

Sondas en Kubernetes: liveness, readiness y startup

Una página para revisar las sondas antes de desplegar y para diagnosticar un CrashLoopBackOff sin abrir el código primero.

1 Qué hace cada sonda cuando falla

SONDA	SI FALLA	ÚSALA PARA
<code>livenessProbe</code>	Kubelet reinicia el contenedor.	Detectar un proceso colgado que no se recupera solo.
<code>readinessProbe</code>	El pod sale de los endpoints del Service y deja de recibir tráfico. No se reinicia.	No enviar tráfico a un pod que ahora no puede atender: arrancando, saturado o sin una dependencia.
<code>startupProbe</code>	Mientras no pasa, las otras dos no se ejecutan. Si agota sus intentos, kubelet reinicia el contenedor.	Apps con arranque lento o variable.

Valores por defecto: `initialDelaySeconds: 0, periodSeconds: 10, timeoutSeconds: 1, successThreshold: 1, failureThreshold: 3`. Una liveness reinicia el contenedor tras `failureThreshold` fallos seguidos, contados desde `initialDelaySeconds` a intervalos de `periodSeconds`.

2 Antes de desplegar

- ☐ Las apps con arranque lento o variable tienen `startupProbe`, con `failureThreshold` × `periodSeconds` por encima del peor arranque medido.
- ☐ La liveness comprueba solo el propio proceso, no la base de datos ni otros servicios. Si una dependencia cae, reiniciar todos los pods no la arregla.
- ☐ La readiness refleja si el pod puede atender tráfico ahora mismo.
- ☐ El `timeoutSeconds` deja margen para picos de CPU, throttling por límites o pausas del recolector de basura.
- ☐ Los endpoints de las sondas responden rápido y sin trabajo pesado: sin consultas ni llamadas a otros servicios.
- ☐ Tras un cambio que alarga el arranque (migraciones, carga de caché, una dependencia nueva) se revisan los tiempos de las sondas.
- ☐ Los valores se ajustan por contenedor, no se copian de una plantilla genérica.

3 Si hay CrashLoopBackOff

```
# Eventos: busca "Liveness probe failed"
# y "will be restarted"
kubectl describe pod <pod>
```

```
# Solo los fallos de sondas
kubectl get events \
  --field-selector reason=Unhealthy
```

```
# Logs del arranque anterior
kubectl logs <pod> --previous
```

Cómo leerlo: si el arranque se corta sin un error de la app y los eventos hablan de la sonda, la causa es la sonda. Si los logs muestran un error de la app, la causa está en el código o en la configuración. Un contenedor reiniciado por la liveness suele aparecer en **Last State** con exit code 137 o 143.

4 Plantilla de partida

```
startupProbe:
  httpGet: { path: /healthz, port: 8080 }
  periodSeconds: 5
  failureThreshold: 30 # hasta 150 s para arrancar
livenessProbe:
  httpGet: { path: /healthz, port: 8080 }
  periodSeconds: 10
  timeoutSeconds: 2
  failureThreshold: 3
```

```
readinessProbe:
  httpGet: { path: /ready, port: 8080 }
  periodSeconds: 5
  timeoutSeconds: 2
  failureThreshold: 3
# /healthz: el proceso responde
# /ready: puede atender tráfico ahora
```

Ajusta los números a los tiempos reales de tu app. No subas el `initialDelaySeconds` de la liveness para cubrir un arranque lento: retrasa la detección de cuelgues reales durante toda la vida del pod.