GRUPO DE SEGURIDAD INFORMÁTICA

Fundamentos de la Seguridad Informática

Principios de diseño seguro

Principios de diseño seguro

- Minimizar el área de ataque
- Establecer configuraciones por defecto seguras
- Principio de menor privilegio
- Principio de defensa en profundidad
- Fallar de forma segura

Principio de diseño seguro (2)

- No confiar en servicios de terceros completamente
- Separación de responsabilidades
- Evitar seguridad por obscuridad
- Mantener las seguridad simple
- Arreglar los problemas de seguridad correctamente

Minimizar el área de ataque

- Cada funcionalidad agrega una cierta cantidad de riesto a la aplicación.
- Reducir el riesgo general por medio la reducción del área de ataque

Establecer seguridad por defecto

- Por defecto, la experiencia de uso de una aplicación debería ser segura. Debería ser el usuario el que pudiera habilitar funcionalidades adicionales (si tiene permiso)

Principio de mínimo privilegio

- Las cuentas deben tener el menor privilegio requerido para realizar los procesos de negocio.

Defensa en profundidad

- Donde un control es razonable, más controles que piensan en los riesgos de differentes formas son mejores.
- Los controles, cuando se usan en profundidad, pueden hacer que las vulnerabilidades graves sean extraordinariamente difíciles de explotar

Fallar de forma segura

- Las aplicaciones fallan de forma regular en procesar transacciones por diversas razones
- Fallar no debe dar al usuario privilegios adicionales, y no debería devolver información sensible no necesaria (ej, logs, DB queries)

No confiar en servicios de terceros completamente

- Muchas aplicaciones usan servicios de terceros para obtener datos adicionales
- El principio dice que no se debe confiar en estos servicios desde el punto de vista de la seguridad
- Quiere decir que debe verificarse la validez de los datos enviados por el tercero y no asignarle altos niveles de privilegio a dichos servicios dentro de la aplicación.

Separación de responsabilidades

- Un control clave para evitar fraudes es separar las responsabilidades. Por ejemplo, alguien que solicita una nueva computadora no puede además firmar para habilitarlo, y no debería recibirla en forma personal
- Los administradores no deberían ser usuarios de la aplicación. Deberían poder cambiar políticas de contraseñas, pero no comprar bienes como si fueran otros usuarios, por ejemplo

Seguridad por oscuridad

- Es un control débil y casi siempre falla cuando es el único
- La seguridad de la aplicación no debe depender de mantener un secreto como una URL oculta
- No confundir con tener una clave, que es un secreto y debe mantenerse en secreto

Mantener la seguridad simple

- La superficie de ataque y la simplicidad van de la mano

Arreglar correctamente los problemas de seguridad

- Luego de identificado un problema es importante crear un test para encontrar la causa
- Si se han usado patrones de diseño, ese problema puede estar esparcido a lo largo del código
- Escribir un test garantiza que no haya regresiones

Bibliografía y material de referencia

- OWASP Developer Guide, https://owasp.org