Outline

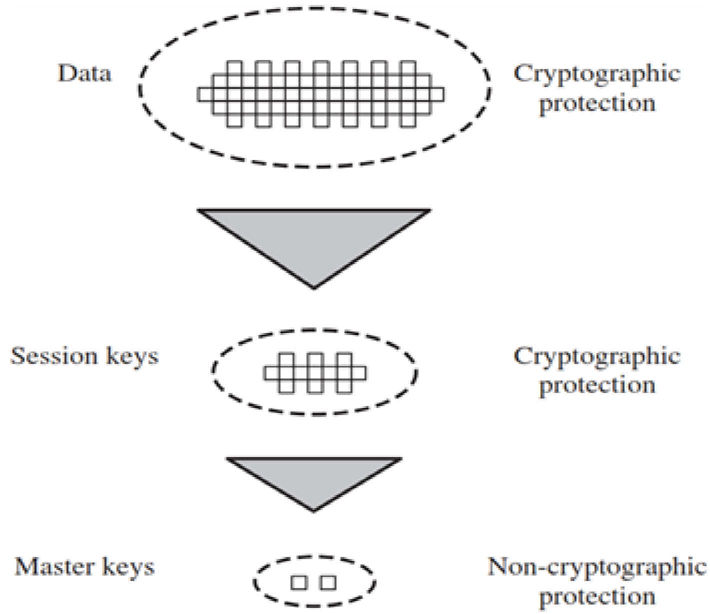
- Key management and distribution
- Symmetric key distribution using symmetric encryption
- Symmetric key distribution asymmetric encryption
- Distribution of public keys
- X.509 certificates
- Public key infrastructure (PKI)

Key Distribution

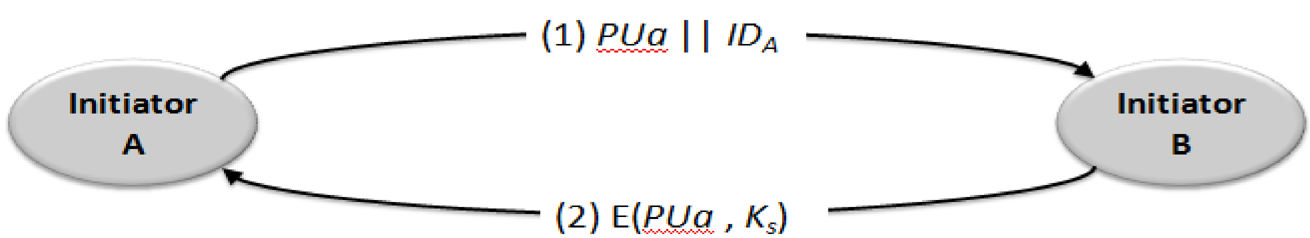
- **Key distribution** is the function that delivers a key to two parties who wish to exchange secure encrypted data.
- Some sort of mechanism or protocol is needed to provide for the secure distribution of keys.
- Key distribution often involves the use of **master keys**, which are infrequently used and are long lasting, and **session keys**, which are generated and distributed for temporary use between two parties.

Key Hierarchy

- Communication between end systems is encrypted using a temporary key, often referred to as a **session key**.
- Session keys are transmitted in encrypted form, using a **master key** that is shared by the key distribution center and an end system or user

