

Chapter 12 DESIGN PRINCIPLES

12.1 Overview

- **Specific design principles underlie the design and implementation of security mechanisms for supporting security policies.**
- **These principles build on two ideas: simplicity and restriction.**
- **Saltzer and Schroeder describe eight principles for the design and implementation of security mechanisms.**
 - Least Privilege
 - Fail-Safe Defaults
 - Economy of Mechanism
 - Complete Mediation
 - Open Design
 - Separation of Privilege
 - Least Common Mechanism
 - Psychological Acceptability
- **Simplicity**
 - Designs and mechanisms are easy to understand.
 - Less can go wrong with simple designs.
 - Reduced potential for inconsistencies within a policy or policy set.
- **Restriction**
 - Minimizes the power of an entity. The entity can access only information it needs.
 - Entities can communicate with other entities only when necessary.

Chapter 12 DESIGN PRINCIPLES

- **EXAMPLE - potential for inconsistencies:**
 - A college rule requires any teaching assistant to report cheating.
 - A different rule ensures the privacy of student files.
 - A TA contacts a student to say that some files for a program were not submitted.
 - The student tells the TA that the files are in the student's directory.
 - The TA sees the files in the directory, and notices 2 sets.
 - One set with the name beginning with "x"
 - The other set beginning with another name.
 - The TA takes the first set, and the comments show that they were written by a second student.
 - The TA takes the second set, and the comments show that they were written by a first student.
 - When the 2 sets are compared, the TA understand that they are identical except for the names in the comments.
 - The TA reports the student for cheating although concerned about a possible countercharge for violation of privacy.
 - The student charges the TA with violating his privacy by reading the first set of files.
 - Hence, the rules conflict. Which charge should be sustained?

Chapter 12 DESIGN PRINCIPLES

- **EXAMPLE – minimization of the power of an entity:**
 - Government officials are denied access to information for which they have no need.
 - They cannot communicate anything that they do not know.
- **EXAMPLE – necessity of communication with other entities:**
 - All communication with prisoners are monitored.
 - Prisoners can communicate on a list (given to the warden) through personal visit or mail.
 - Both the personal visit and mail are monitored.
 - The prisoners are prevented from receiving contraband (e.g., files).
 - There is one exception to the monitoring policy.
 - When the prisoners meet with their attorneys, the communication cannot be monitored.
 - This communication is privileged.

Chapter 12 DESIGN PRINCIPLES

12.2 Design Principles

Principle of Least Privilege

- The *principle of least privilege* states that a subject should be given only those privileges that it needs in order to complete its tasks.
- If a subject does not need an access right, the subject should not have that right.
- If a specific action requires that a subject's access rights be augmented, those rights should be discarded on completion of the action.
- In practice, most systems do not have the granularity of privileges and permissions required to apply this principle precisely.
 - The designers of security mechanisms apply this principle as best as they can.
 - In such systems, the consequences of security problems are often more severe than the consequences for systems that adhere to this principle!
- **EXAMPLE: The UNIX OS**
 - Does not apply access controls to the user *root*.
 - *Root* can terminate any process and read, write, or delete any file.
 - If a user creates backup files, they will be deleted also.
 - The administrator account on Windows has the same powers.

Chapter 12 DESIGN PRINCIPLES

Principle of Fail-Safe Defaults

- The *principle of fail-safe defaults* states that, unless a subject is given explicit access to an object, it should be denied access to that object.
- This principle requires that the default access to an object is none!
- If a subject is unable to complete its action or task, it should undo those changes made in the security state of the system before it terminates.
 - Therefore, even if the action fails, the system is still safe.
- **EXAMPLE:**
 - If the mail server is unable to create a file in the spool directory, it should close the network connection, issue an error message, and stop!
 - The mail server should not try to store the message elsewhere or to expand its privileges to save the message in another location.
 - An attacker may use the mail server's ability to overwrite other files or fill up other disks.
 - The protections on the mail spool directory itself should allow:
 - Create and write access only to the mail server
 - Read and delete access only to the local server
 - No other user should have access to the directory.
- In practice, most systems will allow an administrator access to the mail spool directory.
 - By the principle of least privilege, the administrator should be able to access only subjects and objects involved in the mail queuing and delivery.
 - The mail system can be damaged or destroyed, but nothing else can be!

Chapter 12 DESIGN PRINCIPLES

Principle of Economy Mechanism

- The *principle of economy of mechanism* states that security mechanisms should be as simple as possible.
- If a design and its implementation are simple, fewer possibilities exist for error!
 - As there are fewer components and cases that need to be tested, checking and testing processes are less complex.
- **EXAMPLE: The Identification Protocol (a.k.a., " the Ident Protocol")**
 - The TCP Client Identity Protocol Working Group of IETF is chartered to define a protocol for returning the identity of the user initiating a TCP connection.
 - When a client on host *A* initiates a TCP connection to host *B*, host *B* may query a server on host *A* to determine the identity of the client on host *A*.
 - The primary purpose of this protocol is to record the identity of requesters initiating a connection.
 - *A* (the originating host) is trustworthy.
 - If *B* wants to attack *A*, it can connect and send any chosen identity in response to the ident request.
- **Interfaces other than modules are particularly suspect.**
 - Modules often make implicit assumptions about I/O parameters or the current system state.
 - If some of these assumptions are wrong, the module's action may produce unexpected or erroneous results.
 - Interactions with external entities (e.g., programs, systems, humans) amplify this problem!

