

Kubernetes 101

Edmund Shee – Developer advocate
Mofe Salami – Developer advocate



Agenda

Recap Container Basics & Docker:

- Container architecture
- Advantages
- Docker ecosystem

Kubernetes:

- Kubernetes Basics
- Kubernetes Architecture
- Kubernetes Concepts

Hands-on Workshop

Before we begin...

- Create an IBM Cloud Account:

<https://www.softlayer.com/IBMGEP-application/>

Enter required fields

For Startup Name – use your first and last name

For Category pulldown select: Development

For Affiliate pulldown select: Kubernetes 101

- Install the IBM Cloud CLIs and login:

http://bit.ly/ibm_cli

\$ bx login

- Provision a cluster:

\$ bx cs cluster-create --name <name-of-cluster>

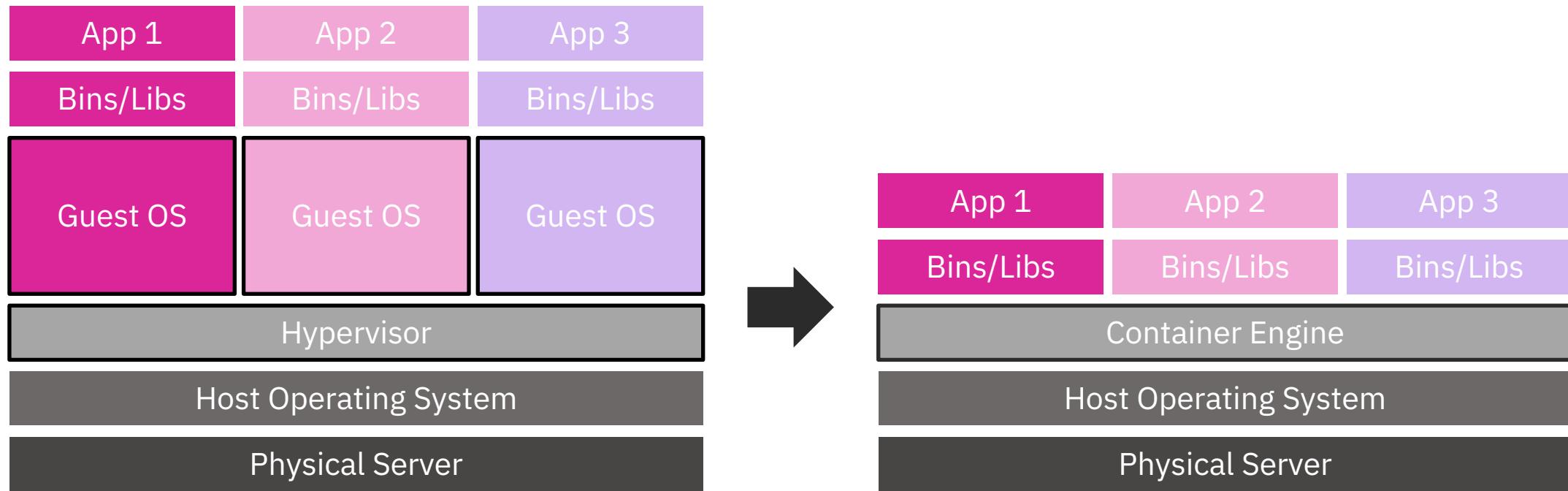


Recap Container Basics
& Docker

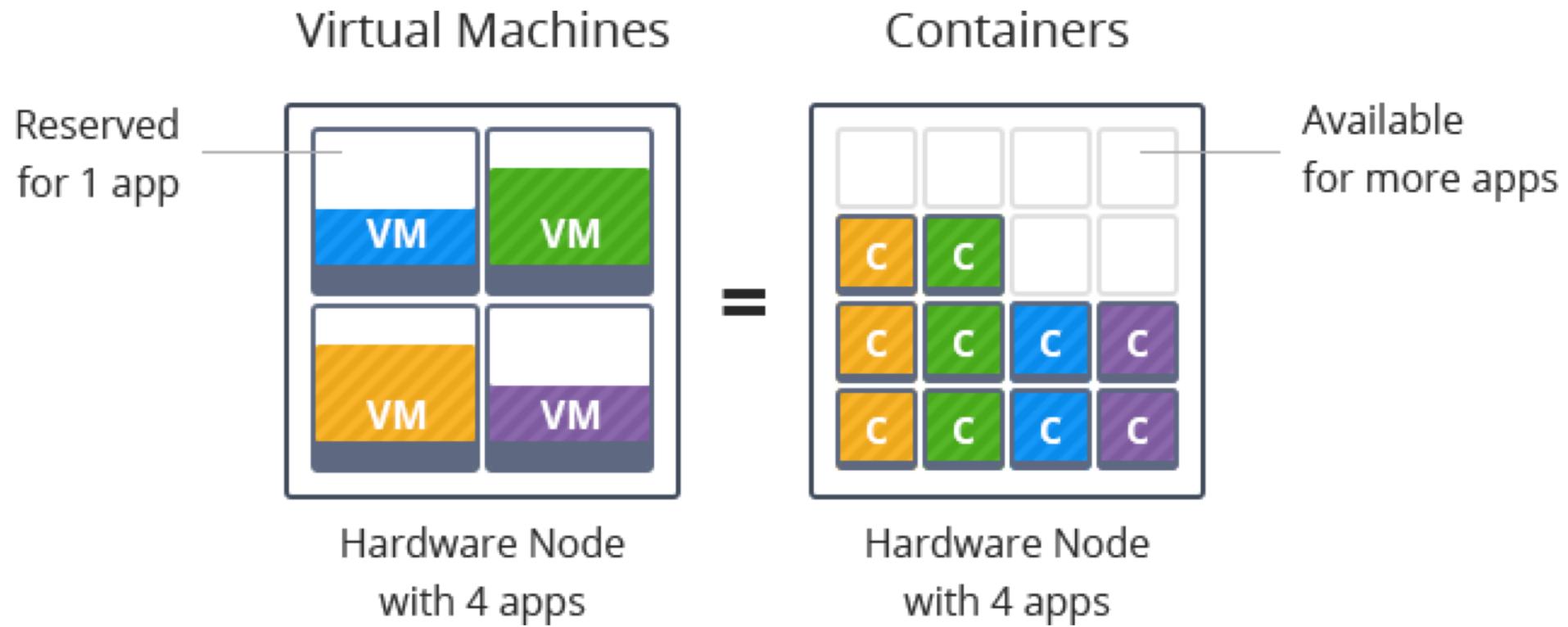
Container Basics

- A container contains!
- Virtualizing the hardware OS subsystems
- Interaction with a single OS

