1) What is K8S

- Open Source Container orchestration tool
- Developed by google
- Helps manage containerized applications in different deployment environments.

Need for Container orchestration tool

- Trend from Monolith to Microservices
- Increased usage of containers
- Demand for a proper way of managing those hundreds of containers.

Features do orchestration tools offer

- High Availability or no downtime
- Scalability or high performance
- Disaster recovery - backup and restore

K8S Architecture

- There is one master node and multiple worker nodes
- Where each worker node has a kubelet process running on it

  Kubelet :- Is actually a kubernetes process that makes it possible for the cluster to communicate each other and execute some tasks on those nodes.

- Each worker node has containers of different applications deployed on it.
- On masternode actually runs several K8S processes that are absolutely necessary to run & manage the cluster properly.

  API Server :- Also is a container an api server entrypoint to K8S cluster

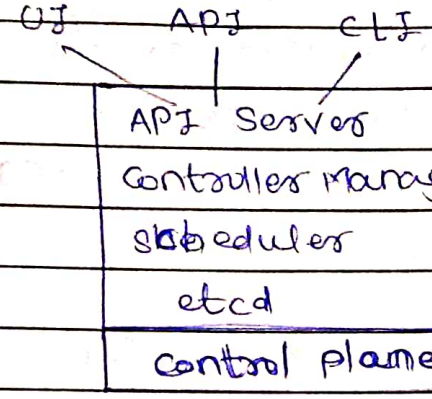
  UI, API, CLI → all talk to api server

Controller Manager :- keeps track of whats happening in the cluster

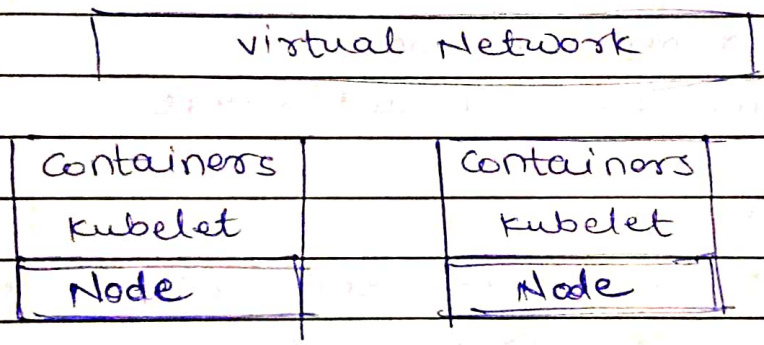
Scheduler :- scheduling containers on different nodes based on the workload

etcd :- K8S backing store

- virtual Network :- creates one unified machine

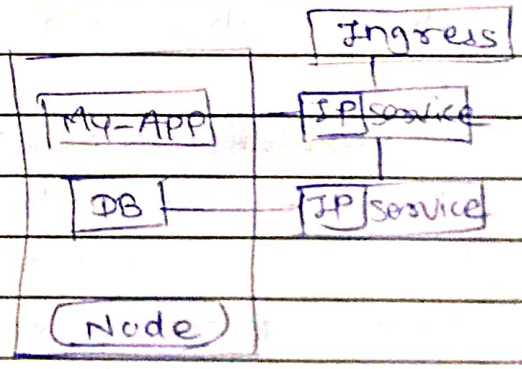


Control Plane Nodes
- handful of master processes
- much more important



Worker Nodes
- higher workload
- much bigger and more resources

Components In K8S



① pod

- smallest unit of K8S
- Abstraction over container
  (You only interact with the K8S layer)
- Usally 1 Application / container per pod
- Each pod Contains its own IP address through that they can communicate
- pods Can die easily
- New IP address on re-creation

② Service (svc)

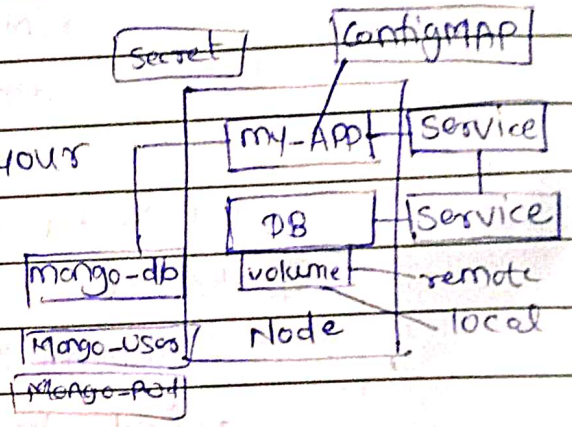
- permanent IP address
- Lifecycle of pod & service not connected.
- You specify the type of service on creation
- Internal service is the default type

③ Ingress (ing)

- Instead of service the request goes first to ingress and it does the forwarding then to the service

④ Configmap (cm)

- External configuration of your application



⑤ Secret

- Used to store secret data
- Reference secret in Deployment / pod

⑥ vol

- storage on local machine
- or remote, outside of the k8s cluster

⑦ Deployment

Define blueprint for pods

- specify how many replicas you want to have
- Blueprint for "my-app" pods
- You create Deployments
- Abstraction of pods

⑧ StatefulSet (sts)

- for databases

Deployment = for stateless Apps
statefulset = for stateful Apps or Databases

Each Configuration file has 3 parts:-

① metadata

e.g. metadata:
name: nginx-deployment

② specification

e.g. spec:
replicas: 2
selector:
template:

- Attributes of "spec" are specific to the kind

③ status

- Automatically generated and added by K8s.
- etcd holds the ~~cust~~ current status of K8S container / component.

