

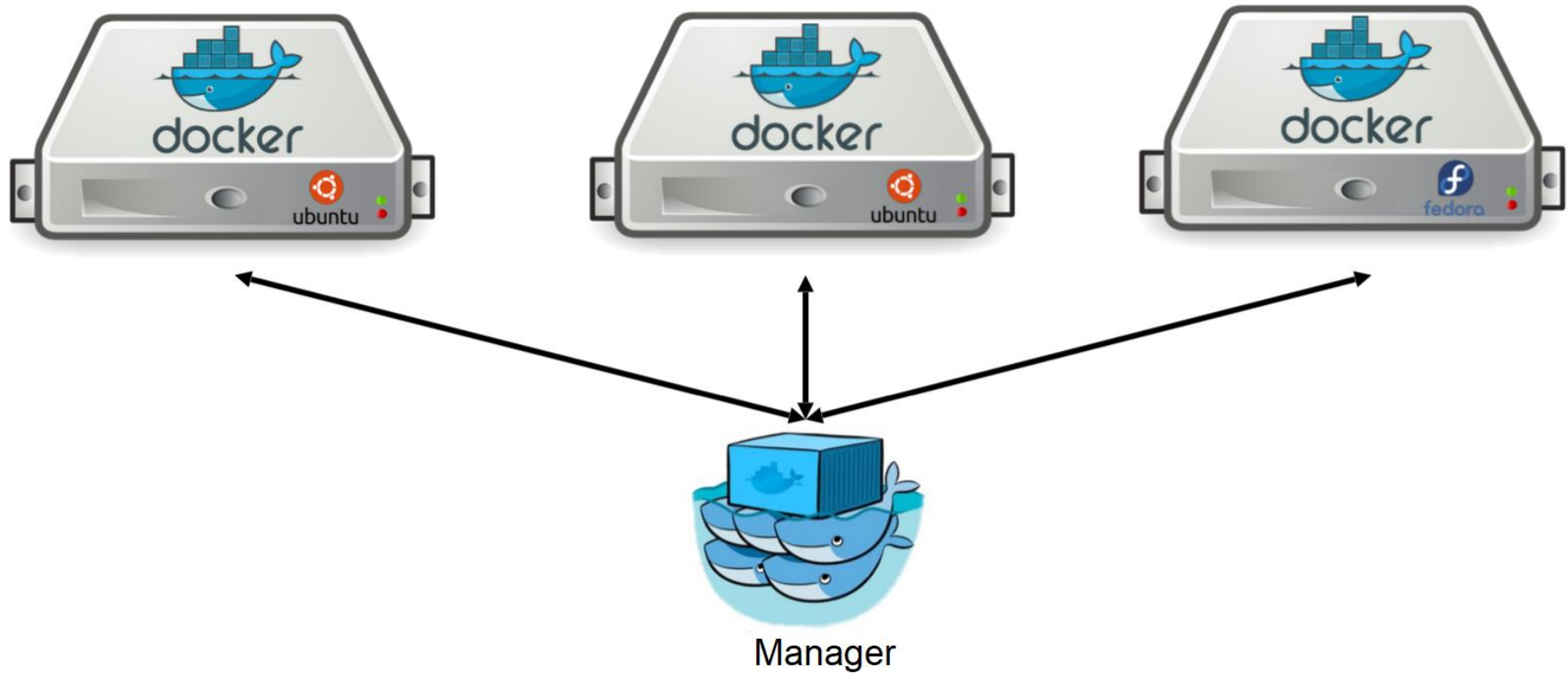
Clustering with (μ)Kubernetes

Narges Mehran, MSc.

Current Topics in Distributed Systems:
Internet of Things and Cloud Computing,

SS2021

Container Orchestration



<https://docs.docker.com/>

Container Orchestration (cont.)

When you have large applications deployed on a lot of containers,

- you need a dedicated monitoring system.

Container orchestration refers to the process of organizing multiple-containerized applications,

- guides container deployment
- automates updates, health monitoring, and failover procedures.

