

Agenda



Challenges Without Container Orchestration

Without Container Orchestration

Orchestration platform is especially beneficial when running microservices in containers

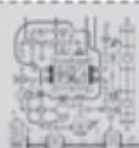
Multiple Services
running inside
containers



Increases the human
cost of running
services



Increases the size of
bills from public cloud
providers



Increases the
complexity of running
something new in
production



Scaling was
difficult



Setting up services
manually



Manual work of
fixing if a node
crashes

Challenges without Container Orchestration

Solution – Orchestration Engine

Kubernetes

1

Auto Scaling

2

Manually configure your Load Balancing settings

3

K8s perform rolling updates to Pods as a whole

4

Shares storage volumes between multiple containers inside the same Pod

5

Provides in-built tools for logging and monitoring



kubernetes

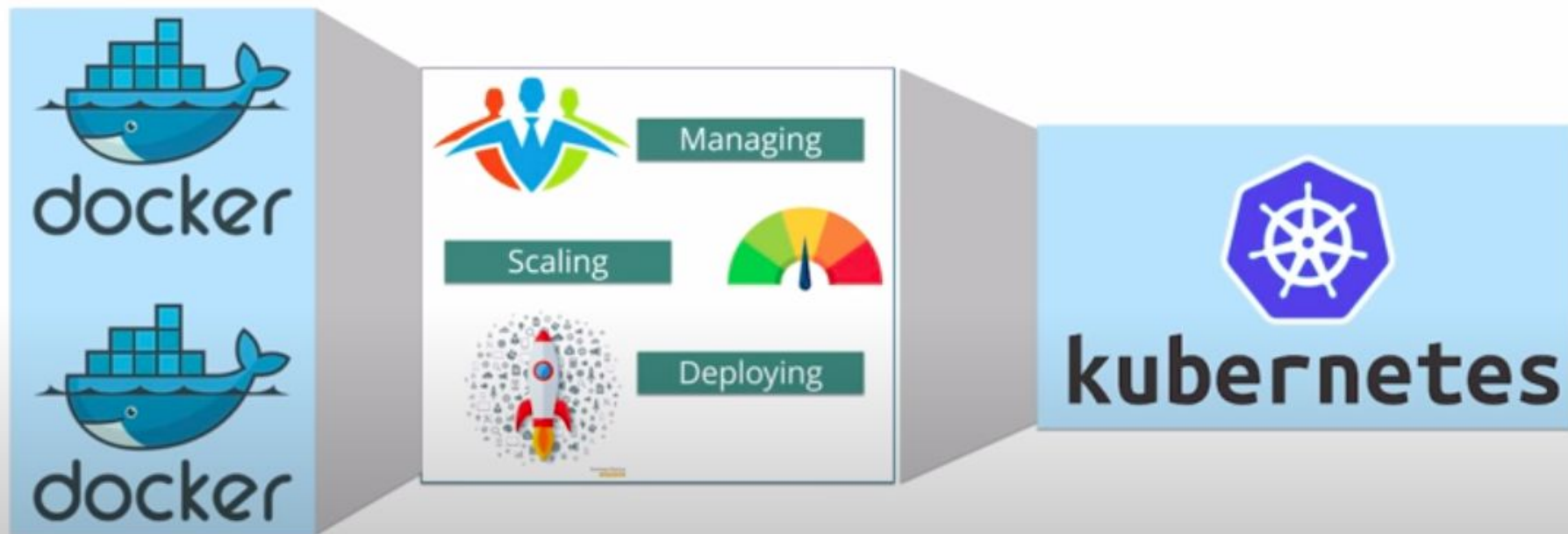


docker docker docker docker docker

Containers need to be managed and connected to the outside world for tasks such as scheduling, load balancing, and distribution, and this is where a container orchestration tool like Kubernetes comes into its own

What is Kubernetes?

Kubernetes handles the work of scheduling containers onto a compute cluster and manages the workloads to ensure they run as the user intended



Kubernetes Features

01

Automated Scheduling

Kubernetes provides advanced scheduler to launch container on cluster nodes

02

Self Healing Capabilities

Rescheduling, replacing and restarting the containers which are died.

