**Dis-Advantages of Docker:**

* We have a docker host where all containers are running

what if docker host crash? we lose all containers

even we use docker volumes, data is still in the host, so we lost data as well

* What if traffic increases/decreases? are our containers scalable
* Even we have multiple containers to balance the load? who is the LB here?
* What if some container crashes? can docker make it available again?
* What about configurations and secrets? where to store them?
* What if we have multiple hosts running with containers

We need an orchestrator to resolve all the above drawbacks.

Orchestrator is like a manager in a team, how a manager divides the workload, assign and manage a team to get the maximum output/efficiency.

In the same way Kubernetes is a container orchestration platform that manages the containers though pods.

**Kubernetes**

* Kubernetes, K8s is an open source container orchestration platform developed by Google.
* It is designed to automate deployment, scaling and management of containerized applications across a cluster of nodes.
* A Node refers to a single virtual machine/environment; a cluster refers to a collection of nodes.
* Kubernetes provides a consistent and reliable way of managing applications regardless of whether they are running on cloud, on-premises and hybrid.
* Micro services are a software architecture patterns that divides a complex application into small, independent and executable services which are deployed as containers.
* Each micro service is designed to perform a specific function or business capability, and communicates with each other over APIs.

**FAQs:**

1. **How kubernetes is different from docker?**

Docker is a containerization tool that helps us to build containers in a single Host Machine or in a single node.

Whereas kunernetes is a container orchestration management tool that help use to deploy the above docker created containers across a cluster of nodes (collection of docker machines).

***Kubernetes can not create containers as docker, it can do only orchestration on the container.***

1. **Is docker mandatory for kubernetes?**

kubernetes can not directly interact with containers as docker can does with docker commands. But kubernetes creates a layer on top of container to maintain abstraction called POD. Kubernets indirectly interact with container through kubernets commands, POD understands these commands communicates with container indirectly.

***YES. Any one of the containerization tools is needed for Kubernetes.***

1. **What are kubernetes alternatives?**

***Docker Swarm, Open Shift by RedHat***

1. **What are AKS, EKS and GKE?**

We knew that kubernetes is open source and free of cost. But there are few Cloud based Kubernetes which includes the customization on original with few additional features and support from the different cloud providers and they costs for the service provides to you.

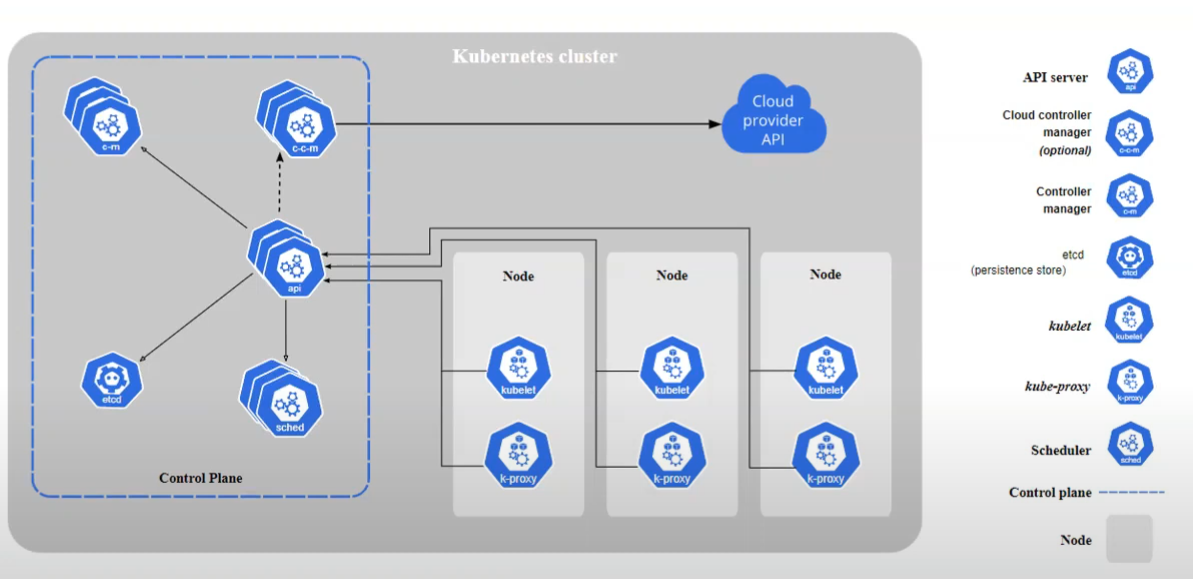
All cloud kubernetes are commercial.

