5. Probe

❖ Probe란?

- 컨테이너에 의해서 주기적으로 실행되는 진단(diagnosis)기능

❖ Probe의 종류

- Liveness Probe : 앱이 살아있는지를 주기적으로 확인
  - 애플리케이션의 상태를 확인해서 서버, 컨테이너가 정상적으로 작동중인지를 검사
  - Pod는 Running 상태지만 애플리케이션에서는 문제가 발생하는 경우를 탐지
  - 사용 예) 애플리케이션의 Deadlock, 메모리 오버플로우 등이 발생했을 경우 Pod를 재시작하려고 할 때
- Readiness Probe : 앱이 준비가 되었을 때 서비스가 트래픽을 전달하도록 함
  - Pod 내의 컨테이너가 요청을 처리할 준비가 되었는지 확인
  - Pod가 새롭게 배포되었더라도 애플리케이션이 요청을 처리할 준비가 되어있지 않을 수 있음.
  - 만일 요청을 처리할 수 있는 상태가 아니라면 서비스가 Pod로 요청을 전달하지 않도록 설정
  - 사용 예) Pod는 시작되었지만 애플리케이션이 시작될 때까지 트래픽이 라우팅되지 않도록 하고 싶을 때
- Startup Probe : 앱이 정상적으로 기동되었는지를 확인함
  - 컨테이너 내부의 애플리케이션이 시작되었는지를 확인
  - Startup Probe가 정상 상태가 되기 전까지 다른 Probe들은 작동되지 않음
  - Startup Probe가 실패하면 kubelet이 Pod를 종료시키고 재시작 정책(Restart policy)에 의해 처리됨
  - 사용 예) 서비스를 시작하는데 오랜 시간이 필요하거나 시간이 불규칙적인 Pod를 설정할 때 사용

5. Probe

❖Handler

- Pod 의 상태를 어떻게 확인할 것인지를 지정하는 방법

❖Handler 종류

- exec
  - 컨테이너 내에서 지정한 명령어를 실행하고 종료후의 상태코드가 0이면 성공으로 간주
- tcpSocket
  - 지정한 IP 주소 포트로 TCP 연결을 시도해 성공 여부를 확인
- httpGet
  - 지정된 주소와 포트로 HTTP GET 요청을 시도해 HTTP 응답 Status가 200~399에 해당하면 성공으로 간주

5. Probe

❖Probe 옵션

- initialDelaySeconds
  - 첫번째 진단을 할 때까지 대기하는 시간(기본값:0s)
- periodSeconds
  - 진단을 수행하는 주기(기본값:10s)
- failureThreshold
  - 몇 번의 진단을 실패하면 정말 실패로 판단할 것인지 횟수를 지정(기본값:3)
- successThreshold
  - 진단이 성공했다고 판단하는 최소 연속 성공 횟수(기본값:1)
  - startup, liveness probe 는 반드시 1라야 함
- timeoutSeconds
  - 진단 시도 후 실패라고 간주하는 timeout 시간(기본값: 1s)
- terminationGracePeriodSeconds
  - kubelet이 진단에 실패한 컨테이너의 종료를 트리거한 후 해당 컨테이너를 강제로 중지하는 사이에 대기하도록 설정한 유예시간(기본값: 30s)

5.1 liveness probe

❖ liveness probe 를 위한 예제1

- 사용할 도커 이미지
  - stepanowon/nodeapp-liveness:1.0.0
  - Mac환경을 위해 멀티 플랫폼 빌드되어 있음
  - 예제는 다음 주소에 있음
    - github.com/stepanowon/nodeapp-liveness
- 예제 코드
  - /healthz 엔드포인트
    - 애플리케이션 시작 후 20초가 지나면 500 오류 발생하도록 작성
    - health 라는 요청 헤더가 check 인 경우에만 실행함

```
// server.js
const express = require('express');
const app = express();
const port = 8080;
let startTime;

app.get('/healthz', (req, res) => {
    const header = req.headers["health"];
    if (header === "check") {
        let duration = Date.now() - startTime.getTime();
        if (duration > 20000) {
            res.status(500).json({
                status: "fail",
                message: "server : not available"
            })
        } else {
            res.json({ status: "ok", message: "server : available." })
        }
    } else {
        res.json({ status: "ok", message: "server : available." })
    }
});

......(생략)
app.listen(port, () => {
    console.log(`liveness probe test server is running on port ${port}`);
    startTime = new Date();
});
```

