Cryptography and Network Security (CS435/890BN)

Part Ten

(Message Authentication Codes)

Message Authentication

- 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:
  - message encryption
  - message authentication code (MAC)
  - hash function

Security Requirements

In the context of communications across a network, the following attacks can be identified:

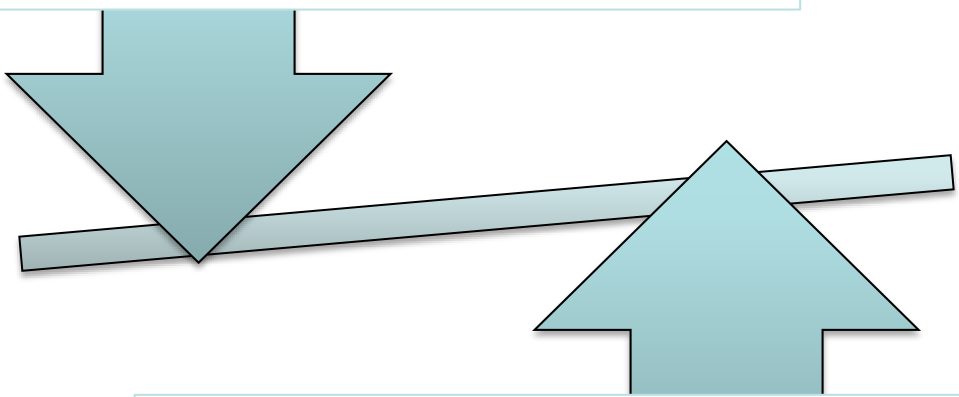
- Disclosure
  - Release of message contents to any person or process not possessing the appropriate cryptographic key
- Traffic analysis
  - Discovery of the pattern of traffic between parties
- Masquerade
  - Insertion of messages into the network from a fraudulent source
- Content modification
  - Changes to the contents of a message, including insertion, deletion, transposition, and modification
- Sequence modification
  - Any modification to a sequence of messages between parties, including insertion, deletion, and reordering
- Timing modification
  - Delay or replay of messages
- Source repudiation
  - Denial of transmission of message by source
- Destination repudiation
  - Denial of receipt of message by destination

Message Authentication Functions

- Two levels of

Lower level

- There must be some sort of function that produces an authenticator



Higher-level

- Uses the lower-level function as a primitive in an authentication protocol that enables a receiver to verify the authenticity of a message

– Hash function
  - A function that maps a message of any length into a fixed-length hash value which serves as the authenticator

– Message encryption
  - The ciphertext of the entire message serves as its authenticator

– Message authentication code (MAC)
  - A function of the message and a secret key that produces a fixed-length value that serves as the authenticator

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
  - know content cannot of been altered
  - if message has suitable structure, redundancy or a checksum to detect any changes

