Introduction to Kubernetes (Facebook DevC Medan – 9th Meetup)

@BukaLapak Medan Office - 23rd February 2019

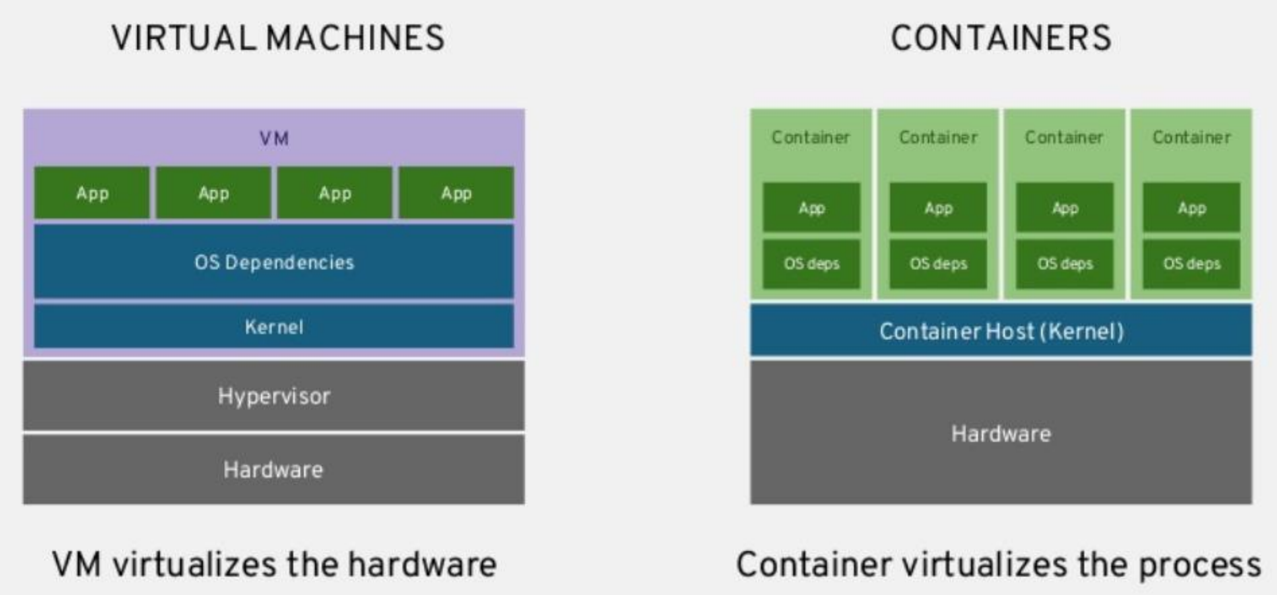
Presenter : Albert Suwandhi
Telegram : @albertsuwandhi
GitHub : github.com/albertsuwandhi

Agenda

- What is Kubernetes?
- Kubernetes Architecture
- Kubernetes Resources
- How to bootstrap your first Kubernetes Cluster
- Demo (Tentative)

Review - What is Container?

- Container is about isolation
- Combination between Linux Kernel features, especially : NameSpace and CGroups
- Docker, RKt, LXC (Container Run Time) are just wrapper around those!



We don’t want this to happen!!!

That’s why we need more than just containers!!

- Scheduling : Decide where to deploy containers
- Health Check : Keep containers running despite of failures
- Discovery : Find other containers
- Monitoring : Visibility into running containers
- Security : Access Control
- Scaling : Scale up and down
- Persistence : Survice data beyond container lifecycle
- Aggregation : Compose application from multiple containers

What is Kubernetes (1)

- Kubernetes is Greek for “Pilot” or “Helmsman of a ship”
- Kubernetes is a platform and container orchestration tool for automating deployment, scaling, and operations of application containers.
- Built from the lessons learned in the experiences of developing and running Google’s Borg and Omega : https://ai.google/research/pubs/pub43438
- Loosely coupled, collection of components centered around deploying, maintaining and scaling workloads
- Supports Containerd (docker), Rkt, Cri-o, Kata containers (formerly clear and hyper) and Virtlet
- Support Multiple Cloud and Bare Metal Environments

What is Kubernetes (2)

- Abstracts away the underlying hardware of the nodes and provides a uniform interface for workloads to be both deployed and consume the shared pool of resources.
- Works as an engine for resolving state by converging actual and the desired state of the system

  Me : “I want 3 healthy instances of NGINX to always be running.”

  Kubernetes : “Okay, I’ll ensure there are always 3 instances up and running.”

  Kubernetes: “Oh look, one has died. I’m going to attempt to spin up a new one.”

- Manage your applications like Cattle instead of like Pets
- Other orchestration engine : Swarm , Mesos , Nomad

Who Managed Kubernetes?

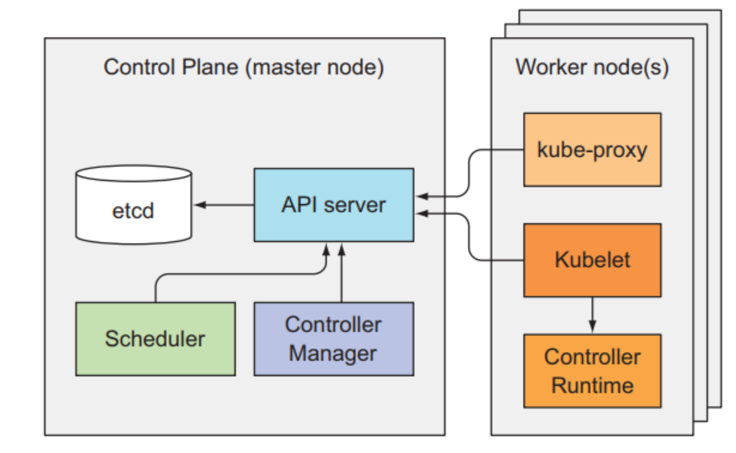
CLOUD NATIVE
COMPUTING FOUNDATION

The CNCF is a child entity of the Linux Foundation and operates as a vendor neutral governance group.

***Other Graduated Projects :** envoy Prometheus CoreDNS

Kubernetes Architecture

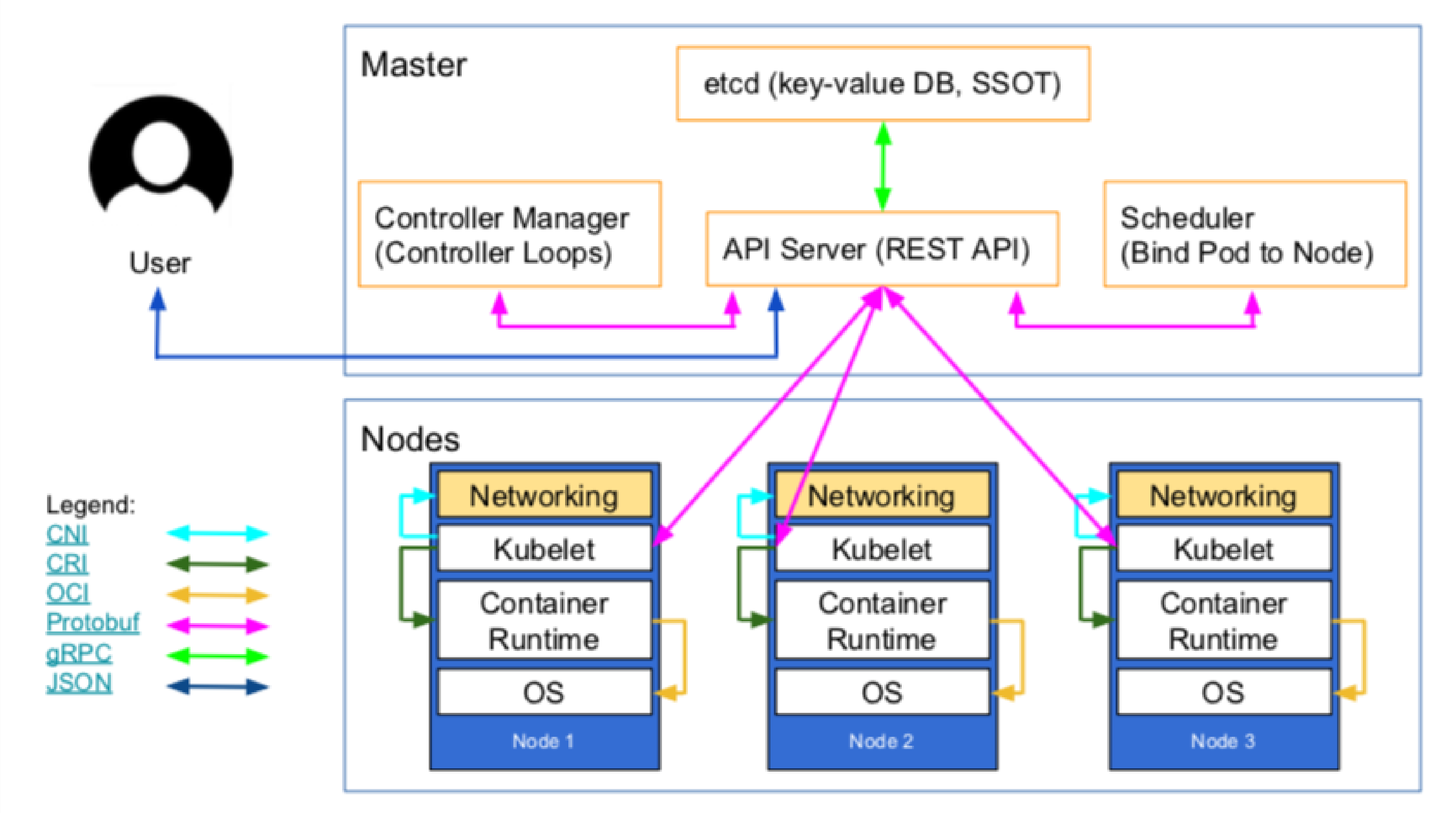
- Controller (Master) Components : API Server, Persistent Data Store, Scheduler and Controller Manager
- Node (Worker) Components : Kubelet, Kube-Proxy and Container Runtime Interface
- Additional Components : Kube-DNS, Container Networking Interface, Metrics API, Kubernetes Dashboard, Ingress Controller



