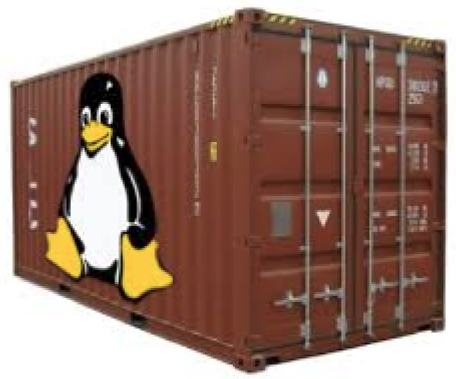
What is container ?

Containers, in short, contain applications in a way that keep them isolated from the host system that they run on. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package



What is container ?

Big internet companies like Google have been utilizing containers for nearly a decade.

The benefits we enjoy include:

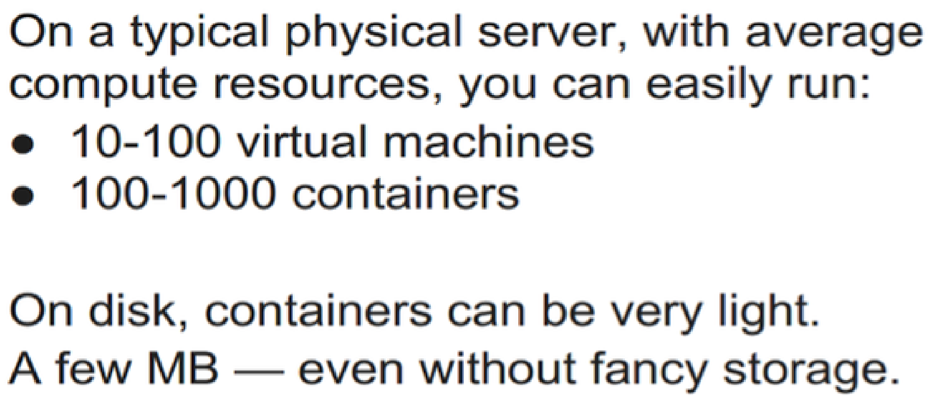
- Very fast provisioning
- Simple, high availability
- Smooth scaling
- Machine-precision consistency
- Better performance

What is container ?

Big internet companies like Google have been utilizing containers for nearly a decade.

The benefits we enjoy include:

- Very fast provisioning
- Simple, high availability
- Smooth scaling
- Machine-precision consistency
- Better performance



On a typical physical server, with average compute resources, you can easily run:

- 10-100 virtual machines
- 100-1000 containers

On disk, containers can be very light.
A few MB — even without fancy storage.

image credit: Jerome Petazzoni from dotCloud)

What is container ?

Everything at Google runs in containers

- Gmail, Web Search, Maps

- Even GCE itself: Vms in containers

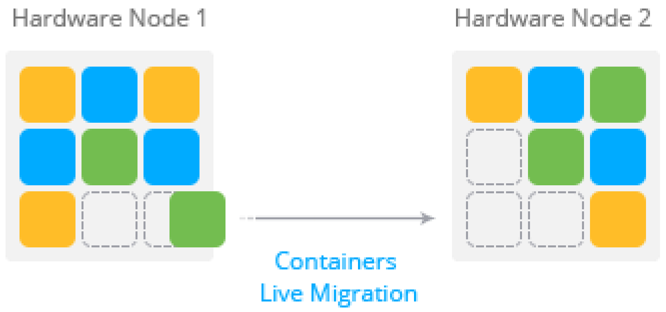
They launch over 2 billion containers per week

What is container ?

