

MANUAL 5: GUÍA DE MONITOREO Y OPERACIONES - SISTEMA CONA

INFORMACIÓN DEL DOCUMENTO

Fecha de Creación: 21 de Julio de 2025
Proyecto: Sistema CONA (Gestión CONAVEG)
Audiencia: DevOps, SRE, Administradores de Sistema
Nivel: Avanzado
Tiempo Estimado: 4-6 horas (configuración y estudio)
Última Actualización: 21 de Julio de 2025

OBJETIVOS DE APRENDIZAJE

Al finalizar este manual, serás capaz de:

- ☒ Configurar un sistema de monitoreo completo para la aplicación CONA.
 - ☒ Crear dashboards y alertas para métricas clave de performance y seguridad.
 - ☒ Realizar procedimientos de mantenimiento preventivo y correctivo.
 - ☒ Gestionar tareas programadas y procesos de limpieza automática.
 - ☒ Diagnosticar y resolver problemas comunes utilizando la guía de troubleshooting.
 - ☒ Entender las estrategias de escalado y optimización de performance.
 - ☒ Aplicar procedimientos de respuesta a incidentes.
-

REQUISITOS PREVIOS

Conocimientos Necesarios:

- Experiencia con herramientas de monitoreo (Prometheus, Grafana).
- Conocimientos de administración de sistemas y redes.
- Familiaridad con la línea de comandos y scripting.
- Comprensión de métricas de aplicaciones Java (JVM, etc.).

Herramientas Recomendadas:

- Prometheus: Para la recolección y almacenamiento de métricas.
- Grafana: Para la visualización de métricas y creación de dashboards.
- Alertmanager: Para la gestión de alertas.
- Cliente de base de datos y acceso a los logs del sistema.



CONFIGURACIÓN DE MONITOREO Y MÉTRICAS

El Sistema CONA expone métricas en un formato compatible con Prometheus a través de Spring Boot Actuator.

Habilitar Endpoints de Métricas:

Asegúrate de que la siguiente configuración esté en `application.properties`:

```
# Exponer endpoints de health, metrics y prometheus
management.endpoints.web.exposure.include=health,metrics,prometheus
management.endpoint.metrics.enabled=true
management.metrics.export.prometheus.enabled=true
```

Una vez habilitado, las métricas estarán disponibles en `http://<host>:<port>/conaveg/actuator/prometheus`.

Métricas Clave a Monitorear:

La documentación completa de métricas se encuentra en `docs/testing/Monitoring_Manual.md`. A continuación, un resumen:

Métricas de Aplicación (JVM):

- `jvm_memory_used_bytes`: Uso de memoria del heap.
- `system_cpu_usage`: Uso de CPU por el proceso de la aplicación.
- `jvm_threads_live_threads`: Número de hilos activos.

Métricas de Performance:

- `http_server_requests_seconds`: Latencia de las solicitudes HTTP.
- `hikaricp_connections_active`: Conexiones activas en el pool de la base de datos.

Métricas de Seguridad y Negocio:

- `security_audit_events_total`: Contador de eventos de auditoría (logins, fallos, etc.).
- `rate_limit_violations_total`: Solicitudes bloqueadas por el rate limiting.
- `password_reset_requests_rate`: Tasa de solicitudes de recuperación de contraseña.



DASHBOARDS Y ALERTAS RECOMENDADAS

Se recomienda crear al menos dos dashboards principales en Grafana.

Dashboard 1: Resumen de Salud General (Overview)

- Paneles:
 - o Estado de la Aplicación: Indicador UP/DOWN (basado en `up` de Prometheus).

- Tasa de Errores HTTP (5xx): Gráfico de líneas.
- Latencia p95: Gráfico de líneas de la latencia del 95% de las solicitudes.
- Uso de CPU y Memoria: Gráficos de medidor (gauge).
- Conexiones de BD Activas: Gráfico de líneas.

Dashboard 2: Seguridad y Autenticación

- Paneles:
 - Tasa de Logins Exitosos vs. Fallidos: Gráfico de barras apiladas.
 - Eventos de Auditoría por Tipo: Gráfico de tarta (pie chart).
 - Violaciones de Rate Limiting (por hora): Gráfico de barras.
 - IPs con más Intentos Fallidos: Tabla con las IPs más activas.