`docker service create --replicas=1000 node`

Platforms such as:

- Apache Mesos,
- Google Kubernetes,
- Docker Swarm(kit),
- Nomad



Nomad

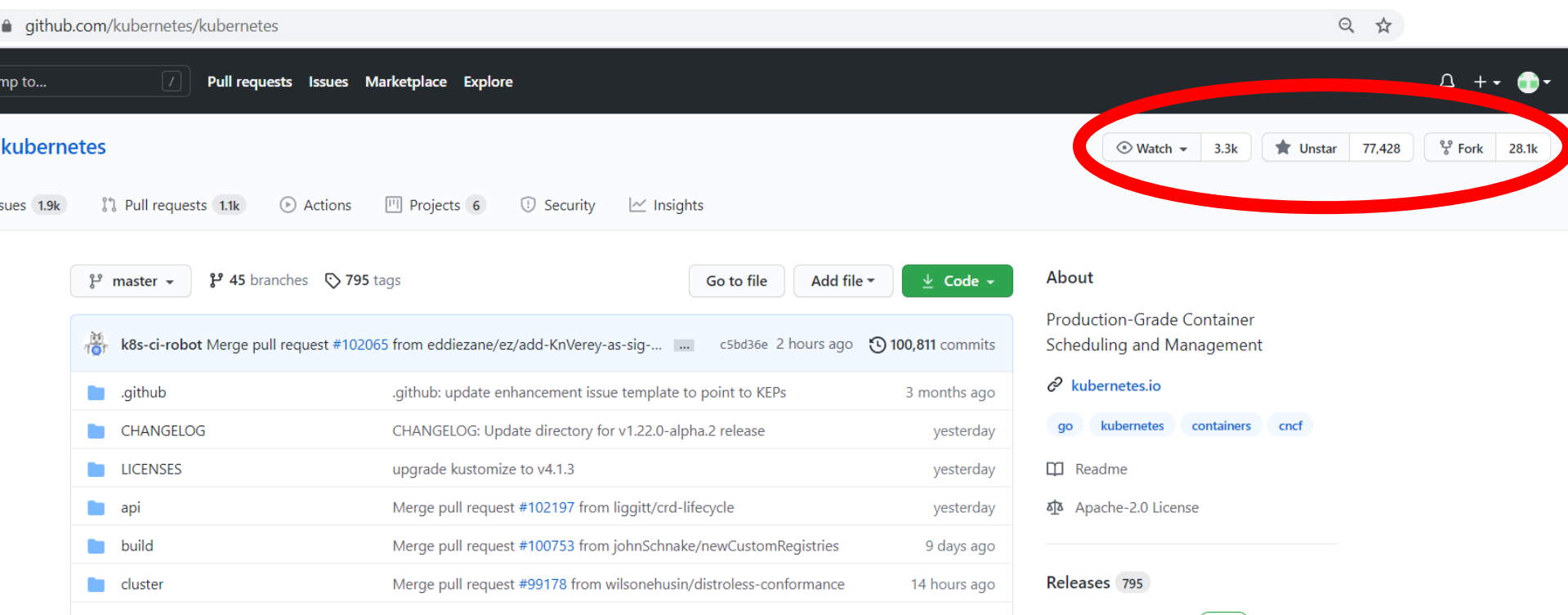
The Two Million Container Challenge

HashiCorp Nomad scheduled 2,000,000 Docker containers on 6,100 hosts in 10 AWS regions in 22 minutes.

Container Orchestration (cont.)

Platforms such as:

- Google Kubernetes → one of the top-ranked projects in GitHub



The screenshot shows the GitHub repository page for `kubernetes/kubernetes`. The URL in the browser is `github.com/kubernetes/kubernetes`. The repository name `kubernetes` is displayed in blue. The interaction bar at the top right, which is circled in red, contains the following buttons: `Watch` (with an eye icon), `3.3k` (star count), `Unstar` (with a star icon), `77,428` (fork count), `Fork` (with a fork icon), and `28.1k` (pull request count). Below the repository name, the main content area shows a list of recent commits. The first commit is by `k8s-ci-robot` merging pull request `#102065` from `eddiezane/ez/add-KnVerey-as-sig-...` 2 hours ago, with 100,811 commits. The commit list includes files like `.github`, `CHANGELOG`, `LICENSES`, `api`, `build`, and `cluster`. On the right side, the `About` section describes the project as "Production-Grade Container Scheduling and Management" and provides links to `kubernetes.io`, `go`, `kubernetes`, `containers`, and `cncf`. The `Releases` section shows 795 releases.

github.com/kubernetes/kubernetes

kubernetes

Watch 3.3k Unstar 77,428 Fork 28.1k

master 45 branches 795 tags

Go to file Add file Code

k8s-ci-robot Merge pull request #102065 from eddiezane/ez/add-KnVerey-as-sig-... c5bd36e 2 hours ago 100,811 commits

| | | |
|-----------|--|--------------|
| .github | .github: update enhancement issue template to point to KEPs | 3 months ago |
| CHANGELOG | CHANGELOG: Update directory for v1.22.0-alpha.2 release | yesterday |
| LICENSES | upgrade kustomize to v4.1.3 | yesterday |
| api | Merge pull request #102197 from liggitt/crd-lifecycle | yesterday |
| build | Merge pull request #100753 from johnSchnake/newCustomRegistries | 9 days ago |
| cluster | Merge pull request #99178 from wilsonchusin/distroless-conformance | 14 hours ago |

About

Production-Grade Container Scheduling and Management

kubernetes.io

go kubernetes containers cncf

Readme

Apache-2.0 License

Releases 795

Kubernetes

Kubernetes is an open-source orchestration system for automating the management, placement, scaling and routing of containers that has become popular with developers and IT operations teams in recent years.

