Proyecto 3 - Administración de Redes y Sistemas Computacionales
Hardening y Seguridad de la Arquitectura Basada en Contenedores Docker

Profesor: **Ricardo Pérez**
riperez@utalca.cl

Semestre 2025-II

Descripción General

Este proyecto es la **continuación directa del Proyecto 2** y tiene como objetivo que implementen una capa completa de **seguridad y hardening** sobre la arquitectura de microservicios ya desarrollada.
El proyecto se enfoca en cumplir con estándares de seguridad empresarial, aplicar mejores prácticas de hardening, implementar HTTPS, proteger datos sensibles y preparar el sistema para resistir amenazas comunes.
Fecha de presentación: Segunda semana de Diciembre con los mismos equipos del Proyecto 2.

⚠ IMPORTANTE

Prerequisito obligatorio: Este proyecto requiere haber completado el Proyecto 2. Los equipos trabajarán sobre la misma arquitectura ya implementada, agregando la capa de seguridad.
La licitación y arquitectura base permanecen iguales. El foco ahora es **asegurar** lo que ya construyeron.

1. Objetivos del Proyecto

- Implementar HTTPS/TLS en todos los servicios expuestos
- Aplicar hardening a contenedores Docker
- Asegurar bases de datos mediante hardening

- Implementar gestión segura de secretos
- Realizar análisis básico de vulnerabilidades
- Implementar logging y auditoría de seguridad
- Aplicar principio de mínimo privilegio
- Documentar políticas de seguridad

2. Requisitos de Seguridad Obligatorios

2.1. HTTPS/TLS - Comunicación Segura

REQUISITO OBLIGATORIO

Todos los servicios web expuestos DEBEN usar HTTPS.
Requisitos mínimos:

- Frontend accesible únicamente por HTTPS
- API Gateway con certificado TLS configurado
- Redirección automática HTTP → HTTPS
- Headers de seguridad HTTP configurados (HSTS, X-Frame-Options, CSP, etc.)

Opciones de implementación (solo una opción):

- Certificados autofirmados con OpenSSL (válido para este proyecto)
- Traefik con gestión automática de certificados (recomendado)
- Let's Encrypt con Certbot (si tienen dominio público)

Entregables:

- Certificados TLS configurados y funcionando
- Configuración de servidor web con HTTPS
- Script de generación de certificados en `scripts/security/`
- Captura de pantalla del navegador mostrando HTTPS activo
- Documentación del proceso en README

2.2. Hardening de Contenedores Docker

REQUISITO OBLIGATORIO

Todos los contenedores deben cumplir con prácticas de hardening.
Requisitos obligatorios en Dockerfiles:

1. **Usuario no privilegiado**: Ejecutar contenedores con usuario diferente a root (excepto casos justificados técnicamente)
2. **Imágenes base con versión específica**: No usar tags `:latest`, especificar versión exacta
3. **Multi-stage builds**: Separar etapas de construcción y ejecución cuando sea posible
4. **Escaneo de vulnerabilidades**: Todas las imágenes deben ser escaneadas

Requisitos en Docker Compose:

- Configurar `security_opt` con `no-new-privileges`
- Eliminar capabilities innecesarias con `cap_drop`
- Configurar límites de recursos (memoria, CPU)
- Políticas de restart apropiadas
- Configuración de logging con rotación

Entregables:

- Todos los Dockerfiles actualizados con medidas de hardening
- Reporte de escaneo de vulnerabilidades (Trivy, Docker Scout, u otra herramienta)
- Docker Compose con configuraciones de seguridad
- Documento `docs/hardening-contenedores.md` explicando medidas aplicadas

2.3. Hardening de Bases de Datos

REQUISITO OBLIGATORIO

Las bases de datos deben estar aseguradas según mejores prácticas. Requisitos obligatorios (aplicables a cualquier motor de BD):

1. **Autenticación obligatoria**
   - Deshabilitar acceso sin contraseña
   - Usar contraseñas fuertes (mínimo 12 caracteres)
   - Implementar el método de autenticación más seguro disponible
2. **Usuario de aplicación con privilegios mínimos**
   - NO usar usuarios administrativos (root, postgres, admin) en la aplicación
   - Crear usuario específico para la aplicación
   - Otorgar solo permisos necesarios (SELECT, INSERT, UPDATE, DELETE)
   - NO otorgar permisos administrativos (DROP, CREATE USER, GRANT)
3. **Gestión segura de credenciales**
   - Usar Docker Secrets o variables de entorno seguras
   - NO hardcodear credenciales en archivos de configuración
4. **Aislamiento de red**
   - Base de datos en red interna Docker
   - NO exponer puertos de BD directamente al host
   - Solo servicios que necesiten acceso en la misma red
5. **Logging y auditoría**
   - Habilitar logs de conexiones y desconexiones
   - Registrar intentos fallidos de autenticación
6. **Configuración de red segura**
   - Limitar hosts que pueden conectarse
   - Usar subredes específicas
   - Rechazar conexiones no autorizadas

