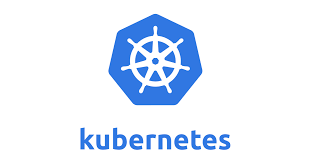
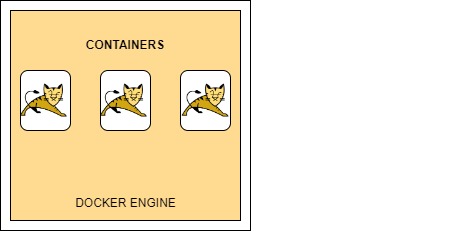
**KUBERNETES**

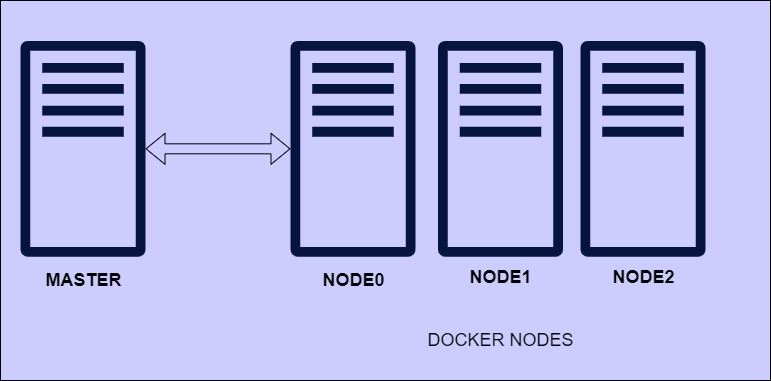
[Kubernetes](https://kubernetes.io/docs/concepts/overview/), also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications.

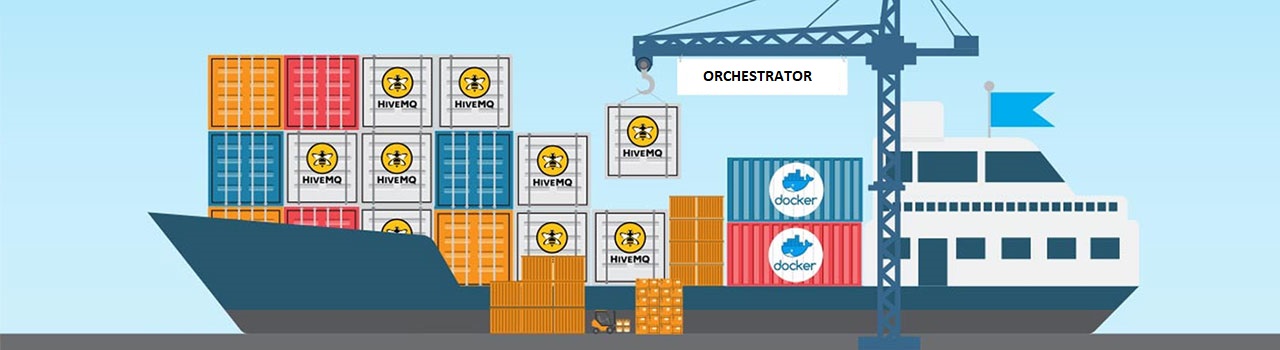
Also known as most famous container orchestration tool.

