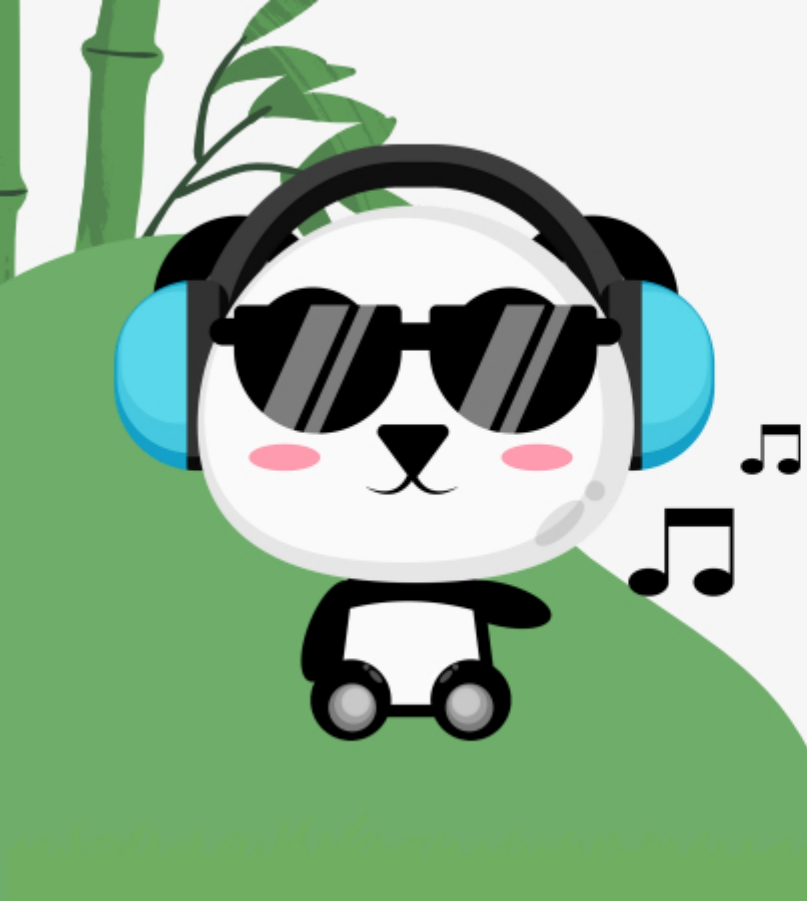


K3

Lesson 9 & 10

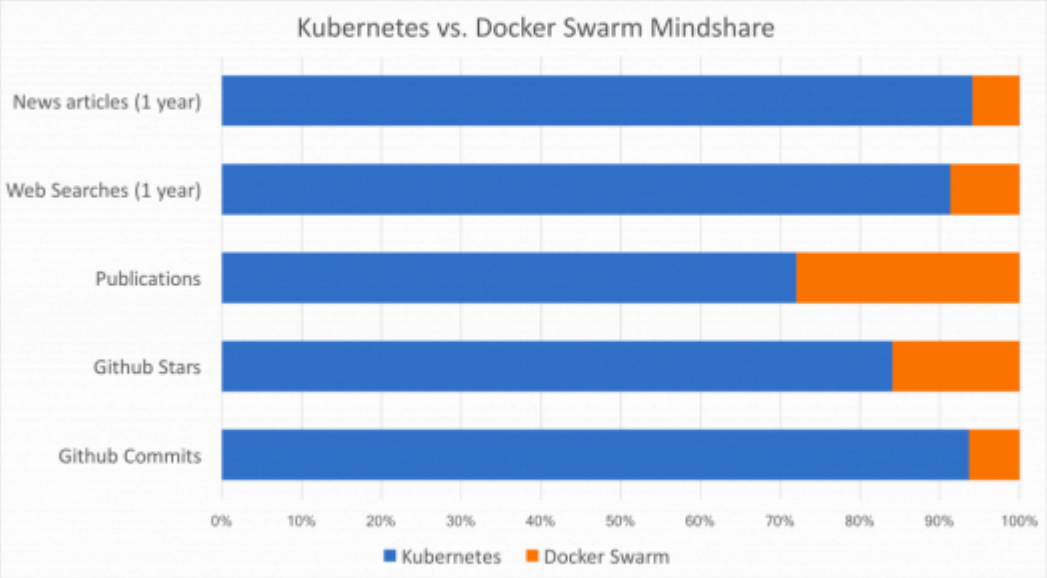
k8s basics

A high-performance tool to serve your models

Learn About

- The architecture of a k8s cluster
- K8s basic commands
- Cost optimization
- Helm

