harishnshetty / **Kubernetes-From-scratch-Project**

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

main ▾ Kubernetes-From-scratch-Project / **9.probes** /

Go to file t · Add file ▾ · ···

harishnshetty readme added probes 67dc0e7 · 1 minute ago

Name	Name	Last commit date
..		
Readme.MD	readme added probes	1 minute ago
liveness-command.yml	readme added probes	1 minute ago
liveness-http-error.yml	readme added probes	1 minute ago
liveness-http-nginx.yml	readme added probes	1 minute ago
liveness-tcp.yml	readme added probes	1 minute ago
readness-http.yml	readme added probes	1 minute ago

Readme.MD

🚀 Kubernetes Health Checks (Liveness & Readiness)

> Production-grade examples of **Liveness** and **Readiness** probes using **exec**, **HTTP**, and **TCP** methods with real YAML manifests.

📑 Table of Contents

🧠 What are Kubernetes Probes?

Kubernetes probes allow the **kubelet** to check container health.

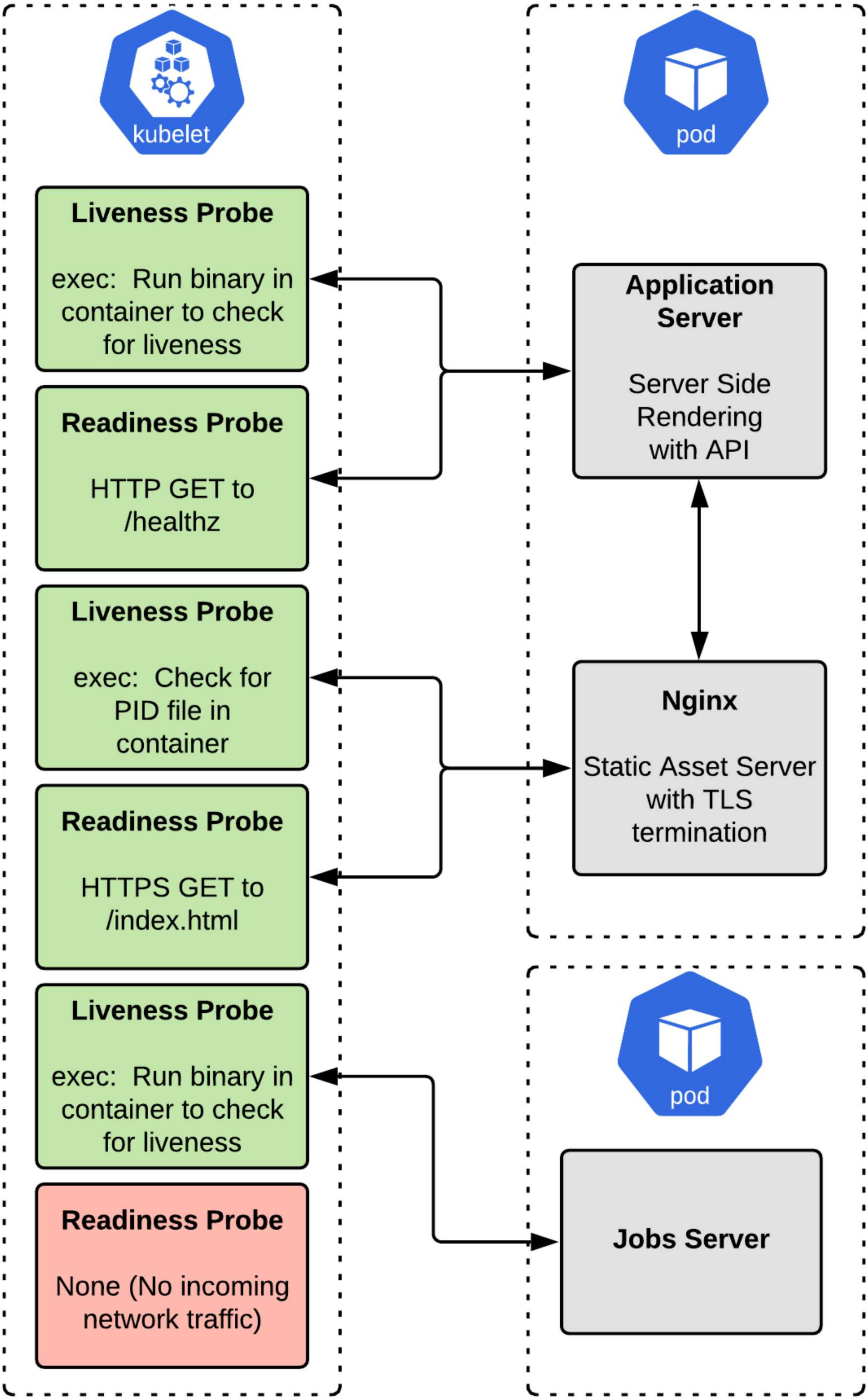
Probe Type	Purpose
Liveness	Is the container alive? If not → restart
Readiness	Is the container ready to serve traffic?
Startup	Is the app finished starting?