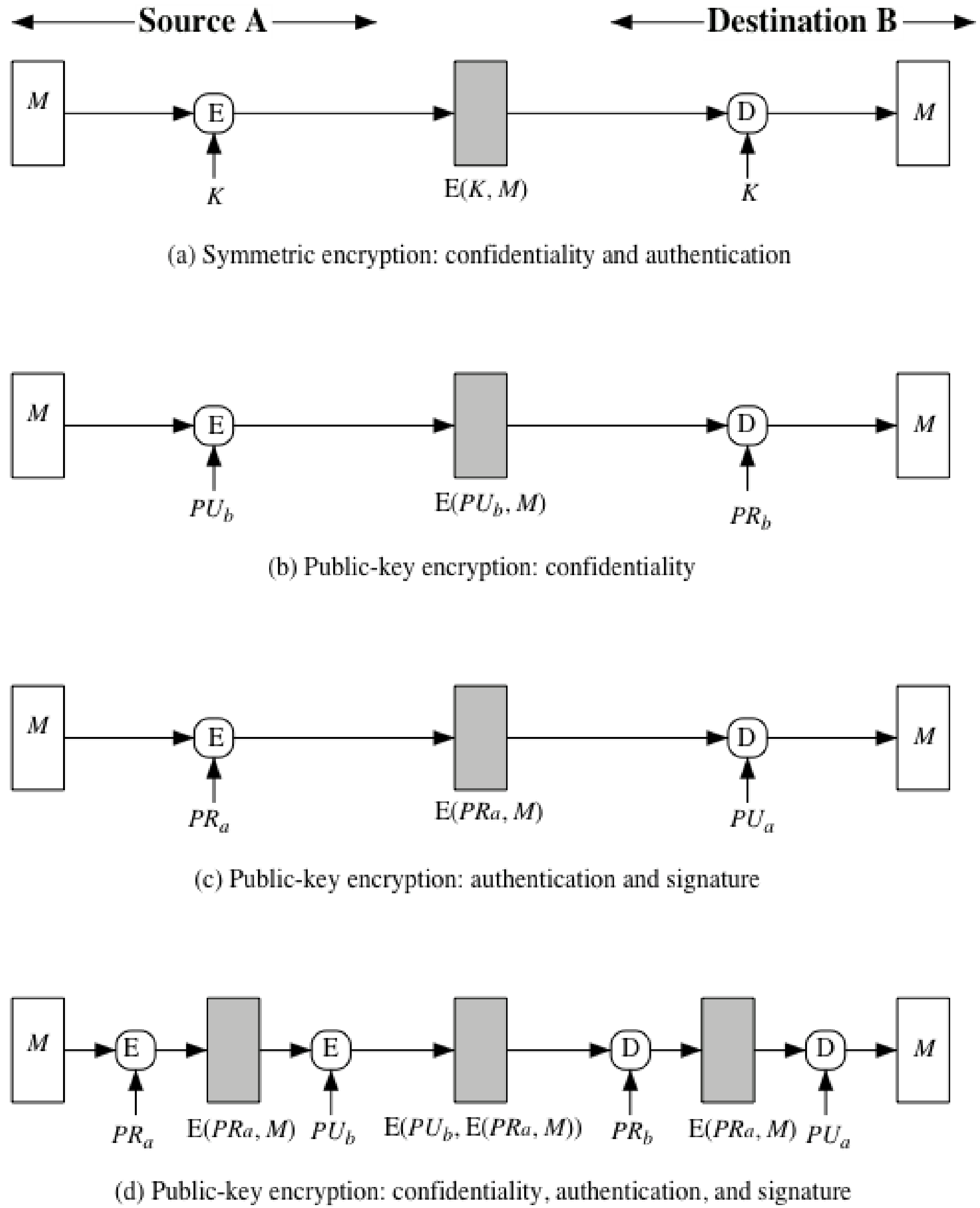


Figure 12.1 Basic Uses of Message Encryption

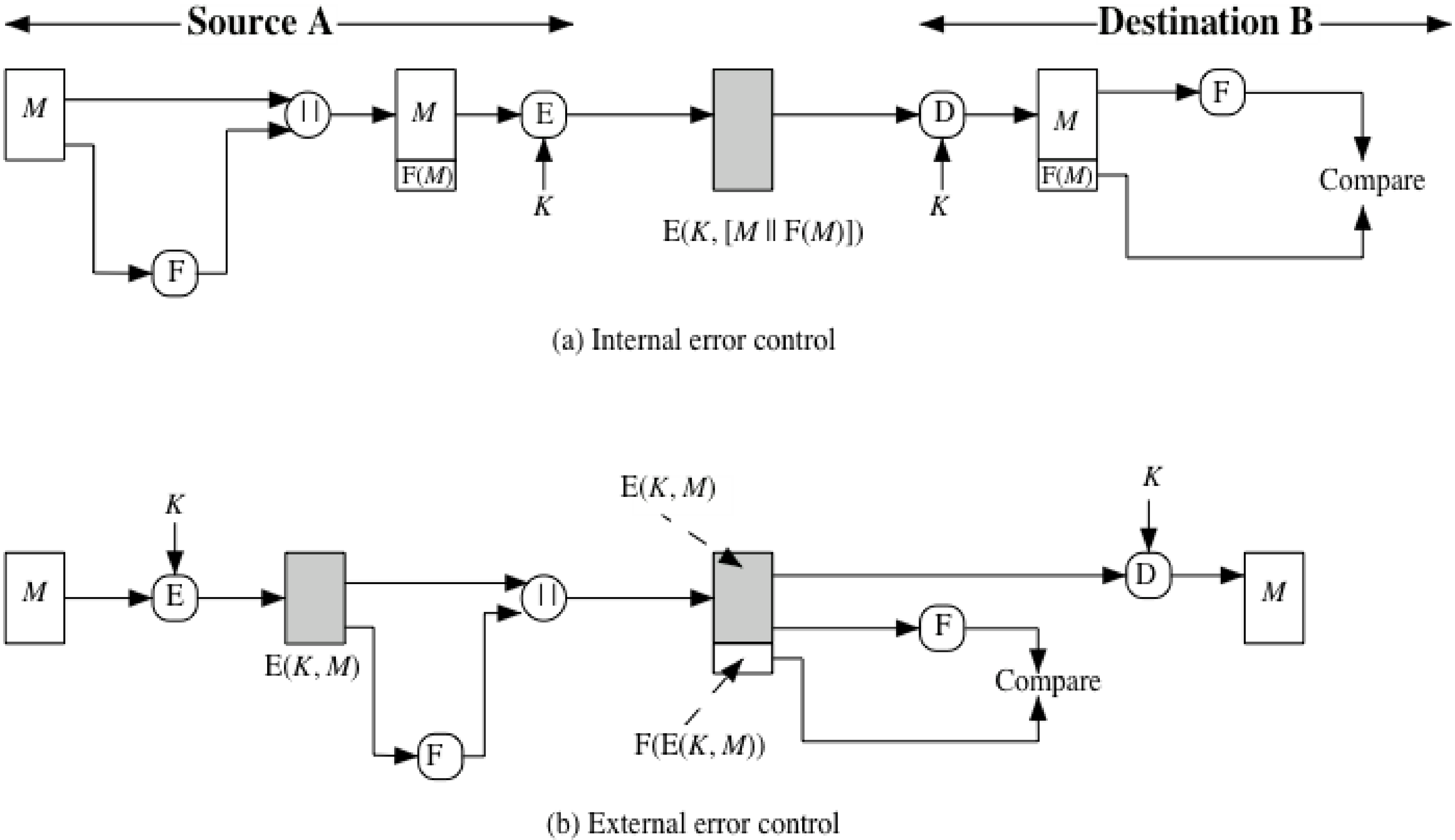


Figure 12.2 Internal and External Error Control

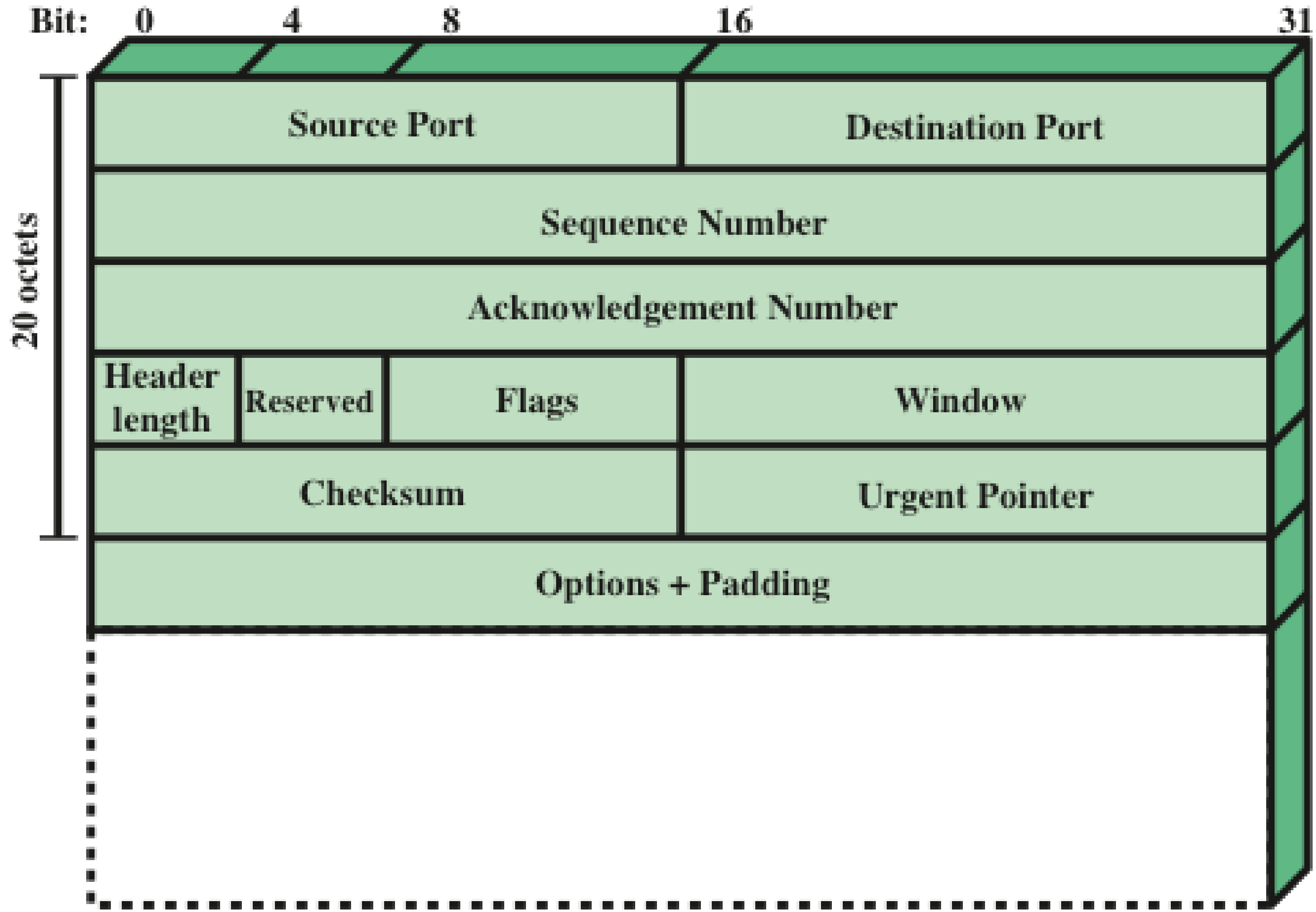


Figure 12.3 TCP Segment

Public-Key Encryption

- The straightforward use of public-key encryption provides confidentiality but not authentication
- To provide both confidentiality and authentication, A can encrypt *M* first using its private key which provides the digital signature, and then using B's public key, which provides confidentiality
- Disadvantage is that the public-key algorithm must be exercised four times rather than two in each communication