Kubernetes Network Model

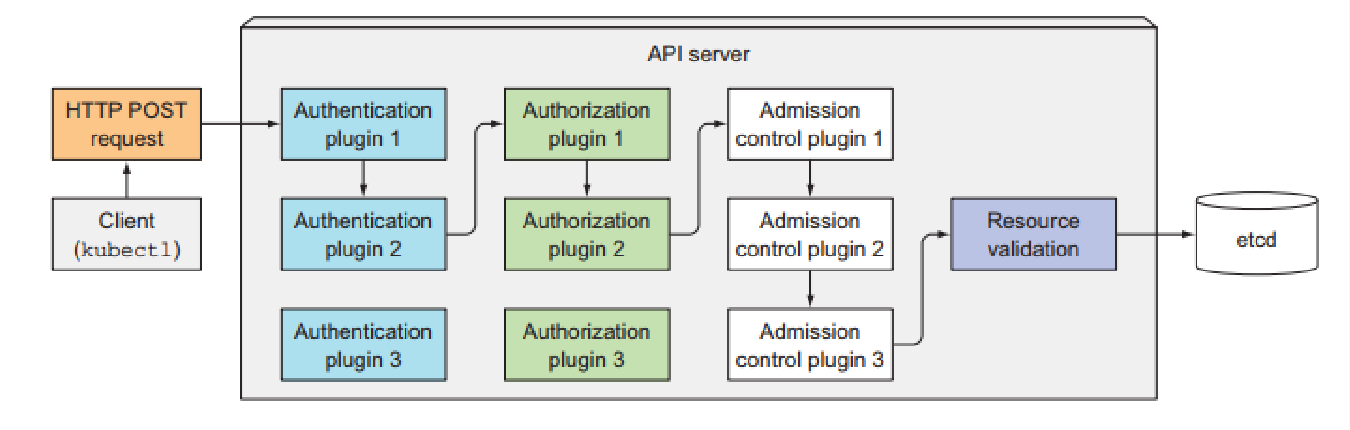
- All containers communicate without NAT
- All nodes communicate with containers without NAT
- Container sees its own IP as others see it
- Kubernetes doesn’t provide default network implementation, it leaves it to third party tools
- Some CNI Plugins Example : Flannel, Calico, Weave, Cilium, Kube-Router , AWS CNI, Kopeio, Romana, etc.
- https://kccncchina2018english.sched.com/event/FuKF/comprehensive-performance-benchmark-on-various-well-known-cni-plugins-giri-kuncoro-vijay-dhama-go-jek

Kubernetes High Architecture Overview



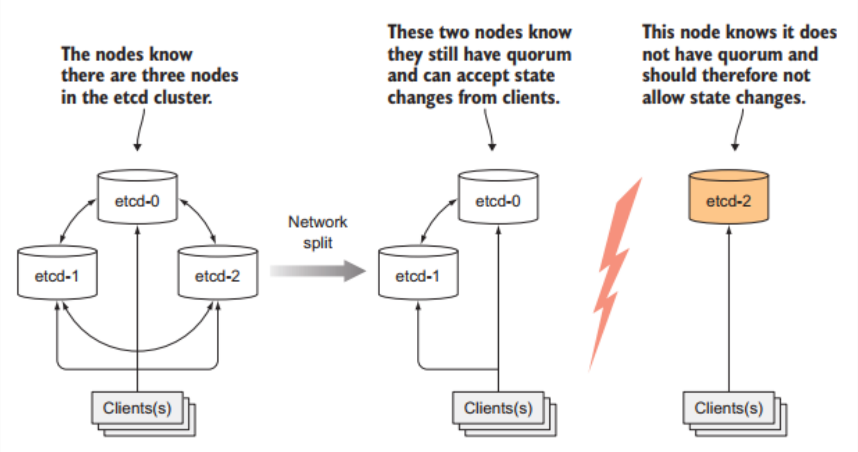
Kube-API Server

- Provides a forward facing REST interface into the kubernetes control plane and datastore.
- All clients and other applications interact with kubernetes strictly through the API Server.
- Acts as the gatekeeper to the cluster by handling authentication and authorization, request validation, mutation, and admission control in addition to being the front-end to the backing data store.



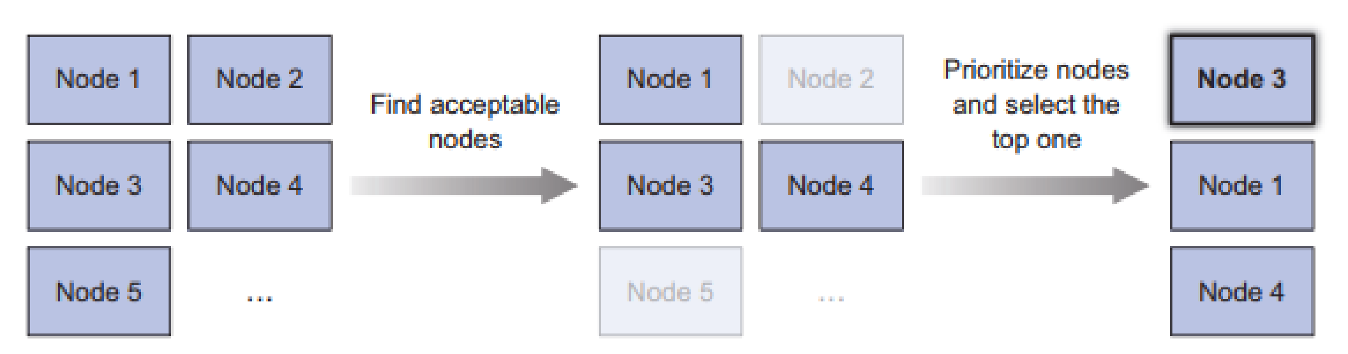
etcd

- etcd acts as the cluster datastore.
- provide a strong, consistent and highly available key-value store for persisting cluster state.
- Stores objects and config information
- Uses “Raft Consensus” among a quorum of systems to create a fault-tolerant consistent “view” of the cluster


The nodes know there are three nodes in the etcd cluster.
These two nodes know they still have quorum and can accept state changes from clients.
This node knows it does not have quorum and should therefore not allow state changes.
etcd-0
etcd-1
etcd-2
Network split
etcd-0
etcd-1
etcd-2
Clients(s)
Clients(s)
Clients(s)