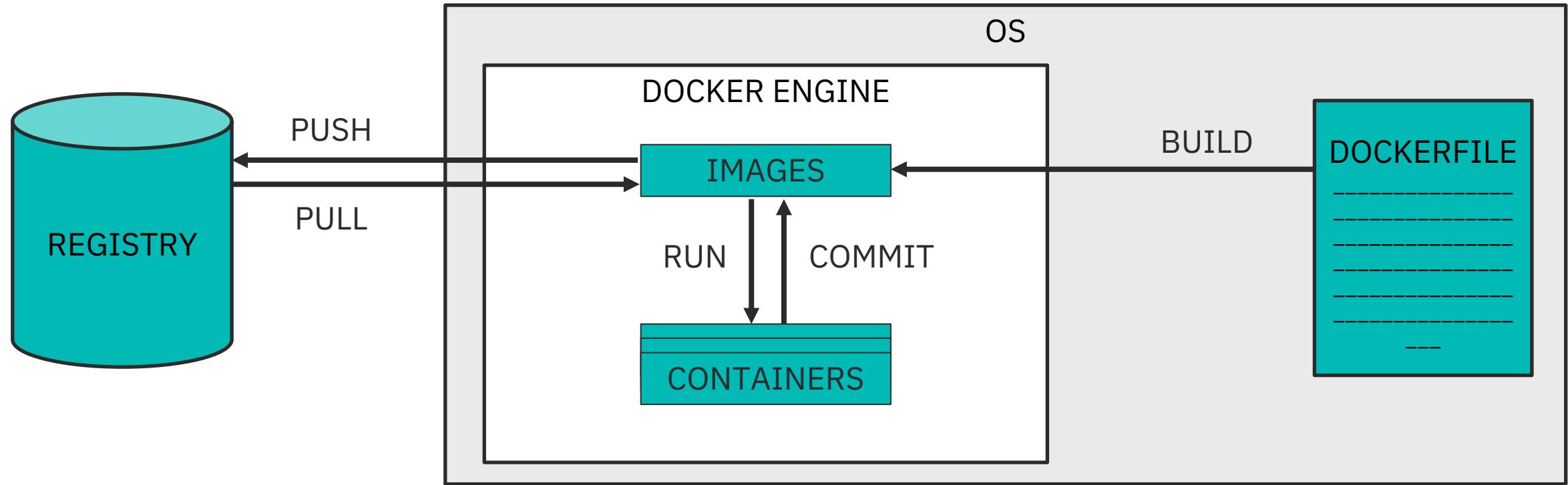
VMs vs. Container Virtualization



Advantages



Docker Architecture



Kubernetes



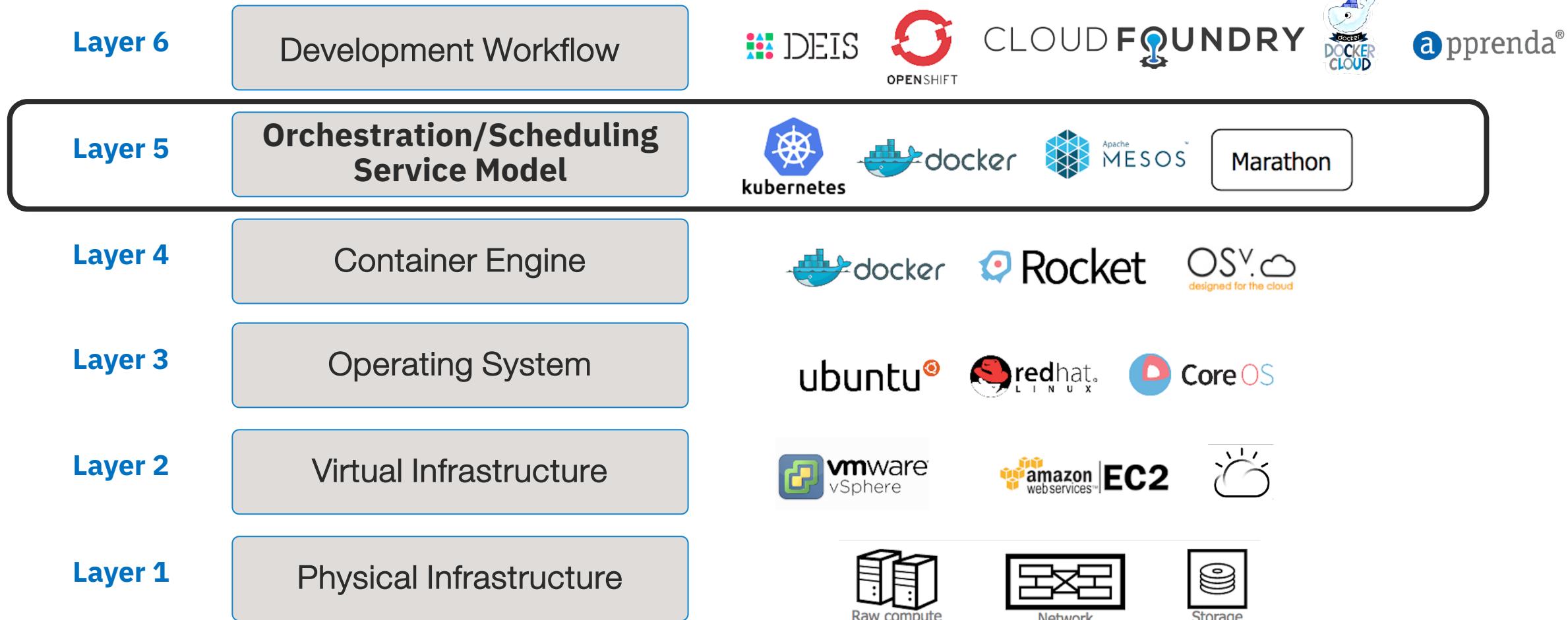
Limitations of Docker

Problem: *I have **multiple applications**, that I wish to run across **multiple servers** with **high availability**. Docker is great but its not that smart!*

- Docker is only aware of the containers running on the machine it runs on
- Horizontal scaling in Docker does not exist
- Docker does not repair applications which become unhealthy and cannot



Container Stack





Kubernetes Capabilities



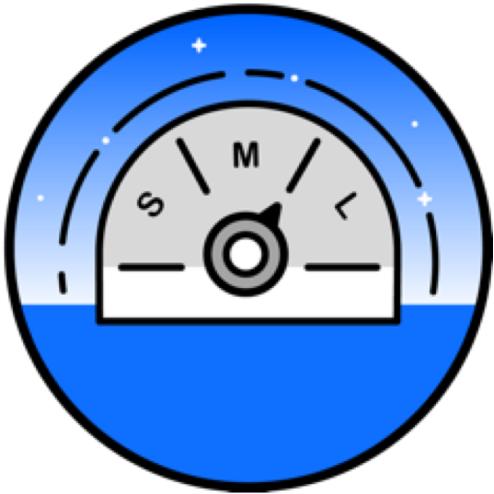
Intelligent Scheduling

- Automatically places containers based on required resources
- Supports mixed workloads to drive increased utilization