03

Automated rollouts and rollback

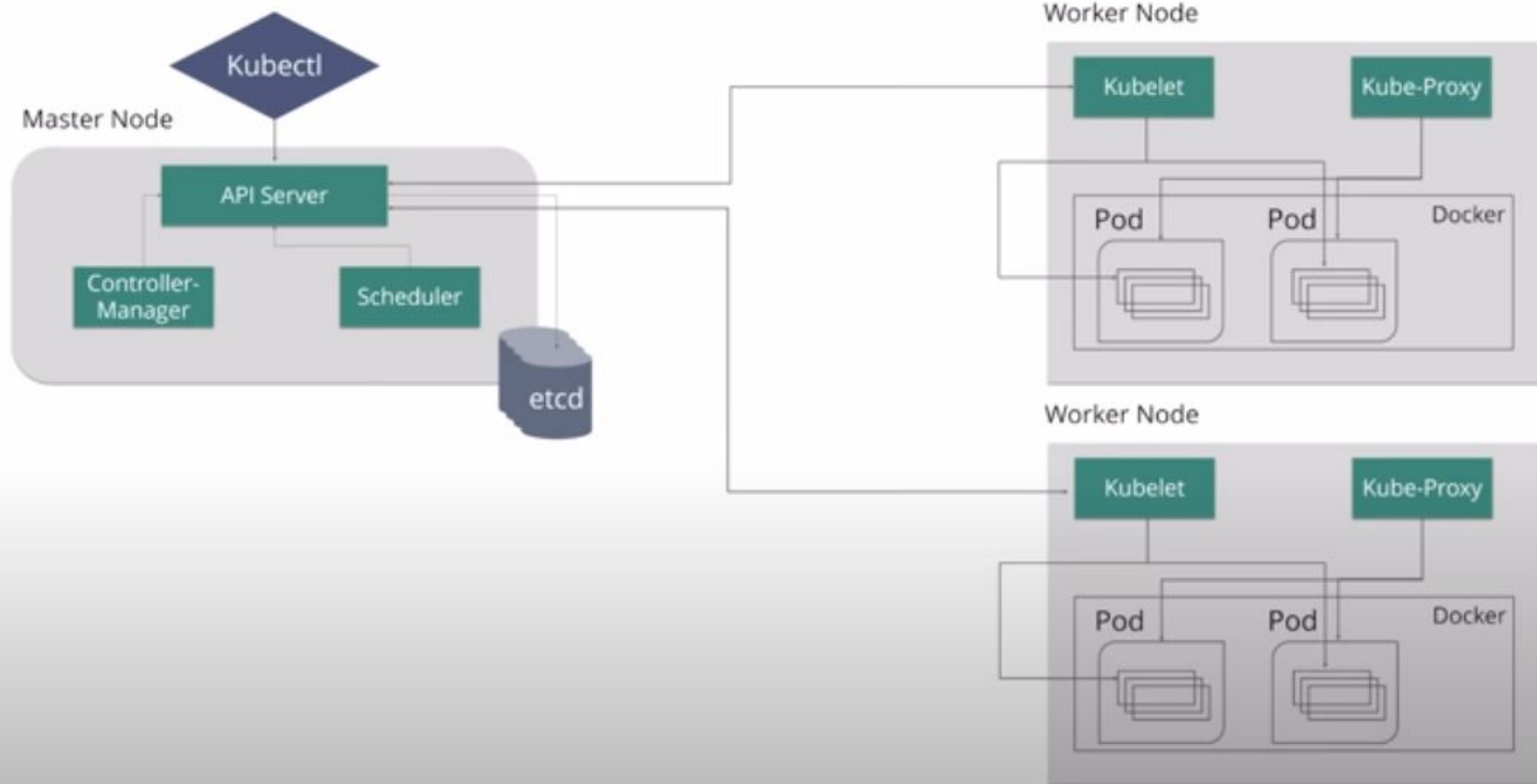
Kubernetes supports rollouts and rollbacks for the desired state of the containerized application