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

MAC Properties

- a MAC is a cryptographic checksum

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

  – 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

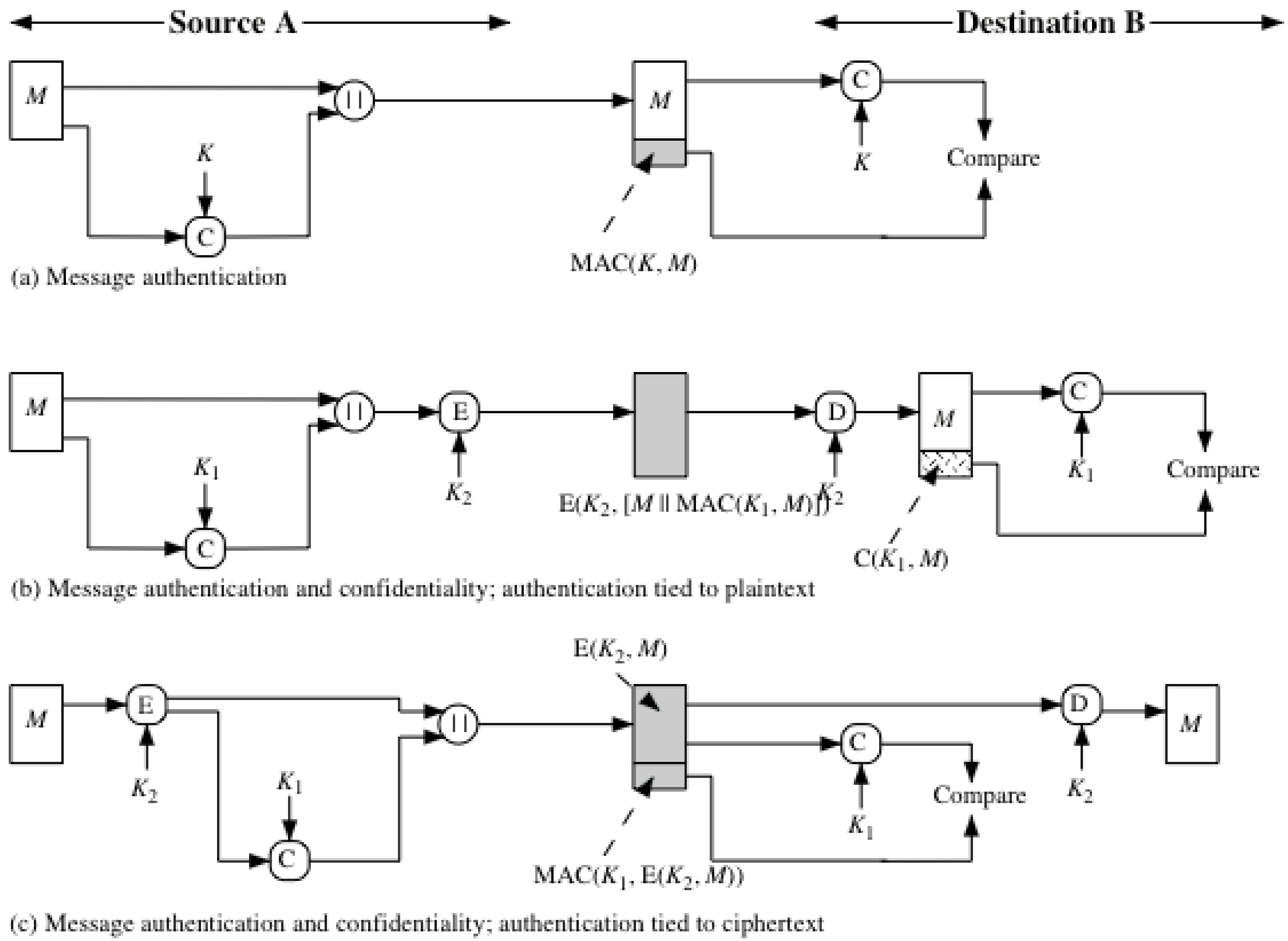


Figure 12.4 Basic Uses of Message Authentication Code (MAC)

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 message authentication is needed
  - sometimes need authentication to persist longer than the encryption (eg. archival use)
- note that a MAC is not a digital signature

Requirements for MACs

Taking into account the types of attacks, the MAC needs to satisfy the following:

The first requirement deals with message replacement attacks, in which an opponent is able to construct a new message to match a given MAC, even though the opponent does not know and does not learn the key

The second requirement deals with the need to thwart a brute-force attack based on chosen plaintext

The final requirement dictates that the authentication algorithm should not be weaker with respect to certain parts or bits of the message than others

Brute-Force Attack

- Requires known message-tag pairs
  - A brute-force method of finding a collision is to pick a random bit string y and check if $H(y) = H(x)$

Two lines of attack:

- Attack the key space
  - If an attacker can determine the MAC key then it is possible to generate a valid MAC value for any input x
- Attack the MAC value
  - Objective is to generate a valid tag for a given message or to find a message that matches a given tag

Cryptanalysis

- Cryptanalytic attacks seek to exploit some property of the algorithm to perform some attack other than an exhaustive search
- An ideal MAC algorithm will require a cryptanalytic effort greater than or equal to the brute-force effort
- There is much more variety in the structure of MACs than in hash functions, so it is difficult to generalize about the cryptanalysis of MACs

HMAC Design Objectives

RFC 2104 lists the following objectives for HMAC:

- **To use, without modifications, available hash functions**
- **To allow for easy replaceability of the embedded hash function in case faster or more secure hash functions are found or required**
- **To preserve the original performance of the hash function without incurring a significant degradation**
- **To use and handle keys in a simple way**
- **To have a well understood cryptographic analysis of the strength of the authentication mechanism based on reasonable assumptions about the embedded hash function**

