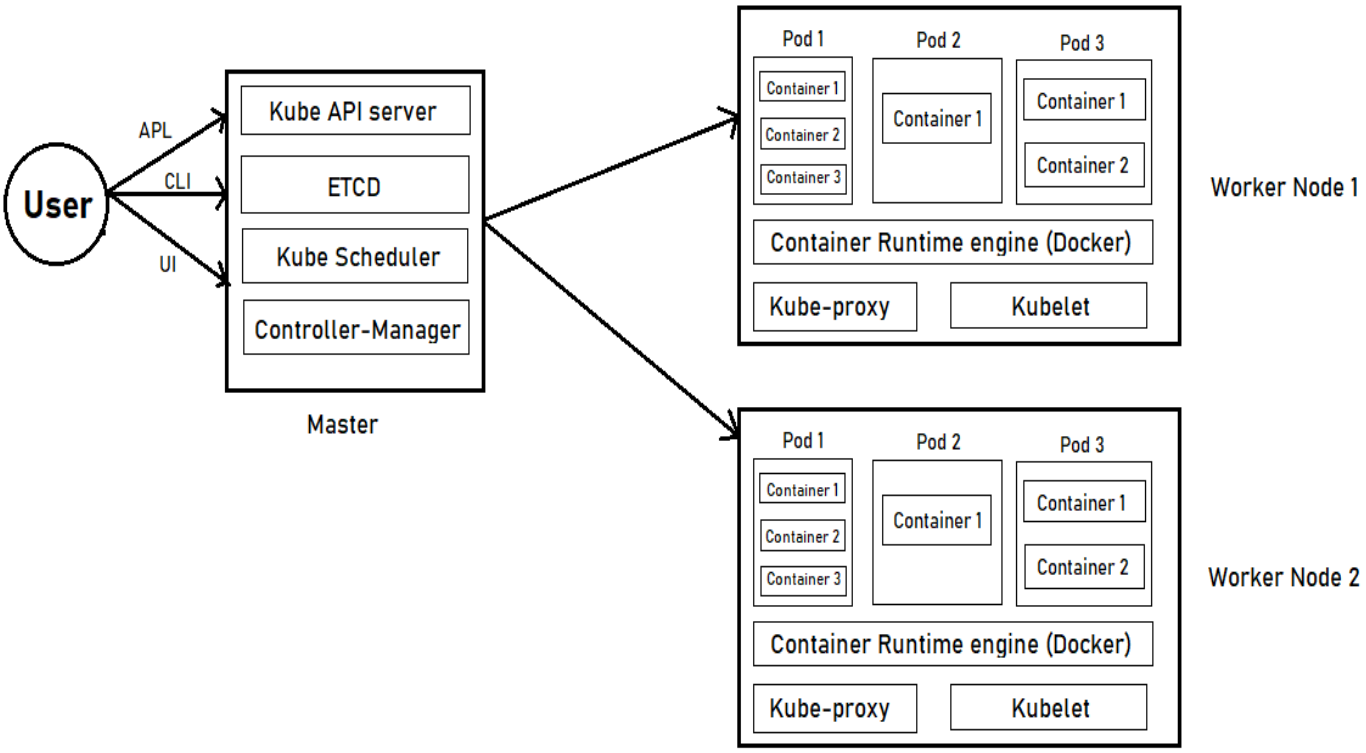
Kubernetes Architecture



Master

A) ETCD: -

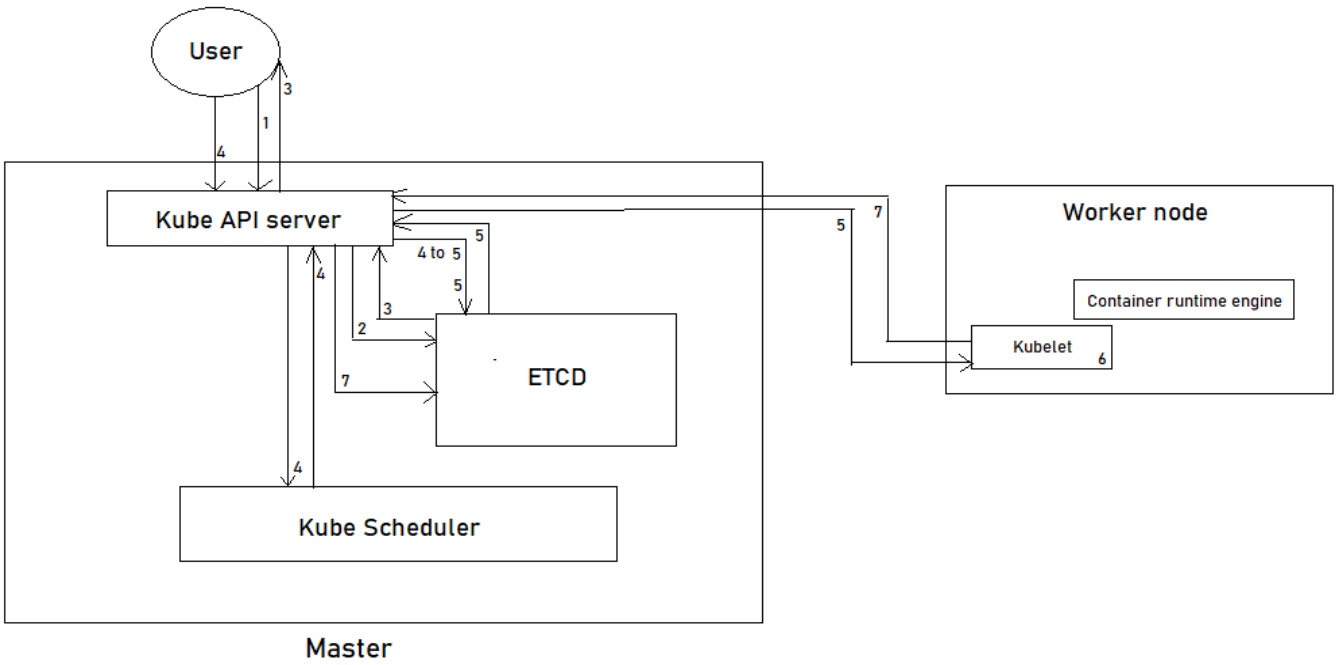
- It is accessible only by kubernetes api-server for security reasons.
- ETCD datastore store the information regarding the cluster such as: -
    1. Nodes
    2. PODs
    3. Configs
    4. Secrets
    5. Accounts
    6. Roles
    7. Bindings
    8. Others

B) Kube API Server: -

- kube API server is the primary management components in Kubernetes. When you run a kubectl command the kubectl utility is in fact reaching to the kube API server.
- Kube API server is user to deal with CLI, GUI, API communications.
- The Kube API server is at the centre of all the different tasks that need to be performed to make a change in cluster.

Example:

Creating a pod



1. The kube API server authenticated the request and validated it.
2. The API server creates a POD object without assigning it to a node and updates the information in the ETCD.
3. Updates the user that the POD has been created. (Retrieves the data from the ETCD and responds back with the requested information)
4. The scheduler continuously monitors the API server and realizes that there is a new pod with no node assigned. The scheduler identifies the right node to place the new POD on and communicates that back to kube API server.
5. The API server then updates the information in the ETCD. The API server then passes that information to the kubelet in appropriate worker node.
6. The Kubelet then creates the POD on the node and instructs the container runtime engine to deploy the application image.
7. Once Pod is created then the kubelet updates the status back to the API server and the API server then updates the data back in the ETCD.

C) Controller Manager: -

