AWS-DEVOPS

DATE	:	28.04.2025
DT/NT	:	DT
LESSON	:	KUBERNETES
SUBJECT	:	PROBES
BATCH	:	B 303

TECHPRO
EDUCATION

techproeducation.com   +1 (585) 304 29 59

Probes

Probes

- In Kubernetes, a "probe" is a diagnostic performed periodically by the kubelet on a Container to determine its health.
- Kubernetes uses probes to decide how to manage Containers, such as when to restart them or not to send traffic to them if they are not responding.

Types of health checks

- Readiness
- Liveness
- Startup

Readiness Probes

- Readiness probes are designed to let Kubernetes know when your app is ready to serve traffic.
- Kubernetes makes sure the readiness probe passes before allowing a service to send traffic to the pod.
- If a readiness probe starts to fail, Kubernetes stops sending traffic to the pod until it passes.

Readiness Probes

- Readiness probes determine whether or not a container is ready to serve requests. If the readiness probe returns a failed state, then Kubernetes removes the IP address for the container from the endpoints of all Services.
- Developers use readiness probes to instruct Kubernetes that a running container should not receive any traffic. This is useful when waiting for an application to perform time-consuming initial tasks, such as establishing network connections, loading files, and warming caches.

Liveness Probes

- Liveness probes let Kubernetes know if your app is alive or dead. If your app is alive, then
- Kubernetes leaves it alone. If your app is dead, Kubernetes removes the Pod and starts a new one to replace it.

Startup Probes

- Startup probes are designed to let Kubernetes know if your app has started successfully.
- Kubernetes uses the startup probe to know when to begin liveness and readiness checks. It prevents the pod from being killed before it's up and running.
- If the startup probe does not succeed within a specified time, Kubernetes restarts the Pod to try again, according to the specified restart policy.

Probes can be configured to perform checks in three different ways:

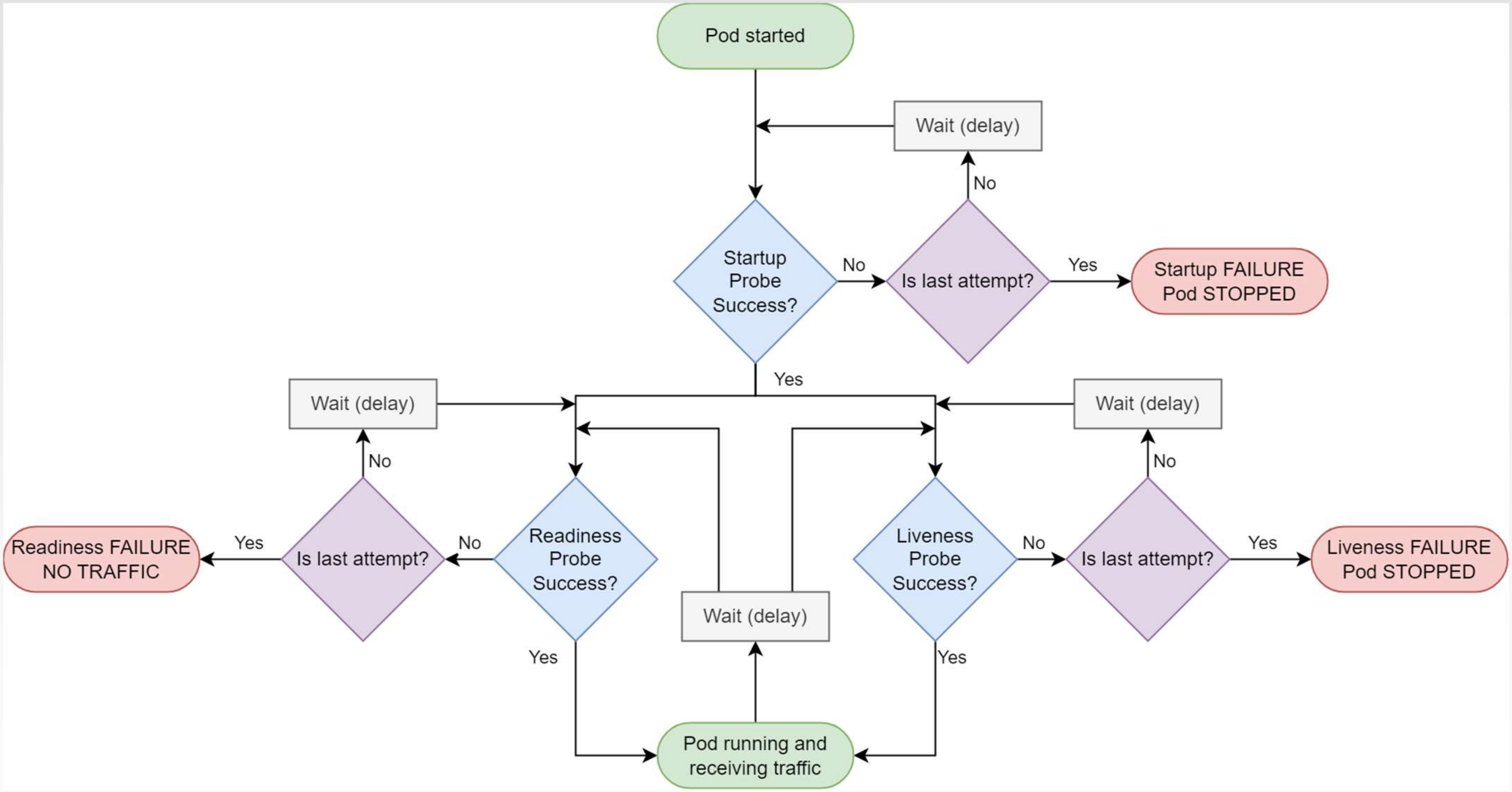
HTTP GET: The probe sends an HTTP GET request to the container.