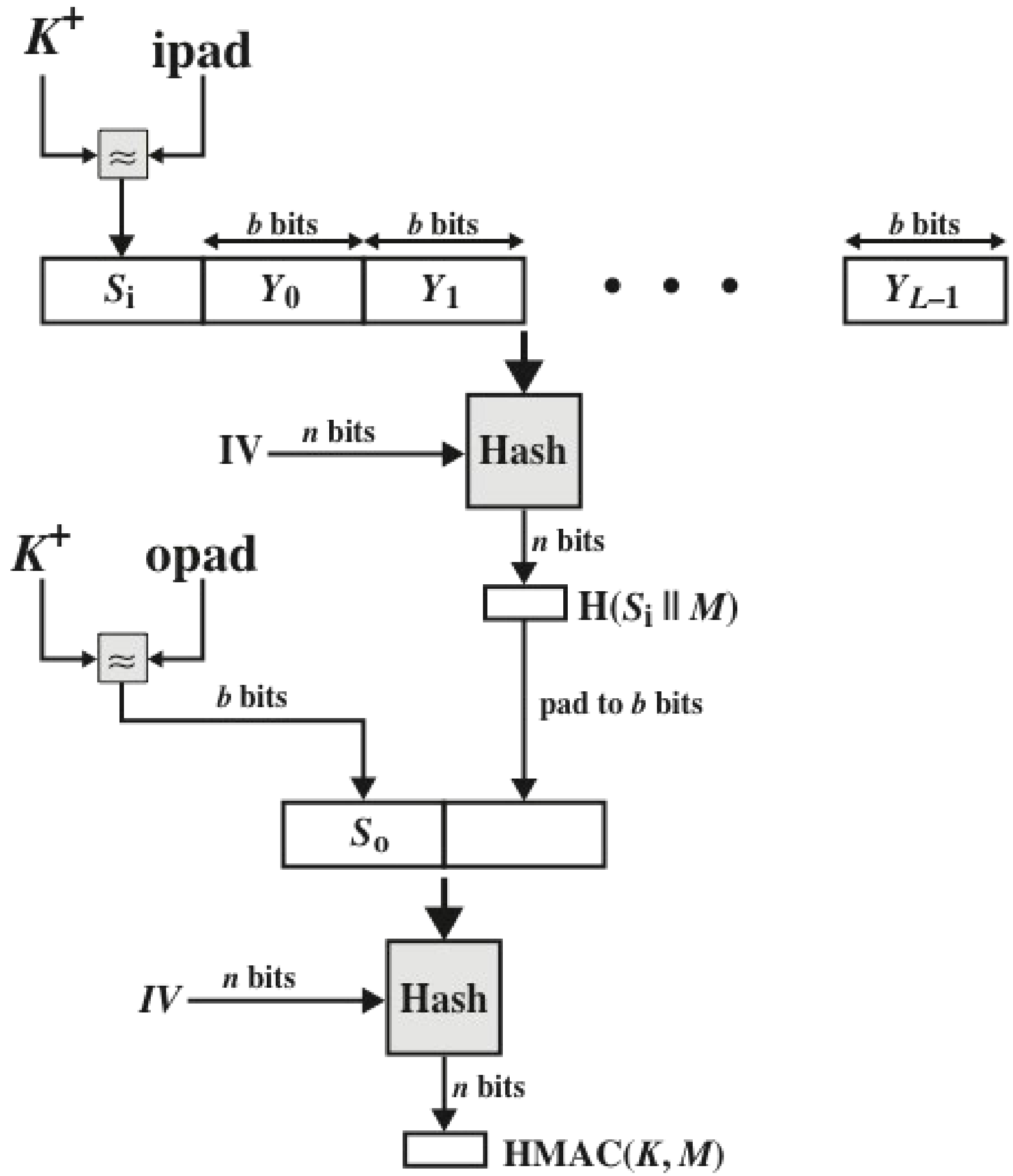


Figure 12.5 HMAC Structure

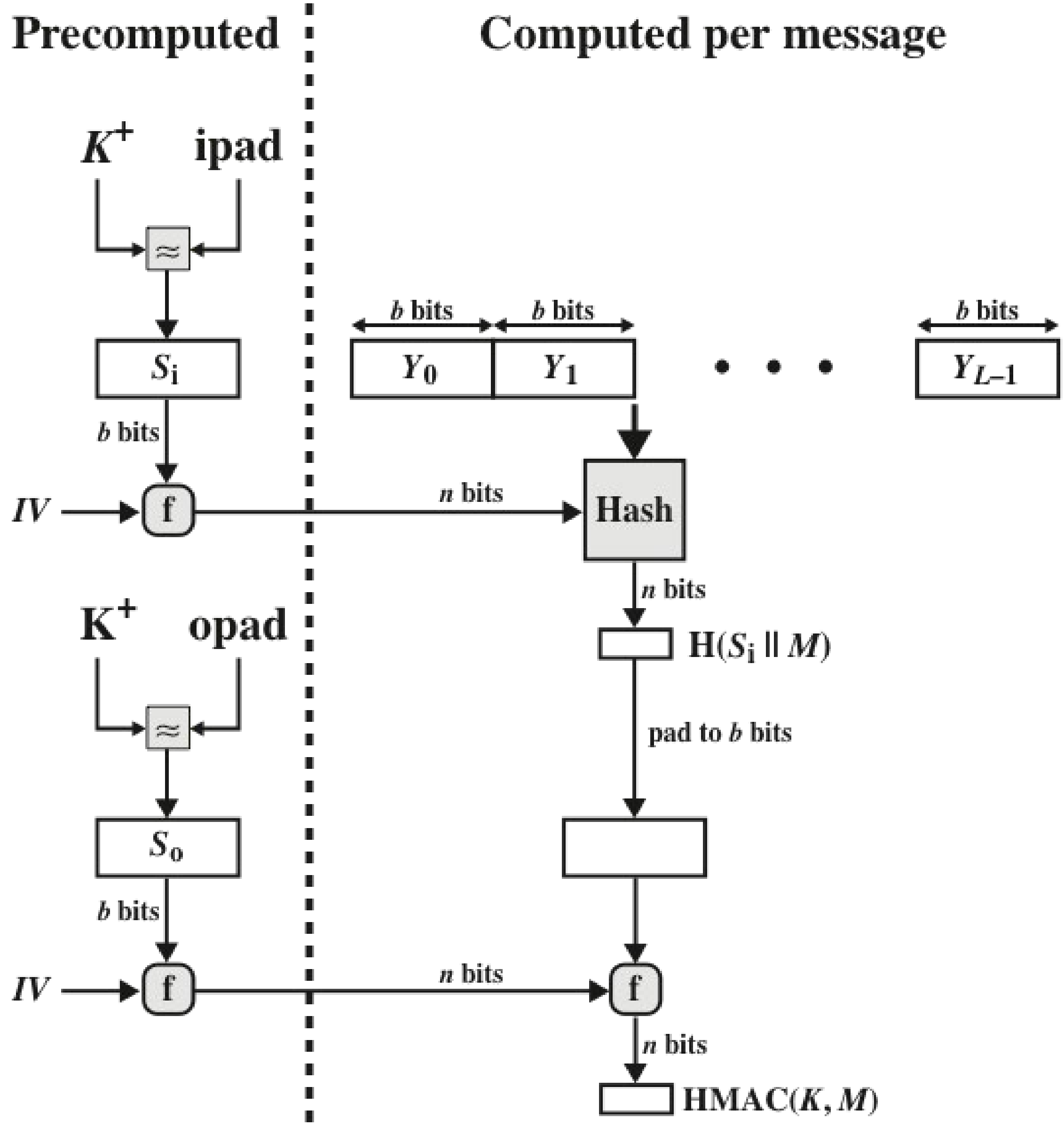
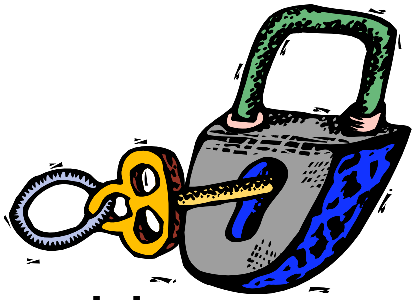


Figure 12.6 Efficient Implementation of HMAC

Security of HMAC



- Depends in some way on the cryptographic strength of the underlying hash function
- Appeal of HMAC is that its designers have been able to prove an exact relationship between the strength of the embedded hash function and the strength of HMAC
- Generally expressed in terms of the probability of successful forgery with a given amount of time spent by the forger and a given number of message-tag pairs created with the same key