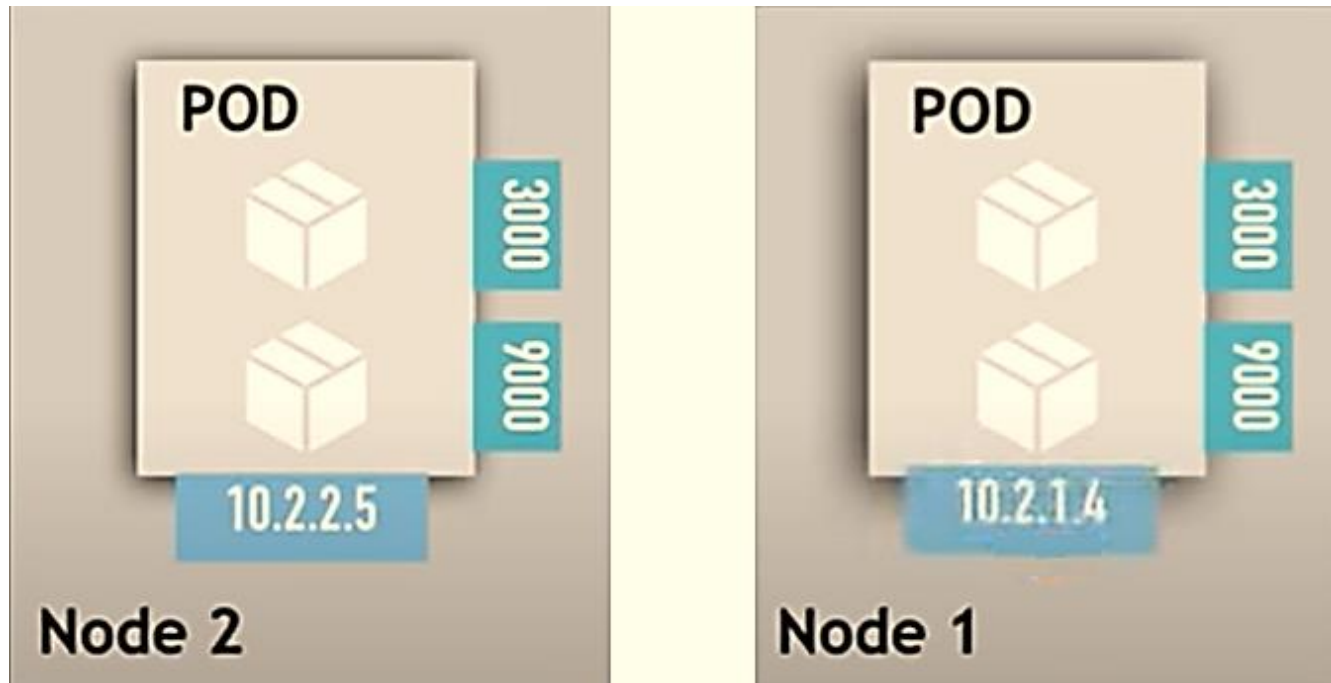
It was first developed by Google and contributed to Open Source in 2014 and is now maintained by the Cloud Native Computing Foundation.

There is an active Kubernetes community and ecosystem developing around Kubernetes with thousands of contributors and dozens of certified partners.

Pod

- Pod is the smallest and simplest Kubernetes object.
- Pod represents a set of running [containers](#) on a cluster.
- Pod is typically set up to run a single primary container.
- Pods are commonly managed by a [Deployment](#).
- The containers belong to the same Pod expose a single private IP address to the rest of Kubernetes system.
- They can communicate with various IP addresses.

Pod (cont.)



Pod (cont.)

