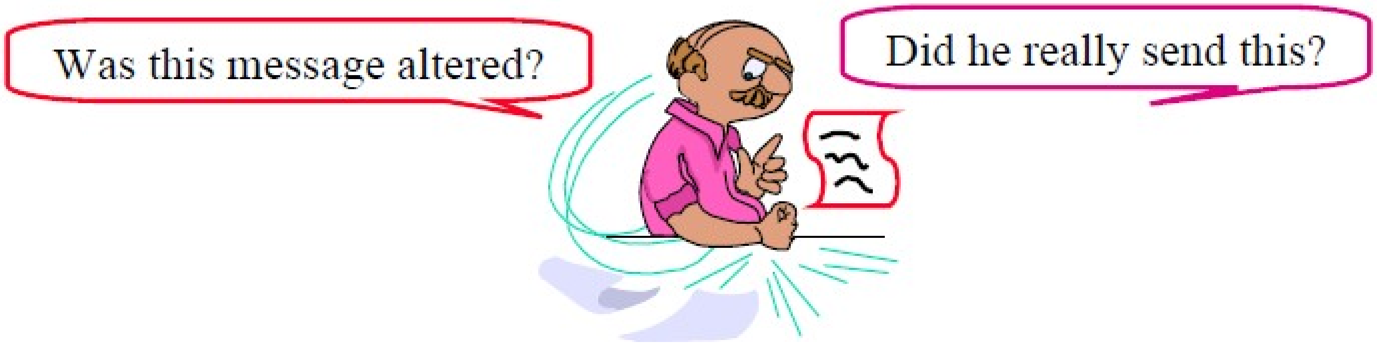
Message Authentication

Dr. V. K. Pachghare
vkp.comp@coep.ac.in

Message Authentication Code



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

Security Requirements

- 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. The number and length of messages between parties could be determined.
- Masquerade – Insertion of messages into the network from a fraudulent source. Creation of messages by an opponent, fraudulent acknowledgment of message receipt by someone other than the message recipient.

- Content modification – Changes to the contents of a message, including insertion, deletion, transposition, and modification.
- Sequence modification – Modification to a sequence of messages between parties, insertion, deletion, re-ording.
- 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

Authentication Functions

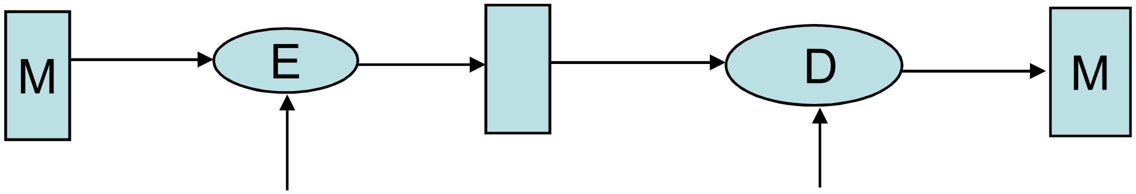
- Message encryption – The ciphertext of the entire message serves as its authenticator
- Message authentication code (MAC) – A public function of the message and a secret key that produces a fixed-length value that serves as the authenticator
- Hash function – A public function that maps a message of any length into a fixed-length hash value, which serves as the authenticator.

Message Encryption

- Message encryption by itself also provides a measure of authentication

    1. Symmetric encryption
    2. Public-key encryption

- if symmetric encryption is used then:
- Source A Destination B



- K Ek(M) K

– receiver knows sender must have created the message

– since only sender and receiver know the key used

– content cannot of been altered

– if message has suitable structure, redundancy or a checksum to detect any changes

Message Encryption: public-key

- encryption provides no confidence of sender
- since anyone potentially knows public-key

