

Introduction to Cryptography

Lecture 8: Message Authentication Codes (MAC)

CYB 650

Introduction to Cryptography

Presented by

Dr. Mohammad Alkhatib

- Introduction
- Message Security Requirements
- Message Authentication Code (MAC)
- MAC Properties and Requirements
- Security of MAC
- Keyed Hash Functions as MAC.
- HMAC Overview

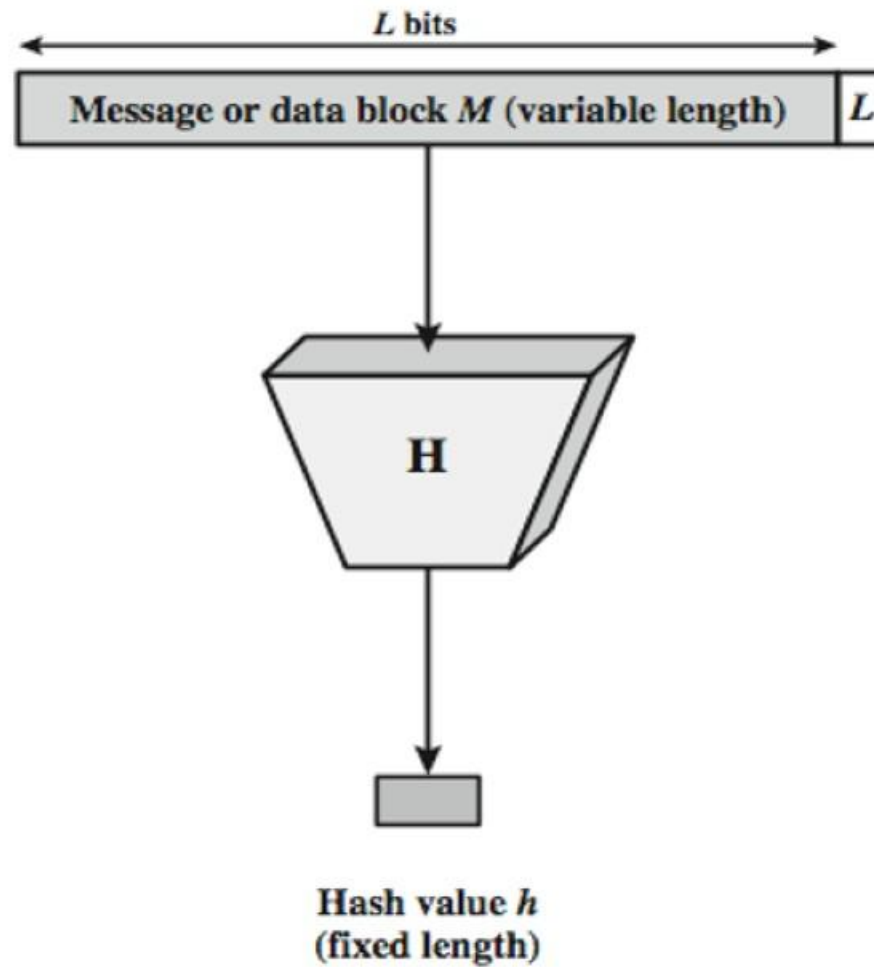
Introduction(Hash Functions)

- Accepts a variable-length block of data as input and produces a fixed-size hash value

$$h = H(M)$$

- Hash function is public
- Detect changes to message
- We want a **cryptographic** hash function
 - ▣ Computationally infeasible to find data mapping to specific hash (one-way property)
 - ▣ Computationally infeasible to find two data to same hash (collision-free property)

