kubernetes

What is Kubernetes ?

Kubernetes is a container management system developed by Google

Kubernetes is an open-source, portable system for automatic container deployment and management

Kubernetes eliminates many of the manual processes involved in deploying and scaling containerized applications

kubernetes

Features of Kubernetes

Automates various manual processes and controls server hosting and launching

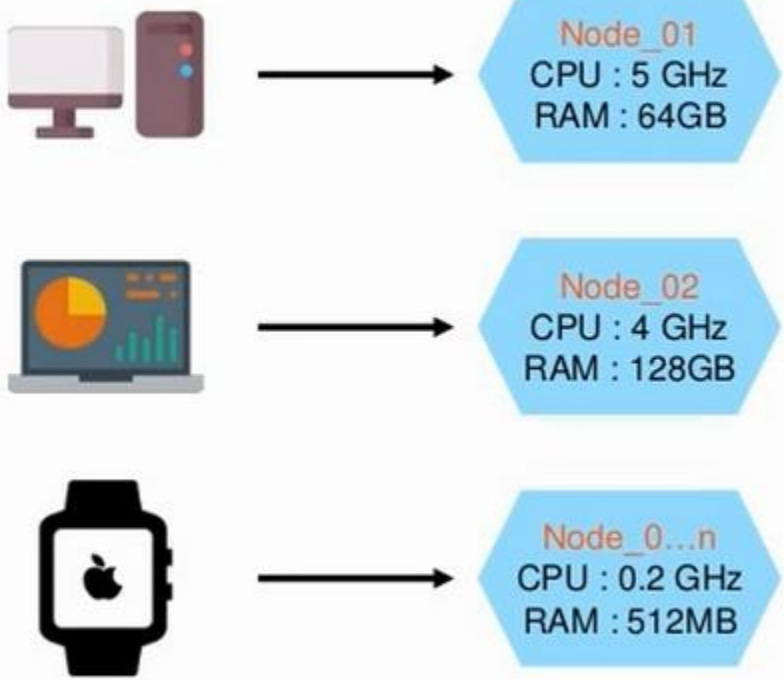
Manages containers, offers security, networking and storage services

It monitors and continuously checks the health of nodes and containers

Automatic rollback for changes that go wrong

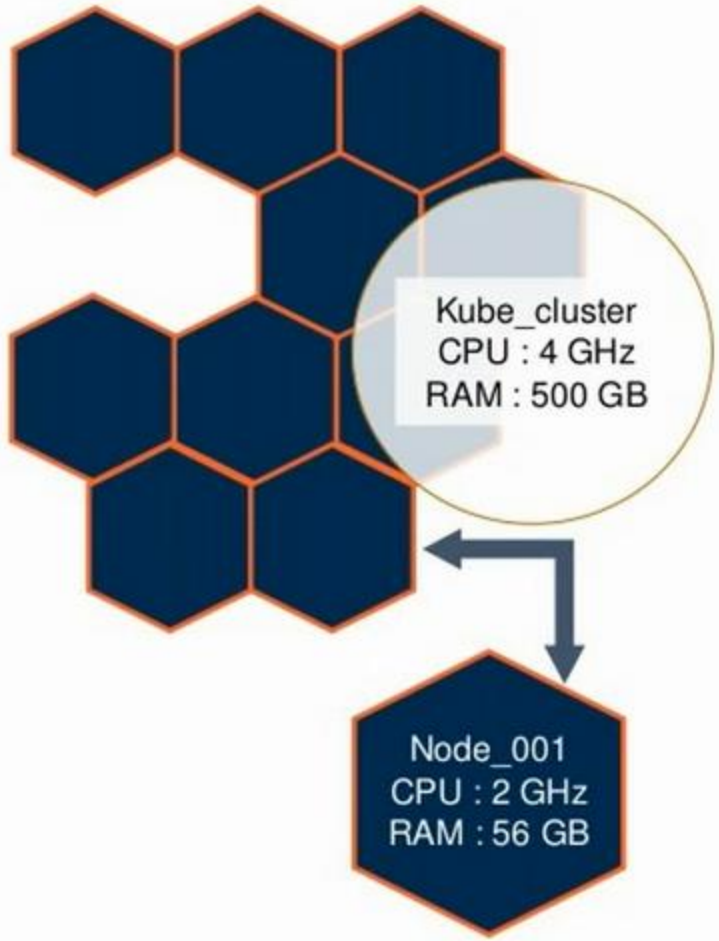
Kubernetes mounts and adds a storage system to run apps

Hardware Component - Nodes



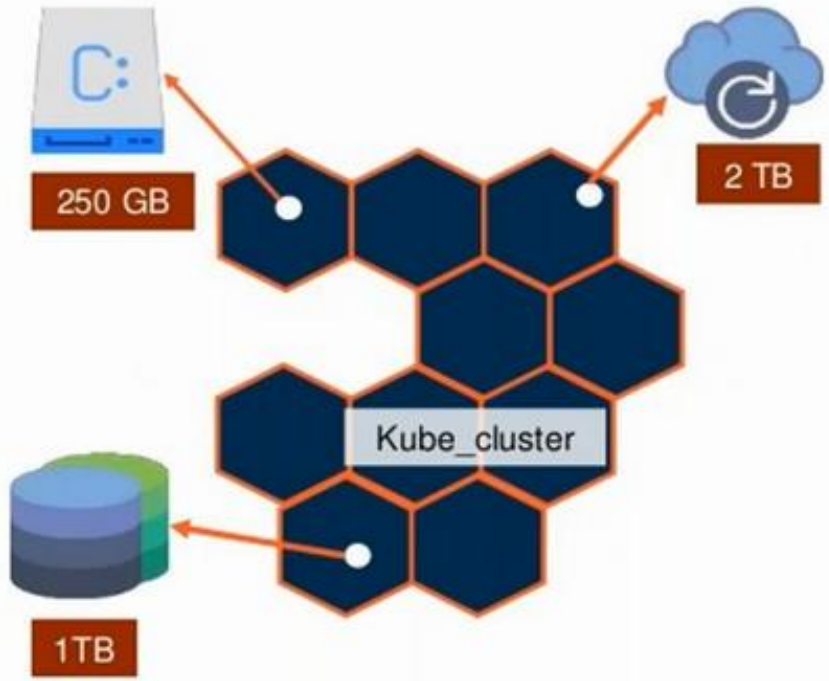
- A node is the smallest unit of hardware in Kubernetes. It is a representation of a single machine in the cluster
- A node is a physical machine in a datacenter, or virtual machine hosted on a cloud provider like Google Cloud Platform