K8s vs Docker Swarm (1)

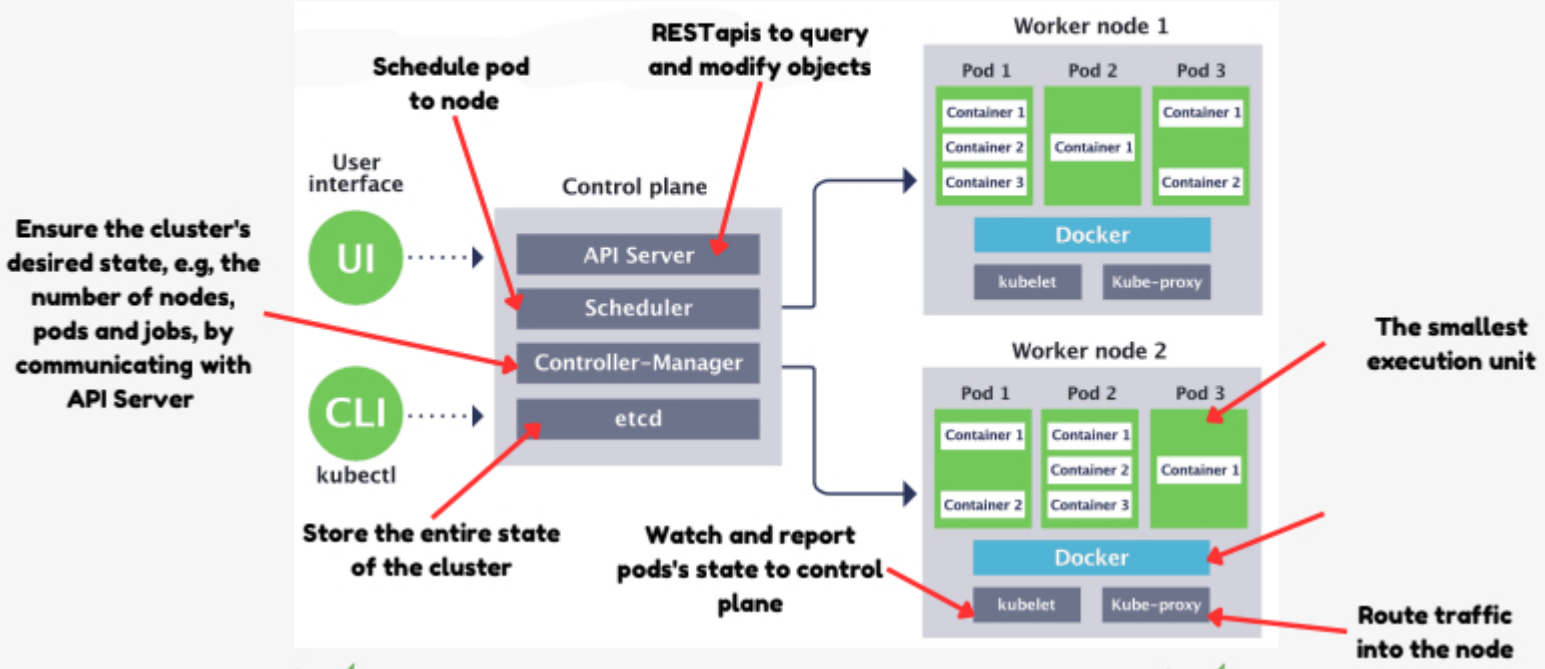


Source: https://platform9.com/blog/kubernetes-docker-swarm-compared/

K8s vs Docker Swarm (2)

	Kubernetes (k8s)	Docker Swarm
Installation Complexity	High	Low
Node Auto-scaling	Yes, and many advanced autoscaling tools are equipped out of the box	Not directly supported
Learning curve	High	Low
CLI	Another CLI (kubectl)	Docker CLI
Quick container deployment	Slower	Faster

