# Cryptography and Network Security (CS435/890BN)

Part Eight

(Key Management)

# Key Management and Distribution

- topics of cryptographic key management / key distribution are complex
  - cryptographic, protocol & management issues
- symmetric schemes require both parties to share a common secret key
- public key schemes require parties to acquire valid public keys
- have concerns with doing both

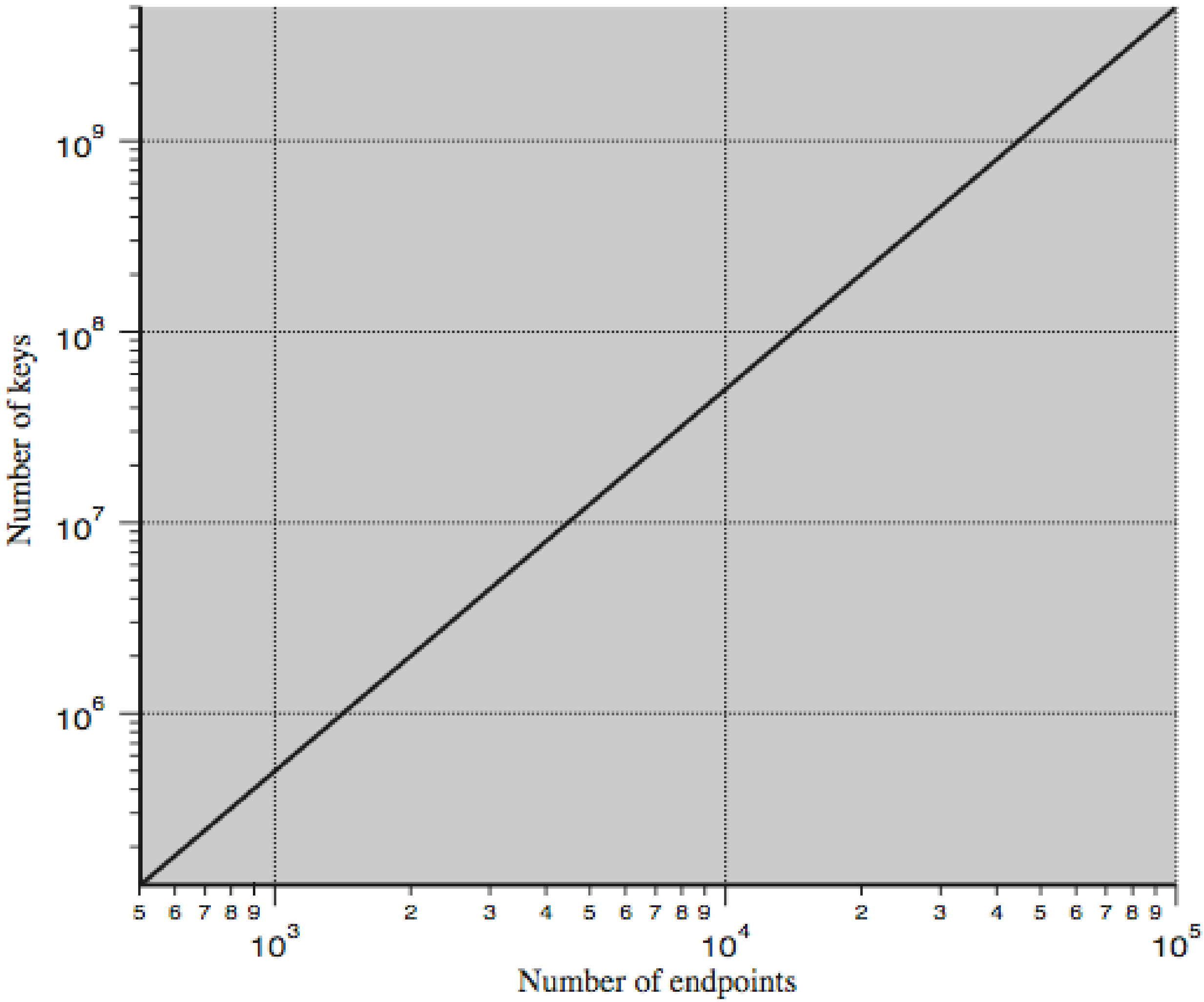
# Key Distribution

- symmetric schemes require both parties to share a common secret key
- issue is how to securely distribute this key
- whilst protecting it from others
- frequent key changes can be desirable
- often secure system failure due to a break in the key distribution scheme

# Key Distribution

- given parties A and B have various **key distribution** alternatives:
  1. A can select key and physically deliver to B
  2. third party can select & physically deliver key to A & B
  3. if A & B have communicated previously can use previous key to encrypt a new key
  4. if A & B have secure communications with a third party C, C can relay key between A & B