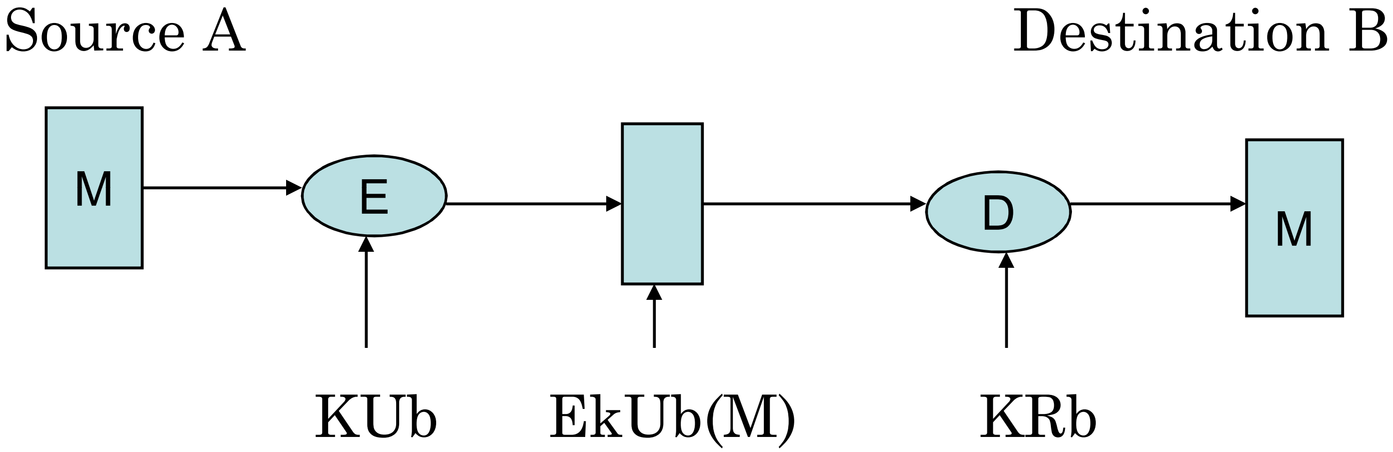


Fig: Public-key encryption: **Confidentiality**

Authentication and signature

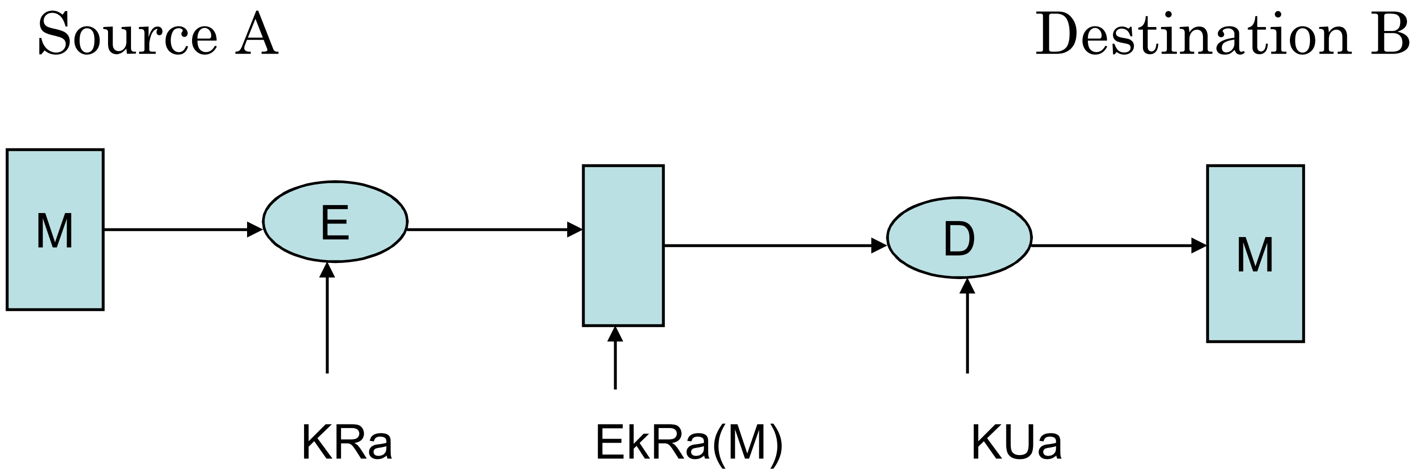
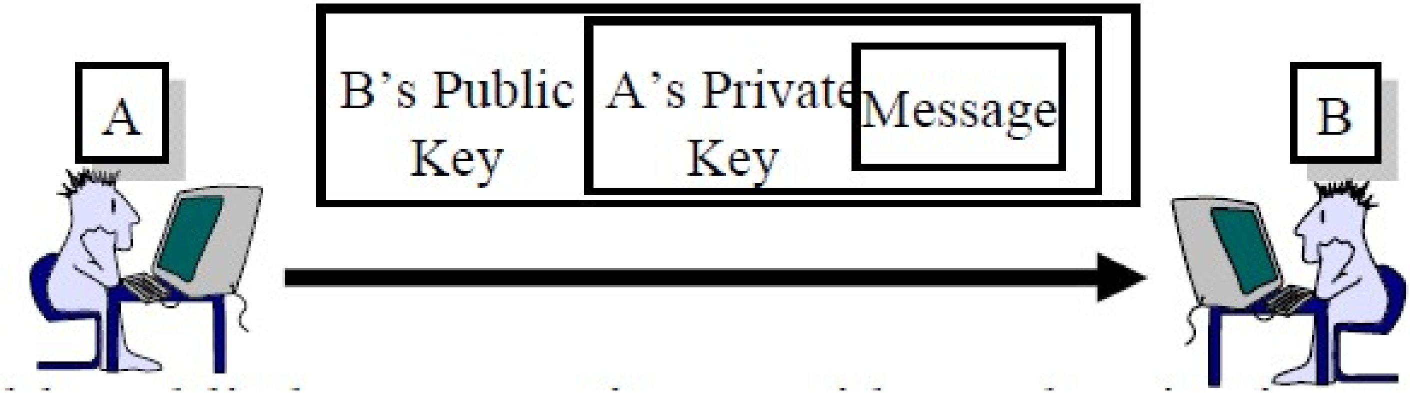


