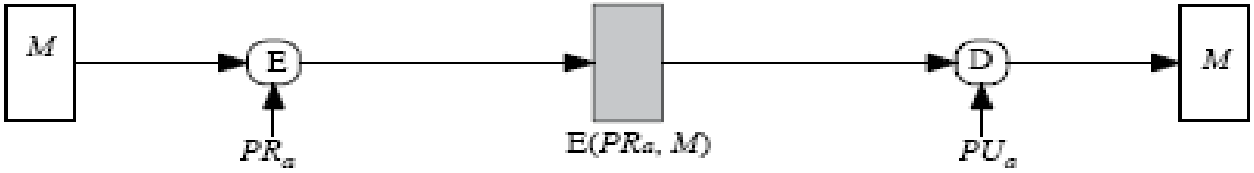
- if public-key encryption is used:
  - encryption provides no confidence of sender
  - since anyone potentially knows public-key
  - however if
    - sender **signs** message using their private-key
    - then encrypts with recipients public key
    - have both secrecy and authentication
  - again need to recognize corrupted messages
  - but at cost of two public-key uses on message



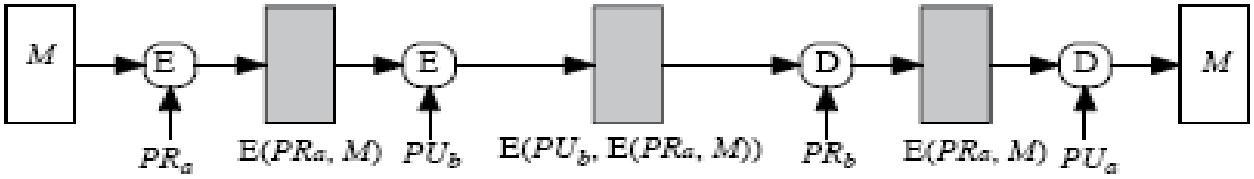
(a) Symmetric encryption: confidentiality and authentication



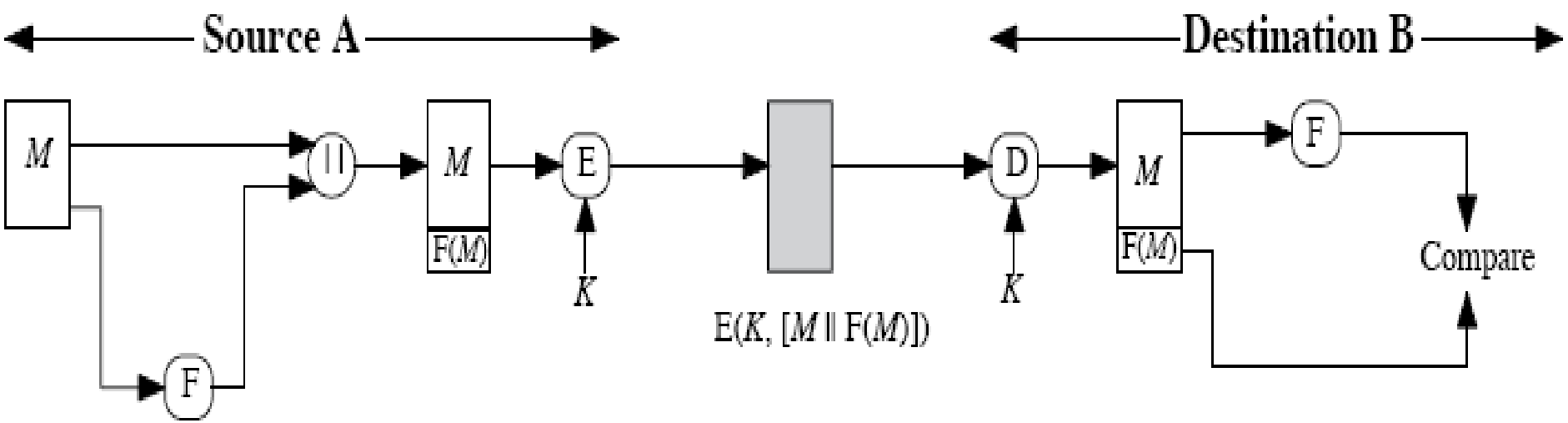
(b) Public-key encryption: confidentiality



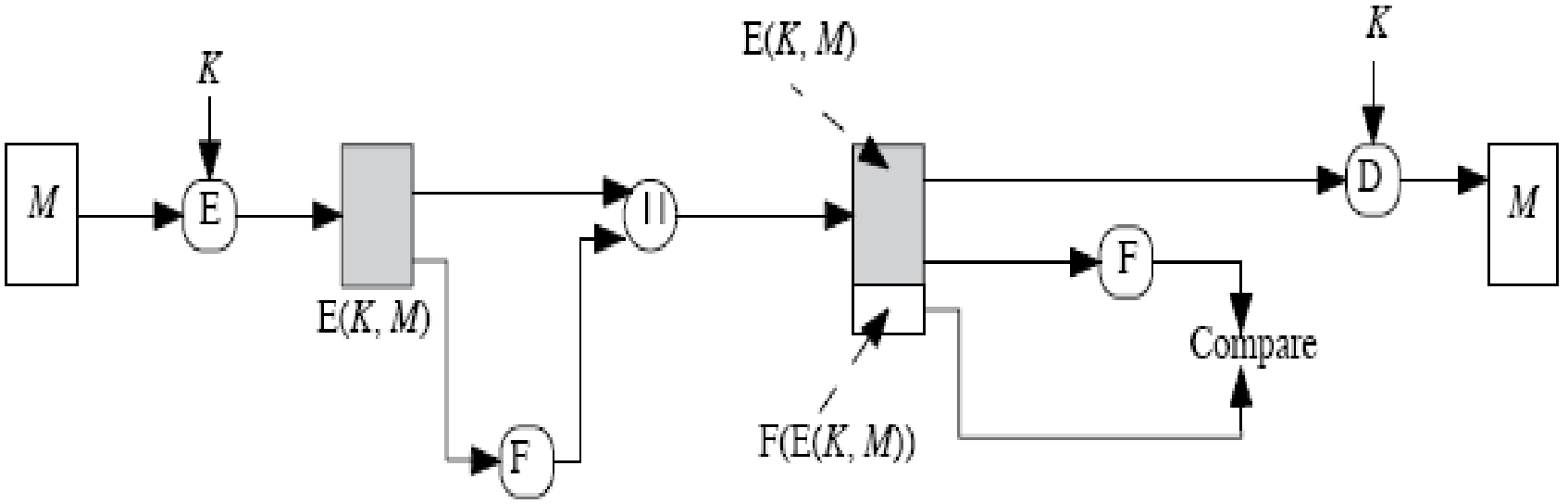
(c) Public-key encryption: authentication and signature



