UTT
UNIVERSIDAD TECNOLÓGICA DE TIJUANA
GOBIERNO DE BAJA CALIFORNIA

Actividad:

Secure Coding Principles Specification

Estudiante:

Ángel Amaury Tienda Lezama

Grupo:

10B

Materia:

Desarrollo movil Integral

Docente:

Ray Brunett Parra Galaviz

Tijuana, Baja California, 15 de Enero del 2025

Introducción

La codificación segura es un conjunto de prácticas y principios destinados a prevenir vulnerabilidades en el software y mitigar riesgos de seguridad. La importancia de una codificación segura radica en proteger la confidencialidad, integridad y disponibilidad de los datos y sistemas, garantizando que sean resistentes a ataques maliciosos. Este enfoque debe estar integrado desde las primeras etapas del desarrollo para reducir costos asociados a la corrección de vulnerabilidades en fases posteriores.

Los principios de codificación segura incluyen varias prácticas fundamentales que guían a los desarrolladores para construir aplicaciones seguras y confiables:

- **Validación de entradas:** Todas las entradas de usuarios deben validarse antes de ser procesadas para evitar ataques como inyección de código SQL, cross-site scripting (XSS) y buffer overflow. Se deben emplear listas blancas, límites estrictos y técnicas de sanitización.
- **Autenticación y gestión de sesiones:** Implementar mecanismos robustos para autenticar usuarios y manejar sesiones. Esto incluye el uso de contraseñas seguras, autenticación multifactor (MFA) y expiración de sesiones inactivas. Además, es fundamental evitar exponer datos sensibles en tokens o cookies.
- **Cifrado de datos sensibles:** Los datos sensibles, tanto en tránsito como en reposo, deben estar protegidos mediante cifrado. Protocolos como TLS/SSL se usan para asegurar la comunicación, mientras que algoritmos robustos como AES garantizan la seguridad de la información almacenada.
- **Principio de privilegios mínimos:** El acceso a recursos y funcionalidades debe restringirse al nivel mínimo necesario para completar una tarea. Esto reduce la superficie de ataque y limita el impacto de posibles vulneraciones.
- **Control de errores y excepciones:** Los mensajes de error no deben revelar información interna del sistema, como estructuras de bases de datos o rutas de archivos. Los errores deben gestionarse de manera controlada y registrada para análisis interno.
- **Protección contra vulnerabilidades comunes:** Utilizar herramientas de análisis de código estático para detectar vulnerabilidades conocidas, como inyecciones, desbordamiento de memoria o uso de bibliotecas desactualizadas. Actualizar regularmente las dependencias y bibliotecas utilizadas.
- **Defensa en profundidad:** Implementar múltiples capas de seguridad en la arquitectura del software. Por ejemplo, combinar autenticación segura, firewalls, controles de acceso y cifrado para dificultar el avance de un atacante.
- **Evitar la codificación insegura:** Evitar el uso de funciones inseguras o desactualizadas en el código, como `strcpy()` en C, que puede generar desbordamientos. Usar alternativas modernas y seguras proporcionadas por los lenguajes y frameworks.
- **Registro y monitoreo:** Implementar un sistema robusto de logs que registre actividades críticas del sistema, como intentos de autenticación fallidos o accesos a recursos protegidos. Esto facilita la detección temprana de ataques y la investigación posterior.
- **Educación y formación continua:** Los desarrolladores deben estar al día con las últimas amenazas de seguridad y mejores prácticas. Participar en capacitaciones, como las ofrecidas por OWASP, es esencial para mantener altos estándares de seguridad.

Conclusión

La especificación de principios de codificación segura es fundamental para desarrollar software confiable y resistente a amenazas. Al adoptar estas prácticas desde las etapas iniciales del desarrollo, se minimizan riesgos, costos y posibles impactos negativos en los sistemas y datos. La seguridad no es un producto final, sino un proceso continuo que debe ser revisado, actualizado y reforzado con cada ciclo de desarrollo.

Referencias

Tes. (2024b, enero 15). *Estándares de codificación segura: aplicación de prácticas de codificación segura con SAST*. Parasoft. https://es.parasoft.com/blog/secure-coding-standards-enforcing-secure-coding-practices-with-sast/

Prácticas de Codificación Segura - Guía rápida de referencia | Prácticas de Codificación Segura | OWASP Foundation. (s. f.-b). https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/stable-es/01-introduction/05-introduction

Michali. (2022b, julio 6). *What is Secure Coding?* Check Point Software. https://www.checkpoint.com/es/cyber-hub/cloud-security/what-is-secure-coding/