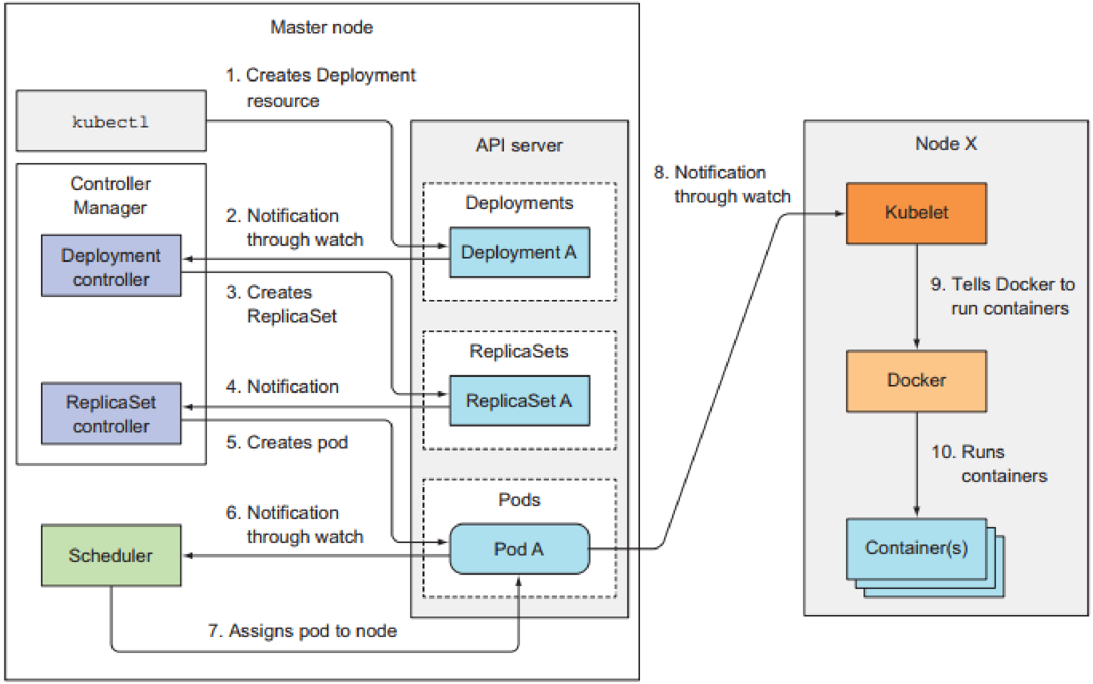
Kube-scheduler

- Verbose policy-rich engine that evaluates workload requirements and attempts to place it on a matching resource.
- Default scheduler uses bin packing.
- Workload Requirements can include: general hardware requirements, affinity/anti-affinity, labels, and other various custom resource requirements.



Kube-controller-manager

- Serves as the primary daemon that manages all core component control loops.
- Monitors the cluster state via the apiserver and steers the cluster towards the desired state



Kubelet

- Acts as the node agent responsible for managing the lifecycle of every pod on its host.
- Kubelet understands YAML container manifests that it can read from several sources:
  - file path
  - HTTP Endpoint
  - etcd watch acting on any change
  - HTTP Server mode accepting container manifests over a simple API

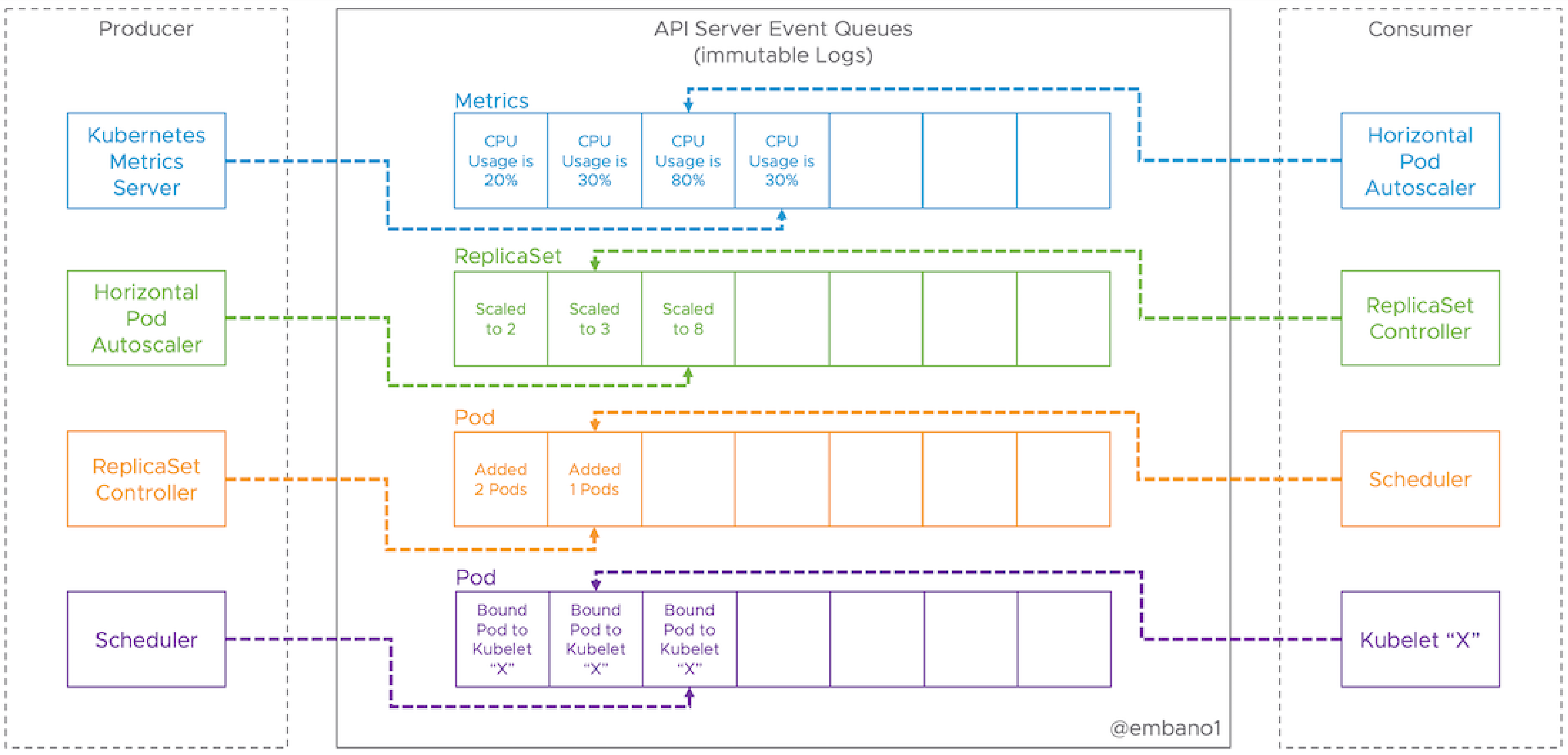
Kube-proxy

- Manages the network rules on each node.
- Performs connection forwarding or load balancing for Kubernetes cluster services.
- Available Proxy Modes:
  - Userspace
  - iptables
  - ipvs (default if supported)

Container Runtime Interface

- A container runtime is a CRI (Container RuntimeInterface) compatible application that executes and manages containers.
  - Docker
  - Cri-o
  - RKt
  - Kata (formerly clear and hyper)
  - Virtlet (VM CRI compatible runtime)

Kubernetes is a event driven architecture



Kubernetes resources explained (1)

	Resource (abbr.) [API version]	Description
	Namespace* (ns) [v1]	Enables organizing resources into non-overlapping groups (for example, per tenant)
Deploying Workloads	Pod (po) [v1]	The basic deployable unit containing one or more processes in co-located containers
	ReplicaSet	Keeps one or more pod replicas running
	ReplicationController	The older, less-powerful equivalent of a ReplicaSet
	Job	Runs pods that perform a completable task
	CronJob	Runs a scheduled job once or periodically
	DaemonSet	Runs one pod replica per node (on all nodes or only on those matching a node selector)
	StatefulSet	Runs stateful pods with a stable identity
	Deployment	Declarative deployment and updates of pods

Kubernetes resources explained (2)

	Resource (abbr.) [API version]	Description
Services	**Service** (svc) [v1]	Exposes one or more pods at a single and stable IP address and port pair
	Endpoints (ep) [v1]	Defines which pods (or other servers) are exposed through a service
	Ingress (ing) [extensions/v1beta1]	Exposes one or more services to external clients through a single externally reachable IP address
Config	ConfigMap (cm) [v1]	A key-value map for storing non-sensitive config options for apps and exposing it to them
	Secret [v1]	Like a ConfigMap, but for sensitive data
Storage	PersistentVolume* (pv) [v1]	Points to persistent storage that can be mounted into a pod through a PersistentVolumeClaim
	PersistentVolumeClaim (pvc) [v1]	A request for and claim to a PersistentVolume
	StorageClass* (sc) [storage.k8s.io/v1]	Defines the type of storage in a PersistentVolumeClaim