***AKS – Azure Kubernetes service***

***EKS – Amazon Kubernetes service***

***GKE – Google Kubernetes Engine***

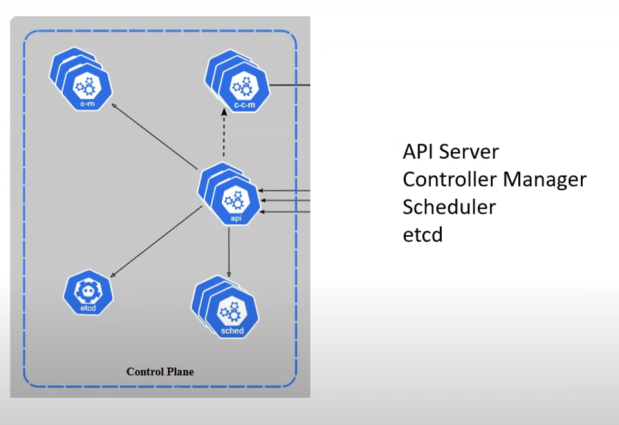
**Kubernetes Architecture:**



* Kubernetes follows master-slave architecture where master is the brain of kubernetes and slaves are like worker nodes which performs the actions instructed by master node.

**Master Node Components:**

* Master node is a VM/physical machine where all the components of Master node are installed.



* A master node has 4 components and each of the components has its own functionality.
* Master node is also known as Control plane as it controls the deployment, scalability, workload across the worker nodes.
* Like worker nodes, we can maintain multiple Master nodes to bring the high availability of master nodes for better safety.
* If the cluster is maintained under cloud, then the responsibility of master nodes cluster will be taken care by the provider, where as if you are doing it on bare minimum machine then its your responsibility to take care of master node cluster.
* There are 2 ways to have multiple Master nodes

1. With stacked control plane nodes: where etcd database nodes are colocated with control plane nodes to form a group.

2. With external etcd nodes: where etcd database runs on separate nodes to the control plane nodes.

1. **API Server:**

It is the main entry point to the kubernetes master node cluster. It communicates between the worker and master node. There is an angent in each of the worker node called “Kubelet”, it always updates the information of woker node up to date to the master through this API Server. With help of the kubelet information via API server master node get to know the availability, space and other parameters about the worker node.

It is the front end and entry point of the control plane.

It also handles authentication and Authorization.

1. **Etcd Database:**

The information carried by the APIs will be stored in etcd Database. It contains the entire cluster data in key value pairs consistently and acts as backup storage**.**

1. **Scheduler:**

With help of etcd data, scheduler checks the capacity, availability, space and identifies free, newly created resoures/nodes and schedules the work/actions/tasks across the worker nodes.

1. **Controller Manager:**

Control manager controls the each & every processes that are running at master.

**Worker Node Components:**

1. **Kube-let:**

It is a agent set by kubernetes at worker node to send the updates and stats of worker nodes.

It make sure that containers are running in a POD and are healthy and updates the PODs health.

1. **Kube-proxy:**

It is a Network proxy service which establishes the fastest and shortest path connections with in a worker node or in a cluster of nodes. It is responsible for the communication among the containers in a quick way in order to reduce the latency of request.

1. **Container Platform:**

At the end, each worker node holds a micro service as containers and to run those container a container engine/platform is needed.

**WorkLoad Components:**

1. **POD:**

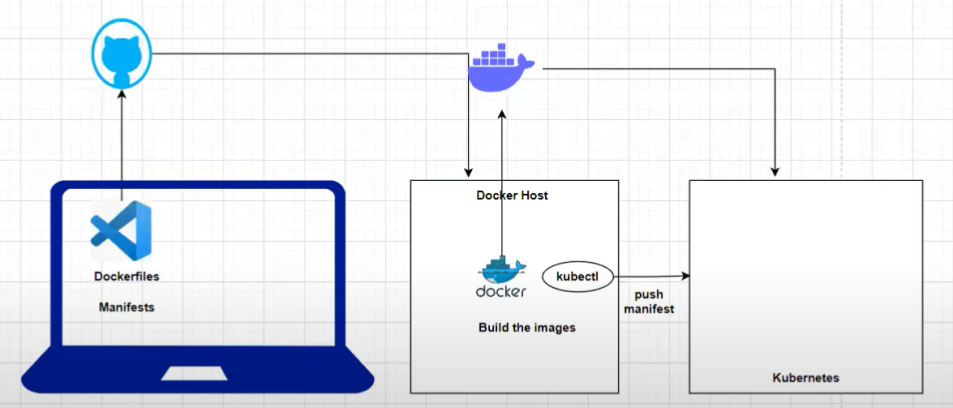
POD is nothing but a container and collection of containers, as we cannot interact with docker containers directly with docker commands in kubernetes. To resolve this kubernetes maintain a layer on top of docker container which gives the CLI for kubernetes to interact.

This layer is known as POD.

So kubernetes interacts with PODs which indirectly interacts with docker containers.

POD is the lowest level of workload component.

1. **Deployments:** collection of PODs can be treated as deployment.
2. **Services:** establish the service connections between the PODS.
3. **Ingress:** establish the network connections between the PODS.
4. **ConfigMaps:** used to store the PODs/services configuration
5. **Secrets :** used to maintain the credentials
6. **Namespaces:** Logical separation of PODS(different environments like develop,test)
7. **Persistent volumes & claims:** Storage purposes



* We push the dockerfiles,manifest to GitHub from there a server(docker host) pulls the files and builds the images. These images push to docker artifactory.
* An agent “kubectl” present in docker host to connect with kubernetes server and executes the kubernetes commands from the manifest into kubernetes server.

**Mini kube:**

Mini kube is a single node(manager& worker node are same) cluster service mainly used for practice.