YAML Configuration Files

- "human friendly" data Serialization standard for all programming languages
- syntax: strict indentat^n
- code editors have plugins for YAML syntax validation
- Store the config file with your Code
- or own git repo.

* What is Minikube?

It is a open source tool. Is basically one node cluster where the master processes and the worker processes both run on one node
And this node will have a docker Container

runtime pre- installed.

* What is kubectl?
- command line tool for K8S cluster
- The most powerful of 3 clients
- Worker process enables pods to run on node

start a cluter passing docker as a driver option
minikube start --driver docker
check a status
minikube status

* Minikube has kubectl as dependency
* No separate installation necessary
Displays all the nodes in a cluster
kubectl get node

kubectl CLI → for configuring the Minikube cluster

Minikube CLI → for start up / deleting the cluster

Kubernetes is the Greek word for helmsman or Captain of a ship

Kubernetes
- Container management (orchestration) tool
- developed by Google lab (and later donated to CNCF) [cloud Native Computing Foundation]
- open source
- Written in Golang

What is Container Management tool?

Container orchestration tool or engine automates deploying, scaling and managing containerized application on a group of servers.

e.g. Kubernetes

Docker swarm

Apache Mesos Marathon

Kubernetes

↓

Container Management Tool

↓

Manages Containerized Applications

↓

Applications available on a container platform like Docker

→ Deploying

Scheduling

scaling

Load balancing.

What is Docker?

~~to~~ Docker is a tool designed to make it easier to deploy and run applications by using containers.

Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package.

organizations have to use multiple containers to
Ensure availability
Load balancing
Scale up and down based on user load

deploying
scheduling
scaling
load balancing
batch execution
roll backs
monitoring

} We have to do that
Here container management
tools comes into picture

Docker → creates containers

Kubernetes → manages containers

* Features of kubernetes

① Automatic bin packing -

We have 5 servers each having 10 GB of memory (RAM) we have a list of jobs to run on these 5 servers Every job has different resource (memory) requirement

Kubernetes will take care of packaging these jobs (containers) in bins (servers) in the most efficient way

Kubernetes automatically packages your application and schedules the container based on the requirements and resources available.

Automatically places containers based on their resource requirements like CPU & Memory (RAM), while not sacrificing availability saves resources.

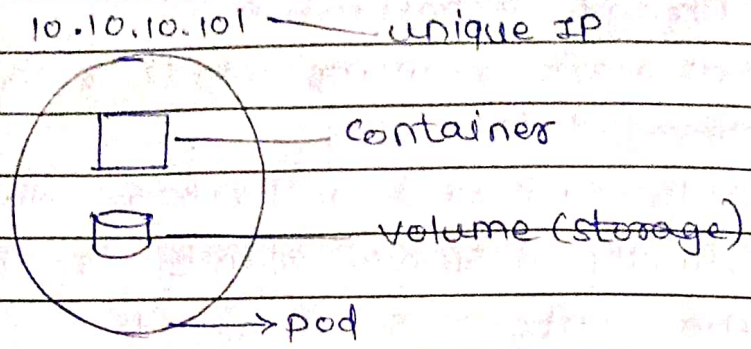
① Pods & Nodes -

We do not interact with containers directly. Containers are raped ~~into~~ inside of the functional unit which is called as a "pod".
Pod can have single container or multiple containers
pods are housed inside nodes.
A node can have a single pod or multiple pods.

When you specify a pod, you can optionally specify how much CPU and memory (RAM) each container needs.
When containers have resource requests specified, the scheduler can make better decisions about which nodes to place pods on

② Service discovery & load balancing -

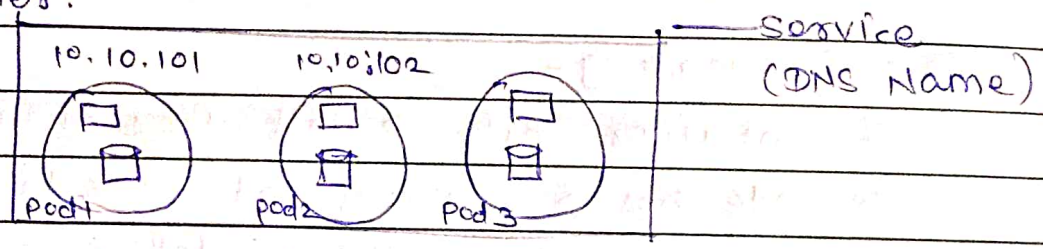
- How k8s orginizes containers
- k8s doesn't run containers directly instead it wrap one or more containers into a higher-level structure called a pod
- A pod contains
  - an application container (or, in some cases, multiple containers)
  - storage resources
  - a unique network IP



