VMUG
VMWARE USER GROUP
BULGARIA

THE JOURNEY TO

VMware Tanzu



Part 1: Getting to know Kubernetes with Minikube

Agenda

- A Word on Containerization
- The Journey to Kubernetes
- Kubernetes Architecture
- Kubernetes Objects
- Work with Kubernetes
- Kubernetes in Action

A Word on Containerization

What is it and how it compares to other technologies?

Containerization

“ OS-level virtualization refers to an operating system paradigm in which the kernel allows the existence of **multiple isolated user space instances** known as **containers, zones, jails, …** ”

https://en.wikipedia.org/wiki/OS-level_virtualization

VMs vs Containers

Virtual Machines

- VMs virtualize the hardware
- Complete isolation
- Complete OS installation
- Require more resources
- Run almost any OS

Containers

- Containers virtualize the OS
- Lightweight isolation
- Shared kernel
- Require fewer resources
- Run on the same OS

Definitions

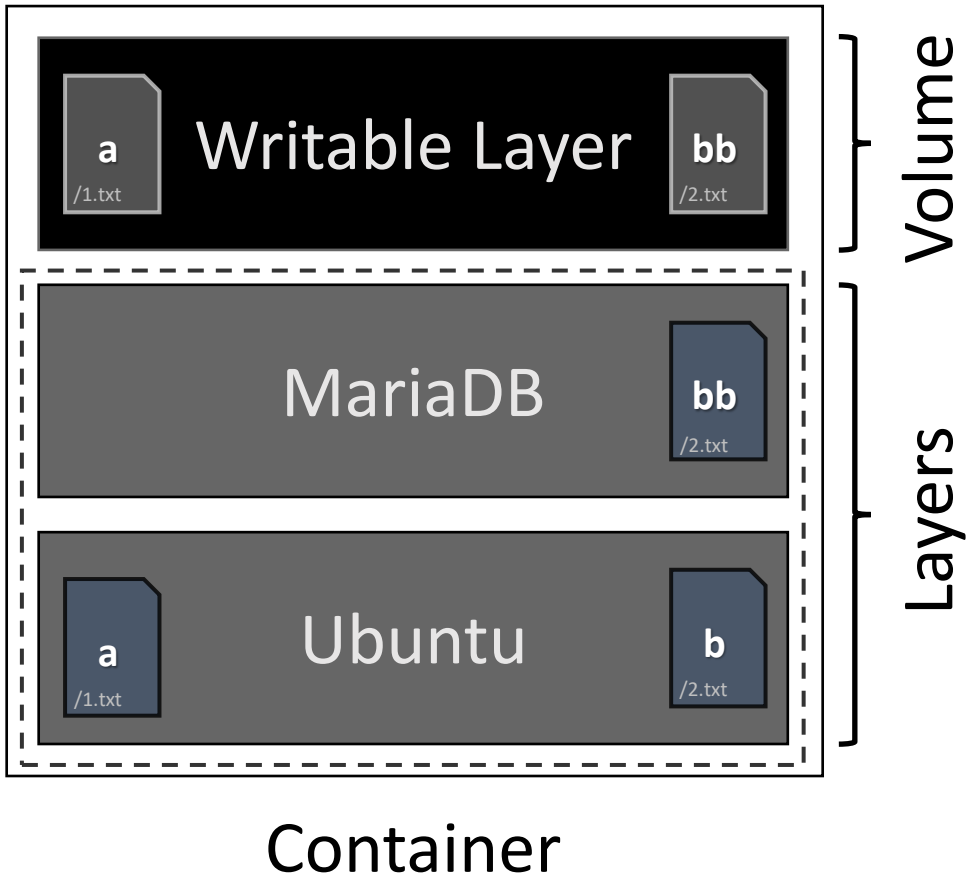
- **Container**

  A runnable instance of an image. Containers are processes with much more isolation
- **Image**

  A read-only template of a container built from layers. Images provide a way for simpler software distribution
- **Repository**

