Security Requirements

Security Requirements

- **Software requirements** typically about **what** the **software should do**

- We also want to have **security requirements**
  - **Security-related goals** (or **policies**)
    - **Example**: One user's bank account balance should not be learned by, or modified by, another user, unless authorized
  - **Required mechanisms for enforcing them**
    - **Example**:

      1. Users identify themselves using passwords,
      2. Passwords must be "strong," and
      3. The password database is only accessible to login program.

Typical *Kinds* of Requirements

- **Policies**
  - **Confidentiality** (and Privacy and Anonymity)
  - **Integrity**
  - **Availability**

- Supporting **mechanisms**
  - **Authentication**
  - **Authorization**
  - **Auditability**

Privacy and Confidentiality

- *Definition*: **Sensitive information not leaked** to unauthorized parties
  - Called *privacy* for individuals, *confidentiality* for data
- **Example** policy: bank account status (including balance) known only to the account owner
- Leaking **directly** or via **side channels**
  - **Example**: manipulating the system to directly display Bob's bank balance to Alice
  - **Example**: determining Bob has an account at Bank A according to shorter delay on login failure

Secrecy vs. **Privacy**? https://www.youtube.com/watch?v=Nlf7YM71k5U

Anonymity

- A specific **kind of privacy**

- **Example**: Non-account holders should be able to browse the bank informational site without being tracked
  - Here *the adversary is the bank*
  - The previous examples considered other account holders as possible adversaries

Integrity

- *Definition*: **Sensitive information not damaged** by (computations acting on behalf of) unauthorized parties

- **Example**: Only the account owner can authorize withdrawals from her account

- Violations of integrity can also be **direct** or **indirect**
  - **Example**: Being able specifically withdraw from the account vs. confusing the system into doing it

Availability

- *Definition*: A system is **responsive to requests**
- **Example**: a user may always access her account for balance queries or withdrawals
- **Denial of Service (DoS)** attacks attempt to **compromise availability**
  - by busying a system with useless work
  - or cutting off network access

Supporting mechanisms

- Leslie Lamport's **Gold Standard** defines mechanisms provided by a system to enforce its requirements
  - **Au**thentication
  - **Au**thorization
  - **Au**dit

- The gold standard is **both requirement and design**
  - The *sorts of policies* that are authorized *determines* the *authorization mechanism*
  - The *sorts of users* a system has *determines* how they should be *authenticated*

Authentication

- What is the **subject of security policies**?
  - Need to define a ***notion of identity*** and a way to ***connect an action with an identity***
    - *a.k.a. a* **principal**

- **How can system tell a user is who he says he is?**
  - What (only) he **knows** (e.g., password)
  - What he **is** (e.g., biometric)
  - What he **has** (e.g., smartphone)
  - Authentication mechanisms that employ more than one of these factors are called **multi-factor authentication**
    - E.g., bank may employ passwords and text of a special code to a user's smart phone

Authorization

- Defines **when a principal may perform an action**
- **Example**: Bob is authorized to access his own account, but not Alice's account
- There are a wide variety of **policies** that define what actions might be authorized
  - E.g., access control policies, which could be originator based, role-based, user-based, etc.

Audit

- Retain enough information to be able to **determine the circumstances of a breach or misbehavior** (or *establish one did not occur*)
  - Such information, often stored in **log files**, must be **protected from tampering**, and from access that might violate other policies

- **Example**: Every account-related action is logged locally and mirrored at a separate site

Defining Security Requirements

- Many processes for deciding security requirements

- Example: **General policy concerns**
  - Due to **regulations**/standards (HIPAA, SOX, etc.)
  - Due **organizational values** (e.g., valuing privacy)

- Example: **Policy arising from threat modeling**
  - Which **attacks** cause the **greatest concern**?
    - Who are the likely adversaries and what are their goals and methods?
  - Which **attacks** have **already occurred**?
    - Within the organization, or elsewhere on related systems?

Abuse Cases

- Abuse cases illustrate security requirements
- Where use cases describe what a system *should* do, **abuse cases describe what it should *not* do**
- Example **use case**: The system allows bank managers to modify an account's interest rate
- Example **abuse case**: A user is able to spoof being a manager and thereby change the interest rate on an account

Defining Abuse Cases

- Using attack patterns and likely scenarios, construct cases in which an **adversary's exercise of power** could **violate a security requirement**
  - Based on the threat model
  - What might occur if a security measure was removed?

- **Example**: *Co-located attacker* steals password file and learns all user passwords
  - Possible if password file is not encrypted
- **Example**: *Snooping attacker* replays a captured message, effecting a bank withdrawal
  - Possible if messages are have no *nonce*