K8s cluster's architecture



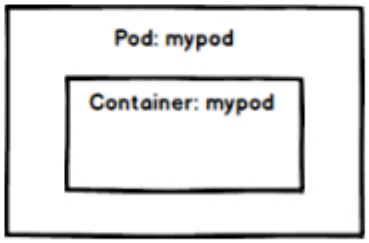
Pod

my_pod_def.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
  - image: nginx
    name: mypod
    resources:
      requests:
        cpu: "0.5"
        memory: "512m"
      limits:
        cpu: "1"
        memory: "1024m"
```

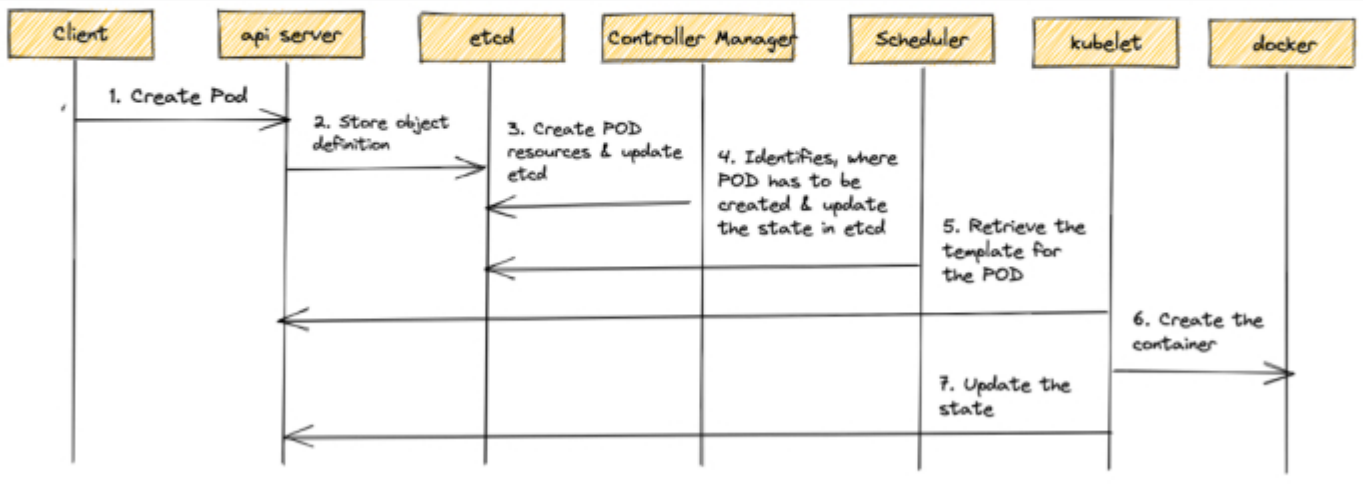
The API used to create a Pod object

Only one container in this pod, and this container is created from the image nginx:latest

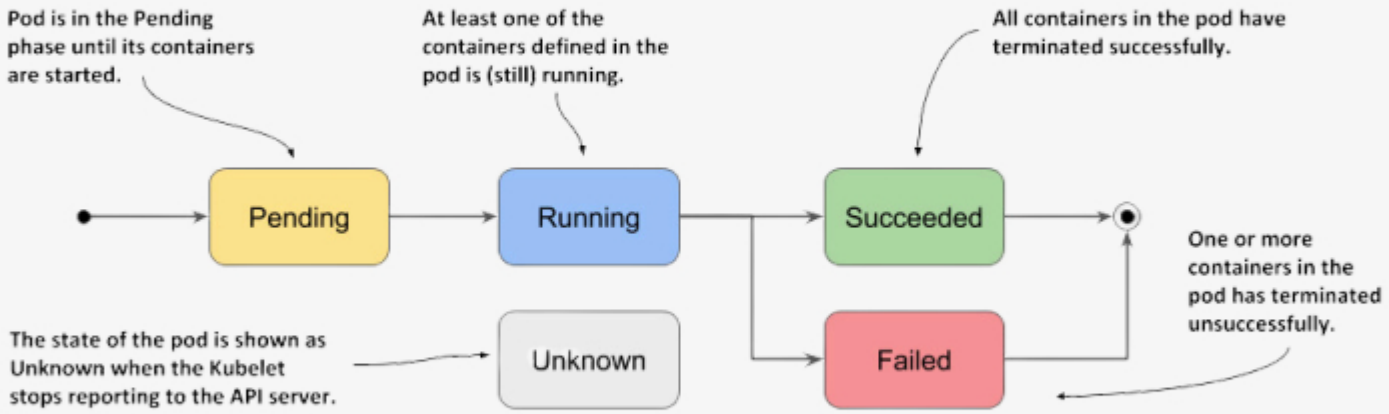


How a pod is created?

k apply -f my_pod_def.yaml -n my_namespace