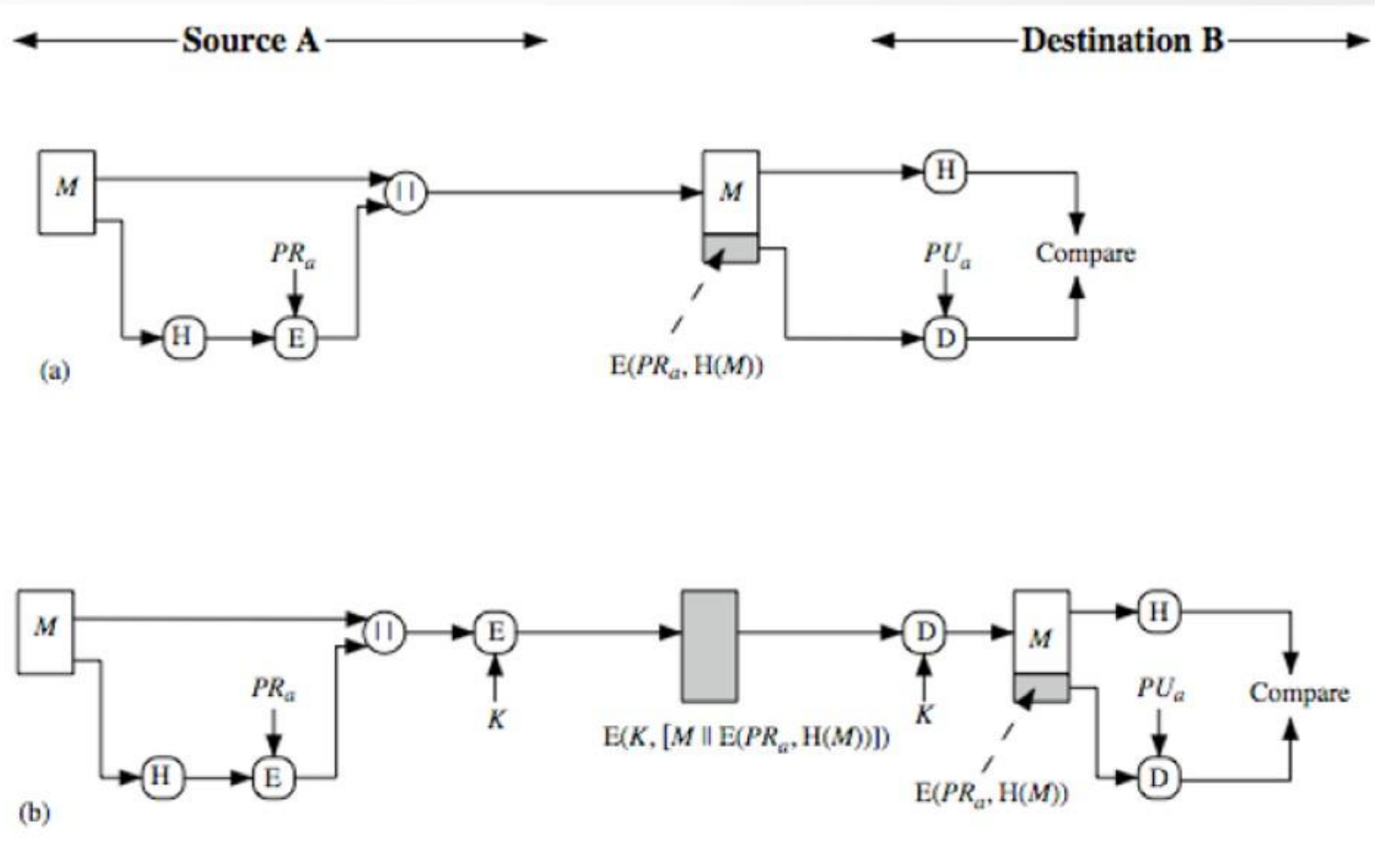
Introduction(Hash Functions)



- ☐ Message Authentication
 - ☐ Verify the integrity of a message (**message digest**)
- ☐ Digital Signatures
 - ☐ Hash value of a message is encrypted with a user's private key
- ☐ One-way password file
 - ☐ Store hash of password not actual password
- ☐ Intrusion detection and virus detection
 - ☐ Keep & check hash of files on system
- ☐ Pseudorandom function (PRF) or pseudorandom number generator (PRNG)

