

# Computer Systems Security

**Dr. Ayman Abdel-Hamid**

College of Computing and Information Technology

Arab Academy for Science & Technology and  
Maritime Transport

## Chapter 12 – Message Authentication

# Message Authentication

- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)
- Which attacks could be dealt with through message authentication?

# Security Attacks

- |                            |   |
|----------------------------|---|
| 1. disclosure              | Dealt with through<br>message confidentiality |
| 2. traffic analysis        |   |
| <hr/>                      |   |
| 3. masquerade              | Dealt with through<br>message authentication  |
| 4. content modification    |   |
| 5. sequence modification   |   |
| 6. timing modification     |   |
| <hr/>                      |   |
| 7. source repudiation      | Digital signatures                            |
| <hr/>                      |   |
| 8. destination repudiation |   |

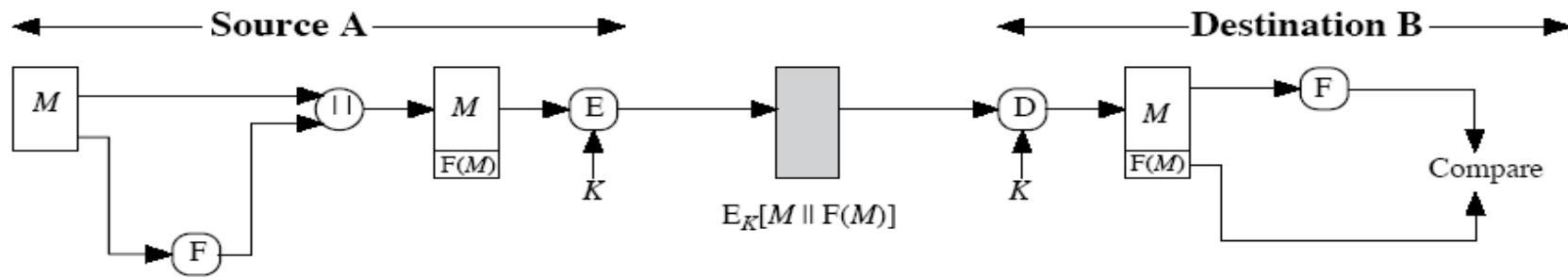
# Authentication Functions

- Two levels
  - Use a *function* to produce an authenticator
  - Use the function as a primitive in a higher-level authentication protocol
    - ❑ Enable receiver to verify authenticity of a message
- Three alternative functions
  - **Message encryption**
    - ❑ Ciphertext of entire message serves as authenticator
  - **Message authentication code (MAC)**
    - ❑ A public function and a secret key produce a fixed-length value serving as the authenticator
  - **Hash function**
    - ❑ A public function that maps a message of any length into a fixed-length hash value, which serves as the authenticator