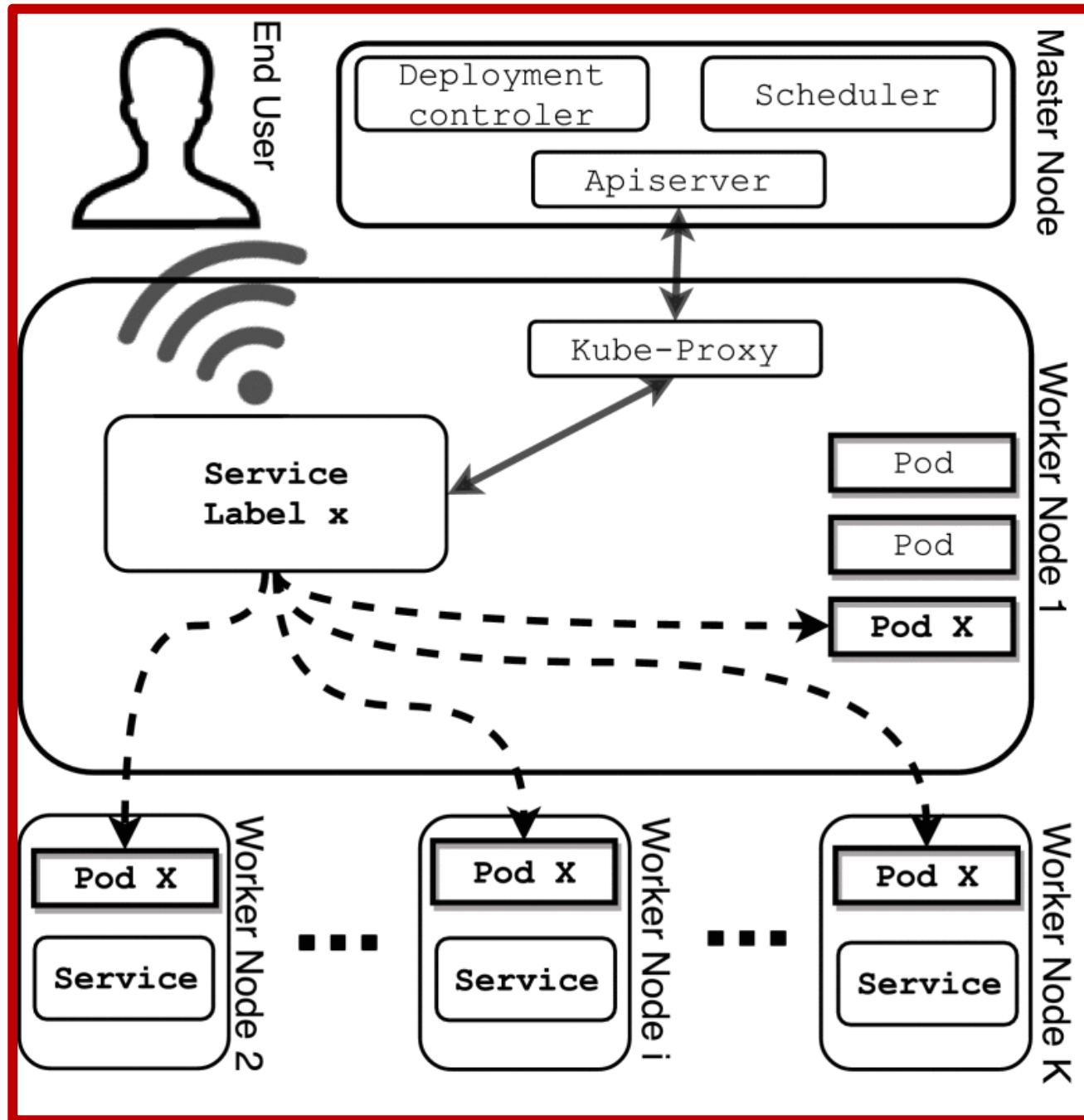
Containers in a pod share a common network interface and each container has access to all storage volumes assigned to the pod.

Pods are not usually managed directly by developers.

Application developers are expected to create a Deployment Controller which oversees the creation and management of a set of identical pods providing the expected functionality.

It can dynamically add, and remove pods to/from the set, for example to adjust the processing capacity according to workload variations or to deal with user's mobility.

Organization of a Kubernetes service.



Fahs, Ali, and Guillaume Pierre.
"Proximity-aware traffic routing
in distributed fog computing
platforms." CCGrid 2019.

Kubernetes command-line tool

The Kubernetes command-line tool for controlling Kubernetes clusters, [kubectl](#)

You can use this to create and inspect objects in your Kubernetes cluster.

