Security Management

- Problem: how do you get keys in the first place?
- Key distribution: securely associate an entity with a key
  - Example: Public Key Infrastructure (PKI)
- Key establishment: establish session keys
  - Use public key cryptography (we already know how to do it)
  - Diffie-Hellman key exchange

Public Key Infrastructure (PKI)

- System managing public key distribution on a wide-scale
- Trust distribution mechanism
- Allow arbitrary level of trust

Components of a PKI

Digital Certificate

- Signed data structure that binds an entity (E) with its corresponding public key (K_E^1)
  - Signed by a recognized and trusted authority, i.e., Certification Authority (CA)
  - Provide assurance that a particular public key belongs to a specific entity
- How?
  - CA generates K_{CA}(E, K_E^1)
  - Everyone can verify signature using K_{CA}^1
Certification Authority (CA)

- People, processes responsible for creation, delivery and management of digital certificates
- Organized in an hierarchy (use delegation – see next)

Registration Authority

- People, processes and/or tools that are responsible for
  - Authenticating the identity of new entities (users or computing devices)
  - Requiring certificates from CA’s.

Certificate Repository

- A database which is accessible to all users of a PKI, contains:
  - Digital certificates,
  - Certificate revocation information
  - Policy information

Example

- Alice generates her own key pair.
  - Bob generates his own key pair.
  - Both sent their public key to a CA and receive a digital certificate

Certificate Revocation

- Process of publicly announcing that a certificate has been revoked and should no longer be used.
- Approaches:
  - Use certificates that automatically time out
  - Use certificate revocation list
**Key Establishment:**
**Diffie-Hellman Key Exchange**

- Agree on two numbers $n$, $g$; both number can be made public!
- Alice and Bob pick two secret numbers $x$ and $y$
- Similar to public-key cryptography
  - Example: For Alice, $K_A = x$, $K_A^* = g^x \mod n$

---

**Secure Group Management**

- Motivation: offer high availability for security services
- How: replicate services
- Problem: how to add a new replica to a group without compromising the integrity of the group?
Authorization Management

- Granting authorization rights
- Related with access control which verifies access rights (see book)

Capabilities (1)

- Capability:
  - Unforgeable data structure for a specific resource R
  - Specify access right the holder has with respect to R
- Capability in Amoeba:

<table>
<thead>
<tr>
<th>Server port</th>
<th>Object</th>
<th>Rights</th>
<th>Check</th>
</tr>
</thead>
<tbody>
<tr>
<td>48 bits</td>
<td>24 bits</td>
<td>8 bits</td>
<td>48 bits</td>
</tr>
</tbody>
</table>

Capabilities (2)

- Generation of a restricted capability from an owner capability

Delegation

- A wants to delegate an operation on a resource to B
- Problem: how does A delegates its access rights to B?
- Solutions: A signs (A, B, R)
  - Avoid this problem using a proxy (Neuman scheme)
  - Proxy: a token allowing its owner to operate with the same or restricted rights as the entity granting the token

Delegation: Neuman Scheme

- The general structure of a proxy as used for delegation:

Delegation: Neuman Scheme

- Using a proxy to delegate and prove ownership of access rights
- In practice $S_{proxy}$, $S_{proxy}$ can be a public-private key pair and N can be a nonce
Kerberos

- Based on Needham-Schroeder authentication scheme
- Developed at MIT

Example: Kerberos

- Authentication in Kerberos:
  - AS: Authentication server
  - TGS: Ticket Granting Service
  - T: timestamp used to avoid replay attacks of message 6

Example: Kerberos

- Setting up a secure channel in Kerberos: