Kubernetes Foundations Training

Tomasz Cholewa <tomasz@cloudowski.com>

Agenda

Day 1

Container basics

Operating docker containers

Running containers from available images

Providing configuration and storage for containers

Multi-container configurations using docker-compose

Building and publishing container images

Agenda

Day 2

Kubernetes architecture and basic components

Managing objects in k8s

Pod and its main features

Scaling of applications runnins as pods

Storing data on persistent storage

Agenda

Day 3

Exposing application using Service object

Exposing http applications using Ingress

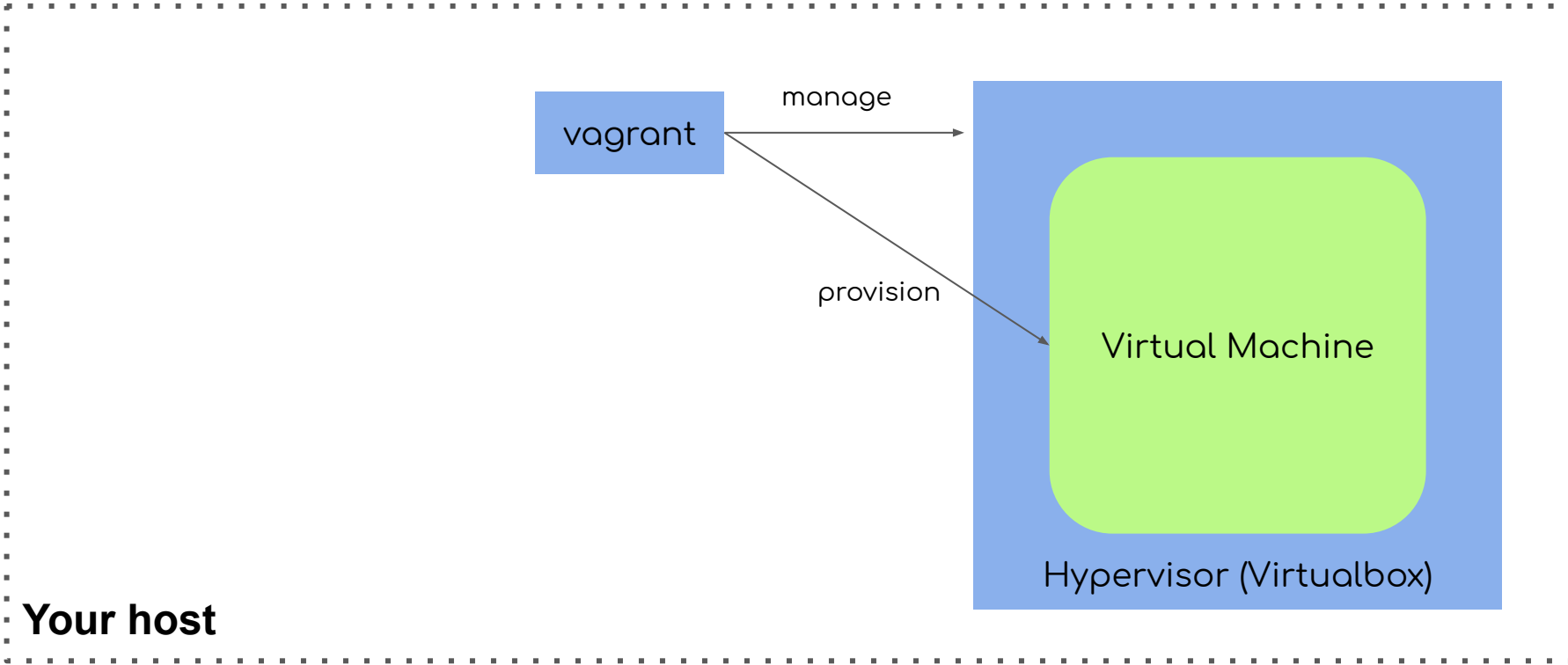
Basic logs maintenance and monitoring

Managing applications using Deployment

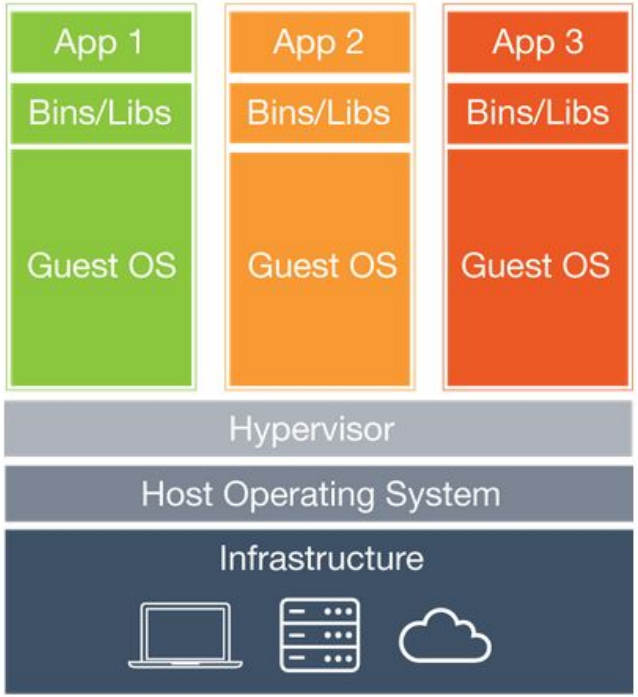
Installing apps using Helm and Charts


Quest

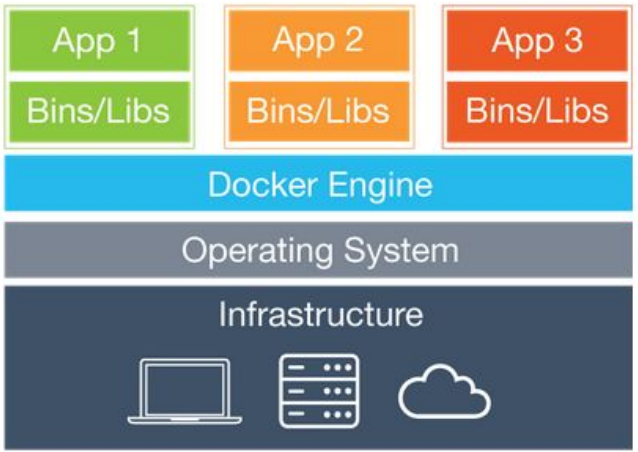
Vagrant


vagrant
manage
provision
Virtual Machine
Hypervisor (Virtualbox)
Your host

Containers vs. Virtual Machines



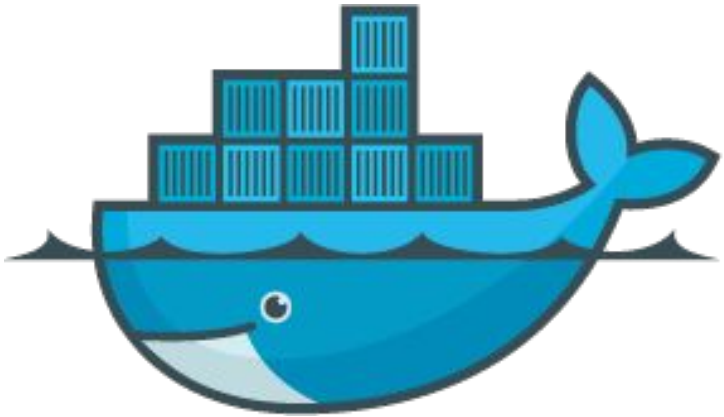
Virtual Machines



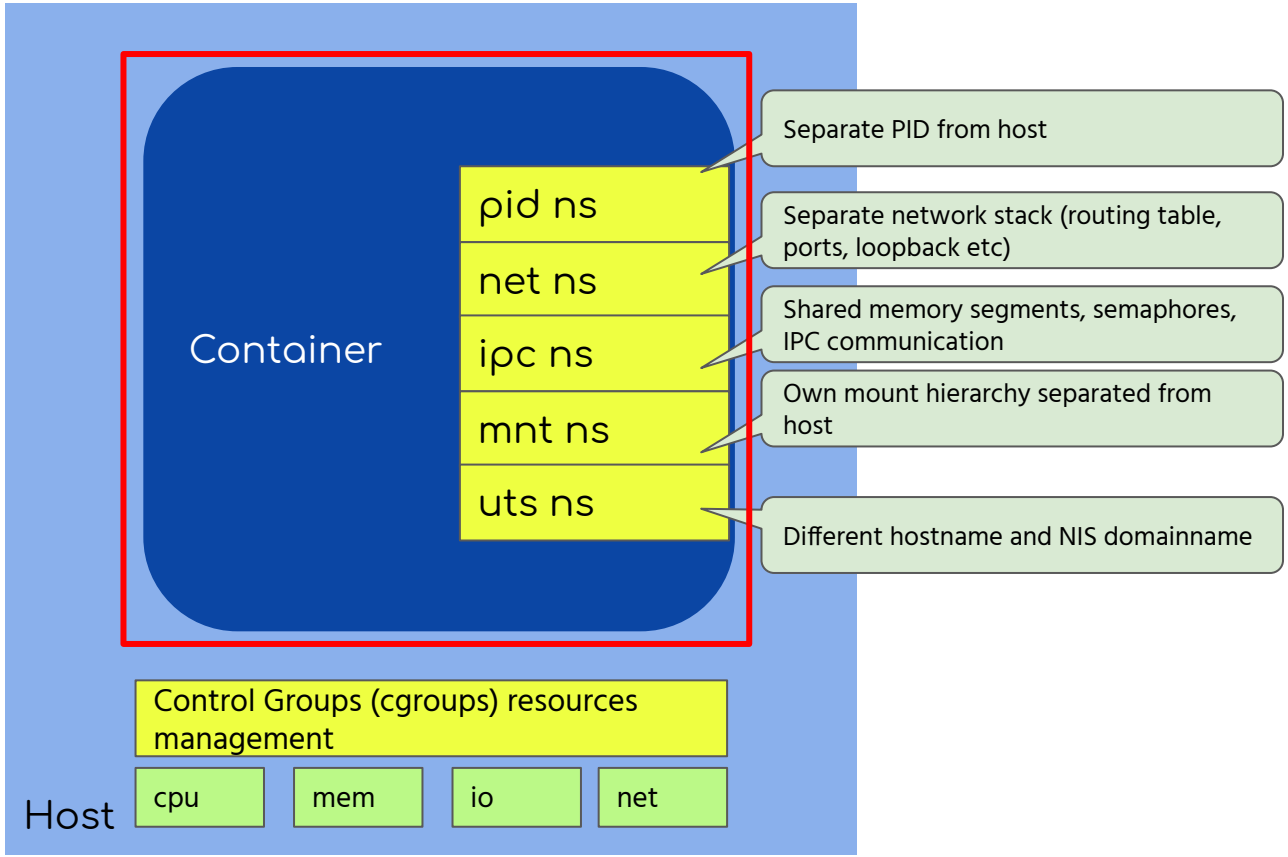
Containers

Why containers?

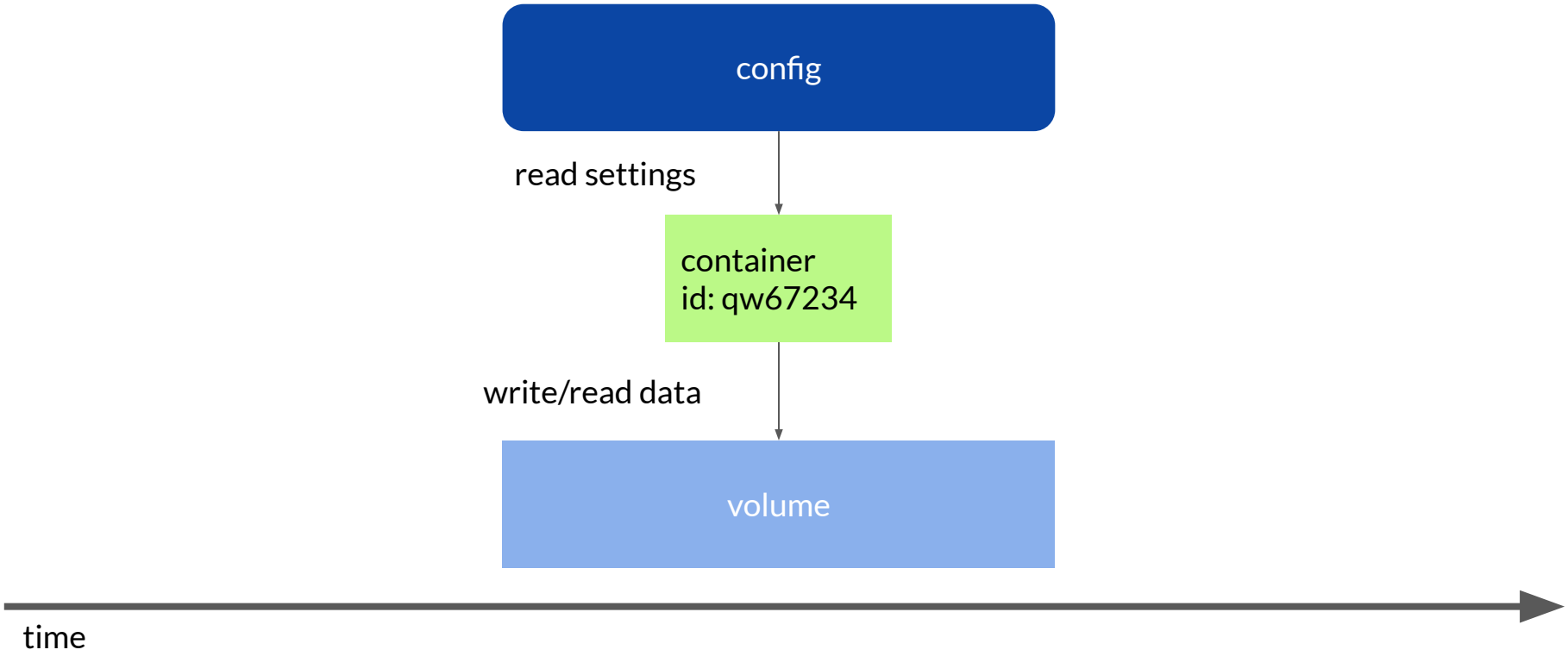
1. Easy to use
2. Portability
3. Security
4. Speed



Container isolation



Containers, peristent data and configuration



Container image naming

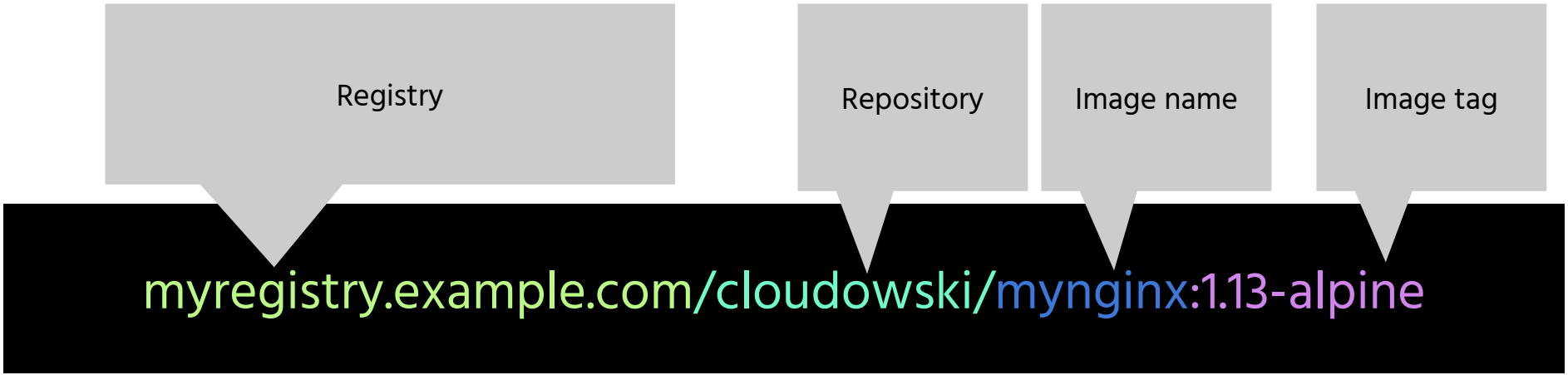
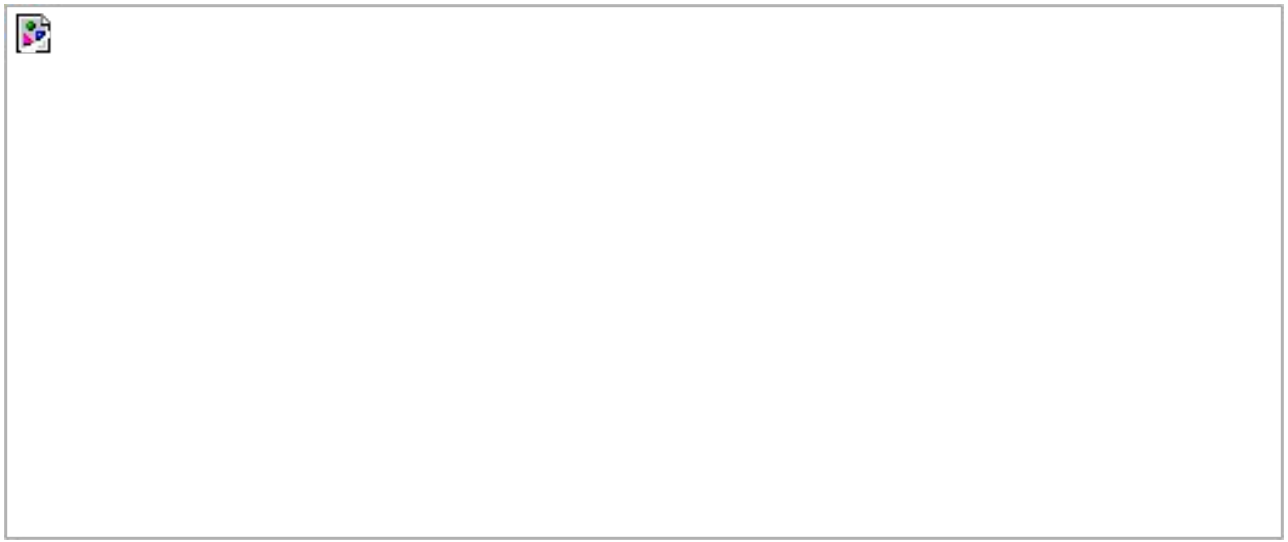
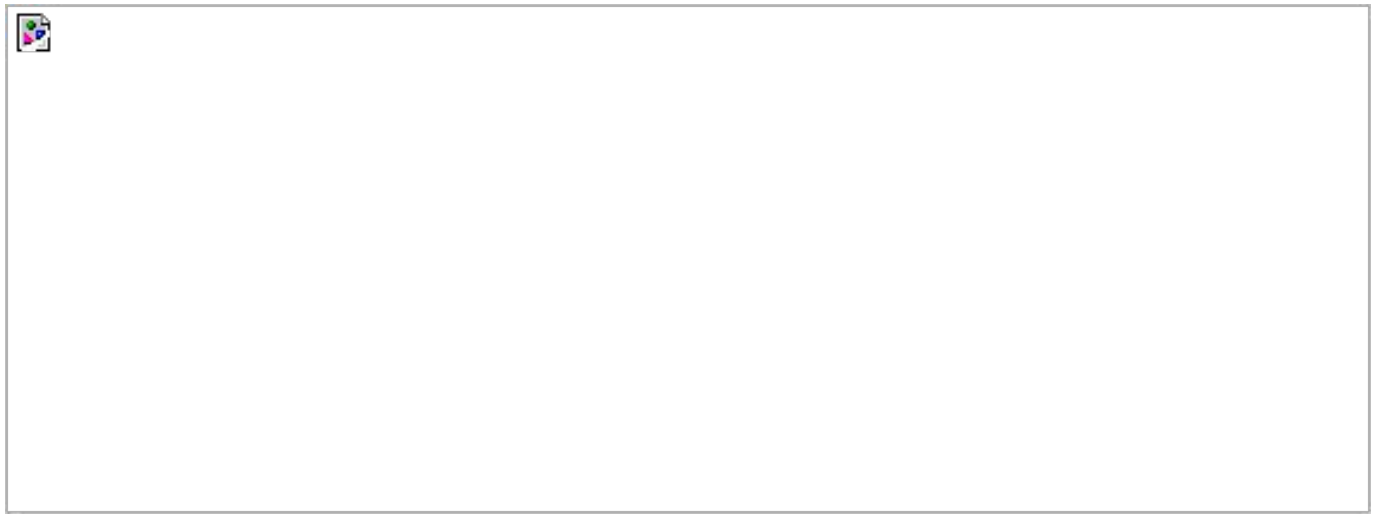


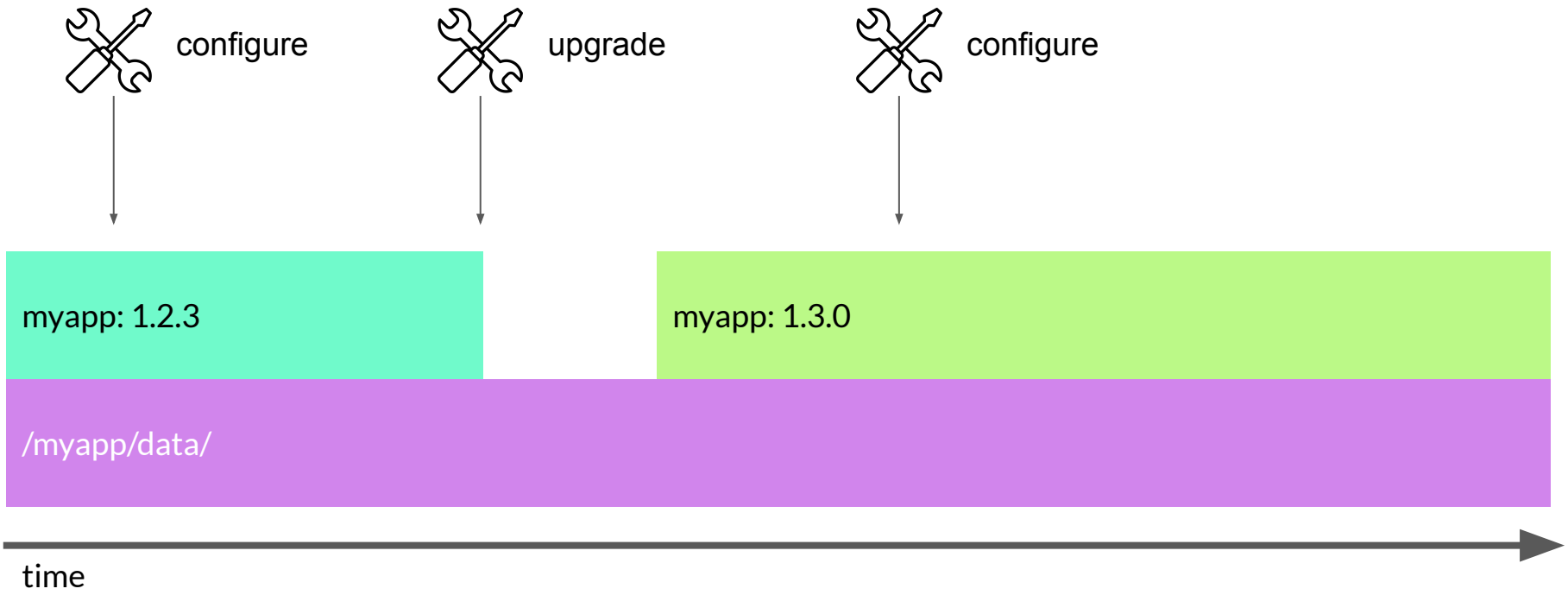
Image layers



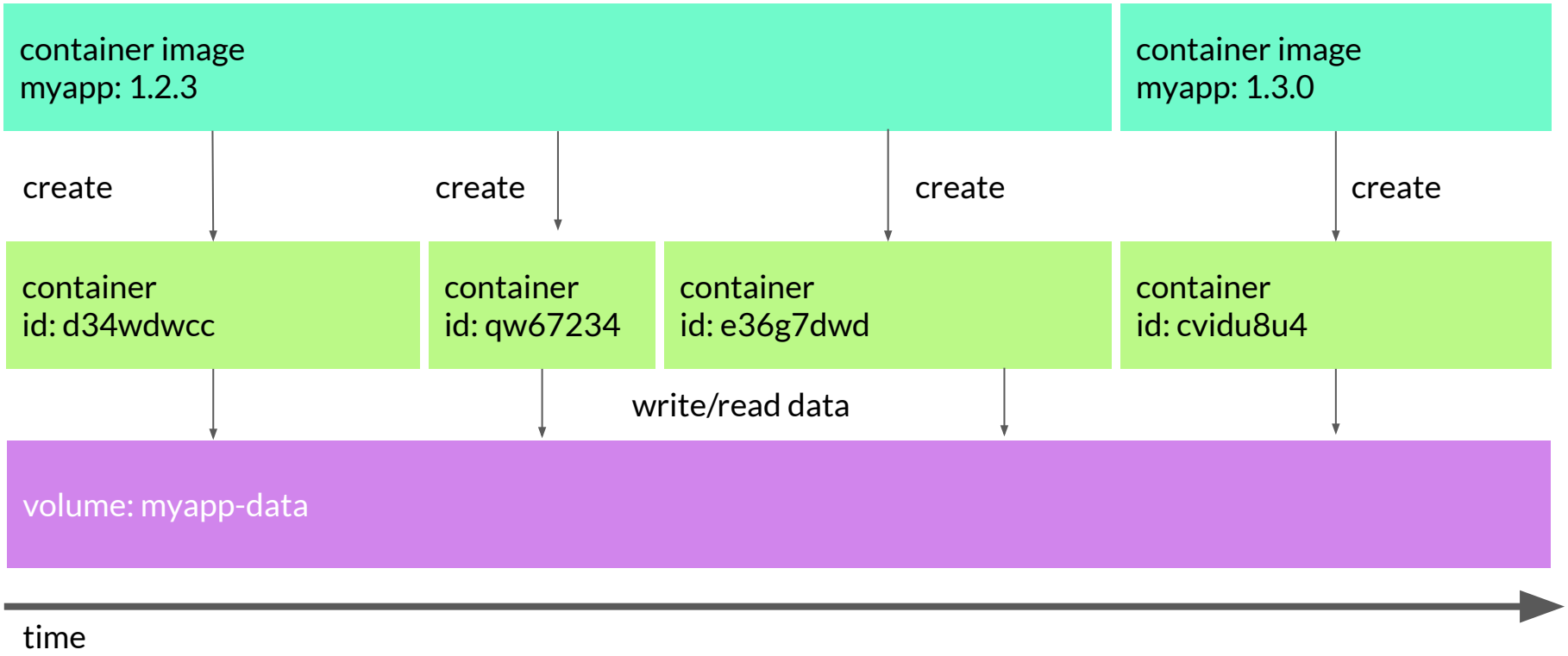
Creating new image from container



Traditional upgrade and configuration



Ephemeral containers with persistent data



10 tips for building good container images

1. Keep them small

10 tips for building good container images

2. Single app/single purpose for single image

10 tips for building good container images

3. Leverage docker caching

10 tips for building good container images

4. Create few layers with only necessary tools

10 tips for building good container images

5. Always put EXPOSE for all exposed ports

10 tips for building good container images

6. Assign proper tags

10 tips for building good container images

7. Create universal Dockerfiles configurable for future releases

10 tips for building good container images

8. Handle signals properly

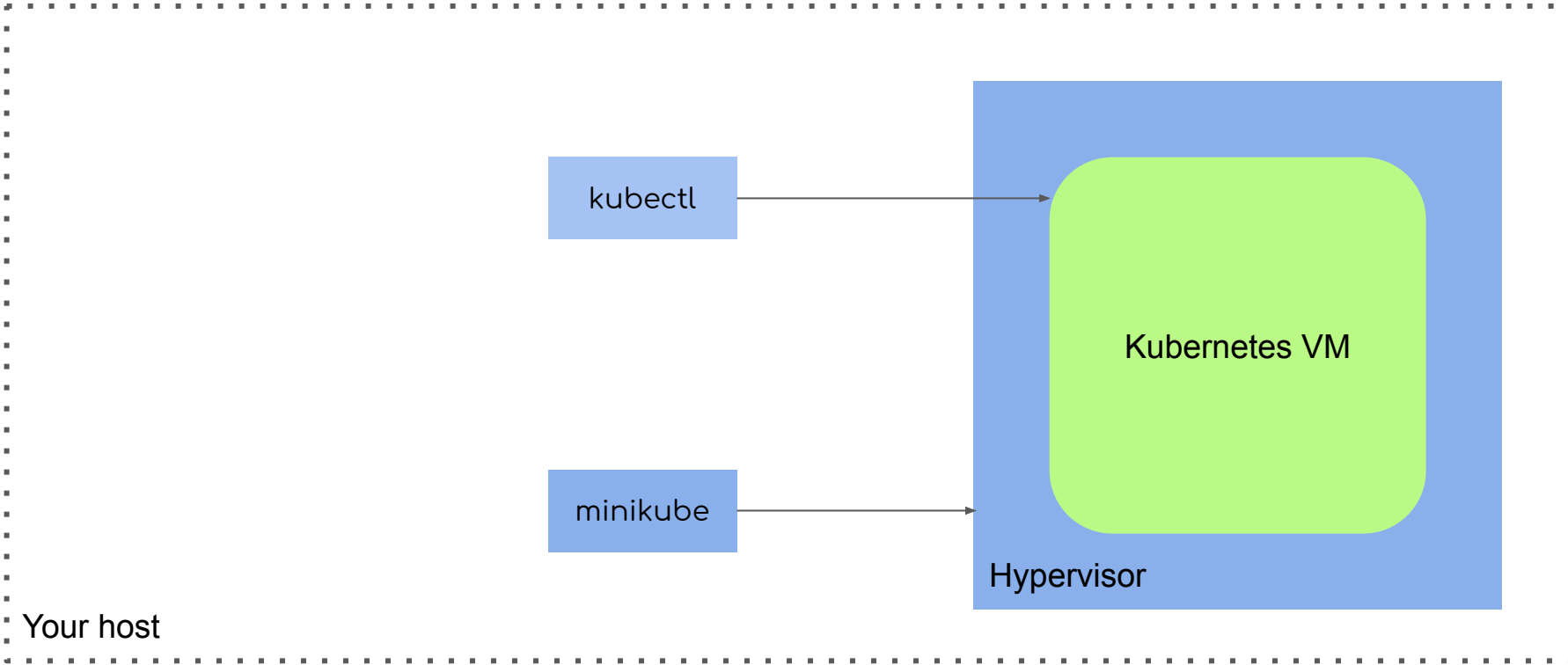
9. Always prefer official images instead of custom or built in-house

10 tips for building good container images

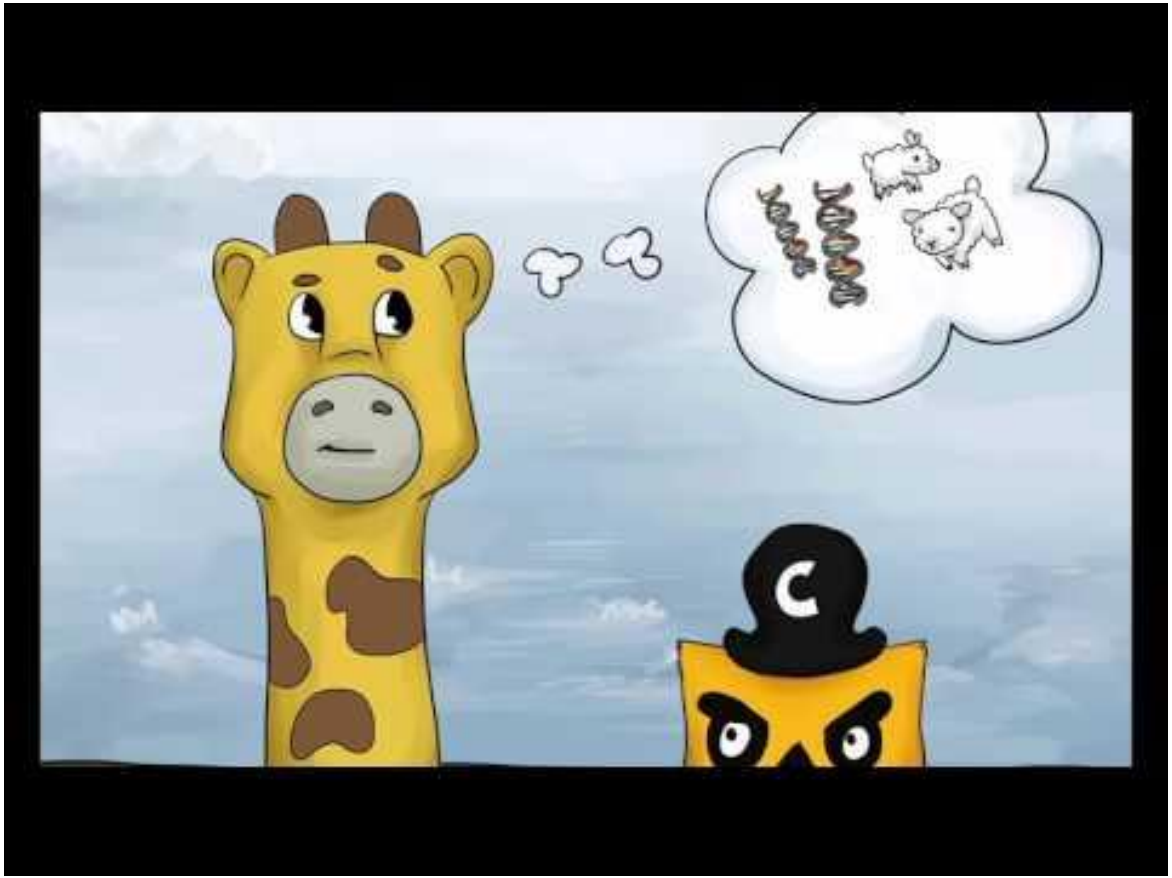
10. Automate build process using CI/CD

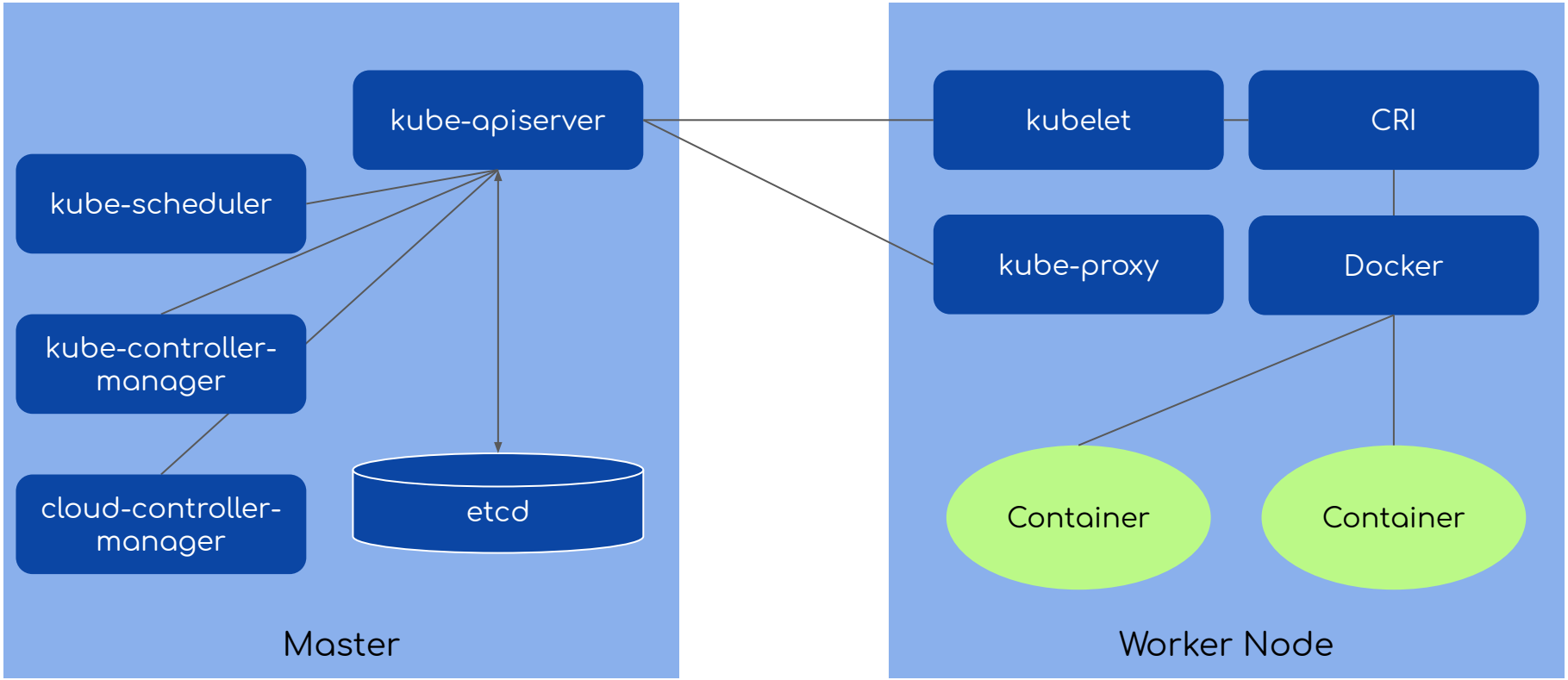
Kubernetes

Minikube overview



What is Kubernetes?



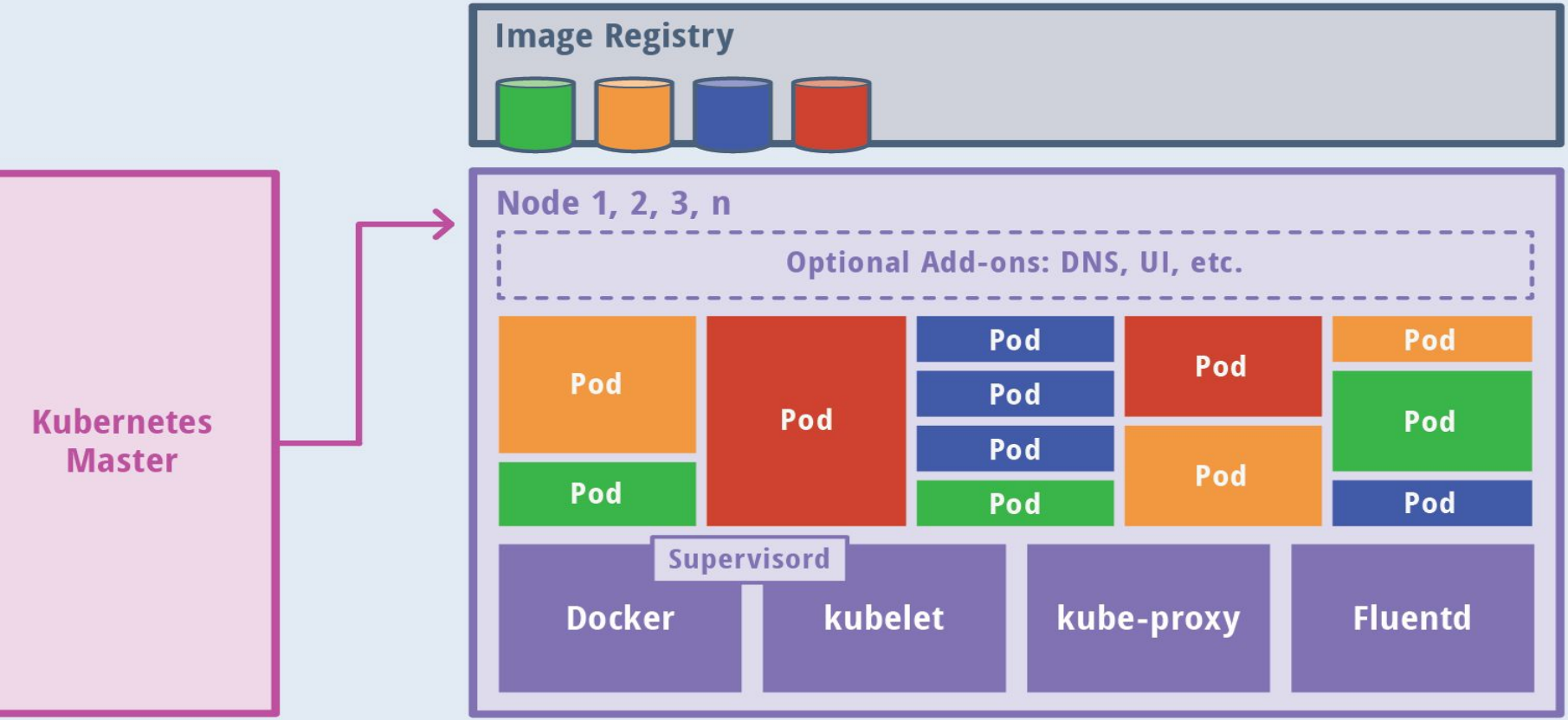


Kubernetes architecture details

Why Kubernetes?

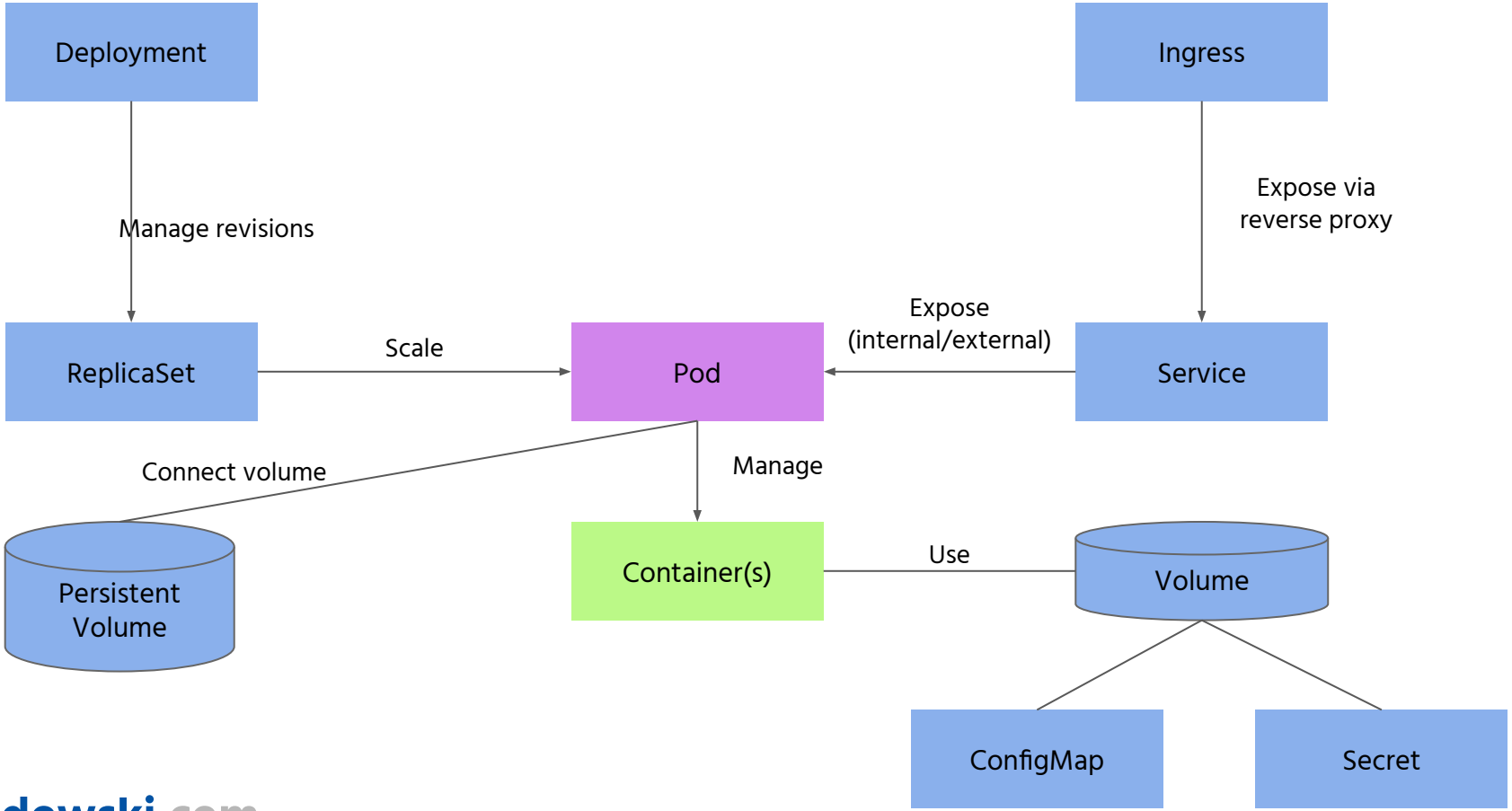
1. Portability
2. Security
3. **Scalability**
4. Speed
5. **Automation**
6. **High Availability**
7. **Extensibility**

Kubernetes Node

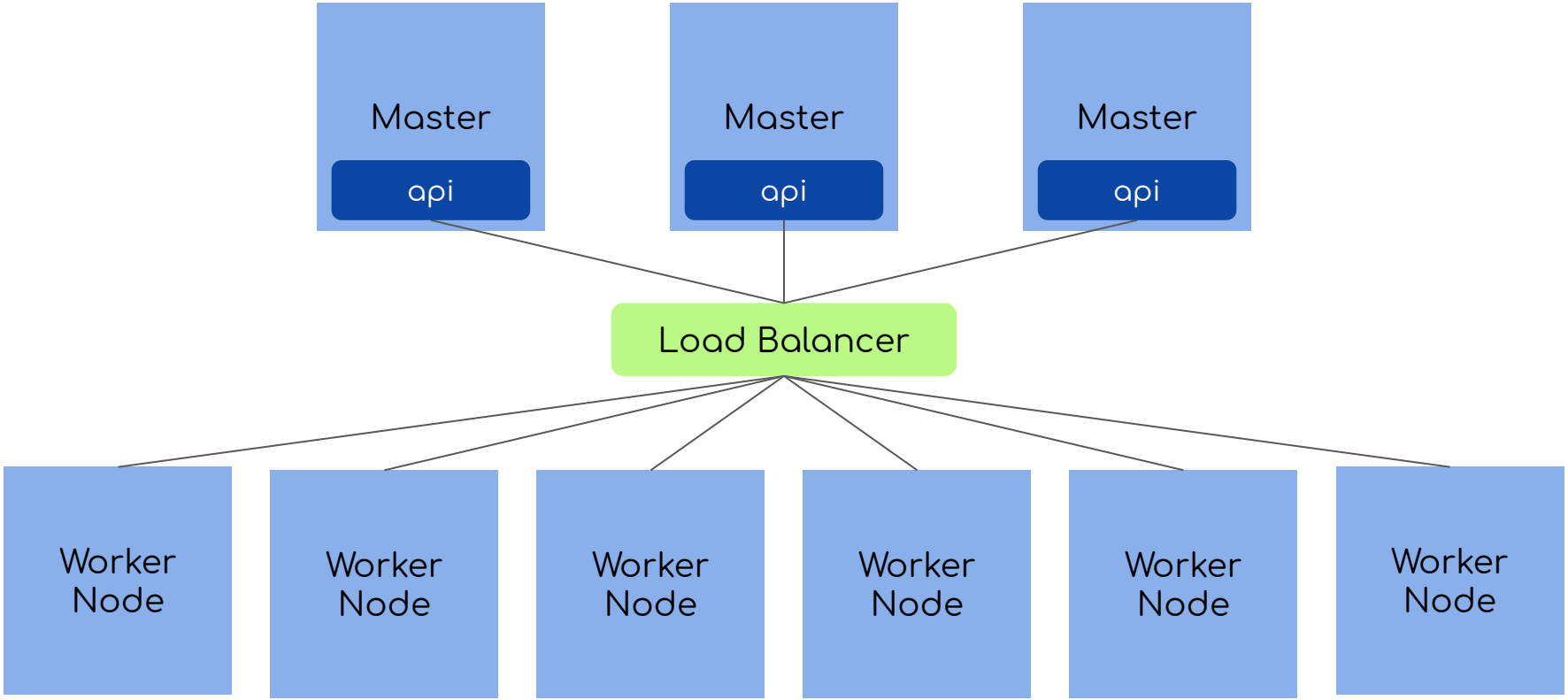


Source: Janakiram MSV

Kubernetes map



Kubernetes networking



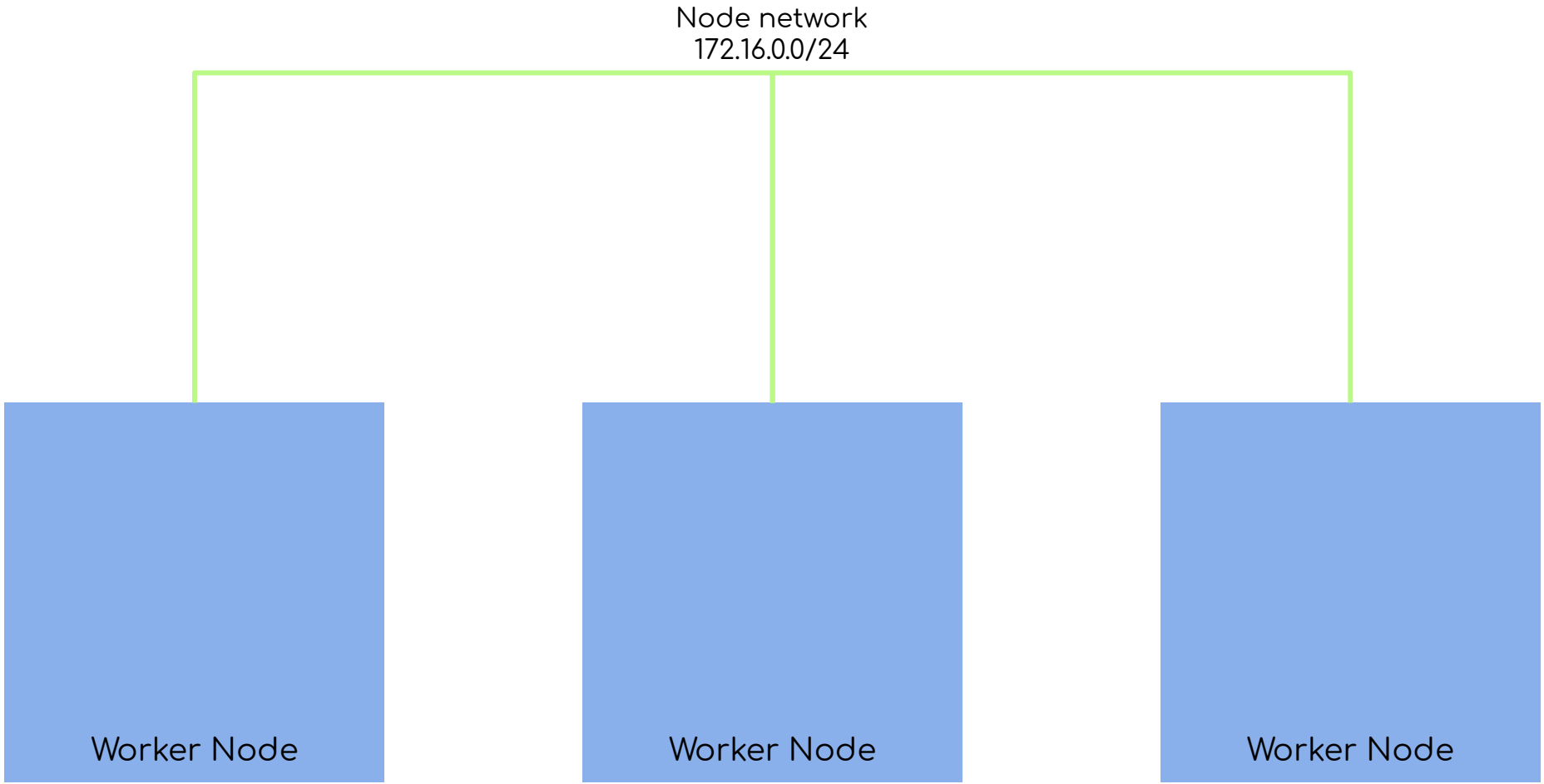
Kubernetes architecture overview

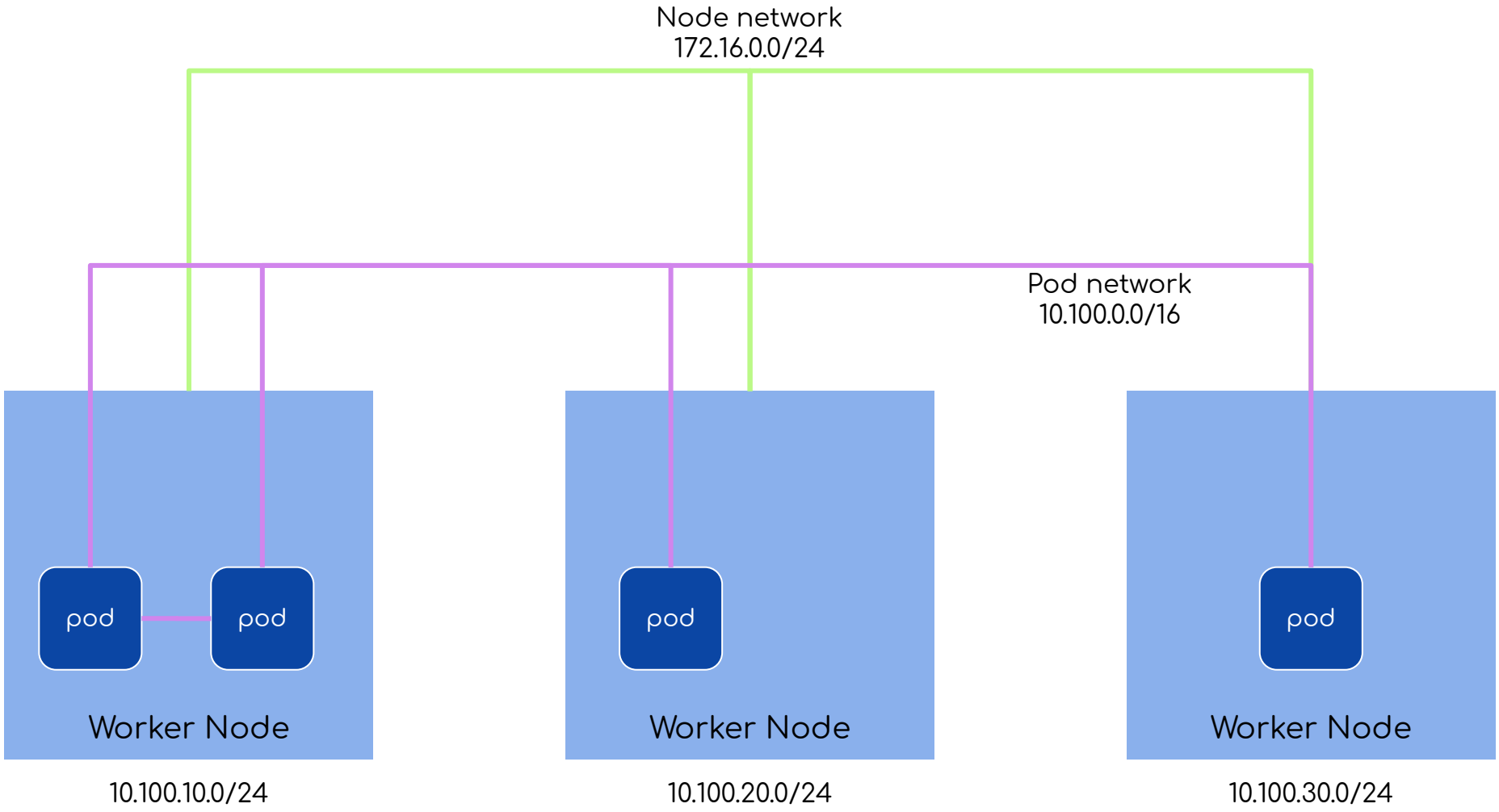
Rules of Kubernetes networking model

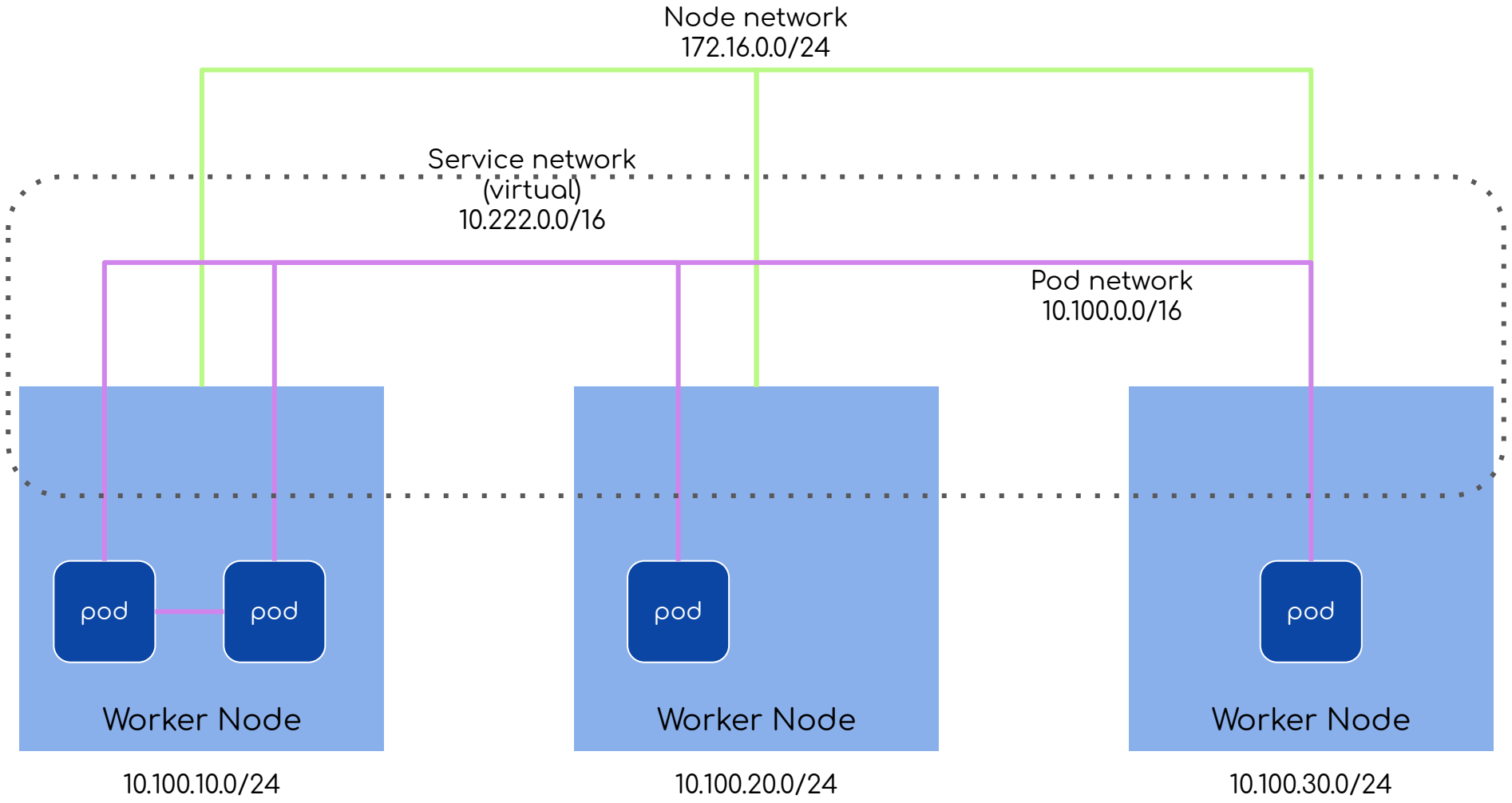
1. All containers can communicate with all other containers without NAT

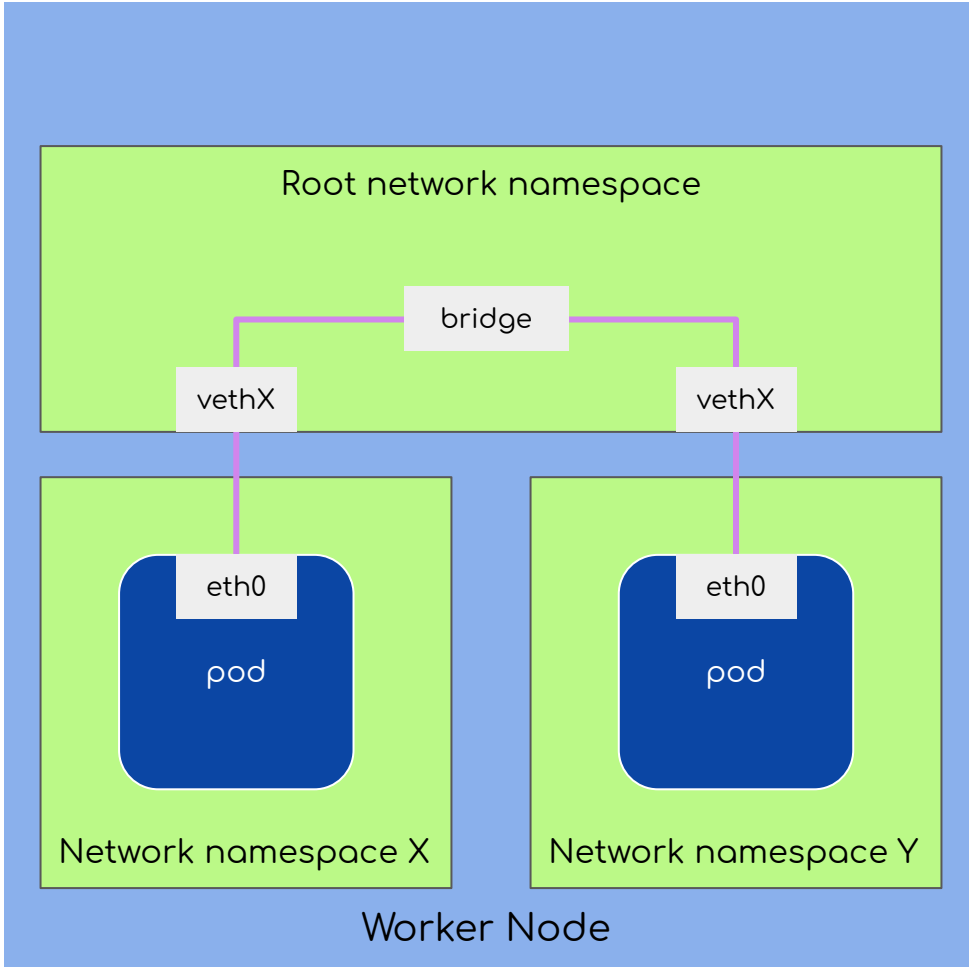
2. All nodes can communicate with all containers (and vice-versa) without NAT

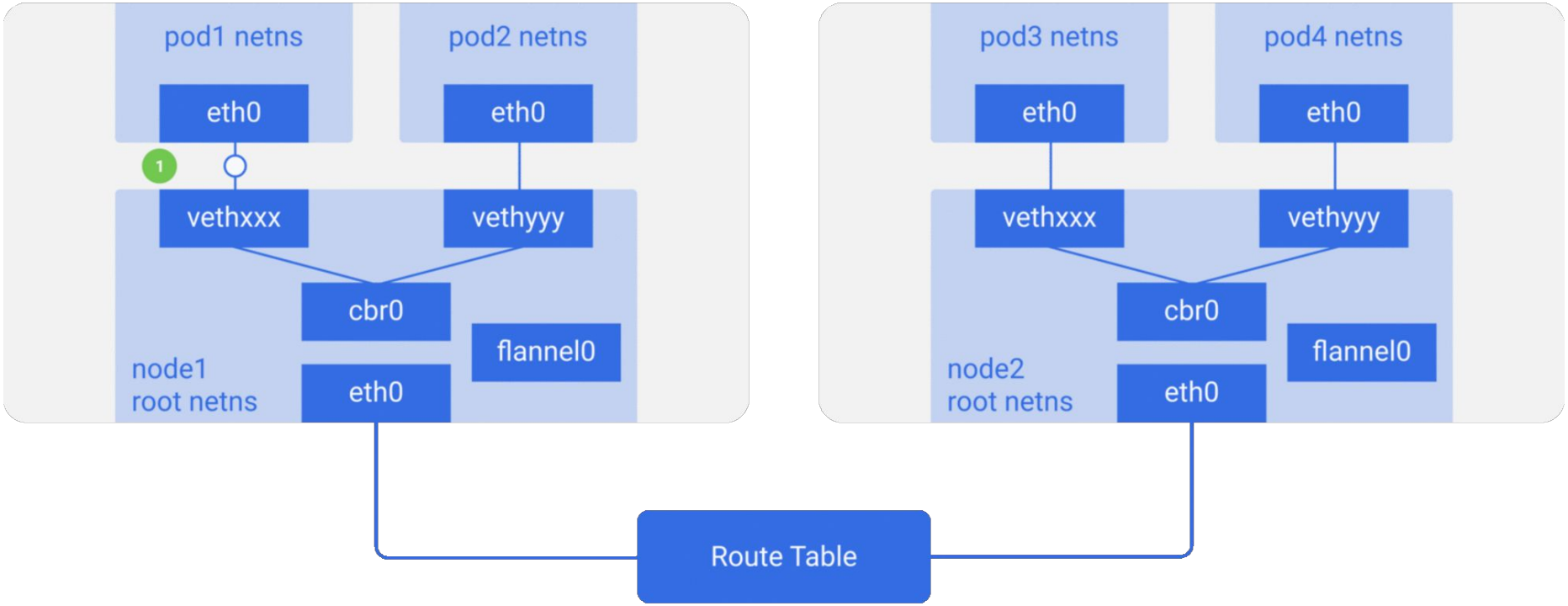
3. The IP that a container sees itself as is the same IP that others see it as


Node network
172.16.0.0/24
Worker Node
Worker Node
Worker Node


Node network
172.16.0.0/24
Pod network
10.100.0.0/16
pod
pod
pod
pod
Worker Node
Worker Node
Worker Node
10.100.10.0/24
10.100.20.0/24
10.100.30.0/24

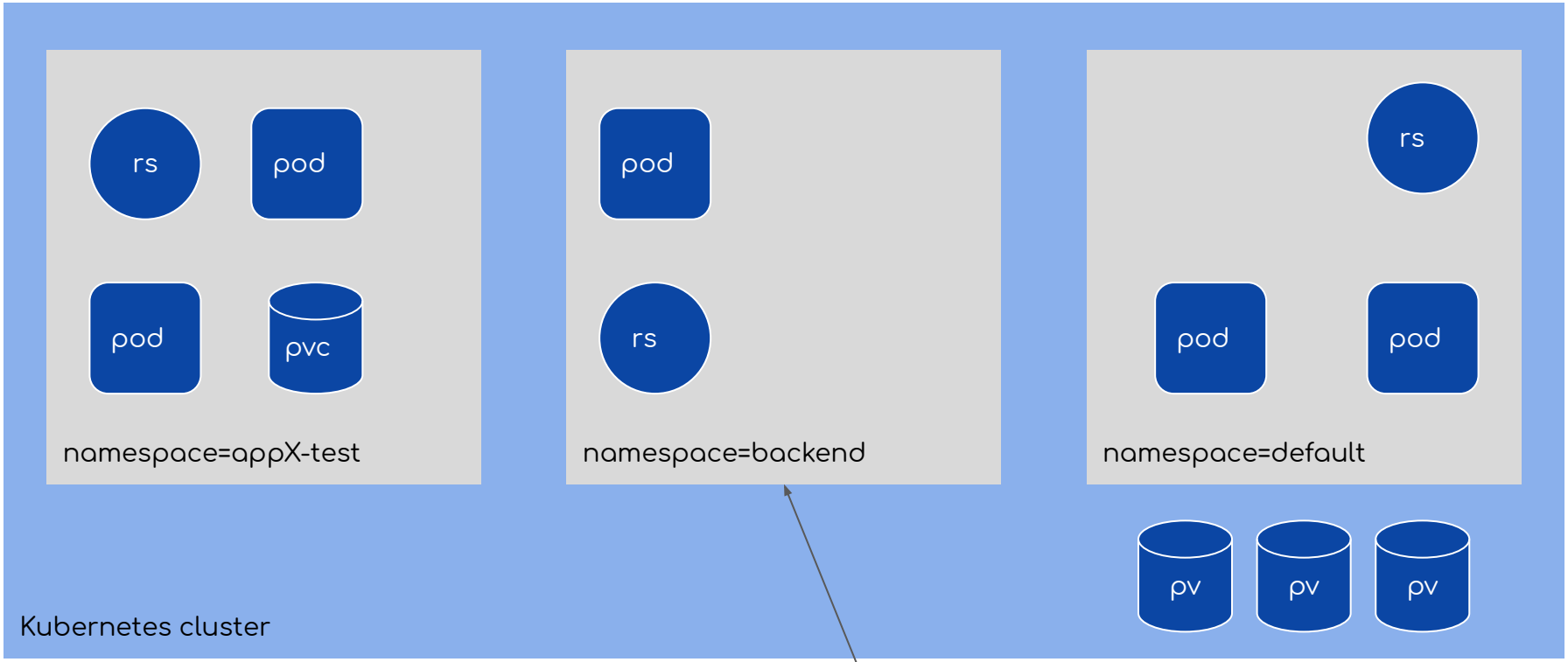

Node network
172.16.0.0/24
Service network
(virtual)
10.222.0.0/16
Pod network
10.100.0.0/16
pod
pod
pod
pod
Worker Node
Worker Node
Worker Node
10.100.10.0/24
10.100.20.0/24
10.100.30.0/24


Root network namespace
bridge
vethX
vethX
eth0
pod
eth0
pod
Network namespace X
Network namespace Y
Worker Node

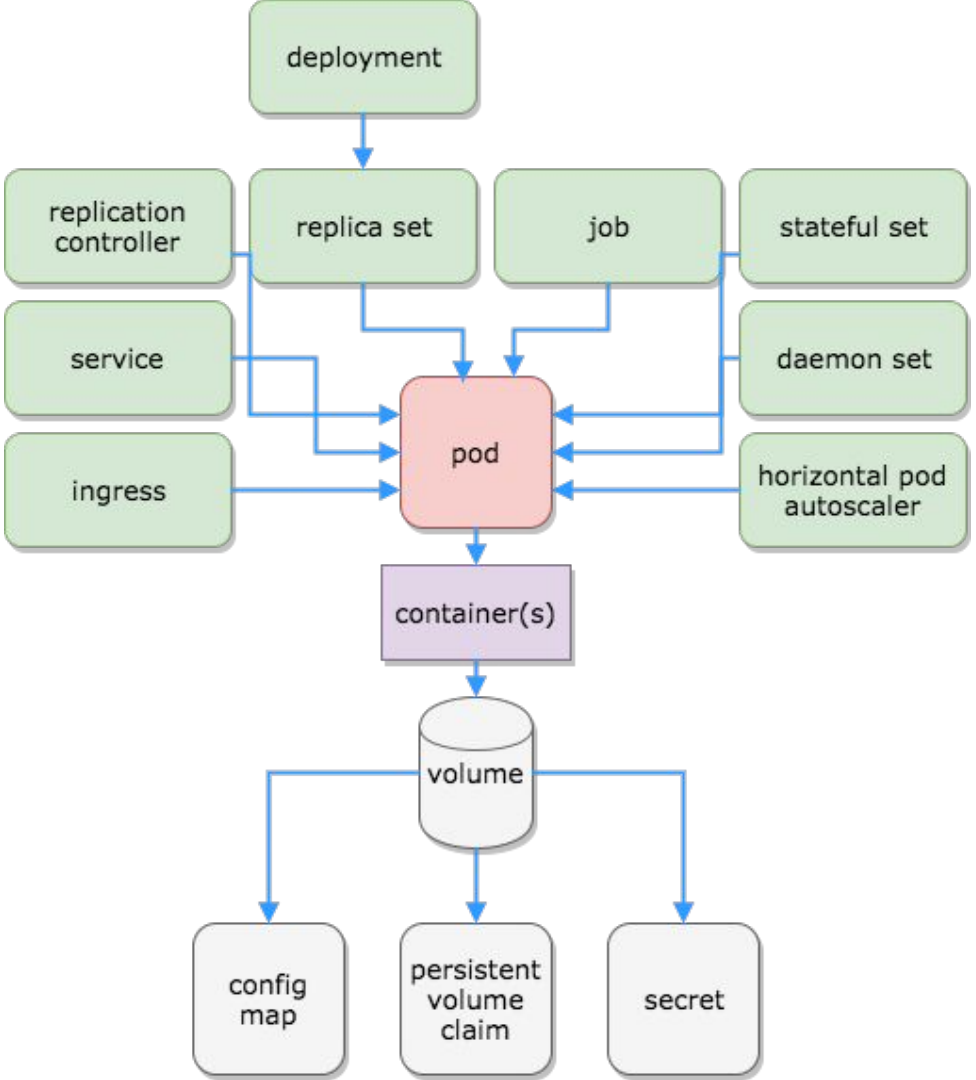


Source: https://medium.com/@ApsOps/an-illustrated-guide-to-kubernetes-networking-part-2-13fdc6c4e24c

Kubernetes objects and namespaces

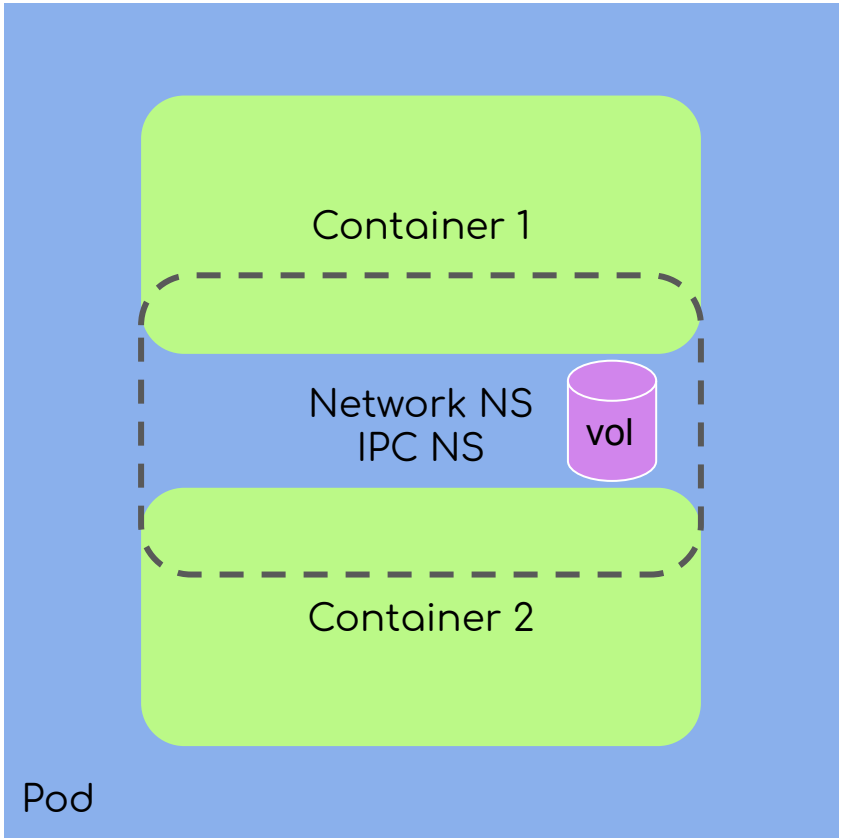

rs
pod
pod
pvc
namespace=appX-test
pod
rs
namespace=backend
rs
pod
pod
namespace=default
pv
pv
pv
Kubernetes cluster
kubectl
~/.kube/config:
- context:
- cluster: ...
- namespace: ...
- user: ...

Namespaces

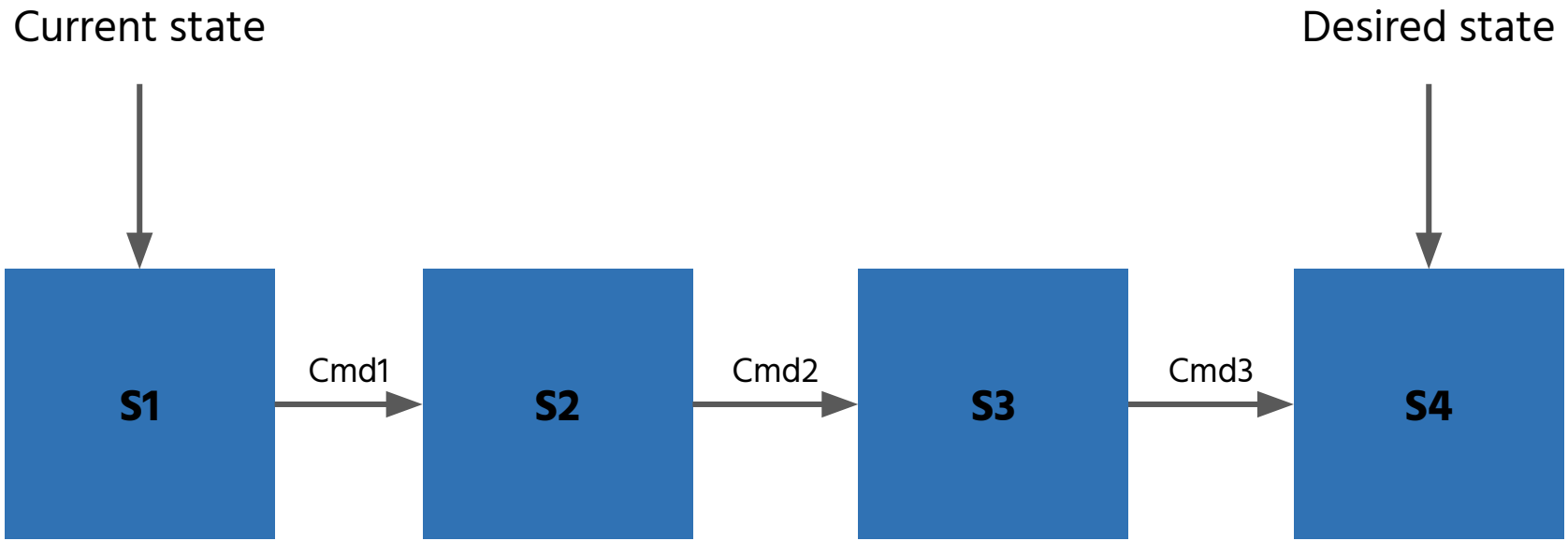


Kubernetes objects

Pod

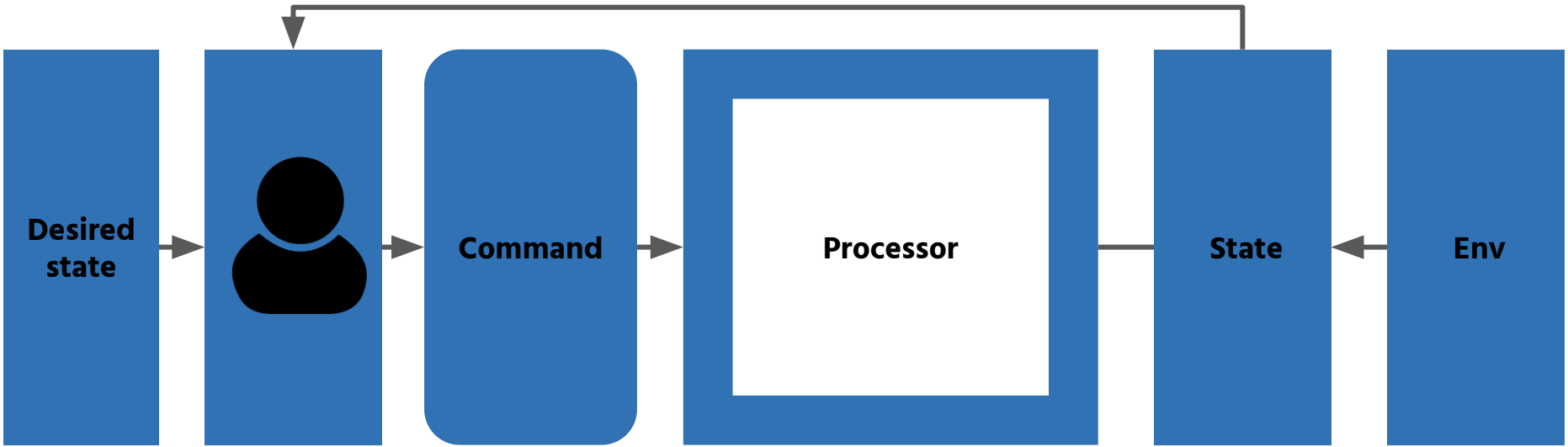


Imperative vs. declarative



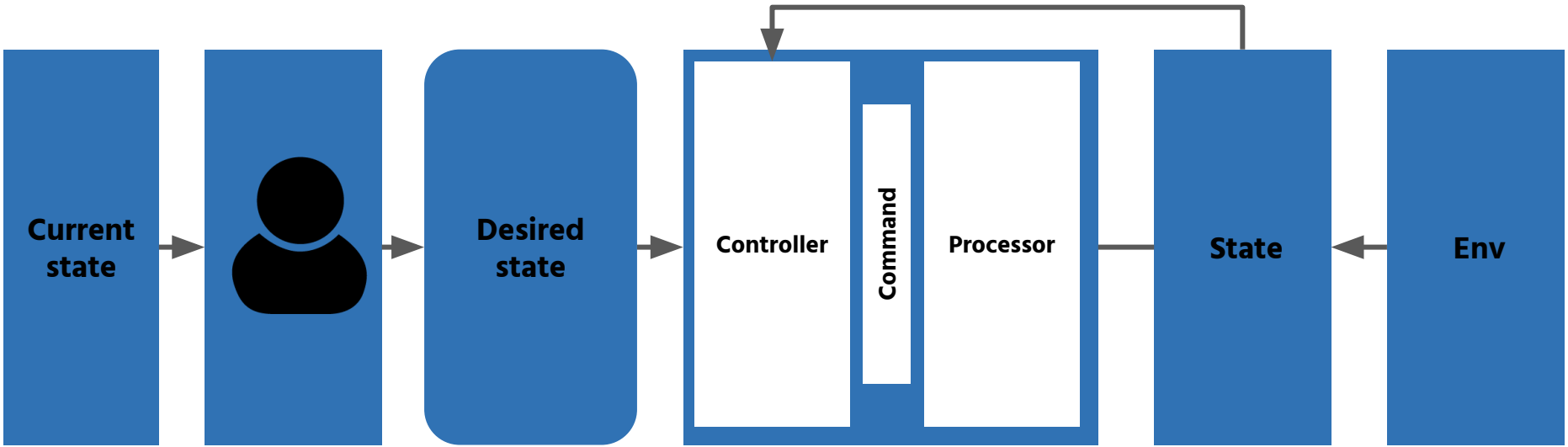
Insipred by: https://medium.com/@dominik.tornow/imperative-vs-declarative-8abc7dcae82e

Imperative systems



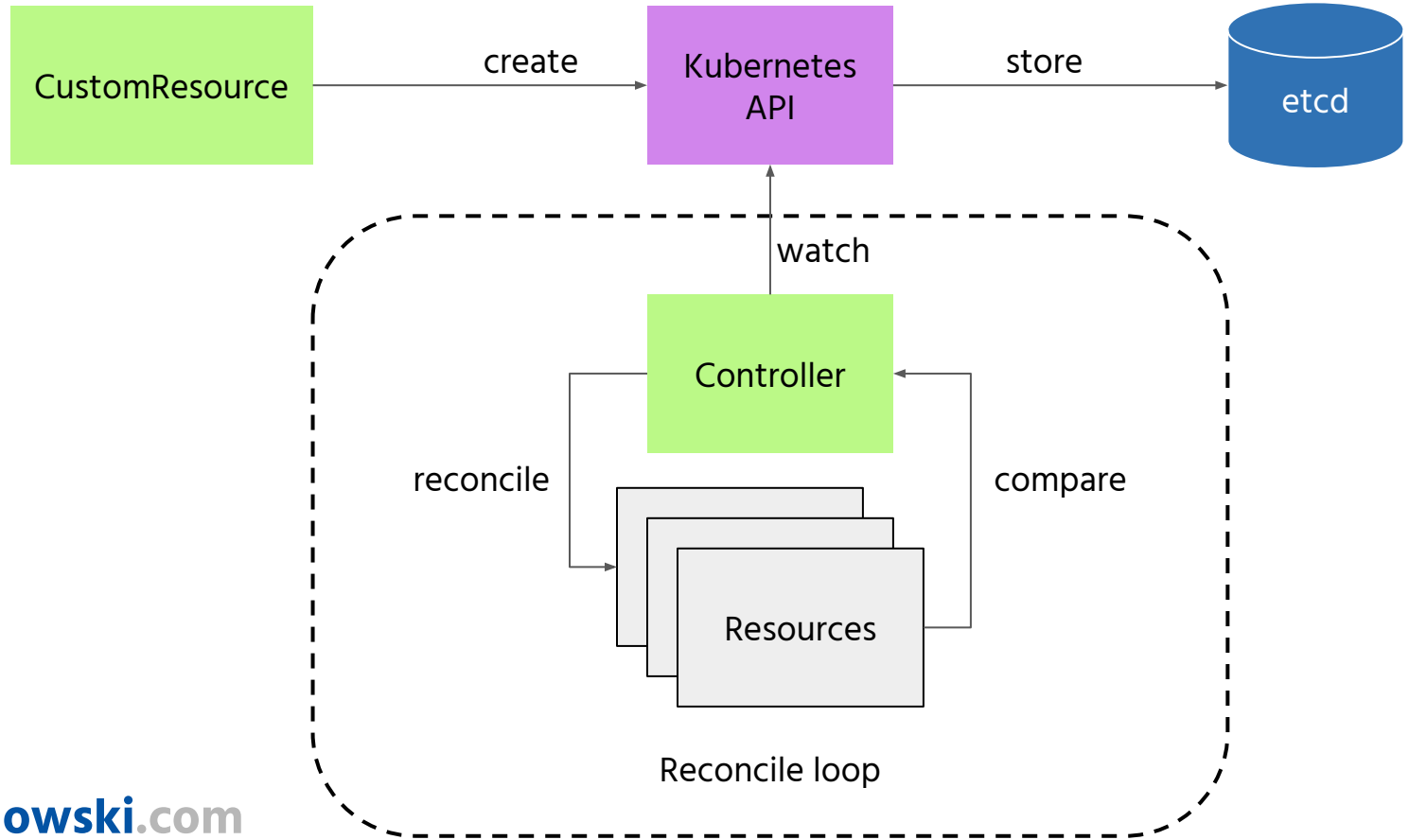
Insipred by: https://medium.com/@dominik.tornow/imperative-vs-declarative-8abc7dcae82e

Declarative systems



Insipred by: https://medium.com/@dominik.tornow/imperative-vs-declarative-8abc7dcae82e

Kubernetes controller with reconcile loop



Yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: fussy
spec:
  containers:
  - image: nginx
    name: fussy
```

Yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: fussy
spec:
  containers:
  - image: nginx
    name: fussy
```

REQUIRED

- different levels of stability and support
  - List with "***kubectl api-versions***"
  - alpha, beta, stable
- v1 = core group

Yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: fussy
spec:
  containers:
  - image: nginx
    name: fussy
```

REQUIRED

Yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: fussy
spec:
  containers:
  - image: nginx
    name: fussy
```

REQUIRED

Valuese required: **name**

Optional: namespace

Yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: fussy
  labels:
    app: fussy
spec:
  containers:
  - image: nginx
    name: fussy
```

OPTIONAL

Key-Value pairs

Yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: fussy
spec:
  containers:
  - image: nginx
    name: fussy
```

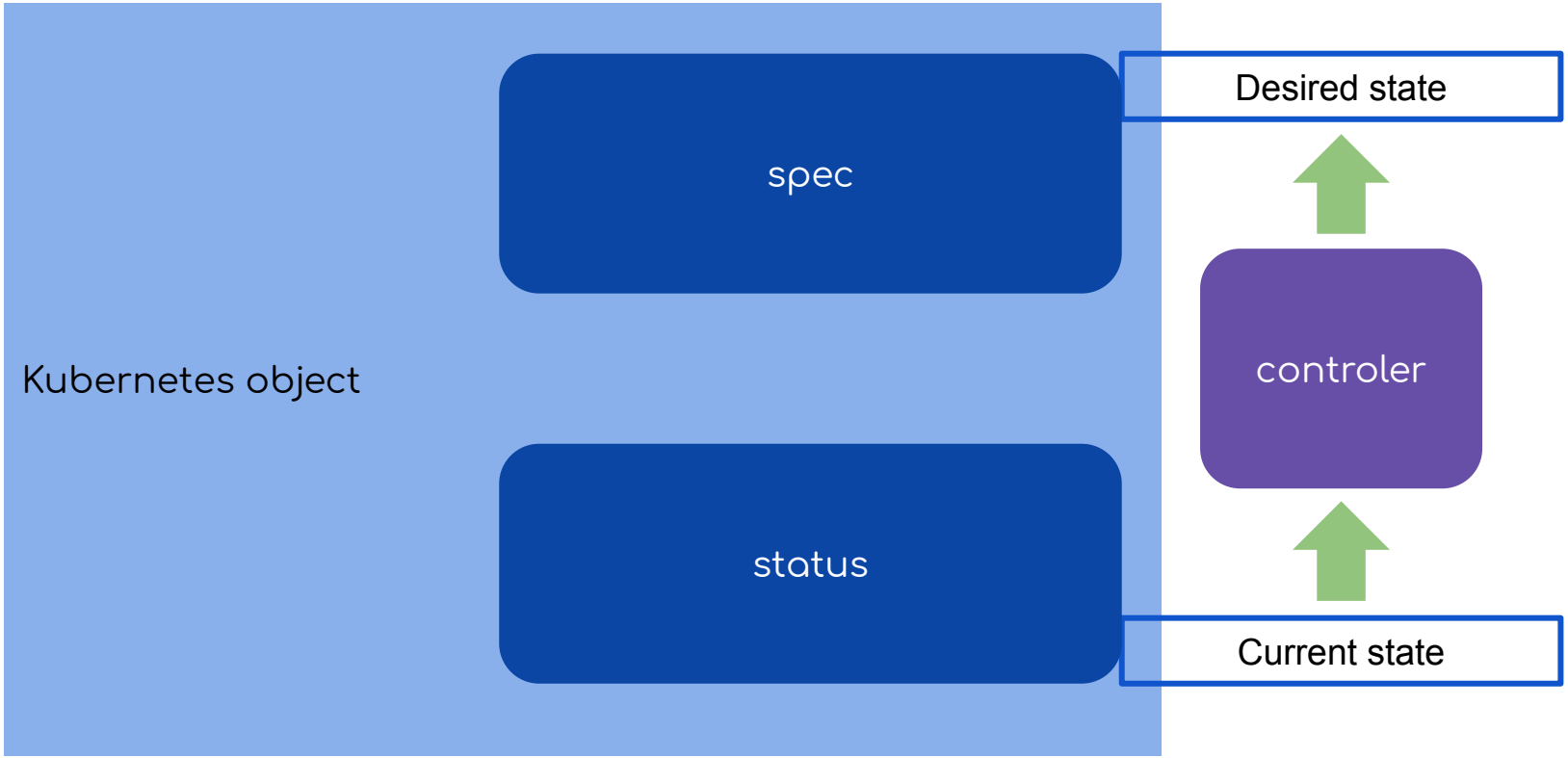
OPTIONAL

Yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: fussy
spec:
  containers:
  - image: nginx
    name: fussy
```

OPTIONAL

Depends on object kind



Declarative nature of objects

Labels vs. Annotations

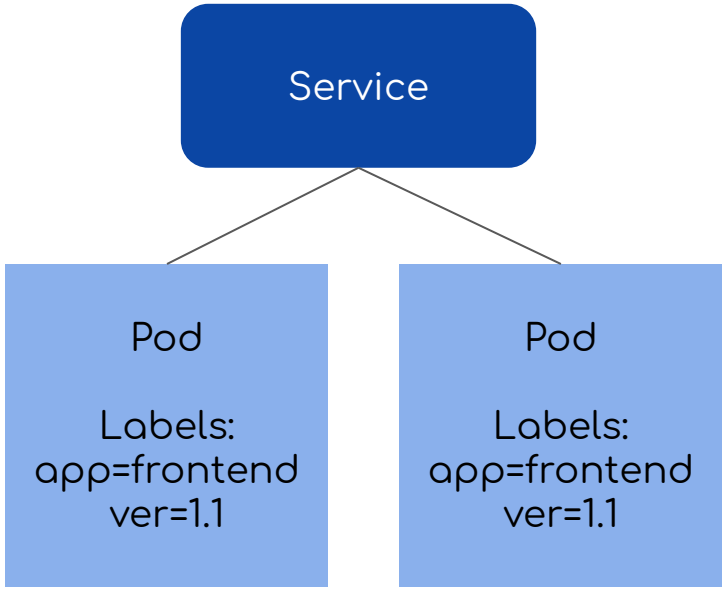
Labels

- Used to organize and select subsets of objects
- Key: Value
- Used internally and by clients

Annotations

- Used for non-identyfying, metadata objects
- Structured, unstructured data
- Used by libraries, clients

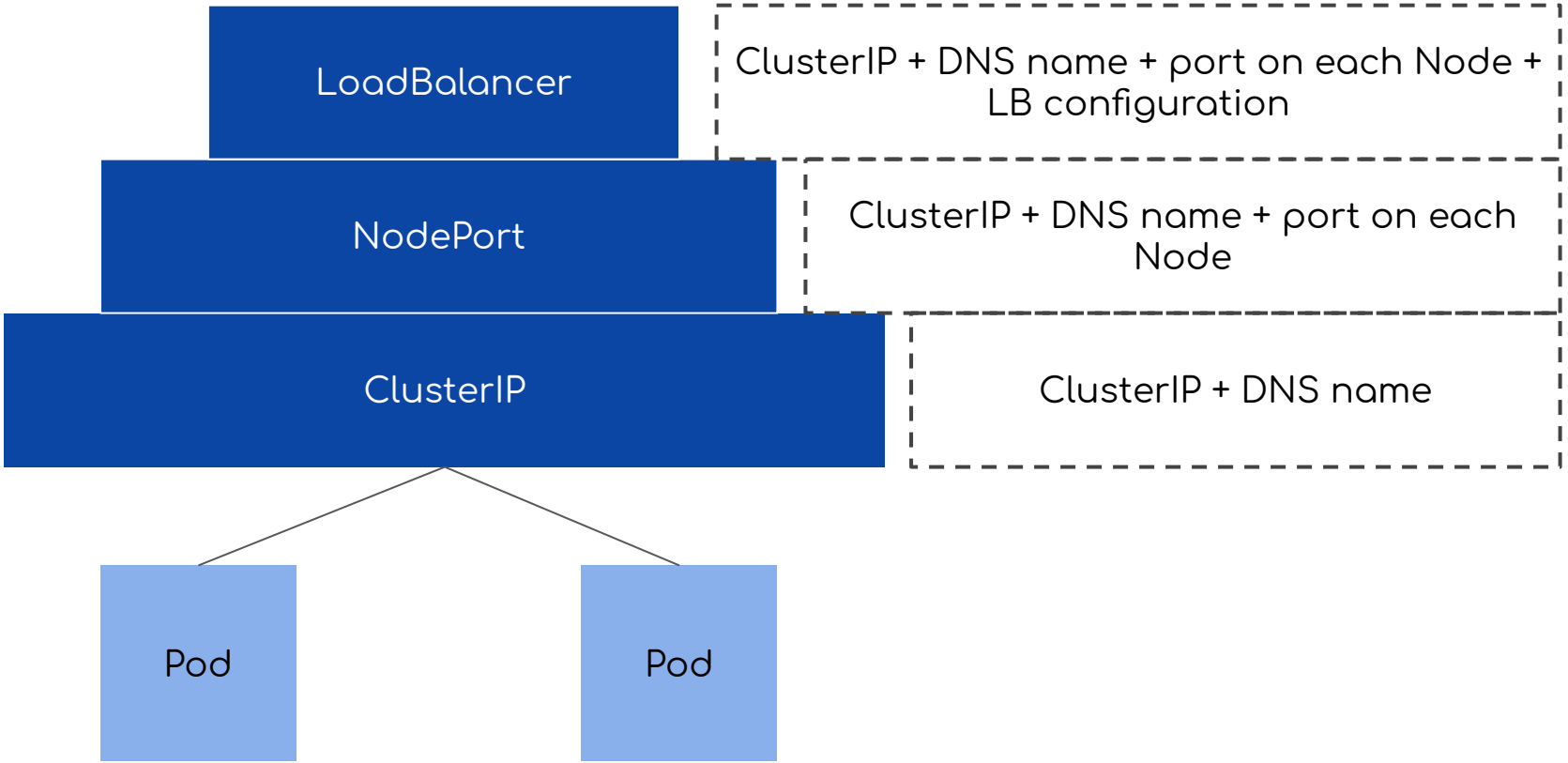
Service



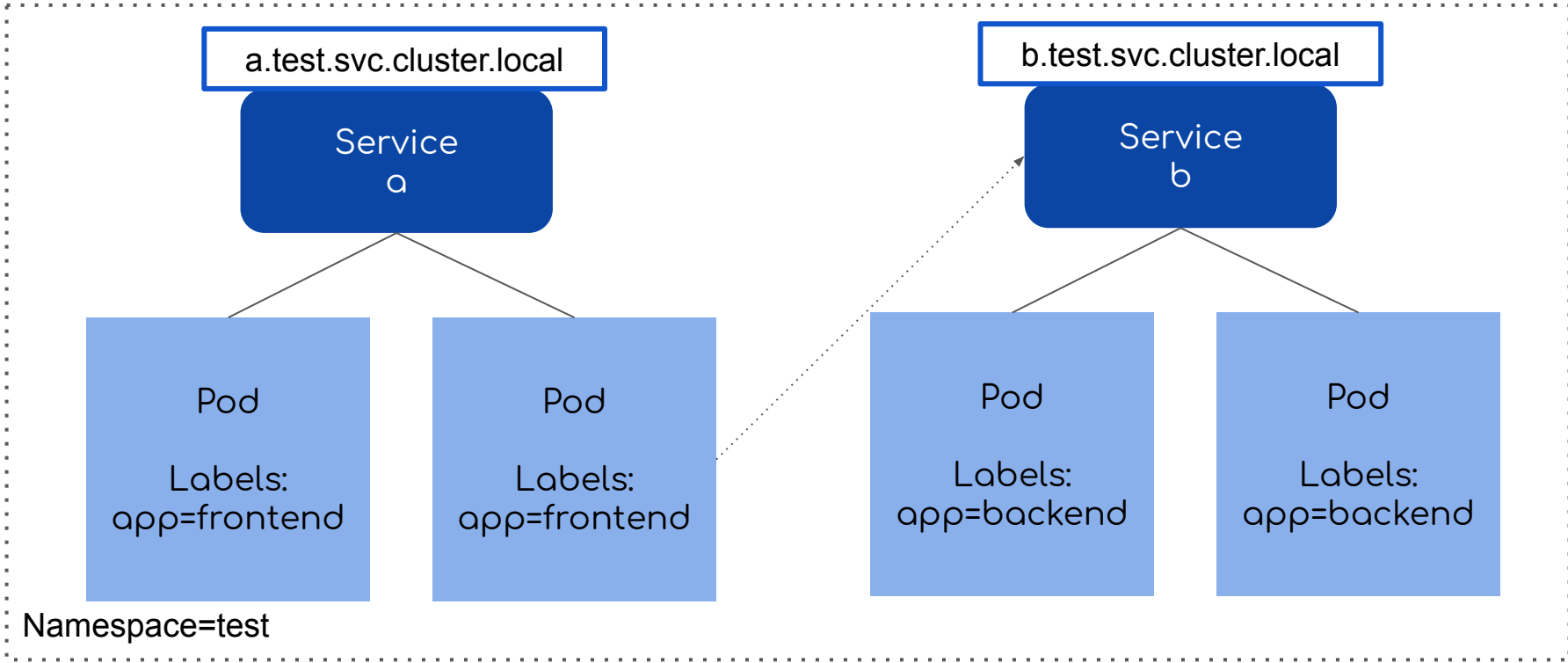
Types

- ClusterIP
- NodePort
- LoadBalancer
- ExternalName

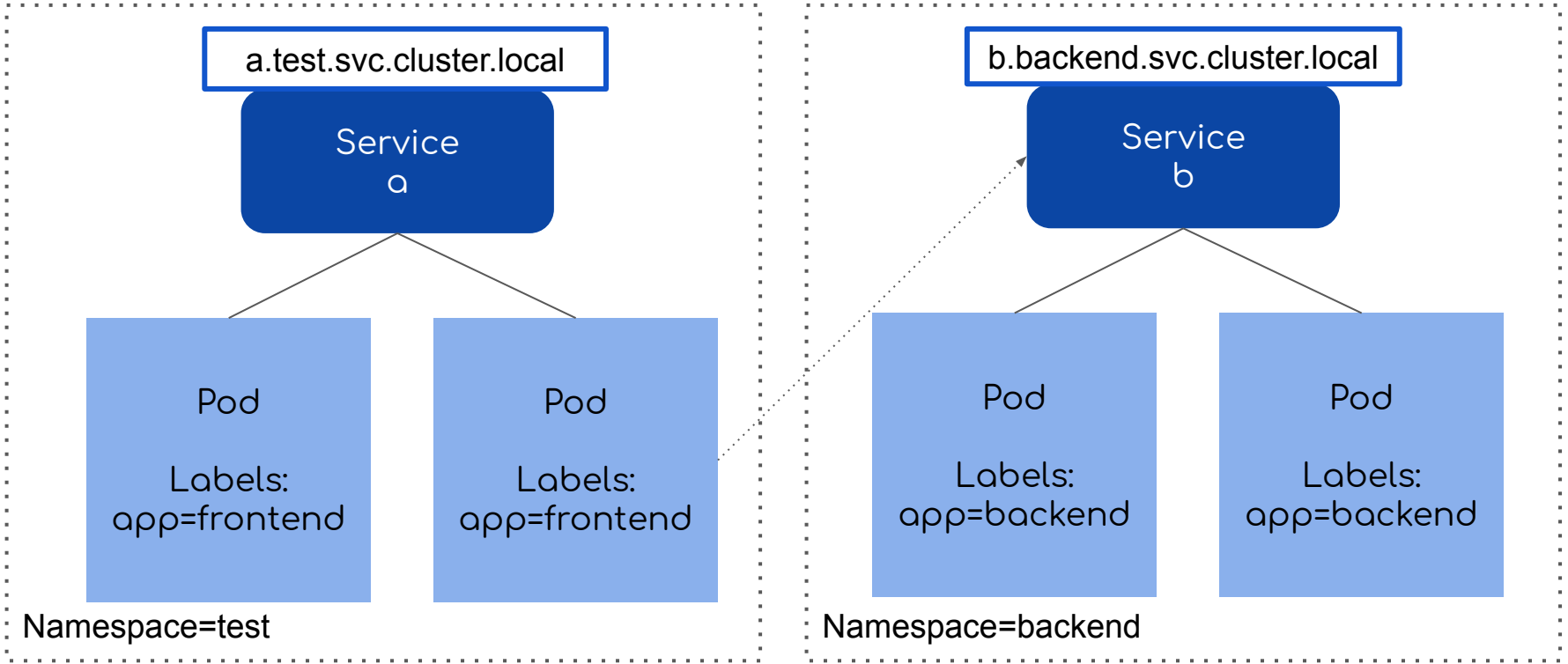
Service types



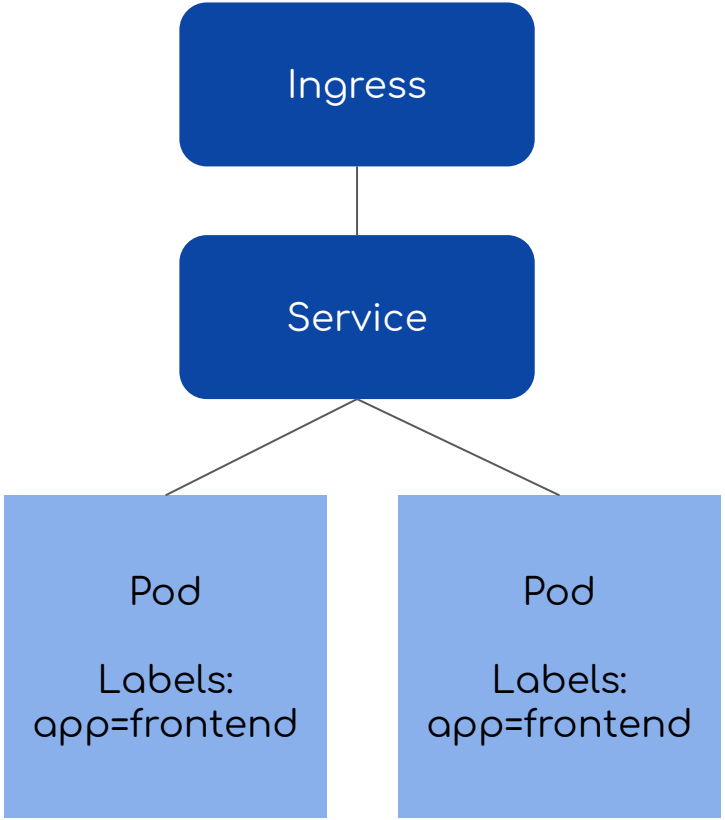
Service Discovery



Service Discovery



Ingress

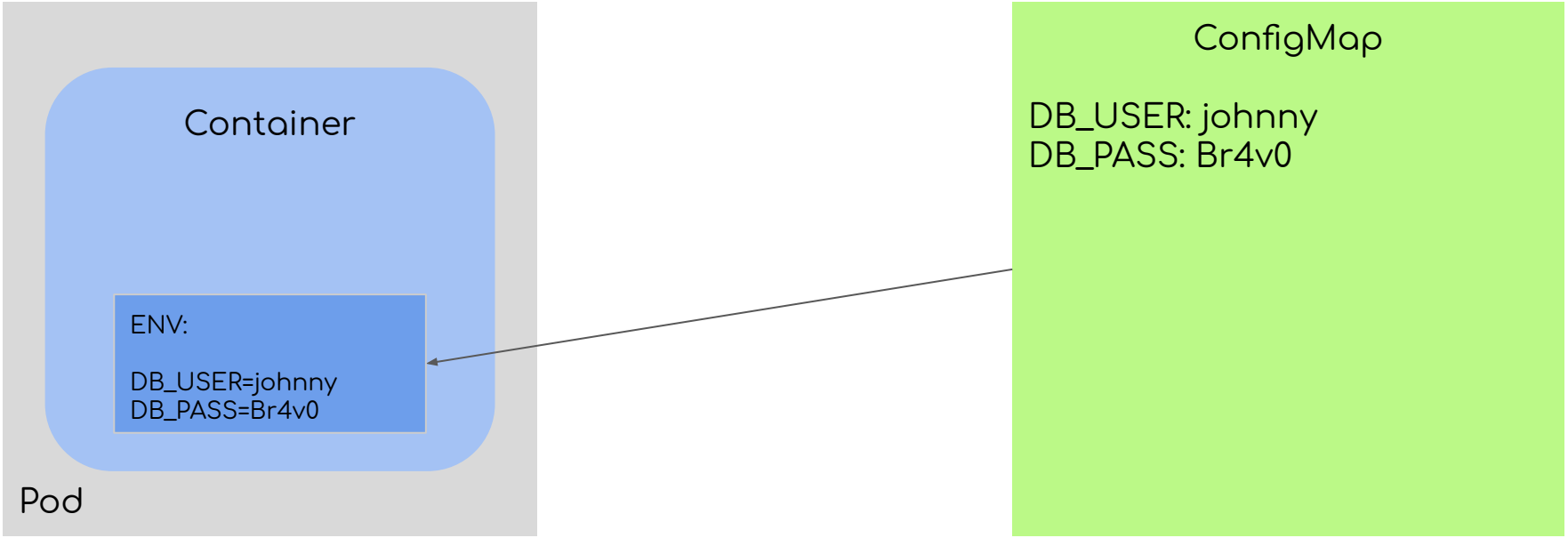


Interface for http(s) traffic

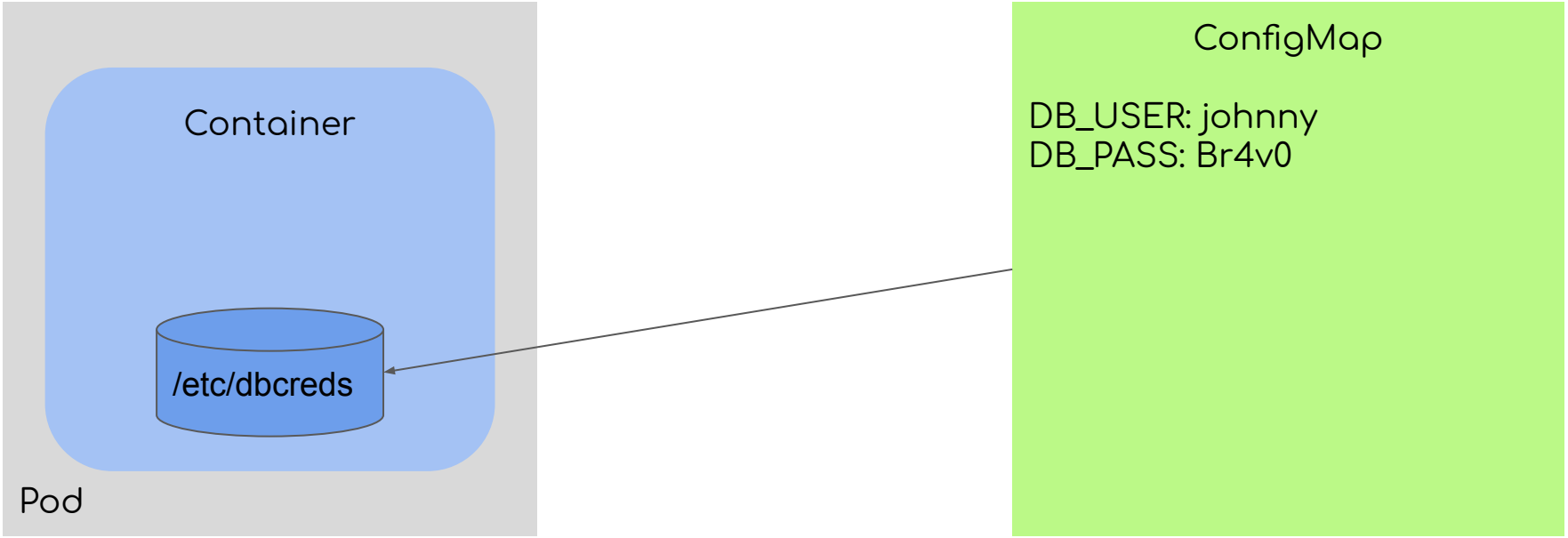
Best known implementation based on Nginx

Controlled fully with declarative code and annotations

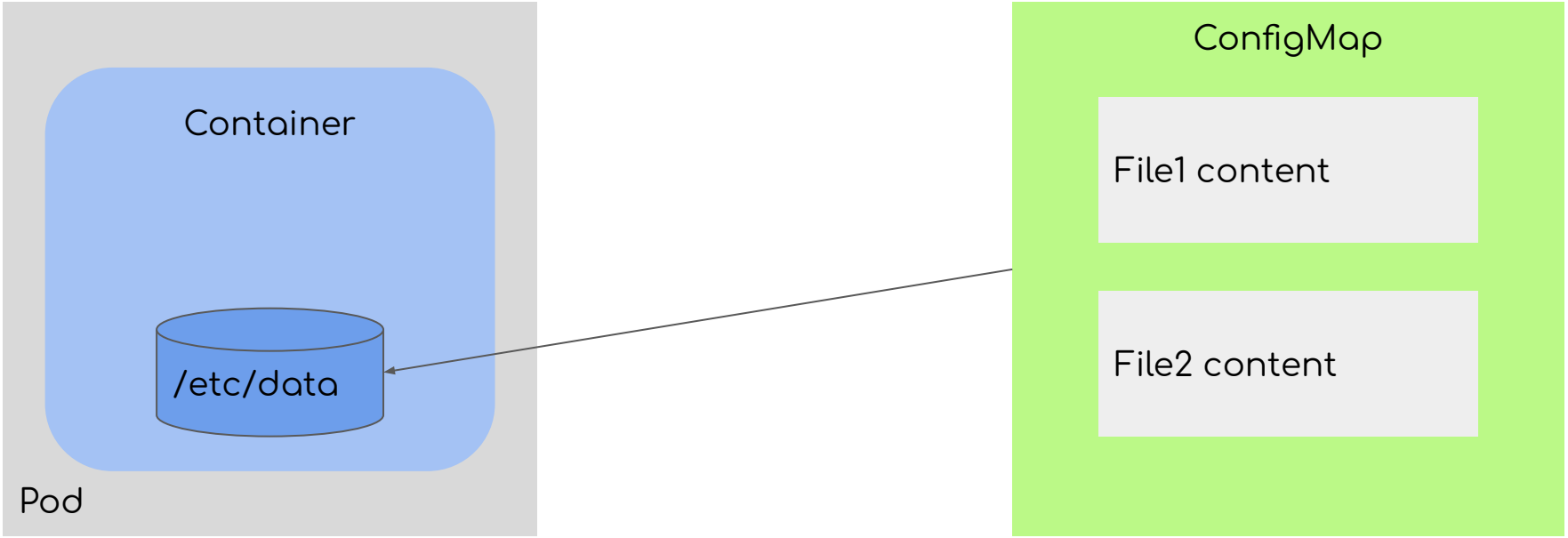
ConfigMap - as environment variables



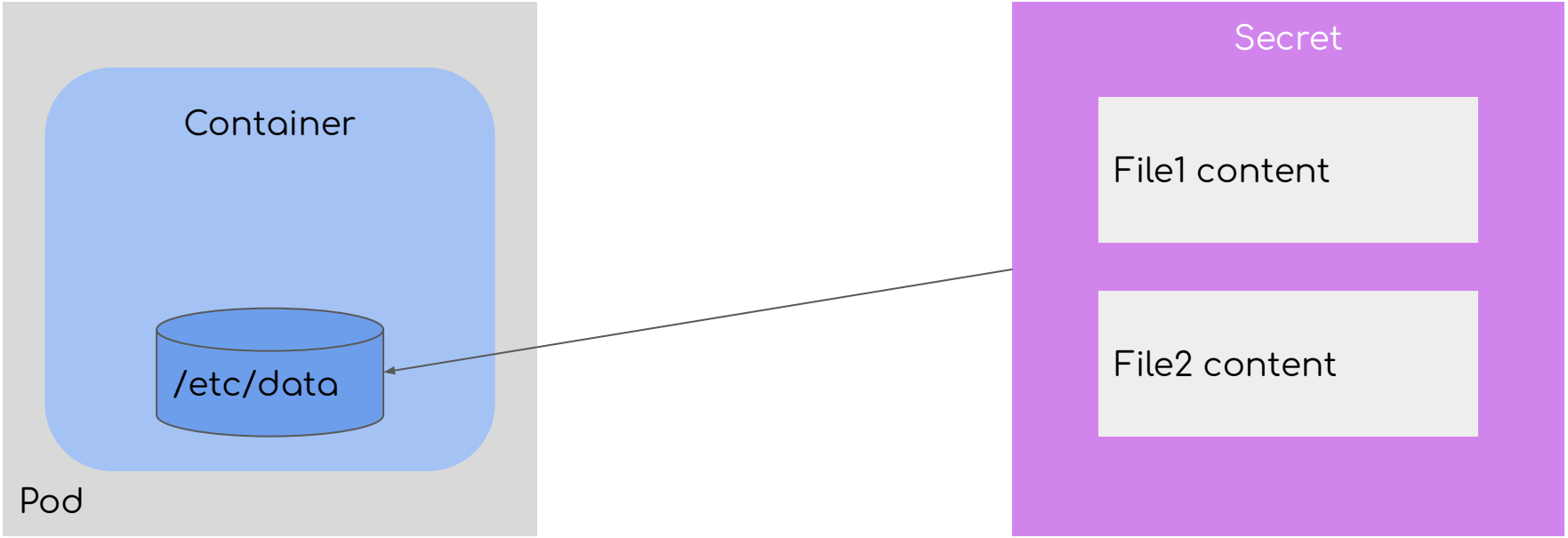
ConfigMap - as volumes



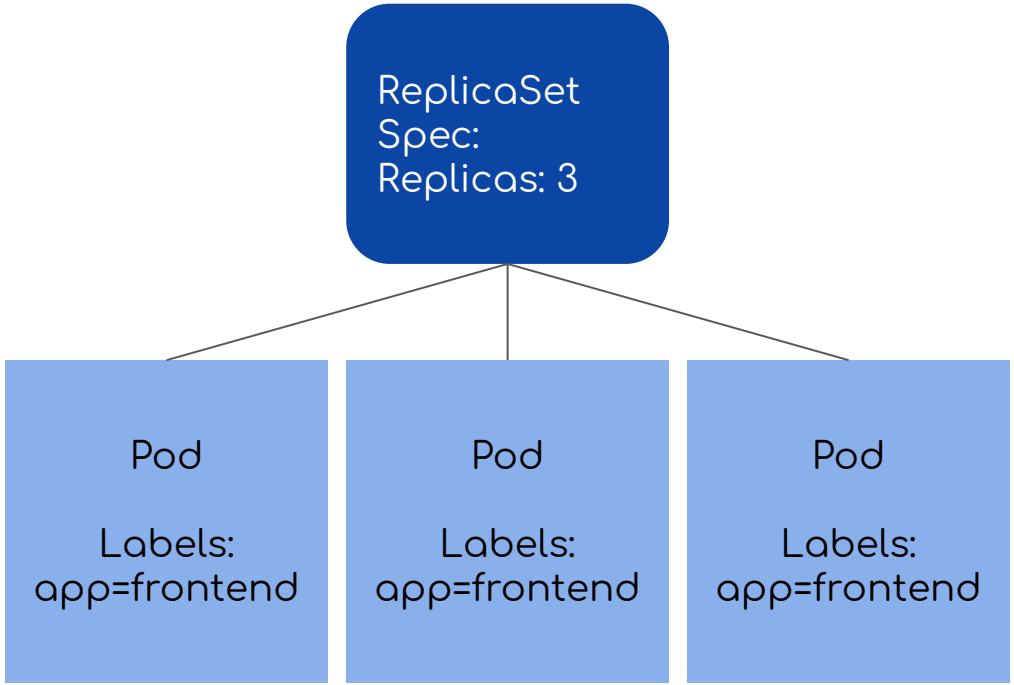
ConfigMap - as volumes



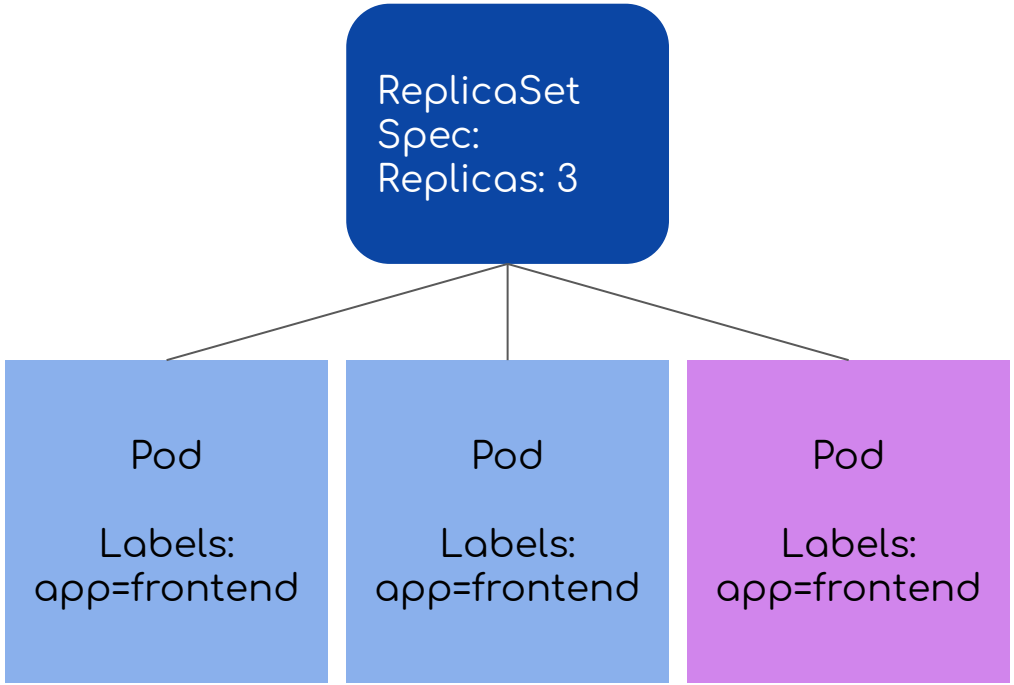
Secrets



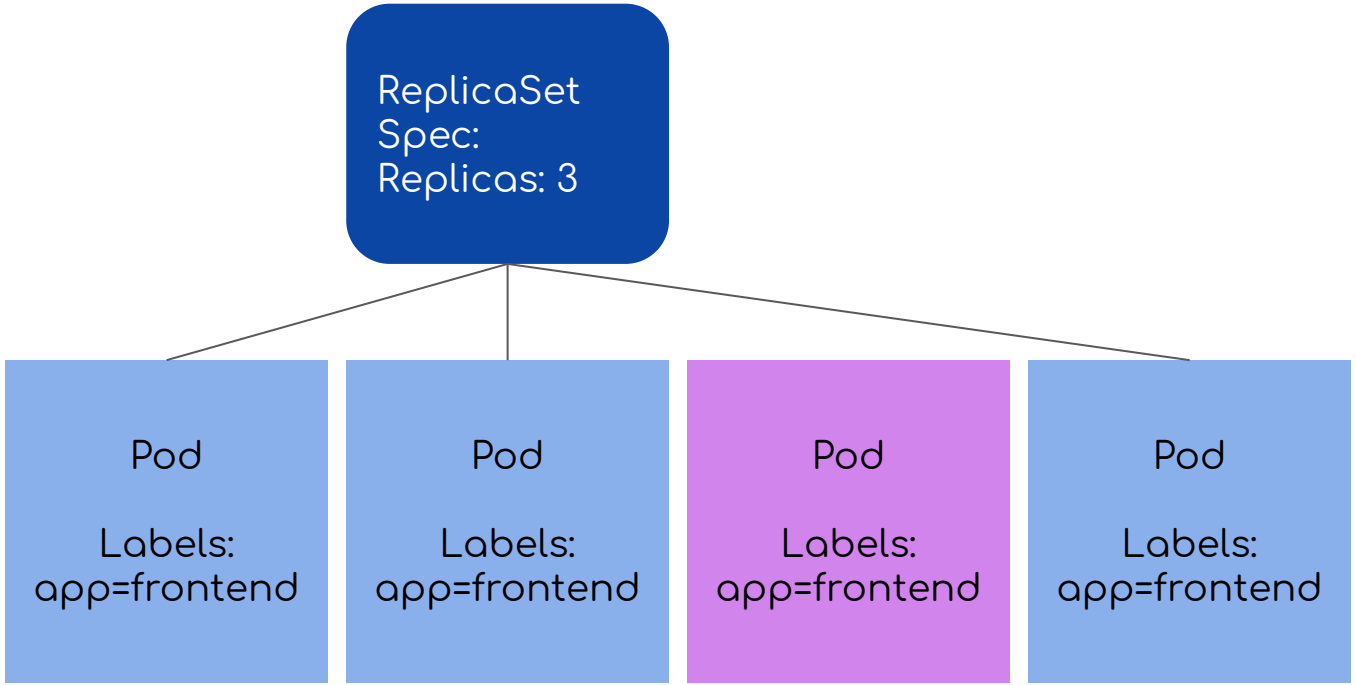
ReplicaSet


ReplicaSet
Spec:
Replicas: 3
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend

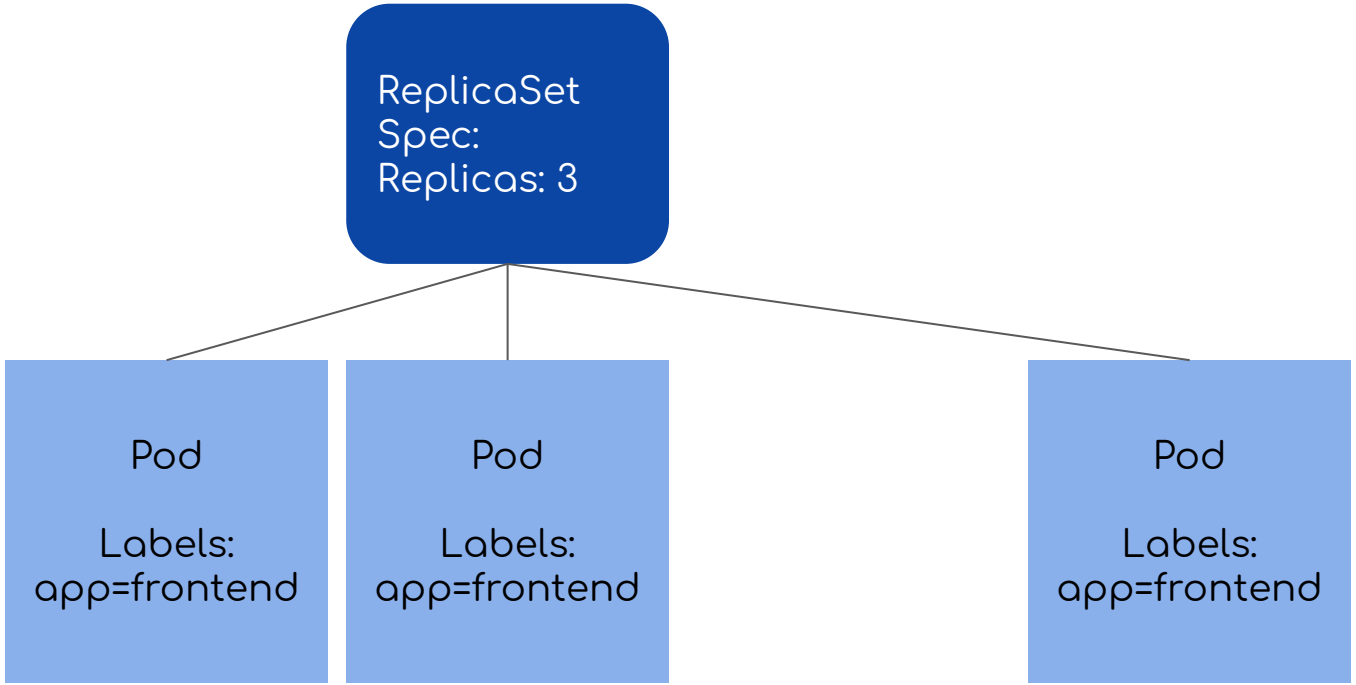
ReplicaSet


ReplicaSet
Spec:
Replicas: 3
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend

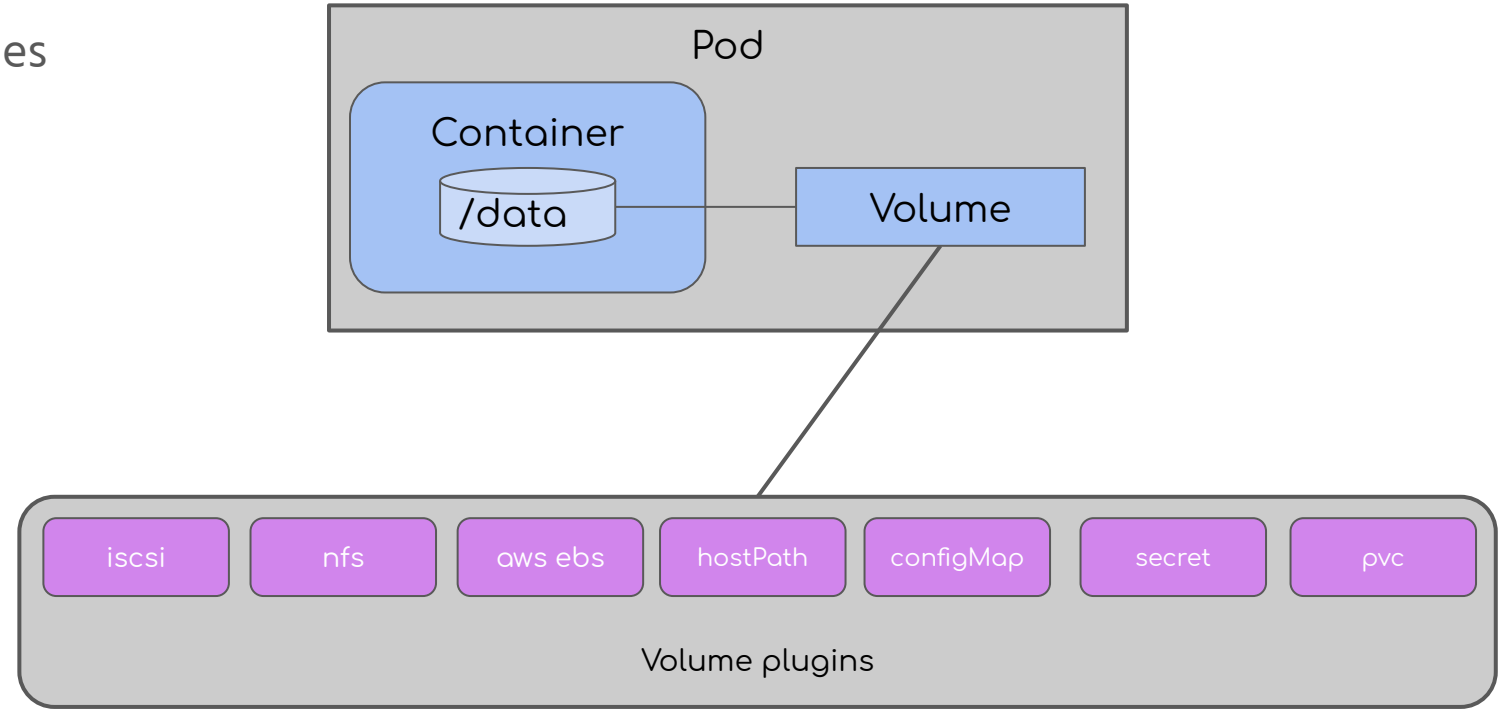
ReplicaSet


ReplicaSet
Spec:
Replicas: 3
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend

ReplicaSet

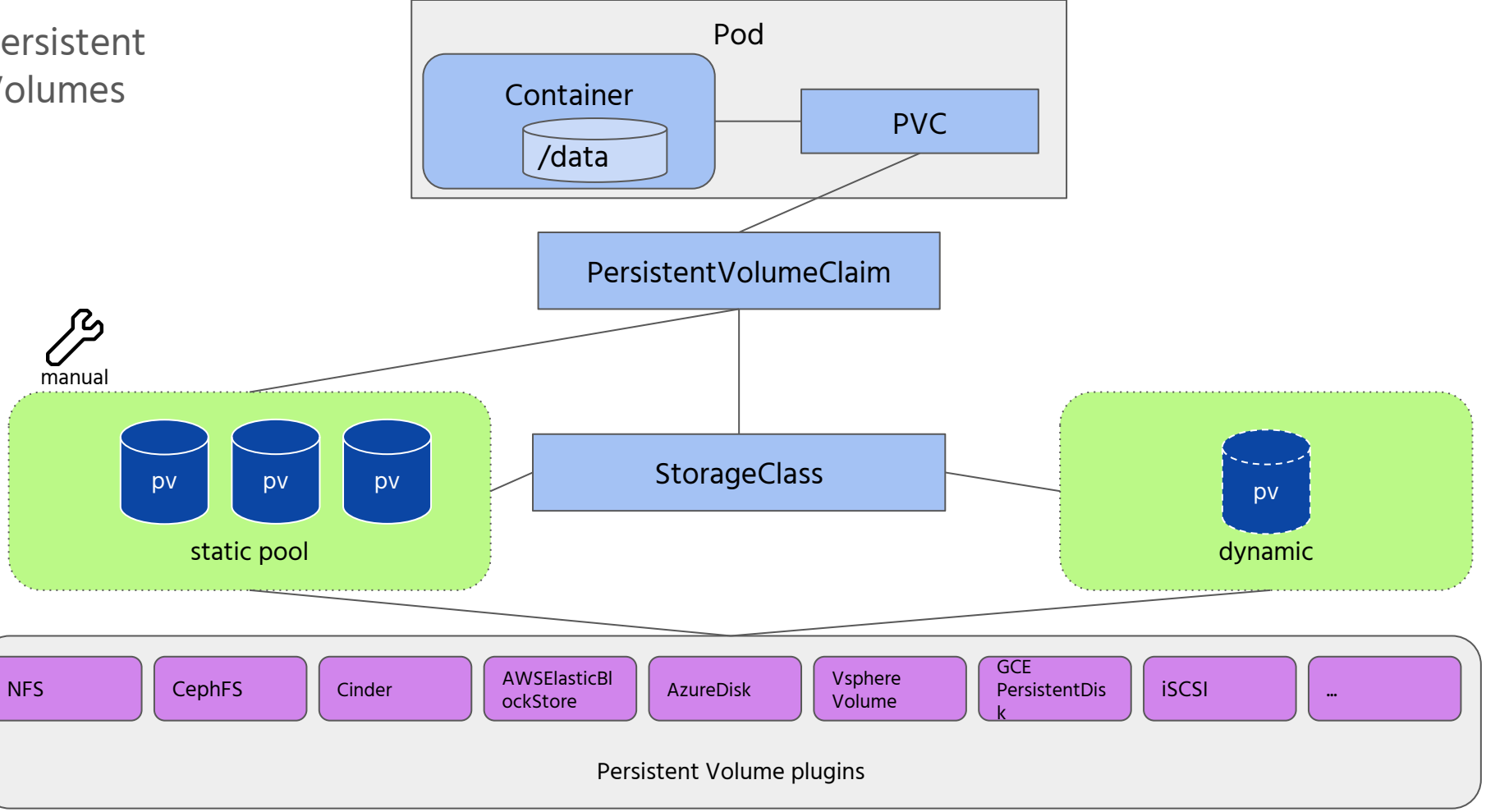

ReplicaSet
Spec:
Replicas: 3
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend
Pod
Labels:
app=frontend

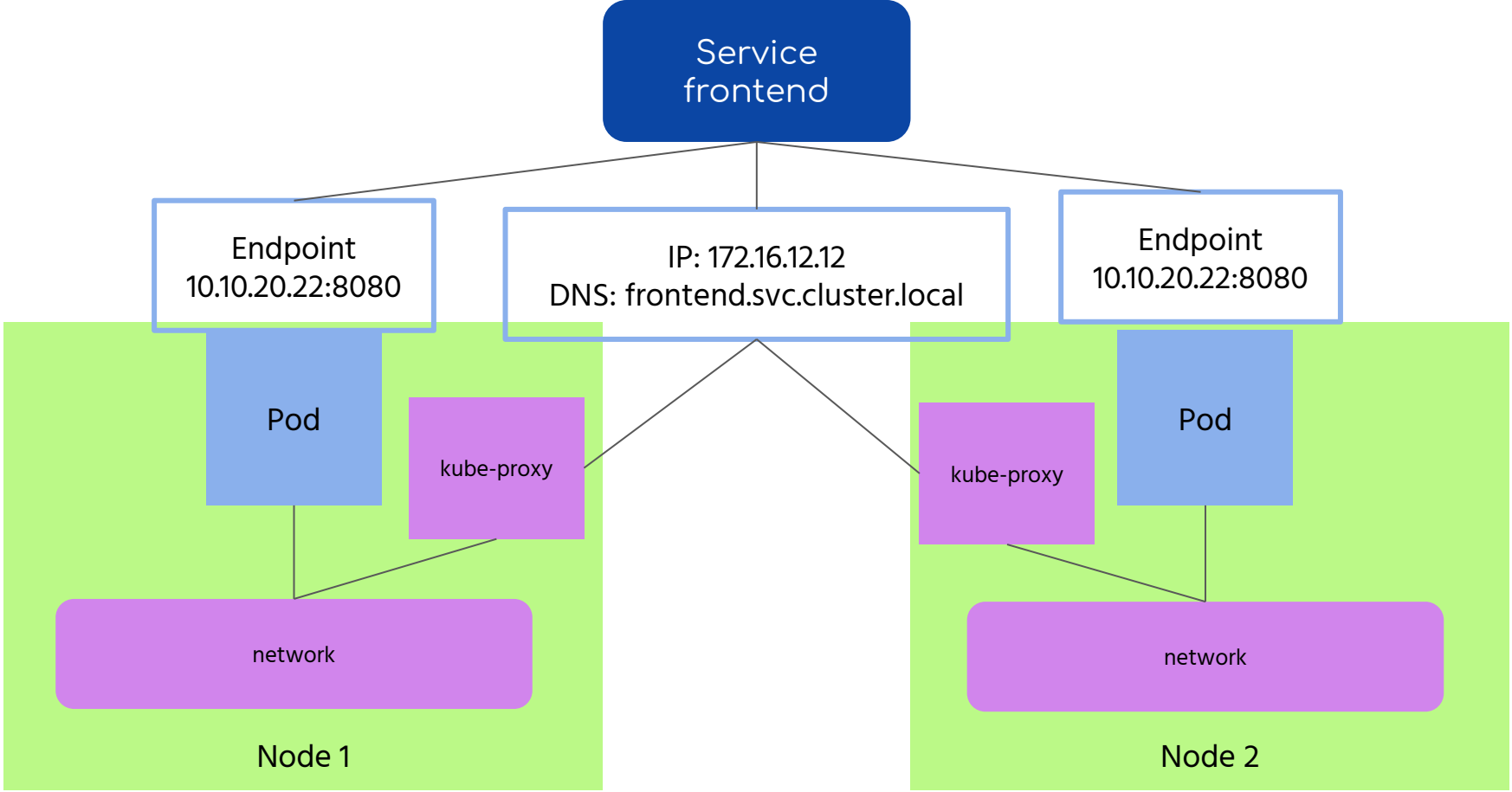
Volumes


Pod
Container
/data
Volume
iscsi
nfs
aws ebs
hostPath
configMap
secret
pvc
Volume plugins

Persistent Volumes

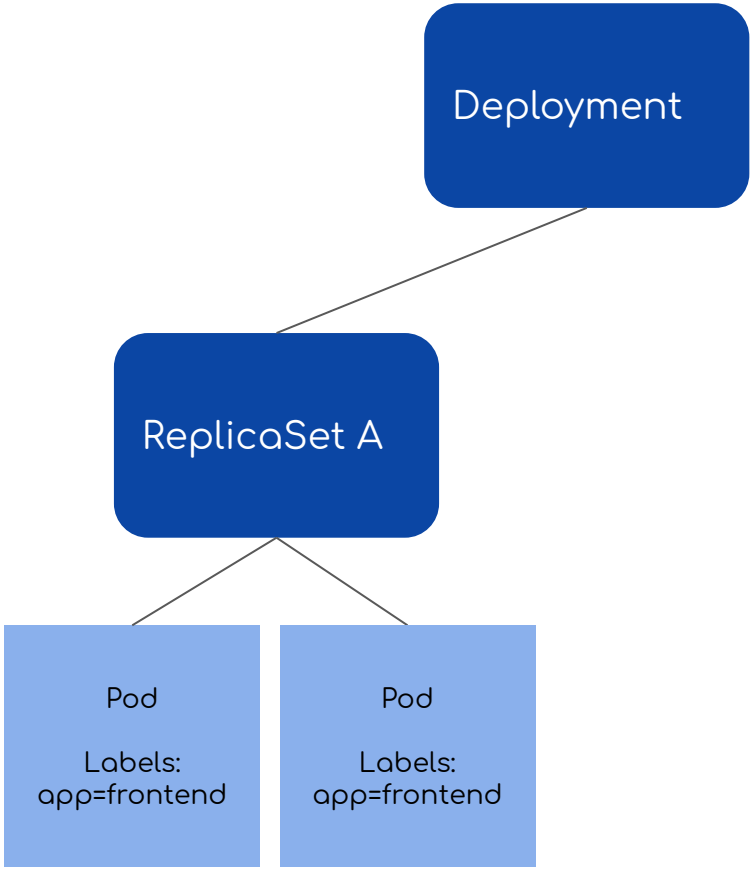
Persistent Volumes



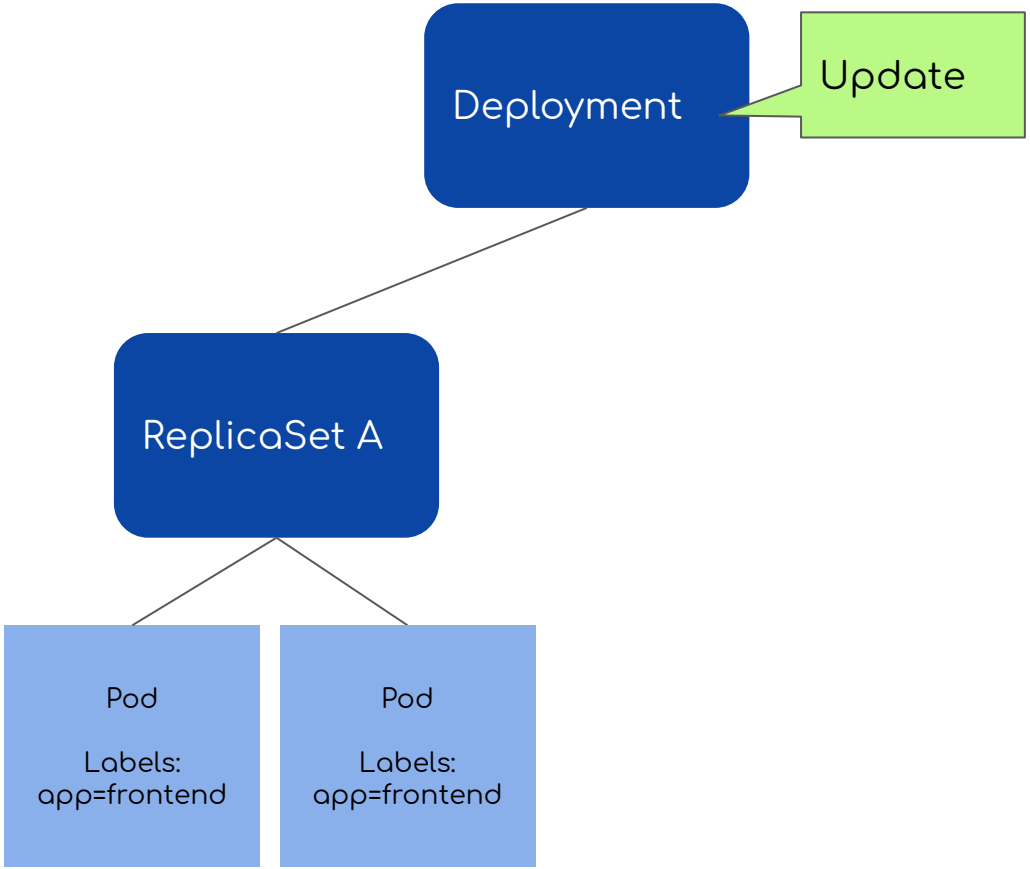


Service implementation details

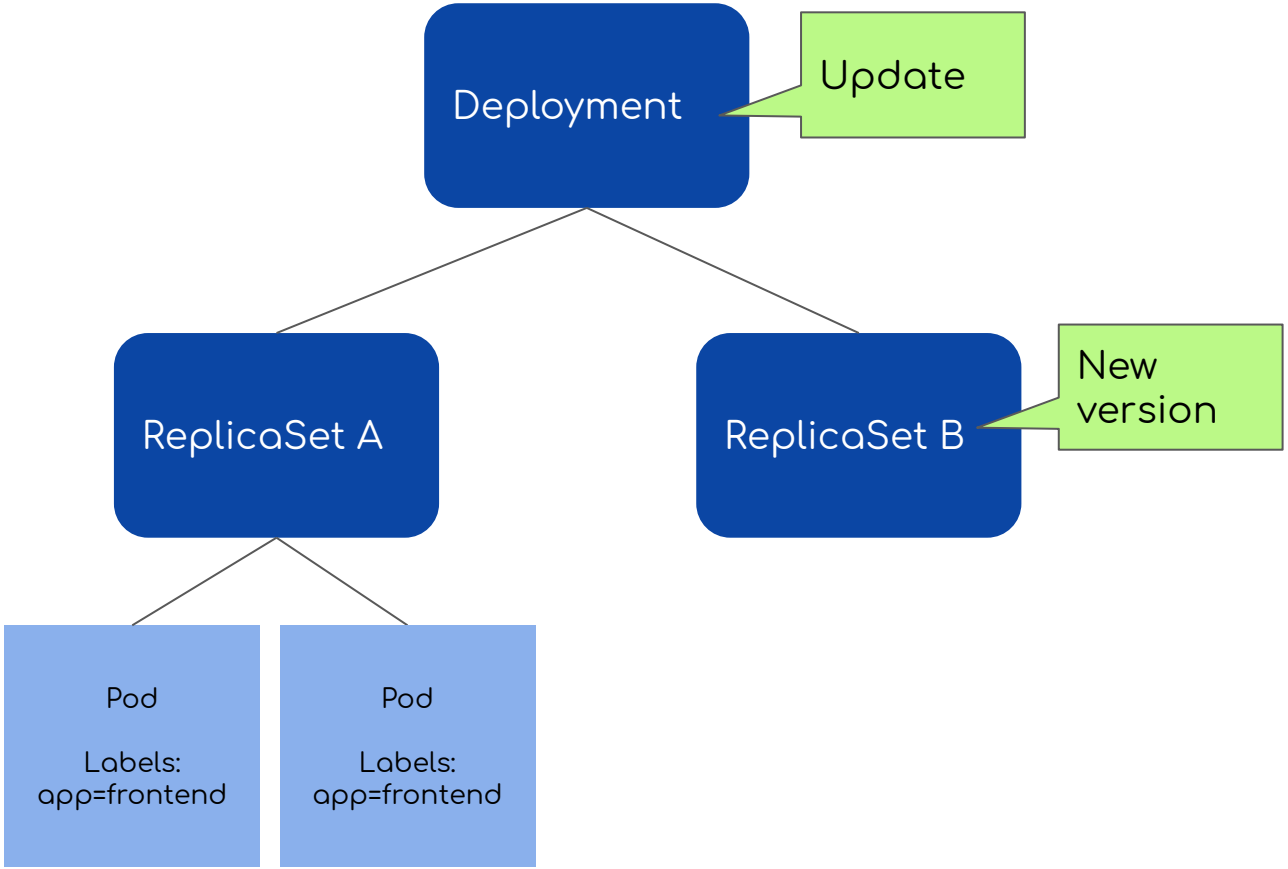
Deployment with rolling update



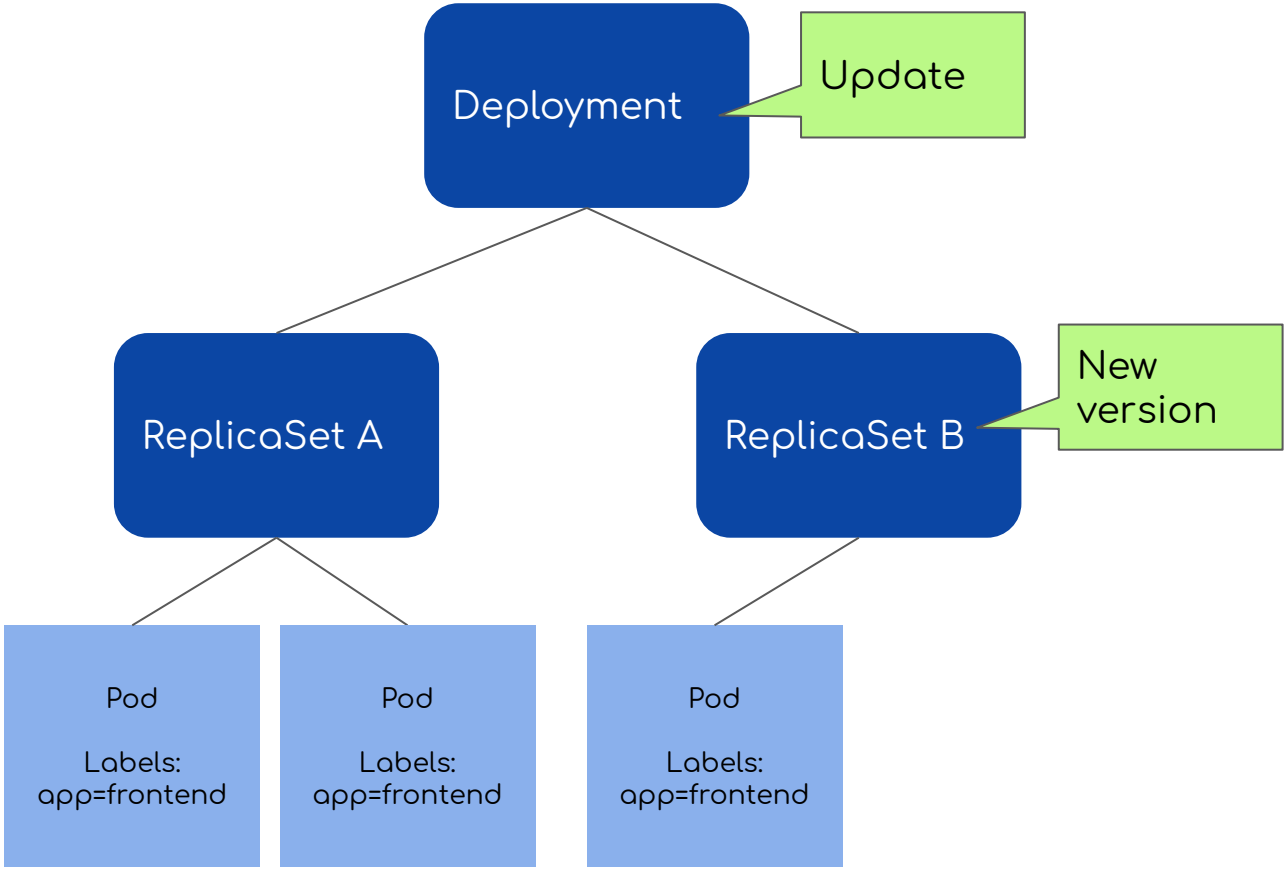
Deployment with rolling update



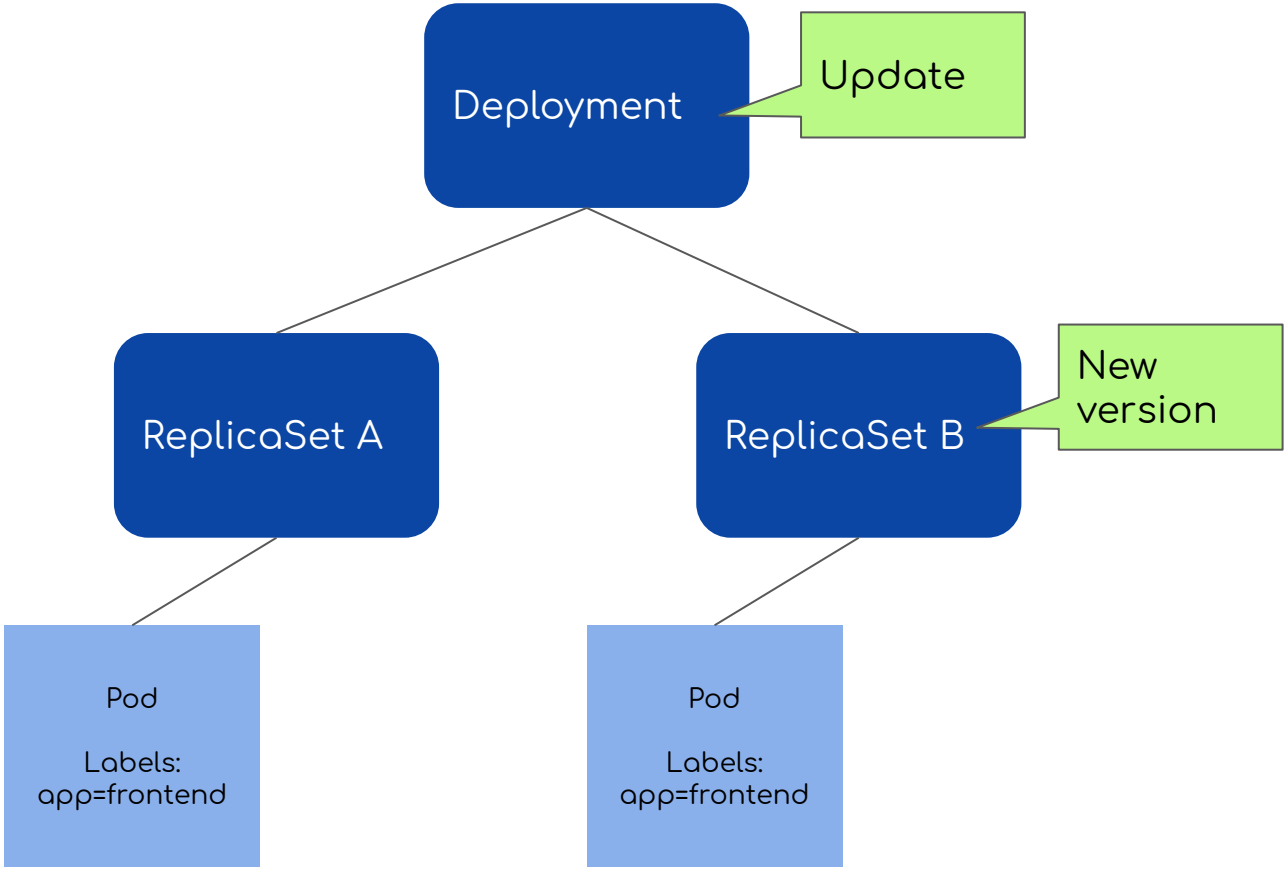
Deployment with rolling update



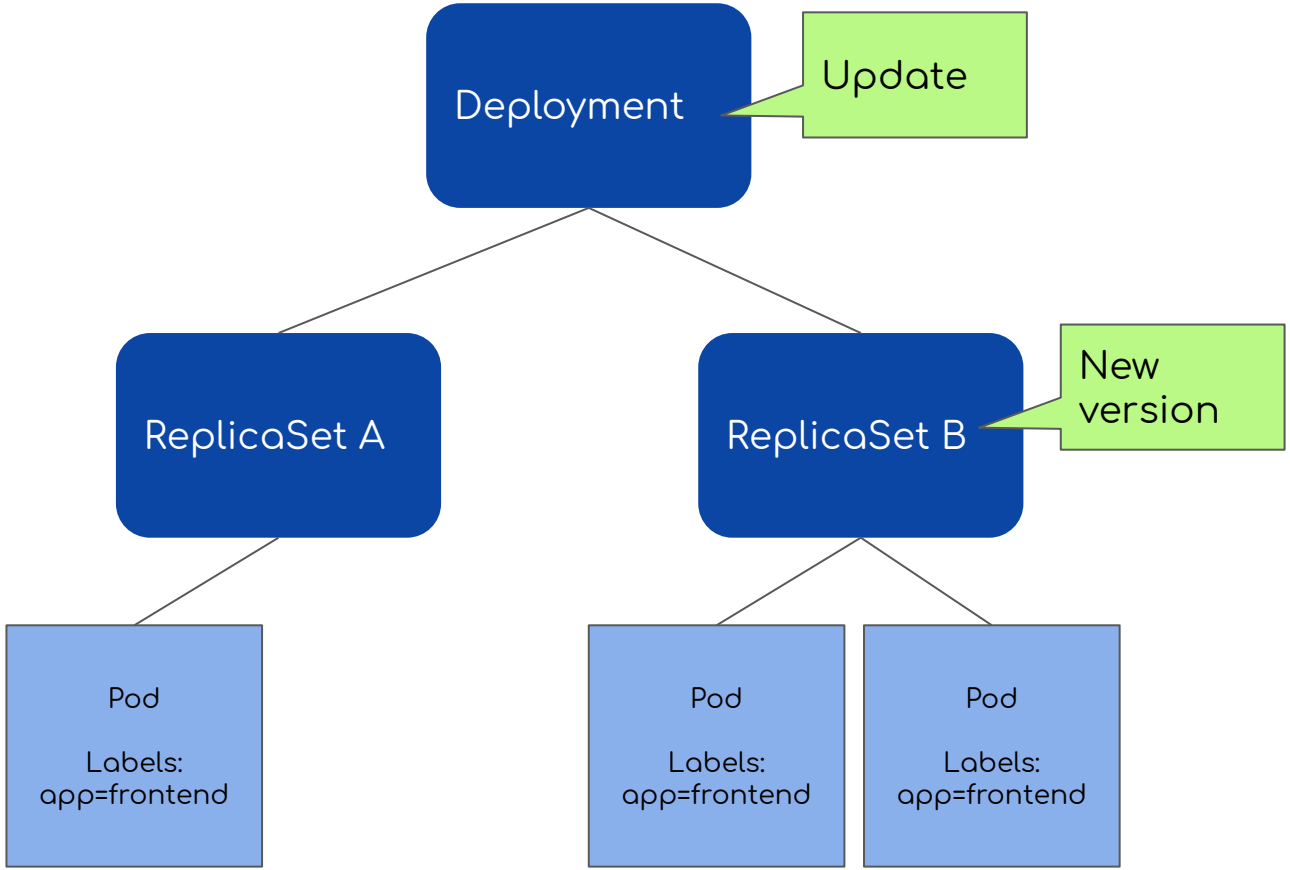
Deployment with rolling update



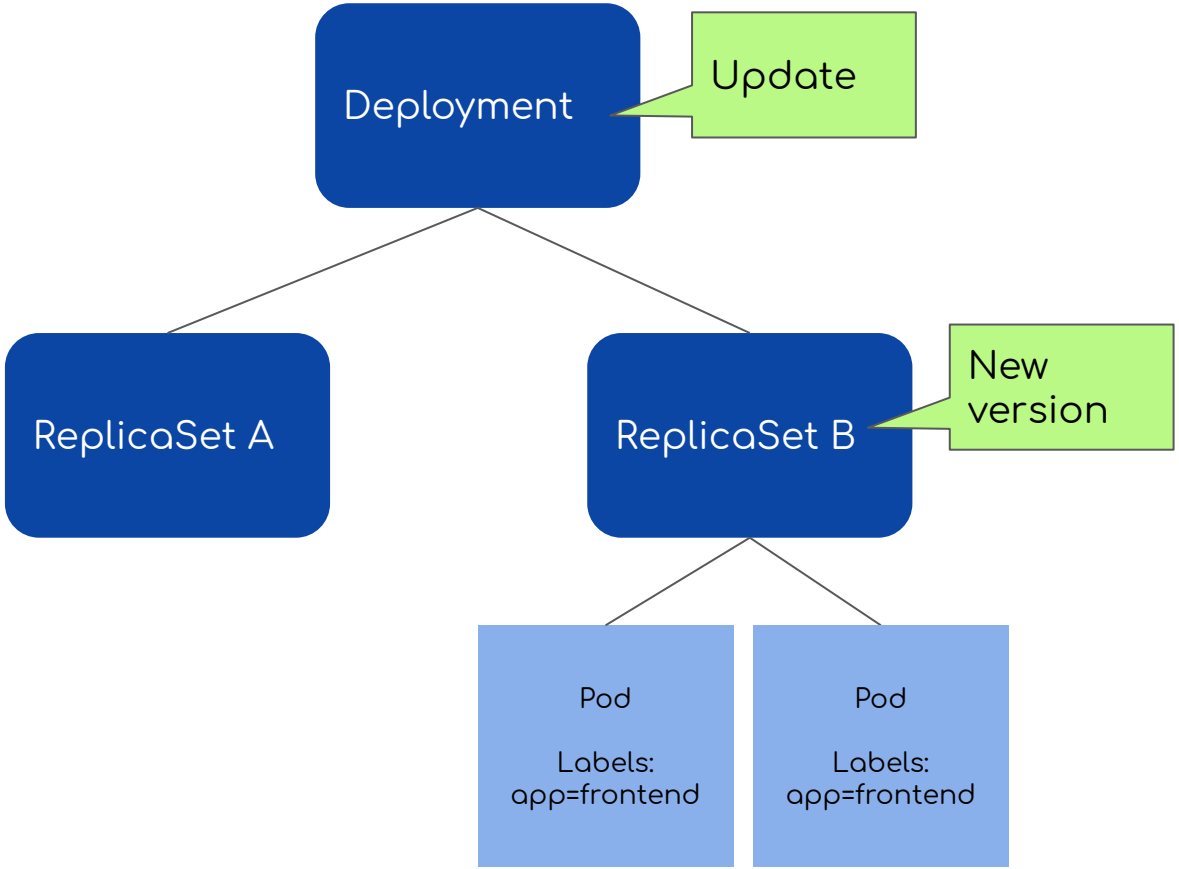
Deployment with rolling update



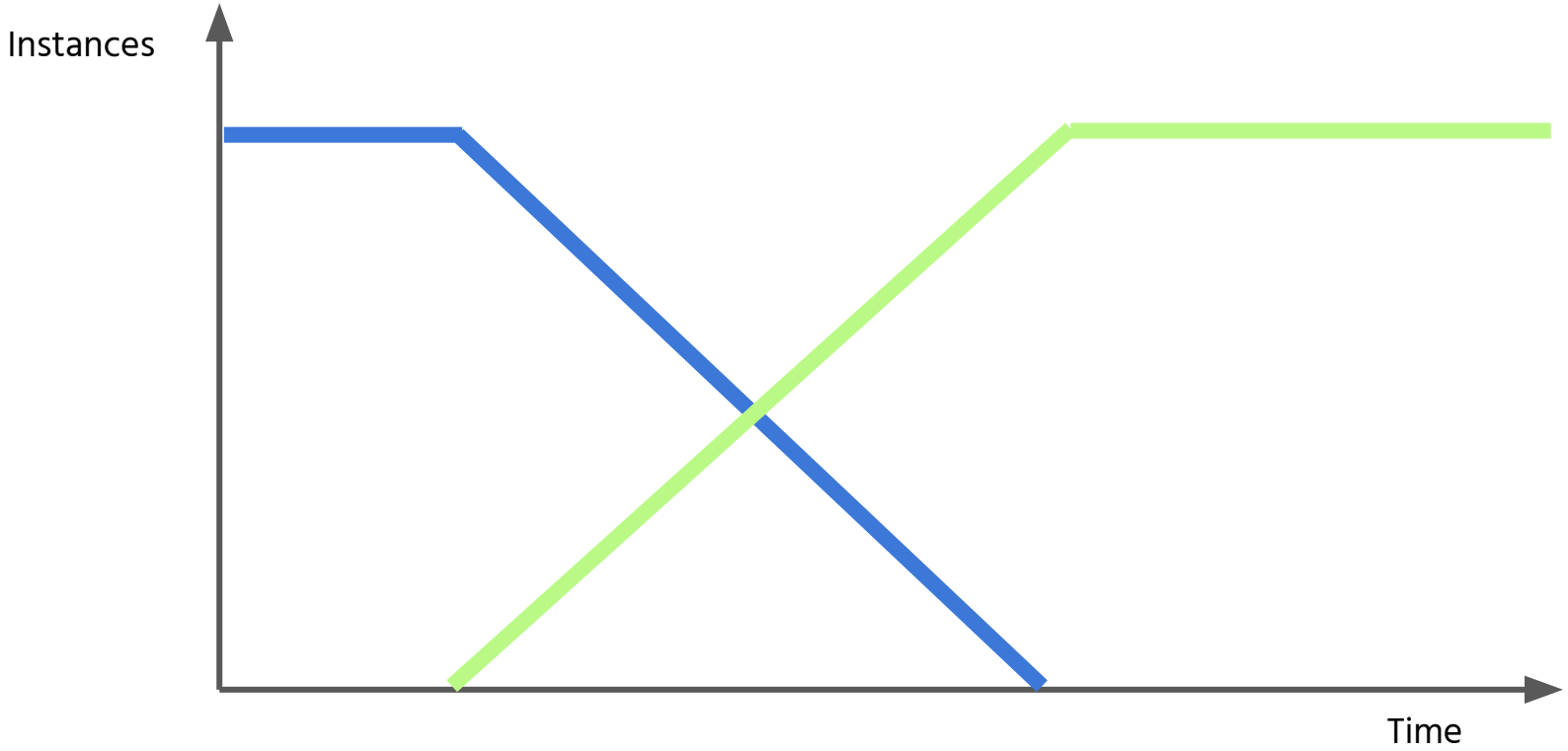
Deployment with rolling update



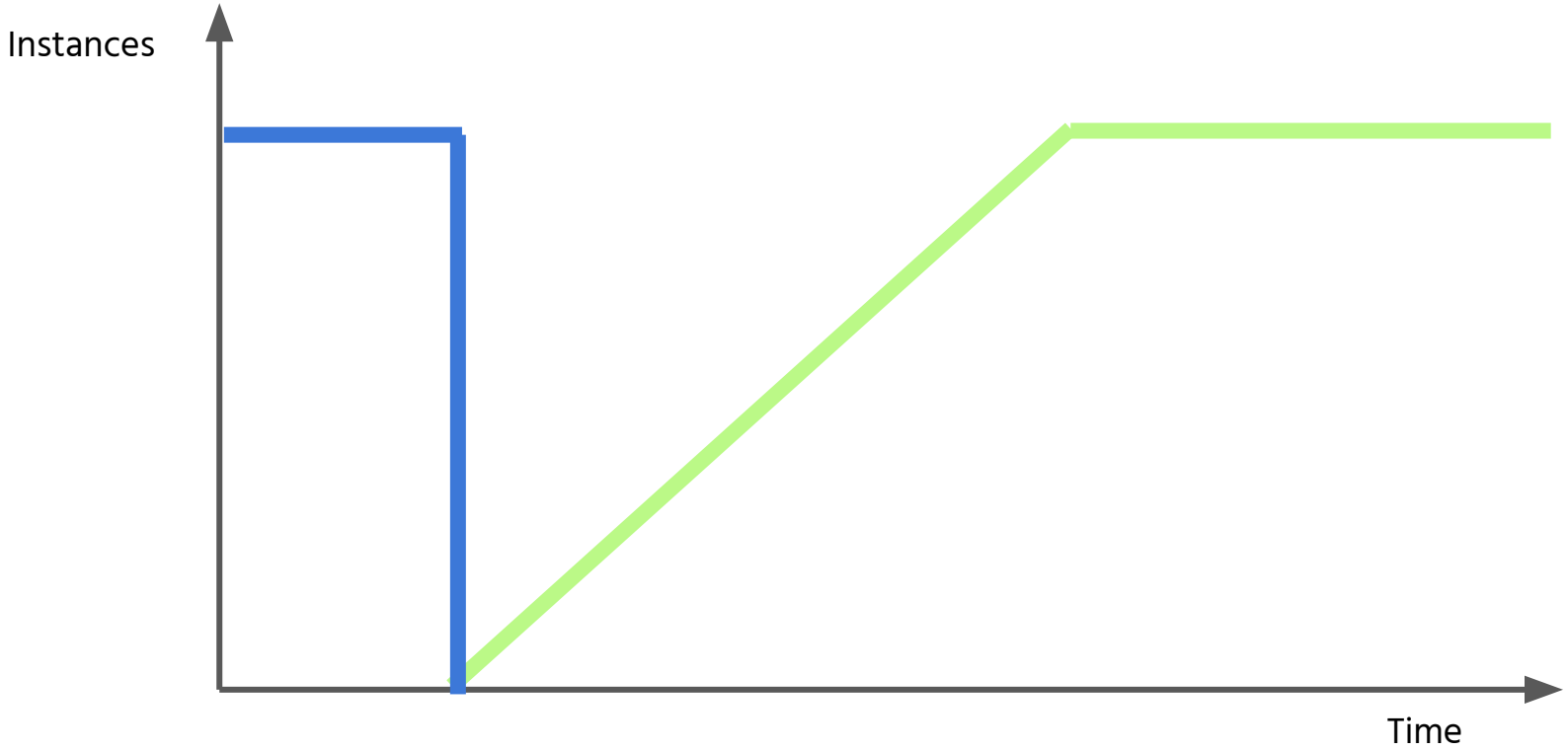
Deployment with rolling update



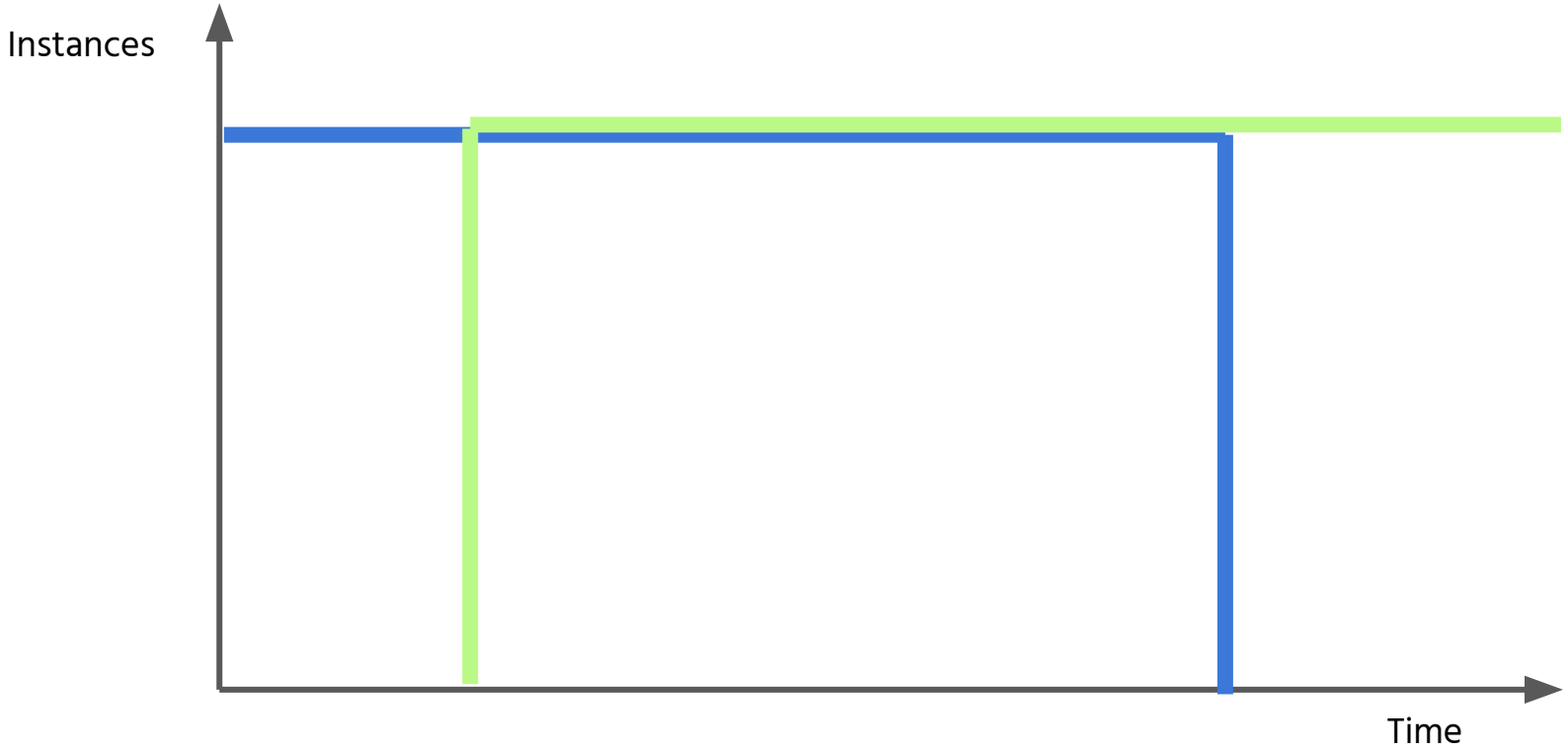
Deployment strategies - Rolling Deployment



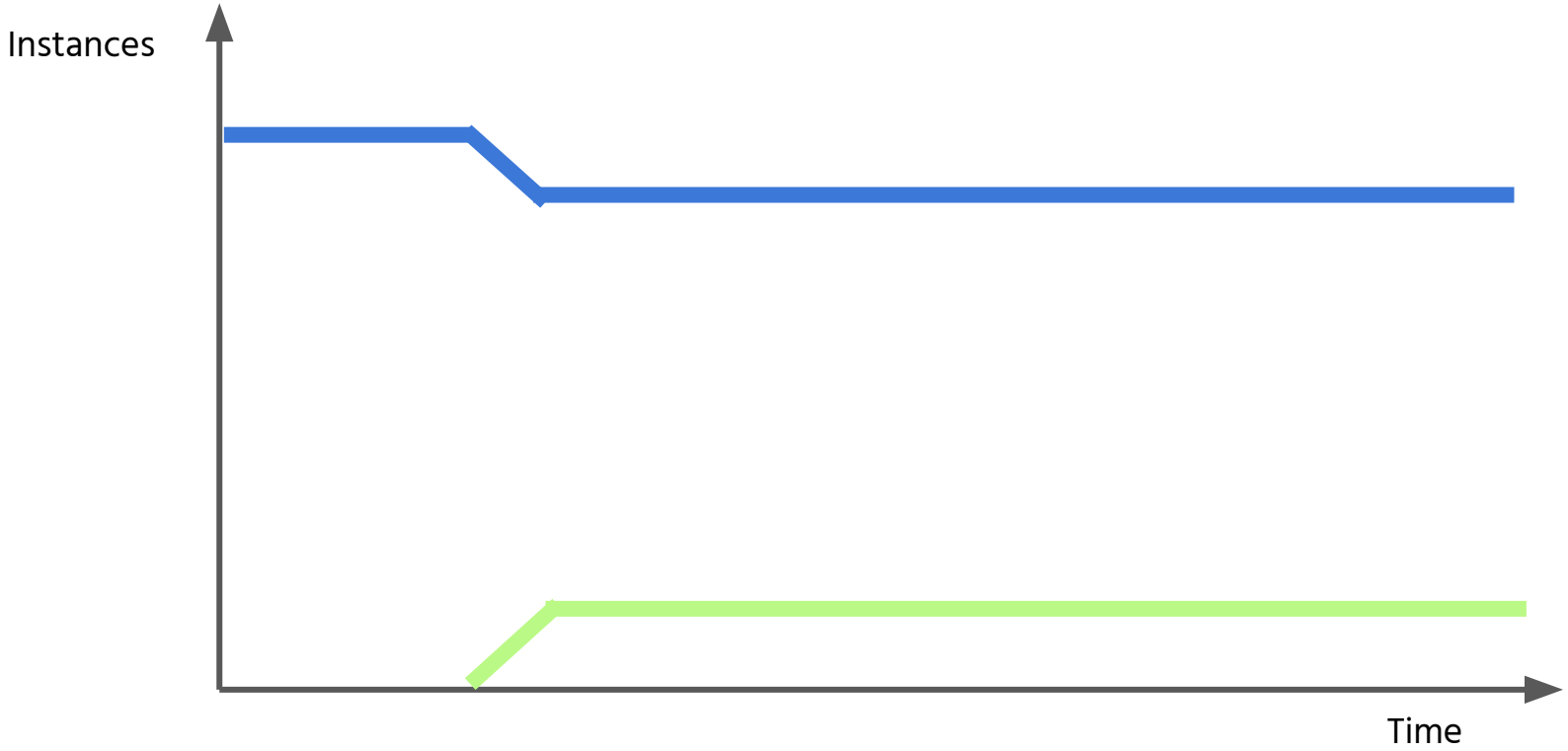
Deployment strategies - Recreate

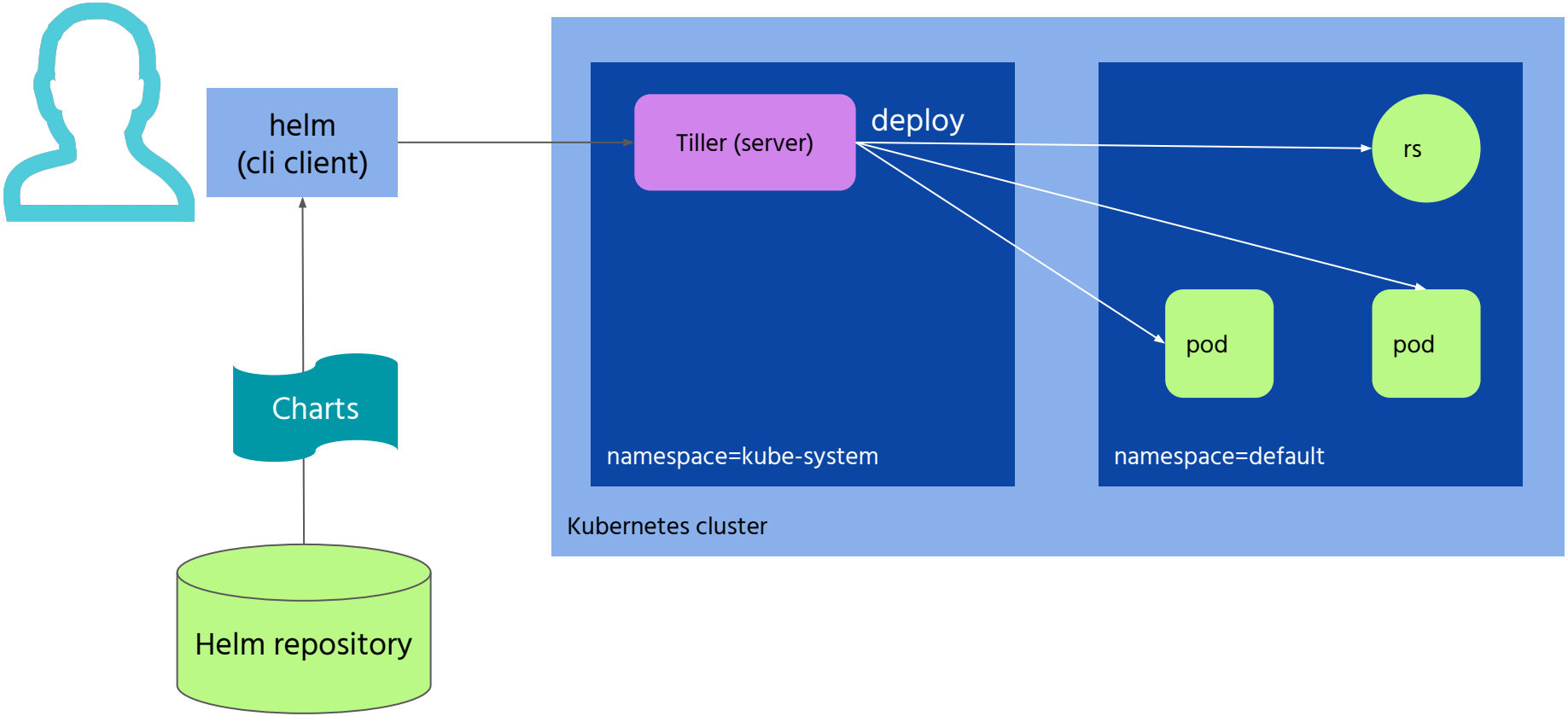


Deployment strategies - Blue-Green

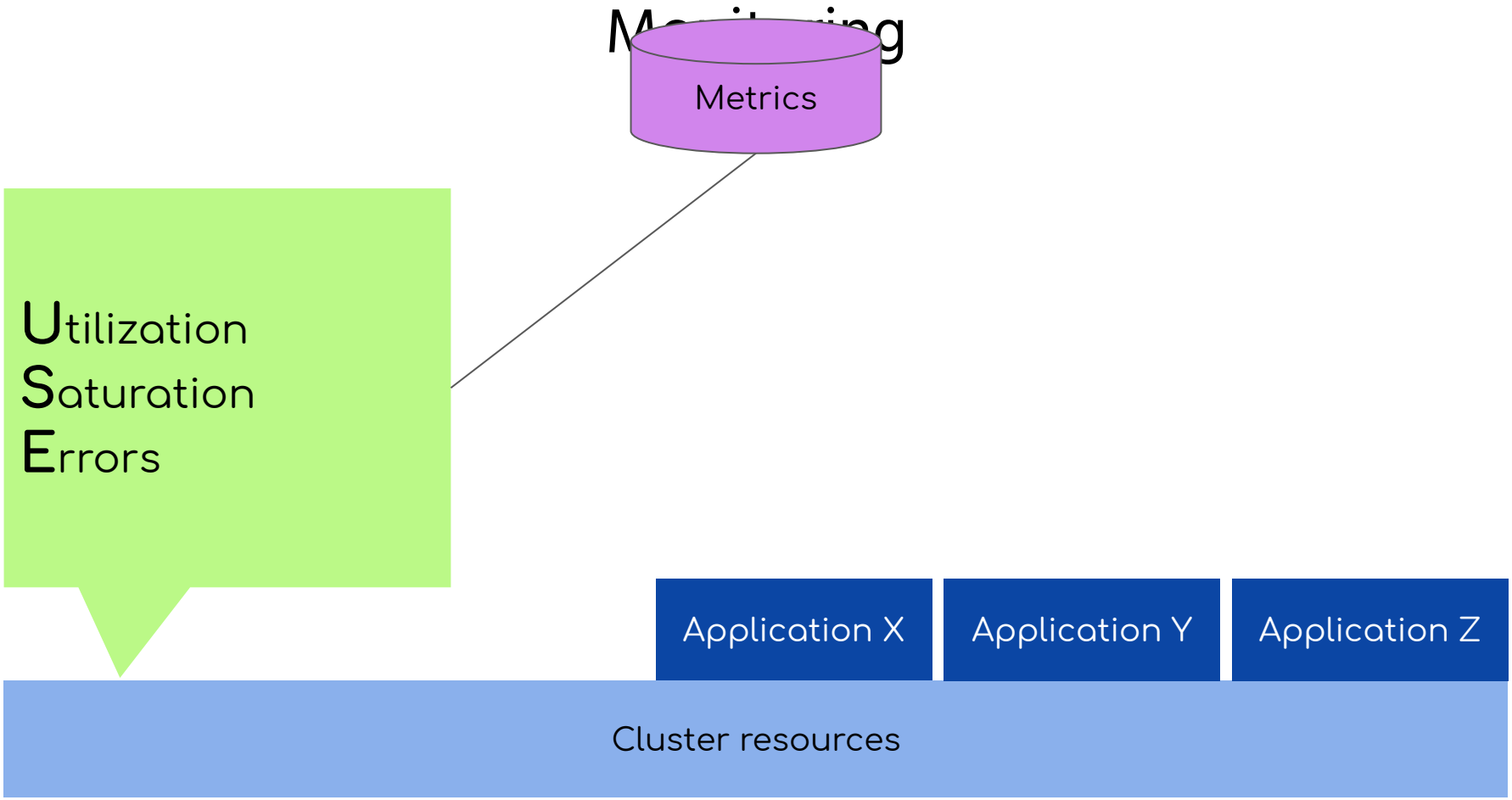


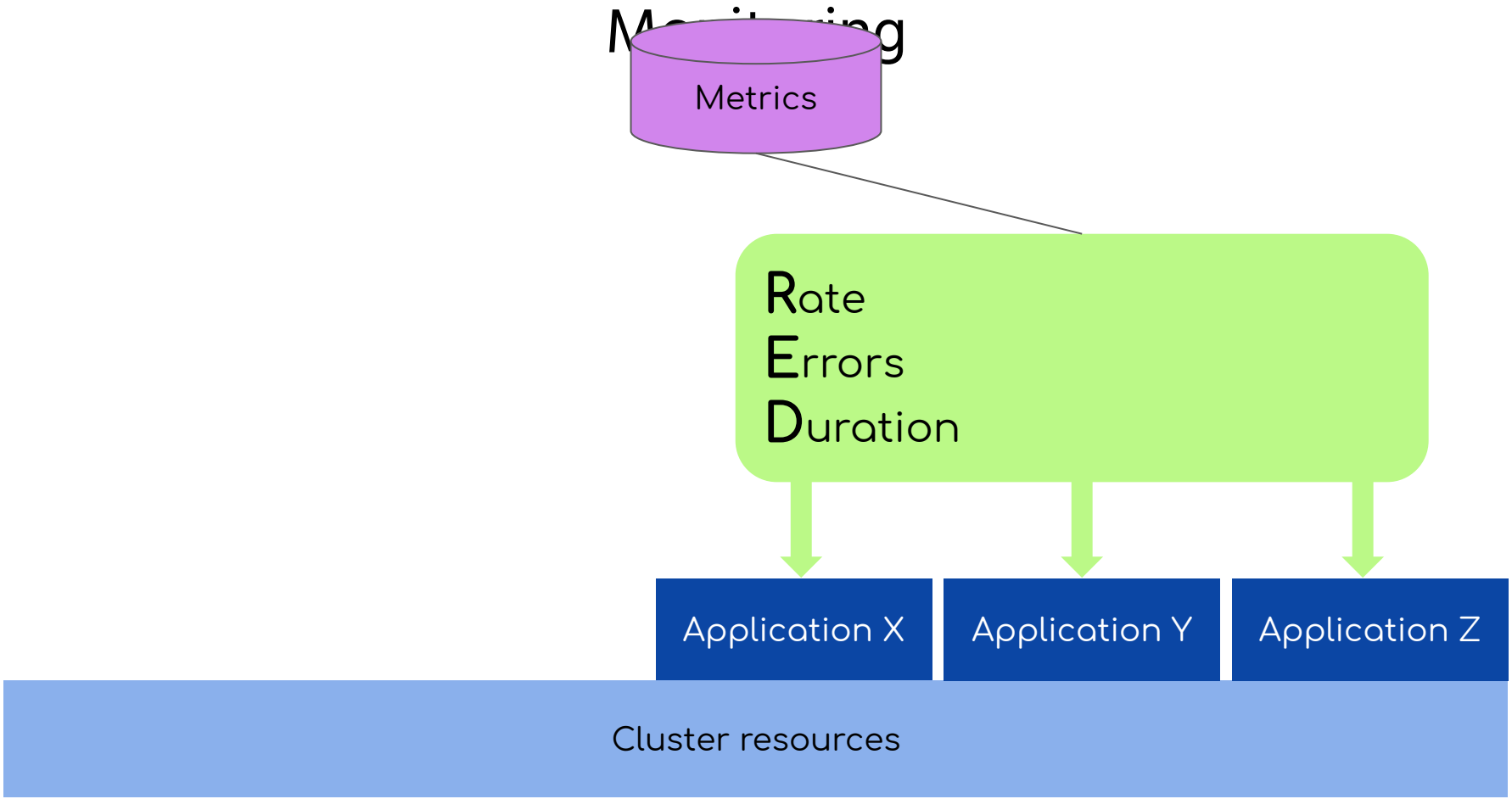
Deployment strategies - Canary Release




helm
(cli client)
Tiller (server)
deploy
rs
pod
pod
namespace=kube-system
namespace=default
Kubernetes cluster
Charts
Helm repository

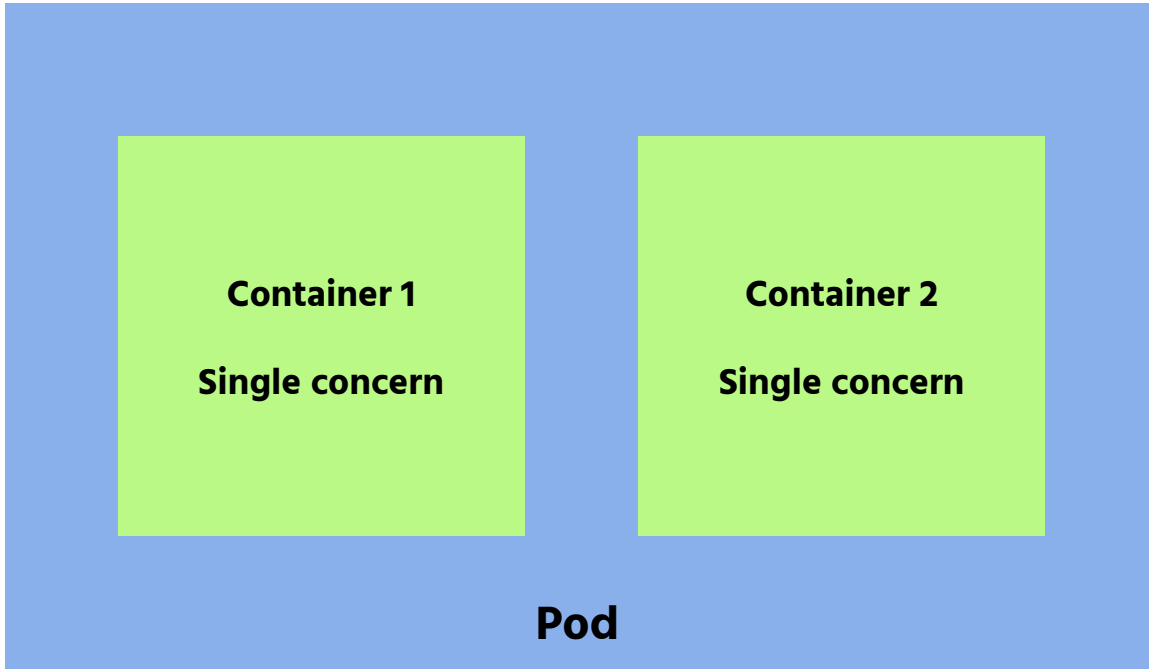
Helm


Monitoring
Metrics
Utilization
Saturation
Errors
Application X
Application Y
Application Z
Cluster resources

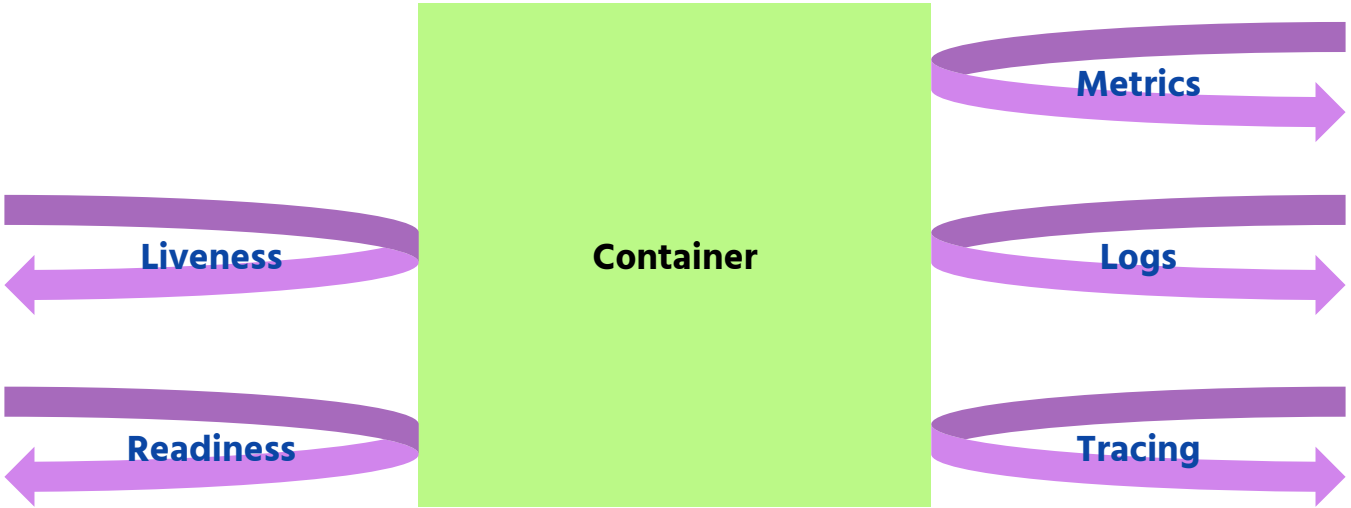

Metrics
Rate
Errors
Duration
Application X
Application Y
Application Z
Cluster resources

Patterns

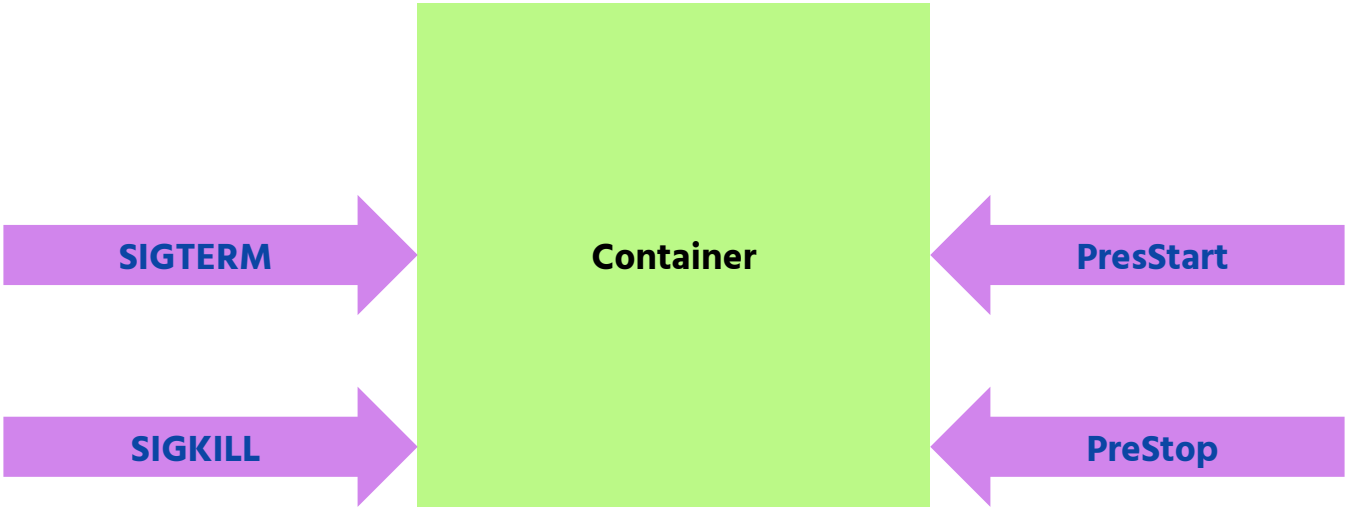
Single Concern Principle



High Observability Principle



Life-cycle Conformance Principle



Process Disposability Principle

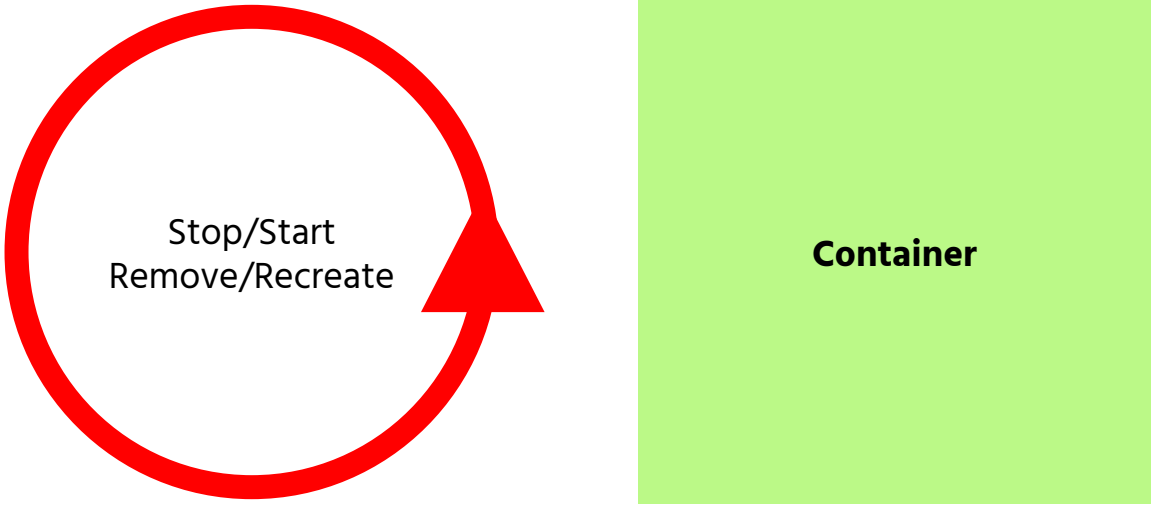
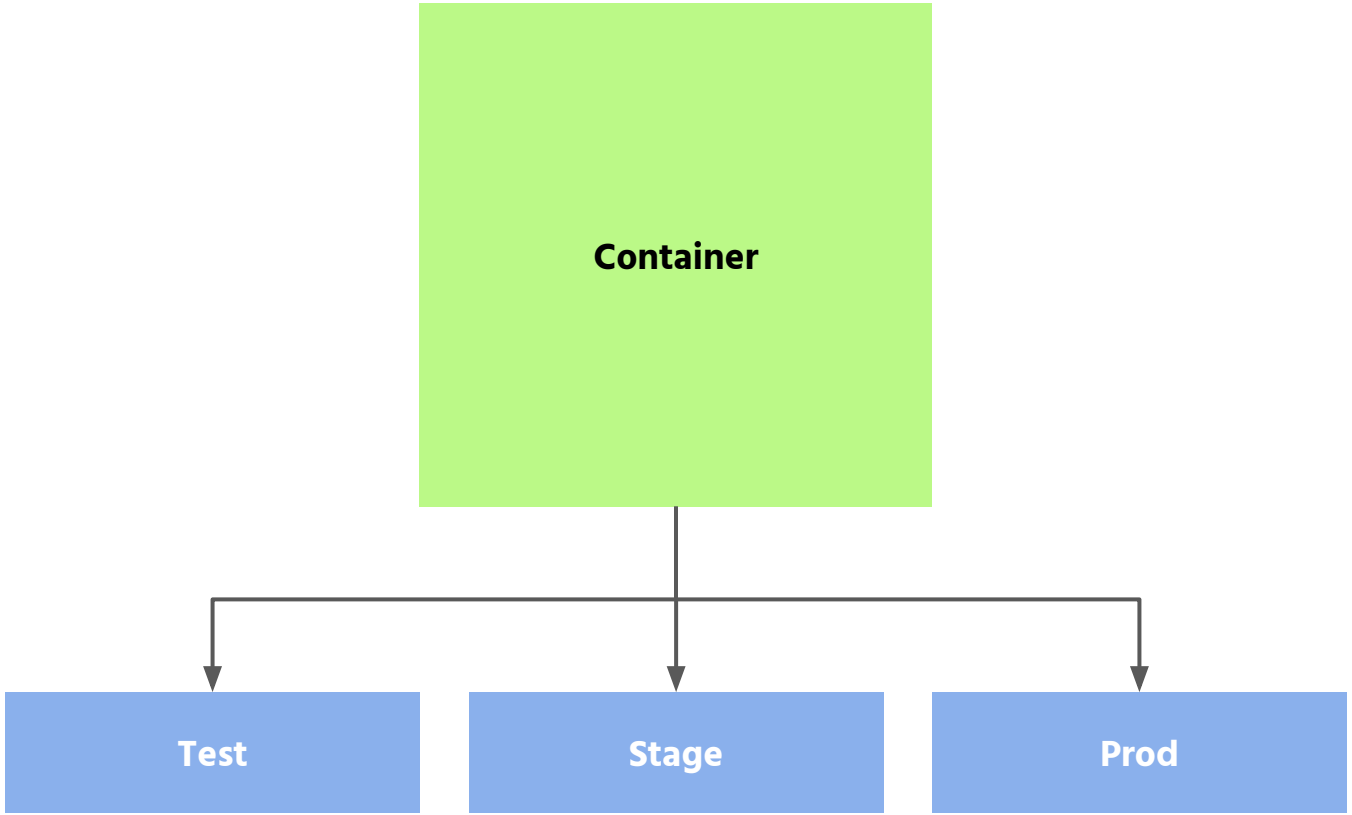
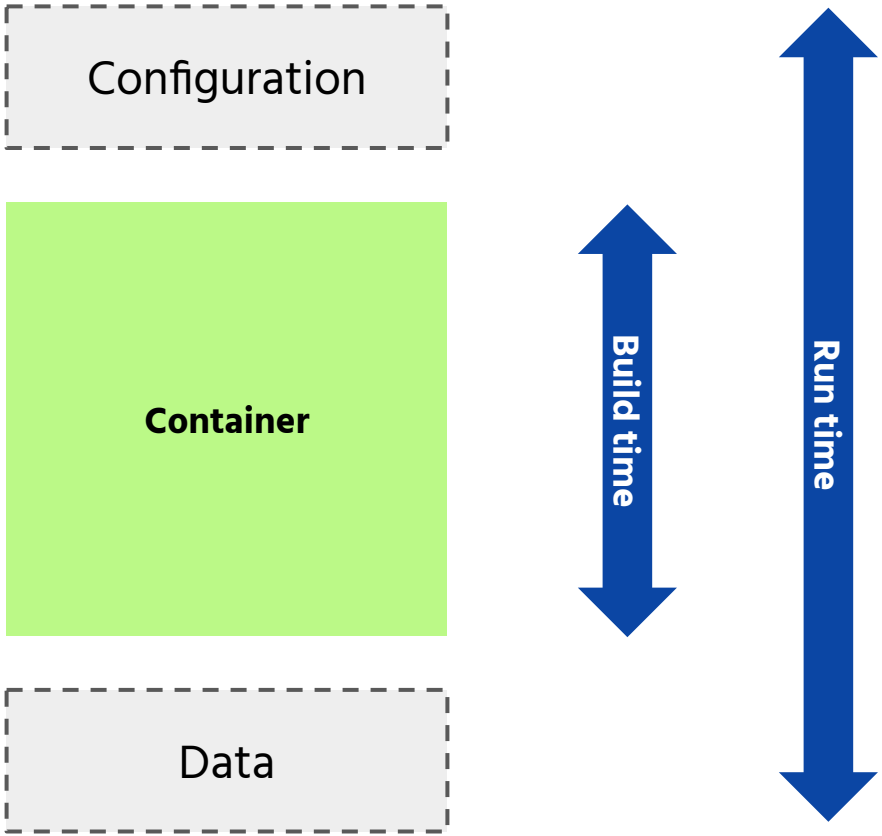


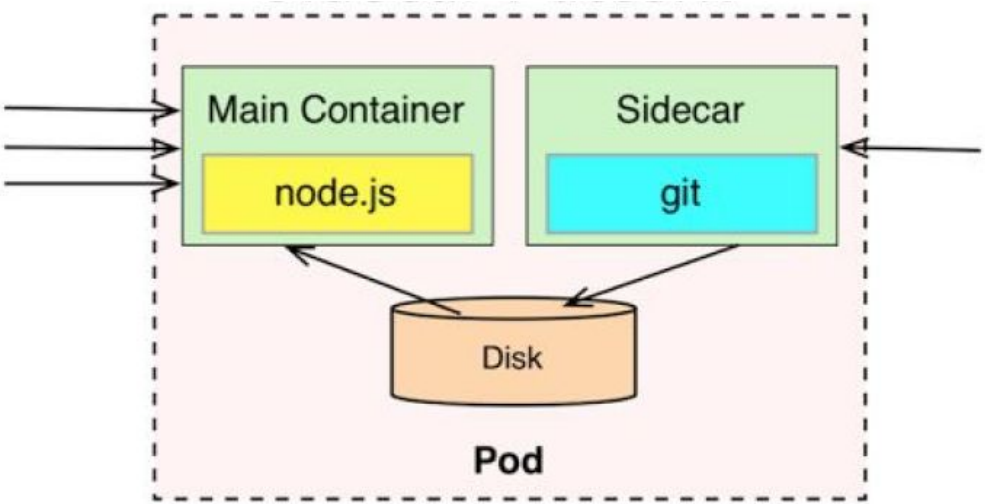
Image Immutibility Principle



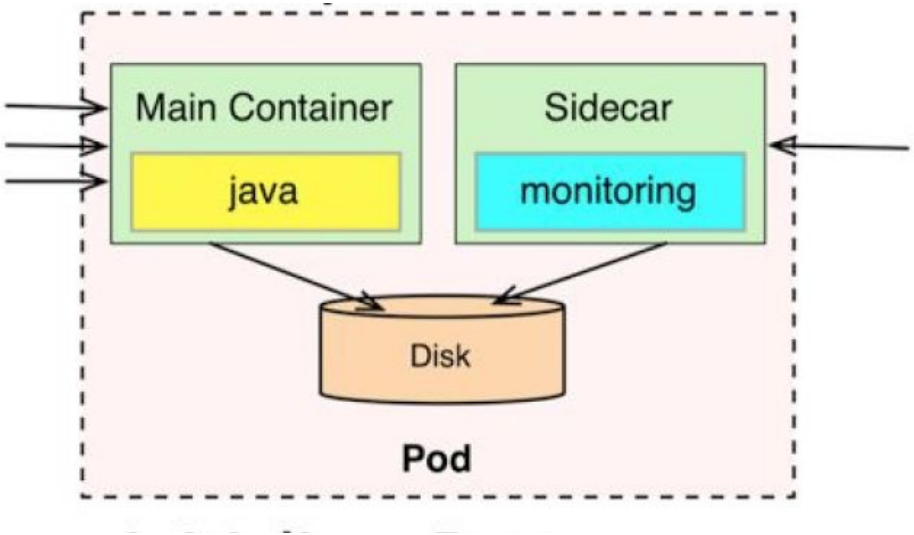
Self-containment Principle



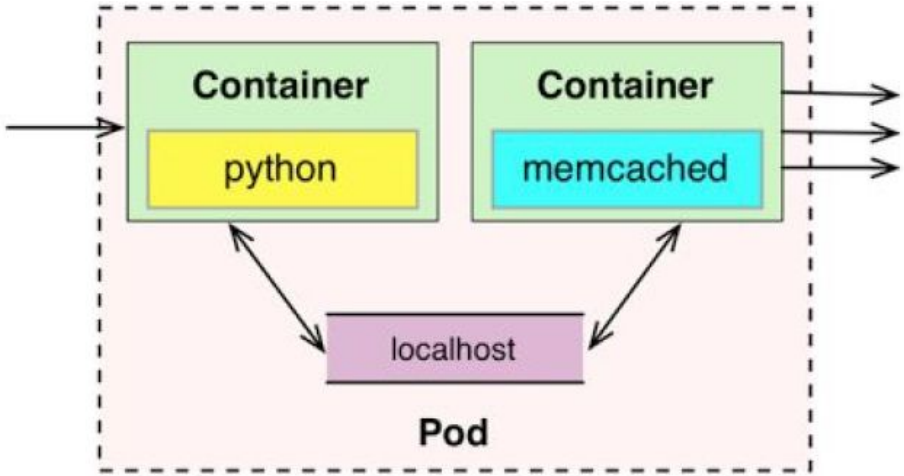
Sidecar Pattern



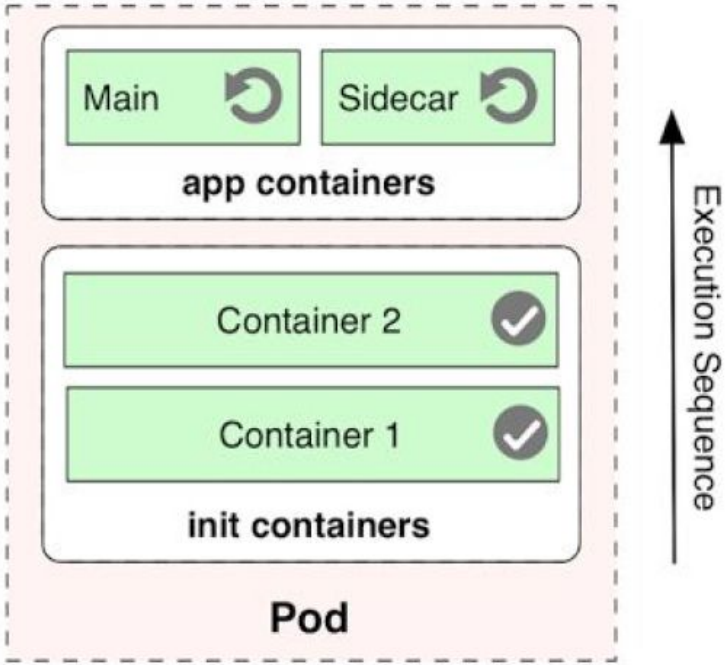
Adapter Pattern



Ambassador Pattern



Initializer Pattern



Self-awareness Pattern