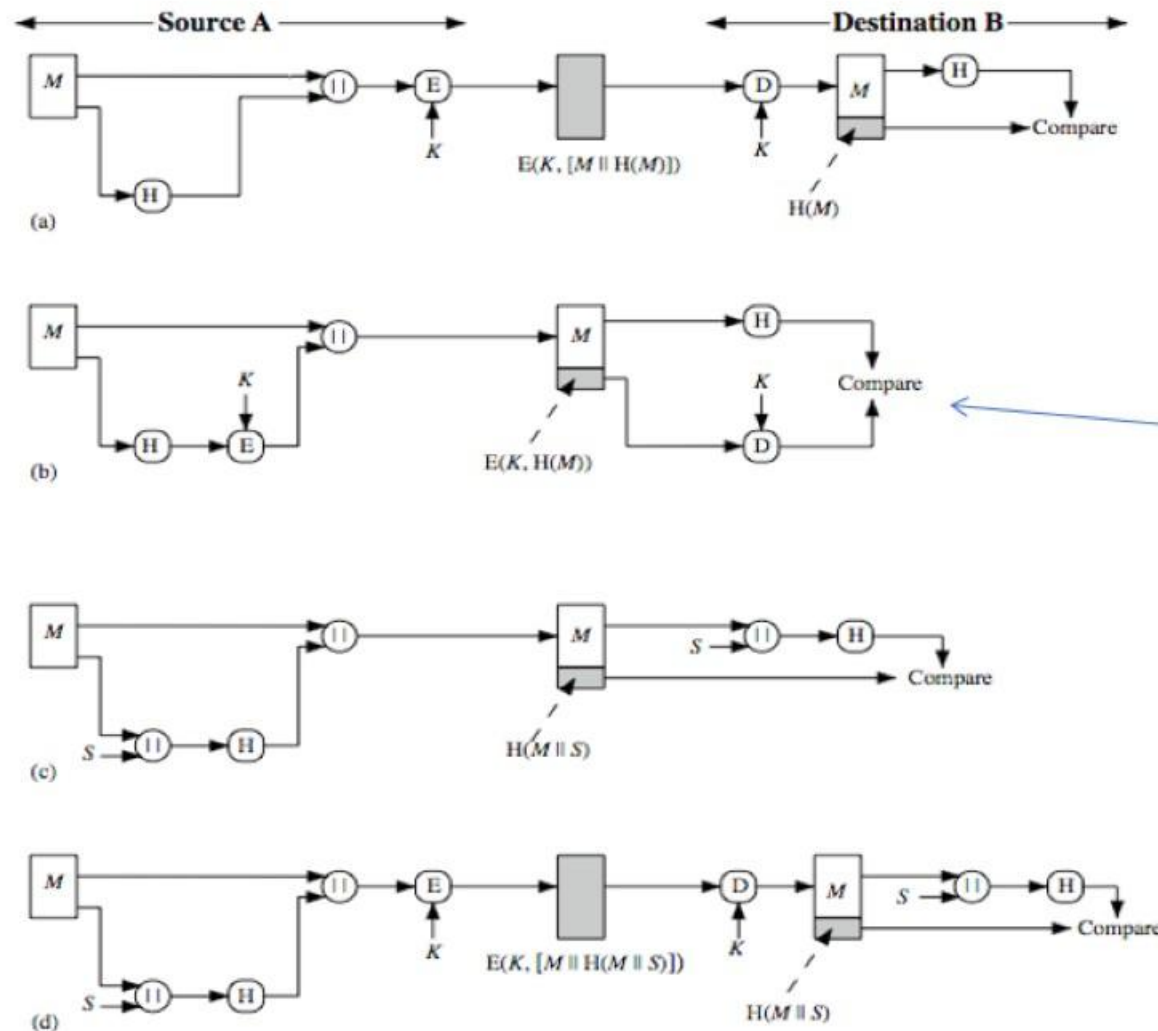
Module 3

Introduction (Hash Functions Applications- Digital Signature)



