KUBERNETES HEALTH PROBES

- Kubernetes uses health probes to monitor the status of applications inside pods.
- Probes ensure that only healthy pods receive traffic.
- Unhealthy pods are either given time to recover or restarted automatically.
- The kubelet is responsible for executing these health checks.

READINESS PROBE:

1. Determine whether the application is ready to serve traffic.
2. If the probe fails, the Pod is marked **Not Ready** and removed from Service endpoints.
3. The container is not restarted when this probe fails.
4. Prevents traffic from being routed to containers that are temporarily unable to handle requests (e.g., during initialization or due to resource pressure).

Example:

```
readinessProbe:
   httpGet:
     path: /readyz
     port: 8080
   initialDelaySeconds: 5   # Wait 5 seconds after container starts before probing
   periodSeconds: 10        # Probe every 10 seconds
```

LIVENESS PROBE:

1. Used by the kubelet to determine when a container should be restarted.
2. Detect issues like **deadlocks** where the application is running but **not progressing**.
3. Checks whether the container is alive and functioning properly.

4. If the liveness probe fails, the container is automatically restarted.

Example:

```
livenessProbe:
   exec:
     command:
     - cat
     - /tmp/healthy
   initialDelaySeconds: 3  # Start checking 3 seconds after the container starts
   periodSeconds: 5        # Check every 5 seconds
```

STARTUP PROBE:

1. Ensure that the container has enough time to fully initialize.
2. Until this probe succeeds, liveness and readiness probes remain inactive.
3. Ideal for legacy or slow-starting applications with unpredictable initialization times.
4. Overcomes limitations of initialDelaySeconds:
    - Too high → unnecessary wait time.
    - Too low → premature restarts during startup.

```
startupProbe:
 httpGet:
   path: /healthz
   port: 8080
 failureThreshold: 30  # Kubernetes (via Kubelet) will attempt the probe up to 30 times before failing
 periodSeconds: 10     # Probe runs every 10 seconds
```

Practicals:

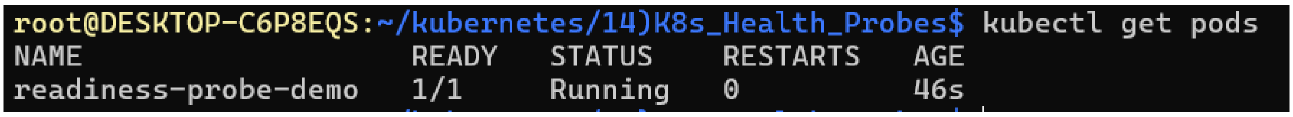
1)Lets Deploy a pod with readiness probe and observe its behaviour:

rp.yaml:

```
apiVersion: v1
kind: Pod
metadata:
  name: readiness-probe-demo
spec:
  containers:
    - name: busybox-app
      image: busybox
      command: ["sh", "-c", "touch /tmp/ready; sleep 3600"] # Creates /tmp/ready file and keeps container running for 1 hour
      readinessProbe:
        exec: # Runs a command inside the container
          command:
            - cat
            - /tmp/ready # Command checks if /tmp/ready file exists
        initialDelaySeconds: 5 # Wait 5 seconds before the first probe
        periodSeconds: 5 # Probe runs every 5 seconds
        failureThreshold: 1 # Mark pod as NotReady after 1 failure
```

A file "ready" is created at /tmp/ location inside the container and readiness probe will check if this file is available.

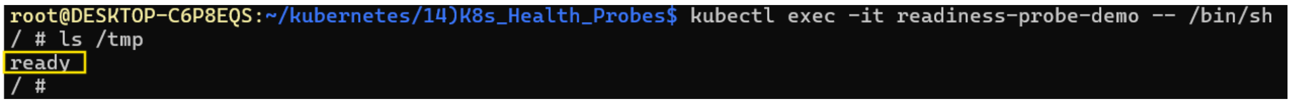
```
kubectl apply -f rp.yaml
```



```
root@DESKTOP-C6P8EQS:~/kubernetes/14)K8s_Health_Probes$ kubectl get pods
NAME                   READY   STATUS    RESTARTS   AGE
readiness-probe-demo   1/1     Running   0          46s
```

Pod is in running state , Readiness probe is able to find /tmp/ready inside the container.