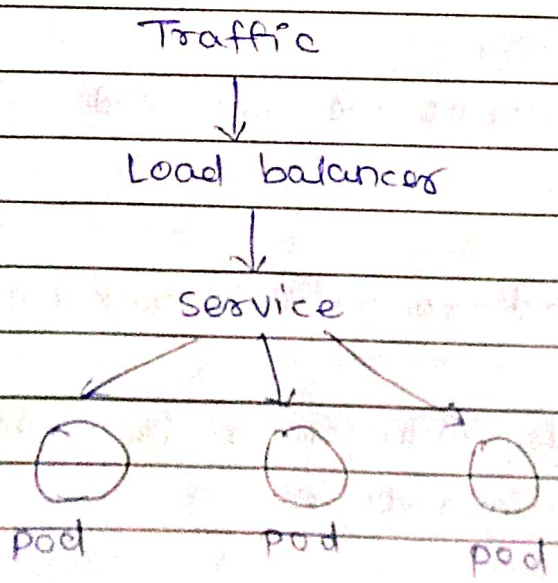
Pods that have the same set of functions are abstracted into sets, called services.

Kubernetes gives pods their own IP addresses and a single DNS name for a set of pods, and can load-balance across them

A kubernetes service is a set of pods that work together.



With this system, K8S has control over network and communication between pods and can load balance across them.



③ storage orchestration -

- Containers running inside a pod may need to store data
- Pods can have a storage volumes
- Usually a single volume is shared within all the containers in a pod.

Kubernetes allows to mount the storage system of your choice Local
cloud (AKS)
Network (NFS)
↓
~~Network file system~~

④ Self - Healing -

If container fails → restarts container
If node dies → replaces and reshedule containers on other nodes.

If container does not respond to user defined health check → kills container

All the processes are controlled by "Replication Controller"

⑤ Automated rollouts & rollbacks -

Rollout: deploy changes to the application or its configuration

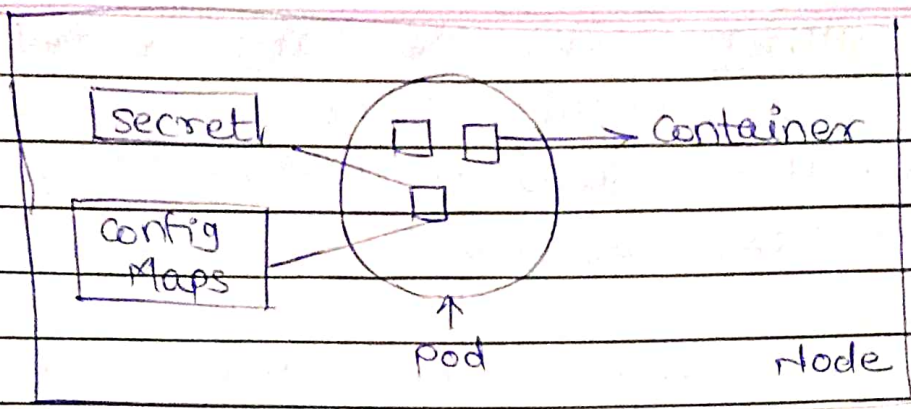
Rollback - revert the changes & restore to previous state

Kubernetes ensures there is no downtime during this process.

K8S progressively rolls out changes to your applicatn or its configuration, while monitoring applicatn health to ensure it doesn't kill all your instances at the same time. If something goes wrong, K8S will rollback the changes for you.

⑥ secret and configuration Management -

secret	config Maps
In K8S sensitive data like passwards, keys, tokens are handled using secrets	In K8S configurations are handled using config Maps
Secret is a K8S object	Config map is a K8S object
created outside pods and containers	created outside pods and containers
Makes sensitive data portable and easy to manage.	Makes configuration portable and easy to manage
** Secrets is a K8S object that separates sensitive data from pods and containers	Config Map is a K8S object that separates configurations from pods & containers.



K8s manages secrets & configuratn details for an application separately from the container image. Deploy and update secrets and application configuratn without rebuilding your image and without exposing secrets in your stack configuratn.

** Secrets & Configurations are stored in ETCD
ETCD is a key-value datastore (database)

The max size limit for a secret is 1 MB

(7) Batch Execution -

- Batch jobs require an executable/process to be run to completion
- In K8s run to completion jobs are primarily used for batch processing.
- Each job creates one or more pods
- During job execution if any container or pods fails, job controller will reschedule the container pods on another node
- Can run multiple pods in parallel and can scale up if required
- As the job is completed the pods will move from running state to shut down state

K8S supports batch execution, long-running jobs and replaces failed containers.