Hardware Component - Cluster



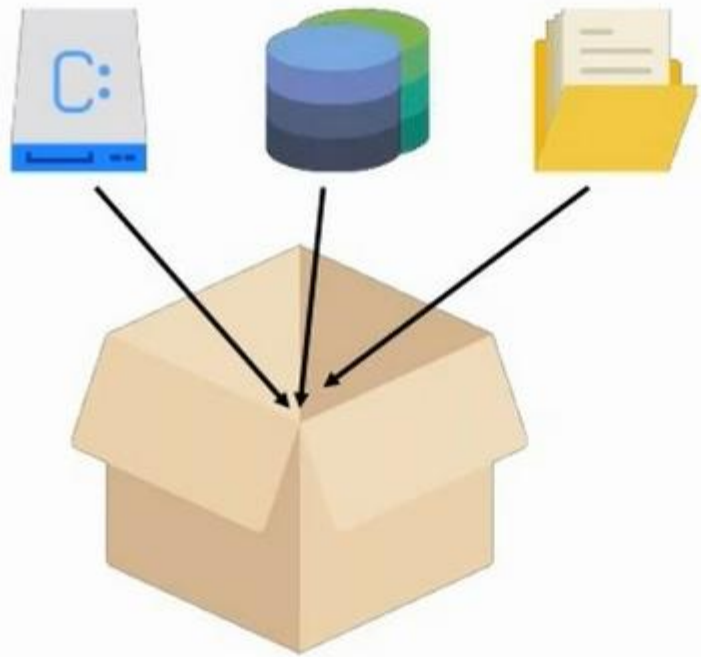
- Kubernetes does not work with individual nodes; it works with the cluster as a whole
- Nodes combine their resources to form a powerful machine known as cluster
- When a node is added or removed, the cluster shifts around the work as necessary

Hardware Component – Persistent Volumes



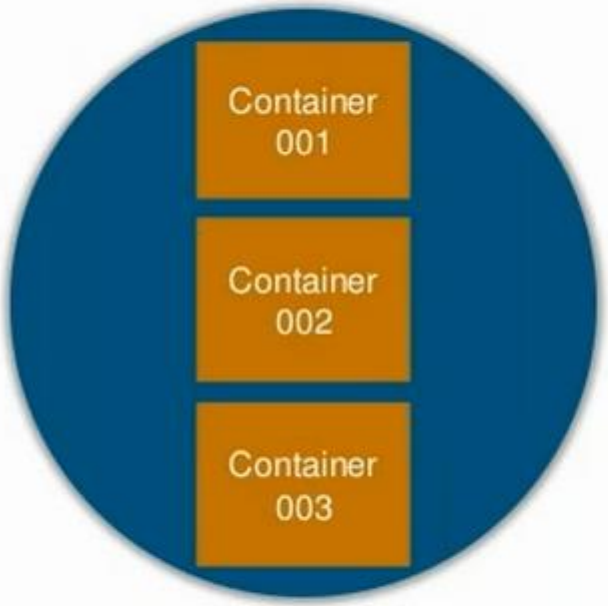
- To store data permanently, Kubernetes uses Persistent Volumes

Software Component - Container



- Containers are used everywhere as they create a self-contained environment for the execution of programs
- The programs are bundled up into a single file (known as container) and then shared on the internet
- Multiple programs are added to a single container. Limit to one process per container