5.1 liveness probe

❖liveness1.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nodeapp-liveness
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nodeapp
  template:
    metadata:
      labels:
        app: nodeapp
    spec:
      containers:
      - name: liveness1
        image: stepanowon/nodeapp-liveness:1.0.0
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
          protocol: TCP
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8080
            httpHeaders:
            - name: health
              value: check
          initialDelaySeconds: 3
          periodSeconds: 3
---
apiVersion: v1
kind: Service
metadata:
  name: nodeapp-lb
  namespace: default
spec:
  type: LoadBalancer
  ports:
    - name: lb-port
      port: 80
      targetPort: 8080
  selector:
    app: nodeapp
```

5.1 liveness probe

❖테스트

- 리소스 생성
  - kubectl apply -f liveness1.yaml
- 실행 여부 확인
  - curl -H "health: check" http://192.168.56.51/healthz
  - 처음에는 정상 실행되지만 20초후부터 500 오류와 함께 에러 메시지 응답

```
bpro :: k8s-cicd-sample/k8s/probe » curl -H "health: check" http://192.168.56.51/healthz
{"status":"ok","message":"server : available."}%
bpro :: k8s-cicd-sample/k8s/probe » curl -H "health: check" http://192.168.56.51/healthz
{"status":"fail","message":"server : not available"}%
```

- kubectl describe pods 명령어를 실행하여 Events 항목 조회
  - Warning Unhealthy 2m49s (x9 over 4m55s) kubelet Liveness probe failed: HTTP probe failed with statuscode: 500
  - Pod가 재시작되었음을 확인
- 테스트 후 리소스 정리
  - kubectl delete -f liveness1.yaml

5.1 liveness probe

❖liveness2.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: liveness2
spec:
  containers:
  - name: liveness2
    image: multiarch/busybox:amd64-slim              #맥에서는 arm64-slim
    args:          # 파일을 생성한 뒤 30초 후에 파일 삭제
    - /bin/sh
    - -c
    - touch /tmp/healthy; sleep 30; rm -f /tmp/healthy; sleep 600
    livenessProbe:
      exec:
        command:          # cat /tmp/healty 명령이 성공하면 0을 리턴
        - cat
        - /tmp/healthy
      initialDelaySeconds: 5
      periodSeconds: 5
```

출처: https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/

5.1 liveness probe

❖liveness2 예제 실행 결과

- kubectl describe pods liveness2 명령으로 상세 정보 조회
  - Events 정보 확인

```
Events:
  Type     Reason     Age                  From               Message
  ----     ------     ----                 ----               -------
  Normal   Scheduled  37m                  default-scheduler  Successfully assigned default/liveness2 to worker1
  Normal   Killing    33m (x3 over 36m)    kubelet            Container liveness2 failed liveness probe, will be restarted
  Normal   Created    33m (x4 over 37m)    kubelet            Created container liveness2
  Normal   Started    33m (x4 over 37m)    kubelet            Started container liveness2
  Warning  Unhealthy  31m (x13 over 36m)   kubelet            Liveness probe failed: cat: can't open '/tmp/healthy': No such file or direct
ory
  Warning  BackOff    7m1s (x79 over 29m)  kubelet            Back-off restarting failed container liveness2 in pod liveness2_default(c1226
908-b3ce-4070-9235-4efbe8ae96f5)
  Normal   Pulled     103s (x14 over 37m)  kubelet            Container image "multiarch/busybox:arm64-slim" already present on machine
```

  - Liveness probe failed : cat: can't open .....
- 테스트 완료 후 리소스 제거
  - kubectl delete -f liveness2.yaml

5.2 readiness probe

❖ 예제 코드

```
// stepanowon/nodeapp-readiness:1.0.0으로 제공
// 예제파일 : github.com/stepanowon/nodeapp-readiness
const express = require('express');

const app = express();
const port = 8080;
let startTime;

app.get('/healthz', (req, res) => {
   let duration = Date.now() - startTime.getTime();

   //서버 시작 후 1분 동안은 사용 불가능한 상태로 대기
   if (duration < 60000) {
      res.status(500).json({
         status: "fail",
         message: "server : not available"
      })
   } else {
      res.json({
         status: "ok",
         message: "server : available."
      })
   }
});
```

```
app.get('/', (req, res) => {
    return res.status(200).send(`
       <div>
         <h2> nodeapp-readiness </h2>
         <h2> Version : 1.0.0 </h2>
         <h2> hostname: ${os.hostname()} </h2>
       </div>
    `);
})

app.listen(port, () => {
    console.log(`readiness probe : port ${port}`);
    startTime = new Date();
})
```

