[실습 5] 클러스터 롤링 업데이트

LAB

Kubernetes Cluster

- 실습을 위한 쿠버네티스 클러스터 구성 정보 확인

```
# LAB005 실습을 위한 경로로 이동
$ cd ~/labhome/lab005

$ kubectl cluster-info
Kubernetes master is running at https://192.168.99.100:8443
KubeDNS is running at https://192.168.99.100:8443/api/v1/namespaces/kube-
system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

# 만약 LAB 실행 중 문제가 있을 경우 아래 두가지 명령어를 이용해 복구할 수 있습니다.
$ labctl --help
Please use corret option [restore|rebuild]

   labctl restore: Quick lab restore
   labctl rebuild: Complete lab rebuild
```

Deployment

Deployment 명세서 예제 내용 확인 및 배포 연습 Deployment 를 통해 Pod, ReplicaSet 이 한번에 같이 배포됨을 확인

```
# LAB005 실습을 위한 경로로 이동
$ cd ~/labhome/lab005

# 이전 실습에서 사용했던 hello-app-rs 내용을 확인합니다.
$ cat hello-app-rs.yml
apiVersion: apps/v1
```

```
kind: ReplicaSet
metadata:
  name: hello-app-rs
  labels:
    app: hello-app
    tier: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      app: hello-app
      tier: frontend
  template:
    metadata:
      labels:
        app: hello-app
        tier: frontend
    spec:
      containers:
      - name: hello-app
        image: gcr.io/google-samples/hello-app:1.0
        ports:
          - containerPort: 8080

# 동일한 hello-app 을 이번에는 deploy 형태로 배포하는 yml 파일입니다.
$ cat hello-app-deploy.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-app-deploy
  labels:
    app: hello-app
    tier: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      app: hello-app
      tier: frontend
  template:
    metadata:
      labels:
        app: hello-app
        tier: frontend
    spec:
      containers:
      - name: hello-app
        image: gcr.io/google-samples/hello-app:1.0
        ports:
          - containerPort: 8080

# hello-app-deploy 를 생성 후 결과 내용을 확인 합니다.
$ kubectl create -f hello-app-deploy.yml
```

```
deployment.apps/hello-app-deploy created

$ kubectl get deploy
NAME               DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
hello-app-deploy   3         3         3            3           5s

$ kubectl describe deploy hello-app-deploy
Name:                   hello-app-deploy
Namespace:              default
CreationTimestamp:      Tue, 21 Aug 2018 08:33:18 +0900
Labels:                 app=hello-app
                        tier=frontend
Annotations:            deployment.kubernetes.io/revision=1
Selector:               app=hello-app,tier=frontend
Replicas:               3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=hello-app
           tier=frontend
  Containers:
   hello-app:
    Image:        gcr.io/google-samples/hello-app:1.0
    Port:         8080/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>

# hello-app-deploy 를 통해 만들어진 replicaset 과 pod 를 확인 합니다.
$ kubectl get rs
NAME                          DESIRED   CURRENT   READY     AGE
hello-app-deploy-77d8794f48   3         3         3         1m

$ kubectl get pod
NAME                                READY     STATUS    RESTARTS   AGE
hello-app-deploy-77d8794f48-896nh   1/1       Running   0          1m
hello-app-deploy-77d8794f48-r6ff4   1/1       Running   0          1m
hello-app-deploy-77d8794f48-zjqdb   1/1       Running   0          1m

# 위에서 배포한 hello-app-deploy 에 하나의 접근 경로를 제공하기 위하여 Service 를 생성합니다.
$ cat hello-app-svc.yml
apiVersion: v1
kind: Service
metadata:
  name: hello-app-svc
  labels:
    app: hello-app
spec:
  selector:
    app: hello-app
    tier: frontend
```

```
  ports:
  - port: 80
    targetPort: 8080

# hello-app-svc 생성 후 부여된 ClusterIP 를 확인합니다.
$ kubectl create -f hello-app-svc.yml
service/hello-app-svc created

$ kubectl get svc
NAME            TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)   AGE
hello-app-svc   ClusterIP   10.99.54.77   <none>        80/TCP    5s
kubernetes      ClusterIP   10.96.0.1     <none>        443/TCP   9h

# hello-app-svc 를 통해 정상적으로 hello-app-deploy 접근할 수 있는지를 확인합니다.
$ kubectl run busyboxplus --image=radial/busyboxplus:curl -i --tty --rm

$ kubectl run busyboxplus --image=radial/busyboxplus:curl -i --tty --rm
If you don't see a command prompt, try pressing enter.
[ root@busyboxplus-5697648fcc-lwrrw:/ ]$ curl 10.99.54.77
Hello, world!
Version: 1.0.0
Hostname: hello-app-deploy-77d8794f48-r6ff4
[ root@busyboxplus-5697648fcc-lwrrw:/ ]$ curl 10.99.54.77
Hello, world!
Version: 1.0.0
Hostname: hello-app-deploy-77d8794f48-896nh
[ root@busyboxplus-5697648fcc-lwrrw:/ ]$ curl 10.99.54.77
Hello, world!
Version: 1.0.0
Hostname: hello-app-deploy-77d8794f48-zjqdb

# hello-app-deploy 에 접근 가능함을 확인하고, busybox 를 종료합니다.
[ root@busyboxplus-5697648fcc-lwrrw:/ ]$ exit
Session ended, resume using 'kubectl attach busyboxplus-5697648fcc-lwrrw -c busyboxplus
-i -t' command when the pod is running
deployment.apps "busyboxplus" deleted

# 현재 상태를 유지하고 다음 실습 단계로 넘어갑니다.
```

