Kubernetes (k8s)

Gene Kuo

Topics

- Basic concepts
- Kubernetes API objects
- minikube
- `kubectl`
- Deployment and service
- Managing minikube

Basic concepts

- Desired vs actual states
- Deployments: zero-downtime rolling upgrade (green/blue and canary deployment)
- Pods scheduling and liveness probe
- Manual and autoscaling of containers
- Specify quotas for resources that containers need
- Isolation on a namespace level

Basic concepts

- Service discovery and load balancing of running pods
- Readiness probe
- Networking with network plugins comply with Container Network Interface (CNI) specification
- Security with Role-Based Access Control (RBAC)

Kubernetes API objects

- Node
- Pod
- Deployment
- ReplicaSet
- Service
    - IP address and DNS name in the internal network of the cluster
    - Round-robin load balancing
    - Cluster IP or, dedicated port on each node, through external load balancer to expose public IP and DNS name

Kubernetes API objects

- Ingress and ingress controller: manage external access to services in a cluster
    - Ingress in front of services
- Namespace
- ConfigMap
    - Mapped into environment variables or files
- Secret
- DaemonSet: ensures that one pod is running on each node in the cluster

Kubernetes runtime components

- Master nodes (control plane)
  - api-server
  - etcd
  - Controller manager and controllers
  - Scheduler
  - Kubernetes DNS

Kubernetes runtime components

- Worker nodes (data plane)
  - kubelet
  - kube-proxy (DaemonSet)
  - Container runtime

Minikube

- Profiles: default is minikube

```
minikube profile my-profile
minikube config get profile
```

`kubectl`

- Imperative commands: create, delete
- Declarative commands: apply
- Retrieving commands: get, describe, logs

kubectl contexts

- A context: cluster, authentication info for a user, default namespace
    - ~/.kube/config or change KUBECONFIG environment variable

kubectl contexts

```
kubectl config get-contexts
kubectl config use-context my-cluster
kubectl config set-context $(kubectl config current-context) --namespace my-namespace
```

Kubernetes cluster

- Unset KUBECONFIG
- Specify minikube profile
- `minikube start`
- Use add-on manager

Sample code

```
unset KUBECONFIG
minikube profile my-profile
minikube start \
  --memory=10240 \
  --cpus=4 \
  --disk-size=30g \
  --kubernetes-version=v1.15.0 \
  --vm-driver=virtualbox
minikube addon enable ingress
minikube addon enable metrics-server
```

Sample code

```
kubectl get nodes
kubectl get pods --namespace=kube-system
```

`ngix-delpoyment.yaml`

- `kind` and `apiVersion`
- `metadata`
- `spec`: the desired state
  - `replicas`
  - `selector`: find the pods it manages according to labels
  - `template`: how pods should be created
    - `metedata`: labels should match `selector`
    - `spec`: details for the creation of a single container in the pod

Deployment

- Can change/apply replicas number and look at `kubectl get all`

Deployment

```
kubectl create namespace my-deploy
kubectl config set-context $(kubectl config current-context) --namespace=my-deploy
kubectl apply -f ngix-delpoyment.yaml
kubectl get all
kubectl delete pod --selector app=nginx-app
kubectl get all
```

`nginx-service.yaml`

- `spec`: the desired state
  - `type: NodePort`: external accessible port on each node
  - `selector`: find available pods
  - `ports`
    - `port`: internally accessible port
    - `nodePort`: externally accessible port on any of the nodes. 30000-32767
    - `targetPort`: the port in the pod for the requests

Sample code

```
kubectl apply -f nginx-service.yaml
kubectl get svc
minukube ip
http://node_ip:30080
kubectl run -i --rm --restart=Never curl-
client \
  --image=tutum/curl:alpine --command --
curl -s 'http://ngix-service:80'
kubectl delete namespace my-deploy
```

Manage minikube

```
minikube stop
minikube start
kubectl config set-context $(kubectl config current-context) --namespace=my-deploy
minikube delete --profile my-profile
rm -r ~/.minikube/profiles/my-profile
kubectl config delete-context my-deploy
```