<https://kubernetes.io/docs/reference/kubectl/overview/>

Some features of *kubectl*

With Docker, you were able to run a single instance of an app, but with Kubernetes you can run a large number or even scale it up:

- `kubectl run --replica=1000 my-web-server`
- `kubectl scale --replica=2000 my-web-server`

It can be done automatically based on the users' load:

- `kubectl rolling-update my-web-server --image=web-server:2`
- `kubectl rolling-update my-web-server --rollback`

Kubernetes Control Plane

The Kubernetes control plane consists of a collection of processes running on a cluster:

The **Kubernetes Master** is a collection of three processes that run on a single node in a cluster, which is designated as the master node,

Master node

Those processes are:

- [kube-apiserver](#)
 - The Kubernetes API server validates and configures data for the API objects which include pods, services, replication-controllers, and others. The API server provides the frontend to the cluster's shared state through which all other components interact.
- [kube-controller-manager](#)
 - Control plane component that runs [controller](#) processes.
- [kube-scheduler](#)
 - It distributes workloads across multiple nodes.

Non-master node

A non-master node in a cluster executes two processes:

- [kubelet](#): which communicates with the Kubernetes Master.
- [kube-proxy](#): which is a network proxy and runs Kubernetes networking services on each node.

Getting started with Kubernetes

There are different options to set up, provision and run Kubernetes.

You can deploy a Kubernetes cluster on

- a local machine,
- Cloud and Fog computing environment,
- on-prem datacenter,
- or choose a managed Kubernetes cluster.

You can also create custom solutions across a wide range of cloud providers, or bare metal environments.

Interactive shell provided by K8s

<https://kubernetes.io/docs/tutorials/kubernetes-basics/explore/explore-intro/>

Build Kubernetes-ready applications on your desktop

[Play with Kubernetes for free](#)

Minikube

A tool for running Kubernetes locally.

Minikube runs a single-node cluster inside a VM on your computer:

- `curl -LO`
`https://storage.googleapis.com/minikube/releases/latest/minikube_latest_amd64.deb`
- `sudo dpkg -i minikube_latest_amd64.deb`
- `minikube start`

MicroK8s

MicroK8s is

- a small, fast, single-package Kubernetes for developers, IoT and edge,

in other words,

- it is a minimal, lightweight Kubernetes you can run and use on practically any machine.
- it works on [42 flavors of Linux](#).

<https://microk8s.io/docs>

```
$ sudo snap install microk8s --channel=1.18 --classic
```

MicroK8s (cont.)

MicroK8s only installs the basics of a usable Kubernetes features:

- api-server
- controller-manager
- scheduler
- Kubelet
- kube-proxy

Adding a node

To create a cluster out of two or more already-running MicroK8s instances, use:

- `$ microk8s add-node`

If you run this command on a MicroK8s instance, it will be the master of the cluster and will host the Kubernetes control plane.

add-node command prints a *microk8s join* command which should be executed on the MicroK8s instance that you wish to join to the cluster:

- `$ microk8s join ip-172-31-20-243:25000/ArvwtYMnHJBQkJrFzFsSmSPNDTEojrIS`

Several commands of (μ)Kubernetes

\$ microk8s.

- `kubectl get services` # List all services in the namespace
- `kubectl get pods -o wide` # List all pods in the current namespace, with more details
- `kubectl get deployment my-dep` # List a particular deployment
- `kubectl get pods` # List all pods in the namespace
- `kubectl get pod my-pod -o yaml` # Get a pod's YAML

➤ \$ microk8s kubectl get pods

| NAME | READY | STATUS | RESTARTS | AGE |
|-------|-------|---------|----------|-----|
| nginx | 1/1 | Running | 0 | 11m |

Namespaces in Kubernetes

- Namespaces are intended for use in environments with many users spread across multiple teams, or projects.
- Namespaces are a way to divide cluster resources between multiple users (via resource quota).

✓ <https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/>

\$ microk8s.

- | | |
|-------------------------------------|--|
| ➤ kubectl get namespace | # List the current namespaces in a cluster |
| ➤ kubectl get all --all-namespaces | # List all pods, services, deployments |
| ➤ kubectl get pods --all-namespaces | # List all pods in all namespaces with more details |