Everything at Google runs in containers

- Gmail, Web Search, Maps

- Even GCE itself: Vms in containers



They launch over 2 billion containers per week

The need for container orchestration

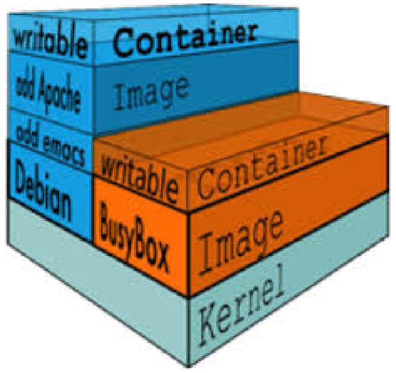
Containers are becoming the standard unit of deployment

Each container image has

- Code
- Binaries
- Configuration
- Libraries
- Frameworks
- Runtime

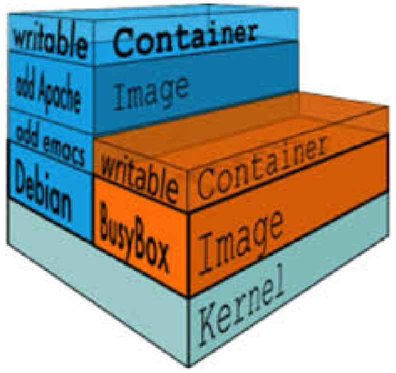
The need for container orchestration

Docker has solved the problem of packaging, deploying and running containerized applications



The need for container orchestration

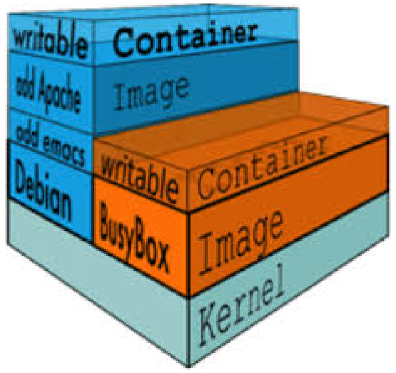
Docker has solved the problem of packaging, deploying and running containerized applications



Docker is great for managing a few containers running on a fewer machines

The need for container orchestration

Docker has solved the problem of packaging, deploying and running containerized applications



Docker is great for managing a few containers running on a fewer machines

Production applications deal with dozens of containers running on hundreds of machines

The need for container orchestration

Tools are evolving to manage the new datacenter infrastructure

- Docker Swarm
- Kubernetes
- Mesosphere DC/OS

Manage the lifecycle of containerized applications running in production

Automate the distribution of applications

Ensure higher levels of utilization and efficiency

What is Kubernetes?

The Kubernetes project was started by Google in 2014.

Kubernetes (from κυβερνήτης: Greek for "helmsman" or "pilot")

Its development and design are heavily influenced by Google's Borg system

- Kubernetes is a platform for hosting Docker containers in clustered environment
- Provides container grouping, load balancing, auto-healing, scaling
- Supports multiple cloud and bare-metal environments easily roll out new versions of application containers