  A collection of different versions of an image identified by tags
- **Registry**

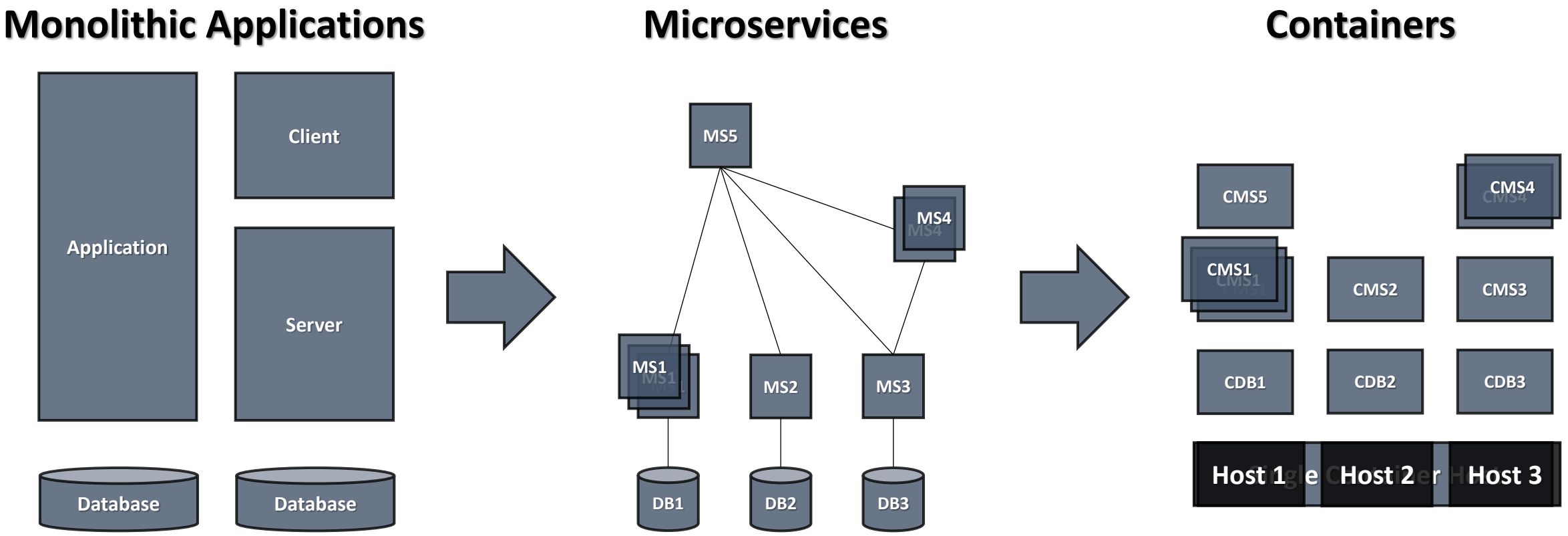
  A collection of repositories



The Journey to Kubernetes

What is it and what problems does it solve?

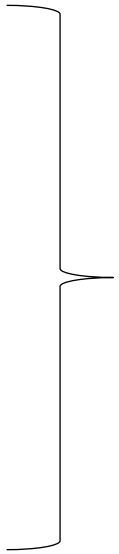
Application Evolution



Microservices != Containers

New Demands

- Workload deployment and distribution
- Resource governance
- Scalability and availability
- Automatization and management
- Internal and external communication



Container Orchestration

Orchestration Solutions *

- Docker Swarm
- Apache Mesos + Marathon
- HashiCorp Nomad
- Kubernetes

* Not a complete list

Kubernetes Origin

- Born out of projects like **Borg** and **Omega** at **Google**
- Written in **Go**
- Donated to **CNCF** in **2014**
- Open source, licensed under **Apache 2.0**
- **Version 1.0** came into existence in **July 2015**. Current is **1.21.0**
- **κυβερνήτης** in Greek means **Helmsman** – s.o. who steers the ship
- Can be seen often shortened as **k8s**

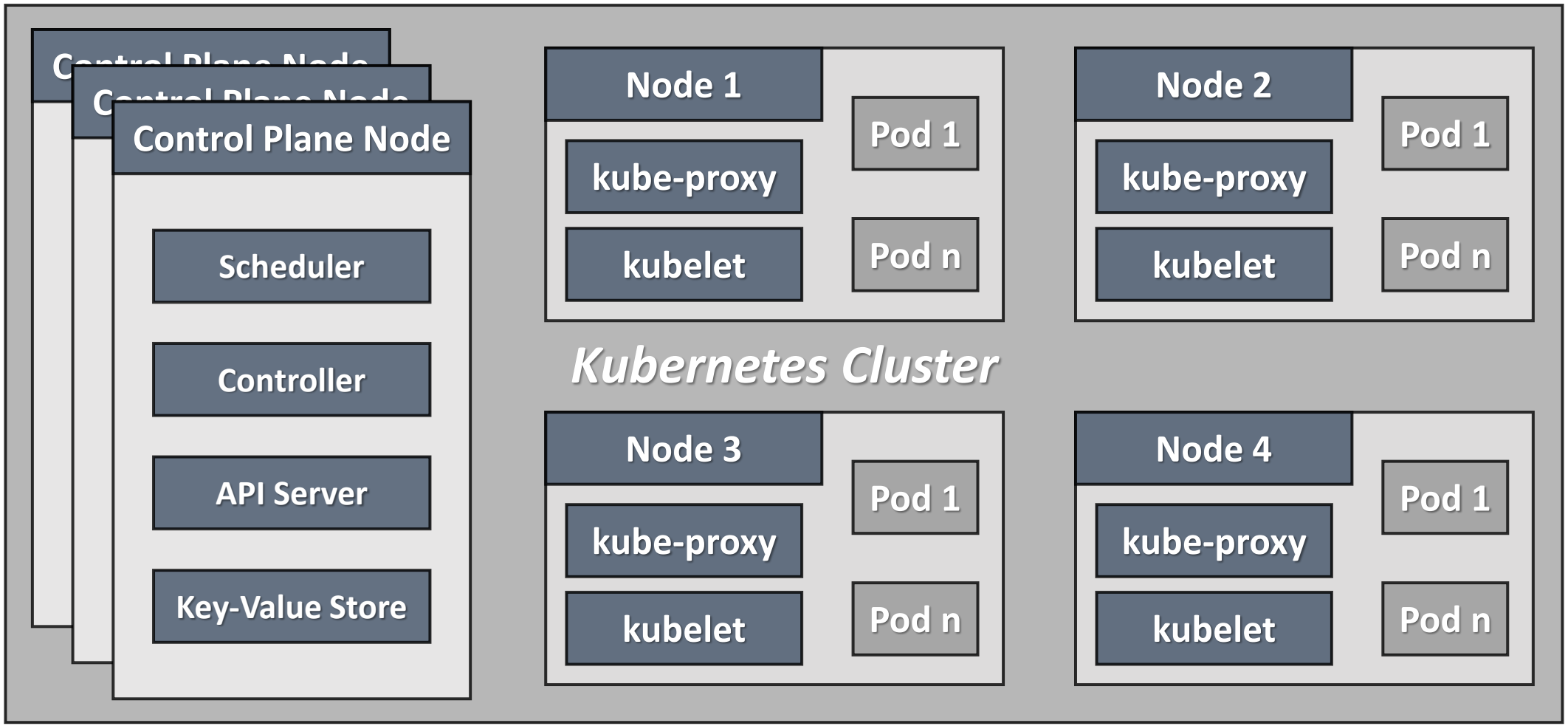
Kubernetes Got You Covered

- Runs a cluster of hosts
- Schedules containers to run on different hosts
- Facilitates the communication between the containers
- Provides and controls access to/from outside world
- Tracks and optimizes the resource usage

Kubernetes Architecture

What is under the hood?