Self-healing

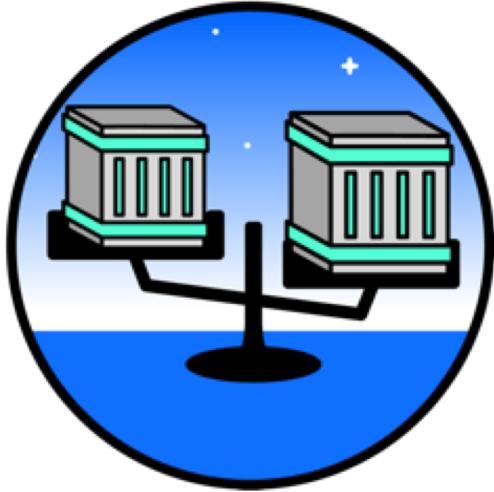
- Restarts containers that fail
- Replaces and reschedules containers when nodes die
- Kills containers that don't respond to your user-defined health check



Horizontal scaling

- Scale your application with a simple command
- Automatic scaling based on real-time usage

Service discovery and load balancing



- Simple discovery of services through a single DNS name
- Manage access to container applications through IP address or HTTP route.
- Automatically load balance traffic and route around failure



Automated rollouts and rollbacks

- Roll out changes to your application or its configuration, while monitoring application health to ensure things stay up
- If something goes wrong, Kubernetes will rollback the change for you

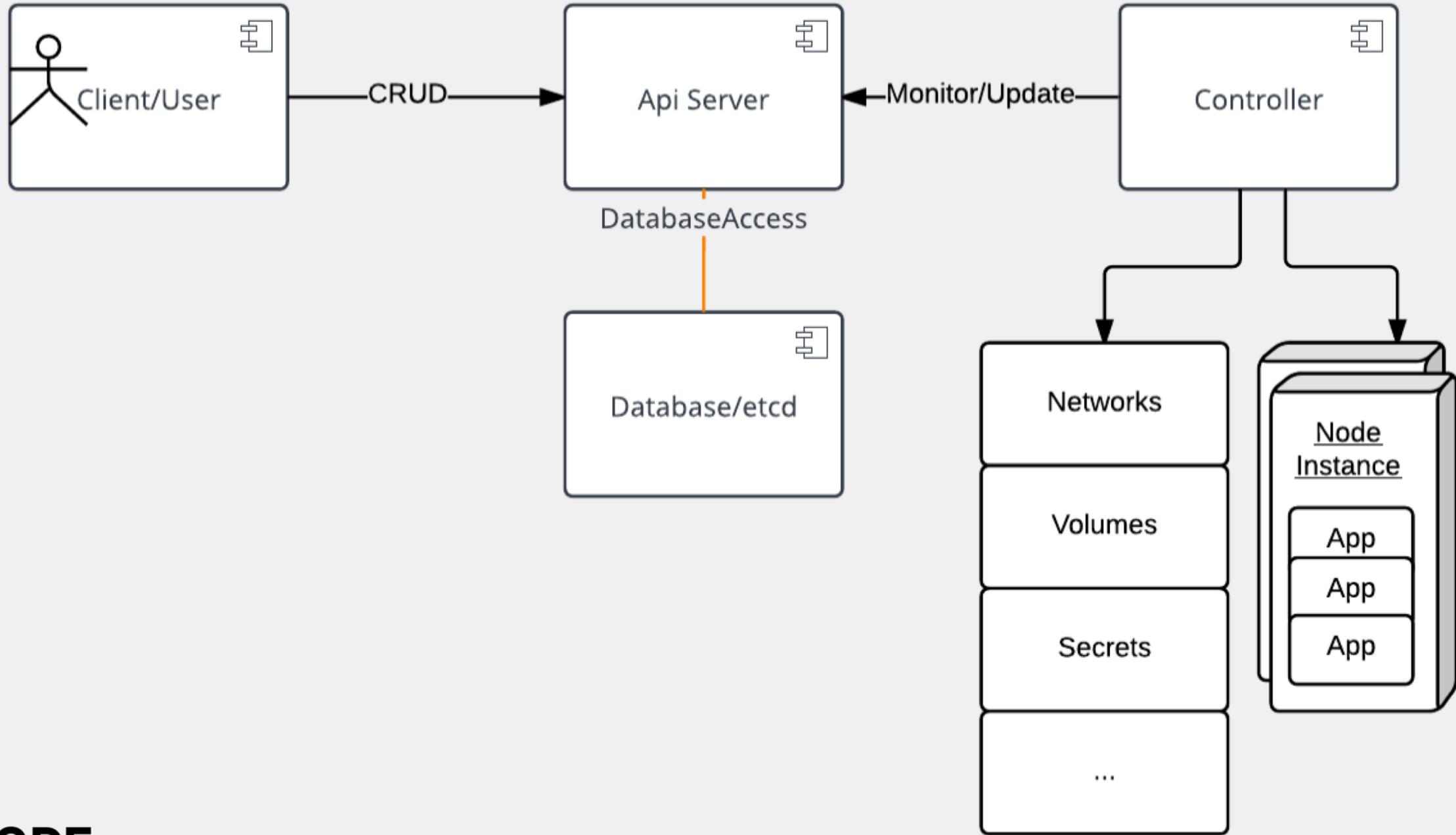


Secret and configuration management

- Safely store application credentials and secrets
- Deploy and update secrets and application configuration without rebuilding your image and without exposing secrets in your stack configuration.