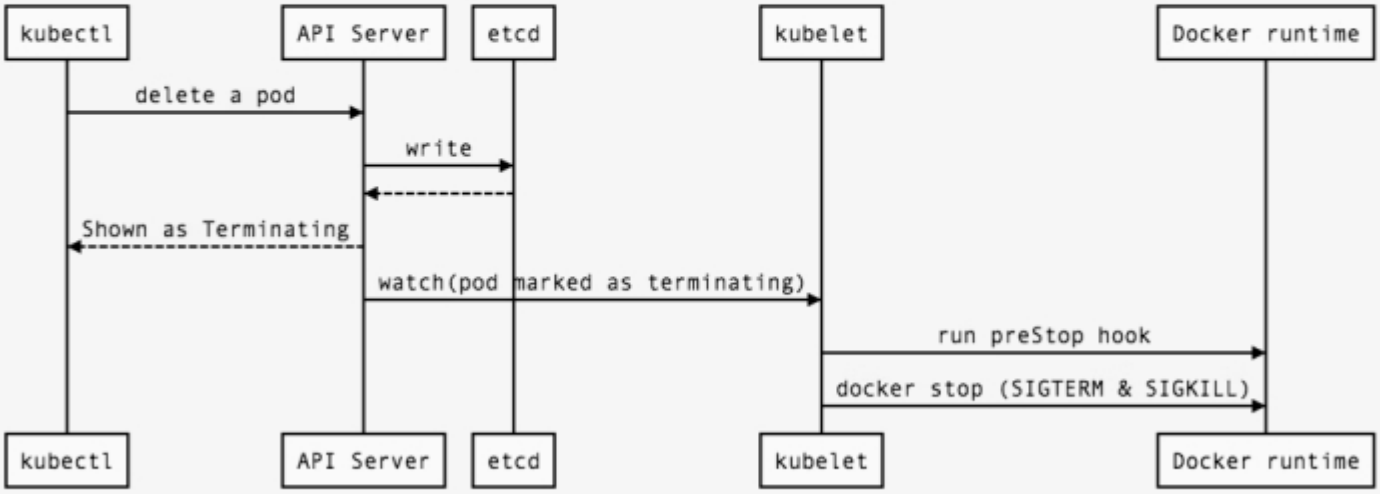


Pod's lifecycle



How a pod is deleted?

k delete -f my_pod_name -n my_namespace



Deployment

We don't often deploy a pod directly, but via a deployment

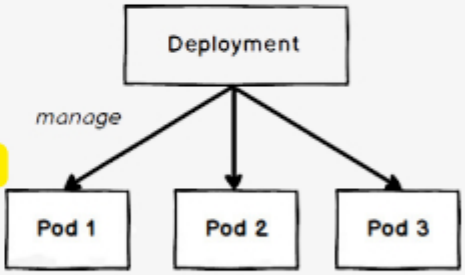
k apply -f my_deployment_def.yaml -n my_namespace

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: my-deploy
  name: my-deploy
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-deploy
  template:
    metadata:
      labels:
        app: my-deploy
    spec:
      containers:
      - image: nginx
        name: nginx
```