Kubernetes resources explained (3)

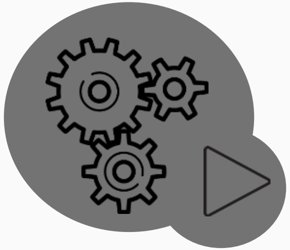
	Resource (abbr.) [API version]	Description
Scaling	HorizontalPodAutoscaler (hpa) [autoscaling/v2beta1**]	Automatically scales number of pod replicas based on CPU usage or another metric
	PodDisruptionBudget (pdb) [policy/v1beta1]	Defines the minimum number of pods that must remain running when evacuating nodes
Resources	LimitRange (limits) [v1]	Defines the min, max, default limits, and default requests for pods in a namespace
	ResourceQuota (quota) [v1]	Defines the amount of computational resources available to pods in the namespace
Cluster state	Node* (no) [v1]	Represents a Kubernetes worker node
	Cluster* [federation/v1beta1]	A Kubernetes cluster (used in cluster federation)
	ComponentStatus* (cs) [v1]	Status of a Control Plane component
	Event (ev) [v1]	A report of something that occurred in the cluster

Kubernetes resources explained (4)

	Resource (abbr.) [API version]	Description
Security	ServiceAccount (sa) [v1]	An account used by apps running in pods
	Role [rbac.authorization.k8s.io/v1]	Defines which actions a subject may perform on which resources (per namespace)
	ClusterRole* [rbac.authorization.k8s.io/v1]	Like Role, but for cluster-level resources or to grant access to resources across all namespaces
	RoleBinding [rbac.authorization.k8s.io/v1]	Defines who can perform the actions defined in a Role or ClusterRole (within a namespace)
	ClusterRoleBinding* [rbac.authorization.k8s.io/v1]	Like RoleBinding, but across all namespaces
	PodSecurityPolicy* (psp) [extensions/v1beta1]	A cluster-level resource that defines which security-sensitive features pods can use
	NetworkPolicy (netpol) [networking.k8s.io/v1]	Isolates the network between pods by specifying which pods can connect to each other

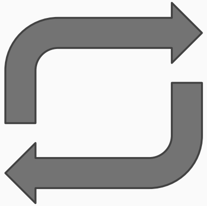
Kubernetes Basic Concepts

Pod



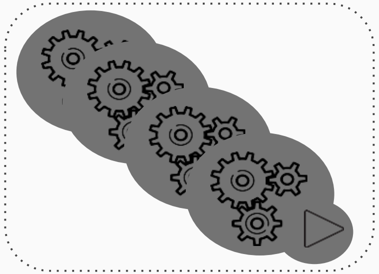
One or More Containers
Shared IP
Shared Storage Volume
Shared Resources
Shared Lifecycle

Replication Controller / Deployment



Ensures that a specified number of pod replicas are running at any one time

Service



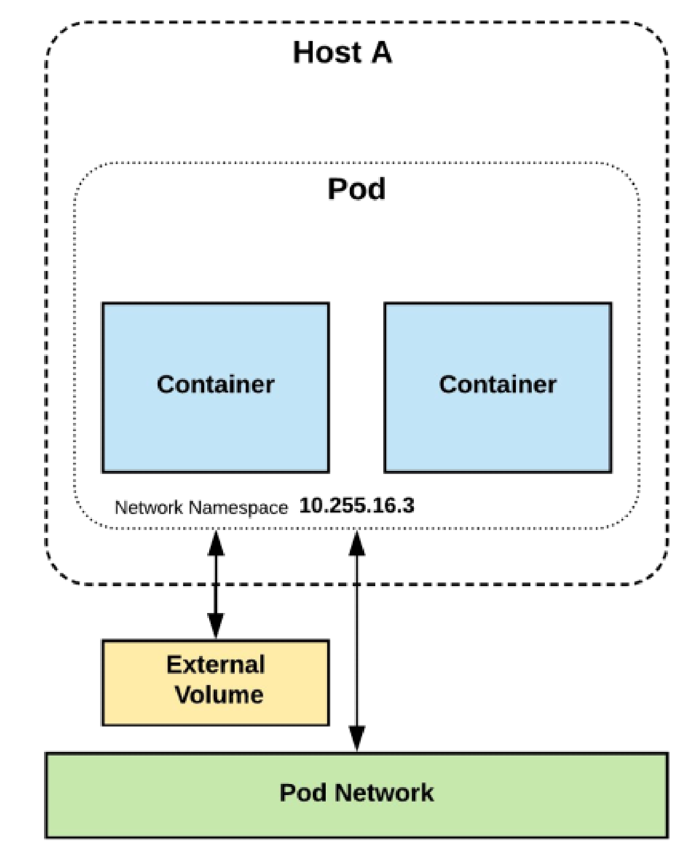
Grouping of pods, act as one, has stable virtual IP and DNS name

Label

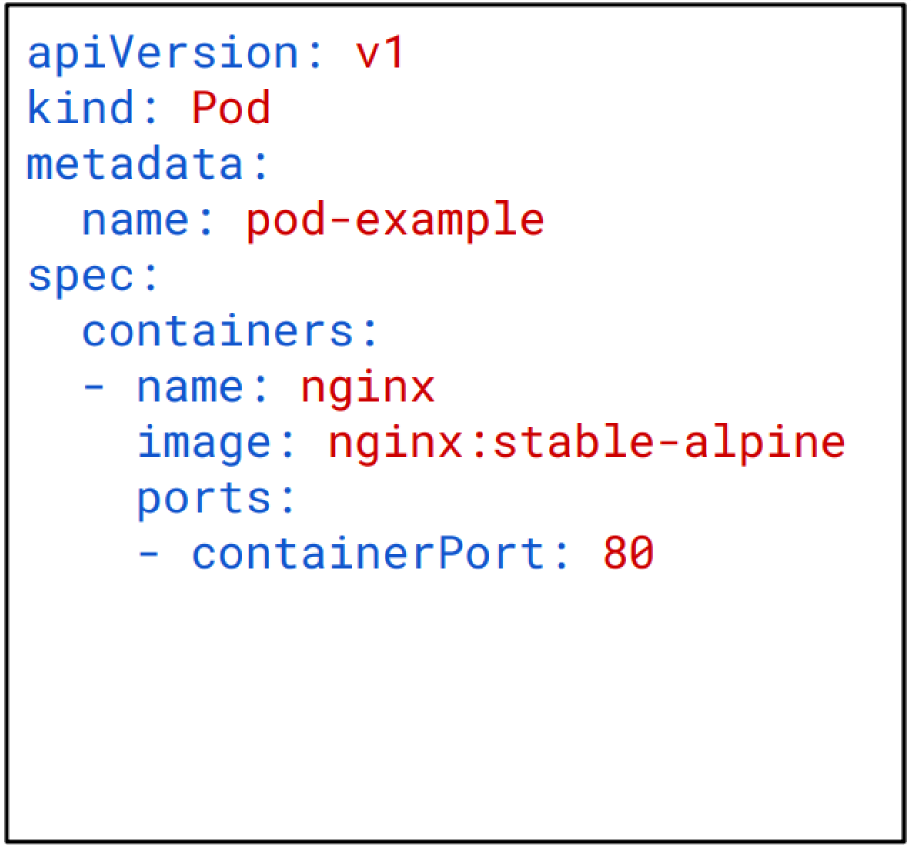
Key/Value pairs associated with Kubernetes objects (e.g. env=production)