Kubernetes Concepts





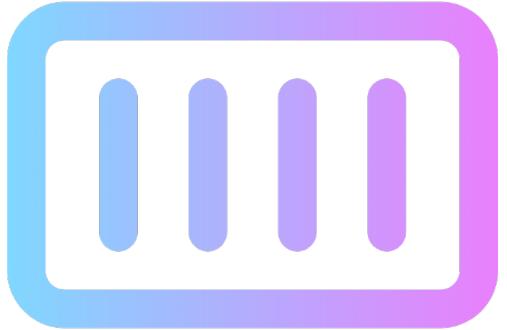
Master Node

- Controls and manages the cluster
- Kubectl (command line)
- REST API (communication with workers)
- Scheduling and replication logic



Worker Nodes

- Hosts the K8s services
- Kubelet (K8s agent that accepts commands from the master)
- Kubeproxy (network proxy service responsible for routing activities for inbound or ingress traffic)
- Runs Docker engine



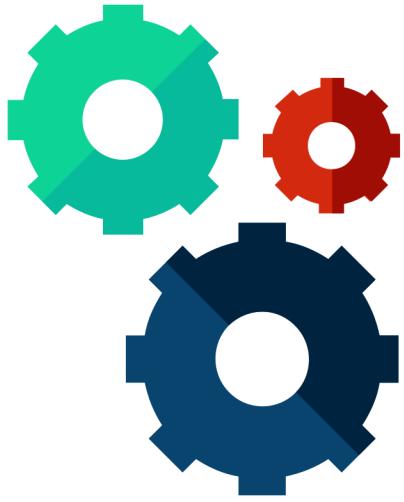
Pods

- Smallest deployment unit in K8s
- Collection of containers that run on a worker node
- Each has its own IP
- Pod shares a PID namespace, network, and hostname



Labels

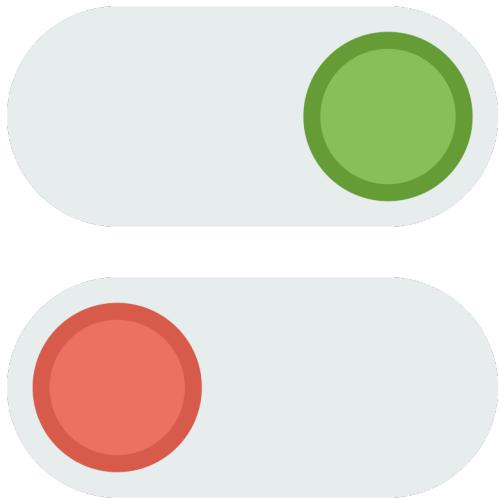
- Metadata assigned to K8s resources
- Key-value pairs for identification
- Critical to K8s as it relies on querying the cluster for resources that have certain labels