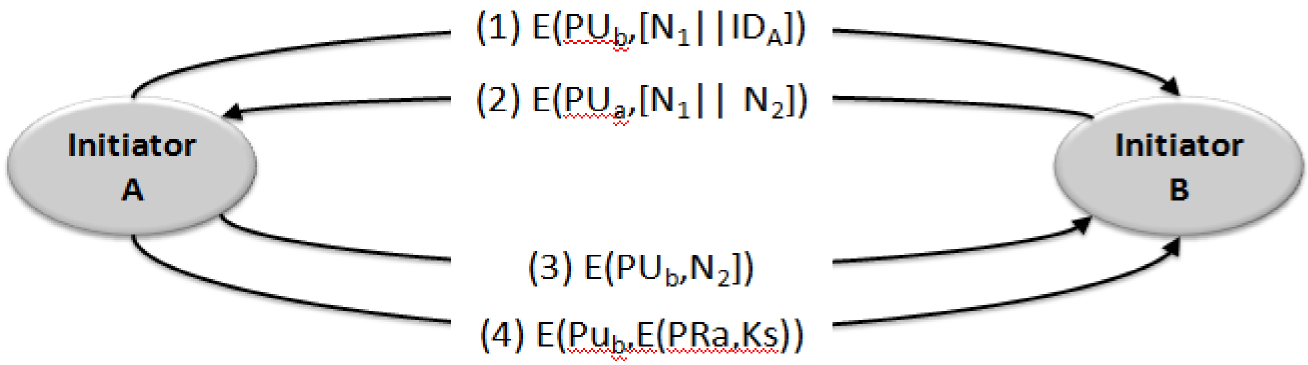


Simple Secret Key Distribution



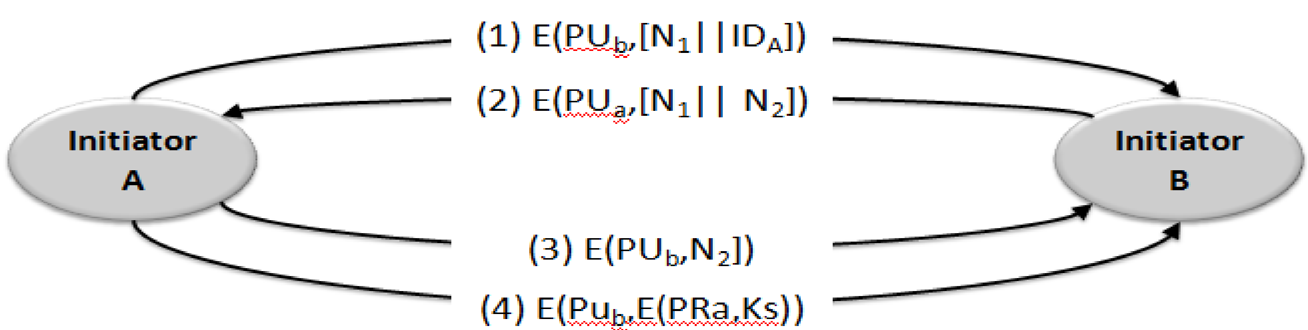
1. **A** generates a public/private key pair **{PUa, PRa}** and transmits a message to **B** consisting of **PUa** and an identifier of **A**, **ID_A**.
2. **B** generates a secret key, **Ks**, and transmits it to **A**, encrypted with **A**'s public key.
3. **A** computes **D(PRa, E(PUa, Ks))** to recover the secret key. Because only **A** can decrypt the message, only **A** and **B** will know the identity of **Ks**.
4. **A** discards **PUa** and **PRa** and **B** discards **PUa**.

Secret Key Distribution with Confidentiality & Authentication



1. A uses B's public key to encrypt a message to B containing an identifier of A (I_A) and a nonce (N_1), which is used to identify this transaction uniquely.
2. B sends a message to A encrypted with PUa and containing A's (N_1) as well as a new nonce generated by B (N_2). Because only B could have decrypted message (1), the presence of N_1 in message (2) assures A that the correspondent is B.