**Docker Engine**



**CLUSTERING**



**Container Orchestration**

**Orchestration Tools**

* **Docker Swarm**

**Self-Managed / Unmanaged Kubernetes**

* **Kubernetes**
* **OpenShift**
* **Rancher**
* **Mesos**

**Managed Kubernetes**

* **AWS EKS & ECS**
* **Azure AKS**
* **Google Container Engine (GKE)**

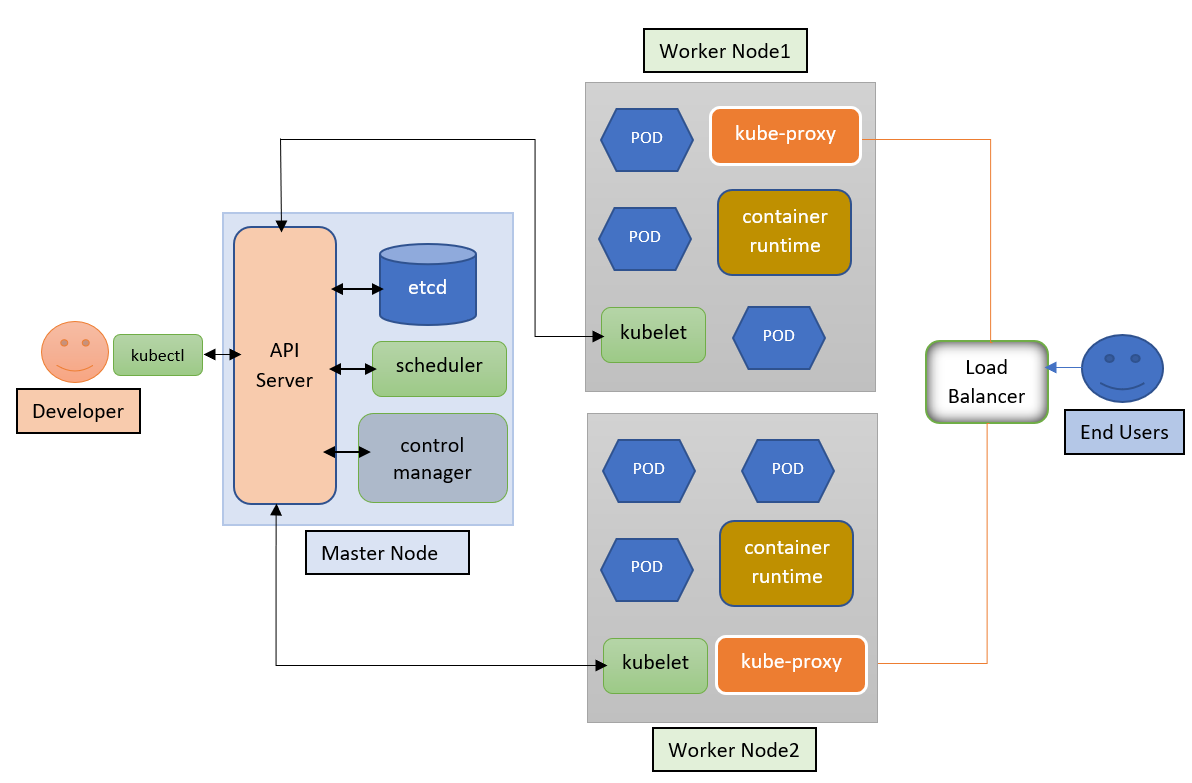
**Kubernetes History**

* Created by **Google** to manage their containers AKA Borg
* **Mid-2014:** Google introduced Kubernetes as an open-source version of Borg
* **July 21-2015:** **Kubernetes v1.0** gets released. Along with the release, google partnered with the Linux Foundation to form the **Cloud Native Computing Foundation (CNCF)**.
* **2016:** Kubernetes Goes Mainstream!
  + Kops, Minikube, Kubeadm etc
  + September 29: **Pokeman GO! Kubernetes Case Study Released**
* **2017:** Enterprise Adoption
  + Google and IBM announce Istio
  + GitHub runs on Kubernetes
  + **Oracle joined** the Cloud Native Computing Foundation

**Kubernetes Provides**

* Service discovery and load balancing
* Storage orchestration
* Automated rollouts and rollbacks
* Automatic bin packing
* Self-healing
* Secret and configuration management

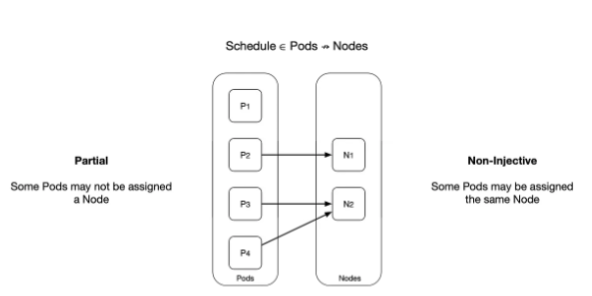
**Kubernetes Architecture**

****

**Master Node / Control Plane Components**

* **Kube API Server:** Main Hero! Handles all the requests and enables communication across stack services. It is the front end of Kubernetes control plane.