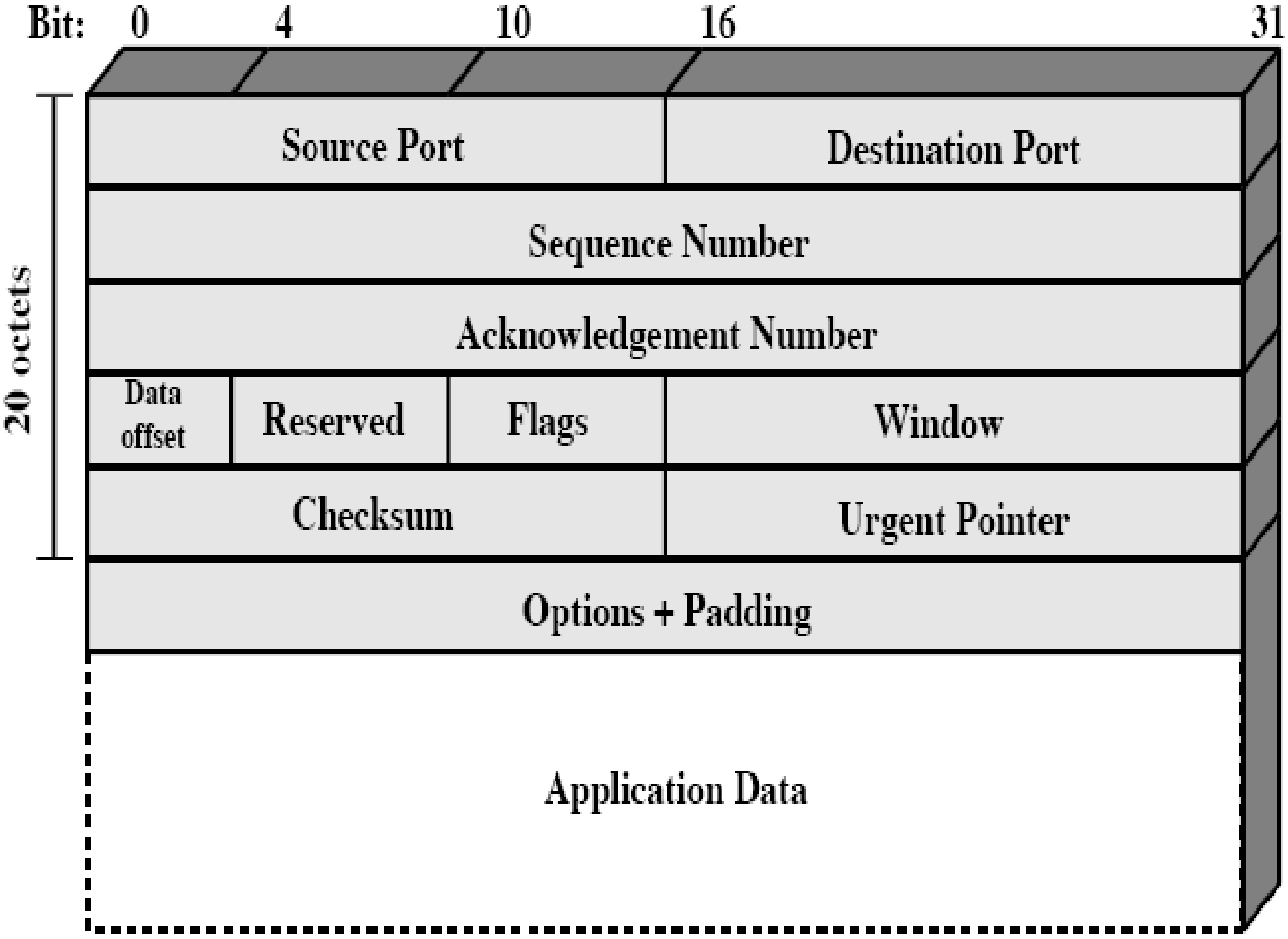
(d) Public-key encryption: confidentiality, authentication, and signature



(a) Internal error control



(b) External error control


Bit:
0
4
10
16
31
20 octets
Source Port
Destination Port
Sequence Number
Acknowledgement Number
Data offset
Reserved
Flags
Window
Checksum
Urgent Pointer
Options + Padding
Application Data

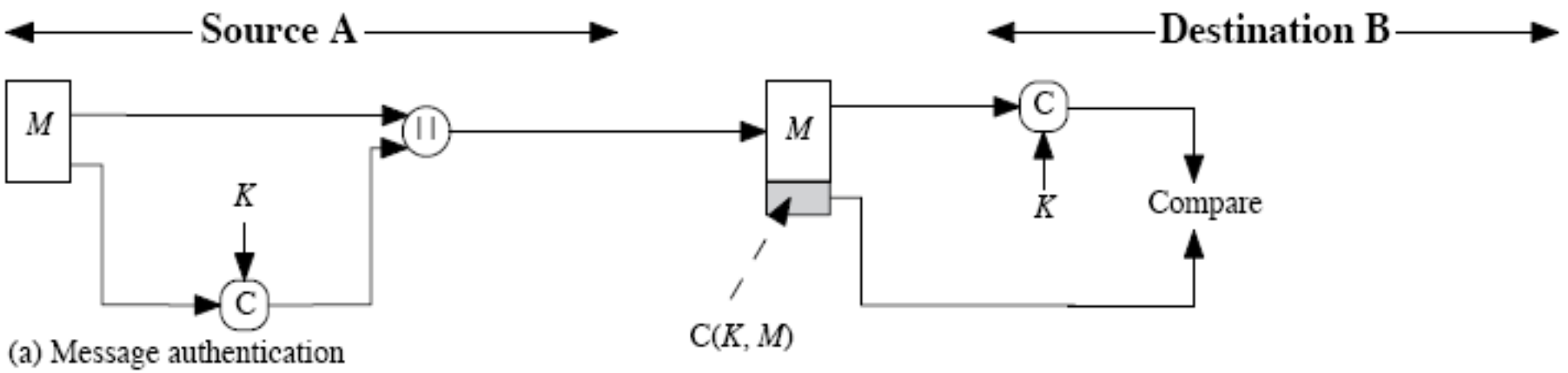
Message Authentication Code (MAC)

