Kubernetes

**Architecture**

**Kubectl**

is used to communicate b/w client and api server

It will convert request into rest

Api server(Default tasks) –

Authenticate

Authorize

Validate – validation of requests (like format)

Talk to ETCD(NoSQL database, here it will perform CRUD operations )

MASTER- KUBE- [“API Server, ETCD ,Scheduler, Controller”]

WORKER- [KUbelet, KUbe-proxy, Containerd/CRIO/Pods]

When a request come to create a pod,

then api will update it into etcd then

scheduler then find out best machine to execute and update back in ETCD

controller will control everything, so many controllers like nodecontroller

Only API server will talk directly to ETCD all others need w.r.t api(scheduler, controller, users etc..)

Kubelet(local manager) – will used to communicate, it will communicate to apiserver

Containerd(Container runtime) – It is similar like docker

Kube-proxy – it will be in every working node(every node) and find out networking of every node

#commands

kubectl run mypod –image=nginx

kubectl get pod

kubectl get nodes

POD –

Group of one or more containers(recommend: -1), and they will be independent

All the containers in pod are co-located and co-scheduled(together they come up) and share namespaces

Stages to create a pod

1. When make a request kubectl send certificates, Request(GET,POST…) to API server
2. Then API will do authenticate, authorize, validate then update the details like podnaame image, Desired state etc..
3. Then scheduler will find out the node to be execute and update it into etcd
4. Controller will find the diff in states do the tasks

apt-cache madison kubeadm | tac

kubeadm join 172.31.36.255:6443 --token is2nqa.j452slpddyejn27x \

--discovery-token-ca-cert-hash sha256:dbf1b7c7e1290ac2305a2bebff8c694133d911dd74f361636585caebe725a6e0

kubectl taint node kmaster node-role.kubernetes.io/control-plane:NoSchedule

[“job controller, node controller, replica controller]”

Kubectl

Yaml file 🡪 In spec under the template its will always about pod

 kubectl scale rs frontend --replicas=3

replicaset.apps/frontend scaled

Rollout is used to check whether it is in stable state or not [“kubectl rolloutstatus deploy <deploy\_name>”]

To rollback to a previous version 🡪 kubectl rollout undo deply nginx-deployment --to-revision=2

Label is used to identify the item using label name

Kubectl grt pod -l env=development [“name=label\_name”]

kubectl get ns

NAME             STATUS  AGE

default          Active  7h4m

kube-node-lease  Active  7h4m

kube-public      Active  7h4m

kube-system      Active  7h4m

root@kmaster:~# kubectl create ns devanns

namespace/devanns created

root@kmaster:~# kubectl run pod -n devanns --image=nginx

pod/pod created

root@kmaster:~# kubectl get po -n devanns

NAME  READY  STATUS   RESTARTS  AGE

pod   1/1    Running  0         15s

Here we can create a namespace and use thet for our pourpose