Admins connect to it using **Kubectl CLI**

* **Kube Scheduler:** It watches newly created pods that have no node assigned, selects a node from them to run on
  + Factors taken into account for scheduling decisions include
    - Individual and collective resource requirements
    - Hardware/software/policy constraints
    - Affinity and anti-affinity specifications
* **ETCD Server:** It stores all the information. Consistent and highly-available key value store used as Kubernetes backing store for all the cluster data.

Kube API stores retrieves information from it.

Should be backed up regularly

* **Controller Manager:** It manages various **controllers** in Kubernetes.

1. **Node Controller**

It is responsible for onboarding new nodes to the cluster handling situations where nodes become unavailable or get destroyed to keep our application running.

If it stops receiving signals from a node, the node is marked unreachable but it waits for 40 seconds before marking it unreachable. After the node is marked unreachable it waits for 5 minutes to come back up if it doesn’t, it removes the [POD’s](https://kubernetes.io/docs/concepts/workloads/pods/) assigned to that node and provisions them on the healthy nodes if [POD](https://kubernetes.io/docs/concepts/workloads/pods/)s are part of the replica set.

1. **Replica Controller**

It is responsible for monitoring the status of replica sets and ensuring that the desired number of [PODs](https://kubernetes.io/docs/concepts/workloads/pods/) are available at all times within the set. If a POD dies it will create another POD. It makes sure that desired number of PODs or at least one POD is in running state. It has the capability to bring up or down the specified no of PODs.

1. **Endpoints Controller**

Populates the Endpoints object (that is, joins services & pods)

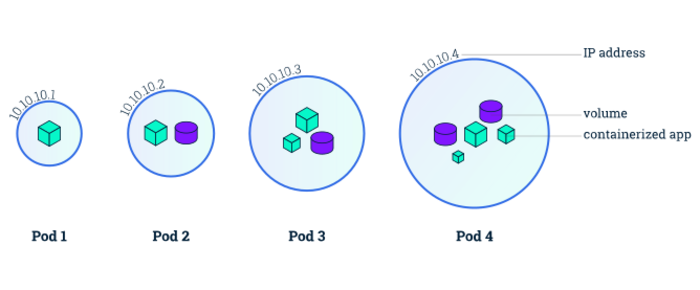
1. **Service Account & Token Controllers:**

Create default accounts and API access tokens for new namespace

**Worker Node Components**

* **Kubelet:** An agent that runs on each node in the cluster. It makes sure that containers are running in a pod.
* **Kubeproxy:** Network proxy that runs on each node in your cluster.
  + Network rule
    - Rules allow network communication to your pods inside or outside of your cluster.
* **Container Runtime:** Kubernetes supports several container runtime
  + Docker
  + Containerd
  + Cri-o, rktlet
  + Kubernetes CRI (Container Runtime Interface)

