

Kubernetes Training Course 2022

Introduction to Kubernetes I: Basic Concepts

Sebastian Beyvers

`sebastian.beyvers@computational.bio.uni-giessen.de`

Marius Dieckmann

`marius.dieckmann@computational.bio.uni-giessen.de`

06 April 2022



What is Kubernetes (K8s) ?



kubernetes

- Container orchestration framework
- Open sourced by Google in 2014
 - Based on a Google internal framework called Borg
- The leading container orchestration framework
 - Adopted by most cloud provider: GCP (GKE), AWS (EKS), Azure (AKS), ...
- Other frameworks lost momentum
 - Mesos -> mainly for deploying infrastructure
 - Docker swarm -> nearly irrelevant

Kubernetes architecture

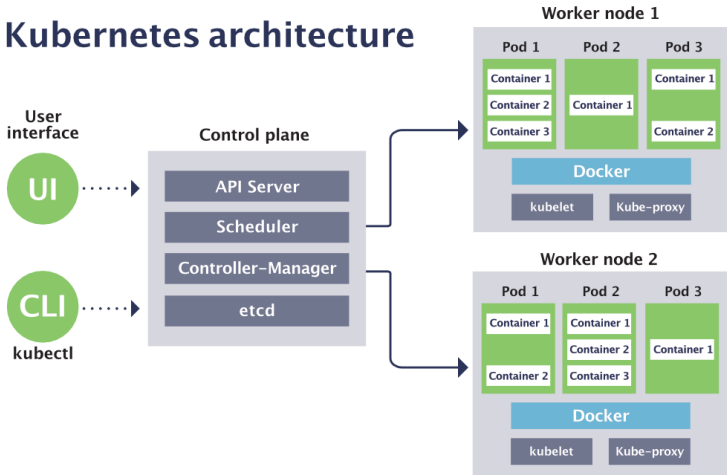


Figure 1: Basic overview of the infrastructure layout of Kubernetes Source: <https://sensu.io/blog/how-kubernetes-works>

Kubernetes - Resources

- Everything in Kubernetes is a resource
- Resources are managed by dedicated controllers
- You can extend K8s with own resources and controllers (CRDs)

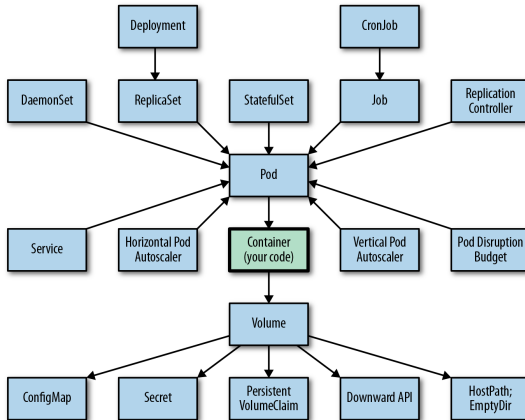


Figure 2: Basic overview of resources that interact with the pod resource:

<https://dev4devs.com/2019/10/20/what-are-the-kubernetes-resources-which-are-most-useful-for-developers/>

Kubernetes - Concepts I

- **Infrastructure**

- Master-Worker (Master – Node) architecture
 - Master runs API, etcd, Scheduler, ...
 - Nodes run kubelet, kube-proxy, ...

- **Deployment units**

- Pod
 - Basic deployment unit
 - Runs 1..n container
- Replica set
 - Creates replicas of Pods
- Deployment
 - Manages replica sets
 - Helps with updating Container images (rolling updates/rollbacks)
- Stateful set
 - Pods that stay on a node and remain "state" -> Databases
- Jobs
 - Pods that run to "completion" -> Workflows
- ...