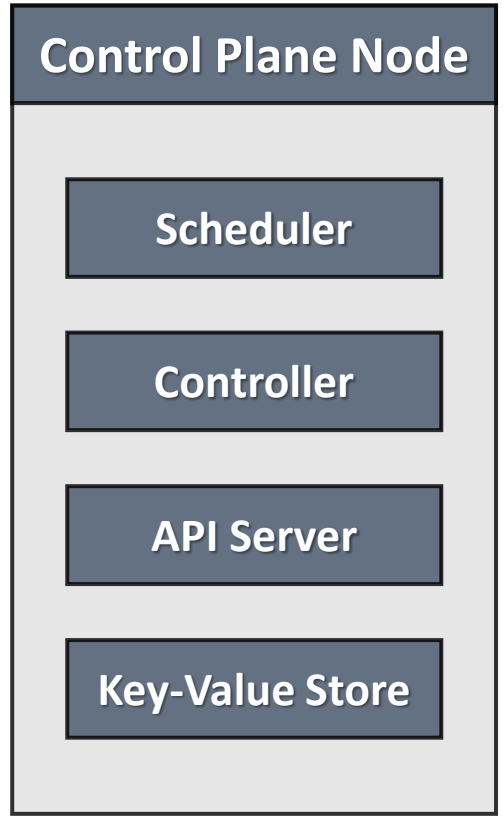
Architecture Overview *



* Not all components are shown

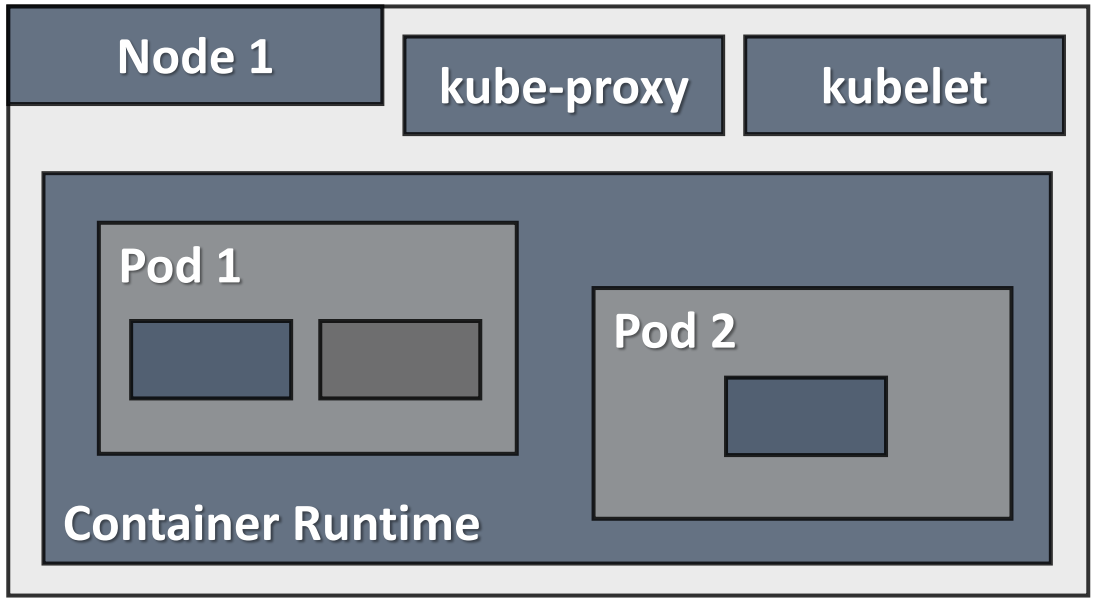
Control Plane (Master) Nodes

- Responsible for managing the cluster
- **Key-Value Store**
  - Persistent storage for cluster state and configuration
- **API Server**
  - Exposes the Kubernetes API
- **Controller**
  - Runs controller processes
- **Scheduler**
  - Assigns work to nodes



(Worker) Nodes

- **Container Runtime**
  - Docker, containerd, CRI-O, etc.
  - Pulls images
  - Starts and stops containers
- **kubelet**
  - Registers nodes in the cluster
  - Communicates with control-plane
  - Creates pods
- **kube-proxy**
  - Provides the networking
  - Load balancing of pods in a service



Kubernetes Objects

Objects Overview

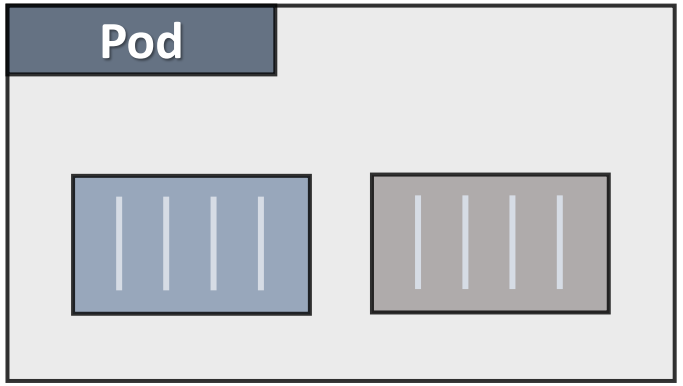
- Kubernetes objects are persistent entities
- They are used to represent the state of the cluster
- An object is a "**record of intent**". Once created, the Kubernetes system will constantly work to ensure that object exists
- Almost every object includes two nested object fields
  - Spec provides a description of the characteristics (***desired state****)*
  - Status describes the ***current state*** of the object