Configuración de Alertas (Alertmanager):

Alertas Críticas (HIGH):

- **ApplicationDown**: Si el endpoint de health está caído por más de 1 minuto.
- **HighErrorRate**: Si la tasa de errores 5xx supera el 5% durante 5 minutos.
- **DatabaseDown**: Si no se puede conectar a la base de datos.
- **SecurityBreachSuspected**: Si hay un pico masivo de logins fallidos desde múltiples IPs.

Alertas de Advertencia (MEDIUM):

- **HighLatency**: Si la latencia p95 supera 1 segundo durante 10 minutos.
- **HighMemoryUsage**: Si el uso de memoria del heap supera el 85%.
- **RateLimitingActive**: Si se bloquean más de 50 solicitudes por hora.



PROCEDIMIENTOS DE MANTENIMIENTO

El mantenimiento regular es clave para la estabilidad del sistema.

Tareas de Mantenimiento Diario:

- Verificar Backups: Asegurarse de que el backup diario de la base de datos se completó con éxito.
- Revisar Logs Críticos: Revisar los logs de seguridad y de aplicación en busca de errores **ERROR** o **WARN** no esperados.

```
grep "ERROR" logs/spring.log
grep "HIGH" logs/security.log
```

Tareas de Mantenimiento Semanal:

- Revisar el Crecimiento de la Base de Datos: Monitorear el tamaño de las tablas, especialmente `security_audit_logs`.
- Actualizar el Sistema Operativo: Aplicar parches de seguridad al SO.

```
sudo apt update && sudo apt upgrade -y
```

- Revisar el Rendimiento: Analizar los dashboards de Grafana en busca de tendencias de degradación del rendimiento.

Tareas de Mantenimiento Mensual:

- Rotación de Claves: Considerar la rotación de `app.jwt.secret` y otras claves sensibles.
- Archivado de Logs: Mover logs antiguos a un almacenamiento secundario.
- Revisión de Índices de BD: Analizar y optimizar los índices de la base de datos si es necesario.



TAREAS PROGRAMADAS Y LIMPIEZA AUTOMÁTICA

El sistema incluye tareas programadas para su auto-mantenimiento.

Tareas Configuradas:

- `cleanupExpiredTokens`: Se ejecuta diariamente a las 2 AM. Elimina los tokens de recuperación de contraseña que han expirado.
- `cleanupOldSecurityLogs`: Se ejecuta semanalmente. Archiva o elimina logs de auditoría con más de 90 días de antigüedad.
- `cleanupOldRateLimitAttempts`: Se ejecuta diariamente. Limpia los registros de intentos de login fallidos.

Verificación de Tareas Programadas:

Los logs de las tareas programadas se pueden encontrar en `logs/spring.log`.

```
grep "ScheduledTasks" logs/spring.log | tail -n 50
```

Problema Común: Tareas no se ejecutan:

- Causa: La anotación `@EnableScheduling` puede estar ausente en la clase principal de la aplicación.
- Solución: Asegurarse de que `ConApplication.java` tenga `@EnableScheduling`.



TROUBLESHOOTING DE PROBLEMAS COMUNES

Una guía de troubleshooting completa y detallada se encuentra en [docs/testing/Troubleshooting_Guide.md](#). A continuación, se presentan los problemas más comunes que un operador puede enfrentar.

Problema: La aplicación no inicia.

- Diagnóstico:
 1. Revisar `logs/spring.log` para ver el error exacto (ej. `Port 8080 was already in use, Failed to connect to database`).
 2. Verificar que la base de datos esté en línea.
 3. Asegurarse de que todas las variables de entorno necesarias estén configuradas.
- Solución: Resolver el problema específico indicado en el log.

Problema: Usuarios reportan lentitud (alta latencia).