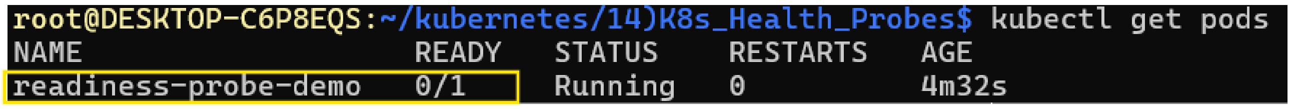
```
kubectl exec -it readiness-probe-demo -- /bin/sh
```



```
root@DESKTOP-C6P8EQS:~/kubernetes/14)K8s_Health_Probes$ kubectl exec -it readiness-probe-demo -- /bin/sh
/ # ls /tmp
ready
/ #
```

Lets delete this file and check the pod:

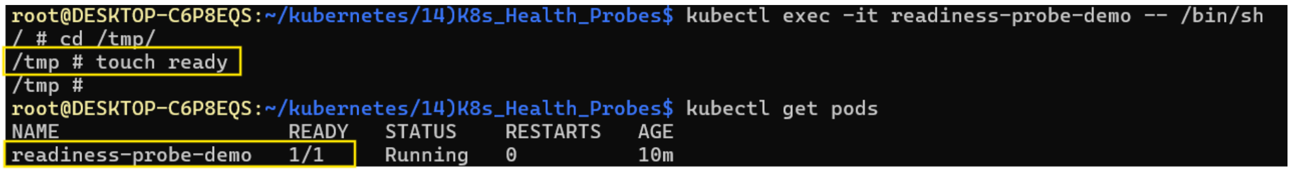
```
rm /tmp/ready
```



```
root@DESKTOP-C6P8EQS:~/kubernetes/14)K8s_Health_Probes$ kubectl get pods
NAME                   READY   STATUS    RESTARTS   AGE
readiness-probe-demo   0/1     Running   0          4m32s
```

- When the readiness probe fails, the pod is marked Not Ready.
- Until the pod becomes Ready again, no traffic is routed to it.

Lets exec into the container and create the file “ready” again:



```
root@DESKTOP-C6P8EQS:~/kubernetes/14)K8s_Health_Probes$ kubectl exec -it readiness-probe-demo -- /bin/sh
/ # cd /tmp/
/tmp # touch ready
/tmp #
root@DESKTOP-C6P8EQS:~/kubernetes/14)K8s_Health_Probes$ kubectl get pods
NAME                   READY   STATUS    RESTARTS   AGE
readiness-probe-demo   1/1     Running   0          10m
```

2)Lets Deploy a pod with Liveness probe and observe its behaviour:

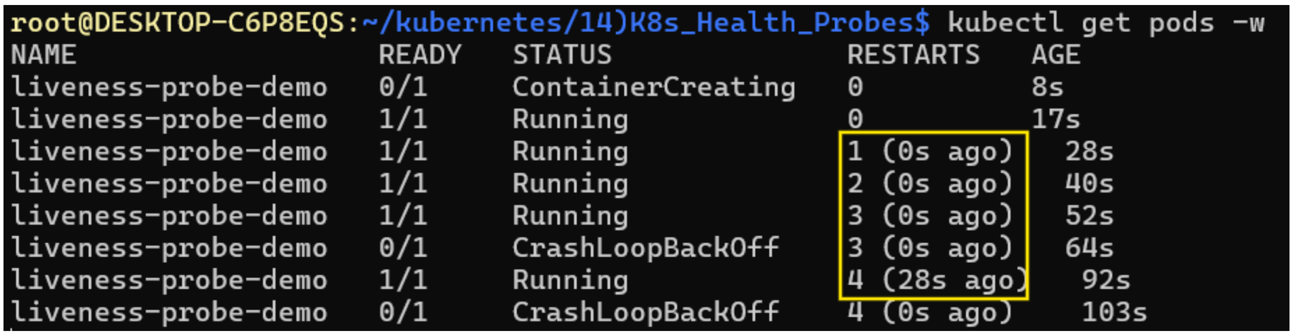
lp.yaml:

```
apiVersion: v1
kind: Pod
metadata:
  name: liveness-probe-demo
  labels:
    test: liveness
spec:
  containers:
    - name: liveness
      image: registry.k8s.io/e2e-test-
images/agnhost:2.40
      args:
        - liveness
      livenessProbe:
        httpGet:
          path: /healthz # Endpoint to check
for health status
```

```
            port: 8080 # Port where the
application is running
            httpHeaders:
              - name: Custom-Header # Custom
header name
                value: Awesome # Custom header
value
          initialDelaySeconds: 3 # Wait 3 seconds
before starting probes
          periodSeconds: 3 # Run the probe every
3 seconds
          timeoutSeconds: 1 # Wait 1 second for a
response before timing out
          failureThreshold: 1 # After 1 failure,
restart the container
```