Secret Key Distribution with Confidentiality & Authentication

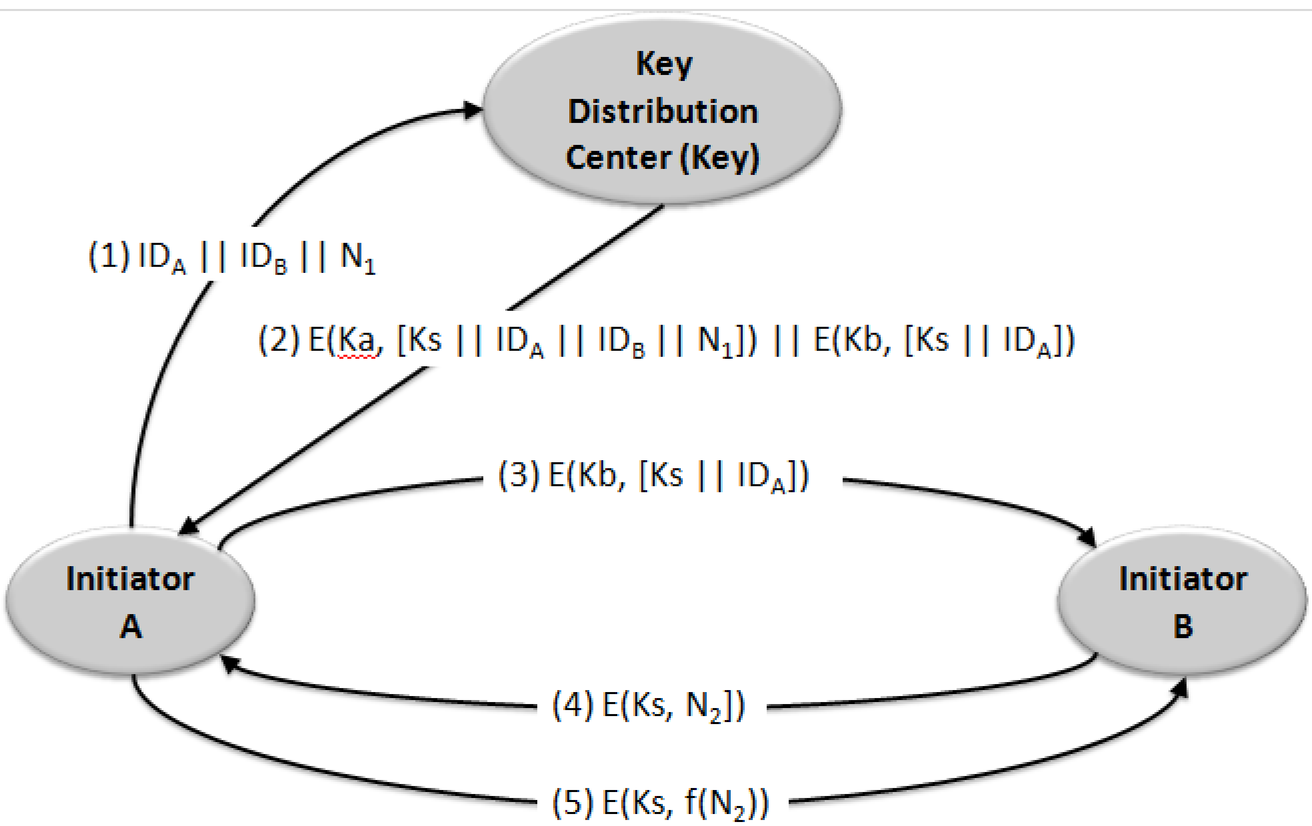


3. A returns N_2, encrypted using B's public key, to assure B that its correspondent is A.
4. A selects a secret key Ks and sends M = E(PUb, E(PRa, Ks)) to B. Encryption with B's public key ensures that only B can read it; encryption with A's private key ensures that only A could have sent it.
5. B computes D(PUa, D(PRb, M)) to recover the secret key.

Symmetric key distribution using symmetric encryption

- Two parties A and B, key distribution can be achieved in a number of ways, as follows:
  1. **A** can select a key and **physically deliver key** to **B**.
  2. Third party can select the key and physically deliver it to **A** and **B**.
  3. If **A** and **B** have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
  4. If **A** and **B** each has an encrypted connection to a third party **C**, **C** can deliver a key on the encrypted links to A and B.

Key Distribution Scenario



Key Distribution Scenario

1. **A** requests from the KDC a session key to protect a logical connection to **B**. The message includes the identity of **A** and **B** and a unique nonce **N1**.
2. The KDC responds with a message encrypted using **Ka** that includes a one-time session key **Ks** to be used for the session, the original request message to enable **A** to match response with appropriate request, and info for **B**
3. **A** stores the session key for use in the upcoming session and forwards to **B** the information from the KDC for **B**, namely, **E(Kb, [Ks || IDA])**.
4. At this point, a session key has been securely delivered to **A** and **B**, and they may begin their protected exchange.
5. Using the new session key for encryption B sends a nonce **N_2** to **A**.
6. Also using **Ks**, **A** responds with **$f(N_2)$**. These steps assure **B** that the original message it received (step 3) was not a replay. Note that the actual key distribution involves only steps 1 through 3 but that steps 4 and 5, as well as 3, perform an authentication function.