Fig: Public-key encryption: authentication and signature

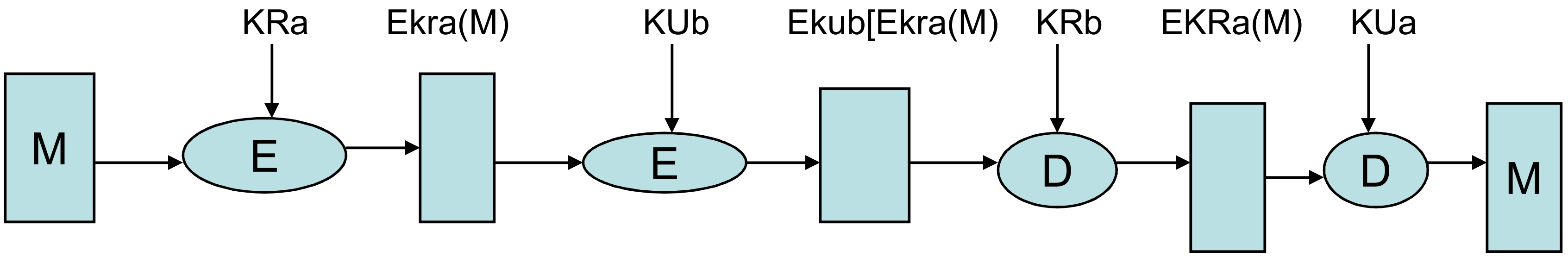
Public--Key Authentication and Secrecy



Double public key encryption provides authentication and integrity.

Double public key => Very compute intensive

PUBLIC-KEY ENCRYPTION CONFIDENTIALITY, AUTHENTICATION, AND SIGNATURE



- Figure: Public-key Encryption Confidentiality, Authentication, and Signature

- 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

- Crypto checksum (MAC) is better.
- Based on a secret key and the message.
- Can also encrypt with the same or different key

Message Authentication Code (MAC)

MAC

- MAC algorithm is a symmetric key cryptographic technique to provide message authentication.
- For establishing MAC process, the sender and receiver share a symmetric key K.
- Essentially, a MAC is an encrypted checksum generated on the underlying message that is sent along with a message to ensure message authentication.

Message Authentication Code (MAC)