Services

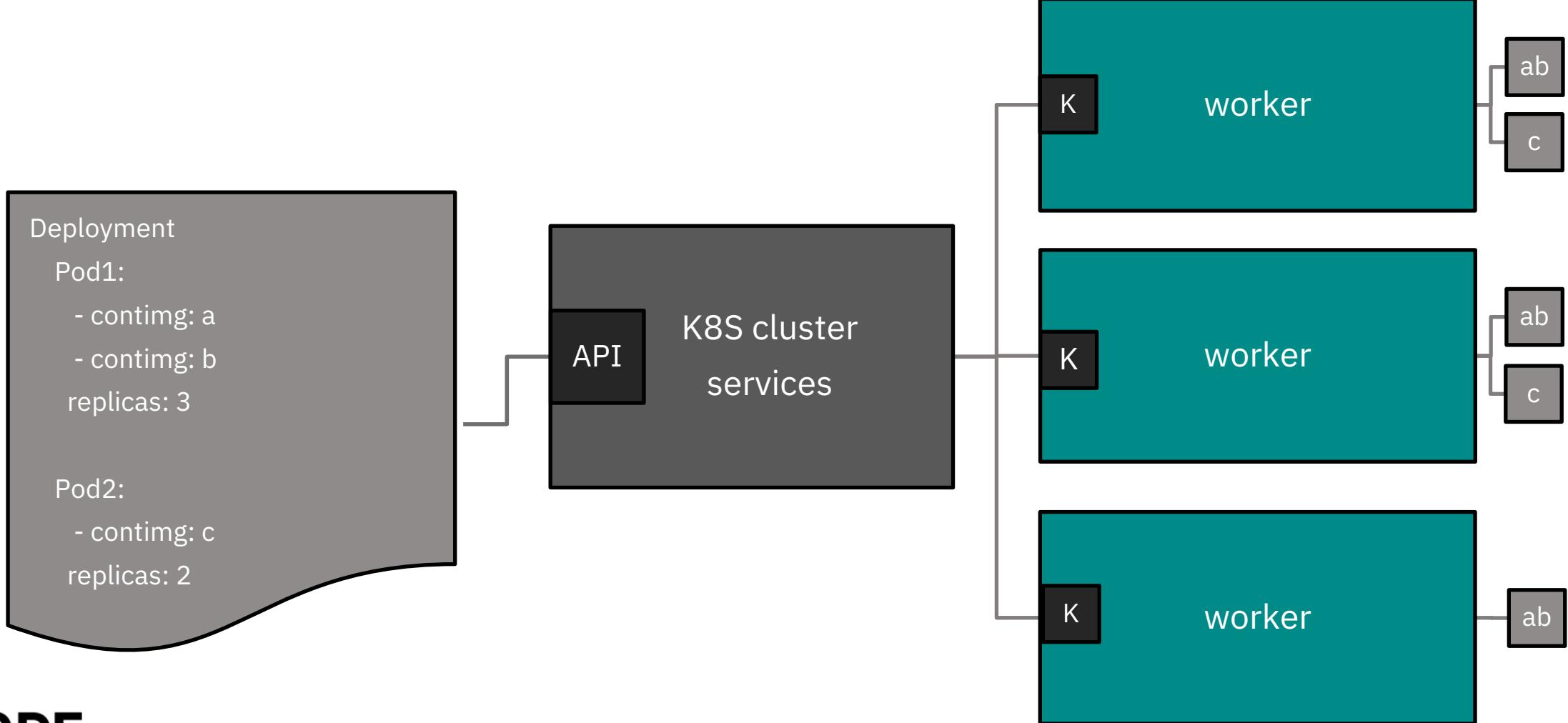
- Collections of pods exposed as an endpoint
- Information stored in the K8s cluster state and networking info propagated to all worker nodes