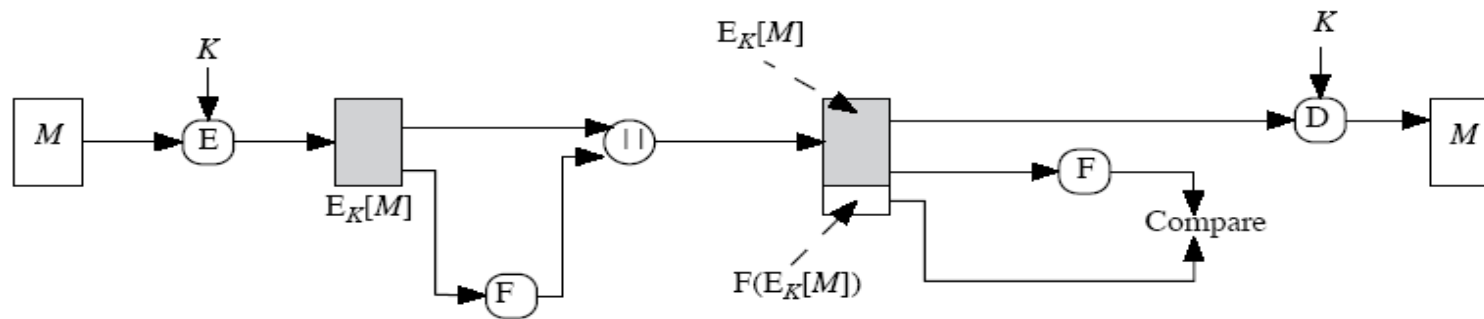
# Message Encryption <sup>1/3</sup>

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

# Message Encryption <sup>2/3</sup>



(a) Internal error control



(b) External error control

**Figure 11.2 Internal and External Error Control**

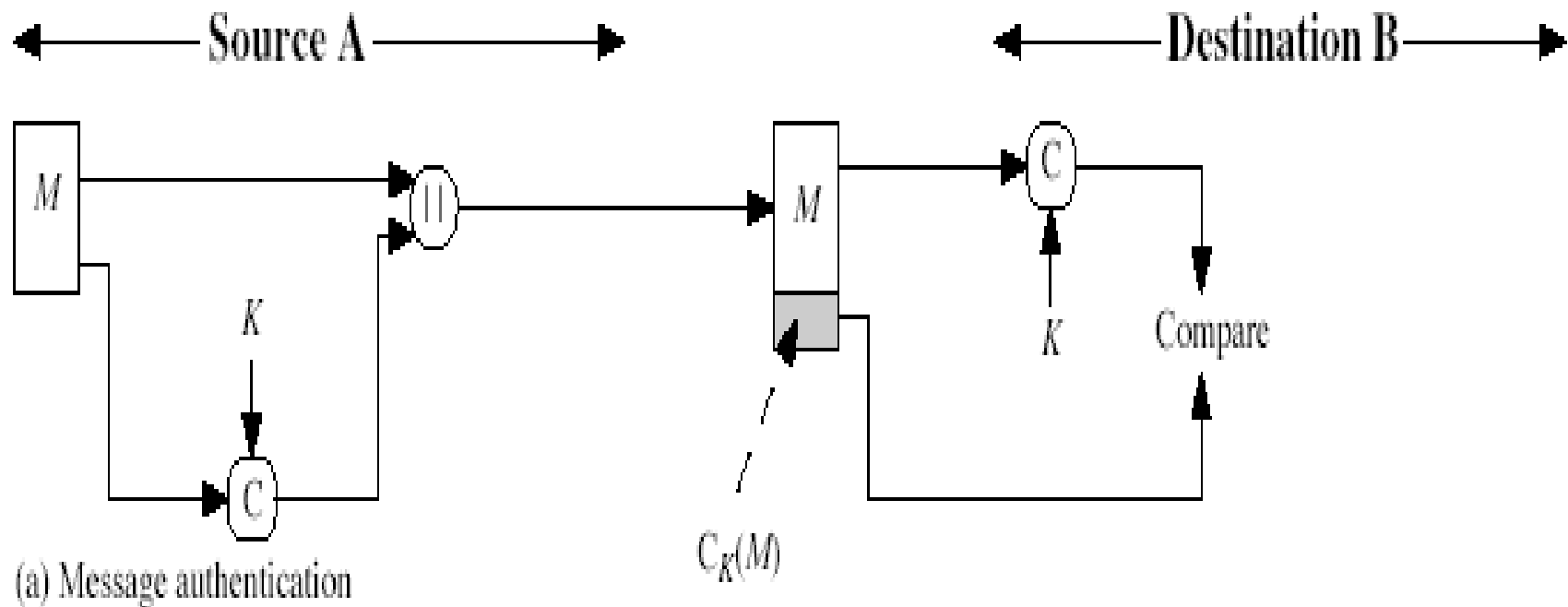
# Message Encryption <sup>3/3</sup>

- 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
- Please see Table 11.1

# 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
  - If message includes a sequence number → receiver assured of proper sequence