- generated by an algorithm that creates a small fixed-sized block
  - depending on both message and some key
  - like encryption though need not be reversible
- appended to message as a **signature**
- receiver performs same computation on message and checks it matches the MAC
- provides assurance that message is unaltered and comes from sender

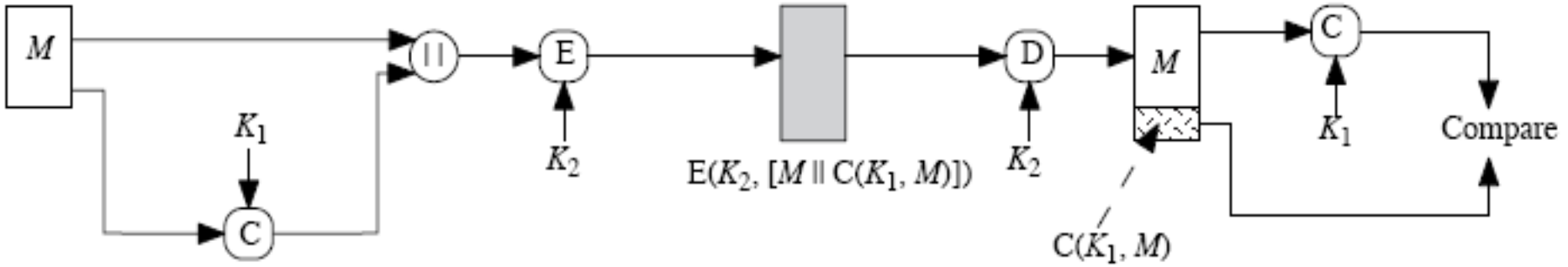
Message Authentication Code

M – input message
C – MAC function
K – shared secret key
MAC – message authentication code

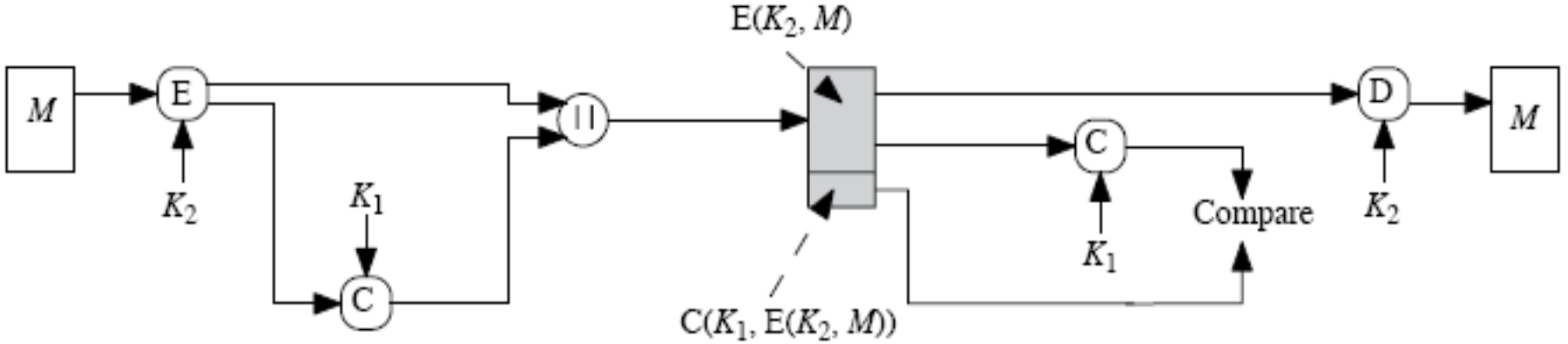
$$\mathbf{MAC} = \mathbf{C}_{\mathbf{K}}(\mathbf{M})$$



(a) Message authentication



(b) Message authentication and confidentiality; authentication tied to plaintext



(c) Message authentication and confidentiality; authentication tied to ciphertext

Message Authentication Codes

- as shown the MAC provides authentication
- can also use encryption for secrecy
  - generally use separate keys for each
  - can compute MAC either before or after encryption
  - is generally regarded as better done before
- why use a MAC?
  - sometimes only authentication is needed
  - sometimes need authentication to persist longer than the encryption (eg. archival use)
- note that a MAC is not a digital signature

MAC Properties

- a MAC is a cryptographic checksum

  $MAC = C_K(M)$

  - condenses a variable-length message M
  - using a secret key K
  - to a fixed-sized authenticator
- is a many-to-one function
  - potentially many messages have same MAC
  - but finding these needs to be very difficult

Requirements for MACs

- taking into account the types of attacks
- need the MAC to satisfy the following:
  1. knowing a message and MAC, is infeasible to find another message with same MAC
  2. MACs should be uniformly distributed
  3. MAC should depend equally on all bits of the message

Hash and MAC Algorithms

- Hash Functions
  - condense arbitrary size message to fixed size
  - by processing message in blocks
  - through some compression function
  - either custom or block cipher based
- Message Authentication Code (MAC)
  - fixed sized authenticator for some message
  - to provide authentication for message
  - by using block cipher mode or hash function

Keyed Hash Functions as MACs

- want a MAC based on a hash function
  - because hash functions are generally faster
  - code for crypto hash functions widely available
- hash includes a key along with message
- original proposal:

  `KeyedHash = Hash(Key|Message)`
  - some weaknesses were found with this
- eventually led to development of HMAC