- Diagnóstico:
 4. Revisar el dashboard de Grafana para identificar si la latencia es general o en endpoints específicos.
 5. Verificar el uso de CPU y memoria. Si es alto, puede ser la causa.
 6. Consultar `SHOW PROCESSLIST;` en la base de datos para ver si hay queries lentas.
- Solución: Puede requerir optimización de queries, escalado de recursos o reinicio de la aplicación.

Problema: Usuarios legítimos son bloqueados por el Rate Limiting.

- Diagnóstico:
 7. Consultar los logs de seguridad para identificar la IP o el email bloqueado.
 8. Verificar si los umbrales de `app.security.rate-limit.*` son demasiado bajos para el tráfico normal.
- Solución: Ajustar los umbrales en `application.properties` y reiniciar. Para casos urgentes, se puede desbloquear una IP manualmente (ver Manual de Seguridad).

ESCALADO Y OPTIMIZACIÓN DE PERFORMANCE

Escalado Vertical:

- Descripción: Aumentar los recursos de la máquina actual (más CPU, más RAM).
- Cuándo usarlo: Es la forma más simple de mejorar el rendimiento si la aplicación no está optimizada para la concurrencia.
- Acción:
 - o Aumentar la memoria heap de la JVM: `java -Xmx4g -Xms1g ...`
 - o Migrar el servidor a una instancia con más capacidad.

Escalado Horizontal:

- Descripción: Añadir más instancias de la aplicación detrás de un balanceador de carga.
- Cuándo usarlo: Cuando la aplicación necesita manejar un alto volumen de solicitudes concurrentes.
- Requisitos:
 - o La aplicación debe ser *stateless*. El Sistema CONA lo es, ya que el estado (JWT) se maneja en el cliente.
 - o Se necesita un balanceador de carga (ej. Nginx, HAProxy) para distribuir el tráfico.

Optimización de Performance:

- Base de Datos: Asegurarse de que todas las columnas utilizadas en cláusulas **WHERE** y **JOIN** estén indexadas.
- Caché: Implementar una capa de caché (como Redis o EhCache) para datos que no cambian frecuentemente.
- JVM Tuning: Ajustar los parámetros del Garbage Collector (GC) para optimizar la gestión de memoria.



PROCEDIMIENTOS DE RESPUESTA A INCIDENTES

En caso de un incidente de seguridad o de rendimiento, siga estos pasos.

Fase 1: Detección y Verificación (0-5 min)

1. Recepción de Alerta: Una alerta se dispara desde Alertmanager.
2. Verificación Rápida:
 - o Consultar el dashboard de Grafana para entender el impacto.
 - o Revisar los últimos logs en busca de errores.
 - o Ejecutar un health check manual: `curl http://<host>:<port>/conaveg/actuador/health`.

Fase 2: Contención y Diagnóstico (5-30 min)

3. Contener el Problema:
 - o Incidente de Seguridad: Si es un ataque, bloquear la IP atacante en el firewall.
 - o Incidente de Performance: Si es una query descontrolada, terminar el proceso en la base de datos.
4. Diagnóstico Profundo: Utilizar la `Troubleshooting_Guide.md` para encontrar la causa raíz.

Fase 3: Erradicación y Recuperación (30-60 min)

1. Aplicar la Solución:




- Desplegar un hotfix.
 - Ajustar la configuración.
 - Reiniciar el servicio.
2. Verificar la Recuperación: Monitorear las métricas para asegurarse de que vuelven a la normalidad.

Fase 4: Post-Incidente (Día Siguiente)



1. Análisis Post-Mortem: Documentar qué pasó, por qué pasó y cómo se resolvió.
2. Plan de Acción: Crear tareas para prevenir que el incidente vuelva a ocurrir.





SOPORTE Y RECURSOS ADICIONALES

Documentación Relevante:

-  [Guía de Troubleshooting](#)
-  [Métricas de Performance](#)
-  [Manual de Seguridad y Administración](#)

Canales de Soporte:

-  Email: ops-support@conaveg.com
-  Slack: #cona-operations

 Fecha de Creación: 21 de Julio de 2025
 Responsable: Equipo de Operaciones CONA
 Estado: Manual Completo y Validado
 Próxima Revisión: 21 de Agosto de 2025