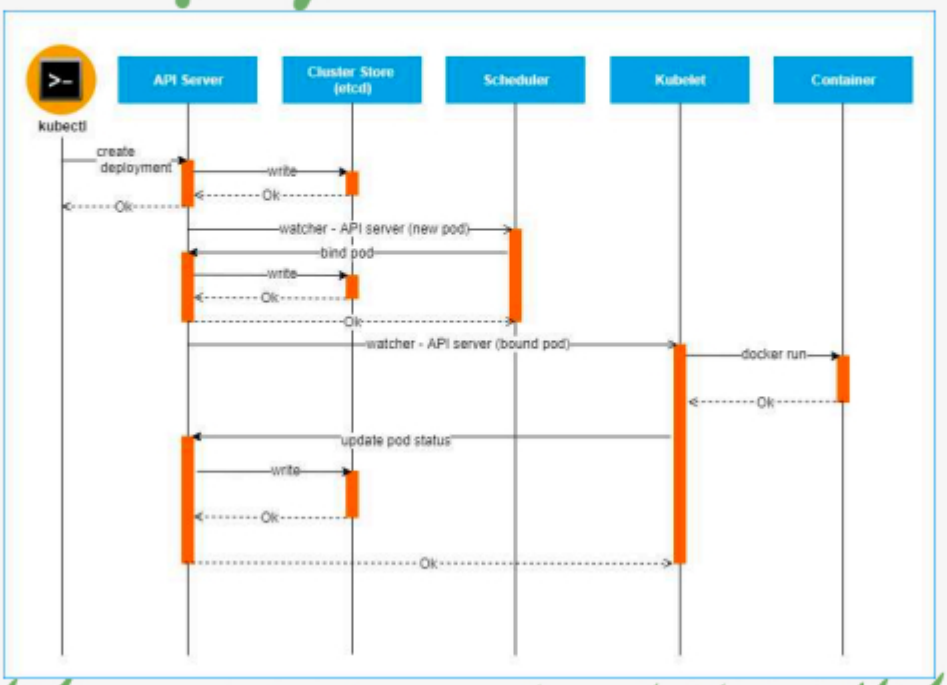
We ask the deployment to maintain 3 pods for our service. Thanks ;)

The pod the deployment will be in charge

Labels for the pods created by the deployment



How a deployment is created?



Service

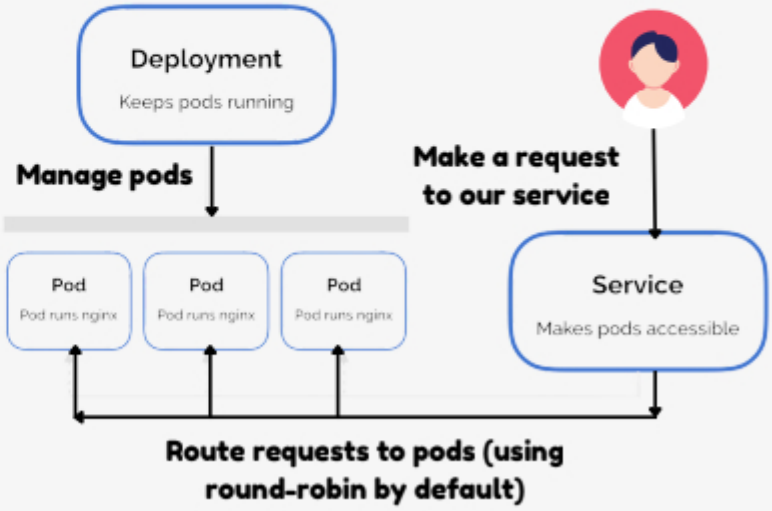
Pods are running, now we need a Service object to help pods visible