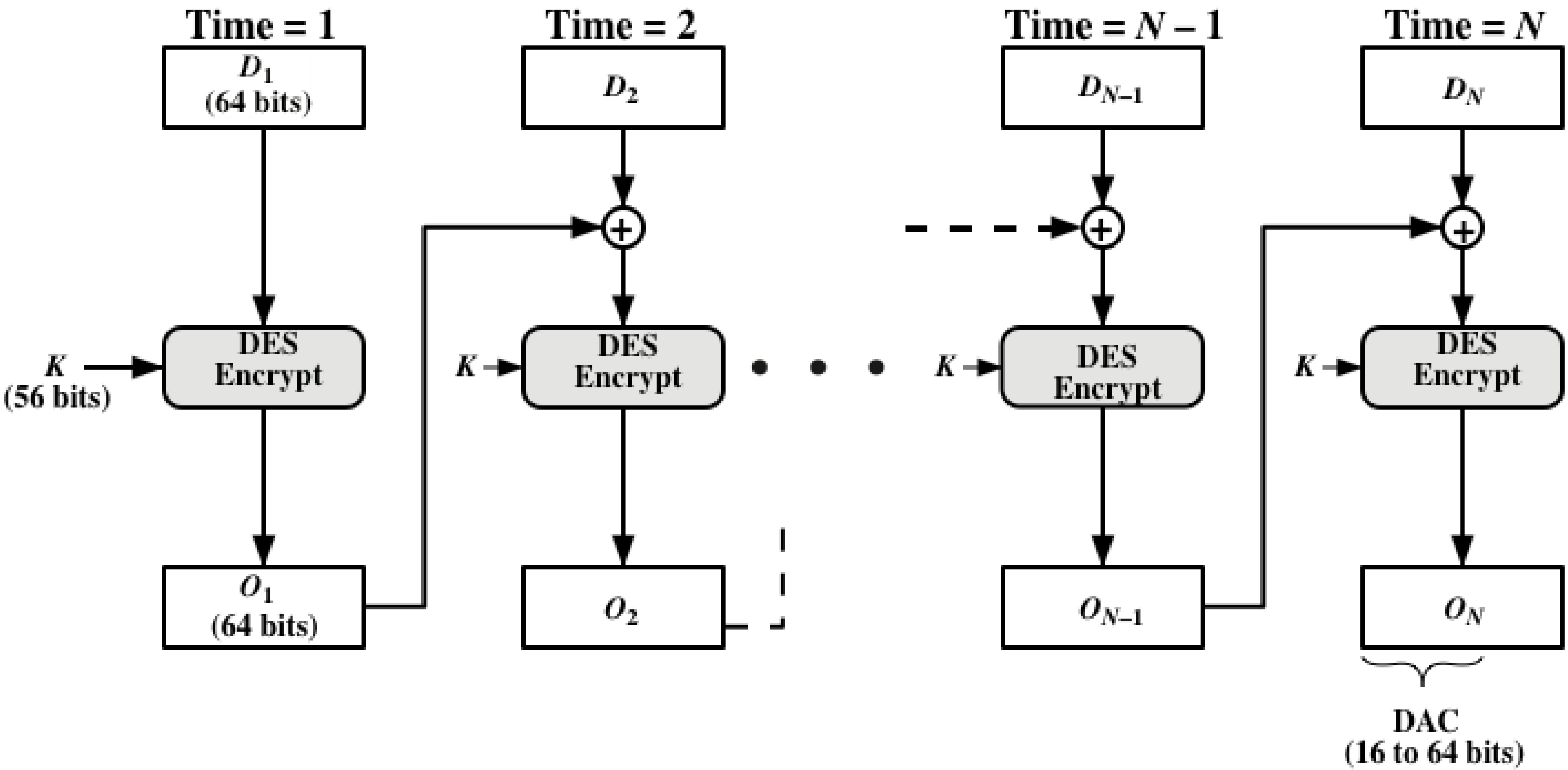
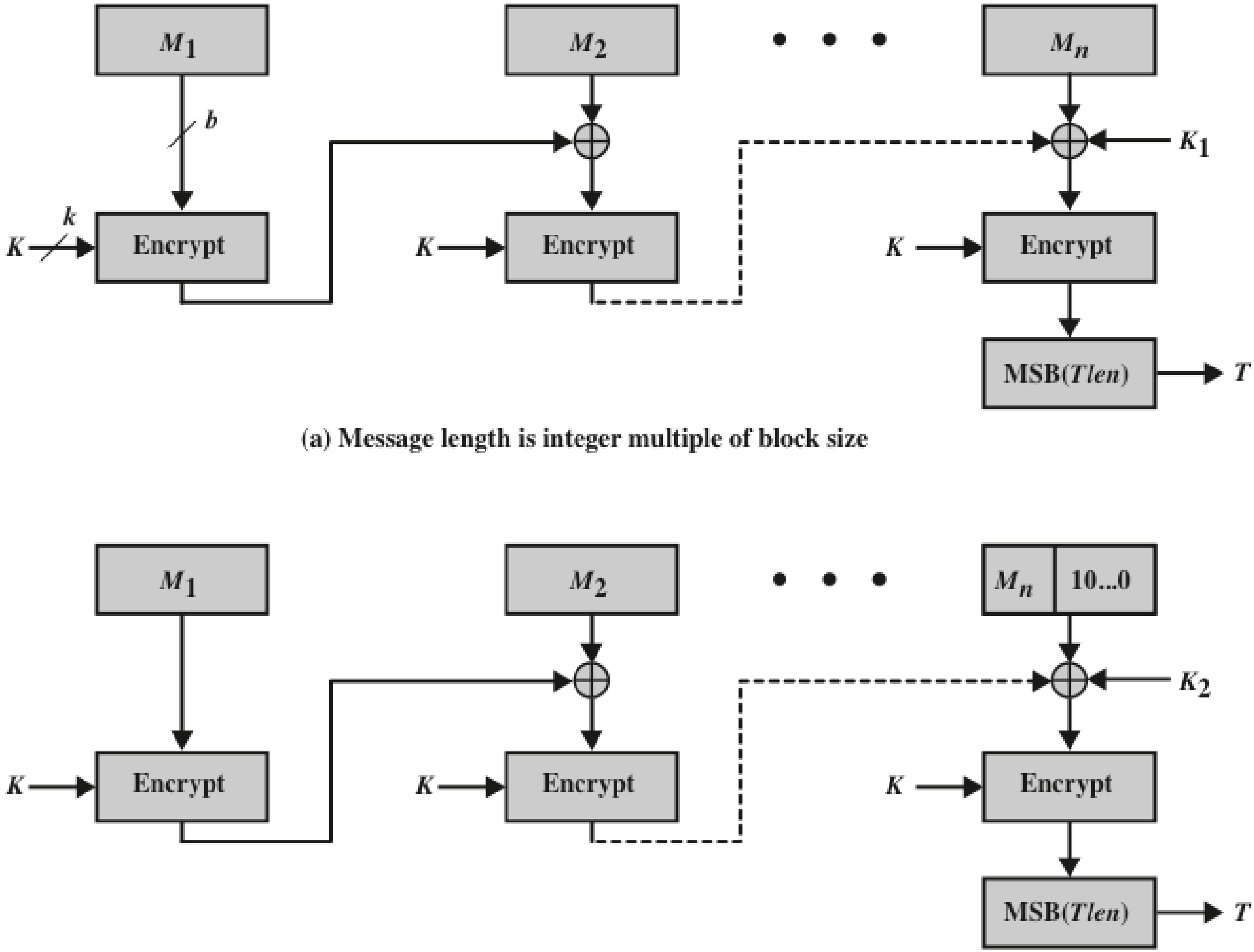


Figure 12.7 Data Authentication Algorithm (FIPS PUB 113)



(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)

Authenticated Encryption (AE)

- A term used to describe encryption systems that simultaneously protect confidentiality and authenticity of communications
- Approaches:
  - Hashing followed by encryption
  - Authentication followed by encryption
  - Encryption followed by authentication
  - Independently encrypt and authenticate
- Both decryption and verification are straightforward for each approach
- There are security vulnerabilities with all of these approaches

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

- Was standardized by NIST specifically to support the security requirements of IEEE 802.11 WiFi wireless local area networks
- Variation of the encrypt-and-MAC approach to authenticated encryption
  - Defined in NIST SP 800-38C
- Key algorithmic ingredients:
  - AES encryption algorithm
  - CTR mode of operation
  - CMAC authentication algorithm
- Single key *K* is used for both encryption and MAC algorithms

The input to the CCM encryption process consists of three elements:

Data that will be both authenticated and encrypted

This is the plaintext message *P* of the data block

Associated data *A* that will be authenticated but not encrypted

An example is a protocol header that must be transmitted in the clear for proper protocol operation but which needs to be authenticated

A nonce *N* that is assigned to the payload and the associated data

This is a unique value that is different for every instance during the lifetime of a protocol association and is intended to prevent replay attacks and certain other types of attacks

