# **Ejercicio 1 (Portafolio): Identificación de Vulnerabilidades**

**Actividad: Exposición y decodificación del token JWT en OWASP Juice Shop**

**Objetivo:** Realizar un análisis de seguridad identificando y decodificando un token JWT en la aplicación OWASP Juice Shop para comprender la exposición de información sensible y sus implicaciones.

**Informe de Vulnerabilidad: Exposición de Token JWT**

**Vulnerabilidad Identificada:** Token JWT almacenado en localStorage.

**Tipo:** Exposición de credenciales / sesión.

**Riesgo Estimado:** Alto. Es susceptible a ataques de Cross-Site Scripting (XSS) que podrían llevar al robo del token de sesión.

**Evidencia:** A continuación, se presenta una captura de pantalla del localStorage del navegador, donde se observa el token JWT almacenado. Esto confirma que el token de sesión es accesible directamente desde JavaScript en el frontend del navegador.

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

**Contenido del JWT:** El token JWT copiado se decodificó utilizando la herramienta jwt.io. El contenido decodificado del DECODED PAYLOAD revela la siguiente información:

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Como se puede observar, el token contiene información sensible del usuario, incluyendo el username, email, un hash de password, y el role del usuario (admin).

**Recomendación de Mitigación:** Para mitigar esta vulnerabilidad, se recomienda:

* Utilizar **cookies httpOnly** para almacenar el token de sesión. Las cookies httpOnly no son accesibles mediante JavaScript del lado del cliente, lo que las hace inmunes al robo por XSS.
* Restringir el acceso al token desde JavaScript.
* Además, se debe evitar almacenar información sensible como el hash de la contraseña directamente en el payload del JWT. El JWT debería contener solo la información esencial para la autenticación y autorización (como el ID de usuario y el rol), sin datos que puedan ser aprovechados si el token es robado.

## **Reflexión Técnica (Opcional):**

**¿Qué implicaciones de seguridad tiene exponer un token en el frontend?**

Exponer un token JWT en el frontend (como localStorage o sessionStorage) lo hace vulnerable a ataques de Cross-Site Scripting (XSS). Si un atacante logra inyectar código malicioso (JavaScript) en la página web (por ejemplo, a través de un campo de entrada no validado), ese script podría acceder al localStorage y robar el token de sesión del usuario. Con este token, el atacante podría suplantar la identidad del usuario y realizar acciones en su nombre sin necesidad de credenciales (robo de sesión).

**¿Cómo podría un atacante explotar esta vulnerabilidad con un XSS?**

Un atacante podría explotar esta vulnerabilidad si encuentra un punto de inyección XSS en la aplicación (por ejemplo, un campo de comentarios donde se permite HTML sin sanitizar). Cuando un usuario legítimo (incluyendo administradores) visite la página con el script inyectado, el JavaScript se ejecutará, robará el token de su localStorage y lo enviará al servidor del atacante.

**¿Qué buenas prácticas se recomiendan para manejar sesiones seguras?**

Además de las cookies httpOnly, otras buenas prácticas incluyen:

* + **Usar JWTs de corta duración:** Limitar el tiempo de vida de los tokens para reducir la ventana de oportunidad en caso de robo.
  + **Implementar tokens de refresco:** Utilizar tokens de corta duración para las solicitudes y un token de refresco de mayor duración (almacenado de forma más segura, quizás en una cookie httpOnly o en una base de datos) para obtener nuevos tokens de acceso.
  + **Invalidación de sesiones:** Permitir a los usuarios y administradores revocar sesiones activas.
  + **Uso de HTTPS:** Asegurar que toda la comunicación se realice a través de HTTPS para prevenir la interceptación del token en tránsito.
  + **No almacenar datos sensibles en el payload del JWT:** Como se mencionó, solo lo estrictamente necesario para la autorización.