Several commands of Kubernetes

```
File Edit View Search Terminal Help
edgegateway@gateway:~$ sudo kubectl get nodes
NAME                STATUS              AGE    VERSION
g-termln            Ready               112d   v1.20.4
node1               Ready               48d    v1.21.0
node10              NotReady            <none>  48d    v1.21.0
node13              Ready               48d    v1.21.0
node14              Ready               48d    v1.21.0
node15              Ready               48d    v1.21.0
node16              Ready               48d    v1.21.0
node17              Ready               48d    v1.21.0
node18              Ready               48d    v1.21.0
node19              Ready               48d    v1.21.0
node2               Ready               61d    v1.21.0
node20              Ready               48d    v1.21.0
node21              Ready               48d    v1.21.0
node22              Ready               48d    v1.21.0
node23              Ready               48d    v1.21.0
node24              Ready               48d    v1.21.0
node4               Ready               61d    v1.21.0
node5               Ready               62d    v1.21.0
node6               Ready               63d    v1.21.0
node7               Ready               48d    v1.21.0
node8               Ready               48d    v1.21.0
node9               Ready               48d    v1.21.0

edgegateway@gateway:~$ sudo kubectl get pods
NAME                READY   STATUS              RESTARTS   AGE
encoding            0/1     CrashLoopBackOff    1361       4d20h
node                0/1     ImagePullBackOff    506        4d21h
pingtest-64f9cb6b84-2vxs4  1/1     Running              3          47d
pingtest-64f9cb6b84-4fcng  1/1     Terminating        1          47d
pingtest-64f9cb6b84-cxg4h  1/1     Running              0          17d
pingtest-64f9cb6b84-dzt7n  1/1     Running              4          47d
pingtest-64f9cb6b84-lklbp  1/1     Running              1          46d
pingtest-64f9cb6b84-sctrs  1/1     Running              3          47d
test-kl-kube-latency-7m885  1/1     Running              265        46d
test-kl-kube-latency-9vnmz  1/1     Running              1          46d
test-kl-kube-latency-b6j6s  1/1     Running              3          46d
test-kl-kube-latency-bcxvc  1/1     Running              2          46d
test-kl-kube-latency-bf5qq  1/1     Running              280        48d
test-kl-kube-latency-csvd7  1/1     Running              1          46d
test-kl-kube-latency-dztsc  1/1     Running              1350       46d
test-kl-kube-latency-fgrbw  0/1     CrashLoopBackOff    1851       46d
test-kl-kube-latency-g56h2  1/1     Running              3          48d
test-kl-kube-latency-gg6j2  1/1     Running              2          46d
test-kl-kube-latency-kd9lf  1/1     Running              0          46d
test-kl-kube-latency-q87m7  1/1     Running              2          46d
test-kl-kube-latency-qlfjf  1/1     Running              1          46d
test-kl-kube-latency-rt5jf  0/1     CrashLoopBackOff    1693       46d
test-kl-kube-latency-t95xp  1/1     Running              1          46d
test-kl-kube-latency-t98gd  0/1     CrashLoopBackOff    1521       48d
test-kl-kube-latency-w6fzj  1/1     Running              1          46d
test-kl-kube-latency-wwk72  1/1     Running              3          48d
test-kl-kube-latency-xmwfr  1/1     Running              2          46d
test-kl-kube-latency-znlms  1/1     Running              2          46d
test-kl-kube-latency-zssnf  1/1     Running              2          46d

edgegateway@gateway:~$ sudo kubectl get deployments
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
pingtest            5/5     5             5           47d

edgegateway@gateway:~$ sudo kubectl get services
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP    PORT(S)                AGE
kube-latency        ClusterIP   10.111.203.30    <none>          8080/TCP                48d
kubemq              ClusterIP   10.101.24.31     <none>          8081/TCP,9091/TCP,50000/TCP  7d18h
kubernetes           ClusterIP   10.96.0.1        <none>          443/TCP                112d
redis-master         ClusterIP   10.105.219.182   <none>          6379/TCP                7d17h
redis-slave          ClusterIP   10.103.20.32     <none>          6379/TCP                7d17h
test-kl-kube-latency ClusterIP   None             <none>          8080/TCP                48d

edgegateway@gateway:~$
```

Generally, *sudo* permission is not needed!

Solution:

`sudo -i`

`root@:~# swapoff -a`

`root@:~# exit`

`strace -eopenat kubectl version`

`mkdir -p $HOME/.kube`

`sudo cp -i /etc/kubernetes/admin.conf`

`$HOME/.kube/config`

`kubernetes-master:~$ sudo chown $(id -u):$(id`

`-g) $HOME/.kube/config`

Check the status

MicroK8s has a built-in command to display its status, by which you can check the list of available addons.

