Kubernetes Lab

Last Edited	@June 28, 2023 10:08 PM
Tags	
study	

I found a quick course pretty interesting, named “Fault-Tolerant Web Hosting on Kubernetes”. The course introduces pod expansion and ingress controllers in Kubernetes. The expans

I spent one hour finishing the course, but I feel I am missing something. Scaling up and down web services is another important feature of a large-scale deployment. When requests for

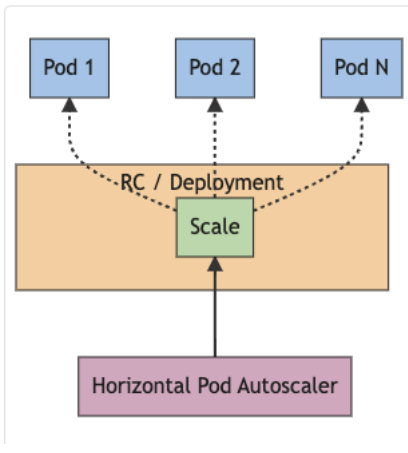
Horizonal Pod AutoScaling

Full document: **Horizontal Pod Autoscaling** https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/

Horizontal Pod Autoscaling means loading more pods or fewer pods,

whereas vertical pod scaling means updating more resources in a pod.

“The HorizontalPodAutoscaler is implemented as a Kubernetes API resource and a controller. The resource determines the behavior of the controller. The horizontal pod autoscaling cc



Kubernetes Horizontal Autoscaler

Full tutorial: **HorizontalPodAutoscaler Walkthrough** https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale-walkthrough/

Here are a few steps I recorded.

1. System version check. I used one project space in educative.io

   The system version is Ubuntu 20.04.1

```
root@7c34f8cd8db31e7d:/usercode# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.1 LTS
Release:        20.04
Codename:       focal
```

2. create a multi-node cluster & metrics server

```
# three node (two workers) cluster config
kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
- role: control-plane
- role: worker
- role: worker
```

```
kind create cluster --name multi-node --config=multi-node.yaml
kubectl get nodes
```

```
root@7c34f8cd8db31e7d:/usercode# kubectl get nodes
NAME                       STATUS   ROLES                  AGE     VERSION
multi-node-control-plane   Ready    control-plane,master   3m56s   v1.21.1
multi-node-worker          Ready    <none>                 3m29s   v1.21.1
multi-node-worker2         Ready    <none>                 3m30s   v1.21.1
```

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
```

3. Create pods and autoscaler for the application

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: php-apache
spec:
  selector:
    matchLabels:
      run: php-apache
  template:
    metadata:
      labels:
        run: php-apache
    spec:
      containers:
      - name: php-apache
        image: registry.k8s.io/hpa-example
        ports:
        - containerPort: 80
        resources:
          limits:
            cpu: 500m
          requests:
            cpu: 200m
---
apiVersion: v1
kind: Service
metadata:
  name: php-apache
  labels:
    run: php-apache
spec:
  ports:
  - port: 80
  selector:
    run: php-apache
```

```
kubectl apply -f php-apache.yaml
kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
# Increase load
kubectl run -i --tty load-generator --rm --image=busybox:1.28 --restart=Never -- /bin/sh -c "while sleep 0.01; do wget -q -O- http://php-apache; done"
kubectl get hpa --watch
```

```
^Croot@ef1335486a4a5560:/# kubectl get hpa --watch
NAME         REFERENCE                     TARGETS         MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache         <unknown>/50%   1         10        0          17s
php-apache   Deployment/php-apache         <unknown>/50%   1         10        1          17s
php-apache   Deployment/php-apache         <unknown>/50%   1         10        1          47s
php-apache   Deployment/php-apache         0%/50%          1         10        1          63s
php-apache   Deployment/php-apache         11%/50%         1         10        1          78s
php-apache   Deployment/php-apache         249%/50%        1         10        1          93s
php-apache   Deployment/php-apache         247%/50%        1         10        4          108s
php-apache   Deployment/php-apache         81%/50%         1         10        5          2m3s
php-apache   Deployment/php-apache         61%/50%         1         10        7          2m18s
php-apache   Deployment/php-apache         69%/50%         1         10        7          2m33s
php-apache   Deployment/php-apache         62%/50%         1         10        7          2m48s
php-apache   Deployment/php-apache         74%/50%         1         10        7          3m3s
php-apache   Deployment/php-apache         78%/50%         1         10        7          3m18s
php-apache   Deployment/php-apache         64%/50%         1         10        7          3m33s
root@ef1335486a4a5560:/#
```

Notice the pods increase from 0 to 7 as the load increases.

More Tutorials:

NGINX Tutorial: Reduce Kubernetes Latency with Autoscaling https://www.nginx.com/blog/microservices-march-reduce-kubernetes-latency-with-autoscaling/

https://itnext.io/autoscaling-ingress-controllers-in-kubernetes-c64b47088485