Pod

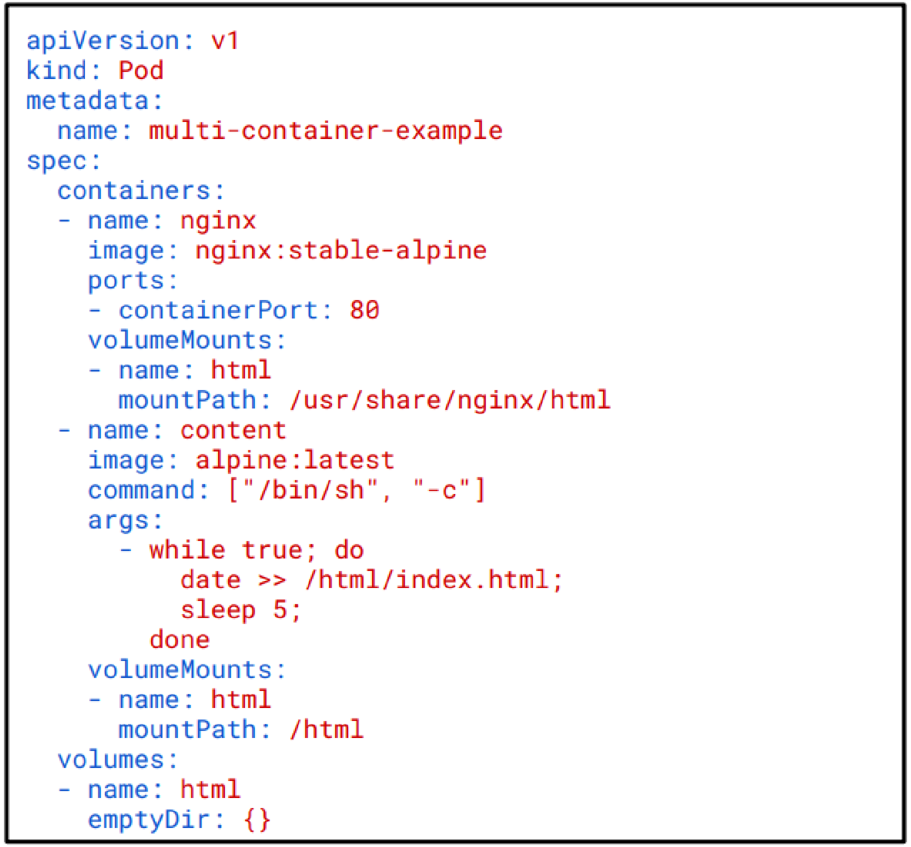
- Atomic unit or smallest “unit of work”of Kubernetes.
- Foundational building block of Kubernetes Workloads.
- Pods are one or more containers that share volumes, a network namespace, and are a part of a single context.
- We almost never manage pods directly, but through other resources.
- Pods are EPHEMERAL



Pod Manifest Example in YAML



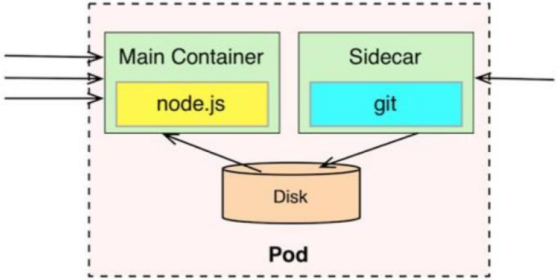
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod-example
spec:
  containers:
  - name: nginx
    image: nginx:stable-alpine
    ports:
    - containerPort: 80
```



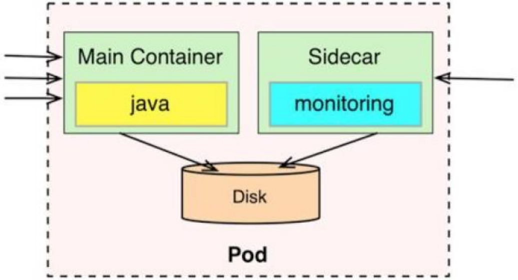
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: multi-container-example
spec:
  containers:
  - name: nginx
    image: nginx:stable-alpine
    ports:
    - containerPort: 80
    volumeMounts:
    - name: html
      mountPath: /usr/share/nginx/html
  - name: content
    image: alpine:latest
    command: ["/bin/sh", "-c"]
    args:
      - while true; do
          date >> /html/index.html;
          sleep 5;
        done
    volumeMounts:
    - name: html
      mountPath: /html
  volumes:
  - name: html
    emptyDir: {}
```

