OWASP: TOP 10 VULNERABILIDADES PRINCIPALES

ALEJANDRO DE LA HUERGA FERNANDEZ

FUNDAMENTOS DE CIBERSEGURIDAD

10/01/2026

Tabla de contenido

OWASP (Open Web Application Security Project)

OWASP es un proyecto de **código abierto** el cual su función consta de determinar y compartir las **causas** que hacen que el **software** sea **inseguro**.

La **fundación OWASP** es una fundación sin ánimo de lucrio que apoya y gestiona los proyectos e infraestucturas de OWASP.

Projectos en los que trabaja OWASP

Como organización OWASP trabaja en diversos projectos relacionados con la ciberseguridad y la seguridad informática colaborando asi en la difusión y elaboración de sitios web seguros.

Dentro de los diferentes proyectos en los que trabaja OWASP se pueden categorizar en 3 bloques bien diferenciados:

- **Proyectos emblemáticos**
- **Proyectos de producción**
- **Otros proyectos**

Proyectos emblemáticos

- **OWASP Amass**: Framework que ayuda a los profesionales de la seguridad de la información a realizar el mapeo de redes de las superficies de ataque y descubrimiento de activos externos.
- **Norma de verificación de seguridad de aplicaciones:** Marco de requisitos de seguridad que se centrar en definir los controles de seguridad requeridos al diseñar, desarrollar y probar aplicaciones web y servicios web modernos.
- **OWASP Defectdojo:** Herramienta líder de gestión de vulnerabilidades de aplicaciones de código abierto creada para DevOps y la integración continua de seguridad.

Proyectos de producción:

- **Proyecto de seguridad de la API de OWASP**: El proyecto API Security se centra en estrategias y soluciones para comprender y mitigar las vulnerabilidades únicas y los riesgos de seguridad de las interfaces de programación de aplicaciones (API).
- **Firewall de aplicaciones web de OWASP Coraza**: OWASP Coraza es un marco WAF de nivel empresarial de golang compatible con Modsecurity y OWASP Core Ruleset.

OWASP TOP 10 2025

El OWASP Top 10 es un documento de concienciación estándar para desarrolladores y seguridad de **aplicaciones web**. Representa un amplio consenso sobre los **riesgos de seguridad más críticos para las aplicaciones web.**

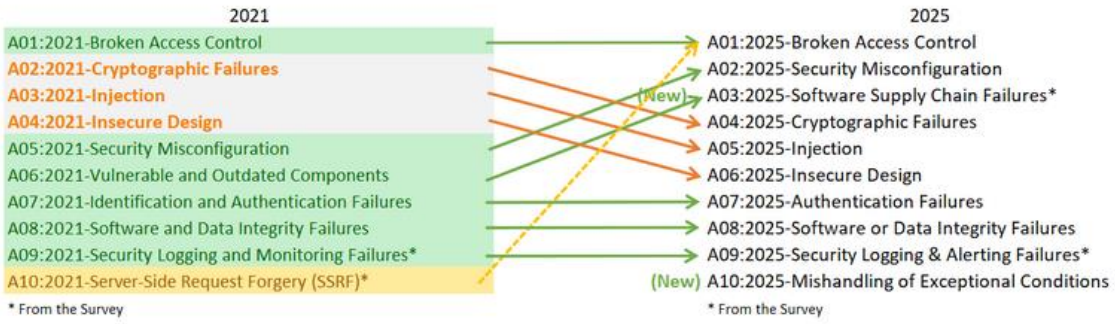
Este 2025 será la **octava entrega** de dicho documento por parte de OWASP.

Presentación del TOP 10 2025 OWASP:

1. Control de acceso roto.
2. Configuración incorrecta de seguridad.
3. Fallas en la cadena de suministro del software.
4. Fracasos criptográficos.
5. Inyección.
6. Diseño Inseguro.
7. Fallas de autenticación.
8. Fracasos de integridad de software o datos.
9. Registro de seguridad y fallas de alertas.
10. Mal manejo de condiciones exceptionales.

Cambios producidos en el Top 10 para 2025:

Se han agregado **dos nuevas categorías** y una consolidación para el top 10 en este 2025, también se ha trabajado en mantener el **enfoque** en el **problema raíz** categorizando así las diferentes vulnerabilidades que aparecen en el top 10.