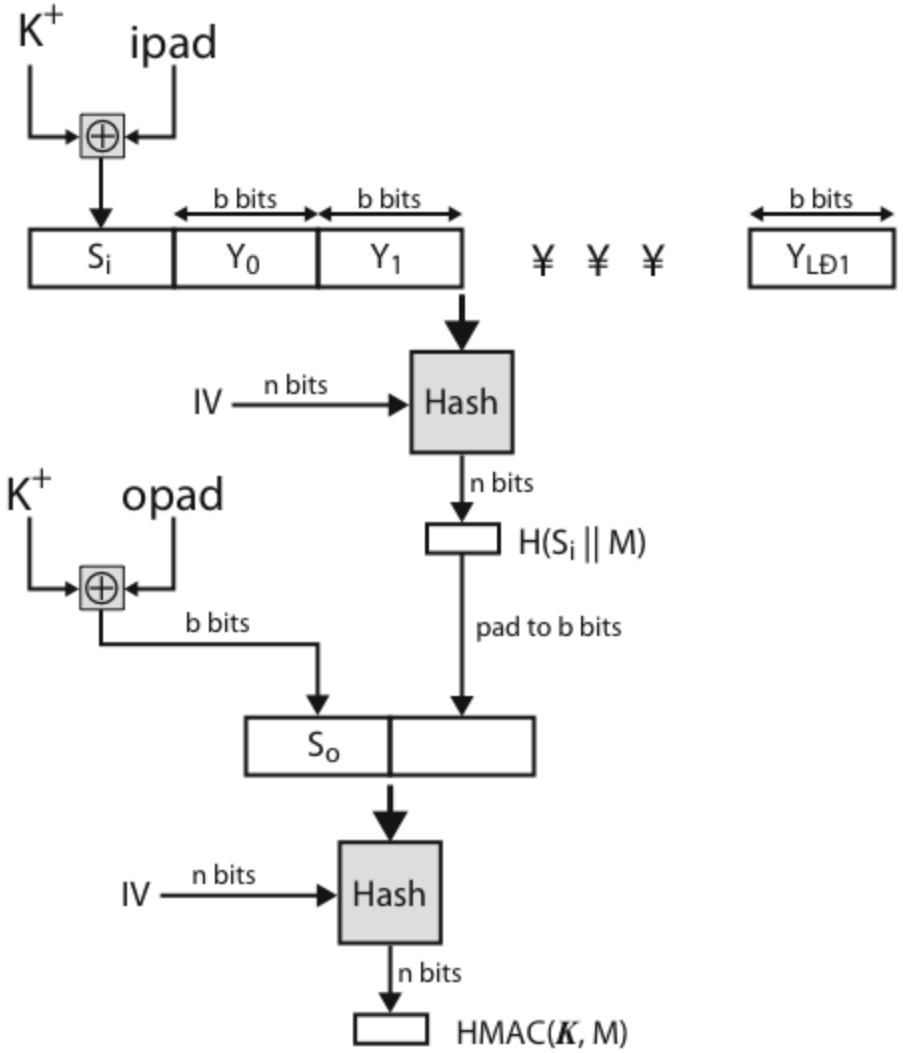
HMAC

- specified as Internet standard RFC2104
- uses hash function on the message:

$$HMAC_K = Hash[(K^+ \text{ XOR opad}) \,||\, Hash[(K^+ \text{ XOR ipad})||M)]]$$

- where K^+ is the key padded out to size
- and opad, ipad are specified padding constants
- overhead is just 3 more hash calculations than the message needs alone
- any hash function can be used
  - eg. MD5, SHA-1, RIPEMD-160, Whirlpool

HMAC Overview



HMAC Security

- proved security of HMAC relates to that of the underlying hash algorithm
- attacking HMAC requires either:
  - brute force attack on key used
  - birthday attack (but since keyed would need to observe a very large number of messages)
- choose hash function used based on speed verses security constraints
- Even broken hash functions (like MD5) can be used in HMAC.

Using Symmetric Ciphers for MACs

- can use any block cipher chaining mode and use final block as a MAC
- **Data Authentication Algorithm (DAA)** is a widely used MAC based on DES-CBC
  - using IV=0 and zero-pad of final block
  - encrypt message using DES in CBC mode
  - and send just the final block as the MAC
    - or the leftmost M bits ($16 \leq M \leq 64$) of final block
- but final MAC is now too small for security

Data Authentication Algorithm

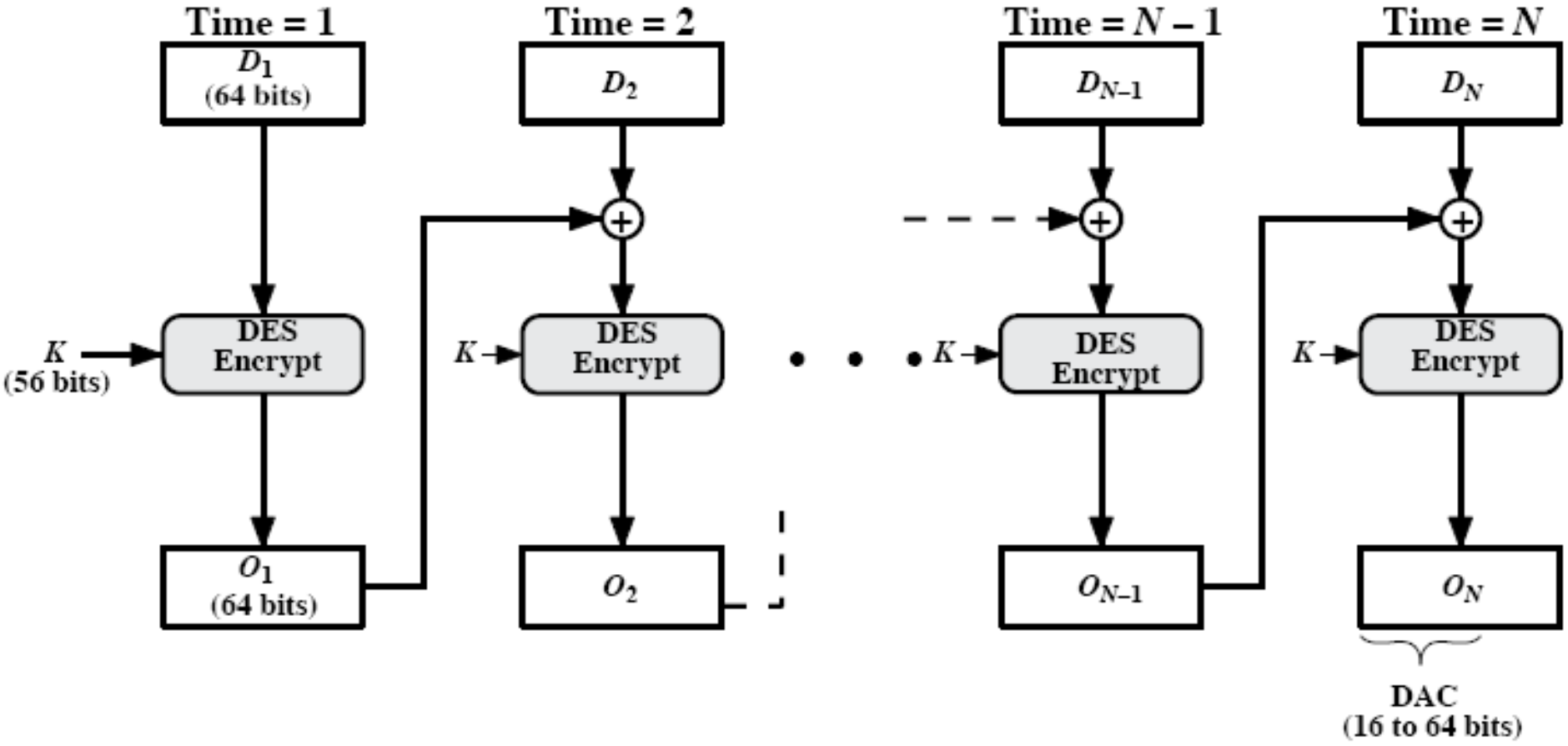
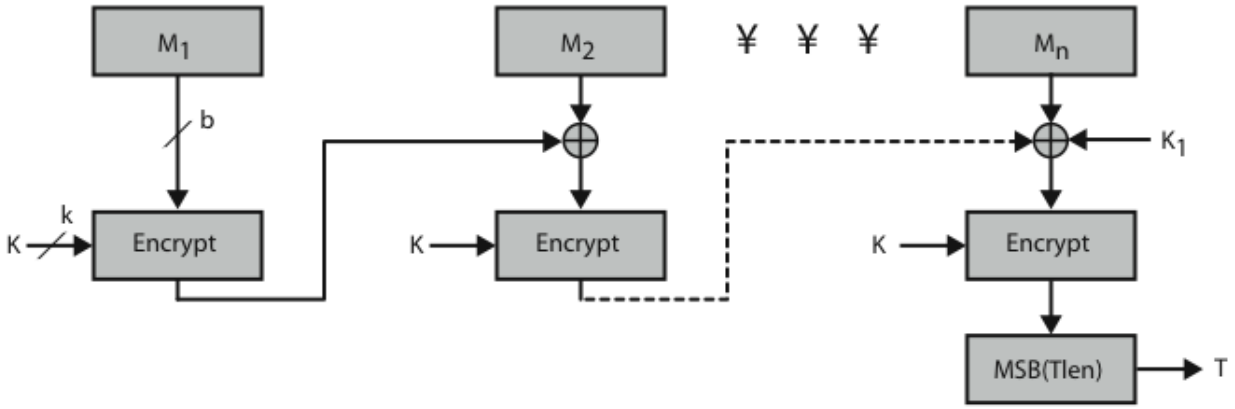


