Key Management; Other Public Key Cryptosystems

Outline

- Key management with asymmetric encryption
- Diffie-Hellman key exchange

Key Management (Public)

- Public-key encryption helps address key distribution problems
- Have two aspects of this:
  - distribution of public keys
  - use of public-key encryption to distribute secret keys

Distribution of Public Keys

- can be considered as using one of:
  - public announcement
  - publicly available directory
  - public-key authority
  - public-key certificates

Public Announcement

- Users distribute public keys to recipients or broadcast to community at large
  - eg. append PGP keys to email messages or post to newsgroups or email list
- Major weakness is forgery
  - anyone can create a key claiming to be someone else and broadcast it
  - until forgery is discovered can masquerade as claimed user

Key Management (cont'd)

- Public announcement

Uncontrolled Public Key Distribution

Publicly Available Directory

- Can obtain greater security by **registering** keys with a public directory
- Directory must be trusted with properties:
  - contains {name, public-key} entries
  - participants register securely with directory
  - participants can replace key at any time
  - directory is periodically published
  - directory can be accessed electronically
    - Weakness: directory must be trusted and still vulnerable to forgery

PUBLIC-KEY PUBLICATION

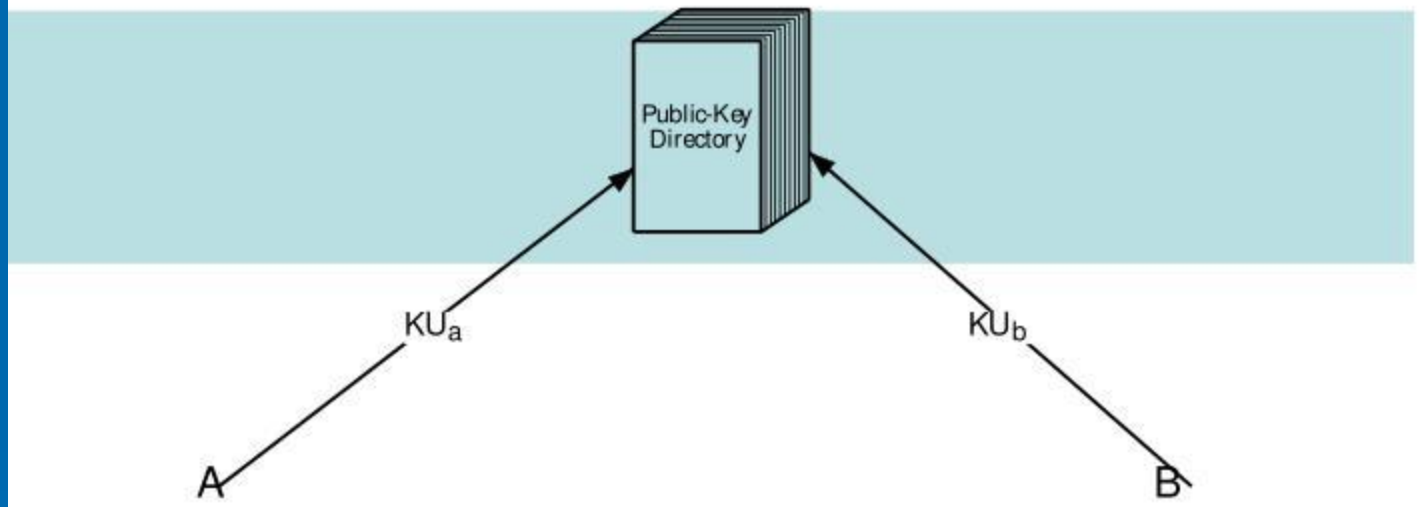
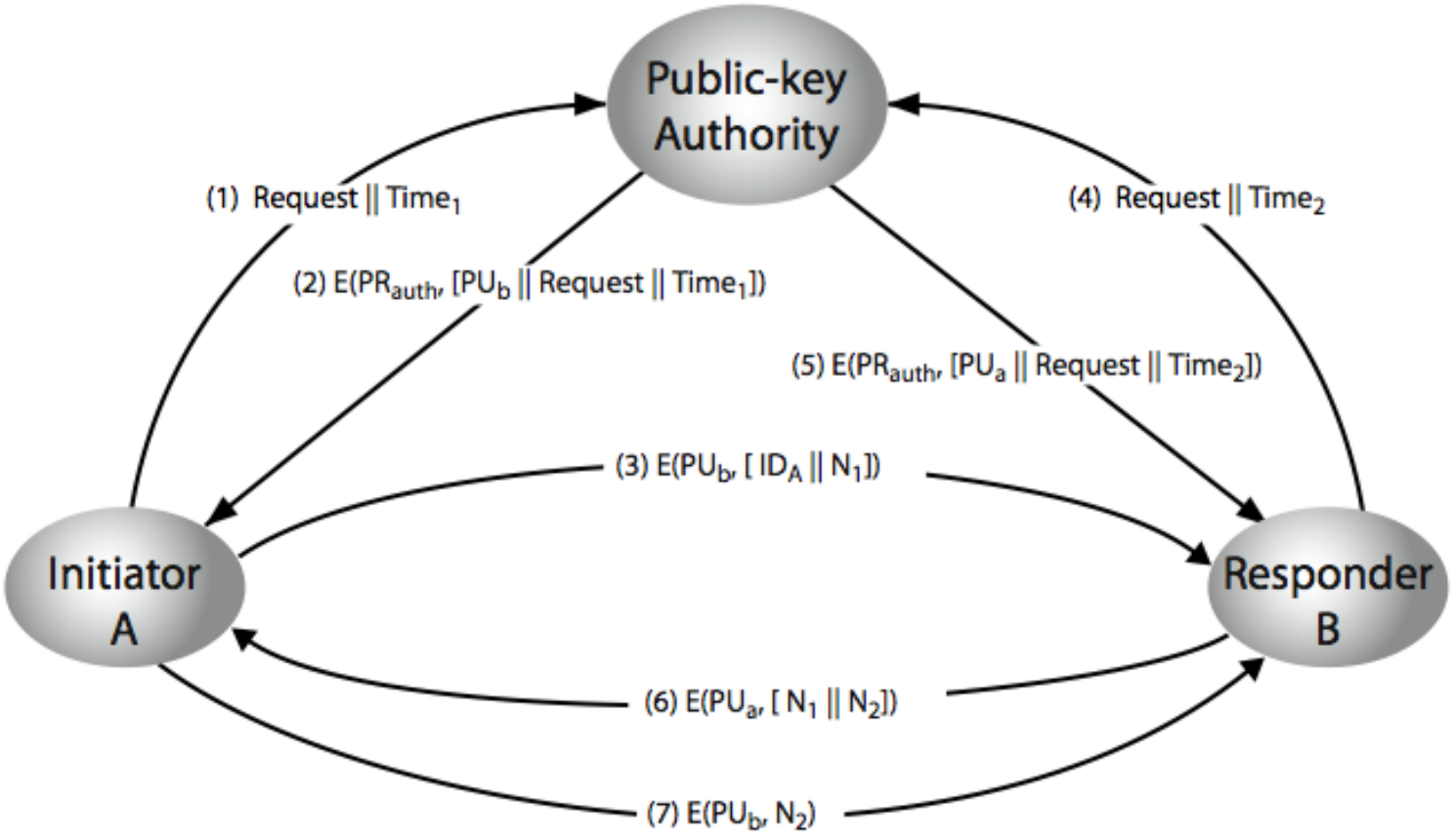


Figure 10.2 Public Key Publication

Public-Key Authority

- Improve security by tightening control over distribution of keys from directory
- Has properties of directory
- And requires users to know public key for the directory
- Then users interact with directory to obtain any desired public key securely
  - does require real-time access to directory when keys are needed