Module 3

Introduction(Hash Functions Applications- Message Authentication)



**Message
Authentication
Code (MAC) or
keyed hash
function**

- ❑ Message authentication is concerned with:
 - ❑ Protecting the integrity of a message
 - ❑ Validating identity of originator
 - ❑ Non-repudiation of origin (dispute resolution)

- ❑ Three alternative functions can be used:
 - ❑ Hash function
 - ❑ Message encryption
 - ❑ Message authentication code (MAC)

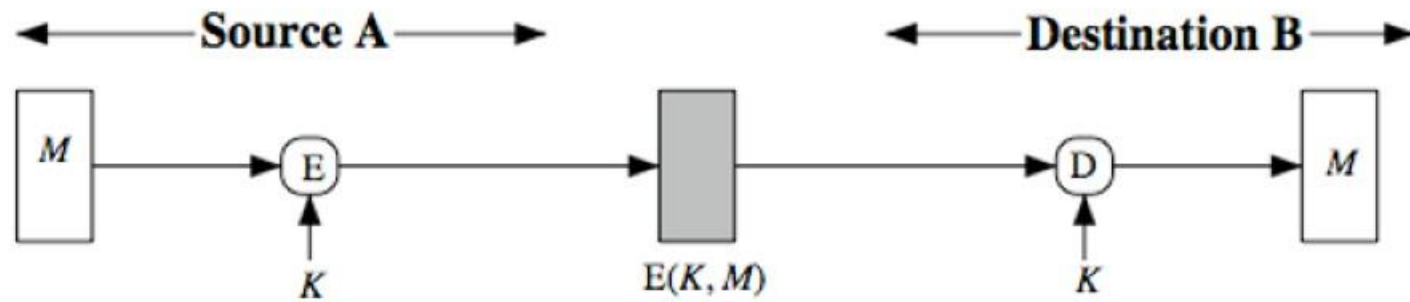
Examples of message security requirements:

- ☐ Disclosure
- ☐ Masquerade
- ☐ Content modification
- ☐ Source repudiation

- ❑ Encryption can also provides authentication
 - ❑ If symmetric encryption is used then:
 - ❑ Receiver know sender must have created it
 - ❑ Since only sender and receiver now key used
 - ❑ Know content cannot of been altered
 - ❑ If message has suitable structure, redundancy or a checksum to detect any changes

