Kubernetes

- K8S is a container orchestration technology that creates, deploy and manages clusters(bunch of docker containers)
- It schedules, runs and manages isolated containers which are running on virtual/physical/cloud machines
- Convert isolated containers running on different H/W into cluster
- All 3 clouds support Kubernetes
- K8S originated at Google

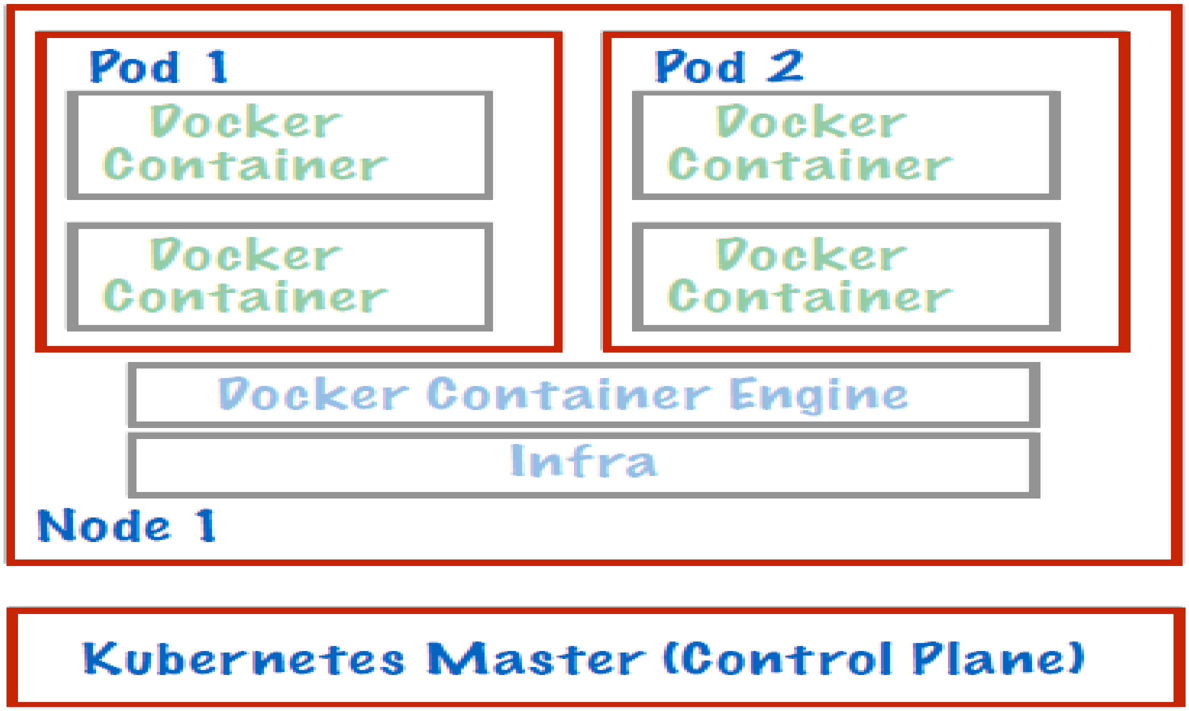
Features of Kubernetes

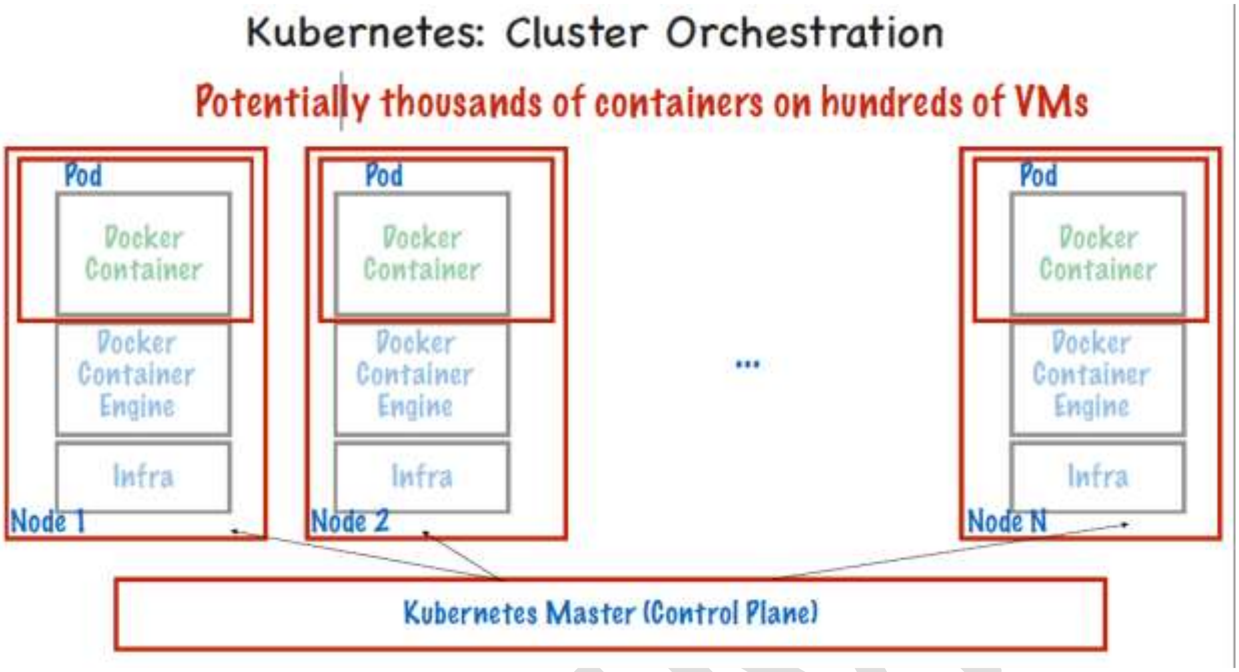
- Orchestration (Clustering of any no of Containers running on different H/W)
- Auto-Scaling (more clients? More demand)
- Auto- Healing (new containers in place of crashed containers)
- Load-Balancing (Distribute client requests)
- Platform Independent (Cloud/Virtualization/Physical)
- Fault tolerance (Node/Pod failures)
- Rollback (Going back to previous versions)