TCP Socket: The probe attempts to open a TCP connection to the container.

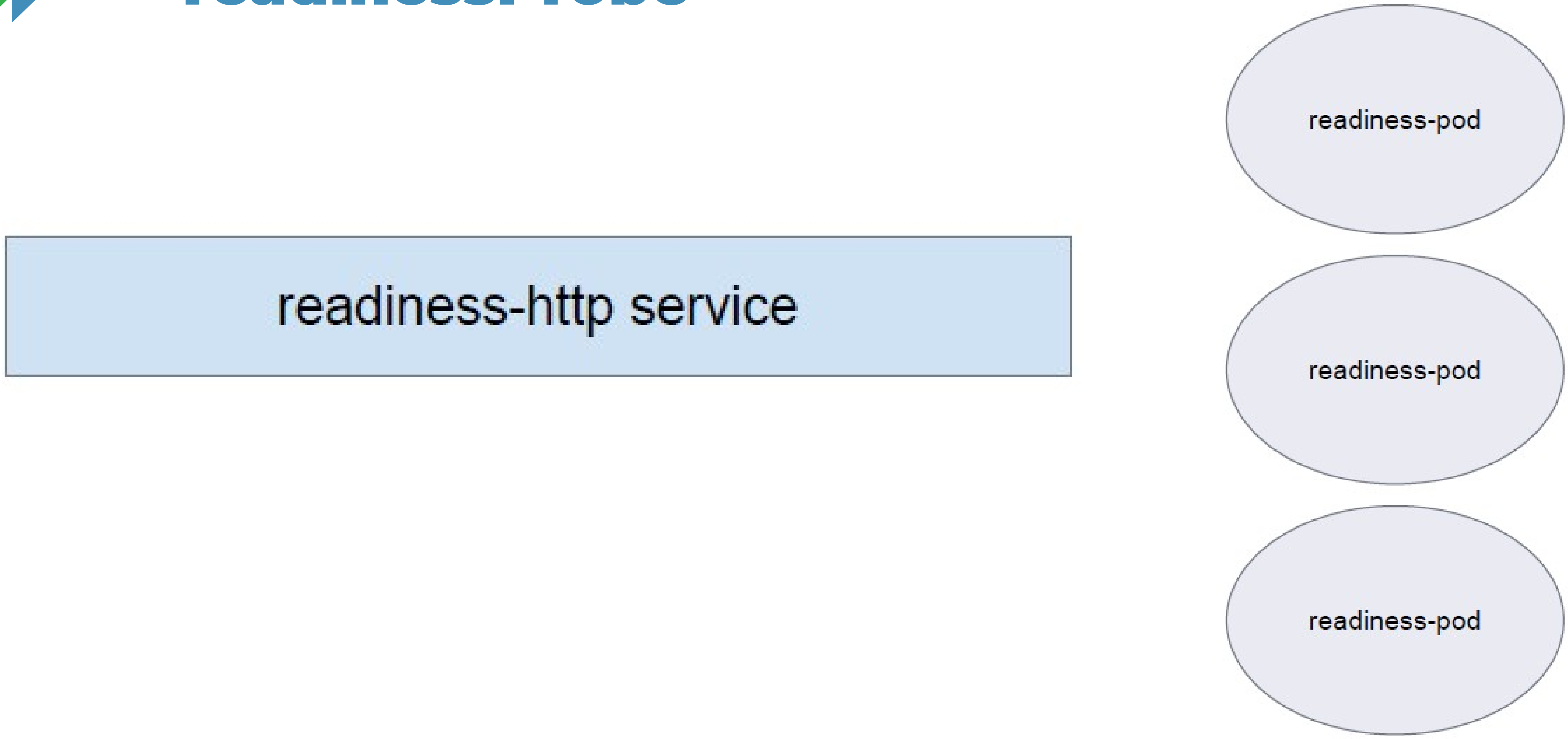
Exec: The probe executes a command inside the container and checks the exit status of the command.