Chapter 12 DESIGN PRINCIPLES

Principle of Complete Mediation

- The *principle of complete mediation* requires that all accesses to objects be checked to ensure that they are allowed.
- If a subject attempts to read an object, the OS should mediate the action.
 - The OS first determines if the subject is allowed to read the object.
 - If the subject is allowed to read the object, the OS provides the resources for the read to occur.
 - If the subject tries to read the object again, the OS should check if the subject is still allowed to read the object.
 - Most systems would not make the second check. They would cache the results of the first check, and base the second access on the cached results.
- **EXAMPLE:**
 - When a UNIX process tries to read a file, the OS determines if the process is allowed to read the file.
 - If the process is allowed to read the file, the process receives a file descriptor encoding the allowed access.
 - Whenever the process wants to read the file, it presents the file descriptor to the kernel. The kernel then allows the access.
 - If the owner of the file disallows the process permission to read the file after the file descriptor is issued, the kernel still allows access.
 - This scheme violates the principle of complete mediation because the second access is not checked!
 - As the cache value is used, it results in the denial of access to be ineffective!

Chapter 12 DESIGN PRINCIPLES

Principle of Open Design

- The *principle of open design* states that the security of a mechanism should not depend on the secrecy of its design or implementation.
- Designers and implementers of a program must not depend on secrecy of the details of their design and implementation to ensure security.
- Others can find out such details either through technical or non-technical means:
 - Technical: disassembly and analysis
 - Non-technical: search through garbage receptacles for source code listings
- **Cryptographic software and systems are particularly vulnerable to attacks.**
 - Cryptography is a highly mathematical object.
 - Companies market cryptographic software or use cryptography to protect user data often keep their algorithms secret.
 - Experience has shown that such secrecy add little to the security of the system.
 - Cryptographic keys or passwords do not violate the principle of open design because they are not algorithms.
 - However, keeping the enciphering and deciphering algorithms would violate the principle.
- **Issues of proprietary software and trade secret complicate the application of the principle of open design.**
- **EXAMPLE: The Content Scramble Systems (CSS)**
 - The CSS is a cryptographic algorithm that protect DVD movies from unauthorized copying.
 - It has 3 levels of keys: master key, disk key, and title key.
 - The CSS was hacked in Europe in 1999.
 - A software called DeCSS appeared on many web sites.
 - The MPAA managed to shut them down successfully.

Chapter 12 DESIGN PRINCIPLES

Principle of Separation of Privilege

- The *principle of separation of privilege* states that a system should not grant permission based on a single condition.
- This principle is equivalent to the separation of duty principle.
 - The principle of separation of duty principle states that if two or more steps are required to perform a critical function, at least two different people should perform the steps.
- **EXAMPLE:**
 - Company checks for more than \$75,000 must be signed by two officers.
 - If either officer does not sign, the check is not valid.
- Similarly, systems and programs that grant access to resources should do so only when more than one condition is met.
 - This provides two things:
 - A fine-grained control over the resource
 - Additional assurance that the access is authorized
- **EXAMPLE:**
 - On Berkeley-based version of the UNIX OS, users are not allowed to change from their accounts to the root account unless two conditions are met:
 - The user knows the root password.
 - The user is in the wheel group (the group with GID 0).

Chapter 12 DESIGN PRINCIPLES

Principle of Least Common Mechanism

- The *principle of least common mechanism* states that mechanisms used to access resources should not be shared.
- Sharing resources provides a channel along which information can be transmitted. Such sharing should be minimized.
- **EXAMPLE:**
 - A web site provides e-commerce services for a major company.
 - Attackers want to deprive the company of the revenue it obtains from that web site.
 - They flood the site with bogus messages and tie up the e-commerce services.
 - Legitimate customers are unable to access the web site, and they their business elsewhere!
 - The sharing of the Internet with the attackers' sites resulted in a successful attack!
 - Some appropriate countermeasures should be able to restrict the attackers' access to the segment of the Internet connected to the web site.
 - Two techniques for restriction are:
 - Proxy servers: target suspect connections.
 - Traffic throttling: reduces the load on the relevant segment of the network indiscriminately.

Chapter 12 DESIGN PRINCIPLES

Principle of Psychological Acceptability

- The *principle of psychological acceptability* states that security mechanisms should not make the resource more difficult to access than if the security mechanisms were not present.
- Configuration and execution of a program should be as easy and as intuitive as possible, and any output should be clear, direct, and useful.
- **EXAMPLE:**
 - The SSH Secure Shell 2 protocol, or SSH2, specifies how a client can connect securely to an SSH server, and then use the resulting secure link to access the server's resources.
 - The SSH2 protocol provides the services of server authentication; encryption; data integrity verification; and client authentication.
 - Server authentication is performed using the DSA or the RSA public key algorithm.
 - For encryption and data integrity verification, a number of algorithms are provided which every SSH2 product can implement.
 - Client authentication can be performed using a password, a public key algorithm such as DSA or RSA, as well as a variety of other methods.
 - The installation and configuration mechanisms for the UNIX version allow the public key to be stored locally without any password protection.
 - In this case, the password is not needed to connect to the remote system.
 - The connection will be enciphered.
 - Hence, this mechanism satisfies the principle of psychological acceptability.

Chapter 12 DESIGN PRINCIPLES

- In practice, the principle of psychological acceptability is interpreted to mean that the security mechanism may add extra burden but that burden must be both minimal and reasonable.
- **EXAMPLE:**
 - A mainframe system allows users to place passwords on files.
 - File access requires the program to supply the password.
 - Although this mechanism violates the principle, it is considered sufficiently minimal to be acceptable.
 - However, file access is more frequent and transient on an interactive system. Hence, this requirement would be a large burden on the system.