KUBERNETES
Core

Kubernetes used for

Host applications in containers so you can easily deploy as many as you want and enable communication between different services within the app.

Uses CRI: Container runtime interface

3 CLI options:

1) ctr
2) nerdctl
3) crictl

ETCD

is a distributed reliable key-value store that is Simple, Secure & Fast

Name	Age	Location	Salary	Grade
John Doe	45	New York	5000	
Dave Smith	34	New York	4000	
Aryan Kumar	10	New York		A
Lauren Rob	13	Bangalore		C
Lily Oliver	15	Bangalore		B

ETCD in Kubernetes

Store information about the cluster such as: Nodes, PODs, configs, secres, accounts, roles, bindings and others

```
{
  "name": "John Doe",
  "age": 45,
  "location": "New York",
  "salary": 5000
}
```

```
{
  "name": "Dave Smith",
  "age": 34,
  "location": "New York",
  "salary": 4000,
  "organization": "ACME"
}
```

Kube API Server

1) Authenticate user
2) Validate request
3) Retrieve data
4) Update ETCD
5) Scheduler
6) Kubelet

Kube Controller Manager

Manages various controllers in Kubernetes

- Watch status
- Remediate situation

Controller: a process that continuously monitors the state of various components within the system and works towards bringing the whole system to the desired function state.

• Node controller: monitor the Kube API every 5 seconds, have a grace period of 40 seconds before making it unreachable, and POD Evocation Timeout after 5 min.

• Replication Controller:monitor the replica sets and the desire number of pods are available within the set > If a POD dies, it’s create a new one.

Kube Scheduler

Deciding which POD goes to which Node.
Take each POD and find the best Node.

Kubelet

Manage, plan, schedule, monitor nodes.

- Register the node with the Kubernetes cluster
- Create PODs
- Monitor Node and PODs

Kube-Proxy

A process that run on each node in the Kubernetes cluster. His job is to look for a new services, and every time a new service is created, it create the appropriate rules on each node, to forward traffic to those services to the back-end PODs by the IP-Table-Rules.

PODs

The goal: deploy out application in the form of containers on a set of machines that are configure as worker nodes in a cluster.

The containers are encapsulated into Kubernetes object- POD

POD: a single instance of an application. It's the smallest object in Kubernetes.

When there are a lot of requests -> Creates more PODs in the same Node.

A single Node can have multiple containers.

Helper Container: supports the main container by handling tasks like monitoring, proxies or syncing data

PODs with YAML

```yaml
apiVersion: v1 # String
kind: Pod # String
metadata: # Dictionary
  name: my-app
  labels: # Dictionary
    app: my-app
    type: front-end
spec:
  containers: # List of Dictionaries
    - name: my-app-container # String
      image: myregistry.com/my-app:1.0 # String

```

PODs Commands

Show all the PODs:

$ kubectl get pods

Create a new POD:

$ kubectl run <name> --image= <type>

Lookat PODs with details:

$ kubectl describe pod <pod-id>

Show the Nodes:

$ kubectl get pods –o wide

Create a POD: pod-definition YAML file

$ kubectl run <name> --image=<type> --dryrun=client –o yaml

$ kubectl create –f <name>.yaml