Chapter 11: Message Authentication and Hash Functions

Fourth Edition
by William Stallings

Lecture slides by Lawrie Brown
(modified by Prof. M. Singhal, U of Kentucky)

Message Authentication

- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)

- three alternative functions used:
  - message encryption
  - message authentication code (MAC)
  - hash function

Broader Set of Attacks

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

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 now key used
  - know content cannot of been altered
  - Provides both: sender authentication and message authenticity.

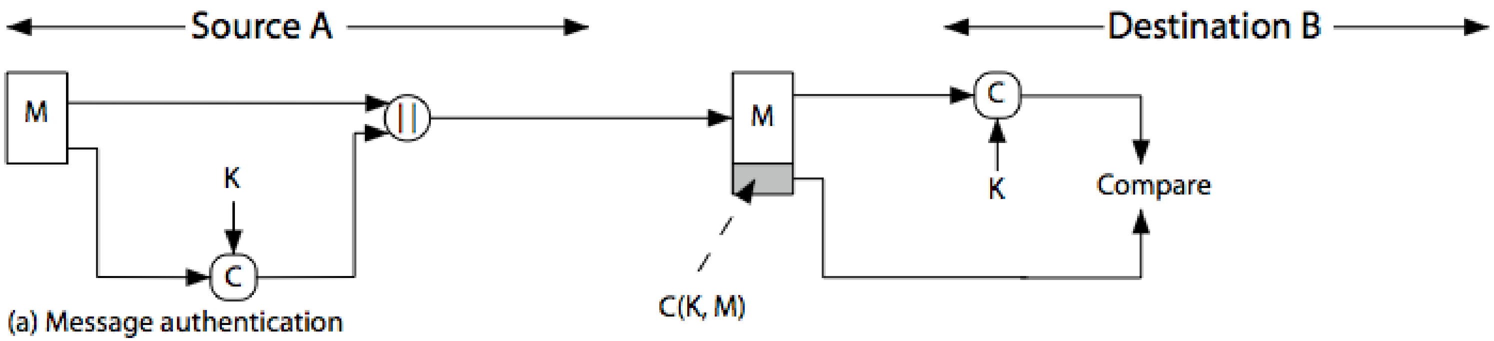
Message Encryption

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

Message Authentication Code (MAC)

- a small fixed-sized block of data:
  - depends on both message and a secret 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



Message Authentication Codes

- MAC provides authentication
- Message can be encrypted 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 Properties