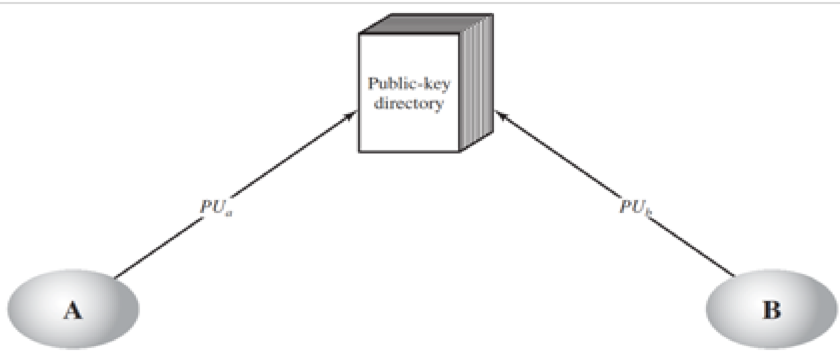
Distribution of Public Keys

1. Public announcement
2. Publicly available directory
3. Public-key authority
4. Public-key certificates

1. Public Announcement

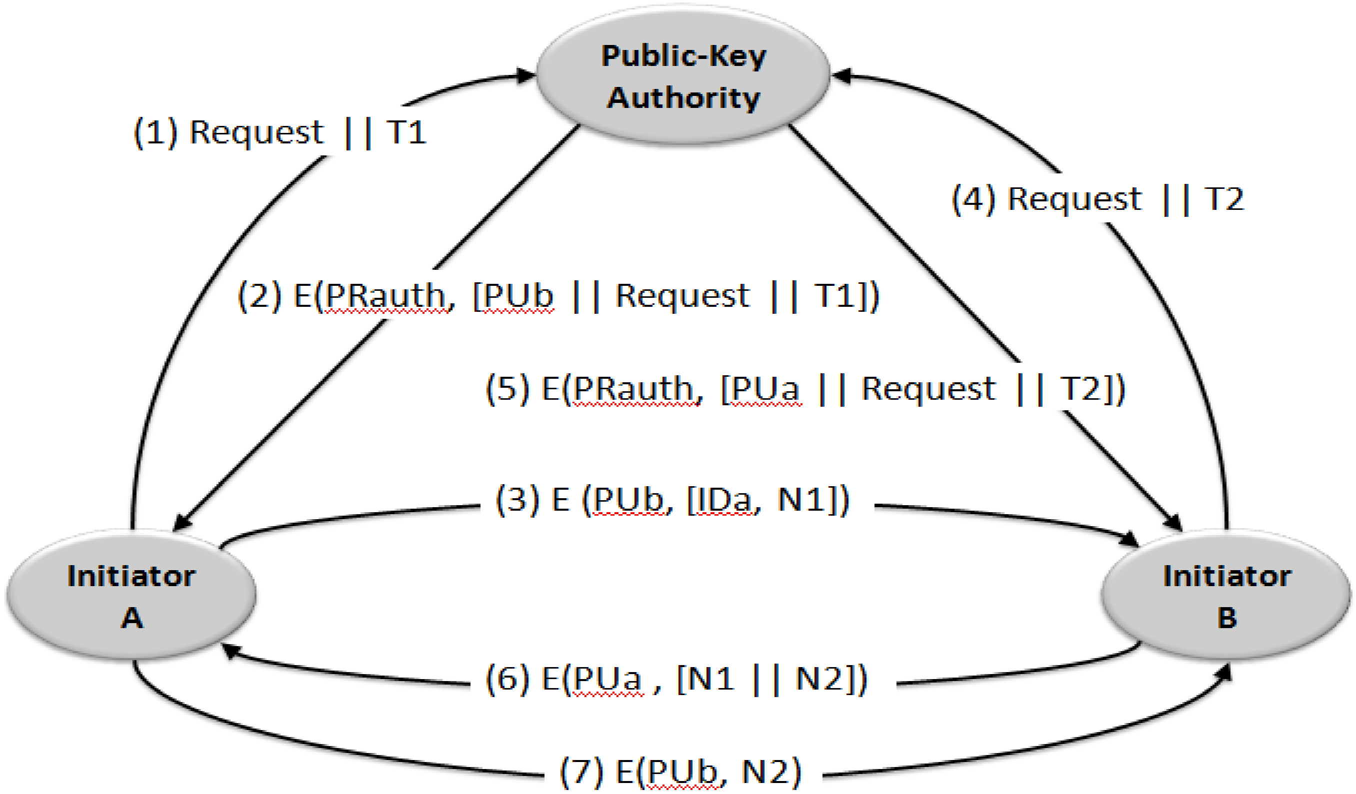
- Some user could pretend to be user A and send a public key to another participant or broadcast such a public key.
- Until such time as user A discovers the forgery and alerts other participants, the forger is able to read all encrypted messages intended for A and can use the forged keys for authentication