Kubernetes

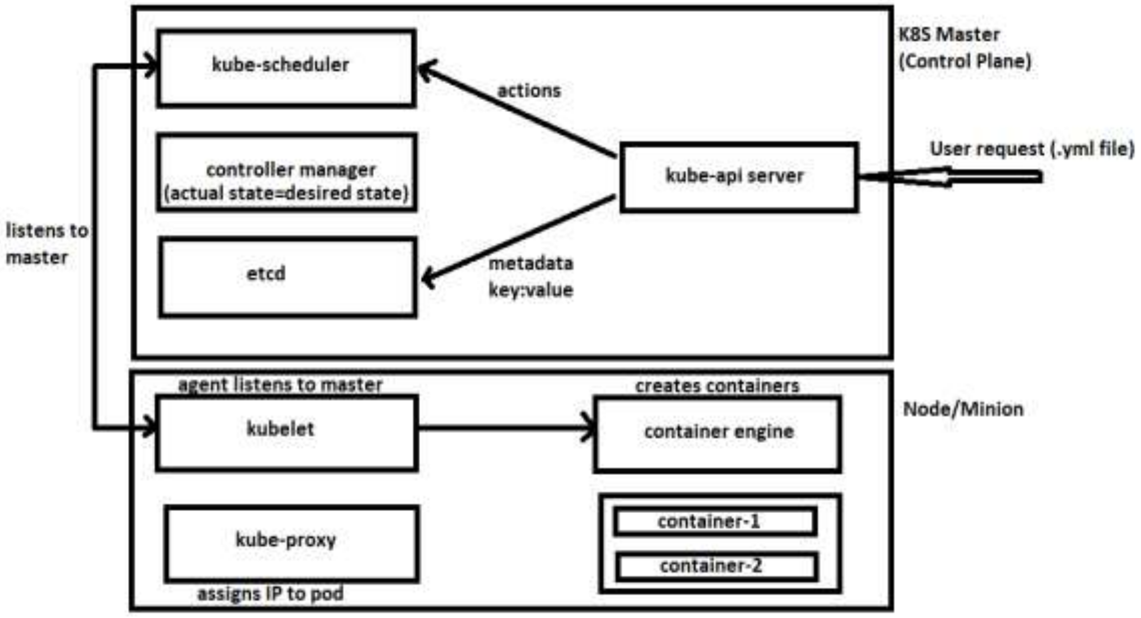
PODS

- Atomic unit of deployment in K8S
- Consists of 1 or more tightly coupled containers
- Pod runs on node, which is controlled by master





Kubernetes Architecture



Working with Kubernetes

- We create manifest (.yml)
- Apply this to cluster (to master) to bring into desired state
- Pod runs on node, which is controlled by master

Role of master node

- Kubernetes cluster contains containers running on Bare metal/VM Instances/Cloud instances/all mix.
- Kubernetes designates one or more of these as master and all others as workers
- The master is now going to run set of K8S processes. These processes will ensure smooth functioning of cluster. These processes are called "Control plane"
- Can be Multi-master for high availability
- Master runs control plane to run cluster smoothly

Constituents of Control plane

kube-apiserver:

- This apiserver interacts directly with user (i.e. we apply yaml or json manifest to kube-apiserver)

- This kube-apiserver is meant to scale automatically as per load.
- Kube-api server is front end of control plane

etcd : (cluster store)