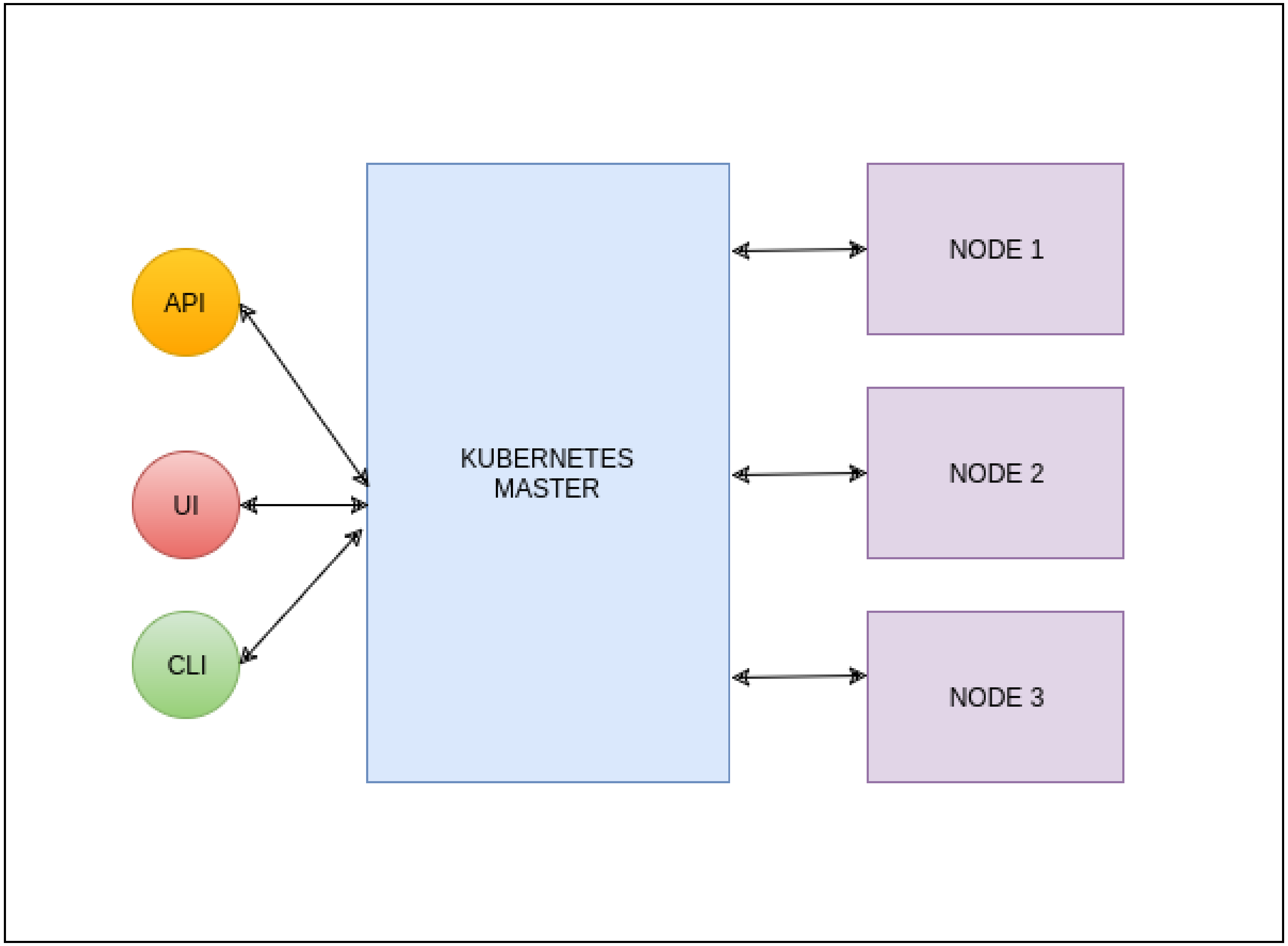
Cluster

A cluster is a group of nodes, they can be physical servers or virtual machines that has the Kubernetes platform installed.

