Blandskron

Informe de Auditoría de Seguridad – `user-service`

1. Hallazgos Críticos (Alta Prioridad)

Gestión insegura de contraseñas

- Las contraseñas son almacenadas y modificadas **en texto plano**, sin ningún tipo de cifrado o hashing.

Riesgo:

- Compromiso total de los datos si hay una filtración de la base de datos.

Mejora recomendada (OWASP A2 - Broken Authentication):

- Implementar `bcrypt` u otra función de hashing para almacenar las contraseñas.

```
const bcrypt = require('bcrypt');

const hashedPassword = await bcrypt.hash(req.body.password, 10);
```

API pública sin autenticación ni autorización

- Todos los endpoints (`GET`, `POST`, `PUT`, `DELETE /users`) están completamente expuestos al público sin tokens ni verificación de identidad.

Riesgo:

- Cualquier atacante puede leer, crear, modificar o eliminar usuarios.

Mejora recomendada (OWASP A5 - Broken Access Control / ISO A.9.1):

- Añadir autenticación con JWT.
- Restringir el acceso a roles (`ADMIN`, `USER`, etc.).

Validaciones de entrada inexistentes

- No se validan los campos de entrada en ninguna ruta.
- No hay verificación de tipos, longitud ni formato de `email`, `password`, `role`.

Riesgo:

- Posibilidad de inyecciones de comandos, inserciones maliciosas, errores inesperados.

Mejora recomendada (OWASP A1 - Injection / A4 - Insecure Design):

- Implementar middleware de validación como `Joi`, `express-validator`, o `Zod`.

2. Hallazgos Moderados

Exposición de Mongo URI con credenciales

- Se incluye `mongodb://admin:admin123@localhost:27017` directamente en el código (`db.js`).

Mejora recomendada (ISO 27001 A.9.2, A.12.3):

- Usar `.env` con `dotenv` y evitar hardcoding de credenciales.

CORS completamente abierto

- Se permite `origin: '*'`, lo cual habilita peticiones desde cualquier origen, incluyendo scripts maliciosos.

Mejora recomendada (OWASP A6 - Security Misconfiguration):

- Restringir orígenes confiables:

```
app.use(cors({
  origin: ['https://frontend.miapp.com']
}));
```

Swagger expone todos los endpoints sin seguridad

- No se declara ningún esquema de seguridad en `swagger.json`.
- Las rutas CRUD están etiquetadas como “sin autenticación”.

Mejora recomendada (OWASP API9 - Improper Assets Management):

- Agregar autenticación en Swagger con `securitySchemes` y requerir `Bearer Token` en todos los endpoints.

No hay protección contra ataques automatizados

- No se usa ningún mecanismo de **rate-limiting** ni captchas.

Mejora recomendada (OWASP A7 - Identification and Authentication Failures):

- Añadir `express-rate-limit` o `slow-down`.

Falta cifrado en tránsito (solo HTTP)

- Todos los ejemplos de Postman (`user_service.json`) indican uso de `http://localhost:4000`.

Mejora recomendada (OWASP A6 - Sensitive Data Exposure):

- Habilitar HTTPS y exigirlo en ambientes productivos.

3. Hallazgos Menores

Falta de logging de eventos críticos

- No hay registros de cambios en usuarios, errores graves, intentos de acceso indebido.

Mejora recomendada (ISO 27001 A.12.4):

- Añadir `morgan`, `winston`, o sistema SIEM externo.

Conformidad con Estándares

Estándar	Cumplimiento Actual	Recomendación
OWASP Top 10	❌ Críticamente débil	Corregir A1, A2, A5, A6, A7
ISO/IEC 27001:2013	❌ Bajo cumplimiento	Aplicar controles A.9 (acceso), A.12, A.18
GDPR (si aplica)	❌ Riesgo extremo	No hay protección de datos sensibles

Acciones Correctivas Recomendadas

Prioridad	Acción
🔴 Alta	Implementar hashing de contraseñas (`bcrypt`)
🔴 Alta	Añadir autenticación y autorización por JWT
🟠 Media	Validar entrada en todos los endpoints

Blandskron

🟠 Media	Mover credenciales a `.env`
🟡 Baja	Restringir orígenes CORS
🟡 Baja	Agregar documentación segura en Swagger