Namespaces

- Kubernetes supports multiple virtual clusters called **namespaces**
- Namespaces **cannot be nested** inside one another
- Namespaces provide a **scope for names**
- Names of resources need to be **unique** within a namespace
- Each Kubernetes resource can **only be in one** namespace
- Deleting a namespace will clean up everything under it

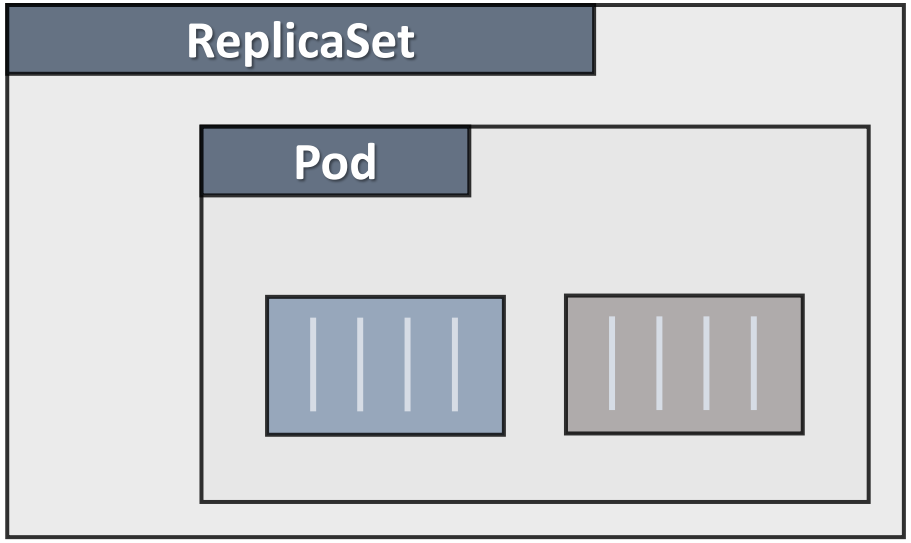
Pods

- Smallest **unit of scheduling**. **Scheduled** on nodes
- **One** or **more** containers. Containers **share** the pod **environment**
- **Deployed as one** and on **one node**. It is **atomic**
- Each pod has a **unique IP** address
- Pods communicate via a **pod network**
- Containers communicate via **localhost** and **port**
- Created via **manifest files**



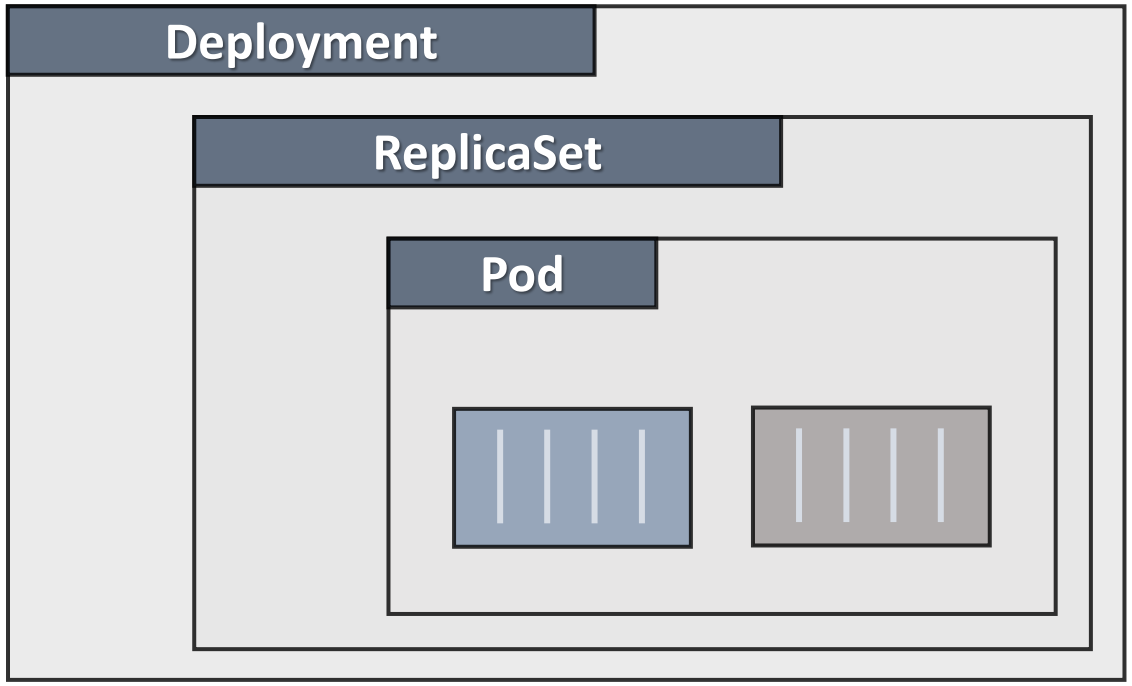
Replica Sets

- **Higher** level workload
- Looks after **pod** or **set of pods**
- **Scales** up/down **pods**
- Sets **Desired State**
- Rarely used alone by itself