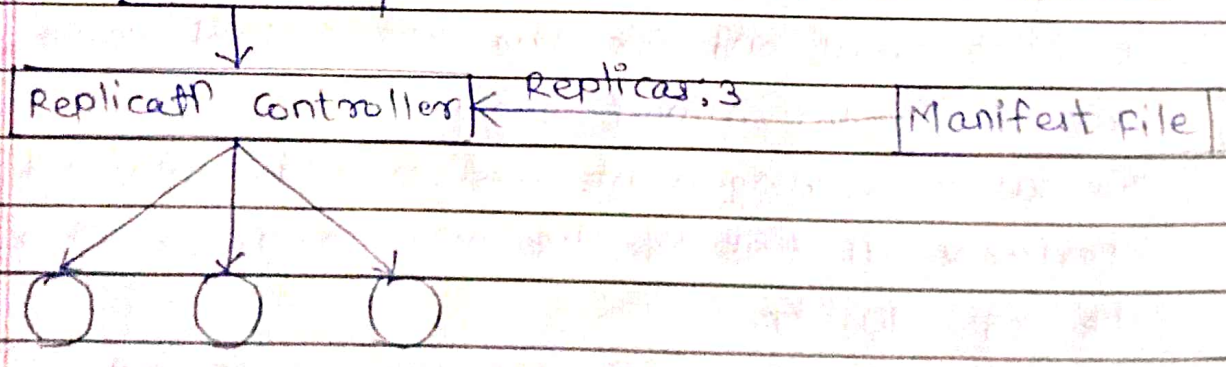
⑧ Horizontal Scaling -

In K8S, we can scale up or down the containers
- using commands
- from the dashboard (K8S ui)
- automatically based on CPU usage

Replication Controller	Manifest File	Horizontal Pod Autoscaler

- Replicatn controller is a structure that enables to create multiple pods, then make sure that the no. of pods always exists.
- If the pod crashes, the replicatn controller replaces it.
- Replicatn controller gets the no. of pods to run and make available at any time from the information provided in manifest file

Horizontal pod Autoscaler	Automatically scales the no. of pods in a replicatn controller based on observed CPU utilization or with custom metrics



Replicatn controller - ensures that the desired no. of pods are maintained always.

* Basic Architecture of K8S -

K8S supports: (V1.16)

No more than 5000 nodes

150000 total pods

300000 total containers

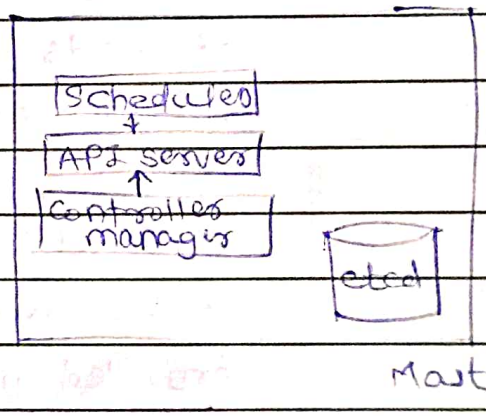
100 pods per node

Master node -

- responsible for managing the cluster
- Monitors nodes & pods in a cluster
- When a node fails, moves the workload of the failed node to another worker node

4 components of master node

- API server - for all communications (JSON over HTTP API)
- scheduler - schedules pods on nodes
- Controller manager - runs controllers
- Etcd - open source, distributed key-value database from CoreOS



① API Server

- APIs allow applicatn to communicate with one another
- It is the front-end for the K8S control plane
- Exposes API for almost every operatn
- The users, management devices and command line interfaces all talk to the API server to interact with the K8S cluster
- users interact with the API using a tool called kubectl
- kubectl is a command line utility to interact with

K8S API

- kubectl is a Go language binary.

② Scheduler -

- Schedules pods across multiple nodes
- Component on the master that watches newly created pods that have node assigned, and selects a node for them to run on
- The scheduler obtains from etcd, via the API Server, resource usage data for each worker node in the cluster.

** - Scheduler gets the info. for hardware configuratn from configuratn file and schedules the pods on nodes accordingly.

③ Control Manager -

- This is a component on the master that runs controllers
  - kube-controller-manager
    - Node controller
    - Replicatn controller
    - Endpoints controller
    - service account & token controller
  - cloud-controller-manager

- Responsible for overall health of the cluster
  - Ensures nodes are running all the time
  - Correct no of pods are running as per spec file

* - Basically, each controller is a separate process but to reduce complexity, they are all compiled into a single binary & run in a single process.

Node controller - Responsible for noticing & responding when nodes go down

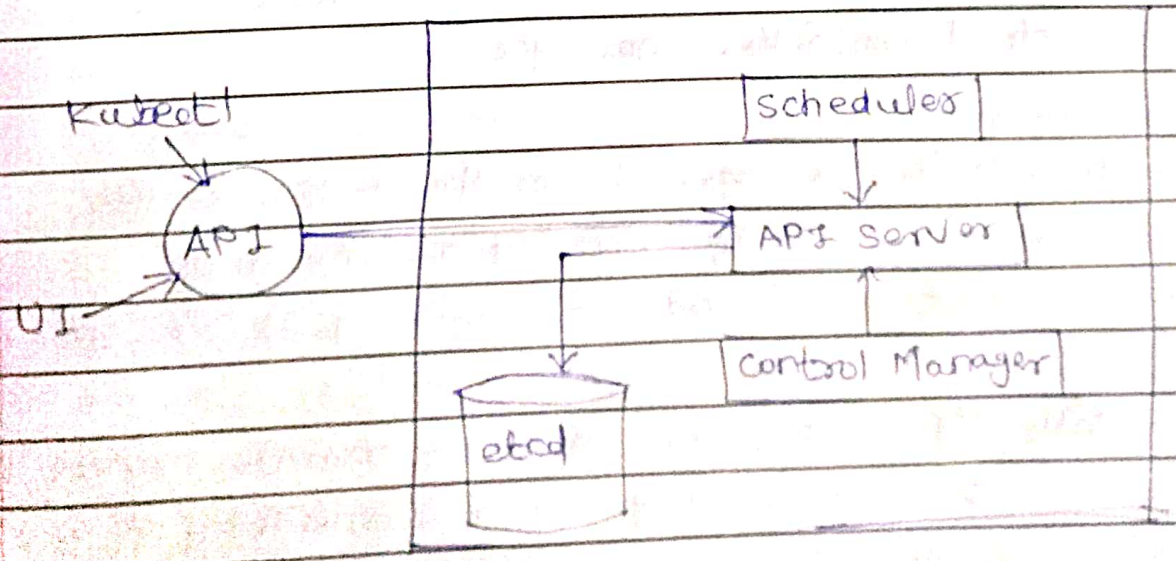
Replicatn controller - Responsible for maintaining the correct no. of pods for every replicatn controller object in the system