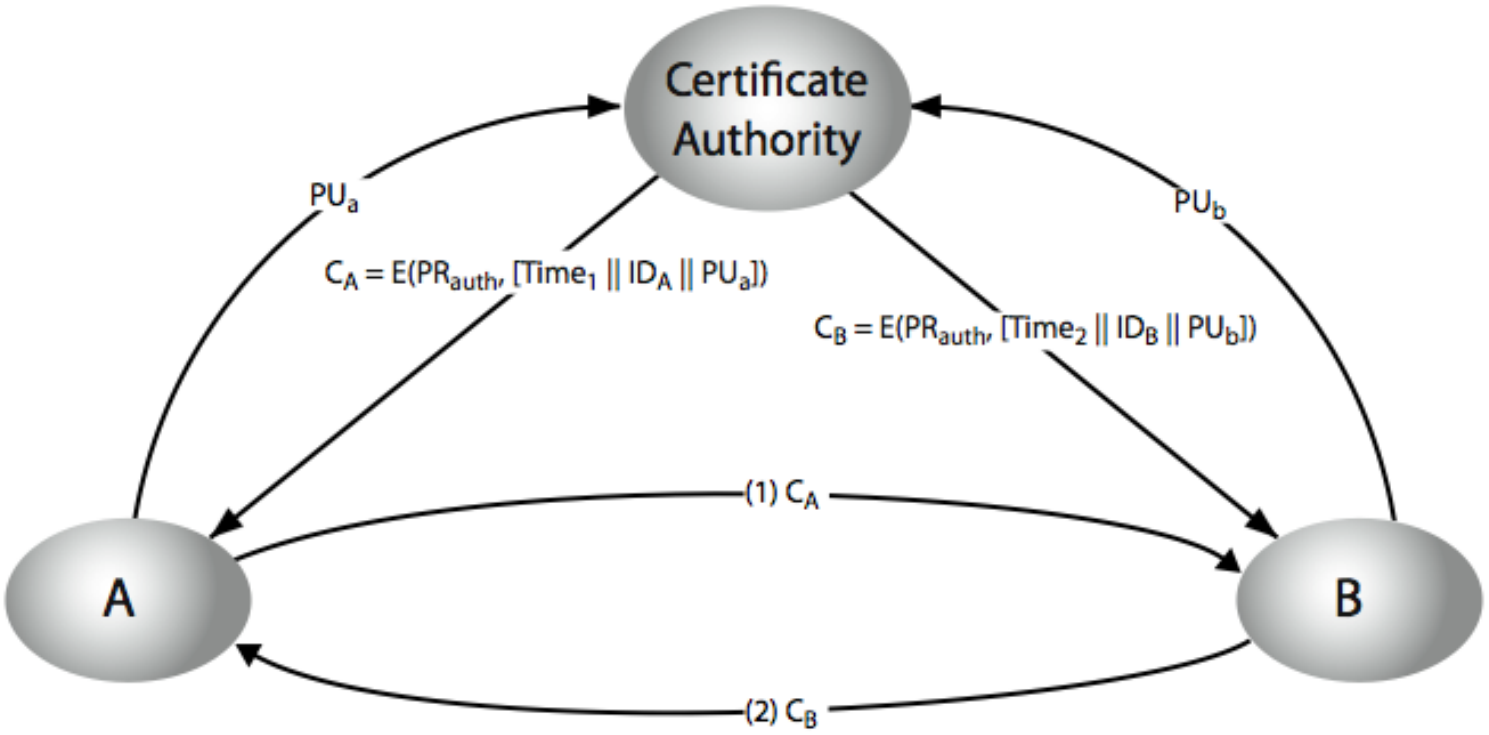
Public-Key Authority



Public-Key Certificates

- Certificates allow key exchange without real-time access to public-key authority
- A certificate binds **identity** to **public key**
  - usually with other info such as period of validity, rights of use etc
- With all contents **signed** by a trusted Public-Key or Certificate Authority (CA)
- Can be verified by anyone who knows the public-key authorities public-key