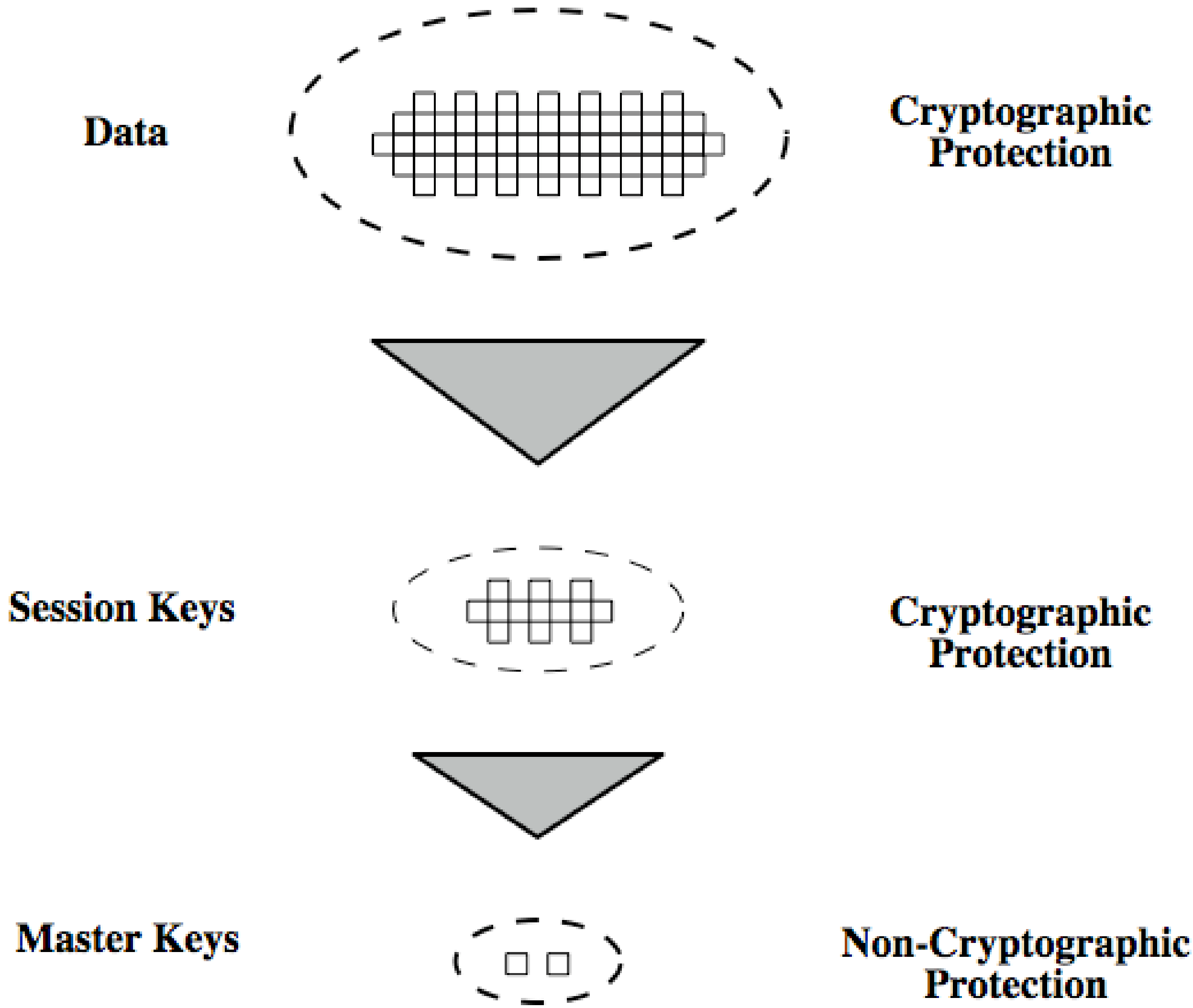
# Key Distribution Task



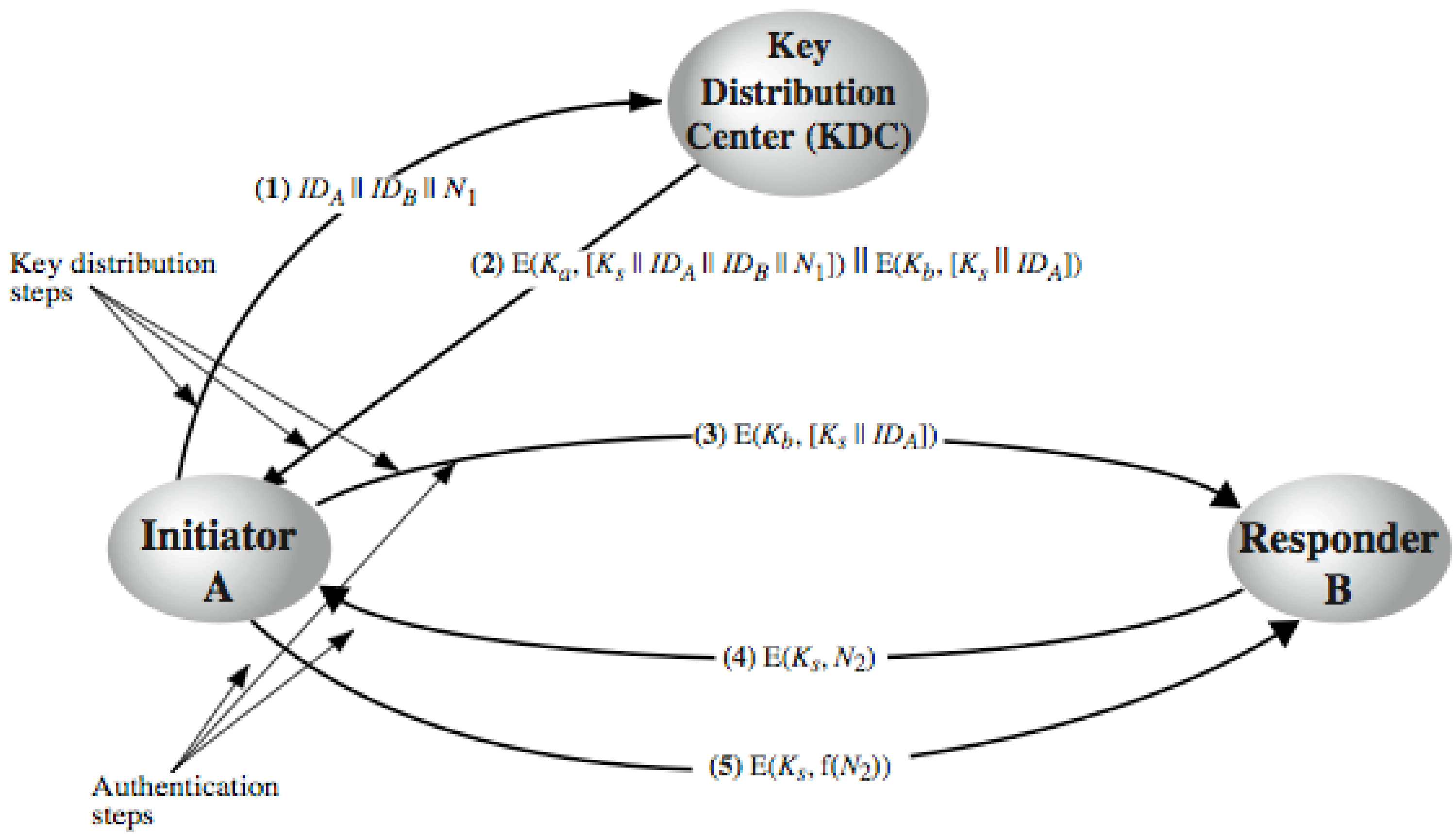
# Key Hierarchy

- typically have a hierarchy of keys
- session key
  - temporary key
  - used for encryption of data between users
  - for one logical session then discarded
- master key
  - used to encrypt session keys
  - shared by user & key distribution center