Endpoints controller - populates the endpoints object (i.e. joins services & pods)

Service account & token controller - create default accounts and API access tokens for new namespaces

④ etcd -

- open source, distributed key-value databases from coreos
- consistent & highly - available key value store used as k8s backing store for all cluster data
- single source of truth for all components of the k8s cluster.
- ** out of all the master components, only the API server is able to communicate with the etcd data store
- etcd can be part of the k8s master or it can be configured externally.



PAGE NO.:
DATE: / /

Worker node -

- can be any physical or virtual machine where containers are deployed
- Every node in a K8S cluster must run a container runtime like Docker
- Has following components:

  Kubelet

  Kube-proxy

  Container-runtime

① Kubelet -

- Is an agent running on each node
- communicates with components from master node
- makes sure that containers are running in a pod
- The kubelet takes a set of podspecs that are provided through various mechanisms & ensures that the containers described in those podspecs are running & healthy
- In case any pod has any issue, kubelet tries to restart the pods on the same node or a diff. nodes.
- * The kubelet dosn't manage containers which were not created by K8S.

② Kube-proxy -

- A network agent which runs on each node responsible for maintaining network configuration and rules.
- Exposes services to the outside world
- Core networking components in K8S.
- * All worker nodes run a daemon called kube-proxy, which watches the API Server on the master node

for the addition and removal of services and endpoi-
nts.

③ Container runtime (pods and containers)

- The container runtime is the software that is responsible for running containers
- Kubernetes supports several container runtimes:
  Docker
  Container D
  crio-
  Rktlet
  K8S CRI (Container Runtime Interface)
- ** K8S does not have the capability to directly handle Containers.
- In order to run & manage a container's lifecycle, K8S requires a container runtime on the node where a pod and its containers are to be scheduled.

④ Addones for DNS, Dashboard, monitoring, logging etc

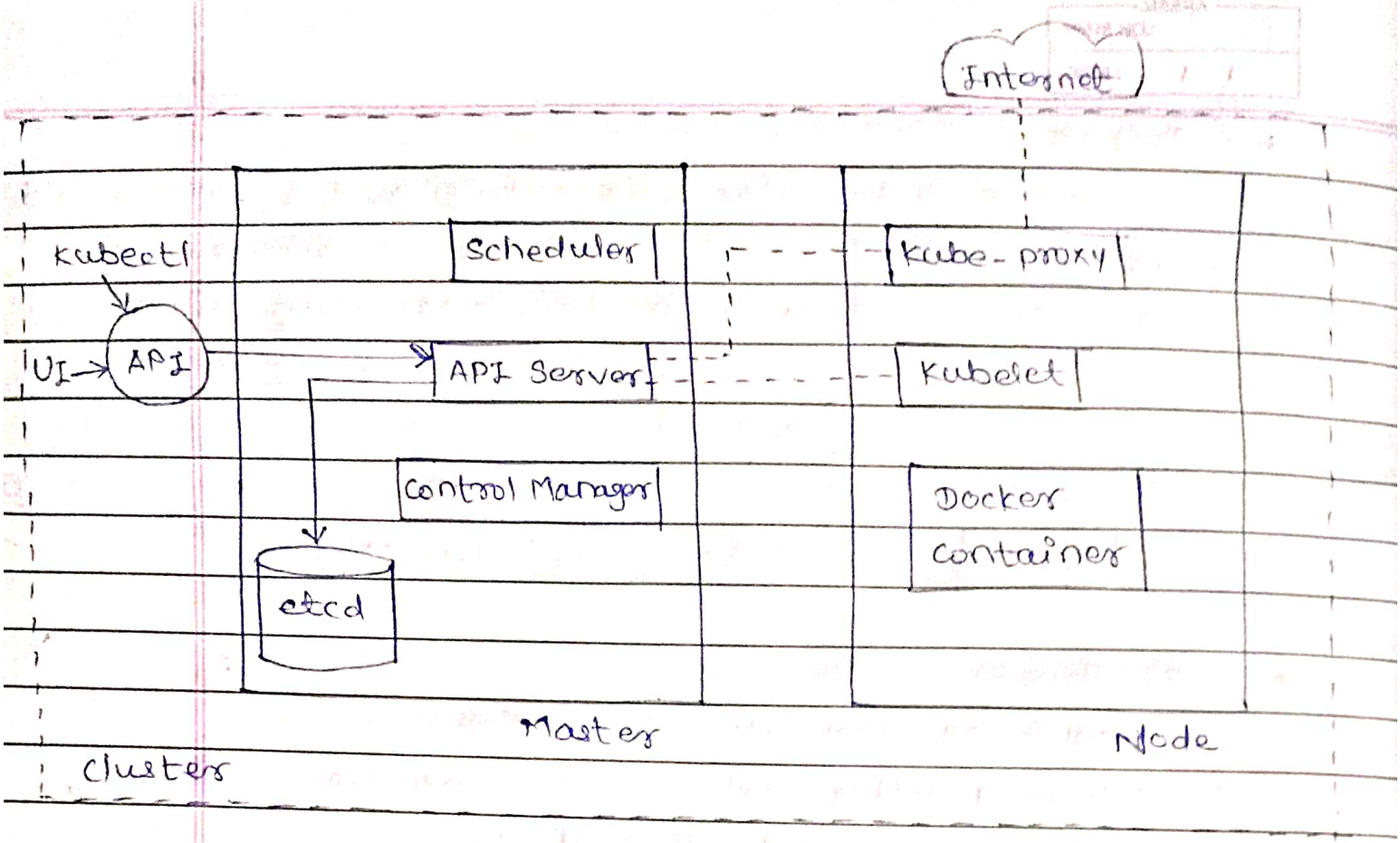
Add-ones extend the functionality of K8S