Common Probe Parameters

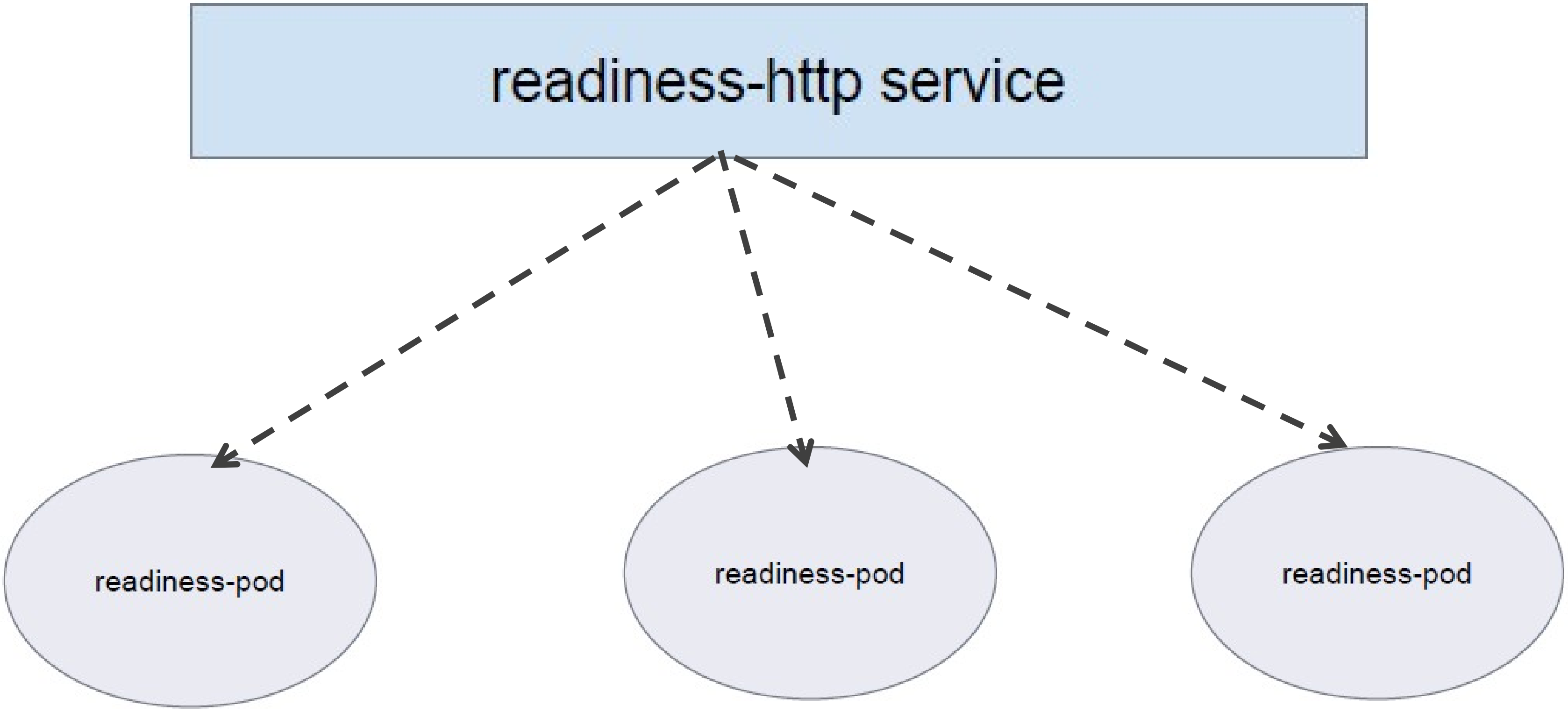
- **initialDelaySeconds:** Seconds after the container started and before probes start. (default: 0)
- **periodSeconds:** Frequency of the pod. (default: 10)
- **timeoutSeconds:** Timeout for the expected response. (default: 1)
- **successThreshold:** How many success results received to transition from failure to a healthy state. (default: 1)
- **failureThreshold:** How many failed results received to transition from healthy to failure state. (default: 3)