Replication Controllers

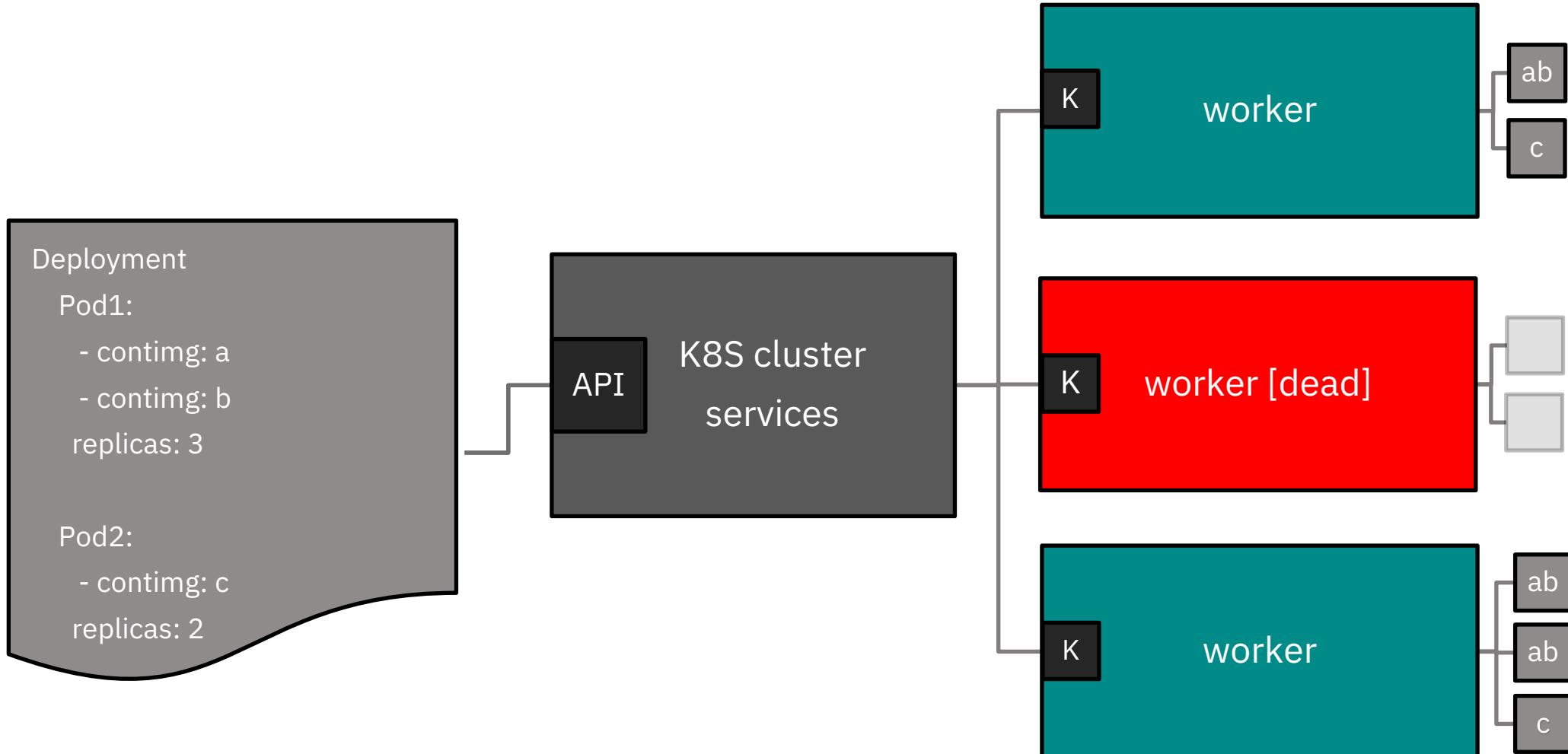


- Ensures availability and scalability
- Maintains the number of pods as requested by user
- Uses a template that describes specifically what each pod should contain