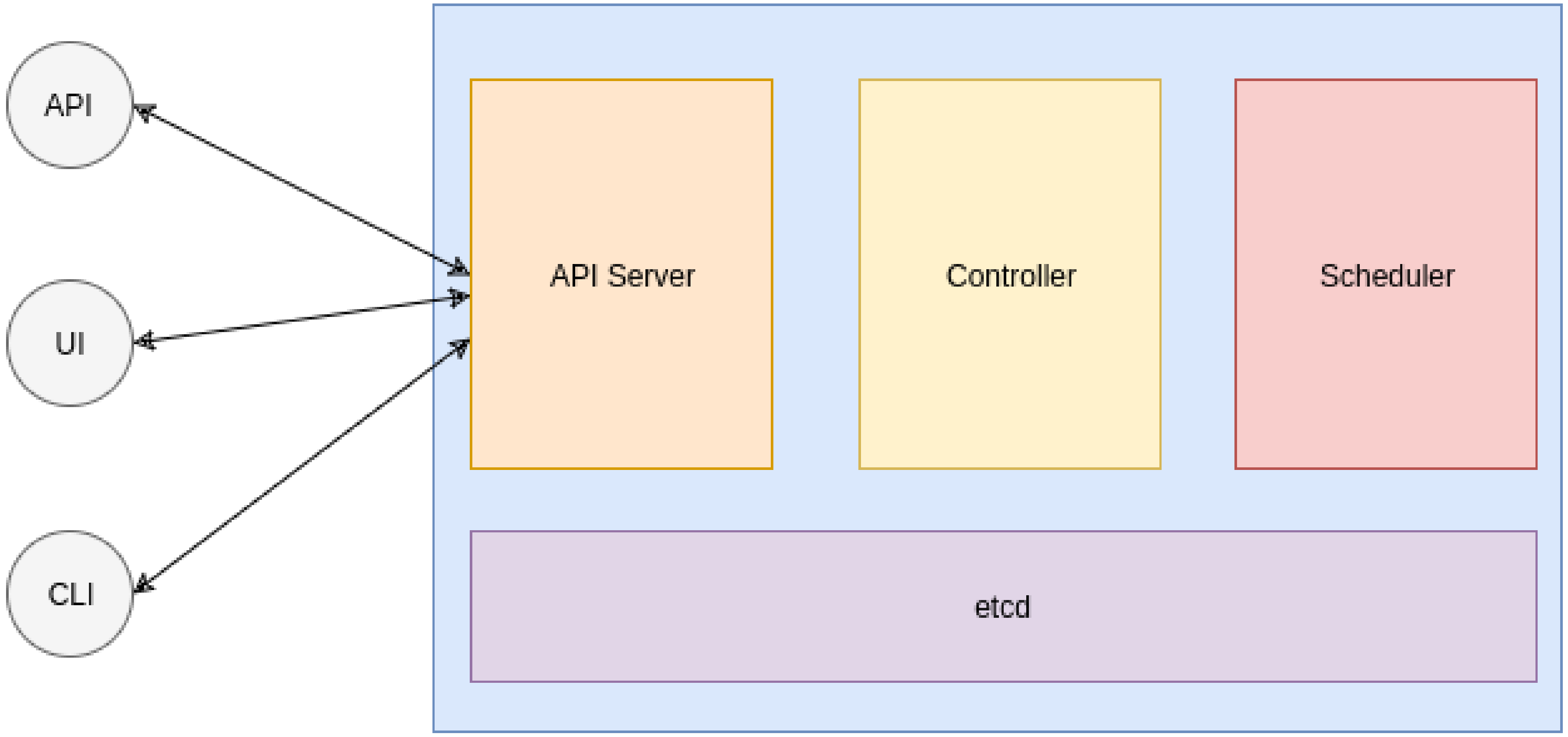
The controlling unit in a Kubernetes cluster is called the master server. It serves as the main management contact point for administrators, and it also provides many cluster-wide systems for the relatively dumb worker nodes.

The master server runs a number of unique services that are used to manage the cluster's workload and direct communications across the system.