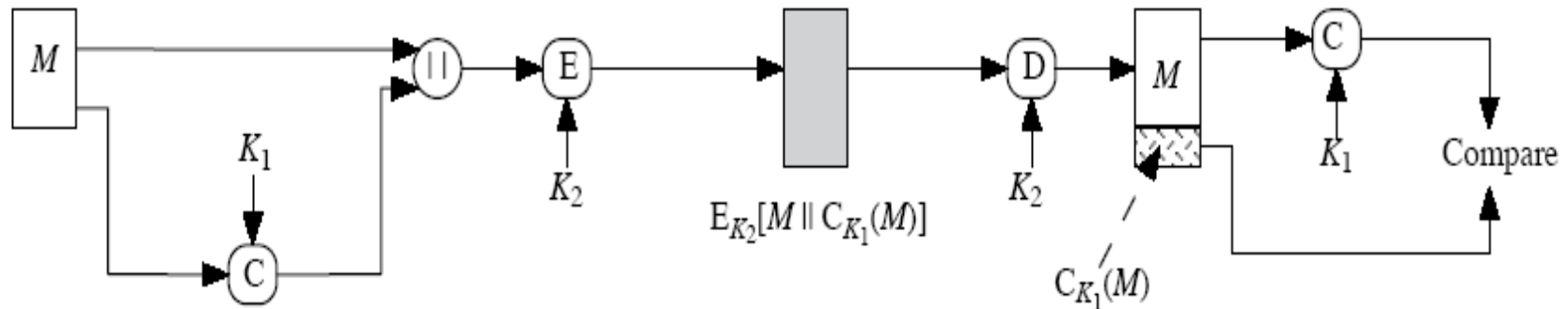
# MAC Use for Authentication



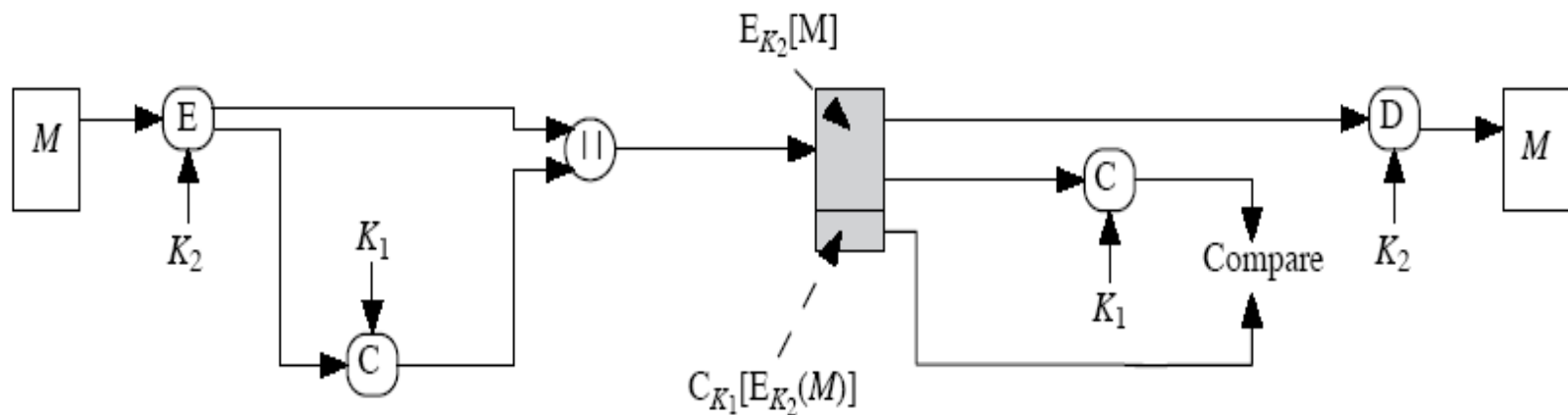
# MAC use for Confidentiality <sup>1/2</sup>

- 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 (e.g. archival use)
- note that a MAC is not a digital signature

# MAC use for Confidentiality <sup>2/2</sup>



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



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

**Figure 11.4 Basic Uses of Message Authentication Code (MAC)**

# MAC Properties

- a MAC is a cryptographic checksum

$$\text{MAC} = C_K(M)$$

- 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
      - ❑ Assume 100-bit messages, and a 10-bit MAC
      - ❑ On average, each MAC is generated by a total of  $2^{90}$  messages