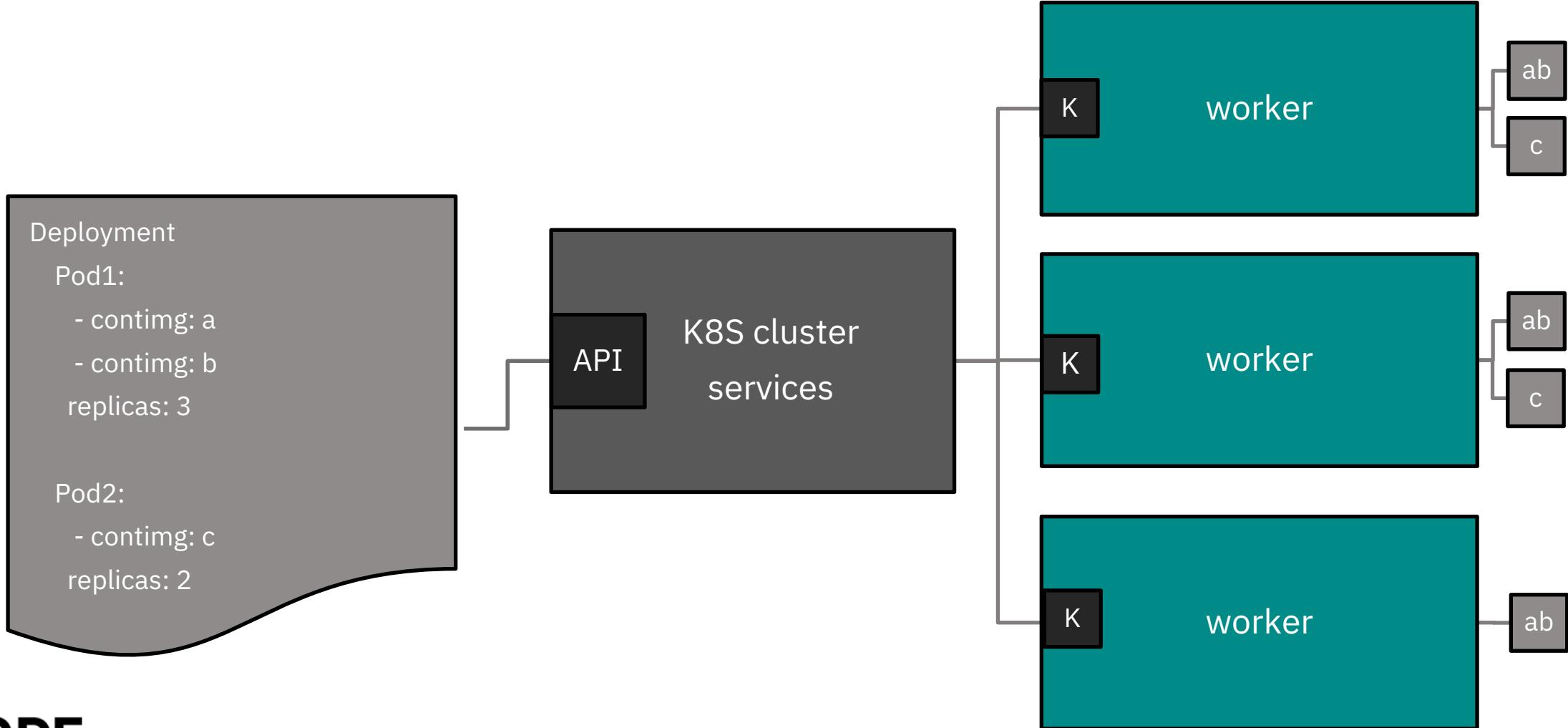
Kubernetes Cluster: Deployment



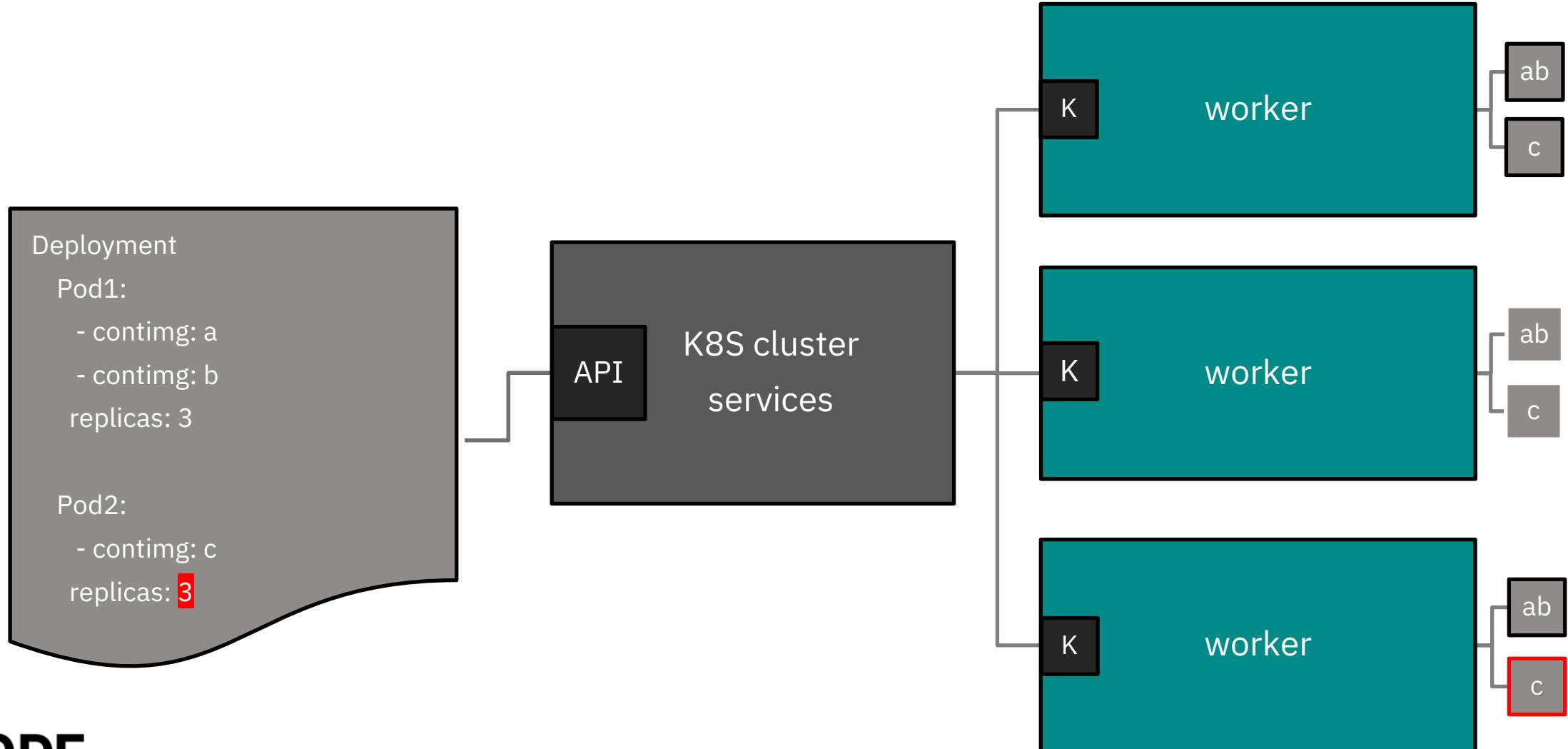
Kubernetes Cluster: Monitoring



Kubernetes Cluster: Scaling



Kubernetes Cluster: Scaling



Kubectl CLI

Create the configuration specified in *configuration.yml* file

- `kubectl create -f configuration.yml`

Show all the pods deployed in k8s in the default name space

- `kubectl get pods`

Show the details of the pod *pod_name*

- `kubectl describe pod pod_name`

Display the logs corresponding to the pod *pod_name* in the default name space

- `kubectl logs pod_name`

Enter an interactive shell inside pod *pod_name*

- `kubectl exec -ti pod_name bash`

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

Workshop time!



<https://github.com/IBMCODELondon/kube101>

