

# Public Key Infrastructure (PKI)

## Contents:

- Secret key distribution.
- Public key distribution.
- Public key certificate.
- Public key infrastructure.

# Secret Key Distribution

- For secure communications based on symmetric key systems, the secret keys have to be distributed before any decryption can be done.
- Distribution of secret keys can be very costly.
  - By post? insecure, unreliable, time delay.
  - Use courier? unreliable, time delay, expensive.
  - Use secure underground cable? Too costly, hard to change.
  - Use public channels? Virtually everyone can get access.

# Secret Key Distribution

- Management of secret key is a big problem.
  - The same key cannot be used too many times.
  - Keys need to be changed from time to time.
  - Cipher code book (a collection of many secret keys): once used in military but not used any more today.
- To enable secure communication between any two out of  $n$  parties, there should be  $n(n - 1)/2$  different secret keys to be distributed!
  - Need to update? all the new keys need to be distributed.

# Secret Key Distribution: D-H Key Exchange

Recall Diffie-Hellman key exchange protocol:

**Common knowledge to  $A$  and  $B$ :** a prime  $p$ ,  
and a random number  $g < p$ .

$A \longrightarrow B$ :  $\alpha = g^{r_1} \bmod p$ , where  $r_1$  is a random  
number.

$B \longrightarrow A$ :  $\beta = g^{r_2} \bmod p$ , where  $r_2$  is random  
number.

$A$  **compute**:  $K_1 = \beta^{r_1} \bmod p$ .

$B$  **compute**:  $K_2 = \alpha^{r_2} \bmod p$ .

**Common key:** Then  $K_1 = K_2$  is the common  
key.

Attack to D-H protocol:

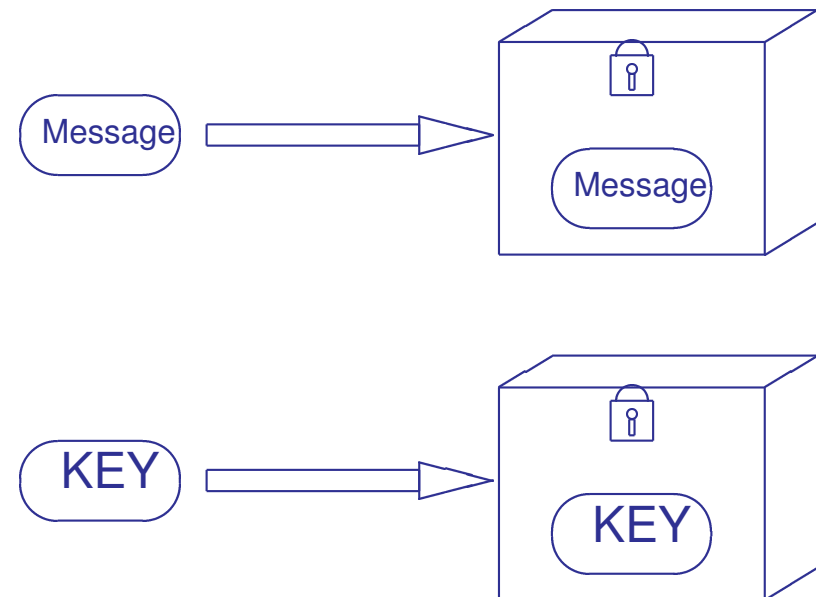
- $E$  pretends to be  $B$  and establishes a common key with  $A$ .
- $A$  thinks it is  $B$  and communicates with  $E$ .
- $E$  gets all the (confidential) information that  $A$  is supposed to share with  $B$ , without  $B$  being aware of it.

Why is the attack successful?

- No authentication, i.e. there is no way for  $A$  and  $B$  to check each other's identity.

# Secret Key Distribution

- ▶ Use public key system? Yes.
  - ▶ Convenient, cheap, and easy to update.
- ▶ But whose public key am I supposed to use?! And how do I trust it?



# Public Key Distribution

But how to distribute public keys?

- $\mathcal{A}$  sends her public key to  $\mathcal{B}$ .
  - To how many people does  $\mathcal{A}$  have to send her public key?
  - How does  $\mathcal{B}$  trust that it is  $\mathcal{A}$ 's public key?
- An important issue in public key distribution is to make sure the key belongs to the right person.

Public key certificate

- A certificate issued by a mutually (by owner of key and whoever uses the certificate) trusted third party (TTP).
- It contains the identification information (not the sensitive information such as credit card number) of the key owner, the public key, issuing authority, expiration date, etc.

# Public Key Certificate

Who issues the public key certificate for you?

- A public key certificate issuing authority.
- The authority's public key is publicly available (and trusted).
- The validity of individual's public

key can be verified using the authority's public key.

- The individual's public key can be retrieved from the public key certificate.
- You have to trust the authority, if you want to use the public key certificate services.

# Public Key Certificate

How to use a public key certificate?

- Whoever wants to use a public key has to get the corresponding public key certificate from a directory, check it's validity and extract the relevant information.
- Public key certificates are updated periodically or on request by the owner. Always check the public key certificate before using the public key.

# Public Key Infrastructure

- To enable public key systems to work properly, a single TTP to manage the public key certificates is not enough.
- **Store the public key certificates:** International standard is ISO authentication framework, known as X.509 protocols.
- **How many TTPs:**

**CAs:** Certificate authority (may be more than one), who issues public key certificates.

**RA:** Registration authority, who verifies the certificates, i.e., to monitor whether CA works properly.

**X.500** One or more directories as specified in X.500 standard, to hold the certificates.

**X.509** Protocol to manage the certificates (how to put, get, update the certificates).

# Public Key Infrastructure

Certification phase.

