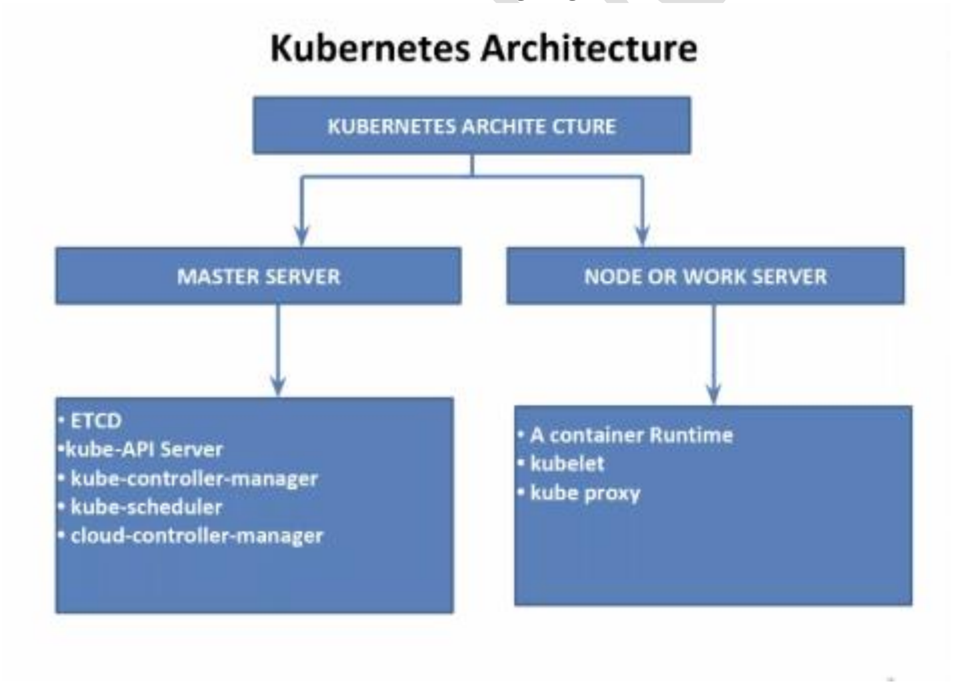
Kubernetes Cluster

Just have a look on below links.
https://www.youtube.com/watch?v=rmf04ylI2K0&authuser=0
https://www.youtube.com/watch?v=R-3dfURb2hA&authuser=0

Kubernetes in an open source container management tool hosted by Cloud Native Computing Foundation (CNCF). This is also known as the enhanced version of Borg which was developed at Google to manage both long running processes and batch jobs, which was earlier handled by separate systems. Kubernetes comes with a capability of automating deployment, scaling of application, and operations of application containers across clusters. It is having capable of creating container centric infrastructure.

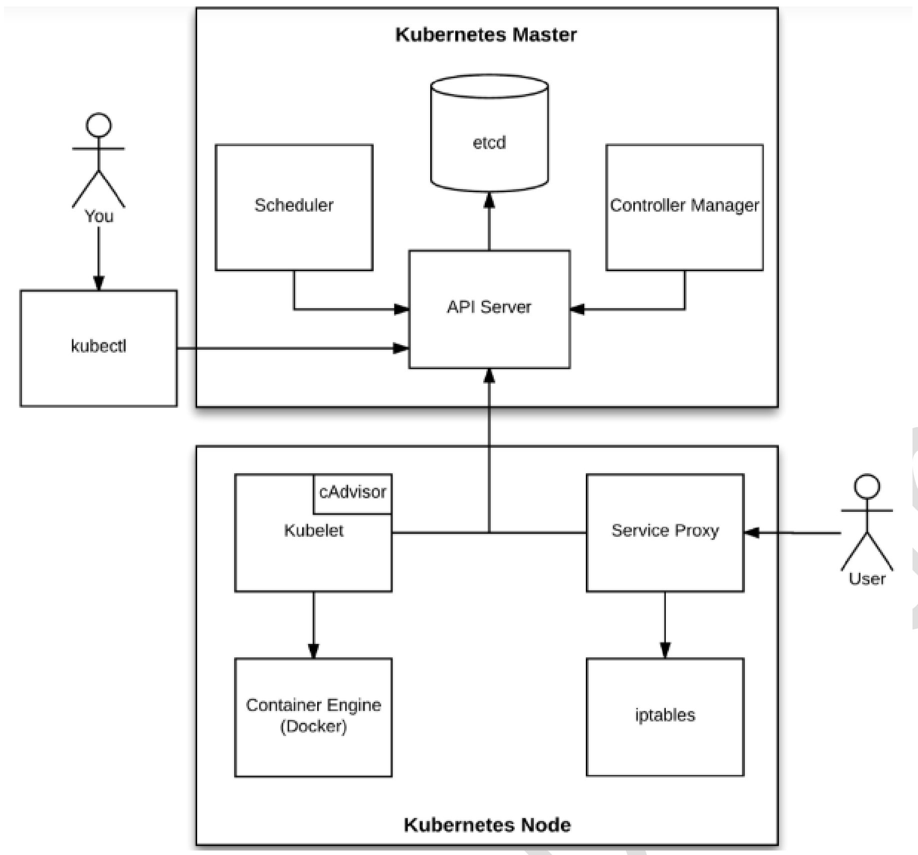
Features of Kubernetes

- Following are some of the important features of Kubernetes.
- Continues development, integration and deployment
- Containerized infrastructure
- Application-centric management
- Auto-scalable infrastructure
- Environment consistency across development testing and production
- Loosely coupled infrastructure, where each component can act as a separate unit
- Higher density of resource utilization
- Predictable infrastructure which is going to be created



Master node:- Entire working principles and logic are present in the master node. minimum 2 gb of RAM and 1 core required for Kubernetes master node.