Dashboard - a general purpose web-based user interface for cluster management

Monitoring - collects cluster-level container metrics & saves them to a central data store

Logging - collects cluter-level container logs and saves them to a central log store for analysis

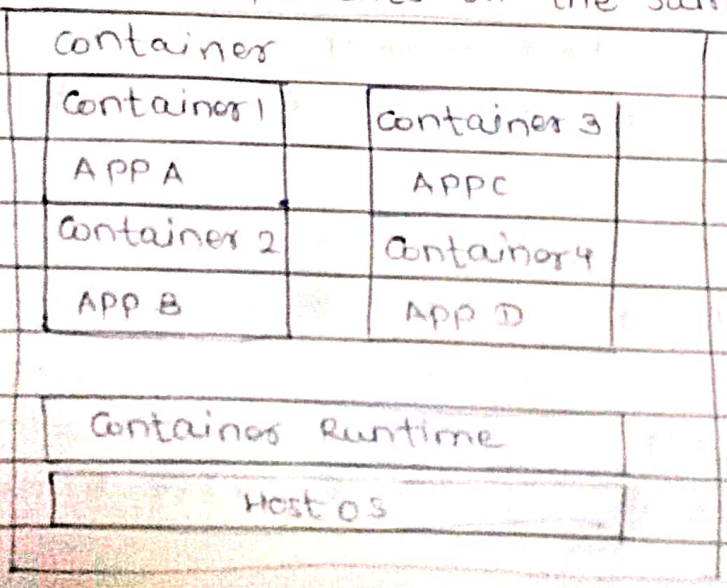
DNS - cluster DNS is a DNS server required to assign DNS records to K8S objects & resources.



* What are containers?

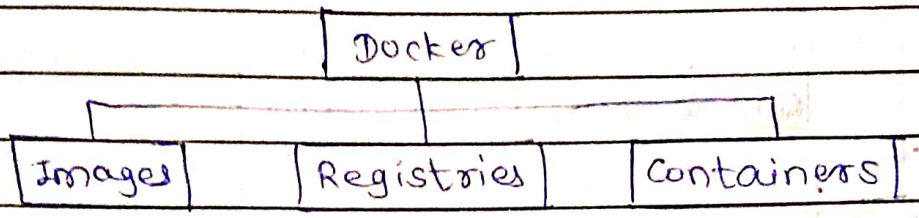
When an application is composed of only smaller numbers of large components, its acceptable to give a dedicated virtual machine and isolate their environments by providing each of them with their own OS instance

Containers are a form of operating system virtualization, which allows you to run higher no. of software components on the same hardware.



* Docker

Docker is a platform for packaging, distributing and running applications. It allows you to package your application together with its whole environment.



* K8S Benefits

- Simplifying Application Deployment
- Achieving Better utilization of Hardware
- Health checking and self-healing
- Automatic Scaling.

```
sudo apt-get update -y
sudo apt-get install curl
sudo apt-get install apt-transport-https
```

} Install some packages

* Install virtual box hypervisor

```
sudo apt install virtualbox virtualbox-ext-pack
```

* Installing minikube & kubectl

Goto official website and install minikube and kubectl

```
sudo kubectl version --client
```

* Installing Docker

Goto the official website of docker & install it.