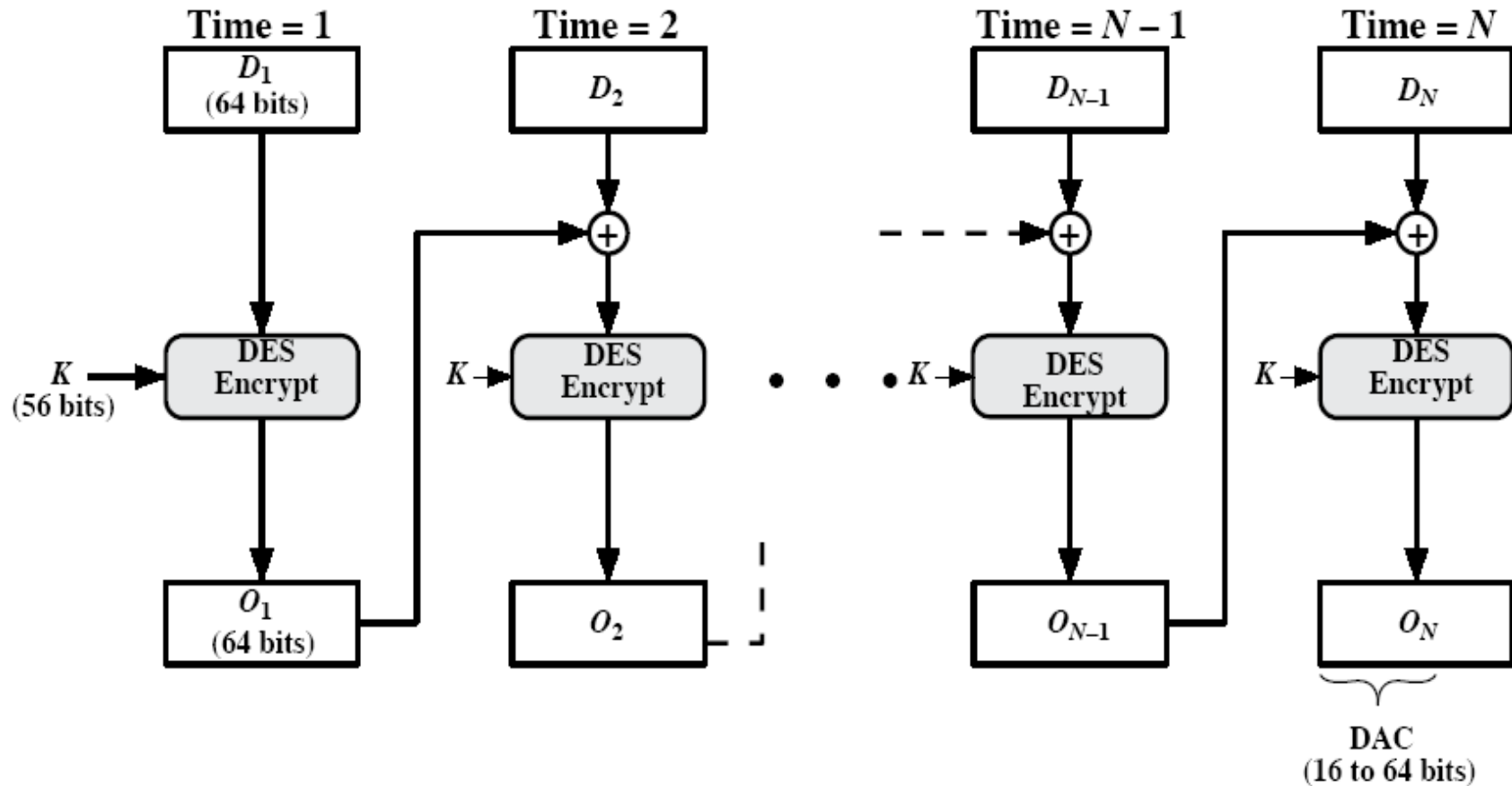
# Requirements for MACs

- 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

# Using Symmetric Ciphers for MACs <sup>1/2</sup>

- 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

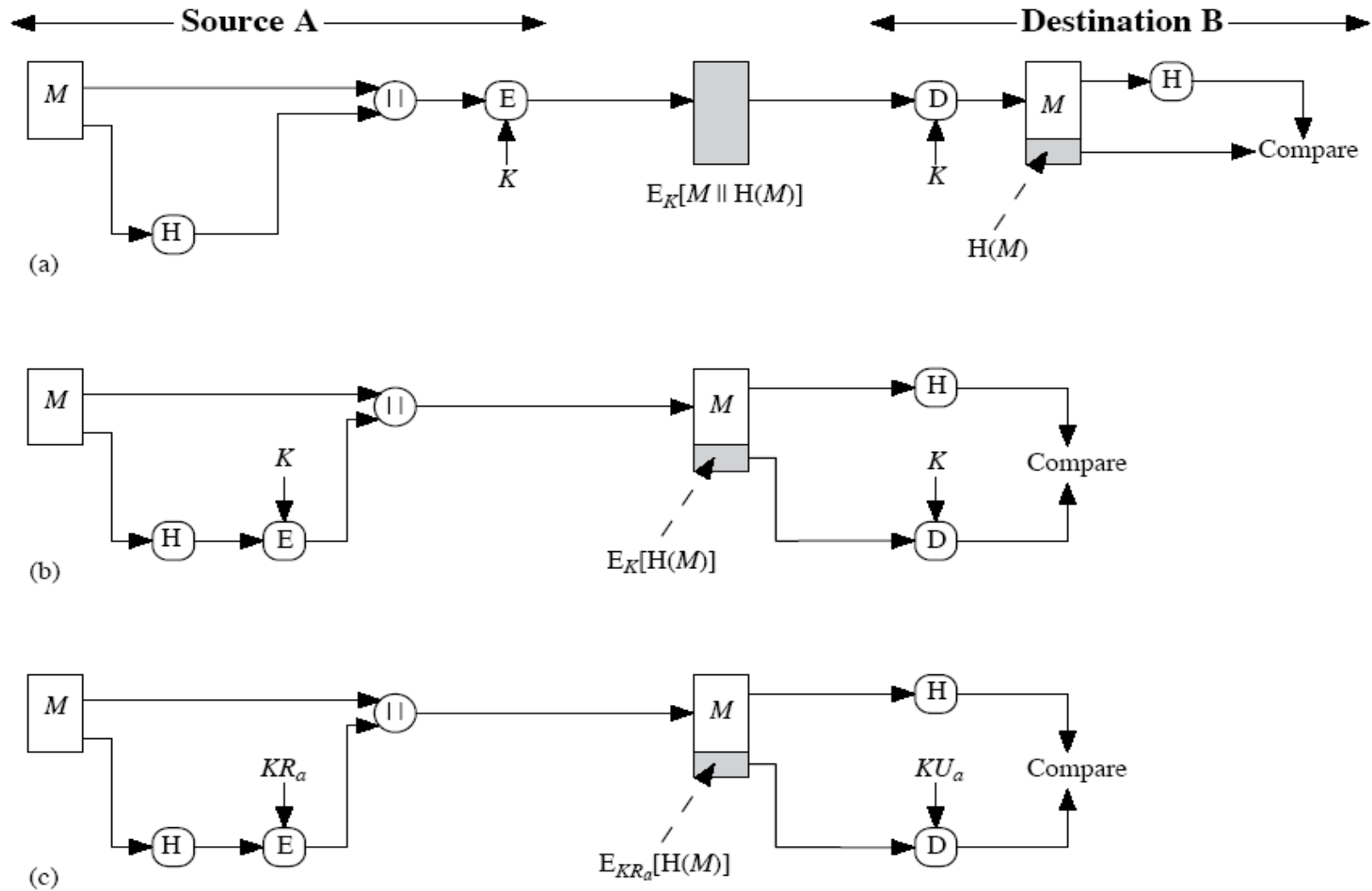
# Using Symmetric Ciphers for MACs <sup>2/2</sup>



