Container Orchestration

==============================

This is the process of handling containers running on multiple linux servers in a distributed environment

Advantages

=================

1 Load Balancing

2 Scaling

3 Rolling update

4 High Availability and Disaster recovery(DR)

Load Balancing

==================

Each container is capable of sustaining a specific user load

We can increase this capacity by running the same application

on multiple containers(replicas)

Scaling

==============

We should be able to increase or decrease the number of containers

on which our applications are running without the end user

experiencing any downtime.

Rolling update

=======================

Application running in a live environment should be upgraded or

downgraded to a different version without the end user having any

downtime

Disaster Recovery

======================

In case of network failures or server crashes still the container

orchestration tools maintain the desired count of containers

and thereby provide the same service to the end user

Popular container orchestration tools

===========================================

1 Docker Swarm

2 Kubernetes

3 OpenShift

4 Mesos

=======================================

**Kubernetes** is a portable, extensible, open source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem.

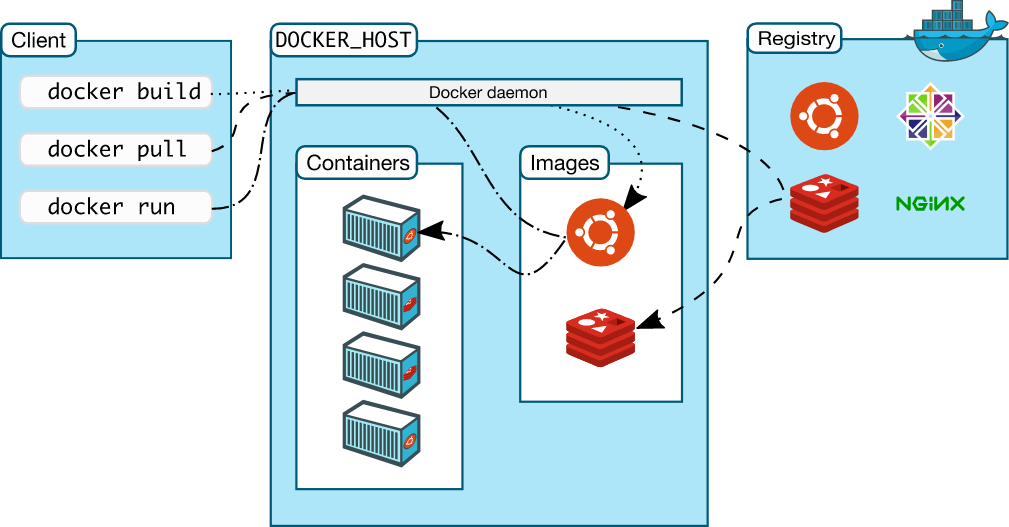
Difference between Docker and Kubernetes?

**Docker is a suite of software development tools for creating, sharing and running individual containers;**

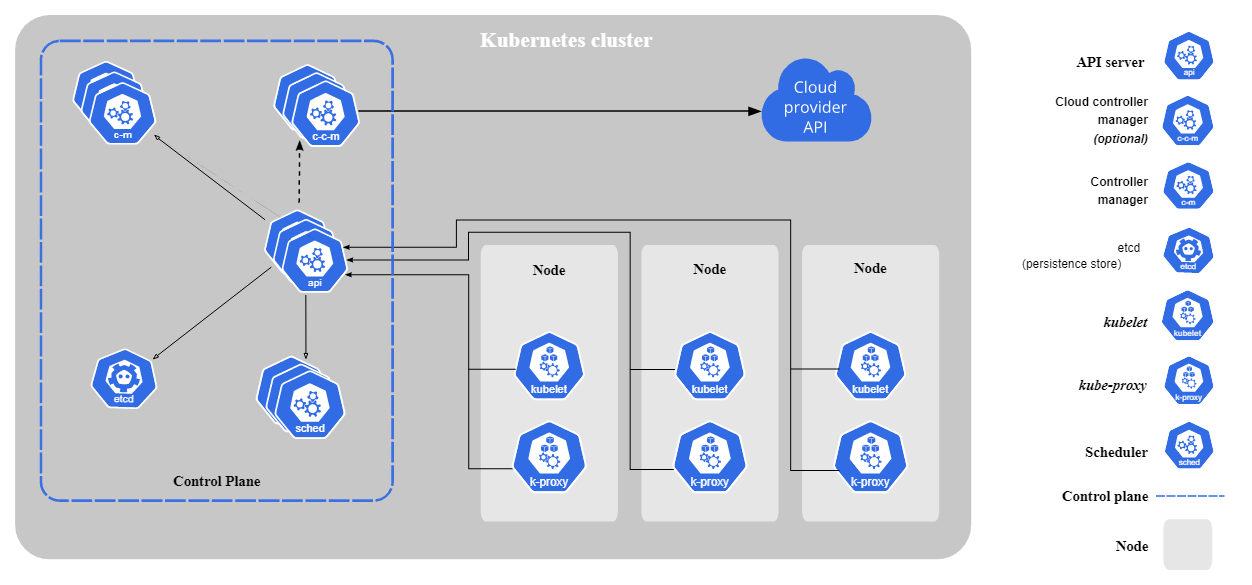
**Kubernetes is a system for operating containerized applications at scale.**