Worker node:- Creating workloads, like containers are running in worker node.



Kubernetes master node components:- (ETCD, kube-apiserver, kube-SCHEDULER, kube-controller manager)

API-SERVER:- From client tool all information will be landed first on API-SERVER only and from there it will take a request of all other components. it has certificate-based authentication & authorization and all communications are happening via API-SERVER only, it will be act as frontend.

ETCD:- It is data store and call it as single source of truth (SSOT), means in cluster entire running state or info will be stored in ETCD only, so suggestion to have an incremental backup in production. in kubernetes cluster, which container is running on which node, what are configuration and which image is being used by container, so all info will be stored in ETCD only.

SCHEDULER:- it will monitor the backend of worker node, when the client tool kubectl put a request for new container, then that will pass the info via apr-server to SCHEDULER, so SCHEDULER will verify the availability of resource on worker node and then it will choose any of worker node to allow the container run. client instruction, SCHEDULER will decide on which node container can be run.

CONTROLLER MANAGER:- Under this around 8 controller process are runnig, it has majority share for maintaining the cluster state. for example, in replicaset if we choose minimum 4 containers, incase one is missing means then controller manager will bring the new container, basically it will watch inside the cluster, whether if any changes made on object level and watch resource utilization.

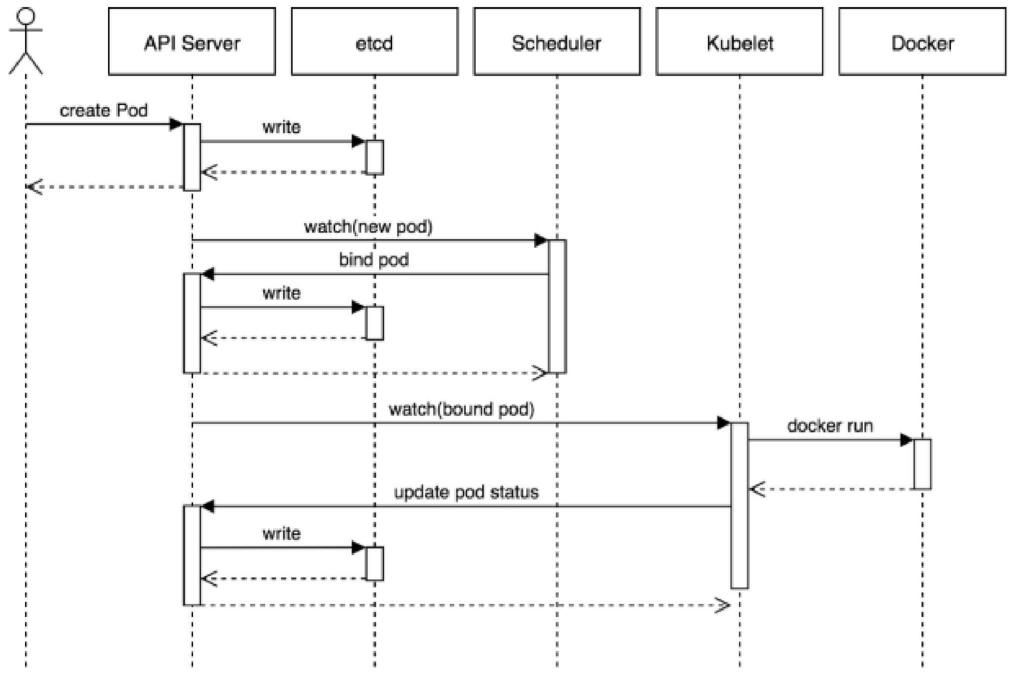
Kubernetes worker node components:- (docker engine, KUBELET, kube-proxy)

DOCKER ENGINE:- compulsory container should run on node, so it required container engine to run on worker node, now default container Kubernetes cluster is docker.

KUBELET:- will establish the communication between master and worker node, if any work on the worker node will be carried out by the master through KUBELET, cadviser is the container monitoring tool.

KUBE-PROXY OR SERVICE PROXY:- on each node, it will set the network components of all containers to expose outside. kube-proxy maintains network rules on nodes. These network rules allow network communication to your Pods from network sessions inside or outside of your cluster. It uses operating system packet filtering layer if there is one and it’s available. Otherwise, kube-proxy forwards the traffic itself.

BELOW FLOW CHART WILL EXPLAIN THE POD CREATION:-



Create Pod Flow. Source: heptio.com

:- There two ways to give instruction on cluster to create pod. (Iterative and Declarative)

1, Iterative method (running command on fly)

2, Declarative method (writing yaml mainfest or jason format to configure).

Kubernetes Cluster Functioning Flowchart

(A) on client side run the kubectl command to create a pod, can do either by command line or manifest yml

(B) then information will go to API-SERVER and API-SERVER will authenticate, then verify the information whether it is valid or not

(C) then information will store in ETCD, then ETCD will update the info and backend it will return the info to api server

(D) then ETCD will pass the information to SCHEDULER and will ask for the feasibility, where the pod can be placed

(E) then SCHEDULER will elect one node to create pod and then confirm the same to API-SERVER

(F) then API-SERVER will take info to ETCD to update then same

(G) ETCD will update the data info and will return info to API-SERVER

(H) on worker node, apr-server will inform to KUBELET with specification info to create pod

(I) then KUBELET will speak with docker engine to pull the specific image to run the container

(J) then KUBELET will inform the up status to API-SERVER and API-SERVER will inform the same to ETCD for information update.