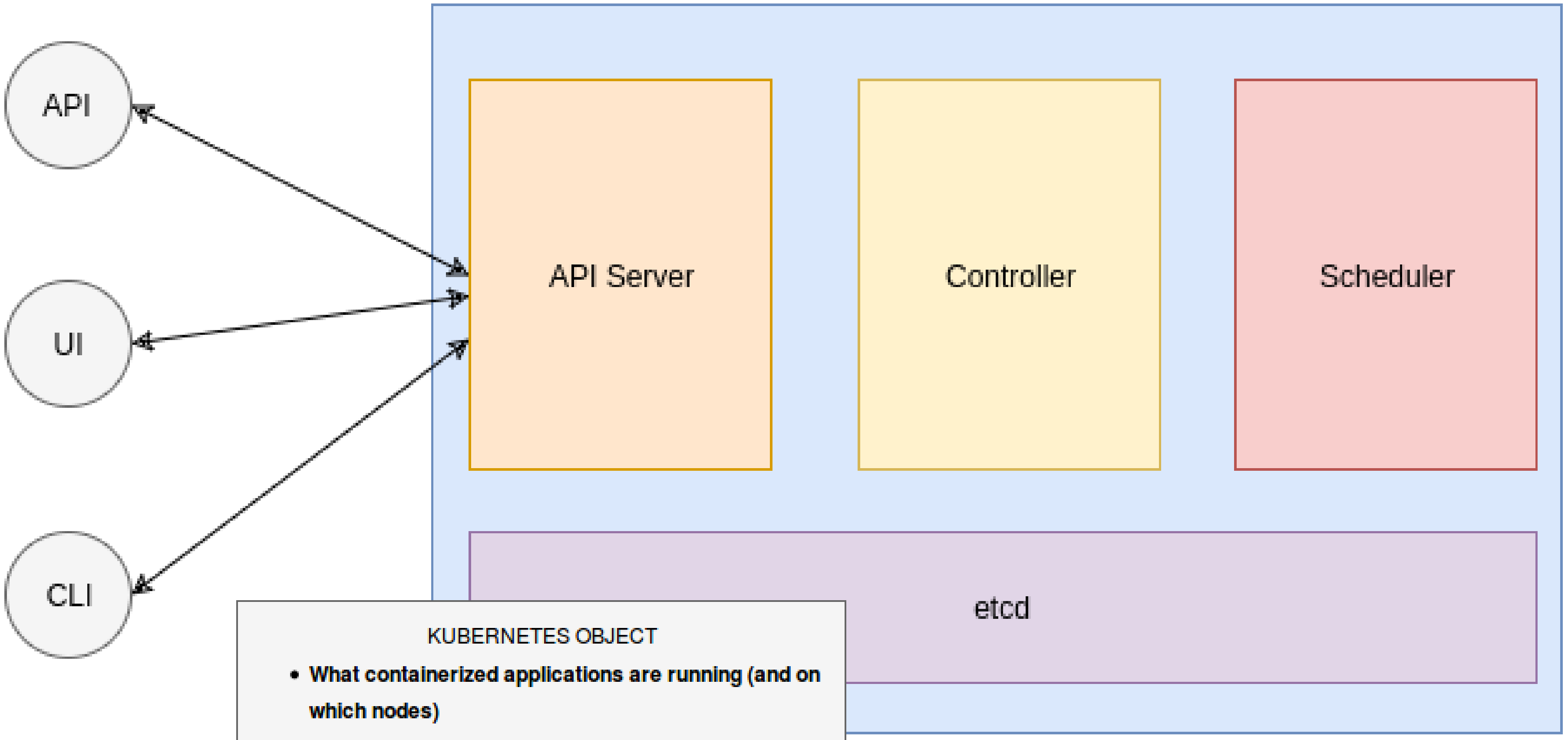
Cluster



Master



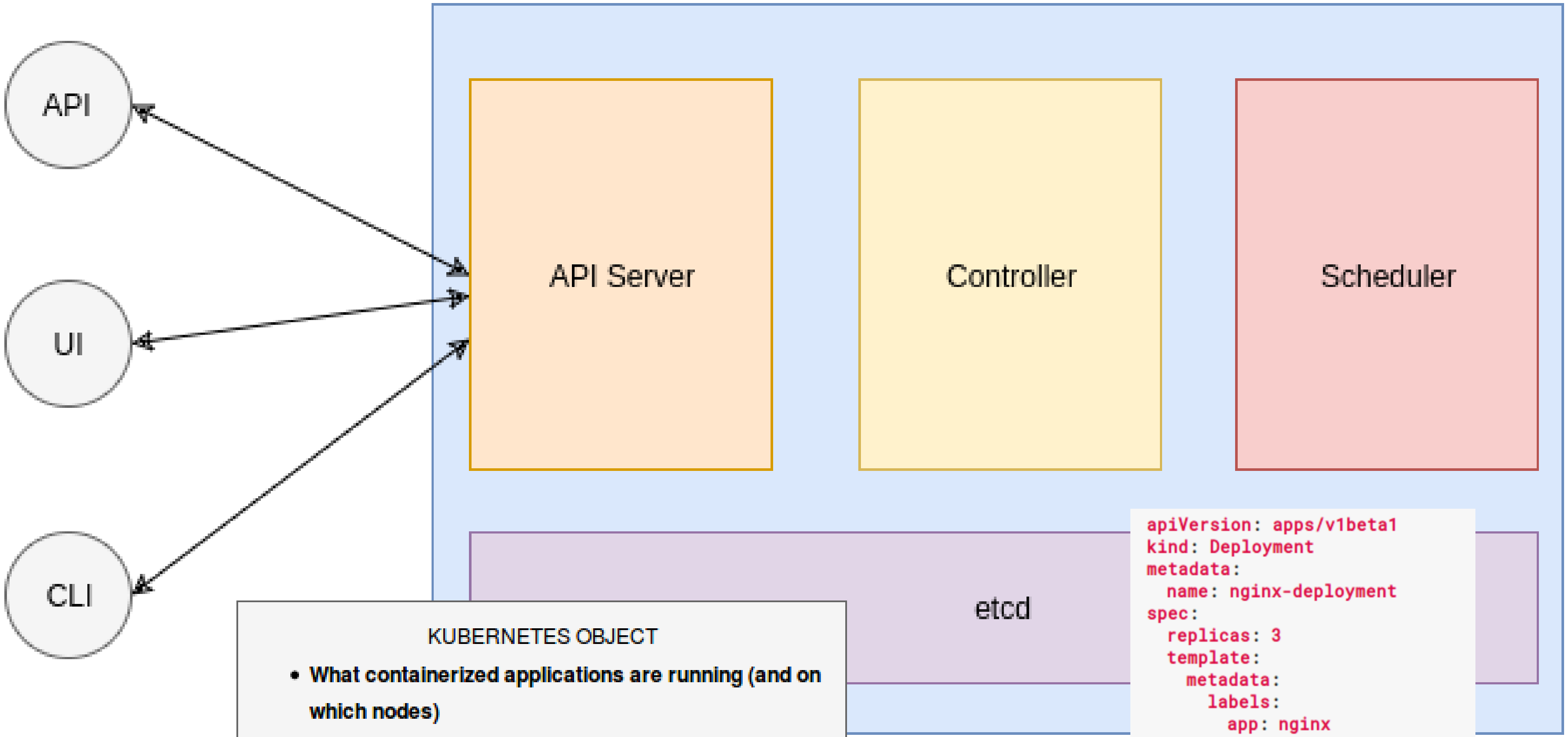
Master



KUBERNETES OBJECT

- What containerized applications are running (and on which nodes)
- The resources available to those applications
- The policies around how those applications behave, such as restart policies, upgrades

Master



KUBERNETES OBJECT

- What containerized applications are running (and on which nodes)
- The resources available to those applications
- The policies around how those applications behave, such as restart policies, upgrades

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

Namespaces

Enable you to manage different environments within the same cluster.

You can create multiple kubernetes objects without worrying about them affecting each other.

A mechanism to attach authorization and policy to a subsection of the cluster.

Kubernetes starts with two initial namespaces:

- **default** The default namespace for objects with no other namespace
- **kube-system** The namespace for objects created by the Kubernetes system

