[Istio] Timeout & Retry

Istio VirtualService 객체를 통해 대상 서비스에 Timeout 과 Retry 를 설정하고, 정상 동작하는지 확인한다.

Instruction

Istio Timeout & Retry

Timeout

카프카 설치 및 주문(Order) 서비스 배포

```
helm repo add incubator https://charts.helm.sh/incubator
helm repo update
kubectl create namespace kafka
helm install my-kafka --namespace kafka incubator/kafka

kubectl get svc my-kafka -n kafka
```

tutorial 네임스페이스에 **Istio Activation**

- 네임스페이스가 없을 시, 생성 후 실행

```
kubectl label namespace tutorial istio-injection=enabled --overwrite
```

Timeout : Fail-Fast 를 통한 **Caller** 자원 보호(장애전파 차단)

- Order Aggregate(Order.java)에 저장전 Thread.sleep 삽입

```
@PrePersist
  public void onPrePersist(){
    try {
        Thread.currentThread().sleep((long) (800 + Math.random() *
220));
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
  }
```

Dockerizing (Image Build, and Push)

- Order 프로젝트 루트로 콘솔 이동

```
mvn package
docker build -t [IMAGE_NAME] .
docker push [IMAGE_NAME]
```

tutorial 네임스페이스에 **Order** 배포

```
kubectl apply -f - <<EOF
  apiVersion: apps/v1
  kind: Deployment
  metadata:
    name: order
    namespace: tutorial
    labels:
      app: order
  spec:
    replicas: 1
    selector:
      matchLabels:
        app: order
    template:
      metadata:
        labels:
          app: order
      spec:
        containers:
          - name: order
            image: 879772956301.dkr.ecr.ap-northeast-2.amazonaws.com/user02-order:v2
            ports:
              - containerPort: 8080
            resources:
              limits:
                cpu: 500m
              requests:
                cpu: 200m
EOF
```

Order 서비스 생성

```
kubectl expose deploy order --port=8080 -n tutorial
```

주문서비스 **Timeout** 이 설정된 **VritualService** 생성

```
kubectl apply -f - <<EOF
  apiVersion: networking.istio.io/v1alpha3
  kind: VirtualService
  metadata:
    name: ye-order-network-rule
    namespace: tutorial
  spec:
    hosts:
      - order
    http:
```

```
    - route:
      - destination:
          host: order
      timeout: 3s
EOF
```

Siege 를 통한 Order 서비스 부하 주입

```
kubectl run siege --image=apexacme/siege-nginx -n tutorial
kubectl exec -it pod/siege -c siege -n tutorial -- /bin/bash
siege -c30 -t20S -v --content-type "application/json"
'http://order:8080/orders POST {"productId": "1001", "qty":5}'
```

- **Order 서비스에 설정된 Timeout 임계치를 초과하는 순간, Istio 에서 연결을 차단하는 것을 확인**

Retry

[skip] tutorial 네임스페이스에 Delivery 서비스 배포

```
kubectl apply -f - <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: delivery
  namespace: tutorial
  labels:
    app: delivery
spec:
  replicas: 1
  selector:
    matchLabels:
      app: delivery
  template:
    metadata:
      labels:
        app: delivery
    spec:
      containers:
        - name: delivery
          image: username/delivery:v1
          ports:
            - containerPort: 8080
          resources:
            limits:
              cpu: 500m
          readinessProbe:
```

```
            cpu: 200m
EOF
```

[skip] Delivery 서비스 생성

```
kubectl expose deploy delivery --port=8080 -n tutorial
```

Order 서비스에 Retry Rule 추가

```
kubectl apply -f - <<EOF
  apiVersion: networking.istio.io/v1alpha3
  kind: VirtualService
  metadata:
    name: vs-order-network-rule
    namespace: tutorial
  spec:
    hosts:
    - order
    http:
    - route:
      - destination:
          host: order
      timeout: 3s
      retries:
        attempts: 3
        perTryTimeout: 2s
        retryOn: 5xx,retriable-4xx,gateway-error,connect-
failure,refused-stream
EOF
```

[skip] Delivery 서비스를 정지

```
kubectl scale deploy delivery --replicas=0 -n tutorial
```

Order 서비스 API 호출

```
kubectl exec -it pod/siege -c siege -n tutorial --/bin/bash
http http://order:8080/orders/ productId=1001 qty=5
http DELETE http://order:8080/orders/1
# Error Log view.
kubectl logs pod/order[객체] -c order -n tutorial
```

- **Jaeger 접속(http://tracing svc EXTERNAL-IP :80) 후, Retry 횟수 확인**
- **검색조건: Service : order.tutorial,**
- **Operation: delivery.tutorial.svc.cluster.local:8080/***
- **검색결과 설정된 Retry 만큼의 재시도 요청이 발생한 것을 확인한다.**

CheckPoints

1. 모든 요구사항을 만족하는가