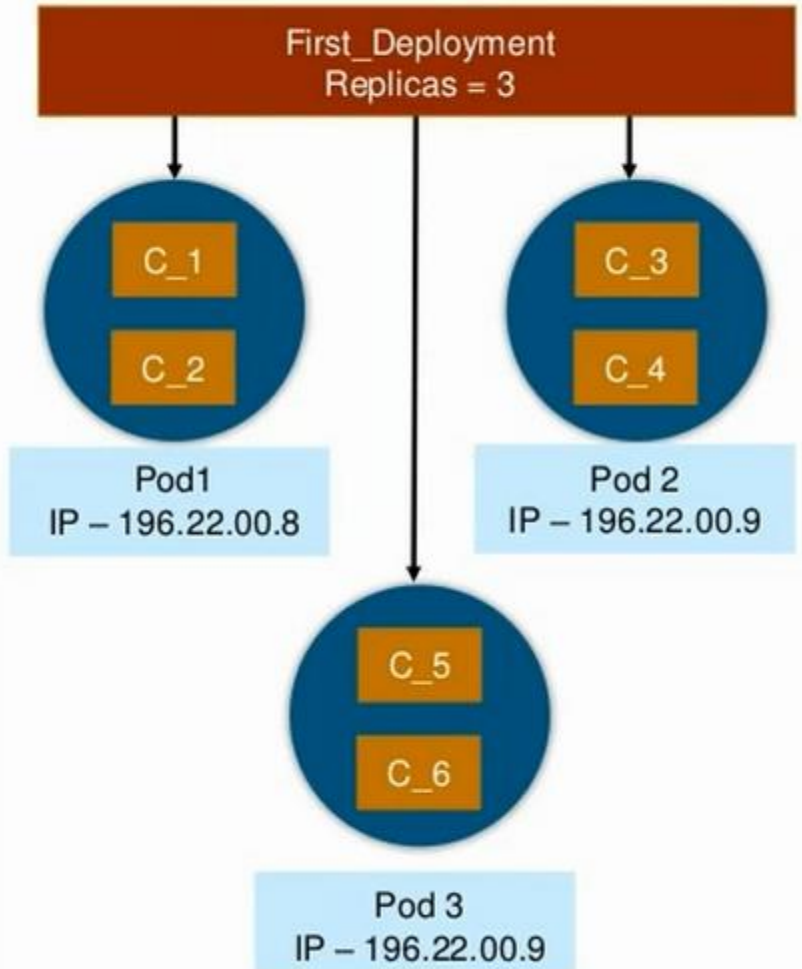
Software Component - Pods



IP – 196.22.00.8

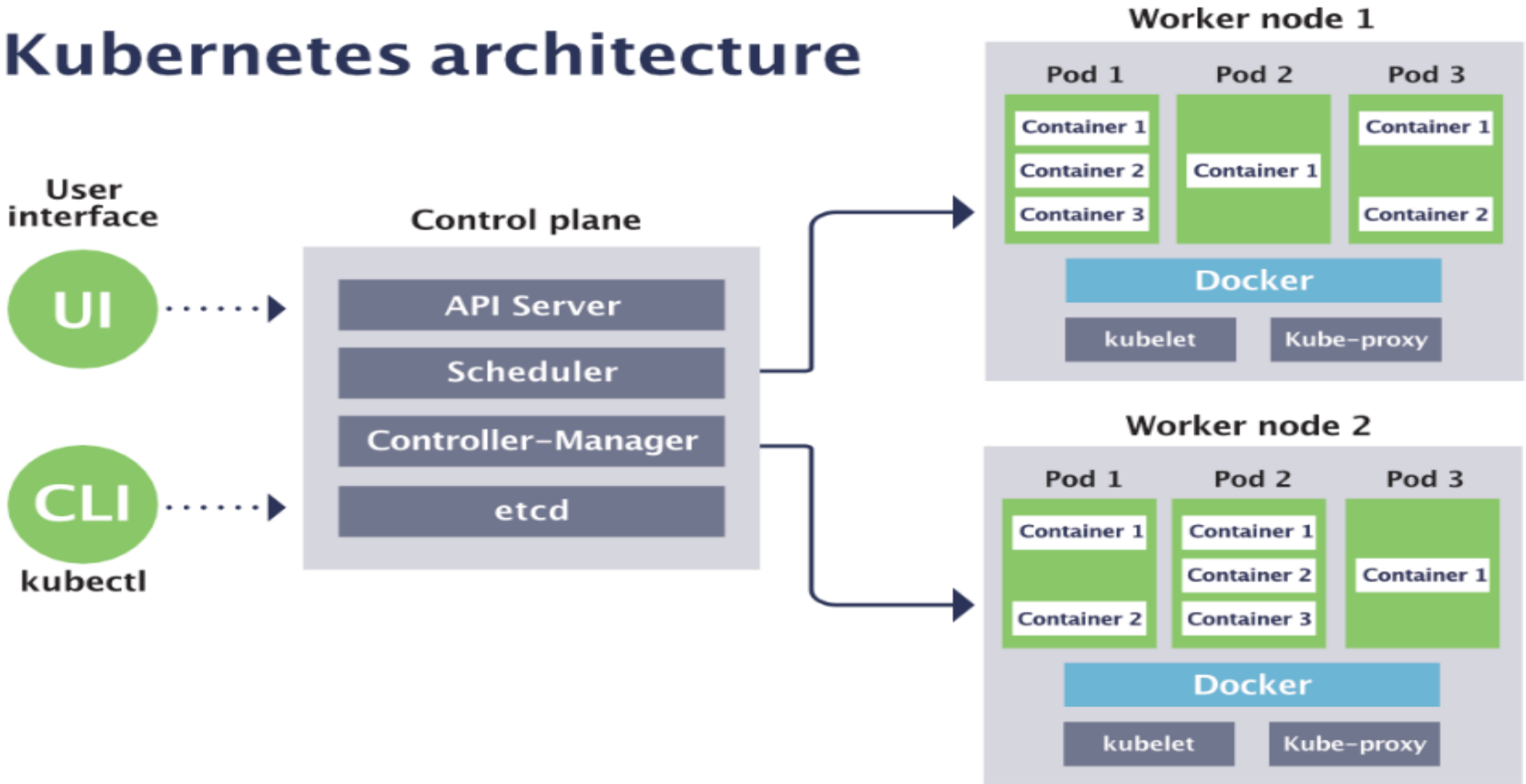
- A Pod represents a group of one or more application containers bundled up together and are highly scalable
- If a pod fails, Kubernetes automatically deploys new replicas of pod to the cluster
- Pods provide two different types of shared resources -networking and storage

