En este documento se hará un estudio sobre el último informe de la OWASP acerca de las 10 vulnerabilidades más comunes en las páginas webs, que data del 2025.

A01 Perdida de control de acceso

Manteniendo su primer puesto como en años anteriores, la pérdida del control de acceso es la vulnerabilidad más común. Según el estudio de la OWASP, el 100% de las páginas web tienen algún fallo relacionado con este aspecto.

El control de acceso obliga ciertas conductas para que los usuarios no puedan actuar más allá de sus permisos. Alguna de las vulnerabilidades más comunes:

- Violación del principio del último privilegio, donde el acceso solo debería ser dado para capacidades, roles o usuarios particulares; sin embargo, es accesible para todos.
- Bypassing el control de acceso mediante la modificación de la URL, el estado interno de la aplicación, de la página HTML o con una herramienta que modifica las peticiones API
- Permitiendo la visualización o edición la cuenta de un tercero por dar su identificador único (referencias a objetos inseguras)
- Una API accesible sin controles de acceso para POST, PUT y DELETE

Recomendaciones de cómo prevenir algunas de las vulnerabilidades.

- Salvo por los recursos públicos, deniega por defecto
- Implementar mecanismos de control de acceso una vez u rehusarlos en el resto de la aplicación, minimizando el uso del CORS
- El control de acceso debe implementar su cumplimiento a nivel de dato y no permitir que el usuario pueda crear, leer, actualizar o borrar cualquier dato.
- Registrar los fallos de control de acceso cuando sean necesarios (muchas veces el mismo fallo)

Para ver más peligros, más formas de prevenirlos y ejemplos de escenarios de ataque sobre el control de acceso:

https://owasp.org/Top10/2025/A01_2025-Broken_Access_Control/

A02 Fallos en la configuración de seguridad

Esta vulnerabilidad comparte con la primera que todas las webs testeadas por la OWASP han tenido alguno fallo de configuración.

Alguna de las vulnerabilidades posibles por una mal configuración:

- Hay características innecesarias instaladas o incluso habilitadas (puertos, servicios, cuentas, privilegios, etc)
- Las cuentas por defecto y sus contraseñas están habilitadas y no modificadas.
- Falta de "security hardening" en alguna parte de la aplicación o permisos mal configurados en los servicios de la nube
- Para sistemas actualizados, las ultimas características de seguridad están desactivadas o no configuradas de forma segura

Algunas medidas de prevención:

- Entornos de Desarrollo, QA y producción deben de estar configurados de forma idéntica, pero con distintas credenciales
- Una plataforma minimalista sin componentes ni características innecesarias. (Desinstalar todo aquello que sea innecesario)
- Enviar directivas de seguridad a los clientes

https://owasp.org/Top10/2025/A02_2025-Security_Misconfiguration/

A03 Fallo en la cadena de suministro de software

Son fallos en la cadena de suministro de software u otros compromisos durante el proceso de construcción. Normalmente, se producen por vulnerabilidades en código third-party, herramientas u otras dependencias de los que el sistema depende.

Alguna de las vulnerabilidades:

- No tienes un seguimiento de las versiones de tus componentes.
- El software es vulnerable, sin soporto o desactualizado.
- No escaneas para buscar vulnerabilidades regularmente

Algunas medidas de prevención:

- No solo hagas seguimiento de tus dependencias directas, sino de las dependencias de tus dependencias
- Elimina todo lo que no uses (dependencias, archivos, documentacion...)
- Consulta regularmente las versiones de los componentes tanto del lado del servidor como del lado del cliente, y sus dependencias.
- Solo descargar componentes de links fiables y de fuentes confiables.

Para ver más peligros, más formas de prevenirlos y ejemplos:
https://owasp.org/Top10/2025/A03_2025-Software_Supply_Chain_Failures/

A04 Fallos criptográficos

Es necesario determinar qué información necesita ser encriptada, y cuál necesita ser extra encriptada. Ejemplos evidentes son las contraseñas o las tarjetas de crédito. Ejemplos menos evidentes son los datos médicos de uno.