Sudo docker run hello-world
sudo docker run busybox echo "Hello world"

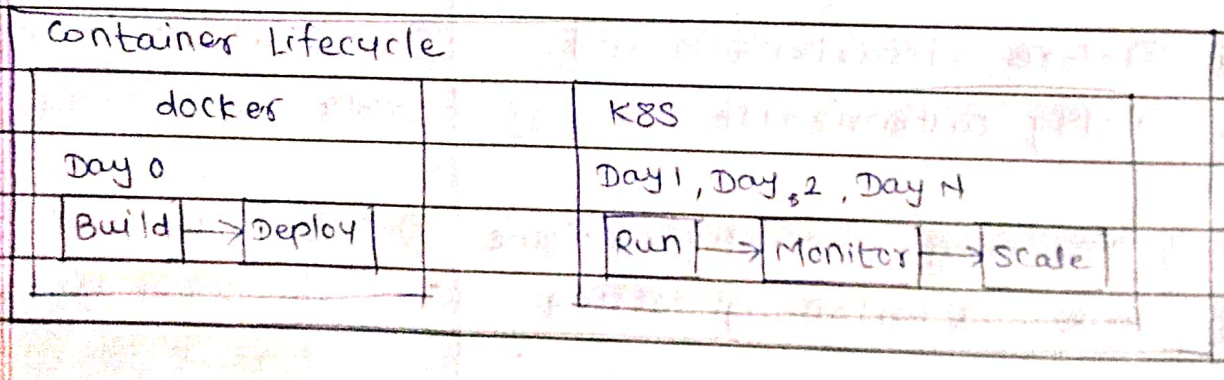
* start minikube cluster
sudo minikube start

* displays all of the namespaces created in K8S
sudo kubectl get po -A

* Access the dashboard of minikube
sudo minikube dashboard

* pull the ubuntu image
Sudo docker pull ubuntu
sudo docker run -it -d ubuntu
sudo docker ps -a
sudo docker images

* Docker + K8S -



containers created with Docker or any of its alternatives can be managed, scaled and moved by K8S, which also ensures failover management and health maintenance of the system.

PAGE NO.
DATE / /

* K8S vs Docker

	K8S	Docker
tup & stallatn	① It requires a series of manual steps to setup K8S master and worker nodes components in a cluster of nodes.	① Installing docker is a matter of one-liner command on linux platform like Debian, ubuntu and centos.
gging	② K8S provides no native storage soln for log data.	② Logging driver plugins are available in Docker 17.05 and higher.
nitoring	③ There are various open-source tools available for K8S applicatn monitoring like prometheus, Grafana, Influx DB etc.	③ They are various tools available for monitoring purposes like prometheus, Docker API etc
ode upport	④ supports upto 5000 nodes	④ supports upto 2000 nodes
mpatibility	⑤ More comprehensive and highly customizable.	⑤ Less extensive and customizable.
ing	⑥ You must manually configure your load balancing setting.	⑥ Does auto load balancing
nies the	⑦ Google, shopify, udemy, slack, etc.	⑦ Twitter, spotify, paypal, eBay etc.

* Containers Problems

Both Linux containers & Docker containers isolate the application from the host.

Faster, Reliable, Efficient, light-weight and scalable

But Not easily scalable

1. Containers could not communicate with each other
2. Containers had to be deployed appropriately
3. Containers had to be managed carefully
4. Auto scaling was not possible
5. Distributing traffic was still challenging

* What is Minikube

Is a tool that lets you run k8s locally. It runs a single-node k8s cluster on your personal computer.

* What is kubectl

· The kubectl command line tool lets you control k8s cluster.

sudo kubectl config get-clusters

sudo kubectl config delete-cluster

(deleting from config file)