Figure 11.6 Data Authentication Algorithm (FIPS PUB 113)

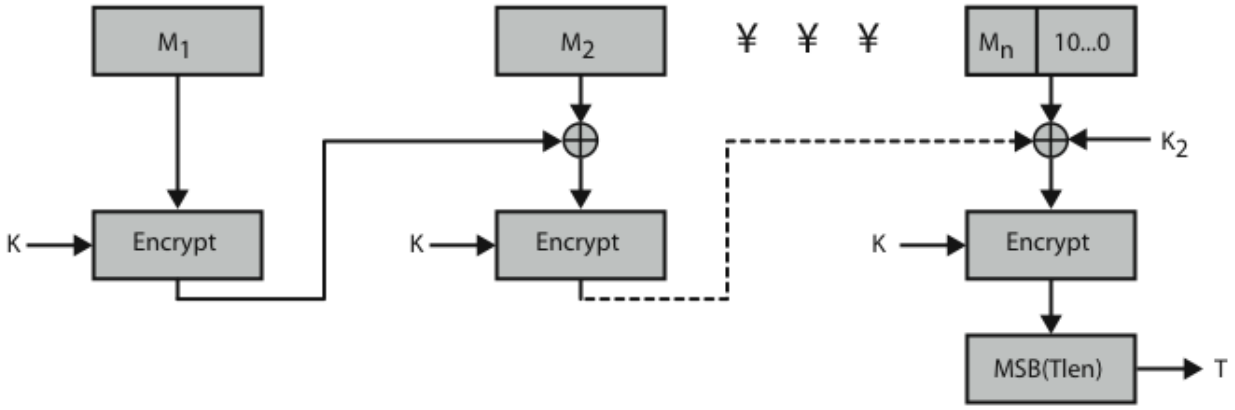
CMAC

- previously saw the DAA (CBC-MAC)
- widely used in govt & industry
- but has message size limitation
- can overcome using 2 keys & padding
- thus forming the Cipher-based Message Authentication Code (CMAC)
- adopted by NIST SP800-38B

CMAC Overview



(a) Message length is integer multiple of block size



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

Figure 12.12 Cipher-Based Message Authentication Code (CMAC)

Authenticated Encryption: CCM and GCM

- Counter with Cipher Block Chaining-Message Authentication Code, abbreviated CCM, that can provide assurance of the confidentiality and authenticity of data.
- CCM is based on an approved symmetric key block cipher algorithm whose block size is 128 bits, such as the Advanced Encryption Standard (AES)
- CCM cannot be used with the Triple Data Encryption Algorithm (3DES)
- CCM is intended for use in a packet environment, i.e., when all of the data is available in storage before CCM is applied;

Authenticated Encryption: CCM and GCM

- CCM is not designed to support partial processing or stream processing.
- The input to CCM includes three elements:
  - 1) data that will be both authenticated and encrypted, called the payload;
  - 2) associated data, e.g., a header, that will be authenticated but not encrypted; and
  - 3) a unique value, called a nonce, that is assigned to the payload and the associated data

Authenticated Encryption: CCM and GCM

- CCM consists of two related processes:
  - generation-encryption and
  - decryption-verification,
- which combine two cryptographic primitives: counter mode encryption and cipher block chaining-based authentication.
- Only the forward cipher function of the block cipher algorithm is used within these primitives.

CCM (generic description,, without details)

Algorithm $\mathrm{CBC}_K\ (M)$

10 Parse M into $M_1 \cdots M_m$ where $|M_i| = n$
11 $C_0 \leftarrow 0^n$
12 **for** $i \leftarrow 1$ **to** m **do**
13 $\quad C_i \leftarrow E_K(M_i \oplus C_{i-1})$
14 **return** C_m

Algorithm $\mathrm{CTR}_K^{\mathbf{N}}\ (M)$

20 $m \leftarrow \lceil |M|/n \rceil$
21 $S \leftarrow E_K(\mathbf{N}) \,\|\, E_K(\mathbf{N}{+}1) \,\|\, \cdots \,\|\, E_K(\mathbf{N}{+}m{-}1)$
22 $C \leftarrow M \oplus S\,[\text{first } |M| \text{ bits}]$
23 **return** C

Figure 1: Algorithms CBC and CTR, building blocks for this writeup. In both cases E : Key $\times \{0,1\}^n \to \{0,1\}^n$ is fixed and $K \in$ Key. For CBC we have $M \in (\{0,1\}^n)^+$ and for CTR we have $M \in \{0,1\}^*$ and $S \in \{0,1\}^n$.

