**Autenticación y Autorización con JWT**

**y Roles en una Aplicación Web**

|  |  |
| --- | --- |
| Título del informe:  Autenticación y Autorización con JWT y Roles en una Aplicación Web | |
| Portafolio:  Lección 3: Autenticación y Autorización con JWT y Roles | Entorno:  Entorno desarrollado para la prueba. |
| Clasificación del documento:  Entorno vulnerable para pruebas – publica | Modulo:  Principios de seguridad en aplicaciones web |
| Autor(es): Sebastián Hernández Téllez | Fecha de elaboración: 23 de agosto de 2025 |

**1. Resumen Ejecutivo**

El laboratorio tuvo como objetivo implementar un sistema de autenticación y autorización basado en JWT para una aplicación de gestión de proyectos. Se configuró el registro de usuarios con contraseñas protegidas mediante bcrypt y un inicio de sesión que genera tokens firmados con rol y expiración definida. Se establecieron tres perfiles: Administrador, con acceso completo; Editor, con permisos de creación y edición; y Usuario común, limitado a la visualización de recursos asignados. Se desarrollaron middlewares de autenticación y control de roles para proteger rutas críticas y se realizaron pruebas que confirmaron la restricción de acceso adecuada. El ejercicio demostró la importancia de aplicar el principio de mínimo privilegio, reforzar la protección de tokens y adoptar buenas prácticas de ciberseguridad desde el desarrollo.

***Escenario***

Se desarrolló una aplicación web de gestión de proyectos con tres perfiles de usuario:

* **Administrador**: CRUD completo de proyectos y tareas.
* **Editor**: Crear y editar proyectos/tareas, pero no eliminarlas.
* **Usuario Común**: Solo puede ver los proyectos y tareas que le fueron asignados.

El objetivo fue implementar un sistema de autenticación y autorización seguro basado en **JSON Web Tokens (JWT)**, con control de acceso diferenciado según el rol.

**2. Metodología**

La práctica se estructuró en los siguientes pasos:

1. **Registro seguro de usuarios** → almacenamiento de contraseñas con **bcrypt**.
2. **Inicio de sesión** → validación de credenciales y emisión de un JWT firmado con secreto seguro y con tiempo de expiración.
3. **Protección de rutas con JWT** → middleware para validar tokens en cada solicitud protegida.
4. **Control de acceso por roles** → validación de permisos antes de permitir operaciones sensibles.
5. **Pruebas de seguridad** → se verificó que cada rol accediera solo a las funcionalidades permitidas.
6. **Cierre de sesión** → invalidación del JWT en cliente (eliminación de token de localStorage/sessionStorage).

**3. Implementación Técnica**

***Registro y autenticación***

* Rutas:
  + POST /register → crea usuario con contraseña hasheada.
  + POST /login → entrega JWT con rol embebido ({ sub: userId, role: "admin" }).
* Librerías: **bcrypt** para hashing y **jsonwebtoken** para emisión/verificación de tokens.

***Middleware de autenticación***

A screen shot of a computer screen

AI-generated content may be incorrect.

***Control de roles***

A computer screen with text on it

AI-generated content may be incorrect.

***Restricción de funcionalidades***

* *Administrador* → acceso completo a /projects y /tasks.
* *Editor* → acceso a creación y edición, no a eliminación.
* *Usuario común* → acceso en modo lectura únicamente a recursos asignados.

***4. Pruebas de Seguridad***

| **Rol** | **Acción** | **Resultado esperado** | **Estado** |
| --- | --- | --- | --- |
| Admin | Eliminar tarea | Permitido | ✅ |
| Editor | Eliminar tarea | Denegado | ✅ |
| Usuario común | Ver proyectos propios | Permitido | ✅ |
| Usuario común | Ver proyectos ajenos | Denegado | ✅ |
| Token inválido | Acceso a cualquier ruta | Rechazado (401) | ✅ |

**5. Riesgos detectados y mitigación**

* **Riesgo:** Uso indebido de tokens robados.
  + **Mitigación:** JWT con expiración corta + refresh tokens seguros.
* **Riesgo:** Acceso indebido por manipulación en cliente.
  + **Mitigación:** Verificación de rol en servidor, no solo en frontend.
* **Riesgo:** Contraseñas débiles.
  + **Mitigación:** Política robusta de contraseñas (mínimo 12 caracteres, complejidad).

**6. Recomendaciones de Ciberseguridad**

1. Implementar **MFA (autenticación multifactor)** para roles críticos.
2. Usar **refresh tokens** almacenados de forma segura para prolongar sesiones sin re-logueo.
3. Configurar cabeceras de seguridad HTTP (**CSP, HSTS, X-Frame-Options**).
4. Monitoreo de sesiones y alertas de actividad sospechosa.
5. Validar input en todas las capas para prevenir inyecciones.

**7. Reflexión**

La práctica demostró cómo los **JWT** permiten un control de acceso flexible en aplicaciones distribuidas, y cómo los **roles** estructuran la seguridad. Se reforzó el principio de **mínimo privilegio**, asegurando que cada usuario solo acceda a lo que necesita. Además, se puso en evidencia la importancia de proteger los tokens en cliente y de validar roles en el backend.

**8. Conclusión**

La implementación de **JWT con control de roles** permitió gestionar de manera segura la autenticación y autorización en la aplicación de proyectos. Las pruebas confirmaron que Administradores, Editores y Usuarios Comunes solo acceden a las funcionalidades correspondientes. El uso de hashing en contraseñas y tokens firmados asegura un nivel de protección adecuado. Este ejercicio refuerza la necesidad de integrar **buenas prácticas de seguridad (OWASP Top 10)** y controles de acceso robustos desde la fase de desarrollo.

**9. Anexos**

**Fragmentos de código fuente implementado como referencia.**

A screen shot of a computer program

AI-generated content may be incorrect.***Modelo de Usuario (hash seguro con bcrypt + rol)***

***Seguridad base de la app (Helmet, Rate limiting, Cookies)  
A computer screen with text and images

AI-generated content may be incorrect.***

***A computer screen shot of code

AI-generated content may be incorrect.Middleware de Autenticación (JWT) y Autorización (roles)***

***Registro y Login (bcrypt + emisión de JWT con rol y expiración)  
A screen shot of a computer program

AI-generated content may be incorrect.***

***A computer screen with text and images

AI-generated content may be incorrect.***

***Rutas protegidas por rol (Administrador / Editor / Usuario)***

***A screen shot of a computer program

AI-generated content may be incorrect.***

***Montaje del servidor (conexión DB + rutas)***

***A computer screen with colorful text

AI-generated content may be incorrect.***

***Variables de entorno (.env)***

***A screenshot of a computer

AI-generated content may be incorrect.***

***Pruebas rápidas (curl / Postman)***

***A computer screen with text

AI-generated content may be incorrect.***