Algunos conceptos a los que prestar atención:

- Buscar si hay algoritmos criptográficos viejos o débiles que son usados por defecto o en código viejo
- Hay claves por defecto en uso, se generan claves criptográficas débiles o claves reutilizadas.
- La encriptación no es forzada o faltan cabeceras.
- El certificado del servidor recibido y la cadena de confianza son debidamente validadas

Algunas medidas de prevención:

- Clasifica y etiqueta los datos procesados, guardados o transmitidos por la aplicación.
- Guarda las claves más importantes en un HSM
- Siempre que sea posible, implementa algoritmos criptográficos confiables.

Para ver más peligros, más formas de prevenirlos y ejemplos:

https://owasp.org/Top10/2025/A04_2025-Cryptographic_Failures/

A05 Inyección

La inyección es un fallo en la aplicación que permite que los inputs de un usuario no seguro sean enviados a un intérprete y que este ejecute parte de esos inputs como comandos

Una aplicación es vulnerable a un ataque cuando :

- Los datos suministrados por el usuario no son validados, filtrados o saneados por la aplicación
- consultas dinámicas o llamadas sin parámetros son usadas directamente por el intérprete
- Datos no saneados son usados en el la búsqueda de parámetros del mapeo de objetos relacionales (ORM) para extraer registros adicionales y sensibles.
- Datos potencialmente hostiles son usados directamente o concatenados. El SQL o el comando contiene la estructura y los datos maliciosos en consultas dinámicas, comandos o procedimientos.

Como prevenirlos:

- La opción preferida es usar una API segura que evita el uso de intérpretes completamente, suministra una interfaz parametrizada o migra para ORMS.
- Hacer validación en el lado del servidor
- Si quedan consultas dinámicas, neutralizar los caracteres especiales usados en la sintaxis de los interpretes

Para ver ejemplos de ataques: https://owasp.org/Top10/2025/A05_2025-Injection/

A06 Diseño inseguro

En esta categoría entran todas las vulnerabilidades que en las que “falta o el control de diseño es ineficiente”.

Formas genéricas de prevenirlos:

- Establecer y usar un ciclo de vida de desarrollo seguro con profesionales que te ayuden a evaluar y diseñar los controles de seguridad y de privacidad.
- Establecer y usar una librería de patrones de diseño seguros.
- Usar un modelado de amenazas para las partes críticas de la aplicación como la autenticación, el control de acceso y lógica de negocio.
- Usar un modelado de amenazas como una herramienta educativa para generar una mentalidad sobre la seguridad.
- Integrar lenguaje de seguridad y controles sobre la historia de usuarios

Ejemplos: https://owasp.org/Top10/2025/A06_2025-Insecure_Design/

A07 Fallos de autenticación

https://alvarogargon.ieslossauces.es/AGGCIBProyectoCIB/doc/OWASP07.pdf

A08 Fallos en la integridad de los datos o de software

Son fallos en la programación y la infraestructura que no protegen contra código o datos inválidos ya que son tratados como válidos y confiables. Un caso típico de esta vulnerabilidad es cuando una aplicación depende de plugins/librerías/módulos que provienen de fuentes no verificadas.

Como prevenirlo:

- Usar firmas digitales o mecanismos similares para verificar el software o los datos provienen de la fuente esperada y sin alterar
- Asegúrate que las librerías y dependencias solo están consumiendo de repositorios confiables.
- Asegúrate que hay un proceso de revisión para los cambios del código y la configuración para minimizar las probabilidades de que código o configuración sea introducido en tu software

Ejemplos de ataques: https://owasp.org/Top10/2025/A08_2025-Software_or_Data_Integrity_Failures/

A09 Fallos en el registro y en el monitoreo

A10 Fallos en el manejo de condiciones excepcionales