- a MAC is a cryptographic checksum

  $MAC = C_K(M)$

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

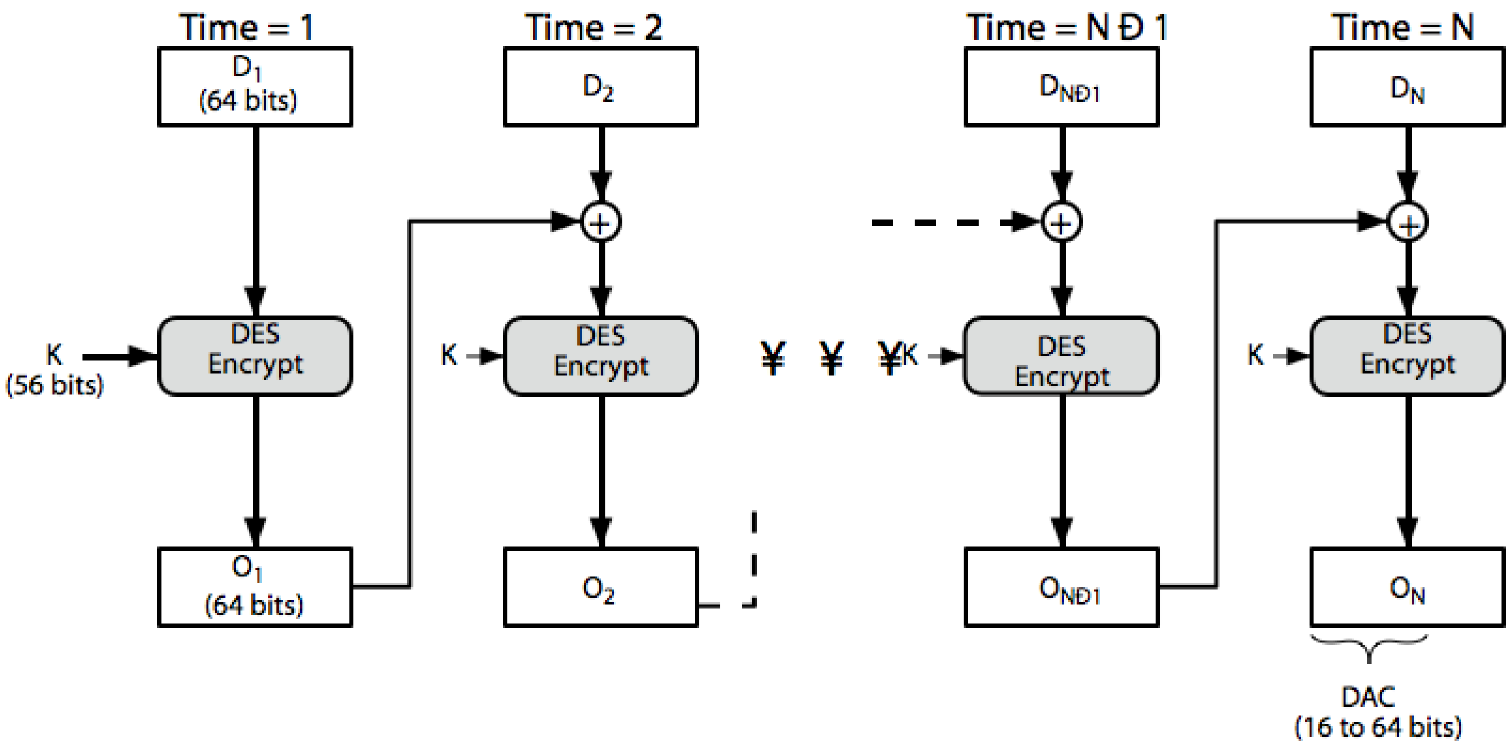
Requirements for MACs

- MAC needs 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

- 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

Data Authentication Algorithm



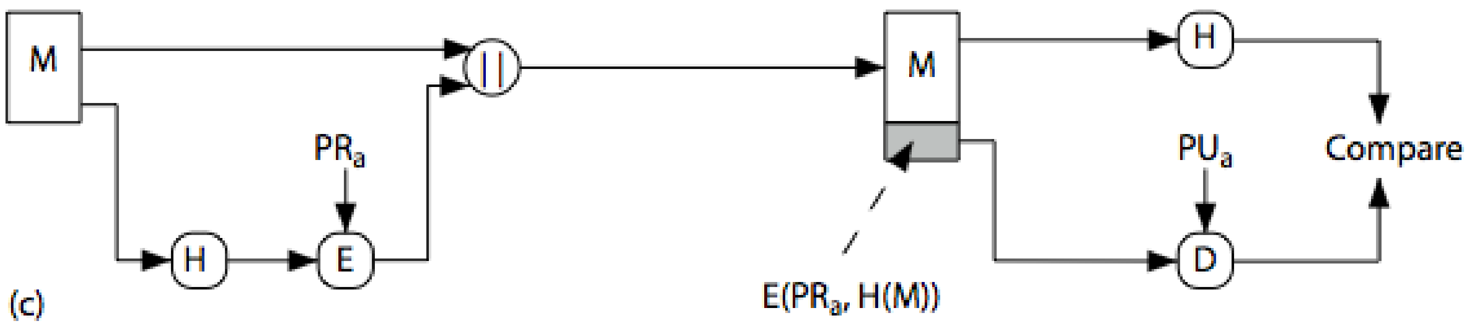
Hash Functions

- A hash function is like a MAC
- condenses arbitrary message to fixed size

  `h = H(M)`
- usually assume that the hash function is public and not keyed

  -note that a MAC is keyed
- hash used to detect changes to message
- can use in various ways with message
- most often to create a digital signature

Hash Functions & Digital Signatures



Requirements for Hash Functions

1. can be applied to any size message `M`
2. produces a 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

- are several proposals for simple functions
- based on XOR of message blocks

  -divide the message into equal size blocks

  -perform XOR operation block by block

  -final output is the hash
- not very secure
- need a stronger cryptographic function (next chapter)

Block Ciphers as Hash Functions

- can use block ciphers as hash functions
  - using H_0=0 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)
  - Vulnerable to attacks

Summary

- have considered:
  - message authentication using
  - message encryption
  - MACs
  - hash functions
  - basic design approach