During installation, you can use the `--wait-ready` flag to wait for the Kubernetes services to get initialized:

- `$ microk8s status --wait-ready`

Deploy an app

To deploy apps and services, use the `kubectl` command. Try installing a demo app:

- `$ microk8s kubectl create -f deployment.yaml`

Run deployment if already created

- `$ microk8s kubectl apply -f deployment.yaml`

It may take a minute or two to install, but you can check the status:

- `$ microk8s kubectl get pods`

Starting and Stopping MicroK8s

MicroK8s will continue running until you decide to stop it. You can stop and start MicroK8s with these simple commands:

- `$ microk8s stop`

... will stop MicroK8s and its services. You can start it again by running:

- `$ microk8s start`

Bulletin-board

node-bulletin-board project
is a simple bulletin board
application, written in
Node.js

[https://github.com/dockersamples
/node-bulletin-board](https://github.com/dockersamples/node-bulletin-board)



A Kubernetes YAML file of two objects

apiVersion

indicates the Kubernetes API that parses this object

kind

indicates the type of object

metadata

applies some names to your objects

spec

specifies all the parameters and configurations of your object.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: bb-demo
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      bb: web
  template:
    metadata:
      labels:
        bb: web
    spec:
      containers:
        - name: bb-site
          image: bulletinboard:1.0
---
apiVersion: v1
kind: Service
metadata:
  name: bb-entripoint
  namespace: default
spec:
  type: NodePort
  selector:
    bb: web
  ports:
    - port: 8080
      targetPort: 8080
      nodePort: 30001
```

A Kubernetes YAML file of two objects

Deployment

describes a scalable group of identical Pods. Here, Pod (under the template: key) has just one container based on your bulletinboard:1.0 image.

Service: NodePort type

routes traffic from port 30001 on your host to port 8080 of pods, and allows a customer to browse your bulletin board service.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: bb-demo
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      bb: web
  template:
    metadata:
      labels:
        bb: web
    spec:
      containers:
        - name: bb-site
          image: bulletinboard:1.0
---
apiVersion: v1
kind: Service
metadata:
  name: bb-entripoint
  namespace: default
spec:
  type: NodePort
  selector:
    bb: web
  ports:
    - port: 8080
      targetPort: 8080
      nodePort: 30001
```

Deploying the *bb* application

➤ `kubectl apply -f bb.yaml`

deployment.apps/bb-demo created

service/bb-entripoint created

➤ `kubectl get deployments`

| NAME | DESIRED | CURRENT | UP-TO-DATE | AVAILABLE | AGE |
|---------|---------|---------|------------|-----------|-----|
| bb-demo | 1 | 1 | 1 | 1 | 48s |

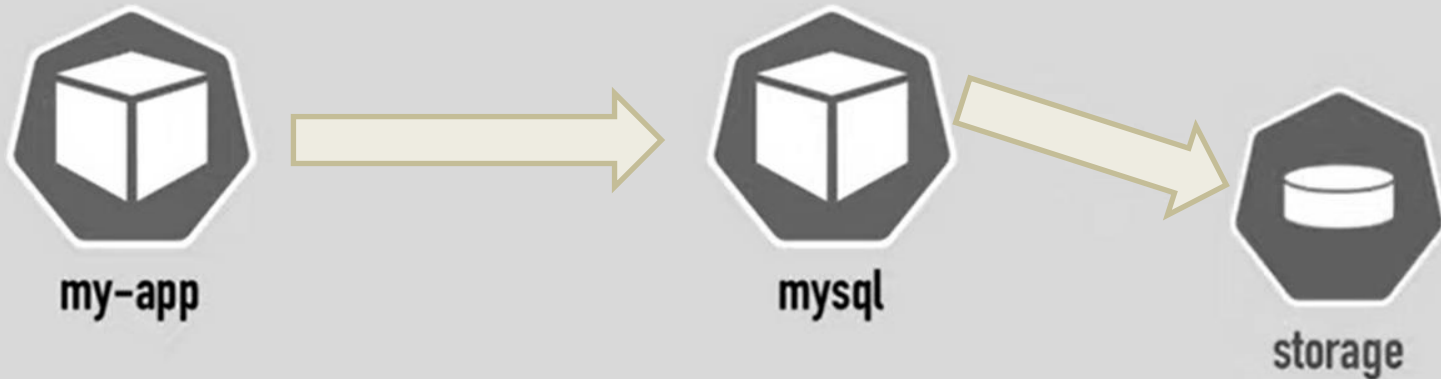
➤ `kubectl get services`

| NAME | TYPE | CLUSTER-IP | EXTERNAL-IP | PORT(S) | AGE |
|---------------|-----------|----------------|-------------|----------------|------|
| bb-entripoint | NodePort | 10.106.145.116 | <none> | 8080:30001/TCP | 53s |
| kubernetes | ClusterIP | 10.96.0.1 | <none> | 443/TCP | 138d |

➤ Open a browser and visit your bulletin board at `localhost:30001`; you should see your bulletin board, the same as when we ran it as a stand-alone container in the previous step of this tutorial.