Kubernetes - Concepts II

- **Networking**

- Flat overlay network
 - By default: Every pod can see every other pod
- Each pod has an individual (non-public) IP

- **High-level organization**

- Service: pods are grouped with a single name -> (internal) DNS
- Ingress
 - Multiple implementation, in our case: nginx ingress

- **Scheduling**

- Fully automated based on resource requests
- Resources:
 - CPU -> 0.5 CPU is a valid resource requests/quota
 - RAM
 - Storage -> Can separate various storage classes
 - GPUs

- RBAC for tenancy and authorization
- Namespaces to support tenancy
- Most resources created by users are bound to a namespace
 - Some (especially internal) resources are not bound to a namespace
- Namespaces do not handle multi-tenancy well
 - Addons and distros (e.g. Rancher) add some capabilities
- Containers (Docker) also have drawbacks for hard multi-tenancy

- **RESTful API**
 - Uses standard HTTP verbs to define an action (POST, DELETE, UPDATE, ...)
- **Verbs applied to resource types (pod, deployments, ...)**
 - Resource types are represented as kind
 - kinds are strictly versioned (semantic versioning)
- **Kubectl -> CLI**
 - Verbs are slightly different (create, get, describe, delete, ...)
 - Similar syntax -> kubectl verb resource [options]
 - Example:
 - kubectl get pods -n namespace
 - Create command often used along with YAML files
 - YAML file often contains the resource configuration

Excursus - YAML

YAML Ain't Markup Language is a human-readable data-serialization language and super-set of JSON.

- Indentation matters (similar to Python)
- You can separate resources with three dashes — — —
- Hierarchical structure, — indicates lists.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: hello
spec:
  template:
    # This is the pod template
    spec:
      containers:
        - name: hello
          image: busybox
          command: ['sh', '-c', 'echo "Hello, Kubernetes!" && sleep 3600']
      restartPolicy: OnFailure
    # The pod template ends here
```

Figure 3: Basic overview of a Pod YAML. Source: <https://kubernetes.io/docs/concepts/workloads/pods/>

Pods

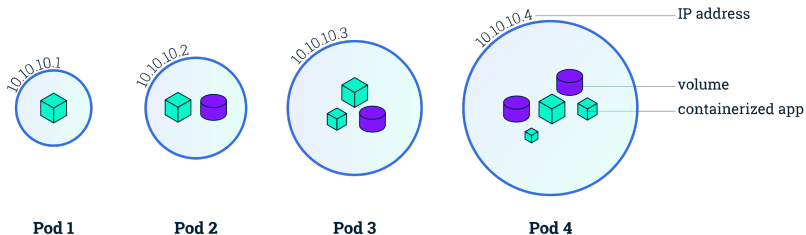


Figure 4: Basic overview of a Job. Source: <https://kubernetes.io/docs/tutorials/kubernetes-basics/explore/explore-intro/>

Additional features for pods:

- Liveness probes
- Init containers
- Have a Lifecycle: Pending, Running, Succeeded, Failed, Unknown

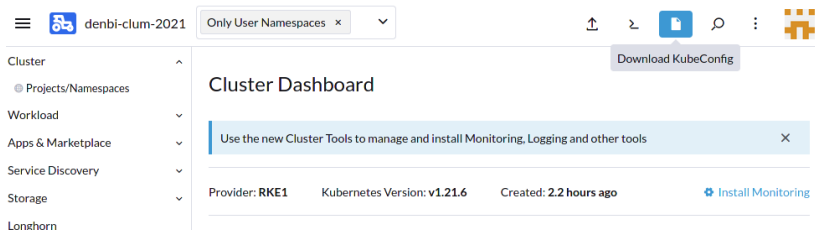
Jobs and Cronjobs

Jobs are pods that run to completion e.g. until the status is either *succeeded* or *failed*. **CronJobs** are jobs that run in a periodic interval.

```
apiVersion: batch/v1
kind: CronJob
metadata:
  name: hello
spec:
  schedule: "*/1 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: hello
              image: busybox
              imagePullPolicy: IfNotPresent
              command:
                - /bin/sh
                - -c
                - date; echo Hello from the Kubernetes cluster
          restartPolicy: OnFailure
```

Excursus - Rancher WebUI

- Rancher is one of many Kubernetes distributions
 - Easier cluster deployment
 - Web GUI for Users and Admins
- For now we use it for getting the kubectl config
- <https://rancher.biokube.org>



Questions?