Software Component - Deployment

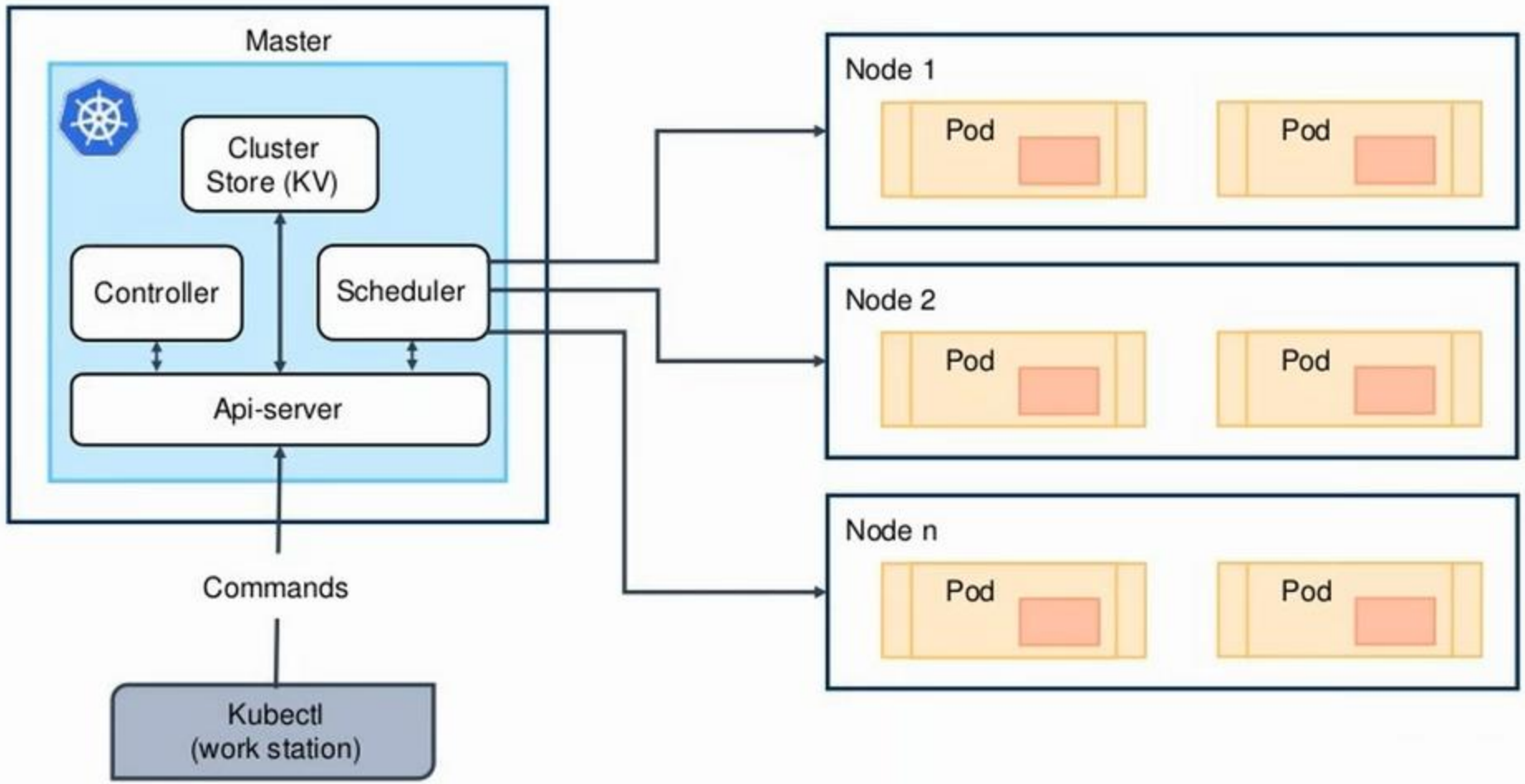


- Pods cannot be launched on a cluster directly; instead, they are managed by one more layer of abstraction- the deployment
- A deployment's fundamental purpose is to indicate how many pods are running at the same time.
- The manual management of pods is eradicated when deployment is used

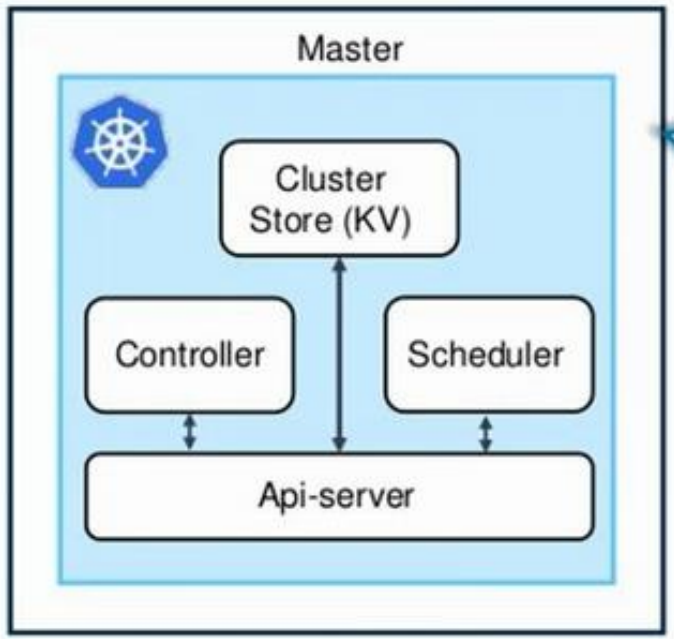
Kubernetes architecture



Kubernetes Architecture



Master Node

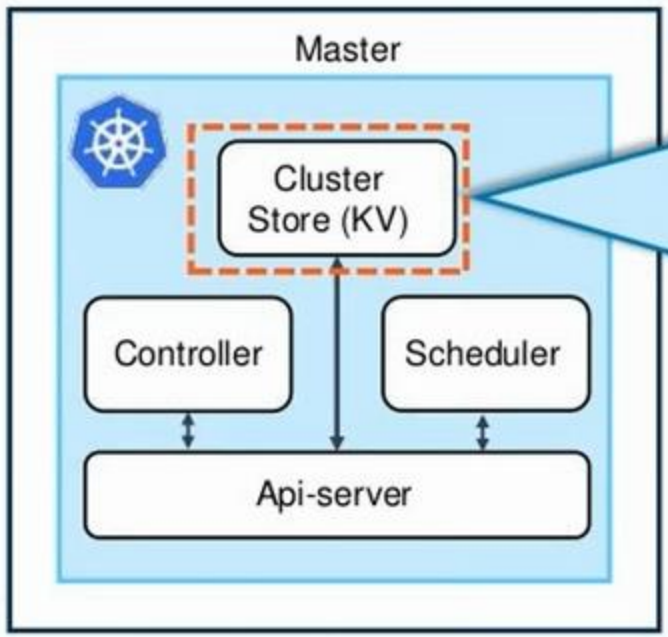


Master Node

- The master node is the most vital component responsible for Kubernetes architecture
- It is the central controlling unit of Kubernetes and manages workload and communications across the clusters
- The master node has various components, each having its process. They are