Public-Key Certificates



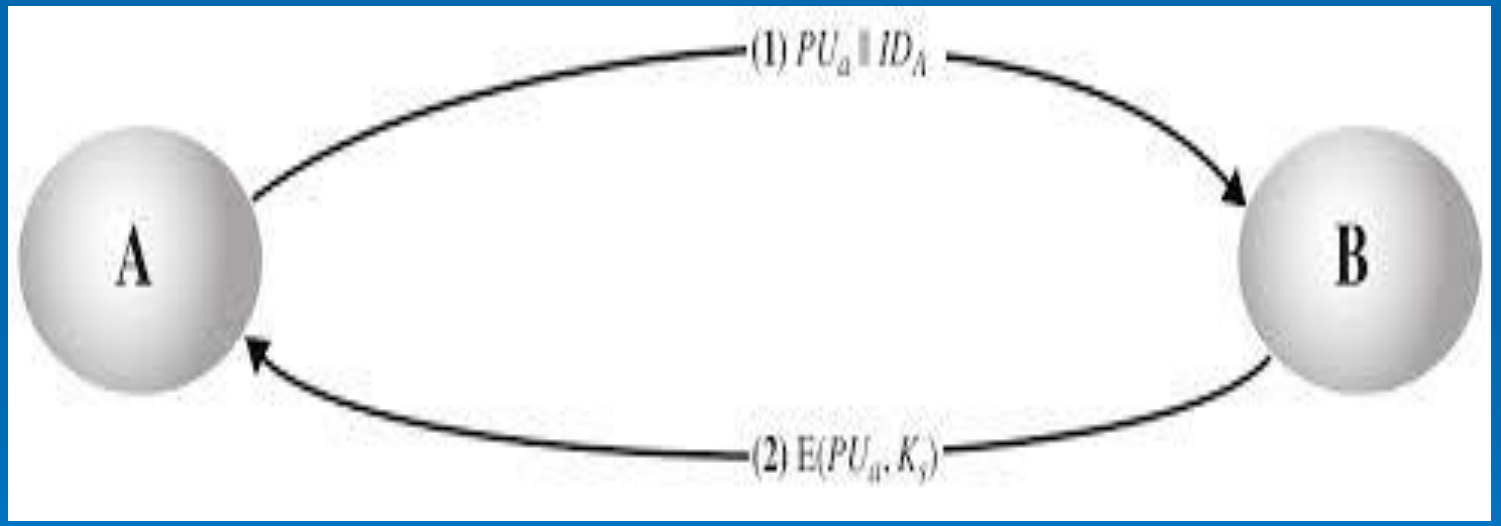
Public-Key Distribution of Secret Keys

- Use previous methods to obtain public-key
- Can use for secrecy or authentication
- But public-key algorithms are slow
- So usually want to use private-key encryption to protect message contents
- Hence need a session key
- Have several alternatives for negotiating a suitable session