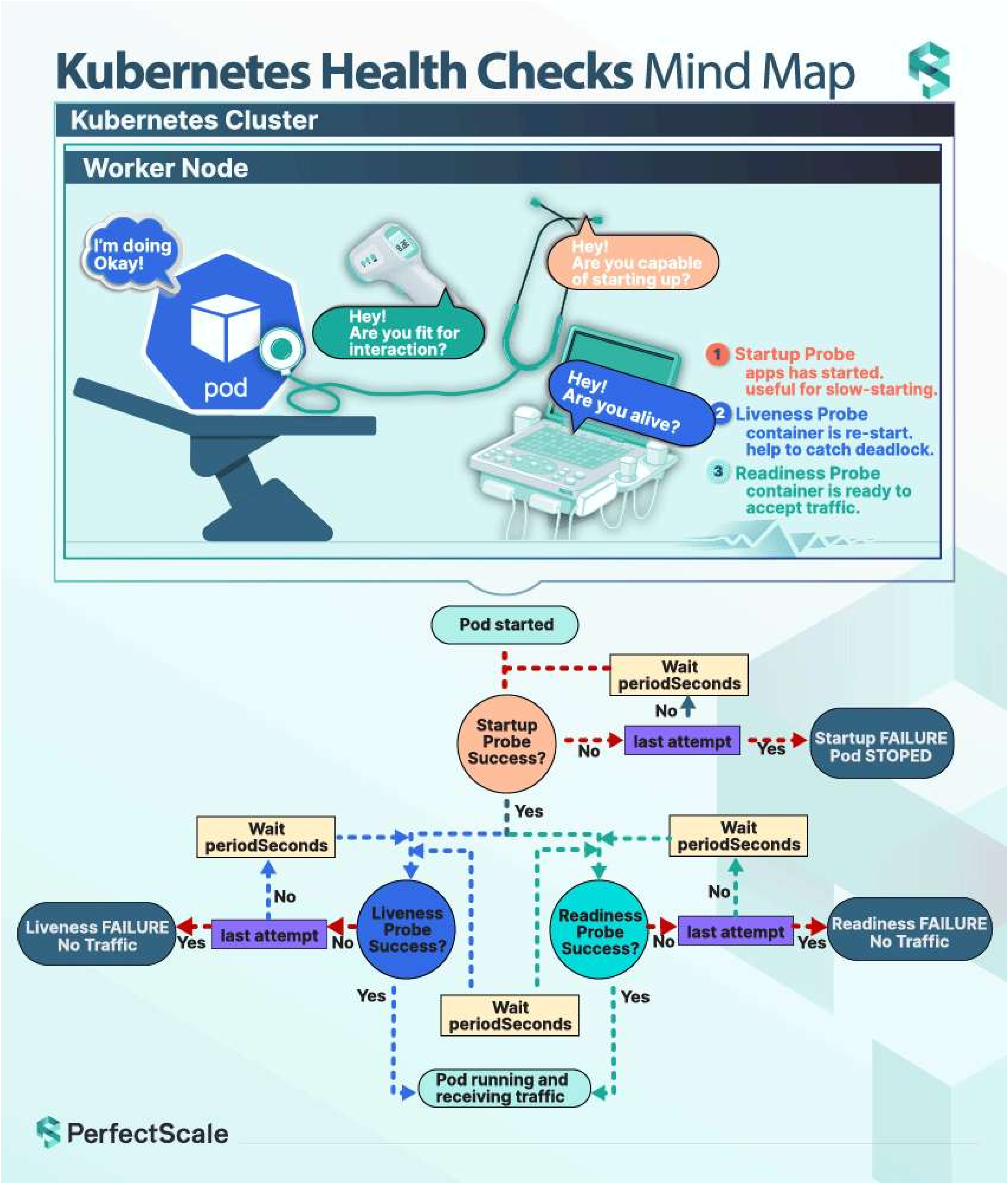
🔍 Liveness vs Readiness (Real Meaning)

```
Liveness  → restart container
Readiness → remove pod from Service endpoints
```

Scenario	Liveness	Readiness
App crashed	❌ restart	❌ no traffic
DB down	✅ running	❌ no traffic
Slow startup	❌ restart	❌ no traffic