# Key Hierarchy



# Key Distribution Scenario



# Key Distribution Issues

- hierarchies of KDC's required for large networks, but must trust each other
- session key lifetimes should be limited for greater security
- use of automatic key distribution on behalf of users, but must trust system
- use of decentralized key distribution
- controlling key usage

# Hierarchical Key Control

- For communication among entities within the same local domain, the local KDC is responsible for key distribution
  - If two entities in different domains desire a shared key, then the corresponding local KDC's can communicate through a global KDC
- The hierarchical concept can be extended to three or more layers
- Scheme minimizes the effort involved in master key distribution because most master keys are those shared by a local KDC with its local entities
  - Limits the range of a faulty or subverted KDC to its local area only

# Session Key Lifetime

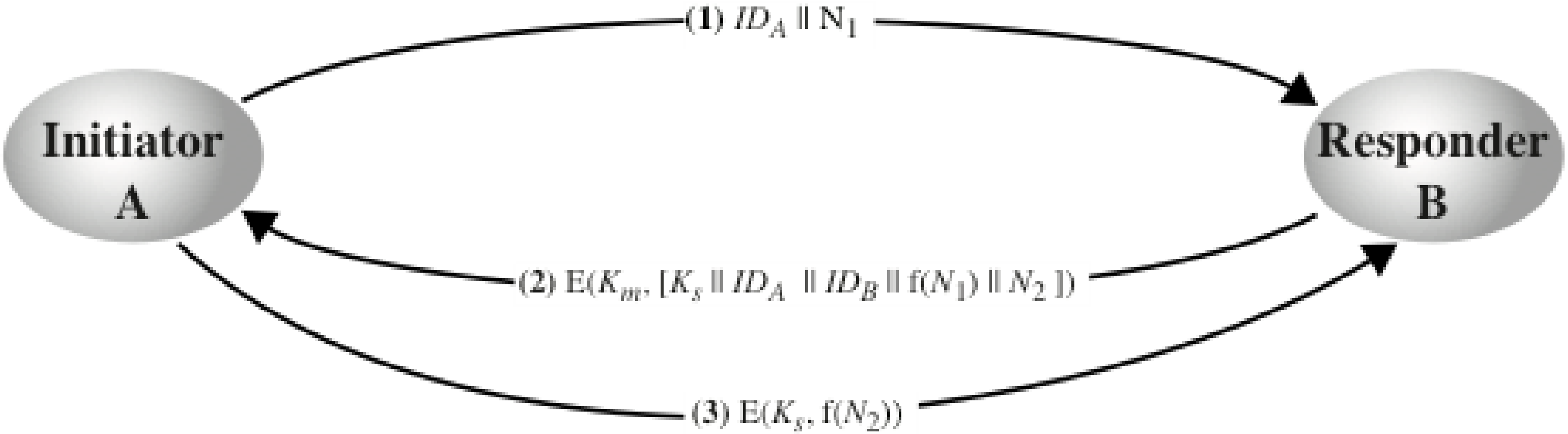
For connection-oriented protocols one choice is to use the same session key for the length of time that the connection is open, using a new session key for each new session

A security manager must balance competing considerations:

- The more frequently session keys are exchanged, the more secure they are
- The distribution of session keys delays the start of any exchange and places a burden on network capacity

For a connectionless protocol there is no explicit connection initiation or termination, thus it is not obvious how often one needs to change the session key

# Decentralized Key Distribution




1. A issues a request to B for a session key and includes a nonce, $N_1$.
2. B responds with a message that is encrypted using the shared master key. The response includes the session key selected by B, an identifier of B, the value $f(N_1)$, and another nonce, $N_2$.
3. Using the new session key, A returns $f(N_2)$ to B.

- Note: A decentralized approach requires that each end system be able to communicate in a secure manner with all potential partner end systems for purposes of session key distribution. Thus, there may need to be as many as $[n(n-1)]/2$ master keys for a configuration with $n$ end systems.

# Controlling Key Usage

- The concept of a key hierarchy and the use of automated key distribution techniques greatly reduce the number of keys that must be manually managed and distributed
- It also may be desirable to impose some control on the way in which automatically distributed keys are used
  - For example, in addition to separating master keys from session keys, we may wish to define different types of session keys on the basis of use

# Key Controls

- Associate a tag with each key
  - For use with DES and makes use of the extra 8 bits in each 64-bit DES key
  - The eight non-key bits ordinarily reserved for parity checking form the key tag
  - Because the tag is embedded in the key, it is encrypted along with the key when that key is distributed, thus providing protection

Drawbacks:

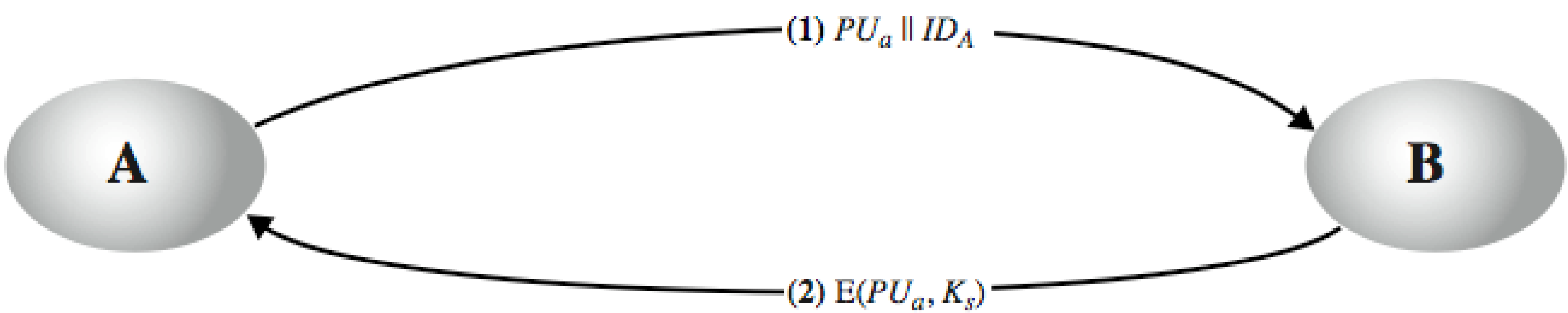
- The tag length is limited to 8 bits, limiting its flexibility and functionality
- Because the tag is not transmitted in clear form, it can be used only at the point of decryption, limiting the ways in which key use can be controlled

# Symmetric Key Distribution Using Public Keys

- public key cryptosystems are inefficient
  - so almost never use for direct data encryption
  - rather use to encrypt secret keys for distribution

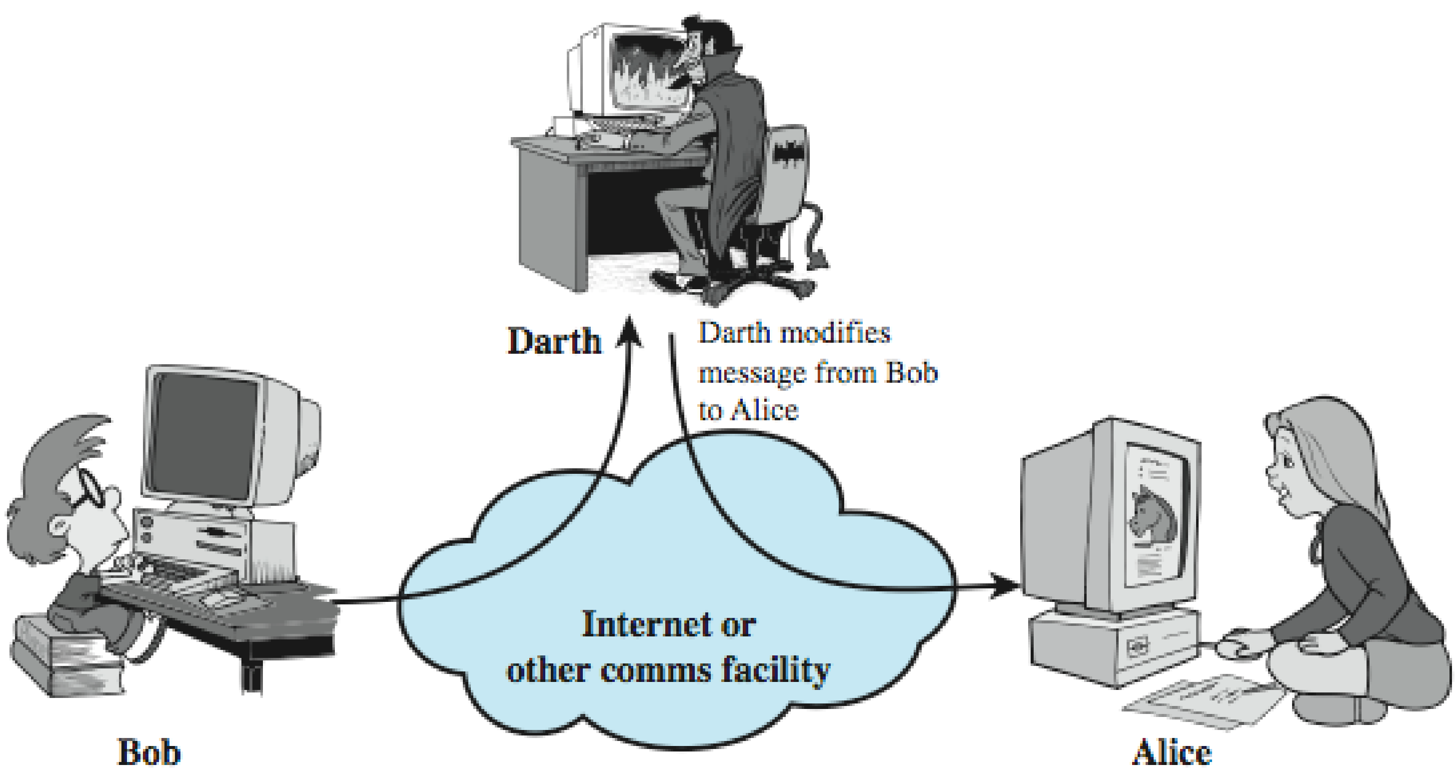
# Simple Secret Key Distribution

- Merkle proposed this very simple scheme
  - allows secure communications
  - no keys before/after exist