5.2 readiness probe

❖readiness.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nodeapp-readiness
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nodeapp
  template:
    metadata:
      labels:
        app: nodeapp
    spec:
      containers:
      - name: readiness
        image: stepanowon/nodeapp-readiness:1.0.0
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
          protocol: TCP
        readinessProbe:
          httpGet:
            path: /healthz
            port: 8080
          initialDelaySeconds: 5
          periodSeconds: 5
---
apiVersion: v1
kind: Service
metadata:
  name: nodeapp-lb
  namespace: default
spec:
  type: LoadBalancer
  ports:
    - name: lb-port
      port: 80
      targetPort: 8080
  selector:
    app: nodeapp
```

5.2 readiness probe

❖테스트

- kubectl apply -f readiness.yaml 명령어 실행
- 1분 동안은 Pod가 running 이긴 하지만 0/1 로 나타남
- Loadbalancer로 요청해도 응답없음
  - curl http://192.168.56.51

```
bpro :: k8s-cicd-sample/k8s/probe » k apply -f readiness.yaml
deployment.apps/nodeapp-readiness created
service/nodeapp-lb created
bpro :: k8s-cicd-sample/k8s/probe » k get all
NAME                                    READY   STATUS    RESTARTS   AGE
pod/nodeapp-readiness-5cc89fd495-wtq4x   0/1     Running   0          4s

NAME                 TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)        AGE
service/kubernetes   ClusterIP      10.96.0.1        <none>          443/TCP        25h
service/nodeapp-lb   LoadBalancer   10.102.253.166   192.168.56.51   80:31619/TCP   4s

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nodeapp-readiness   0/1     1            0           4s

NAME                                          DESIRED   CURRENT   READY   AGE
replicaset.apps/nodeapp-readiness-5cc89fd495   1         1         0       4s
```

5.2 readiness probe

❖테스트(이어서)

- 1분 후에 정상 실행

```
bpro :: k8s-cicd-sample/k8s/probe » curl http://192.168.56.51

        <div>
          <h2> nodeapp-readiness </h2>
          <h2> Version : 1.0.0 </h2>
        </div>
```

- 리소스 삭제
  - kubectl delete -f readiness.yaml

5.3 starup probe

❖startup probe의 정의는 5절 개요 참조

❖startup probe 가 필요한 이유

- 애플리케이션이 시작될 때 시간이 오래 걸리면 liveness probe가 진단을 하면서 Pod의 재시작이 무한 반복될 수 있음
- 이런 상황을 방지하기 위해 startup probe 사용
- startup probe가 정상으로 진단되기 전까지는 readiness probe, liveness probe가 작동되지 않음
  - 반대로 startup probe가 성공으로 진단되면 바로 liveness, readiness probe가 진단을 시작함
- 애플리케이션의 시작 시간이 길 때의 선택지
  - liveness probe만 설정하고 initialDelaySeconds를 길게 가져가는 방법
    - 만일 애플리케이션의 시작이 예외적으로 늦어지면 Pod가 무한 재시작할 수 있음
  - startup probe와 liveness probe를 함께 설정
    - startup probe가 성공으로 진단될 때까지는 liveness probe가 진단을 시작하지 않으므로 pod의 무한 반복 재시작을 방지

5.3 startup probe

❖startup probe 테스트 예제 : stepanowon/nodeapp-startup:1.0.0

```
// 예제 : github.com/stepanowon/nodeapp-startup
const express = require('express');

const app = express();
const port = 8080;
let startTime;

// 애플리케이션이 시작하면 60초동안은 500에러 유발
app.get('/healthz', (req, res) => {
    let duration = Date.now() - startTime.getTime();
    if (duration < 60000) {
        res.status(500).json({
            status: "fail",
            message: "server : not available"
        })
    } else {
        res.json({ status: "ok", message: "server : available." })
    }
});