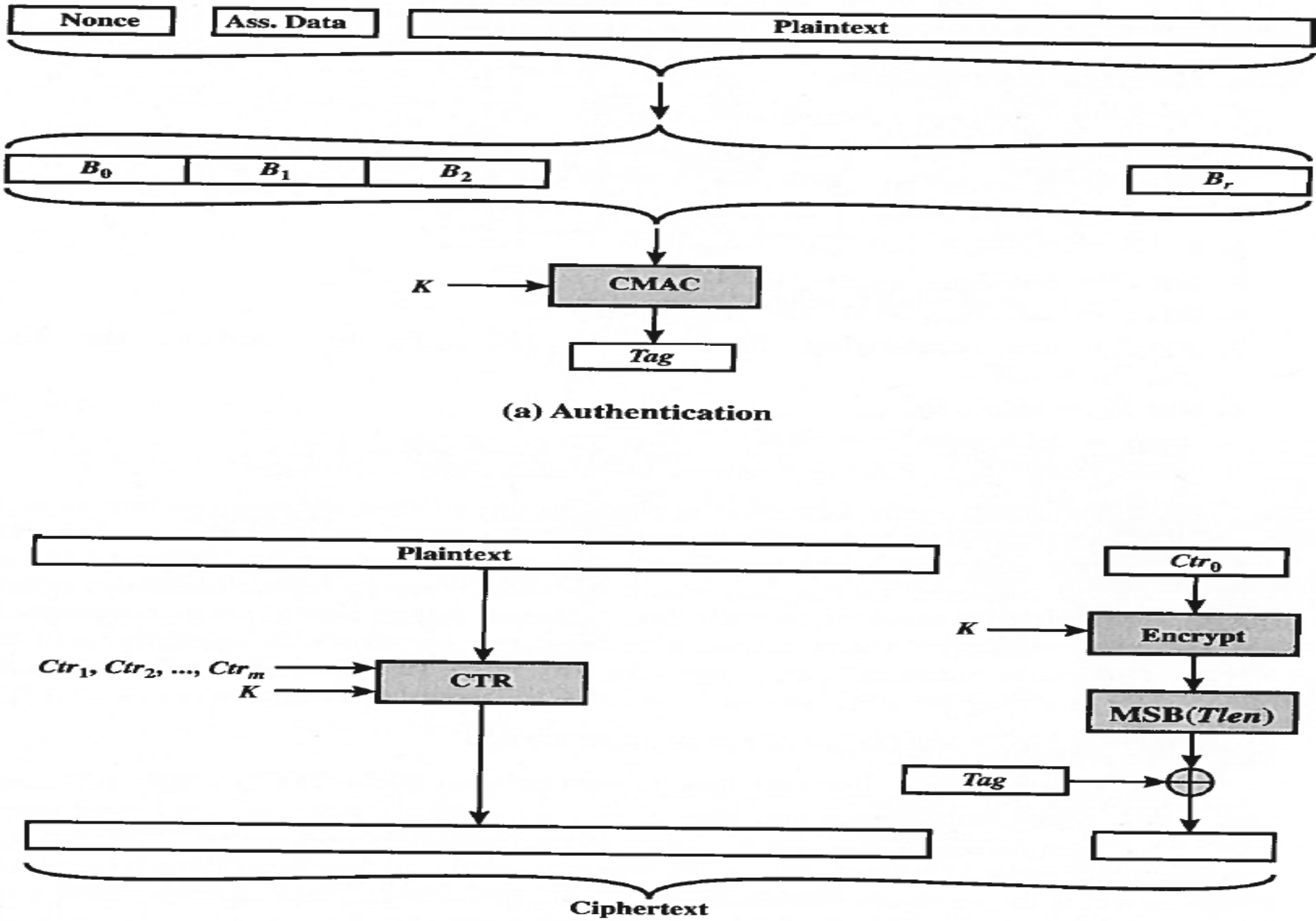


Figure 12.9 Counter with Cipher Block Chaining-Message Authentication Code (CCM)

CCM use and criticism

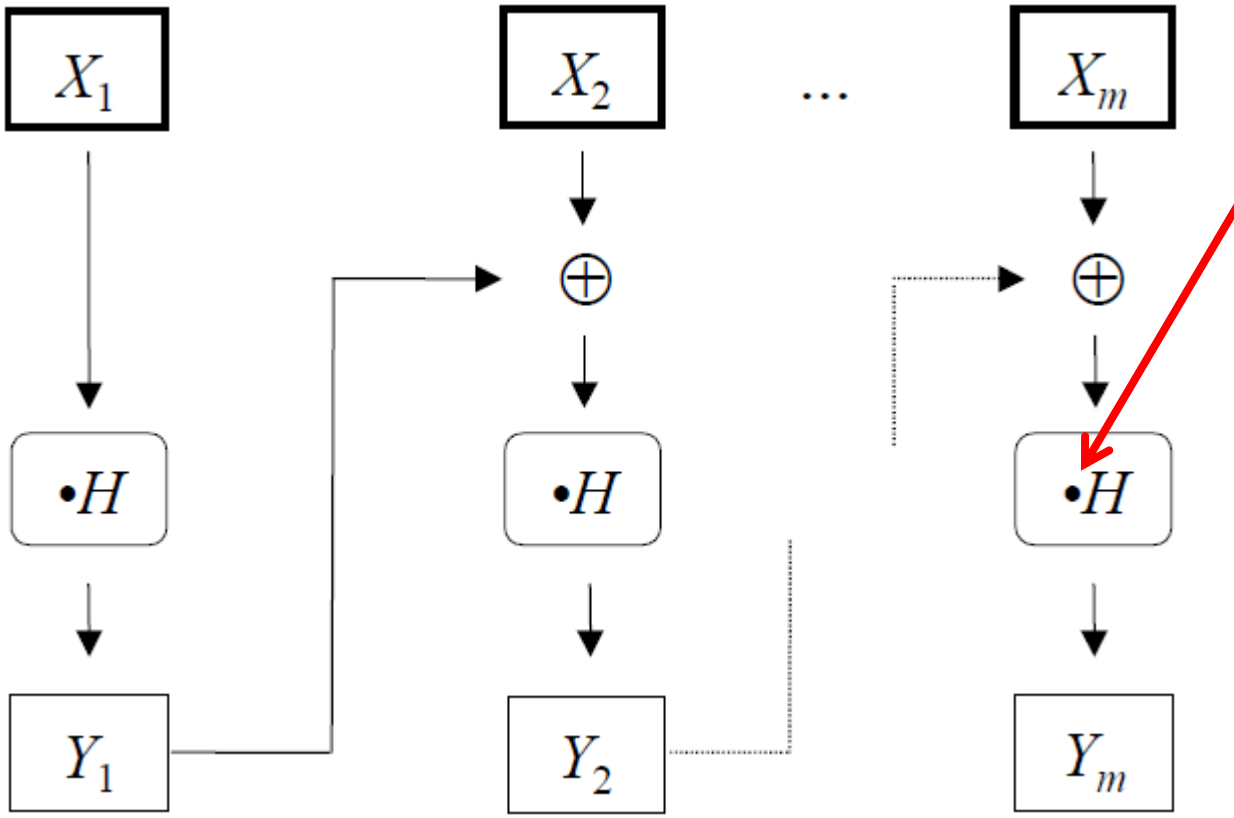
- In many standards of wireless networks such as
  - IEEE 802.11 [22] (WiFi),
  - IEEE 802.15.40 (Wireless Personal Area Network/ZigBee)
- CCM is not on-line,
- CCM disrupts word-alignment,
- CCM can’t preprocess static associated data,
- The main issue is that CCM is not on-line since the sender has to know the length of the message before the beginning of the encryption.