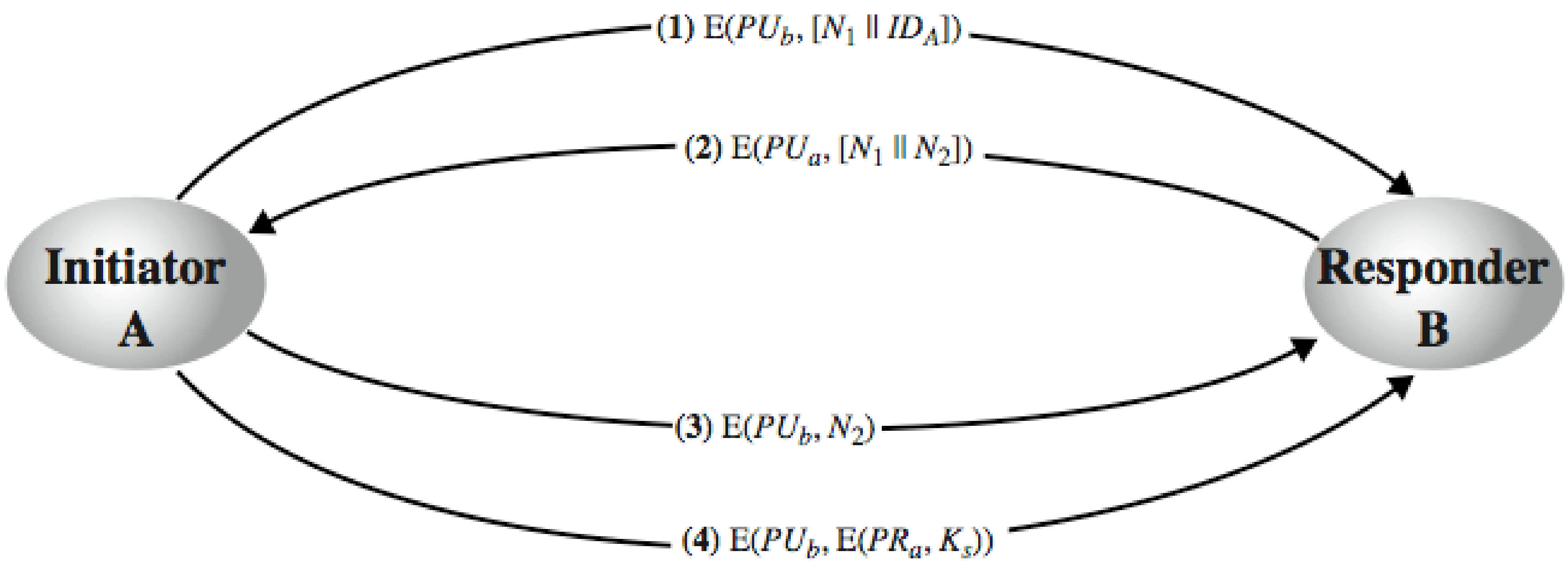


# Man-in-the-Middle Attack

- this very simple scheme is vulnerable to an active man-in-the-middle attack



# Public-Key Distribution of Secret Keys with Confidentiality and 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

# Distribution of Public Keys

- Several techniques have been proposed for the distribution of public keys. Virtually all these proposals can be grouped into the following general schemes:
  - 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 news groups 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

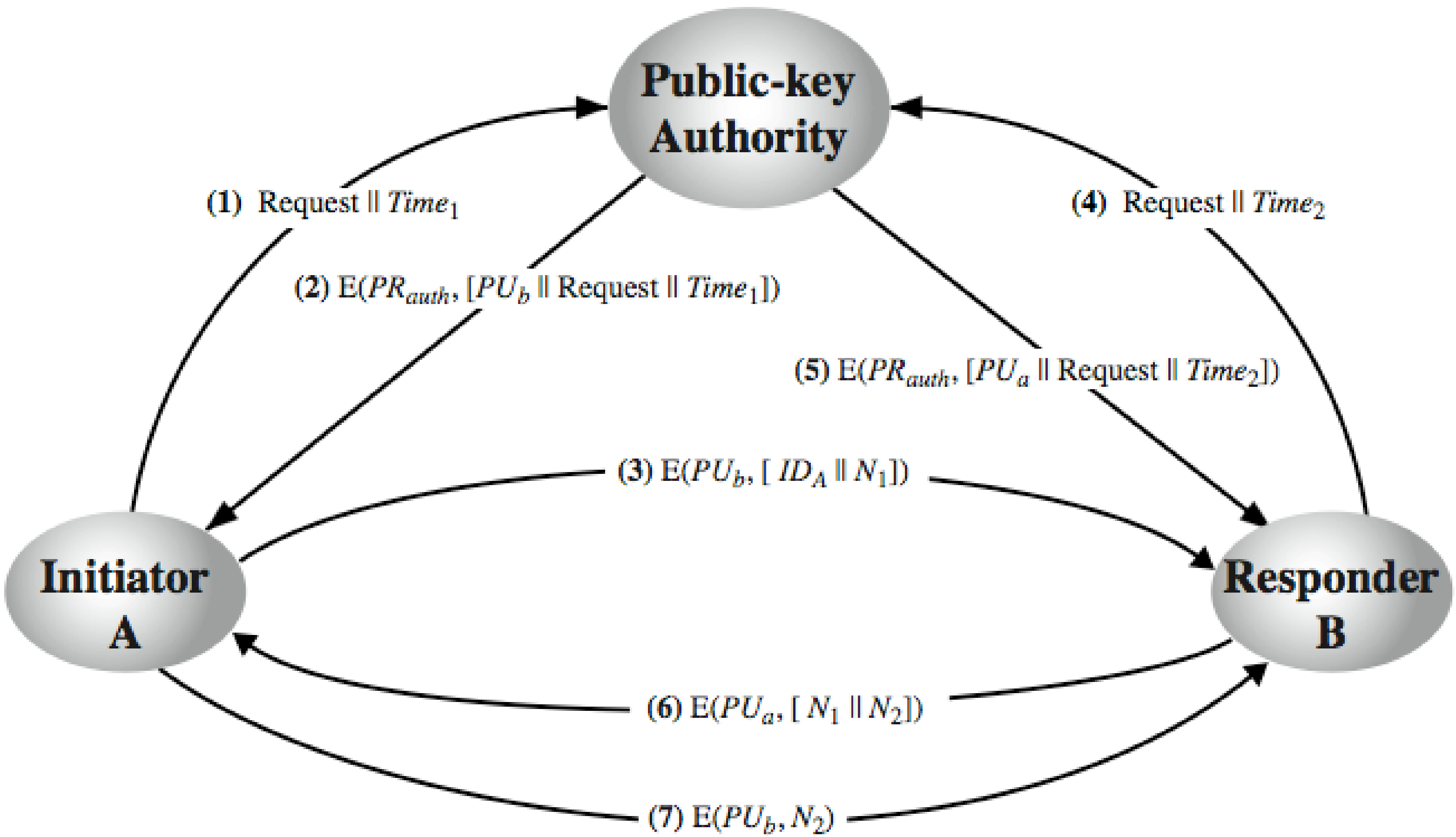
# Publicly Available Directory

- 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
- still vulnerable to tampering or forgery