Deployments

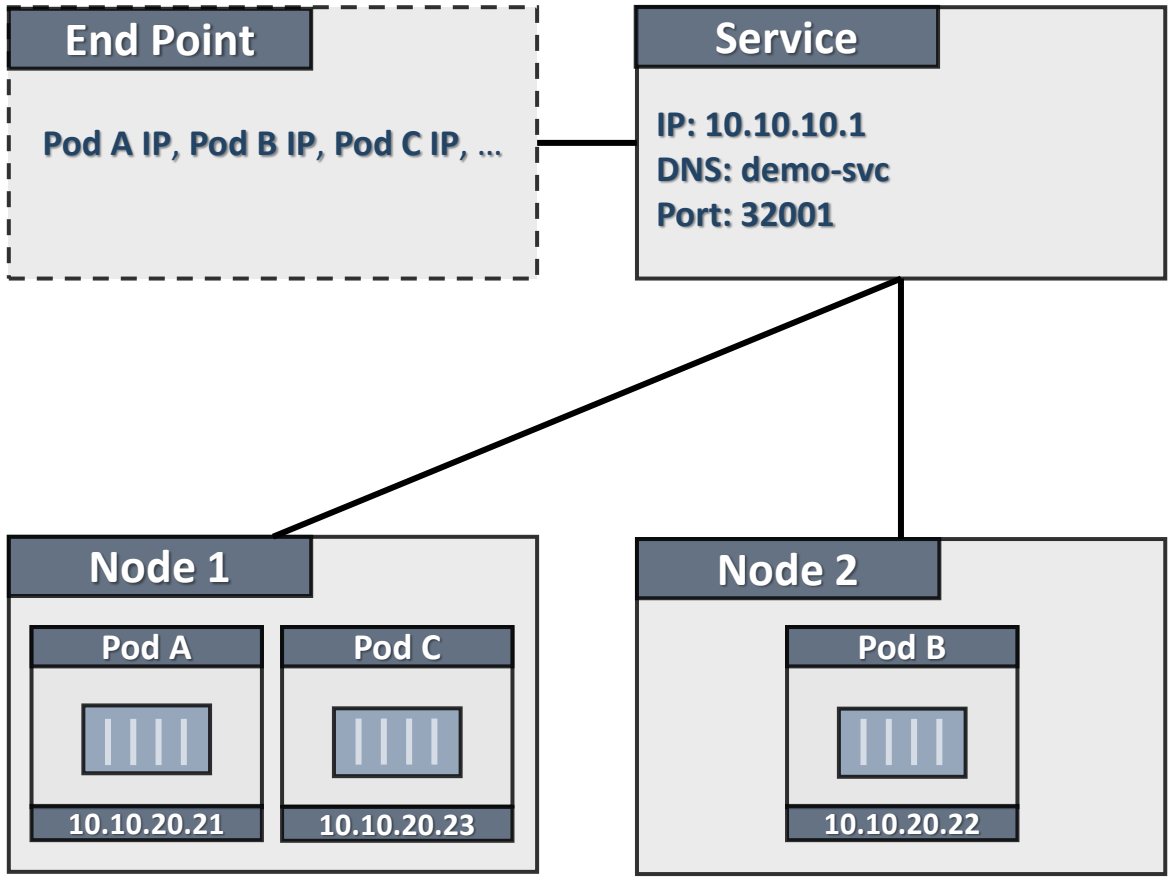
- **Even higher-level** workload
- Simplifies **updates** and **rollbacks**
- Self **documenting**
- Suitable for **versioning**



Labels and Annotations

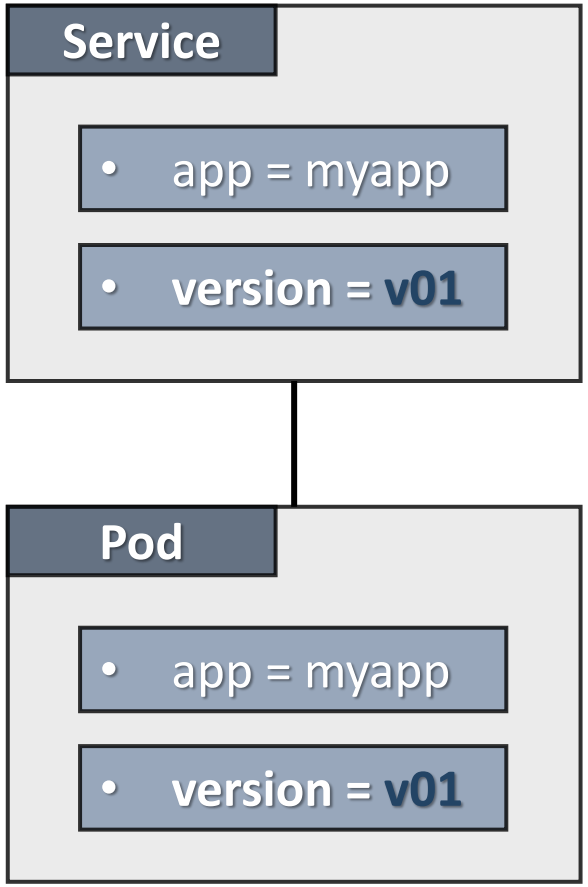
- **Labels**
  - Key-value pairs attached to objects
  - Each object may have multiple labels
  - Each label may be attached to multiple objects
  - Used to identify and group sets of objects
  - Used with **label selectors** to select a group of objects
- **Annotations**
  - Key-value pairs attached to objects
  - Used to store additional information (metadata) like description, creator, etc.