1. ETCD
2. Controller Manager
3. Scheduler
4. API Server

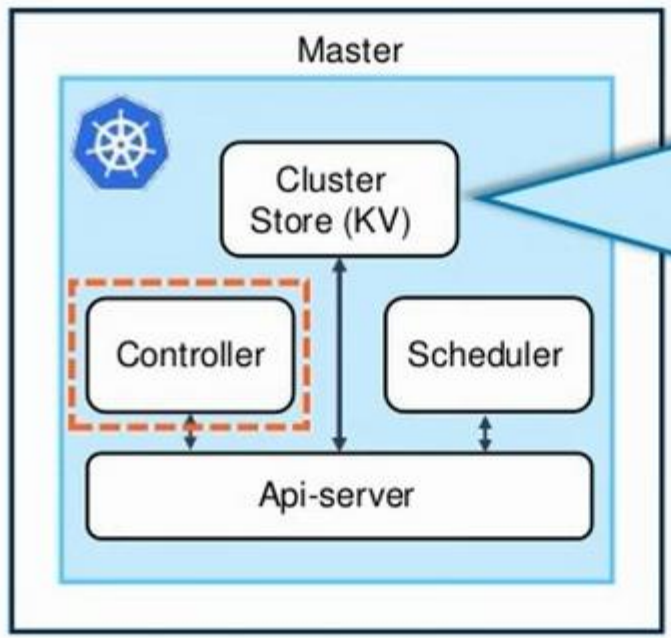
Master Node



Master Node

1. ETCD (Cluster store)
   - This component stores the configuration details and essential values
   - It communicates with all other components to receive the commands and work in order to perform an action
   - It also manages network rules and post forwarding activity

Master Node

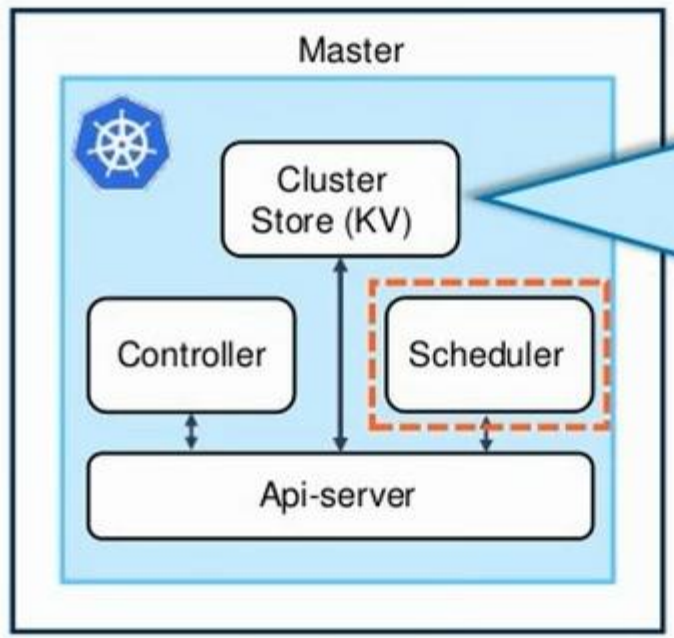


Master Node

2. Controller Manager

- It is responsible for most of the controllers and performs a task
- It is a daemon which runs in a continuous loop and is responsible for collecting and sending information to API server
- The key controllers handle nodes, endpoints, etc.

Master Node

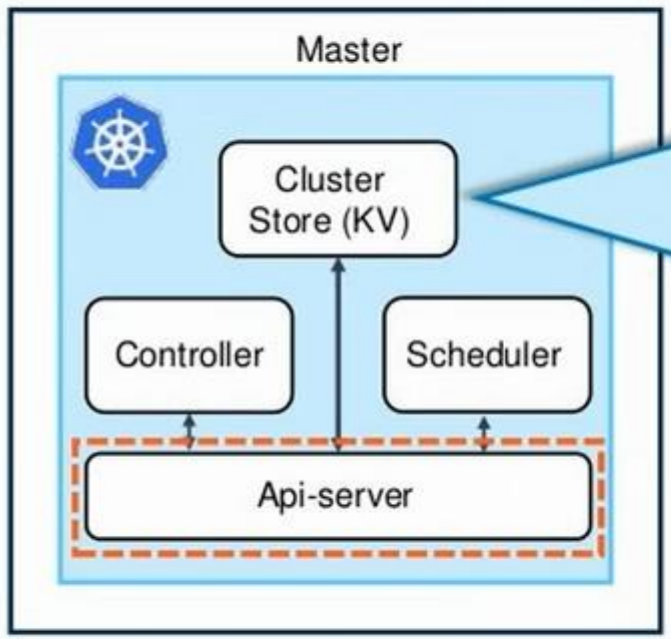


Master Node

3. Scheduler

- It is one of the key components of the master node associated with the distribution of workload
- The scheduler is responsible for workload utilization and allocating pod to a new node
- The scheduler should have an idea of the total resources available as well as resources allocated to existing workloads on each node