Design Pattern

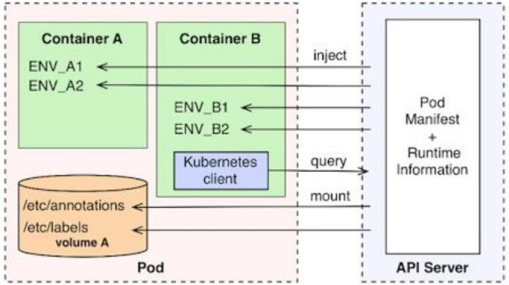
Sidecar Pattern



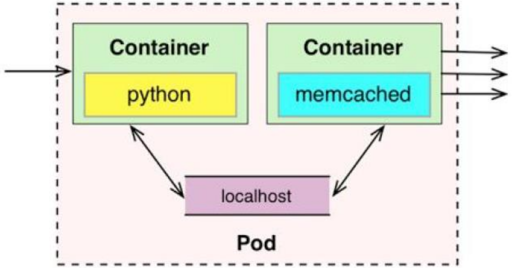
Adapter Pattern



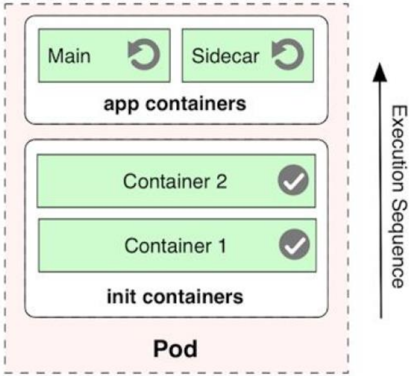
Self Awareness Pattern



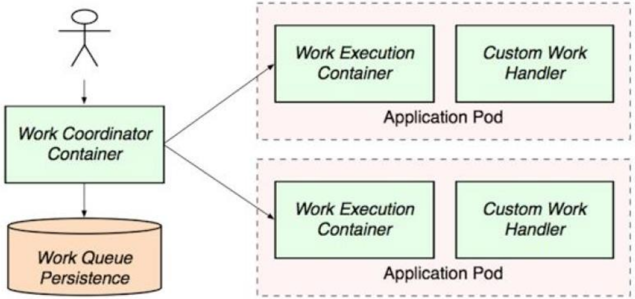
Ambassador Pattern



Initializer Pattern

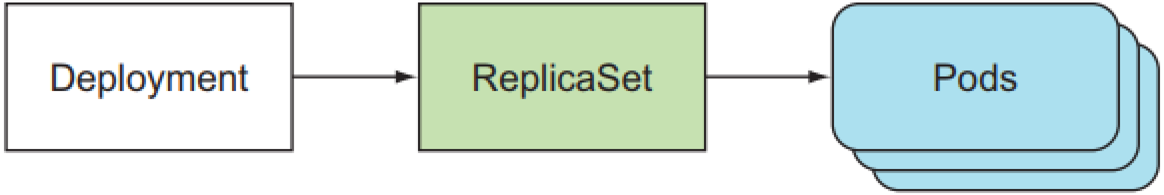


Work Queue Pattern

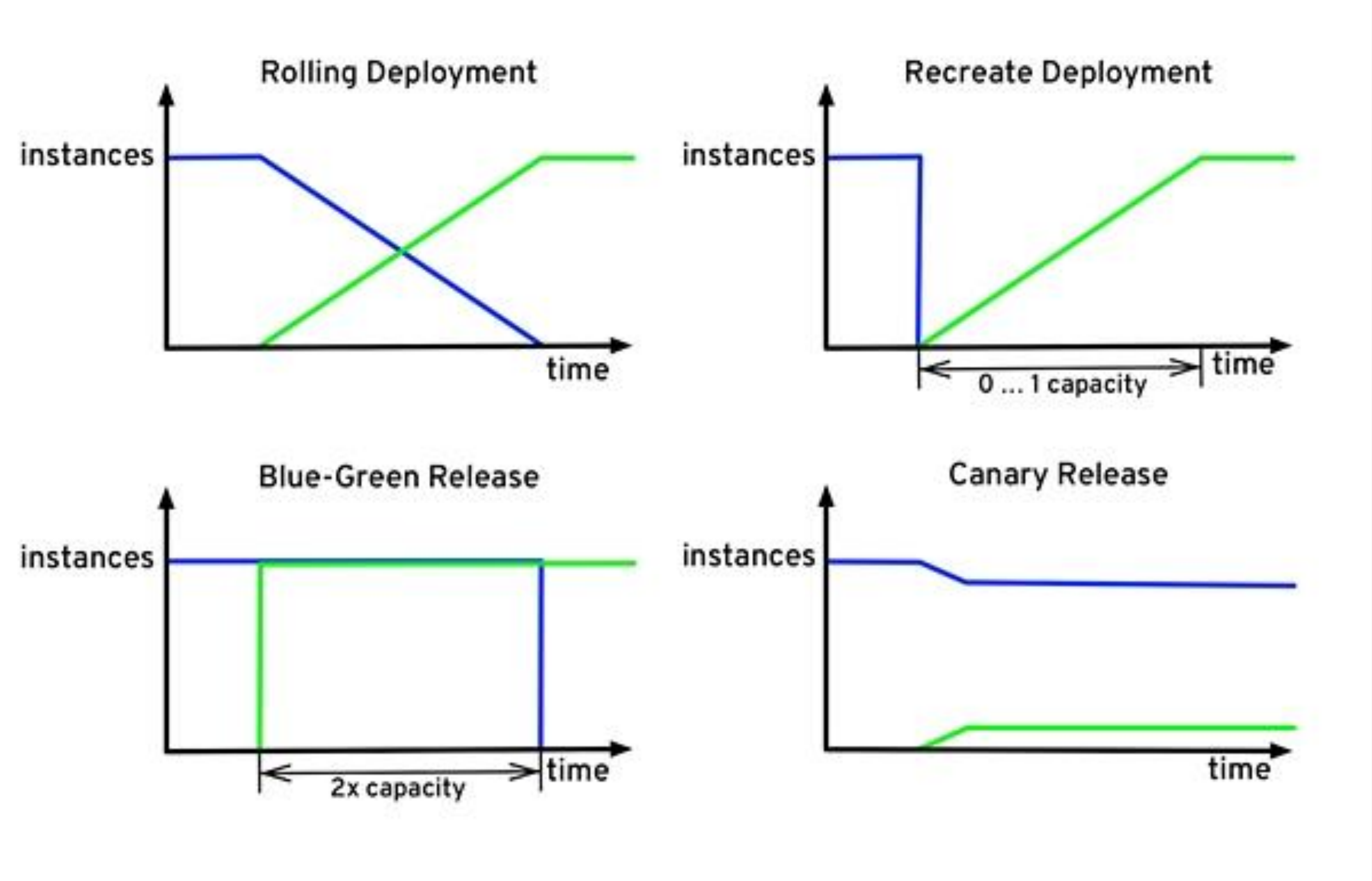


Deployment

- Declarative method of managing Pods via ReplicaSets.
- Provide rollback functionality and update control.
- Updates are managed through the pod-template-hash label.
- Each iteration creates a unique label that is assigned to both the ReplicaSet and subsequent Pods



Deployment Release Strategy

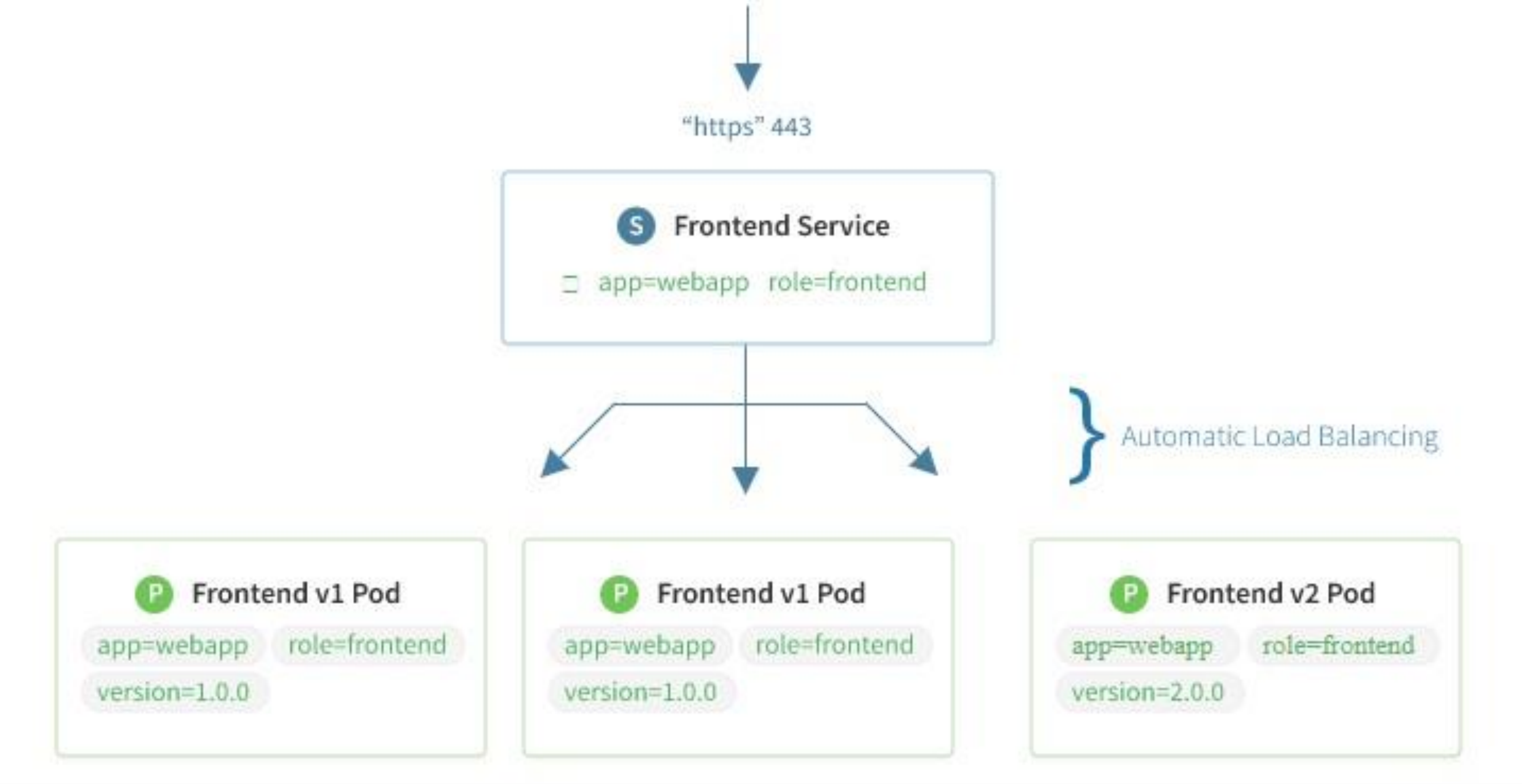


Services

- Unified method of accessing the exposed workloads of Pods.
- Target Pods using equality based selectors.
- Uses kube-proxy to provide simple load-balancing.
- kube-proxy acts as a daemon that creates local entries in the host’s iptables for every service.
- Durable resource (unlike Pods)
  - static cluster-unique IP
  - static namespaced DNS name