2. Publicly Available Directory



1. The **authority** maintains a directory with a **{name, public key}** entry for each participant.
2. Each participant registers a public key with the directory authority.
3. A participant may replace the existing key with a new one at any time.
4. Participants could also access the directory electronically. For this purpose, secure, authenticated communication from the authority to the participant is mandatory.

3. Public-Key Authority



3. Public-Key Authority – Cont...

1. **A** sends a timestamped message to the public-key authority containing a request for the current public key of **B**.
2. The authority responds with a message that is encrypted using the authority's private key .
3. Message contains **PUb**, Original request, Original time stamp **T1**

   A stores **B**'s public key and also uses it to encrypt a message to B containing an identifier of **A(IDa)** and a **nonce(N1)** , which is used to identify this transaction uniquely.

4, 5. **B** retrieves **A**'s public key from the authority in the same manner as **A** retrieved **B**'s public key.

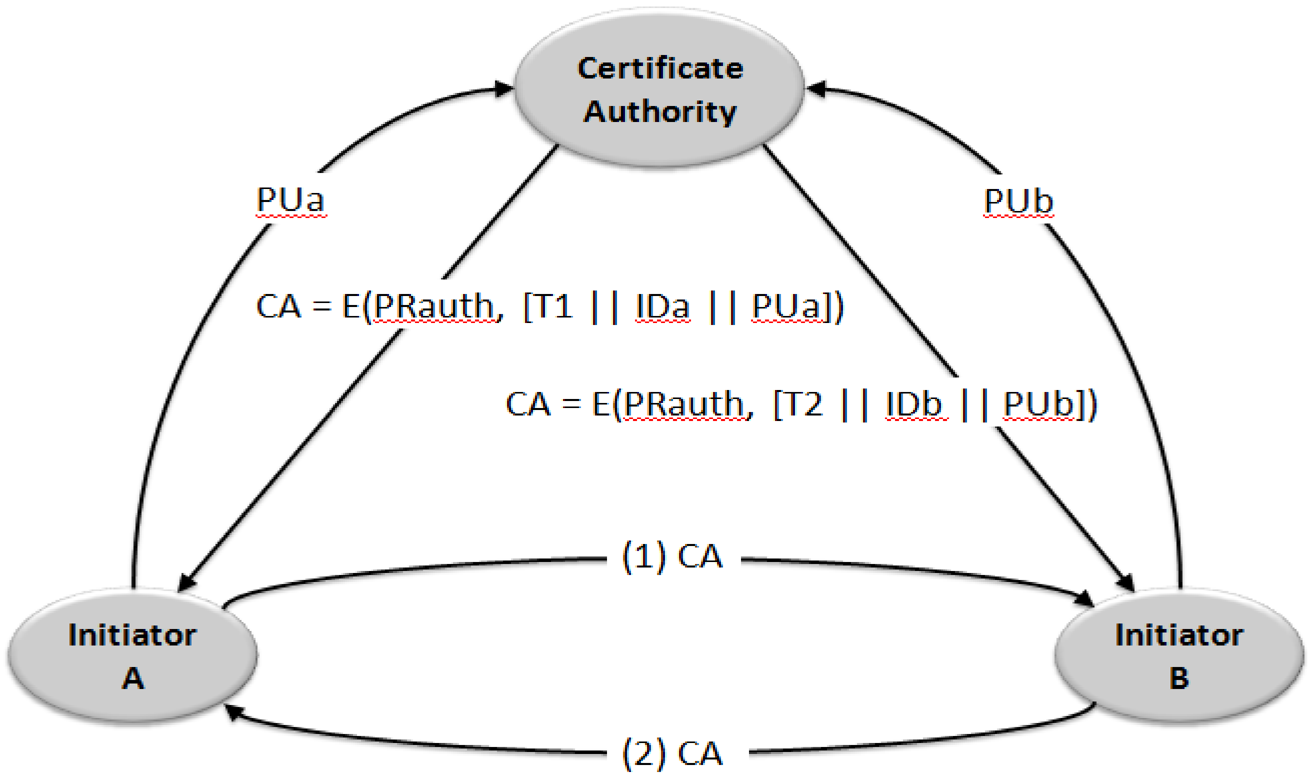
3. Public-Key Authority – Cont...