Master Node

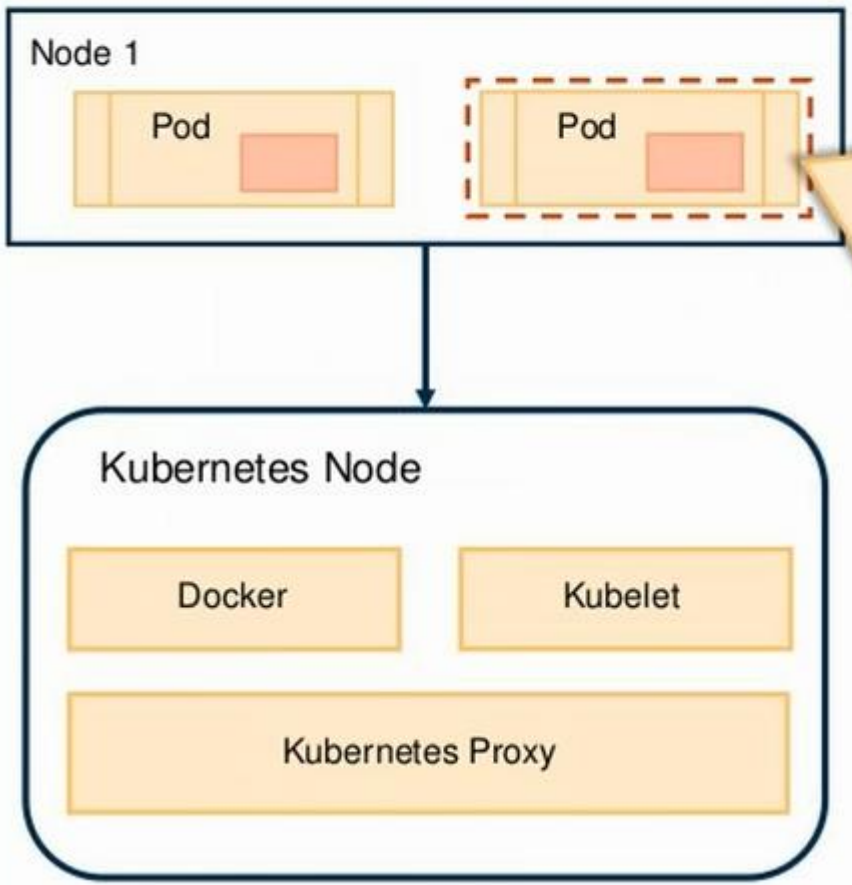


Master Node

4. API Server

- Kubernetes uses API Server to perform all operations on the cluster
- It is a central management entity that receives all REST requests for modifications, serving as a frontend to the cluster
- It implements an interface which means different tools and libraries can communicate effectively

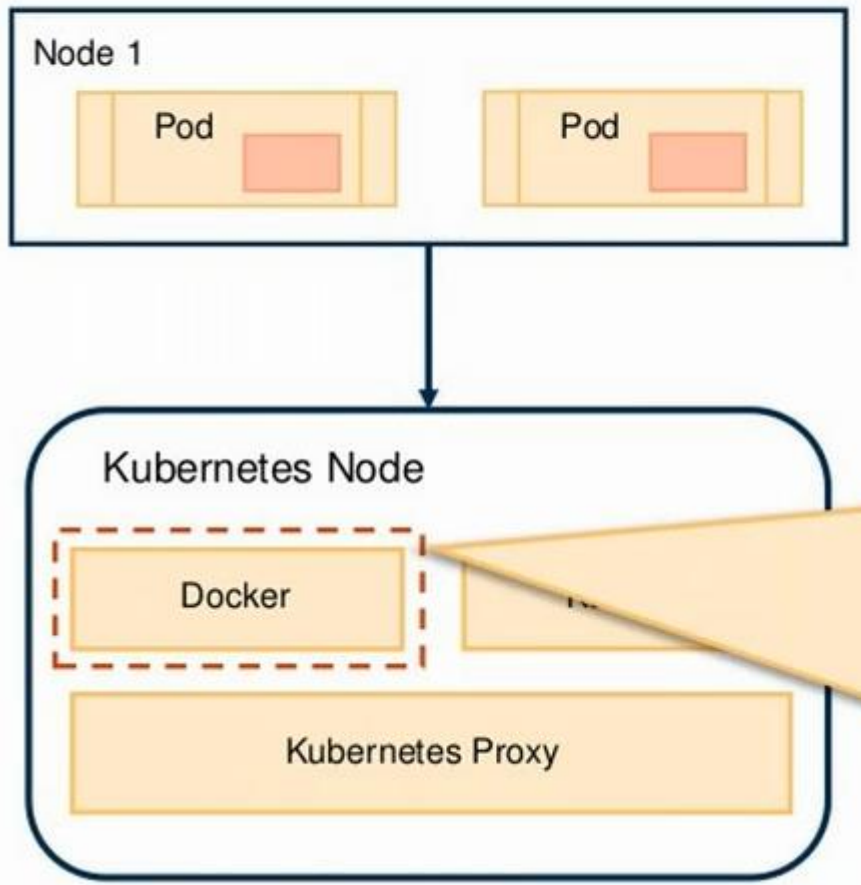
Slave Node



Slave Node

1. Pod
   - A pod is one or more containers controlled as a single application
   - It encapsulates application containers, storage resources, a unique network ID and other configuration on how to run the containers

Slave Node

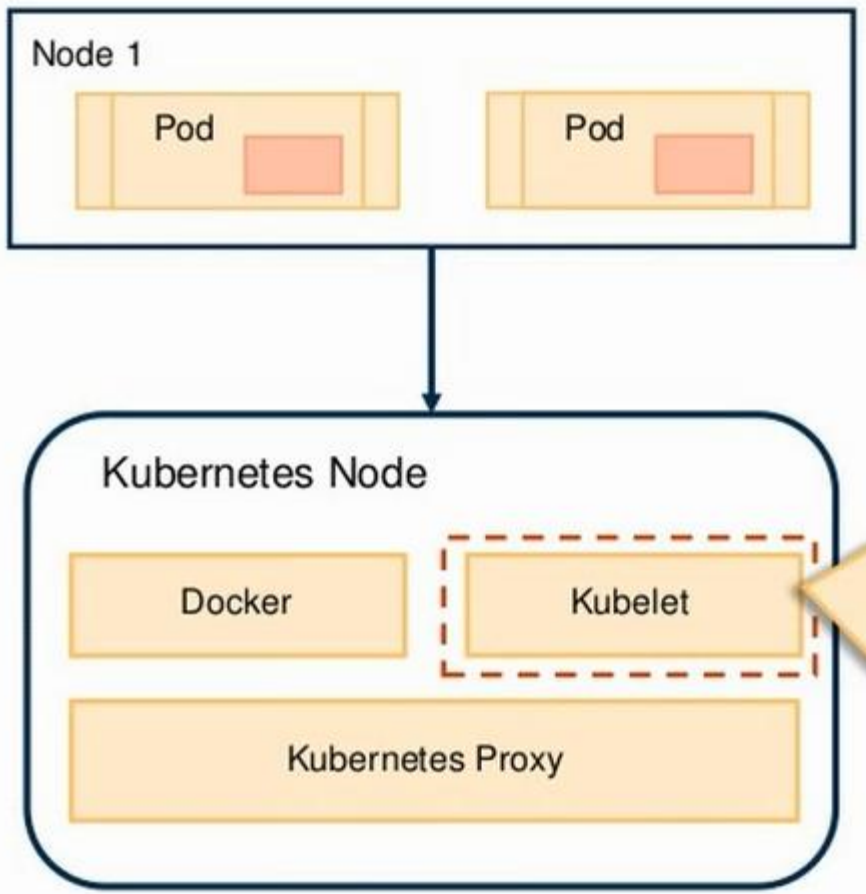


Slave Node

2. Docker

- One of the basic requirement of nodes is Docker
- It helps in running the applications in an isolated but lightweight operating environment. It runs the configured pods
- It is responsible for pulling down and running container from Docker images

