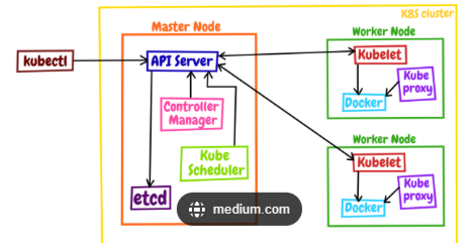
**Basics**: Kubernetes is open source container management tool which automate container deployment, container scaling and load balancing.

* monolithic application: single stone application, every
* Microservice: each task is deploy in diff-2 services, connect with each other via API
* Orchestration tool = container management tool

**Features**:

1. Orchestration
2. Autoscaling
3. Autohealing
4. Load balancing
5. Platform independent
6. Fault tolerance
7. Roll back
8. Health monitoring of containers
9. Batch execution

**Architecture:**



**Components Of master plane:**

1. API Server

2. Kube Scheduler

3. Controller Manager

4. ETCD (not part of k8s but without this k8s won't work so consider this also a part of k8s)

**API Server**:

a. It interact directly with users

b. It meant to scale automatically according to losad or request load

c. Front end of control plane

**ETCD:**

a. Store metadata or status of cluster

b. Consistent and high availability

c. Store data in key value form

Features:

a. Fully replicated: entire state is available on every node of cluster

b. Secure: implements TLS with optional client-certificate authentication

c. Fast: benchmark at 10,000 writes per second

**Kube Scheduler**:

* When user make a request for creation or management of pods , it is going to take the action on his request
* If any mismatch occurs in number of pods runs then it will make them as desired no pods When we are not explicitly assign the node for pod creation then it will automatically decide the best node and create pod there. But if you want to create pod on specific node then assign node in manifest file.
* It's take the hardware configuration information from etcd and that help to decide the best node for pod creation.

**Controller Manager:**

* Make sure actual state equals to desired state
* If k8s on cloud then "cloud controller manager"
* If k8s on non-cloud "kube-controller manager"

Controller Components**:**

a. Node Controller: for checking of nodes that has detect in cloud after it's stop responding.

b. Route Controller: Responsible to setting up n/w, route

c. Service Controller: Responsible for load balancing d. Volume Controller: Managing Volumes

**Components Of Worker Plane:**

1. Kube Proxy

2. Kubelet

3. Pods

4. Container Engine

**1. Kube Proxy:**

* It is responsible for networking and responsible to allocate the IP for pods
* It's runs on each node
* It's communicate to master via the API Server

**2. Kubelet:**

* Agent running on node
* Listen the k8s master (pod creation request)
* Provide pod information to etcd via API Server
* use port 10255 e. Send success / failure status to control plane

**3. Pods:**

* smallest unit of k8s
* One pod can contains multiple container but recommend only one container in one pod
* Pod having it’s IP address but container don't have
* Cluster has at least one master node and one worker node
* K8s cannot start container without pods
* Auto scaling and auto healing by default not provided by pod for this high level k8s object required g. Pod crashed is also one more limitation but fix this by high level object

**4. Container Engine:**

* Work with kubelet
* Pulling image
* Start/stop container
* Expose port which is specified in manifest

Cluster --> node --> pod -->container --> application