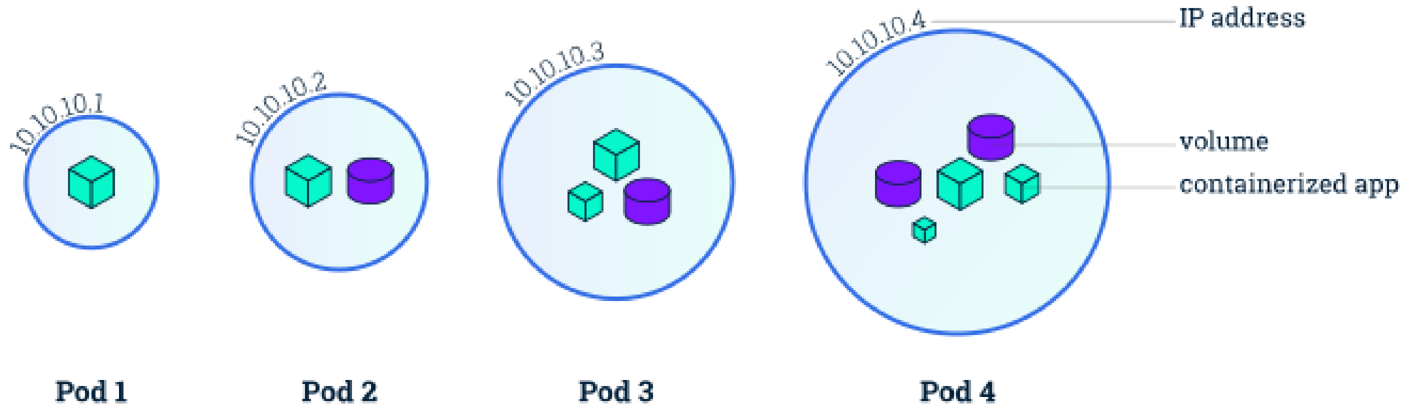
Labels & Selectors

- Key/value pairs associated with Kubernetes objects
- Used to organize and select subsets of objects
- Attached to objects at creation time but modified at any time.
- Labels are the essential glue to associate one API object with other

POD

A group of one or more containers that are always co-located and co-scheduled that share the context

Containers in a pod share the same IP address, ports, hostname and storage



POD

Modeled like a virtual machine -

- Each container represents one process
- Tightly coupled with other containers in the same pod

Pods are scheduled in Nodes

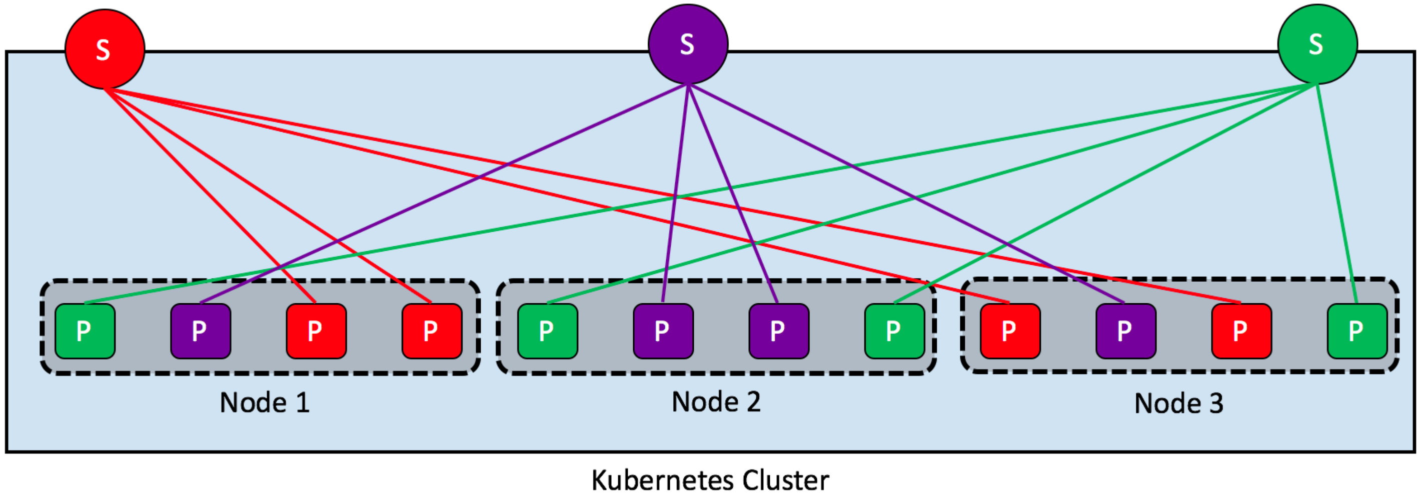
Fundamental unit of deployment in Kubernetes