- generated by an algorithm that creates a small fixed-sized block of data, known as checksum or MAC

$$\mathbf{MAC = C_k(M)}$$

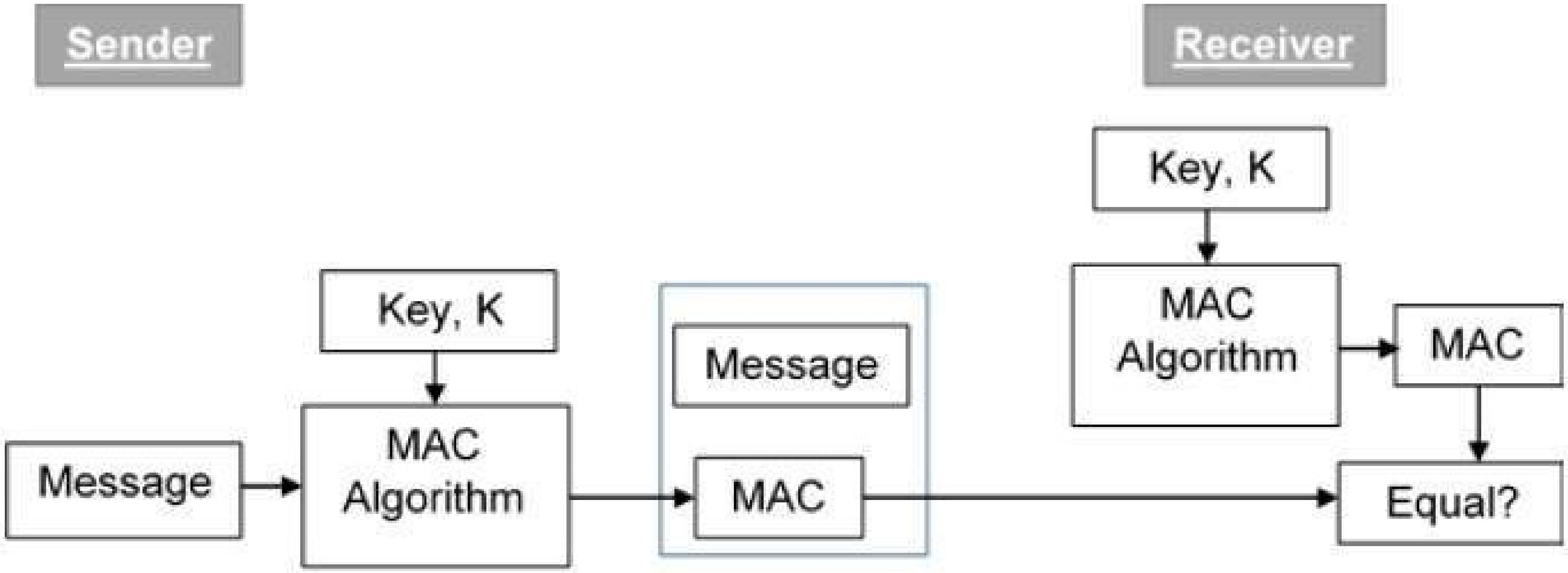
Where M -> Input Message

C -> MAC function

K -> Shared secret key

MAC -> message authentication code

MAC using Symmetric Key



- The sender uses some publicly known MAC algorithm, inputs the message and the secret key K and produces a MAC value.
- Similar to hash, MAC function also compresses an arbitrary long input into a fixed length output. The major difference between hash and MAC is that MAC uses secret key during the compression.

- The sender forwards the message along with the MAC. Here, we assume that the message is sent in the clear, as we are concerned of providing message origin authentication, not confidentiality. If confidentiality is required then the message needs encryption.
- On receipt of the message and the MAC, the receiver feeds the received message and the shared secret key K into the MAC algorithm and re-computes the MAC value.

- The receiver now checks equality of freshly computed MAC with the MAC received from the sender. If they match, then the receiver accepts the message and assures himself that the message has been sent by the intended sender.
- If the computed MAC does not match the MAC sent by the sender, the receiver cannot determine whether it is the message that has been altered or it is the origin that has been falsified. As a bottom-line, a receiver safely assumes that the message is not the genuine.