Slave Node

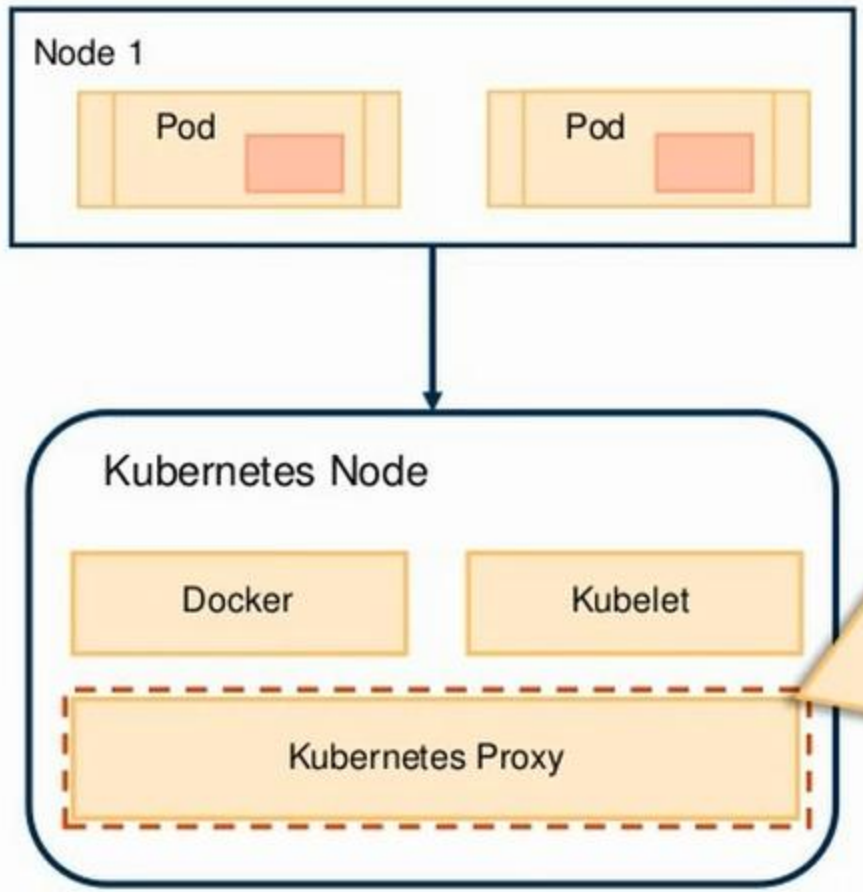


Slave Node

3. Kubelet

- It is responsible for managing pods and their containers
- It deals with pods specifications which are defined in YAML or JSON format
- It takes the pod specifications and checks whether the pods are running healthy or not