- For Http get request, exit code from 200 – 399 shows successful status.
- After 10 seconds, the /healthz endpoint returns a status of 500, indicating failure. **This is how the application in the container is configured.**

```
kubectl apply -f lp.yaml
```



```
root@DESKTOP-C6P8EQS:~/kubernetes/14)K8s_Health_Probes$ kubectl get pods -w
NAME                   READY   STATUS              RESTARTS      AGE
liveness-probe-demo    0/1     ContainerCreating   0             8s
liveness-probe-demo    1/1     Running             0             17s
liveness-probe-demo    1/1     Running             1 (0s ago)     28s
liveness-probe-demo    1/1     Running             2 (0s ago)     40s
liveness-probe-demo    1/1     Running             3 (0s ago)     52s
liveness-probe-demo    0/1     CrashLoopBackOff    3 (0s ago)     64s
liveness-probe-demo    1/1     Running             4 (28s ago)     92s
liveness-probe-demo    0/1     CrashLoopBackOff    4 (0s ago)      103s
```

- The liveness probe detects the failure and restarts the container.

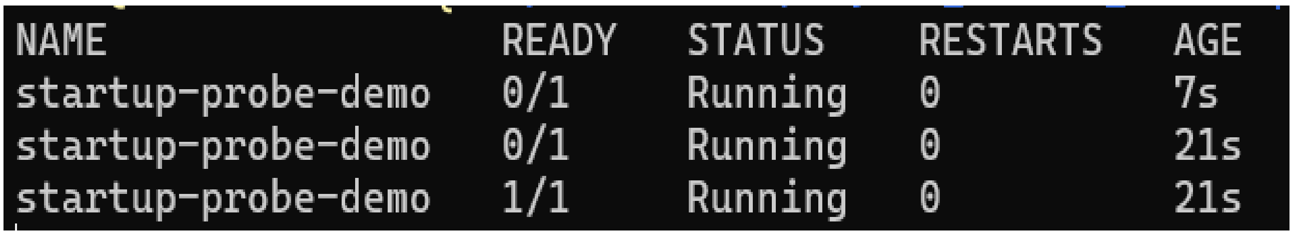
Kubectl.

2)Lets Deploy a pod with Startup probe and observe its behaviour:

Sp.yaml:

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: startup-probe-demo
spec:
  containers:
    - name: tcp-app
      image: ubuntu:latest
      command:
        - sh
        - -c
        - |
          apt update && \
          apt install -y netcat-openbsd && \
          nc -l -p 9444 && \
          sleep 3600
      startupProbe:
        tcpSocket:
          port: 9444 # Checks TCP server
accessibility on port 9444
        failureThreshold: 15 # Allows up to 15
failed attempts
        periodSeconds: 5 # Probe runs every 5
seconds
```

- tcpSocket.port: 9444 → Kubernetes probes port 9444 to check if TCP server is up.
- failureThreshold: 15 → Allows up to 15 failed attempts before marking container as failed.
- periodSeconds: 5 → Probe runs every 5 seconds.
- Container has 75 seconds to start. If it doesn't, Container will be restarted.



```
NAME                 READY   STATUS    RESTARTS   AGE
startup-probe-demo   0/1     Running   0          7s
startup-probe-demo   0/1     Running   0          21s
startup-probe-demo   1/1     Running   0          21s
```