TECNOLÓGICA DEL ORIENTE
INSTITUCIÓN DE EDUCACIÓN SUPERIOR

SEGURIDAD EN SOFTWARE

NO REPUDIO Y TRAZABILIDAD

NO REPUDIO Y TRAZABILIDAD

En el contexto de la seguridad en software, los principios de no repudio y trazabilidad son esenciales para garantizar la responsabilidad en el ciclo de vida del desarrollo y uso de aplicaciones. Ambos conceptos permiten asegurar que las acciones realizadas sobre un sistema como modificaciones al código, aprobaciones, despliegues o accesos puedan ser atribuibles, verificables y auditables, promoviendo así un entorno confiable y seguro (Escrivá Gascó, 2013).

No repudio en el desarrollo y uso del software

El no repudio garantiza que ninguna persona involucrada en el desarrollo o uso de un sistema pueda negar haber ejecutado una acción determinada. En entornos de desarrollo, este principio es clave para:

- Asegurar que los commits de código estén firmados digitalmente.
- Validar que los despliegues en ambientes de producción sean autorizados por personal identificado.
- Confirmar que las solicitudes de cambios o correcciones provienen de fuentes legítimas.

Ejemplo: En un repositorio Git, cada contribución al código fuente puede estar firmada digitalmente. Esto impide que un desarrollador niegue su autoría si luego se detecta una vulnerabilidad en ese fragmento de código.

Trazabilidad en el ciclo de vida del software

La trazabilidad permite registrar y seguir el rastro de todos los eventos relevantes durante el desarrollo, pruebas, despliegue y mantenimiento del software (Costas Santos, 2015). Esto incluye:

- Cambios en el código fuente.
- Historial de aprobaciones de revisiones.
- Auditoría de accesos al sistema y a las bases de datos.
- Registro de incidentes y correcciones aplicadas.

Ejemplo: En una plataforma CI/CD (Integración y Entrega Continua), cada paso de construcción y despliegue queda registrado: quién lo aprobó, qué versión se compiló, en qué entorno se desplegó, y cuándo ocurrió. Esto permite revisar el historial en caso de un fallo o vulnerabilidad.

Beneficios directos para la seguridad del software

- **Prevención de fraudes y sabotajes:** Al poder demostrar quién hizo qué y cuándo, se desincentiva el comportamiento malicioso.
- **Resiliencia ante incidentes:** En caso de una brecha de seguridad, los registros permiten identificar rápidamente el origen del problema.
- **Cumplimiento normativo:** Muchas regulaciones exigen mecanismos de trazabilidad y no repudio para aplicaciones críticas.

Buenas prácticas en entornos de desarrollo seguro

- Uso de firmas digitales en commits y despliegues.
- Activación de logs seguros y centralizados.
- Integración de sistemas de auditoría en herramientas como GitLab, Jenkins, Jira o SonarQube.
- Control estricto de acceso a los entornos de desarrollo, prueba y producción.
- Conservación y análisis periódico de los registros de actividad.

Principio de mínimo privilegio y defensa en profundidad

En el ámbito de la seguridad del software, los principios de mínimo privilegio y defensa en profundidad actúan como pilares fundamentales para la protección de sistemas frente a accesos no autorizados, fallos humanos y ataques maliciosos. Ambos conceptos son complementarios y contribuyen a reducir significativamente el impacto de las vulnerabilidades, fortaleciendo la arquitectura general del software (Costas Santos, 2015).

Principio de mínimo privilegio

El principio de mínimo privilegio establece que cada usuario, proceso o componente del sistema debe operar únicamente con los permisos estrictamente necesarios para realizar sus funciones. Esta limitación reduce la superficie de ataque y evita que errores o acciones malintencionadas escalen a niveles críticos del sistema.

Ejemplo en desarrollo de software:

Si una aplicación tiene un módulo que solo necesita leer datos de una base de datos, dicho módulo debe utilizar un usuario con permisos de solo lectura. Otorgar acceso de escritura innecesario podría permitir que, si el módulo es vulnerado, un atacante modifique o elimine datos.

Ejemplo en usuarios finales:

Un empleado que solo necesita consultar reportes no debería tener acceso al panel de configuración del sistema. Restringir sus privilegios evita que realice cambios que comprometan la seguridad o el funcionamiento del software.

Defensa en profundidad

El principio de defensa en profundidad se basa en la implementación de múltiples capas de seguridad a lo largo de toda la arquitectura del sistema. La idea es que si una barrera es superada, otra capa puede contener o mitigar el ataque (Escrivá Gascó, 2013). Este enfoque no depende de una única medida de seguridad, sino de un conjunto coordinado de mecanismos preventivos, detectivos y correctivos.

Ejemplo práctico:

En una aplicación web crítica, una defensa en profundidad incluiría:

- Autenticación robusta (como 2FA).
- Validación de entradas en el backend.
- Firewalls de aplicación (WAF).
- Registro de auditorías.
- Cifrado de datos en tránsito y en reposo.
- Monitoreo activo de eventos de seguridad.

Aunque una capa pueda ser vulnerada (por ejemplo, una contraseña débil), otras medidas como el monitoreo de accesos o la autenticación multifactor podrían evitar una brecha mayor.

Interacción entre ambos principios

La combinación del mínimo privilegio y la defensa en profundidad permite no solo reducir los vectores de ataque, sino también limitar el daño potencial en caso de una intrusión (Escrivá Gascó, 2013). Por ejemplo, si un atacante logra explotar una vulnerabilidad en un componente de bajo nivel que opera con permisos restringidos, su acceso estará limitado. Y gracias a la existencia de otras capas de defensa, como los registros de actividad o el aislamiento de procesos, se podrá detectar y contener el ataque con rapidez.

Buenas prácticas aplicables

- Diseñar perfiles de usuarios específicos con permisos segmentados.
- Separar funciones críticas en distintos componentes del sistema.
- Utilizar entornos aislados (sandboxing) para probar código inseguro.
- Aplicar cifrado, monitoreo y controles de acceso redundantes.
- Configurar alertas automáticas en caso de intentos de escalación de privilegios.