🏗 Architecture Diagram – Probe Flow


kubelet
pod
Liveness Probe
exec: Run binary in container to check for liveness
Readiness Probe
HTTP GET to /healthz
Application Server
Server Side Rendering with API
Liveness Probe
exec: Check for PID file in container
Readiness Probe
HTTPS GET to /index.html
Nginx
Static Asset Server with TLS termination
pod
Liveness Probe
exec: Run binary in container to check for liveness
Readiness Probe
None (No incoming network traffic)
Jobs Server


Kubernetes Health Checks Mind Map
Kubernetes Cluster
Worker Node
I'm doing Okay!
pod
Hey! Are you fit for interaction?
Hey! Are you capable of starting up?
Hey! Are you alive?
1 Startup Probe apps has started. useful for slow-starting.
2 Liveness Probe container is re-start. help to catch deadlock.
3 Readiness Probe container is ready to accept traffic.
Pod started
Wait periodSeconds
No
Startup Probe Success?
No
last attempt
Yes
Startup FAILURE Pod STOPED
Yes
Wait periodSeconds
Wait periodSeconds
No
No
Liveness FAILURE No Traffic
Yes
last attempt
No
Liveness Probe Success?
Readiness Probe Success?
No
last attempt
Yes
Readiness FAILURE No Traffic
Yes
Wait periodSeconds
Yes
Pod running and receiving traffic
PerfectScale

```
┌──────────────┐
│   Service    │
└──────────────┘
       │  (only Ready pods)
┌──────▼─────────────────────┐
│ Pod                         │
│ ┌───────── Container ─────┐ │
│ │ Liveness Probe           │ │
│ │ Readiness Probe          │ │
│ └─────────────────────────┘ │
└─────────────────────────────┘
          │
          │
       Kubelet
```

🔧 Probe Types Supported

Probe	Method	Example Use
Liveness	exec	Check process/file
Liveness	HTTP	App health endpoint
Liveness	TCP	DB / socket check
Readiness	HTTP	API ready check

📦 YAML Examples

🔹 1. Liveness Probe – Command (`liveness-command.yml`)

```yaml
livenessProbe:
  exec:
    command:
    - cat
    - /tmp/healthy
  initialDelaySeconds: 10
  periodSeconds: 5
```

✔️ Used when:

- Checking file existence
- Checking running process
- Lightweight checks

🔹 2. Liveness Probe – HTTP (NGINX) (`liveness-http-nginx.yml`)

```yaml
livenessProbe:
  httpGet:
    path: /
    port: 80
  initialDelaySeconds: 5
  periodSeconds: 10
```

✔️ Best for:

- Web servers
- REST APIs

🔹 3. Liveness Probe – HTTP Failure (`liveness-http-error.yml`)

```yaml
livenessProbe:
  httpGet:
    path: /healthz
    port: 8080
  failureThreshold: 3
```

❌ If endpoint returns **500 / timeout** → container restarts

🔹 4. Liveness Probe – TCP (`liveness-tcp.yml`)

```
livenessProbe:
  tcpSocket:
    port: 3306
  initialDelaySeconds: 15
  periodSeconds: 20
```

✔️ Ideal for:

- MySQL
- Redis
- Kafka
- Any TCP service

🔹 5. Readiness Probe – HTTP (`readness-http.yml`)

```
readinessProbe:
  httpGet:
    path: /ready
    port: 8080
  initialDelaySeconds: 5
  periodSeconds: 5
```

✔️ Pod receives traffic **ONLY after this passes**

🚀 Apply Manifests

```
kubectl apply -f liveness-command.yml
kubectl apply -f liveness-http-nginx.yml
kubectl apply -f liveness-http-error.yml
kubectl apply -f liveness-tcp.yml
kubectl apply -f readness-http.yml
```

🔎 Verification Commands (SRE Style)

```
kubectl get pods
kubectl describe pod <pod-name>
kubectl get events --sort-by=.metadata.creationTimestamp
```

Watch probe failures live:

```
kubectl logs <pod-name> --previous
```

💥 Failure Scenarios (Important!)

Failure	Result
Liveness fails	Container restarts
Readiness fails	Pod removed from Service
Both fail	Restart + no traffic
Probe misconfigured	CrashLoopBackOff

✅ Best Practices (Production)

✔️ Always use **Readiness** with Services ✔️ Keep Liveness **lightweight** ✔️ Avoid DB checks in Liveness ✔️ Use StartupProbe for slow apps ✔️ Tune `initialDelaySeconds` carefully ✔️ Monitor with Prometheus alerts

🧭 Real-World Recommendation

App Type	Recommended
API	HTTP Liveness + Readiness
Database	TCP Liveness
Legacy App	exec probe
Spring Boot	`/actuator/health`