04

Horizontal Scaling and Load Balancing

Kubernetes can scale up and scale down the application as per the requirements

Kubernetes Architecture

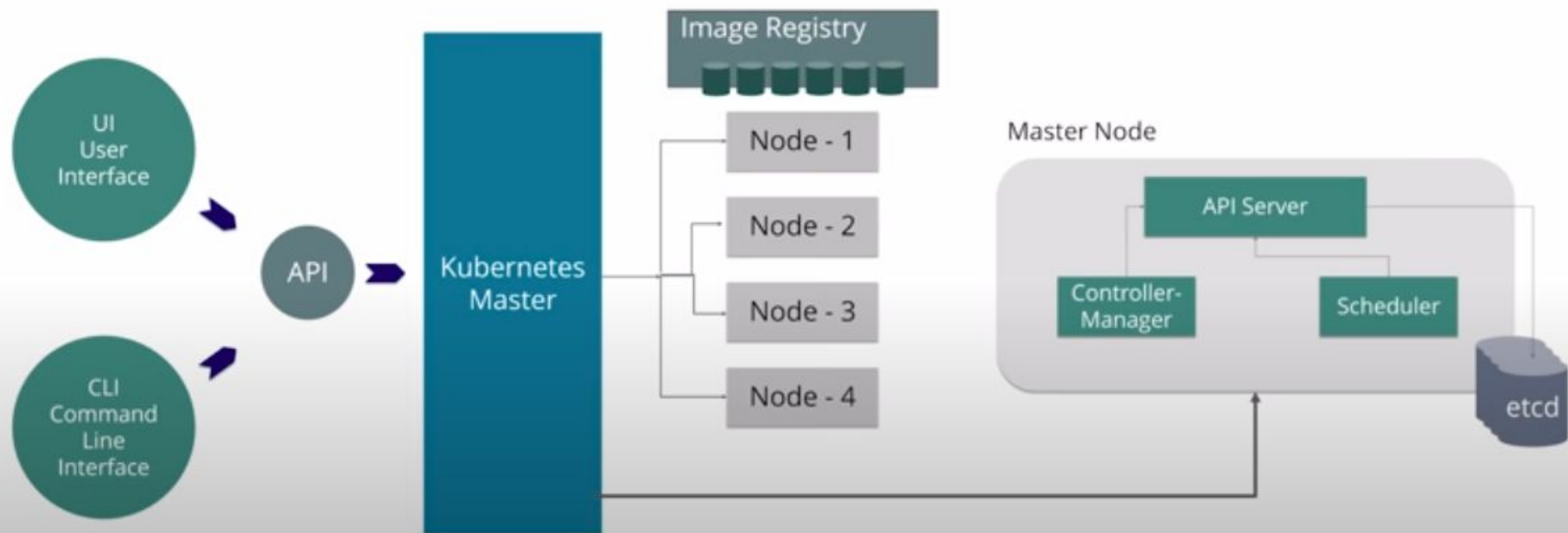




Kubernetes Master

Kubernetes Master Node

The master node is responsible for the management of Kubernetes cluster. This is the entry point of all administrative tasks. The master node is the one taking care of orchestrating the worker nodes, where the actual services are running.

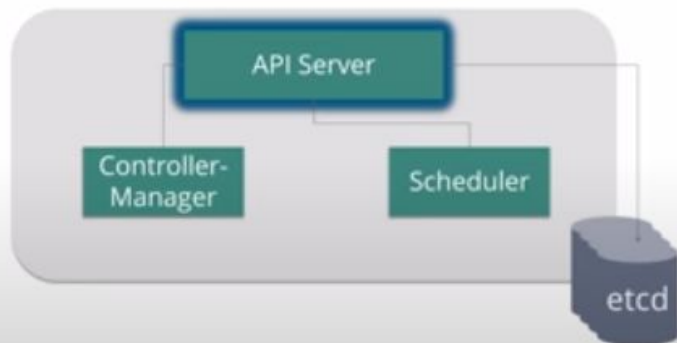


Kubernetes API Server and ETCD

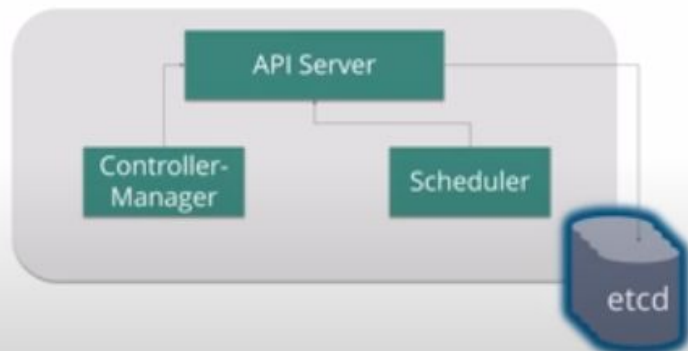
The API server is the entry points for all the REST commands used to control the cluster

It is a simple, distributed, consistent key-value store. It's mainly used for shared configuration and service discovery

