Overview

- What is a secure computer system?
- Secure system concerns:
  - Data
    - Privacy
    - Integrity
    - Availability
  - Users
    - Authentication
    - Privacy
- Environments for Security
  - stand-alone
  - Networked
  - internetworked

What is a Secure Computer System?

- A security **policy** specifies exactly what types of actions are and are not permitted
  - Example security policy:
    - Only authorized users should be able to use the system
    - Users should not be able to read, modify, or delete other user's private files
    - The system's resources should be shared fairly among all users
- A **secure** system always obeys its security policy

Security Breaches

- A violation of a system's security policy is called a **security breach**
- Security breaches can occur:
  - Accidentally – such as when a faulty program causes the system to malfunction
  - Intentionally – such as when a malicious user discovers a way to access another user's files
- Creating a secure system in which security breaches cannot occur can be either quite easy or nearly impossible, depending on:
  - What the security policy requires
  - How the system implements the policy

Secure Computer Systems Design Principles

- The **Policy Simplicity** Principle
  - Security policy should
    - Be as simple as possible, and no simpler
    - Concisely characterize all allowed and forbidden actions
- The **System Functionality** Principle
  - A system should include as much functionality as necessary, and no more
  - It should be able to perform the job it was designed to do, but it should not carry out additional functions

The Policy Simplicity Principle - Justification

- Simplifying security policies makes them easier to
  - get right
  - reason about
  - implement
- Security breaches caused by policy shortcomings are most often due to:
  - An incomplete or inconsistent policy,
  - A misunderstanding of the policy's requirements
  - An error in its implementation

The System Functionality Principle - Justification

- Limiting system functionality reduces avenues of attack
- Security breaches caused by system functionality can be caused by:
  - Software bugs that enable attackers to cause some part of the system to malfunction
  - Unforeseen interactions between system components

Relative Security

- Few, if any, useful systems will be absolutely secure
  - Due to
    - Complexity of computer systems and
    - Limits on our ability to verify their software
- View security in a more **relative** sense
  - Examine how difficult, expensive, and dangerous the system makes it to breach security
- Note: this does **not** mean that careful design and analysis of security policies and verifying security-critical software to the best of our abilities is unimportant
  - Example: safes and padlocks

Trade-offs between Cost and Security

- The value of the items that the system is protecting determines the proper degree of difficulty, danger, and expense a system should impose to form an effective deterrent
- Trade-off between cost and security
  - Increased security only coming at the price of increased costs
- Users should be able to select systems with a level of security appropriate for their needs and only bear the costs of the security that they choose

Trade-offs between Cost and Security

- Example: user authentication
  - System A - authenticates the user sitting at a terminal every five minutes by retinal scan
  - System B - authenticates users once when they log in using a password
- System A is probably more secure than system B, but it is also probably more costly and inconvenient for users
- Is the added security and expense of system A called for?
  - Perhaps, for a corporation with millions of dollars worth of trade secrets to protect
  - Probably not, for an individual with little of value on their system

Chief Concerns of a Secure System

- Data
  - Privacy
  - Integrity
  - Availability
- Users
  - Authentication
  - Privacy
- System
  - Authentication
  - Nonrepudiation

Data Privacy

- Data **privacy** means that information access is limited to authorized entities
- Examples:
  - Certain files on the system can only be accessed by particular users
  - Communications between two users cannot be read by some third party
- Cryptography is an important technique used to protect privacy

Data Integrity

- Data **integrity** means that information can be modified only:
  - by an authorized principal and
  - to the extent of the authorization
- Examples:
  - A bank's system must ensure that only authorized bank personnel can change account balances (privacy is also a concern)
  - A company wants to make sure that its freeware program is not modified to behave maliciously (privacy is not a concern)
- Message-digest functions can be used to protect data integrity

Data Availability

- Data **availability** means that information will be accessible in a timely manner when it is needed
- Examples:
  - A non-working laptop containing a student's notes brought to an open-note exam
  - A working laptop containing a student's notes so poorly organized that the student spends the bulk of the exam searching for relevant information
- Replication and fault tolerance can be used to ensure the availability of data

User Authentication

- User **authentication** means that the system can accurately determine a user's identity
  - Many action's security depends on the identity of the person performing the action
- Examples:
  - Many files marked as readable only by their owner
  - Only certain users should be able to add or delete system accounts
- Passwords, smart cards, and biometrics can all be used to authenticate users

User Privacy

- User **privacy** means that users have some control over what information the system collects and makes available to others
- Examples:
  - Some users may not want others to know at what times and from what locations they log on, what programs they run, or with whom they are communicating
  - Users may not want to receive unsolicited commercial offers through e-mail
- Anonymity can help to protect a user's privacy

Networked and Internetworked Environments

- Additional challenges:
  - Privacy
    - Stand-alone system - the operating system is likely to control all communication channels
    - Networked systems - no host controls the communication medium and eavesdropping is usually easy
  - User authentication
    - Stand-alone system – user must be physically present to use the system
    - Internetworked systems – user may be far away and may be accessing the system over an insecure sommunications channel

Summary

- A **secure** computer system always follows the rules set forth in its security policy, which specifies exactly what actions are and are not permitted
- The principal security concerns in most systems involve protecting:
  - Data **privacy** - access to information is limited to authorized entities
  - Data **integrity** - information can be modified only by an authorized principal
  - Data **availability** - data is accessible in a timely manner when it is needed
  - User **authentication** – accurately determining user's identities
  - User **privacy** - control what information is collected and made available to others
- Additional security risks arise as systems become networked and internetworked