- Stores metadata and status of cluster
- etcd is consistent and high available store (key-value store)
- source-of-touch for cluster state (info about state of cluster)

kube-scheduler:

- when users make request for the creation & management of pods, kube-scheduler is going to take action on these requests.
- Handles pod creation and management
- Kube-scheduler match/assign any node to create and run pods

Controller-manager:

- Make sure actual state of cluster matches to desired state
- 2 possible choices for controller manager
- if K8S on cloud, then it will be
    - Cloud-Controller-Manager
- if K8S on non cloud, then it will be
    - Kube-Controller-Manager

Nodes (kubelet & Container engine)

what runs on each node of cluster?

node/minion is going to run 3 imp pieces of software

Kubelet:

- Agent running on the node
- Listens to kubernetes master (eg: pod creation request)
- Port 10255
- Sends success/fail reports to master

Container engine: (Docker)

- Works with kubelet
- Pulling images
- Start/Stop containers
- Exposing containers on ports specified in manifest

Nodes (kube-proxy)

- Kube-Proxy: (assigns IP to each pod)
- It is required to assign IP addresses to pods (dynamic)
- Kube-proxy runs on each node & this make sure that each pod will get its own unique IP address.

Above 3 components collectively consists "node“

- K8S for Hybrid & Multi-cloud
- Hybrid : On premise datacenter + Public cloud
- Multi cloud : More than one public cloud provider

What is a POD?

- Atomic unit of deployment in kubernetes
- Consists of 1 or more tightly coupled containers
- Pod runs on node, which controlled by master
- Kubernetes only knows about pods (doesn't know about individual containers)
- Cannot start containers without a pod
- 1 pod usually contains 1 container
- Multi-container pods are possible too
- Such containers are tightly coupled
- Multi container pods;
- Share access to memory space
- Connect to each other using localhost:<container port>
- Share access to the same volumes
- Atomic unit of kubernetes
- Containers within pod are deployed in an all-or-nothing manner
- Entire pod is hosted on the same node (Scheduler will decide about which node)

- Pod runs on which node - Scheduler will decide

POD Limitations

No auto-healing or scaling

Pod crashes? must be handled at higher level

- Replica set

- Deployment

Higher level kubernetes objects

Replication set:

- Scaling & Healing

Deployment:

- Versioning and rollback

Service:

- Static (non-ephemeral) IP and networking

Volume:

- Non-ephemeral storage

Important

- kubectl - single cloud
- kubeadm - on premise
- kubefed – federated

2 Object management methods

- Imperative method
- Declarative method
- Imperative : We say actions that we want kubernetes to take
- Declarative : We tell only require output, won't tell how to get

Test environment setup (All 3 machines)

Launch 3 ubuntu servers (Master must have 2 cpu cores & 4GB RAM)

ssh into master

- sudo apt-get update
- Install https package
- sudo apt-get install apt-transport-https

This(https) is needed for intra cluster communication (particularly from control plane to individual pods)

Install docker

- sudo apt install docker.io -y
- docker --version
- sudo systemctl start docker
- sudo systemctl enable docker

Setup open GPG key. This is required for intra cluster communication. It will be added to source key on this node. i.e. when k8s sends signed info to our host, it is going to accept those info b'cos this open GPG key is present in the source key.

- sudo curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add

Edit sources.list file (apt-get install nano)

- sudo nano /etc/apt/sources.list.d/kubernetes.list
- deb http://apt.kubernetes.io/ kubernetes-xenial main

(debian s/w repo) (url from where to get s/w) (distribution) (name of component)

- sudo apt-get update
- Install all packages
- sudo apt-get install -y kubelet kubeadm kubectl kubernetes-cni

Bootstrapping the Master Node (in Master)

To initiate K8S cluster (Be in master)

- sudo kubeadm init

Copy the command to run in nodes & save in note pad

Creates both .kube and its parent directories (-p)

- mkdir -p $HOME/.kube

Copy configurations to kube directory (in config file)

- sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

Provide user permissions to config file

- sudo chown $(id -u):$(id -g) $HOME/.kube/config

Deploy flannel node network for its repository path. Flannel is going to place a binary in each node.

- sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
- sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/k8s-manifests/kube-flannel-rbac.yml

Configure worker nodes (in nodes)

Master has provided one command. That we have to run in each nd every node(add sudo in command).

- sudo kubeadm join --token ....

Individual token and discovery token along with its IP are used by master etcd cluster for establishing mutual trust b/w master and node.

These were provided by master and are being used by node.

This node has joined the cluster:

Certificate signing request was sent to master and a response was received

The Kubelet was informed of the new secure connection details.

So run the same command in all other nodes.

Go to master

- sudo kubectl get nodes

Can see 1-master

2-nodes