6. **B** sends a message to **A** encrypted with **PUa** and containing **A**'s **nonce(N1)** as well as a new nonce generated by **B(N2)**. Because only **B** could have decrypted message (3), the presence of **N1** in message (6) assures **A** that the correspondent is **B**.
7. **A** returns **N2**, which is encrypted using **B**'s public key, to assure **B** that its correspondent is **A**.

4. Public-Key Certificates

- Any participant can read a **certificate** to determine the name and public key of the certificate's owner.
- Any participant can verify that the **certificate** originated from the **certificate authority** and is not counterfeit.
- Only the **certificate authority** can create and update certificates.
- Any participant can verify the currency of the certificate.

4. Public-Key Certificates – Cont...



4. Public-Key Certificates – Cont...

- Each participant applies to the **certificate authority**, supplying a public key and requesting a certificate.
- For participant A, the authority provides a certificate of the form

  CA = E (PRauth, [T || IDa || PUa])

- A may then pass this certificate on to any other participant, who reads and verifies the certificate as follows:

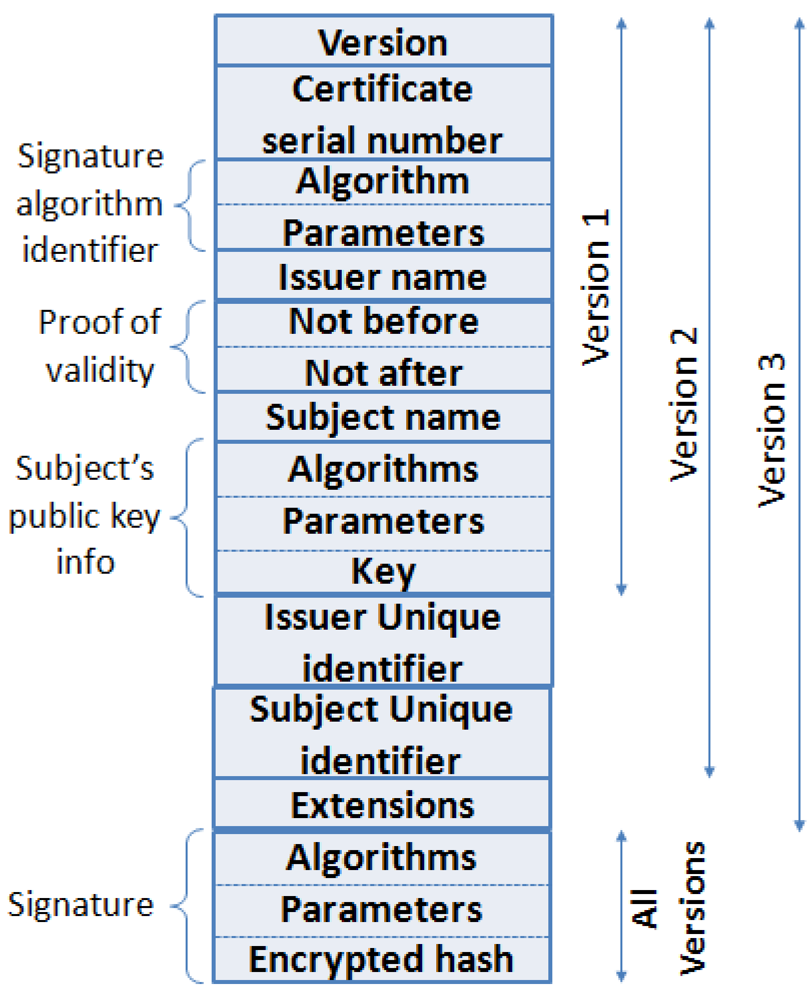
  D(PUauth, CA)

  = D(PUauth, E (PRauth, [T || IDa || PUa]))

  = (T || IDa || PUa)

X.509 Certificates

- **X.509** defines the format for public-key certificates. used in a variety of applications.
- **X.509** defines a framework for the provision of authentication services by the X.500 directory to its users.
- The directory may serve as a repository of public-key certificates.
- Each certificate contains the **public key of a user** and is **signed with the private key** of a trusted certification authority.