Services



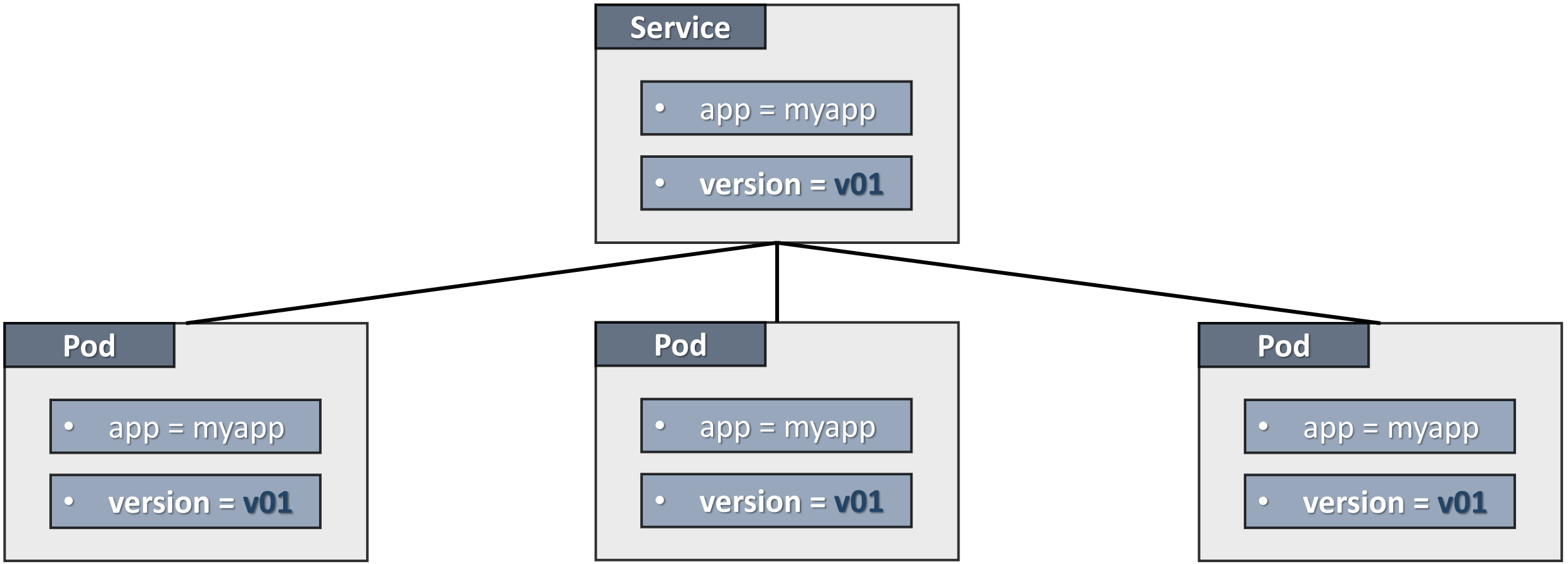
- Provide reliable network endpoint
  - IP address
  - DNS name
  - Port
- Expose pods to the outside world
  - **NodePort** (cluster-wide port)
  - **LoadBalancer** (cloud-based)
- Use **end point** object to track pods
- Use **label selectors** to do their magic

Services in Action



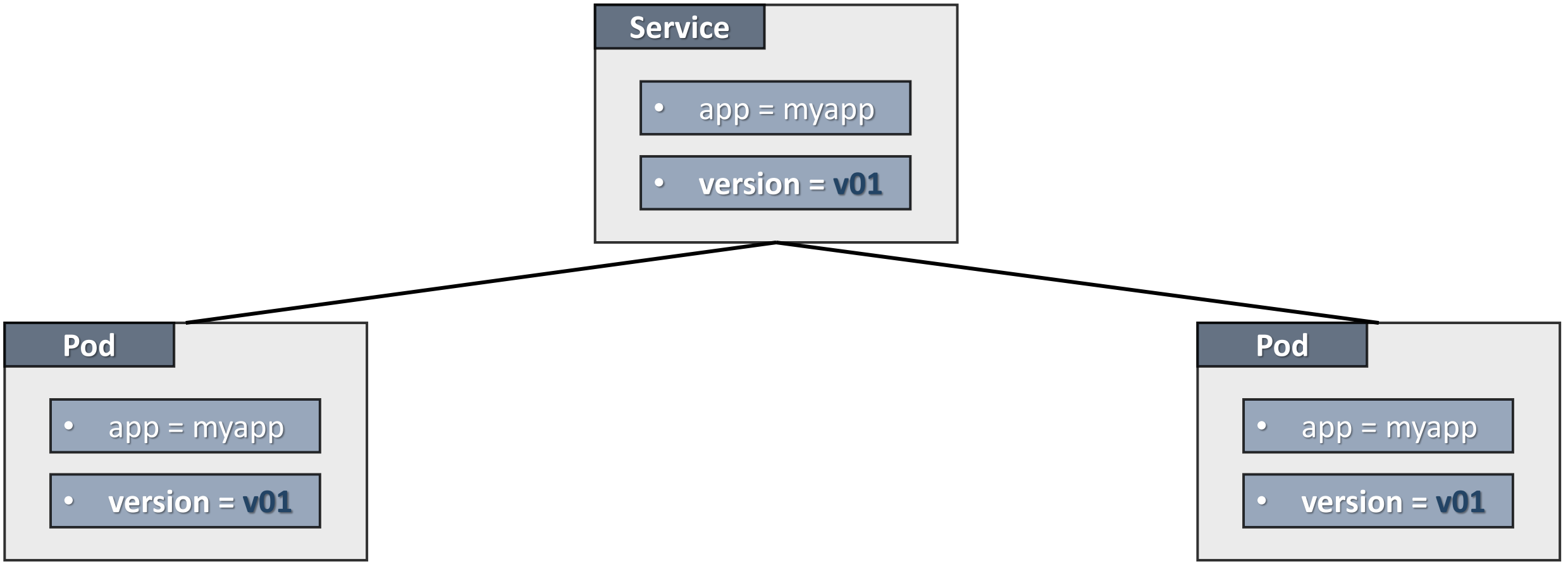
Initial deployment – one pod with version = v01

Services in Action (Scale Up)