La siguiente representa los cambios realizados este 2025 a diferencia del 2021 en el top 10.

Análisis detallado Top 10 OWASP:

En el siguiente apartado vamos a ver un **análisis detallado** de cada una de las vulnerabilidades que componen el top 10 de OWASP para este 2025.

A01:2025 – Control de acceso roto

Manteniendo su posición en el top 1 respecto al pasado año. Los datos aportados indican que el promedio, el **3,73%** de las aplicaciones probadas tenían una o mas de las 40 vulnerabilidades comunes en cuanto a esta categoría se refiere, entre estas dichas vulnerabilidades comunes encontramos:

- **CWE-200:** Exposición de información confidencial a un actor no autorizado.
- **CWE-201:** Exposición de información confidencial a través de datos enviados.
- **CWE-918 (SSRF):** Falsificación de solicitudes del lado del servidor.
- **CWE-352**: Falsificación de petición en sitios cruzados.

COMO PREVENIR: El control de acceso solo es efectivo cuando se implementa en código del lado del servidor o en API sin servidor de confianza, donde el atacante no puede modificar la comprobación del control de acceso o los metadatos.

- Implementar los mecanismos de control de acceso una sola vez y reutilizarlos a lo largo de toda nuestra aplicación web.
- Desactivar la lista de directorios del servidor web y asegurarnos que los **metadatos** y los archivos, por ejemplo **“.git”** junto con los archivos de **copia de seguridad** no estén presentes dentro de las raíces de la web.
- Desactivar los identificadores de sesión deben invalidarse en el servidor después de la conexión.
- Utilizar frameworks o patrones bien establecidos que proporcionen controles de acceso simples y declarativos.

EJEMPLO DE ATAQUE:

Cuando la aplicación utiliza datos no verificados en una llamada SQL que esta accediendo a la información de la cuenta:

```
pstmt.setString(1, request.getParameter("acct"));
ResultSet results = pstmt.executeQuery( );
```

El atacante puede simplemente modificar el parámetro ‘acct’ del navegador para enviar cualquier número de cuenta deseado.

```
https://example.com/app/accountInfo?acct=notmyacct
```

Página Control de Acceso OWASP

A02:2025 – Configuración incorrecta de seguridad

Subiendo del puesto número 5 en el año 2021 al puesto número 2 en este 2025. Las configuraciones erróneas son más frecuentes en este 2025 ya que los datos aportados indican que un 3% de las aplicaciones probadas tenían una o mas de las 40 vulnerabilidades comunes:

- **Configuración CWE-16.**
- **CWE-611: Restricción incorrecta.**

Esto se produce cuando un sistema, aplicación o servicio en la nube se configura incorrectamente desde una perspectiva de seguridad, creando vulnerabilidades.

COMO PREVENIR: Se deben de implementar procesos de instalación seguros que incluyan:

- Configuración de los entornos de desarrollo, control de calidad y producción de manera idéntica con diferentes credenciales utilizadas en cada entorno.
- Retirar o eliminar los marcos no utilizados.
- Una arquitectura de aplicación segmentada proporciona una separación efectiva y segura entre componentes.
- Agregar de forma proactiva una configuración central para interceptar mensajes de error.

EJEMPLOS DE ATAQUE:

El servidor de aplicaciones viene con aplicaciones de ejemplo no eliminadas del servidor de producción, estas aplicaciones de ejemplo vienen con defectos de seguridad que los atacantes utilizan para comprometer el servidor. El atacante encuentra y descarga las clases Java compiladas que luego utiliza para hacer ingeniería inversa.

Página Configuración incorrecta OWASP

A03:2025 – Fallos en la cadena de suministro del software

Esta fue la vulnerabilidad mejor clasificada en la encuesta de la comunidad TOP 10 con un porcentaje del 50%. Apareciendo por primera vez en el top 10 de 2013 como A9.

El uso de componentes con vulnerabilidades conocidas ha crecido en cuento al riesgo ya que incluye todos los fallos producidos en la cadena de suministro. Entre las vulnerabilidades más comunes que afectan a este término tenemos:

- **CWE-477:** Uso de una función obsoleta.
- **CWE- 1104:** Uso de componentes de terceros no mantenidos.
- **CWE-1329:** Dependencia de componentes que no es actualizable.
- **CWE- 1395:** Dependencia de componentes de terceros vulnerable.