Rolling-Update

- 배포된 Deployment 의 컨테이너 이미지를 교체
- 새로운 이미지를 포함한 Pod 를 롤링 업데이트

```
# 이전 실습 단계에서 계속

# 앞선 hello-app-deploy 에서는 hello-app:v1 을 배포하였습니다.
# 여기서는 Rolling Update 를 통해서 서비스를 중단하지 않고 새버전을 배포해보도록 하겠습니다.

# hello-app-deploy 에서 StrategyType 이 RollingUpdate 임을 확인 합니다.
```

```
$ kubectl describe deploy hello-app-deploy | grep RollingUpdate
StrategyType:           RollingUpdate
RollingUpdateStrategy:  25% max unavailable, 25% max surge


# 이번에는 -o yaml 옵션을 이용해 오브젝트의 현재 명세서를 yaml 형태로 확인합니다.
# 컨테이너 이미지가 gcr.io/google-samples/hello-app:1.0 임을 알 수 있습니다.

$ kubectl get deploy hello-app-deploy -o yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: 2018-08-20T23:33:18Z
  generation: 1
  labels:
    app: hello-app
    tier: frontend
  name: hello-app-deploy
  namespace: default
  resourceVersion: "1480"
  selfLink: /apis/extensions/v1beta1/namespaces/default/deployments/hello-app-deploy
  uid: 6ae40404-a4d1-11e8-851a-0800274edd73
spec:
  progressDeadlineSeconds: 600
  replicas: 3
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: hello-app
      tier: frontend
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: hello-app
        tier: frontend
    spec:
      containers:
      - image: gcr.io/google-samples/hello-app:1.0
        imagePullPolicy: IfNotPresent
        name: hello-app
        ports:
        - containerPort: 8080
          protocol: TCP
        resources: {}
        terminationMessagePath: /dev/termination-log
```

```
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30


$ kubectl edit deploy hello-app-deploy
deployment.extensions/hello-app-deploy edited

# 텍스트 편집기에서 image 버전을 1.0 에서 2.0 으로 수정 후 저장
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: hello-app
        tier: frontend
    spec:
      containers:
      - image: gcr.io/google-samples/hello-app:2.0

# kubectl 기본 에디터는 Vim 으로 혹시 다른 에디터를 이용하시면 아래와 같이 변경이 가능합니다.
$ KUBE_EDITOR="nano"

# image 수정 후 바로 새로운 버전의 pod 가 배포됩니다. 이때 기본 배포전략이 RollingUpdate 이므로 하나씩
새로운 버전의 Pod 가 배포됨을 확인할 수 있습니다. (STATUS 부분을 확인해 주십시오.)

$ kubectl get pod
NAME                                READY     STATUS              RESTARTS   AGE
hello-app-deploy-5f84d59749-sfmvb   0/1       ContainerCreating   0          0s
hello-app-deploy-5f84d59749-tlnnt   1/1       Running             0          5s
hello-app-deploy-77d8794f48-8svgq   1/1       Terminating         0          6m
hello-app-deploy-77d8794f48-9rbzw   1/1       Running             0          6m
hello-app-deploy-77d8794f48-nmvv5   1/1       Running             0          6m

$ kubectl get pod
NAME                                READY     STATUS              RESTARTS   AGE
hello-app-deploy-5f84d59749-frzf9   0/1       ContainerCreating   0          1s
hello-app-deploy-5f84d59749-sfmvb   1/1       Running             0          3s
hello-app-deploy-5f84d59749-tlnnt   1/1       Running             0          8s
hello-app-deploy-77d8794f48-8svgq   0/1       Terminating         0          6m
hello-app-deploy-77d8794f48-9rbzw   1/1       Terminating         0          6m
hello-app-deploy-77d8794f48-nmvv5   1/1       Running             0          6m

$ kubectl get pod
NAME                                READY     STATUS        RESTARTS   AGE
hello-app-deploy-5f84d59749-frzf9   1/1       Running       0          6s
hello-app-deploy-5f84d59749-sfmvb   1/1       Running       0          8s
hello-app-deploy-5f84d59749-tlnnt   1/1       Running       0          13s
hello-app-deploy-77d8794f48-8svgq   0/1       Terminating   0          6m

$ kubectl get pod
```

```
NAME                                 READY     STATUS    RESTARTS   AGE
hello-app-deploy-5f84d59749-frzf9    1/1       Running   0          1m
hello-app-deploy-5f84d59749-sfmvb    1/1       Running   0          1m
hello-app-deploy-5f84d59749-tlnnt    1/1       Running   0          1m

# ReplicaSet 을 보면 새로운 버전 배포를 위해 또다른 ReplicaSet 이 만들어 졌음을 알 수 있습니다.
$ kubectl get rs
NAME                           DESIRED   CURRENT   READY     AGE
hello-app-deploy-5f84d59749    3         3         3         1m
hello-app-deploy-77d8794f48    0         0         0         32m
```