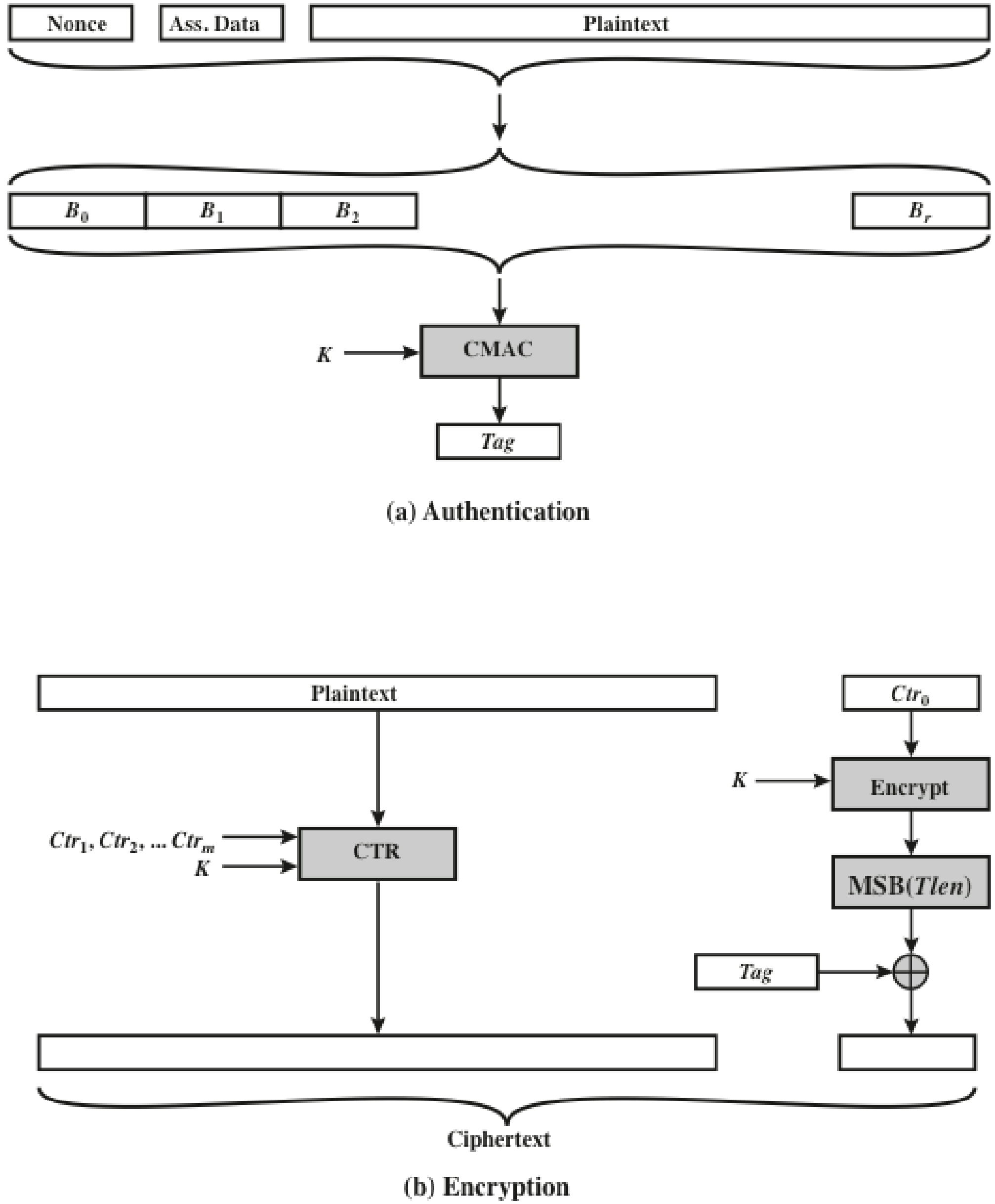
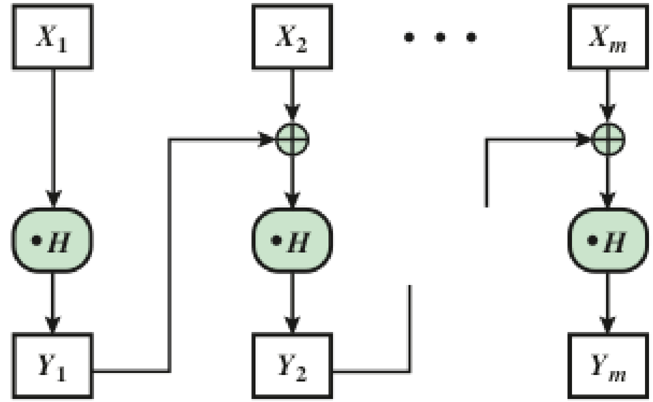


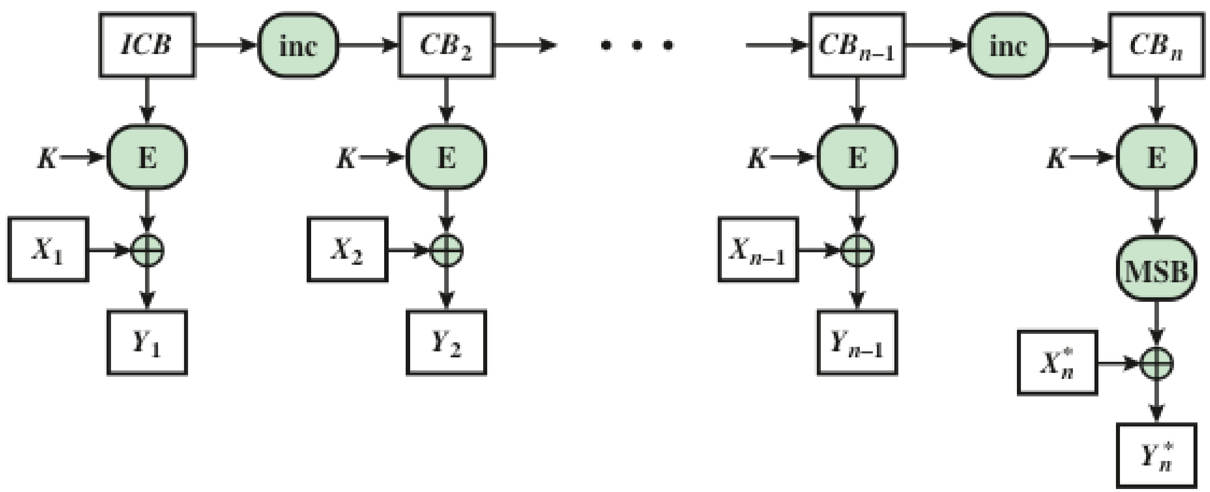
Figure 12.9 Counter with Cipher Block Chaining-Message Authentication Code (CCM)

Galois/Counter Mode (GCM)

- NIST standard SP 800-38D
- Designed to be parallelizable so that it can provide high throughput with low cost and low latency
  - Message is encrypted in variant of CTR mode
  - Resulting ciphertext is multiplied with key material and message length information over GF (2^{128}) to generate the authenticator tag
  - The standard also specifies a mode of operation that supplies the MAC only, known as GMAC
- Makes use of two functions:
  - GHASH - a keyed hash function
  - GCTR - CTR mode with the counters determined by simple increment by one operation



(a) $\mathbf{GHASH}_H(X_1 \,\|\, X_2 \,\|\, \ldots \,\|\, X_m) = Y_m$



(b) $\mathbf{GCTR}_K(ICB, X_1 \,\|\, X_2 \,\|\, \ldots \,\|\, X_n) = Y_1 \,\|\, Y_2 \,\|\, \ldots \,\|\, Y_n$