A Kube Controller is a process that continuously monitors the state of various components with in the system and works towards bringing the whole system to the desired functioning state.

Examples:

a. Node-controller: -
   The node controller is responsible for monitoring the status of the nodes and taking necessary actions to keep the action running. It does that through to kube API server. The node-controller is checking the status if the nodes.
b. Replication-controller: -
   It is responsible for monitoring the status of replica sets and ensuring that the desired number of PODs are available at all time with in the set. If a pod dies it create another one

Kube-Controller-manager server options

"kubectl get pods -n kube-system"

- Kubeadm: -
  You can see options with in the pod definition file located at
  cat /etc/Kubernetes/manifests/kube-controller-manager.yml
- Non-kubeadm: -
  cat /etc/systemd/system/kube-controller-manager.service

D) Kube scheduler: -

It is responsible for distributing or scheduling the workload on the various worker nodes. Scheduler is only responsible for deciding which pod goes on which node. It doesn't actually place the pod on the nodes. The scheduler only decides which pods goes where.

Worker Node

Kubelet: -

The kubelet is the Kubernetes worker node. When kubelet receives instructions to load a container or a pod on the node it requests the container run time engine (docker etc.) to pull the required image and run an instance.

The kubelet then continues to monitor the state of the pod and the container in it and reports to the kube API server on a timely basis.

Kube-proxy: -

Is a process that runs on each node in the Kubernetes cluster. Its job is to look for new services and Every time a new service is created it creates the appropriate rule on each node to forward traffic to those services to the backend pods. It uses IPTABLES rules