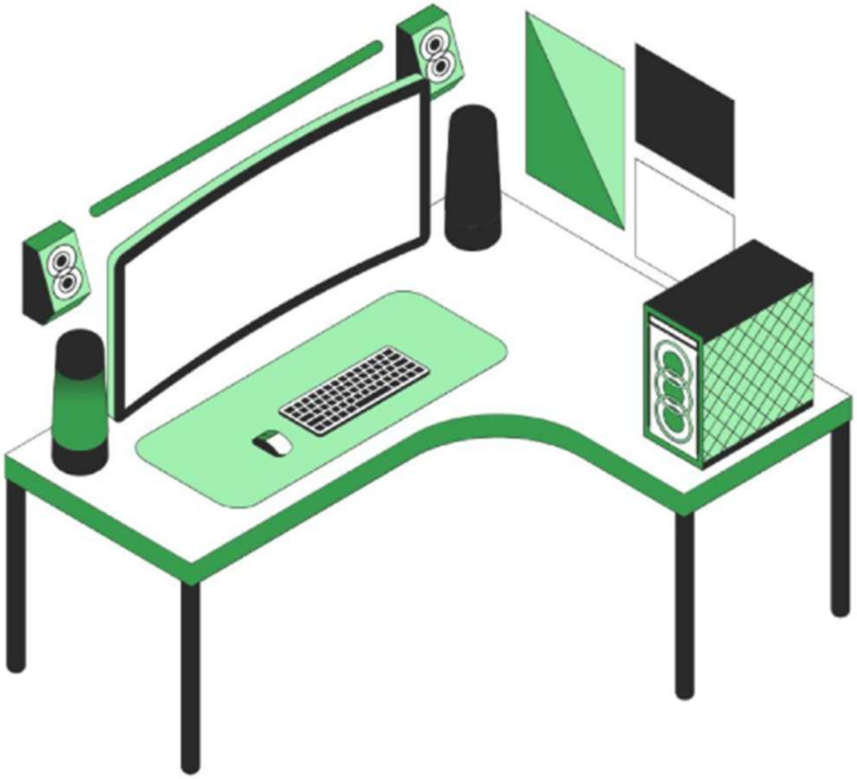

Pod started
Wait (delay)
No
Startup Probe Success?
No
Is last attempt?
Yes
Startup FAILURE Pod STOPPED
Yes
Wait (delay)
Wait (delay)
No
No
Readiness FAILURE NO TRAFFIC
Yes
Is last attempt?
No
Readiness Probe Success?
Liveness Probe Success?
No
Is last attempt?
Yes
Liveness FAILURE Pod STOPPED
Wait (delay)
Yes
Yes
Pod running and receiving traffic

readinessProbe



readinessProbe





Do you have any questions?

Send it to us! We hope you learned something new.