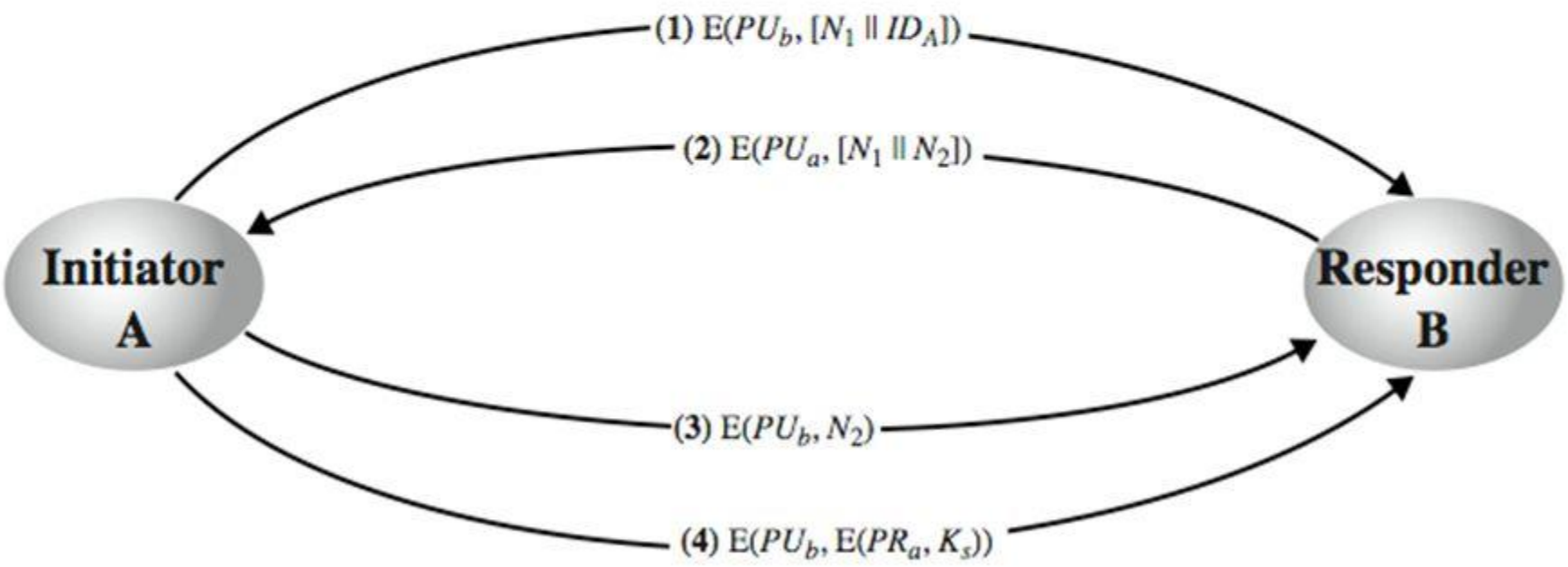
Simple Secret Key Distribution

- Proposed by Merkle in 1979
  - A generates a new temporary public key pair
  - A sends B the public key and their identity
  - B generates a session key K sends it to A encrypted using the supplied public key
  - A decrypts the session key and both use
- Problem is that an opponent can intercept and impersonate both halves of protocol

Simple Secret Key Distribution



Secret Key Distribution with Confidentiality & Authentication



Hybrid Key Distribution

- Retain use of private-key KDC
- Shares secret master key with each user
- Distributes session key using master key
- Public-key used to distribute master keys
  - especially useful with widely distributed users
- Rationale
  - performance
  - backward compatibility

Diffie-Hellman Key Exchange

- First public-key type scheme proposed
- By Diffie & Hellman in 1976 along with the exposition of public key concepts
  - note: now know that Williamson (UK CESG) secretly proposed the concept in 1970
- Is a practical method for public exchange of a secret key
- Used in a number of commercial products

Diffie-Hellman Key Exchange

- A public-key distribution scheme
  - cannot be used to exchange an arbitrary message
  - rather it can establish a common key
  - known only to the two participants
- Value of key depends on the participants (and their private and public key information)
- Based on exponentiation in a finite (Galois) field (modulo a prime or a polynomial) - easy
- Security relies on the difficulty of computing discrete logarithms (similar to factoring) – hard

Diffie-Hellman Setup

- All users agree on global parameters:
  - large prime integer or polynomial q
  - a being a primitive root mod q
- Each user (eg. A) generates their key
  - chooses a secret key (number): $x_A < q$
  - compute their **public key**: $y_A = a^{x_A} \bmod q$
- Each user makes public that key y_A

Diffie-Hellman Key Exchange

- Shared session key for users A & B is K_{AB}:

$$K_{AB} = a^{x_A . x_B} \bmod q$$
$$= y_A^{x_B} \bmod q \quad \text{(which } \mathbf{B} \text{ can compute)}$$
$$= y_B^{x_A} \bmod q \quad \text{(which } \mathbf{A} \text{ can compute)}$$

- K_{AB} is used as session key in private-key encryption scheme between Alice and Bob
- If Alice and Bob subsequently communicate, they will have the **same** key as before, unless they choose new public-keys
- Attacker needs an x, must solve discrete log

Diffie-Hellman Example

- Users Alice & Bob who wish to swap keys:
- Agree on prime $q=353$ and $a=3$
- Select random secret keys:
  - A chooses $x_A=97$, B chooses $x_B=233$
- Compute respective public keys:
  - $y_A=3^{97} \mod 353 = 40$ (Alice)
  - $y_B=3^{233} \mod 353 = 248$ (Bob)
- Compute shared session key as:
  - $K_{AB}= y_B^{x_A} \mod 353 = 248^{97} = 160$ (Alice)
  - $K_{AB}= y_A^{x_B} \mod 353 = 40^{233} = 160$ (Bob)

Key Exchange Protocols

- Users could create random private/public D-H keys each time they communicate
- Users could create a known private/public D-H key and publish in a directory, then consulted and used to securely communicate with them
- Both of these are vulnerable to a meet-in-the-Middle Attack
- Authentication of the keys is needed