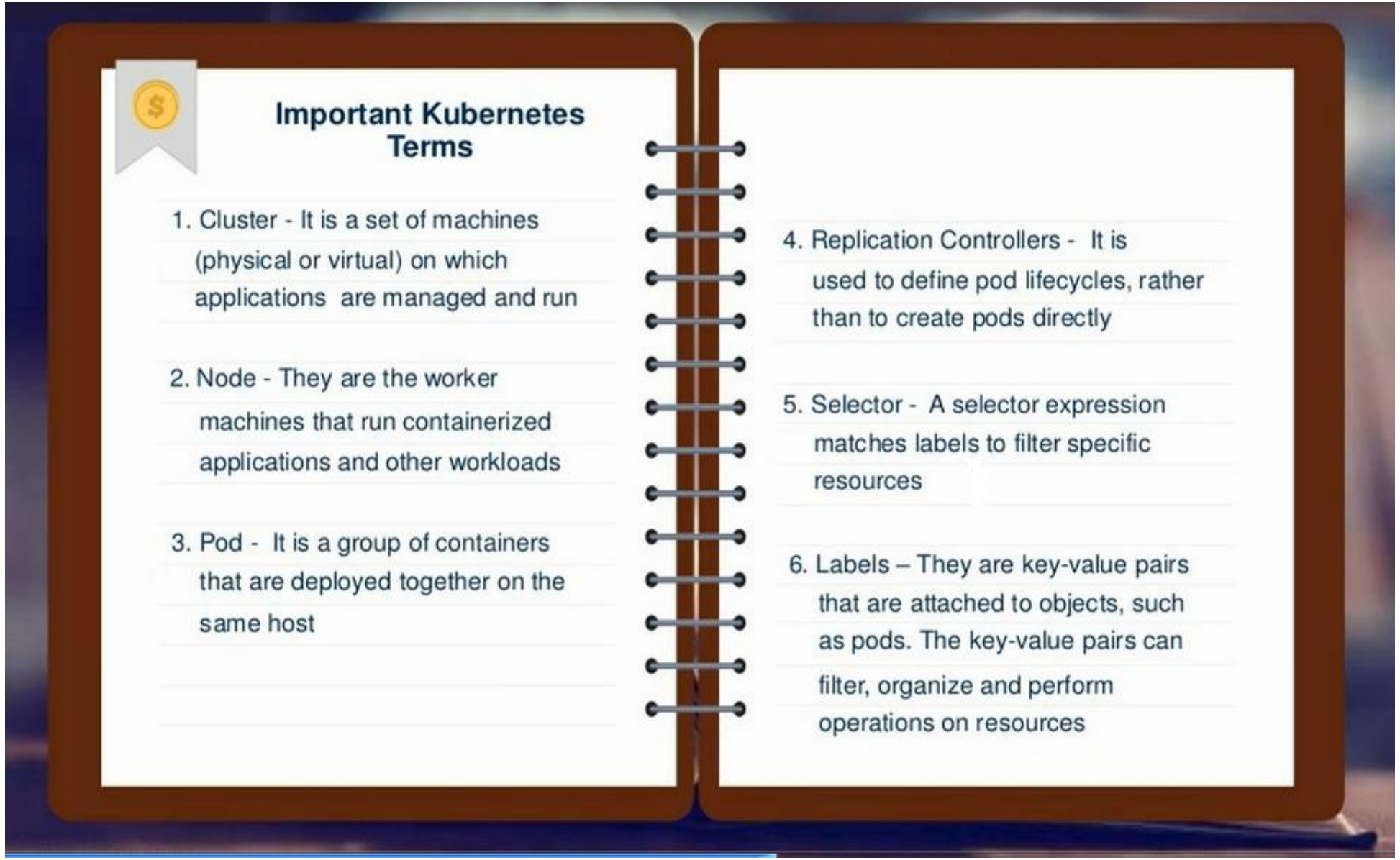
Slave Node

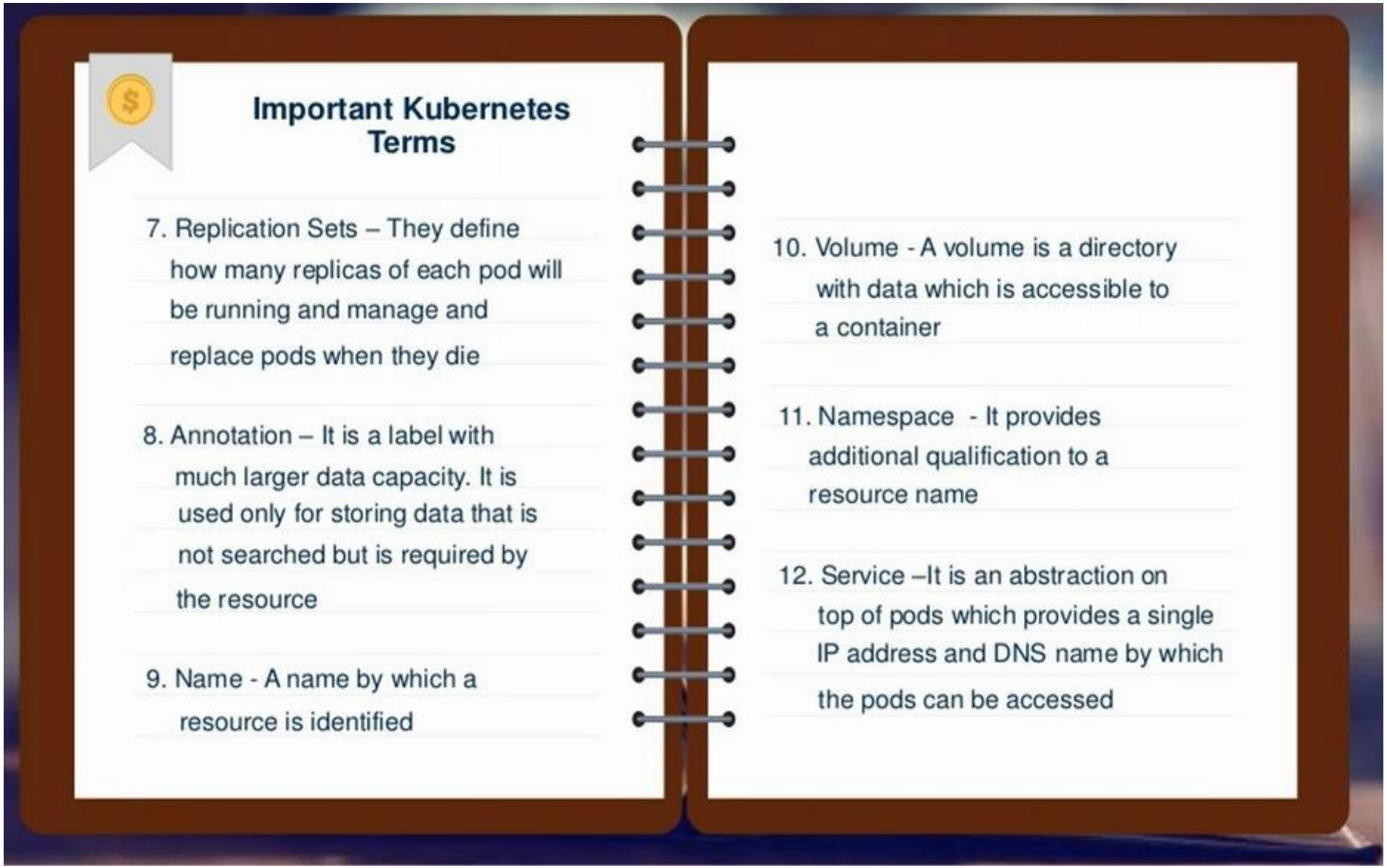


Slave Node

4. Kubernetes Proxy

- IT is a proxy service which runs on each node and helps in making services available to the external host
- Every node in the cluster runs a simple network proxy, and Kube-proxy routes request to the correct container in a node
- It performs primitive load balancing and manages pods on node, volumes, secrets, creating new containers' health checkup, etc.


Important Kubernetes Terms
1. Cluster - It is a set of machines (physical or virtual) on which applications are managed and run
2. Node - They are the worker machines that run containerized applications and other workloads
3. Pod - It is a group of containers that are deployed together on the same host
4. Replication Controllers - It is used to define pod lifecycles, rather than to create pods directly
5. Selector - A selector expression matches labels to filter specific resources
6. Labels – They are key-value pairs that are attached to objects, such as pods. The key-value pairs can filter, organize and perform operations on resources


Important Kubernetes Terms
7. Replication Sets – They define how many replicas of each pod will be running and manage and replace pods when they die
8. Annotation – It is a label with much larger data capacity. It is used only for storing data that is not searched but is required by the resource
9. Name - A name by which a resource is identified
10. Volume - A volume is a directory with data which is accessible to a container
11. Namespace - It provides additional qualification to a resource name
12. Service –It is an abstraction on top of pods which provides a single IP address and DNS name by which the pods can be accessed