Entregables:

- Archivos de configuración hardeneados en `infrastructure/database/`
- Scripts de inicialización con usuarios y permisos
- Documento `docs/hardening-database.md` explicando:
  - Medidas de seguridad aplicadas
  - Gestión de usuarios y permisos
  - Configuración de logging
  - Pruebas de funcionamiento
- Evidencia de que la BD no es accesible directamente desde el host

2.4. Gestión de Secretos

REQUISITO OBLIGATORIO

NO debe haber credenciales en texto plano en el código o repositorio.
Implementar una de estas opciones:

- Docker Secrets (recomendado para este proyecto)
- Variables de entorno con archivo .env (mínimo aceptable)

Requisitos:

- Archivo `.env` debe estar en `.gitignore`
- Proporcionar `.env.example` como plantilla
- Implementar rotación de secretos
- Documentar procedimiento de gestión

Entregables:

- Sistema de gestión de secretos funcionando
- `.env.example` con plantilla (SÍ en git)
- Script de rotación en `scripts/security/rotate-secrets.sh`
- Verificación de que NO hay secretos reales en el repositorio
- Documentación de uso en README

2.5. Web Application Firewall (WAF) / Rate Limiting

REQUISITO OBLIGATORIO

Implementar protección básica contra ataques web comunes.
Opciones de implementación:

- Traefik con middlewares de rate limiting (más simple)
- Nginx con módulos de rate limiting
- ModSecurity + OWASP Core Rule Set (avanzado)

Configuraciones mínimas requeridas:

- Rate limiting (límite de peticiones por IP)
- Bloqueo de user-agents maliciosos conocidos
- Headers de seguridad HTTP
- Logs de peticiones bloqueadas

Entregables:

- WAF o rate limiting configurado y funcionando
- Configuración documentada
- Logs mostrando bloqueos efectivos
- Capturas de pantalla de pruebas de bloqueo

2.6. Análisis de Vulnerabilidades

REQUISITO OBLIGATORIO

Escanear y documentar vulnerabilidades del sistema.
Análisis obligatorios:

1. **Escaneo de imágenes Docker**
   - Usar Trivy, Docker Scout, o Anchore
   - Escanear TODAS las imágenes del proyecto
   - Reportar vulnerabilidades HIGH y CRITICAL
2. **Escaneo de dependencias**
   - npm audit (Node.js)
   - pip safety (Python)
   - Herramientas según el stack utilizado
3. **Escaneo de puertos**
   - Usar Nmap para verificar servicios expuestos
   - Verificar que solo puertos necesarios están abiertos
4. **Análisis web básico**
   - Nikto o herramienta similar
   - Verificar headers de seguridad

Entregables:

- Carpeta `docs/reportes/` con todos los reportes de escaneo
- Documento `vulnerabilidades-remediadas.md` explicando:
  - Vulnerabilidades encontradas
  - Cómo se corrigieron
  - Vulnerabilidades aceptadas con justificación
- Script automatizado en `scripts/security/scan-vulnerabilities.sh`

2.7. Logging y Auditoría de Seguridad

REQUISITO OBLIGATORIO

Sistema de logs centralizado con enfoque en seguridad.
Opciones de implementación:

- Configuración básica con Docker logs y rotación
- Loki + Grafana (recomendado)
- ELK Stack (avanzado)

Eventos de seguridad obligatorios a registrar:

- Intentos de autenticación (exitosos y fallidos)
- Accesos denegados (401, 403)
- Errores de autorización
- Acceso a endpoints sensibles
- Cambios en configuración
- Conexiones a base de datos

Entregables:

- Sistema de logging configurado
- Configuración de logs en docker-compose.yml
- Script para consultar logs de seguridad
- Dashboard o capturas mostrando logs funcionando
- Documentación de eventos registrados

2.8. Políticas y Documentación de Seguridad

REQUISITO OBLIGATORIO

Documentar políticas y procedimientos de seguridad.
Documentos obligatorios en carpeta `docs/`:

1. **Política de Seguridad** (`politica-seguridad.md`)
   - Objetivos de seguridad
   - Responsabilidades del equipo
   - Política de contraseñas
   - Política de actualizaciones
   - Gestión de accesos
   - Retención de logs
2. **Matriz de Riesgos** (`matriz-riesgos.md`)
   - Identificar 8-10 riesgos principales
   - Clasificar por probabilidad e impacto
   - Definir mitigaciones para cada riesgo
3. **Checklist OWASP Top 10** (`owasp-top10.md`)
   - Para cada uno de los 10 riesgos de OWASP 2021
   - Indicar si aplica al sistema
   - Explicar cómo se está mitigando
4. **Plan de Respuesta a Incidentes** (`plan-incidentes.md`)
   - Procedimiento de detección
   - Contención inmediata
   - Análisis y erradicación
   - Recuperación del sistema
   - Lecciones aprendidas
5. **Cumplimiento Normativo** (`cumplimiento-normativo.md`)
   - Ley N° 19.628 - Protección de Datos
   - Ley N° 21.459 - Ciberseguridad
   - Normas de Gobierno Digital