app.listen(port, () => {
    console.log(`startup probe : port ${port}`);
    startTime = new Date();
});
```

5.3 startup probe

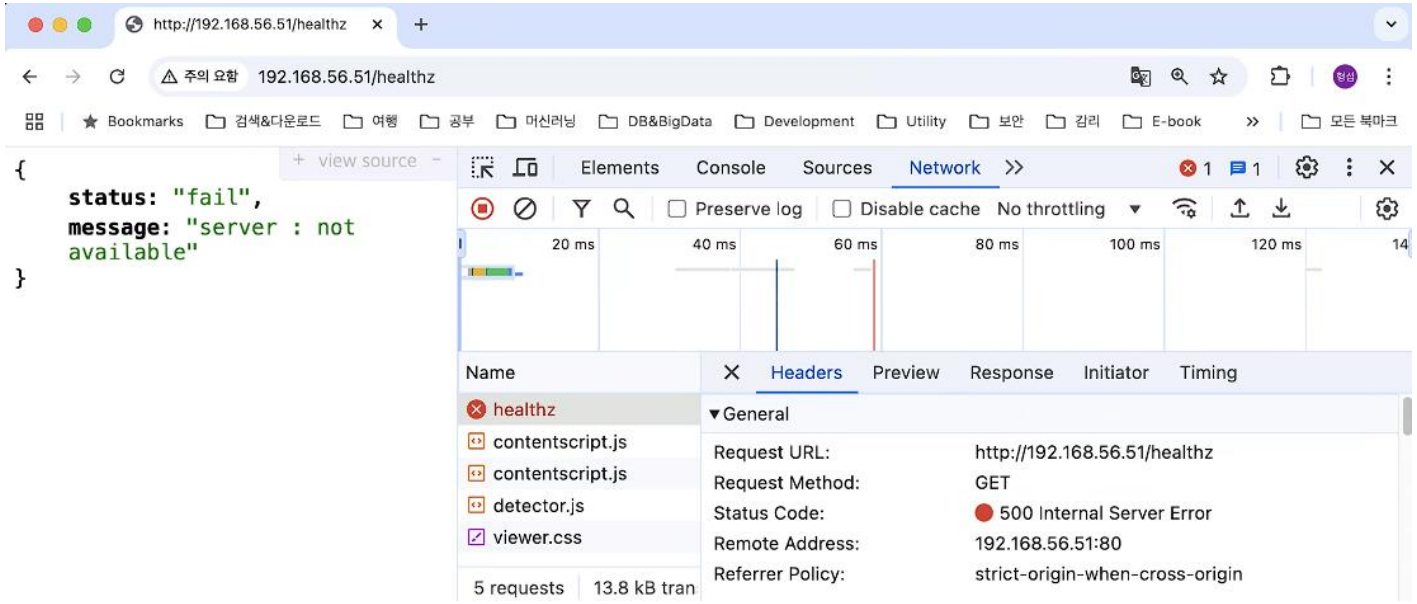
❖startup.yaml - liveness probe만 적용

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nodeapp-startup
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nodeapp
  template:
    metadata:
      labels:
        app: nodeapp
    spec:
      containers:
      - name: startup1
        image: stepanowon/nodeapp-startup:1.0.0
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
          protocol: TCP
        # 3초의 지연 후에 3초 간격으로 진단 시작
        # 두번 실패하면 진단 실패로 판단
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8080
          initialDelaySeconds: 3
          periodSeconds: 3
          failureThreshold: 2
---
apiVersion: v1
kind: Service
metadata:
  name: nodeapp-lb
  namespace: default
spec:
  type: LoadBalancer
  ports:
    - name: lb-port
      port: 80
      targetPort: 8080
  selector:
    app: nodeapp
```

5.3 startup probe

❖테스트

- kubectl apply -f startup.yaml
- 브라우저를 열어서 http://192.168.56.51/healthz 로 요청하여 500 오류 발생 확인



- kubectl get pods 명령을 실행하여 pod가 재시작되고 있음을 확인

```
bpro :: k8s-cicd-sample/k8s/probe » kubectl get pods
NAME                                READY   STATUS    RESTARTS      AGE
nodeapp-startup-799c4cfbd7-fbcc9    1/1     Running   6 (23s ago)   5m3s
```

5.3 startup probe

❖startup.yaml - startup probe 적용하여 문제 해결

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nodeapp-startup
spec:
  ......(생략)
  template:
    ......(생략)
    spec:
      containers:
      - name: startup1
        image: stepanowon/nodeapp-startup:1.0.0
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
          protocol: TCP
```

```
        # 3초의 지연 후에 3초 간격으로 진단 시작
        # 두번 실패하면 진단 실패로 판단
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8080
          initialDelaySeconds: 3
          periodSeconds: 3
          failureThreshold: 2
        # 10초간격으로 진단하여 10번 실패하면 실패로 간주
        startupProbe:
          httpGet:
            path: /healthz
            port: 8080
          periodSeconds: 10
          failureThreshold: 10
---
......(생략)
```

5.3 startup probe

❖startup probe 테스트

- 시작 후 60초 동안
  - 브라우저로 http://192.168.56.51/healthz 로 요청했을 때 응답 없음(사이트에 연결할 수 없음) 상태가 60초간 지속됨
  - kubectl get pods 명령 실행 했을 때 running 상태이긴 하지만 pod가 정상상태가 아님

```
bpro :: k8s-cicd-sample/k8s/probe » kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nodeapp-startup-675fdfd6f-d5wvh     0/1     Running   0          5s
```

- 60초 이후 startup probe가 정상상태가 되면
  - 브라우저에서 정상 응답
  - kubectl get pods 명령 실행 시 정상적으로 running 상태

```
bpro :: k8s-cicd-sample/k8s/probe » kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nodeapp-startup-675fdfd6f-d5wvh     1/1     Running   0          2m38s
```