# **Secure Software Design Principles**

## **Document Control**

|  |  |
| --- | --- |
| Document Title:  | Secure Software Design Principles |
| Version:  | 1.0 |
| Document ID:  |  |
| Approval Date: |  |
| Owner:  | Software Security Governance Committee |
| Approver:  | Chief Information Security Officer |

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

This template is developed by KLC Consulting, Inc. <https://klcconsulting.net>

You are free to use this template under the MIT license.

## **1. Introduction**

At [Company Name], we recognize the importance of incorporating robust security measures into our software development processes. The Secure Software Design Principles outlined in this document provide a foundational framework to ensure the security integrity of our software products from conception through deployment and beyond. By adhering to these principles, we protect our systems against cybersecurity threats, protecting our data, customers, and reputation.

## **2. Scope**

This document applies to all employees, contractors, and third-party developers involved in software solutions' design, development, deployment, and ongoing maintenance at [Company Name]. It encompasses all software applications, systems, and services created or maintained for the Company.

## **3. Secure Design Principles**

### **3.1 Principle of Least Privilege**

* Objective: To ensure that every software component and user operates with only the minimum set of privileges necessary for their function.
* Implementation Guidance:
	+ Integrate Role-Based Access Control (RBAC) in all software applications to fine-tune access rights based on user roles.
	+ Conduct periodic user privileges and system permissions audits to identify and rectify deviations from the least privilege principle.

### **3.2 Principle of Defense in Depth**

* Objective: To layer multiple security controls throughout the software systems, providing redundancy in case one layer fails.
* Implementation Guidance:
	+ Implement varied security measures, including but not limited to application firewalls, encryption, intrusion detection/prevention systems, and anomaly detection algorithms.
	+ Design systems with network segmentation in mind, isolating critical components to minimize potential breach impacts.

### **3.3 Principle of Fail-Safe Defaults**

* Objective: To configure systems to deny by default, granting permissions only as explicitly required.
* Implementation Guidance:
	+ Ensure all system configurations are set to their most secure settings by default, requiring explicit authorization for access or operations.
	+ Design error handling and exception management to fail securely, preventing default behaviors that could expose vulnerabilities.

### **3.4 Principle of Economy of Mechanism**

* Objective: To simplify the design of software systems, reducing complexity and thus the attack surface.
* Implementation Guidance:
	+ Strive for elegant, straightforward solutions, avoiding unnecessary complexity that could obscure potential vulnerabilities.
	+ Regularly review and refactor codebases to simplify architectures and eliminate redundant or obsolete code paths.

### **3.5 Principle of Complete Mediation**

* Objective: To thoroughly authenticate and authorize every access request to system resources.
* Implementation Guidance:
	+ Ensure all access requests are validated against the latest security policies, avoiding reliance on cached permissions or unchecked inputs.
	+ Implement comprehensive logging of access attempts, particularly for sensitive operations, to enable effective monitoring and forensic analysis.

### **3.6 Principle of Open Design**

* Objective: To base the system's security on publicly reviewed and tested standards rather than the obscurity of design or implementation.
* Implementation Guidance:
	+ Utilize well-established, open security protocols and cryptographic standards, ensuring that security mechanisms are transparent and vetted.
	+ Encourage peer reviews and external audits of the system's security architecture to validate its effectiveness.

### **3.7 Principle of Separation of Duties**

* Objective: To distribute responsibilities among multiple parties to reduce the risk of unauthorized actions.
* Implementation Guidance:
	+ Design software and processes that require collaborative verification for sensitive operations, ensuring no single point of failure in the security chain.
	+ Enforce stringent access controls and role definitions within the software, clearly delineating permissions associated with different functions.

### **3.8 Principle of Least Common Mechanism**

* Objective: To minimize the sharing of components among different users and processes.
* Implementation Guidance:
	+ Isolate shared components and services, providing them only as necessary and ensuring they do not become a single point of compromise.
	+ Leverage containerization and other isolation techniques to segregate processes, reducing the risk of one process affecting another.

### **3.9 Principle of General Acceptability**

* Objective: To ensure user-friendly security mechanisms do not deter users from secure practices.
* Implementation Guidance:
	+ Design intuitive security interfaces and prompts that do not disrupt the user experience, encouraging secure interactions.
	+ Offer clear guidance and documentation on security features, making it easy for users to understand and comply with security requirements.

### **3.10 Principle of Trustworthiness**

* Objective: To ensure all software components are dependable and function correctly under all expected conditions.
* Implementation Guidance:
	+ Establish a robust CI/CD pipeline incorporating security testing to continuously validate the trustworthiness of the software.
	+ Schedule regular security audits and engage in third-party vulnerability assessments to verify the integrity of software components.

### **3.11 Principle of Security by Design**

* Objective: To integrate security considerations throughout the SDLC, from the initial design phase to deployment and maintenance.
* Implementation Guidance:
	+ Begin each project with a thorough threat modeling exercise to identify potential security issues and integrate mitigations into the design.
	+ Adhere to secure coding guidelines and perform extensive security testing at every stage of development, ensuring security is an inherent aspect of the software.

## **4. Compliance**

All personnel and third parties associated with [Company Name] must adhere to these Secure Software Design Principles. Non-compliance may lead to disciplinary actions, including termination of employment or contracts, and must be addressed promptly to maintain the integrity of our software systems.

## **5. Governance**

This Standard is overseen by the Software Security Governance Committee, responsible for its enforcement, periodic review, and updates. Amendments to these principles require formal approval. Constructive feedback and recommendations for improvement are welcomed and should be directed to [Contact Information].