# 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
  - may be vulnerable to tampering

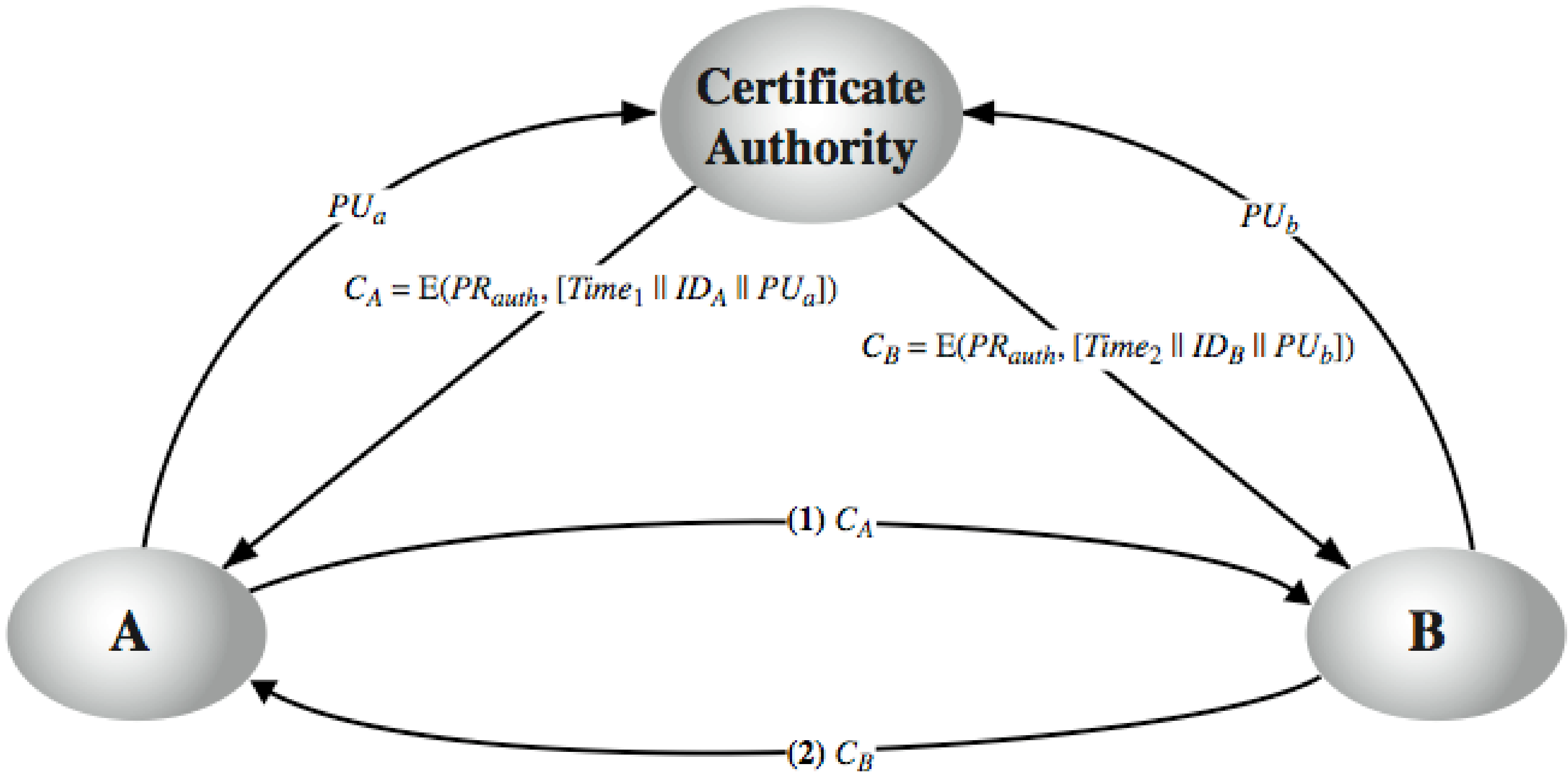
# Public-Key Authority



# 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

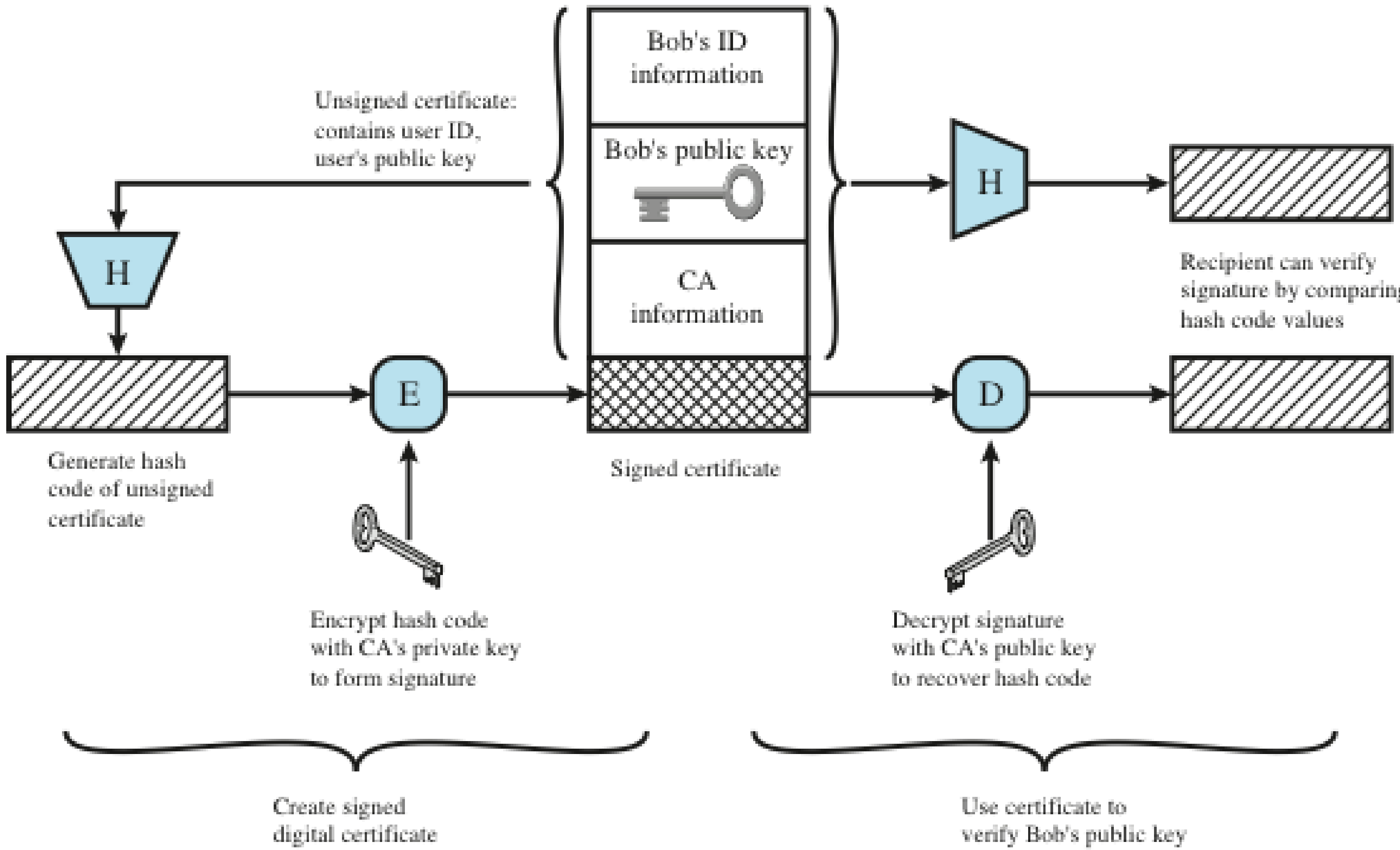
# Public-Key Certificates



# X.509 Certificates

- Part of the X.500 series of recommendations that define a directory service
  - The directory is, in effect, a server or distributed set of servers that maintains a database of information about users
- X.509 defines a framework for the provision of authentication services by the X.500 directory to its users
  - Was initially issued in 1988 with the latest revision in 2012
  - Based on the use of public-key cryptography and digital signatures
  - Does not dictate the use of a specific algorithm but recommends RSA
  - Does not dictate a specific hash algorithm
- Each certificate contains the public key of a user and is signed with the private key of a trusted certification authority
- X.509 defines alternative authentication protocols based on the use of public-key certificates

# Public-Key Certificate Use



# Certificates

Created by a trusted Certification Authority (CA) and have the following elements:

- Version
- Serial number
- Signature algorithm identifier
- Issuer name
- Period of validity
- Subject name
- Subject's public-key information
- Issuer unique identifier
- Subject unique identifier
- Extensions
- Signature