Rollback

```
# 만약 애플리케이션이 Deployment 로 배포되었다면, 새로운 버전에서 문제 발생시 이전 버전으로 쉽게
Rollback 할 수 있습니다.

# 먼저 지금까지 배포된 이력을 아래와 같이 확인하실 수 있습니다.

$ kubectl rollout history  deploy/hello-app-deploy
deployments "hello-app-deploy"
REVISION  CHANGE-CAUSE
1         <none>
2         <none>

$ kubectl rollout undo deploy/hello-app-deploy
deployment.extensions/hello-app-deploy

$ kubectl rollout status deploy/hello-app-deploy
deployment "hello-app-deploy" successfully rolled out

$ kubectl rollout history  deploy/hello-app-deploy
deployments "hello-app-deploy"
REVISION  CHANGE-CAUSE
2         <none>
3         <none>

# 롤백된 애플리케이션 내용 확인을 위해 현재 Service Cluster-IP 확인
$ kubectl get svc
NAME            TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)   AGE
hello-app-svc   ClusterIP   10.99.231.19   <none>        80/TCP    8m
kubernetes      ClusterIP   10.96.0.1      <none>        443/TCP   24m

# Curl 명령어를 통해 v1 으로 롤백되었음을 확인합니다.
$ kubectl run busyboxplus --image=radial/busyboxplus:curl -i --tty --rm
If you don't see a command prompt, try pressing enter.
[ root@busyboxplus-5697648fcc-h54fv:/ ]$ curl 10.99.231.19
Hello, world!
Version: 1.0.0
Hostname: hello-app-deploy-77d8794f48-jl6n6
```

References

- https://kubernetes.io/docs/concepts/workloads/controllers/deployment/