Master Node



Master Node

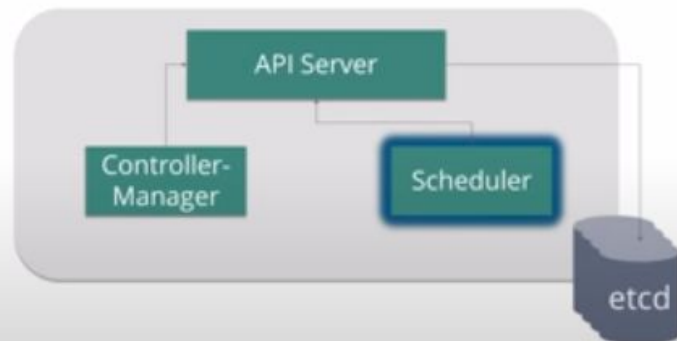


Kubernetes Scheduler and Controller-Manager

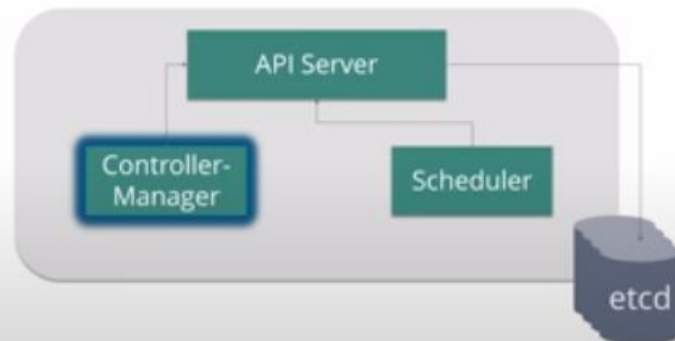
It is responsible for deployment of Pods and services onto the nodes

It allows you to run different controllers inside the master node

Master Node



Master Node

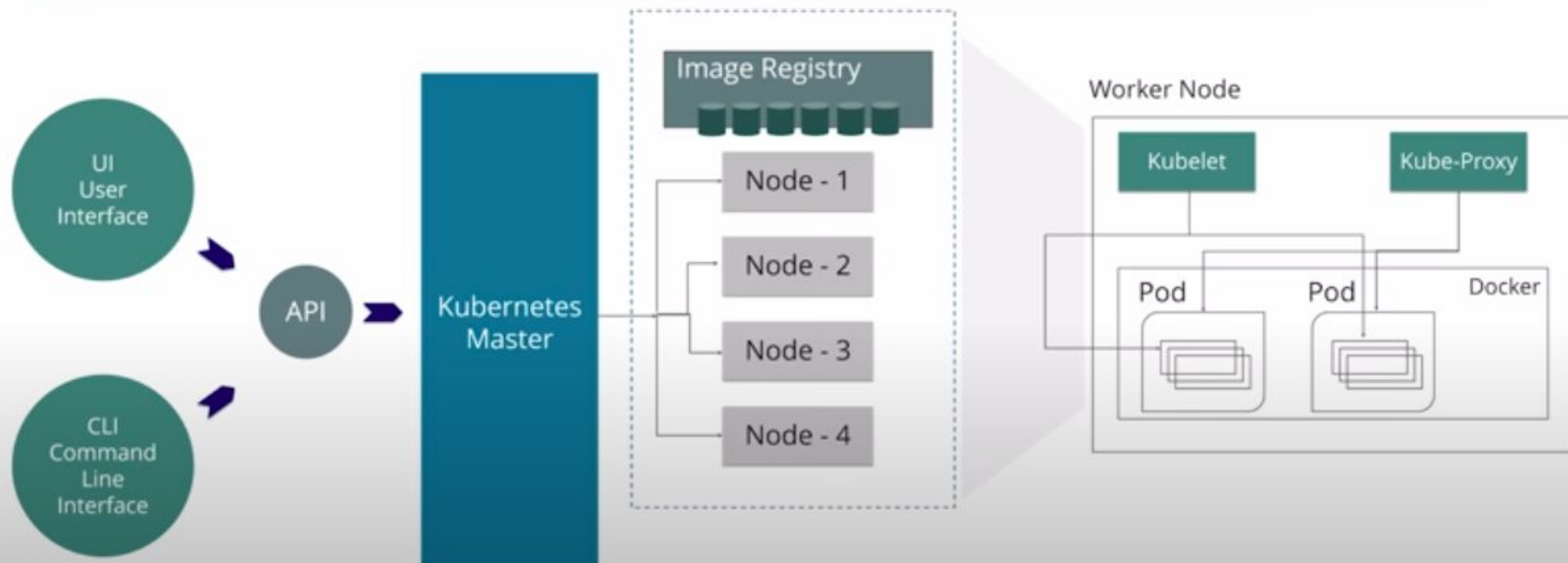




Kubernetes Worker

Kubernetes Worker Node

Worker nodes contains all the necessary services to manage the networking between the containers, communicate with the master node, and assign resources to the containers scheduled



Kubernetes : Kube-Proxy and Kubelet

It acts as a network proxy and a load balancer for a service on a single worker node

It is a command line tool to communicate with the API service and send commands to the master node