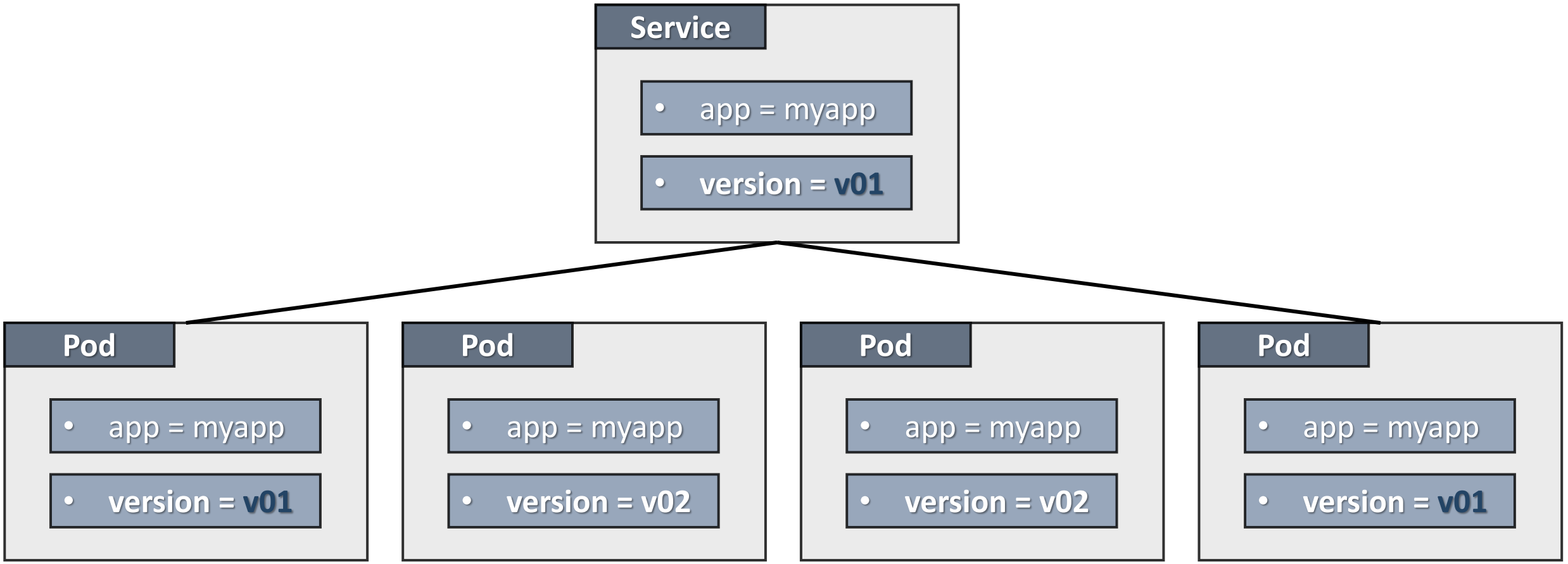
Scale up – two more pods with version = v01

Services in Action (Scale Down)



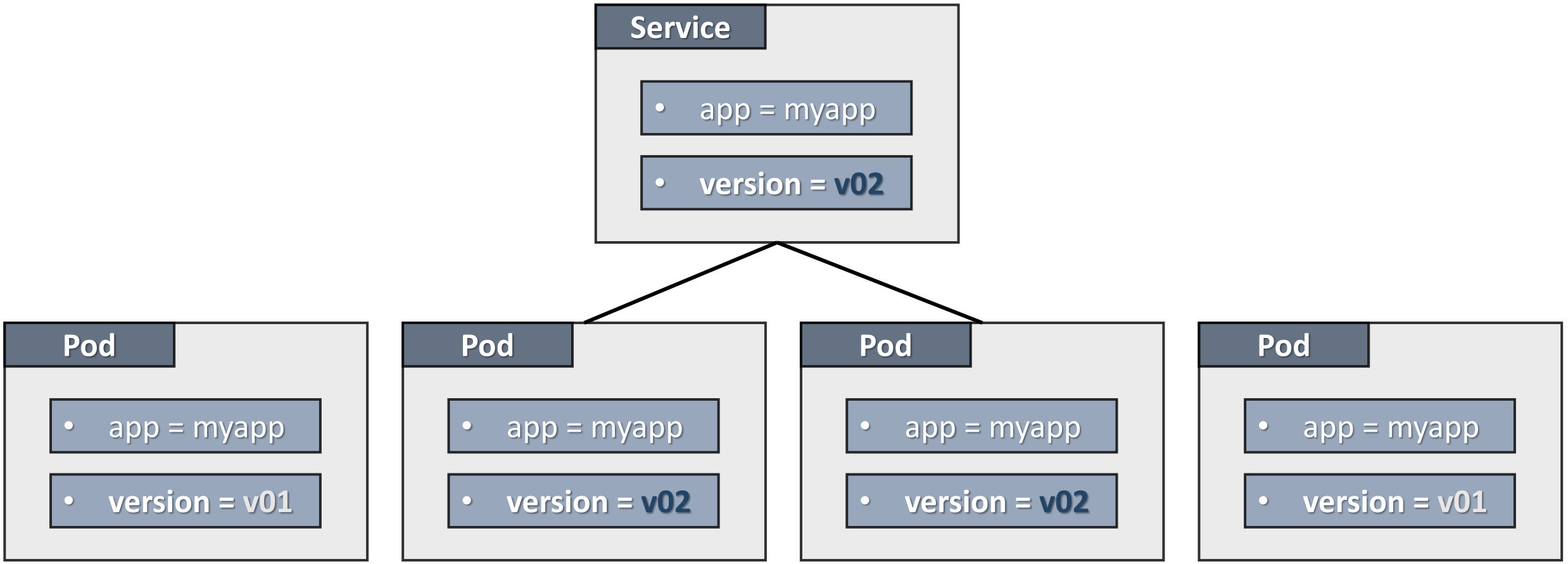
Scale down – remove one pod and end up with two pods with version = v01

Services in Action (App Update)