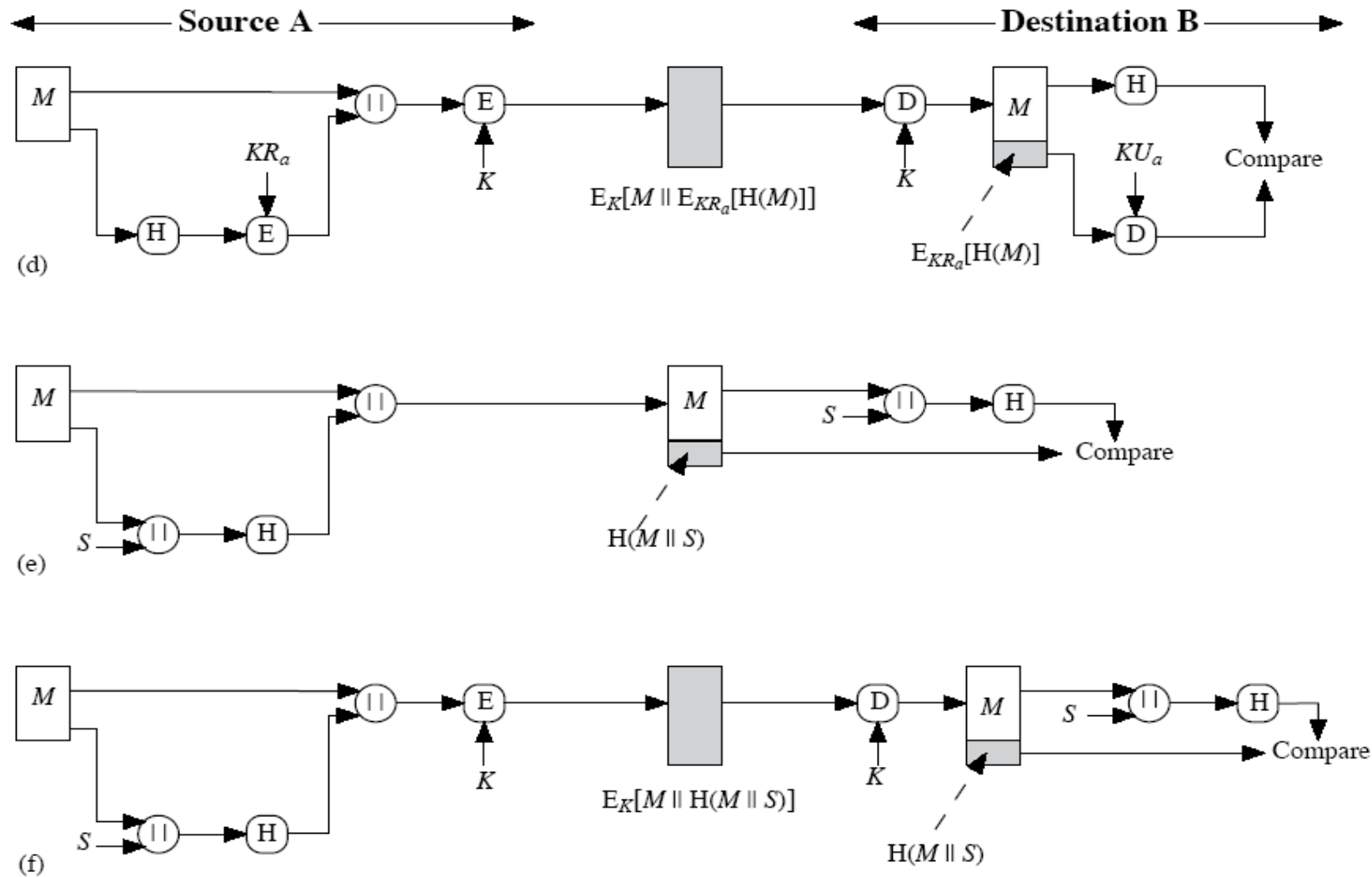
# Hash Functions

- condenses arbitrary message to fixed size
  - $H(M)$  = message digest = hash value
- usually assume that the hash function is public and not keyed
- hash used to detect changes to message
- can use in various ways with message
- most often to create a digital signature

# Basic Uses of Hash Functions <sup>1/2</sup>



# Basic Uses of Hash Functions <sup>2/2</sup>



# Requirements for Hash Functions

1. can be applied to any sized message  $M$
2. produces fixed-length output  $h$
3. is easy to compute  $h=H(M)$  for any message  $M$
4. given  $h$  is infeasible to find  $x$  s.t.  $H(x)=h$ 
  - *one-way property*
5. given  $x$  is infeasible to find  $y$  s.t.  $H(y)=H(x)$ 
  - *weak collision resistance*
6. is infeasible to find any  $x, y$  s.t.  $H(y)=H(x)$ 
  - *strong collision resistance*

# Simple Hash Functions

- Several proposals for simple functions
- based on XOR of message blocks (bit-by-bit XOR of every block)
- not secure since can manipulate any message and either not change hash or change hash also
  - Given a message, produce a new message that yields the hash code
  - Prepare the desired alternate message and then append an  $n$ -bit block to force the new message + block to yield desired hash code
- need a stronger cryptographic function
- Need to use a substantial number of bits a hash size

# Block Ciphers as Hash Functions

- can use block ciphers as hash functions
  - using  $H_0$ =initial value and zero-pad of final block
  - compute:  $H_i = E_{M_i} [H_{i-1}]$
  - and use final block as the hash value
  - similar to CBC but without a key
- resulting hash is too small (64-bit)

# Hash Functions & MAC Security

- like block ciphers have:
- **brute-force** attacks exploiting
  - strong collision resistance hash
    - proposal for h/w MD5 cracker
    - 128-bit hash looks vulnerable, 160-bits better
  - MACs with known message-MAC pairs  $(x_i, C_k(x_i))$ 
    - can either attack key-space (key search) or MAC
    - at least 128-bit MAC is needed for security

# Hash Functions & MAC Security

- **cryptanalytic attacks** exploit structure
  - like block ciphers want brute-force attacks to be the best alternative
- have a number of analytic attacks on iterated hash functions
  - $CV_0=IV; CV_i = f[CV_{i-1}, M_i]; H(M)=CV_N$
  - typically focus on collisions in function  $f$  (compression function)
  - like block ciphers is often composed of rounds
  - attacks exploit properties of round functions