Advantages to encryption

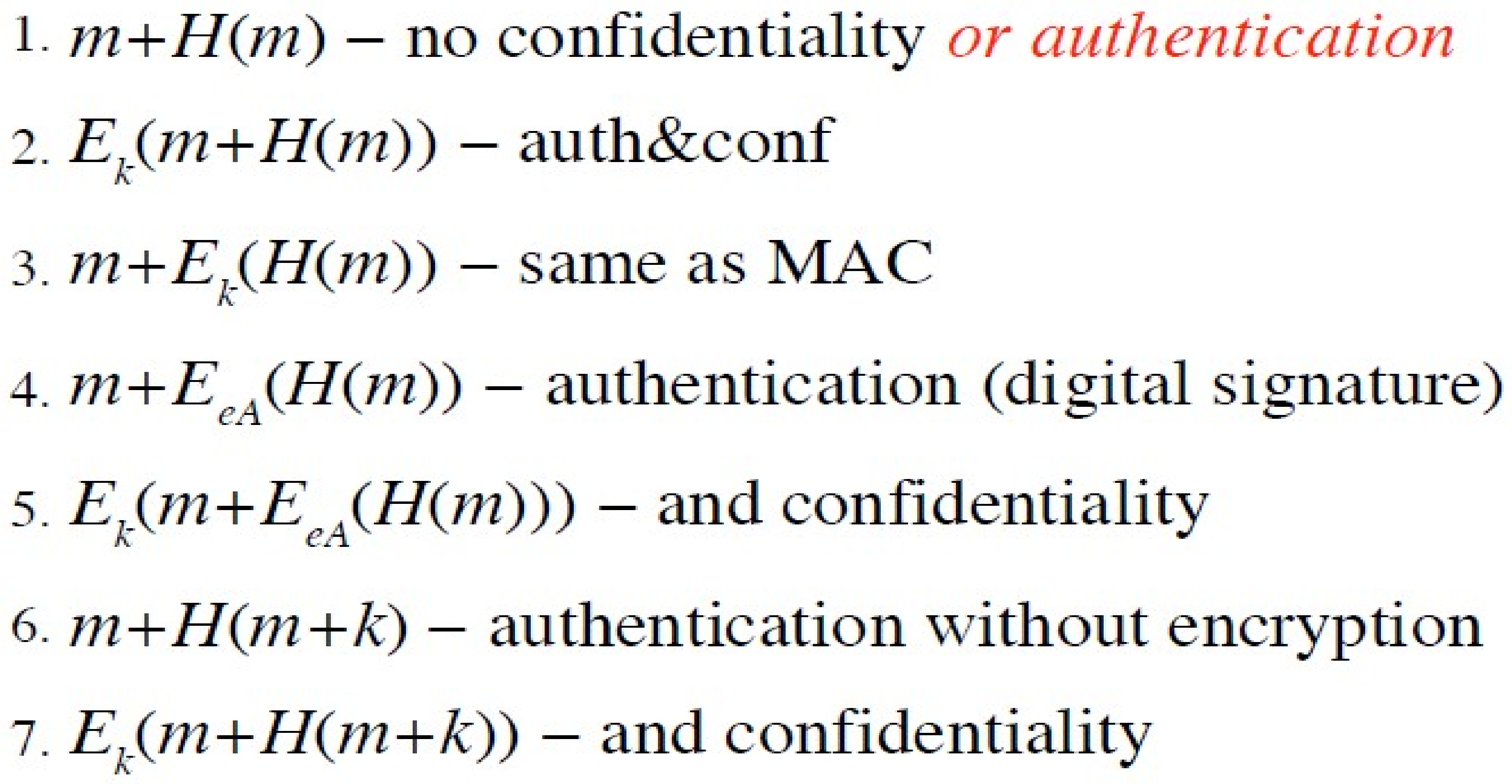
- Faster
- Broadcast message can be checked at only one place
- Random tests possible
- MAC can be kept and checked again any number of times
- Can give authentication without confidentiality

Hash Functions

- Condenses arbitrary message to fixed size
- Hash code may be referred as a MD or hash value
- The hash code is a function of all the bits of the massage and provides an error-detection capability
- A change to any bit or bits in the message results in a change to the hash code.
- Usually assume that the hash function is public and not key