Replication controller / Replica set

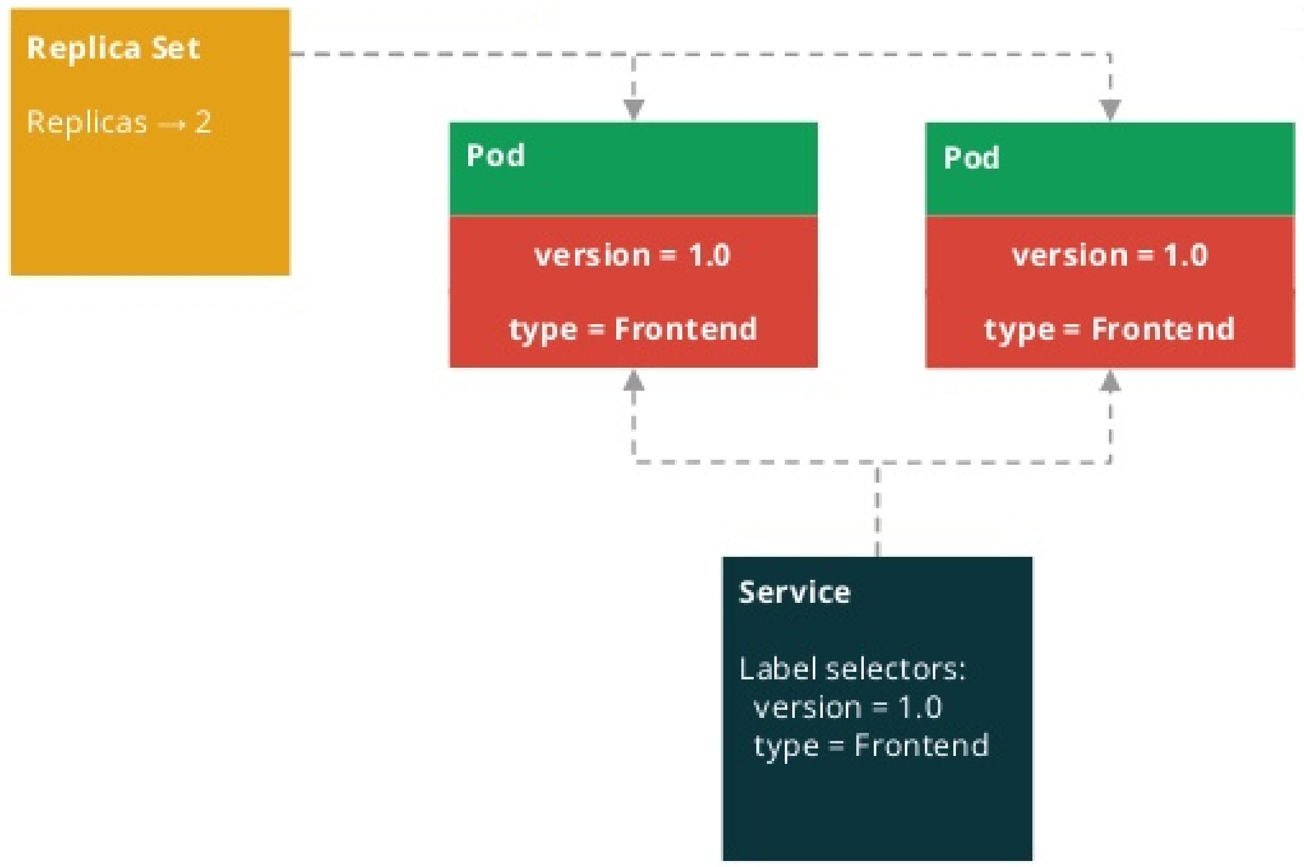
- Ensures that a Pod or set of Pods are always up and available
- Always maintains desired number of Pods
- If there are excess Pods, they get killed
- New pods are launched when they fail, get deleted, or terminated
- Creating a replication controller with a count of 1 ensures that a Pod is always available
- Replication Controller and Pods are associated through Labels

Service

A Kubernetes Service is an abstraction which defines a logical set of Pods running somewhere in your cluster, that all provide the same functionality. When created, each Service is assigned a unique IP address (also called clusterIP).



Summary


Replica Set
Replicas → 2
Pod
version = 1.0
type = Frontend
Pod
version = 1.0
type = Frontend
Service
Label selectors:
version = 1.0
type = Frontend

Summary

- Kubernetes Master runs the API, Scheduler and Controller services
- Each Node is responsible for running one or more Pods
- Pods are the unit of deployment in Kubernetes
- Labels associate one Kubernetes object with the other
- Replication Controller ensures high availability of Pods
- Services expose Pods to internal and external consumers