<service name>.<namespace>.svc.cluster.local

Services – Persistent End Point for Pods



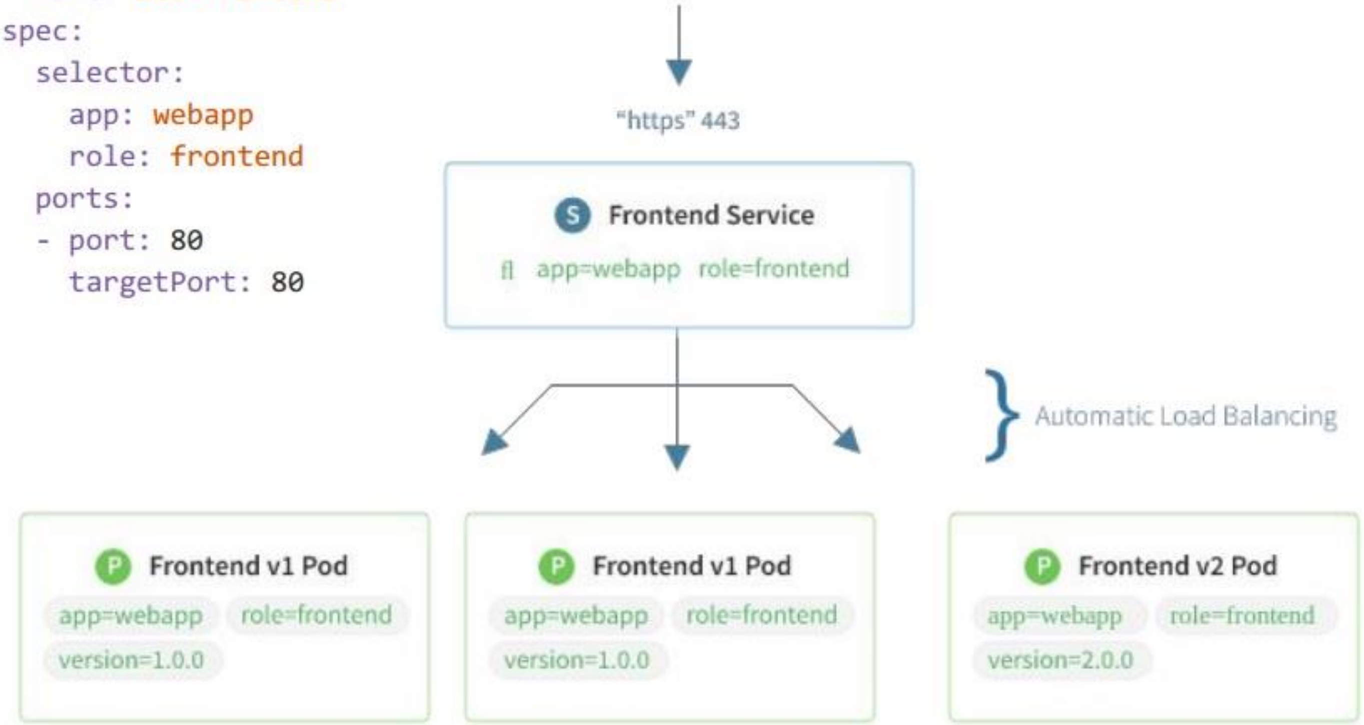
Services Types

- ClusterIP : ClusterIP services exposes a service on a strictly cluster internal virtual IP
- NodePort : NodePort services extend the ClusterIP service and exposes a port on every node’s IP
- Load Balancer : LoadBalancer service extend NodePort. It works in conjunction with an external system (Load Balancer) to map a cluster external IP to the exposed service
- External Name : ExternalName is used to reference endpoints OUTSIDE the cluster. It creates an internal CNAME DNS entry that aliases another

Deployment and Service Manifest Example in YAML

```yaml
apiVersion: apps/v1beta2
kind: Deployment
metadata:
  labels:
    app: webapp
    role: frontend
  name: web-frontend
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: webapp
        role: frontend
    spec:
      containers:
      - image: nginx:1.13.1
        name: nginx
        ports:
        - containerPort: 80
          name: http
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: web-frontend
spec:
  selector:
    app: webapp
    role: frontend
  ports:
  - port: 80
    targetPort: 80
```



Kubernetes is a very broad topics!
We don’t have enough time to discuss all Kubernetes Resources in less than an hour ☺

How to setup our Kubernetes Cluster

- **Local Install : Single Node Cluster with MiniKube**
- **Manual Install : kubeadm (baremetal – cloud)**
- **Amazon Web Services : Kops**
- **Azure : AKS**
- **Google Cloud Platform : GKE**
- **Kubespray**
- **Digital Ocean**
- **[Play with Kubernetes]right away in your browser!**
- **Kubernetes the Hard Way**
- **etc**

do-cka.txt

aws kops

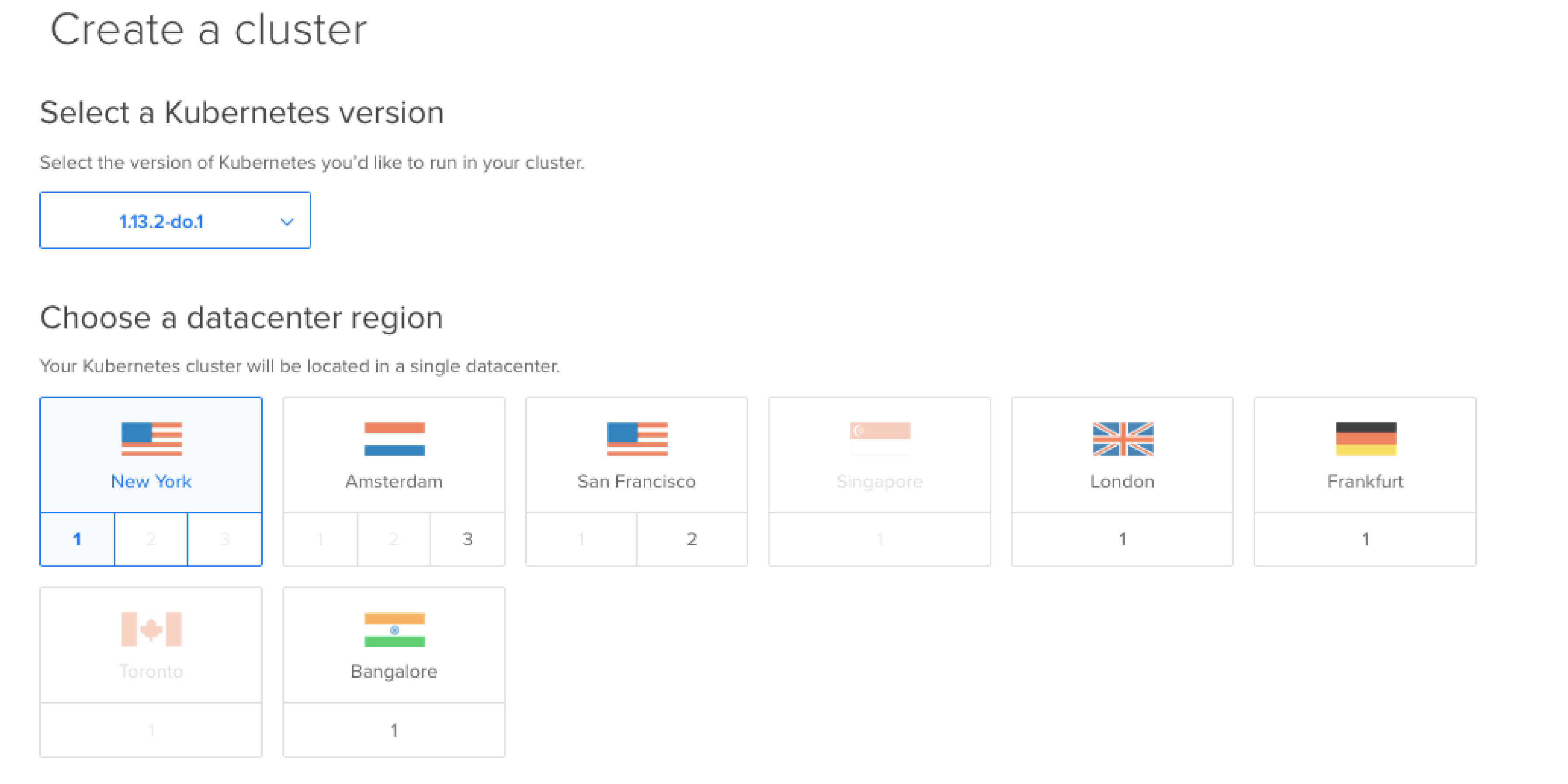
minikube

Kubeadm

Google Cloud Platform - GKE

1. gcloud auth login
2. gcloud projects list
3. gcloud config set project [PROJECT-NAME]
4. gcloud config set compute/zone asia-southeast1-a
5. gcloud container clusters create [CLUSTER NAME] --num-nodes 3 --zone asia-southeast1-a
6. gcloud container clusters get credentials [CLUSTER-NAME]

Digital Ocean – Managed Kubernetes



Create a cluster

Select a Kubernetes version

Select the version of Kubernetes you'd like to run in your cluster.

1.13.2-do.1

Choose a datacenter region

Your Kubernetes cluster will be located in a single datacenter.

New York
1 2 3

Amsterdam
1 2 3

San Francisco
1 2

Singapore
1

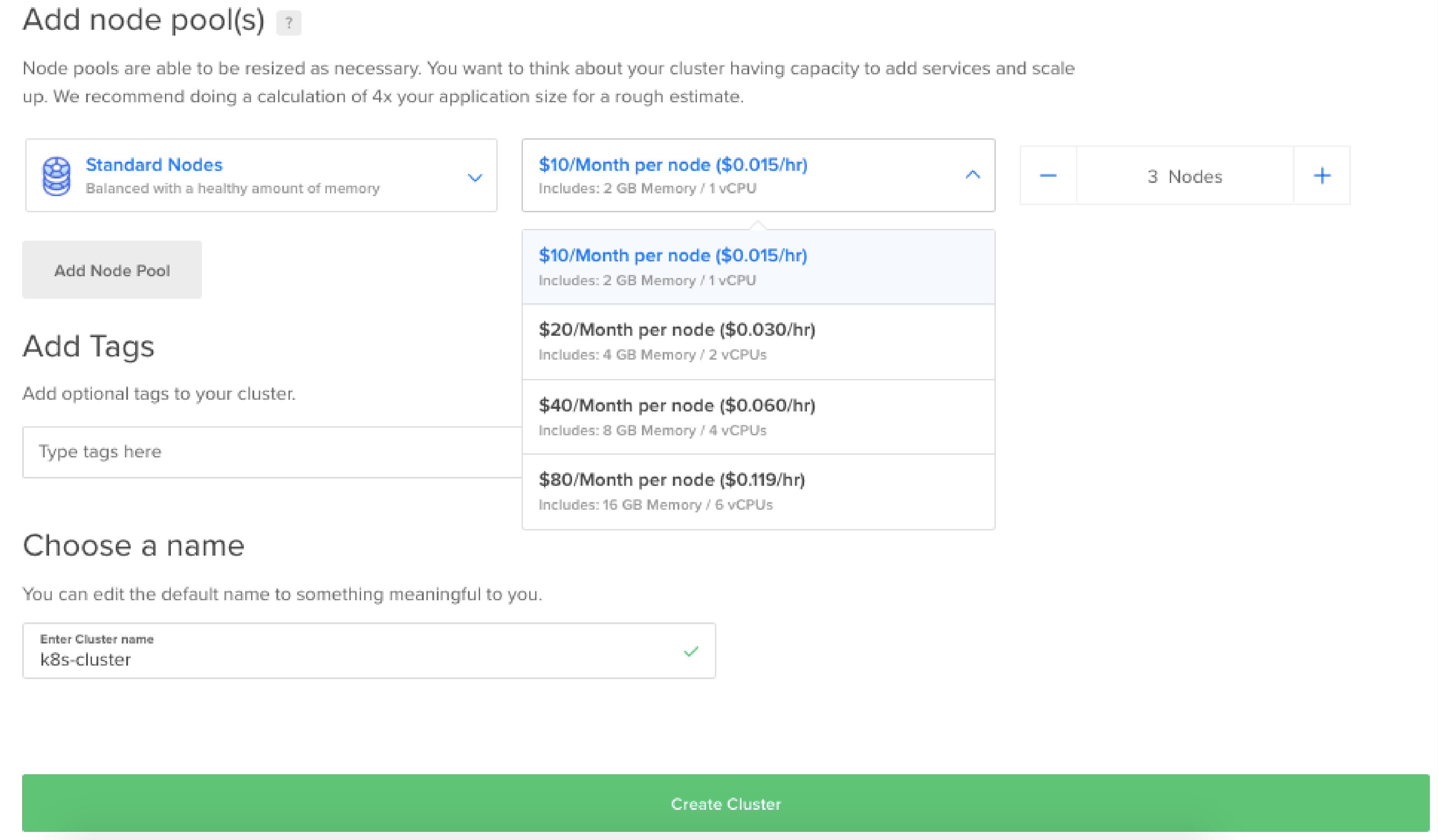
London
1

Frankfurt
1

Toronto
1

Bangalore
1

Digital Ocean – Managed Kubernetes



Add node pool(s)

Node pools are able to be resized as necessary. You want to think about your cluster having capacity to add services and scale up. We recommend doing a calculation of 4x your application size for a rough estimate.

Standard Nodes
Balanced with a healthy amount of memory

$10/Month per node ($0.015/hr)
Includes: 2 GB Memory / 1 vCPU

3 Nodes

- $10/Month per node ($0.015/hr)
  Includes: 2 GB Memory / 1 vCPU
- $20/Month per node ($0.030/hr)
  Includes: 4 GB Memory / 2 vCPUs
- $40/Month per node ($0.060/hr)
  Includes: 8 GB Memory / 4 vCPUs
- $80/Month per node ($0.119/hr)
  Includes: 16 GB Memory / 6 vCPUs

Add Tags

Add optional tags to your cluster.

Type tags here

Choose a name

You can edit the default name to something meaningful to you.

Enter Cluster name
k8s-cluster

kubeadm

- Install Master VM and install components

```
$ sudo apt update; sudo apt install -y kubectl kubelet kubeadm
$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
```

- Install nodes and join to master :

```
$ kubeadm join --token <token> control-plane-ip
```

- Install CNI :

```
$ kubectl apply –f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```

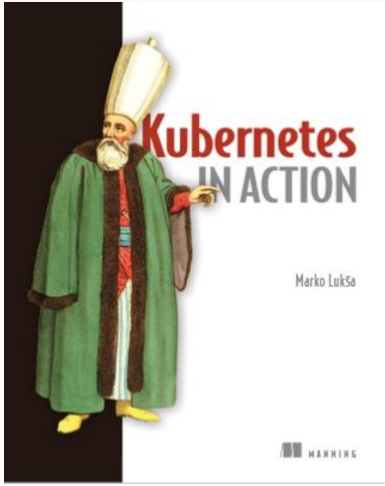
Demo - Kubernetes in Action!

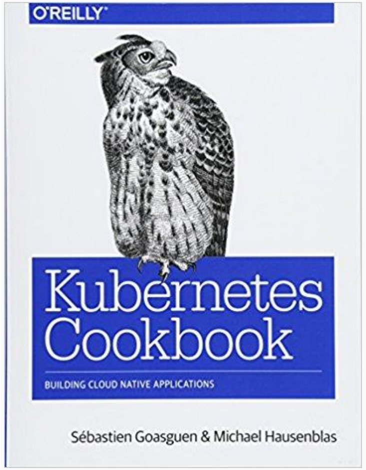
Demo Scenario

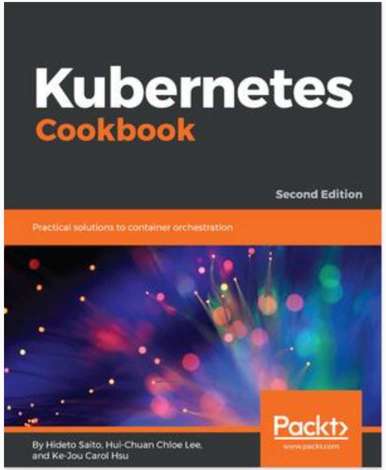
- kubectl basics
- Interacting with API Server
- Deployments
- Kubernetes Dashboard
- Helm Chart

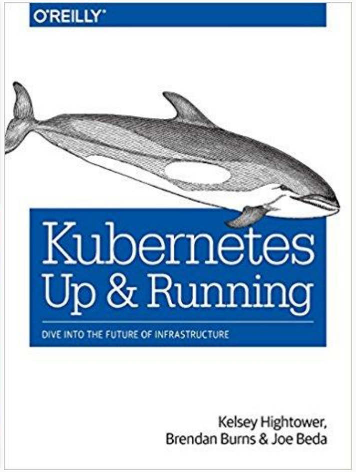
Kubernetes Learning Resources List

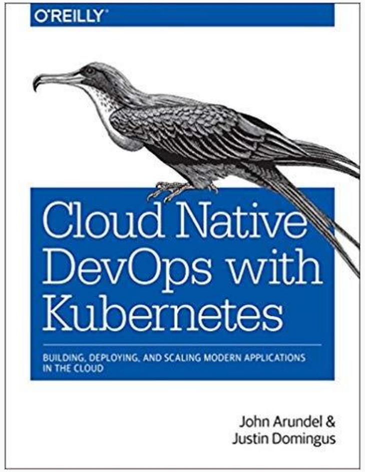
- Almost everything you need to know about Kubernetes : https://bit.ly/K8SResources
- Recommended Books :




O'REILLY
Kubernetes Cookbook
BUILDING CLOUD NATIVE APPLICATIONS
Sébastien Goasguen & Michael Hausenblas


Kubernetes
Cookbook
Second Edition
Practical solutions to container orchestration
By Hideto Saito, Hui-Chuan Chloe Lee, and Ke-Jou Carol Hsu
Packt>
www.packt.com


O'REILLY
Kubernetes Up & Running
DIVE INTO THE FUTURE OF INFRASTRUCTURE
Kelsey Hightower, Brendan Burns & Joe Beda


O'REILLY
Cloud Native DevOps with Kubernetes
BUILDING, DEPLOYING, AND SCALING MODERN APPLICATIONS IN THE CLOUD
John Arundel & Justin Domingus

Thank You

Q&A

Join :

https://www.facebook.com/groups/DevCMedan

https://t.me/kubernetesindonesia