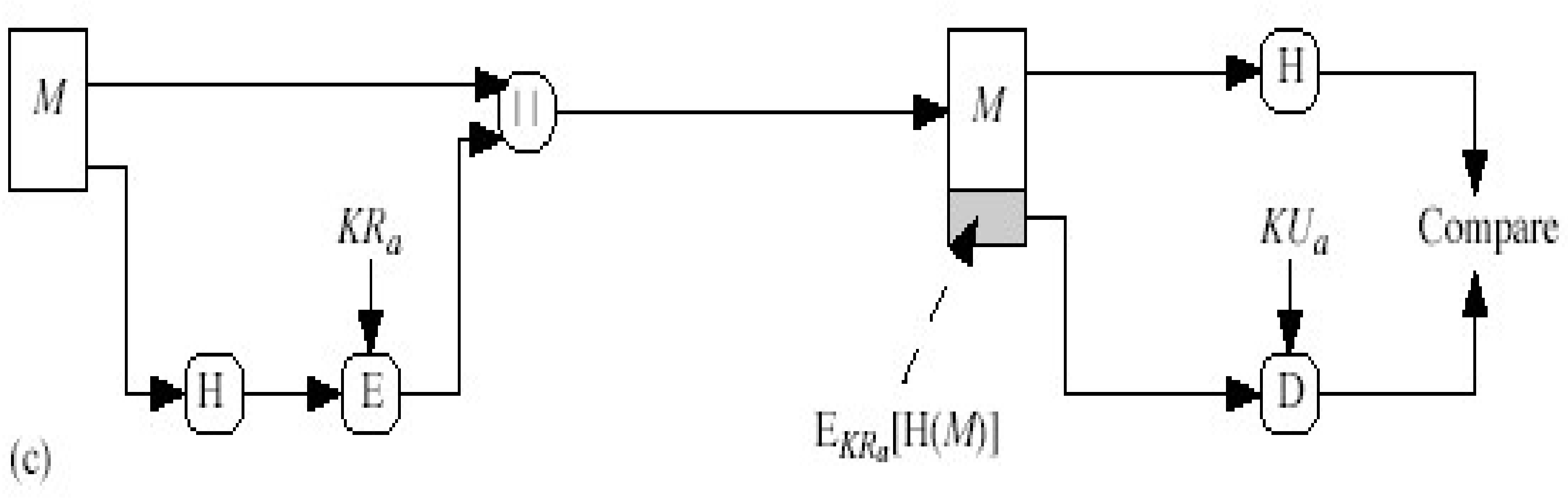
- hash used to detect changes to message
- can use in various ways with message
- most often to create a digital signature

Hash usage



1. $m+H(m)$ – no confidentiality *or authentication*
2. $E_k(m+H(m))$ – auth&conf
3. $m+E_k(H(m))$ – same as MAC
4. $m+E_{eA}(H(m))$ – authentication (digital signature)
5. $E_k(m+E_{eA}(H(m)))$ – and confidentiality
6. $m+H(m+k)$ – authentication without encryption
7. $E_k(m+H(m+k))$ – and confidentiality

Hash Functions & Digital Signatures



Hash Function Properties

- a Hash Function produces a fingerprint of some file/message/data

  h = H(M)

  - condenses a variable-length message M
  - H(M) is the fixed-length hash value
- assumed to be public

Requirements for Hash Functions

H can be applied to any sized message M

produces fixed-length output h

is easy to compute h=H(M) for any message M

For given h it is infeasible to find x such that H(x)=h

This is referred as one-way property

For given block x, it is infeasible to find y such that H(y)=H(x)

This is referred as weak collision resistance

It is infeasible to find any pair (x, y) such that H(y)=H(x)

This is referred as strong collision resistance

Simple Hash Functions

- are several proposals for simple functions
- based on XOR of message blocks
- not secure since can manipulate any message and either not change hash or change hash also
- need a stronger cryptographic function

Birthday Attacks

- might think a 64-bit hash is secure
- but by **Birthday Paradox** is not
- **birthday attack** works thus:
  - opponent generates $2^{m/2}$ variations of a valid message all with essentially the same meaning
  - opponent also generates $2^{m/2}$ variations of a desired fraudulent message

- two sets of messages are compared to find pair with same hash (probability > 0.5 by birthday paradox)
- have user sign the valid message, then substitute the forgery which will have a valid signature

- conclusion is that need to use larger MACs

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)
  - both due to direct birthday attack
  - and to “meet-in-the-middle” attack
- other variants also susceptible to attack

Hash Functions & MAC Security

- like block ciphers have:
- **brute-force** attacks exploiting
  - strong collision resistance hash have cost $2^{m/2}$
    - have proposal for h/w MD5 cracker
    - 128-bit hash looks vulnerable, 160-bits better
  - MACs with known message-MAC pairs
    - can either attack keyspace (of 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
  - $CV_i = f[CV_{i-1}, M_i]; H(M)=CV_N$
  - typically focus on collisions in function f
  - like block ciphers is often composed of rounds
  - attacks exploit properties of round functions

Thanks!!!

?