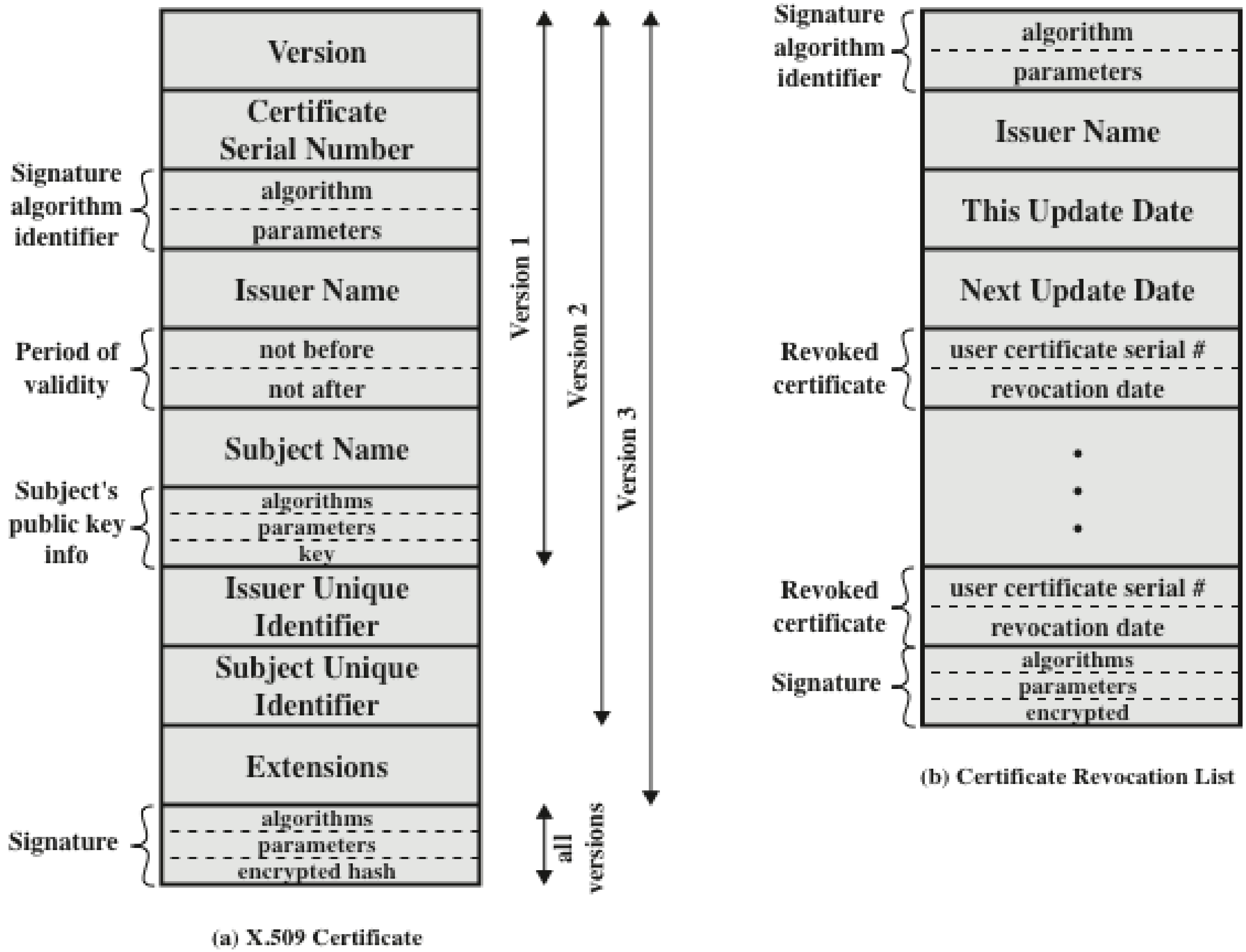



**Figure 14.15 X.509 Formats**

# Obtaining a Certificate

**User certificates generated by a CA have the following characteristics:**

- Any user with access to the public key of the CA can verify the user public key that was certified
- No party other than the certification authority can modify the certificate without this being detected

- Because certificates are unforgeable, they can be placed in a directory without the need for the directory to make special efforts to protect them
  - In addition, a user can transmit his or her certificate directly to other users
- Once B is in possession of A’s certificate, B has confidence that messages it encrypts with A’s public key will be secure from eavesdropping and that messages signed with A’s private key are unforgeable

# CA Hierarchy

- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy
- use certificates linking members of hierarchy to validate other CA's
  - each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy

# Certificate Revocation

- Each certificate includes a period of validity
  - Typically a new certificate is issued just before the expiration of the old one
- It may be desirable on occasion to revoke a certificate before it expires, for one of the following reasons:
  - The user's private key is assumed to be compromised
  - The user is no longer certified by this CA
  - The CA's certificate is assumed to be compromised

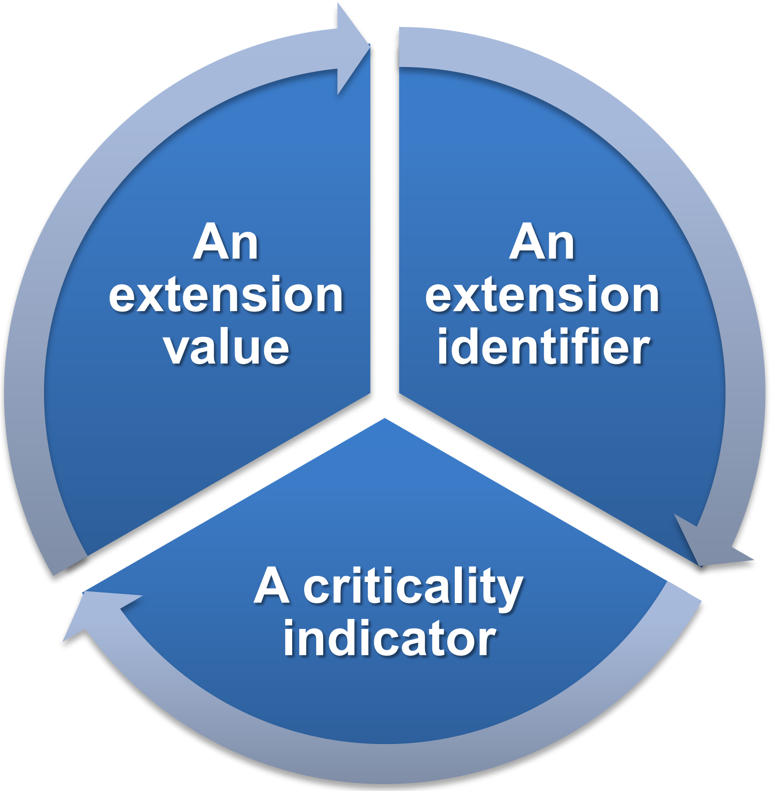
– Each CA must maintain a list consisting of all revoked but not expired certificates issued by that CA
  - These lists should be posted on the directory

# X.509 Version 3

- Version 2 format does not convey all of the information that recent design and implementation experience has shown to be needed