**What is POD in Kubernetes**

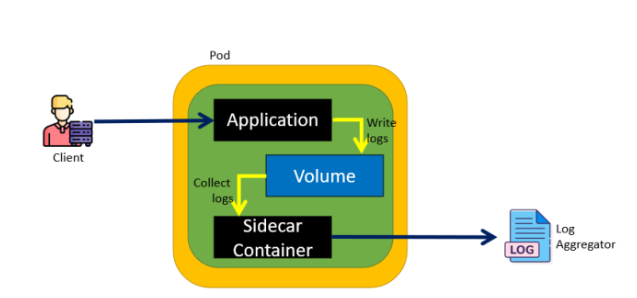


**Use of Multi Container POD:**

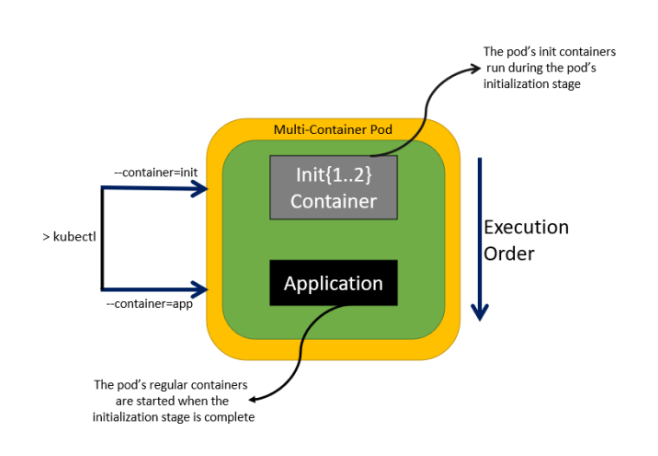
It depends on requirement.

Example 1: Side Car Container along with application container

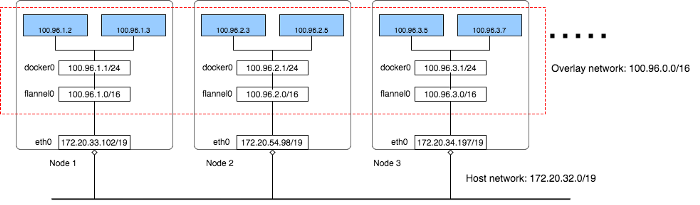
A sidecar is just a container that runs on the same pod as the application container. It shares the same volume and network as the application container, it can “help” or enhance how the application operates.



Example 2: Init Container is short lived container; it will start does some command execution and then it will be dead. Once it dead the main/application container will start.



**POD Networking**

****

**Kubernetes Setup**

* **Hard way:** Manual setup
* **Minikube:**
  + One node Kubernetes Cluster on your laptop
* **Kubeadm:**
  + Multi node Kubernetes Cluster
  + Can be treated on any Platforms VM’s, EC2, Physical machines etc.
* **Kops:**
  + Multi node Kubernetes Cluster on AWS
* **Kind:**
  + Supports multi–Node Cluster on your laptop using docker containers
* GKE:
  + Supports multi-node cluster on Google cloud

**Setup with Minikube**

**Pre-requisites:**

1. Chocolatey

<https://chocolatey.org/install>

1. Oracle virtual box

<https://www.virtualbox.org/wiki/Downloads>

1. Install Docker Desktop & open it from start menu

<https://docs.docker.com/desktop/install/windows-install/>

**Steps**

* Open Powershell as Administrator
* Install with Chocolaty

**choco install minikube Kubernetes-cli -y**

* Open Powershell again

**minikube start**

**Verify it by running below command**

$ kubectl get nodes

$ cat .kube/config

<https://minikube.sigs.k8s.io/docs/start/>

<https://minikube.sigs.k8s.io/docs/handbook/deploying/>

Example Deployment:

$ kubectl create deployment hello-minikube1 --image=kicbase/echo-server:1.0

$ kubectl expose deployment hello-minikube1 --type=NodePort --port=8080

$ minikube service hello-minikube1 –url

Cleanup the Minikube setup

$ minikube stop

$ minikube delete

**Setup with Kops**

**Pre-requisites**

1. **Domain for Kubernetes DNS records**
   1. **E.g., sscademy-k8s.com**
2. **Create a Linux VM and setup**
   1. **Kops, kubectl, ssh keys, awscli**
3. **Login to AWS account and setup**
   1. **S3 bucket, IAM User for AWSCli, Route53 Hosted Zones.**