GCM (Galois/Counter Mode)

- NIST SP 800-38D
- The two functions that comprise GCM:
  - authenticated encryption and
  - authenticated decryption.
- GCM requires one block cipher operation and one 128-bit multiplication in GF(2^{128}) per each block (128 bit) of encrypted and authenticated data.
- Intel in the newest CPUs has added the PCLMULQDQ instruction, highlighting its use for GCM

GCM (Galois/Counter Mode)

$$GHASH_H(X_1 \| X_2 \| \ldots \| X_m) = Y_m$$



Multiplication in a field GF(2^{128}) with a block H.

GCM (Galois/Counter Mode)

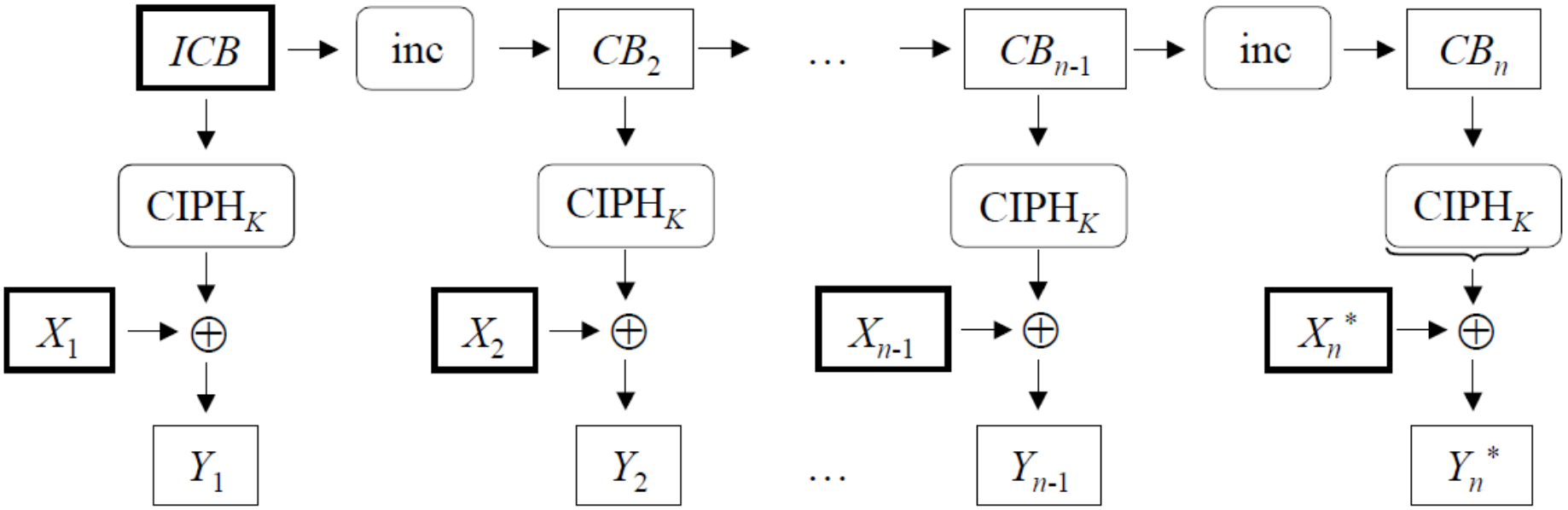


Figure 2: $\mathrm{GCTR}_K(ICB, X_1 \| X_2 \| \ldots \| X_n^*) = Y_1 \| Y_2 \| \ldots \| Y_n^*$.