Module 03

Symmetric Message Encryption

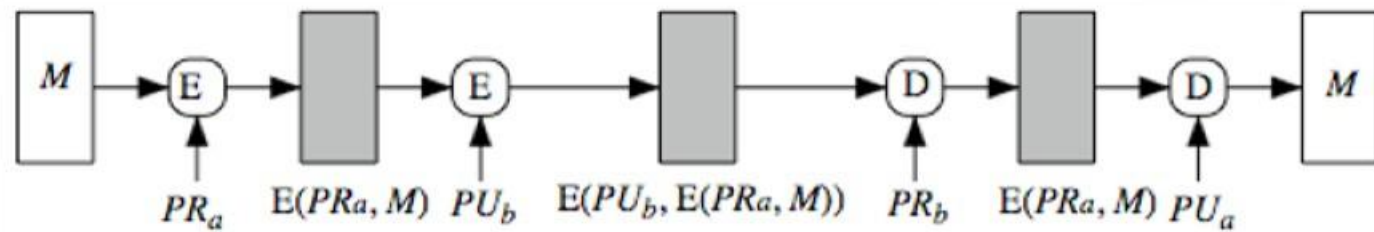


(a) Symmetric encryption: confidentiality and authentication

- ❑ 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

Module 03

Public-Key Message Encryption



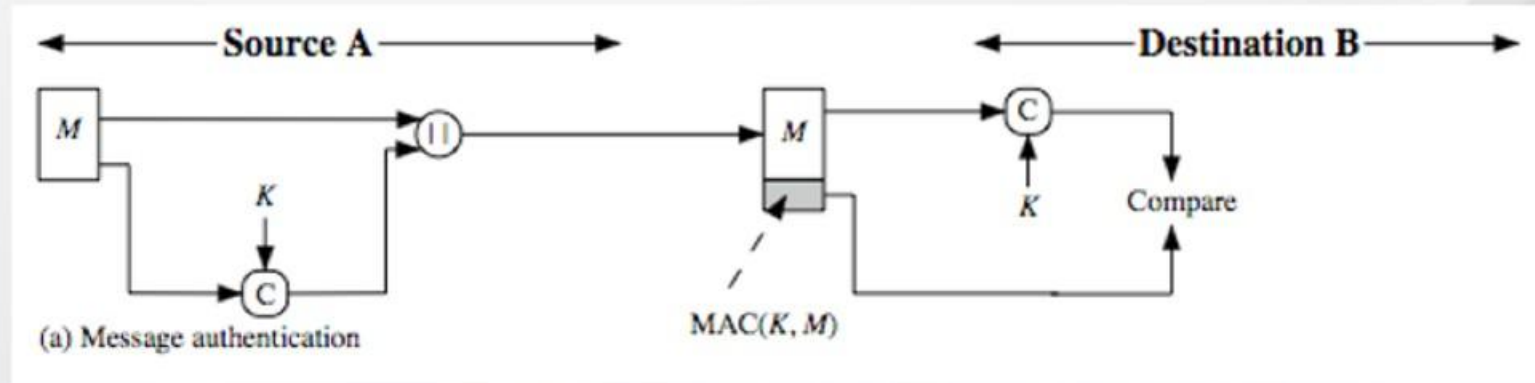
(d) Public-key encryption: confidentiality, authentication, and signature

- ❑ 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

Module 03

Message Authentication Code (MAC)

- ❑ A small fixed-sized block of data
 - ❑ Generated from message + secret key
 - ❑ $MAC = C(K, M)$
 - ❑ Appended to message when sent



- ❑ MAC can also use encryption for secrecy
 - ❑ Generally use separate keys for each
 - ❑ MAC used either before or after encryption
- ❑ Why use a MAC?
 - ❑ Authentication is needed
 - ❑ Authentication to persist longer than the encryption (eg. archival use)
- ❑ MAC is not a digital signature

- ❑ MAC is a cryptographic checksum
 - ❑ $MAC = CK(M)$
 - ❑ Shrinks a variable-length message M
 - ❑ Using a secret key K
 - ❑ Fixed-sized authenticator
- ❑ MAC many-to-one function
 - ❑ Potentially many messages have same MAC
 - ❑ But finding these needs to be very difficult

- ❑ MAC need to satisfy the following:
 - ❑ Knowing a message and MAC, is infeasible to find another message with same MAC
 - ❑ MACs should be uniformly distributed
 - ❑ MAC should depend equally on all bits of the message

- ❑ MAC based on a hash function
 - ❑ Hash functions are generally faster
- ❑ Hash includes a key along with message
- ❑ Original proposal:
 - ❑ $\text{KeyedHash} = \text{Hash}(\text{Key}|\text{Message})$
 - ❑ Some weaknesses were found with this
- ❑ Eventually led to development of HMAC