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  selector:
    tier: frontend
  ports:
  - port: 3000
    protocol: TCP
    targetPort: 80
  type: ClusterIP
```

Select pod to route by labels

Service's port

Pod's port



Service type

Feature	ClusterIP	NodePort	LoadBalancer
Exposition	Internal cluster	External	External
Accessibility	Default service type and Internal clients send requests to a stable internal IP address.	Through a dedicated port (30000-32767) on all nodes	Through a cloud load balancer IP
User Cases	For internal communication	Best for testing public or private access or providing access for a small amount of time.	Widely used For External communication

Note: Another type is ExternalName for creating a CNAME record, there are several types of DNS records you can read more here

k8s basic objects (1)

Object type	Purpose	Explanation
Pod	The smallest execution unit	
Deployment, ReplicaSet, StatefulSet, DaemonSet	Managing pods	• Deployment creates ReplicaSet under the hood. • DaemonSet is used for distributes pods uniformly across nodes. • StatefulSet gives an identity for each pod so that it can find its proper data store in case of failure
Service	Manage services	

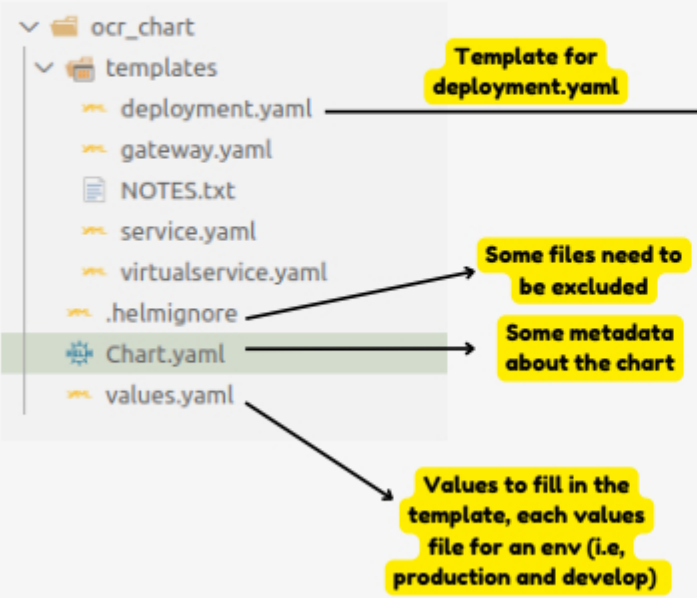
k8s basic objects (2)

Object type	Purpose	Explanation
PersistentVolume (PV), PersistentVolumeClaim (PVC)	Manage disks	A PVC is like a certificate to use PV, so pods are attached with a PVC can access a PV
ConfigMap, Secret	Manage environment variables and secrets	
Namespace	A virtual cluster	Normally, a group of objects having the same functions will be grouped into one namespace
Job, CronJob	Discontinuous running pods	

k8s basic commands

- **Create an object:**
  - **k apply -f my_object_def.yaml -n my_namespace**
- **Read an object:**
  - **k get object_type object_name -o yaml -n my_namespace**
  - **k describe object_type object_name -o yaml -n my_namespace**
- **Update an object:**
  - **k edit object_type object_name -n my_namespace**
- **Delete an object:**
  - **k delete object_type object_name -n my_namespace**
- **Log a pod:**
  - **k logs pod_name -n my_namespace**

Helm: nothing but a template



ocr_chart > templates > deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ .Release.Name }}
  labels:
    app: {{ .Release.Name }}
  namespace: model-serving
spec:
  replicas: 1
  selector:
    matchLabels:
      app: {{ .Release.Name }}
  template:
    metadata:
      labels:
        app: {{ .Release.Name }}
    spec:
      containers:
        - name: {{ .Release.Name }}
          image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"
          imagePullPolicy: {{ .Values.image.pullPolicy }}
          ports:
            - containerPort: 80
```

Helm basic commands

Commands	Explanation
cd ocr_chart && helm upgrade --install release_name .	Install ocr_chart
helm list	List all releases
helm uninstall release_name	Uninstall a release
helm history release_name	List all versions of a release
helm rollback release_name version	Install another version of the release (in the history)

- Another way to install a chart (not recommended):
  - Add the helm repo
    - helm repo add jenkins https://charts.jenkins.io
  - Search for the repo locally
    - helm search repo jenkins
  - Install the repo
    - helm install jenkins jenkins/jenkins

Cost Optimization

- Monitor cost:
  - Using a cloud monitor service, or an open-source tool (e.g, kubecost)
- Right-sizing all your resources
- Buy instances more intelligently:
  - Reserved instances and savings plans
  - Spot instances
- Reduce all:
  - Nodes: pay attention to your replicas
  - Data transfer between different availability zones (AZ) and regions
  - Local storage: clean your trash more often

Thank You!

Created by
Quan Dang