Message Digests

- Given a message $M$ of arbitrary length, a hash function $H$ can be applied to $M$ to produce a fixed-length message digest $H(M)$
  - Example hash functions: MD5 (128-bit digest), SHA (160-bit digest)
- Properties of $H$
  - It’s computationally infeasible to find $M$ given $H(M)$
  - Given $M$ and $H(M)$, it’s computationally infeasible to find another message $M'$ such that $H(M) = H(M')$
  - Given $H$, it’s computationally infeasible to find two different input values $M$ and $M'$ such that $H(M) = H(M')$

Digital Signatures

- How to guarantee that a message has not been altered in transit and it was authorized by a specified entity?
- Sender can encrypt the message digest using its private key, the encrypted message digest is called a digital signature
  - $A$ sends $M$ and $DA(H(M))$ to $B$ ($EA/DA$ is $A$’s public/private key pair)
  - $B$ computes $H(M)$ and $EA(DA(H(M)))$; if they match, $B$ can be sure that $A$ sent the message and the message hasn’t been altered
Secure and Unforgeable Data Transmission: Requirements

- We want to ensure the following in digital communication:
  - **Authentication**: receiver wants to be sure of sender’s identity
  - **Confidentiality**: the transmitted information is kept private between sender and receiver
  - **Integrity**: the message could not have been altered in transit
  - **Non-repudiation**: the sender cannot credibly deny having signed the message

Secure and Unforgeable Data Transmission: Implementation

- A signs and encrypts M as follows:
  - Compute the digital signature $DA(H(M))$
  - Generate a symmetric encryption key $K$ (called a session key) to encrypt both $M$ and the digital signature
  - Encrypt $K$ with B’s public key $EB$
  - Send $(K(M), K(A, DA(H(M))), EB(K))$ to B

- B does the following:
  - Recover $K$ from $EB(K)$ using its private key $DB$
  - Use $K$ to recover $M$ from $K(M)$ and recover $A$ and $DA(H(M))$ from $K(A, DA(H(M)))$
  - Use $A$’s public key $EA$ to recover $H(M)$ from $DA(H(M))$
  - Compute $H(M)$ and compare the result to the hash value recovered in the previous step. If they are the same, B can trust that $M$ is unaltered and has indeed been signed by A
A Centralized Approach to Symmetric Key Distribution

• Two communicating parties can agree on a symmetric key using a Key Distribution Center (KDC)
  – KDC shares a symmetric key with each party
  – For communication between A and B, KDC generates a one-time session key
  – KDC uses A and B's keys to communicate the session key to A and B

Symmetric Key Distribution Using a KDC

KDC sends a ticket (i.e., $K_{A,KDC}(K_{A,B})$) to Alice and lets Alice set up a connection to Bob.
Authentication and Symmetric Key Distribution Using a KDC

1. Alice

2. $K_{B,KDC}(R_{B1})$

3. $R_{A1}, A, B, K_{B,KDC}(R_{B1})$


5. $K_{A,B}(R_{A2}), K_{B,KDC}(A, K_{A,B'} R_{B1})$

6. $K_{A,B}(R_{A2}^{-1}, R_{B2})$

7. $K_{A,B}(R_{B2}^{-1})$

8. Bob

$K_{A}^+$ denotes the public key of $A$, $K_{A}^-$ denotes the private key of $A$
Public Key Distribution

• The public key should be distributed in such a way that the receiver can be sure that the key indeed belongs to a claimed entity
• Two approaches to public key distribution
  – Using a trusted server
  – Using public-key certificates

Public Key Distribution Using a Trusted Server

• Store all public keys in a trusted server
• Everyone has server’s encryption key ES

A gets B’s public key EB from S
Public-Key Distribution Using Public-Key Certificates

• An identity is linked to a public key by means of a **certificate**
• A **certification authority (CA)** binds a public key to a particular entity E
  – E provides “proof of identity” to CA
  – CA creates a certificate binding E to its public key
    • The certificate contains E’s public key, E’s identifier, and E’s public key and identifier digitally signed by CA using its private key
• To ascertain that the public key found in the certificate indeed belongs to the identified entity:
  – Use the public key of the CA to decrypt the digital signature in the certificate
  – If the signature matches the (public key, identifier) pair, then the public key indeed belongs to the identified entity

Lifetime of Certificates

• A certificate should be revoked if the private key of the identified entity is compromised
• Ways to revoke a certificate:
  – CA regularly publishes a Certificate Revocation List (CRL)
    • A client has to check the CRL
  – Restrict the lifetime of a certificate
    • The validity of a certificate automatically expires after some time