Security Design Principles

Eight security design principles (proposed by Saltzer and Schroeder back in 1975)

1. Principle of Economy of Mechanism
2. Principle of Fail-safe Defaults
3. Principle of Complete Mediation
4. Principle of Open Design
5. Principle of Separation of Privilege
6. Principle of Least Privilege
7. Principle of Least Common Mechanism
8. Principle of Psychological Acceptability

All these 8 security principles focus on two main ideas, **Simplicity** and **Restriction.**

Simplicity: Keeping the system design and mechanisms as simple as possible to reduce complexity, which helps in minimizing errors and potential security vulnerabilities.

Restriction: Limiting access and permissions to only what is necessary, thereby reducing the risk of unauthorized access and potential damage from security breaches.

- **Principle of Economy of Mechanism** follows the **KISS** mantra: **Keep It Simple, Stupid.**

  Design systems as simple as possible to avoid complexity that can lead to security vulnerabilities.

  One big aspect of this property is to consider reusing components whenever possible. We don't need to re-implement AES or SHA-1, when there are several widely-used libraries available.

  Complex mechanisms can introduce security vulnerabilities because they often rely on assumptions about the system. If these assumptions are wrong, it can lead to security problems.

- **Principle of Fail-Safe default** says that unless a subject is given explicit access to an object, it should be denied access to the object.

  Default access to an object is deny.

We use Access Control List mechanism to enforce this policy.

UNIX permissions are represented as three triplets: owner group others

Within each triplet:
the first position is r if read access is allowed or – if it is not
the second position is w if write access is allowed or – if it is not
the third position is x if execute access is allowed or – if it is not

For example:

Bishop creates a file and sets it to belong to the group called **"vulner"**.
Bishop wants to:

Read and **write** to the file.
Allow group members (from "vulner") to **read** the file only.
Prevent anyone else from accessing the file.

What would the permission look like?

Answer: r w - r - - - - -

So, the permissions are set to let Bishop fully use the file, let the group read it, and block everyone else from accessing it.

- **Principle of Complete Mediation**: Every access to every object must be checked for authority. This means that each time an object is accessed, the system should verify that the user has the necessary permissions.

  This principle restricts the caching of information because cached data might bypass these checks, potentially leading to security vulnerabilities.

  Example 1: **UNIX file system**

  In the UNIX file system, every time a file is accessed, the system checks the user's permissions to ensure they are authorized to perform the requested operation.

  Example 2: **DNS cache poisoning**

DNS cache poisoning occurs when incorrect DNS information is inserted into the cache. If the system doesn't re-check the authority of this information each time it's used, users could be redirected to malicious sites. By always verifying the authority, such attacks can be mitigated.

- **Principle of Open Design** states that the security of a system should not depend on keeping its design or implementation secret.

  In other words, **Secrecy != Security**, and this design principle discards the concept of security by obscurity.

  Example 1: **Linux is open-source software but yet it is secure.**

  Linux's source code is publicly available, allowing anyone to examine it. Despite this openness, Linux remains secure because its security relies on well-designed mechanisms and frequent updates.

  Example 2: **The security of cryptographic algorithms should not rely on the secrecy of the algorithm, rather the secrecy of the key.**

  Cryptographic algorithms like AES or RSA are well-known and publicly studied, but they remain secure because the protection comes from the secrecy of the encryption keys, not the algorithms themselves.

- **Principle of Separation of Privilege** states that a system should not grant permission based on a single condition.

  Example1: Company checks for more than 75,000 USD must be signed by two officers of the company

  Example 2: If you forget a password (or pretend to do so), you should answer a secret question, and then also you should have access to a phone number/email address

- **Principle of Least Privilege** states that a subject should be given only those privileges that it needs in order to complete its task.

  Remember the “need to know” rule we learned. What is that?

Ans: **A person should only know that amount of information which he needs to know to perform his duty**

For example: If a subject needs to append to an object, but not to alter the information already contained in the object, it should be given append rights and not write rights.

This means that the subject can add new information to the object but cannot change or delete the existing information. This approach minimizes the risk of accidental or malicious modification of the existing data.

- **Principle of Least Common Mechanism** states that mechanisms used to access resources should not be shared. This principle is relatively harder to enforce in modern time. Why?

  Ans: **Internet is shared by everyone**

- **Principle of Psychological Acceptability** is about usability and human factors.

  Principle of Psychological Acceptability is interpreted to mean that the security mechanism may add some extra burden, but that burden must be both minimal and reasonable.

  What message should be displayed when a user supplies the wrong password during login?

  Ans: **username/password is wrong**