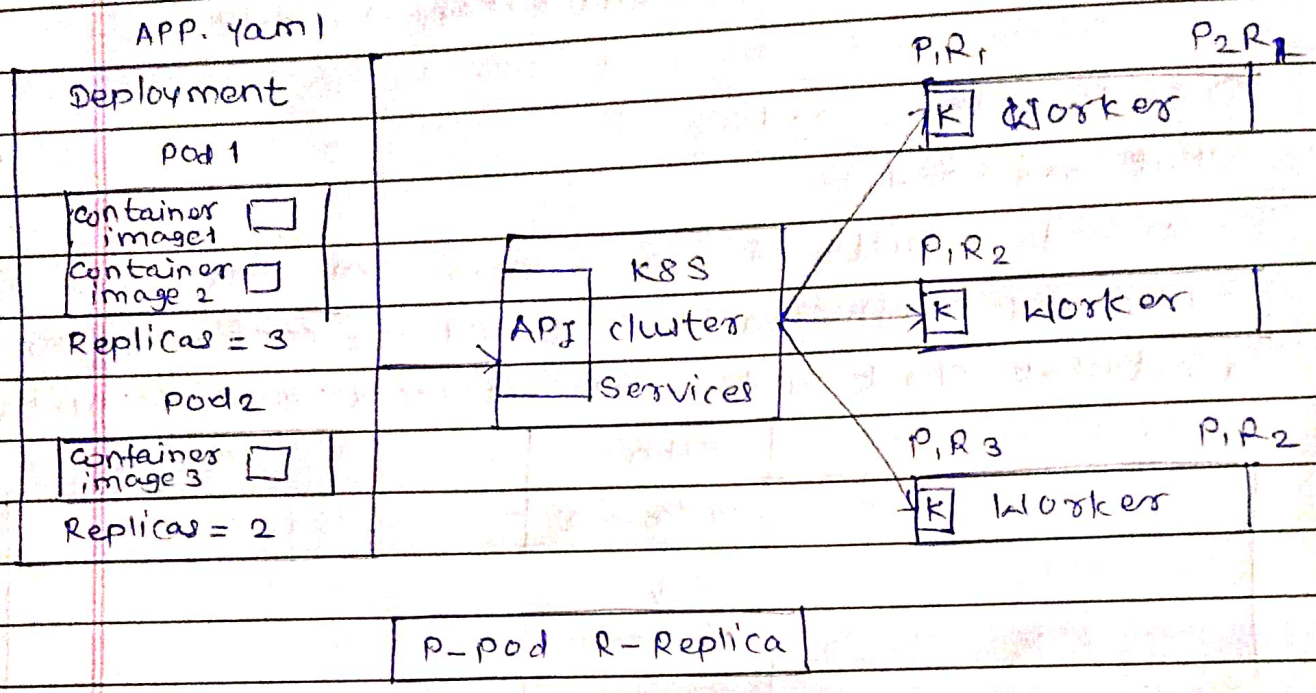
* Contexts

sudo kubectl config get-contexts

→ In k8s is a object which contains the set of access parameters for your cluster. It consists of cluster name-space and user triple. It allows you to quickly switch between different sets of cluster configuration

* **Kubernetes cluster-**

A cluster is a foundation of K8S engine. It has various components such as worker nodes, API's, kubelet processes, deployment files etc.



* **Problems to solve -**

Container-to-container communication
pod-to-pod communication
pod-to-service communication
External-to-Service Communication

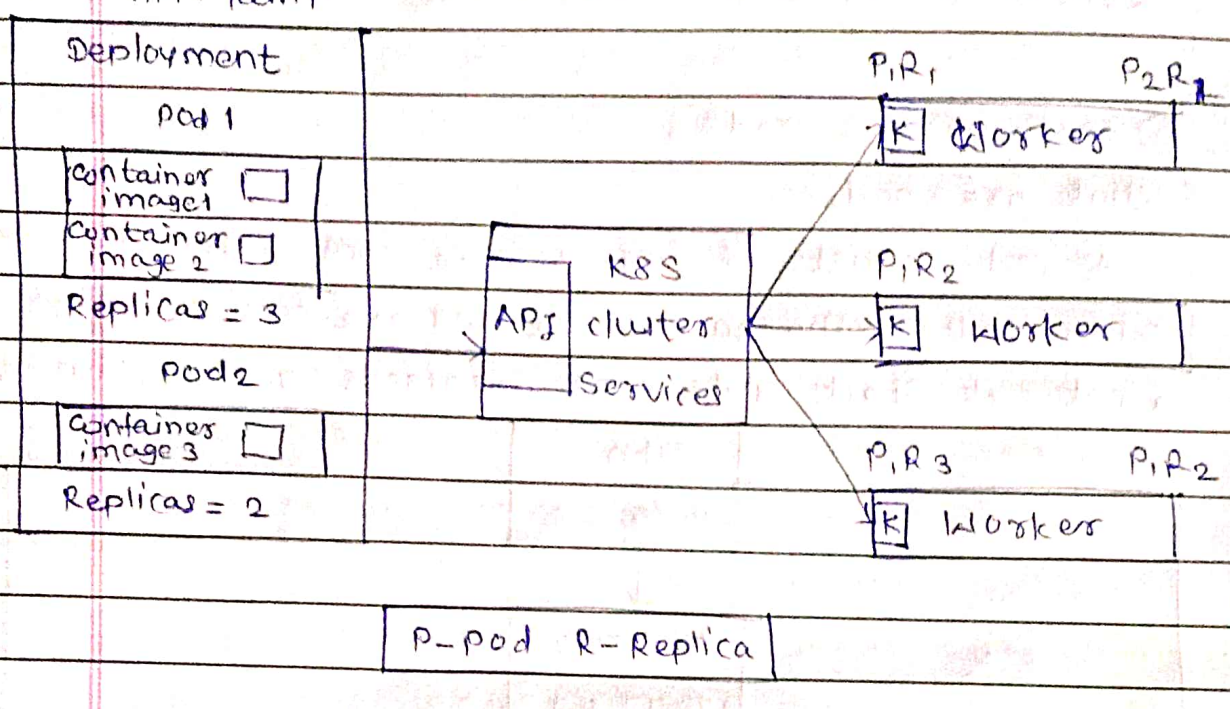
* **Requirements of K8S Model -**

K8S model has few fundamental requirements for any networking implementation.

① All containers can communicate with all other containers without NAT
② All nodes can communicate with all containers (and vise-versa) without NAT
③ The IP that a container sees itself as is the same IP that others see it as.

* Kubernetes cluster-

A cluster is a foundation of K8S engine. It has various components such as worker nodes, APIs, kubelet processes, deployment files etc.



* Problems to solve -

Container-to-container communication

Pod-to-pod communication

Pod-to-service communication

External-to-service communication

* Requirements of K8S Model -

K8S model has few fundamental requirements for any networking implementation.

① All containers can communicate with all other containers without NAT

② All nodes can communicate with all containers (and vise-versa) without NAT

③ The IP that a container sees itself as is the same IP that others see it as.