- **Initiate:** Requester (subject) applies for a public key certificate from CA.
  - Requester presents his/her public key to be certified.
  - Requester shows his/her ID (e.g. off-line).
  - CA generates a public key certificate.
  - RA checks the validity of the certificate, and puts it in a directory.

# Public Key Infrastructure

Certification phase.

- **Update:** Requester (subject) applies for his/her public key certificate to be updated.
  - Requester sends his/her new public key to be certified.
  - Requester signs request message, new public key, ID information (old public key serial number, etc.), using his/her old private key.
  - CA verifies the request information using requester's old public key.
  - CA generates a new public key certificate.
  - RA checks the validity of the new certificate, and updates the old one from the directory.

# Public Key Infrastructure

Verification phase.

- User Bob wants to get Alice's public key.
  - Bob gets Alice's public key certificate.
  - Bob gets CA's public key.
  - Bob verifies the validity of Alice's public key certificate.
  - Bob extracts Alice's public key from the certificate.
- Alice does the same thing to get Bob's public key.
- Alice and Bob can then establish a secure communication, sign documents to each other.

# Public Key Infrastructure

Verification phase: What happens if Alice and Bob do not have the same CA?

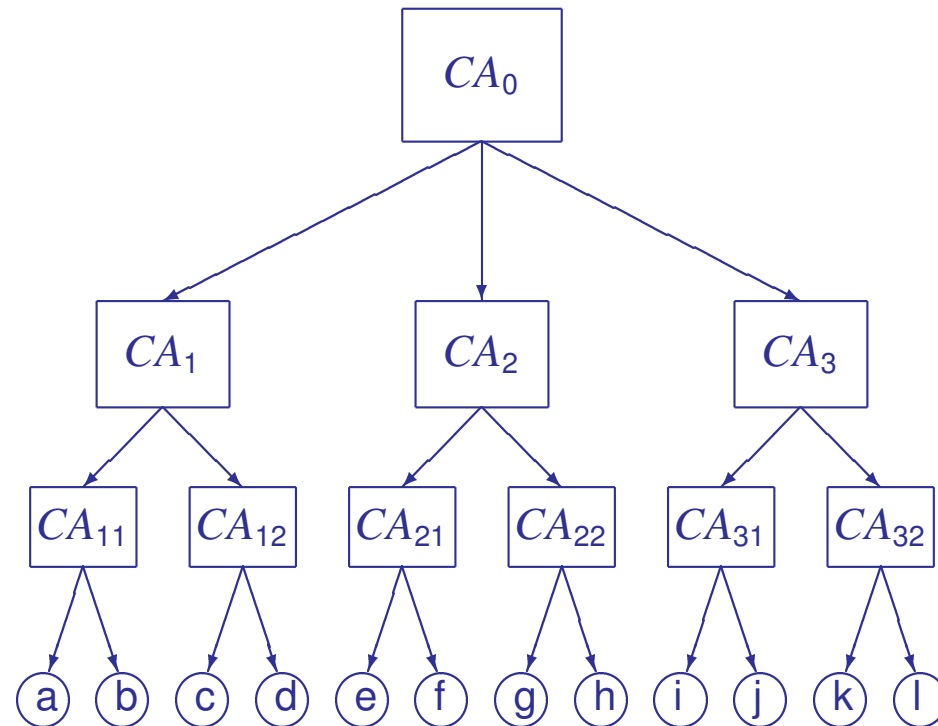
- User Bob wants to get Alice's public key.
  - Bob finds the closest (not necessarily physically) directory to Alice and gets her public key certificate.
  - Bob checks CA's ID (information is associated with Alice's public key certificate).
  - Bob gets and verifies CA's public key certificate (using another CA's public key).
  - Bob verifies the validity of Alice's public key certificate.
  - Bob extracts Alice's public key from the certificate.

# Public Key Infrastructure

Hierarchical structure of  
certificates and issuing authorities

□: Certificate authority.

○: End user.



# Public Key Infrastructure

- Each certification authority has issued a public key certificate for all its associated CA's.
- When a CA changes its public key, all of its associated CA's have to re-issue a public key certificate for the CA.
- If user **A** want to get user **D**'s public key, she has to
  1. get **CA<sub>1</sub>**'s public key certificate issued by **CA<sub>11</sub>**; then
  2. get **CA<sub>12</sub>**'s public key certificate issued by **CA<sub>1</sub>**; then

3. get **D**'s public key certificate issued by **CA<sub>12</sub>**; then

4. retrieve and verify **D**'s public key.

- This process is noted as

$$CA_{11} \ll CA_1 \gg CA_1 \ll CA_{12} \gg$$
$$CA_{12} \ll D \gg$$

- The procedure that **D** can get **A**'s public key is the reverse.
- How can **E** get **A**'s public key (from the above diagram)?

# About Pretty Good Privacy (PGP)

- **What is PGP?** A free software package for email security.
- PGP has its own means of key management. It does not use X.509 certificates.
- PGP uses public key ring, a database on the end user's server. If the public key cannot be retrieved from the public key ring, the public key is not available.
- PGP uses private key ring, a database on the user's local server.
- Retrieval of private key needs a pass phrase (like a password). The private key is encrypted using the pass phrase and stored on the computer.
- It uses public key as well as secret key systems. Public key algorithms are used for session key encryption, while session key is used for real message (body of email) encryption using symmetric key encryption algorithm (triple-DES, or IDEA).
- User can choose encryption only, signature only, or neither or both.