Figure 12.10 GCM Authentication and Encryption Functions

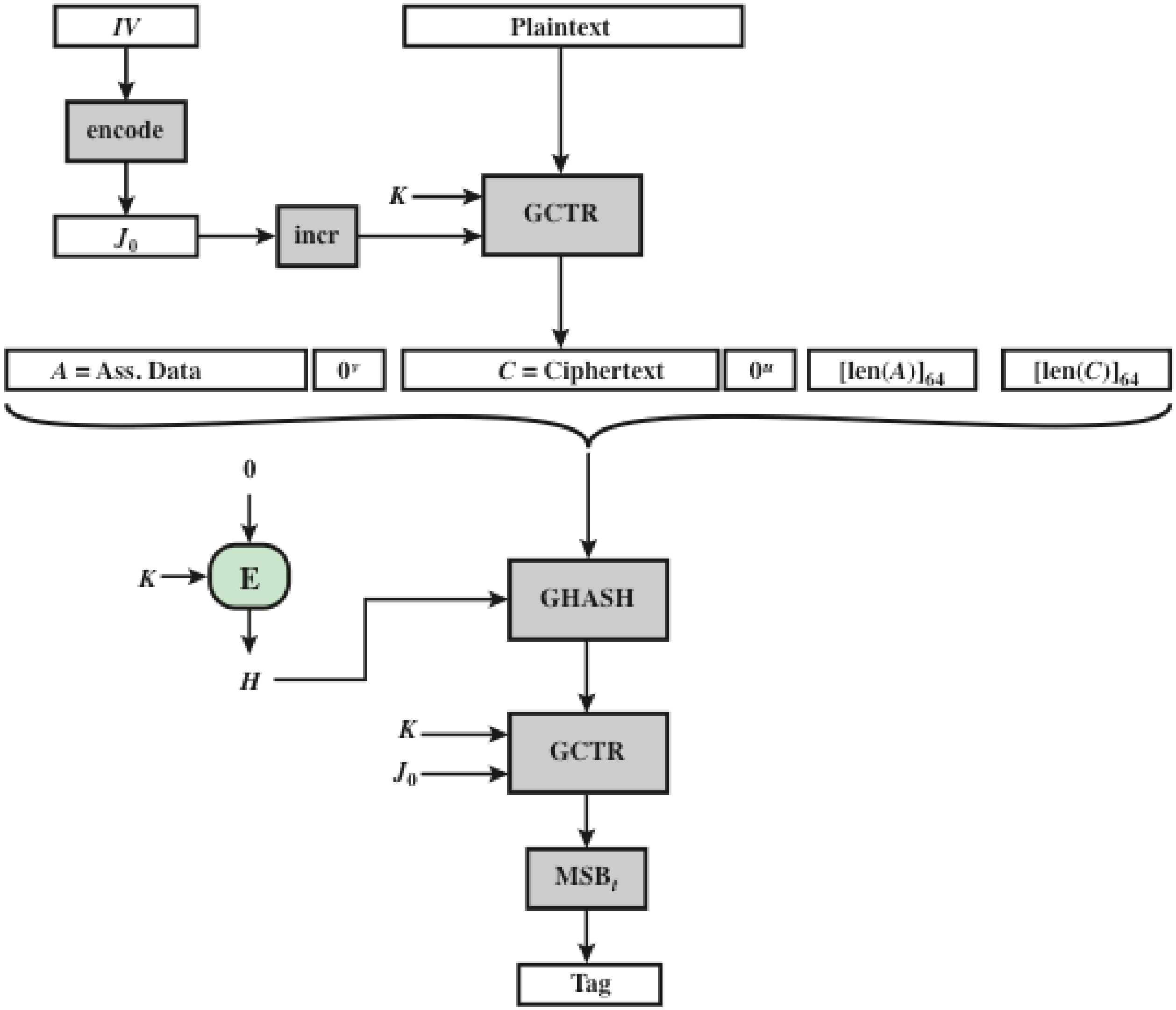


Figure 12.11 Galois Counter - Message Authentication Code (GCM)

Key Wrap (KW)

- Most recent block cipher mode of operation defined by NIST
  - Uses AES or triple DEA as the underlying encryption algorithm
- Purpose is to securely exchange a symmetric key to be shared by two parties, using a symmetric key already shared by those parties
  - The latter key is called a *key encryption key* (KEK)

– Robust in the sense that each bit of output can be expected to depend in a nontrivial fashion on each bit of input