- Rather than continue to add fields to a fixed format, standards developers felt that a more flexible approach was needed
  - Version 3 includes a number of optional extensions

– The certificate extensions fall into three main categories:
  - Key and policy information
  - Subject and issuer attributes
  - Certification path constraints

Each extension consists of:

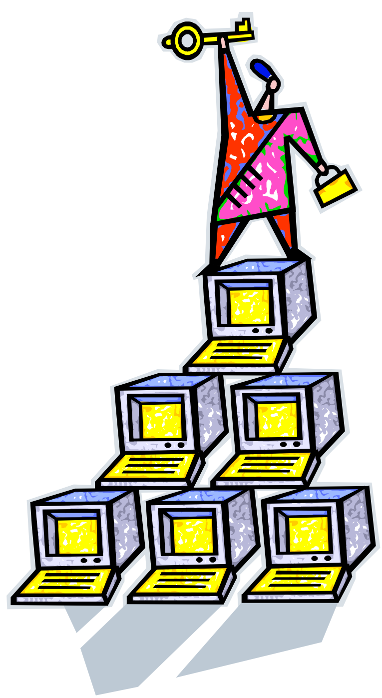


# Key and Policy Information

- These extensions convey additional information about the subject and issuer keys plus indicators of certificate policy
- A certificate policy is a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements

**Included are:**

- Authority key identifier
- Subject key identifier
- Key usage
- Private-key usage period
- Certificate policies
- Policy mappings



# Certificate Subject and Issuer Attributes

- These extensions support alternative names, in alternative formats, for a certificate subject or certificate issuer
- Can convey additional information about the certificate subject to increase a certificate user's confidence that the certificate subject is a particular person or entity
- The extension fields in this area include:
  - Subject alternative name
  - Issuer alternative name
  - Subject directory attributes

# Certification Path Constraints

- These extensions allow constraint specifications to be included in certificates issued for CAs by other CAs
- The constraints may restrict the types of certificates that can be issued by the subject CA or that may occur subsequently in a certification chain
- The extension fields in this area include:
  - Basic constraints
  - Name constraints
  - Policy constraints