Entregables:

- Los 5 documentos mencionados
- Cada documento con mínimo 1-2 páginas
- Formato Markdown
- Referencias a configuraciones reales del proyecto

3. Evaluación

3.1. Criterios de Evaluación (100 %)

Aspecto	Puntos
1. HTTPS/TLS	**15 %**
Certificados funcionando	5 %
Headers de seguridad	4 %
Redirección HTTP → HTTPS	3 %
Documentación	3 %
2. Hardening de Contenedores	**20 %**
Usuarios no privilegiados	6 %
Configuraciones de seguridad	5 %
Escaneo de vulnerabilidades	5 %
Multi-stage builds	4 %
3. Hardening de Bases de Datos	**20 %**
Autenticación y privilegios mínimos	8 %
Aislamiento de red	5 %
Logging y auditoría	4 %
Documentación	3 %
4. Gestión de Secretos	**15 %**
Sistema implementado	7 %
Sin credenciales en Git	5 %
Script de rotación	3 %
5. WAF y Seguridad de Red	**10 %**
WAF/Rate limiting funcionando	5 %
Segmentación de redes	3 %
Pruebas de bloqueo	2 %
6. Análisis de Vulnerabilidades	**10 %**
Reportes completos	5 %
Documento de remediación	3 %
Script automatizado	2 %
7. Logging y Auditoría	**5 %**
Sistema funcionando	3 %
Eventos registrados	2 %
8. Documentación y Políticas	**5 %**
Documentos obligatorios	3 %
README actualizado	2 %
TOTAL	**100 %**

3.2. Defensa del Proyecto

Duración: 15 minutos
Estructura obligatoria:

1. **Introducción (1 min)**: Resumen de medidas implementadas
2. **Demostración HTTPS (3 min)**: Mostrar sitio funcionando con HTTPS, certificado válido, headers de seguridad

3. **Hardening Contenedores (3 min)**: Demostrar usuarios no-root, mostrar escaneo de vulnerabilidades, explicar configuraciones
4. **Hardening BD (2 min)**: Demostrar que BD no es accesible directamente, mostrar usuarios con privilegios mínimos
5. **Prueba de Seguridad (3 min)**: Demostrar rate limiting, WAF bloqueando ataques, verificar secretos no en Git
6. **Documentación (2 min)**: Mostrar estructura docs/, reportes de vulnerabilidades
7. **Preguntas (1 min)**

4. Fechas Importantes

Hito	**Fecha**
Publicación del proyecto	Lunes 18 de Noviembre
Checkpoint opcional	Lunes 25 de Noviembre
Entrega y defensa	**Segunda semana de Diciembre**

5. Recursos Recomendados

5.1. Documentación

- Docker Security: https://docs.docker.com/engine/security/
- CIS Benchmarks: https://www.cisecurity.org/benchmark/docker
- OWASP Top 10: https://owasp.org/www-project-top-ten/
- OWASP Cheat Sheets: https://cheatsheetseries.owasp.org/

5.2. Herramientas

- Trivy: https://github.com/aquasecurity/trivy
- Docker Scout: https://docs.docker.com/scout/
- HashiCorp Vault: https://www.vaultproject.io/
- Traefik: https://doc.traefik.io/traefik/

5.3. Normativas Chilenas

- Ley N° 19.628 - Protección de Datos Personales
- Ley N° 21.459 - Marco sobre Ciberseguridad
- Gobierno Digital: https://digital.gob.cl

6. Preguntas Frecuentes

P: ¿Empezamos de cero?
R: No. Trabajar sobre el Proyecto 2, agregando la capa de seguridad.
P: ¿Certificados autofirmados son válidos?
R: Sí, perfectamente válidos para este proyecto.
P: ¿Qué hacer con vulnerabilidades que no podemos corregir?
R: Documentarlas en el reporte de remediación con justificación.

P: ¿Es obligatorio el WAF completo?
R: Rate limiting básico bien implementado es suficiente.
P: ¿Cuánto tiempo debería tomar?
R: Aproximadamente 3 semanas, 6-8 horas/semana por persona.

7. Recomendaciones Finales

1. Empezar con HTTPS - es la base fundamental
2. Trabajar un requisito a la vez - completar, verificar, documentar
3. Leer reportes de escaneo y entender las vulnerabilidades
4. Probar intentando atacar el propio sistema
5. Documentar mientras trabajan, no al final
6. Usar .gitignore correctamente desde el inicio
7. Crear scripts de verificación automatizados
8. Consultar referencias oficiales (OWASP, CIS)
9. Dividir trabajo pero revisar todo el equipo
10. Practicar la presentación antes del día de defensa