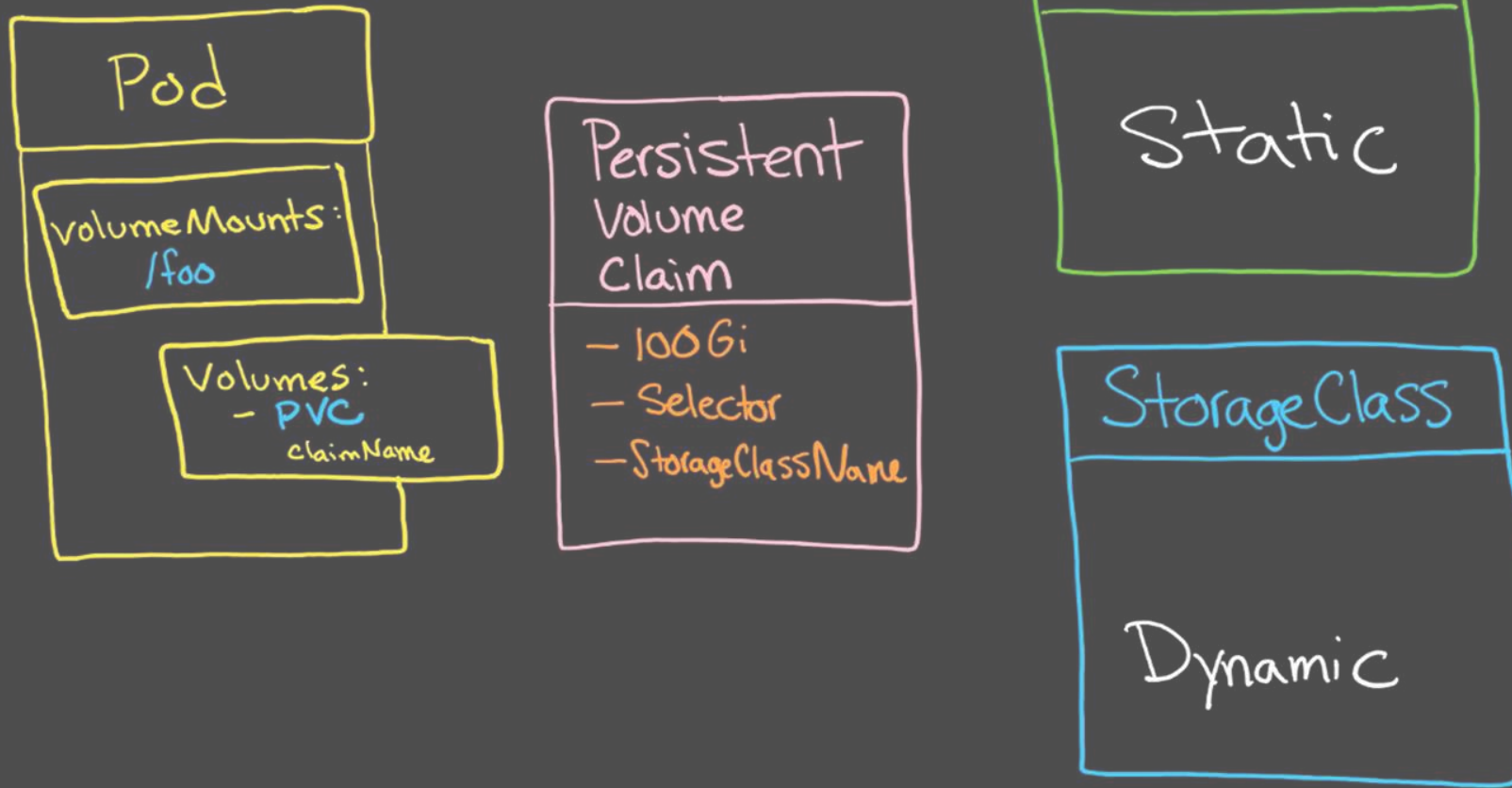
Persistent Storage

Storage Requirements



Storage that **doesn't depend on the pod lifecycle.**

Persistent Storage



Prometheus Software

- Prometheus is a free software application used for event monitoring and alerting.
- It records real-time metrics in a time series database built using an HTTP pull model, with flexible queries and real-time alerting.
- It collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and can trigger alerts if a condition is observed to be true.



Ref

- <https://kubernetes.io/docs/tutorials/>
- <https://kompose.io/>
- <https://phoenixnap.com/kb/install-kubernetes-on-ubuntu>
- <https://minikube.sigs.k8s.io/docs/start/>
- <https://github.com/ubuntu/microk8s>
- <https://docs.docker.com/get-started/part2/>
- <https://docs.docker.com/get-started/kube-deploy/>
- <https://golang.org/doc/code.html>