– Only used for small amounts of plaintext

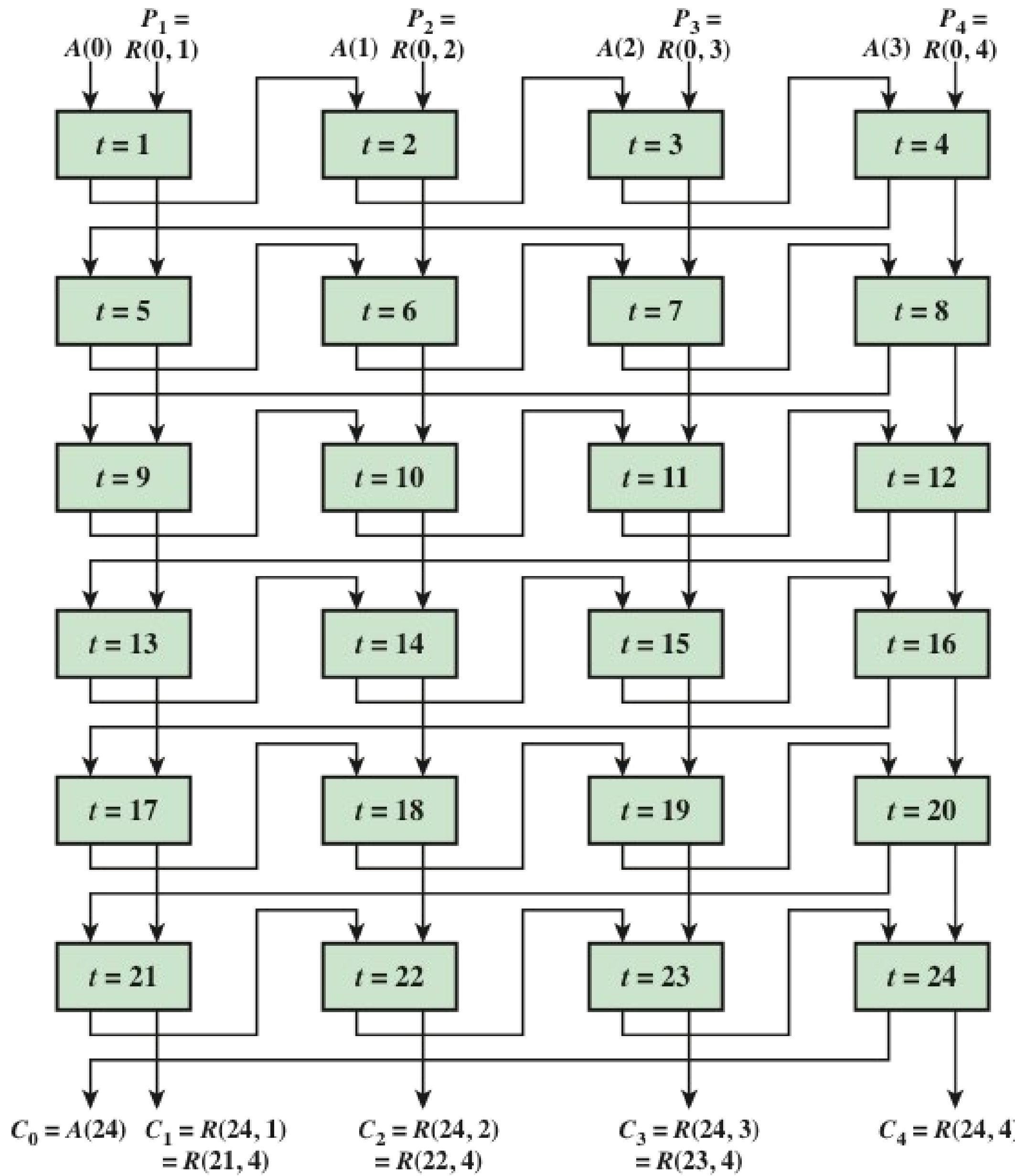


Figure 12.12 Key Wrapping Operation for 256-bit Key

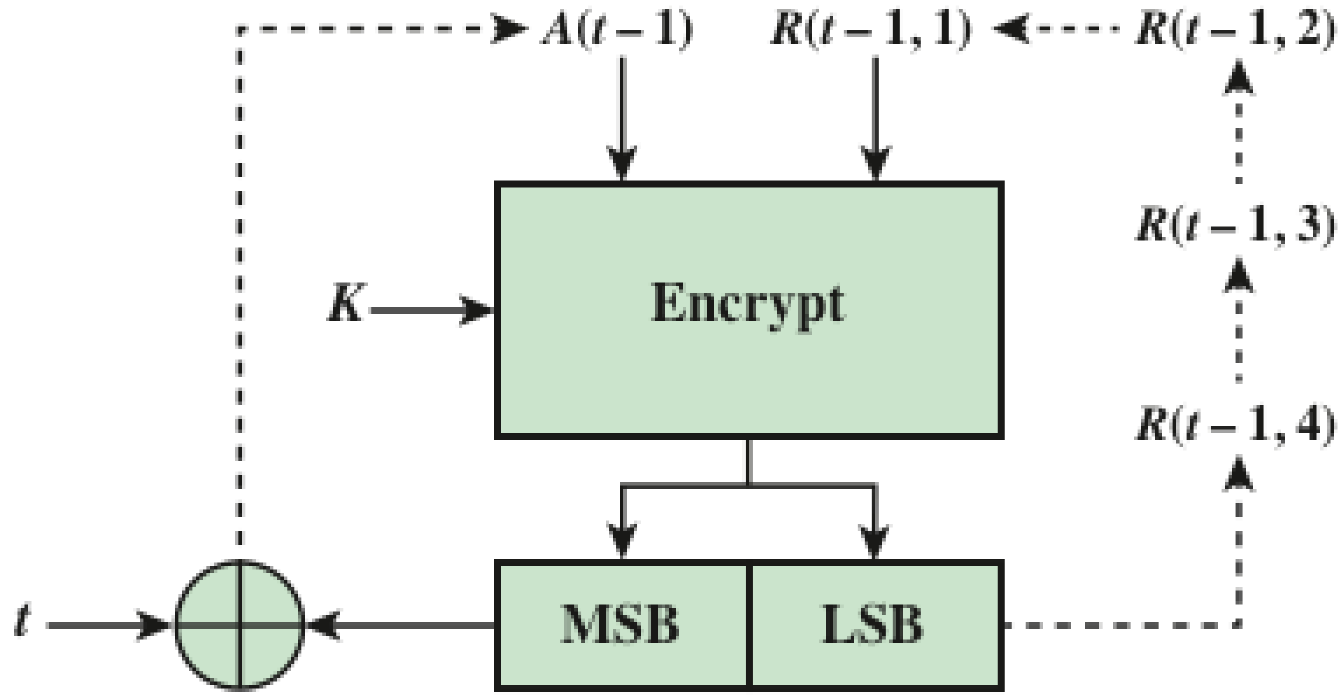
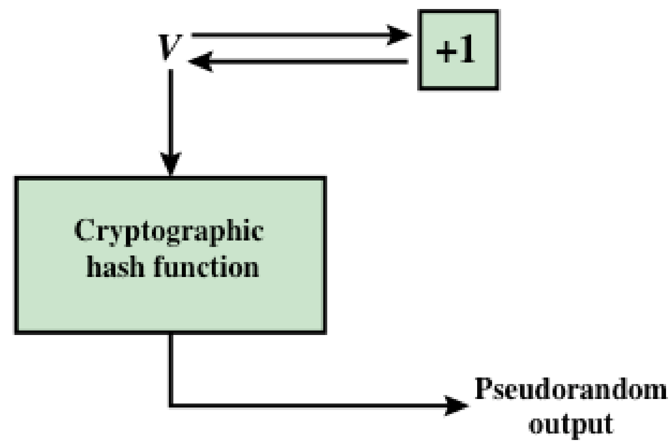


Figure 12.13 Key Wrapping Operation for 256-bit Key: stage *t*

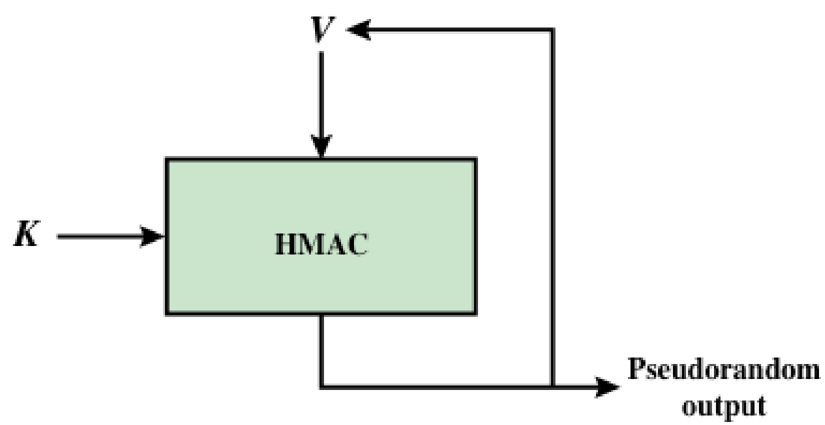
Pseudorandom Number Generation Using Hash Functions and MACs

- Essential elements of any pseudorandom number generator (PRNG) are a seed value and a deterministic algorithm for generating a stream of pseudorandom bits
  - If the algorithm is used as a pseudorandom function (PRF) to produce a required value, the seed should only be known to the user of the PRF
  - If the algorithm is used to produce a stream encryption function, the seed has the role of a secret key that must be known to the sender and the receiver

- A hash function or MAC produces apparently random output and can be used to build a PRNG



(a) PRNG using cryptographic hash function



(b) PRNG using HMAC

Figure 12.14 Basic Structure of Hash-Based PRNGs (SP 800-90)

NIST SP 800-90	IEEE 802.11i	TLS/WTLS

$m = \lceil n/\text{outlen} \rceil$
$w_0 = V$
W = the null string
For i = 1 to m
    $w_i = \text{MAC}(K, w_{i-1})$
    $W = W \parallel w_i$
Return leftmost n bits of W
	$m = \lceil n/\text{outlen} \rceil$
	W = the null string
	For i = 1 to m
	    $w_i = \text{MAC}(K, (V \parallel i))$
	    $W = W \parallel w_i$
	Return leftmost n bits of W
		$m = \lceil n/\text{outlen} \rceil$
		$\text{A}(0) = V$
		W = the null string
		For i = 1 to m
		    $\text{A}(i) = \text{MAC}(K, \text{A}(i-1))$
		    $w_i = \text{MAC}(K, (\text{A}(i) \parallel V)$
		    $W = W \parallel w_i$
		Return leftmost n bits of W

Figure 12.15 Three PRNGs Based on HMAC