Security

- Introduction
- Secure channels
- Access control
- Security management

Security: Dependability Revisited

Basics:
- component provides services to client
- client depends on component

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>Accessible and usable upon demand for authorized entities</td>
</tr>
<tr>
<td>Reliability</td>
<td>Continuity of service delivery</td>
</tr>
<tr>
<td>Safety</td>
<td>Very low probability of catastrophes</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>No unauthorized disclosure of information</td>
</tr>
<tr>
<td>Integrity</td>
<td>No accidental or malicious alterations of information have been performed (even by authorized entities)</td>
</tr>
</tbody>
</table>
Types of Threats

Basics:
- **Subject**: issues request
- **Channel**: carrier of requests and replies
- **Object**: provides service

<table>
<thead>
<tr>
<th>Threat</th>
<th>Channel</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interruption</td>
<td>Preventing message transfer</td>
<td>Denial of service</td>
</tr>
<tr>
<td>Inspection</td>
<td>Reading the content of transferred messages</td>
<td>Reading the data contained in an object</td>
</tr>
<tr>
<td>Modification</td>
<td>Changing message content</td>
<td>Changing an object’s encapsulated data</td>
</tr>
<tr>
<td>Fabrication</td>
<td>Inserting messages</td>
<td>Spoofing an object</td>
</tr>
</tbody>
</table>

Security Mechanisms

- **Encryption**:
  - Transform data into something that an attacker cannot understand (confidentiality)
  - check whether something has been modified (integrity)
- **Authentication**:
  - verify the **identity** of a subject
- **Authorization**:
  - determine if a subject is permitted to request service
- **Auditing**:
  - trace subjects and requests
  - can help catch an attacker
Security policy

Prescribes how to use mechanisms to protect against attacks. Requires that a model of possible attacks is described

**Example:** Globus security architecture
- There are multiple administrative domains
- Local operations subject to local security policies
- Global operations require requester to be globally known
- Inter-domain operations require mutual authentication
- Global authentication replaces local authentication
- Users can delegate privileges to processes
- Credentials can be shared between processes in the same domain
Design Issue: Focus of Control

Layering of Security Mechanisms (1)

The logical organization of a distributed system into several layers
Layering of Security Mechanisms (2)

Several sites connected through a wide-area backbone service.

SSL backbone

Distribution of Security Mechanisms

The principle of RISSC as applied to secure distributed systems.

Reduced Interface for Secure System Components
Cryptography (1)

Intruders and eavesdroppers in communication

Cryptography (2)

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_{A,B}$</td>
<td>Secret key shared by A and B</td>
</tr>
<tr>
<td>$K_{A}^+$</td>
<td>Public key of A</td>
</tr>
<tr>
<td>$K_{A}^-$</td>
<td>Private key of A</td>
</tr>
</tbody>
</table>

Symmetric system:
- Use a single key to encrypt the plaintext and decrypt the ciphertext.
- Sender and receiver share the secret key

Asymmetric system:
- Use different keys for encryption and decryption
- One is private, the other public

Hashing system:
- Only encrypt data and produce a fixed-length digest
- There is no decryption; only comparison is possible
DES: data encryption standard

DES algorithm

one encryption round

Details of per-round key generation
Public-Key Cryptosystems: RSA

Steps of private and public key generation:
1. Choose two very large prime numbers, \( p \) and \( q \)
2. Compute \( n = p \times q \) and \( z = (p - 1) \times (q - 1) \)
3. Choose a number \( d \) that is relatively prime to \( z \)
4. Compute the number \( e \) such that \( e \times d = 1 \mod z \)

2 integers are relatively prime if they share no common positive factors except 1

Hash Functions : MD5 (1)
Secure Channels

- Authentication
- Message Integrity and confidentiality
- Secure group communication

Goal: secure communication between two processes:

- both know who is on the other side (authenticated)
- both know that messages cannot be tampered with (integrity)
- both know messages cannot leak away (confidentiality)


**Authentication and Integrity**

Consider:

Trudy attacks communication from Alice to Bob.

**Authentication without integrity:**

Alice’s message is authenticated, and intercepted by Trudy, who tampers with its content, but leaves the authentication part as is.

Authentication has become meaningless

**Integrity without authentication:**

Trudy intercepts a message from Alice, and then makes Bob believe that the content was really sent by Trudy.

Integrity has become meaningless

---

**Authentication (1)**

- A sends identity to B
- B sends challenge to A
- A sends encrypted response to B
- A sends challenge to B
- B sends encrypted response to A

based on a shared secret key $K_{A,B}$
Authentication (2)

Based on a shared secret key, but using 3 messages?

Authentication (3)

Problem: The reflection attack
Authentication Using a Key Distribution Center (1)

Authentication Using a Key Distribution Center (2)

Using a ticket and letting Alice set up a connection to Bob
Authentication Using a Key Distribution Center (3)

The Needham-Schroeder authentication protocol

Authentication Using Public-Key Cryptography

Mutual authentication in a public-key cryptosystem
Digital Signatures (1)

Digital signing a message using public-key cryptography

Digital Signatures (2)

Digitally signing a message using a message digest
Secure Replicated Services

Using signatures in a group of replicated servers

General Issues in Access Control

General model of controlling access to objects
Access Control Matrix

a) Using an ACL

b) Using capabilities

Protection Domains

The hierarchical organization of protection domains
Firewall protection

Securing Code (1/4)

Java sandbox
Securing Code (2/4)

- Trusted code
- Untrusted code
- Local network
- Sandbox

Securing Code (3/4)

- Local resources accessible through objects
- Protected area
- Unprotected area

Using Java object references as capabilities
Securing Code (4/4)

stack introspection

Key Establishment

Diffie-Hellman key exchange
Key Distribution (1)

Secret-key distribution

Key Distribution (2)

Public-key distribution