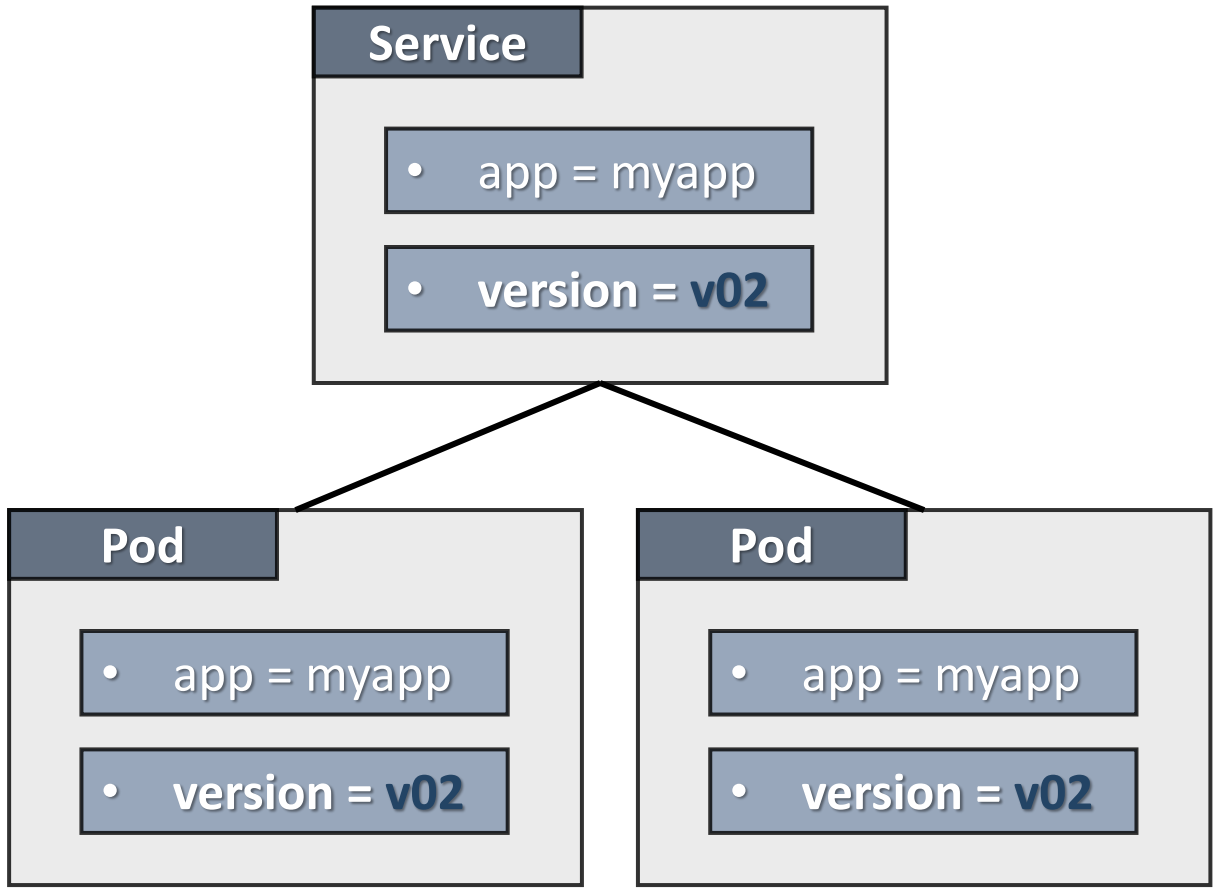
Next step – add two more pods with version = v02

Services in Action (App Update)



Next step – we update the service to look for version = v02

Services in Action (App Update)



Finally, all pods with version = v01 are destroyed

Work with Kubernetes

Distributions, installation options, and tools

Kubernetes Distribution

- A software package that provides a pre-built version of Kubernetes
- Most distributions also offer installation tools or additional software integrations
- On-premise
  - **Minikube**, **MicroK8s**, **K3s**, **k0s**, **openSUSE Kubic**, **OpenShift**, **VMware Tanzu** …
- Cloud-based
  - **Azure Kubernetes Services** (**AKS**), **Elastic Container Service for Kubernetes** (**EKS**), **Google Kubernetes Engine** (**GKE**), …
- Usually, cloud versions are a few versions behind

Installation Scenarios and Tools

- Installation methods
  - **Localhost** (**Minikube** or **Kind**)
  - **On-Premise** (VMs, Bare Metal)
  - **Cloud** (Hosted Solutions, Turnkey Solutions, Bare Metal)
- Configurations
  - **All-in-One Single Node** and different **Multi Node** options
- Installation tools
  - **kubeadm**, **KubeSpray**, **Kops**

Minikube

- Easiest and recommended way for a **local all-in-one cluster**
- Requirements
  - **kubectl**
  - **Hypervisor** (VirtualBox, Hyper-V, KVM, xhyve, VMware Workstation/Fusion)
  - **VT-x**/**AMD-v** enabled CPU
  - Internet connection on the first run
- Supports **Linux**, **macOS**, and **Windows**
- Provides **docker-machine**-like experience, but for **Kubernetes**

kubectl

- Controls the Kubernetes cluster manager
- Expects a file named **config** in the **$HOME/.kube** directory
- Other files can be specified by setting the **KUBECONFIG** environment variable or by setting the **--kubeconfig** flag
- The syntax is

```
kubectl [command] [TYPE] [NAME] [flags]
```

- Where **command** is the operation (**run**, **get**, etc.) and **type** is the resource (**pod**, **service**, etc.). Note that **name** is case-sensitive

Dashboard

- A web-based Kubernetes user interface
- Deployment of containerized applications to a cluster
- Troubleshooting containerized application
- Managing the cluster resources

Kubernetes in Action

A short demonstration using Minikube

Most recent version can be downloaded from **https://github.com/shekeriev/journey-vmware-tanzu**

2021.07.06