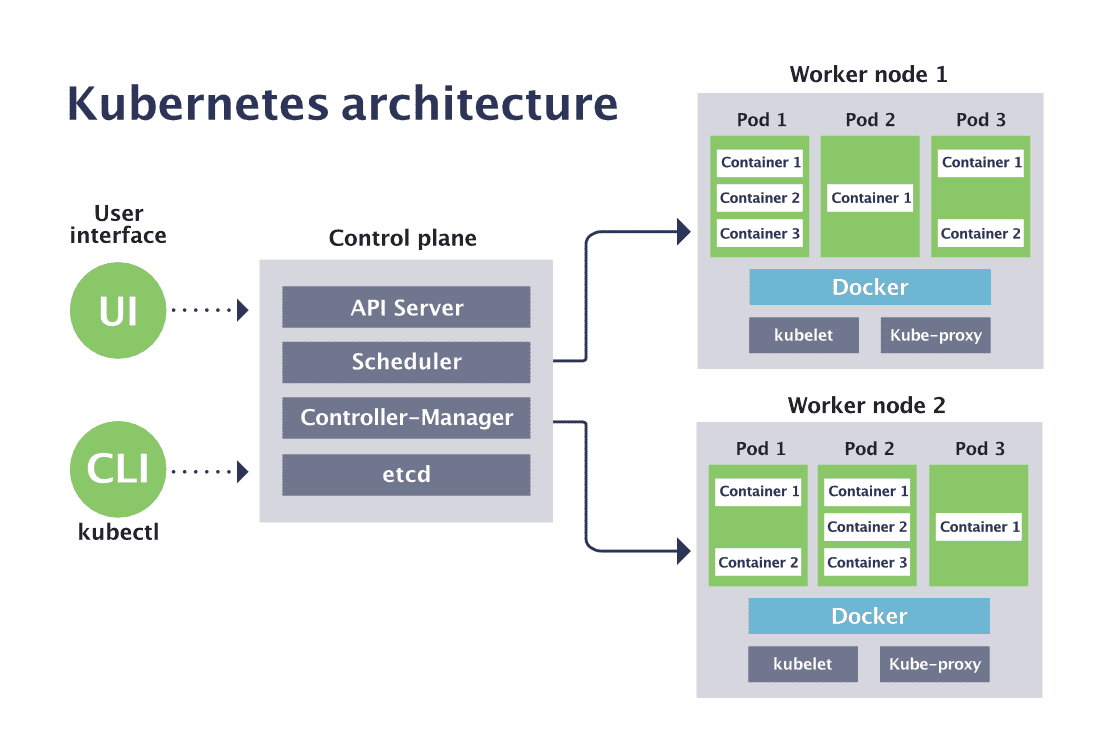
**For Deployment, scaling, Networking, Security the application.**

**Docker Architecture**



**Kubernetes Architecture**





A**pi Server** : API server **validates and configures data for the api objects which include pods, services, replicationcontrollers, and others. It excute the specific command and passes to scheduler .**

**Scheduler :** scheduler is **a control plane process which assigns Pods to Nodes**.

It acts like a Quaram checks the slave have necessary hardware resources and instruct kubelet to deploy application.

**Kubelet**: **Actually process of the Kubelet is to taken orders from scheduler and deploy application on a slave**. watches the API server for pods on that node and makes sure they are running

**Controller-manager** : It is **a daemon that embeds the core control loops shipped with Kubernetes**. it makes the necessary changes attempting to move the current state towards the desired state.

**Etcd** : Kubernetes uses etcd **to store all its data – its configuration data, its state, and its metadata**. Controller and Scheduler read the info from etcd.

**Kubeproxy** : Kubeproxy watches the API server for pods/services changes in order to maintain the network up to date.

**Kubernetes Objects :**

**Pod :** It is alyaer of abstration on top of container it work like a translator .

**Service:**  It is a logical abstraction for a deployed group of pods in a cluster

Since pods are ephemeral, a service enables a group of pods, which provide specific functions (web services, image processing, etc.) to be assigned a name and unique IP address (cluster IP).

**Name space** : It is defined as Creation of partitions in cluster ,it is a way to organize cluster into virtual sub-cluster Any number of namespaces are supported within a cluster, each logically separated from others but with the ability to communicate with each other.

**Secret** : It is an object that contains a small amount of sensitive data such as a password, a token, or a key.

**Replication Controller**: It ensures that a specified number of pod replicas are running at any one time. In other words, a ReplicationController makes sure that a pod or a homogeneous set of pods is always up and available

**Replica set**: It is a Kubernetes object that ensures there is always a stable set of running pods for a specific workload. It is as similar as replication controller but it uses the advanced features .

**Deplyoment**: It is used to create or modify instances of the pods that hold a containerized application. Deployments can help to efficiently scale the number of replica pods, enable the rollout of updated code in a controlled manner, or roll back to an earlier deployment versions.

**Volume** : It is used to preserve the data even when the pods are deleted .

**Statefulset**: It is **the** workload API object used to manage stateful applications. Manages the deployment and scaling of a set of Pods, and provides guarantees about the ordering and uniqueness of these Pods

Kubernetes Definitation files

Objects in kubernetes cluster are deployment are deployed using these files

They are created using yaml file and they generally these are four top level fields are apiversion,kind,metadata,spec.

**Apiversion**: it specifies the code library that has to be imported to create a particular kind of kubernetes object .

**Kind**: It represents the type of kubernetes objects to created while using yaml file like pod,replicaset,deployment,service etc.

**Metadata**: It can be a crucial tool for organizing and understanding the way containers are orchestrated across your many services, machines, availability zones or (in the future) multiple clouds.

**Spec:** It is exact info about object created in specific like container info port mapping/no of replicas etc.