Docker is written in Go Language

**Docker Module Name**

Config

Container

Image

Network

Node

Plugin

Secret

Service

Stack

Swarm

System

Volume

**Container:**

Create container

List container

Remove container

Create container in background

Stop container

Start container

Detach container

Inspect container

**Kubernetes**

**Installation in Window:**

Download minikube

Download kubectl

Create folder in any directory, paste both files

Rename minikube file to minikube

Set path of current folder in environment variable

Open command prompt and run command as “minikube”, will find many command

Run “minikube start” command, it will download files and setup “Kubernetes” cluster.

**kubectl**

It is a client for Kubernetes client. When we run any command using kubectl, it will create a json and hit to API SERVER to execute your command.

**Kubectl get pods**

Get all pods on cluster

**Minikube ip**

Find ip of minikube VM

Default username of minikube VM is “docker” and password is “tcuser”.

**Minikube dashboard**

Access dashboard

Make sure that **SWAP** in all the nodes in kubernetes cluster are off.

To check **SWAP** status, run command as below:

**Free –h**

To check whether SWAP memory is enable or not.

**Swapoff –a**

Disable all the SWAP memory in kubernetes cluster. Once system will restart, SWAP memory will become enable. To disable is permanently, comment all the lines which starts with SWAP in **/etc/fstab** file. These action should be taken on all the nodes.

Kubernetes is an orchestrator platform.

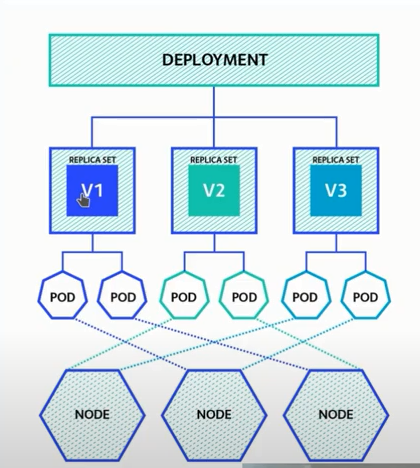
Kubernetes run docker in backend. Kubernetes supports docker runtime as well as containerd, rocket and many more.

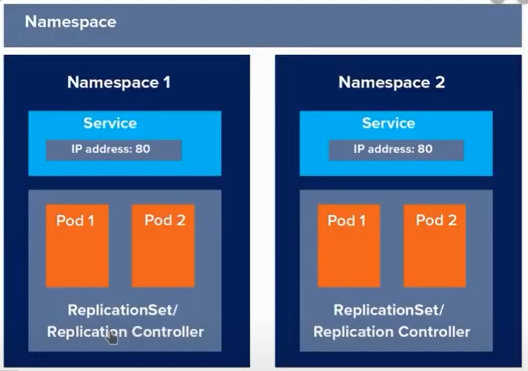
It may happen that Kubernetes will not support latest version of Docker, before you install Docker in Kubernetes, always refer Kubernetes docs.

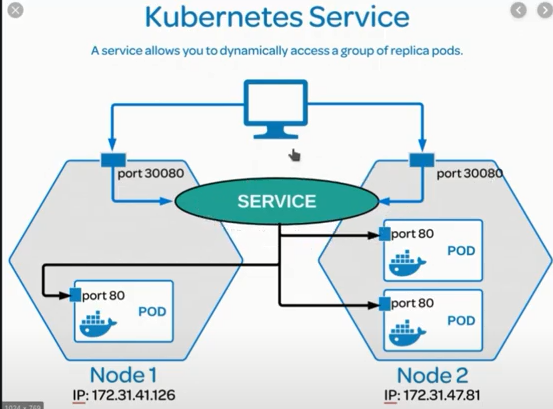
Smallest resource in Kubernetes is **POD**.

POD can contains Containers, Volumes.

Network always assign to POD, not container.







**Kluster Setup:**

**Kubeadm init –pod-network-cidr=10.244.0.0/16**

Run this command on Master to init the kubeadm with pod id range.

Once you run this command it will init the kubeadm and in the end will show 3 commands that we have to run on master.

**Kubectl get nodes**

show all the nodes in cluster.

**Kubectl get pods**

Show all pods from default namespace.

**Kubectl get pods –A**

Show all nodes from all over the namespace.

**Kubectl get pods –o wide**

Show more details about pods that on which node pod is running.

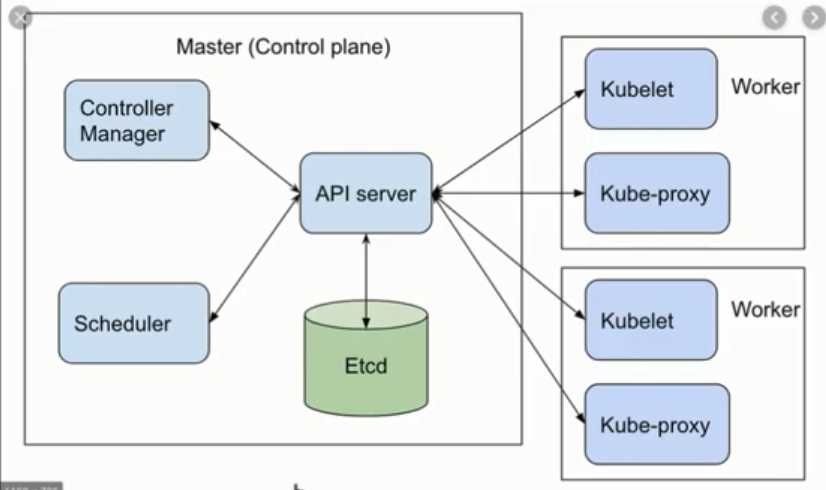
**Kubectl explain pods**

Will explain about Pods. Here we can use specify anything(nodes, replicacontroller, replicaset, namespace) instead of Pods.

**Kubectl describe pod <pod name>**

Will describe pod. It will give in depth details about that pod.

Blue green deployment is called when you want 0 down time from your server.



**API Server**

API server expose API to end user. If you are creating any pod and attaching any label or environment then all these values are stored in **Etcd**. Etcd is a database, which store information in the form of key-value pair. Etcd can store data of node, pod, config, secret, accounts, rolebinding, replica controller, replica set.

Whenever user execute any command, that is converted in JSON format and hit to the API server. End user can contact to only API server, he/she can’t talk to Manager, Scheduler directly. And Manager or Scheduler can’t talk to commands(Kubelet, kube-proxy) directly. All these communication is handled by API Server.

API server is responsible for user authentication/authorization also.

**Scheduler**

Scheduler is responsible for the assignment of PODS on Node.

**Controller**

There are many controllers like Node controller, Replica Controller. They follow the instruction given by end user and properly maintain to which they relate.

Node controller will check whether all the workers are available or not in Kubernetes Cluster. Node controller checks worker status after every 5 second. If one controller is down for 40 seconds, then Node controller will make that worker status as Not-reachable. If that worker will not come to UP state in next 5 minutes, then Kubernetes will transfer all the running pods on other workers.

**Kubelet**

Is responsible for creating and maintain the pods and nodes. If you create a container manually through docker, then maintenance of that container is not handled by Kubelet.

**Kube-proxy**

Is responsible to setup the traffic rule for the communication on the different on workers.

Always install bash completion for better experience, which will help to complete commands.

Kubernetes supports multiple container runtime like docker, rocket, wordlet.