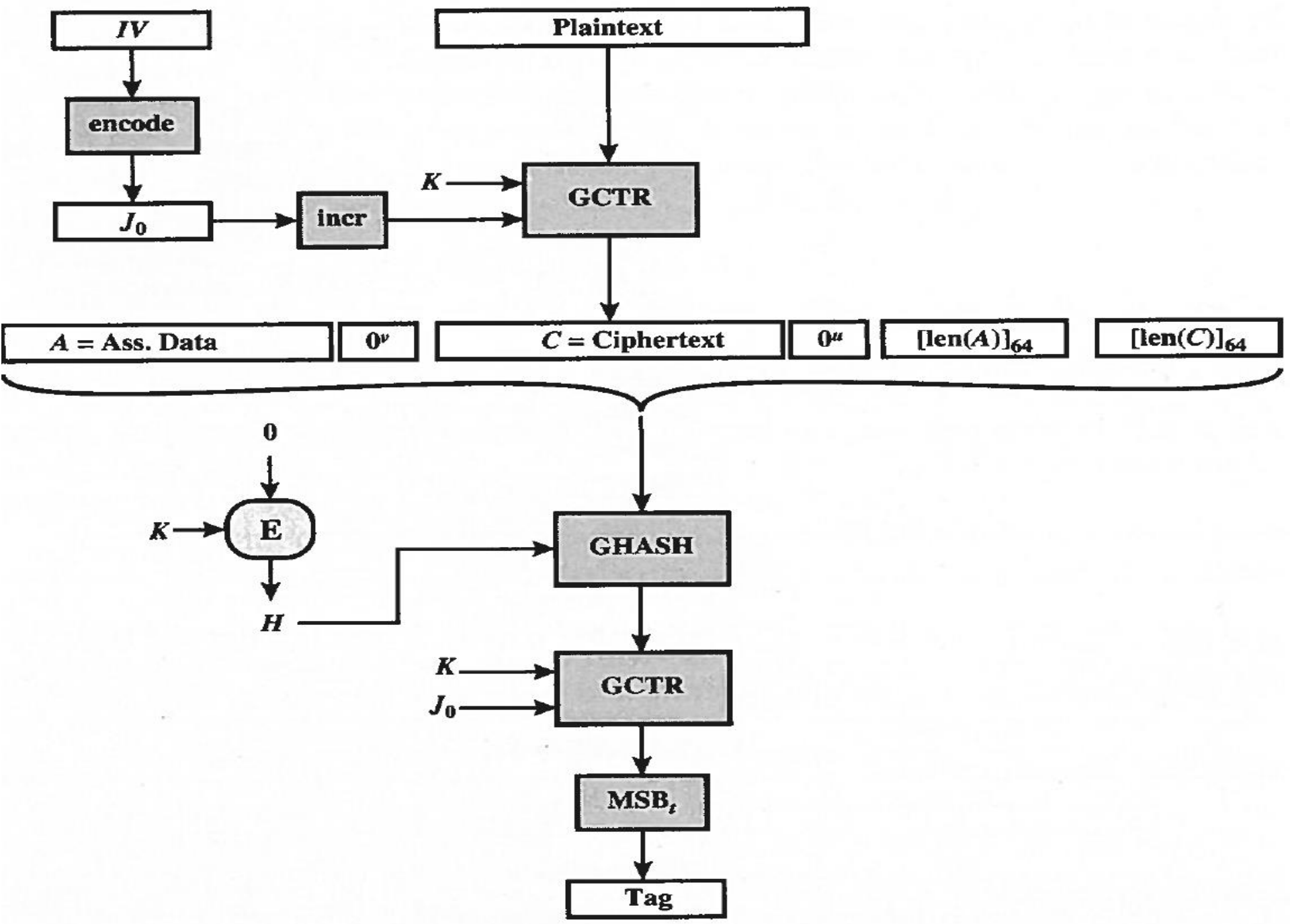
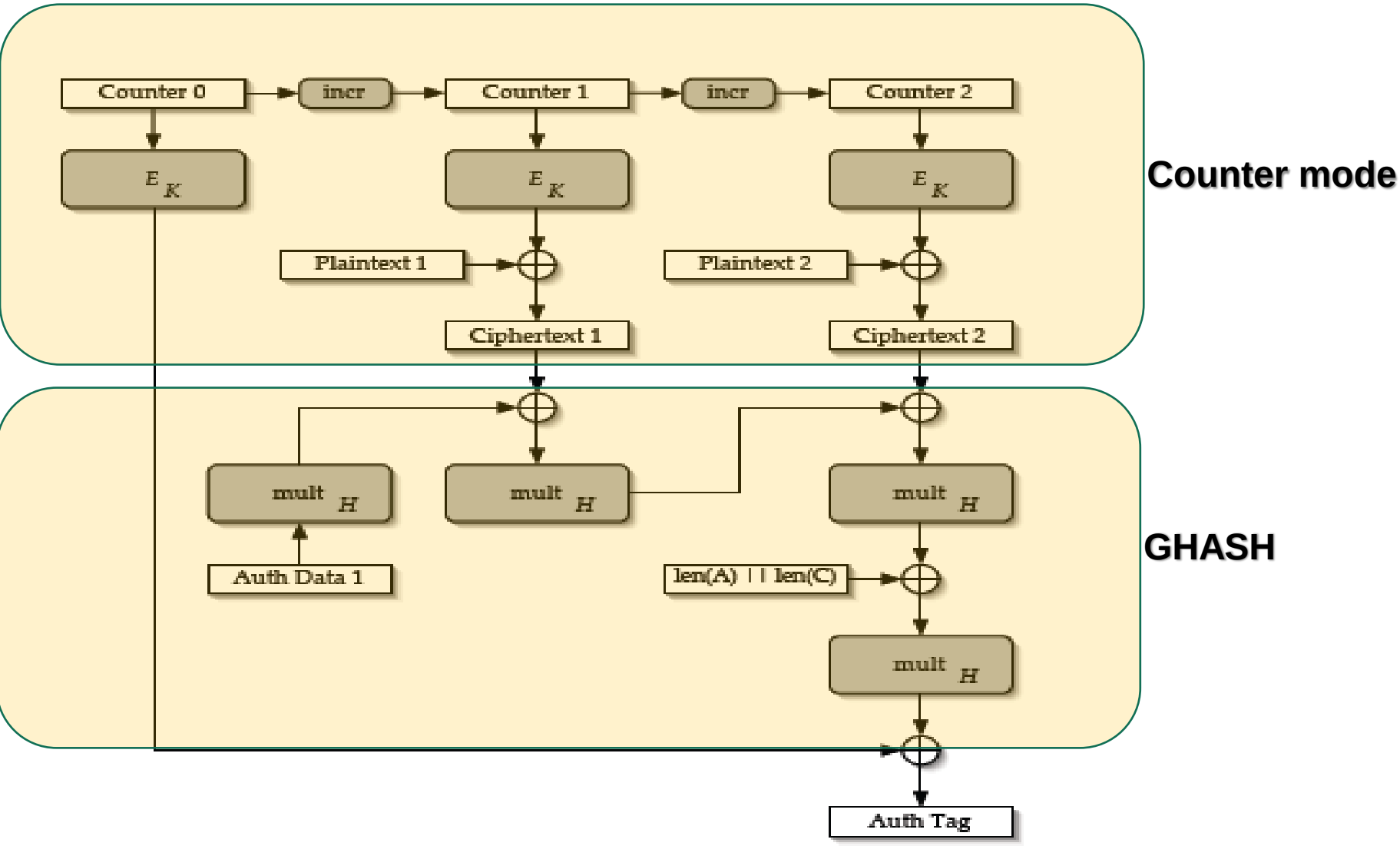


Figure 12.11 Galois Counter— Message Authentication Code (GCM)

GCM (Galois/Counter Mode)



GCM (Galois/Counter Mode)

- GCM mode is used in
  - IEEE 802.1AE (MACsec) Ethernet security,
  - ANSI (INCITS) Fibre Channel Security Protocols (FC-SP),
  - IEEE P1619.1 tape storage,
  - IETF IPsec standards,
  - SSH and TLS/SSL
  - AES-GCM is included into the NSA Suite B Cryptography

Summary

- have considered:
  - MAC
  - HMAC authentication using hash function
  - CMAC authentication using a block cipher
  - CCM
  - GCM