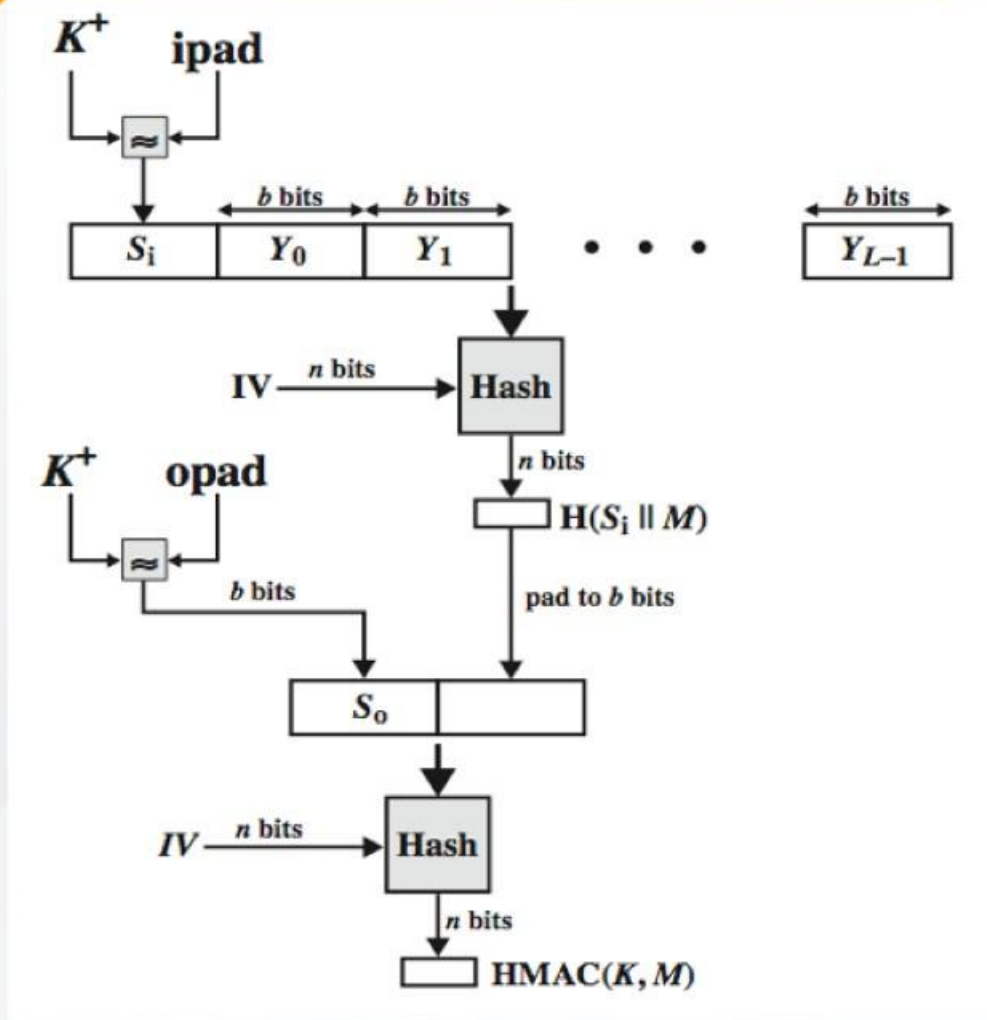
- ❑ Allow for easy replaceability of embedded hash function
- ❑ Preserve original performance of hash function without significant degradation
- ❑ Use and handle keys in a simple way.
- ❑ Have well understood cryptographic analysis of authentication mechanism strength

- ❑ Specified as Internet standard RFC2104
- ❑ Uses hash function on the message:
$$\text{HMACK}(M) = \text{Hash}[(K+ \text{ XOR opad}) \parallel \text{Hash}[(K+ \text{ XOR ipad}) \parallel M]]$$

where $K+$ is the key padded out to size
opad, ipad are specified padding constants
- ❑ Overhead is just 1 more hash calculations than the message needs alone
- ❑ Any hash function can be used
 - ▣ eg. MD5, SHA-1, RIPEMD-160, Whirlpool

Module 03

HMAC Overview



- ❑ 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



Thank You