X.509 Formats

X.509 Format – Cont...

- **Version:** Differentiates among successive versions of the certificate format; the default is version 1.
- **Serial number:** An integer value unique within the issuing CA that is unambiguously associated with this certificate.
- **Signature algorithm identifier:** The algorithm used to sign the certificate together with any associated parameters.
- **Issuer name:** X.500 name of the CA that created and signed this certificate.
- **Period of validity:** Consists of two dates: the first and last on which the certificate is valid.
- **Subject name:** The name of the user to whom this certificate refers.

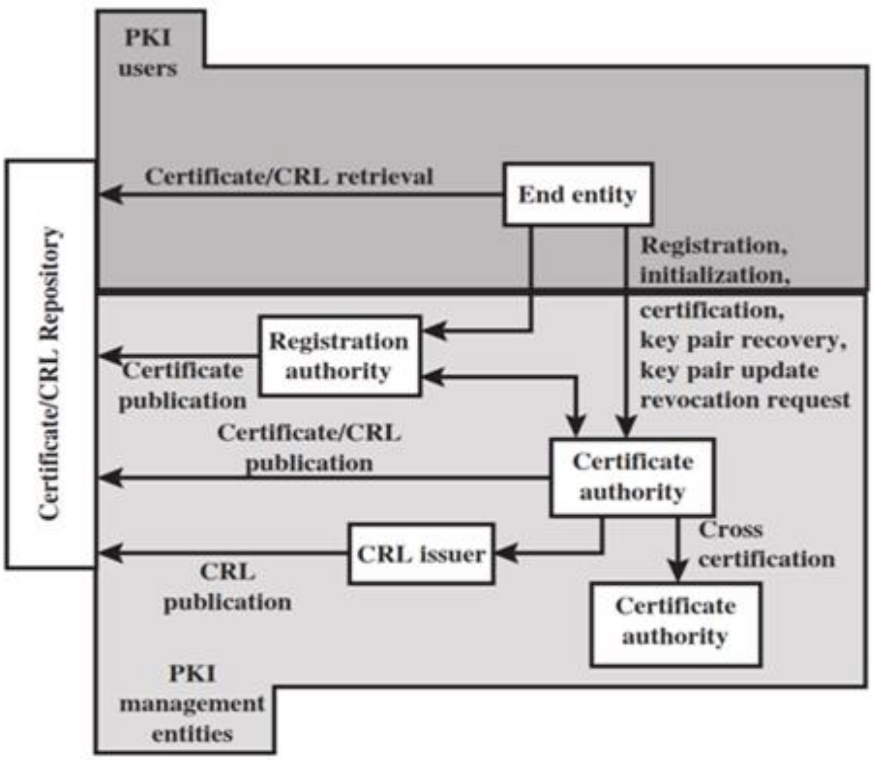
X.509 Format – Cont...

- **Subject's public-key information:** The public key of the subject, plus an identifier of the algorithm for which this key is to be used, together with any associated parameters.
- **Issuer unique identifier:** An optional-bit string field used to identify uniquely the issuing CA in the event the X.500 name has been reused for different entities.
- **Subject unique identifier:** An optional-bit string field used to identify uniquely the subject in the event the X.500 name has been reused for different entities.
- **Extensions:** A set of one or more extension fields.

Public key Infrastructure (PKI)

- A **public-key infrastructure (PKI)** is defined as the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography.
- The principal objective for developing a PKI is to enable secure, convenient, and efficient acquisition of **public keys**.

Public key Infrastructure (PKI)



Public key Infrastructure (PKI) – Cont...

- **End entity:** A generic term used to denote end users, devices (e.g., servers, routers), or any other entity that can be identified in the subject field of a public-key certificate.
- **Certification authority (CA):** The issuer of certificates and (usually) certificate revocation lists (CRLs).
- **Registration authority (RA):** An optional component that can assume a number of administrative functions from the CA.
- **CRL issuer:** An optional component that a CA can delegate to publish CRLs.
- **Repository:** A generic term used to denote any method for storing certificates and CRLs so that they can be retrieved by end entities.