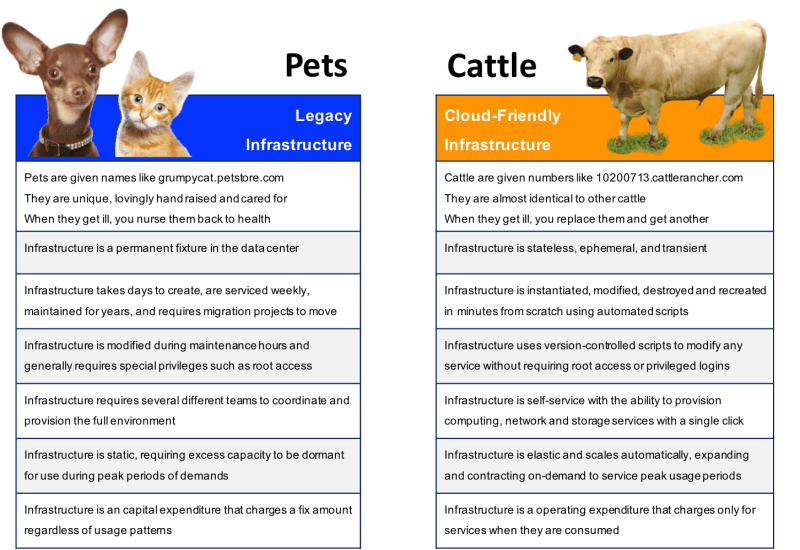
**Kubernetes (k8s)**

* k8s is a portable, extensible, opensource platform for managing containerized workloads and services.
* It facilitates both declarative configuration and automation
* k8s provides
  + Service discovery and load balancing
  + Storage Orchestration
  + Automated rollouts and rollbacks
  + Automatic bin packing
  + Self-healing
  + Secret and configuration management
* Kubernetes is very popular container orchestration platform
* All major cloud providers such as AWS, GCP, Azure have native Kubernetes support
* Cattle vs Pets
  + 

Official documentation on Kubernetes components(<https://kubernetes.io/docs/concepts/overview/components/>)

What Is Kubernetes?

|  |  |  |
| --- | --- | --- |
| Written in | [Go](https://en.wikipedia.org/wiki/Go_(programming_language)) |  |
| [Type](https://en.wikipedia.org/wiki/Software_categories#Categorization_approaches) | [Cluster(master/node) management software](https://en.wikipedia.org/wiki/Computer_cluster#Cluster_management) |  |

Kubernetes (commonly stylized as k8sis an [open-source](https://en.wikipedia.org/wiki/Open-source_software) [container](https://en.wikipedia.org/wiki/Operating-system-level_virtualization)-[orchestration](https://en.wikipedia.org/wiki/Orchestration_(computing)) system for automating [application](https://en.wikipedia.org/wiki/Application_software) deployment, scaling, and management. It was originally designed by Google, and is now maintained by the [Cloud Native Computing Foundation](https://en.wikipedia.org/wiki/Cloud_Native_Computing_Foundation). It aims to provide a "platform for automating deployment, scaling, and operations of application containers across clusters of hosts"

Kubernetes is an open-source container management (orchestration) tool. Its container management responsibilities include container deployment, scaling & descaling of containers & container load balancing.

AKMS in Kubernetes:

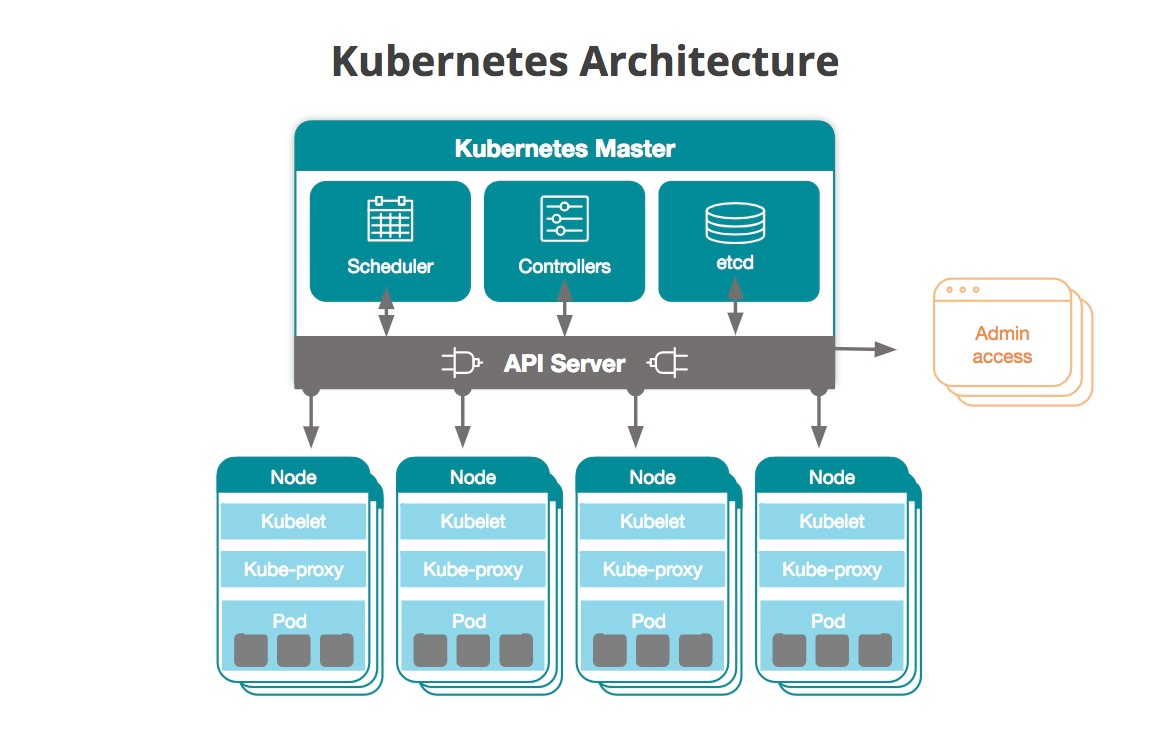


Kubernetes in dictionary: **helmsman = मल्लाह Usage:** एक कुशल'helmsman'अपनी नैया को तूफान से निकाल लाता है.

Why Use Kubernetes?

Companies out there maybe using Docker or maybe simply Linux containers for containerizing their applications. But whatever it is, they use it on a massive scale. They don’t stop at using 1 or 2 containers in Prod. But rather, 10’s or 100’s of containers for load balancing the traffic and ensuring high availability. Keep in mind that, as the traffic increases, they even have to scale up the number of containers to service the ‘n’ no of requests that come in every second. And, they have to also scale down the containers when the demand is less. These kinds of situation can be handled easily by container management (Kubernetes)

Master and Node setup of Kubernetes:



Master: API Manager (communication) schedular(creating stuffs) controller (maintaining state) cluster store(memory)

Nodes kubelet (agent) container engine kube-proxy(networking component)

Manifest:(yml or yaml files) Whatever communication happen between node and master. (it defines desire state)

# Kubernetes Master and Node Components: Kubernetes Master

Note: Kubernetes master runs on Linux nodes

### Master Components

#### kube-apiserver

* This component is Central to Kubernetes. All communications between all components goes through the kube-apiserver
* This component is frontend of the Kubernetes control plane.
* This component exposes a REST API.
* We would interact with this component using **kubectl** by using the YAML files, which are also referred as manifests

#### etcd

* etcd stores the entire configuration and the state of the cluster.
* etcd is consistent and highly available distributed key-value store.

#### kube-scheduler

* It watches for new work tasks and assigns them to healthy nodes in the cluster

#### Controller-Manager

* It is responsible for maintaining desired states mentioned in the manifest.
* It looks like single component, but within it has
  + Node Controller: for noticing & responding when node goes down
  + Replication Controller: for maintaining the correct number of pods for every replication controller object.
  + Endpoints Controller: Populates the Endpoints object

#### Cloud-Controller-manager

* If you run the Kubernetes on a supported cloud platform such as AWS, Azure or Google, your control plane runs the Cloud-Controller-Manager.
* It is responsible for underlying cloud specific controllers.

### Node Components

#### kubelet

* This is an agent which runs on each node in the cluster.
* It watches for the instructions from API Server for new work assignments.
* If it can’t run the task assigned, it reports back to master and lets control plane decide on the actions.
* It is responsible for the node registration process

#### Container Runtime

* This is software which is responsible for running containers.
* Some of them are **Docker, container, cri-o, rktlet**

#### kube-proxy

* Maintains the network rules on nodes
* This is responsible for networking on nodes.
* It’s networking component which provides ip-address to pod.

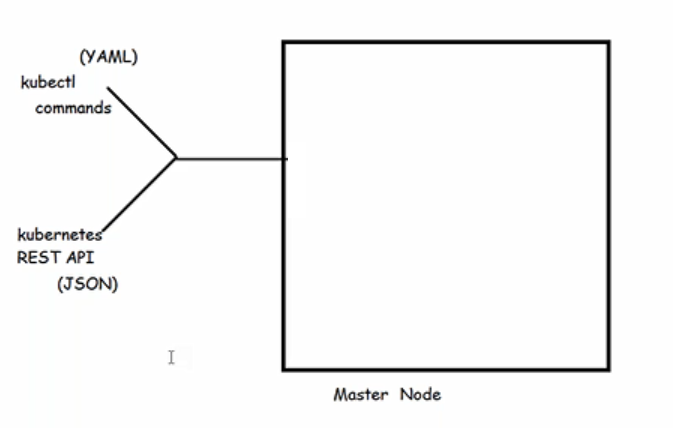
Declarative model and Desire state

Casestudy: Suppose, you hired one software engineer in your team, so now you have 2 options: What to do? -> If he is a fresher. How to do? -> if he is an Experienced.

Creating infrastructure by using yaml file/json file is declarative approach



When to use Json/yaml in Kubernetes?



## 