Los fallos de la cadena de suministro de software son averías u otros compromisos en el proceso de construcción, distribución o actualización del software.

COMO PREVENIR: Para poder prevenirlo debe de haber un proceso de gestión de parches para:

- Generar y gestionar de forma centralizada la lista de materiales del software.
- Rastrear tanto las dependencias directas como de terceros o transitivas.
- Reducir la superficie de ataque eliminado dependencias no utilizadas.
- Obtener únicamente componentes de fuentes oficiales.

EJEMPLOS DE ATAQUE:

Un proveedor de confianza se ve comprometido con el malware, lo que lleva a que sus sistemas informáticos se vean comprometidos cuando se actualizan, un ejemplo claro de todo esto es:

El compromiso de 2019 de SolarWinds llevó a que ~ **18,000 organizaciones se vieran comprometidas**

https://www.npr.org/2021/04/16/985439655/a-worst-nightmare-cyberattack-the-untold-story-of-the-solarwinds-hack

Página fallos en la cadena de suministro OWASP

A04:2025 – Fallos Criptográficos

Esta vulnerabilidad desciende del puesto número dos al cuatro teniendo en cuenta la versión de 2021. Los datos aportados indican que, en promedio, el **3,80%** de las aplicaciones tienen una o mas de las vulnerabilidades comunes de esta categoría.

Esta vulnerabilidad se basa en que todos los datos de transito deben **cifrarse en la capa de transporte (Capa 4 del OSI).**

- **CWE - 261:** Codificación débil para la contraseña.
- **CWE - 319:** Transmisión de información sensible de texto claro.
- **CWE - 322:** Intercambio de claves sin autenticación de entidad.
- **CWE - 327:** Uso de un algoritmo criptográfico roto o arriesgado.

COMO PREVENIR: Hacer lo siguiente como mínimo:

- Clasificar y etiquetar los datos procesados, almacenados o transmitidos por una aplicación.
- Almacenar las **claves mas sensibles** en un hardware o HSM basado en la nube.
- Asegurarnos de cifrar todos los datos confidenciales en reposo.
- No utilizar **protocolos no cifrados** como FTP y STARTTLS.
- Deshabilitar el almacenamiento en caché para las respuestas que contienen datos confidenciales.

EJEMPLOS DE ATAQUE:

Un sitio no utiliza **no aplica TLS** para todas las páginas o admite un cifrado débil. Un atacante monitorea el tráfico de red (por ejemplo, en una red inalámbrica insegura), degrada las conexiones de HTTPS a HTTP, intercepta solicitudes y **roba la cookie de sesión del usuario**.

El atacante **reproduce esta cookie** y secuestra la sesión (autenticada) del usuario.

Página Fallos Criptográficos OWASP

A05:2025 - Inyección

Esta vulnerabilidad desciende del puesto número 3 al 5 respecto a la versión de 2021. Esta vulnerabilidad es una de las categorías mas probadas con **el 100% de las aplicaciones probadas para algún tipo de Inyección.**

Tiene el mayor número de vulnerabilidades comunes para cada categoría: Mas de **14k CWEs para inyección SQL** y por encima de los **30k CWEs para Cross-Site-Scripting**.

- **CWE - 74:** Validación de entrada inadecuada.
- **CWE - 89:** Neutralización incorrecta de elementos especiales utilizando una consulta SQL.
- **CWE - 80:** Neutralización incorrecta de etiquetas HTML relacionadas con scripts en una página web (XSS básico).

COMO PREVENIR: Los mejores medios para prevenir la inyección requieren mantener los datos alejados lo máximo posible de los comandos y consultas.

- Una de las mejores opciones es usar una API segura, que evite usar el intérprete por completo.

Cuando no es posible separar los datos de los comandos, puede reducir las amenazas utilizando las siguientes técnicas.

- Utilizar una buena validación de entrada del lado del servidor.
- Para cualquier consulta residual, elimine los caracteres especiales utilizando sintaxis bloqueante.

Última actualización: 15/01/2026